LABORATORY TRAINING FOR TEACHING DESIGN AND TEST OF DIGITAL CIRCUITS

A. JUTMAN, R. UBAR
TALLINN TECHNICAL UNIVERSITY , ESTONIA

KEYWORDS: Education Methodology, Testing Software and Tools, Web-Based Teaching

ABSTRACT: A conception of practical works for teaching design and test of didital sircuits is given. The works cover essential topics in testing and diagnostics field. They are ment for improving the skills of students to be educated for hardware and SoC design in test related topics. Four designed practical works are described. All the training manuals can be accessed over Internet and therefore can be used by students at any time and any place. The works itself are based on the diagnostic software Turbo Tester that was developed at Tallinn Technical University. A brief description of the Turbo Tester is given along with several examples of possible problems it can illustrate and solve.

INTRODUCTION

The rapid advances in the areas of deep-submicron electron technology and design automation tools are enabling engineers to design larger, more complex, integrated circuits. Until recently, most electronic systems consisted of one or multiple printed circuit boards, containing multiple integrated circuits (IC) each. Recent advances in IC design methods and technologies allow to integrate these complex systems onto one single IC. These developments are driving engineers toward new System on a Chip (SOC) design methodologies. SOC is seen as a major new technology and the future direction for the semiconductor industry. Within the next four years, SOC designers will cut new product development cycle time from an average of 12 months today, to just four months by 2004. The key to this forecast becoming a reality is in placing the power in the hands of the SOC designer.

On the other hand, the more complex are getting electronics systems the more important are getting the problems of test and design for testability, as the costs of verification and testing are becoming the major components of the design and manufacturing costs of a new product. Today, design and test are no longer separate issues. The emphasis on the quality of the shipped products, coupled with the growing complexity of systems design, require testing issues to be considered early in the design process.

At present, most VLSI and system designers know little about testing, so that companies frequently hire test experts to advise their designers on test problems, and they even pay a higher salary to the test experts than to their VLSI designers [1]. This reflects the today's university education: everyone learns about design, but only truly dedicated students learn test. Entering into the SOC era means that test must now become an integral part of the VLSI and system design courses. The next generation of engineers involved with VLSI technology should be made aware of the importance of test, and trained in test technology to enable them to produce high quality, defect-free products.

Design for testability (DFT) is rapidly becoming one of the key considerations in today's SOC designs. Moving towards multi-million gate SOCs makes embedded testing strategies via Buil-In Self-Test (BIST) architectures mandatory. It is critical to ensure that students will be equipped with the skills in DFT and BIST, and also get hands on experience in using CAD for test tools that make them successful designers when they leave university [2].

The National Science Foundation in USA held a workshop in 1998 where it was stated that the present level of "test coverage" in the computer engineering education in USA was inadequate. As a consequence to this statement, a special panel was organized at the International Test Conference in 1999 how to enhance the coverage of test related topics in computer engineering education [3].

In the following a conception is presented how to improve the skills of students to be educated for hardware and SOC design in test related topics.

We present a description of laboratory course where the student can obtain hands on experience on design for test and designing embedded self-test architectures.

THE CONCEPT

The laboratory training is meant to help students to obtain practical skills in the field of testing and diagnostics as an essential addition to the theoretical knowledge they can get from lectures. All the training manuals can be accessed over Internet [7] and therefore can be used by students at any time and any place.

Each manual was composed in HTML language and contains links to related topics in the theory [6], which is a complementary part of the lab training. This helps

students to go to the exact place in the theory instantly and refresh/acquire the theoretical knowledge needed for a particular practical work. The manuals also contain figures and tables to visualize the content of the works. Comprehensive examples and detailed descriptions are also very helpful for students and allow them to work fully independently of the teacher.
A well-structured layout of the training manuals is clear and informative. It consists of following main parts:

- *Objectives* – this part explains which skills and practical knowledge a student can acquire during this work. It also describes main topics of the work.
- *Introduction* – gives the basic information about the subject of the work, depicts industrial arias where it is used, and gives other background information.
- *Work description* – introduces subjects of the work, e.g. circuits to be tested and tools to be used for that, and gives general idea about what should be done in the work.
- *Steps* – this part contains clearly defined enumerated list of tasks students have to carry out during the work.
- *Example* – comprehensive and detailed description of what should be done at each step of the work. The tools which are used during the work described extensively in this part. We give also examples of how it is recommended to represent the experimental data.

THE WORKS

Here we give the basic and brief information about the works. For more details visit the dedicated Internet page [7]. At the moment the whole laboratory training consists of four following works:

- Test Generation
- Design Error Diagnosis
- Built-in Self-Test
- Design for Testability

Test generation work [8] introduces the basics of testing to students. The work provides with skills of composing simple tests manually as well as gives an experience in using of several basic automatic test pattern generators (ATPG). Hence, students can compare different test generation approaches and realize their suitability for different classes of devices or circuits.
Three sorts of circuits are used in the work: a small random circuit, a full 8-bit adder, and comparatively complex ISCAS '85 benchmarks [9]. Such choice has been made to have a possibly full spectrum of combinational circuits that require different approaches in testing. For example, a test for a small circuit can be composed manually about as easily as using automatic tools. A manually composed test for an adder can be of a much smaller length than a test acquired using an ATPG and therefore it can be much more effective. This is valid due to the human ability to recognize the functionality of some circuits and choose the best strategy of test composition in each case. On the contrary, in the case of complex circuits it is reasonable to use only automatic tools (i.e. not to do it manually).

The goal of the *design error diagnosis* work [10] is to introduce the basics of diagnosis using the design error diagnosis problem as an example. During the work students also learn how to compose diagnostic tests and see the important difference between diagnostic and verification tests.
A set of simple circuits has been designed for this work. These circuits have several good properties in the sense of teaching diagnosis. At first, they are 100% testable for stuck-at faults. This eliminates some troubles related to incompletely testable circuits and helps concentrating purely on the problem of diagnosis, not on the circuit's testability (this topic has the dedicated work [12]). Secondly, they have multiple outputs, which is very important feature because there are some diagnostic rules, which can be used only with multiple output circuits.

The idea of the diagnosis is, when having the specification and the implementation, which are not functionally equal, compose diagnostic tests in order to find the exact location of the error. The designed circuits play role of the implementation. The specification is generated on-line by each student and for every circuit. It will be random and different from the specifications other students acquire. So, this ensures the individual work of each involved student. We also implemented the possibility for teacher to check if a student finds the correct fault type and location.

Built-in self-test (BIST) work [11] is dedicated to embedded into a device testing facilities such as BILBO (Built-In Logic Block Observer), CSTP (Circular Self-Test Path), and a Hybrid BIST approach where most test vectors are generated online and some are stored in memory. Using them, students are able to explore and to compare different built-in self-test techniques. Playing with certain parameters of the BIST devices, they learn finding best BILBO and CSTP architectures and initial states (seeds) and best combinations of stored and generated patterns in Hybrid-BIST approach.

Design for testability work [12] is meant to show students the importance of having a good-testable circuit as the result of the design process. Some solutions for improvement of the testability of bad-testable circuits are shown. Students can try all the variants for a certain circuit and see what suits better for the situation.

The idea that two circuits connected in series can produce a complex device, which is hardly controllable and observable lies in the base of the work. A set of modified devices with different degrees of testability is provided to show how simple means can significantly lighten the task of a tester. The modified devices are just the same two circuits connected in series but with some additional pins inserted helping to observe or control some hardly testable (or even not testable at all) lines between the circuits.

At the current moment, the laboratory training on design and test is covering the basic topics in testing and diagnostics field. It is based mostly on the methods working at the lower level (gate level). The future development of

the laboratory training can be directed to the use of high-level verification and simulation approaches.

PC-BASED TOOLS FOR TRAINING TEST

Traditional VLSI test generation and fault simulation software on workstations are both costly and unable to handle large numbers of students simultaneously in educational courses. During the recent years, many different low-cost tools running on PCs have been developed to fill this gap. They include usually the major basic tools needed for IC design: schematic capture, layout editors, simulators and place and route tools. Low-cost systems for solving a large class of tasks from the dependability area - test synthesis and analysis, fault diagnosis, testability analysis, built-in self-test, especially for teaching purposes, are missing. For this reasons, at the Tallinn Technical University a diagnostic software Turbo Tester [4] was developed.

The Turbo Tester consists of several different tools for test pattern generation, simulation, test set optimization, BIST emulation, design error diagnosis, and experimental statistics representation. ATPGs and fault simulators are available for both, sequential and combinational circuits. Combinational circuits can be tested by deterministic, random and genetic algorithm based methods, while for sequential designs a random ATPG is available. In addition to fault simulators, the simulation tools include multi-valued simulation for hazard analysis in combinational circuits. For BIST emulation, BILBO and CSTP approaches can be chosen. For more details refer to the TT manual available at [5].

After theoretical investigation of the test topics given to students at the lectures, a laboratory work follows with more complex designs, where the arbitrary available design software (schematic editor as minimum), and the Turbo Tester diagnostic software is used. The students develop digital circuits as diagnostic objectives, investigate by Turbo Tester tools the testability of circuits, redesign them if necessary for testability, insert self-test structures, analyze the efficiencies and trade-offs of different test solutions and learn to make proper engineering decisions in the field of testable design.

SOME EXAMPLES OF WORK WITH THE TURBO TESTER

Having the wide spectrum of different tools it is good to start practicing with test generation. One of the tasks formulated in the practical works is application of several test generation approaches to an 8-bit full ripple-carry adder and then the comparison of acquired results. In Table 1 we give the results of test pattern generation by all the three ATPGs and by hand.

The table shows the most important parameters of a test pattern generation method such as the number of generated test patterns and test generation time for two modes of test generation. The first mode is called *default mode* when all the ATPG parameters are set with default values. Usually it implies the minimization of test pattern generation time. And we can see that among all the ATPGs the fastest is the deterministic one but in terms of the number of patterns the best is the genetic ATPG. However, neither of them could achieve as best results as we got manually. So, now we are trying to change some of the parameters of the ATPGs (not the deterministic one because it cannot be tuned to get shorter tests) in hope to get better results. These results are given in the "tuned" columns of the table. You can see that the test sequences are already very short but still not as best as that, which we composed by hand. It is a good example, which illustrates that sometimes if we know the functionality of the unit under test, we can compose more efficient tests than those generated by ATPGs.

Another example is taken from the field of design error diagnosis. The Turbo Tester can generate both the specification and the implementation inserting a real single design error in an arbitrary location so that the circuit and the spec are not functionally equal any more. Your task now is to find the erroneous area in the circuit and rectify it so that the new version of the design will be functionally equal to the specification.

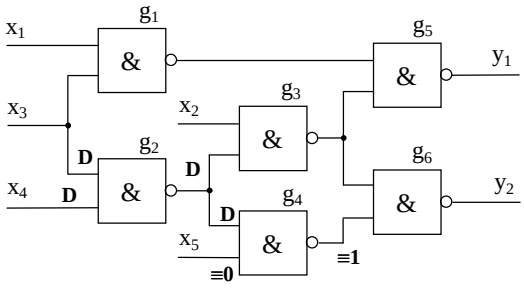

Fig. 1 *Combinational circuit*

It is possible to apply the *prediagnostic* tool first. It is meant for using with bigger designs and it derives a certain area in the circuit called *suspected* area. The remaining gates are assumed to be correct. Then the actual task of a student is to locate the exact erroneous gate in this area. At first, the diagnosis should be formulated in terms of stuck-at faults and then it can be mapped into the domain of design errors [13].

TABLE 1 Comparison of several test pattern generation methods

Test Generation Method	Default		Tuned	
	# of Patterns	Time, s	# of Patterns	Time, s
Deterministic ATPG	20	0.01	NA	NA
Genetic ATPG	13	4.33	9	70.18
Random ATPG	24	0.26	9	6.74
Manually	8	NA	NA	NA

Suppose we have the implementation shown in Fig. 1 and suspected gates are g_2 and g_4. Suspected stuck-at zeros are shown by "0", stuck-at ones are shown by "1", and if both fault types are suspected it is denoted as "D".

Since we have two suspected gates we have to prove that one of them is correct and we have to rectify another one. One of the possible approaches to the solution is the following. We see that the gate g_2 affects both outputs y_1 and y_2 but gate g_4 affects only y_2. Therefore, g_4 can be eliminated as soon as we detect an error at y_1. In the table below, a test is given, which activates both stuck-at faults at both inputs of g_2 and propagates them to y_1 output.

	x_1	x_2	x_3	x_4	x_5
Pattern 1	0	1	1	0	0
Pattern 2	0	1	0	1	0
Pattern 3	0	1	1	1	0

To apply a manually composed test we use vector manager tool and then we find the erroneous output responses using the verification tool. The obtained information is given below.

	y_1	y_2
Pattern 1	Error	Error
Pattern 2	Error	Error
Pattern 3	Error	Error

So, we can see that the error is reflected at the both outputs, that is, our hypothesis was right and we conclude that g_2 is the erroneous gate in the implementation.

Another useful feature of the Turbo Tester is its ability to illustrate statistics in a graphical way. The following example uses test sequences generated by the three ATPGs and the BILBO emulator to visualize the speed of growth of progressive (cumulative) coverage of the mentioined sequences for c432 ISCAS'85 benchmark.

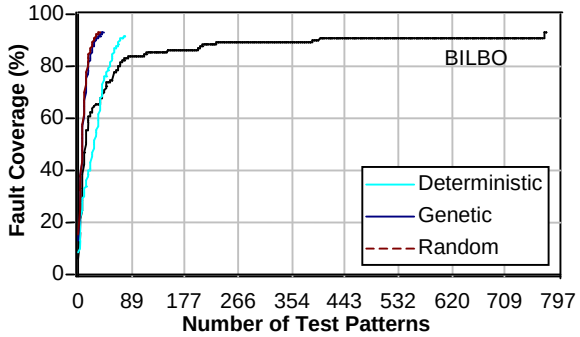

***Fig.2** Progressive Coverage of Test Patterns*

CONCLUSIONS

A conception is presented for improving the skills of students to be educated for hardware and SOC design in test related topics. It is a combination of learning the topic by using internet based simple "living pictures" on one hand, and hands-on training by using a set of commercial design tools and low-cost university tools dedicated for simulating and estimating different test and testability solutions on the other hand. The tasks chosen for hands-on training represent simultaneously real research problems. This allows to foster in students critical thinking, problem solving skills and creativity in a real research environment and atmosphere.

The principal mission of the conception is to inspire students to learn, to inspire them on a journey to knowledge, and to prepare them to develop problem-solving strategies.

THE AUTHORS

Artur Jutman and Prof. Raimund Ubar are with the Computer Engineering Department of Tallinn Technical University, Raja 15, 12618 Tallinn, ESTONIA.

E-mail: artur@pld.ttu.ee

Acknowledgements – This work has been suported partially by the Estonian Science Foundation (under Grant G4300), by German Government (DILDIS project) and the European Community (Copernicus JEP 977133 VILAB).

REFERENCES

[1] M.L. Bushnell. Increasing Test Coverage in a VLSI Design Course. International Test Conference, Atlantic City, NJ, USA, 1999, p. 1133.

[2] J. Harrington. VLSI Design 101 – the Test Module. International Test Conference, Atlantic City, NJ, USA, 1999, p. 1134.

[3] V.D. Agrawal. Increasing Test Coverage in a VLSI Design Course. International Test Conference, Atlantic City, NJ, USA, 1999, p. 1131.

[4] G.Jervan, A.Markus, P.Paomets, J.Raik, R.Ubar, "A CAD System for Teaching Digital Test," *Proc. of the 2nd European Workshop on Microelectronics Education*, Kluwer Academic Publishers, pp. 287-290, Noordwijkerhout, the Netherlands, May 14-15, 1998

[5] Turbo Tester home page URL: http://www.pld.ttu.ee/tt

[6] Theory URL:
http://www.pld.ttu.ee/dildis/diagnostika/theory/theory.html

[7] Laboratory training URL:
http://www.pld.ttu.ee/dildis/diagnostika/labs/

[8] Test Generation lab work URL:
http://www.pld.ttu.ee/dildis/labs/tglab.htm

[9] F. Brglez and H. Fujiwara, "A neutral netlist of 10 combinatorial benchmark circuits and a target translator in FORTRAN," *Int. Symposium on Circuits and Systems, Special Session on ATPG and Fault Simulation*, 1985

[10] Work on diagnosis URL:
http://www.pld.ttu.ee/dildis/labs/derdlab.htm

[11] Built-in Self Test work URL:
http://www.pld.ttu.ee/dildis/labs/bistlab.htm

[12] Design for Testability URL:
http://www.pld.ttu.ee/dildis/labs/dftlab.htm

[13] A. Jutman, R. Ubar, "Design Error Diagnosis in Digital Circuits with Stuck-at Fault Model," *Journal of Microelectronics Reliability. Pergamon Press, Vol. 40, No 2*, 2000, pp.307-320.