ISL-3200™ Digital Imaging Sensor Interface and Test Solution

Table of Contents

1. INTRODUCTION

ISL-3200™ is a cost-effective combination of sensor interface, test electronics and application software that provides complete communications, image capture, and characterization testing, of a wide variety of image sensors.

The ISL software application offers scripting and plug-in functionality, allowing enhanced graphical user interfaces to specific image sensor models as well as customized image processing analysis and characterization testing routines.

The ISL software application is pre-configured with a library of testing and characterization routines, as well as many of the processing tools that are typically needed for thorough evaluation and testing of image sensors.

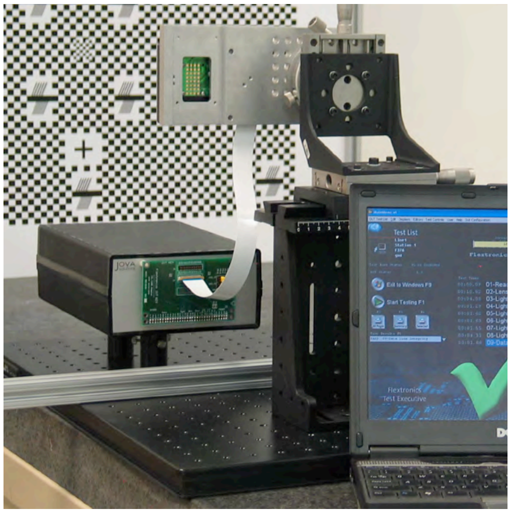
2. CONTACT INFORMATION

2.1 SALES AND SUPPORT

Jova Solutions
965 Mission Street, Suite 600
San Francisco, CA 94103

415-348-1400 Office
415-348-1414 Fax
415-348-1408 Technical Support
800-755-1400 Toll Free Sales

http://www.imagesensorlab.com
info@imagesensorlab.com



3. REFERENCE DOCUMENTS

Description	*Doc. No*	*Company/Author*	*Rev. Date*
ISL-3200 Product Specification	210-0004-08	Jova Solutions	07/14/2009
ISL-3200 Basic User Manual	210-0001-01	Jova Solutions	07/14/2009
ISL-3200 Advanced Analysis Guide	210-0002-01	Jova Solutions	07/14/2009
ISL-3200 Quick Start Guide	210-0008-02	Jova Solutions	07/14/2009
ISL-3200 Test and Automation Suite Guide	210-0003-01	Jova Solutions	07/14/2009

4. GENERAL CONCEPTS

ISL-3200 consists of application software running on a host computer, which communicates with the ISL hardware. Users connect image sensors to the I/O connector on the ISL hardware and use the application software to control the ISL hardware to communicate with and acquire images from the image sensor. The ISL hardware contains all the system components needed to interface many different CMOS image sensors.

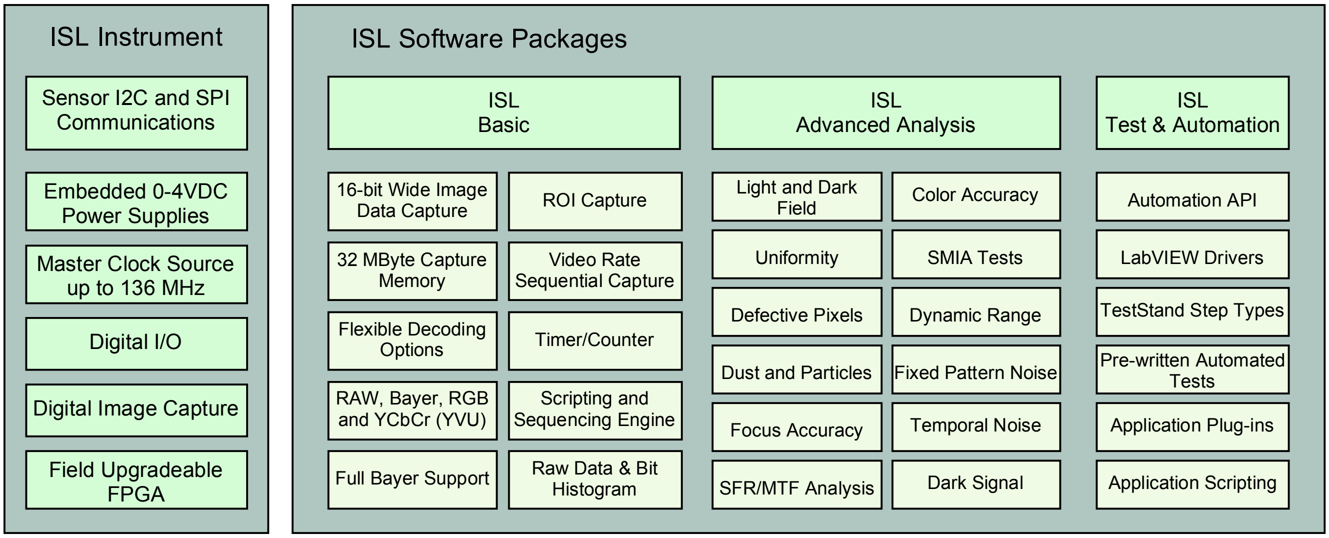
The ISL application software provides flexible image capture, display and analysis capabilities and also provides a set of productivity tools that ease the process of communicating with and testing image sensors. The application software provides full Bayer image support and includes a variety of built-in Bayer decoding and interpolation options. Image Sensor Lab uses four basic image buffer types for Raw, Bayer, RGB, and YCbCr (YUV) image types.

The Image Sensor Lab application software also provides several image analysis and characterization test panels including uniformity, defective pixels, Bayer analysis, dust and particles, focus accuracy, SFR/MTF analysis, and Macbeth color accuracy tests.

The ISL application software also provides a sequencing engine, including application scripting with which users can automate tedious startup configuration or mode changing steps. The ISL hardware, I2C sensor communications, and the ISL application software can all be controlled by simple text sequence files.

A plug-in architecture is included that allows users to write their own display and analysis panels, using the LabVIEW development package from National Instrument (http://www.ni.com).

Timer/Counter measurements are also available on all image sensor connections and can be run in parallel with image sensor communications and image capture.



ISL Instrument

- Sensor I2C and SPI Communications
- Embedded 0-4VDC Power Supplies
- Master Clock Source up to 136 MHz
- Digital I/O
- Digital Image Capture
- Field Upgradeable FPGA

ISL Software Packages

ISL Basic

- 16-bit Wide Image Data Capture
- ROI Capture
- 32 MByte Capture Memory
- Video Rate Sequential Capture
- Flexible Decoding Options
- Timer/Counter
- RAW, Bayer, RGB and YCbCr (YVU)
- Scripting and Sequencing Engine
- Full Bayer Support
- Raw Data & Bit Histogram

ISL Advanced Analysis

- Light and Dark Field
- Color Accuracy
- Uniformity
- SMIA Tests
- Defective Pixels
- Dynamic Range
- Dust and Particles
- Fixed Pattern Noise
- Focus Accuracy
- Temporal Noise
- SFR/MTF Analysis
- Dark Signal

ISL Test & Automation

- Automation API
- LabVIEW Drivers
- TestStand Step Types
- Pre-written Automated Tests
- Application Plug-ins
- Application Scripting

The ISL-3200 application software runs on a host computer and connects, through a single USB 2.0 High Speed cable, to the ISL image sensor interface electronics box. (PCI Express connection is an available option.) The ISL hardware contains all the components needed to interface a variety of CMOS image sensors. Power supplies, I2C communications, digital image capture, and additional I/O are contained within the box and are accessed through the 64-pin I/O connector. (See figure below).

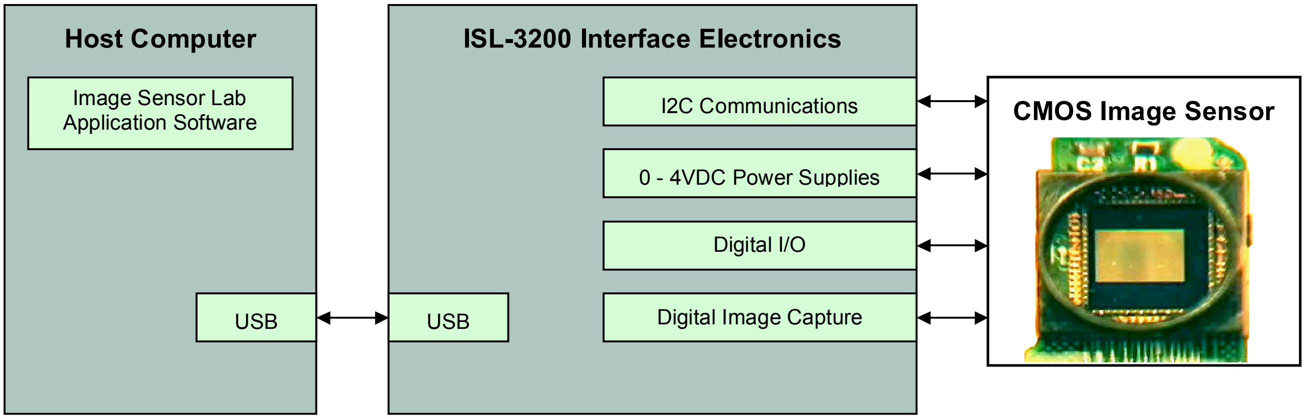
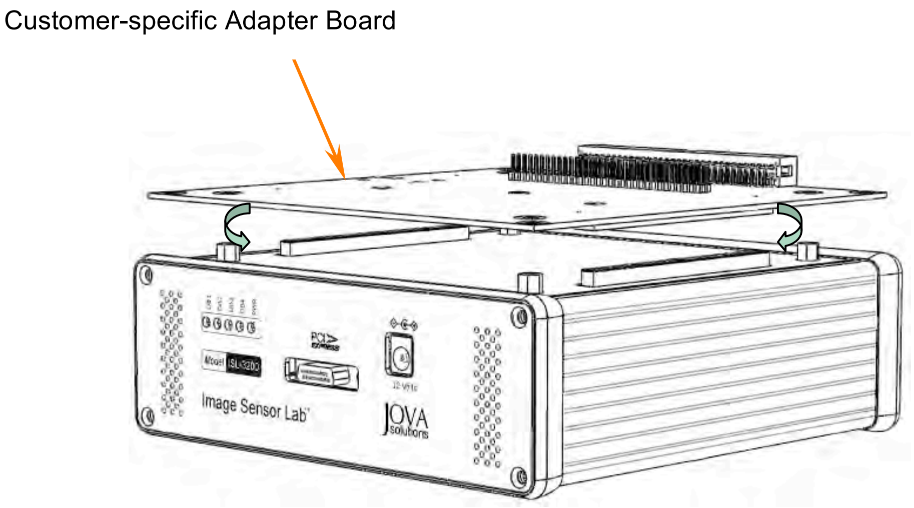
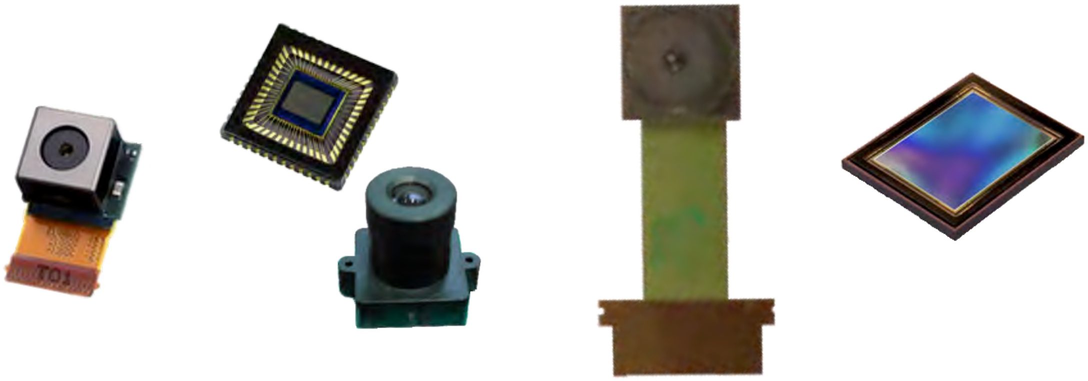


Image sensor adapter boards plug into the two 160-pin connectors on the ISL hardware, as shown in the image below. These adapter boards are specific to the connector type used to mount the image sensor under test and route the ISL unit signals, available through the two 160-pin connectors, to the appropriate pins of the image sensor connector on the adapter board.



Typical CMOS Camera Modules Used with Image Sensor Lab



5. ISL INSTALLATION

Before attempting to plug in and operate the ISL software and ISL hardware, follow the installation instructions below, which ensure proper installation and configuration of the various components that comprise the Image Sensor Lab product. The software must be installed prior to connecting the USB cable so that the needed software components are available when the Windows operating system detects the new hardware.

5.1 SOFTWARE INSTALLATION

The Image Sensor Lab product contains several software components that are automatically placed on your hard disk during the installation procedure.

A. The Image Sensor Lab Application Software (ImagesSensorLab.exe)

B. Low-level USB or PCIe Drivers

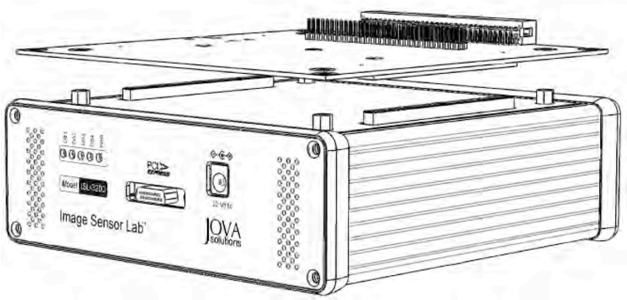
C. National Instruments IMAQ Libraries

5.1.1 Starting the ISL Software Installation

Place the ISL™ software installation CD into the host computer CD drive and run the **Install.exe**.

Follow the prompts to fully install all software components. The installer creates an ISL directory in the C:\Program Files directory and installs low-level USB or PCIe drivers in the Windows directory.

5.2 ISL Hardware Setup



First ensure that an image sensor is connected to the ISL unit. Although this documentation references the demo adapter Micron 1.3 Megapixel image sensor, the instructions are similar, regardless of sensor type. Be sure that the image sensor is properly mounted and connected.

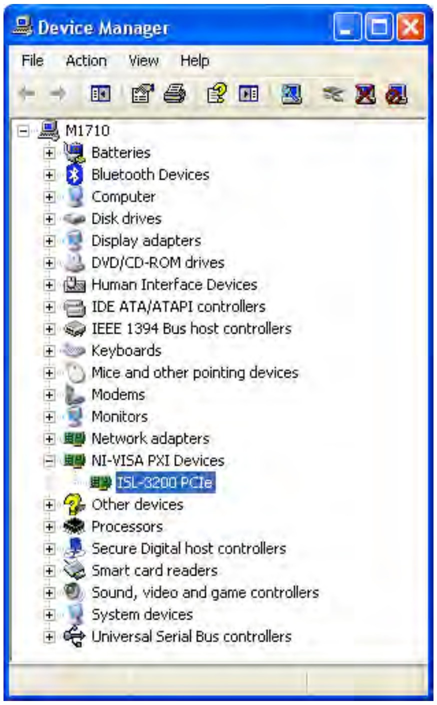
The sensor connection to the ISL-3200 is through the two 160-pin high density connectors on the top of the ISL-3200 unit.

5.3 ISL Host Computer Interface Setup

The ISL hardware contains either a USB 2.0 High Speed interface, or an External PCI Express interface. Refer to the Sections 5.3.1 and 5.3.2 for details specific to each interface.

5.3.1 External PCI Express Interface

5.3.1.1 External PCI Express Driver Installation



Refer to the Image Sensor Lab Getting Started Guide for PCIe driver installation details.

Install the ISL PCIe driver before connecting the ISL hardware to the computer.

The PCIe driver requires the NI-VISA subsystem. The Image Sensor Lab Installer installs the NI-VISA subsystem. Ensure the NI-VISA subsystem is correctly installed on the host computer before installing the ISL PCIe drivers.

The Image Sensor Lab installer also installs the PCIe driver .inf file. This file is located in *C:\Program Files\Image Sensor Lab\Drivers\Jova External PCIe.inf*, and you may be prompted by the operating system for the location of the .inf file.

5.3.1.2 Connecting the ISL hardware to the host computer using External PCIe

Power-down both the host computer and the ISL hardware. Once power is disconnected, insert the PCIe express card into the laptop express slot, or if using a workstation computer install the PCIe adapter card. Connect the ISL hardware to the host computer using the provided External PCIe cable.

Once the cable is securely connected to both the ISL hardware, the host computer adapter power may be applied, first to the ISL hardware and then to the host computer.

⚠ ***NOTE: Once the host computer has booted up, the power to the ISL hardware must not be interrupted. If the power to the ISL hardware is removed, you must shut down the host computer <u>completely</u> before re-applying power to the ISL hardware.***

5.3.2 USB 2.0 High Speed Interface

Attach a USB 2.0 High Speed cable between the ISL HARDWARE and the Host Computer. Be sure to use a root hub on the host computer. DO NOT use an un-powered external USB hub! For optimal performance, choose a root hub with the least number of additional devices connected.

The Windows operating system will detect a new device and prompt the user with a “found new hardware” dialog. Follow the prompts to finish installing the USB device drivers.

6. ISL APPLICATION SOFTWARE

Image Sensor Lab application software is a Windows–based application, which provides a Graphical User Interface (GUI) that you will use to control the ISL Image Sensor Interface electronics and to acquire, display, and process images. Image Sensor Lab is installed in the C:\Program Files\Image Sensor Lab directory by default.

6.1 ISL Configuration and Support Files

Before you run the ISL application software, a short introduction to the various configuration files used by the application is appropriate. There are several configuration files that can be used to quickly change the behavior of the ISL hardware and the ISL application software. These configuration files can be used to rapidly switch from one image sensor type to another, which may have completely different power supply, digital I/O, frame capture or sensor interface board requirements.

The following configuration files are created during the installation process and are required for the Image Sensor Lab.exe to properly execute. Do not delete or move any of these files.

6.1.1 isl_3010.bit

Default Location: C:\Program Files\Image Sensor Lab\isl_3010.bit

The *.bit* files are FPGA firmware files. The .bit file is only for the ISL-1600 model and is loaded into the FPGA when the Image Sensor Lab.exe application starts up or the device is reset through the Image Sensor Lab application software menu. The ISL-3200 does not use this bitfile and contains its code in on-board memory.

6.1.2 ISL.ini

Default Location: C:\Program Files\Image Sensor Lab\ISL.ini

The ISL.ini is the primary Image Sensor Lab.exe configuration file and like most configuration files is located in the root application directory. The first key in this configuration file is the SENSOR_INTERFACE key and the value is set to the power-up default, sensor-specific adapter board configuration file. A second section [Plugins] contains the list of plugin modules that the Image Sensor Lab.exe utilizes. The third and final section, [WindowPositions] is used by the application to remember the position and sizes of all the application panels.

6.1.3 Pdata.cfg

Default Location: C:\Program Files\Image Sensor Lab\pdata.cfg

The *pdata.cfg* file contains the parameter database for Image Sensor Lab. Image Sensor Lab application software has a defined set of parameters that are shared between the Image Sensor Lab application software, user-written plug-in code modules, and the sequencing and scripting engine. This file has a standard configuration file format with section names, keys, and values.

Users can edit this file and add new sections for any new parameters that they may wish to use within plugin code modules or user-written sequence/script files. Do NOT change the Name key or Parameter type of existing sections in this file.

6.1.4 Decoding.cfg

Default Location: C:\Program Files\Image Sensor Lab\decoding .cfg

The *decoding.cfg* file contains the names and decode mappings between the incoming sensor data stream and image pixel values for “Custom Decoding” types. This file is managed by the Image Sensor Lab application software and is NOT intended to be edited by the user.

6.1.5 PluginDecoders.cfg

Default Location: C:\Program Files\Image Sensor Lab\Plugindecoders .cfg

The *Plugindecoders.cfg* file contains the names and decode mappings between the incoming sensor data stream and image pixel values for “Plugin” decoding types. This file is managed by the Image Sensor Lab application software and is NOT intended to be edited by the user.

6.1.6 Recent files.ini

Default Location: C:\Program Files\Image Sensor Lab\recent files.ini

The *recent files.ini* file is used by the Image Sensor Lab.exe application to maintain a list of the most recently used sequence or script files. Every time a sequence file is loaded or executed through the application software, plugin modules, or drivers, this file is updated to reflect the newly executed file.

6.1.7 Sensor Adapter Board Configuration Files

There is a directory of sensor adapter board configuration files located at **\Image Sensor Lab\sensor adapters**. These files are standard Windows text-based configuration files with typical section headers and key value pairs. These configuration files can be opened, viewed, and edited using any standard text editing application (i.e., notepad, etc.). These files contain information about the adapter board, including which ISL power supplies are used and a label for each, which Digital I/O are used and a label for each, as well as additional information discussed later in this document.

The ISL application is generic and flexible, and provides undefined I/O that can be wired in a variety of ways on the sensor adapter boards. The configuration files allow more useful information to be displayed in the ISL application software. For instance, if a user decides to connect ISL power supply “A” (PS-A) to the analog power input of the image sensor, they can specify a label for PS-A to be “Sensor Analog Input” in the configuration file. Then, within the ISL application, wherever PS-A controls are displayed, the more meaningful label “Sensor Analog Input” is also displayed, making the overall operation more intuitive for the user.

Similarly, if a user decides to use an ISL digital output to connect to an image sensor RESET line, then the configuration file for that adapter configuration can include that label (RESET).

NOTE: Many ISL Sensor Adapter boards contain an on-board EEPROM that provides storage of all the configuration file parameters on the adapter board. The information within the EEPROM on the adapter board can be used instead of the information in a disk based adapter board file. When the Image Sensor Lab application software starts up, any connected adapter board is detected and the EEPROM contents are downloaded and used. If an EEPROM is not detected a disk based Sensor Adapter file is used instead. This file is specified in the first key of the ISL.ini file.

6.1.8 ISL Sequence Files

The ISL Application includes a sequencing engine that allows users to “script” ISL hardware configurations, image sensor communications, and ISL application software elements. Scriptable sequencing can automate many tedious setups and configurations and allows for more rapid switching between both ISL and image sensor operating modes. For example, when switching between Bayer-mode image acquisition and RGB565-mode, a user typically must set several image sensor registers, change the ISL acquisition window size and possibly the synchronization settings, as well as the Image Type and Data Decoding options. This sequence can be saved and recalled, using this sequence file mechanism.

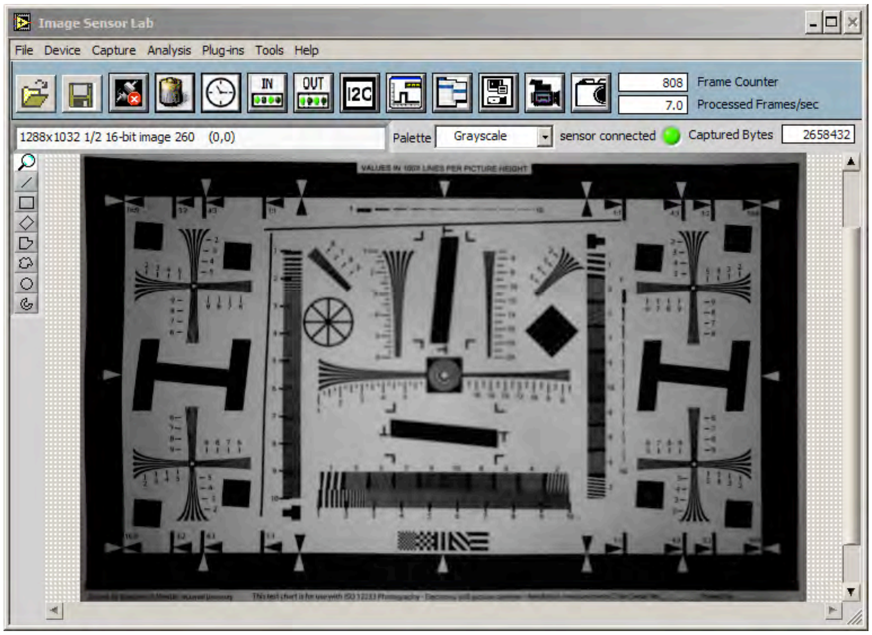
There is a directory of ISL sequence files located at **\Image Sensor Lab\ISL Sequence Files**. These sequence files contain commonly used settings and start-up procedures, some of which are specific to the Demo Micron/Aptina MT9M111 or MT9V111 image sensor and adapter board.

6.2 Running the ISL Application Software

To run the ISL application software locate the application and double click its ICON (typically located at **C:\Image Sensor Lab\Image Sensor Lab.exe**). The Image Sensor Application software displays the user interface shown to the right.

Application functions are accessed through menu items or by clicking on one of the toolbar icons along the top edge of the window.

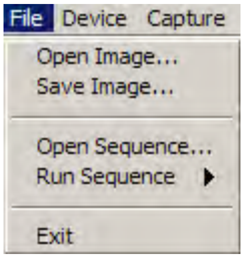
The application software is primarily geared towards image sensor interfacing, communications, and image acquisition and analysis but can also be used to load and process disk images.



808 Frame Counter
7.0 Processed Frames/sec
1288x1032 1/2 16-bit image 260 (0,0)
Palette Grayscale
sensor connected
Captured Bytes 2658432

6.2.1 ISL Menus

The Image Sensor Lab application software is a standard Windows menu driven software application with a “toolbar” of the more frequently used functions located at the top of the application window.



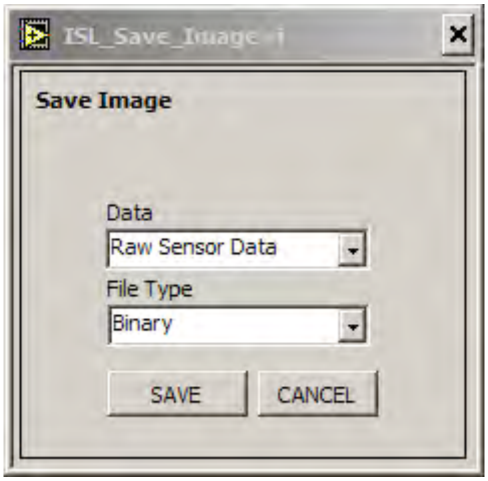
6.2.2 File Menu

The ISL File Menu allows the user to load and save images, open and run sequence-files, and exit the main Image Sensor Lab software application.

The Image Sensor Lab application software requires a live connection to the ISL hardware in order to operate. The software will automatically shut down if the ISL hardware is not detected.

Image Sensor Lab can save either the raw data coming from the image sensor, the image buffer data, which happens after the data is downloaded and decoded. The filtered and processed data can also optionally be saved. The output file formats supported are:

1. BMP
2. JPEG
3. PNG
4. TIFF
5. AIPD


ISL_Save_Image.vi

Save Image

Data: Raw Sensor Data

File Type: Binary

Some of these image file types do not support some of the image types used by the Image Sensor Lab application software and are grayed out and disabled when not appropriate for the current image type. As an example, when the current image type is 16-bits/pixel, BMP file type is disabled since that file format does not support 16-bit/pixel image data.

When loading disk images, the user dialog window shown to the right is presented. Since the disk file may contain raw image pixel data, the user must specify what type of image buffer to put the data into. Image Sensor Lab uses four primary image capture and processing buffers:

1. RAW
2. BAYER
3. RGB
4. YCbCr

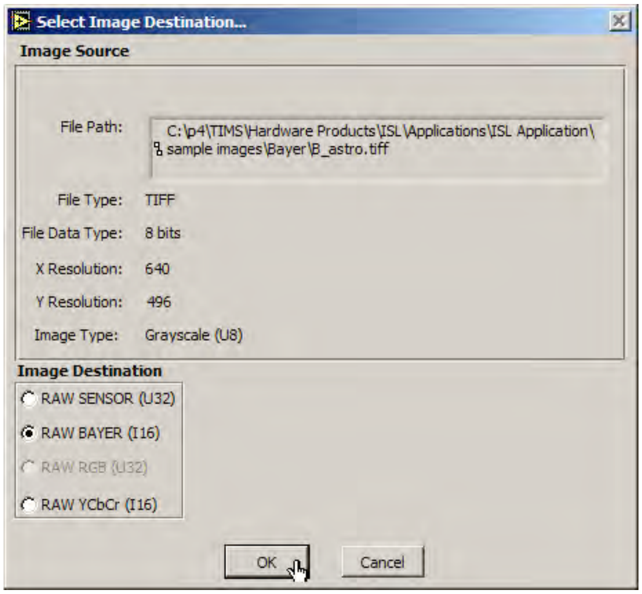

Select Image Destination...

Image Source

File Path: C:\p4\TIMS\Hardware Products\ISL\Applications\ISL Application\sample images\Bayer\B_astro.tiff

File Type: TIFF

File Data Type: 8 bits

X Resolution: 640

Y Resolution: 496

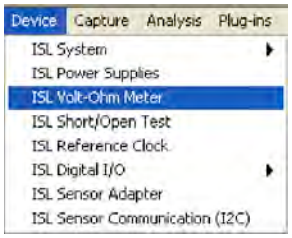
Image Type: Grayscale (U8)

Image Destination

○ RAW SENSOR (U32)
● RAW BAYER (I16)
○ RAW RGB (U32)
○ RAW YCbCr (I16)

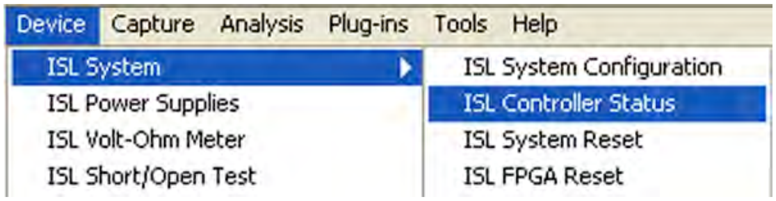
6.2.3 Device Menu

The Device Menu contains instrument panels for all ISL specific hardware functions. This is where the power supply, clock, digital I/O, Volt-Ohm meter, I2C sensor communications, and other functions can be found.



6.2.3.1 ISL System Submenu

The ISL System submenu contains access to an ISL System and FPGA Reset function, an ISL Controller Status panel and also a System Configuration panel detailing all of the installed components along with their serial numbers and versions.

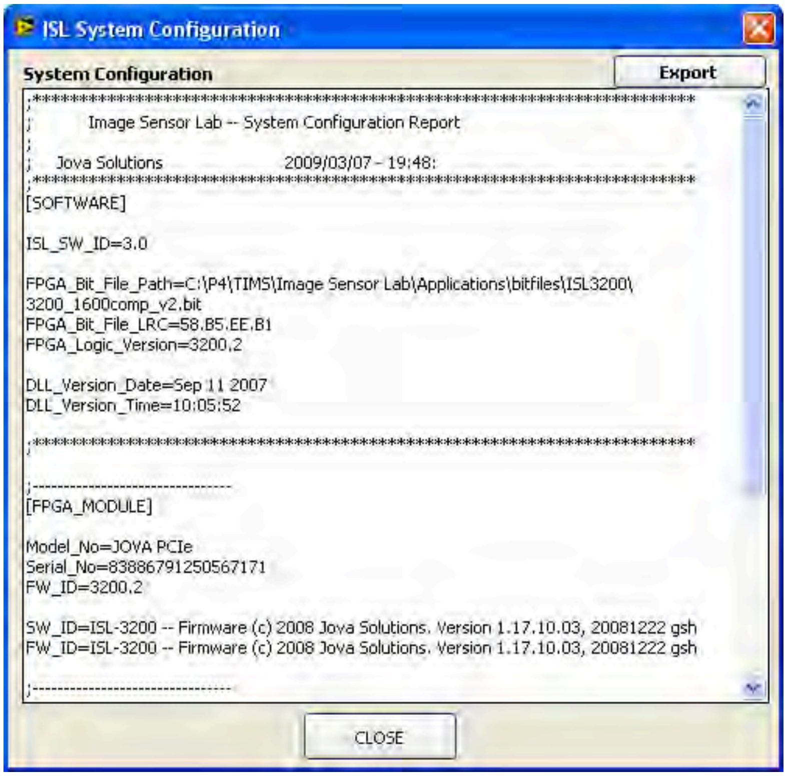


ISL System Configuration

The ISL System Configuration panel contains a complete listing of all installed ISL hardware and firmware components along with their serial number and revision.

When the ISL application software starts up, a system scan is performed to detect all installed components and query them for their configuration information.

When contacting Jova Solutions for technical support, it is always a good idea to include a copy of this information. You can use the Export button at the top of the screen to save a text file with this information.



```
;*****************************************************************************
;        Image Sensor Lab -- System Configuration Report
;
;    Jova Solutions                    2009/03/07 - 19:48:
;*****************************************************************************
[SOFTWARE]

ISL_SW_ID=3.0

FPGA_Bit_File_Path=C:\P4\TIMS\Image Sensor Lab\Applications\bitfiles\ISL3200\
3200_1600comp_v2.bit
FPGA_Bit_File_LRC=58.B5.EE.B1
FPGA_Logic_Version=3200.2

DLL_Version_Date=Sep 11 2007
DLL_Version_Time=10:05:52

;*****************************************************************************


;--------------------------------
[FPGA_MODULE]

Model_No=JOVA PCIe
Serial_No=83886791250567171
FW_ID=3200.2

SW_ID=ISL-3200 -- Firmware (c) 2008 Jova Solutions. Version 1.17.10.03, 20081222 gsh
FW_ID=ISL-3200 -- Firmware (c) 2008 Jova Solutions. Version 1.17.10.03, 20081222 gsh

;--------------------------------
```

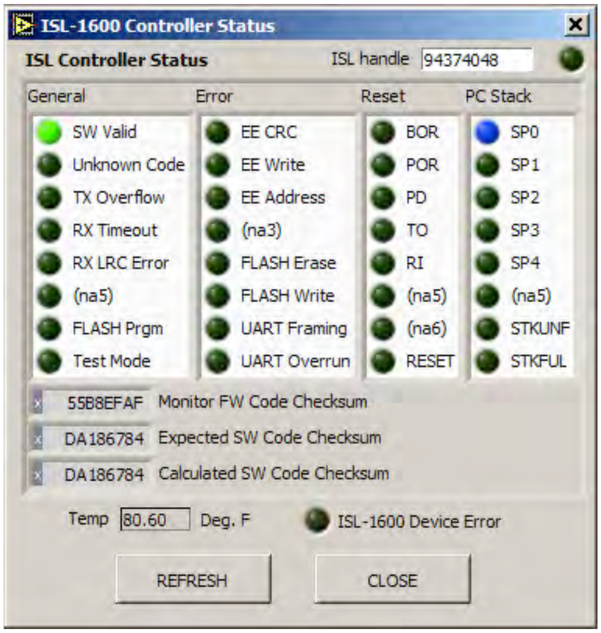
ISL CONTROLLER STATUS

The ISL Controller Status panel contains low-level status bits that are returned from the ISL controller and are typically needed to perform low-level troubleshooting in the event of an ISL hardware device malfunction.

When called for technical support, an engineer might ask that the ISL Controller Status panel be opened and for specific information from this panel.

There are two types of Firmware used by the controller – Monitor and Application. Both firmware checksums are displayed on this panel. This information can be helpful in determining what version of firmware you are running as well as help in determining if the firmware has become corrupt.

This panel also contains the ISL Motherboard temperature sensor readout and can be used to check for suspected overheating problems.



ISL-1600 Controller Status

ISL Controller Status — ISL handle 94374048 ●

General	Error	Reset	PC Stack
● SW Valid	● EE CRC	● BOR	● SP0
● Unknown Code	● EE Write	● POR	● SP1
● TX Overflow	● EE Address	● PD	● SP2
● RX Timeout	● (na3)	● TO	● SP3
● RX LRC Error	● FLASH Erase	● RI	● SP4
● (na5)	● FLASH Write	● (na5)	● (na5)
● FLASH Prgm	● UART Framing	● (na6)	● STKUNF
● Test Mode	● UART Overrun	● RESET	● STKFUL

55B8EFAF Monitor FW Code Checksum

DA186784 Expected SW Code Checksum

DA186784 Calculated SW Code Checksum

Temp 80.60 Deg. F ● ISL-1600 Device Error

ISL SYSTEM RESET

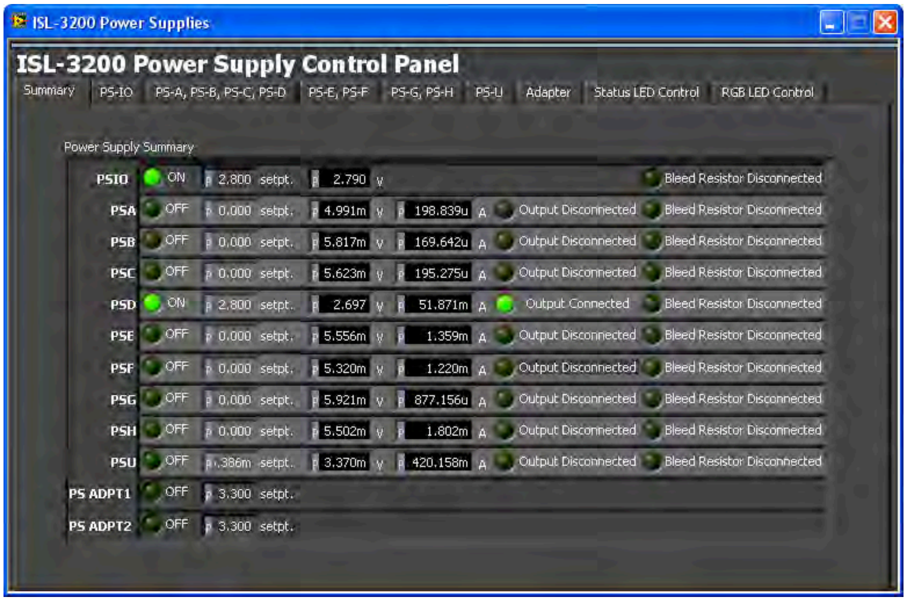
Selecting the ISL System RESET menu item will close and reopen communications with the ISL device. This is equivalent to a power-on reset.

6.2.3.2 ISL Power Supplies Submenu

The ISL Power Supply control panel provides access to the voltage set points and output relays of each installed power supply. Each power supply type has a specific operating range, which is displayed in the tab associated with that particular type (e.g., PS-IO, PSA-PSD, etc.)

The screen shot shown at the right shows the Summary tab. This tab shows the current status of each power supply and indicates if it is on or off, shows the setpoint, and a real-time measurement of the voltage output and current consumption of each supply.

Power supplies A through U have output mechanical relay isolation and the status of those relays can also be found on the Summary tab.



ISL-3200 Power Supplies

ISL-3200 Power Supply Control Panel

Power Supply Summary

PSIO	● ON	2.800 setpt.	2.790 V			● Bleed Resistor Disconnected
PSA	● OFF	0.000 setpt.	4.991m V	198.839u A	● Output Disconnected	● Bleed Resistor Disconnected
PSB	● OFF	0.000 setpt.	5.817m V	169.642u A	● Output Disconnected	● Bleed Resistor Disconnected
PSC	● OFF	0.000 setpt.	5.623m V	195.275u A	● Output Disconnected	● Bleed Resistor Disconnected
PSD	● ON	2.800 setpt.	2.697 V	51.871m A	● Output Connected	● Bleed Resistor Disconnected
PSE	● OFF	0.000 setpt.	5.556m V	1.359m A	● Output Disconnected	● Bleed Resistor Disconnected
PSF	● OFF	0.000 setpt.	5.320m V	1.220m A	● Output Disconnected	● Bleed Resistor Disconnected
PSG	● OFF	0.000 setpt.	5.921m V	877.156u A	● Output Disconnected	● Bleed Resistor Disconnected
PSH	● OFF	0.000 setpt.	5.502m V	1.802m A	● Output Disconnected	● Bleed Resistor Disconnected
PSU	● OFF	.386m setpt.	3.370m V	420.158m A	● Output Disconnected	● Bleed Resistor Disconnected
PS ADPT1	● OFF	3.300 setpt.				
PS ADPT2	● OFF	3.300 setpt.				

Power supplies A through U also have programmable bleed resistors that can be connected when the power supply is off, so that the output moves quickly to ground rather than floating at an arbitrary voltage.

Control for each of the power supply parameters can also be found on the individual power supply tabs of this panel.

6.2.3.3 PS-IO Tab of the ISL-3200 Power Supply Control Panel

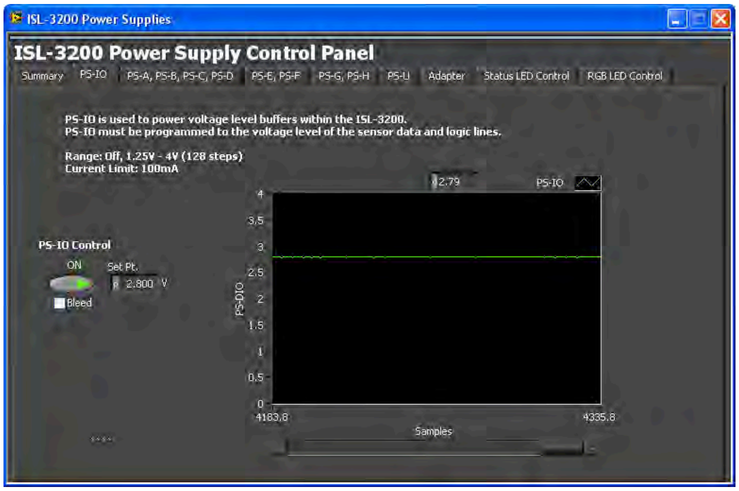
PS-IO is used to power the voltage level buffers within the ISL-3200 and must be programmed to the voltage level of the sensor data and logic lines.

PS-IO can be programmed for an output between 1.25V and 4.00V, with a 128-step resolution within that range.

The PS-IO tab contains the On/Off control and voltage setpoint control. The Bleed resistor can also be connected using the checkbox below the On/Off switch.

This panel also displays a real-time chart of the PS-IO voltage output, which can be useful during troubleshooting.

Right click on the graphs to access context menus that provide additional user configurable options related to the graph.



ISL-3200 Power Supplies

ISL-3200 Power Supply Control Panel

PS-IO is used to power voltage level buffers within the ISL-3200.
PS-IO must be programmed to the voltage level of the sensor data and logic lines.

Range: Off, 1.25V - 4V (128 steps)
Current Limit: 100mA

PS-IO Control: ON
Set Pt. 2.800 V
□ Bleed

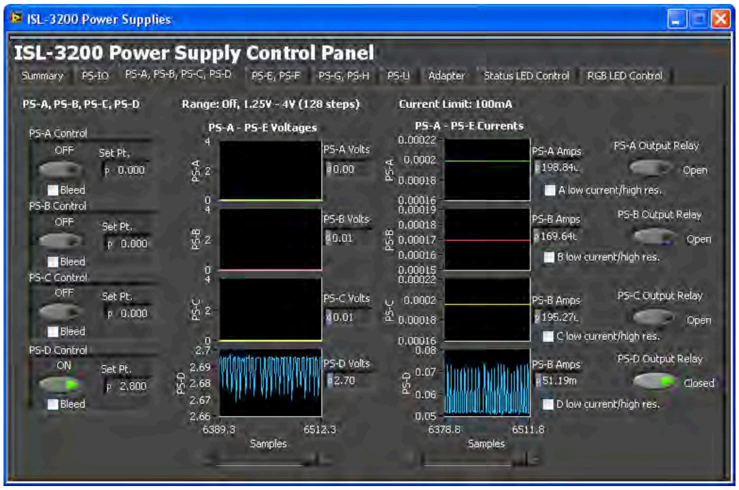
6.2.3.4 PS-A, PS-B, PS-C, PS-D Tab of the ISL-3200 Power Supply Control Panel

PS-A, PS-B, PS-C, and PS-D have a range from 1.25V to 4.00 V with a current limit of 100 mA per supply. This tab allows control of the On/Off, setpoint, and bleed resistor states and values.

The controls for the output relays that connect the power supply outputs to the ISL-3200 I/O connector can be found on the right side of this panel.

Real-time charts showing measured voltage and current draw can also be found on this panel. The charts and graphs can be configured through the context menus by right clicking on the graphs and charts.

When the measured current of any supply goes below about 1mA the “low current high resolution” mode checkbox can be checked which allows for nA resolution current measurements. If this mode is enabled while the current draw is above 1mA, the readings will not be accurate and the voltage output will not be stable nor at the desired setpoint.



ISL-3200 Power Supplies

ISL-3200 Power Supply Control Panel

PS-A, PS-B, PS-C, PS-D

Range: Off, 1.25V - 4V (128 steps)

Current Limit: 100mA

PS-A Control: OFF, Set Pt. 0.000, □ Bleed
PS-B Control: OFF, Set Pt. 0.000, □ Bleed
PS-C Control: OFF, Set Pt. 0.000, □ Bleed
PS-D Control: ON, Set Pt. 2.800, □ Bleed

PS-A Volts 0.00
PS-B Volts 0.01
PS-C Volts 0.01
PS-D Volts 2.70

PS-A Amps 198.84u
PS-B Amps 169.64u
PS-B Amps 195.27u
PS-B Amps 51.19m

PS-A Output Relay: Open
PS-B Output Relay: Open
PS-C Output Relay: Open
PS-D Output Relay: Closed

□ A low current/high res.
□ B low current/high res.
□ C low current/high res.
□ D low current/high res.

NOTE: The Output Relay switches must be in the OPEN position when installing or removing image sensor modules from the sensor adapter boards.

In addition, this button must be in the SENSOR CONNECTED position in order to connect power, I/O, and data lines to the image sensor. Image sensor communications will be prevented if this button is not set to the correct state.

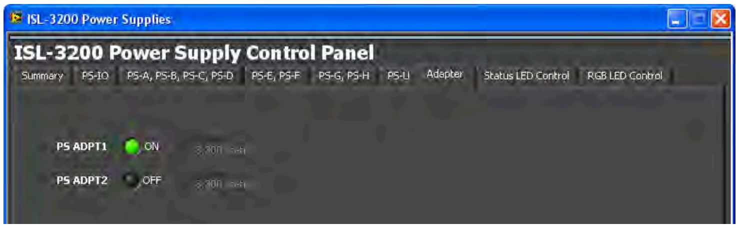
6.2.3.5 PS-E, PS-F, PS-G, PS-H and PS-U Tabs of the ISL-3200 Power Supply Control Panel

PS-E, PS-F, PS-G, PS-H, and PS-U tabs of the ISL-3200 power supply control panel are nearly identical to the PS-A through PS-D tab shown and described above.

6.2.3.6 Adapter Tab of the ISL-3200 Power Supply Control Panel

There are two 3.3V power supplies that can be enabled or disabled and are available at the I/O connectors.

The On/Off controls are found on this tab and the setpoint controls are grayed out as they are fixed at 3.3V.



PS ADPT1 ON 3.300

PS ADPT2 OFF 3.300

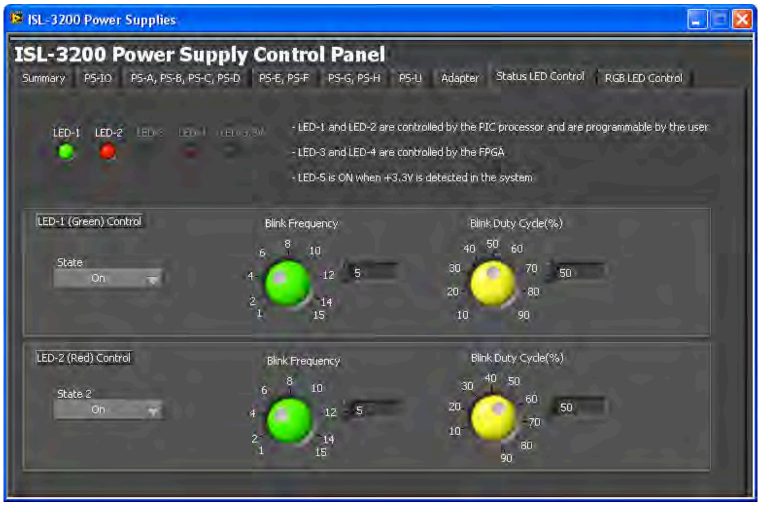
6.2.3.7 Status LED Control Tab of the ISL-3200 Power Supply Control Panel

The Status LED Control tab of the ISL-3200 Power Supply Control Panel can be used to control the LEDs on the ISL-3200 endplate near the power and USB/PCIe connectors.

These LEDs can be controlled to be on, off, or blinking. The blink frequency and duty cycle are also programmable.

The third and fourth LEDs are reserved for use and control by the FPGA.

The fifth LED is on when +3.3V is detected in the system.



- LED-1 and LED-2 are controlled by the PIC processor and are programmable by the user
- LED-3 and LED-4 are controlled by the FPGA
- LED-5 is ON when +3.3V is detected in the system

LED-1 (Green) Control: State On; Blink Frequency 5; Blink Duty Cycle(%) 50

LED-2 (Red) Control: State 2 On; Blink Frequency 5; Blink Duty Cycle(%) 50

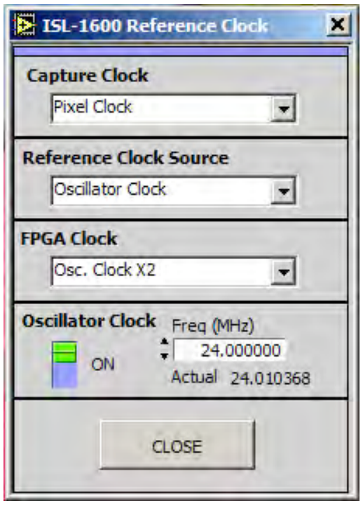
6.2.3.8 ISL Reference Clock Submenu

The ISL unit contains a programmable oscillator clock that can be adjusted up to 68 MHz. When setting the clock frequency, the user display panel also shows the actual clock frequency that was achieved (as shown in the screen capture to the right).

An additional clock source is provided by the FPGA and this panel contains a drop-list of the available FPGA Clock outputs. Currently there is only one option and that is to generate a frequency twice that of the oscillator clock (i.e., the FPGA is providing a "frequency doubler").

The Reference Clock routed to the image sensor can come directly from the Oscillator Clock or alternately from the FPGA Clock. Select the Reference Clock Source from the drop-list to control which clock to use and route to the image sensor.

You can also select which clock is used for capturing image data by selecting from the drop-list at the top of this panel. Image data capture can use either the Reference Clock going to the image sensor or the Pixel Clock coming back from the image sensor.



Capture Clock: Pixel Clock

Reference Clock Source: Oscillator Clock

FPGA Clock: Osc. Clock X2

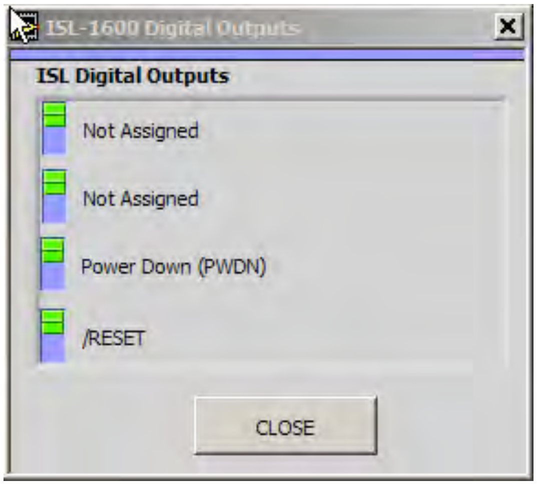
Oscillator Clock: ON; Freq (MHz) 24.000000; Actual 24.010368

6.2.3.9 ISL Digital I/O Submenu

The ISL unit contains four channels of digital input and four channels of digital output. Controlling or viewing these digital bits is accomplished through the two displays provided in this menu.

The ISL digital outputs panel is shown in the screen capture to the right. The four digital outputs can be connected to any of the additional digital inputs of the image sensor. In this example the ISL adapter board routes the Bout-0 digital output to the image sensor /RESET line and ISL digital output Bout-1 to the image sensor Power Down (PWDN) line.

NOTE: The sensor adapter EEPROM can contain information about the sensor connection including label names for the power supplies and digital I/O. The Image Sensor Lab software reads the adapter board EEPROM, pulls the label information out, and uses it in the displays as shown to the right. Here the /RESET and Power Down (PWDN) labels come from the adapter board EEPROM.

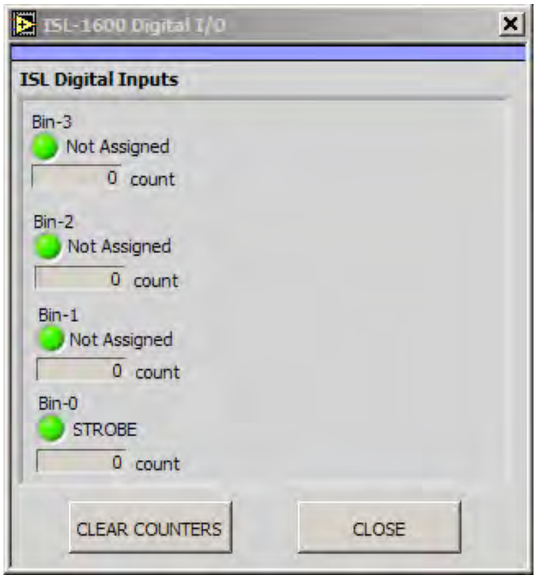


The ISL digital inputs panel is shown in the screen capture to the right. The four digital inputs can be connected to any of the additional digital outputs of the image sensor. In this example, the ISL adapter board routes the Bin-0 digital input to the image sensor STROBE output line.

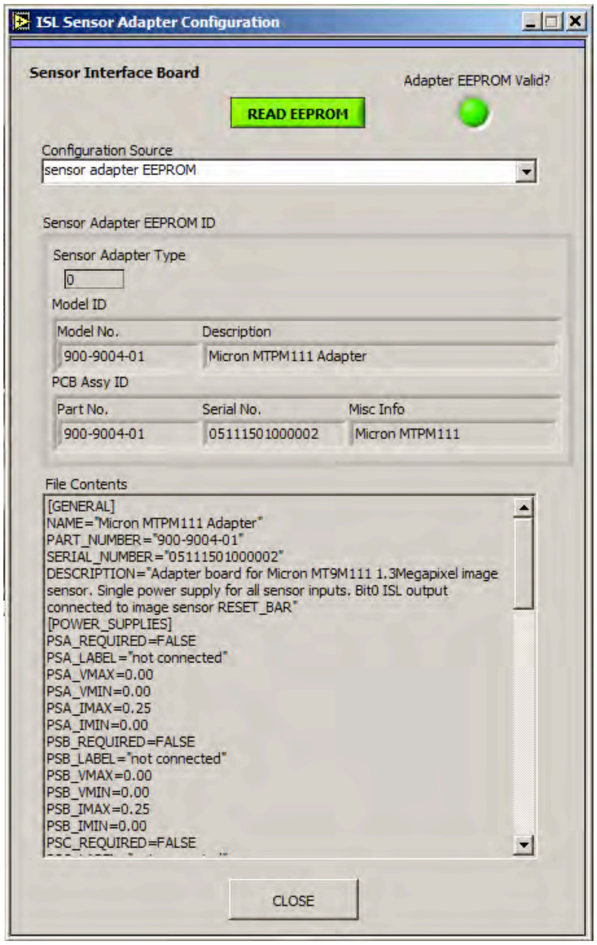
This digital input panel shows not only the current state of each of the four digital input lines, but also an event counter showing if activity has been detected on any of the digital inputs.

The count is not an actual pulse or edge count it is just an indicator that tells if ANY edge has been detected since the last time the count was read.

NOTE: The count should be used as a simple indicator to determine if the line is toggling or not and should not be used to measure actual pulses or transitions.



6.2.4 ISL Sensor Adapter Menu



Configuration Source: sensor adapter EEPROM

Sensor Adapter EEPROM ID

Sensor Adapter Type: 0

Model ID

Model No.	Description
900-9004-01	Micron MTPM111 Adapter

PCB Assy ID

Part No.	Serial No.	Misc Info
900-9004-01	05111501000002	Micron MTPM111

File Contents

```
[GENERAL]
NAME="Micron MTPM111 Adapter"
PART_NUMBER="900-9004-01"
SERIAL_NUMBER="05111501000002"
DESCRIPTION="Adapter board for Micron MT9M111 1.3Megapixel image
sensor. Single power supply for all sensor inputs. Bit0 ISL output
connected to image sensor RESET_BAR"
[POWER_SUPPLIES]
PSA_REQUIRED=FALSE
PSA_LABEL="not connected"
PSA_VMAX=0.00
PSA_VMIN=0.00
PSA_IMAX=0.25
PSA_IMIN=0.00
PSB_REQUIRED=FALSE
PSB_LABEL="not connected"
PSB_VMAX=0.00
PSB_VMIN=0.00
PSB_IMAX=0.25
PSB_IMIN=0.00
PSC_REQUIRED=FALSE
```

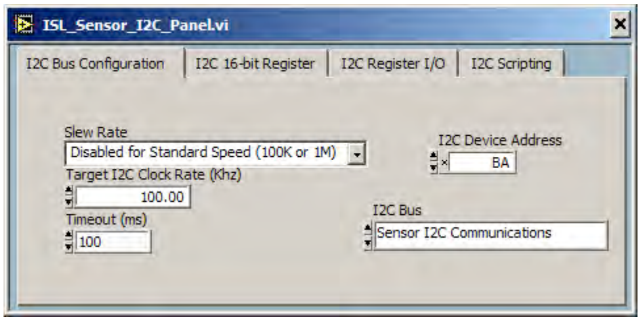
The ISL sensor adapter panel is shown in the screen shot at the right. Some sensor adapter boards contain EEPROMs with embedded configuration information. The Image Sensor Lab application software utilized the configuration information read from the sensor adapter EEPROM in several ways:

1. Sensor Connection Info
2. I2C Device Address
3. I2C Bus Configuration
4. Power supply Labels
5. Digital I/O Labels
6. Power Supply Limits
7. Minimum Requirements for sensor communications

Not all ISL image sensor adapters contain EEPROMs, so you might not see a green light on this panel indicating that the adapter EEPROM is valid. If the adapter board does not contain an EEPROM with configuration information, you can select a disk file with the equivalent information.

This panel shows the part number, serial number, and other descriptive information about the particular sensor adapter installed. In addition to the ID information shown in the Model ID box in the middle of the panel, the File Contents box shows more detailed configuration information, including how many power supplies are required and what voltage levels they need to be set to. Label information for the power supplies and digital I/O bits can also be included in this section and if they are included, the Image Sensor Lab application software will read them and use them as labels throughout the software when displaying the controls for the power supplies and digital I/O.

6.2.4.1 ISL Sensor Communication (I2C)



Slew Rate: Disabled for Standard Speed (100K or 1M)

Target I2C Clock Rate (Khz): 100.00

Timeout (ms): 100

I2C Device Address: x BA

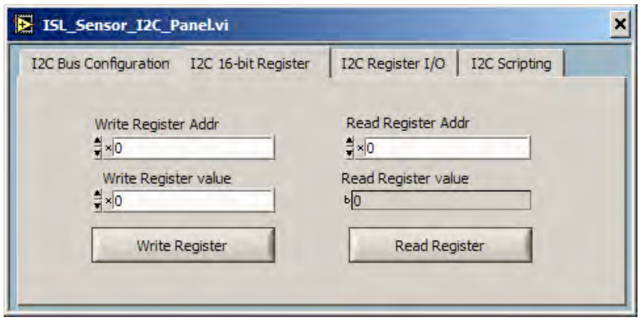
I2C Bus: Sensor I2C Communications

The ISL Sensor Communication (I2C) control panel is the primary image sensor communication interface. The I2C Bus Configuration tab contains communications settings used for I2C sensor communications transactions. The I2C clock rate, slew rate and timeout can be set using the controls shown on the left side of panel window. The I2C Device Address of the image sensor can be set with the upper right control. In this example, we are using device address 0xBA. The ISL contains two I2C busses that are available to the user. I2C Bus-1 is typically used for image sensor communications. I2C Bus-2 is sometimes used to communicate with devices on certain types of adapter boards that require additional devices in addition to the image sensor.

The 16-bit register Read/Write tab contains user controls for reading and writing register data. This particular tab is specific to image sensors with 8-bit register addresses and 16-bit register values. When reading or writing register values in this tab, the configuration options on the first tab are used during the communication transactions.

The I2C Register I/O tab is a more generic image sensor register interface and uses a “hex dot notation” for both the register and value controls and indicators. The example shown to the right has an 8-bit address, F0, and a 16-bit value (00.02). In the hex dot notation, use two characters to designate the hexadecimal value of the byte. If you need more that one byte, separate them with a dot (full-stop or period) symbol.

The I2C Scripting tab is more generic than the 16-bit register tab and can be used to communicate with non 8-bit address-16-bit value image sensors. This screen can also be used to send a series of I2C register writes and reads in one bundled action. In the I2C Tx entry section users can enter a series of bytes, entered in hex and separated by a period. Each line of the script executes a separate I2C communication transaction. Within each line all characters before an “R” or the end of line are written out in a single I2C write operation. If the character “R” is encountered, the characters following are interpreted as the number of bytes to read and then a Device Read I2C transaction is performed and the specified number of bytes are read and returned in the I2C Rx indicator.

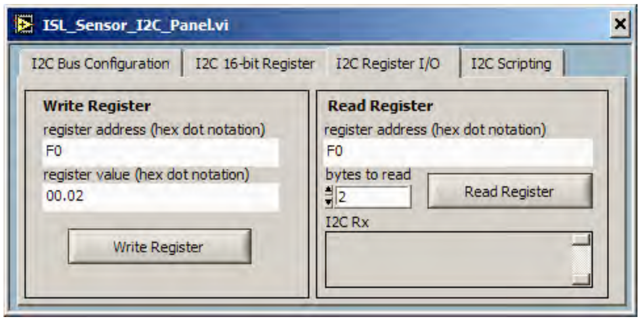


Write Register Addr x 0

Write Register value x 0

Read Register Addr x 0

Read Register value b 0



Write Register

register address (hex dot notation) F0

register value (hex dot notation) 00.02

Read Register

register address (hex dot notation) F0

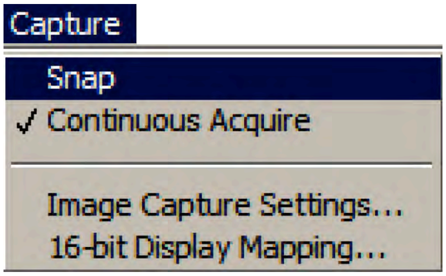
bytes to read 2

I2C Rx

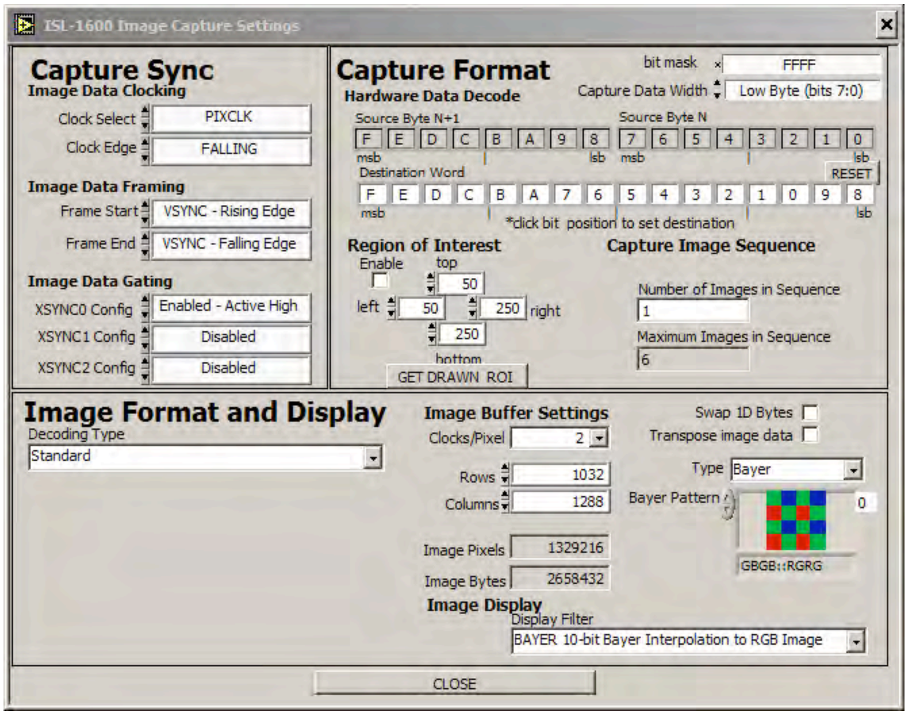
6.2.5 Capture Menu

The ISL Capture Menu provides access to configuration screens for the image data acquisition and decode settings. The image acquisition action is also initiated from this menu through the “Snap” or “Continuous Acquire” menu items.

The Image Sensor Lab application software provides full 16-bit/pixel processing and analysis and the 16-bit Display Mapping can be configured from this menu.



6.2.5.1 Image Capture Settings Control Panel


ISL-1600 Image Capture Settings

Capture Sync

Image Data Clocking
Clock Select: PIXCLK
Clock Edge: FALLING

Image Data Framing
Frame Start: VSYNC - Rising Edge
Frame End: VSYNC - Falling Edge

Image Data Gating
XSYNC0 Config: Enabled - Active High
XSYNC1 Config: Disabled
XSYNC2 Config: Disabled

Capture Format

bit mask: FFFF
Capture Data Width: Low Byte (bits 7:0)

Hardware Data Decode

Source Byte N+1								Source Byte N							
F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
msb							lsb	msb							lsb

Destination Word

F	E	D	C	B	A	7	6	5	4	3	2	1	0	9	8
msb															lsb

*click bit position to set destination

RESET

Region of Interest
Enable ☐
top: 50
left: 50
right: 250
bottom: 250
GET DRAWN ROI

Capture Image Sequence
Number of Images in Sequence: 1
Maximum Images in Sequence: 6

Image Format and Display

Decoding Type: Standard

Image Buffer Settings
Clocks/Pixel: 2
Rows: 1032
Columns: 1288
Image Pixels: 1329216
Image Bytes: 2658432

Swap 1D Bytes ☐
Transpose image data ☐
Type: Bayer
Bayer Pattern: 0
GBGB::RGRG

Image Display
Display Filter: BAYER 10-bit Bayer Interpolation to RGB Image

CLOSE

The **Image Capture Settings Control Panel** contains all the controls that affect how the raw image sensor signals are used to capture the data and then interpret that raw data into real image data.

Starting on the left side of the panel, the hardware “**Capture Sync**” settings are listed. The top section specifies how the image data is clocked in. The **Clock Select** drop-list can set to PIXCLK, the clock coming from the image sensor, or the REFCLK, the clock generated by the ISL hardware for use as a Master Clock. The ISL hardware can also clock image data in on either the rising or falling edge of the specified clock signal.

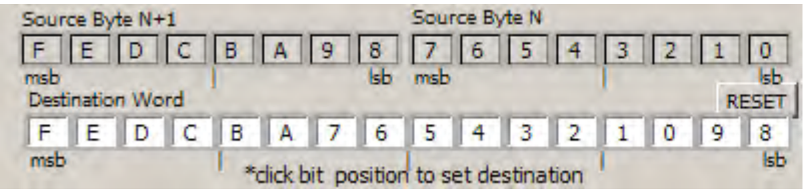
The **Image Data Framing** section allows the frame start and end edge to be specified. Image sensors typically will generate a FRAME VALID signal that is at a high state (logic 1) when the image data is valid, and falls to a low state (logic 0) during the blanking period, when image data is not valid.

The **Image Data Gating** (and Synchronization) section is where settings can be specified for up to three additional synchronization signals. XSYNC0 is typically connected to the HSync image sensor signal.

The upper right section of this panel is dedicated to “**Capture Format**” settings. A 16-bit pixel mask is also available and can be found in the upper right corner of the configuration panel. The pixel mask is a 16-bit value and is specified as 4 hexadecimal characters. A logical “AND” is performed with this value and each 16-bit value captured by the ISL hardware.

The **Sensor Data Width** section specifies which of the 16 available ISL hardware data lines are used to capture image data. If an 8-bit image sensor is used, this setting can be changed to either “Low Byte” or “High Byte” and then only 8-bits per clock are captured and packed into the on-board memory.

The **Number of Images in Sequence** box provides for entry of the number of images to acquire in a sequence. By default the Image Sensor Lab ISL hardware is configured to acquire just one image from the image sensor for each commanded capture. To capture full video rate sequential images set this control value to a number greater than one, and then for each commanded capture an image sequence is actually captured and made available for processing.



Source Byte N+1: F E D C B A 9 8 (msb … lsb)
Source Byte N: 7 6 5 4 3 2 1 0 (msb … lsb)
Destination Word: F E D C B A 7 6 5 4 3 2 1 0 9 8 (msb … lsb)
RESET
*click bit position to set destination

The **Hardware Data Decode** section contains controls that affect how the ISL hardware decodes the raw image sensor data **before** the image capture process. At this time, the decoding options are limited to re-mapping or "swizzling" the image data bits and/or masking the image data bits. This data decoding is performed at hardware speeds, in real-time, and is much more efficient and much faster than performing the same decoding in software on the host computer. The **Source Bytes** are shown at the top and the bits are labeled A7…A1 for the first or lower source byte, and B7…B1 for the 2nd or upper source byte. The **Destination Bytes** shown below have individual drop-lists for each bit position of the Destination Byte. Each bit of either Destination Byte can be mapped to any of the source bits.

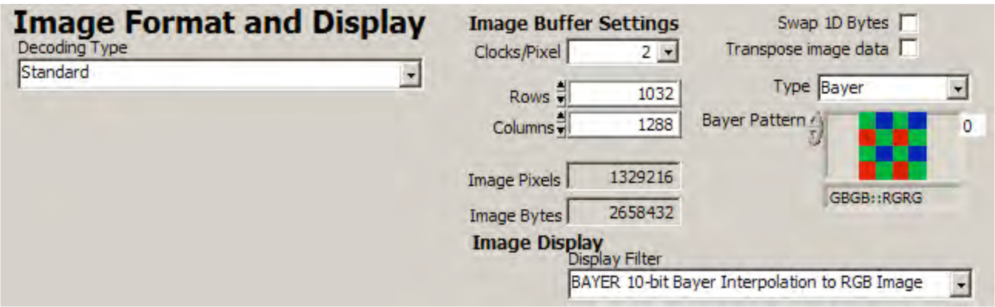


Image Format and Display
Decoding Type: Standard
Image Buffer Settings
Clocks/Pixel: 2
Rows: 1032
Columns: 1288
Image Pixels: 1329216
Image Bytes: 2658432
Swap 1D Bytes □
Transpose image data □
Type: Bayer
Bayer Pattern: 0
GBGB::RGRG
Image Display
Display Filter: BAYER 10-bit Bayer Interpolation to RGB Image

The **Image Format and Display** section contains settings that control how the image data is processed for every acquired image and then mapped into one of four image buffers used by the Image Sensor Lab application software on the host computer.

The decoding type drop-list contains three options: Standard, Custom, and Plug-in type decoding.

When "**Standard**" Decoding Type is selected, users can specify how many clocks per pixel the image sensor uses and also the image buffer type that the data should go into. This is sufficient information for the Image Sensor Lab application software to acquire data from the image sensor and place it in the image buffers used for the processing and display of the images.

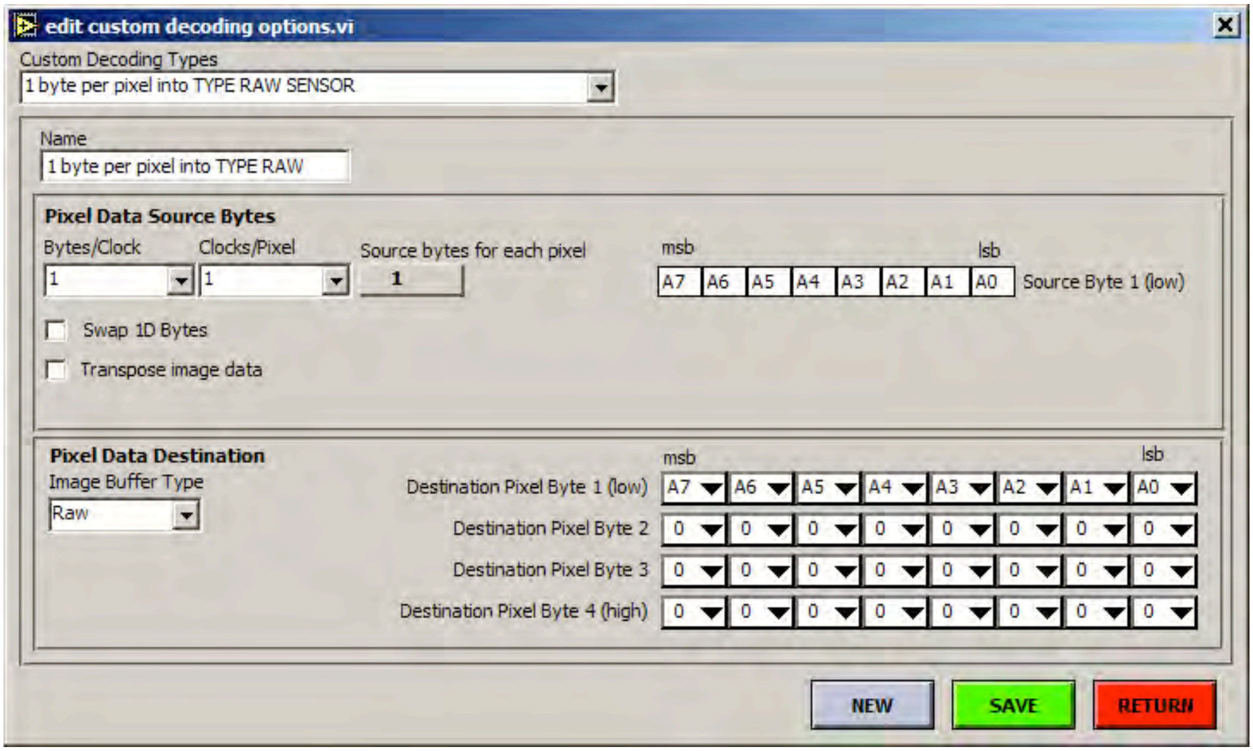
When "**Custom**" Decoding Type is selected, the user panel shown to the right is displayed. Custom decoding allows the user to map any of the bits in the image sensor data stream to any bits in an image buffer pixel. It is a generic mapping utility that is usually not as efficient as the built in or plug-in decoders, but does accommodate virtually any type of image sensor output.

Custom decoders that have been created are listed in the Custom Decoding Types drop-list.

Users can create and edit custom decoders by pressing the "Advanced Decoding Options…" button.

The image dimensions are also specified on this screen. The pixel data is reformatted into an image using these specified dimensions.

The Advanced Decoding Options panel is shown in the screen capture below.


edit custom decoding options.vi

Custom Decoding Types: 1 byte per pixel into TYPE RAW SENSOR

Name: 1 byte per pixel into TYPE RAW

Pixel Data Source Bytes

Bytes/Clock: 1

Clocks/Pixel: 1

Source bytes for each pixel: 1

msb							lsb	
A7	A6	A5	A4	A3	A2	A1	A0	Source Byte 1 (low)

□ Swap 1D Bytes

□ Transpose image data

Pixel Data Destination

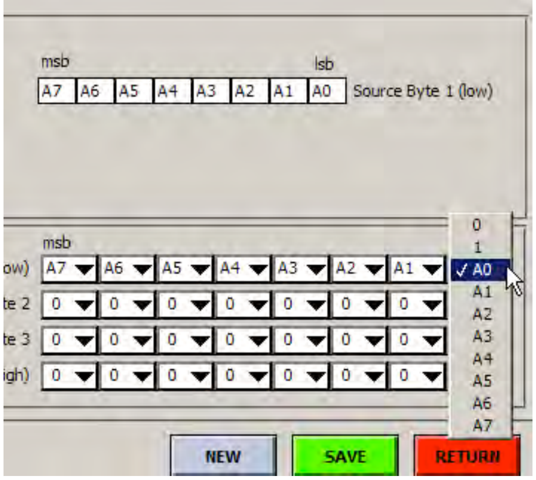
Image Buffer Type: Raw

	msb							lsb
Destination Pixel Byte 1 (low)	A7	A6	A5	A4	A3	A2	A1	A0
Destination Pixel Byte 2	0	0	0	0	0	0	0	0
Destination Pixel Byte 3	0	0	0	0	0	0	0	0
Destination Pixel Byte 4 (high)	0	0	0	0	0	0	0	0

The custom decoder to view or edit is selected from the top drop-list. Custom decoding definitions are contained in ***the Image Sensor Lab\Decoding.cfg*** file. The entries in this file define the items that are shown in the top drop-list. You can create new custom decoding types using this interface panel, or you can edit the ***decoding.***cfg file directly to create or edit the custom decoding types.

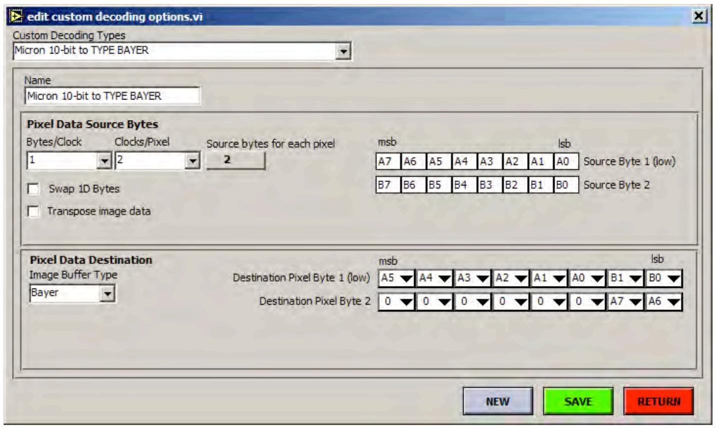
Once a Custom Decoding Type is selected from the top drop-list, the name, bytes/clock, clocks/pixel, as well as the optional byte swap and transpose selections can be edited. You will notice that the source byte and bits are shown graphically to the right of these option settings. This area shows the source data bits that are available for mapping to a destination pixel.

In the section at the bottom, the user first selects the Image Buffer Type from the drop-list. Raw and RGB image buffers use 32-bits/pixel for data storage. Bayer and YCbCr use 16-bits/pixel for data storage. You will notice this in the graphical representation to the right of the drop-list. In the example above a Raw image buffer is selected which contains 32-bits or four bytes of data. The user then can use the drop-lists that make up each bit of the displayed image buffer bytes and select the source bit that gets mapped to this location. The example shown to the right shows a user editing the least significant byte (lsb) or the low byte of the image buffer. Notice that the available selections are 0, 1, A0, …, A7, the available. If the user had selected more than one source byte per pixel, this drop-list would contain more options to match the available bits.


msb							lsb	
A7	A6	A5	A4	A3	A2	A1	A0	Source Byte 1 (low)

	msb						
ow)	A7	A6	A5	A4	A3	A2	A1
te 2	0	0	0	0	0	0	0
te 3	0	0	0	0	0	0	0
igh)	0	0	0	0	0	0	0

The screen capture at the right shows the decoding used for the Micron MT9M111 image sensor in Bayer mode. This sensor has 8-bit data output per clock. It takes this sensor 2 clock cycles to clock out the data for one 10-bit per pixel Bayer mode image pixel. For this particular decoding the first two bits of the second byte are actually the low order bits of the 10-bit value.



edit custom decoding options.vi

Custom Decoding Types: Micron 10-bit to TYPE BAYER

Name: Micron 10-bit to TYPE BAYER

Pixel Data Source Bytes

Bytes/Clock: 1

Clocks/Pixel: 2

Source bytes for each pixel: 2

□ Swap 1D Bytes

□ Transpose image data

	msb							lsb
Source Byte 1 (low)	A7	A6	A5	A4	A3	A2	A1	A0
Source Byte 2	B7	B6	B5	B4	B3	B2	B1	B0

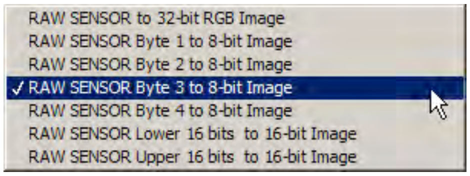
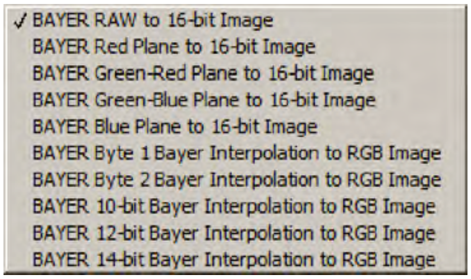
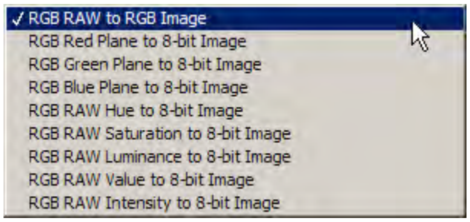
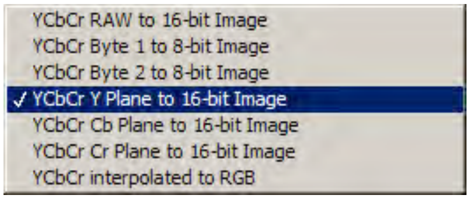
Pixel Data Destination

Image Buffer Type: Bayer

	msb							lsb
Destination Pixel Byte 1 (low)	A5	A4	A3	A2	A1	A0	B1	B0
Destination Pixel Byte 2	0	0	0	0	0	0	A7	A6

Display and Processing Filter Options

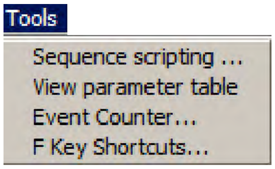
Each image type has a different set of display filters that can be used to further restrict which image information is used for processing and display. The following table details the selections for each type.

RAW SENSOR data can be interpreted and displayed as RGB data or be interpreted as a variety of monochrome 8- or 16-bit data. Any one of the 4 bytes of the pixel can also be selectively viewed and processed.	 RAW SENSOR to 32-bit RGB Image RAW SENSOR Byte 1 to 8-bit Image RAW SENSOR Byte 2 to 8-bit Image ✓ RAW SENSOR Byte 3 to 8-bit Image RAW SENSOR Byte 4 to 8-bit Image RAW SENSOR Lower 16 bits to 16-bit Image RAW SENSOR Upper 16 bits to 16-bit Image
BAYER data can be interpreted in a variety of ways. Users can view the raw Bayer data as a 16-bit monochrome image or select any of the four Bayer color planes. Bayer interpolation algorithms are also provided for conversions to RGB image format.	 ✓ BAYER RAW to 16-bit Image BAYER Red Plane to 16-bit Image BAYER Green-Red Plane to 16-bit Image BAYER Green-Blue Plane to 16-bit Image BAYER Blue Plane to 16-bit Image BAYER Byte 1 Bayer Interpolation to RGB Image BAYER Byte 2 Bayer Interpolation to RGB Image BAYER 10-bit Bayer Interpolation to RGB Image BAYER 12-bit Bayer Interpolation to RGB Image BAYER 14-bit Bayer Interpolation to RGB Image
RGB data can be viewed and analyzed as a whole or as any of the separate R, G, or B color planes. The Hue, Saturation, Luminance, Value, and Intensity components can also be extracted from the RGB image for display and analysis.	 ✓ RGB RAW to RGB Image RGB Red Plane to 8-bit Image RGB Green Plane to 8-bit Image RGB Blue Plane to 8-bit Image RGB RAW Hue to 8-bit Image RGB RAW Saturation to 8-bit Image RGB RAW Luminance to 8-bit Image RGB RAW Value to 8-bit Image RGB RAW Intensity to 8-bit Image
YCbCr data can be viewed as a raw 16-bit image, or either the first or second byte can be viewed as 8-bit monochrome. In addition, the Y, Cb, or Cr image-data component can be extracted for display or analysis, and a YCbCr color interpolation algorithm is also available for viewing the resultant RGB image.	 YCbCr RAW to 16-bit Image YCbCr Byte 1 to 8-bit Image YCbCr Byte 2 to 8-bit Image ✓ YCbCr Y Plane to 16-bit Image YCbCr Cb Plane to 16-bit Image YCbCr Cr Plane to 16-bit Image YCbCr interpolated to RGB

6.2.6 Tools Menu

The ISL Tools Menu provides access to several productivity tools including a sequencing and scripting panel, as well as a built-in timer/counter for measuring any of the image sensor signals connected to the ISL hardware.

There is also an F Key Shortcut Panel where sequence files sets and script mappings can be called out.

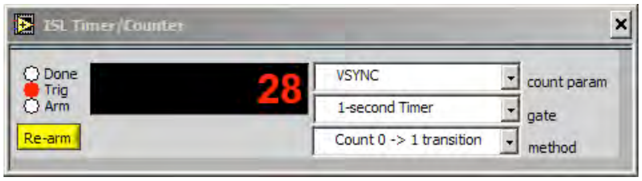


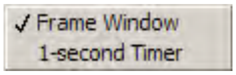
The ISL parameter table can also be viewed from this menu. The parameter table lists all the application variables in a convenient table format for quick scanning of the entire system configuration. This panel is continuously updated and can be viewed while executing scripts.

6.2.6.1 Event Counter Submenu

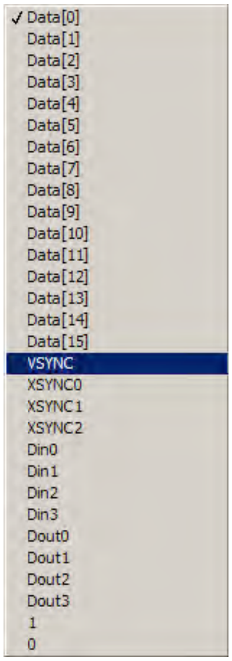
The ISL Event Counter is a small floating window that provides limited counter/timer measurements of any of the image sensor signals connected to the ISL hardware. The event counter can measure signal transitions per frame or per second (i.e., the gate is controlled by the Frame Start to End condition or a one-second timer.) The counter/timer can also count capture-clock transitions when the selected count parameter is at the specified logic level, and over the gate period.

The count parameter drop-list contains a complete list of all sensor data lines, sync signal lines, and the digital input and output lines. In addition, a logic level 0 and logic level 1 are also available.

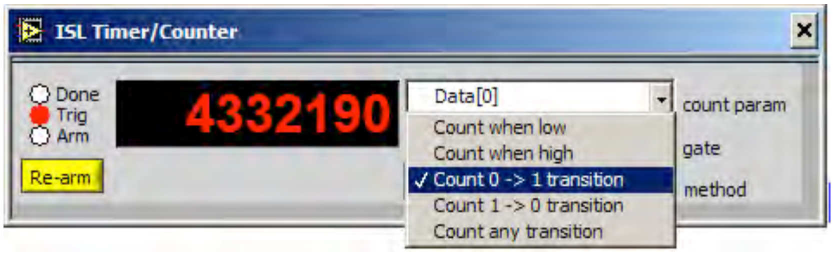




The counter gate period drop-list allows selection of either a one second timer or a Frame Window (the frame window being defined in the Capture Configuration panel).



The method drop-list controls the counting method used. When “Count when low” is selected the counter/timer will count the capture clock transitions when the count parameter is at logic low, and over the specified gate period. To measure the actual count parameter transitions instead of the capture clock transition, use the “Count 0 -> 1 transition” or the “Count 1 -> 0 transition” options.



6.2.7 Sequence Scripting Submenu

The Image Sensor Lab application software contains a sequencing engine and application scripting, which enable the automation of a variety of tasks that could otherwise be tedious and time consuming.

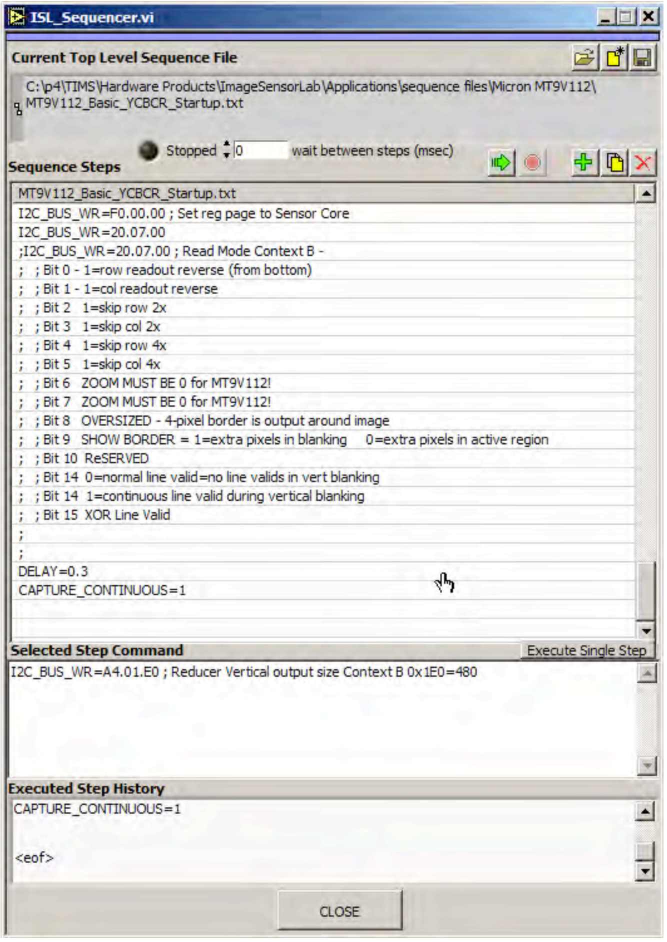
The primary user interface is accessed through this menu item, which launches the interface panel shown to the right. This panel allows you to build, edit, load, save, and execute timed sequences of commands that control the ISL hardware, image sensor, and the Image Sensor Lab application software.

ISL sequence files are simple text-based script files. Each line of the script file contains a single command or comment. Any line starting with a semicolon (;) is considered a comment and is ignored. On a command line, any characters including and following a semicolon (;) are ignored.

Command parameters are separated by commas. There is an equal sign between the command and command parameters. Commas separate multiple command parameters.

The sequence scripting functionality is useful with sensors that require very specific timed sequences of clocks, power, and digital inputs in order to operate.

The top indicator shows the current, top-level sequence file loaded and ready for execution. The contents of the sequence file are shown in the “Sequence Steps” list-box below.



Current Top Level Sequence File

C:\p4\TIMS\Hardware Products\ImageSensorLab\Applications\sequence files\Micron MT9V112\MT9V112_Basic_YCBCR_Startup.txt

Stopped 0 wait between steps (msec)

Sequence Steps

```
MT9V112_Basic_YCBCR_Startup.txt
I2C_BUS_WR=F0.00.00 ; Set reg page to Sensor Core
I2C_BUS_WR=20.07.00
;I2C_BUS_WR=20.07.00 ; Read Mode Context B -
;  ; Bit 0 - 1=row readout reverse (from bottom)
;  ; Bit 1 - 1=col readout reverse
;  ; Bit 2   1=skip row 2x
;  ; Bit 3   1=skip col 2x
;  ; Bit 4   1=skip row 4x
;  ; Bit 5   1=skip col 4x
;  ; Bit 6   ZOOM MUST BE 0 for MT9V112!
;  ; Bit 7   ZOOM MUST BE 0 for MT9V112!
;  ; Bit 8   OVERSIZED - 4-pixel border is output around image
;  ; Bit 9   SHOW BORDER = 1=extra pixels in blanking    0=extra pixels in active region
;  ; Bit 10  ReSERVED
;  ; Bit 14  0=normal line valid=no line valids in vert blanking
;  ; Bit 14  1=continuous line valid during vertical blanking
;  ; Bit 15  XOR Line Valid
;
;
DELAY=0.3
CAPTURE_CONTINUOUS=1
```

Selected Step Command

```
I2C_BUS_WR=A4.01.E0 ; Reducer Vertical output size Context B 0x1E0=480
```

Executed Step History

```
CAPTURE_CONTINUOUS=1


<eof>
```

You can load, create new, or save sequence files using the three small icons just above the **Current Top Level Sequence File** indicator at the top.

To run a sequence file press the ▶ button to start execution. During the sequence file execution, the **Sequence Steps** list-box highlights the step that is currently being executed. Enter a delay in the “Step Delay (msec)” control to slow down the execution if desired.

You can also edit, add, copy, or delete steps using the small icons just above the Sequence Steps list-box.

The currently selected step is also displayed in the “Selected Step Command” panel control. This is where the user edits and executes the single step, if desired.

ISL Sequence Scripting Commands

COMMAND	PARAMETER	DESCRIPTION
DELAY		Delay the sequence file execution
	Delay	Seconds to delay
		Example: **DELAY=1.2 ; Delay 1.2 seconds**
DIALOG		Display a simple modal user dialog with an OK button
	Dialog Text	Text to display in the user dialog window
		Example: **DIALOG=PRESS OK TO CONTINUE**
SCRIPT		Execute the specified script
	File Name	Name of the script file to execute. File must be in the same directory as the calling script file.
		Example: **SCRIPT=POWER OFF.txt**
TALK		Speak a string of text
	Text to speak	The text that will be converted to speech and played on the computers sound output
		Example: **TALK=Setting Sensor to BAYER mode.**
PS_V_SET		Set the voltage output setpoint of an individual power supply
	PS ID	ID of power supply (A, B, C, D, or E)
	Vset	A floating point number between 0.0 and 4.0 that is used as the output setpoint for the specified power supply.
		Example: **PS_V_SET=E,2.8 ; Set PSE to 2.8 Volts**
PS_I_SET		Set the current limit of an individual power supply.
	PS ID	ID of power supply (A, B, C, D, or E)
	Iset	A floating point number between 0.0 and 4.0 that is used as the output setpoint for the specified power supply.
		Example: **PS_I_SET=E,0.1 ; Limit PSE current to 100 mA**
PS_ENABLE		Enables/Disables the specified power supply.
	PS ID	ID of power supply (A, B, C, D, or E)
	ENABLED?	True or False indicating the state of the Power Supply Enable.

COMMAND	PARAMETER	DESCRIPTION
		Example: **PS_ENABLE=E,True ; Enable PSE**
SENSOR_CONNECT (FET_ENABLE)		Enables/Disables the FET switches which connect or disconnect the sensor data, sync, and I/O connections from the ISL hardware. NOTE: This command is the equivalent of performing the following individual commands: - MSB_ENABLE - LSB_ENABLE - SYNC_ENABLE - ENABLE_BITIO
	ENABLED?	True or False indicating the state of the image sensor connection.
		Example: **SENSOR_CONNECT=True ; Connect the sensor to the ISL hardware**
MSB_ENABLE		Enables/Disables the FET switches which connect or disconnect the MSB sensor data connections from the ISL.
	ENABLED?	True or False indicating the state of the sensor data MSB FET switches.
		Example: **MSB_ENABLE=True ; Connect the sensor MSB data lines to the ISL**
LSB_ENABLE		Enables/Disables the FET switches which connect or disconnect the LSB sensor data connections from the ISL.
	ENABLED?	True or False indicating the state of the sensor data LSB FET switches.
		Example: **LSB_ENABLE=True ; Connect the sensor LSB data lines to the ISL hardware**
SYNC_ENABLE		Enables/Disables the FET switches which connect or disconnect the sensor SYNC lines to/from the ISL hardware.
	ENABLED?	True or False indicating the state of the sensor data SYNC line FET switches.
		Example: **SYNC_ENABLE=True ; Connect the sensor SYNC lines to the ISL hardware.**
ENABLE_BITIO		Enables/Disables the FET switches which connect or disconnect the sensor Digital I/O lines to/from the ISL hardware.

COMMAND	PARAMETER	DESCRIPTION
	ENABLED?	True or False indicating the state of the sensor digital I/O line FET switches.
		Example: **ENABLE_BITIO=True ; Connect the sensor SYNC lines to the ISL hardware.**
SENSOR_COMM_ENABLE		Enables/Disables the FET switches which connect or disconnect the sensor communications connections to/from the ISL hardware.
	ENABLED?	True or False indicating the state of the sensor communications FET switches.
		Example: **SENSOR_COMM_ENABLE=True ; Connect the sensor communication lines to the ISL hardware.**
MSB_DIRECTION		Specifies the direction of the image sensor MSB data line buffers. By default the sensor data line buffers are configured as inputs, as the image sensor is usually the one driving those lines.
	Direction	1=Output 0=Input
		Example: **MSB_DIRECTION=0 ; Configure the ISL MSB buffers as inputs from the image sensor.**
LSB_DIRECTION		Specifies the direction of the image sensor LSB data line buffers. By default the sensor data line buffers are configured as inputs, as the image sensor is usually the one driving those lines.
	Direction	1=Output 0=Input
		Example: **LSB_DIRECTION=0 ; Configure the ISL LSB buffers as inputs from the image sensor.**
SYNC_DIRECTION		Specifies the direction of the image sensor SYNC line buffers. By default the sensor SYNC line buffers are configured as inputs, as the image sensor is usually the one driving those lines.
	Direction	1=Output 0=Input
		Example: **SYNC_DIRECTION=0 ; Configure the ISL SYNC buffers as inputs from the image sensor.**
OSCCLK_SET (REFCLK_SET)		Sets the output frequency of the ISL on-board oscillator clock.
	On/Off	Turns the oscillator clock on or off

COMMAND	PARAMETER	DESCRIPTION
	Frequency	Frequency setpoint for the ISL reference clock output. Note: a single scaling character G,M,K,k,,m,u,n, or p is allowed directly following the number.
		Example: **OSCCLK_SET=True, 54M ; Set REFCLK output to 54 MHz.**
REFCLK_SOURCE		Specifies which clock, the ISL on-board oscillator, or the ISL FPGA-based clock, should be routed to the image sensor as the REFCLK.
	Source	0=ISL on-board oscillator 1=ISL FPGA based clock 2=None (leave the sensor clock input disconnected)
		Example: **REFCLK_SOURCE=0 ; Use the ISL on-board oscillator as the REFCLK routed to the image sensor.**
FPGA_CLOCK_SELECT		Specifies which FPGA clock mode to use.
	mode	0=Oscillator clock doubler
		Example: **FPGA_CLOCK_SELECT=0 ; Select the oscillator clock doubler function of the FPGA based clock.**
BOUT_SET		Sets the state of the four ISL digital output lines.
	Bit	Specifies the bit number (0,1,2, or 3) to control
	Output State	True or False to specify the output sate
		Example: **BOUT_SET=0,False ; Set Bit 0 to False**
SENSOR_COMM_TYPE		Specifies the type of sensor communications to be used. Either I2C or SPI communication protocol can be used for sensor communications.
	Type	0=I2C 1=SPI
		Example: **SENSOR_COMM_TYPE=0 ; Use the I2C protocol for sensor communications.**
I2C_BUS_NO		Specifies the use one of two possible I2C busses in the ISL available to users. I2C Bus 1 is typically used to communicate with the sensor. I2C Bus 2 is typically used to communicate with other devices on the sensor adapter board.
	Bus Number	Specifies the bit number (1 or 2) to use for further I2C communications
		Example: **I2C_BUS_NO=1 ; Use I2C Bus 1 for I2C commands**

COMMAND	PARAMETER	DESCRIPTION
I2C_SLEW_RATE		Specifies the slew rate mode for I2C communications.
	Mode	0 = Enabled for High Speed (400KHz) 1= Disabled for Standard Speed (100KHz or 1MHz)
		Example: **I2C_SLEW_RATE=1 ; set slew rate for 100Khz clock.**
I2C__BUS_CLOCK_FREQ		Specifies the clock rate for I2C communications.
	Target Frequency	Specifies the frequency of the clock used for I2C communications
		Example: **I2C_BUS_CLOCK_FREQ=100k ; use 100KHz clock**
I2C_DEVICE_ADDRESS		Specifies the I2C Device Address used for further I2C communications
	Device Address	A 2 digit hex code specifying the 7-bit device address used for I2C communications
		Example: **I2C_DECVICE_ADDRESS=BA ; Use hex addr 0xBA**
I2C_BUS_CONFIG		Configures the I2C Bus Number, Bus Slew Rate, Bus Clock Rate, and Timeout used for I2C communications. Note: This function consolidates several functions above so that all I2C configuration can be handled in one step.
	I2C Bus No	Specifies the bit number (1 or 2) to use for further I2C communications
	I2C Bus Slew Rate	0 = Enabled for High Speed (400KHz) 1= Disabled for Standard Speed (100KHz or 1MHz)
	I2C Clock Rate	Specifies the frequency of the clock used for I2C communications
	I2C Timeout	Specifies the timeout in msec used for I2C communications transactions.
		Example: **I2C_BUS_CONFIG=1,1,100k,100**
I2C_BUS_WR		Perform an I2C Write Operation
	Transmit Data	The I2C Transmit buffer in Hex dot notation.
		Example: **I2C_BUS_WR=F0.00.02 ; Set register 0xF0 to 2.** **; F0 = 8-bit reg addr** **; 16-bit reg value = 2**
I2C_BUS_WR_MSK		Perform an I2C Write Operation with Bit Mask
	Transmit Data	The I2C Transmit buffer in Hex dot notation.

COMMAND	PARAMETER	DESCRIPTION
		Example: **I2C_BUS_WR_MSK=F0.00.02,00.02** **; Set the second bit only of register 0xF0.** **; F0 = 8-bit reg add** **; 16-bit reg value = 00.02** **;16-bit mask value = 00.02**
IMAGE_TYPE		Specifies the ISL Image Buffer Type used for acquisition and display
	Image Type	0= Raw 1= BAYER 2= RGB 3= YCbCr
		Example: **IMAGE_TYPE=BAYER ; use BAYER image buffer**
REWIRE_MASK **(HARDWARE_DECODE)**		Specifies the hardware decoding map used by the FPGA during image data acquisition.
	DecodeMap	The decode map is a 16 character text string where each character in the string represents a bit in the 16-bit data word. The first character in the 16-character string represents the MSB, and if the intended destination of this bit is to remain in the most significant bit location then the DecodeMap string character should be set to “F”. If the intended destination of this bit is to switch it to the lower location or LSB, then the DecodeMap string character should be set to “0”. If the intended destination of this bit is one down from the MSB, the set the DecodeMap character to an “E”. The decode map character is the Hexadecimal equivalent to the bit number of the destination location.
		Example: **HARDWARE_DECODE=FEDCBA9876543210 ; use a straight through mapping (i.e. keep the original bit locations)**
PIXEL_MASK		Specifies a 16-bit pixel mask applied to the image sensor data just before the data is captured into on-board memory.
	Mask	A 4 character hexadecimal string representing the 16-bit mask value applied to the image sensor data.
		Example: **PIXEL_MASK=FFFF ; enable all 16-bits.**
FPGA_RESET		Specifies the state of the FPGA Reset function.
	STATE	1=FPGA in RESET State 0=FPGA normal operation

COMMAND	PARAMETER	DESCRIPTION
		Example: **FPGA_RESET=1 ; Hold the FPGA in RESET state.**
ROI_ENABLE		Enables or Disables Region of Interest (ROI) during image capture.
	Enable	0= Disabled 1=Enabled
		Example: **ROI_ENABLE=1 ; Enable Region of Interest Capture Mode.**
ROI_DESCRIPTION		Specifies the Region of Interest (ROI) used during image capture
	ROI	A string containing the comma separated coordinates of the left, top, right, and bottom of the region of interest (ROI).
		Example: **ROI_DESCRIPTION=100,100,200,200) ; use a square ROI 100 pixels by 100 pixels with the top left corner of the ROI square at location (100,100).**
SEQUENCE_IMAGES		Specifies the number of images to acquire during any given capture. These images are acquired at full video rate and are guaranteed to be sequential.
	Number	Number of sequential images to capture
		Example: **SEQUENCE_IMAGES=5 ; Capture 5 consecutive images from the sensor at full video rate.**
DECODE_TYPE		Specifies the decoding type used to process the sensor image data
	Decode Type	STANDARD, CUSTOM, or PLUGIN **Standard decoding** allows the user to specify the bytes per pixel and image buffer type. **Custom decoding** allows the user to select a pre-configured decoding type that has previously configured to map the image sensor data to pixels within a particular image buffer type. **Plugin decoding** uses compiled LabVIEW modules to perform the decoding and is usually preferred over custom decoding as the execution speed is often faster than the more generic mapping methods used in the custom decoding engine.
		Example: **DECODE_TYPE=PLUGIN ; use a plugin decoder**
CLOCKS_PER_PIXEL		Specifies the number of clocks it takes to clock out one image pixel from the image sensor.

COMMAND	PARAMETER	DESCRIPTION

	Clocks/pixel	(1, 2, 3, or 4)
		This value is dependent on the image sensor design. The ISL image buffers have the following pixel bit depths:
		RAW = 32 bits/pixel
		BAYER = 16 bits/pixel
		RGB = 32 bits/pixel
		YCbCr = 16 bits/pixel
		You cannot select clocks/pixel that will result in pixel bit depths that exceed the capabilities of the buffers. If an image sensor only clocks out 8-bits per clock then the clocks/pixel setting could potentially be set to 4 for an RGB or RAW image buffer type. If the image sensor clocks out 8 to 16 bits per clock then the clocks per pixel can only be set to 1 for BAYER or YCbCr image buffer types.

		Example: **CLOCKS_PER_PIXEL=2 ; Map the data from 2** **; clocks to each pixel**
SWAP_BYTES		Perform a swap byte operation on the image sensor data stream.
	Swap Bytes	(TRUE or FALSE) TRUE will swap bytes in the image sensor data stream output. FALSE is equivalent to a “No Operation”
		Example: **SWAP_BYTES=FALSE**
TRANSPOSE		Transpose the 2D Image data before processing and display
	Transpose	(TRUE or FALSE) TRUE will transpose the image data before processing and display. FALSE is equivalent to a “No Operation”
		Example: **TRANSPOSE=FALSE**
CUSTOM_DECODE_TYPE		Specifies the custom decoding type to use for decoding the image sensor pixel data into an ISL image buffer. NOTE: DECODE_TYPE must be set to “CUSTOM” for this parameter to be used.

COMMAND	PARAMETER	DESCRIPTION
	Decode Type	Numeric index of the custom decoder to use. The ISL custom decoders are specified in the “decoding.cfg” file at the ISL application root. This file is maintained by the Image Sensor Lab application software but may also be edited outside of the application software. This text based configuration file specifies the names and data mapping of each defined custom decoding type.
		Example: **CUSTOM_DECODE_TYPE=1**
PLUGIN_DECODE_TYPE		Specifies the plugin decoding type to use for decoding the image sensor pixel data into an ISL image buffer. NOTE: DECODE_TYPE must be set to “PLUGIN” for this parameter to be used.
	Decode Type	Numeric index of the plugin decoder to use. 0 = 1 byte per pixel into TYPE=RAW SENSOR 1 = 2 bytes per pixel into TYPE=RAW SENSOR 2 = every other byte (lower) to TYPE=BAYER 3 = every other byte (upper) to TYPE=BAYER 4 = Micron 10-bit to TYPE=BAYER 5 = Micron RGB565 to TYPE=RGB 6 = Micron RGB555 to TYPE=RGB 7 = Micron RGB444x to TYPE=RGB 8 = Micron RGBx444 to TYPE=RGB 9 = Micron YCbCr to TYPE=YCBCR 10 = IW 12-bit Bayer to TYPE=BAYER
		Example: **PLUGIN_DECODE_TYPE=4 ;use 10-bit Bayer** **;decoding**
IMAGE_ROWS		Sets the number of rows in the image. This value is used when converting the raw sensor data stream into an image.
	Rows	Number of rows to use when reformatting the sensor data stream into an image
		Example: **IMAGE_ROWS=1024; set number of rows to 1024**
IMAGE_COLUMNS		Sets the number of columns in the image. This value is used when converting the raw sensor data stream into an image.
	Columns	Number of columns to use when reformatting the sensor data stream into an image
		Example: **IMAGE_COLUMNS=1280; set columns to 1280**

COMMAND	PARAMETER	DESCRIPTION
BAYER_PATTERN		Specifies which Bayer encoding is used in the Bayer encoded image data.
	Pattern	0, 1, 2, or 3 0 = GBGB::RGRG 1 = GRGR::BGBG 2 = BGBG::GRGR 3 = RGRG::GBGB
		Example: **BAYER_PATTERN=1 ; use GRGR::BGBG**
RAW_FILTER_TYPE		Specifies the display filter used within the ISL application for RAW image buffer type
	Filter Type	0 = RAW SENSOR to 32-bit RGB Image 1 = RAW SENSOR Byte 1 to 8-bit Image 2 = RAW SENSOR Byte 2 to 8-bit Image 3 = RAW SENSOR Byte 3 to 8-bit Image 4 = RAW SENSOR Byte 4 to 8-bit Image 5 = RAW SENSOR Lower 16 bits to 16-bit Image 6 = RAW SENSOR Upper 16 bits to 16-bit Image
		Example: **RAW_FILTER_TYPE=5 ; map low word to 16-bit** **; image**
BAYER_FILTER_TYPE		Specifies the display filter used within the ISL application for BAYER image buffer type.
	Filter Type	0 = BAYER RAW to 16-bit Image 1 = BAYER Red Plane to 16-bit Image 2 = BAYER Green-Red Plane to 16-bit Image 3 = BAYER Green-Blue Plane to 16-bit Image 4 = BAYER Blue Plane to 16-bit Image 5 = BAYER Byte 1 Bayer Interpolation to RGB Image 6 = BAYER Byte 2 Bayer Interpolation to RGB Image 7 = BAYER 10-bit Bayer Interpolation to RGB Image 8 = BAYER 12-bit Bayer Interpolation to RGB Image 9 = BAYER 14-bit Bayer Interpolation to RGB Image
		Example: **BAYER_FILTER_TYPE=7 ; use 10-bit interpolation**
RGB_FILTER_TYPE		Specifies the display filter used within the ISL application for RGB image buffer type.

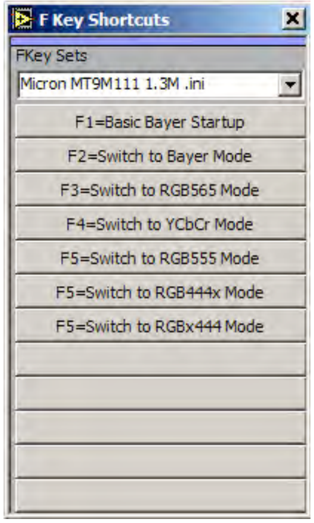
COMMAND	PARAMETER	DESCRIPTION
	Filter Type	0 = RGB RAW to RGB Image 1 = RGB Red Plane to 8-bit Image 2 = RGB Green Plane to 8-bit Image 3 = RGB Blue Plane to 8-bit Image 4 = RGB RAW Hue to 8-bit Image 5 = RGB RAW Saturation to 8-bit Image 6 = RGB RAW Luminance to 8-bit Image 7 = RGB RAW Value to 8-bit Image 8 = RGB RAW Intensity to 8-bit Image
		Example: **RGB_FILTER_TYPE=0 ; use full RGB image**
YCBCR_FILTER_TYPE		Specifies the display filter used within the ISL application for YCbCr image buffer type.
	Filter Type	0 = YCbCr RAW to 16-bit Image 1 = YCbCr Byte 1 to 8-bit Image 2 = YCbCr Byte 2 to 8-bit Image 3 = YCbCr Y Plane to 16-bit Image 4 = YCbCr Cb Plane to 16-bit Image 5 = YCbCr Cr Plane to 16-bit Image 6 = RGB Interpolation to RGB Image
		Example: **YCBCR_FILTER_TYPE=3 ; use luminance plane**
COUNTER_CONFIG		Configures the ISL on-board timer/counter to measure the transitions of the specified “event parameter” over the period specified in the “gate select”
	Gate Select	0 = Frame Window 1 = 1-second Timer
	Event Parameter	0 = Data[0] lsb image sensor data bit. 1 = Data[1] 2 = Data[2] 3 = Data[3] 4 = Data[4] 5 = Data[5] 6 = Data[6] 7 = Data[7] 8 = Data[8] 9 = Data[9] 10 = Data[10] 11 = Data[11] 12 = Data[12] 13 = Data[13] 14 = Data[14] 15 = Data[15] 16 = VSYNC 17 = XSYNC0 18 = XSYNC1 19 = XSYNC2 20 = XSYNC3 21 = DIN0 22 = DIN1 23 = DIN2 24 = PIXCLK 25 = REFCLK

COMMAND	PARAMETER	DESCRIPTION
		Example: **COUNTER_CONFIG=0,17 ; measure Hsync/Vsync**
COUNTER_ARM		Arms the ISL on-board timer/counter. NOTE: There are no parameters for this command.
		Example: **COUNTER_ARM ; Arm the event counter**
CAPTURE_CONFIG		Configures the image capture parameters of the ISL hardware.
	PixelDataSelect	0 = 16-Bit Word (bits 15:0) 1 = Low Byte (bits 7:0) 2 = High Byte (bits 15:8) 3 = 16-Bit Continuous Count
	FrameStart	0 = VSYNC rising edge 1 = VSYNC falling edge
	FrameEnd	0 = VSYNC rising edge 1 = VSYNC falling edge
	XSYNC0	0 = disabled 1 = enabled active high 2 = enabled active low
	XSYNC0	0 = disabled 1 = enabled active high 2 = enabled active low
	XSYNC1	0 = disabled 1 = enabled active high 2 = enabled active low
	XSYNC2	0 = disabled 1 = enabled active high 2 = enabled active low
	ClockSelect	0 = PIXCLK 1 = REFCLK
	ClockEdge	0 = rising edge 1 = falling edge

COMMAND	PARAMETER	DESCRIPTION
		Example: **CAPTURE_CONFIG=1,0,1,1,0,0,0,1** **; ISL capture hardware Low byte only** **; Frame Start at Rising Edge** **; Frame End at Falling Edge** **; Xsync0 enable and active high** **; Xsync1 disabled** **; Xsync2 disabled** **; Clock Select = Pixel Clock** **; Clock Edge = Use falling edge of Pixel Clock**
SNAP		Signals the Image Sensor Lab application software to snap an image. NOTE: This command does not actually cause the ISL hardware to acquire an image. It is just a simple message or “application” command used to control the application software.
		Example: **Snap ; signal the image Sensor Lab Application Software to snap an image.**
CAPTURE_CONTINUOUS		Signals the Image Sensor Lab application software to continuously acquire, process, and display images from the image sensor. NOTE: This command does not actually cause the ISL hardware to acquire images. It is just a simple message or “application” command used to control the application software.
	Enabled	0 = Continuous capture OFF 1 = Continuous capture ON
		Example: **CONTINUOUS_CAPTURE=1 ; enable continuous**

6.2.7.1 F Key Shortcuts

The **F Key Shortcuts** panel contains a bank of 12 buttons that are mapped to the twelve F Keys on most computer keyboards. The drop-list at the top of this panel displays the "F Key Sets" that are currently defined. Once an F Key Set is selected from the drop-list the button labels and functions are updated.


F Key Shortcuts

FKey Sets: Micron MT9M111 1.3M .ini

F Key Sets are defined in simple text files located in the Image Sensor Lab\FKeys directory. User can edit existing F Key Set files or create new ones following the format of the included files. The configuration files simply list a name and sequence file path for each F Key. An example .ini file entry is shown below:

```
[ISLFKEY]
[SetName]
SetName="Micron MT9V112 VGA"
[F1]
Name="F1=Basic Bayer Startup"
Path="C:\p4\TIMS\Hardware Products\ImageSensorLab\Applications\sequence files\Micron
MT9M111\Micron_MT9M111_Startup.txt"
```

6.2.8 Image Analysis Menu

6.2.8.1 Image Pixel Data

This image analysis panel displays the raw image sensor data and includes a histogram graph of both the low and high byte of the data as well as a histogram for the full 16-bit data.

On the right side of this panel, bit count histograms display a different view of the raw data and can be used to detect stuck high, low, or sensor data lines that are stuck to other sensor data lines.

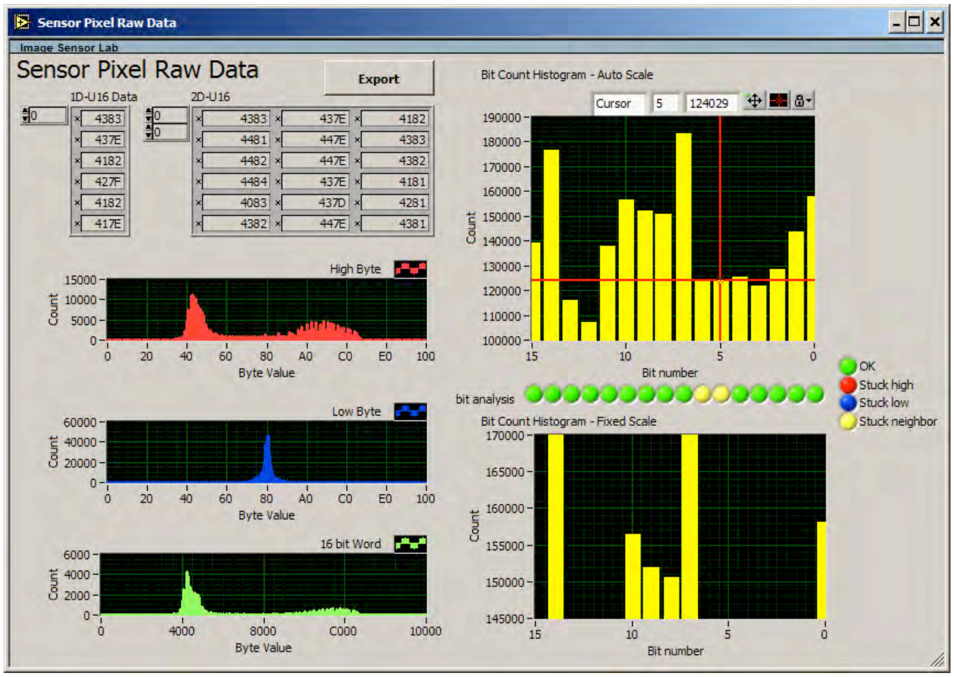

Sensor Pixel Raw Data

Image Sensor Lab

Sensor Pixel Raw Data

1D-U16 Data (index 0)

4383
437E
4182
427F
4182
417E

2D-U16 (index 0, 0)

4383	437E	4182
4481	447E	4383
4482	447E	4382
4484	437E	4181
4083	437D	4281
4382	447E	4381

Cursor 5 124029