Bringing Research Issues into Lab Scenarios on the Example of SoC Testing

Raimund Ubar[1], Artur Jutman,[2] Sergei Devadze [3], Heinz-Dietrich Wuttke [4]

***Abstract* - A conception and a teaching/learning environment is presented to improve the teaching quality in the field of system on chip design and test. Both, gate level and register transfer level test and design for testability problems are covered. In this environment, different embedded built-in self-test architectures and their quality can be evaluated by fault simulation. Since plain low-level methods in this field have lost their importance, hierarchical approaches are supported by this environment. The system supports distance learning as well as a web-based computer-aided teaching. The interactive modules are focused on easy action and reaction, learning by doing, a game-like use, and on encouraging students for critical thinking, problem solving, and creativity.**

Index Terms – computer-aided teaching, distance learning, research-oriented learning, digital systems, system-on-chip, design and test, built-in self-test, gate and register transfer levels, hierarchical approaches.

INTRODUCTION

Recent reviews have discovered that most VLSI and system designers know little about testing and design for testability (DFT) of today's digital systems because of the gap in education. The importance of test and fault diagnosis as a teaching objective is underestimated in traditional engineering education [1]. Test is usually taught as a not very important subtopic in a design course. In most cases it is taught as an independent discipline only when it is a "hobby horse" of a professor. There are two reasons for that. The first one is because the test is interpreted as a non-productive (read: not important) issue vs. design. The second one can be explained by so called Tenhunen's Law which claims that the number of courses that should be taught at universities doubles in a decade [2]. To select courses for curricula is a difficult issue. And often a test course as a component of engineering education is left outside the curricula because of tough competition between courses.

On the other hand, the topics like Testing and Fault Diagnosis are not only Electronics Systems related issues, they have an important didactic role for the engineering education in general [3]. First, testing is a method to learn how to ask right questions, second, it develops the ability of analysing cause-effect relationships, and third, diagnosis is looking for answers to the questions like "what is the reason of that what happened?"

Logic world (computers, digital circuits and systems, systems-on-chip) because of its inherent logical complexity could be the best objective for learning the concepts of testing and diagnostic analysis for any type of system in general. The real targets of education are: creativity, critical thinking, and problem solving skills. Therefore, learning test at a university should be research oriented.

Moving towards multi-million gate System-on-Chips (SoC) makes embedded test strategies via Built-In Self-Test (BIST) architectures mandatory. It is critical to ensure that students will be equipped with skills in DFT and BIST, and will get hands-on experience in solving test problems in digital systems like SoC.

A conception and a teaching/learning environment is presented to increase the teaching quality in the field of electronics DFT. Both, gate level and register transfer level (RTL) test problems are covered. In this environment, different embedded BIST architectures can be emulated and their quality can be evaluated by fault simulation. Traditional low-level test synthesis and analysis methods for digital systems have lost their importance because of the complexity reasons. As a well proven solution, the hierarchical approach to testing complex embedded digital systems can be investigated in hands-on manner. The system supports distance learning as well as web-based computer-aided teaching. The interactive modules are focused on easy action and reaction, learning by doing, a game-like use, and encourage students for critical thinking, problem solving, and creativity.

The paper is organized as follows. Section 2 gives an overview of the new teaching concept supported by the paper. In Section 3 the subsystem for RTL Design is described. Section 4 and 5 present the concepts of teaching test topics, and in Section 6 and 7 the results of practical experiences and conclusions are made.

SYSTEM OVERVIEW

The core of the teaching concept and the environment with remote access presented here is implemented as a Java-applet of a special type, which we call "Living Pictures" [4]. Those applets simulate tricky, quite complicated situations of the learning subject in a graphical form on the computer screen. The graphics is self-explanatory and provides interaction possibilities. By using these possibilities the students can generate examples that are interesting enough to encourage their own experiments but not too complicated for learning.

[1] Raimund Ubar, Tallinn University of Technology, raiub@pld.ttu.ee
[2] Artur Jutman, Tallinn University of Technology, artur@pld.ttu.ee
[3] Sergei Devadze, Tallinn University of Technology, serega@pld.ttu.ee
[4] Heinz-Dietrich Wuttke, Ilmenau Technical University, dieter.wuttke@tu-ilmenau.de

The proposed environment supports four phases of the learning process: listening to the teacher in the classroom, repetition at home of the scenarios discussed in the classroom, practice in form of hands-on solving research problems, and examination phase. The system supports the action based training since for each phase there exists a special application service adapted to the learning process which allows different views and actions using the same interactive module.

The proposed environment allows investigating and solving a lot of different problems related to RT-level digital design and test. More precisely, the teaching, learning and research is supported in the following topic fields [5]:

- design of a digital system consisting of data and control paths on RT level;
- synthesis of algorithms and micro programs for the given computing tasks;
- performing the RT-level system simulation to validate the functionality of design;
- optimization of the architectures of the data part by finding tradeoffs between the performance of the system and the hardware cost;
- performing fault simulation at both gate and RT-levels;
- generating functional test programs;
- generating gate-level and RT-level deterministic test programs using hierarchical approaches;
- calculation of the quality of test programs;
- design of different BIST architectures like logic BIST, circular BIST, functional BIST, etc.;
- analysing the quality of BIST architectures;
- design and optimization of hybrid BIST or hybrid functional BIST solutions;
- design for testability to improve the quality of test programs.

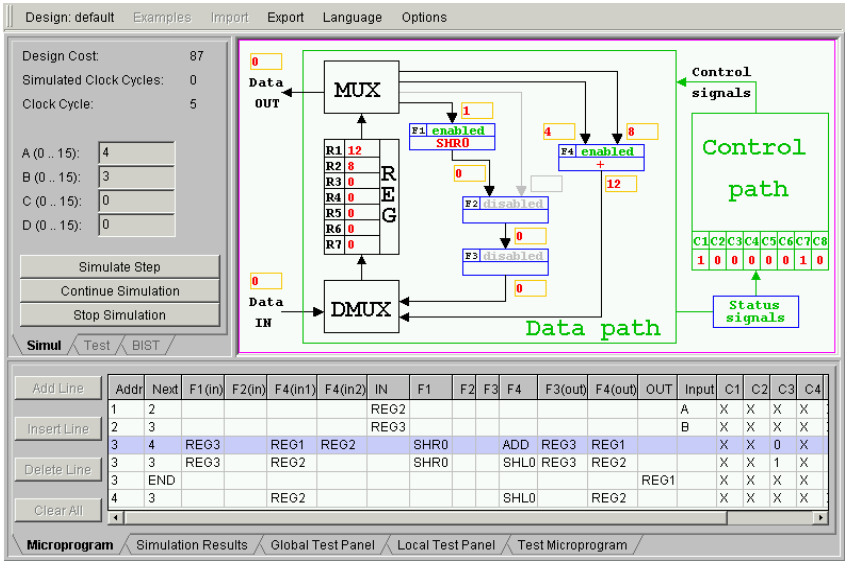

FIGURE 1
OVERVIEW OF THE WORKING FIELDS OF THE TRAINING SYSTEM

To enable learning and research of the listed topics we have developed a convenient user interface (Fig.1) which consists of the following major parts.

- Schematic View panel provides the schematic representation of the target system and the graphical simulation data. The internal structure of the data path is also reflected there.
- Microprogram Table is used to define the behaviour of the control path of the system. During simulation this panel shows which line of the microprogram (clock cycle) is currently executed.
- Simulation and Test tab-panels allow choosing and running RT-level fault and fault-free simulation. The simulation can be performed for a single set of input data (step-by-step or at once) as well as for all the sequence of input operands at once.
- Simulation Results tab-panel reflects the results of fault-free simulation.
- Fault simulation module provides fault simulation for the data path and its units.
- Global Test Panel is used to provide fault coverage information as for the whole data path as for each single unit under test.
- Local Test Panel provides means for manual local test pattern generation for a selected component or subcircuit of the data path. It also displays the gate-level schematic of the unit and the fault coverage for each unit as well as for the data path as a whole.
- Test Micro program is used to organize separate test access to each selected functional unit of the data path.

The applet has a built-in extendable collection of examples or demos – solutions for implementing different algorithms in a form of data path architectures and microprograms. The given examples help users to understand principles of the system operation. For connecting the system to other applications as well as for providing users with a possibility to save the results of their work for further use applet has a data import/export capability.

SUBSYSTEM FOR RT-LEVEL DESIGN

To each functional unit (FU) of the given data path structure (for example, units F1…F4 in Fig.1) a set of controlled micro-operations (unary or binary functions) is corresponding, where each of them can be activated by selected control signals. The user can define one or more micro-operations for each unit of data path when implementing his own algorithm (like multiplication, division etc.). Each micro-operation has a gate-level implementation, and the number of gates determines its cost and in the end the final hardware cost of the whole system. The user can select, thus, a particular structure of the data path and the implementation of his algorithm meeting either the cost or timing requirements. The working speed of the algorithm defined by the number of needed clock cycles is measured by simulation. The simulation is supported by an RT-level model of the system as a whole and by gate-level models of each micro-operation in each FU.

The control path is a micro-programmed controller [6], which implements Mealy Finite State Machine (FSM). The controller consists of a microprogram table and an interpreter. The microprogram is developed by the user to realize a given algorithm based on the selected resources of the data path. The user fills in the rows of microprogram table, which contain information about the address of the current and the next microinstruction, MUX and DMUX configurations, Data IN values, selection of functions in FUs (F1 to F4 in Fig.1) at each microinstruction, and status signal configuration.

In Fig. 1 an example of an algorithm of multiplication of two operands *A* and *B* is presented. The result of the operation is stored in *REG*1 and fed out to the data output.

The RT-Level simulation is carried out at the higher level by using corresponding to functional units Java subroutines which are activated according to condition values by the control signals in the order given in the microprogram table. The simulation data are stored in the Simulation Results sub-panel. These data reflect the states of all the registers, outputs of all the functional blocks, data input and output of the device, current states at each clock cycle and condition signals. The simulation data can be used by the student as a debugging info as well as for improving the efficiency of the operation – either the speed or the cost of the system.

For more details on design-oriented part of our system, please visit the dedicated web page [7] and turn to our previous article [8] in which we made emphasis in RT-level design. In the present article, we will concentrate mostly on test-related topics.

TEACHING RT-LEVEL TEST

The toolkit of the modern design and test engineer contains quite a few methods of testing of a SoC design. Most of them have come from the earlier times and have been adopted for the new paradigm. In general, these commercial tools are first, costly for introducing in the classroom, second, are very complex to handle for newcomers, and not well suited for teaching purposes. With our very simple and low-cost teaching environment we are aimed at showing a variety of different modern testing techniques including functional and deterministic testing, a number of BIST solutions including the most advanced combinations of tools like different hybrid BIST solutions.

Prior to entering the test mode, the system under test must be designed and verified. The user can design a system himself or he can use one of the prepared examples. When the test mode is selected, the microprogram and the structure of the data path are "frozen" and cannot be modified anymore. At the same time the user selects target micro-operations of the data path for test generation and fault simulation. The fault simulation information is reflected (depending on a selected mode) on the Global Test Panel for the whole system and on the Local Test Panel for a single selected unit. In the following we describe the test modes that can be investigated in details.

Functional Test

In this mode the simplest and cheapest test technique is investigated, which does not require designing special test programs and embedding of special DFT structures into the system. The same unmodified microprogram and data path structure are used instead. The required level of fault coverage must be achieved then by only a smart selection of input data. The only checkpoint allowed for observing the erroneous behaviour of the system is the data path primary output. Moreover, it only can be observed at the time when the microprogram outputs the final result.

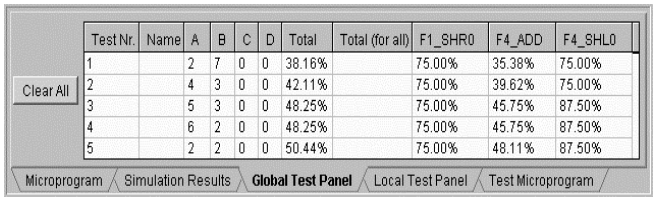

FIGURE 2
GLOBAL FAULT COVERAGE TABLE

The fault simulation information for a multiplier is presented at the Global Test Panel (Fig. 2). The input operands (*A*,*B*,*C*,*D*) are specified first. The same microprogram is used then repeatedly for fault simulation of all the input data. The fault coverage is calculated for each selected FU and for the whole system (as total) as well. As an example, the cumulative fault coverage for five input data pairs for the multiplier is provided in the Global Fault Coverage table (Fig. 3).

The primary task of the student during investigation of functional testing is the selection of "good" operands in order to achieve the targeted total fault coverage as fast as possible. Usually this technique does not allow to achieve sufficient fault coverage because of the low observability. A lot of faults activated during the procedure tend to mask them before the erroneous signals can be observed in the end of the procedure.

Deterministic Test

Because of the complexity of plain gate-level models the test generation for complex digital circuits like cores in SoCs should be performed in a hierarchical way: local test generation for smaller units of the circuits is performed at the gate level whereas the global test program integrating the low level local test patterns is created at the RT level.

The gate-level test generation and fault simulation for each selected FU is carried out separately. The FUs are considered by the user in series and test vectors are generated for each of them. The simulation results can be viewed in the fault table at the Local Test Panel. For each vector its fault coverage is calculated and the information about tested nodes is given. The cumulative fault coverage of the tested FU is also shown for each simulation step. Hierarchical RT-level fault simulation is applied in order to evaluate the global fault coverage of the test vectors for the data path as a whole. For this purposes a test program is composed for each selected FU. The simulation data is reflected in the Global Test Panel in the same way as it is done in the Functional Test mode.

In order to help the user to generate gate-level test vectors, the gate-level schematic of currently selected FU is displayed. The user selects a target fault and generates a test vector. This procedure can be done manually for selected faults or automatically for all faults at once by using Automated Test Pattern Generator (ATPG) [9]. After pressing the "Simulate" button the test vector is fault simulated at the gate level and the result (local fault coverage) is added into the fault table. At the same time, the same vector is sent to RT-level hierarchical fault simulator in order to fill in the Global Test Panel. The test microprogram, used for RT-level fault simulation must provide a good access to the selected FU. A simple version of such a program is generated automatically. It can be used as a template by a student in order to develop a more sophisticated test program if needed.

The primary task of the student working in Deterministic Test mode is the creation of as short as possible short local tests covering maximum amount of faults for each of selected FUs. Another, more advanced, task is the reduction of the overall test length for the whole investigated system by modification of standard test programs and finding optimal set of local test patterns.

DFT and BIST Modes

The Deterministic Test mode is one of the most efficient ways of testing. However, it does not provide access to internal signals of the system under test to achieve sufficient controllability and observability [5]. This problem is addressed by various DFT and BIST solutions. Usually it is a scan-path with a pseudorandom Test Pattern Generator (TPG) and one or more Signature Analyzers (SA). Both of them can be implemented as Linear Feedback Shift Registers (LFSR) [5]. By scan-path technology (Fig. 3) the inputs and the outputs of the combinational blocks in the data path are directly accessible by TPGs, SAs or TPG/SA (combined TPG and SA) [5].

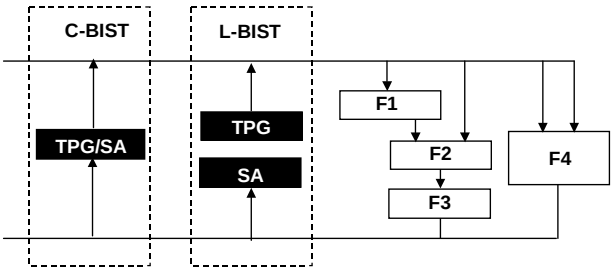

FIGURE 3
SCAN-PATH DESIGN WITH BIST COMPONENTS

Our teaching system allows reconfiguration of internal registers in the BIST mode. Depending on the chosen BIST method some of them can perform functions of TPG, SA or TPG/SA. If the Logic BIST (L-BIST) method is to be evaluated, the TPG and SA functions must be separated and located in different registers. In the case of Circular BIST (C-BIST) approach, both TPG and SA are combined in the same register. In both modes it is possible to configure the TPG on-line from the interactive graphical panel. When the configuration is completed, the gate-level and the hierarchical fault simulation are performed and the results are displayed in the way similar to the one used in Functional and Deterministic test modes.

The described above modes help to illustrate the way of operation of different BIST structures and to show how their efficiency depends on the TPG configuration. The selection of a good configuration of the BIST components for each selected FU is the main problem to solve by the student. Another task is the selection of such a single TPG configuration that allows testing all of the FUs in the shortest possible time.

Functional BIST with DFT

There is another BIST mode, called Functional BIST (FBIST) [10], implemented in the applet.

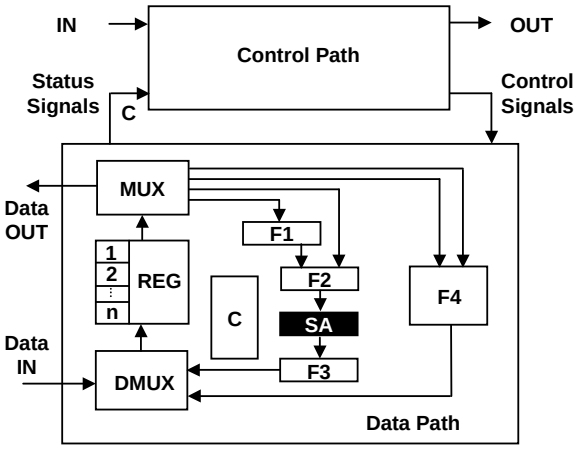

FIGURE 4
FUNCTIONAL BIST WITH INSERTED SA

This mode has very much in common to Functional Testing. The only difference between the two modes is that

now a possibility is provided to insert SAs at any arbitrary point within the data path (Fig.4). In this way we increase the observability of the system, since each such SA is capable of collecting data at each clock cycle compressing the data into an observable signature. The student's task is to improve the efficiency of Functional Testing paradigm by introducing the minimal number of additional test points for inserting SA-s.

RESEARCH ORIENTED LEARNING SCENARIOS

There are several disadvantages of using the described basic BIST approaches. First, the test sequences generated randomly are usually very long; second, they do not guarantee always sufficient fault coverage because of the existence of so called "hard-to-test" faults. To overcome these drawbacks, combinations of several approaches have been proposed.

One of them called hybrid BIST [11] is based on combining on-line generated pseudorandom test patterns with stored pre-generated test patterns. In this approach, at first pseudorandom test sequence with a length L is generated on-line, after that a switch to a stored test approach will take 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 unit under test (UUT) to reach the 100% fault coverage. For generation of stored test patterns ATPGs may be used based on deterministic, random or genetic algorithms [9].

Several problems requiring creative thinking and experimental research have to be solved to find optimal solutions for combining test patterns from different sources. Such problems can be formulated as the following questions:

- What is the best characteristic polynomial and initial state of 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 for a given digital circuits?
- What is the best characteristic polynomial and initial state for LSFR to be used for on-line testing in parallel a given set of circuits, cores or subsystems?
- How to find for a given hybrid BIST (or for a given hybrid functional BIST) the best level of mixing pseudorandom (or, respectively, functional test) and stored deterministic test parts as the trade-off between the memory cost or power consumption and the length of the whole test sequence.

The listed tasks serve as a good basis for laboratory research scenarios which need intensive critical thinking and demand a lot of creativity. 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.

There 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 a clever set of experiments.

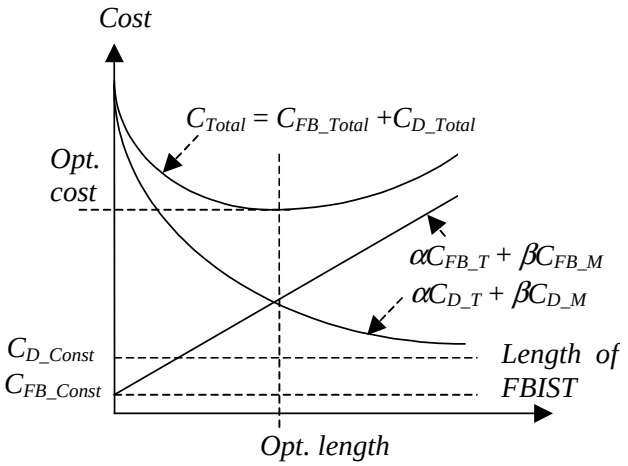

FIGURE 5
FINDING THE BEST MIXED SOLUTION FOR HYBRID BIST

Consider the total test cost C_{TOTAL} of the hybrid FBIST as the sum of total costs C_{FB_Total} and C_{D_Total}, of producing functional and deterministic test patterns, respectively, where

$$C_{FB\_Total} = C_{FB\_Const} + \alpha C_{FB\_T} + \beta C_{FB\_M}$$
$$C_{D\_Total} = C_{D\_Const} + \alpha C_{D\_T} + \beta C_{D\_M}.$$

Here C_{FB_Const} (C_{D_Const}), C_{FB_T} (C_{D_T}), and C_{FB_M} (C_{D_M}) denote, correspondingly, additional logic cost, the cost related to test time, and the cost of memory needed for functional and deterministic test parts, whereas α and β reflect the weights of time and memory costs. An example of the cost curves is shown in Fig.5.

The goal is to find the optimal length of the functional BIST sequence corresponding to the minimum of C_{TOTAL}. The point in this research is that it would be very time consuming to find experimentally all the curves shown in Fig.5 except for C_{FB_Total}. The static component C_{FB_Const} is related to the cost of signature analyzer, and the dynamic component is determined by the number of test operands needed for the functional test [10]. It is easy to find also the cost C_{D_Const} of the additional needed hardware.

The main difficulty is related to finding the curve for the dynamic part of the deterministic test costs C_{D_T} and C_{D_M}. To find these costs we need to generate for each point on the horizontal axis the number of needed deterministic test patterns which will take a lot of time.

The creative part of the solution is in finding the answers to the question how to find the minimum of C_{TOTAL} without having the total curve of $\alpha C_{D\_T} + \beta C_{D\_M}$. In other words, the solution should be found with as few as possible test generation experiments, i.e. with as few as possible values of $\alpha C_{D\_T} + \beta C_{D\_M}$, approaching step by step to the real optimum.

The hybrid BIST can be further improved by using deterministic test patterns as multiple seeds for test generator. Fig. 6 illustrates the usual situation where "hard

to test" faults have spread over the whole range of test patterns in such a way that a single long pseudorandom test sequence is not able to cover all the unique patterns needed for detecting the "hard" faults.

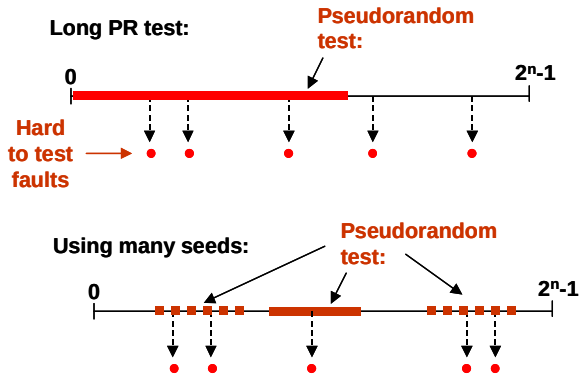

FIGURE 6
DETECTING HARD TO TEST FAULTS

Fig. 6 shows also a strategy how all the unique test patterns can be covered by multiple shorter sequences, where each of them starts with a seed (test pattern) found by deterministic ATPG. The task of the student is to find a clever trade-off between the total test length and memory space, needed for storing seeds, to minimize the total cost according to the curves in Fig. 5. Another task could be to minimize the test length at the given restrictions to the memory cost or power consumption.

PRACTICAL EXPERIENCES

We have used the described laboratory environment [7], [9] at our institute for teaching design and test during recent years, and the feedback from students has been excellent. Three types of study have been dominating: traditional laboratory hands-on training with instructor for bachelors, autonomous research oriented course work over the whole semester for masters, and PhD research.

The master students have evaluated very high the research oriented course work which consists usually of the following tasks: design of a circuit to implement in hardware the given functionality, experimental testability analysis of the design, redesign for testability if needed, inserting into the design one of the preferred BIST solutions, and evaluating the quality of the chosen BIST.

In the first step, the students acquire proper hands-on experience in using commercial CAD tools for designing the given circuit. Thereafter, to perform all the testability analysis, test generation, fault simulation and experimental research with BIST solutions, they use the possibilities of the described design and test environment [7], [9]. Many of the devoted students have expressed that this course work has been one of the most interesting tasks in their whole course track because they have had a chance to learn a lot of CAD and test tools, and to conduct interesting research. This environment has been partially used also in other universities in Sweden and Germany.

CONCLUSIONS

A conception is presented to improve 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 low-cost university tools dedicated for fault simulation and test generation. The free-access basis and self-contained nature makes it easy for students also from other universities to use this system independently of time and place, and learn individually according to their own needs. The tasks chosen for hands-on training represent simultaneously real research problems, which allow fostering 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.

ACKNOWLEDGMENT

This work has been supported by the grants of Estonian Science Foundation G5910 and G7068, EC Framework V project VERTIGO, and Enterprise Estonia ELIKO project.

REFERENCES

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

[2] Tenhunen, H, "*Invited talk:* System-on-Chip Curriculum Challenges", *Proc 5th European Workshop on Microelectronics Education – EWME*, Lausanne Switzerland, April 15-16, 2003.

[3] Prinetto, P, et al., "Panel Session: Design & Test Education and Training in Europe", *7th IEEE Workshop on Design and Diagnostics of Electronic Circuits and Systems – DDECS*, Stara Lesna, Slovakia, April 18-21, 2004.

[4] Ubar,R, Wuttke H-D,."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, Oct. 10-13, 2001, Reno, NV, USA, pp.SIE-2-7.

[5] Wang, L-T, Wu, Ch-W, Wen, X, "VLSI Test Principles and Architectures", *Elsevier*, 2006, 777 p.

[6] Armstrong, J, R, Gray F, G, "Structured logic design with VHDL.", *Prentice-Hall, Englewood Cliffs*, 1993, 482 p.

[7] http://www.pld.ttu.ee/applets/rtl/

[8] Devadze,S, Jutman, A, Sudnitson,A, Ubar,R, "Web-based training system for teaching basics of RT-level Digital Design, Test, and Design for Test," Proc. of 9th International Conference on Mixed Design of Integrated Circuits and Systems - MIXDES 2002, Wroclaw, Poland, June 20-22, 2002, pp. 699-704.

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

[10] Ubar, R, Mazurova, N, Smahtina, J, Orasson, E, Raik, J, "HyFBIST: Hybrid Functional Built-In Self-Test in Microprogrammed Data-Paths of Digital Systems", Int. Conference MIXDES, Szczecin, June 24-26, 2004, pp.497-502.

[11] Ubar,R, Shchenova,T, Jervan,G, Peng,Z>, "Energy Minimization for Hybrid BIST in a System-on-Chip Test Environment", Proc. of the 10th IEEE European Test Symposium, May 22-25, 2005, Tallinn, pp.2-7.