International Journal of Electrical Electronics & Computer Science Engineering
Volume 5, Issue 1 (February, 2018) | E-ISSN : 2348-2273 | P-ISSN : 2454-1222
Available Online at www.ijeecse.com

A Novel Design for Memristor Based Logic Gates and Circuits to be Used in FPGA Architectures Technology

Rita Mahajan[1], Basudha[2], Deepak Bagai[3]
[1]Assistant Professor, [2]PG Scholar, [3]Associate Professor, ECE Department, Punjab Engineering College, Chandigarh, India
[1]ritamahajan@pec.ac.in, [2]basudha.dewan@gmail.com, [3]dbagai@yahoo.com

Abstract: A physical memristor was realized a few years back by Hewlett Packard labs. Instantly it became the topic of interest for many scientists and researchers. Memristor has joined resistor, capacitor and inductor in the list of basic two-terminal passive circuit elements. Field Programmable Gate Arrays (FPGAs) have become a major choice of technology for the implementation of various digital circuits because they offer various features such as quick time to market, reprogrammability, small development overhead, large densities up to millions of gates on a single chip and several others. All these features make FPGAs a suitable choice for implementing the logic. In this paper Memristor Ratioed Logic is proposed to implement the logic gates. These logic gates are then used in special hard-wired adder blocks of FPGAs.

Keywords: Configurable Logic Block (CLB), FPGA, Memristor, Memristor Ratioed Logic (MRL)

I. INTRODUCTION

Application Specific Integrated Circuits (ASICs) is an integrated circuit that consists of pre-defined logic cells. ASICs are customized to perform a particular task.

An FPGA is an integrated circuit that is configured by the customer in the field after the manufacturing process is over. Hence it is called "field" programmable.

FPGA contains a matrix of reconfigurable logic blocks, input/output blocks and programmable interconnects. FPGAs can be reprogrammed to carry out the required task or function after manufacturing process is over. Due to their reprogrammable nature, FPGAs find use in several areas like automotive industry, consumer electronics, medical, wired and wireless communication etc.

When configured, FPGAs creates a hardware implementation of a software application. Logic is processed in FPGA using a dedicated hardware.

Field Programmable Gate Arrays (FPGAs) are becoming a major choice of technology for the implementation of various digital circuits because of the several advantages and features offered by them.

FPGAs have smaller design cycle because their design cycle doesn't include complex and time consuming phases such as floor planning, placement and routing. Therefore they have faster time to market. On the other hand a large ASIC may take a year or more to get designed. Also FPGAs are programmable in field i.e. they can be programmed according to the customer requirement after manufacturing. Therefore FPGAs can be reused. But ASICs are customized to perform a single task i.e. they have limited area of application. Also FPGA operations are parallel in nature. Therefore different processes which are running simultaneously need not compete for resources i.e. there is no problem of resource sharing. The configurability, programmability and versatility of FPGAs make them more popular in comparison to ASICs [1].

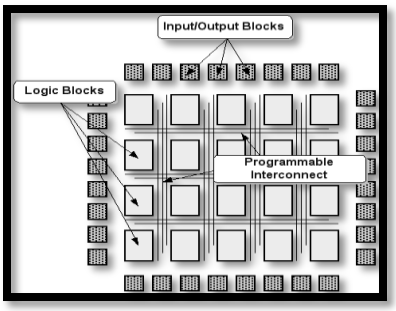

Fig. 1. Conventional FPGA Architecture

Fig. 1 shows the conventional FPGA architecture. It comprises of Configurable Logic Blocks (CLBs), I/O blocks and programmable interconnects.

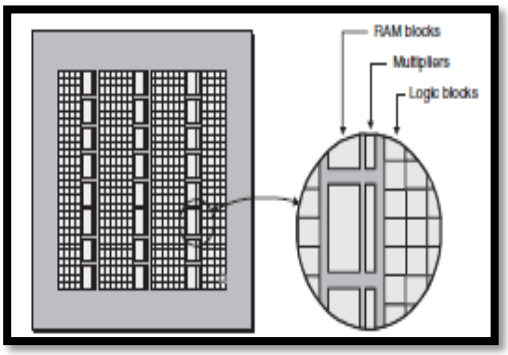

Fig. 2. FPGA Chip with Embedded Multipliers and RAM Blocks

Functions such as multipliers and adders are

International Journal of Electrical Electronics & Computer Science Engineering
Volume 5, Issue 1 (February, 2018) | E-ISSN : 2348-2273 | P-ISSN : 2454-1222
Available Online at www.ijeecse.com

fundamentally time consuming when they are executed by connecting a lot of programmable logic blocks together. Since a lot of applications require these functions, therefore many FPGAs include special hard-wired multiplier and adder blocks. Life becomes a bit easier if the FPGA provide embedded adder and multiplier blocks. These blocks are generally placed close to embedded RAM blocks as shown in Fig. 2.

In this paper a structured design methodology is proposed for logic computation by making use of memristors. A design approach named memristor ratioed logic is discussed in this paper. This approach is useful in designing logic gates which could be further used to design other circuits.

The paper is organized as follow: Section II provides an overview of the memristor and its working. Section III discusses memristor ratioed logic (MRL) and its use in designing various logic gates. Adder circuits using MRL are explained in section IV. In section V proposed adder circuit is presented. The paper concludes in section VI.

II. MEMRISTOR

Leon Chua, in 1971 proposed memristor, on basis of symmetry forefront, as the fourth basic fundamental circuit element apart from resistor, capacitor and inductor [2]. Memristor became the topic of interest for many scientists and researchers due to its simple structure, non-volatile nature, high density, low power and enhanced scalability. Memristor is a non-linear passive device with two terminals whose memristance changes according to the direction and amount of current flowing through it. Say if the current is flowing in clockwise direction then memristance will decrease but if the current starts flowing in anti-clockwise direction then the memristance will increase. Memristor remembers its state when power is turned off [3]. Therefore memristors are non-volatile in nature. So the static power dissipation in case of memristor is negligible.

Memristor provides a relationship between magnetic flux and electric charge by following equation:

$$M(q) = \frac{d\varphi}{dq} \quad (1)$$

where φ is the magnetic flux and q is the electric charge.

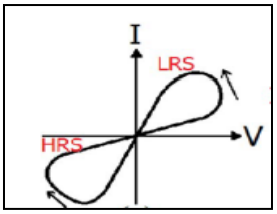

Fig. 3. Memristive Hysteresis Curve

Chua stated that there exists a hysteresis curve when current is plotted against voltage for memristor [4]. It is shown in Fig. 3.

In 2008, a physical memristor was realized by Hewlett Packard labs. A thin layer of titanium dioxide (TiO_2) was sandwiched between two platinum electrodes [5].

Different concentration of oxygen ions is used for doping the bulk layer so that its resistivity can be varied. The region which has higher doping concentration offers lower resistivity and the region which has lower doping concentration offers higher resistivity. Fig. 4 shows the structure of a physical memristor.

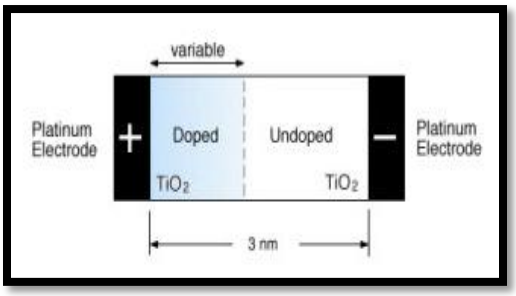

Fig. 4. Physical Memristor

Titanium Dioxide (TiO_2) is used as the resistive material in the bulk layer between these electrodes [6]. In the beginning, oxygen vacancies are asymmetrically distributed in the bulk layer. Due to this asymmetric distribution bulk layer has a very high resistance i.e. memristor is in its high resistance state (HRS). When an appropriate voltage is applied across the electrodes, these oxygen vacancies will get displaced. They will line-up and therefore form a conducting channel between two platinum electrodes.

Therefore the resistance of bulk layer will reduce i.e. the memristor will switch to its low resistance state (LRS). Memristor operates between these two states:

High resistance state and Low resistance state, which is shown in Fig. 5.

.

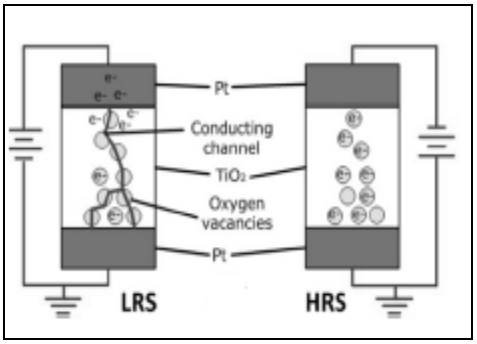

Fig. 5. Switching Process of Memristors Using Opposite Polarities of Voltage.

International Journal of Electrical Electronics & Computer Science Engineering
Volume 5, Issue 1 (February, 2018) | E-ISSN : 2348-2273 | P-ISSN : 2454-1222
Available Online at www.ijeecse.com

III. MEMRISTOR RATIOED LOGIC

Memristor Ratioed Logic (MRL) is the approach that is used for designing logic gates. This approach integrates memristive devices with conventional CMOS technology as it is compatible with the CMOS technology [7]. It needs very few memristors and computational steps to process the logic. But to implement complementary logic like NAND and NOR using this approach, memristor alone is not sufficient. A CMOS inverter is required in addition to memristors to get the logic implemented. The reason for using memristors for evaluating logic is that it decreases physical area and increases logic density [8].

A. AND Gate:

The MRL based AND Gate makes use of two memristors P and Q as shown in Fig. 6. These two memristors are connected to two positive voltages V_P and V_Q. Output is (taken as the logic value of memristor R [2]. We assume that the initial memristance of M_R is Roff, where Roff correspond to logic zero. Also we set $R_R >> R_P, R_Q$.

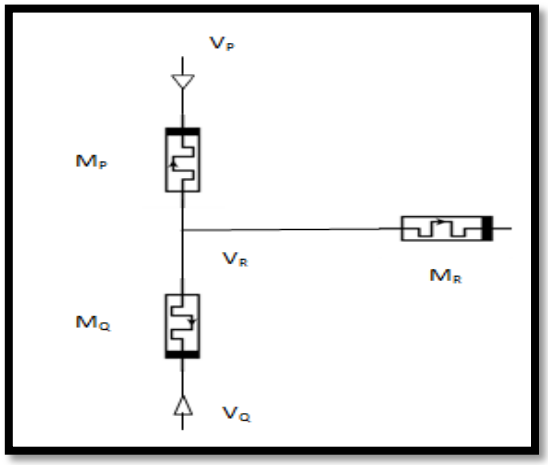

Fig. 6. Design of AND Gate Using MRL Logic

Applying Kirchhoff's law we get,

$$\frac{V_P - V_R}{R_P} + \frac{V_Q - V_R}{R_Q} = \frac{V_R}{R_R} \quad (2)$$

$$V_R\left[\frac{1}{R_R} + \frac{1}{R_P} + \frac{1}{R_Q}\right] = \frac{V_P}{R_P} + \frac{V_Q}{R_Q} \quad (3)$$

$$V_R = V_P\frac{R_S}{R_P} + V_Q\frac{R_S}{R_Q} \quad (4)$$

$$\text{Where } R_s = \left[\frac{1}{R_R} + \frac{1}{R_P} + \frac{1}{R_Q}\right]^{-1} \quad (5)$$

We know $R_R >> R_P, R_Q$

$$\text{Therefore } R_s = \left[\frac{1}{R_P} + \frac{1}{R_Q}\right]^{-1} \quad (6)$$

From (3) and (5) we get,

$$V_R = V_p\frac{R_Q}{R_P + R_Q} + V_Q\frac{R_P}{R_P + R_Q} \quad (7)$$

Four cases of AND Gate can be analyzed as follows:

1) $V_P = V_Q = 1$

From (7) we get,

$$V_R = V_p\left[\frac{R_P + R_Q}{R_P + R_Q}\right] = V_p = 1$$

This positive voltage across M_R will reduce its memristance to R_{ON} after some time. As $R_R = R_{ON}$, therefore logic value stored in M_R is 1.

2) $V_P = V_Q = 0$

From (7) we get,

$$V_R = 0$$

As the output voltage is zero, therefore M_R will retain its value as $R_R = R_{OFF}$ i.e. logic 0.

3) $V_P = 1\,,\ V_Q = 0$

Voltage across M_P is negative which makes $R_P = R_{OFF}$ and voltage across M_Q is positive which makes $R_Q = R_{ON}$.

As $R_{OFF} \gg R_{ON}$, therefore from (7) we get,

$$V_R = V_p\frac{R_Q}{R_P + R_Q} = V_p\frac{R_{ON}}{R_{OFF} + R_{ON}} \approx 0$$

As the output voltage is zero, therefore M_R will retain its value as $R_R = R_{OFF}$ i.e. logic 0.

4) $V_P = 0\,,\ V_Q = 1$

Voltage across M_P is positive which makes $R_P = R_{ON}$ and voltage across M_Q is negative which makes $R_Q = R_{OFF}$.

As $R_{OFF} \gg R_{ON}$, therefore from (7) we get,

$$V_R = V_Q\frac{R_P}{R_P + R_Q} = V_Q\frac{R_{ON}}{R_{OFF} + R_{ON}} \approx 0$$

As the output voltage is zero, therefore M_R will retain its value as $R_R = R_{OFF}$ i.e. logic 0.

Therefore all the four cases of AND gate are verified.

B. OR Gate:

OR gate can be implemented in a similar way as AND gate. The only difference in their implementation is that memristors M_P and M_Q are connected with opposite polarities as shown in Fig. 7.

International Journal of Electrical Electronics & Computer Science Engineering
Volume 5, Issue 1 (February, 2018) | E-ISSN : 2348-2273 | P-ISSN : 2454-1222
Available Online at www.ijeecse.com

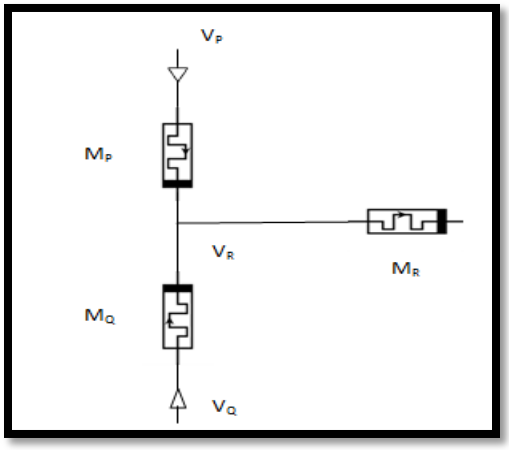

Fig. 7. Design of OR Gate Using MRL Logic

Four cases of OR Gate can be analyzed as follows:

1) $V_P = V_Q = 1$

From (7) we get,

$$V_R = V_p\left[\frac{R_P+R_Q}{R_P+R_Q}\right] = V_p = 1$$

This positive voltage across M_R will reduce its memristance to R_{ON} after some time. As $R_R = R_{ON}$, therefore logic value stored in M_R is 1.

2) $V_P = V_Q = 0$

From (7) we get,

$V_R = 0$

As the output voltage is zero, therefore M_R will retain its value as $R_R = R_{OFF}$ i.e. logic 0.

3) $V_P = 1\,,\ V_Q = 0$

Voltage across M_P is positive which makes $R_P = R_{ON}$ and voltage across M_Q is negative which makes $R_Q = R_{OFF}$.

As $R_{OFF} \gg R_{ON}$, therefore from (7) we get,

$$V_R = V_p\frac{R_Q}{R_P+R_Q} = V_p\frac{R_{OFF}}{R_{OFF}+R_{ON}} \approx 1$$

This positive voltage across M_R will reduce its memristance to R_{ON} after some time. As $R_R = R_{ON}$, therefore logic value stored in M_R is 1.

4) $V_P = 0\,,\ V_Q = 1$

Voltage across M_P is positive which makes $R_P = R_{ON}$ and voltage across M_Q is negative which makes $R_Q = R_{OFF}$.

As $R_{OFF} \gg R_{ON}$, therefore from (7) we get,

$$V_R = V_Q\frac{R_P}{R_P+R_Q} = V_p\frac{R_{OFF}}{R_{OFF}+R_{ON}} \approx 1$$

This positive voltage across M_R will reduce its memristance to R_{ON} after some time. As $R_R = R_{ON}$, therefore logic value stored in M_R is 1.

Therefore all the four cases of OR gate are verified.

C. NOT Gate:

Only one memristor is required to implement a not gate using MRL logic as shown in Fig. 8. This memristor is connected to a voltage. We assume that the initial memristance of M_R is R_{ON}, where R_{ON} correspond to logic one.

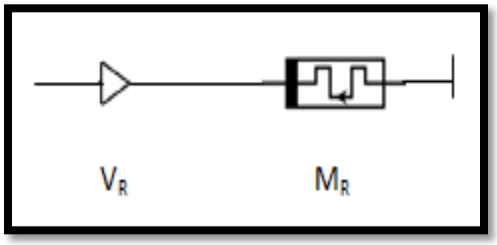

Fig. 8. Design of NOT Gate Using MRL Logic

Two cases of NOT Gate can be analyzed as follows:

1) $V_R = 0$

Since the applied voltage is zero, therefore M_R will retain its value as $R_R = R_{ON}$ i.e. logic 1.

2) $V_R = 1$

Since the applied voltage is one, therefore memristance M_R changes to $R_R = R_{OFF}$ after some time. Therefore M_R will store logic 0.

Therefore both the cases of NOT gate are verified.

D. NAND Gate:

To obtain NAND gate CMOS inverter is added to the output of AND gate. At point P in the circuit shown in Fig. 9, we get P=A.B which is then fed to the CMOS inverter. Therefore the final output of the circuit is $Y = \overline{A.B}$.

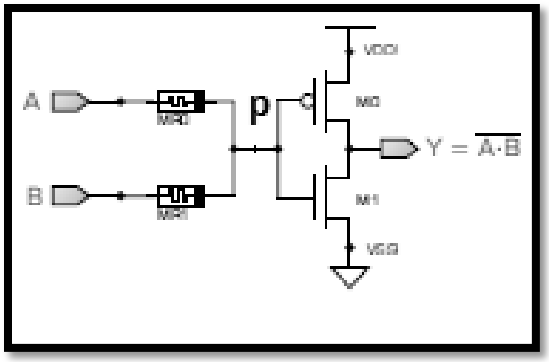

Fig. 9. Design of NAND Gate Using MRL Logic

International Journal of Electrical Electronics & Computer Science Engineering
Volume 5, Issue 1 (February, 2018) | E-ISSN : 2348-2273 | P-ISSN : 2454-1222
Available Online at www.ijeecse.com

E. NOR Gate:

Similarly we can design NOR gate by just reversing the polarity of memristors in figure 9.

F. XOR Gate:

To design a XOR gate two CMOS inverters are used to complement both the inputs. Two AND operations are performed using 4 memristors and one OR operation is performed using 2 memristors. Therefore to implement XOR gate using MRL logic, two CMOS inverters and six memristors are required. Design of XOR gate is shown in Fig. 10.

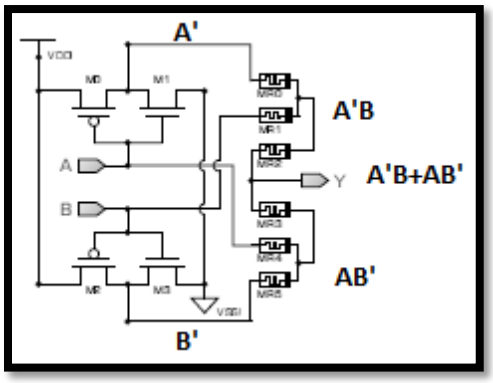

Fig. 10. Design of XOR Gate Using MRL Logic

G. XNOR Gate:

Similarly we can design XNOR gate by just adding a CMOS inverter to the output of Fig. 10.

IV. ADDER CIRCUITS USING MRL LOGIC

Half Adder Circuit:

A half adder circuit adds two numbers and produces two outputs, sum(S) and carry(C). If A and B are the two inputs to half adder circuit then sum is (A XOR B) and carry is (A AND B).

.

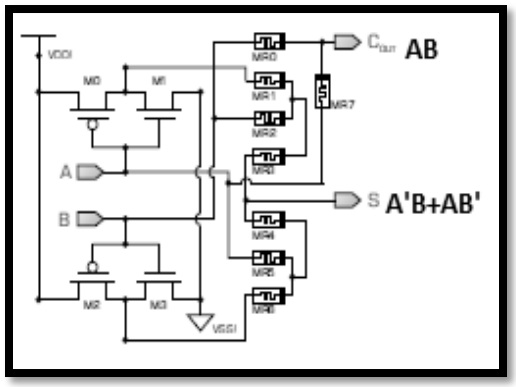

Fig. 11. Design of Half Adder Circuit Using MRL Logic

The circuit shown in Fig. 11 implements a half adder. Sum which is A XOR B is obtained from the XOR circuit discussed in Fig. 10. To obtain carry which is A AND B, two memristors, MR0 and MR7 are added which perform AND operation between inputs A and B. Therefore, we get,

$S = A\ XOR\ B$

$C = A\ AND\ B$

This design of half adder makes use of eight memristors and four MOSFETs to get the logic implemented

Full Adder Circuit:

A full adder circuit adds three numbers and produces two outputs, sum(S) and carry(C). If A, B and Cin are the three inputs to a full adder circuit then sum is (A XOR B XOR Cin) and carry is (A XOR B).C+(A.B). The circuit shown in Fig. 12 implements a full adder.

Two half adder circuits are cascaded along with two additional memristors MR10 and MR11 to get a full adder implemented.

The output of first half adder which is A XOR B is fed to second half adder along with second input Cin. Therefore the sum output produced by second half adder is A XOR B XOR Cin.

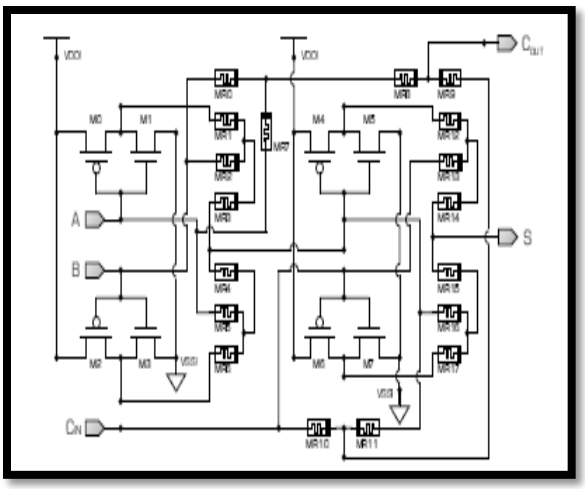

Fig. 12. Design of Full Adder Circuit using MRL logic

To get Cout two additional memristors MR10 and MR11 are used in the circuit. These memristors act as OR gate. They add the carry outputs of both the half adders. Carry output of first half adder is AB and Carry output of second half adder is (A XOR B)Cin. Therefore final carry output is (A XOR B)Cin+ AB.

This design of full adder makes use of eighteen memristors and eight MOSFETs to get the logic implemented.

International Journal of Electrical Electronics & Computer Science Engineering
Volume 5, Issue 1 (February, 2018) | E-ISSN : 2348-2273 | P-ISSN : 2454-1222
Available Online at www.ijeecse.com

V. PROPOSED ADDER CIRCUIT

In order to reduce the area utilization and delay in the circuit, we need to optimize the circuit by incorporating minimum number of components. Half adder discussed in section IV makes use of eight memristors and four MOSFETs to get the logic implemented. While the full adder discussed makes use of eighteen memristors and eight MOSFETs to get the logic implemented.

The proposed circuit provides different functionalities at the same time i.e. AND, OR and XOR. It makes use of four memristors and two MOSFETs. Memristors M1 and M2 form an AND gate, memristors M3 and M4 form an OR gate. Memristors M3 and M4 are placed in source of PMOS transistor. The functionality of the circuit is as follows:

A. a=b=1

X1= a AND b = 1

X2= a OR b = 1

As X1=1, therefore NMOS is ON and PMOS is OFF.

X3 is connected to ground.

⇒X3=0.

B. a=b=0

X1= a AND b =0

X2= a OR b = 0

As X1=0, therefore NMOS is OFF and PMOS is ON.

X3 is connected to X2.

⇒X3=0.

C. a=0, b=1

X1= a AND b =0

X2= a OR b = 1

As X1=0, therefore NMOS is OFF and PMOS is ON.

X3 is connected to X2.

⇒X3=1.

D. a=1, b=0

X1= a AND b =0

X2= a OR b = 1

As X1=0, therefore NMOS is OFF and PMOS is ON.

X3 is connected to X2.

⇒X3=1.

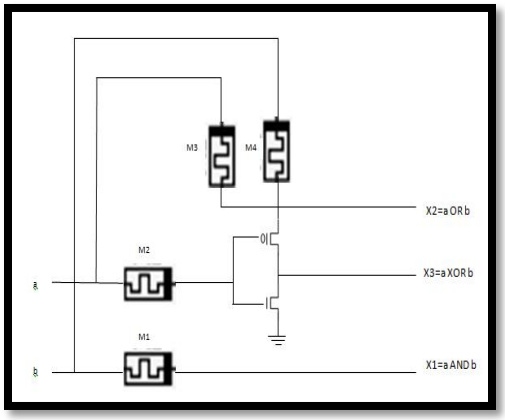

Fig. 13. Proposed Circuit To Implement Half Adder With Fewer Components

Since the circuit proposed in Fig. 13 generates both AND and XOR outputs, therefore it can be used as a half adder circuit.

The proposed half adder circuit requires only four memristors and two MOSFETs compared to eight memristors and four MOSFETs in the half adder circuit discussed in section V. So the number of components reduces to half in the proposed architecture.

Full adder circuit can also be designed by cascading two half adders and one OR gate. Implementation of full adder circuit is shown in Fig. 14.

The proposed full adder circuit requires only ten memristors and four MOSFETs compared to eighteen memristors and eight MOSFETs in the full adder circuit discussed in section V. So the number of components reduces in the proposed architecture which would be beneficial in terms of area utilization and delay.

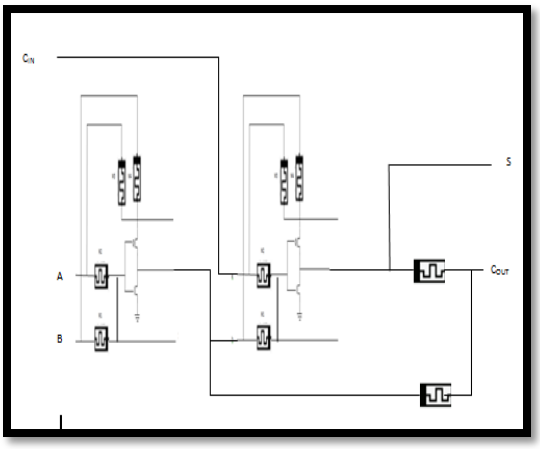

Fig. 14. Proposed Circuit To Implement Full Adder With Fewer Components

Table I compares the number of memristors and

MOSFETs that are required in memristor ratioed logic and the proposed logic to implement a half adder and full adder circuit.

Table I. Comparison Of Components In MRL Logic and Proposed Logic

	Memristors		MOSFETs	
Circuit	MRL Logic	Proposed Logic	MRL Logic	Proposed Logic
Half Adder	8	4	4	2
Full Adder	18	10	8	4

VI. CONCLUSION

In this paper, memristors have been used to implement basic logic gates. A new approach, memristor ratioed logic, is used to deign logic gates. This approach is compatible with the conventional CMOS technology and it needs fewer memristors and computational steps to process the logic. These logic gates are then used to design half adder and full adder circuits. But the designed circuits make use of a lot of memristors and MOSFETs. In this paper optimized circuits for half and full adder are proposed which utilizes fewer memristors and MOSFETs as compared to previously discussed circuits. Reduction of components in the proposed architecture would be beneficial as it will decrease the area utilization and delay of the circuit.

VII. REFERENCES

[1] Ho, Patrick WC, Haider Abbas F. Almurib, T. Nandha Kumar, "Configurable memristive logic block for memristive-based FPGA architectures", Integration, the VLSI Journal, Vol. 56, pp. 61-69, Sept. 2016.

[2] Zhang, Y., Shen, Y., Wang, X. and Cao, L., "A novel design for memristor-based logic switch and crossbar circuit", IEEE Transactions on Circuits and Systems I: Regular Papers, Vol. 62, no. 5, pp.1402-1411, 2015.

[3] Teimoory, M., Amirsoleimani, A., Ahmadi, A. and Ahmadi, M.,A Hybrid Memristor-CMOS Multiplier Design Based on Memristive Universal Logic Gates.

[4] Chua, L., "Memristor-the missing circuit element", IEEE Transactions on circuit theory, Vol. 18, no. 5, pp.507-519. 1971.

[5] Papandroulidakis, G., Vourkas, I., Vasileiadis, N. and Sirakoulis, G.C.,"Boolean logic operations and computing circuits based on memristors", IEEE Transactions on Circuits and Systems II: Express Briefs, Vol. 61, no. 12, pp.972-976, 2014.

[6] Wald, N. and Kvatinsky, S., "Design methodology for stateful memristive logic gates", In Science of Electrical Engineering (ICSEE), IEEE International Conference, pp. 1-5, Nov. 2016.

[7] Kvatinsky, S., Wald, N., Satat, G., Kolodny, A., Weiser, U.C. and Friedman, E.G, "MRL—Memristor ratioed logic", In Cellular Nanoscale Networks and Their Applications (CNNA), 2012 13th International Workshop, pp. 1-6, Aug. 2012.

[8] Emara, A.S., Madian, A.H., Amer, H.H., Amer, S.H. and Abdelhalim, M.B., "Testing of memristor ratioed logic (MRL) XOR gate", In Microelectronics (ICM), 2016 28th International Conference , pp. 181-184, Dec. 2016.