Hardware in loop emulation of efficient Morphological image processing operations on Xilinx FPGA

Manne Archana, J. Venkateswara Rao

***Abstract*—Image processing has a wide application range like satellite images capturing and morphing of images. We have the different morphological image processing techniques and these are all implemented by mathematical expressions.**

In this project we present an FPGA (Field Programmable Gate Array) based optimized architecture for morphological image processing. The signal processing architecture is able to reuse intermediate results of 2D morphological operations, can handle different shapes of structuring elements and uses only a minimum amount of FPGA logic and block RAM resources for intermediate image line storage. By using system generator tool we are implementing this concept instead of sample level description of the project. By using the system generator input image will be transferred from PC to FPGA board and after performing the required filtering/processing the output image will be transferred back to PC. In PC both the images will be displayed and results will be validated. The HDL implementation uses basic blocks registers, adders, multipliers, control logic, system generator etc.

The Xilinx System generator tool will be used to test the simulation results as well as the FPGA inside results while the logic running on FPGA.

***Index Terms*—VHDL, System Generator (MATLAB, XILINX), Morphological Image Processing, Simulink.**

I. INTRODUCTION

An image may be defined as a two-dimensional function, f(x, y), where x and y are spatial (plane) coordinates, and the000 amplitude of f at any pair of coordinates (x, y) is called the intensity or gray level of the image at that point. When x, y, and the amplitude values of f are all finite, discrete quantities, we call the image a digital image. The field of digital image processing refers to processing digital images by means of a digital computer. Note that a digital image is composed of a finite number of elements, each of which has a particular location and value. These elements are referred to as picture elements, image elements, pels, and pixels. Pixel is the term most widely used to denote the elements of a digital image. Vision is the most advanced of our senses, so it is not surprising that images play the single most important role in human perception. However, unlike humans, who are limited to the visual band of the electromagnetic (EM) spectrum, imaging machines cover almost the entire EM spectrum, ranging from gamma to radio waves. They can operate on images generated by sources that humans are not accustomed to associating with images. These include ultra-sound, electron microscopy, and computer-generated images. Thus, digital image processing encompasses a wide and varied field of applications.

II. MORPHOLOGICAL IMAGE PROCESSING

The word morphology commonly denotes a branch of biology that deals with the form and structure of animals and plants. We use the same word here in the context of mathematical morphology[1] a tool for extracting image components that are useful in the representation and description of region shape, such as boundaries, skeletons, and the convex hull. We are interested also in morphological techniques for pre- or post processing, such as morphological filtering, thinning, and pruning.

The language of mathematical morphology is set theory. As such, morphology offers a unified and powerful approach to numerous image processing problems. Sets in mathematical morphology represent objects in an image. The below figure 1 shows block level implementation of the morphological operations.

III. BLOCK DIAGRAM

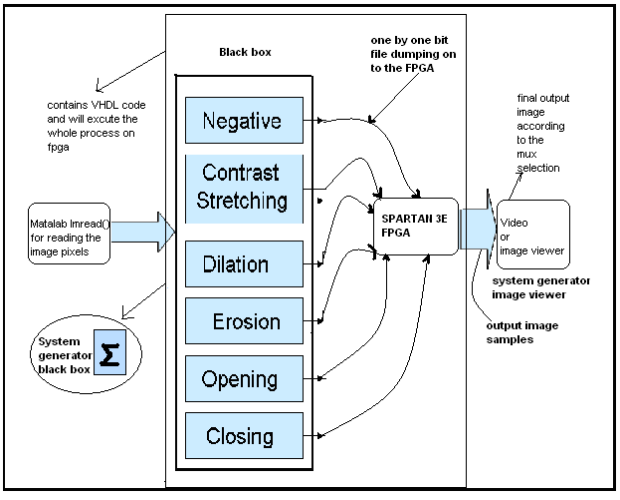

Fig. 1 Block Diagram of Morphological operations.

ISSN: 2278 – 7798

The block diagram indicates image morphological operations, which are Negative, Contrast, Dilation, Erosion, Opening, Closing. These operations will be designed individually by system generator tool. So by this system generator concept we can see the both input and output images of the each and every block.

IV. System Generator

Now a day's Field Programmable Gate Array (FPGA) technology is having more importance for the algorithms which are suited to DSP applications. Field-programmable gate arrays (FPGAs) are non conventional processors built CLB's(configurable logic blocks) and these are connected by programmable interconnectors. Each CLB is having one or more lookup tables (LUTs) and memory. Hardware can be implemented on FPGA by interconnecting the CLB's, internally logic gates and registers form the circuit diagrams onto CLB's. CLB's having slices, slices having LUT's, flipflops, logic gates, muxes.

MATLAB Simulink environment is having libraries which are having the built in blocks. By using this built in block we can have a design depends on application for modeling, simulating, and analyzing the dynamical systems. For DSP operations a tool called Xilinx System Generator (XSG) which offers block libraries that plugs into Simulink tool. This tool containing accurate models of the FPGA's of particular math, logic, and DSP functions.

V. Xilinx System Generator

Xilinx System Generator[2] is a DSP design tool from Xilinx. By using this tool we can have a model-based design for the FPGA hardware design. By using the Xilinx specific block set we can capture DSP friendly simulink modeling environment by this system-level modeling tool. Xilinx block set extends Simulink in many ways to provide a modeling environment that is well suited for hardware design on FPGA. The tool provides high-level abstractions that are automatically compiled into an FPGA at the push of a button. All the operations means the implementation steps which are synthesis, mapping, place & route will perform automatically to generate a bit stream file of programming file. Xilinx block set contains 90 DSP building blocks for simulink. By these Xilinx DSP blocks the design can be optimized for the selected device. Below figure 2 and 3 shows the system generator symbol and system generator black box.

Fig. 2 System generator symbol.

Fig. 3 Black Box Symbol.

A. References

In system generator we have the black box environment. The black box is a Xilinx system generator block which it incorporates hardware description language (HDL) models into System Generator. In system generator this block provides both the simulation behavior in Simulink and the implementation files to be used during code generation. The ports which are mentioned in HDL file and attached to the black box produce and consume the same sorts of signals as other System Generator blocks. When a black box is translated into hardware, the associated HDL entity is automatically incorporated and wired to other blocks in the resulting design.

Either VHDL or Verilog can be incorporated with the black box into a Simulink model. The HDL which is incorporated with the Black box can be co-simulated with Simulink using the System Generator interface to either ISE simulator or the ModelSim simulation software.

The below figure 4 shows the hardware co simulation environment on FPGA.

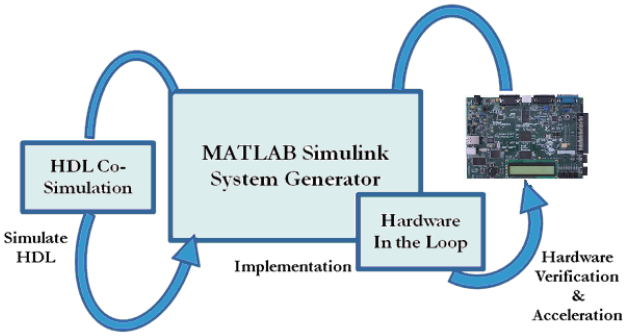

Fig. 4 FPGA Based Hardware-Software Co-Simulation Environment.

A. Hardware Co-Simulation

The word Hardware co-simulation will comes at system generator environment. Hardware co-simulation is a hardware design technique. This technique offers that the simulation of the design into a hardware form. The design can be used to loaded and verified by this technique whether it is correctly working or not on the FPGA. ISim is used for complementary flow to the software-based HDL simulation.

B. Hardware-in-the-loop

Hardware-in-the-loop (HIL) simulation is a technique which is used for the development and test of complex real-time embedded systems. In general the simulation

ISSN: 2278 – 7798

process refers to the testing the design by applying different test inputs. In the same way in this hardware-in-the-loop concept we are testing our design before going to the hardware implementation on FPGA means hardware co-simulation process.

C. *FPGA Structure*

General Architecture of the Field Programmable Gate Array (FPGA) will be shown in below figure 5.

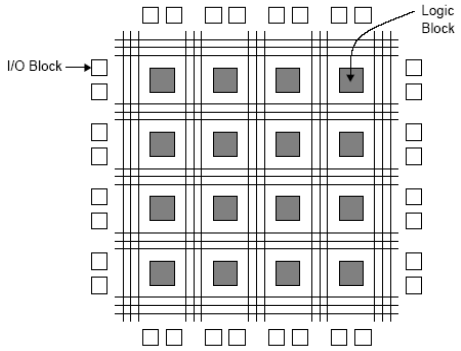

Fig. 5 FPGA structure

VI. Simulation And Hardware Results

The below figure shows the system generator model file for the application level implementation.

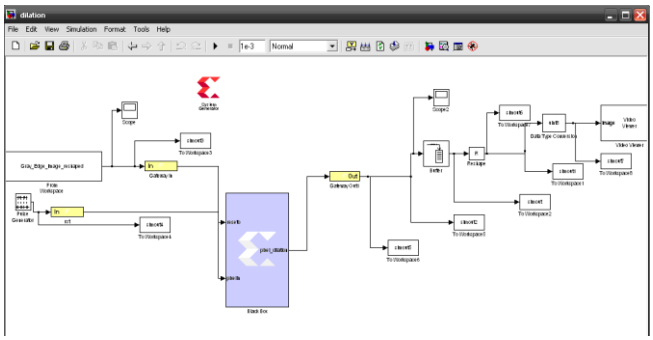

Fig. 6 System Generator Model File.

The below figures 7(a), 7(b), 7(c), 7(d), 7(e), 7(f), 7(g) shows input image and the output images of the morphological image processing applications like negative, contrast, Erosion, Dilation, Opening, Closing. And the first image is the input image which we are giving to the system generator.



Fig. 7(a) Input Image 7(b) Fig Negative Fig. 7(c) Contrast



Fig. 7(d) Erosion Fig. 7(e) Dilation Fig. 7(f) Opening

Fig. 7(g) Closing

VII. Conclusion

Here we are giving the proposed method as compared to the existing one means sample level description of the image output pixels is, we introduced the new concept called system generator. By using this system generator concept we can see the both input and output images. And here we did the morphological operations of the image like negative, contrast, erosion, dilation, opening and closing operations by using system generator black box environment.

REFERENCES

[1] D. Baumann, J. Tinembart, "Designing Mathematical Morphology Algorithms on FPGAs: An Application to Image Processing", Computer Analysis of Images and Patterns, Springer LNCS, Volume 3691, pp. 562–569, 2005.

[2] Xilinx System Generator User's Guide, www.xilinx.com.

Manne Archana is presently pursuing final semester M. Tech in VLSI at Vignan Institute of technology and science.

J. Venkateswara Rao, M.tech, working as Associate Professor in the department of Electronics and Communication Engineering in Vignan Institute of technology and science.

ISSN: 2278 – 7798