Volume : 3 | Issue : 5 | May 2014 • ISSN No 2277 - 8179

Research Paper

A VLSI Implementation of Word Memories, by Using BIST&BIRA

Engineering

KEYWORDS : BIST, BISR, BISD, BIRA, SOC, CRESTA

Reshmi.R.Das M-Tech VLSI Design, Vedavyasa Institute of Technology, Karadparamba, Malappuram

ABSTRACT

This paper describes the technology of BIST, which is implemented in 3 different circuits. Technologies are changing day by day. All the functions are implemented in a single chip. As the size reduces by increasing the number of functions testing becomes more and more complicated. One of the promising solution to this is Built In Self Test (BIST).Here a VLSI implementation of built in repair analyzer (BIRA) to word oriented memories is done as a first phase. This is done in both 32 &64 bits. Both BIST and (BISR) Built in Self Repair is implemented in it. It does not need automatic test equipments (ATE).With the trend of SOC technology; good embedded memories are needed i.e. with high density and high capacity for successful implementation of system products. This paper is implemented in VHDL HDL.Simulation and synthesis is done using Modelsim&XilinxISE8.1 tools.

I.INTRODUCTION:

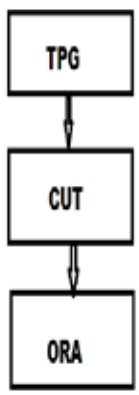
A SOC that integrates all the components into a single chip. Testing a soc is very much difficult. One of the greatest barrier is achieving good manufacturing yeild.integrating number of memory bits per chip create powerful SOC.Which is mainly used for large memory hungry applications. A main disadvantage of that is large die size and poor manufacturing yield .in 2001International Technology Roadmap for Semiconductors (ITRS) saying that embedded memories are occupying from 52 % to 94%by 2014.[1]so if any fault occurs the whole die is wasted. In early times the size of the system is large. But the trend now is reduced size with all applications. So testing embedded memory is an important fact. Thus for testing embedded memories we are implementing BIST.not only BIST, after that diagnosing the fault and repairing it with reconfiguration. This is done in word memories. The flow is as follows in Fig.1

Fig.1.Flow of BIST to BISR

BIST is a design for testability technique, that automatically test the Circuit under Test (CUT).The basic BIST architecture is given bellow. [2]. In Fig.2 there are 3 main parts. Test pattern generators, a response analyzer, and a test controller (output response analyzer). The test pattern generator generates the test patterns for the CUT. A response analyzer is a comparator with stored responses or a Linear Feedback Shift Register (LFSR) used as a reference (signature) analyzer. It compresses and analyzes the test responses and determines the correctness of the CUT. [3]. A control block is must to activate the test and analyze the responses. However, in general, many test-related functions can be simulated through a test controller circuit. In the most common type of BIST, test responses are compressed in output response compactor to form (fault) signatures. The response signatures are compared with reference signatures generated and the error signal indicates whether chip is good or faulty.BIST reduces testing and maintenance cost because it requires simpler and less expensive ATE. Also it reduces cost of automatic test pattern generation (ATPG), storage and maintenance of test patterns. It can test many units in parallel. It takes very short time to test CUT .It can test at functional system speed. On-line BIST is usually implemented with the twin goals of complete fault coverage and low fault latency. These are Advantages of BIST. Test generation (TG) and response monitor (RM) are often implemented by simple, counter circuits, especially linear-feedback shift registers (LFSRs).

International Conference on Emerging Trends in VLSI technology and Systems (ICETVTS'14) April 30, 2014

Fig.2.BIST architecture



The LFSR is simply a shift register formed from standard flip-flops. The outputs of selected flip-flops being fed back (modulo-2) to the shift register's inputs. When used as a TG, an LFSR is set to cycle rapidly through a large number of its states. These states, whose choice and order depend on the design parameters of the LFSR, define the test patterns. In this mode of operation, an LFSR is seen as a source of (pseudo) random tests, applicable to any fault types. An LFSR can also serve as an RM by counting the responses produced by the tests. An LFSR RM's final contents after applying a sequence of test responses forms a fault signature, which can be compared to a known or generated good signature, to see if a fault is present [4].

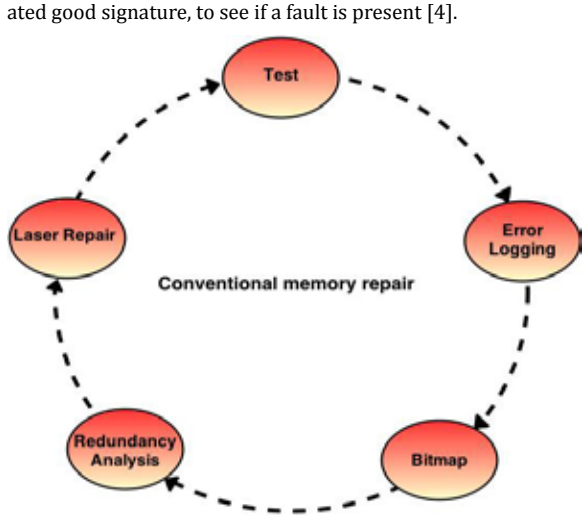


Fig.3.Conventional memory repair.

Not only diagnose the fault it also must be repaired. Conventional Memory Repair Flow is as follows in Fig.3.Repair is one popular technique for memory yield improvement yield improvement. Memory repair consists of three basic steps Test

Volume : 3 | Issue : 5 | May 2014 • ISSN No 2277 - 8179

Redundancy analysis, Redundancy analysis and Repair delivery [5]. This conventional memory repair is replaced in to memory BISR.By changing in to this test time reduces. The memory BISR flow is shown below in Fig.4.the functions of each module are listed by side. Built in redundancy analyzer is done (BIRA) as a part of BISR.

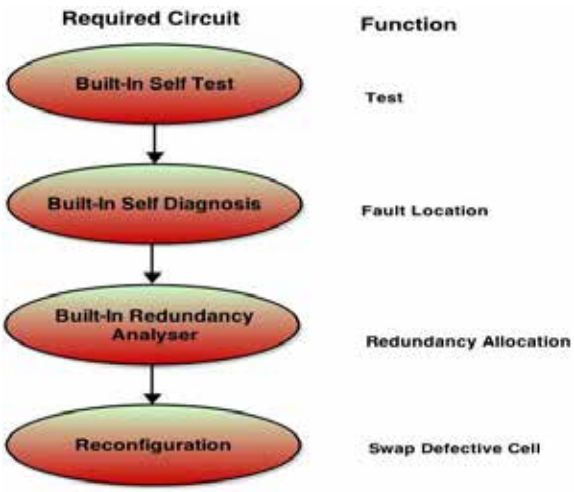


Fig.4. Memory BISR flow.

II.WORDMEMORIES:

Designed a BIST and BISR for word memories. Here the memory used is CAM Memory and used a BIRA approach. The approach name is CRESTA (Comprehensive Real Time Exhaustive Test search Analysis) [6].

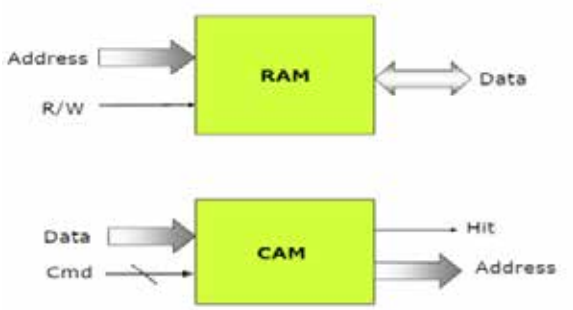


Fig.5.CAM vs. RAM

The difference between a CAM memory and a RAM memory is in RAM we are giving address as input we will get the corresponding data in that address. But in a CAM we are giving data as input and will get address. Location of that data. Hence its name is CAM (Content Addressable memory). The CRESTA which consists of several parallel sub analyzers.

$$\text{repair rate} = \frac{\#\text{ of repaired chips}}{\#\text{ of total bad chips}}$$

$$\text{Normalised repair rate} = \frac{\#\text{ of repaired chips}}{\#\text{ of repairable chips}}$$

Each one searches for a solution. The sub analyzer consists of CAM with r rows and c columns.CRESTA requires a total number of sub analyzer. Each CAM consists of r row entries and c column entries. The total number of CAM entries is given by .this is done in both for a single word (32 bit) &64 bit memories.

III.PROPOSED METHOD

In the classical spare allocation, we consider a bit oriented memory array with r spare (repair) rows and c spare (repair) columns. Any fault row (column) can be replaced with a spare row (column). A repair solution is a set of at most row addresses and at most column addresses that cover all faults. If a repair solution exists for a memory array, the memory array is repairable. A repair strategy is a string of the alphabet {"R", "C"} such that R occurs r times and C occurs c times. Thus there are repair strategies. For example, if r=3 and c=3, the set of all possible repair strategies is{"RRRCCC ","CCCRRR" ,"RRCCCR ","RCCCRR ","CCRRRC","CRRRCC" }.Given a sequence of fault addresses, a repair strategy can generate a repair solution candidate and the solution space consists of Solution candidates. For example, given a sequence of fault addresses {(R1, C2), (R5, C8), (R4, C9), (R7, C2)}.A repair strategy RRCC generates a solution {(R1, R5, C9, C2)} [7]. Fig.6.shows the infrastructure for bit-oriented memories. For extending in to word memories we are modifying the must repair analyzer.In this which has a BIST module, a SOLVER module, CAM memory, registers etc.it consists of must repair analysis and final analysis. BIST engine tests a memory array and provides fault addresses whenever detected. The must-repair analysis is performed concurrently with the test, while the final analysis is done after the test is completed. The must repair analyzer (MRA) is shown in Fig.7. The MRA consists of a pair of CAMs for fault addresses, called the fault-list, and a pair of CAMs for a repair solution, called the solution record. In the fault-list, each CAM has one extra valid bit for each word, and the valid bits are initialized to "0" in the beginning. [8]. During the test, if the BIST engine detects a fault, it sends the fault address to the MRA through row _add and col_add, and continues the test. The row (column) fault address is compared against row (column) CAM entries, and the number of matched entries is counted by a parallel counter

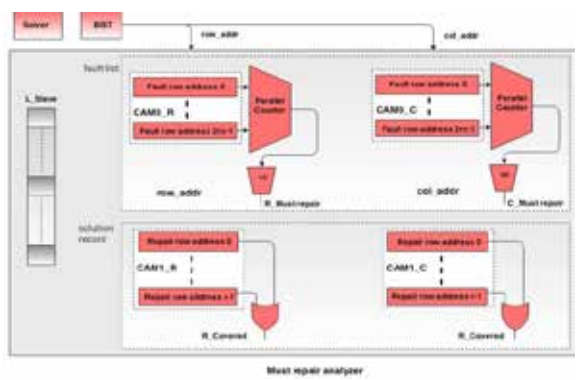


Fig.6. on-chip infrastructure for bit-oriented memories

The L registers are used as valid bits for the solution record and also determine the next available CAM entry. The SOLVER includes the address, and faults on the address do not affect the final analysis. Therefore, such faults do not need to be stored, and we can collect all necessary information for the final analysis [9] [10]. The SOLVER module controls the MRA. The operation of the SOLVER and the MRA in the final analysis phase is illustrated in Fig.7

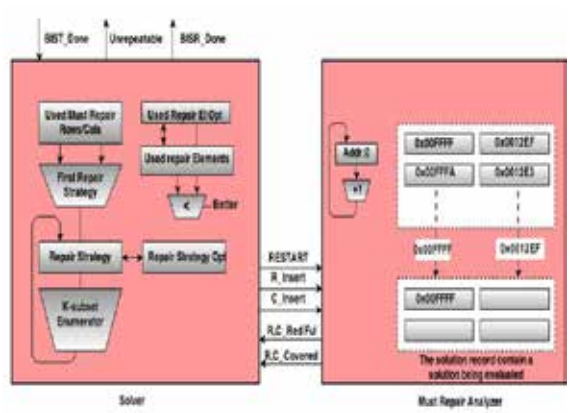


Fig.7.SOLVER and the MRA in the final analysis phase

The SOLVER will generate repair strategies one by one .If "R" and "C" are mapped to "1" and "0", respectively. The SOLVER generates the first repair strategy and the MRA reads each fault address in the fault-list in order until the RESTART signal is arrived or no faults ahead The SOLVER generates the next repair strategy and asserts the RESTART signal. The next repair strategy can be generated directly from the current repair strategyby a combinational circuit called K-subset enumerator.

Repair strategy	Bit representation
RRCC	1100
RCRC	1010
RCCR	1001
CRRC	0110
CRCR	0101
CCRR	0011

TABLE 1

Bit Representations.
We can define a cost function the cost is defined as the number of used spare elements the current cost is compared against the minimum cost which generates the Better signal. If the Better signal goes down during the evaluation, the SOLVER immediately asserts the RESTART signal and moves on to the next repair strategy. If the Better signal stays at "1" until the end of the evaluation, the SOLVER saves the current repair strategy and its cost [11] [1 2] [13].

V1. RESULTS
Here 32 bit constitute a word i.e. a single word. Time taken for transmitting a single word is about 2408ns.when read operation matches write operation there is no fault but if it fails it indicates corresponding location as fault. Here the address location is indicated by fault row column address. That error is repaired by using must repair analyzer

And corrected. Without asking additional time. The primary change between 64 bit and 32 bit is the width of the address field, which has increased to 8 bytes from 4 bytes. So, evidently, more the number of address fields, pointers in our application, more is the memory consumption in 64 bit. Here 64 bit constitute 2 word i.e. time taken for transmitting a single word is about 5000 ns.when read operation matches write operation there is no fault. But if it fails it indicates corresponding location as fault. Here the address location is indicated by fault row column address. That error is repaired by using must repair analyzer and corrected. Without taking additional time. Compared to 32 bit the band width increased. The amount of data transmitted also increased. This can be used for large memory storage applications. No wastage of memory space.

VII.CONCLUSION
In this work importance of testing is done .with the increase of submicron technology the number of transistors is increasing day by day. Here word memory, for word memories BISR was also done, by BIRA approach. We can use a very powerful cam memory .which can be done in both 32 bit and a 64 bit memory. For example 64 bit can be implemented in laptops.32 bit can be implemented in computers. For 64 bit wide amount of data can be stored and transmitted. Bandwidth is more when compared to 32 bit. In UART which has normal mode and testing mode. In testing mode it has different sub modes. If any fault or mismatch occurs BIST module can be easily detected. The LFSR replaces the function of an external tester and give 100 percent fault coverage. Like that in first in first out, if it does not occur correctly fault appears. As a future enhancement we can implement BIRA&BIST to both UART &FIFO.Thus BIST is one of the promising solutions to testing and achieving good yield thus productivity increases.

REFERENCE

[1] Sehgal, A. Dubey, E. Marinissen, C. Wouters, H. Vranken, and K. Chakrabarty, "Redundancy modelling and array yield analysis for repairable embedded memories," IEE Proc. Comput. Digit.Techn., vol. 152, no. 1, pp. 97–106, 2005. | [2] D. K. Bhavsar, "An algorithm for row-column self-repair of rams and its implementation in the alpha 21264," in Proc. Int. Test Conf., 1999, | pp. 311–318. | [3] X. Du and W.-T. Cheng, "At-speed built-in self-repair analyzer for embedded word-oriented memories," in Proc. Int. Conf. VLSI Design,2004, pp. 895–900. | [4] S.-Y. KuoandW. Fuchs, "Efficient spare allocation for reconfigurable arrays," IEEE Design Test Comput., vol. 4, no. 1, pp. 24–31, Feb. 1987 | [5] P. Oehler, S. Hellebrand, and H.-H.Wunderlich, "An integrated built-in test and repair approach for memories with 2D redundancy,"inProc. Eur. Test Symp., 2007, pp. 91–96. | [6]P. Oehler, S. Hellebrand, and H.-J. Wunderlich, "Analyzing test and repair times for 2D integrated memory built-in test and repair," in Proc. Design Diag. Electron. Circuits Syst., 2007, pp. 1–6 | [7] W. Jeong, J. Lee, T. Han, K. Lee, and S. Kang, "An advanced BIRA for memories with an optimal repair rate and fast analysis speed by using a branch analyzer," IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., vol. 29, no. 12, pp. 2014–2026, Dec. 2010. | [8] C.-T. Huang, C.-F.Wu, J.-F.Li, and C.-W. Wu, "Built-in redundancy analysis formemory yield improvement," IEEE Trans. Reliab., vol. 52, no. 4, pp. 386–399, Dec. 2003. | [9] A.Ferris and G. Work, "Memory circuit capable of replacing a faulty column with a spare column," U.S. Patent 5 163 023, Nov. 10, 1992. | [10] B. Fitzgerald and E. Thoma, "Circuit implementation of fusible redundant addresses on RAMs for productivity enhancement," IBM J. Res. Develop., vol. 24, no. 3, pp. 291–298, 1980. | [11] R. Rajsuman, "Design and test of large embedded memories: Anoverview," IEEE Design Test Comput., vol. 18, no. 3, pp. 16–23, May2001. | [12] S. Hamdioui, G. Gaydadjiev, and A. van de Goor, "The state-of-art and future trends in testing embedded memories," in Proc. Records Int.Workshop Memory Technol., Design, Test., 2004, pp. 54–59. | [13] Y. Zorian and S. Shoukourian, "Embedded-memory test and repair: Infrastructure IP for SOC yield," IEEE Design Test Comput., vol. 20, no. 3, pp. 58–66, May/Jun. 2003. | |