C H A P T E R 1

INTRODUCTION

1.1 Motivation

In the last decade, the digital design process has undergone a rapid change. Traditionally, design engineers created digital systems pictorially and built hardware prototypes to check their logic. With the advent of hardware description languages (HDLs) and synthesis tools [13, 45], software prototypes of digital systems are built and checked instead. The designer translates the system specification to an HDL program which is then input to a synthesis tool that automatically produces a gate level circuit. The

design of Application Specific Integrated Circuits (ASICs) has dramatically benefited from this development. Because a high percentage of ASIC designs are Digital Signal Processing (DSP) circuits [49], this new design process has had its greatest impact in the DSP area.

The VHISC Hardware Description Language (VHDL) [14] is a popular hardware description language in today's electronic industry. It was designed to be not only a hardware description language but also a system design language. VHDL has the capability to model digital circuits at various levels of abstraction, which is extremely useful in the DSP area. With VHDL, DSP designers frequently start with real number models, which are then converted to fixed point integer models and finally to bit-vector models that are synthesizable. Also, behavioral synthesis tools are emerging which process the VHDL code programmed at a high level of abstraction and produce the gate level circuits directly.

Accompanying these changes is the increasing need to test VHDL models before they are synthesized. From a testing viewpoint, testing VHDL models is similar to testing software programs. The focus is on the errors in the VHDL code itself rather than in the fabrication process to come. This suggests that the techniques successfully applied in software testing may also be effective in testing hardware designs in VHDL [46].

Completely testing a design requires that all input combinations be exercised, which is impractical if not impossible as the size of the model grows. A limited number of test cases should be carefully chosen. To test a model, one develops a test bench which is a testing framework into which the model under test (MUT) is inserted. The test bench is configured according to a test scenario, and test vectors are generated and sent to the MUT for simulation. The MUT response is compared with the expected response, and the result is used to determine if a failure occurs. This process is repeated as many times as there are test cases. A test plan helps with documenting all the test scenarios and serves as a guide to testing the MUT. However, all these tasks are labor intensive and time consuming. Besides, manually performing all these tasks repetitively is expensive and unreliable. Therefore, it would be a significant contribution to the hardware design community to develop efficient approaches to VHDL model testing and create a software system that automates the testing process.

1.2 Contributions

The principal objective of this dissertation is to develop an efficient high level test planning system to validate VHDL models, in particular, in the DSP application area. A functional testing approach is adopted; that is, the implementation of the model under test is treated as a black box. The system requirements parameterized from the specifications constitute the input space and serve as generics of test benches.

This dissertation presents a methodology for library-based structural test bench development. Development of hierarchical VHDL libraries is discussed. A configuration declaration is created for every test case to specify the library models to be bound to the test bench structure and provides generic values. Use of control files for simulation control is addressed. The simulation efficiency of test benches is emphasized. Approaches to dramatically reducing time and memory required in simulation of a radar return are proposed.

This dissertation also proposes a test planning framework as a vehicle of planning and documenting the testing activities [47, 48]. It uses a goal tree structure to represent test plans where test goals are given based on the specifications and test groups are defined to satisfy the test goals. They constrain the system requirements and thus partition the input space into smaller and thus more manageable subspaces. A set of test strategies are presented which can be applied to the test groups for test case design. Some of them sample the interior and boundary points from the input spaces to form test cases. Others search for the extreme points from the output spaces by exploring the boundary conditions that are believed to produce very effective tests [50, 51]. According to the test cases specified in the test plan, test benches generate test vectors against which the MUT is tested. The MUT results are compared by a comparator and verdicts are reached by test oracles.

The proposed methodology has been implemented in software with more than 32,000 lines of C code on a Solaris platform. The software integrates the elements of the test planning system including the RASSP test bench generator. It also links to the specification repository so that the test plan can immediately reflect any changes in the system requirements. The integrated test planning system includes a goal tree editor that is a friendly graphical tool implemented in an X Window system for easy development of goal trees. A goal tree system manipulates and traverses goal trees. Based on the goal trees, it commands the test bench generator and controls the testing processes. The test planning system automates all the testing activities such as test planning, stimulus generation, comparator configuration, test vector generation, MUT simulation, and simulation result comparison. MUT configuration and model selection are provided by a test bench generator interface. A test planning graphical user interface has been developed to provide the user with a friendly environment to operate the test planning system. It provides a test plan driven mode under which all test cases given in the test plan are performed back-to-back. It also provides a user driven mode which allows the user to set up and simulate one single test at a time. Both test modes support reuse of test vectors through file I/O [13, 14]. The test planning system also serves as a query system in which the user selects test goals and sequences of tests are executed on the MUT to answer the queries. Both test modes support the file I/O feature to allow reuse of pregenerated test files for the purpose of, for example, post testing analysis or diagnosis.

This dissertation illustrates the proposed methodology using the Synthetic Aperture Radar system (SAR) [9, 11, 20~24, 28] as a case study. The example model under test [17] is a VHDL algorithmic model that describes the behavior of the range compression processing algorithm of the SAR system. The proposed approaches to simulation efficiency improvement are applied to the SAR test bench. These turn the original inefficient SAR test bench generator, which used to crash a Sun Sparc 10 workstation after 32 hours of full-scaled simulation, into an efficient one which takes only 2 minutes of simulation on the same machine, without having to downsize the problem. All the testing activities are performed and automated using the integrated test planning system, which saves the tester a significant amount of time and effort. The simulation results are discussed in detail. Completeness and effectiveness of the generated test set are evaluated with the aid of the test planning system using file I/O. An analysis shows that the test set achieves 100% statement coverage, 100% branch coverage, and 94.0% mutation coverage. All of the qualities discussed above combine to show that the test planning approaches and the automated test planning system are very efficient, effective, and easy to use. Although this dissertation focuses on VHDL models in DSP applications, the methodology presented could be applied to other areas.

Another contribution of this dissertation is the proposed approach to hierarchical faulty module isolation for hierarchical circuits. Exposability is proposed to measure the extent that signal values are revealed to the tester and is used as a cost function for the

faulty module search problem. An expanded goal tree which explores the functional and structural aspects of a hierarchical circuit is presented.

1.3 Contents

This rest of this dissertation is organized as follows.

Chapter 2, “Background,” provides an overview on testing approaches and test benches, and the RASSP test bench generator. It also gives an overview of the Synthetic Aperture Radar system which is used as a case study throughout this dissertation.

Chapter 3, “Test Bench Development,” describes the methodology for developing a library-based structural test bench. It also outlines the enhancements that the author has made to the implementation of the stimulus generator to improve simulation efficiency.

Chapter 4, “Test Planning Framework and Goal Tree System,” addresses the concept of the test planning framework and presents the development of the goal tree system.

Chapter 5, “Test Planning System Integration,” discusses the integration of modules of the test planning system and the test planning graphical user interface.

Chapter 6, “Results and Quality Evaluation,” contains the results of the tests that were performed on the SAR model using the test planning system. It also reports the experimental investigation into the quality of the test set generated by the SAR goal tree.

Chapter 7, “Hierarchical Diagnosis,” discusses approaches to faulty module isolation and proposes the exposability measures as cost functions. It also presents an expanded goal tree that explores the structural aspect of a hierarchical circuit.

Chapter 8, “Conclusions and Future Work,” summarizes this dissertation. It draws conclusions based on the results of this work, and suggests prospective new research directions.