Efficient Design of Logical Structures and Functions using Nanotechnology Based Quantum Dot Cellular Automata Design

S.Karthigai lakshmi,
Asst Professor, Dept of ECE,
SONA College of Technology,
Salem.

G.Athisha
Professor, Dept of ECE,
PSNA CET,
Dindigul,

ABSTRACT

In this paper, the different types of logical structure based on Quantum dot Cellular Automata (QCA) design are discussed. The QCA offers a new transistorless computing paradigm in nanotechnology. It has the potential for attractive features such as faster speed , smaller size and low power consumption than transistor based technology .By taking the advantages of QCA we are able to design interesting computational architecture. The basic logic elements used in this technology are the inverter and the majority gate. The other logical structures are designed using the basic elements. Here, we have designed the logical structures using 2cells inverter instead of 7cells inverter. By applying this method, the hardware requirements for a QCA design can be reduced. These structures are designed and simulated using QCADesigner. The QCA designer is a design and simulation tool for quantum dot cellular automata. The standard functions and the simplified majority expressions corresponding to these standard functions are also presented.

Keywords: QCA, QCADesigner, majority gate.

1. INTRODUCTION

Current silicon transistor technology faces challenging problems, such as high power consumption and difficulties in feature size reduction [4][5]. Nanotechnology is an alternative to these problems. The Quantum dot cellular automata (QCA) is one of the attractive alternatives. Since QCAs were introduced in 1993 by lent et al, and experimentally verified in 1997. QCA is expected to achieve high device density, extremely low power consumption and very high switching speed.

QCA structures are constructed as an array of quantum cells with in which every cell has an electrostatic interaction with its neighboring cells. QCA applies a new form of computation, where polarization rather than the traditional current, contains the digital information. In this trend, instead of interconnecting wires, the cells transfer the information throughout the circuit. [2]

The fundamental QCA logic primitives are the three input majority gate, wire and inverter. Each of these can be considered as a separate QCA locally interconnected structure, where QCA digital architectures are combination of these cellular automata structures.

This paper describes the design of different logical structures in QCA such as AND, OR, NOT, NAND, NOR, EX-OR and EX-NOR. These structures are designed based on the basic logical devices. The design follows the conventional design approaches but due to the technology differences, they are modified for the best performances in QCA.

Basically the inverter which consists of 7cells is used to design the logical structures such as NAND, NOR, EX-OR and EX-NOR. But in this paper we have designed the logical structures using 2cells inverter. Hence the proposed method can be used to minimize the area and complexity.

The paper is organized as follows, in section 2, describes the QCADesigner tool. The background of QCA technology is explained in section 3. Section 4, provides the QCA clocking

and in section 5, explains the QCA wire crossings. Section 6 shows the design and implementation of logical structures in QCA and standard functions using majority gates are presented in section 7. Simulation results follow in section 8 and conclusions are presented in section 9.

2. QCA DESIGNER

QCA logic and circuit designers require a rapid and accurate simulation and design layout tool to determine the functionality of QCA circuits. QCADesigner gives the designer the ability to quickly layout a QCA design by providing an extensive set of CAD tools. As well, several simulation engines facilitate rapid and accurate simulation. It is the first publicly available design and simulation tool for QCA. Developed at the ATIPS Laboratory, at the University of Calgary, QCADesigner currently supports three different simulation engines, and many of the CAD features required for complex circuit design. [5],[11]

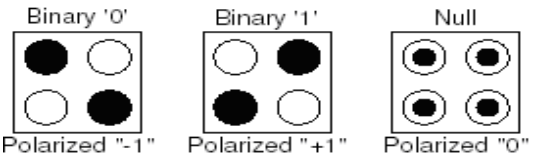
Included in the current version of QCADesigner are three different simulation engines. The first is a digital logic simulator, which considers cells to be either null or fully polarized. The second is a nonlinear approximation engine, which uses the nonlinear cell-to-cell response function to iteratively determine the stable state of the cells within a design. The third uses a two-state Hamiltonian to form an approximation of the full quantum mechanical model of such a system. Each of the three engines has a different and important set of benefits and drawbacks. Additionally, each simulation engine can perform an exhaustive verification of the system or a set of user-selected vectors. [10], [11].

3. BACKGROUND MATERIAL

A. QCA Basics

QCA technology is based on the interaction of bi-stable QCA cells constructed from four quantum dots. The cell is charged with two free electrons, which are able to tunnel between adjacent dots. These electrons tend to occupy antipodal sites as a result of their mutual electrostatic repulsion. Thus, there exist two equivalent energetically minimal arrangements of the two electrons in the QCA cell, as shown in Fig. 1. These two arrangements are denoted as cell polarization P= +1and P= -1.By using cell polarization P =+1 to represent logic "1" and P =-1 to represent logic "0," binary information is encoded in the charge configuration of the QCA cell. [4-6].

Fig.1.QCA cell polarization.



B.QCA Logic Devices

The fundamental QCA logic primitives include a QCA wire, QCA inverter, and QCA majority gate [4]–[6], as described below.

QCA Wire: In a QCA wire, the binary signal propagates from input to output because of the electrostatic interactions between cells. The propagation in a 90˚ QCA wire is shown in Fig. 2. Other than the 90˚ QCA wire, a 45˚ QCA wire can also be used. In this case, the propagation of the binary signal alternates between the two polarizations.

Fig. 2.1 .QCA wire (90˚).

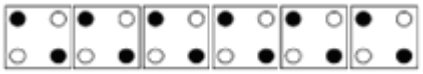
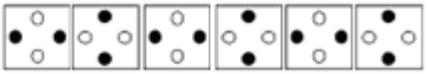
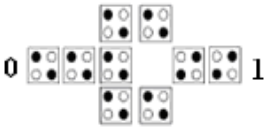


Fig. 2.2 QCA wire (45˚).



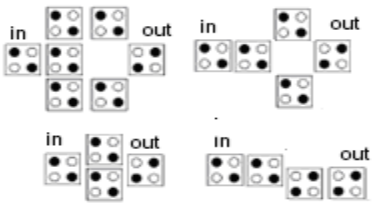
QCA Inverter: The QCA cells can be used to form the primitive logic gates. The simplest structure is the inverter shown. Fig. 3, which is usually formed by placing the cells with only their corners touching.

Fig.3.1.QCAInverter



The electrostatic interaction is inverted, because the quantum-dots corresponding to different polarizations are misaligned between the cells.

Fig.3.2.Several inverter types.



QCA Majority Gate: The QCA majority gate performs a three-input logic function. Assuming the inputs are A, B and C, the logic function of the majority gate is

$$M(A, B, C) = AB+BC+CA. \quad (1)$$

Fig. 4.1 QCA majority gate

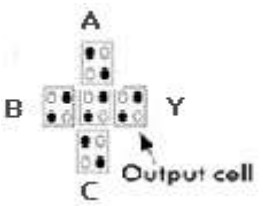
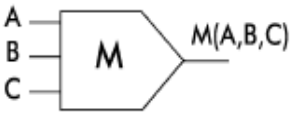


Fig. 4.2 Majority gate Symbol



A layout of a QCA majority gate is shown in Fig. 4. The tendency of the majority device cell to move to a ground state ensures that it takes on the polarization of the majority of its neighbors. The device cell will tend to follow the majority polarization because it represents the lowest energy state. By fixing the polarization of one input to the QCA majority gate as logic "1" or logic "0," an AND gate or OR gate will be obtained, respectively, as follows:

$$M(A,B,0) = AB \quad (2a)$$

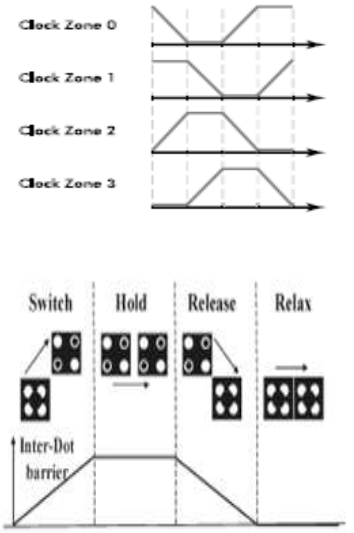
$$M(A,B,1) = A+B \quad (2b)$$

Thus, we can base all QCA logic circuits on three-input majority gates. In order to achieve efficient QCA design, majority gate-based design techniques are required. The study of majority gates mainly focuses on the threshold logic. An n-input majority function using threshold logic. An n-input majority gate produces a logic "1" output if the majority of its inputs are logic"1";two out of three in the case of a three-input majority function. If n is even, n/2+1 inputs must be at logic "1" to produce an logic"1".The three-input QCA majority gate is a special case of the n-input majority function.[4].

4. CLOCKING

The QCA circuits require a clock , not only to synchronize and control information flow but also to provide the power to run the circuit since there is no external source for powering cells(serial add, shifter). With the use of four phases clocking scheme in controlling cells, QCA processes and forwards information within cells in an arranged timing scheme. Cells can be grouped into zones so that the field influencing all the cells in the zones will be the same. A zone cycles through 4 phases. In the **Switch** phase, the tunneling barriers in a zone are raised. While this occurs, the electrons within the cell can be influenced by the Columbic charges of neighboring zones. Zones in the **Hold** phase have a high tunneling barrier and will not change state, but an influence other adjacent zones. Lastly, the **Release** and **Relax** decrease the tunneling barrier so that the zone will not influence other zones. These zones can be of irregular shape, but their size must be within certain limits imposed by fabrication and dissipation concerns. Proper placement of these zones is critical to design efficiency. This clocking method makes the design of QCA different from CMOS circuits. [9], [14]

Figure 5.The four phases of the QCA clock



The Fig. 5. Shows the four available clock signals. Each signal is phase shifted by 90˚ degrees. When the clock signal is low the cells are latched. When the clock signal is high the cells are relaxed and have no polarization. In between the cells are either latching or relaxing when the clock is decreasing/increasing respectively.

5. CROSSOVER DESIGN

In QCA, there are two crossover options. [3], [5]. There are coplanar crossings and multilayer crossovers. Coplanar crossings use only one layer, but require using two cell types (regular and rotated). The regular cell and the rotated cell do not interact with each other when they are properly aligned, so rotated cells can be used for coplanar wire crossings. Published information suggests that coplanar crossings may be very sensitive to misalignment. In the coplanar crossing, rotated cells are used when two wires cross. By choosing the connection point from rotated cells, either an original or an inverse of the input is available. In a coplanar crossing, there is a possibility of a loose binding of the signal which causes a discontinuity of the signal propagation and there is the possibility of back-propagation from the far side constant input. So putting enough clock zones between the regular cells across the rotated cells is required. On the other hand, a multilayer crossover is quite straightforward from the design perspective and the signal connection is steadier. The implementation process is less well understood than that for coplanar crossings. The multilayer crossovers use more than one layer of cells like multiple metal layers in a conventional IC. An example layout of coplanar and multilayer wire crossings are shown in Figs. 6 and 7.

Fig. 6 .Layout of coplanar crossings.

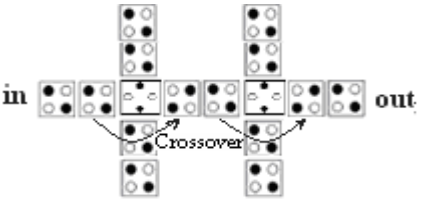
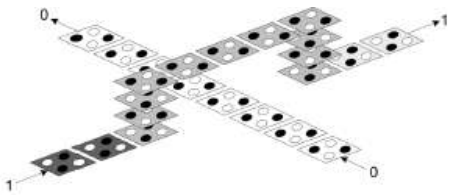


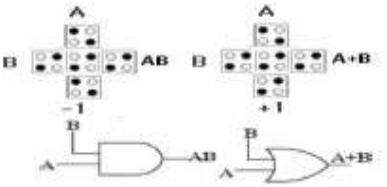
Fig. 7 .Layout of coplanar crossings.



6. QCA IMPLEMENTATION

The AND and OR gates are realized by fixing the polarization to one of the inputs of the majority gate to either P = −1 (logic “0”) or P = 1 (logic “1”).

Fig. 8 .Layout of AND and OR gates.



The NAND function is the complement of AND function. It is realized by connecting AND gate (MG) followed by an inverter. Similarly the NOR gate is realized by connecting OR gate (MG) followed by an inverter. If the last two cells are arranged as shown in the following figure then it acts as an inverter. By using this 2cell inverter, the area required and complexity can be minimized.

Fig. 9 .Layout of NAND gate.

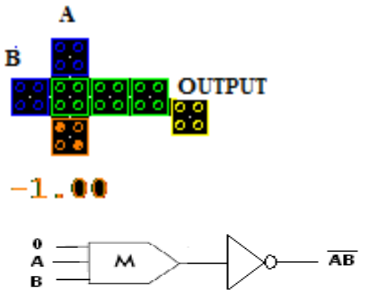
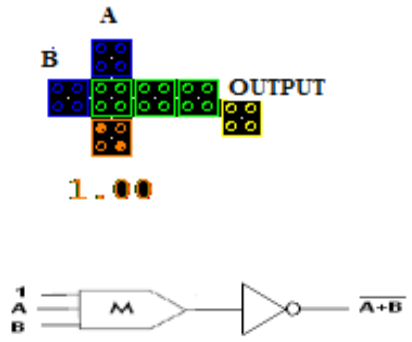


Fig. 10 .Layout of NOR gate



The 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 to follow. The realization is done making use of majority gates (MGs) and following the equations as follows

$$A \oplus B = A'B + AB' \quad (3)$$

.

Fig.11.XOR schematic

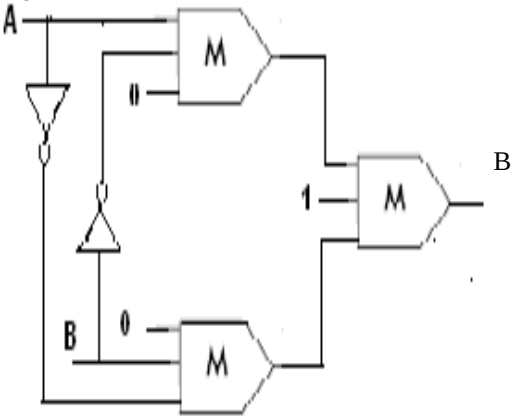


Fig.12. Layout of XOR gate

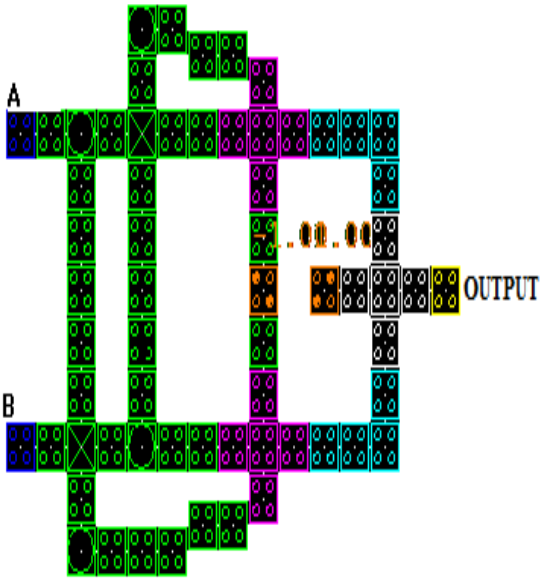


Fig.13.Ex-NOR schematic.

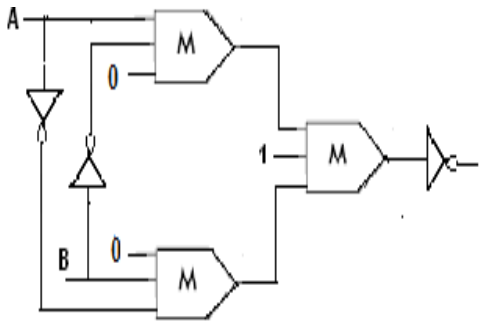


Fig.14. Layout of Ex-NOR gate

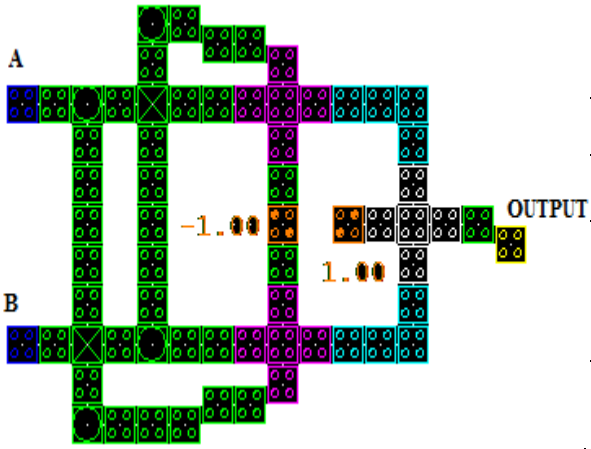


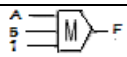
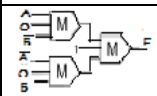
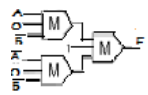
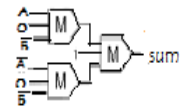
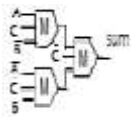
Table.1.Comparision of Logical Structures

Logical structures	Previous structures		New structures		Clocking
	Complexity	Area	Complexity	Area	
Inverter	7 cells	80nmx 58nm	2 cells	44nm x 30nm	Simple
NAND	13 cells	139nm x 58nm	7 cells	63nm x 103nm	Simple
NOR	13 cells	139nm x58nm	7cells	63nm x 103nm	Simple
XOR	87 cells	290nm x260n m	64 cells	300nm x 220nm	1
Ex-NOR	93 cells	290nm x350n m	65 cells	327nm x 220nm	1

7. STANDARD FUNCTIONS

The three variables A,B and C to facilitate the conversion of a sum-of-products expression to minimized majority logic. Based on that to obtain the efficient majority expression for any given three-variable Boolean function amenable to QCA implementation. The simplified majority expressions for some standard functions are given in the table.

Table.2.Majority gate representation of functions

Name of the function	Function	Function in Majority Gate	Majority Gate schematic
OR	F=A+B	M(A,B,1)	
AND	F=AB	M(A,B,0)	
XOR	F=A'B+AB'	M(M(A',B,0) M(A,B',0) ,1)	
Ex-NOR	F=AB+A'B'	M (M(A,B,0), M(A',B',0),1)	
Half Adder	Sum=A'B+B'A Cout=AB	Sum= M (M(A',B,0) M(A,B',0) ,1) Cout = M(A,B,0)	
Full Adder	Sum= ABC+A'B'C +A'BC'+AB'C' Cout =AB+BC+AC	Sum= M (M(A',B,C) , M(A,B',C)	

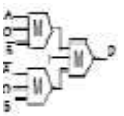
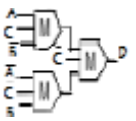
		,C') Cout = M(A,B,C)	
Half Subtractor	D= A'B+B'A Bout=A'B	D=M (M(A',B,0) M(A,B',0) ,1) Bout= M(A',B,0)	
Full subtractor	D= ABC+A'B'C +A'BC'+AB'C' Bout = AB+BC+AC	D= M (M(A',B,C) , M(A,B',C) ,C') Bout = M(A,B,C)	

Fig.17. Simulation result of XOR gate

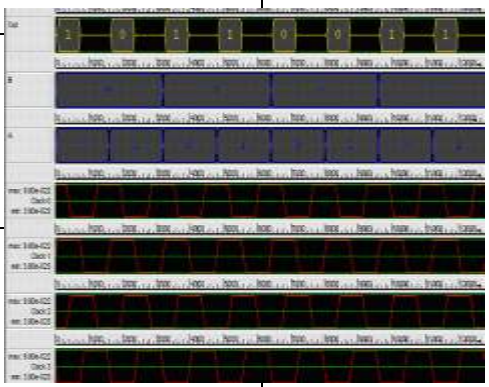
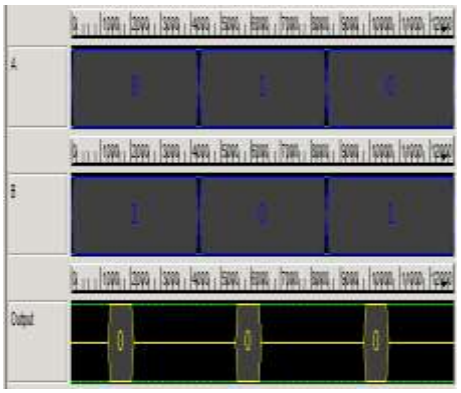


Fig.17. Simulation result of Ex-NOR gate



8. SIMULATION RESULTS

With QCADesigner ver.2.0.3, the circuit functionality is verified. The following parameters are used for a bistable approximation: Cell size 20nm, Number of samples 12800, Convergence tolerance 0.001000, Radius of effect 65nm, Relative permittivity 12.9, Clock high 9.8e-22J, Clock low3.8e-23J, Clock amplitude factor2, Layer separation 11.5nm, Maximum Iterations per sample 10000.

Fig.15. Simulation result of NAND gate

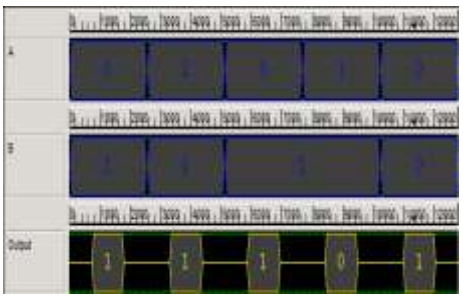
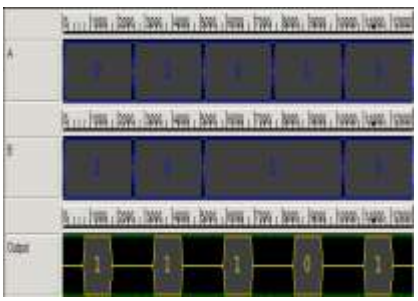


Fig.16. Simulation result of NOR gate



9. CONCLUSION

The logical structures using QCA has been designed and tested using QCADesigner software. The operation of the structures has been verified according to the truth table. The design works satisfactorily and produces required results. The implementation of this design may lead to the efficient use of a logical unit in various applications. The logical unit thus designed may be used as a basic building block of a general purpose Nano processor which may be a future technical advancement of this work. This may pave way for the design and development of other application oriented processors.

REFERENCES

[1] International Technology Roadmap for Semiconductors. (ITRS) 2005
[Online]. Available: http://www.itrs.net

[2] K. Walus, G.A. Jullien, and V. Dimitrov. Computer arithmetic structures for quantum cellular automata. *Asilomar Conference on Signals, Systems, and Computers*, November 2003.

[3] W. Wang, K. Walus, and G.A. Jullien. Quantum-dot cellular automata adders. *IEEE Nano Conference*, August 2003.

[4] R. Zhang, K. Walus, W. Wang, and G. A. Jullien, "A method of majority logic reduction for quantum cellular automata," *IEEE Trans.Nanotechnol.*, vol. 3, no. 4, pp. 443–450, Dec. 2004.

[5] Heumpil Cho, Student Member, and Earl E. Swartzlander, Jr., Fellow, IEEE. "Adder Design and Analyses for Quantum-Dot Cellular Automata", *IEEE Trans. Nano.*, Vol.6, No.3, may 2007.

[6] K. Walus, G. Schulhof, G. A. Jullien, R. Zhang, and W. Wang, "Circuit design based on majority gates for applications with quantum-dot cellular automata," in *Conf. Rec. 38th Asilomar Conf. Signals, Systems and Computers*, 2004, vol. 2, pp. 1354–1357.

[7] W. J. Townsend and J. A. Abraham, "Complex gate implementations for quantum dot cellular automata," in *Proc. 4th IEEE Conf. Nanotechnology*, 2004, pp. 625–627.

[8] Heumpil Cho, Student Member, and Earl E. Swartzlander, Jr., Fellow, IEEE. "Adder Design and Analyses for Quantum-Dot Cellular Automata", *IEEE Trans. Nano.*, Vol.6, No.3, may 2007.

[9] 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, January 2007.

[10] K. Walus, G. Schulhof, and G. A. Jullien, "High level exploration of quantum-dot cellular automata (QCA)," in *Conf. Rec. 38th Asilomar Conf. Signals, Systems and Computers*, 2004, vol. 1, pp. 30–33.

[11] K. Walus, T. Dysart, G. Jullien, and R. Budiman, "QCADesigner: A rapid design and simulation tool for quantum-dot cellular automata,"*IEEE Trans. Nanotechnol.*, vol. 3, no. 1, pp. 26–29, Mar. 2004.

[13] A. Chaudhary *et al.*, "Eliminating wire crossings for molecular quantum-dot cellular automata implementation," in *Proc. IEEE/ACM Int. Conf. Computer-Aided Design*, 2005, pp. 565–571.

[14] A. Vetteth *et al.*, "Quantum-dot cellular automata carry-look-ahead adder and barrel shifter," presented at the IEEE Emerging Telecommunications Technologies Conf., Richardson, TX, 2002.

[14] S. Frost, A. F. Rodrigues, A. W. Janiszewski, R. T. Raush, and P. M. Kogge, "Memory in motion: A study of storage structures in QCA," presented at the First Workshop Non-Silicon Computation, Boston, A, 2002.

[15] V. Vankamamidi, M. Ottavi, and F. Lombardi, "A line-based parallel memory for QCA implementation," *IEEE Trans. Nanotechnol.*, vol. 4, no. 6, pp. 690–698, Nov. 2005.

[16] R. Zhang, K. Walus, W. Wang, and G. A. Jullien, "Performance comparison
of quantum-dot cellular automata adders," in *Proc. IEEE Int. Symp. Circuits and Systems*, 2005, vol. 3, pp. 2522–2526.

[17] C. S. Lent and B. Isaksen, "Clocked molecular quantum-dot cellular automata,"
IEEE Trans. Electron Devices, vol. 50, no. 9, pp. 1890–1896,Sep.2003.