Curling Delivery Real Time Feedback and Monitoring System

Curling Coach

Nicole Stodola, Chris Pederson, and Gerry Finlay

This system uses a video camera and targets to verify an individual's correct stance during their curling delivery.

Abstract

This project aims to assist curlers with improving their delivery slide from the hack through the use of a real time feedback video analysis system. By utilizing a RCA connected video camera and the Altera DE2 FPGA prototyping board we are able to track the athlete through each video frame to detect if there has been undesired movement. Targets on the shoulders of an athlete allows us to track the athlete efficiently. The most important body position during a curling delivery is that the athlete's shoulders remain level, the system will be able to detect if the shoulders are not level by referencing the other shoulder. The captured video is displayed on a monitor with markers superimposed over the image to show the tracking. If the system detects an undesired movement, such as the athlete's shoulders falling out of level, a fault is triggered and a real time alarm will sound. The alarm sounds will be created and output using C code. The main benefit of this system over current video capture and tracking systems is that the curler receives immediate feedback of faults in their slide, rather than after the entire delivery is complete.

Table of Contents

Functional Requirements

The Curling Coach system uses two markers placed on the shoulders of a curler and an RCA connected video camera in front of the athlete to verify that their shoulders remain level throughout their delivery. By tracking the placement of the shoulder markers in a video frame and analyzing this placement in each subsequent frame, we are able to track the movement of the athlete and send an audio flag to the athlete which informs them of their incorrect stance at the moment an error occurs.

The original project scope had a Review, Tracking and Save mode. The Tracking mode simply evaluated the shoulder markers and detected errors in the levelness of the curlers shoulders while displaying the modified video. The Save Mode would just capture and save the video for later playback and analysis and the Review mode played back the saved video and performed analysis during playback. The menu selection between these different modes was to be done using a LCD touch screen. Backup plans had included using the LCD touch screen for the video output only and using the 4-button switches on the DE2 as the menu selection.

The features removed from the original project scope were the three modes and the menu display. The real time analysis of the video (Tracking Mode) was the main aspect of the project and was accomplished. The Save mode and Review mode were removed from the project because the SD card integration and video compression were more difficult and time consuming than originally thought. The LCD touch screen was removed since a menu was no longer needed with the removal of the different modes. Using the LCD touch screen to display the video was also removed from the project because the integration of the LCD touch screen was not possible in the given time frame. A feature that was added to the original project scope was a video input configuration mode. The configuration mode allows the user to set the RGB threshold values so as to get the best possible tracking of the targets. This feature allows the system to be used in different lighting conditions with optimal tracking.

Hardware Design

The hardware component of our system is based on a SOPC centric design. We use many of the Altera University Program (UP) Video IP cores to achieve basic video functionality. In addition to the UP IP cores, we also have designed a fully custom VHDL block which monitors the video that comes through our system. The video chain uses the Avalon Streaming interface for the high speed nature of video data.

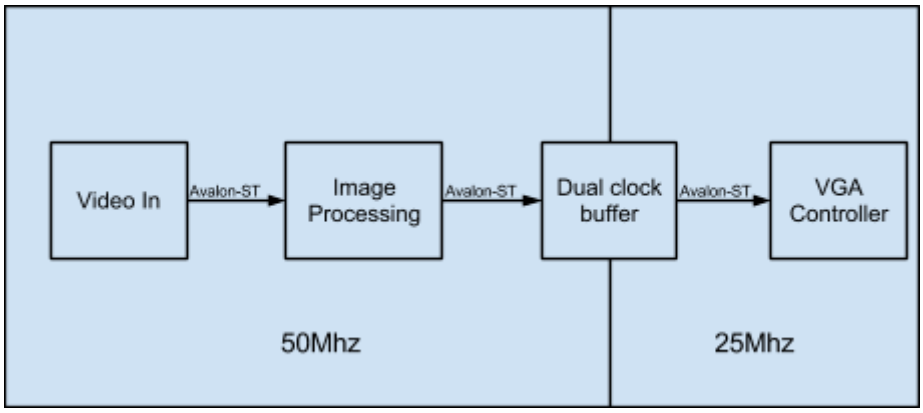


Figure 1: Video Chain

Figure 1 is a simplified version of the video chain used in our system. The Video In component consists of:

- Video Decoder
  - De-interlaces incoming video signal
- Video chroma resampler
  - Resamples colour channels from 4:2:2 YCbCr to 4:4:4 YCbCr
- Video clipper
  - Clips video resolution from 720x244 to 640x240
- Video scaler
  - Scales video from 640x240 resolution to 320x240
- Video DMA controller
  - Two of these units take care of moving the incoming video data to and from the SRAM
- Video scaler
  - Scales our video up from 320x240 to 640x480
- Video RGB resampler
  - Converts colour space from 4:4:4 YCbCr to RGB

Note: The first video scaler is needed to fit the video frames into the 512KB SRAM.

The DE2 system clock operates at 50 MHz and the VGA controller requires a 25 MHz clock. The dual clock buffer allows the data to cross the clock domains and has a built in buffer so that there are no dropped frames.

The VGA Controller IP core takes care of generating synchronization signals and outputs the video data to the VGA monitor.

Image Processing Block

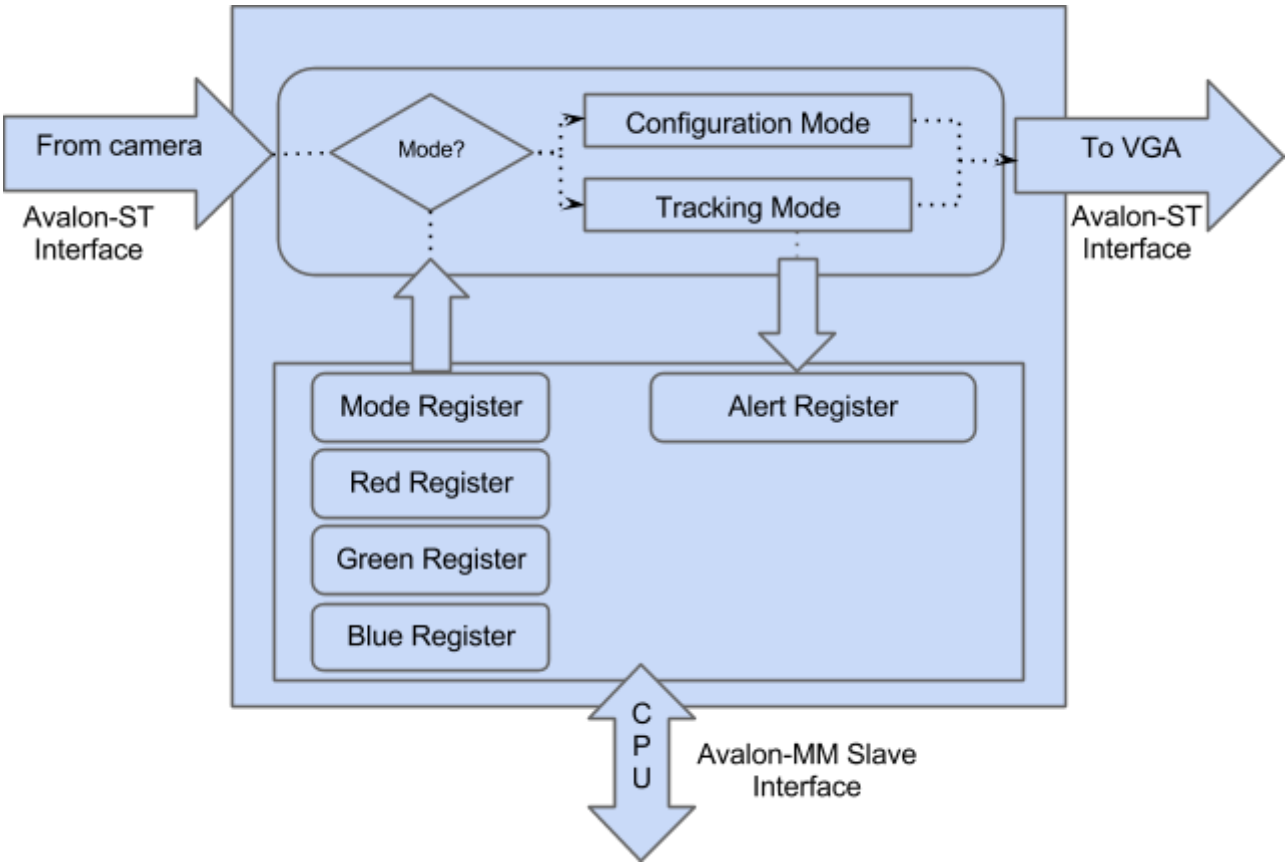


Figure 2: Image Processing Block

The image processing block is a custom coded VHDL module that adheres to the Altera SOPC naming guidelines to make it SOPC Builder friendly. This allows the compiled module to be connected directly to the other Altera UP IP cores that are used in our system.

The image processing block implements three separate interfaces: two Avalon-ST interfaces; one for video in and one for video out. As well as an Avalon-MM slave interface to allow the software component to read and write the five registers.

The component has five separate registers, one to control the mode of operation, three to hold the RGB threshold values and one for the alert condition. Each register is 32 bits wide to simplify addressing them from software.

There are three separate VHDL processes which implement all of the functionality required. There is a read and write process which handle reading and writing to and from our five registers as well as a main process which is responsible for implementing the image processing.

In the main process there are two separate modes of operation: configuration mode and tracking mode. Figure 3 shows the simplified flowchart for our main process.

Figure 3: Hardware main process flowchart

Configuration Mode

In configuration mode every pixel which satisfies the criteria of the RGB thresholds is replaced with a white pixel and output to the VGA monitor. The user can then make changes to the three threshold values to enhance the marker tracking.

Tracking Mode

In tracking mode, the first pixel which meets the criteria in each active region is marked as the top left corner of the target and a red marker is drawn on screen at that location. For the

purposes of tracking green markers, the criteria is red and blue values less than the red and blue thresholds, respectively. However, the green value is greater than the green threshold. The criteria is designed like this so it is easier to identify the green pixels. If the deviation between the y coordinates of each marker is greater than the allowed distance, an alert condition is written to the alert register to be read by the software. Figures 3 to 5 show the process of scanning each pixel and saving the position of the first criteria meeting pixel in each active region so that the system can calculate the difference in their y coordinates. The green squares in figures 4 to 6 represent the targets being tracked.

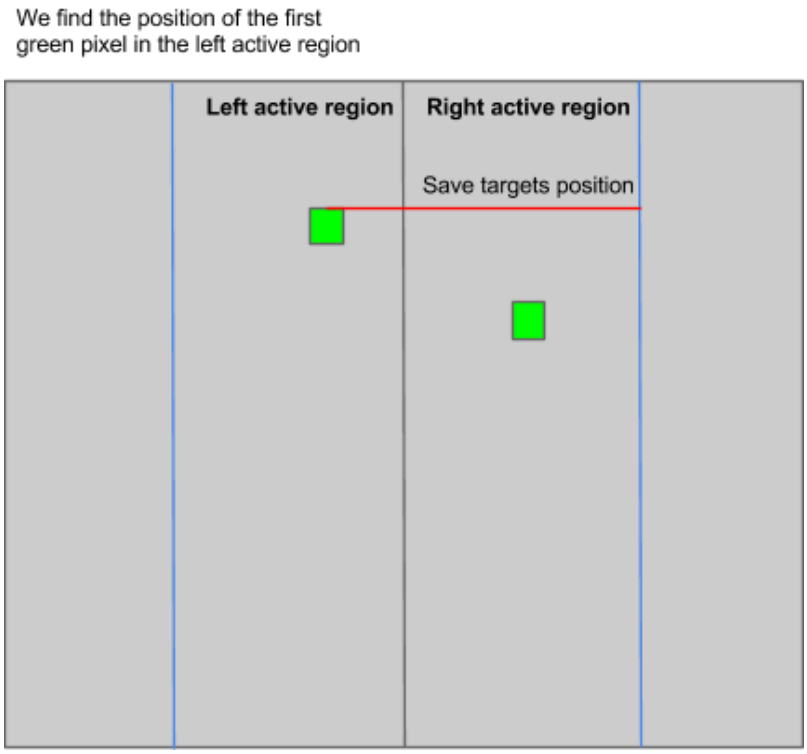


Figure 4: Locating first green pixel in left region

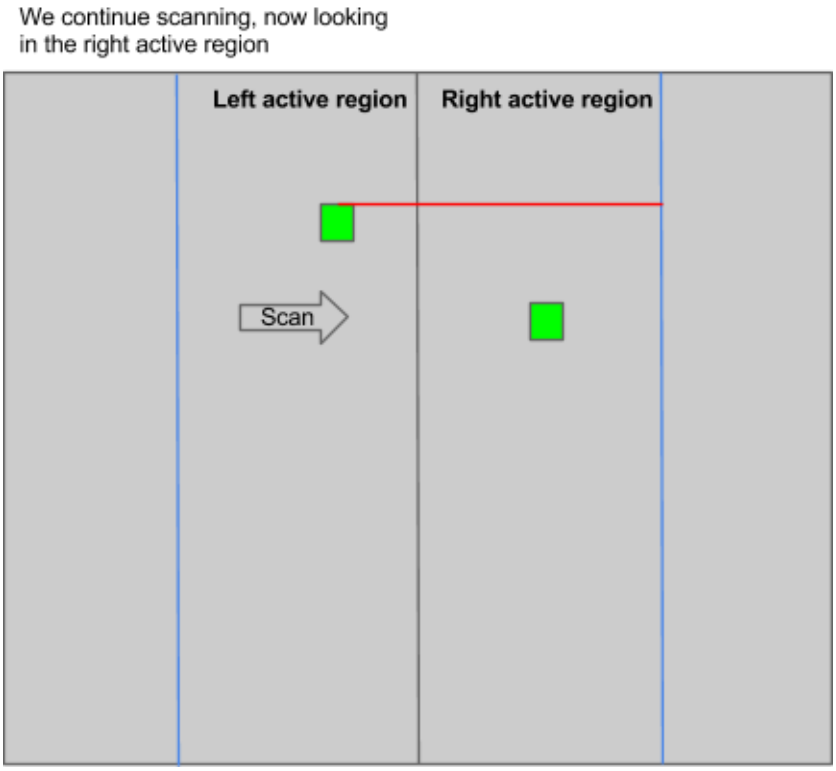


Figure 5: Locating first green pixel in right region

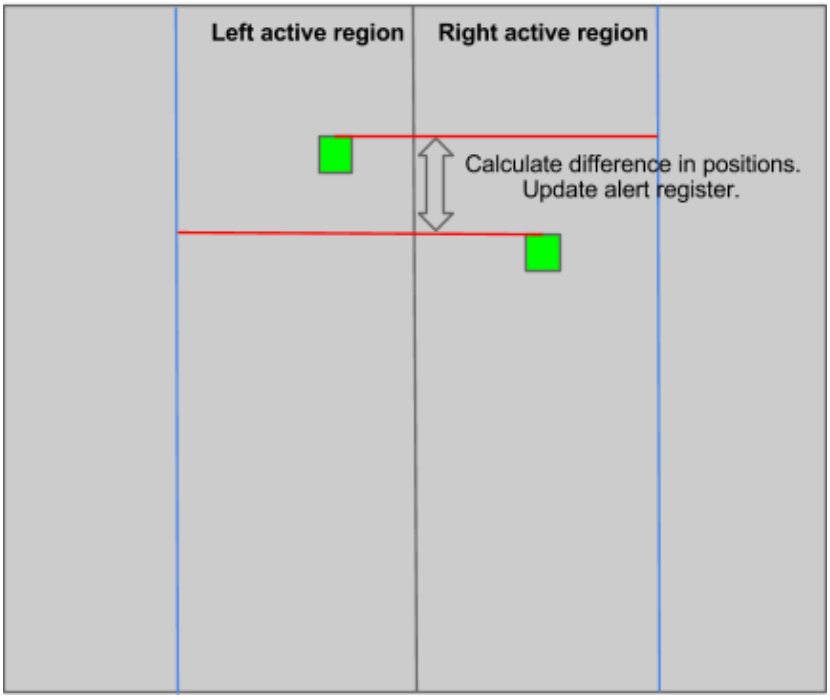


Figure 6: Calculating the difference in Y coordinates

Software Design

The software component consists of a single µC/OS II task. Since the inputs to the system are slow, (users pressing buttons or flipping switches), a single polling task is used. This allows the complexity of the software to be reduced. The flow chart of this task is shown in Figure 7.

The task is responsible for the following:

1 Initialize the LCD, audio samples and RGB threshold values.
2 Allow the user to update RGB threshold values.
3 Play an audio sample when an alert condition has been recognized by the VHDL image processing block.

Steps 2 and 3 occur in an endless loop.

Step 1: Initialization

Occurs only once when the DE2 board is powered on. Initialization of the LCD, audio samples and RGB threshold values are straightforward and do not require any further explanation. The red and blue thresholds are initialized to 100 and the green threshold is initialized to 1000. This sets the system to look for pixels less than the red and blue thresholds and green values that are greater than the green threshold. With this initialization, almost all pixels are within range.

Step 2: Updating the RGB threshold values.

Our system uses all four of the push buttons on the DE2 board. Buttons 3, 2 and 1 are mapped to R, G and B respectively and button 0 is used to change the count mode of our system. Button 0 will toggle whether pressing the RGB buttons will increase or decrease their current value.

Since the software is polling the push buttons very quickly, a delay has been created so that the values do not change too quickly. The way the polling issue is dealt with is by checking if the previous button pressed is the same as the current button being pressed. If this is the case we increase a counter, once the counter reaches an appropriate number (in this case the number is 1500; which was determined experimentally) we can then register the button press. By using the counter, the quick polling of the buttons does not cause a rapid change in the value that the button corresponds to.

Step 3: Playing audio sample

This section of the software is also very straightforward. It takes the audio samples created during initialization and puts them in the audio codec output buffer using the provided Altera HAL functions.

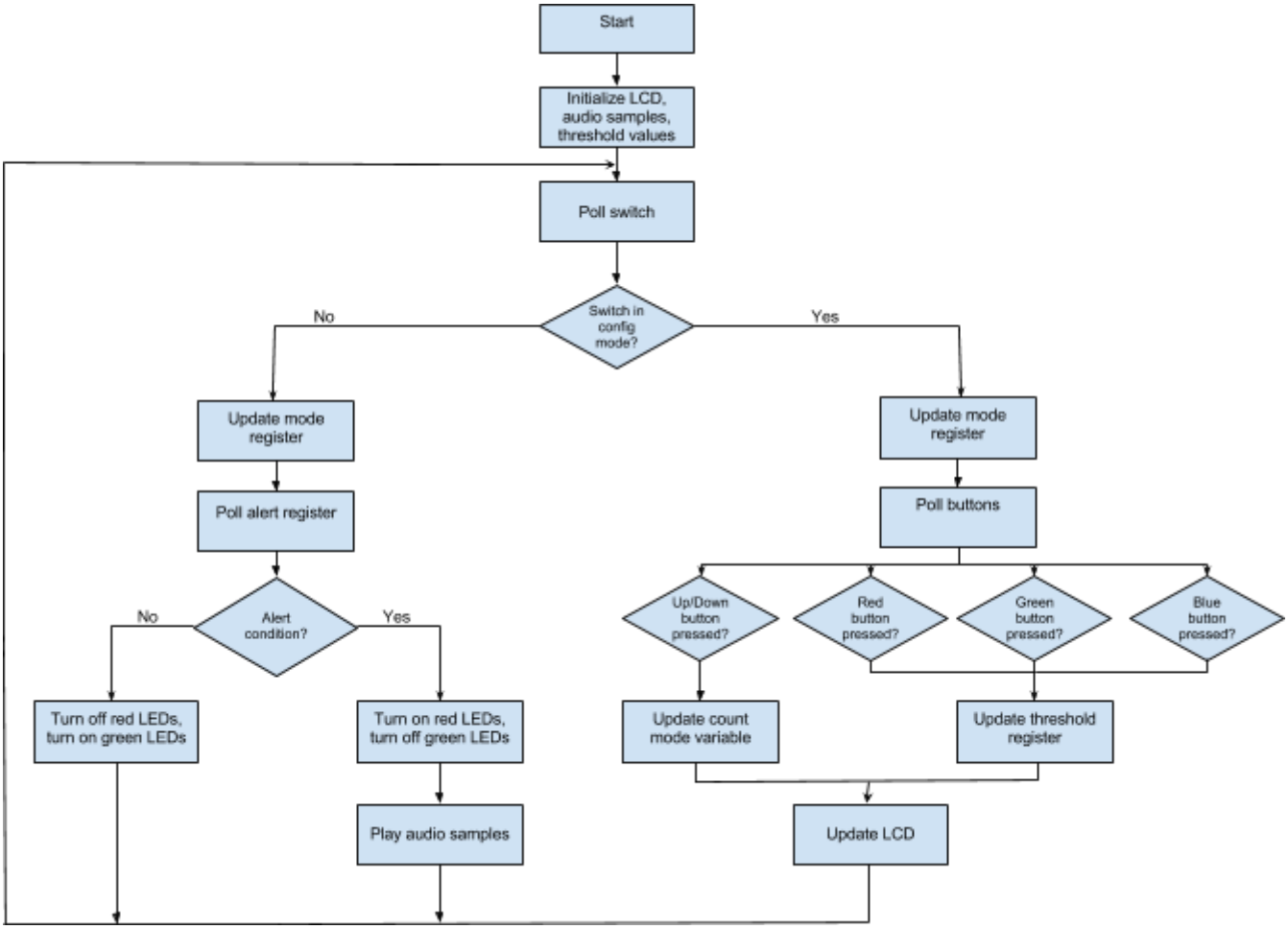


Figure 7: Software Flow Diagram

Description of Operation

1. Turn on system.
2. Audio samples generated, message output to LCD.
3. Once samples have been created, system starts in configuration mode.
    a. The system thresholds are initialized to let a large number of pixel values meet the threshold criteria and hence be output as pure white on the screen.
4. The user then uses key 3 to adjust red, 2 to adjust green and 1 to adjust blue threshold values and key 0 to change the count direction from increasing to decreasing.
    a. The system looks for red and blue values that are less than the number input, and green values that are greater than the number input i.e. r < red threshold, g > green threshold and b < blue threshold. Doing so allows us to target pixels which have strong green characteristics.
    b. As the user presses the keys to update threshold values, the new values are presented on the LCD display.
5. After the user configures the threshold values appropriately they move the mode switch to the ‘Run’ mode.

    a. Move Switch 0 from down to up position, the system is now in Tracking mode.
6. The system is now running the analysis mode and searching each active region for the first pixel which meets the threshold criteria.

The design and operation of our project will begin with the athlete wearing two green coloured markers made from tape. These targets will be placed on the left and right shoulder of the athlete.

The video playback feature will use the VGA output and a VGA compatible monitor.

Parts List

Terasic DE2 Development Board with Altera Cyclone II EP2C35F672C6 FPGA
-To interface the entire system
-Price: $495
-Datasheet: http://www.terasic.com.tw/cgi-bin/page/archive.pl?Language=English&CategoryNo=165&No=30&PartNo=2
-Specs:
-VGA DAC with VGA out connector
-TV Decoder (NTSC/PAL) and TV in connector
-SD card socket
-24-bit CD quality Audio Codec with line-out
-4 Push-button switches
-4 Mbyte Flash Memory
-512 Kbyte SRAM
-8 Mbyte SDRAM

Digital camera with NTSC video output
-To capture the athlete during delivery
-Price: $450
-Specs:
-Nikon Coolpix P5000
http://imaging.nikon.com/lineup/coolpix/p/p5000/spec.htm
-NTSC output
-320x240 pixel video output

Camera tripod
-For stability of the capture video
-Price: $100
-Specs: N/A

Computer speakers
-For playing audio flag
-Price: $20
-Specs:
-3.5mm plug

-120V AC power supply

Computer monitor with VGA input

-For displaying captured video

-Price: $100

-Specs:

-Sun 19” LCD monitor

-640x480 resolution capable

-VGA input

-120V AC power supply

Bill of Materials

Item	**Cost (Approximate)**
Terasic DE2 Development Board	$495
VGA Monitor	$100
VGA Cable	$10
Computer Speakers	$20
NTSC capable Digital Camera	$450
Camera Tripod	$100
TOTAL	**$1175**

Describe and Evaluate Different Available Sources of Reusable Design Units

Avalon to External Bus Bridge:

This *Altera University Program Avalon to External Bus Bridge IP Core*[1] provides us with a simple interface that allows a peripheral device to connect to the Avalon Switch Fabric as a slave device. The bridge creates a connection that is similar to a bus and provides a means to connect more than one slave device to the Avalon fabric. This will enable the project to enhance the external LCD, video input and video and audio output data transfers and simplify the overall design.

Audio:

Use of the *Altera University Program Audio IP Core*[1] will enable the project to utilize audio files stored on the SD card that can be used to signal the athlete when an alarm condition has

occurred. This IP core also requires the *Development Board External Interface* to create the necessary clock signals and *Audio/Video Configuration Core* to automatically set the required registers in the audio controller chip. The SOPC Builder Audio Core configuration allows access to the Audio Out signals with a data width of 16, 20, 24 or 32 bits. For this project, we will be using 16 bits (eight per channel) since the audio files will be short, alarm type sounds that will not require higher bit rates for higher quality playback.

Audio/Video Configuration:

The *Altera University Program Audio/Video Configuration IP Core*[1] that is provided by the Altera University Program interacts with the Audio CODEC and the video input on the DE2 board. This IP core is convenient for configuration and initialization of both the Audio CODEC and Video-In chips.

Chroma Resampler:

The *Altera University Program Chroma Resampler IP Core*[1] is used to convert the video stream from the 8 bit by 2 plane to the 8 bit by 3 plane. This allows the Colour Space Converter and Video In Decoder to communicate with each other.

Colour Space Converter

The *Altera University Program Colour Space Converter IP Core*[1] takes the video stream from the Chroma Resampler and converts it to 24-bit RGB colour space from the YCrCb colour space.

DMA Controller for Video:

The *Altera University Program DMA Controller IP Core*[1] is a controller for the movement of the video streams. The core stores or retrieves the video stream from memory using the Avalon memory-mapped master interface.

Dual Clock FIFO:

The *Altera University Program Dual Clock FIFO IP Core*[1] is a buffer that allows for the transfer of the video stream between the two clock domains. The data that is transferred between the domains uses the First In First Out method of data transfer.

RGB Resampler:

The *Altera University Program RGB Resampler IP Core*[1] can convert the video stream to the different RGB colour space formats, except for the Bayer pattern format.

Scaler:

The *Altera University Program Scaler IP Core*[1] modifies the video stream's resolution by adding or dropping rows or columns of pixels.

VGA Controller:

The *Altera University Program VGA Controller IP Core*[1] is responsible for the generation of the on-board VGA DAC's required timing signals.

Video In Decoder:

The *Altera University Program Video In Decoder IP Core*[1] takes the video stream from the camera and converts it to 720x244 (odd frames) and 720x243 (even frames) with 8 bit colour on 2 planes.

Video Clipper:

The *Altera University Program Video Clipper IP Core*[1] is very similar to the scaler, but it does not modify the pixels. Rather the clipper modifies the video frames in the addition and deletion of rows or columns.

W12 G4 Motion Detection Project:

The Motion Detection project[2] from the previous year will also aid in our development of software that can pick out the targets on the athlete's shoulders and head. It will also provide us with a baseline to determine if our system will be able to provide an in-time feedback solution.

Data Sheet

Avalon to External Bus:

The key signals for the Avalon to External Bus Bridge include *Address*, *WriteData*, *BusEnable*, *ByteEnable*, *RW*, *IRQ*, *Acknowledge* and *ReadData*. The address may be up to 32 bits wide. The *WriteData* and *ReadData* can be 8, 16, 32, 64, or 128 bits depending on the peripheral device. The *BusEnable* is a single bit to indicate that signals are good and data transfer can proceed. The *ByteEnable* is either 1, 2, 4, 8 or 16 bits and each bit indicates if a corresponding byte is to be read or written. The *ByteEnable* signal is active high. *IRQ* is used to interrupt the Nios II processor and *Acknowledge* is used by the peripheral to indicate that the data transfer is complete. The global clock and I2C signalling pins are listed in the table below.

Global Clock I/O Signals	Cyclone II EP2C35F672C6 Pin Number
CLOCK_50	N2

CLOCK_27	D13 but changed to C16
I2C_SCLK	A6
I2C_SDAT	B6

Audio:

The required input signals from the FPGA to the WM8731 CODEC chip are AUD_XCK, AUD_BCLK, AUD_DACDAT, AUD_DACLRCK, I2C_SCLK and I2C_SDAT. The signal from the WM8731 CODEC chip to the FPGA is the I2C_SDAT to provide programming to the registers required to drive the audio chip. The signals from the WM8731 chip to the line out jack is *LHPOUT* for the left channel and *RHPOUT* for the right channel.

Audio Output Signals	Cyclone II EP2C35F672C6 Pin Number
AUD_BCLK	B4
AUD_DACDAT	A4
AUD_DACLRCK	C6
AUD_XCK	A5

VGA:

The required signals from the FPGA to the ADV7123 chip are *VGA_R[0:9]*, *VGA_G[0:9]* and *VGA_B[0:9]* which are the digital outputs from the FPGA. The *TD_RESET*, *VGA_CLOCK*, *VGA_BLANK*, *VGA_SYNC*, *VGA_HS* and *VGA_VS* are signal lines from the FPGA to the ADV7123 chip that provide a reset function, 50MHz clock input, VGA blanking function to blank the output and vertical and horizontal synchronization signals.
The main IC for the video output is the Analog Devices ADV7123. The main signals for this on-board chip are the three, 10 bit input busses *R[0:9]*, *G[0:9]* and *B[0:9]* which receive the incoming video data from the FPGA. For the digital inputs, V_{IH} is minimum 2V and V_{IL} is maximum 0.8V. The analog outputs, *IOR*, *IOG*, and *IOB* are then used to drive the VGA display with *VGA_HS* and *VGA_VS* providing the synchronizing signals. The output current is between 2mA and 26.5mA. The chip voltage requirements are +5V/+3.3V.

VGA Output Signals	Cyclone II EP2C35F672C6 Pin Number
VGA_CLK	B8
VGA_BLANK	D6
VGA_SYNC	B7

VGA_VS	D8
VGA_HS	A7
VGA_R0	C8
VGA_R1	F10
VGA_R2	G10
VGA_R3	D9
VGA_R4	C9
VGA_R5	A8
VGA_R6	H11
VGA_R7	H12
VGA_R8	F11
VGA_R9	E10
VGA_G0	B9
VGA_G1	A9
VGA_G2	C10
VGA_G3	D10
VGA_G4	B10
VGA_G5	A10
VGA_G6	G11
VGA_G7	D11
VGA_G8	E12
VGA_G9	D12
VGA_B0	J13
VGA_B1	J14
VGA_B2	F12
VGA_B3	G12
VGA_B4	J10

VGA_B5	J11
VGA_B6	C11
VGA_B7	B11
VGA_B8	C12
VGA_B9	B12

The main IC for the video input is the Analog Devices ADV7181B. The input signals for this on-board chip are the *YPrPb* S-video input signals in analog format and the *I2C_SCLK* and *I2C_SDAT* to provide access to the programmable registers required to drive the ADV7181B chip. The chip requires a 27MHz clock input. The eight bit digital output is then passed into the FPGA through *TD_D[0:7]*. The 27MHz clock is also sent to the FPGA on *TD_CLK27*. The voltage required for the chip is +3.3V and the voltage for the digital outputs are V_{OH} at a minimum of 2.4V and V_{OL} at a maximum of 0.4V. The voltage levels for the digital inputs are V_{IH} at a minimum of 2V and V_{IL} at a maximum of 0.8V.

Video Input Signals	Cyclone II EP2C35F672C6 Pin Number
TD_CLK27	B14
TD_RESET	C4
TD_DATA0	J9
TD_DATA1	E8
TD_DATA2	H8
TD_DATA3	H10
TD_DATA4	G9
TD_DATA5	F9
TD_DATA6	D7
TD_DATA7	C7

Switches/Keys:

The 4 keys on the DE2 are used to change the RGB threshold values and to change the input method to either increase or decrease the RGB values. Switch 0 on the DE2 is used to toggle the system between Configuration mode and Tracking mode.

Input/Output Signals	Cyclone II EP2C35F672C6 Pin Number

KEY[0] Up/Down	G26
KEY[1] Blue Threshold Adjust	N23
KEY[2] Green Threshold Adjust	P23
KEY[3] Red Threshold Adjust	W26
SW[0] Mode Select	N25

LCD:

The LCD module on the DE2 is used to display the RGB threshold values that are entered by the use during the initialization process.

LCD Signals	Cyclone II EP2C35F672C6 Pin Number
LCD_RW	K4
LCD_EN	K3
LCD_RS	K1
LCD_ON	L4
LCD_BLON	K2
LCD_DATA[0]	J1
LCD_DATA[1]	J2
LCD_DATA[2]	H1
LCD_DATA[3]	H2
LCD_DATA[4]	J4
LCD_DATA[5]	J3
LCD_DATA[6]	H4
LCD_DATA[7]	H3

LEDs:

The LEDs on the DE2 are used to distinguish between correct shoulder target positioning with all eight green LEDs being illuminated and incorrect shoulder target positioning with all 18 red LEDs being illuminated.

LED Output	Cyclone II EP2C35F672C6 Pin Number

LEDR[0]	AE23
LEDR[1]	AF23
LEDR[2]	AB21
LEDR[3]	AC22
LEDR[4]	AD22
LEDR[5]	AD23
LEDR[6]	AD21
LEDR[7]	AC21
LEDR[8]	AA14
LEDR[9]	Y13
LEDR[10]	AA13
LEDR[11]	AC14
LEDR[12]	AD15
LEDR[13]	AE15
LEDR[14]	AF13
LEDR[15]	AE13
LEDR[16]	AE12
LEDR[17]	AD12
LEDG[0]	AE22
LEDG[1]	AF22
LEDG[2]	W19
LEDG[3]	V18
LEDG[4]	U17
LEDG[5]	U17
LEDG[6]	AA20
LEDG[7]	Y18
LEDG[8]	Y12

Operating Conditions

The operating conditions for the project are the DE2 board and camera must be able to operate inside a curling rink where the temperature is maintained between one to five degrees Celsius with a relative humidity level of between 60 to 75%.

The project hardware and camera should be setup on the ice surface about four to five feet from the hog line as shown in figure 8 and denoted by the star. This provides enough room for the curler, shown as two green blocks, to slide out to the hog line without worrying about hitting the camera and allows the camera to be able to frame the full width of the house with only a marginal amount of zoom. For examining different slide paths from the hack, the equipment setup may need to be moved so the camera maintains a square perspective to the curler.

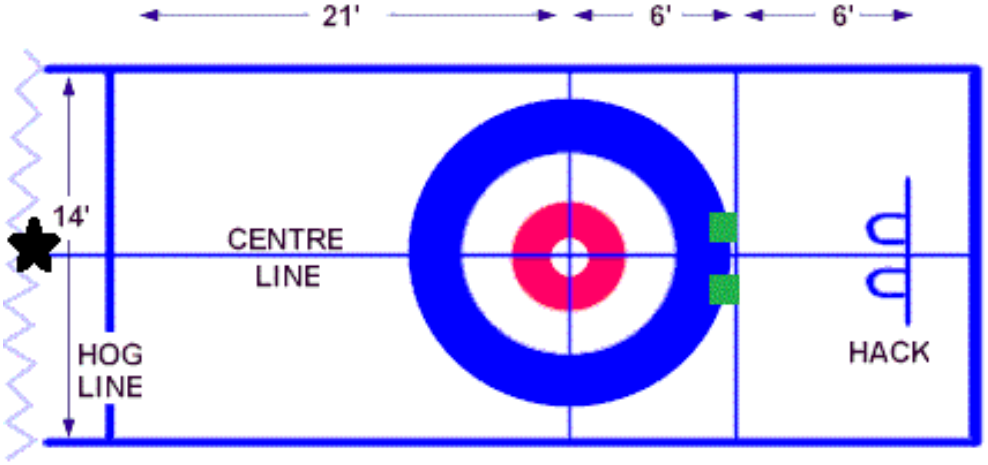


Figure 8: Curling Coach hardware setup location on the ice surface

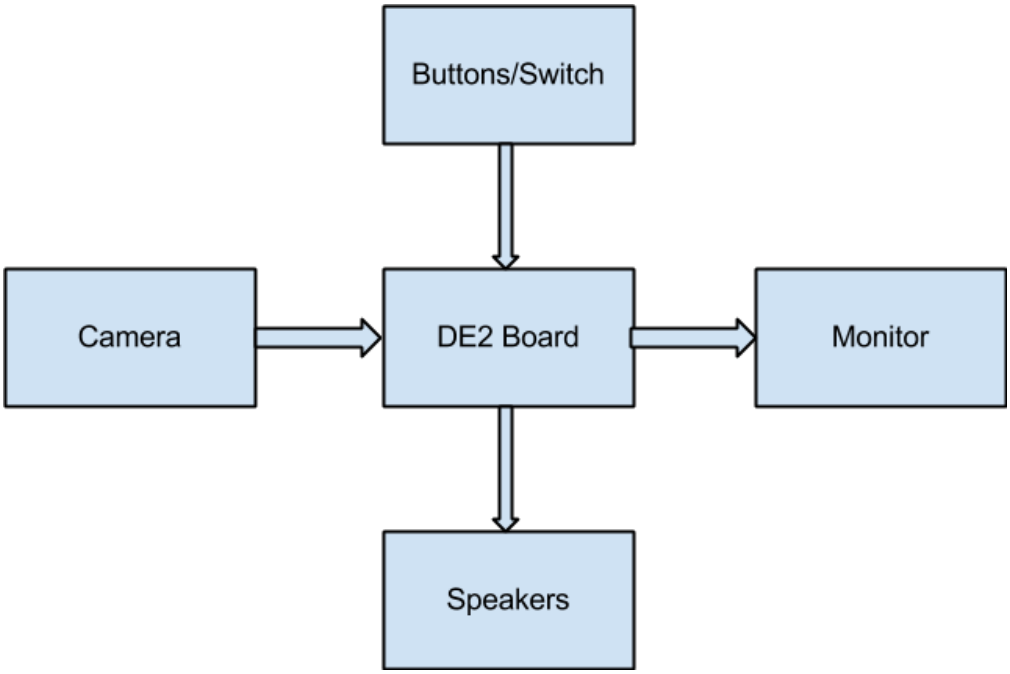


Figure 9: User Perspective Block Diagram

Power Consumption

Mode	Initialization	Configuration	Tracking - Good	Tracking - Error
Current (mA)	415	429	428	447
Voltage (V)	9.04	9.04	9.04	9.04
Power (W)	3.75	3.88	3.87	4.04

Background Reading

Designs that accomplished similar functionality as ours were, and continue to be, invaluable. For the configuration of the TV Decoder chip, video in and video out to the VGA the DE2_TV[3] example project has been very useful.

In the design all of the needed HDL files needed to configure the TV decoder, store the decoded frames and output them to the VGA pins were included and worked well with the Altera-DE2. All that was needed was to add our image processing code into the design.

Originally we had hoped to implement some sort of limb detection to track the movement of the athletes arms and [4] was a good starting place for us to begin reading however the techniques shown in their paper are too computationally expensive for us to use.

Copies of the above readings have been put into the attached zip file ‘Group 15 Background reading’.

Test Plan

Software Testing

Due to the relative simplicity of the software no formal testing was done on it. The software does no dynamic memory allocations so it is impossible that any of the code could contain memory leaks. The software was constructed in an iterative manner that allows acceptance testing to be done as functionality was added.

Hardware Testing

The only hardware testing that was needed for the project was to test the Image Processing component. All the other components were provided by Altera. To ensure that the provided components worked as intended, the video chain was built without the image processing component. Since the system could successfully capture video data and output it to the VGA monitor it was verified that all the Altera components worked as expected.

For the custom Image Processing component, a testbench was created that tested the several different aspects of its functionality. Figures 10, 11 and 13 through 15 show the results of these tests.

The testbench exercised each of the following three processes in the Image Processing VHDL block:

1 Write process
2 Read process
3 Main process

The write process is triggered on the rising edge of the Avalon-MM Slave *write* signal. When the signal is asserted the data on the Avalon-MM Slave *writedata* bus is written to the memory location that is on the Avalon-MM Slave *address* bus. In this case, each address represents one of the RGB registers. As seen in Figure 10, at the first rising edge of *write* the value on *writedata* is 0xFF000000 and the value of 'red register' goes from XXXXXXXX to 0xFF000000.

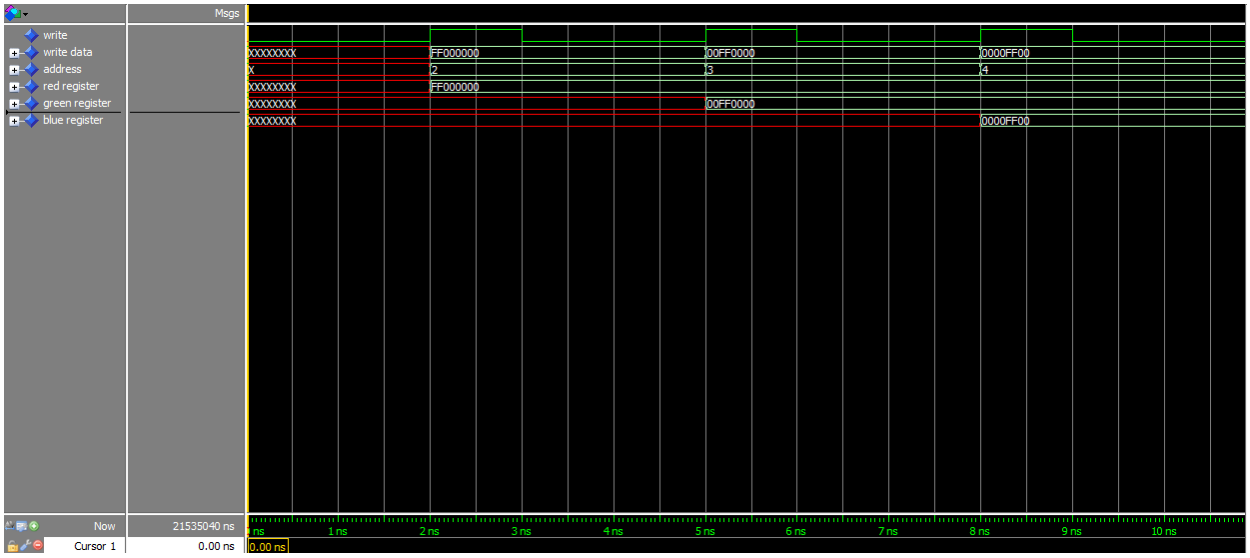


Figure 10: Write process testbench

The read process takes the values in the RGB registers and puts their value onto the Avalon-MM Slave *readdata* bus depending on the address on the *address* bus. Figure 11 shows the testbench signals verifying correct operation. At the rising edge of each *read* signal, the value of the appropriate register is placed on to the *readdata* bus.

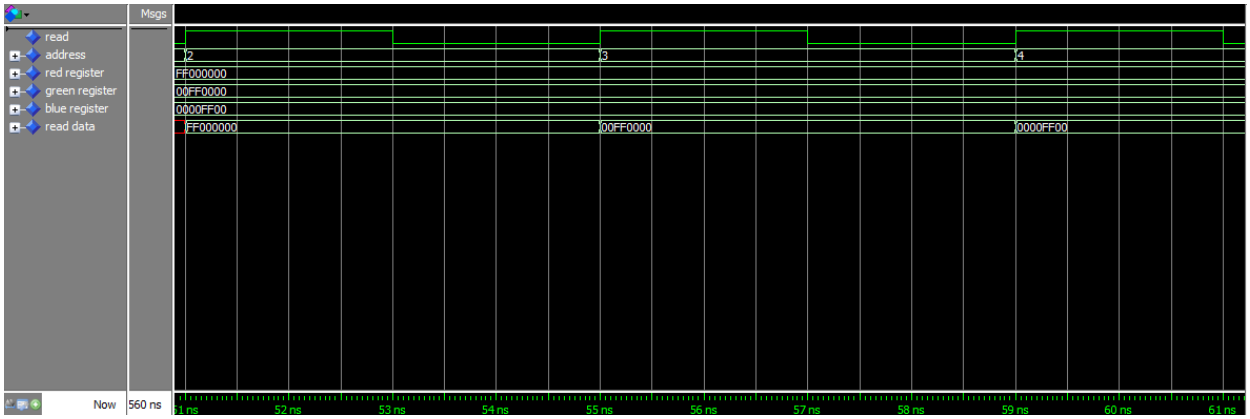


Figure 11: Read process testbench

The main process has quite a bit of functionality that needs to be tested. During development it was difficult to determine if the image processing block was keeping the correct pixel coordinates. To ensure that the correct x and y coordinates were tracked a testbench was created. The testing of the remainder of the system, such as tracking the green targets, is tested via acceptance testing as this is an easier way to verify functionality.

Testing of the counters essentially ended up verifying that the x and y counters did not exceed their limits. Because the output resolution is 640x480, the x counter ranges from 0 to 639 and the y counter ranges from 0 to 479. Figure 12 shows how the Altera UP video IP cores transmits video data.

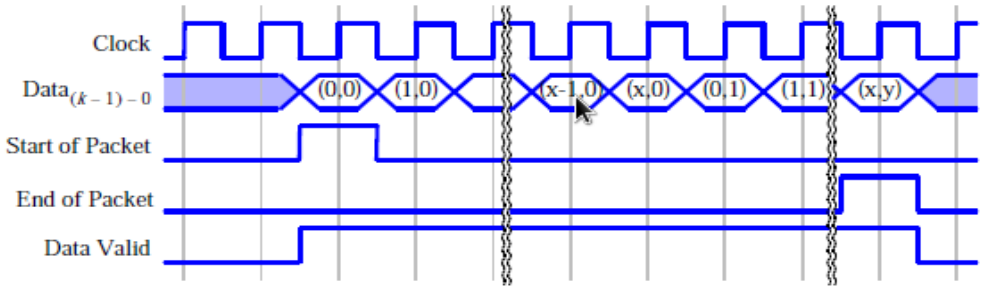


Figure 12: Altera UP Video IP signals

On each rising clock edge, the Data line contains the colour information of the current pixel. Pixels are transmitted in a row major order and each video packet is equivalent to one frame of data. Figures 13 through 15 show the various rollover conditions of the counters and verifies their correct operation.

Figure 13 the initial condition of the system as well as the first few clock pulses. At the rising edge of each clock pulse the current x value in incremented by one.

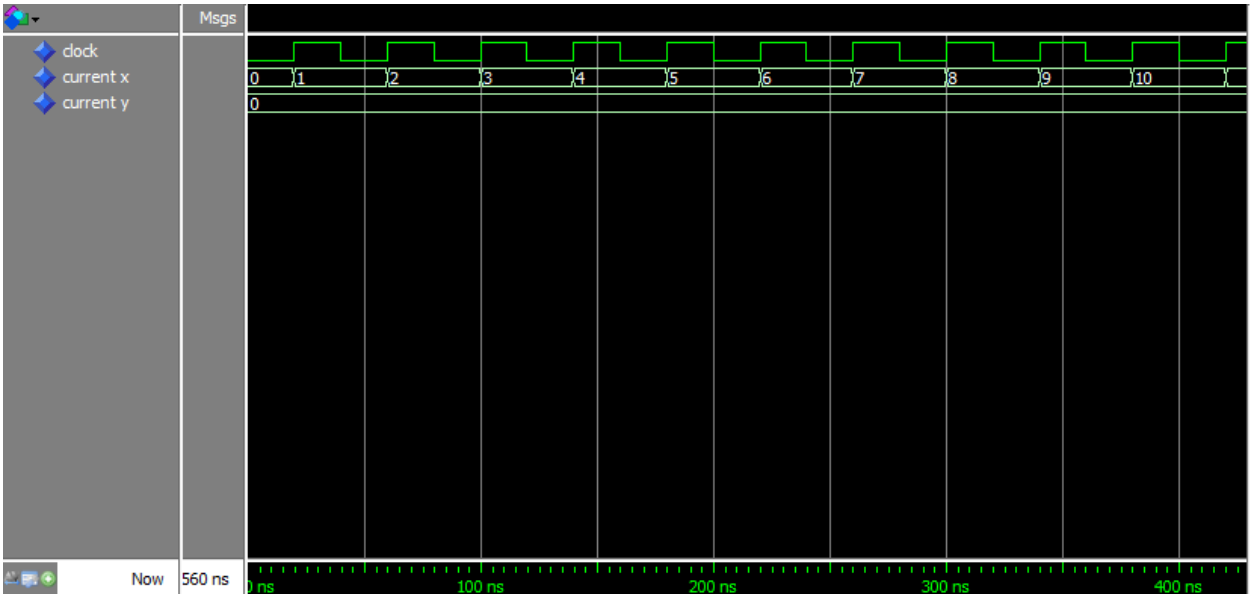


Figure 13: Main process testbench

Figure 14 shows what happens when the current x value reaches its maximum value. Instead of increasing to 640 the x value resets to 0 and the y value is increased by one.

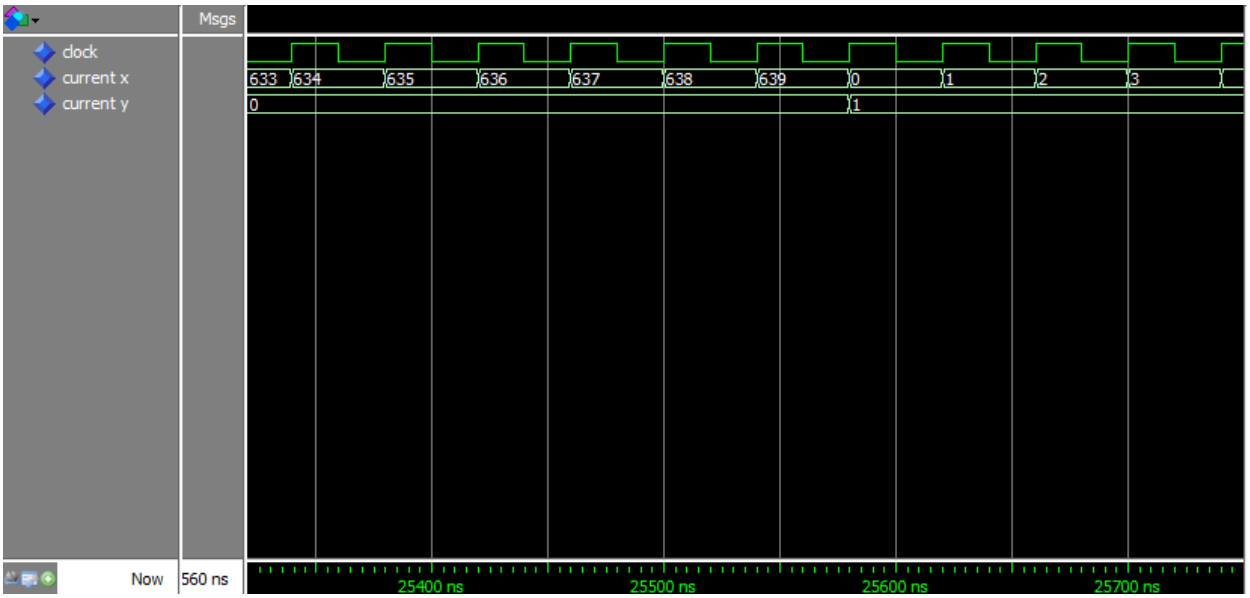


Figure 14: Main process testbench

Figure 15 shows the behaviour when both current x and y reach their maximum values. Both values are reset to 0 and we can begin receiving the next frame of data.

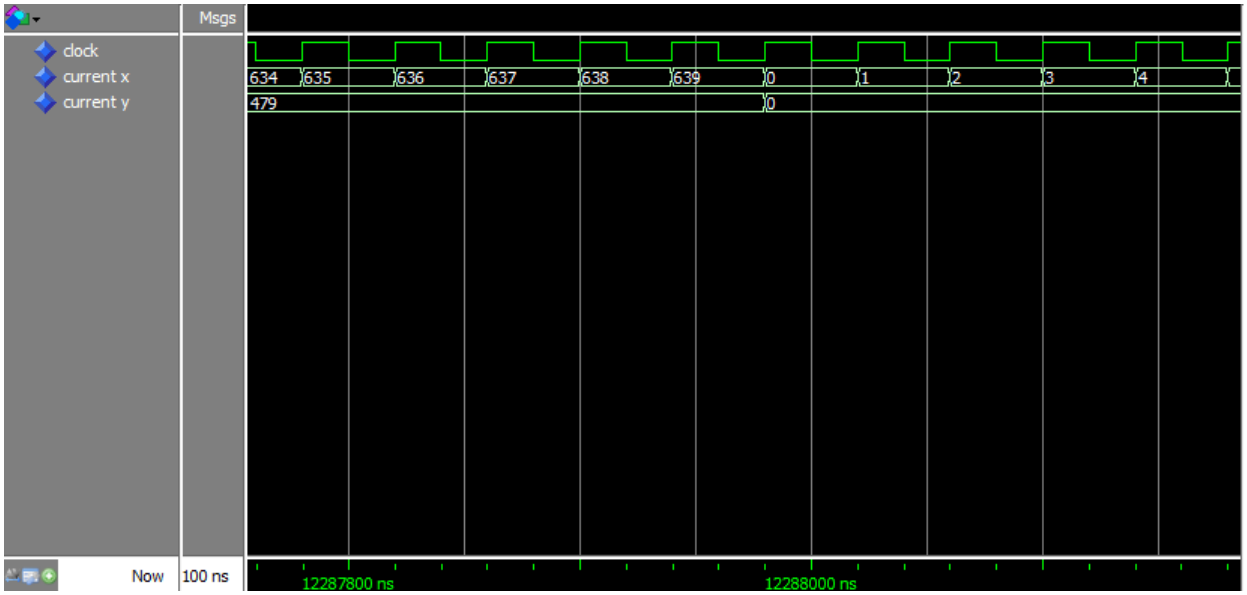


Figure 15: Main process testbench

Results of experiments and characterization

- Tried different hues of green tape to find the best RGB recognition
    - Painters tape was not green enough as it had a blue undertone. The Fisher Scientific cleanroom tape worked great because it was a brighter green and did not have a blue undertone.
- Experimented with the zoom of the camera to ensure visibility of targets throughout the entire delivery process. Figure 16 shows the athlete at the start of their delivery and in the middle of the delivery.

Figure 16: Camera zoom testing

- The image processing algorithm was originally implemented in Java and tested with a single image. From this experiment, it was concluded that simply checking the RGB values of each pixels is sufficient for the purposes of this project.
    - Algorithm takes 0.1s to run as Java executable.
    - Will be reduced to hardware which will reduce time.
    - Can be optimized by reducing the pixels that are operated on.

Figure 17 shows an image before and after running the image processing algorithm. Blue pixels that meet the threshold are changed to yellow.

Figure 17: Image Processing algorithm test

References

[1] Altera, *Altera University Program Computer Organization - IP Cores*, Web. 4 Feb. 13, <http://www.altera.com/education/univ/materials/comp_org/ip-cores/unv-ip-cores.html>.

[2] Rachita Bhatia and Jordan Tymburski, *Motion Detection Final Project 2012*, University of Alberta, ECE Department, Web. 4 Feb. 2013, <http://www.ece.ualberta.ca/~elliott/ece492/projects/2012w/g4_motion_detection/>.

[3] Altera, */opt/altera/DE2_system_v1.6.zip*/*DE2_TV Demonstration,* CD. 12 Feb 2013

[4] Siddiqui, Matheen and Mediono, G´erard, *Robust Real-Time Upper Body Limb Detection and Tracking,* University of Southern California, 2006. Web. 6 Mar 2013, <http://iris.usc.edu/outlines/papers/2006/siddi-medioni-vssn06.pdf>.

Appendix

Quick start manual

1. Connect the DE2 board to any camera via the NTSC connection.
2. Connect the DE2 board to monitor with VGA.
3. Connect the speakers to the DE2 board through the audio port.
4. Connect the power to the DE2 board.
5. Turn on system.
6. Open curlingcoach.qsp and compile design or program directly with curlingcoach.sof.
7. Program the DE2 board using JTAG interface and USB cable.
8. Audio samples are automatically generated and a message displayed on the LCD.
9. Once the audio samples have been created, system starts in configuration mode. The system thresholds are initialized to let a large number of pixel values meet the threshold criteria and hence be output as pure white on the screen.
10. The user then uses push buttons 3 to adjust red, 2 to adjust green and 1 to adjust blue pixel threshold values and button 0 to change the count direction from increasing to decreasing. Change the RGB values until only the pixels on the markers have been changed to a white pixel. The new values are presented on the LCD display as the threshold values are changed.
11. After the user configures the threshold values appropriately they move the mode switch to the 'Run' mode by moving Switch 0 from the down to up position. The system is now in Tracking mode.
12. The system is now running the analysis mode and searching each active region for the first pixel which meets the threshold criteria. When there is a vertical deviation in the markers the green LEDs will turn off and the red LEDs will turn. The audio alert will also be played.

Future Work

Possible options and extensions that could be added to our project in the future are:
Use additional targets and alarms for other misaligned body parts. Specifically, one target placed on the curlers trailing leg would ensure that the curler is sliding from the hack in a straight line towards the camera. If this target is located at any time during the curlers slide an alarm condition would occur. The other main target we would implement would be placed on the rock to ensure the rock is in the correct position, in front of the curler, centered between the two shoulder targets.
For the audio alarm, we would also look into the ability to store pre-recorded verbal cues in flash memory or on the SD card. This would benefit the curler so that when an alarm condition occurs, the curler's attention could be drawn to that specific target area (shoulders, rock or

trailing foot). This would allow the curler to be able to sense what is happening to cause the alarm faster and correct the errant action.
Creation of a simple bar with larger LED lights to indicate that a fault has occurred or if the curler is in the correct position. This addition would aid the curler as they would be able to see and hear if an alarm has been triggered rather than relying on the system operator to relay the error condition based on the DE2 LED output.
More research into the signalling and correct usage of the LCD module would also benefit the project since a touch screen control would make the system more intuitive to use and also removes the requirement of the large, bulky VGA monitor for video output.
The final change that we would implement is the SD card functionality for video storage for future playback and analysis. This would complement the system as a whole and make the project more marketable.

Source Code

toplevel.vhd

Compiled without errors
The toplevel file connects our SOPC generated system with all of the external pins we need.

ImageProcessing.vhd

Compiled without errors
This code file is the core of the system. It allows the video to have the configuration mode annotations or the tracking mode annotations displayed onto the monitor. This core talks closely with the Altera University Cores.

ImageProcessing_tb.vhd

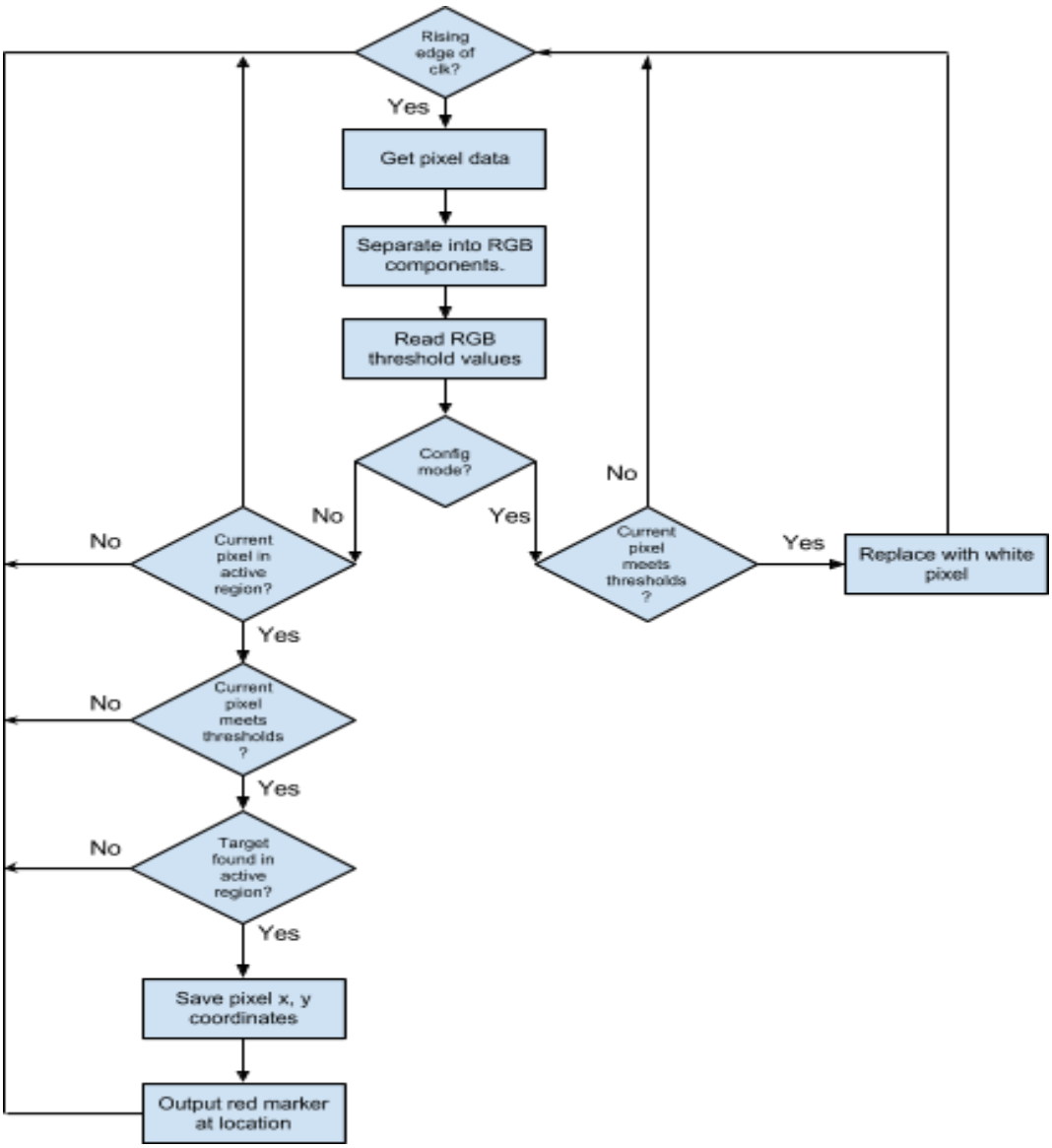
Tested and Passed
This is the testbench that was used to ensure that the system could:

1. Read to and write from the image processing registers.
2. Keep track of the current pixel coordinate.

hello_ucosii.c

Compiled without errors
This µC/OS II task initializes the threshold values and audio sample. It then polls the switches and button for the user input to set the threshold values and moves the system into and out of tracking mode.

Hardware Flow Chart


Rising edge of clk?
Yes
Get pixel data
Separate into RGB components.
Read RGB threshold values
Config mode?
No
Yes
No
Current pixel in active region?
No
Yes
Current pixel meets thresholds ?
Yes
Replace with white pixel
Yes
Current pixel meets thresholds ?
No
Yes
Target found in active region?
No
Yes
Save pixel x, y coordinates
Output red marker at location

Software Flow Chart