01 Oct 2006

Efficient and Robust Delay-Insensitive QCA (Quantum-Dot Cellular Automata) Design

Minsu Choi
Missouri University of Science and Technology, choim@mst.edu

Myungsu Choi

Zachary D. Patitz

Nohpill Park

Efficient and Robust Delay-Insensitive QCA (Quantum-Dot Cellular Automata) Design

Minsu Choi[1], Myungsu Choi[2]*, Zachary Patitz[2] and Nohpill Park[2]
[1]Dept of ECE, University of Missouri-Rolla, Rolla, MO 65409-0040, USA
choim@umr.edu
[2]Dept of CS, Oklahoma State University, Stillwater, OK 74078, USA
{myung, patitzz, npark}@cs.okstate.edu

Abstract

The concept of clocking for QCA, referred to as the four-phase clocking, is widely used. However, inherited characteristics of QCA, such as the way to hold state, the way to synchronize data flows, and the way to power QCA cells, make the design of QCA circuits quite different from VLSI and introduce a variety of new design challenges. The most severe challenges are due to the fact that the overall timing of a QCA circuit is mainly dependent upon its layout. This issue is commonly referred to as the "layout=timing" problem. To circumvent the problem, a novel self-timed circuit design technique referred to as the Locally Synchronous, Globally Asynchronous Design for QCA has been recently proposed. The proposed technique can significantly reduce the layout-timing dependency from the global network of QCA devices in a circuit; therefore, considerably flexible QCA circuit design is be possible. Also, the proposed technique is more scalable in designing large-scale systems. Since a less number of cells is used, the overall area is smaller and the manufacturability is better. In this paper, numerous multi-bit adder designs are considered to demonstrate the layout efficiency and robustness of the proposed globally asynchronous QCA design technique.

1 Introduction

QCA (Quantum-Dot Cellular Automata) is one of the six promising technologies for nano-scale computing listed in the International Technology Roadmap for Semiconductors (ITRS) 2004 [1]. In the QCA paradigm, a regular array of cells, each interacting with its neighboring cells, is employed in a locally interconnected architecture [2, 3]. The coupling between the cells is given by their electrostatic interactions. Such arrays are in principle capable of encoding digital information. The fundamental unit of QCA is the QCA cell created with four quantum-dots positioned at the vertices of a square. The cell is loaded with two extra electrons which tend to occupy the diagonals due to electrostatic repulsion. Binary information is encoded in the two possible polarizations (i.e., +1 or -1). The cell will switch from one polarization to the other when the electrons quantum-mechanically tunnel from one set of dots to the other [4]. Nanomagnet-based QCA are also gaining significant attention since they are easier to fabricate and possible to operate in room temperature [5].

---

*Myungsu Choi is currently with LG Electronics, Seoul, Korea.

The concept of clocking for QCA is referred to as the four-phase clocking [6, 7, 8, 9]. Four-phase clocking signal applied to four adjacent buried wires. Each wire has a voltage that raises and lowers linearly in order to adiabatically switch the QCA cells placed above it. Adjacent wires have a $\pi/2$ phase shift so that every fourth wire has an identical signal. This method will induce a roughly sinusoidal clocking field that propagates along the QCA surface.

When the clock signal is high the potential barriers between the dots are low and the electrons effectively spread out in the cell and no net polarization exists; i.e. P=0. As the clock signal is switched low, the potential barriers between the dots are raised high and the electrons are localized such that a polarization is developed based on the interaction of their neighbors; i.e. they take on the polarization of their neighbors. Basically clock high means cell is unlatched, clock low means cell is latched. So, the four-phase clocking scheme emulates classical shift register behavior since a binary digit of information can be stored in each cell's latched polarization [7, 8, 10].

With the four-phase clocking scheme, it is possible to design QCA circuits while ensuring each QCA cell is powered and the information is processed and forwarded in timely manner. However, inherited characteristics of QCA, such as the way to hold state, the way to synchronize data flows, and the way to power QCA cells, make the design of QCA circuits quite different from VLSI and introduce a variety of new design challenges and the most severe challenges are due to the fact that the overall timing of a QCA circuit is mainly dependent upon its layout. This fact is commonly referred to as the **"layout=timing" problem** [9, 11]. To address the various layout-induced design challenges, considerable research efforts have been done in the field of architecture and design automation of QCA [11, 12].

Unlike the previous approaches, the proposed methodology is to completely eliminate such "layout=timing" constraints from the global network of QCA gates in a circuit. The key idea is to introduce delay-insensitive data encoding scheme such as NCL (Null Convention Logic) [14] to the circuit-level to eliminate the global layout=timing problem, while preserving the 4-phase clocking scheme for individual gates. Since each gate is locally synchronized by corresponding clocking zone(s), appropriate data forwarding and synchronization are guaranteed at the gate-level and the QCA cells within the gate can be properly powered, as well. As a result, the fundamental "layout=timing" problem for QCA circuits can be completely removed, since the overall layout of the circuit is no more the determinant factor of the global timing. Numerous self-timed gates were designed in QCA and reported in [13].

In this paper, we will present a globally delay-insensitive full adder design in QCA, first. Then, we will demonstrate the layout efficiency and robustness of the proposed robust GALS QCA by considering n-bit adder designs based on it.

2   Preliminaries and Review

A synchronous QCA full adder consists of three MV gates, two inverters, and 5 different clock zones and 165 cells are used in the design as shown in Figure 1. Although this full adder has logically minimized form, it is not free from the "layout=timing" problem. For example, let us use the full adder as a building block to develop a 2-bit full adder. This is simply done by making a connection between the carry-out of the first bit full adder and

IEEE
COMPUTER
SOCIETY
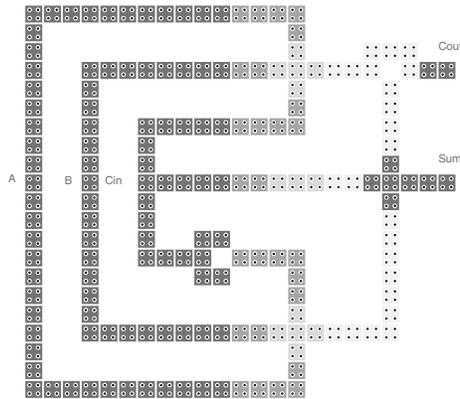
Figure 1. Layout of synchronous QCA full adder.

the carry-in of the second bit full adder and adding more buffers for synchronization. Since the carry-in value of the second bit full adder will be delayed by 5 clock zones, the other two inputs of the second bit full adder should be delayed by 5 clock zones. In addition, output of the first digit full adder should also be delayed by 5 clock zones. The layout of the synchronous 2-bit full adder is shown in Figure 2(A). It is obvious that the buffers consist of clocked wires and that they waste a lot of cells and space. This design may cause complications due to the "layout=timing" problem which was discussed in previous section.

In order to address the "layout=timing" problem with conventional QCA with 4-phase clocking scheme, a novel self-timed design method has been recently proposed based on a delay-insensitive data encoding and signaling protocol. The proposed novel QCA-specific delay-insensitive design approach is referred to as the "Globally Asynchronous, Locally Synchronous QCA (GALS QCA)". Notably, the local network of QCA cells within the gate is fully synchronized and powered by 4-phase clocking scheme, but the gate itself is delay-insensitive, since it has hysteresis behavior. It would be possible to design delay-insensitive QCA circuits using QCA-implemented NCL gates. GALS QCA design has two major benefits over its synchronous counterpart. Firstly, it addresses the "layout=timing" problem; meaning that individual gates can be freely allocated and interconnected without inserting delay buffers to synchronize switching operations. Secondly, it enhances the overall robustness of the circuit. Since it has better scalability over the synchronous QCA, it may use less number of cells; especially, in case of large-scale circuit design. So, it is easier to assemble and less prone to defects and faults. Also, it is immune to stuck-at faults, since such faults simply prevent the GALS QCA circuit to make transitions between DATA and NULL. So, the circuit will simply halt.

Threshold gates with hysteresis are the basic building blocks of NCL designs. One type of threshold gate with hysteresis is the M-of-N threshold gate with hysteresis. This is denoted as THmn gate, and has n number of inputs and one output signal where $1 \leq m < n$ [14]. THmn gate has two special properties: threshold behavior and hysteresis behavior. The threshold behavior means that the output of the THmn gate becomes 1 when at least m of n inputs becomes 1. The hysteresis behavior means that the output only changes after a sufficiently complete set of inputs have been established. For example, if the current output of TH23 gate is 0, then the output remains 0 until at least 2 of 3 inputs become 1. In addition, to change the output from 1 to 0, all 3 inputs must be 0.

NCL circuits switch between a logic based data representation of DATA and a control representation of NULL. This separation between control and data representations provides a self-synchronization throughout the design. Therefore,no global synchronization is needed. Threshold gates provide the basic building block of NCL designs. Threshold gate inputs and outputs can be in one of two states, DATA or NULL. A threshold gate starting with its output in a NULL state will remain in the NULL state until the specified number of inputs are placed in the DATA state. Once the gate reaches the DATA state, it remains in this state until all of the inputs return to the NULL state. The hysteresis in the threshold gate provides the threshold needed to keep from switching during the intermediate state when the number of inputs in the DATA state is greater than zero, but less than the threshold limit. In addition, hysteresis provides the storage to remain at DATA until all of the inputs have returned to NULL. Since these gates use two values, as traditional Boolean logic does, they can be constructed with traditional digital logic design processes.
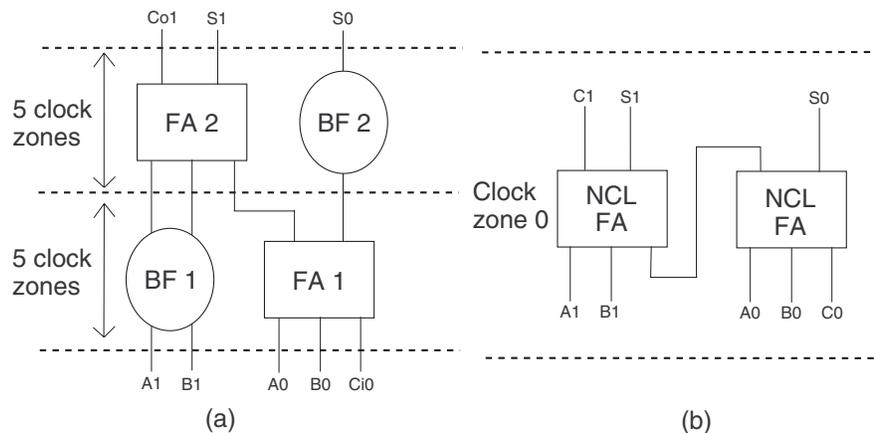

Figure 2. (a) The 2-bit full adder using synchronous QCA full adder as building block. In order to synchronize the circuit, two buffers are added. (b) NCL 2-bit full adder scheme. It is assumed that all inputs and outputs of full adders are in a same clocking phase (i,e,. "Switch" phase for the clock zone 0).

3   Delay-Insensitive QCA 1-Bit Full Adder Design

A dual rail full adder consists of two TH23 gates and two TH34W2 gates [14] as shown in Figure 3. Since both TH23 and TH34W2 gates are delay-insensitive, they can be located anywhere in the given area and the corresponding interconnects and fan-ins and fan-outs can be also freely placed, because the global timing dependency is fully removed. The NCL full adder has inherited threshold and hysteresis behavior from TH23 and TH34W2 gates. This NCL full adder as well as any NCL circuit consisting of NCL full adders does not obey the "layout=timing" constraint. For example, when we design a 2-bit full adder by using a NCL full adder as a building block, a 2-bit full adder is simply obtained by connecting the carry-out of the first digit NCL full adder to the carry-in of the second digit NCL full adder. Each NCL full adder in 2-bit full adders is delay-insensitive; so they can be placed

in a same clock zone as shown in Figure 2(B). Therefore, any NCL circuit designed with NCL full adders also has the same properties that a NCL full adder does.
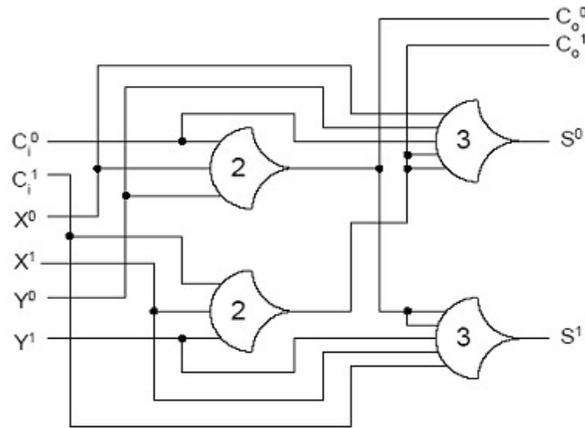

Figure 3. Dual-rail full adder design. Two TH23 and two TH34W2 gates are used. The design is placement-insensitive when implemented in QCA, since all wires outside THgates are in a single clock zone and THgates are delay-insensitive. The design can be located anywhere in the given area. Figure adopted from [14].

The NCL full adder implemented in QCA consists of 1158 QCA cells and takes up 1.9 μm^2 space, but the synchronous full adder consists of 165 cells and takes up 0.21 μm^2 space only. Approximately, the NCL full adder is 7 times bigger than the synchronous full adder. So, the NCL full adder implementation seems to be inefficient especially when the overall system is small. However, the NCL full adder is quite efficient for larger systems in which multiple instances are allocated (i.e., multi-bit adder design). If n number of NCL full adders are used to develop n-bit full adder, NCL full adders can be located without any synchronization. Also, no buffer insertion for synchronization is required, because of their delay-insensitivity. In the same case, synchronous n-bit full adder needs $\sum(n-1)$ number of input buffers and $\sum(n-1)$ number of output buffers to synchronize. So, NCL is more suitable to a big size circuit design. For example, in order to design a 4-bit synchronous full adder, additional six input buffers and six output buffers are necessary for synchronization reason, but 4-bit NCL full adder can be designed with four 1-bit full adders. Addition of delay buffers may cause serious loss in both cell count and clock zone space in case of synchronous design. Figure 5 describes 4-bit full adder example.

An output buffer is nothing but a clocked array of cells. Since the width between input and output of synchronous full adder is 23, cell count of an output buffer is also 23. This is minimized cell count for output buffer. An input buffer has two clocked arrays, and length of each array is also 23. Cell count of an input array, therefore 46. That is also the minimized cell count for the input buffer. Obviously, carry-out wire of current adder is only an extension of the carry-in wire to the next adder. This extension wire is not a delay buffer, but a simple extension wire between neighboring adders. So, there are $n-i$ extension wires in n-bit adder. If the cell count of an extension wire is e, then the total cell count of synchronous n-bit adder can be estimated by :
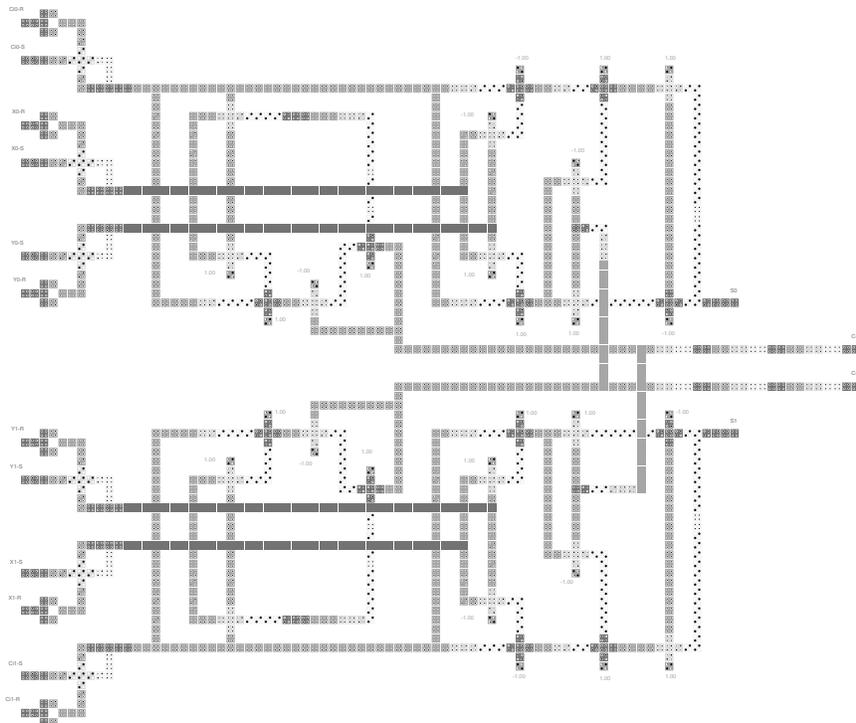
Figure 4. Layout of QCA-based NCL full adder and six flip-flops that substitute a NCL register. Flip-flops generate input signals which represent hysteresis behavior. 1158 cells of 1350 total cells are used for full adder design in $1.9 \ \mu m^2$ space.

(A)   (B)
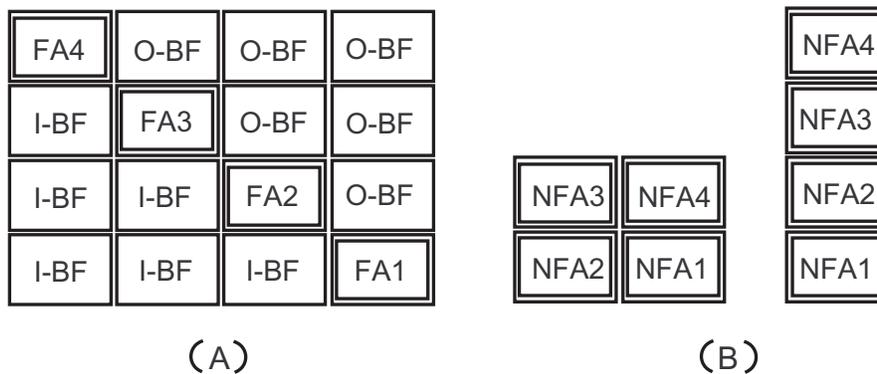
Figure 5. 4-bit full adder example. (A) Synchronous 4-bit full adder. Six input delay buffers and six output delay buffers are used with four 1-bit synchronous full adder. (B) NCL 4-bit full adder. Only four NCL 1-bit full adders are used. Shape of 4-bit full adder is either rectangular or linear, since each NCL full adder can be placed anywhere in the given area.

$$165n + \sum_{i=1}^{n} 23(i-1) + \sum_{i=1}^{n} 46(i-1) + e(n-1) = 165n + \sum_{i=1}^{n} 69(i-1) + e(n-1) \qquad (1)$$

A synchronous 1-bit full adder takes up 0.21 μm^2 space. Both Adders and buffers occupy the same size space like Figure 5(A). Total used space of synchronous n-bit adder is $n^2 \times 0.21$ μm^2.

Total cell count of NCL n-bit adder is simple. 1158 cells are used in each full adder. NCL n-bit adder consists of n number of NCL full adders. Even though NCL n-bit adder does not require any additional buffers to synchronize, it still needs extension wires between the two neighboring adders. Since the NCL full adder employs a dual rail encoding scheme, two extension wires should be inserted between neighboring adders. So, total cell count of extension wires is shown in terms of $2e' \times (n-1)$, where e' is cell count of each extension wire. Total cell count of n-bit full adder is:

$$1158 \times n + 2e' \times (n-1). \qquad (2)$$

Total used area can be calculated in same manner. Size of NCL full adder is 1.9 μm^2. Therefore, total used area of NCL n-bit adder is:

$$n \times 1.9 \ \mu m^2. \qquad (3)$$

The cell count of extension wire e and e' are not fixed. Since the size of e and e' are mainly dependent on the layout of the circuit, they can be reduced and minimized by proper floor planning. For example, if all the carry-out outputs are directly connected to the carry-in input of next digit adder without an extension wire, value of both e and e' will be 0. Considering this ideal case, we can remove unknown value e and e' from formula 1 and 2. Table 1 has been developed without both e and e'.

		$n=8$	$n=16$	$n=30$	$n=32$	$n=64$	$n=128$
Cell	FA	3,252	10,920	34,965	39,524	149,664	581,952
Count	NFA	9,264	18,528	34,740	37,056	74,112	148,224
Used	FA	13.44	53.76	189.0	215.04	860.16	3440.64
Area	NFA	15.2	30.4	57.0	60.8	121.6	243.2

Table 1. Total used cell count and used space for two different versions of n-bit full adder.

The synchronous full adder seems to be more efficient when n is less than 27. However, synchronous full adder wastes more space which is occupied by delay buffers. Insertion of a delay buffer reduces circuit density. For example, when $n = 28$, total cell count of both synchronous full adder and NCL full adder are almost same, but the used space of synchronous full adder is 3.5 times larger than used space of NCL full adder. When n becomes over 30, the NCL full adder is more efficient in both cell count and used space aspects. Thus, GALS QCA seems to be more efficient in designing complex systems, in which a large number of sub-circuits are interconnected. Assuming that there is a possible

IEEE
COMPUTER
SOCIETY

way to reduce the size of a buffer. However, this may not reduce cell count, because the cell count of both input and output buffers is already minimized (i.e., 23 cell for output buffer and 46 cell for input buffer). It makes global floorplanning more complex, since space reduction may change each buffer's shape and clock zone assignment. On the other hand, no complex floorplanning is required for the NCL full adder. Since NCL full adder is delay insensitive, each can be located within any spot in a given area. The overall timing of QCA circuits is mainly dependent upon its layout [9]. Asynchronous n-bit adder is an extreme case of this timing constraint that is referred to as the "Layout=Timing" problem which has been discussed in the previous chapter. QCA-based NCL devices are free from this severe constraint in designability. GALS QCA design method also enhances the overall robustness of the system since it has better scalability. If the cell defect rate is denoted by λ and the cell yield and the system yield are assumed to follow the Poisson yield model, then the yield of each cell and the yield of the whole system can be calculated by $e^{-\lambda}$ and $e^{-\lambda \cdot n}$, respectively. In Figure 6, the 128-bit NFA has higher yield over the 128-bit FA. This result is obvious, since 128-bit NFA design utilizes much less QCA cells.
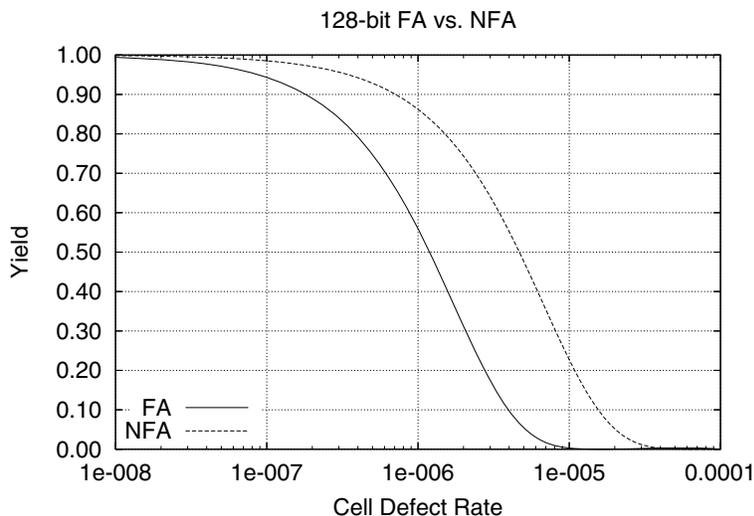
Figure 6. 128-bit FA and NFA yield curves.

4   Conclusion

The concept of clocking for QCA, referred to as the four-phase clocking, is widely used and it is possible to design QCA circuits while ensuring each QCA cell is powered and the information is processed and forwarded in a timely manner. However, the synchronous QCA suffers from this strict timing constraints referred to as the "layout=timing" problem. Applying an asynchronous design methodology to QCA paradigm addresses this problem, and it also provides numerous advantages such as easier floorplanning, increased circuit density, reduced complexity, decreased cell count, reduced space and enhanced manufacturability.

In order to realize the global asynchrony in QCA, one of the CMOS-based asynchronous methodologies, namely NCL, has been used. THmn gates are basic building units for NCL circuit designs, and they have threshold and hysteresis behaviors. These behaviors

are significant since they provide delay-insensitivity for the NCL circuits. In order to synchronize, NCL circuits refer to control data represent NULL rather than time. Therefore, they do not need delay buffers and zone assignment to synchronize. QCA-based TH23 and TH34W2 gates have been developed to optimize QCA asynchronous methodology, and they also have the same properties that CMOS based THmn gates have. Basically, "Layout=Timing" problem forces synchronous QCA circuits to waste many buffers for synchronization purpose. Applying NCL to QCA eliminates the "Layout=Timing" problem as well as the delay buffers from QCA circuits. QCA-based NCL 1-bit full adder has been developed using two TH23 gates and two TH34W2 gates. In order to verify validity of the QCA-based NCL full adder, six modified SR flip-flops that substitute the NCL register are added into the NCL full adder design. Simulation result of the QCA-based NCL full adder indicates that NCL can be realized in the QCA paradigm. QCA-based NCL full adder has succeeded two important behaviors such as hesteresis behavior and threshold behavior from TH23 and TH34W2 gates. NCL circuits do not need buffers to synchronize, since they are delay-insensitive. A globally asynchronous 1-bit full adder design has been proposed and some multi-bit adders based on it have been considered to demonstrate the layout efficiency and robustness of the proposed technique.

References

[1] International Technology Roadmap for Semiconductors, "International Technology Roadmap for Semiconductors (ITRS) 2004," *http://public.itrs.net*, 2004.

[2] G. Snider , A. Orlov, I. Amlani, G. Bernstein, C. Lent, J. Merz and W. Porod, "Quantum-Dot Cellular Automata," Microelectronic Engineering, Vol 47, pp 261-263, 1999.

[3] G. Toth and C. Lent, "Quasiadiabatic Switching for Metal-Island Quantum-Dot Cellular Automata," Journal of Applied Physics, Vol 85, pp 2977-2984, 1999.

[4] F. Rojas, E. Cota and S. Ulloa, "Magnetic Field and Dissipation Effects on the Charge Polarization in Quantum Cellular Automata," IEEE Transactions on Nanotechnology, Vol 3, pp. 41-, 2004.

[5] A. Imre, G. Csaba, L. Ji, A. Orlov, G. H. Bernstein and W. Porod, "Majority Logic Gate for Magnetic Quantum-Dot Cellular Automata," Science, Vol 311, No. 13, pp. 205-208, Jan 2006.

[6] C. Lent and B. Isaksen, "Clocked Molecular Quantum-Dot Cellular Automata," IEEE Transactions on Electron Devices, Vol 50, pp. 1890-1896, 2003.

[7] A. Orlov, R. Kummamur, R. Ramasubramaniam, C. Lent, G. Bernstein and G. Snider, "Clocked Quantum-Dot Cellular Automata Shift Register," Surface Science, Vol 532, pp.1193-1198, 2003.

[8] A. Orlov, R. Kummamuru, R. Ramasubramaniam, C. Lent, G. Bernstein and G. Snider, "A Two-Stage Shift Register for Clocked Quantum-Dot Cellular Automata," Journal of Nanoscience and Nanotechnology, Vol 2, pp 351-355, 2002.

[9] M. Niemier and P. Kogge, "Problems in Designing with QCAs: Layout Equals Timing," International Journal of Circuit Theory and Applications, Vol 29, pp 49-62, 2001.

[10] R.K. Kummamuru, A. Orlov, R. Ramasubramaniam, C. Lent, G. Bernstein and G. Snider, "Operation of a Quantum-Dot Cellular Automata (QCA) Shift Register and Analysis of Errors," IEEE Transactions on Electron Devices, Vol 50, pp. 1906-1913, 2003.

[11] D.A. Antonelli, D.Z. Chen, T.J. Dysart, X.S. Hu, A.B. Kahng, P.M. Kogge, R.C. Murphy and M.T. Niemier, "Quantum-Dot Cellular Automata (QCA) Circuit Partitioning: Problem Modeling and Solutions," The 41st Design Automation Conference (DAC), June 2004.

[12] M.T. Niemier, "Designing Digital System in Quantum Cellular Automata," MS CSE Thesis, Univ of Notre Dame, Apr 2000.

[13] Minsu Choi and Nohpill Park, "Locally Synchronous, Globally Asynchronous Design for Quantum-Dot Cellular Automata (LSGA QCA)," IEEE International Conference on Nanotechnology, pp. 121-124, July 2005.

[14] J.S.Yuan and W.Kuang, "Teaching Asynchronous Design in Digital Integrated Circuits," IEEE Transactions on Education, Vol. 47, No.3, pp397-404 2004.

IEEE
COMPUTER
SOCIETY