S15C-MGN Process Data AOI Guide, v4
September 12th, 2023

This document covers the installation and use of an Add-On Instruction (AOI) for the Logix Designer software package from Rockwell Automation. This AOI handles cyclic IO-Link Process Data In and Process Data Out to and from a Banner S15C-MGN device via an IO-Link Master to an Allen-Bradley PLC. The AOI covers parsing and display of the S15C-MGN Process Data In and Process Data Out. The AOI has three User Defined Tag data types.

Components

Banner_S15C_MGN_PD_v4_AOI.L5X

UDT Packaged with the AOI

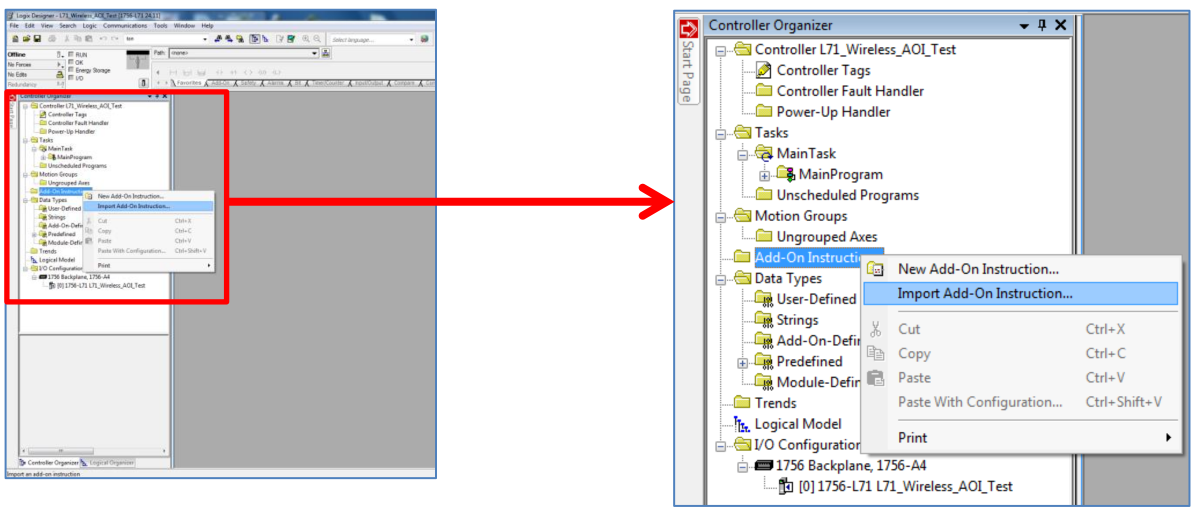
Banner_S15C_MGN_PD_v4
Banner_S15C_MGN_Read_DS_v4
Banner_S15C_MGN_Write_DS_v4

Contents

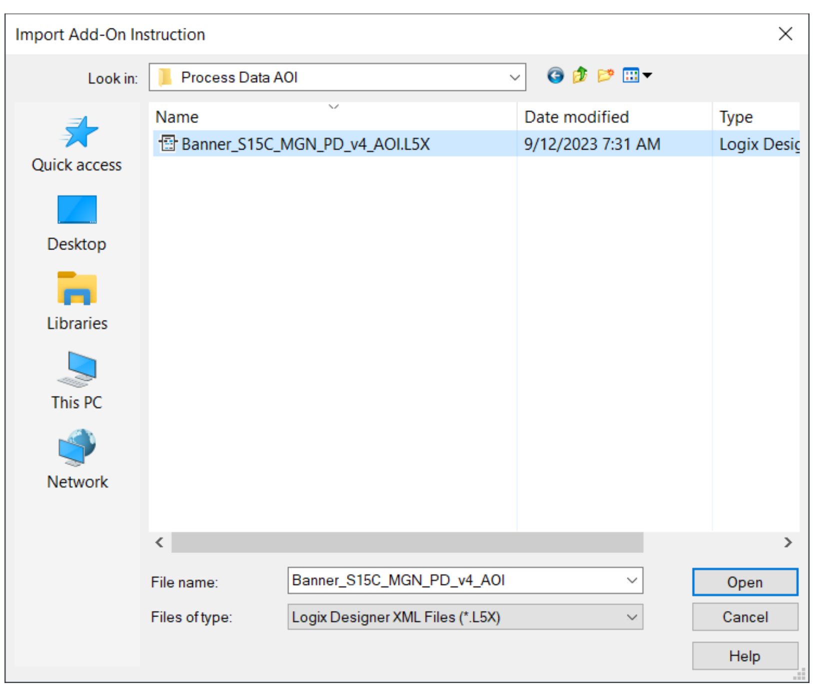
1. Installation Process

This section describes how to install the AOI in Logix Designer software.

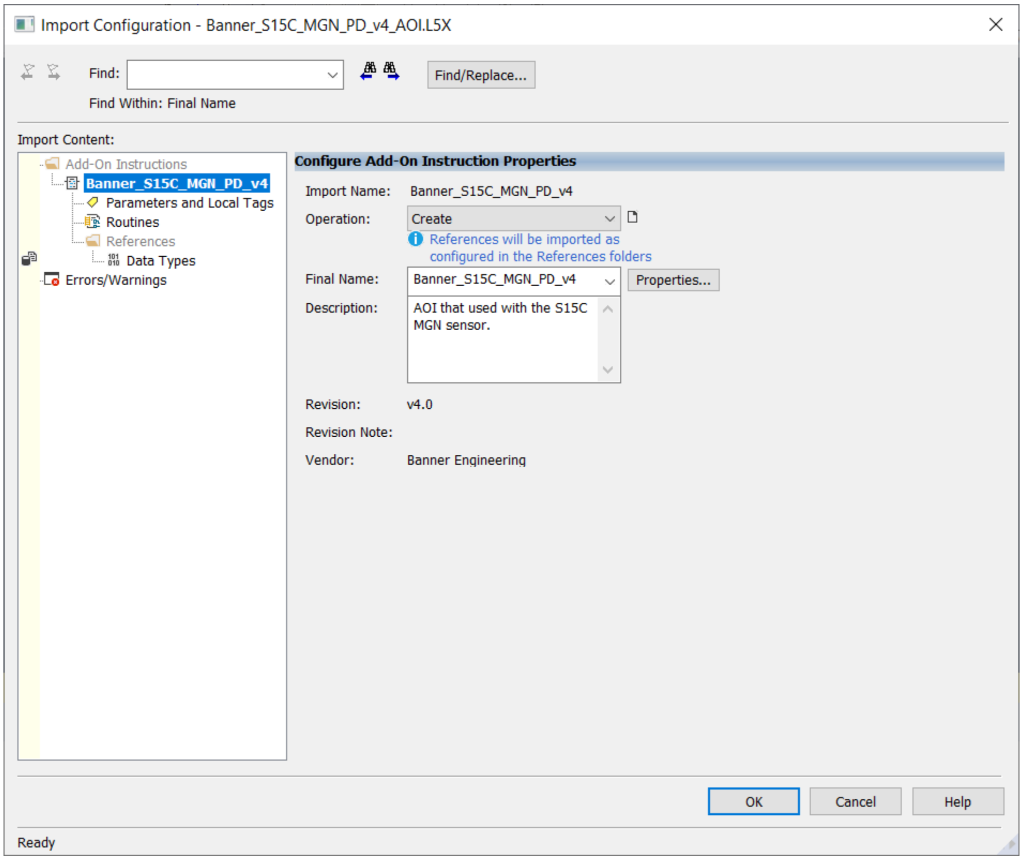
1. Open a project.
2. In the Controller Organizer window, right-click on the Add-On Instruction folder. Select the Import Add-On Instruction option.

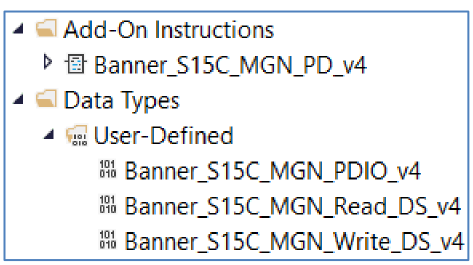


3. Navigate to the correct file location and select the AOI to be installed. In this example the "Banner_S15C_MGN_PD_v4_AOI.L5X" file will be selected. Click the Open button.



4. The Import Configuration window will pop up. The default selection will create all the necessary items for the AOI. Click the OK button to complete the import process.





5. The AOI is added to the Controller Organizer window and should look similar like the picture at left.
6. AOI installation into the Logix Designer software complete.

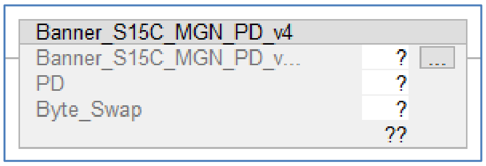
2. Configuring the IO-Link Master

Make an EtherNet/IP connection to the IO-Link Master.

Create an Ethernet communications module for the IO-Link Master device. The controller tags generated include Input (I) and Output (O) Assembly Instances. Each Assembly has a corresponding tag array. Creating this Class 1 EtherNet/IP implicit IO connection will provide the PLC access to the IO-Link device Process Data. Each port on the IO-Link Master is given a dedicated group of I and O registers. See the relevant IO-Link Master User’s Guide for more information.

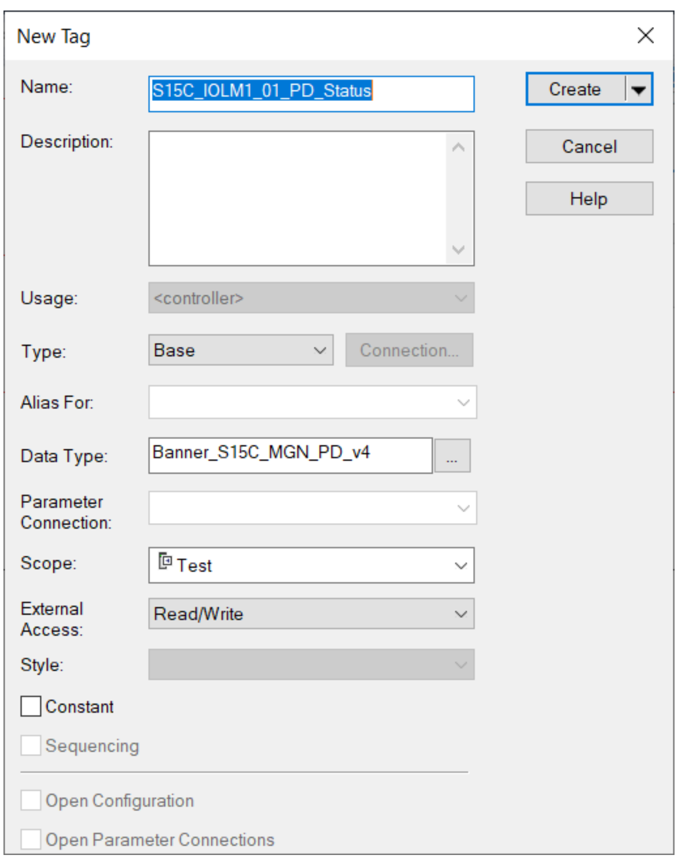
3. Configuring the AOI

1. Add the "Banner_S15C_MGN_PD_v4" AOI to your ladder logic program. For each of the question marks shown in the instruction we need to create and link a new tag array. The AOI includes a new type of User Defined Tags (UDT): a custom array of tags meant specifically for this AOI.



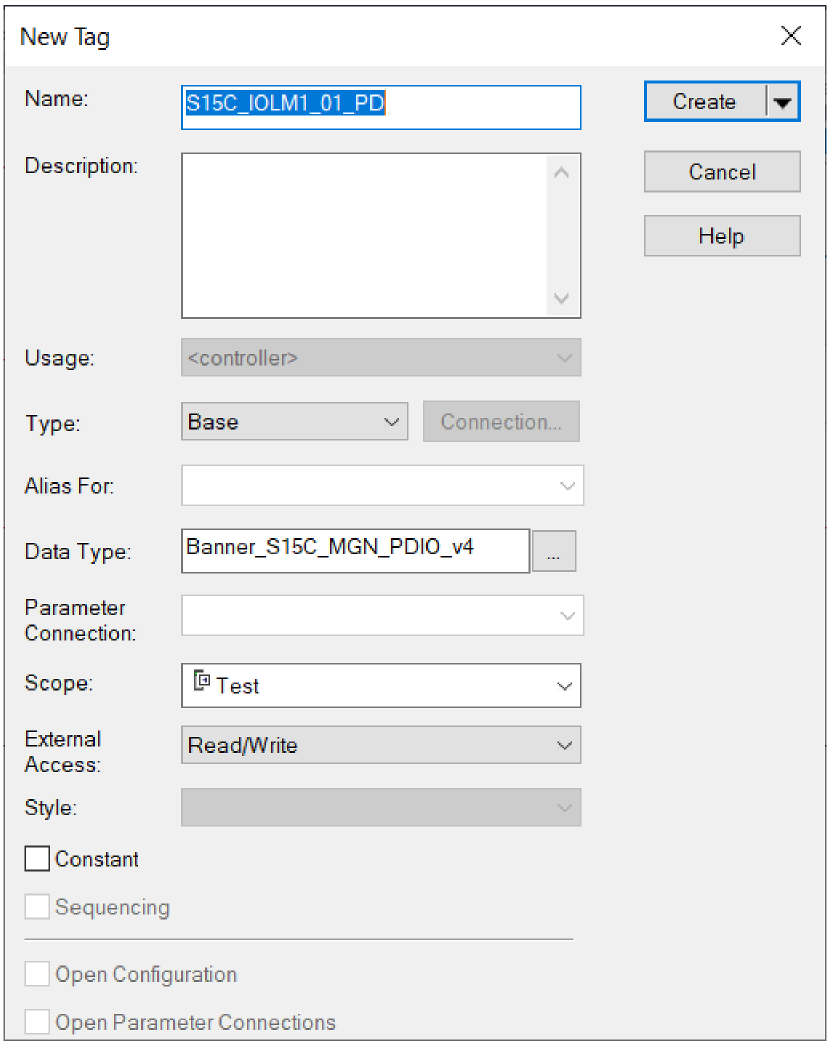
2. In the AOI, right-click on the question mark on the line labeled "Banner_S15C_MGN_PD_v4". Click New Tag. Name the new tag. This example uses the name "S15C_IOLM1_01_PD_Status". The example naming convention accounts for this being a S15C-MGN device connected to IO-Link Master #1, port #1, in our program. More masters could be named IOLM2, IOLM3, and different sensors could be connected at other port numbers, etc.

   Note that the Data Type is the User-Defined Data Type (UDT) entitled "Banner_S15C_MGN_PD_v4". This custom-made array of registers is specially built to handle the memory needs of this AOI. Click Create to make the tag array.



3. Now we will right-click on the question mark on the line labeled "Process_Data" in the AOI. Click on "New Tag". Give the tag a name. This example uses the name "S15C_IOLM1_01_PD". Notice that the Data Type is "Banner_S15C_MGN_PDIO_v4". Click Create.

   This array will handle the displaying of the parsed Process Data In and Process Data Out for the S15C-MGN.

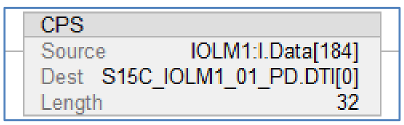


4. The next line in the AOI is a setting to account for byte swapping. In the case of the S15C-MGN, the Process Data In is 32 bytes long. IO-Link Masters may read each pair of bytes in either order, so this AOI must be ready to perform a byte swap. Enter a "0" or a "1" to toggle this setting. See Appendix B for more information. In this example, the byte swap is set to a value of "0".

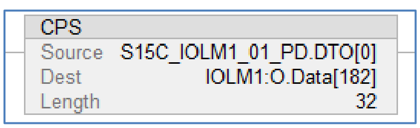
   NOTE: If the IO-Link Master you are using requires byte swapping be set to "1", the single byte of Process Data Out the S15C-MGN uses will show up in one register higher than that listed on table 1 in Appendix B.

5. The final two steps required before we download and run the S15C-MGN Process Data AOI involve a pair of File Synchronous Copy (CPS) instructions. These instructions allow the AOI to read from and write to the raw Process Data values found in the register tags of the IO-Link Master.

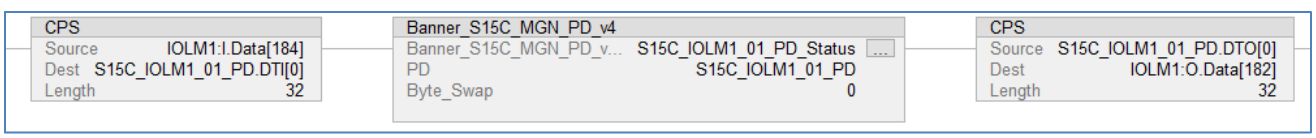
   Add a CPS instruction before the AOI on the ladder rung that looks like the one seen below. Refer to Appendix B for which byte to start with in the "Source" area. In this case, the IO-Link Master (Banner for this example) in question has the raw Process Data In values for a device connected to port 1 starting at byte 184. For the "Destination", we will enter the "PDI_DT[0]" location, as seen below. Finally, the length will be 32 bytes, as that is the size of the S15C-MGN Process Data In.



   Another CPS instruction is added to the AOI rung, this time after the AOI. This CPS instruction is used to copy Process Data Out from the AOI into the raw Process Data Out registers used by the IO-Link Master. See Appendix B for more information. In this example, we will connect the AOI's "Write_Data_Set" to the starting byte location for port 1 in the Process Data Out. In this example, that is byte 182. The size to be copied is 32 bytes.

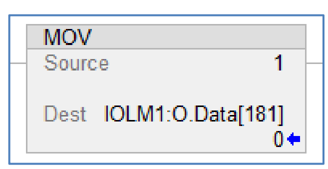


   Here is what the entire rung looks like when completed.



   If a Banner IO-Link Master is being used, setup a Move block. Send a 1 to the Activate Outputs array value (see table for each port's value). As an example, if port 1 needs the process data outputs active then send a 1 to 181.

IO-Link Master Port	Activate Outputs
1	181
2	215
3	249
4	283
5	317
6	351
7	385
8	419



   The "Banner_S15C_MGN_PD_v4" AOI is now ready for use.

4. Using the AOI

The "Banner_S15C_MGN_PD_v4" Add-On Instruction has created a group of tags representing the S15C-MGN Process Data In and Process Data Out, broken out into its component parts.

Look in the Controller Tags to find the name you used in Step 4 above. This example used the name "S15C_IOLM1_01_PD". The tag array, seen below, has individual pieces of information instead of a group of unlabeled bits. The parameter "Set_to_Read" controls which of the four (0 to 3) data sets that is currently being read. In the example below, the value is set to 0. This means Read Data Set 0 is read. The valid options are 0, 1, 2, and 3. The reading of the data set is working when the "Read_Success" has a value of 1. If it has a value of 0 than the data is not valid. The parameter "Read_Data_Set" should also come back with the same number as the "Set_to_Read". This is just confirming the data set. The read information is stored in the Read_Regs array.

The Process Data Out is shown near the bottom of the image. The "Set_to_Read" controls which of the 4 data sets is read. The "Write_Regs" is an array of 15 elements that is used to write data Modbus Registers.

The Read and Write data sets are configured via Parameter Data. Please see the Parameter Data pdf for information on that.

◢ S15C_IOLM1_01_PD	{...}
◢ S15C_IOLM1_01_PD.Read	{...}
▷ S15C_IOLM1_01_PD.Read.Read_Data_Set	0
▷ S15C_IOLM1_01_PD.Read.Read_Success	0
▷ S15C_IOLM1_01_PD.Read.Write_Success	0
▷ S15C_IOLM1_01_PD.Read.Register_Verify	0
▷ S15C_IOLM1_01_PD.Read.Counter_Value	0
▷ S15C_IOLM1_01_PD.Read.Read_Regs	{...}
▷ S15C_IOLM1_01_PD.Write	{...}
▷ S15C_IOLM1_01_PD.DTI	{...}
▷ S15C_IOLM1_01_PD.DTO	{...}

Appendix A S15C-MGN Process Data

The S15C-MGN has 32 bytes of Process Data In and 31 bytes of Process Data Out, as shown below. There are three modes for this Process Data, called Register Sets to Read. The Process Data Out controls which of the Registers Sets defines the Process Data In (0, 1, or 2).

ProcessDataIn "Process Data In" id=V_Pd_InData

bit length: 256
data type: 256-bit Record (subindex access not supported)

subindex	bit offset	data type	allowed values	default value	acc. restr.	mod. other var.	excl. from DS	name	description
1	0	4-bit UInteger	0..1					Register Set To Read	Register Set To Read
2	4	Boolean	false = False, true = True					Register Read Successful	Register Read Successful
3	5	Boolean	false = False, true = True					Register Write Successful	Register Write Successful
4	6	Boolean	false = False, true = True					Register Verify Successful	Register Verify Successful
5	8	8-bit UInteger	0..255					Counter Value	Counter Value
6	16	16-bit UInteger	0..65535					Read Set Register 01 Value	Register Value. Data that was read from register.
7	32	16-bit UInteger	0..65535					Read Set Register 02 Value	Register To Read. Register to read from ModBus device.
8	48	16-bit UInteger	0..65535					Read Set Register 03 Value	Register Value. Data that was read from register.
9	64	16-bit UInteger	0..65535					Read Set Register 04 Value	Register To Read. Register to read from ModBus device.
10	80	16-bit UInteger	0..65535					Read Set Register 05 Value	Register Value. Data that was read from register.
11	96	16-bit UInteger	0..65535					Read Set Register 06 Value	Register To Read. Register to read from ModBus device.
12	112	16-bit UInteger	0..65535					Read Set Register 07 Value	Register Value. Data that was read from register.
13	128	16-bit UInteger	0..65535					Read Set Register 08 Value	Register Value. Data that was read from register.
14	144	16-bit UInteger	0..65535					Read Set Register 09 Value	Register Value. Data that was read from register.
15	160	16-bit UInteger	0..65535					Read Set Register 10 Value	Register Value. Data that was read from register.
16	176	16-bit UInteger	0..65535					Read Set Register 11 Value	Register Value. Data that was read from register.
17	192	16-bit UInteger	0..65535					Read Set Register 12 Value	Register Value. Data that was read from register.
18	208	16-bit UInteger	0..65535					Read Set Register 13 Value	Register Value. Data that was read from register.
19	224	16-bit UInteger	0..65535					Read Set Register 14 Value	Register Value. Data that was read from register.
20	240	16-bit UInteger	0..65535					Read Set Register 15 Value	Register Value. Data that was read from register.

ProcessDataOut "Process Data Out" id=V_Pd_OutData

bit length: 248
data type: 248-bit Record (subindex access not supported)

subindex	bit offset	data type	allowed values	default value	acc. restr.	mod. other var.	excl. from DS	name	description
1	0	8-bit UInteger	0..3					Register Set To Read	Register Set To Read
2	8	16-bit UInteger	0..65535					Write Set Register 01 Value	Value To Write into register. Register value to writre to ModBus device.
3	24	16-bit UInteger	0..65535					Write Set Register 02 Value	Register To Read. Register to read from ModBus device.
4	40	16-bit UInteger	0..65535					Write Set Register 03 Value	Value To Write into register. Register value to writre to ModBus device.
5	56	16-bit UInteger	0..65535					Write Set Register 04 Value	Register To Read. Register to read from ModBus device.
6	72	16-bit UInteger	0..65535					Write Set Register 05 Value	Value To Write into register. Register value to writre to ModBus device.
7	88	16-bit UInteger	0..65535					Write Set Register 06 Value	Register To Read. Register to read from ModBus device.
8	104	16-bit UInteger	0..65535					Write Set Register 07 Value	Value To Write into register. Register value to writre to ModBus device.
9	120	16-bit UInteger	0..65535					Write Set Register 08 Value	Value To Write into register. Register value to writre to ModBus device.
10	136	16-bit UInteger	0..65535					Write Set Register 09 Value	Value To Write into register. Register value to writre to ModBus device.
11	152	16-bit UInteger	0..65535					Write Set Register 10 Value	Value To Write into register. Register value to writre to ModBus device.
12	168	16-bit UInteger	0..65535					Write Set Register 11 Value	Value To Write into register. Register value to writre to ModBus device.
13	184	16-bit UInteger	0..65535					Write Set Register 12 Value	Value To Write into register. Register value to writre to ModBus device.
14	200	16-bit UInteger	0..65535					Write Set Register 13 Value	Value To Write into register. Register value to writre to ModBus device.
15	216	16-bit UInteger	0..65535					Write Set Register 14 Value	Value To Write into register. Register value to writre to ModBus device.
16	232	16-bit UInteger	0..65535					Write Set Register 15 Value	Value To Write into register. Register value to writre to ModBus device.

Appendix B IO-Link Master Cheat Sheet

Different IO-Link Masters behave differently in several ways. For one, the register locations where Process Data is stored varies. For another, some IO-Link Masters require byte-swapping and/or word-swapping. The tables below aim to define some of these differences. Note that these numbers are when using all default settings. IO-Link Masters can change the register locations to which Process Data is mapped in response to non-default, optional settings. See relevant IO-Link Master documentation for more information.

PDI (Process Data In) is found in the IO-Link Master's T->O (PLC "Input") Assembly Instance.

PDO (Process Data Out) is found in the IO-Link Master's O->T (PLC "Output") Assembly Instance.

Table 1. First Register of Process Data "SINT0"

	Allen-Bradley*		Comtrol		Balluff		Turck		ifm		Banner	
Port	PDI	PDO	PDI	PDO	PDI	PDO	PDI	PDO	PDI	PDO	PDI	PDO
1	I.Ch0Data[0]	O.Ch0Data[0]	4	0	8	6	6	4	190	46	184	182
2	I.Ch1Data[0]	O.Ch1Data[0]	40	32	56	38	38	36	222	78	218	216
3	I.Ch2Data[0]	O.Ch2Data[0]	76	64	104	70	70	68	254	110	252	250
4	I.Ch3Data[0]	O.Ch3Data[0]	112	96	152	102	102	100	286	142	286	284
5	I.Ch4Data[0]	O.Ch4Data[0]	148	128	200	134	134	132	318	174	320	318
6	I.Ch5Data[0]	O.Ch5Data[0]	184	160	248	166	166	164	350	206	354	352
7	I.Ch6Data[0]	O.Ch6Data[0]	220	192	296	198	198	196	382	238	388	386
8	I.Ch7Data[0]	O.Ch7Data[0]	256	224	344	230	230	228	414	270	422	420

*see relevant Banner Allen-Bradley IO-Link Master AOI Guide and Allen-Bradley User Guides for more information on using device IODD files to aid in integration.

Note: Murr IO-Link Masters have configurable process data. Refer to the Murr IO-Link Master Instruction Manual for Process Data mappings.

Table 2. Byte-Swap

IO-Link Master	Byte Swap
Allen-Bradley	0
Comtrol	1
Balluff	0
Turck	1
ifm	1
Murr	0
Banner	0

Specific hardware used in both tables (all default settings):

- Allen-Bradley Armor Block I/O IO-Link Master (1732E-8IOLM12R)
- Comtrol 8-EIP IO-Link Master (99608-8)
- Balluff BNI006A (BNI EIP-508-105-Z015)
- Turck TBEN-L5-8IOL
- ifm AL1122
- Murr Impact67 E DIO 12 DIO4/IOL4 4P (Art.-No. 55144)

Banner IO-Link Masters (DXMR90-4K) has a port status register. The register gives the status of the port. It gives information on if the port has an IO-Link device connected and if Process Data is valid. This is optional information but is useful for troubleshooting. The data comes into the PLC as bytes while the literature shows the value as a word. The table below gives the upper- and lower-byte data location in the PLC. The upper byte includes bits 15 through 8, while the lower byte has bits 7 through 0.

IO-Link Master Port	Upper Bits 15 - 8	Lower Bits 7 - 0
1	182	183
2	216	217
3	250	251
4	284	285
5	318	319
6	352	353
7	386	387
8	420	421

Port Status:

Bit0 = Connected?
Bit1 = Process Data Valid?
Bit2 = Event Pending?
Bit3 = Ready for ISDU?
Bit4 = Pin4 SIO State
Bit5 = Pin2 SIO State

Bit6-7 = Pin4 Mode:
SDCI Mode = 0
SIO Input Mode = 1
SIO Output Mode = 2

Bit8-10 = Pin2 Mode:
Disabled = 0
Input Normal = 1
Output = 2
Diagnostic Input = 3
Inverted Input = 4