A Novel Approach for Optimization of Microcode BIST by Enhanced Faults Coverage for Embedded Memory

Maheswari Veeranki,
M.Tech Student, ECE Department,
VIKAS Group of Institutions,
Andhra Pradesh, India.

Sekhar Reddy Gaddam,
Asst. Professor, ECE Department,
VIKAS Group of Institutions,
Andhra Pradesh, India.

***Abstract*: SoC (system-on-a-chip) is a high performance microprocessor, since we can program and give instruction to the Microprocessor to do whatever you want to do. SoC testing is the testing of system-on-a-chip devices. Testing them becomes an increasing challenge as these devices become more complex. A SoC design is typically built block by block; efficient testing is also best done block by block System-on-chip (SoC) designs are moving from logic dominant chips to memory dominant chips to be able to meet future application requirements. Embedded memory density and area on-chip is increasing day by day. In order to achieve good memory yield, an at-speed test technique such as Built-In Self Test (BIST) must be implemented to test these embedded memories. Memory Built-in Self Test (MBIST) is the popular approach to test embedded memories. MBIST usually use the deterministic pattern such as MARCH test algorithm to test memories. In MARCH test algorithm, the patterns are generated according to specified predetermined values. The existing March algorithms consist of as many as four or seven operations per March element. Therefore, it is essential to define new test algorithms which fulfill the need of detecting new faults. A new March BLC tests having number of operations per element according to the today's growing needs of embedded memory testing with enhanced fault using Verilog HDL as a primary language and used Xilinx as simulation tool.**

Keywords: SoC (system-on-a-chip), Memory Built-In Self Test (MBIST), Embedded memory fault, MARCH test algorithm, Hardware Descriptive Language (HDL)

I. INTRODUCTION

According to the 2001 ITRS, today's system on chips(SoCs) are moving from logic dominant chips to memory dominant chips in order to deal with today‟s and future application requirements. The dominating logic (about 64% in 1999) is changing to dominating memory (approaching 90% by 2011) [6] as shown in Fig.1.

Also, as the memories grow in size and speed, the signal lines, that is bit lines, word lines and address decoder preselect lines will have high parasitic capacitance in addition to a high load. This increases their sensitivity for delay and timing related faults.

Moreover, the significance of the resistive opens is considered to increase in current and future technologies. Since the partial resistive opens behave as delay and time related faults, these faults will become more important in the deep-submicron technologies [1].

The following considerations for fault modeling for new technologies also have to be taken into account, for example:

- Transistor Short channel effect: lowering the threshold voltage may make the drain leakage contribution significant.
- Cross talk effect and noise from power lines.
- The impact of process variation

The above cited newer defects are a source of new fault models. The development of new, optimal, high coverage tests and diagnostic algorithms allow for dealing with the new defects. The greater the fault detection and localization coverage, the higher the repair efficiency; hence higher the obtained yield.

IJERT

II.TYPES OF FAULTS IN EMBEDDED MEMORY

Generally the memory faults are due to

1. Open connections
2. Shorted connections
3. Storage cells stuck at 0/1

Memories fail in a number of different ways. The three main parts,

1. Address decoder logic
2. Memory cell array
3. Read/write logic, can each have flaws that cause the device to fail. Memory testing, while similar to random logic testing, focuses on testing for these memory-specific failures

The types of memory faults

1. Stuck at faults
2. Transition faults
3. Coupling faults
4. Neighborhood pattern sensitive faults

Thus, the new trends in Memory testing will be driven by the following items:

Fault modeling: New fault models should be established in order to deal with the new defects introduced by current and future (deep-submicron) technologies.

Test algorithm design: Optimal test/diagnosis algorithms to guarantee high defect coverage for the new memory technologies and reduce the DPM level.

BIST: The only solution that allows at-speed testing for embedded memories.

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

III.WHY MARCH C+ ALGORITHM?

Among the all March based algorithms March C+ has less number of operations (14n) and more defect coverage. Due to less no of operations it requires less testing time to test the memory, under test. And it detects more no of faults because of its more defect coverage. That's why we have selected the March C+ algorithm in the implementation of hardwired-based memory BIST controller.

March C+ is derived from the modified March C algorithm, which modifies the original March C algorithm by adding an extra read operation after each stage of the march. While increasing the algorithm from 10n to 13n, this extra read allows additional fault detection, most notably, stuck-open faults for all types of RAM.

The March C+ algorithm detects the same faults as March C and also stuck open faults, timing faults, if the test is performed at speed. The aim of this is to generate test patterns (data, address and control signals) by using the March C+ algorithm. These test patterns are applied to memory, which is under test, through memory interfacing unit. The inputs to the access unit are generated by the BIST control unit. Finally three blocks (control unit, access unit and memory interfacing unit) are integrated. This BIST controller is inserted into memory model. Finally this controller gives the pass/fail information of the memory (under test).

IV.TYPES OF MEMORY BIST CONTROLLERS

Memory BIST (Built-in self test) circuit is used for testing of embedded memories. Within the chip a special circuitry is provided for BIST along with memory. This circuitry tests the embedded memory and gives the information that whether all memory locations within the memory are working correctly or not.

A.Microcode-based BIST A microcode-based memory BIST features a set of predefined instructions, or microcode, which is used to write the selected test algorithms. The written tests are loaded in the memory BIST controller. This microcode-based type of memory BIST allows changes in the selected test algorithm with no impact on the hardware of the controller. This flexibility, however, may come with the cost of higher logic overhead for the controller.

B.Processor-based memory BIST The processor-based memory BIST is done by executing an assembly-language program in the on-chip microprocessor to generate test patterns including the address sequence, data patterns, and control signals. The memory outputs are then compared with the expected correct data.

C.Hardwired-based BIST A hardwired-based controller is a hardware realization of a selected memory test algorithm, usually in the form of a Finite State Machine (FSM). This type of memory BIST architecture has optimum logic overhead, however, lacks the flexibility to accommodate any changes in the selected memory test algorithm.

The below table gives the various design trade-offs among the three implementation schemes.

TABLE 1: Design trade-offs among BISTs.

Scheme	Test Time	Area OH	Routing OH	Flexibility
Hardwired	Short	Low	High	Zero
Microcode	Average	High	Low	Low
Processor	Long	Zero	Zero	High

V. MICROCODE BIST CONTROLLER

A microcode-based controller is a hardware realization of a selected memory test algorithm, usually in the form of a Finite State Machine (FSM). This type of memory BIST architecture has optimum logic overhead, however, lacks the flexibility to accommodate any changes in the selected memory test algorithm. This results in re-design and re-implementation of the hardwired-based memory BIST for any minor changes in the selected memory test algorithm. Although it is the oldest memory BIST scheme amongst the three, hardwired-based BIST is still much in use and techniques have been kept developing.

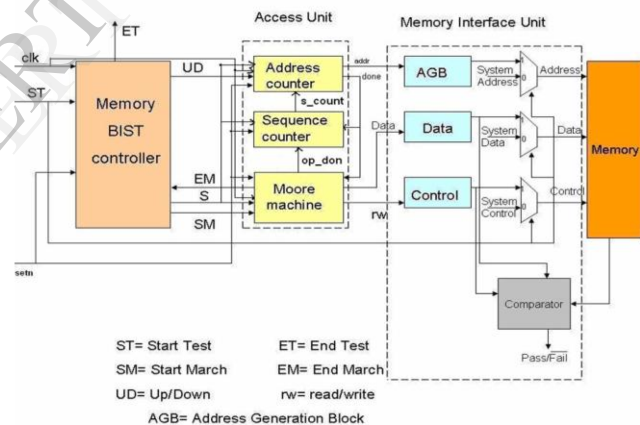


Fig.1. Block diagram of microcode BIST Controller

In the above block diagram shown hardwired BIST controller consists of 3 blocks. They are 1.Control Unit (CU).2.Access Unit (AU). 3. Memory Interface Unit (MIU)

A.Control Unit

The memory BIST controller is one of the blocks in Hardwired BIST. The block diagram is as shown in Figure2.

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

TABLE 2: Pin Description of CU

Pin name	Direction	Function	Interface
Clk	Input	It synchronizes all the memory operations	Externally applied to the memory BIST controller
Reset_n	Input	'0' resets all the registers, address counter and sequence counter '1' memory BIST is allowed to start the working	Applied to memory BIST controller and Access unit
start_test	Input	'0' Controller stays in the idle mode only '1' Controller will start the memory testing	Applied to the memory BIST controller
end_test	Output	'0' indicates the memory testing is not over when ST=1 '1' indicates the testing is completed	Memory BIST controller will assert after the testing
start_march	Output	'0' indicates waiting for a march element '1' indicates the access unit is enabled	It is the intermediate signal between memory BIST controller and access unit
end_march	Output	'0' indicates the march element operation is carrying out when SM=1 '1' indicates the March element is over	---do--
Updown_order	Output	'1' indicates the addressing order is incremented '0' indicates the addressing order is decremented	---do--
march_ele	Output	It is a 2 bit code indicating the type of March element.	Signal from controller to access unit

Initially the power to the circuit is switched on. Now we apply the reset_n signal to the BIST controller. Then all the registers are initialized. When the signal start_test is asserted then the BIST controller automatically selects the March C+ algorithm.

Then the March elements and addressing orders in the March C+ algorithm are given to Access unit through march_ele and updown_order signals by asserting SM signal.After receiving the end_march signal from the access unit memory BIST controller sends another March element.The completion of test will be indicated by end_test signal.The Pin Description of CU is shown in Table 2.

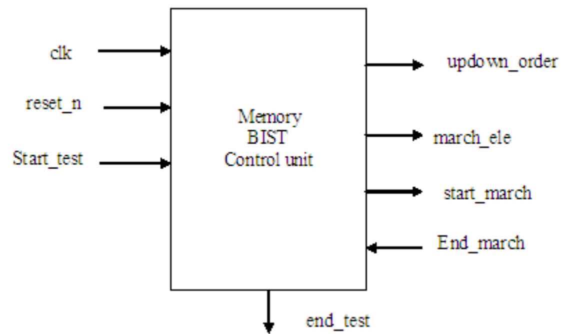


Fig.2. Block diagram of MBIST Control Unit

B.Memory Interface Unit

Memory Interface Unit will take the address, data and rw signals from the access unit and stored in the Address, Data and Control registers. The signals will be given to multiplexers as shown in Figure 3.. Depending on start_test value, respective signals are selected. If start_test =1 then test data and control will be considered otherwise memory is used for normal functions.

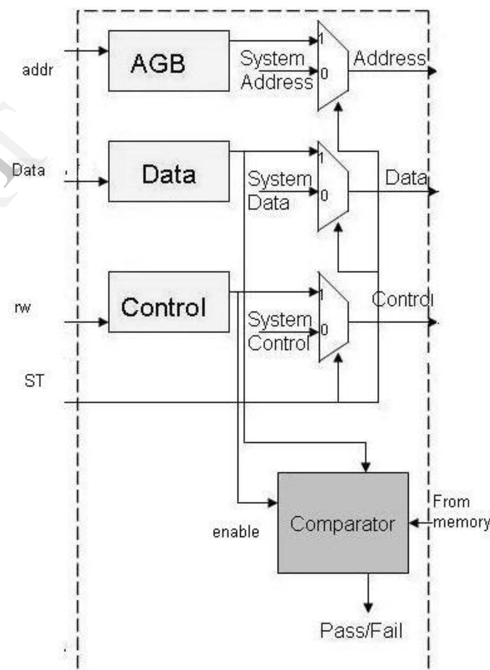


Fig.3. Memory Interface Unit

Blocks inside MIU

1.Address Generation Block (AGB)
2.Data register
3.Control register
4.Comparator

IJERT

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

a)Address Generation Block: This block stores the address of memory on which the operations to be performed. This address will be used for the diagnosis purpose if the fault is occurred in the respective location

b) Data register: It stores the data to be written onto the memory location.

c) Control register: It consists of the memory operation to be performed on the memory location indicated by Address Generation Block

d) Comparator: It is used to compare the expected data and data from the memory and tells the memory location is pass or fail. Comparison is done when the enable signal is '0'.The enable signal is nothing but control signal.

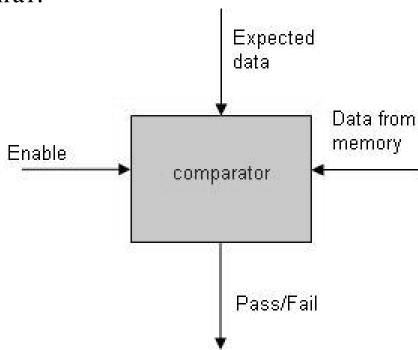


Fig.4. Block diagram of Comparator

C.Access Unit

Access unit is like a heart for the hardwired memory BIST controller. It takes the inputs from the control unit and generates the test patterns. These test patterns are applied to the memory through memory interfacing unit.

IJERT

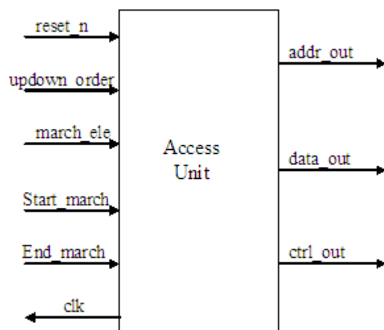


Fig.5. Block diagram of Access Unit

The internal block diagram of the access unit is shown in Figure 6. This access unit consists of address counter, sequence counter and Moore machine. Address counter generates the address of memory under test (MUT) by taking sequence count value. Moore machine gives the data and control (read/write) signals of MUT.

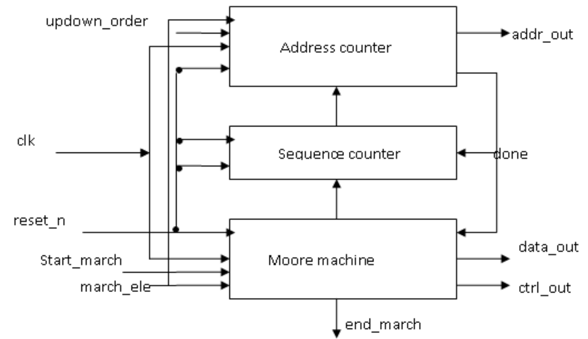


Fig.6. Internal block Diagram of Access Unit

a)Address counter: It is initialized when reset_n=0. It takes the addressing order (updown_order) signal as input from control unit and generates the addr_out signal to memory interfacing unit. It is also synchronized with moore machine and checks whether address counter reaches the maximum value or not.

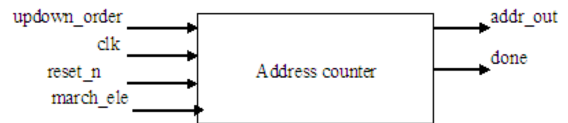


Fig.7. Block diagram of Address Counter

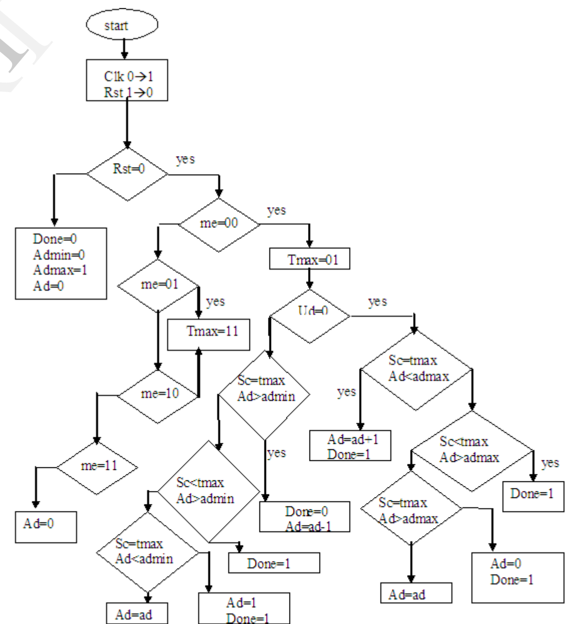


Fig.8. Flowchart for Address Counter

Depends on march element value march operation are performed in each and every address location. When sequence count reaches its maximum value, address count value is incremented. When the address counter reaches its reaches maximum value, it generates the 'done' signal. This done signal is given to moore machine and sequence counter. Depending upon the above operations a flow chart is drawn for address counter.

The Simulated result of Address counter is shown in the below figure.

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

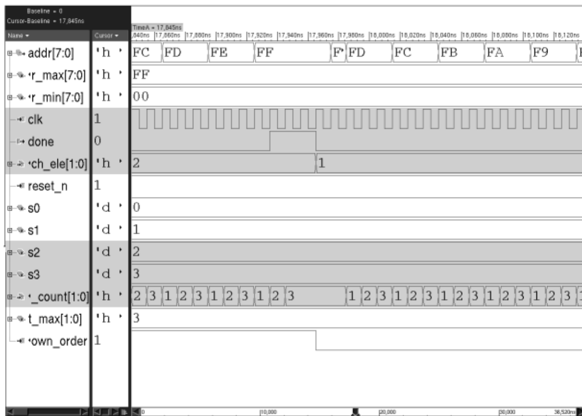
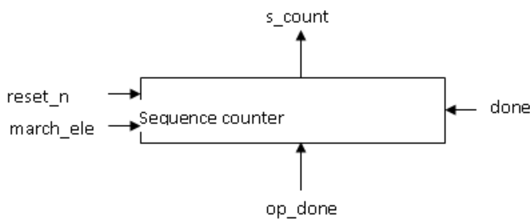


Fig.9. Simulated result of Address Counter

b) Sequence Counter: This block is executed with respect to the op_done signal. Sequence counter is initialized when reset_n is zero. It increments every time when the operation is completed with respect to march_ele value. When it reaches the maximum value it is reset by



“00”.

Figure 10: Block diagram of Sequence Counter

From the operations flow chart is drawn for sequence counter is shown in Figure 11.

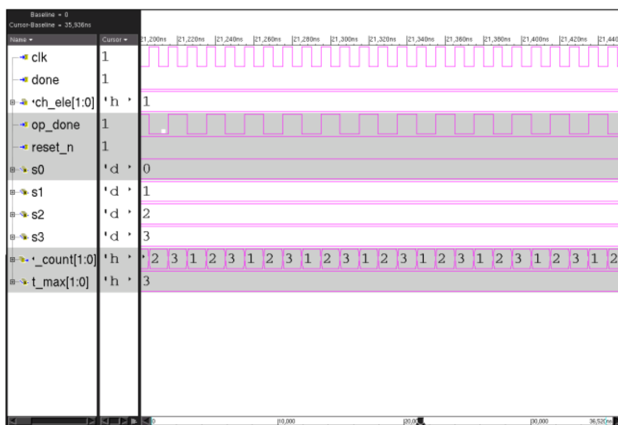
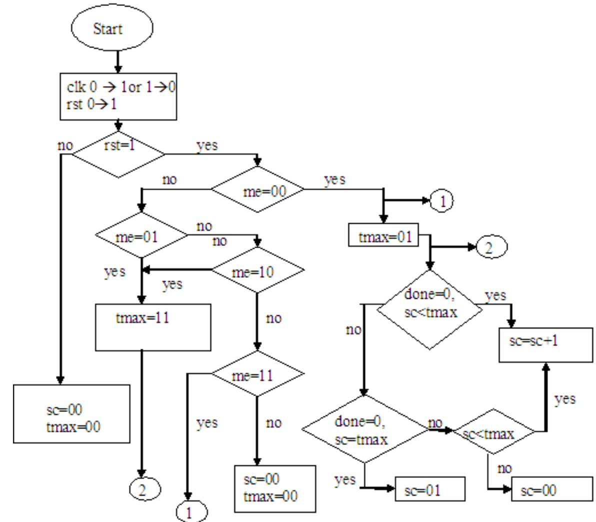


Fig.11.Flowchart for Sequence Counter



Simulated result of Sequence counter can be observed in Figure12.

Fig.12. Simulated result of Sequence Counter

c)Moore Machine: Block diagram is shown in Figure 13 and its operation is explained as follows.

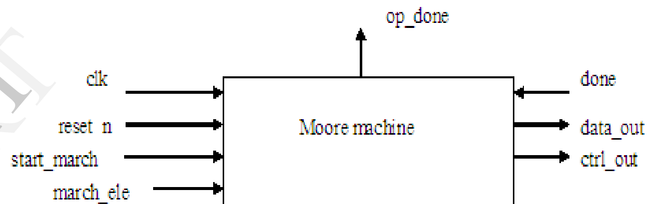


Fig.13. Moore Machine

1.Moore machine will perform the operations for a particular element when the start_ march signal is asserted.

2.The march_ele signal will fetch the March element from the Memory BIST controller. It consists of an FSM which will explain the flow of operations.

3.Moore machine generates data and control signal and send them to MIU(memory Interface Unit). The sequence counter (s_count) value is incremented after performing each operation of a particular March element.

4.When the sequence counter reaches maximum value, it will check for the ‘done’ signal.

5.If the ‘done’ is high then end_march signal is asserted and start_march signal made to zero. The Moore machine will wait for the next March element (idle state) otherwise the above process is repeated for remaining address locations.

IJERT

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

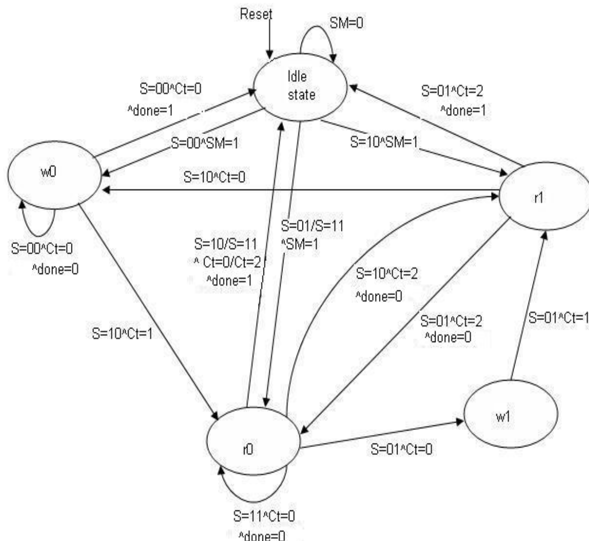


Fig.14. FSM for Moore Machine

The FSM is in idle state until the start_march value is asserted. The When start_march =1, then its state changes to another depending on march_ele value.

After performing each operation the sequence counter value (s_count) is incremented by 1.When the sequence counter reaches max value then it checks for 'done' signal.

If it is '0' then the operations are repeated for remaining address locations otherwise it enters into idle state.

The Simulated result of Moore machine is shown in the figure15.

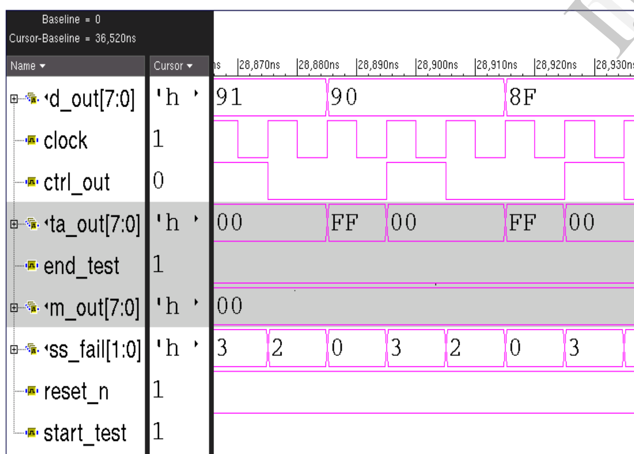


Fig.15.Simulated result of Moore Machine

Control unit, access unit and memory interfacing unit are integrated into a single block to form a hardwired BIST controller. This BIST circuit is wrapped around a memory which is to be tested. This controller generates the test patterns (data, address and read/write control signals) and supplied to the memory. It writes the data into the memory, read back from the memory and compares with the actual data. If both data is same it gives a pass signal. Otherwise it gives a fail signal. Whenever all memory locations are over it asserts the end of test signal, which means the testing is over.

The Simulated result of Access Unit which includes address counter, sequence counter and Moore machine is shown in the below figure.

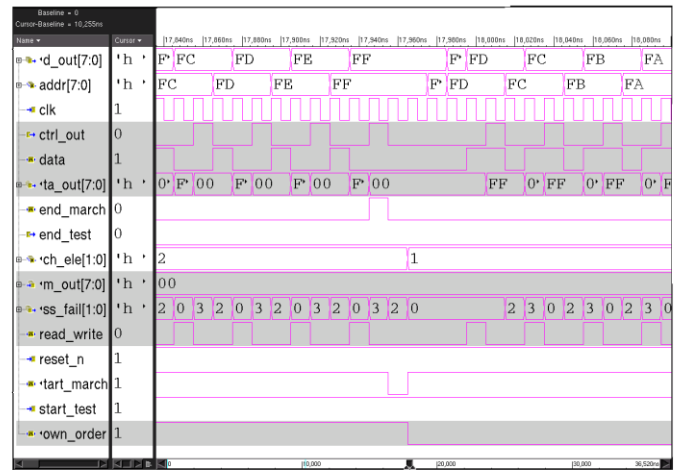


Fig.16. Simulated result of Access Unit

Control unit takes clock, reset and start test as input signals and generates the addressing order (UD), start march (SM) and march element address (S) as output signals. These output signals are applied to the access unit as in Figure 1.

By taking the addressing order (UD), start march (SM) and march element address (S) as input signals access unit generates the address, data and control(r/w) signals as output. These are applied as inputs to the memory interfacing unit.

Memory interfacing unit (MIU) supplies these patterns to the memory (under test) though the multiplexers. And the comparator in the MIU compares the memory output with the original data and gives a pass signal if both are matching. Otherwise it gives a fail signal which means the particular memory locations are not working properly due to faults presented.

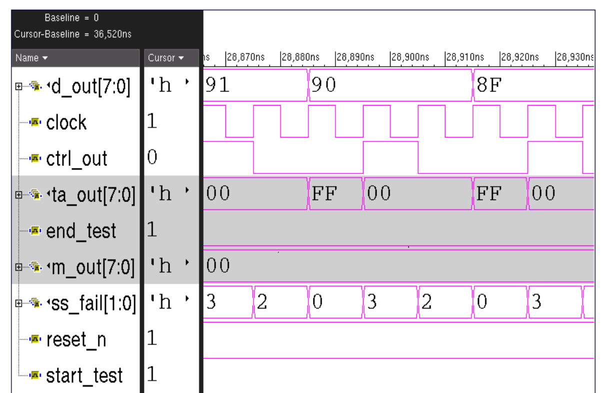


Fig.17.Simulated result of Microcode-Based BIST

IJERT

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

VI.CONCLUSION

The above waveforms have shown that the micro-code MBIST architecture above is an effective testing method to test embedded memories as it provides a flexible approach and better fault coverage. Just as March C+, any new march algorithm can be implemented using the same BIST hardware by replacing the microcode storage unit, without the need to redesign the entire circuitry.

REFERENCES

1. S. Hamdioui, G.N. Gaydadjiev, A.J .van de Goor, "State-of-art and Future Trends in Testing Embedded Memories", International Workshop on Memory Technology, Design and Testing (MTDT"04), 2004.
2. Sandra Irobi Zaid Al-Ars Said Hamdioui.Memory Test Optimization for Parasitic Bit Line Coupling in SRAMs. IEEE International Test Conference,2010.
3. N. Z. Haron, S.A.M. Junos, A.S.A. Aziz, "Modelling and Simulation of Microcode Built-In Self test Architecture for Embedded Memories", In Proc. of IEEE International Symposium on Communications and Information Technologies pp. 136-139, 2007.
4. Sungju Park et al, "Microcode-Based Memory BIST Implementing Modified March Algorithms", Journal of the Korean Physical Society, Vol. 40, No. 4, April 2002, pp. 749-753.
5. B. F. Cockburn: "Tutorial on Semiconductor Memory Testing"Journal of Electronic Testing: Theory and Applications, 5, pp 321-336 1994 Kluwer Academic Publishers,Boston.
6. I.S. Irobi, Z. Al-Ars, and S. Hamdioui. Detecting memory faults in the presence of bit line coupling in sram devices. IEEE International Test Conference, 2010
7. Dr. R.K. Sharma Aditi Sood. "Modeling and Simulation of Microcode based Built-In Self Test for Multi-Operation Memory Test Algorithms", IJCSI International Journal of Computer Science Issues, Vol. 7, Issue 3, No. 2, May 2010 pp.36-40.

IJERT

(This work is licensed under a Creative Commons Attribution 4.0 International License.)