Combining Learning, Training and Research in Laboratory Course for Design and Test

R. Ubar, E.Orasson, J.Raik

Tallinn Technical University
Raja 15, 12617 Tallinn, Estonia
{raiub, elmet, jaan}@pld.ttu.ee

H.-D.Wuttke

Ilmenau Technical University
Ilmenau, Germany
Dieter.Wuttke@theoinf.tu-ilmenau.de

ABSTRACT: A conception is described how to combine learning, training and research phases in a laboratory course for educating today's VLSI and system designers. A method is explained how to use in the introductory teaching and learning phase interactive internet based modules - "living pictures". Further on, the hands-on training phase follows where commercial design tools and low-cost in-lab-made test oriented tools are used for solving real engineering tasks with close relationship to not yet solved research problems.

1 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 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 [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.

At first, we present a learning method based on using so-called *living pictures* [4]. On one hand teachers can

show more complex examples and immediate demonstrate the influence of changing parameters by using computer simulated living pictures in their lessons. On the other hand students can use the same simulations on their home computer, if the living pictures are available on the Internet. Second, 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.

2 Learning with “living pictures”

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?

3 “Living pictures” for learning test

The teaching software developed 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. The core of that concept are some Java-applets (the interactive modules) running on any browser connected to the internet By using interaction possibilities the students can generate examples that are interesting enough to encourage own experiments. 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.* There is also an *explanation component*, describing the unknown method step by step, using the actual chosen or generated example. The same software described above can support several phases of the learning process as written in [5]. 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.

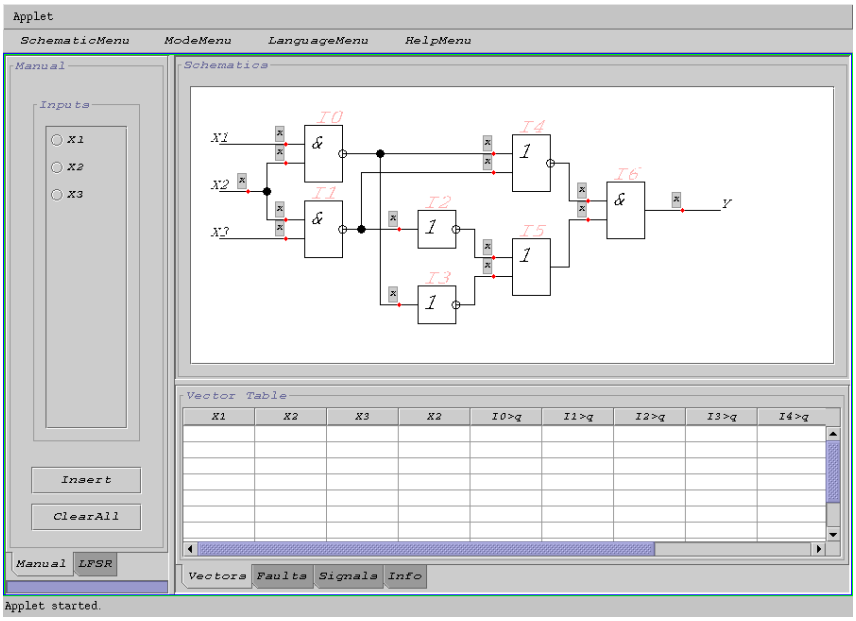


Fig.1. Living picture for fault diagnosis

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. Work window of this program (see Figure) consists of three main parts - vector insertion panel, view panel for design schematics, and view panel for data tables and waveforms. Vector insertion panel has two subpanels - one for manually inserting single test vectors and another one for automated generation of a set of random test vectors. First subpanel is also used for creating diagnostic test vectors for fault location. Second subpanel 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 [6]. 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 or how they can be detected by test patterns. Detected faults, signal conflicts etc are displayed as colored bold wires.

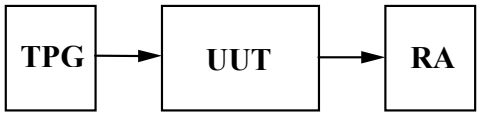
4 PC-based tools for training test

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 [7] 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-offs of different test solutions and learn to make proper engineering decisions in the field of testable design.

5 Training and research in design for BIST

In the following the main principles of one laboratory research are described based on using commercial design tools (like Cadence or other) and the test generation, fault simulation and BIST simulation tools of the Turbo Tester tool set [7].

Built-in self-test is the capability of a circuit to test itself, In a large variety of a BIST methodologies we concentrate ourselves here in a off-line BIST [6] consisting of a test pattern generator (TPG), unit under test (UUT) and a response analyser (RA) (Fig.2).



Fig, 2. BIST architecture

TPG is usually a pseudorandom pattern generator, and RA – a signature analyser, both based on linear feedback shift registers (LFSR) [6]. There are several disadvantages of such a structure. First, the test sequences generated randomly are usually very long, second, they do not quarantee always a sufficient fault coverage because of existence of so called "hard-to-test" faults. To overcome these drawbacks, combinations of several approaches have been proposed. One of them is based on combining on-line generated pseudorandom test patterns with stored pregenerated test patterns. In this approach, at first pseudorandom test sequence with a length L is generated on-line, after that a swith to a stored test approach takes place. For the stored test approach, previously generated and then in the memory stored test patterns are read one by one from the memory and applied to the UUT to reach the 100% fault coverage. For pregeneration of stored test patterns arbitrary software test generators may be used based on deterministic, random or genetic algorithms.

There are now several problems to be solved which still have not found sufficient solutions in the research and industrial community:

1) What is the shortest LFSR and what is 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 the pseudorandom test sequence?
2) What is the shortest LFSR and what is the best characteristic polynomial for the LSFR to be used for response (signature) analysis the quarantee the minimum loss of accuracy in fault detection?

3) How to find the best level of mixing the pseudorandom test and stored test as the tradeoff between the memory cost and testing time.

To find solutions for the mentioned question will be the task of the 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.

The are not available straightforward algorithms or software tools to find directly solutions for the mentioned problems. The only method is to set up hypotheses and check them by experiments.

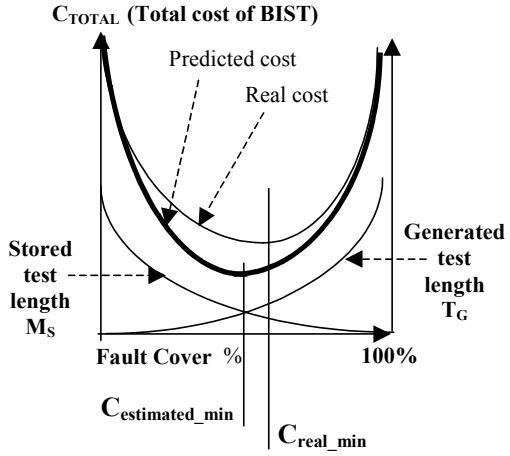


Fig.3. Finding the best mixed solution for BIST

Fig.3 shows a graphical solution for finding the optimum of mixing pseudorandom and stored test approaches as the tradeoff between the memory cost and testing time. Let have the whole cost of the BIST to be found as

$$C_{TOTAL} = C_{TIME} + C_{HW} = \alpha T_G + \beta M_S$$

where C_{TIME} is the cost related to the time needed for test, C_{HW} is the hardware cost related to the BIST architecture, T_G is the length of the test generated by LFSR, M_S is is the number of patterns to be stored, and α,β are unknown constants to map the test length and memory space to the costs of two parts of test solutions to be mixed.

The problem is that it would be very time consuming to find experimentally all the curves shown in Fig.3 except the generated test length T_G. The practical way would be in trying to find the curve for M_S with as least as possible number of experiments, and to try to predict the curve on the basis of experimental data, and then by choosing additional as few as possible experiments to approach step by step to the real optimum.

6 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, 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.

Acknowledgements: This work has been suported partially by the Estonian Science Foundation (under Grant G3658), 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. ITC, 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] R. Ubar, H.-D. Wuttke. Action-Based Learning System for Teaching Digital Electronics and Test. 3rd European Workshop on Microelectronics Education. Aix en Provence, France, May 18-19, 2000.

[5] H.-D. Wuttke, K. Henke, R. Peukert. Internet Based Education - An Experimental Environment for Educational Purposes. Proc. of IASTED, May 6-8, Philadelphia, PA USA, pp. 50-54, 1999.

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

[7] 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.