45th International Scientific Colloquium
Ilmenau Technical University
October 04-06, 2000

R.Ubar/H.-D.Wuttke/E.Orasson

Interactive Teaching Software "Introduction To Digital Test"

1. INTRODUCTION

This work is a result of a co-operation work carried out in 1999 between Technical University of Ilmenau and Technical University of Tallinn (Estonia) under the project “DILDIS” supported by the Ministry of Education in Thüringen.

Our society is becoming increasingly dependent on computing systems. This dependency is especially felt upon the occurrence of failures in systems. That is the reason why design for dependability, verification and test are becoming more and more important in all the life periods of a system, and therefore, these issues should be considered also in teaching tomorrows system engineers. Young engineers should be trained on integrating design and test solutions. Teaching in this domain should be facilitated by using integrated CAD tools that support design verification, testability analysis, design for testability, test generation, fault simulation, built-in self-test, fault diagnosis and fault tolerance.

Traditional teaching methods either start by explaining a theory and showing some examples or giving an example to introduce a theory. However one of the both methods is chosen and fixed by the teacher independently of what is best for most of the students. The students mostly get some accompanying materials such as scripts, books etc. After listening to a lecture they can consult only their notes and try to solve some problems by using the new learned method as good as they remember. Mostly there is not enough time for lots of examples during a lecture and the students' notes include some errors. To correct these errors and to give some more examples the students usually replicate the subject of the lesson at home and during an exercise. In this paper the questions are discussed like how can computers, connected to the internet, be used to make this learning process more effective, and what kind of software is required?

The paper is organized as follows. In Section 2 we discuss why it is important to teach digital test for system engineers? In Section 3 we present a learning method based on using so-called *living pictures* [1]. Section 4 shows an example how to use this approach of living pictures in teaching test issues, and in Section 5, we give a description of the laboratory course where students can obtain hands on experience on design for test and designing embedded self-test architectures.

2. WHY TO TEACH DIGITAL TEST FOR SYSTEM ENGINEERS?

Our society is becoming increasingly dependent on computing systems. This dependency is especially felt upon the occurrence of failures in systems. On the other hand, the cost of testing and verification because of the increasing complexity of systems has become a significant part of the total cost of electronic products. That is the reason why design for dependability, verification and test are becoming more and more important in all the life periods of a system - in development, production and exploitation - and therefore, these issues should be considered also in teaching tomorrows system engineers.

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 bout 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 [2]. This reflects the today's university education: everyone learns about design, but only truly dedicated students learn test. Entering into the System-on-Chip (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 Built -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 [3].

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 [4].

From the above it follows that the curricula in system and electronics engineering at technical universities should incorporate a greater emphasis on design for testability (DFT) and on the concepts of digital testing. Young engineers should be trained on integrating design and test solutions. Teaching in this domain should be facilitated by using integrated CAD tools that support design verification, testability analysis, design for testability, test generation, fault simulation, built-in self-test, fault diagnosis and fault tolerance.

3. LIVING PICTURES

For teaching basic knowledge in the test domain professional CAD tools are to complex. On the other site students get better learning results, if they can do tool supported experiments. That's why we use an action based teaching method. Action based training via internet names a new teaching concept where the learning process is characterized by several features like computer aided learning and teaching, offering a set of tools to inspect the subject, access to multiple learning modules, a big reservoir of examples and the possibility to generate new ones, focus on correct solutions, easy action and reaction (click and watch) by using "living pictures", the possibility of distance learning, multilingual descriptions, individual depth and duration, learning by doing, funny and game-like context. This concept offers teachers and students the possibility of free acting in the learning process. Core of that concept are some JAVA-applets (the interactive modules) running on any browser connected to the Internet. We call these applets "living pictures" and explain it as follows:

Given a tricky, quite complicated situation of the learning subject in a graphical form on the computer. The graphic has to be self-explanatory and involving interaction possibilities. By using interaction possibilities the students can generate examples that are interesting enough to encourage own experiments but not too complicated for

learning. They can produce input stimuli and watch the reactions. In reaction of the inputs a simulation component starts, executing the method that has to be taught, and presenting its results using a visualization component. Thus the students immediately get a correct reaction of their inputs. In that phase it is important that there is a simple interface and no assessments of the students' inputs occur. They can use it like a game: they act and the system reacts by using a yet unknown method (that one, that should be learned). In addition to the simulation component there is also an explanation component, describing the unknown method step by step, using the actual chosen or generated example.

Comparing this method with traditional learning methods using books and notes, taken during a lecture, the learning process becomes more effective. During the same time the number of examples that can be inspected is higher and the students loose no time in going wrong ways because of wrong notes they made. They can always be sure that they get the right results. The game-like character of the living pictures raises the students' curiosity and encourages them to do some own experiments and examples. The same software described above can support several phases of the learning process as written in [1].

4. TEACHING SOFTWARE BASED ON JAVA APPLETS

The teaching software developed for test curricula supports the action based training via internet. The software offers a set of tools to inspect the objective to be learned, access to multiple learning modules, a big reservoir of examples and the possibility to generate new ones. It provides easy action and reaction (click and watch) by using "*living pictures*", the possibility of distance learning, and learning by doing. It is written in Java 2, and it can be run over network, using standard browsers like "Netscape" and "Internet Explorer" with Java 1.2 runtime plug-in, or with Java 2 applet viewer.

The software can be used for teaching the basics of Digital Test and Testable Design as illustrative tool explaining the problems of fault modeling and simulation in digital systems as well as test generation and fault diagnosis problems.

The work window of this program (see Fig.1) consists of three main parts - vector insertion panel, view panel for design schematics, and view panel for data tables and waveforms. The vector insertion panel has two sub panels - one for manually inserting single test vectors and another one for automated generation of a set of random test vectors. The first sub panel is used for creating diagnostic test vectors for fault location. The second sub panel has controls for LFSR (for learning basics of different self-test strategies) – for inserting initial state vectors, register feedback configuration polynomials, and length of test to be generated. The register can work in different self-test modes, like BILBO or CSTP [5]. The view panel for design schematics displays currently selected schematics and small boxes for wire states. These boxes are clickable during manual test generation and fault diagnosis. The student may insert different possible faults, and watch how they change the circuit's behavior at different input patterns, how the test patterns can be generated to detect the given fault, or how the faults can be localized by test patterns. Signal values, signal conflicts, detected faults etc. are displayed as colored bold wires.

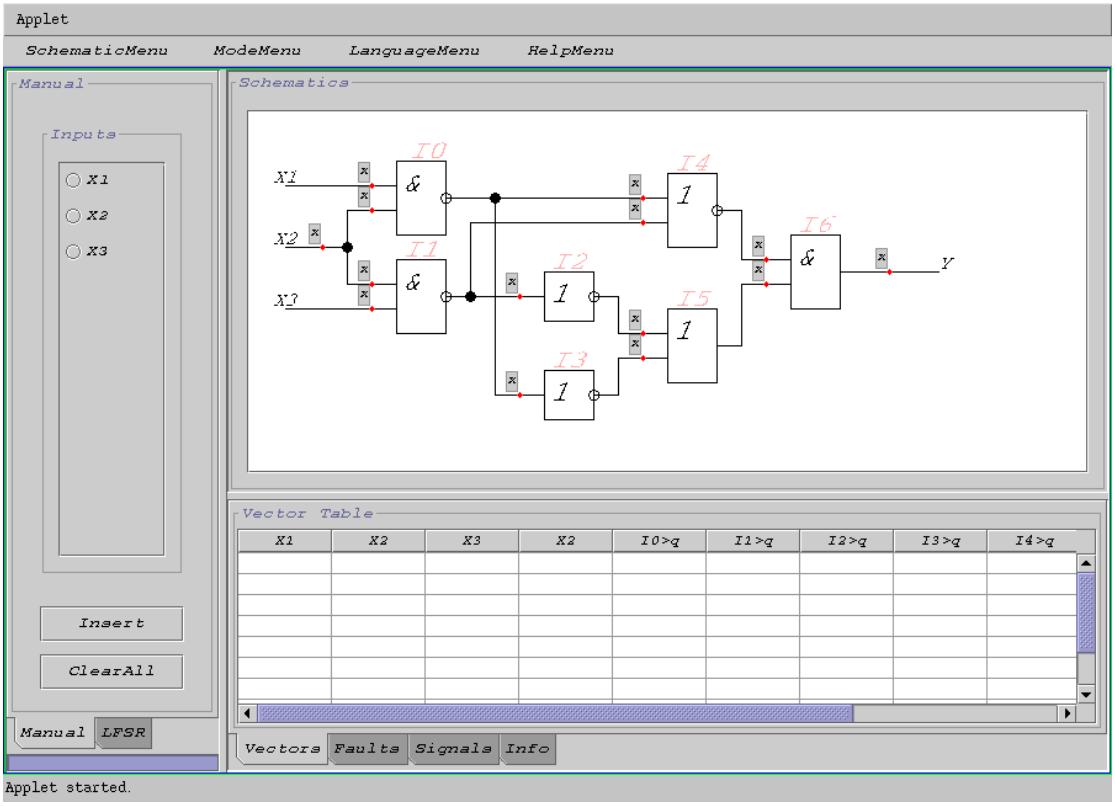


Fig.1. Work window for teaching test issues

The following topics are supported by the software: fault simulation, test generation, testability analysis and design of self-test. The key problems can be taught and learned using the software on different examples.

5. LABORATORY TRAINING BY DIAGNOSTIC PC-BASED TOOLS

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 [6] was developed.

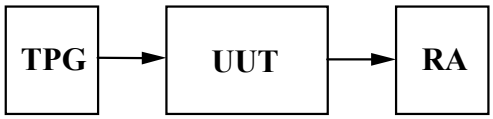
After theoretical investigation of the test topics described in the previous section, a laboratory work follows with more complex designs, where the arbitrary available design software (schematic editor as minimum), and the Turbo-Tester diagnostic software is used. The students develop digital circuits 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-off of different test solutions and learn to make proper engineering decisions in the field of testable design.

The following laboratory works are developed to train the engineering skills for the field of test: test generation, design for testability, built-in self-test, and design error diagnosis.

Test generation. The goal of this work is to get acquainted with the problem and CAD tools of creating test patterns for digital circuits At first, tests for the given circuit are generated manually. The quality of the manual tests is evaluated by the fault simulation tool. Then three different test generating tools (based on deterministic, random and genetic algorithms) are used and compared with each other.

Design for testability. The goal of this work is to show how the management of controllability and observability of test points in the circuit can improve the quality of testing. When designing a circuit every designer must think not only about the performance, cost and other essential requirements the circuit must conform, but he must also pay attention to the testability subject. At first, a testability analysis is carried out for the given circuit by using test generation and fault simulation tools. Then based on the testability information achieved, the circuit should be redesigned with the goal to get finally a test with good quality i.e. with good fault coverage. Tradeoff problems between the redesign cost and test quality are investigated.

Built-in self-test (BIST) is the capability of a circuit to test itself. Students concentrate themselves in a off-line BIST [6] consisting of a test pattern generator (TPG), unit under test (UUT) and a response analyzer (RA) (Fig.2).



Fig, 2. BIST architecture

TPG is usually a pseudo-random pattern generator, and RA – a signature analyzer, both based on linear feedback shift registers (LFSR) [6]. There are several disadvantages of such a structure: the test sequences generated are usually very long, and, they do not guarantee always a sufficient fault coverage because of existence of “hard-to-test” faults. To overcome these drawbacks, combining on-line generated pseudo-random test patterns with pre-generated and stored test patterns may be used. In this approach, the following problems should be solved by a test engineer:

1) To find the shortest LFSR and the best characteristic polynomial for the LSFR to be used for on-line test generation to achieve the highest fault coverage at the minimum length of pseudo-random test sequence.
2) To find the shortest LFSR and the best characteristic polynomial for the LSFR to be used for response (signature) analysis to guarantee the minimum loss of accuracy in fault detection.
3) To find the best level of mixing pseudo-random test and stored test as the tradeoff between memory cost and testing time.

To find solutions for these problems will be the task of this laboratory research for students. The students are not asked to carry out boring measurements, to press simply on buttons for starting a program and getting results which are nothing but a simple confirmation of what they already know from lectures. Instead, they are asked to solve a series of engineering problems, they have at their disposal a set of tools, and they themselves have to plan and carry out experiments to find answers for the given questions.

Design error diagnosis. The goal of this work is to learn how to compose diagnostic tests to localize the design errors in a given circuit. Suppose, a combinational circuit has been designed, and it has to meet the functionality described in the specification. Suppose, the verification step has shown that the implementation and the specification are not functionally equal. The task of the student is to find the erroneous area in the

circuit and rectify it so that the new version of the design will be completely functionally equal to the specification. At first, the student generates by a test generation tool test patterns which have the capability to detect and not to localize the faults. By using the automatic design error diagnosis tool the student determines a set of suspected faulty gates. Usually, additional test patterns are needed to extract from this suspected set of gates the actually faulty gate. These tests should be generated manually. The main idea is to eliminate by manual tests step by step the gates which are not faulty from the initial set of suspected gates. Using iteratively CAD tools, theoretical reasoning and manual work for generating additional tests, students get a good experience in solving extremely demanding engineering problem of error diagnosis in a given design.

6. CONCLUSIONS

A conception is presented for improving the skills of students to be educated for digital 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, which 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.

Acknowledgement: This work has been supported partially by the Estonian Science Foundation (under Grant G3658), by German Government (DILDIS project) and the European Community (Copernicus JEP 977133 VILAB).

References:

[1] H.-D. Wuttke, K. Henke, R. Peukert. Internet Based Education - An Experimental Environment for Various Educational Purposes. Proc. of the IASTED Int. Conf. on Computers and Advanced Technology in Education, May 6-8, Philadelphia, PA USA, *conference proceedings* pp. 50-54, 1999.

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

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

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

[5] M.Abramovici et al. Digital Systems Testing and Testable Design. IEEE Press, 1999, 652 p.

[6] G.Jervan, A.Markus, P.Paomets, J.Raik, R.Ubar. Turbo Tester: A CAD System for Teaching Digital Test. In "Microelectronics Education". *Kluwer Academic Publishers*, pp.287-290, 1998.

Authors:

Raimund Ubar, Elmet Orasson

Tallinn Technical University
Raja 15, 12617, Tallinn, ESTONIA,
Tel: 327 620 2252
Fax: 327 620 2253
E-Mail: raiub@pld.ttu.ee

Heinz-Dietrich Wuttke
Ilmenau Technical University
D-98684 Ilmenau, GERMANY,
E-Mail: Dieter.Wuttke@theoinf.tu-ilmenau.de