AC 2011-1218: DEVELOPING AND IMPLEMENTING DIGITAL SYSTEMS TESTING COURSE FOR ENGINEERING AND TECHNOLOGY CURRICULUM

Muhammad Zafrul Hasan, Texas A&M University

Muhammad Zafrul Hasan received the B.Sc. in Electrical and Electronic Engineering from Bangladesh University of Engineering and Technology. He received the Master of Electronic Engineering from Eindhoven University of Technology (The Netherlands) under a Philips postgraduate scholarship program. He subsequently held several faculty positions in an engineering college and in a university in Malaysia. He obtained the Ph.D. in Computer Engineering from New Jersey Institute of Technology. He was awarded the NJIT Hashimoto Fellowship in the academic year 2005-06. He is currently an Assistant Professor of Engineering Technology and Industrial Distribution at TAMU. His research interests include the design, implementation, and testing of dynamically reconfigurable computing systems, performance evaluation of computer architectures, and behavioral synthesis and testing of digital systems, energy efficient embedded systems.

Developing and Implementing Digital Systems Testing Course for Engineering and Technology Curriculum

Abstract

Digital systems have penetrated in all aspects of everyday life – starting from household ovens to professional supercomputers. As such, curriculum in electrical, computer, and telecommunication engineering disciplines incorporate the basics of digital systems as a mandatory course. Such courses traditionally contain the design aspects of digital systems. Testing of such systems is seldom covered in those courses [1, 2]. However, testing of such rapidly growing systems is both complex and costly [3]. As a result, it has become an important part of the overall life cycle of any digital system. Due to its importance and significance in the real world, digital system testing needs to be accommodated in the curriculum of the above mentioned disciplines. Testing of combinational and sequential circuits, scan testing, Boundary Scan Testing (BST) of integrated circuits and their assembly are quite relevant in this respect. Also, Built-In Self Test (BIST) of complex systems, memory and processor testing and testing of cellular arrays are important. This paper presents an outline of such a course with laboratory activities and assessment strategies. Also, it assesses the performance of students in related courses in order to justify the appropriateness of such course in the above mentioned engineering disciplines.

Introduction

Motivation and rationale: Digital systems are built from simple logic gates of assorted types. Building and testing such gates in isolation is a trivial task for electrical, computer, and telecommunication engineering students. However, once they are combined to form a complex function, the above tasks are no longer trivial. Moreover, when several such functions are combined in an integrated circuit (IC), the task of testing becomes huge. And finally, when ICs of different kind are combined to form a system level board, the task becomes enormous and challenging. At the same time, it is important to test each individual system to ensure sufficient level of reliability. These two factors contribute to a high level of cost for digital system testing. It is shown in [3] that the cost of testing increases linearly up to a certain level of integration. Beyond that point, it increases almost exponentially as shown in Figure 1. This is also supported by the fact that it is ten times more expensive to detect a fault at each next step as it migrates from the manufacturer to different vendors and finally to the end user [3]. As such, there needs to be a systematic approach to testing in order to restrain the overall cost. This demands a careful study of probable faults in digital systems and the techniques to detect them efficiently. Students in the related engineering disciplines need to be educated accordingly.

Limitation of traditional courses: Most of the traditional courses deal with the design elements of digital systems. It covers topics such as combinatorial logic design and Boolean algebra, sequential logic and state machine design. Although it prepares the graduates with the skills of designing complex systems, they lack the skills of testing them effectively. Moreover, the designers may not be aware of the challenges of testing their design when it is used in

conjunction with other devices. These issues are not addressed in most contemporary courses. This trend of overlooking the test activity in colleges is also reflected by the limited contribution of papers from the academia in recent test related conference [4]. A case of test problem for a complete processor is worth mentioning here. At Texas A&M University, engineering technology students design a complete 8-bit processor with its datapath, arithmetic logic unit

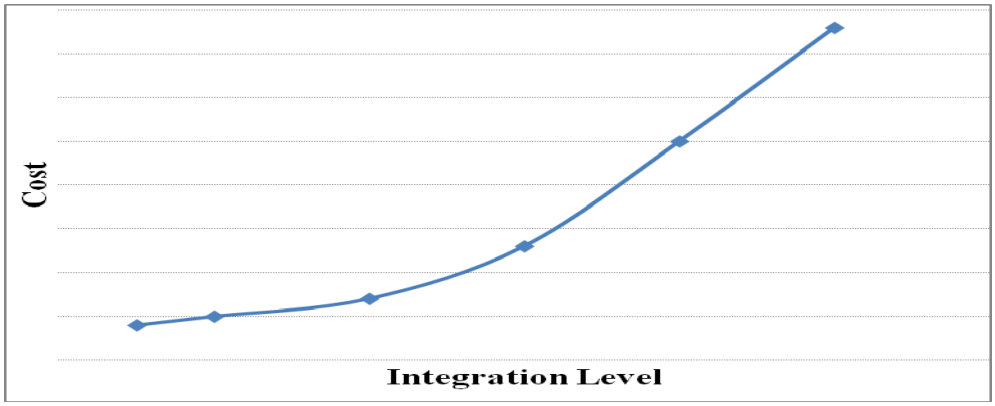

Figure 1: Trends in Test Cost

(ALU) and the control sections [5]. Datapath is built from a set of registers. As they all are connected by a shared bus, the resulting bus-contention needs to be detected and resolved before it can be connected to the control unit. Observing certain lower level signals at the system output in such a hierarchical design was tedious for students. This could have been solved either by testing the lower level modules individually or keeping some provision of test at the onset of design.

Way to overcome: The above mentioned shortcomings of the traditional Digital Systems courses could be overcome by introducing test relevant topics in lectures and associated activities in their laboratory. These topics could be injected in existing courses by creating a balance between design and test. Alternatively, a new course could be introduced altogether with only the relevant test related topics. This would prepare the engineering graduates to consider the challenge of testing from the very beginning of design process of a system. With this knowledge and education, the graduates can incorporate design for testability (DFT) features in their designs. Also, those graduates who will not be involved in design of systems would still be able to cope up with test challenges easily with such awareness.

Lecture Contents

The set of learning outcomes for the course under discussion are as follows. At the end of the course, the students should be able to:

1) describe the importance of testing,
2) explain various fault models,
3) distinguish different fault detection techniques,
4) adopt design for testability approach, and
5) implement the above concepts in a laboratory set-up

Lecture contents can be developed as described in the following paragraphs.

Motivation for testing / Introduction: This topic is included in order to make the case of testing important. The enhanced complexity and cost of testing integrated circuits (ICs) and systems need to be highlighted. Effect of an undiagnosed fault may be huge when detected later. The case of Intel floating-point bug is an example.

Testing Combinational Logic: The stuck-at model needs to be introduced. Testing individual gate and a network of gates are to be covered. Techniques of path-sensitizing and fault propagation are to be explored as well [5].

Testing Sequential Logic: Testing of such devices by observing only the system outputs is to be discussed. Detection of stuck-at fault in such circuit is also treated here. Limitations of these approaches need to be discussed.

Scan Testing: Here, the comprehensive testing of sequential logic by observing its internal states is discussed. It covers the technique of connecting all internal test points into a serial chain for efficient observation of internal conditions.

Boundary Scan Testing (BST): This is a more comprehensive topic that covers testing at a hierarchical fashion. The features provided by BST for testing both individual components as well as an assembly are to be highlighted. The associated IEEE standard for BST needs to be emphasized.

Built-in Self Test (BIST): In order to minimize the involvement of expensive testers, many present day components contain self-testing circuitry within themselves. These circuits are capable of generating and applying test vectors to the functional components as well as capturing the responses. A generic BIST arrangement is shown in Figure 2.

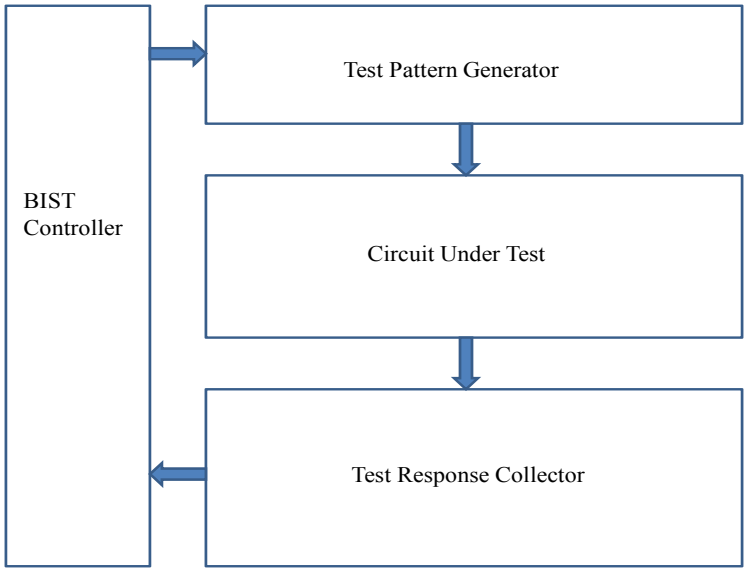

Figure 2: Generalized arrangement for BIST

Multiple Input Signature Register (MISR): It is a type of BIST suited for testing memories. They have the capability to compress the responses and compare that with expected results. A representative arrangement for MISR as applied to RAM is shown in Figure 3. For ROM, the arrangement is simpler with no data-generator.

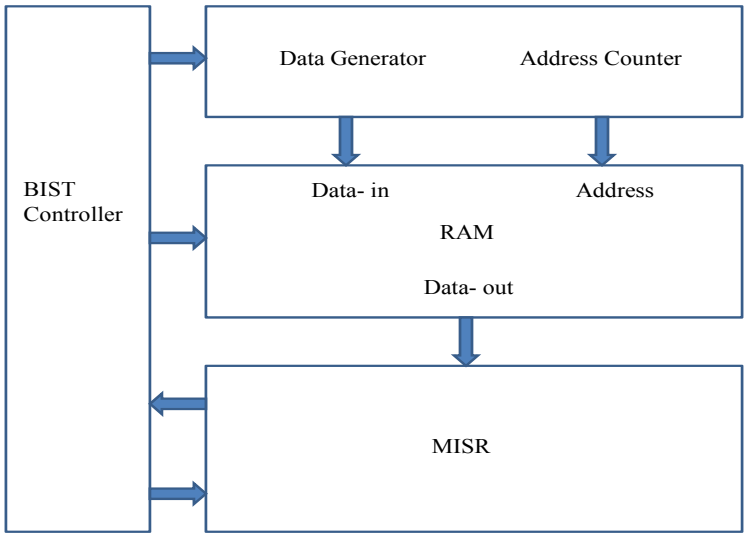

Figure 3: RAM test using MISR

Liner Feedback Shift Register (LFSR): It is often used to generate test patterns as well as to compress test outputs into signatures. It has a serial input that is derived from the outputs of some stages by passing them through an X-OR gate as shown in Figure 4. The generated outputs are pseudo random in nature. LFSR can be used to build MISR as well.

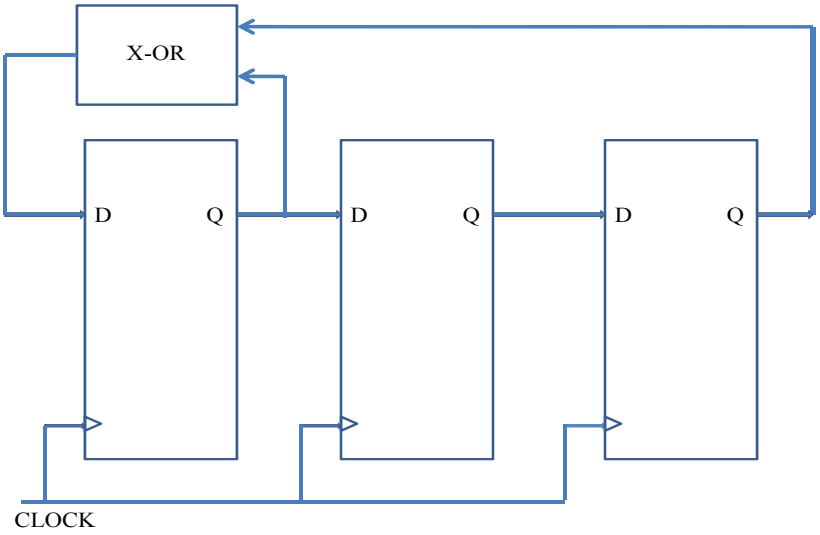

Figure 4: A simple LFSR

Built-in Logic Block Observer (BILBO): These are registers placed in between logic blocks. They can be used as pattern generators (PG) or signature registers (SR) as needed. Their roles are often interchanged in order to test different logic blocks. Figures 5 and 6 depict the concept [5].

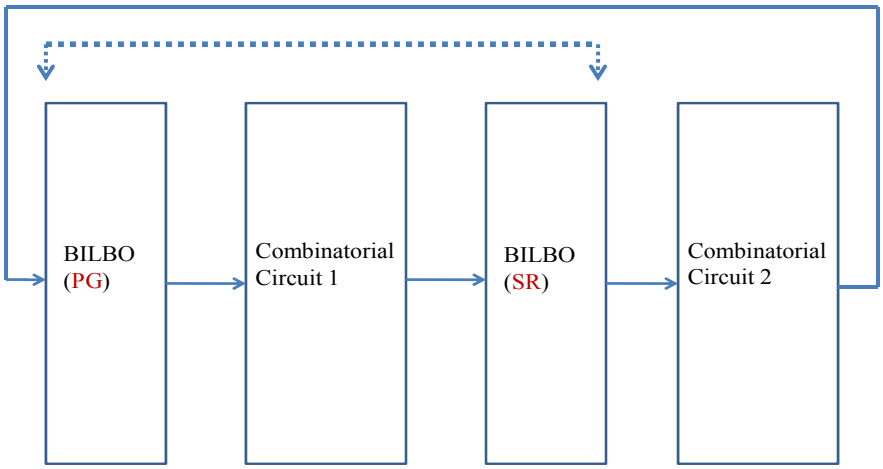

Figure 5: BILBO for testing the first block

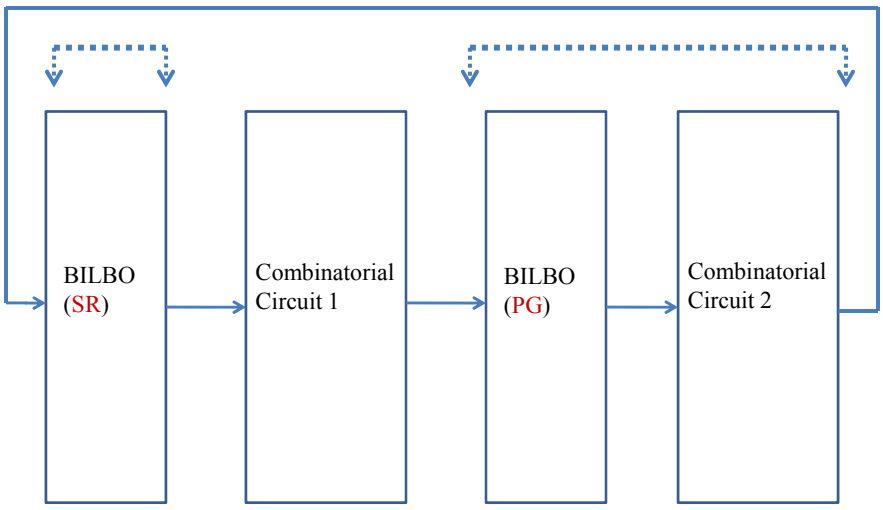

Figure 6: BILBO for testing the second block

Self Testing using MISR and Parallel SRSG (STUMPS): The technique is used for testing multi-chip modules. There are several scan chains fed with test patterns. These patterns are applied to the devices under test (DUT) in normal operation mode for one cycle. Then the responses are captured in scan chains and shifted out to the response analyzer. Figure 7 illustrates the idea of STUMPS.

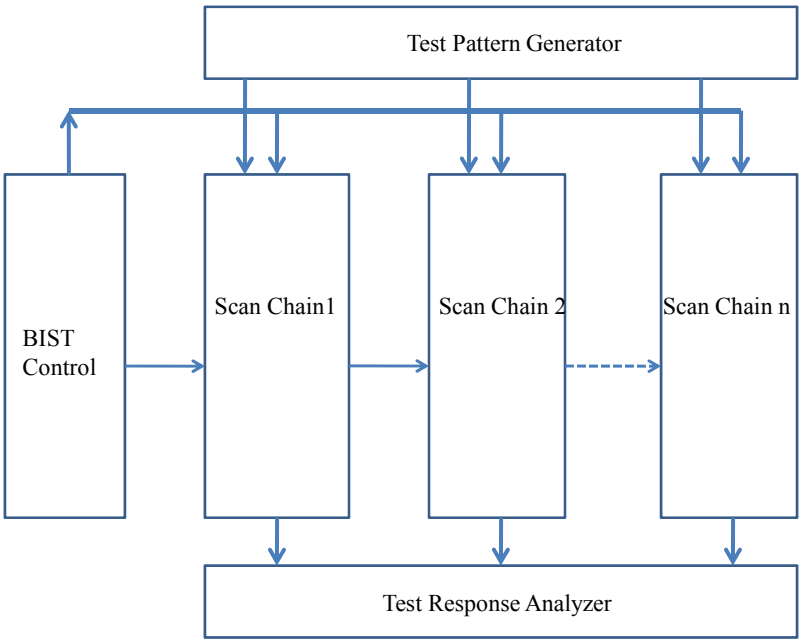

Figure 7: STUMPS for testing multiple-chip devices

RAM Testing: As they contain random values, testing a RAM involves writing and reading values as well as comparing them for diagnosis. Objective here is to ensure that each cell can store '0' and '1' correctly. Some of the off-line test patterns used are marching, walking and diagonal values. On-line RAM test includes use of error detection / correction code such as Hamming code. Some standard BIST techniques are also used in testing embedded RAM.

Testing Microprocessors: As these devices are complex, the stuck-at model is not employed as done at the gate-level. Instead, functional testing is more appropriate for processors. Some of the tests could be program counter test, stack pointer and index register test, ALU test. Having some accessibility for test purposes, the bus could be isolated and tested for stuck-at fault. Similarly, other components could be isolated and tested by means of its' own test strategy. Finally, the total system could be tested using a subset of the available instructions and known responses. One such subset of instructions could be used to cover data transfer from all sources to all destinations [6].

Fault Tolerant Processors: They employ either of the following strategies. In fail-soft operation, the faulty section can be cut or reconfigured to maintain system operation. In fail-safe operation, faulty section never gives an output that is potentially dangerous. In fault-tolerant operation, the system still produces the correct or acceptable output.

Testing Cellular Arrays: Testing these two dimensional multi input / output structures of combinational or sequential logic may involve by passing certain cells in test mode. Thus, it would allow signal flow to other cells [6].

In addition to all these, processor performance can be tested against benchmark applications. There are several benchmark application programs measuring processor performance in a particular domain. One such benchmark suite is provided by EDN Embedded Microprocessor Benchmark Consortium (EEMBC) [7]. It covers the domains of consumer, networking, and telecom among others. The concepts of benchmark testing can be introduced in the lecture. Experiments can be designed to test a processor against some or all of these domains.

Moreover, reconfigurable hardware, such as Field Programmable Gate Array (FPGA) or Complex Programmable Logic Devices (CPLD), can be used to support both normal functionality as well as testing [8]. During normal operation these devices can implement the desired functionality of the system while during testing they can be configured to support specific test requirements. These devices can be used as test generator for other neighboring component(s) or response collector from them. These responses can be analyzed to identify faults in the system. As such, the dual use of reconfigurable devices can be an interesting academic material at this level.

Laboratory Contents

The following experiments may be chosen to support the above mentioned objectives and the lecture contents. All the following experiments involve design of hardware and inserting faults in them. The hardware could be designed using VHDL and implemented on an FPGA for verification and analysis. For all the experiments, references [5, 6, 9, 10] can provide excellent background. These activities would brush-up (for students) general skill of hardware design using hardware description language and would provide additional skills of fault simulation and fault detection.

● Inserting and detecting stuck-at (0 / 1) faults at the inputs / output for: AND, OR, XOR, NAND, NOR, XNOR gates. These could be done by connecting one (or more) inputs or the output to a fixed logic. Fixing the logic of a signal is easily achieved in VHDL by assigning an appropriate value to it. This saves a lot of wiring as compared to a traditional bread-board approach. With such fault inserted, the students need to apply possible inputs and observe the output to identify the fault in the gate.

● Detecting stuck-at (0 / 1) faults in combinational logic by fault propagation using path sensitization: In this case, a network of gates is formed and stuck-at fault is inserted into it. Although the fault could be present at the inputs / outputs, an intermediate location is preferred. Students need to find out a path from the point of consideration to the output to propagate the value to be observed. This path needs to be sensitized in order to propagate the value of interest unchanged. By investigating these values, students are expected to detect the fault. The concept of 'controllable inputs and observable outputs' is manifested here.

● Transition verification of sequential circuit: It is carried out by observing only the outputs of strongly connected state graphs (SCSG). SCSG represents machines in which any state can be reached from any other state. For such machines, students are expected to generate distinguishing sequences. These sequences, when applied to the machine, are capable of generating different outputs for different states. Test of sequential circuit for stuck-at faults can also be undertaken as a laboratory exercise following an approach similar to that of combinatorial logic.

● Scan path testing of sequential circuit: The purpose of scan testing is to observe the state of internal Flip-Flops that are inaccessible normally. These flip flops are connected in a serial chain and the output is made observable. During normal operation, the chain is disabled. Students need to form such a chain in a state machine and have a control signal to switch it into the test mode. An intentional fault could be introduced in the machine and the students need to detect it by observing the output of the scan path.

● Boundary Scan Test (BST): This is a standard developed from the concept of scan path testing [11]. The objective of this exercise would be to incorporate such standard in an existing design. Test Access Port (TAP) needs to be defined, all the registers at the inputs / outputs need to placed in the boundary scan chain, TAP controller state machine needs to be designed following the standard, BST instructions and their usage need to be exercised. Time and facility permit, Boundary Scan Description Language (BSDL), board level test - interconnections between ICs can also be covered with appropriate exercises.

Evaluation Strategy

The course outlined above can carry 3-credits (2 hours Lecture and 2 hours laboratory per week). 67% of the total weight is to be evaluated by means of coursework such as homework and quizzes, one midterm, and a final examination. The breakdown of the weight is shown in the Table 1.

Table 1: Proposed Evaluation Rubric for the Course

Item	Weight
Homework 1	7
Homework 2	7
Homework 3	8
Midterm	22
Final	23
Laboratory	33

Objective of the homework and quizzes will be to test the understanding of the cumulative concepts developed at a point of time and to prepare the students for the examinations. Examinations are non-cumulative, i.e. the topics covered are exclusive for the midterm and the final. 33% of the total point is to be evaluated by means of laboratory work. It includes evaluation of performance during the laboratory sessions and the reports submitted for each group of activities. Extra credits can be assigned for guest lectures and industry visits that enhance the concepts developed throughout the semester.

Sequence of activities is shown in the following table. Students' performance is evaluated in the highlighted items.

Table 2: Sequence of activities (left to right)

Lesson	**HW1**	HW1 discussion	**Midterm Test**	Midterm Test discussion	Lesson	**HW2**	HW2 discussion	Lesson	**HW3**	HW3 discussion	**Final**
Instructor	Students Perform	Instructor	Students Perform	Instructor	Instructor	Students Perform	Instructor	Instructor	Students Perform	Instructor	Students Perform

Plan to Incorporate

One possible placement of the course could be in the junior year after Digital Logic Design and Microprocessor programming courses. However, VHDL needs to be incorporated in a prerequisite course. A course that involves VHDL programming and design implementation of an eight bit processor is already ongoing [1]. Initially, the course could be offered as an elective within the curriculum. Once the content is matured, it can be offered as a core course. Alternatively, the concepts could be partially distributed into one single existing course or be fully distributed into multiple existing courses. This course is being considered in the institution of the author. Student performance over four semesters in that course is summarized in Figure 8. As seen, the performance is skewed towards the high achievers end. As such, it is expected that a follow-on course as outlined above would be appropriate for these students having necessary background. To support the laboratory, FPGA boards need to be furnished in the laboratory. Necessary (complimentary) SW can be obtained from the vendors. The books enlisted in the bibliography could serve as the basis for developing both the lecture and the laboratory materials.

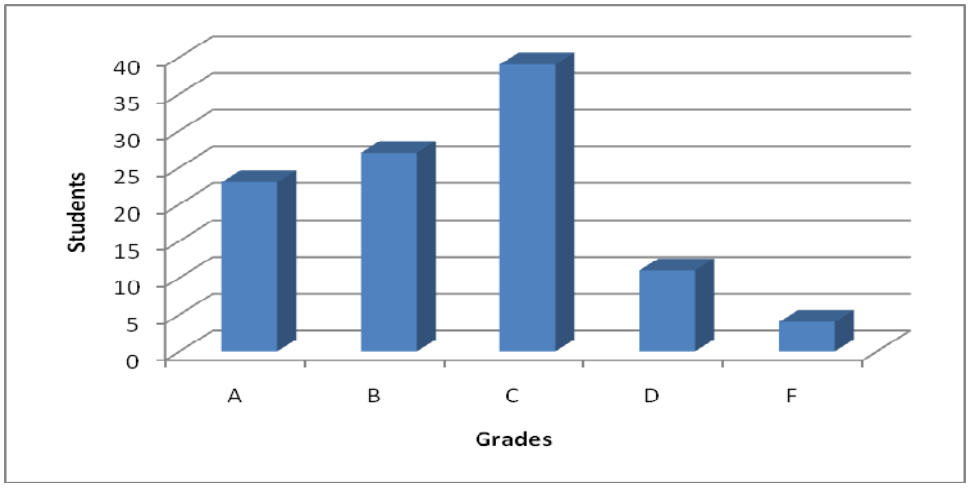

Figure 8: Summarized Student Performance in a Prerequisite Course

Conclusions

An outline of a course involving digital system testing is presented in this paper. The focus is on fault models in both combinatorial and sequential logic systems. Testing of more integrated systems like memory, PLA, and microprocessor is also covered. Significant emphasis is placed on the laboratory work that involves logic circuit design and fault simulation. Reconfigurable hardware platform is proposed as a suitable vehicle for laboratory exercises as it can be configured easily to insert fault into the system without requiring any wiring. Future improvements to this proposed course could include use of industrial testers to test off-the-shelf medium scale integration (MSI) and large scale integration LSI devices. Furthermore, automatic test pattern generation and various board-level testing can also be incorporated into the course.

Bibliography

1. Abul K. M. Azad, "Design and Development of an Introductory Digital Electronics Course within an Undergraduate Program", Journal of Engineering Technology, Spring 2008.

2. Ronald Hayne, "Beyond VHDL Simulation to on-chip Testing", ASEE Annual Conference Proceedings, June 2008.

3. Colin Maunder , "The Board Designer's Guide to Testable Logic Circuits", Addison-Wesley, 1992.

4. IEEE International Test Conference Proceedings, November 2009.

5. M. Z. Hasan, "Course Development in Digital Systems Targeting Reconfigurable Hardware," ASEE Annual Conference, June, 2009.

6. Stanley L. Hurst, "VLSI Testing : Digital and Mixed Analogue/Digital Techniques", The Institution of Engineering and Technology, 1999.

7. EEMBC, http://www.eembc.org/home.php (January, 2011).

8. S. Devadze, A. Jutman, I. Aleksejev, R. Ubar, “Fast Extended Test Access via JTAG and FPGAs,” IEEE International Test Conference, November, 2009.

9. Charles Roth and Lizy Kurian, "Digital System Design using VHDL", Thomson Publishing, 2008.

10. P. K. Lala, "Digital Circuit Testing and Testability", Prentice-Hall, 1997.

11. M.Z. Hasan and M. U. Siddiqui, “Hierarchical Test Approach using Boundary Scan Testing”, IEEE International Conference on Semiconductor Electronics, November, 2000.