INTERACTIVE STUDIO

ESC532 – Final Design Report

Farshad Safavi **Shaoyu Chen** **Shadi Dashmiz** **Harry Wang**

The Edward S. Rogers Sr. Department
of Electrical & Computer Engineering
UNIVERSITY OF TORONTO

APRIL 8, 2016

TABLE OF CONTENTS

1. OVERVIEW

1.1 BACKGROUND

In this project the main goal is to create an interactive studio employing chroma keying effect. Chroma keying is a post-production technique to overlay a background color with another image or stream of video [1], [2]. This method has been applied heavily in many different fields such as news casting, weather forecasting and filming industry. Background color can be chosen arbitrarily, but green and blue colors have the highest contrast with most of human body skin color which makes it more effective for replacement purposes [3].

In weather forecast studio a person stands in front of a blue screen and the part of background which he is not covering is substituted with a weather map. Interactive part starts when the person points to a specific spot on the weather map. As we point to any location, weather condition appears on the map in real time.

1.2 GOALS

The main goal of this project is to create a studio which is able to replicate chroma keying effect with addition of interactive elements. A blue color background is streamed into the camera and is replaced with a stored image of Canadian weather map. A laser pointer motion is traced as it moves across the background. As laser pointer points to different cities on the map, different animations of sun, rain and snow will pop up based on different weather conditions. Fig.1 shows the functionality of chroma keying effect which has been implemented during this project.

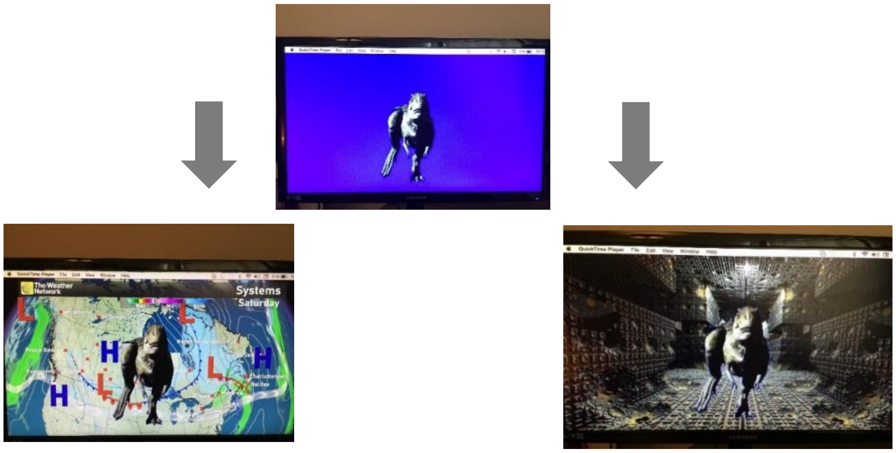

Fig 1. Chroma keying effect implemented during the project

Fig.2 shows motion detection. As you can see, blue background is replaced with a weather map. A moving red dot is added to emulate red laser. A yellow dot will track the red dot which illustrates laser tracking feature.

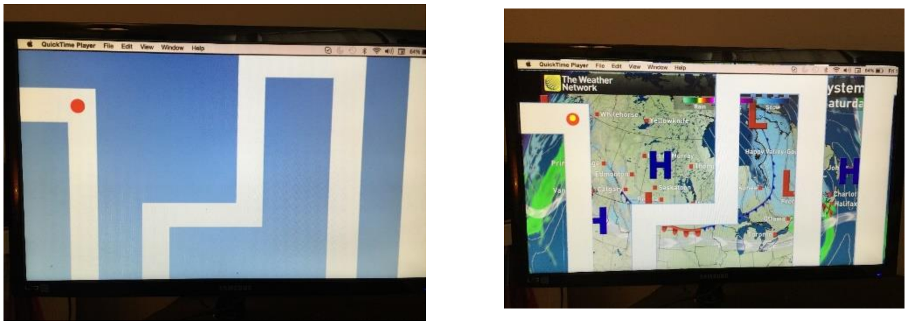

Fig 2. Detecting a red circle implemented in this project

The final system is exactly what we initially proposed so no further changes or modifications where needed.

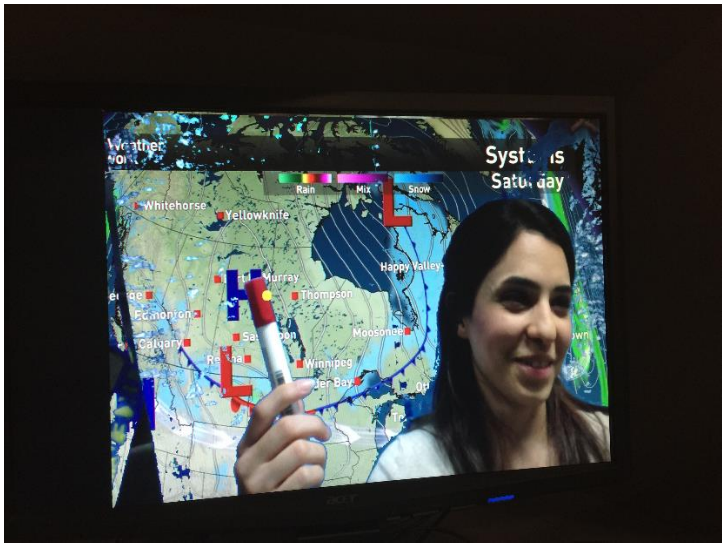

Fig 3. Final Interactive Studio

1.3 Block Diagram

Block diagram of complete system including custom and existing IPs is shown in Fig.4.

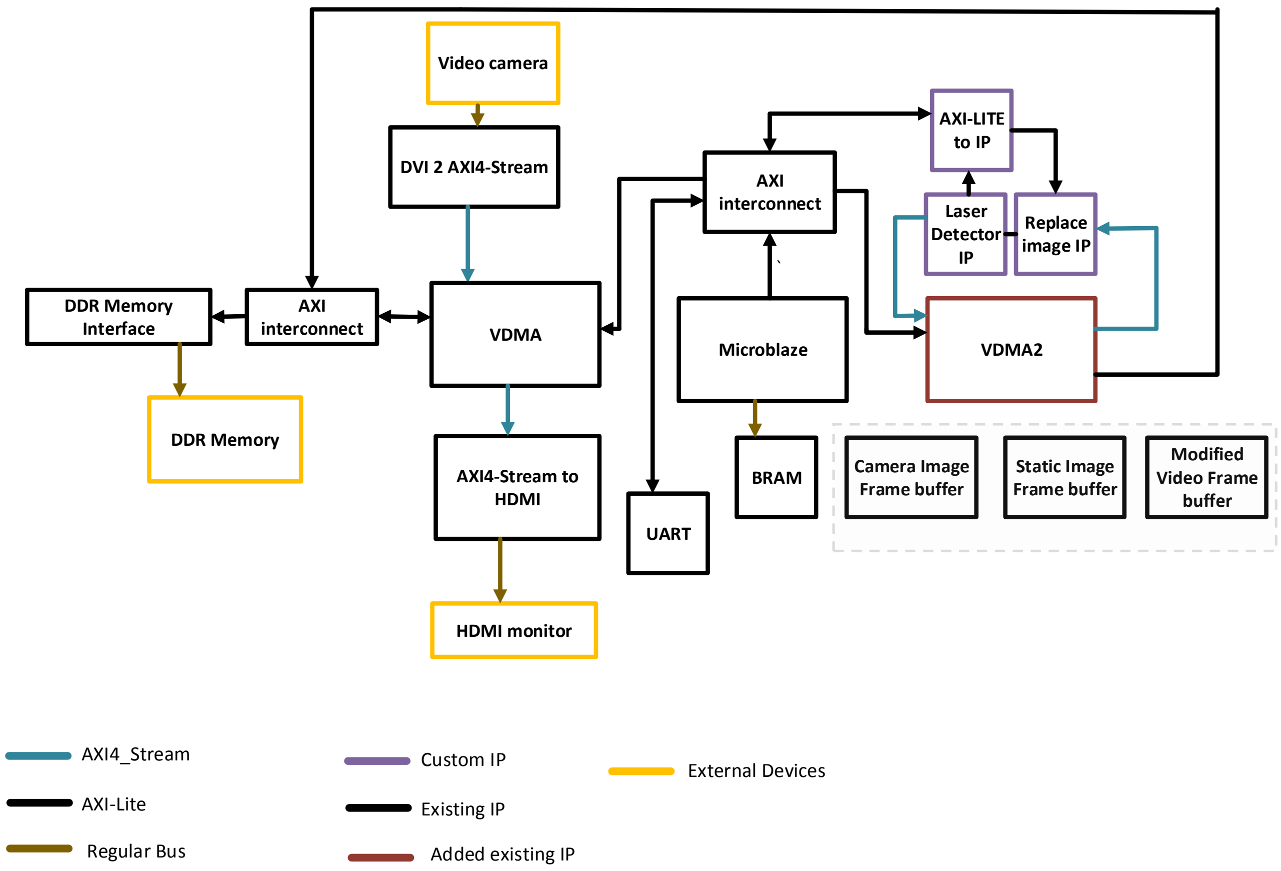

Fig. 4 Block diagram of completed system

1.4 BRIEF DESCRIPTION OF IPS

IP	Function	Origin
Video Camera	Stream of Video.	Group
DVI-2-RGB(1.5)	Video decoder.	DIGILENT
AXI VDMA(6.2)	Provides high-bandwidth direct memory access between memory and AXI4-Stream video type target peripherals including peripherals which support the AXI4-Stream Video protocol [4].	Xilinx
AXI4-Stream to HDMI	It provides a bridge between video processing cores with AXI4-Stream interfaces and a video output. It includes Xilnx IP AXI4-Stream to video out and DIGILENT IP RGB to DVI video encoder.	Xilinx + DIGILENT
HDMI Monitor	Displays the video stream.	Group
DDR Memory interface	It includes memory interface generator (mig 7 series version 2.4). Creating memory controller and physical interface for DDR memory.	Xilinx
Microblaze(9.5)	An embedded soft core reduced instruction set computer optimized for implementation in Xilinx FPGAs [5].	Xilinx
AXI Uartlite (2.0)	Interface and controller between UART data transmission and Axi-interface [6].	Xilinx
BRAM	The BRAM Block is a configurable memory module that attaches to BRAM interface controller.	Xilinx
COLOR_MK2 IP	AXI stream IP which is modified to perform color replacement and laser detection: 1) Inputs frames from video stream and static image, it performs the blue detection and replaces the blue pixel with image and outputs the replaced frame back to VDMA. 2) Recognizes and tracks a red laser pointer precisely.	Xilinx AXI Stream IP modified by our Group

AXI-Lite to IP	Communicates between microblaze and laser detection and replacement IP to set the range of blue color for detection and returns the center of laser pointer.	Xilinx AXI Lite IP modified by our Group
Software running on MicroBlze	to perform the tasks: 1) Controlling and configuring first VDMA for streaming video [7]. 2) Controlling and configuring the second VDMA for blue color. 3) Detection and replacement with a static image. 4) Read and Write through the second VDMA. 5) Drawing animation on the weather map. 6) Controlling the range of blue color to be replaced. 7) Transmission of static image via UART.	DIGILENT Group Group Group Group Group Group
TeraTerm(2.3)	In this project weather map has been transferred as .bin file using TeraTerm [8].	Open source software
Processing	Convert any image to a binary file [9].	Open source software

2 OUTCOME

2.1 RESULTS

Overall, our design project was successful in achieving most of our initial proposed goals and requirements. These goals along with their final status are summarized in the table below.

Feature	Status	Comments
Color replacement of blue background image with another image stored in memory	Complete	The design stores the background image and the replacement image in two different buffers and replaces the background image when a certain color threshold is met. Currently, the performance of the replacement on a 1280x720 resolution is approximately 24 frames/sec.

Laser detection and tracking on background image	Complete	A red laser pointer is detected on the provided blue background image using a similar color threshold as the color replacement function. The location of the laser is sent out to the Microblaze, allowing for interactive animations to occur. The detection of the red pointer depends largely on lighting. For our camera, the red laser point is actually white in the center, therefore, impacting the accuracy of our tracking. To compensate, we used a red marker cap instead for greater accuracy.
Interactive drawing at laser location	Complete	When the location of the laser pointer is known, an interactive element can be drawn at this location. Simple shapes such as lines and circles can be drawing easily using the Bresenham's line algorithm. However, more complex shapes are more difficult as it involves storing actual drawings inside memory. Therefore, our project drew various simple shapes of different colors to emulate a cruder version of what we initially proposed.

2.2 POSSIBLE FURTHER IMPROVEMENTS

One way to improve upon what we have designed is to make use of additional buffers and synchronization protocol to increase the frame rates of our streaming video. Currently, our design makes use of three frames buffers to store the original frame, replacement image, and the replaced frame. This method allowed us to not consider potential synchronization issues would occur between frames as there is only one frame to handle each action. However, the tradeoff is that we are limited to how much of an image we can stream at a time. With additional frame buffers, we would be able to stream multiple parts of a frame simultaneously, thus increasing the overall FPS of our system. However, achieving this would require synchronization between the frame buffers to ensure that the image being stream align correctly.

Another method of improvement for our design is to improve our replacement algorithm. Currently, the IP used to perform the color replacement would stream data into a FIFO, replace the data in the FIFO, and then stream out the replaced image data, in sequential order. This method was the easiest in terms of implementation. However, once again, the trade-off is a reduction in overall performance as Microblaze would have to wait for each action to be finished for each section of data. By making it such that the stream-in and replacement occur simultaneously, this reduces the overhead, thus increasing the speed. However, once again, this would require synchronization between the stream-in and read functions to ensure proper alignment.

As for future extension of our project, connecting Ethernet capabilities would be one potential option. Our original goal was the design the system mimicking a weather forecast station. The current design simply draws shapes such as rain, sun, or snow based on predefined locations of our background image. By connecting Ethernet, the weather of a certain location could be updated in real-time and a dynamic interaction can appear based on this data.

3 DESCRIPTION OF BLOCKS

3.1 COLOR_MK2 IP

The color_mk2 IP contains the color replacement and laser detection functionality. The color detection range for both functions and the x and y location of the detected laser need to be configured and read using Microblaze code through the AXI_to_colorIP block of AXI-Lite interface.

There are 3 customizable parameters: H_RESOLUTION, V_RESOLUTION and NUM_LINES. Set the H_RESOLUTION and V_RESOLUTION to the desired horizontal and vertical display resolution. Set the NUM_LINES as the number of lines you want the IP to process at a time. 4 lines are the maximum you are allowed to set for the Nexys video FPGA board. For this project these parameters are set to H_RESOLUTION = 1920, V_RESOLUTION = 1080, NUM_LINES = 4 (fig.4 below shows the top level block diagram for the IP).

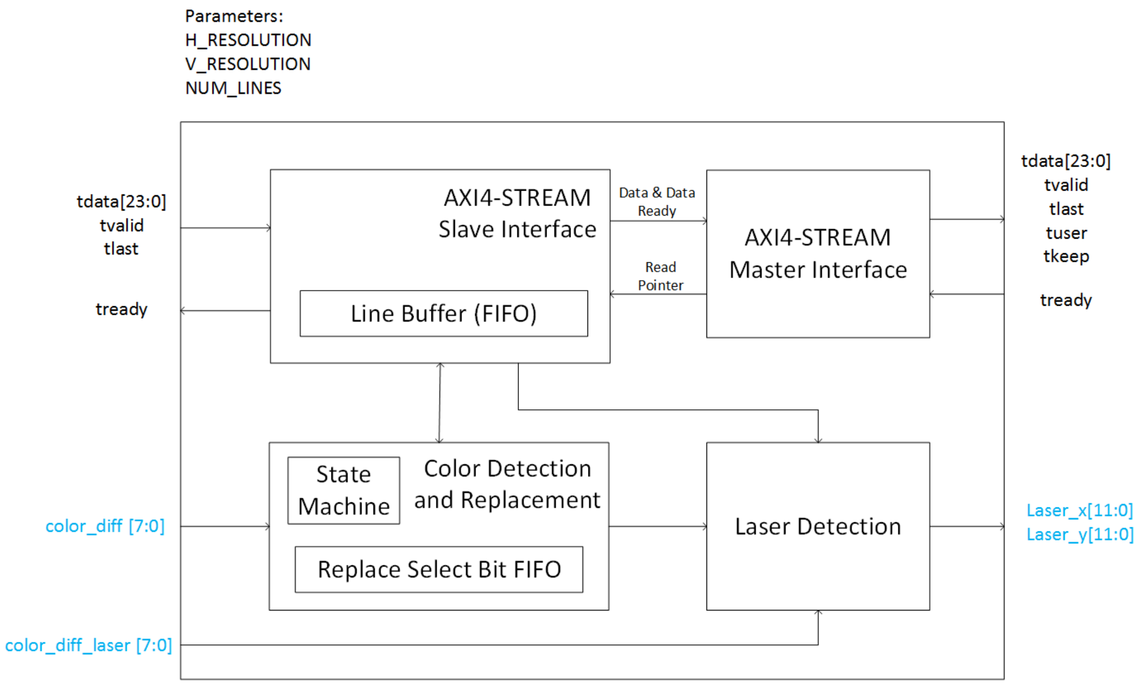

Fig.5 Block diagram for the color_mk2 IP.

3.1.1 Color Replacement

It takes two frames (in the form of one or multiple lines) from the VDMA read channel, and replaces all the blue pixels of the first frame that fall into the detection range with the pixels of the second frame. Then the new frame is send back to the VDMA via the write channel.

The criteria for replacement is based on the 8-bit color_difference value. The pixel will be replaced if:

$$\text{blue pixel value} - (\text{red pixel value} + \text{green pixel value})/2 >= \text{color\_difference}$$

It uses three states to achieve color replacement: Detection, Replacement and Stream. In the Detection state, the first frame is stored in the Line Buffer FIFO. As the write pointer of the FIFO increments; if the pixel color is identified as blue, its corresponding Replace Select Bit will be set to 1. In the Replacement state, any pixel that had its Replace Select Bit set to 1 will be replaced by the pixel from the second frame. At the end of this state, a Data Ready signal is sent to the AXI4-Stream Master interface and the state transit to the Stream state. The replaced frame stored in the FIFO will be send to the VDMA write channel.

The AXI-Stream slave and master interface is modified from the template code that is available when initially creating an AXI-Stream IP in Vivado. The M00_axis_tuser and

M00_axis_tkeep signals are added to the master interface. M00_axis_tuser indicates the start of a frame, and M00_axis_tkeep is a 3-bit constant binary number 111 indicating all the 3 bytes (24-bit) of M00_axis_tdata is valid [10].

In order to use this IP, the VDMA need to be configured in Microblaze code. For the read channel, you need to set its frame size as (H_RESOLUTION * 3 * NUM_LINES) by 1, assign two frame buffer to read and enable circular buffer and set the frame counter to be 2.

For the write channel, you need to set its frame size as (H_RESOLUTION * 3 * NUM_LINES) by 1, assign a frame buffer to write and disable the circular buffer and frame counter.

The buffer memory address for both the write and read channel need to be reconfigured for every NUM_LINES lines. For example for the write channel, if H_RESOLUTION = 1920, V_RESOLUTION = 1080 and [4] [5] [6, 7]NUM_LINES = 4, then for every 4 lines, the address need to be incremented by (1920 * 3 * 4). The target frame buffer will be filled after 1080 / 4 = 270 iterations.

3.1.2 Laser Detection

The laser detection IP operates relatively similarly to the color detection IP described in the previous section. The input to this IP is the same data stored in the FIFOs of the color replacement IP. When the color replacement IP is in the detection state, IE, when it is reading all the pixels of the original image to determine the blue background, the laser detection IP would begin to function. Otherwise, it would remain idle. Beside the input data, an additional color_difference_laser variable is used to determine the color threshold of the laser that would be detected.

To implement the laser detection algorithm, four variables – x1, x2, y1, y2 – were used to determine the borders of the laser. Due to our background being completely blue, this was easy to accomplish as any color that is not blue would indicate the location of the laser. More precisely, if the previous pixel and current pixel are different in color by the set color_difference_laser threshold, then one of the four borders has been found. Once all four border values are obtained, the average of the x and y values is then taken to determine the center of the laser. This output location is sent through an AXI-Lite IP connected to the Microblaze, allowing the Microblaze to process this location for drawing various different interactive shapes.

3.2 AXI_TO_COLOR IP

This IP is used to configure the color_difference and color_laser_diff value of the color_mk2 IP in software via AXI-Lite interface, as well as access the laser location of each frame. Simply connect the AXI-Lite interface to the AXI Interconnect and connect the ports showing in the following table to the ports with the same name on the color_mk2 IP [12].

Table: Port information

Port Name	Data width(bit)	Address Offset	Read	Write
color_difference	8	0	Yes	Yes
color_diff_laser	8	4	Yes	Yes
laser_x	12	8	Yes	No
laser_y	12	12	Yes	No

3.3 AXI VIDEO DIRECT MEMORY ACCESS

AXI VDMA is a soft IP core version 6.2 developed by Xilinx. Two VDMAs are used in our system. Both VDMAs perform read and write on both channels [4]. The first VDMA reads one frame from the memory and write to the display buffer. The second VDMA is used as an image provider for AXI-Stream IP. It frequently reads four lines of video frame and four lines of static image and writes four lines of modified frame to display buffers. The second VDMA is configured to use frame counter feature. This feature associated with read in circular transfer mode allows us to continuously pass two different addresses of video frame and static image simultaneously. The feature help to optimize the software to call read function once instead of two times.

3.4 FIRMWARE RUNNING ON MICROBLAZE

The original firmware controls video capturing and writing to three display buffers. C program running on Microblaze, originally, had options to generate different pattern, change the displayt resolutions and video start and stop. Following functions are added by our group to the original firmware of Nexys Video HDMI Demo project [7]:

- A function which receives an image from UART and store raw image in DDR.
- Configure and initialize the second VDMA.

- Configuring read and write channel of the second VDMA.
- Configure frame counter feature of the second VDMA.
- A function to send video frame and static image to the IP and display the modified image after color replacement and laser detection.
- Functions to set the AXI lite color difference registers.
- Functions to read laser pointer locations.
- Functions to draw animations on a specific location on the screen.

After the laser has been detected the center is provided to Microblaze, depending which part of map has been shined by laser different animation will appear on the screen. All the animation are hard coded into the Microblaze program. Sun has been simulated with blinking yellow circle. Snow is simulated with rows of white pixels and rain is simulated with dropping lines randomly chosen by rand function during in each frame.

3.5 SOFTWARE

A program in Processing software is developed which converts any image loaded into Processing into a binary (".bin") file [9].

3.6 EXISTING IPS

This section briefly describes the blocks which existed in the original Nexys Video HDMI Demo project [7].

3.6.1 DVI-to-RGB Video Decoder

This IP interfaces directly to raw transition-minimized differential signaling (TMDS) clock and data channel inputs as defined in DVI 1.0 specs for Sink devices. It decodes the video stream and outputs 24-bit RGB video data along with the pixel clock and synchronization signals recovered from the TMDS link [13].

3.6.2 Video In to AXI4-Stream

It interfaces from a 24-bit RGB video source to the AXI4-Stream Video Protocol Interface. This core works with the timing detector portion of the Xilinx Video Timing Controller core [14]. It provides a bridge between a video input and video processing cores with AXI4-Stream Video Protocol interfaces [15].

3.6.3 Video Timing Controller

It is a general purpose video timing generator and detector. The input side of this core automatically detects horizontal and vertical synchronization pulses, polarity, blanking timing and active video pixels. While on the output, it generates the horizontal and vertical blanking and synchronization pulses used with a standard video system including support for programmable pulse polarity [14].

3.6.4 AXI4-Stream to Video Out

This IP interfaces from the AXI4-Stream interface implementing a Video Protocol to a 24-bit RGB video source. This IP works with the Xilinx Video Timing Controller (VTC) IP [13]. It provides a bridge between video processing cores with AXI4-Stream interfaces and a video output [16].

3.6.5 RGB-to-DVI Video Encoder

This IP interfaces directly to raw TMDS clock and data channel outputs as defined in DVI 1.0 specs for Source devices. It encodes 24-bit RGB video data along with the pixel clock and synchronization signals [17].

3.6.6 Microblaze (9.5)

This IP is a Xilinx 32-bit RISC Harvard architecture soft processor which provide high level control logic for the system. Microblaze is used in our project to control data flow of video streaming [5].

3.6.7 AXI Uartlite (2.0)

This IP is universal asynchronous receiver transmitter lite interface connected to AXI lite interface. It is used to transfer our binary file to DDR and display our menu on the screen. The baud rate of UART is 115200 for transfer data and display menu options [6].

3.6.8 AXI Interconnect (2.1)

This IP is connects one or more AXI memory-mapped master devices to slave devices. The AXI interconnect core is specialized for memory-mapped transfers [11].

3.6.9 Memory Interface Generator (2.4)

Mig_7_series is a memory controller which generates a clock and manages transfer data on both edges of the clock for double data-rate operations. Memory interface generator provides an interface for our system to access memory [18].

4 SCHEDULE

This table illustrates our original milestone schedule vs. tasks which we have accomplished. In overall, we have been able to follow our milestones on schedule. The deviation from the schedule occurred in week 3 and 4; however, we come back on track by working in parallel. To work in parallel, two members debugging color replacement block and the other two members developed laser detection. Our team realistically allocated time for verification, validation and integration in the original schedule. For example, week 7 is scheduled only for validation and integration. It helps the group complete the project on time and account for unforeseen difficulties.

WEEK 1	**February 8th**
Proposal	**Tasks**
FPGA Interface with Camera (video decoder) and FPGA interface with display (TFT controller, VGA).	• Accomplished streaming video from the camera to display (monitor) using HDMI reference design.
Milestone 1: Streaming a video and display it on monitor	Milestone 1 is achieved.
WEEK 2	**February 22nd**
Proposal	**Tasks**
Store image in memory.	• Developed a program to converts any image to a binary (".bin") file. • Transfer binary file over the serial port to UART and store image in DDR memory.
Developing Chroma Key Algorithm in RTL.	• RTL design for Chroma Key algorithm outlined.
Start adding Color Replacement block and verify in simulation.	• Hardware architecture complete. • Create a new AXI stream block and VDMA.
Milestone 2: Video streaming and display static Image.	Milestone 2 is achieved. The static image is displayed on monitor.
WEEK 3	**February 29th**
Proposal	**Tasks**
Integrate RTL block into the system	• RTL development for color replacement completed.
Validate color replacement IP.	• Integrate new VDMA into the system and perform read and write. • Prototype color replacement in software and validate color replacement in software.
Debug and regression.	• RTL behavioral simulation tests are passed.
Milestone 3: Video streaming and replacing green background with a stored image	Milestone 3 postponed for one week. Color replacement passed in simulation but integration and validation pushed one week back.

WEEK 4	**March 7th**
Proposal	**Tasks**
Debug color replacement RTL block.	- Different versions of RTL is written. - The new IP with much simpler and robust design is selected which passed both behavioral and post synthesis simulation. - Add a new color replacement IP to the system and program the IP using VDMA.
Adding a laser detector block for user interaction purposes.	- Algorithm for detecting the laser pointer is outlined.
Milestone 4: Integrate modules with a new feature (interactive feature)	Milestone 4 delayed for one week. To come back on schedule, two members continue integrating color replacement IP and two members worked on laser detection IP.
WEEK 5	**March 14th**
Proposal	**Tasks**
Integrate the color replacement IP block to the system.	- Developed a simplified version of the system for system level simulation. - Accomplished color replacement in video streaming.
Creating the laser detection IP.	- RTL development for laser detection block completed. - Simulation of laser detection passed. - Prototyping laser detection in software.
Display animation when the object motion is detected.	- Draw a very simple shape as an animation.
Milestone 5: Display animation on screen	We come back on schedule for milestone 5.
WEEK 6	**March 21st**
Proposal	**Tasks**
Integrate software and hardware to the system	- Integrate and validate the color replacement and laser detection IP into the system.
Combining two features (Chroma keying + display interactive animation)	- Accomplished color replacement and draw a simple shape when the device detects red color. - Add registers to control blue and red color from the software. - Add registers to read locations of laser pointer.
Final regression and confirm system work properly.	- Start optimizing the system in software by using frame counter feature.
Milestone 6: Streaming live video with interactive animations between the object's motion and the background image	Milestone 6 is achieved.
WEEK 7	**March 28th**
Proposal	**Tasks**

Validation and verification of the system	• Optimizing both hardware and software to read and write multiple lines.
Debug possible issues for final demo and presentation.	• Draw different shapes as animations and verify they respond to the interactive element.
Milestone 7: Test final project and prepare for demo.	Milestone 7 is achieved.
WEEK 8	**April 4th**
Proposal	**Tasks**
Final project checkoff	• Integrate all components of the project together. • Test and validate whole system in the design lab. • Rehearse and prepared for presentations.
Project demos and presentation	Milestone 8 is achieved.

5 DESIGN TREE DESCRIPTION

The summary of key files which are created by our group is provided in this section:

Src: main hdmi.xpr project

- **color_laser_detect_ip_v1.44**: the AXI stream IP contains color replacement and laser detection functionality
    - **color_mk2_v1_0.v**: modified AXI stream IP, read data from vdma and perform color replacement and laser detection.
- **AXI_Lite_IP_v1.1:** AXI lite IP contains registers to write blue and red detection range and read registers for the x and y location of the detected laser.
    - **AXI_to_colorIP_v1_0_S00_AXI.v :** contains two write registers and two read registers which are connected to color_mk2 IP.
- **hdmi.sdk**
    - **video_demo/src**
        - **video_demo.c:** contains all functions to store the image, control data flow for display video after color replacement and laser detection and animations.
        - **video_demo.h**
        - **vdma**
            - **vdma.c:** All the functions to configure the second vdma read and write channel, in addition to operate read, write, parking and reset.
            - **vdma.h**

Docs: contains final report, video demonstration of the project and slides for final presentations.

- **Group Report**
- **Slides**
- **Video**

6 REFERENCES

[1] "MEDIA COLLEGE," [Online]. Available: http://www.mediacollege.com/glossary/c/chroma-key.html. [Accessed 8 April 2016].

[2] "waybackmachine," 14 2 2005. [Online]. Available: http://web.archive.org/web/20050315004609/http://en.chinabroadcast.cn/2246/2005-2-14/90@206385.htm. [Accessed 8 April 2016].

[3] "techopedia," [Online]. Available: https://www.techopedia.com/definition/476/chroma-key. [Accessed 8 April 2016].

[4] "Xilinx Inc. AXI VDMA," 18 November 2015. [Online]. Available: http://www.xilinx.com/support/documentation/ip_documentation/axi_vdma/v6_2/pg020_axi_vdma.pdf. [Accessed 8 April 2016].

[5] "Xilinx Inc. Microblaze," [Online]. Available: http://www.xilinx.com/products/intellectual-property/microblazecore.html#overview. [Accessed 8 April 2016].

[6] "Xilinx Inc. AXI UART Lite," 18 November 2015. [Online]. Available: http://www.xilinx.com/support/documentation/ip_documentation/axi_uartlite/v2_0/pg142-axi-uartlite.pdf. [Accessed 8 April 2016].

[7] "Digilent Inc HDMI Demo," [Online]. Available: https://reference.digilentinc.com/nexys-video:hdmi6. Hdmi demo: https://reference.digilentinc.com/nexys-video:hdmi. [Accessed 8 April 2016].

[8] "TeraTerm," [Online]. Available: https://ttssh2.osdn.jp/index.html.en. [Accessed 8 April 2016].

[9] "Processing," [Online]. Available: https://processing.org/. [Accessed 8 April 2016].

[10] "Xilinx Inc. AXI4-Stream Interconnect," [Online]. Available: http://www.xilinx.com/products/intellectual-property/axi4-stream_interconnect.html. [Accessed 8 April 2016].

[11] "Xilinx Inc. AXI Interconnect," 18 November 2015. [Online]. Available: http://www.xilinx.com/support/documentation/ip_documentation/axi_interconnect/v2_1/pg059-axi-interconnect.pdf. [Accessed 8 April 2016].

[12] "Xilinx Inc. AXI4-Lite IP Interface," [Online]. Available: http://www.xilinx.com/products/intellectual-property/axi_lite_ipif.html. [Accessed 8 April 2016].

[13] E. Gyorgy, "Digilent Inc. DVI-to-RGB (Sink)," 24 July 2015. [Online]. Available: www.digilentinc.com. [Accessed 8 April 2016].

[14] "Xilinx inc. Video Timing Controller," [Online]. Available: http://www.xilinx.com/products/intellectual-property/ef-di-vid-timing.html. [Accessed 8 April 2016].

[15] "Xilinx Inc. Video In to AXI4 Stram," [Online]. Available: http://www.xilinx.com/products/intellectual-property/video_in_to_axi4_stream.html. [Accessed 8 April 2016].

[16] "Xilinx Inc. AXI 4 Stram to Video Out," [Online]. Available: http://www.xilinx.com/products/intellectual-property/axi4_stream_to_video_out.html. [Accessed 8 April 2016].

[17] E. Gyorgy, "Digilent Inc. RGB-to-DVI (Source)," 23 April 2015. [Online]. Available: www.digilentinc.com. [Accessed 8 April 2016].

[18] "Xilinx Inc. mig_7series," 18 November 2015. [Online]. Available: http://www.xilinx.com/support/documentation/ip_documentation/mig_7series/v2_4/ug586_7Series_MIS.pdf. [Accessed 8 April 2016].