International Journal of Applied Engineering Research ISSN 0973-4562 Volume 13, Number 7 (2018) pp. 5484-5490

Advanced Testing Methods for Reversible Logic

G. Vamsi Krishna[1*], G. Srinivasa Rao[2], Y. Amar Babu[3]

*[*1]M. Tech, LBRCE, L.B. Nagar, Mylavaram-521230, Andhra Pradesh, India.*
[2]Professor, LBRCE, L.B. Nagar, Mylavaram-521230, Andhra Pradesh, India.
[3]Professor, LBRCE, L.B. Nagar, Mylavaram-521230, Andhra Pradesh, India.

**Orcid: 0000-0002-8875-2981*

Abstract

Testing is an essential step that ensures the designed circuit realizes desired functionality. Testing an integrated circuit is the time-consuming task nowadays. Different methods are needed to get shorter test time. This paper presents an original approach and a practical system for implementation and testing of reversible logic. These testing methods, based on DFT (Design for testability) techniques. Reversible ALU is a testing circuit in this process. This ALU is tested by using two techniques of DFT which improves the controllability and observability of internal nodes, so that embedded functions can be tested. A node is said to be testable if it is easily controlled and observed. The techniques are 1) Ad hoc method and 2) Simple BIST (Built-in self-test) method, BIST belongs to the structured technique of DFT. This design is with Verilog HDL and simulated using ISIM simulator and implemented on Spartan3E (XC3S500E-FG320-5) FPGA. This proposed designed architecture provides delay of 41.054ns for Simple BIST and 40.774ns for Ad-hoc test methods, with an area coverage of 7% and 5% on Spartan 3E implemented by using Xilinx ISE Design Suite.

Keywords: DFT (design for testability), Simple BIST (Built-In-Self Test), Reversible Logic

INTRODUCTION

Fault finding and correction of the faults are the two vital stages of testing the circuits. In the first stage, the fault is identified and in the second stage the fault is corrected. In a chip, several faults occur during manufacturing like bridging faults, missing gate faults etc. Each stage of correction the circuit has its own technique and complexity. Technically VLSI circuits are needed to be tested so that the tested product obtains designed functionality. s having more advantages with reversible logic in digital circuits, they need to be tested before release. Such testing has different techniques with advantages.

The above Figure.1 shows the flow diagram of the design the circuit is designed and circuit is optimized and the optimized circuit is sent to verification. If the verification fails it is sent to debugging mode. In this debugging mode, the source of the error is identified if the error is found then it is sent to synthesis and circuit optimization mode for error correction and if the verification holds then the circuit holds the logic verification and the circuit is said to be ok.

The rest of this paper organized as follows. Section II presents the reversible logic concept Section III presents a review of earlier work Section IV is about ALU and how DFT, is used, In Section V the DFT techniques are proposed. Finally, Section VI gives simulation results and conclusion.

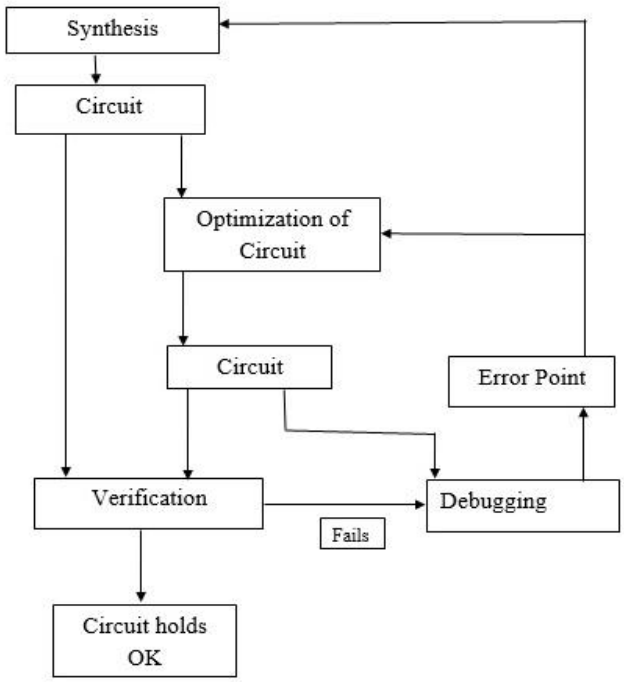

Figure 1. Flow diagram of the design

REVERSIBLE LOGICS

[1]The concept of reversible logic gates is used for reducing power consumption and loss of data. Reversible computing is the application principle of recycling to the computing. It is because input can be reconstructed from the output. This logic uses the reversible gates which have the same number of inputs and outputs it is shown in the Figure.2. Some of the cost metrics like garbage outputs, number of gates, Quantum cost, constant outputs are used to estimate the performance of reversible circuits. A Reversible circuit design can be modelled as a Sequence of discrete time slices and depth is a summation of total time slices.

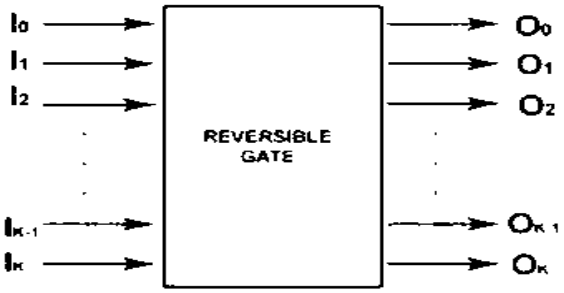

Figure 2. A n × n Reversible Gate

In Digital Electronics, the binary decoder is a combinational logic circuit that converts the binary integer value to the associated output pattern. Various techniques are given to the design of combinational and sequential circuits in the undergoing research. The design of different combinational circuits like a binary comparator, Full adder, Full subtractor, Multiplexer and ALU circuits using reversible decoder is proposed with optimum Quantum cost. The reversible logic gates are

NOT GATE:

A NOT gate is the single Reversible Logic gate. It is 1×1 Reversible Logic Gate. The Not gate gives merely the complementary of the input-output as shown in the Figure.3.

Figure 3. NOT Gate

FEYNMAN GATE (FG):

Feynman gate is a 2×2 reversible gate as shown in below Figure 4. The Feynman gate is used to replicate the required outputs.

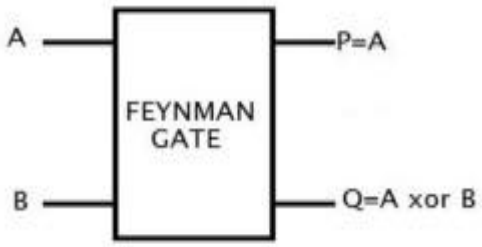

Figure 4. Feynman Gate

DOUBLE FEYNMAN GATE (DFG):

Double Feynman Gate is a 3×3 reversible gate. The outputs are expressed as shown in the below Figure.5 This gate can be used for replicate outputs.

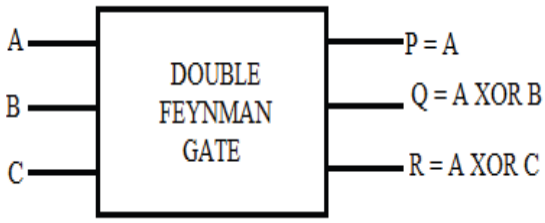

Figure 5. Double Feynman Gate

TOFFOLI GATE(TG):

Toffoli Gate is a 3×3 reversible gate. The outputs expressed as shown in the below Figure.6.

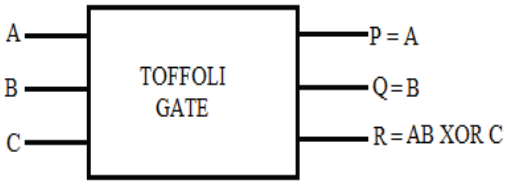

Figure 6. Toffoli Gate

FREDKIN GATE (FDG):

Fredkin Gate is a 3×3 reversible gate. The outputs expressed as shown in the below Figure.7

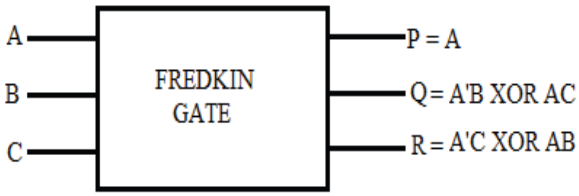

Figure 7. Fredkin Gate

PERES GATE (PG):

Peres Gate is a 3×3 reversible gate. The outputs expressed as shown in the below Figure.8.

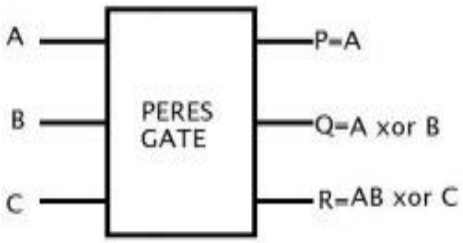

Figure 8. Peres Gate

TR GATE:

TR Gate is a 3×3 reversible gate. The outputs expressed as shown in the below Figure.9

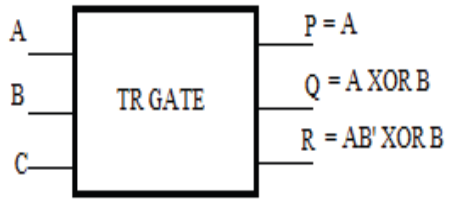

Figure 9. TR Gate

ALU AND DESIGN FOR TESTABILITY

Arithmetic and logic unit (ALU):

An ALU is the fundamental building block of the central processing unit (CPU). An ALU performs basic arithmetic and logic operations. Examples of arithmetic operations are addition, subtraction, multiplication, and division. Examples of logic operations are comparisons of values such as NOT, AND, and OR.

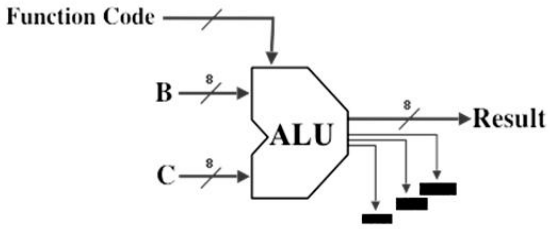

Figure 10. A simple 8 bit ALU

In the Above Figure.10, the function code is referred as the selection of ALU operations which can be either logical or arithmetic operation

Design for testability (DFT):

Design-for-testability improves controllability and observability of the circuit. Hence the function embedded can be tested. The two fundamental properties determine the testability of a node are:

• ***Controllability*:** The complexity of setting internal circuit nodes to 0 or 1 by assigning values to primary inputs (PIs)

• ***Observability*:** The difficulty of observing the state of a logic signal to a primary output (PO).

A node is said to be testable if it is easily controlled and observed. Design for testability (DFT) refers to those design methods that make the task of subsequent testing easier.

DFT TECHNIQUES

There also is no single DFT technique, which is effective for all kinds of circuits. DFT techniques can largely be divided into two categories, i.e., ad hoc techniques and structured (systematic) techniques.

DFT methods for digital circuits:

1) Ad-hoc methods

2) Structured methods: Built-in self-test

Ad-hoc techniques:

Circuits that are large should be subdivided into smaller circuits to lessen the test cost. One of the vital steps in designing a chip for testing is appropriately partitioning the chip. The circuit must be divided into modules at every level of the design. The divided sub circuit should be functional according to the operational boundaries. Partitioning is done using multiplexers and/or scan chain.

The goal was to target modules of circuits that are difficult to test to add circuitry (test point) to improve Controllability Observability.[22] Basic Idea of Ad-hoc techniques is Add MUXs to provide access to/from internal circuitry Controllability & Observability, add gates to provide control to internal circuitry Controllability only. Add these "test points" only where needed in circuit Low area overhead penalty. Little (if any) performance impact Critical paths can often be avoided Target difficult to test sub circuits Potential for significant increase in fault coverage. Creative testability solutions on a case-by-case basis but, we must figure out what & where those are. Some Benefits Provide test points for controllability & observability and easier initialization for logic simulation and design verification. Partitioning the logic into more comfortable test pieces provide access to embedded blocks; Core tests can be re-used Bypass clock generation circuits (oscillators, one-shots, etc.). Avoid or bypass asynchronous logic Break feedback loops (when they are the problem) break up large counters into smaller ones. Disable intentional redundant logic for testing.

BUILT-IN SELF-TEST (BIST)

A built-in self-test (BIST) is a technique that enables a chip to self-test. BIST is a design-for-testability technique that places the testing functions physically with the circuit under test (CUT). The basic BIST architecture requires the addition of three hardware blocks to a digital circuit: a test pattern generator, a response analyzer, and a check controller. The test pattern generator (Linear Feedback Shift Register (LFSR) is implemented in this design) generates the test patterns for the CUT shown in the Figure.11.

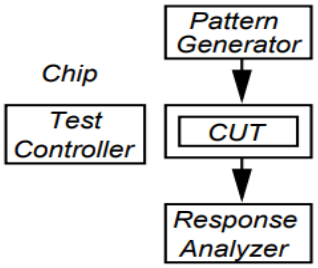

Figure 11. Block diagram of BIST

[21]The main purpose of BIST is to reduce the complexity, and thereby decrease the cost and reduce reliance upon external (pattern-programmed) test equipment. BIST reduces cost in two ways:

- Reduces test-cycle duration
- Reduces the complexity of the test/probe setup, by reducing the number of I/O signals that must be driven/ examined under tester control.

[20]BIST can be used for non-concurrent, on-line testing of the logic and memory parts of a system. It can readily be configured for event-triggered testing, in which case, the BIST control can be tied to the system reset so that testing occurs during system start-up or shut down. BIST can also be designed for periodic testing with low fault latency. This requires integrating a testing process into the CUT that promises the recognition of all target faults within a fixed time. On-line BIST is implemented with dual objectives of total fault coverage and less fault latency. Hence, the test generation (TG) and response monitor (RM) are generally designed to guarantee coverage of specific fault models, minimum hardware overhead, and reasonable set size. These goals are met by different techniques in different parts of the system.

SIMULATION RESULTS

32bit ALU with Reversible Logic

In this Testing of Reversible Logic, we use 32-Bit ALU as circuit under Test(CUT). [1]The reversible 32-Bit ALU is designed An ALU is the edifice integrant of microprocessors and microcontrollers. ALU performs operations as per the designed circuit architecture. The 32-Bit reversible ALU is shown in the Figure.12 and its simulation in the Figure.13. Initially, the reversible logic gates are designed like Feynman, Toffoli, Fredkin, Double Feynman, Peres, TG, etc. Using these gates, we designed reversible decoder combinational circuits like a full adder, full subtractor, multiplexer, etc. These designed circuits used in developing of reversible ALU.

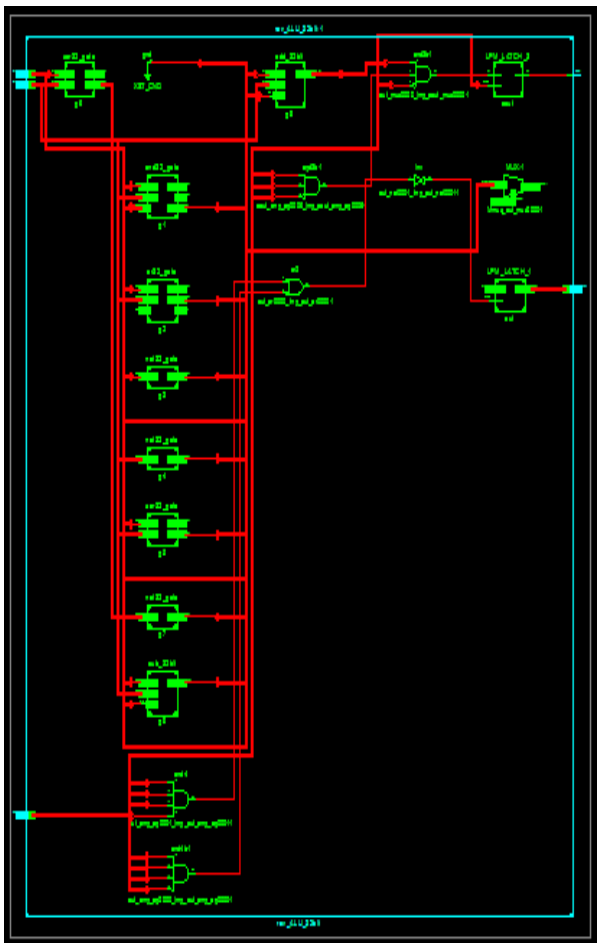

Figure 12. RTL Schematic of 32 bit reversible ALU

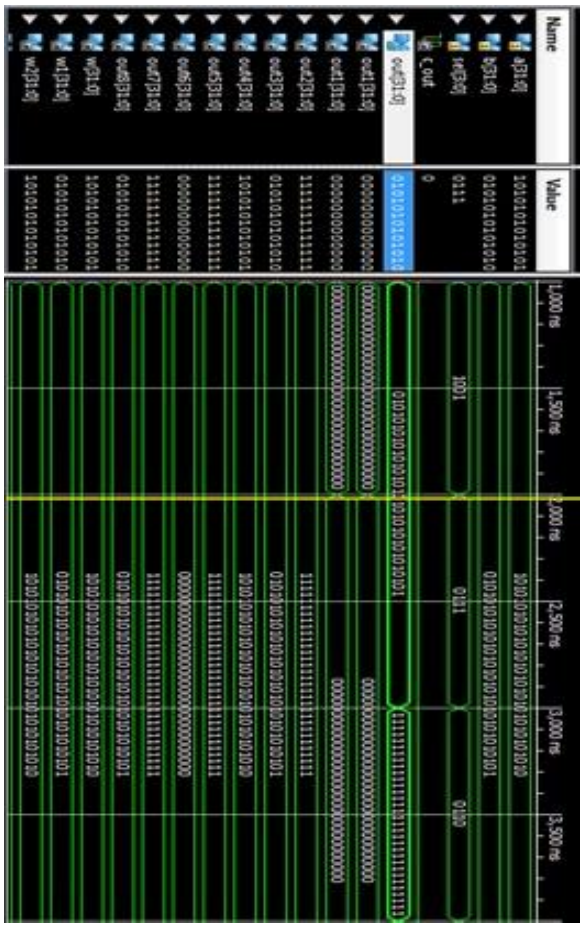

Figure 13. Simulation output of 32 bit ALU

BIST simulation results:

In the simulation results of Simple BIST, Figure.14 shows the RTL schematic of Simple BIST and Figure.15 shows the simulated output of Simple BIST. In this design the area occupied by the Simple BIST circuit on Spartan 3E is 7%

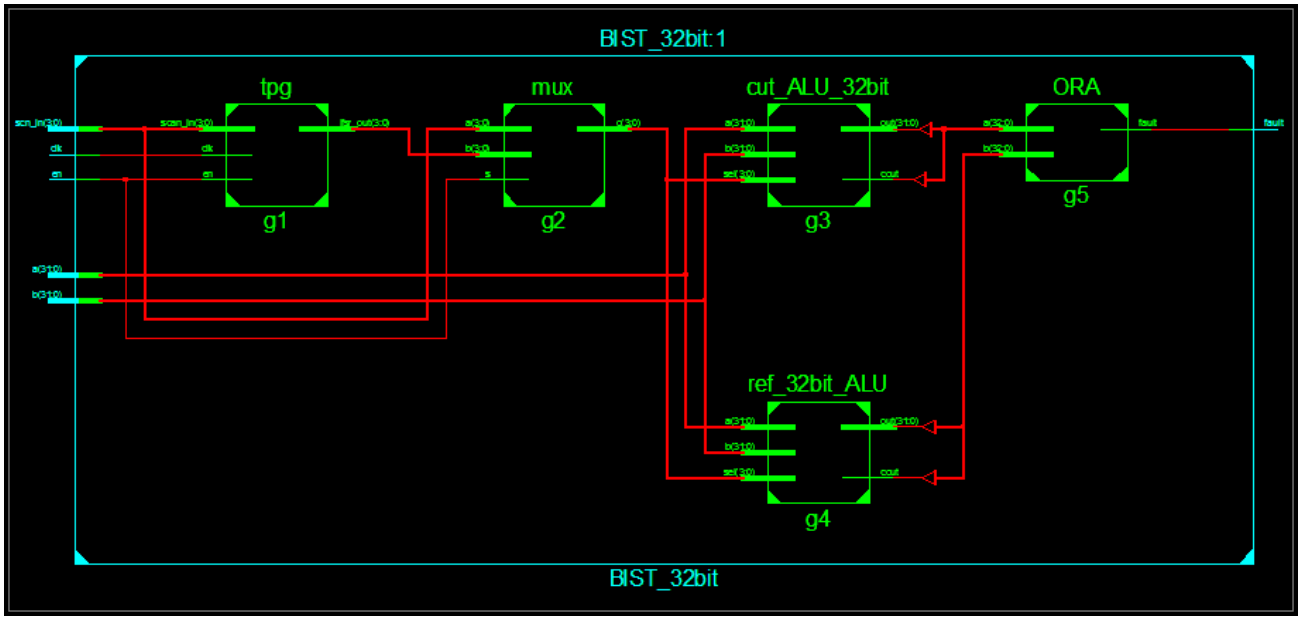

Figure 14. RTL schematic of BIST Test

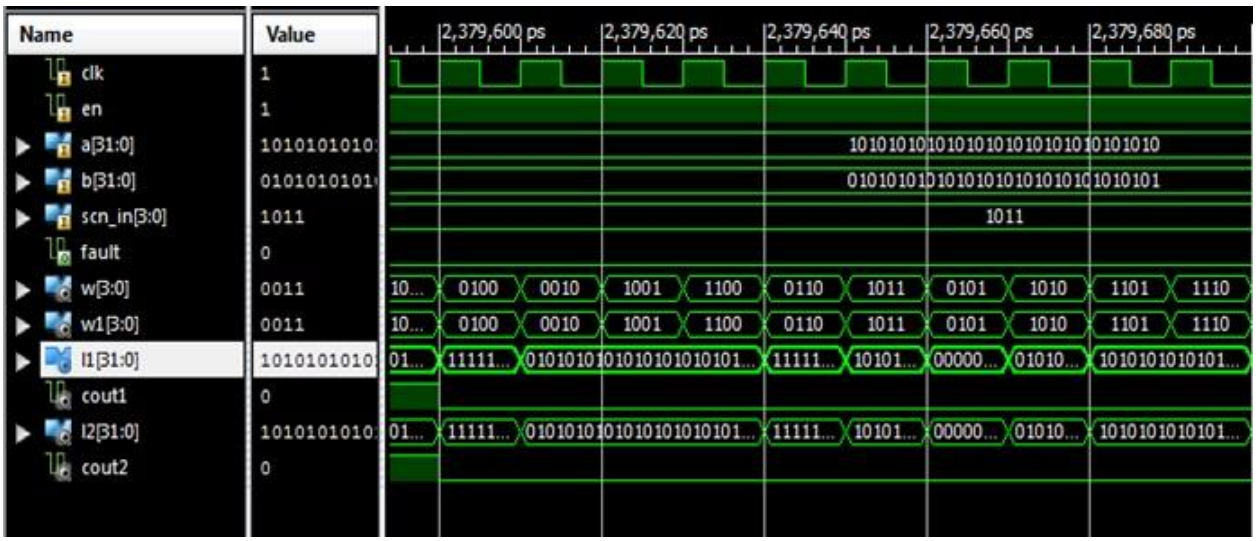

Figure 15. Simulation results of BIST test

This simple BIST architecture provides

Minimum period: 1.932ns

(Maximum frequency:520.034MHz)

Minimum input arrival time before clock: 34.991ns

Maximum output required time after clock: 13.920ns

Maximum combinational path delay: 41.054ns

AD-HOC Testing simulation results:

In the simulation results of Ad-hoc Test, Figure.16 shows the RTL schematic of Ad-hoc test and Figure.17 shows the simulated output of Ad-hoc Test. In this design The area occupied by the Ad-hoc circuit on Spartan 3E FPGA is 5%.

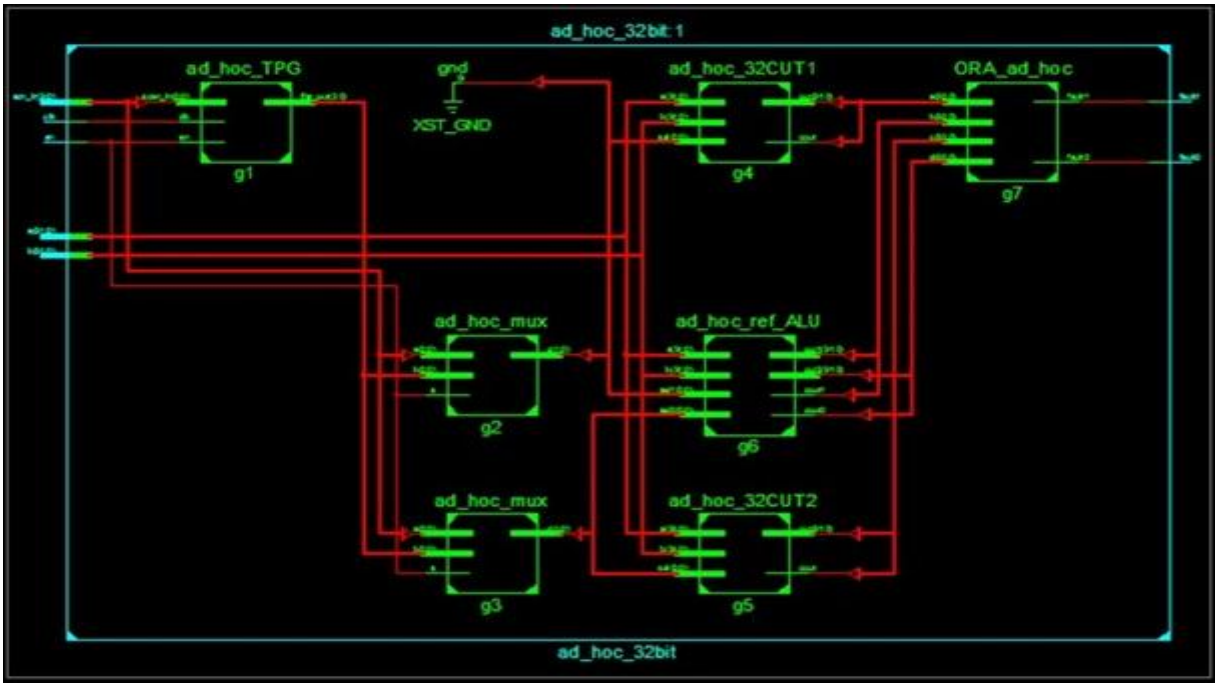

Figure 16. RTL schematic of Ad-hoc Test

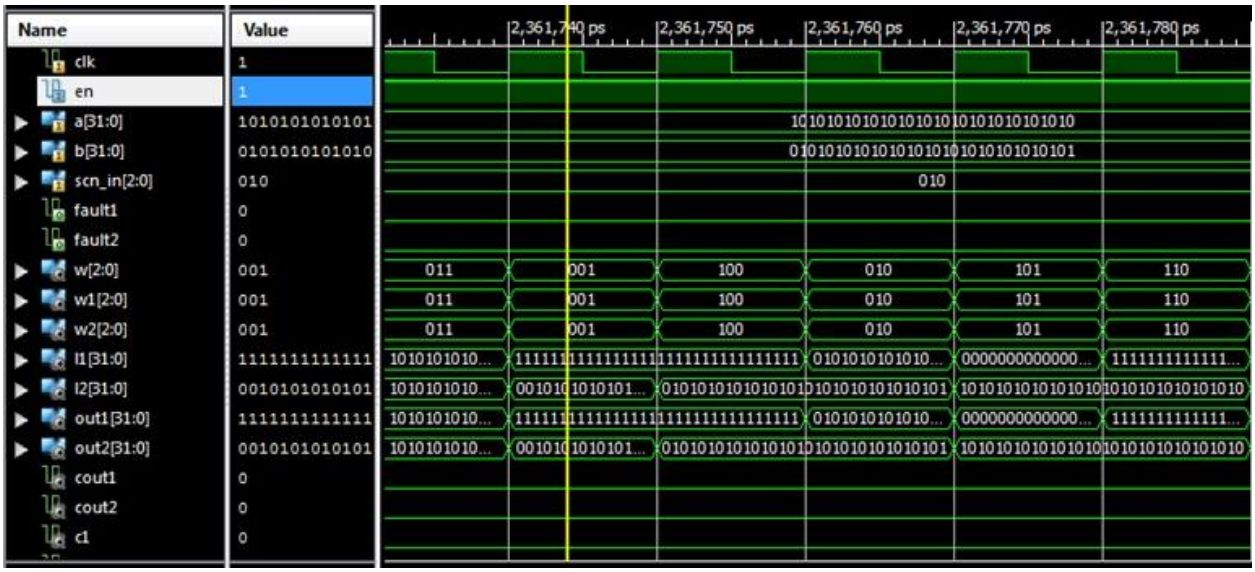

Figure 17. Simulation results of Ad-hoc Test

This simple Ad-hoc architecture provides

Minimum period: 1.932ns

(Maximum frequency:520.034MHz)

Minimum input arrival time before clock: 3.220ns

Maximum output required time after clock: 12.901ns

Maximum combinational path delay: 40.774ns

CONCLUSION

Testing of internal nodes in a complex circuitry is becoming a more significant problem, and thus it is essential that a designer must consider how the IC will be tested, and new structures will be incorporated in the design. Ad hoc and BIST techniques give the best way to check any

combinational circuits. Such testing provides a valid circuitry. This design is implemented with Verilog HDL and simulated using ISIM simulator and implemented on Spartan3E (XC3S500E-FG320-5) FPGA. This proposed designed architecture provides path delay of 41.054ns for Simple BIST and 40.774ns for Ad-hoc technique. Ad-hoc technique occupies 2% less area when compared to Simple BIST. Ad-hoc test will reduce test time compared to BIST. Furthermore, testing methods are to be developed in Reversible logic.

REFERENCES

[1] Krishna, G. Vamsi, G. Srinivasa Rao, and Y. Amar Babu. "An FPGA Implementation of Low Dynamic Power & Area Optimized 32-Bit Reversible ALU." International Journal of Applied Engineering Research 13.5 (2018): 2926-2932.

[2] C.H. Bennett, "Logical Reversibility of Computation", IBM J. Research and Development, pp. 525-532, November 1973.

[3] C H Bennett, "Notes on the History of Reversible Computation", IBM Journal of Research and Development, vol. 32, pp. 16-23, 1998.

[4] A. Peres, "Reversible logic and quantum computers", phys. rev.A, Gen.Phys., vol. 32, no. 6, pp. 32663276, Dec. 1985.

[5] J. Rajski, J. Tyszer, M. Kassab, and N. Mukherjee, ``Embedded deterministic test," IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst., vol. 23,no. 5, pp. 776_792, May 2004.

[6] F. Brglez, D. Bryan, K. Kozminski, "Combinational Profiles of sequential benchmark circuits", *Proc. IEEE ISCAS*, pp. 1929-1934, 1989.

[7] A. S. Abu-Issa and S. F. Quigley, "Bit-swapping LFSR and scan-chain ordering: A novel technique for peak- and average-power reduction in scan-based BIST," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 28, no. 5, pp. 755–759, May 2009.

[8] V. D. Agrawal, C. R. Kime, and K. K. Saluja, "A tutorial on built-in self test. I. Principles," *IEEE Des. Test Comput.*, vol. 10, no. 1, pp. 73–82,Mar. 1993.

[9] J. Rajski, J. Tyszer, G. Mrugalski, and B. Nadeau-Dostie, "Test generator with preselected toggling for low power built-in self-test," in *Proc. Eur.Test Symp.*, May 2012, pp. 1–6.

[10] Y. Sato, S. Wang, T. Kato, K. Miyase, and S. Kajihara, "Low power BIST for scan-shift and capture power," in *Proc. IEEE 21st Asian Test Symp.*, Nov. 2012, pp. 173–178.

[11] E. K. Moghaddam, J. Rajski, M. Kassab, and S. M. Reddy, "At-speedscan test with low switching activity," in *Proc. IEEE VLSI Test Symp.*,Apr. 2010, pp. 177–182.

[12] A. Peres, "Reversible logic and quantum computers", phys.rev.A, Gen.Phys., vol. 32, no. 6, pp. 32663276, Dec. 1985.

[13] J.M. Rabaey and M. Pedram, "Low Power Design Methodologies," Kluwer Academic Publisher, 1997.

[14] T. Toffoli., "Reversible Computing", Tech memo MIT/LCS/TM-151,MIT Lab for Computer Science 1980.

[15] M. L. Bushnell and V. D Agarwal, "Essentials of Electronic Testing" Kluwer academic Publishers, Norwell, MA, 2000.

[16] M. Abramovici, M.A. Breuer, and A.D. Friedman, "Digital Systems Testing and Testable Design", *IEEE Press* 1990.

[17] V.D. Agrawal, C.R. Kime, and K.K. Saluja, "ATutorial on Built-In Self-Test, Part 1: Principles," *IEEE Design and Test of Computers*, Vol. 10, No. 1, Mar. 1993, pp. 73-82.

[18] V.D. Agrawal, C.R. Kime, and K.K. Saluja, "ATutorial on Built-In Self-Test, Part 2: Applications," *IEEE Design and Test of Computers*, Vol. 10, No. 2, June 1993, pp. 69-77.

[19] S. DasGupta, R. G. Walther, and T. W. Williams, " An Enhencement to LSSD and Some Applications of LSSD in Reliability," in Proc. Of the International Fault-Tolerant Computing Symposium.

[20] http://nptel.ac.in/courses/108105057/Pdf/Lesson-40.pdf

[21] https://en.wikipedia.org/wiki/Built-in_self-test

[22] https://www.coursehero.com/file/7352294/Design-For-Testability/