International Journal of MC Square Scientific Research Vol.1, No.1,2009

Low Power Dissipation and High Fault Coverage Fault Emulation of Synchronous Sequential Circuits on FPGA

R. Madhusudhanan[1] and T. Kaliraja[2]
[1st] Research scholar, Sathyabama University, Chennai, India
[2nd]Assistant Professor, Department of ECE, SKR Engineering College, Chennai, India
[1]Email: Madhujeppiaar@india.com

Abstract

A feasibility study of accelerating fault simulation by emulation on field programmable gate arrays (FPGAs) is described. Fault simulation is an important subtask in test pattern generation and it is frequently used throughout the test generation process. The problems associated with fault simulation of sequential circuits are explained. Alternatives that can be considered as trade-offs in terms of the required FPGA resources and accuracy of test quality assessment are discussed. In addition, an extension to the existing environment for re-configurable hardware emulation of fault simulation is presented. It incorporates hardware support for fault dropping. This paper presents a low hardware overhead test pattern generator (TPG) for Fault Emulation that can reduce switching activity in circuits under test (CUTs) during testing and also achieve very high fault coverage with reasonable lengths of test sequences. The proposed test pattern generation decreases transitions that occur at scan inputs during scan shift operations and hence reduces switching activity in the CUT. The proposed method is comprised of two TPGs: LT-RTPG and 3- weight WRBIST. Test patterns generated by the LT-RTPG detect easy-to-detect faults and test patterns generated by the 3-weight WRBIST detect faults that remain undetected after LT- RTPG patterns are applied. The proposed approach allows simulation speed-up of 40–500 times as compared to the state-of-the-art in software-based fault simulation. On the basis of the experiments, it can be concluded that it is beneficial to use emulation for circuits/methods that require large numbers of test vectors while using simple but flexible algorithmic test vector generating circuits, for example built-in self-test.

Keywords: LT-RTPG, 3-weight WRBIST, fault simulation.

1. Introduction

The increasing complexity of modern very large scale integration (VLSI) circuits has made test generation one of the most complicated and time-consuming problems in the domain of digital design. As the sizes of circuits grow, so do the test costs. Test costs include not only the time and resources spent for testing a circuit but also time and resources spent to generate appropriate test vectors. The most important subtask of any test generation approach is the

suitability analysis of a given set of test vectors. Fault simulation is the most often used way for that purpose. Efficient fault simulation algorithms for combinational circuits have been known for some time. However, it is the large sequential designs that drive the need for faster implementation (e.g. by hardware emulation). As stated by representatives of the industry (e.g. Motorola at IP-SoC Conference 2003, Grenoble), fault grading times of functional tests for their cores would take up to a decade. Far longer run times have been estimated for larger processor cores .At the same time, reconfigurable hardware devices have been found useful as system modeling environments. This has been made possible by the availability of multi-million-gate field programmable gate arrays (FPGAs). For academic purposes, cheaper devices with rather large capacity, for example the newest Spartan3 chips can be used.

The availability of large FPGAs does not merely allow implementation of the circuit under test with fault models but, additionally, allows the inclusion of test vector generation and output response analysis circuits into a single reconfigurable device. It is important to stress that the current approach is not aiming at testing the FPGA itself or simulating any defects in it: the FPGA is assumed to be tested by the manufacturer. The goal of this paper is to speed- up fault simulation mainly for application- specific integrated circuit (ASIC) and System- on-Chip projects using FPGAs simply as a fast emulation environment. The use of built-in self- test (BIST) would simplify the component testing and the approach proposed in this paper can be used to select optimal BIST structures. For example, the emulation environment can be used for fast fault grading when looking for the best BIST configuration for larger combinational designs. BIST approaches for sequential circuits are lacking a wider acceptance. However, there are classes of sequential cores, where it has been found useful. Schubert and Anheier have proved that most of the cryptographic algorithms are pseudorandom testable; thus, crypto cores can be easily tested by implementing a global BIST approach. In addition, many arithmetic and digital signal processor (DSP) cores are pseudorandom testable. This fact is well supported by the experiments carried out by Raik et al.. Alternatives that can be considered as trade-offs in terms of required FPGA resources and quality of test vectors are discussed in this paper. In addition, an experimental environment for reconfigurable hardware emulation of fault simulation is presented, which supports these different trades- offs.

2. Sequential Fault Emulation

Fault emulation approaches for sequential circuits encounter two major problems summarized in what follows. 1. The sequential fault emulation process takes place as follows. First, fault-free emulation of a circuit is performed. Then, faults are injected one-by-one and for each of them the test stimuli set is emulated. This repetitive emulation requires the circuit to be in an initial state at the beginning of every subsequent emulation stage. The main problem is caused by the fact that after power-up, registers of a circuit will be in an unknown state unless they are set or reset explicitly.

2. The environment, presented in this paper, could be planned to be used with signature analysis, e.g. a multiple input signature register (MISR) for speeding up the fault response analysis. A single X at an input of the MISR would compromise the self-test by making the fault-free signature unpredictable. In that case, all the issues that are related to signature analysis in sequential BIST apply to fault emulation too. In other words, we have to solve the problem of compressing 'don't care' responses to a meaningful signature.

In the following, we present two alternative approaches to sequential fault emulation, which contain the two issues stated previously. The alternatives can be considered as trade-offs in terms of required FPGA resources and fault grading accuracy. Fault emulation with all registers resettable: It is possible to get rid of both the problems mentioned earlier if all the registers in the design are made resettable and the global reset signal is applied at the beginning of the test set. In the general case, not all the registers in sequential designs could be reset. There are two possibilities to solve this issue.

First, it is possible to alter the initial design to contain resettable reg isters only. This approach is used in the current paper. Another option is to modify the circuit only when emulated on the FPGA without modifying the final design itself. The drawback here is that, using this approach, we cannot compare the results directly to the ones of software fault simulation. However, it is a reasonable approach if fast fault grading is needed with limited FPGA resources available.

Encoded 'don't care' value approach: An alternative solution for the problem is to rely on the encoded 'don't care' value approach. It is based on binary encoding of three-valued logic (0, 1, X). Only two bits (a0, a1) are needed for encoding. 0 is encoded as (1, 0), 1 as (0, 1) and X as (0, 0). Table 1 shows the logic used for evaluating the basic gates AND, OR, inverter and XOR. The gates have inputs A, B and output Y. A similar approach has been used in many applications, including the parallel fault simulation method PROOFS. An obvious shortcoming is the duplication of hardware necessary for emulation. The main advantage lies in the fact that the fault emulation results directly match the software simulation.

Table 1Logic used for evaluating the basic gates

	Y_0	Y_1
AND	$A_0 \mid B_0$	$A_1 \,\&\, B_1$
OR	$A_0 \,\&\, B_0$	$A_1 \mid B_1$
Inverter	A_1	A_0
XOR	$(A_0 \,\&\, B_0) \mid (A_1 \,\&\, B_1)$	$(A_0 \,\&\, B_1) \mid (A_1 \,\&\, B_0)$

3. Emulation environment

The emulation environment was designed to work together with a turbo tester (TT): a diagnostic software package developed at the Department of Computer Engineering of Tallinn University of Technology]. The TT software (Fig. 1) consists of several test-related tools – test generation by different algorithms (deterministic, random and genetic), test program optimization, fault simulation for combinational and sequential circuits, testability analysis and BIST simulation, to name but a few. The main advantage of TT is that different methods and algorithms for various test problems have been implemented and can be investigated separately of each other or working together in different combinations. So far we have experimented with only the 'Fault Simulation' part (black part in Fig. 1) to be replaced by the proposed emulation approach. In the future, other simulation-related parts might also be considered (grey parts in Fig. 1).

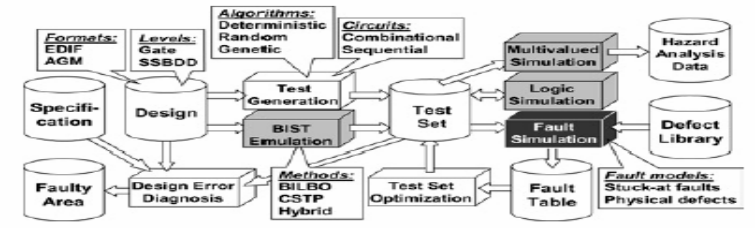

Figure 1 proposed emulation approach

4. Initial emulation environment

It was created keeping in mind that the main purpose was to evaluate the feasibility of replacing fault simulation with emulation. On the basis of that, the main focus was on how to implement circuits to be tested on FPGAs. Less attention has been paid to how to organize a data exchange between hardware and TT. For the first series of experiments, we considered combinational circuits only. Results of experiments with these circuits can be found in a previous work. For sequential circuits, most of the solutions used for combinational circuits could be exploited. The main modification was an extra loop in the controller because sequential circuits require not a single input combination but a sequence consisting of tens or even hundreds of input combinations. Also, instead of hard-coded test sequence generators and output analyzers, loadable modules were introduced. The authors of this paper carried out their first experiments with sequential circuits.

5. Fault insertion

The main problem here was how to represent non-logic features, faults, in such a way that they can be synthesized using standard logic synthesis tools. Since most of the analysis is done using stuck-atone and a stuck-at-zero fault model, the use of multiplexers at fault points was the most obvious one. Also, since a single fault is analyzed at a time typically, decoders were introduced to activate faults (Fig. 2a). The extra multiplexers will increase the gate count (_3–4 times) and will make the circuit slower (typically 5–10 times). It is not a problem for smaller circuits but may be too prohibitive for larger designs – the circuit may not fit into the target FPGA. A solution is to insert faults selectively. A selection algorithm, essentially a fault set partitioning, is the subject of future research.

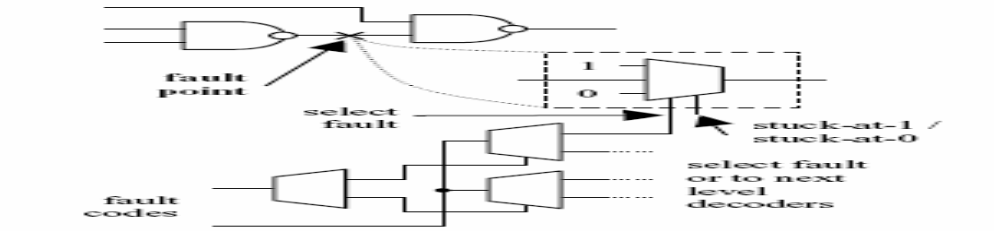

Figure 2a Analysis of single fault

The main advantage is that any fault can be selected in a single clock cycle, that is there is no need to shift the code of a fault into the corresponding register. This difference comes from the way a fault point is activated – multiplexers receive activation through distributed decoders (simplified in Fig. 2a), but when using shift- registers, the activation is shifted from one flip-flop to another (Fig. 2b). It should also be noted that the order of fault points in the shift-register is important because this affects the length of the wires between different flip-flops. The decoders, on the other hand, require more control lines between the circuit under test and controller. Combining both approaches may be the best solution and a part of our future work is planned in that direction.

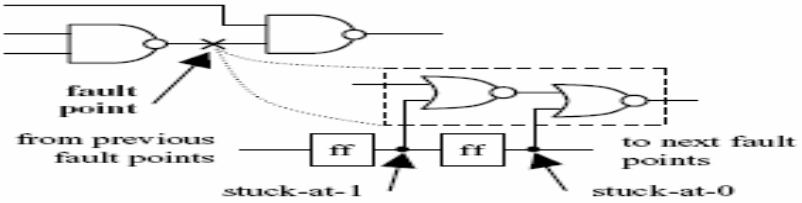

Figure 2b Activation of Fault point

6. Test vector generation: 3-WEIGHT WRBIST

we assume that the sequential CUT has primary and state inputs, and employs full-scan. Even though the proposed BIST TPG is applicable to scan designs with multiple scan chains, we assume that all primary and state inputs are driven by a single scan chain unless stated otherwise (application to multiple scan chains is discussed separately in Section V) only for clarity and convenience of illustration. A test cube is a test pattern that has unspecified inputs. The detection probability of a fault is defined as the probability that a randomly generated test pattern detects the fault [1]. In the 3-weight WRBIST scheme, fault coverage for a random pattern resistant circuit is enhanced by improving detection probabilities of RPRFs; the detection probability of an RPRF is improved by fixing some inputs of the CUT to the values specified in a deterministic test cube for the RPRF. A generator or weight set is a vector that represents weights that are assigned to inputs of the circuit during 3-weight WRBIST. Inputs that are assigned weight 1 (0) are fixed to 1 (0) and inputs that are assigned weight 0. 5 are driven by outputs of the pseudorandom pattern generator, such as an LFSR and a CA. A generator is calculated from a set of deterministic test cubes for RPRFs.

The proposed BIST is comprised of two TPGs: an LT-RTPG and a 3-weight WRBIST Fig.3 . The multiplexer, which drives the input of scan chain, selects a test pattern source between the LT-RTPG and the 3-weight WRBIST. In the first test session, test patterns generated by the LT-RTPG are selected and scanned into the scan chain to detect easy-to-detect faults. In the second session, test patterns that are generated by the 3- weight WRBIST are selected to detect the faults that remain undetected after the first session. Considering the fact that an LT-RTPG can be implemented with very little hardware overhead (only one flip-flop and one AND gate in addition to an LFSR), overall hardware overhead to implement the proposed TPG is determined by hardware overhead for the decoding logic of the 3-weight WRBIST.

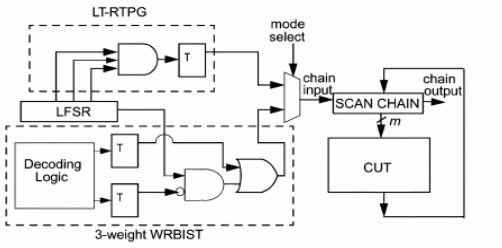

Figure 3. 3 weight WRIST

Automation of emulation environment generation was rather easy because of the modular structure of the hardware part. All commonly used modules are written in very high-speed integrated circuit hardware description language (VHDL) that allows parameterizing design units (Fig). CUT – circuit under test, generated by the fault insertion Program CUT-pkg and CUT-top – parameters of CUT and wrapper for CUT to interface with the generic test environment, generated by the wrapper program Two LFSRs – to generate test vectors and to calculate the

output signature Three counters – one to count test vectors, one to count test sequences and one to count modelled faults (generic VHDL module)

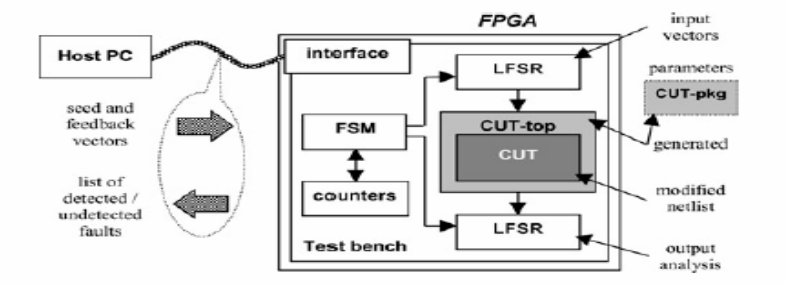

Figure 4 Generic module for VHDL

7. Experimental results

For the experiments, a with a VirtexE chip XCV2000E was used. All the test patterns were generated by a simulation-based ATPG SBGEN from the TT system. In all the experiments, a fault simulator based on a parallel sequential fault simulation algorithm similar to that in the work of Niermann et al. belonging to TT was implemented. Test circuits were selected from ISCAS'89 and HLSynt'92 benchmark sets to evaluate the speed-up when replacing fault simulation with emulation on FPGA. Results of some benchmarks are presented in this paper to illustrate the gains and losses of our approach. Experiments with a few sequential circuits – s27, s953 – from ISCAS'85 benchmark are presented for comparison. Columns labeled 'Number of faults' illustrates the complexity of the test circuits.

Table I comparison of sequential circuits s27,s953 from ISCAS'85

Circuit	Resettable registers Number of faults	Fault coverage, %
s27	46	100
s953	1222	98.2
s5378	5508	66.9
s13207	12 356	26.2
s15850	13 382	32.0

8. Conclusion

This paper has presented a new approach to fault emulation for synchronous sequential circuits. The method implements fully resettable and encoded third value fault grading acceleration. Furthermore, it includes hardware support for fault dropping. The description of an FPGA- based emulation environment has been presented. Different trade-offs in terms of required FPGA resources and accuracy of test quality assessment for fault emulation has been researched. Faults that escape LT-RTPG test sequences are detected by test patterns generated by

the 3-weight WRBIST. The number of weight sets (generators) is minimized by guiding the proposed ATPG with cost functions that reflect the number of conflicting inputs to be incurred by setting an input to a binary value. Experimental results for large industrial circuits demonstrate that the proposed TPG can significantly improve fault coverage of LFSR generated test sequences with low hardware overhead.

References

1. Bardell, P. H., W. H. McAnney, and J. Savir, "Built-In Test for VLSI: Pseudorandom Techniques", New York: Wiley, 1987.

2. Hellebrand, S., J. Rajski, S. Tarnick, S. Venkataraman, and B. Courtois, "Built-In test for circuits with scan based on reseeding of multiple polynomial linear feedback shift registers," IEEE Trans. Comput., vol.44, no. 2, pp. 223–233,1995.

3. Zacharia, N., J. Rajski, and J. Tyszer, "Decompression of test data using variable-length seed LFSRs", in Proc. IEEE 13th VLSI Test Symp., pp. 426–433, 1995.

4. Hellebrand, S., S. Tarnick, and J. Rajski, "Generation of vector patterns through reseeding of multiple-polynomial linear feedback shift registers," in Proc. IEEE Int. Test Conf., pp. 120– 129, 1992.

5. Touba, N. A, and E. J. McCluskey, "Altering a pseudo-random bit sequence for scan-based BIST", in Proc. IEEE Int. Test Conf.,pp.167–175, 1996.

6. Chatterjee, M, and D. K. Pradhan, "A new pattern biasing technique for BIST", in Proc. VLSITS, pp. 417–425, 1995.

7. Tamarapalli, N, and J. Rajski, "Constructive multi-phase test point insertion for scan-based BIST", in Proc. IEEE Int. Test Conf, pp. 649–658, 1996.

8. Savaria, Y., B. Lague, and B. Kaminska, "A pragmatic approach to the design of self-testing circuits", in Proc. IEEE Int. Test Conf., pp.745–754, 1989.

9. Hartmann, J, and G. Kemnitz, "How to do weighted random testing for BIST", in Proc. IEEE Int. Conf. Comp. - Aided Design, pp. 568–571, 1993.

10. Waicukauski, J., E. Lindbloom, E.Eichelberger, and O. Forlenza, "A method for generating weighted random test patterns", IEEE Trans. Comp., vol. 33, no. 2, pp. 149–161, 1989.

11. Tsai, H.C., K.T. Cheng, C.J. Lin, and S. Bhawmik, "Efficient test point selection for scan-based BIST", IEEE Trans. VLSI System, vol. 6, no. 4, pp. 667–676, 1998.

12. Li, W., C. Yu, S. M. Reddy, and I. Pomeranz, "A scan BIST generation method using a markov source and partial BIST bit-fixing", in Proc. IEEE-ACM Design Autom. Conf, pp.554–559, 2003.

13. Basturkmen, N. Z., S. M. Reddy, and I. Pomeranz, "Pseudo random patterns using markov sources for scan BIST", in Proc. IEEE Int. Test Conf., 2002, pp. 1013–1021.

14. Zorian, Y. "A distributed BIST control scheme for complex VLSI devices", in Proc. VLSI Testing Symp., pp. 4–9, 1993.

15. Golomb, S. W., "Shift Register Sequences. Laguna Hills", CA: Aegean Park, 1982.

16. Tsui, C.Y., M. Pedram, C.A. Chen, and A. M. Despain, "Low power state assignment targeting two-and multi-level logic implementation", in Proc. IEEE Int. Conf. Comput.-Aided Des., pp. 82–87, 1994.

17. Girard, P., L. Guiller, C. Landrault, and S. Pravossoudovitch, "A test vector inhibiting technique for low energy BIST design", in Proc. VLSI Test. Symp., pp. 407–412, 1999.

18. Dabholkar, V., S. Chakravarty, I. Pomeranz, and S. Reddy, "Techniques for minimizing power dissipation in scan and combinational circuits during test application", IEEE Trans. Comput. Aided Des. Integr. Circuits Syst, vol. 17, no. 12, pp.1325–1333,1998.

19. Chou, R. M., K. K. Saluja, and V. D. Agrawal, "Scheduling tests for VLSI systems under power constraints," IEEE Trans. VLSI Syst., vol. 5, no. 2, pp. 175–185, 1997.