Optimization of Scheduling Technique Rooted on Software Test Routines for Embedded Cores using Quality Factor

Puranik Vishal .G, Dilip D. Shah

Abstract- ***Most of the research work to test the fast processors is carried out using external devices as testers;but it was not technically & financially workable. To fulfill the required performance along with providing efficient functionality, an appropriate testingism must be employed by the digital circuits. The best way is to follow testing as an integral part that is self-test. Conventionally large amount of data was stored in an external tester.But there was a difficulty in at speed testing performance using these external hardware. Hence, Built-in-self-test was invented which verifies failure free nature of circuit under test (CUT) with a test mechanism as a part of system itself. It is observed that, if testing of any hardware is carried out with the help of built-in self test, it increases the requirement of additional area and indirectly responsible for forfeits due to degradation in performance.. If a powerful and power optimized core is to be designed, hardware BIST cannot be afforded due to these limitations. To overcome these disadvantages, a new software based BIST techniques is introduced which relies on software test patterns. Here this paper focuses on rooting of software test routines which works using optimization of scheduling and also a Q- factor is proposed to evaluate the nature of proposed method.***

Keywords— Fault Coverage/detection, Optimization,Q-factor. RISC MIPS Processor, Software test routines, Software-based Self-Testing.

I. INTRODUCTION

As the digital circuits are becoming more complex day by day, they demands a qualititative and easily available testing technique, known as built-in-self-test(BIST)[1]. This method is introduced to reduce complexity of system and to avoid exploring of physical complexity. The features already available in the hardware are utilized in testing using test generation, test imposition and response analysis.

In the architecture of this BIST, three hardware sections are put together in a circuit: A test pattern generator (TPG), An compactor for output data (ODC) , and a controller for examining the tests.The pattern generator initiates the stimuli required for tests of the circuit undert test. These stimuli are to be saved and compared with expected responses. It is based on a divide and conquer strategy, in which a chip is divided into blocks of regular structures, a logic defined randomly, I/O regions, and control. Each section is applied its predefined task of testing and the requirement of hardware.

This results in assurity of optimized testing performance at device-level.The results of most of the Experiments has stated that one scheme does not fit all in enhancing the testability of an entire chip. Traditional hardware self-test (or built-in self-test—BIST) moves the testing task from external resources (ATE) to internal hardware, synthesized to generate test patterns and evaluate test responses of the circuit under test. Hardware self-test achieves at-speed testing, reducing the overall test costs of the chip [3]. To make the design of system BIST ready, major changes in design are required.These changes are incorporated in hardware based Logic BIST (LBIST) [2] and are followed by large industrial designs. In this method the changes in application of tests are done sothat device under test will not enter into an unknown state. Due to these changes, the compressed responses will be stored and saves the insertion of extensive test points during testing. It is expected that the circuits which are having randon pattern resistance, they should gain the desirable percentage of coverage of faults[5],[6]. However, these changes demands for more device area results in poor performance. Hence, LBIST fails to employ for powerful and power optimized embedded cores [7],[8]. The test development designed in the proposed technique is applicable to the register transfer logic circuits The development of test patterns is carried out at RT level in synchronization with simulation and synthesis of system[9], [10]. This technique ensures high convenience and flexibility.The proposed technique performs effectively on the circuits which are not depending on the gates. Using these stratrgies, high fault coverage results are achieved, [12],[13].

In our approach, we have proposed a methodology based on application of test routines to a proposed core is presented. This proposed technique works using functionality of processor. It employs the instruction set of the processor to perform self-testing and software test routines are appied. The topic II explains the concept of scheduling used in SBST to veify correctness of the operations. The different phases of SBST are also explained in this topic. Topic III shows the experimental results given by self designed embedded core considering two methods. In method A, different arithmetic and logical operations are verified in presence of faults. In method B, a quality factor (Q-factor) is proposed which decides the overall operation of system.

Revised Manuscript Received on January 5, 2020

Puranik Vishal .G, Ph.D. Student, Dr. Babasaheb Ambedkar Marathwada University, Aurangabad, Maharashtra puranikvg@gmail.com

Dr. Dilip D. Shah, Jayawant Shikshan Prasarak Mandal , Pune, Maharashtra dilip.d.shah@gmail.com

II. SBST SCHEDULING

An optimization of scheduling of various operations is presented in our method. This approach is used to verify the working of the processor.The imposition and detection of various faults decides the correctness of proposed embedded core. The proposed methodology of optimization is applied on Self designed RISC MIPS Core and Faults and Iterations are varied to get the results. Figure-1 presents different phases of scheduling which are followed for testing of a digital circuit.

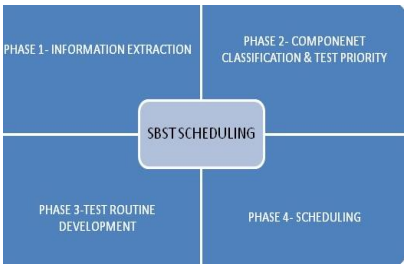

Figure 1: Four Phases of operations.

While determining the coverage, it is observed that the fault detection increases with increase in number of faults and increase in iterations.The experimental analysis showed that as the introduction of faults in the processor is varied from one fault, two faults and all the three faults, there is gradual increase in the fault coverage. On the other side as there is increase in the iterations in introduction of faults, the coverage or detection percentage increases upto the maximum value (98.64 % in our case) and then it remains constant to a steady value.

As shown in table-I, it is also observed that for effective fault detection one should ignore first 2 to 3 iterations in the faults which get added during any operation.

The core achieves expected outcome at iterations from 4 to 20, and remains constant after 20 iterations. Hence we proposed scheduling methodology along with faults / iterations analysis.

Table I: Effect of faults/iterations on Facult detection

S.N	NO.OF FAULTS	NO.OF. ITERATIONS	FAULT DETECTION (IN %)
1	1	0-2	NOT DETECTED
2		3-19	75-98.64 %
3		20 onwards	STEADY AT 98.64 %
4	2	0-1	NOT DETECTED
5		3-19	50-98.55 %
6		20 onwards	STEADY AT 98.55 %
7	3	0-2	NOT DETECTED
8		3-19	85-98.64 %
9		20 onwards	STEADY AT 98.64 %

III. EXPERIMENTAL RESULTS

For experimentation and to carry analysis of scheduling of SBST, an embedded core is of 16-bit is proposed and using RT level logic it is designed, simulated and synthesized using VHDL. RTL is performed after synthesis process.It has a programmable memory. This memory consists of decoder and executor which fetches, decodes and executes various instructions respectively. from which instruction is fetched and passed to instruction decoder unit which has execution unit in it. The results are stored in the register file after performing operations on data which is available in executor.

Table-II: Method of Introduction of Faults.

S.N	FAULT	CATEGORY	BIT LOCATION
I	F1	Stuck at '0'	At bit number 2
II	F2	Stuck at '1'	At bit number 5
III	F3	Transient	At bit number 6

For software based self test a fault model is generated as shown in Table-II and as per positions defined three types of faults are introduced during the clock cycles. The effect of above faults are studied and analyzed to verify correct operation of proposed embedded core.

Method A:- Analysis of Arithmetic & Logical operations

As per the method presented using category of fault and bit numbers of input data in Table-2, faults are added during the input cycle.the faults (F1,F2,F3) being added and those being detected are tabulated.The results for verification and obtained result from process are compared. The percentage of fault detection is determined using combination of faults categorized in table-2 and inputs applied.After performing various operations with different type of input combination and imposing different faults, it is noted that ,the effect of corresponding fault is observed at the same bit location at which faults are imposed as shown in figure 2, 3 and 4. If we substract two inputs and during this operation different faults are added.The number of faults added and detected are used to determine the fault coverage in % as shown in figure-2. The Table-III gives the faults/operation analysis during addition operation.Considering the subtraction operation Figure-2 shows is performed using two data inputs.

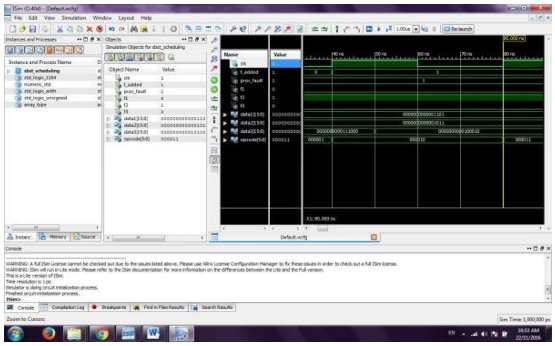

Figure 2:Output of Subtraction Operation with F2 added.

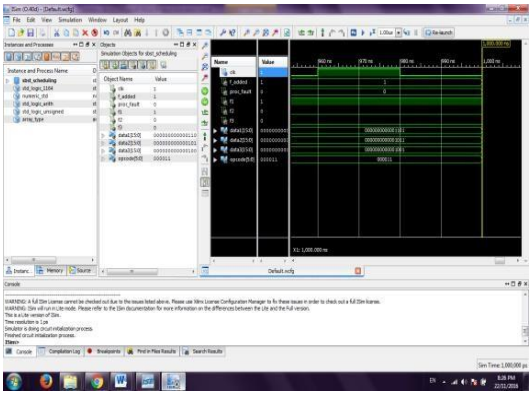

Figure 3:Output of Logical Operation in presence of F1

Figure-3 and figure-4 shows the logical and CRC operations performed using two data inputs and during these operations, different faults are added, the number of faults added and detedted are used to determine the fault coverage in %. Table-IV gives the faults/operation analysis during logical operation.

Figure -4: CRC Operation Output with fault F3 added.

Table-III: Analysis of addition operation in presence of faults

S.N	Type of fault	No.of Faults Added	No.of Faults Covered	Fault Coverage (In %)
1	F3	30	27	90.00
2	F2	16	15	93.75

Table-IV: Analysis of logical operation in presence of faults

S.N	Type of fault	No.of Faults Added	No.of Faults Covered	Fault Coverage (In %)
1	F1	16	15	93.75
2	F3	16	16	100

Hence, the more fault coverage is achieved using this adopted SBST scheduling technique as compared to previous techniques.

Method B:-Quality Factor Analysis

Using the concept of quality factor usded in electrical circuits as shown in Figure-5, a Q- factor ai propose for the same SBST technique. In our case, the proposed Q factor decides the quality of the system depending upon, the of number of iterations and iterations at which desired fault coverage is obtained, decides Quality factor of the system.

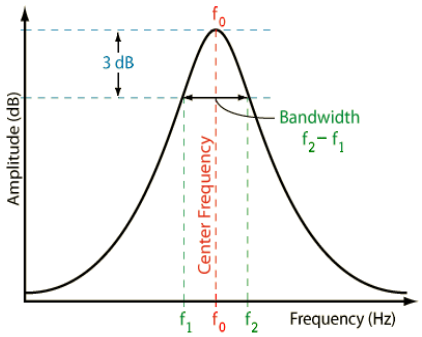

Figure- 5:Quality factor in electrical systems

Therefore,

$$\text{Q Factor} = \frac{\text{Max.allowed No of Iterations}}{\text{2*Iterations at which expected Fault Coverage is achieved}}$$

We have,

$$\text{Quality Factor (Q)} = \frac{\text{Center Frequency}}{\text{Bandwidth}}$$

Here,

For this System, Center Frequency = Expected No.of Iterations = Max.allowed No of Iterations

and,

Bandwidth = Fault Detection (FC)

Therefore,

$$\text{Q Factor} = \frac{\text{Max.allowed No of Iterations}}{\text{Difference of Iterations}}$$

Hence,

1. If No of Iterations are less than Max.allowed Iterations, then Quality Increases and that System is having Good Quality Factor.
2. If No of Iterations are more than Max.allowed Iterations, then Quality decreases and that System is having Poor Quality Factor.
3. If No of Iterations are equal to Max.allowed Iterations, then Quality doesn't affects and that System is Performing as per expectations.

IV. CONCLUSIONS

We have proposed an optimization technique for SBST scheduling using fault/ iterations analysis and Q-factor demonstrated its application on proposed embedded core architectures.The proposed embedded core is following RISC technology and operates at gate level. The experimental results presented shows the high percentage of faults are detected (more than 98 %). This detection percentage is achieved by keeping minimal timing for development of software test routines and also keeping minimum cost for

development and application.Table-V shows Comparision of various SBST techniques.The proposed technique is compared with all existing methods in terms of target processor being used and fault coverage obtained.

Table-V: Comparision of various SBST techniques

S.N	Work carried out	AUTHOR	PROCESSR	FC IN %
1	SBST is appied to processors	Li Chen, Srivasths Ravi, Anand Raghunathan, Sujit Dey	Xtensa Processor	74.3
2	SBST is applied to basic microprocessors	F. Corno, M.Sonza Reorda, G. Squillero, M.Violante	8051	85.19
3	SBST is applied to basic microprocessors with automated testing	F. Corno, M.Sonza Reorda, G. Squillero, M.Violante	i8051	90.77
4	SBST is appied to processors more effectively	N. Kranitis, A. Paschalis D. Gizopoulos Y. Zorian	PARWAN CPU	91.1
5	SBST is appied based on instructions	N. Kranitis, A. Paschalis D. Gizopoulos Y. Zorian	PARWAN CPU	91.34
6	SBST is appied to processors more precisely	N.Krantis, A. paschalis, D. Gizopoulus, G.Xenoulis	MIPS R 3000	92.6
7	SBST is appied to processors using resistor transfer logic	Parisa Shaafi Kabiri, Zainalabedin Navabi	PLASMA	96.01
8	Proposed Technique based on rooting of software test routines and optimization of scheduling.	Puranik Vishal .G DR. Dilip.D.Shah	PROPOSED EMBEDDED CORE	98.64

REFERENCES

1. V.D. Agarwal, ,C.R. Kime and K.K. Saluja, " A Tutorial on Built-In-Self-Test, Part 1:Principles", IEEE Design & Test of Computers, Vol.10, March 1993, pp.73-82.
2. H.-J. Wunderlich and G. Kiefer, "Bit-Flipping BIST," in Proceedings in the International Conference on CAD (ICCAD), pp. 337-343,ACWEEE, November 1996.
3. S. Venkataraman, J. Rajski, S. Hellebrand, and S. Tarnick, "An Efficient BIST Scheme Based on Reseeding of Multiple Polynomial Linear Feedback Shift Registers," in Proceedings of the International Conference on CAD (ICCAD), pp. 572-577, November 1993.
4. Nur A. Touba, "Synthesis Techniques for Pseudo-Random Built-In Self-Test", the Center for Reliable Computing, Stanford University, 1996
5. "Effective RT-Level Software-Based Self-Testing of Embedded Processor Cores" ,Parisa Sha'afi Kabiri and Zainalabedin Navabi , *(DDECS), 2012 IEEE 15th International Symposium* ,Pages: 209 – 212.
6. "Exploring Diagnostic Capabilities of Software-based Self-Tests and In-Field Applications", T. Koal, S. Scharoba and H.T.Vierhaus, *2015 IEEE International Workshop of Electronics, Control, Measurement, Signals and their application to Mechatronics (ECMSM)*, Pages:1-6
7. "Systematic generation of diagnostic software-based self-test routines for processor components", Mario Schölzel, Tobias Koal, and Heinrich T. Vierhaus, 19th *IEEE European Test Symposium (ETS), 2014* , Pages: 1-6.
8. "Systematic Software-Based Self-Test for Pipelined Processors", Dimitris Gizopoulos, Mihalis Psarakis, Miltiadis Hatzimihail, Michail Maniatakos, Antonis Paschalis, Anand Raghunathan and Srivaths Ravi. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 16, no. 11*, November- 2008, Pages: 1441 - 1453.
9. ."Effective Software Self-Test Methodology for Processor Cores", N. Kranitis, A. Paschalis D. Gizopoulos and Y. Zorian, *Design, Automation and Test in Europe Conference and Exhibition, 2002. Proceedings*, Pages: 592 - 597.
10. "Software-based self-test techniques of computational modules in dual issue embedded processors", P. Bernardi, C. Bovi, R. Cantoro, S. De Luca, R. Meregalli, D. Piumatti, E.Sanchez and A. Sansonetti, 2015 *20th IEEE European Test Symposium (ETS)*, Pages 1-2.
11. C:\Users\shivk\refernces ph D\base paper 2.pdf "32-bit RISC CPU Based on MIPS", "Kui YI and Yue-Hua DING, *International Joint Conference on Artificial Intelligence, 2009* IEEE.
12. "Design of FPGA-based Instruction Fetch and Decode Module of 32-bit RISC (MIPS) Processor", Mrs. Rupali S. Balpande and Mrs. Rashmi S. Keote, *International Conference on Communication Systems and Network Technologies, 2011 IEEE*.
13. "Microprocessor Software-Based Self-Testing", Mihalis Psarakis, Dimitris Gizopoulos ,Ernesto Sanchez and Matteo Sonza Reorda,, *IEEE Design & Test of Computers-2010, vol.27, no. 3*, Pages 4-19.
14. "An Efficient Adaptive Method of Software-Based Self Test Generation for RISC Processors ", Ján Hudec, *4th Eastern European Regional Conference on the Engineering of Computer Based Systems, 2014*, Pages- 119-121.

AUTHOR'S PROFILE

Main Author:- Puranik Vishal G .
Affililation: Ph.D. Student, Dr. Babasaheb Ambedkar Marathwada University, Aurangabad,Maharashtra
E Mail address:- puranikvg@gmail.com

About Author:-

Puranik Vishal G. born in Maharashtra (India), completed M.E in Electronics from Dr. Babasaheb Ambedkar Marathwada University, Aurangabad , Maharashtra in 2003. He is doing Ph.D. in Electronics

Engineering from Dr. Babasaheb Ambedkar Marathwada University, Aurangabad , Maharashtra. He is Life time member of ISTE, and other professional bodies. He has teaching experience of 15 years. He has published 15 plus papers at National & International Level. His area of research is VLSI & Embedded Systems.

Co-Author:- Dr. Dilip Devchand Shah.
Affililation:- Principal, Jayawant Shikshan Prasarak Mandal , Pune, Maharashtra
E Mail address:- dilip.d.shah@gmail.com

About Co-Author:-

Dr. Dilip D. Shah has born in Maharashtra(India). He has completed M.E (Electronics) from Walchand College of Engineering, Sangli, affiliated to Shivaji University, Kolhapur and Ph.D from IIT Mumbai in year 2003. He Has got Acadmic & Administrative Experience of 38 years.He is Life time member of ISTE and other professional bodies.He has published about 60 plus research papers are at National & International coneferences and journals.Currently he is working as Principal Jayawant Shikshan Prasarak Mandal , Pune, Maharashtra.