Design and Co-Simulation of Depth Estimation Using Simulink HDL Coder and Modelsim

FARIDA MEMON*, AAMIR HUSSAIN MEMON**, SHAHNAWAZ TALPUR***,
FAYAZ AHMED MEMON****, AND RAFIA NAZ MEMON*****

RECEIVED ON 25.02.2016 ACCEPTED ON 11.05.2016

ABSTRACT

In this paper a novel VHDL design procedure of depth estimation algorithm using HDL (Hardware Description Language) Coder is presented. A framework is developed that takes depth estimation algorithm described in MATLAB as input and generates VHDL code, which dramatically decreases the time required to implement an application on FPGAs (Field Programmable Gate Arrays). In the first phase, design is carriedout in MATLAB. Using HDL Coder, MATLAB floating- point design is converted to an efficient fixed-point design and generated VHDL Code and test-bench from fixed point MATLAB code. Further, the generated VHDL code of design is verified with co-simulation using Mentor Graphic ModelSim10.3d software. Simulation results are presented which indicate that VHDL simulations match with the MATLAB simulations and confirm the efficiency of presented methodology.

Key Words: Depth Estimation, Shape From Focus, Hardware Description Language Coder, Field Programmable Gate Arrays.

1. INTRODUCTION

Within the broad area of CV (Computer Vision), depth recovery approaches have been extensively developed and gained significant attention over recent decades. The recovered depth information is used in various applications such as component inspection, robotic manipulations, autonomous vehicle guidance, and 3D endoscopy. Most solutions of the 3D reconstruction problem have been implemented in software running on general purpose processors. Software implementation of most image processing algorithms is associated with various limitations like processing speed and portability [1]. Real-time image processing is an emerging trend in the field of CV. FPGAs are often used as a hardware platform for implementing the algorithms. FPGAs have become a preferred choice due to their re-configurability, high performance and low power consumption [2]. Hardware implementation provides greater speed than software, but associated time lengthens the development of hardware design. FPGA implementations are often written in the HDLs using VHDL or Verilog, which take a very long implementation time.

* Department of Electronic Engineering, Mehran University of Engineering & Technology, Jamshoro.
** Institute of Information & Communication Technology, Allama II Kazi Campus, University of Sindh, Jamshoro.
*** Department of Computer Systems Engineering, Mehran University of Engineering & Technology, Jamshoro.
**** Department of Computer Systems Engineering, Quaid-e-Awam University of Engineering, Science & Technology, Nawabshah.
***** Department of Information Technology, Quaid-e-Awam University of Engineering, Science & Technology, Nawabshah.

Model based hardware design has become very popular in recent years. This design methodology provides a framework for the integration of different phases of the development process. It reduces the lead time to create a self-sufficient model. The design phases associated with the model based design allow the designer to locate and correct errors prior to system prototyping. Thus, model based design effectively serves as a tool for rapid prototyping, system validation and testing.

In this research an effective approach for FPGA implementation of 3D shape recovery algorithm using model based design methodology is proposed. In order to achieve required objective MATLAB/Simulink which includes the HDL Coder tool is used. It saves a significant time and provides optimization options to achieve predefined performance benchmarks. A novel framework based on model based design is developed to generate hardware described in VHDL from a MATLAB algorithm of an efficient depth estimation algorithm using passive SFF (Shape From Focus) method. Simulink HDL Coder toolbox is used for translating M-code into VHDL code. Subsequently, the generated VHDL code of design is verified with co-simulation using Mentor Graphic ModelSim software. A MATLAB System Object and ModelSim are used to verify a RTL (Register Transfer Level) design of a depth estimation algorithm. With HDL Verifier software by using co-simulation wizard, a co-simulation model of design is created. Simulation results indicate that VHDL simulations match with the MATLAB simulations and accurate depth estimation is achieved.

2. BACKGROUND OF DEPTH ESTIMATION

Estimating depth from the 2D images is a key issue in CV. Now-a-days, many advanced applications require 3D data. The depth information plays an important role in the analysis of dynamic or static environments. It is the fundamental clue for other relevant machine vision and AI (Artificial Intelligence) applications such as recognition, classification and modeling. Real world applications also include the surveillance and robotic field which exploit depth information to gain better environmental analysis. In the medical field, the emerging research trends also require reliable depth data. Moreover, the domains of 3D image processing, digital photography, multimedia, 3D visualization and augmented reality utilize 3D information largely.

SFF is one of the passive optical method used to retrieve the depth of the object. In this method an image set of object is acquired with various focus positions [3-5]. The purpose is to find the best focused point through the image stack and constitute a depth map. SFF technique contains two key stages: focus operator and reconstruction technique [6]. The basis of SFF methods is the focus measure which measures the focus quality. The essential idea underlying practical measures of focus quality is to respond high-frequency content in the image. Focus operator is used to calculate the degree of focus in the image sequence having different focus positions. Ideally, it should produce maximum response when the image area is perfectly focused. Once a sequence of images is acquired at different focal positions and the relative degree of focus is measured for all their pixels using some focus measure operators, second stage in SFF applies a reconstruction scheme that uses the focus information of those images in order to estimate the depth of every point of the scene. The results of the focus measures are refined using such a reconstruction scheme.

In literature several focus operators are proposed in spatial and frequency domains. Among which many focus operators are based on image derivatives. Tenenbaum [7], has presented a focus operator based on image gradient. Subbarao and Choi [8,9] developed and evaluated a group of focus measures using band-

pass filtering analysis. Various focus operators are proposed by researchers using the Laplacian operator. Nayar and Nakagawa [5] have developed the focus measure based on SML (Sum-of-Modified-Laplacian) operator. Several focus operators are developed in [10,11] using the statistical analysis of image intensities. Krotkov [12] has presented the GLV (Gray Level Variance) operator using image intensities. In addition, some focus operators are also proposed by researchers in the transform domain. Baina and Dublet [13] developed a focus operator by utilizing the energy of the AC part of discrete cosine transform. Whereas Kubota and Aizawa [14] have presented two focus operators in the wavelet domain. Once a focus profile is achieved through some focus criteria, second step in SFF uses a reconstruction technique to compute the depth of every point of scene. To constitute a more refine depth map several reconstruction techniques have been proposed in literature. Numerous researchers have utilized interpolation techniques for scene reconstruction [5,15,16]. In addition, Subbarao and Choi [17] have developed the FIS (Focused Image Surface) using planar surface approximations. Some researchers have used reconstruction techniques taking account for continuous nature of the imaged object. The proposed techniques include neural networks [18], and dynamic programming [19], among others [20-22].

In summary, it is analyzed that the existing and available developments are only confined to simulation extent without taking into account any hardware implementation. This research is based on a model based hardware implementation of depth estimation of objects on a FPGA platform. The SFF method is used to estimate the depth by searching for the best focused scene points from image set which are taken with different focus settings.

In this work, depth estimation algorithm is developed that models the XSML (Cross-Sum-Modified-Laplacian) focus measure variations at each point as a Gaussian interpolation to obtain more accurate depth estimates than existing developed methods [5,7,12]. XSML incorporates diagonal points in calculation and hence performs better than the SML operator [5].

3. MODEL BASED HARDWARE DESIGN WITH SIMULINK HDL CODER

Model based design with Simulink HDL Coder offers an opportunity of obtaining HDL code without handwriting by using an automatic code generation process [24]. HDL Coder [25] can be used to produce HDL code from MATLAB code, Simulink models and Stateflow finite-state machines. It provides a workflow advisor which can be used to automate the programming of Xilinx and Altera FPGA devices. Subsequent design and trial process is completely iterative and the designer can regress to original model, carry out alterations concurrently and stimulate code at any stage.

Using Coder, designers and system architects may utilize more time on fine-tuning algorithms; system prototyping, testing and less time on HDL coding. Simulink HDL Coder can also control the HDL architecture, implementation and produce resource utilization reports [26]. For rapid verification of generated codes, Coder generates test benches and EDA (Electronic Design Automation) co-simulation models, and also provides code traceability [27]. Once the testing phase of design is accomplished, after that generated HDL code can be synthesized for hardware realization. Finally, the procedure is followed to obtain the bit file of the design that can be downloaded to FPGA boards.

4. DESIGN METHODOLOGY

A model based design using HDL Coder is proposed in order to develop hardware design of depth estimation on

a FPGA platform. The proposed design is an alternative to traditional FPGA design flow using manual HDL coding which is tedious, error prone and highly time consuming. The model based design flow of depth estimation algorithm using HDL Coder is shown in Fig. 1. Methodology includes designing of depth estimation algorithm in MATLAB, generating VHDL code from MATLAB algorithm using HDL Coder and verification with co-simulation using ModelSim simulator.

4.1 MATLAB Algorithm Design

The MATLAB design of depth estimation algorithm is initiated to produce the depth map and focused image of a simulated cone object from a series of 25 defocused images with dimensions of 200x200 pixels. For all images in the sequence, a XSML focus measure is computed at each pixel using a 7x7 image neighborhood across the pixel. Thus, the set of focus measures are achieved at each image point and the image having highest focus value is found. Finally, Gaussian interpolation technique is applied to interpolate the computed focus values and obtained the refined depth map. Besides this, from the obtained focus profile, all in focused image is created. In the SFF method, using any focus operator best image focused points from the image set are calculated. Then through reconstruction technique by utilizing this focus profile, all in focused or full focused 2D image and depth map can be retrieved.

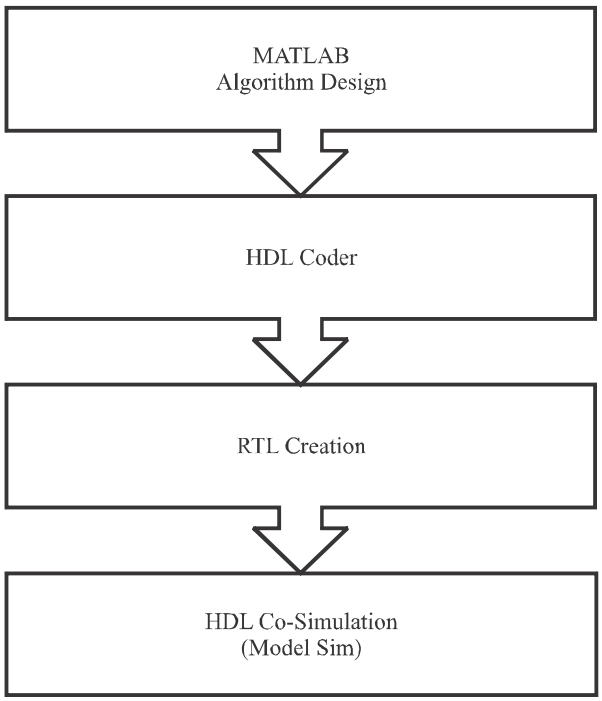


FIG. 1. MODEL BASED HARDWARE DESIGN FLOW OF DEPTH ESTIMATION ALGORITHM

The complete depth estimation algorithm is illustrated in Fig. 2.

4.2 MATLAB to VHDL Workflow

The overview of the MATLAB to VHDL generation workflow of depth estimation algorithm is given in Fig. 3.

4.2.1 Prepare

HDL Coder permits a subset of the M-code for code generation with limitations in the usage of functions, operators and syntax in the program. Therefore, the depth estimation algorithm is prepared with supported language features. Using HDL Coder VHDL code generation process starts by splitting the MATLAB model into function-file and test-bench. The function-file is a MATLAB design function which is to be converted into VHDL, while a test bench is a script file used to verify the algorithm in the MATLAB function.

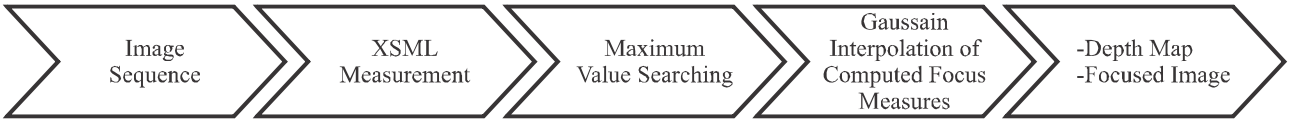


FIG. 2. THE DIAGRAM OF DEPTH ESTIMATION ALGORITHM

4.2.2 Verify Floating-Point

The MATLAB floating-point model of design is simulated before the generation of VHDL code, to verify that it runs and offers a baseline to compare with the generated VHDL code. Coder executes the test bench of floating point model and shows all plots, variables and outputs of design. Errors in this model will propagate through all subsequent steps and present in the final bit stream.

4.2.3 Generate Fixed-Point

When targeting hardware, the floating point models need to be transformed to use fixed-point operations for power, cost, and performance reasons. The framework here produces a fixed-point model from the floating-point design using MATLAB test bench. Next, the objective is to describe an optimized fixed-point model which minimizes the code size and the execution time of design. This optimization is achieved using Fixed-point advisor tool through the selection of different word lengths and fraction lengths for data variables. From optimization process to achieve fixed-point model equivalent to floating-point, word length and fraction length values of 14 and 4 are chosen respectively. The following files of design are produced during this conversion process as shown in **Table 1**.

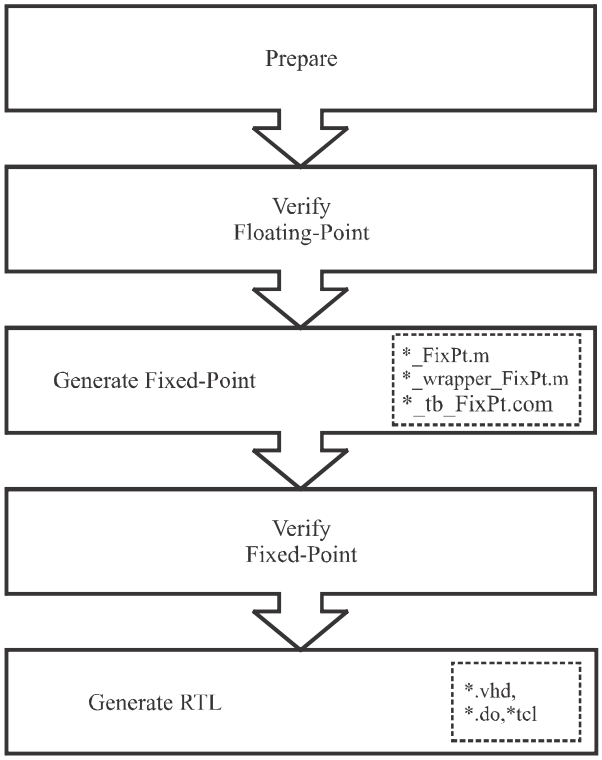


FIG. 3. VHDL GENERATION WORKFLOW

4.2.4 Verify Fixed-Point

In this step fixed-point model of design is executed and subsequent output is compared with floating-point design output. The design is annotated with more directives if the obtained results are not satisfactory. This iteration is performed until corresponding results are achieved.

4.2.5 Generate RTL

After the verification of fixed-point model with the corresponding floating point design, Coder generates a RTL model from the fixed-point MATLAB code. In addition to generating VHDL code, a test-bench is also created to verify the generated code. Following files of the design are produced during this step as shown in Table 2.

TABLE 1. GENERATED FILES IN THE FIXED-POINT CONVERSION

File Name	Description
depth_FixPt.m	fixed-point file of design
depth_wrapper_FixPt.m	a design wrapper file
depth_tb_FixPt.m	fixed-pointtest-bench

TABLE 2. GENERATED FILES DURING RTL CREATION

File Name	Description
depth.vhd	VHDL code of the design
depth_tb.vhd	test-bench of the design
depth_synplify.tcl	Synplify synthesis script
depth_compile.do	ModelSim compilation script
depth_tb_sim.do	ModelSim simulation script

4.3 HDL Co-Simulation

HDL Coder supports system level verification using co-simulation to verify the functionality of RTL implementation. With HDL Verifier software integrated with HDL Coder, system-level test benches are reused to perform co-simulation. HDL Verifier includes the co-simulation wizard through which co-simulation interface for a given HDL simulator can be generated. The co-simulation wizard creates a customized MATLAB function, MATLAB System Object, or Simulink HDL co-simulation block interfaces used for co-simulation with a supported HDL simulator. The co-simulation diagram for depth estimation algorithm is presented in **Fig. 4**.

As shown in Fig. 4, MATLAB System Object and ModelSim tools are used to verify a RTL design of a depth estimation algorithm. With co-simulation wizard, a MATLAB System Object test-bench is created. The co-simulation wizard takes the provided VHDL file of the design as its input. It also collects user input required for setting up co-simulation in each step. The co-simulation wizard generates a MATLAB script that instantiates a configured HDL co-simulation System Object, a MATLAB script that compiles VHDL design, and a MA TLAB script that launches ModelSim for co-simulation.

5. RESULTS AND DISCUSSION

The depth estimation algorithm is tested on a series of 25 gray scale images of simulated cone object each having a dimension of 200x200. Simulated cone object is taken into account owing to its dense texture and can be reconstructed easily. A non-linear model of defocus [6] is utilized to produce image set of a simulated cone object. Defocus has been simulated for a 3.3 mm focal length camera focusing between 50 and 100 mm. The focus sequences generated synthetically allow the availability of a ground truth for an objective estimation of the performance of the image fusion process.

A small selection of image frames and ground truth depth map is presented in **Fig. 5(a-e).** The depth map and focused image of a simulated cone is obtained by applying described depth estimation algorithm presented in **Fig. 6(a-b)**. As compared with ground truth depth map given in Fig. 5(e), it depicts that the proposed method estimates accurate depth map whereas, every part of the image is in sharp focus in the retrieved focused image.

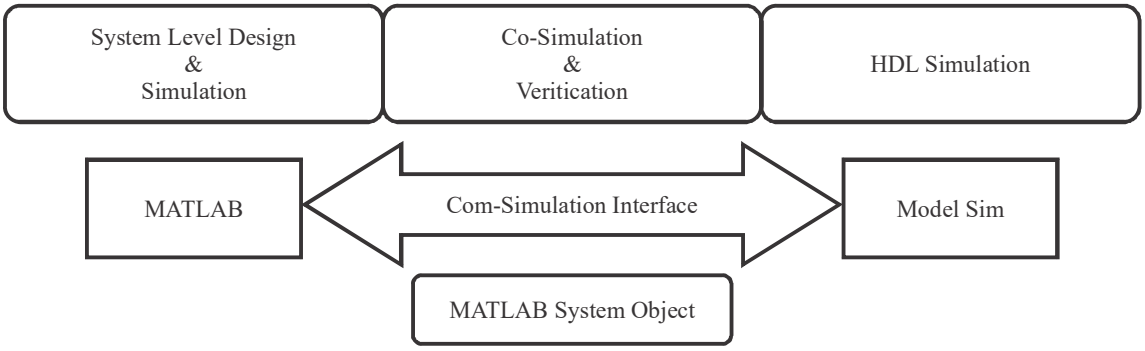
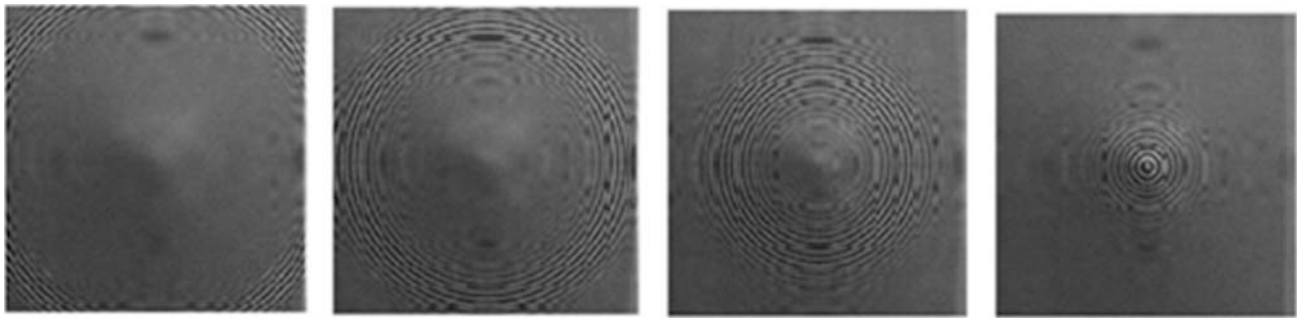
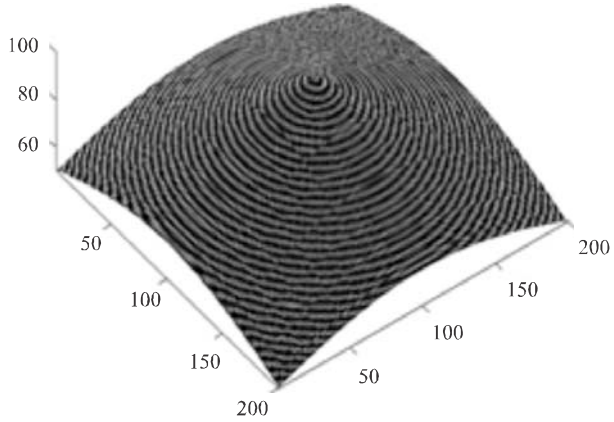


FIG. 4. THE CO-SIMULATION DIAGRAM FOR DEPTH ESTIMATION ALGORITHM

For verification of VHDL design of a depth estimation algorithm, the co-simulation model is executed with ModelSim 10.3dsimulator.The experimental results are represented in ModlSim simulator and also in MATLAB environment as shown in subsequent results. The VHDL simulations of the design in the MATLAB environment are given in Fig. 7(a-b). By comparing VHDL results of the depth map and focused image of a



(a) FRAME5 *(b) FRAME10* *(c) FRAME15* *(d) FRAME20*



(e) GROUND TRUTH DEPTH MAP IMAGE

FIG. 5. IMAGE FRAMES OF A SIMULATED CONE IMAGE SEQUENCE

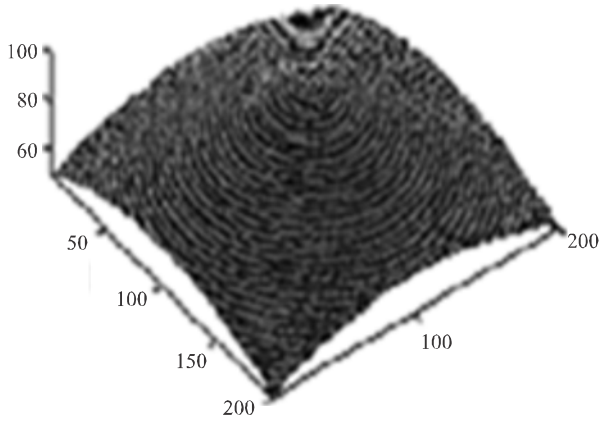


FIG. 6(a). OBTAINED DEPTH MAP

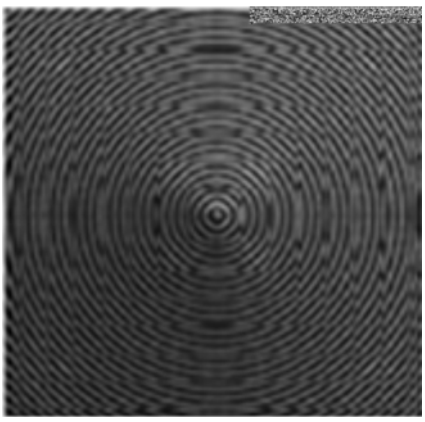
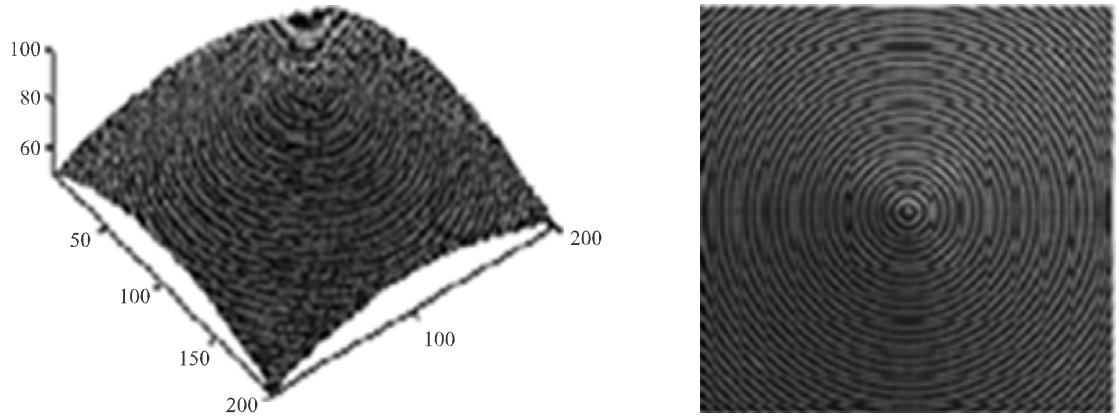


FIG. 6(b). FOCUSED IMAGE

simulated cone object with MATLAB processing shown in Fig. 6, it is found that VHDL simulations match with the MATLAB simulations. Moreover, Fig. 8 presents the slight portion of the inputs and outputs waveforms of the design in the ModelSim wave window .Additional plots for scalar output depth map "Z_{out1}" and focused image "I_{out2}"are given in Figs. 9-10, respectively, where x-axis displays the image matrix values using vector representation and y-axis shows the depth map values in the range of 0.05-0.1m in Fig. 9 and intensity values for an eight-bit image in Fig. 10. Figs. 9-10 show the plots for reference design, co-simulation as well as the difference between the two. From the plots it is clear that, both algorithms are similar.



(a) DEPTH MAP *(b) FOCUSED IMAGE*

FIG. 7. VHDL PROCESSED SIMULATION

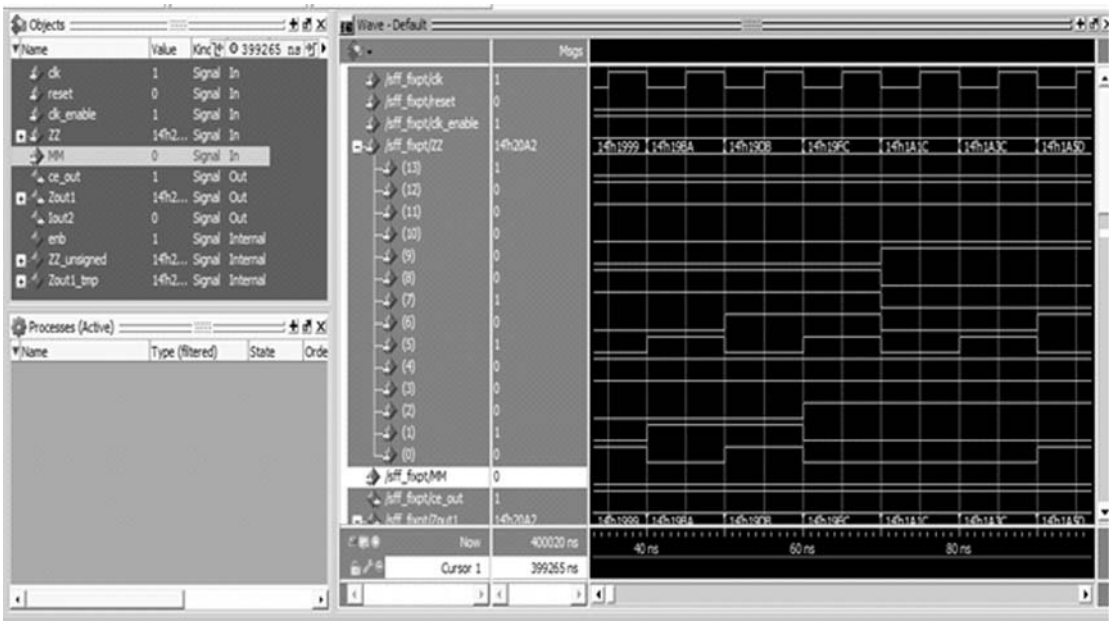


FIG. 8. THE INPUT AND OUTPUT WAVEFORMS OF DEPTH ESTIMATION DESIGN

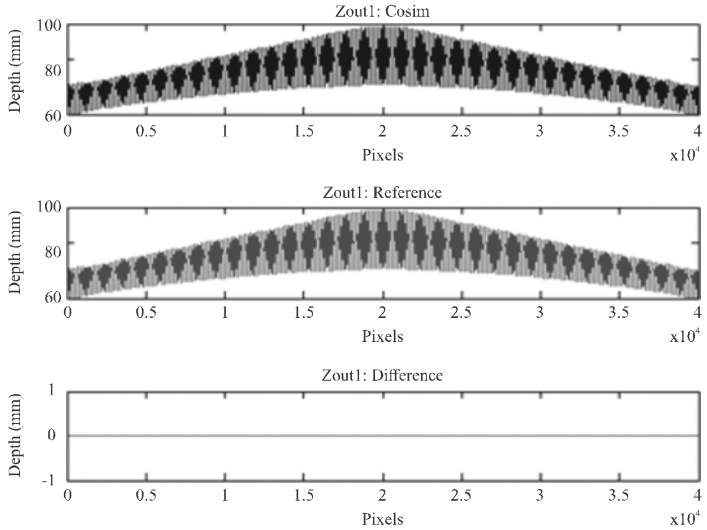


FIG. 9. DEPTH OUTPUT OF THE REFERENCE DESIGN AND MODELSIM

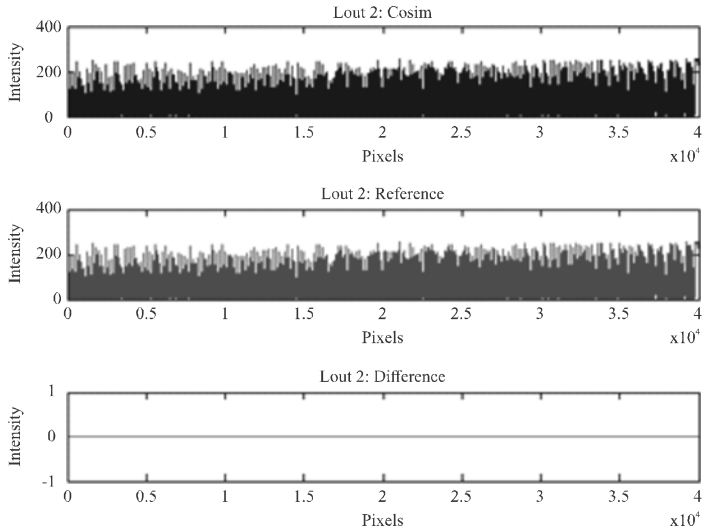


FIG. 10. FOCUSED IMAGE OUTPUT OF THE REFERENCE DESIGN AND MODELSIM

6. CONCLUSION

The modern programmable devices in combination with appropriate software packages for synthesis and simulation give a significantly accelerated design process of electronic systems. Simulink HDL Coder is a simple and useful tool for developing and analyzing the algorithms. This approach for automatic generation of hardware descriptions models and systems saves significant amount of time for code writing, debugging and verification of the generated hardware. A novel frame work is developed which generates a hardware description in VHDL for a depth estimation algorithm from a MATLAB model, suitable to run on an FPGA. The generated hardware has been successively verified functionally by co-simulation using ModelSim software. Experimental results indicate that VHDL estimated depth map is comparable in accuracy to the full precision MATLAB's output and accurate depth is estimated using this method. Due to the drastic reduction of design time, this automatic method of converting an algorithm into a hardware design proved to be feasible. Future work includes implementation and evaluation of generated VHDL design on FPGA architecture.

ACKNOWLEDGEMENTS

This research is supported by Endowment Fund, Institute of Information & Communication Technologies, Mehran University of Engineering & Technology, Jamshoro, Sindh, Pakistan.

REFERENCES

[1] Beveridge, J.R., Bohm, A.P.W, and Ross, C., "Accelerated Image Processing on FPGAs", IEEE Transactions on Image Processing, Volume 12, No. 12, pp. 1543-1551, December, 2003.

[2] Johnston, C.T., Gribbon, K. T., and Bailey, D. G., "Implementing Image Processing Algorithms on FPGAs", Proceedings of 11th Conference on Electronics, pp. 118-123, Palmerston North, New Zealand, November, 2004.

[3] Ens, J., and Lawrence, P., "An Investigation of Methods for Determining Depth from Focus", IEEE Transactions on Pattern Analysis and Machine Intelligence, Volume 15, No. 2, pp. 97-108, 1993.

[4] Lertrusdachakul, I., Fougerolle, Y., and Laligant O., "A Novel 3D Reconstruction Approach by Dynamic (de) Focused Light" , Proceedings of SPIE Conference, Volume 7538, 2010.

[5] Nayyar, S.K., and Nakagawa Y., "Shape from Focus: An Effective Approach for Rough Surfaces", Proceedings of IEEE International Conference on Robotics and Automation, Volume 2, pp. 218-225, 1990.

[6] Pertuz, S., "Analysis of Focus Measure Operators for Shape-from-Focus", Pattern Recognition, pp. 1415-1432, Elsevier, 2013.

[7] Tenenbaum, J., "Accommodation in Computer Vision", Ph.D. Thesis, Stanford University, Stanford, CA, USA, October, 1970.

[8] Subbarao, M., Choi, T., and Nikzad, A., "Focusing Techniques", Journal of Optical Engineering, Volume 32, pp. 2824-2836, 1993.

[9] Subbarao, M., and Choi, T., "A New Method for Shape from Focus", Proceedings of SPIE Conference, Volume 2064, September, 1993.

[10] Wee, C., and Paramesran, R., "Measure of Image Sharpness Using Eigenvalues", Information Society, pp. 2533-2552, 2007.

[11] Xu, X., Wang, Y., Tang, J., Zhang, X., and Liu, X., "Robust Automatic Focus Algorithm for Low Contrast Images using a New Contrast Measure", pp. 8281-8294, Sensors. 2011.

[12] Krotkov, E.P., "Focusing", International Journal of Computer Vision, pp. 223-237, 1987.

[13] Baina, J., and Dublet, J., "Automatic Focus and Iris Control for Video Cameras", Proceeding of 5ht International Conference on Image Processing and its Applications, pp. 232-235, Edinbergh, UK, 4-6 July, 1995.

[14] Kubota, A., and Aizawa, K., "Reconstructing Arbitrarily Focused Images from Two Differently Focused Images using Linear Filters", IEEE Transaction on Image Processing, Volume 14, pp. 1848-1859, 2005.

[15] Zhang, Y., Zhang, Y., and Wen, C., "A New Focus Measure Method using Moments", Image Vision Computing, Volume 18, pp. 959-965, 2000.

[16] Favaro, P., Soatto, S., Burger, M., and Osher, S., "Shape from Defocus via Diffusion", IEEE Transaction on Pattern Analysis and Machine Intelligence, pp. 518-531, 2008.

[17] Subbarao, M., and Choi, T.S., "Accurate Recovery of Three Dimensional Shape from Image Focus", IEEE Transactions on Pattern Analysis and Machine Intelligence, Volume 17, pp. 266-274, March, 1995.

[18] Asif , M., and Choi, T.S., "Shape from Focus using Multilayer Feedforward Neural Networks", IEEE Transactions on Image Processing, Volume 10, pp. 1670-1675, 2001.

[19] Ahmad, M., and Choi, T.S., "A Heuristic Approach for Finding Best Focused Shape", IEEE Transactions on Circuits and Systems for Video Technology, Volume 15, pp. 566-574, 2005.

[20] Shim, S.O., and Choi, T.S., "A Novel Iterative Shape from Focus Algorithm Based on Combinatorial Optimization", Pattern Recognition, Volume 43, pp. 3338-3347, 2010.

[21] Muhammad, M., and Choi, T.S., "Sampling for Shape from Focus in Optical Microscopy", IEEE Transactions on Pattern Analysis and Machine Intelligence, Volume 34, pp.564-573, 2012.

[22] Mahmood, M.T., and Choi, T.S., "Nonlinear Approach for Enhancement of Image Focus Volume in Shape from Focus", IEEE Transactions on Image Processing, Volume 21, pp. 2866-2873, 2012.

[23] Thelen, A., Frey, S., Hirsch, S., and Hering, P., "Improvements in Shape-from-Focus for Holographic Reconstructions with Regard to Focus Operators, Neighborhood-Size, and Height Value Interpolation", IEEE Transactions on Image Processing, Volume 18, pp. 151-157, 2009.

[24] Smith, P., Prabhu, S., and Friedman, J., "Best Practices for Establishing a Model-Base Design Culture", Mathworks Inc., 2007.

[25] "Simulink HDL Coder 1.5 User's Guide", The Math Works, Inc. 2014.

[26] Heo, S.W., Sung, H.K., and Lee, D., "Simulink Model Based Design and FPGA Implementation of Multi-Channel DTV Transmitter", International Journal of Control and Automation, Volume 6, No. 5, pp. 353-360, 2013.

[27 "Simulink HDL Coder", MathWorks, Inc. Available at: http://www.mathworks.in/help/pdf_doc/hdlcoder/hdlcoder_ug.pdf.