IJARSCT

ISSN (Online) 2581-9429

International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)

International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal

Impact Factor: 7.53

Volume 4, Issue 2, May 2024

An Efficient Design of 2-BIT Arithmetic Logical Unit in Quantum Dot Cellular Automata

Gaurang Tyagi, Sadhana Kumari, Ayushi Ojha, Shivanshu Nanda

Department of Electronic and Communication Engineering

JSS Academy of Technical Education, Noida, (U.P), India

gaurangtyagi19@gmail.com, sadhana.kumari@jssaten.ac.in, ayushiojha1012@gmail.com, shivanshun33@gmail.com

Abstract: *For the past two decades, CMOS technology has reigned supreme in the world of Very Large Scale Integration (VLSI). But Quantum Dot Cellular Automata (QCA) is stepping into the spotlight, offering a fresh solution to the growing challenges of CMOS. This paper delves into the potential of QCA circuits by presenting a 2-bit Arithmetic Logic Unit (ALU) crafted with this groundbreaking technology, while also benchmarking it against conventional CMOS designs. The proposed QCA-based ALU delivers a trifecta of benefits: streamlined circuit design, exceptional space efficiency, and reduced quantum cost—all while keeping performance sharp in terms of latency and area usage. This 2-bit ALU encompasses a suite of operations: addition, subtraction, multiplication, division, bitwise AND, OR, XOR, and XNOR. QCA technology addresses some of CMOS's major pain points, like slow switching speeds, and doesn't require additional power sources, making it both efficient and environmentally friendly. This is not just an incremental improvement; it's a bold leap into the future of circuit design*

Keywords: Clock Signal; Adder; Subtractor; Multiplier; Switching Speed; Demultiplexer

I. INTRODUCTION

Quantum Dot Cellular Automata (QCA) employ quadratic cells containing four potential wells situated at the corners, linked by electron tunnel junctions. Each cell accommodates precisely two electrons, which, due to their repulsive forces, occupy opposing corners. This yields two possible configurations, one representing binary 0 (low) and the other binary 1 (high).

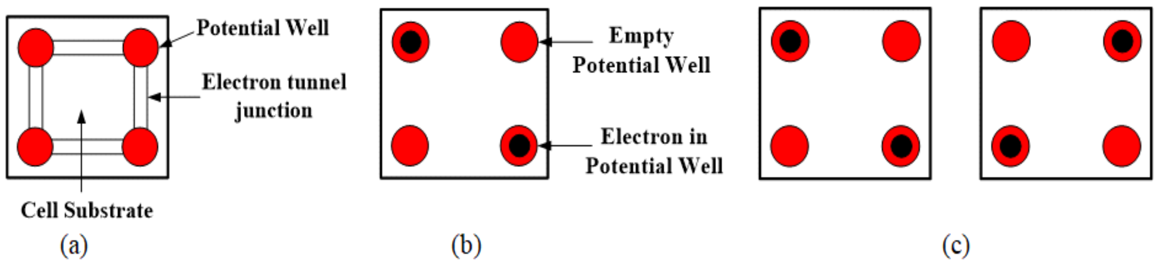


Fig. 1(a) QCA Cells with electron tunnel junction (b) QCA Cell with potential well (c) Cells with polarization p = -1 (Logic '0') and p = 1 (Logic '1')

CLOCKING

Quantum Cellular Automata (QCA)-based circuits function through four distinct clock phases: Switch, Hold, Release, and Relax.

During the Switch phase, neighboring cells influence the polarization of additional electrons within a cell, resulting in the cell acquiring a specific binary value. Tunnel junctions attempt to close, leading to a continuous increase in the potential barrier. In the Hold phase, the potential barrier reaches its maximum, closing the tunnel junctions to prevent electron switching and maintaining their polarity. In the Release phase, the potential barrier gradually decreases, prompting the tunnel junctions to open and causing cells to lose their polarity. During the Relax phase, the potential barrier is minimal, and tunnel junctions remain open, resulting in cells having no impact on neighboring cells. In QCA, cells of different colors signify they are synchronized to the same clock. Specifically, Green represents clock 0, Violet represents clock 1, Blue represents clock 2, and White represents clock 3.

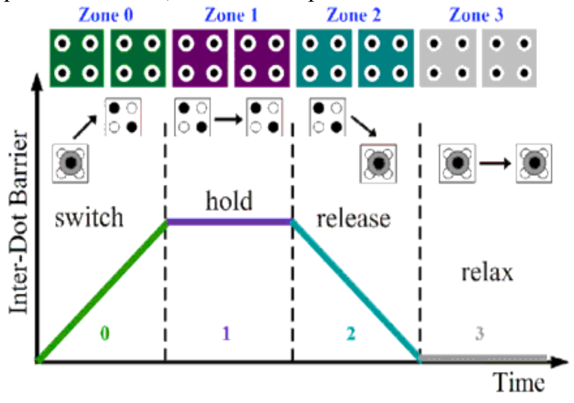


Fig. 2: Clocking in QCA

II. PROBLEM STATEMENT

CMOS (Complementary Metal-Oxide-Semiconductor) technology has been the backbone of the semiconductor industry for several decades, providing the foundation for the manufacturing of integrated circuits (ICs). While CMOS has been extremely successful, it is not without its challenges. Some of the key issues with CMOS technology include:

Size Limitations: As semiconductor technology advances and the demand for smaller, more powerful devices increases, the size limitations of CMOS become a challenge. As components on a chip shrink, quantum effects become more pronounced, leading to issues such as increased leakage currents and reduced reliability.

Power Consumption: With the miniaturization of transistors, power consumption becomes a critical concern. As the size of transistors decreases, the power density increases, leading to higher power consumption and heat generation. This can result in challenges related to cooling and energy efficiency.

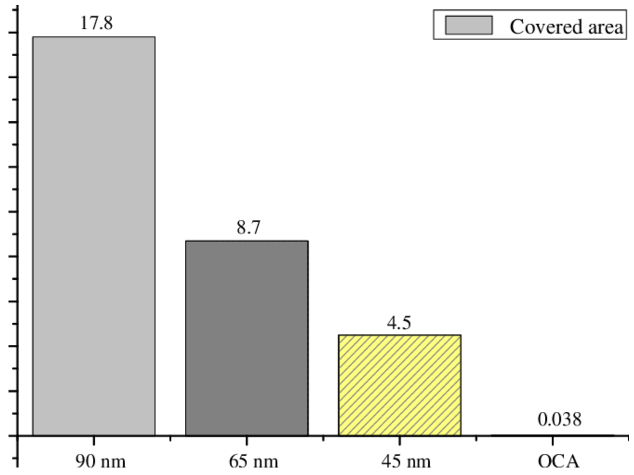


Fig. 3 Area in CMOS and QCA

Moore's Law Challenges: Moore's Law, which observed that the number of transistors on a chip doubles approximately every two years, has been a driving force in the semiconductor industry. However, as feature sizes approach the physical limits imposed by quantum mechanics, sustaining this rate of scaling becomes more challenging. The industry has faced difficulties in continuing to adhere to Moore's Law, leading to discussions about its potential limitations.

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
Impact Factor: 7.53
Volume 4, Issue 2, May 2024

Heat Dissipation: As more transistors are packed onto a chip, managing the heat generated becomes a significant challenge. Excessive heat can degrade the performance and reliability of the components, necessitating advanced cooling solutions.

III. OBJECTIVE

The project's goal is to compare the performance of a conventional FPGA-based Arithmetic Logic Unit (ALU) with that of a Quantum-Dot Cellular Automata (QCA) based ALU, focusing on power consumption and size. It will assess individual arithmetic and logic operations like addition, subtraction, multiplication, division, as well as bitwise logical operations such as AND, OR, XOR, and XNOR. By doing so, the project aims to shed light on the practicality and effectiveness of utilizing QCA technology for ALU functions, thereby contributing to the development of computing architectures that are both energy-efficient and high-speed.

IV. PROPOSED DESIGN AND IMPLEMENTATION

The Arithmetic Logic Unit (ALU) proposed in this study presents a compact yet versatile computational framework for performing arithmetic and logical operations on two 2-BIT numerical values.
Various components of the Arithmetic Logical Unit have been fully simulated using QCA technology. Compared to CMOS, QCA has significantly advanced the miniaturization of hardware devices. The ALU implementation relies on Quantum-dot Cellular Automata (QCA) as its foundational computational framework. The block diagram of proposed ALU structure is illustrated in Figure [3]. The interconnectedness of these blocks forms a unified architecture that facilitates effective arithmetic computation while ensuring a smooth user experience

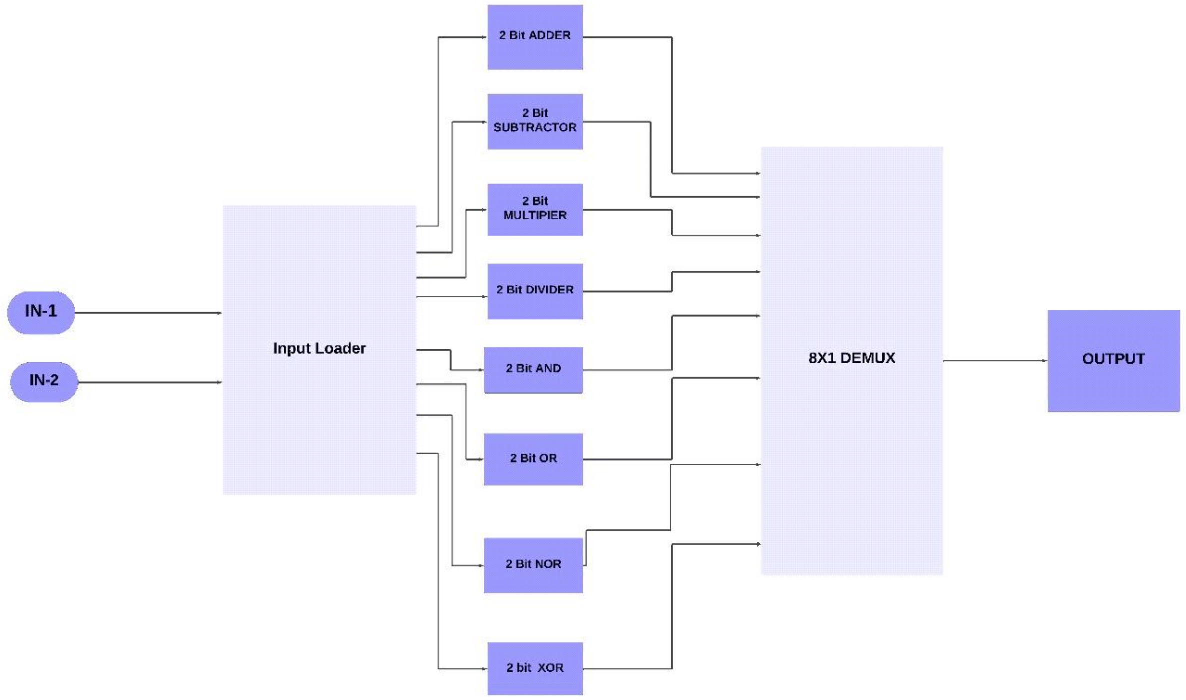


Fig. 3(a): Proposed Block Diagram

The ALU encompasses vital components, meticulously crafted and simulated utilizing QCA technology. Each component is discussed in detail below.

ADDER

A 2-bit adder is a type of digital circuit that performs addition on two binary numbers, each containing two bits. It typically uses logic gates and full adders to achieve this task.
In a 2-bit binary system, the numbers A and B can be described as follows:

Number A: Made up of two bits, with a1 representing the most significant bit (MSB) and a0 representing the least significant bit (LSB).
Number B: Similarly comprises b1 and b0 with the same structure as number A.

OUTPUT
The output of a 2-bit adder consists of three components:
Sum bits: These are the resulting two bits of the addition, denoted as s1 and s0 epresenting the most and least significant bits of the sum, respectively.
Carry bit: A single bit, referred to as the carry, which is used when an overflow occurs during addition.
This setup allows the 2-bit adder to calculate the sum of numbers A and B, giving a two-bit result and a carry bit if there's an overflow.

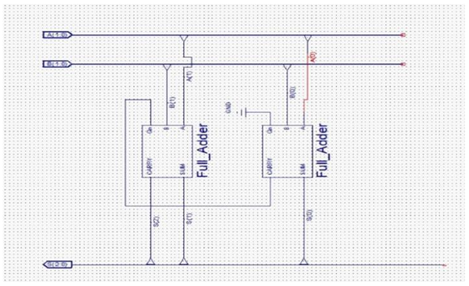
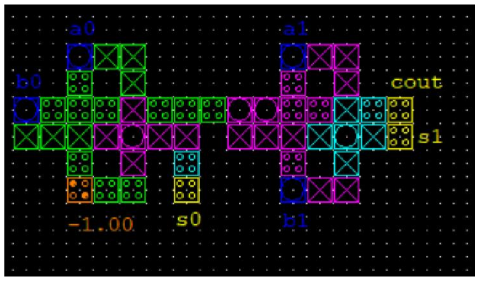


Fig. 4(a) Xilix Implementation of 2-bit adder



(b) QCA implementation of 2-bit adder

SUBTRACTOR
A 2-bit subtractor is a digital circuit that calculates the difference between two binary numbers, each composed of two bits. It accounts for "borrowing" when the first number (the minuend) is smaller than the second number (the subtrahend). Here's how a typical 2-bit subtractor operates:
The 2-bit subtractor processes the following steps:

Subtraction of the Least Significant Bit (LSB):
Start by subtracting the LSBs A0 - B0
If *A*0 is greater than or equal to B0, the difference is straightforward, and there's no borrowing.
If *A*0 is smaller than B0, borrowing is required from A1, and the resulting difference is 2+ A0-B0.

Subtraction of the Most Significant Bit (MSB)
Now subtract the MSB, considering whether borrowing occurred from the previous step.
If borrowing did occur, the effective minuend becomes A1 -1.The new calculation is A1 - B1 - 1.
This paper presents a 2-bit subtractor built using QCA, with Figure 5(a) illustrating its implementation through Xilinx technology.
The design places emphasis on handling borrowing and overflow to ensure that the subtraction operation provides accurate results.

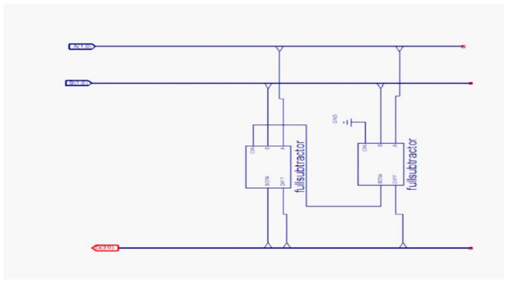
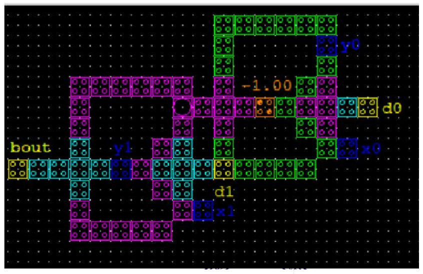


Fig. 5(a) Xilinx Implementation of subtractor



(b) QCA implementation of 2-bit subtractor

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
Impact Factor: 7.53
Volume 4, Issue 2, May 2024

MULTIPLIER

A multiplier is a digital circuit or functional unit that performs multiplication operations on binary numbers. It takes two binary inputs (operands) and outputs their product.
In binary multiplication, the process involves a series of shift-and-add steps. Given two binary numbers as inputs, the multiplier generates the product through Generating Partial Products.This step involves multiplying each bit of one operand by each bit of the other operand, creating several partial products. These partial products are then summed to get the final result.

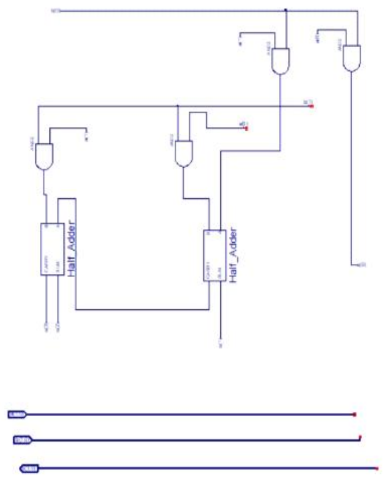
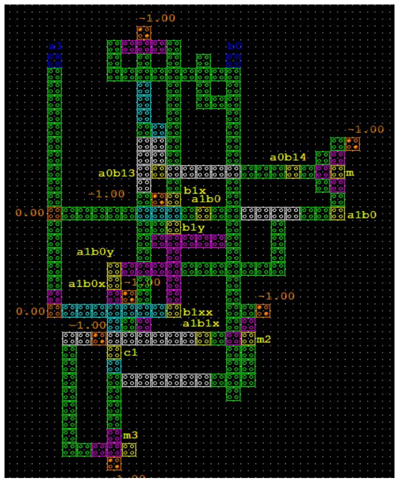


Fig. 6(a) Xilinx Implementation of multiplier



(b) QCA implementation of multiplier

DIVIDER

A 2-bit divider is a digital circuit designed to perform division operations on two-bit binary numbers. This kind of divider receives two binary inputs: the dividend, and the divisor and produces a quotient and a remainder as outputs. Given the small size of these binary numbers, a 2-bit divider is a simple circuit, but it still must handle division rules, including special cases like division by zero.

Components of a 2-bit Divider

Dividend: A 2-bit binary number representing what is being divided. It has bits $d1,0$, where d1 is the most significant bit (MSB) and $d0$ is the least significant bit (LSB).
Divisor: A 2-bit binary number that divides the dividend. It also has bits $b1,0$
Quotient: The result of the division. This is also a 2-bit binary number.
Remainder: What is left over after the division operation. This is a single-bit or 2-bit value, depending on the divisor.

Operation

The operation of a 2-bit divider involves the following steps:
Checking for Division by Zero: The first step in any division operation is to ensure the divisor is not zero. If it is, the operation should trigger an error or exception, as division by zero is undefined.

Performing Division:

Finding Quotient: Get the quotient, the divider determines how many times the divisor can fit into the dividend. This is often done using subtraction and bit-shifting.
Calculating Remainder: After finding the quotient, the remainder is what is left after dividing as many times as possible without exceeding the dividend. If the dividend is smaller than the divisor, the quotient is zero, and the remainder is the dividend itself.

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
Impact Factor: 7.53
Volume 4, Issue 2, May 2024

Producing Output: The outputs are the quotient and the remainder. The quotient is a 2-bit value, while the remainder might be one or two bits, depending on the size of the divisor.

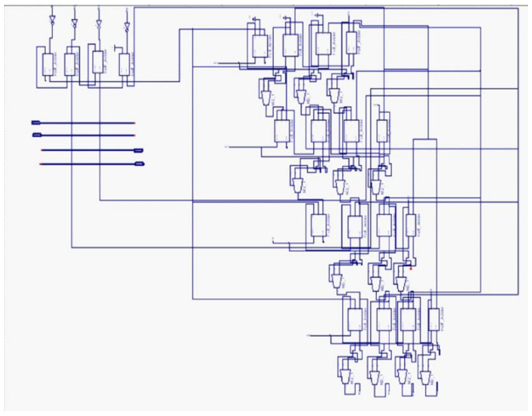
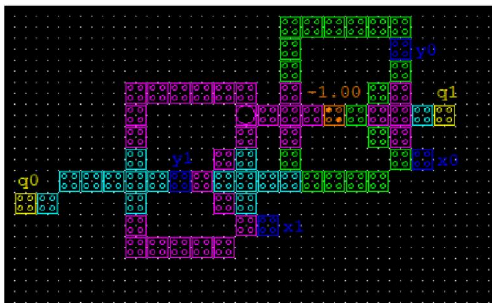
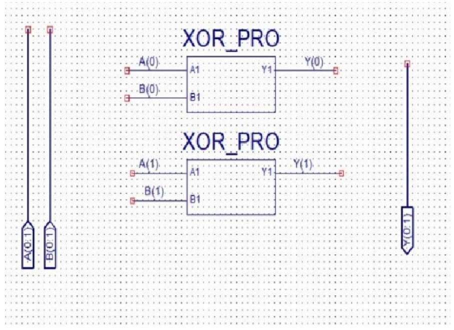



Fig. 7(a) Xilinx Implementation of divider (b) QCA implementation of divider

BITWISE XOR

Bitwise XOR (exclusive OR) is a fundamental binary operation that acts on binary numbers bit by bit. It compares corresponding bits of two binary numbers and returns a result based on the following rule: the output bit is 1 if the corresponding bits of the input numbers are different, and 0 if they are the same.



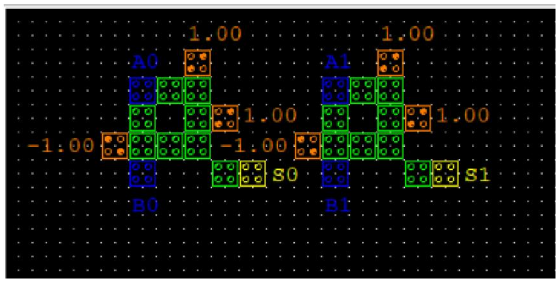
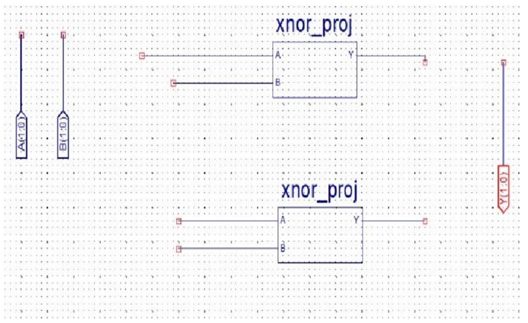


Fig. 7(a) Xilix Implementation of 2-bit XOR (b) QCA implementation of 2-bit XOR

BITWISE XNOR

Bitwise XNOR (exclusive NOR) is a fundamental binary operation that acts on binary numbers bit by bit. It's the complement or negation of the XOR (exclusive OR) operation. While XOR returns 1 when two corresponding bits are different and 0 when they are the same, XNOR does the opposite: it returns 1 when the bits are the same and 0 when they are different.



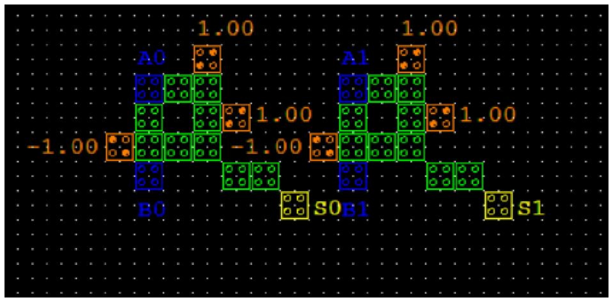


Fig. 8(a) Xilix Implementation of 2-bit XNOR (b) QCA implementation of 2-bit XNOR

IJARSCT

ISSN (Online) 2581-9429

International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)

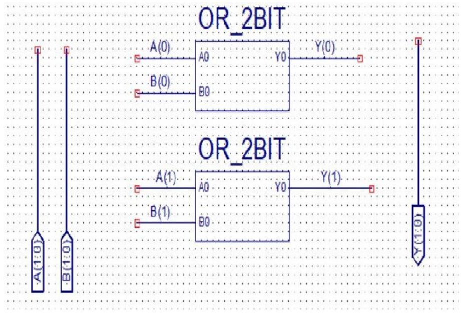
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal

Impact Factor: 7.53

Volume 4, Issue 2, May 2024

BITWISE OR

Bitwise OR is a binary operation that compares corresponding bits of two binary numbers and returns a result where each bit is set to 1 if either of the corresponding bits in the input numbers is 1. It is used in various applications, including digital circuit design, data manipulation, and logic operations.



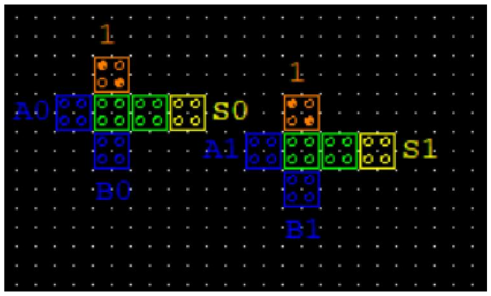
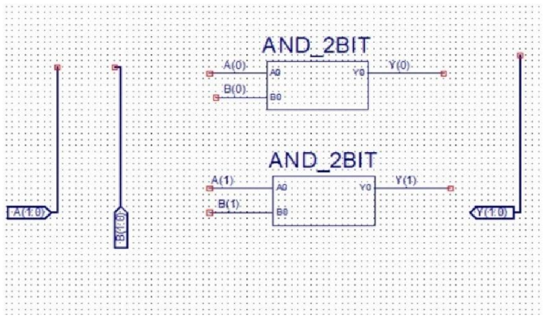


Fig. 9(a) Xilix Implementation of 2-bit OR (b) QCA implementation of 2-bit OR

BITWISE AND

Bitwise AND is a binary operation that works on two binary numbers, comparing each corresponding pair of bits and returning a result where each bit is set to 1 if both corresponding bits from the input numbers are 1. Otherwise, the output bit is set to 0. This operation is commonly used in computer programming, digital circuit design, and logical operations.



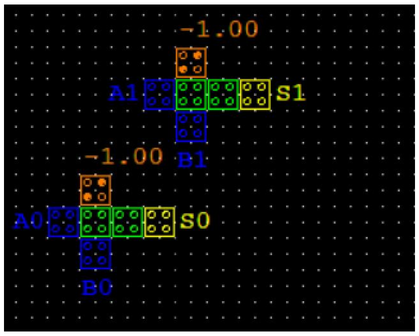


Fig. 4(a) Xilix Implementation of 2-bit AND (b) QCA implementation of 2-bit AND

V. COMPARISION

Summary of Findings

In this study, we conducted a comparative analysis of Arithmetic Logic Unit (ALU) components implemented in Quantum-dot Cellular Automata (QCA) and Complementary Metal-Oxide-Semiconductor (CMOS) technologies. Our investigation focused on assessing the size and energy consumption of various ALU components in both QCA and CMOS architectures.

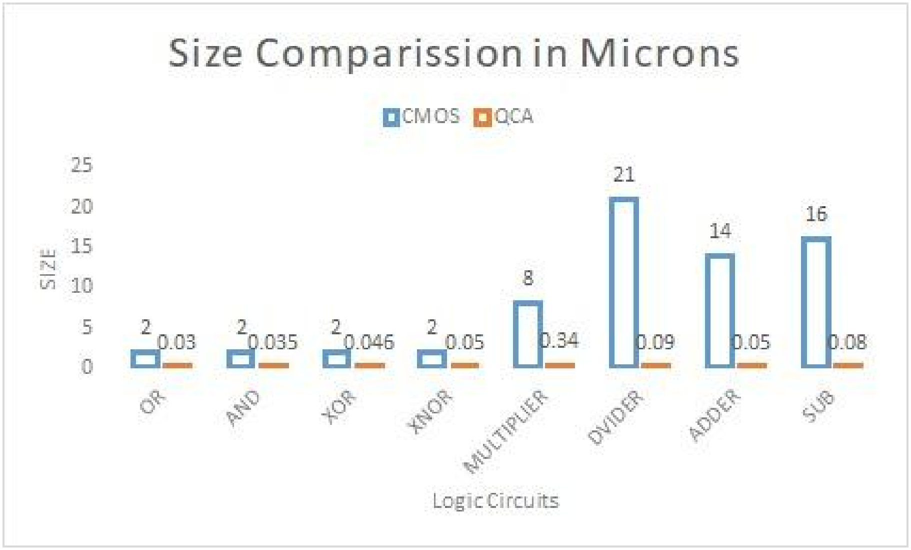
Analysis of Data

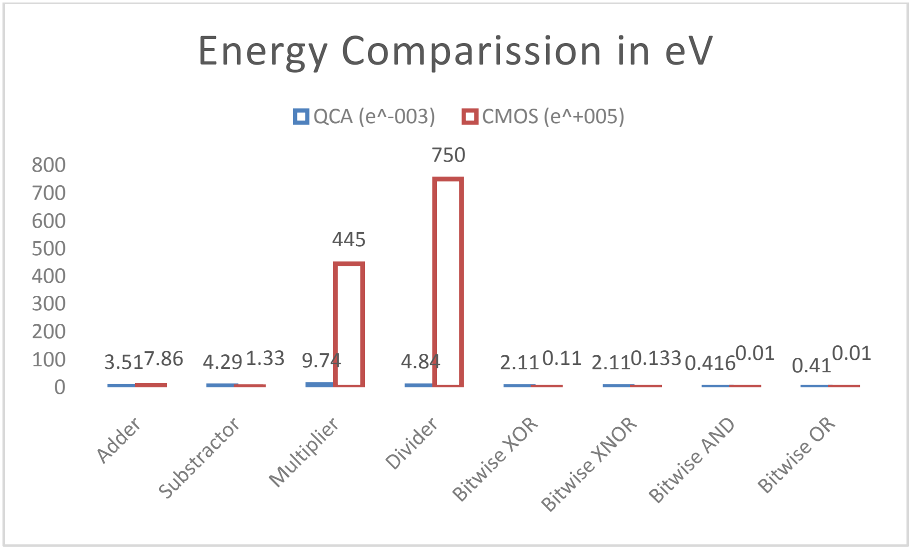
The following table presents a detailed comparison of ALU components in QCA and CMOS technologies:

Component	Technology	Size(μm²)	Energy Consumption (eV)
Adder	QCA	0.05	3.51e-003
	CMOS	14	7.86e+005
Subtractor	QCA	0.08	4.29e-003
	CMOS	16	1.33e+005
Multiplier	QCA	0.34	9.74e-003
	CMOS	8	4.45e+007

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
Impact Factor: 7.53
Volume 4, Issue 2, May 2024

Divider	QCA	0.09	4.84e-003
	CMOS	21	7.5e+007
Bitwise XOR	QCA	0.046	2.11e-003
	CMOS	2	1.1e+004
Bitwise XNOR	QCA	0.05	2.11e-003
	CMOS	2	1.33e+004
Bitwise AND	QCA	0.035	4.16e-004
	CMOS	2	1e+003
Bitwise OR	QCA	0.03	4.16e-004
	CMOS	2	1.1e+003





VI. CONCLUSION

The data presented in the table demonstrates that QCA-based ALU components generally exhibit smaller sizes and lower energy consumption compared to their CMOS counterparts. This can be attributed to the inherent advantages of QCA technology, such as the absence of leakage current and reduced parasitic capacitance.

Furthermore, the size reduction in QCA-based ALU components can lead to higher integration density and potentially lower manufacturing costs. However, it's essential to consider the trade-offs associated with QCA technology, such as the requirement for specialized fabrication processes and the limited scalability of QCA circuits.

In conclusion, our comparative analysis highlights the potential advantages of QCA technology in terms of size and energy efficiency for ALU implementation. While CMOS technology remains dominant in current integrated circuit design, the findings of this study suggest that QCA holds promise as a viable alternative, particularly for applications demanding compactness and low power consumption.