Ain Shams University
Faculty of Engineering

Automated Verification of Digital Systems

A Thesis
Submitted in partial fulfillment for the requirements of
Doctorate of Philosophy of Science degree in Electrical Engineering

Submitted by:

Mohamed Hanafy Sayed Radwan
M.Sc. of Electrical Engineering (2012)
Computer and Systems Engineering Dept.
Ain Shams University, Cairo, Egypt

Supervised by:

Prof. Dr. Ayman M. Wahba
Professor of Computer and Systems Engineering
Computer and Systems Engineering Dept.
Ain Shams University,
Cairo.

Dr. Hazem Said
Assistant Professor of Computer and Systems Engineering
Computer and Systems Engineering Dept.
Ain Shams University,
Cairo.

Cairo – 2015

Ain Shams University
Faculty of Engineering
Computers and Systems Department

Name : Mohamed Hanafy Sayed Radwan
Thesis : Automated Verification of Digital Systems
Degree : Doctor of Philosophy in Electrical Engineering

Examiner Committee

Name, Title, and Affiliate **Signature**

Prof. Magdy Bayoumi
Computers Engineering Professor
University of Louisiana – USA

Prof. Ashraf Salem
Computers and Systems Engineering Department
Faculty of Engineering
Ain Shams University

Prof. Ayman Mohamed Wahba (Supervisor)
Computers and Systems Engineering Department
Faculty of Engineering
Ain Shams University

13th December 2015

Abstract

This research introduces a new methodology for digital design properties extraction from simulation traces. A new Breadth-First Decision Tree (BF-DT) mining algorithm is innovated for complete properties extraction from simulation traces. The new mining engine supports both bit-level and word-level values of different design variables. The contributed mining technique could extract all design properties activated by the simulation traces. Such properties are relating the design output target variables with their corresponding cone of interest feature variables. The new methodology for automatic generation of assertions has been tested for different designs with different sizes. The generated assertions completely match with all design properties covered in the input simulation traces. Moreover, the generated assertions are at the highest possible level of abstraction leading to the best coverage for the input data space. The simulation results show that the proposed methodology has proven superior efficiency in extracting both bit-level and word-level complete assertions of digital design in superior quality, and feasible time and memory consumption.

Keywords: verification, assertion, coverage, mining, decision tree.

Acknowledgments

All thanks are to ALLAH for his non countable blessings. This dissertation would not have been possible to complete without the help of many people whom ALLAH has put them in my life, and has given them the ability and love to help me. I hope I can give them the acknowledgment they deserve. May Allah bless them.

My deep gratitude is to my advisors Dr. Hazem Said and Prof. Ayman Wahba. I have been amazingly fortunate to have advisors who gave me the freedom to explore on my own and at the same time the guidance to recover when my steps faltered. Their guidance and encouragement helped me shape and realize this work.

My deep gratitude is also to my manager in Mentor Graphics consulting division Ahmed Gharieb. He encouraged me to start my PhD, and supported me several times during its different stages.

I owe my sincer gratitude to my parents, my brother, my wife and my daughter for their continued support, their enthusiasm for my work, their patience, and their constant care and support not only during my research work but also during my whole life.

CONTENTS

List of Figures

List of Tables

CHAPTER 1

1 Introduction

This chapter includes an introduction about design flow of digital systems, digital design verification, and assertion based verification. It also shows the importance of assertions in the verification process and explains the mechanism of extracting them. Finally, the importance of automatic generation of assertions is discussed to introduce for the main contribution of this dissertation.

1.1 Design Flow of Digital Systems

As detailed in [1], figure 1.1 presents a conceptual design flow of digital Integrated Circuits (ICs) from specifications to final product. The flow in the figure shows a top-down approach that is very simplified. But, the reality of an industrial development is much more complex. Much iteration should be involved through various portions of this flow until the final product converges to a form that meets the specification requirements of functionality, area, timing, power and cost.

The design specifications are generally presented as a document describing a set of functionalities that the final solution will have to provide. They are also presented as a set of constraints that must be satisfied. In this context, the *functional design* is the initial process of deriving a potential and realizable solution from this design specifications and requirements. This is sometimes referred to as modeling and includes such activities as hardware/software tradeoffs and a micro-architecture design.

Because of the large scale of the problem, the development of a functional design is usually carried out using a hierarchical approach. Therefore single designer can concentrate on a portion of the model at any given time. Thus, the architectural description provides a partitioning of the design in distinct modules. Each of them contributes a specific functionality to the overall design. These modules have well defined input/output interfaces and protocols for communicating with the other components of the design. Among the results of this design phase is a high level functional description – often a software program in SystemC [2] or

similar programming language – that simulates the behavior of the design with the accuracy of one clock cycle and reflects the module partition. It is used for performance analysis and also as a reference model to verify the behavior of the more detailed designs developed in the following stages.

From the functional design model, the hardware design team proceeds

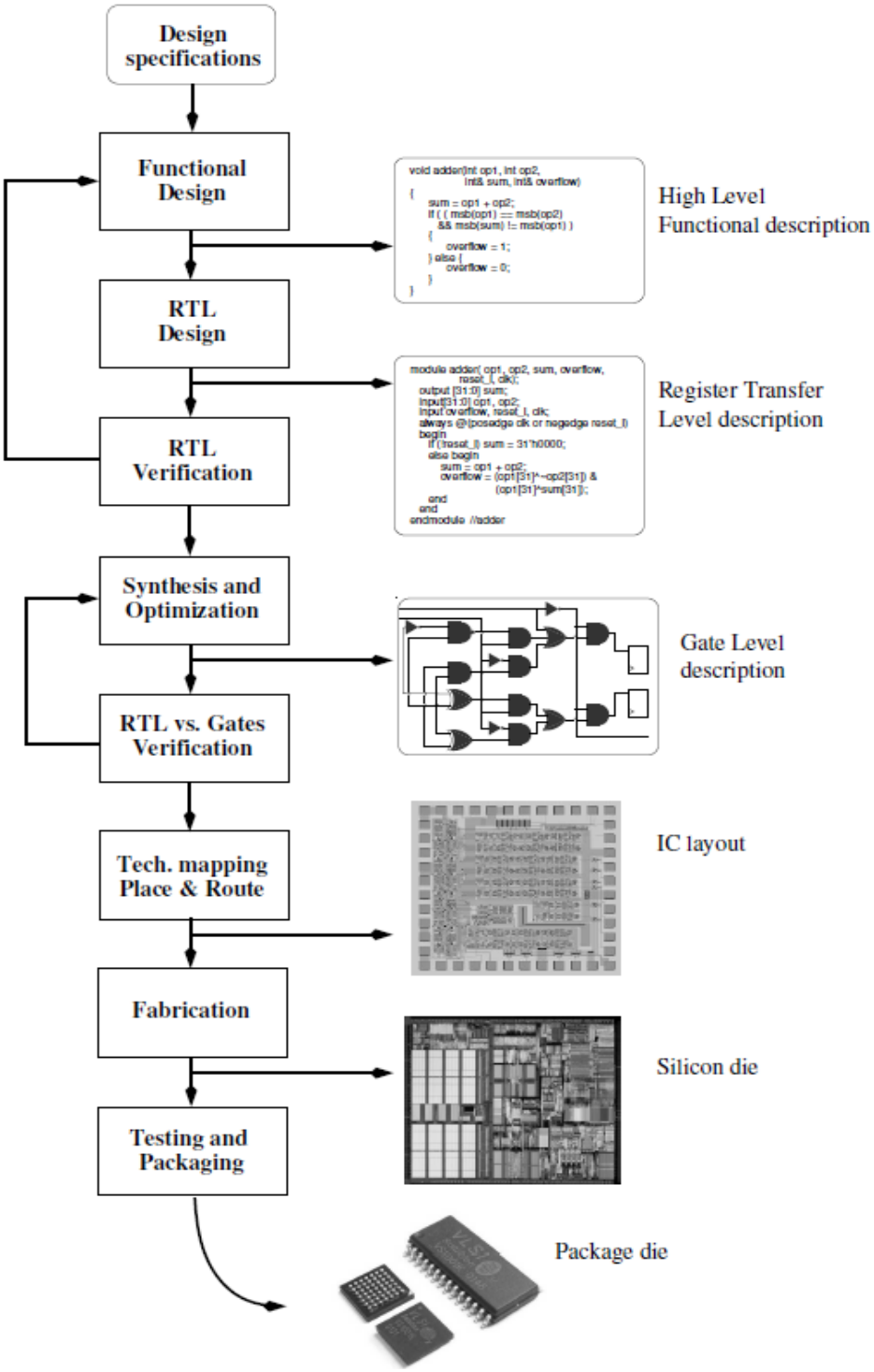

Figure 1.1: : Design Flow of a Digital System

to the *Register Transfer Level (RTL) design* phase. During this phase, the architectural description is further refined: memory element and functional components of each model are designed using a Hardware Description Languages (HDL). This phase also checks the development of the clocking system of the design and architectural trade-offs such as speed/power.

With the RTL design, the functional design of the digital system ends and its functional verification begins. *RTL verification* consists of acquiring a reasonable confidence that a circuit will function correctly, under the assumption that no manufacturing fault is present. The underlying motivation is to remove all possible design errors before proceeding to the expensive chip manufacturing.

Each time functional errors are found, the model needs to be modified to reflect the proper behavior. During RTL verification, the verification team develops various techniques and numerous suites of tests to check that the design behavior corresponds to the initial specifications. When that is not the case, the functional design model needs to be modified to provide the correct behavior specified and the RTL design updated consequently. It is also possible that the RTL verification phase reveals incongruences or overlooked aspects in the original set of specifications and this latter one needs to be updated instead.

In the diagram of Figure 1.1, RTL verification appears as one isolated phase of the design flow. However, in practical designs, the verification of the RTL model is carried on in parallel with the other design activities and it often lasts until chip layout. An overview of the verification methodologies that are common in today's industrial developments is presented in the next section.

The next design phase consists of the *Synthesis and optimization* of the RTL design. The overall result of this phase is to generate a detailed model of a circuit which is optimized based on the design constraints. For instance a design could be optimized for power consumption or the size of its final realization (IC area) or for the ease of testability of the final product. The detailed model produced at this point describes the design in terms of its basic logic components, such as *NAND, NOR,* or *XOR* and memory elements.

Optimizing the netlist, or gate level description, for constraints such as timing and power requirements is an increasingly challenging activity for current developments. It usually involves multiple iterations of trial-and-error attempts before it converges to a solution that satisfies both these requirements. Such optimizations may in turn introduce functional errors that require additional RTL verification.

Most of the activities starting from synthesis and optimization are semi-automatic or at least heavily supported by Computer Aided Design (CAD) tools. Automating the RTL verification phase is the next challenge that the CAD industry is facing in providing full support for digital systems development.

The synthesized model needs to be verified. The objective of *RTL versus gates verification*, or equivalency checking, is to guarantee that no errors have been introduced during the synthesis phase. It is an automatic activity requiring minimal human interaction. It compares the pre-synthesis RTL description to the post-synthesis gate level description in order to guarantee the functional equivalence of the two models.

At this point, it is possible to proceed to *technology mapping* and *placement and routing*. The result is a description of the circuit in terms of a geometrical layout used for the fabrication process. Finally the design is *fabricated*, and the microchips are *tested and packaged*.

This design flow is obviously a very ideal, conceptual case. For instance, usually there are many iterations of synthesis, due to changes in the specification or to the discovery of flaws during RTL verification. Each of the new synthesized version of the design needs to be put through again all the subsequent design phases.

One of the main challenges faced by design teams, for instance, is in satisfying the ever increasing market pressure to produce designs with faster and faster clock cycles. These tight timing specifications force engineering teams to push the limits of their designs. The limits can be achieved by optimizing them at every level: architectural, in the components choice and sizing, and in placement and routing. Achieving timing closure, that is developing a design that satisfies the timing constraints set in the specifications. While still operating correctly and

consistently, most often requires optimizations that go beyond the abilities of automatic synthesis tools. Besides, it could force the engineers to intervene manually, at least in some critical portions of the design.

Often, it is only possible to check if a design has met the specification requirements after the final layout has been produced. If these requirements are not met, the engineering team comes up with alternative optimizations or architectural changes and creates a new model that needs to go through the complete design flow.

1.2 Functional Verification of Digital Design

As shown in the previous section, the correctness of a digital circuit is a major consideration in the design of digital systems. The increasing costs of manufacturing microchips are extremely high. Therefore, the consequences of flaws going unnoticed in system designs until after the production phase would be very expensive. At the same time, RTL verification is still one of the most challenging activities in digital system development.

Till few years ago, verification is carried on mostly with ad-hoc tests, scripts and often even tools developed by the design and verification teams specifically for the current design. In the best case, these verification infrastructure developments can be amortized among a family of designs with similar architecture and functionality. Moreover, verification methodology still lacks a unique standalone general standard or even a world-wide commonly accepted approach.

Each hardware engineering team could have its own distinct verification practices. These practices often change with subsequent designs by the same team, due to the insufficient correctness confidence level that any of the current approaches provide. The latest methodology Universal Verification Methodology UVM [3] hasn’t proven yet to be the unique world-wide general verification standard. However, it could be in the coming years especially with the new enhancements and added features in its latest releases. Given this scenario, it is easy to see why many digital IC development teams report that more than 70% of the design time and engineering resources are spent in verification [4]. That is

why verification is thus the bottleneck in the time-to-market for integrated circuit development.

1.2.1 Evolution of Verification Methodologies

A fast review for the evolution of verification methodologies could be stated as follows. SystemVerilog [5] was started to merge a number of disjoint verification languages such as Vera [6] and *e* [7] that were built as a layer on top of Verilog and VHDL. Each of these languages had their own proprietary methodologies (RVM [8] and eRM[9]) that provided a re-useable framework to construct, configure, and execute tests. Once SystemVerilog became established, it needed its own methodology.

Mentor Graphics created the AVM (Advanced Verification Methodology) [10] in 2006 that was derived from concepts in SystemC. Synopsys converted their Vera-based RVM (Reuse Verification Methodology) library to SystemVerilog and called it VMM (Verification Methodology Manual) [11], but did not make it publicly available. Mentor Graphics and Cadence joined together and created the OVM (Open Verification Methodology) [12] in 2008. OVM was the merging of the existing AVM with concepts from *e*RM. Finally by 2011, Mentor Graphics, Cadence, and Synopsys joined together through Accellera and created the UVM (Universal Verification Methodology).

One of the most efficient verification techniques is the Assertion-Based Verification (ABV). It is recommended by [4] to enhance both the verification quality and debugging time. Some projects using ABV experienced a decrease in verification time by as much as 50% [13]. ABV is predominately adopted into different industry's verification methodologies due to its efficiency in verifying complex SoC designs [14]. The importance of assertions in the digital design verification process and ABV are more detailed in the coming section.

1.2.2 Functional Validation Procedures

The workhorse of the industrial approach to verification is *functional validation*. The functional model of a design is simulated with meaningful