THE DILDIS-PROJECT- USING APPLETS FOR MORE DEMONSTRATIVE LECTURES IN DIGITAL SYSTEMS DESIGN AND TEST

Raimund Ubar[1], Heinz- Dietrich Wuttke [2]

Abstract *— This paper presents a new teaching concept supporting the learning process by several features. It supports the possibility of distance learning as well as a web-based computer-aided teaching, offers a set of tools ("interactive modules") to inspect the teaching topics and carries out laboratory research. A big reservoir of examples and the possibility to generate own ones makes the learning process more interesting and allows learning at an individual depth and duration. The interactive modules are focused on correct solutions, easy action and reaction, multilingual descriptions, learning by doing, a game-like use, and fostering students in critical thinking, problem solving skills and creativity.*

Index Terms — Digital Test and Testable Design, JAVA-applets, Learning technologies, Web-Based Training.

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. System on a Chip (SOC) is seen as a major new technology and the future direction for the semiconductor industry. On the other hand, the more complex are getting electronics systems the more important will be the problems of test and design for testability because of the very high costs of verification and testing of designs and new products. Design and test are no longer separate issues.

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. It is critical to ensure that students will be equipped with the skills in Design for Testability (DFT) and Built-in Self-Test (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* [4,5]. The method presented deals with the goal, to put interactive teaching modules to the Internet that can be used in a lecture as well as for individual self-studies [4]. They can be accessed independent of time and place. On one hand, teachers can show more complex examples and immediately 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 a laboratory course where the students can obtain hands on experience on design for test and on designing self-testing architectures.

THE CONCEPT OF LIVING PICTURES

Teaching the basics of Digital Test and Testable Design means to teach a lot of complex connections that have to be explained at first one by one but then in their dynamic interactions. Traditional teaching methods using desk, overhead or "PowerPoint"- presentations can explain those connections only partially. After the lecture the dynamic changes get lost and the students have only a static part of the whole scenario in their notices. An enhancement of this situation can be reached, if the whole materials, containing all scenes of the dynamic process can be accessed via the Internet. The only problem in this situation is that students have no possibility for an active interaction with the materials. So the motivation to work with the "lifeless" material is not very high.

The core of the teaching concept presented are some JAVA-applets (the interactive modules) running on any browser connected to the Internet. We call this type of applet "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-

[1] Raimund Ubar, Tallinn Technical University, Computer Engineering and Diagnostics, Ehitajate tee 5, Tallinn, EE0026, Estonia, raiub@pld.ttu.ee
[2] Heinz-Dietrich Wuttke, Ilmenau Technical University, Faculty of Informatics and Automation, post box 10 0565, D-98684 Ilmenau, wuttke@theoinf.tu-ilmenau.de

0-7803-6669-7/01/$10.00 © 2001 IEEE **October 10 - 13, 2001 Reno, NV**
31st ASEE/IEEE Frontiers in Education Conference

explanatory and involving interaction possibilities. By using interaction possibilities the students can generate examples that are interesting enough to encourage their 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.

Because the JAVA-applets are running on any Internet-browser, it is also possible to work with the well-known browser functions for copy and paste to transfer data from one applet to another. The results or questions during the work with the applets can also be copied into any other program and sent by e-mail to teachers or other students. In that way the applets support also a step by step workflow in a design process. In the paper we want to demonstrate, how to use the applets during lectures and in self-studies in the field of Design and Test of digital circuits. We want to show the increased learning efficiency and the motivation for own experiments by the students.

LIVING PICTURES FOR TRAINING TEST

The program for representing "living pictures" for teaching Digital Test is written in Java 1.3. It can be run over network, using standard browsers like Netscape and Internet Explorer with Java 1.3 runtime plug-in, or with Java 1.3 applet viewer. The program can be used for teaching the basics of testing digital systems, test generation, fault simulation and fault diagnosis.

The work window of the applet consists of three main parts:

- Vector insertion panel
- View panel for design schematics
- View panel for displaying information (data tables and waveforms)

The vector insertion panel shown in Figure 1 has two sub panels:

- Subpanel "I" for inserting single input test vectors (Ii) and
- Subpanel "S" for setting up the feedback configuration (Si) of a Linear Feedback Shift Register (LFSR) to be used for automatically generating test vectors.

In the LFSR mode, the first subpanel is used for initializing the LFSR.

The LFSR based Automated Test Pattern Generator (ATPG) is used for emulating different BIST ideas. The following well known architectures can be emulated by the applet:

- Built-In-Logic-Block-Observer (BILBO) or
- Circular-Self-Test-Path (CSTP) [6,7].

The first subpanel is also used when creating test vectors for specific fault detection. In this case the values are inserted one by one into the signal boxes at connections of the design schematics, and the input test vector will be deduced from these internal signal values.

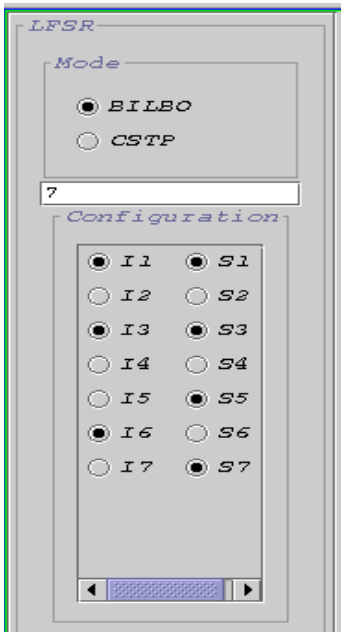


FIGURE 1
VECTOR INSERTION SUBPANEL

The schematics panel shown in Figure 2 displays currently selected schematics. The small boxes at the lines display internal signal values on the connections. The boxes are clickable during manual test vector generation and fault diagnosis. In the test generation mode, the needed signal values for fault activation or fault propagation can be inserted directly at the connections. In the fault diagnosis mode, by clicking the boxes, a guided probing procedure can be simulated. A click on the box shows the result of measuring the "real" signal on the corresponding connection of the simulated faulty circuit.

Detected faults, signal conflicts etc. are displayed as colored bold wires. Color coding is as following:

- red - stuck-at-1 fault is detectable
- green - stuck-at-0 fault is detectable
- gray - undefined (don't care) signal
- blue - conflicting signals.

The data panel shown in Figure 3 displays information about simulated test vectors and detected faults.

In the fault simulation mode you can click on the row of a given test vector and have a visualization which faults are detected by this vector. In the signal (waveform) mode you can select all the signals in interest and leave out those which are not.

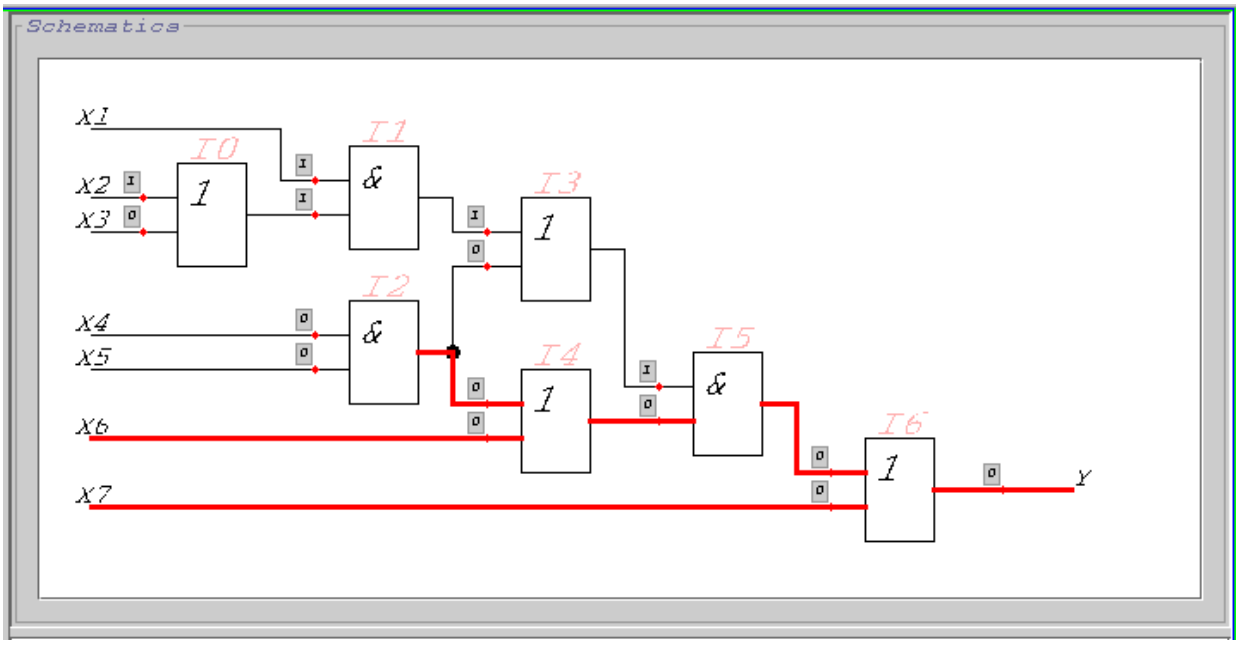


FIGURE 2
SCHEMATICS SUBPANEL

The statistics sub panel (info), shown as part of figure 4, gives some details about the currently selected design.

There are four main menus used with the applet:

- schematics,
- mode,
- language, and
- help.

The schematics menu contains a list of predefined circuits. By the language menu the user may choose one of the currently supported languages from the given list. The help menu provides with useful tips and explanations.

The mode menu tells the applet what is to be done - test vector insertion, manual test vector generation, fault simulation or fault diagnosis (two possible modes: sequential and combinational diagnosis).

We start working with the applet by selecting in the schematics menu a circuit from a set of predefined circuits. Then we can carry out different experiments with this circuit by selecting a proper working mode from the mode menu.

In the vector insertion mode we can choose test vectors either automatically by using LFSR, or by inserting vectors manually. In the manual mode, we generate step by step input patterns which are simultaneously simulated. The boxes at the lines on the schematics subpanel display the result of simulation – the values of internal signals on the connections. The waveforms can be viewed on the data subpanel .

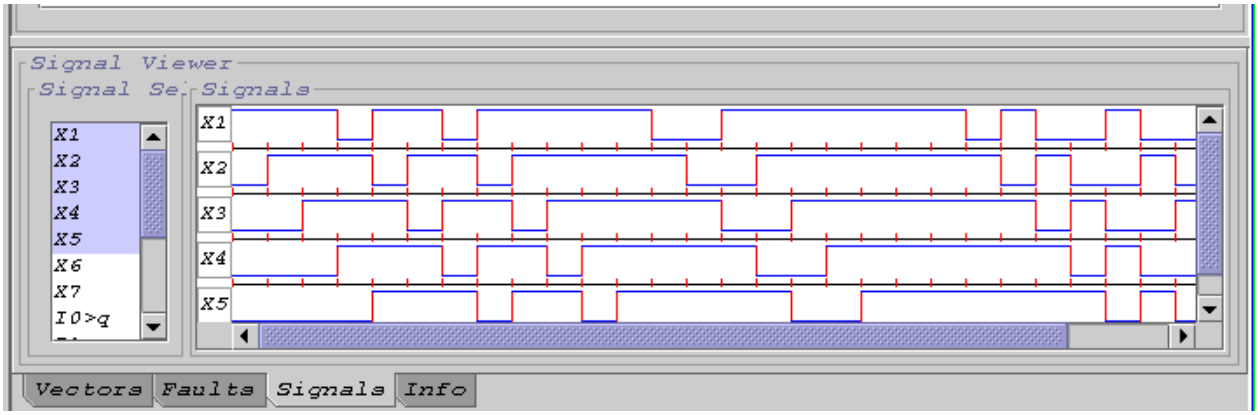


FIGURE 3
WAVE TABLE SUBPANEL

When using LFSR, we have to specify the initial state (subpanel I), to set up the feedback structure (subpanel S), and to specify the length of the test sequence. By LFSR we can simulate the BIST concept in two ways: Built-In Logic Block Observer (BILBO) and Circular Self-Test Path (CSTP) [6,7]. In the BILBO mode a pseudo-random test sequence is generated autonomously by LFSR based on the settings on the subpanels I and S. In the CSTP mode, the pseudo-random test sequence is generated by both, LFSR and the circuit selected by menu Schematics. By changing the setting on the subpanel S we can emulate different feedback structures of the chosen BIST architecture.

In the test generation mode we choose a target fault in the schematic and create step by step proper activated paths in the ciruit for activating the fault at his site and for propagating the error signals caused by the activated fault towards output by clicking the needed values into boxes on the lines. From these values finally, an input vector will be deduced. The colours on lines help us to understand the current status of the task: activated faults and activated paths are marked by red and green lines, the inconsistencies of the signal values are highlighted by blue colour. As the result of the procedure, a test pattern will be generated. The detected by the test vector faults can be displayed also on the data panel in form of a row in the fault table.

In the fault simulation mode, a fault table is generated and shown on the data panel for all the test vectors created by the given moment. By selecting a vector on the data panel, all the faults detected by this vector will be highlighted by colours on the schematic panel.

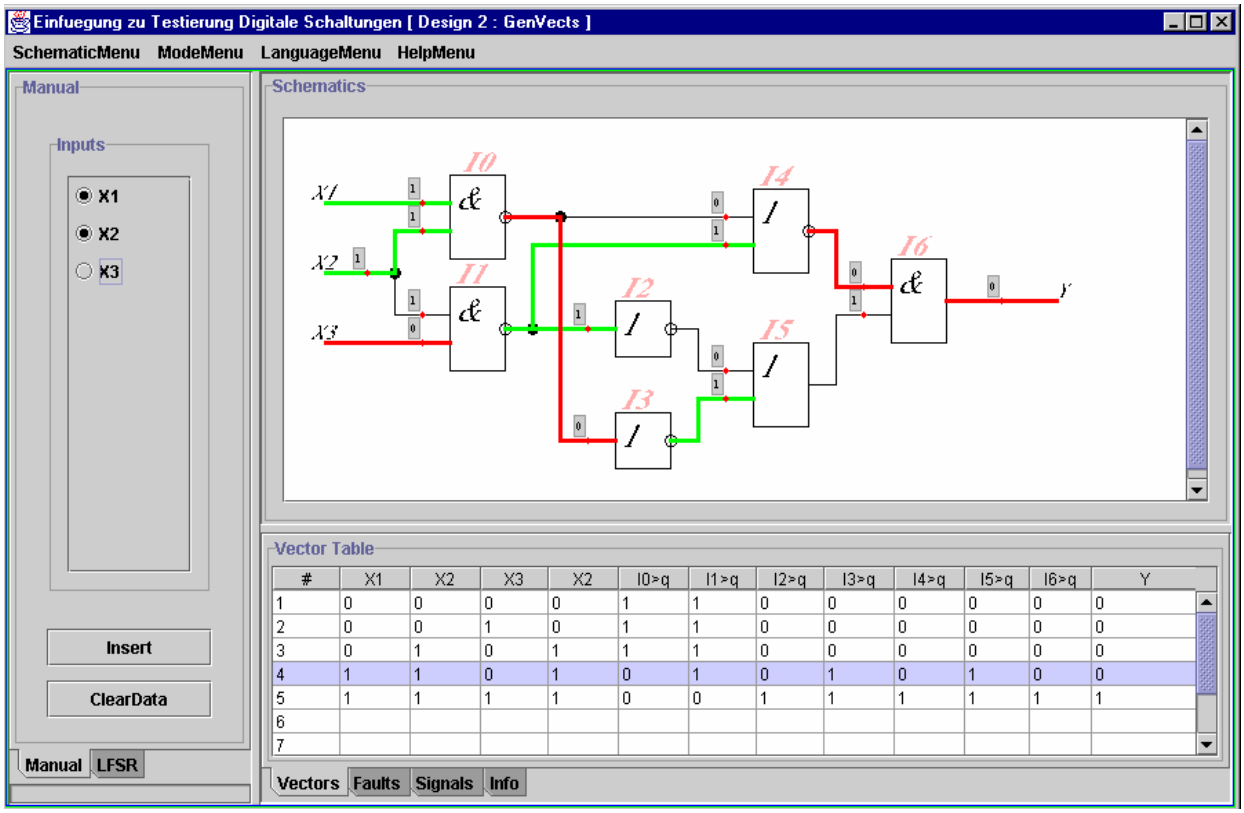
In the fault diagnosis mode we need at first, to create a fault table by running the fault simulator for a set of previously generated test vectors. Entering into the diagnosis mode will insert a random fault into the circuit.

The following diagnosis strategies can be investigated: combinational and sequential diagnosis.

For learning the first combinational diagnostic strategy, a single vector or a subset of vectors can be selected and applied to the erroneous circuit (imitating test experiments). The applet shows the results of testing, and displays also the subset of suspected faults. To improve the diagnostic resolution, additional test vector(s) may be generated and used in the repeated test experiment.

Sequential diagnosis is based on the guided probing strategy. A test pattern is applied and the expected behavior of the circuit is displayed. By clicking on the connection boxes the real values of signals of the faulty circuit can be measured.

The main didactive point in learning the both diagnostic strategies is to try to localize the fault by as few test vectors



#	X1	X2	X3	X2	I0>q	I1>q	I2>q	I3>q	I4>q	I5>q	I6>q	Y
1	0	0	0	0	1	1	0	0	0	0	0	0
2	0	0	1	0	1	1	0	0	0	0	0	0
3	0	1	0	1	1	1	0	0	0	0	0	0
4	1	1	0	1	0	1	0	1	0	1	0	0
5	1	1	1	1	0	0	1	1	1	1	1	1
6												
7												

FIGURE 4
The whole Living Picture for Teaching Digital Systems Design and test

(in the combinational approach) or by as few measurements (in the case of sequential approach) as possible. In this task a competition between students can be carried out which makes the "play" with the applet even more exciting.

As an example in Figure 4, a test generation mode is activated for a design No 2. The result is shown on the schematic panel in the form of activated paths and detected faults. The values in boxes show the behaviour of connection lines of the circuit for the test vector No 4. The activated faults are highlighted by coloured lines, the value 0 (or 1) in the boxes means that the fault stuck-at-1 (or stuck-at-0) is activated. Only a single path from the input X3 through the gates I1, I4 and I6 is activated up to the output Y, hence, only the faults on that path can be detected and observed on the output. The detected faults are: stuck-at-0 on the gate output I1, and stuck-at-1 on the input X3 and on the gate outputs I4 and I6. There are other faults also highlighted and activated inside the circuit on the inputs X!, and X2, and on the gate outputs I0 and I3, however, these faults are not detected because the activated path is broken on the gate output I5, and the error signal is not propagating up to the observable output Y. The data panel shows 5 simulated test patterns, the pattern No 4 which was analysed above is selected.

The described program can be used for teaching the basics of testing digital systems. The teacher can use the applet during the lecture explaining the basics of the topic. The applet can be used also during the exam for giving some tasks to students. Students can use the same applet for training purposes. They can insert different possible faults, and watch how the faults change the circuit's behavior at different input patterns, how the test patterns can be generated to detect a given fault, or how the faults can be localized by test patterns.

After acquiring some experience in these topics by "playing" with applets for relatively small and trivial predesigned circuits, the students can start laboratory experiments on larger circuits designed by themselves or given by the teacher. For that purpose a set of low-cost PC-based tools for training digital test has been developed at TU Tallinn.

PC-TOOLS FOR ADVANCED 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: schematics capture, layout editors, simulators and place and route tools. Low-cost systems for solving a large class of tasks from the dependability area - test generation and fault analysis, fault simulation and diagnosis, testability analysis, BIST modeling and quality analysis, especially for teaching purposes, are missing. For this reason, at TU Tallinn the diagnostic software Turbo-Tester [8,9] was developed.

After theoretical investigation of the basic 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 (TT) diagnostic software is used.

The TT software consists of a set of tools for solving different test related tasks by different methods and algorithms: test pattern generation by deterministic, random and genetic algorithms, test program optimization (test compaction), fault simulation and fault grading for combinational and sequential circuits, multi-valued simulation for detecting hazards and analyzing dynamic behaviour of circuits, testability analysis and fault diagnosis. The representation of the circuit can be given at gate- and macro levels which gives a possibility to investigate the complexity issues of different test algorithms.

TT can be installed under MS Windows/NT and Solaris operating systems. TT can read the schematic entries of various contemporary VLSI CAD tools, e.g. Cadence, Synopsys, Mentor Graphics, Viewlogic, Compass, OrCAD, etc. which makes TT independent of the existing design environment. Many of the commercially available and in-house test design systems have usually problems with the design interface. Similarly to the Java-based living pictures, TT tools are accessible over Internet [10].

LABORATORY COURSES FOR TRAINING TEST

On the basis of TT software, advanced laboratory courses have been developed whose aim is to teach and train students to integrate design and test, and to give them knowledge on how to create testable designs or designs with self-testing capabilities and how to obtain test patterns of better quality. The following laboratory works were developed to train the engineering skills for the field of test:

- Test generation
- Design for testability
- Built-in self-test
- Design error diagnosis

Test generation. The goal is to get acquainted with the problem and to get experience of working with CAD tools in creating test patterns for digital circuits. At first, tests for the given circuit are generated manually. The fault simulation tool evaluates the quality of the manual tests. Then three different test-generating tools (based on deterministic, random and genetic algorithms) are used and compared with each other. Test generation is carried out at gate and macro levels. Macro-level description affords less complexity compared to the gate-level. One of the goals of the work is to analyze how the complexity influences on the productivity e.g. on the speed of test generation.

Design for testability. The goal is to show how the management of controllability and observability of test

points in the circuit can improve the quality of testing. 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 received, the circuit will be redesigned with the goal to get finally a test with a good quality i.e. with a 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 [6,7] consisting of a test pattern generator (TPG), unit under test (UUT) and a response analyzer (RA).

TPG and RA are usually based on a LFSR. There are several disadvantages of such a structure: the tests generated are usually long, and they do not guarantee sufficient fault coverage. To overcome these drawbacks, combining on-line generated pseudo-random test patterns with pre-generated and stored test patterns (hybrid BIST approach) may be used. In this approach, a test engineer should solve the following problems:

- to find the best LSFR configuration for on-line test generation to achieve the highest fault coverage at the minimum length of pseudo-random test sequence;
- to find the best LFSR for response analysis to guarantee the minimum loss of accuracy in fault detection;
- to find the best level of mixing pseudo-random and stored tests as tradeoff between memory cost and testing time.

To find solutions for these problems 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 laboratory work on optimizing the hybrid BIST structure is a step towards teaching research for students in the framework of a university lab.

Design error diagnosis. The goal is to learn how to compose diagnostic tests to localize the faults in a given circuit. Iteratively using CAD tools, theoretical reasoning and manual work for generating additional "better" tests, students will get experience in solving extremely demanding engineering challenges.

The courses have received good credits from students of Michigan State University (USA), Helsinki University of Technology (Finland), Jonköping University (Sweden). It is under consideration to utilize TT SW for teaching Design for Testability in other universities of East- and West-Europe.

CONCLUSIONS

By the use of web-based media we achieve: presentation of course material independent of place and time, individual learning according to the students' own needs, quick cross-referencing by hyper-linked texts, new forms of communication between teachers and students (chat, joint editing), up-to-date course material.

The conception presented allows to improve 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 allow 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 is supported by the Thuringien Ministry of Science, Research and Art (Germany), by EU Copernicus project VILAB, and by the Estonian Science Foundation, grants No 3658 and 4300. We also thank Elmet Orasson for programming the applets, Jaan Raik for the Turbo Tester software development, and Artur Jutman for setting up the laboratory course.

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] Wuttke, H.-D., Henke, K., Peukert, R., "Internet Based Education - An Experimental Environment for Various Educational Purposes". *Proc. of the IASTED Int. Conf. on Computers and Advanced Technology in Education*, Philadelphia, PA USA. IASTED/Acta Press No. 292, May 6-8 1999, pp. 50-54.

[5] Ubar, R., Wuttke, H.-D., "Action Based Learning System For Teaching Digital Electronics And Test" *,3rd European Workshop on Microelectronics Education" (EWME 2000), Aix-en-Provence, Kluwer Academic Publishers,* May 18-19 2000, pp. 107-110.

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

[7] M.L. Bushnell, V.D. Agrawal. Essentials of Electronic Testing for Digital, Memory and Mixed-Signal VLSI Circuits. Kluwer Academic Publishers, 2000, 690 pp.

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

[9] Ubar, R., "Teaching Dependability Issues in System Engineering at the Technical University of Tallinn", *Global J. of Eng. Education*, Vol.2, No 2, 1998, pp. 215-218.

[10] http://www.pld.ttu.ee/tt/