International Journal of Recent Technology and Engineering (IJRTE)
ISSN: 2277-3878, Volume-8 Issue-3, September 2019

Realization of Image Processing Algorithm based on FPGA

Alka Singh, Kanika Jindal

Abstract- This paper proposes the use of Xilinx System Generator for image processing. Several categories of algorithms are assisted by necessary libraries in Xilinx System Generator. This work integrates Matlab Simulink environment. The image processing algorithms are implemented by design approaches based on model design. The results are verified by hardware co-simulation. The system generator blocks are used for various algorithms of image processing for image negatives, RGB to grayscale, dilation, etc.

Keywords: Image processing, dilation, FPGA, Xilinx, System generator, threshold.

I. INTRODUCTION

A wide variety of applications of image processing are found in this era, ranging from computer vision, medical imaging, digital encryption, decryption and satellite imaging. The image processing algorithms are helpful in enhancing the quality of image which is helpful in robotic application, medical imaging and surveillance for identifying and tracking the target[1]. The real time processing of image consumes time and the algorithm is implemented at hardware level only. Each function is built by separate hardware in FPGA implementations. The software's programmable flexibility is retained at a low cost at high speed real time applications because FPGA's are processed in parallel. Xilinx System Generator is used in this paper for processing algorithms of image processing. A model based design approach is followed for implementing the algorithms on hardware.

A.Xilinx System Generator: FPGA hardware design is facilitated by system level modelling tool Xilinx Generator from Xilinx. The suitable modelling environment for designing hardware is provided by it by extending Simulink in various ways. The conversion of high level DSP block of system to RTL is done by the software. The Xilinx's FPGA ISE tools are used for synthesizing the result. The Xilinx based FPGAs are verified, debugged, implemented and its design process is automated by system generator. The simulation performance is increased 1000 times and a high speed HDL co-simulation interfaces are provided. For importing RTL into Simulink, a block of black box is supported by system generator. The co-simulation is done either by Xilinx ISE simulator or by Modelsim[2].

Design flow for Xilinx System Generator: Xilinx Blockset is used for building models after developing the algorithms. Matlab/Simulink environment is used for simulating these models. A video viewer reflects the result. For suitably implementing FPGA, configuration of results of system generator is done. The verification, synthesis and implementation of behavioural model is done on FPGA. The user constraints file(.ucf), test vectors and test bench are generated by generator for architecture testing.

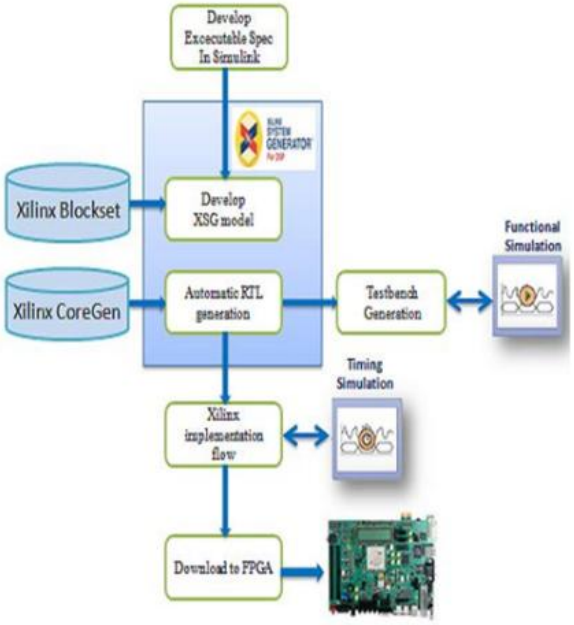

Fig. 1 Design flow of image processing algorithms based on FPGA

II. INTERFACING WITH DESIGN OF SYSTEM GENERATOR

A number in a simulation is represented by using a "double" in Simulink environment. This number system is not useful for FPGA because a lot of resources are consumed by this number system. The n bit fixed point numbers are used by Xilinx blocksets and for communicating Xilinx blocks with Simulink blocks, a conversion is mandatory[2]. This conversion uses Gateway In, Sampling and Gateway Out.

III. HARDWARE IMPLEMENTATION OF IMAGE PROCESSING

The implementation of all required algorithms of hardware in done between pre and post image processing.

Manuscript published on 30 September 2019
* Correspondence Author
Alka Singh*, Department of Information Technology, Noida Institute of Engineering and Technology. Email:researchnietip@gmail.com
Kanika Jindal, Department of Electronics and Communication Engineering, Noida Institute of Engineering and Technology.. Email:researchnietip@gmail.com

The entire application of image processing has common image source, image pre-processing and post-processing units and image viewer and implementation is done in Simulink.

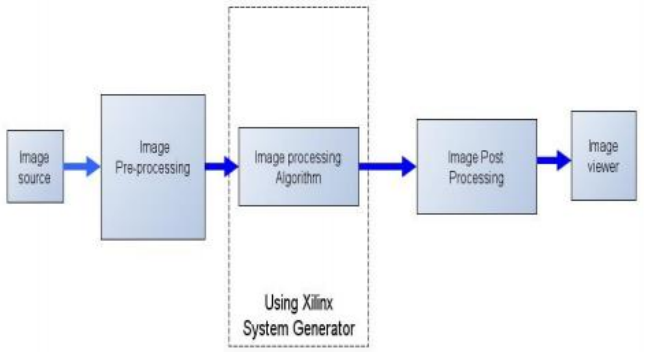

Fig. 2 Hardware implementation of image processing (design flow)

IV. IMAGE PROCESSING UNIT

This unit in Matlab provides an input as a test vector appropriate for bit stream compiling of FPGA by using System Generator[3].

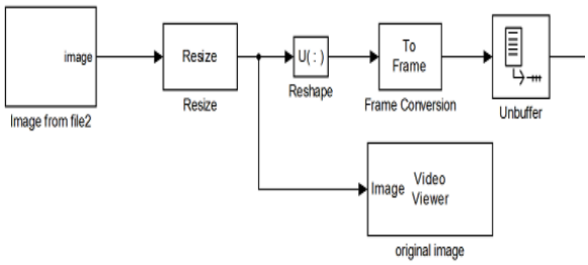

Fig. 3 Image pre-processing unit

Figure 3 illustrates the pre-processing unit block of an image. It converts 2 dimensional image to 1 dimensional, resizes it, and performs conversion of frame and this unit implements unbuffer. Since FPGAs function in 1-D data only, the conversion of data from 2 dimension to 1 dimension is necessary.

V. IMAGE POST PROCESSING UNIT

An image from a 1 dimensional array is recreated by image post-processing. There are four blocks in this unit: data type conversion, convert 1 dimensional to 2 dimensional, buffer and video viewer. The conversion of image signal to an unsigned integer is done in first block. Second block is responsible for conversion of scalar samples at low sampling rate to frame output. The conversion of 1 dimensional image signal into 2 dimensional image matrix is done by third block. The output image is displayed back on monitor by last block.

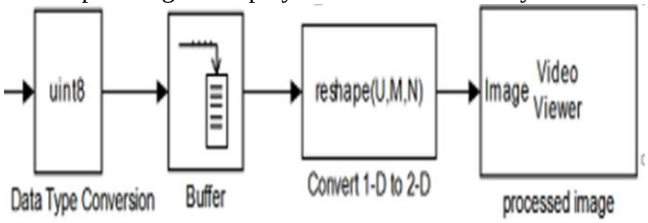

Fig. 4 Post processing unit

A. Gray scale conversion algorithm

The processing time is reduced, conversion of color images into a grayscale is done based on pixel's color containing green (G), red (R), blue (B) and green (G)[4][3]. The following equation is used for the conversion of RGB image to a grayscale image.

$$Y=0.3*R + 0.59*G + 0.11*B$$

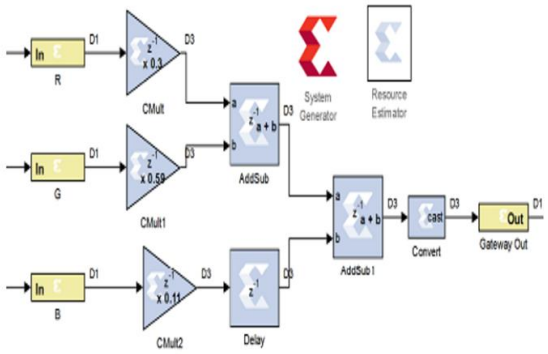

Fig. 5 Grayscale conversion algorithm

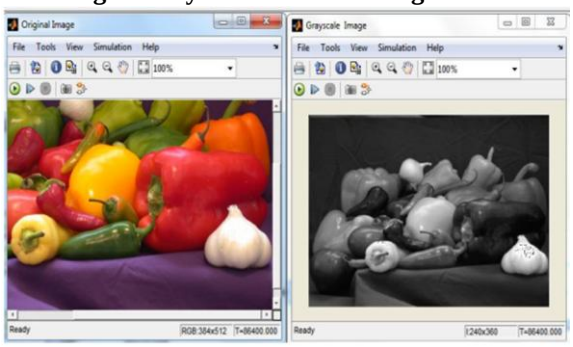

Fig. 6 Result obtained after grayscale conversion

B. Algorithm for grayscale image negative

The image matrix is simply inverted to obtain negative image. Such image produced like this seems like a film's negative. In Matlab, image source is inverted by using NOT gate to obtain this kind of image or Addsub block is used. The steps are simpler because both the NOT and AddSub gates are available in the library of Xilinx System Generator.

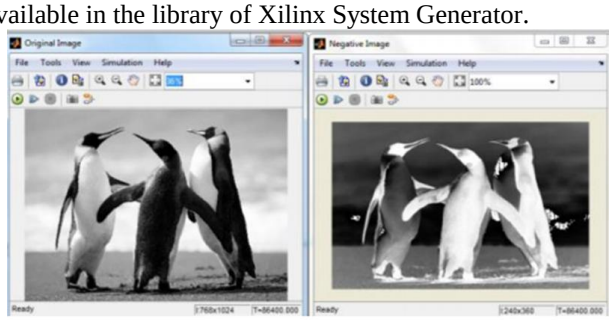

Fig. 9 Result obtained from grayscale negative

C. Algorithm for enhancing an image

The perception and interpretation of information delivered by images for an individual is improved by image enhancement [5][6]. The obtained results are shown in figure 11.

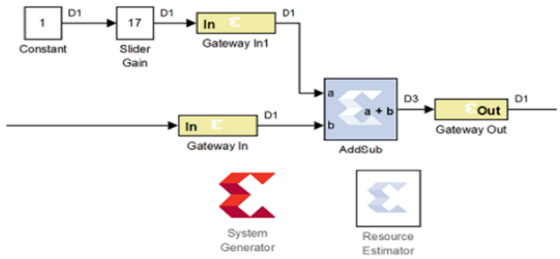

Fig. 10 Algorithm for enhancing grayscale image

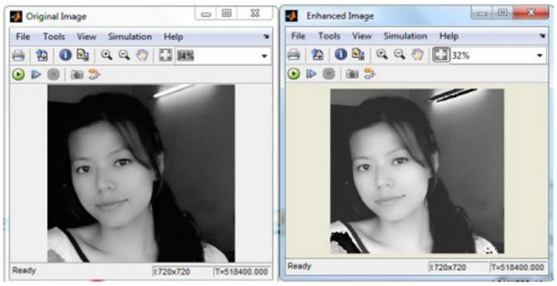

Fig. 11 Results obtained after grayscale image enhancement

D. Algorithm for stretching image contrast

The following equation is done for stretching the contrast of an image.

New_ pixel =3(old pixel-127) +112

New_pixel is the result obtained after transformation.

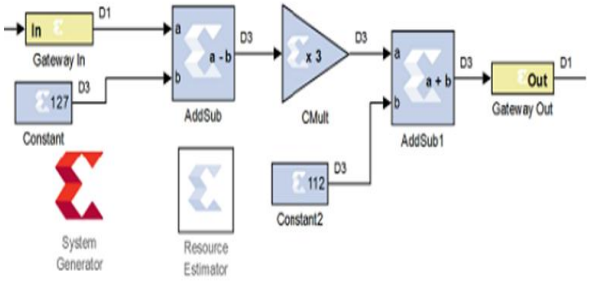

Fig. 12 algorithm for enhancement of image

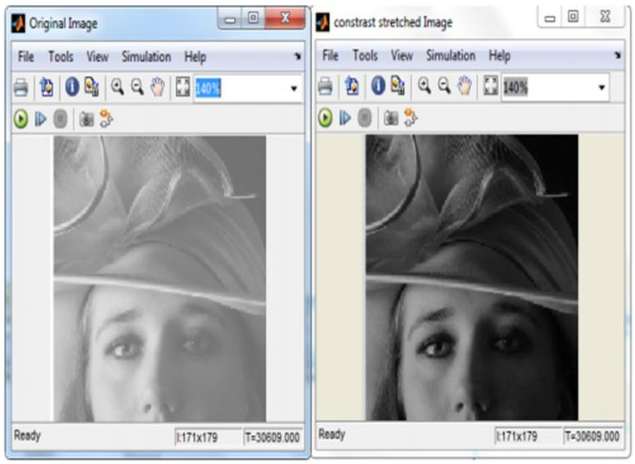

Fig. 13 Result from grayscale contrast stretching

In the image intensity a constant value is fixed, when smaller than that value, each pixel of an image is substituted with black pixel and if greater than constant value then substituted with a white pixel [7][3]. This is known as image thresholding.

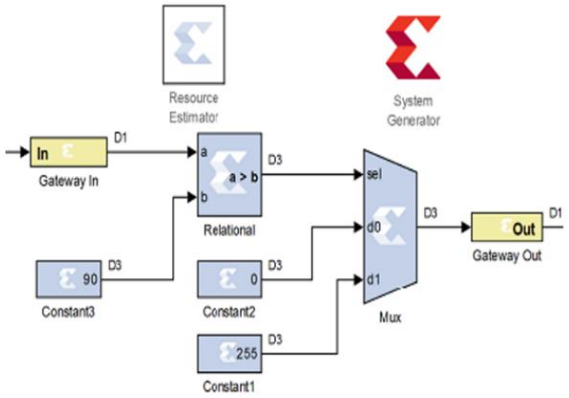

Fig. 14 Algorithm for thresholding image

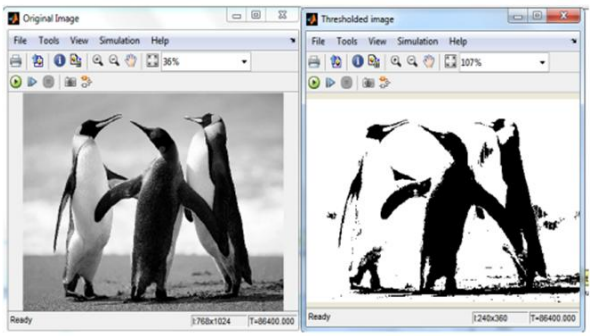

Fig. 15 Result obtained from grayscale image thresholding

E. Background Image subtraction

The AddSub blocks are used to obtain images in figure 16 and the results obtained are shown in figure 17.

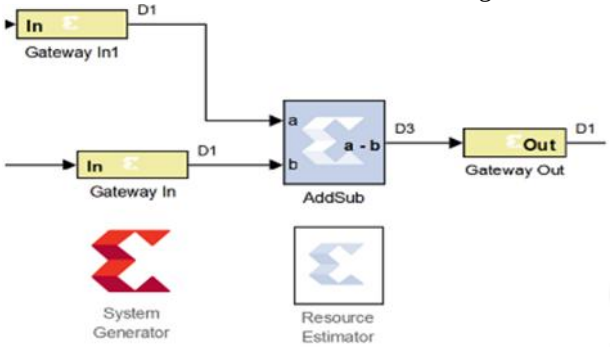

Fig. 16 Algorithm of background subtraction of image

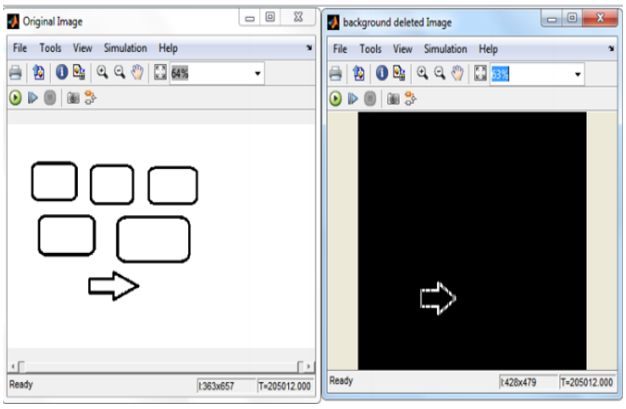

Fig. 17 Result of background subtraction

F. Dilation

Within output image, the shape of structure element replaces each pixel with dilation. The operation of dilation is indicated in figure 18 and 19.

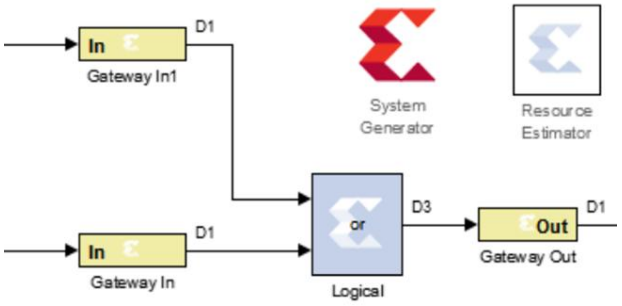

Fig. 18 Dilation algorithm

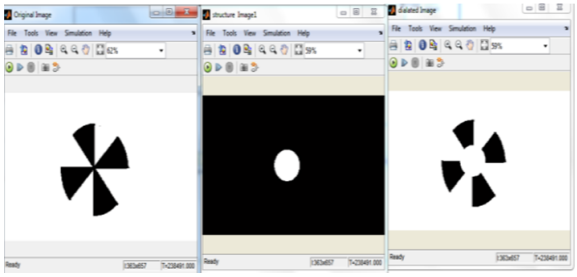

Fig. 19 Result of Dilation Algorithm

G. Co-Simulation of Software/Hardware in System Generator System

A design which runs on FPGA is directly simulated on Simulink by hardware co-simulation provided by system generator. After simulating the result in Simulink, FPGA hardware calculates the compiled portion results. This leads to faster time of simulation as well as accuracy of hardware is also checked.

VI. CONCLUSION

The integration of Matlab Simulink with Xilinx System Generator for image processing algorithms in real time. The FPGA verification uses hardware co-simulation.

REFERENCES

1. R. C. Gozalez and R. E. Woods, “Digital Image Processing, 3rd edition,” *IEEE Trans. Biomed. Eng.*, 2013.
2. A. A. Bazil Raj and A. A. Bazil Raj, “Digital Signal Processing with Field-Programmable Gate Array,” in *FPGA-Based Embedded System Developer’s Guide*, 2018.
3. N. P. R. Neha. P. Raut, “FPGA Implementation for Image Processing Algorithms Using Xilinx System Generator,” *IOSR J. VLSI Signal Process.*, 2013.
4. A. M. Sánchez, R. Alvarez, and S. Sánchez, “Architecture for filtering images using Xilinx System Generator,” *Int. Filter. Images using Xilinx Syst. Gener.*, 2007.
5. P. Turcza and M. Duplaga, “Hardware-efficient low-power image processing system for wireless capsule endoscopy,” *IEEE J. Biomed. Heal. Informatics*, 2013.
6. S. Bauer, S. Köhler, K. Doll, and U. Brunsmann, “FPGA-GPU architecture for Kernel SVM pedestrian detection,” in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Workshops, CVPRW 2010*, 2010.
7. M. Balaji and S. A. Christe, “FPGA implementation of various image processing algorithms using xilinx system generator,” in *Smart Innovation, Systems and Technologies*, 2015.