Color and Edge Detection Using FPGA Based Smart Camera

Final Report

ENTS 689Y Multimedia Communications over Networks

By

Sakinder Ali

Under the Supervision Of

Dr. Q. Zang

A. JAMES CLARK
SCHOOL OF ENGINEERING

December 20, 2011

Table of Contents

1. Abstract:

Object recognition and tracking with real-time smart camera plays an important role in many modern vision applications such as work force tracking, intelligence surveillance, fire detection and many more. The main motivation of this project is to extract the color object and recognition using the Sobel edge detection and pass the detected signal to wireless channel using Zigbee (802.15.4). Experimental results demonstrated that object detection is very important for selected color tracking in the industrial automation and work places. Computer based Software programming such as java, C++ and C# cannot satisfy the speed in terms of real-time parallel processing of vast amount of video data. Therefore in order to improve the performance of real time video processing, FPGA is an effective device to rely on. This device is capable of real-time parallel processing of vast amounts of video data. The presented results from given design can be used as color and edge detection on a streaming video which is the basic building block necessary for further object detection.

For detection of edge, various detection operators are used. Most of these operators are applied with convolution masks and are based on differential operations.

2. Introduction:

The idea of this project is to design and develop a stand-alone, FPGA-based smart camera that can perform the processing tasks such as Sobel, Prewitt, dilation edge detection, color object detection, emboss effects and output the alarm signal in the case of detection to short-range wireless systems using the zigbee.

With the rapid development of remote sensing camera has become a popular sensor device for monitoring and surveillance systems. It is the first step of video analysis and understanding with remote sensing video especially the detection of object which is necessary step to extract the information from stream of binary raw data from the sensors. The purpose of the detection is to discover the information about the identified object in the presence of the surrounding objects. The sensor device used is a THDB-D5M, which is five mega pixel digital CMOS color image sensor and it contains a power voltage circuit, a low noise amplifier and clock driver circuit. Its flexibility and ease of use makes it a good choice for the system.

The raw data of color video stream and edge detection is very large, so speed of video processing is a difficult problem. Therefore in order to improve the performance of real time video processing FPGA is an effective device to realize real-time parallel processing of vast amounts of video data. Moreover, adopting FPGA as a development platform than PC and other IC is to perform data intensive video processing tasks with real requirements. Therefore, architecture of detection is designed with a FPGA chip called Cyclone II 2C70, and it can process 800 x 480 x 10 RGB color scale video successfully. This device can locate the detected color object and the edge of the gray frame quickly and efficiently.

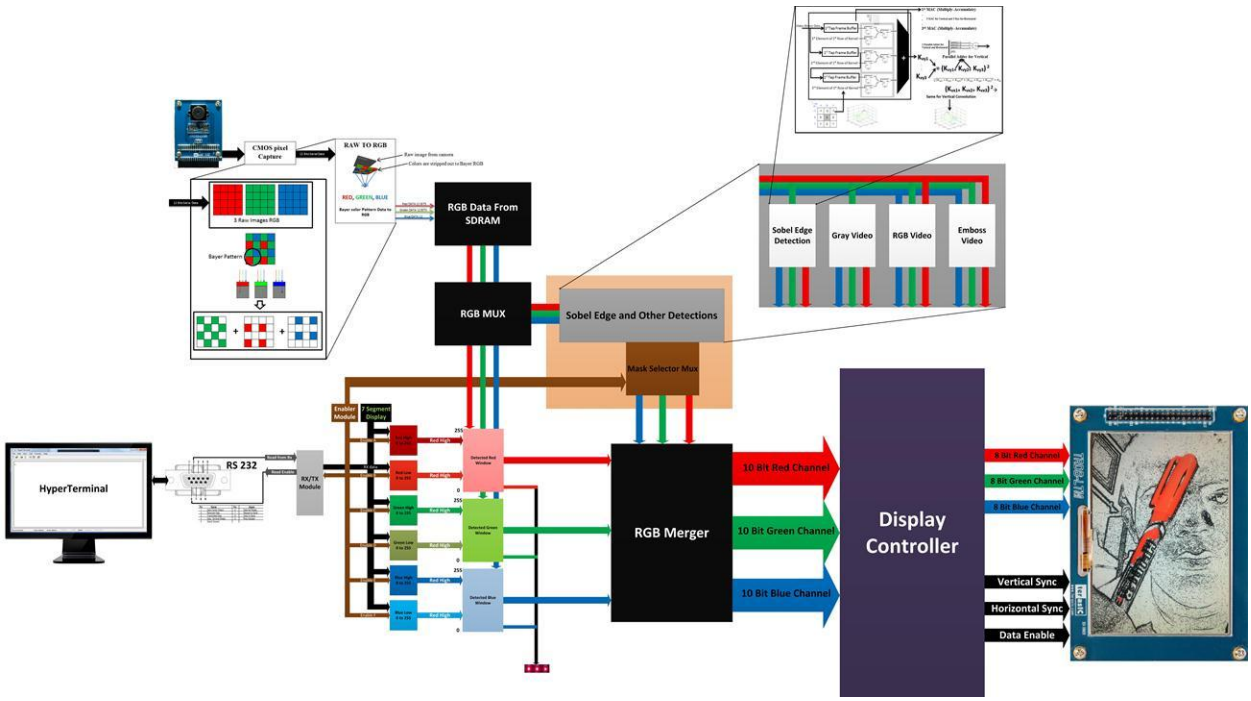

Fig 1: System Architecture block diagram

3. Project description:

The basic principle of this project is shown in the fig 4.The system architecture is divided into main three parts: an image capture unit (D5M camera), an FPGA-based signal processing unit (Cyclone II), and a display unit (LTM).

3.1 CMOS Camera D5M:

The idea behind to use the D5M camera for the FPGA was to have the easy connection of 40 pins between these two devices. This 40 pins connection provide input/output communication using the pixel clock, pixel data (12 bits), power (3.3V),Serial clock, serial data(I^2C),frame valid and line valid.

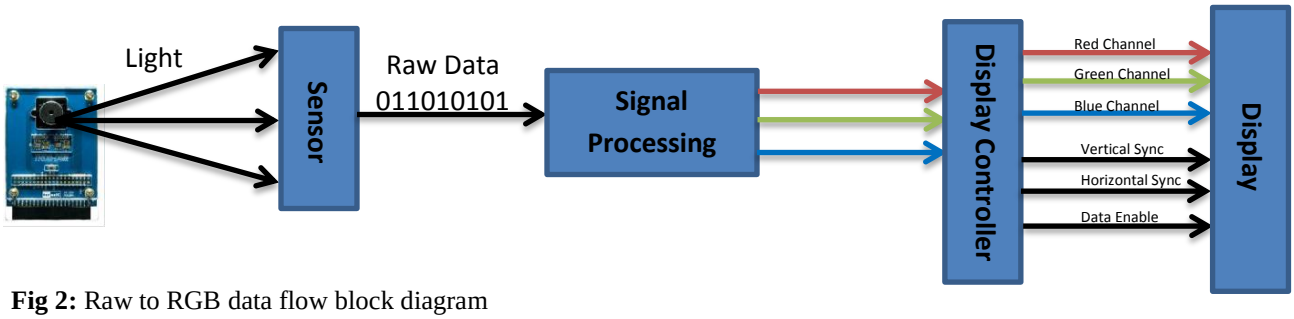

Fig 2: Raw to RGB data flow block diagram

A typical camera system is shown in figure 1. At the heart of every digital camera is an image sensor either CCD image senor or a CMOS sensor. Both types of sensor capture the light through the lens and convert into the electrical signal which is further processed by ADC. Nowadays, majority of sensors consist of high performance ADC converters which are employed to produce the digital output. Most CMOS image sensors have ADC resolutions of 10 to 12 bits. In this project, D5M camera has 12 bits ADC resolution. The D5M pixel array consists a 2752-coloumn by 2004-row matrix. The pixel array is addressed by column and row. The array consists of a 2592-column by 1944-row active region and a border region as shown below.

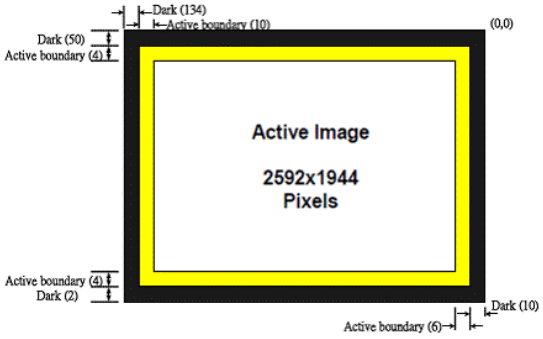

Fig 3: Pixel array description

The output images are divided into frames, which are further divided into lines. The output signals frame valid and lines valid are used to indicate the boundaries between the lines and frames [3]. Pixel clock is used as a clock to latch the data. For each pixel clock cycle, one 12-bit pixel data outputs on the data out pins [3]. When both frame valid and line valid are asserted, the pixel is valid. Pixel clock cycles that occur when frame valid is negated are called vertical blanking. Pixel clock cycles that occur when only line valid is negated are called horizontal blanking. The camera has an array of 255 register, a lot of them are

configurable, and it is possible to set up the operation of the camera by writing to these registers. This communication is performed using a generic two wire serial interface commonly referred to as I^2C[3]. There are two different communication modes, control / configuration mode which included read/write values to the register in the D5M card, and pixel data read out which consists of reading the pixel data from the camera card [3].

3.2 FPGA Cyclone II:

An FPGA is a silicon device that consists of millions of transistors connected to perform logic functions. These logic functions are connected in a way that creates hardware implementation of the applications. FPGAs are truly parallel in nature so different processing operations do not have to compete for the same resources [1].Aircraft, automobiles, radar, missiles, medical and computer are just some of the systems that use FPGAs.

There are two major types of FPGA: SRAM-based and antifuse-based which are supplied by the device vendors (Actel, Altera, Atmel, Lattice, QuickLogic and Xilinx).FPGA have proven faster performance over conventional general purpose processors. Especially for image processing operations, the speed-ups can be increased up to 800 times by exploiting of parallelism and parallel I/O capabilities [2].Edge detection operator like Prewitt is 80 times faster(1.96ms) on Xilinx XV2000E FPGA than on Pentium III (158.08ms) for 512X512 8 bit image[2]. Therefore, considering the parallelism of FPGA, Cyclone II DE2-70 FPGA development board from Altera was chosen for detection application.

3.3 LTM (LCD Touch Panel Module):

The LTM consists of three major components: LCD touch panel module, AD converter, and 40-pin expansion header. All of the interfaces on the LTM are connected to Altera DE2-70 board via the 40-pin expansion connector. The LCD and touch panel module will take the control signals provided directly from FPGA as input and display images on the LCD panel. Finally, the AD converter will convert the coordinates of the touch point to its corresponding digital data and output to the FPGA via the expansion header [101].

3.4 Clock Issue:

Timing plays a very important role in interconnecting the devices and modules. In this project, camera runs at 50 MHz and LTM receiving the data at the clock rate of 27MHz.The clock rate of 27 MHz can be generated from the 50MHz oscillator. To generate an update rate of 27 MHz for a digital line by looping timer function set for 1.85 ticks. FPGA main clock frequency is 50 MHz, so that each clock cycle and unit of time is 20 nanoseconds (ns).
27 MHz =0.03703 us=37.03 ns
37.03 ns/ 20 ns per tick = 1.85 ticks
Since we cannot choose specific partial clock cycles we would have to choose 1 clock cycles which is a delay of 20 ns and corresponds to a frequency of 50 MHz .The next lower frequency we could use is 2 ticks or 25 MHz .Using the 50 MHz FPGA clock, we can only update the digital lines at 25 MHz or 50 MHz, but not in between of these frequencies. Therefore, 27 MHz can't be generated from the 50 MHz clock oscillator but can generate 25MHz clock. The solution is to use external 27 MHz Oscillator for the LTM clock.

3.5 Verilog Implementation:

The software placed in the FPGA is implemented in Verilog. It provides module level simulation and verification which allows is to build the design up from smaller modules. Once the simulation, synthesis and time verification is verified in the Quartus software, then programmer device tool will load the bit file of verilog onto the FPGA using the USB blaster. The process of programming hardware language has to several steps before being loaded onto the FPGA. Figure below shows the basic concept of HDL programming which is used in this project before the project689.bit being loaded to the cyclone II FPGA.

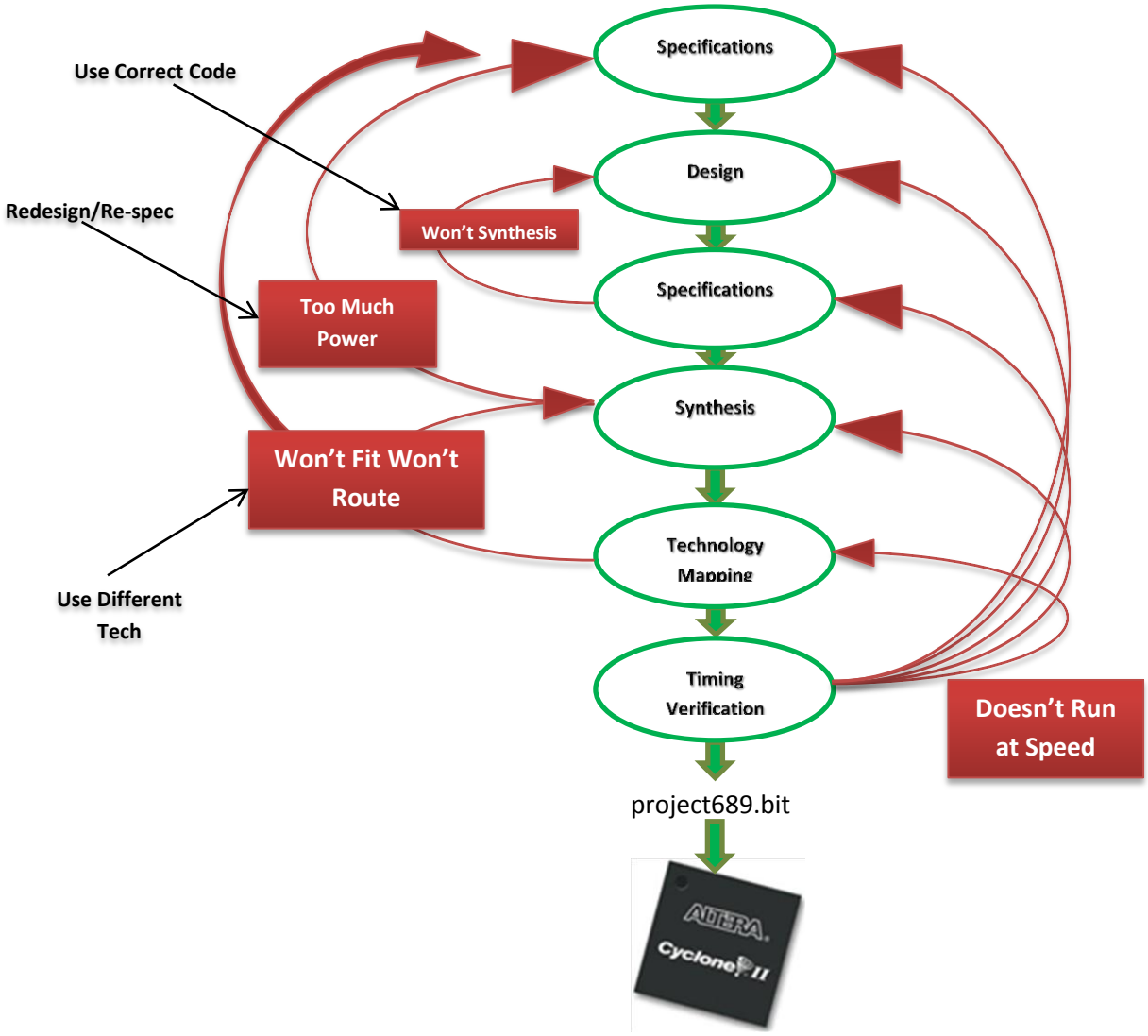

Fig 4: HDL programming process flow system

4. Detailed description of functionality:

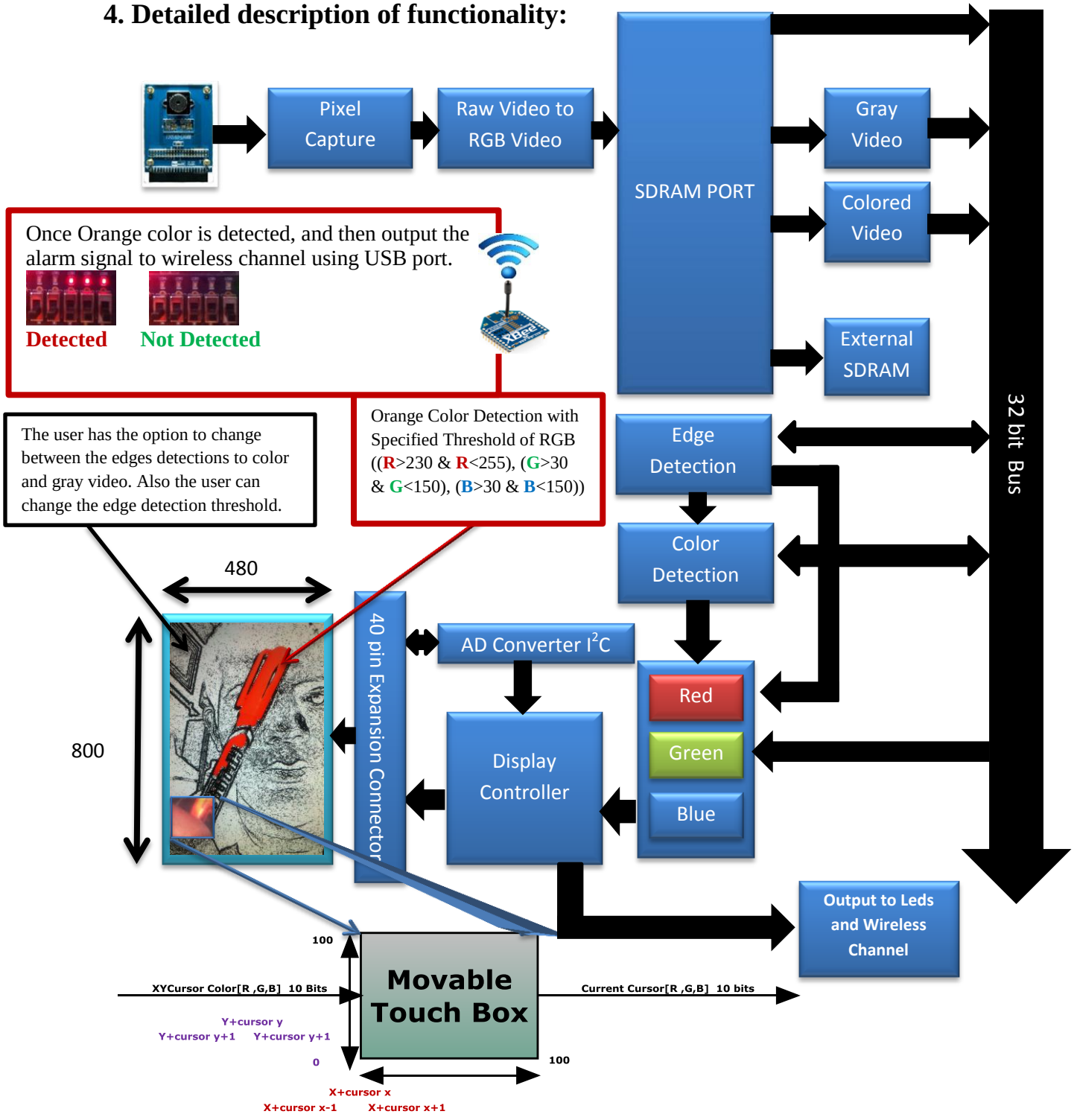

Fig 5: Video Color Object and Edge Detection Top-level Hardware Structure Design of the proposed system

4.1 CMOS pixel Capture:

This is the first module inside the FPGA which communicate with camera. It receives the data of 12 bits per pixel at each clock cycle from the cmos camera when the frame valid and line valid are asserted high. Pixel clock (50MHz) is used to latch the data on the rising edge of the clock. In the case of when clear is set, then it captures on the falling edge of the pixel clock. Vertical blank occurs when frame valid is high. However, horizontal blank only occurs when both frames line valid and frame valid are high. In order to pause, restart, snapshot and change the exposure level of the video, certain registers need to be assigned the hex values. These hex values are given in the datasheet of this camera. Setting these values will enable the features and functionality of the camera. These features were enabled by using the I^2C bus which has two wires SDA and SCL. SDA is the data line. SCL is the clock line which is used to synchronize all data transfers over the I^2C bus. The SCL & SDA lines are connected to the FPGA and the camera on the I^2C bus. Once the registers were assigned the values and valid signal are asserted, the camera output the 12 bits data at the maximum data rate of 96Mp/s. Input clock for the camera is 96MHz which gives the maximum data rate but this data rate should be latched at the 50MHz clock.

Active pixels: 2,592H x 1,944V
Pixel size: 2.2µm x 2.2µm
Color filter array RGB Bayer pattern
Full resolution Programmable up to 15 fps Frame rate
VGA (640 x 480) Programmable up to 70 fps
ADC resolution: 12-bit
Pixel dynamic range: 70.1dB
SNRMAX: 38.1dB
Supply Power Voltage: 3.3V
I/O Voltage: 1.7V～3.1V

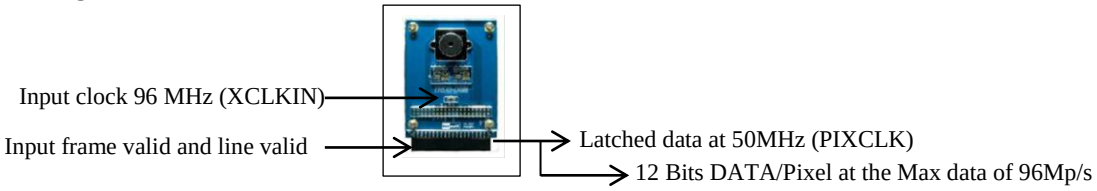

Fig 6: I/O Timing Characteristics

4.2 RAW TO RGB:

This module generates a 36-bit RGB value for three channels by buffering the incoming serial data stream of 12 bits. Serial data enters into the module in a certain order which is stripped out, into three parallel channels only when signal is valid from camera otherwise assigned blank. This order must be maintained during the image processing in the FPGA so that the frame is read properly at the display controller of the LTM. When the frame is displayed on the LTM it would appear as if you are looking at a mirror image.

Serial data is filtered into bayer RGB pattern values. The code shown below assigns serial buffer data into parallel RGB bayer pattern stream.

```
Line Buffer Modula A   (
                       .clken(valid),
                       .clock(clockat50Mhz),
                       .shiftin(inputData),
```

```
                    .shiftout(),
                    .taps2(Data0),
                    .taps1(Data1),
                    .taps0(Data2)
                    );
……….
always@(posedge clockat50Mhz or negedge Reset)
        begin
                Red<=Data1;
                Green<=Data0 +Data1;
                Blue<=Data0;
        end
```

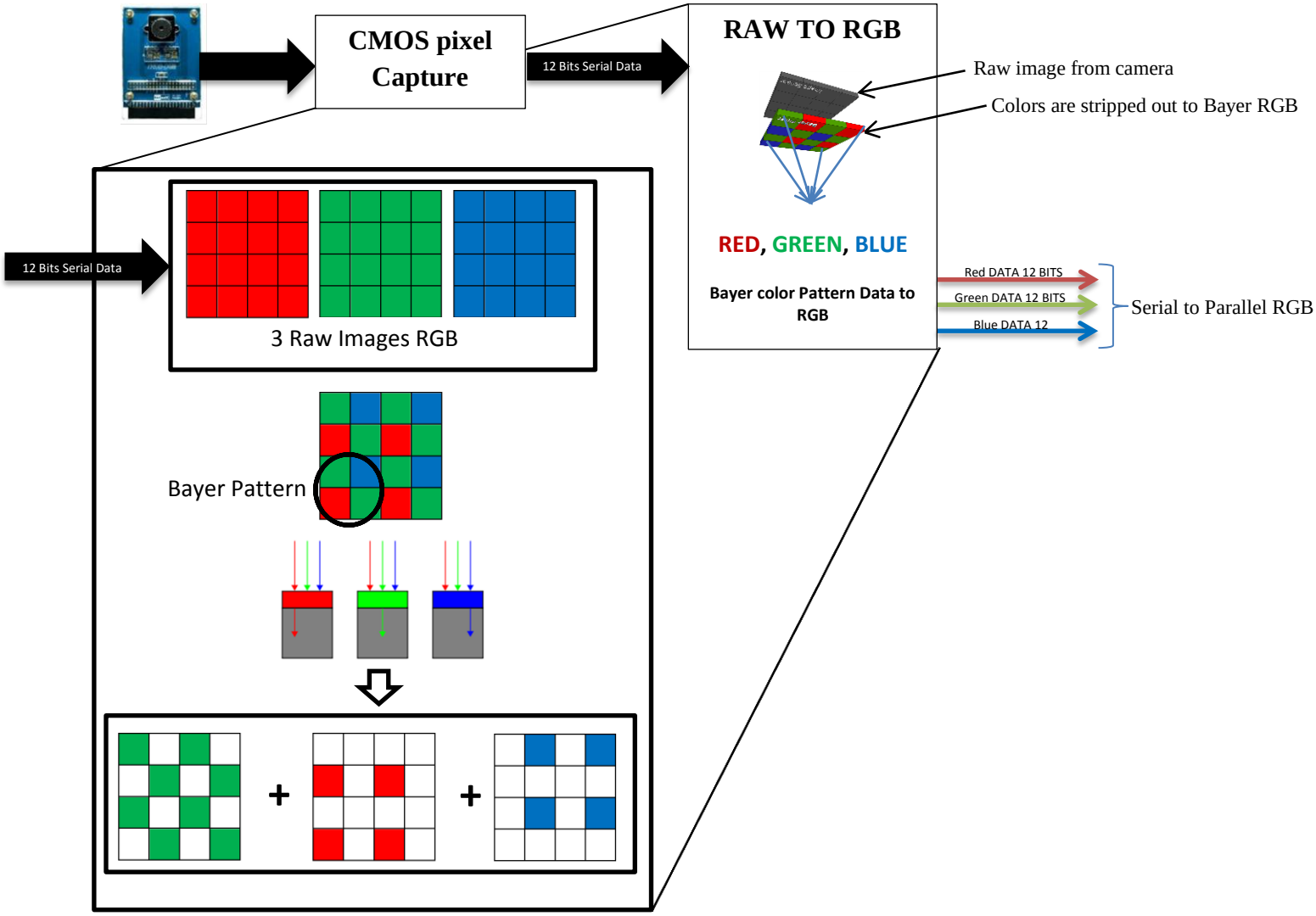

Fig 7: I/O Timing Characteristics

4.3 SDRAM Controller:

To handle high data rate stream and volume, it has to be stored somewhere so image processing can be implemented at slower rate. The solution is to use memory that can support enough capacity and data rate. DE2-70 board provides two SDRAM chips each have a capacity of 256Mbits (32 Mbytes).
Each chip is organized as 4M x 16 bits x 4 banks. All of the signals, except the clock, can be provided by the SDRAM Controller. SDRAM controller is the most important module of this project. Since it handle to write and read from SDRAM and it operate using 4 FIFO ports sides. Two fifo ports are used to write into SDRAM and other two fifo ports are used read from SDRAM. All signals needed to communicate with a SDRAM controller used for this project are shown in Figure 7.RGB data enters into two write fifo ports from left side of the SDRAM controller at 50MHz clock rate and then stored into the SDRAM chip. To read from SDRAM chip, two other fifo ports used to read side1 and side 2 at 27 MHz clock rate.

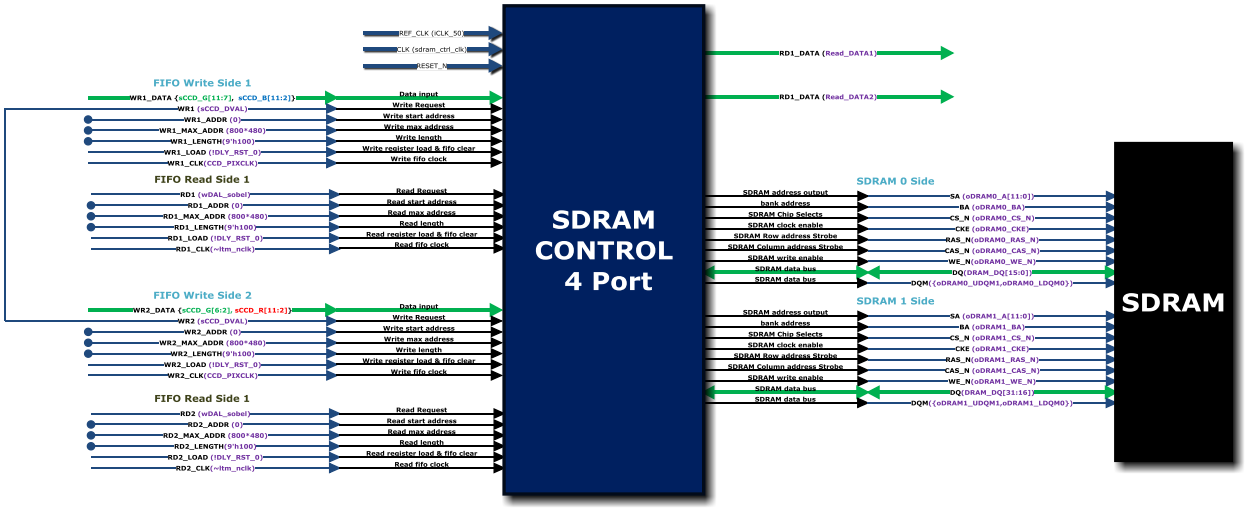

Fig 8: I/O of SDRAM controller

4.4 RGB Video:

In this module, RGB data interface with the display controller module without implementing any image processing. Normal RGB video is displayed once the switches [SW#13→High, SW#14→Low, SW#15→Low, SW#16→Low] are selected.

4.5 Gray Scale Video:

In this module, RGB data is converted into the Grayscale by using the following formula:
GrayVideo= (Red Channel>>2) + (Green Channel [8:1]>>1) + (Blue Channel >>2)
Red and blue Channel are shifted right by 2.In decimal; it's dividing the value by 4, for example 10100 (20 in decimal) in binary is shifted right twice gives us 101 (5 in decimal) so 20/4=5.Whereas green channel is shifted right once. Grayscale module also interface with the Display controller once the switches [SW#13→High, SW#14→Low, SW#15→Low, SW#16→High] are selected.

4.6 Sobel Filter:

For the Sobel edge detection algorithm, two sobel filters are used. There are two distinct filters, one which computes vertical edge left and one for horizontal edge right. The filters are convolved with the entire input 3-line buffer which holds the pixel information. The vertical edge component is calculated with vertical filter K_y into input the first tap1 of 8 bits and then tap2 and tap3 sum to 3 $K_y$taps(3 elements of 1st rows of filter).The Same calculation is applied to horizontal filter 3 columns K_x gives the output of sum 3 $K_x$taps(3 elements of 1st column of filter). The resulted sum 3 $K_y$taps gives us the final sum of Vertical $Kv_{y1,2,3}$ and 3 sum for horizontal $Kv_{x1,2,3}$ .Final output of vertical and horizontal is given by R_{xy} which is the square root of sum Kv_y and Kv_x. $R_{xy} = \sqrt[2]{(K_{vy1} + K_{vy2} + K_{vy3})^2 + (K_{vy1} + K_{vy2} + K_{vy3})^2}$. The Figure 3 below illustrates the concept of Sobel Filter for Video Edge detection.

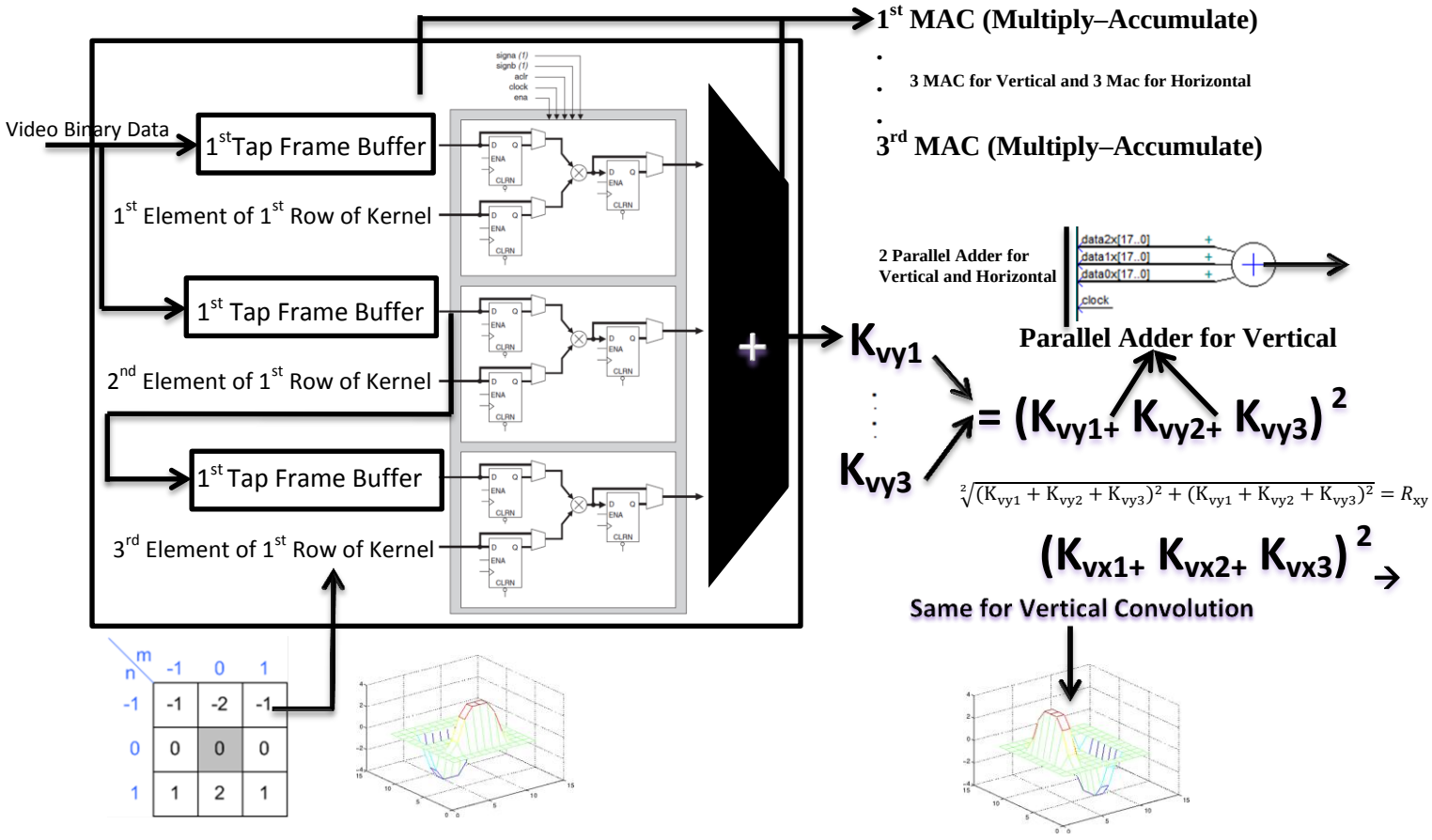

Fig 9: 3-Line buffer connecting 3x3 Filter for vertical Sobel Edge Detection and output from the Mac

Three lines video buffer have used for a filter with 3x3 kernel for both vertical and horizontal edge detection .Then the video buffer enters the filter in a parallel form. Each 3 t-taps module takes the data taps from the line buffer with a rate faster than system clock rate by 3 times and multiply them by the 3 filter coefficients. The output from each multiplications process is accumulated in the accumulator. Then finally sum all three 3-Tap filter from the accumulator fed into parallel adder which form one processed pixel. Once the Rxy is calculated the value is compared with threshold which is the user input from the switches (12 to 1).If the pixel output from R_{xy} is greater than the threshold, the pixel is detected as the edge otherwise no output to the RGB channel. The output from the sobel channel is fed into the red, green and blue channel which is further processed for detection in the detection module. Sobel edge detection

module interface with the Display controller and the color detector module through the mux of switches when [SW#13→ Low, SW#14→Low, SW#15→ High] are selected.

4.7 Color detector in sobel edge detection technique:

In this module, 30 bits RGB data from SDRAM is assigned to three parallel signals R_{in}, G_{in} and B_{in}. Depth of each channel from SDRAM is 10 bits wide. Bits near MSB contain more bright pixels value information rather than the bits near LSB which is the dark side of the pixels. If we choose the 3 bytes RGB values (0, 0, 0) give black color pixel. If we choose the RGB values (56, 0, 0) give dark red pixel and finally if we choose the RGB values (255, 0, 0) give us bright red color. Bright red color also requires more bits than dark red color. Therefore, in order to achieve more pixel information and brightness for the color detection, First MSB of 8 bits are selected for each channel. Another reason to select 8 bits per channel is because display module LTM only accepts 8 bits per channel. The selected depth of $R_{in}G_{in}B_{in}$ signals is 24bits from the 30bit RGB data of SDRAM. User can specifies the selected color range window for each channel. Each window has high and low values such that $R_{window}\,G_{window}B_{window}$ is less than the display max bits for each channel and greater than display min bits.

In this project:

$RGB_{DISPLAY\ max} = 255$ (Hex FF)
$RGB_{DISPLAY\ min} = 0$ (Hex 00)

Parameters (High and Low RGB Channels)

$G_{high} = G_{window} < RGB_{DISPLAY\ max}$ & $G_{window} > G_{Low}$
$G_{Low} = G_{window} < G_{high}$ & $G_{window} > RGB_{DISPLAY\ min}$

$R_{high} = R_{window} < RGB_{DISPLAY\ max}$ & $R_{window} > R_{Low}$
$R_{Low} = R_{window} < R_{high}$ & $R_{window} > RGB_{DISPLAY\ min}$

$B_{high} = B_{window} < RGB_{DISPLAY\ max}$ &$B_{window} > B_{Low}$
$B_{Low} = B_{window} < B_{high}$ &$B_{window} > RGB_{DISPLAY\ min}$

For Red Channel:

VGA_Red = **R_{in}** $R_{in} > R_{Low} \,\&\, R_{in} < R_{high},\ G_{in} > G_{Low} \,\&\, G_{in} < G_{high},\ B_{in} > B_{Low} \,\&\, B_{in} < B_{high}$
Sobel Edge *otherwise*

For Green Channel:

VGA_Green = **G_{in}** $R_{in} > R_{Low} \,\&\, R_{in} < R_{high}, Gin > G_{Low} \,\&\, G_{in} < G_{high},\ B_{in} > B_{Low} \,\&\, B_{in} < B_{high}$
Sobel Edge *otherwise*

For Blue Channel:

VGA_Blue = **B_{in}** $R_{in} > R_{Low} \,\&\, R_{in} < R_{high}, G_{in} > G_{Low} \,\&\, G_{in} < G_{high},\ B_{in} > B_{Low} \,\&\, B_{in} < B_{high}$
Sobel Edge *otherwise*

Normal RGB stream is displayed on the screen

$P(RGB_{in})$ =The Whole Frame

Pixels within the Region (Color Detected):

$P((RGB)_{in} \cup (RGB)_{window}) = True \rightarrow (overlap\ (RGB_{out}=RGB_{in}))$

Pixels not within the Region (Sobel Detection):

$P((RGB)_{in} \cap (RGB)_{window}) = False \rightarrow (not\ overlap(RGB_{out}=RGB_{in}))$

Predicate P defines the region which involves RGB pixel values or area feature.

$(RGB)_{window}$ are the parameters defined by the user.

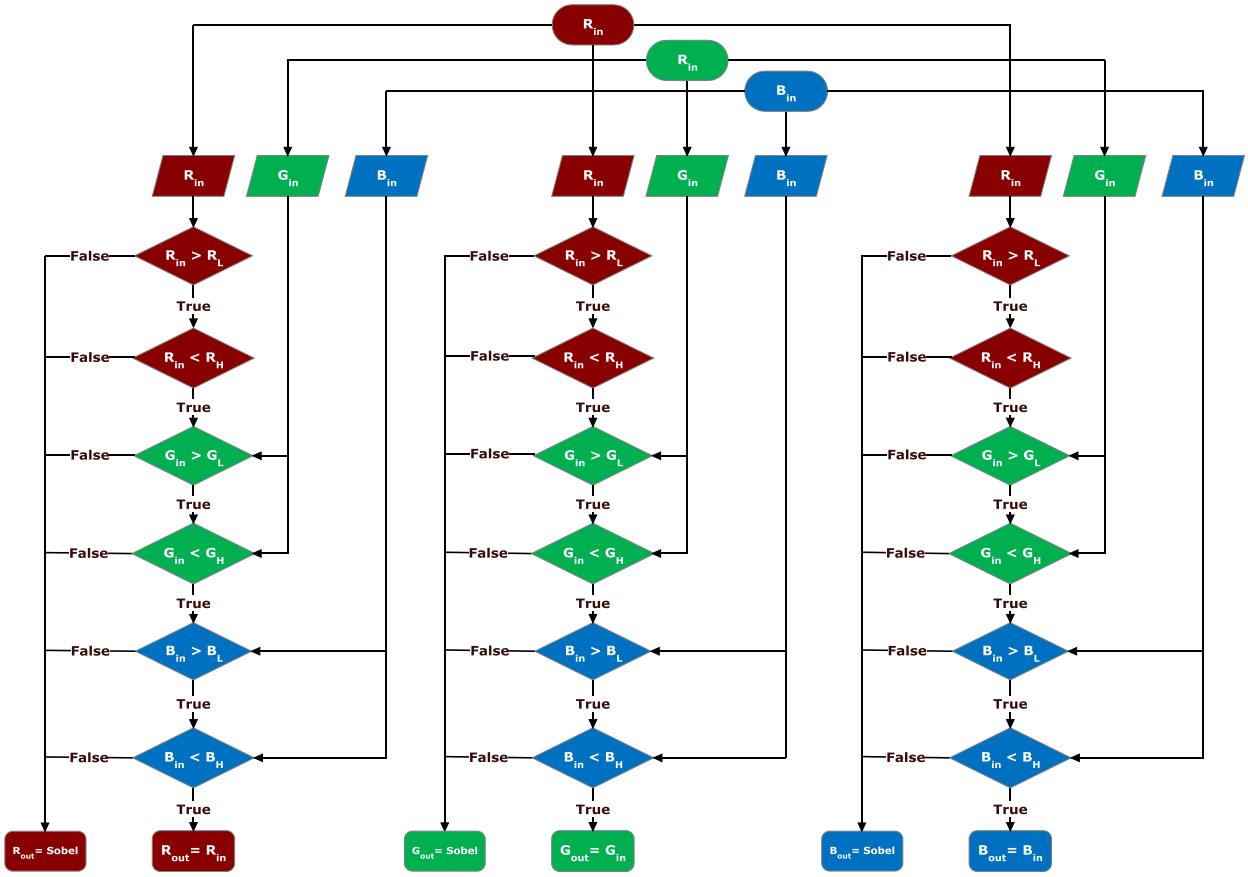

Fig 10: Color and edge detection technique flow diagram

4.8 Orange Color Detector

For Orange color detection, we use the previous model for color detection. If the pixel lies within a predefined color region, then it is classified as color object detection.

Following are the parameters set for orange color window.

Parameter R_{high} = 8'b11111111; //255 FF
Parameter R_{Low} = 8'b11100110; //230 E6
Parameter G_{high} = 8'b10010110; //150 96
Parameter G_{Low} = 8'b00011110; //30 1E
Parameter B_{high} = 8'b10010110; //150 96
Parameter B_{Low} = 8'b00011110; //30 1E

Once the parameters are assigned and pass through different signals, the ended result would be either sobel edge detection or the original orange color of the predefined area of the object. User can even assign its own color to the object .In this module, RGB data from SDRAM interface with the display controller module once the switches [SW#13→High, SW#14→Low, SW#15→ High] are selected.

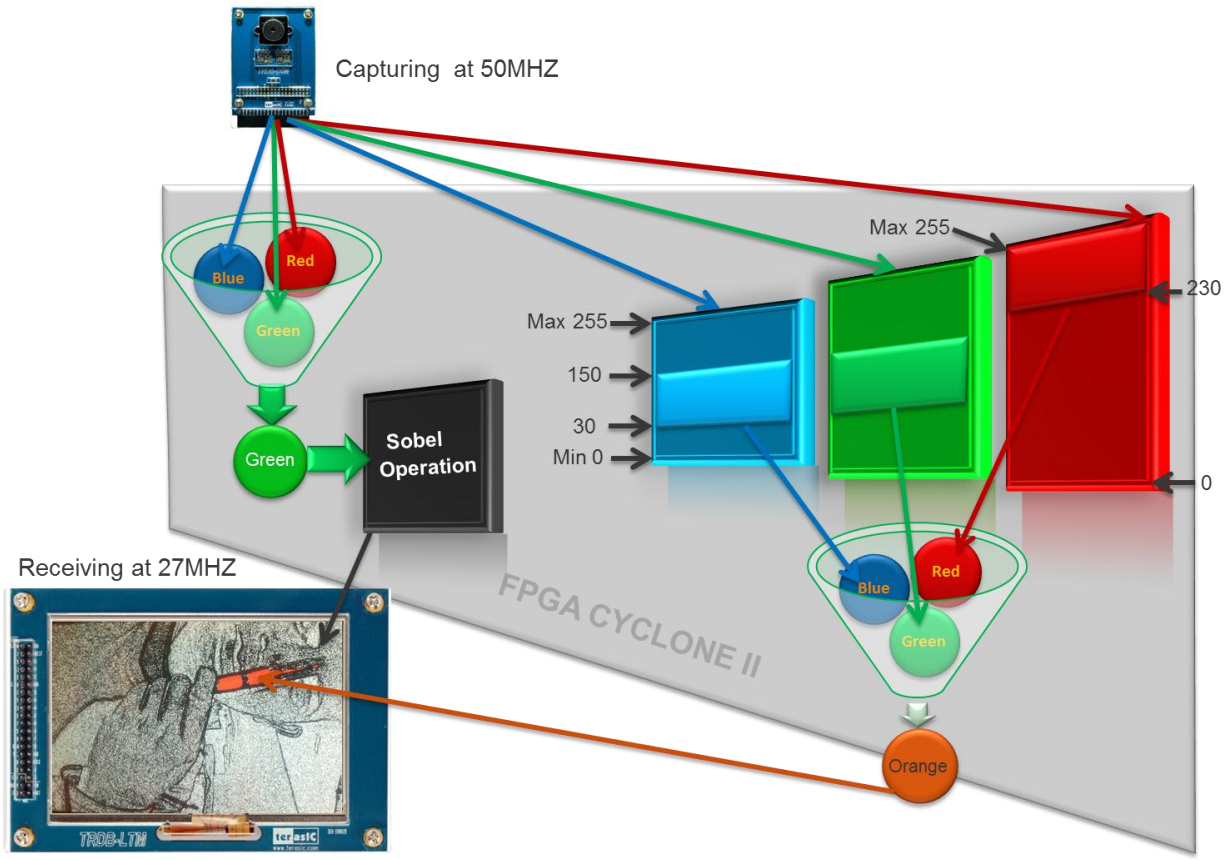

Fig 11: Color and edge detection window selection

4.9 Dilation:

Dithering is used to transform a grayscale video into binary video, while preserving some grayscale information. Each grayscale value of the original video is translated into a pattern of binary values in the output image.

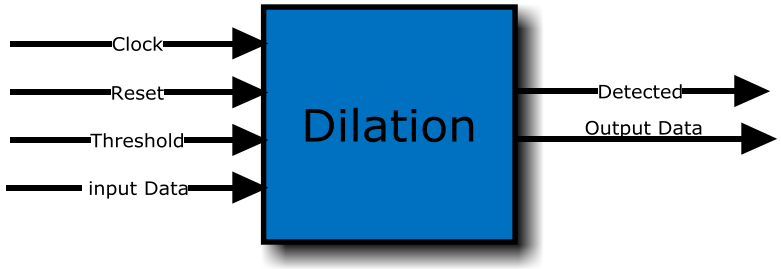

Fig 12: Dilation Module I/O signals

4.10 Display Controller:

This module generates the signals which drive the LCD touch screen. This module also transports the RGB 24-bits data bus to the LTM and generates a horizontal sync, a vertical sync and data enable signals.

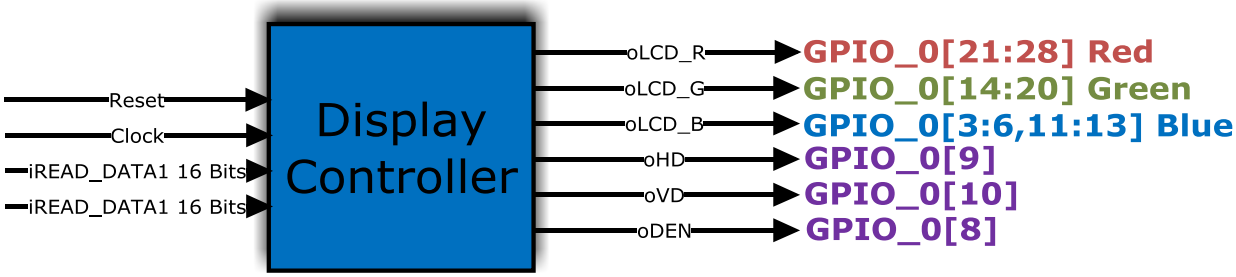

Fig 13: Display Controller Module I/O signals

Before RGB data enters this module, the user has the option to select the desired detection from the parallel channels of RGB which is to be displayed on the screen. More functionally of the threshold is enabled through the 12 switches [12:1].

Detection:	Switches
Gray Video with color Object Detection	SW→15=Low , SW→14=Low, SW→13=Low
Prewitt Edge Detection	SW→15=Low , SW→14=Low, SW→13=High
Cartoon Effect	SW→15=Low , SW→14= High, SW→13=Low
Dilation Edge Detection	SW→15=Low , SW→14= High, SW→13= High
Sobel Edge Detection	SW→15= High, SW→14=Low, SW→13=Low
Sobel with color Object Detection	SW→15= High, SW→14=Low, SW→13= High
RGB Video with Sobel Edge Detection	SW→15= High, SW→14= High, SW→13=Low
Emboss Effect	SW→15= High, SW→14= High, SW→13= High

RGB VALUES FROM RS232 ASCII HEX TO DECIMAL

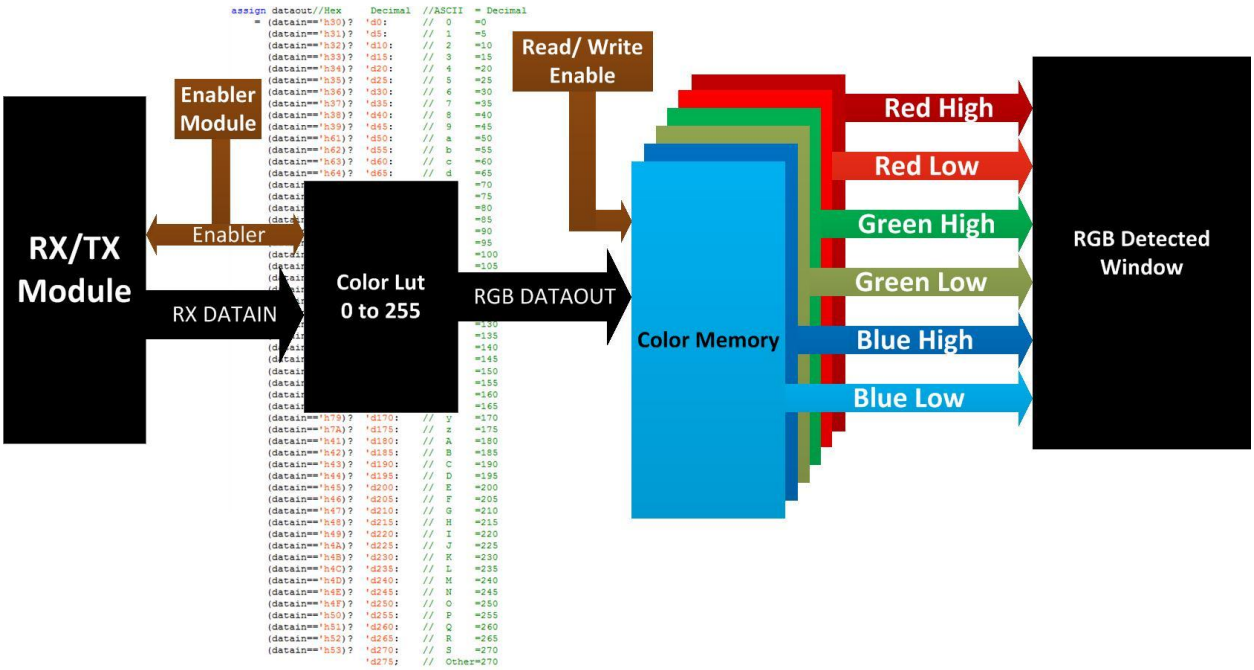

Fig 14: RGB values through RS232 PORT block diagram

5. Platform:

5.1 Hardware:

Hardware used for this projects are DE2-70 board, LTM display, D5M camera and two zigbee.

5.2 Software:

A wide varity of software are used for this project, such as Quartus [4], Synplify [5], ModelSim [6] and XCTU [7]. Quartus is programmable logic device design software from Altera. Its features include:

- An implementation of VHDL and Verilog for hardware description.
- Visual edition of logic circuits.
- Vector waveform simulation.

ModelSim is used to simulate and verify the functionality of verilog code. Synplify is the primary tool used to create the hardware design which gives the RTL view to visualize the design and to find the errors in advance. The ModelSim computer software is a great tool for debugging the code.

To enable the direct XBee communication, Digi's X-CTU application is used to read and write the settings of XBee module.

6. Test Results:

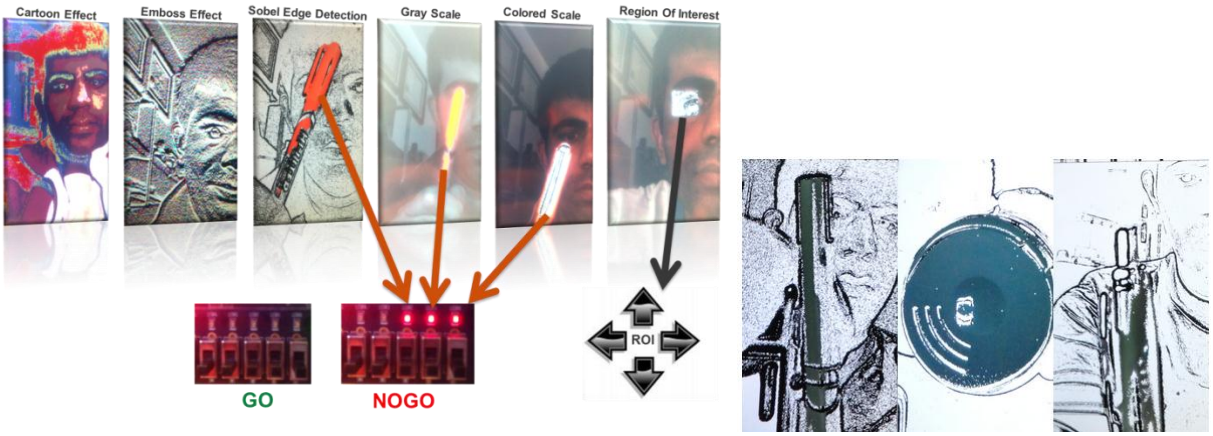

Fig 15: Results of detection and effects

Flow Status	Successful - Tue Dec 20 20:38:57 2011
Quartus II Version	11.0 Build 208 07/03/2011 SP 1 SJ Web Edition
Revision Name	DE2_70
Top-level Entity Name	ents689y
Family	Cyclone II
Device	EP2C70F896C6
Timing Models	Early Fitter Estimation
Total logic elements	4,503 / 68,416 (7 %)
Total combinational functions	3,625 / 68,416 (5 %)
Dedicated logic registers	2,762 / 68,416 (4 %)
Total registers	2832
Total pins	433 / 622 (70 %)
Total virtual pins	0
Total memory bits	137,856 / 1,152,000 (12 %)
Embedded Multiplier 9-bit elements	87 / 300 (29 %)
Total PLLs	2 / 4 (50 %)

Fig 16: Resource usage in the FPGA chip.

Filter Results:

Fig 17: Filters Results

7. Applications:

Defective items are identified and prevented from reaching to the customer.
Following are the applications of using this device in the industry.

Smart camera can used to provide better stability in measurement. The camera performs RGB processing of marks, such as print on cardboard, half-filled drink bottle to reliably detect subtle differences.

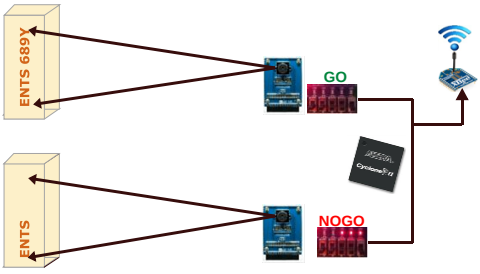

Fig 13: The color of print on cardboard is easily registered and identified as RGB information.

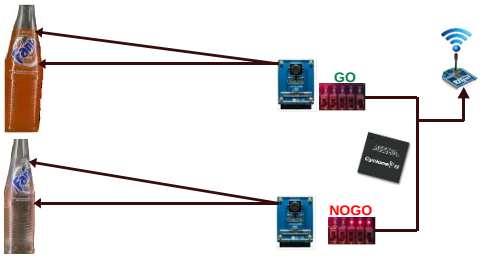

Fig 15: The color of bottle is easily registered and identified as RGB information.

8. Conclusion:

Overall the results from the project were satisfied. In this paper, we have shown how to create a pixel which is based on color window detector. We elaborated the concept of edge detection using FPGA device. Improvement can be made for the future work with more in depth parallel and pipeline structure using low level programming techniques. Finally, the least the color detection of any range in FPGA chip also inspires more intelligent control of the system in the future. This stand-alone, FPGA-based smart camera is useful in the tracking of a moving object in the factories, worker's vest, operator position and missing labels on the object.

References:

[1] http://nptel.iitm.ac.in/courses/Webcoursecontents/IIT%20Kharagpur/Embedded%20systems/Pdf/Lesson-20.pdf
[2] http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=5716147
[3] http://www.secs.oakland.edu/~ganesan/ece576f10project/index_files/Page361.htm
[4] Quartus FPGA design software, from http://www.altera.com/products/software/quartus-ii/web-edition/qts-we-index.html
[5] The Synplify solution Synplicity, from http://www.synplicity.com/products/synplifypro/
[6] Mentor Graphics Modelsim, from http://www.mentor.com/products/fpga/simulation/modelsim
[7] http://www.digi.com/support/kbase/kbaseresultdetl.jsp?kb=125