International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 06 Issue: 09 | Sep 2019 www.irjet.net p-ISSN: 2395-0072

Sobel Edge Detection on Zynq based Architecture with Vivado

Neol Solanki[1], Neel tailor[2]

[1]*Student-B.Tech, Electronics Department, Birla Vishvakarma Mahavidhyalaya, Anand, Gujarat-India*
[2]*Student-B.Tech, Electronics Department, Birla Vishvakarma Mahavidhyalaya, Anand, Gujarat-India*

Abstract - *Edge detection is tool used in many image processing application for withdrawing information from image. Sobel edge detection is gradient based edge selection method to find edge pixels in image. This paper proposes an implementation of sobel edge detection algorithm to find edge pixels in gray scale image. We present a video streaming architecture and IP implementation using high level synthesis. A video with 720p resolution streamed from the HDMI source and edge detected video captured in VGA monitor. For implementation zybo board (based on zynq 7000) is used, which has provided adequate peripherals for implementation.*

Key Words: Sobel edge detection, Real time video processing, Video processing using Zybo, Video processing using HLS ,Video processing using zynq based architecture, Video processing with FPGA.

1. INTRODUCTION

The objective of this project is to detect the edge of the image using FPGA. Edge detection is algorithm used to detect edge by finding the boundaries of objects within images. It works by detecting discontinues in the brightness. Edge detection is used in object detection, machine vision, computer vision and image segmentation. Typical algorithms include Sobel, Canny, Prewitt, Roberts.

To implement this project we have used the zybo (ZYnq BOard), which is the member of zynq-7000 family, the Z-7010. The Z-7010 has dual-core ARM cortex-A9 processor with Xilinx 7-series field programmable gate array logic based on SOC (System on chip). The opulent set of multimedia and connectivity peripherals on Zybo can be conductive to individual for creating whole system. We have created IP for Sobel edge detection with the help of Vivado HLS and then implemented video pipelining architecture on Vivado IP integrator.

1.1 Sobel Edge detection

Sobel edge detection was first proposed by the Irwin sobel and Gary Feldman in 1968 at SAIL. Out of two types of edge detection gradient and laplacian, Sobel is based on former one. Gradient of intensity of each pixel is calculated in Sobel. It uses 3x3 kernel one for vertical other for horizontal for change in respective direction. Derivatives are calculated by convolution of kernel with source image. If $\mathbf{G_x}$ and $\mathbf{G_y}$ are images containing horizontal and vertical derivatives, then,

$$\mathbf{G}_x=\begin{pmatrix}1 & 0 & -1\\ 2 & 0 & -2\\ 1 & 0 & -1\end{pmatrix}*\boldsymbol{A} \quad , \quad \mathbf{G}_y=\begin{pmatrix}-1 & -2 & -1\\ 0 & 0 & 0\\ 1 & 2 & 1\end{pmatrix}*\boldsymbol{A}$$

where $\boldsymbol{A}$ is the source image. The edge is measured by amplitude and direction given by

$$\mathbf{G} = \sqrt{\mathbf{G_x} + \mathbf{G_y}} \quad , \quad \theta=\tan^{-1}\frac{\mathbf{G_y}}{\mathbf{G_x}}$$

Where $\mathbf{G}$ represents the magnitude and θ represents the direction.

1.2 HLS and IP Integrator Motivation

HLS can do synthesis of digital system directly from high level languages like C, C++ and also generate VHDL/Verilog from C/C++ source. The most arresting feature of HLS is that hardware implementation and designed functionality are isolated. The hardware is not implicitly fixed by the C-based description providing more flexibility. HLS extracts control and dataflow from source code and implementations are carried out based on the user applied directives. HLS provides more than one implementation for the same source and enables user to explore the design and find the most optimal design. Vivado accelerates the development of highly integrated, complex designs by providing the intelligent IP integration with features like auto-correction of key IPs, one click IP subsystem generation, real time DRCs, interface change propagation with powerful debug capability. It supports all design domains, hierarchy and advanced design services.

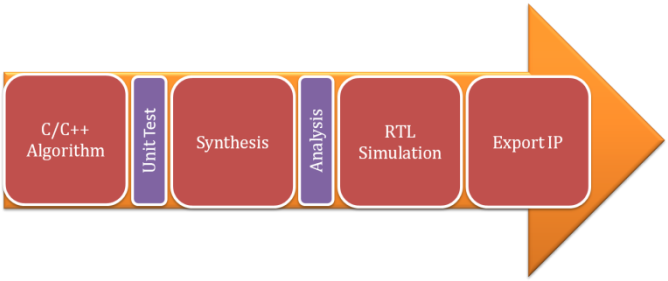

Fig -1: HLS Design flow

2. IP Creation using HLS

We have used processing function that accepts an AXI-stream RGB video input and outputs the similarity formatted processed video data. The project requirements are 1280x720 resolution and a stable video feed. Sobel edge detection algorithm takes RGB matrix and then its processes its value and generates the sobel edge detected output according to that image. First of all the data is in the AXI format which need to be converted into the matrix format and function used for the same is: *AXIvideo2mat (stream_in, img0).* Function used for converting matrix into gray scale is *CvtColor<HLS_RGB2GRAY> (img0, img1)*.

After the conversion in gray scale we can do the sobel edge detection. There is the function in HLS called sobel which we have used. *Sobel<1, 0, 3> (img1, img2).*This 1 signifies the x order and y order is 0 so it first calculates the x image derivative and 3 is the kernel size, which is 3x3 kernel.

Then to convert gray scale image into RGB format. *CvtColor<HLS GRAY2RGB> (img2, img3).*After converting in the RGB format we have to convert the matrix format back to its original form i.e. in AXI format, using function *Mat2AXIvideo (img3, stream_out).*

The utilization estimate for same in HLS in given by:

Summary

Name	BRAM_18K	DSP48E	FF	LUT
DSP	-	-	-	-
Expression	-	-	-	-
FIFO	0	-	80	334
Instance	9	3	1016	1583
Memory	-	-	-	-
Multiplexer	-	-	-	-
Register	-	-	11	-
Total	9	3	1107	1917
Available	120	80	35200	17600
Utilization (%)	7	3	3	10

Fig -2: Resource utilization (HLS)

3. Implementing video pipelining in IPI

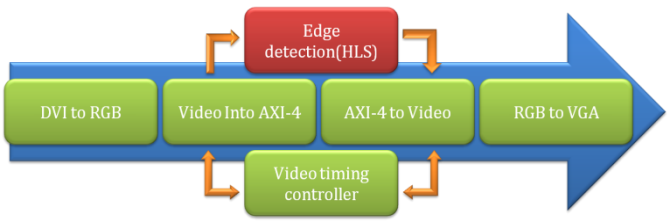

Fig -3: IPI block diagram

DVI to RGB converts the input from the HDMI source into the RGB format sampled from 1280x720 resolution video. After that Video into AXI-4 stream converts the RGB format into AXI mapping, this is fed to the edge detection ip which detects the edge of the video into AXI form. Video timing controller ensures the timing constrains between all the IPs. Edge detection ip propagates the edge detected image into AXI-4 to video stream, which further converts the AXI mapping into video and finally fed to VGA source capable of supporting same resolution.

Final board resources utilization is given by:

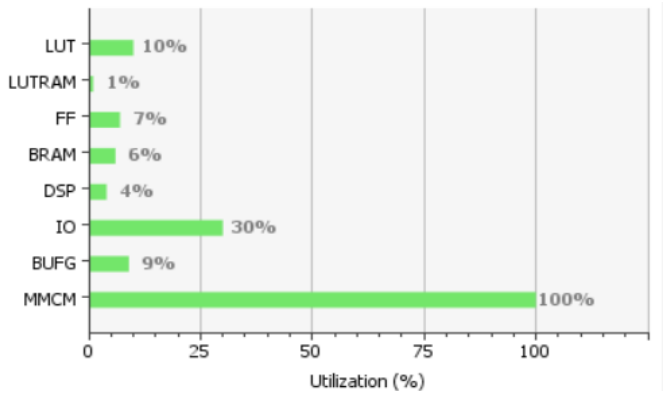

Fig -4: Board resources utilization

4. Test results

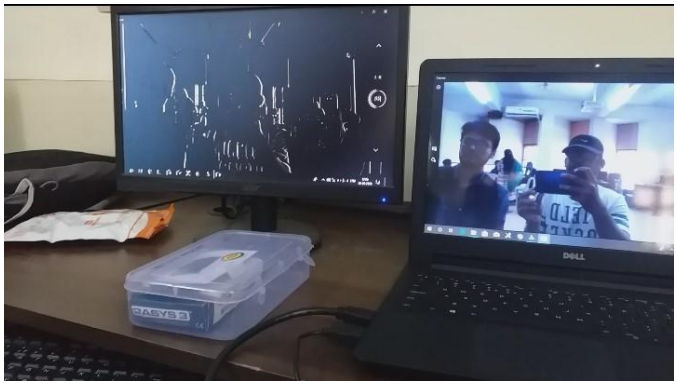

Fig -5

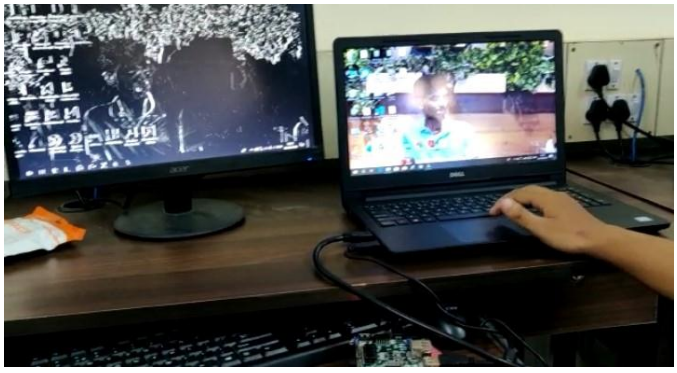

Fig -6

Figure 5 and 6 shows the edge detected live video stream in VGA monitor (left) from the HDMI source (right).

5. CONCLUSION

The paper has illustrated how efficiently Zybo board is capable of implementing the entire video processing system singularly with low power consumption, smaller physical size and with minimal resources utilization evidenced from the figure 4. Implication of implementing entire system singularly is notion of SoC. Individual can implement any complex design entirely using Zynq based architecture. In our example we used HLS and saw how good alternative it is to HDL language and can be time saving. Vivado provides tightly integration of all IPs and peripherals and also reusability. The video pipelining architecture designed in our example can be used for any video application is future.

6. REFERENCES

1. The zynq e-Book

2. Michrozed chronicles by Adam Taylor

3. Xilinx, Accelerating OpenCV Applications with Zynq-7000 All Programmable SoC using Vivado HLS Video Libraries, Stephen Neuendorffer, Thomas Li, and Devin Wang

4. DIGILENT Zybo video workshop