International Journal for Innovative Engineering and Management Research

A Peer Revieved Open Access International Journal

www.ijiemr.org

COPY RIGHT

ELSEVIER

SSRN

2019IJIEMR.Personal use of this material is permitted. Permission from IJIEMR must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works. No Reprint should be done to this paper, all copy right is authenticated to Paper Authors

IJIEMR Transactions, online available on 013th Nov 2019. Link :http://www.ijiemr.org/downloads.php?vol=Volume-08&issue=ISSUE-11

Title **SCALABLE CONTROLLER BASED PMBIST DESIGN FOR MEMORY BISTS**

Volume 08, Issue 11, Pages: 34–40.

Paper Authors

MOHAMMAD IRFAN, Y.BASAVARAJU

SRI KRISHNADEVARAYA ENGINEERING COLLEGE, GOOTY, AP, INDIA

USE THIS BARCODE TO ACCESS YOUR ONLINE PAPER

To Secure Your Paper As Per UGC Guidelines We Are Providing A Electronic Bar Code

International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org

SCALABLE CONTROLLER BASED PMBIST DESIGN FOR MEMORY BISTS

MOHAMMAD IRFAN, Y.BASAVARAJU
PG SCHOLAR.DEPT OF ECE, SRI KRISHNADEVARAYA ENGINEERING COLLEGE, GOOTY, AP, INDIA
ASSISTANT PROFESSOR, DEPT OF ECE, SRI KRISHNADEVARAYA ENGINEERING COLLEGE, GOOTY, AP,INDIA

Abstract: With increasing design complexity in modern SoC design, many memory instances with different sizes and types would be included. To test all of the memory with relatively low cost becomes an important issue. Providing user-defined pattern for screening out various manufacturing defects is also a major demand. To ease the tradeoff between the hardware cost and test flexibility, Programmable Built-In Self-Test (P-MBIST) method is an opening approach to complete the memory testing under these circumstances. Many researches have been focused on P-MBIST design. Processor-based architecture provides high test flexibility, but it increases the test development costs while applying to various processor family. To lower the design cost, a customized processor and instruction have been developed. It uses program memory to store the test program. To further reduce the hardware cost, the instruction can be serially input and saved in one internal register by adopting simple controller. However, the issue for multiinstance testing in parallel is not discussed in the literature. In this project, we implement a hardware sharing architecture to test the memory with same type in parallelism. The proposed method uses only one address counter to generate the required address for March-based algorithm, including row scan and column scan. The controller can be applied to different memory types with the same read/write cycle. Higher testing speed can be achieved by inserting two pipeline stages.Programmable Built-In Self-Test (P-MBIST) solution provides a certain degree of flexibility with reasonable hardware cost, based on the customized controller/processor. In this work, we propose a hardware sharing architecture for P-MBIST design. Through sharing the common address generator and controller, the area overhead of P-MBIST circuit can be significantly reduced. Higher testing speed can be achieved by inserting two pipeline stages. Finally, the proposed P-MBIST circuit can be automatically generated from the user-defined configuration file

1. INTRODUCTION

With increasing design complexity in modern SoC design, many memory instances with different sizes and types would be included. To test all of the memory with relatively low cost becomes an important issue. Providing user-defined pattern for screening out various manufacturing defects is also a major demand. To ease the tradeoff between the hardware cost and test flexibility, Programmable Built-In Self-Test (P-MBIST) method is an opening approach to complete the memory testing under these circumstances. Many researches have been

International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org

focused on P-MBIST design. Processor-based architecture provides high test flexibility, but it increases the test development costs while applying to various processor families. To lower the design cost, a customized processor and instruction have been developed. It uses program memory to store the test program. To further reduce the hardware cost, the instruction can be serially input and saved in one internal register by adopting simple controller. However, the issue for multi instance testing in parallel is not discussed. Hardware sharing architecture to test the memory with same type in parallelism has been proposed. The proposed method uses only one address counter to generate the required address for March-based algorithm, including row scan and column scan. The controller can be applied to different memory types with the same read/write cycle. Higher testing speed can be achieved by inserting two pipeline stages. Finally, the proposed P-MBIST engine can be automatically generated from the user-defined configuration file.

2. MEMORY BIST ARCHITECTURE

Semiconductor memories are dedicated circuits designed to store digital information, they are the most used IP in modern SoCs. Memories incorporate the greatest concentration of transistors per square area for a given semiconductor technology, pushing the chip fabrication process to its limit. Consequently, memories are more failure prone than logic. Testing such embedded memories can be a challenging task as each type of memory has peculiarities that make it more susceptible to distinct types of faults. The objective of this paper is not to discuss various fault models a memory can present but instead to show how to implement a Built-In Self-Test (BIST) Architecture that could cover them all.The focus of the discussion in this paper will be static RAM memories (SRAM), which are the most common memory type used in SoCs. The memory test is in general a well understood process [2]. It consists of identifying all physical processes that can lead a memory to malfunction; these are usually known as physical faults. The list of such faults is then mapped to observable behaviors that do not correspond to normal (expected) memory behaviors, this is called functional-faults or fault-models. Note that many different physical-faults can exhibit themselves as the same malfunction. We could think of the physical-faults as the disease while the malfunctions as the symptoms. Finally, specific sequences of operations are designed to sensitize and detect these symptoms. In most cases (except when fault analysis is intended) the reasons for the symptoms are irrelevant. It is enough to detect the misbehavior to conclude that the memory has problems.

In the case of SRAMs, the sequence of operations designed to detect faults is called a “march sequence” or “march algorithm”. Some professionals may consider the march algorithm as being the whole memory test, but for this paper, we prefer to consider a march algorithm to be only the sequences of reads and writes performed in the memory.

3. MARCH BASED ALGORITHM

Many efficient testing algorithms have been proposed to detect different fault models .

However, to implement various testing algorithms in the same P-BIST design would

International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org

require high area cost. Thus, the selection of test algorithm families should be carefully considered. Many March test algorithms[8] have been proposed to detect the above faults . We list a few important ones in Table 8.1, For word-oriented memories, the read and write operations in the March tests are extended to reading and writing a word (called the background word, background pattern, or data back- ground). For example, the word-oriented MATS++ is represented as { (wa); ⇑ (ra;wa); ⇑ (ra;wa; ra)}, where a is a background word. M

The typical memory BIST implements a March algorithm . A March test is composed of a sequence of March elements, each one corresponding to a series of read/write operations to be sequentially performed on the memory cells. The March elements are executed in ascending or descending address order (or undetermined). The operations can be writing or reading of the 0 and 1 values, e.g., w0, w1, r0 and r1. An example of the common March Test notation follows (complexity 4n, where n is the number of memory addresses):

{⇕(w0);↑ (r0, w1);↓(r1)}

Different approaches have been proposed in the literature in order to implement BIST- based March test algorithms.

The hardwired BIST approach is the most widely used. It consists in adding a custom circuitry to each core, implementing a suitable BIST algorithm . The main advantages of this approach are that the test application time is short and the area overhead is relatively small. Hardwired BIST is also a good way to protect the intellectual property contained in the core: the memory core provider needs only to deliver the BIST activation and response commands for testing the core without disclosing its internal design. At the same time, this approach provides very low flexibility: any modification to the test algorithm requires redesigning the BIST circuitry.The soft BIST approach assumes that a processor is already present in the SoC: the processor is exploited to run a program that performs the test of the other cores. The test program executed by the processor applies test patterns to each core under test and checks the results; it is stored in a memory containing the test patterns, also. This approach uses the system bus for applying test patterns and reading test responses, and it guarantees a very low area overhead, limited to the chip-level test infrastructure. The disadvantage of this approach is mainly related to the strict dependence of the test program on the available processor. As a result, the core vendor needs to develop for the same core different test programs, one for each processor family, thus increasing the test development costs. Moreover, intellectual property is not well protected, as the core vendor supplies to the user the test program for the core under test. Finally, this approach can be applied only to cores directly connected to the system bus; the approach cannot be applied if the core is not completely controllable and observable.

An alternative approach is the one usually denoted as programmable BIST. The core vendor develops a DFT logic, which wraps the core under test and includes a custom processor, which is exclusively devoted to test the core. The advantages of

International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org

this architecture are manifold the intellectual property can be protected, only one test program has to be developed, and the design cost for the test is very reduced; the technique provides high flexibility since any modification of the algorithm simply requires a change in the test program; the test application time can be taken under control thanks to the efficiency of the custom test processor, and the test can consequently be executed at-speed. Finally, each core is autonomous even from the test point of view, and its test only requires activating the test procedure and reading the results, as for hardwired BIST.

March-based algorithm[8] is an important class to detect a large variety of faults . A March test consists of a finite sequence of March elements. Each element performs a series of “reading” and “writing” operations to all memory cells. The addressing order of each element is executed in ascending , descending , or either ( ) way. It contains three March elements. The first element performs writing 0 to each memory cell in either addressing order. The second element performs two operations to each address in ascending order.

Finally, the third element reads all the address in descending order. The complexity order is noted to 4N, where N is the number of memory addresses.

Test sequence (4N): { (w0), ↑(r0, ↓(r1)}

Beside, based on the March-based test sequence, the dynamic faults can be tested by repeatly performing the same operation in the same memory cell. To efficiently implement March-based algorithm, we define the corresponding instruction for P-MBIST architecture. The advantage is that one can program the instruction to modify the testing algorithm during run time. In our shared architecture, only one address generator is required for testing all memory. Also, different memory types with the same read/write cycle can share the same controller.

4. P-MBIST ARCHITECTURE

To support flexible March elements and diagnostic procedures, the proposed P-MBIST[2] provides the following features: (1) different data background, (2) programmable March-based element, (3) column-scan addressing mode, (4) capturing the address, syndrome and state of failures, (5) parallel and sequential test, (6) pipeline mode for higher test speed. Table 9.I lists the detailed programmable items of instructions. The “folded” and “inverted” bits can invert the “data_bg” while the switching of the specified address bit. By setting the “data_bg” to 0 and configuring both “folded” and “interted” bits to 1, the Checkerboard pattern can be generated. Column scan addressing can be controlled by cs_sel. Finally, each March operation description uses three bits, EOC, R/W, and Polarity, to define the operation. Fig. 10.1 gives an instruction example of performing three operations in a March element. By setting up the “diagnosis” bit to 1, the failure information will be dumped out while dectecting the mismatching memory output.

Fig. 1 shows the block diagram of the proposed P-MBIST design[2]. Firstly, the instruction is serially shifted into instruction_read module. To reduce the hardware cost, the address counter is shared to test all memory instances. Moreover, decoder blocks can support different

International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org

memory type testing without increasing any state (i.e. each memory has the same read/write cycle). The memory in the same type can be grouped and tested in the either sequential or parallel way. The supported memory types include single/dual/twoport SRAM, one/two-port register file and ROM. Finally, according to the user-defined configuration file, the proposed P-MBIST engine can be automatically generated from the developed software program.

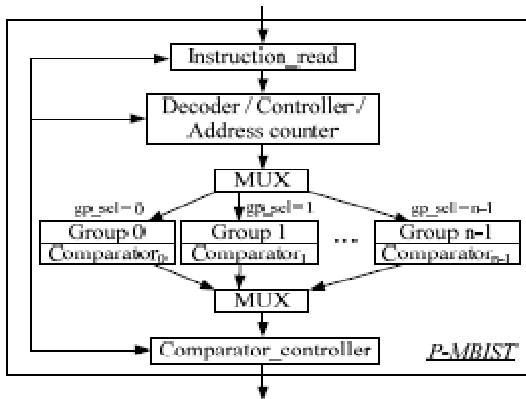

Fig. 1 Block diagram of P-MBIST design

5. ADDRESS GENERATOR

To share the common address generator[2], it is a simple way to choose an address counter with the largest address bit among all the under-tested memory instances. As shown in Fig. 10.3, the low-bit of address counter can be used to indicate the low-address memory instance, by properly controlling the chip-select signal for each memory instance.

As for the column-scan mode, the address counter is not just ascending or descending by 1. Table 10.2 provides an example, which defines the column Mux to 4 and the maximum address to 11. In Table 10.2, the 1st column shows the address counter output for row scan (i.e. ascending by 1). The 2nd column gives the address for column scan (i.e. ascending by 4). To share the row-scan and column-scan adder, we use the up-scrambler to adjust the added bit for column-scan mode. As shown in the 3rd column, the low bit of A[1:0] is shifted to the position of most significant bit, which means the row adder can directly apply to the A[2] bit. Similarly, after the process of down-scrambler, the column scan address for ascending by 4 can be generated. By adding two scrambler blocks, the row adder can support the column addressing.To continuously perform the operation to each address in column-scan way, it encounters two special cases due to overflow of address-up/-down counting. Table 10.3 lists an example for 8-bit address, where the column Mux is 4 and the maximal address is 199. In the up-counting case, if the column-scan addressing exceeds the maximum address, the word-line address is reset to 0 and the bit-line address is switched to the next column cell. For the down-counting case, the maximum word-line address should be set to hi-bit part to deal with the overflow case. The overall block diagram of address generator is shown in Fig. 10.5. The path of column scan and row scan are highlighted. Finally, by the control signals, fail_h and EOC, address counter is held while dumping failure data (i.e. fail_h) and executing continuous operations in the same March element (i.e. EOC). A built-in look-up table is required to store the maximal address and column MUX number of all under-tested memories.

International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org

6. SIMULATION RESULTS

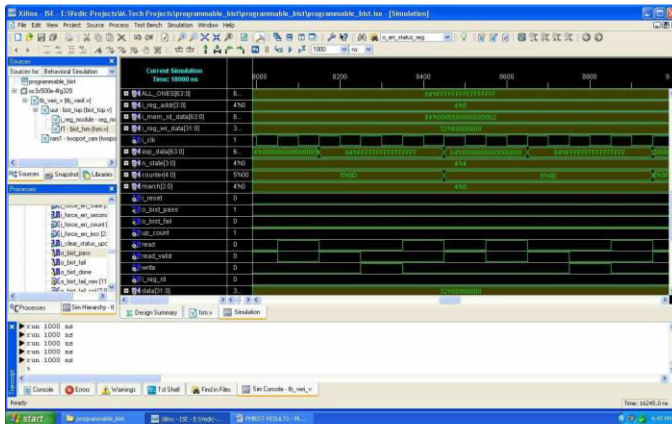

Fig. 11.1 Simulation Result

7. CONCLUSION AND FUTURE SCOPE

In this work, an efficient architecture for PMBIST circuit has been designed based on the hardware sharing method, the common address counter has been used to provide to provide all memory instances, including row-scan and column-scan addressing mode. Moreover, the controller can be extended to different memory types in the same read/write cycle condition, without increasing any state. Thus, the hardware cost can be greatly reduced. Finally, the P- MBIST circuit can be automatically generated from the user-defined configuration file.We can extend this work to implement Programmable Memory Built-In Self Test with less hardware and with minimum number of read and write operations. To the extend this work we can

REFERENCES

[1] D. Appello, V. Tancorre, P. Bernardi, M. Grosso, M. Rebaudengo and M.S. Reorda, "Embedded Memory Diagnosis: An Industrial Workflow," in Proc. ITC, pp:1 - 9, Oct. 2006.

[2] I. Bayraktaroglu, O. Caty and W. Yickkei, "Highly configurable programmable built-in self test architecture for high-speed memories," in Proc. VLSI Test Symposium, pp:21 - 26, May 2005.

[3] D. Xiaogang, N. Mukherjee, C. Wu-Tung; S.M. Reddy, "Fullspeed field-programmable memory BIST architecture," in Proc. ITC, pp:45.3, Nov. 2005.

[4] S. Boutobza, M. Nicolaidis, K.M. Lamara and A. Costa, "Programmable memory BIST, " in Proc. ITC, pp:45.2, Nov. 2005.

[5] S. Boutobza, M. Nicolaidis, K.M. Lamara and A. Costa, "A Transparent based Programmable Memory BIST," in Proc. IEEE European Test Symposium, pp. 89 - 96, May 2006.

[6] M. Miyazaki, T. Yoneda and H. Fujiwara, "A memory groupingmethod for sharing memory BIST logic," in Proc. Asia and South Pacific Design Automation Conference, pp. 671-676, Jan. 2006.

[7] R.S Sable, R.P. Saraf, R.A. Parekhji and A.N. Chandorkar, "Builtin self-test technique for selective detection of neighbourhood pattern sensitive faults in memories," in Proc. VLSI Design conference, pp. 753 - 756, 2004.

[8] C.F. Wu, C.T. Huang and C.W. Wu, "RAMSES: a fast memory fault simulator," in Proc. Defect and Fault Tolerance in VLSI Systems Symposium, pp. 165 - 173, Nov. 1999.

[9] A.J. Van de Goor and J. de Neef "Industrial Evaluation of DRAM Tests," in Proc. Design, Automation Test in Europe, pp. 623-631,\ March 1999.

[10] Chung-Fu Lin, Chia-Fu Huang, De-Chung Lu, Chih-Chiang Hsu, “A Low-Cost

Programmable Memory BIST Design for Multiple Memory Instances" INTERNATIONAL TEST CONFERENCE 2008 IEEE.
[11] Po-Chang Tsai, Sying-Jyan Wang and Feng-Ming Chang, "FSM-Based Programmable Memory BIST with Macro Command" Proceedings of the 2005 IEEE International Workshop on Memory Technology, Design, and Testing (MTDT'05)