



P2M2HBVE MODUFLEX ETHERNET/IP

ADD-ON INSTRUCTION FOR ROCKWELL PLC

QUICK START GUIDE

**PREFACE**

This Quick Start Guide (QSG) is designed to help integrate Parker Hannifin’s P2M Ethernet/IP valve manifold into an Allen-Bradley (AB) PLC environment.

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\_P2M2HBVE\_EIP\_PRM\_Rx” AOI facilitates the call-up of the acyclic service data.

The “AB\_P2M2HBVL\_EIP\_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 ESD configuration file, this QSG, a sample RSLogix5000 file and the full P2M manual here:

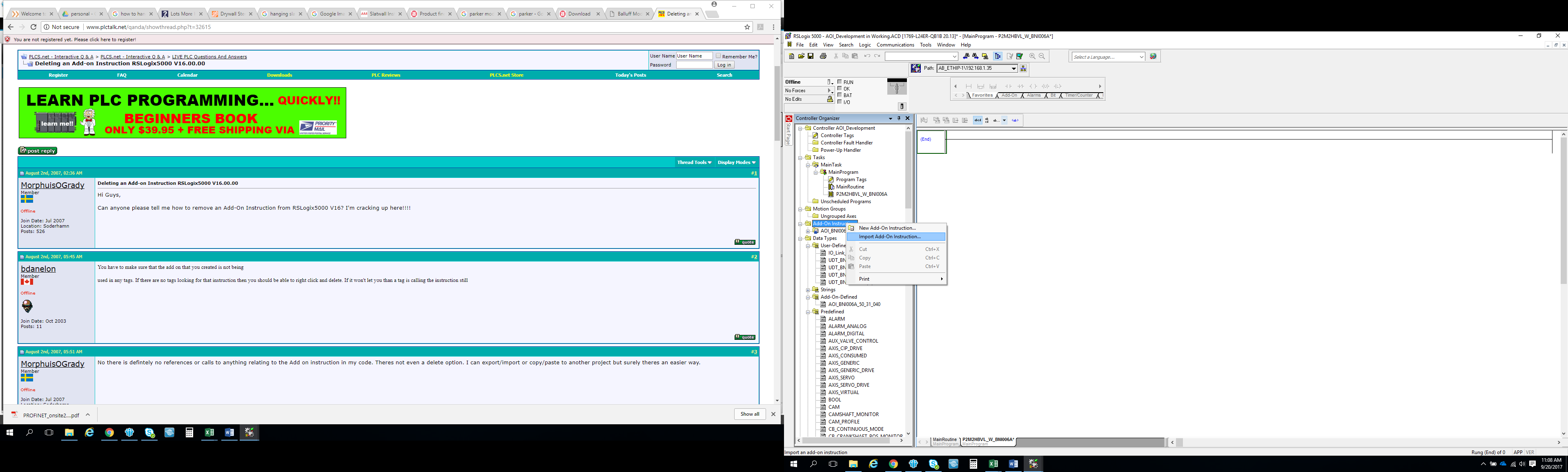
[*http://www.parker.com/pdn/P2M\_IE*](http://www.parker.com/pde/P2M_IE)

**Process Data Add-On Instruction**

The “AB\_P2M2HBVL\_EIP\_PD\_Rx” AOI simplifies the usage of Parker P2M Ethernet/IP devices with Allen-Bradley CompactLogix, ControlLogix and GuardLogix PLCs when connected via Ethernet/IP. Data is mapped to user-friendly control and diagnostic tags on the PLC side.

**IMPORTING THE INSTRUCTION**

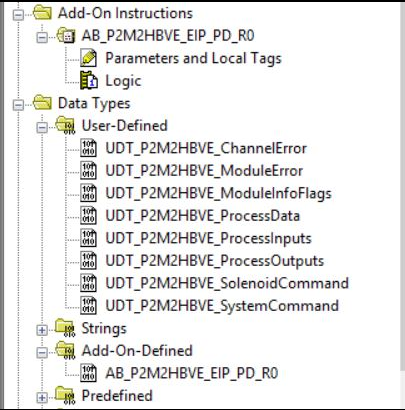
1. Right click Add-On Instruction in Controller Organizer and select “Import Add-On Instruction…”



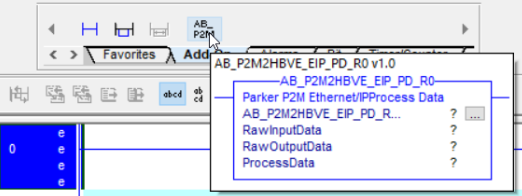
1. Select the “AB\_P2M2HBVL\_EIP\_PD\_Rx” where \_Rx is the revision of AOI.

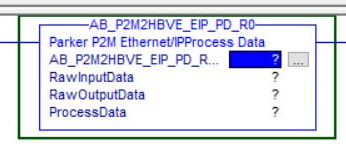


Choose OK on Import Configuration Window and you should then see the new AOI instance along with User-Defined and Add-On Defined data types created in the controller organizer.

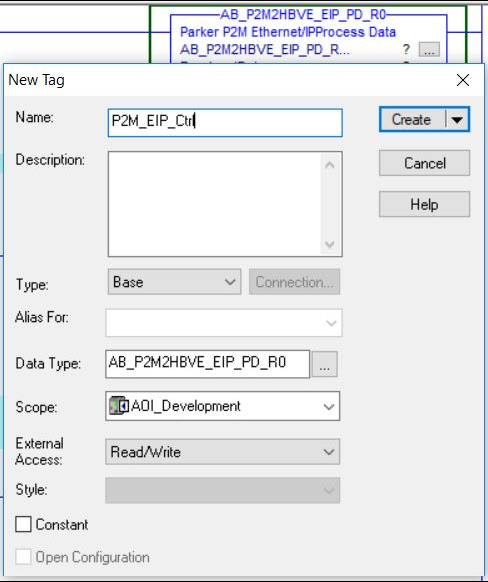
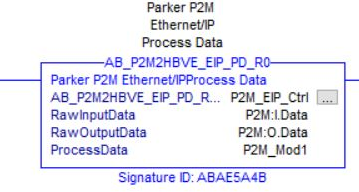


1. Add instance of instruction to an empty rung of ladder by clicking on the AB\_P2M under the Add-On tab in the top toolbar. The instruction will drop onto the selected rung.





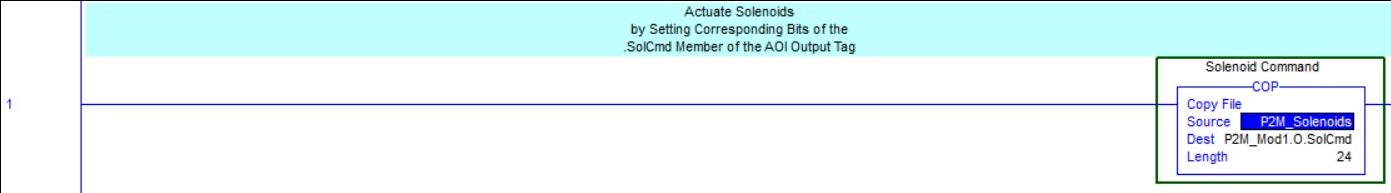
1. Assign an instance name for the AOI and create other tags necessary for operation. Right click on the question marks and select “New Tag”. Note that the name must be unique for each tag and each instance of the P2M AOI. The scope and data type fields will auto-populate with the correct values, so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

**USING THE INSTRUCTION**

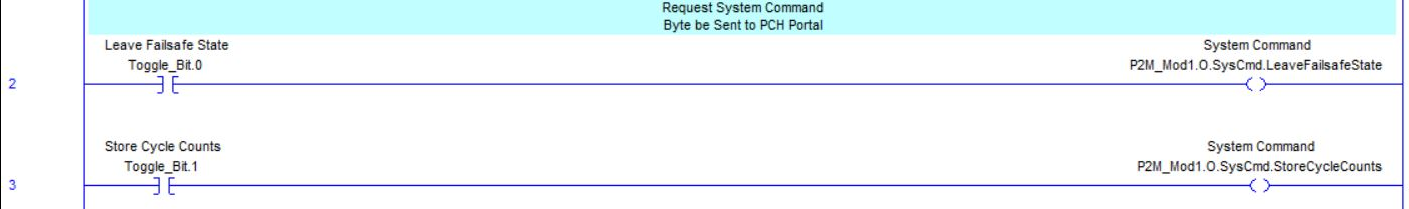
It is important to note the difference between cyclic and acyclic data. Process Data (cyclic) is updated without a request; whereas Parameter Data (acyclic) requires the program to toggle a bit to pull data. Cyclic data includes input module status, valve output control and system command. This means that P2M\_Mod1.I.xxx and P2M\_Mod1.O.xxx are live tags with real data just by calling the AOI. See appendix for all data points available. See ladder logic examples below:

* 1. **Toggling Solenoid Valves (Cyclic)**



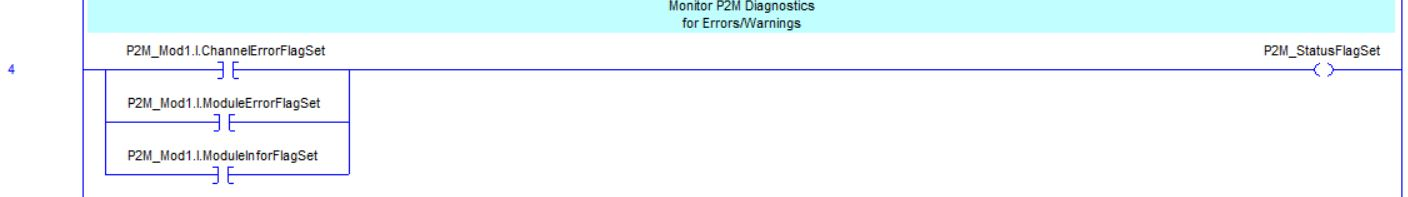


* 1. **Writing System Commands (Cyclic)**



Setting “SysCmd.LeaveFailsafeState” output element is equivalent to writing 0x01 to the first byte of output process data.  
Setting “SysCmd.StoreCycleCounts” output element is equivalent to writing 0x02 to the first byte of output process data.

* 1. **Monitoring Status Bits (Cyclic)**



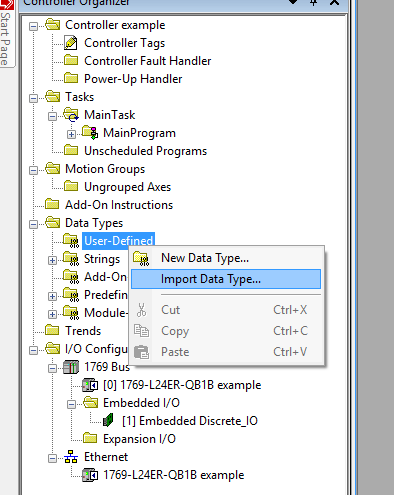
If any of the status bits are set, further information can be attained by evaluating specific diagnostic bits.



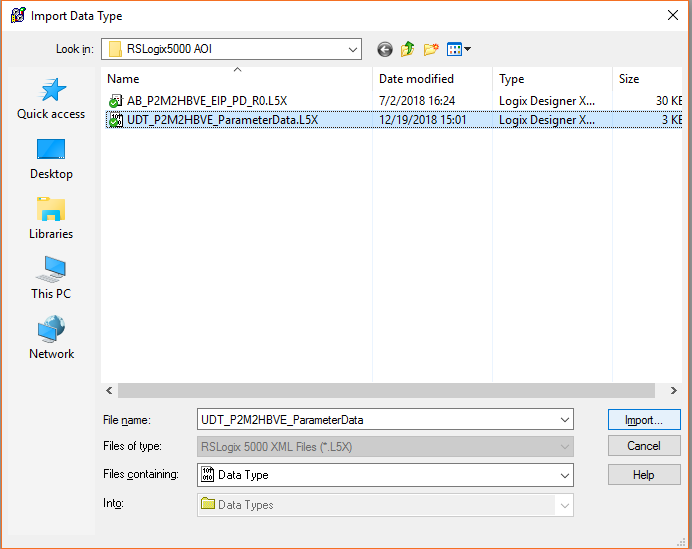
**Parameter Data Example Code**

**IMPORTING User-Defined Data Type**

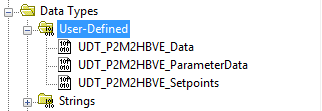
1. Right click User-Defined in Controller Organizer and select “Import Data Type…”



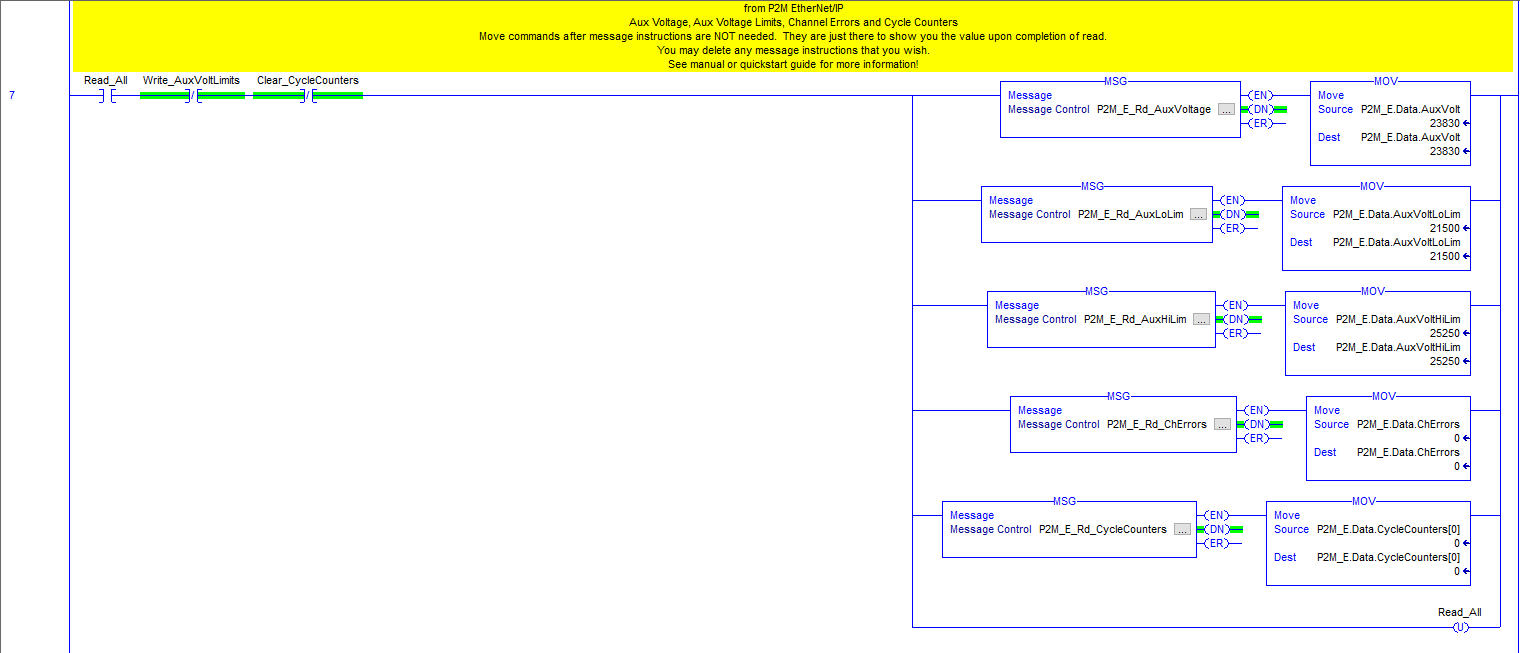
1. Select the UDT\_P2M2HBVE\_ParameterData.L5X and choose “Import…”



Choose OK on Import Configuration Window and you should then see the new User-Defined data types created in the controller organizer.



1. Parameter Code Examples
   1. Reading all parameter data, see below for logic example as well as message configuration for each parameter. You may rename the Message Control tags to match your tag naming convention.



4

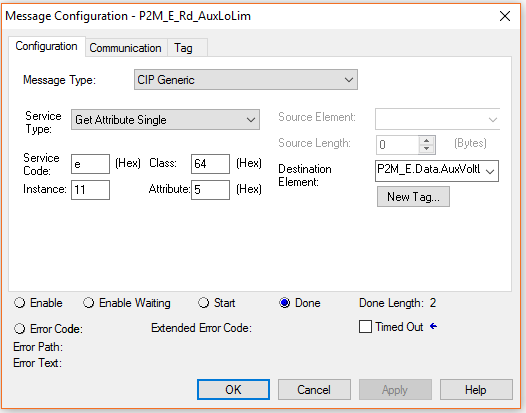
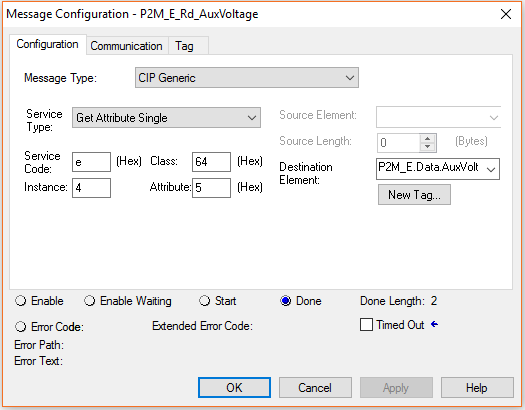
5

3

2

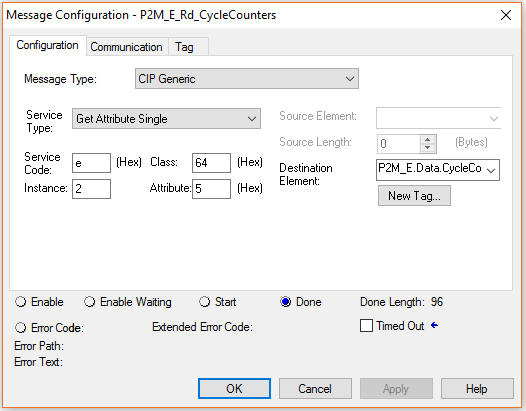
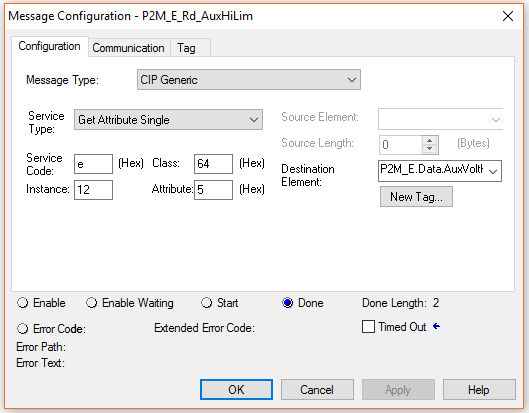
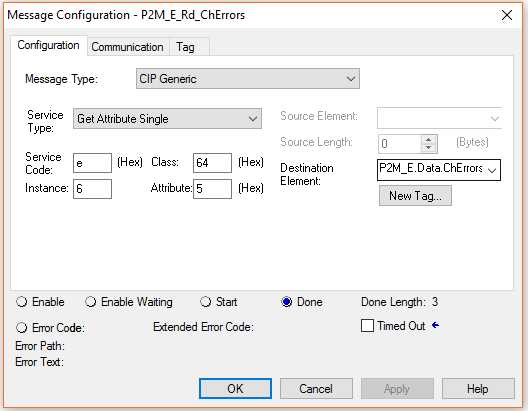
1

Access the configuration window by clicking the ellipsis button in the MSG instruction. Please note the following parameters should **ALWAYS** be as follows: Service Code = e, Class = 64, Attribute = 5. Instance and destination element will change depending on parameter. Read instructions should be set to CIP Generic – Get Attribute Single.



2

1



5

4

3

2

1

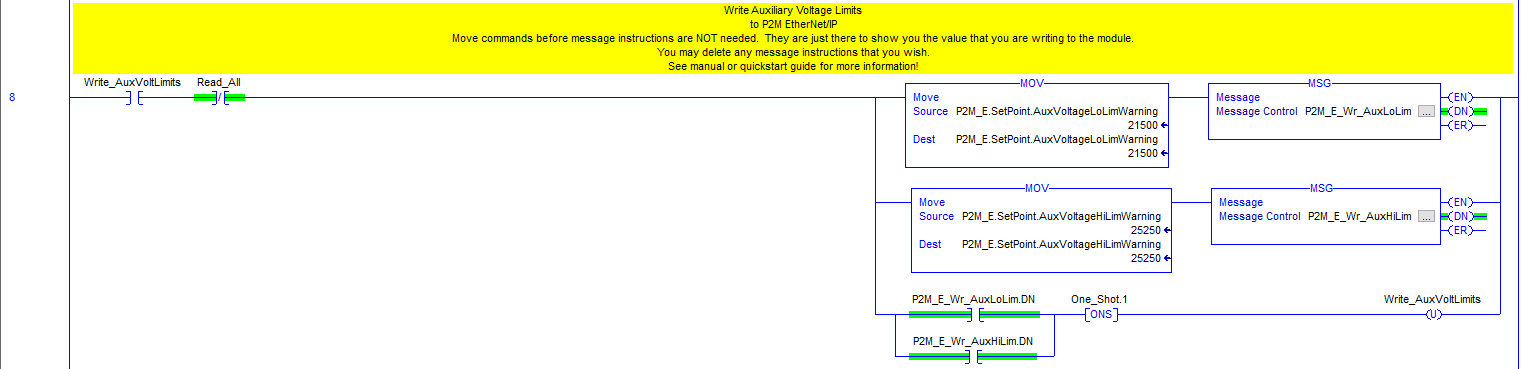
3

4

5

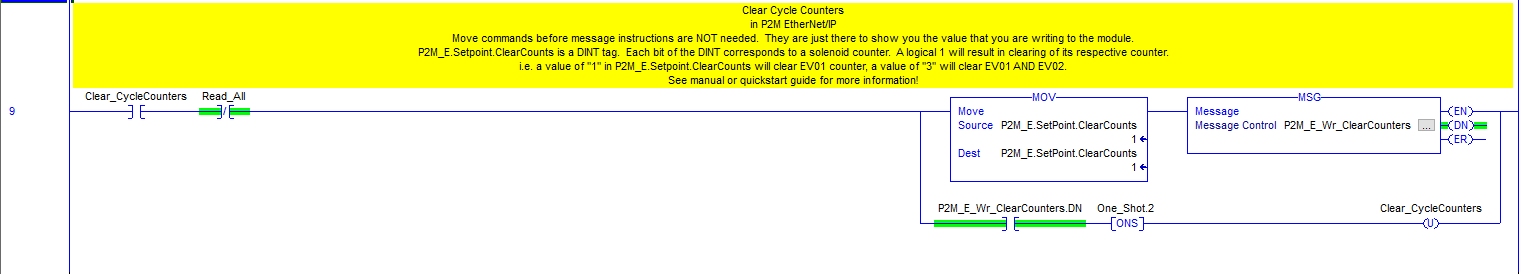
|  |  |  |
| --- | --- | --- |
| Description | Instance | Destination Element |
| Read Auxiliary Voltage | 4 | P2M\_E.Data.AuxVolt |
| Read Auxiliary Voltage Low Limit Warning | 11 | P2M\_E.Data.AuxVoltLoLim |
| Read Auxiliary Voltage High Limit Warning | 12 | P2M\_E.Data.AuxVoltHiLim |
| Read Channel Errors | 6 | P2M\_E.Data.ChErrors |
| Read Cycle Counters | 2 | P2M\_E.Data.CycleCounters |

* 1. Writing to Auxiliary Voltage Low and High Limits and clearing cycle counters example



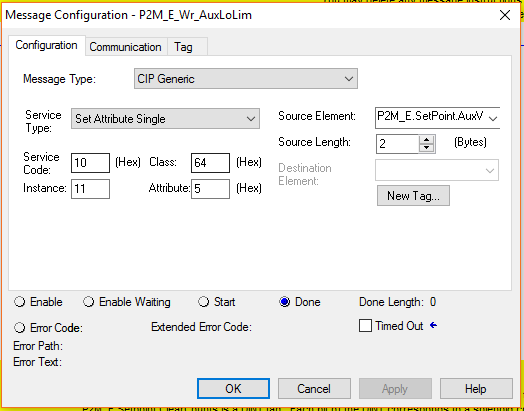
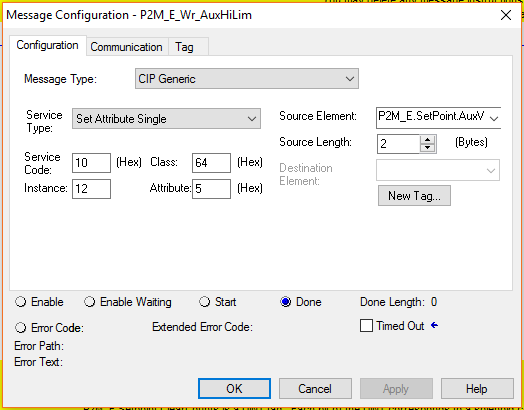
7

6



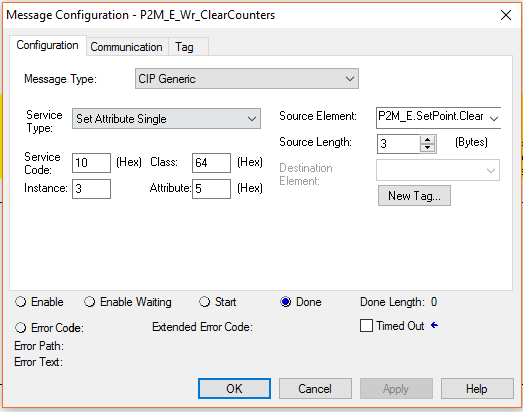
8

Access the configuration window by clicking the ellipsis button in the MSG instruction. Please note the following parameters should **ALWAYS** be as follows: Service Code = 10, Class = 64, Attribute = 5. Instance, Source Element, and Source Length will change depending on parameter. Write instructions should be set to CIP Generic – Set Attribute Single.



6

7



8

8

6

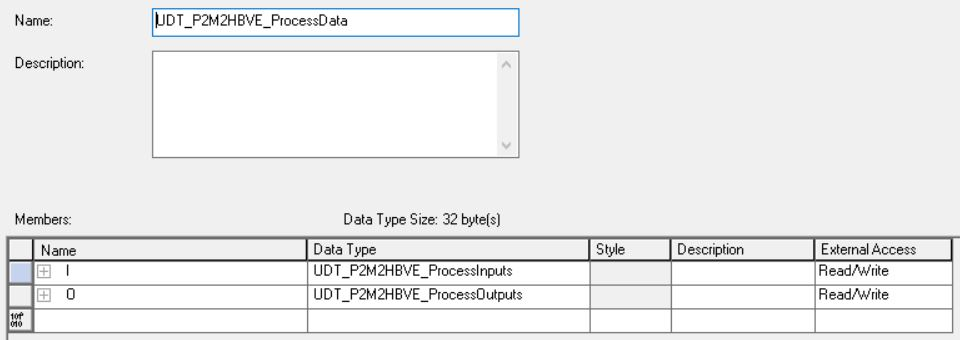
7

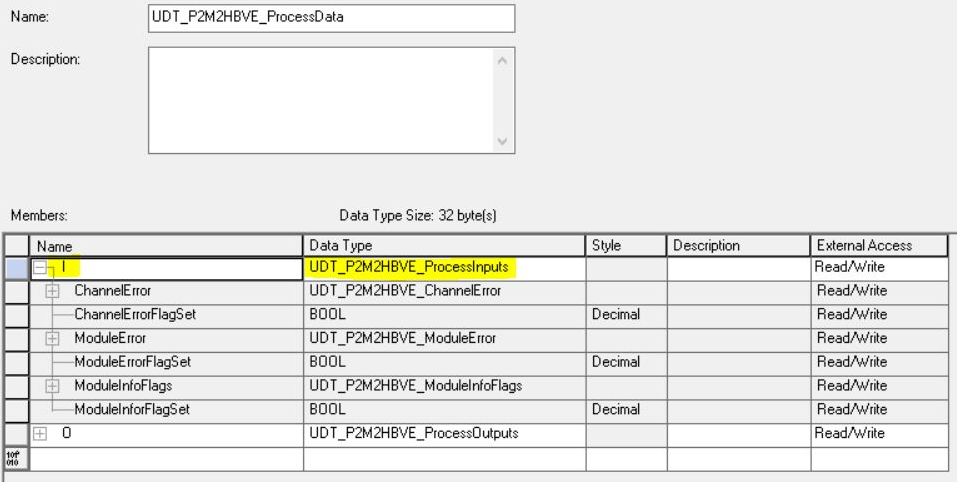
|  |  |  |  |
| --- | --- | --- | --- |
| Description | Instance | Source Element | Source Length  (bytes) |
| Write Auxiliary Voltage Low Limit Warning | 11 | P2M\_E.SetPoint.AuxVoltageLoLimWarning | 2 |
| Write Auxiliary Voltage High Limit Warning | 12 | P2M\_E.SetPoint.AuxVoltageHiLimWarning | 2 |
| Clear Cycle Counters | 3 | P2M\_E.SetPoint.ClearCounts | 3 |

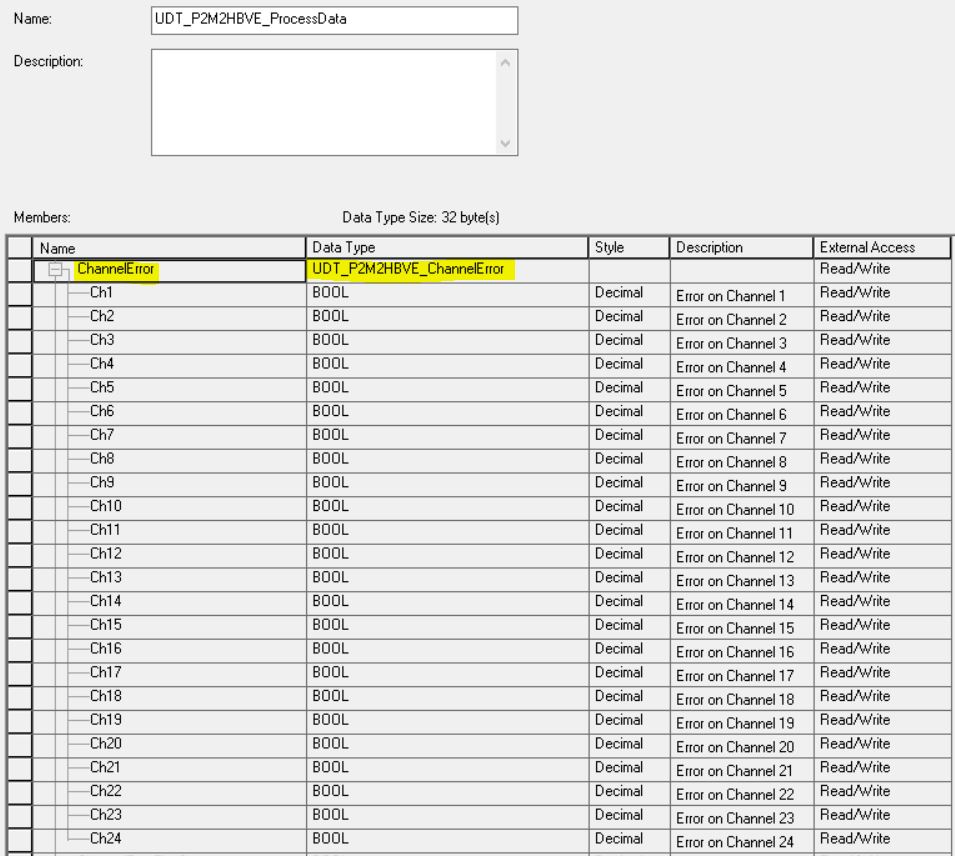
**APPENDIX**

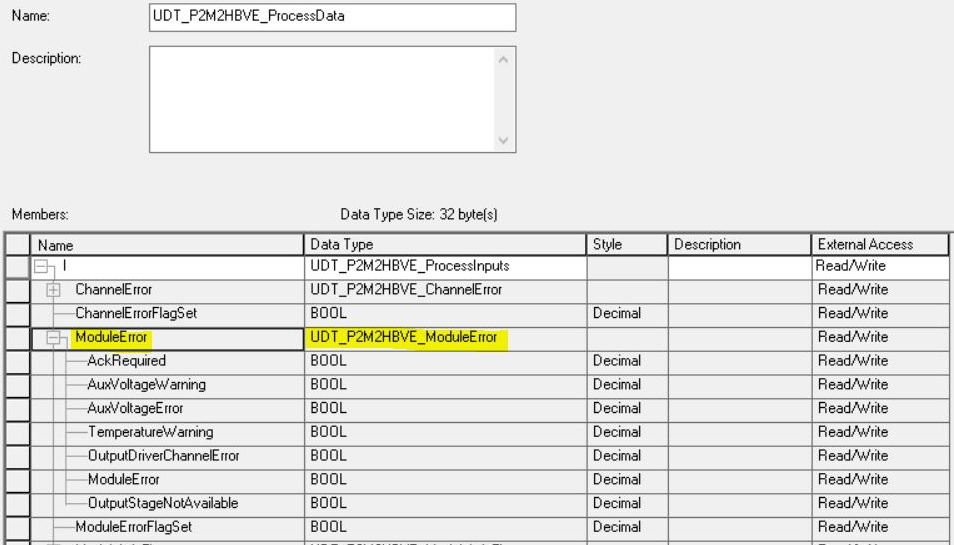
**Process Data Structures**

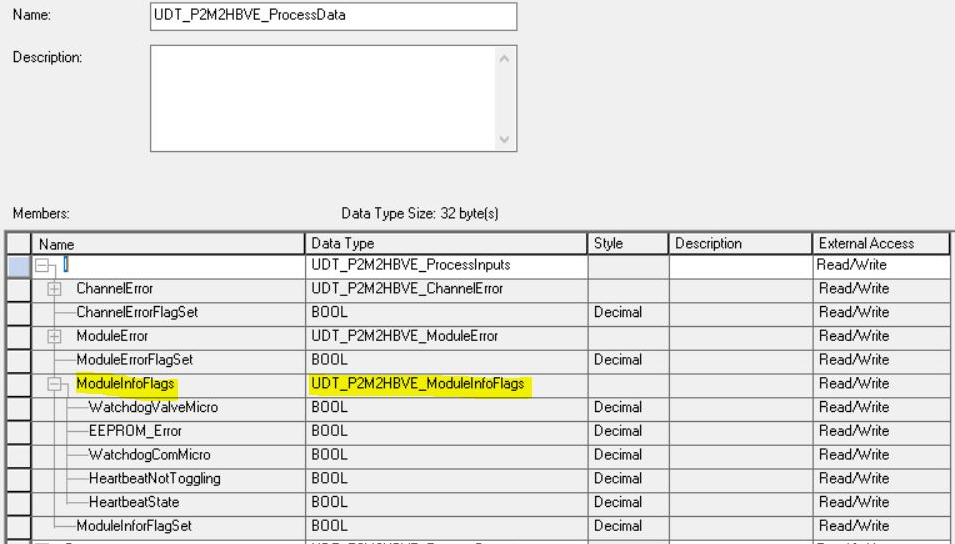
User Defined / Add-On Defined Data Structures utilized by AOI “AB\_P2M2HBVL\_EIP\_PD\_Rx”

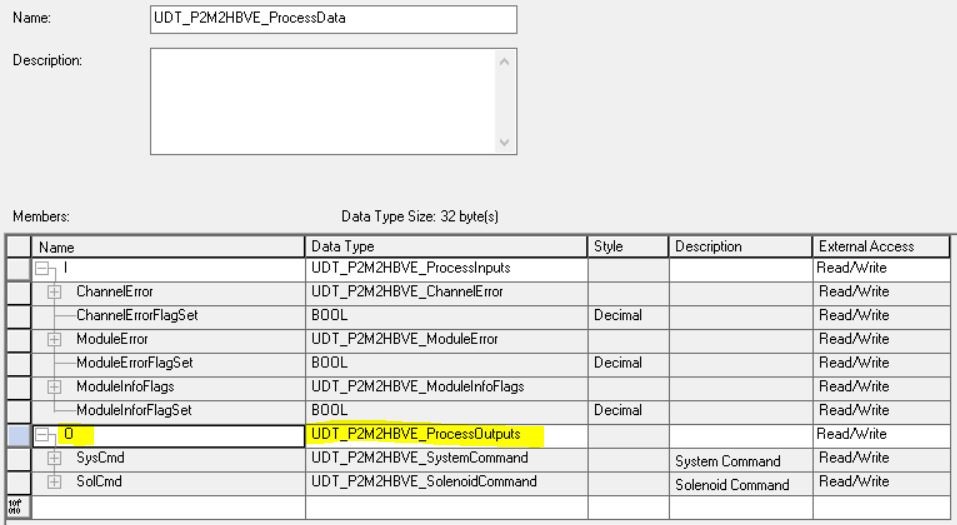


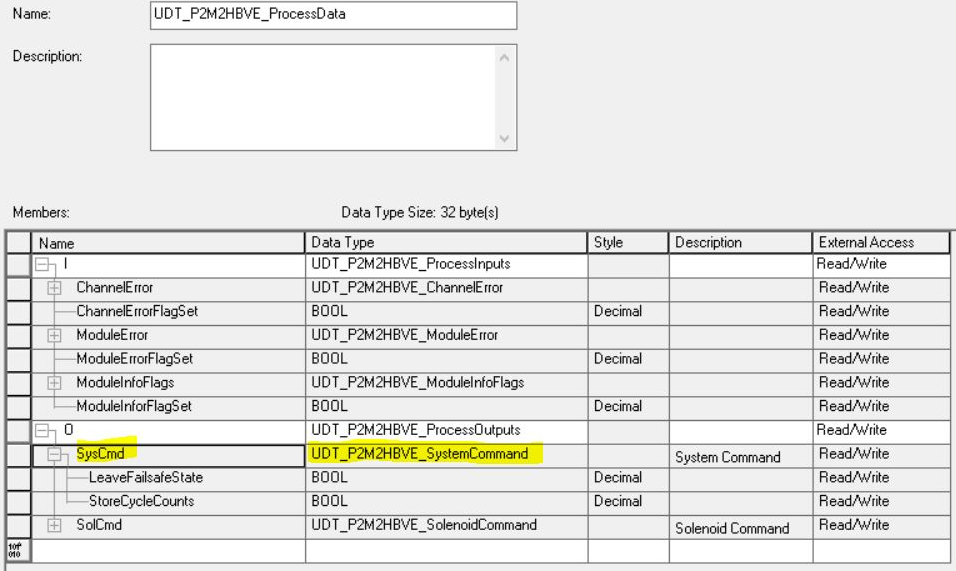


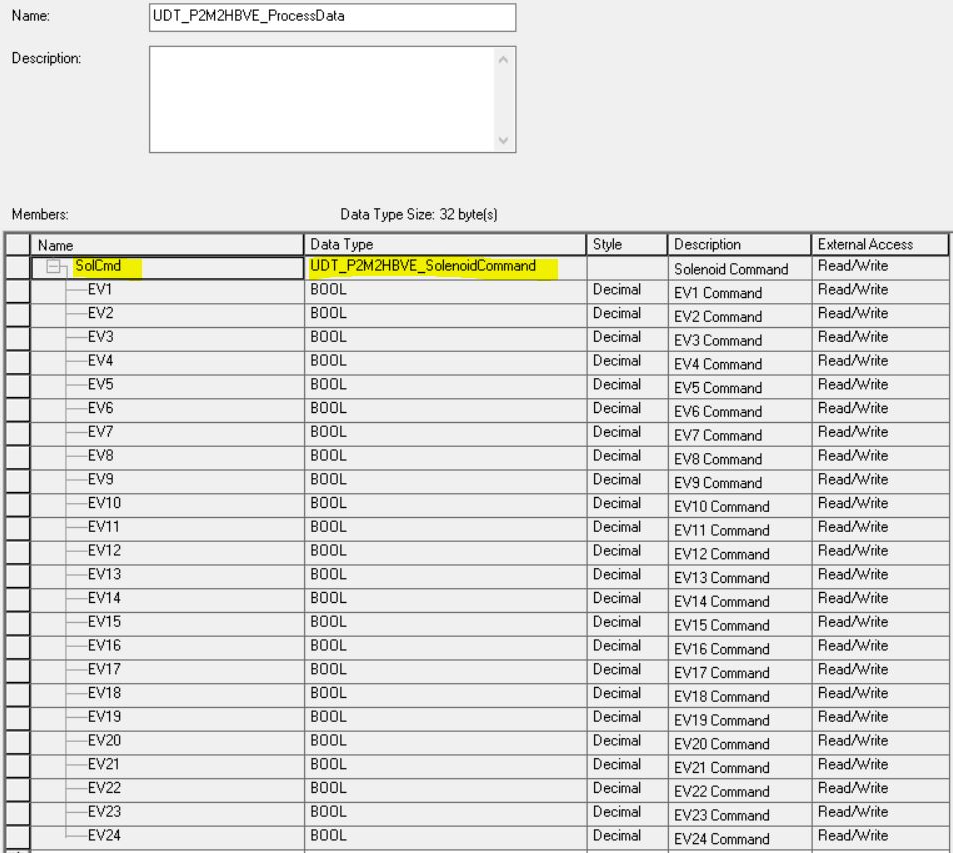












**Parameter Data Structures**

User Defined / Add-On Defined Data Structures utilized in example code

