

# CCDSP Viewer Application Users Manual



Revision 1.2.2 July 31, 2009



## Table of Contents

<b>PREFACE</b> .....	<b>5</b>
<b>REVISION HISTORY</b> .....	<b>5</b>
<b>GETTING STARTED</b> .....	<b>5</b>
<b>Installation</b> .....	<b>5</b>
<b>Overview</b> .....	<b>8</b>
<b>WINDOWS</b> .....	<b>8</b>
<b>Main Window</b> .....	<b>8</b>
Status Bar.....	9
File Menu.....	11
Edit Menu.....	11
Plugins Menu.....	12
View Menu.....	12
Windows Menu.....	13
Tools Menu.....	14
Help Menu.....	14
<b>Snapshot Viewer Window</b> .....	<b>15</b>
Display Tab.....	15
Tools Tab.....	16
<b>Plot Viewer Window</b> .....	<b>16</b>
Display Tab.....	18
Tools Tab.....	18
<b>DIALOGS AND DOCK WIDGETS</b> .....	<b>18</b>
<b>Record/Playback dialog</b> .....	<b>18</b>
<b>Connect Dialog</b> .....	<b>19</b>
USB Connection.....	20
Ethernet Connection.....	20
Serial Number Connection.....	21
<b>Snapshot Control Dialog</b> .....	<b>21</b>
Background Subtraction.....	22
Test Pattern.....	23
Number of Frames.....	24
Capture Controls.....	25
Timing Control.....	25
Area Control.....	26
Binned Image Summary.....	26
<b>Binning Configuration</b> .....	<b>26</b>

## CCDSP Viewer Users Manual

Vertical Area Definition .....	28
Horizontal Area Definition .....	29
Bin Setup Display and Miscellaneous Functions.....	30
<b>Image Statistics Dialog .....</b>	<b>31</b>
Plot Statistics .....	31
<b>Peak Finder Dialog.....</b>	<b>32</b>
<b>Link Test Dialog .....</b>	<b>32</b>
<b>Calibration Dialog .....</b>	<b>33</b>
<b>Factory Configuration Dialog.....</b>	<b>34</b>
Hardware Params.....	34
Timing File .....	35
Substrate .....	35
Network .....	36
<b>I/O Board Test Dialog .....</b>	<b>37</b>
<b>HOW TO .....</b>	<b>37</b>
<b>DEVELOPER REFERENCE.....</b>	<b>37</b>
<b>CAMERA DLL API CALL REFERENCES .....</b>	<b>38</b>

## Preface

The CCDSPViewer application serves a few different purposes. First and foremost, it allows the user to exercise the various calls in the camera DLL (clcamiface3.dll on windows). The intent is to have the program exercise every feature in the DLL. This will allow the user to see how the camera API functions can be used in a real-world scenario. The CCDSPViewer application can also be used to configure and test the camera during development and production. Each application window will be described in its own section of this manual and the DLL calls it uses will be listed.

## Revision History

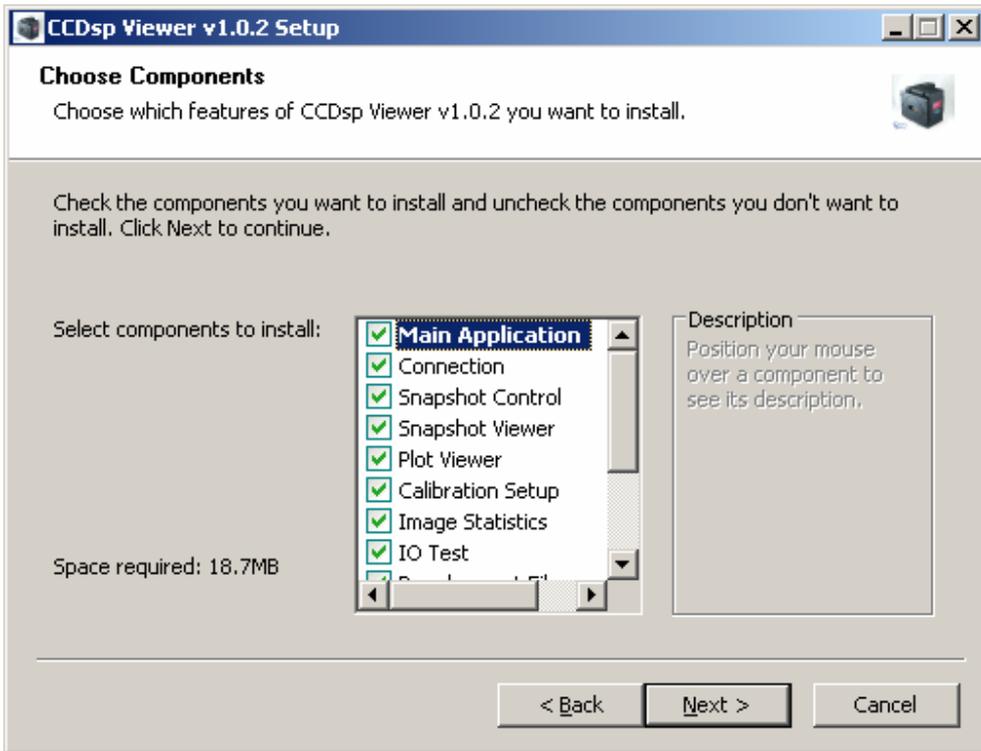
Revision	Date	Notes
-	6 July 2009	
1.1.0	16 July 2009	Updated for version 1.1.0 of the CCDsp Viewer.

## Getting Started

### *Installation*

Installing the CCDSP Viewer application is as simple as running the setup application (CCDsp\_setup.exe). This will install the application on your computer and create shortcuts to start it under the standard menus (**Start→Programs→Critical Link→CCDSP** on a windows based computer). During the installation, you will be able to select which plugin elements you wish to install. Some of the plugins are designed for factory and/or development use and will require keys to install.

The installer presents the user with a list of components to install (see Figure 1). The default set should be adequate for the majority of users. If you need the legacy application or wrapper DLL, be sure to select “Legacy Sample App”.



**Figure 1 Installer Components**

The installer will also pre-install the USB drivers by default, which will allow you to semi automatically load the correct drivers when the CCDsp camera is plugged into a USB port. There are actually 2 USB drivers that need to be installed for the CCDsp camera. When you first plug the camera in, you will see the “Found New Hardware Wizard” (twice). For each set of dialogs, select the following options

Dialog	Action
1	Select “No, not this time”.
2	Select “Install the software automatically”
3	Select “Continue Anyway”
4	Click “Finish”

**Table 1 CCDSP Loader Installation Dialogs**



1



2

# CCDSP Viewer Users Manual



3



4

**Table 2 CCDSP Driver Installation Dialogs**



1



2



3



4

Even if you select the “Configure Windows Firewall” option in the installer (the default), sometimes windows will alert you that the CCDspViewer application is trying to use the network (see Figure 1). If you see this dialog, select “Unblock” to allow the application to search for cameras on the Ethernet network. Note that it is the `clcamiface3.dll` DLL that is actually using the network, so your application (if you create one) will also need to be un-blocked to have the `CLLocateEnetCameras()` API call work.



Figure 2 Windows Firewall Alert Dialog

## Overview

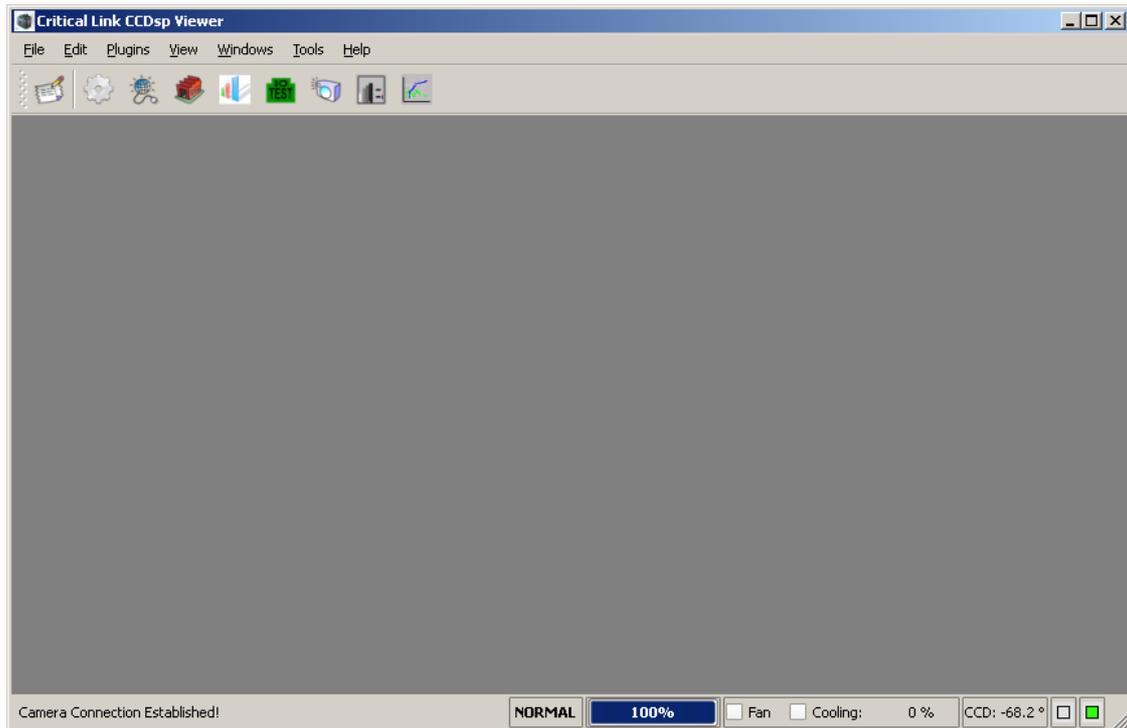
The CCDSP Viewer application is designed as a modular application. The main window provides the basic application shell and core user interface functions. Most of the features of the application are provided by plug-ins. When the application starts, it scans its installation directory for suitable plugins and loads them. These plugins provide various features (camera control, image viewing, data processing, etc). This setup allows the application to be tailored to fit the desired functionality without cluttering it up with features not required by the user (i.e. Internal factory configuration and test plugins are not installed users that just need to use the camera to collect data). The CCDSP Viewer application stores its settings in an ini file called “%APPDATA%\Critical Link LLC\CCDsp Viewer.ini”.

The %APPDATA% path is usually C:\Documents and Settings\\Application Data.

## Windows

### Main Window

The CCDSP Viewer application consists of a main window that serves as a container for the other windows in the program.



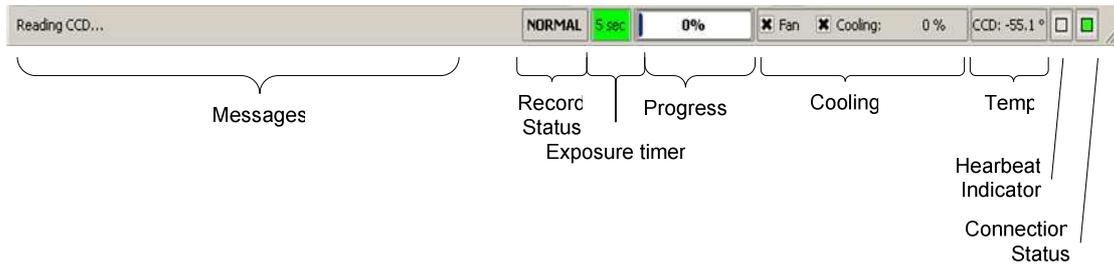
**Figure 3 Main Window**

It also serves as the main point of control for the camera interface. Many of the main camera callback interfaces are hooked into the main window; including `CLSetErrorCallback()`, `CLSetHeartBeatCallback()`, and `CLSetImageDataCallback()`. A plugin with a user interface will be either a dockable widget or a window. Plugins that are dockable widgets can be arranged by docking them (attaching them) to one of the edges of the main window, or left as floating windows that can be outside of the main window area. Window widgets (like the snapshot and plot view widgets) must remain inside the main application window (although they can be minimized, maximized, tiled, etc). Each loaded plugin will place an icon on the main

toolbar  (which can also be docked or left floating). The icons on the toolbar are used to show/hide the docking widgets, or create a new window. The main toolbar can shown/hidden from the “View” menu. Each menu in the application is described below.

## Status Bar

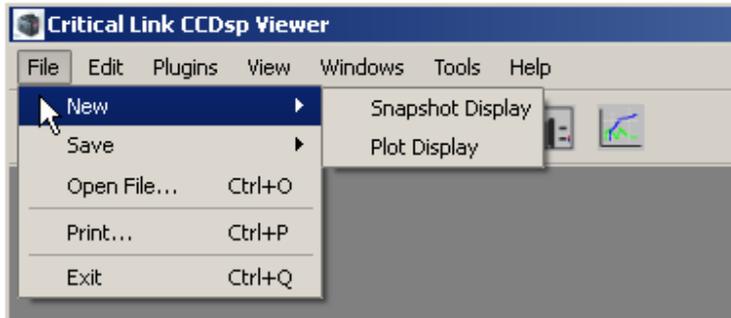
The status bar has a number of controls and indicators that are quite important.



**Figure 4 Main Window Status Bar**

Application messages are displayed on the left side of the status bar. Important messages are also sent to the log window. The application playback/record state is shown next. This field changes from NORMAL to RECORD or PLAYBACK based on the current state of the application and changes color accordingly. During camera exposure, the remaining exposure time is indicated by the exposure timer label (using information in the camera `tsHeartBeatData` structure). The progress bar is updated upon receipt of every heartbeat message from the camera by calling `CLGetReadProgress()`. The cooling controls allow you to control both the fan (using `CLEnableFan()`) and the TE cooler (using `CLSetCooling()`). There is also a small chart showing the relative amount of cooling being applied (as a percent). If the amount of cooling exceeds 95%, the chart will turn red (versus the normal green color). The actual temperature of the CCD sensor is shown next (in degrees Celsius). The heartbeat indicator blinks every time a heartbeat message is received from the camera, and will turn red if the camera stops sending them (while connected). The connection status indicator will be green if connected to a camera, or red if not.

## File Menu



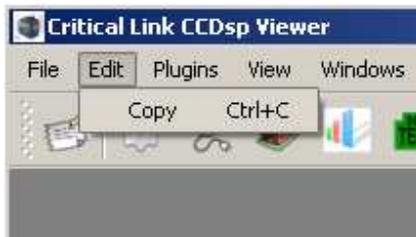
The File menu provides the standard sets of actions you would expect.

Figure 5 File Menu

Table 3 File Menu Actions

Action	Description
<b>New</b>	Provides a sub-menu allowing you to create a new instance of each type of window plugin. These actions are also present on the main toolbar.
<b>Save</b>	If the currently active window is capable of saving its data, this menu entry allows you to do so. It will be disabled if the active window does not support saving data.
<b>Open File...</b>	If the currently active window supports loading data from a file, this entry will do so. Its specific behavior depends on the active window and is documented therein.
<b>Print...</b>	This menu item will print the currently active window or the main window if no window is active.
<b>Exit</b>	Exits the application.

## Edit Menu



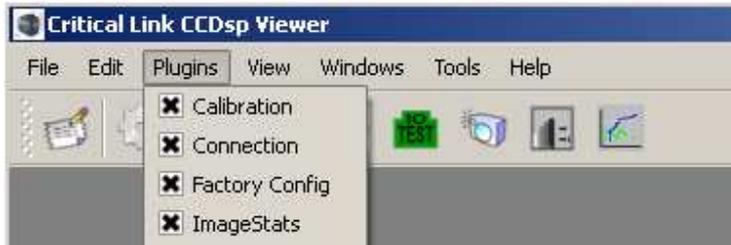
The Edit menu allows you to copy a window.

Figure 6 Edit Menu

Table 4 Edit Menu Actions

Action	Description
<b>Copy</b>	Copies the active window to the clipboard (or the main window if there is no active window). The default copy behavior is to copy an image of the window. If a specific window behaves differently, it will be documented in its section.

## Plugins Menu



**Figure 7 Plugins Menu**

checkbox next to them.

The plugins menu shows the currently loaded plugins. If a plugin is checked, it means that it is a dock widget plugin and its corresponding widget has been loaded. Plugins that provide windows do not have a

## View Menu



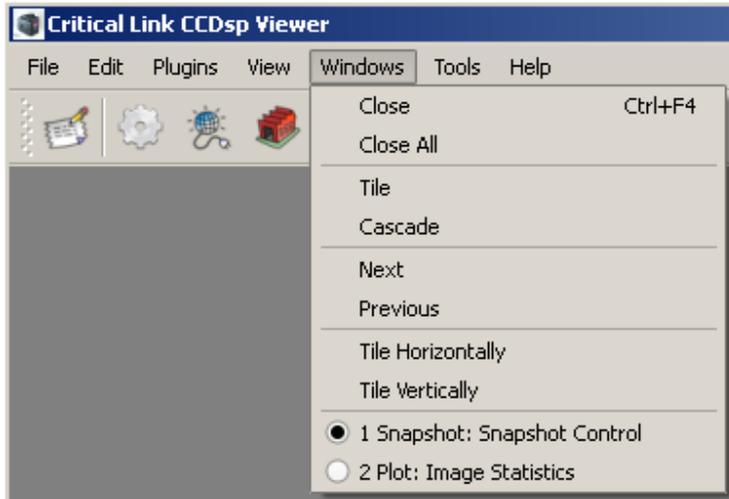
**Figure 8 View Menu**

The View menu allows the user to control various visual aspects of the application. Each item on the view menu is described in the table below.

**Table 5 View Menu Actions**

<b>Action</b>	<b>Description</b>
<b>Main Toolbar</b>	Shows and hides the main toolbar.
<b>Show Log</b>	Shows the application log window.
<b>Cameras</b>	Shows a report of the different types of cameras that have been connected to this application (for troubleshooting).
<b>Playback / Record</b>	Shows the Record/Playback dialog.
<b>Tab Windows</b>	When checked, the application will display the windows as a set of tabs. Uncheck this item to display them as “normal” windows.
<b>Style</b>	Displays a sub-menu allowing you to set the style (visual appearance) of the application. The default style is “Plastique”. The selected style will be remembered in the ini file for next time the program is run.

## Windows Menu



The windows menu provides the standard set of window manipulation actions you would expect in an application. Each item on the view menu is described in the table below. Note that the windows menu does not affect dock widgets.

Figure 9 Windows Menu

Table 6 Windows Menu Actions

Action	Description
<b>Close</b>	Closes the active window.
<b>Close All</b>	Closes all the windows.
<b>Tile</b>	Arranges all the windows in a tile pattern (favoring the active window).
<b>Cascade</b>	Arranges all the windows in a cascade pattern.
<b>Next</b>	Activates the next window. The windows are activated in the order in which they are created.
<b>Previous</b>	Activates the previous window. The windows are activated in the order in which they are created.
<b>Tile Horizontally</b>	Arranges the windows left to right using the full height of the main window area for each window. The windows are arranged in the order in which they are created.
<b>Tile Vertically</b>	Arranges the windows top to bottom using the full width of the main window area for each window. The windows are arranged in the order in which they are created.
<b>Window list</b>	Each window will be listed, allowing the user to directly activate one. The currently active window is indicated.

## Tools Menu



Figure 10 Tools Menu

The tools menu allows you to launch your own tools from within the CCDSP Viewer application. Selecting the “Add tool...” item prompts you to select an application or batch file

using the standard open file dialog box. If a valid file is selected, you are prompted to give the tool a name for the menu (it defaults to the file name). The program will use the application icon on the menu if available to help identify the tool to the user. The path to the tool is also shown in the main window status bar as it is highlighted in the menu. To remove a tool from the menu, right-click over its menu entry and select the “Remove Tool” option from the context menu. Information about items in the tool menu is stored in the application ini file.

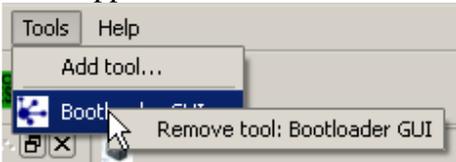


Figure 11 Remove tool menu

## Help Menu

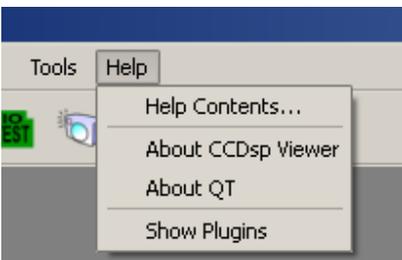


Figure 12 Help Menu

The Help menu provides access to this manual and various application information used when reporting an issue with the program. Each item on the help menu is described in the table below.

Table 7 Help Menu Actions

Action	Description
Help Contents...	Shows this manual
About CCDSP Viewer	Shows the about box for this application, including version information used for bug reporting.
About QT	Shows the QT about box (The CCDSP Viewer application is built using the QT application framework).
Show Plugins	Shows the list of loaded plugins and information about them. Plugins that did not successfully load are also shown in this dialog.

## Snapshot Viewer Window

The snapshot viewer window is the primary means of viewing data acquired by the camera. The window displays a spectrogram plot of the acquired data using a user-selectable colormap.

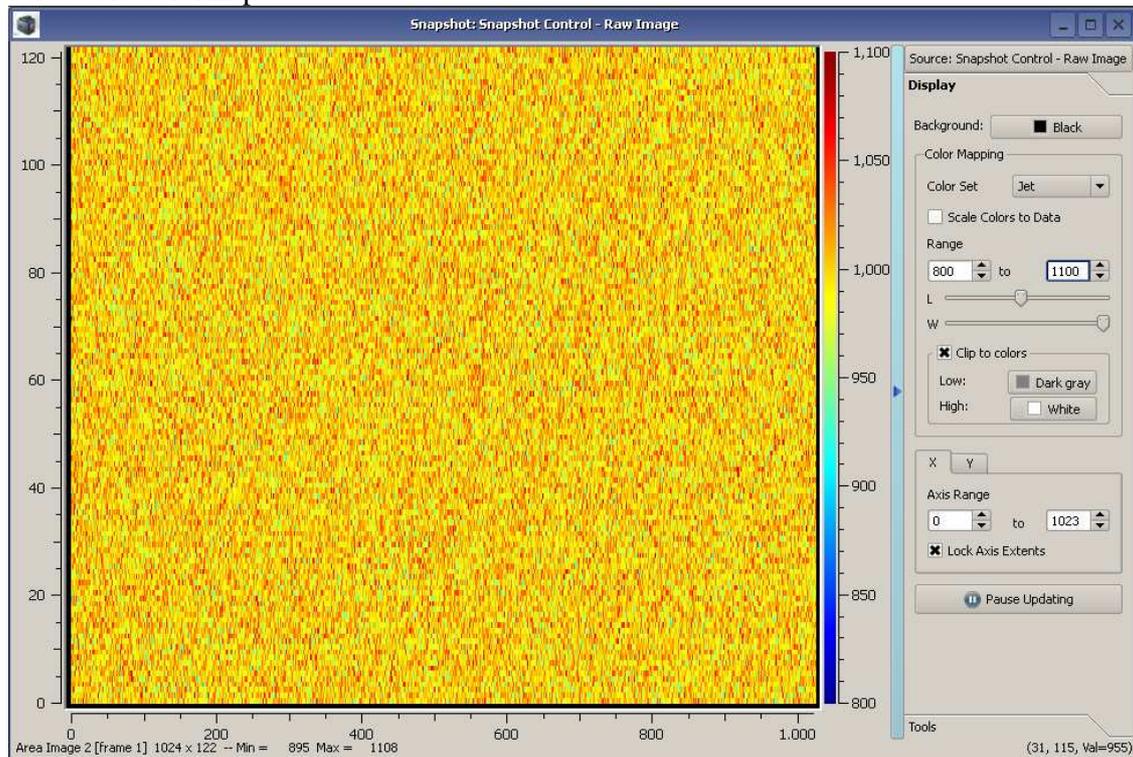


Figure 13 Snapshot Viewer Window

You create new snapshot windows from the **File**→**New**→**Snapshot Display** menu option. The blue bar to the right of the image display is used to show and hide the plot settings. At the top of the plot settings area is a menu button that allows you to select the source of the plot data. All plugins that can provide image data are displayed in this menu. Under each plugin, a sub-menu shows the different types of image data the plugin can provide for plotting. The window title will be changed to reflect the current data source and type. The image information (type, size, etc) is displayed in the lower left corner of the window. The position and value of the pixel under the cursor is displayed in the lower right corner of the window.

## Display Tab

Just below the data source selection is a toolbox with controls for modifying the data display. You can change the background color and the color mapping of the data. Several color maps are provided and can be chosen using the **Color Set** combo box. If you select the **Scale Colors to Data** checkbox, the colormap will be re-mapped to the data such that the extents of the colormap align with the extents of the data (this obviously uses more CPU resources, so you may want to disable this feature if acquiring images with a high

data rate, or large plots). You can also manually set the colormap limits using the **Range** controls (the image min/max values are shown as part of the image description at the bottom-left of the window). Below the **Range** controls are two sliders labeled **L** and **W** (which stands for Level and Window). These controls act like brightness and contrast. The level control adjusts the range center up and down, while the window control adjusts the range in or out. The last set of controls in the **Color Mapping** group is the **Clip to colors** group. When this setting is enabled, pixel values outside of the colormap extents are shown using the specified color. If this group is not enabled, pixel values outside of the colormap range are shown using the last color in the colormap (high or low accordingly).

Below the **Color Mapping** controls are the axis controls. For both the **X** and **Y** axis, you can specify the range of axis. If you deselect the **Lock Axis Extents** (the default is on), the snapshot viewer will scale the axis to the extent of the image every time a new image is loaded. The first time the snapshot viewer window loads an image, the x and y axis ranges are set to the extent of the image (active cols x active rows), unless the user has already set the range manually.

The last control in the **Display** group is the **Pause Updating** button. This is a toggle button, and when active, the snapshot display will not update when new images are acquired.

## **Tools Tab**

The **Tools** tab has the **Data Report** button that will create a csv version of the image data and display it in a text viewer. You can then save the text as a file to import into another application.

## ***Plot Viewer Window***

The plot viewer window allows you to plot data from any of the plugins that provide it. It can be used to view rows or columns of image data, various image statistics, and sensor data (temperatures, etc).

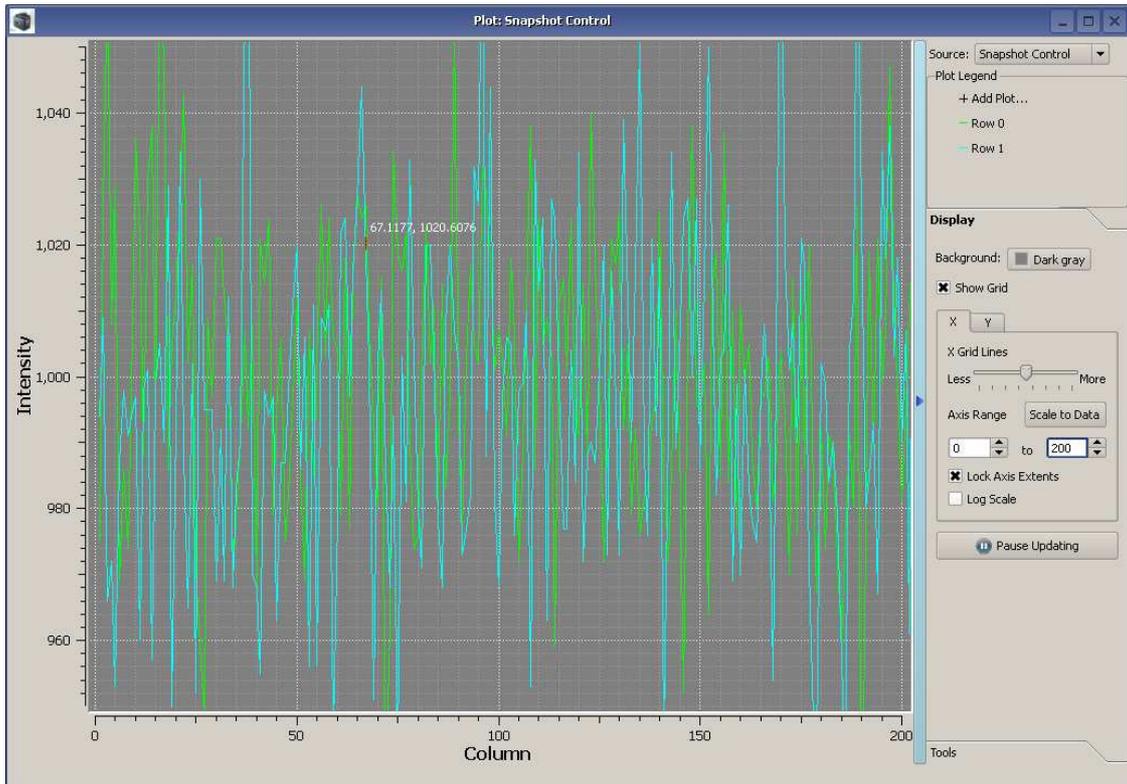


Figure 14 Plot Viewer Window

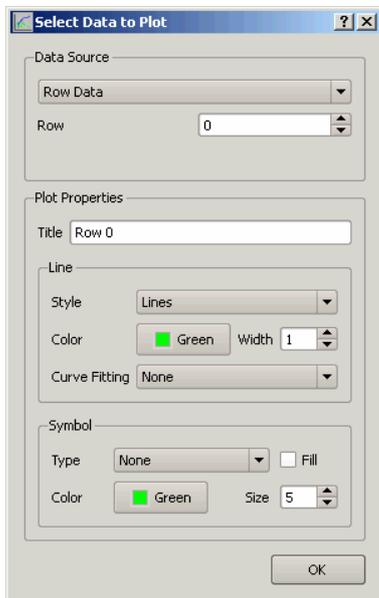


Figure 15 Plot Properties Dialog settings and Tools.

Each plot window can be tied to one data source using



the **Source** combo box . You

then click on **+ Add Plot...** in the **Plot Legend** area to bring up the plot selection dialog . The **Data Source** group varies based on the plugin selected to source the data, but works the same way for all plugins. There is a combo box showing the types of data available from the plugin, and a spin box to select the instance of that data (i.e. row number for image row data). This dialog also has the **Plot Properties** group where you can set the line and symbol styles for the curve. Clicking on the curve in the legend also brings up this dialog to allow you to edit the curve properties. Right clicking on the legend item brings up a menu where you can remove the item from the plot. Below the **Plot Legend** area is a toolbox for **Display**

## Display Tab

The **Display** tab has several controls affecting the data presentation. You can change the background color and the color of the plot, show/hide the grid and control the axis scales. Using the **X** and **Y** axis controls, you can specify the range of axis. If you select the **Lock Axis Extents** (the default is off), the plot viewer will not scale the axis to the extent of the image every time a new image is loaded. Pressing the **Scale to Data** button will perform an auto scaling without affecting the **Lock Axis Extents** setting. The **Log Scale** checkbox changes the given axis scale to a log10 based one.

The last control in the **Display** group is the **Pause Updating** button. This is a toggle button, and when active, the snapshot display will not update when new images are acquired.

## Tools Tab

The **Tools** tab has the **Data Report** button that will create a csv version of the image data and display it in a text viewer. You can then save the text as a file to import into another application.

You can control how the mouse interacts with the plot using the Mouse Mode group.



Figure 16 Mouse Mode Controls



Disables mouse interaction.



Uses the mouse to zoom. Left button to drag and define zoom region, right button to zoom back one.



Show crosshairs on display when left mouse button is down.



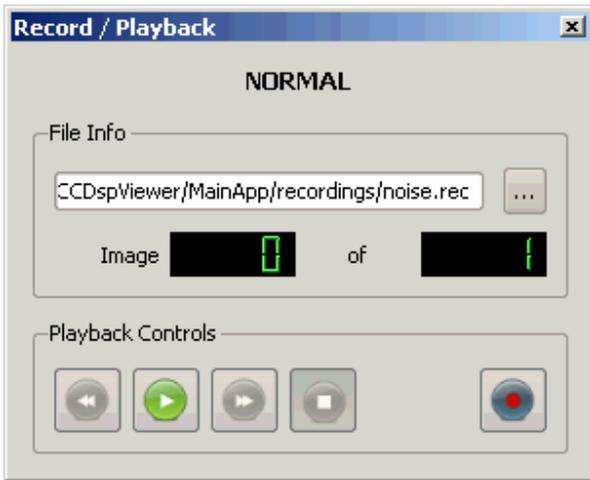
Find the nearest point on any curve and show it in the bottom right corner when left button is down.

## Dialogs and Dock Widgets

The CCDSP Viewer application uses several dock widgets to control various camera attributes. We will go over each of the dialogs individually first. Later in the manual there are some procedures outlined for completing common tasks.

### **Record/Playback dialog**

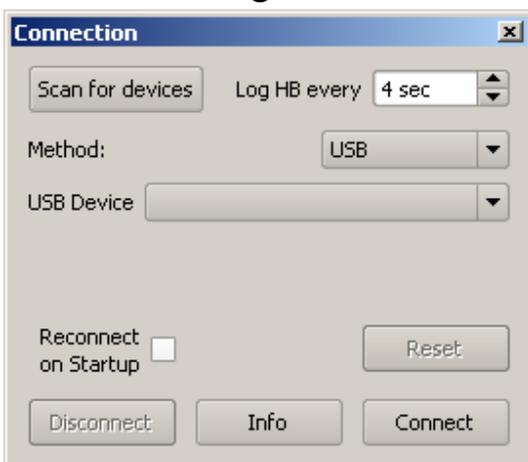
The Record/Playback dialog (shown in Figure 17)



**Figure 17 Record/Playback Dialog**

is used to create and play back record files. To select a record file, click the  button and either select an existing file (for playback) or create a new one (for record). The dialog will show how many images are in the record set. To start recording, press the  button (the dialog will prompt you if you will be overwriting an existing record set). Each image acquired is saved to disk before being passed to the rest of the application for processing. Similarly, images played back are injected as if they came from the camera DLL (before any application processing). To stop the recording or playback, push the stop  button. Also, during playback, the play  button changes to a pause  button. The record/playback status is shown at the top of this dialog as well as on the main status bar.

### **Connect Dialog**



**Figure 18 USB Connection Dialog**

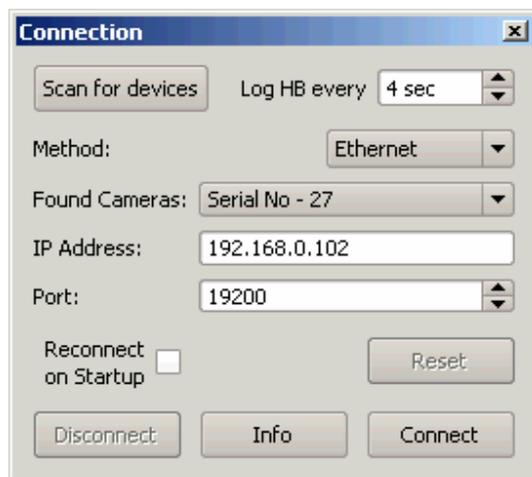
The Connect dialog (*camapp\_connect.dll*) is used to specify the camera connection mechanism (USB, Ethernet, etc) and the specific camera you wish to connect to. The

Connect dialog is accessed using the “Connect” toolbar item . The CCDSP Viewer application polls for both available USB and Ethernet devices [using the `CLLocateHSUSBCameras()` and `CLLocateEnetCameras()` DLL calls ]. Pressing the **Disconnect** button closes the camera connection using the `CLCloseCamera()` DLL call. To reconnect to the same camera every time the application is launch, check the **Reconnect on Startup** check box. If the **Log HB every** control is set (not OFF), the application will log the heartbeat message from the camera to a log file based on the camera model and serial number. The files are stored in the *hblogs* directory under the installation directory. This data can be useful for watching trends with camera performance vs temperature, etc. The file can be loaded back in from the Camera Connection plot chooser dialog (if not connected to a camera). The file is stored in csv format to make it easy to import into other programs.

## USB Connection

The USB connection dialog is show in **Error! Reference source not found.** If the camera is connected to the USB port but not communicating (i.e. powered off), it will be shown as a generic CCDSP device like this . Devices that are functioning will have their model and serial number shown in the **USB Device** combo box. Select the device you want to connect to and press the **Connect** button to initiate a connection to the camera. This connection type utilizes the `CLOpenHSUSBCamera()` DLL call.

## Ethernet Connection



**Figure 19 Ethernet Connection**

The Ethernet connection dialog is shown in . Cameras that have been located on the network will be shown in the **Found Cameras** combo box.

Because the camera discovery protocol uses UDP broadcast technology, cameras not on the same local network (subnet) will

not be listed. You can still connect to a camera by manually entering the IP address of the desired camera.

Selecting a camera from the Found Cameras combo box will fill out the IP address and Port information on the dialog. The default port is 19200 and should not normally be changed. Select the device you want to connect to and press the **Connect** button to initiate a connection to the camera. This connection type utilizes the `CLOpenENetCamera()` DLL call.

### Serial Number Connection



Figure 20 Serial Number Connection

To connect to a camera based on its serial number, select “Serial Number” from the Method combo box. This will show a text field allowing you to enter the serial number of the camera you wish to connect to. When you click the Connect button, the application will use the `CLOpenCameraBySerialNumber()` DLL call to try and establish a connection to the camera. This call tries all available high-speed USB cameras [using the `CLLocateHSUSBCameras()` DLL call ] and then polls for Ethernet based cameras [using the `CLLocateEnetCameras()` DLL call ] to try and find a camera whose serial number matches the specified value. If the correct camera is found the corresponding DLL call [`CLOpenHSUSBCamera()` or `CLOpenENetCamera()` ] is used to open it.

### Snapshot Control Dialog

The Snapshot Control  dialog (`camapp_snapshotcontrol.dll` ) allows you to configure the image acquisition parameters of the camera. There are many sections of this dialog, and they all interact to control the image, so pay attention! Figure 22 shows the Snapshot control dialog configured for an area readout from the camera using the `CLReadCCDArea()` call. The Capture Type has been set to **Area Readout**.

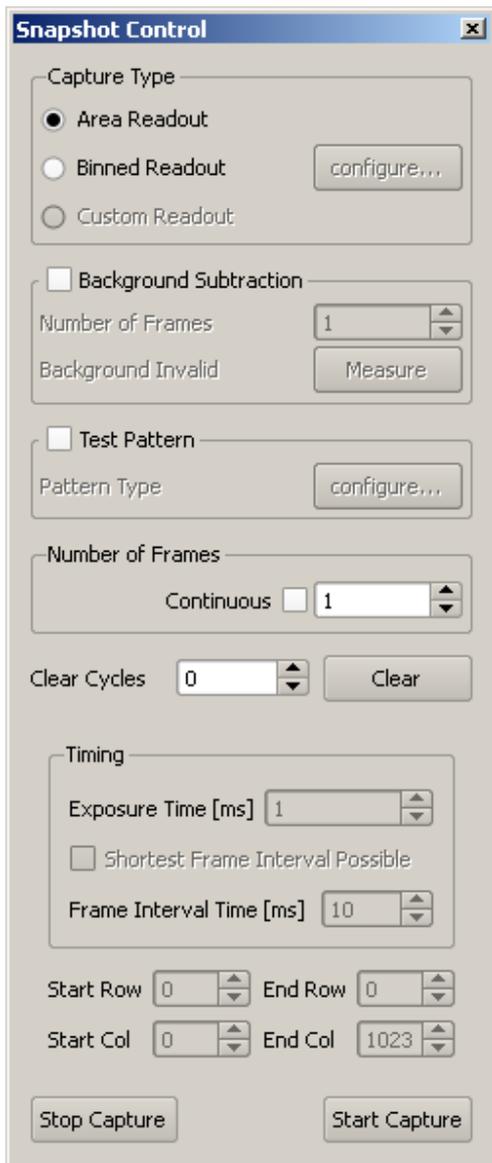


Figure 21 Snapshot Control Dialog (Area Readout)

## Background Subtraction

To enable background subtraction processing, check the box  Background Subtraction for this group. You can then enter the number of frames to use for this processing and click **Measure** to build the background image. This will first clear the CCD by calling `CLClearCCD()`, and then call either `CLReadCCDArea()` or `CLReadCCDBinned()` depending on the type of image selected to start acquiring frames for the background estimate. The frames collected during the measurement phase are averaged to form the background estimate. This image is then subtracted from acquired images while this feature is enabled. The application checks for the image type (area, bin, etc) and geometry to only operate on like images. It does not check the exposure timing. If you change the exposure parameters, you should re-run the measurement process.

## Test Pattern

To substitute a test pattern for the acquired image [using `CLSetTestPattern()` ] check this box  Test Pattern . The Configure button will become enabled and will allow you to change the type and parameters of the test pattern.

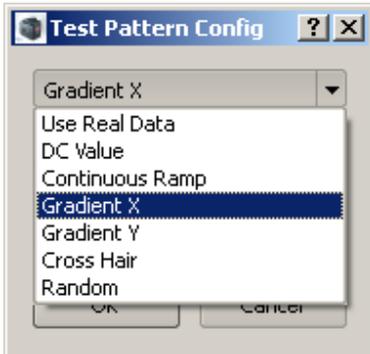


Figure 22 Test Patterns

The different types of test pattern available are shown in the figure to the left. Each type of test pattern is described below.

Table 8 Test Pattern Types

Type from <code>clcamiface.h</code>	Description
<code>PATTERN_TYPE_NONE</code>	Use real CCD data
<code>PATTERN_TYPE_DC</code>	Each read pixel from camera is replaced with DC value in <code>mnSimParams[0]</code>
<code>PATTERN_TYPE_RAMP</code>	Pixels are incremented by one starting at zero
<code>PATTERN_TYPE_GRADIENT_X</code>	Pixels are scaled from <code>PatternParams[0]</code> to <code>PatternParams[1]</code> by <code>PatternParams[2]</code> in X dimension
<code>PATTERN_TYPE_GRADIENT_Y</code>	Pixels are scaled from <code>PatternParams[0]</code> to <code>PatternParams[1]</code> by <code>PatternParams[2]</code> in Y dimension
<code>PATTERN_TYPE_CROSSHAIR</code>	Two values, cross hairs DC value in <code>PatternParams[0]</code> , value 2 in <code>PatternParams[1]</code> , every [2] in X, [3] in Y
<code>PATTERN_TYPE_RANDOM</code>	Pixels mean is <code>PatternParams[0]</code> with std of $0.1 * \text{PatternParams[1]}$

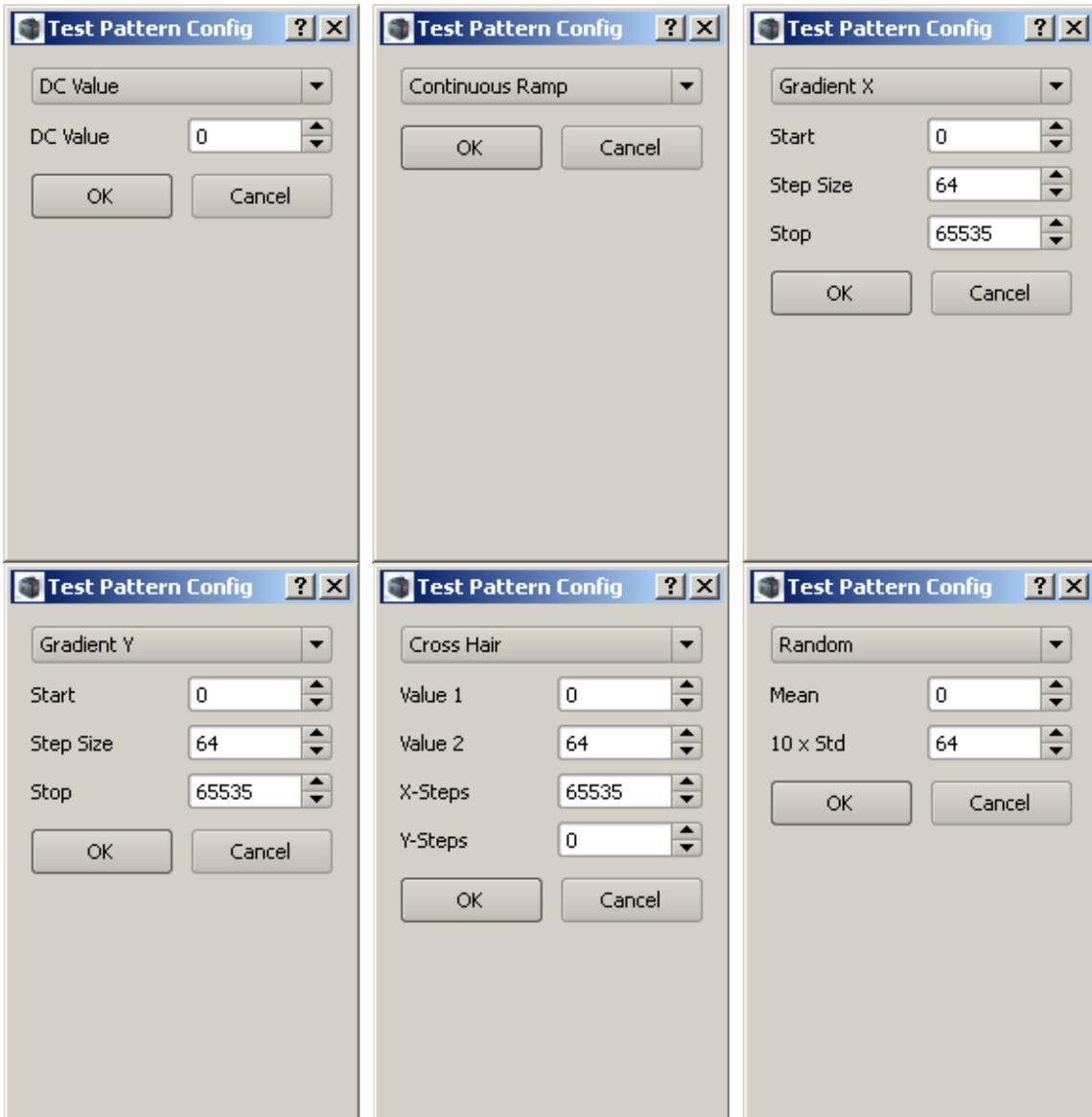


Figure 23 Test Pattern Configuration

## Number of Frames

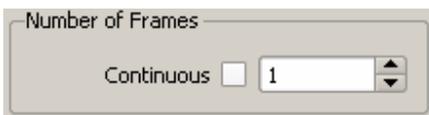


Figure 24 Number of Frames

To adjust the number of frames to acquire, change the value in the spin box (which sets the NumImages parameter to the CLReadCCDArea() DLL API call). Selecting the **Continuous** check box will, as its name implies, cause the camera to acquire frames continuously until you press the **Stop Capture** button (the NumImages parameter to the CLReadCCDArea() DLL API call is set to 0). Normally the camera will stop acquiring images after the set number of frames.

## Capture Controls

To clear the CCD before acquiring an image, use the **Clear Cycles** spin box to set a value > 0. This will set the `NumClears` parameter in the `CLReadCCDArea()` and/or `CLReadCCDBinned()` calls. Pressing the **Clear** button will call `CLClearCCD()` immediately and has no affect on subsequent captures.

## Timing Control

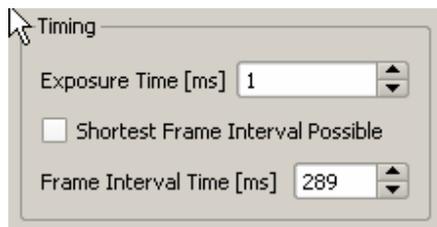


Figure 25 Timing Controls

The Timing controls allow you to adjust the timing parameters of the capture. Either area read or binned read acquisition modes can be configured with a precise exposure time between successive image reads or image clear/read cycles. The **Exposure Time [ms]** control sets the exposure time using the `CLSetExposure()` API command. The **Frame Interval Time [ms]** control uses the

`CLSetImageInterval()` API call to set the frame interval. Selecting the **Shortest Fame Interval Possible** control uses the `CLGetImageIntervals()` API call after setting an image interval of 1 ms [using the `CLSetImageInterval()` call]. The camera will compute the minimum amount of time required to acquire the frame and return it.

The CCDsp uses a command pipeline to process requests from the controlling application. Upon receiving a `CLReadCCDArea()` or `CLReadCCDBinned()` command, the CCDsp will wait the configured exposure time and then begin the image acquisition process.

In order to achieve precise timing between an image clear and an image read, the application should send a `CLClearCCD()` and then a `CLReadCCDArea()` or `CLReadCCDBinned()` command back-to-back so that, upon completing a clear command, the read command is immediately executed.

Note: if the application does not intend to specifically clear the CCD but rather issue back-to-back image reads the same technique can be used. However, the total exposure time between clocking out the CCD data will also include the transfer time of the image.

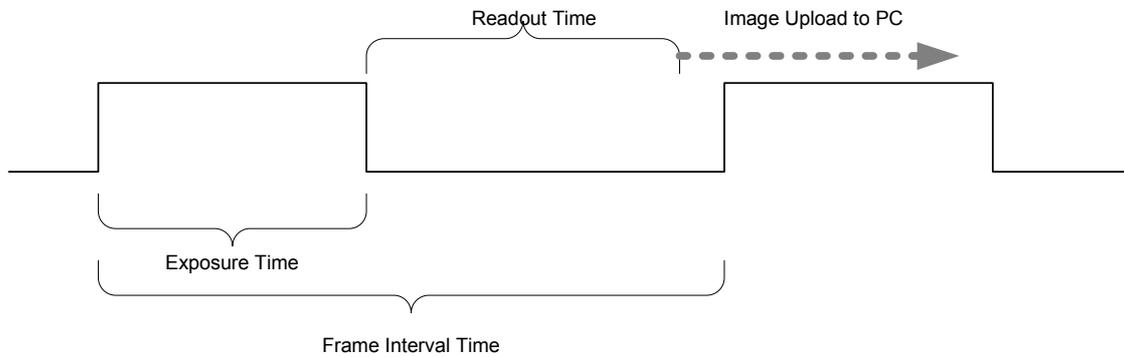


Figure 26 Image Acquisition Timeline

### Area Control

If the **Capture Type** is set to **Area Readout**, the geometry of the acquired image can be controlled using the start/end controls. The values set with these controls are passed as arguments to the `CLSetCCDArea()` API call.



Figure 27 Area Controls

### Binned Image Summary

If the **Capture Type** is set to **Binned Readout**, the application will show a summary image, showing the binned areas (see Figure 28). This image does not update as images are acquired, it is meant to be a visual indicator of the sensor areas that will be acquired with the current binning.



Figure 28 Binning Summary Image

### Binning Configuration

Clicking the **configure...** button next to the **Binned Readout** checkbox brings up the Binning Setup dialog. This dialog can be quite complex, but defaults to a simpler setup that should satisfy most acquisition needs. When the dialog is opened, it retrieves the binning information from the camera using the `CLGetBinParam()`, `CLGetHorBinParam()`, and `CLGetBinningSequence()` API calls.

The CCDsp contains a CCD charge readout register (prior to analog to digital conversion) and a digital row register (after analog to digital conversion). The CCDsp binning mode allows binning of CCD rows in the CCD charge readout register, the digital row register, or both.

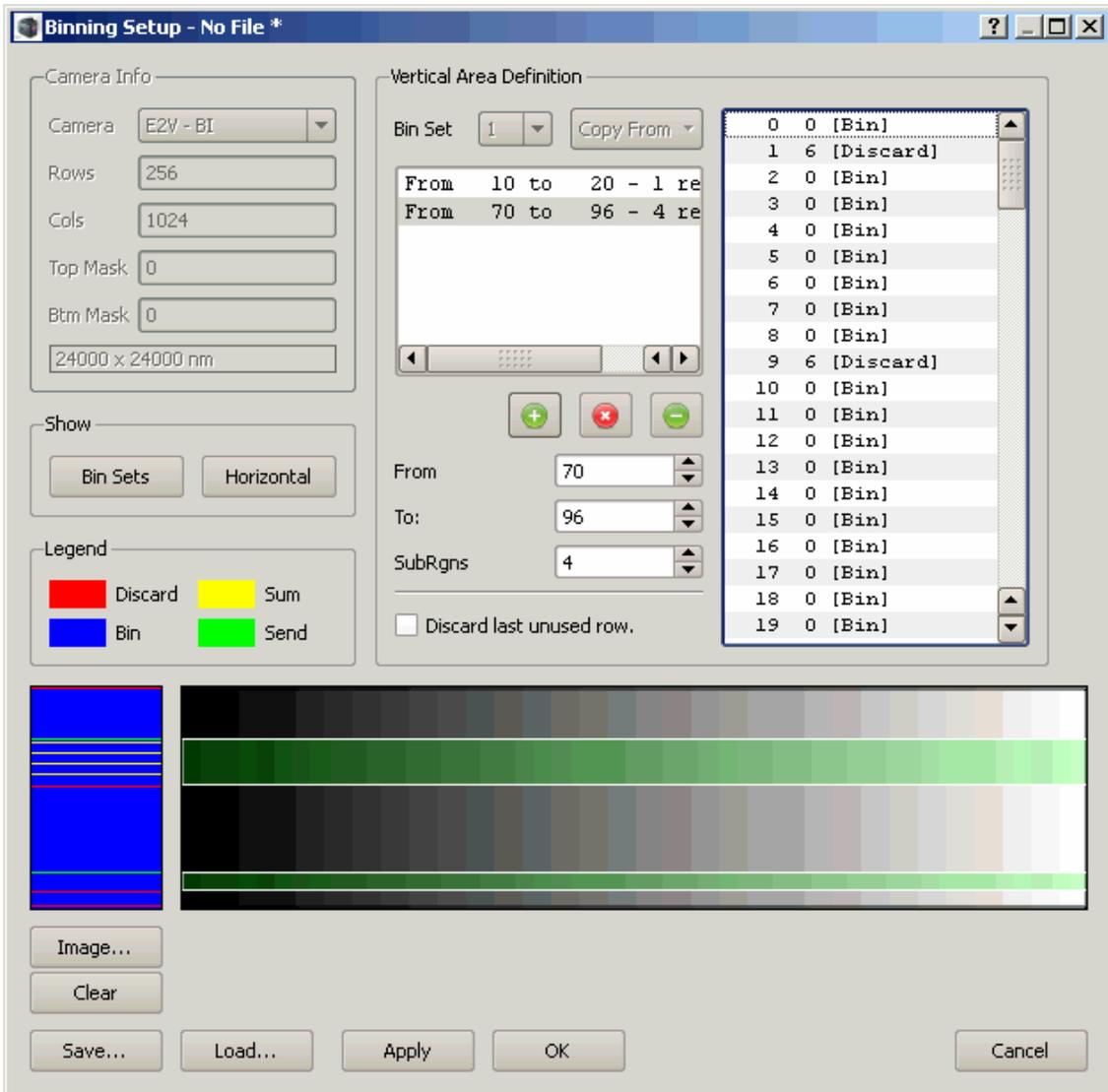


Figure 29 Vertical Binning Setup Dialog

The API call requires configuration of what the CCDsp should do with the data stored in each of the active rows in the CCD array. One of four possible operations may be selected for each row:

1. **Bin In CCD** – the current row charge should be accumulated with the charge already present in the CCD readout register. This operation results in CCD binning. Note that the readout register is always cleared prior to reading the first row of the CCD on an acquisition.
2. **Discard** – the current row charge is accumulated into the CCD readout register. The CCD readout register is then cleared without sampling/storing the data. This mode is primarily used for removing unwanted portions of the CCD from the transfer process.

3. **Send** – The current row charge is accumulated into the CCD readout register. The readout register contents are read (and cleared), analog to digital converted, summed with any contents in the CCDsp row contents and finally transferred to the application. The CCDsp row contents are then subsequently cleared.
4. **Sum** – The current row charge is accumulated into the CCD readout register. The readout register contents are read (and cleared), analog to digital converted, and summed in the CCDsp processor row contents. This mode provides the capability to bin data using the CCDsp processor hardware. This allows growth in dynamic range as the CCDsp processor accumulates the 16 bit ADC data into a 32 bit result code.

Horizontal binning can also be utilized, although there is no “**Bin**” capability. The horizontal binning is all performed by the DSP in the digital domain (*after* any defined vertical binning is performed).

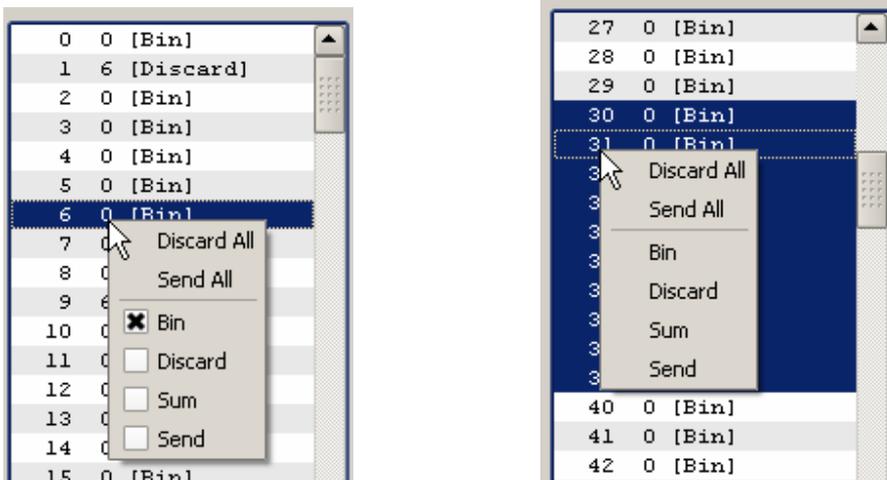
The simplest mode of binning configuration is to set up a single set of vertical binning parameters. Figure 29 shows this dialog. The camera information is shown in the top-left corner of the dialog for reference. The camera defaults to sending all rows.

### Vertical Area Definition

The Vertical Area Definition group provides a number of controls to configure vertical binning. In the simple view, the **Bin Set** controls are all disabled, and the camera is instructed to use the first bin set. To start a new vertical bin definition, clear the set using

the  button. You can then enter the start (**From**) and end (**To**) rows to bin. If you want to sum groups of rows within the set (to avoid overflow/saturation), use the **subRgns** control to add summing operations to the set. The application divides the binning region evenly into however many regions you define, binning and then sending.

The  button adds the group to the set. The  button deletes the selected group from the binning set. To edit a group, select it in the table and adjust the parameters. The complete bin set is shown in the list to the right of this group. You can change individual entries in the list using the popup menu. You can also select a number of rows in the set and change them.

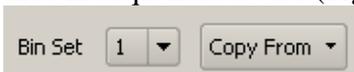


**Figure 30 Bin setup list menu**

The application will automatically discard the top and bottom masked rows (as defined by the camera info). If you select the **Discard last unused row** checkbox, the software will mark the last row of the sensor as discard to clear charge accumulation before the next readout.

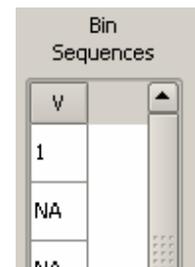
### Bin Sets

The CCDsp maintains a list of up to 16 bin sets to cycle through various combinations of the vertical and horizontal binning patterns for a multi-image acquisition (e.g., calls to `CLReadCCDBinned()` with `NumImages > 1`). The default configuration is to use only bin set 1 (vertical and horizontal). If you need to use a varied sequence of bin sets (either vertical or horizontal), toggle the **Bin Sets** control in the **Show** group. This will enable the **Bin Set** controls in the **Area Definition** group(s) and show a list enabling you to edit the bin sequences to use (Figure 31).



**Figure 31 Bin Set controls**

The Bin Set controls let you pick which of the 8 bin sets you are editing. The Copy From button allows you to quickly copy from one of the other bin sets. The **Bin Sequences** list allows you to define the order in which the bin sets are executed. Right click on the row to change and select the bin set to use from the popup menu.



### Horizontal Area Definition

Toggling the **Horizontal** control in the **Show** group shows the **Horizontal Area Definition** group. This group behaves much the same as the vertical group, except:

- There is no Bin option (as binning is an analog function and horizontal binning is done in the digital domain).
- There are no sub regions

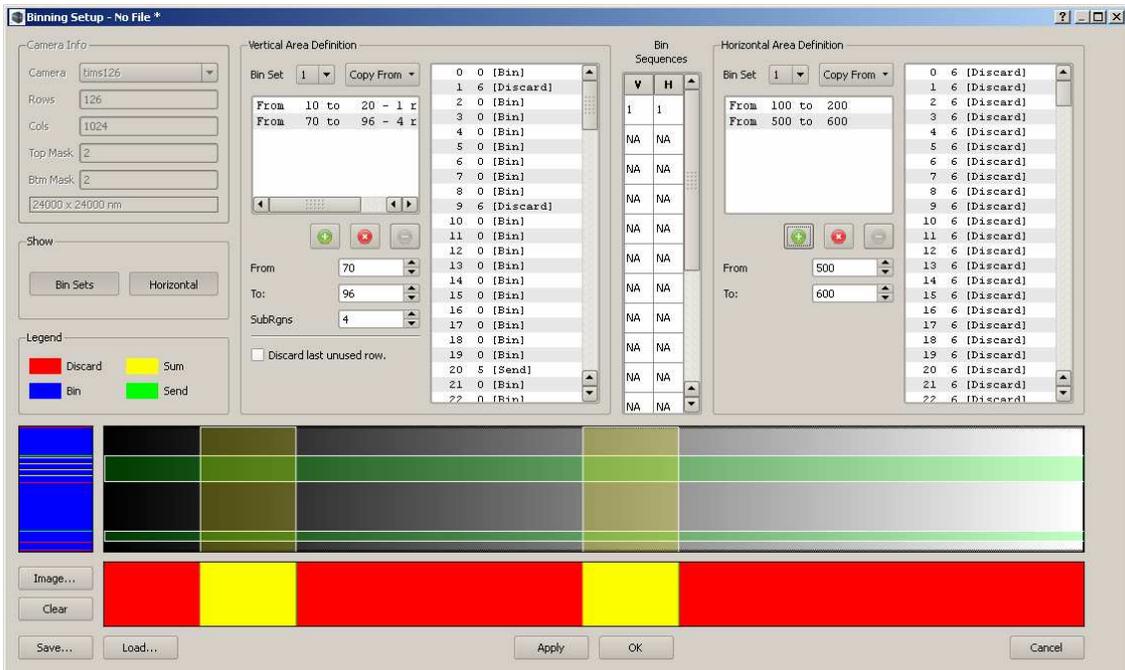


Figure 32 Full Binning Setup Dialog

### Bin Setup Display and Miscellaneous Functions

Below the area definition groups is the binning summary display. This display is intended to show a graphical representation of the binning configuration. On the left is a small display of the vertical binning. The vertical binning list is used to highlight rows in the display according to the legend. The horizontal binning is also shown below the summary display. The summary display overlays the bin sets (H and V) over a sample image to help show the sensor areas being processed. If an area image has been acquired before opening this dialog, it will be shown in the summary display. You can also use the **Image...** button to load an image from disk. Using this button you can load a standard image (.png, .jpg, etc) or an image from a recorded set (.cdi). The **Clear** button clears the summary background image.

The **Save...** and **Load...** buttons allow you to store and retrieve bin set configurations from a file on disk.

The **Apply** button sends the binning configuration to the camera. If the horizontal binning is not displayed, a default (send every column) bin set is sent to the camera (using `CLSetHorBinParam()`). Similarly, if bin sets are not used, just the first bin set is sent to the camera (using the `CLSetBinParam()`). The **OK** button performs an Apply and then closes the dialog. The **Cancel** button closes the dialog without sending anything to the camera.

Note that the camera does not store binning configuration data in its NVRAM, and will default to one vertical and one horizontal bin set sending every pixel.

## Image Statistics Dialog

The Image Noise Statistics dialog is shown using its icon  on the toolbar. When the **Enable** box is checked, this dialog will compute image statistics on every acquired image. When acquiring images at high frame rates, you may need to disable this dialog to achieve the desired throughput (depending on your PC's CPU). The dialog computes the image mean pixel value, the noise (std deviation) value, and shows how many pixels fall outside of 1,2,3, and 4 sigma from the noise value as a percentage. The nominal values for these percentages (based on a Gaussian distribution) are shown to the right of the values). You can reduce the size of the considered area by adjusting the **X Margin** and **Y Margin** controls. These values are applied to both edges of the X or Y limits.

## Plot Statistics

The Image Noise Statistics dialog can compute image statistics suitable for plotting using the plot display window. Clicking the **Generate** button will compute the following data sets:

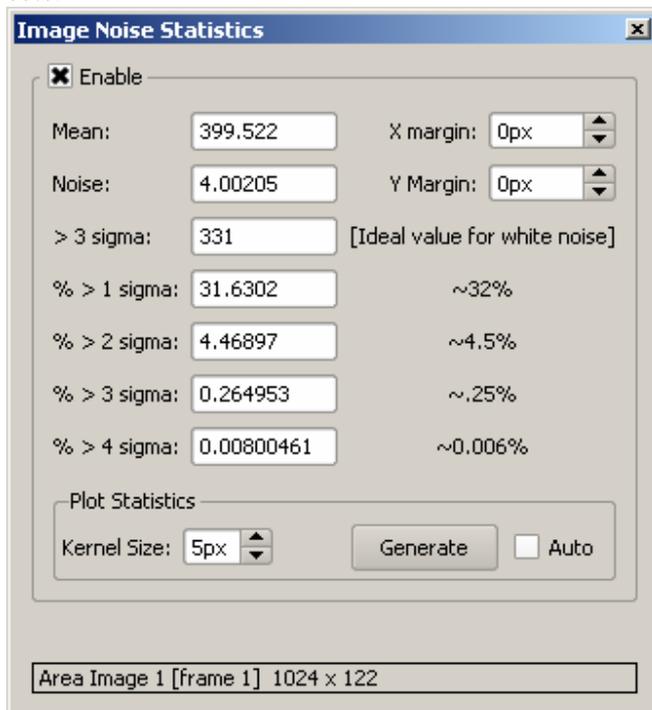


Figure 33 Image Noise Statistics Dialog

- StdDev by Row      StdDev for each pixel in a row
- StdDev by Column      StdDev for each pixel in a column
- Mean by Row      Mean value across a row
- Mean by Column      Mean value along a column
- Data by Row      Image data in a given row
- Data by Column      Image data in a given column
- FFT by Row      FFT [1024 bin] of the data in a row
- Histogram      Image histogram.

Checking the **Auto** box will cause the dialog to compute image plot statistics for every image acquired. You may find this is too much for your CPU to handle.

### Peak Finder Dialog

The Peak Finder dialog can be used to examine a row in an image to find the peak values.

Click the  icon in the toolbar to open this dialog.

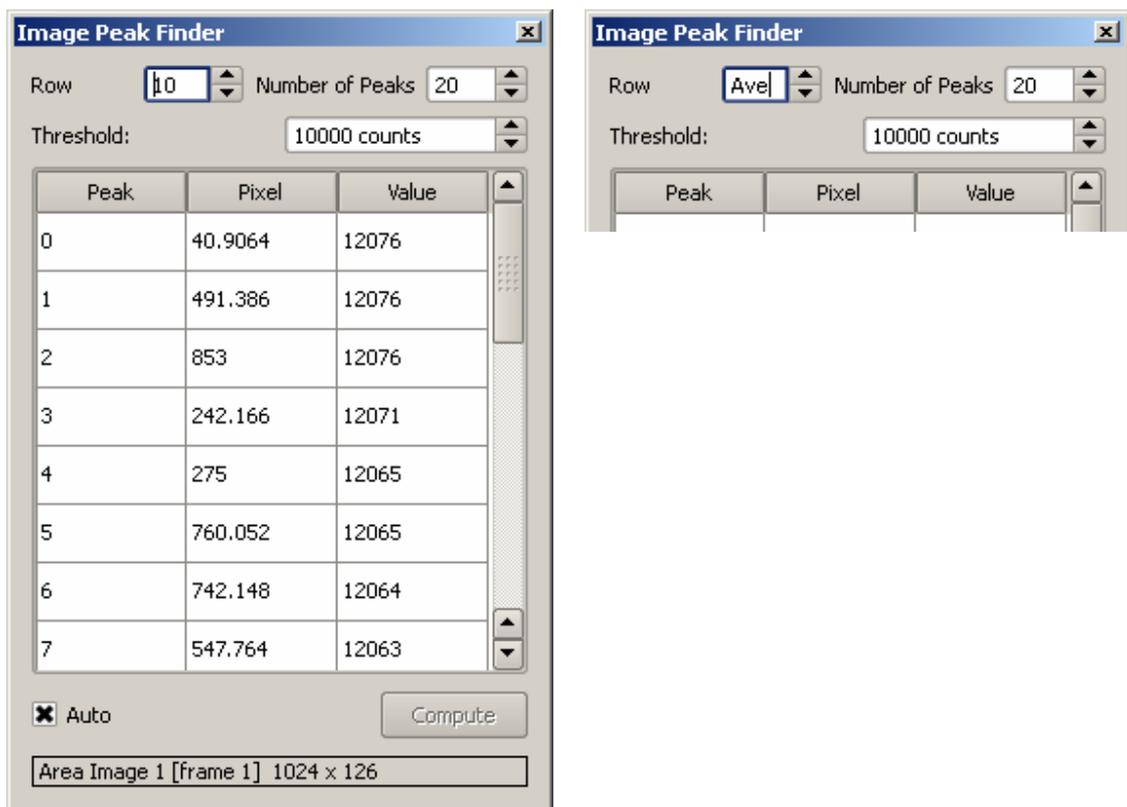


Figure 34 Peak Finder Dialog

Use the **Row** control to select which row to find peaks on (or select “Ave” to average all the rows in the image [its one down from “0”]). The Number of Peaks control limits the number of peaks extracted. Only pixels whose value exceeds the threshold are considered in the peak processing. The Pixel value shown is interpolated using a centroid algorithm and thus is a floating point pixel equivalent. The Value shown is the pixel value of the peak from the image (not interpolated).

If the **Auto** button is checked the peaks are located for every image acquired, otherwise click the **Compute** button to populate the table.

### Link Test Dialog

The link test dialog is used to verify link integrity and measure I/O bandwidth. This dialog is for Critical Link internal use.

## Calibration Dialog



Figure 35 Calibration Dialog

The Calibration dialog is shown using its icon  on the toolbar. The Calibration dialog is used to set up base parameters for the camera. The **Shutter Timing** group is used to set the shutter parameters. The camera will open the shutter (TTL pulse '1' when *ActiveHigh* is non-zero, '0' otherwise). The camera will delay for *OpenMs* milliseconds, expose for the exposure time from *CLSetExposure()*. The camera will close the shutter (TTL pulse '0' when *ActiveHigh* is non-zero, '1' otherwise), delay *CloseMs* milliseconds, then the CCD pixel data will be read/shifted from the CCD array.

The **Set Points** group allows you to set the DAC target offset and desired CCD temperature using the *CLSetCCDOffset()* and *CLSetCCDTargetTemp()* API calls respectively. The state of the **Zero Mean Binned Data** box is passed as the *ApplyZeroMean* parameter to the *CLSetCCDOffset()* call. The CCD Temp Setpoint field is populated from the camera using the *CLGetCCDTargetTemp()* call.

The **Shearing** group allows you to enter the parameters to the *CLSetShearingParams()* API call. The **X Param** and **Y Param** values must be odd. If a pixel exceeds the local mean by the **Threshold** value, it is replaced by the local mean.

The Mirroring group allows you to set the horizontal and vertical mirroring (using the *CLSetCCDImageMirroring()* API call). See the API documentation for specifics regarding how mirroring is accomplished in the camera.

The GPIO group allows you to configure the 4 GPIO pins available on the CCDsp camera using the *CLSetGPIO()* API call. Each pin can be configured as an input or output. If a pin is configured as an input, the **Trigger** checkbox is used to set a specific pin as the trigger via the **Figure 36 GPIO Pin configuration** *CLSetTrigger()* API call. The different states you can assign to the GPIO pins are listed in Table 9.

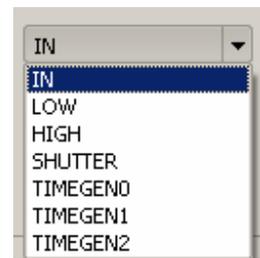


Table 9 GPIO States

State	Description
eeINPUT	set the GPIO as in INPUT

State	Description
eeOUTLOW	drive GPIO low
eeOUTHIGH	drive GPIO high
eeSHUTTER	drive GPIO with shutter signal
eeOUTTIMGEN0	drive GPIO with custom timing strobe 0
eeOUTTIMGEN1	drive GPIO with custom timing strobe 1
eeOUTTIMGEN2	drive GPIO with custom timing strobe 2

### Factory Configuration Dialog

The Factory Configuration dialog is used to do the basic setup of the camera. It is used to set up the basic model and serial number information, timing file, substrate configuration, and network settings. The vast majority of this data should not be changed by anyone outside of the Critical Link engineering and/or production departments. There are 4 views on this dialog, selectable from the radio buttons at the top.

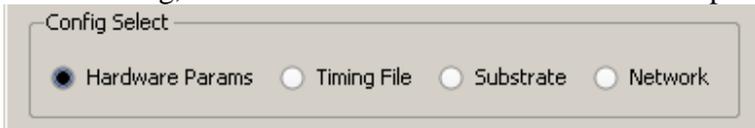
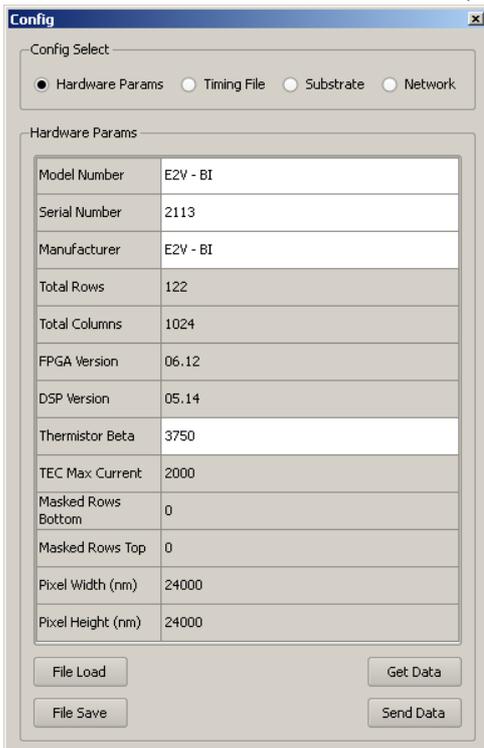


Figure 37 Factory Config Selection

### Hardware Params

The Hardware Params view shows (and lets you edit some of) the parameters shown in Table 10. You can save these current parameters off to a file using the File Save button and subsequently load them back from the file with the File Load button. The parameters are retrieved from the camera automatically, but you can retrieve them manually using the Get Data button. This will make a call to the `CLGetCameraHardwareInfo()` API call. To update the parameters on the camera, press the Send Data button (this uses internal engineering commands to send the configuration to the camera). End users should normally have to use this feature.



**Figure 38 Hardware Parameters Dialog**

**Table 10 Hardware Parameters**

<b>Parameter</b>	<b>R/W</b>	<b>Description</b>
Model Number	<b>Y</b>	Camera model number (as determined by MFG).
Serial Number	<b>Y</b>	Camera serial number.
Manufacturer	<b>Y</b>	Camera Manufacturer.
Total Rows	<b>N</b>	Total number of rows in the sensor.
Total Columns	<b>N</b>	Total number of columns in the sensor.
FPGA Version	<b>N</b>	FPGA version number.
DSP Version	<b>N</b>	DSP software version number.
Thermistor Beta	<b>Y</b>	CCD temperature thermistor beta factor.
TEC Max Current	<b>N</b>	TE cooler current limit.
Masked Rows Bottom	<b>N</b>	Number of rows masked (and discarded) at bottom of sensor.
Masked Rows Top	<b>N</b>	Number of rows masked (and discarded) at top of sensor.
Pixel Width (nm)	<b>N</b>	Pixel width in nanometers.
Pixel Height (nm)	<b>N</b>	Pixel height in nanometers.

### Timing File

The Timing File view shows the contents of the timing file for the camera. This file controls the low-level timing of reading the sensor, as well as defining the basic attributes of the sensor. Timing files are generated by Critical Link engineering and are not meant to be modified by end users.

### Substrate

The Substrate configuration view is used to configure the substrate DAC setting. You can enter the value manually, or have the application run a calibration loop acquiring dark current images, stepping the DAC setting until **the Max Dark Current** value is met. This loop uses the `CLSetExposure()` and `CLReadCCDArea()` API calls. The actual setting of the DAC value is done via internal engineering functions.

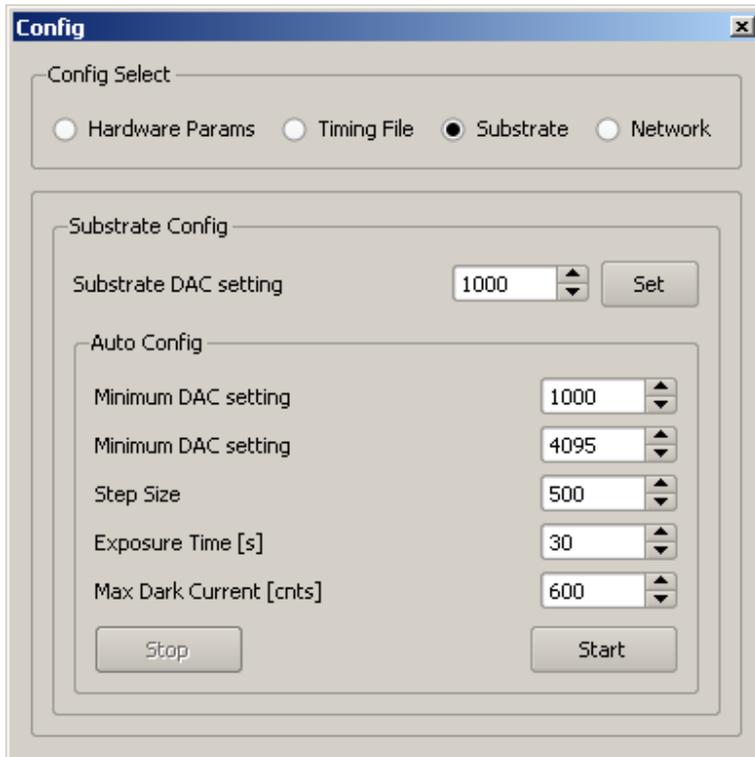
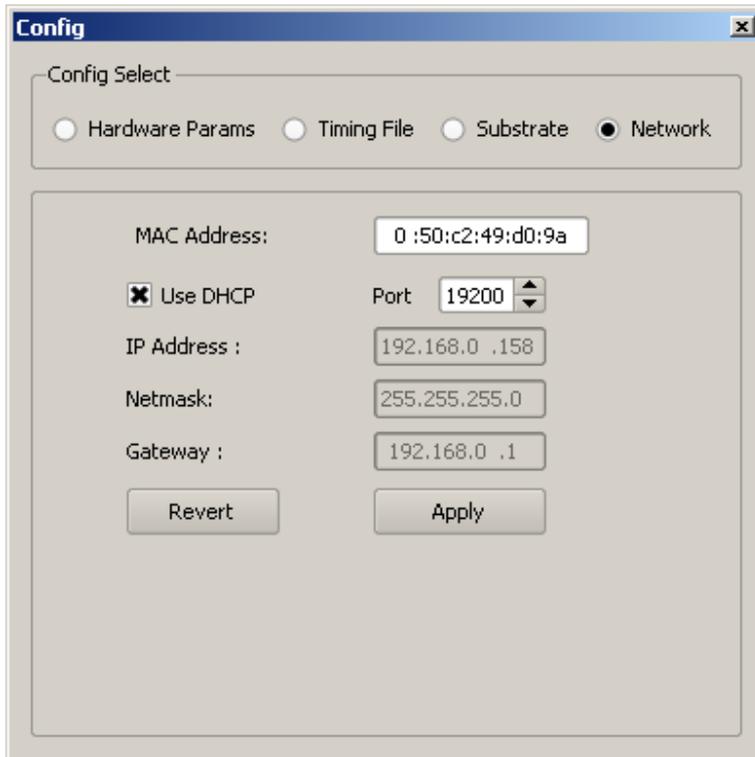


Figure 39 Substrate Configuration Dialog

## Network

The **Network** view allows you to reconfigure the CCDsp camera's network configuration. You can select the Use DHCP box to enable DHCP network settings, or enter the configuration manually. The Apply button uses the `CLSetCameraNetworkConfig()` to send the settings to the camera. The **Revert** button will revert the dialog settings to those currently in the camera.



### ***I/O Board Test Dialog***

The I/O Board Test Dialog is used to check out the camera I/O board. This dialog is for internal Critical Link Engineering use only.

## **How To**

## **Developer Reference**

## Camera DLL API Call References

CLClearCCD( ) .....	19, 23	CLSetCCDOffset( ) .....	26
CLCloseCamera() .....	16	CLSetCCDTargetTemp( ) .....	26
CLEnableFan( ) .....	7	CLSetCooling( ) .....	7
CLGetCameraHardwareInfo() .....	27	CLSetErrorCallback.....	6
CLGetCCDTargetTemp( ) .....	26	CLSetExposure().....	23, 26, 28
CLGetImageIntervals( ) .....	23	CLSetGPIO( ) .....	26
CLGetReadProgress() .....	7	CLSetHeartBeatCallback.....	6
CLLocateEnetCameras().....	16, 18	CLSetImageDataCallback.....	6
CLLocateHSUSBCameras().....	16, 18	CLSetImageInterval( ) .....	23
CLOpenCameraBySerialNumber().....	18	CLSetShearingParams().....	26
CLOpenENetCamera() .....	17, 18	CLSetShutterParams() .....	26
CLOpenHSUSBCamera().....	17, 18	CLSetTestPattern( ) .....	21
CLReadCCDArea( ).	18, 19, 22, 23, 28	CLSetTrigger( ) .....	26
CLReadCCDBinned( ) .....	19, 23		
CLSetCCDArea( ) .....	24		