# Sandia Project Document
## Version 3.0


| | | | |
|---|---|---|---|
| Author: | John Frye, Max Bobrov | Date: | July 13, 2010 |
| Reviewer: | Don Figer | Date: | July 13, 2010 |

## 1.0 Introduction

This document describes the objectives of the Sandia image acquisition project as well as the methodologies used in achieving them. The goal of the project is to enable multi-interface image acquisition from a variety of FLIR cameras. The acquired video data is to be captured and displayed over a variety of interfaces. A commercial off-the-shelf (COTS) FPGA evaluation board has been selected as the development tool for this purpose.

## 2.0 Project Statement of Work

The objective of the Sandia image acquisition project is to produce a hardware system for the purpose of capturing and displaying video rate data from specific infrared cameras over multiple interfaces. Custom data acquisition software for the purpose of image display and capture on the PC is also required. A system-level block diagram is presented in Figure 1.

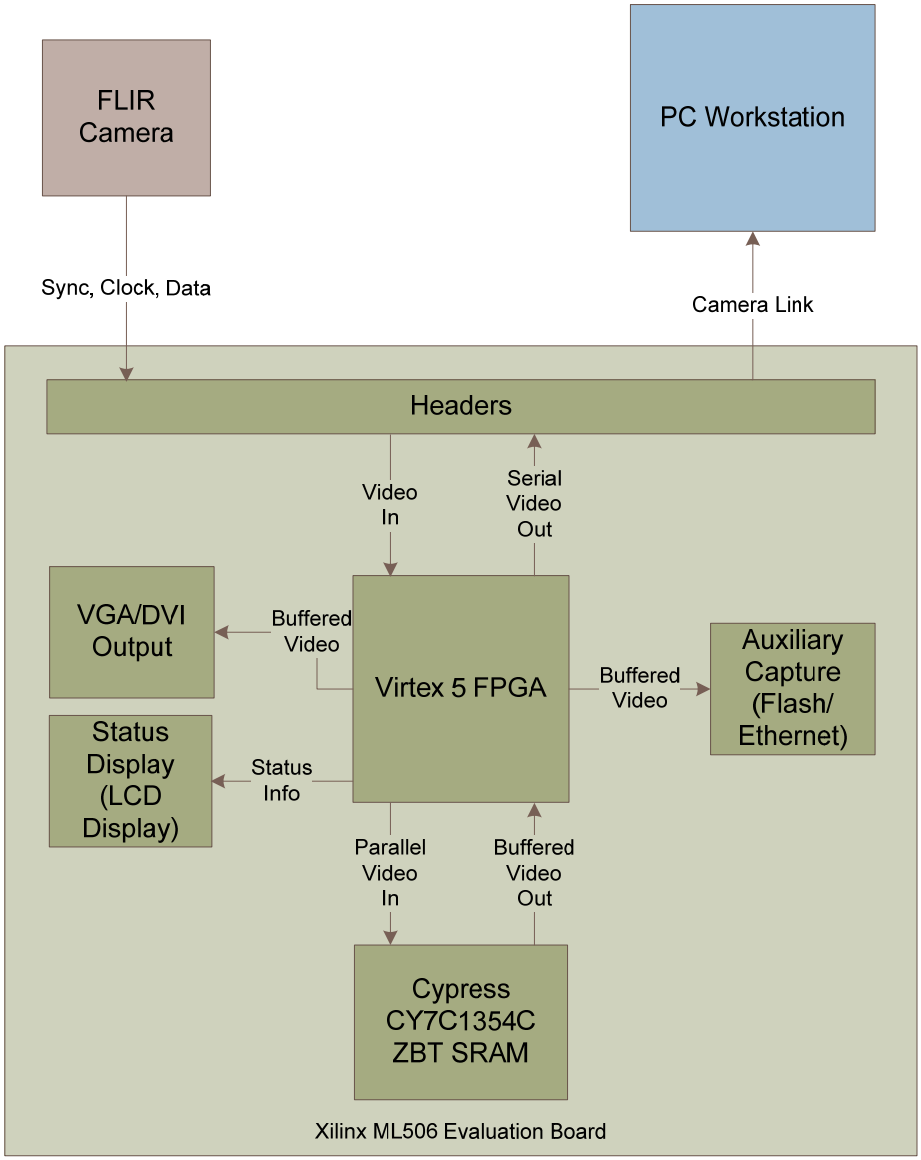



*Figure 1: System Block Diagram*

For development purposes, the Xilinx ML506 and Xilinx ML402 evaluation boards have been selected to operate with the Indigo Photon 320, Indigo Photon 640, and DRS U6000 cameras in order to prototype the desired system. The Indigo Photon 320 camera is shown in Figure 2 and Figure 3, the Indigo Photon 640 is shown in Figure 4 and Figure 5, the DRS U6000 is shown in Figure 6 and Figure 7, and the evaluation boards are presented in Figure 8 and Figure 9. Each of the above cameras transfers data to the evaluation board over multiple LVDS pairs, which physically connect to differential headers on the board. After configuration, the FPGA present on

the evaluation board buffers the incoming data into an onboard ZBT SRAM chip and outputs it over multiple interfaces including Camera Link, VGA, DVI, and Ethernet. Status information is output to the integrated LCD display. A detailed listing of supported functionality in different configurations is presented in Table 1.

| | Evaluation Board | | | | |
|---|---|---|---|---|---|
| | ML506 | | | ML402 | |
| **Feature** | DRS U6000 | Photon 640 | Photon 320 | Photon 640 | Photon 320 |
| Camera Link | ✓ | ✓ | ✓ | ✓ | ✓ |
| Ethernet | ✓ | ✓ | ✓ | | |
| VGA | | | | ✓ | ✓ |
| DVI | ✓ | ✓ | ✓ | | |
| LCD Status Display | ✓ | ✓ | ✓ | ✓ | ✓ |
| Camera Selectable via DIP-Switch | ✓ | ✓ | ✓ | ✓ | ✓ |
| Camera Selectable via Software | ✓ | ✓ | ✓ | ✓ | ✓ |

*Table 1: Features Supported in Different Board/Camera Configurations*

Camera selection for input to the Xilinx evaluation boards can occur physically via DIP switches or through software via RS-232 communication. In addition, the status information is displayed on the integrated two-line LCD screen. The board reports whether or not the selected camera is detected.

*Figure 2: Indigo Photon 320 Infrared Camera*

*Figure 3: Indigo Photon 320 Infrared Camera Output Port*


*Figure 4: Indigo Photon 640 Infrared Camera*

*Figure 5: Indigo Photon 640 Infrared Camera Output Port*

*Figure 6: DRS U6000 Infrared Camera*


*Figure 7: DRS U6000 Infrared Camera DB-25 Output Port and Power/RS-232 Connector*

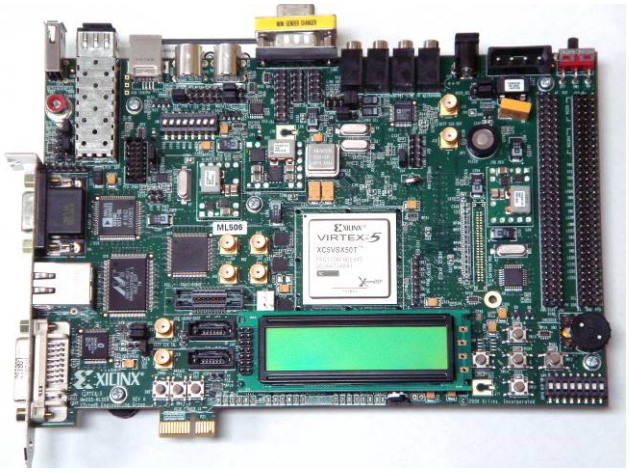



*Figure 8: Xilinx ML506 Virtex 5 Evaluation Board*

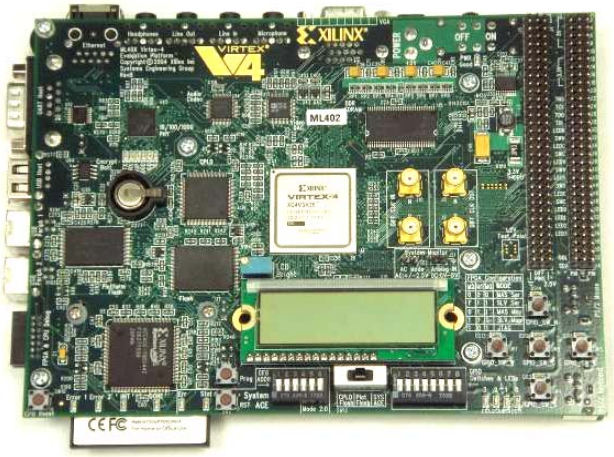



*Figure 9: Xilinx ML402 Virtex 4 Evaluation Board*

In order to acquire video data on a PC via Camera Link, a Camera Link-compatible frame grabber must be installed. Two frame grabbers were selected for development: an Imperx FrameLink Express frame grabber, and a National Instruments PCIe-1427 image acquisition card. These are presented in Figure 10 and Figure 11, respectively.


*Figure 10: Imperx FrameLink Express Camera Link Frame Grabber*

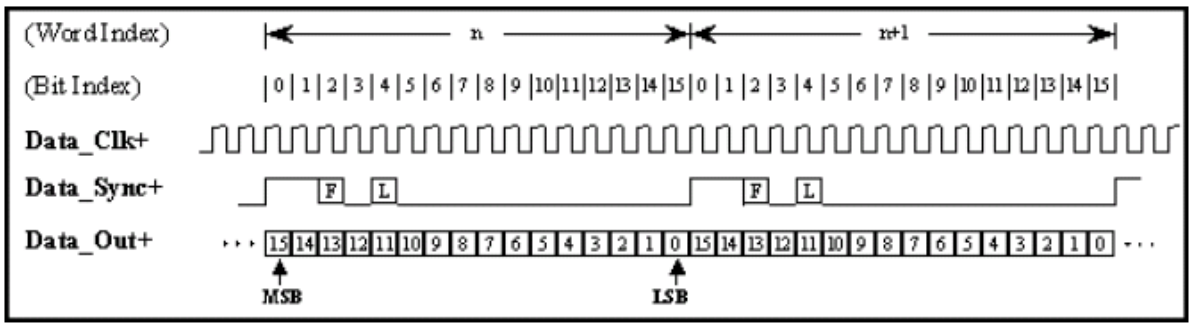
*Figure 11: National Instruments PCIe-1427 Image Acquisition Card*

## 3.0 Camera-FPGA Interface

The camera devices utilized in this project output digitized pixel values over several LVDS differential pairs. Three cameras, namely the Indigo Photon 320, Indigo Photon 640, and the DRS U6000, have been selected for prototyping of the interface. These devices output digitized pixel data on one or more differential pairs along with clock and synchronization bits. By creating VHDL processes with sensitivities to these lines, it is possible to devise state machines which can parse, parallelize, and store the digitized image data to a data structure onboard the FPGA. In the current implementation, the digital video data is buffered in the onboard ZBT-compatible SRAM.

## 3.1 Indigo Photon 320 Interface

The Indigo Photon 320 outputs digital video data on a single LVDS pair at a clock rate of 73.636 MHz. Each frame transmitted by the Indigo Photon 320 is 324x256x14 bits per pixel (bpp), and the video rate is 29.97 frames per second. A timing diagram for the digital output of the Indigo Photon 320 camera is presented in Figure 12, while the state machine diagram for de-serializing the data is shown in Figure 13. Each valid pixel is read into a frame buffer for later display or transmission.




**F = frame sync; logic high on the word starting the frame, logic low otherwise**
**L = line sync; logic high during valid pixel data, logic low otherwise**

*Figure 12: Indigo Photon 320 Digital Data Timing Diagram*

Note: The Indigo Photon 320 camera allows for selection between 8-bit and 14-bit pixel resolutions. While the 8-bit mode is processed by the camera's automatic gain control (AGC) algorithm, the 14-bit mode output is not. Consequently, visual display of the 14-bit data will exhibit good contrast unless it has undergone scaling of some kind.

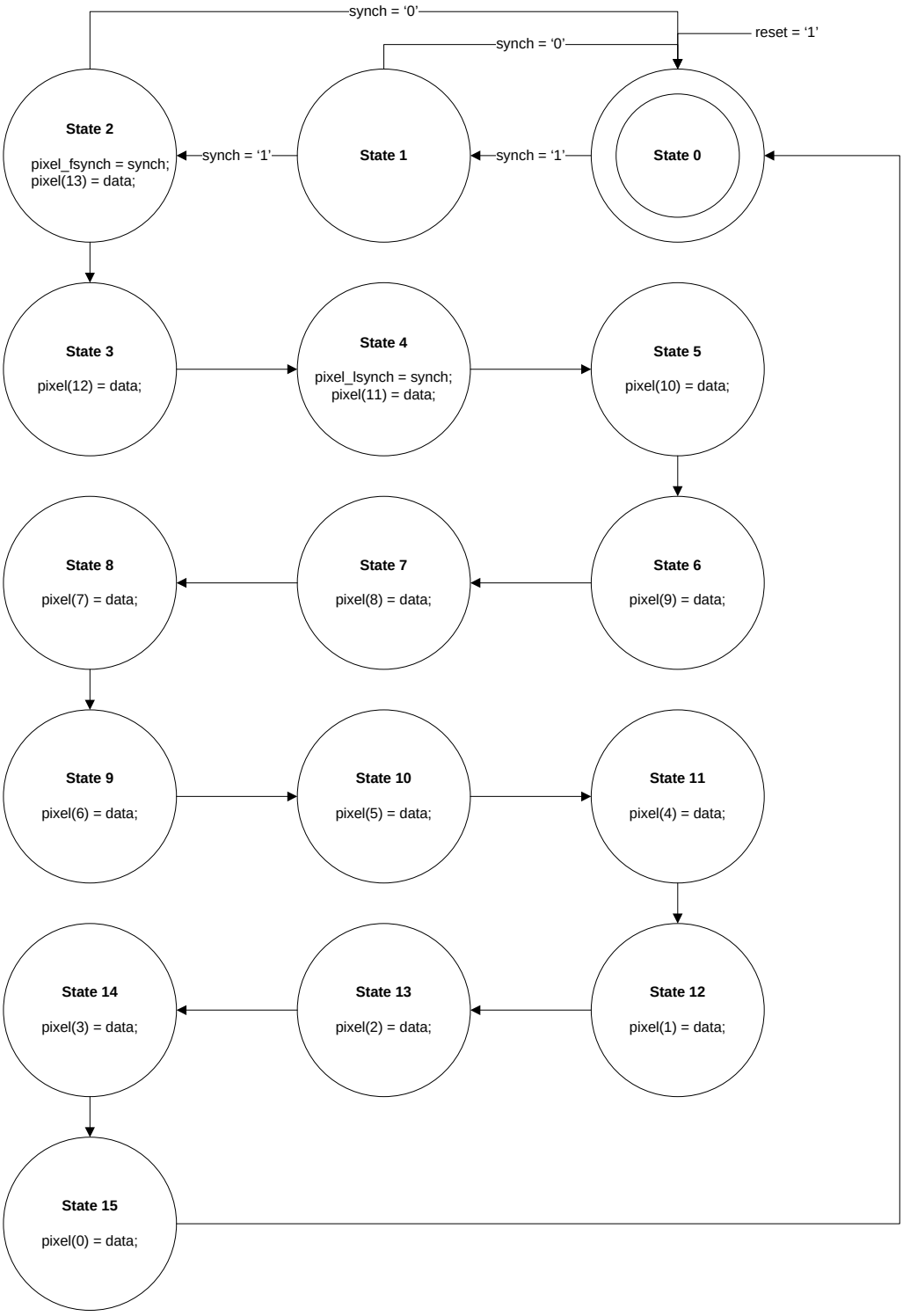
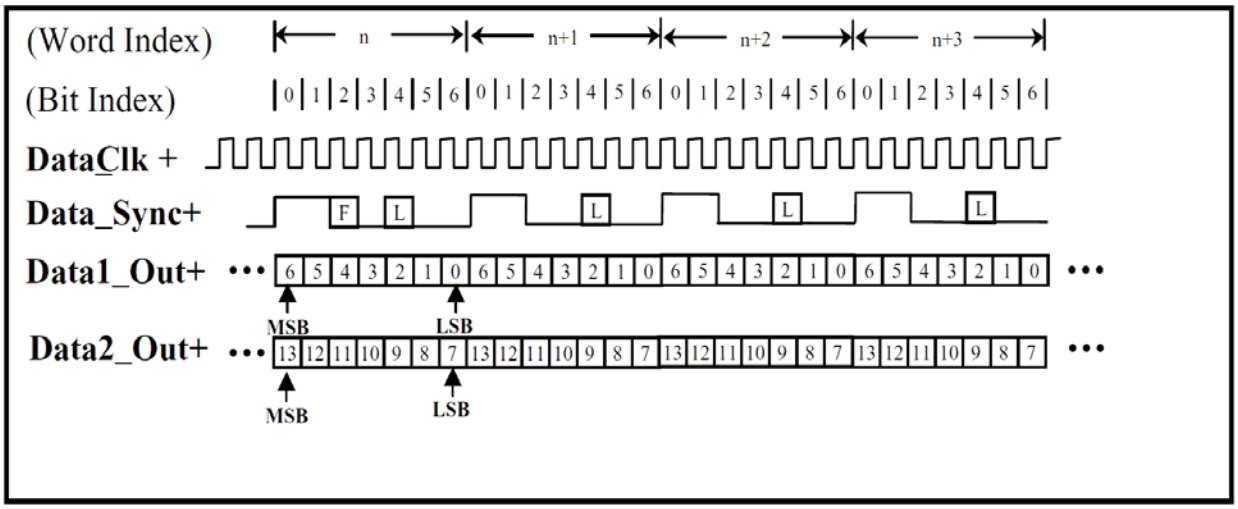



*Figure 13: Indigo Photon 320 Image Acquisition State Diagram*

## 3.2 Indigo Photon 640 Interface

The Indigo Photon 640 outputs digital video data over two LVDS data pairs at a clock rate of 73.636 MHz. Each frame transmitted by the Indigo Photon 640 is 644x512x14 bpp, and the video rate is 29.97 frames per second. A timing diagram for the digital output of the Indigo Photon 640 camera is presented in Figure 14, while the state machine diagram for de-serializing the data is shown in Figure 15. Each valid pixel is read into a frame buffer for later display or transmission.




**F = frame sync; logic high on the word starting the frame, logic low otherwise**
**L = line sync; logic high during valid pixel data, logic low otherwise**

*Figure 14: Indigo Photon 640 Digital Data Timing Diagram*

Note: The Indigo Photon 640 camera allows for selection between 8-bit and 14-bit pixel resolutions. While the 8-bit mode is processed by the camera's automatic gain control (AGC) algorithm, the 14-bit mode output is not. Consequently, visual display of the 14-bit data will exhibit good contrast unless it has undergone scaling of some kind.

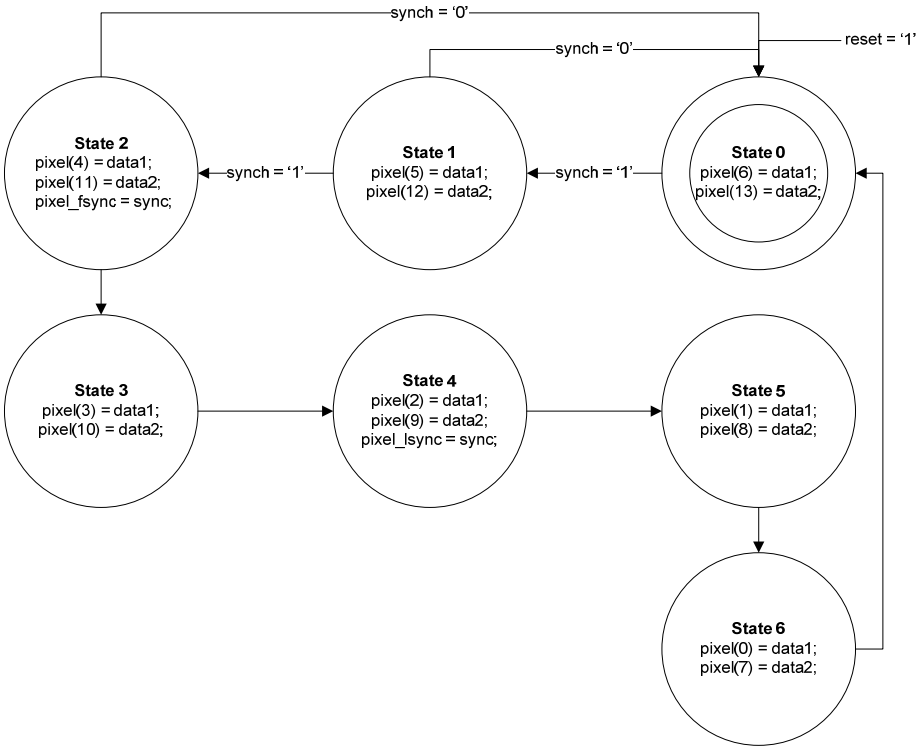



*Figure 15: Indigo Photon 640 Image Acquisition State Diagram*

## 3.3 DRS U6000 Interface

The DRS U6000 transmits 640x480x14 bpp interlaced video at a rate of approximately 30 fps. Digital video data is output over eight LVDS pairs. In addition, one pair transmits a 25 MHz clock, another transmits a data valid signal, and third transmits a frame synch signal for a total of eleven LVDS pairs. The camera output is composed of control and data words, with one byte transmitted per clock. Control bytes can be distinguished from data word bytes by checking the value of the data valid signal; the (active low) data valid signal is '1' for control characters, and '0' for data values. The data format is detailed in Figure 16 below.

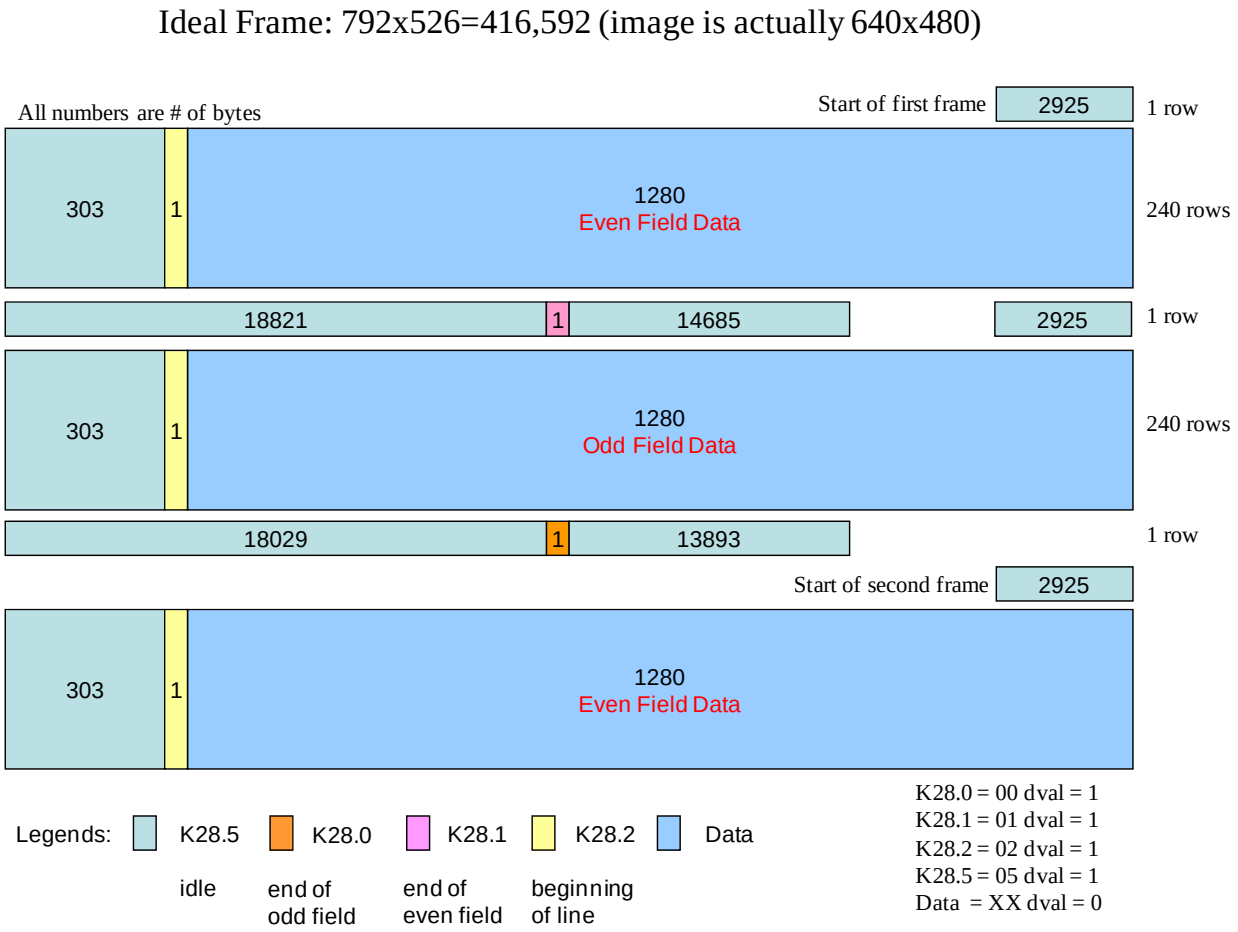



*Figure 16: DRS U6000 Output Format*

Although 14-bit video data is output from the DRS U6000 camera, only eight-bits can be transmitted over the data channels per clock. Consequently, two bytes must be shifted into a register with the lower 14-bits being the pixel value. Consequently, the pixel clock is half of the camera clock, or 12.5 MHz. The VHDL DRS U6000 interface module is responsible for de-interlacing the video for progressive scan output. This is accomplished by buffering the first field and subsequently alternating output of even and odd lines, updating the FIFO read address as necessary. The FIFO buffer clock operates at 25 MHz, although two bytes are read or written every clock; this makes for an effect clock speed of 2x the pixel clock. The FIFO state machine is depicted in Figure 17. It is important to note that the phase relationship between the camera clock and the pixel clock must remain fixed in order to maintain proper functionality.

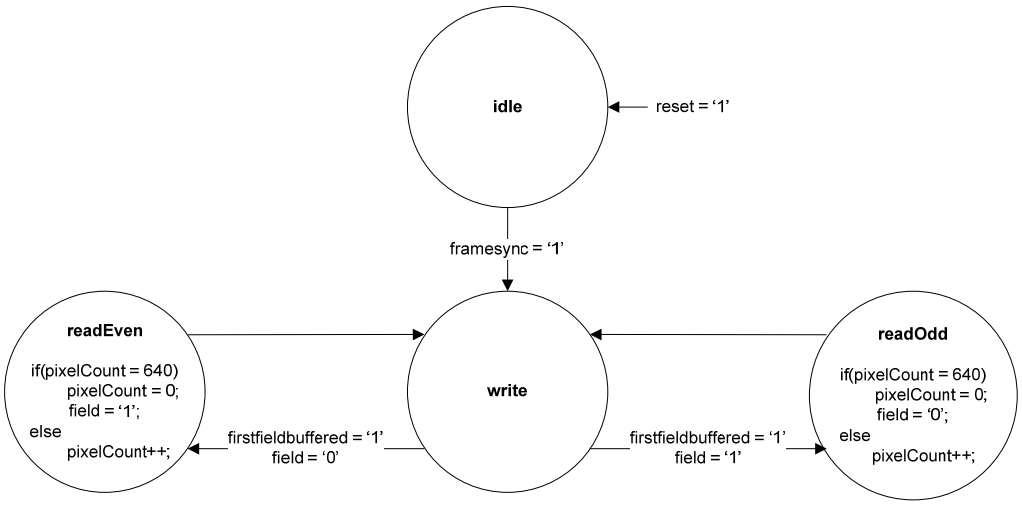



*Figure 17: FIFO Buffer State Diagram*

## 4.0 FPGA-Camera Link Interface

As per the project requirements, the digital LVDS data output by the cameras is to be reformatted for image acquisition over the Camera Link interface. The official AIA Camera Link standard was obtained in order to facilitate development of the required digital system. The Camera Link module serializes the buffered pixel data in a 7:1 ratio for transfer over 4 LVDS data pairs with reference to a clock seven times faster than the pixel clock, as shown in Figure 18. A Xilinx digital clock manager (DCM) is used to obtain the multiplied clocks. Table 2 provides a list of pixel and Camera Link clocks for each of the specified cameras.

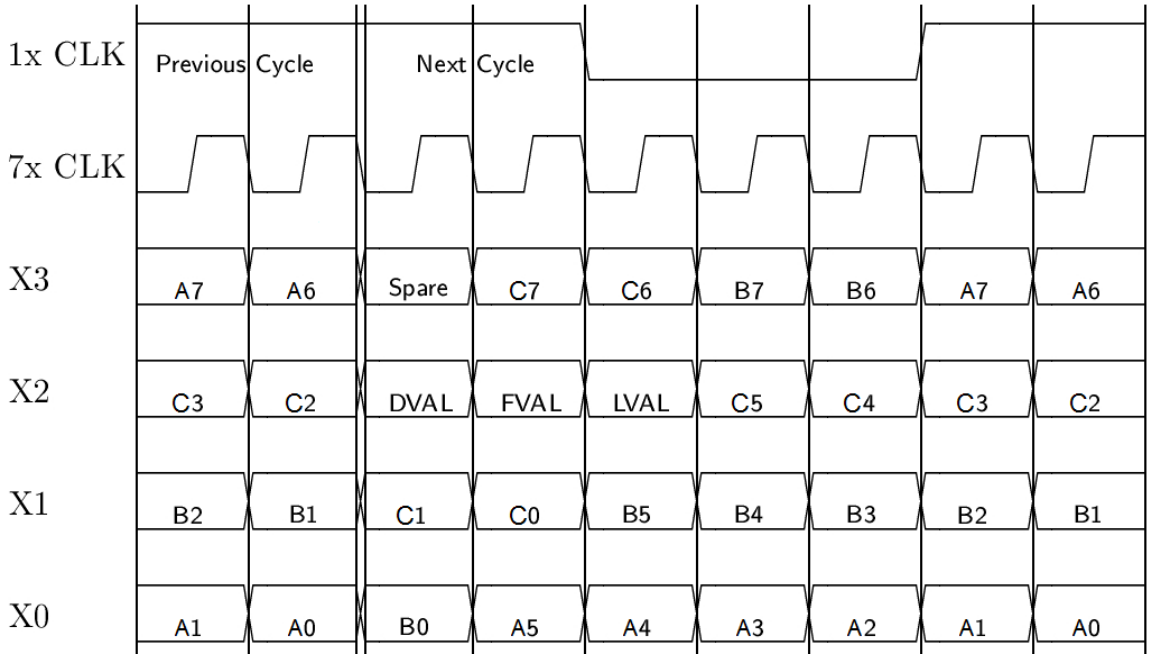



*Figure 18: Camera Link Serialization and Timing Diagram*

| Camera | Data Clock (MHz) | Clocks Per Pixel | Pixel Clock (MHz) | Camera Link Clock (MHz) |
|---|---|---|---|---|
| Indigo Photon 320 | 73.636 | 16 | 4.60225 | 32.21575 |
| Indigo Photon 640 | 73.636 | 7 | 10.51942857 | 73.636 |
| DRS U6000 | 25 | 4 | 6.25 | 43.75 |
| BAE MIM500X | TBD | TBD | TBD | TBD |

*Table 2: Camera Timing*

The base-configuration Camera Link standard defines three 8-bit “ports” (A, B, C in Figure 18) to be transferred over the four LVDS pairs; the pixels to be transmitted are mapped to these ports depending on the number of bits per pixel and the number of taps. For example, for single output 14-bit mode, the lower 8 bits of the pixel are mapped to A7-A0, while the upper 6 bits are mapped to B5-B0. A complete listing of the supported bit modes as defined in the specification are shown in Table 3.

| Base Configuration | | | | | | |
|---|---|---|---|---|---|---|
| **Port/bit** | **8-bit x 1~3*** | **10-bit x 1~2** | **12-bit x 1~2** | **14-bit x 1** | **16-bit x 1** | **24-bit RGB** |
| Port A0 | A0 | A0 | A0 | A0 | A0 | R0 |
| Port A1 | A1 | A1 | A1 | A1 | A1 | R1 |
| Port A2 | A2 | A2 | A2 | A2 | A2 | R2 |
| Port A3 | A3 | A3 | A3 | A3 | A3 | R3 |
| Port A4 | A4 | A4 | A4 | A4 | A4 | R4 |
| Port A5 | A5 | A5 | A5 | A5 | A5 | R5 |
| Port A6 | A6 | A6 | A6 | A6 | A6 | R6 |
| Port A7 | A7 | A7 | A7 | A7 | A7 | R7 |
| Port B0 | B0 | A8 | A8 | A8 | A8 | G0 |
| Port B1 | B1 | A9 | A9 | A9 | A9 | G1 |
| Port B2 | B2 | nc | A10 | A10 | A10 | G2 |
| Port B3 | B3 | nc | A11 | A11 | A11 | G3 |
| Port B4 | B4 | B8 | B8 | A12 | A12 | G4 |
| Port B5 | B5 | B9 | B9 | A13 | A13 | G5 |
| Port B6 | B6 | nc | B10 | nc | A14 | G6 |
| Port B7 | B7 | nc | B11 | nc | A15 | G7 |
| Port C0 | C0 | B0 | B0 | nc | nc | B0 |
| Port C1 | C1 | B1 | B1 | nc | nc | B1 |
| Port C2 | C2 | B2 | B2 | nc | nc | B2 |
| Port C3 | C3 | B3 | B3 | nc | nc | B3 |
| Port C4 | C4 | B4 | B4 | nc | nc | B4 |
| Port C5 | C5 | B5 | B5 | nc | nc | B5 |
| Port C6 | C6 | B6 | B6 | nc | nc | B6 |
| Port C7 | C7 | B7 | B7 | nc | nc | B7 |
| * If only using a single channel, use Port A. If using two channels, use Port A and B. | | | | | | |

*Table 3: Camera Link Bit/Port Mapping*

The Camera Link portion of the code has a separate buffer in which all pixels (even invalid pixels) and their corresponding synchronization values are stored. The synchronization value for each pixel is retransmitted with the pixel data in order to satisfy Camera Link requirements. The state diagram for the Camera Link re-serialization component is presented in Figure 19.

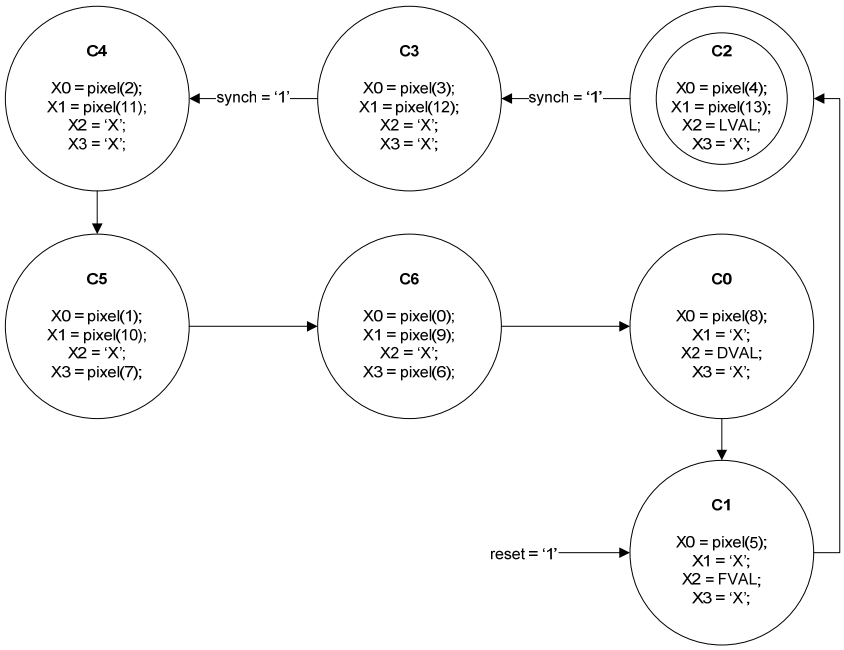



*Figure 19: Camera Link Serialization State Machine. Note that 'X' denotes a "don't care" condition.*

The data is acquired on a PC via an installed Camera Link-compatible frame grabber. For the purposes of this project, we have obtained and installed an Imperx FrameLink Express frame grabber and a National Instruments PCIe-1427 image acquisition card.

## 5.0 FPGA-VGA Interface

The FPGA development board is used to output camera data to a standard VGA port as it is being collected. Currently, VGA output has only been implemented on the ML402 prototype board since the ML506 board does not possess a VGA output port. In the system design, valid pixels read from the ZBT SRAM are input to VGA frame buffer, which is sized dependent on the selected camera. As the video data is monochromatic, the lower eight bits of each pixel is replicated on three 8-bit output lines (RGB) which are readout in accordance with standard VGA timing. Currently, the image is output at a resolution of 644x512 with a 60 Hz vertical refresh rate. For cameras which have a resolution smaller than this, the image is drawn in a subsection of this active region; in the case of the Indigo Photon 320, the video image is drawn in the upper-left quadrant of the display. The ML402 board supports only the Photon 320 and Photon 640 cameras, and the DRS camera output cannot be displayed via VGA. A VGA timing diagram is presented in Figure 20.

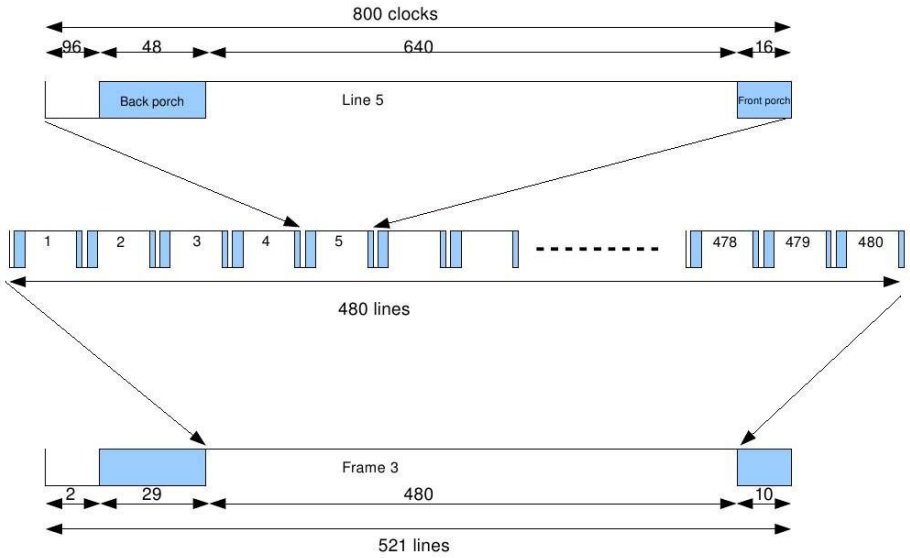
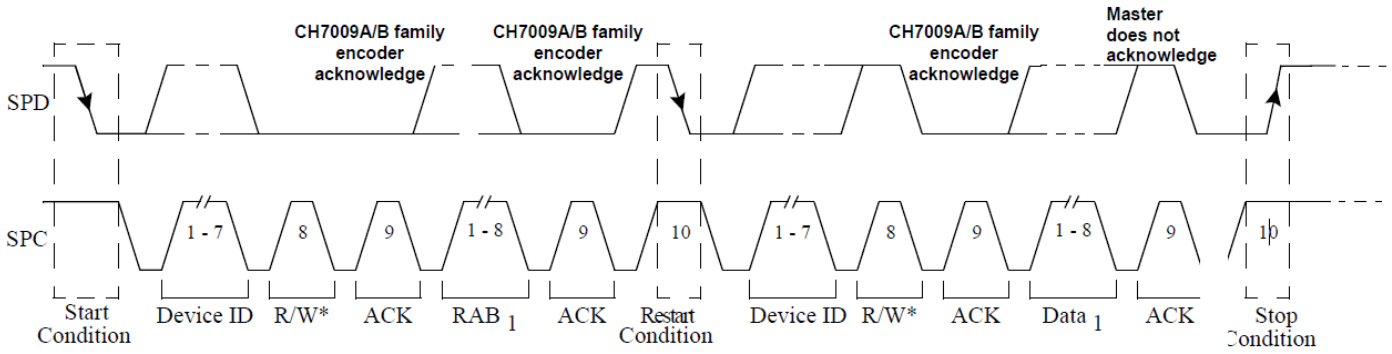



*Figure 20: VGA Timing Diagram*

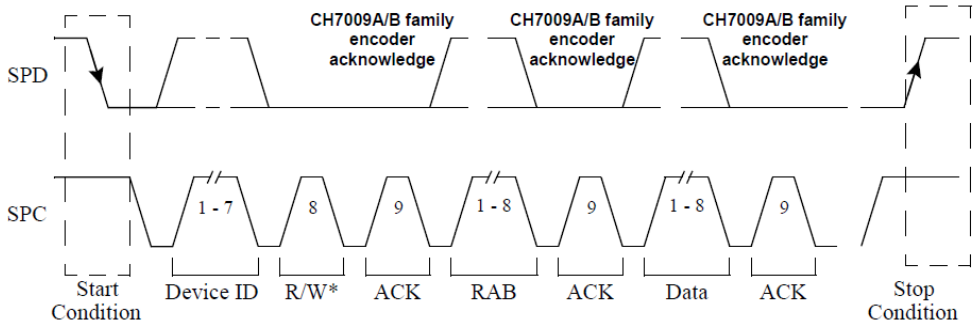
## 6.0 FPGA-DVI Interface

DVI signaling on the ML506 prototype board is generated using the Chrontel CH7301C DVI transmitter device. The chip is configured via $I^2C$ (two-wire interface) communication and accepts video signals in VGA format as input. Consequently, the VGA and frame-buffer modules are re-used within the DVI VHDL module. The "I2C Controller Core" available on OpenCores.org was used for the purpose of register configuration. Details of the $I^2C$ protocol are presented in the Chrontel CH7301C documentation; read and write cycle diagrams are present in Figure 21 and Figure 22 below.




**Note:** The acknowledge is from the CH7009A/B family encoder (slave).

*Figure 21: I2C Single Cycle Read Operation*




**Note:** The acknowledge is from the CH7009A/B family encoder (slave).

*Figure 22: I2C Single Cycle Write Operation*

The registers onboard the CH7301C chip must be configured for successful operation. An exhaustive listing of the available register settings is available in the Chrontel documentation. Table 4 lists the registers that the DVI module sets as well as their corresponding values.

| Register | Address (hex) | Value (hex) |
|---|---|---|
| Clock Mode Register | 0x1C | 0x04 |
| Input Clock Register | 0x1D | 0x43 |
| DAC Control Register | 0x21 | 0x09 |
| DVI PLL Supply Control Register (TPVT) | 0x35 | 0x30 |
| DVI PLL Filter Register (TPF) | 0x36 | 0x60 |
| Power Management Register | 0x49 | 0xC0 |

*Table 4: Chrontel 7301C Register Settings*

Currently, the image is output at a resolution of 644x512 with a 60 Hz vertical refresh rate. For cameras which have a resolution smaller than this such as the Indigo Photon 320 and the DRS U6000, the image is drawn in a subsection of this active region. Depending on monitor settings, this subsection could either be displayed in native resolution or expanded to display across the whole screen.

## 7.0 FGPA-Ethernet Interface

As an alternative method for viewing and capturing video, an Ethernet interface was designed. This interface consists of an Ethernet MAC controller communicating with a soft-core processor running a TCP/IP stack. The MicroBlaze soft processor core was selected to simplify the design. It provides a simple software interface to the on-board hardware MAC controller and has a built in support for the Lightweight TCP/IP software (LWIP). The Xilinx Platform Suite (XPS) was used to design the hardware and software.

The MicroBlaze processor was configured with an interface to the DDR2 RAM, xps_lltemac, 64KB of instruction memory, 64KB of data memory and a timer which was required for LWIP to function properly. Additionally, two Fast Simplex Link (FSL) busses were configured to facilitate communication between the MicroBlaze and the rest of the xicam vhdl.

Software, written in C, communicates with the xps_ll_temac MAC controller wrapper and is executed by the MicroBlaze processor. LWIP was enabled and communicates over the UDP protocol; lwip4 must be added to the linker options of each project requiring the use of LWIP. The MicroBlaze receives buffered video data from the FIFO over the inbound FSL bus, which is subsequently encapsulated with the proper TCP/IP protocol headers and transmitted using Ethernet to a PC workstation.

The PC workstation captures this data using a custom adaptor to the MATLAB Image Acquisition toolbox; the IP address or host name of the FPGA must be provided as well as the digital output mode for the photon cameras. Before Matlab can properly access he adaptor, it must be registered using the imaqregister command. Once this is complete, the adaptor, called XicamEthernetimaq, can be accessed from the software GU in the “Acquisition Adaptor” drop down menu.

## 8.0 LCD Status Display

The onboard LCD display is used to relay system status information. Currently, the status information consists of whether or not the selected camera has been connected as well as the FPS of the incoming video signal. The ML506 and ML402 LCD displays utilize a 4-bit data interface; however, all characters and commands consist of 8 bits. As a result, data must be sent in two phases: the upper 4 bits first, followed by the lower 4-bits.

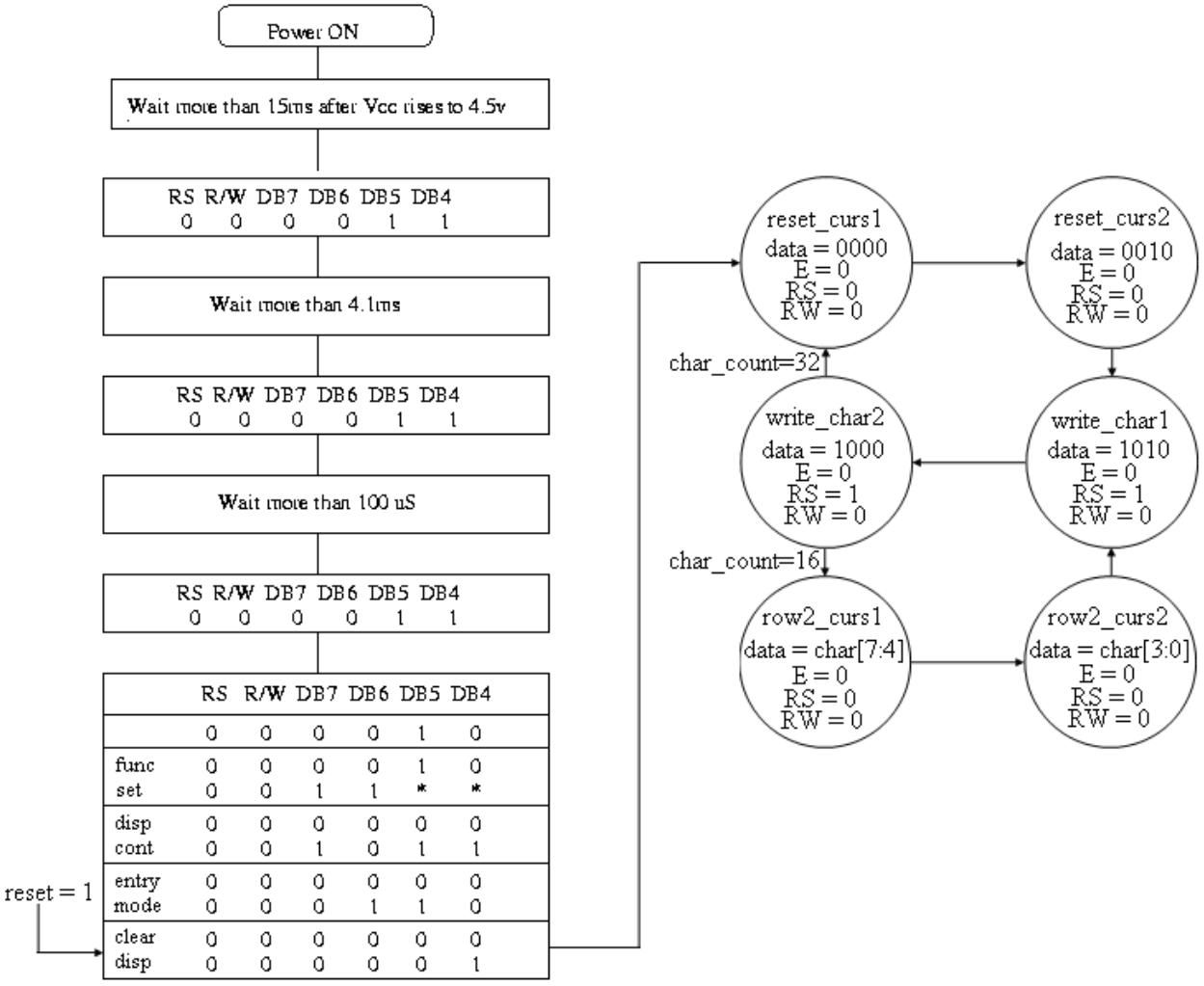



*Figure 23: LCD State Diagram*

This process consists of five major steps. First, a start up sequence is sent to initialize the LCD and inform it to function in 4-bit mode. Next, the LCD is programmed to function in 4-bit mode with two display lines and 5x11 dot font type. Third, the display is turned on with the cursor deactivated. Next, the display is set to function in left-to-right mode. Finally, the display is cleared, completing the initialization process and making it ready to display characters. The actual writing of characters consists of three major steps. First, the cursor is reset to the start of line 1. Next, the FPGA begins to output the first line of characters. When the first line is complete (*i.e.*, 16 characters are written to the LCD), the LCD must move the cursor to the start of the second row. It then continues to write an additional 16 characters, completing the second line. At this point, the cursor is reset again and process starts over. This sequence is depicted in

the state machine available in Figure 23. Table 5 details the instruction codes sent to the LCD display during each state.

| Instruction | Instruction Code | | | | | | | | | | Description | Execution time (fosc= 270 kHz) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | RS | R/W | DB7 | DB6 | DB5 | DB4 | DB3 | DB2 | DB1 | DB0 | | |
| Clear Display | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | Write "20H" to DDRAM and set DDRAM address to "00H" from AC | 1.53 ms |
| Return Home | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | - | Set DDRAM address to "00H" from AC and return cursor to its original position if shifted. The contents of DDRAM are not changed. | 1.53 ms |
| Entry Mode Set | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | I/D | SH | Assign cursor moving direction and enable the shift of entire display. | 39 µs |
| Display ON/ OFF Control | 0 | 0 | 0 | 0 | 0 | 0 | 1 | D | C | B | Set display(D), cursor(C), and blinking of cursor(B) on/off control bit. | 39 µs |
| Cursor or Display Shift | 0 | 0 | 0 | 0 | 0 | 1 | S/C | R/L | - | - | Set cursor moving and display shift control bit, and the direction, without changing of DDRAM data. | 39 µs |
| Function Set | 0 | 0 | 0 | 0 | 1 | DL | N | F | - | - | Set interface data length (DL: 8-bit/4-bit), numbers of display line (N: 2-line/1-line) and, display font type (F:5×11dots/5×8 dots) | 39 µs |
| Set CGRAM Address | 0 | 0 | 0 | 1 | AC5 | AC4 | AC3 | AC2 | AC1 | AC0 | Set CGRAM address in address counter. | 39 µs |
| Set DDRAM Address | 0 | 0 | 1 | AC6 | AC5 | AC4 | AC3 | AC2 | AC1 | AC0 | Set DDRAM address in address counter. | 39 µs |
| Read Busy Flag and Address | 0 | 1 | BF | AC6 | AC5 | AC4 | AC3 | AC2 | AC1 | AC0 | Whether during internal operation or not can be known by reading BF. The contents of address counter can also be read. | 0 µs |
| Write Data to RAM | 1 | 0 | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | Write data into internal RAM (DDRAM/CGRAM). | 43 µs |
| Read Data from RAM | 1 | 1 | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | Read data from internal RAM (DDRAM/CGRAM). | 43 µs |

* "-": don't care

*Table 5: LCD Instruction Codes*

The FPS calculation is accomplished by counting the number of elapsed clock cycles while waiting for 20 frame sync signals. This value is than converted to a decimal FPS value. If the FPS is determined to be zero, the LCD displays "camera disconnected". Otherwise, the actual FPS value is displayed.

## 9.0 Camera Selection

The system must be configured for a specific camera in order to function; this can be done by way of software, via RS-232 communication, or by selecting the setting the GPIO DIP switches appropriately on the evaluation board itself. The user GPIO DIP switches are located by the bottom right-hand corner on both the ML506 and ML402 boards. For the purposes of this document, these DIP switches are addressed 1-8 from left to right.

## 9.1 Hardware Configuration via DIP Switches

In order to enable hardware selection, DIP switch 3 must be set high. Table 6 lists the appropriate DIP switch settings for different board/camera configurations.

| | | DIP Switches 1&2 | |
|---|---|---|---|
| DIP Switch 3 | Camera | ML506 | ML402 |
| 1 (Hardware Configuration) | DRS U6000 | 00 | N/A |
| | Photon 640 | 01 | 01 |
| | Photon 320 | 10 | 10 |
| 0 (Software Configuration) | DRS U6000 | Set camera via RS-232 | N/A |
| | Photon 640 | | Set camera via RS-232 |
| | Photon 320 | | |

*Table 6: DIP Switch Configurations for Hardware Camera Selection*

## 9.2 Software Configuration via RS-232

When DIP switch 3 is low, camera selection can occur via RS-232 communication. After establishing a connection with the board at 115200 baud, a single ASCII character (with an ASCII new line character as the terminator) must be sent indicating the camera to be selected. The ASCII characters for each camera/board configuration are detailed in Table 7.

| | Evaluation Board | |
|---|---|---|
| | ML506 | ML402 |
| Camera | ASCII Char | ASCII Char |
| DRS U6000 | D | N/A |
| Photon 640 | 6 | 6 |
| Photon 320 | 3 | 3 |

*Table 7: RS-232*

The RS-232 module implemented in the FPGA repeats the received character over the RS-232 link so that the acquisition software can verify that the camera selection command was successfully received. This has been implemented in the MATLAB acquisition GUI further detailed in section 10.0.

## 10.0 Acquisition Software

A custom MATLAB GUI was developed in conjunction with Sandia in order to acquire and analyze data from the system via Camera Link and Ethernet. The GUI is capable of camera selection via software; the user is presented with a list of detected COM ports and chooses the one to which the system is connected. After selecting the camera, the user depresses the "Set" button, which sends the camera selection command to the system. If a valid acknowledgment is not received (likely due to an erroneous selection or bad connection), an error message is displayed. A screen capture of the software is present in Figure 24.

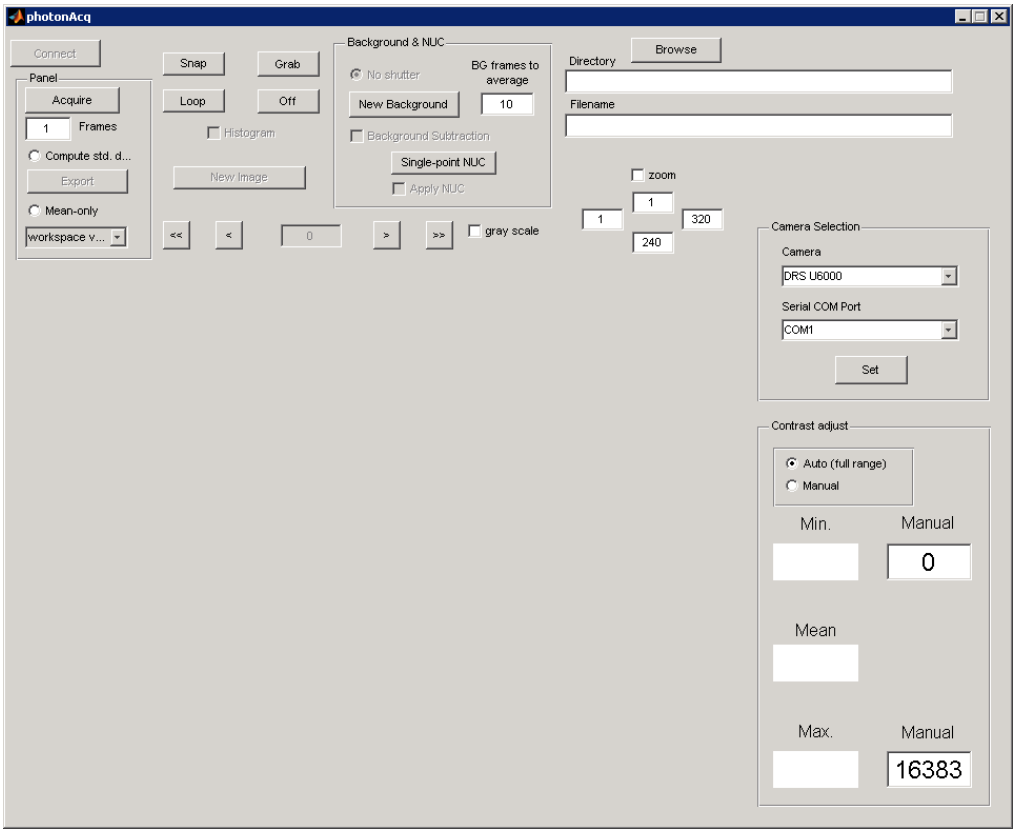



*Figure 24: Screen Capture of MATLAB Acquisition GUI*

The software is also capable of displaying and acquiring live video from the system. Camera files are passed to the MATLAB IMAQ adapter dependent upon the camera selection of the user. Camera files for the Imperx FrameLink Express frame grabber have a .cxf extension, and camera files for the NI PCIe-1427 frame grabber have a .icd extension. NI PCIe-1427 camera files have the additional requirement of having a .iid interface file for each camera file; all .icd and .iid files must be placed in the National Instruments "Data" folder in order to be properly recognized by the "ni" IMAQ adaptor.

## 11.0 Satisfaction of Requirements

Table 8 details the requirements associated with each deliverable as well as the subtasks contained within each requirement. The validation method for each subtask is also detailed.

| Deliverable | Requirement | Tasks | Validation Method | Validation Date |
|---|---|---|---|---|
| Camera Link Data Transfer Code | The system shall use Xilinx ML506 snd ML402 development boards to transfer real time video data from an Indigo Photon 320 infrared camera to a Camera Link-compatible frame grabber. | write VHDL code that can transfer video rate data from the Indigo Photon 320 infrared camera to a computer at both 8 and 14 bits per pixel resolution | validate that the full range of pixel values by examining output test pattern | 3/3/2010 |
| | | write VHDL code that can transfer video rate data from the Indigo Photon 320 infrared camera to a computer at a frame rate of 29.97 frames per second | use MATLAB to acquire frames and timestamps for calculation of video rate. | 3/3/2010 |
| | The system shall use Xilinx ML506 snd ML402 development boards to transfer real time video data from an Indigo Photon 640 infrared camera to a Camera Link-compatible frame grabber. | write VHDL code that can transfer video rate data from the Indigo Photon 640 infrared camera to a computer with both 8 bits per pixel and 14 bits per pixel | validate that the full range of pixel values by examining output test pattern | 3/3/2010 |
| | | write VHDL code that can transfer video rate data from the Indigo Photon 640 infrared camera to a computer at a frame rate of 29.97 frames per second | use MATLAB to acquire frames and timestamps for calculation of video rate | 3/3/2010 |
| | The system shall use a Xilinx ML506 Virtex 5 FPGA Development Board to transfer real time video data from a DRS U6000 infrared camera to a Camera Link-compatible frame grabber. | write VHDL code that can transfer video rate data from the DRS U6000 infrared camera to a computer with 14 bits per pixel resolution | validate that the full range of pixel values by examining output test pattern | 3/3/2010 |
| | | write VHDL code that can transfer video rate data from the DRS U6000 infrared camera to a computer at a frame rate of 29.97 frames per second | use MATLAB to acquire frames and timestamps for calculation of video rate | 3/3/2010 |
| **Deliverable** | **Requirement** | **Tasks** | **Validation Method** | **Validation Date** |
| Camera Compatibility Selection Code | Camera compatibility shall be selectable via DIP switches on the Xilinx ML506 and ML402 development boards. | write VHDL code to support camera capability selection via DIP switches | demonstrate capability | 3/4/2010 |
| | Camera compatibility shall be selectable via software | write VHDL code to communicate with software and accept camera selection commands via RS-232 | demonstrate capability | 5/26/2010 |
| System Status Display Code | The system shall use Xilinx ML506 and ML402 development boards to display status information on the integrated LCD display. | write VHDL code to display status information on the LCD screen on the development board. | demonstrate capability | 6/11/2010 |
| DVI Video Output Code | The system shall use a Xilinx ML506 development board to output real time video data from a camera to a DVI-compatible display. | write VHDL code that can output video rate data from the Indigo Photon 320, Indigo Photon 640, and DRS U6000 cameras to a DVI port using standard 644x512 @60Hz timing. | visual inspection of display | 3/30/2010 |
| VGA Video Output Code | The system shall use a Xilinx ML402 development board to output real time video data from a camera to a VGA-compatible display. | write VHDL code that can output video rate data from the Indigo Photon 320, Indigo Photon 640, and DRS U6000 cameras to a VGA port using standard 644x512 @60Hz timing. | visual inspection of display | 3/30/2010 |
| Ethernet Data Transfer Code | The system shall use a Xilinx ML506 Virtex 5 FPGA Development board to transfer real time video from the Indigo Photon 320, Indigo Photon 640, and DRS U6000 infrared cameras using the UDP protocol over Ethernet | write VHDL code that can transfer video rate data from the infrared cameras over TCP/IP using the ethernet port on the Xilinx board | open saved video/images on TCP/IP client | 3/16/2010 |

*Table 8: Sandia Project Requirements Table*

## 11.1 Frame Rate Verification Methodology

Frame rate verification was performed via a MATLAB script which uses the Image Acquisition Toolbox to calculate the source video rate. This code uses the MATLAB Image Acquisition Toolbox to acquire a sequence of frames and log the timestamp of each one. After acquisition, the script calculates the video rate of the image source and plots the timestamp of each frame as well as the average time difference between each frame. This allows verification of the consistency of incoming frames. Figure 25 contains a screen capture verifying the frame rate of the system (29.97 fps) when connected to an Indigo Photon 320 camera. Frame consistency plots are available in Figure 26 and Figure 27.

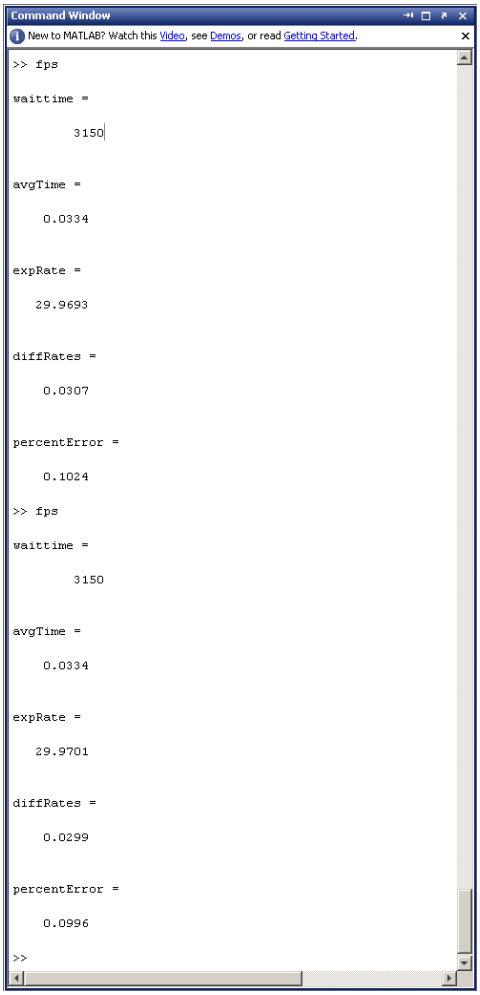



*Figure 25: Frame Rate Calculation via MATLAB Image Acquisition Toolbox*

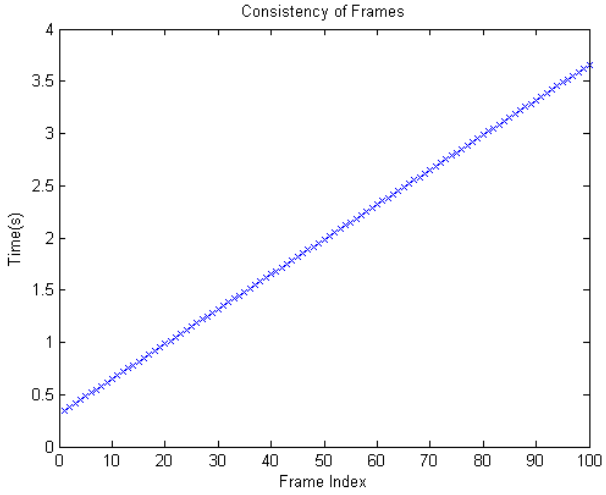



*Figure 26: Consistency of Frame Acquisition via MATLAB Image Acquisition Toolbox*

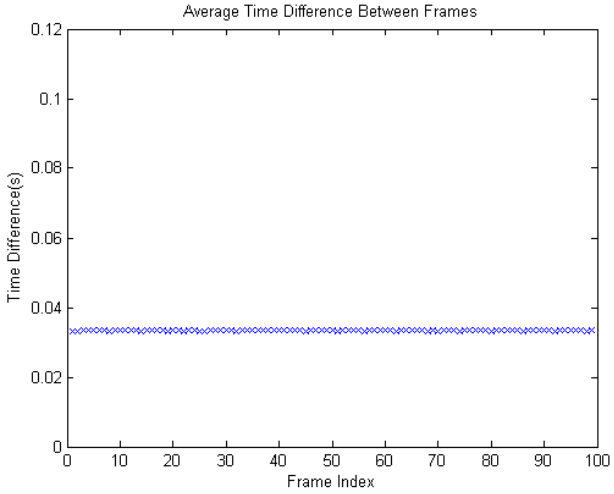



*Figure 27: Average Time Difference Between Frames via MATLAB Image Acquisition Toolbox*

## 12.0 Appendix A: Connections and Pinout

Refer to Table 9 for the connections between the Xilinx ML506 evaluation board headers and the Indigo Photon 320 digital data wires. Table 10 contains the pinout for the Indigo Photon 640 digital data wires.

| Connection | Wire # | Header Group | Header Pin | FPGA Pin |
|---|---|---|---|---|
| Synchronization - | 12 | J5 | 2 | Y18 |
| Synchronization + | 2 | | 4 | AA18 |
| CLK - | 11 | | 6 | W19 |
| CLK + | 7 | | 8 | Y19 |
| Data - | 13 | | 10 | Y21 |
| Data + | 8 | | 12 | Y20 |

*Table 9: Indigo Photon 320 Camera-FPGA Interface Pinout*

| Connection | Wire Color | Header Group | Header Pin | FPGA Pin |
|---|---|---|---|---|
| Synchronization - | Pink | J5 | 2 | Y18 |
| Synchronization + | Brown | | 4 | AA18 |
| CLK - | White | | 6 | W19 |
| CLK + | Orange | | 8 | Y19 |
| Data1 - | Purple | | 10 | Y21 |
| Data1 + | Blue | | 12 | Y20 |
| Data2 - | Green | | 14 | W24 |
| Data2 + | Black | | 16 | W23 |

*Table 10: Indigo Photon 640 Camera-FPGA Interface Pinout*

Refer to Table 11 for the connections between the Xilinx ML506 evaluation board headers and the Camera Link cable wires.

| Connection | Wire # | Header Group | Header Pin | FPGA Pin |
|---|---|---|---|---|
| CLK - | 5 | J5 | 18 | Y23 |
| CLK + | 18 | | 20 | Y22 |
| X0- | 2 | | 22 | AA20 |
| X0+ | 15 | | 24 | AA19 |
| X1- | 3 | | 26 | AA17 |
| X1+ | 16 | | 28 | Y17 |
| X2- | 4 | | 30 | AC20 |
| X2+ | 17 | | 32 | AB20 |
| X3- | 6 | | 34 | AD21 |
| X3+ | 19 | | 36 | AE21 |
| Drain 1 | N/A | | 55 | N/A |
| Drain 2 | N/A | | 57 | N/A |
| Drain 3 | N/A | | 59 | N/A |
| Drain 4 | N/A | | 61 | N/A |
| Drain 5 | N/A | | 63 | N/A |

*Table 11: FPGA-Camera Link Interface Pinout*

## 13.0 Appendix B: DRS U6000 Camera Errata/Confusion

The following details erroneous/confusing information that was presented in the "SNL UFPA Hardware Interface Charts" for the DRS U6000 Camera:

1. Slide 1 details the following: *"HotLink requires "double-pump" to maintain 14-bit data along with frame sync (FPA) & field sync (frame grabber) bits. Each 10-bit chunk has 8 data bits (7 data bits + 1 unused bit) + 1 data clock bit + 1 special character bit. Unused data bit is either MSB or LSB (?)"*. This isn't how pixel values are output by the camera. The first two bits of the first byte of a pixel are unused; in other words, each pair of bytes should be accumulated into a shift register, and the lower 14-bits make up the pixel value.
2. K28.1 control bytes do not signal the beginning but instead the end of even and odd fields, respectively.
3. In slide 9, the even and odd fields are shown to be equal in size with respect to the number of data and control bytes. However, the idle field after the K28.0 control byte is not actually 15477 bytes long, as is listed, but instead 13893, a full row of pixels less than indicated. The de-interlacing logic must be designed to accommodate this.
4. There is no "special character bit". This was noted to be a line synch channel, but it does not appear to output any meaningful data.
5. The DRS camera, in general, seems to output 16-bits, eight of which are the FF and FE bytes detailed in slide 9. The LSB of these additional eight bits can be used as a line synch/data valid channel.

## 14.0 Appendix C: Obsolete Work

This section details work which was previously done but has been rendered irrelevant or obsolete.

### 14.1 FPGA-UART Interface: Camera Control via RS-232

RS-232 camera control was implemented only for the Indigo Photon 320 and 640 cameras. The FPGA controlled the camera using the RS-232 protocol over the UART interface. The required RS-232 settings are detailed in Table 12. The Photon 320 and 640 expects a specific packet format, detailed in Table 13. A subset of the commands are able to be sent from the FPGA using the DIP switch inputs; the commands that have been implemented are detailed in Table 14 along with the corresponding packet values and DIP switch configuration.

| Parameter | Value |
|---|---|
| Baud rate: | 57600 |
| Data bits: | 8 |
| Parity: | None |
| Stop bits: | 1 |
| Flow Control: | None |

*Table 12: Indigo Photon 320 RS-232 Communication Settings*

| Byte # | Upper Byte |
|---|---|
| 1 | Process Code |
| 2 | Status |
| 3 | Reserved |
| 4 | Function |
| 5 | Byte Count (MSB) |
| 6 | Byte Count (LSB) |
| 7 | CRC1 (MSB) |
| 8 | CRC1 (LSB) |
| | (Data) |
| … | (Data) |
| N | (data) |
| N+1 | CRC2 (MSB) |
| N+2 | CRC2(LSB) |

*Table 13: Indigo Photon 320 RS-232 Packet Structure*

| Command | DIP Switch Configuration (Switches 1 and 2) | Parameter | DIP Switch Configuration (Switches 3-8) | Function Byte | Parameter Byte |
|---|---|---|---|---|---|
| Divital Video Output Mode | 00 | 14-bit Raw | 00XXXX | 0x12 | 0x0003 |
| | | 14-bit Filtered | 01XXXX | | 0x0000 |
| | | 8-bit | 1XXXXX | | 0x0001 |
| FFC Mode (Manual/Auto) | 01 | Manual | 00XXXX | 0x0B | 0x0000 |
| | | Auto | 01XXXX | | 0x0001 |
| | | External | 1XXXXX | | 0x0002 |
| FFC Interval | 10 | # Frames | Binary Input (DIP 3 is MSB) | 0x0D | 0x0000-0xFFFF |
| DO FFC | 11 | N/A | XXXXXX | 0x0C | N/A |

*Table 14: Implemented Indigo Photon 320 Commands with Corresponding Packet Values and Input Configuration*

## 14.2 Acquisition Software

In addition to the MATLAB GUI application, a C++ function titled “grab” was written. This function utilizes the NI IMAQ (image acquisition) API to capture a still image from the acquisition card and return it to MATLAB in the form of an array for viewing or processing. The software is compatible with the 8-bit and 14-bit modes of the Indigo Photon 320 camera.