



AREA EFFICIENT 1-BIT ARITHMETIC & LOGICAL UNIT USING QUANTUM DOT CELLULAR AUTOMATA

Dr.Mrs.Vrushali.A.Kailuke , Rutuja Bhusari, Shravya Somkuwar, Twinkle Mishra,

Nandini Badole

Department of Electronics and Telecommunication Engineering, Priyadarshini College of Engineering, Nagpur, Maharashtra, India.
Email: vrushali.nasre@pcenagpur.edu.in, rutujabhushari316@gmail.com , shravyasomkuwar@gmail.com,
mishratwinkle384@gmail.com, badolenandini2005@gmail.com,

Abstract

The growing demand for compact, energy-efficient computing systems highlights the limitations of conventional CMOS circuits, which consume excessive power. Cellular Automata (QCA) offers a promising alternative by encoding information through electron arrangements in quantum dots, eliminating current-based switching. This enables the development of nanoscale logic circuits with low power consumption and reduced size. This review details the design and simulation of a 1-bit Arithmetic Logic Unit (ALU) using QCA technology. The ALU carries out arithmetic and logic operations using main QCA components like majority gates, inverters, wires, and multiplexers. Its single-layer, crossover-free design makes the layout simpler and avoids complex signal crossings. The circuit is built and tested in QCA Designer, and its performance is measured by cell count, area, latency, and energy use.

The circuit is designed and simulated in QCA Designer, with performance evaluated by cell count, area, latency, and energy consumption. Results show that the proposed architecture is more compact and efficient than existing QCA-based ALUs. Its low complexity and energy requirements make it suitable for future nanoscale digital systems where area and energy efficiency are essential. These results confirm the potential of QCA technology as a viable alternative to CMOS for compact arithmetic and logic circuits.

Keywords: Quantum-dot cellular automata, QCA, 1-bit ALU, arithmetic unit, logic unit, majority gate, full adder, multiplexer, QCADesigner

INTRODUCTION

1. CMOS Technology Overview

Complementary Metal-Oxide-Semiconductor (CMOS) technology was introduced in 1963 by Frank Wanslehurst while working at Fairchild Semiconductor. It combines N-type and P-type MOSFETs and is used to build integrated circuits, microprocessors, microcontrollers, detectors, RAM, and many other digital circuits. Early PMOS and NMOS logic suffered from very high static power consumption. This was solved by logic built from complementary p- and n-type MOSFETs, so that the circuit draws almost no power in a static state. CMOS has two main advantages: it resists noise well and uses very little power when idle[7]. CMOS has followed Moore's observation that the number of transistors on an integrated circuit doubles roughly every 18 to 24 months [1]. To meet this demand, CMOS has been scaled from millimeters down to nanometres [2]. Industry then moved to non-planar (3D) transistors, and feature sizes were reduced to 7 nm [3]. Fig. 1 of the paper shows the trend from 10 μm in 1970 to 7 nm in 2017 [3]-[6]. New innovations are essential to keep shrinking CMOS integrated circuits [6]. Graphene is being explored as a future channel material [4]. It has high electron and hole mobility, a flat structure, good thermal conductivity, high current-carrying capacity, and is very thin.[5].

2. QCA Technology Overview

Motivation. Since the 1970s, the field-effect transistor (FET) has been the device used for higher levels of integration, and its size has been halved roughly every 18 to 24 months. Your paper says the ITRS projected that scaling of FET-based ICs would reach manufacturing limits by 2020, so the industry has to move to an alternative paradigm of information representation and processing. Quantum dots offer such an alternative to CMOS. In QCA, binary information is encoded in the charge of quantum dot cells, so computation is done with quantum dots [8]. The paper's survey structure follows this overview [9]. A QCA circuit is built from QCA cells, and how these cells are arranged decides what the circuit does. Each standard cell contains four quantum dots and two electrons. The electrons sit on opposite corners, and each diagonal stands for either logic "1" or logic "0". A QCA wire is just a line of these cells. A regular wire passes the input signal through, but a rotated wire can either keep or flip the signal, depending on how it is set up and how many cells it has [10].

The majority gate and inverter are key parts of QCA circuits. The main building block is the 3-input majority voter, which is written as $M(A, B, C) = AB + BC + CA$.

Fixing one input at logic 0 gives an AND gate, and fixing it at logic 1 gives an OR gate. Together with the inverter, these form the basis for complex QCA circuit designs [10].

QCA clocking. QCA works with a synchronization system that has four clock phases: switch, hold, release, and relax. During these phases, the inter-dot potential barriers are raised and lowered, so signals propagate sequentially through different zones [10]. The clock controls the cells through their tunneling barriers. When the barriers are low, electrons tunnel freely, and when they are high, the cell locks its state [11].

QCA crossover. Complex circuits need signals to cross without interference. Coplanar crossovers use rotated cells in the same plane. Multilayer crossovers route signals on separate layers [10].

QCADesigner tool. QCADesigner is open-source software that helps users quickly and accurately design and simulate QCA circuits. The Walus Group at the University of British Columbia created it, and it is available under the GNU GPL. To use it, you lay out the cells, set up the inputs and outputs, configure the simulation engine, run the simulation, and debug until the waveforms look right [11].

2.1 QCA Cells and Wiring

A QCA circuit is built from QCA cells, and how these cells are arranged decides what the circuit does [10]. Each standard QCA cell has four quantum dots, with two of them holding mobile electrons. Because of electrostatic repulsion, these electrons settle in opposite corners of the cell.

Each diagonal arrangement forms a stable polarization state that represents a binary logic value. One diagonal represents logic “1” (polarization $P = +1$), while the other represents logic “0” ($P = -1$). Figure 1 shows the structure of a QCA cell and its two configurations for logic ‘1’ and logic ‘0’. Unlike CMOS, this cell stores binary information in the cell's electron polarization rather than in current flow.

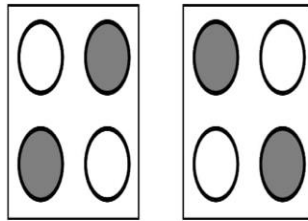


Fig. 1: Polarization in QCA cell [10]

A QCA wire is simply a sequence of QCA cells. One cell's polarization influences its neighbor through electrostatic interaction, so the logic state transfers from cell to cell along the wire [10]. In a normal wire, the input signal is transmitted unchanged along the cable. In a rotated wire, where the cells are rotated, the signal can be reversed or maintained, depending on the configuration and the number of cells [10]. Fig. 2,3 illustrates signal propagation in the normal and the rotated QCA wire.

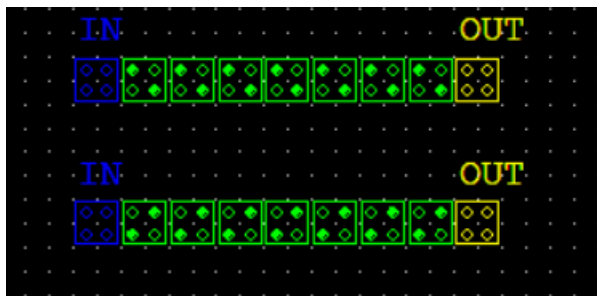


Fig. 2: 90° wiring in QCA

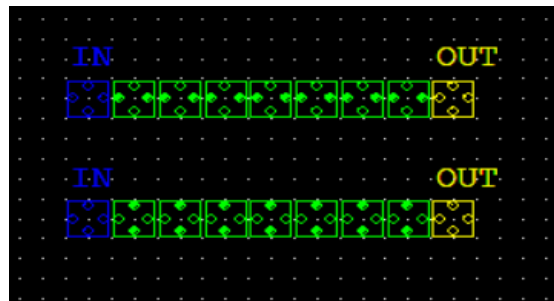


Fig. 3: 45° wiring in QCA

2.2 Inverter & Majority Gate

In QCA, the basic gates are the inverter, the majority gate, and the XOR gate. These gates form the foundation of larger circuits, including multiplexers, full adders, and comparators [4]. Two common inverter layouts, corner and robust, and two three-input majority-gate layouts, original and rotated, are commonly used [4]. The three-input majority gate is written as $M(A, B, C) = AB + BC + CA$. By setting one input to logic “0,” the majority gate can function as an AND gate [4]. The standard QCA majority gate has three inputs and one output, which makes it

irreversible. This means it loses information and dissipates energy into the environment. In fact, the majority gate is the main source of energy loss in QCA circuits [9]. A fully reversible majority gate addresses this issue by copying the input data, so it has three inputs and three outputs, and loses no data [9]. Because the majority gate is so important in QCA digital circuit design, researchers are actively studying ways to improve its fault tolerance.

Most earlier work improved reliability by adding QCA cells, which increased the area [10]. A newer three-input majority gate uses only 11 cells and $0.006 \mu\text{m}^2$ of area. It tolerates single-cell omission defects by 86% and extra-cell deposition defects by 75% [10]. It has been used to build a full adder/subtractor, a 2-to-1 multiplexer, and a one-bit ALU with four operations [10]. Inverters and majority gates together form complete circuits. In the reversible design of [9], the XOR gate uses two inverters and three majority gates. The half-adder uses four majority gates and two inverters, and the half-subtractor uses two inverters and three majority gates. The 2-to-1 multiplexer uses three majority gates and one inverter, while the 4-to-1 multiplexer uses nine majority gates and three inverters [9]. A 1-bit comparator uses two original majority gates, three inverters, and one XOR gate. It requires 38 cells, covers $0.03 \mu\text{m}^2$ of area, and has a delay of 0.5 clock cycles [4]. QCA circuits that use majority gates with more than three inputs benefit from the real clocking technique [9].



Fig. 4: Majority gate

Fig. 5: Inverter

2.3 QCA Clocking

Clocking is essential in QCA circuits because it coordinates the flow of data. An external clock changes the strength of the tunneling barriers between QCA cells, and this is how clocking control is achieved [9]. The standard method is adiabatic switching, introduced by Lent and Tougaw to control timing, avoid metastability, and make pipelining possible [9]. The QCA array is split into groups of cells called clock zones. Each zone performs its computation, stores its state, and sends the result to the next zone [9].

A long wire risks cells not switching accurately because of thermodynamic limits, so dividing it into zones works like splitting it into several shorter wires [9].

Each clock cycle has four phases :

Switch: The cells start unpolarized with low barriers. As the barriers rise, the cells polarize according to the computation, and the phase ends with high barriers and fixed cell states.

Hold: The barriers stay at their current height.

Release: The barriers are lowered, and the cells depolarize.

Relax: The barriers stay low, and the cells remain unpolarized. The cycle then starts again.

Adiabatic switching keeps the system in its instantaneous ground state and makes sure that no signal reaches a gate before the other inputs arrive [9]. However, the one-dimensional scheme has problems. Variations in wire length, clock-zone capacity, and the number of cells per zone can block feedback paths, leaving some areas unused [9]. Two-dimensional clocking, proposed by Vankamamidi et al., uses a grid of equal square zones. It gives higher performance and lower power by exploiting spatial and temporal parallelism, but the overhead of feedback paths remains a challenge. In larger circuits, long lines between zones increase delay and sensitivity to thermal fluctuations [9]. The USE (universal, scalable and efficient) scheme, developed by Campos et al., addresses these limits. It supports feedback loops of any size, standard cell libraries, and simpler routing [9]. It has four clock zones numbered 1 to 4, which form one complete clock cycle, and each tile is a 5×5 cluster of cells [9]. Synchronization can be local or global. With local synchronization, data moves only between zones that have consecutive numbers. In global synchronization, new data enters the circuit every cycle, which keeps all gate inputs aligned.

Most researchers treat local synchronization as essential, while the importance of global synchronization is disputed [9]. Without clock synchronization, the next stage may receive wrong bits, which leads to incorrect data transfer [9]. Real clocking, in its pipeline or dynamic style, can reduce manufacturing cost and make physical implementation easier. Circuits that use majority gates with more than three inputs gain the most advantage from this approach [9].

Delay in QCA circuits is counted in clock zones. Four clock zones make one clock cycle [9].

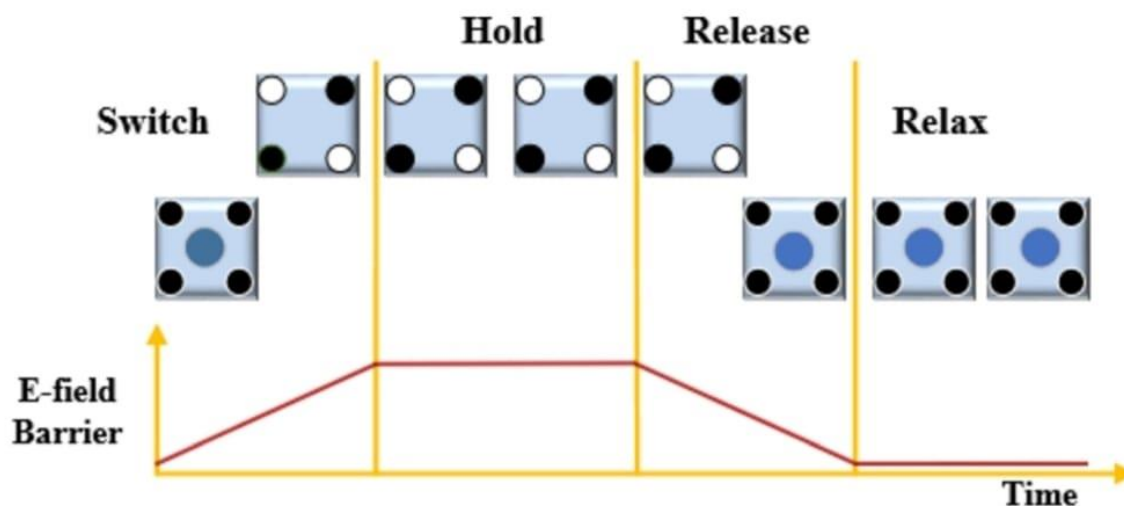


Fig. 6: Clocking in QCA [13]

2.4 QCA Crossover

One of the main challenges in digital logic design is handling wire junctions. In QCA, crossovers address this problem. There are two options: coplanar and multilayer [6]. In a coplanar crossover, both signals cross within the same plane of cells. Many QCA designs use this approach, for example, the 1-bit comparator in [4]. Coplanar layouts have also been used for a fault-tolerant full adder/subtractor and a one-bit ALU built on a robust majority gate [10]. Another coplanar design uses fundamental gates built from a single cell that is not rotated [13]. A multilayer crossover uses more than one layer of cells [6]. Bajec and Pecar's multilayer technique uses three separate layers to address the wire junction problem [9]. Multilayer designs give more reliable circuits [9] and more robust simulation outputs [6]. However, making a multilayer crossover is considerably more expensive than a coplanar one, which affects manufacturing feasibility [6]. In simulations, the layers are separated by 11.5 nm [9]. The choice of crossover affects the cost figures. For example, [4] notes that a multilayer comparator has a slightly smaller cell count and area than its own coplanar design. It attributes this to the added layers rather

than to the logic design [4]. Different crossover methods are also used across ALU designs. A reversible 16-operation ALU uses the multilayer method [9], and a 2-bit ALU with multilayer wiring needs 294 cells and $0.61 \mu\text{m}^2$ [6].

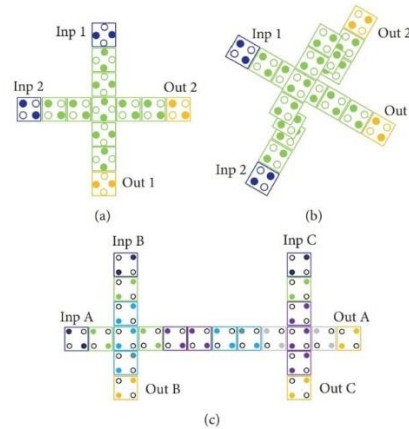


Fig. 7: QCA crossover: (a)coplanar crossing, (b)multilayer crossing, (c)logical crossing[3]

2.5. QCA Designer Tool

QCADesigner is a popular computer-aided design tool for QCA technology, used for both layout and post-layout simulation of QCA circuits [9]. Its coherence vector simulation engine (CVSE) models QCA cell operation at the quantum and microscopic levels [9]. The tool helps evaluate circuit performance, area cost, latency, and reliability [9]. If a simulation gives unexpected output, the design is treated as failed. The designer then goes back and checks gate placement, pin locations, routing, and layout verification [9]. QCADesigner-E includes a power-dissipation feature in the coherence-vector model used by QCADesigner [9]. It uses an iterative, fixed-timestep method to solve the differential equations that describe how the quantum mechanical density matrix changes over time [9]. A timestep of 0.1 fs gave an acceptable numerical energy conservation error, below 5% [9]. QCADesigner-Energy (QD-E) version 2.0.3 was first applied to find the overall dissipated energy of QCA adder and subtractor circuits in [3]. It has since been used with QCADesigner 2.0.3 for area and energy analysis of majority-gate-based circuits [10]. Designs are commonly verified in QCADesigner version 2.0.3 [4]. In [4], the usual simulation parameters include 12800 samples, a 65 nm radius of effect, a convergence tolerance of 0.001, and a relative permittivity of 12.9. The clock low and high energies are 3.8×10^{-23} J and 9.8×10^{-22} J, with a clock amplitude factor of 2. [9] uses these technology parameters:

Size of Quantum dot: 5 nm

Size of 1 cell: 18×18 nm

Distance of cell: 2 nm

Layer separation : 11.5 nm

Radius of effect: 80 nm

Relative permittivity: 12.9

Operating temperature: 1 K

Clock signal shape: Gaussian

The design in [9] has two stages. In the first stage, the circuit is synthesized at the logical level and validated with Logisim 2.7.1. In the second stage, the physical layout is created in QCADesigner and verified with post-layout simulation. The energy dissipation is then evaluated in QCADesigner-E [9].

DESIGN OF LOGICAL UNIT OPERATIONS

1. AND GATE

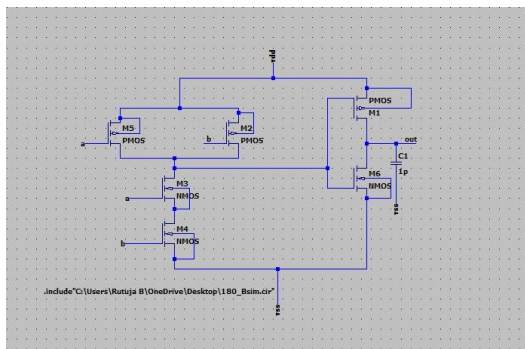


Fig. 8: AND gate in CMOS

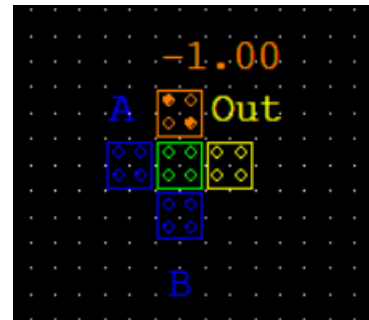


Fig.9: AND gate in QCA

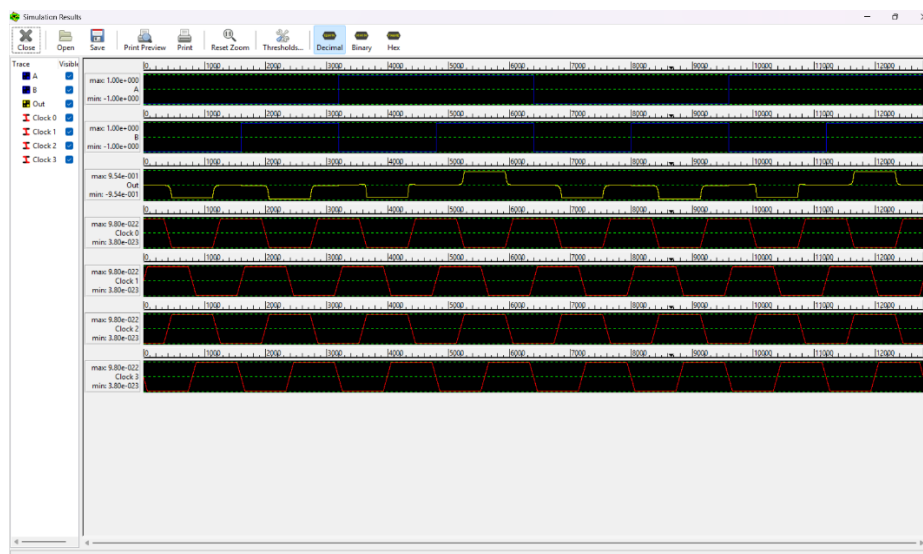


Fig. 10: Output of AND gate in QCA

A CMOS AND gate is built in two stages. A NAND stage, with its two PMOS transistors in parallel and two NMOS transistors in series, produces $(A \cdot B)'$. A single PMOS/NMOS inverter then flips this to give $Y = A \cdot B$. LTspice transient simulation shows a full-swing output that goes high only when both inputs are high, and it also shows the switching delay.

A QCA AND gate needs no dedicated structure. Tying one input of a three-input majority gate to logic "0" gives $\text{AND}(A, B) = M(A, B, 0) = A \cdot B$ [4]. The layout is a pair of input lines, one fixed logic-0 cell, and an output cell. QCADesigner shows the output cell polarizing to $P = +1$ (logic "1") only when A and B are both "1". A reversible

QCA AND gate has been reported to dissipate just 0.009 meV in total and 0.002 meV on average, which is below the Landauer limit [9].

2. OR GATE

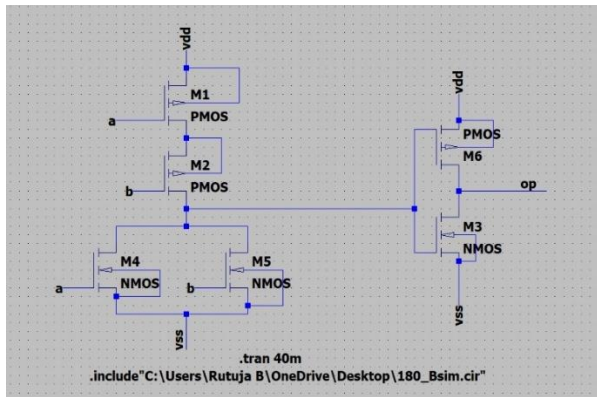


Fig. 11: OR gate in CMOS

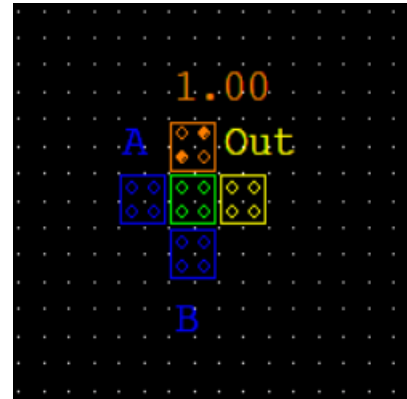


Fig. 12: OR gate in QCA

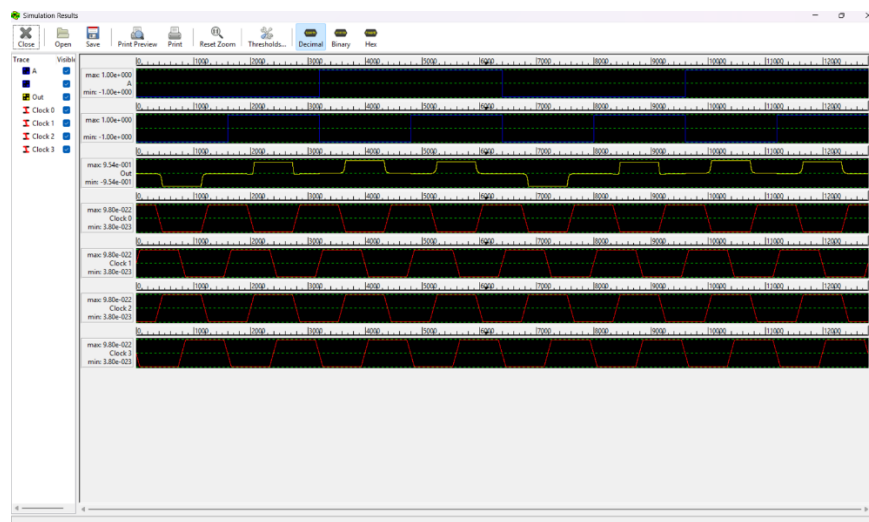


Fig. 13: output of OR gate in QCA

A CMOS OR gate is also built in two stages. A NOR stage, with two series PMOS transistors for pull-up and two parallel NMOS transistors for pull-down, produces $(A+B)'$. A PMOS/NMOS inverter then flips this to give $Y = A+B$. LTspice transient simulation shows the output going low only when both inputs are low, and it also shows the switching delay.

A QCA OR gate is again a three-input majority gate, this time with one input fixed to logic "1", so $OR(A, B) = M(A, B, 1) = A+B$. The layout has the two input lines, one fixed logic-1 cell, and an output cell. QCADesigner shows the output cell polarizing to $P = +1$ when either input or both inputs are "1", and to $P = -1$ only when both are "0". In the 16-operation reversible ALU of [9], OR is selected by control inputs $S0 = 0$, $S1 = 0$, $S2 = 1$, which gives OR at output 1 and NOR at output 2. Its reversible OR gate dissipates 0.009 meV in total and 0.002 meV on average, which is below the Landauer limit [9].

3.NOT GATE

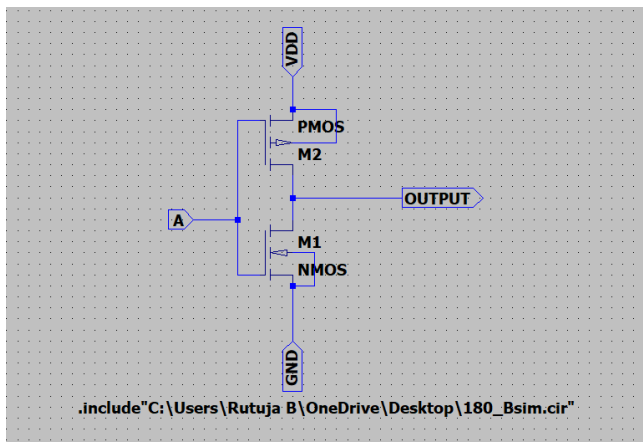


Fig. 14: NOT gate in CMOS

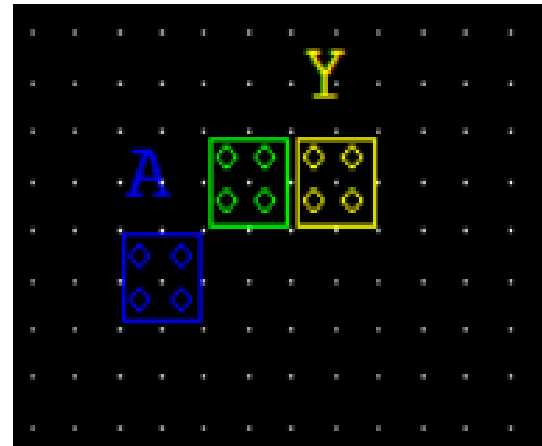


Fig. 15: NOT gate in QCA

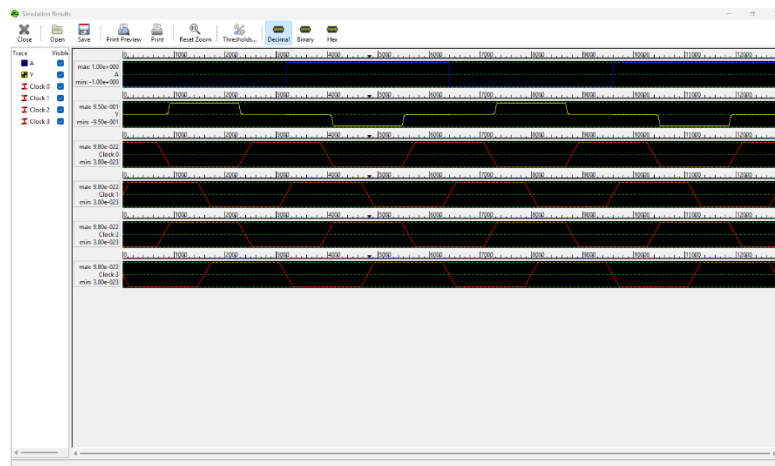


Fig. 16: Output of NOT gate in QCA

A CMOS NOT gate (inverter) is the simplest CMOS gate. It uses just one PMOS and one NMOS transistor, so it needs two transistors. The PMOS connects to VDD, the NMOS to ground, and the input drives both gates. The output is taken from their common drain, giving $Y = A'$. LTspice simulation shows correct inversion with full voltage swing, fast switching, and low static power. A QCA NOT gate is one of the basic gates of the technology, along with the majority gate and the XOR gate, and it is available in two layouts: the corner inverter and the robust inverter. In both, the output is the inverted polarization of the input [4]. The inverter works through electrostatic interaction. The output cell sits diagonally to the input cell, so it settles in the opposite polarization, and the gate then gives $Y = A'$. QCADesigner shows the complementary output for both input levels. In the 16-operation reversible ALU of [9], NOT is selected by control inputs $S0 = 0$, $S1 = 0$...

The control inputs for NOT in [9] are $S0 = 0$, $S1 = 1$, $S2 = 0$, where output 1 is the buffer and output 2 is the inverter [9]. Inverters also appear inside larger circuits. For example, the reversible XOR in [9] uses two inverters and three majority gates, and a 1-bit comparator in [4] uses three inverters [4].

4. NAND GATE

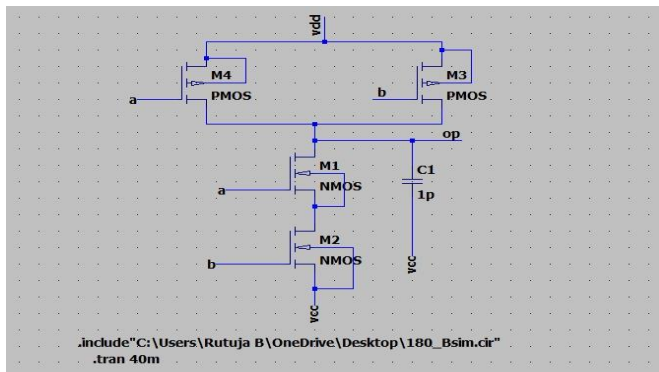


Fig. 17: NAND gate in CMOS



Fig.18: NAND gate in QCA

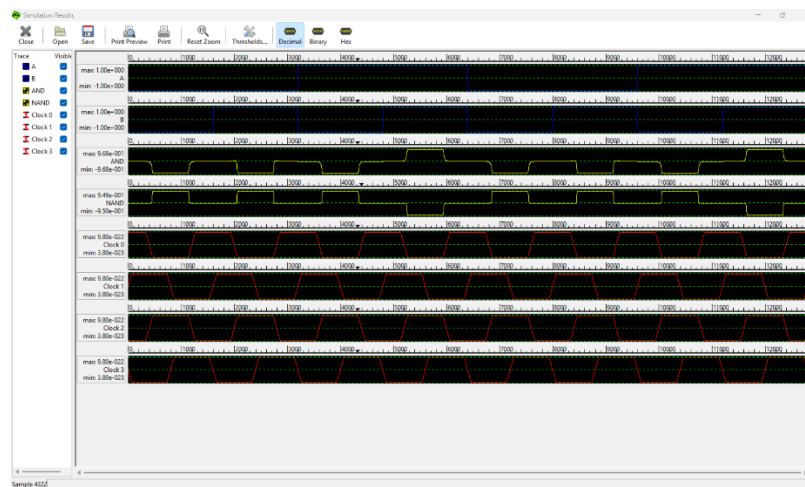


Fig. 19: output of NAND gate in QCA

A CMOS NAND gate is the basic CMOS logic gate and needs only four transistors. Two PMOS transistors in parallel form the pull-up network, and two NMOS transistors in series form the pull-down network. The output is $Y = (A \cdot B)'$, so it goes low only when both inputs are high. LTspice simulation confirms this and shows the rise/fall times and the propagation delay.

In QCA, a NAND gate is built in two steps. A three-input majority gate with one input fixed to logic "0" gives the AND function, and a QCA inverter then inverts it, so $\text{NAND}(A, B) = (A \cdot B)'$ [4]. The layout has input lines A and B, fixed logic-0 cells, an AND region, and an inverter section. QCADesigner shows the output cell staying at logic "1" unless both inputs are "1". NAND also appears in the 16-operation reversible ALU of [9]. It is Output 2, the inversion of AND, selected by control inputs $S0 = 0$, $S1 = 0$, $S2 = 0$ [9].

5. NOR GATE

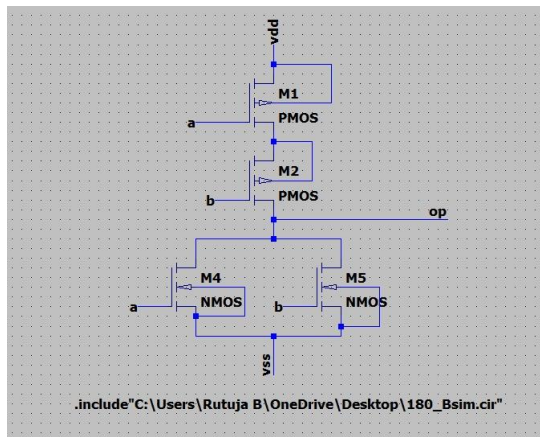


Fig. 20: NOR gate in CMOS

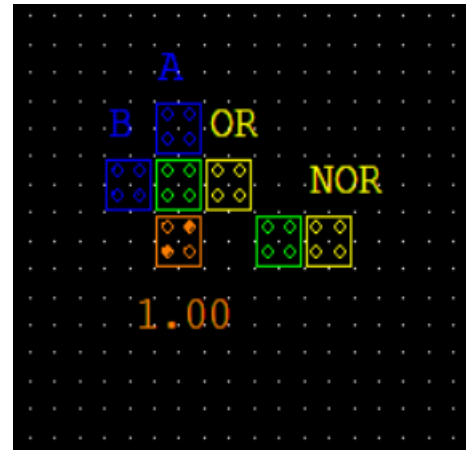


Fig.21: NOR gate in QCA

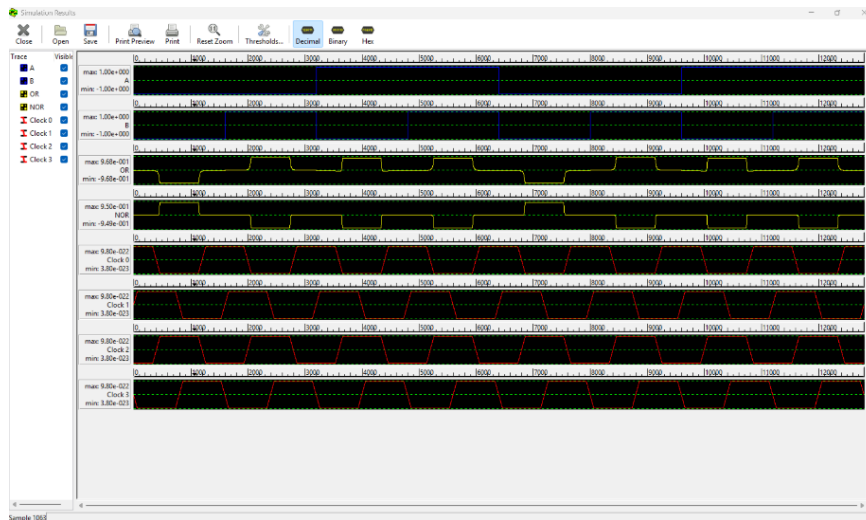


Fig. 22:Output of NOR gate in QCA

A CMOS NOR gate needs four transistors. Two PMOS transistors in series form the pull-up network, and two NMOS transistors in parallel form the pull-down network. The output is $Y = (A+B)'$, so it is high only when both inputs are low. LTspice simulation shows the correct logic levels and the transition times.

In QCA, a NOR gate is built in two steps. A three-input majority gate with one input fixed to logic "1" gives the OR function, and a QCA inverter then inverts it, so $NOR(A, B) = (A+B)'$ [4]. The layout has input lines A and B, a fixed logic-1 cell, an OR region, and an inverter section. QCADesigner shows the output cell polarizing to logic "1" only when both inputs are "0", and the four-phase clock keeps the propagation synchronized. NOR also appears in the 16-operation reversible ALU of [9]. It is Output 2, the inversion of OR, selected by control inputs $S0 = 0$, $S1 = 0$, $S2 = 1$ [9].

6. XOR GATE

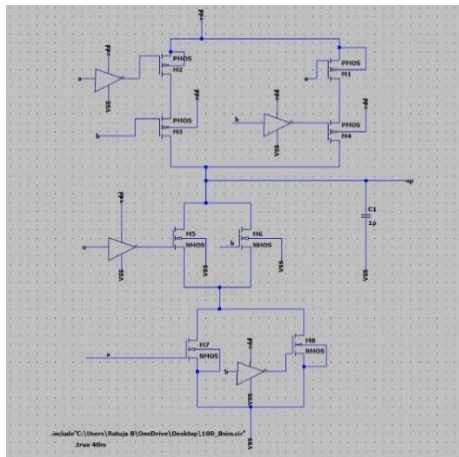


Fig.23: XOR gate in CMOS

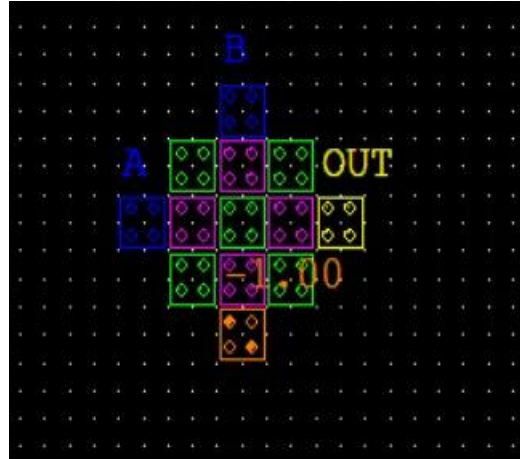


Fig.24: XOR gate in QCA

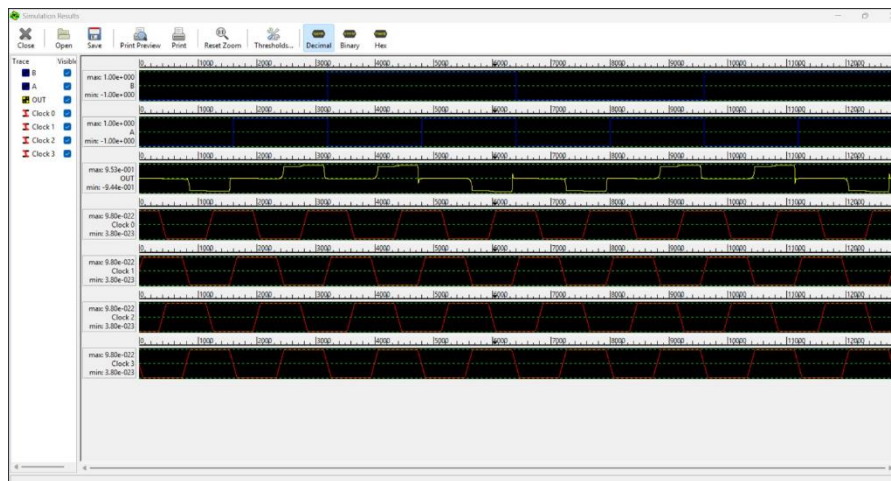


Fig. 25: Output of XOR gate in QCA

An XOR gate gives a high output only when its two inputs are different, $Y = A \oplus B = A \cdot B' + A' \cdot B$. In CMOS, it has no single-stage form like NAND or NOR. It is built from a combination of NAND/NOR gates and inverters, so it needs more transistors than the basic gates.

In QCA, XOR is one of the three fundamental gates, along with the majority gate and the inverter [4]. It is used inside larger circuits, for example, in the 1-bit comparator of [4]. A reversible QCA XOR can be built from two inverters and three reversible majority gates. Its output is $(A+B) \cdot (A'+B')$ [9]. The layout needs 101 QCA cells and $0.15 \mu\text{m}^2$ of area, with a delay of eight clock zones, which is two clock cycles [9]. Its energy dissipation is 0.054 meV in total and 0.014 meV on average, below the Landauer limit [9]. In the reversible ALU, XOR is Output 1, and XNOR is Output 2 at control inputs $S_0 = 0$, $S_1 = 1$, $S_2 = 1$ [9].

DESIGN OF ARITHMETIC UNIT OPERATIONS

1. FULL ADDER

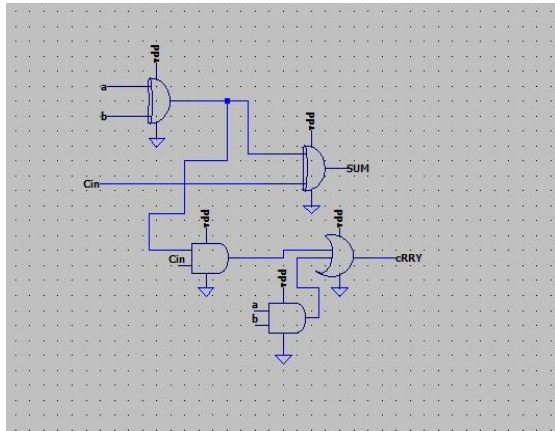


Fig. 26: FULL ADDER in CMOS.



Fig. 27: FULL ADDER in QCA

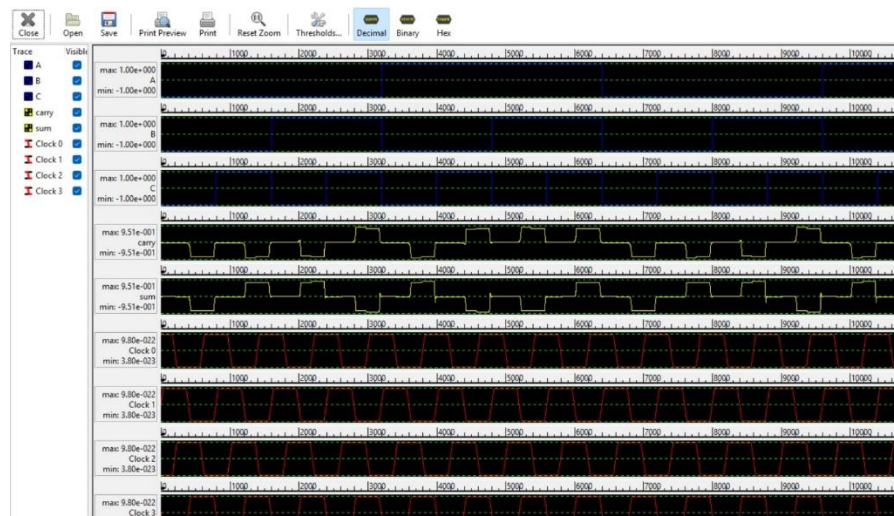


Fig. 28: Output of FULL ADDER in QCA

A full adder adds three one-bit inputs, A, B, and carry-in Cin, and produces a Sum and a Carry-out:

$$\text{Sum} = A \oplus B \oplus \text{Cin}, \text{Cout} = AB + BC\text{in} + AC\text{in}$$

Because Cout is a majority function of the three inputs, a single three-input majority gate gives it directly in QCA: $\text{Cout} = M(A, B, \text{Cin})$.

In QCA, full adders are usually built from majority gates, inverters, and XOR gates. A full adder and subtractor have been designed around a unique three-input XOR gate with redundant cells. The design is a single layer without cross-wiring, and its energy dissipation was analyzed [3]. A fault-tolerant three-input majority gate has also been used to build a full adder/subtractor [10]. The half-adder is the building block of the full adder. A reversible QCA half-adder uses four reversible majority gates and two inverters. The Sum output is $\text{Sum} = A \cdot B' + A' \cdot B$, and the Carry output is $\text{Carry} = A \cdot B$. This design requires 156 cells and covers $0.27 \mu\text{m}^2$ of area. Its delay is 12 clock zones, equal to three clock cycles [9]. Its energy dissipation is 0.099 meV in total and 0.025 meV on average, below the Landauer limit [9].

2. FULL SUBTRACTOR

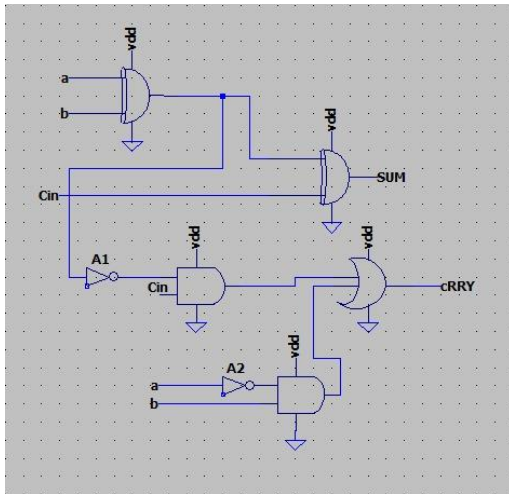


Fig. 29: FULL SUBTRACTOR in CMOS

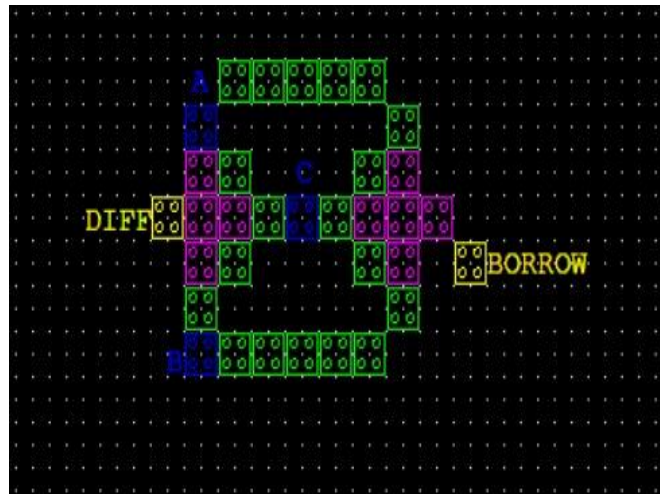


Fig. 30: FULL SUBTRACTOR in QCA

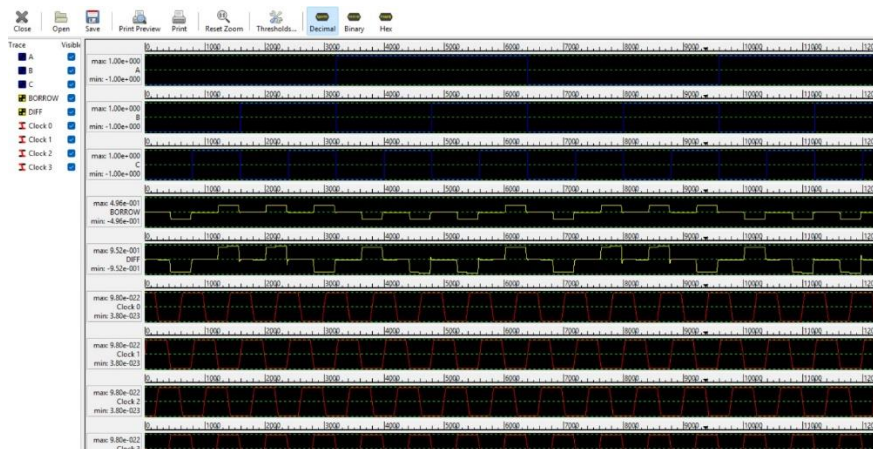


Fig. 31: Output of FULL SUBTRACTOR in QCA

A full subtractor subtracts two one-bit inputs, $A - B$, along with a borrow-in B_{in} , and produces a Difference and a Borrow-out:

$$\text{Diff} = A \oplus B \oplus B_{in}, \text{Borrow} = A'B + A'B_{in} + B \cdot B_{in}$$

The Difference has the same form as the Sum of a full adder. The Borrow is the full adder's Carry equation with input A inverted, so $\text{Borrow} = M(A', B, B_{in})$.

In CMOS, a full subtractor therefore has a design similar to a full adder. The same structure is used, and the only addition is a NOT gate (a PMOS/NMOS inverter pair) on input A, which feeds the Borrow logic.

In QCA, the same idea applies, since the borrow needs one inverter on A in front of a three-input majority gate. A QCA full adder and subtractor have been designed together using a unique three-input XOR gate with redundant cells. The design is a single layer without cross-wiring, and its energy dissipation was analyzed [3]. A fault-tolerant three-input majority gate has also been used to build a full adder/subtractor [10]. The half-subtractor forms the basis

for building a full subtractor. A reversible QCA half-subtractor requires two inverters and three reversible majority gates, with $\text{Diff} = A' \cdot B + A \cdot B'$ and $\text{Borrow} = A' \cdot B$. It uses 116 cells and $0.15 \mu\text{m}^2$ of area, with a delay of eight clock zones, which is two clock cycles [9]. Its energy dissipation is 0.063 meV in total and 0.016 meV on average, below the Landauer limit [9].

Table 1 Comparison results of the full adder

In	Cells	Area (	Latency	Layer-type
[10]	99	0.09	1	Single-layer
[11]	80	0.06	2	Multi-layer
[13]	50	0.04	1	Coplanar
[5]	44	0.06	1.25	Coplanar
Proposed	28	0.03	2	Coplanar

SIMULATION AND PERFORMANCE ANALYSIS OF FULL ADDERS

We evaluated the proposed full adder in QCA Designer 2.0.3 to verify its functionality. Simulation waveforms confirm that the circuit generates the expected Sum and Carry outputs for all input combinations. Performance metrics were compared with previously reported full adder designs, as summarized in Table 1.

The proposed design uses only 28 cells and occupies $0.03 \mu\text{m}^2$ in a coplanar layout. Compared to designs in [10], [11], [13], and [5], it reduces cell count by 71.72%, 65%, 44%, and 36.36%, respectively, and area by 66.67%, 50%, 25%, and 50%. These results demonstrate that the proposed full adder achieves a more compact layout with fewer QCA cells and less area than previous designs.

The proposed full adder operates in two clock cycles, matching the latency of the design in [11]. In comparison, designs in [10], [13], and [5] require 1, 1, and 1.25 clock cycles, respectively. This reflects a trade-off between area efficiency and latency. While the latency is slightly higher than some existing designs, the substantial reduction in cell count and area makes the proposed full adder well-suited for compact, area-efficient QCA implementations.

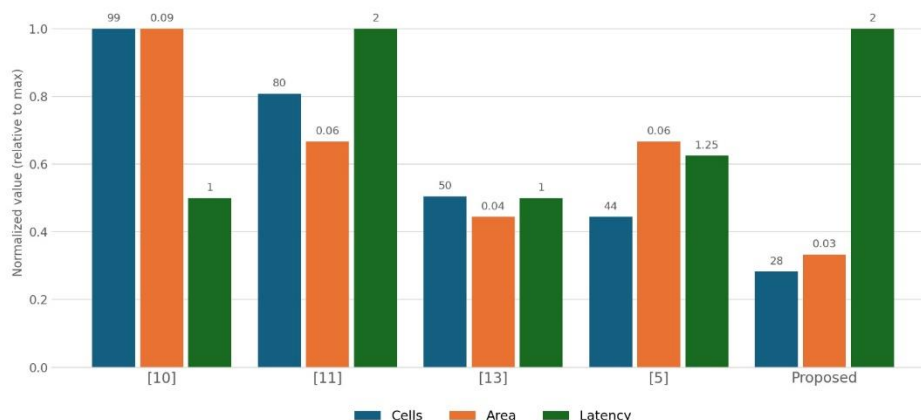


Fig.32: The comparison of different full adders

CONTROL UNIT IN QCA

1. 2x1 MUX

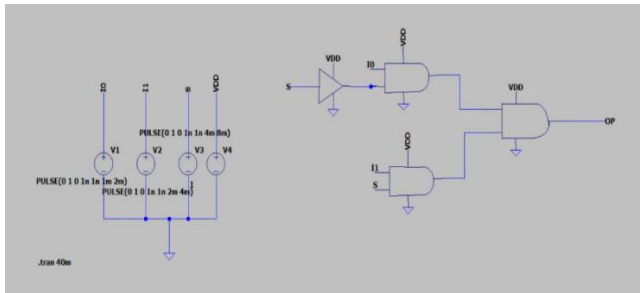


Fig.33: 2:1 MUX in CMOS

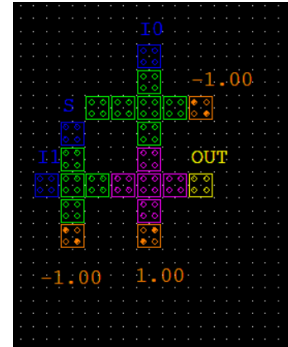


Fig.34: 2:1 MUX in QCA

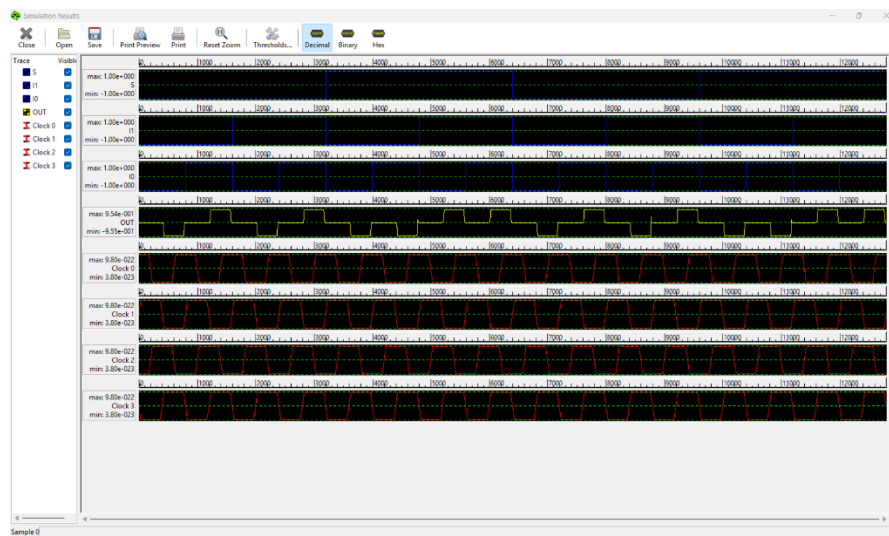


Fig.35: Output of 2:1 MUX in QCA

A 2:1 multiplexer picks one of two data inputs, A or B, using a select line S, so $Y = S' \cdot A + S \cdot B$. If $S = 0$, the output follows A, and if $S = 1$, it follows B.

In CMOS, it can be built from basic gates (two AND stages, an OR stage, and an inverter on the select line). It can also be built with transmission gates, where S and its inverse control which input reaches the output.

In QCA, the multiplexer comes directly from the AND/OR behavior of the majority gate. Each AND term uses a majority gate with one input fixed to "0" [4], and the final OR joins the two terms. This gives $Y = M(M(A, S', 0), M(B, S, 0), 1)$. The total is three majority gates plus one inverter for S' . This count matches the reversible 2:1 multiplexer described in [9], which uses three reversible majority gates and one inverter [9]. A fault-tolerant three-input majority gate can also build a 2:1 multiplexer [10].

In summary, a CMOS multiplexer consists of a network of transistors or transmission gates, whereas a QCA multiplexer is constructed using majority gates and a single inverter.

2. 4:1 MUX

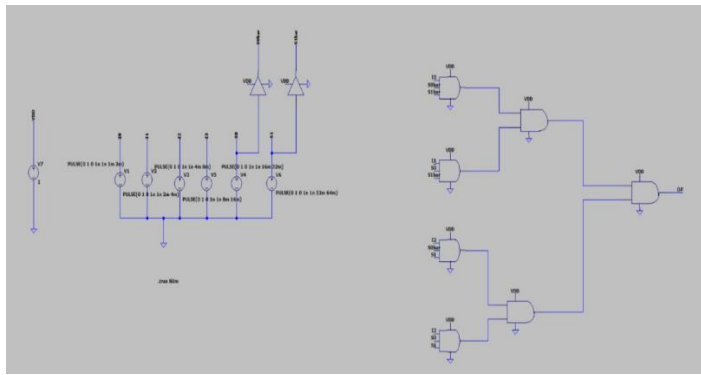


Fig.36: 4:1 MUX in CMOS

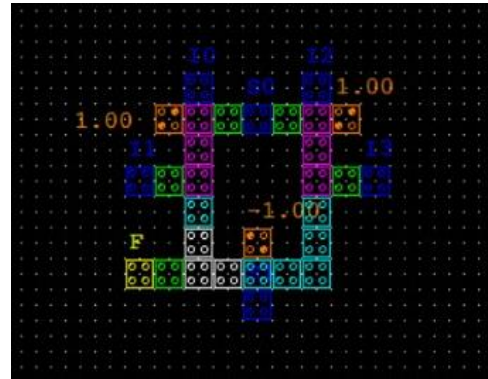


Fig.37: 4:1 MUX in QCA

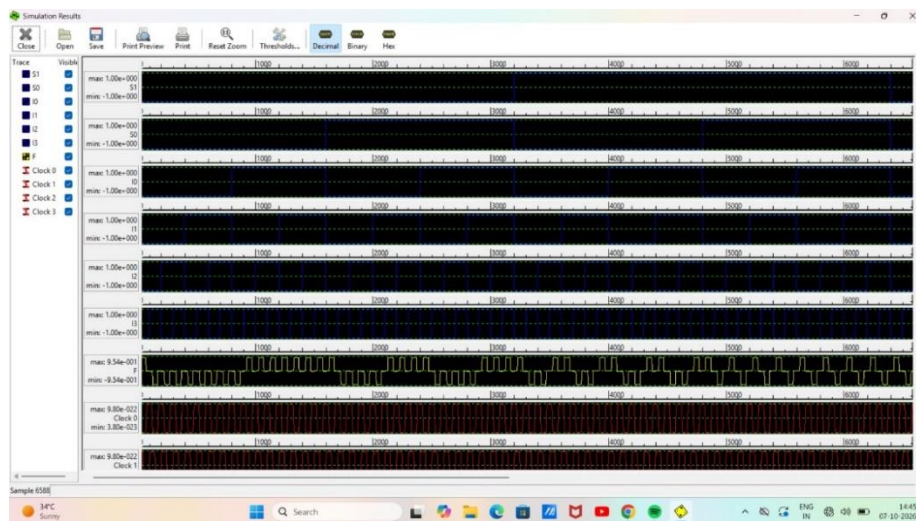


Fig.38: Output of 4:1 MUX in QCA

A 4:1 multiplexer picks one of four data inputs, D0 to D3, using two select lines S1 and S0:

$$Y = S1'S0'D0 + S1'S0'D1 + S1S0'D2 + S1S0'D3$$

In CMOS the circuit can be constructed using basic gates or by using transmission gates; a typical approach is to use a tree consisting of three 2:1 multiplexers. Two of these are controlled by S0 and select between D0/D1 and D2/D3, while a third one, controlled by S1, chooses between the two outputs thus obtained.

QCA employs the same concept based on the tree structure; since each 2:1 stage requires three majority gates and one inverter, the complete 4:1 multiplexer needs three stages. The reversible 4:1 multiplexer described in [9] makes use of nine majority gates and three inverters, which is precisely three times the number required for the 2:1 stage [9]. As in the case of the 2:1 multiplexer, each AND term is a majority gate with one of its inputs held at "0" [4].

To sum up, in both CMOS and QCA a 4:1 multiplexer consists of three 2:1 multiplexers connected in a tree structure.

PROPOSED ARITHMETIC & LOGICAL UNIT

1.ARITHMETIC UNIT

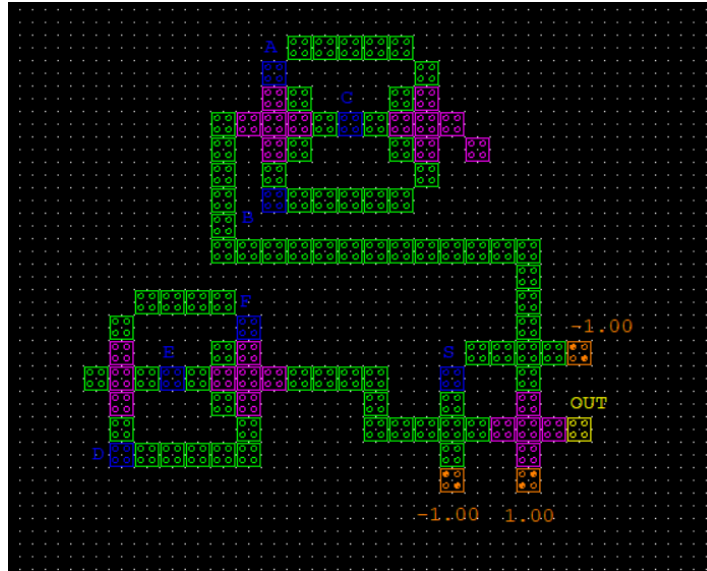


Fig.39: Proposed Arithmetic Unit

The proposed Arithmetic unit consists of 109 cells and has an area of 169694.00 nm^2 . It contains 2 operations addition & subtraction.

2.LOGICAL UNIT

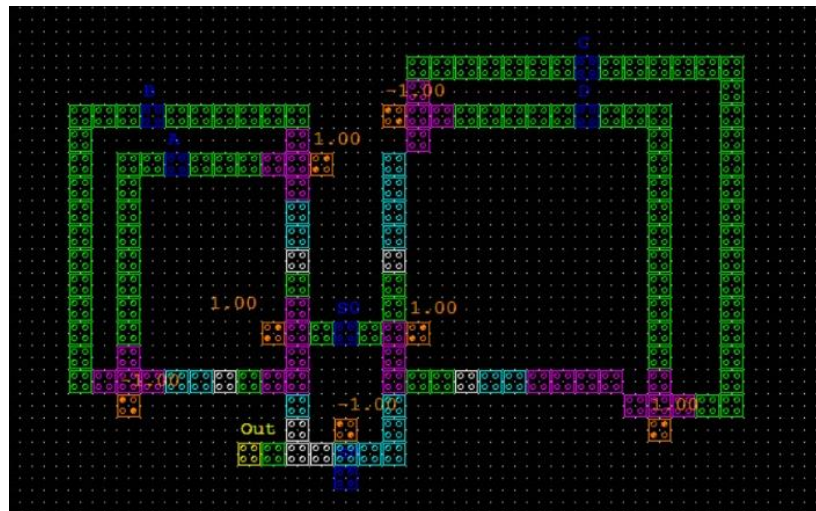


Fig. 40: Proposed Logical Unit

The proposed logical unit consists of 152 cells and has an area of 212598.00 nm^2 . It contains 4 operations AND,NOR,NAND, & NOR

COMPARATIVE REVIEW OF CMOS AND QCA

Table 2: Comparison of CMOS and QCA

PARAMETER	CMOS	QCA
Information carrier	Voltage level, with current flowing through transistors	Electron position (cell polarization)
Basic element	N-type and P-type MOSFET pair	Cell with four quantum dots and two electrons
Logical building blocks	Complementary pull-up and pull-down networks	Majority gate and inverter
Signal transfer	Wires carrying current	Electrostatic (Coulomb) coupling between neighbouring cells
Power consumption	Higher, with leakage and switching losses growing as devices shrink	Very low in theory, since no current flows
Heat dissipation	A major concern at high densities	Much lower
Speed	Limited by transistor switching and wire delay	Potentially very high, but limited in practice by clock zones
Device density	Limited by lithography and short-channel effects	Much higher potential density
Clocking	Conventional clock signal	Four-phase clock that controls data flow zone by zone
Wire crossing	Metal layers	Coplanar (rotated cells) or multilayer crossover
Fabrication	Mature, standardized, mass-produced	Still experimental, needs nanoscale precision
Reliability	High, with well-understood defects	Sensitive to missing, extra or misplaced cells
Operating temperature	Room temperature	Often cryogenic for semiconductor or metal-dot versions; molecular QCA is proposed for room temperature
Design tools	Mature (SPICE, Cadence, Verilog flows)	Limited (QCADesigner, QCADesigner-E)
Commercial status	Dominant in industry	Research stage

CONCLUSION :

This study focused on designing and simulating a 1-bit Arithmetic Logic Unit (ALU) using Quantum-dot Cellular Automata (QCA) technology. The work included building basic logic gates, a full adder, a full subtractor, a 2×1 multiplexer, and a 4×1 multiplexer to see whether QCA cells can build digital circuits.

A Logic Unit was created to perform four logical operations, AND, OR, NAND, and NOR, with a 4×1 multiplexer selecting the desired operation. Additionally, an Arithmetic Unit (AU) was developed to perform addition and subtraction using a full adder and a full subtractor, with a 2×1 multiplexer selecting between the two operations. The design was verified using QCA design software to ensure it worked correctly. We measured the number of cells and the area each circuit uses to evaluate the design requirements. These results show how circuit complexity relates to the number of QCA cells and the area each design uses.

This study shows that QCA technology can build simple digital circuits with separate arithmetic and logic units.

The simulation results and area measurements provide a foundation for further research on designing more complex QCA-based ALUs, with an emphasis on reducing the number of cells and area without compromising correct operation.

REFERENCES

1. N. Safoev and J. C. Jeon, "Cell Interaction Based QCA Multiplexer for Complex Circuit Design," *Advanced Science Letters*, Vol. 23, No. 10, pp. 10097–10101, 2017, DOI: 10.1166/asl.2017.10396.
2. K. S. Zaka and N. Kaulgud, "Quantum Dot Cellular Automata (QCA) Design for the Realization of Basic Logic Gates," *2017 International Conference on Electrical, Electronics, Communication, Computer and Optimization Techniques (ICEECOT)*, 2017.
3. M. Abdullah-Al-Shafi and A. N. Bahar, "An Architecture of 2-Dimensional 4-Dot 2-Electron QCA Full Adder and Subtractor with Energy Dissipation Study," *Active and Passive Electronic Components*, 2018.
4. A. Shiri, A. Rezai and H. Mahmoodian, "Design of Efficient Coplanar 1-Bit Comparator Circuit in QCA Technology," *Facta Universitatis, Series: Electronics and Energetics*, Vol. 32, No. 1, pp. 119–128, 2019, DOI: 10.2298/FUEE1901119S.
5. S. Zoka and M. Gholami, "A Novel Efficient Full Adder–Subtractor in QCA Nanotechnology," *International Nano Letters*, 2019.
6. A. Singh, A. S. Dubish, S. Shreyasi, A. Naik and V. Nithya, "Design and Simulation of Arithmetic Logic Unit Using Quantum Dot Cellular Automata," *International Journal of Electrical Engineering and Technology*, Vol. 3, No. 11, pp. 173–180, 2020.
7. B. Safaiezadeh, E. Mahdipour, M. Haghparast, S. Sayedsalehi and M. Hosseinzadeh, "Novel Design and Simulation of Reversible ALU in Quantum Dot Cellular Automata," *The Journal of Supercomputing*, Vol. 78, No. 1, 2022, DOI: 10.1007/s11227-021-03860-y.
8. R. N. and V. M. B., "Design of Efficient ALU Blocks in Quantum Dot Cellular Automata (QCA)," *Global Transitions Proceedings*, 2022.
9. M. Alharbi, G. Edwards and R. Stocker, "Reversible Quantum-Dot Cellular Automata-Based Arithmetic Logic Unit," *Nanomaterials*, Vol. 13, No. 17, 2445, 2023, DOI: 10.3390/nano13172445.
10. S. Riki and F. S. Hassani, "Designing a One-Bit Coplanar QCA ALU Using a Novel Robust Area-Efficient Three-Input Majority Gate Design," *The Journal of Supercomputing*, Vol. 79, No. 16, 2023, DOI: 10.1007/s11227-023-05280-6. (Preprint version: Research Square, 2022.)
11. M. Ebrahimi, M. Gholami, H. Adrang and R. Yosefi, "A Novel Low-Latency ALU in the One-Dimensional Clock Scheme in QCA Nanotechnology," 2024.
12. R. Faraji and A. Rezai, "Design of a Multilayer Reversible ALU in QCA Technology," *The Journal of Supercomputing*, Vol. 80, No. 12, 2024, DOI: 10.1007/s11227-024-06102-z.
13. M. Zohaib, N. J. Navimipour, M. T. Aydemir and S.-S. Ahmadpour, "A Nano-Scale Design of Arithmetic and Logic Unit for Energy-Efficient Signal Processing Devices Based on a Quantum-Based Technology," *Cluster Computing*, Vol. 28, Article 340, 2025, DOI: 10.1007/s10586-024-05073-3.