Distance Learning Tools for the Field of Electronics Design and Test

Raimund Ubar
Department of Computer Engineering
Tallinn Technical University
Raja 15, 12618 Tallinn
Estonia
raiub@pld.ttu.ee

***Abstract* – A conception of a training system for teaching design and test of electronic circuits is presented. A set of tools was designed for exercising design, test and diagnostics related problems in gate-level digital circuits and register transfer (RT) level digital systems. Such topics as investigation of tradeoffs between speed and hardware cost in a digital design, gate- and RT-level simulation, fault simulation, test pattern and test program generation, built-in self-test (BIST) and other similar problems are covered by the described distance learning environment. Part of the tools are implemented in a form of Java applets and can be freely accessed via Internet. A set of circuits as diagnostic objectives for the applets are predesigned and can be selected by the student. Another part of the system consists of diagnostic tools which can be used for solving test related tasks for circuits or systems designed by students themselves, using any standard CAD tools. The access to the tools via internet makes it easy for students even from foreign universities to use the distance-learning environment any time and in any place. The Java applets have built-in multilingual support to ensure easy integration into teaching courses of universities over the world**.

I. INTRODUCTION

Distance-Learning is currently a hot research and development area. Benefits of internet-based education are clear: asynchronous learning, classroom and platform independence. An application installed and supported in one place can be used in other places all over the world. Most of internet-based courses made available in recent years are not more than a network of static hypertext pages. The challenge is to develop web-based educational applications, which can offer some amount of interactivity and adaptivity.

Rapid advances in the areas of deep-submicron electron technology and design automation tools are enabling engineers to design larger and more complex circuits and to integrate them into one single IC. System on a Chip (SoC) design methodology is seen as a major new technology and the future direction for semiconductor industry. On the other hand, the cost of test and verification because of the increasing complexity of systems has become a significant part of the total cost of electronic products. This 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 electronics engineers. This is why the curricula in system and electronics engineering should incorporate a greater emphasis on design for testability (DFT) and on the concepts of test. Teaching in this domain should be facilitated by using integrated CAD tools that support verification, design for testability, test generation, built-in self-test, fault diagnosis and fault tolerance.

In the following a set of tools for supporting distance learning in the fields of design and test of electronics systems will be presented. The functionality of the tools, and the scenarios of using the tools in the laboratory research will be described.

The paper has the following structure. In Section 2, the roles of the tools for two different levels of learning (learning of basic problems, and research-oriented learning) will be specified. Section 3 describes the conception of learning based on "living pictures" which are implemented as Java applets. In Section 5 the applets for learning test in gate-level circuits, and in Section 6 the applets for learning test in register transfer level circuits are described. Section 6 presents the tool set for research oriented training in digital test. Finally, in Section 7 some conclusions are drawn.

II. TWO LEVELS OF LEARNING DESIGN AND TEST

In the paper a conception is presented for training students to be educated for electronics and SoC design to improve their understanding and skills in test related topics. The training consists of two parts: interactive learning of basic problems by using web-based interactive learning modules, and laboratory research based learning of advanced problems by using low-cost diagnostic tools.

The first part of the training is based on using so-called *living pictures* [1]. The method presented deals with the goal to put interactive training modules to the Internet that can be used in a lecture as well as for individual self-studies [2]. They can be accessed independent of time and place. On one hand, teachers can show more complex examples during the lecture 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.

Two types of applets have been developed: applets for investigating and learning test problems in simple gate level circuits, and applets for practicing design and test problems in more complex digital systems, consisting of control and data paths.

The second part of training has a target to give students already equipped with basic knowledge the possibility to go more deeper into the essence of design and test problems, to get aquainted with different tools, methods and models, and to get hands-on experience in solving

some practical problems. This part of training is based on using a special set of tools for test generation and fault simulation to be used in the complex circuits designed by students themselves.

The second part can be carried out partly in laboratory, partly by using Internet.

III. LEARNING BY JAVA APPLETS

The teaching software developed for learning electronics testing supports the action based training via internet. The software offers a set of tools to inspect the objective to be learned and access to multiple learning modules. 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 Java-applets (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 (either manually or using built-in generators) and watch the reactions. In reaction of the inputs, *simulation or diagnostic components* can be started, executing a selected method that has to be taught, and presenting the results using *visualization components*. There are also *explanation components*, describing unknown methods step by step on the actual chosen or generated example.

The software 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 appletviewer.

IV. APPLET FOR LEARNING DIAGNOSTICS OF ELECTRONICS CIRCUITS

The software developed for excercising gate-level circuits can be used for teaching the basics of Digital Test and Testable Design as illustrative tool explaining the problems of fault modeling, fault simulation, test generation and fault diagnosis [3].

The work window of this program (Fig.1) consists of three parts - vector insertion panel, circuit view panel for presenting design schematics, and *data view* panel for presenting test vectors, fault tables and waveforms. Vector insertion panel is used for stimulating the circuit by test patterns. In the circuit view panel signals can be inserted directly on the lines when test generation procedures are learned The boxes at lines on schematics are clickable for inserting proper signals for activating faults and propagating error signals through the circuit. In the data view panel the results of different procedures are reported: the test patterns after they have been generated, and the fault tables after the faults are simulated for the given test patterns.

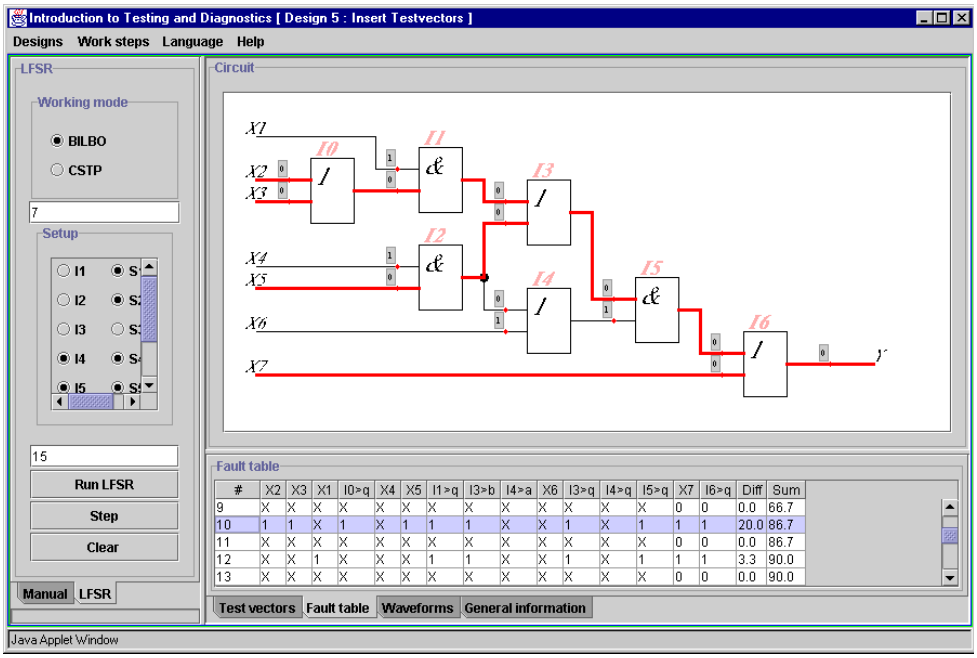

Fig.1. Java Applet for learning electronics test

The following test related topics are supported by the "living picture":

- fault simulation,
- test generation,
- testability analysis, and
- built-in self-test.

The key problems can be taught and learned using the software on different examples.

In the *test generation mode* the students can choose a target fault in the schematic, create step by step proper activated paths in the ciruit to activate the fault on the site, and to propagate the error signals caused by the 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 the students 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 faults are displayed also on the data panel in form of a row of 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 test vector on the data panel, all the detected faults will be highlighted by colours on the schematic panel. The fault table generated by fault simulation is used in diagnostic working modes.

Different gate-level diagnosis and fault localization strategies can be investigated: combinational and sequential ones.

For learning the *combinational diagnostic* strategy, a single vector or a subset of vectors is selected and applied to the erroneous circuit (imitating a test experiment). The applet shows the results of testing, and displays the subset of suspected faults. To improve the diagnostic resolution, additional test vector(s) can be generated by the applet 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. The principle of guided-probe testing is to backtrace an error from the output where it has been observed to its source (faulty gate). By clicking on the connection boxes, the actual values of signals of the faulty circuit can be measured. A faulty gate is located if it has been found that the signal on the output of the gate is faulty, while only expected signals are observed at its inputs.

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

V. FROM SIMPLE CIRCUITS TO COMPLEX SYSTEMS BY JAVA APPLETS

Entering the SoC era with its new concepts means teaching on higher levels of abstraction like core level, register transfer level (RTL), instruction set architecture or behavioral levels. Another Java applet has been developed for learning high-level design and test problems in control intensive digital systems [4]. Such topics as investigation of tradeoffs between working speed and hardware cost in digital design, RT-level simulation, fault simulation, test generation, built-in self-test (BIST) and other related problems are covered by the training system.

While *designing a system* (implementing a given algorithm or a function like multiplication, division etc.) a student can select needed microoperations for each unit of data path from the whole set of possible predesigned microoperations. Different architectures can be chosen for implementation of a given function. The control path is a microprogrammed controller which implements a finite state machine (FSM). The microprogram is developed by the student to realize a given algorithm based on the available (selected in prior) resources of the data path. The user fills in the microprogram table represented as a subpanel of the applet. Each microoperation has a gate-level implementation, and the number of gates determines the cost of the microoperation. By selecting a set of microoperations for the whole data path the student will get also the cost of the data path in the number of gates. Students can compare different solutions and find the tradeoffs. For every chosen architecture, the system calculates the cost of hardware. The speed (the number of clock cycles the microprogram needs) can be measured by simulation.

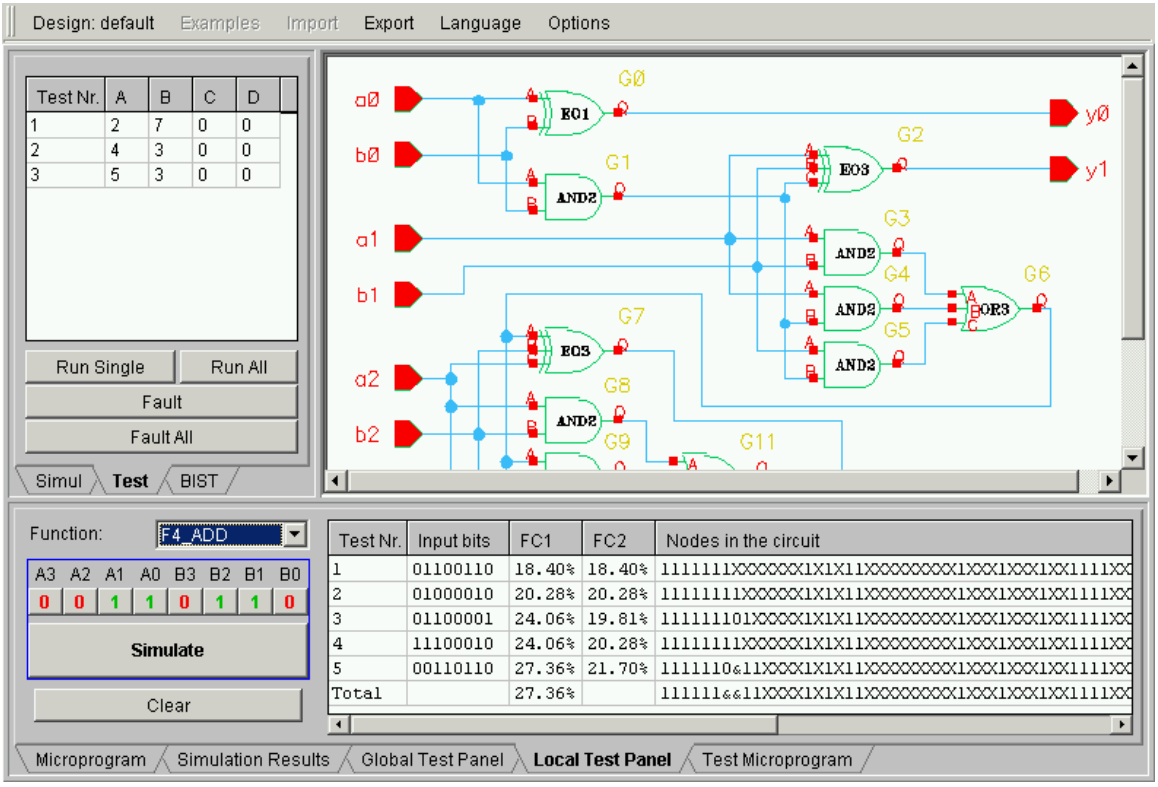

Fig. 2. Java applet for learning register transfer level test

Simulation of microprograms is carried out at the higher register transfer level by using Java subroutines (corresponding to functional units) which are activated by the control signals in the order given in the microprogram table.

For investigating test related problems of digital systems two-level simulation is used. For the fault simulation, stuck-at fault model is used. Fault insertion is carried out on the gate level whereas fault propagation is processed on the register transfer level.

Different strategies for testing can be exercised:

- self-testing by the implemented user microprograms (called "functional testing"),
- generating special test microprograms for selected blocks,
- different Built-in Self-Test (BIST) architectures like (BILBO, circular self test path, "store and generate" a.o.) [6] for selected cores or blocks.

In the case of self-testing by user microprograms, the quality of test (fault coverage) will be the objective of research. By fault simulation the fault coverage for a selected block of the system will be calculated. The quality of test can be improved by selecting proper data for the given microprogram. The students can work creatively by choosing these data.

In Fig.2, the fault simulation results for two target blocks FC1 and FC2 are depicted, which show how the test coverage is increasing when processing the microprogram with different data.

Self-testing by user microprograms, however, will have its limits when high fault coverage for functional blocks is needed. The reason of the low fault coverage is that the user microprograms are not developed for testing purposes, and the fault coverage can be controlled only by selecting the data.

To improve the test quality, special test microprograms can be developed by the designer. The target of the particular test microprogram will be selected: a block or several blocks in the data path. The microprogram will be developed on the higher register transfer level whereas the test data for testing the target block(s) are generated at the gate-level.

In Fig.2, a gate-level implementation of a target block is shown. The fault simulation results for a target block will be also shown (like for F1 and F2) in a particular column of the report table.

Self-testing by dedicated test microprograms, however, may have also its limits in reaching high fault coverages for functional blocks. The reason lays usually in the low testability of the system. To improve the quality of test the logic BIST approach can be used [6].

In the applet different approaches of BIST can be investigated. The corresponding BIST architecture can be selected and emulated by the applet, and the quality of the chosen test approach for the particular circuit can be measured in terms of fault coverage.

New emerging technique called "functional BIST" [7] can be also investigated by combining different configurations of blocks with the goal to estimate their suitability for functional BIST.

Special BIST subpanel with BIST results subpanel can be opened for the BIST operation mode. By scan-path technology the inputs and the outputs of the combinational blocks F1, F2, F3 or F4 in the data path can be made directly accessible (Fig.3). Pseudorandom test pattern generator (TPG), signature analyzer (SA), and TPG/SA (combined TPG and SA) can be configured and emulated by the applet.

The mentioned BIST architectures can be implemented in two ways: by choosing different settings for each combinational circuit to be tested, or by choosing the same setting for all the circuits. The aim of a student's work is to find best settings. For setting the polynomial, the initial state of the TPG, and the number of clocks to be used for test generation, there is a special subpanel.

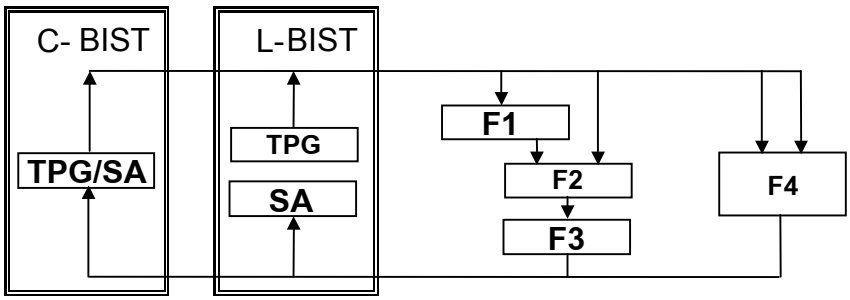

Fig. 3. Scan -path design and BIST

The described applet gives the students a possibility to investigate, compare and "play" with most important testing conceptions used in today's complex digital systems like SoC.

VI. COMPREHENSIVE TOOL SET FOR LABORATORY TRAINING IN DIGITAL TEST

In the described applets, a restricted set of designs prepared earlier are used by students for diagnostic experiments. In the following, a laboratory environment will be presented where students can exercise their own designes created by any standard CAD tools.

Traditional VLSI test generation and fault simulation software on workstations is 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, place and route tools etc. 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 TU Tallinn a diagnostic software Turbo-Tester (TT) was developed [8]. The main functions of TT are depicted in Fig.4.

After learning the basics of theory by "playing" with above described applets, a laboratory work follows with more complex and realistic designs, where any available design software (schematic editor as minimum), and TT diagnostic software can be used in combination. For this combined work dedicated interfaces have been built. Access to TT is possible via internet.

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 design for testability,
- fault and design error 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.. Similarly to the Java-based living pictures, TT tools are accessible over Internet [9].

A lot of different options available in TT give the students opportunity to compare different models and methods used in the testing practice.

The students develop circuits as diagnostic objectives, investigate by TT the testability of circuits, redesign them if neccessary for improving the controllability and observability of test points, insert self-test BIST structures, analyse the efficiency and trade-offs of different BIST solutions, and learn to make proper engineering decisions in the field of testable design.

An example of a research scenario students can exercise by using TT tools is as follows. By means of any CAD tools (TT has EDIF interface which is a link to most commercial CAD tools), a design can be created for further diagnostic experiments. To investigate how different models of the design can influence to the efficiency of test tools (e.g. to the speed or quality of test generation or fault simulation), two representation levels can be chosen: gate-level and macro-level networks. The latter one means that the whole circuit is partitioned into a set of subcircuits, where each subcircuit is represented by a compressed model which allows to reduce the complexity of the whole model of the system compared to the initial one.

The Fig.4 depicts some possible experimental work flows targeted to testing and fault diagnosis. Test patterns can be generated by different algorithms with the goal of comparing the efficiency of algorithms. Test optimization is an option to improve the available tests, to reduce the test length and to reach faster test experiments. On the other hand, shorter test sequences will provide less diagnostic resolution. This conflict between the length of the test and its diagnostic resolution present another attractive research issue to look for tradeoffs between these two objectives.

Several interesting laboratory research scenarios have been developed based on using the environment of TT:

- Test generation in digital circuits. Comparison of different methods, analysis of the influence of complexity of circuits to the performance of tools.
- Fault simulation in digital circuits. Analysis of the influence of complexity of circuits to the performance of tools.
- Design for testability. Evaluation of the testability of a given circuit, and redesign of the circuit for improving its testing quality.
- Fault and design error diagnosis. Improving the diagnostic resolution of the given test sequence by generating additional test patterns.
- Built-in Self-Test. Comparison of different methods and finding tradeoff between the speed and quality of testing. The following architectures like BILBO, Circular-Self-Test-Path, and Store-and-Generate can be emulated.
- Functional Self-Test. Evaluation of the quality.

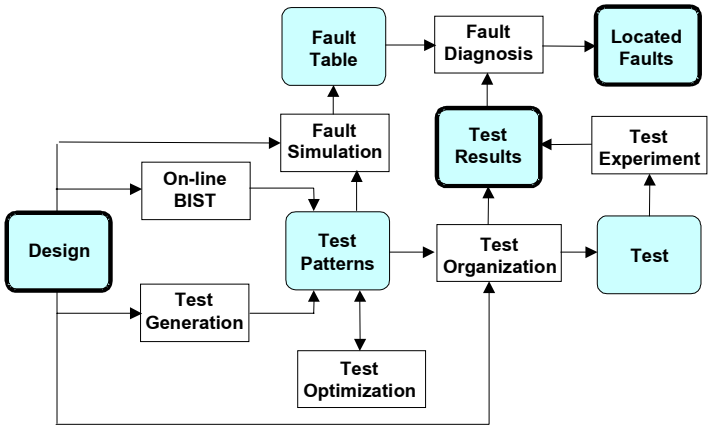

Fig.4. Turbo-Tester tool environment

The listed scenarios have been introduced into the curricula at Tallinn Technical University in Estonia, and at the University Jönköping in Sweden. The laboratory research scenarios have been successfully used also in the summer school for master students at TU Darmstadt in Germany.

VII. CONCLUSIONS

An original set of tools supported by learning scenarios in teaching electronics design and test was developed at Tallinn Technical University.

The tools and methodology support the distance-learning conception of education. By the use of web-based media we achieve: presentation of teaching material independent of place and time, individual learning according to the students‘ own needs, new forms of communication between teachers and students, availability of up-to-date learning materials.

The environment and conception presented allow to improve the skills of students to be educated for digital hardware and SoC design connected with test related topics. The system fully reflects the “easy action and reaction” conception which was taken as the major target for creation of the environment.

The tasks chosen for training represent simultaneously real research problems. This provides students with dynamic environment to experiment with and to find interesting solutions for stated problems. The main target of the described system is to foster in students critical thinking, problem solving skills and creativity in a real research environment and atmosphere.

VIII. ACKNOWLEDGEMENTS

This work has been supported by EU V Framework projects REASON and eVIKINGS II, by the Thuringien Ministry of Science, Research and Art (Germany), and by the Estonian Science Foundation, grant No 4300.

The author thanks the colleagues and students of TU Tallinn involved in the development, especially Jaan Raik, Artur Jutman, Elmet Orasson and Sergei Devadze. Special thanks of the author belong to prof. H.-D.Wuttke from TU Ilmenau (Gemany) for numerous discussions during the development of the conception and tools.

IX. 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, 1999.
2. R.Ubar, H.-D.Wuttke, “The DILDIS-Project – Using Applets for More Demonstrative Lectures in Digital Systems Design and Test,” *Proc. of the 31st ASEE/IEEE Frontiers in Education Conference, FIE’2001,* Reno, NV, USA, Oct. 10-13, 2001, pp.. SIE-2-7.
3. R.Ubar, A.Jutman, E.Orasson, J.Raik, T.Evartson, H.-D.Wuttke, “Internet-Based Software for Teaching Test of Digital Circuits,” *In the book "Microelectronics Education", Marcombo Boixareu Ed*., 2002, pp.317-320.
4. S.Devadze, A.Jutman, A.Sudnitson, R.Ubar, H.-D.Wuttke, “Teaching Digital RT-Level Self-Test Using a Java Applet,” *20th IEEE Conference NORCHIP’2002*, Copenhagen, Denmark, November 11-12, 2002, pp. 322-328.
5. R.Ubar, “Multi-Valued Simulation of Digital Circuits with Structurally Synthesized Binary Decision Diagrams,” *OPA (Overseas Publishers Assotiation) N.V. Gordon and Breach Publishers, Multiple Valued Logic,* Vol.4, 1998, pp. 141-157.
6. M. Abramovici et. al., “Digital Systems Testing & Testable Designs,” *Computer Science Press,* 1995, 653 p.
7. S.Cataldo, S.Chiusano, P.Prinetto, H.-J.Wunderlich, “Optimal Hardware Pattern Generation for Functional BIST,” *Int. Conf. Of Design Automation and Test in Europe – DATE,* Paris, March 27-30, 2000, pp.292-297.
8. 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,* 1998, pp.287-290.

9. http://www.pld.ttu.ee/tt/