

E-LEARNING TOOLS FOR DIGITAL TEST

S. Devadze, R. Gorjachev, A. Jutman, E. Orasson, V. Rosin, R. Ubar

Tallinn Technical University, Tallinn, Estonia

Abstract– This paper describes tools used in an e-learning environment developed at Tallinn Technical University in the frames of the project REASON. The environment makes use of a diagnostic software package called Turbo Tester and a set of Java applets. It is aimed at teaching basics as well as advanced topics from the area of digital testing and diagnostics of integrated circuits.

1. Introduction

In this paper, we present an overview of latest developments taking place at Tallinn Technical University (TTU) in the area of e-learning supported by European project REASON (REsearch And Training Action for System On Chip Design) [11]. Created environment is the result of close cooperation between TTU, TUI (Technical University of Ilmenau, Germany), WUT (Warsaw University of Technology, Poland), IISAS (Institute of Informatics of the Slovak Academy of Science), and other REASON partners.

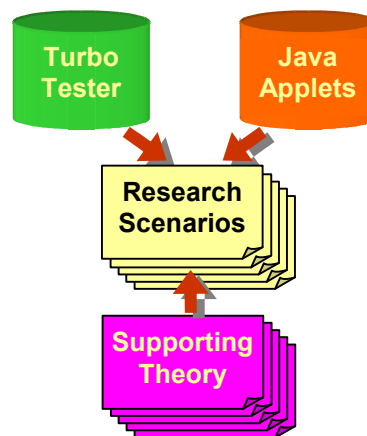


Fig. 1 Overview of the e-learning environment

The environment consists of several interrelated modules shown in Fig. 1. The PC-based tools installed locally and Java applets invoked remotely via Internet supplement each other and form the engine of the whole concept. The PC-based tool set is called Turbo Tester (TT) [1,9,10] and consists of the following main tools: test generators based on various algorithms, logic and fault simulators, a test optimizer, a module for hazard analysis, a simulator and test generator for defects, built-in self-test simulators, design verification and design error diagnosis tools. This range of compatible diagnostic tools forms, via their interaction and complementary operation, a homogeneous research environment, which provides good possibilities for laboratory training and experimental research.

The general idea behind the Java applets is a bit different. They mainly aimed at supporting the concept of game-like style of learning via easy action and reaction, learning by doing, and concentration on most important topics in the simplest possible way. There are three applets available by now [12]. They are: “Applet on Basics of Test & Diagnostics”, “Applet on RT-Level Design and Test”, and “Applet on Boundary Scan Standard”. The fourth

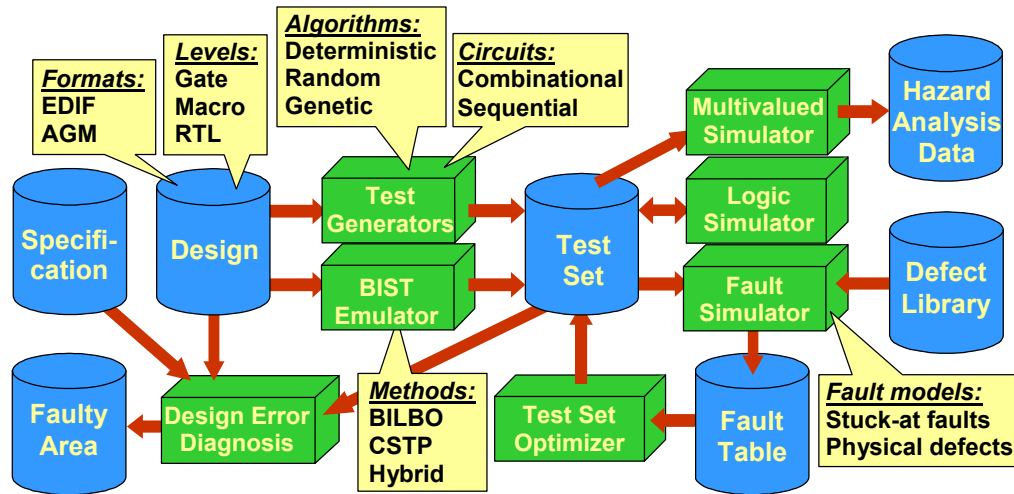


Fig. 2 Overview of Turbo Tester package

applet, which represents a simple schematic and decision diagram (DD) editor is currently under development.

The central point of the presented concept is research scenarios – a set of problems and experiments that represent virtual laboratory works, where students learn diagnostic software and get acquainted with common concepts and problems of testing and diagnostics. There are scenarios for beginners and for advanced study. The user-friendly graphical environment created by the applets is the best option for beginners. A bit more advanced study is possible with scenarios, which make use of TT tools. The full text of the scenarios is available in the Web [13].

In the next section we present an overview of the Turbo Tester package and its main tools. It is followed by Section 3, which describes Java applets. Conclusions are given in Section 4.

2. Overview of Turbo Tester Package

There is a number of scientific papers describing research carried out using Turbo Tester that have been published in international conferences as well as reviewed journals [2,3,4,5,6,7,8]. This became possible due to a homogeneous environment of TT (Fig. 2) where different methods and algorithms for various test problems are implemented and can be investigated as separately of each other as working together in different combinations. The latter provides a variety of tasks optimization of a given particular problem.

Model Synthesis. The component library of Turbo Tester consists of Binary Decision Diagram (BDD) representations for the library components of the circuits to be processed. The model generator creates a BDD-representation of the design from the netlist of the design, produced by e.g. schematic editor. The special kind of BDDs is used in Turbo Tester. They are called Structurally Synthesized BDDs (SSBDD) and provide a uniform approach to solving a wide scale of testing tasks, based on a uniform model and a restricted set of standard procedures [5].

Test Generation. For automatic test pattern generation (ATPG), there are random, deterministic and genetic test pattern generators (TPG) implemented [3]. Mixed TPG strategies based on different methods can also be investigated. Tests can be generated for both, combinational and sequential circuits. Stuck-at faults and physical defects can be considered.

Test Pattern Analysis. There are single-fault simulation, parallel fault simulation, and critical path tracing fault analysis methods implemented in the system. These competing approaches can be investigated and compared for circuits of different complexities and

structures. As the result of using these tools, fault tables are calculated and test quality is evaluated for given test sequences. In a defect-oriented simulation mode the fault simulator uses a special defect library [2]. The physical defect model includes short (bridging) faults and will be soon extended by open faults.

Test Set Optimization. The tool minimizes the number of test patterns in the test set by means of static compaction. The technique implements effective representation of fault matrices by weighted bipartite graphs. The proposed method offers significantly fast performance in terms of run times [7].

Multi-valued Simulation. In Turbo Tester, a multi-valued simulation is applied to model possible hazards that can occur in logic circuits. The dynamic behavior of a logic network during one single transition period can be describes by a number of symbols of some given alphabet corresponding to representative waveforms on the output. The multi-valued simulator of TT implements 5-valued and 8-valued alphabets [8].

Design Error Diagnosis. After a digital system has been designed according to specifications, it might go through a refinement process in order to be consistent with certain design requirements (e.g. timing specifications). The changes introduced by this process (by a human or a CAD system) may lead to undesired functional inconsistencies compared to the original design. Such design errors should be identified via design verification. A design error diagnosis technique should be applied afterwards in order to locate and correct the error. In Turbo Tester we implemented an approach, which advantage consists in usage of a common fault-detection test instead of a diagnostic test [6].

Testability Analysis. The total product cost can be minimized by regarding the design and test of a product as one integral activity rather than the two disjoint unrelated activities. Such an approach is called design for testability (DFT). The testability analysis tools of the system can be used for enumerating untestable faults, for selecting statistically hard-to-test faults, and for estimating the controllability, observability and testability characteristics for the nodes of the design. These tools are used for finding out where to alter the design to improve the testability.

Evaluation of Built-In Self Test (BIST) Quality. The BIST approach is represented by applications for Built-In Logic Block Observer (BILBO) and Circular Self-Test Path (CSTP) emulation. Different BIST architectures can be simulated and the self-test quality of these architectures can be evaluated. There is a tool, which utilizes a genetic search algorithm for automatically finding good BIST architectures. It is possible to use also a "reseeding" approach to BIST optimization. A Hybrid BIST is another approach of combination pseudo-random patters with deterministic ones making it possible to achieve higher fault coverage by shorter test sequence [4].

Design Interface and Portability. Via powerful EDIF 2.0.0 design interface, TT can read schematic entries of various contemporary VLSI CAD tools, e.g. Cadence, Synopsys, Mentor Graphics, Viewlogic, Compass, OrCAD, etc. which makes TT independent of a particular design environment. There are Turbo Tester versions available for MS Windows, Linux, and Solaris operating systems. The software is free of charge and it can be downloaded together with the user's manual from the Web [10].

3. Overview of Java Applets

There are three applets already available in the Web [12] and one more is currently under development. Figure 3 represents an overall structure of relations between the applets, Turbo Tester, and the research scenarios.

We have selected the Java technology for our interactive teaching system since it is well supported by main operating systems like Windows, Linux, and Solaris. The applets are

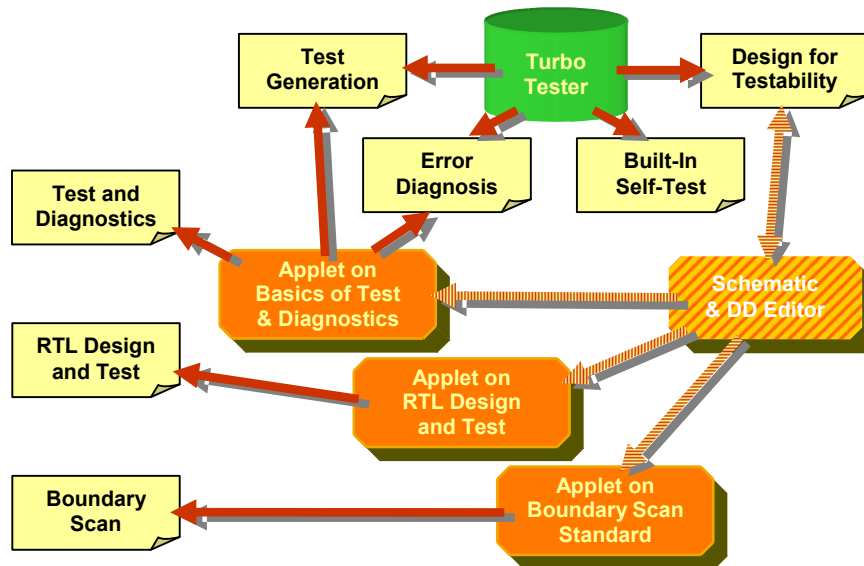


Fig. 3 Relationship between the applets, TT, and research scenarios

available in the Internet free of charge. The latter makes it easy for students from universities all over the globe to use this system at any location.

Java applet on basics of test & diagnostics. Main principles of logic-level test generation and fault diagnosis are the fundamentals of the modern VLSI CAD and diagnostic software. Good understanding of these basics gives students vital background and skills in their future profession. The applet provides with possibility of manual and pseudo-random test pattern generation (TPG), fault simulation, combinational and sequential fault diagnosis, and investigation of BIST techniques. All the designs used in the applet are combinational circuits of a rather small size represented and visualized on the logic level.

In the test generation mode one chooses a target fault and step by step activates needed paths in the circuit in order to detect the fault. Sensitization of the fault and propagation of the fault effect towards the output are carried out by clicking the corresponding lines and

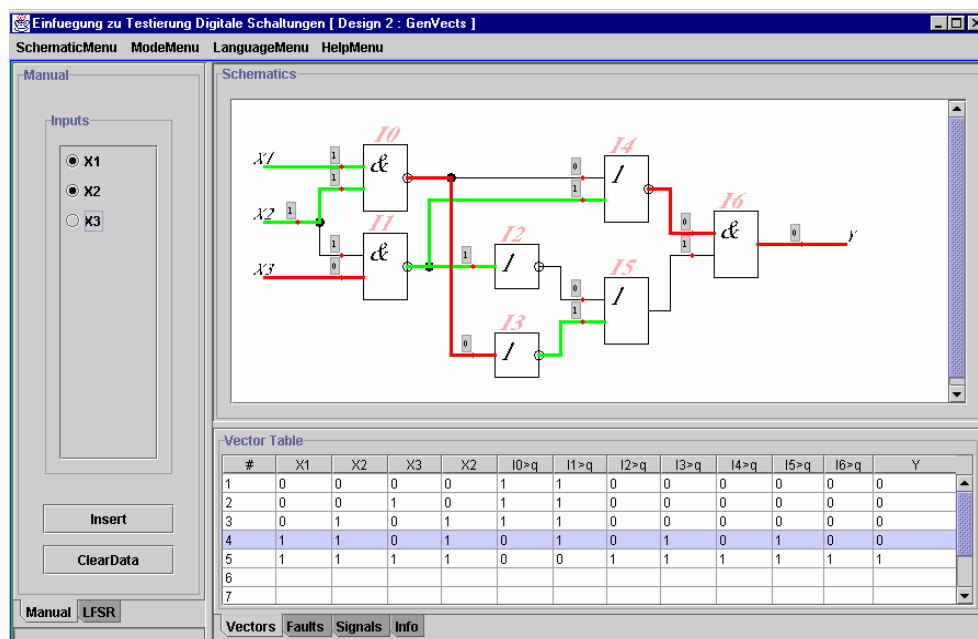


Fig. 4 Java applet on basics of test & diagnostics

selecting needed values. At the same time, the lines (wires) change their color helping the student in selection of proper values.

If the specific fault detection is not of a primary interest, the user can specify the input values only. In this way one can set up as many vectors as he wants. The pseudo-random TPG is also implemented. It provides two modes: BILBO and CSTP.

All the vectors can be then simulated in the fault simulation mode. The results of simulation are represented in the form of fault table. By selecting a vector in the table, all the faults detected by this vector will also be highlighted on the design schematic. The same fault table is used for the combinational fault diagnosis as well. In this mode, a subset of vectors is selected and applied to the erroneous circuit (imitating test experiments). The applet shows the results of fault diagnosis by highlighting the faulty area. If the results are not satisfactory, additional vectors could be generated.

Another supported method of fault diagnosis, the sequential diagnosis is based on the guided probing strategy. In this mode, students should sample the values by clicking signal lines and comparing observed values and correct ones. The goal is to find the precise fault location using as few samples as possible.

Java applet on RT-level design and test. In this applet, we combine and illustrate many different problems related to RT-level control intensive digital design, test, and design for test. Therefore, the applet gives a unique possibility to teach these topics in a consecutive iterative approach. The range of considered problems includes:

- design of a data path and control path (microprogram) on RT level
- investigation of tradeoffs between speed of the system & HW cost
- RT-level simulation and validation
- gate-level deterministic test generation and functional testing
- fault simulation
- logic and circular BIST, functional BIST, etc.
- design for testability

The applet provides the representation of the target system on RT level as divided into *datapath* and *control unit*. The structure of datapath is shown schematically, while the *control*

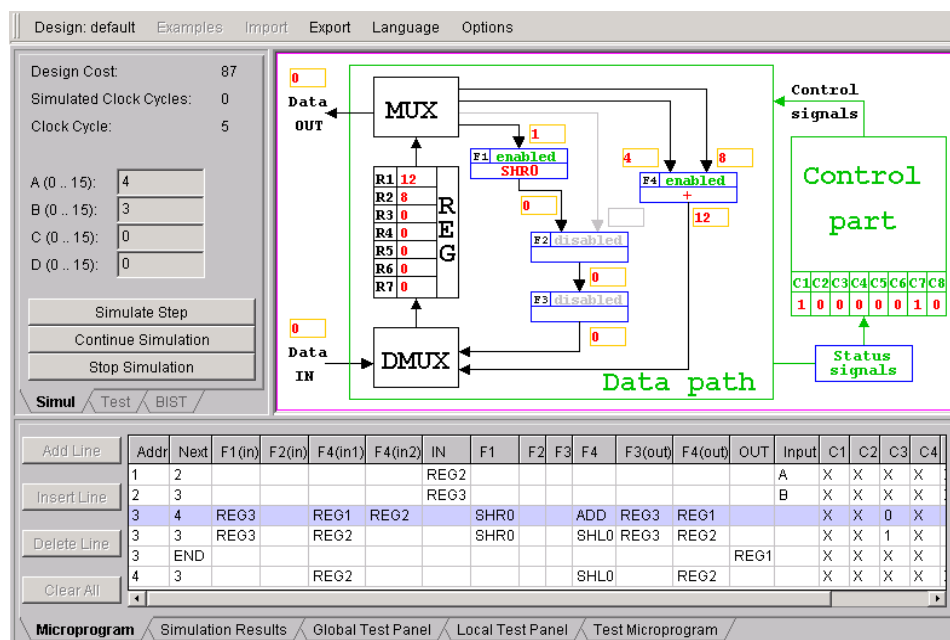


Fig. 5 Java applet on RT-level design and test

part of the system is defined by the *Microprogram table*. The RT-level fault and fault-free simulation can be performed for a single set of input data as well as for all the sequence of input operands at once. The fault simulation is performed for the datapath and its units but not for the controller. During the simulation the active line of the microprogram is highlighted. The simulation data is also reflected in a graphical way right at the datapath. There is a manual test patterns generation mode for a selected microoperation separately. The gate-level representation of this microoperation is shown at that time. The test access to the selected unit is provided by a special *Test Microporogram*. There are various possibilities to experiment with BIST provided in the *BIST module*. The user can select locations of pseudo-random TPGs and signature analyzers within the data path as well as their configurations.

The applet has a flexible design. The RT-level system model is described in a form of text-files. Hence, any custom design can be described and loaded just as simple as the original ones. The applet has a built-in extendable collection of examples implementing different algorithms. For connecting the applet to other applications as well as for providing users with a possibility to save the results of their work, the applet has a data import/export capability. It also has a built-in multilingual support.

Java applet on Boundary Scan (BS) standard IEEE 1149.1. There are two different modes of BS simulation. The first one, the *TAP Controller Mode*, provides a very detailed illustration of operation of BS registers and the TAP controller. This mode is intended for the beginners and for teachers. It helps to understand all the needed basics.

Another mode, the *Command Mode*, can be used for faster simulation of BS commands like EXTEST, SAMPLE/PRELOAD, etc. with different predefined input data. This mode is also useful for *fault diagnosis*. The applet supports possibility of random or specific fault insertion. The operation of the faulty device can be then simulated and the fault can be diagnosed.

Our applet is provided with lots of built-in examples. Nevertheless, we decided to allow users to generate their own examples by creating fully custom chips or boards. In the chip editing mode, the applet reads the description of BS structures using BSDL (Boundary Scan Description Language) format which is a part of the BS standard. Such BSDL descriptions are

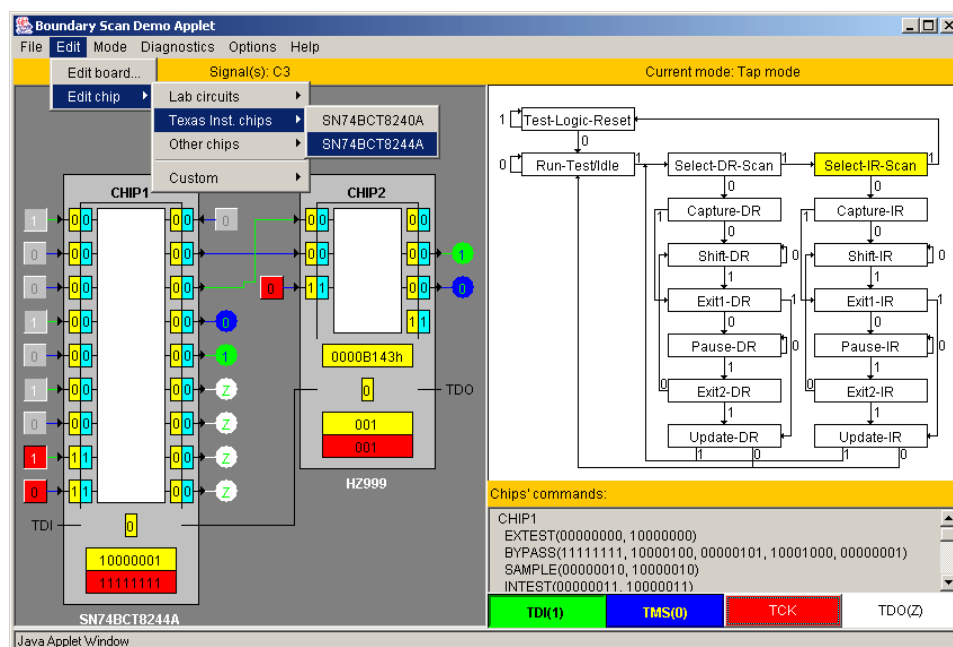


Fig. 6 Java applet on Boundary Scan standard

usually free of charge and widely available via Internet, which makes the work with the applet easier and more exciting, since the student can visualize behaviour of many different chips available in the market.

Currently, the chip's internal logic can only be described by SSBDD format, which is obtained by Turbo Tester from commonly used EDIF format. However, this format is optional since the internal structure of the chip can be neglected when simulating most of BS modes.

Acknowledgements

This work was supported partly by the EU Framework V project REASON, by the Thuringian Ministry of Science, Research and Art (Germany), and by the Estonian Science Foundation Grant No 5649.

4. Conclusions

In this paper we have described tools used in the e-learning environment developed at Tallinn Technical University. This work is still in progress. All the components of the environment are available via the Web free of charge.

The core of the system is a set of research scenarios or virtual laboratory works based on two types of software: *a)* diagnostic package Turbo Tester and *b)* set of Java applets. While the TT should be installed on a local computer, the applets are invoked remotely via Internet. The Turbo Tester provides a homogeneous research environment, which allows for interesting experimental research to be conducted. The Java applets, in their turn, provide an attractive game-like milieu, which is important especially for beginners.

Due to the facts above, the conception presented here is suitable for a broad audience of learners who are interested in studying different concepts of testing and diagnostics of integrated digital circuits.

References

- [1] M.Aarna, E.Ivask, A.Jutman, E.Orasson, J.Raik, R.Ubar, V.Vislogubov, H.-D.Wuttke. "Turbo Tester - Diagnostic Package for Research and Training," in *Proc of East-West Design and Test Conf.* Alushta, Ukraine, Sept. 17-21, 2003.
- [2] M. Blyzniuk, FT. Cibakova, E. Gramatova, W. Kuzmicz, M. Lobur, W. Pleskacz, J. Raik, R. Ubar. "Hierarchical Defect-Oriented Fault Simulation for Digital Circuits," *IEEE European Test Workshop*, Cascais, Portugal, Mai 23-26, 2000, pp.151-156.
- [3] E. Ivask, J. Raik, R. Ubar. "Comparison of Genetic and Random Techniques for Test Pattern Generation," *Proc. of the 6th Baltic Electronics Conference*, Oct. 7-9, 1998, Tallinn, pp. 163-166.
- [4] G. Jervan, Z. Peng, R. Ubar. "Test Cost Minimization for Hybrid BIST," *IEEE Int. Symp. on Defect and Fault Tolerance in VLSI Systems*. Tokio, October 25-28, 2000, pp.283-291.
- [5] A. Jutman, J. Raik, R. Ubar, "SSBDDs: Advantageous Model and Efficient Algorithms for Digital Circuit Modeling, Simulation & Test," in *Proc. of 5th International Workshop on Boolean Problems (IWSBP'02)*, Freiberg, Germany, Sept. 19-20, 2002, pp. 157-166.
- [6] A. Jutman, R. Ubar, "Design Error Diagnosis in Digital Circuits with Stuck-at Fault Model," *Journal of Microelectronics Reliability*. Pergamon Press, Vol. 40, No 2, 2000, pp.307-320.
- [7] A. Markus, J. Raik, R. Ubar. "Fast and Efficient Static Compaction of Test Sequences Using Bipartite Graph Representation," *Proc. of the Second Electronic Circuits and Systems Conference ECS'99*, pp. 17-20, Bratislava, Slovakia, Sept. 6-8, 1999.
- [8] R. Ubar. "Dynamic Analysis of Digital Circuits with Multi-Valued Simulation," *Microelectronics Journal*, Elsevier Science Ltd., Vol. 29, No. 11, Nov. 1998, pp.821-826.
- [9] *Turbo Tester Reference Manual*, Version 02.10, Tallinn TU, Estonia, October 2002. Available at [11].
- [10] Turbo Tester home page URL: <http://www.pld.ttu.ee/tt>
- [11] REASON project home page: <http://reason.imio.pw.edu.pl>
- [12] Java applets home page: <http://www.pld.ttu.ee/applets>
- [13] Laboratory training URL: <http://www.pld.ttu.ee/diagnostika/labs>