Title: Cognex In-Sight Explorer 7000 Series Overview	
Product(s): Cognex In-Sight Explorer 7000 Series, MP3300iec, MotionWorks IEC	**Doc. No.** AN.MPIEC.15

Cognex In-Sight Explorer 7000 Series

YASKAWA

Title: Cognex In-Sight Explorer 7000 Series Overview

Product(s): Cognex In-Sight Explorer 7000 Series, MP3300iec, MotionWorks IEC	**Doc. No.** AN.MPIEC.15

Table of Contents

1 Purpose

This application note provides a brief summary of the configuration and programming environment for the Cognex vision system. It will conclude by walking through the creation of a simple Cognex vision application. This simple vision application will locate parts in the vision sensor's field of view and transfer the part's location (i.e. X, Y, and Angle) information to a remote system via Ethernet/IP.

2 Cognex Hardware Components

Item	**Part Number**
Vision Sensor (i.e. Camera)	821-0084-5R
I/O Cable	CCB-PWRIO-05
Ethernet Cable	CCB-84901-1003-05

3 The Cognex IDE - In-Sight Explorer

This application note uses In-Sight Explorer version 4.8.4, and the firmware of the vision sensor has version 4.8.5.

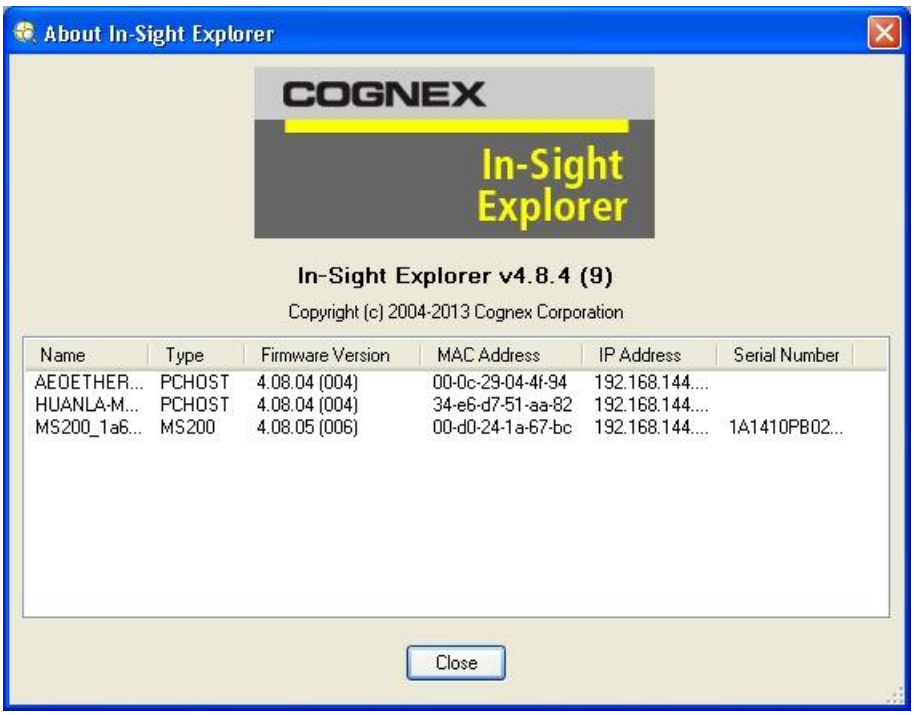

Figure 1: In-Sight Explorer Version 4.8.4.

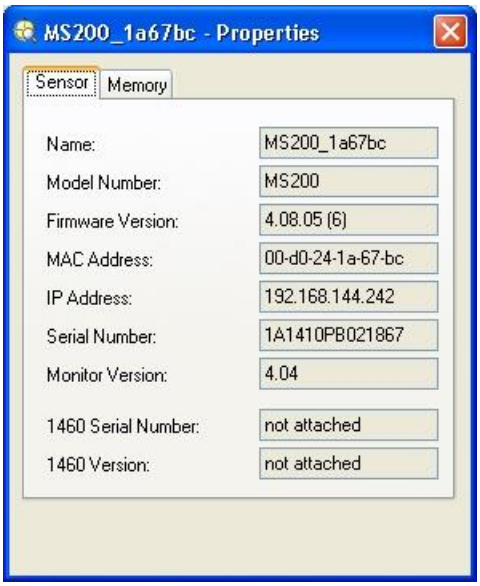

Figure 2: In-Sight 7000 Series Camera Firmware Version 4.8.5.

When In-Sight Explorer is opened for the first time, it will probably be in EasyBuilder mode (See section 3.4 for a description of In-Sight Explorer programming modes).

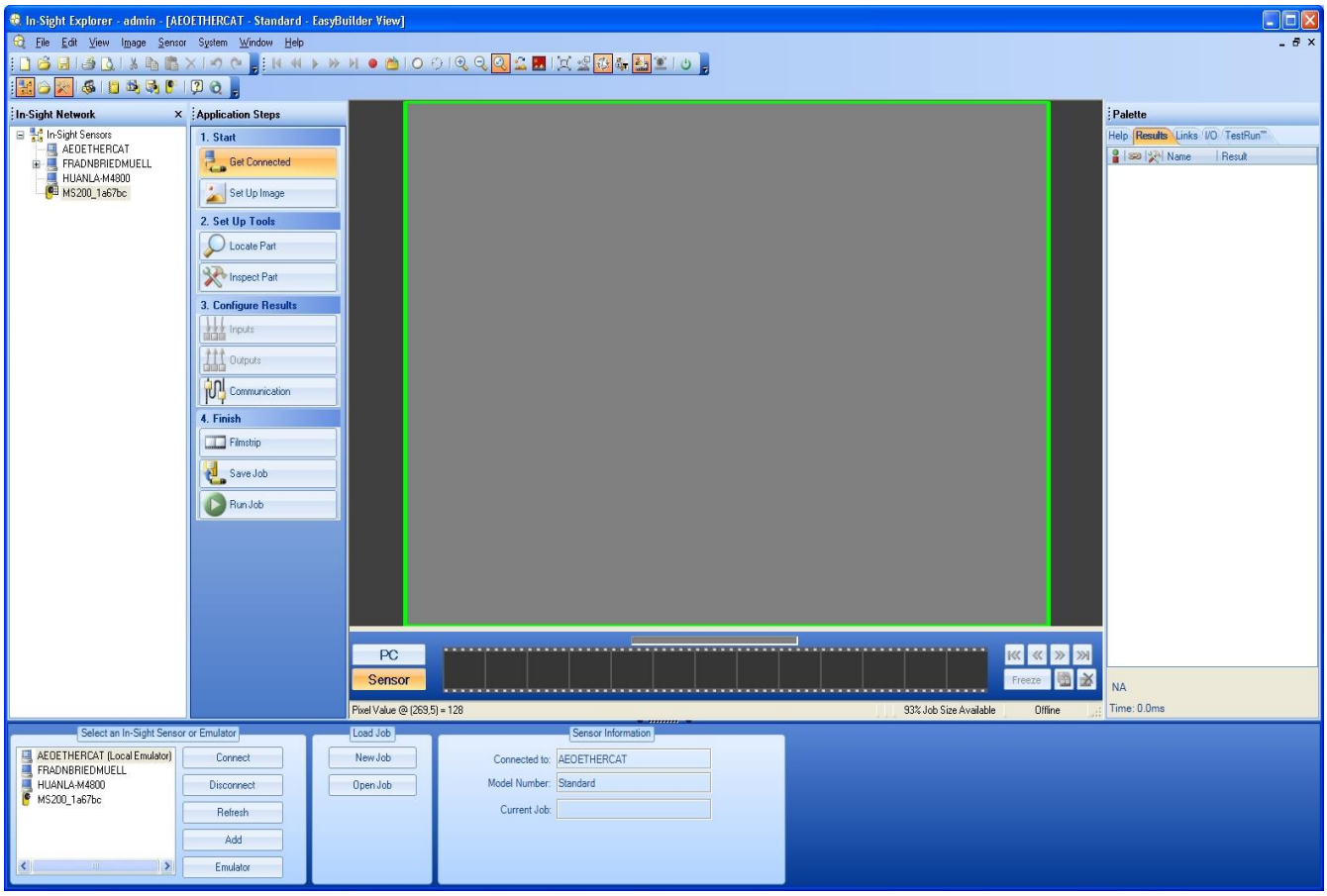

Figure 3: In-Sight Explorer in EasyBuilder Mode.

Set In-Sight Explorer to Spreadsheet mode; in the In-Sight Explorer menu bar use the item "Windows -> Show Spreadsheet View" to switch to spreadsheet mode. In-Sight Explorer files are called "jobs". Once in Spreadsheet mode, open a new job file using the menu item, "File -> New Job". In the "New Job" dialog that pops open, click the "Yes" button to "... clear all data from current job." Figure 4 below is a screenshot of a new job file in Spreadsheet mode.

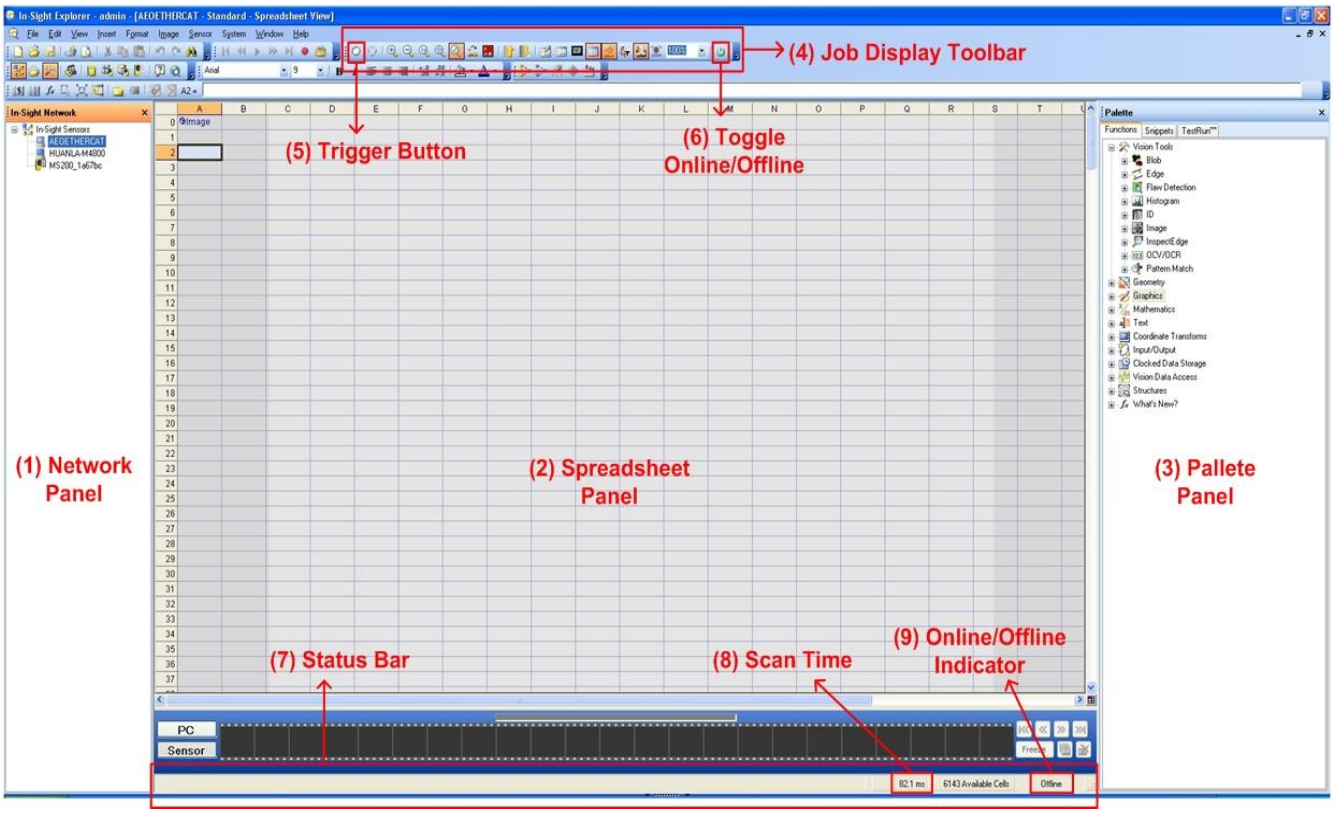

Figure 4: New Job in Spreadsheet Mode of In-Sight Explorer.

Things to note from Figure 4:

- Network panel: displays all the Cognex vision sensors and emulators that exist on the network.
- Palette panel: A user creates a vision program by dragging and dropping items from this panel into the Spreadsheet panel.
- Spreadsheet panel: cell A0 contains the "AcquireImage" function. This function gets the image from the camera and makes it available to other vision functions in the program.

3.1 The Network Panel

In-Sight Explorer communicates with other Cognex devices using Ethernet. In-Sight Explorer auto detects all the Cognex sensors and emulators that are connected on the same local area network.

In the Network panel, a user can double click on the icon for a Cognex device to connect it.

Once connected to a device, its network properties (e.g. IP Address) can be set using the menu bar item: Sensor -> Network Settings...

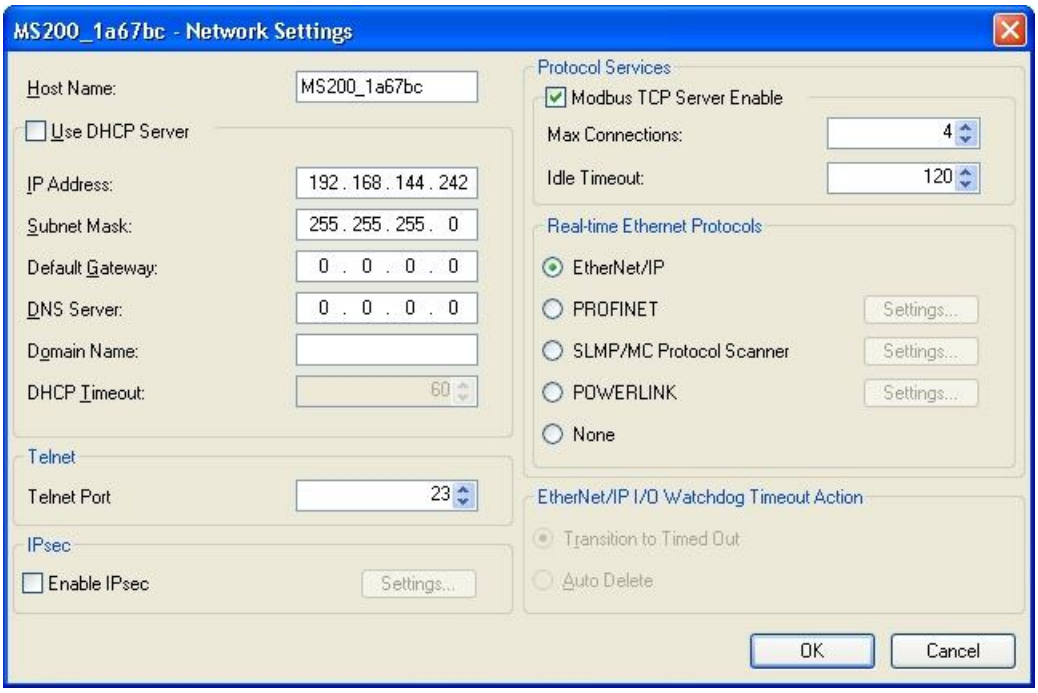

Figure 5: In-Sight Camera Network Settings Dialog.

3.2 The Palette Panel

This panel contains logic and vision functions that are used to compose vision programs.

- The Functions tab lists individual vision functions.
- The Snippets tab lists groups of functions for frequently performed vision tasks.
- The TestRun tab is used to configure tests on the vision application.

3.3 The Spreadsheet Panel

This area of In-Sight explorer contains the vision program. Drag and drop functions from the Palette panel into the spreadsheet cells. The cell execution order is left-to-right, top-to-bottom.

3.4 In-Sight Explorer Programming Modes: Spreadsheet vs. EasyBuilder

In-Sight explorer offers two programming modes: Spreadsheet and EasyBuilder modes. The Spreadsheet programming mode mimics a standard spreadsheet application. The vision program is organized around placing vision functions into spreadsheet cells and later referencing the function's output in other cells.

The EasyBuilder programming mode is a higher level abstraction built on the spreadsheet. In this mode the vision program is specified using a Wizard GUI that walks the user through the steps needed to create a vision application. An EasyBuilder program has an underlying spreadsheet which can be later accessed for further program customization if necessary.

In the In-Sight Explorer menu bar use the items "Windows -> Show Spreadsheet View", and "Windows -> Show EasyBuilder View" to switch between the Spreadsheet and EasyBuilder modes.

3.5 In-Sight Explorer Help

Help documentation can be accessed from the In-Sight explorer menu bar: "Help -> In-Sight Explorer Help". Also, context sensitive help can accessed by selecting an item in In-Sight Explorer then clicking the F1 key on the keyboard. This will navigate to the help documentation for the selected item.

4 Sample Cognex Application

The figure below is a screenshot of the sample vision application used for this application note. The application is written in Spreadsheet mode. The application identifies the location of a pre-defined pattern in the camera's field of view. This location information is then transferred to a remote system (i.e. MPiec controller) using implicit Ethernet/IP messaging.

A0 = AcquireImage(6,1,8,0,950,10,0,0,600,800,0,600,2,1,0,0,0,0,128,0,0,"","",0,0,140,180,320,440,0,0,0,0,600,800,1,40,0,0,1)

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
0	Image													
1														
2		Index	Row	Col	X	Y	Score	Area	Elongation	Holes	Perimeter	Spread		
3	Blobs	0.000	30.101	457.289	177.800	152.400	100.000	357.000	0.000	0.000	86.000	0.165		
4		1.000	252.253	680.596	0.000	330.200	100.000	227.000	0.000	0.000	74.000	0.159		
5		2.000	33.207	677.051	177.800	330.200	100.000	285.000	0.000	0.000	76.000	0.159		
6		3.000	252.935	233.644	0.000	0.000	100.000	275.000	0.000	0.000	76.000	0.159		
7		4.000	30.673	235.011	406.400	0.000	100.000	275.000	0.000	0.000	76.000	0.159		
8														
9	Calib													
10														
11														
12	Image													
13														
14		Index	Row	Col	Angle	Scale	Score			**Found**	**X**	**Y**	**Angle**	
15	Patterns	0.000	29.534	176.618	18.867	89.844	52.891			1.000	29.534	176.618	18.867	
16		1.000	-7.707	258.058	-58.783	112.109	50.633			1.000	-7.707	258.058	-58.783	
17		2.000	#ERR	#ERR	#ERR	#ERR	0.000			0.000	0.000	0.000	0.000	
18		3.000	#ERR	#ERR	#ERR	#ERR	0.000			0.000	0.000	0.000	0.000	
19		4.000	#ERR	#ERR	#ERR	#ERR	0.000			0.000	0.000	0.000	0.000	
20		5.000	#ERR	#ERR	#ERR	#ERR	0.000			0.000	0.000	0.000	0.000	
21		6.000	#ERR	#ERR	#ERR	#ERR	0.000			0.000	0.000	0.000	0.000	
22		7.000	#ERR	#ERR	#ERR	#ERR	0.000			0.000	0.000	0.000	0.000	
23		8.000	#ERR	#ERR	#ERR	#ERR	0.000			0.000	0.000	0.000	0.000	
24														
25									**Number_Found**	2.000				
26										Buffer	Buffer	Buffer	Buffer	
27														
28	Buffer	28.000												

Figure 6: Vision Application in Spreadsheet Panel (See file AN_MPIEC_15.job in AN.MPIEC.15.zip).

The functions used in the application are accessible from the Palette panel, under the Functions tab. Once placed in cells on the spreadsheet, the functions can be customized by editing their properties.

Each step of the application and the functions used to implement the steps are described below.

4.1 Acquire an Image - AcquireImage Function

By default, cell A0 of every new spreadsheet file is populated with the AcquireImage function. This function gets the image data from the camera. Since the function is in cell A0, it will be the first function executed when the vision program is run.

Double click on cell A0, to open a dialog box that lists the properties of this function.

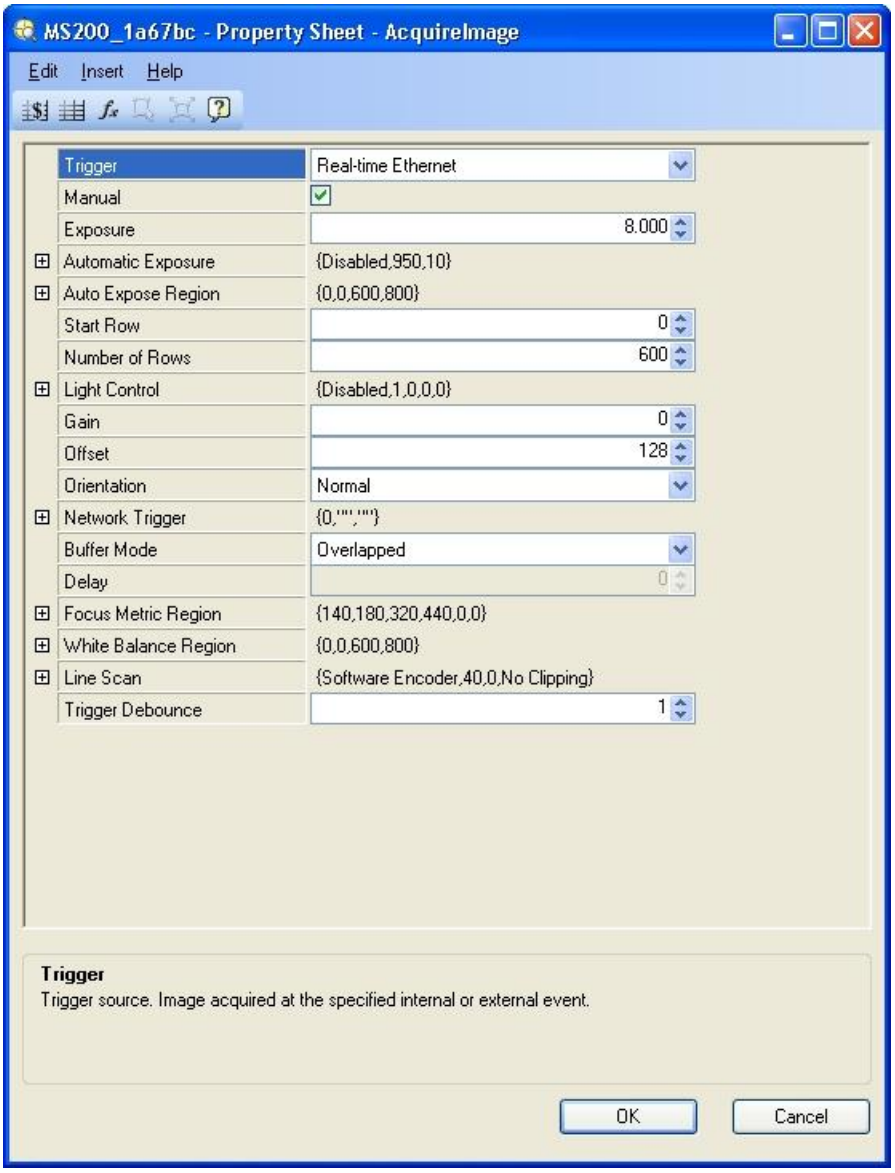

Figure 7: AcquireImage Function Properties Dialog.

The AcquireImage function properties can be edited to customize it for a particular application. For example, the trigger source for acquiring an image from the camera can be specified here.

See the help documentation for the AcquireImage function for a detailed description of all its properties and their possible values.

4.2 Calibration - ExtractBlobs + CalibrateAdvanced + CalibrateImage Functions

The goal of the calibration step is to define how to convert pixel coordinates into real world coordinates. This is accomplished using a group of functions: ExtractBlobs, CalibrateAdvanced, and CalibrateImage.

The ExtractBlobs function (Figure 6, cell A3) will find the pixel location of points in a calibration grid placed in the camera's field of view. The values under the "Row" and "Col" headers (i.e. cells, C3:D7) are the pixel locations of the calibration grid points that were identified by the ExtractBlobs function. The values under the "X" and "Y" headers (i.e. cell range, E3:F7) are the real world coordinates that correspond to the image pixel coordinates. The real world coordinate values are entered manually into the spreadsheet.

The CalibrateAdvanced function (Figure 6, cell A9) encapsulates the calibration information in cells C3:F7.

The CalibrateImage function (Figure 6, cell A12) establishes a relationship between the raw image from the camera (Figure 6, cell A0) and the calibration information from the CalibrateAdvanced function. Other vision functions in the application can reference the calibrated image in A12, as opposed to the raw image in A0, to output their results in real world coordinates instead of pixel coordinates.

4.2.1 Cell State

The functions in the calibration step should only execute during application development. When the application is deployed the calibration grid is removed from the camera's field of view, and the ExtractBlobs function will no longer output any useful results. Furthermore, we don't want to change the calibration information that is already encapsulated in the calibrated image. So, the ExtractBlobs should be disabled when the application is deployed.

A vision function can be disabled by setting the state of the cell that it occupies to "Disabled". For example, the ExtractBlobs function is disabled by right clicking on cell A3 and selecting the option "Cell State..." from the context menu. In the "Cell State" dialog box that opens, select the "Disable" option and click the "OK" button.

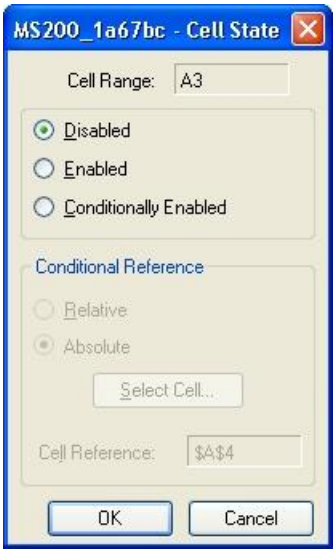

Figure 8: Enable/Disable a Spreadsheet Cell with the Cell State Dialog.

Notice that in Figure 6, cell A3 is grayed out; this indicates that the cell (therefore the ExtractBlobs function) is disabled.

4.3 Find Patterns - FindPatterns Function

The FindPatterns function (Figure 6, cell A15) will identify parts in the camera's field of view and output the location and orientation (X, Y, Angle) for an identified part. To customize the function, double click on the cell that the function occupies to open its properties dialog box.

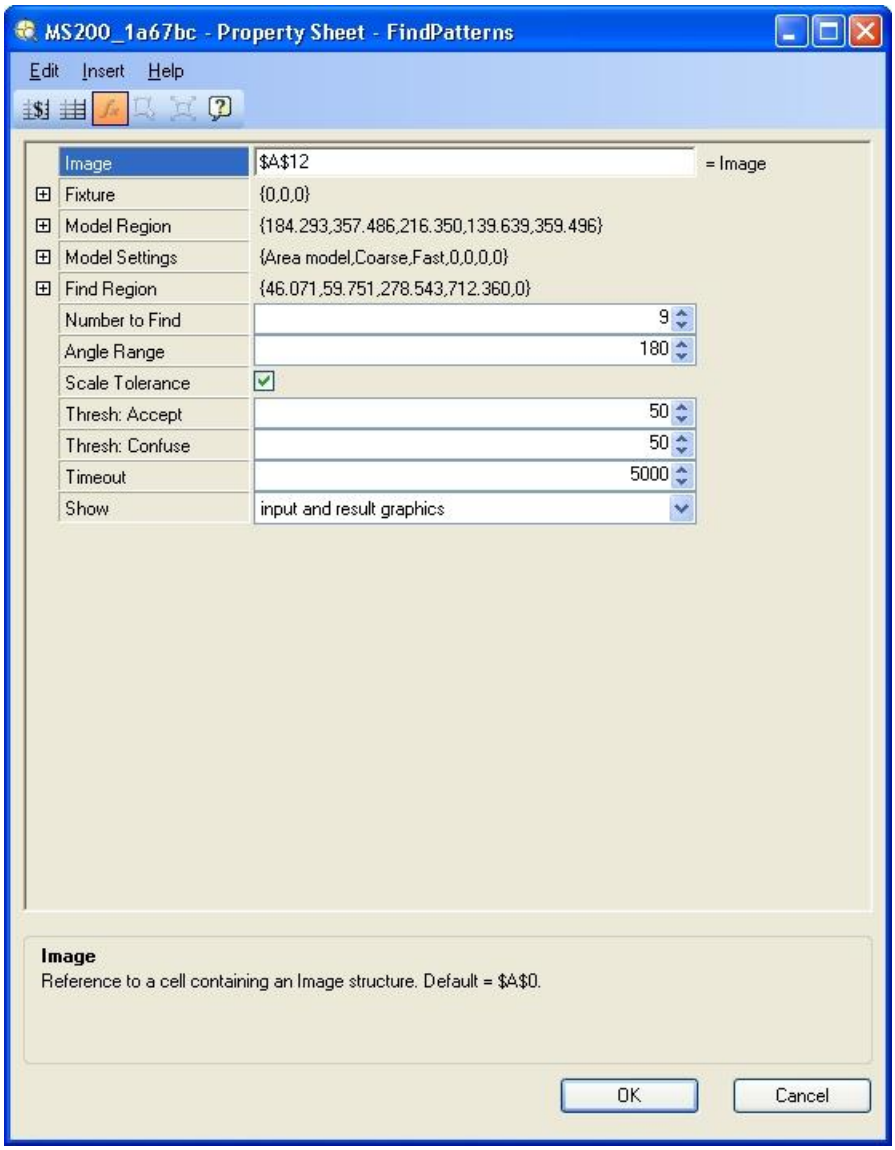

Figure 9: FindPatterns Function Properties Dialog.

At a minimum the application should specify the following FindPatterns properties: Image, Model Region, and Find Region.

4.3.1 FindPatterns Properties

- Image: The image to process; in this case we are using the calibrated image in A12 (not the raw image in A0).
- Model Region: Defines a pattern of interest in the image.
- Find Region: The area in camera's field of view in which to search for the pattern defined by the Model Region.

The specified property values can have a big impact on the function's processing time and performance. So, the vision application developer typically needs to experiment with different property values to find the combination of values that work best for a particular application context.

See the help documentation for more details on the FindPatterns function and its properties.

4.3.2 FindPatterns Results

Figure 6 shows the output of the FindPatterns function in cell range B15:G23. The X and Y values are in real world coordinates since this function references the calibrated image in cell A12.

In this sample application, FindPatterns is configured to look for at most 9 instances of the trained pattern; Figure 6 shows that FindPatterns located two instances of the trained pattern.

The FindPatterns function outputs are formatted for Ethernet/IP transfer in cells J15:M23, and cell J26.

The values in the "Found" column (J15:J23) are set to "1" if the corresponding "Score" values (G15:G23) are greater than "50", or the values are set to "0" otherwise.

The value in cell J26 is the total number instances of the trained pattern identified by the FindPatterns function.

The values in the X, Y, and Angle columns (K15:M23) are set to "0" if the corresponding values in the Row, Col, and Angle columns (C15:E15) are set to "ERR". If the Row, Col, or Angle value is not "ERR" (i.e. an actual number), then the value is copied to the corresponding X, Y, or Angle cells (K15:M23).

4.4 Ethernet/IP - FormatOutputBuffer + CombineOutputBuffers + WriteEIPBuffer Functions

The FormatOutputBuffer (J26:M26) and CombineOutputBuffers (A28) functions are used to populate the Ethernet/IP buffer with the formatted results of the FindPatterns functions.

Each cell in the range J26:M26 contains a FormatOutputBuffer function that packages the formatted FindPatterns result in its corresponding column. Double click on a cell containing the FormatOutputBuffer function to open a dialog box that allows you to specify the information that goes into the buffer.

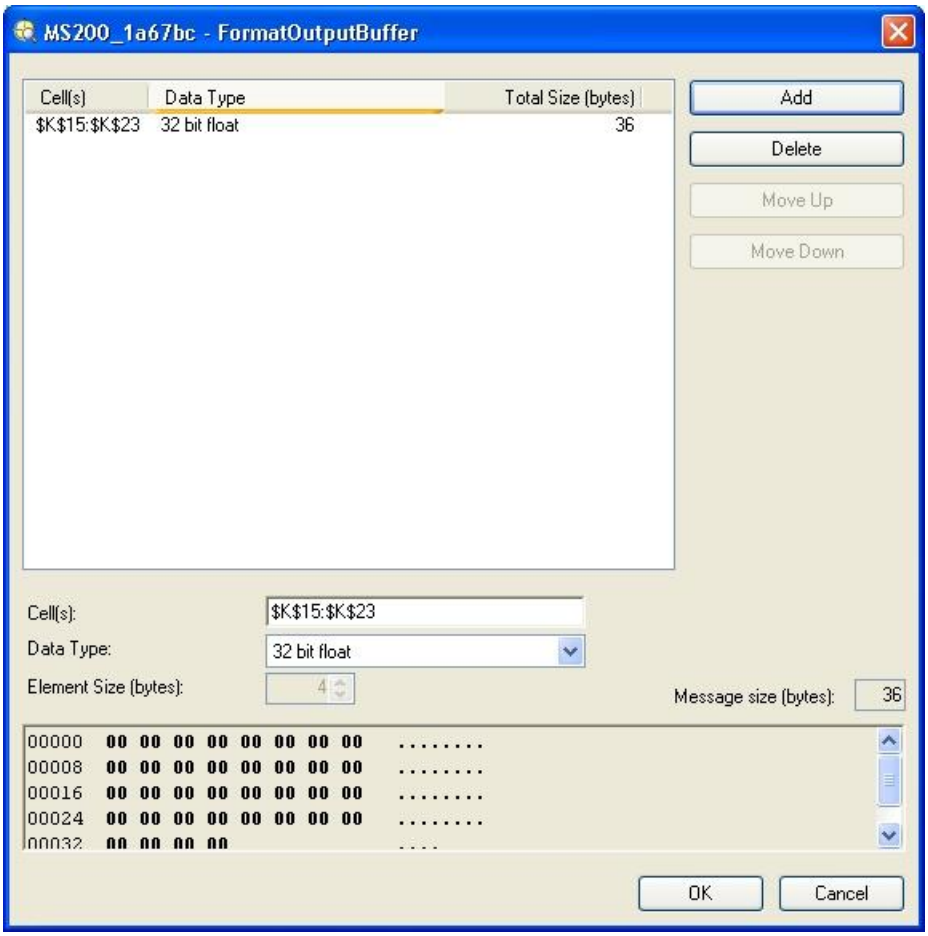

Figure 10: Add the value of the X coordinates, cells K15:K23 in Figure 6, to this Output Buffer

The FormatOutputBuffer function supports a maximum of 248 characters in its spreadsheet formula. So, there is a limit to how many cells you can specify in its formula. So, multiple FormatOutputBuffer functions are used to reduce the number of cells each function will reference.

This is the reason the CombineOutputBuffers function in needed. The CombineOutputBuffers function aggregates the information in each of the output buffers defined in cells J26:M26.

The WriteEIPBuffer (B28) function writes the content of the combined buffer into the output Ethernet/IP buffer of the Cognex system. The Cognex system then makes this data available to a remote system via an implicit Ethernet/IP write. The value in cell B28 indicates the number of cells that the WriteEIPBuffer function writes to the Ethernet/IP buffer. In this example (see Figure 6) the cells written to the Ethernet/IP buffer are cells J25 and K15:M23 which is a total of 28 cells.

5 Sample MPiec Controller Application

5.1 MPiec Controller Ethernet/IP Configuration

Figure 11 and Figure 12 show how to add the Cognex In-Sight system as an Ethernet/IP adapter in the hardware configuration tool for an MPiec controller.

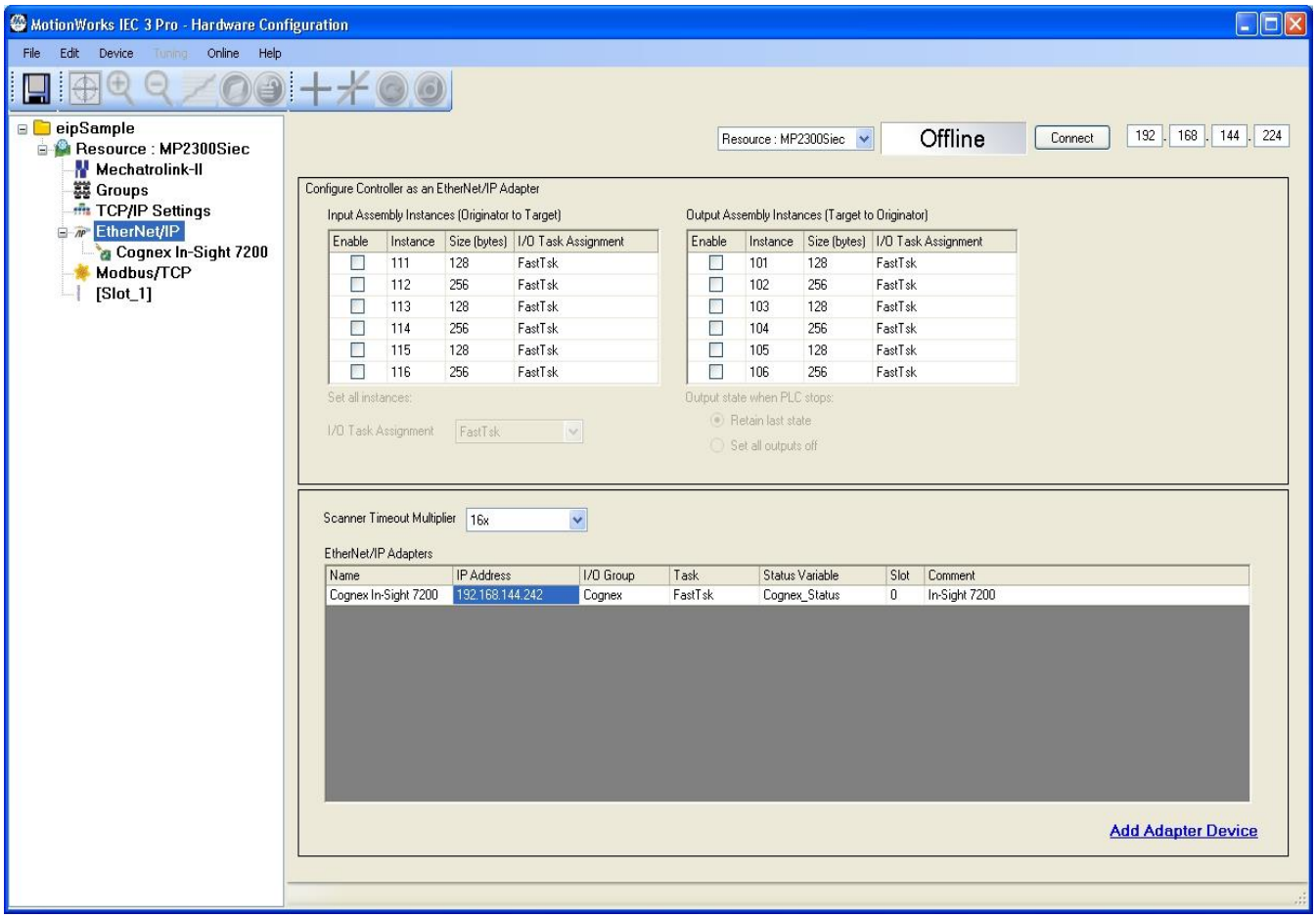

Figure 11: Add Ethernet/IP Adapter Device for Cognex System in MotionWorks IEC Hardware Configuration Tool.

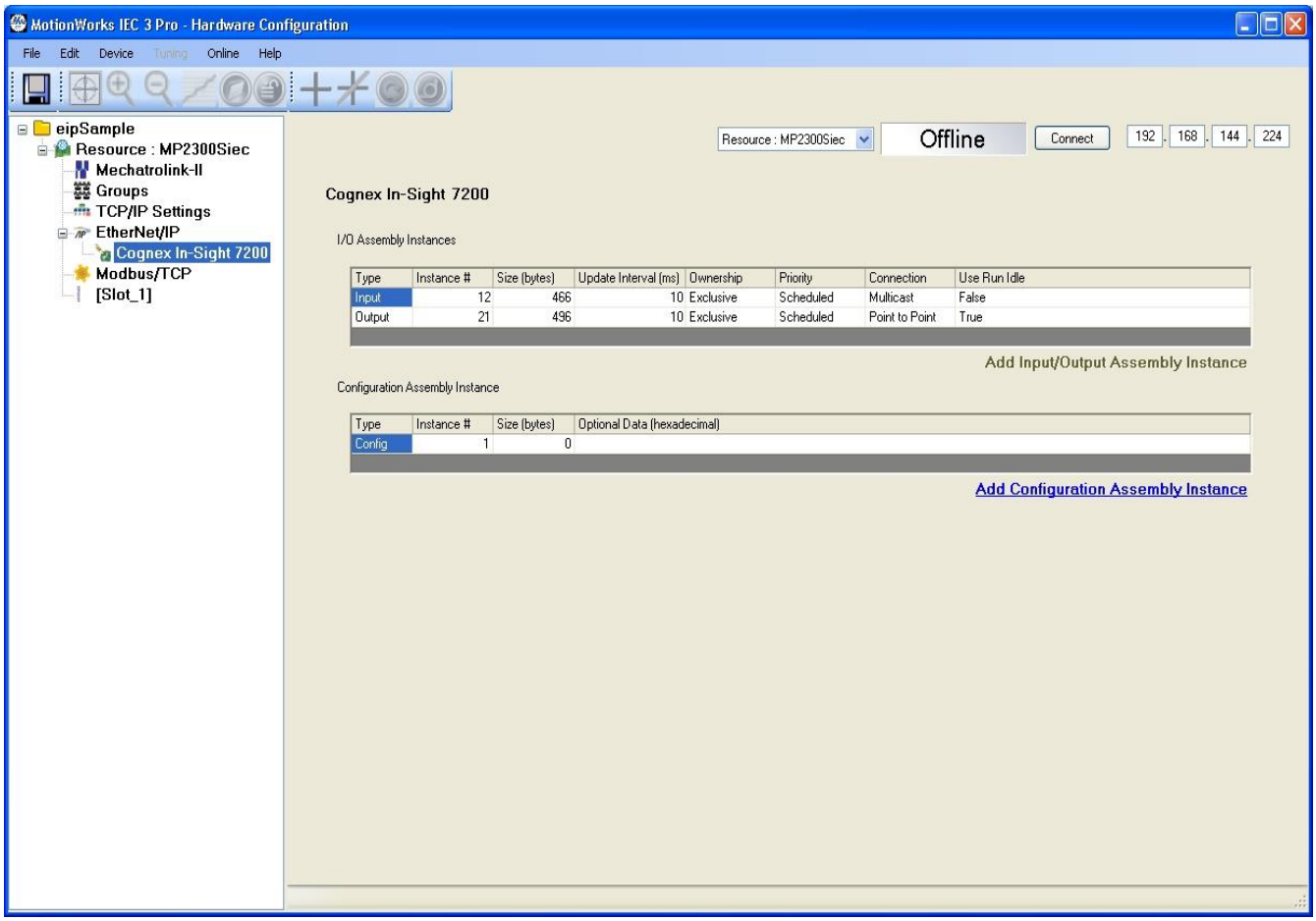

Figure 12: Specify I/O and Configuration Assembly Instances for Cognex System in MotionWorks IEC Hardware Configuration Tool.

5.2 MPiec Controller Application Logic

Once the configuration from section 5.1 is saved to an IEC project, the global variable table of the project is populated with two sections into which the IEC application programmer can define Ethernet/IP variables.

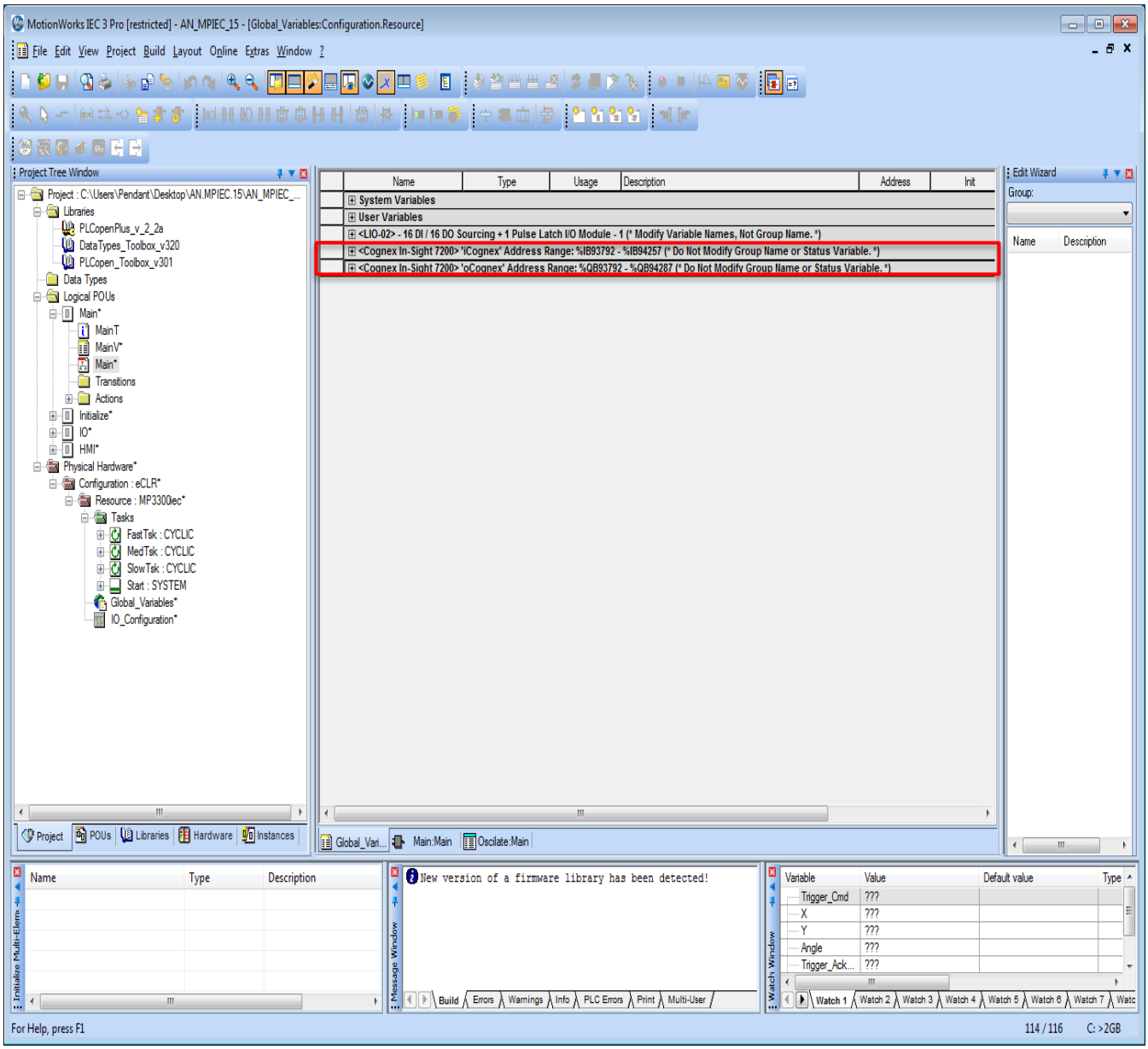

Figure 13: Input and Output Ethernet I/P Groups in Global Variable Table (See AN_MPIEC_15.zwt in AN.MPIEC.15.zip).

Figure 14 below shows the Ethernet/IP variables defined in the global variable table of the IEC application that accompanies the Cognex application (see Figure 6 in Section 4).

Name	Type	Usage	Description	Address	Init	Retain	PDD	OPC	TB	Hid..	Init...	Default Hid...
⊞ System Variables												
⊟ User Variables												
Trigger_Cmd	BOOL	VAR_GLOBAL										
AXIS65	AXIS_REF	VAR_GLOBAL	External Encoder - 65 (* Do Not Modify!! *)		(AxisNum := 65)							
⊞ <LIO-02> - 16 DI / 16 DO Sourcing + 1 Pulse Latch I/O Module - 1 (* Modify Variable Names, Not Group Name. *)												
⊟ <Cognex In-Sight 7200> 'iCognex' Address Range: %IB93792 - %IB94257 (* Do Not Modify Group Name or Status Variable. *)												
Vision_Status	BYTE	VAR_GLOBAL		%IB93792								
Number_Found	REAL	VAR_GLOBAL		%ID93804								
IsFront_0	REAL	VAR_GLOBAL		%ID93808								
IsFront_1	REAL	VAR_GLOBAL		%ID93812								
IsFront_2	REAL	VAR_GLOBAL		%ID93816								
IsFront_3	REAL	VAR_GLOBAL		%ID93820								
IsFront_4	REAL	VAR_GLOBAL		%ID93824								
IsFront_5	REAL	VAR_GLOBAL		%ID93828								
IsFront_6	REAL	VAR_GLOBAL		%ID93832								
IsFront_7	REAL	VAR_GLOBAL		%ID93836								
IsFront_8	REAL	VAR_GLOBAL		%ID93840								
iX_0	REAL	VAR_GLOBAL		%ID93844								
iX_1	REAL	VAR_GLOBAL		%ID93848								
iX_2	REAL	VAR_GLOBAL		%ID93852								
iX_3	REAL	VAR_GLOBAL		%ID93856								
iX_4	REAL	VAR_GLOBAL		%ID93860								
iX_5	REAL	VAR_GLOBAL		%ID93864								
iX_6	REAL	VAR_GLOBAL		%ID93868								
iX_7	REAL	VAR_GLOBAL		%ID93872								
iX_8	REAL	VAR_GLOBAL		%ID93876								
iY_0	REAL	VAR_GLOBAL		%ID93880								
iY_1	REAL	VAR_GLOBAL		%ID93884								
iY_2	REAL	VAR_GLOBAL		%ID93888								
iY_3	REAL	VAR_GLOBAL		%ID93892								
iY_4	REAL	VAR_GLOBAL		%ID93896								
iY_5	REAL	VAR_GLOBAL		%ID93900								
iY_6	REAL	VAR_GLOBAL		%ID93904								
iY_7	REAL	VAR_GLOBAL		%ID93908								
iY_8	REAL	VAR_GLOBAL		%ID93912								
iA_0	REAL	VAR_GLOBAL		%ID93916								
iA_1	REAL	VAR_GLOBAL		%ID93920								
iA_2	REAL	VAR_GLOBAL		%ID93924								
iA_3	REAL	VAR_GLOBAL		%ID93928								
iA_4	REAL	VAR_GLOBAL		%ID93932								
iA_5	REAL	VAR_GLOBAL		%ID93936								
iA_6	REAL	VAR_GLOBAL		%ID93940								
iA_7	REAL	VAR_GLOBAL		%ID93944								
iA_8	REAL	VAR_GLOBAL		%ID93948								
Cognex_Status	WORD	VAR_GLOBAL	(* Do Not Modify. *) EtherNet/IP Adapter Status Variable	%IW94258								
⊟ <Cognex In-Sight 7200> 'oCognex' Address Range: %QB93792 - %QB94287 (* Do Not Modify Group Name or Status Variable. *)												
Trigger_Enable	BOOL	VAR_GLOBAL		%QX93792.0	TRUE							
Trigger_EIP	BOOL	VAR_GLOBAL		%QX93792.1								

Figure 14: Ethernet/IP variables in Global Variable Table.

The addresses of the EIP variables, in the IEC application, are defined to line-up with the variables in the EIP buffer of the Cognex system.

Figure 15 and Figure 16 below have been reproduced from the help documentation included with Congex's In-Sight Explorer v4.8.4. Figure 15 and Figure 16 show the layout of the input and output areas of the Ethernet/IP memory of the Cognex system.

a) EIP Input Area

Instance 12: In-Sight Add-On Profile (AOP) Revision 10 included within In-Sight Software 4.8.0 and later

Instance	Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
12	0	Online	Offline Reason			Missed Acq	Acquiring	Trigger Ack	Trigger Ready
	1	Reserved	Job Load Failed	Job Load Completed	Job Loading	Results Valid	Results Buffer Overrun	Inspection Completed	Inspecting
	2	Reserved							
	3	Reserved			Job Pass	Exposure Complete	Reserved		
	4	Current Job ID (16-bit integer)							
	5								
	6	Acquisition ID (16-bit integer)							
	7								
	8	Inspection ID (16-bit integer)							
	9								
	10	Inspection Result (16-bit integer)							
	11								
	12	Inspection Results 0							
	...								
	499	Inspection Results 487							

Figure 15: Input Area of Cognex System's Ethernet/IP Memory.

- The 1st byte contains some information about the Cognex system; this is captured by the "Vision_Status" variable in the IEC application.
- The user application data in the Cognex system begins at byte #12; the "Number_Found" variable in the IEC application is defined to start at byte #12.
- The "Cognex_Status" variable in the IEC application was added by the MWIEC configuration tool; it contains status information about the Ethernet/IP adapter.

b) EIP Output Area

I/O Assembly Data Attribute Format - Output Assemblies

Instance	Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
21	0	Set Offline	Reserved	Reserved	Initiate Job Load	Inspection Results Ack	Buffer Results Enable	Trigger	Trigger Enable
	1	Soft Event 7	Soft Event 6	Soft Event 5	Soft Event 4	Soft Event 3	Soft Event 2	Soft Event 1	Soft Event 0
	2	Job Load ID							
	3								
	4	User Data 0							
	...								
	495	User Data 491							

Figure 16: Output Area of Cognex System's Ethernet/IP Memory.

- The 1st bit in the 1st byte is set to enable triggering via the Trigger bit (i.e. bit #1); clear this field to disable the Ethernet/IP triggering mechanism.
- The 2nd bit in the 1st byte can be used to trigger the camera to take an image.

Figure 17 and Figure 18 below show the simple IEC application logic that continuously triggers the Cognex camera. In the structured text for the "Oscilate" action, line 25 is used to trigger the Cognex camera via an Ethernet/IP signal. Also, note that line 22 can be used to trigger the camera via an I/O signal. To trigger via an I/O signal, uncomment line 22 and comment out line 25.

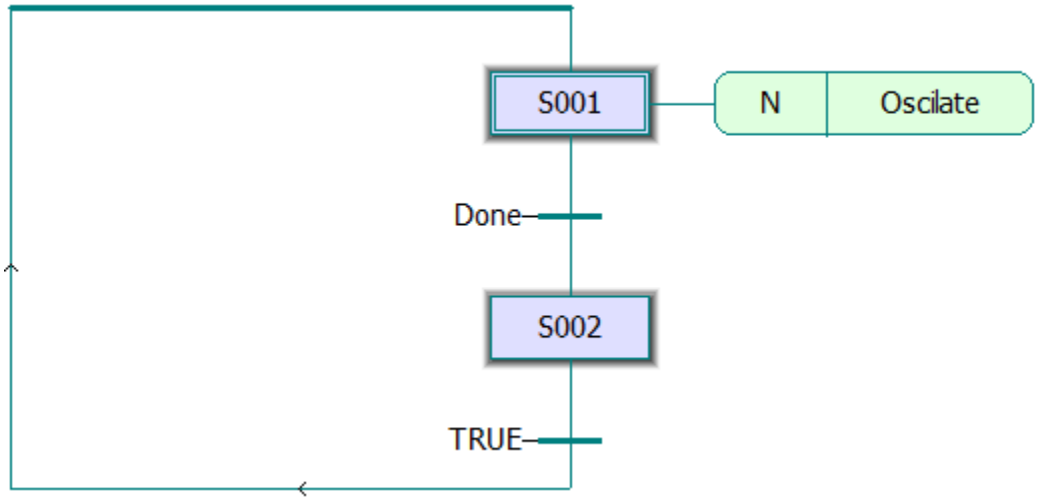

Figure 17: MPiec SFC

```
1    R_TRIG_1(CLK:=S001.X);
2    IF(R_TRIG_1.Q) THEN
3      Trigger_Cmd := TRUE;
4    END_IF;
5
6    TON_1(IN:=S001.X,PT:=T#50ms);
7    IF(TON_1.Q) THEN
8      Trigger_Cmd := FALSE;
9    END_IF;
10
11   TON_2(IN:=S001.X,PT:=T#500ms);
12   IF(TON_2.Q) THEN
13     Done := TRUE;
14   END_IF;
15
16   F_TRIG_1(CLK:=S001.X);
17   IF(F_TRIG_1.Q) THEN
18     Done := FALSE;
19   END_IF;
20
21   (* I/O Trigger *)
22   (* MO1_DO_00 := Trigger_Cmd; *)
23
24   (* EIP Trigger *)
25   Trigger_EIP := Trigger_Cmd;
```

Figure 18: Structured Text for "Oscilate" Action in Figure 17.

6 Other Information

6.1 The Job Display Toolbar

The "Job Display" toolbar is highlighted by item 4 in Figure 4. This toolbar contains a button (Figure 4, item 5) to manually trigger the vision sensor which in turn executes one scan of the vision program.

The toolbar also contains a button (Figure 4, item 6) to toggle the Cognex vision system between online and offline mode. Note that to transfer Ethernet/IP data to a remote system, the Cognex system needs to be in online mode.

6.2 The Status Bar

The status bar is highlighted by item 7 in Figure 4.

The Status bar contains an item (Figure 4, item 8) that indicates how much time it took to execute one scan of the vision program each time the camera is triggered. Also, in the status bar, there is an indicator (Figure 4, item 9) that shows if the Cognex system is online or offline.