IND360 POWERCELL
EtherNet/IP Sample Code
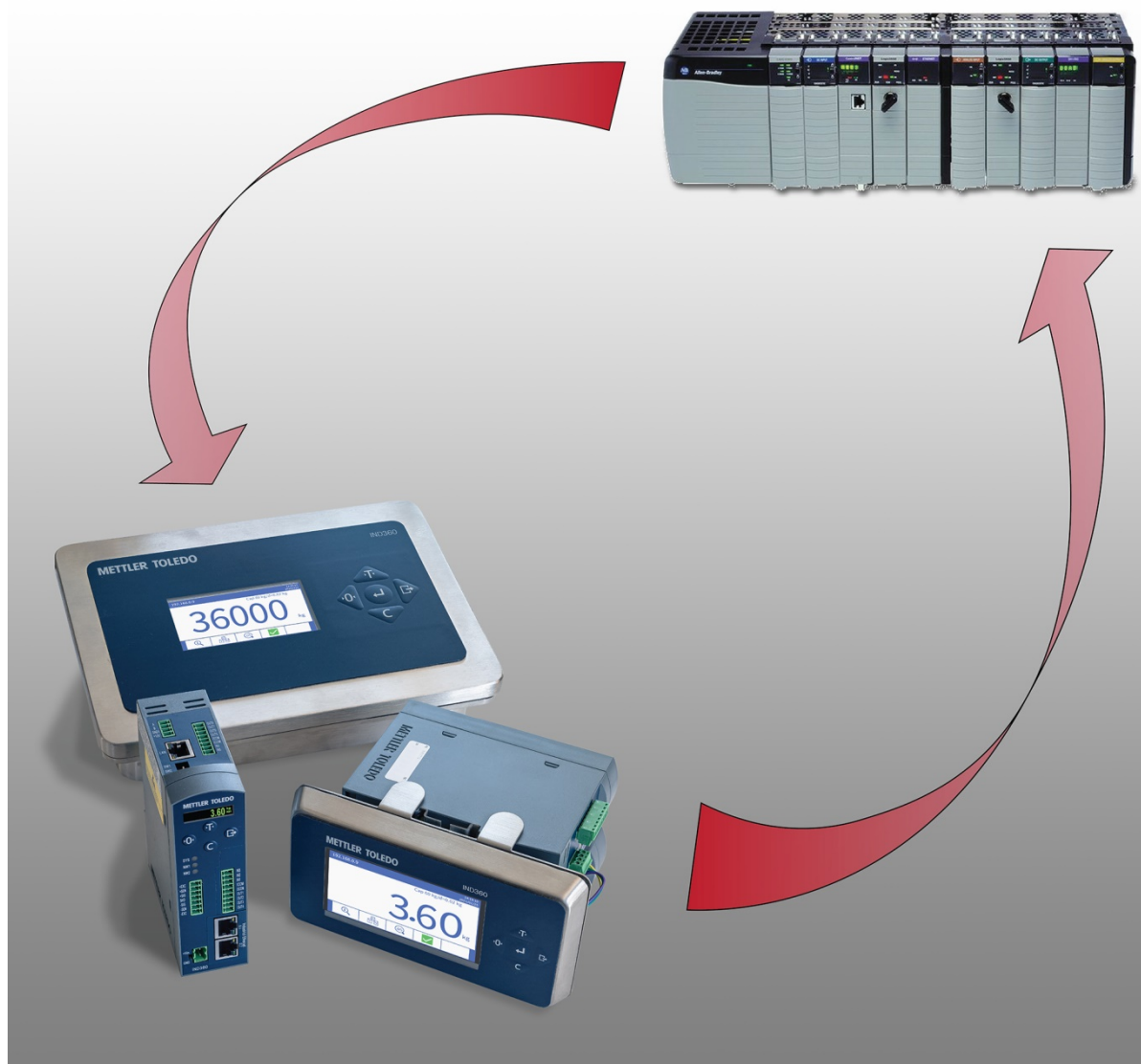
METTLER TOLEDO

Table of Contents

NOTICE	
NOTE: THE CONFIGURATION USED IN THIS SAMPLE CODE IS BASED ON THE DEFAULT SETTINGS:	
Rockwell Studio5000:	Version 24
PLC:	1769-L30ER
SAI DATA FORMAT:	2-BLOCK FORMAT (DEFAULT), 8-BLOCK FORMAT
IND360 IP ADDRESS:	192.168.0.2
EDS FILE:	MT_IND360_EIP_V1.1_20200728
IND360 DEVICE FIRMWARE VERSION:	V1.00.0012

It is recommended to integrate one IND360 into the PLC EtherNet/IP network and go through the sample code to understand the functionality of each Add-On Instruction (AOI).

1. Configure Development Environment

1.1. Confirm EDS Installation

This sample code project utilizes an EDS file for the IND360. These files can be found on www.mt.com/ind-ind360-downloads.

To confirm installation of IND360 EDS file:

1. In any Studio 5000 project, right click on "Ethernet" within the I/O Configuration folder in the controller organizer.

2. Select New Module.
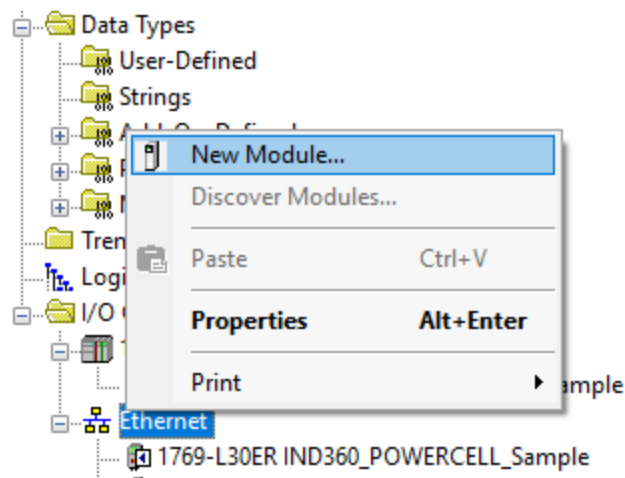
Figure 1-1: Try to add a new module to confirm EDS is installed

3. Search IND360
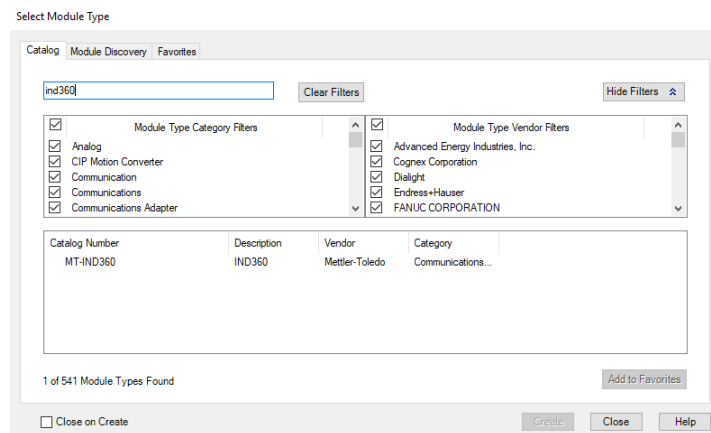
Figure 1-2: Search for IND360

If the EDS is installed, there should be an option for MT-IND360. If the search returns no results, follow these steps to install the EDS:

1. Go to the IND360 download page: www.mt.com/ind-IND360-downloads

2. Click the EDS file to begin the download.

3. Once the download is complete, unzip the folder

4. Use the EDS installation tool in Studio5000 to install the EDS.
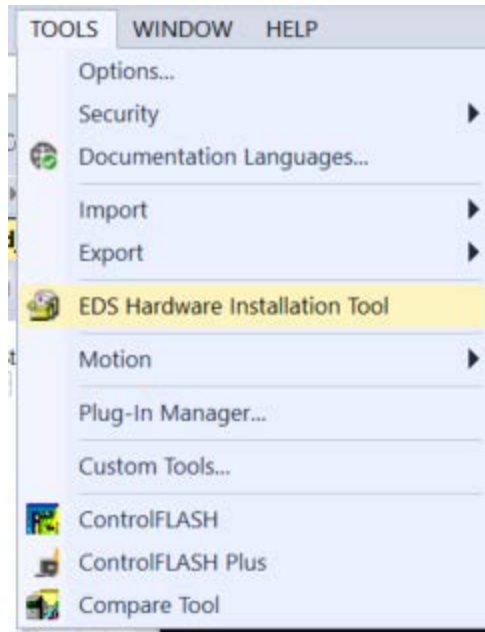
Figure 1-3: Use the EDS Hardware Installation Tool to complete installation

1.2. Import Example as a New Project

You need Studio 5000 V24 or above to import the examples.

Import the project in Studio5000, click "File-> Open"

Select the .ACD file and click open. The project will load.
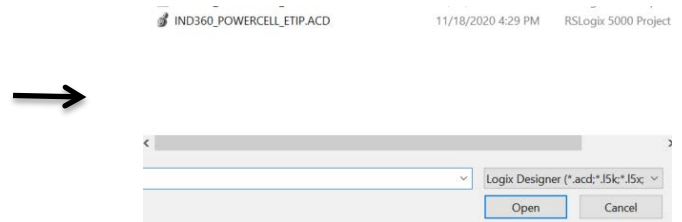
Figure 1-4: Open Project

1.3.   Import Example to an Existing Project

1. Add an MT-IND360 to the I/O Configuration in the existing project. See the first steps of Section 3 for more information on how to complete this. Using the name "IND360" and the IP Address 192.168.0.2 will require no changes to the sample code. If a different name or IP address is required, steps explaining what changes to make are provided below.
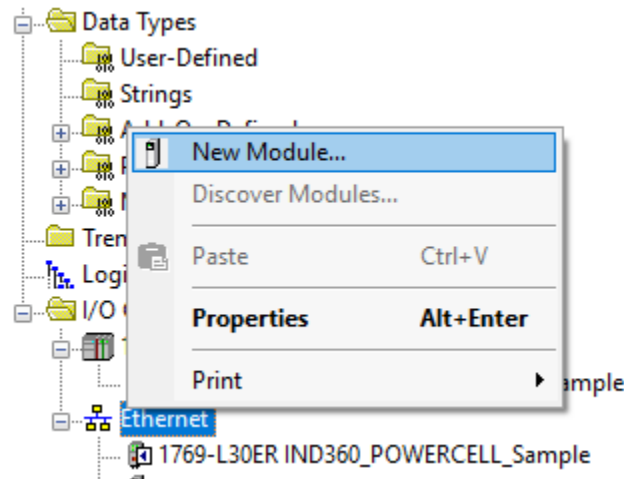

Figure 1-5: Add IND360 to the existing project

2. Copy the Add-On Instructions from the Add-On Instructions folder in the Controller Organizer of the sample project and paste in the same location in the existing project.
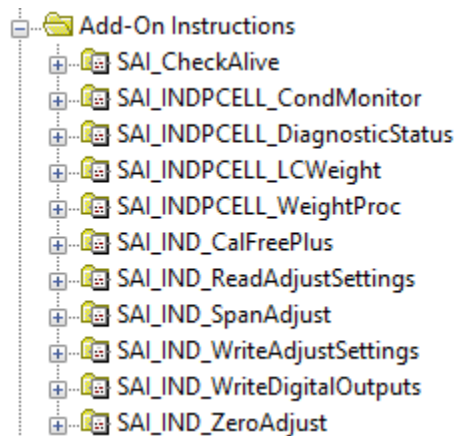
Figure 1-6: Copy/Paste AOIs

3. Copy the controller tags from the sample code project and paste in the controller tags of the existing project. Make sure not to copy the IND360:I and IND360:O tags since those are already present in the existing project.

Figure 1-7: Copy/Paste Controller Tags

4. Copy the Main Program local tags from the sample project and paste in the tags for the existing project.

Figure 1-8: Copy/Paste Main Program Local Tags

5. Copy the "MT_IND360_Application" routine from the sample project and paste in the existing project.
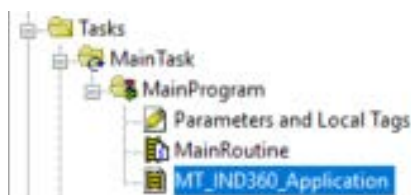
Figure 1-9: Copy/Paste the Routine

6. Make sure something in the existing project calls the MT_IND360_Application. Any AOIs that automatically monitor weight conditions will not run if nothing calls this routine.

7. If a name other than "IND360" was used as the name of the transmitter in the project, replace every use of "IND360" in the AOI instances with the name given to the transmitter in the project.

Figure 1-10: Example of name "IND360_POWERCELL" used in project

8. If an IP address other than 192.168.0.2 was used for the IND360 in the project, open the message configuration for every message in every AOI instance and in the communication tab, browse to the appropriate IND360 to set the path for the message.
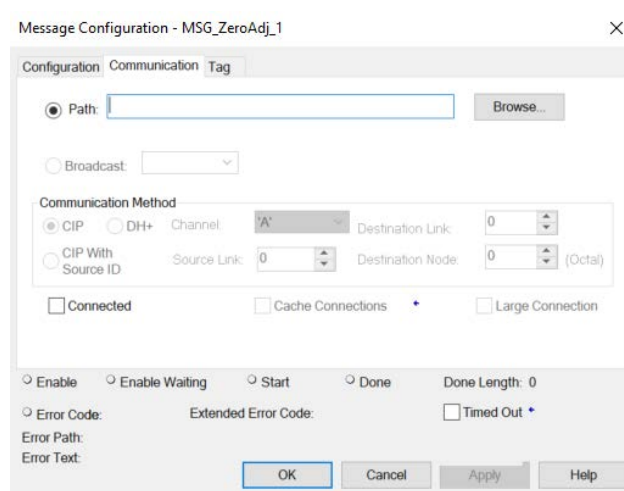
Figure 1-11: Set the Communication Path for each Message

1.4. Configure Controller Type

Please note that this is only necessary if using the sample code as the basis for the PLC project. If importing the routine and AOIs into an already existing project, this is unnecessary.

Right-click the project's controller, select Properties, and set the controller type.
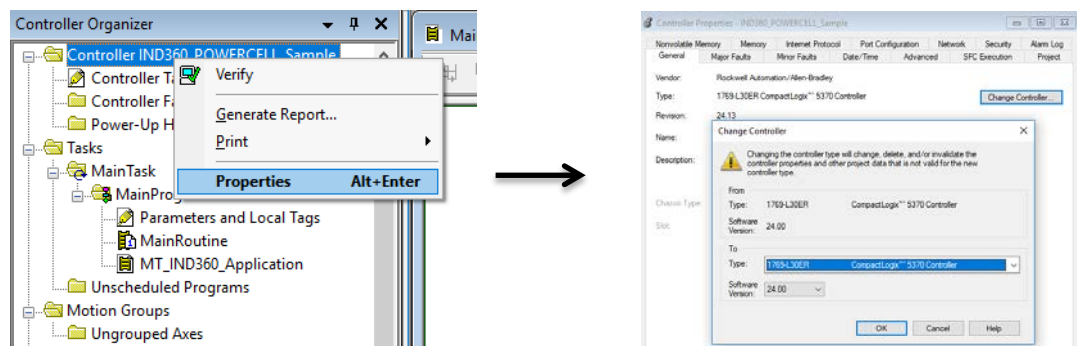
Figure 1-12: Configure controller type

Download the project to the controller and test.

2. Add-On Instructions(AOI)

About the configuration of the Messages in the add-on instructions:

Add-On Instructions that use acyclic communication functions, such as zero point adjustment, span adjustment, etc. need to create and configure Message variables, set paths and indicate which device to use for acyclic communication. The path of each device is different when multiple devices are networked.
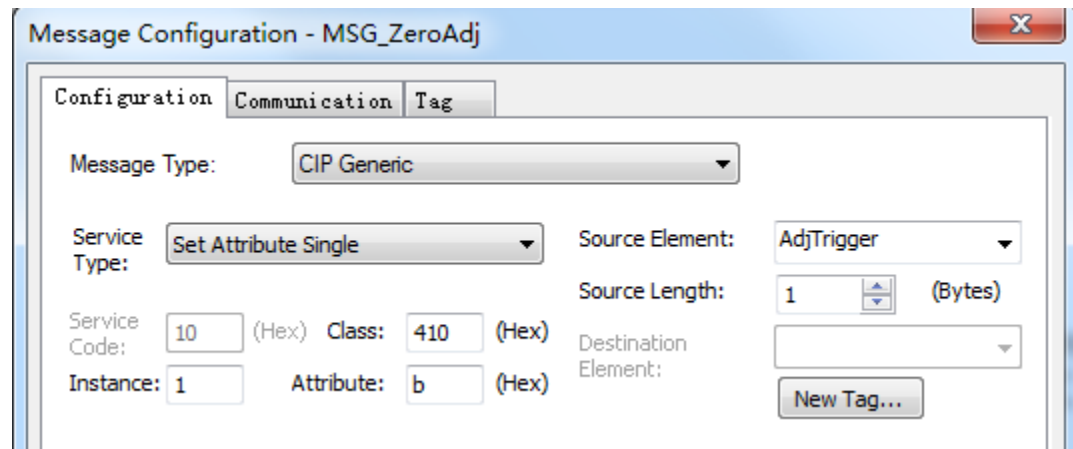
Figure 2-2-1: Configure Message Variable
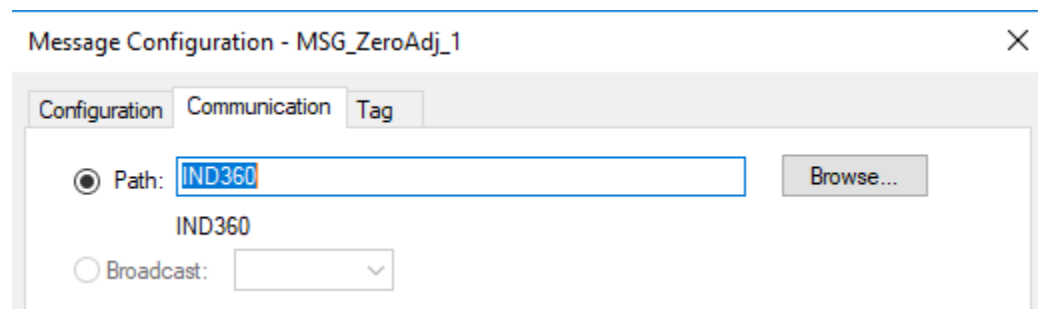
Figure 2-2-2: Set the path of the Message variable

Message configuration must be accessed via the AOI block in the main program. The message configuration cannot be accessed inside of the AOI logic files.
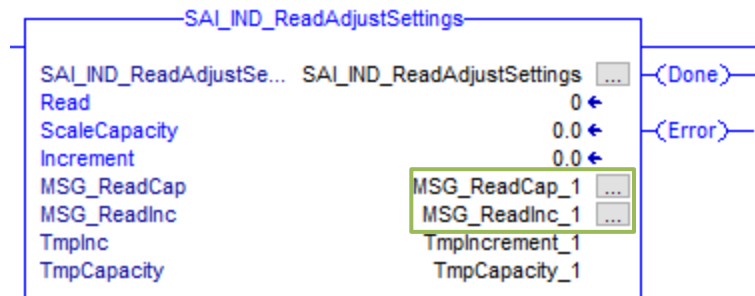
Figure 2-3: Access Message Configuration

2.1. Cyclic Weight Data

Read the real-time and stable weight from the transmitter. When performing zero and tare commands, the weight will stop updating.

Trigger execution of stable tare, stable zero, immediate tare, immediate zero and clear tare by setting that particular bit high. The response can be read, and there are flags for execution success and failure to indicate the result.

After the zero and tare commands are completed, the AOI will automatically restore whatever command is in WeightCmd and weight will be reported again. Typical values for WeightCmd are 0 (report gross weight) or 3 (report net weight). The DataOK bit is reset to 0 during overload, underload, adjustment and several other scenarios which can be used to judge abnormal conditions.
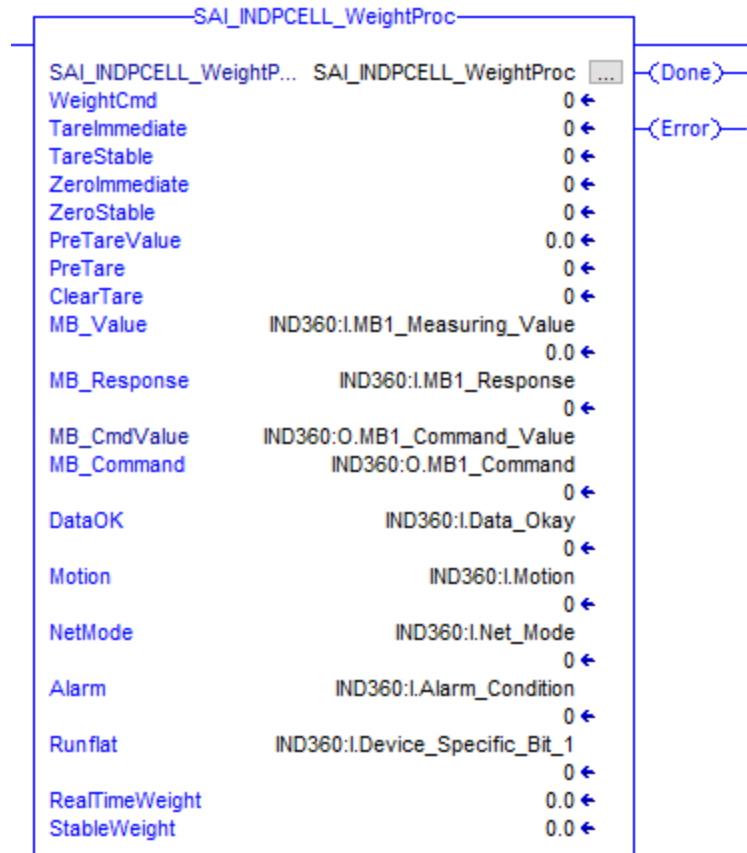
Figure 2-2-4: SAI_INDPCELL_WeightProc AOI

Input Parameters	Data Type	Description
WeightCmd	INT	Use this value to request the IND360 to report weight. When a zero or tare cyclic command is sent, the IND360 stops reporting weight. This AOI will automatically restore this command once the zero or tare command completes. 0 or 1 = Report gross weight 2 = Report tare weight 3 = Report net weight (default value in AOI) 5 = Report gross weight value (with internal resolution) 6 = Report tare weight value (with internal resolution) 7 = Report net weight value (with internal resolution)
TareImmediate	BOOL	Set = 1 to issue tare command regardless of whether the weight value is stable or not. Net weight will not automatically be reported after tare command is issued. Recommended to use cyclic command 3 (report net weight) in WeightCmd input to receive net weight
TareStable	Bool	Set = 1 to issue tare command to IND360 when weight is stable. Command will timeout if remain within the stability criteria (+/- 1d within 0.3 seconds default) for a predefined timeout range (3 seconds default). Net weight will not automatically be reported after tare command is issued. Recommended to use cyclic command 3 (report net weight) in WeightCmd input to receive net weight

ZeroImmediate	BOOL	Set = 1 to issue zero command regardless of whether the weight value is stable or not. This is only intended for minor changes to the zero point due to drifting. For a formal zero adjustment, use the SAI_IND_ZeroAdjust AOI. Command will return an error if weight value is not within the zero range (+/- 2% default).
ZeroStable	BOOL	Set = 1 to issue zero command to IND360 when ewight is stable. Command will timeout if remain within the stability criteria (+/- 1d within 0.3 seconds default) for a predefined timeout range (3 seconds default). This is only intended for minor changes to the zero point due to drifting. For a formal zero adjustment, use the SAI_IND_ZeroAdjust AOI. Command will return an error if weight value is not within the zero range (+/- 2% default).
PreTareValue	Real	Configure with preset tare value. This value will not be sent to IND360 until Pretare input is set to 1.
Pretare	BOOL	Set = 1 when ready to perform preset tare using the value from PreTareValue.
ClearTare	BOOL	Set = 1 to clear the current tare value.
MB_Value	Real	This should always be set to the MB1_Measuring_Value of the IND360. This will provide weight data for the AOI
MB_Response	INT	This should always be set to MB1_Response value of the IND360. Once a cyclic command is successfully executed, MB_Response = MB_Command. The AOI uses this information to detect if a command has been executed successfully or if an error has occurred.

DataOK	BOOL	This bit gets set to 0 when the device is still operational but the value being reported cannot be guaranteed to be valid. The following conditions cause the Data Okay bit to be set to 0:
		• Device is powering up
		• Device is in setup mode
		• Device is in test mode
		• Over capacity condition occurs
		   - When the A/D converter is at its limit
		   - Product dependent over capacity that occurs when the device determines it cannot trust the weight
		• Under capacity condition occurs
		   - When the A/D converter is at its limit
		Product dependent under capacity that occurs when the device determines it cannot trust the weight

Motion	Bool	This should always be set to Motion bit of IND360. Motion bit is high when the weight value is not stable. ZeroStable and TareStable commands will not complete while the Motion bit is high.
NetMode	BOOL	This should always be set to the Net_Mode bit of the IND360. NetMode = 1 after a tare command has been executed. Just because NetMode = 1, does not mean net weight is being reported by the IND360. Net weight must be requested by the PLC (cyclic command 3).
Alarm	BOOL	This should always be set to Alarm_Condition bit of IND360. Bit will go high when alarm conditions are present. Bit will automatically go low when no alarm conditions are present. See SAI manual for more information on what causes the Alarm_Condition to go high.
Runflat	Bool	This should always be set to Device_Specific_Bit_1 of IND360. This bit will go high when RunFlat algorithm is triggered. This can be used to continue running a process at reduced accuracy if a load cell goes bad. See the IND360POWERCELL user manual for more details.

Output Parameters	Data Type	Description
MB_Command	INT	This should always be set to MB1_Command value of the IND360. Value of the last cyclic command sent to the IND360. Once successfully executed, MB_Response = MB_Command.
RealTimeWeight	Real (32 bits)	Current weight of the scale. This value is updated constantly while the AOI is enabled.
StableWeight	Real (32 bits)	Latest stable weight reading of the scale. This value does not update whenever the Motion bit is high.
Done	BOOL	Will be latched high when zero or tare command has successfully completed. When a new zero or tare command begins, bit will be unlatched until command completes successfully
Error	BOOL	Will be latched high when zero or tare command fails to complete. When a new zero or tare command begins, bit will be unlatched until a command fails to complete.

2.2. Communication Heart Beat Monitoring

Monitoring communication between the controller and IND360. If Alive bit is set, cyclic communications between the controller and transmitter are active.
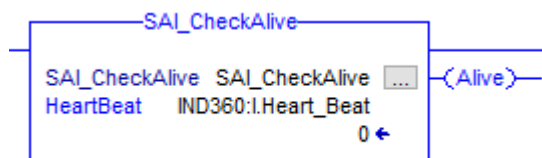
Figure 2-2-5: SAI_CheckAlive AOI

Input Parameters	Data Type	Description
HeartBeat	BOOL	This should always be set to Heart_Beat bit of IND360. This bit will pulse on and off each second if cyclic communications between the IND360 and the controller are established
Output Parameters	Data Type	Description
Alive	BOOL	This bit = 1 if cyclic communications are established between the IND360 and the controller.

2.3. Diagnostic Status Monitoring

This AOI automatically reports a variety of diagnostic information to the Boolean output bits. If a load cell stops communicating, the bit number of LC_OfflineInfo corresponding to that load cell number will go high.
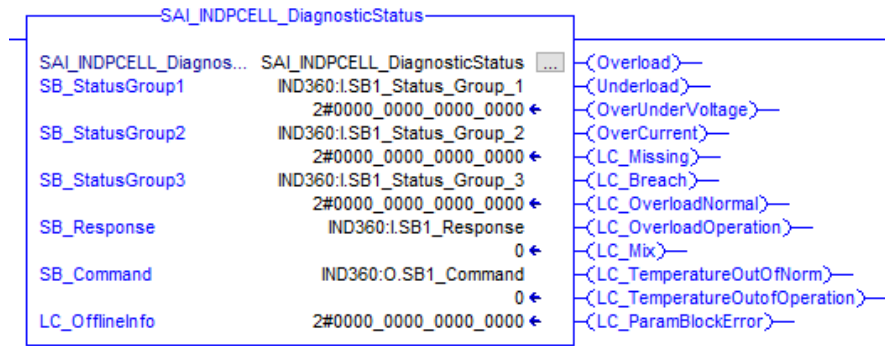
Figure 2-6 SAI_INDPCELL_DiagnosticStatus AOI

Input Parameters	Data Type	Description
SB_StatusGroup1	INT	This should always be set to SB1_Status_Group_1 of the IND360. If set to anything else, diagnostic information will not be reported correctly.
SB_StatusGroup2	INT	This should always be set to SB1_Status_Group_2 of the IND360. If set to anything else, diagnostic information will not be reported correctly.
SB_StatusGroup3	INT	This should always be set to SB1_Status_Group_3 of the IND360. If set to anything else, diagnostic information will not be reported correctly.
SB_Response	INT	This should always be set to SB1_Response of the IND360. If set to anything else, diagnostic information will not be reported correctly.
Output Parameters	**Data Type**	**Description**
SB_Command	INT	This should always be set to SB1_Command of the IND360. If set to anything else, diagnostic information will not be reported correctly.
Overload	BOOL	High when the weight goes above the user programmed overload limit
Underload	BOOL	High when the weight goes below the under zero blanking value
OverUnderVoltage	BOOL	High when POWERCELL CAN network voltage is outside of accepted range
OverCurrent	BOOL	High when current in POWERCELL CAN network is above 1A
LC_Missing	BOOL	High if POWERCELL load cells are not all communicating with the IND360. See LC_OfflineInfo to determine which load cells are not communicating

LC_OfflineInfo	INT	Individual bits will go high when corresponding POWERCELL node is no longer communicating to IND360. (e.g. Bit 0 will go high if node 1 is no longer communicating.)
		
		Word:
		0000 0000 XXXX XXXX
		          I………..I
		       LC#8 LC#1

LC_Breach	BOOL	High if load cell's enclosure has been compromised and vulnerable to outside influences. The sensor will fail in

		many cases if the breach is not corrected or if the sensor is not replaced. Not applicable to all POWERCELL models.
LC_OverloadNormal	BOOL	High if weight on load cell is between 101-150% of rated capacity. Use SAI_INDPCELL_LCWeight AOI to determine the current weight on each load cell.
LC_OverloadOperation	BOOL	High if weight on load cell is greater than 150% of rated capacity. Use SAI_INDPCELL_LCWeight AOI to determine the current weight on each load cell.
LC_Mix	BOOL	High if different models of POWERCELL load cells are used together.
LC_TemperatureOutofNorm	BOOL	High if temperature of a load cell is outside of its compensated range. Use SAI_INDPCELL_CondMonitor AOI to determine current temperature for each load cell
LC_TemperatureOutOfOperation	BOOL	High if temperature of a load cell is outside of its operating range. Use SAI_INDPCELL_CondMonitor AOI to determine current temperature for each load cell
LC_ParamBlockError	BOOL	High if load cell parameter block checksum error occurs

2.4.   Read Scale Adjustment Settings

This AOI reads the capacity and increment values used by the IND360. Set Read = 1 to read values. It is useful to know the current scale settings before performing any scale adjustment procedure.
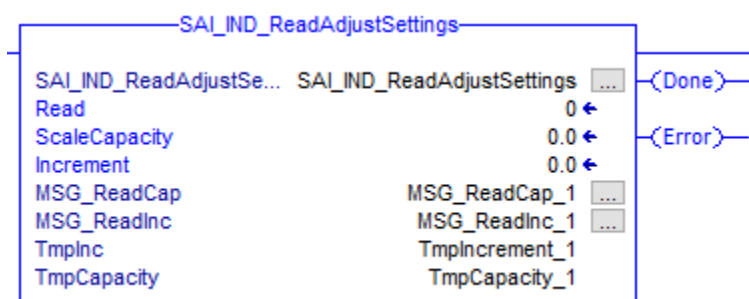
Function 2-2-7: SAI_IND_ReadAdjustSettings AOI

Input Parameters	Data Type	Description
Read	BOOL	Set = 1 to read capacity and increment values from IND360
Output Parameters	**Data Type**	**Description**
ScaleCapacity	REAL	Capacity for the scale set in IND360.
Increment	REAL	Increment size used for weight values coming from the IND360
Done	BOOL	Latched high when values are successfully read. Unlatched when Read input is set.
Error	BOOL	Latched high if an error occurred and values could not be read. Unlatched when Read input is set. Check the errors of the messages for this AOI to troubleshoot
In/Out Parameters	**Data Type**	**Description**
MSG_ReadCap	Message	**Message Type:** CIP Generic **Service Type:** Get Attribute Single **Class:** 417 (Hex) **Instance:** 1

		Attribute: 17 (Hex) **Destination Element:** TmpCapacity_1 **Communication -> Path:** Browse for the appropriate IND360
MSG_ReadInc	Message	**Message Type:** CIP Generic **Service Type:** Get Attribute Single **Class:** 417 (Hex) **Instance:** 1 **Attribute:** 16 (Hex) **Destination Element:** TmpIncrement_1 **Communication -> Path:** Browse for the appropriate IND360
TmpInc	Real	Temporary variable used with message
TmpCapacity	Real	Temporary variable used with message

2.5. Write Scale Adjustment Settings

Configure ScaleCapacity and Increment values and then set Write = 1 when ready to write values to IND360POWERCELL. Capacity and Increment value parameters must satisfy 500 <= capacity / increment <= 100000
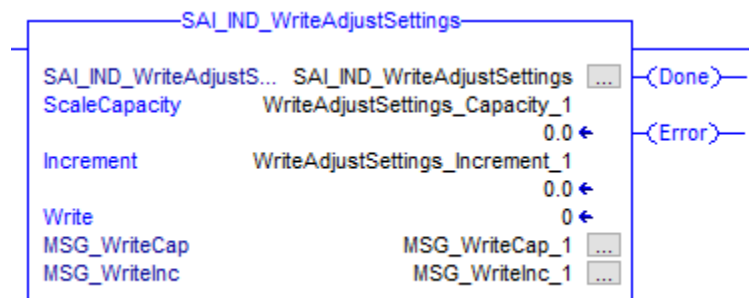
Figure 2-8: SAI_IND_WriteAdjustSettings AOI

Input Parameters	Data Type	Description
ScaleCapacity	REAL	Configure capacity for the scale. Capacity and Increment value parameters must satisfy 500 <= capacity / increment <= 100000
Increment	REAL	Configure increment size for the scale. Capacity and Increment value parameters must satisfy 500 <= capacity / increment <= 100000
Write	BOOL	Set = 1 to write capacity and increment values to IND360
Output Parameters	Data Type	Description
Done	BOOL	Latched high when values are successfully written. Unlatched when Write input is set.
Error	BOOL	Latched high if an error occurred and values could not be written. Unlatched when Write input is set. Check the errors of the messages for this AOI to troubleshoot
In/Out Parameters	Data Type	Description
MSG_WriteCap	Message	**Message Type:** CIP Generic **Service Type:** Set Attribute Single **Class:** 417 (Hex)

		Instance: 1 Attribute: 17 (Hex) Source Element: WriteAdjustSettings_Capacity_1 Source Length: 4 (Bytes) Communication -> Path: Browse for the appropriate IND360
MSG_WriteInc	Message	Message Type: CIP Generic Service Type: Set Attribute Single Class: 417 (Hex) Instance: 1 Attribute: 16 (Hex) Source Element: WriteAdjustSettings_Increment_1 Source Length: 4 (Bytes) Communication -> Path: Browse for the appropriate IND360

2.6.   Zero Adjustment

A zero adjustment must be performed first before either a span adjustment or CalFree+. Make sure the scale is empty before starting this procedure.

Set the Start bit = 1 to start the zero adjustment. The Done and Error flags will indicate the result. The DataOK bit is cleared during the adjustment process.
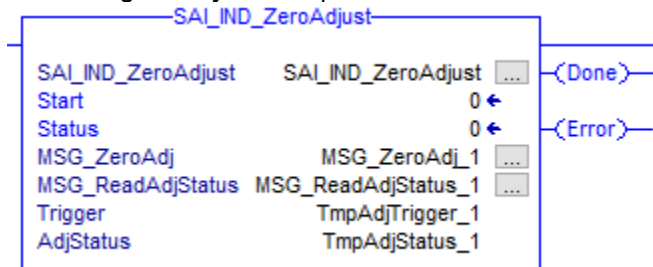
Figure 2-9: SAI_IND_ZeroAdjust AOI

Input Parameters	Data Type	Description
Start	BOOL	Set = 1 to begin zero adjustment process
Output Parameters	Data Type	Description
Done	BOOL	Latched high when zero adjustment successfully completes. Unlatched when a new zero adjustment begins.
Error	BOOL	Latched high if an error occurred and zero adjustment could not complete. Unlatched when a new zero adjustment begins. Check the errors of the messages for this AOI to troubleshoot
Status	INT	0 = zero adjustment has not begun. 2047 = zero adjustment in process. 2046 = zero adjustment completed successfully
In/Out Parameters	Data Type	Description
MSG_ZeroAdj	Message	Message Type: CIP Generic Service Type: Set Attribute Single Class: 410 (Hex) Instance: 1 Attribute: b (Hex) Source Element: TmpAdjTrigger_1

		Source Length: 1 (Bytes)
		Communication -> Path: Browse for the appropriate IND360
MSG_ReadAdjStatus	Message	Message Type: CIP Generic
		Service Type: Get Attribute Single
		Class: 410 (Hex)
		Instance: 1
		Attribute: 7 (Hex)
		Destination Element: TmpAdjStatus_1
		Communication -> Path: Browse for the appropriate IND360
Trigger	SINT	Temporary value for use with AOI message
AdjStatus	INT	Temporary value for use with AOI message

2.7. Span Adjustment

Before executing this AOI, a zero adjustment must be completed. The zero point is the first adjustment point used for all linearity ranges.

First configure the linearity range. LinearityRange:

0 = Two point adjustment;
1 = Three point adjustment,
2 = Four point adjustment;
3 = Five point adjustment;
Note that the zero adjustment already completed was the first adjustment point.

Configure all weights to be used in the adjustment process. Start at Hi_Weight with highest weight value and proceed in descending order. An error will occur if adjustment weight values are not in correct, descending order or if a weight value of zero is used. Set Start = 1 when ready to begin adjustment.

LoadWeight will go high when the IND360 is prepared for the next adjustment point. Place the adjustment weight on the scale that matches the value in CurrentWeight. Set ConfirmWeight = 1 to begin the adjustment step. Repeat this step for each adjustment point.

If the Done or Error flag is set, the adjustment is complete.
The adjustment process can be cancelled any time by setting Cancel = 1.
The DataOK bit is cleared during the calibration process.

```
                        SAI_IND_SpanAdjust
    SAI_IND_SpanAdjust      SAI_IND_SpanAdj  ...    (Done)
    LinearityRange          LinearityRange_1
                                        0  ←        (Error)
    Hi_Weight                         0.0  ←
    Mid_Weight                        0.0  ←        (Cancelled)
    Low_Weight                        0.0  ←
    xLow_Weight                       0.0  ←        (LoadWeight)
    Start                               0  ←
    CurrentWeight                     0.0  ←
    ConfirmWeight                       0  ←
    Cancel                              0  ←
    DataOK                  IND360:I.Data_Okay
                                        0  ←
    Status                              0  ←
    CurrentPoint                        0  ←
    MSG_CancelAdj           MSG_CancelAdj_1     ...
    MSG_StartSpanAdj        MSG_StartSpanAdj_1  ...
    MSG_LinearityRange      MSG_LinearityRange_1 ...
    MSG_ReadAdjStatus       MSG_ReadAdjStatus_1 ...
    MSG_ValidateAdj         MSG_ValidateAdj_1   ...
    TmpTrigger              TmpAdjTrigger_1
    TmpAdjStatus            TmpAdjStatus_1
    TmpSpanAdjWeight        TMPSpanAdjWeight_1
```

Figure 2-10: SAI_IND_SpanAdjust AOI

```
Specify the                Linearity      no    Linearity      no    Linearity      no    Linearity
LinearityRange             Range = 0            Range = 1            Range = 2            Range = 3
(0, 1, 2, 3)                  │yes                 │yes                 │yes                 │yes
    │                                                                                           
Configure all the      LoadWeight = 1;     LoadWeight = 1;     LoadWeight = 1;     LoadWeight = 1;
required reference     CurrentPoint = 2;   CurrentPoint = 2;   CurrentPoint = 2;   CurrentPoint = 2;
weights                CurrentWeight =     CurrentWeight =     CurrentWeight =     CurrentWeight =
    │                     Hi_Weight           Mid_Weight          Low_Weight          xLow_Weight
Toggle Start           ConfirmWeight       ConfirmWeight       ConfirmWeight       ConfirmWeight
input bit              Adjustment in       Adjustment in       Adjustment in       Adjustment in
                         Process             Process             Process             Process

                                          LoadWeight = 1;     LoadWeight = 1;     LoadWeight = 1;
                                          CurrentPoint = 3;   CurrentPoint = 3;   CurrentPoint = 3;
                                          CurrentWeight =     CurrentWeight =     CurrentWeight =
                                             Hi_Weight           Mid_Weight          Low_Weight
                                          ConfirmWeight       ConfirmWeight       ConfirmWeight
                                          Adjustment in       Adjustment in       Adjustment in
                                            Process             Process             Process

                                                              LoadWeight = 1;     LoadWeight = 1;
                                                              CurrentPoint = 4;   CurrentPoint = 4;
                                                              CurrentWeight =     CurrentWeight =
                                                                 Hi_Weight           Mid_Weight
                                                              ConfirmWeight       ConfirmWeight
                                                              Adjustment in       Adjustment in
                                                                Process             Process

                                                                                  LoadWeight = 1;
                                                                                  CurrentPoint = 5;
                                                                                  CurrentWeight =
                                                                                     Hi_Weight
                                                                                  ConfirmWeight
                                                                                  Adjustment in
                                                                                    Process

Cancelled  ←  Cancel

           Success     Validate
Done  ←                Adjustment
           Failed
Error  ←
```
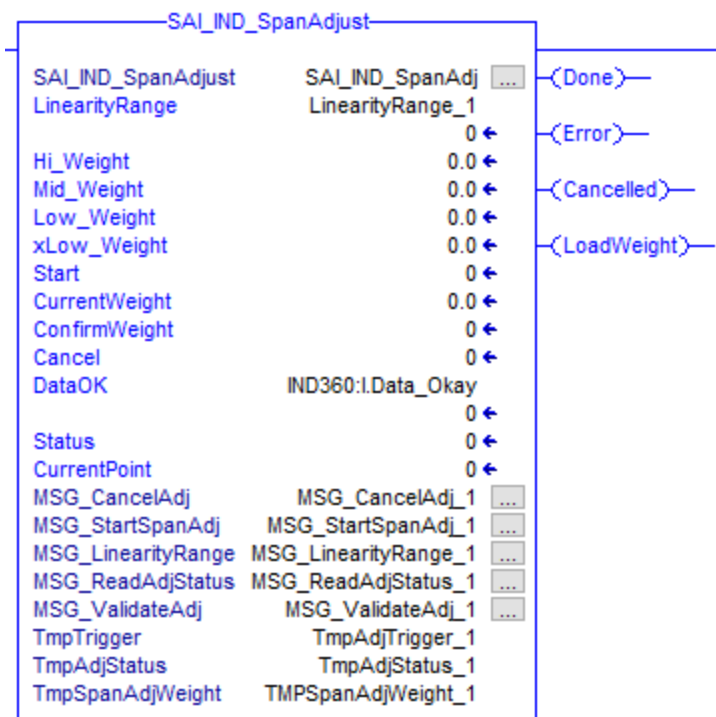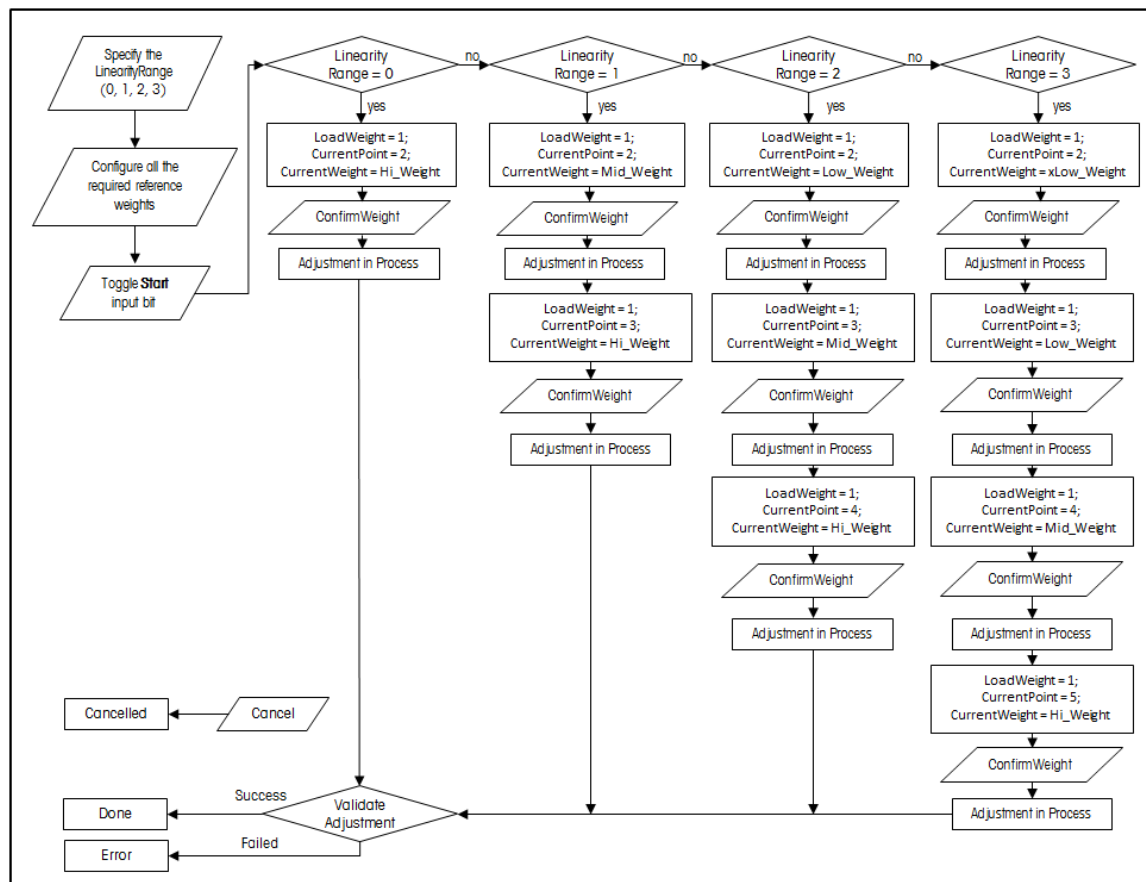
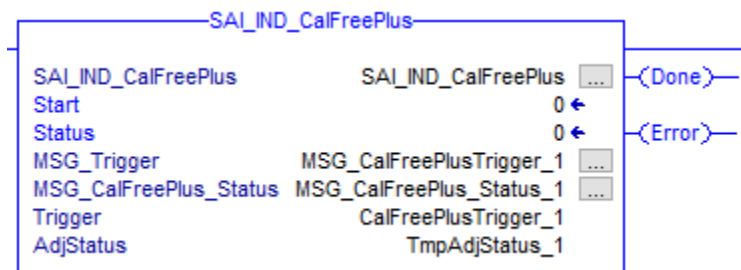| Input Parameters	Data Type	Description
LinearityRange	INT	0 = 2-point adjustment; Hi_Weight 1 = 3-point adjustment; Hi_Weight, Mid_Weight 2 = 4-point adjustment; Hi_Weight, Mid_Weight, Low_Weight 3 = 5-point adjustment; Hi_Weight, Mid_Weight, Low_Weight, xLow_Weight
Hi_Weight	REAL	Highest adjustment weight used in adjustment process. Used with all linearity ranges
Mid_Weight	REAL	Middle weight used in adjustment process. Used with three point, four point and five point linearity ranges
Low_Weight	REAL	Low weight used in adjustment process. Used with four and five point linearity ranges
xLow_Weight	REAL	xLow weight used in adjustment process. Used with five point linearity range
Start	BOOL	Set = 1 once LinearityRange and adjustment weight values are configured to begin process.
ConfirmWeight	BOOL	Set = 1 when LoadWeight output = 1 and the test weight matching the value in CurrentWeight is loaded onto the scale
Cancel	BOOL	Set = 1 to cancel adjustment process
Output Parameters	Data Type	Description
CurrentWeight	REAL	The value of the test weight that should be placed on the scale
Done	BOOL	Latched high when adjustment completes successfully. Unlatched when another span adjustment begins.
Error	BOOL	Latched high if an error occurred and adjustment could not complete. Unlatched when another span adjustment begins. Check the errors of the messages for this AOI to troubleshoot
Cancelled	BOOL	Latched high if the adjustment process was cancelled. Unlatched when another span adjustment begins.
LoadWeight	BOOL	Set high when user needs to load test weight corresponding to the value CurrentWeight output.
Status	INT	Value generally only needed for troubleshooting purposes. 0 = span adjustment has not begun or has completed. 2047 = span adjustment in process. 2045 = Ready for next adjustment point.
CurrentPoint	DINT	Value = current adjustment point being calibrated
In/Out Parameters	Data Type	Description
MSG_CancelAdj	Message	**Message Type:** CIP Generic **Service Type:** Set Attribute Single **Class:** 410 (Hex) **Instance:** 1 **Attribute:** 4 (Hex) **Source Element:** TmpAdjTrigger_1 **Source Length:** 1 (Bytes) **Communication -> Path:** Browse for the appropriate IND360
MSG_StartSpanAdj	Message	**Message Type:** CIP Generic **Service Type:** Set Attribute Single **Class:** 410 (Hex) **Instance:** 1 **Attribute:** 9 (Hex) **Source Element:** TMPSpanAdjWeight_1 **Source Length:** 4 (Bytes) **Communication -> Path:** Browse for the appropriate IND360

MSG_LinearityRange	Message	**Message Type:** CIP Generic **Service Type:** Set Attribute Single **Class:** 410 (Hex) **Instance:** 1 **Attribute:** a (Hex) **Source Element:** LinearityRange_1 **Source Length:** 2 (Bytes) **Communication -> Path:** Browse for the appropriate IND360
MSG_ReadAdjStatus	Message	**Message Type:** CIP Generic **Service Type:** Get Attribute Single **Class:** 410 (Hex) **Instance:** 1 **Attribute:** 7 (Hex) **Destination Element:** TmpAdjStatus_1 **Communication -> Path:** Browse for the appropriate IND360
MSG_ValidateAdj	Message	**Message Type:** CIP Generic **Service Type:** Set Attribute Single **Class:** 410 (Hex) **Instance:** 1 **Attribute:** 10 (Hex) **Source Element:** TmpAdjTrigger_1 **Source Length:** 1 (Bytes) **Communication -> Path:** Browse for the appropriate IND360
TmpTrigger	SINT	Temporary value for use with AOI message
TmpAdjStatus	INT	Temporary value for use with AOI message
TmpSpanAdjWeight	REAL	Temporary value for use with AOI message

2.8. CalFree+

CalFree+ is a method to perform a scale adjustment without using test weights. This is based on the POWERCELL load cell rated capacity and count value. This method can be used for initial testing of systems or when a large structure is used as the weighing vessel and it is not possible to apply test weights to the structure.

METTLER TOLEDO highly recommends that test weights or the RapidCal™ method be used whenever possible as these methods provide the most accurate adjustment results.

Set Start = 1 to begin CalFree+ process. The result is displayed with the Done or Error flag.

Figure 2-11: SAI_IND_CalFreePlus function block

Input Parameters	Data Type	Description
Start	BOOL	Set = 1 to begin CalFreePlus process
Output Parameters	**Data Type**	**Description**
Done	BOOL	Latched high when CalFreePlus successfully completes. Unlatched when CalFreePlus is triggered again.
Error	BOOL	Latched high if an error occurred and CalFreePlus could not complete. Unlatched when CalFreePlus is triggered again. Check the errors of the messages for this AOI to troubleshoot
Status	INT	0 = CalFreePlus has not begun or has completed. 2047 = CalFreePlus in process.
In/Out Parameters	**Data Type**	**Description**
MSG_Trigger	Message	**Message Type:** CIP Generic **Service Type:** Set Attribute Single **Class:** 410 (Hex) **Instance:** 1 **Attribute:** 1e (Hex) **Source Element:** CalFreePlusTrigger_1 **Source Length:** 2 (Bytes) **Communication -> Path:** Browse for the appropriate IND360
MSG_ReadAdjStatus	Message	**Message Type:** CIP Generic **Service Type:** Get Attribute Single **Class:** 410 (Hex) **Instance:** 1 **Attribute:** 1e (Hex) **Destination Element:** TmpAdjStatus_1 **Communication -> Path:** Browse for the appropriate IND360
Trigger	SINT	Temporary value for use with AOI message
AdjStatus	INT	Temporary value for use with AOI message

2.9. Read Individual Load Cell Weight Value (Gross or Net)

This AOI will report the weight for each individual load cell. Configure the Net input first. 0 = Read gross weight from each individual load cell, 1 = Read net weight of each individual load cell. Once the Net input is configured, set Read bit to 1 to begin reading the load cell weight values.

The total scale weight and each individual load cell weight will be placed into individual output parameters. If less than the maximum number of load cells (14) is used, all unused load cell output parameters will have a value of 0. Once the process completes, the Done bit will = 1. The error bit will = 1 if an error occurs.
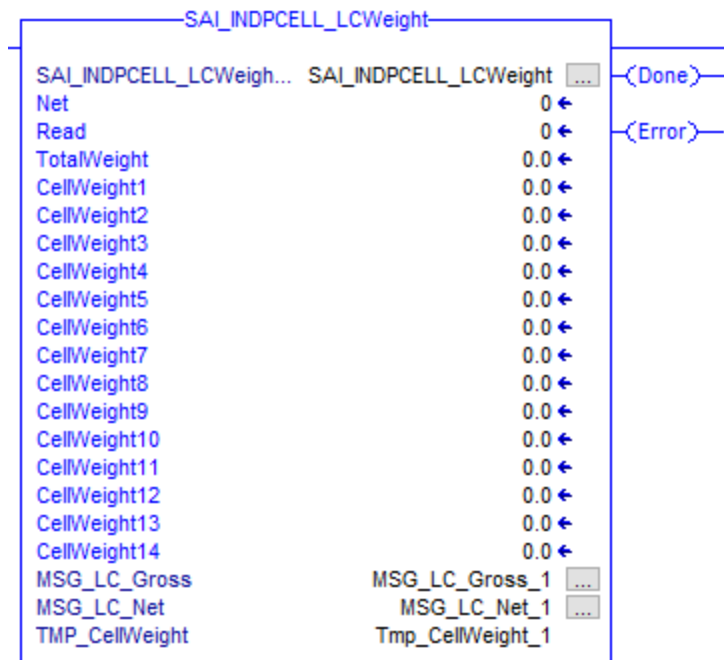
Figure 2-12 SAI_INDPCELL_LCWeight AOI

Input Parameters	Data Type	Description
Net	BOOL	Configure whether gross weight or net weight should be reported. 0 = report gross weight, 1 = report net weight
Read	BOOL	Set = 1 to read weight values. Will not continue to update weight values automatically
Output Parameters	Data Type	Description
Done	BOOL	Latched high when update successfully completes. Unlatched when update is triggered again.
Error	BOOL	Latched high if an error occurred and update could not complete. Unlatched when update is triggered again. Check the errors of the messages for this AOI to troubleshoot
TotalWeight	REAL	Total weight reported for the scale.
CellWeight1	REAL	Weight reported by POWERCELL node 1
CellWeight2	REAL	Weight reported by POWERCELL node 2
CellWeight3	REAL	Weight reported by POWERCELL node 3
CellWeight4	REAL	Weight reported by POWERCELL node 4
CellWeight5	REAL	Weight reported by POWERCELL node 5
CellWeight6	REAL	Weight reported by POWERCELL node 6
CellWeight7	REAL	Weight reported by POWERCELL node 7
CellWeight8	REAL	Weight reported by POWERCELL node 8
CellWeight9	REAL	Weight reported by POWERCELL node 9
CellWeight10	REAL	Weight reported by POWERCELL node 10
CellWeight11	REAL	Weight reported by POWERCELL node 11
CellWeight12	REAL	Weight reported by POWERCELL node 12
CellWeight13	REAL	Weight reported by POWERCELL node 13
CellWeight14	REAL	Weight reported by POWERCELL node 14
In/Out Parameters	Data Type	Description
MSG_LC_Gross	Message	**Message Type:** CIP Generic **Service Type:** Get Attribute Single

		Class: 417 (Hex) Instance: 1 Attribute: 24 (Hex) Destination Element: Tmp_CellWeight_1 Communication -> Path: Browse for the appropriate IND360
MSG_LC_Net	Message	Message Type: CIP Generic Service Type: Get Attribute Single Class: 417 (Hex) Instance: 1 Attribute: 25 (Hex) Destination Element: Tmp_CellWeight_1 Communication -> Path: Browse for the appropriate IND360
Tmp_CellWeight	REAL[15]	Temporary value for use with AOI message

3. Steps to Add New IND360s

Because EtherNet/IP uses IP addresses to distinguish different devices, when multiple IND360s are networked, the default IP address needs to be modified first. Each IND360 must have a different IP address.

1) Click "Communication-> Industrial Ethernet -> IP Address" in the IND360 Advanced Service Mode in order to modify the IP address
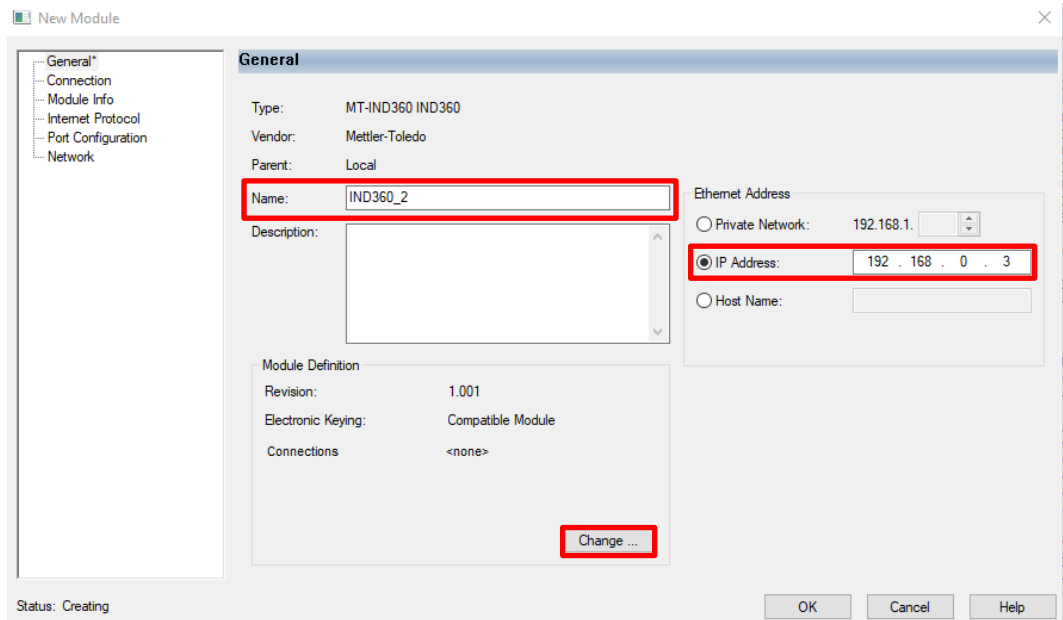

Figure 3-1: IND360 IP Address Menu

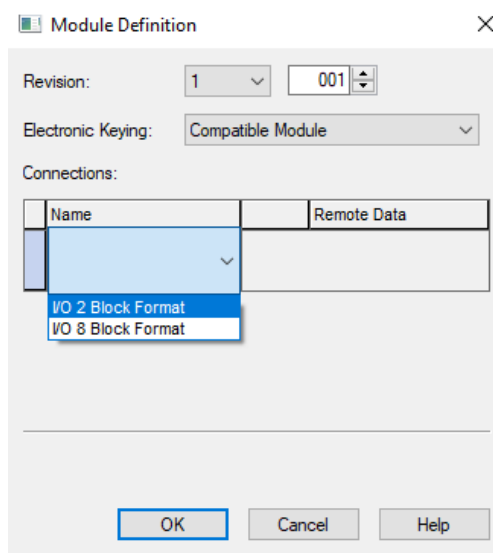2) Add an MT-IND360 to "I / O Configuration-> Ethernet" in Studio5000.
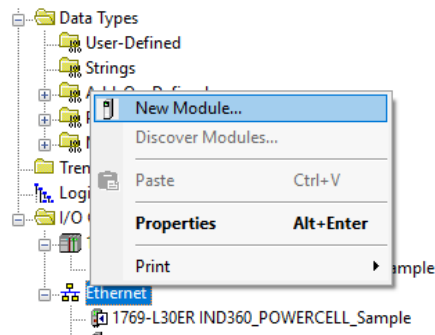
Figure 3-2: Add a device

3) Configure the name and IP address. Each device needs a unique name and IP address, and then click "Change".

Figure 3-3: Configure name and IP address

4) Select "I/O 2 Block Format "

Figure 3-4: Module Definition Configuration

5) Copy the controller tags relating to the sample code and paste in the same location in order to create a duplicate set of tags. Please note that since all tags end with "_1", Studio 5000 will create duplicates that all end with "_2" instead.

Figure 3-5: Copy/Paste Tags to Create Duplicates

6) Copy and paste the Add-On Instructions and configure the instance name along with the input and output parameters. Refer to Section 2 for additional instructions for configuring the Message parameters. Each device must correspond to a unique instance of the AOI. As shown in the figure below, both devices call the AOI SAI_CheckAlive, but the corresponding instances are SAI_CheckAlive and SAI_CheckAlive_1. Notice that the Heartbeat parameter is also

configured with different devices for these two instances. See the Add-On Instructions section of this note for information on configuring parameters for a particular AOI. Make sure that all tags for the second device for instance now end in "_2" as opposed to "_1" for the first device.
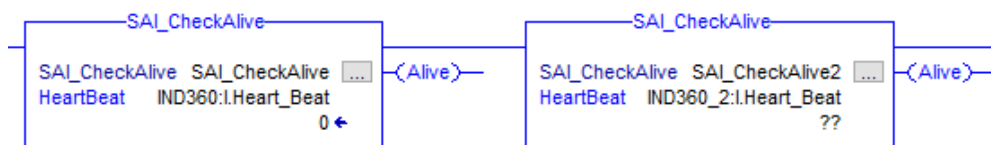
Figure 3-6: Two instances of the SAI_CheckAlive AOI for two IND360s

7) Repeat steps 1 to 6 until the configuration of all devices is completed.

4. Steps to Use 8 Block Format Instead of 2 Block Format

The 8 Block Format for SAI is incredibly powerful for viewing more information simultaneously compared to the 2 Block Format. The sample code by default is configured for the 2 Block Format, but changing the format is incredibly simple.

a) Right click on the IND360 in the Controller Organizer
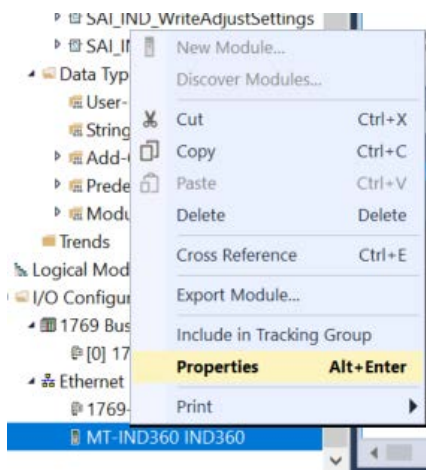
b) Click "Properties"

Figure 4-1 Select Properties

c) Click "Change" under the Module Definition
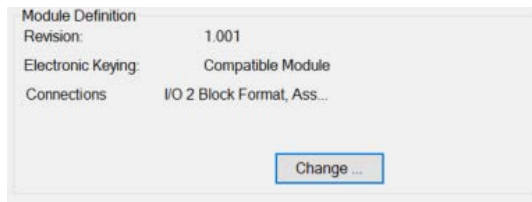
Figure 4-2 Click "Change"

d) Select the drop-down arrow next to "I/O 2 Block Format" and Select "I/O 8 Block Format
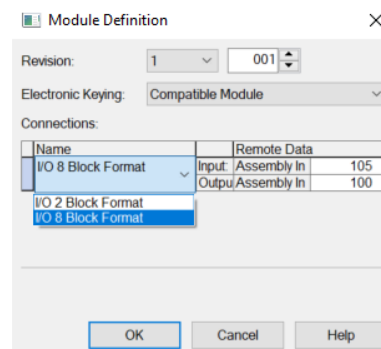
Figure 4-3 Select "I/O 8 Block Format"

At this point, the project has the IND360 configured for the 8 block format. No changes to the AOIs are required since the 8 Block Format just expands upon the 2 Block format utilized by the AOIs. There is now an additional AOI in the sample project that can be used though. The final step is to confirm the IND360 itself is configured for the 8 block format. This setting can be found in the IND360 setup menu or via the web interface at Communication -> Industrial Ethernet -> Format.

4.1. Add-On Instruction for 8 Block Format

The AOI for the 8 block format is available in the controller organizer under Assets -> Add-On Instructions -> SAI_IND_8Block. This AOI can be dragged from the controller organizer into the "MT_IND_360" routine. Information for configuring the AOI can be found below.

In the 8 block format, there is one measuring block and seven status blocks. The first measuring block and status block are also used in the 2 block format. What is gained by changing to the 8 block format is measuring blocks 2-7.

This AOI allows the user to enter a cyclic command (MBx_Command_In) and an optional parameter associated with the command (MBx_Value_In), see the result from the command (MBx_Measuring Value), whether the command executed (Done_MBx) and if there was an error executing the command (Error_MBx).

Figure 4-4 SAI_IND_8Block

Input Parameters	Data Type	Description
MB2_Measuring_Value	Real	This should always be set to the MB2_Measuring_Value of the IND360. Measured value returned from the last successfully executed command for MB2. Ignore value if neither Done_MB2 nor Error_MB2 is set since that means a command is in the process of executing.
MB3_Measuring_Value	Real	This should always be set to the MB3_Measuring_Value of the IND360. Measured value returned from the last successfully executed command for MB3. Ignore value if neither Done_MB3 nor Error_MB3 is set since that means a command is in the process of executing.
MB4_Measuring_Value	Real	This should always be set to the MB4_Measuring_Value of the IND360. Measured value returned from the last successfully executed command for MB4. Ignore value if neither Done_MB4 nor Error_MB4 is set since that means a command is in the process of executing.

MB5_Measuring_Value	Real	This should always be set to the MB5_Measuring_Value of the IND360. Measured value returned from the last successfully executed command for MB5. Ignore value if neither Done_MB5 nor Error_MB5 is set since that means a command is in the process of executing.
MB6_Measuring_Value	Real	This should always be set to the MB6_Measuring_Value of the IND360. Measured value returned from the last successfully executed command for MB6. Ignore value if neither Done_MB6 nor Error_MB6 is set since that means a command is in the process of executing.
MB7_Measuring_Value	Real	This should always be set to the MB7_Measuring_Value of the IND360. Measured value returned from the last successfully executed command for MB7. Ignore value if neither Done_MB7 nor Error_MB7 is set since that means a command is in the process of executing.
MB2_Command_In	INT	Place the cyclic command to be executed using MB2 here. See the SAI manual for terminals and transmitters to see all valid cyclic commands. Data reported will be found in MB2_Measuring Value. Optional parameter information should be placed in MB2_Value_In before changing the command. Some examples of common cyclic commands: 0 or 1 = Report gross weight 2 = Report tare weight 3 = Report net weight 5 = Report gross weight value (with internal resolution) 6 = Report tare weight value (with internal resolution) 7 = Report net weight value (with internal resolution)
MB3_Command_In	INT	Place the cyclic command to be executed using MB3 here. See the SAI manual for terminals and transmitters to see all valid cyclic commands. Data reported will be found in MB3_Measuring Value. Optional parameter information should be placed in MB3_Value_In before changing the command. Some examples of common cyclic commands: 0 or 1 = Report gross weight 2 = Report tare weight 3 = Report net weight 5 = Report gross weight value (with internal resolution) 6 = Report tare weight value (with internal resolution) 7 = Report net weight value (with internal resolution)
MB4_Command_In	INT	Place the cyclic command to be executed using MB4 here. See the SAI manual for terminals and transmitters to see all valid cyclic commands. Data reported will be found in MB4_Measuring Value. Optional parameter information should be placed in MB4_Value_In before changing the command. Some examples of common cyclic commands: 0 or 1 = Report gross weight 2 = Report tare weight 3 = Report net weight 5 = Report gross weight value (with internal resolution) 6 = Report tare weight value (with internal resolution) 7 = Report net weight value (with internal resolution)

MB5_Command_In	INT	Place the cyclic command to be executed using MB5 here. See the SAI manual for terminals and transmitters to see all valid cyclic commands. Data reported will be found in MB5_Measuring Value. Optional parameter information should be placed in MB5_Value_In before changing the command. Some examples of common cyclic commands: 0 or 1 = Report gross weight 2 = Report tare weight 3 = Report net weight 5 = Report gross weight value (with internal resolution) 6 = Report tare weight value (with internal resolution) 7 = Report net weight value (with internal resolution)
MB6_Command_In	INT	Place the cyclic command to be executed using MB6 here. See the SAI manual for terminals and transmitters to see all valid cyclic commands. Data reported will be found in MB6_Measuring Value. Optional parameter information should be placed in MB6_Value_In before changing the command. Some examples of common cyclic commands: 0 or 1 = Report gross weight 2 = Report tare weight 3 = Report net weight 5 = Report gross weight value (with internal resolution) 6 = Report tare weight value (with internal resolution) 7 = Report net weight value (with internal resolution)
MB7_Command_In	INT	Place the cyclic command to be executed using MB7 here. See the SAI manual for terminals and transmitters to see all valid cyclic commands. Data reported will be found in MB7_Measuring Value. Optional parameter information should be placed in MB7_Value_In before changing the command. Some examples of common cyclic commands: 0 or 1 = Report gross weight 2 = Report tare weight 3 = Report net weight 5 = Report gross weight value (with internal resolution) 6 = Report tare weight value (with internal resolution) 7 = Report net weight value (with internal resolution)
MB2_Value_In	Real	Place the optional parameter associated with MB2_Command_In here. For example, if executing a pretare command, MB2_Value_In would be set to the pretare value before changing MB2_Command_In
MB3_Value_In	Real	Place the optional parameter associated with MB3_Command_In here. For example, if executing a pretare command, MB3_Value_In would be set to the pretare value before changing MB3_Command_In
MB4_Value_In	Real	Place the optional parameter associated with MB4_Command_In here. For example, if executing a pretare command, MB4_Value_In would be set to the pretare value before changing MB4_Command_In
MB5_Value_In	Real	Place the optional parameter associated with MB5_Command_In here. For example, if executing a pretare command, MB5_Value_In would be set to the pretare value before changing MB2_Command_In
MB6_Value_In	Real	Place the optional parameter associated with MB6_Command_In here. For example, if executing a pretare command, MB6_Value_In would be set to the pretare value before changing MB6_Command_In

| MB7_Value_In	Real	Place the optional parameter associated with MB7_Command_In here. For example, if executing a pretare command, MB7_Value_In would be set to the pretare value before changing MB7_Command_In
MB2_Response	INT	This should always be set to MB2_Response value of the IND360. Once a cyclic command is successfully executed, MB2_Response = MB2_Command_In. The AOI uses this information to detect if a command has been executed successfully or if an error has occurred.
MB3_Response	INT	This should always be set to MB3_Response value of the IND360. Once a cyclic command is successfully executed, MB3_Response = MB3_Command_In. The AOI uses this information to detect if a command has been executed successfully or if an error has occurred.
MB4_Response	INT	This should always be set to MB4_Response value of the IND360. Once a cyclic command is successfully executed, MB4_Response = MB4_Command_In. The AOI uses this information to detect if a command has been executed successfully or if an error has occurred.
MB5_Response	INT	This should always be set to MB5_Response value of the IND360. Once a cyclic command is successfully executed, MB5_Response = MB5_Command_In. The AOI uses this information to detect if a command has been executed successfully or if an error has occurred.
MB6_Response	INT	This should always be set to MB6_Response value of the IND360. Once a cyclic command is successfully executed, MB6_Response = MB6_Command_In. The AOI uses this information to detect if a command has been executed successfully or if an error has occurred.
MB7_Response	INT	This should always be set to MB7_Response value of the IND360. Once a cyclic command is successfully executed, MB7_Response = MB7_Command_In. The AOI uses this information to detect if a command has been executed successfully or if an error has occurred.
Output Parameters	Data Type	Description
MB2_Command_Out	INT	This should always be set to MB2_Command of the IND360. Value of the last cyclic command sent to the IND360. Once successfully executed, MB2_Response = MB2_Command_Out = MB2_Command_In.
MB3_Command_Out	INT	This should always be set to MB3_Command of the IND360. Value of the last cyclic command sent to the IND360. Once successfully executed, MB3_Response = MB3_Command_Out = MB3_Command_In.
MB4_Command_Out	INT	This should always be set to MB4_Command of the IND360. Value of the last cyclic command sent to the IND360. Once successfully executed, MB4_Response = MB4_Command_Out = MB4_Command_In.
MB5_Command_Out	INT	This should always be set to MB5_Command of the IND360. Value of the last cyclic command sent to the IND360. Once successfully executed, MB5_Response = MB5_Command_Out = MB5_Command_In.
MB6_Command_Out	INT	This should always be set to MB6_Command of the IND360. Value of the last cyclic command sent to the IND360. Once successfully executed, MB6_Response = MB6_Command_Out = MB6_Command_In.
MB7_Command_Out	INT	This should always be set to MB7_Command of the IND360. Value of the last cyclic command sent to the IND360. Once successfully executed, MB7_Response = MB7_Command_Out = MB7_Command_In.

MB2_CmdValue_Out	Real	This should always be set to MB2_Command_Value of IND360
MB3_CmdValue_Out	Real	This should always be set to MB3_Command_Value of IND360
MB4_CmdValue_Out	Real	This should always be set to MB4_Command_Value of IND360
MB5_CmdValue_Out	Real	This should always be set to MB5_Command_Value of IND360
MB6_CmdValue_Out	Real	This should always be set to MB6_Command_Value of IND360
MB7_CmdValue_Out	Real	This should always be set to MB7_Command_Value of IND360
Done_MB2	BOOL	Will be latched high when command has successfully completed for MB2. When a new command begins, bit will be unlatched until command completes successfully
Done_MB3	BOOL	Will be latched high when command has successfully completed for MB3. When a new command begins, bit will be unlatched until command completes successfully
Done_MB4	BOOL	Will be latched high when command has successfully completed for MB4. When a new command begins, bit will be unlatched until command completes successfully
Done_MB5	BOOL	Will be latched high when command has successfully completed for MB5. When a new command begins, bit will be unlatched until command completes successfully
Done_MB6	BOOL	Will be latched high when command has successfully completed for MB6. When a new command begins, bit will be unlatched until command completes successfully
Done_MB7	BOOL	Will be latched high when command has successfully completed for MB7. When a new command begins, bit will be unlatched until command completes successfully
Error_MB2	BOOL	Will be latched high when command fails to complete for MB2. When a new command begins, bit will be unlatched until a command fails to complete.
Error_MB3	BOOL	Will be latched high when command fails to complete for MB3. When a new command begins, bit will be unlatched until a command fails to complete.
Error_MB4	BOOL	Will be latched high when command fails to complete for MB4. When a new command begins, bit will be unlatched until a command fails to complete.
Error_MB5	BOOL	Will be latched high when command fails to complete for MB5. When a new command begins, bit will be unlatched until a command fails to complete.
Error_MB6	BOOL	Will be latched high when command fails to complete for MB6. When a new command begins, bit will be unlatched until a command fails to complete.
Error_MB7	BOOL	Will be latched high when command fails to complete for MB7. When a new command begins, bit will be unlatched until a command fails to complete.

5.   Frequently Asked Questions

1.   Q: How do I access the parameters in the AOI variables within my PLC program?

A: You can use the format "instance_name.parameter" to access parameters in your PLC program. For example, if we create an instance of the SAI_CheckAlive AOI and name the instance "IND360_Comm", we can monitor the alive bit by looking at "IND360_Comm.Alive"
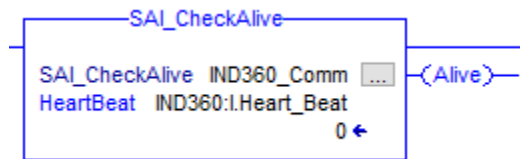
Figure 4-1: SAI_CheckAlive AOI with different instance name

2. Q: Does my AOI instance always have to match the name of the AOI?

    A: No, the AOI instance can be named anything as long as the name is unique. They must be unique so that if we are using multiple of the same AOI, we can distinguish between them in the code. See Figure 4-1 for an example of an AOI instance name that does not match the AOI name but is still valid.

3. Q: How do I read gross, tare or net weight?

    A: Use the WeightCmd parameter of the SAI_INDPCELL_WeightProc AOI to issue different weight commands. A value of 0 = report gross weight, 2 = report tare weight and 3 = report net weight. The weight will be updated in StableWeight if the weight value is stable. StableWeight will freeze with the last reported stable weight while the scale is in motion. RealTimeWeight will constantly update the weight value regardless of whether the scale is in motion.

4. Q: How do I know the source of the error in the SAI_INDPCELL_WeightProc AOI?

    A: Typical errors in this AOI include:

    o   Pushbutton zero failure -> tried to zero when the weight value is outside of the pushbutton zero range (+/- 2% by default). If a substantial zero adjustment needs to be made, use the SAI_IND_ZeroAdjust AOI instead.

    o   Tare failure -> Typically seen if trying to tare a negative gross weight value. Try to tare again with a positive gross weight.

    o   Stability failure -> Can occur with either the ZeroStable or Tare Stable command. The weight value must meet the stability criteria (no more than 1 division of change occurring in a 0.3 second period by default) at some point before a timeout occurs (3 seconds by default).

5. Q: How do I know if the scale is overloaded or underloaded?

    A: Monitor the Overload and Underload outputs from the SAI_INDPCELL_DiagnosticStatus AOI. This AOI is also useful for monitoring many important potential errors in the weighing system.

6. Q: An AOI is very close to what I want to do in my PLC logic, but I need to make a few changes. How can I do that?

A: If necessary to view or modify the logic of an AOI, simply use the Controller Organizer view in Studio 5000 to navigate to Add-On Instructions, expand the AOI you are interested in viewing and double-click "Logic". This will show you the ladder logic used in the AOI and can be changed as necessary for your particular application.
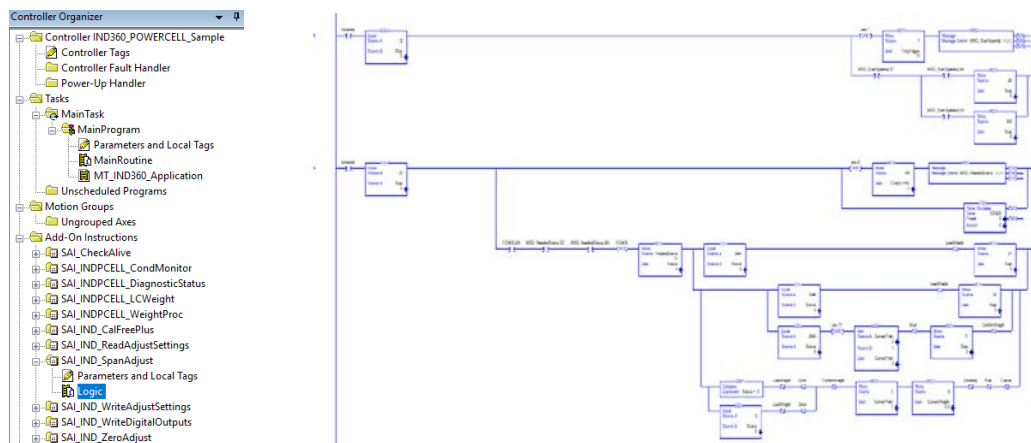

Figure 4-2: Example of AOI ladder logic