Implementation of Code Converters in QCAD

Pallavi A[1] N. Moorthy Muthukrishnan[2]
[1]Student [2]Professor
[1]Department of Electronics & Communication Engineering 2
[1,2]GNITS, Andhra Pradesh, India

Abstract— Quantum-dot Cellular Automata (QCA) is an leading technology for development of digital logic circuits in the field of nanotechnology, and is an potential alternative for designing high performance computers over existing transistor-based technology. The basic logic in QCA represents the binary state by polarization of electrons in quantum cells unlike the CMOS devices that represent logic levels using voltages. Clocking is used in QCA circuit to synchronize and control the information flow and to provide the power to run the circuit. One of the main differences between circuit design in QCA and in conventional CMOS technologies and devices is that QCA based circuit has no control over the clocks that is information is transmitted through each cell and not retained. Each cell erases its own state every cycle of the clock. Quantum cell is basic building block of QCA. These cells are used to design majority gate, using which all classical logic gates can be implemented. QCA Designer (QCAD) is software used for design and simulation of QCA-based circuits. In this paper, 3-bit binary-to-gray and 3-bit gray-to-binary code converters have been implemented using QCAD software.

Key words: Nano electronics, QCA cell, Majority gate, Code converters, QCA Designer

I. INTRODUCTION

Nanotechnology is an emerging area of interest which offers alternative design technologies like carbon nano-tubes, quantum dot structures, molecular devices and micro-fluidic biochips. As the CMOS technologies approach their fundamental physical limits, there has been extensive research in the recent years in development of nanotechnology for future generation ICs. Shrinking transistor size down to submicron level, results in the degraded performance of the circuits. Physical limits such as quantum effects and nondeterministic behavior of small currents and technological limits like power dissipation and design complexity may hinder the further progress of microelectronics using conventional circuit scaling [6]. Therefore researchers have proposed an approach to computing with quantum dots, the Quantum Cellular Automata (QCA). Quantum-Dot Cellular Automata (QCA) is an emerging paradigm which allows operating frequencies in range of THz [8], and device integration densities about 900 times more than the current end of CMOS scaling limits, which is not possible in current CMOS technologies. It has been predicted as one of the future nanotechnologies in Semiconductor Industries Association's International Technology Roadmap for Semiconductors (ITRS) [2].

Each quantum cell contains two electrons which interact coulombically with neighboring cells. The charge distribution in each cell tends to align along one of two perpendicular axes, which allows the encoding of binary information using the state of the cell. The state of each cell is affected in a very nonlinear way by the states of its neighbors. A line of these cells can be used to transmit binary information.

The fundamental QCA logic primitives are the wire, inverter, and the three-input majority gate [9]. As the Majority Gate (MG) is not functionally complete, the majority gate with inverter, called Majority Inverter (MI), and is then used for realizing various QCA designs. Extensive work is going on QCA for circuit design due to low power consumption and regularity in the circuit. Clocking is used in QCA circuit to synchronize and control the information flow and to provide the power to run the circuit. The paper is organized as follows: In section I and II the background of QCA technology is presented. In section III QCA Designer is described briefly. In section IV QCA implementation of Code converter circuits using QCA Designer is shown. In section V conclusions are presented.

II. QUANTUM-DOT CELLULAR AUTOMATA

QCA cells are of two types. A type-1 QCA cell as shown in Fig. 1(a) consists of four dots at the corners with two excess electrons (dark dots in a cell) that can tunnel between the dots. A type-2 QCA cell as shown in Fig. 1(b) consists of four dots at the midpoint of the sides of cells. Due to Columb repulsion, the two excess electrons always occupy diagonally opposite dots. In both types of QCA cells, there are two energetically equivalent polarization states designated as +1 and −1. A basic QCA logic element is a majority gate and shown in Fig. 1(c). QCA cells A, B, and C is input cells, and F is the output cell that is polarized according to the polarization of the majority of the input cells. In this example, since two (out of three) input QCA cells are polarized to −1, the output cell is also polarized to −1. QCA cells can also be used to construct wires. A QCA wire constructed using type-1 cells is shown in Fig. 1(d). When an input is applied to the left input cell, the binary information propagates from the left to the right. When all cells in a wire settle down to their ground states, they have the same polarization. A QCA wire constructed using type-2 QCA cells is shown in Fig. 1(e). When all cells in such a wire settle down, each cell has the opposite polarization than its neighbors in the wire [4].

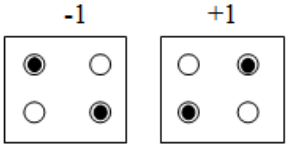


Fig. 1: (a) Two type-1 (90◦) QCA cells with opposite polarizations.

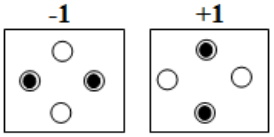


Fig. 1: (b) Two type-2 (45◦) QCA cells with opposite polarizations.

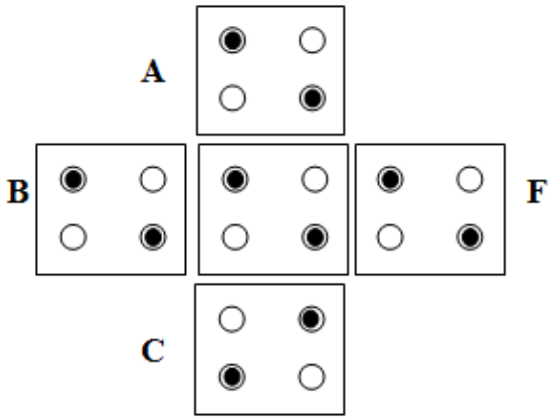


Fig. 1: (c) Majority gate.

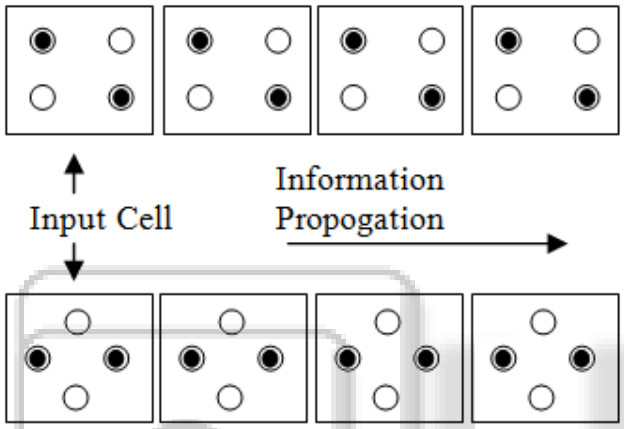


Fig. 1: (d) QCA wire using type-1 cells. (e) QCA wire using type-2 cells.

Timing/synchronization in QCA is accomplished by the cascading the clock of four periodic and distinct phases as shown in Fig 2 [10]. During the first (switch) phase, the tunneling barrier between two dots of a QCA cell starts to rise. This is the phase during which computation takes place. The second (hold) phase is reached when the tunneling barriers are high enough to prevent electrons from tunneling. In the third (release) phase, barrier falls from high to low. The final phase (relax) ensures there is no inter-dot barrier and the cell remains unpolarized. Each cell has to pass either of these clocking zones. The polarization of a QCA cell switches between the two polarization states when the two excess electrons tunnel between neighboring dots within the cell. When the inter-dot barrier is raised, a QCA cell will maintain its current polarization and will not react to the changes in the polarization of its neighbors.

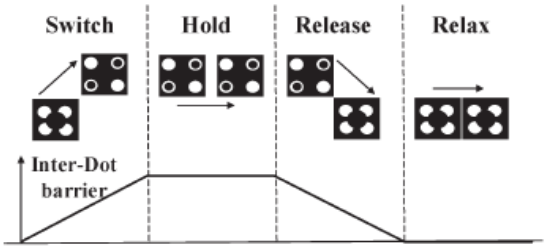


Fig. 2: QCA Clock.

Fig. 2 shows that the inter-dot barrier of a QCA cell can be controlled (similar to a clock-based control in CMOS technology) [1]. Let us call this modulation of the inter-dot barrier in a QCA cell a clock. A clock cycle in QCA logic can in turn be divided into four phases, namely: 1) switch; 2) hold; 3) release; and 4) relax. During the switch phase, the inter-dot barrier is gradually raised, and the QCA cell settles down to one of the two ground polarization states as influenced by its neighbors. During the hold phase, the inter-dot barrier is held high, suppressing electron tunneling and maintaining the current ground polarization state of the QCA cell. During the release and relax phases, the inter-dot barriers are lowered, and the excess electrons gain mobility. In these two phases, a QCA cell remains unpolarized. Overall, the polarization of a QCA cell is determined when it is in its switch phase by the polarizations of its neighbors that are in switch and hold phases.

The unpolarized neighbors in release and relax phases have no effect on determining the state of the QCA cell. In general, a clocked QCA design uses four clocks, namely: 1) Clock 1; 2) Clock 2; 3) Clock 3; and 4) Clock 4, as shown in Fig 2. Each clock is 90◦ out-of-phase from its previous clock. An array of QCA cells with a clock sequence (Clock 1 → Clock 2→ Clock 3 → Clock 4) functions as a D-latch [1]**.** QCA cells controlled by a clock get latched and stay latched until the cells controlled by the next clock switch to the hold phase. This is called the self-latching or adiabatic pipelining property of QCA logic and QCA wire arrays [5]. A clock cycle in a clocked QCA design starts at the switch phase of Clock 1 and ends at the relax phase of Clock 1. During a clock cycle, all four clocks will traverse four phases, however, with different starting phases.

III. QCA DESIGNER

QCA Designer (QCAD) is a software used for design and simulation of QCA-based circuits. QCAD was initially developed at the ATIPS Laboratory, University of Calgary, Calgary, Canada. One of the main problems in implementing more accurate simulations is the lack of experimental data for QCA systems with a large number of cells. However, several small QCA systems have been developed as proof-of-concept experiments [3]. As a result, further research must be carried out in order to implement such devices which can develop a dialog between circuit designers and implementers. The current QCA technology does not specifically set the possible operating frequency and actual propagation delays, but it can be analyzed as an important parameter in future works. One of the most important design specifications is that other developers should be able to easily integrate their own utilities into it. QCAD is capable of simulating complex QCA circuits on most standard platforms.

As well, several simulation engines facilitate rapid and accurate simulation. But here the operation of QCA circuits is simulated using QCAD bistable vector simulation. These circuits can be used in constructing of nano-scale low power consumption information processing system and can stimulate higher digital applications in QCA.

IV. QCA IMPLEMENTATION

In addition to AND, OR, NOT, NAND and NOR gates, exclusive-OR (XOR) and exclusive-NOR (XNOR) gates are also used in the design of digital circuits. These XOR/XNOR gates are particularly useful in arithmetic operations as well as error-detection and correction circuits. These gates are generally found to be two-input gates. Multiple-input gates are complex to fabricate with hardware [7]. Exclusive or also known as Exclusive disjunction and symbolized by XOR, is a logical operation on two operands that results in a logical value of true if and only if one of the operands, but not both, has a value of true. This forms a fundamental logic gate in many operations.
The XOR performs the following logic operation:

$$A \oplus B=A'B+AB' \quad (1)$$

Exclusive-OR functions are very useful in systems using parity bits for error-detection. A parity bit is used for the purpose of detecting errors during transmission of binary information. A parity bit is an extra bit included with a binary message to make the total number of 1's in this message (including the parity bit) either even or odd. The sequence of bits can be transmitted and then checked at the receiving end for errors. An error can be detected only when if the checked parity does not correspond with the one transmitted. The circuit that generates the parity bit at the transmitter side is called a parity generator. The circuit that checks the parity at that receiver side is called parity checker.

Consider a 3-bit message to be transmitted together with an even parity bit. For even parity, whenever the message bits (X, Y, and Z) have an odd number of 1's, the parity bit P must be 1, otherwise P must be 0. Hence,

$$P=X \oplus [Y \oplus Z] \quad (2)$$

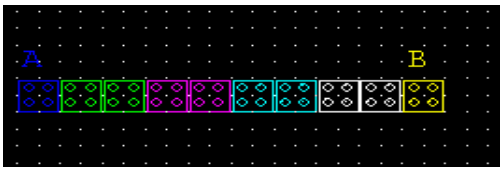
The 4 bits (X, Y, Z and P) are transmitted to their destination, where they are applied to parity-checker circuit in order to check for errors in the transmission. Since the information was transmitted with even parity, the received four bits must have an even number of ones. The parity checker generates an error signal (C=1), whenever the received four bits have an odd number of 1's. The logic implementation is done using:

$$C=X \oplus [Y \oplus [Z \oplus P]] \quad (3)$$

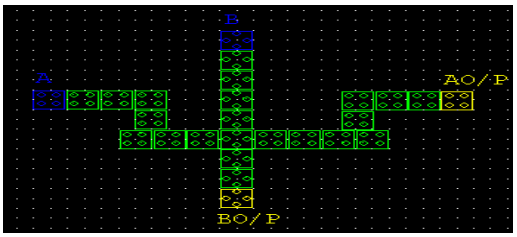
Parity generator can be implemented using parity checker circuit if input P is connected to logic-0 and the output is marked with P. This is because the value of Z id pass through the gate unchanged. Therefore same circuit can be used at both the ends of transmission and reception.

In digital systems code conversion is a widely used process for reasons such as enhancing security of data, reducing the complexity of arithmetic operations and thereby reducing the hardware required, dropping the level of switching activity leading to more speed of operation and power saving etc.,

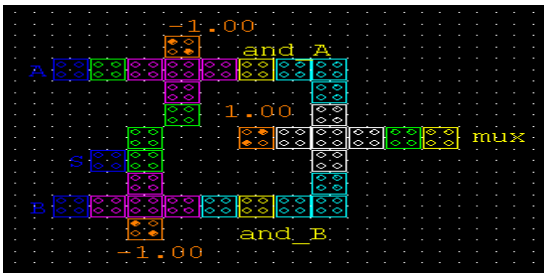
Following figures illustrates QCA implementation of various circuits using QCAD followed by the simulation results.



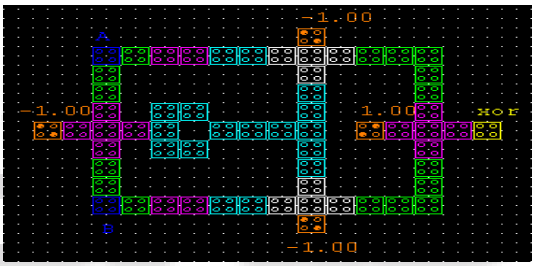
(a) Binary wire



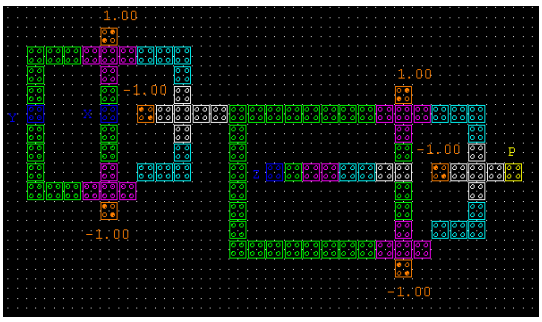
b) Coplanar wire



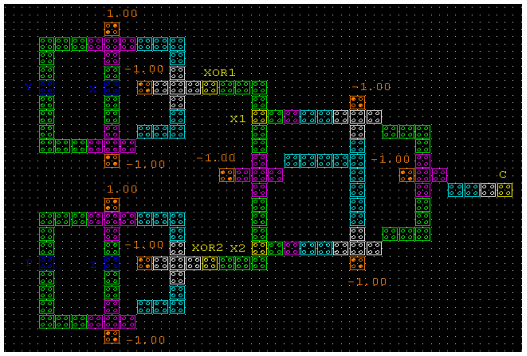
(c) 2x1 Multiplexer



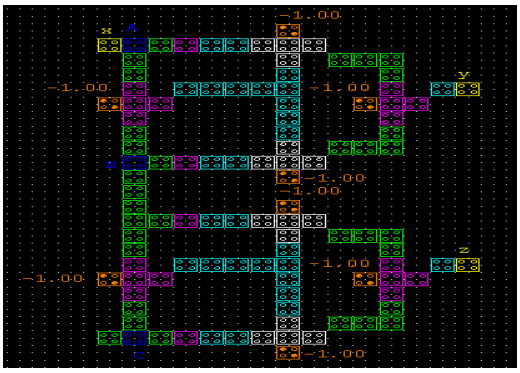
(d) XOR



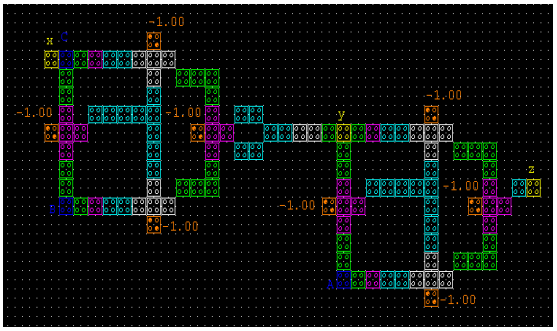
(e) Parity Generator



(f) Parity Checker



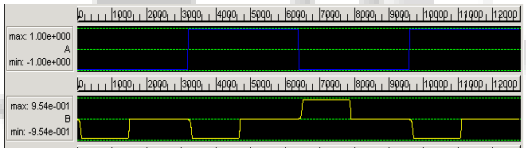
(g) 3-bit Binary-to-Gray Converter



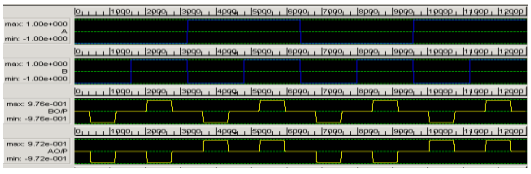
(h) 3-bit Gray-to-Binary Converter

Fig . 3: Implementations of various circuits

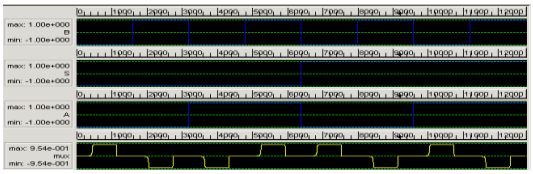
From above figures, two aspects such as number of crossovers and cell count are determined. In the simulation tool QCAD it mentions area of the circuit which is another important parameter. For simulating the circuit one can either use exhaustive or a particular vector table for verifying the concerned truth table.



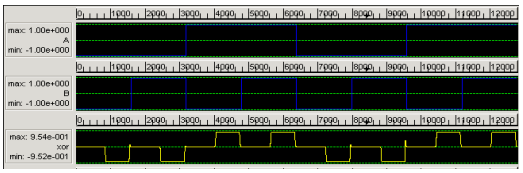
(a) Binary wire



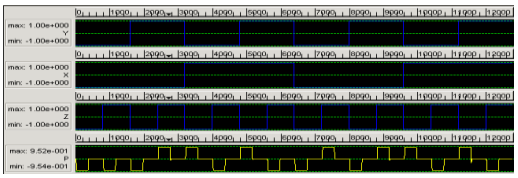
(b) Coplanar wire



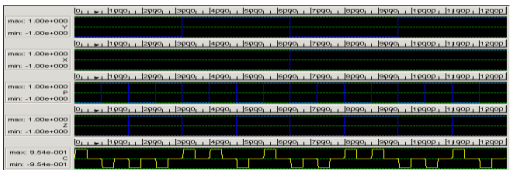
(c) 2x1 Multiplexer



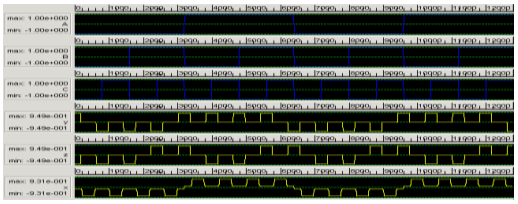
(d) XOR



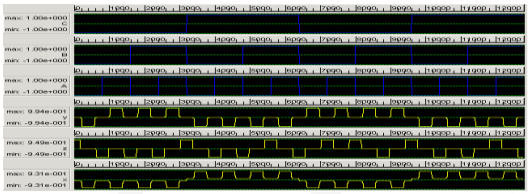
(e) Parity Generator



(f) Parity Checker



(g) 3-bit Binary-to-Gray Converter



(h) 3-bit Gray-to-Binary Converter

Fig. 4: Simulation results of various circuits

From the simulation results a parameter called latency can be determined and also truth table is verified. The latency can be same or different depending on complexity or the circuit used.

The features of basic logic circuits, complex circuits and code converters implemented using QCAD are listed in Tables 1 to 4.

Feature	AND	OR	NAND	NOR
Area (μm^2)	0.01	0.01	0.01	0.01
Cell count	10	10	7	7
No. of crossovers	0	0	0	0
Latency	0.5	0.5	0.5	0.5

Table 1: Features of various basic logic circuits.

Feature	MG	XOR	XNOR
Area (μm^2)	0.01	0.09	0.08
Cell count	10	62	41
No. of crossovers	0	0	0
Latency	0.5	1.5	2

Table 2 : Features of various basic logic circuits.

Feature	2X1 MUX	Parity generator	Parity checker
Area (μm^2)	0.08	0.17	0.28
Cell count	34	99	148
No. of crossovers	0	0	0

Latency	1	2	1.5

Table 3: Features of some complex arrays.

Feature	**Binary to Gray code converter**	**Gray to Binary code converter**
Area (μm^2)	0.17	0.23
Cell count	111	114
No. of crossovers	0	0

Table 4: Features of some code converters.

V. CONCLUSIONS

This paper presents the design, layout and simulation of some combinational circuits based on XOR gate. The layouts were simulated using QCA designer, the design and simulation tool for QCA based circuits. These designs are efficient in terms of area and power consumption. All these designs are coplanar. There are further opportunities for optimization and can also be used to build ALU's.

REFERENCES

[1] Hennessy K and Lent C.S, "Clocking of molecular quantum-dot cellular automata," J. Vac. Sci. Technol. B, Microelectron. Process. Phenom., vol. 19, no. 5, pp. 1752–1755, Sep. 2001.
[2] ITRS: international roadmap for semiconductor, 2005, http://www.itrs.net/
[3] Konrad Walus, Timothy Dysart J, Graham Jullien A and Arief Budiman R "QCADesigner: A Rapid Design and Simulation Tool for Quantum-Dot Cellular Automata", IEEE Transactions on Nanotechnology, Vol. 3, March 2004, pp. 26-31.
[4] Kyosun Kim, Kaijie Wu, and Ramesh Karri, "The Robust QCA Adder Designs Using Composable QCA Building Blocks," IEEE transactions on computer-aided design of integrated circuits and systems, vol. 26, no. 1, pp 176-183, 2007.
[5] Lent C.S and Isaksen B, "Clocked molecular quantum-dot cellular automata," IEEE Trans. Electron Devices, vol. 50, no. 9, pp. 1890–1896, Sep. 2003.
[6] Lent C.S, Tougaw P.D, Porod W.D, and Bernstein G.H, "Quantum cellular automata," Nanotechnology, vol. 4, no. 1, pp.49–57, 1993.
[7] Niemier M T, Designing digital systems in quantum cellular automata, Master's Thesis, University of North Dame, 2004.
[8] Seminario J.M, Derosa P.A, Cordova L.E and Bozard B.H, "A molecular device operating at terahertz frequencies," IEEE Transactions on Nanotechnology, vol. 3, no. 1, pp. 215–218, 2004.
[9] Tougaw P.D and Lent C.S, "Logical devices implemented using quantum cellular automata," Journal of Applied Physics, vol. 75, no. 3, pp. 1818–1825, 1994.
[10] Vankamamidi V, Ottavi M, and Lombardi F, "Clocking and cell placement for QCA," in Proceedings of the 6th IEEE Conference on Nanotechnology (IEEE-NANO '06), vol. 1, pp. 343–346, June 2006.

