Automatic Test Generation For Digital Circuits

S. Hemalatha

ABSTRACT: Current VLSI manufacturing processes suffer from larger defective parts ratio, partly due to numerous emerging defect types. While traditional fault models, such as the stuck at and transition delay fault models are still widely used, they have been shown to be inadequate to handle these new defects. The main aim is to develop a complete behavioral fault simulation and automatic test pattern generation (ATPG) system for digital circuits modeled in verilog and VHDL. An integrated Automatic Test Generation (ATG) and Automatic Test Executing/Equipment (ATE) system for complex boards is developed here. An ATG technique called Behavior-Based Automatic Test Generation technique (namely BBATG) is developed. BBATG uses the device behavior fault model and represents a circuit board as interconnection of devices. The other method used here is novel test pattern generator (TPG) for built-in self-test. A multiplexer is developed to generate a class of minimum transition sequences. The entire hardware is realized as digital logical circuits and the test results are simulated in Xilinx and Model sim software. Gate level simulation is not an effective solution for complex microcircuits. The low cost, versatile and reconfigurable FPGA-based ATE is implemented called FATE to support in ASIC development phase. The results of this research show that behavioral fault simulation will remain as a highly attractive alternative for the future generation of VLSI and system-on-chips (SoC).

Index terms: Automatic Test Equipment (ATE), BBATG method, BIST method, Test Generation, Detect Test, Diagnosis Test, FPGA, VLSI Digital Circuits.

1. INTRODUCTION

The design complexities and density of digital circuits in recent years is a growing which leads to the exponential rise in the test generation complexity and an increasing need for high quality test vectors. Circuit boards test is crucial to digital system design, and high test costs make the validation of VLSI circuits more and more critical which can be overcome by ATE System. Two main categories of ATE machines are available nowadays on the market: high-end ATE and low-cost ATE. For example Verigy , Advantest are high-end ATE and others trademarks participate in a competition in the arena of low cost ATE Inovys , Nextest High-end ATEs are characterized by high grade of automation on the other hand they are very expensive and require an accurate setup and skilled people so ASIC manufacturers also needs some other testing solution, which can be executed in house during preliminary chip evaluation phase. In this paper we propose a reconfigurable system called FATE (FPGA-based ATE). FATE allows executing home-made digital tests, only using a Laptop and a FPGA-based board and can be easily modified accordingly to the DUT (Device Under Test) saving time and money. Test pattern generation plays the major role in ATE machine. There are two types of ATG Random ATG and Deterministic ATG. Random ATG is less complex than deterministic ATG. Random ATG may be able to quickly generate tests initially, but it would be very inefficient to achieve higher fault coverage. Deterministic ATG can be applied to both combinational circuits and sequential circuits. In practice sequential ATG is in orders of magnitude more complex than combinational ATG.

- *S. Hemalatha*
- *Department of Electronics and Communication Engineering,*
- *Sri Venkateswara College of Engineering, Pennalur, Irungattukotai, Tamil Nadu, India,*
- *hemakavitha89@gmail.com, latha_h89@yahoo.co.in*

D-algorithm ,PODEM and FAN algorithm is commonly used path sensitizer. Most ATG using stuck-at at fault model can't deduce tests for complex circuit boards with LSI/VLSI Unlike most other deterministic ATG using the stuck-at fault model, Behavioral based automatic test generation(BBATG) technique is proposed which use the device behavior fault model which can completely describe functions and timing relations for a combination or sequence device, and then needn't decompose devices to gate-level descriptions. At the same time, we use the structural model to describe circuit board as an interconnection of devices

2. NEED FOR BBATG TECHNOLOGY

In this section, we described the three main components behind the BBATG technique:1) a behavior-based device model; 2) a set of reusable device test libraries; 3) an automatic test deduction algorithm.

A. Behavior-Based Device Model

In this paper, we use a device model and a circuit board model similar to the one described by Rooster [5,6].

1) Circuit board model: A circuit board is defined as a system of an interconnection of components. In such a structural model, devices are at the lowest level and consequently are considered as primitive components.
2) Device model: A device is described by input pins of a device may take "0", "1", "x" and its output pins may take "0", "1", "x" ,"u", where "x" represents arbitrary value (may be "0" or "1") and "u" represents unknown value.
3) Device Behavior fault model: A device has a behavior fault when its behavior fails. In system test deductions, we only review device behavior faults. Wire faults between individual devices in a system are attributed as faults at the connecting devices. In order to detect and diagnose fan-in/fan-out faults at system initial input/output pins, users can treat each system connector interface as one device so that wire faults are converted to device faults.

4) Behavior Error Signal*:* In BBATG, a behavior error signal is presented as an “H” or “L”. “H” represents “1” when the signal is normal and “0” when fault; “L” implies “0” when the signal is normal and “1” when fault.
5) Single Behavior Fault Assumption: During the course of an error deduction in system we assume that the behavior fault in consideration is only generated by the H/L error of the original device. The device’s other behaviors and behaviors of other devices are normal.
6) Detection Test and Diagnostic Test

We consider two kinds of test tasks: detection test and diagnostic test. A detection test is to find if a circuit has a behavior fault. A diagnostic test is to locate the behavior fault inside the circuit. For circuit boards, we will discuss both detection test and diagnostic test. A circuit test is the set of its input stimulus and its expected output response, as described in the following formula:

$$T=\{X, Z\} \quad (1)$$

T is a test, X is the test stimulus applied on the inputs, and Z is the expected output response on the outputs. If there is a fault in the circuit, the output response would change from Z to W, and

$$Z \neq W \quad (2)$$

If we find there are differences between the actual output and the expected output, we say can detect the fault in the circuit.

B. Device Behavior Libraries

For deducing tests of a circuit board on device behavior level, we have designed "Device Test Description Language(BBTDL)” and created Device Behavior Libraries. Currently, we have implemented libraries for most TTL devices and for some common LSI/VLSI. Users can add their own device test groups with the help of BBTDL.

1) A Device Error Library consists of various device error groups that provide H/L error sources for circuit board test deductions.

Each error group maintains the following data set:

(1) a pin array to represent the relationships of the pin reference names, their device pin numbers, their positions in test vectors, and its I/O type .
(2) initialization vector arrays for the device behavior tests if necessary.
(3) one or more test vector sequence for each testable device behavior.

2) A Device Transfer Library contains various device transfer groups that contain transfer functions of each device.

These TRN support H/L behavior err signals and sensitizing paths driving in circuit board test deductions. Similar to the device error group, each transfer group is described with a pseudo-C function.

C. System Test Deduction

BBATG first read system input file, and accordingly pick up device ERR groups and device TRN groups from the device test libraries. The system input file contains system wire table, system interface table, initial data, selected ERR listing, preset data, and etc. After accomplishing pre-process, BBATG will perform ERR test deductions one by one.As same as most deterministic ATG, the system test deduction by BBATG is heuristic. It includes forward err H/L driving stage and backtrack driving stage. When inconsistent justification results happen, a backdating process would be performed.

1) Forward H/L driving

Forward H/L driving starts with reading a device’s H/L ERR from a device error group. BBATG then applies device transfer groups to drive forward the H/L err signal to the system initial outputs. The H/L output from previous stage device must be driven to the corresponding inputs of the current device specified by system wire table.During the process of err H/L signal driving, whenever BBATG finds that at least one of the fan-outs arrives at the system initial output, it will conclude that forward driving has succeeded.

2) Backtrack driving

When forward H/L driving finishes, BBATG would start backtrack the values for every input pins of the ERR device and the H/L sensitizing path except arriving at the system initial inputs. BBATG reuse the device transfer groups in backtrack driving process. Selecting a TRN of the previous stage device, BBATG conversely compute the required input variable values according to the FUNC representation, and then justify it with their previous specified values. After variable values substituting and justifying, the PROC would get some ‘0’ or ‘1’ vector-bit values which must backtrack further. The process will continue until every ‘0’ / ‘1’ values arriving at the system initial inputs.

3) Backdating

On the process of forward H/L driving and backtrack driving, when inconsistent justification results happen, BBATG should select another TRN or backdate to the past device, then it can select different sensitive value or an alternative path.. If the trail fail, then the device behavior error cannot be tested in the system.

BIST Method

Built-in Self Test, or BIST, is the technique of designing additional hardware and software features into integrated circuits to allow them to perform self-testing, i.e., testing of their own operation (functionally, parametrically, or both) using their own circuits, thereby reducing dependence on an external automated test equipment (ATE). BIST is a Design-for-Testability (DFT) technique, because it makes the electrical testing of a chip easier, faster, more efficient, and less costly. The concept of BIST is applicable to just about any kind of circuit, so its implementation can vary as widely as the product diversity that it caters to. As an example, a common BIST approach for DRAM's includes the incorporation onto the chip of additional circuits for pattern generation, timing, mode selection, and go-/no-go diagnostic tests. The methods used in BIST methods are Checker board method, marching test, Pseudorandom number generator.

Marching test
The marching-test algorithm initializes the memory array to all 0's, and then scans the memory cells in ascending and descending orders. For each cell, scanning involves reading the cell for the expected value, writing the complement value, and reading it again.

Checker board method
The algorithm fills the memory array with a checkerboard pattern by writing 0's and 1's in alternate cells.

PRNG method
It is an algorithm for generating a sequence of numbers that approximates the properties of random numbers.

Basic LFSR logic using bits
A Modulo Counter and a Pseudo Random Number Generator are available in ACT gen using an LFSR architecture. The Linear Feedback Shift Register offers an efficient architecture for building very fast Modulo Counters and Pseudo Random Number Generators (PRNG). ACT gen also provides a tool to compute the nth value of the LFSR sequence should you need to decode a specific sequence number. In many cases, only a simple Modulo Counter is required to generate a stream of clock pulses. Nonlinear counters can also be used to generate memory addresses. In a FIFO, for example, the order in which the memory is accessed is irrelevant as long as the data is stored and retrieved in the same order. Such a memory addressing technique can also be used for a ROM look-up table (LUT). The LUT data is stored in ROM in a pseudo-random number (PRN) sequence using a simple software routine to pre calculate the address.

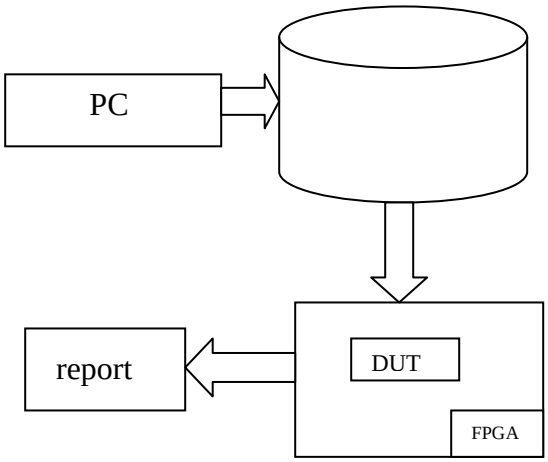

Fig. 2. Block Diagram of Proposed System

3. SIMULATION RESULTS

BBATG METHOD
The proposed system has been successfully used to test tens of prototypes of two Systems on Chip (SoC) developed to realize control systems based on Micro Electro Mechanical (MEM) sensors. The DUT that we consider is the combination of Digital Components like flip flops, mux and Full Adder. A detection test is to find if a circuit has a behavior fault. A diagnostic test is to locate the behavior fault inside the circuit. Since devices are primitive components, fault detection test is only followed. Digital part testing of this kind of ASIC is complex, so full test has been split in two complementary tests which have been separately executed: BBATG based on functional and pin timing information. BBATG is used to check DUT memory elements and is used to validate all DUT standard cells and connections. Time requested by executing aforementioned two digital tests of a single unit is valid for our application. Complete test run of these two tests, which globally consists of an amount of 4.500.000 test vectors, takes about 20 minutes. Actual FATE implementation can work at a maximum frequency of 6 MHz, (1/10 of 60 MHz on-board oscillator) which guarantees to cover the addressed sensor-conditioning applications. For example aforementioned BIST and ATPG tests can be successfully realized at 1 MHz, accordingly to the specifics ordered by designers. We used the proposed architecture on tens of these ASICs for both SoCs and this approach provided to be very useful. The Behavioral simulation result for this DUT can be given by comparing the fault free output with the faulty result. The example DUT when tested by giving some faults in the circuit such as behavioral fault and by giving some delay in the circuit the result as shown in Fig. 4 is observed. Patterns generation and verification loop is now complete, designer can launch the same simulation used in ASIC project phase with the further result of obtaining patterns. After the HDL code is simulated in software the configuration file is then transfer to the new hardware that is ready to be synthesized and fitted on FPGA. Our implementation we use a commercial development board DS-KIT-3SMB1500-EURO by Memec InsightTM, featuring the XC3S1500-4FG676C a Xilinx® SpartanTM 3 family FPGA with 1.5M system gates and 576Kbits of Memory resources. The board is connected to the DUT by means of an application specific module (InsightTM P160 prototype module) in order to allow prototyping and testing of external devices by providing them the power supply from FPGA and simplifying the physical connections. Software, which constantly interfaces Laptop and user, controls hardware operations. It manages data transferring, compiles others user friendly files, processes patterns and generates the machine format of source user text configuration file. User may configure test execution in debug mode or in fully automatic mode.

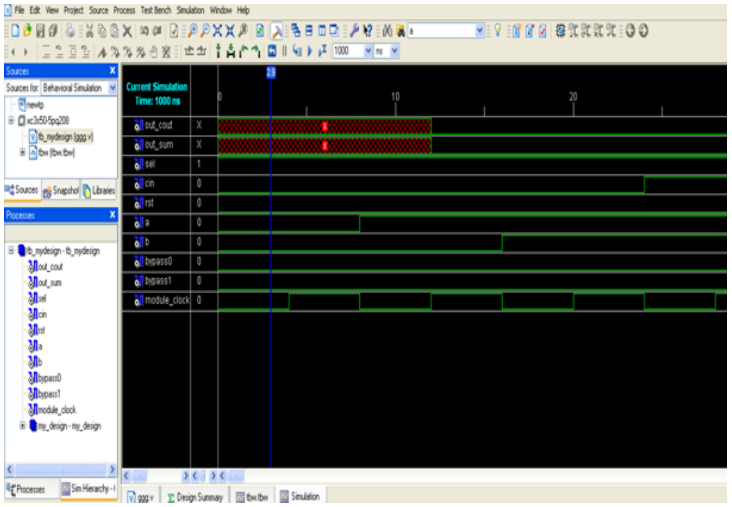

Fig. 3. Simulation Output for given DUT for BBATG

BIST METHOD

The simple multiplexer is used as DUT for BIST method. The coding also written in VHDL and simulated in Model sim software. The results also shown below for three methods.

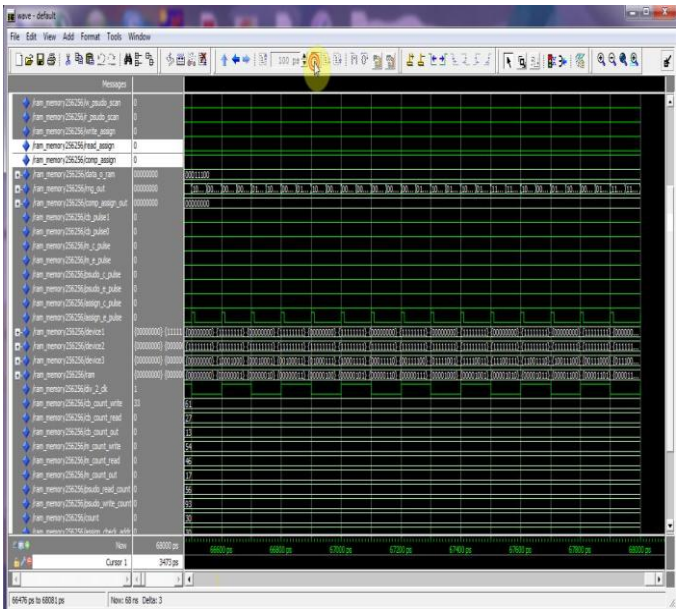

Fig. 4. ATPG METHOD SIMULATION RESULT

4. DISCUSSION AND CONCLUSION

In this paper, a simple test patterns are generated based on the behavioral functionalities of any DUT is processed. Based on that DUT stimulus is generated, tested and samples should be collected. The DUT was simulated in Xilinx and Model sim software. Testing of both synchronous and asynchronous circuits was tested. The results of this research show that behavioral fault simulation will remain as a highly attractive alternative for the future generation of VLSI and system-on-chips (SoC). Since the BIST methods has area overhead, the behavioral method of testing is very useful for testing complex boards. Areas of future research include the refinement of existing faults and defining new behavioral faults to reduce the gap between the behavioral fault models and gate level failures and the development of a better test pattern generator for each target faults. Another valuable enhancement would be to run fault simulation in parallel, by embedding multiple faulty models into one test bench to simulate many faults concurrently. This would speed up the behavioral fault simulation than the gate based synthesis. The implementation of the simulated results yet to be done in FPGA kit.

ACKNOWLEDGEMENTS

I would like to thank my guide Mrs. K. Srividhya., Assistant professor, who guided me to complete this project and also thank everyone those who supported me to achieve my target.

REFERENCES

[1]. Huxun Chen, DeQing Chen, Jinlin Ye, Weizhou Cao, Lei Gao “An Integrated Automatic Test Generation and Executing System” IEEE Conference Sept 2011.

[2]. Luca Mostardini, Luca Bacciarelli, Luca Fanucci, Lorenzo Bertini, Marco Tonarelli “FPGA-based Low-cost Automatic Test Equipment for Digital Integrated Circuits” IEEE International Workshop Sept 2009.

[3]. Mohmed Elbably” An Automatic testing of PLDs”Nineteenth National Radio Science Conference ,March 19-23,2002

[4]. C. Giaconia, A. Di Stefano, G. Capponi, “Reconfigurable digital instrumentation based on FPGA System-on-Chip for Real-Time Applications,” 2003. Proceedings. The 3rd IEEE International Workshop, 30 June-2 July 2003, pp. 120 – 122.

[5]. Ranin Roosta, VLSI Test Automation: Fault Modeling, Ph.D Dissertation, EE623, Department of Electrical and Computer Engineering, California State University, Northridge.

[6]. IEEE Std 1149.1-2001, Annex B: “Boundary-scan description language”

[7]. http://www.verigy.com/ (Verigy website)

[8]. http://www.advantest.com/ (Advantest website)

[9]. W. Quddus, A. Jas, and N.Touba, “Contiguration Self-Test In FPGA-Based Reconfigurable Systems,” Master Thesis, Dept. of ECE university of Texas at Austin, May 1999.

[10]. http://www.inovys.com/ (Inovys website)

[11]. http://www.nextest.com/ (Nextest website)

[12]. S. Hauch, “The roles on FPGA’s in reprogrammable systems,” in Proceedings of The IEEE, vol. 86, no. 4, April 1998.

[13]. S. Konala, P. Chen, and M. Abramovici, “Built-In-Self-Test of Logic Blocks in FPGAs,” Proc. IEEE VLSI Test Symposium, 1996, pp. 387-392.

[14]. C. Stroud, E. Lee, S. Yonala, and M. Abramovici, “Using ILA Testing for BIST in FPGAs,” Proc. Of International Test Conference IEEE, 1996, pp. 68-75.

[15]. E.M. Saad, I.E. Talkhan, I.M. Sayed, “A proposed ATE for digital systems Radio Science Conference,” 1996. NRSC '96., Thirteenth National, 19-21 March, 1996, pp. 209 – 220.

[16]. Nier mann, T., Patel, J.H. HITEC: “A test generation package for sequential circuits”, Proceeding of the European Conference on Volume, Issue, 25-28 Feb 1991 Pages;214-218