



IC70-16P-KQ Pro Process Data AOI Guide, v4 **January 17th, 2025**

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 Out to a Banner IC70 device via an IO-Link Master connected to an Allen-Bradley PLC. The AOI covers parsing and display of the IC70-16P-KQ Pro Process Data Out. The AOI has six User Defined Tag data types and two AOIs.

Components

Banner_IC70_PDIO_v4_AOI.L5X

Packaged with the AOI

Banner_IC70_16P_PDI_v4

Banner_IC70_16P_PDO_v4

Banner_IC70_16P_PD_v4

Other AOIs Available Separately

Banner has AOI files for controlling other Banner IO-Link devices and for a variety of IO-Link Masters. Banner also has AOI files for easily handling Banner device Process Data.

Contents

1. Installation Process 1

2. Configuring the IO-Link Master 3

3. Configuring the AOI..... 4

4. Using the AOI..... 7

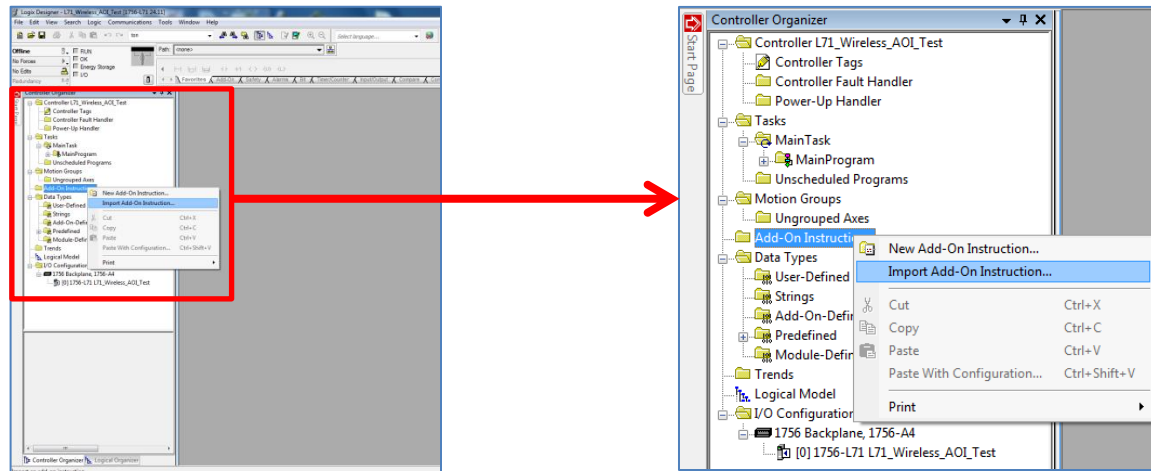
Appendix A IC70 16P Pro Process Data 8

Appendix B IO-Link Master Cheat Sheet 10

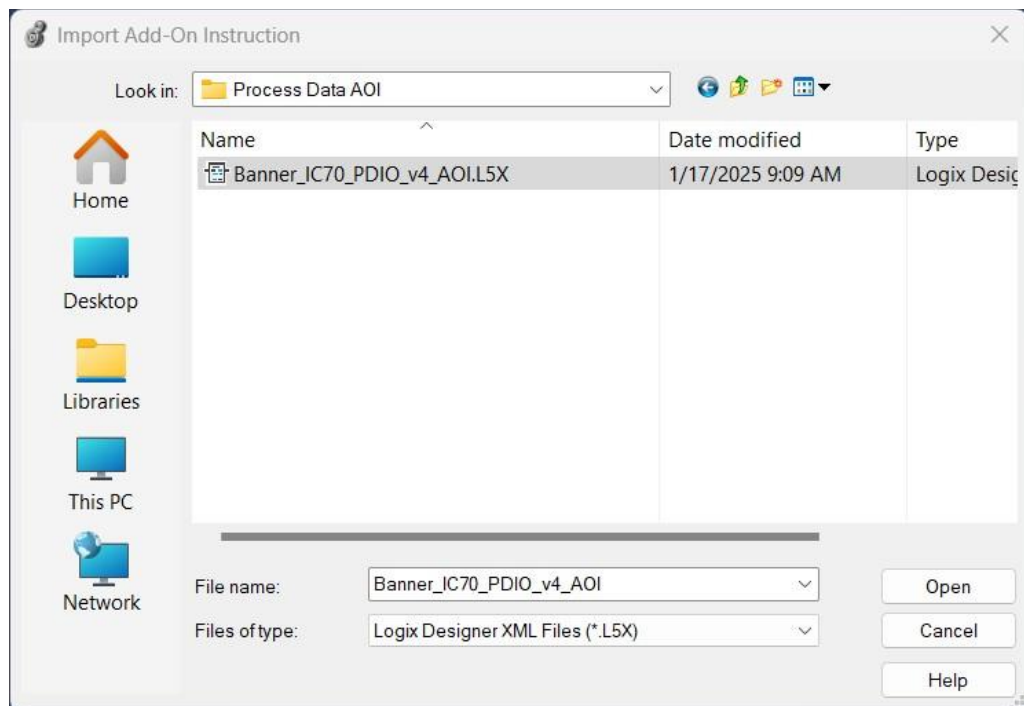
1. Installation Process

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

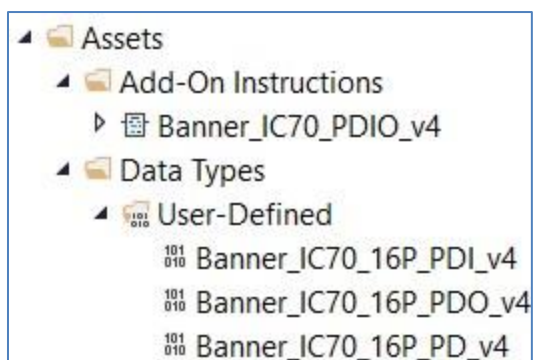
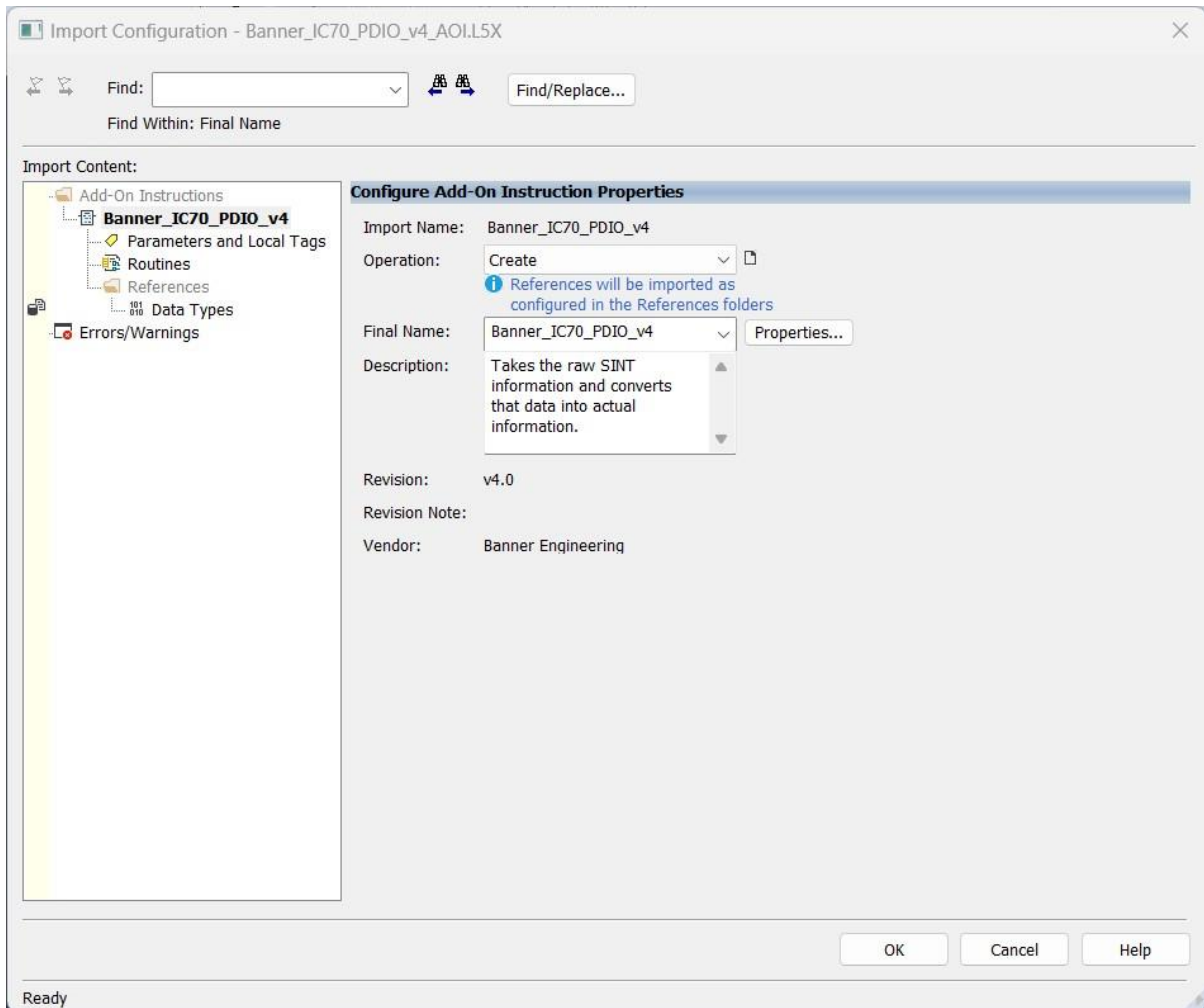
1. Open a project.
2. In the Controller Organizer window, right-click on the Add-On Instruction folder. Select the Import Add-On Instruction option.



3. Navigate to the correct file location and select the AOI to be installed. In this example the "Banner_IC70_PDIO_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 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 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.

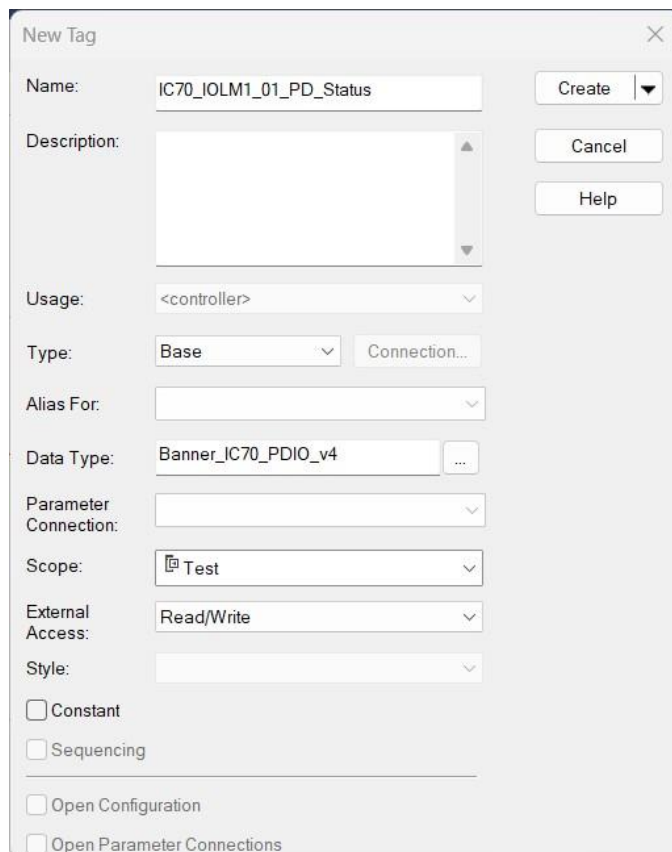
3. Configuring the AOI

1. Add the “Banner_IC70_PDIO_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.



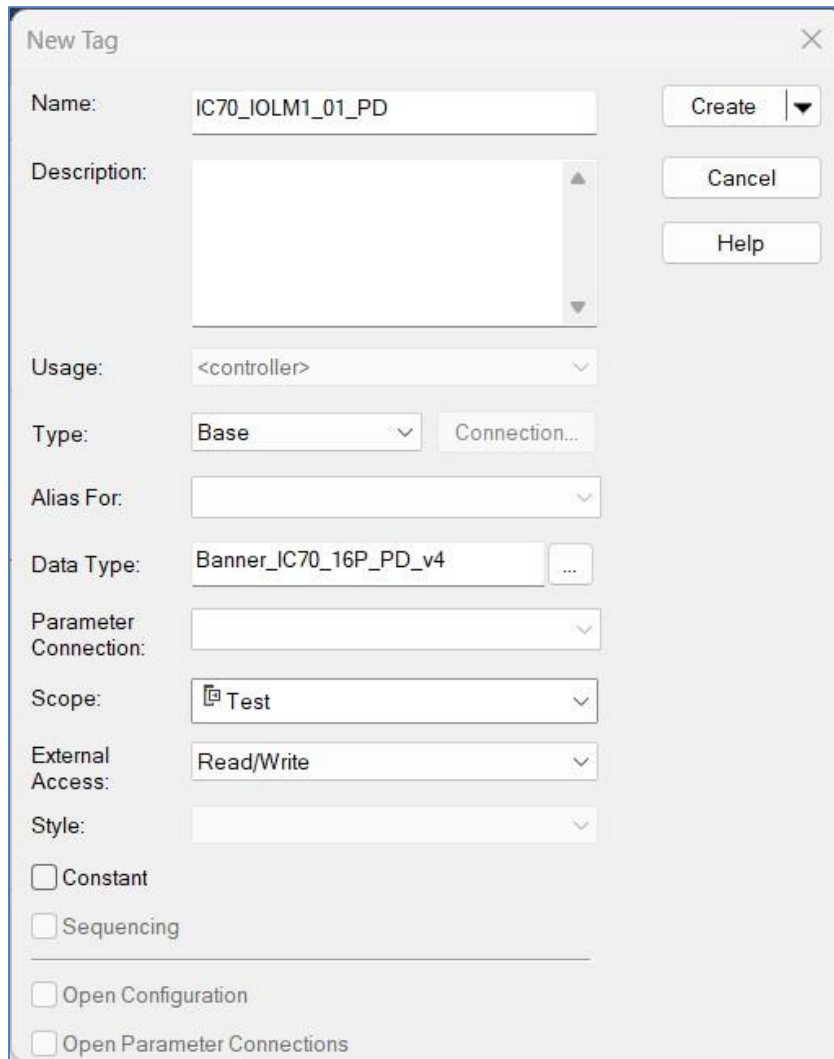
2. In the AOI, right-click on the question mark on the line labeled “Banner_IC70_PDIO_v4”. Click New Tag. Name the new tag. This example uses the name “IC70_IOLM1_01_PD_Status”. The example naming convention accounts for this being an IC70 device connected to IO-Link Master #1, port #1, in our program. 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_IC70_PDIO_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.



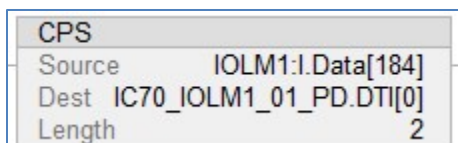
- Now we will right-click on the question mark on the line labeled "Process_Data" in the AOI. Click on "New Tag". Give the tag a name. This example uses the name "IC70_IOLM1_01_PD". Notice that the Data Type is "Banner_IC70_16P_PDIO_v4". Click Create.

This array will handle the displaying of the parsed Process Data Out for the IC70 device.

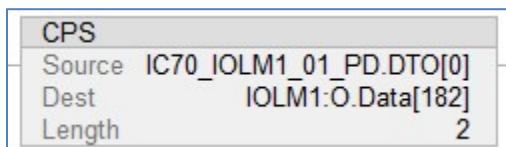


- The next line in the AOI is a setting to account for byte swapping. In the case of the IC70, the Process Data In and Out is 2 bytes long. IO-Link Masters may read each pair of bytes in either order, so this AOI must be ready to perform a byte swap. Enter a "0" or a "1" to toggle this setting. See Appendix B for more information.

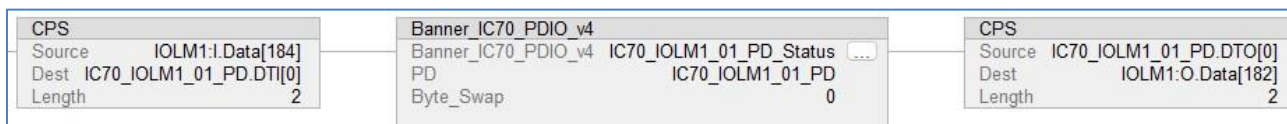
5. The final step required before we download and run the IC70 Process Data AOI involves two File Synchronous Copy (CPS) instructions. A CPS instruction is added before and after the AOI. These CPS instructions are used to copy Process Data In and Out from the AOI into the raw Process Data registers used by the IO-Link Master. See Appendix B for more information. In this example, we will connect the raw data for port 1 (Data[184] for a Banner IO-Link Master) to the DTI[0] AOI's. This creates the link for the Process Data In.



The example for CPS for the Process data out is shown below. Here the DTO[0] is taken and copied to the raw output register for the IO-Link Master (Banner's Output address for Port 1 is 182).

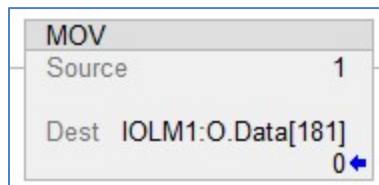


Here is what the entire rung looks like when completed.



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_IC70_16P_PD_v4" AOI is now ready for use.

4. Using the AOI

The “Banner_IC70_PDIO_v4” Add-On Instruction has created a group of tags representing the SD50 Pro Process Data, broken out into its component parts.

Look in the Controller Tags to find the name you used above. This example used the name “IC70_IOLM1_01_PD”. The tag array, seen below, has individual pieces of information instead of unlabeled bits.

There are two locations to be interested in. The first is PDI which is for Process Data In. The other is PDO which is Process Data Output. The PDI has 16 tags in the format of C#_In. The C represents Channel while the # represents the channel number. When the input is detected a value of 1 is shown. C#_Out follows the same format as the input. When the output is set to one the output is activated.

▲ IC70_IOLM1_01_PD
▶ IC70_IOLM1_01_PD.DTI
▶ IC70_IOLM1_01_PD.DTO
▶ IC70_IOLM1_01_PD.PDI
▶ IC70_IOLM1_01_PD.PDO

▲ IC70_IOLM1_01_PD.PDI	{...}
IC70_IOLM1_01_PD.PDI.C1_In	1
IC70_IOLM1_01_PD.PDI.C2_In	0
IC70_IOLM1_01_PD.PDI.C3_In	0
IC70_IOLM1_01_PD.PDI.C4_In	0
IC70_IOLM1_01_PD.PDI.C5_In	0
IC70_IOLM1_01_PD.PDI.C6_In	0
IC70_IOLM1_01_PD.PDI.C7_In	0
IC70_IOLM1_01_PD.PDI.C8_In	0

▲ IC70_IOLM1_01_PD.PDO	{...}
IC70_IOLM1_01_PD.PDO.C1_Out	1
IC70_IOLM1_01_PD.PDO.C2_Out	0
IC70_IOLM1_01_PD.PDO.C3_Out	0
IC70_IOLM1_01_PD.PDO.C4_Out	0
IC70_IOLM1_01_PD.PDO.C5_Out	0
IC70_IOLM1_01_PD.PDO.C6_Out	0
IC70_IOLM1_01_PD.PDO.C7_Out	0
IC70_IOLM1_01_PD.PDO.C8_Out	0

Appendix A IC70 16P Pro Process Data

The IC70 has 2 bytes of Process Data In and Out.

This AOI intelligently parses this Process Data into its component pieces.

Process Data format is shown here. If the full array is needed look at the IODD file for the device.

ProcessDataIn "Process Data Input" id=PD_ProcessDataIn									
bit length: 16									
data type: 16-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	8	Boolean	false = Inactive, true = Active					Channel 1 Input State	true (1) = Channel Input Active. Note - even if is configured as an output, the active state will be reflected at the input
2	9	Boolean	false = Inactive, true = Active					Channel 2 Input State	true (1) = Channel Input Active. Note - even if is configured as an output, the active state will be reflected at the input
3	10	Boolean	false = Inactive, true = Active					Channel 3 Input State	true (1) = Channel Input Active. Note - even if is configured as an output, the active state will be reflected at the input
4	11	Boolean	false = Inactive, true = Active					Channel 4 Input State	true (1) = Channel Input Active. Note - even if is configured as an output, the active state will be reflected at the input
5	12	Boolean	false = Inactive, true = Active					Channel 5 Input State	true (1) = Channel Input Active. Note - even if is configured as an output, the active state will be reflected at the input
6	13	Boolean	false = Inactive, true = Active					Channel 6 Input State	true (1) = Channel Input Active. Note - even if is configured as an output, the active state will be reflected at the input
7	14	Boolean	false = Inactive, true = Active					Channel 7 Input State	true (1) = Channel Input Active. Note - even if is configured as an output, the active state will be reflected at the input
8	15	Boolean	false = Inactive, true = Active					Channel 8 Input State	true (1) = Channel Input Active. Note - even if is configured as an output, the active state will be reflected at the input
9	0	Boolean	false = Inactive, true = Active					Channel 9 Input State	true (1) = Channel Input Active. Note - even if is configured as an output, the active state will be reflected at the input
10	1	Boolean	false = Inactive, true = Active					Channel 10 Input State	true (1) = Channel Input Active. Note - even if is configured as an output, the active state will be reflected at the input
11	2	Boolean	false = Inactive, true = Active					Channel 11 Input State	true (1) = Channel Input Active. Note - even if is configured as an output, the active state will be reflected at the input
12	3	Boolean	false = Inactive, true = Active					Channel 12 Input State	true (1) = Channel Input Active. Note - even if is configured as an output, the active state will be reflected at the input

Process Data Out

ProcessDataOut "Process Data Output" id=PD_ProcessDataOut

bit length: 16

data type: 16-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	8	Boolean	false = Off, true = On					Channel 1 Output State	true (1) = Channel Output Active
2	9	Boolean	false = Off, true = On					Channel 2 Output State	true (1) = Channel Output Active
3	10	Boolean	false = Off, true = On					Channel 3 Output State	true (1) = Channel Output Active
4	11	Boolean	false = Off, true = On					Channel 4 Output State	true (1) = Channel Output Active
5	12	Boolean	false = Off, true = On					Channel 5 Output State	true (1) = Channel Output Active
6	13	Boolean	false = Off, true = On					Channel 6 Output State	true (1) = Channel Output Active
7	14	Boolean	false = Off, true = On					Channel 7 Output State	true (1) = Channel Output Active
8	15	Boolean	false = Off, true = On					Channel 8 Output State	true (1) = Channel Output Active
9	0	Boolean	false = Off, true = On					Channel 9 Output State	true (1) = Channel Output Active
10	1	Boolean	false = Off, true = On					Channel 10 Output State	true (1) = Channel Output Active
11	2	Boolean	false = Off, true = On					Channel 11 Output State	true (1) = Channel Output Active
12	3	Boolean	false = Off, true = On					Channel 12 Output State	true (1) = Channel Output Active
13	4	Boolean	false = Off, true = On					Channel 13 Output State	true (1) = Channel Output Active
14	5	Boolean	false = Off, true = On					Channel 14 Output State	true (1) = Channel Output Active
15	6	Boolean	false = Off, true = On					Channel 15 Output State	true (1) = Channel Output Active
16	7	Boolean	false = Off, true = On					Channel 16 Output State	true (1) = Channel Output Active

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"

Port	Allen-Bradley*		Comtrol		Balluff		Turck		ifm		Banner	
	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 and DXMR110-8K) have 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