Design of a Self-Test Unit for Microprocessor in On-Line Scenario using Verilog

Swathi G Gudadawar[1] Dr. Meghana Kulkarni[2] Rohit B Malagi[3]
[1,3]PG Scholar [2]Associate Professor
[1,2,3]Department of PG Studies
[1,2,3]VTU-Belagavi, Karnataka, India.

Abstract— As the technology grows day by day, testing of an embedded microprocessor in a SoC environment is becoming more complex. The design is explicitly focused on achieving testing speed and less area of the test program in a SoC. In this paper, a proposed technique which presents an innovative solution to the SoC testing. Here, architecture for the embedded microprocessor testing has been designed by using Verilog language. The proposed architecture restricts the processor to execute a test sequence during the normal operation in an online scenario, to save the execution time. Instead, it is connected to the system bus like a normal memory core to perform the test operation. The design saves execution time and uses less area for its testing unit. Nowadays, the Software-based Self-Test (SBST) and Built-in Self-Test (BIST) techniques are very popular in testing Microprocessors to reduce the testing challenges. But, SBST technique is the most preferable one in the on-line scenario. The Proposed architecture which combines BIST and SBST principle gives the perfect system to reduce the testing time in on-line scenario. The benefits of using this architecture is that it doesn't require a memory of the system to store the data, which tries to maintain IP of the processor core and saves the execution time. The proposed solution is designed and simulated using Xilinx 13.1.

Key words: SoC, BIST, SBST

I. INTRODUCTION

As VLSI technology increases its innovation quickly makes possible the integration into a single chip. An SoC can be defined as a system on an IC which integrates both hardware and software IP by using many design methodologies in order to perform a specific functionality defined by the proposed system behavior. Intellectual property means developing any one of the application by using someone's knowledge and effort and making it to work as intended. The benefits of using IP's are:

- Avoids reinvention for every new products
- To increase the development of new products
- To avoid the chances of failure based on the design and verification of a block for the first time.

II. LITERATURE SURVEY

In this paper, test generation programs have been developed in order to test the microprocessor. Using instruction set and the functions which are performed at any of the microprocessor will be easily constructed manually. A graphic-theoretical model has been used for microprocessor testing. [1] The principle of Motorola microchip has been designed for no specific test components. In a Motorola microprocessor, the built-in-self-test is considered as a small until the RAM test application feature has been recast. [2] The main aim of SBST is to make use of on-chip programmable resources to execute ordinary programs which tests the processor itself. The processor produces and brought functional test-patters using its own instruction set which removes the requirement of additional specific hardware for testing. [3] By making the target as a structural test requires manageable components with the help of processor functionality. The technique presented had the advantage of fault coverage of deterministic structural testing and the advantage of functional testing at high speed. The methodology here presented had been done on a simple accumulator-based microprocessor. [4]This paper explains how to avoid data hazards and to improve the performance using register forwarding and pipeline interlocking.[5] In modern Microprocessor, Speculative execution instructions will help to improve the performance. By using speculative mechanism, control and data dependencies can be reduced. In this paper, generic software-based test method has been implemented with the available processor resources. [6]

III. TESTING

As deep submicron technology increases in a SoC presents a great difficulty in its testing. In SoC, Self –testing technique will play a unique role. These are of two types

- Hard-ware Based Self-Test
- Soft-ware Based Self-Test

A. Hardware Based Self-Test:

Hardware Self-test requires an external hardware structures for an example: Built-in Self-test and Logical Built in Self-test could be included. These techniques are particularly suitable for IP cores but not for timing and power consumption constraints. Built-in Self-Test is one the method used for self-testing where embedded hardware test generator and test response analyzer will be used to apply and to generate test patterns on chip at the speed of the circuit.

B. Software Based Self-Test:

Another self-testing methodology named as Software-Based Self-Testing (SBST) based on testing a microprocessor using its instructions. Software Based Self-test methodologies are suitable for embedded microprocessor core testing. It depends upon the suitable test pattern generation for execution, it doesn't require any external hardware part for testing and it requires only the functionality which exits in the processor core. The structure of the processor can be tested by executing the test patterns. Both SBST and BIST run at the speed of processor but there are some of the disadvantages of BIST over SBST which is more important one.

- SBST executes the program in the functional mode of the processor whereas the BIST executes in the non-functional mode.

- BIST depends on the pseudorandom test patterns application and its generation. So the fault coverage could be low because of processor's random pattern-resistibility.
- SBST will not require any of the hardware part or doesn't require any changes in the processor during testing. But BIST circuit has to achieve testing that might cause some problems in area, performance and power consumption.

All these problems which are encountered in BIST have been removed out and it has been improved.

C. Types of Testing:

1) Manufacturing Testing:

The performance of the target device can be evaluated at the production time by using Manufacturing Testing. The main goal of the manufacturing test is to arrange good devices as a separate category from bad devices. Also it helps to identify the defects in any phase of the fabrication process by performing some type of diagnosis in the produced chip. Some methods and equipments determine that manufactured chip behaves as intended and has no defects in it after the complete manufacturing process. Manufacturing test can be performed by applying the test patterns to a device by means of Automatic test equipment and sometimes by using Design for-test features on the device itself. The test responses from the device which is under test is received and determines the good products.

2) On-Line Testing:

Online testing can be defined as a group of techniques which is aimed to detecting the occurrence of a fault during the product production time and the possible misbehaviors can be corrected at that time only. A semiconductor device also depends on the assembly, use, and also on the environmental conditions. Stress factor will also affect device reliability including gas, dust, contamination, voltage, current density, temperature, humidity, mechanical stress, vibration, shock, radiation, pressure, and intensity of magnetic and electrical fields. If there should be a guarantee for a system or component throughout lifetime, only the manufacturing test is not sufficient. Other testing procedures are also needed to perform during its useful lifetime. There are different techniques for online testing with different levels of certainty and results for the application. When a system is at start, before it enters the actual service, testing of all or some of its components can be tested called as Start-up test. It is valuable for frameworks that are reliably Restarted all through their main goal time, similar to the Power-On Self-Test (POST) of a computer memory. Non-concurrent online testing is another system consisting in testing the framework while the application is running, yet with a few performances of services. The testing can be performed at one component of the system or at one segment of the component which will not be useful for the application testing. Third one is the concurrent online testing; it performs testing the device or a system when the application is fully executed in the device during test. This is less disturbing strategy from the user application point of view. However, finding the proper stimuli and collection of results and methodology analysis are not very easy work. Concurrent error detection (CED) mechanism is one of the online testing methods for memories and processors, where a predictor of an output characteristics and a checker, can detect errors when the application is running.

3) A New Method for Memory Testing:

A memory core testing in an SOC is not new concept. Arrangements in view of furnishing centers with suitable BIST hardware are generally received and speak to a viable arrangement. Since there is a great demand for new testing techniques from last few years a new technique has arrived. As there is a great demand in the electronic system in the safety critical applications along with the higher sensitivity of semiconductor technology and some other phenomenon will cause failure thus significantly increases the importance of device testing during the operational phase only i.e. online-phase . The BIST solutions are not suitable for the embedded memories sometimes to match the constraints explained above because the BIST circuits can't be active during the operational phase or it has not designed to support at the operational phase. Sometimes by using the traditional BIST approaches are is not possible to test the faults which will affect the interconnections between the memory and circuit surrounded by it. As a result, during the operational phase the testing of embedded core are performed by forcing the processor core which runs the suitable test programs accessing the memory and the same sequence of read/write operations mandated by a given algorithm can be implemented in software. This is very effective in terms of cost since it doesn't require any hardware and the implemented test algorithm can be easily changed so flexibility is more and one more advantage of this software BIST approach the test can be easily triggered whenever it is required. But still these approaches presents some of the drawbacks that it can't be used if the memory can't be accessible by the processor.

Sometimes it will be very hard to keep the many implementation details secretly in order to preserve the IP, which are provided by the memory core provider. Software BIST will be not able to detect fault coverage as how the hardware BIST detects, in terms of delay faults and speed related faults. Moreover, the Software BIST approach needs the test code to store somewhere in the application memory, thus the cost will also increase and some issues will also rise in embedded applications in terms of memory. To address the above issues, at-speed test, and cost constraints, any suitable implementation of a memory test solution to adopt during the operational phase should satisfy the following requirements:

- Modularity: Without having the complete knowledge of the details of the different IP memory cores, solutions have to be designed.
- Programmability/Flexibility: The solutions have to be changed easily with facing any difficulty to deal with the unexpected situations.
- Scalability: With any design and technology the solutions should be scalable independent on the complexity.
- Compatibility: Solutions should be compatible with standards.

IV. METHODOLOGY

There are an increasing number of embedded memory cores in the system-on-chip, where the testing operation is a requirement again in the safety critical applications. The

proposed solution will be easily adopted for the embedded memory modules testing during the operational phase moreover the solution are modular and do not require any modification neither in the memory cores nor in the processor. It requires fewer resources in terms of memory so that the cost will be reduced and also saves the test time compared with hardware BIST solutions. In order to overcome the problems presented by SBST, a novel methodology is proposed called as Self-Test Unit for Microprocessors, which will be suitable for online testing .This, method will combine some features of hardware based and functional based techniques i.e. the instructions which are executed by the processor are generated by the on-chip in the Self-Test unit during the test mode by avoiding the use of data or code memory and it is completely based on the instructions so still it falls in the SBST domain. The proposed methodology is predominantly in light of two thoughts:

- The system can be either in normal mode or in the test mode: during the test mode, the processor executes the instructions provided by the Self-Test unit and whatever the processor is performing the normal operation instructions are provided by code memory.
- The unit encrypts the input data before applying to the Self-Test for Microprocessor unit, so that it will provide some security to the Self-Test for Microprocessor unit.

During test mode, the processor will execute the instructions which are provided by the Self-Test unit, but the control will not depend on the flow of instructions execution. The instruction flow will be decided by the Self-Test unit independently of address generated by the processor.

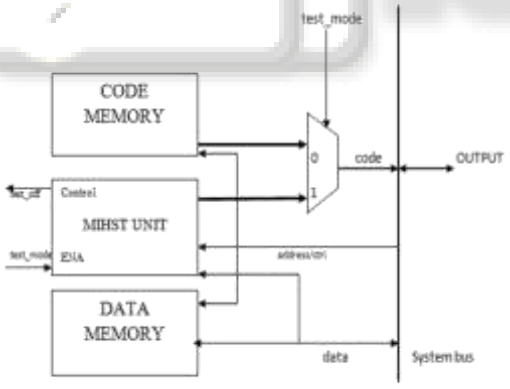

Fig. 1: Architecture of a System with Test Unit

One of the advantage is introduced in this paper is that the reduction in terms of accessible memory area with respect to the SBST, i.e. Self-Test for Microprocessor doesn't require the system memory in order to store the code and data. Including this, the Self-Test for Microprocessor unit reduces the execution time. The Self-Test unit can be programmed with highly encoded data that can be hardwired in it or can be uploaded from outside of the unit. The proposed strategy for microprocessor testing unions a portion of the principle of Software based self-test and Built-in self-test. The technique chooses an infrastructure-IP which is intended to connect to the system bus. The interconnection of the I-IP inside of a processor based on SoC is upheld by a multiplexer. The Microprocessor Self-Test unit will not be recommended for any changes in the processor's core which are similar to BIST. The I-IP occurs only during the test phase, and transparent during normal mode. Concentrating on on-line testing, the methodology will take the advantage of microprocessor interrupt features to save its present state; moreover the data memory content will not be affected. The testing unit behavior mainly based on two principles:

- The unit will force the instruction sequence on the bus which is completely independent on the address generated by processors.
- The functional test program is completely used to store the instructions which save time and space.

Instructions are entered in the bus according to the flow which is chosen by Self-Test unit for Microprocessor and not related to the internal condition of the processor.

A. Architecture of a Self-Test Unit for Microprocessor and its behavior:

The Self-Test unit for Microprocessor will operate like a normal memory core, giving the values to processor that will enable the execution of the test program. A multiplexer allows operating with data bus by using instructions coming as a alternative of from the code memory or from Self-Test unit for Microprocessor. The processor core doesn't use any cache or it can be disabled during test. During the test, in a system along with a Self-Test unit for Microprocessor the system memories can't be accessed. All the read, write and fetch operations are directly performed with the test unit and it behave like a cache towards the processor.

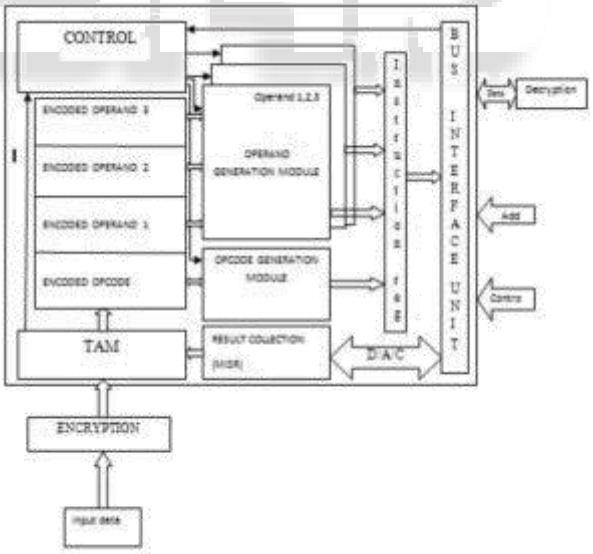

Fig. 2: Schematic perspective of the Self-Test Unit for Microprocessor Architecture

During the test mode, the system uses the Self-Test unit for Microprocessor instead of code memory and communicates with the processor by performing the following operations:

- It will instruction fetch cycle is detected
- Instructions are read and decrypted form its memory
- Instruction codes are generated for the processor.

– By compacting the values which are travelling on the address, data and control buses the processor behavior can be observed.

The unit reads the address, control and data buses in order to monitor those and to conform to the bus protocol when it must compose on them. Architecture includes the modules which are represented in the fig 2 explained below:

– Encryption: The data can be encrypted before applying to the Self-Test for Microprocessor unit.
– Test Access Mechanism: It's an optional module, when the Self-Test for Microprocessor unit has to be interfaced with the Automatic test Equipment by that
– Time the instructions can be uploaded from the outside .It is not compulsory that it has to be uploaded from the out, the test program can be hardwired in it when dealing with on-line testing. When the data has to be uploaded TAP (test access port) can be used. To support test functionalities which are built-in into the components, a general-purpose port, Test Access Port (TAP) will provide access including the test logic is already defined.
– Opcode: Instructions are identified along with the information of instruction execution including the Self-Test unit for Microprocessor.
– Operands: It describes which operands to be applied along with the gradual development of test program.
– Opcode Generation Module: It generates the instruction Opcode for the microprocessor
– Operand Generation Module: It generates the instruction operands. IT can be replicated more than one time depending on the maximum number of instruction operands which are under test. Some simple manipulations like increment, shift can be brought into an action by using this module.
– Control Unit: It manages all the application flow which works together with the bus interface unit.
– Bus Interface Unit: It reads and write the data on the system bus
– Result Collection: A Multiple input shift register (MISR) is implemented as result collection module. The results are stored in the result collection module for the purpose of read operation.
– Decryption: The data which has to be written on the system bus has to be decrypted which is encrypted already at the beginning.

B. Self-Test Unit for Microprocessor in Online Scenario:

During on-line testing, the test is activated by software means by the operating system. For this purpose the exception handling or an interrupt feature can be used for the processor. The complete system for the On-line testing is shown below:

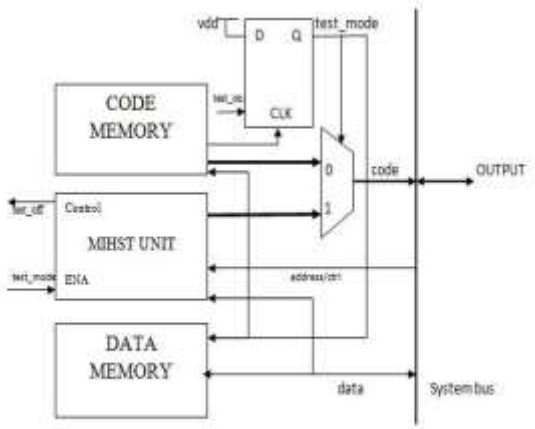

Fig. 3: A complete system along with the Self- Test unit for Microprocessor under test

When the test is activated, an interrupt occurs by saying that it has to perform a test operation when the test mode is on indicated by the output of the D-FF. Then the occurred Interrupt Service Routine is executed carefully.

The beginning and ending address with their respective data is stored in the code memory of the system. The following steps will be carried out during the execution of an interrupt:

1) The interrupt arises for the On-line test in the processor.
2) The microprocessor saves the system data in terms of PC and flags depending on the processor architecture.
3) The ISR takes the control of the system, stored in the code memory.
– The general purpose register which are used by the test program is saved
– The enable signal of Self-Test for Microprocessor unit becomes high which forces the system to enter the on-line mode
4) The Self-Test for Microprocessor unit takes the control over the bus in which,
– In order to clean the pipeline, a sufficient number of NOP instructions are performed
– Unit sends all the instructions related to the test program
– The instructions are executed and finally the data is written on the bus and the data which is stored in the result collection unit of Self-Test for Microprocessor module will be read out.
– When the Enable signal of Self-Test for Microprocessor becomes low, the processor exits the test mode and disables the Self-Test for Microprocessor unit.
5) The control of the bus is given back to the ISR which restores the general purpose register contents and executes the return instruction to perform the normal operation.

V. ADVANTAGES

– Confidentiality: Since the input of the test program is encrypted, it provides security to the module.

- Low cost: The experimental result shows that it is cheaper since it doesn't require any hardware and it is relatively small.
- Speed: The solutions are faster than the Software BIST.
- Programmability: The test program which is executed by the processor can be changed by uploading a new encrypted test program into the Self-Test unit for Microprocessor, which might be equipped with an internal Flash memory accessible from the outside through a suitable interface.
- Ease of integration: The method can be easily integrated into an existing system and test flows and it doesn't require any modification in the Processor and the memory core.
- Modularity: A memory core provider may only give to the customer the Self-Test unit for Microprocessor in order to a test solution to the customer and which is a plug-and-play component will support on-line test.
- Scalability: The same testing unit can be re-utilized for the test of different memory cores which are present inside of the same SoC, gave that they can be access to by the processor; neither the Self-Test unit nor the embedded memory size rely on upon the size of the memory to be tried .

VI. Results and Analysis

The project is designed using Xilinx 13.1. The basic gates and basic modules of the architecture have been designed, simulated and verified. The Simulation Output of the Self-test Unit is shown in the figure 4. First the input is encrypted and given to self-test unit and the output from the unit is decrypted and is appearing on the system bus as shown in the figure 5.The Simulation results of Self-Test Unit during the off-line mode is shown in the figure 5.

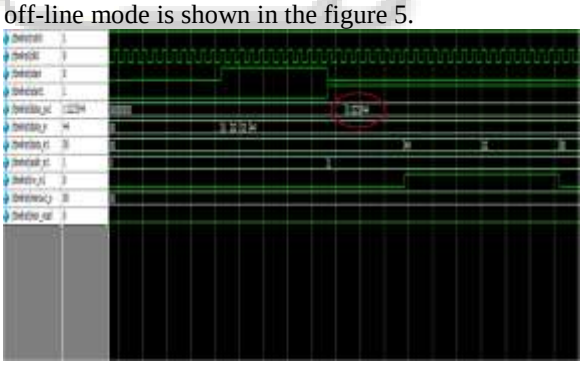

Fig. 4: Simulation Result of Self-Test Unit

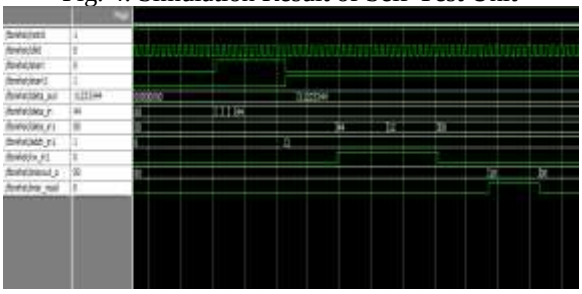

Fig. 5: Simulation Result of Self-Test Unit during On-line Mode

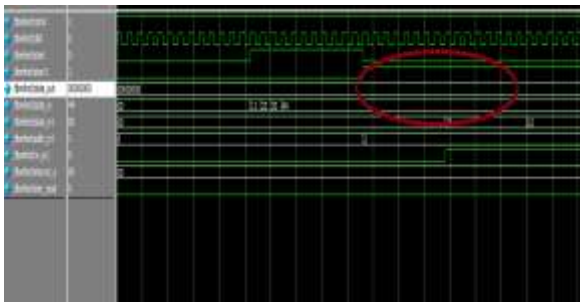

Fig. 6: Simulation Result of Self-Test Unit during Off-line Mode

VII. Conclusion

The Self-Test Unit and the system required for testing have been successfully designed using Xilinx 13.1 by using Verilog HDL programming language. The approach depends on the introduction of the bus, without any changes in the processor and the memory core. The Unit stores the encrypted data in the unit which will be executed on the memory and during the test mode these instructions are executed on the bus. The method reduces some of the limitations presented by the SBST approach in the online scenario; the test program size has been reduced and speed has been improved. The minimum period used by design is 3.367ns and the frequency is about 297.03MHz.The memory used by the design is about 277MB and the number of slices register is about 1% in this design. The data memory and the code memory requirement have been reduced for the testing purpose; the invasiveness of test program is reduced with respect to the normal memory behavior. The cost has been reduced in terms of test time and it helps to preserve the processor core intellectual property. The results are simulated using Xilinx ISE 13.1 tool using Verilog.

References

[1] S.M. Thatte and J.A. Abraham, "Test Generation for Microprocessors," IEEE Trans. Computers, vol. 29, no. 6, pp. 429-441, June 1980.
[2] R.G. Daniels and W.C. Bruce, "Built-In Self-Test Trends in Motorola Microprocessors," IEEE Design and Test of Computers, vol. 2, no. 2, pp. 64-71, Apr. 1985..
[3] M. Psarakis, D. Gizopoulos, E. Sanchez, and M. Sonza Reorda, "Microprocessor Software-Based Self-Testing," IEEE Design and Test of Computers, vol. 27, no. 3, pp. 4 19, May/June 2010.
[4] L. Chen and S. Dey, "DEFUSE: A Deterministic Functional Self-Test Methodology for Processors," Proc. Very Large Scale Integration (VLSI) Test Symp., pp. 5-262, 2000.
[5] P. Bernardi et al., "A Functional Test Algorithm for the Register Forwarding and Pipeline Interlocking Unit in Pipelined Microprocessors," Proc. Int'l Design and Test Symp., pp. 1-6, 2012.
[6] M.Hatzimihail, M. Psarakis, D. Gizopoulos, and A. P1aschalis, "A Methodology for Detecting Performance Faults in Microprocessors via Performance Monitoring Hardware," Proc. Int'l Test Conf., pp. 1-10, 2007.

[7] P.Bernardi, L. Ciganda, M. Grosso, E. Sanchez, and M. Sonza Reorda, "A SBST Strategy to Test Microprocessors' Branch Target Buffer," Proc. Int'l Symp.Design and Diagnostics of Electronic Circuits and Systems, pp. 306-311, 2012.
[8] Y.S. Chang, S. Chakravarty, H. Hoang, N. Thorpe, and K. Wee, "Transition Tests for High Performance Microprocessors," Proc. Very Large Scale Integration (VLSI) Test Symp., pp. 29-34, 2005.
[9] G. Giles, "Is Scan (Alone) Sufficient to Test Today's Microprocessors? Not Quite, But We Can't Get the Job Done without It," Proc. Int'l Test Conf., p. 1197, 2002.
[10] A. Krstic, W.C. Lai, K.T. Cheng, L. Chen, and S. Dey, "Embedded Software-Based Self- Test for Programmable Core-Based Designs," IEEE Design and Test of Computers, vol. 19, no. 4, pp. 18-27, July/Aug. 2002.
[11] "MIHST: A Hardware Technique For Embedded Microprocessor Functional On-Line Self-Test" IEEE, Transactions On Computers,, Vol. 63, No. 11, November 2014.
[12] P. Bernardi et al., "On-Line Software-Based Self-Test of the Address Calculation Unit in RISC Processors," Proc. European Test Symp., pp. 1-6, 2012.
[13] C. Stroud, J. Sunwoo, S. Garimella, and J. Harris, "Built In Self- Test for System-on-Chip: A Case Study," Proc. Int'l Test Conf., pp. 837-846, 2004.
[14] IEEE Computer Society, "IEEE, 1149.1-2001-Standard Test Access Port and Boundary-Scan Architecture," 2001, ISBN 0-7381-2944–5.
[15] N.Kranitis, A.Paschalis, D.Gizopoulos, and G.Xenoulis, "Software- Based Self-Testing of Embedded Processors," IEEE Trans. Computers, vol. 54, no. 4, pp. 461-475, Apr. 2005.
[16] P. Bernardi, L.M. Ciganda, E.E. Sanchez, and M. Sonza Reorda, "An Effective Methodology for On-Line Testing of Embedded Microprocessors," Proc. IEEE Int'l On-Line Testing Symp., pp. 270-275, 2011.

