Built in self test for RAM Using VHDL

M.H. Husin, S.Y. Leong, M.F.M. Sabri, R. Nordiana
Faculty of Engineering
Universiti Malaysia Sarawak
94300 Kota Samarahan, Sarawak
hhmaimun@feng.unimas.my

Abstract— **This project emphasized mainly on software analysis. Modelsim-Altera 6.4a is the software that used to generate every single module of the Built-in-Self-Test (BIST) for Random access Memory (RAM) architecture. There are three key things to be concern in the BIST for RAM which is the Test Pattern Generator (TPG), Output Response Analysis (ORA) and RAM. The output of counter which is a type of TPG is analyzed to provide a pattern for March test algorithm. At the mean time, the ORA compare the output from decoder and the RAM output itself which modeled under the theory of numerical autonomy of error vectors from the circuit under test. The output of ORA, the comparator, will show pass or fail for faulty detection of RAM. The system has been successfully developed and vector waveform is used to examine the result of the system. From the result obtained, it showed that the system is working as expected with satisfactory result.**

Keywords-component; BIST, RAM, TPG, ORA

I. INTRODUCTION

For any kind of system, testing and fault diagnosis play a significant role to identify unwanted defects. There are some complicated applications have severe cost constraint but still require high-level of performance and safety. Hence, the utmost concern is where the diagnosis of the defects has to be done even under inadequate resource and time. An approach to solve this dilemma, namely BIST, has been widely implemented in semiconductor industry. With a built in test system positioned inside the circuit, the analysis of every single part within the circuit could become simple task as the period and cost for testing are cut down. This BIST technology is capable of saving the time and cost of maintenance that also allow on line diagnosis, which can deal with even greater advance of embedded systems in future.

The objective of this project is to develop a BIST for RAM by using VHSIC Hardware Description Language (VHDL).

II. LITERATURE REVIEW

A. General BIST architecture

BIST technique can be categorized into several general architectures and often classified to be centralized or distributed [1]. These two basic BIST architectures are illustrated in Fig. 1.

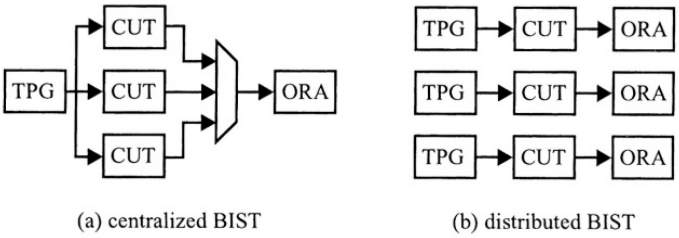

Figure 1. (a) centralized BIST and (b) distributed BIST architecture [1]

B. TPG

Fault models produce a direct function which is the fault coverage of the test patterns generated by TGP and apply to Circuit Under Test (CUT). There are several classes of test patterns, e.g. Deterministic test patterns, Algorithmic test patterns, Exhaustive test patterns, Pseudo-exhaustive test patterns, Pseudo-random test patterns, Weighted pseudo-random test patterns and Random test patterns. As a result, TPGs are sometimes classified according to the class of test patterns they produce. Fig. 2 and Fig. 3 show the example of TGP.

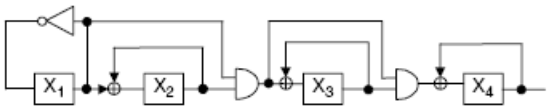

Figure 2. Example of Binary Counter [2]

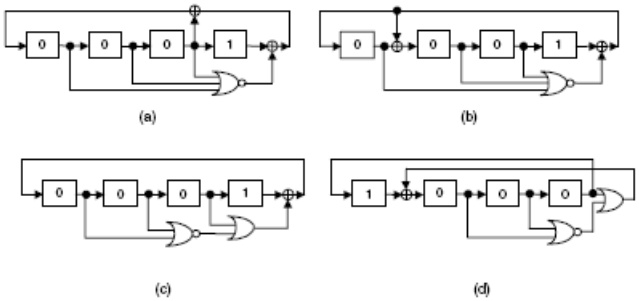

Figure 3. Example of LSFR [2]

C. ORA

In most ORA techniques, the output responses of the CUT to the test patterns are "compacted" into a "signature" which is compared to the expected signature for the fault-free circuit. As we shall see, the Pass/Fail indication is often composed of a set of data bits that contain the signature of the BIST sequence, rather than a single Pass/ Fail bit. Note that the term compaction is used rather than compression since compression implies no loss of information; since most ORA techniques incur some loss of information; compaction is a more accurate term [3].

Since all ORAs perform data compaction, there will be some loss of information. When the lost information contains fault detection data, it is possible for the BIST results to indicate that a faulty CUT is fault-free. This is often referred to as fault masking or, in the cases of some ORA techniques, signature aliasing [3].

D. RAM BIST architecture

There are a few of test algorithms for RAM testing that are rather simple for BIST implementations include variations of the Modified Algorithmic Test Sequence (MATS) and March tests including the March Y algorithm that used for the FSM-based TPG [4].

Among these test algorithms, the March C- algorithm is practical to offer the highest fault coverage [4]. For detail, the capabilities of the fault detection of March test algorithms are summarized in Table 1. The test algorithms are planned for RAMs with one data bit per word. Nonetheless, multiple bits per word can be applied as well. In order to increase the fault detection rate, modification on the algorithm is required for sensitivity and coupling faults in RAM [5].

TABLE 1: THE SUMMARIZED FAULT DETECTION OF MARCH TEST ALGORITHMS [5]

	Faults Detected			
Algorithm	**Stuck-at**	**Address**	**Transition**	**Coupling**
MATS	yes	some	no	no
MATS+	yes	yes	no	no
MATS++	yes	yes	yes	no
March X	yes	yes	yes	some
March Y	yes	yes	yes	some
March C-	yes	yes	yes	yes

III. METHODOLOGY

The test begins starting with the BIST_Controller that gives input and determine the circuitry either in BIST mode or normal mode. As the counter receives input from BIST_Controller, it starts to count from all 0s until reaches all 1s. The test pattern generated by counter is written into memory starting from 0s that stored in location 0. The counter increments when the same pattern is stored into all RAM location and cause the LSB of the address to turn over to all 0s. The RWbar signal becomes '1' at the same time when the same data being read from all memory location and the address loop back at 0. Then, the next test pattern begins to be written into all locations. This process iterate for the eight test patterns generated by decoder. Fig. 4 shows the block diagram of BIST for RAM.

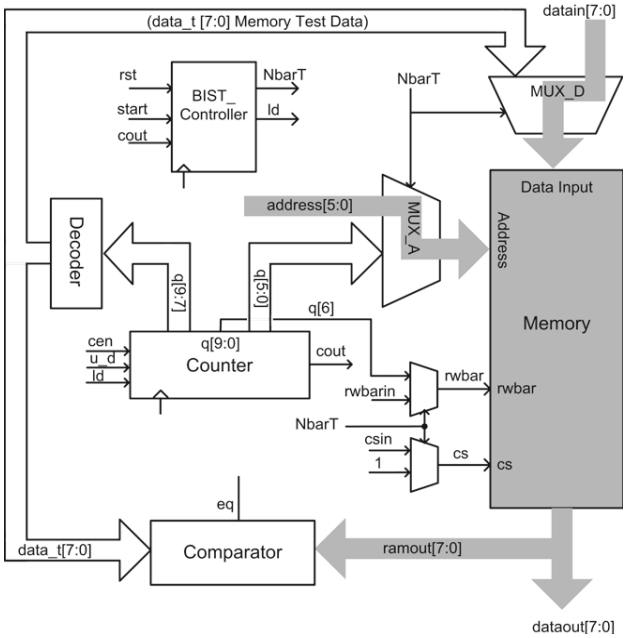

Figure 4: Block diagram of BIST for RAM

A. Counter

The address, input data and switching ability between writing and reading the memory are supplied by a counter. The address for all locations of the memory is provided by the least significant bits of the counter. The counter bit to the left of the address group of bits toggles between write and read operations. The eight test vector is generated by the decoded three most significant bits of the counter. The carry out of counter shows '1' when the count achieves its maximum.

B. Decoder

A combinational circuit that converts binary information from n input lines to a maximum of 2^n unique output lines. Decoder receives 3 bits input from Counter and produces an 8 bits memory test data to RAM and Comparator. There are a total of 8 test pattern used for memory testing begin with a sequence from "0000 0000" that will be discussed in the following chapter.

C. Comparator

A comparator is designed to checks the memory data with decoder output. A set of same test pattern is written into all memory locations initially. Then, the stored data is read out from all memory locations. The test patterns remain unchanged as the data are being written and read the decoder input. As a result, the comparator should have same data on both of its inputs when the memory is being tested.

D. BIST Controller

The BIST controller directs the control signals of counter and multiplexer. This means BIST controller determine the circuitry in either BIST mode or in normal mode.

E. BIST Test Bench

After that, a test bench is built to verify the design of the previous work in a good condition. In any RTL design, test bench plays an important role in design validation and verification. Test bench is normally created to model or enumerate the components around the designs. Design needs to "communicate" and handshake with the other components around it to form complete system or platform. In this project, the external file data is loaded into RAM at 5ns when *opr* is TRUE. There are some random times where data is read from or written into the RAM.

F. RAM

The memory consists of dump, write and read operations. TEXTIO package is used for read and write function. The file type used by this method is in TEXT pattern. When *opr* becomes true, *init_mem* is called to initialize the memory array. Else, *dump_mem* is called for write operation. The *init_mem* function reads the memory location by using size of the second index from its data file and put the vector data into the memory location one bit at a time. As for writing operation, the *dump_mem* function reads data bits by bits from the memory location into *stdvalue* variable and writes them into a buffer. When all bits of word are written, the buffer that is written to the external file is called.

IV. RESULT

In the software analysis, the design and test bench for the sub module of BIST for RAM is obtained to show the functionality for every sub module includes counter, decoder, comparator, multiplexer, BIST controller, RAM and BIST. Simulation waveforms are shown in every sub module together with the tabulated simulation results.

A. Counter Module

The RAM test pattern is generated by counter as shown in Fig. 5. The output obtained when the inputs are manipulated as desired. The waveform works only with the condition where the clock is at positive edge and the counter is enabled. When the input u_d is enabled, the counter begins to increment until it reaches the its maximum.

B. Decoder

Fig. 6 shows the decoder memory test patterns when the 3 bits input vector is given. There are a total of 8 simple March test pattern generated including "0000 0000", "0000 1111", "0011 0011", "0101 0101", "1111 1111" and "1010 1010".

C. Multiplexer

Fig. 7 shows the vector waveform of multiplexer. The in1 of the multiplexer is set to "1111" where in2 is set to "0000". When the select of multiplexer is in low state, in1 will be directed to output. Else, in2 will be shown as output.

D. Comparator

Fig. 8 shows 3 outputs of comparator with 3 conditions. The output "eq" shows '1' when input a is equal to input b. The second condition is when input a is greater than input b thus giving '1' in output gt. The last output shows output lt in high state when input a is less than input b.

E. RAM

Fig. 9 shows the simulation vector waveform of RAM. When the *opr* is true, the memory data is called from *memdata.dat* file. Else way, the memory data is called from *memdump.dat* file. RAM only giving output when the chip select is in '1'. The *rwbar* is set as always '1' in order to read the memory. The RAM is designed with input and output as 8 bit vectors, the address as 6 bit vectors. By relating the port giving that the fact where the memory consists of 32 8 bit words.

F. BIST

Fig. 10 shows the simulation vector waveform of RAM BIST. The test session is initiated at 50ns where start is in high state. The test ends when all location of RAM is tested. When the memory testing operation is carried out, the other operation such as read and write are ignored.

V. CONCLUSION

As a brief conclusion, a random access memory with built-in-self-test has been successfully designed. The objective of studying the DFT method is achieved by reviewing various techniques such as built in self test and external test. The BIST techniques also divided into online and offline testing. This project is successfully implemented with the online testing via coding.

This project is an entry level design of VHDL digital system for simple BIST for RAM. Hence, there are a lot of modifications and additional architectures can be added to increase the robustness of the design.

i. A March C- algorithm can be implemented into the design by modifying the decoder part. This is because March C- algorithm provides better fault coverage compare to the other test algorithm.

ii. The BIST design covers only 8 bits of input and output of memory which can be increase the bit vector so that more memory location can be tested.

iii. BIST is able to scan the golden signature of RAM with no faulty but not able to insert the fault model such as stuck at fault. Future works can be done to insert and detect a set of fault models.

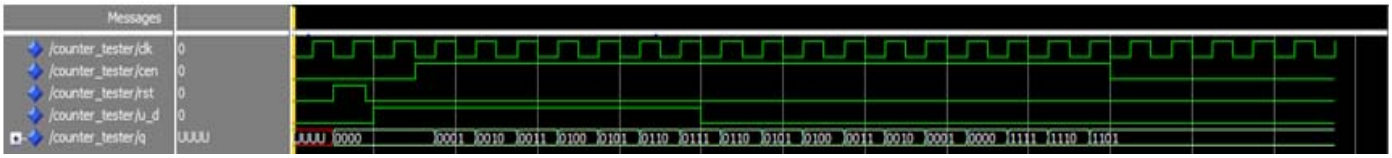

Figure 5. Counter simulation vector waveform

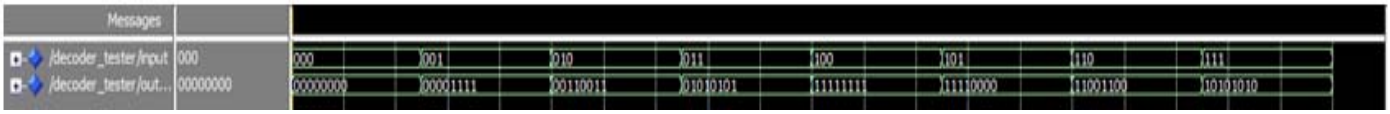

Figure 6. Decoder simulation vector waveform

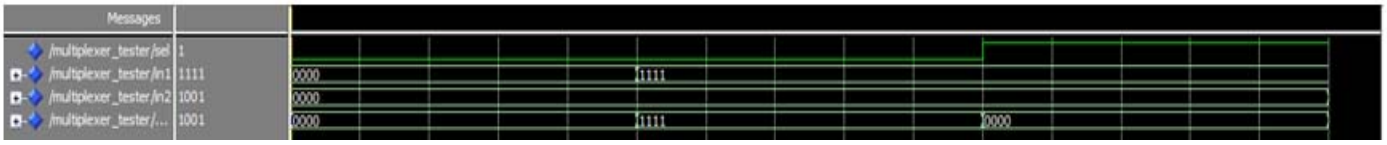

Figure 7. Multiplexer simulation vector waveform

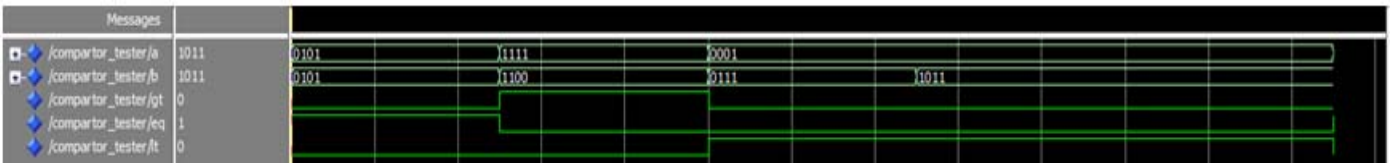

Figure 8. Comparator simulation vector waveform

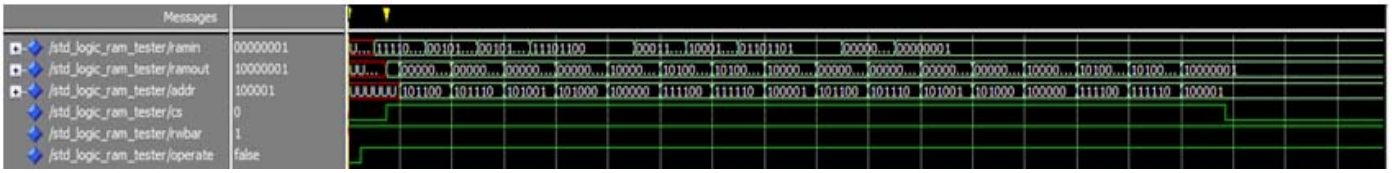

Figure 9. RAM simulation vector waveform

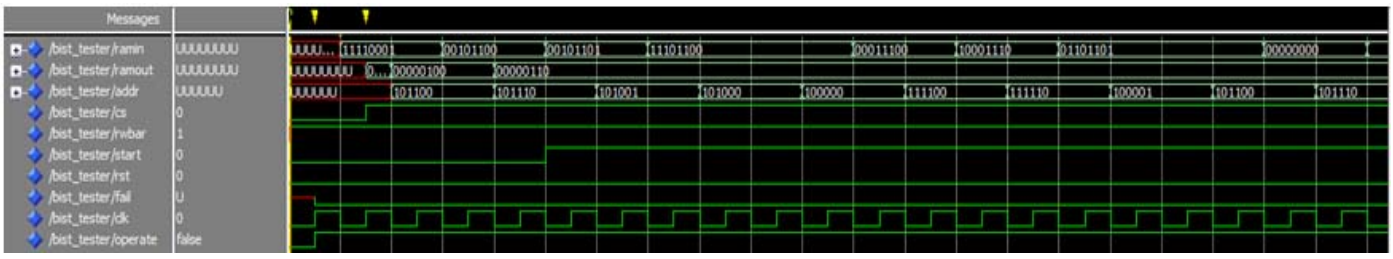

Figure 10. BIST simulation vector waveform

iv. The comparator and counter of BIST design can be replace by MISR and LFSR for smaller hardware coverage as well as giving better performance when CUT consist of larger area.

v. VHDL can be replaced with Verilog HDL for lesser command line for future work.

vi. Due to resource and time constraint, this project is not able to program into actual FPGA hardware device. Future works can be done to run the testing via FPGA hardware device to verify the VHDL design.

ACKNOWLEDGMENT

The author wishes to thank Universiti Malaysia Sarawak for supporting the dissemination of this project.

REFERENCES

[1] M. Abramovici, M. Breuer and A. Friedman, Digital Systems Testing and Testable Design, Piscataway, New Jersey: IEEE Press, 1994.

[2] Stroud, Charles E., "A Designer's Guide To Built-in Self-test". Springer, p. 61, 2004.

[3] E. McCluskey, "Built-In Self-Test Techniques," IEEE Design & Test of Computers,Vol. 2, No. 2, pp. 21-28, March 1985.

[4] P. Veenstra, F. Beenker and J. Koomen, "Testing Random Access Memories"

[5] A. van de Goor and I. Tlili, "March Tests for Word-Oriented Memories," Proc.Design Automation and Test in Europe, 1998, pp. 501-508. Feb.2001.