A RELIABLE TECHNIQUE OF FAULT TOLERANCE THROUGH CONFIGURATION FRAMES

[1]**Suman Lokurte**
[1]MTECH PG Scholar (VLSI Design and Embedded Systems)
[1]Department of VLSI Design and Embedded Systems,
[1]VTU PG Center, Gulbarga, Karnataka, India

Abstract - Fault tolerance is the system's ability to perform its specified operation in the presence of faults to increase its reliability. In the project, PDR based technique is used to tolerate stuck at faults in configurable logic blocks. Configuration frames are reconfigured by shifting the configuration bits such that the faulty SRAM cells are replaced by the spare SRAM cells to tolerate the faults. Since pre-allocated CLBs are not required, this technique has very less area overhead. Modalism is used to simulate the design.

Keyword - Configuration frames, SRAM based FPGA, fault tolerance, Simulation.

I. INTRODUCTION

Fault tolerance is the ability of a device to normally operate given that there is a presence of resources which are malfunctioning, defects and faults. Methods to integrate fault tolerance in FPGAs include providing redundancy and FPGA reprogramming to get over from the fault. The reliability is most of the time achieved through TMR (triple modular redundancy) hardware which is more often not so effective solution in terms of cost [1] [2]. Architecture of an SRAM based FPGA proposed by Ito and Doumar [3] included a method for enhancing the yield as well as fault tolerance. Lach [4] [5] proposed a technique called offline tolerance of logic that requires partitioning FPGA into array of tiles. Electronic designers of spacecraft and aircraft demand for FPGA reconfigurable hardware due to their flexibility in reaching many requirements like very high performance, faster turnaround duration and low cost [6]. Hardware reconfigurable systems, such as the SRAM based FPGA has succeeded in making it possible to integrate the fault tolerance more cheaply into the systems. Whenever a charged particle collides with a memory cell within the program memory, its stored value gets inverted which alters the functionality of a design resulting in SEU. Upsets affect combinational logic within FPGA. It evokes a change in bit value in LUT's cells or the cells which control the routing. This kind of upset has a lasting effect that could be distributed to other components of the circuit due to the fact that the implemented hardware can be modified. To maintain the correct operation of the system, whenever the SEU is observed in FPGA by scrubbing of bit stream, partially dynamic reconfiguration can be used in FPGA. PDR feature is available in FPGA implementations. By using this feature, there is a possibility of reloading of configuration memory and modifying it while the system is properly working. This property is used for designs which have high dependable systems.

II. SRAM BASED FPGA

The layout of SRAM based FPGA consists of a set of configurable logic resources and a set of interconnections. The significance of configuration is that it selects a path for signals to flow from input to output. Signals travel through logic blocks to achieve a devoted application. In SRAM based FPGA, this standard is supported by the use of SRAM cells which contains the configuration data. Fig.1 shows the two layers of SRAM based FPGA, Configuration layer and Operative layer.

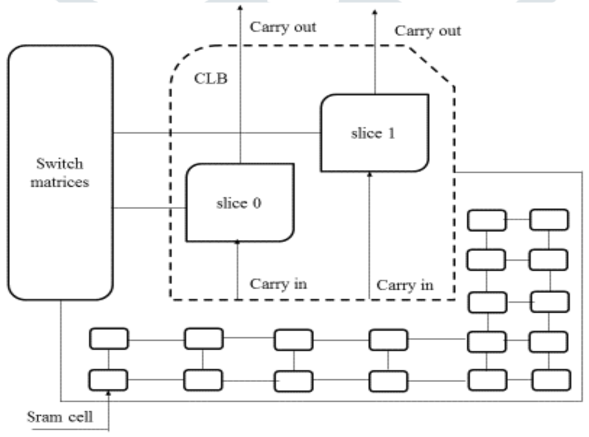


Fig.1. Partitioning of SRAM based FPGA

Operative layer consists of applications made up of sequential and combinational resources. To activate these applications, another layer called configuration layer is used. It includes interconnect points which are configurable. These points are either loaded or unloaded to activate the specific application. Configuration memory is formatted as a set of frames in Virtex 5 FPGA. All the frames are not contiguous, but are tiled all over the device. These are the smallest segments of the configuration memory that can be addressed.

Frames are reconfigured one at time. It is the smallest set of bits which can be read or written one at a time. In Virtex 5 FPGA, the size of each frame is 1312 bits [7]. Each frame covers nearly 20 CLBs. So, each reconfiguration alters minimum 20 CLBs. Fig.2 shows configuration words present in the Bit stream and a frame showing configuration bits within it. There are 41 words within a frame. Each word is of 32 bits.

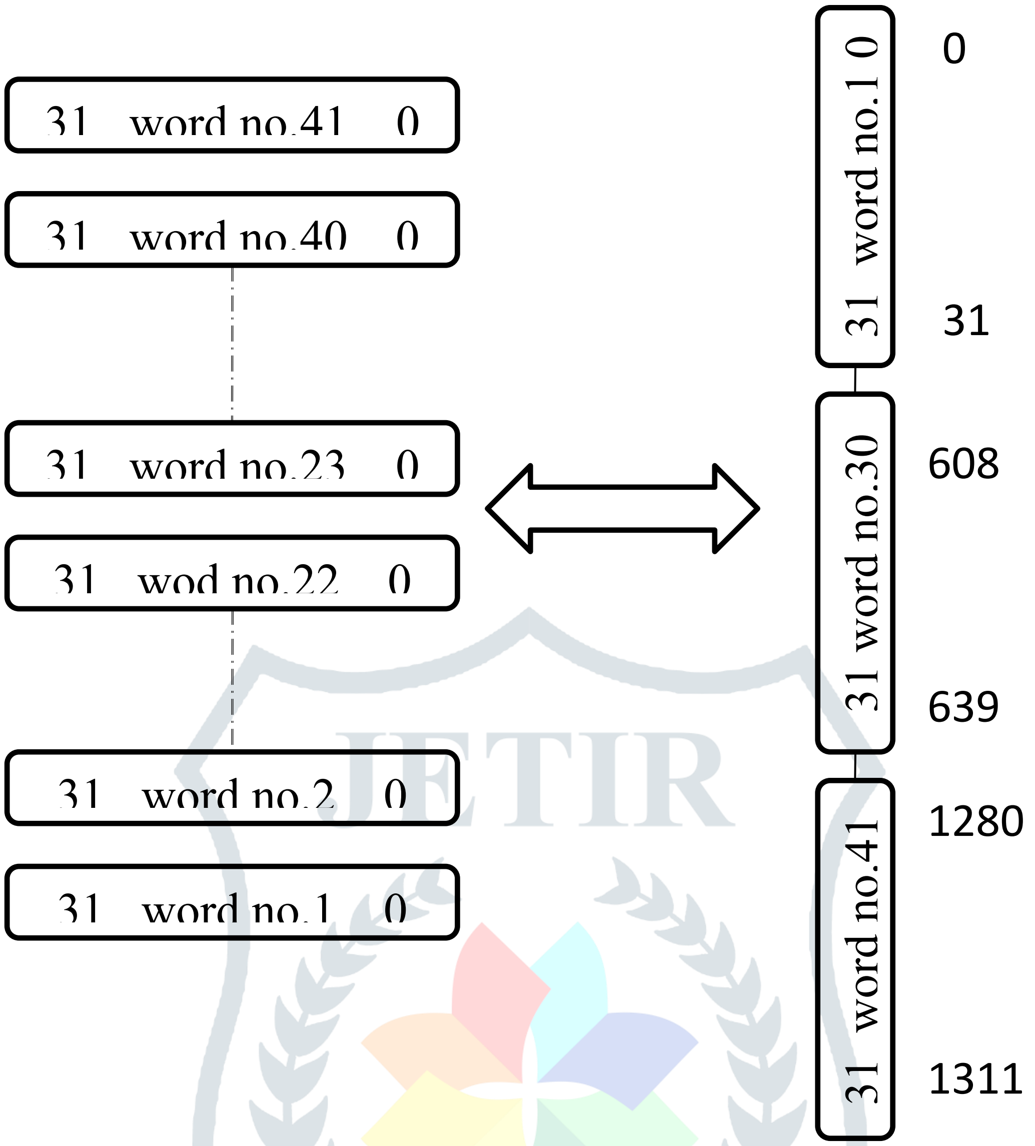
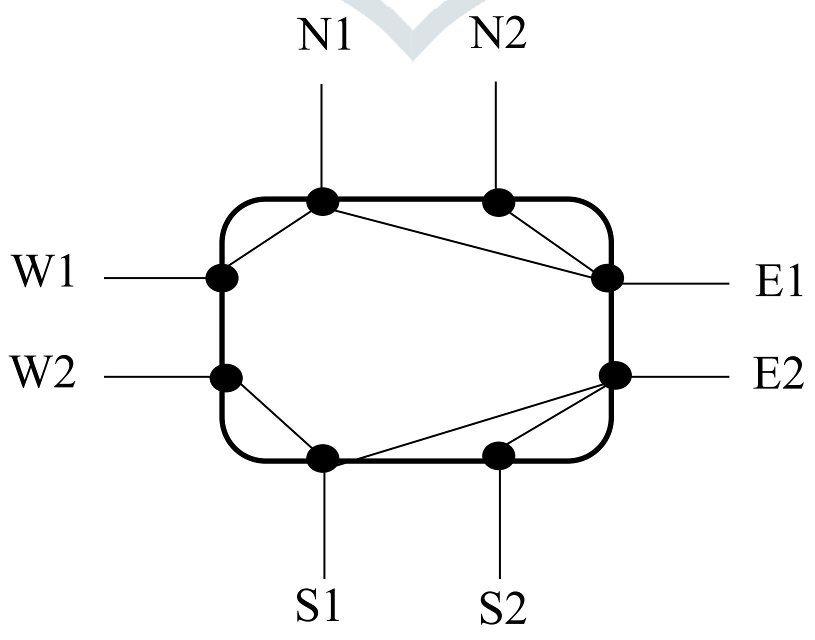


Fig.2 Configuration data words and data bits within a frame

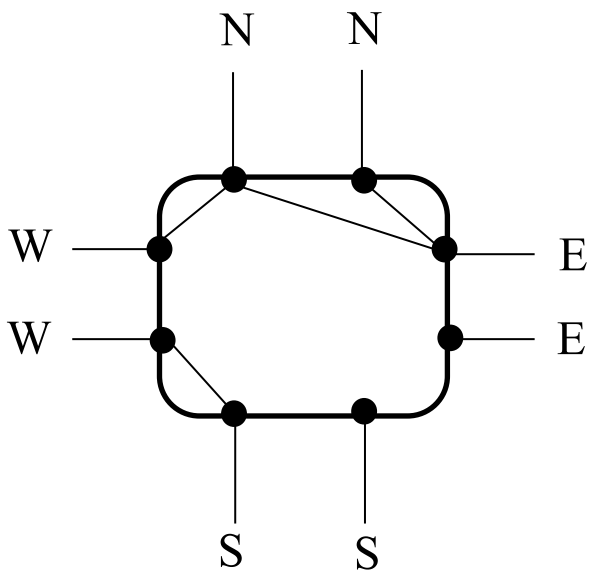
III. FAULTS IN SRAM BASED FPGA

<u>**Fault models considered for this paper are –**</u>

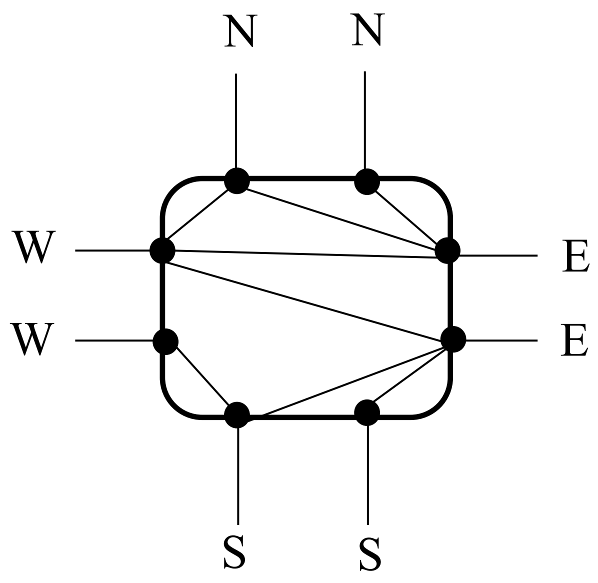
1. Stuck off and stuck on faults: - Faulty value in the SRAM cell which is connected to the gate of a pass transistor causes it to turn on or off unintentionally, irrespective of the value present in the bit stream. Fig.3 illustrates these faults.
2. Look up table stuck at 0 or stuck at 1 fault: - The value of a SRAM cell at the LUT entry within a configurable logic block is always stuck at 1 or 0 value irrespective of that present in bit stream. Fig.4 illustrates these faults.



(a)

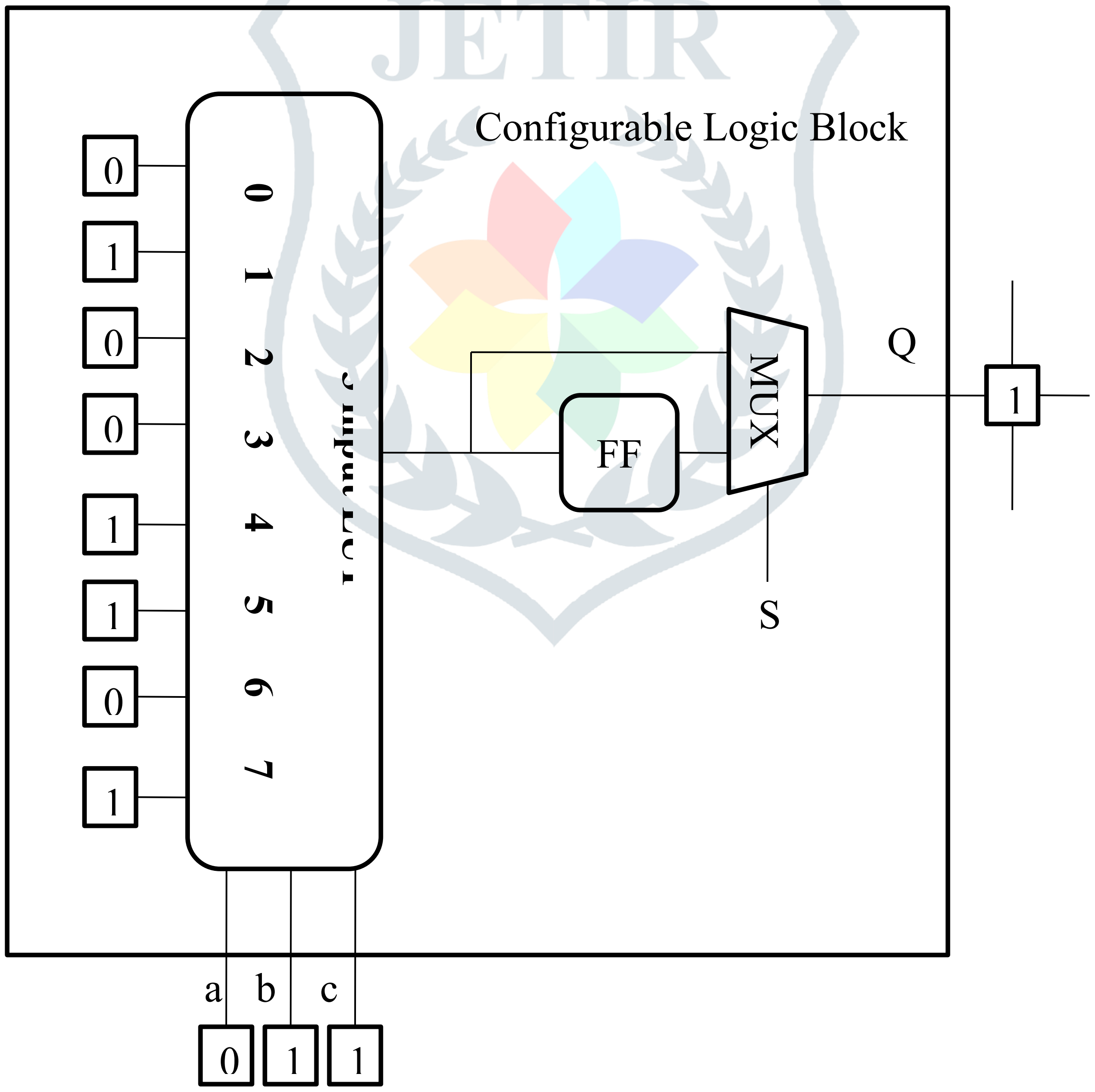


(b)

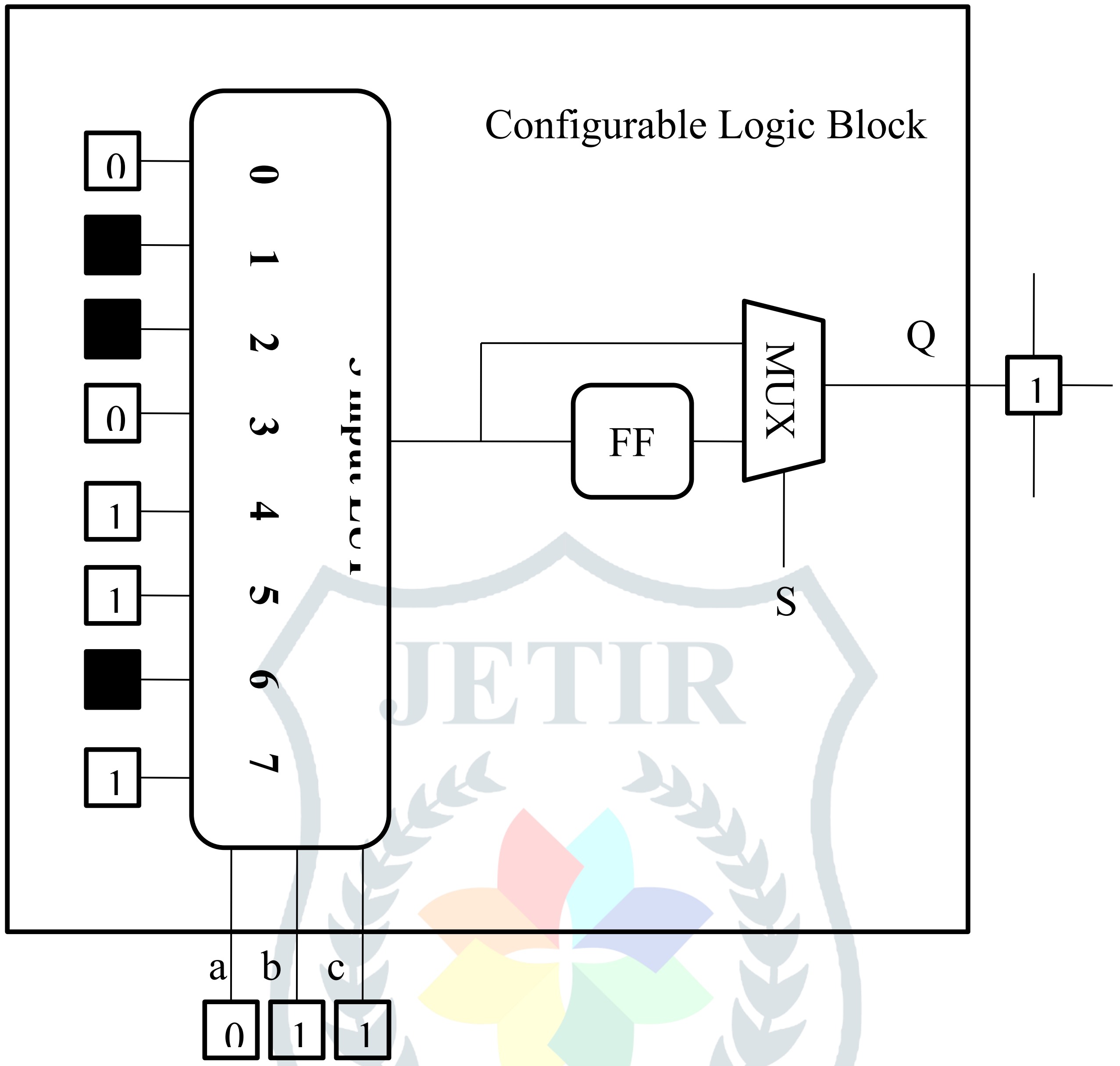


(c)

Fig.3 Stuck-at faults in routing resources: (a) Routing without faults. (b) Signals missed due to faults. (c) Signals misrouted due to faults.



(a)



(b)

Fig.4 Stuck-at faults in 3-input LUT: (a) 3 input LUT implementing Q function. (b) LUT affected by Stuck at faults.

IV. PROPOSED TECHNIQUE FOR FAULT TOLERANCE

Once the location of faults is detected through testing in the SRAM based FPGA, it is partially reconfigured in such a way that a faulty SRAM cell will be replaced with its neighbour. The neighbour is in turn replaced by its neighbour and so on till a spare SRAM cell is reached. This is shown as covering graph in fig.5. Every 32 bit word in a frame contains spare SRAM cells.

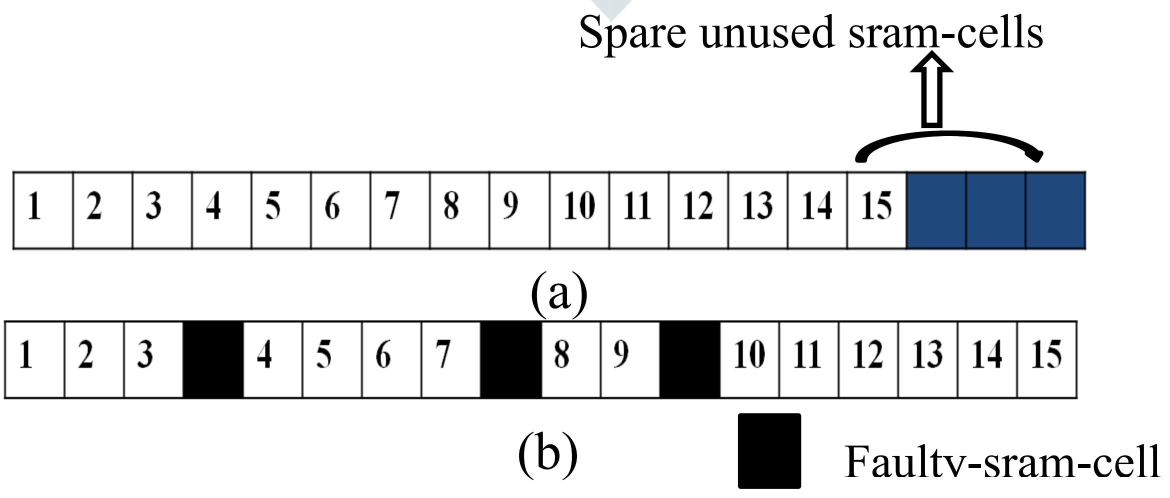


Fig.5 (a) Covering graph of FT reconfiguration technique using used SRAM-cells and spare SRAM-cells. (b) Reconfigured graph after the occurrence of faults in SRAM-cells.

Faulty SRAM at location bit 4 is replaced by the SRAM cell at location 5, which is in turn replaced by the SRAM cell at location 6 and so on. Spare SRAM cells will be used in place of faulty SRAMs. This approach is a new class of fault tolerance where the partially faulty components (Routing resources and CLBs) are reused. Because if a fault occurs in an interconnect point (Fig.3) or a LUT (Fig.4), it is guaranteed that complete CIP or LUT will not be defective, only a part of it will be defective. Faults in routing resources misroutes or disconnects the signals as shown in fig.3. Stuck at faults in one or more LUT entries modify the logic functionality of a function as shown in fig.4.

The basis of this technique is to replace only the faulty part of a partially faulty LUT instead of replacing the whole CLB. Shifting operation is carried out at the level of configuration layer as shown in fig.6.

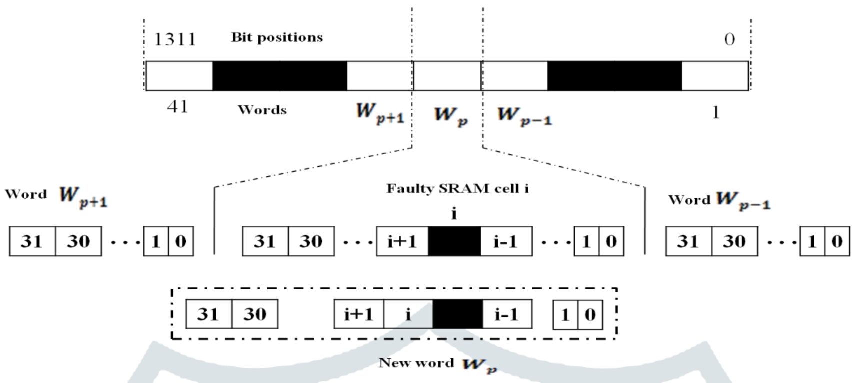


Fig.6 Reconfiguration – avoiding faulty SRAM cells to form fault free words

A frame is the quantity of configuration information which can be accessed at a time. It is a vertical stack consisting of 1312 bits. These bits cover the complete height of a row. Each row has a stack of blocks such as 40 IOBs, 20 CLBs and 4 block RAMs. So, out of 1312 bits of a frame, 32 bits reside in HCLK tile, 640 bits reside in the blocks above the HCLK tile and remaining bits reside in the blocks below HCLK tile. Words made up of 32 configuration bits of a frame are shifted into frame-data-register. To make a frame of 1312 configuration bits, 41 words are necessary. Fig.2 shows 41 words of a bit stream counted from 1 to 41. It also shows how they are arranged to form a frame of 1312 bits.

Let the number of configuration frames of the SRAM based FPGA be M. F(C) be a portion of the design which is mapped to a frame F_M such that 1 < F_M< M. C= ($c_{1,}$ $c_{2,}$…., c_k). On mapping C with S such that c_i= s_j or c_i= $\sim s_j$, a portion of design will be represented by H ($c_{1,}$ $c_{2,}$…., c_k) = F ($s_{1,}s_{2,}$…., s_k). If a fault occurs at $s_j th$ SRAM cell, then the word W_m is said to be faulty. The frame reorganizes its words using a technique named partial dynamic reconfiguration to avoid faulty SRAM cells. If the fault is found at $s_j th$ SRAM cell, then H becomes

$$H(c_1, c_2,\ldots., c_{j-1}, c_j, c_{j+1},\ldots., c_k) = F(s_1, s_2,\ldots., s_{j-1}, s_q, s_{j+1}, s_{j+2}\ldots., s_k) \qquad (1)$$

Here, s_q is the SRAM cell at the rightmost end of the word. Fig.6 shows this concept.

In this way this technique reorganizes the frame Fm in such a way that all 41 words are fault free as illustrated in fig.6. This is possible because of pre-allocated spare cells in every word as shown in fig.5 as well as unused SRAM cells.

V. RTL VIEWS

The RTL view of Configurable logic block with fault tolerance technique block is as shown in fig.7. It gives graphical representation at the register transfer level.

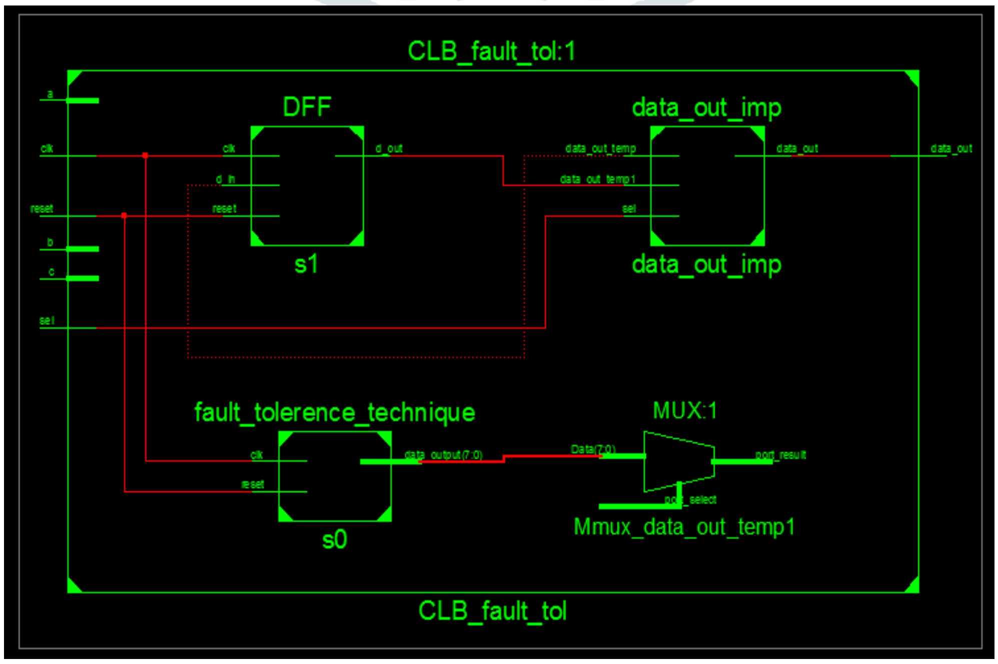


Fig.7 RTL view of Configurable logic block with FT technique

Configurable logic block implements a function Q = $a\bar{b}+\bar{b}c+ac$. Inputs to CLB are clk (clock), reset (reset), a, b, c (input variables of Q function), sel (select input to select latched or unlatched output. Output of CLB is the correct output of Q function after tolerating faults. Truth table of Q function is as shown in table 1.

Table 1. Truth table of Q function

Inputs	Output
a b c	Q
0 0 0	0
0 0 1	1
0 1 0	0
0 1 1	0
1 0 0	1
1 0 1	1
1 1 0	0
1 1 1	1

Inputs to fault tolerance technique are CLK (clock), reset (reset) and faulty word whose bits represent the LUT entries provided by some fault free and some faulty SRAM cells are declared as registers. This technique is applied after knowing the locations of faults. Thus a register representing positions of faults is declared. Two arrays are declared, one of which stores faulty locations and the other stores fault free locations. A flag is used to determine the faulty location. If it is set to one, next fault free location is determined and then the fault free value of the former is stored in later location. And spare cells are used instead of faulty cells to output the results. Depending on input combination a, b and c, the correct output value is displayed at the output port. RTL view of fault tolerance technique block is as shown in fig.8.

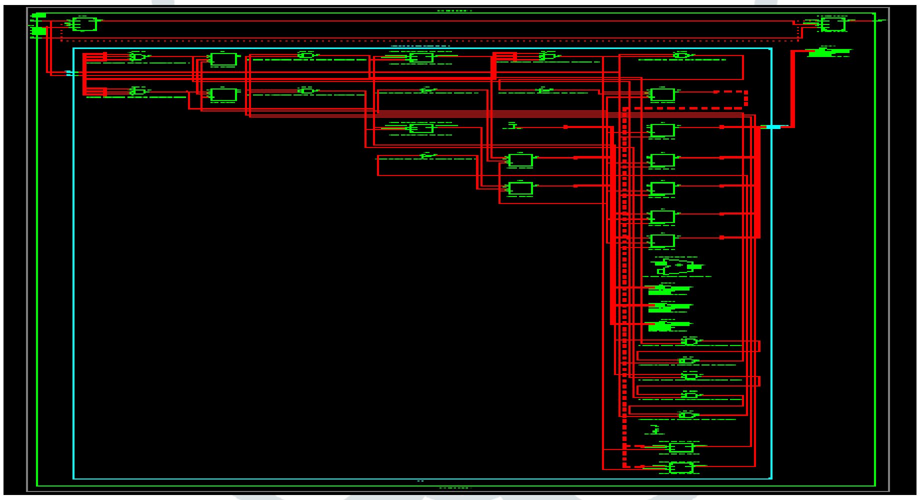


Fig.8 RTL view of fault tolerance technique

VI. SIMULATION RESULTS

Normal CLB

Fig.9 shows the output of a configurable logic block implementing a Q function with no faults in any SRAM cells configuring the look up table of CLB.

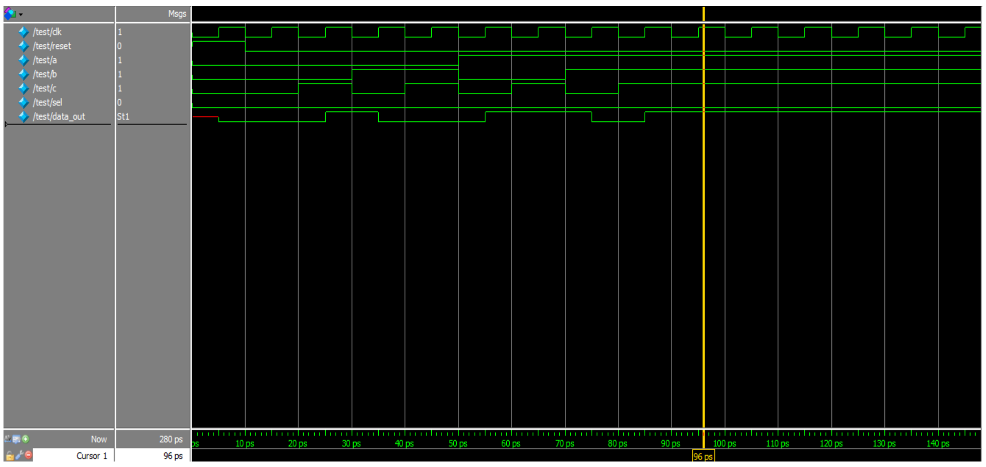


Fig.9 Output for Normal CLB

Faulty CLB

Fig.10 shows the output of a configurable logic block implementing a Q function with faults in some SRAM cells configuring the look up table of a CLB.

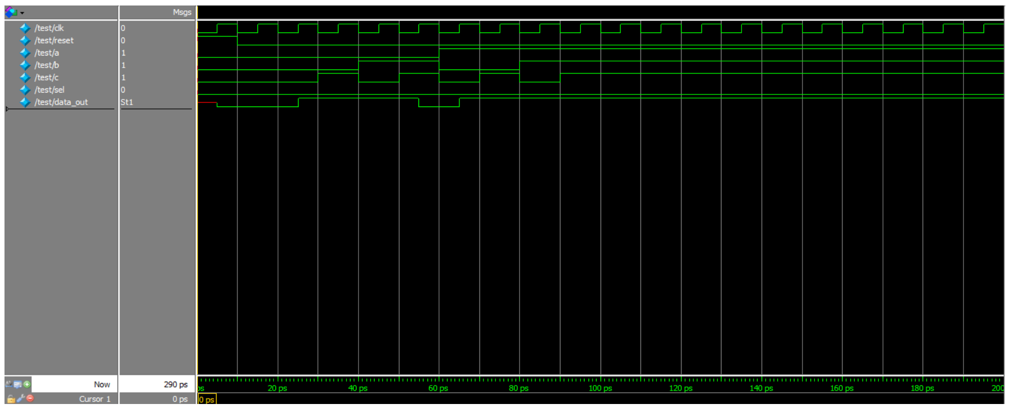


Fig.10 Output for Faulty CLB

Fault tolerated Proposed CLB

Fig.11 shows the fault tolerated correct output of the faulty CLB after reconfiguration.

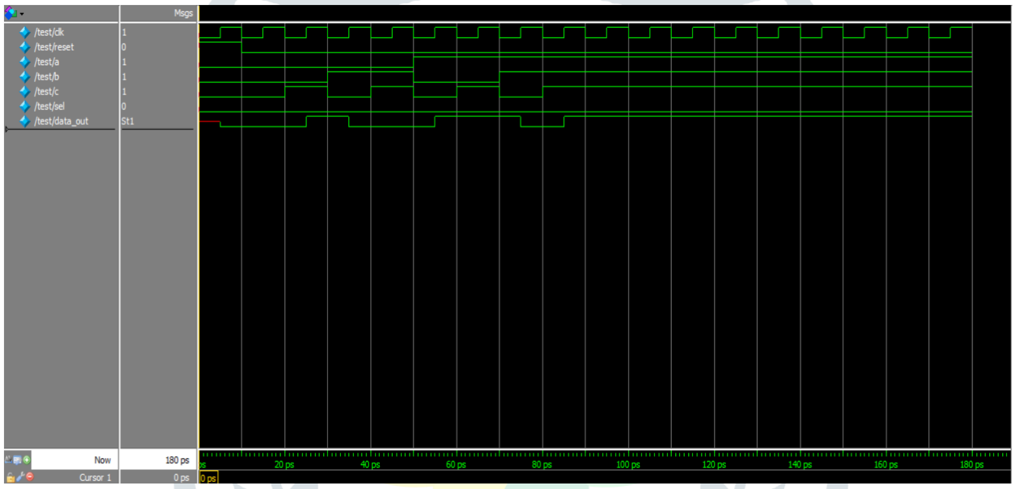


Fig.11 Output for Proposed CLB

VII. CONCLUSION

Fault tolerance technique is proposed to increase the reliability of FPGAs by reconfiguring the faulty words of a frame. Shifting operation is done in every word such that the faulty cells are avoided and spare cells are used. Any faults which affect the cells configuring the CLBs are avoided. The design of this technique is simulated in Modelsim. The design is implemented on Elbert V2 development board featuring Spartan 3A FPGA. The results of this technique have been matched with the expected results, which was the aim of the project. This technique can also be used to reconfigure wiring defects to increase reliability of SRAM based FPGA. This approach has very less area and performance overhead.

REFERENCES

[1] S. D'Angelo, C. Metra, S. Pastore, A. Pogutz, and G. Sechi, "Fault tolerant voting mechanism and recovery scheme for TMR FPGA-based systems," in Proc. IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems, 1998, pp. 233–240.

[2] L. Sterpone and M. Violante, "A new reliability-oriented place and route algorithm for SRAM-based FPGAs," IEEE Transactions on Computers, pp. 732–744, 2006.

[3] A. Doumar and H. Ito, "Detecting, diagnosing, and tolerating faults in SRAM-based field programmable gate arrays: a survey," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 11, no. 3, pp. 386–405, 2003.

[4] J. Lach, W. Mangione-Smith, and M. Potkonjak, "Low overhead faulttolerant FPGA systems," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 6, no. 2, pp. 212–221, 1998.

[5] J. Lach and W. Mangione-Smith, "Enhanced FPGA reliability through efficient run-time fault reconfiguration," IEEE Transactions on Reliability, vol. 49, no. 3, pp. 296–304, 2000.

[6] N. Howard, A. Tyrrell, and N. Allinson, "The yield enhancement of field-programmable gate arrays," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 2, no. 1, pp. 115–123, 1994.

[7] Xilinx, "Virtex 5 FPGA Application and Configuration User Guide," Xilinx User Guide (version 3.8) UG191, vol. 2, 2009.