

R95C MDR Process Data Function

January 12th, 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 to a Banner R95C MDR light via an IO-Link Master from Siemens PLC. The function covers parsing and display of the R95C MDR sensor Process Data.

Components

Banner R95C MDR v16.zal16

There are two methods for 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 manufacturers' 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.



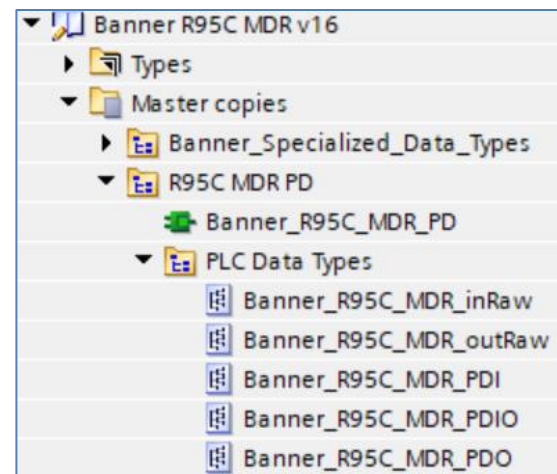
3. Switch the “Files of type” to Compressed libraries. Go to the location of the compressed library.
4. Press the Open button and the library will be uncompressed and opened.
5. The library is now accessible in the Libraries tab in v16 or greater.

Setup of R95C MDR 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 16/16 byte is required for R95C MDR. Make note of the I and Q address for Slot 2 which represents Port 1 (%I10 & %Q1).

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 16/16 Byte + Status...	0	2	10...29	1...30	IO-Link In/Out 16/16 Byte + Status

4. Drag the necessary tag from Banner_Specialized_Data_Types. The tag used in this example is "Banner_16In" and "Banner_16out".
5. Drag the necessary files from the R95C MDR PD Folder.
 - a. Move Banner_R95C_MDR_inRaw, Banner_R95C_MDR_outRaw, Banner_R95C_MDR_PDI, Banner_R95C_MDR_PDIO, and Banner_R95C_MDR_RDO to the PLC Data Types area.
 - b. Move Banner_R95C_MDR_PD to the Program Blocks area.



6. Go to PLC Tags. Create four tags. Two tags are for the full data structure while the second set is a tag to represent the raw Process Data from the IO-Link Master. In this example, Tag table_1 was created, then the tag "MDR IOLM1 01 PDI" was created using a Data Type of "Banner_16In", while the "MDR IOLM1 01 PDO" created with Data Type of "Banner_16Out". 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 "I" and "Q" address found in step 3 (%I10 & %Q1) is tied to these tags. The second set created are "MDR IOLM1 01 iRaw" and "MDR IOLM1 01 oRaw". Link iRaw tag to the "I" address found in step 3 plus 4 (%I14) and the oRaw tag to the "Q" address found in step 3 plus 2 (%Q3).

Name	Data type	Address
▶ MDR IOLM1 01 PDI	"Banner_16In"	%I10.0
▶ MDR IOLM1 01 PDO	"Banner_16Out"	%Q1.0
▶ MDR IOLM1 01 iRaw	"Banner_R95C_MDR_inRaw"	%I14.0
▶ MDR IOLM1 01 oRaw	"Banner_R95C_MDR_outRaw"	%Q3.0

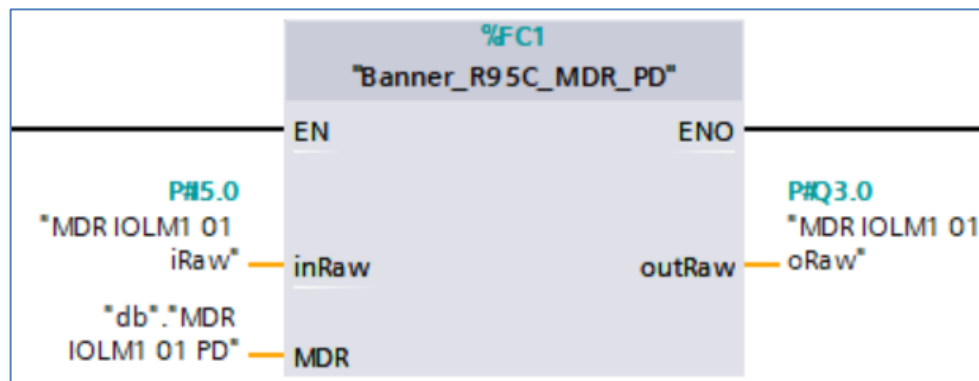
7. Go to Program blocks. Add a new Data block if necessary. In this example the new data block is named "db".
8. In the new data block, create a new tag to represent the parsed Process Data for our R95C MDR. The tag name again calls out the type of sensor, the IO-Link Master, and the port number. Use the data type "Banner_R95C_MDR_PDIO" for the new tag.

▼ MDR IOLM1 01 PD	"Banner_R95C_MDR_PDIO"
■ ► PDI	"Banner_R95C_MDR_PDI"
■ ► PDO	"Banner_R95C_MDR_PDO"

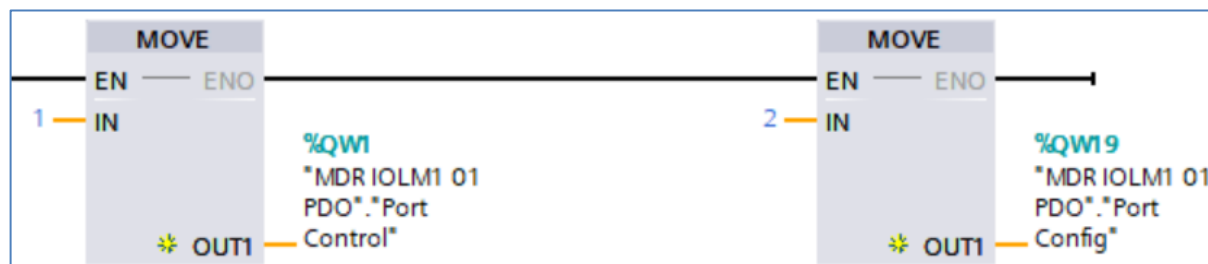
▼ PDI	"Banner_R95C_MDR_PDI"
■ ► Short Circuit	Array[1..4] of Bool
■ ► Overcurrent	Array[1..4] of Bool
■ OverVoltage	Bool
■ UnderVoltage	Bool
■ Block Temperature	Bool
■ Aux Power	Bool
■ ► Port Pin2 - Channel2	Array[1..4] of Bool
■ ► Port Pin4 - Channel1	Array[1..4] of Bool
■ ► Port Pin5 - Analog Value	Array[1..4] of Int

▼ PDO	"Banner_R95C_MDR_PDO"
■ ► Motor Current Fault Reset	Array[1..4] of Bool
■ ► Port Pin2 - Channel2	Array[1..4] of Bool
■ ► Port Pin4 - Channel1	Array[1..4] of Bool
■ ► Port Pin5 - Analog Value	Array[1..4] of Int

9. Add the “Banner_R95C_MDR_PD” function to an OB ladder. Link the “inRaw” to the tag “MDR IOLM1 01 iRaw” created from step 6. Link the “outRaw” to the tag “MDR IOLM1 01 oRaw” created from step 6. Link “MDR” to the tag “db.MDR IOLM1 01 PD” created from step 8.



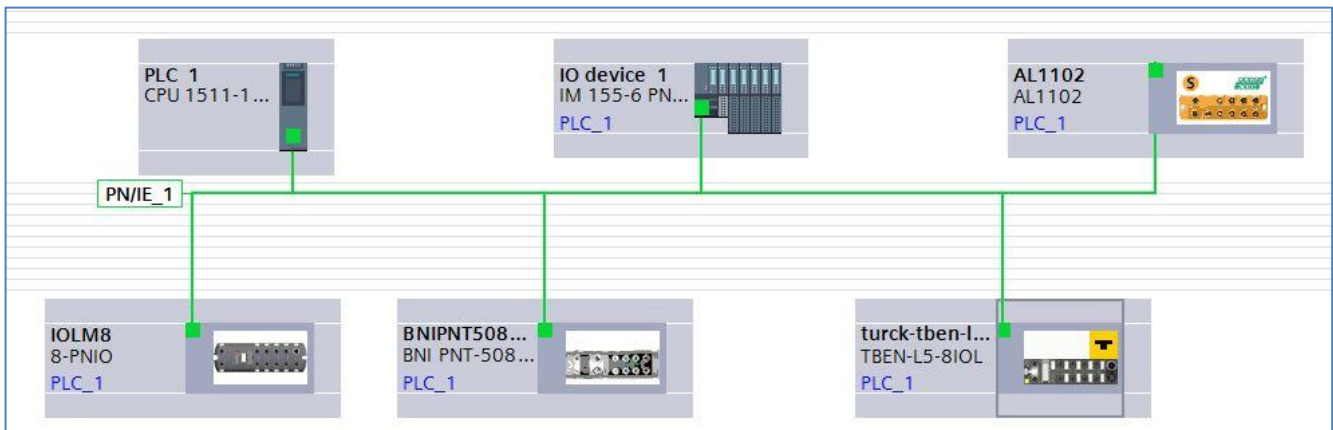
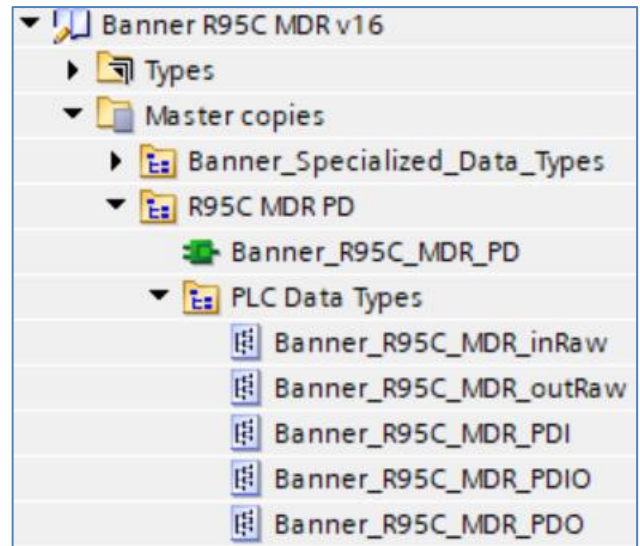
10. The final step is to configure the IO-Link output control. This is done by sending a 1 to Port Control and a 2 to Port Config. Both parameters are part of the tag created in step 6 “R95C MDR IOLM1 01 PDO”.



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 now control the R95C 4B4UI via the data located here.

Setup of R95C MDR with other IO-Link Masters

1. The Banner R95C MDR Library will now be in the Global Library List. Expand the Master copies section.
2. Drag Banner_R95C_MDR_PD to the Program Blocks area under your PLC.
3. Move Banner_R95C_MDR_inRaw, Banner_R95C_MDR_outRaw, Banner_R95C_MDR_PDI, Banner_R95C_MDR_PDIO, and Banner_R95C_MDR_PDO to the PLC Data Types area.
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.

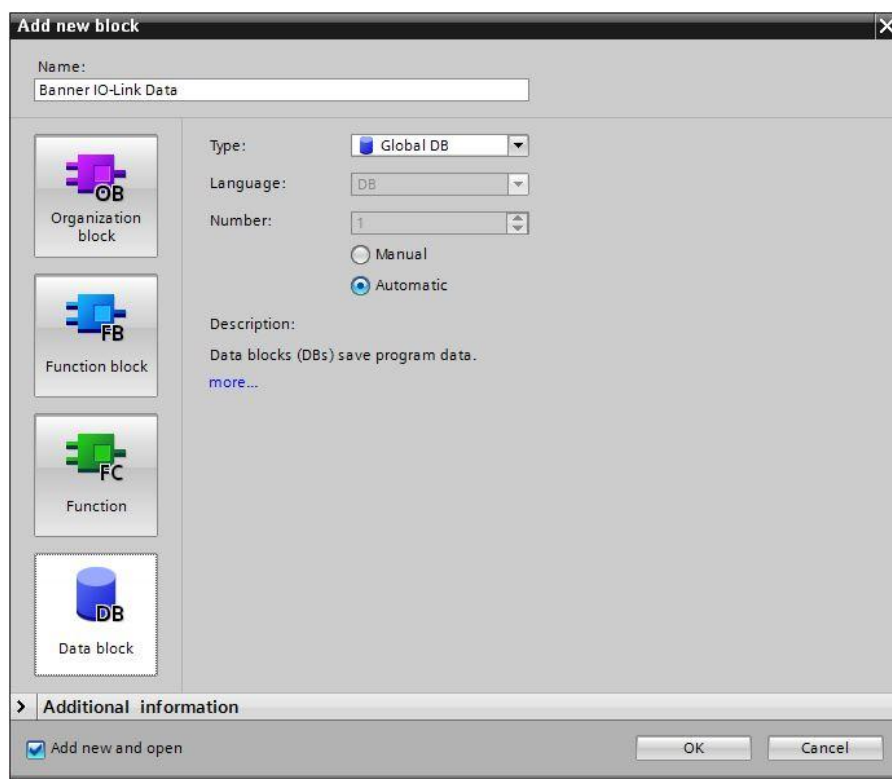


5. Click on the relevant device and configure the IO-Link Master as necessary. Refer to the documentation for the IO-Link Master. Recall that a R95C requires 11 bytes of space for the Process Data In and 10 bytes of space for the Process Data Out.
6. Record the “I” and “Q” address where this R95C Process Data is to be stored, as the address will be required in the next step. In this example, 16 bytes of Process Data In for port 1 on the IO-Link Master will be stored starting at I68 while the Process Data out starts at Q68.

7. Go to PLC Tags. Add a new tag table, then create a new tag to represent the raw Process Data to be sent to the IO-Link Master. In this example, Tag table_1 was created, then the tag “MDR IOLM1 01 iRaw” was created using a Data Type of “Banner_R95C_MDR_inRaw”. Also create a tag for the outputs called “MDR IOLM1 01 oRaw” was created using a Data Type of “Banner_R95C_MDR_outRaw”. 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. Reference IO-Link Documentation for addressing. This example uses %I68 and %Q68.

▶ MDR IOLM1 01 iRaw	"Banner_R95C_MDR_inRaw"	%I68.0
▶ MDR IOLM1 01 oRaw	"Banner_R95C_MDR_outRaw"	%Q68.0

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”.



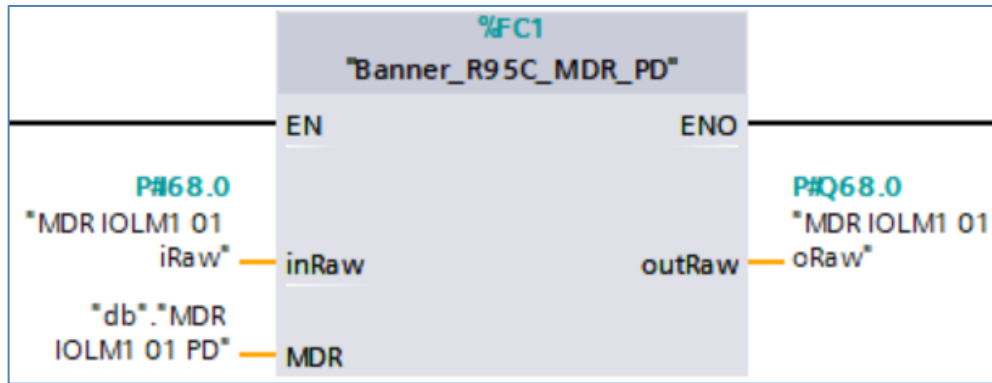
9. In the new data block, create a new tag to represent the parsed Process Data In for our R95C MDR. The tag name again calls out the type of sensor, the IO-Link Master, and the port number. Use the data type “Banner_R95C_MDR_PDIO” for the new tag.

▼ MDR IOLM1 01 PD	"Banner_R95C_MDR_PDIO"
■ ► PDI	"Banner_R95C_MDR_PDI"
■ ► PDO	"Banner_R95C_MDR_PDO"

▼ PDI	"Banner_R95C_MDR_PDI"
■ ► Short Circuit	Array[1..4] of Bool
■ ► Overcurrent	Array[1..4] of Bool
■ OverVoltage	Bool
■ UnderVoltage	Bool
■ Block Temperature	Bool
■ Aux Power	Bool
■ ► Port Pin2 - Channel2	Array[1..4] of Bool
■ ► Port Pin4 - Channel1	Array[1..4] of Bool
■ ► Port Pin5 - Analog Value	Array[1..4] of Int

▼ PDO	"Banner_R95C_MDR_PDO"
■ ► Motor Current Fault Reset	Array[1..4] of Bool
■ ► Port Pin2 - Channel2	Array[1..4] of Bool
■ ► Port Pin4 - Channel1	Array[1..4] of Bool
■ ► Port Pin5 - Analog Value	Array[1..4] of Int

10. Add the “Banner_R95C_MDR_PD” function to an OB ladder. Link the “inRaw” and “ouRaw” to the raw Process Data variable from step 7. Link “MDR” 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 R95C MDR Process Data In, like that shown below.

Appendix A

R95C MDR Process Data Out

The R95C MDR has 11 bytes of Process Data In and 10 bytes of Process Data Out.

Partially Shown Process Data In (partially shown PDI)

ProcessDataIn "Process Data Input" id=PI_PDIn

bit length: 88

data type: 88-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	83	Boolean	false = Inactive, true = Active					MDR 4 Short Circuit	MDR 4 Short Circuit
2	82	Boolean	false = Current below limit, true = Current above limit					MDR 4 Overcurrent	MDR 4 Overcurrent
3	81	Boolean	false = Inactive, true = Active					MDR 3 Short Circuit	MDR 3 Short Circuit
4	80	Boolean	false = Current below limit, true = Current above limit					MDR 3 Overcurrent	MDR 3 Overcurrent
5	79	Boolean	false = Inactive, true = Active					MDR 2 Short Circuit	MDR 2 Short Circuit
6	78	Boolean	false = Current below limit, true = Current above limit					MDR 2 Overcurrent	MDR 2 Overcurrent
7	77	Boolean	false = Inactive, true = Active					MDR 1 Short Circuit	MDR 1 Short Circuit
8	76	Boolean	false = Current below limit, true = Current above limit					MDR 1 Overcurrent	MDR 1 Overcurrent
9	75	Boolean	false = Voltage within Range, true = Voltage above Range					MDR OverVoltage	MDR OverVoltage
10	74	Boolean	false = Voltage within Range, true = Voltage below Range					MDR UnderVoltage	MDR UnderVoltage
11	73	Boolean	false = Temp below limit, true = Temp above limit					Block Temperature	Block Temperature
12	72	Boolean	false = No Aux Power, true = Aux Power Present					Aux Power	Aux Power

ProcessDataOut "Process Data Out" id=PO_PDout

bit length: 80

data type: 80-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	75	Boolean	false = No action, true = Reset					Motor 4 Current Fault Reset	Motor 4 Current Fault Reset
2	74	Boolean	false = No action, true = Reset					Motor 3 Current Fault Reset	Motor 3 Current Fault Reset
3	73	Boolean	false = No action, true = Reset					Motor 2 Current Fault Reset	Motor 2 Current Fault Reset
4	72	Boolean	false = No action, true = Reset					Motor 1 Current Fault Reset	Motor 1 Current Fault Reset
5	71	Boolean	false = Inactive, true = Active					Port 4 Pin2-Channel 2 Control	Port 4 Pin2-Channel 2 Control
6	70	Boolean	false = Inactive, true = Active					Port 4 Pin4-Channel 1 Control	Port 4 Pin4-Channel 1 Control
7	69	Boolean	false = Inactive, true = Active					Port 3 Pin2-Channel 2 Control	Port 3 Pin2-Channel 2 Control
8	68	Boolean	false = Inactive, true = Active					Port 3 Pin4-Channel 1 Control	Port 3 Pin4-Channel 1 Control
9	67	Boolean	false = Inactive, true = Active					Port 2 Pin2-Channel 2 Control	Port 2 Pin2-Channel 2 Control
10	66	Boolean	false = Inactive, true = Active					Port 2 Pin4-Channel 1 Control	Port 2 Pin4-Channel 1 Control
11	65	Boolean	false = Inactive, true = Active					Port 1 Pin2-Channel 2 Control	Port 1 Pin2-Channel 2 Control
12	64	Boolean	false = Inactive, true = Active					Port 1 Pin4-Channel 1 Control	Port 1 Pin4-Channel 1 Control
13	48	16-bit Integer						Port 4 Pin 5 - Analog Output Control (mV)	Port 4 Pin 5 - Analog Output Control (mV)
14	32	16-bit Integer						Port 3 Pin 5 - Analog Output Control (mV)	Port 3 Pin 5 - Analog Output Control (mV)
15	16	16-bit Integer						Port 2 Pin 5 - Analog Output Control (mV)	Port 2 Pin 5 - Analog Output Control (mV)
16	0	16-bit Integer						Port 1 Pin 5 - Analog Output Control (mV)	Port 1 Pin 5 - Analog Output Control (mV)