View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by Universiti Teknologi Malaysia Institutional Repository

AUTOMATION SCRIPTS FOR GENERIC SYNTHESIS AND CO-SIMULATION FOR XRONOS GENERATED HDLS.

KHALIL ASYRANI BIN SULAIMAN

UNIVERSITI TEKNOLOGI MALAYSIA

AUTOMATION SCRIPTS FOR GENERIC SYNTHESIS AND CO-SIMULATION FOR XRONOS GENERATED HDLS.

KHALIL ASYRANI BIN SULAIMAN

A project report submitted in partial fulfilment of the
requirements for the award of the degree of
Master of Engineering (Computer and Microelectronic Systems)

Faculty of Electrical Engineering
Universiti Teknologi Malaysia

2 JUNE 2016

Specially dedicated to *Mama, Abah* and my beloved wife, Siti Aisyah.

I really miss all of you.

May Allah bless all of you always.

ACKNOWLEDGEMENT

Firstly, I would like to give the greatest gratitude to Dr.Eileen Su Lee Ming from Faculty of Electrical Engineer, Universiti Teknologi Malaysia for the guidance in the past undergraduate which also continue motivating me to completing my postgraduate project – “Automation Scripts for HDL Synthesis and Co-Simulation for Better Optimization on Generated RVC Framework (ORCC) in HDL Tools”, as well as Dr. Al-Hadi that giving me guide in using most of the tools such as Xilinx Vivado. It would be impossible to complete without helps, advices and hospitality given by them.

In addition, I would like to thanks to Chin Yong Huan and Khoo Zhi Yion for helping me in providing walkthtough in HDL Tools. Apart from that, I also would like to give my appreciation to all my friends, specifically Wan, Faiz and Syukri for helping me in testing, giving idea, and feedbacks for this project.

Lastly, for the others that had helped me in this project which I did not mention, their help is appreciated.

ABSTRACT

With growing developments in applications such as multimedia, technologies has pushed boundaries in the demands and popularity which used for a wide range of application domains. However, as complexity of development process of these applications that involve time-consuming and error-prone tasks increases, it is often scaled-up with development of the devices on the market to supports these applications. ORCC has helped in reducing the complexity of application developments by providing a set of modern tools based on dataflow programming. However, though Xronos generated HDLs has proven to be useful, yet it does not optimized to the resource utilization, power and timing analysis. It also yet to have a wide coverage across High-Level Synthesis (HLS) tools such as Altera Quartus and Synopsys Design Compiler. These coverages include the codes generated can be synthesize with other HDL tools. The critical part of this paper is generating of generic HDL for Altera Quartus or Synopsys Design Compiler from original HDL generated from ORCC and to provide co-simulation, automation environments for the RVC-CAL framework of Altera Quartus's testbench by extracting simulation data that available from RVC-CAL framework, in which can be used to verify data from both ends (RVC-CAL's and Quartus's). Apart from that, in order to fully test generated code, comparison will be made with Xilinx Vivado HLS to benchmark functional test that are made with different HLS tools. This paper present the automation scripts that helps to produce better optimization of resource, power, and timing analysis from Xronos generated HDLs and providing automatic testbench that can be tested with Xilinx Vivado and other HDL tools.

ABSTRAK

Sehubung dengan aliran perkembangan dalam aplikasi seperti multimedia, had teknologi telah menembusi sempadan sama ada permintaan mahupun populariti yang menggunakan pelbagai bentuk aplikasi untuk pelbagai domain. Walau bagaimanapun, proses pembangunan aplikasi ini melibatkan masa dan sering kali banyak kesilapan yang berlaku, pembangunan peranti ini selari dengan pasaran bagi menyokong aplikasi ini untuk terus berkembang sekalipun ia rumit. ORCC membantu dalam mengurangkan kerumitan dalam membangunkan aplikasi dengan menyediakan satu set program yang menggunakan pengaturcaraan aliran data. Namun, walaupun terbukti kod HDL yg terhasil dari Xronos berguna, tetapi ia tidak dioptimumkan untuk sumber, tenaga dan kadar masa yang terdapat dalam sesebuah design. Selain itu, ia juga tidak mempunyai liputan luas di dalam alat “High-Level Syntesis” (HLS) seperti “Altera Quartus” dan “Synopsys Design Compiler” yang juga merangkumi kod yang terjana dari “backend Verilog” dengan bantuan Xronos dan ORCC. Bahagian yang penting dalam penyelidikan ini termasuk kaedah menghasilkan HDL generik untuk “Altera Quartus” atau “Synopsys Design Compiler” dari HDL asal dijana daripada ORCC dan untuk menyediakan ruang untuk automasi dan simulasi bagi merangka semula RVC-CAL “testbench” untuk “Altera Quartus” dengan mengekstrak data simulasi yang boleh didapati dari rangka kerja RVC-CAL, di mana boleh digunakan untuk mengesahkan data dari kedua-dua arah (RVC-CAL dan Quartus ini). Selain itu, untuk menguji sepenuhnya kod dijana, perbandingan akan dibuat dengan “Xilinx Vivado HLS” untuk menguji fungsi yang diuji oleh alat HLS yang berbeza. Kertas penyelidikan ini membentangkan skrip automasi yang mampu mengoptimum sumber, tenaga dan kadar masa. Selain itu, skrip ini juga mampu menyediakan “testbench” yang boleh disinthesis oleh Xilinx Vivado dan perisian HDL yang lain.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATION

ASIC	-	Application-Specific Integrated Circuit
FPGA	-	Field-Programmable Gate Arrays
RTL	-	Register Transfer Level
FSM	-	Finite States Machine
SDK	-	Software Development Kits
HLS	-	High-Level Synthesis
NRE	-	Non-recurring Engineering
CAL	-	Cal Actor Language
RVC	-	Reconfigurable Video Coding
ORCC	-	Open RVC-CAL Compiler
EDA	-	Electronic Design Automation
AST	-	Abstract Syntax Tree
ESL	-	Electronic Design Level
LUT	-	Look-up Table
FF	-	Flip Flops
DSP	-	Digital Signal Processing
IOB	-	Input Output Blocks
GP	-	Genetic Programming
TCL	-	Tool Command Language
HDL	-	Hardware Description Language
EOP	-	Event Observability Ports
API	-	Application Programming Interface
QOR	-	Quality of Result
SBSM	-	Software-based Sleep Mode
HBSM	-	Hardware-based Sleep Mode
UVM	-	Universal Verification Methodology

DUV	-	Design under Verification

LIST OF APPENDICES

CHAPTER 1

INTRODUCTION

1.1 Problem Background

With growing complexity in circuit design, the needs to manage resources within electronic circuit such available in FPGA which consisting of LUT, FF, DSP, and IOB is becoming essential in maintaining design specification [1]. Recent discoveries in late 1990's with application-specific integrated circuits (ASICs) indicate designer now can apply describe-and-synthesize methodology to certain level of hardware abstraction [1]. The behavioral model of ASICs now can be describe with register-transfer level (RTL) with algorithms, flow-charts, dataflow graphs or generalize finite state machines (FSMs) in which every state can perform specific complex computations that leads to synthesizable of ASICs at high-level synthesis (HLS) technique [1].

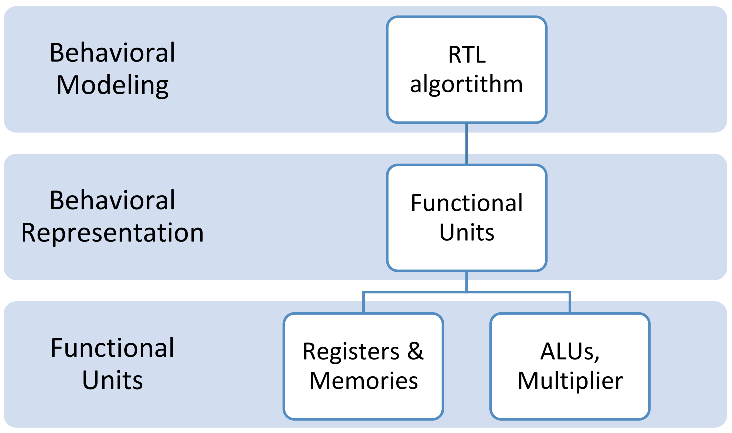

Figure 1.0 : Basic design flow that describe involvement HLS in implementation

HLS can be describe as task that align in a sequence to transforms into a behavioral representation of specified RTL design. From there, design can be break down into functional units such as ALUs, multipliers, and storage units like memories or register files [1]. All of this information can be summarize in the Figure 1.0 in which already assisting designer to solve growing complexity and size of today's design. However, without tackling the verification as parts of the design cycle, it almost quite hard to speed up the verification of the design process itself.

Automation or automatic program in HLS have already exist since 1990's [2-3]. Automation in HLS has become essential as growing complexity and time-to-market pressure that already exist in today's electronic digital systems business [3]. Although there is debate to resist the changes by moving to automatic behavioral synthesis approach as it lack of interactivity and low-quality on the result, but in previous work proves that the HLS tools that combined with genetic programming (GP) can significantly improve the quality of the design as well as exploring the potential of the design space [3]. Thus, HLS will continue evolve the potential in improving the IC design to new level.

HLS tools already can implement advance automation of functional verification by integrating the source code which function as the references and generated design as the device under test into one testbench which can also be generated together in the design. With further exposure of automation in the flow, many optimization beside verification such as resource managements also can be fully utilize to reduce power consumption [5]. Thus, by reducing the amount of power consumed in a circuit, many possibility can be achieved such as reduction in total system cost and prolong battery life for mobile-based devices [3].

Although field programmable gate arrays (FPGAs) provide many benefits over ASICs such as reduced non-recurring engineering (NRE) and consume less time for marketing. But, when implementation of design take place on FPGAs, it come with cost for degrade performance, increase in silicon area, as well as power consumption [4]. It is important to understand some application are constraints to certain power specification and resource utilization [2]. Thus, ASICs is more reliable when advocating design to specific or dedicate function and implementation.

An introduction to Open RVC CAL Compiler (ORCC) with improvements from Xronos, has provided back-end support to generate from CAL languages, which a form of HLS to obtain specifically Xronos-based Verilog HDL backends [6]. From generated backends, although it is fully synthesizable to Xilinx Vivado's compiler, there are drawback of on generated backends, as the Verilog is not highly optimize and contain bloated signals in which need further optimization to prevent wastage on resources, power and timing degradation on synthesized design.

1.2 Problem Statement

Normally, the generated Xronos Verilog from the ORCC is not fully optimizing the resources usage for synthesize the RTL into netlist, in which will be describe in layout level. It is very important to have some modification to the generated codes in which helps to improve the quality of design aspect such as power consumed by the circuit during testing or simulation phase. Another feature that is not available for other HDL tools is the testability on functional unit of the generated block. The generated codes only provides testbench that can be used for Xilinx Vivado's TCL command window in which create less robustness to test this testbench on other HDL tools. It is best to obtain a generic testbench that can synthesizable and tested by other HDL tools.

Generic HDL description is very essential in today's design in obtaining the fully functional hardware description that synthesizable into the netlist which can be use in testing or simulating the result across other platform of HDL tools. These generic HDL description are such as System Verilog, Verilog or VHD. The limitation of current tools to generate such robustness it is difficult to test the ASICs-based synthesis to any FPGA-based HDL. Thus, the need of Cal Actor Language (CAL) from RVC-CAL to be able generated into generic HDL is very critical in deliver the significant benchmark ASICs hardware model into FPGA simulation such that several modification need to perform in order to fully support other HDL tools besides Xilinx Vivado.

1.3 Research Objectives

The objectives of this research focuses on three main issues derived from the problem statement of this project;

1. To create automation that helps in optimize resource utilization by more than 5% specifically at IO’s of generated HDL.
2. To obtain better resource, power, and timing analysis from generated HDL.
3. To provide co-simulation automation environments for RVC-CAL framework of Altera Quartus’s testbench.
4. To generate generic HDL for Altera Quartus II, Mentor Graphic ModelSim or Synopsys Design Compiler from original HDL generated from ORCC.

1.4 Research Hypothesis

These are hypothesis outlined is to proves the objectives that presented:

1. Generated HDL from ORCC now is better than 5% resource utilization specifically at IO’s and has significant improvements in timing, and power consumption.
2. Testbench can be generated for Altera Quartus’s testing and simulation.
3. Generated HDL from ORCC can be port over across HDL tools for synthesis, analysis and simulation

1.5 Research Scope

In order to fulfill the needs and specification of this developments automated scripts, several scope are focused within this research or development. Firstly, as for fundamental needs, few theoretical studies on Open RVC-CAL Compiler (ORCC) as HLS that can produce backends for various application such C and Xronos Verilog and how it can be synthesized to targeted HDL tools such Xilinx Vivado. Secondly, this developments will use majorly Verilog HDL as baseline of hardware description language. The automated scripts that will be written in either Tool Command Language (TCL) or Perl to optimize the generated HDL from ORCC that uses the Xronos Cal Actor Language (CAL) application such as AddArray and JPEG Encoder.

The assessment will cover two phase. As for phase one, the identification of the sub-block that are not synthesizable by Quartus II compiler or simulator, followed by the optimization in which break down into top modules and sub modules. The further description are stated in methodology section. Part two will cover the development of scripts in generating testbench that synthesizable in HDL tools. By taking top modules as references for the testbench structuring, the automation involve is to get all the necessary information. The generated testbench are also will be further describe in methodology section.

REFERENCES

1. Gajski, Daniel D., et al. High—Level Synthesis: Introduction to Chip and System Design. Springer Science & Business Media, 2012.

2. Roman, Gruia-Catalin. "A taxonomy of current issues in requirements engineering." Computer 18.4 (1985): 14-23.

3. Araújo, Sérgio G., et al. "Optimized datapath design by evolutionary computation." System-on-Chip for Real-Time Applications, 2003. Proceedings. The 3rd IEEE International Workshop on. IEEE, 2003.

4. Kuon, Ian, et al. "Measuring the gap between FPGAs and ASICs." Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on 26.2 (2007): 203-215.

5. Wipliez, Matthieu. Compilation infrastructure for dataflow programs. Diss. INSA de Rennes, 2010.

6. Chavarrias, M., et al. "A multicore DSP HEVC decoder using an actor-based dataflow model." Consumer Electronics (ICCE), 2015 IEEE International Conference on. IEEE, 2015.

7. Yviquel, Hervé, et al. "Orcc: Multimedia development made easy."*Proceedings of the 21st ACM international conference on Multimedia*. ACM, 2013.

8. Meeus, Wim, et al. "An overview of today’s high-level synthesis tools." Design Automation for Embedded Systems 16.3 (2012): 31-51.

9. Bezati, Endri, et al. "High-level dataflow design of signal processing systems for reconfigurable and multicore heterogeneous platforms." Journal of real-time image processing 9.1 (2014): 251-262.

10. Monson, Joshua S., et al. "Using Source-Level Transformations to Improve High-Level Synthesis Debug and Validation on FPGAs." Proceedings of the 2015 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays. ACM, 2015.

11. Simpson, Philip. *FPGA design*. Springer, 2010.

12. McFarland, Michael C., et al. "The high-level synthesis of digital systems." Proceedings of the IEEE 78.2 (1990): 301-318.

13. Zuo, Wei, et al. "New solutions for system-level and high-level synthesis." Integrated Circuits (ISIC), 2014 14th International Symposium on. IEEE, 2014.

14. Hemmert, Karl S. Source Level Debugging of Circuits Synthesized from High Level Language Descriptions. Diss. Brigham Young University, 2004.

15. Calagar, Nazanin, et al. "Source-level debugging for FPGA high-level synthesis." Field Programmable Logic and Applications (FPL), 2014 24th International Conference on. IEEE, 2014.

16. Goeders, Jeffrey, et al. "Effective fpga debug for high-level synthesis generated circuits." Field Programmable Logic and Applications (FPL), 2014 24th International Conference on. IEEE, 2014.

17. Curreri, John, Greg Stitt, et al. "High-level synthesis techniques for in-circuit assertion-based verification." Parallel & Distributed Processing, Workshops and Phd Forum (IPDPSW), 2010 IEEE International Symposium on. IEEE, 2010.

18. Ben Hammouda, Mohamed, et al. "A design approach to automatically synthesize ansi-c assertions during high-level synthesis of hardware accelerators." Circuits and Systems (ISCAS), 2014 IEEE International Symposium on. IEEE, 2014.

19. Monson, Joshua S., et al. "New approaches for in-system debug of behaviorally-synthesized FPGA circuits." Field Programmable Logic and Applications (FPL), 2014 24th International Conference on. IEEE, 2014.

20. Zuo, Wei, et al. "New solutions for system-level and high-level synthesis." Integrated Circuits (ISIC), 2014 14th International Symposium on. IEEE, 2014.

21. Zuo, Wei, et al. "Improving high level synthesis optimization opportunity through polyhedral transformations." Proceedings of the ACM/SIGDA international symposium on Field programmable gate arrays. ACM, 2013.

22. Rose, Jonathan, et al. "Architecture of field-programmable gate arrays." *Proceedings of the IEEE*81.7 (1993): 1013-1029.

23. Marquardt, et al. "Using cluster-based logic blocks and timing-driven packing to improve FPGA speed and density." *Proceedings of the 1999 ACM/SIGDA seventh international symposium on Field programmable gate arrays*. ACM, 1999.

24. Sentovich, Ellen M., et al. *SIS: A system for sequential circuit analysis*. Vol. 41. Tech. Report No. UCB/ERL M92, 1992.

25. Cong, Jason, et al. "FlowMap: An optimal technology mapping algorithm for delay optimization in lookup-table based FPGA designs."*Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on* 13.1 (1994): 1-12.

26. Betz, Vaughn. "Architecture and CAD for Speed and Area Optimization of FPGAs." (1998).

27. Betz, Vaughn, et al. "VPR: A new packing, placement and routing tool for FPGA research." *Field-Programmable Logic and Applications*. Springer Berlin Heidelberg, 1997.

28. Betz, Vaughn, Jonathan Rose, et al. *Architecture and CAD for deep-submicron FPGAs*. Vol. 497. Springer Science & Business Media, 2012.

29. Saldanha, Alexander. "Functional timing optimization." *Proceedings of the 1999 IEEE/ACM international conference on Computer-aided design*. IEEE Press, 1999.

30. Bsoul, Assem AM, et al. "Implementation of an FPGA-based low-power video processing module for a head-mounted display system." *Consumer Electronics (ICCE), 2013 IEEE International Conference on*. IEEE, 2013.

31. "Smart Eyewear and Smart Goggles for Sports | Recon Instruments." *Recon Instruments*. Web. 30 May 2016.

32. Sengupta, Dipanjan, et al. "Low-power FPGA-based display processing module for head-mounted displays." *2011 IEEE International Conference on Consumer Electronics (ICCE)*. 2011.

33. Sutherland, Stuart, et al. "UVM Rapid Adoption: A Practical Subset of UVM."

34. Da Silva, Karina RG, et al. "An automatic testbench generation tool for a SystemC functional verification methodology."*Integrated Circuits and Systems Design, 2004. SBCCI 2004. 17th Symposium on*. IEEE, 2004.

35. Rashinkar, Prakash, et al. *System-on-a-chip verification: methodology and techniques*. Springer Science & Business Media, 2007.

36. Bergeron, Janick. *Writing testbenches: functional verification of HDL models*. Springer Science & Business Media, 2012.