

# TL50 Process Data Function

---

January 16<sup>th</sup>, 2026

This document covers the installation and use of a function for Siemens's TIA Portal software package. This function handles cyclic IO-Link Process Data Out to a Banner TL50 light via an IO-Link Master from a Siemens PLC. The function covers parsing and display of the TL50 light Process Data Out.

## **Components**

Banner TL50 Library V16.zal16

There are two methods for the process data. The first is used when creating a connection to Banner's IO-Link masters. The second set of instructions are for systems using other manufacturer's IO-Link masters.

### **Installation Instructions**

1. Open a project.
2. Go to the Open Global Library option in the Libraries tab in TIA Portal v16 or greater.



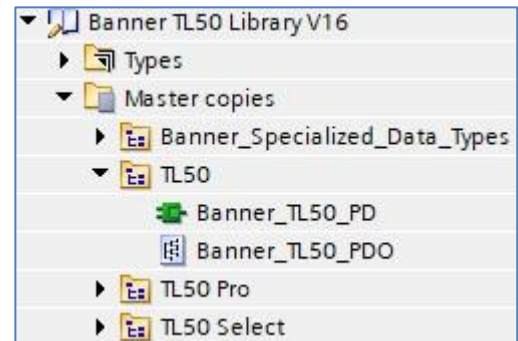
3. Switch the “Files of type” to Compressed libraries. Go to the location of the compressed library.
4. Press the Open button and the library will be uncompressed and opened.
5. The library is now accessible in the Libraries tab in v16 or greater.

### **Setup of TL50 with a Banner DXMR**

1. Go to Device and Networks to configure the DXMR. Add the DXMR if it has yet to be added to the system.
2. Add Banner IO-Link Master Info to Slot 1. This sets the DXMR for IO-Link mode.
3. Open the IO-Link Generic Devices and select the proper module. The 2/2 byte is required for TL50. Make note of the I address for the Slot 2 which represents Port 1. Slot 2 starts are 1 for outputs. The other number needed is I3. The data for the port start at that point (I3). The previous two bytes Port Control.

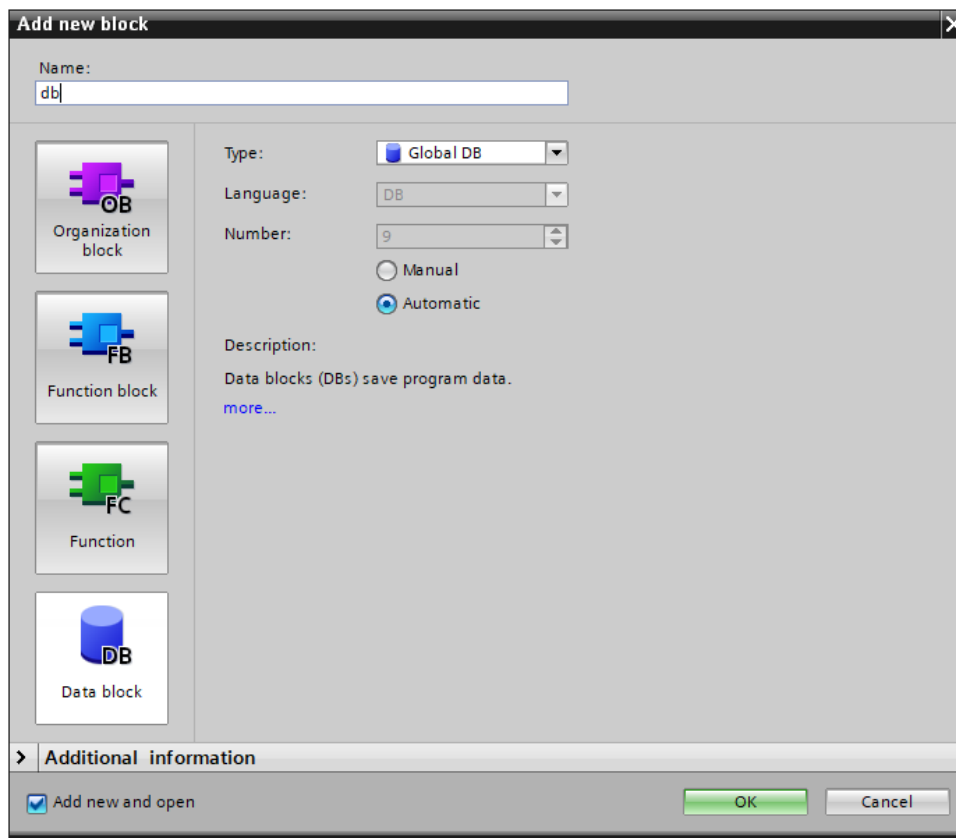
Module	Rack	Slot	I address	Q address	Type
▼ dxm	0	0			1-port Device
▶ Interface	0	0 X1			dxm
Banner IO-Link Master Info_1	0	1	1...9		Banner IO-Link Master Info
IO-Link In/Out 2/ 2 Byte + Status_1	0	2	10...15	1...16	IO-Link In/Out 2/ 2 Byte + Status

4. Drag the necessary tag from IOLM\_Control > Banner > Banner\_Specialized\_Data\_Types. The tag used in this example is “Banner\_2out”. This tag represents the full raw process data along with port status information.
5. Drag the necessary files from the TL50 Folder.
  - a. Move Banner\_TL50\_PDO to the PLC Data Types area.
  - b. Move Banner\_TL50\_PD to the Program Blocks area.
6. Go to PLC Tags. Create two tags. One tag is for the full data structure while the second creates a tag to represent the raw Process Data from the IO-Link Master. In this example, Tag table\_1 was created, then the tag “TL50 IOLM1 01 PDO” was created using a Data Type of “Banner\_2out”. This naming convention calls out the type of device in question as well as the specific IO-Link Master and port number where the sensor is connected. A different IO-Link Master might be named IOLM2 or IOLM3, for instance, and other specific sensors may be connected to different port numbers. The “Q” address found in step 2 (%Q1) is tied to this new tag. The second is “TL50 IOLM1 01 outRaw” and uses the “I” address found in step 2 (%Q3). This uses the “UINT” data type. This is the tag that will be used in the Function block.



Name	Data type	Address
▶ TL50 IOLM1 01 PDO	"Banner_2Out"	%Q1.0
TL50 IOLM1 01 outRaw	UInt	%QW3

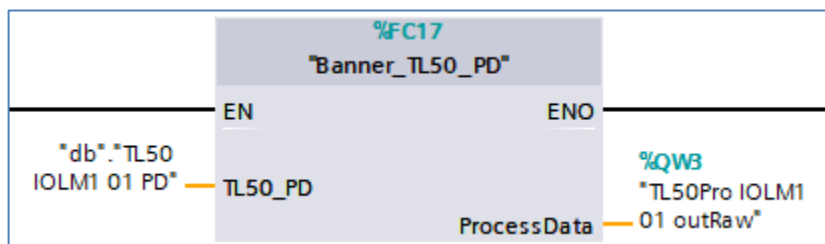
7. Go to Program blocks. Add a new Data block if necessary. In this example the new data block is named “db”.



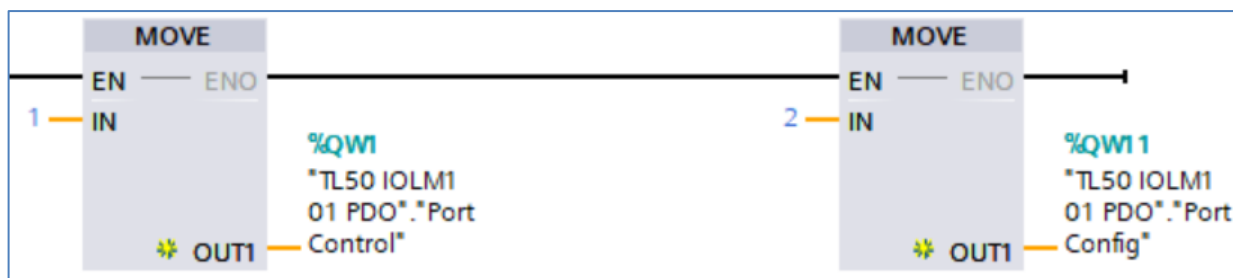
8. In the new data block, create a new tag to represent the parsed Process Data In for our TL50. The tag name again calls out the type of sensor, the IO-Link Master, and the port number. Use the data type “Banner\_TL50\_PDO” for the new tag.

Name	Data type
▼ Static	
■ ▼ TL50 IOLM1 01 PD	"Banner_TL50_PDO"
■ ▼ Segment	Array[1..7] of USInt
■ Segment[1]	USInt
■ Segment[2]	USInt
■ Segment[3]	USInt
■ Segment[4]	USInt
■ Segment[5]	USInt
■ Segment[6]	USInt
■ Segment[7]	USInt

9. Add the “Banner\_TL50\_PD” function to an OB ladder. Link the “Process Data Out” to the raw Process Data variable from step 5. Link “TL50 PDO” to the parsed Process Data variable from step 7.



10. The final step is to configure the IO-Link output control. This is done by sending a 1 to Port Control and a 2 to Port Config. Both parameters are part of the tag created in step 6 “TL50 IOLM1 01 PDO”.

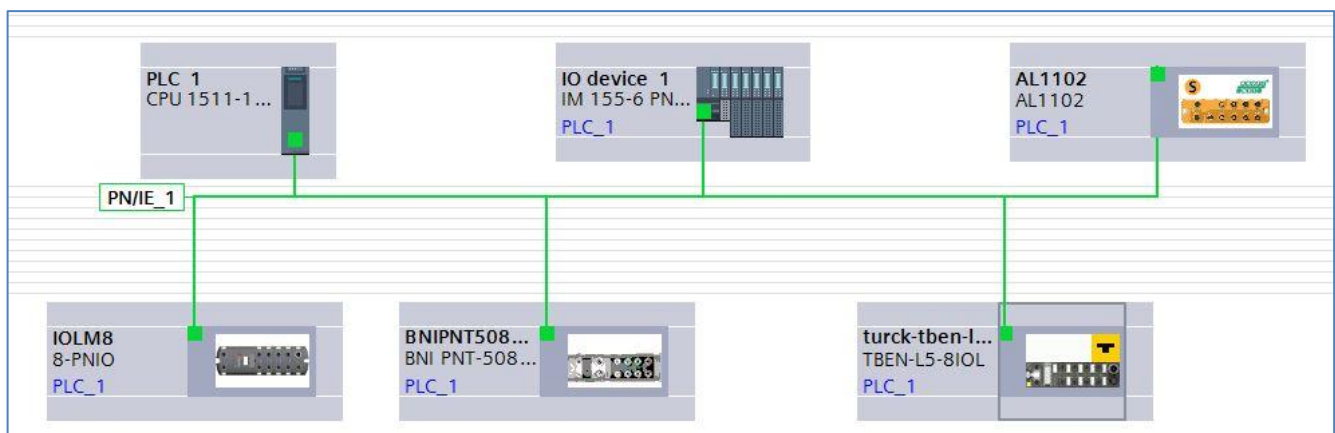
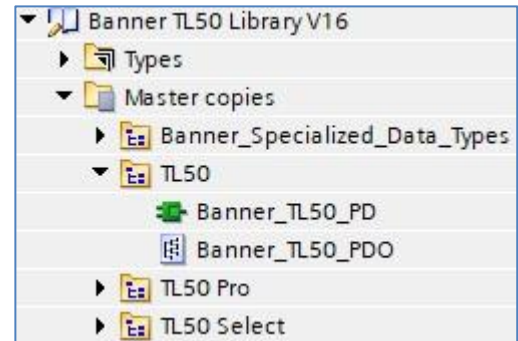


11. Process Data setup is complete.  
12. Compile and download the configuration to the PLC, then go online. Open the “db” data block and click Monitor all. You can control the TL50 via the Process Data Out.

▼ TL50 IOLM1 01 PD	"Banner_TL50_PDO"		
▼ Segment	Array[1..7] of USInt		0=Off, 1=On, 2=Flash.
Segment[1]	USInt	1	0=Off, 1=On, 2=Flash.
Segment[2]	USInt	0	0=Off, 1=On, 2=Flash.
Segment[3]	USInt	0	0=Off, 1=On, 2=Flash.
Segment[4]	USInt	0	0=Off, 1=On, 2=Flash.
Segment[5]	USInt	0	0=Off, 1=On, 2=Flash.
Segment[6]	USInt	0	0=Off, 1=On, 2=Flash.
Segment[7]	USInt	0	0=Off, 1=On, 2=Flash.

### Setup of TL50 with other IO-Link Masters

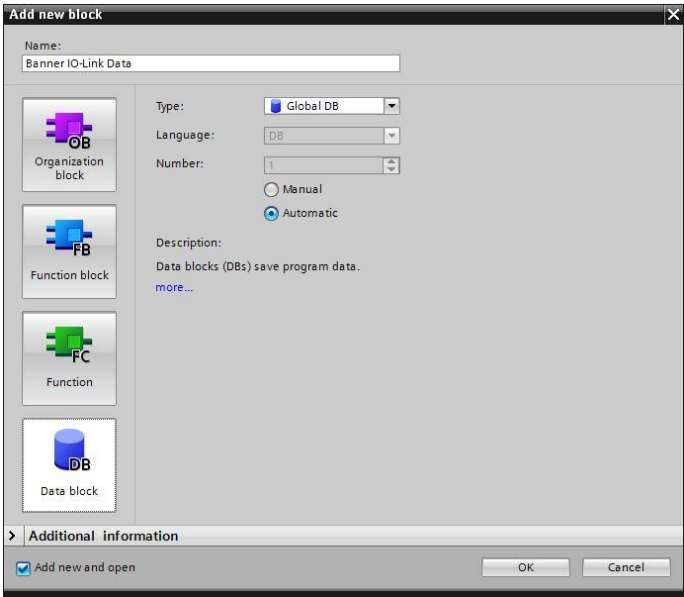
1. The Banner IO-Link Library will now be in the Global Library List. Expand the Master copies section.
2. Drag Banner\_TL50\_PD to the Program Blocks area under your PLC.
3. Drag the Banner\_TL50\_PDO to the PLC Data Types area under your PLC.
4. Go to Devices and networks to configure the system as necessary. Below is an example of what a configuration might look like. This example shows 5 different IO-Link Masters connected to the same PLC.



5. Click on the relevant device and configure the IO-Link Master as necessary. Refer to the documentation for the IO-Link Master. Recall that a TL50 requires 2 bytes of space for the Process Data.
6. Record the “Q” address where this TL50 Process Data is to be stored, as the address will be required in the next step. In this example, 2 bytes of Process Data Out for port 1 on the IO-Link Master will be stored in Q3 and Q4.
7. Go to PLC Tags. Add a new tag table, then create a new tag to represent the raw Process Data Out to be sent to the IO-Link Master. In this example, Tag table\_1 was created, then the tag “TL50 IOLM1 01 PD” was created using a Data Type of “Word”. This naming convention calls out the type of sensor in question as well as the specific IO-Link Master and port number where the sensor is connected. A different IO-Link Master might be named IOLM2 or IOLM3, for instance, and other specific sensors may be connected to different port numbers. The “Q” address found in step 6 is tied to this new tag.

TL50 IOLM1 01 outRaw	UInt	%QW3
----------------------	------	------

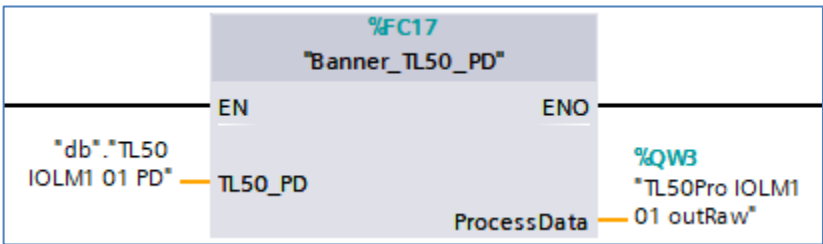
8. Go to Program blocks. Add a new Data block if necessary. In this example the new data block is named “Banner IO-Link Data”.



9. In the new data block, create a new tag to represent the parsed Process Data In for our TL50. The tag name again calls out the type of sensor, the IO-Link Master, and the port number. Use the data type “Banner\_TL50\_PDO” for the new tag.

Name	Data type
▼ Static	
■ ▼ TL50 IOLM1 01 PD	"Banner_TL50_PDO"
■ ▼ Segment	Array[1..7] of USInt
■ Segment[1]	USInt
■ Segment[2]	USInt
■ Segment[3]	USInt
■ Segment[4]	USInt
■ Segment[5]	USInt
■ Segment[6]	USInt
■ Segment[7]	USInt

10. Add the “Banner\_TL50\_PD” function to an OB ladder. Link the “ProcessData” to the raw Process Data variable from step 7. Link “TL50\_PD” to the parsed Process Data variable from step 9.



11. Process Data setup is complete.
12. Compile and download the configuration to the PLC, then go online. Open the “Banner IO-Link Data” data block and click Monitor all. You should see parsed TL50 Process Data, like that shown below.

▼ TL50 IOLM1 01 PD	"Banner_TL50_PDO"		
■ ▼ Segment	Array[1..7] of USInt		0=Off, 1=On, 2=Flash.
■ Segment[1]	USInt	1	0=Off, 1=On, 2=Flash.
■ Segment[2]	USInt	0	0=Off, 1=On, 2=Flash.
■ Segment[3]	USInt	0	0=Off, 1=On, 2=Flash.
■ Segment[4]	USInt	0	0=Off, 1=On, 2=Flash.
■ Segment[5]	USInt	0	0=Off, 1=On, 2=Flash.
■ Segment[6]	USInt	0	0=Off, 1=On, 2=Flash.
■ Segment[7]	USInt	0	0=Off, 1=On, 2=Flash.



## Appendix A

## TL50 Process Data

The TL50 has 2 bytes of Process Data Out, as shown below.

ProcessDataOut "Process Data Out" id=V_Pd_OutSegState									
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	0	2-bit UInteger	0 = Off, 1 = On, 2 = Flash					Segment 1 - off/on/flash	
2	2	2-bit UInteger	0 = Off, 1 = On, 2 = Flash					Segment 2 - off/on/flash	
3	4	2-bit UInteger	0 = Off, 1 = On, 2 = Flash					Segment 3 - off/on/flash	
4	6	2-bit UInteger	0 = Off, 1 = On, 2 = Flash					Segment 4 - off/on/flash	
5	8	2-bit UInteger	0 = Off, 1 = On, 2 = Flash					Segment 5 - off/on/flash	
6	10	2-bit UInteger	0 = Off, 1 = On, 2 = Flash					Segment 6 - off/on/flash	
7	12	2-bit UInteger	0 = Off, 1 = On, 2 = Flash					Segment 7 - off/on/flash	

This Process Data is mapped to a specific group of PROFINET addresses. The 16-bits of Process Data actually encode 7 separate pieces of information, as shown above.

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