Design of the Best Area-delay Adders with QCA Majority Logic Gates by using VHDL

K.Vijaya Krishna[1], R.Radha Krishna[2]

[1]M.Tech Student, ECE Department, Indur Institute of Engineering&Technology, Siddipet, Pin no.502 277
[2] Associate Professor, ECE Department, Indur Institute of Engineering&Technology, Siddipet, Pin no.502 277

ABSTRACT

The quantum-dot cellular automata (QCA) are the New Technology at Nano metric scale, which is having more advantages like Less Space and Less Power Consumption. , Transistors cannot find much lesser than their current size .The QCA logic approach represents one of the possible solutions in overcoming this physical edge, level. Using this QCA technology we developed AND gate ,OR gate and NOT gate. With the help of Universal gates we can implement any Digital Logic Function, In this paper we have realized different high speed adders like Ripple Carry Adders(RCA) , Carry Look ahead adder(CLA), Brent-Kung adder(BKA), Kogge-Stone Adder(KSA) and Ling Adder by using Majority gates(MG) that out performs all state-of-the art competitors and achieves the best area-delay tradeoff. We finally compared the different properties like delay (speed), power consumption, area, ADP and PDP.

Keywords *- Adders, quantum-dot cellular automata (QCA), Delay, power consumption, Majority gates (MG)*

1. INTRODUCTION

The Quantum-dot cellular automata (QCA) are an attractive emerging technology suitable for the development of ultra-dense low-power high-performance digital circuits. For this reason, in the last few years, the design of efficient logic circuits in QCA has received a great deal of attention. Special efforts are directed to arithmetic circuits, with the main interest focused on the binary addition that is the basic operation of any digital system. Of course, the architectures commonly employed in traditional CMOS designs are considered a first reference for the new design environment. Ripple-carry (RCA), carry look-ahead (CLA), and conditional sum adders were presented in[11] . The carry-flow adder (CFA) shown in Fig. was mainly an improved RCA in which detrimental wires effects were mitigated. Parallel prefix architectures, including Brent–Kung (BKA), Kogge–Stone, and Han–Carlson adders, were analyzed and implemented in QCA. Recently, more efficient designs were proposed for the CLA and the BKA, and for the CLA and the CFA. In this brief, an innovative technique is presented to implement high-speed low-area adders in QCA. Theoretical formulations demonstrated for CLA and parallel-prefix adders are here exploited for the realization of a novel 2-bit addition slice. The latter allows the carry to be propagated through two subsequent bit-positions with the delay of just one majority gate (MG). In addition, the clever top level architecture leads to very compact layouts, thus avoiding unnecessary clock phases due to long interconnections. An adder designed as proposed runs in the RCA fashion, but it exhibits a computational delay lower than all state-of the-art competitors and achieves the lowest area-delay product (ADP).

2. QCA TECHNOLOGY

Quantum-dot cellular automata (QCA) is a square nanostructure of electron wells having free electrons. Each cell has four quantum dots. The four dots are located in the four corners .The cell can be charged with two free electrons. By using the clocking mechanism, the electrons tunnel to proper location during the clock transition. Thus there exist two equivalent energetically arrangements of the two electrons in the QCA cell as shown in Fig.2.3 .These two arrangements can represent logic 1 and logic 0 respectively so that binary information can be encoded. Inverter is represented in Fig 2.1. and Majority gate in Fig 2.2.

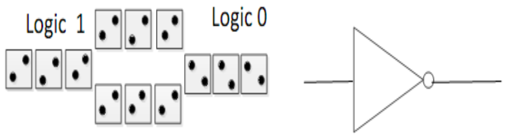


Fig.2.1: Inverter

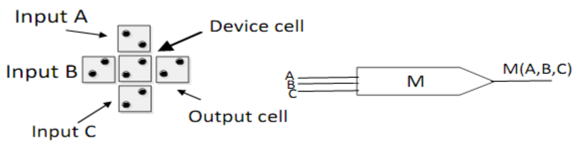


Fig.2.2: Majority Gate

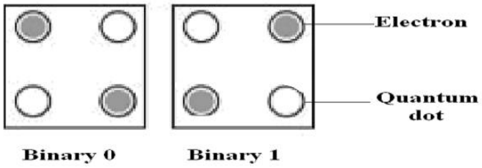


Fig.2.3: QCA Cells with Electrons and Quantum Dot

3. PARALLEL PREFIX ADDER

The Parallel prefix adder is the most flexible and widely used for binary addition. Parallel Prefix adders are best suited for VLSI implementation. Numbers of parallel prefix adder structures have been proposed over the past years intended to optimize area, fan-out, logic depth and inter connect count.

Binary addition is the most fundamental and frequently used arithmetic operation. A lot of work on adder design has been done so far and much architecture has been proposed. When high operation speed is required, tree structures like parallel-prefix adders are used. Parallel-prefix structures are found to be common in high performance adders because of the delay is logarithmically proportional to the adder width.

PPA's basically consists of 3 stages

- Pre computation
- Prefix stage
- Final computation

The Parallel-Prefix Structure is shown in fig:3.1.

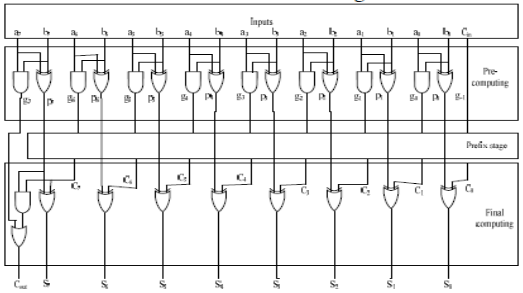


Fig.3.1: Parallel-Prefix Structure with carry save notation

4. RIPPLE-CARRY ADDER

A quantum-dot cellular automaton (QCA) is an attractive emerging technology suitable for the development of ultra dense low-power higher performance digital circuits. For this reason in the final few years, the design of capable logic circuits, QCA has received a great deal of attention. Particular efforts are going to arithmetic circuits, with the major interest focused on the binary addition that is the basic operation of any digital system. Of course, the designs commonly working in traditional CMOS designs are considered a first reference for the new design environment. Ripple-carry, carry look-ahead (CLA), and conditional sum adders were implemented. The carry-flow adder shown in was mainly an improved RCA in which damaging wires effects were mitigated. Parallel-prefix architectures, as well as Brent–Kung, Kogge–Stone, Ladner–Fischer, and Han–Carlson adders, were analyzed and implemented in QCA. Recently, further efficient designs were proposed for the CLA and the BKA, and for the Carry look ahead Adder and the Carry Flow Adder. In this brief, an original technique is presented to implement high speed low-area adders into QCA. Theoretical formulations established for CLA and parallel-prefix adders are here exploited for the realization of a novel 8-bit addition portion. The former allows the carry to be propagated through two subsequent bit-positions with the delay of just one majority gate (MG). In addition, the clever top level architecture leads to very compact layouts in, thus avoiding unnecessary clock phases due to lengthy interconnections

It is possible to create a logical circuit using multiple full adders to add N-bit numbers. Each full adder inputs a Cin, which is the Cout of the previous adder. This kind of adder is called a ripple-carry adder, since each carry bit "ripples" to the next full adder. Note that the first (and only the first) full adder may be replaced by a half adder (under the assumption that Cin = 0).

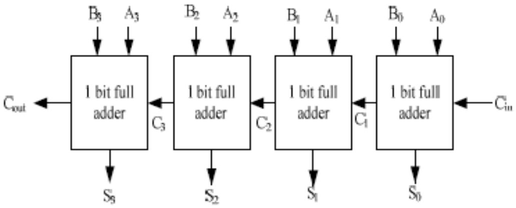


Fig.4.1: 4bit ripple carry adder

The layout of a ripple-carry adder is simple, which allows for fast design time; however, the ripple-carry adder is relatively slow, since each full adder must wait for the carry bit to be calculated from the previous full adder. The gate delay can easily be calculated by inspection of the full adder circuit. Each full adder requires three levels of logic.

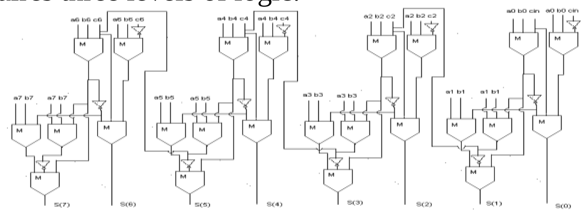


Fig.4.2: Ripple Carry Adder Using Majority Gate

5. CARRY LOOKAHEAD ADDER (CLA)

QCA cells are used for both logic structures and interconnections that can develop either the coplanar cross or the bridge technique. The primary logic gates naturally available within the QCA technology are the not and the majority gate. Given three inputs a, b, and c, the majority gate performs the logic function reported in (2) provided that all input cells are connected to the same clock signal clk x (with x ranging from 0 to 3), whereas the remaining cells of the MG are linked to the clock signal clkx+1

$$M(a,b,c) = a\cdot b+a\cdot c+b\cdot c. \ldots\ldots\ldots\ldots\ldots.. (2)$$

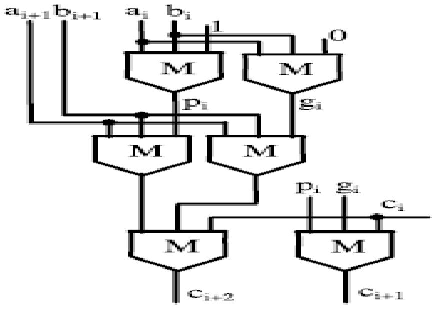
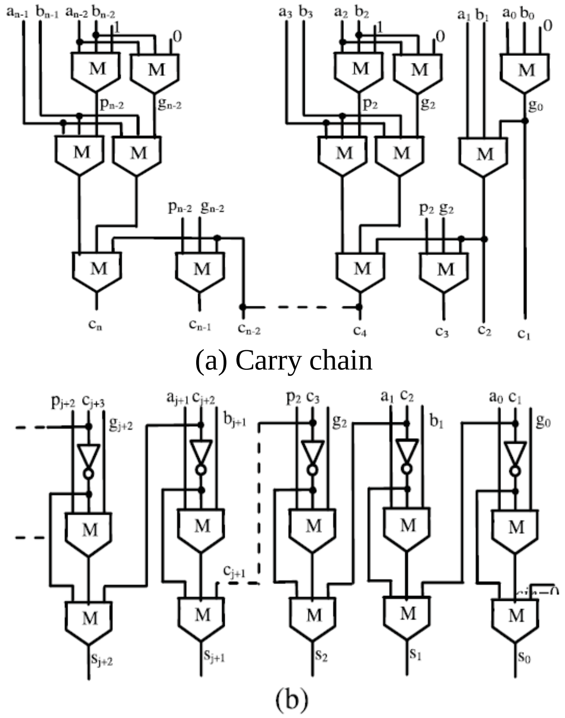


Fig. 5.1: Novel 2-bit basic module

a number of designs of adders in QCA exist in text. The RCA and the CFA process n-bit operands by cascading n full-adders (FAs). Even while these addition circuits use different topologies of the generic full adder, they have a carry input to carryout put path consisting of one majority gate, and a carry-in to sum bit path containing two MGs plus one inverter. As an importance, the worst case computational paths of the n-bit RCA and the n-bit CFA consist of (n+2) MGs and one inverter. A CLA architecture created by 4-bit slices was also presented. In particular, the auxiliary generated to and generate signals, namely pi = ai + bi and gi = ai · 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 collected of 7+4× (log4 n) cascaded MGs and single not gate. This can be easily verified by observe that, given the spread 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 necessary for each factor of four in the operands length. This means that, to process n number of 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, exploits more efficient basic CLA logic structures. As its main advantage over the previously described adders in the BKA can achieve lower computational delay. When n-bit operands are processed, its most terrible case computational path consists of 4 × 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×log2n−2 stages. From the logic equations provided, 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 middle stages introduce in the critical path two cascaded MGs each. Finally, for the computation of the addition bits, added two cascaded MGs and one inverter are added.



(a) Carry chain

(b)

Fig 5.2: Novel n-bit adder (b) Sum block

With the main point of trading off area and delay, the hybrid adder (HYBA) described and combines a parallel-prefix adder with the Ripple carry adder. In the attendance of n-bit operands, this structural design has a most terrible computational path consisting of 2×log2 n+2 cascaded MGs and one inverter. When the methodology recently proposed and was exploited, the worst case path of the CLA is reduced to 4 × _log4 *n*_ + 2 ×_log4 *n*_ − 1.By applying the decomposition method demonstrated the computational paths of the CLA and the CFA are reduced to 7 + 2×log2 (n/8) MGs and one not gate and to (n/2)+3 MGs and one inverter, respectively.

6. KOGGE-STONE ADDER (KSA)

KSA is a parallel prefix form carry look ahead adder. It generates carry in O (log n) time and is generally considered as the fastest adder and is widely used in the industry for high performance arithmetic circuits. In KSA, carries are computed fast by computing them in parallel at the cost of increased area. The complete performance of KSA can be easily comprehended by analyzing it in terms of three distinct parts

1. Pre processing: This step involves computation of generate and propagate signals consequent too each pair of bits in A and B. These signals are specified by the logic equations below:

$$p_i = A_i \text{ xor } B_i$$
$$g_i = A_i \text{ and } B_i$$

2. Carry look ahead network: This block differentiates KSA from other adders and is the main force behind its high performance. It uses group propagate and create as intermediate signals.

$$p = p_i \text{ and } p_{i\ prev}$$
$$s = (p_i \text{ and } g_{i\ prev}) \text{or } g_i$$

3. Post processing: This is the final step and is com -mon to all adders of this family (carry look ahead). It involves calculation of sum bits. Sum bits are comp- uted by the reason given below.

$$s_i = p_i \text{ xor } c_{i-1}$$
$$c_i = g_i$$

The Kogge–Stone adder is a parallel prefix form carry look-ahead adder. It generates the carry signals in O(log n) time, and is widely considered the fastest adder design possible. It is the common design for high-performance adders in industry. It takes more area to implement than the Brent–Kung adder, but has a lower fan-out at each stage, which increases performance. Wiring congestion is often a problem for Kogge–Stone adders as well.

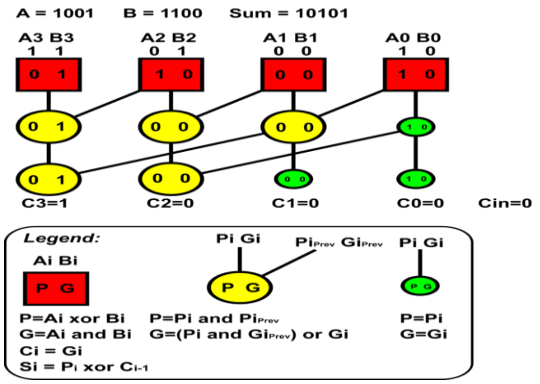


Fig.6.1: Example of a 4-bit Kogge–Stone adder with zero carry-in

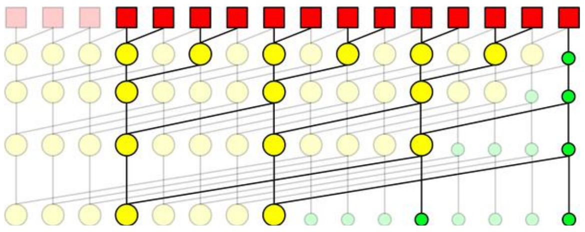


Fig 6.2:an example of a Kogge–Stone adder with sparsity-4.

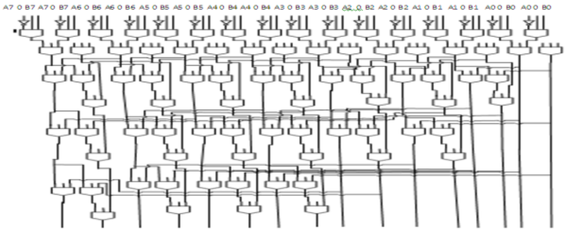


Fig.6.3: 8 bit kogge-stone adder using Majority Gates

7. BRENT-KUNG ADDER (BKA)

The parallel prefix adders are more flexible and are used to speed up the binary add-ons. Parallel prefix adders are obtained from Carry Look Ahead (CLA) structure.

The construction of parallel prefix adder involves three stages 1. Pre-processing stage 2. Carry generation network 3. Post processing.

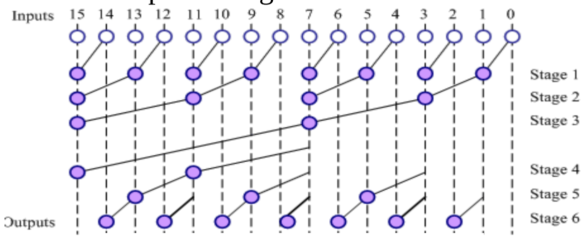


Fig.7.1: 16 bit Brent kung adder

The Brent-Kung adder is a parallel prefix adder. Parallel prefix adders are special class of adders that are based on the use of generate and propagate signals. Simpler Brent -Kung adders was been proposed to solve the disadvantages of Kogge-Stone adders. The cost and wiring complexity is greatly reduced. But the logic depth of Brent-Kung adders increases to 2log (2n-1), so the speed is lower. The block diagram of 16-bit Brent-Kung adder is shown in Fig 7.1

Using the dot operator explained above the Equation can be written for the different carry signals as

$$Co,0 = G0 + P\ Ci,0 = a\ (\ G0,P0)$$
$$Co,1 = G1 + G0\ P1 = a\ ((G1\ ,\ P1\ )\cdot(G0,\ P0))$$
$$C0,k = a\ ((Gk,Pk)\cdot(Gk\text{-}1,Pk\text{-}1)\cdot\ldots\cdot(G0,P0))$$

.

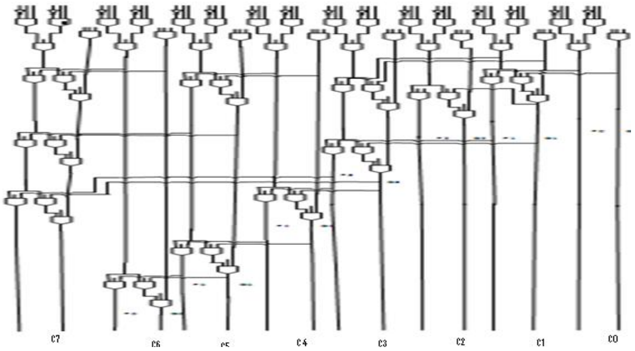


Fig.7.2: 8-bit Brent Kung Structure using majority gates

8. LING ADDER

Ling introduced the pseudo carry notion and showed that the delay in the carry path could be reduced by forming the pseudo-carry term ($Hn:0 = gn + Gn-1:0$). Ling's pseudo carry is less complex than the $Gn:0$ as the fan-in of each AND gate to compute Ling pseudo carry gets reduced by one. Thus, $Hn:0$ is easier to implement than $Gn:0$.Ling also showed that recursion can be applied to produce group pseudo carry by hierarchically combining the pseudo carries over sub-groups in a similar way as in a prefix adder. This means that the pseudo carry $Hj:i$ of a group from position j to position i can be constructed from the pseudo carries $Hj:k$ and $Hk-1:i$ of two subgroups formed one from position j to k and other from position $k - 1$ to i, respectively. $Hj:i = Hj:k + \neg Kj-1:k-1Hk-1:i$ Ling adder using valency-2 at its first level of carry computation will have the equation for group generate as $Hn:n-1 = gn + gn-1 = (an \bullet bn) + (an-1 \bullet bn-1)$.

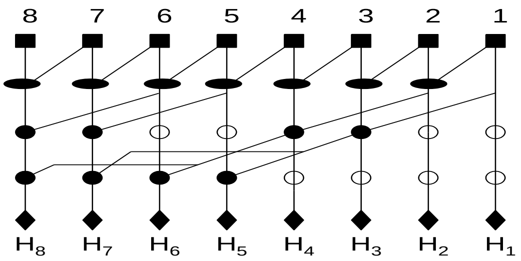


Fig.8.1: an example of a 8 bit ling adder

9. RESULTS

Output waveforms

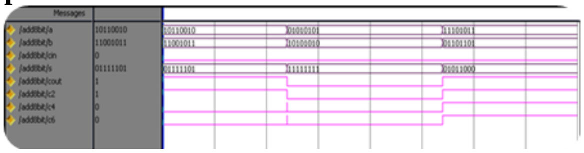


Fig.9.1: shows ripple carry adder output waveforms

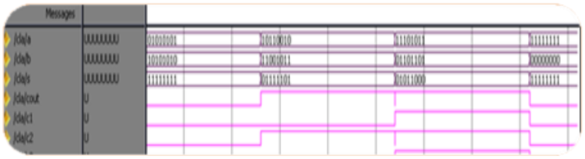


Fig.9.2: shows carry look ahead adder output waveforms

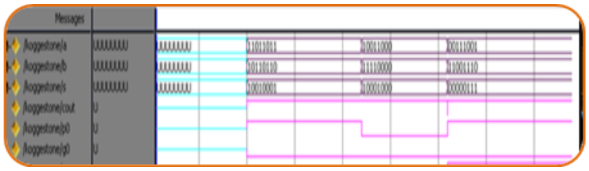


Fig. 9.3: shows kogge stone adder output waveforms

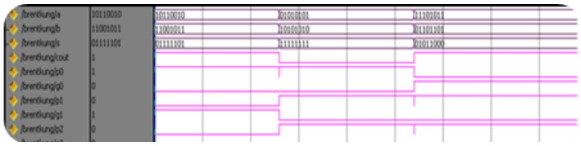


Fig.9.4:shows Brent Kung adder output waveforms

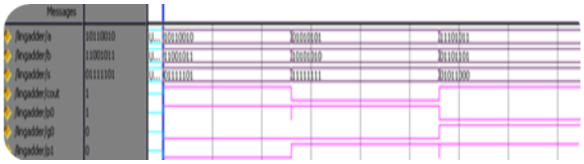
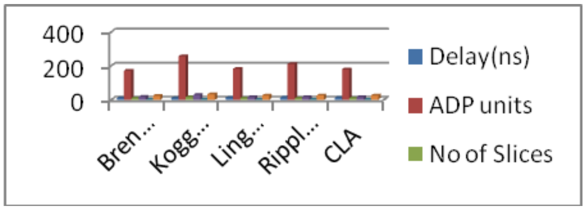


Fig.9.5 shows Ling adder output waveforms

10. CONCLUSIONS AND FUTURE SCOPE

Table 10.1: Comparison of Different Adders

Adder	n	Delay	ADP units	No .of Slices	No. of LUT's	Power(W)	PDP units
Brent Kung	8	10.045ns	170.765	9	17	0.1307	22.3189
Kogge Stone	8	9.154ns	256.312	16	28	0.1390	35.6273
Ling Adder	8	12.142ns	182.13	9	15	0.1305	23.7679
Ripple Carry	8	13.169ns	210.704	9	16	0.1306	27.5179
CLA	8	11.947ns	179.205	9	15	0.1305	23.3862



In this paper, we have realized different high speed adders like Brent Kung, Kogge stone, Ling, Ripple carry adder and Carry look ahead adder of 8 bit. It is implemented with only QCA Majority gate logic. which is a latest approach, we have implemented, simulated, synthesized all the VHDL codes on Spartan 3E FPGA board of number XC3S500e-5cp132.We have observed that Kogge stone is having Less Delay more area. It is only suitable where area is not constraint, If area is less and Delay is not constraint Ling Adder is suitable. Out of all adders Brent Kung adder is having good Area Delay Product (ADP), and Power Delay Product (PDP), so we are concluding that Brent kung adder implementation using QCA Majority gate is more Area, Power, and Delay efficient. Our work can be extended for 16 bit, 32 bit and 64 bit also for further analysis.

REFERENCES

[1] Stefania Perri,Pasquale Corsonello,and Giuseppe Cocorullo. "Area-Delay Efficient Binary Adders in QCA"IEEE Trans. Very Large Scale Integration (VLSI) Syst.,vol. 22, no. 5, , pp. 1174–1179, May 2014.

[2] L. Lu, W. Liu, M. O'Neill, and E. E. Swartz lander, Jr., QCA systolic array design," IEEE Trans. Comput., vol. 62, no. 3, pp. 548–560,Mar. 2013.

[3] S. Perri and P. Corsonello, “New metho dology for the design of efficient binary addition in QCA,” IEEE Trans.Nanotechnol., vol. 11, no. 6,pp. 1192–1200, Nov. 2012.

[4] V. Pudi and K. Sridharan, “New Decom Posi -tion 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.

[5] V. Pudi and K. Sridharan, “Low complexity design of ripple carry and Brent–Kung adder in QCA,” IEEE Trans. Nanotechnol., vol. 11, no. 1, pp. 105–119, Jan. 2012.

[6] M. Janez, P. Pecar, and M. Mraz, “Layout design of manufacturable quantum-dot cellular automata,” Microelectron. J., vol. 43, no. ,pp. 501–513, 2012.

[7] V. Pudi and K. Sridharan, “Efficient design of a hybrid adder in quantum dot cellular automata,” IEEE Trans. Very Large Scalentegr (VLSI) Syst.,vol. 19, no. 9, pp. 1535–1548, Sep. 2011.

[8] J.D.Wood and D.Tougaw, “Matrix multi- plication using quantum dot cellular automata to implement conventional microelectronics,” IEEE Trans. Nanotechnol., vol. 10, no. 5, pp. 1036–1042, Sep. 2011.

[9] W. Liu, L. Lu, M. O’Neill, and E. E. Swartz lander, Jr., “Design rules for quantum-dot cellular automata,” in Proc. IEEE Int. Symp. Circuits Syst., May 2011, pp. 2361–2364.

[10] 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.

[11] 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.

[12] S. Bhanja, M. Ottavi, S. Pontarelli, and F. Lom - bar-di, “QCA circuits for robust coplanar crossing,” J. Elec-tron. Testing, Theory Appl., vol. 23, no. 2, pp. 193–210, Jun. 2007.

[13] 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.

[14] J. Huang and F. Lombardi, Design and Test of Digital Circuits by Quantum -Dot Cellular Automata. Norwood, MA, USA: Artech House,2007.

[15] 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.

[16] 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. Ex-hibit., Mar. 2005, pp. 1214–1219.

[17] A. Gin, P. D. Tougaw, and S. Williams, “An alterna-tive geometry for quantum dot cellular automata,” J. Appl. Phys., vol. 85, no. 12,pp. 8281–8286, 1999.

[18] C. S. Lent, P. D. Tougaw, W. Porod, and G. H. Bernestein, “Quantum cellular automata,” Nano- technology, vol. 4, no. 1, pp. 49–57, 1993.