Design of 64 Bit Arithmetic Logic Unit Using Quantum Dot Cellular Automata

Siji N M[1] Rachana M K[2]
[1]Student [2]Assistant Professor
[1,2]Department of Electronics and Communication Engineering
[1,2]IES College of Engineering, Thrissur, India

Abstract— Quantum-dot Cellular Automata (QCA) is a revolutionary nano-scale technology that, by significantly improving electrical circuit design, can be regarded a viable alternative to CMOS technology. The Arithmetic Logic Unit (ALU) is a part of the Central Processing Unit (CPU) that performs arithmetic and logical operations. A novel low-complexity QCA 4:1 and QCA 8:1 multiplexer, which is application-specific to the proposed ALU, is proposed. The 12 operations in the One Bit Arithmetic Logic Unit (ALU) are designed in both arithmetic and logic. This system has been proposed and simulated designs for 4 Bit,8 Bit, 16 Bit, 32 Bit & 64 Bit with 15 operations. This is designed and simulated in Xilinx vivaldo 14.2 ISE Design Tool using verilog code. The proposed structures outperform counterpart approaches in terms of delay and power consumption, according to simulation of data.

Keywords: QCA, Multiplexer, Adder, ALU

I. INTRODUCTION

CMOS technology is used in the design of today's Very Large Scale Integration (VLSI) circuits, but several of the technology's limitations, such as physical, material, power-thermal, technological, and economic concerns, have led to the development of QCA as a new technology to address these issues. This technology is unique because of its patented requirements, which include incredibly small feature sizes at the molecular level.

A quantum cell is made up of four charge containers or dots that are arranged in the corner of a square. Two extra mobile electrons exist in the cell, which quantum can mechanically tunnel between dots but not between cells. Columbic repulsion forces the electrons into corner locations. In the QCA cell for an evacuated cell, there are two fiercely irrelevant approaches of two electrons, Cell polarisation P=+1 and P=-1 are expected respectively. Cell polarisation P=+1 alludes to parallel 1 and cell polarisation P=-1 alludes to relating 0.

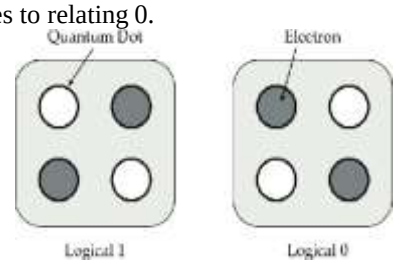

Fig. 1.1: QCA Cell

The essential part of this technology is a QCA cell, which consists of a nano-scale square with four charged containers in its corners, known as quantum dots, and two mobile electrons. A single QCA cell has the ability to generate. In terms of electrical current, there is none in QCA computations, and the power consumption is far lower than in standard CMOS circuits. The most frequent system used in QCA technology is the four-zone plan, also known as Landauer clocking, which contains the Relax, Switch, Hold, and Release phases. Furthermore, there are a variety of techniques to cross the wires on each other, including single-layer and double-layer.

The universal gate of QCA Logic is the majority gate. The polarity of the centre cell is determined by the three input cells of the Majority Voter, and the output received is the Majority Voted Polarity. By setting the input polarity of one input to +1 or -1, it can be utilised as an AND or OR gate.

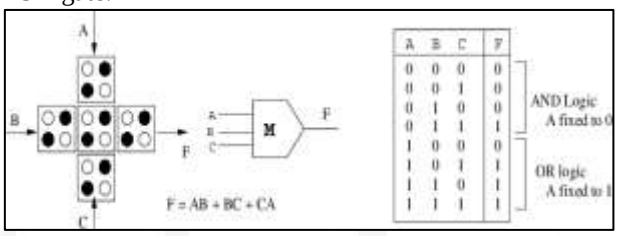

Fig. 1.2: Majority Gate

The tunnelling barriers of the cells above a QCA clock go through four stages. The tunnelling walls begin to rise in the first stage. When the tunnelling barriers are high enough to prohibit electrons from tunnelling, the second stage is reached. When the high barrier begins to lower, the third stage begins. Finally, in the fourth stage, the tunnelling obstacles are removed, allowing electrons to tunnel freely once more. In simple terms, electrons are free to tunnel when the clock signal is strong. The cell becomes latched when the clock signal is low. Data would naturally flow from left to right if a horizontal wire had 8 cells and each subsequent pair, starting from the left, was linked to each consecutive clock. The latching of the first pair of cells will continue until the latching of the second pair of cells, and so on. Data flow direction can be controlled in this fashion using clock zones.

Based on the explicit interactions between QCA cells, a new application-specific QCA 4:1 multiplexer is developed. Furthermore, a new QCA full adder is presented to increase the associated operations based on the extracted specific features from the arithmetic operations of ALU, as well as the fact that most arithmetic operations of ALU can be carried out through the sum operation. In addition, based on the suggested complete adder and the application-specific QCA 4:1 multiplexer, a novel multilayer 1-bit ALU is designed to perform 12 various operations, including four logical and eight arithmetic operations. Furthermore, the proposed structures' structural and energy efficiency are assessed in comparison to competing designs. QCA 2:1 multiplexer, QCA 4:1 multiplexer, QCA 8:1 multiplexer, and complete adder are among the proposed structures. The proposed QCA ALU structure is also included. It examines

All rights reserved by www.ijsrd.com

the proposed structures' complexity and compares them to their counterparts.

II. RELATED WORK

"High-performance multiplexer architecture for quantum-dot cellular automata" [1] was described by Hamid Rashidi, Abdalhossein Rezai, and Sheema Soltany. They demonstrated and compared three multiplexer architectures in the QCA technology: a new and efficient 2:1 multiplexer architecture, a 4:1 multiplexer architecture, and an 8:1 multiplexer architecture. Based on the 2:1 QCA multiplexer, the 4:1 and 8:1 QCA multiplexer architectures are presented. The coplanar crossover technique is used to implement the proposed designs. The QCADesigner tool version 2.0.1

"Efficient Design of Full Adder and Subtractor utilising 5-input Majority Gate in QCA," [2] explained Ramanand Jaiswal, Trailokya Nath Sasamal. With the requirement of a smaller cell area, fewer cell counts, and a shorter clock cycle delay, an efficient full adder and subtractor circuit was constructed in "QCA Designer 2.0.3" using a 5-input majority gate. QCAPro is also used to analyse the energy dissipation of a one-bit full adder circuit. Using a 5-input majority gate, a full adder and subtractor circuit is proposed. The new full adder and subtractor lowered the number of cells, occupied area, and energy dissipation requirements. The QCAPro tool is used to calculate energy dissipation.

Several QCA decimal full adders are trivially built by the aforementioned direct mapping, according to Dariush Abedi and Ghassem Jaberipur's paper [3] "Decimal Full Adders Specially Designed for Quantum-Dot Cellular Automata.". Only one proposal in the literature, however, attempts to make better use of majority gates while also replacing many AND and ORgates with PUMs. In this paper, they presented a QCA decimal full adder that is mostly made up of completely used majority gates (i.e., no constant inputs) and only uses PUMs on rare occasions. We propose a QCA decimal full adder in this study that is largely made up of fully used majority gates (i.e., no constant inputs) and rarely uses PUMs. The suggested circuit was constructed and tested using QCADesigner, and the results were compared to relevant earlier studies, showing improvements in cell count, area, and delay of 39 percent, 78 percent, and 12 percent, respectively. This is predicted to result in fewer overall majority gates, as in the designof binary complete adders with only three majority gates.

III. 64 BIT QCA ALU

The Arithmetic and logical unit (ALU) is a critical component of the Central Processing Unit (CPU). A digital circuit that performs arithmetic and logical operations is known as an ALU. QCA technology procedures can be performed using the majority gate. 15 OPERATIONS are done in ALU. There are 8 operations in the logical unit and 7 operations in the arithmetic unit. An 8:1 multiplexer is used in a logical unit with a majority gate. A 4:1 Multiplexer and RCA Full Adder are used in the Arithmetic Unit with Majority Gate. Majority gates are utilised with all multiplexers and adder. Three control signals determine the operation of the ALU. M is the mode control variable used to select between arithmetic and logical operations. S2,S1 and S0 are used in combination with M to select between the eight Logical operations of ALU. Cin, S1 and S0 are used in combination with M to select between the eight arithmetic operations of ALU. All the combinational circuits are implemented with the majority gates.

A. Design of Arithmetic Unit

The arithmetic operations have distinct characteristics, therefore the sum operation is constant throughout all eight arithmetic functions. As a result, the suggested full adder can easily perform all arithmetic operations. Furthermore, equivalent operations have an addition with '0' in four circumstances when Cin is equal to '0' (Cin = 0), and corresponding operations have a sum with'1' in other cases when Cin is equal to '1' (Cin =1). Furthermore, in all eight arithmetic operations, operand 'A' and addition with input carry (Cin) are fixed, allowing them to be coupled to the whole addition. By removing the operands 'A', 'addition with '0', and 'addition with '1' from Table I, remaining values such as '0', '1', 'B', and 'B' should be entered to the full adder in the various situations. Full Adder (FA) is one of the most frequently utilized components in the arithmetic operations. This block in addition to the sum operation can be applied in other arithmetic operations such as multiplication, division, and subtraction.

$$Z_2 = [(\bar{S_1}\bar{S_0} \times 0) + (\bar{S_1}S_0 \times 1) + (S_1\bar{S_0} \times B) + (S_1S_0 \times \bar{B})]$$
$$= [(\bar{S_1}S_0) + (S_1\bar{S_0}B) + (S_1S_0\bar{B})]$$

S1	S0	Out
0	0	0
0	1	1
1	0	B
1	1	B'

Fig. 2.1: Truth Table of Arithmetic Unit

B. Design of Logic Unit

The logical unit must able to carry out the XOR, AND, OR, and NOT operations. This unit is designed based on the cell interaction. It generates $Z1$ as output, which $Z1$ can be generated.

$$Z_1 = \bar{S_1}\bar{S_0}\,(A \oplus B) + \bar{S_1}S_0\,(AB) + S_1\bar{S_0}\,(A + B) + S_1S_0\,((\sim B))$$

$$Z1 = S_1S_0\bar{A} + \bar{S_1}\bar{S_0}AB + \bar{S_1}\bar{S_0}A + \bar{S_1}A(S_0 \oplus B)$$

C. Design of 2:1 Multiplexer

This simple 2-1 line multiplexer circuit, made up of ordinary NAND gates, has an input A that controls which input (I0 or I1) is sent to the output at Q. When the data select input, A, is LOW at logic 0, input I1 transfers its data through the NAND gate multiplexer circuit to the output, while input I0 is blocked, as shown in the truth table above. When data choose A is HIGH at logic 1, the situation is reversed, and input I0 now delivers data to output Q while input I1 is blocked.

So, by applying a logic "0" or "1" at A, we may select the proper input, I0 or I1, with the circuit behaving similarly to a single pole double throw (SPDT) switch. It can be only switched 2 inputs since we only have one control line

All rights reserved by *www.ijsrd.com*

(A), therefore in this simple example, the 2 -input multiplexer connects one of two 1-bit sources to a shared output, resulting in a 2-to-1-line multiplexer. The following Boolean expression demonstrates this. Utilizing the same technique, we may expand the number of data inputs that can be selected even further, and larger multiplexer circuits can be built using smaller 2-to-1 multiplexers as their basic building blocks.

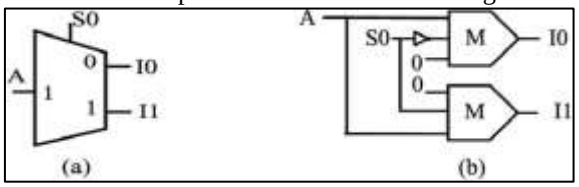

Fig. 2.2: Logic Diagram of QCA 2:1 Multiplexer

Truth Table

x	y	f	f→y
0	0	0	f = y
0	1	1	
1	0	1	f = 1
1	1	1	

Fig. 2.3: Truth Table of 2:1 Multiplexer

D. Design of 4:1 Multiplexer

With inputs A through D and data select lines a, b, the Boolean expression for this 4-to-1 Multiplexer is Q = abA + abB + abC + abD. Only ONE of the four analogue switches is closed at anygiven time in this example, connecting only one of the input lines A to D to the single output at Q. The addressing input code on lines "a" and "b" determines which switch is closed. To choose input Bto the output at Q in this example, the binary input address would need to be "a" = logic "1" and "b" = logic "0." As a result, we may plot the data selection through the multiplexer as a function of the data choose bits. By adding more control address lines (n), the multiplexer will be able to manage more inputs because it can switch 2n inputs, but each control line configuration will only connect ONE input to the output. Then, in order to implement the Boolean expression above using individual logic gates, seven separate gates consisting of AND, OR, and NOT gates would be required, as illustrated.

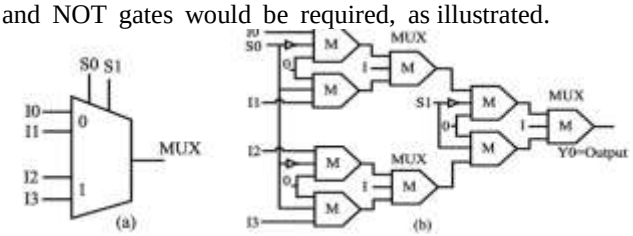

Fig. 2.4: Logic Diagram of QCA 4:1 Multiplexer

Truth Table

S0	S1	Y
0	0	I0
0	1	I1
1	0	I2
1	1	I3

So, final equation,

Y = S0′.S1′.I0 + S0′.S1.I1 + S0.S1′.I2 + S0.S1.I3

Fig. 2.5: Truth Table of QCA 8:1 Multiplexer

E. Design of 8:1 Multiplexer

The data line is connected to the output line for the combination of a selection input. An 8*1 multiplexer is depicted in the circuit below. Eight AND gates, one OR gate, and three selection lines are required for the 8-to-1 multiplexer. The AND gate receives a combination of selection inputs as an input, together with the accompanying input data lines. All of the AND gates are connected in a similar way. In this 8*1 multiplexer, one AND gate produces a value of 1 for every selection line input, while the remaining AND gates all give 0. Finally, all the AND gates are added using OR gates, and the result is equal to the desired value.

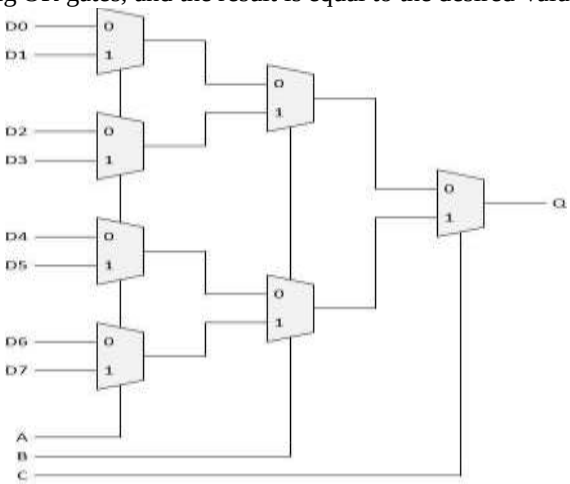

Fig. 2.6: Logic Diagram of 8:1 multiplexer

Select Data Inputs			Output
S2	S1	S0	Y
0	0	0	D0
0	0	1	D1
0	1	0	D2
0	1	1	D3
1	0	0	D4
1	0	1	D5
1	1	0	D6
1	1	1	D7

Fig. 2.7: TruthTable of 8:1 Multiplexer

F. Design of Full Adder

A full adder circuit is central to most digital circuits that perform addition or subtraction.It is so called because it adds together two binary digits, *plus* a carry-in digit to produce a sum and carry-out digit. It therefore has three inputs and two outputs.

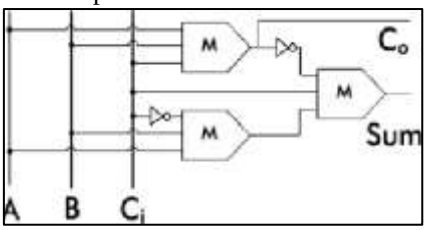

Fig. 2.8: Logic Diagram of Full Adder

All rights reserved by *www.ijsrd.com*

A	B	Cin	Sum	Cout
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Fig. 2.9: Truth Table of Full adder

1) Operations of Logical Unit

OPERATION	M	S2	S1	S0	OUTPUT
LOGIC	0	0	0	0	A&B
	0	0	0	1	A|B
	0	0	1	0	A ⊙ B
	0	0	1	1	A ⊕ B
	0	1	0	0	~A
	0	1	0	1	~B
	0	1	1	0	~(A+B)
	0	1	1	1	~(AB)

G. Operations of Arithmetic Unit

OPERATION	M	S1	S0	CIN	OUTPUT
ARITHMETIC	1	0	0	0	A
	1	0	0	1	A+1
	1	0	1	0	A-1
	1	1	0	0	A+B
	1	1	0	1	A+B+1
	1	1	1	0	A+~B
	1	1	1	1	A-B

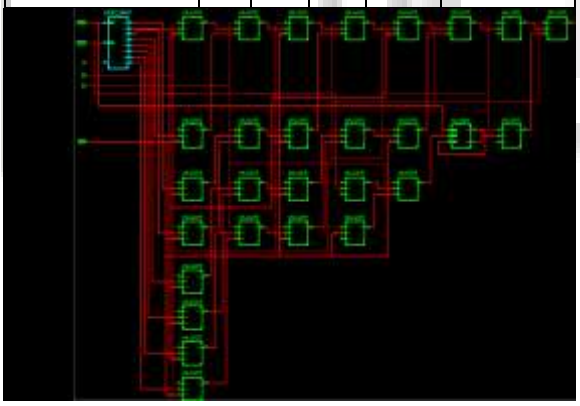

Fig. 2.10: RTL Schematic Diagram of 64 Bit QCA ALU

IV. SIMULATION RESULTS

Fig. 3.1: Simulation of 4 bit QCA ALU

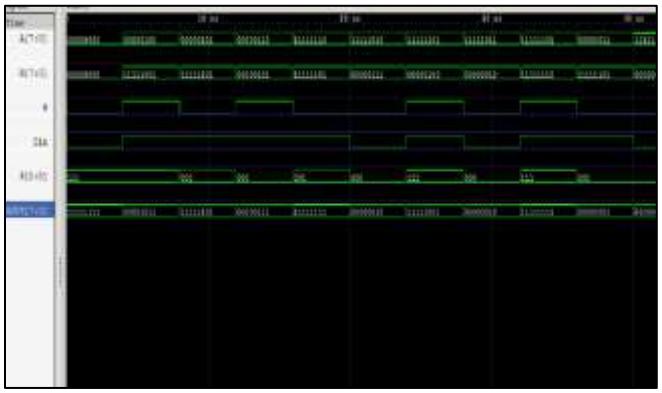

Fig. 3.2: Simulation of 8 bit QCA ALU

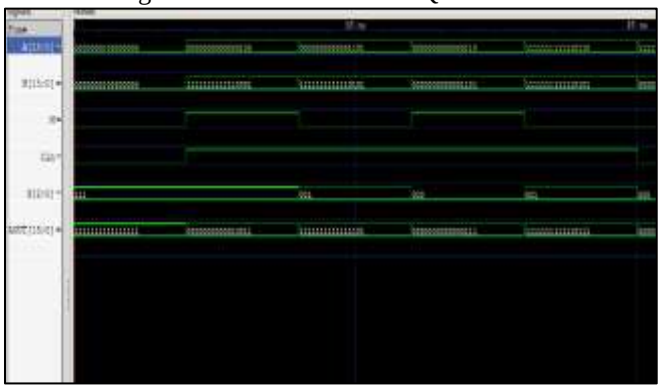

Fig. 3.3: Simulation of 16 bit QCA ALU

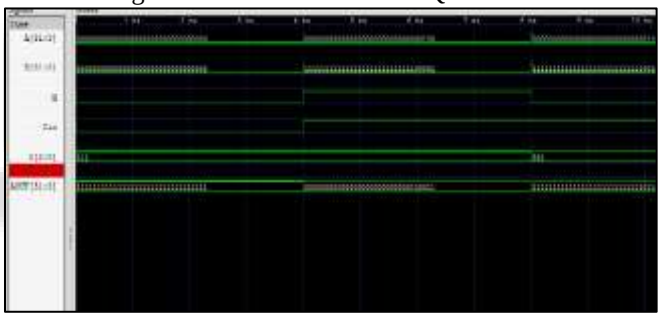

Fig. 3.4: Simulation of 32 bit QCA ALU

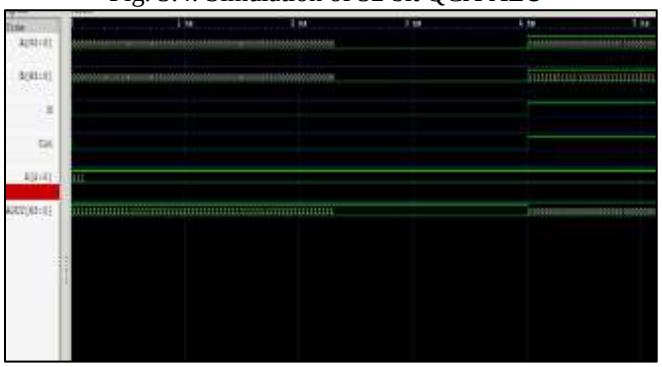

Fig. 3.5: Simulation of 64 bit QCA ALU

A. Comparison between Various Types of ALU Using QCA

	DELAY(ns)
1 BIT ALU	0.418
4 BIT ALU	3.041
8 BIT ALU	3.902
16 BIT ALU	5.905
32 BIT ALU	9.998
64 BIT ALU	18.013

All rights reserved by *www.ijsrd.com*

V. CONCLUSION

The goal of this work is to create a 64-bit arithmetic logic unit using Quantum-Dot Cellular Automata Technology. This ALU performs 15 operations with the help of multiplexers and adders, which have been replaced with majority gates. Xilinx Vivado 14.2 ISE Design Tool with Verilog coding was used to design and simulate this. Higher Bit operations can be increased in the future with this coding. Because multiplexers are used, it is simple to handle.

REFERENCES

[1] H. Rashidi, A. Rezai, and S. Soltany, "High-performance multiplexer architecture for quantum-dot cellular automata," *J. Comput. Electron.*, vol. 15, no. 3, pp. 968–981, 2016.
[2] R. Jaiswal and T. N. Sasamal, "Efficient design of full adder and subtractor using 5-input majority gate in QCA," in *10th International Conference on Contemporary Computing, (IC3)*, 2017, pp. 1–6.
[3] D. Abedi and G. Jaberipur, "Decimal Full Adders Specially Designed for Quantum-Dot Cellular Automata," *IEEE Trans. Circuits Syst. II Express Briefs*, vol. PP, no. 99, pp. 1–5, 2017.
[4] B. Ghosh, A. Kumar, and A. K. Salimath, "A Simple Arithmetic Logic Unit (12 ALU) Design Using Quantum Dot Cellular Automata," *Adv. Sci. Focus*, vol. 1, no. 4, pp. 279–284, Dec. 2013.
[5] M. G. Waje and P. K. Dakhole, "Design and implementation of 4-bit arithmetic logic unit using Quantum Dot Cellular Automata," 2013, pp. 1022–1029.

All rights reserved by *www.ijsrd.com*