FUNCTIONAL TEST GENERATION USING MICRO OPERATION FAULT MODEL

ONG HUI YIEN

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

Faculty of Electrical Engineering
Universiti Teknologi Malaysia

MAY 2011

To my beloved parents, wife and son

ACKNOWLEDGEMENT

This project would not have been successful without the endless support from many great people. Firstly, I would like to express my highest gratitude to my project supervisor, Dr. Ooi Chia Yee for her continuous guidance, patience and support for me throughout the project.

I would also like to take this opportunity to thank my fellow colleagues in Intel, Jonie Lim and Yoong Yaw for their guidance and knowledge sharing in SystemVerilog language and VCS tool. With their help, I was able to build the correct constraint model, generate quality sets of test pattern and this leads to better analysis in this project. Besides, I would also like to thank my peers, Alice Koh, Kevin Leong, Wee Soon and Si Long for their help and encouragement.

Lastly and most importantly, thanks to my wife and son who have been very supportive, patient and encouraging throughout this project.

ABSTRACT

As semiconductor technology advances further into nanometer regime, integrated circuit testing and validation continues to play a very important role to ensure high quality product. Conventionally, test patterns are generated from a gate level netlist using test generation tool. However, as the digital design increases in complexity, the gate level test generation process becomes more complicated and time consuming. As an extended alternative to this, functional fault model like micro operation fault model was introduced. However, in order to implement this, proper automation is necessary while minimizing intensive manual labor. Unfortunately, currently there is only proprietary version of automation available. In this project, an automated platform to generate test pattern using micro operation fault model was built using Perl programming language. The methodology involves conversion of behavioral model of design under test into extended finite state machine. This is followed by micro operation fault detection, fault activation and fault propagation with all the corresponding constraint sequences captured and converted into constraint model using SystemVerilog, a hardware description language. These models of fault free and intended faulty circuit were fed into a constraint solver tool, VCS by Synopsys to generate the test pattern. For verification purpose, these test patterns were validated by simulating the circuit using Altera Quartus II tool. The result of this project shows that reasonable fault coverage was achieved using this methodology.

ABSTRAK

Dalam era pembangunan teknologi nanometer separa pengalir terkini, pengujian dan pengesahan litar bersepadu masih memainkan peranan yang amat penting untuk memastikan kualiti produk atau komponen elektronik tetap tinggi. Secara lazimnya, corak ujian untuk tujuan pengesahan adalah dihasilkan daripada litar di peringkat get. Akan tetapi, apabila teknologi reka bentuk digital menjadi kian rumit dan kompleks, proses penjanaan corak ujian turut menjadi semakin rumit dan memakan masa yang agak lama. Sebagai alternatif untuk menyelesaikan masalah ini, permodelan kerosakan di peringkat fungsional seperti permodelan kesalahan operasi mikro telah diperkenalkan. Walau bagaimanapun, automasi adalah penting untuk pelaksanaan dan memudahkan proses penjanaan corak ujian sambil mengurangkan kerja manual. Dalam projek ini, pengautomatan untuk menghasilkan corak atau pola ujian menggunakan konsep permodelan kerosakan operasi mikro telah dilaksanakan dengan menggunakan bahasa pengaturcaraan Perl. Kaedahnya melibatkan penukaran model perilaku reka bentuk litar ke *extended finite state machine*. Ini diikuti oleh pengesanan dan pengaktifan kerosakan operasi aritmetik, dan perambatan kerosakan ini ke keluaran primer litar. Jujukan yang didapati daripada proses ini djadikan model kekangan dengan menggunakan bahasa penggambaran SystemVerilog. Seterusnya model-model akan diberikan kepada perisian penyelesai kekangan VCS untuk menjanakan corak ujian. Untuk tujuan pengesahan, corak-corak ujian ini disahkan oleh simulasi litar menggunakan perisian Altera Quartus II. Keputusan projek ini menunjukkan liputan kerosakan yang munasabah dapat dicapai dengan menggunakan metodologi atau kaedah yang dibincangkan.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

ATPG	-	Automatic Test Pattern Generation
CMOS	-	Complementary Metal Oxide Semiconductor
CS	-	Constraint Sequence
DFT	-	Design For Test
DUT	-	Design-Under-Test
EDA	-	Electronic Design Automation
EFSM	-	Extended Finite State Machine
HDL	-	Hardware Description Language
HDVL	-	Hardware Description and Verification Language
IC	-	Integrated Circuit
ITC'99	-	1999 International Test Conference
LSA	-	Line-Stuck-At
LSI	-	Large Scale Integrated
VCS	-	Verilog Compiler Simulator
VHDL	-	VHSIC Hardware Description Language
VHSIC	-	Very-High-Speed Integrated Circuit
VLSI	-	Very-Large-Scale Integration
OUT	-	Operation-Under-Test
RTL	-	Register Transfer Level
SST	-	Single-State-Transition

LIST OF APPENDICES

CHAPTER 1

INTRODUCTION

1.1 Project Background

Semiconductor technology has been improving rapidly lately and as it advances further into the nanometer regime, manufacturing processes become more defect-prone. Integrated circuit (IC) validation and testing play a very important role in ensuring and maintaining product quality while meeting the constraint of time-to-market. Ultimately the main goal is to obtain fault model that provides high fault coverage and requires short time for test generation.

Besides, the ever increasing complexity of digital designs are causing gate-level sequential test generation to become more time consuming and challenging. Many studies and researches have been done to achieve faster and better test sequence generation. Subsequently, there were some attempts to obtain higher fault coverage within reasonable time. One of them is the design-for-testability (DFT), such as scan methodology but this technique introduces additional hardware and causes area overhead.

Alternatively, we can perform test generation process at higher level of abstraction of the digital design, called functional test generation technique. In this technique, desired circuit functions are specified using hardware description languages (HDLs). The HDLs do all this without burdening the designer with the structural details of the circuit's implementation.

Since HDLs are used to describe hardware at a high-level, by definition, the language constructs must be related to the actual hardware. This relationship has a higher degree of abstraction than the relationship between the gate-level representation of a design and the hardware. The test generation algorithms designed for the high-level models are usually direct extensions of those for the gate-level models, in which the functional modules are treated as primitive components and this allows improvement because fewer components are evaluated during test generation. The potential performance advantage of the reduced structural complexity makes this approach more attractive.

Since the conventional single LSA (Line Stuck-At) fault model is no longer suitable at this abstraction level, functional fault model is introduced to support this high level test generation platform. Currently, more complete functional fault models which can represent failure at more syntax of HDL description are Fummi's fault model (Fummi *et al*., 1998) and Chen's fault model (Chen, 2003). Fummi introduced a fault model that consists of bit failure and condition failure. Bit failure covers LSA of each bit of variable, signal or port, while condition failure covers LSA in each condition which may remove some execution paths in the erroneous HDL description.

Micro operation fault was introduced in Chen's fault model, where it is a failure of micro operation to perform its intended function. The operators for the micro operation can be logical operators, relational operators, unary operators and arithmetic operators. An operator may fail to any other operator in its category. This fault is mapped by replacing the operator considered with its counter operator which must be defined. This project looks into Chen's fault model for micro operation fault, specifically on arithmetic operators like addition, subtraction, multiplication and division.

1.2 Problem Statement

A better functional fault model called *Enhanced Micro Operation Fault Model* (Ooi and Fujiwara, 2010) was introduced to overcome Fummi's and Chen's fault models in the early 2010. In order to implement these fault models effectively without too much manual labor, an automated functional test generation platform is required especially when these fault model targets hard-to-test circuit. Unfortunately, currently only proprietary version of automation is developed (Chen and Noh, 1998) and it is not available for public usage.

Full automation is necessary in this context because in order to generate test pattern using micro operation fault model, it involves steps like conversion of behavioral HDL into Extended Finite State Machine (EFSM). Also, with automation, one can easily trigger intended fault, and find its state justification path and propagation path to build the constraint sequence. Besides that, the conventional method to build constraint model is using manual effort and the designer has to always refer back to the behavioral model of the circuit. All these can be eliminated with the automated test pattern generation platform.

1.3 Objectives

The main objective of this project is to build an automated constraint models generation platform via series of software programming which basically reads in the behavioral model of a design under test (DUT), generates the corresponding constraint sequence and constraint model. The constraint models generated here shall be useful for high-level test pattern generation. This project does not only focus on the implementation part, but also to be tested on dedicated validation benchmark circuits from the 1999 International Test Conference (ITC'99). This is to analyze its functionality and effectiveness in terms of fault coverage and test generation time.

1.4 Scopes of Study

This project involves a series of research and work on functional fault modeling, high level automatic test pattern generation and constraint solving in order to develop a software program that is capable of generating test pattern based on micro operation fault model. The scope of work begins with the understanding of behavioral model of DUT in VHDL (very-high-speed-integrated-circuit hardware description language). This is because the benchmark circuits from ITC'99 are all released in VHDL code. It is important to differentiate the design entity of the DUT which basically consists of entity declaration and architecture block. In VHDL behavioral modeling, the code contains sequential statements that are executed sequentially in a predefined order. This order of the statements in the code is important and may affect the semantics of the code.

EFSM was studied to understand all of its entities. EFSM plays a very important role to model the DUT from VHDL and to derive test sequences. These test sequences are derived such that all the transitions of the EFSM are traversed to guarantee that every statement in the behavioral description is verified. Constraint models using SystemVerilog are generated from these test sequences.

By using constraint solver tool called VCS (Verilog Compiler Simulator, from Synopsys) to solve the constraint models, the corresponding test pattern can be generated. These test patterns are used to activate and propagate targeted faults. Another software tool called Altera Quartus II is used to simulate the DUT with the test pattern obtained earlier. The simulation waveforms can be used to validate and verify the quality of the test patterns.

The main software program is written using Perl programming language. Perl was chosen because it has very strong regular expression functions and its multiple levels of hashes of hashes can be easily implemented as database to store all the information of the EFSM. Besides that, Perl is a good scripting language and can be easily used to detect an arithmetic operation and trigger an intended fault to obtain the test sequence.

1.5 Organization of the Report

This report consists of six chapters with the brief description of each chapter as stated below:

Chapter 1 presents the introduction to this project, including the background of the project, problem statements, objectives of the project and scopes of the study.

Chapter 2 provides literature reviews from previous research work on high level ATPG and functional fault modeling.

Chapter 3 discusses the methodology applied in this project, including the project planning and schedule, design and implementation workflow, and software involved in this project.

Chapter 4 describes the software design and test bench design of the project. This chapter discuss about the architecture and structure of the main program and how SystemVerilog test bench is built as input for test pattern generation.

Chapter 5 discusses the result from the main software program, test bench building and VCS run. The verification of the run result is discussed as well, followed by analysis of the result.

Chapter 6 concludes the overall work done in this project. Limitations of the design in this project were discussed, including recommendation for future work.

REFERENCES

T. M. Sarfet, R.G. Markgraf, M.H. Schulz and E. Trischler (1992). A hierarchical test pattern generation system based on high-level primitives. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*. January 1992. Volume 11, 34-44.

I. Ghosh and M. Fujita (2001). Automatic test pattern generation for functional register-transfer level circuits using assignment decision diagrams. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems. March 2001. Volume 20, 402-415.

Jaan Raik and Raimund Ubar (2004). Enhancing hierarchical ATPG with a functional fault model for multiplexers. Proceeding of DDECS. 219-222.

K. T. Cheng and J.Y. Jou (1992). A functional fault model for sequential machines. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems. September 1992. 1065-1073.

K. T. Cheng and A. S. Krishnakumar (1993). Automatic Functional Test Generation Using the Extended Finite State Machine Model. 30th Conference on Design Automation. June 1993. 86-91.

F. Ferrandi, F. Fummi and D. Sciuto (1998). Implicit Test Generation for Behavioral VHDL Models. Proceedings International Test Conference 1998. 18-23 October. 587-596.

C-I. H. Chen (2003). Behavioral test generation/fault simulation. IEEE Potentials. Feb/Mac 2003. Volume 22, 27-32.

C. Y. Ooi and H. Fujiwara (2010). Constraint Driven Functional Test Generation Using Enhanced Micro Operation Fault Model. UTM, Malaysia and NAIST, Japan.

C. Y. Ooi and H. Fujiwara (2010). Enhanced Functional Fault Model for Micro Operation Faults. NAIST Information Science Technical Report, No.2010004. July 2010.

C-I. H. Chen and T. H. Noh (1998). VHDL behavioral ATPG and fault simulation of digital systems. IEEE Transactions on Aerospace and Electronic Systems. April 1998. Volume 34, 428-447.

Accellera Organization, Inc (2004). SystemVerilog 3.1a Language Reference Manual.