VS-06 Smart Camera

VS-06 User´s Manual

Disclaimer

The information and specifications described in this manual are subject to change without notice.

Latest Manual Version

For the latest version of this manual, see the Download Center on our web site at:
www.di-soric.com.

Technical Support

For technical support, e-mail: **service@di-soric.com**.

Warranty and Terms of Sale

For Standard Warranty information contact:

di-soric GmbH & Co. KG
Steinbeisstraße 6
DE-73660 Urbach
Fon: +49 (0) 71 81 / 98 79 - 0
Fax: +49 (0) 71 81 / 98 79 - 179
info@di-soric.com
www.di-soric.com

Contents

PREFACE Welcome!

Purpose of This Manual

This manual contains detailed information about the VS-06 Smart Camera.

Manual Conventions

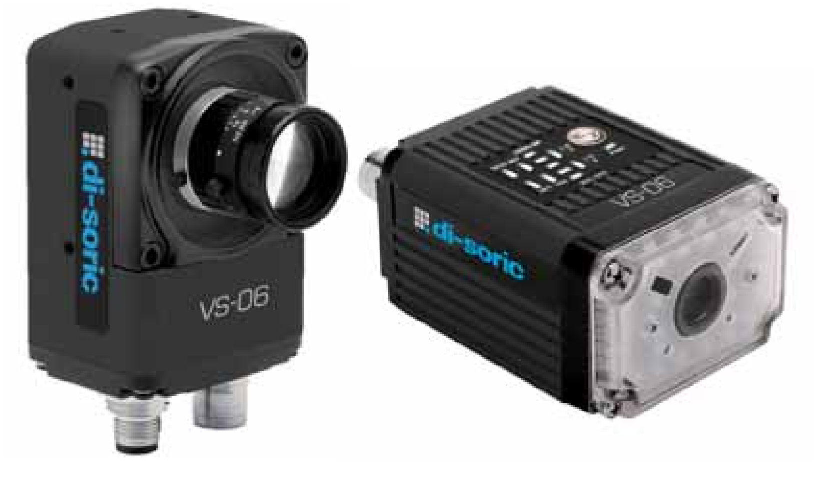
The following typographical conventions are used throughout this manual.

- Items emphasizing important information are **bolded**.
- Menu selections, menu items and entries in screen images are indicated as: Run (triggered), Modify..., etc.

CHAPTER 1

Introduction

FIGURE 1–1. VS-06 Smart Camera, C-Mount and Standard Models



Product Summary

The VS-06 Smart Camera is a compact industrial smart camera that provides powerful machine vision capabilities with a small form factor and intuitive software interface. The VS-06 is designed for industrial environments where IP65/67 enclosure and rugged M12 connectivity are required.

Fully-integrated I/O and communications make the VS-06 easy to incorporate in virtually any machine vision application. Patented liquid lens autofocus and modular optical zoom enables the VS-06 to inspect objects at distances from 33 mm to 2 m and beyond.

Pressing the AutoVision button at the back of the VS-06 enables real time dynamic autofocus. When an object is centered in the field of view and the AutoVision button is pressed, the camera automatically adjusts focal distance and sets internal parameters to optimize image captures.

AutoVision software, designed for use with the VS-06, provides an intuitive interface, step-by-step configuration, and a library of presets that allow easy setup and deployment. For more complex vision applications, the system can be upgraded from AutoVision to Visionscape.

Features and Benefits

- Standard and C-Mount models available
- SXGA (1280 x 960), WVGA (752 x 480), and WUXGA (2048 x 1088, C-Mount model only) resolutions available
- World’s first vision system with liquid lens autofocus (standard models)
- Integrated lighting (standard models)
- Integrated Ethernet
- Flexible programming options for custom applications
- AutoVision button for automatic targeting, calibration, and triggering
- Simplified configuration with AutoVision software
- Fully scalable with Visionscape
- Applications can be ported to Visionscape PC-based machine vision

Applications

- Automotive assembly verification
- Part identification
- Label positioning
- Contents verification
- Electronics assembly verification and identification
- Semiconductor packaging and component inspection
- Auto ID (Data Matrix and other 2D symbologies, 1D, OCR)

Package Contents

Before you install VS AutoVision software and connect your VS-06 Smart Camera, please take a moment to confirm that the following items are available:

- VS-06 Smart Camera — Your package contains one of the available models listed in Table 1–1
- di-soric Tools Drive — USB flash drive containing AutoVision software
- Required accessories such as a power supply or power cable

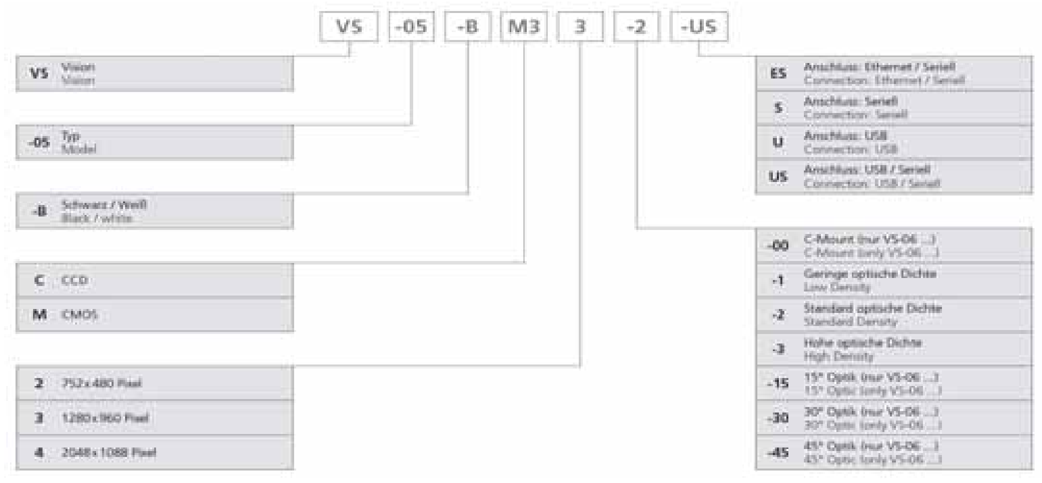
VS-06 Smart Camera Models

Table 1–1 lists and describes the VS-06 Smart Camera models.

TABLE 1–1. VS-06 Smart Camera Models

Part Number	VS-06 Smart Camera Model
VS-06-BC3-00-ES	VS-06, SXGA, AutoVision, C-Mount
VS-06E-BC3-00-ES	VS-06, SXGA, AutoVision + Visionscape, C-Mount
VS-06-BM2-00-ES	VS-06, WVGA, AutoVision, C-Mount
VS-06E-BM2-00-ES	VS-06, WVGA, AutoVision + Visionscape, C-Mount
VS-06-BM2-15-ES	VS-06, WVGA, Built-In Light, AutoVision, 15° Lens
VS-06-BM2-30-ES	VS-06, WVGA, Built-In Light, AutoVision, 30° Lens
VS-06-BM2-45-ES	VS-06, WVGA, Built-In Light, AutoVision, 45° Lens
VS-06E-BM2-15-ES	VS-06, WVGA, Built-In Light, AutoVision + Visionscape, 15° Lens
VS-06E-BM2-30-ES	VS-06, WVGA, Built-In Light, AutoVision + Visionscape, 30° Lens
VS-06E-BM2-45-ES	VS-06, WVGA, Built-In Light, AutoVision + Visionscape, 45° Lens
VS-06-BC3-15-ES	VS-06, SXGA, Built-In Light, AutoVision, 15° Lens
VS-06-BC3-30-ES	VS-06, SXGA, Built-In Light, AutoVision, 30° Lens
VS-06-BC3-45-ES	VS-06, SXGA, Built-In Light, AutoVision, 45° Lens
VS-06E-BC3-15-ES	VS-06, SXGA, Built-In Light, AutoVision + Visionscape, 15° Lens
VS-06E-BC3-30-ES	VS-06, SXGA, Built-In Light, AutoVision + Visionscape, 30° Lens
VS-06E-BC3-45-ES	VS-06, SXGA, Built-In Light, AutoVision + Visionscape, 45° Lens
VS-06-BM4-00-ES	VS-06, WVXGA, AutoVision, C-Mount
VS-06E-BM4-00-ES	VS-06, WVXGA, AutoVision + Visionscape, C-Mount

Part Number Structure


VS
-05
-B
M3
3
-2
-US
VS Vision Vision
-05 Typ Model
-B Schwarz / Weiß Black / white
C CCD
M CMOS
2 752x480 Pixel
3 1280x960 Pixel
4 2048x1088 Pixel
ES Anschluss: Ethernet / Seriell Connection: Ethernet / Seriell
S Anschluss: Seriell Connection: Seriell
U Anschluss: USB Connection: USB
US Anschluss: USB / Seriell Connection: USB / Seriell
-00 C-Mount (nur VS-06 ...) C-Mount (only VS-06 ...)
-1 Geringe optische Dichte Low Density
-2 Standard optische Dichte Standard Density
-3 Hohe optische Dichte High Density
-15 15° Optik (nur VS-06 ...) 15° Optic (only VS-06 ...)
-30 30° Optik (nur VS-06 ...) 30° Optic (only VS-06 ...)
-45 45° Optik (nur VS-06 ...) 45° Optic (only VS-06 ...)

CHAPTER 2

System Components

This section contains information about system components as well as information to help you connect the VS-06 Smart Camera. Specific information describes connectors, adapters, cables, pinouts, and signals.

Note: There are no user-serviceable parts inside.

Hardware Components

Table 2-1 lists VS-06 Smart Camera hardware components.

TABLE 2–1. VS-06 Smart Camera Hardware Components

Part Number	Description
Part Number	**VS-06 Smart Camera Model**
VS-06-BC3-00-ES	VS-06, SXGA, AutoVision, C-Mount
VS-06E-BC3-00-ES	VS-06, SXGA, AutoVision + Visionscape, C-Mount
VS-06-BM2-00-ES	VS-06, WVGA, AutoVision, C-Mount
VS-06E-BM2-00-ES	VS-06, WVGA, AutoVision + Visionscape, C-Mount
VS-06-BM2-15-ES	VS-06, WVGA, Built-In Light, AutoVision, 15° Lens
VS-06-BM2-30-ES	VS-06, WVGA, Built-In Light, AutoVision, 30° Lens
VS-06-BM2-45-ES	VS-06, WVGA, Built-In Light, AutoVision, 45° Lens
VS-06E-BM2-15-ES	VS-06, WVGA, Built-In Light, AutoVision + Visionscape, 15° Lens
VS-06E-BM2-30-ES	VS-06, WVGA, Built-In Light, AutoVision + Visionscape, 30° Lens

TABLE 2–1. **VS-06 Smart Camera Hardware Components (Continued)**

Part Number	Description
VS-06E-BM2-45-ES	VS-06, WVGA, Built-In Light, AutoVision + Visionscape, 45° Lens
VS-06-BC3-15-ES	VS-06, SXGA, Built-In Light, AutoVision, 15° Lens
VS-06-BC3-30-ES	VS-06, SXGA, Built-In Light, AutoVision, 30° Lens
VS-06-BC3-45-ES	VS-06, SXGA, Built-In Light, AutoVision, 45° Lens
VS-06E-BC3-15-ES	VS-06, SXGA, Built-In Light, AutoVision + Visionscape, 15° Lens
VS-06E-BC3-30-ES	VS-06, SXGA, Built-In Light, AutoVision + Visionscape, 30° Lens
VS-06E-BC3-45-ES	VS-06, SXGA, Built-In Light, AutoVision + Visionscape, 45° Lens
VS-06-BM4-00-ES	VS-06, WVXGA, AutoVision, C-Mount
VS-06E-BM4-00-ES	VS-06, WVXGA, AutoVision + Visionscape, C-Mount
Power Supplies	
VSID-PS-24V-ES	Power Supply, M12 12-pin Socket, 1.3 m
Communication Devices and Cables	
VSID-IB-ES	Interface Device
VKHM-Z-1/12/DB9-K	Cordset, Host, Serial M12 12 pin Socket (Screw-on) to DB9 Socket, 1M
Accessories	
VSID-BW-004	L-Bracket Kit
VSID-R90-002	Right Angle Mirror Kit
VSID-W-K-000	Window Replacement Kit
VSID-L-S-15	15° Lens Kit
VSID-L-S-30	30° Lens Kit
VSID-L-S-45	45° Lens Kit
VSID-W-G-850	Glass WIndow Kit with Infrared (IR) Filter
VSID-W-G-000	Glass Window Kit
VSID-LP-C-48	Lens Protection Housing, Standard Length (up to 48mm)
VSID-LP-C-72	Lens Protection Housing, Long (up to 72mm)
VSID-06-BE-3	Blue light LED-board for VS-06 (exect C-Mount)
VSID-06-BE-5	White light LED-board for VS-06 (exect C-Mount)
VS-UP-AV/VS	Upgrade AutoVision to Visionscape
VS-UP-AV/OCV	Upgrade AutoVision to Verification / OCV
VS-UP-AV/VS-OCV	Upgrade AutoVision VisionScape + Verification / OCV

Note: Additional hardware components and cables are available in the di-soric Product portfolio.

Power Supplies	
VSID-PS-24V-ES	Power Supply, M12 12-pin Socket, 1.3 m

TABLE 2–1. VS-06 Smart Camera Hardware Components (Continued)

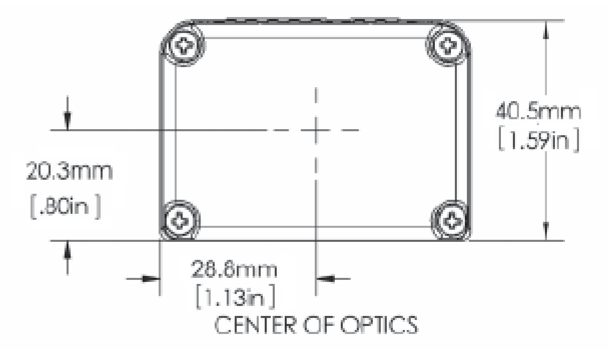
Part Number	Description
Communication Devices and Cables	
VKHM-Z-5/12-A	Power I/O cable, 12 Pin, shielded, 5m
VKHM-Z-5/RJ45	Ethernet cable, flex-chain, 5m
VKSM-Z-5/12-A	Power I/O cable, 12 Pin, shielded, flex-chain, 5m

Note: Additional hardware components are available in the di-soric Product Pricing Catalog.

Standard Front

Figure 2-1 shows the front of the VS-06 Smart Camera.

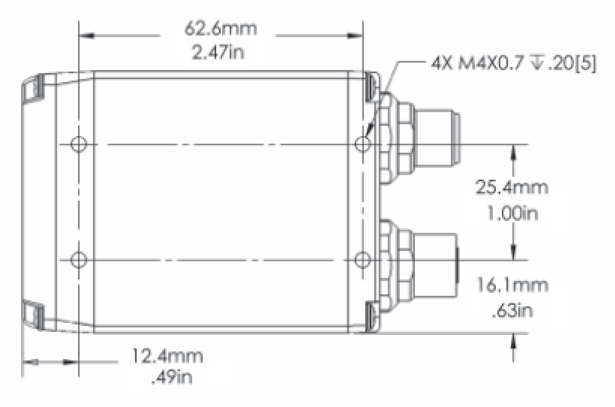
FIGURE 2–1. Front



Standard Base

Figure 2–2 shows the base of the VS-06 Smart Camera.

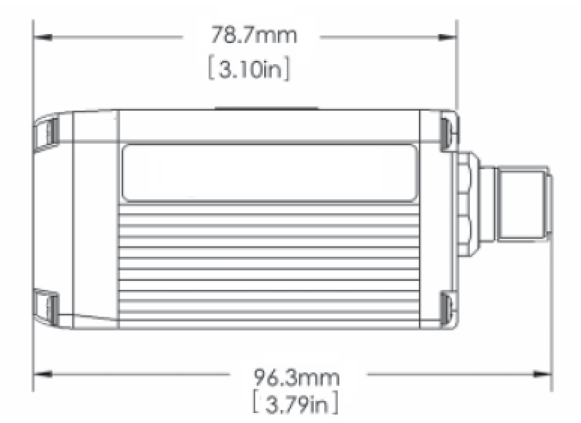
FIGURE 2–2. Base



Standard Side

Figure 2-3 shows the side of the VS-06 Smart Camera.

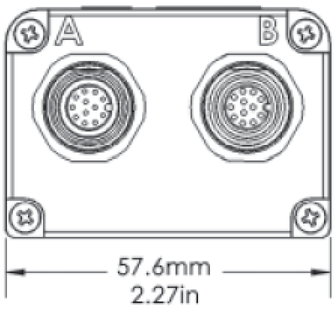
FIGURE 2–3. Side



Standard Back

Figure 2-4 shows the back of the Smart Camera.

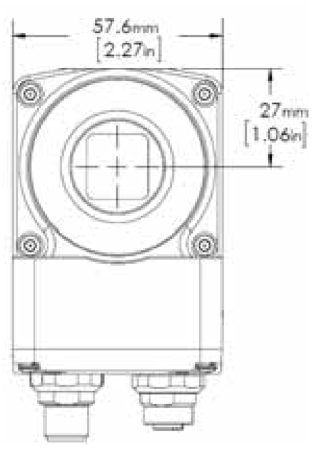
FIGURE 2–4. Back



C-Mount Front

Figure 2-5 shows the front of the C-Mount Smart Camera.

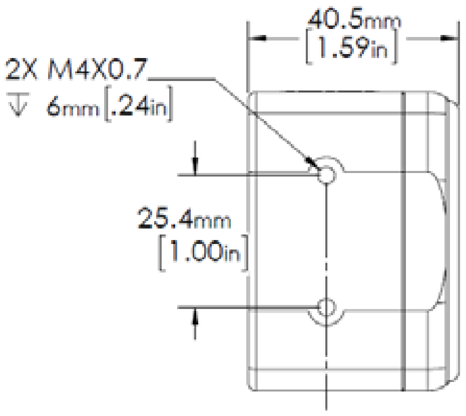
FIGURE 2–5. Front



C-Mount Base

Figure 2–6 shows the top of the C-Mount Smart Camera.

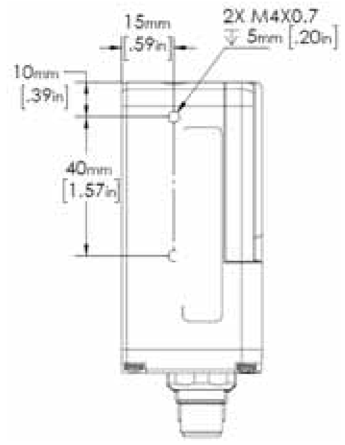
FIGURE 2–6. Top



C-Mount Side

Figure 2-7 shows the side of the C-Mount Smart Camera.

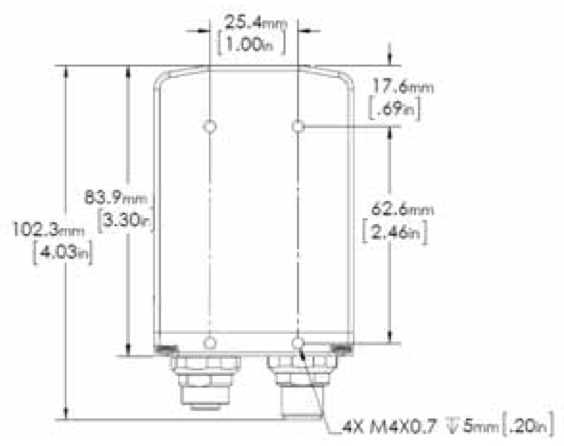
FIGURE 2–7. Side



C-Mount Back

Figure 2-8 shows the back of the C-Mount Smart Camera.

FIGURE 2–8. Back



Important Label Information

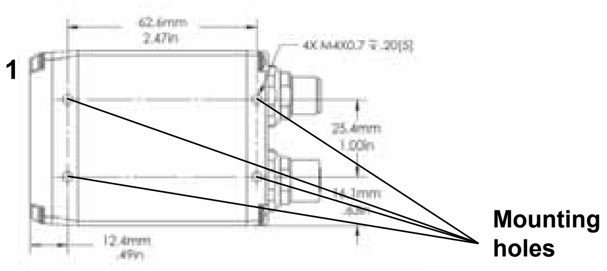
Each VS-06 Smart Camera has its own label, which contains important information about that camera.

- P/N – The di-soric part number of your VS-06 Smart Camera.
- S/N — The serial number of your VS-06 Smart Camera.
- MAC — The MAC address of your VS-06 Smart Camera.

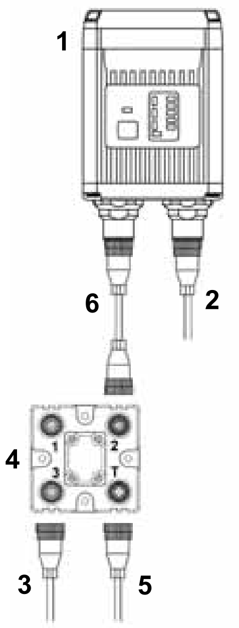
Mounting and Wiring the VS-06 Smart Camera

Important: Pins 9 (Host RxD) and 10 (Host TxD) must be tied to Ground (Pin 7) when using a flying lead cable and the serial port is not being used. The camera may not boot to completion if RxD and TxD are not grounded.

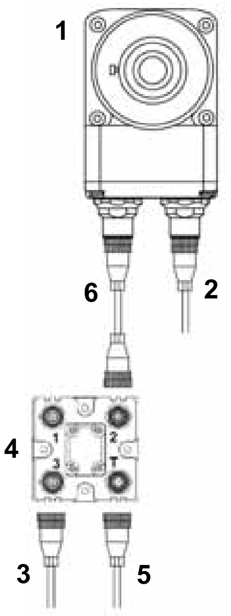
- Mount the camera (**1**) securely as required by the application.



- Connect the Ethernet cable (**2**) from “B” on the camera (**1**) to the network.
- Connect the power supply cable (**3**) to “3” on the VSID-IB-ES (**4**).
- Connect the trigger (**5**) to “T” on the VSID-IB-ES (**4**).
- Connect the “Common” cable (**6**) from “A” on the camera (**1**) to “2” on the VSID-IB-ES (**4**).
- Plug in the power supply (**3**).



Standard VS-06

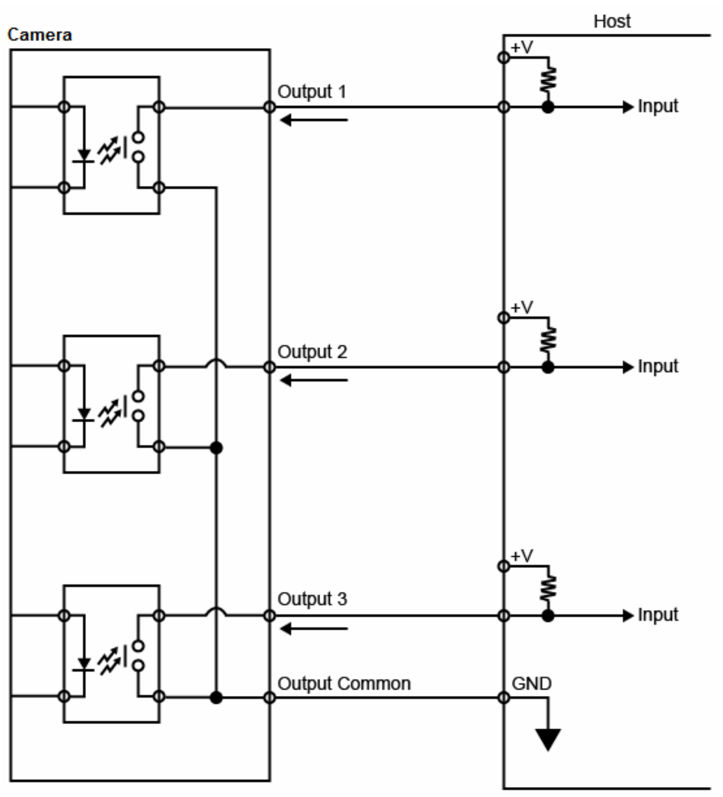


VS-06 C-Mount

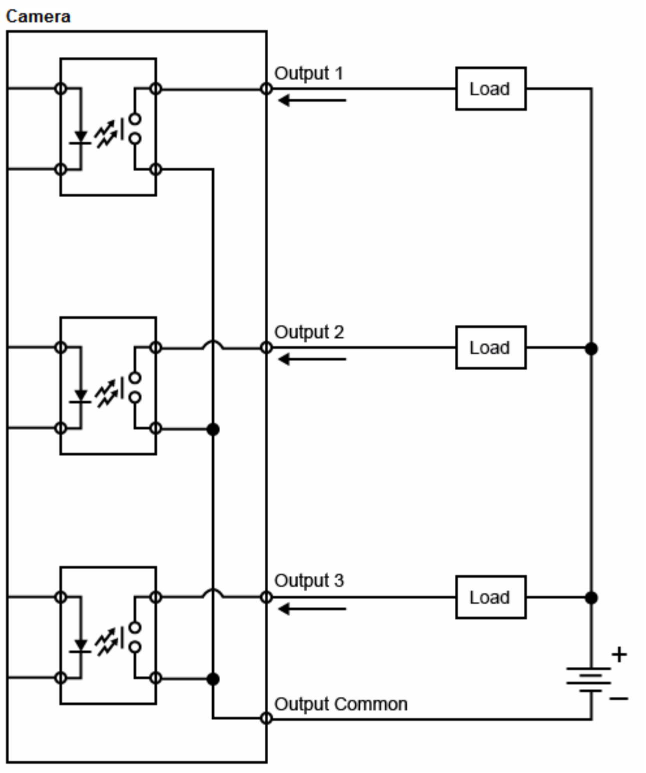
Optoisolated Outputs

The reader has optoisolated outputs that can transfer signals from the camera to peripherals. Outputs can be configured as either NPN or PNP, but NPN and PNP cannot be mixed in a system, because the output common is shared by all outputs.

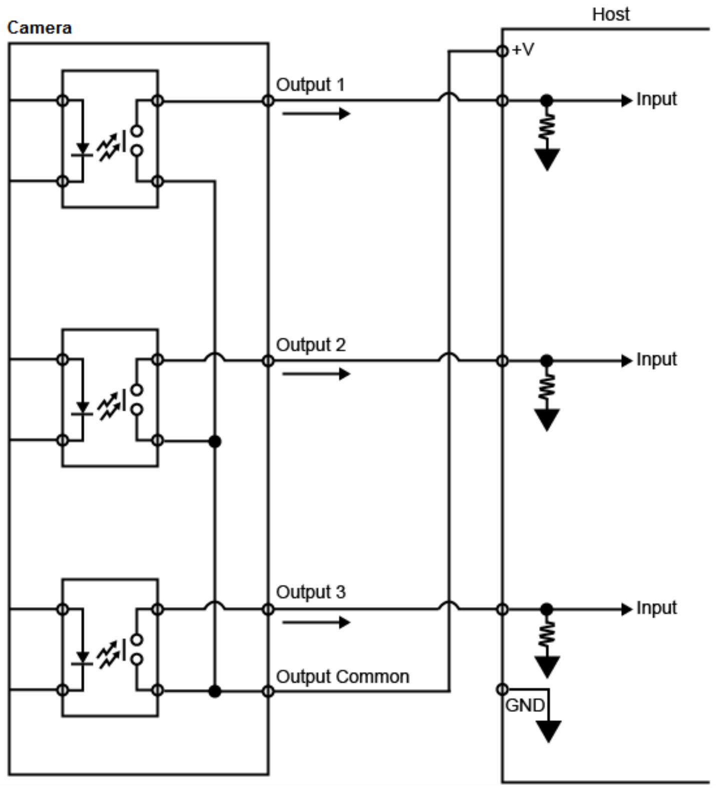
NPN Output for Host Input



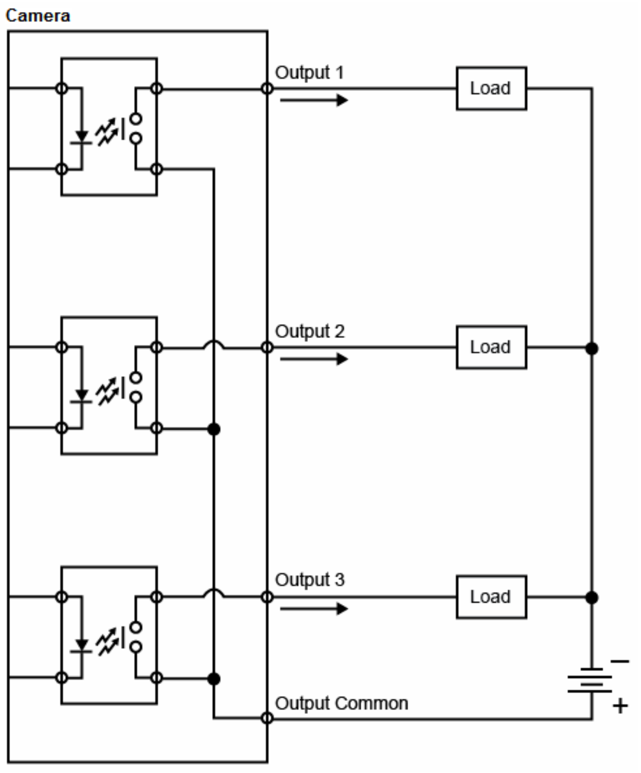
NPN Output for External Load



PNP Output for Host Input



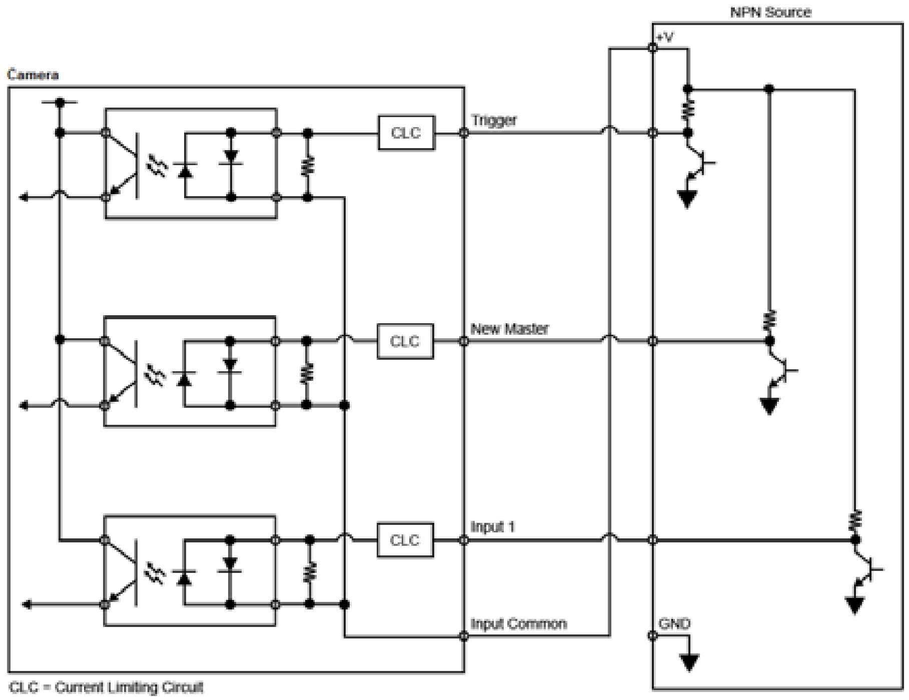
PNP Output for External Load



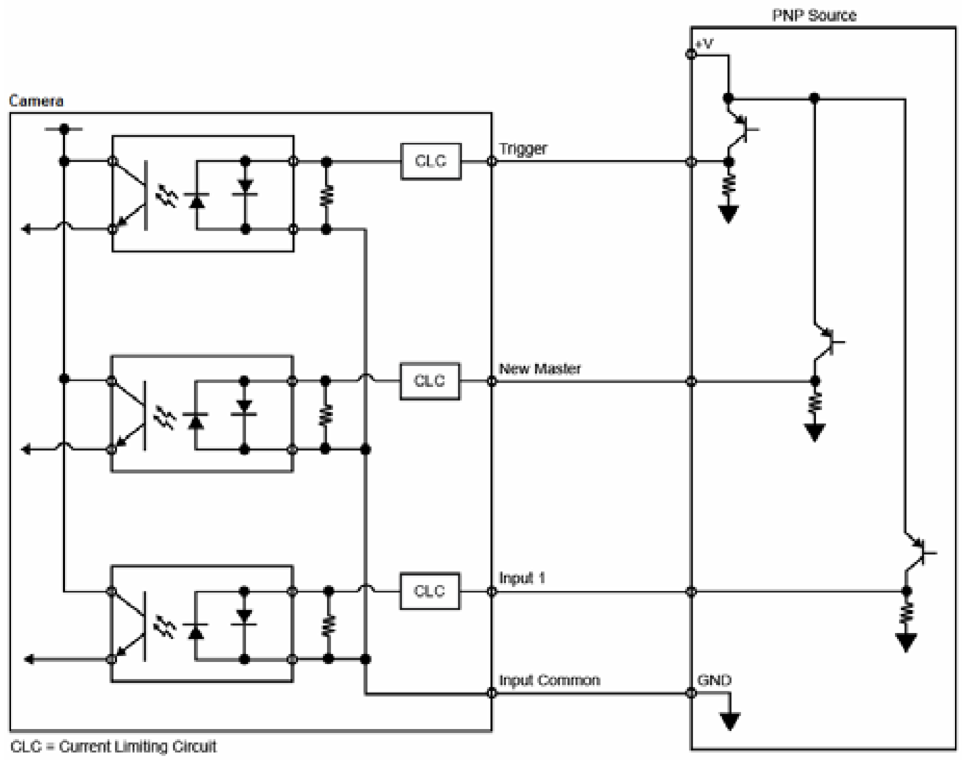
Optoisolated Inputs

All discrete inputs are optoisolated. Inputs can be configured as either NPN or PNP, but NPN and PNP cannot be mixed in a system, because the input common is shared by all inputs.

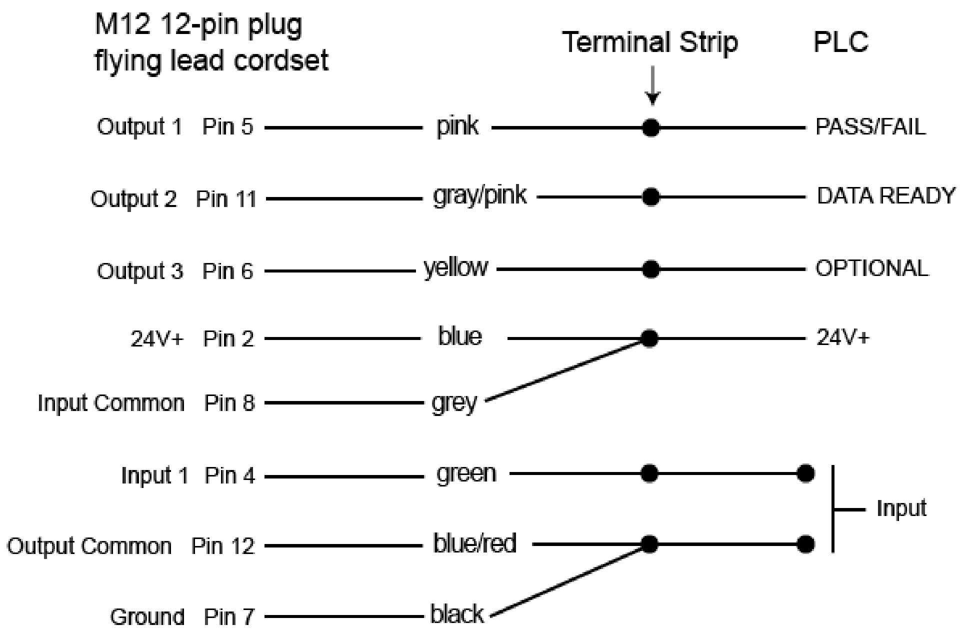
NPN



PNP

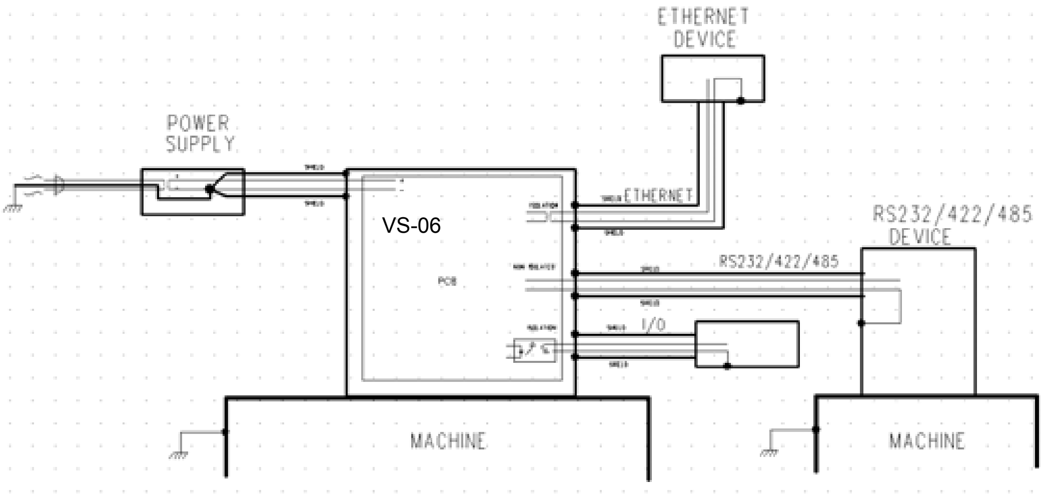

PNP Source
+V
Camera
CLC
Trigger
CLC
New Master
CLC
Input 1
Input Common
GND
CLC = Current Limiting Circuit

Input/Output Wiring



Ground and Shield Considerations

Proper grounding is necessary for operator safety, noise reduction, and the protection of equipment from voltage transients. Buildings, including any steelwork, all circuits, and all junction boxes must be grounded directly to an earth ground in compliance with local and national electrical codes.

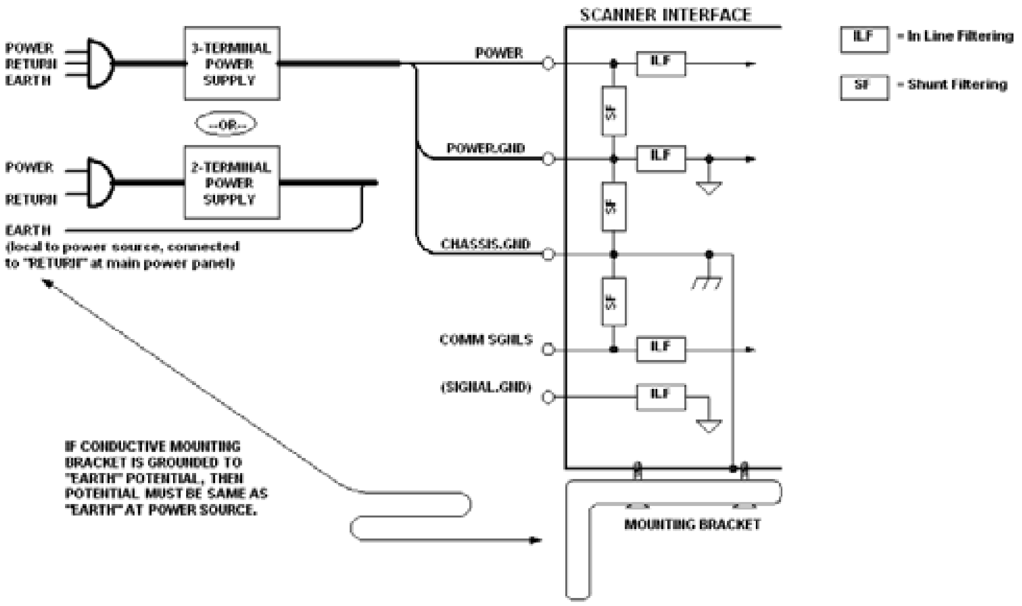


An earth ground is provided through the cable shields and chassis of the imager.

Ground Loops

Ground loops (signal degradation due to different ground potentials in communicating devices) can be eliminated or minimized by ensuring that both the host, imager, and their power supplies are connected to a common earth ground.

Expected Power and Ground Connections for Proper Operation



Grounding Notes:

- Ensure that mounting bracket “Earth” is at the same potential as power source “Earth”.
- Supply “Return” and “Earth” ground must be stable, low-impedance reference points.
- “2-Terminal Power Supply” must still provide an “Earth” connection to the imager.
- “Signal Ground” can be used for communications and/or discrete signal ground reference. It must **not** be used as Power Ground or Earth Ground.

Power Requirements

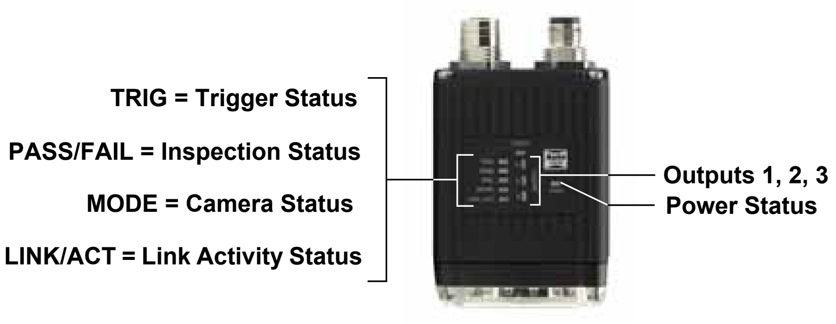
Refer to Table 2-3 when determining the power supply requirements for your camera.

TABLE 2–3. Camera Power Requirements

Component	
VS-06 Smart Camera, CCD, SXGA	5-28VDC, 200mV p-p max ripple, 170mA at 24VDC (typ.) 15.5 watts (max.)
VS-06 Smart Camera, CMOS, WVGA	5-28VDC, 200mV p-p max ripple, 135mA at 24VDC (typ.) 13 watts (max.)
VS-06 C-Mount Smart Camera, CCD, SXGA	5-28VDC, 200mV p-p max ripple, 170mA at 24VDC (typ.) 7 watts (max.)
VS-06 C-Mount Smart Camera, CMOS, WVGA	5-28VDC, 200mV p-p max ripple, 135mA at 24VDC (typ.) 4 watts (max.)
VS-06 C-Mount Smart Camera, CMOS, WUXGA	5-28VDC, 200mV p-p max ripple, 140mA at 24VDC (typ.) 5.7 watts (max.)

Status Indicators

The top of the VS-06 Smart Camera has multiple LEDs that indicate different trigger, inspection, camera, communication, and power states.

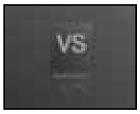


TRIG	**On Steady**	Continuous Trigger
	Off	Waiting for Trigger Event
	On Flashing	Trigger Event
PASS/FAIL	**On**	Active State
	Off	Inactive State
MODE	**On Steady**	Unit Ready
	Off	Unit Not Ready
LINK/ACT	**On Steady**	Link Established
	Off	No Link/Activity
	On Flashing	Link Established and Activity on Link
PWR	**On**	Power On
	Off	No Power Applied to Unit
OUTPUTS	**On**	Signal Sent to External Output
	Off	No Signal Sent to External Output

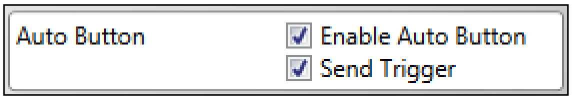
Additional User Feedback

- Green Flash – A green flash from the front of the unit indicates a Good Read.
- Red X Targeting Pattern – The red X targeting pattern from the front of the unit allows the user to center an object in the camera’s field of view.
- Beeper – The beeper is an audible verification that either a Pass or a Fail has occurred.

AutoVision Button



The AutoVision Button has three positions, selectable by the length of time the button is held down, and indicated by one, two, or three beeps and LED flashes in succession. It can also be used to send a trigger signal when **Send Trigger** is checked in AutoVision software's **Connect** view. When the trigger functionality is enabled, pushing the AutoVision Button triggers the camera to capture an image.



Auto Button ✓ Enable Auto Button ✓ Send Trigger

1st Position: Red Targeting Pattern

The first AutoVision Button position turns the targeting system on. This overrides any other targeting modes that have been configured.

2nd Position: Auto Calibration

The second AutoVision Button position starts the Auto Calibration process, which selects the appropriate photometry and focus settings for the camera. The selected values are then saved for power-on.

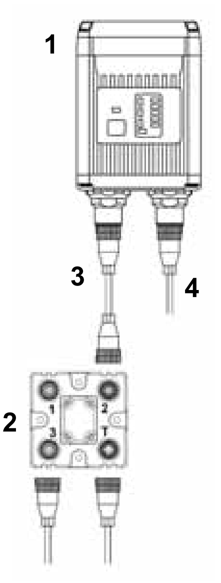
3rd Position: Teach

The third AutoVision Button position sets the Match String to the next OCR string or symbol data that is decoded.

Setting Up a Job in AutoVision

AutoVision is a critical component of the VS-06's functionality. Designed for use with the VS-06, AutoVision provides an intuitive interface, step-by-step configuration, and a library of presets that allow easy setup and deployment. For more complex vision applications, the system can be upgraded from AutoVision to Visionscape.

1. Configure VS-06 hardware.



See Appendix A, **Connector Pinouts**, for VS-06 pin assignments.

Item	***Description***	***Part Number***
1	VS-06 Smart Camera	VS-06-BXX-XX-ES
2	Interface Device	VSID-IB-ES
3	Power Supply, M12 12-pin Socket, 1.3 m	VSID-PS-24V-ES
4	Cordset, Host, Ethernet, M12 8-pin Plug to RJ45, 5m	VKHM-Z-5/RJ45

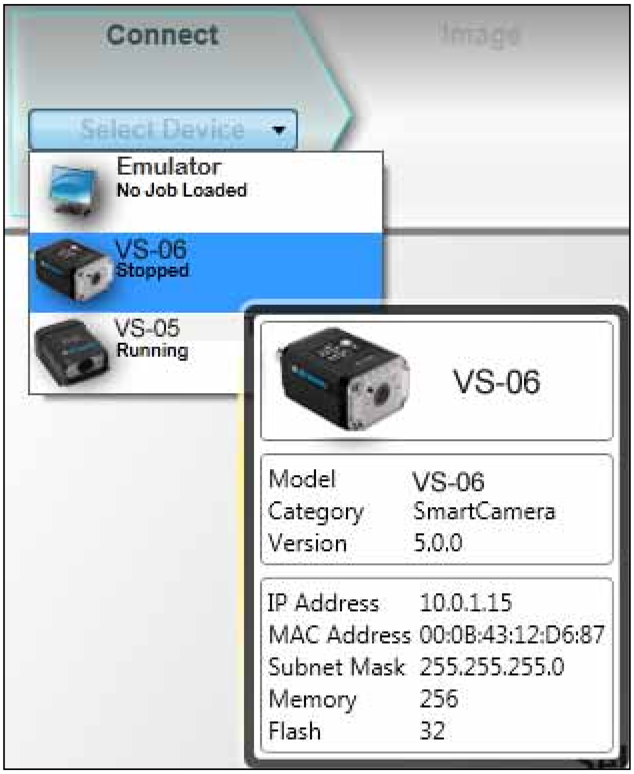
Note: Additional cables available in the di-soric Product Pricing Catalog.

– Mount the camera as required by the application.
– Connect the Ethernet cable from "B" on the camera to the network.
– Connect the power supply to "3" on the VSID-IB-ES.
– Connect the photo sensor to "T" on the VSID-IB-ES.

- Connect the "Common" cable to "2" on the VSID-IB-ES and "A" on the camera.
- Plug in the power supply.

2. Select your VS-06 in the AutoVision Connect view, create a job, and adjust camera settings.

AutoVision's **Connect** view allows you to select your device and configure its settings, and to create a new job. The **Select Device** dropdown menu provides a list of available devices. Hover the mouse over a device to see its details.

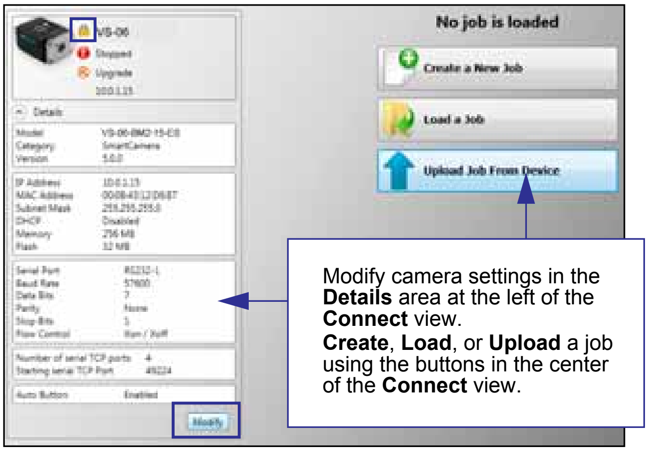


Model VS-06
Category SmartCamera
Version 5.0.0

IP Address 10.0.1.15
MAC Address 00:0B:43:12:D6:87
Subnet Mask 255.255.255.0
Memory 256
Flash 32

Click the lock icon to take control of the camera. When you have control of the camera, the **Modify** button will appear beneath the camera settings. Click the Modify button to adjust camera settings.

Note: The default IP address of the camera is: **192.168.0.10**. Be sure your PC is on the same subnet (**192.168.0.100**, for example).



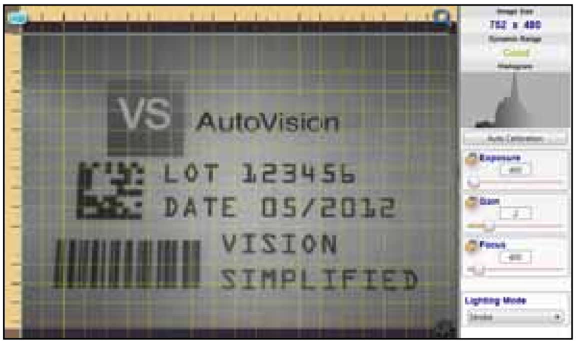
Model	VS-06-BM2-15-C8
Category	SmartCamera
Version	5.0.0
IP Address	10.0.1.15
MAC Address	00:08:43:12:D6:87
Subnet Mask	255.255.255.0
DHCP	Disabled
Memory	256 MB
Flash	32 MB
Serial Port	RS232-1
Baud Rate	57600
Data Bits	7
Parity	None
Stop Bits	1
Flow Control	Xon / Xoff
Number of serial TCP ports	4
Starting serial TCP Port	49224
Auto Button	Enabled

Modify camera settings in the **Details** area at the left of the **Connect** view.

Create, **Load**, or **Upload** a job using the buttons in the center of the **Connect** view.

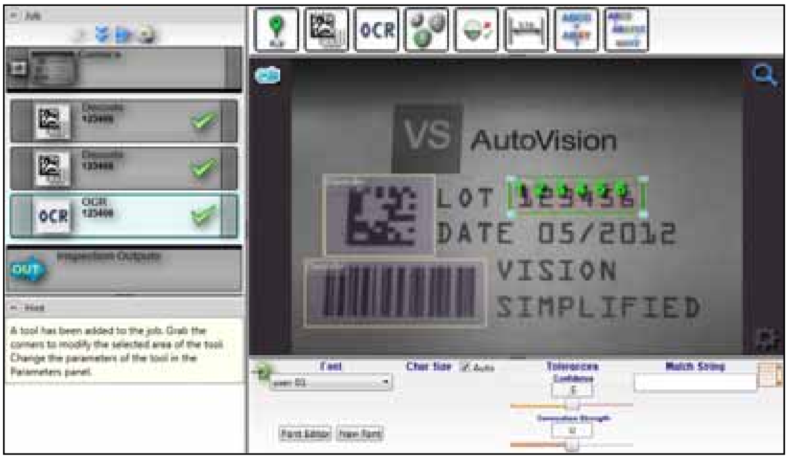
Important: When modifying camera settings, you will need to enter a username and password for the camera if a password has been defined.

Once you have selected your camera, adjusted its settings, and created a new job, you will move to the **Image** view. This view allows you to **Auto Calibrate** the camera, and to manually adjust the camera's Exposure, Gain, and Focus, and also to set the Lighting Mode (On, Off, or Strobe).



3. Edit the Job in VS AutoVision.

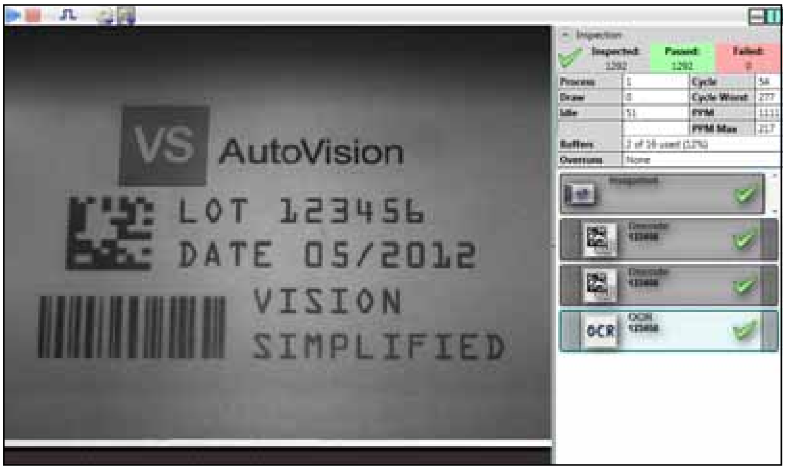
   After you have created a new job, loaded a job from your PC, or uploaded a job from the camera, you will proceed to the Edit view to refine your machine vision job. The Camera parameters below the captured image allow you to set Gain, Exposure, Focus, Trigger, and Lighting. Inspection Outputs options allow you to connect your job to the outside world. This is also the view where you can add multiple tools to the job. The tool icons are located above the main view area.



A tool has been added to the job. Grab the corners to modify the selected area of the tool. Change the parameters of the tool in the Parameters panel.

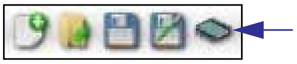
4. Run the Job in AutoVision.

   Going to the **Run** view will automatically download your job to the camera and start running.



5. Save the Job.

   Click the **Save to Camera** icon on the File menu bar to save the job to the VS-06.



Trigger Debounce

Trigger Debounce is the ability of the system to accomodate switching noise on a trigger state change – a common issue with relays that have some intermittent contact while engaging.

Trigger overruns (when the vision system is triggered faster than the device can process) can be avoided by increasing the “debounce” time in the camera definition file located in the C:\di-soric\Vscape\Drivers\CamDefs directory.

The IO Line Debounce High Time and IO Line Debounce Low Time can be added to the file as in the example below. The default debounce time is 1 ms (1,000 µs).

Note: Although the value entered for the "IO Line Debounce Time" is in microseconds, it will only be rounded up to a millisecond value. For example, entering the value **1001** will resolve to 2 ms; entering a value of **2800** will resolve to 3 ms.

The min value for "IO Line Debounce Time" is 0, which disables software debounce altogether. The maximum value is 100000 (100 ms).

Camera Definition File Example

```
IO Line Debounce High Time 2000           //usecs
IO Line Debounce Low Time  2000         //usecs
```

CHAPTER 3

Optics and Lighting

This section describes the optical and illumination characteristics of the VS-06 Smart Camera.

Optics

The VS-06 Smart Camera is available with a built-in CMOS sensor or CCD sensor.

Optics Specifications

Part Number	VS-06-BM2-15-ES	VS-06-BM2-30-ES	VS-06-BM2-45-ES	VS-06-BC3-15-ES	VS-06-BC3-30-ES	VS-06-BC3-45-ES
Sensor	1/3”, SXGA (1280 x 960) CCD, up to 20 fps			1/3”, WVGA (752 x 480) CMOS, up to 60 fps		
Sensor Color	Monochrome					
Focal Range	1” (33 mm) to ∞ (liquid lens autofocus)					
Shutter	Global Shutter; Exposure: 6μs to 100ms (1/150,000 to 1/10) Default = 666μs (1/1,500)			Global Shutter; Exposure: 25μs to 100ms (1/40,000 to 1/10) Default = 400μs (1/2,500)		
Part Number	**VS-06-BM2-15-ES**	**VS-06-BM2-30-ES**	**VS-06-BM2-45-ES**	**VS-06-BC3-15-ES**	**VS-06-BC3-30-ES**	**VS-06-BC3-45-ES**
Sensor	1/3”, SXGA (1280 x 960) CCD, up to 20 fps			1/3”, WVGA (752 x 480) CMOS, up to 60 fps		
Sensor Color	Monochrome					
Focal Range	1” (33 mm) to ∞ (liquid lens autofocus)					
Shutter	Global Shutter; Exposure: 6μs to 100ms (1/150,000 to 1/10) Default = 666μs (1/1,500)			Global Shutter; Exposure: 25μs to 100ms (1/40,000 to 1/10) Default = 400μs (1/2,500)		
Part Number	**VS-06-BC3-00-ES**	**VS-06E-BC3-00-ES**	**VS-06-BM2-00-ES**	**VS-06E-BM2-00-ES**	**VS-06-BM4-00-ES**	**VS-06E-BM4-00-ES**
Sensor	1/3”, SXGA (1280 x 960) CCD, up to 20 fps		1/3”, WVGA (752 x 480) CMOS, up to 60 fps		2/3”, WUXGA (2048 x 1088) CMOS, up to 48 fps	
Sensor Color	Monochrome					
Focal Range	Depends on lens					
Shutter	Global Shutter; Exposure: 6μs to 100ms (1/150,000 to 1/10) Default = 666μs (1/1,500)		Global Shutter; Exposure: 25μs to 100ms (1/40,000 to 1/10) Default = 400μs (1/2,500)		Global Shutter; Exposure: 25μs to 100ms (1/40,000 to 1/10) Default = 400μs (1/2,500)	

Lens Substitution

The following procedure will change the appropriate settings in the VS-06 to allow the camera to focus properly after the lens has been changed. Please note that the VS-06 camera will use default lookup tables for the focus when the lens selection is changed, so the actual focus distances may not be as accurate as the lens that was shipped with the unit that was factory calibrated. Since default lookup tables are used, the VS-06 may not focus over the full focus range that is normally seen when using the factory calibrated lens.

After the lens has been changed via the parameters below, the new values will take effect the next time that the lens focus is modified.

1. Boot the VS-06 Smart Camera.
2. Connect to the VS-06 via Telnet using the IP address of the camera.
3. Send the following command after the VS-06 has booted:

   stopAll

   The response should be **"value = 1 = 0x1"**.
4. Send the following command:

   GetCurrentLense()

   One of these 3 responses will be seen:

   1 = 15deg

   2 = 30deg

   3 = 45deg
5. After camera has booted, send the following command (choose the appropriate command based on the lens):

   SetCurrentLense(1)(to change to 15 degree lens)

   The response should be:

   "Now Set to 1 = 15deg"

   "value = 0 = 0x0"

SetCurrentLense(2)(to change to 30 degree lens)

The response should be:

"Now Set to 2 = 30deg"

"value = 0 = 0x0"

SetCurrentLense(3)(to change to 45 degree lens)

The response should be:

"Now Set to 3 = 45deg"

"value = 0 = 0x0"

6. Send the following command:

   startAll

   The response should be "value = 1 = 0x1"

Illumination

The standard version of the VS-06 Smart Camera has built-in lighting (red LEDs for SXGA models and white LEDs for QXGA models). The LEDs can be configured to operate in multiple modes – Continuous, Strobe, and Off.

Warning: Running a red LED board on a camera with a white LED color profile will damage both the board and the camera.

Important: The VS-06 C-Mount does not have built-in lighting. The Machine Vision Lighting Principles on the following page provide some suggestions for how to determine the appropriate external lighting for your application.

Lighting Specifications

The VS-06 version with built-in lighting is standardly delivered with a red LED-Lighting board with 617 nm wave length.

Optional additional LED-boards as follows:

VSID-06-BE-3	LED board for VS-06 with integrated lighting, blue color
VSID-06-BE-5	LED board for VS-06 with integrated lighting, white color

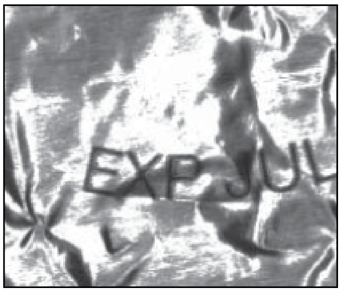
Machine Vision Lighting Principles

Proper lighting is critical to the success of a machine vision application. Depending on the requirements of your application, you may also need to add external lighting from di-soric NERLITE family of machine vision lighting products.

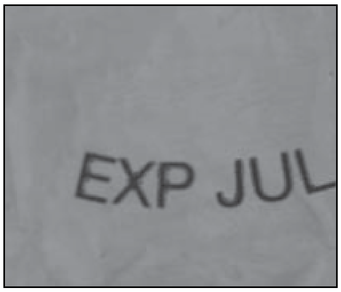
Consider the following when setting up your application:

- Is the surface of the object flat, slightly bumpy, or very bumpy?
- Is the surface matte or shiny?
- Is the object curved or flat?
- What is the color of the object or area being inspected?
- Is the object moving or stationary?

Machine vision lighting should maximize contrast of the areas or features being inspected while minimizing the contrast of everything else.



Before correct lighting



After correct lighting

CHAPTER 4

Using EtherNet/IP

This section provides information necessary for using the VS-06 in an EtherNet/IP environment.

VS-06 EtherNet/IP

Throughout this document, EtherNet/IP may be referred to as “EIP”, and VS-06 may be abbreviated “VS”. The EIP interface version described here is 1.1. This version number is associated with the EIP interface for di-soric Device Type of 100, Machine Vision Smart Cameras. It is not the software version of AutoVision, VisionScape, or VS-06 firmware.

Overview

The EIP interface will be identified as Vendor Specific (100). The interface is designed to support Class 1 Implicit IO data exchange, and Class 3 Explicit messages for serial commands not accessible with Implicit messaging.

Necessary Tools

The following tools are helpful for configuring the EIP:

- AutoVision and FrontRunner
- EtherNet/IP Messaging Tool – can be a PLC or Software Tool, must be capable ofsending explicit messages and establishing Class 1 connections. EIPScan from Pyramid Solutions is an example of such a tool.
- Terminal emulation or serial communication tool that can connect to serial uart and TCP socket, such as HyperTerminal or Putty.

EtherNet/IP Terms of Use

EtherNet/IP Technology is governed by the Open DeviceNet Vendor Association, Inc (ODVA). Any person or entity that makes and sells products that implement EtherNet/IP Technology must agree to the Terms of Usage Agreement issued by ODVA. See **www.odva.org** for details.

EtherNet/IP Object Model

VS-06 uses Class 1 connected messaging to communicate most of its data and services in a single connection.

EIP Identity

Device Type

Device type is 100, Vendor Specific, Machine Vision Smart Camera.

Vendor ID

di-soric ODVA Vendor ID is 1095.

Product Code

The Product Code is 6899.

Interface Revision

Major.Minor = 1.1

Connection Properties: Class 1 Implicit Messaging

Input Assembly Instance (to PLC/client): 102

Output Assembly Instance (to VS-06): 114

Size: Fixed, 320 bytes in both directions

Input Trigger/Trigger Mode: Cyclic

RPI (Requested Packet Interval): Greater than 20 ms recommended. 10 ms to 3.2 s allowed.

Input Type/Connection Type:

- Point-to-Point (PLC OUT, O->T)
- Point-to-Point and Multicast (PLC IN, T->O)

Connection Priority: Scheduled

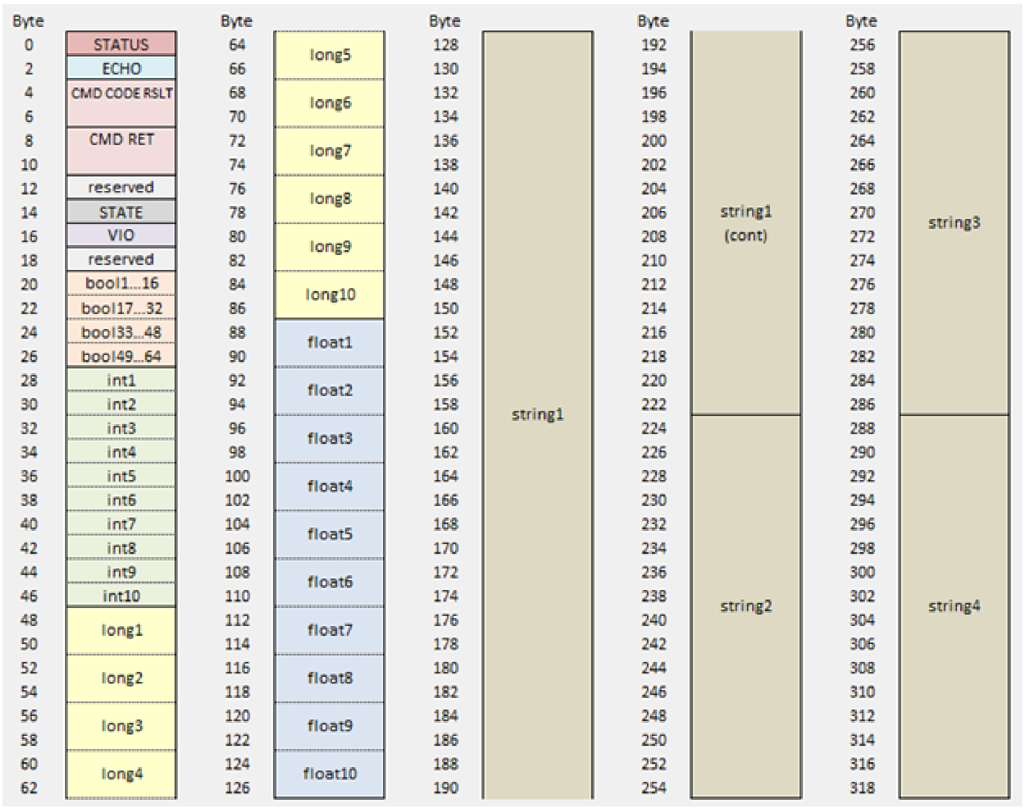
Assembly Layout

Input Assembly

The input assembly layout is described below and shown in the following diagram.

Bytes	Name	Description
0...1	STATUS	Status register of the camera, each bit of this register represents a different state item. See *Camera Status Register* for bit descriptions
2...3	ECHO	This 16 bit word value reflects back to the PLC the value that the PLC wrote to the output assembly ECHO register. The PLC can verify the output assembly has been written to the camera when this value matches the written value.
4...7	CmdCodeRslt	When Status.ExeCmdAck goes active in response to Control.ExeCmd, CmdCodeRslt reflects the result of the command invoked by Control.CmdCode. See *CmdCodeRslt* for definitions.
8...11	CmdRet	When Status.ExeCmdAck goes active in response to Control.ExeCmd, CmdRet contains the data returned from the command invoked by Control.CmdCode. See *CmdRet* for definitions.
12...13	reserved	Reserved for future use.
14...15	State	Device State register. Depending on the current state of the camera, certain STATUS and CONTROL features may or may not be operational. See *State* for definitions.
16...17	VIO	Each bit reflects the state of a virtual IO point. The least significant bit reflects vio point 145, the most significant bit vio point 160
18...19	reserved	Reserved for future use.
20...27	bool1-64	Each bit represents a bool value. The least significant bit of byte 20 reads the value of bool1. The most significant bit of byte 27 reads bool64.
28...47	int1-10	Each pair of sequential bytes represents a 16 bit signed integer value. The 20 bytes represent 10 integers. From bytes 28 & 29 for the value of int1 through bytes 46 & 67 for the value of int10.
48...87	long1-10	Each group of 4 bytes represents a 32 bit signed integer value. The 40 bytes represent 10 long integers. From bytes 48-51 for the value of long1 through bytes 84-87 for the value of long10.
88...127	float1-10	Each group of 4 bytes represents a 32 bit signed integer value. The 40 bytes represent 10 long integers. From bytes 48-51 for the value of long1 through bytes 84-87 for the value of long10.
128...223	string1	These 96 bytes can store a string of up to 92, 8 bit characters, with the first 4 bytes containing the length value.
224...255	string2	Each of these 32 byte groups can store a string of up to 28, 8 bit characters, with the first 4 bytes containing the length value.
256...287	string3	
288...319	string4	

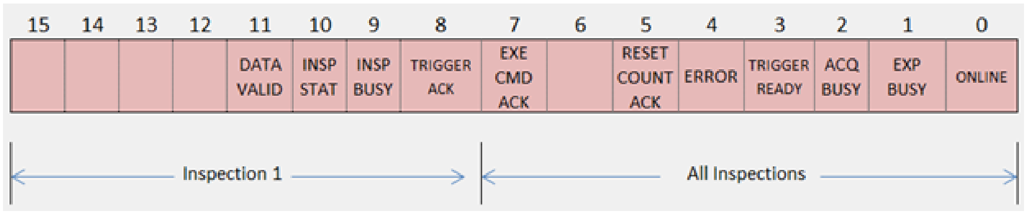
The input assembly layout is shown here:



Byte		Byte		Byte		Byte		Byte	
0	STATUS	64	long5	128	string1	192	string1 (cont)	256	string3
2	ECHO	66		130		194		258	
4	CMD CODE RSLT	68	long6	132		196		260	
6		70		134		198		262	
8	CMD RET	72	long7	136		200		264	
10		74		138		202		266	
12	reserved	76	long8	140		204		268	
14	STATE	78		142		206		270	
16	VIO	80	long9	144		208		272	
18	reserved	82		146		210		274	
20	bool1...16	84	long10	148		212		276	
22	bool17...32	86		150		214		278	
24	bool33...48	88	float1	152		216		280	
26	bool49...64	90		154		218		282	
28	int1	92	float2	156		220		284	
30	int2	94		158		222		286	
32	int3	96	float3	160		224	string2	288	string4
34	int4	98		162		226		290	
36	int5	100	float4	164		228		292	
38	int6	102		166		230		294	
40	int7	104	float5	168		232		296	
42	int8	106		170		234		298	
44	int9	108	float6	172		236		300	
46	int10	110		174		238		302	
48	long1	112	float7	176		240		304	
50		114		178		242		306	
52	long2	116	float8	180		244		308	
54		118		182		246		310	
56	long3	120	float9	184		248		312	
58		122		186		250		314	
60	long4	124	float10	188		252		316	
62		126		190		254		318	

Status: Camera Status Register (16 bit)

Each bit of this register represents a different state of the camera’s operation. A high value of 1 indicates that state is active (true).



Bit	Name	Description
0	ONLINE	Inspections are running
1	EXP BUSY	The camera is busy capturing an image. The camera should not be triggered or the part under inspection moved during this time if illuminated.
2	ACQ BUSY	The camera is busy acquiring an image. The camera cannot be triggered while busy.
3	TRIGGER READY	The camera is ready to be triggered. This is equivalent to ONLINE == 1 and ACQ BUSY == 0.
4	ERROR	An error has occurred. Set the RESET ERROR control bit high to clear.
5	RESET COUNT ACK	This bit mirrors the RESET COUNT control bit. The PLC can be certain the reset command was received by the camera when this goes high. The PLC can then bring the RESET COUNT control signal back low.
7	EXE CMD ACK	This bit mirrors the EXE CMD control bit.
8	TRIGGER ACK	This bit mirrors the TRIGGER control bit.
9	INSP BUSY	This bit is high when inspection 1 is busy processing an image.
10	INSP STAT	This bit represents the inspection 1 status result. It is 1 if the inspection passes. It is only valid when DataValid goes high.
11	DATA VALID	This bit goes high when inspection 1 is complete. The PLC should clear this signal by setting RESET DV high once it has read results.

CmdCodeRslt (32 bit)

The value of CmdCodeRslt is only valid when ExeCmdAck is active (1), in response to ExeCmd being active.

CmdCodeRslt value (base 16 hex)	Meaning
0x0000_0000	Success
0x0100_0000	Fail. Possible reasons: Camera under PC control. Job cannot be changed.
0x0200_0000	Fail: No Job in slot.
0x0300_0000	Fail: Unknown cmd.

CmdRet (32 bit)

The value of CmdRet is only valid when ExeCmdAck is active (1), in response to ExeCmd being active, and CmdCodeRslt is 0 (Success). The following chart shows which CmdCodes return data in the CmdRet register.

CmdRet value (32 bit)	Associated CmdCode	Meaning
0	0x1000_0000 to 0x1300_0000 (Job Change type)	Na
1 – 255	0x1800_0000 (Query Active Job Slot)	Active Job Slot #

State (16 bit)

State reflects the following operational condition of the camera:

State value (16 bit)	Meaning	Typical action required by the client (plc), or system operator
0	Offline	Perform job change or put camera online.
1	Online	Normal runtime operation: Monitor TriggerReady and DataValid signals. Trigger the camera.
2	Changing Vision Job	If camera is under pc control: Wait until State changes to Offline or Online. If plc is controlling the job change: Use ExeCmd, CmdCode, ExeCmdAck, and CmdCodeRslt to complete the operation.
3	Booting*	Wait for camera to transition to Online or Offline.
4	Empty (no Vision Job)	Load a new job from AutoVISION or Front Runner.

*Booting (3) State: This will rarely be seen by the plc.

The value of State determines which Control and Status signals are available:

Control/Status Signal	State 0 (Offline)	1 (Online)	2 (Job Change)	3 (Booting)	4 (Empty)
Control.GO ONLINE	Y				
".GO OFFLINE		Y			
".RESET ERROR					
".RESET COUNT	Y	Y			
".EXE CMD	Y	Y	Y		Y
".TRIGGER		Y			
".RESET DATA VALID		Y			
Status.ONLINE	Y	Y	Y	Y	Y
".ERROR					
".RESET COUNT ACK	Y	Y			
".EXE CMD ACK	Y	Y	Y		Y
".EXP BUSY		Y			
".ACQ BUSY		Y			
".TRIGGER READY		Y			
".TRIGGER ACK		Y			
".INSP BUSY		Y			
".INSP STAT		Y			
".DATA VALID		y			

Where:

Y = Signal is valid for this State

Empty cell = Signal is not valid for this State

VIO Register Bits

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
v160	v159	v158	v157	v156	v155	v154	v153	v152	v151	v150	v149	v148	v147	v146	v145

Output Assembly

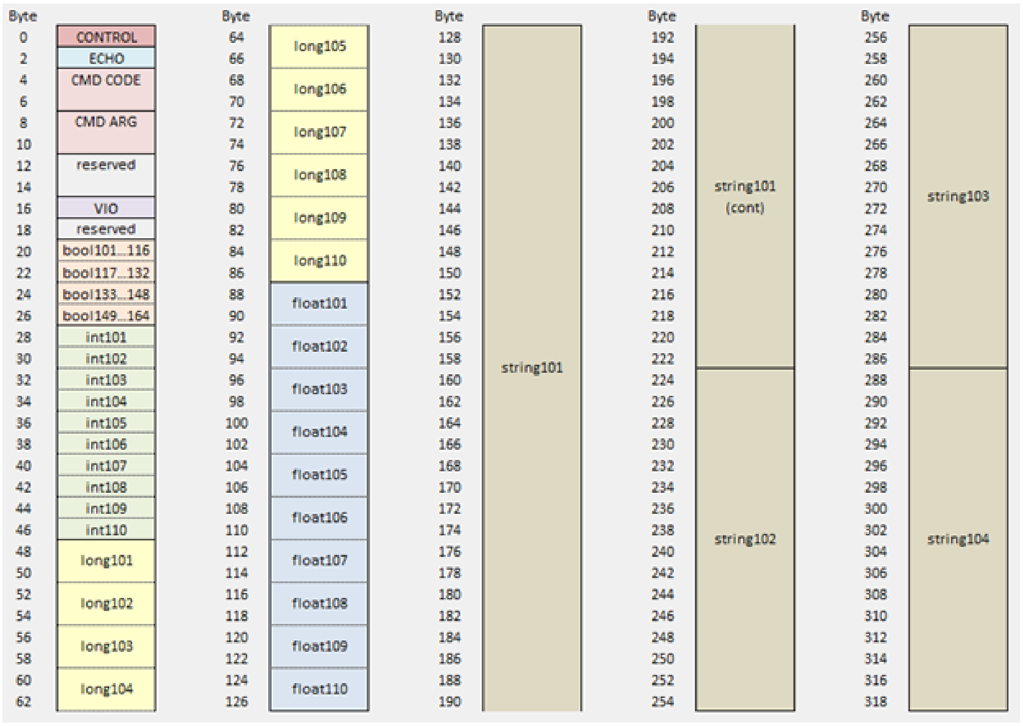
The output assembly layout is described below and shown in the following diagram.

Bytes	Name	Description
0...1	CONTROL	Control register of camera. Each bit of this register represents a different status item. See *Camera Control Register* for bit descriptions
2...3	ECHO	This 16 bit value is reflected back to the PLC in the input assembly ECHO register. The PLC can verify the output assembly has been written to the camera when the input assembly matches this written value.
4...7	CmdCode	Specifies the process invoked in the camera when Control.ExeCmd goes active. See *CmdCode* for definitions.
8...11	CmdArg	Additional argument data for the CmdCode. See *CmdArg* for definition.
12...15	reserved	Reserved for future use.
16...17	VIO	Each bit reflects the state of a virtual IO point. The least significant bit reflects vio point 129, the most significant bit is vio point 144
18...19	Reserved	Reserved for future use.
20...27	bool	Each bit represents a bool value. The least significant bit of byte 20 writes the value of bool101. The most significant bit of byte 27 writes bool164.
28...47	int101-110	Each pair of sequential bytes represents a 16 bit signed integer value. The 20 bytes represent 10 integers. From bytes 28 & 29 to write the value of int101 through bytes 46 & 67 for the value of int110.
48...87	long101-110	Each group of 4 bytes represents a 32 bit signed integer value. The 40 bytes represent 10 long integers. From bytes 48-51 for the value of long101 through bytes 84-87 for the value of long110.
88...127	float101-110	Each group of 4 bytes represents a 32 bit signed integer value. The 40 bytes represent 10 long integers. From bytes 48-51 for the value of long101 through bytes 84-87 for the value of long110.
128...223	string101	These 96 bytes can store a string of up to 92 bytes, with the first 4 bytes containing the length value.
224...255	string102	Each of these 32 byte groups can store a string of up to 28 bytes, with the first 4 bytes containing the length value.
256...287	string103	
288...319	string104	

4

Using EtherNet/IP

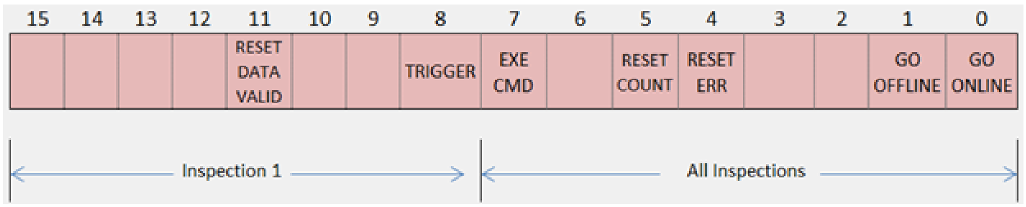
The output assembly layout is shown here:



Byte		Byte		Byte		Byte		Byte	
0	CONTROL	64	long105	128	string101	192	string101 (cont)	256	string103
2	ECHO	66		130		194		258	
4	CMD CODE	68	long106	132		196		260	
6		70		134		198		262	
8	CMD ARG	72	long107	136		200		264	
10		74		138		202		266	
12	reserved	76	long108	140		204		268	
14		78		142		206		270	
16	VIO	80	long109	144		208		272	
18	reserved	82		146		210		274	
20	bool101...116	84	long110	148		212		276	
22	bool117...132	86		150		214		278	
24	bool133...148	88	float101	152		216		280	
26	bool149...164	90		154		218		282	
28	int101	92	float102	156		220		284	
30	int102	94		158		222		286	
32	int103	96	float103	160		224	string102	288	string104
34	int104	98		162		226		290	
36	int105	100	float104	164		228		292	
38	int106	102		166		230		294	
40	int107	104	float105	168		232		296	
42	int108	106		170		234		298	
44	int109	108	float106	172		236		300	
46	int110	110		174		238		302	
48	long101	112	float107	176		240		304	
50		114		178		242		306	
52	long102	116	float108	180		244		308	
54		118		182		246		310	
56	long103	120	float109	184		248		312	
58		122		186		250		314	
60	long104	124	float110	188		252		316	
62		126		190		254		318	

Control: Camera Control Register (16 bit)

Each bit of this register controls a function on the camera. Transitions from a low state of 0, to a high state of 1, initiates the associate operation. The PLC should return the state of the control bit back to 0 after it has acknowledged the camera has processed the control. Unused bits should remain 0.



Bit	Name	Description
0	GO ONLINE	Start all inspections running
1	GO OFFLINE	Stop all inspections
4	RESET ERROR	Reset ERROR in the Status register
5	RESET COUNT	Reset all inspection counts
7	EXE CMD	Execute the command specified by Control.CmdCode
8	TRIGGER	Trigger Inspection 1. The inspection must be configured for a triggered image acquisition.
11	RESET DATA VALID	Reset the Data Valid signal of the Status register

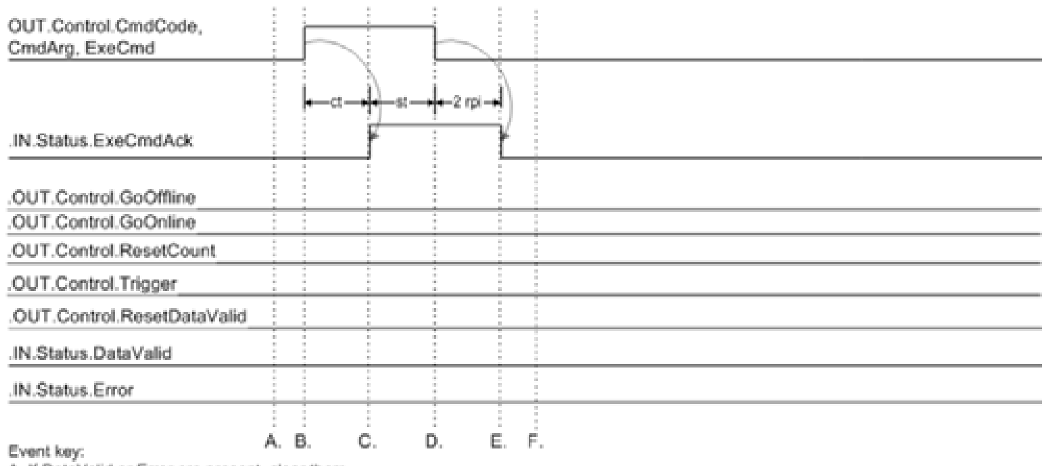
CmdCode and CmdArg (32 bit)

Specifies the process invoked in the camera when Control.ExeCmd goes active.

List of available CmdCodes, and associated CmdArg

CmdCode value	**CmdArg**	**Operations performed**
0x1000_0000	Job Slot (1-255)	Go Offline, Load job from specified slot
0x1100_0000	Job Slot (1-255)	Go Offline, Load job from specified slot, Go Online
0x1200_0000	Job Slot (1-255)	Go Offline, Load job from specified slot, Make it the boot job
0x1300_0000	Job Slot (1-255)	Go Offline, Load job from specified slot, Make it the boot job, and Go Online
0x1800_0000	na	Query active job slot. *CmdRet* will contain the active job slot number when the operation is done.

CmdCode and ExeCmd Operation



Event key:
A. If DataValid or Error are present, clear them.
Set the following control signals idle, and keep them idle while the command is processed by the camera:
GoOffline, GoOnline, Trigger, ResetDataValid, ResetCount, ResetError.
If the command operation is a job change, populate the output tags required to configure the new job (bool, int, long, float, string).
B. Populate CmdCode and CmdArg, then activate ExeCmd.
C. Camera executes the command (may take up to a minute). While processing a Job Change command, State will be 2. Camera activates ExeCmdAck when it is done processing the command.
D. When the PLC sees an active ExeCmdAck, verify CmdCodeRslt is 0, and Error is 0. Process CmdRet if needed, then clear ExeCmd.
E. Camera clears ExeCmdAck when ExeCmd goes inactive. When ExeCmdAck goes inactive, CmdCodeRslt and CmdRet are no longer valid, and it may take a few seconds for the camera State and Online signals to settle to a final value (typically Online or Offline).
F. Camera can now be put online and triggered.

Notes:
st = PLC program scan time
ct = Command processing time in the camera. May take up to a minute for some commands.
rpi = Requested Packet Interval. Configured in the plc's EIP module connection properties. Allowed rpi is 10 ms to 3.2 s.
All signals represent the state of plc tags.

VIO Register Bits

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
v144	v143	v142	v141	v140	v139	v138	v137	v136	v135	v134	v133	v132	v131	v130	v129

Connection Properties: Class 3 Explicit Messaging

All Class 1 IO assembly data and additional data are accessible via Explicit message. Input data (VS-06 to PLC/Client) occupies attributes 1 to 100 of the classes. Output data (PLC/Client to VS-06) occupies attributes 101 to 200.

Service:

- Get Attribute Single (0xE)
- Set Attribute Single (0x10)

Classes:

- bool = 104 (0x68)
- int = 105 (0x69)
- long = 106 (0x6A)
- float = 107 (0x6B)
- string = 108 (0x6C)
- control/status (mixed data types) = 109 (0x6D)

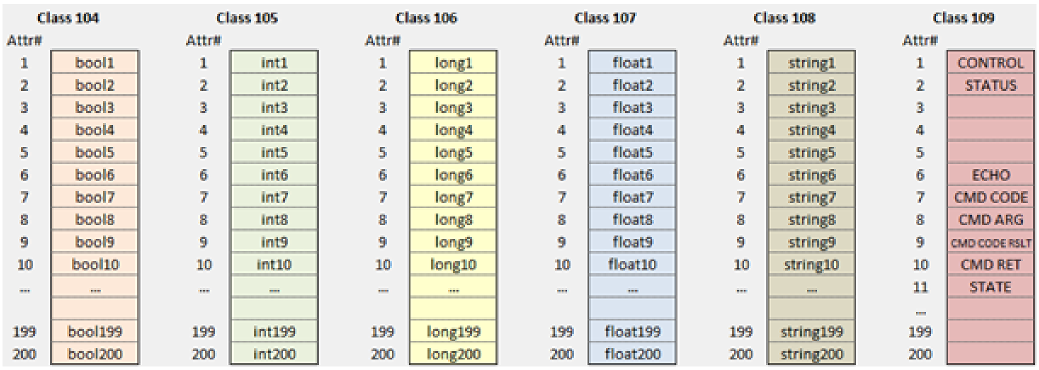
Instance: 1

Attribute:

- 1 to 100 = In to PLC/Client
- 101 to 200 = Out to VS

Attribute Layout

When using explicit EIP messaging, all global data objects can be read or written. Each data type is stored in its own class object and an instance of 1 to read the global data. For example to read float2 the EIP request would be for Service Code 14 (0xE), Class 107 (0x6B), Instance 1, Attribute 2.



Class 104		Class 105		Class 106		Class 107		Class 108		Class 109	
Attr#		Attr#		Attr#		Attr#		Attr#		Attr#	
1	bool1	1	int1	1	long1	1	float1	1	string1	1	CONTROL
2	bool2	2	int2	2	long2	2	float2	2	string2	2	STATUS
3	bool3	3	int3	3	long3	3	float3	3	string3	3	
4	bool4	4	int4	4	long4	4	float4	4	string4	4	
5	bool5	5	int5	5	long5	5	float5	5	string5	5	
6	bool6	6	int6	6	long6	6	float6	6	string6	6	ECHO
7	bool7	7	int7	7	long7	7	float7	7	string7	7	CMD CODE
8	bool8	8	int8	8	long8	8	float8	8	string8	8	CMD ARG
9	bool9	9	int9	9	long9	9	float9	9	string9	9	CMD CODE RSLT
10	bool10	10	int10	10	long10	10	float10	10	string10	10	CMD RET
...	...	...	...	...	...	...	...	...	...	11	STATE
										...	
199	bool199	199	int199	199	long199	199	float199	199	string199	199	
200	bool200	200	int200	200	long200	200	float200	200	string200	200	

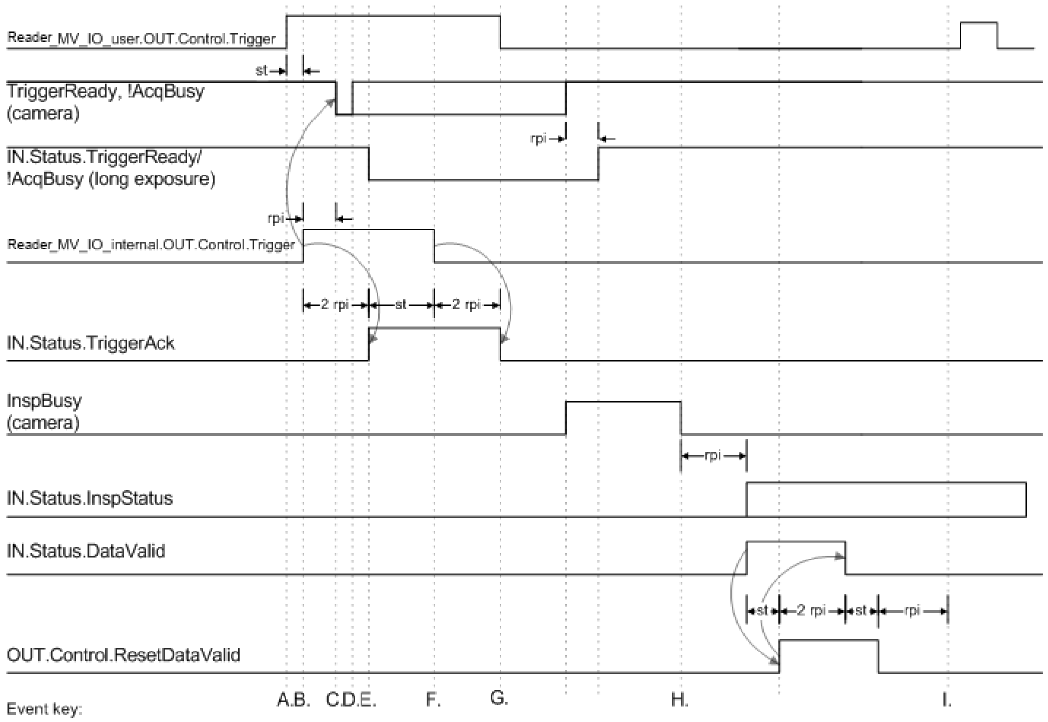
The value received in response to Get Attribute Single depends on the type:

- bool will return a 16 bit word with 0 for false or 1 for true
- Ints will return a 16 bit signed integer
- longs will return a 32 bit signed integer
- floats will return a 32 bit floating point number
- strings will return a counted string. Total size of a string data item is 2048 bytes. This includes a 4 byte “length” field followed by 2044 eight bit characters. When accessing strings explicitly, they are not limited to the size in the IO assemblies. Eg. string3 is limited to 28 bytes in the input assembly. If the actual string is longer than 28 bytes, it will be truncated when reading via the assembly, but not truncated when reading the same string via an attribute explicitly.

Assembly Class 109 can be used to read and write special EIP specific registers.

Attr#	Name	Description
1	CONTROL	The control register (16 bit). See *Camera Control Register* for bit definitions.
2	STATUS	The status register (16 bit). See *Camera Status Register* for bit definitions.
6	ECHO	The ECHO register (16 bit) (read only if implicit write is enabled)
7	CMD CODE	The command code register (32 bit). See *CmdCode*.
8	CMD ARG	The command argument register (32 bit). See *CmdArg*.
9	CMD CODE RSLT	The command code result register (32 bit). See *CmdCodeRslt*.
10	CMD RET	The command return value register (32 bit). See *CmdRet*.
11	STATE	The device state register (16 bit). See *State* for definitions.

EIP Control/Status Signal Operation


Reader_MV_IO_user.OUT.Control.Trigger
st
TriggerReady, !AcqBusy (camera)
rpi
IN.Status.TriggerReady/ !AcqBusy (long exposure)
rpi
Reader_MV_IO_internal.OUT.Control.Trigger
2 rpi
st
2 rpi
IN.Status.TriggerAck
InspBusy (camera)
rpi
IN.Status.InspStatus
IN.Status.DataValid
st
2 rpi
st
rpi
OUT.Control.ResetDataValid
A.B.
C.D.E.
F.
G.
H.
I.

Event key:
A. On rising edge of system trigger, the user app activates Reader_MV_IO_user.OUT.Control.Trigger to trigger the demo code.
B. Demo code detects rising edge of Reader_MV_IO_user.OUT.Control.Trigger, and if the camera is ready, sends a trigger to the camera.
C. Camera acquisition begins (may be delayed by one rpi).
D. If the camera's exposure time is shorter than the rpi, no change will be seen in TriggerReady and AcqBusy plc IN tags.
E. Camera firmware acks the trigger. The demo code may not see the ack until two rpi after the trigger was sent (event B).
F. Demo code detects TriggerAck and clears the Trigger.
G. Demo code detect falling edge of TriggerAck and clears the user Trigger.
H. Camera internal signal DataValid will go high when InspBusy goes low
I. Plc logic must delay one rpi time before re-asserting ResetDataValid

Notes:
1. The chart shows the workings of the Trigger and ResetDataValid Control signals, and the TriggerAck and DataValid Status signals.
2. st = plc program scan time
3. rpi = Requested Packet Interval. Configured in the plc's EIP module connection properties. Allowed rpi is 10 ms to 3.2 s.
4. All signals represent the state of plc tags, except where noted as "(camera)". The cam signals shown are visible in the EIP interface, but the state of the plc tags and internal firmware signals will be different for at least one or two requested packet intervals (rpi).
5. The plc is running the demo code distributed with the camera. The demo code and user app use the Reader_MV_IO_user tag set as the primary control, status, and data interface for the user app. All signal operations are still true even if the plc demo code is not used.
6. TriggerReady/!AcqBusy: VS-06 camera exposure times can range from less than 1 ms, up to 100 ms.

Data Type Descriptions and Equivalents in PLC and EDS/CIP Environments

AV / VisionScape	Description	RSLogix equivalent	Description	EDS / EIP equivalents	Description
Bool	1 bit	BOOL	1 bit	BOOL	1 bit
				WORD	16 BOOLs
				LWORD	64 BOOLs
Int	16 bit signed integer	INT	16 bit signed integer	INT	16 bit signed integer
Long	32 bit signed integer	DINT	32 bit signed integer	DINT	32 bit signed integer
Float	32 bit floating point	REAL	32 bit floating point	REAL	32 bit floating point
String	32 bit length field followed by 8 bit ASCII characters	STRING	32 bit length field followed by 8 bit ASCII characters	DINT + USINT[]	DINT (length) + USINT array of characters. USINT = 8 bit integer

PLC Tags and Serial Command Names

PLC tags are separated into IN and OUT for data direction. Within the IN and OUT groups, the tags are sub-divided into fixed "Status" and "Control" fields, plus user-defined linked data fields. This table shows how PLC tag names correspond to serial commands.

IN (VH to PLC)			OUT (PLC to VH)		
PLC tag prefix	Serial cmd prefix	Tag name	PLC tag prefix	Serial cmd prefix	Tag name
IN.Status.	eip.status.	Online (1)	OUT.Control.	eip.control.	GoOnline[i]
IN.Status.	eip.status.	Online (0)	OUT.Control.	eip.control.	GoOffline[ii]
IN.Status.	eip.status.	Error	OUT.Control.	eip.control.	ResetError
IN.Status.	eip.status.	ResetCountAck	OUT.Control.	eip.control.	ResetCount
IN.Status.	eip.status.	TriggerAck	OUT.Control.	eip.control.	Trigger
IN.Status.	eip.status.	DataValid	OUT.Control.	eip.control.	ResetDataValid
IN.Status.	eip.status.	ExeCmdAck	OUT.Control.	eip.control.	ExeCmd
IN.Status.	eip.status.	TrigReady[iii]	-	-	-
IN.Status.	eip.status.	AcqBusy	-	-	-
IN.Status.	eip.status.	ExpBusy	-	-	-
IN.Status.	eip.status.	InspBusy	-	-	-
IN.Status.	eip.status.	InspStat	-	-	-
IN.Status.	eip.	Echo	OUT.Control.	eip.	Echo
IN.Status.	eip.	CmdCodeRslt	OUT.Control	eip.	CmdCode
IN.Status	eip.	CmdRet	OUT.Control	eip.	CmdArg
IN.Status.	eip.	State	-	-	-
IN.vio.	io.	v[145-160]	OUT.vio.	io.	v[129-144]
IN.bool.	eip.	bool[1-100]	OUT.bool.	eip.	bool[101-200][iv]
IN.int.	eip.	int[1-100]	OUT.int.	eip.	int[101-200][v]
IN.long.	eip.	long[1-100]	OUT.long.	eip.	long[101-200]
IN.float.	eip.	float[1-100]	OUT.float.	eip.	float[101-200]
IN.string.	eip.	string[1-100]	OUT.string.	eip.	string[101-200]

[i] When GoOnline is changed from 0 to 1, Online goes to 1.
[ii] When GoOffline is changed from 0 to 1, Online goes to 0.
[iii] TrigReady, AcqBusy, ExpBusy, InspBusy, and InspStat are all IN-direction data only.
[iv] bool1-bool64 are mapped to PLC tags in the IN assembly. Bool101-bool164 are mapped to PLC tags in the OUT assembly. Bool members numbered 65-100 and 165-200 are accessible via Explicit Message only.
[v] For int, long, float, and string data:
Data members numbered 1-10 are mapped to PLC tags in the IN assembly.
Data members numbered 101-110 are mapped to PLC tags in the OUT assembly.
Data members numbered 11-100 and 111-200 are accessible via Explicit Message only.

A

APPENDIX A

Connector Pinouts

This section contains information about VS-06 Smart Camera connectors:

- M12 12-Pin Plug on page A-2
- M12 8-Pin Socket on page A-3

VS-06 Smart Camera Connectors

Connector A – M12 12-Pin Plug – Power, I/O, and Serial

Figure A–1 shows the M12 12-pin plug at connector A.

FIGURE A–1. VS-06 Connector A – M12 12-Pin Plug

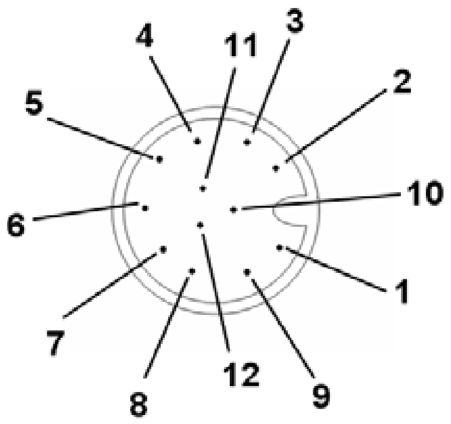


Table A–1 describes the M12 12-pin plug signals.

TABLE A–1. VS-06 Connector A – M12 12-Pin Plug

Pin	Function
1	Trigger
2	Power
3	Default
4	Input 1
5	Output 1
6	Output 3
7	Ground
8	Input Common
9	Host RxD
10	Host TxD
11	Output 2
12	Output Common

Connector B – M12 8-Pin Socket – Ethernet

Figure A-2 shows the M12 8-pin socket at connector B.

FIGURE A–2. Connector B – M12 8-Pin Socket

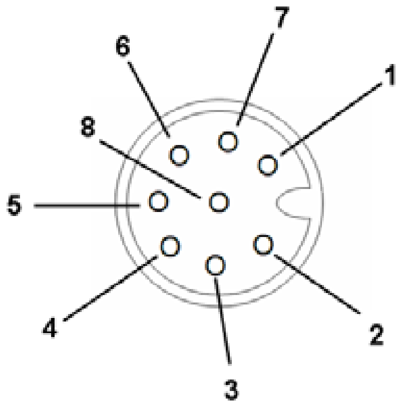


Table A-2 describes the M12 8-pin socket signals.

TABLE A–2. Connector B – M12 8-Pin Socket

Pin	Function
1	Terminated
2	Terminated
3	Terminated
4	TX (–)
5	RX (+)
6	TX (+)
7	Terminated
8	RX (–)

APPENDIX B Cable Specifications

This section contains information about VS-06 Smart Camera cables.

Note: Cable specifications are published for information only. di-soric does not guarantee the performance or quality of cables provided by other suppliers.

TABLE B–1. Cable Part Numbers and Descriptions

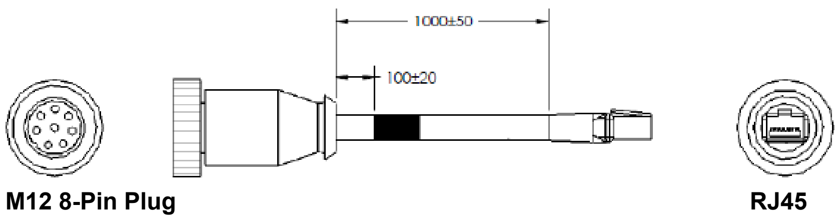
Part Number	Descriptions
VKHM-Z-5/RJ45	Cable, Host, Ethernet, M12 8-pin Plug to RJ45, 5m
VSID-PS-24V-ES	Power Supply, M12 12-pin Socket, 1.3 m
VSID-IB-ES	Interface box for Ethernet devices

VKHM-Z-5/RJ45 Cable, Host, Ethernet, M12 8-pin Plug to RJ45, 5 m

The VKHM-Z-5/RJ45 Cable, Host, Ethernet, M12 8-pin Plug to RJ45, 5 m is a meter cable with an 8-pin M12 connector on one end and a standard RJ45 connector on the other end.

Figure B-1 shows the VKHM-Z-5/RJ45 Cable, Host, Ethernet, M12 8-pin Plug to RJ45, 5m.

FIGURE B–1. Cable, Host, Ethernet, M12 8-pin Plug to RJ45, 1 m



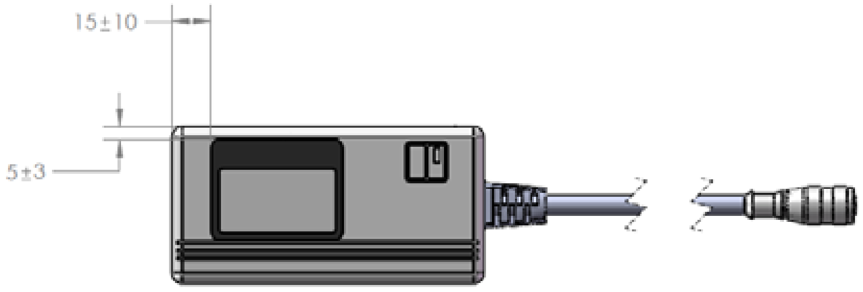
Important: Be sure that the retaining clip on the RJ45 connector has locked into place in the Ethernet receptacle on the PC and is not being impeded by the rubber housing.

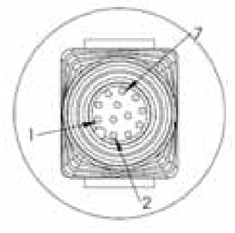
VSID-PS-24V-ES Power Supply, M12 12-pin Socket, 1.3 m

The VSID-PS-24V-ES Power Supply, M12 12-pin Socket, 1.3 m is a 90-254 VAC, +24VDC power supply.

Figure B-3 shows the VSID-PS-24V-ES Power Supply, M12 12-pin Socket, 1.3 m.

FIGURE B–2. Power Supply, M12 12-pin Socket, 1.3 m





M12 12-Pin Socket

C

C

APPENDIX C

General Specifications

This section contains specifications and dimensions for the VS-06 Smart Camera and VS-06 C-Mount Smart Camera.

TABLE C–1. General Specifications

Part Number	**VS-06-BM2-15-ES**	**VS-06-BM2-30-ES**	**VS-06-BM2-45-ES**	**VS-06-BC3-15-ES**	**VS-06-BC3-30-ES**	**VS-06-BC3-45-ES**
Sensor	WVGA (752 x 480) CMOS			SXGA (1280 x 960) CCD		
Sensor Color	Monochrome					
Height	1.59" (40.5 mm)					
Width	2.27" (57.6 mm)					
Depth	3.79" (96.3 mm)					
Weight	10 oz. (280 g)					
Power	5-28VDC, 200mV p-p max ripple, 170mA at 24VDC (typ.), 15.5 watts (max.)			5-28VDC, 200mV p-p max ripple, 135mA at 24VDC (typ.), 13 watts (max.)		
Connector	M12 12-pin Ultra-Lock (Connector A) and M12 8-pin Ultra-Lock (Connector B)					
Lens Type	Built-In Liquid Lens					
Communications	RS-232 or Ethernet					

TABLE C–1. **General Specifications (Continued)**

Illumination	**High Output LEDs:** .564mW, 470, 525, 617nm	
Laser Output	5.0mW max.; **Type:** Laser diode; **Output Wavelength:** 655nm nominal; **Operating Life:** 50,000 hours @ 25° C; **Safety Class:** Class 1 Visible Laser	
Indicators	**LEDs:** Trigger, Pass, Fail, Mode, Power, Network Activity, I/O; **Green Flash:** Pass; **Red X:** Target	
I/O	**Learn/Trigger:** Bi-directional, optoisolated, 4.5–28V rated, (13mA at 24VDC); **Outputs (1, 2, 3):** Bi-directional, optoisolated, 1–28V rated, (I_{CE} <100mA at 24VDC, current limited by user)	
Image Acquisition	Progressive scan, square pixel	
Focal Range	1” (33 mm) to ∞ (liquid lens autofocus - standard VS-06 only)	
Shutter	6µs to 100ms (1/150,000 to 1/10) Default = 666µs (1/1,500)	25µs to 100ms (1/40,000 to 1/10) Default = 400µs (1/2,500)
Operating Temperature	0° to 45° C (32° to 113° F)	0° to 50° C (32° to 122° F)
Storage Temperature	–29° to 70° C (–20° to 158° F)	
Humidity	Up to 90% (non-condensing)	
Compliance	**CDRH**, **FCC**, **UL/cUL**, **CE (General Immunity for Light Industry:** EN 55024:1998 ITE Immunity Standard; **Radiated and Conducted Emissions of ITE Equipment:** EN 55022:98 ITE Disturbances**)**, **CB**, **BSMI**	

TABLE C–1. Specifications (Continued)

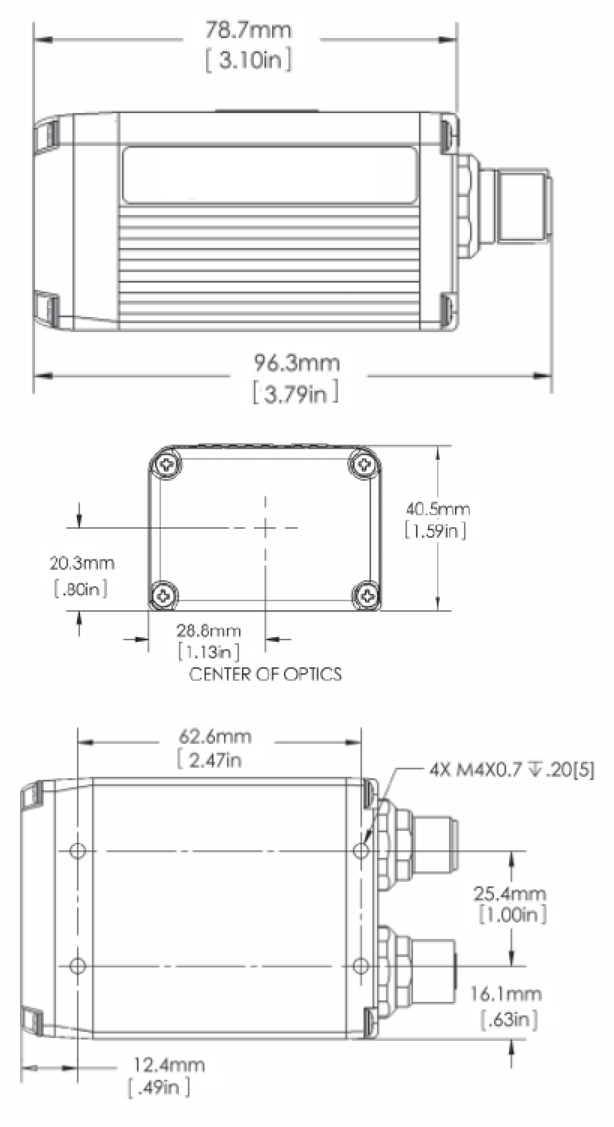
Part Number	VS-06E-BC3-15-ES	VS-06E-BC3-30-ES	VS-06E-BC3-45-ES	VS-06E-BM2-15-ES	VS-06E-BM2-30-ES	VS-06E-BM2-45-ES
Sensor	1/3”, SXGA (1280 x 960) CCD, up to 20 fps			1/3”, WVGA (752 x 480) CMOS, up to 60 fps		
Sensor Color	Monochrome					
Height	1.59” (40.5 mm)					
Width	2.27” (57.6 mm)					
Depth	3.79” (96.3 mm)					
Weight	10 oz. (280 g)					
Power	5-28VDC, 200mV p-p max ripple, 170mA at 24VDC (typ.)			5-28VDC, 200mV p-p max ripple, 135mA at 24VDC (typ.)		
Connector	M12 12-pin Ultra-Lock (Connector A) and M12 8-pin Ultra-Lock (Connector B)					
Lens Type	Built-In Liquid Lens (standard VS-06 only)					
Communications	Ethernet					
Illumination	**High Output LEDs:** .564mW, 470, 525, 617nm					
Laser Output	5.0mW max.; **Type:** Laser diode; **Output Wavelength:** 655nm nominal; **Operating Life:** 50,000 hours @ 25° C; **Safety Class:** Class 1 Visible Laser					
Indicators	**LEDs:** Trigger, Pass, Fail, Mode, Power, Network Activity, I/O; **Green Flash:** Pass; **Red X:** Target					
Discrete I/O	**Learn/Trigger:** Bi-directional, optoisolated, 4.5–28V rated, (13mA at 24VDC); **Outputs (1, 2, 3):** Bi-directional, optoisolated, 1–28V rated, (I_{CE} <100mA at 24VDC, current limited by user)					
Image Acquisition	Progressive scan, square pixel					
Focal Range	1” (33 mm) to ∞ (liquid lens autofocus - standard VS-06 only)					
Shutter	6µs to 100ms (1/150,000 to 1/10) Default = 666µs (1/1,500)			25µs to 100ms (1/40,000 to 1/10) Default = 400µs (1/2,500)		
Operating Temperature	0° to 45° C (32° to 113° F)			0° to 50° C (32° to 122° F)		
Storage Temperature	–29° to 70° C (–20° to 158° F)					
Humidity	Up to 90% (non-condensing)					
Compliance	**CDRH**, **FCC**, **UL/cUL**, **CE (General Immunity for Light Industry:** EN 55024:1998 ITE Immunity Standard; **Radiated and Conducted Emissions of ITE Equipment:** EN 55022:98 ITE Disturbances**)**, **CB**, **BSMI**					

TABLE C–1. **Specifications (Continued)**

Part Number	**VS-06-BC3-00-ES**	**VS-06E-BC3-00-ES**	**VS-06-BM2-00-ES**	**VS-06E-BM2-00-ES**	**VS-06-BM4-00-ES**	**VS-06E-BM4-00-ES**
Sensor	1/3”, SXGA (1280 x 960) CCD, up to 20 fps		1/3”, WVGA (752 x 480) CMOS, up to 60 fps		2/3”, WUXGA (2048 x 1088) CMOS, up to 48 fps	
Sensor Color	Monochrome					
Height	4.03” (102.3 mm)					
Width	2.27” (57.6 mm)					
Depth	1.59” (40.5 mm)					
Weight	11 oz. (320 g)					
Power	5-28VDC, 200mV p-p max ripple, 170mA at 24VDC (typ.)		5-28VDC, 200mV p-p max ripple, 135mA at 24VDC (typ.)		5-28VDC, 200mV p-p max ripple, 140mA at 24VDC (typ.)	
Connector	M12 12-pin Ultra-Lock (Connector A) and M12 8-pin Ultra-Lock (Connector B)					
Lens Type	C-Mount Lens					
Communications	Ethernet					
Illumination	External Illumination Required					
Laser Output	N/A					
Indicators	**LEDs:** Trigger, Pass, Fail, Mode, Power, Network Activity, I/O					
Discrete I/O	**Learn/Trigger:** Bi-directional, optoisolated, 4.5–28V rated, (13mA at 24VDC); **Outputs (1, 2, 3):** Bi-directional, optoisolated, 1–28V rated, (I_{CE} <100mA at 24VDC, current limited by user)					
Image Acquisition	Progressive scan, square pixel					
Focal Range	Depends on lens					
Shutter	6μs to 100ms (1/150,000 to 1/10) Default = 666μs (1/1,500)		25μs to 100ms (1/40,000 to 1/10) Default = 400μs (1/2,500)		25μs to 100ms (1/40,000 to 1/10) Default = 400μs (1/2,500)	
Operating Temperature	0° to 45° C (32° to 113° F)		0° to 50° C (32° to 122° F)			
Storage Temperature	–29° to 70° C (–20° to 158° F)					
Humidity	Up to 90% (non-condensing)					
Compliance	**CDRH**, **FCC**, **UL/cUL**, **CE (General Immunity for Light Industry:** EN 55024:1998 ITE Immunity Standard; **Radiated and Conducted Emissions of ITE Equipment:** EN 55022:98 ITE Disturbances**)**, **CB**, **BSMI**					

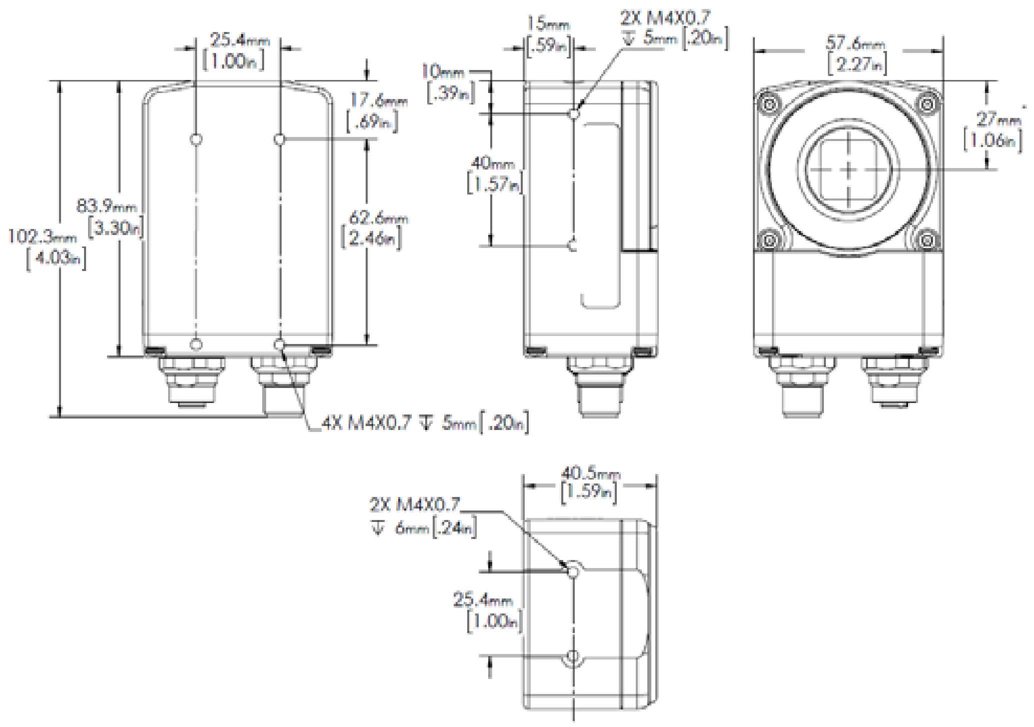
Dimensions

FIGURE C–1. VS-06 Smart Camera Dimensions



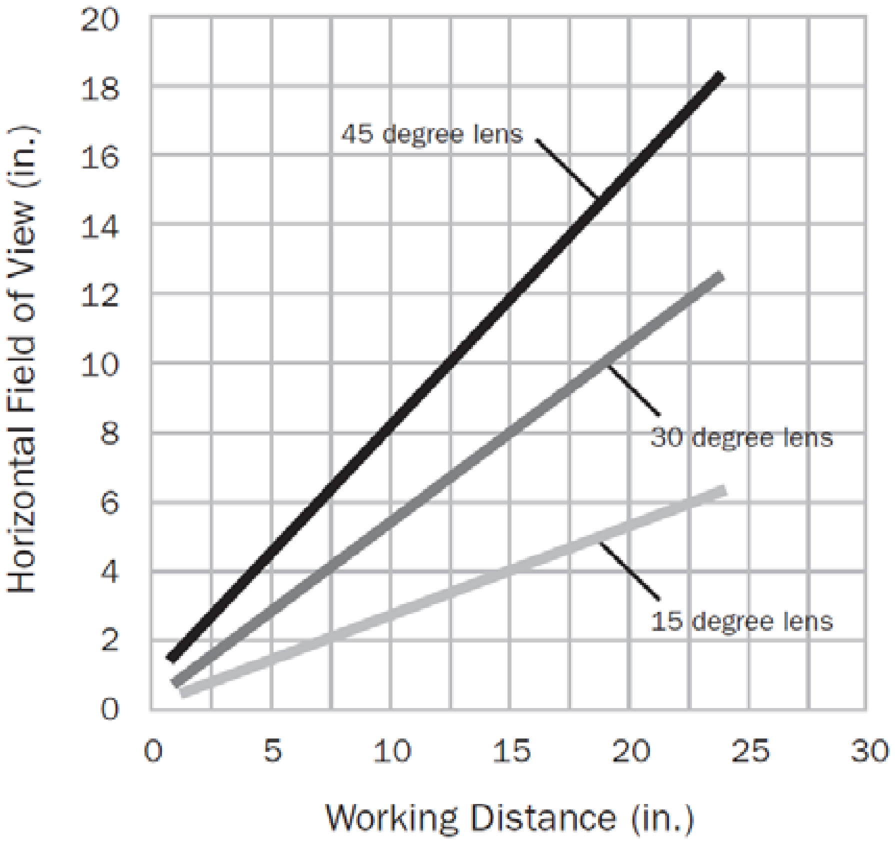
Note: Nominal dimensions shown. Typical tolerances apply.

FIGURE C–2. VS-06 C-Mount Smart Camera Dimensions



Note: Nominal dimensions shown. Typical tolerances apply.

Field of View and Working Distance



APPENDIX D

Web HMI for VS-06

This appendix contains information about VS-06 support for Visualization HMIs.The VS-06 features a built in runtime monitoring web page that can be viewed from any supported browser on the same network. Supported browsers include:

- Internet Explorer 5.0 or later
- Firefox 3.0 or later

A built-in runtime HTML monitoring page suitable for HMI Panels that support Internet Explorer 5.0 or later browser such as the SIMATIC M277 Panel is available on the VS-06. Note that the runtime page can also be displayed with the Firefox or Safari web browsers.

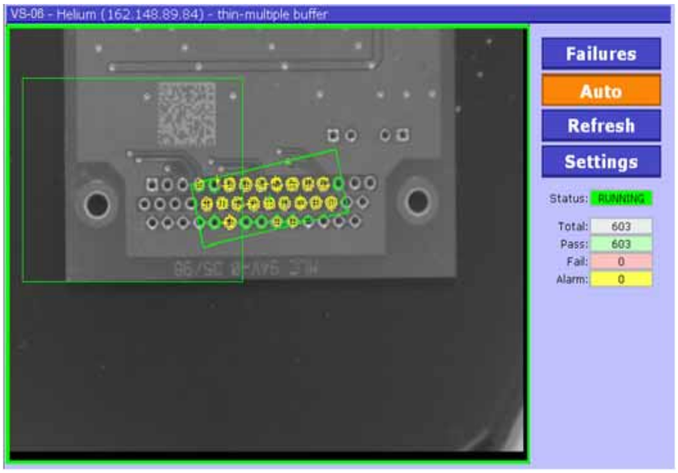
The Runtime Page shows an image from the VS-06, along with inspection counters and buttons to control certain aspects of the display. A title bar displays the camera name, ip address and resolution. Options are available to change if and where the counters, buttons, and titlebar are displayed. Additionally, up to 10 results values from the job can be displayed along with each image. These values can either be overlayed over the image, or shown as a tabular report underneath the image.

All settings and options are set by the user via a series of option pages which can appear over the main display. All parameters are saved as cookies in the web browser environment, so that the next time the Runtime Page is loaded for that device, the layout and settings are retained.

The Runtime Image Page is accessed via a URL which contains the IP address of the camera, and optional parameters. The default page is accessed by simply specifying the IP address of the camera in a web browser, for example:

http:// 161.218.121.58 (example only, actual IP address of the VS-06 should be used)

If no previous settings have been set by the user, the display will be similar to the following:

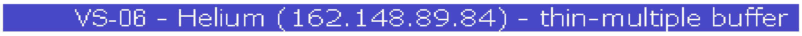


The default behavior is:

- Images and counters are for the first inspection in the job
- All images (pass & fail) are shown
- The display is automatically refreshed at regular intervals (auto=on)
- Graphics are overlaid on the image (note: not all graphics are available)
- A border is drawn around the image signifying the status of the inspection: green=pass, red=fail

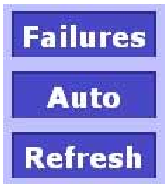
The web page includes the following elements:

- **Title Bar** specifying the name of the camera, IP address, and job (avp) filename. Note that the file extension (.avp) is removed from the displayed filename.

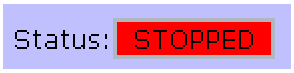


VS-06 - Helium (162.148.89.84) - thin-multiple buffer

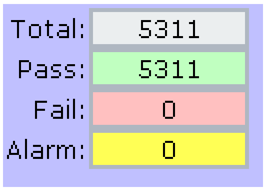
- **Failures** Push Button – when this button is selected, only images related to failed inspections are displayed
- **Auto** Push Button – when this button is selected, the image and counters are updated automatically. If the button is not selected, both the image and counters are frozen.
- **Refresh** Push Button – pushing this button manually updates the image and counters



- **Status** – the run status of the inspection – RUNNING or STOPPED



- **Counters** – the Total, Pass, Fail and Alarm counters are shown for the selected inspection



Total:	5311
Pass:	5311
Fail:	0
Alarm:	0

Adding Options to the Base URL

An option can be specified by adding it to the end of the URL as follows:

http://ip_address/?option=value

Note the question mark "?" separating the URL from the optional parameter(s).

Additional options are specified by separating them with the ampersand "&" character.

http://ip_address/?option1=value1&option2=value2&option3=value3

Basic Options

NOTE: Some basic options can be changed by specifying optional values at the end of the URL. A much richer superset of these options can be configured by using the Settings Pages described below. It is possible to completely control the behavior of the Runtime Page without the use of optional parameters in the URL.

The graphics overlay can be turned on or off by using the "graphics" URL option. This is a setting that can have the value "on" or "off". As an example, to turn the display of graphics off, the web page can be launched with the following URL:

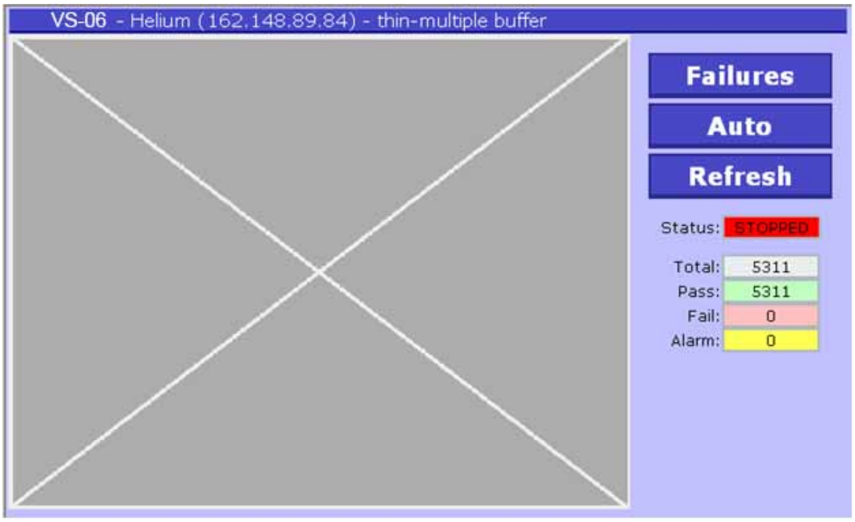
http:// 161.218.121.58/?graphics=off

Option Name	Values	Default	Comments
graphics	on, off	on	**on** = graphics are shown overlaid on the image. (note: not all graphics are supported for web page display) **off** = graphics are not shown
type	last, failed	last	**last** = show last image for the selected inspection **failed** = show last failed image for the selected inspection
passfailborder	on, off	on	**on** = show a border around the image **off** = no border displayed

Note: ROI graphics are not produced by applications created by AutoVision. They are displayed for applications created in VisionScape FrontRunner.

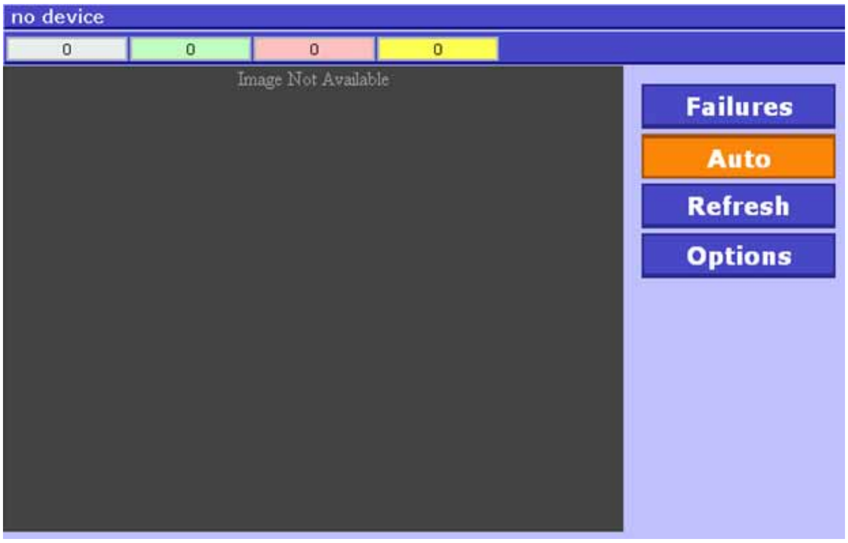
Layout Options

The overall layout of the Runtime Page can be configured. To change the layout, use the Settings screens as described in the next section. The following shows a default configuration:

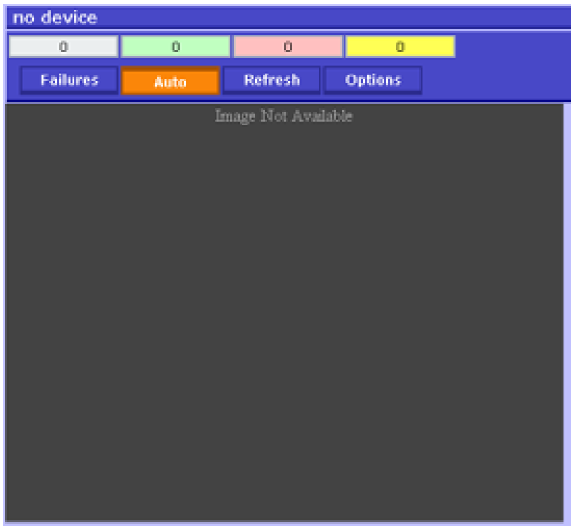


Buttons, status, and counters appear to the right of the image area. The buttons are size for use via a touch screen.

The following illustrates that the layout has been changed to position the counters at the top, shown without titles to save room. Additionally, an Options button now appears in the right side area.



Another example with buttons and counters at the top:



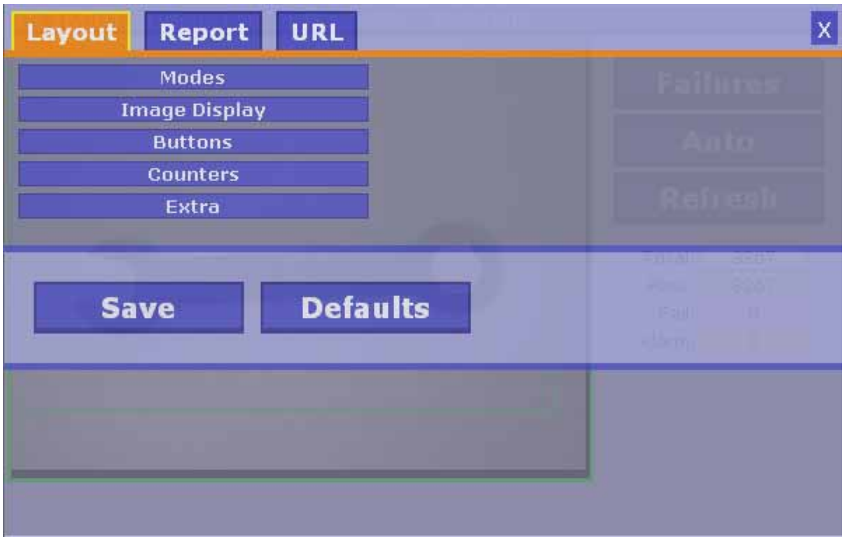
It is also possible to hide all elements except the image.

Settings Pages

All of the options and settings can be configured by using the settings pages. By default, there is no “Settings” button, so to show the settings pages, the URL should be specified with the “setopt=1” parameter as follows:

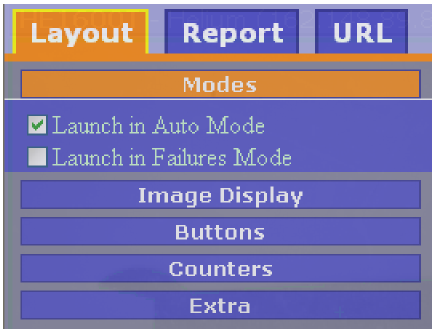
http://ip_address/?setopt=1

This will display the Runtime Page overlayed with the Options Setup page as follows:



The tabs at the top of the screen can be used to navigate between the several setup pages. To close the setup screens and return to the main display, use the close button (“X”) at the upper right corner.

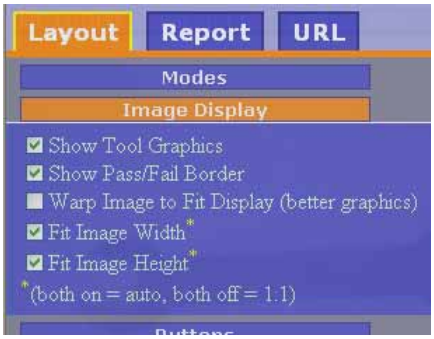
The Layout page controls many layout features, which are organized into groups. Selecting the Modes group results in the following options being displayed:



✓ Launch in Auto Mode
☐ Launch in Failures Mode

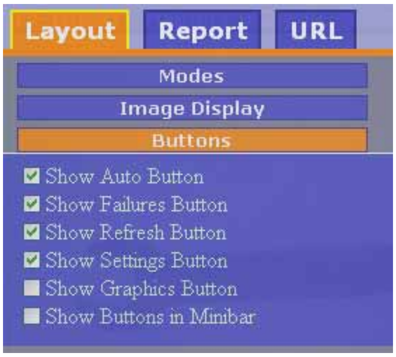
Refer to the table below for a description of the various settings. The other groups of settings appear as follows:

Image Display:



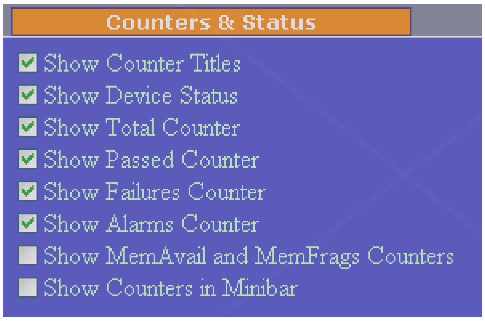
✓ Show Tool Graphics
✓ Show Pass/Fail Border
☐ Warp Image to Fit Display (better graphics)
✓ Fit Image Width*
✓ Fit Image Height*
*(both on = auto, both off = 1:1)

Buttons:



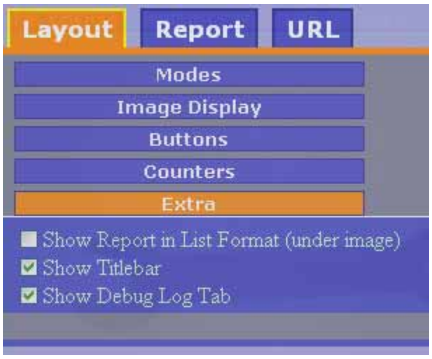
- [✓] Show Auto Button
- [✓] Show Failures Button
- [✓] Show Refresh Button
- [✓] Show Settings Button
- [ ] Show Graphics Button
- [ ] Show Buttons in Minibar

Counters and Status:



- [✓] Show Counter Titles
- [✓] Show Device Status
- [✓] Show Total Counter
- [✓] Show Passed Counter
- [✓] Show Failures Counter
- [✓] Show Alarms Counter
- [ ] Show MemAvail and MemFrags Counters
- [ ] Show Counters in Minibar

Extra Settings:



- [ ] Show Report in List Format (under image)
- [✓] Show Titlebar
- [✓] Show Debug Log Tab

As each option is checked or unchecked, the effect can be seen immediately by observing the layout of the Runtime Page shown behind the Options Setup Page.

Option Name	Function	Default
Launch in Auto Mode	Determines if the Runtime Page defaults to be in auto-refresh mode when launched.	On
Launch in Failures Mode	Determines if the Runtime Page defaults to be in show failures mode when launched.	Off
Show Tool Graphics	Shows tool graphics overlayed on the image. (NOTE: not all tool graphics are supported)	On
Show Pass/Fail Border	Shows a border around the image (green = pass, red = fail)	On
Warp Image to Fit Display	Scaling the image to fill the display area can have an adverse effect on the quality of the graphics displayed. As an example, lines can be missing. This setting improves the quality of the displayed graphics. Turning this off will reduce the overhead on the VS-06	On
Fit Image Height Fit Image Width	These two settings determine how the image is scaled to fit the display area. If both are off, then no scaling is performed and the image is displayed 1:1. If both are on, then auto scaling is performed, fitting the width or height depending on which fits the display area better. Otherwise, the image is scaled either to fit the width or height.	Auto (both on)
Show Report	Shows the report configured using the Report Setup page.	On
Show Report in List Format	If on, then the report is shown in tabular form below the image. If off, then the report is overlayed on top of the image.	Off
Show Titlebar	If on, the titlebar is shown	On
Show Buttons in Minibar	If on, the buttons are shown in the Minibar area, which appears under the titlebar. If off, the buttons will be	Off

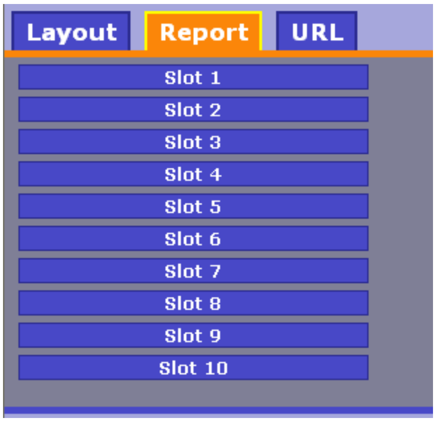
	shown to the right of the image area.	
Show Counters in Minibar	If on, the counters are shown in the Minibar area, which appears under the titlebar. If off, the counters will be shown to the right of the image area.	Off
Show Auto Button	If on, the Auto button is shown	On
Show Failures Button	If on, the Failures button is shown	On
Show Refresh Button	If on, the Refresh button is shown	On
Show Options Button	If on, the Options button is shown. This button, when pressed, displays the Options Setup screen.	Off
Show Graphics Button	If on, the Graphics button is shown. This button controls if the graphics are overlayed on the image	Off
Show Counter Titles	If on, a title is displayed to the left of each counter.	On
Show Device Status	If on, the device status (RUNNING, STOPPED) is displayed	On
Show Total Counter	If on, the total counter is shown	On
Show Passed Counter	If on, the passed counter is shown	On
Show Failures Counter	If on, the failures counter is shown	On
Show Alarms Counter	If on, the alarms counter is shown	On
Show MemAvail and MemFrags Counters	If on, two counters are shown which display memory use status for the VS-06.	Off

Clicking the Save button will save these settings so that they become the default behavior the next time the page is launched.

Clicking the Defaults button will reset the stored settings to the original defaults the next time the page is launched.

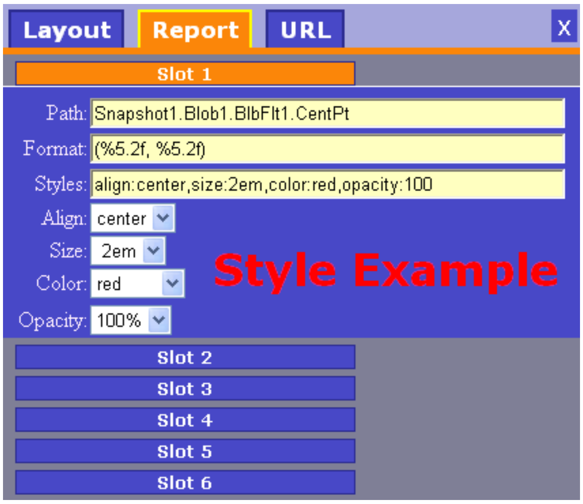
The Close button (“X” in upper right corner) will return to the main Runtime Page.

The Report Tab brings up the following Report Setup screen:



Data Values from datums in the selected inspection can be formatted and overlaid on the displayed image or shown in a table below the image. This is specified by assigning one of 10 data report slots. If overlayed on the image, each of these slots will represent a row in the display area, which is evenly split into 10 equal sized rows. The spacing will depend on the overall size of the display area, which in turn is dependant on the dimensions of the browser window. If the report is shown in list form, each slot corresponds to one of 10 rows.

Selecting a slot to configure results in the following display:



Path: Snapshot1.Blob1.BlbFlt1.CentPt
Format: (%5.2f, %5.2f)
Styles: align:center,size:2em,color:red,opacity:100
Align: center
Size: 2em
Color: red
Opacity: 100%
Style Example

At a minimum the path to a datum must be specified. The inspection is implied, so it is not in the path. In the example above, the path Snapshot1.Blob1.BlbFlt1.CentPt is specified in the first slot.

This would display the value overlaid over the image near the top of the image display area. If D5 had been used instead, it would appear closer to the center.

By default, the displayed format will be appropriate for the datum type requested. However the format can be changed by specifying a printf style format string.

The format codes must be consistent with the expected data types. If the result is an integer, then a %d format is expected, floating point numbers require %f type formats. The list of format codes is not documented here, refer to printf documentation.

For array values, each element of the array will be passed in turn to the format string. For example, if a PointDm is being used, there are four expected array values corresponding to X, Y, angle, scale. (The order is the same as for variant access via VB). An example of using a format for PointDm:

(%.2f,%.2f) angle=%.1f scale=%.1f

This will display a result similar to:

(23.23,45.10) angle=3.2 scale=1.0

The later array values can be considered optional and can be omitted if desired. For example, to display just the x and y values of a PointDm, use the format string:

center = (%6.2f, %6.2f)

This will display a result similar to:

center = (134.22, 452.12)

If no format string is specified, an appropriate default format is used. For example, for a LineDm, by default the datum value will be displayed as:

A = value, B = value, C = value

Style

The default display of a report value is left justified, and uses a default font and color. If desired, all visual aspects of the displayed report value can be modified. If the Style field is used, it has the format:

style:value,style:value,…

For example, set the text size to 9pt, and align to the right, the following can be specified:

size:9pt,align:right

Possible style values:

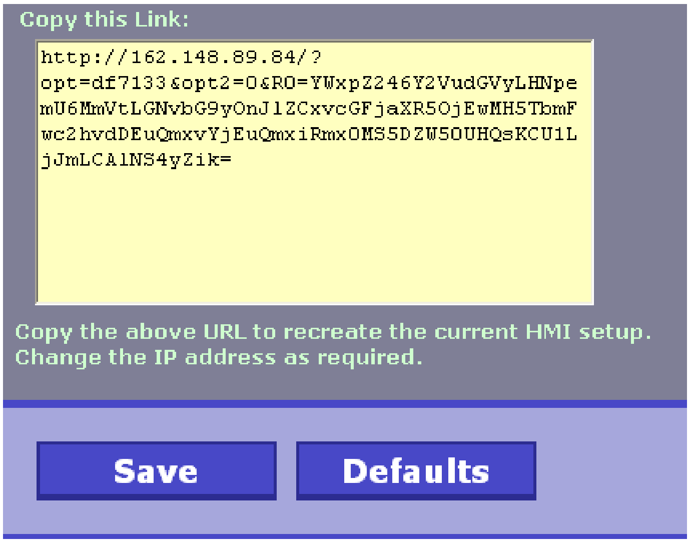
Style Name	Values	Default	Comments
align	left, right, center	left	
size	CSS text size values (examples: 3em, 9pt, 22px)	9pt	
color	any named HTML color (red, blue, etc) or hexadecimal HTML color code (FF0000=red)	yellow	
opacity	number range 0..100	100	Setting this number to less than 100 will cause the displayed text to be translucent
CSS identifier	CSS values		

It is permitted to use CSS identifiers to alter other display aspects. For example, the following will show a red background color for the text:

backgroundColor:red

To set some of the more common styles, the combo boxes for Style, Size, Color, and Opacity can be used. The styles field will automatically be updated.

Selecting the URL tab brings up the following display:



```
http://162.148.89.84/?
opt=df7133&opt2=0&RO=YWxpZ246Y2VudGVyLHNpe
mU6MmVtLGNvbG9yOnJlZCxvcGFjaXR5OjEwMH5TbmF
wc2hvdDEuQmxvYjEuQmxiRmx0MS5DZW50UHQsKCU1L
jJmLCA1NS4yZik=
```

Copy the above URL to recreate the current HMI setup.
Change the IP address as required.

The displayed URL can be copied and then used in a browser window to completely replicate the current setup.

Additional Notes:

- Line breaks can be inserted into format strings by using embedded HTML codes. To introduce a line break, use “<br />”
- Commands and options are case sensitive. This is a limitation of javascript and CSS.
- A new Frontrunner feature allows copying a path of a datum to the clipboard. Right-click the Datum name in the DatumGrid display, and select the “Copy path to clipboard” option.

APPENDIX E

Allen-Bradley PLC Setup via EDS

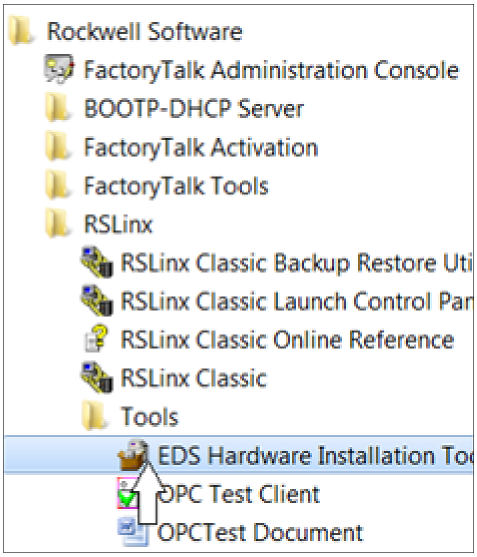
This section describes how to set up an Allen-Bradley PLC via EDS file for use with the VS-06.

AB Rockwell RSLogix 5000 v20 PLC Integration with EDS

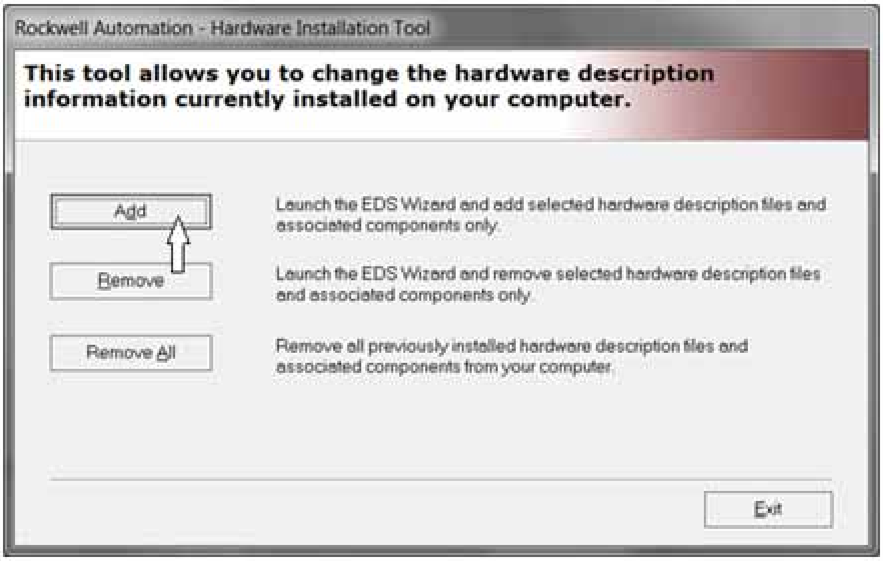
This section was created and run on the following Allen Bradley/Rockwell components:

- RSLogix 5000 Version 20.00.00 (CPR 9 SR 5)
- 756-L61 ControlLogix5561 Controller, firmware rev 20.11
- 1756-ENBT/A EtherNet/IP interface card, firmware rev 4.1

Run the Rockwell “EDS Hardware Installation Tool”.


Rockwell Software
FactoryTalk Administration Console
BOOTP-DHCP Server
FactoryTalk Activation
FactoryTalk Tools
RSLinx
RSLinx Classic Backup Restore Uti
RSLinx Classic Launch Control Par
RSLinx Classic Online Reference
RSLinx Classic
Tools
EDS Hardware Installation To
OPC Test Client
OPCTest Document

Select Add:


Rockwell Automation - Hardware Installation Tool

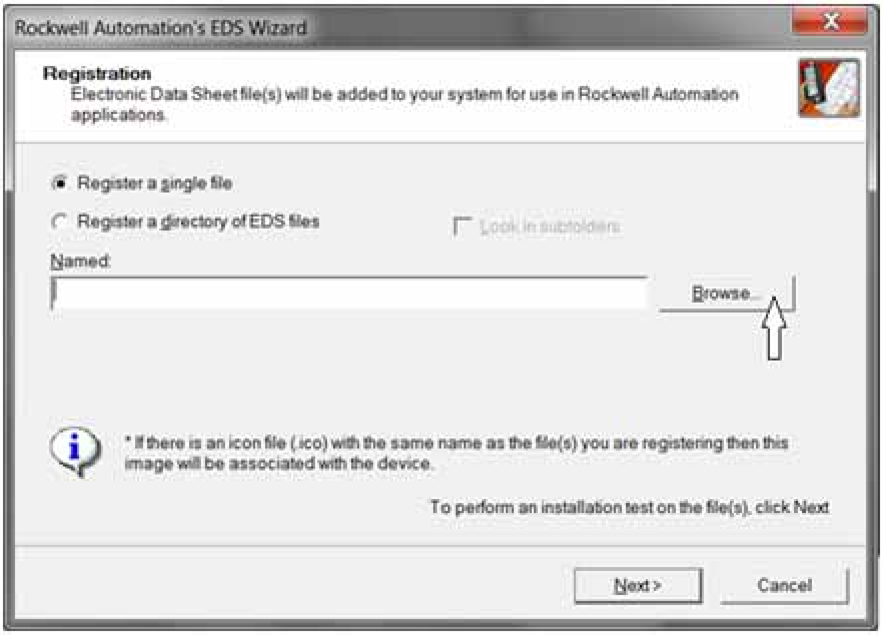
This tool allows you to change the hardware description information currently installed on your computer.

Launch the EDS Wizard and add selected hardware description files and associated components only.

Launch the EDS Wizard and remove selected hardware description files and associated components only.

Remove all previously installed hardware description files and associated components from your computer.

Select Browse:


Rockwell Automation's EDS Wizard

Registration
Electronic Data Sheet file(s) will be added to your system for use in Rockwell Automation applications.

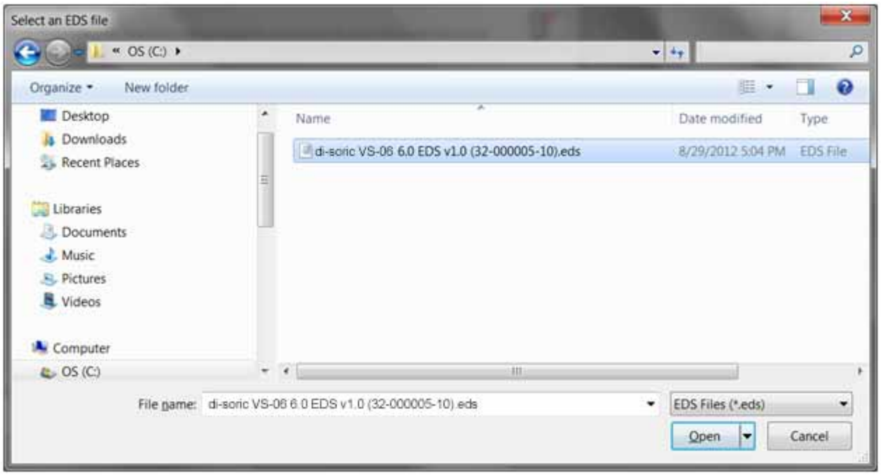
(●) Register a single file
( ) Register a directory of EDS files
[ ] Look in subfolders

Named:

* If there is an icon file (.ico) with the same name as the file(s) you are registering then this image will be associated with the device.

To perform an installation test on the file(s), click Next

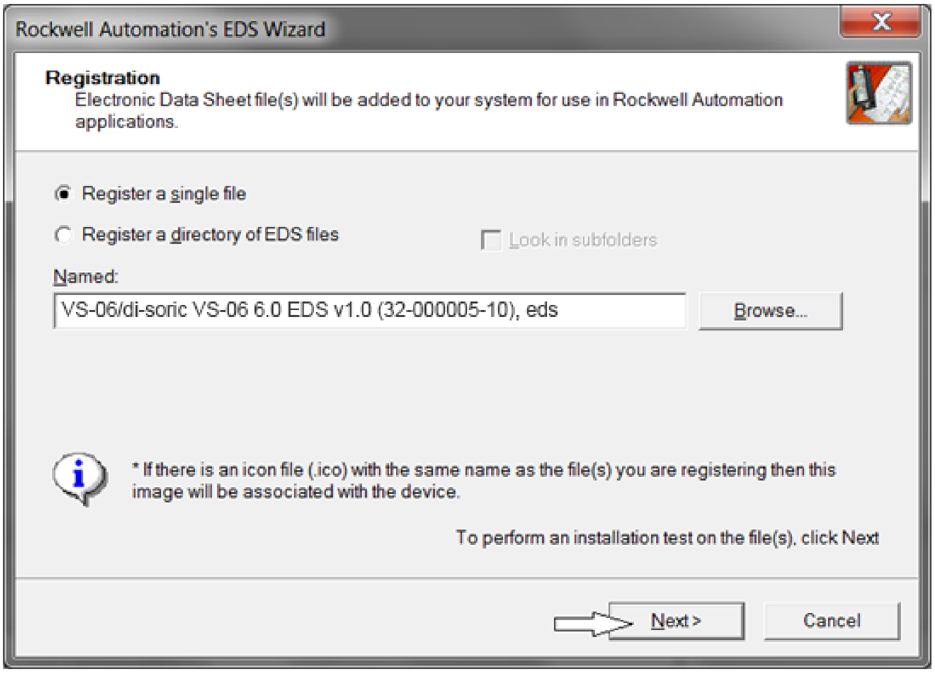
Navigate to the VS-06 EDS file, then Open it. The default install location is C:\di-soric\Vscape\Firmware\eds\VS-06.



Name	Date modified	Type
di-soric VS-06 6.0 EDS v1.0 (32-000005-10).eds	8/29/2012 5:04 PM	EDS File

File name: di-soric VS-06 6.0 EDS v1.0 (32-000005-10).eds

Keep clicking Next > until the Finish button is displayed:



Registration
Electronic Data Sheet file(s) will be added to your system for use in Rockwell Automation applications.

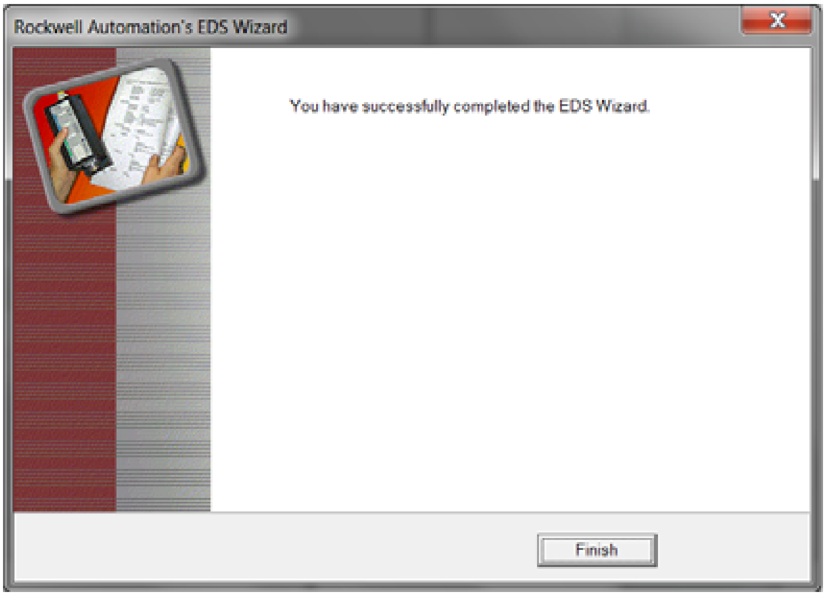
● Register a single file
○ Register a directory of EDS files
☐ Look in subfolders

Named:
VS-06/di-soric VS-06 6.0 EDS v1.0 (32-000005-10), eds

* If there is an icon file (.ico) with the same name as the file(s) you are registering then this image will be associated with the device.

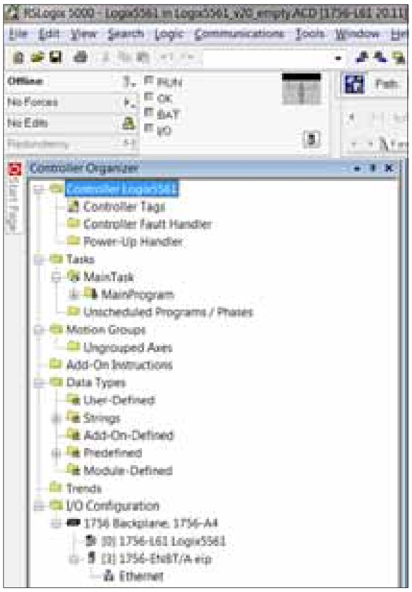
To perform an installation test on the file(s), click Next

Click Finish:

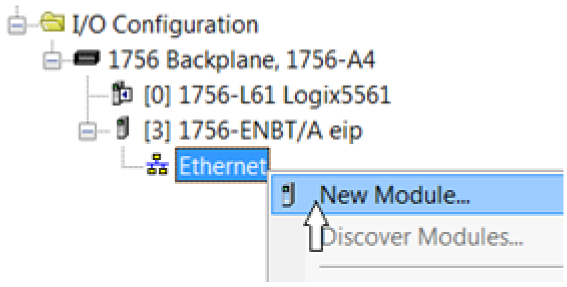


You have successfully completed the EDS Wizard.

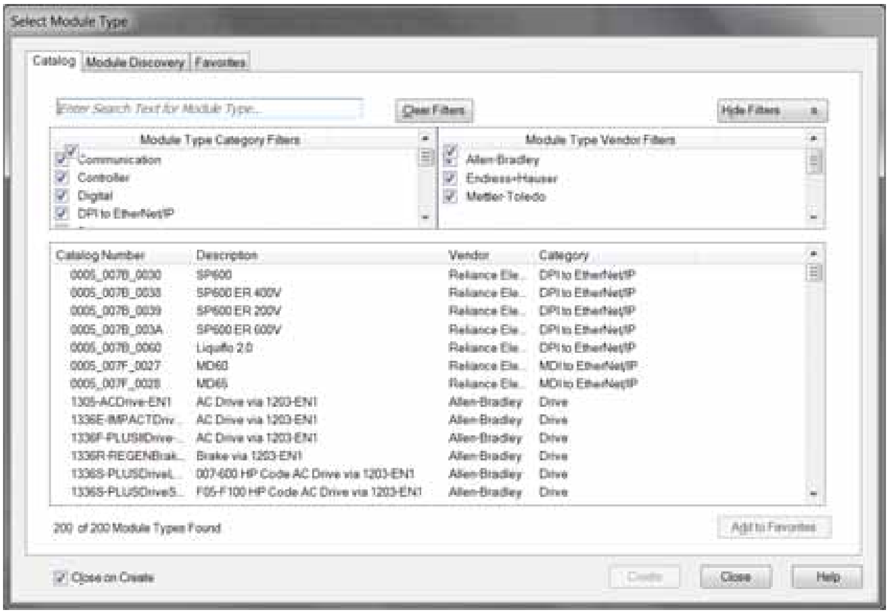
Open RSLogix 5000 v20 and create the I/O Configuration for the base system, including the system's Ethernet interface:



Right click on Ethernet and select New Module:



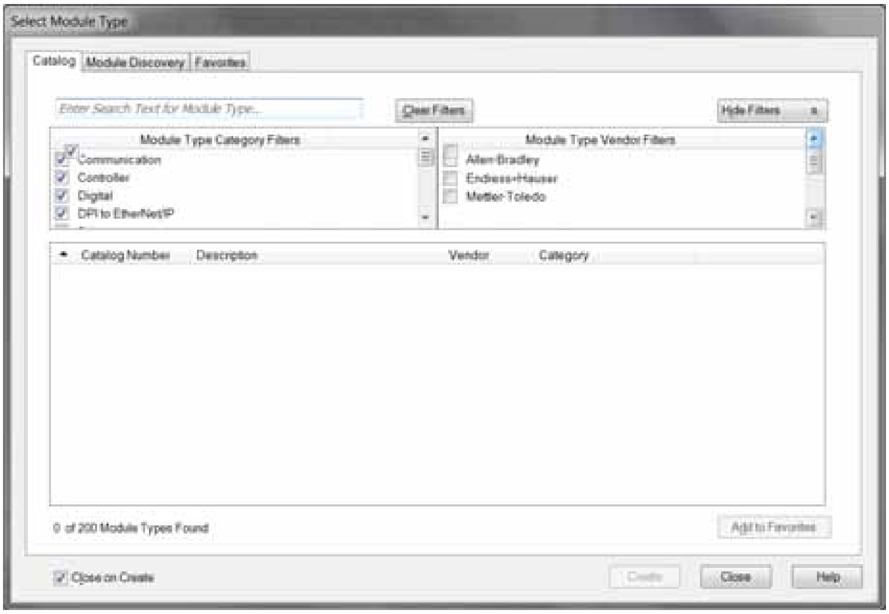
The Select Module Type dialog is displayed:



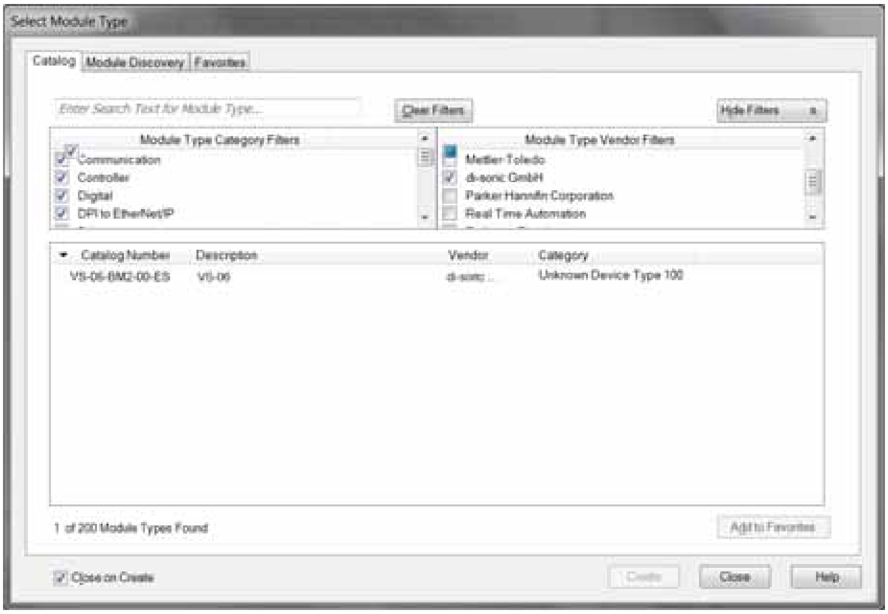
Catalog Number	Description	Vendor	Category
0005_007B_0030	SP600	Reliance Ele...	DPI to EtherNet/IP
0005_007B_0038	SP600 ER 400V	Reliance Ele...	DPI to EtherNet/IP
0005_007B_0039	SP600 ER 200V	Reliance Ele...	DPI to EtherNet/IP
0005_007B_003A	SP600 ER 600V	Reliance Ele...	DPI to EtherNet/IP
0005_007B_0060	Liquiflo 2.0	Reliance Ele...	DPI to EtherNet/IP
0005_007F_0027	MD60	Reliance Ele...	MDI to EtherNet/IP
0005_007F_0028	MD65	Reliance Ele...	MDI to EtherNet/IP
1305-ACDrive-EN1	AC Drive via 1203-EN1	Allen-Bradley	Drive
1336E-IMPACTDriv...	AC Drive via 1203-EN1	Allen-Bradley	Drive
1336F-PLUSIIDrive-...	AC Drive via 1203-EN1	Allen-Bradley	Drive
1336R-REGENBrak...	Brake via 1203-EN1	Allen-Bradley	Drive
1336S-PLUSDriveL...	007-600 HP Code AC Drive via 1203-EN1	Allen-Bradley	Drive
1336S-PLUSDriveS...	F05-F100 HP Code AC Drive via 1203-EN1	Allen-Bradley	Drive

200 of 200 Module Types Found

Clear the Module Type Vendor Filters:



Scroll down the Module Type Vendor Filters until di-soric comes into view, then select di-soric:



Select Module Type

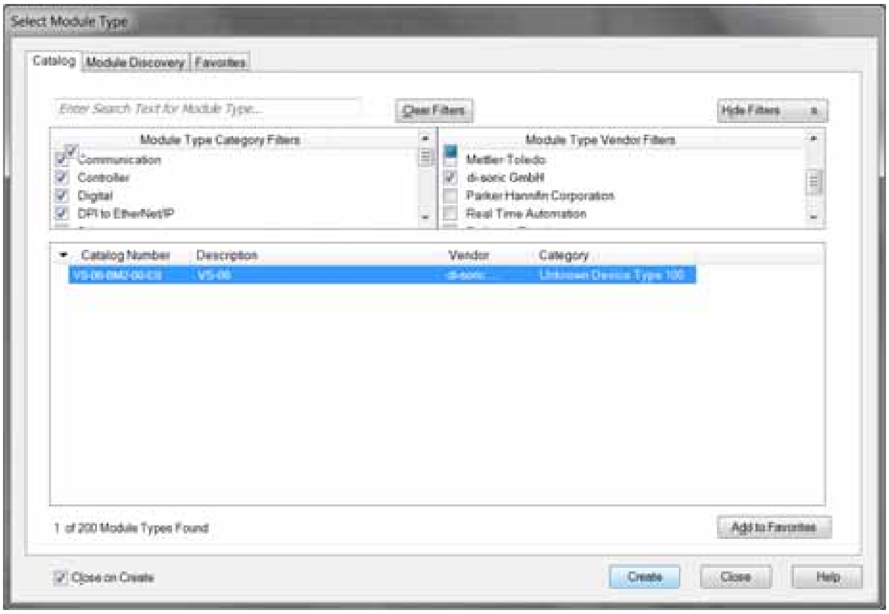
Module Type Category Filters
✓ Communication
✓ Controller
✓ Digital
✓ DPI to EtherNet/IP

Module Type Vendor Filters
Mettler-Toledo
✓ di-soric GmbH
Parker Hannifin Corporation
Real Time Automation

Catalog Number	Description	Vendor	Category
VS-06-BM2-00-ES	VS-06	di-soric ...	Unknown Device Type 100

1 of 200 Module Types Found

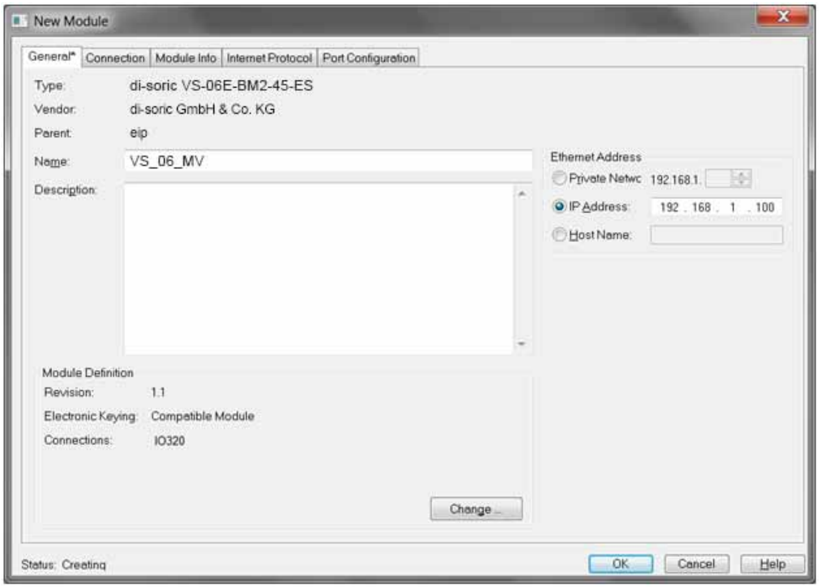
Click the required camera and select Create:



Catalog Number	Description	Vendor	Category
VS-06-BM2-00-ES	VS-06	di-soric ...	Unknown Device Type 100

1 of 200 Module Types Found

The **New Module** dialog is displayed. Type a unique name for this camera and its IP address:



Type:	di-soric VS-06E-BM2-45-ES
Vendor:	di-soric GmbH & Co. KG
Parent:	eip
Name:	VS_06_MV
Description:	

Ethernet Address

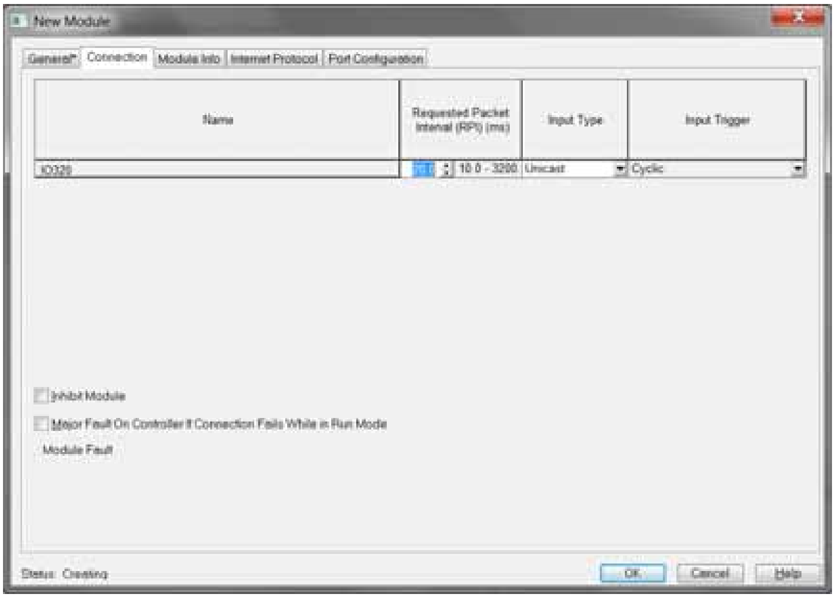
- Private Netwc 192.168.1.
- [✓ IP Address:] 192 . 168 . 1 . 100
- Host Name:

Module Definition

Revision:	1.1
Electronic Keying:	Compatible Module
Connections:	IO320

Status: Creating

Go to the Connection tab and set the Requested Packet Interval required for the application:



New Module
General* | Connection | Module Info | Internet Protocol | Port Configuration

Name	Requested Packet Interval (RPI) (ms)		Input Type	Input Trigger
IO320	10.0	10.0 - 3200	Unicast	Cyclic

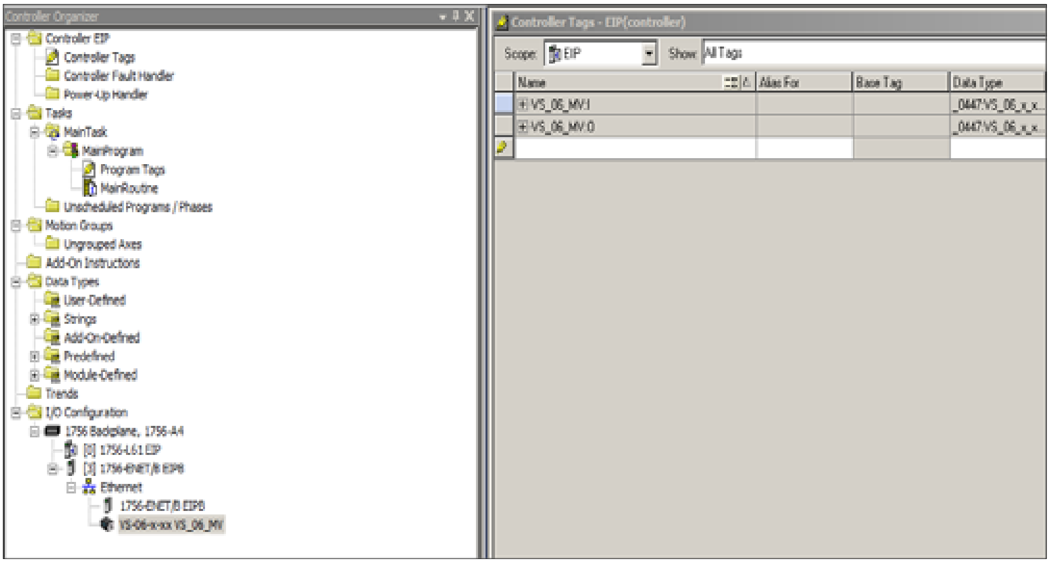
☐ Inhibit Module

☐ Major Fault On Controller If Connection Fails While in Run Mode

Module Fault

Status: Creating

Click OK, verify the camera was added to the Ethernet network, then open the Controller Tags to verify that :I and :O tag sets were created:



Controller Organizer

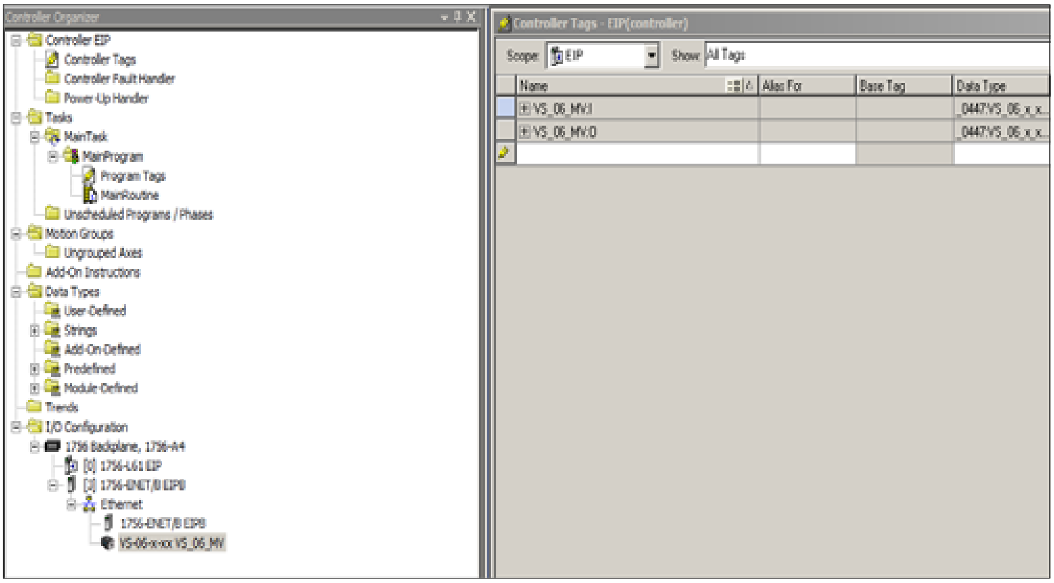
- Controller EIP
  - Controller Tags
  - Controller Fault Handler
  - Power-Up Handler
- Tasks
  - MainTask
    - MainProgram
      - Program Tags
      - MainRoutine
  - Unscheduled Programs / Phases
- Motion Groups
  - Ungrouped Axes
- Add-On Instructions
- Data Types
  - User-Defined
  - Strings
  - Add-On-Defined
  - Predefined
  - Module-Defined
- Trends
- I/O Configuration
  - 1756 Backplane, 1756-A4
    - [0] 1756-L61 EIP
    - [3] 1756-ENET/B EIPB
      - Ethernet
        - 1756-ENET/B EIPB
        - VS-06-x-xx VS_06_MV

Controller Tags - EIP(controller)

Scope: EIP Show: All Tags

Name	Alias For	Base Tag	Data Type
VS_06_MV:I			0447:VS_06_x_x...
VS_06_MV:O			0447:VS_06_x_x...

Open the Main Routine:



Controller Organizer

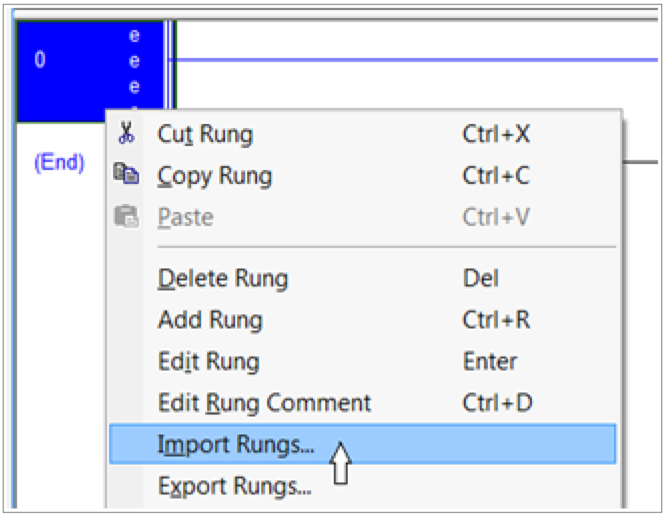
- Controller EIP
  - Controller Tags
  - Controller Fault Handler
  - Power-Up Handler
- Tasks
  - MainTask
    - MainProgram
      - Program Tags
      - MainRoutine
  - Unscheduled Programs / Phases
- Motion Groups
  - Ungrouped Axes
- Add-On Instructions
- Data Types
  - User-Defined
  - Strings
  - Add-On-Defined
  - Predefined
  - Module-Defined
- Trends
- I/O Configuration
  - 1756 Backplane, 1756-A4
    - [0] 1756-L61 EIP
    - [3] 1756-ENET/B EIPB
      - Ethernet
        - 1756-ENET/B EIPB
        - VS-06-x-xx VS_06_MV

Controller Tags - EIP(controller)

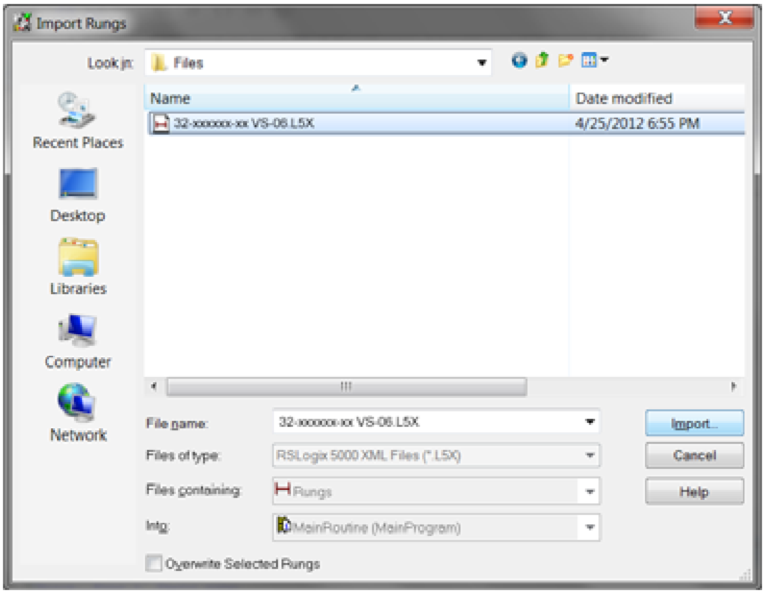
Scope: EIP Show: All Tags

Name	Alias For	Base Tag	Data Type
VS_06_MV:I			0447:VS_06_x_x...
VS_06_MV:O			0447:VS_06_x_x...

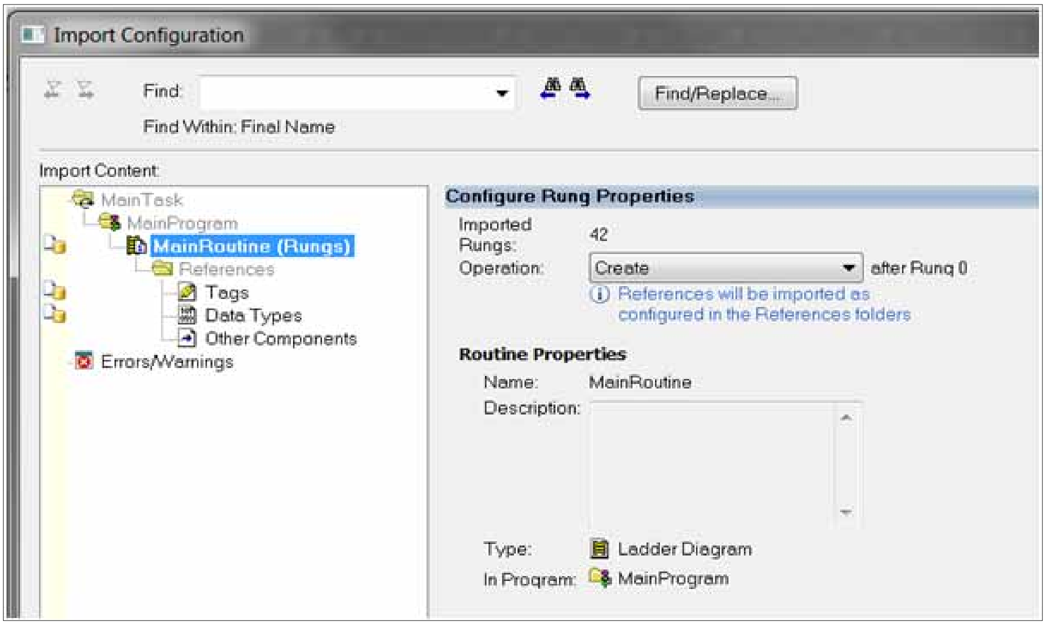
Right-click rung 0, and select Import Rungs:



Navigate to the VS-06 32-000003-lx.L5X file and select Import. The default install directory is C:\di-soric\Vscape\Tutorials and Samples\VS-06\EIP Demo.



The Import Configuration dialog is displayed:



Import Configuration

Find:

Find Within: Final Name

Find/Replace...

Import Content:

- MainTask
  - MainProgram
    - MainRoutine (Rungs)
      - References
        - Tags
        - Data Types
        - Other Components
- Errors/Warnings

Configure Rung Properties

Imported Rungs: 42

Operation: Create after Rung 0

References will be imported as configured in the References folders

Routine Properties

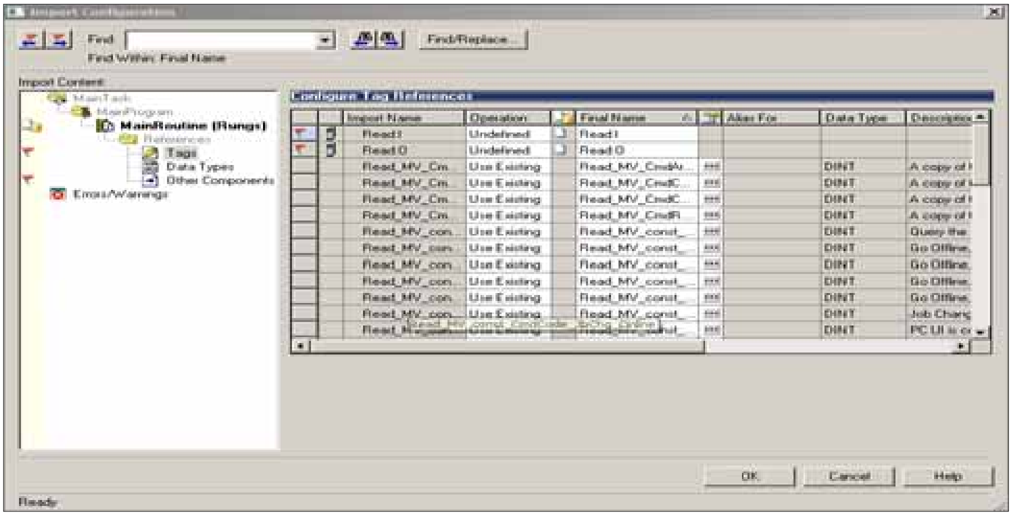
Name: MainRoutine

Description:

Type: Ladder Diagram

In Program: MainProgram

Select Tags:



Import Configuration

Find:

Find Within: Final Name

Find/Replace...

Import Content:

- MainTask
  - MainProgram
    - MainRoutine (Rungs)
      - References
        - Tags
        - Data Types
        - Other Components
- Errors/Warnings

Configure Tag References

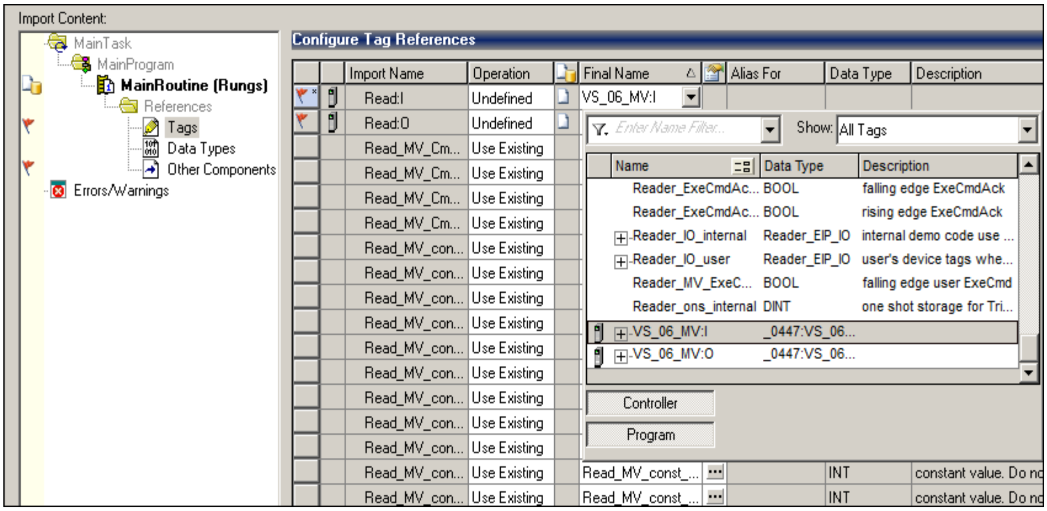
Import Name	Operation	Final Name	Alias For	Data Type	Description
Read I	Undefined	Read I			
Read O	Undefined	Read O			
Read_MV_Cm	Use Existing	Read_MV_CmdA		DINT	A copy of
Read_MV_Cm	Use Existing	Read_MV_CmdC		DINT	A copy of
Read_MV_Cm	Use Existing	Read_MV_CmdC		DINT	A copy of
Read_MV_Cm	Use Existing	Read_MV_CmdR		DINT	A copy of
Read_MV_con	Use Existing	Read_MV_const_		DINT	Query the
Read_MV_con	Use Existing	Read_MV_const_		DINT	Go Offline,
Read_MV_con	Use Existing	Read_MV_const_		DINT	Go Offline,
Read_MV_con	Use Existing	Read_MV_const_		DINT	Go Offline,
Read_MV_con	Use Existing	Read_MV_const_		DINT	Go Offline,
Read_MV_con	Use Existing	Read_MV_const_		DINT	Job Chang
Read_M[illegible]	[illegible]	[illegible]		DINT	PC UI is c

Read_MV_const_CmdCode_JbChg_Online

OK Cancel Help

Ready

In the Final Name column, click on VS-06:I, then click on the down arrow that shows up on the right:



Import Content:

- MainTask
  - MainProgram
    - MainRoutine (Rungs)
      - References
        - Tags
        - Data Types
        - Other Components
- Errors/Warnings

Configure Tag References

Import Name	Operation	Final Name	Alias For	Data Type	Description
Read:I	Undefined	VS_06_MV:I			
Read:O	Undefined				
Read_MV_Cm...	Use Existing				
Read_MV_Cm...	Use Existing				
Read_MV_Cm...	Use Existing				
Read_MV_Cm...	Use Existing				
Read_MV_con...	Use Existing				
Read_MV_con...	Use Existing				
Read_MV_con...	Use Existing				
Read_MV_con...	Use Existing				
Read_MV_con...	Use Existing				
Read_MV_con...	Use Existing				
Read_MV_con...	Use Existing				
Read_MV_con...	Use Existing				
Read_MV_con...	Use Existing				
Read_MV_con...	Use Existing	Read_MV_const_...		INT	constant value. Do no
Read_MV_con...	Use Existing	Read_MV_const_...		INT	constant value. Do no

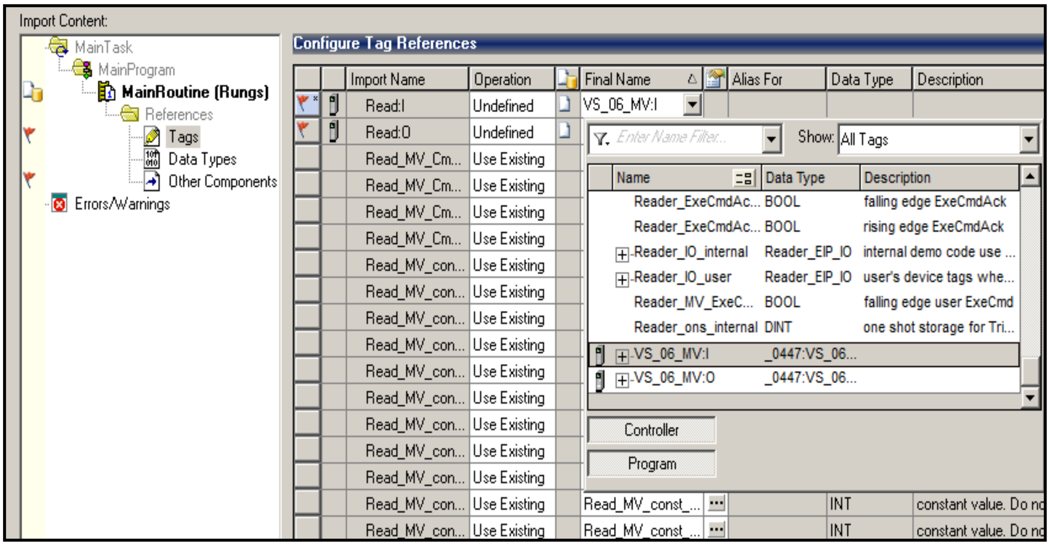
Enter Name Filter... Show: All Tags

Name	Data Type	Description
Reader_ExeCmdAc...	BOOL	falling edge ExeCmdAck
Reader_ExeCmdAc...	BOOL	rising edge ExeCmdAck
+Reader_IO_internal	Reader_EIP_IO	internal demo code use ...
+Reader_IO_user	Reader_EIP_IO	user's device tags whe...
Reader_MV_ExeC...	BOOL	falling edge user ExeCmd
Reader_ons_internal	DINT	one shot storage for Tri...
+VS_06_MV:I	_0447:VS_06...	
+VS_06_MV:O	_0447:VS_06...	

Controller

Program

Double-click on the Read MV:I (or whatever name that was assigned to the device) tag:



Import Content:

- MainTask
  - MainProgram
    - MainRoutine (Rungs)
      - References
        - Tags
        - Data Types
        - Other Components
- Errors/Warnings

Configure Tag References

Import Name	Operation	Final Name	Alias For	Data Type	Description
Read:I	Undefined	VS_06_MV:I			
Read:O	Undefined				
Read_MV_Cm...	Use Existing				
Read_MV_Cm...	Use Existing				
Read_MV_Cm...	Use Existing				
Read_MV_Cm...	Use Existing				
Read_MV_con...	Use Existing				
Read_MV_con...	Use Existing				
Read_MV_con...	Use Existing				
Read_MV_con...	Use Existing				
Read_MV_con...	Use Existing				
Read_MV_con...	Use Existing				
Read_MV_con...	Use Existing				
Read_MV_con...	Use Existing				
Read_MV_con...	Use Existing				
Read_MV_con...	Use Existing	Read_MV_const_...		INT	constant value. Do no
Read_MV_con...	Use Existing	Read_MV_const_...		INT	constant value. Do no

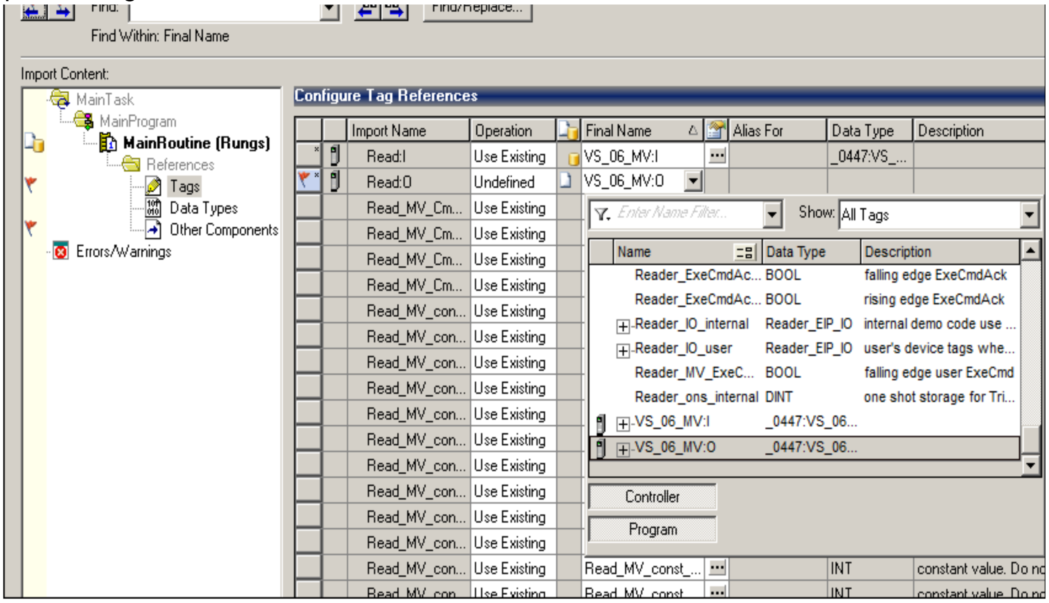
Enter Name Filter... Show: All Tags

Name	Data Type	Description
Reader_ExeCmdAc...	BOOL	falling edge ExeCmdAck
Reader_ExeCmdAc...	BOOL	rising edge ExeCmdAck
+Reader_IO_internal	Reader_EIP_IO	internal demo code use ...
+Reader_IO_user	Reader_EIP_IO	user's device tags whe...
Reader_MV_ExeC...	BOOL	falling edge user ExeCmd
Reader_ons_internal	DINT	one shot storage for Tri...
+VS_06_MV:I	_0447:VS_06...	
+VS_06_MV:O	_0447:VS_06...	

Controller

Program

In the “Final Name” column, click on Read_MV:O, then click on the down arrow that shows up on the right:



Find:

Find Within: Final Name

Import Content:

- MainTask
  - MainProgram
    - MainRoutine (Rungs)
      - References
        - Tags
        - Data Types
        - Other Components
- Errors/Warnings

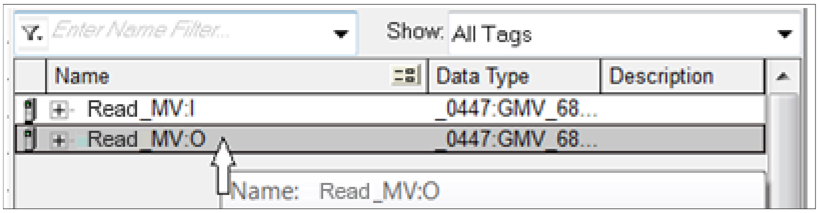
Configure Tag References

Import Name	Operation	Final Name	Alias For	Data Type	Description
Read:I	Use Existing	VS_06_MV:I		_0447:VS_...	
Read:O	Undefined	VS_06_MV:O			
Read_MV_Cm...	Use Existing				
Read_MV_Cm...	Use Existing				
Read_MV_Cm...	Use Existing				
Read_MV_Cm...	Use Existing				
Read_MV_con...	Use Existing				
Read_MV_con...	Use Existing				
Read_MV_con...	Use Existing				
Read_MV_con...	Use Existing				
Read_MV_con...	Use Existing				
Read_MV_con...	Use Existing				
Read_MV_con...	Use Existing				
Read_MV_con...	Use Existing				
Read_MV_con...	Use Existing				
Read_MV_con...	Use Existing				
Read_MV_con...	Use Existing	Read_MV_const_...		INT	constant value. Do nc
Read_MV_con	Use Existing	Read_MV_const		INT	constant value. Do nc

Enter Name Filter... Show: All Tags

Name	Data Type	Description
Reader_ExeCmdAc...	BOOL	falling edge ExeCmdAck
Reader_ExeCmdAc...	BOOL	rising edge ExeCmdAck
Reader_IO_internal	Reader_EIP_IO	internal demo code use ...
Reader_IO_user	Reader_EIP_IO	user's device tags whe...
Reader_MV_ExeC...	BOOL	falling edge user ExeCmd
Reader_ons_internal	DINT	one shot storage for Tri...
VS_06_MV:I	_0447:VS_06...	
VS_06_MV:O	_0447:VS_06...	

Double click on the Read_MV:O (or whatever name that was assigned to the device) tag:

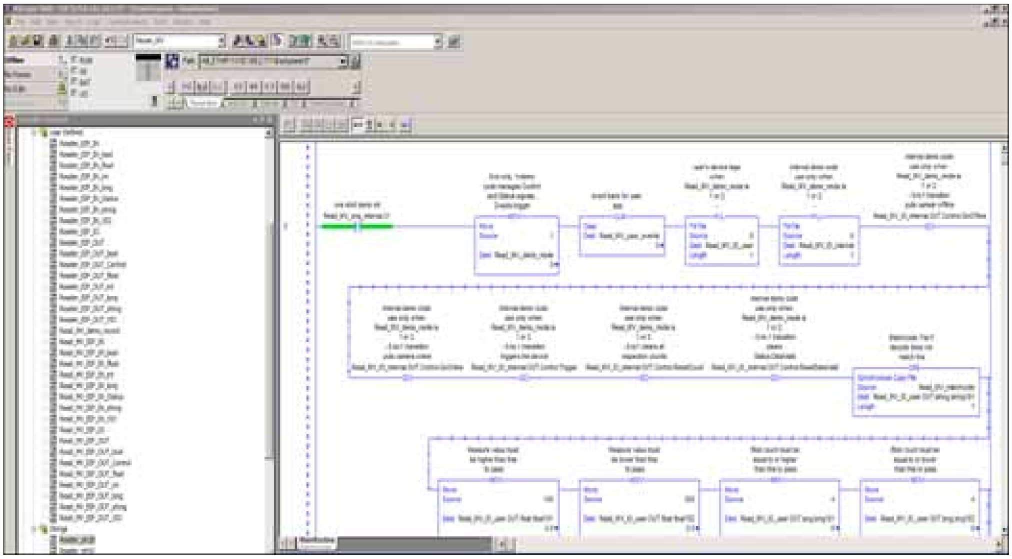


Enter Name Filter... Show: All Tags

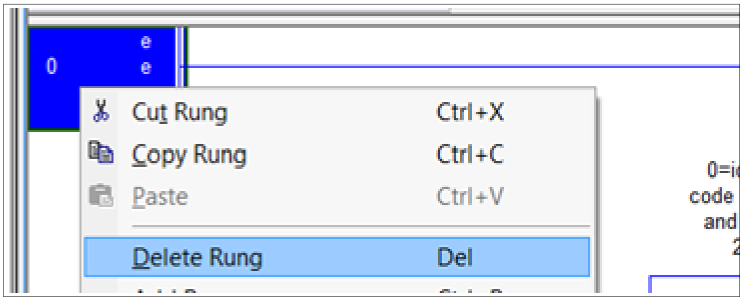
Name	Data Type	Description
Read_MV:I	_0447:GMV_68...	
Read_MV:O	_0447:GMV_68...	

Name: Read_MV:O

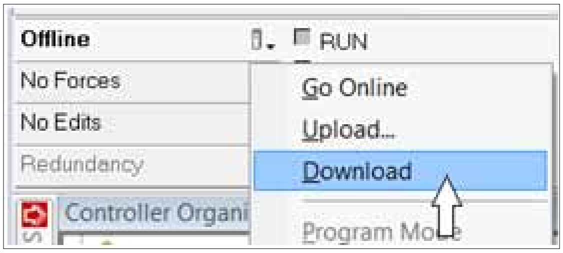
Click **OK** and the Main Routine and User Defined tags will be populated:



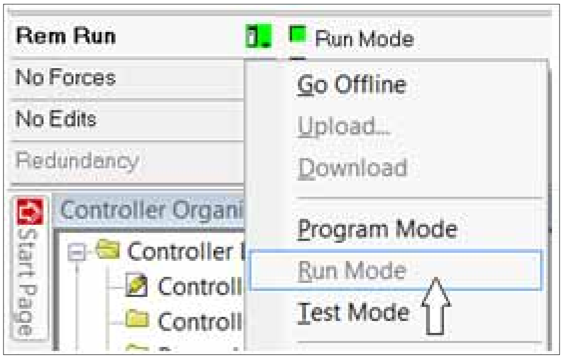
Delete any empty rungs (check rung 0):



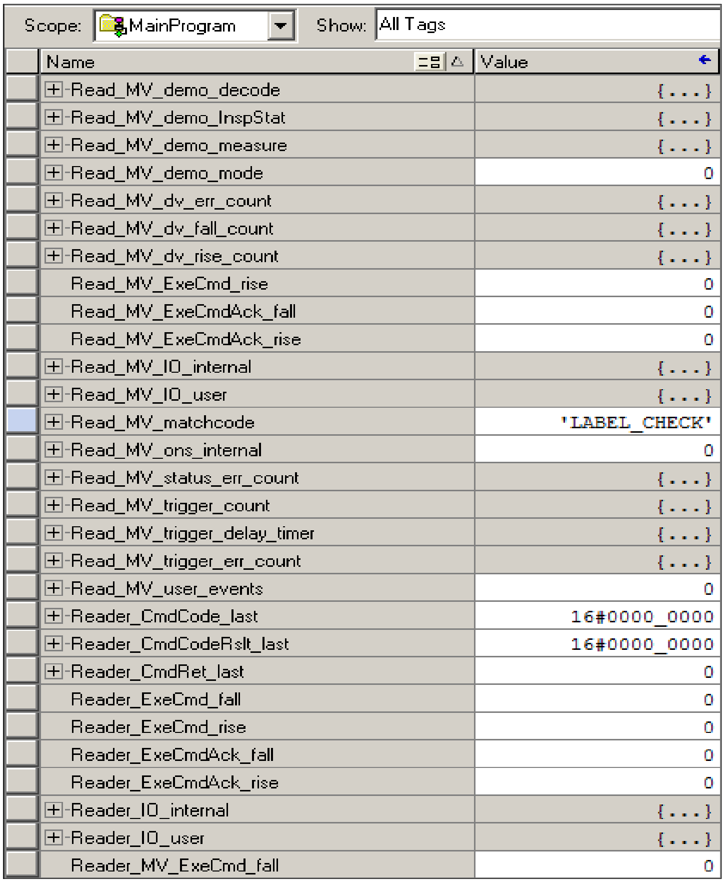
Download the project to the PLC:



Put the PLC into Run Mode:



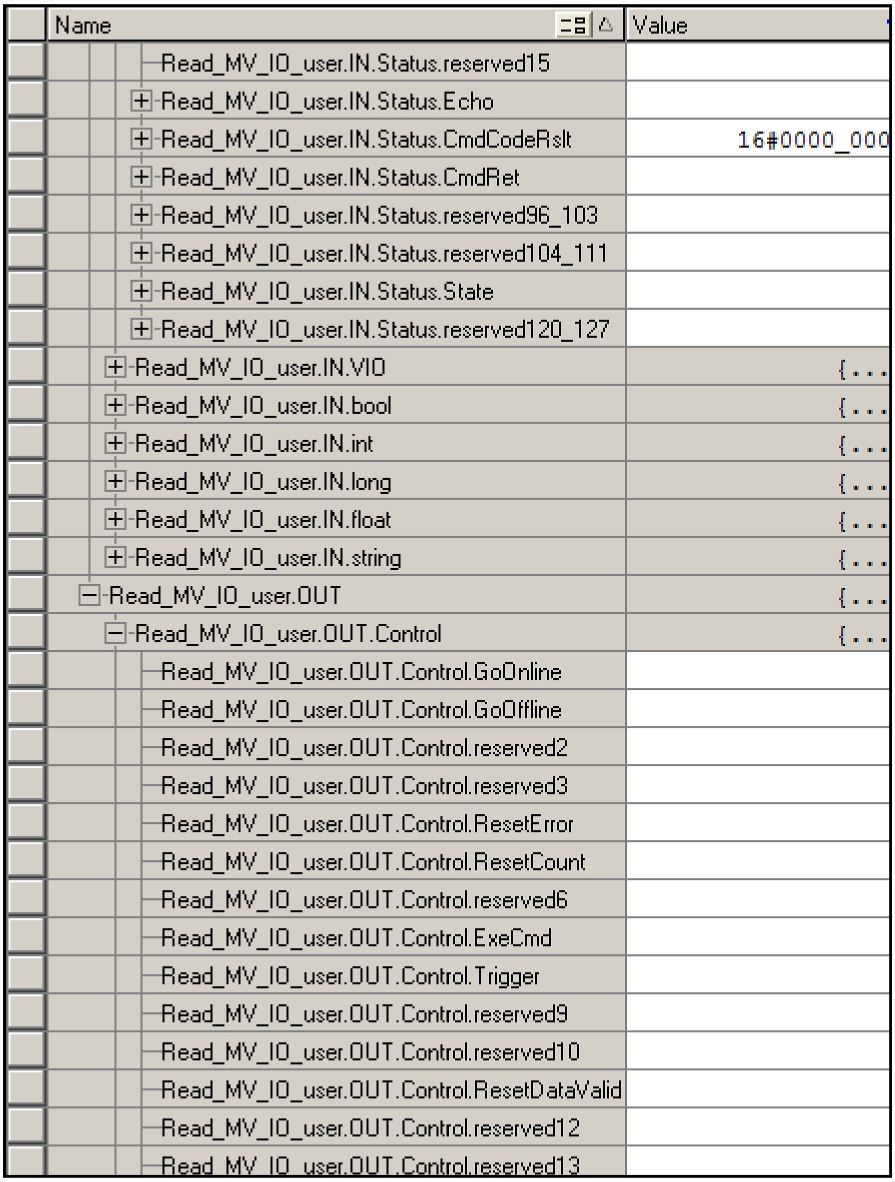
Open the Program Tag window and select “Monitor Tags”:



Scope: MainProgram   Show: All Tags

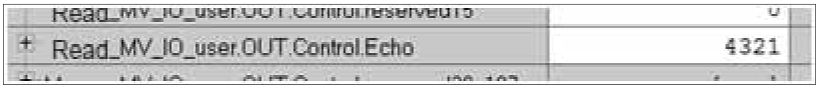
Name	Value
⊞-Read_MV_demo_decode	{...}
⊞-Read_MV_demo_InspStat	{...}
⊞-Read_MV_demo_measure	{...}
⊞-Read_MV_demo_mode	0
⊞-Read_MV_dv_err_count	{...}
⊞-Read_MV_dv_fall_count	{...}
⊞-Read_MV_dv_rise_count	{...}
Read_MV_ExeCmd_rise	0
Read_MV_ExeCmdAck_fall	0
Read_MV_ExeCmdAck_rise	0
⊞-Read_MV_IO_internal	{...}
⊞-Read_MV_IO_user	{...}
⊞-Read_MV_matchcode	'LABEL_CHECK'
⊞-Read_MV_ons_internal	0
⊞-Read_MV_status_err_count	{...}
⊞-Read_MV_trigger_count	{...}
⊞-Read_MV_trigger_delay_timer	{...}
⊞-Read_MV_trigger_err_count	{...}
⊞-Read_MV_user_events	0
⊞-Reader_CmdCode_last	16#0000_0000
⊞-Reader_CmdCodeRslt_last	16#0000_0000
⊞-Reader_CmdRet_last	0
Reader_ExeCmd_fall	0
Reader_ExeCmd_rise	0
Reader_ExeCmdAck_fall	0
Reader_ExeCmdAck_rise	0
⊞-Reader_IO_internal	{...}
⊞-Reader_IO_user	{...}
Reader_MV_ExeCmd_fall	0

Expand Read_MV_IO_user so that the Echo in the .IN.Status and .OUT.Control structures are visible:



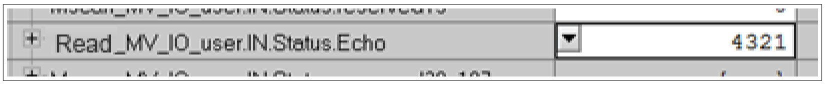
Name	Value
Read_MV_IO_user.IN.Status.reserved15	
+ Read_MV_IO_user.IN.Status.Echo	
+ Read_MV_IO_user.IN.Status.CmdCodeRslt	16#0000_000
+ Read_MV_IO_user.IN.Status.CmdRet	
+ Read_MV_IO_user.IN.Status.reserved96_103	
+ Read_MV_IO_user.IN.Status.reserved104_111	
+ Read_MV_IO_user.IN.Status.State	
+ Read_MV_IO_user.IN.Status.reserved120_127	
+ Read_MV_IO_user.IN.VIO	{...
+ Read_MV_IO_user.IN.bool	{...
+ Read_MV_IO_user.IN.int	{...
+ Read_MV_IO_user.IN.long	{...
+ Read_MV_IO_user.IN.float	{...
+ Read_MV_IO_user.IN.string	{...
− Read_MV_IO_user.OUT	{...
− Read_MV_IO_user.OUT.Control	{...
Read_MV_IO_user.OUT.Control.GoOnline	
Read_MV_IO_user.OUT.Control.GoOffline	
Read_MV_IO_user.OUT.Control.reserved2	
Read_MV_IO_user.OUT.Control.reserved3	
Read_MV_IO_user.OUT.Control.ResetError	
Read_MV_IO_user.OUT.Control.ResetCount	
Read_MV_IO_user.OUT.Control.reserved6	
Read_MV_IO_user.OUT.Control.ExeCmd	
Read_MV_IO_user.OUT.Control.Trigger	
Read_MV_IO_user.OUT.Control.reserved9	
Read_MV_IO_user.OUT.Control.reserved10	
Read_MV_IO_user.OUT.Control.ResetDataValid	
Read_MV_IO_user.OUT.Control.reserved12	
Read_MV_IO_user.OUT.Control.reserved13	

Change .OUT.Control.Echo to non-zero:



Read_MV_IO_user.OUT.Control.Echo 4321

Verify Read_MV_IO_user.IO.IN.Status.Echo is the same value as the .OUT.Control.Echo:



Read_MV_IO_user.IN.Status.Echo 4321

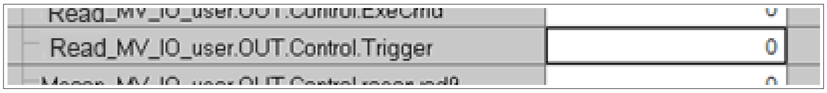
This confirms that the PLC and camera have successful two-way communication.

The demo code expects a demo vision job to be loaded on the camera, which populates the following input tags (camera to PLC) with vision tool results:

- .IN.bool.bool1, bool2, and bool3
- .IN.long.long1
- .IN.float.float1
- .IN.string.string1

The demo code will operate the Control and Status signals of the camera regardless of whatever vision job is loaded. For a more detailed overview of the demo code and vision job, please see the associated appendix **Demo PLC Code**.

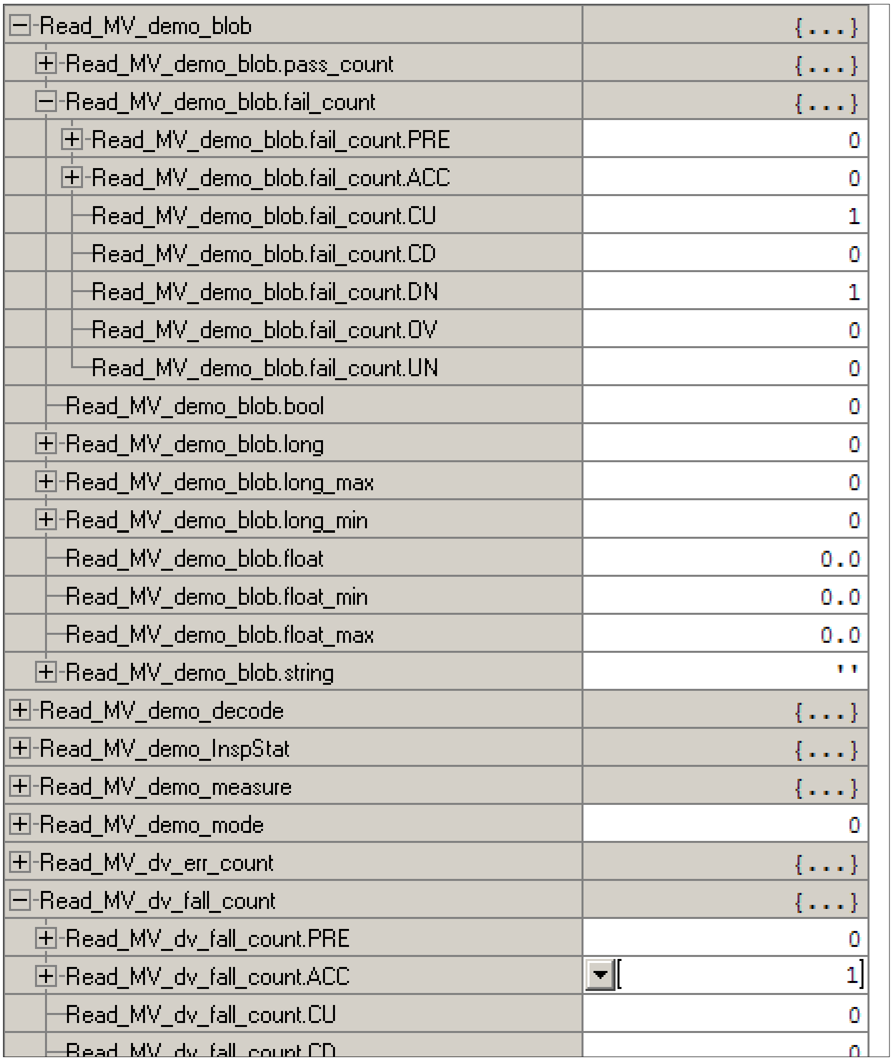
To send a trigger to the camera, scroll to Read_MV_IO_user.Control.Trigger:



Read_MV_IO_user.OUT.Control.Trigger 0

Set the Trigger to 1. This causes the demo code to trigger the camera, process the new inspection data, record the results in the Read_MV_demo_xxxx tags, and clear the DataValid status signal.

The user of the demo code can know that the camera was triggered when the Trigger control changes to 0. All processing is done when the counter Read_MV_dv_fall_count increments, along with the pass/fail counters in the Read_MV_demo_xxxx tags. For example:



⊟ Read_MV_demo_blob	{...}
⊞ Read_MV_demo_blob.pass_count	{...}
⊟ Read_MV_demo_blob.fail_count	{...}
⊞ Read_MV_demo_blob.fail_count.PRE	0
⊞ Read_MV_demo_blob.fail_count.ACC	0
Read_MV_demo_blob.fail_count.CU	1
Read_MV_demo_blob.fail_count.CD	0
Read_MV_demo_blob.fail_count.DN	1
Read_MV_demo_blob.fail_count.OV	0
Read_MV_demo_blob.fail_count.UN	0
Read_MV_demo_blob.bool	0
⊞ Read_MV_demo_blob.long	0
⊞ Read_MV_demo_blob.long_max	0
⊞ Read_MV_demo_blob.long_min	0
Read_MV_demo_blob.float	0.0
Read_MV_demo_blob.float_min	0.0
Read_MV_demo_blob.float_max	0.0
⊞ Read_MV_demo_blob.string	''
⊞ Read_MV_demo_decode	{...}
⊞ Read_MV_demo_InspStat	{...}
⊞ Read_MV_demo_measure	{...}
⊞ Read_MV_demo_mode	0
⊞ Read_MV_dv_err_count	{...}
⊟ Read_MV_dv_fall_count	{...}
⊞ Read_MV_dv_fall_count.PRE	0
⊞ Read_MV_dv_fall_count.ACC	1
Read_MV_dv_fall_count.CU	0
Read_MV_dv_fall_count.CD	0

APPENDIX F

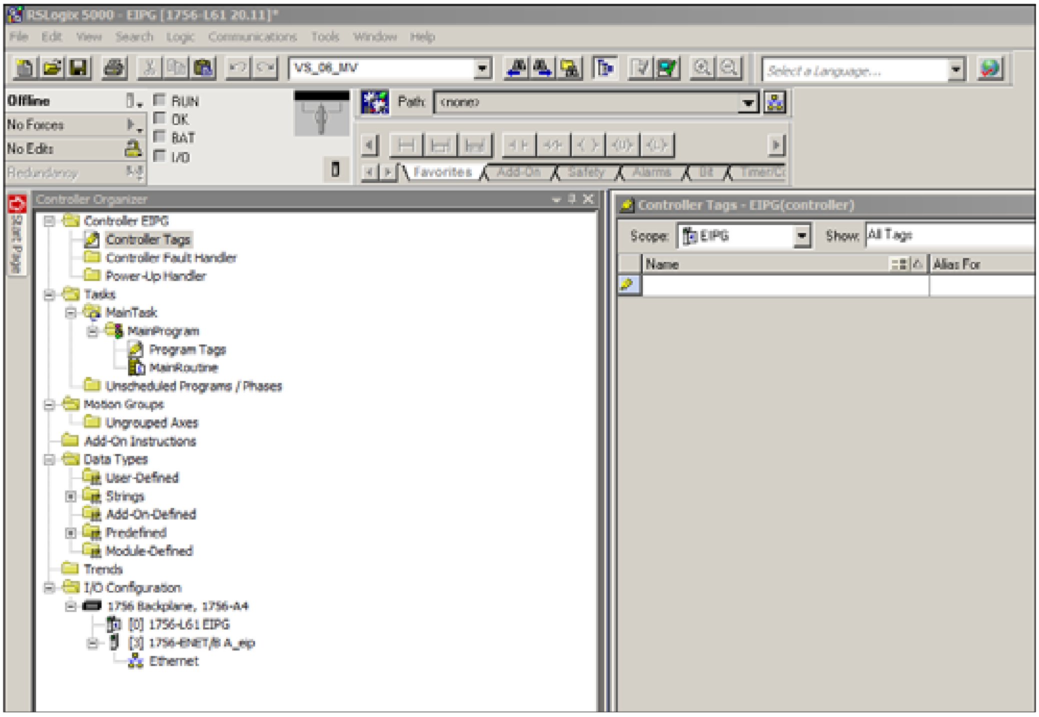
Allen-Bradley PLC Setup via Generic Ethernet Module

This section describes how to set up an Allen-Bradley PLC via Generic Ethernet Module for use with the VS-06.

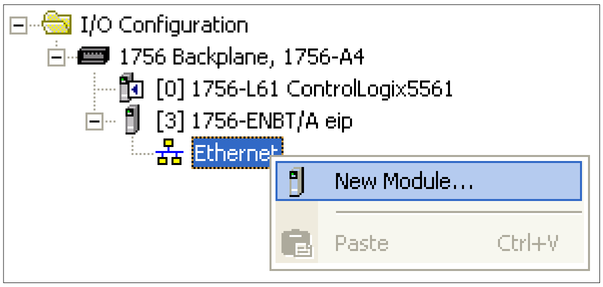
Prepare the PLC: Integrate the Camera into a PLC Environment

This section assumes you are using an Allen Bradley PLC with Rockwell RSLogix 5000 v16 or newer. RSLogix v19 and v20 may look slightly different than the screen shots shown, but the integration process is still valid.

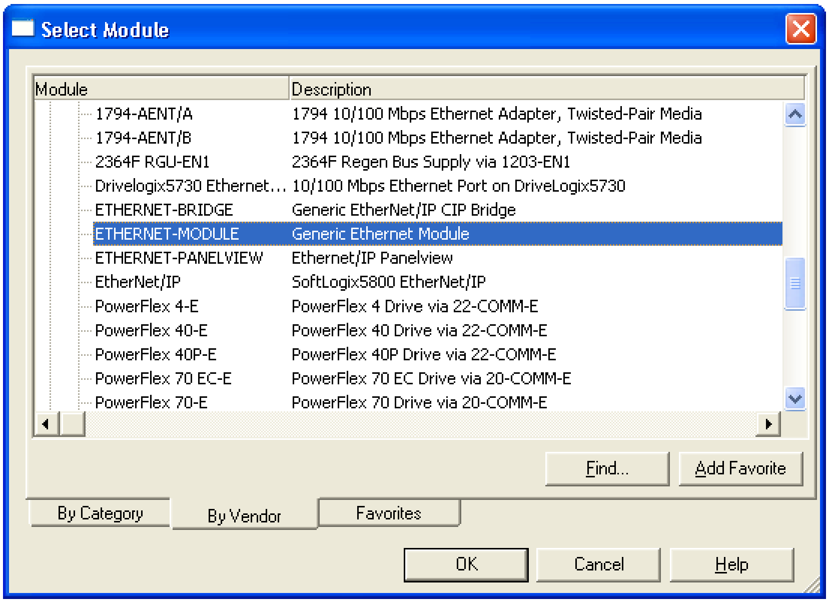
Create the I/O Configuration for the base system, including the system's Ethernet interface:



Add the camera by right-clicking on the Ethernet interface, and select “New Module”:



Select “ETHERNET-MODULE Generic Ethernet Module”, and click OK:



Module	Description
1794-AENT/A	1794 10/100 Mbps Ethernet Adapter, Twisted-Pair Media
1794-AENT/B	1794 10/100 Mbps Ethernet Adapter, Twisted-Pair Media
2364F RGU-EN1	2364F Regen Bus Supply via 1203-EN1
Drivelogix5730 Ethernet...	10/100 Mbps Ethernet Port on DriveLogix5730
ETHERNET-BRIDGE	Generic EtherNet/IP CIP Bridge
ETHERNET-MODULE	Generic Ethernet Module
ETHERNET-PANELVIEW	Ethernet/IP Panelview
EtherNet/IP	SoftLogix5800 EtherNet/IP
PowerFlex 4-E	PowerFlex 4 Drive via 22-COMM-E
PowerFlex 40-E	PowerFlex 40 Drive via 22-COMM-E
PowerFlex 40P-E	PowerFlex 40P Drive via 22-COMM-E
PowerFlex 70 EC-E	PowerFlex 70 EC Drive via 20-COMM-E
PowerFlex 70-E	PowerFlex 70 Drive via 20-COMM-E

Configure the following fields:

“Name” = A useful name to remember the unit. The example here is “Read_MV”.

“IP Address” = The IP Address of the camera

“Comm Format” = “Data – DINT”

“Input” “Assembly Instance” = 102

“Input” “Size” = 80

“Output” “Assembly Instance” = 114

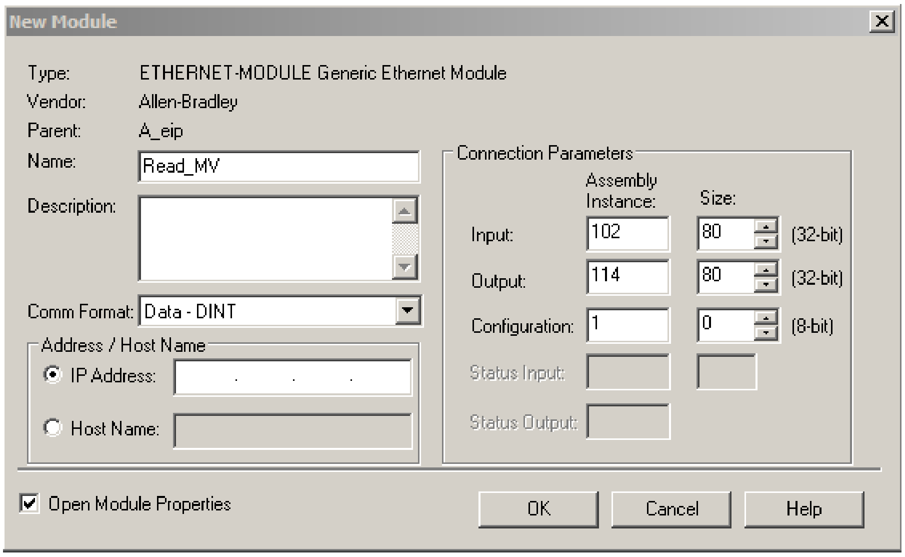
“Output” “Size” = 80

“Configuration” “Assembly Instance” = 1

“Configuration” “Size” = 0 (none)

Click OK when done.

Example:



New Module

Type: ETHERNET-MODULE Generic Ethernet Module
Vendor: Allen-Bradley
Parent: A_eip
Name: Read_MV
Description:
Comm Format: Data - DINT

Address / Host Name
● IP Address: . . .
○ Host Name:

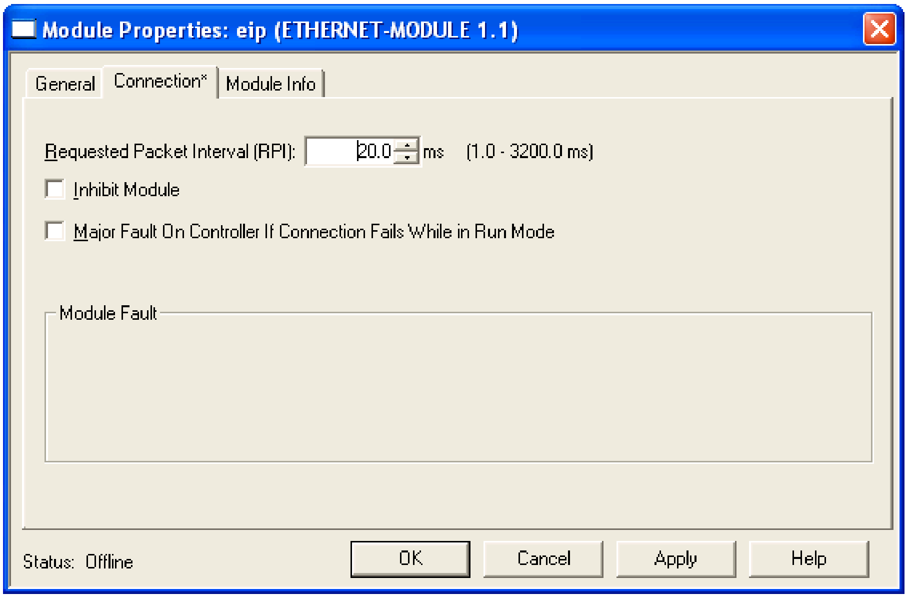
Connection Parameters

	Assembly Instance:	Size:	
Input:	102	80	(32-bit)
Output:	114	80	(32-bit)
Configuration:	1	0	(8-bit)
Status Input:			
Status Output:			

☑ Open Module Properties

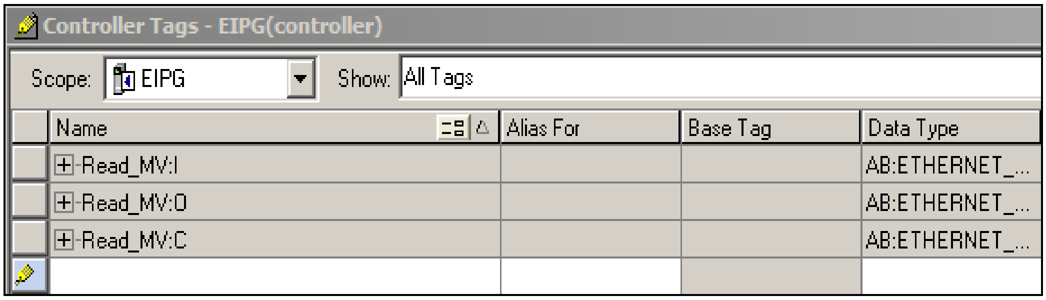
Configure the “Required Packet Interval (RPI)” and click OK.

10 ms is the minimum allowed by the camera, 20 ms or higher is recommend, as required by the application:



Requested Packet Interval (RPI): 20.0 ms (1.0 - 3200.0 ms)
☐ Inhibit Module
☐ Major Fault On Controller If Connection Fails While in Run Mode
Module Fault
Status: Offline

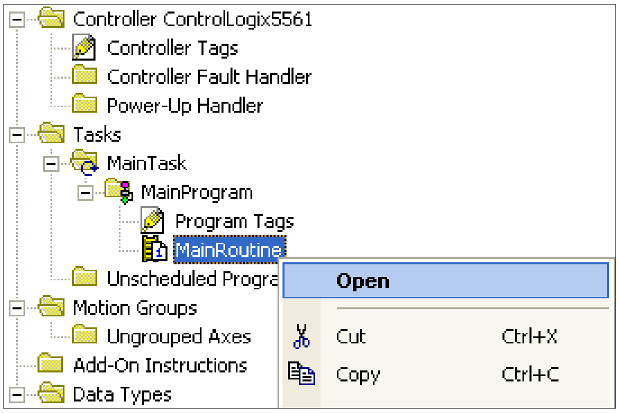
Double-click on the “Controller Tags” item, and verify VH’s :I and :O tags appear in the Controller Tags window:



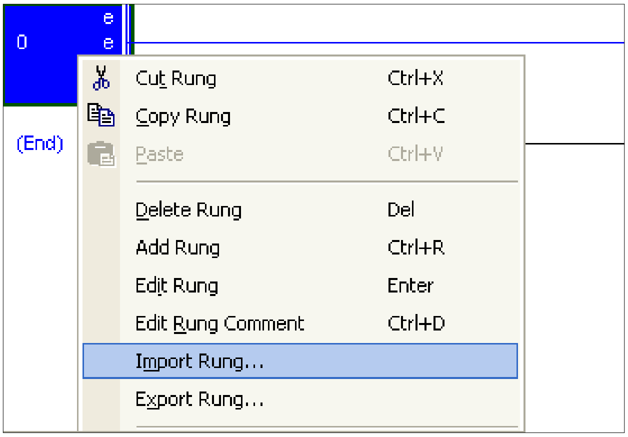
Scope: EIPG Show: All Tags

Name	Alias For	Base Tag	Data Type
+-Read_MV:I			AB:ETHERNET_...
+-Read_MV:O			AB:ETHERNET_...
+-Read_MV:C			AB:ETHERNET_...

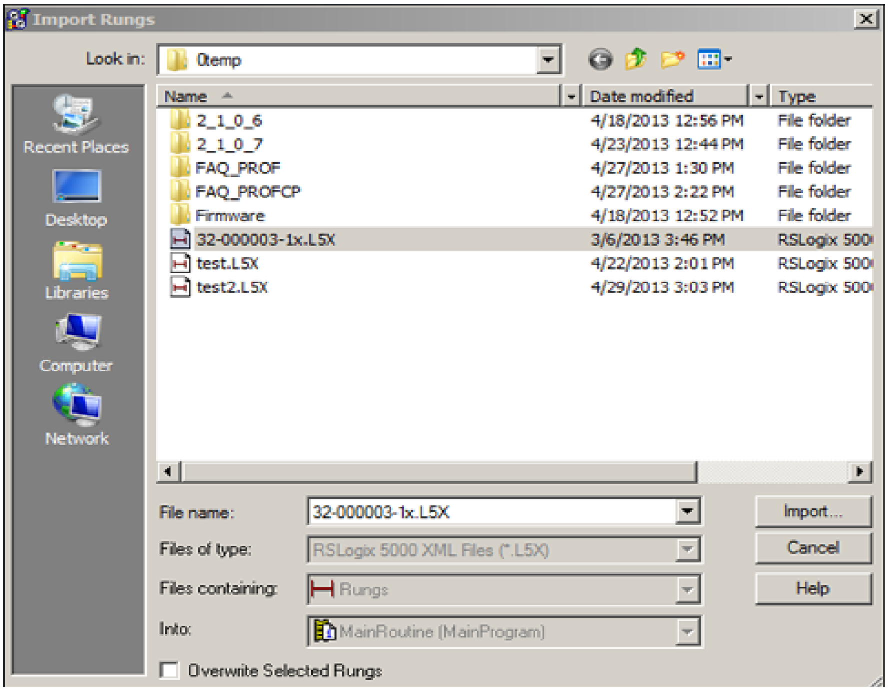
Open the “Main Routine”:



Right-click on the top rung and select “Import Rung”:



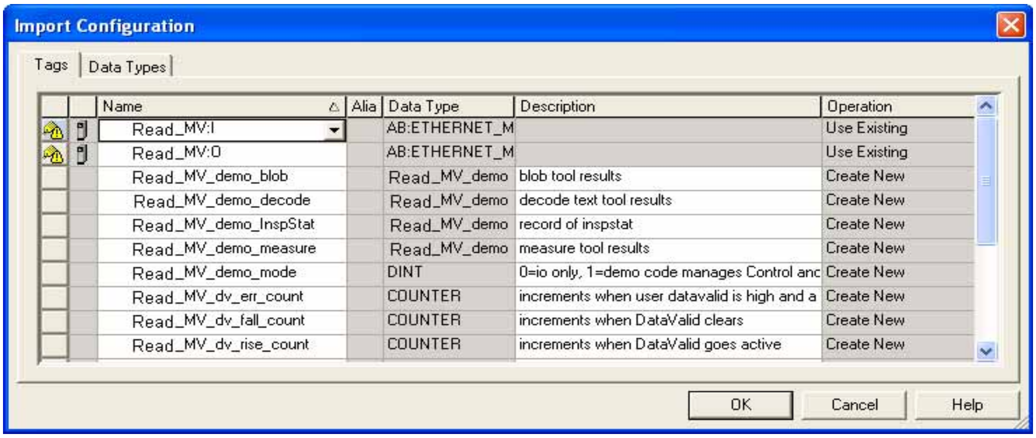
Navigieren Sie zu dem VS-06 32-000003 lx.L5X-Datei und wählen Sie Importieren. Das Standard-Installationsverzeichnis ist C: \ di-soric \ Vscape \ Tutorials und Samples \ VS-06 \ EIP Demo:



Name	Date modified	Type
2_1_0_6	4/18/2013 12:56 PM	File folder
2_1_0_7	4/23/2013 12:44 PM	File folder
FAQ_PROF	4/27/2013 1:30 PM	File folder
FAQ_PROFCP	4/27/2013 2:22 PM	File folder
Firmware	4/18/2013 12:52 PM	File folder
32-000003-1x.L5X	3/6/2013 3:46 PM	RSLogix 500
test.L5X	4/22/2013 2:01 PM	RSLogix 500
test2.L5X	4/29/2013 3:03 PM	RSLogix 500

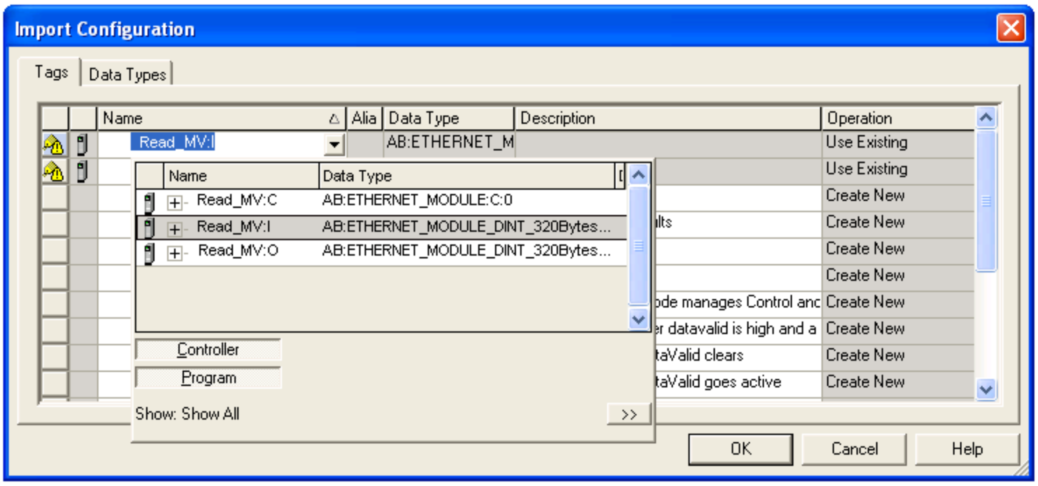
File name: 32-000003-1x.L5X
Files of type: RSLogix 5000 XML Files (*.L5X)
Files containing: Rungs
Into: MainRoutine (MainProgram)
Overwrite Selected Rungs

At the “Import Configuration” window, find the Module Name that was assigned to the Generic Module. Here the module name is “Read_MV”:



Name	Alia	Data Type	Description	Operation
Read_MV:I		AB:ETHERNET_M		Use Existing
Read_MV:O		AB:ETHERNET_M		Use Existing
Read_MV_demo_blob		Read_MV_demo	blob tool results	Create New
Read_MV_demo_decode		Read_MV_demo	decode text tool results	Create New
Read_MV_demo_InspStat		Read_MV_demo	record of inspstat	Create New
Read_MV_demo_measure		Read_MV_demo	measure tool results	Create New
Read_MV_demo_mode		DINT	0=io only, 1=demo code manages Control anc	Create New
Read_MV_dv_err_count		COUNTER	increments when user datavalid is high and a	Create New
Read_MV_dv_fall_count		COUNTER	increments when DataValid clears	Create New
Read_MV_dv_rise_count		COUNTER	increments when DataValid goes active	Create New

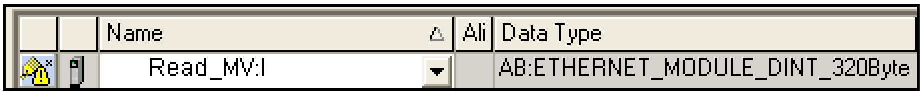
Click on Read_MV:I, then click on the down-arrow, then double click on the “Read_MV:I” that appears below it:



		Name	Alia	Data Type	Description	Operation
		Read_MV:I		AB:ETHERNET_M		Use Existing
						Use Existing
						Create New
					lts	Create New
						Create New
						Create New
					ode manages Control anc	Create New
					r datavalid is high and a	Create New
					taValid clears	Create New
					taValid goes active	Create New

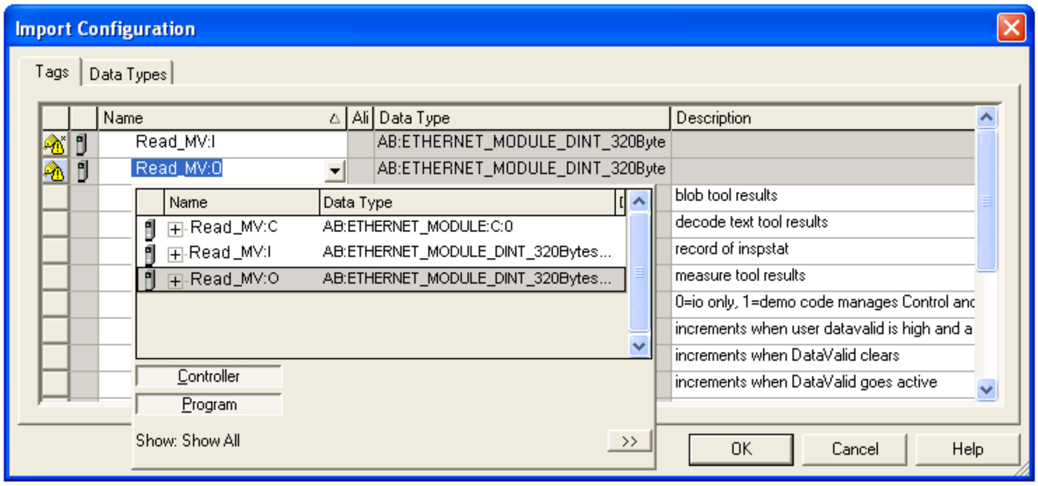
Name	Data Type
Read_MV:C	AB:ETHERNET_MODULE:C:0
Read_MV:I	AB:ETHERNET_MODULE_DINT_320Bytes...
Read_MV:O	AB:ETHERNET_MODULE_DINT_320Bytes...

Note the “*” that appears on the far left of the dialog box for the Read_MV:I line:



		Name	Ali	Data Type
		Read_MV:I		AB:ETHERNET_MODULE_DINT_320Byte

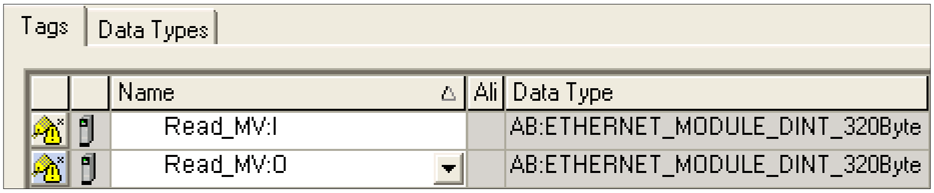
Click on Read_MV:O, then click on the down-arrow, then double click on the “Read_MV:O” that appears below it:



		Name	Ali	Data Type	Description
		Read_MV:I		AB:ETHERNET_MODULE_DINT_320Byte	
		Read_MV:O		AB:ETHERNET_MODULE_DINT_320Byte	
					blob tool results
					decode text tool results
					record of inspstat
					measure tool results
					0=io only, 1=demo code manages Control anc
					increments when user datavalid is high and a
					increments when DataValid clears
					increments when DataValid goes active

Name	Data Type
Read_MV:C	AB:ETHERNET_MODULE:C:0
Read_MV:I	AB:ETHERNET_MODULE_DINT_320Bytes...
Read_MV:O	AB:ETHERNET_MODULE_DINT_320Bytes...

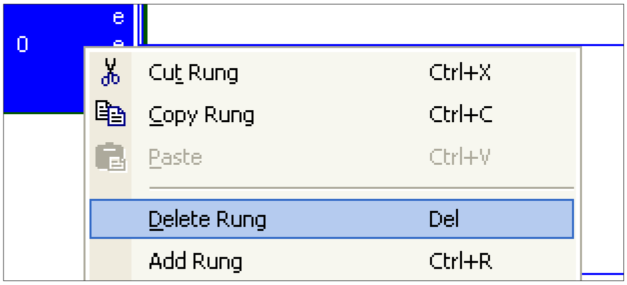
Note the "*" that appears on the far left of the dialog box for the Read_MV:O line:



		Name	Ali	Data Type
		Read_MV:I		AB:ETHERNET_MODULE_DINT_320Byte
		Read_MV:O		AB:ETHERNET_MODULE_DINT_320Byte

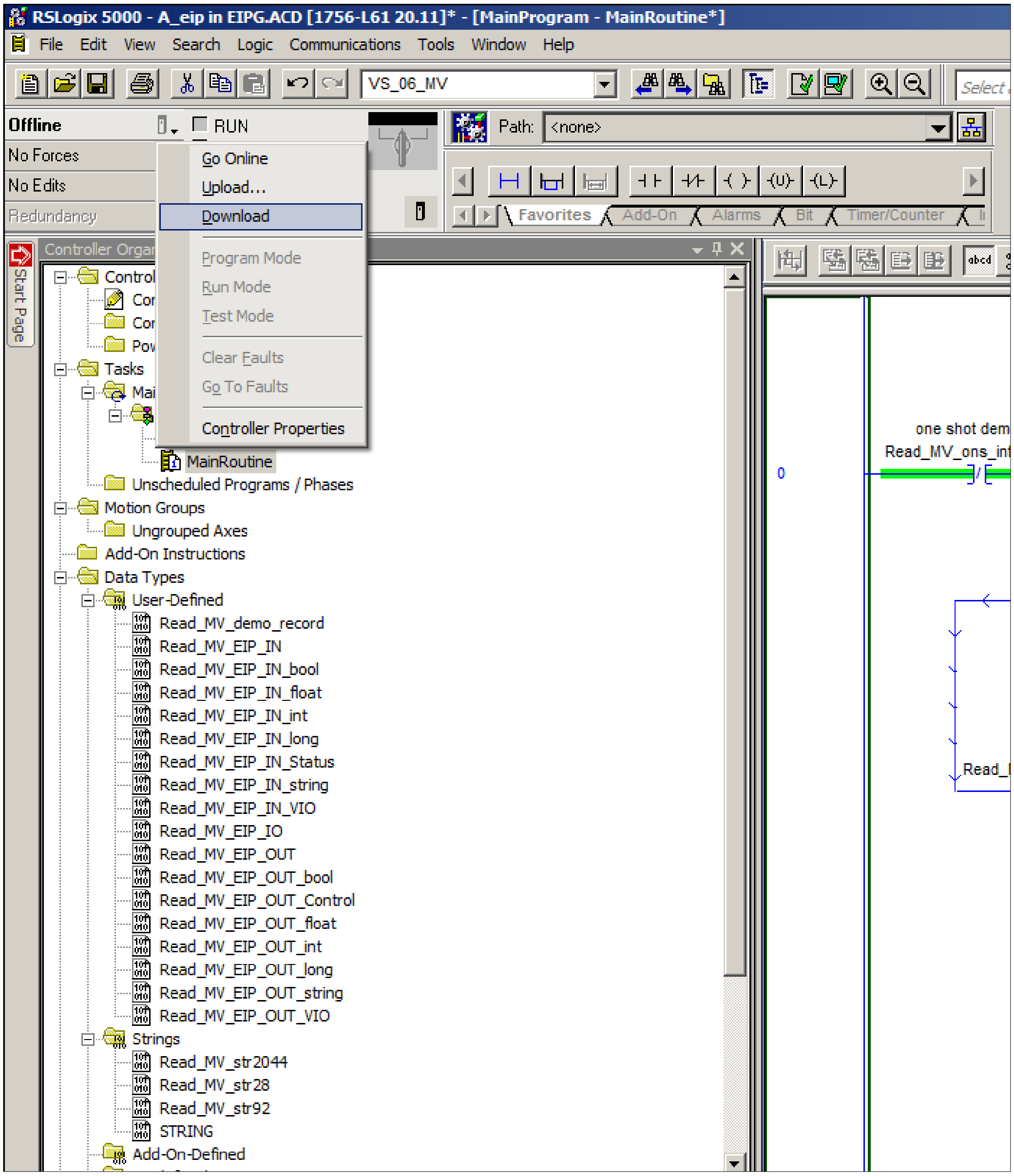
Click OK.

Delete any empty rungs (rung 0 may be empty):

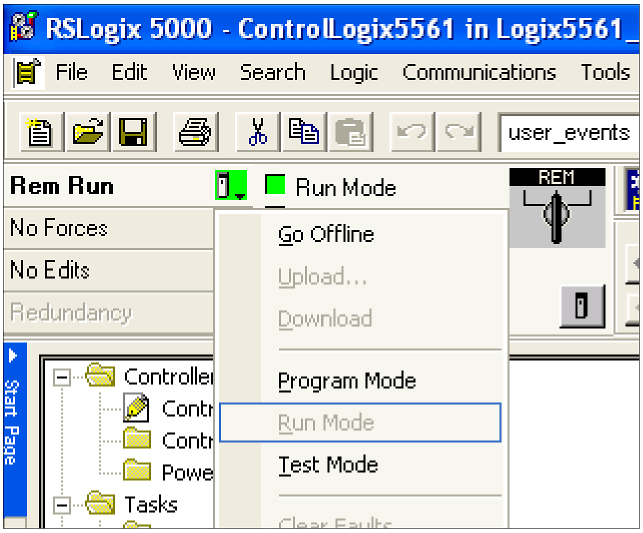


The tags and main program are now configured sufficiently to test communication with the camera.

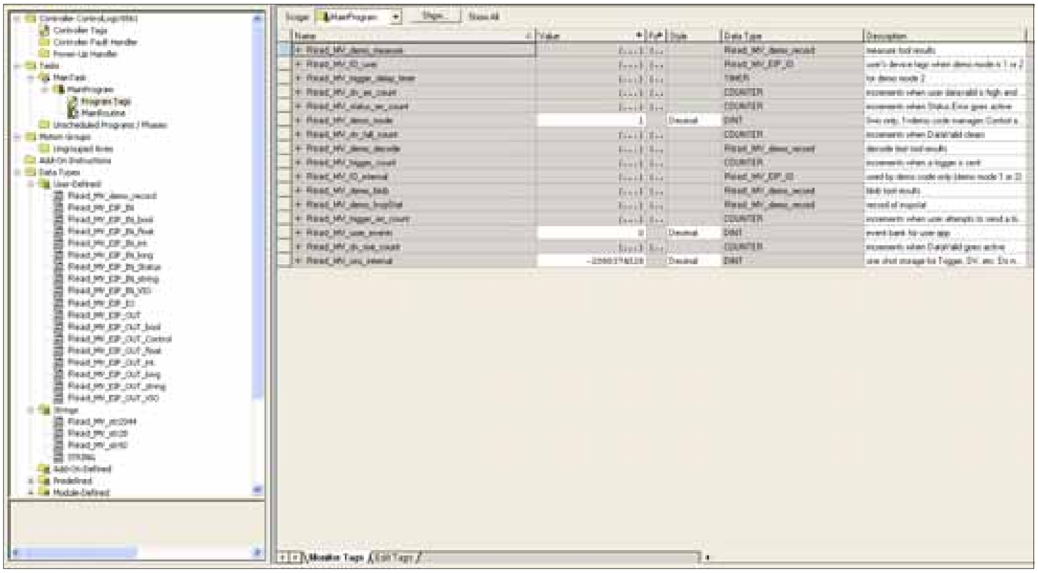
Select the control button next to “Offline”, and select “Download”:



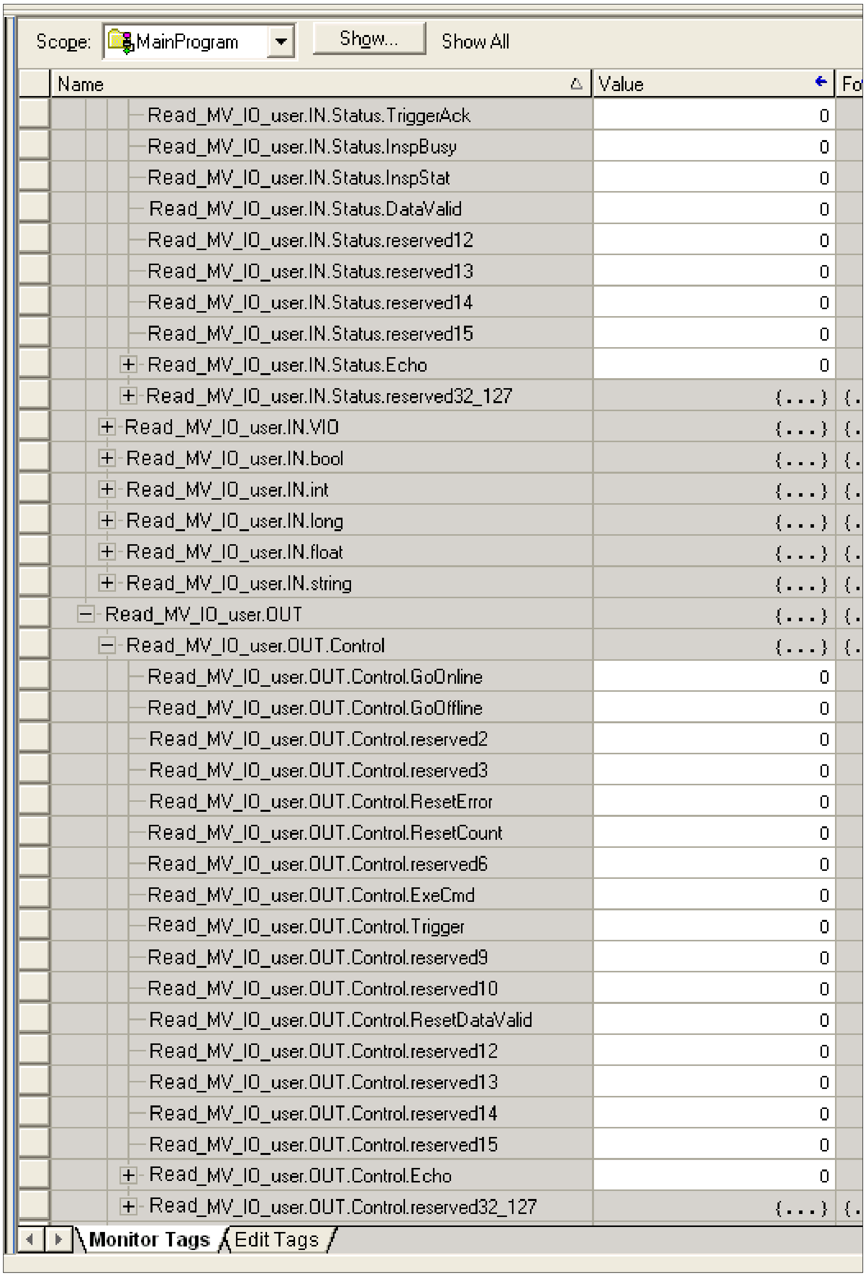
Once the program has downloaded, make sure the PLC is in Run Mode:



To open the Program Tags, double-click on "Program Tags", then select the "Monitor Tags" tab at the bottom of the tag window:

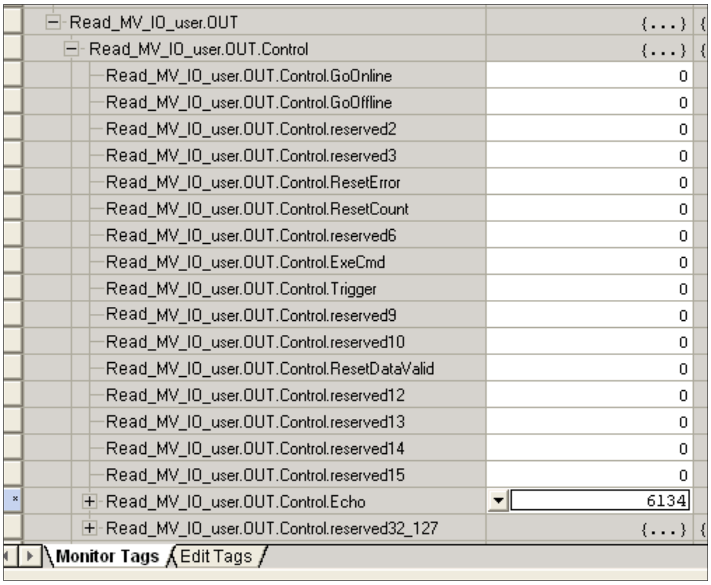


Expand Read_MV_IO_user so that the .IN.Status and .OUT.Control structures are visible, then scroll the window so Read_MV_IO_user.OUT.Control.Echo is visible:



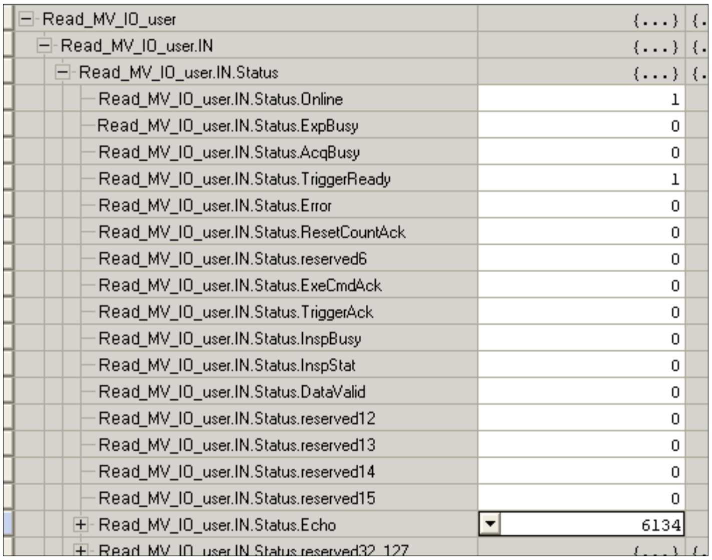
Name	Value	Fo
Read_MV_IO_user.IN.Status.TriggerAck	0	
Read_MV_IO_user.IN.Status.InspBusy	0	
Read_MV_IO_user.IN.Status.InspStat	0	
Read_MV_IO_user.IN.Status.DataValid	0	
Read_MV_IO_user.IN.Status.reserved12	0	
Read_MV_IO_user.IN.Status.reserved13	0	
Read_MV_IO_user.IN.Status.reserved14	0	
Read_MV_IO_user.IN.Status.reserved15	0	
+ Read_MV_IO_user.IN.Status.Echo	0	
+ Read_MV_IO_user.IN.Status.reserved32_127	{...}	{.
+ Read_MV_IO_user.IN.VIO	{...}	{.
+ Read_MV_IO_user.IN.bool	{...}	{.
+ Read_MV_IO_user.IN.int	{...}	{.
+ Read_MV_IO_user.IN.long	{...}	{.
+ Read_MV_IO_user.IN.float	{...}	{.
+ Read_MV_IO_user.IN.string	{...}	{.
− Read_MV_IO_user.OUT	{...}	{.
− Read_MV_IO_user.OUT.Control	{...}	{.
Read_MV_IO_user.OUT.Control.GoOnline	0	
Read_MV_IO_user.OUT.Control.GoOffline	0	
Read_MV_IO_user.OUT.Control.reserved2	0	
Read_MV_IO_user.OUT.Control.reserved3	0	
Read_MV_IO_user.OUT.Control.ResetError	0	
Read_MV_IO_user.OUT.Control.ResetCount	0	
Read_MV_IO_user.OUT.Control.reserved6	0	
Read_MV_IO_user.OUT.Control.ExeCmd	0	
Read_MV_IO_user.OUT.Control.Trigger	0	
Read_MV_IO_user.OUT.Control.reserved9	0	
Read_MV_IO_user.OUT.Control.reserved10	0	
Read_MV_IO_user.OUT.Control.ResetDataValid	0	
Read_MV_IO_user.OUT.Control.reserved12	0	
Read_MV_IO_user.OUT.Control.reserved13	0	
Read_MV_IO_user.OUT.Control.reserved14	0	
Read_MV_IO_user.OUT.Control.reserved15	0	
+ Read_MV_IO_user.OUT.Control.Echo	0	
+ Read_MV_IO_user.OUT.Control.reserved32_127	{...}	{.

Change .OUT.Control.Echo to non-zero:



Read_MV_IO_user.OUT	{...}
Read_MV_IO_user.OUT.Control	{...}
Read_MV_IO_user.OUT.Control.GoOnline	0
Read_MV_IO_user.OUT.Control.GoOffline	0
Read_MV_IO_user.OUT.Control.reserved2	0
Read_MV_IO_user.OUT.Control.reserved3	0
Read_MV_IO_user.OUT.Control.ResetError	0
Read_MV_IO_user.OUT.Control.ResetCount	0
Read_MV_IO_user.OUT.Control.reserved6	0
Read_MV_IO_user.OUT.Control.ExeCmd	0
Read_MV_IO_user.OUT.Control.Trigger	0
Read_MV_IO_user.OUT.Control.reserved9	0
Read_MV_IO_user.OUT.Control.reserved10	0
Read_MV_IO_user.OUT.Control.ResetDataValid	0
Read_MV_IO_user.OUT.Control.reserved12	0
Read_MV_IO_user.OUT.Control.reserved13	0
Read_MV_IO_user.OUT.Control.reserved14	0
Read_MV_IO_user.OUT.Control.reserved15	0
Read_MV_IO_user.OUT.Control.Echo	6134
Read_MV_IO_user.OUT.Control.reserved32_127	{...}

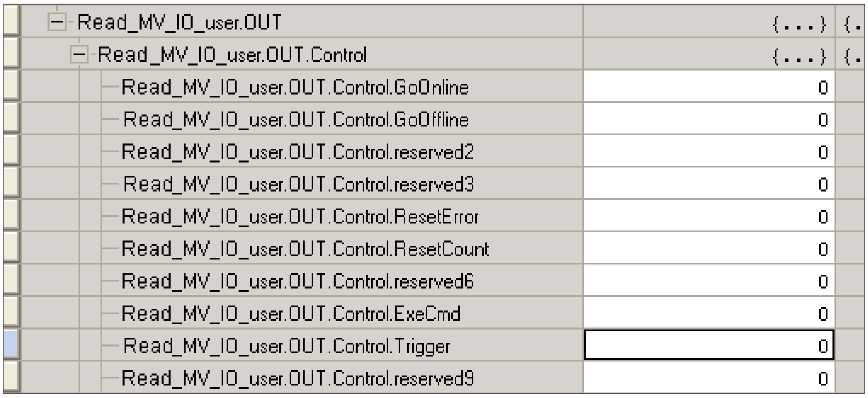
Scroll the window so Read_MV_IO_user.IO.IN.Status.Echo is visible, and verify it is the same value as the .OUT.Control.Echo:



Read_MV_IO_user	{...}
Read_MV_IO_user.IN	{...}
Read_MV_IO_user.IN.Status	{...}
Read_MV_IO_user.IN.Status.Online	1
Read_MV_IO_user.IN.Status.ExpBusy	0
Read_MV_IO_user.IN.Status.AcqBusy	0
Read_MV_IO_user.IN.Status.TriggerReady	1
Read_MV_IO_user.IN.Status.Error	0
Read_MV_IO_user.IN.Status.ResetCountAck	0
Read_MV_IO_user.IN.Status.reserved6	0
Read_MV_IO_user.IN.Status.ExeCmdAck	0
Read_MV_IO_user.IN.Status.TriggerAck	0
Read_MV_IO_user.IN.Status.InspBusy	0
Read_MV_IO_user.IN.Status.InspStat	0
Read_MV_IO_user.IN.Status.DataValid	0
Read_MV_IO_user.IN.Status.reserved12	0
Read_MV_IO_user.IN.Status.reserved13	0
Read_MV_IO_user.IN.Status.reserved14	0
Read_MV_IO_user.IN.Status.reserved15	0
Read_MV_IO_user.IN.Status.Echo	6134
Read_MV_IO_user.IN.Status.reserved32_127	{...}

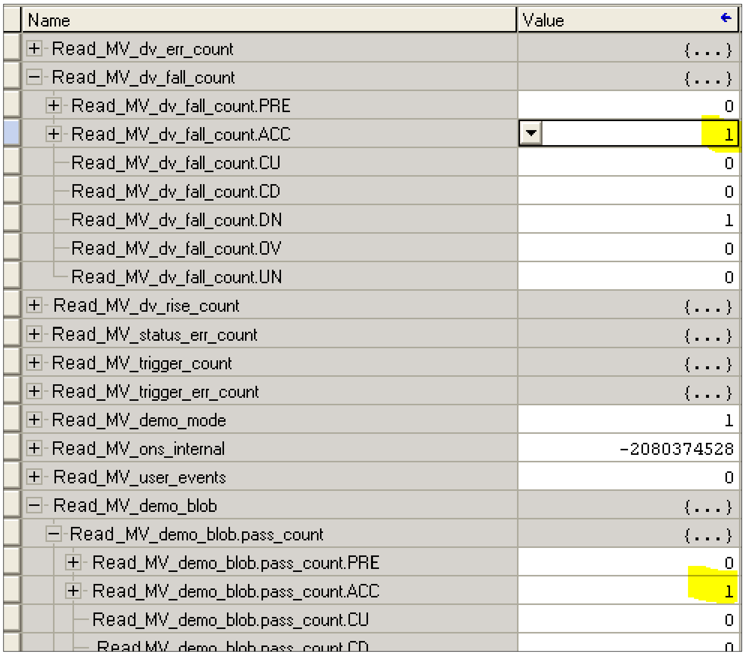
This confirms that the PLC and camera have successful two-way communication.

To send a trigger to the camera, scroll to Read_MV_IO_user.Control.Trigger:



Read_MV_IO_user.OUT	{...}
Read_MV_IO_user.OUT.Control	{...}
Read_MV_IO_user.OUT.Control.GoOnline	0
Read_MV_IO_user.OUT.Control.GoOffline	0
Read_MV_IO_user.OUT.Control.reserved2	0
Read_MV_IO_user.OUT.Control.reserved3	0
Read_MV_IO_user.OUT.Control.ResetError	0
Read_MV_IO_user.OUT.Control.ResetCount	0
Read_MV_IO_user.OUT.Control.reserved6	0
Read_MV_IO_user.OUT.Control.ExeCmd	0
Read_MV_IO_user.OUT.Control.Trigger	0
Read_MV_IO_user.OUT.Control.reserved9	0

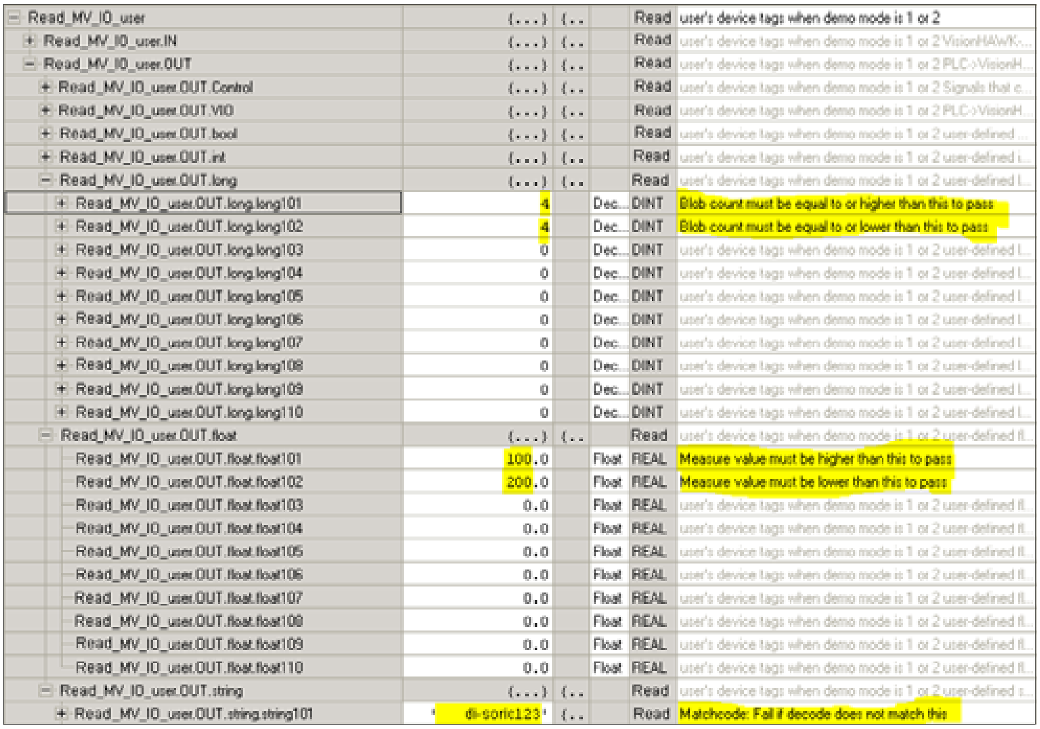
Set the Trigger to 1. This causes the demo code to trigger the camera, process the new inspection data, record the results in the Read_MV_demo_xxxx tags, and clear the DataValid status signal. The user can know that the camera was triggered when the Trigger control changes to 0. The user can know that all processing is done when the counter Read_MV_dv_fall_count increments, along with the pass/fail counters in the Read_MV_demo_xxxx tags. For example:



Name	Value
Read_MV_dv_err_count	{...}
Read_MV_dv_fall_count	{...}
Read_MV_dv_fall_count.PRE	0
Read_MV_dv_fall_count.ACC	1
Read_MV_dv_fall_count.CU	0
Read_MV_dv_fall_count.CD	0
Read_MV_dv_fall_count.DN	1
Read_MV_dv_fall_count.OV	0
Read_MV_dv_fall_count.UN	0
Read_MV_dv_rise_count	{...}
Read_MV_status_err_count	{...}
Read_MV_trigger_count	{...}
Read_MV_trigger_err_count	{...}
Read_MV_demo_mode	1
Read_MV_ons_internal	-2080374528
Read_MV_user_events	0
Read_MV_demo_blob	{...}
Read_MV_demo_blob.pass_count	{...}
Read_MV_demo_blob.pass_count.PRE	0
Read_MV_demo_blob.pass_count.ACC	1
Read_MV_demo_blob.pass_count.CU	0
Read_MV_demo_blob.pass_count.CD	0

Parameterize the Camera

Open the Read_MV_IO_user.OUT.long, float, and string tags and verify they are configured as follows:



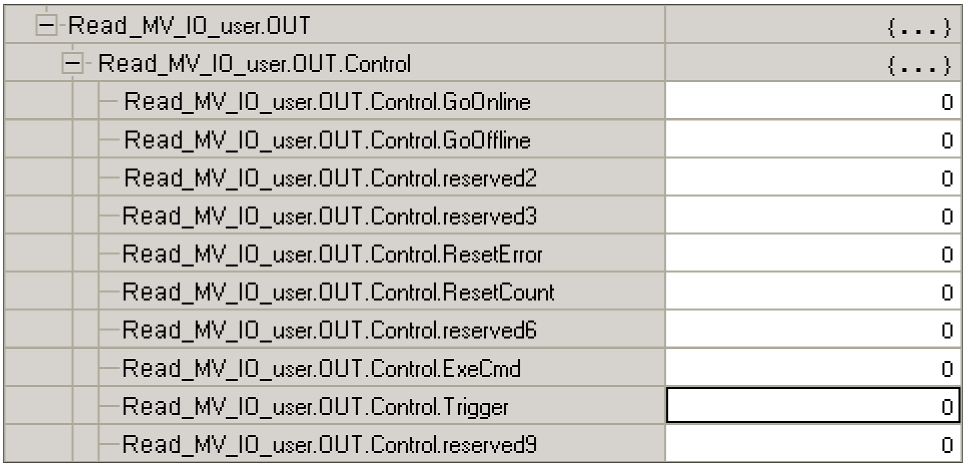
⊟ Read_MV_IO_user	{...}	{..		Read	user's device tags when demo mode is 1 or 2
⊞ Read_MV_IO_user.IN	{...}	{..		Read	user's device tags when demo mode is 1 or 2 VisionHAWK-...
⊟ Read_MV_IO_user.OUT	{...}	{..		Read	user's device tags when demo mode is 1 or 2 PLC->VisionH...
⊞ Read_MV_IO_user.OUT.Control	{...}	{..		Read	user's device tags when demo mode is 1 or 2 Signals that c...
⊞ Read_MV_IO_user.OUT.VIO	{...}	{..		Read	user's device tags when demo mode is 1 or 2 PLC->VisionH...
⊞ Read_MV_IO_user.OUT.bool	{...}	{..		Read	user's device tags when demo mode is 1 or 2 user-defined ...
⊞ Read_MV_IO_user.OUT.int	{...}	{..		Read	user's device tags when demo mode is 1 or 2 user-defined i...
⊟ Read_MV_IO_user.OUT.long	{...}	{..		Read	user's device tags when demo mode is 1 or 2 user-defined l...
⊞ Read_MV_IO_user.OUT.long.long101	4		Dec...	DINT	Blob count must be equal to or higher than this to pass
⊞ Read_MV_IO_user.OUT.long.long102	4		Dec...	DINT	Blob count must be equal to or lower than this to pass
⊞ Read_MV_IO_user.OUT.long.long103	0		Dec...	DINT	user's device tags when demo mode is 1 or 2 user-defined l...
⊞ Read_MV_IO_user.OUT.long.long104	0		Dec...	DINT	user's device tags when demo mode is 1 or 2 user-defined l...
⊞ Read_MV_IO_user.OUT.long.long105	0		Dec...	DINT	user's device tags when demo mode is 1 or 2 user-defined l...
⊞ Read_MV_IO_user.OUT.long.long106	0		Dec...	DINT	user's device tags when demo mode is 1 or 2 user-defined l...
⊞ Read_MV_IO_user.OUT.long.long107	0		Dec...	DINT	user's device tags when demo mode is 1 or 2 user-defined l...
⊞ Read_MV_IO_user.OUT.long.long108	0		Dec...	DINT	user's device tags when demo mode is 1 or 2 user-defined l...
⊞ Read_MV_IO_user.OUT.long.long109	0		Dec...	DINT	user's device tags when demo mode is 1 or 2 user-defined l...
⊞ Read_MV_IO_user.OUT.long.long110	0		Dec...	DINT	user's device tags when demo mode is 1 or 2 user-defined l...
⊟ Read_MV_IO_user.OUT.float	{...}	{..		Read	user's device tags when demo mode is 1 or 2 user-defined fl...
Read_MV_IO_user.OUT.float.float101	100.0		Float	REAL	Measure value must be higher than this to pass
Read_MV_IO_user.OUT.float.float102	200.0		Float	REAL	Measure value must be lower than this to pass
Read_MV_IO_user.OUT.float.float103	0.0		Float	REAL	user's device tags when demo mode is 1 or 2 user-defined fl...
Read_MV_IO_user.OUT.float.float104	0.0		Float	REAL	user's device tags when demo mode is 1 or 2 user-defined fl...
Read_MV_IO_user.OUT.float.float105	0.0		Float	REAL	user's device tags when demo mode is 1 or 2 user-defined fl...
Read_MV_IO_user.OUT.float.float106	0.0		Float	REAL	user's device tags when demo mode is 1 or 2 user-defined fl...
Read_MV_IO_user.OUT.float.float107	0.0		Float	REAL	user's device tags when demo mode is 1 or 2 user-defined fl...
Read_MV_IO_user.OUT.float.float108	0.0		Float	REAL	user's device tags when demo mode is 1 or 2 user-defined fl...
Read_MV_IO_user.OUT.float.float109	0.0		Float	REAL	user's device tags when demo mode is 1 or 2 user-defined fl...
Read_MV_IO_user.OUT.float.float110	0.0		Float	REAL	user's device tags when demo mode is 1 or 2 user-defined fl...
⊟ Read_MV_IO_user.OUT.string	{...}	{..		Read	user's device tags when demo mode is 1 or 2 user-defined s...
⊞ Read_MV_IO_user.OUT.string.string101	'di-soric123'	{..		Read	Matchcode: Fail if decode does not match this

This configures the Measure (float101 and float102), Decode (string101) and Count Blob (long101 and long102) tools in the same way they were configured in AutoVision during Try Out.

Note the Description column. It offers a hint for what each linked tag does for the vision job.

Trigger the Camera

To send a trigger to the camera, scroll to Read_MV_IO_user.Control.Trigger:



Read_MV_IO_user.OUT	{...}
Read_MV_IO_user.OUT.Control	{...}
Read_MV_IO_user.OUT.Control.GoOnline	0
Read_MV_IO_user.OUT.Control.GoOffline	0
Read_MV_IO_user.OUT.Control.reserved2	0
Read_MV_IO_user.OUT.Control.reserved3	0
Read_MV_IO_user.OUT.Control.ResetError	0
Read_MV_IO_user.OUT.Control.ResetCount	0
Read_MV_IO_user.OUT.Control.reserved6	0
Read_MV_IO_user.OUT.Control.ExeCmd	0
Read_MV_IO_user.OUT.Control.Trigger	0
Read_MV_IO_user.OUT.Control.reserved9	0

Set the Trigger to 1. When the Trigger returns to a value of 0, the camera may be retriggered.

If you connect to the camera with AutoVision, it will display a new Inspection result each time the camera is triggered. Recall that the vision job was created with pre-defined images to produce predictable “Passed” and “Failed” results. The camera’s illumination lights will not flash when triggered.

The Inspection results can be seen in the PLCs’s IN tags, and well as in AutoVision. Open the RSLogix tag window so Read_MV_IO_user.IN.Status and bool are visible.

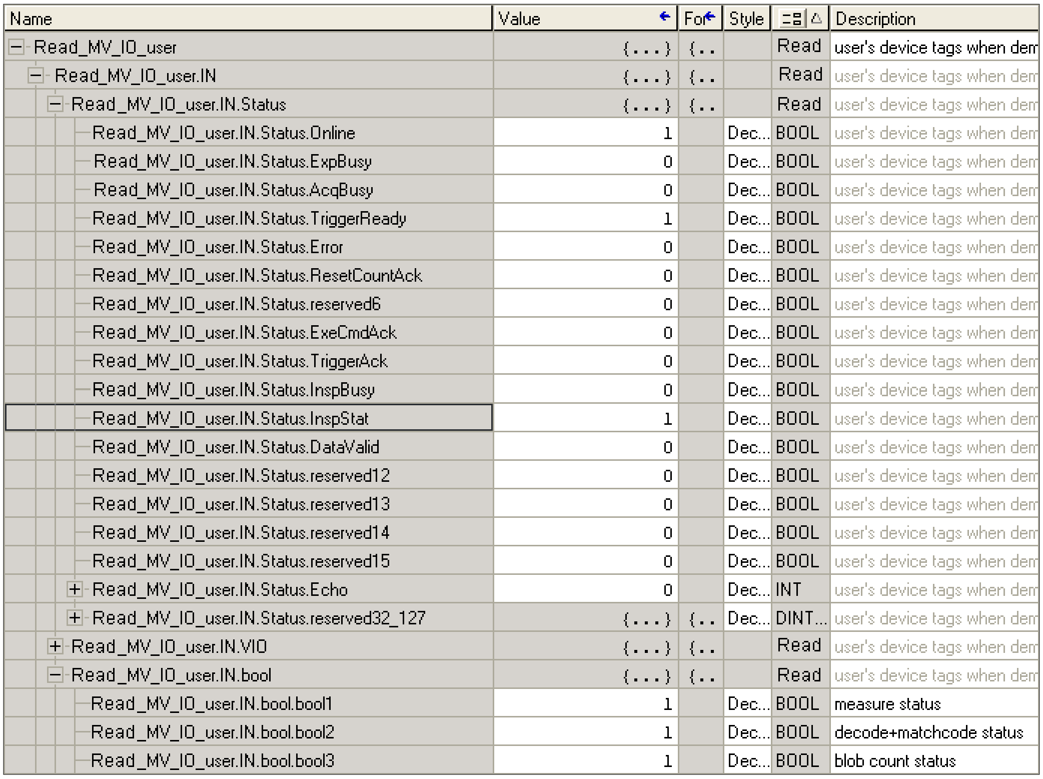
This example shows a “Passed” inspection, where the following tags are all 1:

IN.Status.InspStat

IN.bool.bool1 (Measure status)

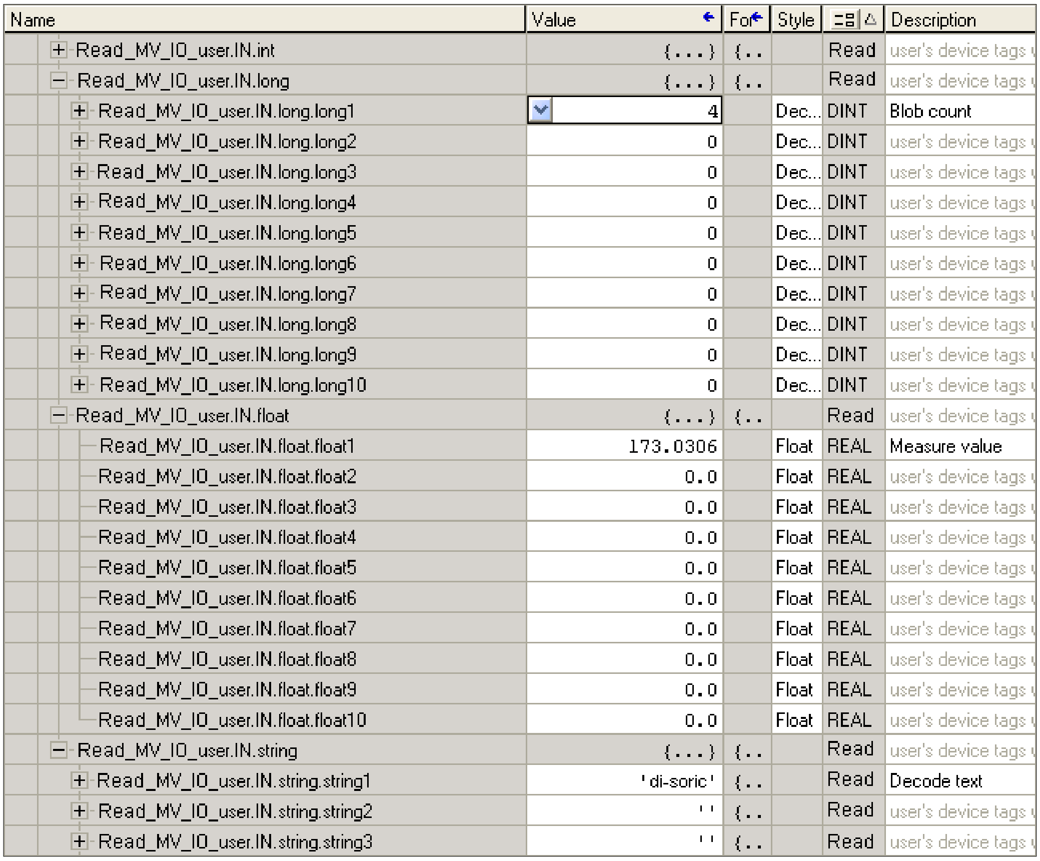
IN.bool.bool2 (decode+matchcode status)

IN.bool.bool3 (count blob status)



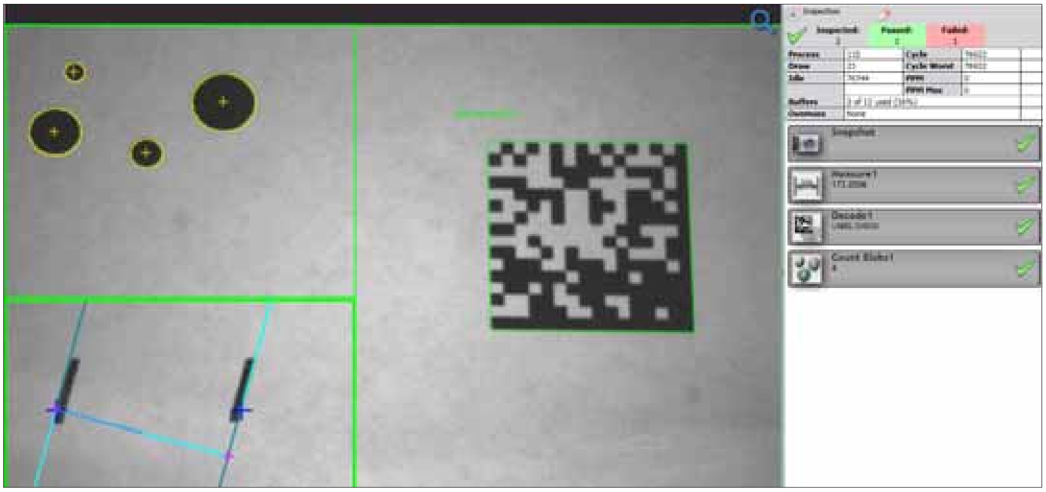
Name	Value	For	Style		Description
Read_MV_IO_user	{...}	{..		Read	user's device tags when dem
Read_MV_IO_user.IN	{...}	{..		Read	user's device tags when dem
Read_MV_IO_user.IN.Status	{...}	{..		Read	user's device tags when dem
Read_MV_IO_user.IN.Status.Online	1		Dec...	BOOL	user's device tags when dem
Read_MV_IO_user.IN.Status.ExpBusy	0		Dec...	BOOL	user's device tags when dem
Read_MV_IO_user.IN.Status.AcqBusy	0		Dec...	BOOL	user's device tags when dem
Read_MV_IO_user.IN.Status.TriggerReady	1		Dec...	BOOL	user's device tags when dem
Read_MV_IO_user.IN.Status.Error	0		Dec...	BOOL	user's device tags when dem
Read_MV_IO_user.IN.Status.ResetCountAck	0		Dec...	BOOL	user's device tags when dem
Read_MV_IO_user.IN.Status.reserved6	0		Dec...	BOOL	user's device tags when dem
Read_MV_IO_user.IN.Status.ExeCmdAck	0		Dec...	BOOL	user's device tags when dem
Read_MV_IO_user.IN.Status.TriggerAck	0		Dec...	BOOL	user's device tags when dem
Read_MV_IO_user.IN.Status.InspBusy	0		Dec...	BOOL	user's device tags when dem
Read_MV_IO_user.IN.Status.InspStat	1		Dec...	BOOL	user's device tags when dem
Read_MV_IO_user.IN.Status.DataValid	0		Dec...	BOOL	user's device tags when dem
Read_MV_IO_user.IN.Status.reserved12	0		Dec...	BOOL	user's device tags when dem
Read_MV_IO_user.IN.Status.reserved13	0		Dec...	BOOL	user's device tags when dem
Read_MV_IO_user.IN.Status.reserved14	0		Dec...	BOOL	user's device tags when dem
Read_MV_IO_user.IN.Status.reserved15	0		Dec...	BOOL	user's device tags when dem
Read_MV_IO_user.IN.Status.Echo	0		Dec...	INT	user's device tags when dem
Read_MV_IO_user.IN.Status.reserved32_127	{...}	{..	Dec...	DINT...	user's device tags when dem
Read_MV_IO_user.IN.VIO	{...}	{..		Read	user's device tags when dem
Read_MV_IO_user.IN.bool	{...}	{..		Read	user's device tags when dem
Read_MV_IO_user.IN.bool.bool1	1		Dec...	BOOL	measure status
Read_MV_IO_user.IN.bool.bool2	1		Dec...	BOOL	decode+matchcode status
Read_MV_IO_user.IN.bool.bool3	1		Dec...	BOOL	blob count status

If you scroll down to the IN.long, float and string values, you will see the literal results of the vision tools:

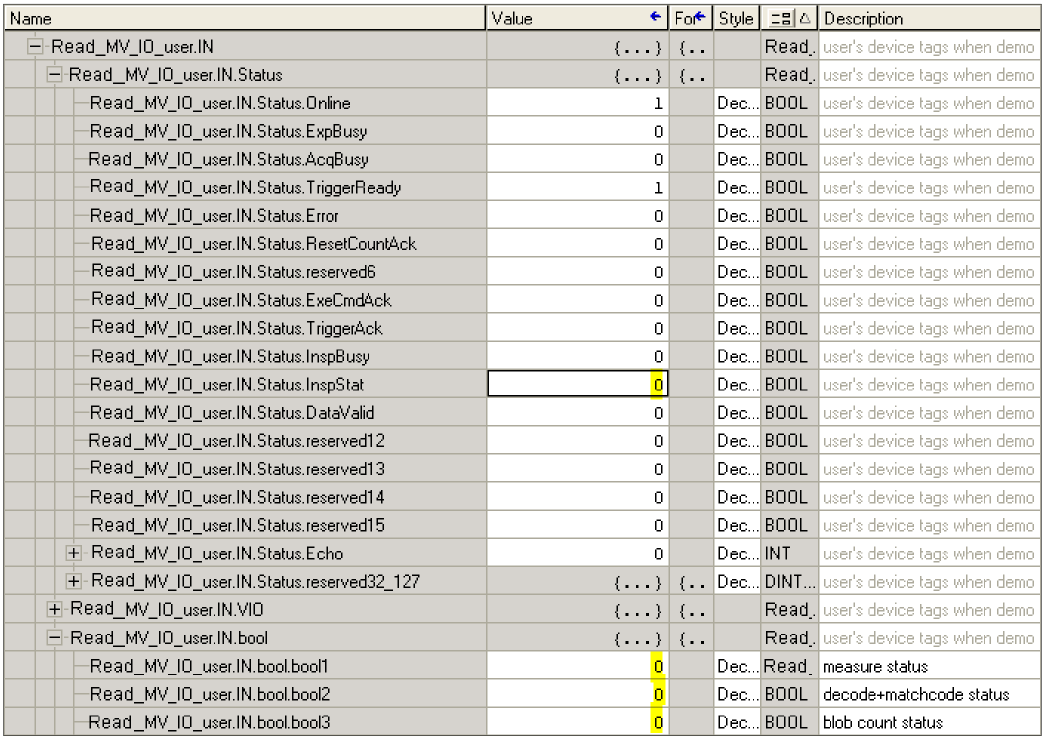


Name	Value	For	Style		Description
+ Read_MV_IO_user.IN.int	{...}	{..		Read	user's device tags
− Read_MV_IO_user.IN.long	{...}	{..		Read	user's device tags
+ Read_MV_IO_user.IN.long.long1	4		Dec...	DINT	Blob count
+ Read_MV_IO_user.IN.long.long2	0		Dec...	DINT	user's device tags
+ Read_MV_IO_user.IN.long.long3	0		Dec...	DINT	user's device tags
+ Read_MV_IO_user.IN.long.long4	0		Dec...	DINT	user's device tags
+ Read_MV_IO_user.IN.long.long5	0		Dec...	DINT	user's device tags
+ Read_MV_IO_user.IN.long.long6	0		Dec...	DINT	user's device tags
+ Read_MV_IO_user.IN.long.long7	0		Dec...	DINT	user's device tags
+ Read_MV_IO_user.IN.long.long8	0		Dec...	DINT	user's device tags
+ Read_MV_IO_user.IN.long.long9	0		Dec...	DINT	user's device tags
+ Read_MV_IO_user.IN.long.long10	0		Dec...	DINT	user's device tags
− Read_MV_IO_user.IN.float	{...}	{..		Read	user's device tags
Read_MV_IO_user.IN.float.float1	173.0306		Float	REAL	Measure value
Read_MV_IO_user.IN.float.float2	0.0		Float	REAL	user's device tags
Read_MV_IO_user.IN.float.float3	0.0		Float	REAL	user's device tags
Read_MV_IO_user.IN.float.float4	0.0		Float	REAL	user's device tags
Read_MV_IO_user.IN.float.float5	0.0		Float	REAL	user's device tags
Read_MV_IO_user.IN.float.float6	0.0		Float	REAL	user's device tags
Read_MV_IO_user.IN.float.float7	0.0		Float	REAL	user's device tags
Read_MV_IO_user.IN.float.float8	0.0		Float	REAL	user's device tags
Read_MV_IO_user.IN.float.float9	0.0		Float	REAL	user's device tags
Read_MV_IO_user.IN.float.float10	0.0		Float	REAL	user's device tags
− Read_MV_IO_user.IN.string	{...}	{..		Read	user's device tags
+ Read_MV_IO_user.IN.string.string1	'di-soric'	{..		Read	Decode text
+ Read_MV_IO_user.IN.string.string2	''	{..		Read	user's device tags
+ Read_MV_IO_user.IN.string.string3	''	{..		Read	user's device tags

This is equivalent to the AutoVision inspection result:

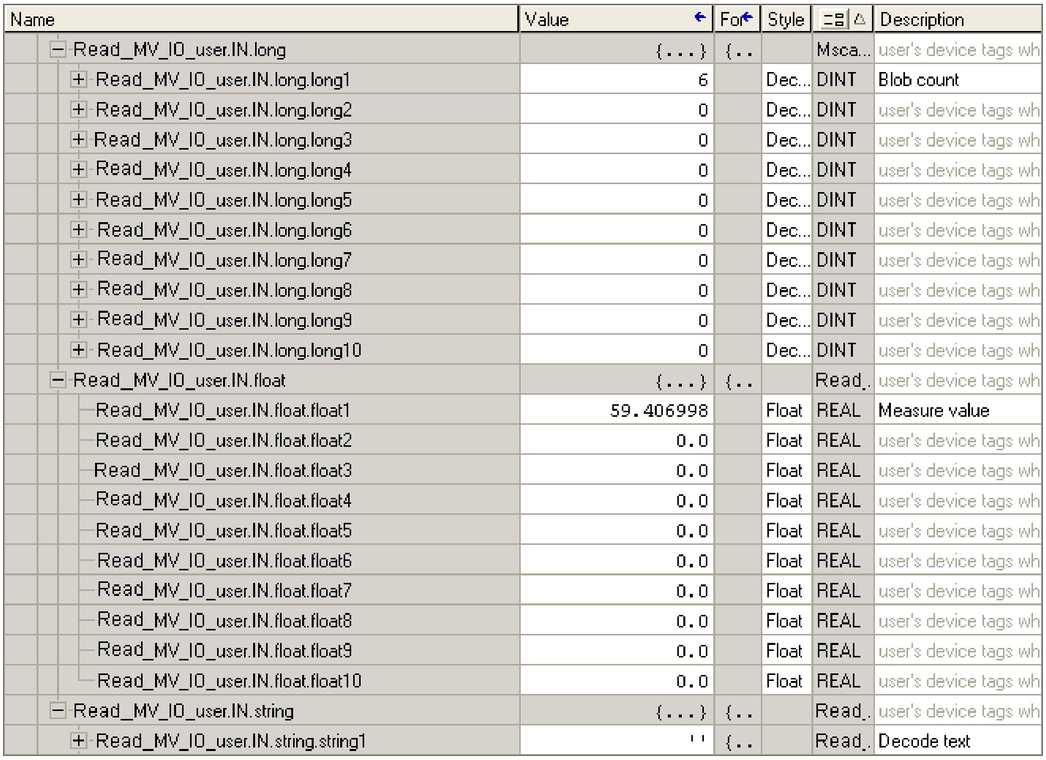


This example shows a “Failed” inspection, where every tool reports a fail:



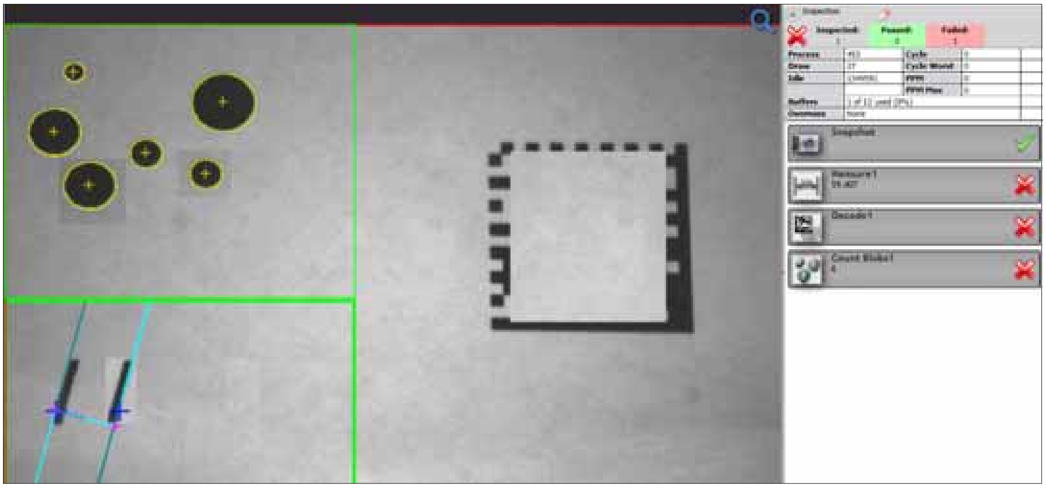
Name	Value	For	Style		Description
⊟ Read_MV_IO_user.IN	{...}	{..		Read_	user's device tags when demo
⊟ Read_MV_IO_user.IN.Status	{...}	{..		Read_	user's device tags when demo
Read_MV_IO_user.IN.Status.Online	1		Dec...	BOOL	user's device tags when demo
Read_MV_IO_user.IN.Status.ExpBusy	0		Dec...	BOOL	user's device tags when demo
Read_MV_IO_user.IN.Status.AcqBusy	0		Dec...	BOOL	user's device tags when demo
Read_MV_IO_user.IN.Status.TriggerReady	1		Dec...	BOOL	user's device tags when demo
Read_MV_IO_user.IN.Status.Error	0		Dec...	BOOL	user's device tags when demo
Read_MV_IO_user.IN.Status.ResetCountAck	0		Dec...	BOOL	user's device tags when demo
Read_MV_IO_user.IN.Status.reserved6	0		Dec...	BOOL	user's device tags when demo
Read_MV_IO_user.IN.Status.ExeCmdAck	0		Dec...	BOOL	user's device tags when demo
Read_MV_IO_user.IN.Status.TriggerAck	0		Dec...	BOOL	user's device tags when demo
Read_MV_IO_user.IN.Status.InspBusy	0		Dec...	BOOL	user's device tags when demo
Read_MV_IO_user.IN.Status.InspStat	0		Dec...	BOOL	user's device tags when demo
Read_MV_IO_user.IN.Status.DataValid	0		Dec...	BOOL	user's device tags when demo
Read_MV_IO_user.IN.Status.reserved12	0		Dec...	BOOL	user's device tags when demo
Read_MV_IO_user.IN.Status.reserved13	0		Dec...	BOOL	user's device tags when demo
Read_MV_IO_user.IN.Status.reserved14	0		Dec...	BOOL	user's device tags when demo
Read_MV_IO_user.IN.Status.reserved15	0		Dec...	BOOL	user's device tags when demo
⊞ Read_MV_IO_user.IN.Status.Echo	0		Dec...	INT	user's device tags when demo
⊞ Read_MV_IO_user.IN.Status.reserved32_127	{...}	{..	Dec...	DINT...	user's device tags when demo
⊞ Read_MV_IO_user.IN.VIO	{...}	{..		Read_	user's device tags when demo
⊟ Read_MV_IO_user.IN.bool	{...}	{..		Read_	user's device tags when demo
Read_MV_IO_user.IN.bool.bool1	0		Dec...	Read_	measure status
Read_MV_IO_user.IN.bool.bool2	0		Dec...	BOOL	decode+matchcode status
Read_MV_IO_user.IN.bool.bool3	0		Dec...	BOOL	blob count status

This is the Failed inspection's literal data:



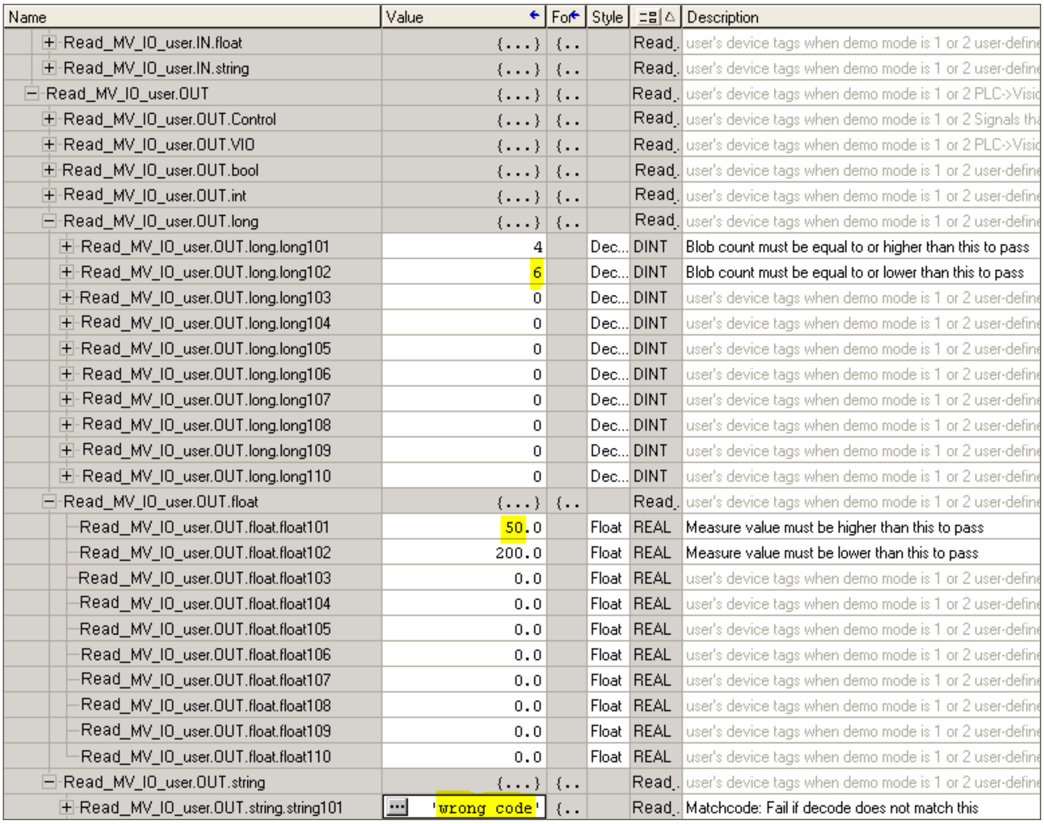
Name	Value	For	Style		Description
⊟ Read_MV_IO_user.IN.long	{...}	{..		Msca...	user's device tags wh
⊞ Read_MV_IO_user.IN.long.long1	6		Dec...	DINT	Blob count
⊞ Read_MV_IO_user.IN.long.long2	0		Dec...	DINT	user's device tags wh
⊞ Read_MV_IO_user.IN.long.long3	0		Dec...	DINT	user's device tags wh
⊞ Read_MV_IO_user.IN.long.long4	0		Dec...	DINT	user's device tags wh
⊞ Read_MV_IO_user.IN.long.long5	0		Dec...	DINT	user's device tags wh
⊞ Read_MV_IO_user.IN.long.long6	0		Dec...	DINT	user's device tags wh
⊞ Read_MV_IO_user.IN.long.long7	0		Dec...	DINT	user's device tags wh
⊞ Read_MV_IO_user.IN.long.long8	0		Dec...	DINT	user's device tags wh
⊞ Read_MV_IO_user.IN.long.long9	0		Dec...	DINT	user's device tags wh
⊞ Read_MV_IO_user.IN.long.long10	0		Dec...	DINT	user's device tags wh
⊟ Read_MV_IO_user.IN.float	{...}	{..		Read_	user's device tags wh
Read_MV_IO_user.IN.float.float1	59.406998		Float	REAL	Measure value
Read_MV_IO_user.IN.float.float2	0.0		Float	REAL	user's device tags wh
Read_MV_IO_user.IN.float.float3	0.0		Float	REAL	user's device tags wh
Read_MV_IO_user.IN.float.float4	0.0		Float	REAL	user's device tags wh
Read_MV_IO_user.IN.float.float5	0.0		Float	REAL	user's device tags wh
Read_MV_IO_user.IN.float.float6	0.0		Float	REAL	user's device tags wh
Read_MV_IO_user.IN.float.float7	0.0		Float	REAL	user's device tags wh
Read_MV_IO_user.IN.float.float8	0.0		Float	REAL	user's device tags wh
Read_MV_IO_user.IN.float.float9	0.0		Float	REAL	user's device tags wh
Read_MV_IO_user.IN.float.float10	0.0		Float	REAL	user's device tags wh
⊟ Read_MV_IO_user.IN.string	{...}	{..		Read_	user's device tags wh
⊞ Read_MV_IO_user.IN.string.string1	''	{..		Read_	Decode text

This is equivalent to the AutoVision inspection report:



Parameterize the Camera Again

The Measure and Count Blob tools can be parameterized by the PLC so they always pass. The Decode tool can be parameterized so it always fails, either due to no decode, or a matchcode mismatch. Scroll the tag window so OUT.long, float and string are visible, then change them as shown here:



Name	Value	For	Style		Description
⊞ Read_MV_IO_user.IN.float	{...}	{..		Read.	user's device tags when demo mode is 1 or 2 user-define
⊞ Read_MV_IO_user.IN.string	{...}	{..		Read.	user's device tags when demo mode is 1 or 2 user-define
⊟ Read_MV_IO_user.OUT	{...}	{..		Read.	user's device tags when demo mode is 1 or 2 PLC->Visio
⊞ Read_MV_IO_user.OUT.Control	{...}	{..		Read.	user's device tags when demo mode is 1 or 2 Signals tha
⊞ Read_MV_IO_user.OUT.VIO	{...}	{..		Read.	user's device tags when demo mode is 1 or 2 PLC->Visio
⊞ Read_MV_IO_user.OUT.bool	{...}	{..		Read.	user's device tags when demo mode is 1 or 2 user-define
⊞ Read_MV_IO_user.OUT.int	{...}	{..		Read.	user's device tags when demo mode is 1 or 2 user-define
⊟ Read_MV_IO_user.OUT.long	{...}	{..		Read.	user's device tags when demo mode is 1 or 2 user-define
⊞ Read_MV_IO_user.OUT.long.long101	4		Dec...	DINT	Blob count must be equal to or higher than this to pass
⊞ Read_MV_IO_user.OUT.long.long102	6		Dec...	DINT	Blob count must be equal to or lower than this to pass
⊞ Read_MV_IO_user.OUT.long.long103	0		Dec...	DINT	user's device tags when demo mode is 1 or 2 user-define
⊞ Read_MV_IO_user.OUT.long.long104	0		Dec...	DINT	user's device tags when demo mode is 1 or 2 user-define
⊞ Read_MV_IO_user.OUT.long.long105	0		Dec...	DINT	user's device tags when demo mode is 1 or 2 user-define
⊞ Read_MV_IO_user.OUT.long.long106	0		Dec...	DINT	user's device tags when demo mode is 1 or 2 user-define
⊞ Read_MV_IO_user.OUT.long.long107	0		Dec...	DINT	user's device tags when demo mode is 1 or 2 user-define
⊞ Read_MV_IO_user.OUT.long.long108	0		Dec...	DINT	user's device tags when demo mode is 1 or 2 user-define
⊞ Read_MV_IO_user.OUT.long.long109	0		Dec...	DINT	user's device tags when demo mode is 1 or 2 user-define
⊞ Read_MV_IO_user.OUT.long.long110	0		Dec...	DINT	user's device tags when demo mode is 1 or 2 user-define
⊟ Read_MV_IO_user.OUT.float	{...}	{..		Read.	user's device tags when demo mode is 1 or 2 user-define
Read_MV_IO_user.OUT.float.float101	50.0		Float	REAL	Measure value must be higher than this to pass
Read_MV_IO_user.OUT.float.float102	200.0		Float	REAL	Measure value must be lower than this to pass
Read_MV_IO_user.OUT.float.float103	0.0		Float	REAL	user's device tags when demo mode is 1 or 2 user-define
Read_MV_IO_user.OUT.float.float104	0.0		Float	REAL	user's device tags when demo mode is 1 or 2 user-define
Read_MV_IO_user.OUT.float.float105	0.0		Float	REAL	user's device tags when demo mode is 1 or 2 user-define
Read_MV_IO_user.OUT.float.float106	0.0		Float	REAL	user's device tags when demo mode is 1 or 2 user-define
Read_MV_IO_user.OUT.float.float107	0.0		Float	REAL	user's device tags when demo mode is 1 or 2 user-define
Read_MV_IO_user.OUT.float.float108	0.0		Float	REAL	user's device tags when demo mode is 1 or 2 user-define
Read_MV_IO_user.OUT.float.float109	0.0		Float	REAL	user's device tags when demo mode is 1 or 2 user-define
Read_MV_IO_user.OUT.float.float110	0.0		Float	REAL	user's device tags when demo mode is 1 or 2 user-define
⊟ Read_MV_IO_user.OUT.string	{...}	{..		Read.	user's device tags when demo mode is 1 or 2 user-define
⊞ Read_MV_IO_user.OUT.string.string101	'wrong code'	{..		Read.	Matchcode: Fail if decode does not match this

Trigger the Camera Again

Trigger the camera twice, and you will see the Status results stay the same for all triggers:

bool2 (decode+matchcode status) = 0

Why: Decode+Matchcode status always fails because the matchcode has been changed to “wrong code”, or there is no decode.

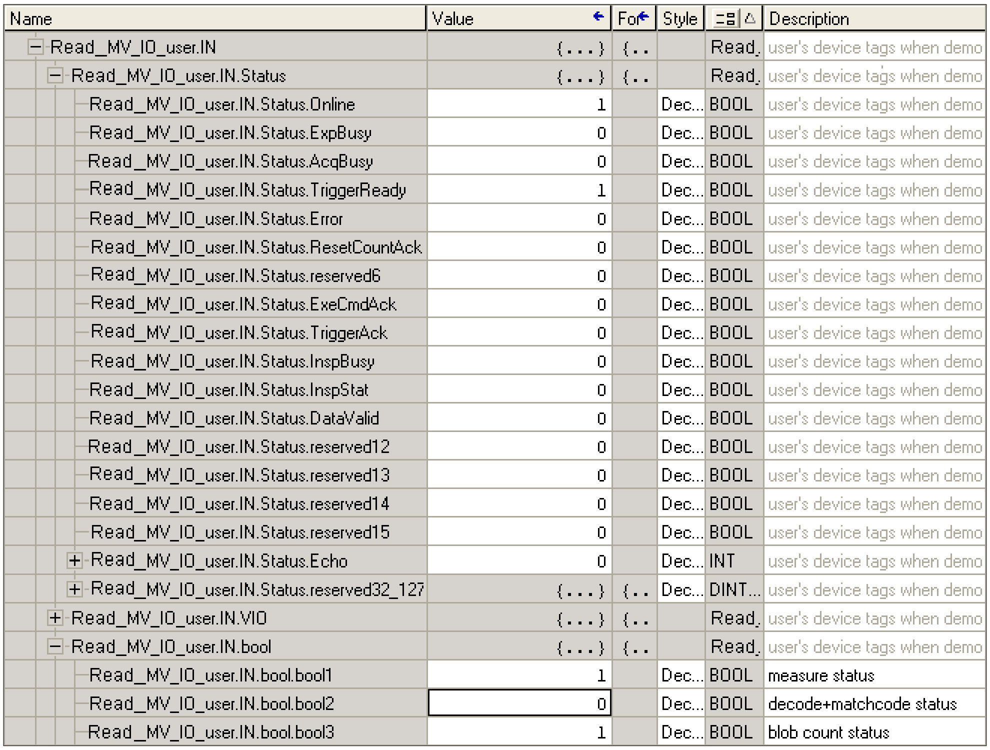
bool1 (Measure status) and bool3 (count blob status) = 1

Why: The inspected values are now in tolerance.

InspStat = 0

Why: The Decode tool fails, so the overall Inspection result is a Fail.

PLC tags:



Name	Value	For	Style		Description
⊟ Read_MV_IO_user.IN	{...}	{..		Read.	user's device tags when demo
⊟ Read_MV_IO_user.IN.Status	{...}	{..		Read.	user's device tags when demo
Read_MV_IO_user.IN.Status.Online	1		Dec...	BOOL	user's device tags when demo
Read_MV_IO_user.IN.Status.ExpBusy	0		Dec...	BOOL	user's device tags when demo
Read_MV_IO_user.IN.Status.AcqBusy	0		Dec...	BOOL	user's device tags when demo
Read_MV_IO_user.IN.Status.TriggerReady	1		Dec...	BOOL	user's device tags when demo
Read_MV_IO_user.IN.Status.Error	0		Dec...	BOOL	user's device tags when demo
Read_MV_IO_user.IN.Status.ResetCountAck	0		Dec...	BOOL	user's device tags when demo
Read_MV_IO_user.IN.Status.reserved6	0		Dec...	BOOL	user's device tags when demo
Read_MV_IO_user.IN.Status.ExeCmdAck	0		Dec...	BOOL	user's device tags when demo
Read_MV_IO_user.IN.Status.TriggerAck	0		Dec...	BOOL	user's device tags when demo
Read_MV_IO_user.IN.Status.InspBusy	0		Dec...	BOOL	user's device tags when demo
Read_MV_IO_user.IN.Status.InspStat	0		Dec...	BOOL	user's device tags when demo
Read_MV_IO_user.IN.Status.DataValid	0		Dec...	BOOL	user's device tags when demo
Read_MV_IO_user.IN.Status.reserved12	0		Dec...	BOOL	user's device tags when demo
Read_MV_IO_user.IN.Status.reserved13	0		Dec...	BOOL	user's device tags when demo
Read_MV_IO_user.IN.Status.reserved14	0		Dec...	BOOL	user's device tags when demo
Read_MV_IO_user.IN.Status.reserved15	0		Dec...	BOOL	user's device tags when demo
⊞ Read_MV_IO_user.IN.Status.Echo	0		Dec...	INT	user's device tags when demo
⊞ Read_MV_IO_user.IN.Status.reserved32_127	{...}	{..	Dec...	DINT...	user's device tags when demo
⊞ Read_MV_IO_user.IN.VIO	{...}	{..		Read.	user's device tags when demo
⊟ Read_MV_IO_user.IN.bool	{...}	{..		Read.	user's device tags when demo
Read_MV_IO_user.IN.bool.bool1	1		Dec...	BOOL	measure status
Read_MV_IO_user.IN.bool.bool2	0		Dec...	BOOL	decode+matchcode status
Read_MV_IO_user.IN.bool.bool3	1		Dec...	BOOL	blob count status

This concludes the EtherNet/IP demo.

G

APPENDIX G Demo PLC Code

This section describes how to use di-soric demo PLC code with a vision job and camera target.

The EIP demo files can be found where AutoVision is installed, in the folder C:\di-soric\Vscape\Tutorials and Samples\VS-06\EIP demo. Open the EIP_demo.avp with AutoVision and download it to the camera.

During PLC integration, import the 32-000003-lx.L5X file to create the camera’s demo tags and ladder logic.

Glossary of Terms

The following terms are used in the description of di-soric demo PLC program.

Camera

The di-soric Smart Camera used in this application, which has an EtherNet/IP communication interface.

User App

The PLC logic code written by the end user or system integrator.

Demo Code

The PLC logic code distributed by di-soric that can be imported into the PLC's ladder logic area. It encapsulates most of the device Control and Status management.

The demo code expects a demo vision job loaded on the camera. However, the demo code will operate whether or not the demo vision job is loaded on the camera.

Activate / Set High

Writing a 1 value to a single Control bit, or any other bool bit.

Active

A Control, Status, bool, or PLC logic "contact" in a 1 state.

Clear

A Control, Status, bool, or PLC logic "contact" in a 0 state.

One Shot

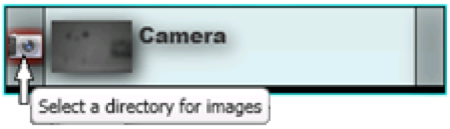
PLC tag write operation that is performed once, typically in reaction to an event. After a one shot operation, the PLC logic does not write to the same tag again unless another event occurs.

Demo Setup

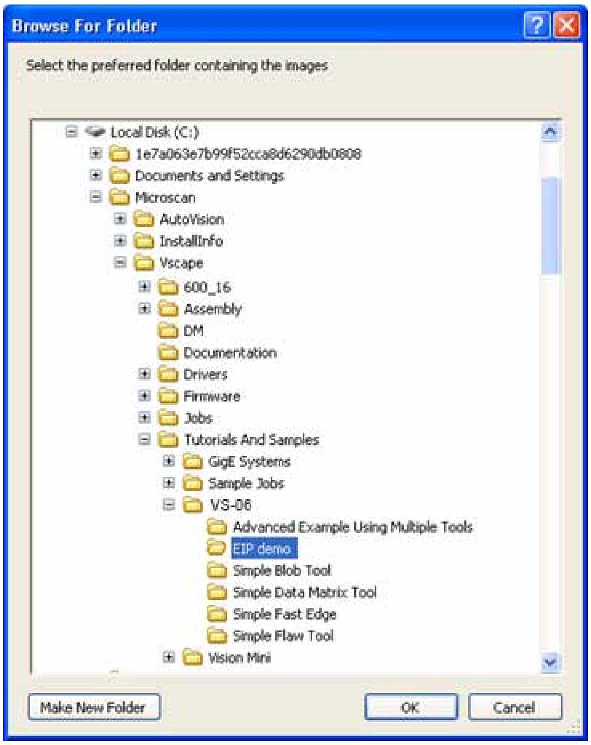
Vision Job Setup

The EIP demo files can be found where AutoVision is installed, where the default folder is C:\di-soric\Vscape\Tutorials and Samples\VS-06\EIP demo.

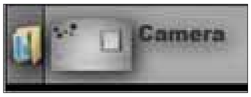
1. Open EIP_demo.avp with AutoVision.
2. To use pre-defined images, select the camera icon on the Camera tool.



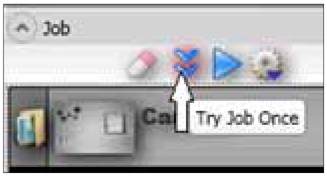
3. Browse to the EIP demo folder, select it, and click OK.



After the EIP demo folder has been enabled for image load, the camera icon will change to a folder:

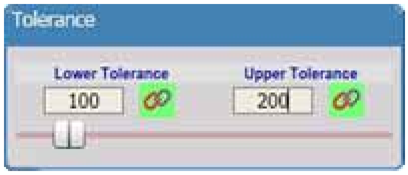


4. While in Edit mode, Try Out can be used to get an understanding of what to expect after the job is sent to the camera.



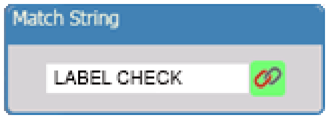
Before Try Out can be effective, the Measure, Decode, and Count Tool parameters must be specified. After job download, the tool parameters will be supplied by the PLC.

Measure Tolerance:



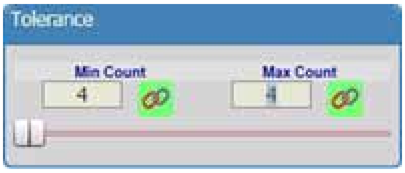
Lower Tolerance: 100
Upper Tolerance: 200

Decode Matchstring:



LABEL CHECK

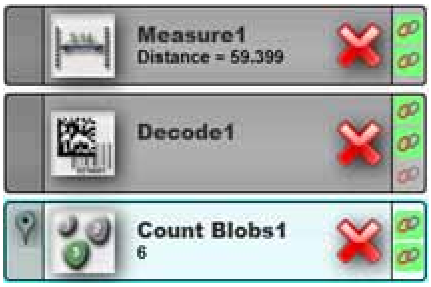
Count Tolerance:



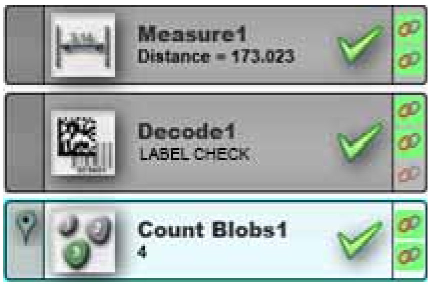
Min Count: 4
Max Count: 4

With these tool parameter configured as shown, Try Out will show the following Pass/Fail results.

Fail:



Pass:



5. Download the job to the camera.

6. Add the camera and demo code to the PLC environment (see the next section).

PLC Demo Code Setup

During PLC integration, import the 32-000003-lx.L5X file, found in the EIP demo folder, to create the camera's demo tags and ladder logic. Please refer to **Allen-Bradley PLC Setup**.

Description of PLC Tags

Read_MV_demo_mode

Purpose

This is intended for demonstration purposes only, to modify the operation of the demo code. It allows the first-time user to control the device directly with no assistance from the demo code, or allow the demo code to manage the Control and Status signals fully.

The demo mode tag takes three different values, putting the demo code into one of three modes of operation:

- Exchange IO data only
- Actively operate device controls, status, and demo data
- Automatically trigger the device after one second of idle time

User App Method

User app can set the demo mode with one of three values to define the demo code's mode of operation.

0=Exchange IO data only

In this mode, the user directly accesses the Read_MV_IO_internal tag set. The demo code only exchanges data with the camera, doing nothing to control the device or respond to events from it.

1=Operate device controls and respond to device events

This is the default mode of the demo code. In this mode, the user app accesses the Read_MV_IO_user tag to control and monitor the camera. The user app must not access the Read_MV_IO_internal tag set.

In this mode, the user activates the controls in Read_MV_IO_user.OUT.Control (Trigger, ResetCount, GoOnline, GoOffline, ResetError, ExeCmd), and the demo code handles the rest.

2=Auto-trigger

In this mode, the demo code fully manages the Control and Status signals, the same as when the mode is set to 1. It also activates the camera's trigger after one second of idle time. The timer used to drive the trigger is Read_MV_trigger_delay_timer.

Demo Code Usage

Depending on the mode, the demo code will run the appropriate level of code.

- In mode 0, only the IO exchange rungs are executed. All others are bypassed (ladder jmp).

- In mode 1, the auto-trigger rungs are bypassed. This is the default mode of the demo code.
- In mode 2, all rungs are executed.

Read_MV_IO_user

Purpose

User-accessible IO data for the camera. The user app reads and writes these IO tags, and the demo code handles the actual on-the-wire control of the camera.

User App Method

Activate a Control by setting its value to 1.

The user app can determine that the Control is done when the Control is clear (demo code changes the Control to a bit/bool value of 0). Do not attempt to activate a Control unless it is clear.

The user app should activate the Controls using one-shot writes. The use app should not continuously hold a Control in an active state. Holding a Control in an active state will prevent the demo code from notifying the user app that the Control operation is complete by clearing the Control.

Usually, when a Control is clear (0), the camera is ready for the Control to be activated again. Please see the **Specific Control Guidelines** and **Specific Status Guidelines** below for qualifications.

Demo Code Usage

The demo code waits for the user app to activate a Control. When the user app activates a Control, the demo code handles all handshaking and confirmation that the Control operation is performed by the camera. When the operation is complete, the demo code clears the Control back to 0.

VS-06 Specific Control Guidelines

GoOnline and GoOffline

In order to take the camera Online and Offline, only one of these Controls can go active (change from 0 to 1), and be active, at any given time.

ResetCount

After the user app activates ResetCount, the demo code will clear ResetCount when the operation is complete. The next Inspection output will be #1 (as can be seen if AutoVision is connected to the camera in run mode).

VS-06 Trigger

Do not trigger the camera unless the TriggerReady Status is active. If the Trigger goes active when TriggerReady is not active, the demo code increments the counter Read_MV_trigger_err_count, and immediately clears the Trigger Control, without attempting to trigger the camera.

After the user app activates the Trigger, the demo code will clear the Trigger when the camera indicates it has accepted the Trigger.

Do not re-trigger the camera until DataValid in the Status register goes active, all Inspection data has been processed, and the DataValid is cleared using the ResetDataValid Control.

ResetDataValid

When the user app sees DataValid go active, it should process the Inspection data, then clear DataValid by activating ResetDataValid.

See **Data Valid** for more details.

ResetError

To clear the Error Status, activate ResetError.

ExeCmd, CmdCode, CmdArg

These Controls can be used to perform a job change, and query the active job slot. Refer to the CmdCode section of the EIP chapter for available command codes, command result codes, and a diagram of command execution. The demo code includes tags with pre-defined **CmdCode** and **CmdCodeRslt** definitions:

Tag	Meaning
Read_MV_const_**CmdCode**_GetActiveJob	Request active job#
Read_MV_const_**CmdCode**_JbChg	Change job, stay offline
Read_MV_const_**CmdCode**_JbChg_Online	Change job and go online
Read_MV_const_**CmdCode**_JbChg_MkBt	Change job, make it the boot job, stay offline
Read_MV_const_**CmdCode**_JbChg_MkBt_Online	Change job, make it the boot job, go online
Read_MV_const_**CmdCodeRslt**_Success	Command completed successfully
Read_MV_const_**CmdCodeRslt**_Fail_No_Job	Job change command requested an empty slot
Read_MV_const_**CmdCodeRslt**_Fail_UI	Camera is controlled by AutoVision or FrontRunner.
Read_MV_const_**CmdCodeRslt**_Fail_Unk_Cmd	Unknown command

The ExeCmd, CmdCode, and CmdArg controls are used in combination with these Status signals:

Control signal	Status signal
ExeCmd	ExeCmdAck
CmdCode	CmdCodeRslt
CmdArg	CmdRet

The demo code records the final result of the command operation by copying **CmdCode**, **CmdArg**, **CmdCodeRslt** and **CmdRet** to the following tags:

Source Control/Status tag	Final result tag
Read_MV_IO_user.OUT.Control.**CmdCode**	Read_MV_**CmdCode**_last
Read_MV_IO_user.OUT.Control.**CmdArg**	Read_MV_**CmdArg**_last
Read_MV_IO_user.IN.Status.**CmdCodeRslt**	Read_MV_**CmdCodeRslt**_last
Read_MV_IO_user.IN.Status.**CmdRet**	Read_MV_**CmdRet**_last

The demo code will automate the command process when Read_MV_demo_mode is 1, which is the default value at program startup, similar to how it assists the Triggering and DataValid Controls. The PLC integrator can initiate command operation by accessing the demo code’s Read_MV_IO_user tag set for Control and Status signals. While a command operation is active, the demo code forces all Control signals to an inactive state, except for the Echo. No Controls can be activated until the command operation is completed. To verify the camera is still “alive” during command execution, the Control.Echo can be incremented, and the Status.Echo will update accordingly.

When the demo code automates the command process, the PLC integrator is responsible for the following steps:

1. Deactivate all Controls and clear DataValid and Error status signals. This is a “best practice” measure, to ensure that the PLC has transitioned from a state of triggering and processing inspections, to issuing a command.

2. If a job change command is to be issued, populate the output tags required to configure the new job (bool, int, long, float, string).

3. Write the required CmdCode (see Read_MV_const_CmdCode_xxxx tags) and CmdArg, then activate ExeCmd.

4. Wait for ExeCmd to go inactive (per typical demo mode 1 operation). Note that job changes can take up to a minute. While a job change command is being executed, the Status.State tag will be 2.

5. When ExeCmd goes inactive, verify the following:

   Read_MV_CmdCodeRslt_last is 0 (Success)

   Read_MV_CmdRest_last contains the returned data from the command (if any)

   Status.State has changed to 0 (Offline) or 1 (Online)

   ExeCmdAck is inactive (0)

   Status.Error is inactive (0)

6. Put the camera online (if necessary), and continue with normal runtime operation.

VS-06 Specific Status Guidelines

Online

The camera cannot be Triggered or generate Inspection data unless Online is active. See the GoOnline Control.

TriggerReady

Do not attempt to trigger the camera unless TriggerReady is active.

See the description of the **Trigger** for more details.

TriggerAck and ResetCountAck

Used by the demo code to complete the respective operations.

VS-06 DataValid

When DataValid goes active, the user app should process the Inspection data, then clear DataValid using the ResetDataValid control. This is handled, by the demo code in mode 1 and 2, as a demonstration for the user app.

If the camera's DataValid goes active, but the user app has not cleared a previous DataValid event, the demo code does not overwrite Read_MV_IO_user with new Inspection data. Instead, the demo code increments the counter Read_MV_dv_err_count. The new Inspection data remains stranded in the Read_MV_internal tag set, and is effectively lost.

Read_MV_trigger_count

Incremented by the demo code when a new trigger is issued to the camera over the EtherNet/IP interface (Trigger Control activated).

Read_MV_trigger_err_count

Incremented by the demo code if the user app attempts to trigger the camera when TriggerReady is not active.

Read_MV_dv_err_count

Incremented by the demo code when new Inspection data is received from the camera, but the user app has not cleared the previous DataValid.

Read_MV_status_err_count

Incremented by the demo code whenever the Error Status goes active.

Read_MV_demo_blob, Read_MV_demo_decode, Read_MV_demo_InspStat, Read_MV_demo_measure

Purpose

These tags record counts and min and max values of several EIP IN data members.

The demo code expects a demo vision job to be loaded on the camera, and a demo target to be in the camera's field of view. The demo PLC code will operate without the demo vision job being loaded on the camera. However, the data records will not be valid.

The demo vision job has the following data members linked to certain job tools:

IN

Bool1 = Measure status (pass/fail)

Bool2 = Decode+Matchcode status (pass/fail)

Bool3 = Blob count status (pass fail)

Long1 = Blob count

Float1 = Measure value

String1 = Decode text

OUT

Long101 = Blob count minimum count tolerance

Long102 = Blob count maximum count tolerance

Float101 = Measure lower tolerance

Float102 = Measure upper tolerance

String101 = Matchcode

Each tag set records the following data for each vision job tool result received in the Inspection report:

Tool Result	**Record tag**	**EIP IN tag**

Measurement Status	Read_MV_demo_measure.
	bool = last status (pass/fail)
	pass_count = count of passes
	fail_count = count of fails
		Read_MV_user_IN.bool.bool1

Measurement	Read_MV_demo_measure.
	float = last value
	float_max=max value recorded
	float_min=minimum value recorded
		Read_MV_user_IN.float.float1

Decode Text Status (matchcode)	Read_MV_demo_decode.
	bool = last status (pass/fail)
	pass_count = count of passes
	fail_count = count of fails
		Read_MV_user_IN.bool.bool2

Decode Text	Read_MV_demo_decode. string= text of the last barcode decode attempt (nul if noread)	Read_MV_user_IO.string.string1

Blob Status	Read_MV_demo_blob.
	bool = last status (pass/fail)
	pass_count = count of passes
	fail_count = count of fails
		Read_MV_user_IN.bool.bool3

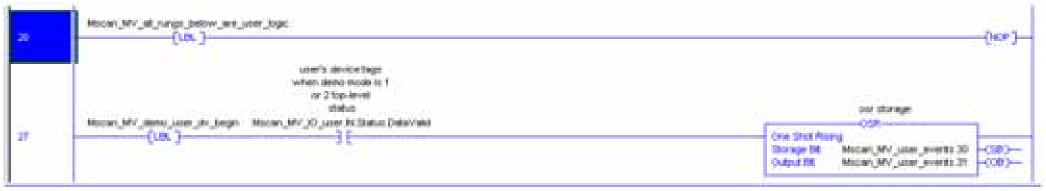
Blob Count	Read_MV_demo_blob.
	long=last value
	long_max=max value recorded
	long_min=minimum value recorded
		Read_MV_user_IN.long.long1

User App Method

The user app can follow the demo code’s usage of these tags for further application logic development.

During runtime, the user app can change the OUT data members, and observe the change in tool status after a new trigger.

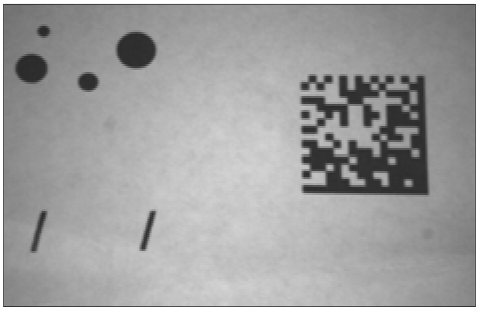
Specifically, the PLC integrator would typically modify the logic beginning at the following rungs:


26
Mscan_MV_all_rungs_below_are_user_logic
LBL
NOP
user's device tags when demo mode is 1 or 2 top-level status
27
Mscan_MV_demo_user_sv_begin
LBL
Mscan_MV_IO_user.In.Status.DataValid
osr storage
OSR
One Shot Rising
Storage Bit
Mscan_MV_user_events.30
Output Bit
Mscan_MV_user_events.31
OSB
OB

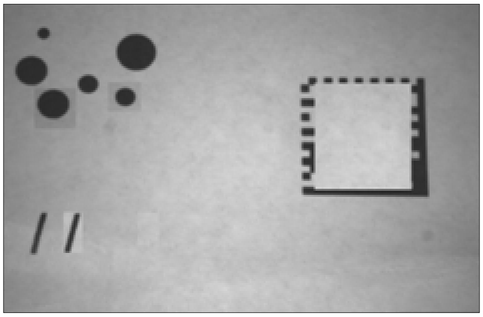
Demo Target

The demo vision job uses predefined images. It is unnecessary to have the camera aimed at any specific target. If preferred, the job can be changed to enable the camera’s image sensor. In this case, the demo targets should be printed approximately 2.5 inches (63 mm) wide by 1.6 inches (40 mm) tall, centered on white paper larger than the camera’s field of view, and presented to the camera with the Data Matrix symbol on the right:

“Pass” Image



“Fail” Image



Demo Code Usage

The user app example of the demo code watches for Data Valid. When it goes active, the user app example processes the user IO data, updates each demo record with the results, then uses ResetDataValid to clear DataValid.

Read_MV_IO_internal, Read_MV_ons_internal

Purpose

Used by the demo code to manage the camera.

User App Method

None. The user app must not attempt to read or write to this tag set.

Demo Code Usage

The demo code uses this tag set to abstract the on-the-wire control of the camera from the user app.

Run the Camera: Runtime Operation of EtherNet/IP Demo

At this point in the evaluation, it is assumed that you have downloaded the demo vision job to the camera, your PLC is running the EIP demo code and is exchanging data with the camera. The PLC can now parameterize, trigger and monitor the camera over EtherNet/IP.

H

APPENDIX H

Serial Commands

This section provides descriptions of the serial commands that can be sent to the camera via TCP (Telnet) port, AutoVision Terminal, or HyperTerminal.

SET {tagname}{value}

Sets value of a global tag.

The tagname must correspond to one of the supported tags within the device.

The value can contain spaces.

The command is terminated by a carriage return and/or line feed character.

The value can be a list of comma-separated items to set a sequence of tags:

Send **SET int1 1, 2, 3** to set int1 = 1, int2 = 2, int3 = 3.

The AVP service allows setting of step and datum information from the job tree using forward slash '/' in the symbolic name path. **SET avp/insp1/snapshot1/acq1/gain 2.0** paths are not case-sensitive and do not need to be fully qualified if unique.

SET avp/acq1/gain 2.0 will set the same gain value if there is only one acquire.

Control tags in the AVP service such as **START**, **STOP**, and **TRIGGER** act as momentary switches. **SET avp.start 1** is equivalent to the **ONLINE** command. **avp.start** will reset immediately and always read as **0**.

Success Return: On success will return **!OK** followed by an echo of the command. For example:

!OK SET matchstring1 ABCD

Fail Return: On failure will return **!ERROR** followed by the reason for the failure. For example:

!ERROR Tag matchstring66 not found

GET {tagname}

Gets value of a global tag.

The tagname must correspond to one of the supported tags within the device.

The command is terminated by a carriage return and/or line feed character.

Include an index to get a single value from an array such as **GET int1**. If the index is omitted, the full array of values will be returned in a comma-separated list of values.

Send **Get {tagname}** to get the value of a tag within the global data service. To get the value of a tag within another service, prefix the tagname with the service name. For example, a **GET {service.tagname}** command such as **GET eip.input** for the EIP input assembly.

The AVP service allows retrieval of step and datum information from the job tree using forward slash '/' in the symbolic name path. **GET avp/insp1/snapshot1/status** paths are not case-sensitive and do not need to be fully qualified if unique.

GET avp/snapshot1/status will return the same result if there is only one inspection.

When issued against a step, **GET avp/snapshot1** will return the values for all datums.

Success Return: On success will return the value stored in the tag. For example:

ABCD

Fail Return: On failure will return **!ERROR** followed by the reason for the failure. For example:

!ERROR Tag matchstring66 not found

INFO {service.tagname or service}

Gets information about a tag or service.

INFO with no arguments gets a list of services.

INFO {service} gets a list of tags in that service.

INFO {service.tagname} gets attributes of the tag as well as a list of subtags.

The AVP service allows retrieval of step and datum information from the job tree using forward slash '/' in the symbolic name path. **INFOavp/insp1/snapshot1/status** paths are not case-sensitive and do not need to be fully qualified if unique.

INFO avp/snapshot1/status will return the same result if there is only one inspection.

When issued against a step, **INFO avp/snapshot1** returns properties of the step, a list of child datums, and a list of child steps. Child steps are indicated by a trailing forward slash.

GETIMAGE {-transfer=ymodem} {-type=failed}{-format=[jpg|png]} {-quality=n} {-inspection=n} {woi=l,t,r,b}

Initiates serial transfer of inspection image.

-transfer=ymodem is not currently optional - only Ymodem protocol is supported.

-type=failed to retrieve the last failed image. If omitted, the current image is returned.

-format=[jpg|png] specifies the format of the image. If omitted, the image format is JPG.

-quality=*n* specifies a JPG compression quality of *n* less than or equal to 100. The default quality is **80** if not specified.

-inspection=*n* specifies the inspection from which to retrieve an image. The image will be from the first snapshot within that inspection. If not specified, the image will be from the first inspection that does contain a snapshot.

woi=left,top,right,bottom specifies a rectangular area of the image to be included in the output image. If omitted, the full image buffer is returned.

ONLINE

Starts all inspections.

OFFLINE

Stops all inspections.

TRIGGER {inspection index}

Triggers inspection. If {inspection index} is omitted, inspection 1 is triggered.

VT (Virtual Trigger) Command

Triggers an inspection by pulsing a Virtual I/O point. For example:

VT 1

will return pulse **VIO1**. The inspection will run if it is configured to use **VIO 1** as a trigger.

Syntax: VT [VIO Index]

- If specified, the VIO index must be in the allowed range for Virtual I/O points within Visionscape. The virtual I/O line will be set high then low.
- If VIO Index is not specified, VIO1 is assumed

Success Return: Nothing is echoed on success of the VT command.

Fail Return: Return **!ERROR** followed by the reason for the failure. For example:

!ERROR No such trigger

JOBSAVE [-slot=]n

Save job to slot *n.*

JOBLOAD [-slot=n][-r]

Load job from slot *n*.

-r = Start inspections.

JOBDELETE [-slot=n]

Delete job in slot *n.*

JOBINFO [[-slot=]n][-v]

Get job summary or info about slot *n*.

-v = Verbose. This option shows the amount of space that would be freed if the job were deleted. It also lists the total disk space and free disk space.

JOBBOOT {-slot=n}

Set bootup job slot *n.*

JOBDOWNLOAD [-transfer=]{YMODEM}

Download .avz job packaged via transfer method.

JOBDELETE -all

Delete all jobs in job slots.

Important: Does not delete the current job loaded in camera memory.

GET SYSTEM.JOBSLOT

Retrieve the slot of the current job. Note that the current job in the camera can be loaded from a job slot or the PC. If it isn’t loaded from a job slot then this command will return **-1**.

APPENDIX I

Boot Modes

This section describes the VS-06's Diagnostic Boot Mode and Boot Error Mode.

Diagnostic Boot Mode

The VS-06 supports a special boot mode used for diagnostics and recovery. There are two ways in which the camera can be put into this mode:

1. This method requires an Ethernet connection between the host PC and VS-06. Power-on the unit and hold the AutoVision button down until the green flash illuminates once. For C-Mount versions, hold the button down for approximately 30 seconds. The unit is now configured for IP address 192.168.0.10 with subnet mask 255.255.255.0. Establish a telnet connection between the host PC and VS-06. The [SAFE-KERNEL] prompt is displayed.
2. This method requires a VSID-IB-ES and a serial connection between the host PC running a terminal emulator and VS-06 camera. Power-on the unit and hold down the Tab key down for several seconds. The unit will boot to a [SAFE-KERNEL] prompt with communication settings of 115200, N, 8, 1 (baud, parity, data bits, stop bits).

Once the unit is booted, there are many possible actions the user can take. However, the most useful actions are listed below.

In rare situations, the boot job executed at camera startup can cause unexpected behavior. If this is the suspected case, it is possible to disable loading and running of the boot job at startup using the following command.

[SAFE-KERNEL] BP_UpdateStartupOptions(0, 0)

Note that the loading and running of the boot job is automatically re-enabled the next time a job is saved to camera flash from AutoVision or FrontRunner.

At boot time, the system configures itself using a set of information known as boot parameters. To obtain a list of the current configuration's boot parameters, issue the following command:

[SAFE-KERNEL] BP_Dump()

Should your device need to be configured with different IP information, follow the example below and substitute the appropriate settings for IP address, subnet mask, and gateway address, respectively.

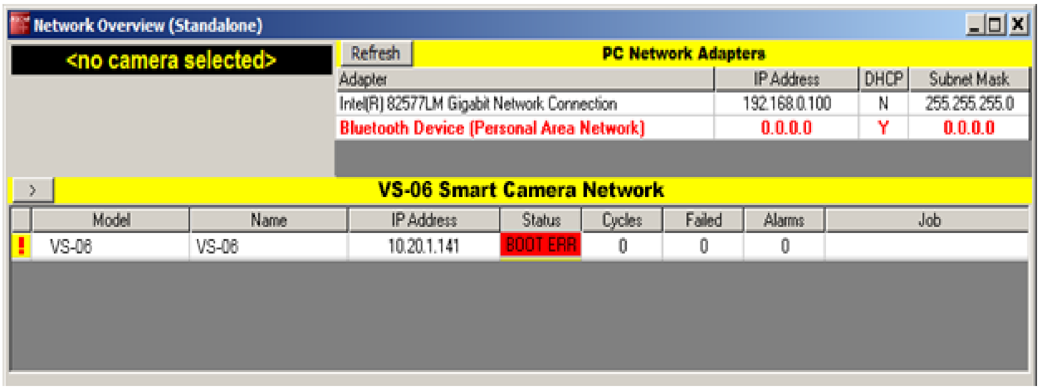
[SAFE-KERNEL] BP_UpdateIP(“192.168.0.10”, “255.255.255.0”, “192.168.0.100”)

It is possible to configure the system to acquire its IP address via DHCP or to use a static IP address. Issue the following command with a ‘0’ for static IP or a ‘1’ for DHCP.

[SAFE-KERNEL] BP_UpdateDHCP(0)

Boot Error Mode

The VS-06 enters an error mode on boot if it’s unable to fully load Visionscape. This mode is visually displayed to the user by flashing the Error LED along with the OUTPUT 1, OUTPUT 2, and OUTPUT 3 LEDs on the front of the unit. Additionally, this mode is represented as a “BOOT_ERR” in the Network Overview tool.



Adapter	IP Address	DHCP	Subnet Mask
Intel(R) 82577LM Gigabit Network Connection	192.168.0.100	N	255.255.255.0
Bluetooth Device (Personal Area Network)	0.0.0.0	Y	0.0.0.0

	Model	Name	IP Address	Status	Cycles	Failed	Alarms	Job
!	VS-06	VS-06	10.20.1.141	BOOT ERR	0	0	0	

If you encounter this error condition, you will need to reload the firmware using the Smart Camera Firmware Update Tool.