Memristor-Based Fault-Tolerant Structure With Concurrent Reconfigurability

Debarchana Jena,
College of Engineering Bhubaneswar

***Abstract*— One of the primary concerns with electronic equipment is fault tolerance, particularly when it comes to managing and overseeing operations in harsh settings. This letter presents a novel hybrid approach based on memristors that introduces a new concurrent reconfigurable structure by combining the appealing aspects of both static and dynamic methods. Compared to prior research, the simulation results demonstrate that the suggested structure tolerates both transient and permanent errors during run-time with little latency, power consumption, and area overhead. During the fault recovery phase, the suggested approach also achieves the lowest possible silicon protection factor with zero stoppage time.**

I. Introduction

RECENT developments in high-tech industries have led to the increase of the demands for designing extremely radiation tolerant integrated circuits (ICs). Single event effects and total ionizing dose are some of the radiation effects in complementary metal oxide semiconductor (CMOS) ICs which usually lead to transient and permanent faults occurrence in electronic devices [1]. In general, the existing fault tolerance approaches in electronic circuits can be divided into three categories: 1) static; 2) dynamic; and 3) hybrid redundancy methods. In static methods the continuous operation of the system is guaranteed, because they cover and hide faults, instead of detecting them. The most common form of static redundancy is triple modular redundancy (TMR) which includes three copies of the original circuit and a majority voter; and it tolerates only one permanent fault. The dynamic redundancy methods are defined by the detection of faults and taking responsibility of some activities for recovery. Built-in-self-repair (BISR) technique is one of the popular active techniques to tolerate the permanent faults in electronic circuits which is implemented in four steps: 1) fault detection; 2) fault diagnosis; 3) fault isolation; and 4) redundancy allocation steps. The dynamic partial reconfiguration property of the field-programmable gate arrays (FPGAs) provides a suitable platform to implement the BISR technique, because it allows a system to repair itself by changing its architecture in the presence of the radiation induced faults [1]. Although reconfigurability of FPGAs makes them general purpose, field programmable, and fault tolerant devices, it causes some additional problems, such as high fault rate, high power consumption, high area occupation, and low speed [2]. As an example, more than 90% of the SRAM cells in FPGAs are used as configuration memory cells (termed as CRAM) which are mostly placed in routing elements for interconnect and programmable logic blocks for functionality. A transient fault occurrence in a CRAM cell can lead to system failure by changing the circuit functionality or even circuit structure. The main drawback of the existing dynamic approaches is that they are not concurrent fault recovery approaches. During the recovery time, the output value is invalid and the system operation should be paused. In some critical applications such as: the international thermonuclear experimental reactor (ITER), a short pause time in system operation is unacceptable and it can lead to inconvenient events [3]. The hybrid approaches are usually very costly and they are utilized in critical applications; because they are a combination of the attractive features of the both static and dynamic methods. “TMR with spares” is one of the most popular techniques in hybrid redundancy category. This approach combines both ideas of static and dynamic methods. In each period, the referee output is compared with the outputs of every single module. If a discrepancy is detected, the module is considered faulty and is replaced with a reserve module [4].

II. Proposed Method

In this letter a new online fault tolerant approach has been proposed to improve the reliability of electronic circuits in harsh environments. The proposed structure combines the reconfiguration property of the dynamic methods and fault masking feature of the static methods. The first feature reduces the redundant cost and the second one guarantees the continuous operation of the system without any pause time in system operation.

A. Structure

In the proposed structure, the desired logic function is implemented using some homogenous basic cells. In a homogenous structure, the faulty cell can be quickly replaced by a redundant cell. A basic cell structure with more details is shown in Fig. 1. Accordingly, each basic cell includes an original unit, a tester block, and a selector unit.

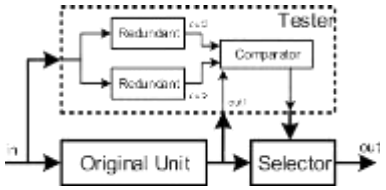

Fig. 1. Proposed basic cell.

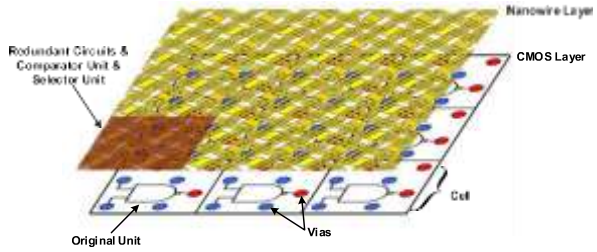

Fig. 2. Our FPNI-based architecture.

1) Original Unit: The original unit is used to implement the desired functionality and it can be made using different technologies. Without losing the generality of the proposed structure, we will describe our proposed structure using a NAND-based circuit.

2) Tester Block: The tester block has a duplicate structure, in which the output of two replica and original unit are compared by comparator. So, the fault occurrence in the original unit can be detected quickly. In other words, the tester output is "1" when the compared values are different and it is "0" in other cases. In this unit we have used two redundant circuits to ensure the correctness of their outputs, and therefore, the transient fault tolerance of the proposed structure is improved.

3) Selector Unit: As shown in Fig. 1, the final output value of each basic cell is defined by the selector unit. It selects either the original unit output or its inverted value for final output. In fault free state, the final output value is defined by the output value of the original unit, and after a fault occurrence in the original unit, the tester block warn the selector unit to set the final output as the inverted value of the original unit output.

B. Implementation

The proposed structure is implemented using a new memristor-based reconfigurable architecture which is based on field-programmable nanowire interconnect (FPNI) [5]. The main advantages of this architecture are high flexibility, high function density, and low fabrication cost. It includes a CMOS layer which is used to implement the logic functions, and a nanowire layer which is used for wired-OR logic and signal routing. Fig. 2 shows more details of our FPNI-based architecture.

Accordingly, the original unit is implemented in CMOS layer to reduce the power consumption and delay, and the other units are implemented in memristor layer to reduce the area overhead. Fig. 3 shows more details about the different parts of the proposed basic cell. Accordingly, the memristor-based parts of the basic cell are implemented using

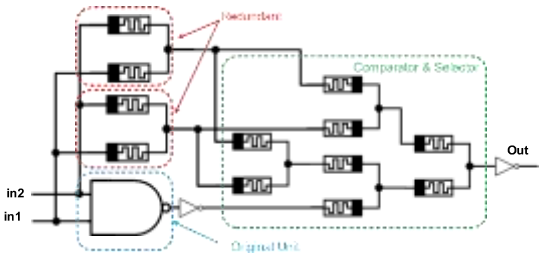

Fig. 3. Different parts of the proposed basic cell.

memristor-ratioed logic (MRL) technique [6]. To overcome the MRL drawback for NOT implementation, we have used AND gates as the redundant circuits [6]; the comparator block and the selector unit are also merged to reduce the resources consumption.

C. Operational Modes

In general, the operating modes of the proposed structure can be divided into three categories: 1) fault free mode; 2) faulty mode; and 3) fault recovery mode.

1) Fault Free Mode: In fault free mode, the output values of the original unit and the redundant circuits are the same, hence the output value of the tester unit is 0 and the final output value is obtained by the original unit.

2) Faulty Mode: After a fault occurrence in the original unit, the faulty mode operation will be started. In this mode, the output values of the original unit and the redundant circuits are different and therefore the output value of the tester block becomes 1.

3) Fault Recovery Mode: After fault detection, the fault recovery phase should be started to replace the faulty cell with a fault free one. Our design seeks to use the tester unit as a temporary alternative circuit when a permanent fault occurs in the original unit. By this technique, the system can continue its normal operation during the fault isolation step without any interruption. The fault recovery phase includes two steps: 1) fault isolation; and 2) redundancy allocation.

In fault isolation step, the wrong data transfer to the final output should be prevented by isolating the faulty original unit. This operation is done by reprogramming the input/output memristors of the faulty cell to be open. A simple circuit includes four NAND gates and its implementation is shown in Fig. 4. To isolate the faulty original unit, the memristors #1, #2, and #3 should be reprogrammed to be open. During this step, the circuit continues its normal operation and the output value of the faulty cell is defined by its tester unit [see Fig. 4(b)].

In redundancy allocation step, the faulty original unit should be replaced by a spare. We use the reconfiguration property of the memristors to replace the faulty unit with a fault free one. For this sake, the input signals should be connected to the spare circuit and the output signal should be switched to the output port of the spare circuit. These connections are configured by reprogramming the input/output memristors of the spare cell to be closed. The redundancy allocation step is shown using a simple circuit in Fig. 5. To replace the faulty

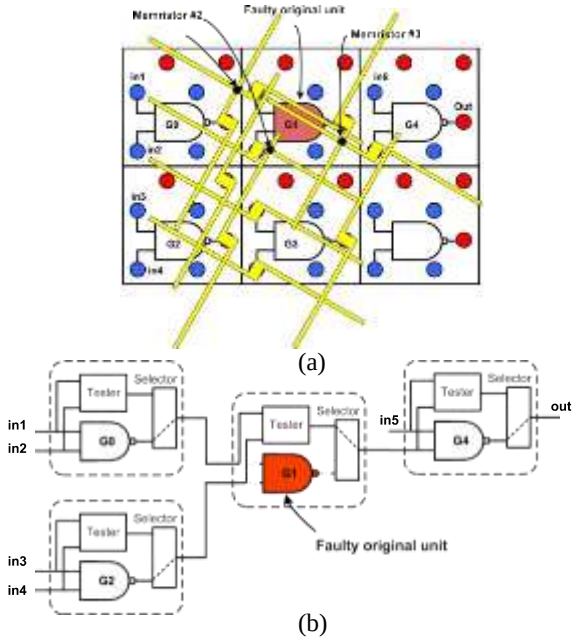

Fig. 4. Fault isolation step. (a) Proposed FPNI-based architecture. (b) Functional circuit.

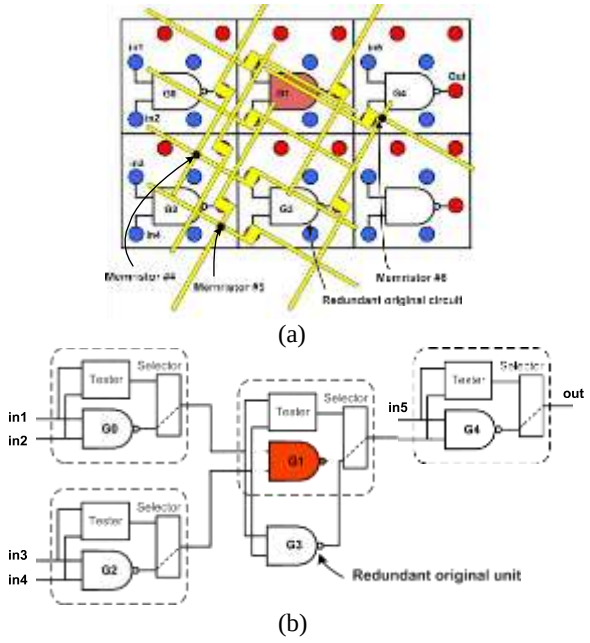

Fig. 5. Redundancy allocation step. (a) Proposed FPNI-based architecture. (b) Functional circuit.

circuit with a spare, the memristors #4, #5, and #6 should be reprogrammed to be closed. During this step, the final output value is also given by the spare circuit.

It should be noted, if the fault effects remain unchanged after the original unit replacement process, this means the original unit is fault free and the other parts (includes: redundant units, tester block, and comparator unit) are faulty, so, the basic cell should be replaced in the same way.

III. Simulation Results

In this section the simulation results of the proposed method have been compared to some static, dynamic, and hybrid fault tolerant approaches. Performance of these methods is evaluated using several criteria, such as area, delay, power consumption, and reliability. In Table I, the numerical results of the proposed structure for C17 circuit, a 4-bit Adder and a 32-channel multiplexer are compared to some static, dynamic, and hybrid approaches, such as TMR, 5MR, FPGA, and duplex-triplex architecture (DTA). In general, the classic *n*-modular redundancy is a fault-tolerant system with *n* processing units and a majority-voter to produce a single output. If $(n-1)/2$ of the processing units fail, the other units correct and mask the fault. For instance, the 5MR system has five processing units and a majority-voter and tolerates two permanent faults. DTA is a hybrid redundancy technique which combines two ideas of "duplication with comparison" and TMR. The TMR technique is used to mask the error, while the duplication with comparison technique is used to detect the errors and therefore omitting the faulty module from voting process dynamically [4].

The occupied area of the given methods is computed using transistor counting technique. According to the presented results in Table I, the occupied area of the proposed method is much less than the required area by the TMR, 5MR, DTA, and the FPGA-based multiplexers. HSPICE and VPR tools are also used to compute the delay and power consumption of the mentioned methods. According to the results, although the delay of the proposed multiplexer is lower than the FPGA-based multiplexer, but the TMR, 5MR, and DTA methods have less delay than the proposed approach. This difference arises from the effects of the capacitance and resistance of the nanowires and the closed-junction resistance on the input/output signals. The power consumption of the proposed multiplexer is also lower than the power consumption of the 5MR, DTA, and the FPGA-based multiplexers. In contrast, the TMR-based multiplexer consumes less power than the proposed multiplexer.

Two different metrics have used in order to evaluate the reliability. In the first metric, we compute the maximum number of permanent faults that the mentioned methods can tolerate proportional to their occupied areas. According to the presented results in Table I, although the TMR, 5MR, DTA, and our proposed methods can tolerate 124, 248, 248, and 124 permanent faults, respectively, our proposed method occupies less area than the other methods. In contrast, the FPGA-based method needs additional space to tolerate the permanent faults. The minimum required area to tolerate a permanent fault in the FPGA-based method is computed according to its configurable logic block (CLB) area. The simulation results show the minimum occupied area of a CLB is equal to 1.39×10^4 μm^2, and the additional space to tolerate 124 permanent faults is equal to 1.73×10^6 μm^2. It should be noted that unlike the static methods, the number of tolerable faults in our proposed method is flexible, and it can tolerate more than 124 permanent faults for some additional spaces. According to simulation results, our proposed method can tolerate a permanent fault per about 1.76×10^1 μm^2 additional spaces. In other words, our proposed method needs 1.09×10^3 μm^2 additional spaces to

TABLE I
NUMERICAL RESULTS OF PROPOSED STRUCTURE AND SOME WELL-KNOWN APPROACHES

		Static		Dynamic	Hybrid	Proposed
		TMR	5MR	FPGA	DTA [4]	Method
C17	Area $(\mu m)^2$	$1.96 * 10^2$	$7.53 * 10^2$	$1.39 * 10^4$	$5.12 * 10^2$	$1.28 * 10^2$
	Delay (s)	$1.47 * 10^{-10}$	$2.97 * 10^{-10}$	$1.35 * 10^{-9}$	$2.68 * 10^{-10}$	$3.00 * 10^{-10}$
	Power (nW)	$3.78 * 10^3$	$1.18 * 10^4$	$4.79 * 10^4$	$9.99 * 10^4$	$1.42 * 10^4$
	Maximum Tolerable Faults	6	12	0	12	6
	SPF	6	9	7	9	41
4-bit Adder	Area $(\mu m)^2$	$1.17 * 10^3$	$4.52 * 10^3$	$2.78 * 10^4$	$3.07 * 10^3$	$7.68 * 10^2$
	Delay (s)	$9.15 * 10^{-10}$	$1.87 * 10^{-9}$	$5.11 * 10^{-9}$	$1.63 * 10^{-9}$	$1.92 * 10^{-9}$
	Power (nW)	$4.95 * 10^4$	$1.62 * 10^5$	$1.40 * 10^5$	$6.94 * 10^5$	$1.20 * 10^5$
	Maximum Tolerable Faults	36	72	0	72	36
	SPF	29	47	38	43	159
MUX 32-Channel	Area $(\mu m)^2$	$4.05 * 10^3$	$1.56 * 10^4$	$9.75 * 10^4$	$1.01 * 10^4$	$2.65 * 10^3$
	Delay (s)	$5.05 * 10^{-10}$	$1.01 * 10^{-9}$	$4.74 * 10^{-9}$	$9.33 * 10^{-10}$	$1.07 * 10^{-9}$
	Power (nW)	$2.04 * 10^5$	$6.36 * 10^5$	$3.88 * 10^5$	$2.68 * 10^6$	$3.42 * 10^5$
	Maximum Tolerable Faults	124	248	0	248	124
	SPF	95	157	131	145	850

tolerate 248 faults which is less than the occupied area of the 5MR and DTA methods.

Silicon protection factor (SPF) [7] is the second metric used to evaluate the reliability which obtains the protection amount of a given fault tolerance technique and it is defined in (1). Accordingly, SPF is the average number of faults needed to cause a failure divided by the area overhead of the fault tolerant approach, and higher SPF value indicates a higher fault tolerance capability

$$\text{SPF} = \frac{\text{Mean No. of faults to failure}}{\text{Fault tolerant design area}/\text{baseline area}}. \quad (1)$$

In addition to less area occupation, our proposed method has some advantages relative to the static techniques, such as TMR. Each TMR block can tolerate only one permanent fault. In other words, the system will be failed if two permanent faults occur in a TMR block; while our proposed multiplexer can tolerate several permanent faults regardless of the fault occurrence location.

IV. CONCLUSION

This letter proposes a novel hybrid fault tolerant strategy that combines the fault masking feature of the static methods with the reconfiguration property of the dynamic methods. These characteristics are employed to lower unnecessary costs and ensure the ongoing functionality of the

system, in turn. The resulting structure has the lowest possible latency, power consumption, and area overhead when it comes to handling many persistent faults during runtime. The findings show that the suggested approach's fault tolerance capability is far higher than that of the specified static, dynamic, and hybrid techniques based on its SPF value.

REFERENCES

[1] V. Dumitri, L. Kirischian, and V. Kirischian, "Run-time recovery mech- anism for transient and permanent hardware faults based on distributed, self-organized dynamic partially reconfigurable systems," *IEEE Trans. Comput.*, vol. 65, no. 9, pp. 2835–2847, Sep. 2016.

[2] U. Farooq, Z. Marrakchi, and H. Mehrez, *Tree-Based Heterogeneous FPGA Architectures*. New York, NY, USA: Springer-Verlag, 2012, p. 188.

[3] P. Leroux *et al.*, "Design of a MGy radiation tolerant resolver-to-digital convertor IC for remotely operated maintenance in harsh environments," *Fusion Eng. Design*, vol. 89, nos. 9–10, pp. 2314–2319, Oct. 2014.

[4] A. Sengupta and S. Bhadauria, "Bacterial foraging driven exploration of multi cycle fault tolerant datapath based on power-performance tradeoff in high level synthesis," *Expert Syst. Appl.*, vol. 42, no. 10, pp. 4719–4732, Jan. 2015.

[5] D. B. Strukov and K. K. Likharev, "CMOL FPGA: A reconfigurable architecture for hybrid digital circuits with two-terminal nanodevices," *Nanotechnology*, vol. 16, no. 6, pp. 888–900, 2005.

[6] S. Kvatinsky *et al.*, "MRL—Memristor Ratioed logic," in *Proc. 13th Int. Workshop Cellular Nanoscale Netw. Appl. (CNNA)*, 2012, pp. 1–6.

[7] K. Constantinides *et al.*, "BulletProof: A defect-tolerant CMP switch architecture," in *Proc. 12th Int. Symp. High Perform. Comput. Archit. (HPCA)*, 2006, pp. 5–16.