HAMAMATSU

PHOTONIC SYSTEMS

A DIVISION OF HAMAMATSU CORPORATION

Video Capture Library for LabVIEW

December 2012

Version 3.1

HAMAMATSU

Homepage Address http://www.hamamatsu.com

HAMAMATSU PHOTONICS K.K., Systems Division
812 Joko-cho, Hamamatsu City, 431-3196, Japan, Telephone: (81)53-431-0124, Fax: (81)53-435-1574, E-mail:export@sys.hpk.co.jp

U.S.A. and Canada: Hamamatsu Photonic Systems: 360 Foothill Road, Bridgewater, N.J. 08807-0910, U.S.A., Telephone: (1)908-231-1116, Fax: (1)908-231-0852, E-mail: usa@hamamatsu.com
Germany: Hamamatsu Photonics Deutschland GmbH: Arzbergerstr. 10, D-82211 Herrsching am Ammersee, Germany, Telephone: (49)8152-375-0, Fax: (49)8152-2658, E-mail: info@hamamatsu.de
France: Hamamatsu Photonics France S.A.R.L.: 8, Rue du Saule Trapu, Parc du Moulin de Massy, 91882 Massy Cedex, France, Telephone: (33)1 69 53 71 00, Fax: (33)1 69 53 71 10, E-mail: infos@hamamatsu.fr
United Kingdom: Hamamatsu Photonics UK Limited: 2 Howard Court, 10 Tewin Road Welwyn Garden City Hertfordshire AL7 1BW U.K., Telephone: (44)0 1707-294888, Fax: (44)0 1701-325777, E-mail: info@hamamatsu.co.uk
North Europe: Hamamatsu Photonics Norden AB: Smidesvägen 12, SE-171-41 Solna, Sweden, Telephone: (46)8-509-031-00, Fax: (46)8-509-031-01, E-mail: info@hamamatsu.se
Italy: Hamamatsu Photonics Italia S.R.L.: Strada della Mois, 1/E 20020 Arese (Milano), Italy, Telephone: (39)02-935 81 733, Fax: (39)02-935 81 741, E-mail: info@hamamatsu.it

Overview

The Hamamatsu Video Capture Library software is a collection of VIs designed to control and gather image data from a Hamamatsu camera from within LabVIEW. Because this is built using Hamamatsu's DCAMAPI, applications created with this library will be able to detect and use many of the Hamamatsu line of cameras. And because it is made for LabVIEW, a programmer will be able to create large virtual instruments in a relatively short amount of time.

This library can be used with IMAQ Vision from National Instruments. IMAQ Vision is a powerful image processing library that is made for LabVIEW. IMAQ Vision is not required to use this software.

It is understood that the user is familiar with the Hamamatsu camera(s) and its functions. If you are unfamiliar with some of the capabilities of the camera, please refer to your camera manual. You will find technical specifications and installation instructions. If you encounter a problem with your camera that you cannot solve, please contact our support group.

It is also understood that the user is familiar with National Instruments LabVIEW. If you need further assistants with LabVIEW, please refer to your LabVIEW manual or contact National Instruments.

System Requirements

- Microsoft Windows XP or higher
- Hamamatsu camera with compatible DCAMAPI module
- National Instruments LabVIEW 2011 or higher

What's New In Version 3.0

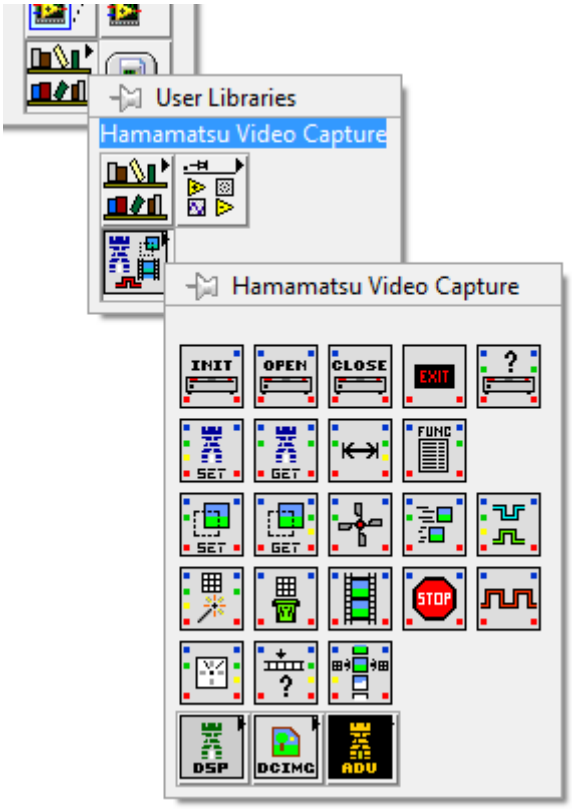
This version of the Video Capture Library has updated the VIs to include additional functions that provide you additional controls. Preparing a capture session and starting a capture session are now to different functions. Waiting for a new frame to arrive and collecting the data into LabVIEW are also two different functions. These changes have been made to allow for maximum performance so that you can collect all of your frame data from some of our more demanding camera systems.

Also included are functions for the new Hard Disk Recording feature. These functions allow you to capture data and write the frames directly to the hard disk into DCIMG format. This method is ideal for capturing a large number of frames in the most efficient way possible. These DCIMG files can be read back later with the functions in this library.

Installation

Hamamatsu Video Capture Installation

1. Be sure that LabVIEW is not running. If so, close all LabVIEW windows. Also be sure that your camera drivers have been properly installed. If not, please see the Camera Driver Installation below.
2. Run the SETUP.EXE program that comes with the installation software. For the 64-bit version of LabVIEW, run the SETUP_X64.EXE program instead to install the 64-bit version.
3. Follow all of the prompts during the installation process.
4. If you have LabVIEW 2011 or higher, the setup will install all of the proper files.
5. To access your new video capture functions, just follow the following diagram to locate the library.

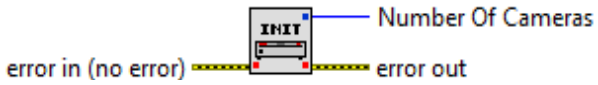


Camera Driver Installation

1. If you do not have a DCAM-API driver disk, you can download the camera driver from http://www.dcamapi.com and unzip the contents to a new directory.
2. If the CD does not automatically run, launch SETUP.EXE from the root directory.
3. Select and click the button from the menu of the interface that your camera is using.
4. Follow the on-screen instructions to install the driver you selected.

Standard Functions

TM_INITIALIZE



This is the first function that you need to run before you can use any camera control functions in the library. Its purpose is to initialize all of the required resources for the video capture library, find all supported camera on the system, and returns the number of cameras available. Any camera you wish to use with your system must be connected to your system and powered on before this function is called.

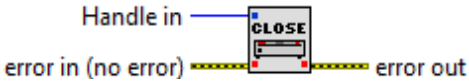
TM_DEINITIALIZE

This will cleanly free all resources used to by the video capture library. You will not be able to open any additional cameras after this function is called. It is recommended to stop any image acquisitions and close all cameras before calling this function.

TM_OPENCAMERA

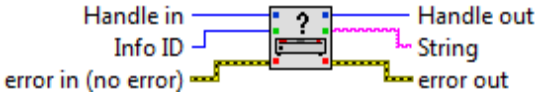
This will activate a camera for operation at the specified Camera Index and will return a handle for that camera. This handle must be used by the other camera control functions to determine the camera to be controlled. The index values start at 0, so if there are 3 cameras available on the system, the valid input for Camera Index would be 0, 1, and 2. This function can only be used to activate cameras that were found by TM_INITIALIZE. The default camera index is 0.

TM_CLOSECAMERA



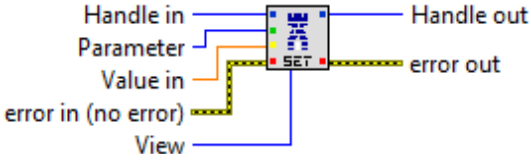
This will close the camera associated with the handle and free resources that it used. Once this function is called, the handle will become invalid and cannot be used anymore. The camera can still be opened again with TM_OPENCAMERA.

TM_GETCAMERAINFO



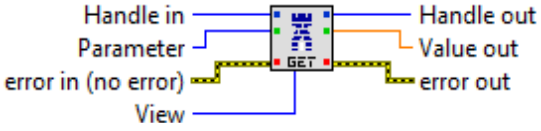
This function will return information about the camera. The possible items to get information are the camera model, camera ID, camera bus, vendor, driver version, camera version, module version, and DCAMAPI version. This function can also be used on unopened cameras by using the camera index instead of the camera handle.

TM_SETPARAMETER



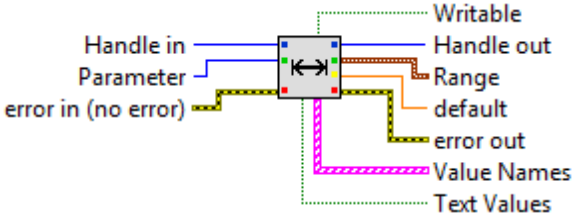
This function can adjust the camera settings. These settings may include gain, offset, exposure time, binning, etc. Not all cameras are capable of every listed function. A list of functions that can be controlled by this VI can be obtained with TM_GETFUNCTIONLIST. While you can adjust some settings during a capture, settings that can change the image resolution should not be changed during a capture. For cameras with multiple views, View will select which will be set. Please refer to your camera manual to determine the capabilities of your camera.

TM_GETPARAMETER



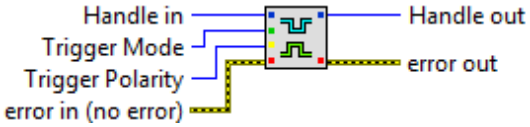
This function gets the current value of a selected camera setting. Not all cameras are capable of every function. For cameras with multiple views, View will select which value to read.

TM_GETRANGE



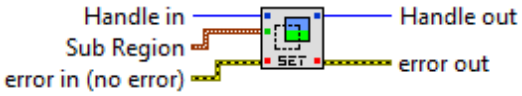
This function returns the range of the input parameter. The values you will receive will include minimum, maximum, stepping, and default values. If the parameter is a mode function, it will haves associated text values and the Text Values output will return true. If that value returns true, then Value Names will contain an array of text values for this function along with their associated number values.

TM_SETTRIGGER



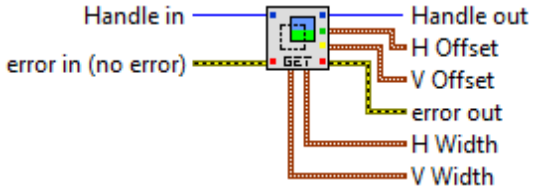
Hamamatsu cameras allow you to change the trigger mode and trigger polarity to fit your application. Please refer to your camera manual to see what modes are available.

TM_SETAREA



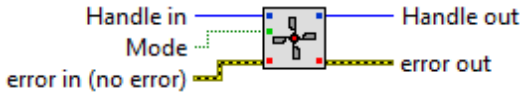
Some Hamamatsu cameras are capable of selecting a specific sub-region of the camera to capture image data from. With this function, you will be able to select the area you wish you capture through the Sub Region cluster input. Because of the limitations of the different cameras, the values you enter may not be the same values that were set. It is important to use TM_GETAREA to see the values that were set to the camera. This should not be called while capturing images.

TM_GETAREA



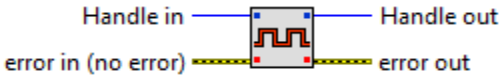
This function will return the value of the current Sub Region. This function will also return the current minimum, maximum, and increment values of each variable. These limits can change whenever you set a new sub-array or binning. It is always a good idea to call this function to get the current valid limits.

TM_SETCOOLING



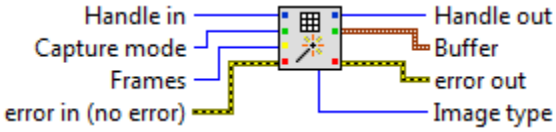
If your camera comes with a controllable cooling fan, it will be possible to power the unit on or off. If MODE is set to TRUE, the fan will turn on. If MODE is set to FALSE, the fan will turn off.

TM_FIRETRIGGER.VI



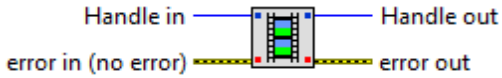
If your camera supports software trigger and is currently in software trigger mode, no images will be acquired until a software trigger is fired. When a software trigger is fired with this function, the camera will begin exposing a new image. This function is very useful for applications where timing is critical.

TM_PREPARECAPTURE



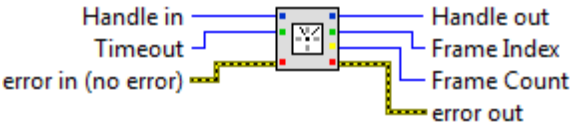
This function prepares the driver and camera for capture. There are two inputs available to configure your capture function. CAPTURE MODE allows you to adjust the behavior of the capturing cycle. In Snap mode, the camera will capture the specified number of frames and will stop at the end of the cycle. In Sequence mode, the camera will collect data continuously without stopping at the end of the cycle. Once it reaches the end of a cycle of frames, it will continue on the first frame. FRAMES allow you to adjust the number of frames of images you wish to capture. In Sequence mode, setting frames is unnecessary and will automatically be set to 3. This function will also return an Image Type value which is used by IMAQ Vision to determine the image data type. This function does not start the camera to capture images.

TM_STARTCAPTURE



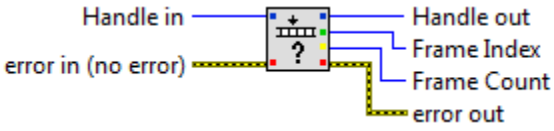
This begins capturing of image data from the camera to your computer. You will need to have called TM_PREPARECAPTURE to setup the driver for capture before calling this function.

TM_WAITNEXTFRAME



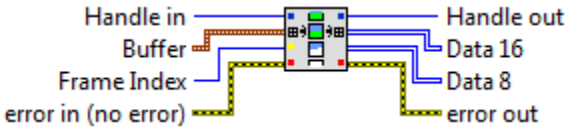
This function will wait for the next frame to become available. When a frame arrives, this function will return and you are able to work with the data. By default, the timeout is set to exposure time + readout time + 1 second in internal mode. In external and software trigger modes, the default timeout is infinite. The timeout value can be adjusted by setting the timeout input with a new value. The frame index is the buffer index location where the data was stored. Frame count is the number of frames captured since calling TM_STARTCAPTURE.

TM_GETCAPTUREINFO



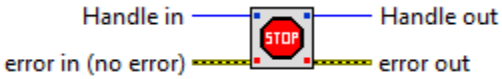
This function will return the current frame index and frame count of the capture sequence. The frame index is the buffer index location where the data is stored. Frame count is the number of frames captured since calling TM_STARTCAPTURE.

TM_GETFRAME



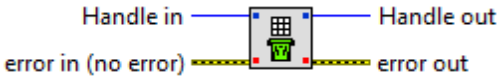
This function will retrieve the frame in the buffer at the specified frame index. This function requires an array input for the data buffer created by TM_PREPARECAPTURE. If the output data is 16 bit, the Data 16 output pin will be valid and contain the data in a U16 array. If the output data is 8 bit, the Data 8 output pin will be valid and it will contain the data in a U8 array.

TM_STOPCAPTURE



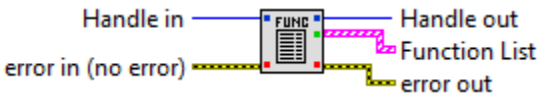
This function will stop the current capture session. The current capture mode and internal buffers create will remain intact so you will be able to start a new capture session immediately with TM_STARTCAPTURE.

TM_UNPREPARECAPTURE.VI



This function will free all resources created with TM_PREPARECAPTURE. This step is important if you wish to change certain camera settings such as the binning or region of interest. If the camera is capturing data, you must stop it first with TM_STOPCAPTURE before calling this function.

TM_GETFUNCTIONLIST.VI

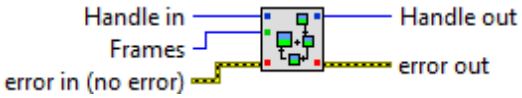


The Hamamatsu camera line is expanding with an ever growing number of functions. This VI will output an array which contains a list of the supported functions available to the camera. This array can be connected directly to the StringsAndValues[] property of a menu ring. The list of supported functions will vary depending on the camera. Additional information of these functions can be obtained from TM_GETRANGE. And the function values can be controlled with TM_GETPARAMETER and TM_SETPARAMETER.

DSP Functions

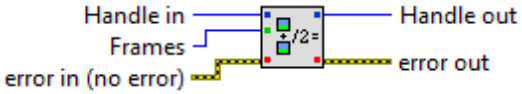
Some newer Hamamatsu cameras have imaging processing functions built into the camera hardware. Since these tools have been placed in the hardware, you will not need to use computer resources to perform some of the more common image processing tasks. Please check your camera manual to determine if your camera has these capabilities.

TM_RECURSIVEFILTER.VI



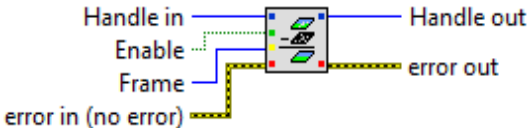
This DSP function allows you enable recursive filtering. Each output image of the camera will be the average of the number of frames specified by the Frames input. You can disable this function by setting Frames to 1.

TM_FRAMEAVERAGING.VI



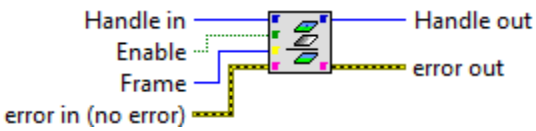
This DSP function allows you enable frame averaging. Each output image of the camera will be the average of the number of frames specified by the Frames input. You can disable this function by setting Frames to 1.

TM_SETBACKGROUNDFRAME.VI



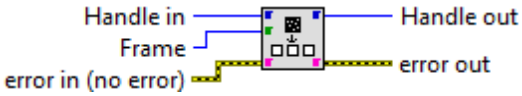
This DSP function allows you to enable a background image to be subtracted from the output data. You must specify the background frame you wish to subtract with the Frame input.

TM_SETSHADINGFRAME.VI



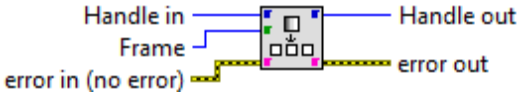
This DSP function allows you to enable a shading correction image to be applied to the output data. You must specify the correction frame you wish to apply with the Frame input.

TM_STOREBACKGROUNDFRAME.VI



This DSP function will store a frame of data into the camera memory frame specified by the input Frame for background subtraction. This function will fail if the camera is capturing images.

TM_STORESHADINGFRAME.VI

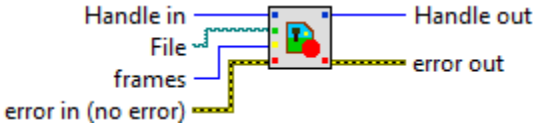


This DSP function will store a frame of data into the camera memory frame specified by the input Frame for shading correction. This function will fail if the camera is capturing images.

Hard Disk Recording Functions

New to the Hamamatsu Video Capture Library software are the hard disk recorder functions. These functions allow you to capture image data directly to disk. This is especially useful if you are capturing a large amount of data but do not have enough RAM to store the data. The output data is written to an DCIMG file which is a proprietary Hamamatsu image stack format. In order to view images from a DCIMG file, you can use the DCIMG reader functions available with this capture library or you can use any other program designed to read DCIMG files.

TM_STARTRECORDER.VI

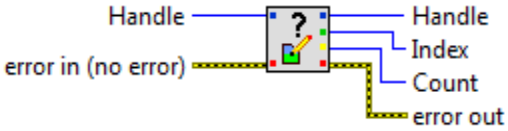


This function will enable the hard disk recorder. The recorder will write captured images directly to disk to the specified DCIMG file. This recorder will write up to the specified number of frames. Once that maximum number of frames have been reached, the recording will stop. This function must be called after TM_PREPARECAPTURE.

TM_STOPRECORDER.VI

This function will end the current recorder session and close the DCIMG file. Once the DCIMG file is closed, it may be opened by a DCIMG reader including the DCIMG reader functions available with this capture library.

TM_GETRECORDERSTATUS.VI



This function will provide the current index of the newest frame recorded as well as the total number of frames recorded.

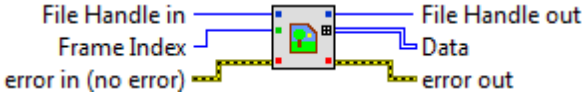
TM_OPENDCIMGFILE.VI

This function will open a DCIMG file for reading. It will return a file handle which will be used for all other DCIMG related functions. This file handle is different from the camera handle and should not be used for camera control functions. It will also return the total number of frames that are available in the DCIMG file. Once a DCIMG file is opened with this

TM_CLOSEDCIMGFILE.VI

This function will close a DCIMG file and free any resources used for it. The DCIMG file will be available to be opened by another reader. This will make the file handle invalid and unusable by the other functions.

TM_GETDCIMGFRAME.VI



This function will retrieve the frame data from a DCIMG file specified by the file handle and the frame index.

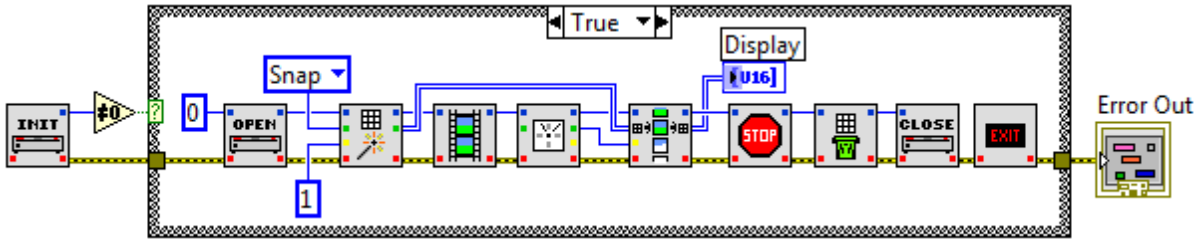
Error Message

When using this library, you may encounter an error when running a LabVIEW program. If the error was generated from one of the functions supplied in this library, then you will receive the error code as well as the particular VI that caused the error. This information will provide a clue into determining why the program failed and how it could be corrected. The following is a list of possible error messages that you may encounter.

TMERR_ABORT – The operation has been aborted.
TMERR_BUSY – The camera is currently busy and cannot execute your request.
TMERR_FAILEDOPENRECFILE – Failed to open the specified DCIMG file.
TMERR_FAILEDREADDATA – Failed to read from the DCIMG file.
TMERR_FAILEDWRITEDATA – Failed to write to the DCIMG file.
TMERR_FAILOPENCAMERA – The system was unable to active the camera.
TMERR_FAILREADCAMERA – The camera could not read the command.
TMERR_FAILWRITECAMERA – The camera could not change the parameter.
TMERR_INVALIDCAMERA – The selected device is invalid.
TMERR_INVALIDHANDLE – The specified camera handle is invalid.
TMERR_INVALIDPARAM – The specified parameter is invalid.
TMERR_INVALIDPROPERTYID – The property ID is invalid.
TMERR_INVALIDRECHANDLE – The DCIMG file handle is invalid.
TMERR_INVALIDSUBARRAY – The subarray setting is invalid.
TMERR_INVALIDVALUE – The specified value is invalid.
TMERR_LOSTFRAME – A frame has been lost during transfer from the camera.
TMERR_NOCAMERA – No camera is available.
TMERR_NOMEMORY – Your computer does not have enough memory for your request.
TMERR_NORESOURCE – Your computer does not have enough resources for your request.
TMERR_NOTREADY – The camera is not ready for the command.
TMERR_NOTWRITABLE – The property does not have a writable value.
TMERR_NOTREADABLE – The property does not have a readable value.
TMERR_NOTSUPPORT – The function is not supported by the camera.
TMERR_OUTOFRANGE – The value is out of range.
TMERR_TIMEOUT – The driver has timed out while waiting for the frame data.
TMERR_UNKNOWNERROR – An unknown error has occurred.

Samples

Single Frame Acquisition

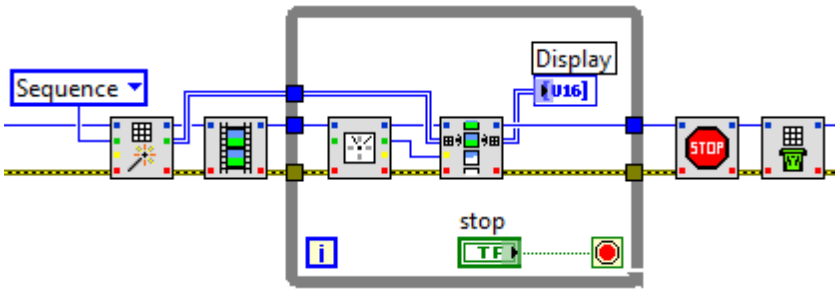


In this example, we are initializing the library by calling TM_INITIALIZE to find all cameras in the system. If it finds at least one camera, we call TM_OPENCAMERA with a camera index of 0 to activate the first camera that is found. If successful the function will return a handle to the camera. This handle will be use for all of the other functions.

To acquire a single image, TM_PREPARECAPTURE needs to be called to setup the capture. We set the capture mode to Snap and we set only 1 frame. Then we call TM_STARTCAPTURE to begin the capture session. TM_WAITNEXTFRAME is called to wait for a new frame to be available. Once that function returns, we call TM_GETFRAME. We take the Frame Index output from TM_WAITFRAME to determine which frame we are going to retrieve from the capture library. The output 2D array will contain the data for the image. We then take this data and send it to the display.

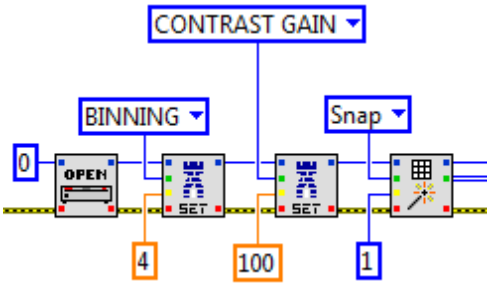
To clean up the acquisition, we first call TM_STOPCAPTURE to stop the capture process. Even though this is a snap of 1 image, this is still a necessary step. Then we call TM_UNPREPARECAPTURE to free the internal image buffers. Then TM_CLOSECAMERA is called to release the camera and all of the resources that were allocated for it. And finally TM_DEINITIALIZE is called to unload the capture library.

Preview Capture



Setting up for a preview is very similar to acquiring a single image. After opening the camera with TM_OPENCAMERA, we need to start a sequence mode capture with TM_STARTCAPTURE. For inputs, you will only need to set capture mode to Sequence as the frame buffer will automatically be set for you. After the capture is started, we can enter our while loop so we can continuously capture images. TM_WAITNEXTFRAME will wait for the next frame TM_GETFRAME will retrieve the next frame from the buffer.

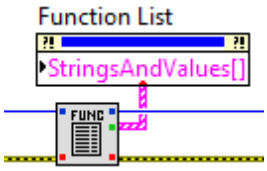
Adjusting Camera Settings



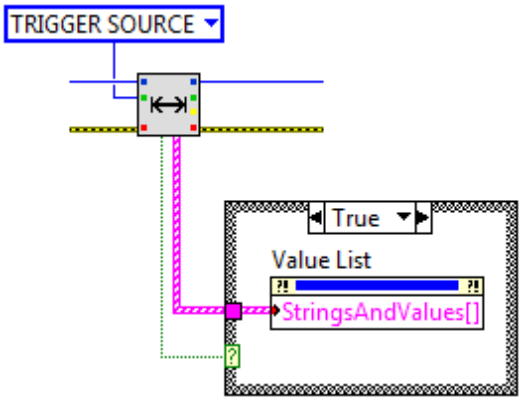
In this example, we use TM_OPENCAMERA to create a handle to the Hamamatsu camera. We use TM_SETPARAMETER to adjust binning mode. For the Parameter input, we use BINNING. And for the Value In input, we use 4 to specify that we want 4x4 binning. We use TM_SETPARAMETER again, but this time to adjust the gain. For the Parameter input, we use CONTRAST GAIN. For the Value In input, we use 100 to specify that we want to have a gain of 100. Now that we have the settings that we desire, we can then prepare a capture session. Some functions can be adjusted during a capture, however functions that can change the image resolution can not be changed during a capture. Please refer to your camera manual to determine the capabilities of your camera.

Function Enumeration

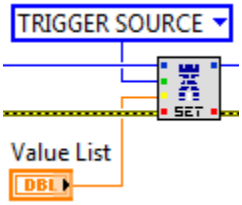
If you are creating a VI capable of handling different models of Hamamatsu cameras, you may choose to enumerate the available functions of the cameras. If you use TM_GETFUNCTIONLIST with TM_GETRANGE you will be able to see what functions are available to the camera.



In the above example, the output of TM_GETFUNCTIONLIST is being used to set the StringsAndValues[] property of the menu ring item Function List. This will populate the menu ring with all of the functions available to this camera and associate them with the proper ID value. You may also use the dynamically created Function List menu ring with TM_GETRANGE. In addition, if the function is a mode function, you can get the available list of mode values for the function using TM_GETRANGE. Just as in TM_GETNFUNCTIONLIST, this output can be used to set the StringsAndValues[] property of a menu ring.

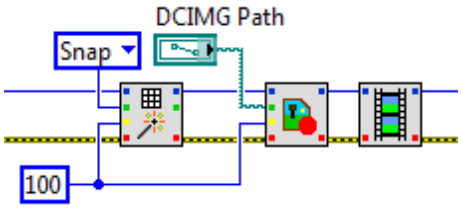


The dynamically created Function List and/or Value List menu rings can be used with TM_SETPARAMETER to control the camera. Be sure that you are using the correct Value List for the function that you are trying to control.

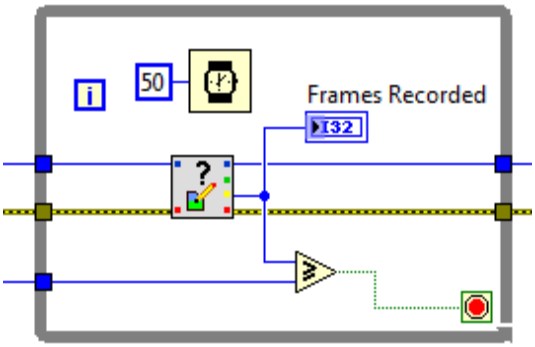


Recording to a DCIMG File

It is possible to acquire data from the camera and record it directly to the hard drive. Some Hamamatsu cameras are very demanding on the computer that you may not have enough system resources to acquire all of your frames through the standard method. Recording directly to the hard drive removes the need for the data to be processed in LabVIEW first thus allowing more resources to be used to collect the data.



To start a recording, you must first prepare a capture session with TM_PREPARECAPTURE. You can either snap for the number of frames you wish to record, or set a sequence. Depending on the number of frames you are recording it may be best to set a sequence since this will only create 3 image buffers. If you are recording a large number of frames, you may not have enough RAM. After that function is called, you may now call TM_STARTRECORDER with the file path and the number of frames to record. Now when images are being captured, they will write directly to the file that you specified.

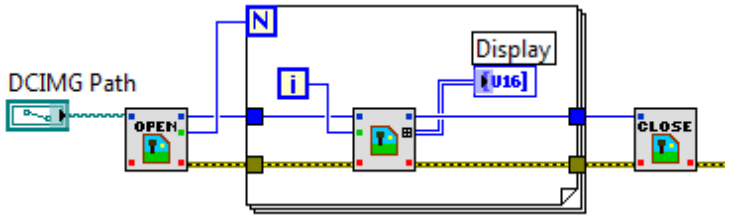


While it is capturing data, in order to determine the number of frames that were recorded, you will need to use TM_GETRECORDERSTATUS. This will provide the total number of frames recorded. This will help you to determine if the recording is completed.

Once the recording is done, you should call TM_STOPRECORDER. This will close the DCIMG file and allows it to be opened by a DCIMG reader. This function can be called anytime during capture.

Reading a DCIMG File

If you have data that you wish to read from a DCIMG file, you can use the functions available with this capture library. These functions are separate from the main camera control functions and can be used without a camera connected to the computer.



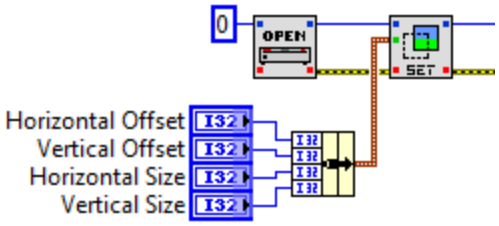
To read a DCIMG file, you will first need to open the file with TM_OPENDCIMGFILE and provide the complete file name and path. This will give you the file handle as well as the number of frames available in the DCIMG file. The file handle must be used with the other reader functions.

Once the file is open, you can begin reading the images that are contained in the file by using TM_GETDCIMGFRAME. This function requires the file handle provided by TM_OPENDCIMGFILE and a frame index. The output will be a 2D array of the image data.

When you are finished with the file, you can close the file with TM_CLOSEDCIMGFILE. This will free any resources used to open the file as well as allow any other DCIMG reader to open that file.

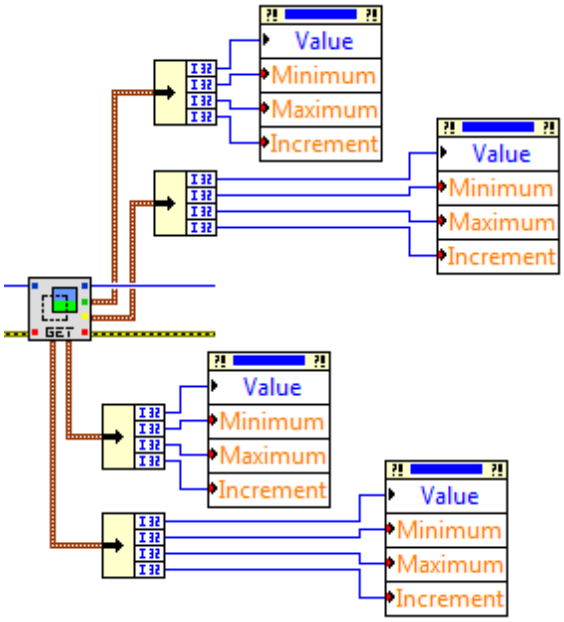
Setting Region of Interest

The region of interest can be setup in different ways. To set the region, create a cluster of four 32-bit numbers and use this cluster for the Sub Region terminal of TM_SETAREA. These values are horizontal offset, vertical offset, horizontal size, and vertical size respectfully. You can set the region of interest when the capture session has not been prepared yet.



Getting Region of Interest

To get the current region of interest or to determine the acceptable values for TM_SETAREA, you can use TM_GETAREA. Each of the four outputs are a cluster of four 32-bit values that represent the limits of that variable. In this example, the different values are being used to set the limits of the sub-region controls.



Parameter Definitions

Some of the capabilities of the Hamamatsu cameras are mode functions. These modes can be changed with this software, however the names of these modes are not always defined. Below is a list of the different mode functions and the definition of their values. Not all cameras are capable of each function or mode. Please refer to your camera manual for more information.

FUNCTION	VALUE	DEFINITION
BINNING	1	1x1
	2	2x2
	4	4x4
	8	8x8
CCD MODE	1	NORMAL CCD
	2	EM CCD
SENSOR MODE	1	AREA
	3	SLIT
	4	TDI
LIGHT MODE	1	LOW LIGHT
	2	HIGH LIGHT
HIGH DYNAMIC RANGE MODE	1	OFF
	2	ON
TRIGGER SOURCE	1	INTERNAL
	2	EXTERNAL
	3	SOFTWARE
TRIGGER ACTIVE	1	EDGE
	2	LEVEL
	3	SYNCREADOUT
	4	POINT
TRIGGER MODE	1	NORMAL
	3	PIV
	6	START
TRIGGER POLARITY	1	NEGATIVE
	2	POSITIVE
TRIGGER GLOBAL EXPOSURE	1	NONE
	2	ALWAYS
	3	DELAYED
	4	EMULATE

FUNCTION	VALUE	DEFINITION
FIRST TRIGGER BEHAVIOR	1	START EXPOSURE
	2	START READOUT
OUTPUT TRIGGER POLARITY	1	NEGATIVE
	2	POSITIVE
SENSOR COOLER FAN	1	OFF
	2	ON
MECHANICAL SHUTTER	1	AUTO
	2	CLOSE
	3	OPEN
FRAMEBUNDLE MODE	1	OFF
	2	ON