Journal of Theoretical and Applied Information Technology
10th December 2013. Vol. 58 No.1

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

# DIGITAL IMAGE PROCESSING USING SOBEL EDGE DETECTION ALGORITHM IN FPGA

**DHANABAL R[#1],BHARATHI V[*2] , S.KARTIKA [#3]**
[#1]Assistant Professor (Senior Grade) ,VLSI division, SENSE, VIT University,
[*2]Assistant Professor, GGR College of Engineering, Vellore,
[#3]Btech Student ,VIT University, Vellore- 632014,Tamil Nadu, India

E-mail: rdhanabal@vit.ac.in,bharathiveerappan@yahoo.co.in, kartikasekar001@yahoo.co.in

**ABSTRACT**

Image processing is important in modern data storage and data transmission especially in progressive transmission of images, video coding (teleconferencing), digital libraries, and image database, remote sensing. It has to do with manipulation of images done by algorithm to produce desired images. Digital Signal Processing (DSP) improve the quality of images taken under extremely unfavourable conditions in several ways brightness and contrast adjustment, edge detection, noise reduction, focus adjustment, motion blur reduction etc. The advantage is that image processing allows much wider range of algorithms to be applied to the input data in order to avoid problems such as the build-up of noise and signal distortion during processing. Digital image processing has applications reaching out into our everyday life such as medicine, surveillance, automated industry inspection and many more. Implementing such applications on an application specific hardware offers much greater speed than on a general purpose computer where it can be done easier. In this project, implementation a co-processor for image processing is done. The co-processor is modeled for edge detection of images. The Edge detection algorithm will be implemented on an FPGA, where the inherent parallelism offers a better performance. The architecture is like ARM processor which acts as the master is having the images which has to be processed. ARM will transfer the image to the FPGA for processing and after the image is being processed, the FPGA will display the processed image through a VGA display. The image send by ARM will be stored in an instantiated memory in FPGA. Edge detection core implemented in FPGA then reads the image from memory, process it and stores the processed image back in the memory. VGA controller designed reads the processed image from the memory and displays it. Sobel Edge detection algorithm is used for edge detection of images which is efficient in getting smooth edges and also less sensitive to noise.

**Keywords:** *Image processing, Digital Signal Processing(DSP), Co-processor, Field Programmable Gate Array(FPGA), ARM processor, VGA controller, Sobel Edge detection algorithm*

## 1. INTRODUCTION

Digital image processing is an ever expanding and dynamic area with applications reaching out into our everyday life such as medicine, space exploration, surveillance, authentication, automated industry inspection and many more areas. Applications such as these involve different processes like image enhancement and object detection. With advances in the VLSI technology hardware implementation has become an attractive alternative. Implementing complex computation tasks on hardware and by exploiting parallelism and pipelining in algorithms yield significant reduction in execution times

There are two types of technologies available for hardware design. Full custom hardware design also called as Application Specific Integrated Circuits (ASIC) and semi custom hardware device, which are programmable devices like Digital signal processors (DSPs) and Field Programmable Gate Arrays (FPGA's). Full custom ASIC design offers highest performance, but the complexity and the cost associated with the design is very high. The ASIC designs are used in high volume commercial applications. DSPs are a class of hardware devices that fall somewhere between an ASIC and a PC in terms of the performance and the design complexity. DSPs are specialized microprocessors, typically programmed in C, or with assembly code for improved performance. It is well suited to extremely complex math intensive tasks such as image processing.

Journal of Theoretical and Applied Information Technology
10th December 2013. Vol. 58 No.1

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Field Programmable Gate Arrays are reconfigurable devices. Hardware design techniques such as parallelism and pipelining techniques can be developed on a FPGA, which is not possible in dedicated DSP designs. Implementing image processing algorithms on reconfigurable hardware minimizes the time-to-market cost, enables rapid prototyping of complex algorithms and simplifies debugging and verification. Therefore, FPGAs are an ideal choice for implementation of real time image processing algorithms. In this project our objective is to design and implement a co-processor for image processing in FPGA. The co-processor will be designed for edge detection of images. Edge detection will be performed using Sobel Edge Detection algorithm.

## 2. DESIGN TOOLS USED

### 2.1 Xilinx ISE 10.1

Xilinx ISE 10.1 is used for design, simulation and synthesis of hardware system modeled in verilog HDL. In Xilinx the target device used to implement the design is xc3s1500-4fg676.

### 2.2 TLL Power Monitor Application

This application monitors the power supply to the board. The board has a housekeeping ARM processor which enables a software control over the power supply to the board. Also the programming of the hardware is done with the same.

### 2.3 MATLAB 7.5

MATLAB is used to model the whole system for an easy understanding of hardware implementation of the algorithm. Once the algorithm was realized successfully Edge Detection of a real time video streamed from web was done in Simulink.

### 2.4 Eclipse IDE

The C codes for the ARM was designed and build in Eclipse IDE. Also another C program to convert the bitmap image to a pixel data format of RGB565 was also build in the Eclipse.

## 3. EDGE DETECTION

Edges are considered to be most important image attributes that provide valuable information for human image perception. Edge detection is a very complex process affected by deterioration due to different level of noise. In early processes, the edge detection was mainly performed on software due to its large hardware requirement and also the application-specific integrated circuits have not gain much advancement. But present researches on programmable devices make it possible to implement edge detection algorithms on these devices whose design turn-around time varies from few hours to few days. An edge is the boundary between an object and the background. Edge detection is identifying points in a digital image at which the image brightness changes sharply or more formally has discontinuities. The purpose of detecting sharp changes in image brightness is to capture important events and changes in properties of the world. Discontinuities in image brightness are likely to correspond to

- discontinuities in depth
- discontinuities in surface orientation
- changes in material properties
- variations in scene illumination

## 4. SOBEL EDGE DETECTION

In this project a FPGA implementation of Sobel edge detection algorithm has done. The choice of Sobel edge detection operator is motivated by the fact that they incorporate both the edge detection as well as smoothing operator so that they have good edge detection capability in noisy conditions**.**

The Sobel operator performs a *2-D* spatial gradient measurement on images. Transferring a *2-D* pixel array into statistically uncorrelated data set enhances the removal of redundant data; as a result, reduction of the amount of data is required to represent a digital image. The Sobel edge detector uses a pair of *3 x 3* convolution masks, one estimating gradient in the x-direction and the other estimating gradient in *y*–direction. The Sobel detector is incredibly sensitive to noise in pictures, it effectively highlight them as edges. Hence, Sobel operator is recommended in massive data communication found in data transfer

The operator consists of a pair of 3×3 convolution kernels as shown in Figure. One kernel is simply the other rotated by 90°.

| | | |
|---|---|---|
| -1 | 0 | +1 |
| -2 | 0 | +2 |
| -1 | 0 | +1 |

Gx

| | | |
|---|---|---|
| +1 | +2 | +1 |
| 0 | 0 | 0 |
| -1 | -2 | -1 |

Gy

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

The kernels are designed to respond maximally to edges running vertically and horizontally relative to the pixel grid, one kernel for each of the two perpendicular orientations. The kernels can be applied separately to the input image, to produce separate measurements of the gradient component in each orientation (call these *Gx* and *Gy*). The gradient magnitude is given by:

$|G| = \sqrt{(Gx^2+Gy^2)}$

Typically, an approximate magnitude is computed using:

$|G| = |Gx|+|Gy|$

The implementation of Sobel Edge detection algorithm is as follows; Consider we have the image G(c,r), Then Gx is given by

Gx= [G(c+1, r-1) +2*G(c+1, r) +G(c+1, r+1)-G(c-1, r+1)-G(c-1, r)-G(c-1, r-1)];

Also Gy is given by

Gy= [G(c-1, r-1) +G(c, r-1) +G(c+1, r-1)-G(c-1, r+1)-2*G(c, r+1)-G(c+1, r+1)];

Once you have the Gx and Gy its sum is calculated to obtain the gradient magnitude and is compared with the threshold value. If the compared value is higher than threshold then the pixel is replaced by a one otherwise with a zero. The same is done with all the pixels of the image to find the edges.

## 5. IMPLEMENTATION METHODOLOGY

The traditional hardware implementation of image processing uses Digital Signal Processors (DSPs) or Application Specific Integrated Circuits (ASICs). The growing need for faster and cost-effective systems triggers a shift to Field Programmable Gate Arrays (FPGAs), where the inherent parallelism results in better performance. Computationally demanding functions like convolution filters, motion estimators, two dimensional Discrete Cosine Transforms (2D DCTs) and Fast Fourier Transforms (FFTs) are better optimized when targeted on FPGAs.

### 5.1 Design Approach

Using MATLAB procedural routines to operate on images represented as matrix data, this software algorithm were designed to resemble the hardware algorithm as closely as possible.

TLL 5000 board were used as it's equipped with SPARTAN 3 FPGA and also as far as design of Co-Processor is concerned an 80 pin mezzanine connector was also available which was made use for master co-processor communication. Also TLL 6219 board with an ARM 9 processor which comes with same 80 pin mezzanine connector was used as master. The board is also equipped with a 24 pin GPIO port which can be used to access a 1.3 Mp camera for a real time image capture.

Once the algorithm was implemented in MATLAB the hardware design was done using verilog HDL. Xylinx ISE 10.1 was used for the design and simulation. Once the design gets in XSVF (xylinx serial vector format) file format its dumped to the FPGA for verification.

### 5.2 System Implementation Flow

- Edge Detection is done with Sobel Edge Detection algorithm.
- The model is realized in Matlab.
- A model for real time edge detection for video signals is done using Simulink.
- Edge Detection Module is developed using verilog HDL and simulated.
- VGA Monitor Controller for the display is implemented in FPGA(XC3S1500) and color patterns are displayed on the monitor.
- ARM processor is made master and the image stored in its SD RAM is transferred to FPGA for image processing, the processed image is displayed through the VGA.
- As a first step to interconnect ARM and FPGA an eight bit data is written to the data bus and got it written back to FPGA.
- Then image is got transferred from ARM to FPGA and got displayed through VGA
- Finally edge detection core is inserted for image processing.

### 5.3 ARM

The TLL 6219 board is populated with an i.MX21 processor from Freescale. The i.MX21 utilizes the ARM926EJ-S core connected via the AMBA bus to broad range of peripherals. Key features of the board includes

- Freescale i.MX21 SOC ARM9 based processor.
- Memory: 64MB SDRAM & 16MB NOR Flash.
- Two 90 pin mezzanine connectors.
- µmon boot-loader for low-level real time programming applications.

Journal of Theoretical and Applied Information Technology
10th December 2013. Vol. 58 No.1

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

The i.MX21 utilizes a 32 bit address that s capable of addressing a 4 GB physical address space. The external interface is used for communication between ARM and FPGA .CS1 is used to select the mezzanine memory. The CS1 signal is output from TLL 6219 to select 64MBof memory attached to FPGA.CS1 decodes the memory in the range of 0*CC00_0000 to 0*CFFF_FFFF. CS1 maps 16MB of onboard (TLL 5000) SD RAM, 16MB of onboard Flash.

**i.MX21 Signal Descriptions**

*A [25:0] Address bus signals

*D [31:0] Data bus signals

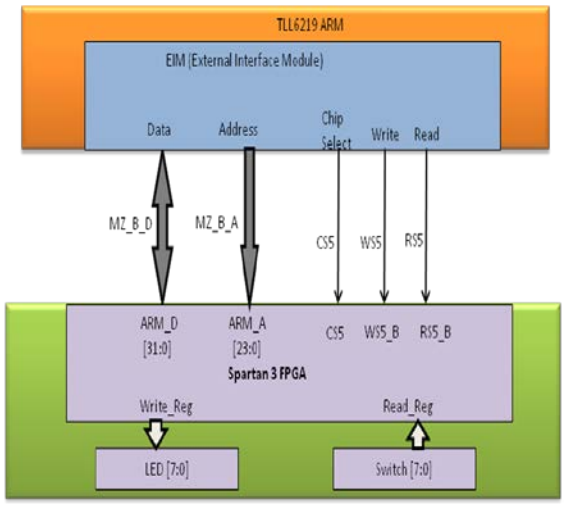
*RW RW signal Indicates whether external access is a read (high) or write (low) cycle.

*CS [5:0] Chip Select - The chip select signals CS [3:2] are multiplexed with CSD [1:0] and are selected .

by the Function Multiplexing Control Register (FMCR) in the System Control chapter. By default CSD [1:0] is selected. DTACK is multiplexed with CS4.

CS5 is used to select mezzanine registers. The CS5 signal is output from TLL6219 to select 16MB of registers in FPGA.CS5 decodes memory in the range of 0*D300_0000 to 0*D3FF_FFFF. The image is send to ARM via hyper terminal where you can access ARM with the help of µmon. The image format used is RGB565, where 5 bits are used for both red and blue and 6 bits for green.

### 5.4 FPGA –ARM Interface




*FPGA-ARM Interfacing*

FPGA and ARM are connected through a 80 pin mezzanine connector. The image stored in the ARM will be transferred to FPGA through the 32 bit data bus. The address specified will be corresponding to the instantiated memory in FPGA. A xylinx IP Core for a dual port memory is used. The data width is kept at 16 bit and depth of 19200 is used.

### 5.5 Field Programmable Gate Arrays (FPGAs)

Field Programmable Gate Arrays (FPGAs) represent reconfigurable computing technology which is in some ways ideally suited for video processing. FPGAs generally consist of a system of logic blocks (usually look up tables and flip-flops) and some amount of Random Access Memory (RAM), all wired together using a vast array of interconnects. All of the logic in an FPGA can be rewired, or reconfigured, with a different design as often as the designer likes. Today, FPGAs can be developed to implement parallel design methodology, which is not possible in dedicated DSP designs.

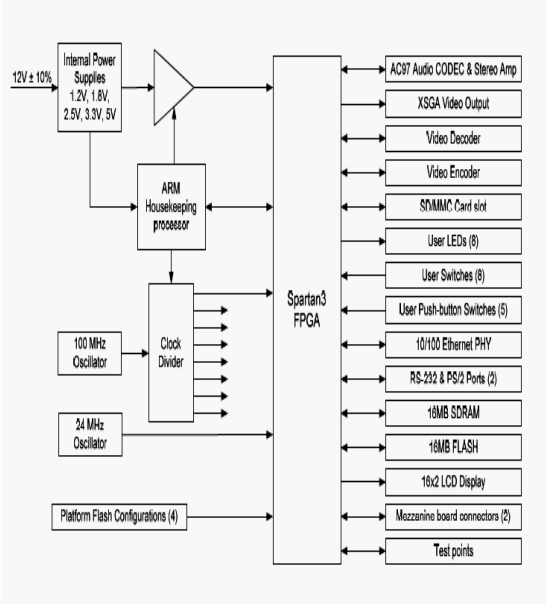
A Spartan 3 FPGA from xylinx is used in the project. The image received is stored in the instantiated memory. An IP core for a dual port ram from Xylinx is used as memory. The pixel data will be the read by sobel edge detection core for processing and the processed image will be stored back. Once the processed image is ready the VGA controller designed can read the pixel data and display it on the VGA monitor. Image displayed will be of QQVGA ie. The resolution will be of 120 *160. This low resolution is kept for less memory usage and ease of handling the data.

#### 5.5.1 TLL 5000 Development Board

The TLL5000 Development System provides an advanced hardware/software platform that consists of a high-capacity Spartan3 Platform FPGA surrounded by a comprehensive collection of peripheral components that can be used to create a complex digital system.

Journal of Theoretical and Applied Information Technology
10th December 2013. Vol. 58 No.1

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195



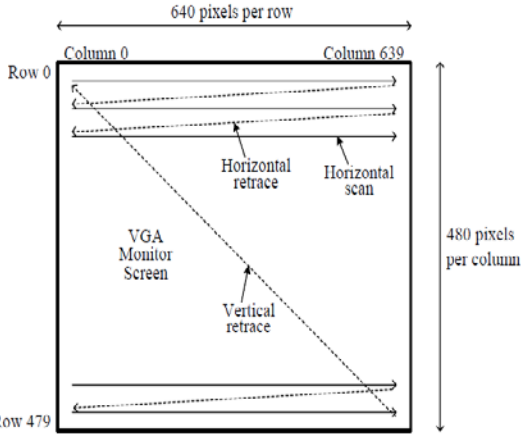

*TLL5000 Development System Block Diagram*

The Board is equipped with Xilinx Spartan3 XC3S1500 FPGA device packed in 676-lead fine-pitch ball grid array package and Speed Grade -4. The TLL5000 includes a video DAC and 15-pin high-density D-sub connector to support XSGA output. The video DAC can operate with a pixel clock of up to 180 MHz. Only VESA-compatible output of 640 x 480 at 60 Hz refresh is supported by software. TLL5000 system includes two 80 pin (2 x 40) mezzanine board connectors. Every connector provides 40 Spartan3 I/O pins, JTAG signals, two differential clocks synchronized to the on-board 100Mhz master clock, and 3.3V/3.5A and 18V/0.5A power supply lines.

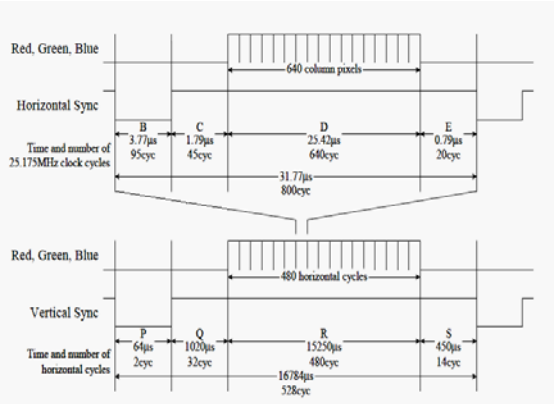
The verilog code is written for a VGA controller which can be used to display colors on the VGA monitor. The sobel edge detection core is also coded in verilog.

### 5.6 VGA Controller Implementation

The monitor screen for a standard VGA format contains 640 columns by 480 rows of picture elements called pixels as shown in Figure. The monitor continuously scans through the entire screen turning on or off one pixel at a time at a very fast speed. The scan starts from row 0, column 0 at the top left corner, and moves to the right until it reaches the last column in the row. When the scan reaches the end of a row, it continues at the beginning of the next row. When the scan reaches the last pixel at the bottom right corner of the screen, it goes back to the top left corner of the screen, and repeats the scanning process again. In order to reduce flicker on the screen, the entire screen must be scanned 60 times per second or higher.




The VGA monitor is controlled by five signals: red, green, blue, horizontal synchronization, and vertical synchronization. The three colour signals, referred to collectively as the RGB signal, are used to control the colour of a pixel at a location on the screen. In order to produce more colours, each analog color signal must be supplied with a voltage between 0.7 to 1.0 volts for varying the intensities of the colors. The horizontal and vertical synchronization signals are used to control the timing of the scan rate. The horizontal synchronization signal determines the time to scan a row, while the vertical synchronization signal determines the time to scan the entire screen. By manipulating these five signals, images are formed on the monitor screen.




The horizontal and vertical synchronization signals timing diagram is shown above. The start of a row scan begins with the horizontal sync signal going low for 3.77 µsec as shown by region B. This is followed by a 1.79 µsec high on the signal as shown by region C. Next, the data for the three

Journal of Theoretical and Applied Information Technology
10th December 2013. Vol. 58 No.1

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

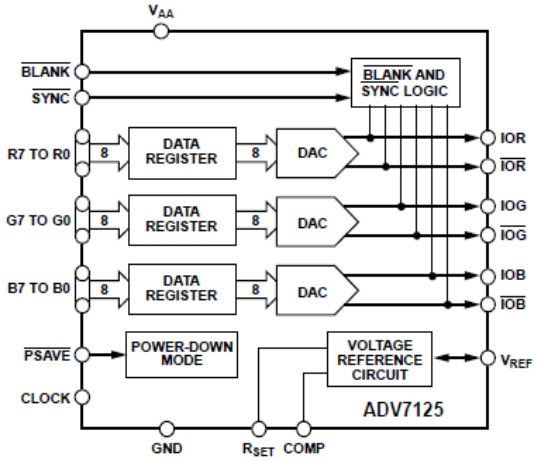
colour signals are sent, one pixel at a time, for the 640 columns as shown in region D for 25.42 µsec. Finally, after the last column pixel, there is another 0.79 µsec of inactivity on the RGB signal lines as shown in region E before the horizontal sync signal goes low again for the next row scan. The total time to complete one row scan is 31.77µsec.

The 64µsec active low vertical sync signal resets the scan to the top left corner of the screen as shown in region P, followed by a 1020 µsec high on the signal as shown by region Q. Next, there are the 480 row scans of 31.77µsec each, giving a total of 15250µsec as shown in region R. Finally, after the last row scan, there is another 450µsec as shown in region S before the vertical sync signal goes low again to start another complete screen scan starting at the top left corner.

The total time to complete one complete scan of the screen is 16784µsec. Assuming a clock frequency of 25.175MHz. The clock period is then 1 / 25.175×106, or about 0.0397µsec per clock cycle. For region B in the horizontal synchronization signal, we need 3.77µsec, which is approximately 3.77 / 0.0397 = 95 clock cycles. For region C, we need 1.79µsec, which is approximately 45 clock cycles. Similarly, we need 640 clock cycles for region D for the 640 columns of pixels, and 20 clock cycles for region E. The total number of clock cycles needed for each row scan is, therefore, 800 clock cycles. Notice that with a 25.175MHz clock, region D requires exactly 640 cycles, giving us the 640 columns per row.

### 5.6.1 Video DAC

The TLL 5000 board consists of a High Speed Video DAC ADV7125 from Analog Devices. The ADV7125 is a triple high speed, digital-to-analog converter on a single monolithic chip. It consists of three high speed, 8-bit video DACs with complementary outputs, a standard TTL input interface, and a high impedance, analog output current source. The ADV7125 has three separate 8-bit-wide input ports. The VGA Controller implemented in FPGA can input signals to the video DAC. The functional block diagram of ADV7125 as follows;




**Pin Function Descriptions:**

i. **Blank:** Composite Blank Control Input (TTL Compatible). A Logic 0 on this control input drives the analog outputs, IOR, IOB, and IOG, to the blanking level. The BLANK signal is latched on the rising edge of CLOCK. While BLANK is a Logic 0, the R0 to R7, G0 to G7, and B0 to B7 pixel inputs are ignored.

**ii. SYNC:** Composite Sync Control Input (TTL Compatible). A Logic 0 on the SYNC input switches off a 40 IRE current source. This is internally connected to the IOG analog output. SYNC does not override any other control or data input; therefore, it should only be asserted during the blanking interval. SYNC is latched on the rising edge of CLOCK. If sync information is not required on the green channel, the SYNC input should be tied to Logic 0.

iii. **PSAVE:** Power Save Control Pin. Reduced power consumption is available on the ADV7125 when this pin is active.

iv. **CLOCK:** Clock Input (TTL Compatible). The rising edge of CLOCK latches the R0 to R7, G0 to G7, B0 to B7, SYNC, and BLANK pixel and control inputs. It is typically the pixel clock rate of the video system. CLOCK should be driven by a dedicated TTL buffer.

v. **G0 to G7, B0 to B7, R0 to R7:** Red, Green, and Blue Pixel Data Inputs (TTL Compatible). Pixel data is latched on the rising edge of CLOCK. R0, G0, and B0 are the least significant data bits. Unused pixel data inputs should be connected to either the regular printed circuit board (PCB) power or ground plane.

vi. **IOR, IOG, IOB:** Red, Green, and Blue Current Outputs. These high impedance current

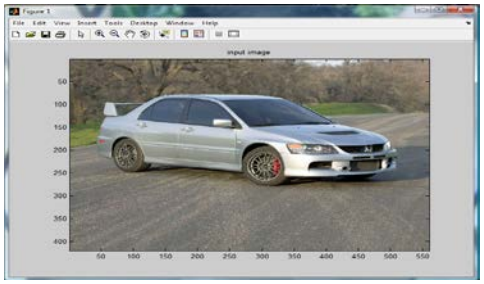
Journal of Theoretical and Applied Information Technology
10th December 2013. Vol. 58 No.1

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

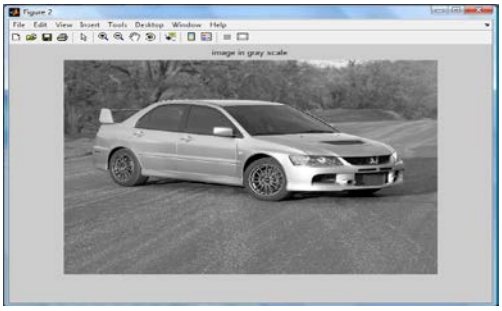
sources are capable of directly driving a doubly terminated 75 Ω coaxial cable. All three current outputs should have similar output loads whether or not they are all being used.

## 6. IMPLEMENTATION RESULTS

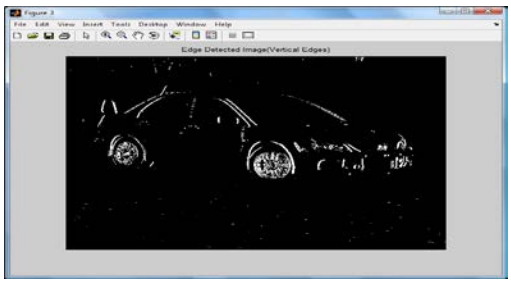
### 6.1 Implementation Snap Shots (MATLAB)



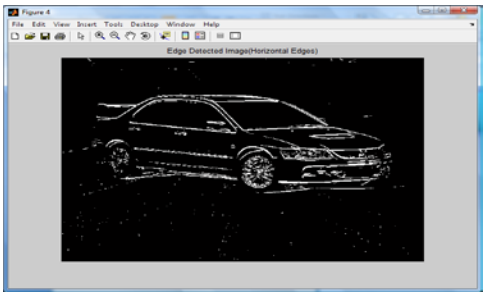
*Input Color Image*



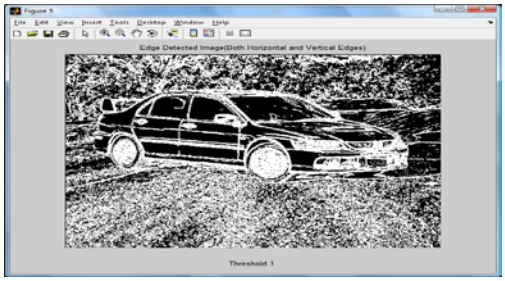
*Grey Scale Converted Image*



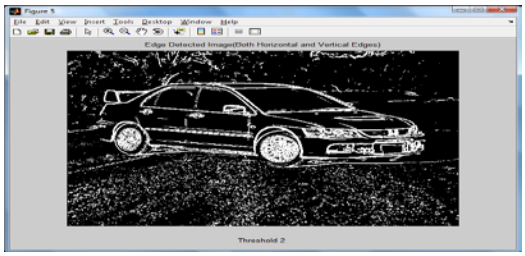
*Vertical Edge Detected image*



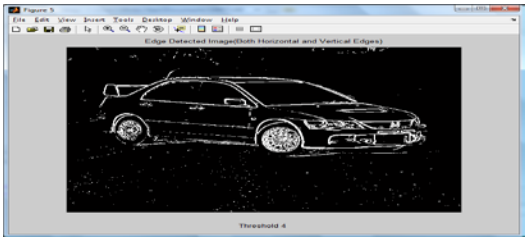
*Horizontal Edge Detected Image*



*Both Horizontal And Vertical Edges Detected Image For Threshold 1*



*Both Horizontal And Vertical Edges Detected Image For Threshold 2*



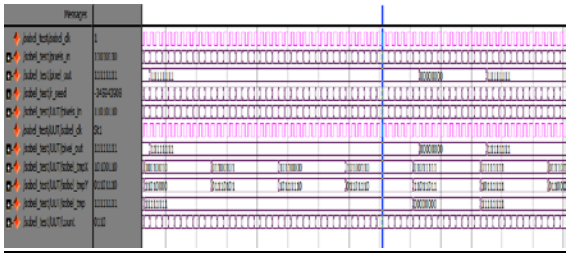
*Both Horizontal And Vertical Edges Detected Image For Threshold 3*

Journal of Theoretical and Applied Information Technology
10th December 2013. Vol. 58 No.1

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

*Color Image 1 Displayed With VGA Controller Snap Shot .*

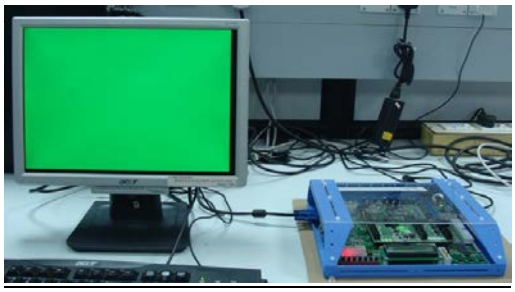
### 6.2 Simulation Results (Sobel Edge detection core)



*Simulation Result for Sobel Edge Detection core*

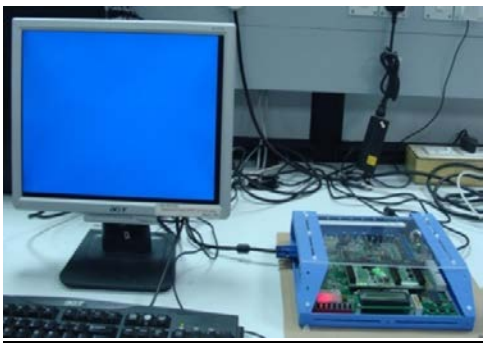
### 6.3 Implementation Snap Shots (VGA Controller)

*Red color displayed Using VGA Controller Implemented in FPGA*



*Green color displayed Using VGA Controller Implemented in FPGA*

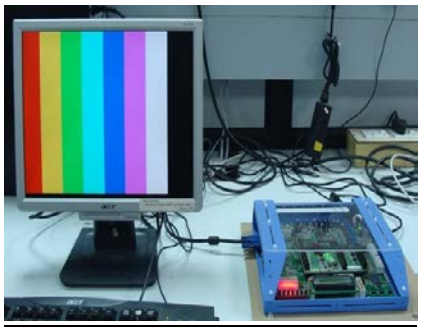
*Snap shot of color image 2 displayed with VGA Controller*



*Blue color displayed Using VGA Controller Implemented in FPGA*

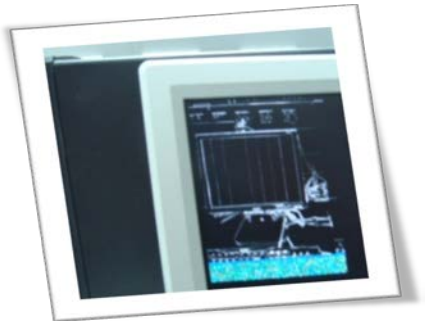
Journal of Theoretical and Applied Information Technology
10th December 2013. Vol. 58 No.1

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195



*Color Bands displayed Using VGA Controller Implemented in FPGA*

*Snap shot of Edge detected image 1 displayed with VGA Controller*



*Snap shot of Edge detected image 2 displayed with VGA Controller*

## 7. CONCLUSION AND FUTURE WORK

### 7.1 Conclusion

The project mainly focused on implantation of sobel Edge Detection algorithm in FPGA making it a co-processor while keeping an ARM processor as master. The master co-processor communication is tested. The processed image is shown on VGA display from the FPGA board.

### 7.2 Future Work

Instead of taking the stored image from the ARM memory a real time image can be easily captured with the help of a camera and put at the same place where the image is currently stored and the rest can be made the same and real time image processing can be done.

## REFRENCES:

[1] "A Classified and Comparative Study of Edge Detection Algorithms" by Mohsen Sharifi, Mahmoud Fathy, Maryam Tayefeh Mahmoudi Department of Computer Engineering, Iran University of Science and Technology. Proceedings of the International Conference on Information Technology: Coding and Computing (ITCC.02) 2002 IEEE

[2] "A Simple and Efficient Edge Detection Algorithm" Guangyu Luan and Rensheng Che Department of Automatic Measurement and Control Harbin Institute of Technology, Harbin, China. 2008 International Symposium on Computer Science and Computational Technology.

[3] "An efficient architecture for hardware implementations of image processing algorithms" Farzad Khalvati and Hamid R. Tizhoosh Department of Systems Design Engineering University of Waterloo, Waterloo, Ontario, Canada. 2009 IEEE

[4] "Optimized implementation of real time image processing algorithms on FPGA' Ailton F. Dias, Christopher Lavare, Mohamed akil, Yves Sorel. Proceedings of ICSP '93.

[5] "A host/co-processor FPGA-based Architecture for Fast Image Processing" John A. Kalomiros 1, John Lygouras IEEE International Workshop on Intelligent Data Acquisition and Advanced Computing Systems Technology and Applications 6-8 September 2007, Dortmund, Germany.

[6] "Coarse-Grain Dynamically Reconfigurable Coprocessor for Image processing in SOPC" by Almudena Lindoso, Luis Entrena, Juan Izquierdo, Judith Liu-Jimenez, Electronic Technology Department, University Carlos III of Madrid. ©2008 IEEE.

[7] Gonzalez, R., & Woods, R.,& Eddins,S. "Digital image processing Using Matlab" .Prentice-Hall Inc.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[8] "A Descriptive Algorithm for Sobel Image Edge Detection", O. R. Vincent,Clausthal University of Technology, Germany, O. Folorunso ,University of Agriculture,Abeokuta, Nigeria. Proceedings of Informing Science & IT Education Conference (InSITE) 2009.

[9] Comparison of Edge Detectors A Methodology and Initial Study Mike Heath,y Sudeep Sarkar,y Thomas Sanocki,z and Kevin Bowyery Y Computer Science & Engineering, Department of Psychology, University of South Florida, Tampa.

[10] "Edge Detection Tutorial" http://www.pages.drexel.edu/~weg22/index.html