Iterations Based Optimization of Fault detectionTechnique for Software Based Self Test Scheduling

Vishal G. Puranik[1], Dr. Dilip D. Shah[2]
[1]*Ph.D. Scholar,* [2]*Principal*
Dr. BAMU, Aurangabad, M.S. (India)[1]
JSPM's Imperial College of Engineering & Research, Wagholi, Pune, M.S.(India)[2]

Abstract - Most of the traditional Fault detection or coverage measurement techniques are following Built-in self-test (BIST) using test patterns employing embedded hardware components or external testers[1]. Available methods doesn't require any external tester as fault testing is performed at the speed in line with the testee. As a result of this, it faces the difficulties like Area overhead, Performance penalties. Additionally these methods require more design time with extra efforts. A Scheduling technique using software test routines is introduced which overcomes the requirement as in previous methods.
In this paper, optimization of Fault detection method using software test routines is presented. An Evolutionary Software based scheduling test scheduling technique is developed and tested on to a RISC MIPS Processor (An Embedded Core designed & implemented in VHDL).The analysis of fault occurance and coverage is based on the execution of the self-test programs and scheduling is carried out by designed core. In addition with this an evolutionary technique to optimize the fault coverage depending on number of iterations and number of faults is explained.

Keywords - Software Test routines, Optimization, Fault detection, Iterations, Embedded Core.

I. INTRODUCTION

Using different levels applied for division of the devices, BIST employed a best key over system level test. Hence this technique is followed for wafer testing at manufacturing level and device level testing at system level proving Vertical Testability.
In hardware built-in self-test, internal resources are preferred instead of using external testers the entire cost of system is minimized with provision of at speed testing [3],[5]. One of the drawbacks of employing such method is increase in physical space and performance penalties. A better solution is to impose test routines on embedded cores is being used popularly. In Software-based self-testing, functionality of processor along with its instruction set is utilized which yields in an optimized processor design. To perform functionality of a processor, it is bifurcated into its instruction set and modules following arithmetic & logical functions [4]. Thus such technique leads to minimal cost as it requires less test application and development cost. Using following objects an efficient software-based self-testing may be achieved -

A. Coverage of maximum number of faults.
B. The size of test program must be as small as possible
C. The time required to execute the test routines must be as small & as fast as possible.

By introducing as small and fast as possible software codes the last two objectives can be achieved. These self-generated code routines are introduced at higher processor speed.

II. SBST SCHEDULING: BASIC CONCEPT

In this technique using test programs the operations (test code) and operands (test data) are determined. To design effective tests, the specifications of the CPU are also taken into account during program generation. The generation of test routines is performed only once and the resultant test routines can be reused throughout the whole life cycles of the CPU[2],[4]. Before test application, test programs are to be applied onto the chip.
Three phases in together employs SBST methodology for embedded cores[3] [6]. These phases are shown and explained in Figure1.

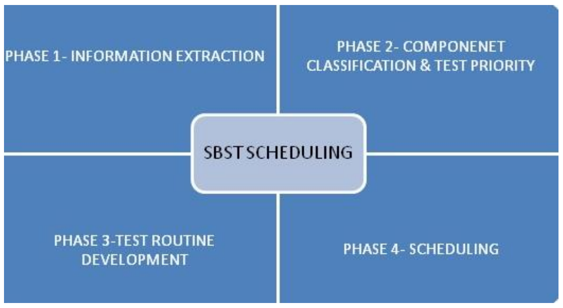
PHASE 1- INFORMATION EXTRACTION
PHASE 2- COMPONENET CLASSIFICATION & TEST PRIORITY
SBST SCHEDULING
PHASE 3-TEST ROUTINE DEVELOPMENT
PHASE 4- SCHEDULING

Fig 1: SBST Methodology Phases

III. PREVIOUS WORK

The SBST approaches found so far in the literature can be classified in two different categories. The first category includes the SBST approaches that have a high level of abstraction and are functional in nature. The second category includes the SBST approaches which are structural in nature and require structural fault driven test development.
The SBST methodology for RISC MIPS Embedded Cores introduced in this work consists of the four phases shown in Fig. 1 and explained as following:

Phase A - In this phase the processor components are identified along with their related operations. After that the instructions which decide excitation of component operations are also introduced. The recognition of controlling and observing Processor registers is also carried out as shown in fig 2[10].

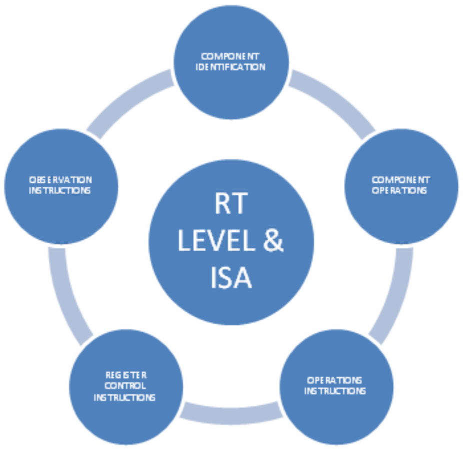

Fig 2: Phase A of SBST

Phase B - Based upon properties, the selected components are categorized in this phase.These classified processor components are ordered for test development depending upon precedence as shown in fig 3[9].

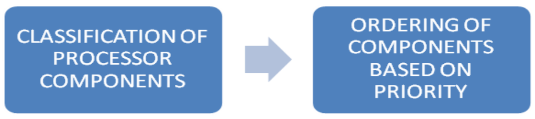
CLASSIFICATION OF PROCESSOR COMPONENTS
ORDERING OF COMPONENTS BASED ON PRIORITY

Fig 3: Phase B of SBST

Phase C - In this phase, small software test routines are developed using available instructions loops.These routines will execute at high speed to achieve effective analysis. These generated test routines then applied on self-designed embedded core architecture for fault coverage as shown in fig 4[7],[8].

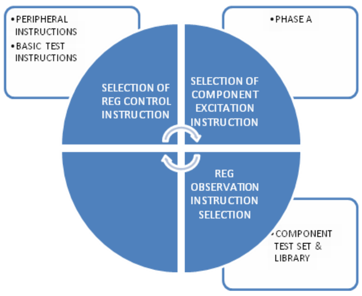
•PERIPHERAL INSTRUCTIONS
•BASIC TEST INSTRUCTIONS
•PHASE A
SELECTION OF REG CONTROL INSTRUCTION
SELECTION OF COMPONENT EXCITATION INSTRUCTION
REG OBSERVATION INSTRUCTION SELECTION
•COMPONENT TEST SET & LIBRARY

Fig 4: Phase C of SBST

Phase D - In addition with SBST, a scheduling technique is applied so as to determine the operation of the designed processor as Faulty or Not Faulty depending on introduction and detection of various faults.

IV. OPTIMIZATION TECHNIQUE

While determining fault 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 and then it remains constant to a steady value.

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.

V. EXPERIMENTAL RESULTS

The proposed methodology of optimization is applied on Self designed RISC MIPS Core and Faults and Iterations are varied to get the results as stated in table 1.

S.NO	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 %

Table 1: Statistical analysis of effect of Faults and 25 iterations on Fault Detection

All these results are plotted using matlab and 3-D plots are observed for different combinations of input values as shown and explained below-

The effect of number of faults and number of iterations on fault detection is plotted in 3-D plot using matlab

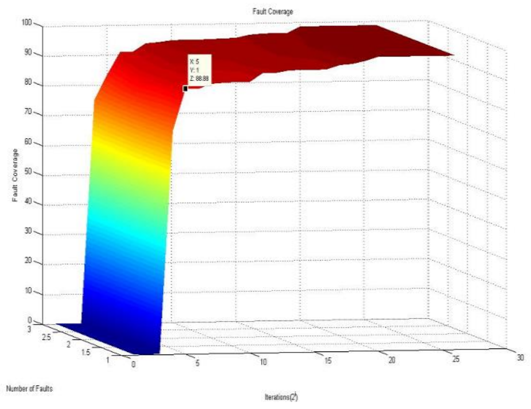

Fig 5: 3-D plot of Effect of F3 and 5 iterations on FC.

Figure 5 shows the 3-D plot during addition operation which shows that as number of faults are increasing and as number of iterations are increasing there is gradual increase in fault detection. It shows effect of fault F3 at 5 iterations the fault detection is 88.88%.

Figure 6 shows effect of fault F1 at 24 iterations, the fault detection is 98.55 %.

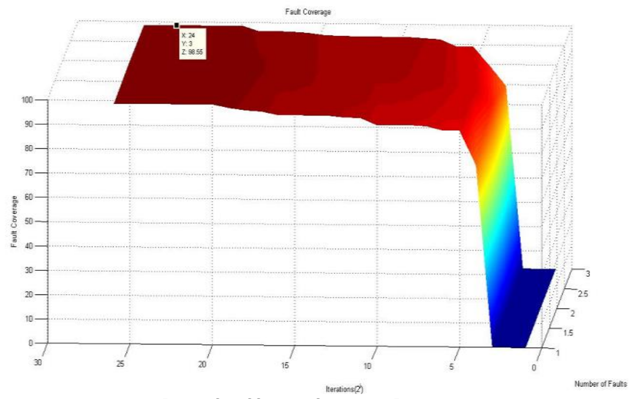

Fig 6: 3-D plot of Effect of F1 and 24 iterations on FC

Figure 7 shows effect of fault F2 at 17 iterations, the fault detection is 96.45 %.

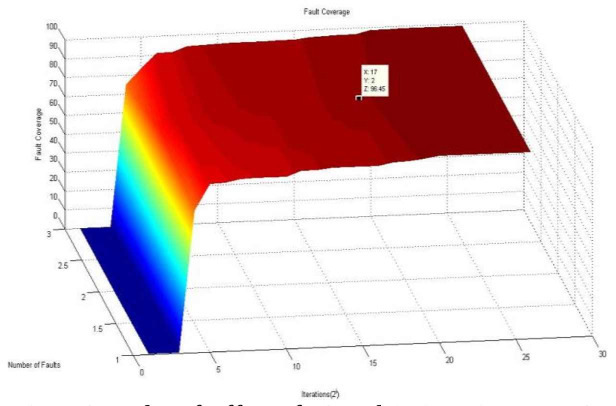

Fig 7: 3-D plot of Effect of F2 and 17 iterations on FC

Figure 8 shows effect of fault F1 F2 & F3 at 8 iterations, the fault detection is 90.90 %.

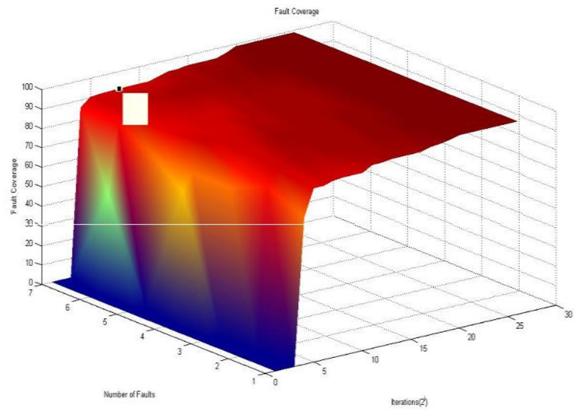

Fig 8: 3-D plot of Effect of F1 F2 & F3 and 8 iterations on FC

VI. CONCLUSIONS

A Software-based self-test methodology is analyzed and demonstrated on a self designed RISC instruction set. The proposed SBST methodology does not rely on manual or automatic tester and works depending upon processor components or any sequential ATPG at the gate level. It canbe applied to any Embedded core whose instructions and its real time behavior is known.The methodology designs small software test routines using fast processing ability for the most important and test critical components of the processor to achieve maximum fault detection. This analysis and detection is costeffective.

We have demonstrated that high test quality (more than 98% fault coverage) is achieved with low test development and test application costs.

Additionally we have presented a method to analyze the effect of number of faults and iterations on fault detection which showed that after certain iterations the fault detection increases with increase in iterations and also as per increase in number of faults.

VII. REFERENCES

[1]. Li Chen and SujitDey, “Software-Based Self-Testing Methodology for Processor Cores”, IEEE trans on computer-aided design of integrated circuits and systems, VOL. 20, NO. 3, MARCH 2001, Pages: 369 – 380.

[2]. N. Kranitis, A. Paschalis,D. Gizopoulos and Y. Zorian,“Effective Software Self-Test Methodology for Processor Cores”, *Design, Automation and Test in Europe Conference and Exhibition*, 2002. Proceedings, Pages: 592 - 597.

[3]. JianShen and Jacob A. Abraham, “Native mode functional test generation for processors with applications to self-test and design validation”, *Test Conference, 1998 Proceedings,* Pages: 990 – 999.

[4]. Praveen Parvathala, Kailas Maneparambil andWilliam Lindsay, “FRITS-A Microprocessor Functional BIST method”, *Test Conference Processing’s, 2002*, Pages: 590 - 598.

[5]. F. Corno, M. Sonza Reorda, G.Squillero and M.Violante,“ On the Test of Microprocessor IP Cores”, *Proc-Design Automation & Test in EuropeConf,2001* Pages: 209-213.

[6]. L. Chen, S. Ravi, A.Raghunathan and S. Dey, “A Scalable Software Based Self-Testing Methodology for Programmable Processors”, *Proc-Design Automation Conference*,2003 Pages: 548-553.

[7]. Sarika U. Kadam, “Design of Five Stage Pipeline RISC Processor”, *IJECS Volume 05 Issue 2* February 2016 pp. 15867-15869.

[8]. Mrs. Rupali S. Balpande and Mrs. Rashmi S. Keote, “Design of FPGA-based Instruction Fetch and Decode Module of 32-bit RISC (MIPS) Processor”, *IEEE International Conference on Communication Systems and Network Technologies*, 2011.

[9]. N.Krantis, A. paschalis,D. Gizopoulus and Y.Zorian,“ Instruction Based Self-Testing of Processor Cores”,*Electronic Testing-Theory & Applications*,2003 Pages: 103-112.

[10]. N.Krantis, A.Paschalis, D.Gizopoulus and G.Xenoulis,“Software Based Self-Testing of Embedded Processors”, IEEE trans on Computers Vso;-54,No-4, April-2005.

About Authors:

1. Vishal Gangadhar Puranik .

Vishal G. Puranik born in Maharashtra (India). He has completed his M.E (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 14 years. He has published 15 plus papers at National & International Level. His area of research is VLSI & Embedded Systems. Currently he is working as Assistant Professor in JSPM's Bhivarabai Sawant Institute of Technology & Research,Pune, Maharashtra.

2. Dr. Dilip Devchand Shah.

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 at JSPM's Imperial College of Engineering & Research,Pune, Maharashtra