

WLF Process Data Function

February 28th, 2024

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 WLF LC25C light via an IO-Link Master from a Siemens PLC. The function covers parsing and display of the WLF Pro sensor Process Data Out.

Components

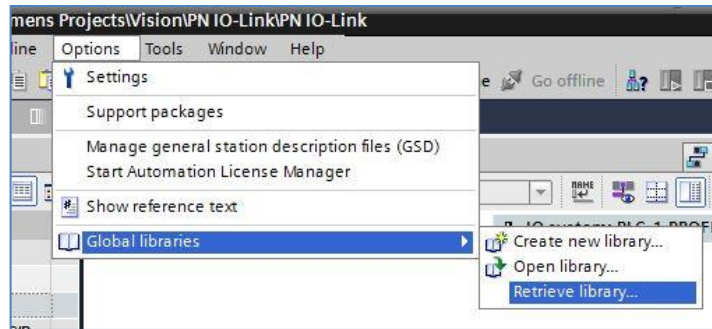
Banner WLS WLF Library v15.zal15

NOTE: The WLF, depending on length, can draw more current than a typical IO-Link Master can provide. A splitter arrangement may be necessary, where the light is powered from a different source.

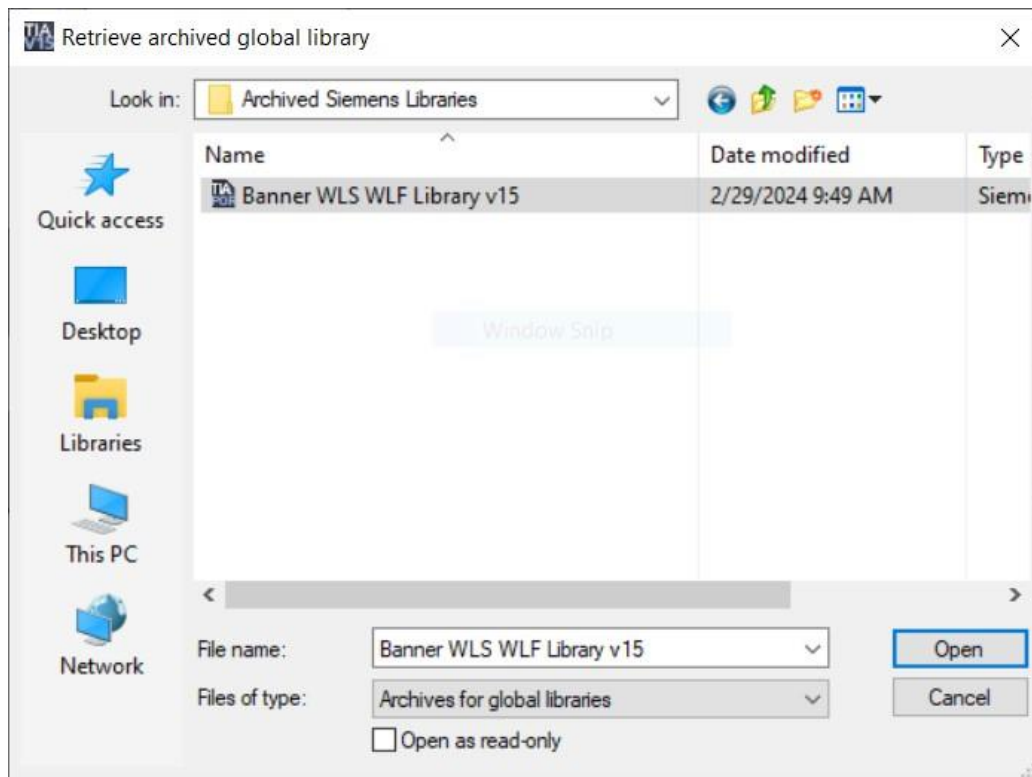
There are two methods for the process data. The first is used when creating a connection to Banner's IO-Link masters. The second set of instructions are for systems using other manufacturer's IO-Link masters.

Installation Instructions

1. Open a project.
2. Go to Options > Global Libraries > Retrieve Library.



3. Select the Banner WLS WLF Library. Click Open.



4. The library is now accessible in the Libraries tab.
5. Go to page 3 for Banner IO-Link and to page 7 for all other IO-Link Masters.

Setup of WLF with a Banner DXMR90-4K

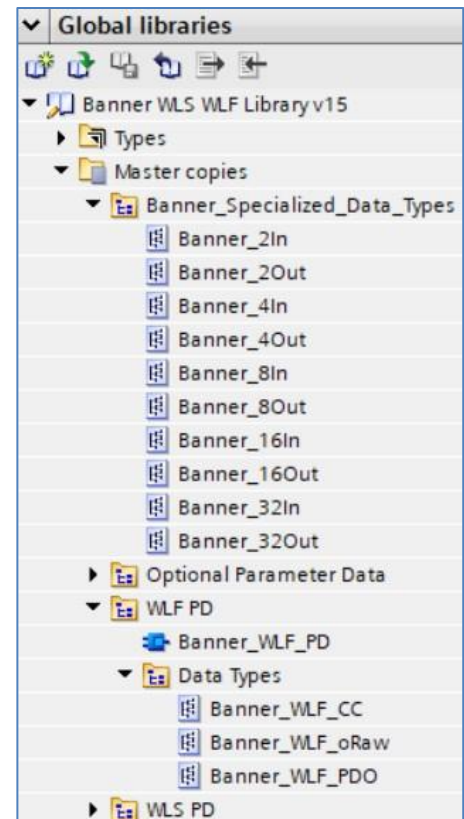
1. Go to Device and Networks to configure the DXMR90-4K. Add the DXMR90-4K if it has yet to be added to the system.
2. Add Banner IO-Link Master Info to Slot 1. This sets the DXMR90-4K for IO-Link mode.

Banner IO-Link Master Info_1	0	1	1...9	Banner IO-Link Master Info
------------------------------	---	---	-------	----------------------------

3. Open the IO-Link Generic Devices and select the proper module. The 32/32 byte is required for Pro. Make note of the Q address for Slot 2 which represents Port 1. Slot 2 starts are 1 for outputs. The other number needed is Q3. The data for the port starts at that point (I3). The previous two bytes Port Control.

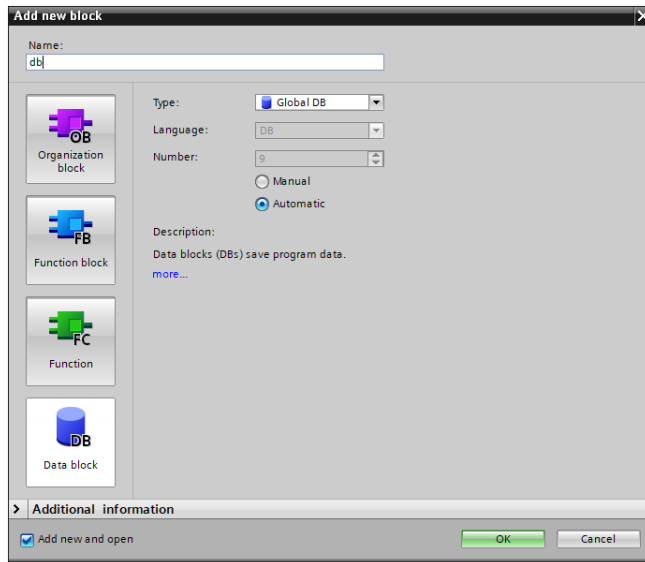
IO-Link In/Out 32/32 Byte + Status_1	0	2	10...45	1...46	IO-Link In/Out 32/32 Byte + Status
--------------------------------------	---	---	---------	--------	------------------------------------

4. Drag the necessary tag from IOLM_Control > Banner > Banner_Specialized_Data_Types. The tag used in this example is "Banner_32out". This tag represents the full raw process data along with port status information.
5. Drag the necessary files from the WLF PD Folder.
 - a. Move Banner_WLF_CC, Banner_WLF_oRaw, and Banner_WLF_PDO to the PLC Data Types area.
 - b. Move Banner_WLF_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 "WLF IOLM1 01 PDO" was created using a Data Type of "Banner_32out". This naming convention calls out the type of device in question as well as the specific IO-Link Master and port number to which 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 "WLF IOLM1 01 outRaw" and uses the "Q" address found in step 2 (%Q3). This is the tag that will be used in the Function block.



Name	Data type	Address
▶ WLF IOLM1 01 PDO	"Banner_32Out"	%Q1.0
▶ WLF IOLM1 01 outRaw	"Banner_WLF_oRaw"	%Q3.0

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 Output for our Pro. The tag name again calls out the type of sensor, the IO-Link Master, and the port number. Use the data type “Banner_WLF_PDO” for the new tag.

Name	Data type
▼ Static	
▼ WLF IOLM1 01 PD	"Banner_WLF_PDO"
▶ 0-SegMode	Array[1..10] of USInt
▶ 1-RunMode	"Banner_WLF_CC"
2-LevelMode	UInt
3-DimBlendMode	UInt
4-GaugeMode	UInt
▶ 5-LED Color	Array[1..64] of USInt

9. Add the “Banner_WLF_PD” function to an OB ladder. Link the “PD” to the raw process data variable from step 5. The tag name again calls out the type of device, IO-Link Master, and the port number. Use the variable was called “WLF IOLM1 01 outRaw” in this example. The “WLF PDO” needs to be linked to the variable created in step 7. It was called “WLF IOLM1 01 PD” for this example.

The last variable, “Operational Mode”, allows the function to correctly interpret the Process Data Out. In the case of the WLF Pro , there are six user-selected modes for the Process Data Out. This function needs to know what choice has been made in the WLF Pro for this Operational Mode variable.

There are two ways to achieve this goal. We can simply type in the correct number for Operational Mode (see Fig. 1), or we can link this WLF Pro Process Data Function to the WLF Pro Parameter Data Function Block (see Fig. 2). See Appendix A for more information about WLF Pro Process Data Out.

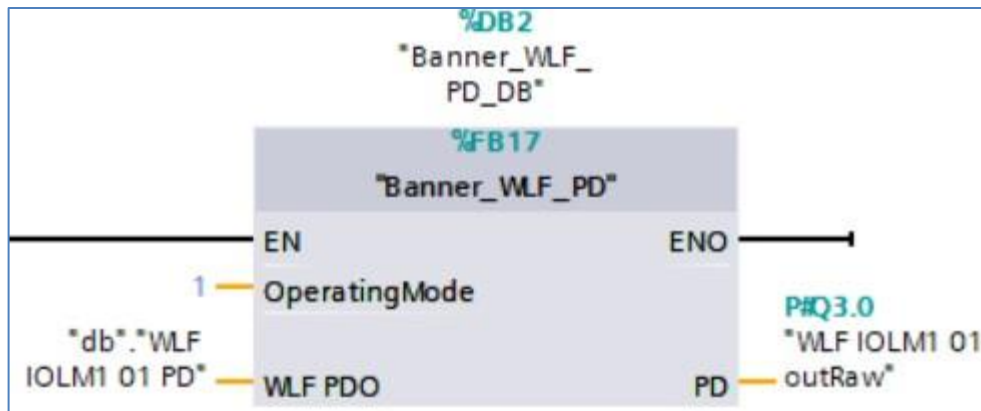


Figure 1: Hand type correct number for Operational Mode

NOTE: if you type in the incorrect number (i.e. it does not match the work light's current Operational Mode configuration) you will get incorrectly displayed Process Data Out information.

Operational Mode: the options here are "0" (Segment Mode; on/off/flash/animation state for up to 10 segments plus audible), "1" (Run Mode; a situation where the entire work light acts as one device), "2" (Level Mode; where the entire work light behaves as a level indicator), "3" (Dim and Blend Mode; where the process data controls the amount of blending between colors), "4" (Gauge Mode; where the entire work light changes states based on the Gauge settings), and "5" (Advanced Mode; full RGB control off all LEDs), The default is "1".

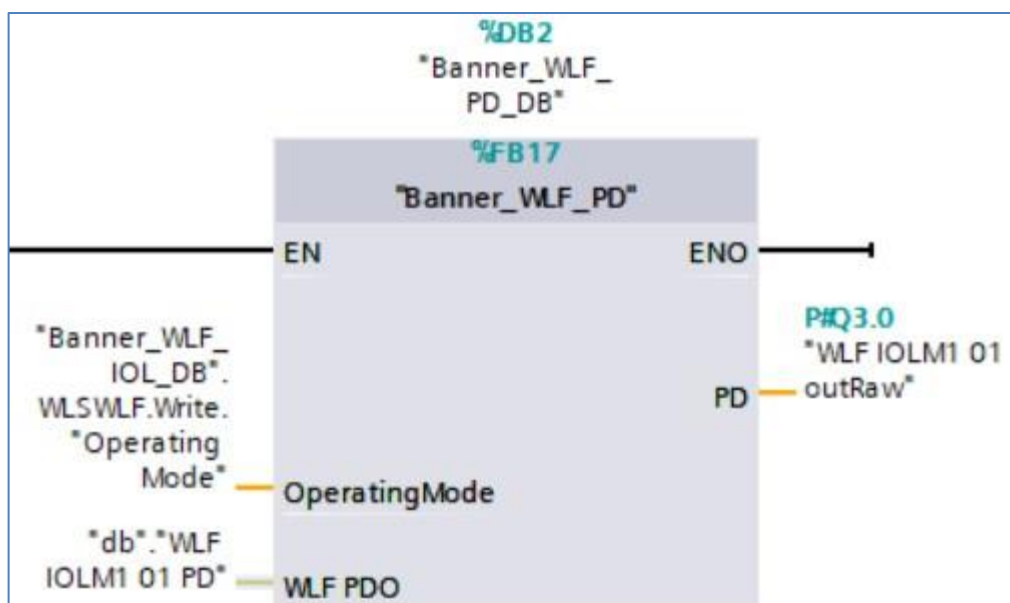
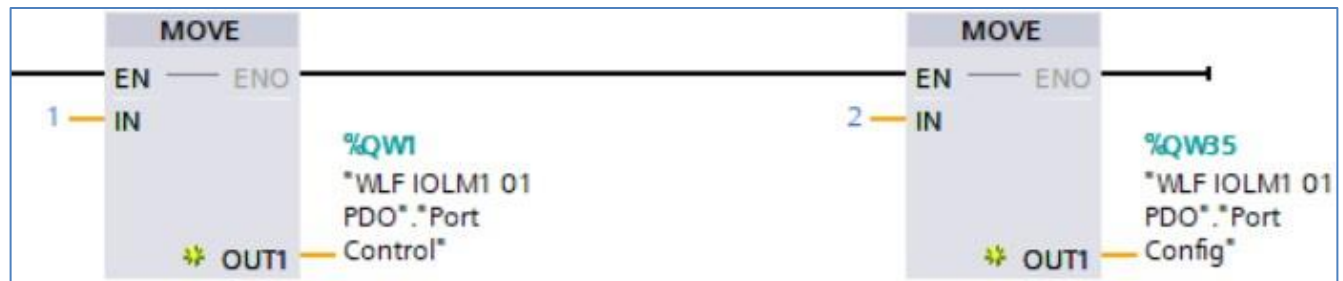


Figure 2: Linking Operational Mode variable to WLF Pro Parameter Data Function Block

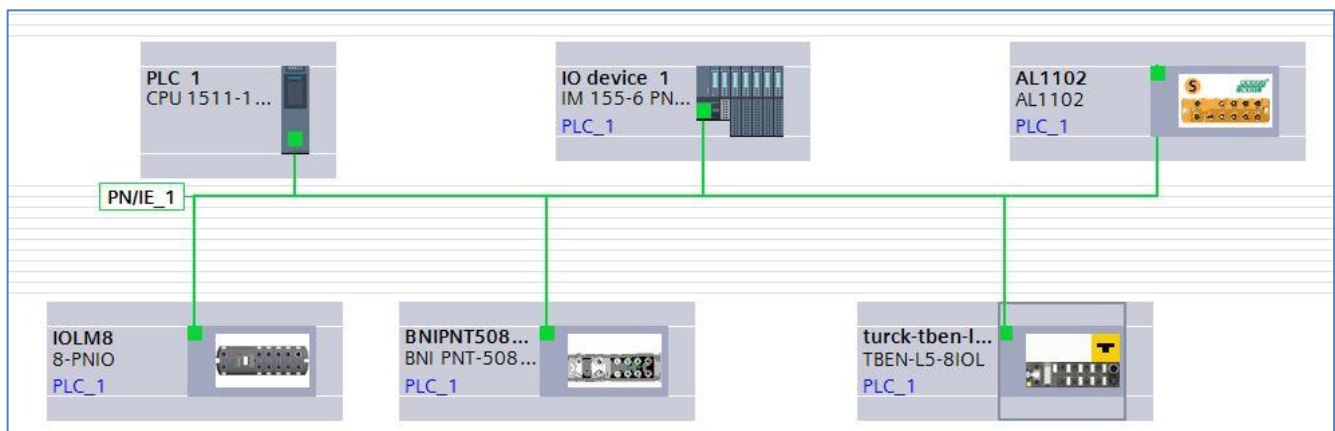
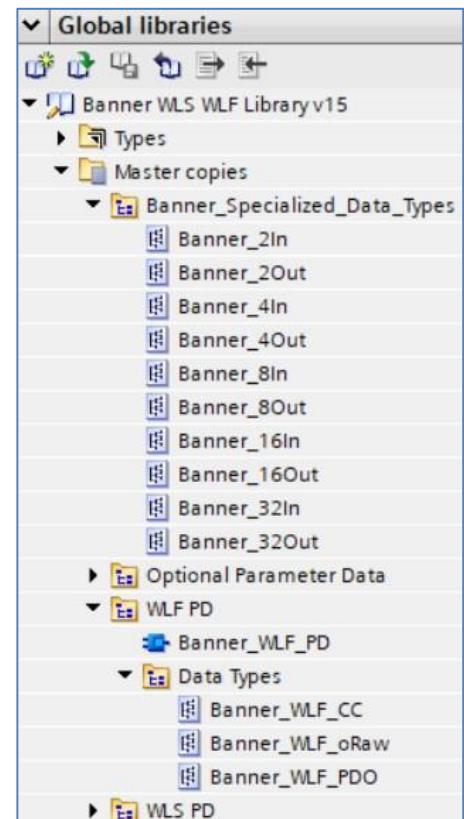
10. The final step is to configure the IO-Link output control. This is done by sending a 1 to Port Control and a 2 to Port Config. Both parameters are part of the tag created in step 6 “WLF IOLM1 01 PDO”.



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. The WLF Pro can be controlled now.

Setup of WLF Pro with other IO-Link Masters

1. The Banner WLF Library will now be in the Global Library List. Expand the Master copies section. The WLF Pro folder contains elements for both Process Data and Parameter Data connections to a WLF Pro device. As Process Data is the focus of this paper, we will concern ourselves with these items: Banner_WLFPro_CC, Banner_WLFPro_oRaw, and Banner_WLFPro_PDO.
2. Drag Banner_WLFPro_PD to the Program Blocks area under your PLC.
3. Drag the other items listed above 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.

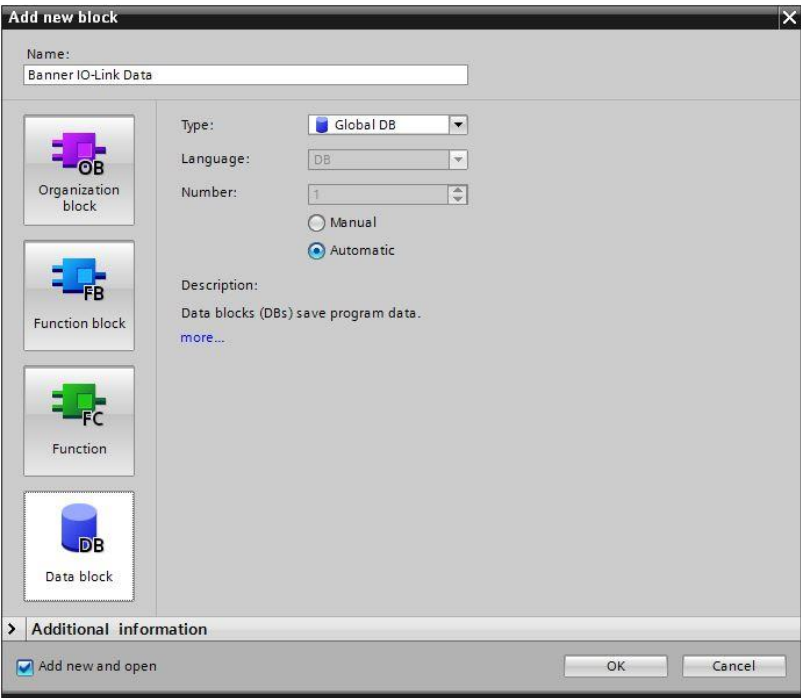


5. Click on the relevant device and configure the IO-Link Master as necessary. Refer to the documentation for the IO-Link Master. Recall that a WLF Pro requires 32 bytes of space for the Process Data.
6. Record the “Q” address where this WLF Pro Process Data is to be stored, as the address will be required in the next step. In this example, 32 bytes of Process Data Out for port 2 on the IO-Link Master will be stored in addresses starting at Q2.

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 “WLF IOLM1 01 PD” was created using a Data Type of “Banner_WLF_oRaw”. 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 IOLM1 or IOLM2, 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.

Name	Data type	Address
▶ WLF IOLM1 01 PDO	"Banner_32Out"	%Q1.0
▶ WLF IOLM1 01 outRaw	"Banner_WLF_oRaw"	%Q3.0

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 WLF. The tag name again calls out the type of sensor, the IO-Link Master, and the port number. Use the data type “Banner_WLF_PDO” for the new tag.

Name	Data type
▼ Static	
■ ▼ WLF IOLM1 01 PD	"Banner_WLF_PDO"

Add the “Banner_WLF_PD” function to an OB ladder. Link the “PD” to the raw Process Data variable from step 7. Link “WLF PDO” to the parsed Process Data variable created in this step.

The last variable, “Operating Mode”, allows the function to correctly interpret the Process Data Out. In the case of the WLF Pro, there are seven user-selected modes for the Process Data Out. This function needs to know what choice has been made in the WLF Pro for this Operating Mode variable.

There are two ways to achieve this goal. We can simply type in the correct number for Operating Mode (see Fig. 1), or we can link this WLF Pro Process Data Function to the WLF Pro Parameter Data Function Block (see Fig. 2). See Appendix A for more information about WLF Pro Process Data Out.

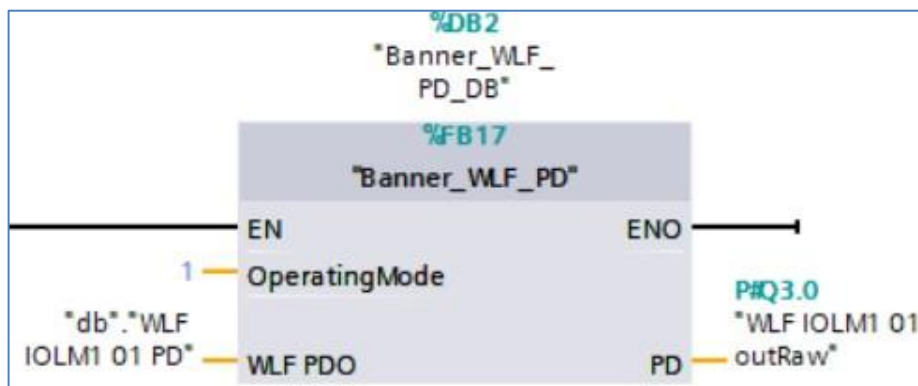


Figure 3: Hand type correct number for Operating Mode

NOTE: if you type in the incorrect number (i.e. it does not match the light’s current Operating Mode configuration) you will get incorrectly displayed Process Data Out information.

Operating Mode: the options here are “0” (Segment Mode), “1” (Run Mode), “2” (Level Mode), and “3” (Dim and Blend Mode), “4” (Gauge Mode), “5” (LED Mode), “6” (Demo Mode). The default is “0”.

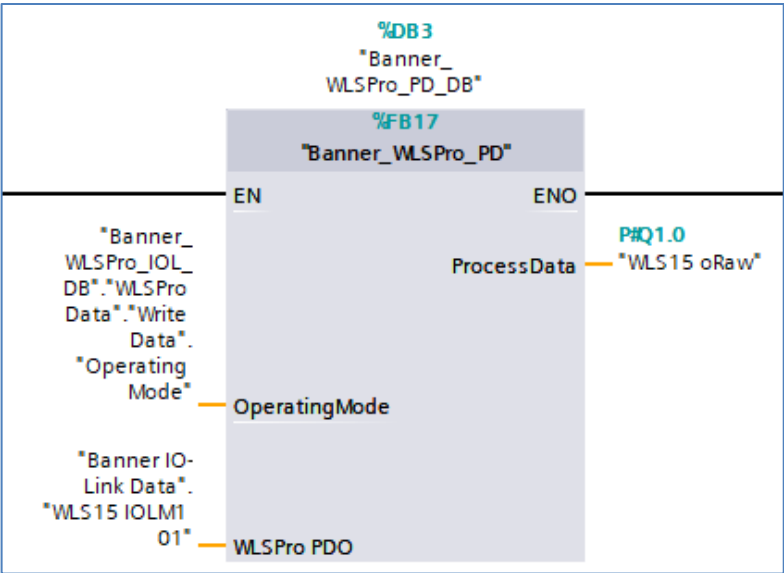


Figure 4: Linking Operating Mode variable to WLF Pro Parameter Data Function Block

- 10. Process Data setup is complete.
- 11. Compile and download the configuration to the PLC, then go online.
- 12. Open the “Banner IO-Link Data” data block and click Monitor all. You should see five pieces of data with a number in front of them. When the Operating Mode has a value of 0 to 4 the corresponding data is being used to control the light. Default Operating Mode is a value of 1.

Name	Data type
▼ Static	
■ ▼ WLF IOLM1 01 PD	"Banner_WLF_PDO"
■ ▶ 0-SegMode	Array[1..10] of USInt
■ ▶ 1-RunMode	"Banner_WLF_CC"
■ ▶ 2-LevelMode	UInt
■ ▶ 3-DimBlendMode	UInt
■ ▶ 4-GaugeMode	UInt
■ ▶ 5-LED Color	Array[1..64] of USInt

13. Each operating mode for the Process Data Out has its own tag array. If the WLF Pro device is in operating mode “1” (Run Mode), use the tags found under “1-RunMode” array, as seen below. If the operating mode is “2” (Level Mode), use the corresponding tags in “2-LevelMode” instead.

Name	Data type	Start value	Monitor value
▼ Static			
■ ▼ WLF IOLM1 01 PD	"Banner_WLF_PDO"		
■ ▶ 0-SegMode	Array[1..10] of USInt		
■ ▼ 1-RunMode	"Banner_WLF_CC"		
■ Animation	USInt	0	1
■ Color 1	USInt	0	9
■ Color 1 Intensity	USInt	0	0
■ Speed	USInt	0	0
■ Pulse Pattern	USInt	0	0
■ Color 2	USInt	0	0
■ Color 2 Intensity	USInt	0	0
■ Scroll Bounce	USInt	0	0
■ % Width of Col...	USInt	0	0
■ Direction	USInt	0	0
■ 2-LevelMode	UInt	0	0
■ 3-DimBlendMode	UInt	0	0
■ 4-GaugeMode	UInt	0	0

14. In the above example, Animation Type has a 1 in it which turns the WLF to Steady (Solid On) mode. Color 1 controls the color of the WLF, which in this case is Blue.

Appendix A**WLF Pro Process Data Out**

The WLF Pro has 32 bytes of Process Data Out, mapped into 7 different modes, as shown below.

This Process Data is mapped to a specific group of PROFINET addresses. The 256-bits of Process Data encode many separate pieces of information, as shown below.

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

First is the Segment mode (mode 0). This controls the basic on/off/flash/animation state of each segment.

ProcessDataOut "Process Data Out Segment Mode" id=V_Pd_OutSegment									
bit length: 256 data type: 256-bit Record									
subindex	bit offset	data type	allowed values	default value	acc. restr.	mod. other var.	excl. from DS	name	description
1	0	16-bit UInteger	0 = Off, 1 = On, 2 = Flash, 3 = Animation					Segment 1	The state of the segment. Related parameters defined in Segment Parameter Data
2	16	16-bit UInteger	0 = Off, 1 = On, 2 = Flash, 3 = Animation					Segment 2	The state of the segment. Related parameters defined in Segment Parameter Data
3	32	16-bit UInteger	0 = Off, 1 = On, 2 = Flash, 3 = Animation					Segment 3	The state of the segment. Related parameters defined in Segment Parameter Data
4	48	16-bit UInteger	0 = Off, 1 = On, 2 = Flash, 3 = Animation					Segment 4	The state of the segment. Related parameters defined in Segment Parameter Data
5	64	16-bit UInteger	0 = Off, 1 = On, 2 = Flash, 3 = Animation					Segment 5	The state of the segment. Related parameters defined in Segment Parameter Data
6	80	16-bit UInteger	0 = Off, 1 = On, 2 = Flash, 3 = Animation					Segment 6	The state of the segment. Related parameters defined in Segment Parameter Data
7	96	16-bit UInteger	0 = Off, 1 = On, 2 = Flash, 3 = Animation					Segment 7	The state of the segment. Related parameters defined in Segment Parameter Data
8	112	16-bit UInteger	0 = Off, 1 = On, 2 = Flash, 3 = Animation					Segment 8	The state of the segment. Related parameters defined in Segment Parameter Data
9	128	16-bit UInteger	0 = Off, 1 = On, 2 = Flash, 3 = Animation					Segment 9	The state of the segment. Related parameters defined in Segment Parameter Data
10	144	16-bit UInteger	0 = Off, 1 = On, 2 = Flash, 3 = Animation					Segment 10	The state of the segment. Related parameters defined in Segment Parameter Data

Here is the information for Run mode (mode 1).

ProcessDataOut "Process Data Out Run Mode" id=V_Pd_OutRunMode									
bit length: 256 data type: 256-bit Record									
subindex	bit offset	data type	allowed values	default value	acc. restr.	mod. other from var.	excl. from DS	name	description
1	0	8-bit UInteger	0 = Off, 1 = Steady, 2 = Flash, 3 = Two Color Flash, 4 = Two Color Shift, 5 = Ends Steady, 6 = Ends Flash, 7 = Scroll, 8 = Scroll from Center, 9 = Bounce, 10 = Bounce from Center, 11 = Intensity Sweep, 12 = Two Color Sweep, 13 = Color Spectrum					Animation	The Animation type
2	16	8-bit UInteger	0 = Green, 1 = Red, 2 = Orange, 3 = Amber, 4 = Yellow, 5 = Lime Green, 6 = Spring Green, 7 = Cyan, 8 = Sky Blue, 9 = Blue, 10 = Violet, 11 = Magenta, 12 = Rose, 13 = Daylight White (5000K), 14 = Custom 1, 15 = Custom 2, 16 = Incandescent White (2700K), 17 = Warm White (3000K), 18 = Fluorescent White (4100K), 19 = Neutral White (5700K), 20 = Cool White (6500K)					Color 1	The main color of the Animation. Custom Colors are defined in Parameter data
3	32	8-bit UInteger	0 = High, 1 = Low, 2 = Medium, 3 = Off, 4 = Custom					Color 1 Intensity	The Intensity of Color 1, Custom Intensity defined in Parameter Data
4	48	8-bit UInteger	0 = Medium, 1 = Fast, 2 = Slow, 3 = Custom Flash Rate					Speed	The speed of the Animation
5	64	8-bit UInteger	0 = Normal, 1 = Strobe, 2 = Three Pulse, 3 = SOS, 4 = Random					Pulse Pattern	The pattern of Animation
6	80	8-bit UInteger	0 = Green, 1 = Red, 2 = Orange, 3 = Amber, 4 = Yellow, 5 = Lime Green, 6 = Spring Green, 7 = Cyan, 8 = Sky Blue, 9 = Blue, 10 = Violet, 11 = Magenta, 12 = Rose, 13 = Daylight White (5000K), 14 = Custom 1, 15 = Custom 2, 16 = Incandescent White (2700K), 17 = Warm White (3000K), 18 = Fluorescent White (4100K), 19 = Neutral White (5700K), 20 = Cool White (6500K)					Color 2	The secondary color of the Animation. Only used if Animation has two colors. Custom Colors are defined in Parameter data
7	96	8-bit UInteger	0 = High, 1 = Low, 2 = Medium, 3 = Off, 4 = Custom					Color 2 Intensity	The Intensity of Color 2, Custom Intensity defined in Parameter Data
8	112	8-bit UInteger	0 = Solid, 1 = Tail, 2 = Ripple					Scroll/Bounce Style	The style of scrolling Segment
9	128	8-bit UInteger	1..100 = Percent Width of Color 1					Percent Width of Color 1	The size of scrolling Segment
10	144	8-bit UInteger	0 = Forward, 1 = Backward					Direction	The direction of Animation

Here is Level mode (mode 2).

ProcessDataOut "Process Data Out Level Mode" id=V_Pd_OutLevelMode									
bit length: 256 data type: 256-bit Record									
subindex	bit offset	data type	allowed values	default value	acc. restr.	mod. other var.	excl. from DS	name	description
1	0	16-bit UInteger						Level Mode Value	Value describing the level of the device, range determined in Level Mode Parameter Data

Here is Dim and Blend mode (mode 3).

ProcessDataOut "Process Data Out Dim and Blend Mode" id=TI_PD_Out_DimAndBlendMode									
bit length: 256 data type: 256-bit Record									
subindex	bit offset	data type	allowed values	default value	acc. restr.	mod. other var.	excl. from DS	name	description
1	0	16-bit UInteger						Dim and Blend Mode Value	Value describing the amount to blend between colors selected in Dim and Blend Mode Parameter Data

Here is Gauge mode (mode 4).

ProcessDataOut "Process Data Out Gauge Mode" id=V_PD_OutGaugeMode									
bit length: 256 data type: 256-bit Record									
subindex	bit offset	data type	allowed values	default value	acc. restr.	mod. other var.	excl. from DS	name	description
1	0	16-bit UInteger						Gauge Mode Value	Value describing the position of the main animation

Here is a small portion of the data included in LED mode (mode 5). This mode allows the user full control over every LED in the WLF.

ProcessDataOut "Process Data Out LED Mode" id=V_Pd_OutLedMode									
bit length: 256									
data type: 256-bit Record									
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 = Off, 1 = Green, 2 = Red, 3 = Orange, 4 = Amber, 5 = Yellow, 6 = Lime Green, 7 = Spring Green, 8 = Cyan, 9 = Sky Blue, 10 = Blue, 11 = Violet, 12 = Magenta, 13 = Rose, 14 = Daylight White (5000K), 15 = Custom 1					LED 1 Color	LED Mode Color control
2	4	4-bit UInteger	0 = Off, 1 = Green, 2 = Red, 3 = Orange, 4 = Amber, 5 = Yellow, 6 = Lime Green, 7 = Spring Green, 8 = Cyan, 9 = Sky Blue, 10 = Blue, 11 = Violet, 12 = Magenta, 13 = Rose, 14 = Daylight White (5000K), 15 = Custom 1					LED 2 Color	LED Mode Color control
3	8	4-bit UInteger	0 = Off, 1 = Green, 2 = Red, 3 = Orange, 4 = Amber, 5 = Yellow, 6 = Lime Green, 7 = Spring Green, 8 = Cyan, 9 = Sky Blue, 10 = Blue, 11 = Violet, 12 = Magenta, 13 = Rose, 14 = Daylight White (5000K), 15 = Custom 1					LED 3 Color	LED Mode Color control
4	12	4-bit UInteger	0 = Off, 1 = Green, 2 = Red, 3 = Orange, 4 = Amber, 5 = Yellow, 6 = Lime Green, 7 = Spring Green, 8 = Cyan, 9 = Sky Blue, 10 = Blue, 11 = Violet, 12 = Magenta, 13 = Rose, 14 = Daylight White (5000K), 15 = Custom 1					LED 4 Color	LED Mode Color control
5	16	4-bit UInteger	0 = Off, 1 = Green, 2 = Red, 3 = Orange, 4 = Amber, 5 = Yellow, 6 = Lime Green, 7 = Spring Green, 8 = Cyan, 9 = Sky Blue, 10 = Blue, 11 = Violet, 12 = Magenta, 13 = Rose, 14 = Daylight White (5000K), 15 = Custom 1					LED 5 Color	LED Mode Color control
6	20	4-bit UInteger	0 = Off, 1 = Green, 2 = Red, 3 = Orange, 4 = Amber, 5 = Yellow, 6 = Lime Green, 7 = Spring Green, 8 = Cyan, 9 = Sky Blue, 10 = Blue, 11 = Violet, 12 = Magenta, 13 = Rose, 14 = Daylight White (5000K), 15 = Custom 1					LED 6 Color	LED Mode Color control
7	24	4-bit UInteger	0 = Off, 1 = Green, 2 = Red, 3 = Orange, 4 = Amber, 5 = Yellow, 6 = Lime Green, 7 = Spring Green, 8 = Cyan, 9 = Sky Blue, 10 = Blue, 11 = Violet, 12 = Magenta, 13 = Rose, 14 = Daylight White (5000K), 15 = Custom 1					LED 7 Color	LED Mode Color control
8	28	4-bit UInteger	0 = Off, 1 = Green, 2 = Red, 3 = Orange, 4 = Amber, 5 = Yellow, 6 = Lime Green, 7 = Spring Green, 8 = Cyan, 9 = Sky Blue, 10 = Blue, 11 = Violet, 12 = Magenta, 13 = Rose, 14 = Daylight White (5000K), 15 = Custom 1					LED 8 Color	LED Mode Color control
9	32	4-bit	0 = Off, 1 = Green, 2 = Red, 3 = Orange, 4 = Amber, 5 = Yellow, 6 = Lime Green, 7 = Spring Green, 8 = Cyan, 9 = Sky Blue, 10 = Blue,					LED 9	LED Mode Color

Demo mode (mode 6), has no defined Process Data Out.