LOW AREA PROGRAMMABLE MBIST ARCHITECTURE FOR TESTING FIFO's

BATHINI SANDHYA RANI, PG Scholar, CMR Institute of Technology, Hyderabad, TS, India
Mrs. MUNIPRAVEENA RELA, Associate Professor, CMR Institute of Technology, Hyderabad, TS, India
DR. PRADEEP KUMAR, Associate Professor, CMR Institute of Technology, Hyderabad, TS, India.

Abstract:

As semiconductor transistor dimensions shrink and increasing amounts of IP block functions are added to a chip, the physical infrastructure that carries data on the chip and guarantees quality of service begins to crumble. Many of today's systems-on-chip are too complex to utilize a traditional hierarchal bus or crossbar interconnect approach. NOC (Network on Chip) has become the better communication system with bus based network for complex designs reducing problems in bandwidth, power dissipation etc. Like SoC, NOC is also tested for faults and defects. NOC testing involves testing of routers and router links. In NOC, part of area is occupied by routers which are in turn occupied by FIFO buffers and Buffer Logic. Run time faults in FIFO and Buffer logic are large compared to others component faults of NOC. Therefore, testing of NOC involves testing of FIFO and BUFFER LOGIC to ensure no faults and physical defects also.

To avoid this, FIFO and routing logic is tested by transparent test algorithms targeting permanent FIFO faults. This algorithm is used for periodic testing of buffer at each location without effecting overall throughput of NOC except buffers

This paper proposed Programmable MBIST architecture for testing of FIFO's. It gives the flexibility of choosing March algorithm after fabrication too. Proposed algorithm is based on IEEE 1500 standard protocol.

Area of BIST circuit can also be reduced as compared to BIST with dedicated hardware when applied to large number of BIST algorithms.

1. Introduction

To meet the growing computation-intensive applications and the needs of low-power, high-performance systems, the number of computing resources in single-chip has enormously increased, because current VLSI technology can support such an extensive integration of transistors. By adding many computing resources such as CPU, DSP, specific IPs, etc to build a system in System-on-Chip, its interconnection between each other becomes another challenging issue. In most System-on-Chip applications, a shared bus interconnection which needs an arbitration logic to serialize several bus access requests, is adopted to communicate with each integrated processing unit because of its low-cost and simple control characteristics. However, such shared bus interconnection has some limitation in its scalability because only one master at a time can utilize the bus which means all the bus accesses should be serialized by the arbitrator. Therefore, in such an environment where the number of bus requesters is large and their required bandwidth for interconnection is more than the current bus, some other interconnection methods should be considered.

Such scalable bandwidth requirement can be satisfied by using on-chip packet-switched micro-network of interconnects, generally known as Network-on-Chip (NoC) architecture. The basic idea came from traditional large-scale multi-processors and distributed computing networks. The scalable and modular nature of NoCs and their support for efficient on-chip communication lead to NoC-based system implementations. Even though the current network technologies are well developed and their supporting features are excellent, their complicated configurations and implementation complexity make it hard to be adopted as an on-chip interconnection methodology. In order to meet typical SoCs or multi-core processing environment, basic module of network interconnection like switching logic, routing algorithm and its packet definition should be light-weighted to result in easily implemental solutions.

2. Low power Programmable MBIST Design Testing for FIFO's

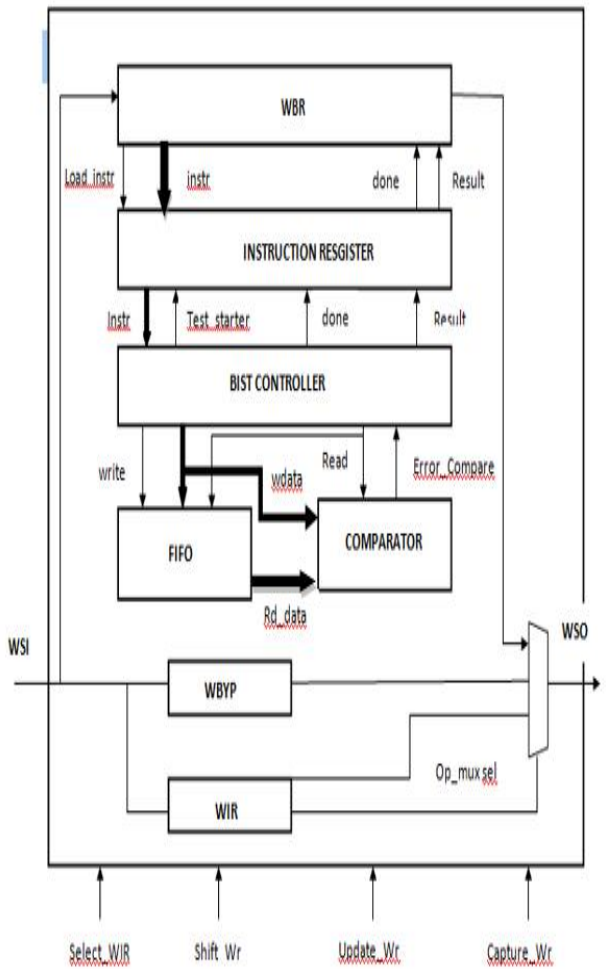

Figure 2.1 Block diagram of programmable MBIST testing for FIFO's

In figure 2.1, It have different block such as WBR (Wrapper Boundary Register), WIR (Wrapper Instruction Register), WBYR (Wrapper Bypass Register), FIFO (First In First Out), COMPARATOR, BIST CONTROLLER, INSTRUCTION RESGISTER.

Wrapper boundary Register (WBR) is shown in figure 2.2, there are many standard cells which contain memory. They transfer the data in serial/parallel format. The standard cell surrounded the DUT to transmit the data in serial/parallel format.

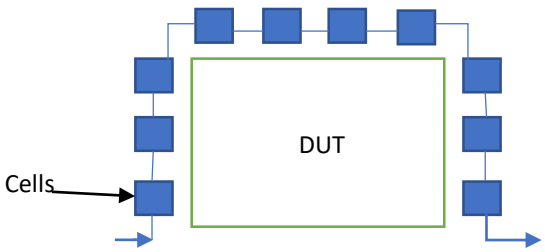

Figure 2.2 WBR (wrapper Boundary Register)

There are three modes to transmit the data into DUT without changing any design and any module. The figure 2.3 is standard cell, this three modes can operate on standard cell.

1. Shift_wr
2. Update_wr
3. Capture_wr

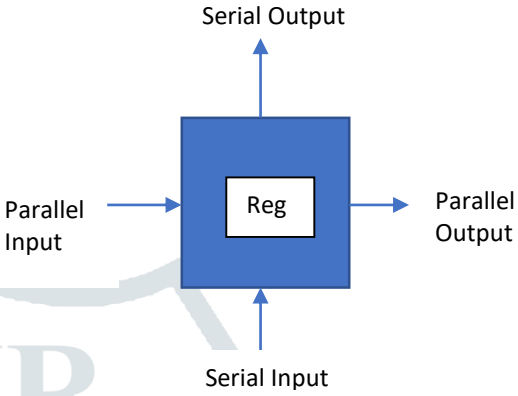

Figure 2.3 Standard cell

1.Shift_wr mode:

If shift_wr is set to 1 then serial input data is stored in Register and serial output is update with register data. i.e Reg=SI ; SO = Reg

2. Update_wr mode:

If update _wr is set to 1 then parallel output is update from Register. i.e PO=Reg;

3. Capture_wr mode:

If capture_wr is set to 1then register is update the date from Parallel Input. i.e Reg=PI

Wrapper Instruction Register is operated by Two Mode

1. BYPASS Mode
2. EXTEST mode

Using SELECT_WIR pin, we can select the WIR (wrapper instruction register).Write "001" to WIR for EXTEST instruction.Similarly write "000" to WIR for BYPASS INSTRUCTION.

Programmable MBIST:

Most of Memory BIST approaches concerns the programmability of the memory test algorithm. To enabling programmability of all components of memory test, test algorithm, test data, address sequence.

The programmable memory BIST proposed has several advantages:

- It enables programming both test algorithms and test data.
- It implements test algorithm programmability at low cost, by extracting

the different levels of hierarchy of the test algorithm and associating a hardware block to each of them, resulting on low cost hardware.

- It enables low-cost implementation of full-data programmability by adapting the transparent memory test approach in a manner that uses the memory under test for programming the test data.

The architecture for programming march test algorithms proposed in the fig. This architecture uses an instruction register specifying the current match test sequence by means of several fields indicating.

In the block diagram, programmable MBIST contains 4 important parts.

1. BIST controller
2. FIFO
3. Comparator
4. Instruction Register

BIST Controller:

Built-in self-test (BIST) is a design technique that allows a circuit to test itself It is a set of structured-test techniques for combinational and sequential logic, memories, multipliers and other embedded logic blocks. The principle is to generate test vectors, apply them to the circuit under test or device under test, and then verify the response. Being an automated testing, BIST enables testing at high speed and high fault coverage.

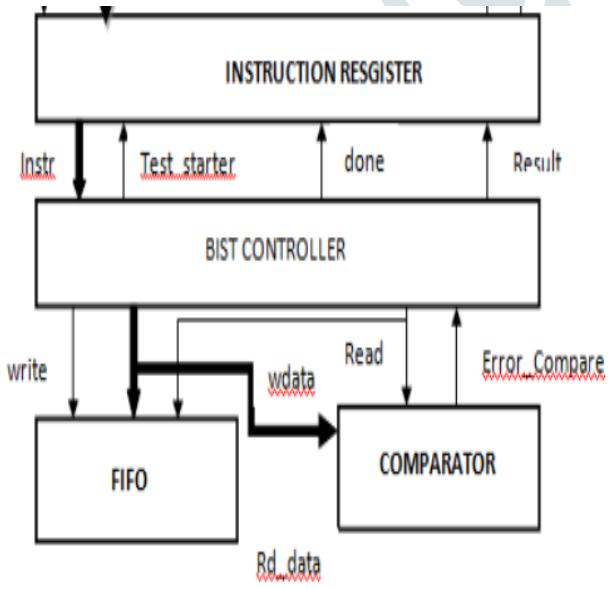

Figure 2.4 Block diagram of programmable MBIST

BIST controller coordinates the operations of different blocks of the BIST. Based on the test mode(TM) input to the controller, the system either operates in the normal mode or in thetest mode. In this paper we explain an implementation of a restart able logic BIST controller for a combinational logic circuit using VHDL. It allows us to suspend the signature generation at any desired point in the test sequence. In this case, the BIST circuit is considered to comprise hold logic and a signature generation element. The hold logic will be implemented such that an external signal (HOED) can temporarily suspend signature generation in the signature generation element at specified times during the BIST session.

FIFO: Every memory in which the data word that is written in first also comes out first when the memory is read is a first-in first-out memory

Shift register – FIFO with an invariable number of stored data words and, thus, the necessary synchronism between the read and the write operations because a data word must be read every time one is written

Exclusive read/write FIFO – FIFO with a variable number of stored data words and, because of the internal structure, the necessary synchronism between the read and the write operations

Concurrent read/write FIFO – FIFO with a variable number of stored data words and possible asynchronism between the read and the write operation

Comparator:

It compares the write data (wr_data) and read data (rd_data) of both BIST controller and FIFO.

It checks the error in the information and gives signal as PASS/FAIL to BIST controller based on result.

Instruction Register:

Instruction Register

Operatin_1	Priority_1	Opertion_0	Priority_0	No. of Operations	Data (16bits)	Test Enable	Test Done	Test Result
23	22	21	20	19	18.....3	2	1	0

It is 24 bit Instruction Register, load the instructions are based on Operation and Priority pins. Data is taken from WBR (Wrapper Boundary Register).Test done and Test Result is update by BIST Controller.

3. Implementation

Programmable MBIST for Efficient in filed testing of FIFO in NOC contains Top module and DUT.

- In Top Module, again there are 3 modules
  - WRB (wrapper Boundary Register)
  - WRI (wrapper instruction register)
  - WBYR (wrapper bypass register)
- In DUT, there are 4 modules
  - FIFO
  - BIST Controller
  - Instruction Register
  - Comparator.

3.1 DUT Implementation:

In this project, DUT (design under Test) is designed shown in figure 3.1

Specification are INSTRUCTION_IN, CLK, LOAD_INSTRUCTION, RST, TEST_DONE, and TEST_RESULT.

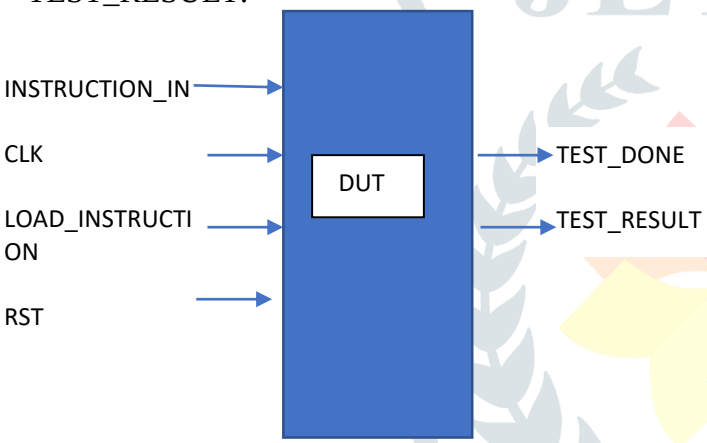

Figure 3.1 Design For Dut

FIFO Implementation:

FIFO – first in first out, shown in figure 3.2 it is used for communication in between modules on NoC. It used as temporary register which transmit the instructions.

Specification are DATA_IN, Calk, Rd, Rst, WR, DATA_OUT, FIFO_EMPTY, and FIFO_FULL.

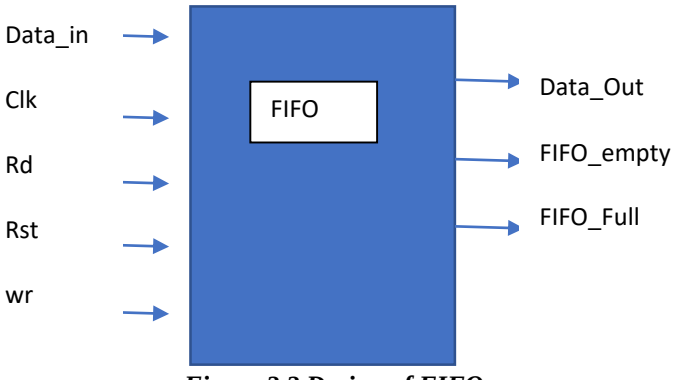

Figure 3.2 Design of FIFO

BIST Implementation

BIST is built in Self in test shown in figure 3.3, it generates test patterns, and controls the processor and instructions

Specifications are INSTRUCTION, CLK, FIFO_EMPTY, FIFO_FULL, RST, WRITE_DATA, READ, TEST_DONE, TEST_STARTED, and WRITE.

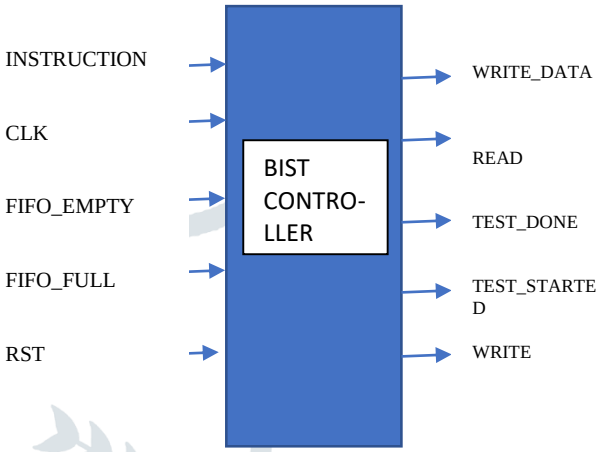

Figure 3.3 Design Of BIST Contorller

Instruction Register Implementation:

Instruction register is shown in figure 3.4 .It is 24 bit register, based of operations and priority information is processed. Instructions are taken from boundary register.

Specification are INTRUSCTION_IN, CLK, LOAD_INSTRUCTION, RST, TEST_DONE, TEST_RESULT_IN, TEST_STARTED, INSTRUCTION_OUT, TEST_DONE, and TEST_RESULT.

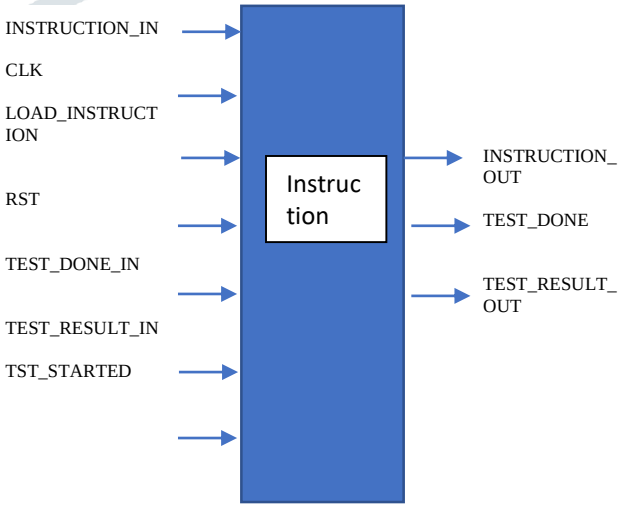

Figure 3.4 Is Design of Instruction Register

Comparator Implementation:

Comparator is shown in figure 3.5. It compare instruction register data and wrapper boundary register (WBR) data and gives the error result.

Specification are ACTUAL_DATA, EXPECTED_DATA, CLK, COMPARE, RST, and ERROR

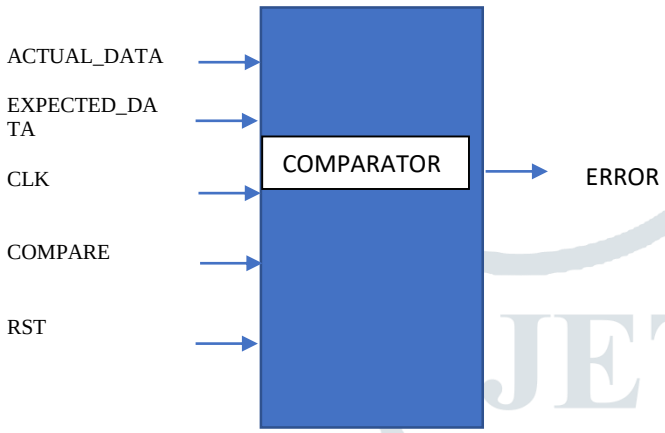

Figure 3.5 Design Of Comparator

3.2 Top Module Implementation:

Top module is shown in figure 3.6

Specification are CAPTURE_WR, SELECT_WR, SHIFT_WR, TRANSER_DR, UPDATR_WR, WRCK, WRSTN, WSI, and WSO.

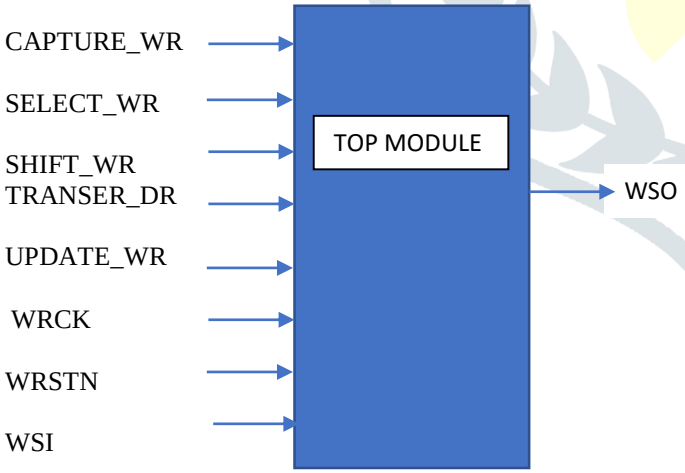

Figure 3.6 design of top module

4. Simulation And Synthesis Results

Figure 4.1 is the simulation wave form of project, Here the instruction is 37ffff (hexa decimal notation) give as input through WSI(wrapper serial input) to WBR (wrapper boundary register). Output is WSO(wrapper serial output) also in serial format.

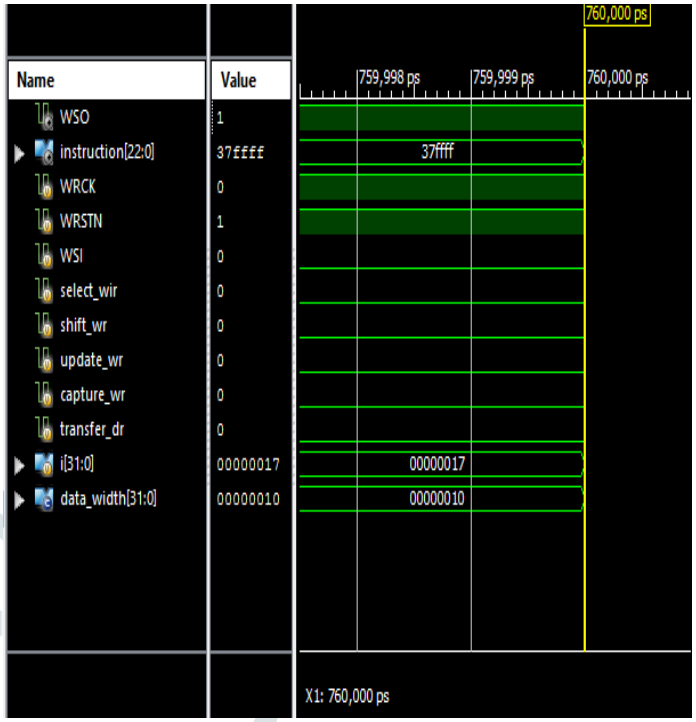

Figure 4.1 Wave form of Top Module (programmable MBIST for efficient in-field testing of FIFO buffers in NOCs)

Figure 4.2 is the simulation wave form of DUT.DUT consist of Instruction register, FIFO, BIST controller, Comparator. Here the instructions is taken from WBR to INSTRUCTION REGISTER, that instruction is controlled by BIST over the FIFO. BIST gives Test_ result, Test_done output to instruction.

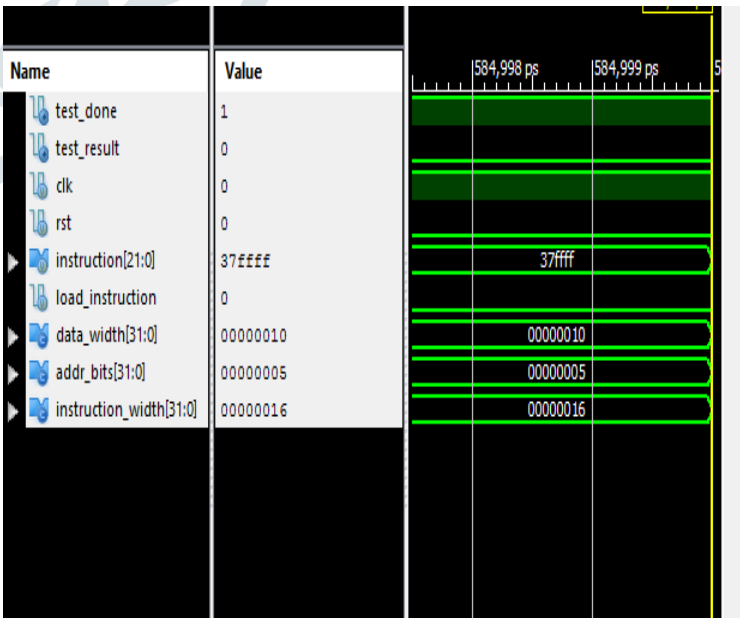

Figure 4.2 wave form of DUT (Design under test)

5. Conclusion

In Noc's, FIFO's are used to communicate in between different blocks. FIFOs are controlled by BIST and INSTRUCTION Registers .Any block has to change in NOCs, it would be tough to reconstruct the design and change the instructions.

To overcome this, Wrapper boundary register is used. It is based on IEEE 1500 standard protocol.

Without changing the design, without disturbing the Noc we can change the properties using Wrapper Boundary Register.

Advantages:

- Complexity is reduced in processing the instructions
- change the instructions without disturb of NOC

Disadvantages:

- Delay may increase due to serial communication in wrapper boundary register

Future scope:

As future work, we would like to modify the proposed FIFO testing technique that will allow incoming data packets to the router under test without interrupting the test.

Author Profile:

Bathini Sandhya Rani She received Bachelors' of Degree in 2015 from Electronics and Communication of Engineering from MallaReddy college of Engineering for women. She is pursuing M.Tech in VLSI System Design from CMR Institute of Technology.

Mrs. M. Munipraveena Rela M.Tech. (Electronics) , (PhD). She is working as Associate Professor in CMR Institute of Technology.

Dr.Pradeep kumar, M.Tech. (Applied Electronics), Ph.D. He is working as Associate Professor, Co-Dean Innovation & Incubation Cell in CMR Institute of Technology.

Reference:

1. Semiconductor Industry Association, "International technology roadmap for semiconductors (ITRS), 2003 edition," Hsinchu, Taiwan, Dec.2003.
2. C. Stapper, A. Mclaren, and M. Dreckman, "Yield model for Productivity Optimization of VLSI Memory Chips with redundancy and Partially good Product," IBM Journal of Research and Development, Vol. 24, No. 3, pp. 398-409, May 1980.
3. W. K. Huang, Y. H. shen, and F. lombrardi, "New approaches for repairs of memories with redundancy by row/column deletion for yield enhancement," IEEE Transactions on Computer-Aided Design, vol. 9, No. 3, pp. 323-328, Mar. 1990.
4. P. Mazumder and Y. S. Jih, "A new built-in self-repair approach to VLSI memory yield enhancement by using neuraltype circuits," IEEE transactions on Computer Aided Design, vol. 12, No. 1, Jan, 1993.
5. W. J. Dally and B. Towles, "Route packets, not wires: On-chip interconnection networks," in *Proc. 38th Annu. Design Autom. Conf.*, 2001, pp. 684–689.
6. Bondavalli, S. Chiaradonna, F. Di Giandomenico, and F. Grandoni, "Threshold-based mechanisms to discriminate transient from intermittent faults," *IEEE Trans. Comput.*, vol. 49, no. 3, pp. 230–245, Mar. 2000.
7. M. Radetzki, C. Feng, X. Zhao, and A. Jantsch, "Methods for fault tolerance in networks-on-chip," *ACM Comput. Surv.*, vol. 46, no. 1, pp. 1–38, Jul. 2013, Art. ID 8.
8. S. Ghosh and K. Roy, "Parameter variation tolerance and error resiliency: New design paradigm for the nanoscale era," *Proc. IEEE*, vol. 98, no. 10, pp. 1718–1751, Oct. 2010

.

JETIR