ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

# Scan Based Built-In Self-Repair (BISR) For Embedded Memories

**B. Navya**
**Department of Electronics & Communication Engineering, HITAM, Hyderabad, Telangana-502401, India.**

**K. Bindumadhavi**
**Department of Electronics & Communication Engineering, HITAM, Hyderabad, Telangana-502401, India.**

**K. Anil Kumar**
**Department of Electronics & Communication Engineering, HITAM, Hyderabad, Telangana-502401, India.**

***ABSTRACT***

***The production of better power droop (PD) during at-speed analysis conducted by Logic Built-In Self test (LBIST) is a big problem for latest ICs. In fact, the PD based during analysis may lag signal transitions of the circuit under test (CUT): an aftereffect that may be afield accustomed as delay faults, with consistent erroneous production of analysis fails and increase in yield loss. In this paper, we addressed a atypical scalable access to compress the PD during at-speed analysis of sequential circuits with scan-based LBIST application the launch-oncapture method. This is accomplished by decreasing the action factor of the CUT, by able changing of the test vectors produced by the LBIST of sequential ICs. Our scalable solution allows us to abate PD to a value same as that occurring at the time of CUT in field work, without multiplying the amount of test vectors needed to accomplish required fault coverage (FC).***

## INTRODUCTION

The advancing ascent of microelectronic technology is enabling the artifact of added circuitous ICs. Together with several allowances (improved performance, decreased amount per function, etc.), this poses austere challenges in agreement of analysis and believability [1]. In particular, during at-speed analysis of high-performance microprocessors, the IC action agency (AF) induced by the activated analysis vectors is decidedly college than that accomplished during in acreage operation [2]. Consequently, boundless ability bend (PD) may be generated, which will apathetic down the ambit beneath analysis (CUT) arresting transitions. This abnormality is acceptable to be afield accustomed as due to adjournment faults.

As a result, a apocryphal analysis abort will be generated, with consistent access in crop accident [3].At-speed analysis of argumentation blocks is nowadays frequently performed application Argumentation BIST (LBIST) [4], which can yield the anatomy of either combinational LBIST or scan-based LBIST, depending on whether the CUT is a combinational ambit or a consecutive one with browse [5, 6],

In case of scan-based LBIST, two basal capture-clocking schemes abide:
the launch-on-shift (LOS) arrangement and
the launch-on-capture (LOC) scheme
In LOS schemes, analysis vectors are activated to the CUT at the endure alarm (CK) of the about-face phase, and the CUT acknowledgment is sampled on the browse chains at the afterward abduction CK. In the LOC scheme, instead, analysis vectors are aboriginal loaded into the scan-chains during the about-face phase; then, in a afterward abduction phase, they are aboriginal activated to the CUT at a barrage CK, and the CUT acknowledgment is captured on the browse chains in a afterward abduction CK [7, 8].

In this paper, we accede the case of consecutive CUTs with scan-based LBIST adopting an LOC scheme, which is frequently adopted for high-performance microprocessors. They ache from the PD problems discussed above, abnormally during the abduction phase, due to the top AF of the CUT induced by the activated analysis patterns.

**Cite this article as:** B. Navya, K. Bindumadhavi & K. Anil Kumar, "Scan Based Built-In Self-Repair (BISR) For Embedded Memories", International Journal & Magazine of Engineering, Technology, Management and Research, Volume 6 Issue 1, 2019, Page 81-88.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

Solutions acceptance designers to abate PD during the abduction appearance in scan-based LBIST are accordingly needed. While several approaches accept been proposed to abate PD for combinational LBIST, alone a few solutions abide for scan-based LBIST. In [2], PD is bargain by a multicycle BIST arrangement with fractional observation. This access does not appulse on accountability advantage (FC) (actually, it presents a slight FC access of 5% compared with accepted scan-based-LBIST), but enables abridgement of PD by 33% only, compared with accepted scan-based-LBIST.

## PROPOSED ARCHITECTURE

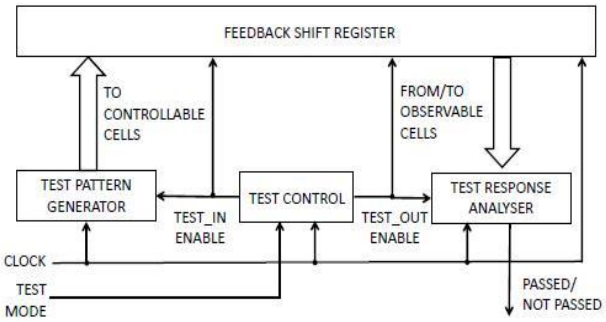


**Fig 1: Block diagram illustrating presented method**

The method presented in this paper is similar to the traditional scan design in that is provides a simple way of setting and observing each flip-flop in a circuit. However, unlike in scan, we do not connect flip-flops in scan chains. Instead, to support the test mode, we modify the original FSR as follows

1) The input of the original flip flop becomes the functional input of the multiplexer (MUX). The test input of MUX is connected to the Test Pattern Generator (TPG).

2) The duplicated output is connected to the Test Response Analyzer (TRA) through a switch. When the test mode is selected, the flip-flops with multiplexed inputs become inputs to the combinational logic. The flip-flops which have a switch on the output become outputs of the combinational logic. As in a scan design, this increases controllability and observability, making possible testing a sequential circuit with tests for combinational logic.

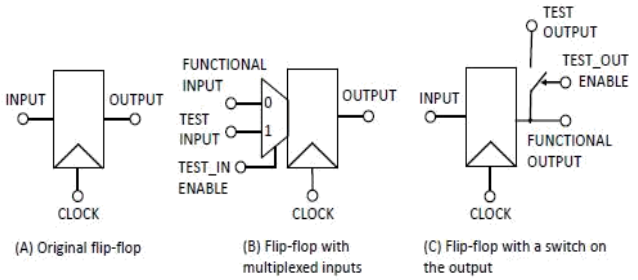


**Fig 2: Modifications of FSR flip-flops to support test mode.**

Note that such a technique does not affect the propagation delay of the original circuit. In the traditional scan, the propagation delay is always increased by the delay of a MUX. We add MUX'es only to the controllable cells, whose feedback functions are trivial. Therefore, the propagation delay is still determined by the observable cells, whose feedback functions are non-trivial.

The following signals are added to the FSR to control and observe its cells:

1) Test in enable signal controls the application of test vectors. When it is asserted, controllable cells are connected to the TPG and TPG is connected to the clock. Otherwise, controllable cells are connected to their predecessor cells and TPG is not connected to the clock.

2) Test out enable signal controls output response analysis. When it is asserted, observable cells are connected to both the TRA and their successor cells and TRA is connected to the clock. Otherwise, observable cells are connected to their successor cells only, and TRA is not connected to the clock.

For each observable cell i, the value at the test output is compared to the expected value of fi using an XOR gate. The outputs of all XORs are fed into an OR gate the output of which indicates the presence/absence of a fault.

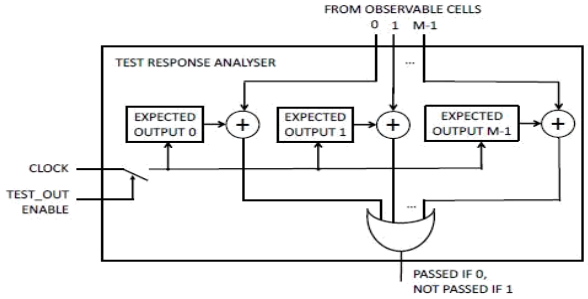


**Fig 3: The structure of the TRA**

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

### Detecting faults in the combinational logic

1) Assert the test in enable signal to connect test inputs of controllable cells controllable cells to the TPG.
2) Apply one clock to load the test vector ti 2 T, i 2 {1,2, . . . ,k+3}, from the TPG to all controllable cells in parallel.
3) Apply one clock to evaluate the non-trivial feedback functions for the input assignment defined by ti. The resulting output responses are captured at the observable cells. At the same clock cycle, the next test vector ti+1 from T is loaded from the TPG to the controllable cells.
4) Assert the test out enable signal to connect test outputs of observable cells to the TRA.
5) Apply one clock to upload the responses to ti from all observable cells to the TRA in parallel. The TRA compares the computed responses to the expected responses. If they agree, TRA outputs ”passed”. Otherwise, it outputs ”not passed”. At the same clock cycle, non-trivial feedback functions are evaluated for the input assignment defined by ti+1. The resulting output responses are captured at the observable cells. The next test vector ti+2 from T is loaded from TPG to the controllable cells.
6) Repeat the steps 2, 3 and 5 until all test vectors from T are applied.

This Procedure completes the application of all tests from T and evaluation of all output responses in k+5 clock cycles.

### Detecting remaining faults in FSR

1) Set the test out enable signal low to disconnect test outputs of observable cells from the TRA.
2) Assert the test in enable signal to connect test inputs of controllable cells to the TPG.
3) Apply one clock to load the test vector t1 2 T1 from the TPG into all controllable cells in parallel.
4) Repeat d−1 times: Apply one clock to evaluate the non-trivial feedback functions for the input assignment defined by t1. The resulting output responses are captured at the observable cells. All internal cells capture the value of their predecessors. At the same clock cycle, the same test vector t1 from T is loaded again from the TPG to the controllable cells.
5) Set the test in enable signal low to connect functional inputs of controllable cells to their predecessors.
6) Apply one clock to capture the value of the predecessors of controllable cells into the controllable cells.
7) Apply one clock to evaluate the non-trivial feedback functions for the input assignment defined by the controllable cells. The resulting output responses are captured at the output flip-flops.
8) Assert the test out enable signal to connect test outputs of observable cells to the TRA.
9) Apply one clock to upload the responses from all observable cells to the TRA in parallel. The TRA compares the computed responses to the expected responses. If the two responses agree,
10) Repeat the steps 1-9 for the test vector t2 € T1.

This Procedure completes the application of tests and evaluation of all output responses in 2d+6 clock cycles.

### Detection of faults in the TPG

Consider the case when a single stuck-at fault occurs of the output j of the TPG, j 2 {0E,0D,1, . . .,k}. Such a fault will manifest itself as a multiple stuck-at fault at the controllable cells connected to the output j. Since none of the state variables occurs in more than one ANF, each faulty input will affect only one fi. Therefore, the change in values caused by the fault will not be cancelled out and the fault will be detected by the Procedure 1.

### Detection of faults in the TRA

TRA stores the expected responses to the tests and compares them to the computed responses. We have shown in the previous Section that for T2 the expected responses may differ for functions whose dependence set have different sizes. In the worse case, all non-trivial functions may have dependence sets of different sizes. Then, in order to store the expected responses, we need (k+3)×m bits, where m is the number of non-trivial feedback functions.

Note that the TRA circuit shown in Fig.3.4 handles not only single stuck-at faults in the FSR, but also single stuck-at faults which occurs in the TRA itself, except the stuck-at-0 and stuck-at-1 fault at the output of the OR

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

gate. To allow for detection of these faults, the OR gate can be duplicated.

### Proposed Model with CUT

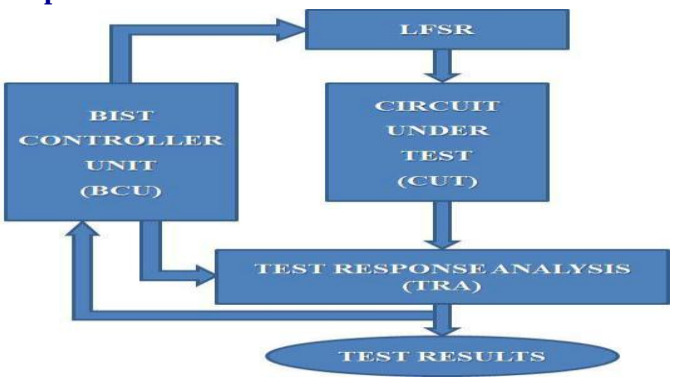


**Fig 4: Proposed Model with CUT**

### Test Pattern Generator (TPG)

This module generates the test patterns required to sensitize the faults and propagate the effect to the outputs (of the CUT). As the test pattern generator is a circuit (not equipment) its area is limited. So storing and then generating test patterns obtained by ATPG algorithms on the CUT using the hardware test pattern generator is not feasible. In other words, the test pattern generator cannot be a memory where all test patters obtained by running ATPG algorithms (or random pattern generation algorithms) on the CUT are stored and applied during execution of the BIST. Instead, the test pattern generator is basically a type of register which generates random patterns which act as test patterns. The main emphasis of the register design is to have low area yet generate as many different patterns (from 0 to 2n, if there are *n* flip-flops in the register) as possible.

The proposed low power LFSR technique uses bit swapping technique to reduce the peak power. By connecting multiplexers on the LFSR register. The numbers of transitions are decreased for that cell which are under bit swapping. The number of transitions in each register in LFSR without applying bit swapping technique here two cells in an *n*bit LFSR are considered to be adjacent if the output of one cell feeds the input of the second directly (i.e., without an intervening XOR gate). Each cell in a maximal-length *n*-stage LFSR (internal or external) will produce a number of transitions equal to 2*n*-1 after going through a sequence of 2*n* clock cycles.

The sequence of 1s and 0s that is followed by one bit position of a maximal-length LFSR is commonly referred to as an *m* sequence. Each bit within the LFSR will follow the same *m* sequence with a one-time-step delay. The *m*-sequence generated by an LFSR of length *n* has a periodicity of 2*n*-1. It is a well-known standard property of an *m*-sequence of length *n* that the total number of runs of consecutive occurrences of the same binary digit is 2*n*-1.The beginning of each run is marked by a transition between 0 and 1.Therefore, the total number of transitions for each stage of the LFSR is 2*n*-1.

Test Response Analysis (TRA): It analyses the value sequence on PO and compares it with the expected output.

### BIST Controller Unit (BCU):

It controls the test execution; it manages the TPG, TRA and reconfigures the CUT and the multiplexer.
Basic BIST architecture includes functions which are necessary to execute the self-testing feature so that testing is accomplished without the aid of external hardware. It has two major components named Hardware or Test Pattern Generator (TPG) and Output Response Compacter or Analyzer (ORA).

The TPG produces a sequence of patterns for testing the Circuit Under Test (CUT) while the ORA compacts the output responses of the CUT into some type of pass/fail indication which decides good or faulty result. The other two functions needed for system-level use of BIST include the test controller and input MUX. Besides the normal input/output (I/O) pins, the incorporation of BIST may also require additional I/O pins for activating the BIST sequence and to give valid results. The input test patterns can be stored in a Read Only Memory (ROM). Expected responses are read from ORA ROM and are compared to the actual output response of the CUT for each test vector. Any mismatch detected by the

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

comparator is latched to indicate a failure has occurred during the BIST sequencing.

### Circuit under Test (CUT):

It is the portion of the circuit tested in BIST mode. It can be sequential, combinational or a memory. It is delimited by their Primary Input (PI) and Primary Output (PO).

In CUT we are using UART (Universal Asynchronous receiver/Transmitter). Serial data is transmitted via its serial port. A serial port is one of the most universal parts of a computer. It is a connector where serial line is attached and connected to peripheral devices such as mouse, modem, and printer and even to another computer. In contrast to parallel communication, these peripheral devices communicate using a serial bit stream.

Design with BIST Capability protocol (where data is sent one bit at a time). The serial port is usually connected to UART, an integrated circuit which handles the conversion between serial and parallel data.

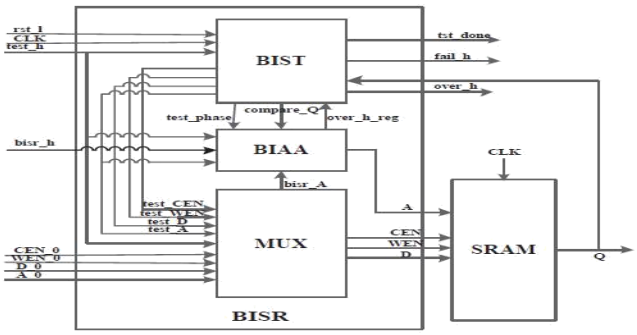


Fig 5: BISR Architecture

### BASIC ARCHITECTURE

The basic architecture of a static RAM includes one or more rectangular arrays of memory cells with support circuitry to decode addresses, and implement the required read and write operations. Additional support circuitry used to implement special features, such as burst operation, may also be present on the chip. Figure shows a basic block diagram of a synchronous SRAM. As you read, you may wish to refer to the diagram to help you visualize how the SRAM works.

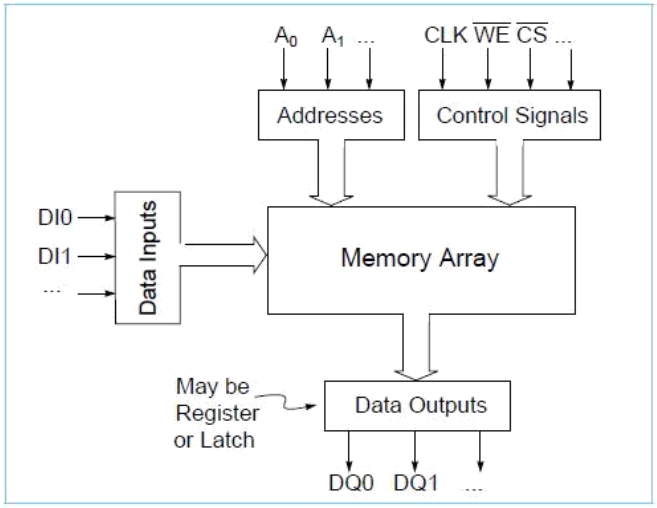


Fig 6: Basic Architecture

### Memory Arrays:

SRAM memory arrays are arranged in rows and columns of memory cells called word lines and bit lines, respectively. In IBM SRAMs, the word lines are made from poly silicon while the bit lines are metal. Each memory cell has a unique location or address defined by the intersection of a row and column. Each address is linked to a particular data input/output pin. The number of arrays on a memory chip is determined by the total size of the memory, the speed at which the memory must operate layout and testing requirements, and the number of data I /Os on the chip.

### Memory Cell:

An SRAM memory cell is a bi-stable flip-flop made up of four to six transistors. The flip-flop may be in either of two states that can be interpreted by the support circuitry to be a 1 or a 0. Many of the SRAMs on the market use a four transistor cell with a poly silicon load. Suitable for medium to high performance, this design has a relatively high leakage current, and consequently high standby current. Four transistor designs may also be more susceptible to various types of radiation induced soft errors. IBM's SRAMs all use a six transistor memory cell (also called a six-device cell) that is highly stable, relatively impervious to soft errors, and has low leakage and standby currents.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

## BIAA

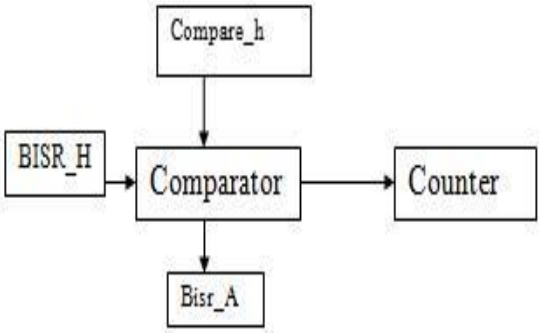


Fig 7: BIAA Architecture

If The BISR_H=0 then initially BIAA stores 0 address when the compare_h=0 then comparator can take address from the mux and directly it will given to the counter. If The BISR_H=1 then counter increments the values from 0 t0 8 why because in the bist comparator and test_v_ram having 0 t0 8 addresses only.

The proposed SRAM BISR strategy is flexible. The SRAM users can decide whether to use it by setting a signal. So the redundancy of the SRAM is designed to be selectable. In another word, some normal words in SRAM can be selected as redundancy if the SRAM needs to repair itself. We call these words Normal-Redundant words to distinguish them from the real normal ones. We take a 64 × 4 SRAM for example, as shown in Figure 1. There are 60 normal words and 4 Normal-Redundant words. When the BISR is used, the Normal-Redundant words are accessed as normal ones. Otherwise, the Normal-Redundant words can only be accessed when there are faults in normal words. In this case, the SRAM can only offer capacity of 60 words to users. This should be referred in SRAM manual in details.

## BISR PROCEDURE

Figure 8 shows the proposed BISR block diagram. The BISR starts by resetting the system (rst_l = 0). After that if the system work in test mode, it goes into TEST phase. During this phase, the BIST module and BIAA module work in parallel.

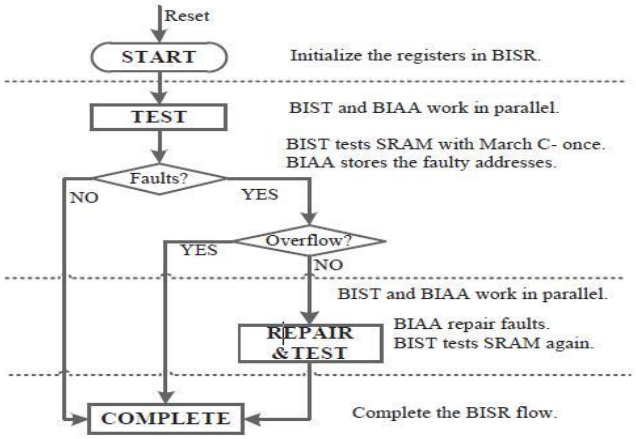


Fig 8: Block Diagram of BISR

The BIST use March C- to test the normal addresses of SRAM. As long as any fault is detected by the BIST module, the faulty address will be sent to the BIAA module. Then the BIAA module checks whether the faulty address has been already stored in Fault-A-Mem. If the faulty address has not been stored, the BIAA stores it and the faulty address counter adds 1. Otherwise, the faulty address can be ignored. When the test is completed, there will be two conditions. If there is no fault or there are too many faults that overflow the redundancy capacity, BISR goes into COMPLETE phase. If there are faults in SRAM but without overflows, the system goes into REPAIR&TEST phase. The same as during TEST phase, the BIST module and BIAA module work at the same time in REPAIR&TEST phase. The BIAA module replaces the faulty addresses stored in Fault-A-Mem with redundant ones and the BIST module tests the SRAM again. There will be two results: repair fail or repair pass. By using the BISR, the users can pick out the SRAMs that can be repaired with redundancy or the ones with no fault.

## BISR Features

Firstly, the BISR strategy is flexible. TABLE I lists the operation modes of SRAM. In access mode, SRAM users can decide whether the BISR is used base on their needs. If the BISR is needed, the Normal-Redundant words will be taken as redundancy to repair fault. If not, they can be accessed as normal words.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

**TABLE 1: SRAM OPERATION MODES**

| Modes | *Repair selection* | *Operation* |
|---|---|---|
| Test mode (*test_h*=1) | Default: repair (*bisr_h*=1) | Access normal words. Repair faults and test. |
| | Don't repair (*bisr_h*=0) | Access normal words. Test only. |
| Access mode (*test_h*=0) | Repair (*bisr_h*=1) | Access normal words. Repair faults and write/read SRAM. |
| | Don't repair (*bisr_h*=0) | Access Normal- Redundant and normal Words. Write/read SRAM only. |

Secondly, the BISR strategy is efficient. On one hand, the efficiency reflects on the selectable redundancy which is described as flexible above. No matter the BISR is applied or not, the Normal-Redundant words are used in the SRAM. It saves area and has high utilization. On the other hand, each fault address can be stored only once into Fault-A-Mem. As said before, March C- has 6 steps. In another word, the addresses will be read 5 times in one test. Some faulty addresses can be detected in more than one step. Take Stuckat- 0 fault for example, it can be detected in both 3rd and 5$^{th}$ steps. But the fault address shouldn't be stored twice. So we propose an efficient method to solve the problem in BIAA module.

Figure 3.4 shows the flows of storing fault addresses. BIST detects whether the current address is faulty. If it is, BIAA checks whether the Fault-A-Mem overflows. If not, the current fault address should be compared with those already stored in Fault-A-Mem. Only if the faulty address isn't equal to any address in Fault-A-Mem, it can be stored. To simplify the comparison, write a redundant address into Fault-A-Mem as background. In this case, the fault address can be compared with all the data stored in Fault-A-Mem no matter how many fault addresses have been stored.

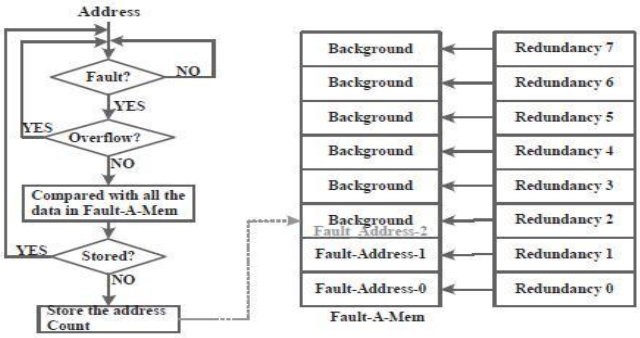


**Fig 9: Flows of Storing Fault Addresses**

At last, the BISR strategy is high-speed. Once a fault address is stored in Fault-A-Mem, it points to a certain redundant address. The fault addresses and redundant ones form a one-to-one mapping. Using this method, the BISR can quickly get the corresponding redundant address to replace the faulty one.

## RESULTS AND ANALYSIS

In this initially we can give rst=0 then we are not getting any output we can get xxxx and also we can give tm_nm=0 then the circuit works under normal mode condition means output depends on user giving input.

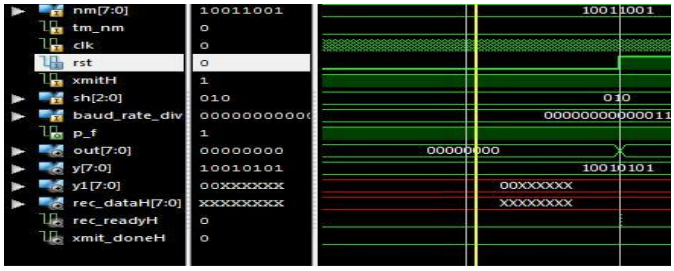


**Fig 10: Simulation Result for top module (rst=0,tm_nm=0)**

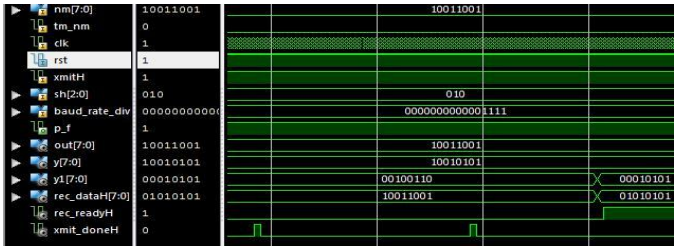


**Fig 11: Simulation Result for top module (rst=1,tm_nm=0)**

In this we can change rst=1 then we are not getting any output we can get xxxx because of CUT. In CUT we can get output when baud_div value reaches to 9,600 bps means we can give baud_div=4'h000f.

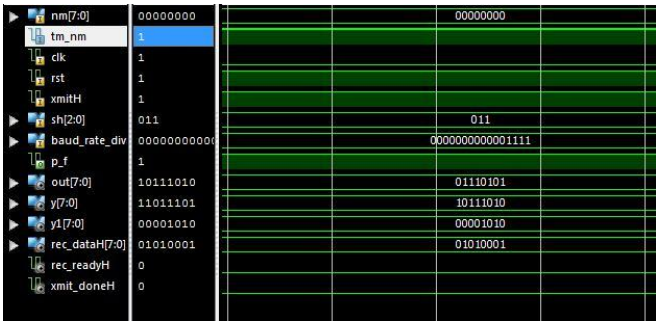


**Fig 12: Simulation Result for top module (rst=1,tm_nm=1)**

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

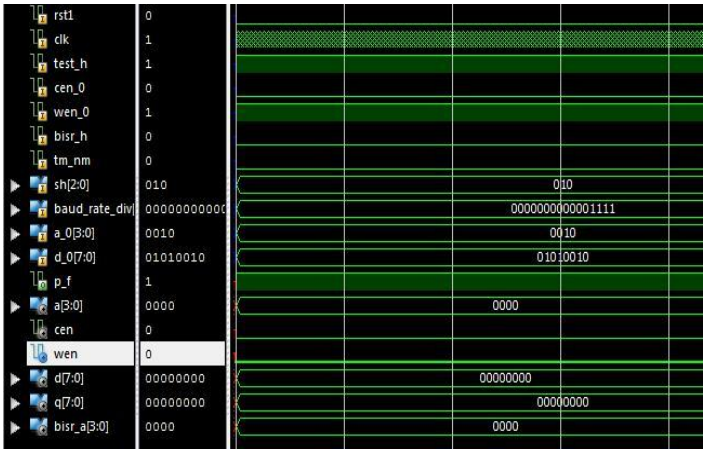


**Fig 13: Simulation Results for BISR rst=0(rd=0, wr=1)**

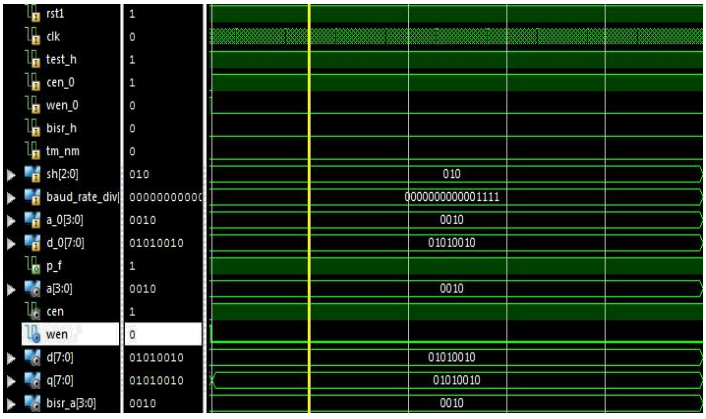


**Fig 14: Simulation Results for BISR rst=1(rd=1, wr=0)**

## CONCLUSION

An efficient BISR strategy for SRAM IP with selectable redundancy has been presented in this paper. It is designed flexible that users can select operation modes of SRAM. The BIAA module can avoid storing fault addresses more than once and can repair fault address quickly. The function of BISR has been verified by the post simulation. The BISR can work at up to 150MHz at the expense of 20% greater area.

## REFFERENCES

[1] J. Rajski, J. Tyszer, G. Mrugalski, and B. Nadeau-Dostie, "Test generator with preselected toggling for low power built-in self-test," in Proc. Eur.Test Symp., May 2012, pp. 1–6.

[2] Y. Sato, S. Wang, T. Kato, K. Miyase, and S. Kajihara, "Low power BIST for scan-shift and capture power," in Proc. IEEE 21st Asian Test Symp., Nov. 2012, pp. 173–178.

[3] E. K. Moghaddam, J. Rajski, M. Kassab, and S. M. Reddy, "At-speed scan test with low switching activity," in Proc. IEEE VLSI Test Symp., Apr. 2010, pp. 177–182.

[4] S. Balatsouka, V. Tenentes, X. Kavousianos, and K. Chakrabarty, "Defect aware X-filling for low-power scan testing," in Proc. Design, Autom. Test Eur. Conf. Exhibit., Mar. 2010, pp. 873–878.

[5] I. Polian, A. Czutro, S. Kundu, and B. Becker, "Power droop testing," IEEE Design Test Comput., vol. 24, no. 3, pp. 276–284, May/Jun. 2007.

[6] X. Wen et al., "On pinpoint capture power management in at-speed scan test generation," in Proc. IEEE Int. Test Conf., Nov. 2012,pp. 1–10.

[7] S. Kiamehr, F. Firouzi, and M. B. Tahoori, "A layout-aware X-filling approach for dynamic power supply noise reduction in at-speed scan testing," in Proc. IEEE Eur. Test Symp., May 2013, pp. 1–6.

[8] M. Nourani, M. Tehranipoor, and N. Ahmed, "Low-transition test pattern generation for BIST-based applications," IEEE Trans. Comput., vol. 57, no. 3, pp. 303–315, Mar. 2008.