



## **QCM50-K1 Process Data AOI Guide, v2**

**5/23/2019**

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 from a Banner QCM50-K1 sensor via an IO-Link Master to an Allen-Bradley PLC. The AOI covers parsing and display of the QCM50-K1 sensor Process Data In. The AOI has one User Defined Tag data type.

### **Components**

Banner\_QCM50\_K1\_PD\_v2.L5X

### **UDT Packaged with the AOI**

Banner\_QCM50\_K1\_PDI\_v2

### **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 QCM50-K1 Process Data..... 8

Appendix B IO-Link Master Cheat Sheet ..... 9

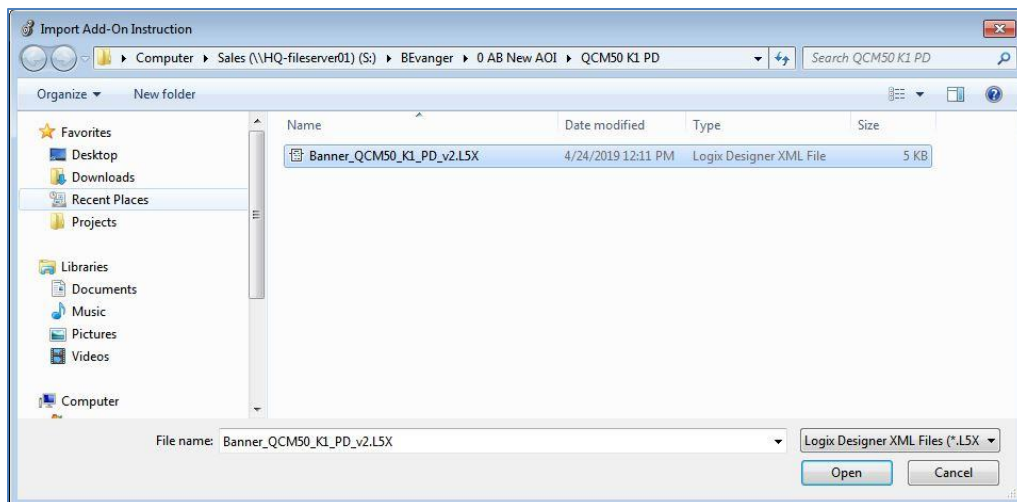
## 1. Installation Process

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

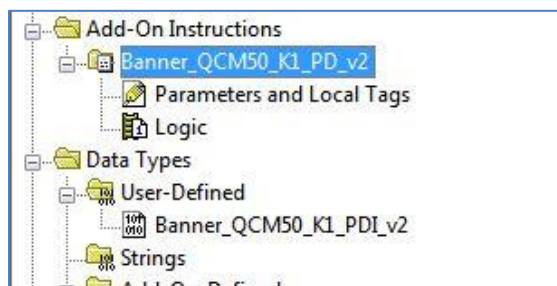
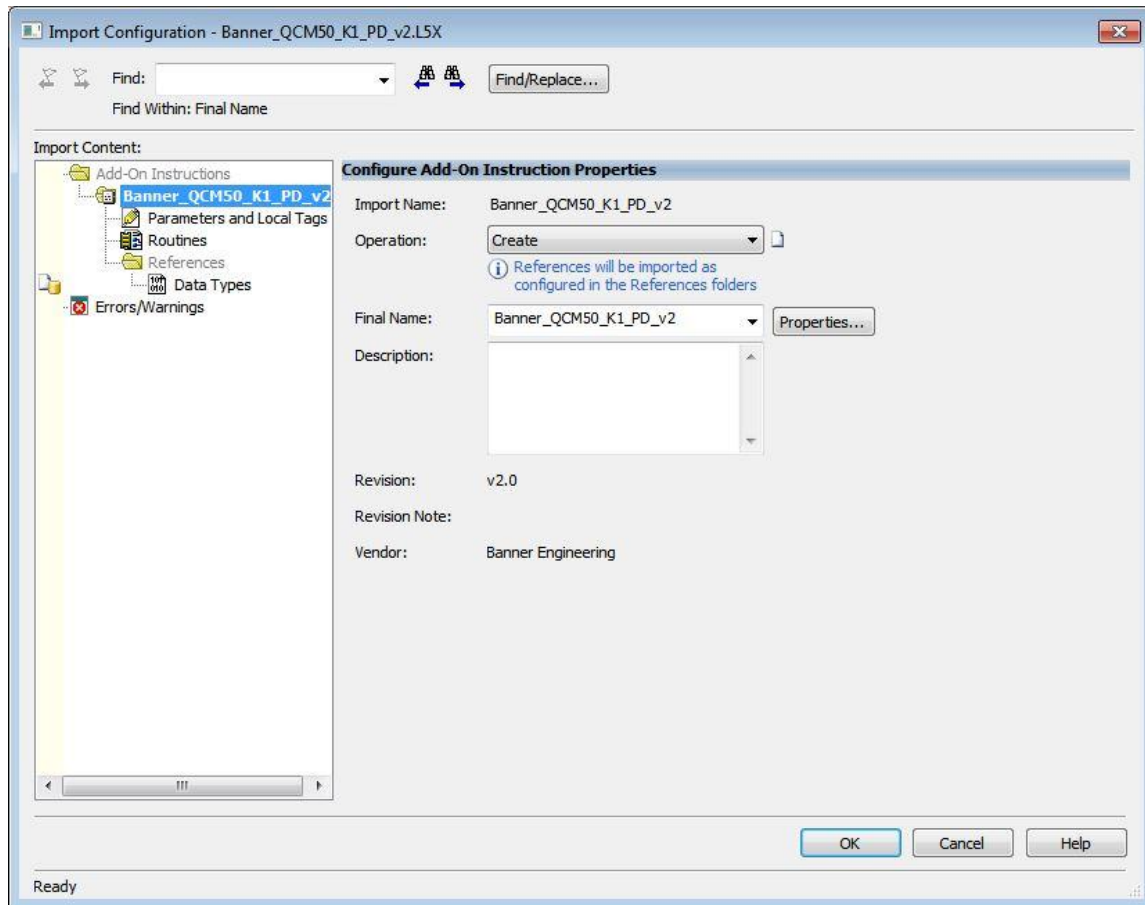
1. Open up a project.
2. In the Controller Organizer window, right-click on the Add-On Instruction folder. Select the Import Add-On Instruction option.



3. Navigate to the correct file location and select the AOI to be installed. In this example the "Banner\_QCM50\_K1\_PD\_v2.L5X" file will be selected. Click the Open button.



4. The Import Configuration window will pop up. The default selection will create all of 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 to 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\_QCM50\_K1\_PD\_v2” 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\_QCM50\_K1\_PD\_v2”. Click New Tag. Name the new tag. This example uses the name “QCM50\_K1\_IOLM2\_5\_PD\_Status”. The example naming convention accounts for this being a QCM50 sensor connected to IO-Link Master #2, port #5, in our program. More masters could be named IOLM1, 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\_QCM50\_K1\_PD\_v2”. This custom-made array of registers is specially built to handle the memory needs of this AOI. Click Create to make the tag array.

New Tag

Name: QCM50\_K1\_IOLM2\_5\_PD\_Status Create

Description:

Usage: <controller>

Type: Base Connection...

Alias For:

Data Type: Banner\_QCM50\_K1\_PD\_v2

Parameter Connection:

Scope: Test

External Access: Read/Write

Style:

☒ Constant

☐ Sequencing

☐ Open Configuration

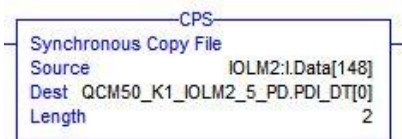
☐ Open Parameter Connections

- Now we will right-click on the question mark on the line labeled “PDI” in the AOI. Click on “New Tag”. Give the tag a name. This example uses the name “QCM50\_K1\_IOLM2\_5\_PD”. Notice that the Data Type is “Banner\_QCM50\_K1\_PDI\_v2”. Click Create.

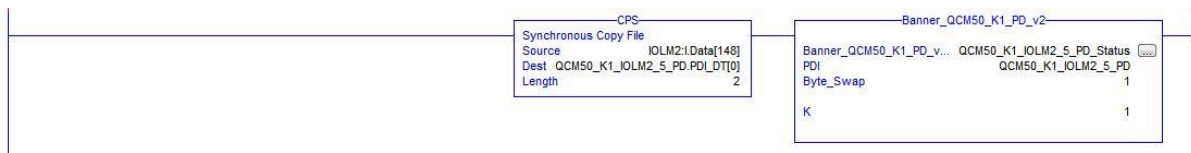
The screenshot shows the "New Tag" dialog box. The "Name" field is populated with "QCM50\_K1\_IOLM2\_5\_PD". The "Data Type" field is populated with "Banner\_QCM50\_K1\_PDI\_v2". The "Scope" dropdown is set to "Test". The "External Access" dropdown is set to "Read/Write". The "Constant" checkbox is checked. The "Sequencing", "Open Configuration", and "Open Parameter Connections" checkboxes are unchecked.

- The “Byte\_Swap” line in the AOI is a setting to account for byte swapping. In the case of the QCM50-K1, the Process Data is 2 bytes long. IO-Link Masters may read those bytes in either order, so this AOI has to be ready to perform a byte swap. Enter a “0” or a “1” to toggle this setting. See Appendix B for more information.
- For the line labeled “K”, enter a 1.

6. The final step required before we download and run the QCM50 Process Data AOI involves a File Synchronous Copy (CPS) instruction. A CPS instruction is added to the AOI rung, before the AOI. This CPS instruction is used to copy Process Data In into the AOI from the raw Process Data registers used by the IO-Link Master. See Appendix B for more information. In this example, we will connect the starting byte location for port 5 in the Process Data In to the AOI’s “PDI\_DT[0]” array. Specifically, we are copying from the IO-Link Master Input Assembly Instance, starting at byte 148. The size to be copied is 2 bytes.



Here is what the entire rung looks like when completed.



The “Banner\_QCM50\_K1\_PD\_v2” AOI is now ready for use.



## 4. Using the AOI

The “Banner\_QCM50\_K1\_PD\_v2” Add-On Instruction has created a group of tags representing the QCM50 Process Data, 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 “QCM50\_K1\_IOLM2\_5\_PD”. The tag array, seen below, has individual pieces of information instead of 16 unlabeled bits.

[-] QCM50_K1_IOLM2_5_PD	{...}	{...}		Banner_QCM50_K1_PDI_v2
[+] QCM50_K1_IOLM2_5_PD.Signal_Quality	1		Decimal	INT
[+] QCM50_K1_IOLM2_5_PD.Switching_Quality	0		Decimal	SINT
[+] QCM50_K1_IOLM2_5_PD.Q1	2		Decimal	SINT
[+] QCM50_K1_IOLM2_5_PD.PDI_DT	{...}	{...}	Decimal	SINT[2]

## Appendix A

## QCM50-K1 Process Data

The QCM50-K1 has 2 bytes of Process Data In, as shown below.

ProcessDataIn "Switching channel" id=PDIN_SwitchingOutput									
bit length: 16 data type: 16-bit Record									
subindex	bit offset	data type	allowed values	default value	acc. restr.	mod. other var.	excl. from DS	name	description
1	8	8-bit UInteger	0 = no signal, 1 = overflow, 2..100		ro			Signal quality	
2	1	Boolean			ro			Switching quality	
3	0	Boolean			ro			Q1	
Octet 0									
bit offset	15	14	13	12	11	10	9	8	
subindex									
element bit	7	6	5	4	3	2	1	0	
Octet 1									
bit offset	7	6	5	4	3	2	1	0	
subindex	1	2	3	4	5	6	7	8	

This Process Data is mapped to a specific group of EtherNet/IP registers. The 16-bits of Process Data In actually encode three separate pieces of information.

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

## 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) 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