

# LabVIEW drivers for PixelFly Version 1.0

## Programmer's Reference and User Guide



The Cooke Corporation  
1091 Centre Rd. - Suite 100  
Auburn Hills, MI  
48326-2670

(248) 276 8820  
(248) 276 8825 fax

## CONTENTS

<b>1</b>	<b>Before you start .....</b>	<b>3</b>
<b>2</b>	<b>Installing the LabVIEW drivers .....</b>	<b>3</b>
<b>3</b>	<b>Overview of PixelFly operations .....</b>	<b>7</b>
3.1	Initialize the hardware and set up an exposure .....	7
3.2	Set up a buffer to store the image.....	8
3.3	Start and exposure and wait for the data to be ready .....	10
3.4	Get image data, free up buffers and terminate the program.....	11
<b>4</b>	<b>Error clusters and messages .....</b>	<b>14</b>
<b>5</b>	<b>Sample programs .....</b>	<b>15</b>
<b>6</b>	<b>Individual VI descriptions .....</b>	<b>16</b>
6.1	Driver setup functions.....	16
6.1.1	initboard.vi .....	16
6.1.2	closeboard.vi.....	17
6.2	Status functions.....	18
6.2.1	get parameters.vi .....	18
6.2.2	Get sixes.vi.....	20
6.2.3	gettemp.vi.....	21
6.2.4	Get buffer status.vi .....	22
6.3	Control functions.....	25
6.3.1	Set mode.vi.....	25
6.3.2	Start camera.vi .....	26
6.3.3	Stop camera.vi .....	27
6.3.4	Trigger.vi .....	29
6.4	Buffer functions .....	30
6.4.1	Allocate buffer.....	30
6.4.2	Map buffer.vi .....	31
6.4.3	Add buffer to list.vi .....	32
6.4.4	Remove buffer from list.vi.....	34
6.4.5	Unmap buffer.vi .....	35
6.4.6	Free buffer.vi .....	36
6.5	Image formatting functions .....	37
6.5.1	Colorpicture.vi .....	37
6.5.2	Get BW from buffer.vi.....	38
6.5.3	get color from buffer.vi.....	40
<b>7</b>	<b>Technical Support .....</b>	<b>41</b>

## 1. Before you start

Thank you for choosing the Cooke PixelFly for your imaging application. The PixelFly's high dynamic range, excellent sensitivity and programmable features, coupled with the ease of programming and processing power of LabVIEW make a powerful combination.

Before you try to implement your LabVIEW application, we recommend that you install your PixelFly hardware in the computer you intend to use. You should also install the camera software and drivers that came with your PixelFly. LabVIEW uses these drivers to control the camera, so they must be installed for your LabVIEW application to run. We also recommend that, once the hardware and software are installed, you perform a few simple tests on the camera using the CamWare software provided with your PixelFly, to confirm the camera is operating correctly before you attempt to control the camera from LabVIEW. A detailed manual is provided with your PixelFly software to guide you through the process of installing and running the CamWare software.

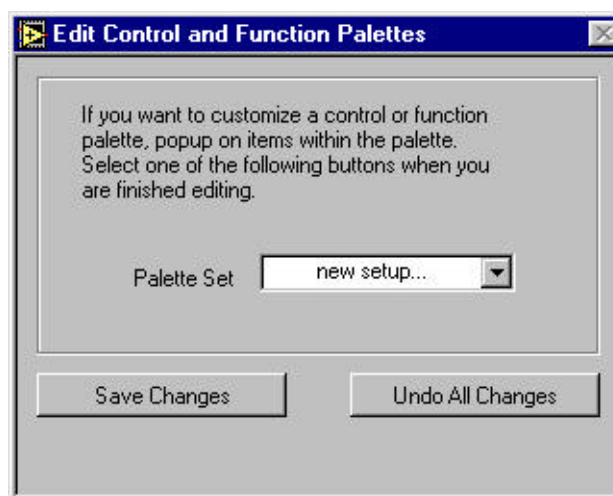
## 2. Installing the LabVIEW drivers.



Install the LabVIEW drivers by running the "SETUP.EXE" program on the CD provided. This will install the driver library in the "PIXELAB" directory, or another that you have specified. Now, run LabVIEW and open a new VI, and open the tools palette on the block diagram. The default tools palette will look similar to the one shown at left, depending on which LabVIEW components you have installed.

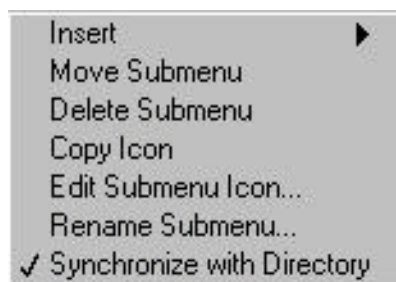


Use the “Edit Control and function Palettes...” option from the “Edit” menu to add the PixelFly functions to your controls palette. The following pop up window will appear:

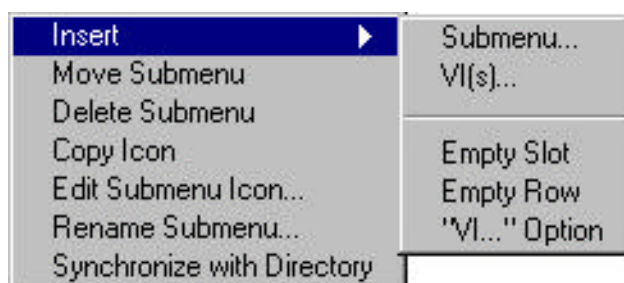


If you select “new setup...” you will be prompted for a name for the new palette configuration. This will allow you to customize your palette without affecting the default set-up. Otherwise, you can choose to modify an existing palette. Do not press the “Save Changes” button until you have added the PixelFly sub-palette.

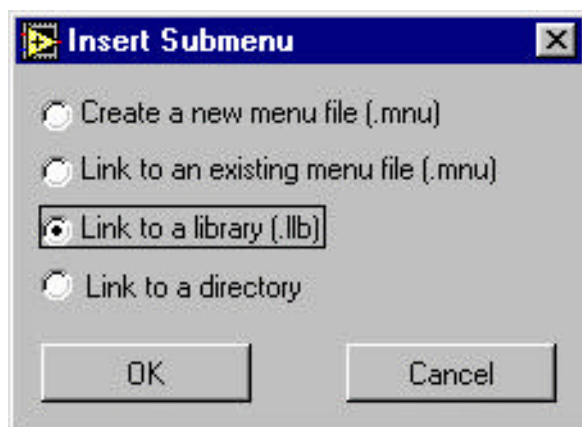
Right-click on any one of the sub-palettes on the function palette. You should see this menu appear:



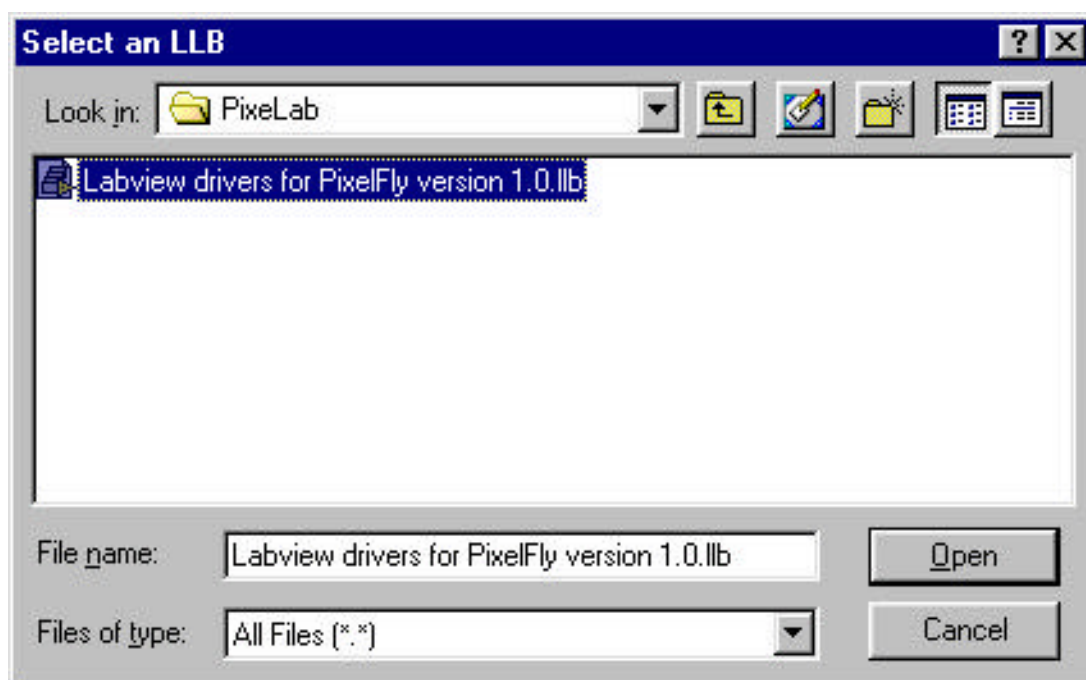
Select the option to insert submenu:



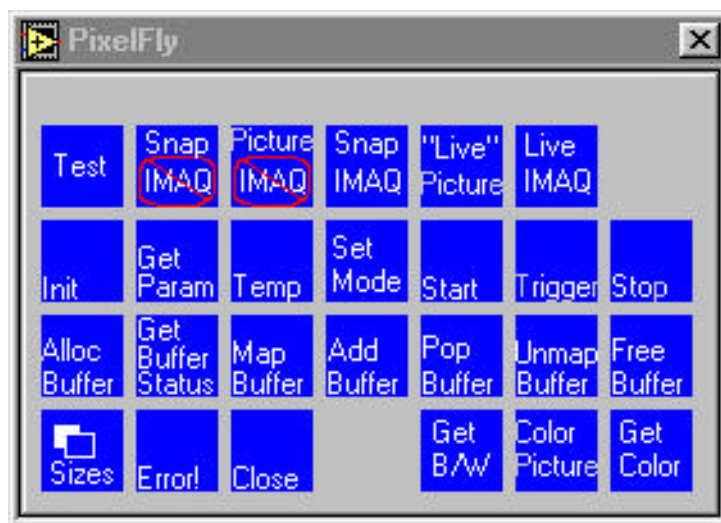
Select the “Link to a library” option



Browse your computer to find the “LabVIEW drivers for PixelFly version 1.0.llb”. The default location for installation of these files is the “PIXELAB” directory:



Click “Open”, then press the “Save Changes” button. This should create a sub-palette in the functions palette, similar to the one shown below. Clicking on the “PixelFly” icon should pop up the sub-palette with and display the PixelFly VI’s. “Use Ctrl-H” to bring up the on-line help. A brief description of each VI will appear when you position the cursor over it.



### 3. Overview of PixelFly operations

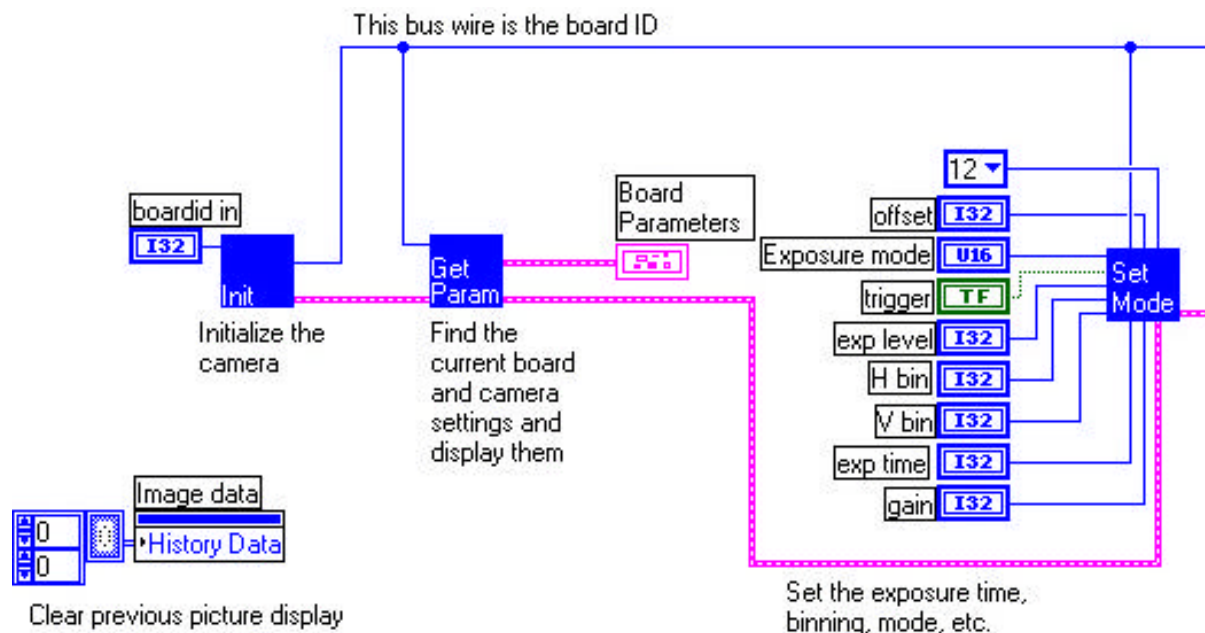
There are two major components to the PixelFly system: the PCI interface board and the camera head. LabVIEW communicates with the PCI interface board, which communicates with the camera head over a high-speed serial link.

The basic steps for operating the PixelFly are:

- Initialize the hardware
- Set up an exposure
- Set up a buffer to store the image
- Start an exposure
- When data is ready, retrieve and display it
- Free up the buffer
- Release the hardware

Each of these steps will be illustrate by examining the sample program “Take a single exposure without IMAQ.vi”, provided with the driver library.

#### 3.1. Initialize the hardware and set up an exposure:





In this example:



**Init** - “**initboard.vi**” – Initialize the camera. This VI establishes communications with the camera, and assigns it unique number. Calls to subsequent VI’s uses this number to determine which camera to act upon. Up to four cameras can be accessed in this way.

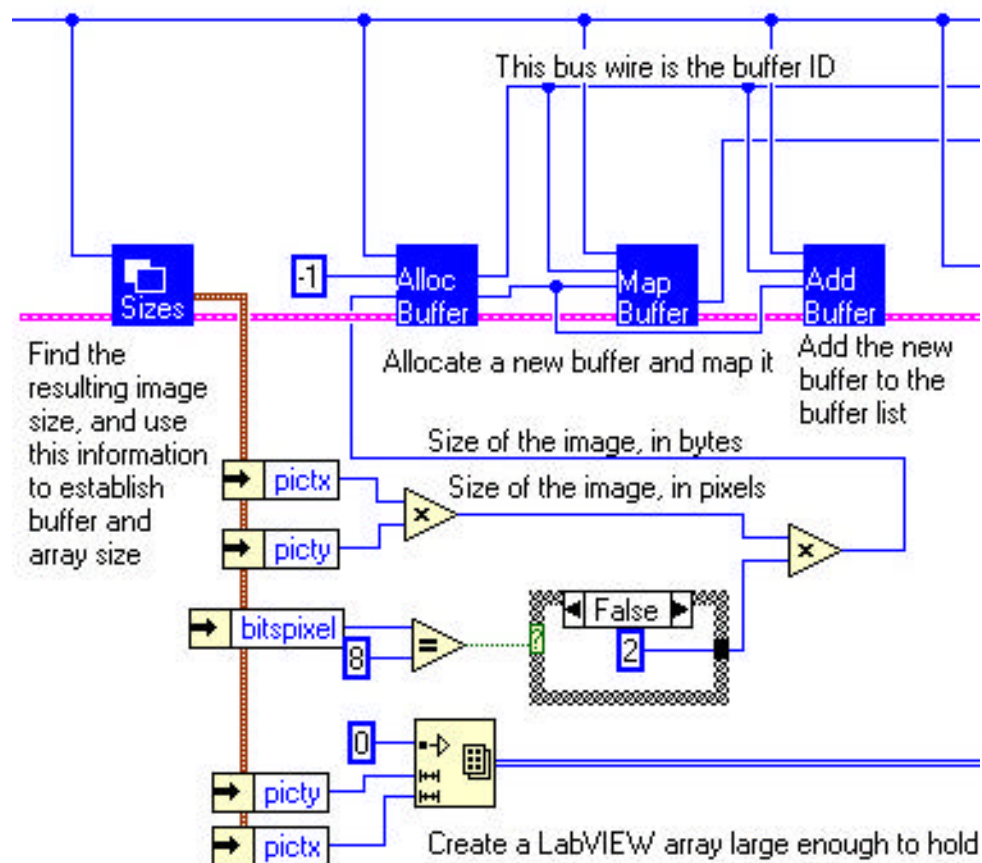


**Get Param** - “**get parameters.vi**” – Retrieves information on the type of camera connected. This will tell you if your camera is color or monochrome, what its maximum resolution is and other useful information.



**Set Mode** - “**set mode.vi**” – Sets the various parameters needed to make an exposure, i.e. exposure time, binning, trigger mode, etc.

### 3.2 Set up a buffer to store the image







- “**get sizes.vi**” - Uses this VI to find out the resolution of the image after the exposure has been set up. The resolution is a function of the binning settings, so it is best to use this function after calling “set mode.vi”. This information is used to determine the size of the buffer needed to store the image. If the PixelFly is set to acquire 12 bit images, two bytes must be made available for each pixel. Note that a separate LabVIEW array is initialized to hold the output data. The data buffers are not directly accessible to LabVIEW, but VI’s are provided to format the data from the buffer as a LabVIEW array (see below).



- “**allocate buffer.vi**” - A portion of memory is assigned to contain the image data. A value of -1 is wired to the buffer number input to indicate that a new buffer is to be created. The size of the buffer is calculated as the product of the vertical and horizontal resolution, times the number of bytes per pixel.

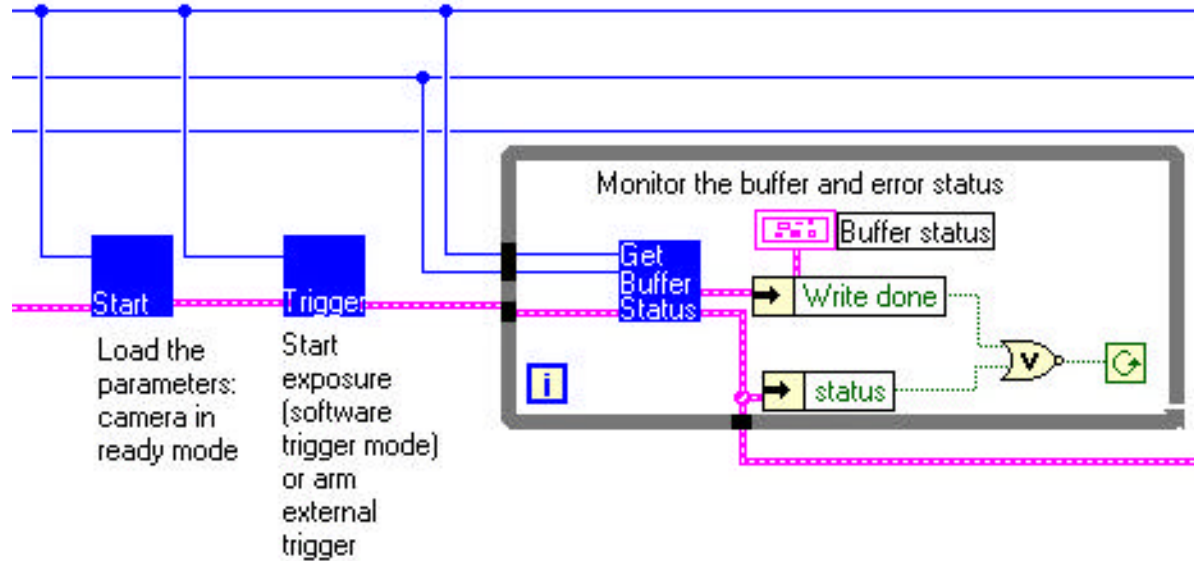


- “**map buffer.vi**” – The buffer must be mapped before LabVIEW can access it.



- “**add buffer to list.vi**” – Creates a queue of buffers. Each subsequent exposure will go to a new buffer in the queue. Since we are only taking one exposure, we have a queue of only one buffer.

### 3.3. Start and exposure and wait for the data to be ready

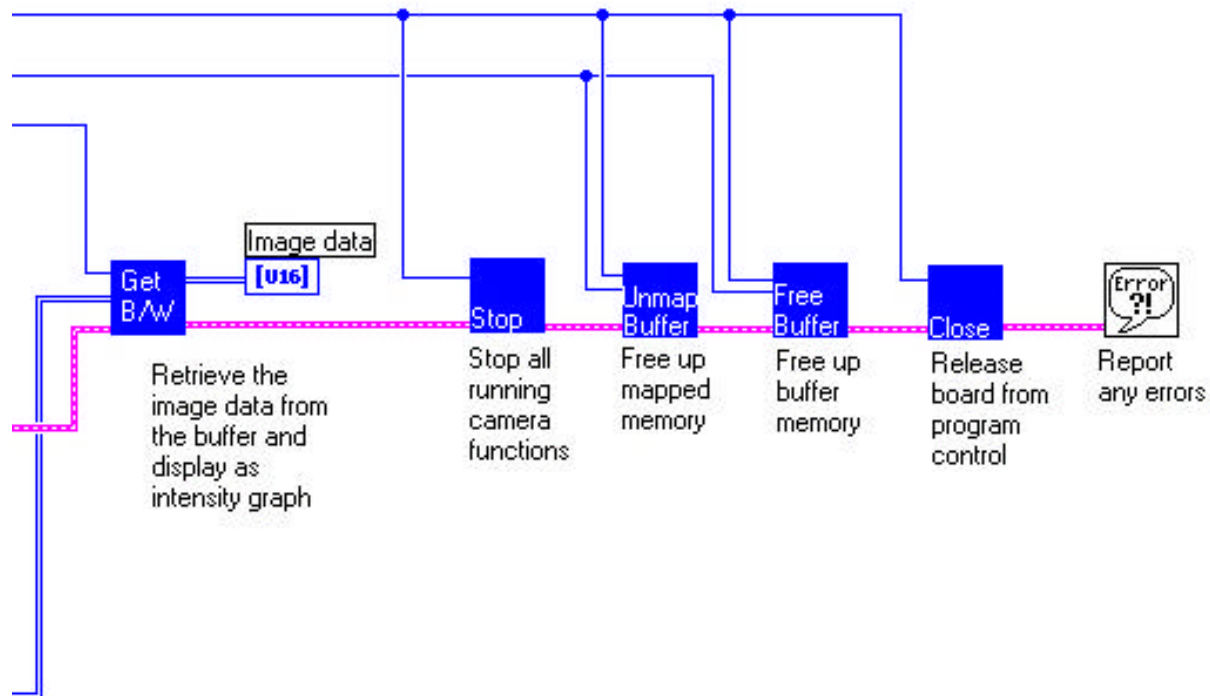


**Start** - "**start camera.vi**" – Put the camera into ready mode, waiting for the exposure to start.

**Trigger** - "**trigger camera.vi**" – If the trigger mode is "internal", an exposure starts when this VI is called. In "external" trigger mode, the exposure starts at the next leading edge of the trigger input after the VI is called.

**Get Buffer Status** - "**get buffer status.vi**" – Find out if the exposure has finished and if the data has been written to the buffer. Note that there is extra logic in the loop to query the error status, and terminate the loop if an error occurs.

### 3.4. Get image data, free up buffers and terminate the program



**Get B/W** - “**get BW from buffer.vi**” – Transfers the data from the buffer to an array of 16 bit integers. In this example, the data is displayed in a LabVIEW intensity graph. “get BW from buffer” returns the raw data from the sensor, and works with both color or monochrome cameras. Other VI’s are available to display data as color images.

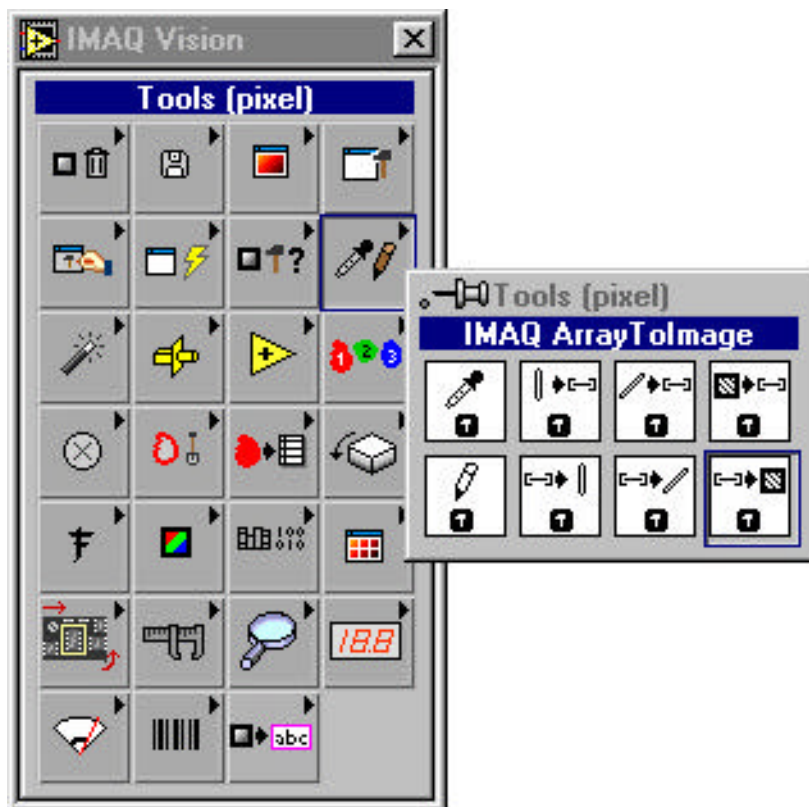
**Stop** - “**stop camera.vi**” – Stops all running camera operations

**Unmap Buffer** - “**unmap buffer.vi**” – Releases the mapped buffer

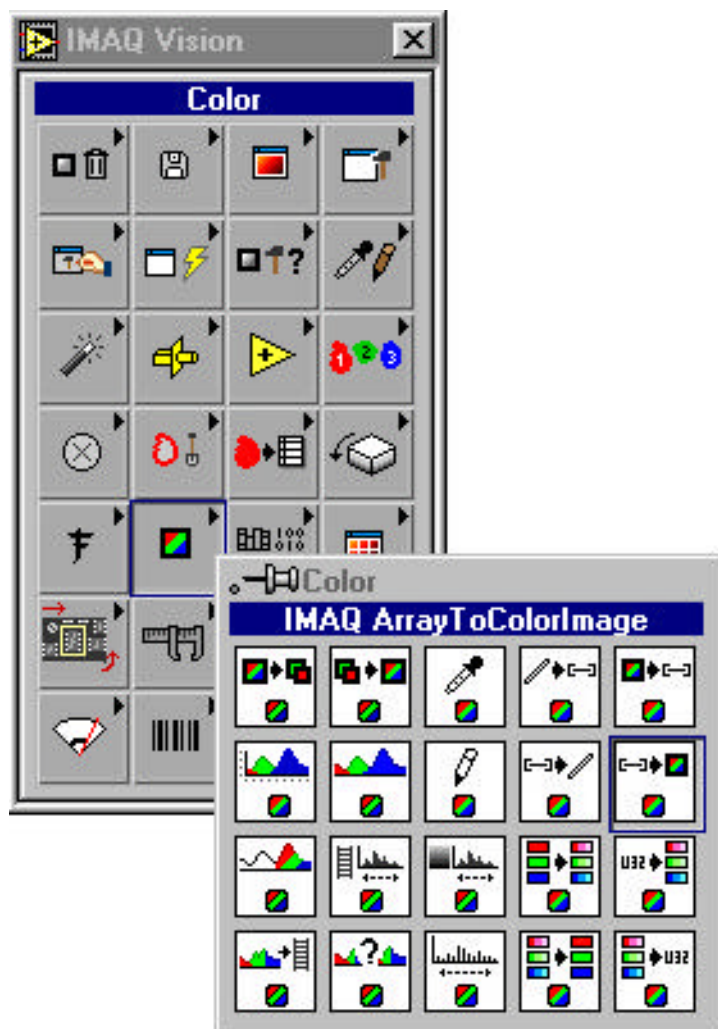
**Free Buffer** - “**free buffer.vi**” – De-allocates the memory that was used by the buffer.

**Close** - “**closeboard.vi**” – Releases the camera from program control and frees up any resources used by that camera.

Users of the IMAQ vision software can convert the array of image data into an IMAQ image cluster using the “IMAQ ArrayToImage.vi” in the “Tools (pixel)” IMAQ sub-palette:



Color IMAQ images can also be created using the “IMAQ ArrayToColorImage.vi”:

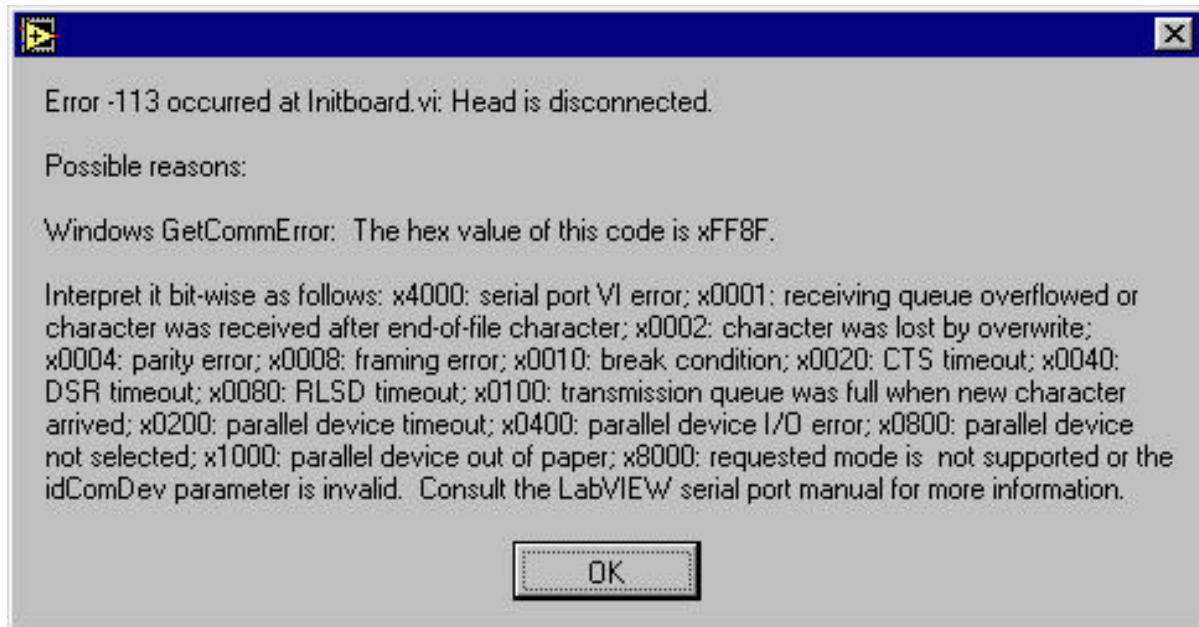


See the example programs “Take a single exposure with IMAQ.vi” and “Continuous acquire and display with IMAQ.vi” in the driver library for examples of how to use these functions.

## 4 Error clusters and messages

The LabVIEW drivers for the PixelFly use standard LabVIEW error clusters to report errors in the camera operation. If a VI generates an error, subsequent VI's will detect it and pass the error along. In this way the source of the error can be traced back.

The error cluster consists of a Boolean that is set true if an error condition exists, an integer giving the error code, and a text description of the error, identifying the VI where the error occurred. The PixelFly drivers return error codes generated by the driver software, which are negative numbers ranging from -1 to -233. These error codes are also used by other LabVIEW functions, so two interpretations of the error will be displayed by the LabVIEW error handler, both the PixelFly description and the LabVIEW version. For example, you may see a message like this from the LabVIEW error handler VI:



However, the context of the message (i.e. which VI caused the error) should allow you to determine which is the correct interpretation.

## 5. Sample programs

Several sample programs are provided to illustrate the use of the various sub-VI's. These sample programs are accessible through the PixelFly function palette. Drop the Icon on a new wiring diagram, and then double click to open the front panel. Use Ctrl-H for descriptions of the various inputs and outputs.



- "Simpler test.vi" is the most basic program. It tests communications with the PixelFly and obtains information about the model of camera connected.



- "Take a single exposure without IMAQ.vi" uses the LabVIEW color chart to display the image, so users with the base version of LabVIEW should be able to run this VI.



- LabVIEW can also display images as pictures. LabVIEW picture controls and indicators display bitmapped data. For users that want to experiment with picture controls, we have provided the "Take single exposure – display as picture.vi" and "Continuous acquire – display as picture.vi" examples.



For users who have purchased the IMAQ Vision package from National Instruments, other VI's are provided to display images in an IMAQ window. - "Take a single exposure with IMAQ.vi" and "Continuous acquire and display with IMAQ.vi".

Studying the wiring diagrams for these VI's will provide some insight into how LabVIEW is used to control the PixelFly and obtain image data from the camera.



## 6. Individual VI descriptions

The VI's in the PixelFly library are listed here, in the order that they might be used in an application, i.e. initialization VI's first, followed by status and set-up.

### 6.1 Driver setup functions

#### 6.1.1 initboard.vi

Initializes the PixelFly interface board and camera head. This function must be called once for each camera in the system. Up to four cameras can be supported. Each call to the function must supply a unique value for "boardid in".



**boardid in** Each call to the function must supply a unique value for "boardid in". Subsequent calls to other functions will use this number to select which board the driver will communicate with. Up to four cameras can be supported.

**error in (no error)** error in describes an error that you want to check. If you leave error in unwired, this VI checks error code for errors.

**status** The status boolean is either TRUE (cross) for an error, or FALSE (checkmark) for no error or a warning.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

**code** The code input identifies the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

**source** The source string describes the origin of the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

**Error** Error number assigned by the PixelFly driver. Error codes are explained in the "error out" cluster.

**boardid** Unique number assigned to a particular camera and board. Echoes "boardid in".

**error out** error out is a cluster containing the same information as in status out, code out, and source out. It has the same structure as error in.

**status** The status boolean is either TRUE (cross) for an error, or FALSE (checkmark) for no error or a warning.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

**code** The code input identifies the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.



**source** The source string describes the origin of the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

### 6.1.2 closeboard.vi

Releases the resources used by the PixelFly driver for this board. This must be called once for each open board (each with a unique handle number) before exiting the LabVIEW application.



**error in** (no error) error in describes an error that you want to check. If you leave error in unwired, this VI checks error code for errors.



**status** The status boolean is either TRUE (cross) for an error, or FALSE (checkmark) for no error or a warning.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.



**code** The code input identifies the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.



**source** The source string describes the origin of the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.



**Handle in** An integer assigned by the PixelFly driver to identify which camera is referenced by this VI call. Up to four cameras can be supported.



**Error out** Error number assigned by the PixelFly driver. Error codes are explained in the "error out" cluster.



**error out** error out is a cluster containing the same information as in status out, code out, and source out. It has the same structure as error in.



**status** The status boolean is either TRUE (cross) for an error, or FALSE (checkmark) for no error or a warning.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.



**code** The code input identifies the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information

about the error displayed.



**source** The source string describes the origin of the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

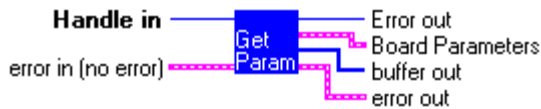


**Handle out** An integer assigned by the PixelFly driver to identify which camera is referenced by this VI call. Up to four cameras can be supported. Calling this VI will not change the value of "Handle out" from the value specified by "handle in".

## 6.2 Status functions:

### 6.2.1 get parameters.vi

Returns camera type, resolution, current gain, binning settings, etc. for the specified camera.



**error in** (no error) error in describes an error that you want to check. If you leave error in unwired, this VI checks error code for errors.



**status** The status boolean is either TRUE (cross) for an error, or FALSE (checkmark) for no error or a warning.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.



**code** The code input identifies the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.



**source** The source string describes the origin of the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.



**Handle in** An integer assigned by the PixelFly driver to identify which camera is referenced by this VI call. Up to four cameras can be supported.



**Error out** Error number assigned by the PixelFly driver. Error codes are explained in the "error out" cluster.



**error out** error out is a cluster containing the same information as in status out, code out, and source out. It has the same structure as error in.



**status** The status boolean is either TRUE (cross) for an error, or FALSE (checkmark) for no error or a warning.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

**I32** **code** The code input identifies the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

**abc** **source** The source string describes the origin of the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

**U32** **buffer out** Array of 32 bit integers containing status information. This data is used to form the "Board parameters" cluster. The "Board parameters" cluster formats this data into a much more useful state, and is the preferred means of capturing status information. However, you can use the information in the "buffer out" array to perform custom calculations on status elements if desired.

**U32**

**FTI** **Board Parameters** Detailed information about the status and configuration of the interface board and camera head.

**U32** **Type** Code for interface board revision history. Report this number to Cooke service if you have a problem running any camera function.

**U32** **Number** Number assigned to this interface board by the driver. Up to four boards can be supported.

**TF** **Initialized** Indicates that the board and camera head have been successfully initialized by the driver

**TF** **Running** Indicates that an exposure is running.

**TF** **Lost head** Indicates that the board has lost contact with the camera head

**U32** **CCD X size** Maximum horizontal resolution of CCD.

**U32** **CCD Y size** Maximum vertical resolution of CCD.

**U32** **Mode** Last assigned operating mode.

**U32** **Exposure time** Most recently assigned exposure time

**U32** **Exposure level** Most recently assigned exposure level. For light metering cameras only.

**U32** **Horizontal bin** Current horizontal binning setting.

**U32** **Vertical bin** Current vertical binning setting.

**TF** **Reg W** Register write successful.

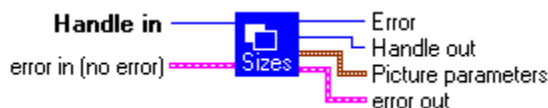
**U32** **Gain** Current gain setting

**I32** **Pixel depth** Number of bits per pixel. This number will be either 8 or 12.

- U32 **Bit Shift** Most recently assigned shift value for 12 to 8 bit conversion
- U32 **Offset** Offset into buffer for last data transfer
- U32 **Last exposure** Exposure time of last complete exposure.
- TF **Double** Indicates that camera is double shutter capable
- TF **Light meter** Indicates that the camera has an internal prism for light metering. Found in auto-exposure cameras.
- TF **Color** TRUE if camera head is color, FALSE if monochrome head is connected.
- U32 **Open count** Counter
- TF **Active buffer** Indicates that this board is queued to the active buffer
- TF **Next Buffer** Indicates that this board is assigned to the next buffer in the queue

### 6.2.2 Get sizes.vi

Returns maximum and current resolution, along with the pixel depth for the current camera configuration.



- I32 **Handle in** An integer assigned by the PixelFly driver to identify which camera is referenced by this VI call. Up to four cameras can be supported.
- F1 **error in (no error)** error in describes an error that you want to check. If you leave error in unwired, this VI checks error code for errors.
- TF **status** The status boolean is either TRUE (cross) for an error, or FALSE (checkmark) for no error or a warning.  
  
The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.
- I32 **code** The code input identifies the error or warning reported.  
  
The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.
- abc **source** The source string describes the origin of the error or warning reported.  
  
The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.
- I32 **Error** Error number assigned by the PixelFly driver. Error codes are explained in the "error out" cluster.
- I32 **Handle out** An integer assigned by the PixelFly driver to identify which camera is referenced by this VI call. Up to four cameras can be supported. Calling this VI will not

The COOKE Corporation

1091 Centre Road, Suite 100, Auburn Hills, MI 48326

Tel: (248) 276-8820 Fax: (248) 276-8825 Email: [info@cookecorp.com](mailto:info@cookecorp.com) Website: [www.cookecorp.com](http://www.cookecorp.com)

change the value of "Handle out" from the value specified by "handle in".



**error out** error out is a cluster containing the same information as in status out, code out, and source out. It has the same structure as error in.



**status** The status boolean is either TRUE (cross) for an error, or FALSE (checkmark) for no error or a warning.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.



**code** The code input identifies the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.



**source** The source string describes the origin of the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.



**Picture parameters** Contains information on the selected camera's resolution and pixel depth.



**ccdsize** Maximum horizontal resolution of CCD.



**ccdy** Maximum vertical resolution of CCD.



**pictx** Horizontal resolution of image in the current configuration. Will differ from "ccdsize" if binning is higher than 1.



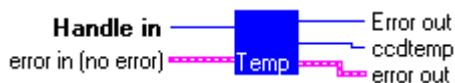
**picty** Vertical resolution of image in the current configuration. Will differ from "ccdy" if binning is higher than 1.



**bitapixel** Dynamic range as number of bits per pixel. This number will be either 8 or 12.

### 6.2.3 gettemp.vi

Obtains the current temperature inside the camera head.



**error in (no error)** error in describes an error that you want to check. If you leave error in unwired, this VI checks error code for errors.



**status** The status boolean is either TRUE (cross) for an error, or FALSE (checkmark) for no error or a warning.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.



**code** The code input identifies the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.



**source** The source string describes the origin of the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.



**Handle in** An integer assigned by the PixelFly driver to identify which camera is referenced by this VI call. Up to four cameras can be supported.



**Error out** Error number assigned by the PixelFly driver. Error codes are explained in the "error out" cluster.



**error out** error out is a cluster containing the same information as in status out, code out, and source out. It has the same structure as error in.



**status** The status boolean is either TRUE (cross) for an error, or FALSE (checkmark) for no error or a warning.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.



**code** The code input identifies the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.



**source** The source string describes the origin of the error or warning reported.

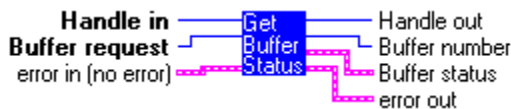
The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.



**ccdtemp**

#### 6.2.4 Get buffer status.vi

Returns the status of the selected buffer, including information on queue position, DMA status and any errors encountered.



**error in (no error)** error in describes an error that you want to check. If you leave error in unwired, this VI checks error code for errors.



**status** The status boolean is either TRUE (cross) for an error, or FALSE (checkmark) for no error or a warning.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.





**code** The code input identifies the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.



**source** The source string describes the origin of the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.



**Handle in** An integer assigned by the PixelFly driver to identify which camera is referenced by this VI call. Up to four cameras can be supported.



**Buffer request** This buffer number must be one of the numbers assigned by the "allocate buffer" VI. This will select which of the assigned buffers to status check.



error out error out is a cluster containing the same information as in status out, code out, and source out. It has the same structure as error in.



**status** The status boolean is either TRUE (cross) for an error, or FALSE (checkmark) for no error or a warning.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.



**code** The code input identifies the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.



**source** The source string describes the origin of the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.



**Handle out** An integer assigned by the PixelFly driver to identify which camera is referenced by this VI call. Up to four cameras can be supported. Calling this VI will not change the value of "Handle out" from the value specified by "handle in".



**Buffer number** Outputs the number of the buffer for this status check.



**Buffer status** Cluster containing status information, addressable by name. See the individual indicators in the cluster for more details.



**Write** Flag that indicates a write to buffer is in progress when true.



**Write done** Flag that indicates a write to buffer is completed when true.



**Queued** Flag that indicates that the selected buffer has been queued.



**Cancelled** Indicates that a transfer to this buffer has been cancelled at user request.



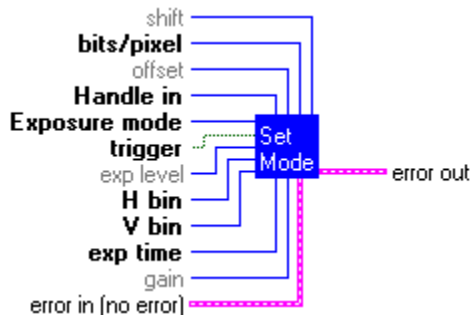
**Select** Indicates that this buffer will be removed from the queue and will be selected as the recipient of the next data transfer.

- [TF] Select done** Indicates that this buffer has been removed from the queue and will be the recipient of the next data transfer.
- [TF] Burst** Indicates an error has occurred in a DMA transfer burst operation.
- [TF] Size** Indicates the number of bytes to transfer is too large for the selected buffer.
- [TF] Event** Indicates an error occurred while attempting to generate a buffer event.
- [TF] Timeout** Indicates time out error in transfer of data to buffer. Can be caused by mismatch between buffer size and requested number of bytes.
- [I32] Open** Indicates number of open buffers
- [I32] Map** Physical address where data is mapped.
- [I32] Exposure** Exposure time of image in buffer.
- [I32]**
- [I32] Offset** Offset into buffer as specified by "add buffer to list" VI.
- [I32]**
- [I32] Size** Size of buffer in bytes
- [I32]**
- [I32] Transfer** Number of bytes transferred
- [I32]**
- [I32] Total Size** Size of buffer in bytes

## 6.3 Control functions:

### 6.3.1 Set mode.vi

Configures the camera for operating mode, trigger mode, exposure time, binning and pixel depth.



**error in (no error)** error in describes an error that you want to check. If you leave error in unwired, this VI checks error code for errors.

**status** The status boolean is either TRUE (cross) for an error, or FALSE (checkmark) for no error or a warning.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

**code** The code input identifies the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

**source** The source string describes the origin of the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

**Handle in** An integer assigned by the PixelFly driver to identify which camera is referenced by this VI call. Up to four cameras can be supported.

**Exposure mode** Camera operating modes:

Asynch shutter - fast shuttering from 10 usec to 10 ms  
 Double shutter - two images in rapid sequence  
 Video mode - long exposures from 10 ms to 10 s  
 Auto exposure - Adjusts exposure time to light level

Note: Double shutter and Auto exposure modes require special camera heads. The "get parameters" VI will interrogate the camera head to determine which modes are available.

**trigger** Toggles between internal (software) or external (hardware trigger).

**exp level** In light metering cameras, this sets the target exposure level as a percent of full scale. Exposure time will be adjusted to accommodate shifting light levels.

**I32** **exp time** In Asynch and Double shutter modes, this is the number of microseconds that the camera will expose when triggered.

In Video mode, this is the number of milliseconds that the camera will expose when triggered.

**I32** **H bin** Horizontal binning control. Two pixels can be binned in this direction for increased sensitivity.

**I32** **V bin** Vertical binning control. Two pixels can be binned in this direction for increased sensitivity.

**I32** **gain** Analog gain stage select. Gain can be increased by a factor of 2.

**I32** **offset** Not implemented in this release. Do not change from the default value.

**I32** **bits/pixel** Sets pixel dynamic range as the number of bits. Possible values are 12 bits (4096 grey levels) or 8 bits (255 grey levels).

**I32** **shift** When bits/pixel is set to 8 bits, the driver implements a right shift in the data before truncating 12 bits to the most significant 8. This can be used to maximize the dynamic range of the resulting 8 bit image.

**Err** **error** out error out is a cluster containing the same information as in status out, code out, and source out. It has the same structure as error in.

**TF** **status** The status boolean is either TRUE (cross) for an error, or FALSE (checkmark) for no error or a warning.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

**I32** **code** The code input identifies the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

**abc** **source** The source string describes the origin of the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

### 6.3.2 Start camera.vi

Starts a camera acquisition sequence, placing the camera in wait state until hardware or software trigger is received.



**Err** **error in (no error)** error in describes an error that you want to check. If you leave error in unwired, this VI checks error code for errors.

**TF** **status** The status boolean is either TRUE (cross) for an error, or FALSE (checkmark) for no error or a warning.


The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.


 **code** The code input identifies the error or warning reported.


The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.


 **source** The source string describes the origin of the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

 **Handle in** An integer assigned by the PixelFly driver to identify which camera is referenced by this VI call. Up to four cameras can be supported.

 **Error out** Error number assigned by the PixelFly driver. Error codes are explained in the "error out" cluster.

 **error out** error out is a cluster containing the same information as in status out, code out, and source out. It has the same structure as error in.

 **status** The status boolean is either TRUE (cross) for an error, or FALSE (checkmark) for no error or a warning.


The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

 **code** The code input identifies the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

 **source** The source string describes the origin of the error or warning reported.


The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

 **Handle out** An integer assigned by the PixelFly driver to identify which camera is referenced by this VI call. Up to four cameras can be supported. Calling this VI will not change the value of "Handle out" from the value specified by "handle in".

### 6.3.3 Stop camera.vi

Aborts a camera acquisition sequence. Call this if camera state is unclear. Once camera is stopped, mode binning, exposure time, etc. can be modified



 **error in (no error)** error in describes an error that you want to check. If you leave error in unwired, this VI checks error code for errors.


 **status** The status boolean is either TRUE (cross) for an error, or FALSE

(checkmark) for no error or a warning.


The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.


 **code** The code input identifies the error or warning reported.


The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.


 **source** The source string describes the origin of the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

 **Handle in** An integer assigned by the PixelFly driver to identify which camera is referenced by this VI call. Up to four cameras can be supported.

 **Error out** Error number assigned by the PixelFly driver. Error codes are explained in the "error out" cluster.

 **error out** error out is a cluster containing the same information as in status out, code out, and source out. It has the same structure as error in.

 **status** The status boolean is either TRUE (cross) for an error, or FALSE (checkmark) for no error or a warning.


The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

 **code** The code input identifies the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

 **source** The source string describes the origin of the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

 **Handle out** An integer assigned by the PixelFly driver to identify which camera is referenced by this VI call. Up to four cameras can be supported. Calling this VI will not change the value of "Handle out" from the value specified by "handle in".

### 6.3.4 Trigger.vi

Triggers an exposure. If the PixelFly is in internal trigger mode, the exposure starts as soon as this VI is called. In external trigger mode, the PixelFly waits for a transition on the external trigger input before exposing.



**error in (no error)** error in describes an error that you want to check. If you leave error in unwired, this VI checks error code for errors.

**status** The status boolean is either TRUE (cross) for an error, or FALSE (checkmark) for no error or a warning.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

**code** The code input identifies the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

**source** The source string describes the origin of the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

**Handle in** An integer assigned by the PixelFly driver to identify which camera is referenced by this VI call. Up to four cameras can be supported.

**Error out** Error number assigned by the PixelFly driver. Error codes are explained in the "error out" cluster.

**error out** error out is a cluster containing the same information as in status out, code out, and source out. It has the same structure as error in.

**status** The status boolean is either TRUE (cross) for an error, or FALSE (checkmark) for no error or a warning.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

**code** The code input identifies the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

**source** The source string describes the origin of the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

**Handle out** An integer assigned by the PixelFly driver to identify which camera is

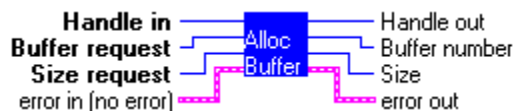


referenced by this VI call. Up to four cameras can be supported. Calling this VI will not change the value of "Handle out" from the value specified by "handle in".

## 6.4 Buffer functions:

### 6.4.1 Allocate buffer.vi

Allocates an image buffer in computer memory. The size of the buffer is specified as the number of bytes required. To allocate a new buffer, use -1 as the buffer number. This function can also re-size an existing buffer



**error in (no error)** error in describes an error that you want to check. If you leave error in unwired, this VI checks error code for errors.

**status** The status boolean is either TRUE (cross) for an error, or FALSE (checkmark) for no error or a warning.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

**code** The code input identifies the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

**source** The source string describes the origin of the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

**Handle in** An integer assigned by the PixelFly driver to identify which camera is referenced by this VI call. Up to four cameras can be supported

**Buffer request** Specifies which buffer to allocate or re-size. Default is -1, which specifies that a new buffer is to be allocated. If the number of an existing buffer is used, that buffer is re-sized.

**Size request** Request the number of bytes to allocate for the buffer. Large buffers can be re-sized to this input number of bytes

**error out** error out is a cluster containing the same information as in status out, code out, and source out. It has the same structure as error in.

**status** The status boolean is either TRUE (cross) for an error, or FALSE (checkmark) for no error or a warning.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

**code** The code input identifies the error or warning reported.

The COOKE Corporation

1091 Centre Road, Suite 100, Auburn Hills, MI 48326

Tel: (248) 276-8820 Fax: (248) 276-8825 Email: [info@cookecorp.com](mailto:info@cookecorp.com) Website: [www.cookecorp.com](http://www.cookecorp.com)

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

**source** The source string describes the origin of the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

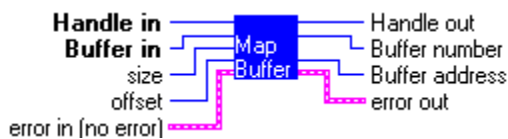
**Handle out** An integer assigned by the PixelFly driver to identify which camera is referenced by this VI call. Up to four cameras can be supported. Calling this VI will not change the value of "Handle out" from the value specified by "handle in".

**Buffer number** Assigned number of buffer. For new buffers, this will be a unique integer. For re-allocated buffers, this will be the same as "buffer request".

**Size** Actual size of buffer allocated. This may be slightly larger than the requested size, in order to maintain buffer limits on page boundaries.

#### 6.4.2 Map buffer.vi

Maps a specified buffer into main memory space, starting at the "buffer address" output. "Buffer address" is then used by the image transfer routines.



**error in (no error)** error in describes an error that you want to check. If you leave error in unwired, this VI checks error code for errors.

**status** The status boolean is either TRUE (cross) for an error, or FALSE (checkmark) for no error or a warning.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

**code** The code input identifies the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

**source** The source string describes the origin of the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

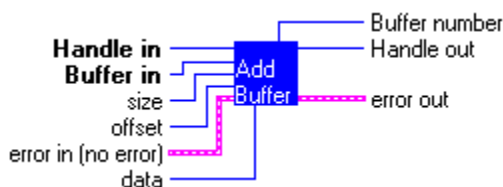
**Handle in** An integer assigned by the PixelFly driver to identify which camera is referenced by this VI call. Up to four cameras can be supported.

**Buffer in** Number of the buffer to be mapped into main memory. This must be a valid number from a previous call to "Allocate buffer.vi"

- I32 size** Size, in bytes, of the buffer to be mapped. This must match the size of the buffer referenced by the "Buffer in" value.
- I32 offset** Not yet implemented. Do not change this value from the default.
- Err error out** error out is a cluster containing the same information as in status out, code out, and source out. It has the same structure as error in.
- TF status** The status boolean is either TRUE (cross) for an error, or FALSE (checkmark) for no error or a warning.
- The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.
- I32 code** The code input identifies the error or warning reported.
- The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.
- abc source** The source string describes the origin of the error or warning reported.
- The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.
- I32 Handle out** An integer assigned by the PixelFly driver to identify which camera is referenced by this VI call. Up to four cameras can be supported. Calling this VI will not change the value of "Handle out" from the value specified by "handle in".
- I32 Buffer number** Echos the "Buffer in" value.
- I32 Buffer address** Linear (absolute) address of the buffer in main system memory.

#### 6.4.3 Add buffer to list.vi


Places a buffer in the acquisition queue. If the specified buffer is already in the queue, an error is generated. The queue can hold up to 32 buffers. Each subsequent call to this VI adds a buffer to the end of the queue. Once the data has been transferred from the camera to the buffer it is removed from the queue, and the next data transfer will use the next buffer in the queue.




- Err error in (no error)** error in describes an error that you want to check. If you leave error in unwired, this VI checks error code for errors.
- TF status** The status boolean is either TRUE (cross) for an error, or FALSE (checkmark) for no error or a warning.
- The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.


 **code** The code input identifies the error or warning reported.


The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.


 **source** The source string describes the origin of the error or warning reported.


The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.


 **Handle in** An integer assigned by the PixelFly driver to identify which camera is referenced by this VI call. Up to four cameras can be supported.


 **Buffer in** This buffer number must be one of the numbers assigned by the "allocate buffer" VI. This will select which of the assigned buffers to de-allocate.

 **size** Number of bytes to send from the camera to the buffer. 12 bit data is sent as 2 bytes. The number of bytes must be equal to or less than the size of the whole image in bytes. If the requested number is larger than the camera size, a timeout error will occur. Size must be larger than 4096 bytes.

 **offset** Specifies an offset into the buffer where the data transfer is to begin. If the image size is smaller than the buffer size, "offset" can be used to store multiple small images in the buffer.

 **data** This input is reserved for the use of future driver functionality. Do not change the value from the default setting.

 error out error out is a cluster containing the same information as in status out, code out, and source out. It has the same structure as error in.

 **status** The status boolean is either TRUE (cross) for an error, or FALSE (checkmark) for no error or a warning.


The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

 **code** The code input identifies the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

 **source** The source string describes the origin of the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

 **Handle out** An integer assigned by the PixelFly driver to identify which camera is referenced by this VI call. Up to four cameras can be supported. Calling this VI will not change the value of "Handle out" from the value specified by "handle in".

 **Buffer number** Outputs the number of the buffer that was added to the list.

#### 6.4.4 Remove buffer from list.vi

Removes a buffer from the acquisition queue. If the specified buffer is not in the queue, an error is generated. Once the data has been transferred from the camera to the buffer it is removed from the queue, so typically this function needs to be called when an acquisition is aborted, and before exiting the application.



**error in (no error)** error in describes an error that you want to check. If you leave error in unwired, this VI checks error code for errors.

**status** The status boolean is either TRUE (cross) for an error, or FALSE (checkmark) for no error or a warning.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

**code** The code input identifies the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

**source** The source string describes the origin of the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

**Handle in** An integer assigned by the PixelFly driver to identify which camera is referenced by this VI call. Up to four cameras can be supported.

**Buffer in** Number of the buffer to be removed from the queue.

**error out** error out is a cluster containing the same information as in status out, code out, and source out. It has the same structure as error in.

**status** The status boolean is either TRUE (cross) for an error, or FALSE (checkmark) for no error or a warning.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

**code** The code input identifies the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

**source** The source string describes the origin of the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

**Handle out** An integer assigned by the PixelFly driver to identify which camera is referenced by this VI call. Up to four cameras can be supported. Calling this VI will not change the value of "Handle out" from the value specified by "handle in".

**132** **Buffer number** Number of the buffer removed by this operation

#### 6.5.4 Unmap buffer.vi

Unmaps a specified buffer in main memory space. All mapped buffers must be unmapped before exiting the application.



**571** **error in (no error)** error in describes an error that you want to check. If you leave error in unwired, this VI checks error code for errors.

**TF** **status** The status boolean is either TRUE (cross) for an error, or FALSE (checkmark) for no error or a warning.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

**132** **code** The code input identifies the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

**abc** **source** The source string describes the origin of the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

**132** **Handle in** An integer assigned by the PixelFly driver to identify which camera is referenced by this VI call. Up to four cameras can be supported.

**132** **Buffer in** Number of the buffer to be unmapped from main memory. This must be a valid number from a previous call to "Allocate buffer.vi", and the specified buffer must have been mapped using "map buffer" VI.

**571** **error out** error out is a cluster containing the same information as in status out, code out, and source out. It has the same structure as error in.

**TF** **status** The status boolean is either TRUE (cross) for an error, or FALSE (checkmark) for no error or a warning.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

**132** **code** The code input identifies the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

**abc** **source** The source string describes the origin of the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information

The COOKE Corporation

1091 Centre Road, Suite 100, Auburn Hills, MI 48326

Tel: (248) 276-8820 Fax: (248) 276-8825 Email: info@cookecorp.com Website: www.cookecorp.com

about the error displayed.

**Handle out** An integer assigned by the PixelFly driver to identify which camera is referenced by this VI call. Up to four cameras can be supported. Calling this VI will not change the value of "Handle out" from the value specified by "handle in".

**Buffer number** Echoes the "Buffer in" value.

#### 6.4.6 Free buffer.vi

Frees (de-allocates) a buffer created by the "allocate buffer" VI. If the specified buffer was put into a queue using the "Add buffer to list" VI, you must call "Remove buffer from list" before "free buffer".



**error in (no error)** error in describes an error that you want to check. If you leave error in unwired, this VI checks error code for errors.

**status** The status boolean is either TRUE (cross) for an error, or FALSE (checkmark) for no error or a warning.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

**code** The code input identifies the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

**source** The source string describes the origin of the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

**Handle in** An integer assigned by the PixelFly driver to identify which camera is referenced by this VI call. Up to four cameras can be supported.

**Buffer in** This buffer number must be one of the numbers assigned by the "allocate buffer" VI. This will select which of the assigned buffers to de-allocate.

**error out** error out is a cluster containing the same information as in status out, code out, and source out. It has the same structure as error in.

**status** The status boolean is either TRUE (cross) for an error, or FALSE (checkmark) for no error or a warning.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.


**code** The code input identifies the error or warning reported.


The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.



 **source** The source string describes the origin of the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

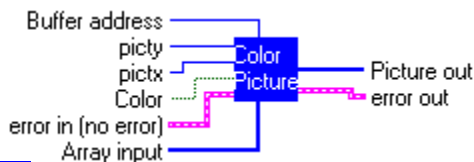
 **Handle out** An integer assigned by the PixelFly driver to identify which camera is referenced by this VI call. Up to four cameras can be supported. Calling this VI will not change the value of "Handle out" from the value specified by "handle in".


 **Buffer number** Outputs the number of the buffer, which was de-allocated.

## 6.5 Image formatting functions:


### 6.5.1 ColorPicture.vi


Retrieves color data from buffer in a format compatible with LabVIEW picture indicators.




 **picty** Vertical size (number of pixels) of the image in the buffer.


 **pictx** Horizontal size (number of pixels) of the image in the buffer.

 **Buffer address** Address where camera data is stored. This address is provided by "Map buffer.vi"

 **Array input** Array to hold color byte data. Array size must be equal to the product of the X and Y extent of the image, times 3 (for one byte each of R G and B). For example, a 640 X 480 color image would require 921600 elements.

 **Element Data**

 **error in (no error)** error in describes an error that you want to check. If you leave error in unwired, this VI checks error code for errors.

 **status** The status boolean is either TRUE (cross) for an error, or FALSE (checkmark) for no error or a warning.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

 **code** The code input identifies the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

 **source** The source string describes the origin of the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.



**Color** Sets mode for buffer conversion. If input image is color, the output is formatted as RGB. If input is monochrome, the intensity values are replicated in each of the R, G and B channels, so that the output appears in grey scale.



**Picture out** Array containing color byte data. Array size will be equal to the product of the X and Y extent of the image, times 3 (for one byte each of R G and B). For example, a 640 X 480 color image would require 921600 elements. This RGB data can be flattened and displayed using the LabVIEW picture indicator.



**Element Data**



**error out** error out is a cluster containing the same information as in status out, code out, and source out. It has the same structure as error in.



**status** The status boolean is either TRUE (cross) for an error, or FALSE (checkmark) for no error or a warning.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.



**code** The code input identifies the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.



**source** The source string describes the origin of the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

### 6.5.2 Get BW from buffer.vi

Obtains an image from a buffer and returns it as an array of 12 bit integers. An array of U16 integers must first be defined that matches the horizontal and vertical dimensions of the image in the buffer. This data format can be used for BW images, particularly if displayed in an Intensity chart. For IMAQ applications, use the "Array to IMAQ image" vi.



**Buffer address** Linear address, from the "Map buffer" VI, of the buffer from which to retrieve the data. Buffer must have been previously defined, mapped and queued.



**Array input** Array for image data storage. Must be previously defined to match the horizontal and vertical dimensions of the image in the buffer.



**Element Data**



**error in (no error)** error in describes an error that you want to check. If you leave error in unwired, this VI checks error code for errors.



**status** The status boolean is either TRUE (cross) for an error, or FALSE (checkmark) for no error or a warning.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.



**code** The code input identifies the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.



**source** The source string describes the origin of the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.



**Image data** Color image data retrieved from specified buffer. Use "Array to IMAQ image" VI to display.



**Element Data**



**error out** error out is a cluster containing the same information as in status out, code out, and source out. It has the same structure as error in.



**status** The status boolean is either TRUE (cross) for an error, or FALSE (checkmark) for no error or a warning.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.



**code** The code input identifies the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

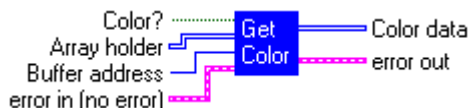


**source** The source string describes the origin of the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

### 6.5.3 get color from buffer.vi

Obtains an image from a buffer and returns it as an array of 32 bit integers, with 8 bits each of RGB and 0 padding. Compatible with IMAQ Array to Image VI. An array of U32 integers must first be defined that matches the horizontal and vertical dimensions of the image in the buffer.



**[U32] Array holder** Array for image data storage. Must be previously defined to match the horizontal and vertical dimensions of the image in the buffer.

**[U32] Element Data**

**[TF] Color?**

**[I32] Buffer address** Linear address, from the "Map buffer" VI, of the buffer from which to retrieve the data. Buffer must have been previously defined, mapped and queued.

**[F+I] error in (no error)** error in describes an error that you want to check. If you leave error in unwired, this VI checks error code for errors.

**[TF] status** The status boolean is either TRUE (cross) for an error, or FALSE (checkmark) for no error or a warning.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

**[I32] code** The code input identifies the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

**[abc] source** The source string describes the origin of the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

**[U32] Color data** Array containing image data. Size will match that of "Array holder". Each element is a 32 bit integer containing 8 bits each of R, G and B data, 0 padded. This array can be wired to the "Array to IMAQ image" vi.

**[U32] Element Data**

**[F+I] error out** error out is a cluster containing the same information as in status out, code out, and source out. It has the same structure as error in.

**[TF] status** The status boolean is either TRUE (cross) for an error, or FALSE (checkmark) for no error or a warning.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

**[I32] code** The code input identifies the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.



**source** The source string describes the origin of the error or warning reported.

The pop-up option Explain Error (or Explain Warning) gives more information about the error displayed.

## 7. Technical support:

Please contact us if you have any questions, or have encountered any problems with our software.

**The Cooke Corporation**  
**1091 Centre Rd. - Suite 100**  
**Auburn Hills, MI**  
**48326-2670**

**(248) 276 8820**  
**(248) 276 8825 fax**

[service@cookecorp.com](mailto:service@cookecorp.com).

Check our website for updated information:

[www.cookecorp.com](http://www.cookecorp.com)

Copyright 2000, The Cooke Corporation Ltd.

LabVIEW is a trademark name of National Instruments Inc.