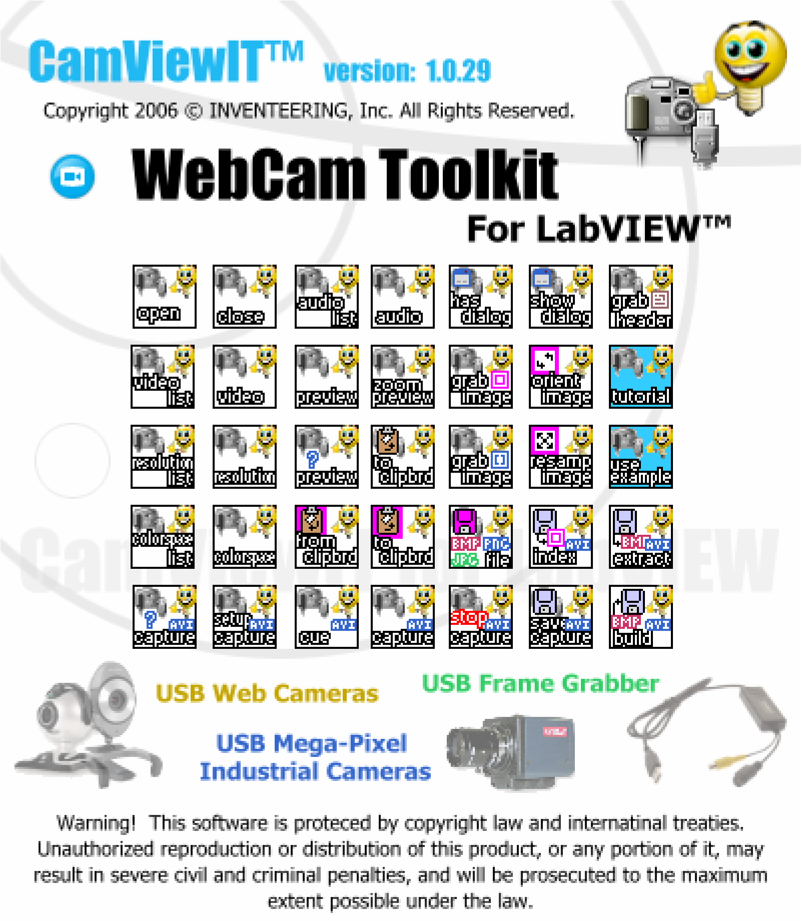
CamViewIT VIs and Functions

Refer to the CamViewIT help files, which you can access from the **Help** menu, for more information about those VIs. Use the CamViewIT VIs with DirectShow compatible video/audio devices to acquire and display images. Most USB web-cameras, USB analog video frame grabbers, and a few PCI bus analog video frame grabbers have a DirectShow compatible Windows Driver that will work with CamViewIT.



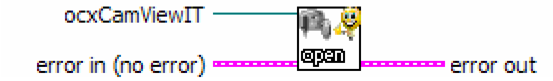
Copyright 2006 © INVENTEERING, Inc. All Rights Reserved.

Warning! This software is proteced by copyright law and internatinal treaties. Unauthorized reproduction or distribution of this product, or any portion of it, may result in severe civil and criminal penalties, and will be prosecuted to the maximum extent possible under the law.

CamViewIT.ctl

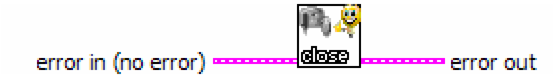
CamViewIT Control is an ActiveX Control **ocxCamViewIT**. In the block diagram this control is used as an input into **Open.vi** On the front panel the CamViewIT Control serves as Preview Box. This Preview Box will display a live video stream of a video device when the preview mode is set to True. The CamViewIT Control can be resized programmatically.
Also see **Open.vi**, **Preview. I**, **Previewing.vi**, and **ZoomPreview.vi.**

Open.vi



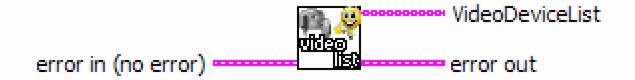
Initializes the CamViewIT library. Is the first VI that must be called before calling any other CamViewIT library VIs. Also see **Close.vi.**

Close.vi



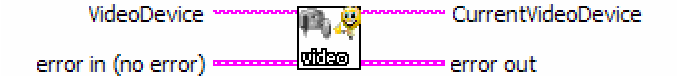
Shutdowns the connection to the CamViewIT library. Is the last VI that must be called when shutting down program. Also see **Open.vi.**

VideoDeviceList.vi



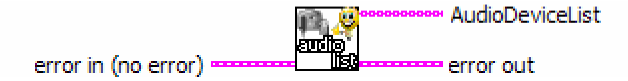
Returns a string array of DirectShow compatible video devices available on your PC. Input one of the devices strings in this array to the **VideoDevice.vi** and CamViewIT is then connected to that device.
Also see **VideoDevice.vi**

VideoDevice.vi



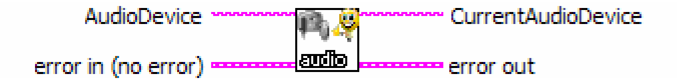
Connect to the DirectShow compatible video devices specified by the input string. If an empty “” string is input no devices is attempted to be connected, but if one is already connected it will return the current devices name in the output string. To disconnect from a device input “None” into the input string. Available DirectShow compatible video devices available on your PC can be retrieved by using **VideoDeviceList.vi.** Also see **VideoDeviceList.vi**

AudioDeviceList.vi



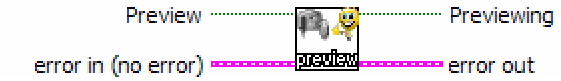
Returns a string array of DirectShow compatible audio devices available on your PC. Input one of the devices strings in this array to the **AudioDevice.vi** and CamViewIT is then connected to that device. Also see **AudioDevice.vi**

AudioDevice.vi



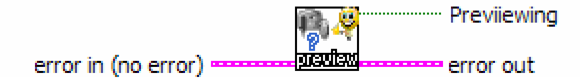
Connect to the DirectShow compatible audio devices specified by the input string. If an empty "" string is input no devices is attempted to be connected, but if one is already connected it will return the current devices name in the output string. To disconnect from a device input "None" into the input string. Available DirectShow compatible audio devices available on your PC can be retrieved by using **AudioDeviceList.vi.** Also see **AudioDeviceList.vi**

Preview.vi



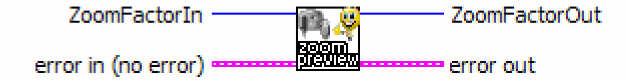
Starts or stops the video stream in the CamViewIT Control. The preview state is determined by the Boolean input. Preview must be on to use **GrabImage.vi** or **GrabImageToMatrix.vi**. To get the current state of Preview use Previewing.vi. Also see **GrabImage.vi**, **GrabImageToMatrix.vi**, **Previewing.vi**

Previewing.vi



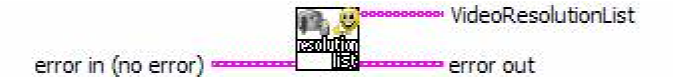
Gets the current state of Preview. Also see **Preview.vi**

ZoomPreview.vi

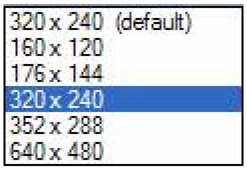


Zooms the streaming video in the CamViewIT Control. The Zoom Factor is a percentage of (1 to N) will retain the aspect ratio with in the CamViewIT Control and will not exceed the minimum dimension. The Zoom Factor set to 0 will stretch the streaming video to fit the CamViewIT Control and will not retain the aspect ratio. Also see **CamViewIT.ctl**

VideoResolutionList.vi

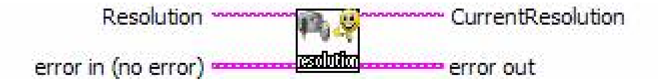


Returns a string array of resolutions available to the video device connected by the **VideoDevice.vi**. Input one of the resolution strings in this array to the **VideoResolution.vi** and output resolutions for the Preview and Capture modes will be changed to new resolutions. Some typical resolutions:



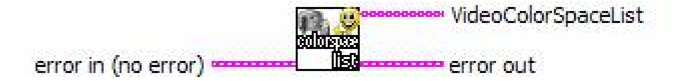
The Video Resolution List is the same those set by the **Video Capture Pin Dialog Window**.
Also see **VideoResolution.vi** and **VideoDevice.vi**.

VideoResolution.vi

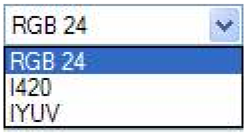


Sets the output resolutions for the Preview and Capture modes of the video device selected by the **VideoDevice.vi**. The new resolution will be returned in the CurrentResolution string. If an empty string is input to this VI, the current resolution will be returned in the CurrentResolution string.
Also see **VideoResolutionList.vi**.

VideoColorSpaceList.vi

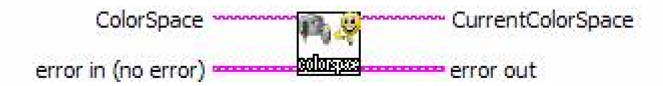


Returns a string array of color spaces available to the video device connected by the **VideoDevice.vi**. Input one of the resolution strings in this array to the **VideoColorSpace.vi** and output color space for the Preview and Capture modes will be changed to color space. Some typical color space:



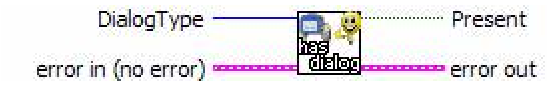
The Video Color Space List is the same those set by the **Video Capture Pin Dialog Window**.
Also see **VideoColorSpace.vi** and **VideoDevice.vi**.

VideoColorSpace.vi



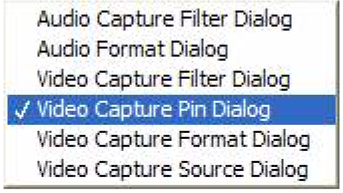
Sets the output color space for the Preview and Capture modes of the video device selected by the **VideoDevice.vi**. The new color space will be returned in the CurrentColorSpace string. If an empty string is input to this VI, the current color space will be returned in the CurrentColorSpace string. Also see **VideoColorSpaceList.vi**.

HasDialog.vi



Checks to see if video device connected by the **VideoDevice.vi** has a selected DialogType and returns a True or False in the Present BOOLEAN. These Dialog Windows are specific to the video device and may reflect specific features of only available to video device. There are six DialogType:

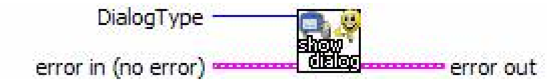
DialogType:



Audio Capture Filter Dialog
Audio Format Dialog
Video Capture Filter Dialog
✓ Video Capture Pin Dialog
Video Capture Format Dialog
Video Capture Source Dialog

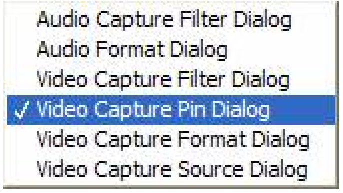
Also see **ShowDialog.vi**.

ShowDialog.vi



Show the Dialog Window specified by the input DialogType. The DialogType available is determined by running **HasDialog.vi**. These Dialog Windows are specific to the video device and may reflect specific features of only available to video device. There are six DialogType:

DialogType:

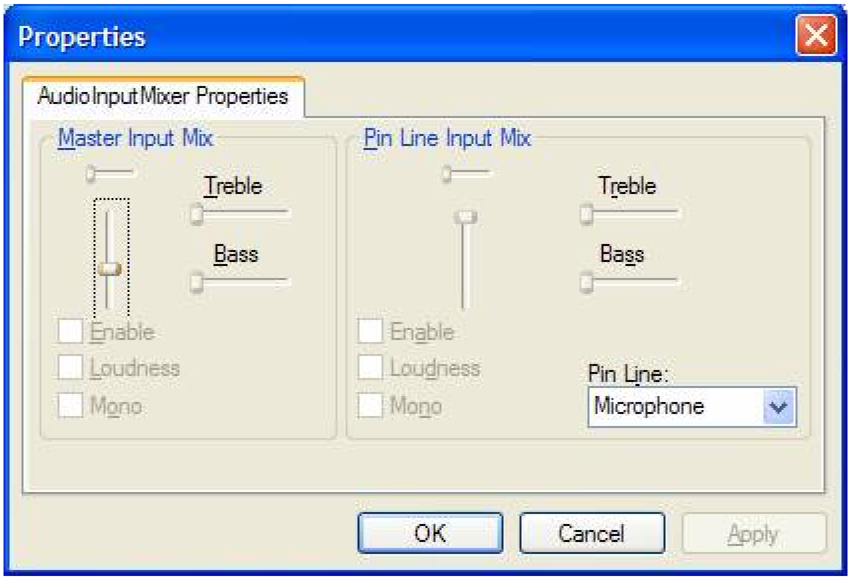


Audio Capture Filter Dialog
Audio Format Dialog
Video Capture Filter Dialog
✓ Video Capture Pin Dialog
Video Capture Format Dialog
Video Capture Source Dialog

Also see **HasDialog.vi**.

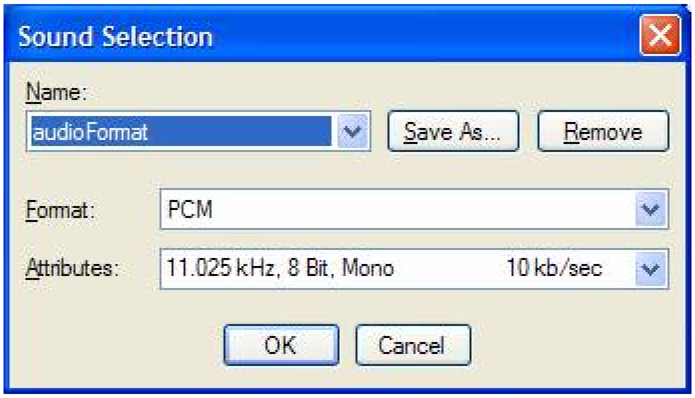
Here are some typical dialogs:

Audio Capture Filter Dialog



Pin Line: Microphone

Audio Format Dialog

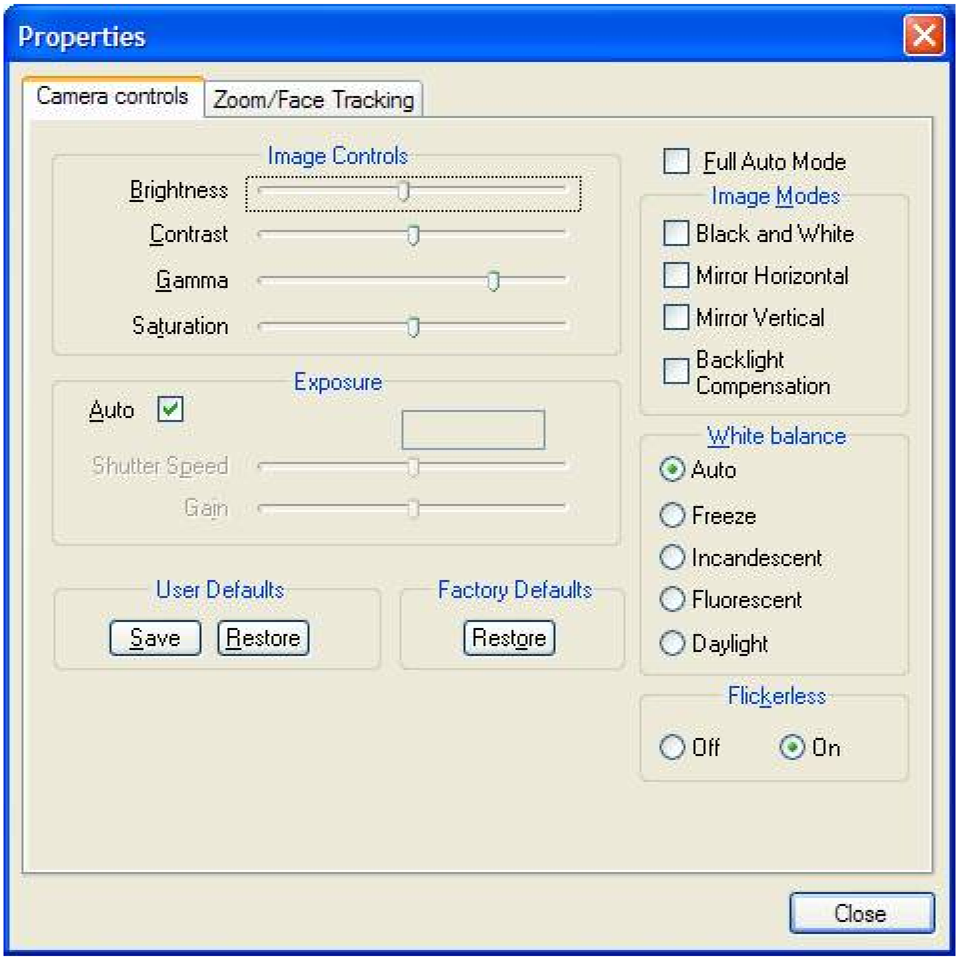


Name: audioFormat

Format: PCM

Attributes: 11.025 kHz, 8 Bit, Mono 10 kb/sec

Video Capture Filter Dialog



□ Full Auto Mode

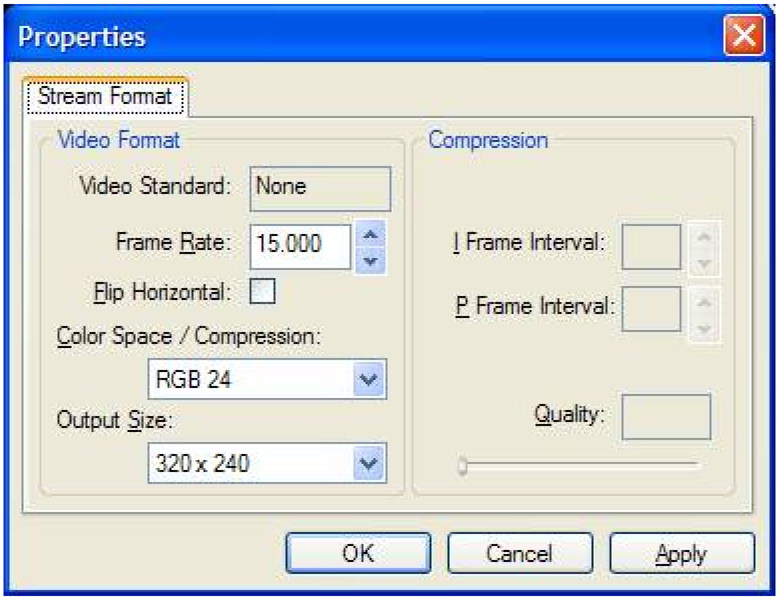
Image Modes: □ Black and White □ Mirror Horizontal □ Mirror Vertical □ Backlight Compensation

Exposure: Auto ☑

White balance: ● Auto ○ Freeze ○ Incandescent ○ Fluorescent ○ Daylight

Flickerless: ○ Off ● On

Video Capture Pin Dialog



Video Standard: None

Frame Rate: 15.000

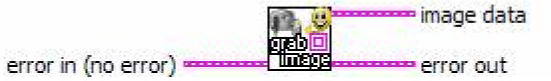
Flip Horizontal: □

Color Space / Compression: RGB 24

Output Size: 320 x 240

Video Capture Format Dialog and **Video Capture Source Dialog** are Video For Windows (VFW) legacy dialogs. Also see **HasDialog.vi**.

GrabImage.vi



Grab a frame from streaming video preview and loaded it into an **image data cluster**. For **GrabImage.vi** to work there must be a video device connected by the **VideoDevice.vi** and the **Preview.vi** must be set to True. The **image data cluster's** Rectangle show the video input resolution. This can also be retrieved using **GrabImageHeader.vi**.

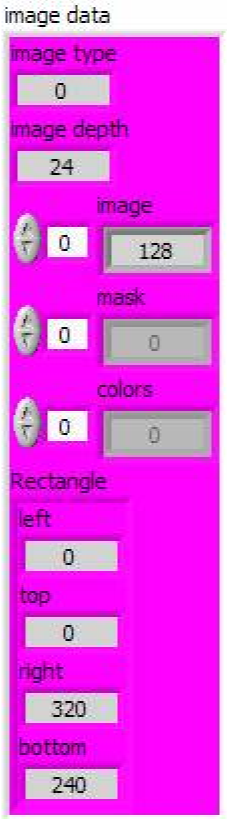
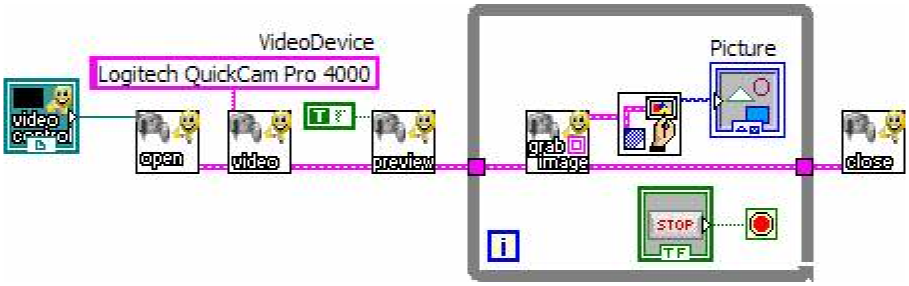


image data
image type 0
image depth 24
image 0 128
mask 0 0
colors 0 0
Rectangle
left 0
top 0
right 320
bottom 240

The grab rate will depend on the video devices interface type (USB 1.1 vs. USB 2.0 vs. PCI); and also will depends on CPU speed, input resolution and program continuity. A typical grab rate of about 10 to 15 grabs/sec has been achieved under the following conditions:

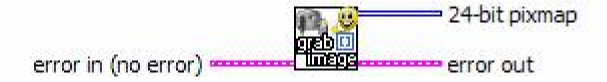
- o USB 1.1 Logitech QuickCam Pro 4000
- o 3GHz Pentium PC
- o 320x240 Resolution
- o Displayed in a Picture Control

This method is not the same as the Capture AVI method which is faster and captures both Video and Audio.

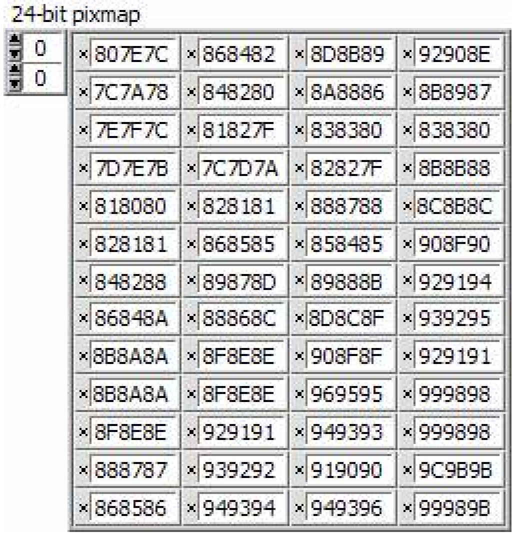


Also see **GrabImageToMatix.vi**, **GrabImageHeader.vi**, and **CaptureAVI.vi**.

GrabImageToMatrix.vi



Grab a frame from streaming video preview and loaded it into a **24-bit pixmap**. For **GrabImageToMatrix.vi** to work there must be a video device connected by the **VideoDevice.vi** and the **Preview.vi** must be set to True. The **24-bit pixmap** size dimensions are the video input resolution. This can also be retrieved using **GrabImageHeader.vi**.



24-bit pixmap

0
0

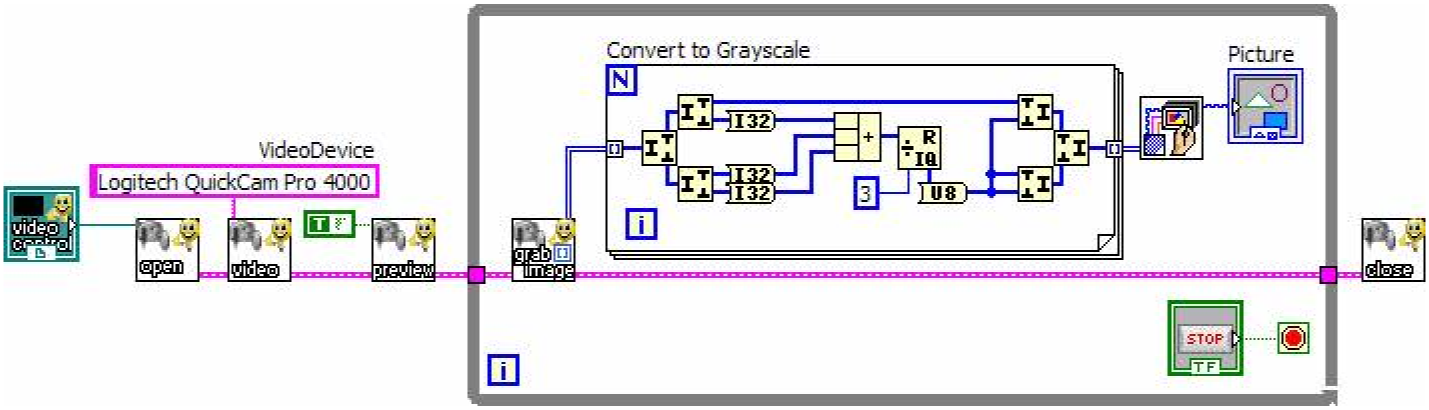
807E7C	868482	8D8B89	92908E
7C7A78	848280	8A8886	8B8987
7E7F7C	81827F	838380	838380
7D7E7B	7C7D7A	82827F	8B8B88
818080	828181	888788	8C8B8C
828181	868585	858485	908F90
848288	89878D	89888B	929194
86848A	88868C	8D8C8F	939295
8B8A8A	8F8E8E	908F8F	929191
8B8A8A	8F8E8E	969595	999898
8F8E8E	929191	949393	999898
888787	939292	919090	9C9B9B
868586	949394	949396	99989B

The grab rate will depend on the video devices interface type (USB 1.1 vs. USB 2.0 vs. PCI); and also will depends on CPU speed, input resolution and program continuity. A typical grab rate of about 10 to 15 grabs/sec has been achieved under the following conditions:

- o USB 1.1 Logitech QuickCam Pro 4000
- o 3GHz Pentium PC
- o 320x240 Resolution
- o Displayed in a Picture Control

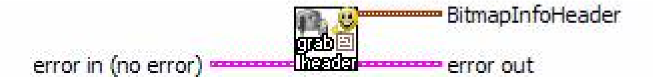
This method is not the same as the Capture AVI method which is faster and captures both Video and Audio.

GrabImageToMatrix.vi can be used with the **Image ArrayToColorImate.vi** to import data into **IMAQ**.

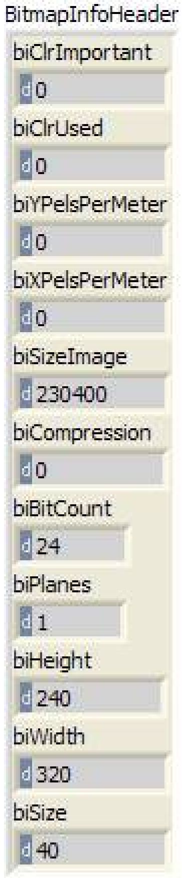


Also see **GrabImage.vi**, **GrabImageHeader.vi**, and **CaptureAVI.vi**.

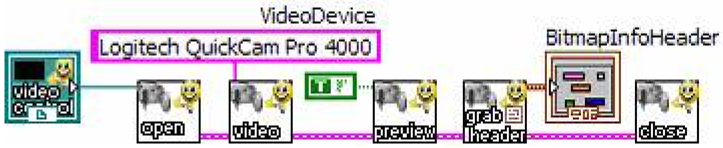
GrabImageHeader.vi



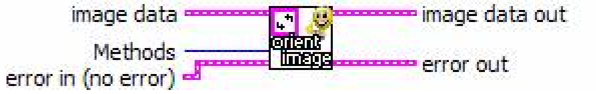
Grab Image Header from streaming video preview and loaded it into a **BitmapInfoHeader** Cluster. Also see **GrabImage.vi**, **GrabImageToMatrix.vi**


BitmapInfoHeader

biClrImportant 0
biClrUsed 0
biYPelsPerMeter 0
biXPelsPerMeter 0
biSizeImage 230400
biCompression 0
biBitCount 24
biPlanes 1
biHeight 240
biWidth 320
biSize 40

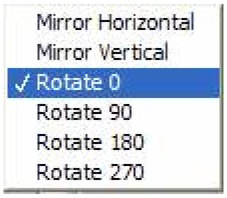


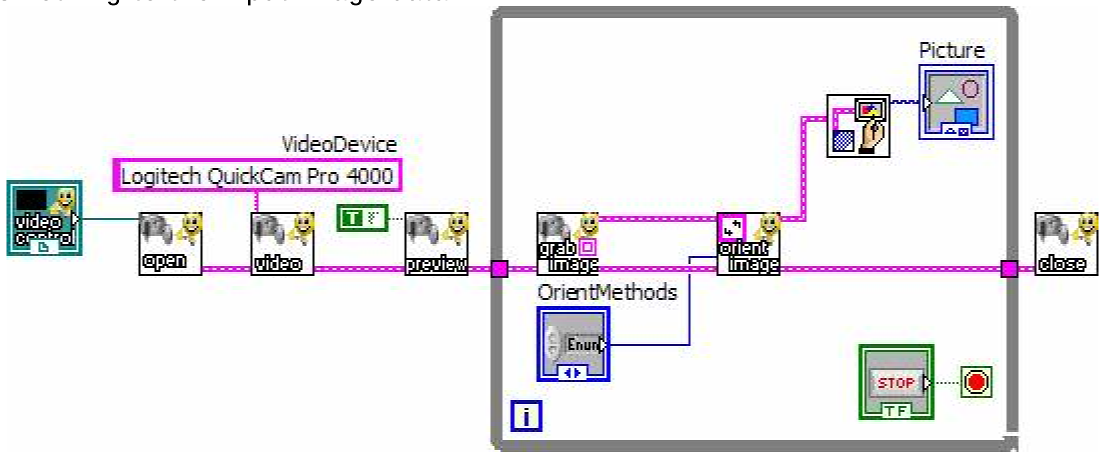
OrientImage.vi



This VI will orient image data according to the input enum **Methods** and outputs new image data. Rotate 0 actually does nothing to the input image data.

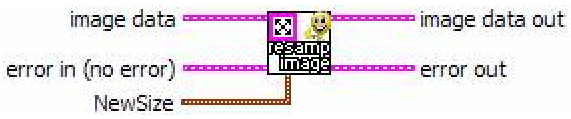
Methods are:



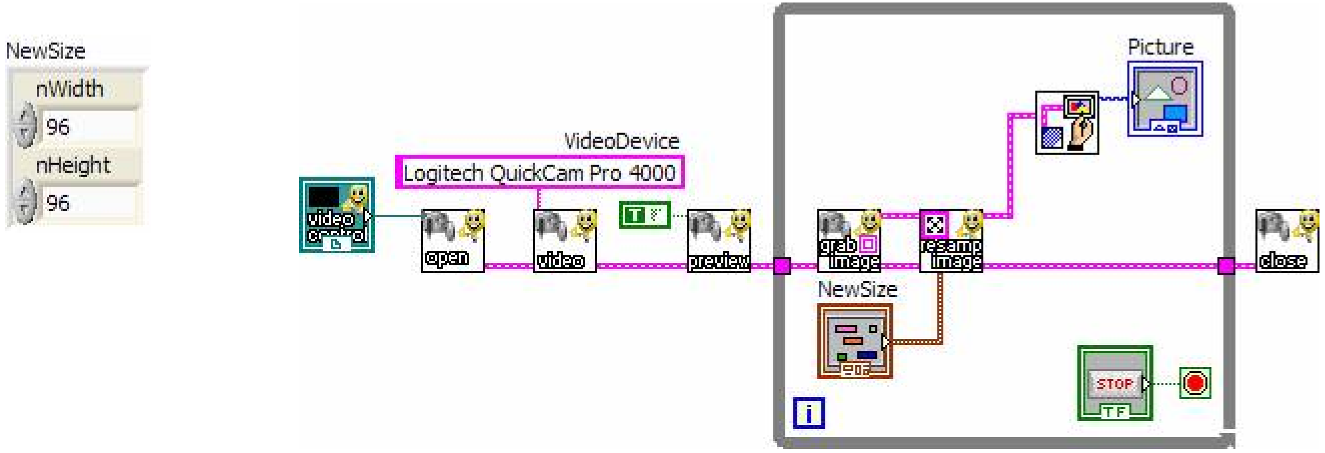


Also see **GrabImage.vi**, **ResampleImage.vi**

ResampleImage.vi

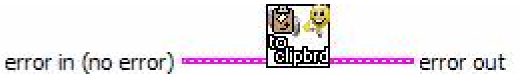


This VI will resample image data according to the input **NewSize** Cluster and outputs new image data. A ***Bicubic Algorithm*** is used to generate the output image data and produces very good looking images in both the reduced and expanded size. Aspect ratio is not automatic; if you want aspect ratio maintained you will need to do the math on what **NewSize** to input.

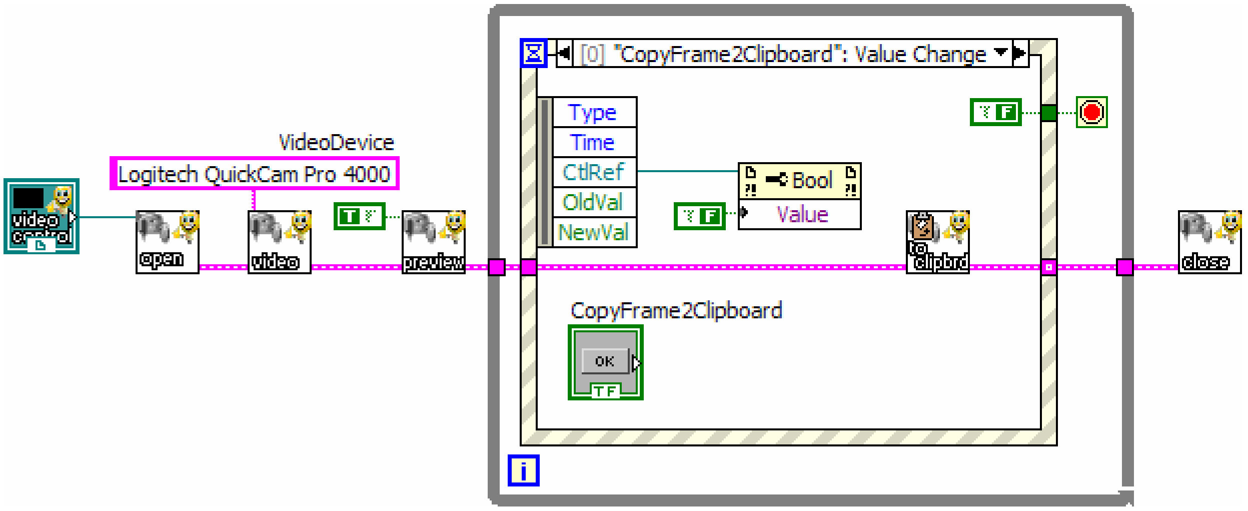


Also see **GrabImage.vi, OrientImage.vi**.

CopyFrameToClipboard.vi

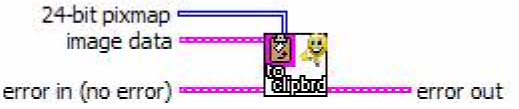


Copies a frame from streaming video preview into the systems clipboard as a standard bitmap image. For **CopyFrameToClipboard.vi** to work there must be a video device connected by the **VideoDevice.vi** and the **Preview.vi** must be set to True.

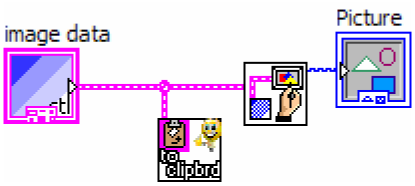


Also see **GrabImage.vi**, **CopyImageToClipboard.vi**, and **CopyImageFromClipboard.vi**.

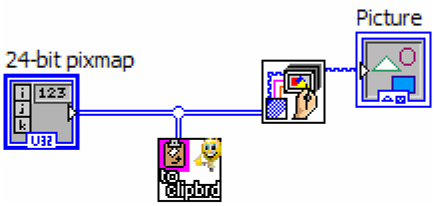
CopyImageToClipboard.vi



Copies an **image data cluster** or **24-bit pixmap** into the systems clipboard as a standard bitmap image. **CopyImageToClipboard.vi** does not require a video device connected by the **VideoDevice.vi**.

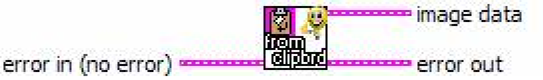


or

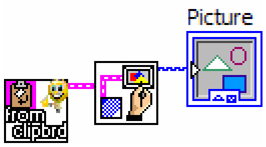


Also see **CopyFrameToClipboard.vi**, and **CopyImageFromClipboard.vi**.

CopyImageFromClipboard.vi

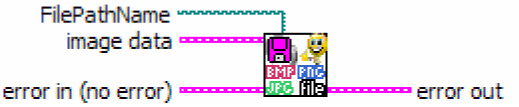


Copies an standard bitmap image from the systems clipboard to a **image data cluster**. **CopyImageFromClipboard.vi** does not require a video device connected by the **VideoDevice.vi**.



Also see **CopyFrameToClipboard.vi**, and **CopyImageToClipboard.vi**.

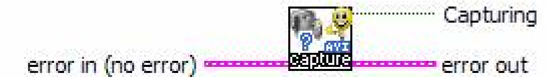
SaveImageAs.vi



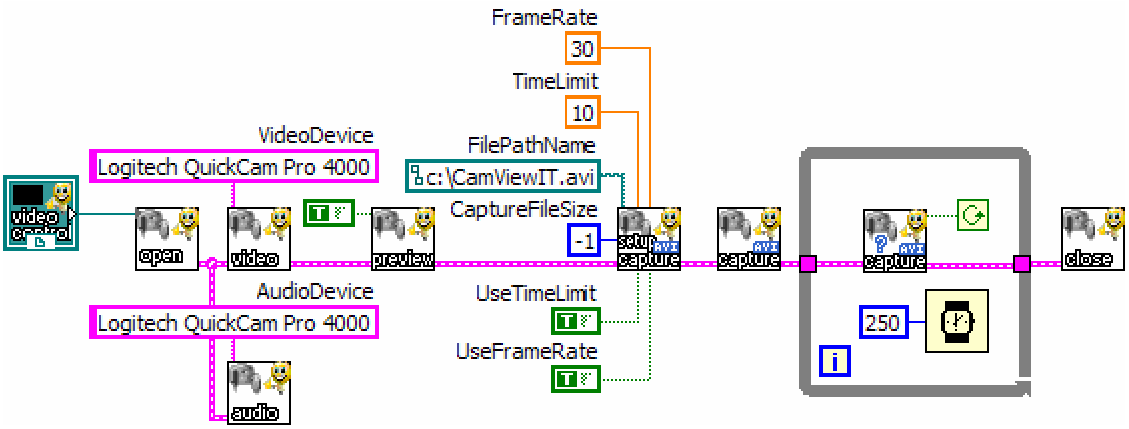
Saves an **image data cluster** to a file type that is automatically saved to one of the three types depending on the **FilePathName** extension used:

***.bmp ... Windows BITMAP Format**
***.jpg ... Joint Photographic Group Format**
***.png ... Portable Network Graphics Format**

Capturing.vi

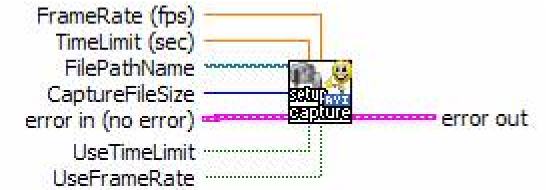


Gets the current state of AVI Capture. The following example would capture an AVI video file at 30 frames per sec for 10 seconds and then stop.



Also see **Capture.vi**

SetupCapture.vi



Sets up AVI capture parameters. This VI is required to run at least once before **Capture.vi**. The following are the input parameters:

FilePathName is the file built during the capture and should use the file extension (.avi).

CaptureFileSize pre-allocates disk space in Megabytes for the capture file. Is ignored if set to -1.

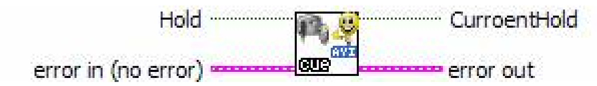
FrameRate (fps) is used to set the capture frame rate to a range (0 < FPS <= 30) if the **UseFrameRate** Boolean is set True.

TimeLimit (sec) is used to set the capture period if the **UseTimeLimit** Boolean is set True. The **StopCapture.vi** can override this parameter and stop the capture.

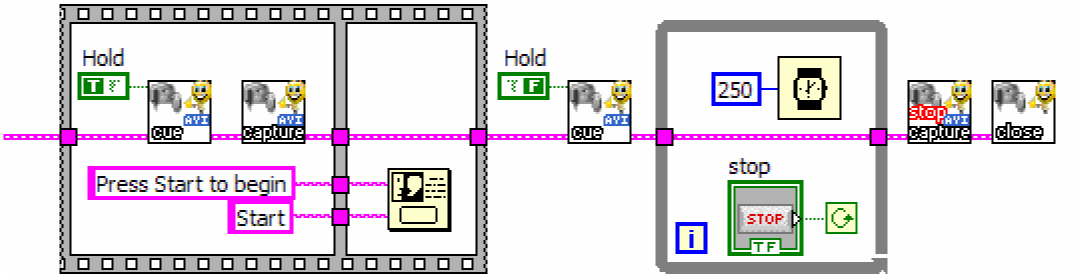
Factors like disk speed, CPU speed, bus type (USB 1.1, USB 2.0, PCI), and video device may determine the frame rate achieved while capturing AVI file.

Also see **Capture.vi** and **StopCapture.vi**.

CueCapture.vi

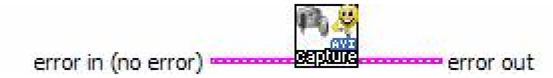


This VI is used to set an initial hold condition on an AVI capture. This is to allow the stream to stabilize and to allow video controls like AGC (automatic gain control) to stabilize. It also allows you to synchronize the beginning of your capture with an external event. To use this VI run it once with a **Hold** set **True** before a **Capture.vi** is run; and the again after with the **Hold** set **False** to release the hold condition.



Also see **Capture.vi** and **StopCapture.vi**.

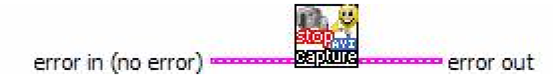
Capture.vi



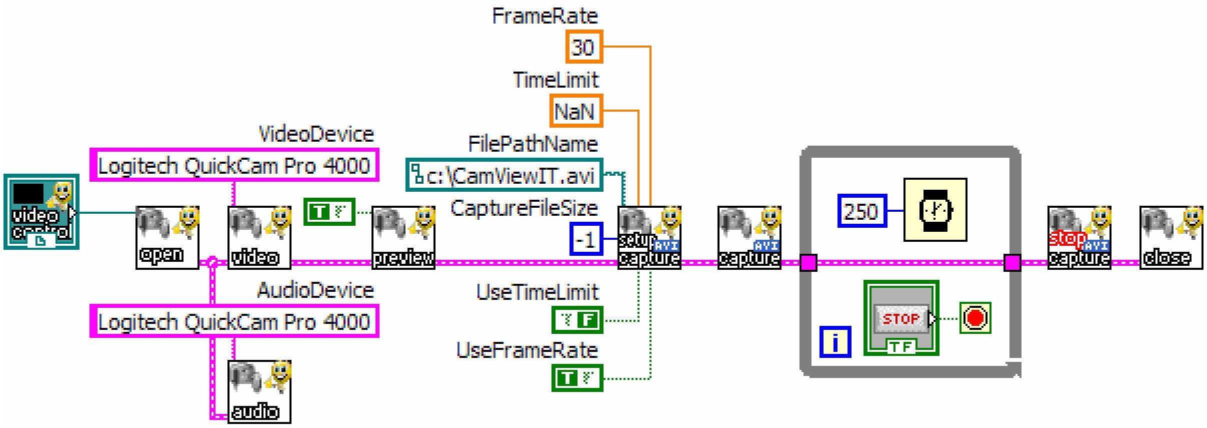
This VI is used to start the AVI capture. **CueCapture.vi** affects this VI function by setting and releasing a hold condition. **StopCapture.vi** also affects this VI function by stopping the capture.

Also see **SetupCapture.vi**, **CueCapture.vi** and **StopCapture.vi**.

StopCapture.vi

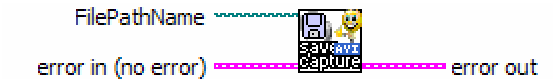


This VI is used to stop the AVI capture. The following example would capture an AVI video file 30 frames per sec until you press the Stop.



Also see **Capture.vi** and **CueCapture.vi**.

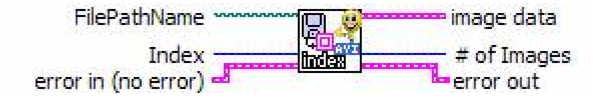
SaveCaptureFileAs.vi



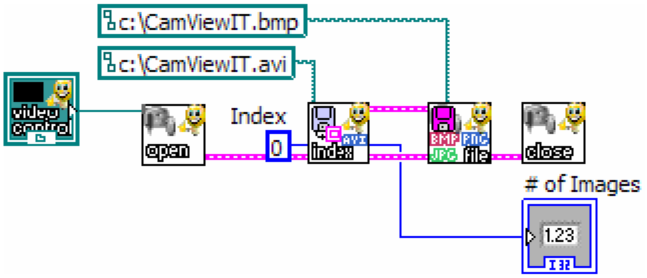
Saves the last AVI capture as the **FilePathName**. This VI is like a post-Save. For example, if you did a AVI capture that was saved to **c:\CamViewIT.avi** and afterward ran this VI and used a new **FilePathName**, like **c: \My Videos\LabExp1.avi**. It would be the same as coping **c:\CamViewIT.avi** to **c:\My Videos\LabExp1.avi**.

Also see **SetupCapture.vi**.

IndexAVI.vi

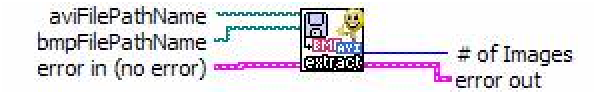


Returns the **image data cluster** of the AVI file specified by **FilePathName** at **index**. The following example will get index frame 0 from **c:\CamViewIT.avi** and save it to bitmap file **c:\CamViewIT.bmp**.



Also see **ExtractAVItoBMP.vi**.

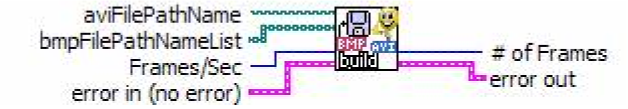
ExtractAVItoBMPs.vi



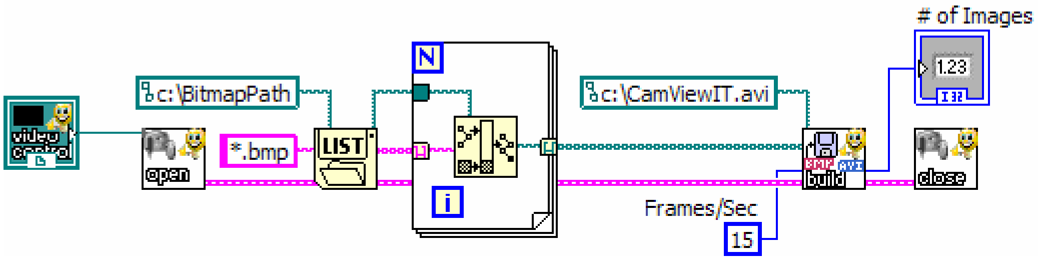
This VI will extract all the frames of an AVI file specified by **aviFilePathName** and write each frame as a bitmap file to the directory specified by **bmpFilePathName**. The bitmap files will be numbered sequentially with the root name being derived from **bmpFilePathName**. For example if **bmpFilePathName** is **c:\My_CamViewIT_BMPs\image.bmp**, the path would be **c:\My_CamViewIT_BMPs\** and the root name is **image.bmp** and the files will be named sequentially: **image1.bmp**, **image2.bmp**, **image3.bmp**, **imageN.bmp** where **N** would be **# of Images**.

Also see **BuildAVIfromBMP.vi**.

BuildAVIfromBMPs.vi



This VI is the opposite of **ExtractAVItoBMP.vi**. This VI will build an AVI file from a list of bitmap files specified by the **bmpFilePathNameList**. The AVI files frame rate will be specified by **Frames/Sec**.



Also see **ExtractAVItoBMP.vi**.

Tutorial.vi

error in (no error)
tutorial
error out

This tutorial VI consists of a large group of simple example uses of the functions in the CamViewIT Toolkit.

UserExample.vi

error in (no error)
use example
error out

This VI is the top level VI to a working example that uses all the functions in the CamViewIT Toolkit.