ISSN 2348 – 8034
Impact Factor- 3.155

GLOBAL JOURNAL OF ENGINEERING SCIENCE AND RESEARCHES

ESTIMATION OF AREA, DELAY IN EFFICIENT BINARY ADDRESS USING QCA

Krishna Naik Dungavath[*1] and Dr.V.Vijayalakshmi[2]
[*1,2]Department of Electronics and Communication Engineering, Pondicherry Engineering College , Pudducherry.

ABSTRACT

As transistors decrease in size much more of them can be embedded in a one chip, , so increasing chip computational features. Anyhow , transistors will not take so less size than the required size. current size. The technique of QCA denotes is the one method of many reliable solutions to achieve this limited physical size . the implementation of logic modules of QCA will not correct. In this concept, the postulated new adder that operated in all states of art competitors and reaching goals of best in chip size. these advantages are taken by with over all area same to the cheaper implement of known literature. The 128 - bit version of the novel adder achieves the best area - delay tradeoff.

Keywords- *Adders, noncompeting, quantum-dot cellular automata (QCA).*

I. INTRODUCTION

Nanotechnology draws much attention from the public now-a-days. Because the current silicon transistor technology faces challenging problems, such as high power consumption and difficulties in feature size reduction, alternative technologies are sought from researchers. Quantum-dot cellular automata (QCA) is one of the promising future solutions. Since it was first introduced in 1993, experimental devices for semiconductor, molecular, and magnetic approaches have been developed. Quantum dot cellular automata, which is an array of coupled quantum dots to implement Boolean logic functions. The advantage of QCA is high packing densities due to the small size of the dots, simplified interconnection and low area delay product.

1. Adders

In electronics, an adder or summer is a digital circuit that performs addition of numbers. In many computers and other kinds of processors, adders are used not only in the arithmetic logic unit(s), but also in other parts of the processor, where they are used to calculate addresses, table indices, and similar operations. Although adders can be constructed for many numerical representations, such as binary-coded, decimal or excess-3, the most common adders operate on binary numbers. In cases where two's complement or ones' complement is being used to represent negative numbers, it is trivial to modify an adder into an adder subtract or. Other signed number representations require a more complex adder. Adders are fundamental circuits for most digital systems and several adder designs in QCA have been proposed, and a performance comparison was improved. Better adder performance depends on minimizing the carry propagation delay and reducing the area.

2. Quantum dot Cell

In 1993, Lent et al. proposed a physical implementation of an automaton using quantum dot cells. The automaton quickly gained popularity and it was first fabricated in 1997. Lent combined the discrete nature of both cellular automata and quantum mechanics, to create nano-scale devices capable of performing computation at very high switching speeds and consuming extremely small amounts of electrical power. Today, standard solid state QCA cell design considers the distance between quantum dots to be about 20 nm, and a distance between cells of about 60 nm. Quantum dot Cellular Automata are based on the simple interaction rules between cells placed on a grid. A QCA cell is constructed from four quantum dots arranged in a square pattern. These quantum dots are sites electrons can occupy by tunneling to them. Because of Columbic repulsion, the two electrons will always reside in opposite corners. The locations of the electrons in the cell (also named polarizations *P)* determine two possible stable states that can be associated to the binary states 1 and 0. Although adjacent cells interact through electrostatic forces and tend to align their polarizations, QCA cells do not have intrinsic data flow directionality.

The basic QCA cell consists of four quantum dots in a square array coupled by tunnel barriers. The physical mechanism for interaction between dots is the Coulomb interaction and the quantum-mechanical tunneling. Electrons are able to tunnel between the dots, but they cannot leave the cell. If two mobile electrons are placed in the cell, in the ground state and in the absence of external electrostatic influence, Coulomb repulsion will force the electrons to dots on the opposite corners.

ISSN 2348 – 8034
Impact Factor- 3.155

The Figure 1 shows a simplified diagram of a quantum-dot cell. If the cell is charged with two electrons, each free electron to tunnel to any site in the cell, these electrons will try to occupy the furthest possible site with respect to each other due to mutual electrostatic repulsion. Therefore, two distinguishable cell states exist. Figure 2 shows the two possible minimum energy states of a quantum dot cell. The state of a cell is called its polarization, denoted as P. Although arbitrarily chosen, using cell polarization P = -1 to represent logic "0" and P = +1 to represent logic "1" has become standard practice.

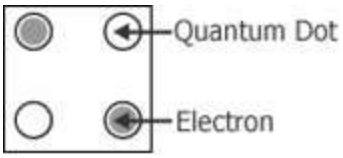

Fig 1: Simplified Diagram of QCA Cell

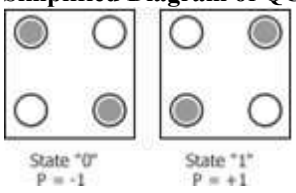

Fig 2: Four Dot Quantum Cell

II. BACKGROUND

A QCA is a nanostructure having as its basic cell a square four quantum dots structure charged with two free electrons able to tunnel through the dots within the cell [1]. Because of Columbic repulsion, the two electrons will always reside in opposite corners. The locations of the electrons in the cell (also named polarizations *P)* determine two possible stable states that can be associated to the binary states 1 and 0. Although adjacent cells interact through electrostatic forces and tend to align their polarizations, QCA cells do not have intrinsic data flow directionality. To achieve controllable data directions, the cells within a QCA design are partitioned into the so-called clock zones that are progressively associated to four clock signals, each phase shifted by 90°. This clock scheme, named the zone clocking scheme, makes the QCA designs intrinsically pipelined, as each clock zone behaves like a D-latch

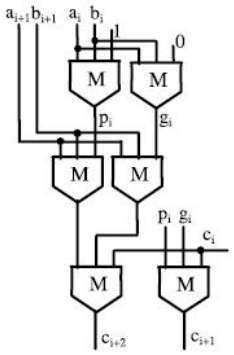

Fig. 3. Novel 2-bit basic module

.

QCA cells are used for both logic structures and interconnections that can exploit either the coplanar cross or the bridge technique The fundamental logic gates inherently available within the QCA technology are the inverter and the MG. Given three inputs *a*, *b,* and *c*, the MG performs the logic function reported in (1) provided that all input cells are associated to the same clock signal clk *x* (with *x* ranging from 0 to 3), whereas the remaining cells of the MG are associated to the clock signal clk*x*+1

$M(abc) = a \cdot b + a \cdot c + b \cdot c.$ (1)

Several designs of adders in QCA exist in literature. The RCA and the CFA process *n*-bit operands by cascading *n* full-adders (FAs). Even though these addition circuits use different topologies of the generic FA, they have a carry-in to carry-out path consisting of one MG, and a carry-in to sum bit path containing two MGs plus one inverter. As a consequence, the worst case computational paths of the *n*-bit RCA and the *n*-bit CFA consist of (*n*+2) MGs and

ISSN 2348 – 8034
Impact Factor- 3.155

one inverter. A CLA architecture formed by 4-bit slices was also presented in . In particular, the auxiliary propagate and generate signals, namely $pi = ai + bi$ and $gi = ai \cdot bi$, are computed for each bit of the operands and then they are grouped four by four. Such a designed n-bit CLA has a computational path composed of $7 + 4 \times (\log4\ n)$ cascaded MGs and one inverter. This can be easily verified by observing that, given the propagate and generate signals (for which only one MG is required), to compute grouped propagate and grouped generate signals; four cascaded MGs are introduced in the computational path. In addition, to compute the carry signals, one level of the CLA logic is required for each factor of four in the operands word-length. This means that, to process n-bit addends, $\log4\ n$ levels of CLA logic are required, each contributing to the computational path with four cascaded MGs. Finally, the computation of sum bits introduces two further cascaded MGs and one inverter.

The parallel-prefix BKA demonstrated in [13] exploits more efficient basic CLA logic structures. As its main advantage over the previously described adders, the BKA can achieve lower computational delay. When n-bit operands are processed, its worst case computational path consists of $4 \times \log2\ n - 3$ cascaded MGs and one inverter. Apart from the level required to compute propagate and generate signals, the prefix tree consists of $2 \times \log2n - 2$ stages. From the logic equations provided in [13], it can be easily verified that the first stage of the tree introduces in the computational path just one MG; the last stage of the tree contributes with only one MG; whereas, the intermediate stages introduce in the critical path two cascaded MGs each. Finally, for the computation of the sum bits, further two cascaded MGs and one inverter are added.

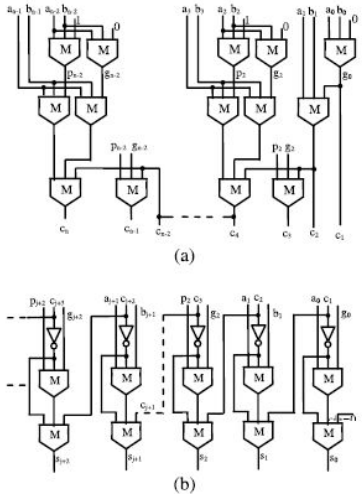

Fig. 4. Novel *n*-bit adder (a) carry chain and (b) sum block.

With the main objective of trading off area and delay, the hybrid adder (HYBA) described in [14] combines a parallel-prefix adder with the RCA. In the presence of n-bit operands, this architecture has a worst computational path consisting of $2\times\log2\ n + 2$ cascaded MGs and one inverter. When the methodology recently proposed in [15] was exploited, the worst case path of the CLA is reduced to $4 \times \_\log4\ n\_ + 2 \times \_\log4\ n\_ - 1$ MGs and one inverter. The above-mentioned approach can be applied also to design the BKA. In this case the overall area is reduced with respect to [13], but maintaining the same computational path. By applying the decomposition method demonstrated in [16], the computational paths of the CLA and the CFA are reduced to $7 + 2\times\log2(n/8)$ MGs and one inverter and to $(n/2) + 3$ MGs and one inverter, respectively.

III. NOVEL QCA ADDER

To introduce the novel architecture proposed for implementing ripple adders in QCA, let consider two n-bit addends $A = an-1, \ldots, a0$ and $B = bn-1, \ldots, b0$ and suppose that for the ith bit position (with $i = n - 1, \ldots, 0$) the auxiliary propagate and generate signals, namely $pi = ai + bi$ and $gi = ai \cdot bi$, are computed. ci being the carry produced at the generic $(i-1)$th bit position, the carry signal $ci+2$, furnished at the $(i+1)$th bit position, can be computed using the conventional CLA logic reported in (2). The latter can be rewritten as given in (3), by exploiting Theorems 1 and *2* demonstrated in [15]. In this way, the RCA action, needed to propagate the carry ci through the two subsequent bit positions, requires only one MG. Conversely, conventional circuits operating in the RCA fashion, namely the RCA and the CFA, require two cascaded MGs to perform the same operation. In other words, an RCA adder designed as proposed has a worst case path almost halved with respect to the conventional RCA and

ISSN 2348 – 8034
Impact Factor- 3.155

CFA. Equation (3) is exploited in the design of the novel 2-bit module shown in Fig. 1 that also shows the computation of the carry $ci+1 = M(pi\ gi\ ci)$. The proposed n-bit adder is then implemented by cascading $n/2$ 2-bit modules as shown in Fig. 2(a). Having assumed that the carry-in of the adder is $cin = 0$, the signal $p0$ is not required and the 2-bit module used at the least significant bit position is simplified. The sum bits are finally computed as shown in Fig. 2(b). It must be noted that the time critical addition is performed when a carry is generated at the least significant bit position (i.e., g0 = 1) and then it is propagated through the subsequent bit positions to the most significant one. In this case, the first 2-bit module computes $c2$, contributing to the worst case computational path with two cascaded MGs. The subsequent 2-bit modules contribute with only one MG each, thus introducing a total number of cascaded MGs equal to $(n - 2)/2$. Considering that further two MGs and one inverter are required to compute the sum bits, the worst case path of the novel adder consists of $(n/2) + 3$ MGs and one inverter

$ci+2 = gi+1 + pi+1 \cdot gi + pi+1 \cdot pi \cdot ci$ (2)

$ci+2 = M(M\_ai+1, bi+1, gi \quad M\_ai+1, bi+1, pi\ ci)$. (3)

1. Logic gates

The logic elements of QCA are an inverter and majority gate. An inverter is designed by positioning cells diagonally from each other to achieve the inversion functionality. A majority gate consists of five QCA cells that realize the function of $M(a; b; c) = ab + bc + ac$. Two-input AND gate and OR gates can be designed by fixing one of the majority gate inputs to ”0” and ”1”, respectively shown as follows.

AND = M(a,b,0)

OR = M(a,b,1)

If one input is set to 0, then the output is the AND of the other two inputs. If one input is set to 1, then the output is the OR of the other two inputs. With ANDs, ORs, and inverters, any logic function can be realized.

Carry-look ahead is arguably the most important technique in the design of fast adders, especially large ones. In straightforward addition, e.g. in a ripple adder, the operational time is limited by the (worst-case) time allowed for the propagation of carries and is proportional to the number of bits added. So faster adders can be obtained by devising a way to determine carries before they are required to form the sum bits. Carry-lookahead does just this, and, in certain cases the resulting adders have an operational time that is independent of the operands' word-length. A carry, *Ci*, is produced at bit-stage *i* if either one is *generated* at that stage or if one is *propagated* from the preceding stage. So a carry is generated if both operand bits are 1, and an incoming carry is propagated if one of the operand bits is 1 and the other is 0. Let *Pi* and *Gi* denote the generation and propagation, respectively, of a carry at stage *i*, Ai and Bi denote the two operands bits at that stage, and Ci-1 denote the carry into the stage. Then we have

Gi=AiBi

Pi = Ai^Bi

Ci = Gi + PiCi-1

and the sum can be written as Si== Pi^Ci-1 which allows the use of shared logic to produce Si and Pi.

C0 = G0 + P0Ci-1

C1 = G1 + P1P0C-1+ P1G0

.

.

.

Ci = Gi + Pi-1Gi-1+PiPi-1Gi-2+…+PiPi-1Pi-2…P0C-1

where Ci-1 is the carry into the adder. The equation for Ci states that there is a carry from stage i if there is a carry generated at stage i, or if there is a carry that is generated at stage i-1 and propagated through stage i or if , or if the initial carry-in, Ci-1, is propagated through stages 0,1,… i. The complete set, of equations show that, in theory at least, all the carries can be determined independently, in parallel, and in a time (three gate delays) that is independent of the number of bits to be added. The same is also therefore true for all the sum bits, which require only one additional gate delay.

ISSN 2348 – 8034
Impact Factor- 3.155

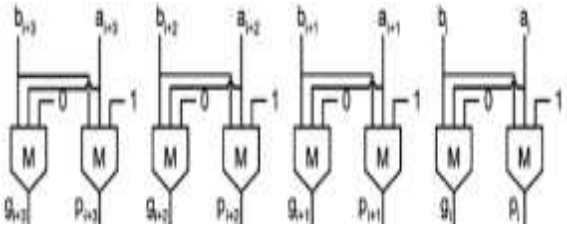

Fig 5: Generation of propagate and generate bits

Gi = AiBi
Pi = Ai^Bi

Compared with a ripple adder, as well as some of the other adders, a pure carry-look ahead adder has high logic costs. Furthermore, high fan-in and fan-out requirements can be problematic: the fan-out required of the Gi and Pi signals grows rapidly with n, as does the fan-in required to form Ci. For sufficiently large values of *n*, the high fan-in and fan-out requirements will result in low performance, high cost, or designs that simply cannot be realized.

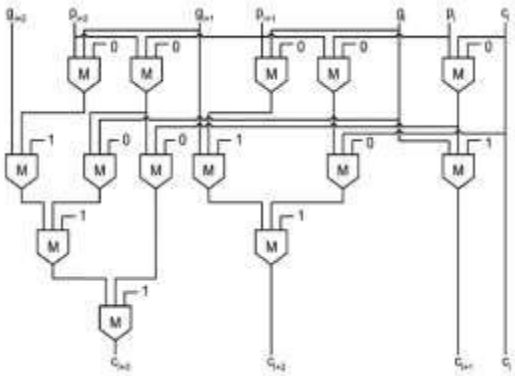

Fig 6: Carry block

This carry block is cascaded with the propagate and generate block. So, that carry is obtained with the following equation.
Ci = Gi + PiCi-1

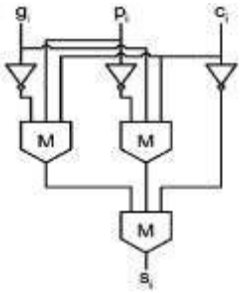

Fig 7: Sum block

This sum block is cascaded with the above carry block to obtain the sum. The following equation gives the sum bit Si = Pi ^ Ci-1.
The following shows the carry block which generate the carry bits.

ISSN 2348 – 8034
Impact Factor- 3.155

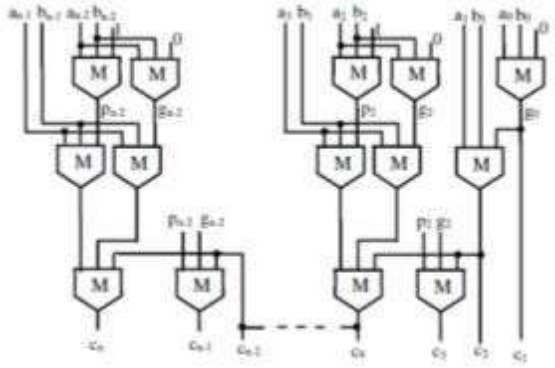

Fig8: Carry block

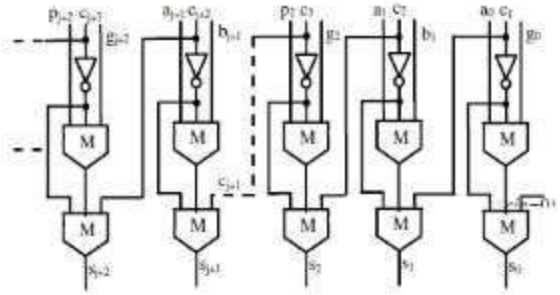

Fig 9: Sum block

The above carry block is cascaded with the sum block which generate the sum bits. The following are the equations for the carry bits and the sum bits.
Ci+2= M(M(ai+1,bi+1,gi)M(ai+1,bi+1,pi)ci)
For sum block:
For odd
Sj+1=M(~Cj+3M(aj+2,~Cj+3,bj+2),Cj+2)
For even
Sj+2=M(~Cj+3M(Pj+2,~Cj+3,Gj+2),Cj+2)

IV. RESULTS

The proposed addition architecture is implemented for several operands word lengths using the QCA Designer tool adopting the same rules and simulation settings used in [11]–[16]. The QCA cells are 18-nm wide and 18-nm high; the cells are placed on a grid with a cell center-to-center distance of 20 nm; there is at least one cell spacing between adjacent wires; the quantum-dot diameter is 5 nm; the multilayer wire crossing structure is exploited; a maximum of 16 cascaded cells and a minimum of two cascaded cells per clock zone are assumed. Layouts for the 16-, 32- and 64-bit versions of the novel adder are shown in Figs. 3–5, respectively. Simulation results for the 64-bit adder is shown in Fig. 6. There, the carry out bit is included in the output sum bus. Because of the limited QCA Designer graphical capability, input and output busses are split into two separate more significant and less significant busses. Fig. 6 also shows the polarization values of few single output signals (i.e., *sum*64, *sum*32, *sum*31, and *sum*). Simulations performed on 32- and 64-bit adders have shown that the first valid result is outputted after five and nine latency clock cycles, respectively.

ISSN 2348 – 8034
Impact Factor- 3.155

Fig. simulation result of qca 128 bit

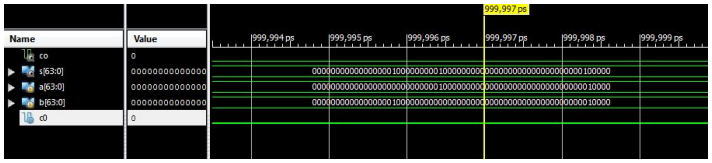

Fig. simulation result of qca 64 bit

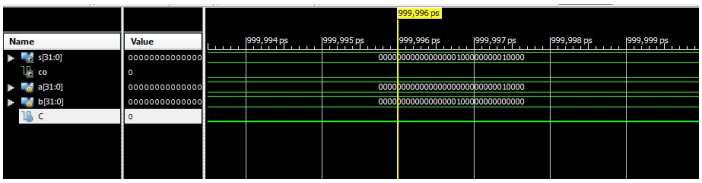

Fig. simulation result of qca 32 bit

Fig. simulation result of qca 16 bit

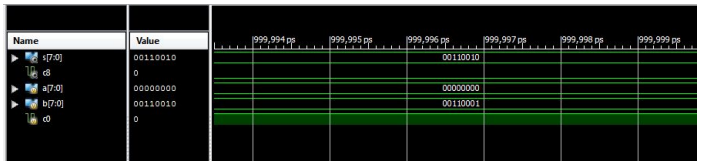

Fig. simulation result of qca 8 bit

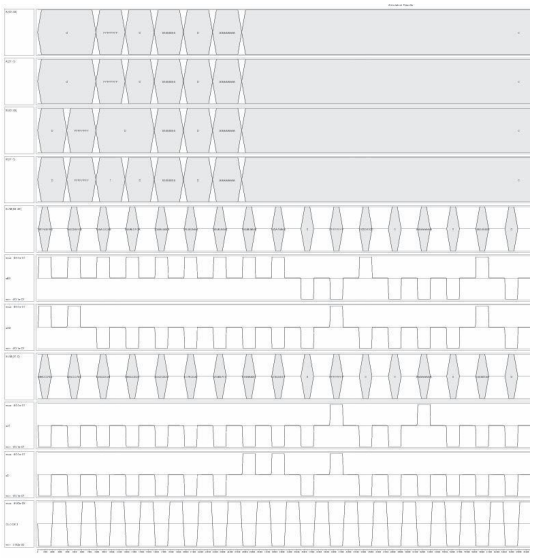

Fig. 10. Simulation results obtained for the novel 64-bit adder.

ISSN 2348 – 8034
Impact Factor- 3.155

V. CONCLUSION

A new adder designed in QCA was presented. It achieved speed performances higher than all the existing QCA adders, with an area requirement comparable with the cheap RCA and CFA demonstrated in [13] and [16]. The novel adder operated in the RCA fashion, but it could propagate a carry signal through a number of cascaded MGs significantly lower than conventional RCA adders. In addition, because of the adopted basic logic and layout strategy, the number of clock cycles required for completing the elaboration was limited. A 64-bit binary adder designed as described in this brief exhibited a delay of only nine clock cycles, occupied an active area of 18.72 μm2, and achieved an ADP of only 168.48.

REFERENCES

[1] C. S. Lent, P. D. Tougaw, W. Porod, and G. H. Bernestein, "Quantum cellular automata," Nanotechnology, vol. 4, no. 1, pp. 49–57, 1993.
[2] M. T. Niemer and P. M. Kogge, "Problems in designing with QCAs: Layout = Timing," Int. J. Circuit Theory Appl., vol. 29, no. 1, pp. 49–62, 2001.
[3] J. Huang and F. Lombardi, Design and Test of Digital Circuits by Quantum-Dot Cellular Automata. Norwood, MA, USA: Artech House, 2007.
[4] W. Liu, L. Lu, M. O'Neill, and E. E. Swartzlander, Jr., "Design rules for quantum-dot cellular automata," in Proc. IEEE Int. Symp. Circuits Syst., May 2011, pp. 2361–2364.
[5] K. Kim, K. Wu, and R. Karri, "Toward designing robust QCA architectures in the presence of sneak noise paths," in Proc. IEEE Design, Autom. Test Eur. Conf. Exhibit., Mar. 2005, pp. 1214–1219.
[6] K. Kong, Y. Shang, and R. Lu, "An optimized majority logic synthesis methology for quantum-dot cellular automata," IEEE Trans. Nanotechnol., vol. 9, no. 2, pp. 170–183, Mar. 2010.
[7] K. Walus, G. A. Jullien, and V. S. Dimitrov, "Computer arithmetic structures for quantum cellular automata," in Proc. Asilomar Conf. Sygnals, Syst. Comput., Nov. 2003, pp. 1435–1439.
[8] J. D. Wood and D. Tougaw, "Matrix multiplication using quantumdot cellular automata to implement conventional microelectronics," IEEE Trans. Nanotechnol., vol. 10, no. 5, pp. 1036–1042, Sep. 2011.
[9] K. Navi, M. H. Moaiyeri, R. F. Mirzaee, O. Hashemipour, and B. M. Nezhad, "Two new low-power full adders based on majority-not gates," Microelectron. J., vol. 40, pp. 126–130, Jan. 2009.
[10] L. Lu, W. Liu, M. O'Neill, and E. E. Swartzlander, Jr., "QCA systolic array design," IEEE Trans. Comput., vol. 62, no. 3, pp. 548–560, Mar. 2013.
[11] H. Cho and E. E. Swartzlander, "Adder design and analyses for quantum-dot cellular automata," IEEE Trans. Nanotechnol., vol. 6, no. 3, pp. 374–383, May 2007.
[12] H. Cho and E. E. Swartzlander, "Adder and multiplier design in quantum-dot cellular automata," IEEE Trans. Comput., vol. 58, no. 6, pp. 721–727, Jun. 2009.
[13] V. Pudi and K. Sridharan, "Low complexity design of ripple carry and Brent–Kung adders in QCA," IEEE Trans. Nanotechnol., vol. 11, no. 1, pp. 105–119, Jan. 2012.
[14] V. Pudi and K. Sridharan, "Efficient design of a hybrid adder in quantumdot cellular automata," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 19, no. 9, pp. 1535–1548, Sep. 2011.
[15] S. Perri and P. Corsonello, "New methodology for the design of efficient binary addition in QCA," IEEE Trans. Nanotechnol., vol. 11, no. 6, pp. 1192–1200, Nov. 2012.
[16] V. Pudi and K. Sridharan, "New decomposition theorems on majority logic for low-delay adder designs in quantum dot cellular automata," IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 59, no. 10, pp. 678–682, Oct. 2012.
[17] K. Walus and G. A. Jullien, "Design tools for an emerging SoC technology: Quantum-dot cellular automata," Proc. IEEE, vol. 94, no. 6, pp. 1225–1244, Jun. 2006.
[18] S. Bhanja, M. Ottavi, S. Pontarelli, and F. Lombardi, "QCA circuits for robust coplanar crossing," J. Electron. Testing, Theory Appl., vol. 23, no. 2, pp. 193–210, Jun. 2007.
[19] A. Gin, P. D. Tougaw, and S. Williams, "An alternative geometry for quantum dot cellular automata," J. Appl. Phys., vol. 85, no. 12, pp. 8281–8286, 1999.
[20] A. Chaudhary, D. Z. Chen, X. S. Hu, and M. T. Niemer, "Fabricatable interconnect and molecular QCA circuits," IEEE Trans. Comput. Aided Design Integr. Circuits Syst., vol. 26, no. 11, pp. 1977–1991, Nov. 2007.
[21] M. Janez, P. Pecar, and M. Mraz, "Layout design of manufacturable quantum-dot cellular automata," Microelectron. J., vol. 43, no. 7, pp. 501–513, 2012.