



ZMX AOI or UDT Guide

4/13/2023

This document covers the installation of a Banner ZMX in Rockwell Studio 5000. There are two possible paths. There is an AOI (Add-On Instruction) that allows for full data collection and control and there is a UDT (User-Defined Tags) structure that allows for a much smaller footprint on the PLC but requires more manual coding. Follow the instructions for the path that works best for the application.

Components

Banner_ZMX_Control_AOI.L5X

OR

Banner_ZMX_Data_DataType

UDT Packaged with the Banner_ZMX_Control_AOI

Banner_ZMX_Data

Banner_ZMX_DataIn

Banner_ZMX_DataOut

Banner_ZMX_Full_Recipe

Banner_ZMX_Recipe_Parameters

UDT Packaged in the Banner_ZMX_Data_DataType

Banner_ZMX_Data

Banner_ZMX_DataIn

Banner_ZMX_DataOut

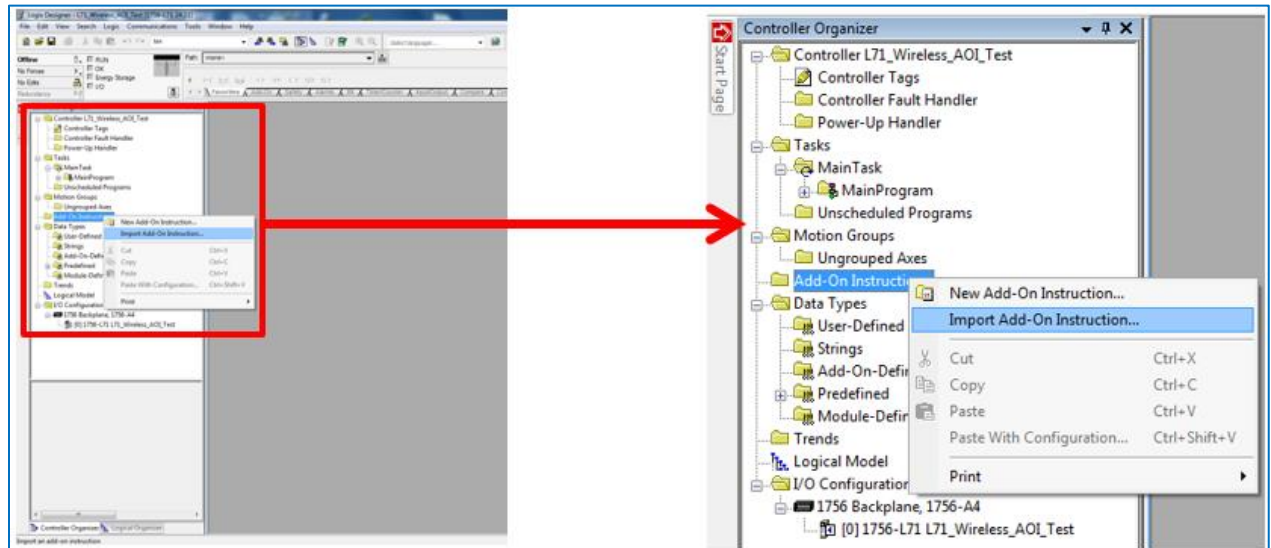
Contents

- 1. ZMX AOI Installation Process2
- 2. Configuring the ZMX Ethernet Connection4
- 3. Configuring the ZMX AOI5
- 4. Using the AOI.....9
- Appendix A. ZMX UDT 11
 - Installation Process 11
 - Setting Up the UDT 11

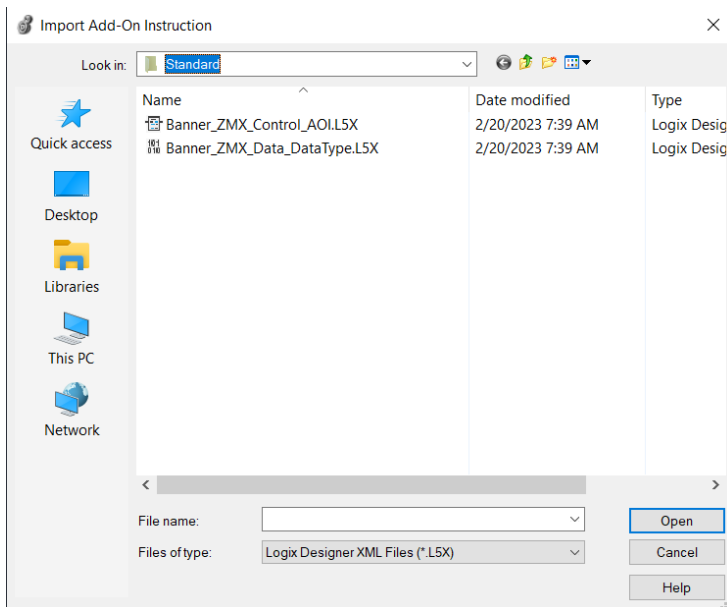
1. ZMX AOI Installation Process

This section describes how to install the AOI in Logix Designer software. Only follow these steps if the AOI is needed for the system. The AOI converts the raw data and allows for updating parameters for the ZMX.

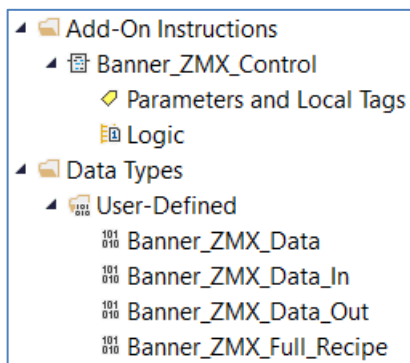
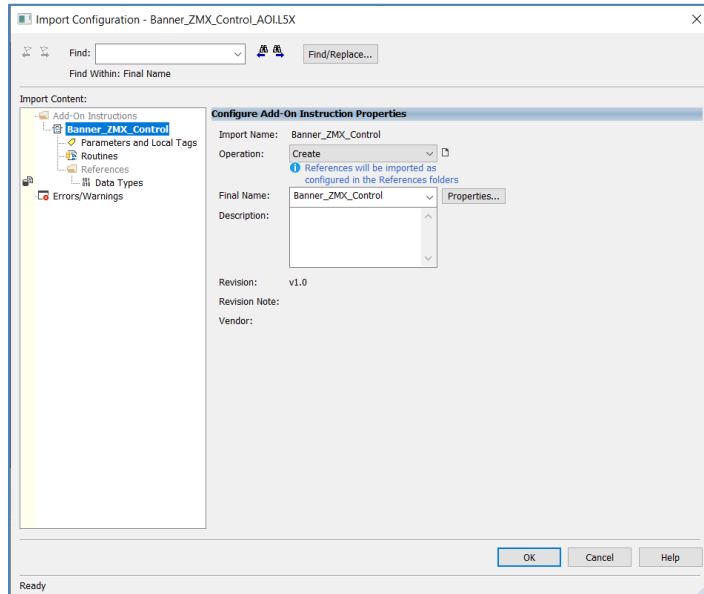
1. Open a project.
2. In the Controller Organizer window, right-click on the Add-On Instruction folder. Select the Import Add-On Instruction option.



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

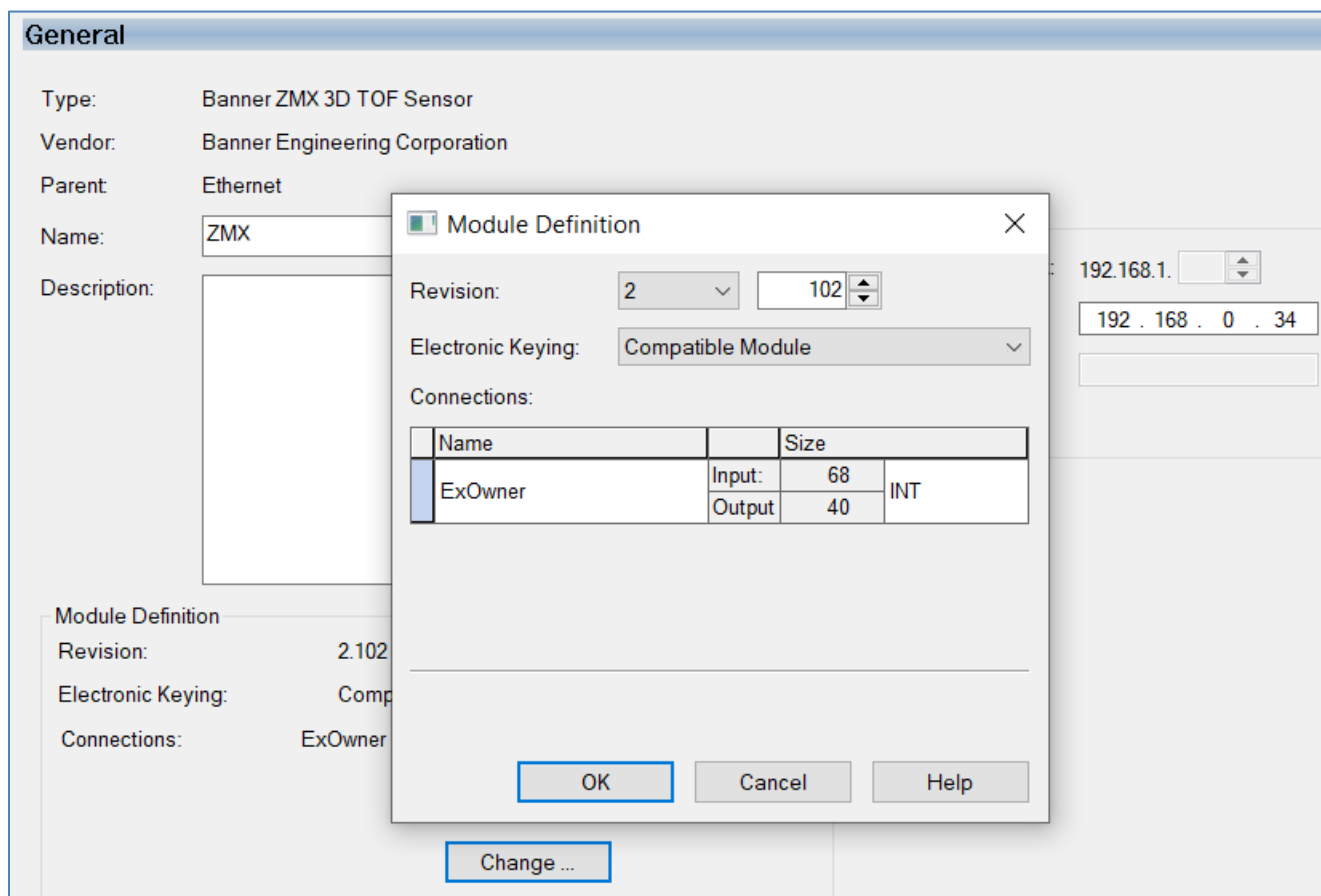


5. The AOI is added to the Controller Organizer window and should look like the picture at left.
6. AOI installation into the Logix Designer software is complete.

2. Configuring the ZMX Ethernet Connection

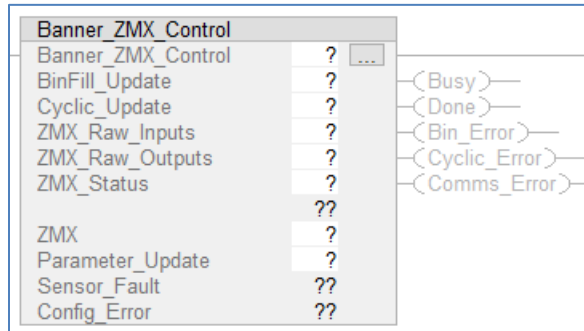
Make an EtherNet/IP connection to the ZMX sensor using the ZMX EDS. Install the ZMX EDS file if necessary.

Create an Ethernet communications module for the ZMX 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 ZMX data. Make sure that INTs are selected for the connection. See the ZMX User's Guide for more information.



3. Configuring the ZMX AOI

1. Add the “Banner_ZMX_Control” 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_ZMX_Control”. Click New Tag. Name the new tag. This example uses the name “ZMX_Status”.

Note that the Data Type is the User-Defined Data Type (UDT) entitled “Banner_ZMX_Control”. 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 "BinFill_Update" in the AOI. Click on "New Tag". Give the tag a name. This example uses the name "ActivateBin". Notice that the Data Type is "Bool". Click Create. Do the same steps for the next line labeled "Cyclic_Update" (this tag is named "ActivateCyc" in the example below). These two tags are used to update the ZMX with new parameters.

The image displays two side-by-side screenshots of the "New Tag" dialog box, illustrating the configuration for two different tags.

Left Dialog (ActivateBin):

- Name: ActivateBin
- Description: (Empty text area)
- Usage: <controller>
- Type: Base
- Alias For: (Empty dropdown)
- Data Type: BOOL
- Parameter Connection: (Empty dropdown)
- Scope: Test
- External Access: Read/Write
- Style: Decimal
- Constant: ☐
- Sequencing: ☐
- Open Configuration: ☐
- Open Parameter Connections: ☐

Right Dialog (ActivateCyc):

- Name: ActivateCyc
- Description: (Empty text area)
- Usage: <controller>
- Type: Base
- Alias For: (Empty dropdown)
- Data Type: BOOL
- Parameter Connection: (Empty dropdown)
- Scope: Test
- External Access: Read/Write
- Style: Decimal
- Constant: ☐
- Sequencing: ☐
- Open Configuration: ☐
- Open Parameter Connections: ☐

- The next two line are linked to the ZMX Input and Output data generated when the connection was created. In our example the items are "ZMX:I.Data" and "ZMX:O.Data", as the communications modules for the sensor was named "ZMX".
- The ZMX_Status is linked to the connection fault value that is part of the ZMX EDS file connection. This tells the AOI if there is a connection to the ZMX.

6. In the AOI, right-click on the question mark on the line labeled “ZMX”. Click New Tag. Name the new tag. This example uses the name “ZMX_Data”.

Note that the Data Type is the User-Defined Data Type (UDT) entitled “Banner_ZMX_Data”. This custom-made array of registers is specially built so the raw data is converted into usable data. Click Create to make the tag array.

New Tag

Name:

Description:

Usage:

Type:

Alias For:

Data Type:

Parameter Connection:

Scope:

External Access:

Style:

☐ Constant

☐ Sequencing

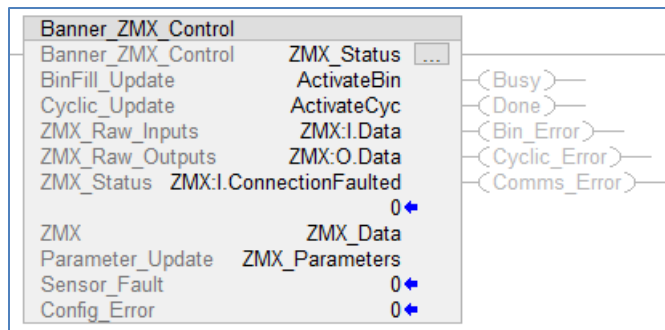
☐ Open Configuration

☐ Open Parameter Connections

7. Lastly, right-click on the question mark on the line labeled “Parameter_Update”. Click New Tag. Name the new tag. This example uses the name “ZMX_Parameters”. This tag is used to update the parameter settings for the ZMX.

Note that the Data Type is the User-Defined Data Type (UDT) entitled “Banner_ZMX_Full_Recipe”. This custom-made array of registers is specially built to move a full setup of parameter data to the ZMX.

8. The AOI should look like the below example.



The “Banner_ZMX_Control” AOI is now ready for use.

4. Using the AOI

The “Banner_ZMX_Control” Add-On Instruction has created a group of tags representing the ZMX data, broken out into its component parts.

Look in the Controller Tags to find the name you used above. This example used the name “ZMX_Data”. The tag array, seen below, has two options. They are Data_In for Inputs and Data_Out for Outputs. Each of these elements has an array of individual pieces of information instead of unlabeled bits. The Data_In and Data_Out examples only show a portion of the possible results. Data_In gives all the data that ZMX is processing. Unevenness is the last data item. This data tells how much of a difference there is between the maximum height and the minimum height.

◀ ZMX_Data	{...}	Banner_ZMX_Data
▶ ZMX_Data.Data_In	{...}	Banner_ZMX_Data_In
▶ ZMX_Data.Data_Out	{...}	Banner_ZMX_Data_Out
▶ ZMX_Data.Unevenness	0 Decimal	INT

ZMX Input Data (only a partial example)

◀ ZMX_Data	{...}	Banner_ZMX_Data
◀ ZMX_Data.Data_In	{...}	Banner_ZMX_Data_In
▶ ZMX_Data.Data_In.Input_Ack	0 Decimal	INT
▶ ZMX_Data.Data_In.Reserved1	0 Decimal	INT
▶ ZMX_Data.Data_In.Sensor_Fault_Code	0 Decimal	INT
▶ ZMX_Data.Data_In.Boot_Count	43 Decimal	INT
▶ ZMX_Data.Data_In.Up_Time	1697232 Decimal	DINT
▶ ZMX_Data.Data_In.Missed_Trigger_Count	0 Decimal	DINT
▶ ZMX_Data.Data_In.Reserved2	0 Decimal	INT
▶ ZMX_Data.Data_In.Reserved3	0 Decimal	INT
▶ ZMX_Data.Data_In.Reserved4	0 Decimal	INT
▶ ZMX_Data.Data_In.Reserved5	0 Decimal	INT
▶ ZMX_Data.Data_In.Reserved6	0 Decimal	INT
▶ ZMX_Data.Data_In.Reserved7	0 Decimal	INT
▶ ZMX_Data.Data_In.Reserved8	0 Decimal	INT
▶ ZMX_Data.Data_In.Reserved30	0 Decimal	INT
▶ ZMX_Data.Data_In.Frame_Number	6988 Decimal	DINT
▶ ZMX_Data.Data_In.Invalid_Pixel_Count	13162 Decimal	DINT
▶ ZMX_Data.Data_In.Saturated_Pixel_Count	1 Decimal	DINT
▶ ZMX_Data.Data_In.Conf_Error_Code	0 Decimal	INT

The AOI also allows for the efficient updating of ZMX parameters. The parameters are broken up into two separate groups (see Tables below). All data elements must have their values set to what is required for the system. The full data set is updated when activated. As an example, if ROI Length X, Y, and Z need to be adjusted, Fill Level, Peak Height, Discrete Out 1 & 2 Control Modes, and ROI Anchor Point X, Y, and Z also need to be correctly filled in (even if the values are the same as those found in the sensor). In this example the data parameters to be sent to the ZMX are set in the ZMX_Parameters tag.

Bin Fill Parameters (set "ActivateBin" bit to 1)
Fill Level Limit Setpoint %
Peak Height Limit Setpoint
Discrete Output 1 Control Mode
Discrete Output 2 Control Mode
ROI Anchor Point X
ROI Anchor Point Y
ROI Anchor Point Z
ROI Length X
ROI Length Y
ROI Length Z

Cyclic Parameters (set "ActivateCyc" bit to 1)
Trigger Mode
Trigger Period
Illumination Power
Pitch Angle
Roll Angle
Yaw Angle
Discrete IO Polarity

After the data is ready to be transferred, set the appropriate bit to 1. The AOI will act and set the bit back to 0 when done. This example uses "ActivateBin" and "ActivateCyc" for the two bits that update Bin Fill and Cyclic parameters respectively.

If there is an error during the update process the Bin_Error or Cyclic_Error will be turned ON.

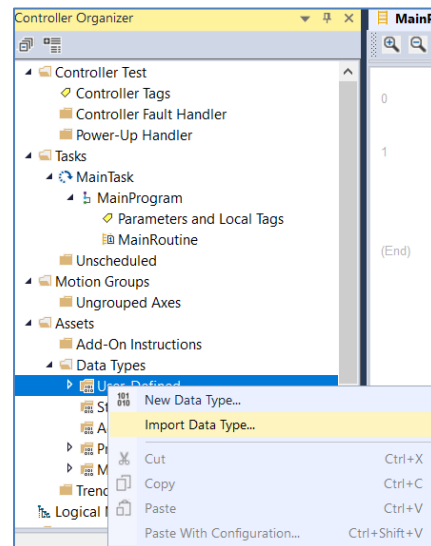
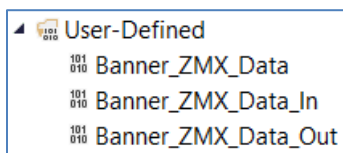
▲ ZMX_Parameters	{...}	Banner_ZMX_Full_Recipe
▶ ZMX_Parameters.Trigger_Mode	6 Decimal	INT
▶ ZMX_Parameters.Trigger_Period	250 Decimal	INT
▶ ZMX_Parameters.Illumination_Power	49 Decimal	INT
▶ ZMX_Parameters.Pitch_Angle	0 Decimal	INT
▶ ZMX_Parameters.Roll_Angle	0 Decimal	INT
▶ ZMX_Parameters.Yaw_Angle	0 Decimal	INT
▶ ZMX_Parameters.Discrete_IO_Polarity	0 Decimal	INT
▶ ZMX_Parameters.Fill_Level_Limit_Setpoint_Percent	85 Decimal	INT
▶ ZMX_Parameters.Reserved13	0 Decimal	INT
▶ ZMX_Parameters.Peak_Height_Limit_Setpoint	250 Decimal	INT
▶ ZMX_Parameters.Reserved14	0 Decimal	INT
▶ ZMX_Parameters.Discrete_Output_1_Control_Mode	0 Decimal	INT
▶ ZMX_Parameters.Discrete_Output_2_Control_Mode	0 Decimal	INT
▶ ZMX_Parameters.ROI_Anchor_X	0 Decimal	INT
▶ ZMX_Parameters.ROI_Anchor_Y	0 Decimal	INT
▶ ZMX_Parameters.ROI_Anchor_Z	800 Decimal	INT
▶ ZMX_Parameters.ROI_Length_X	400 Decimal	INT
▶ ZMX_Parameters.ROI_Length_Y	300 Decimal	INT
▶ ZMX_Parameters.ROI_Length_Z	300 Decimal	INT

Appendix A. ZMX UDT

Installation Process

This section describes how to install the ZMX UDT in Logix Designer software. The ZMX UDT can be used instead of the ZMX AOI to save PLC memory space.

1. Open a project.
2. Import the UDT for the ZMX.
3. Navigate to the correct file location and select the UDT to be installed. In this example the “Banner_ZMX_Data_DataType.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 UDT. Click the OK button to complete the import process.
5. Items will now be populated in the User-Defined area.



6. Installation instructions complete.

Setting Up the UDT

This section shows how to convert just the raw data for ZMX using UDTs. This approach gives a much smaller footprint than using the AOI. If parameter adjustment is required, the AOI is recommended, however. This section is not necessary when the AOI is being used.

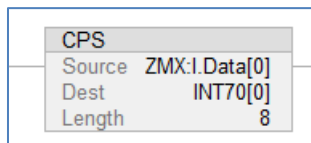
1. Start by creating a tag of the data type “Banner_ZMX_Data”. This example uses the name “ZMX_Data”.

Name	Alias For	Base Tag	Data Type
▸ ZMX_Data			Banner_ZMX_Data

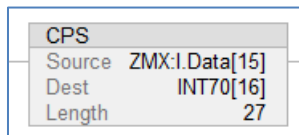
2. Next create another tag of the data type INT. Make the INT and array of 70 elements. This example uses the name “INT70”.

Name	Alias For	Base Tag	Data Type
▸ INT70			INT[70]

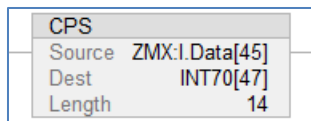
3. Now create 4 CPS instructions. These use the ZMX Input assembly instance and links the data to the “INT70” array. This is needed to line up all the data so it can correctly be placed into input tag under “ZMX_Data”.
 - a. First CPS links ZMX:I.Data[0] to INT70[0]. The length should be set to 8. Names used match the values created in earlier steps. Match the names as used in your application.



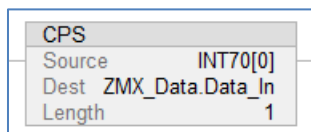
- b. Second CPS links ZMX:I.Data[15] to INT70[16]. The length should be set to 27.



- c. Next CPS links ZMX:I.Data[45] to INT70[47]. The length should be set to 14.



- d. Final CPS links INT70[0] to ZMX_Data.Data_In. Use a Length of 1.



4. When the controller is online data will now appear in the Input area for the ZMX.

Name	Value	Style	Data Type
└ ZMX_Data.Data_In	{...}		Banner_ZMX_Data_In
▸ ZMX_Data.Data_In.Input_Ack	0	Decimal	INT
▸ ZMX_Data.Data_In.Reserved1	0	Decimal	INT
▸ ZMX_Data.Data_In.Sensor_Fault_Code	0	Decimal	INT
▸ ZMX_Data.Data_In.Boot_Count	43	Decimal	INT
▸ ZMX_Data.Data_In.Up_Time	342533200	Decimal	DINT
▸ ZMX_Data.Data_In.Missed_Trigger_Count	0	Decimal	DINT
▸ ZMX_Data.Data_In.Reserved2	0	Decimal	INT
▸ ZMX_Data.Data_In.Reserved3	0	Decimal	INT
▸ ZMX_Data.Data_In.Reserved4	0	Decimal	INT
▸ ZMX_Data.Data_In.Reserved5	0	Decimal	INT
▸ ZMX_Data.Data_In.Reserved6	0	Decimal	INT
▸ ZMX_Data.Data_In.Reserved7	0	Decimal	INT
▸ ZMX_Data.Data_In.Reserved8	0	Decimal	INT
▸ ZMX_Data.Data_In.Reserved30	0	Decimal	INT
▸ ZMX_Data.Data_In.Frame_Number	2739525	Decimal	DINT
▸ ZMX_Data.Data_In.Invalid_Pixel_Count	11465	Decimal	DINT
▸ ZMX_Data.Data_In.Saturated_Pixel_Count	1	Decimal	DINT
▸ ZMX_Data.Data_In.Conf_Error_Code	0	Decimal	INT
▸ ZMX_Data.Data_In.Reserved9	0	Decimal	INT
▸ ZMX_Data.Data_In.Fill_Level_Percent	4	Decimal	INT
▸ ZMX_Data.Data_In.Peak_Height	48	Decimal	INT

5. If manual output control is needed for the ZMX then add one more CPS instructions.
 6. The CPS should link ZMX_Data.Data_Out to ZMX:O.Data[0]. The length is set to 42.

CPS		
Source	ZMX_Data.Data_Out	
Dest	ZMX:O.Data[0]	
Length	42	

7. It is now possible to update the ZMX. Reference the ZMX Manual for instructions on this procedure.