FACTA UNIVERSITATIS (NIŠ)
SER.: ELEC. ENERG. vol. 20, no. 3, December 2007, 541-560

Binary Multipliers on Quantum-Dot Cellular Automata

Ismo Hänninen and Jarmo Takala

Abstract: This article describes the design of ultra-low-power multipliers on quantum-dot cellular automata (QCA) nanotechnology, promising very dense circuits and high operating frequencies, using a single homogeneous layer of the basic cells. We construct structures without the earlier noise problems, verified by the QCADesigner coherence vector simulation. Our results show that the wiring overhead of the arithmetic circuits grows quadratically with the operand word length, and our pipelined array multiplier has linearly better performance-area efficiency than the previously proposed serial-parallel structure. Power analysis at the fundamental Landauer's limit shows, that the operating frequencies will indeed be bound by the energy dissipated in information erasure: under irreversible operation, the limits for the clock rates on molecular QCA are much lower, than the switching speeds of the technology.

Keywords: Low-power design, nanotechnology, quantum-dot-cellular automata, arithmetic, multiplier, reversible computing.

1 Introduction

Traditional digital circuit technologies like complementary metal oxide semiconductor (CMOS) are reaching their practical and theoretical limits, as the continuous down-scaling of electronics becomes more difficult. Most technologies use transistors as current switches, representing binary information as currents and voltages, but the devices have several problems when they get small: the on/off levels become inadequate, the resistance high, the charge quantized, and the wires very large in comparison. The most severe problem is the heat generation, as the circuit capacitances are charged to a potential and again discharged to ground, usually dissipating nearly all the energy contained in the logic signal. At molecular device

Manuscript received September 25, 2007.

The authors are with the Department of Computer Systems, Tampere University of Technology, Korkeakoulunkatu 1, 33720 Tampere, Finland (e-mail: [ismo.hanninen, jarmo.takala]@tut.fi).

densities and operational frequencies of hundreds of gigahertz, a transistor moving even only *one* electron across one volt potential, the power densities reach the *megawatt* range per square centimeter. Careful adiabatic charging can lower the power dissipation, but not enough to enable true molecular electronics. [1]

Quantum-dot cellular automata (QCA) is a promising nanotechnology, offering circuit densities and clock frequencies several decades higher than the technological peak of the CMOS. The concept was introduced in early 1990s [2, 3] and has been demonstrated in laboratory environment [4–6]. However, adopting it into general use requires advances both in manufacturing and designing methodology due to extremely small feature size and the revolutionary operating principle: encoding binary information into the local position of particles, instead of currents and voltages. The key energetic benefit is that there is no need to repeatedly dissipate most of the signal energy, which in all systems has to be large enough to overcome the thermal noise floor. QCA is not quantum computing, since the information is contained in classical degrees of freedom, instead of the superposition of states, and the quantum effects are used only to enable switching.

In this paper, we design and evaluate feasible multiplier units, which avoid the serious noise coupling problems inherent to the nanotechnology [7, 8], and we aim to minimize the area and performance penalty paid for this robustness. The paper is based on our earlier work on basic adders [9] and an array multiplier [10], which is now extended with a more thorough analysis of the two known multiplier proposals for the technology. Our results show that the arithmetic units on QCA have a quadratic wiring overhead, which in the case of the serial-parallel multiplier even dominates nearly 90% of the area. The performance of the massive array multiplier excels over the much smaller serial-parallel structure (as expected) but when the throughput is normalized with the area cost of the complete unit, the array has also the best relative area efficiency, getting even linearly better with growing word length. Thus the array multiplier gives the cheapest multiplication results.

The power characteristics of the designs are analyzed in respect to the ultimate physical limit of computation, the Landauer's principle, stating the inevitable energy cost of discarding information [11]. The bit erasure dissipation is expected to be the dominant power component in molecular QCA technologies, which can reach operating frequencies in the terahertz regime [12]. We show that the maximum clock frequencies of the multipliers will indeed be limited by this law of nature, under standard irreversible operation, and of our implementations, the array reaches higher clock frequencies than the serial-parallel unit.

This article is organized as follows: In Section 2, related work is summarized, while section 3 covers the background of QCA nanotechnology. Section 4 describes the practical implementation of the serial-parallel multiplier, and section 5

our pipelined array multiplier. Section 6 describes the verification of the designs, while section 7 presents analysis of the area, section 8 the performance, and section 9 the power issues. Section 10 concludes, including issues for future work.

2 Related Work

There has been a considerable amount of research into circuit and systems design for QCA, aiming to shorten the time required to master this sunrise technology, and even to guide its development with knowledge of actual performance and cost metrics. Previously proposed arithmetic circuits for QCA include adders [7, 9, 13–16], a shifter [17], a comparator [18], and serial-parallel multipliers [13, 19]. There are no designs with a systolic structure, which would match the signal locality requirement of the nanotechnology.

A serious problem with most previous designs, including the serial-parallel multipliers, is the recently discovered noise coupling mechanism, which has tremendous effect on the cost and performance of arithmetic units. The phenomena have to be dealt with at very low level, but they have large consequences on the higher design levels. This has not been analyzed in the previous designs, and the failure to address the issue results in unreliable operation. [7, 8]

We have proposed a pipelined array multiplier in [10], the first systolic structure covering the noise problem, and we design also a practical version of the serial-parallel multiplier (carry delay multiplier in [19]). Establishing the key parameters of QCA arithmetic, we describe the implementations, and present analysis showing that the array multiplier has the best performance, produces the multiplications with the smallest relative cost, and reaches the highest clock frequencies.

Previous power analyses on QCA concentrate on the energetic characteristics of the technology and the primitive elements [12, 20–24], while, in this paper, we consider the effects in larger systems and at architectural level. Our study connects the structure of arithmetic units to the inevitable dissipation in the actual circuit layouts, and predicts the limits of performance and operating frequencies.

3 Background of QCA Nanotechnology

The QCA concept is very intuitive: we have bistable cellular automata, which are operated under clocked control. There are various ways to construct the physical cells and apply the clocking, but the implementation technologies are still under development. As a result, we limit our short tutorial here to an abstract level, without the physical details, describing only the common principles and phenomena.

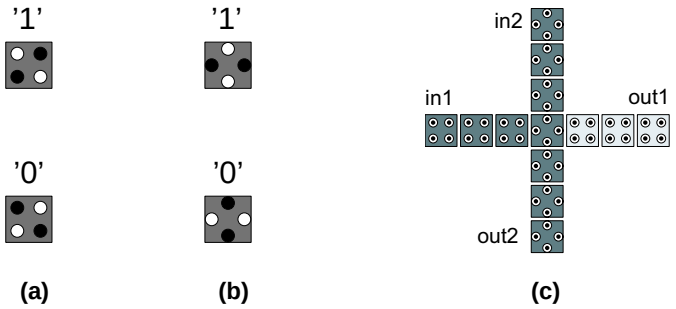


Fig. 1. QCA cell types and wires: a) cell type 1 polarizations, b) cell type 2 polarizations, and c) coplanar wire crossing, with the gray levels corresponding to different clocking zones.

3.1 Cellular automata

The information storage and transport on quantum-dot cellular automata is not based on the flow of electrical particle current, but on the local position of the charged particles inside a small section of the circuit, called a cellular automaton. This QCA cell has a limited number of quantum-dots, which the particles can occupy, and these dots are arranged such that the cell can have only two polarizations (two degenerate quantum mechanical ground states), representing binary value zero or one. A cell can switch between the two states by letting the charged particles tunnel between the dots quantum mechanically.

The cells exchange information by classical Coulombic interaction. An input cell forced to a polarization drives the next cell into the same polarization, since this combination of states has minimum energy in the electric field between the charged particles in neighboring cells. Information is copied and propagated in a wire consisting of the cell automata. Figure 1 shows the available two cell types, which are orthogonal and have minimal interaction with each other, enabling the coplanar wire crossing, where the wires consist of different cell types and can operate independently on the same fabrication layer. A traditional multi-layer crossing can be constructed with either cell type, but the technology has not been demonstrated yet.

The QCA cells can form the primitive logic gates shown in Fig. 2. The simplest structure, the inverter, is usually formed by placing the cells with only their corners touching. The electrostatic interaction is inverted, because the quantum-dots of different polarizations are misaligned between the cells. The other gates are usually based on a three-input majority gate, settling into minimum energy between the input and output cells. The gate performs the two-input AND-operation when the third input is fixed at logical zero, and the two-input OR-operation when the third input is fixed at logical one. Together with the inverter this forms a universal logic set, capable of implementing any combinatorial computation. [2]

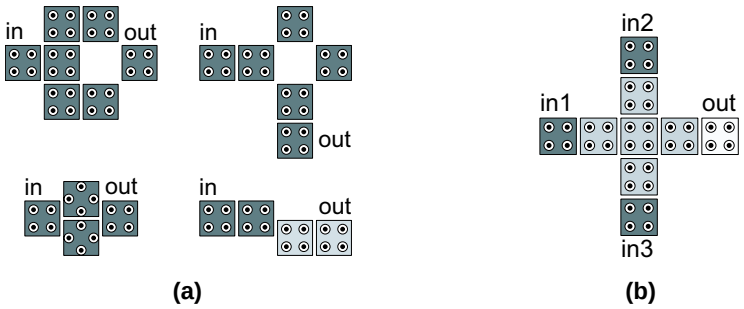


Fig. 2. QCA primitive logic gates: a) several inverter types, and b) 3-input majority gate, with the gray levels corresponding to different clocking zones.

3.2 Clocking

In QCA, a clocking mechanism determines via an electric field when the cells are un-polarized, latch their input values, and start driving other cells. It is used both for designing sequential circuits and forcing the circuit to stay in the quantum mechanical ground state, which depends on the inputs of the circuit, and represents the correct computational result and successful signal propagation. The clock provides also additional energy, enabling true signal gain on the nanotechnology.

A large array of cells switching at the same time can get stuck in a local energy minimum of the combined electric field (a *kink* state), never reaching the ground state, producing an erroneous computation result. To prevent this, the active phase of the clock is applied only to a small section of the circuit at each instant, making the probability of false energy minimum to diminish [3]. The maximum section size is not yet determined, but thermal fluctuations set another upper limit: on molecular QCA, a single majority gate would function up to the temperature of 450 K, and a wire segment of 50 cells would still operate correctly at room temperature [25].

The section size can be restricted by dividing the cell array into zones controlled by different clocks, usually four different phases for adjacent zones. During a complete clock cycle, each zone goes through the four phases. The clock transition speed is limited to enable semi-adiabatic switching, thus reducing heat dissipation by changing the cell states with re-used signal energy. [3]

3.3 Noise coupling

The ideal QCA cell is coupled with its direct neighbors only, but in real implementations, the electric field and quantum coherence transmit the interaction farther, although very weakly. This weak interaction was earlier believed to be adequately canceled out by layout symmetry, but a recent study showed that the lack of *timing*

symmetry can create sneak noise paths in crucial circuit structures [7, 8].

The coplanar wire crossing is extremely sensitive to noise coupling: In static case (when all the cells have settled to a polarization), the wire of the cell type two causes a symmetrical and effectively self-canceling interaction to the output side of the other wire, but in dynamical case (when the signals are arriving to the crossing), the polarization is at first present only at the input side, so that the compensating interaction from the output side is missing. This small unbalance causes the signal to couple into the output segment of the type one wire, followed by rapid copying from cell to cell and amplification, making the later compensation insignificant. When the real signal of the type one wire arrives to the crossing, it will not be strong enough to switch the output segment already settled in strong polarization.

The majority gate is also sensitive to signal timing [26], but both of these circuit primitives can be fixed with a simple arrangement of the QCA clocking zones, ensuring that the real input signal is always present and driving strongly, when the crucial circuit section begins to switch. A coplanar wire crossing functions correctly, when the output section of the cut wire is switched only after the other parts have firmly settled. Similarly, a majority gate functions reliably when it is placed on three clocking zones: the first zone secures the inputs, the second zone performs majority voting, and the third zone latches the result. Such zone assignments are shown with different gray levels in Fig. 1 and 2. [7, 8]

4 Serial-Parallel Multiplier

4.1 Logical structure

The serial-parallel multiplier [13] (recently named carry delay multiplier and optimized in [19]) is formed by a chain of identical functional units, corresponding to a summation row in the paper-and-pencil multiplication algorithm with n-bit binary input operands $A = (a_{n-1}, \ldots, a_1, a_0)$ and $B = (b_{n-1}, \ldots, b_1, b_0)$, resulting in a $2n$-bit output value $M = (m_{2n-1}, \ldots, m_1, m_0)$, where a_0, b_0, and m_0 are the least significant bits, respectively:

$$
\begin{array}{cccccccc}
 & & & & a_{n-1} & \cdots & a_1 & a_0 \\
\times & & & & b_{n-1} & \cdots & b_1 & b_0 \\
\hline
 & & & & a_{n-1}b_0 & \cdots & a_1b_0 & a_0b_0 \\
 & & & a_{n-1}b_1 & \cdots & a_1b_1 & a_0b_1 & 0 \\
 & & & & \vdots & & 0 & 0 \\
+ & a_{n-1}b_{n-1} & \cdots & a_1b_{n-1} & a_0b_{n-1} & 0 & 0 & 0 \\
\hline
 & m_{2n-1} & \cdots & \cdots & \cdots & \cdots & m_1 & m_0
\end{array}
$$

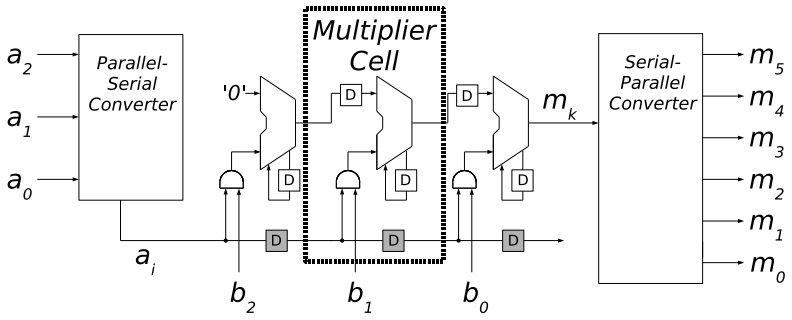


Fig. 3. Logical structure of the serial-parallel multiplier (3-bit case shown).

The multiplier cell within the dashed box in Fig. 3 computes a single bit multiplication of bits a_i from the serial and b_j from the parallel operand with an AND-gate, forming a *summand* s_j, which is combined by a full adder with a sum output sum_{j-1} from a cell to the left, and a previous carry output $carry_j$. Corresponding to a row in the paper-and-pencil equation, on each cycle, the chain of units forms a *partial product*, and sums it with the previous one. The final result is available in serial form on the output m_k on the right, on consecutive clock cycles.

4.2 Pipeline

The serial operand cannot reach every multiplier cell on the same clock cycle, since distance translates directly into timing on QCA. We have to feed the operand through a shifting pipeline, spreading the computation of a partial product both in space and time. The critical path goes through every full adder, the implementation having a carry latency of one and a sum latency of two clock cycles, enabling serial operation without pipeline stalls, while the summands are formed in parallel. The total latency is defined by the critical path propagating the accumulated carries to the final result, growing linearly with the operand word length n, the least significant bit (LSB) appearing after $L_{LSB} = n+3$, and the most significant bit (MSB) after $L_{MSB} = 3n+2$ clock cycles. (With optimized structure, $L_{MSB} = 2n$ cycles [19]).

The registers shown with shaded boxes in Fig. 3 correspond to the delays required by the serial operand wiring, while the other registers can be absorbed into the clocked adder units. Figure 4 shows the input/output timing of the complete n-bit unit, including the data format converters, showing the precise setting and holding of the parallel input words and additional zero values. The total latency L of the parallel output is $L = L_{MSB} = 3n+2$ cycles.

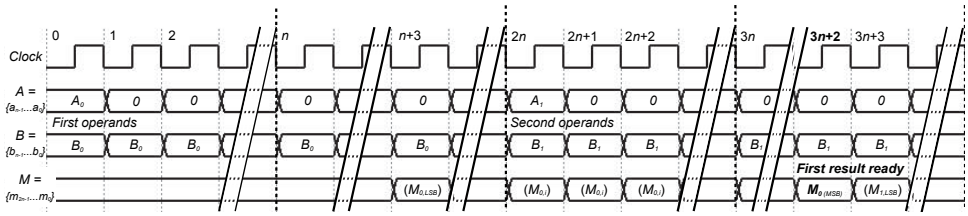


Fig. 4. Timing of the complete n-bit serial-parallel multiplier.

4.3 Layout

The functional cell of the multiplier is based on our dense serial/full adder implementation [9], following minimal majority logic formulation. The layout within the dashed box in Fig. 5(a) has at most a section of 12 QCA cells switching at the same time, forming the carry feedback loop of the serial adder. The bottom-left majority gate performs the AND-operation producing the summand, which the serial adder receives at the same instant as the sum from the left cell and the carry from previous cycle. In the bottom portion, the serial operand bit a_i is routed to the next cell, crossing over operand bit b_j line, consuming one clock cycle.

An n-bit multiplier is formed by combining n multiplier cells to the regular chain shown in Fig. 5(a) and 5(b). The wiring of the parallel operand B is shown on the bottom-left, delaying each bit according to the input stage on the chain. Without this wiring, the inputs would be spread on a large area, which is not practical, if there is a compact bus feeding the unit. The parallel-serial converter of operand A is located on the top-left, while the bit-serial result m_k emerges from the right end of the chain, followed by the serial-parallel converter producing the result M.

5 Pipelined Array Multiplier

5.1 Logical structure

The textbook array multiplier [27] and also our implementation [10] is formed by a regular lattice of identical functional units, following the paper-and-pencil multiplication algorithm with n-bit binary input operands $A = (a_{n-1}, \ldots, a_1, a_0)$ and $B = (b_{n-1}, \ldots, b_1, b_0)$, resulting in a $2n$-bit output value $M = (m_{2n-1}, \ldots, m_1, m_0)$, where a_0, b_0, and m_0 are the least significant bits, respectively. The paper-and-pencil computation is mapped straight on the hardware, the smallest unit with all the necessary functionality having three-bit operands.

Figure 6(a) shows the logical structure of the multiplier cell, which computes a single bit multiplication of bits a_i and b_j using an AND-gate, forming a *summand*

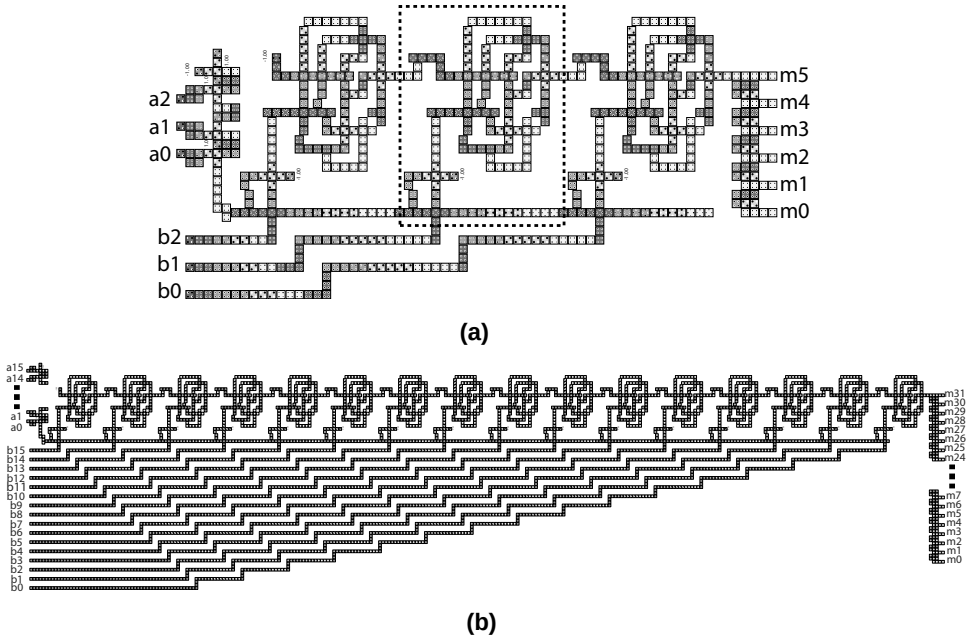


Fig. 5. The QCA layout of the serial-parallel multiplier: a) 3-bit multiplier, including the format converters and distribution wiring, having a latency of 11 clock cycles, and b) 16-bit multiplier, including the format converters and distribution wiring, having a latency of 50 clock cycles.

$s_{i,j}$, which is combined by a full adder with a previous sum output $sum_{i,j-1}$ from a cell above, and a previous carry output $carry_{i-1,j}$ from a cell to the right. Corresponding to a row in the paper-and-pencil equation, each row in the multiplier array shown in Fig. 6(b) forms a *partial product*, and sums it with the output of a row above. The final result is available in parallel form on the outputs $sum_{i,j}$ of the bottom cells in each column, and the most significant bit (MSB) appears on the last carry output $carry_{n-1,n-1}$. (The logic of the outer perimeter cells can be optimized, having a linear effect on circuit area. The current design feeds zero values into the extra inputs, resulting in equivalent logical operation.)

5.2 Pipeline

The critical path of a cell is formed by the full adder, which has a carry latency of one clock cycle and a sum latency of two cycles, while the summand is formed in parallel. The dependencies between the multiplier cells dictate that the multiplication proceeds from top-right to bottom-left corner, the diagonal critical path latency L growing linearly with the operand word length n: $L = 4n - 1$ cycles.

The input operands must be delayed according to the computational order of

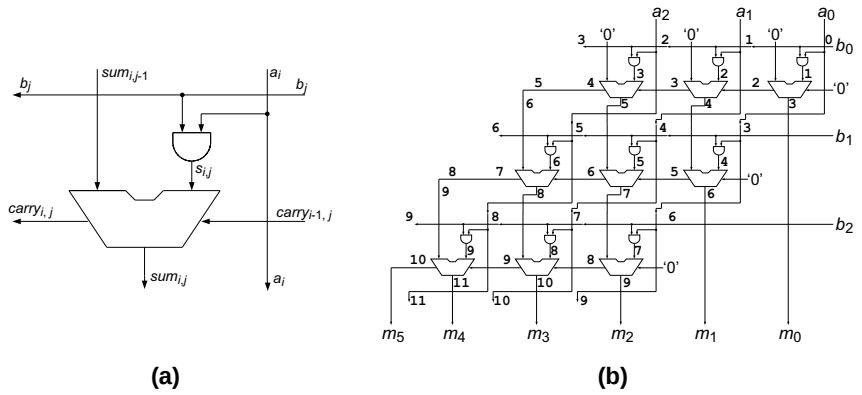


Fig. 6. Logical structure of the array multiplier: a) multiplier cell, and b) 3-bit multiplier.

the cells in different rows and columns. The least significant bit (LSB) pair (a_0, b_0) can be fed into the array instantly, but the other operand bits a_i must be delayed by i cycles, and the bits b_j by $3j$ cycles. The multiplier outputs have to be synchronized into a parallel word, by delaying each sum output corresponding to result bit $m_k, k = 0 \ldots (2n-2)$ by $(4(n-1)-2j-i)$ cycles, where j is the row and i the column index of the output cell in the paper-and-pencil organization, while the last carry output, the MSB result bit m_{2n-1}, must be delayed by one cycle. The numbers shown in Fig. 6(b) correspond to the signal delays at the core array inputs and pipeline stages, while Fig. 7 shows the input/output timing of the complete n-bit unit, including the synchronization.

5.3 Layout

The multiplier cell is based on a dense full adder implementation, following the minimal majority logic formulation presented in [14]. The layout shown in Fig. 8(a)

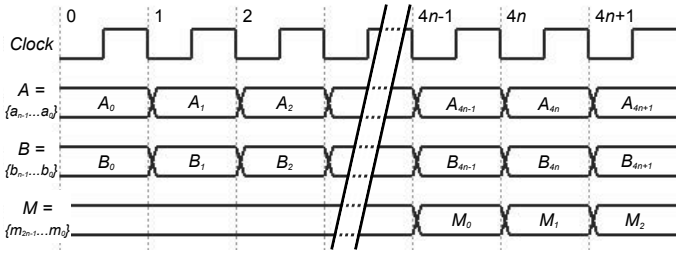


Fig. 7. Timing of the complete n-bit array multiplier.

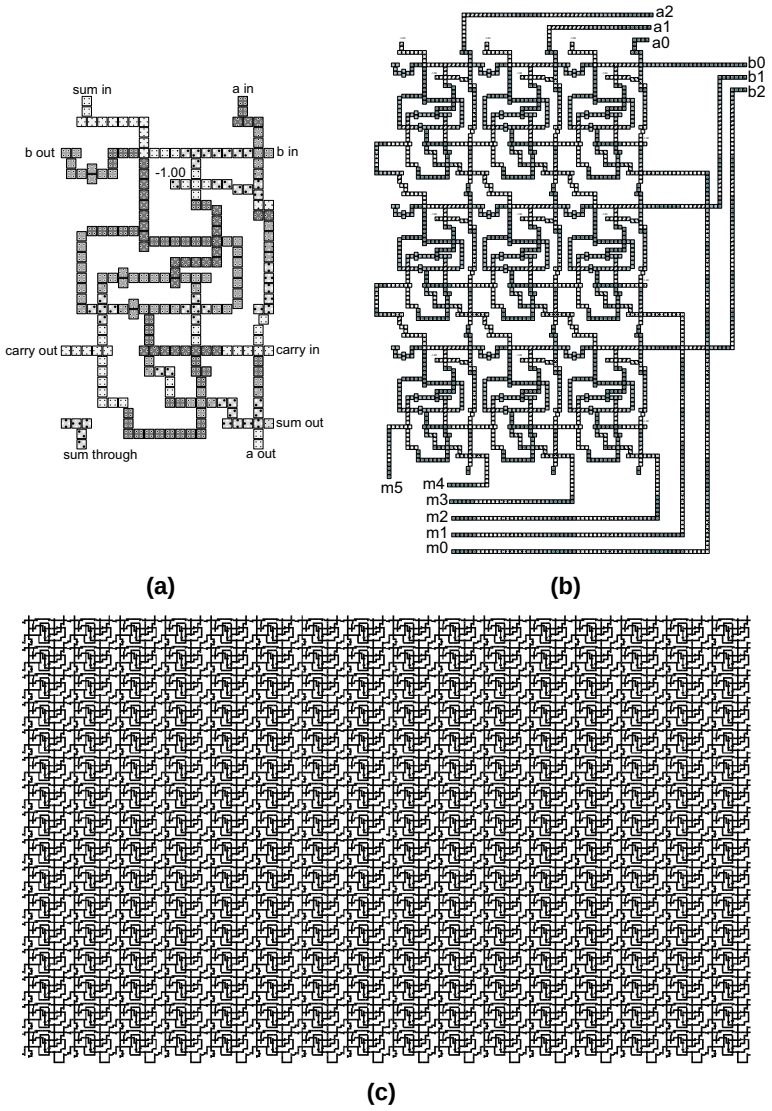


Fig. 8. The QCA layout of the array multiplier: a) multiplier cell, with a delay of 2 clock cycles for the sum, 1 for the carry, 3 for operand a_i path, and 1 for operand b_j path, b) 3-bit multiplier including operand and result delay lines, having a latency of 11 clock cycles, and c) 16-bit multiplier without operand and result delay lines, having a latency of 63 clock cycles (rotated 90° counter clockwise).

has been optimized for multiplier use by placing the inputs and outputs so that the wires between the cells are kept short, and the number of wire crossings is minimal. The largest section of QCA cells switching at the same time is limited to 20 cells, placed in the continuous zone forming the sum input distribution network.

The topmost majority gate performs the AND-operation producing the summand, which the full adder receives at the same instant as the upper sum and right side carry inputs. On the right side, the operand bit a_i is routed to the next row, crossing over operand bit b_j, carry input $c_{i-1,j}$ and sum output $s_{i,j}$, consuming three clock cycles. In the top portion, the operand bit b_j is routed to the next column on the left, crossing over operand bit a_i and sum input $s_{i,j-1}$, consuming one clock cycle.

An n-bit multiplier is formed by combining n^2 multiplier cells to the regular array shown in Fig. 8(b) and 8(c). The rectangular shape was handcrafted iteratively, as the routing and the clocking zones of the cell were placed to ensure that noise interaction appears only after the correct signals have firmly settled. The operand bits are delayed before entering the array, the wiring shown in Fig. 8(b): operand A is located in the top portion, without any wire crossings, and operand B on the right side, crossing some of the multiplier output signals. Half of the outputs emerge from the right side of the array, and the others from the bottom, where the results bits are gathered and synchronized into the parallel word M.

6 Verification

The logical correctness of the designs was checked with paper-and-pencil analysis, while the handcrafted layouts were simulated with the QCADesigner tool, using the coherence vector engine [28, 29]. The computation is based on the Hartree-Fock approximation, which models each QCA cell as a single coherent quantum mechanical system, while the interaction between the cells is modeled with only classical electrostatic mechanism. The simulation is marched forward in a time-dependent manner, which is able to reveal the circuit sections that are susceptible to noise coupling and amplification, giving early prediction of signal race conditions.

The multiplier cells were simulated exhaustively (covering all the pipeline states), 3-bit multipliers with all possible input combinations, and larger n-bit units with various operand sizes and representative input cases (since the number of states grows exponentially, preventing exhaustive runs). Correct multiplication results were obtained with extremely high simulated clock frequencies (several thousands of gigahertz), indicating that unwanted signal coupling is controlled throughout the circuits, and the designs are robust with all operand word lengths.

The simulated layouts were based on a QCA cell sized 18×18 nm, with 4 quantum dots each having a diameter of 5 nm, and the distance between the center of cells was 20 nm (corresponding to a semiconductor technology, for comparison). The parameters used in QCADesigner version 2.0.3 coherence vector engine were the defaults: temperature 1 K, relaxation time 1 fs, time step 0.1 fs, clock high 9.8×10^{-22} J, clock low 3.8×10^{-23} J, clock shift 0, clock amplitude factor 2,

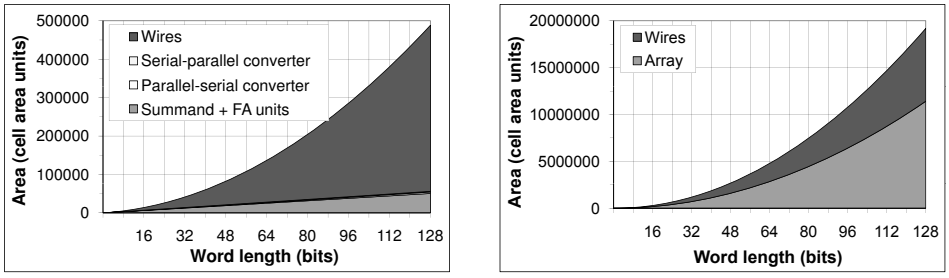


Fig. 9. The area of the multiplier structures: a) serial-parallel multiplier, and b) array multiplier.

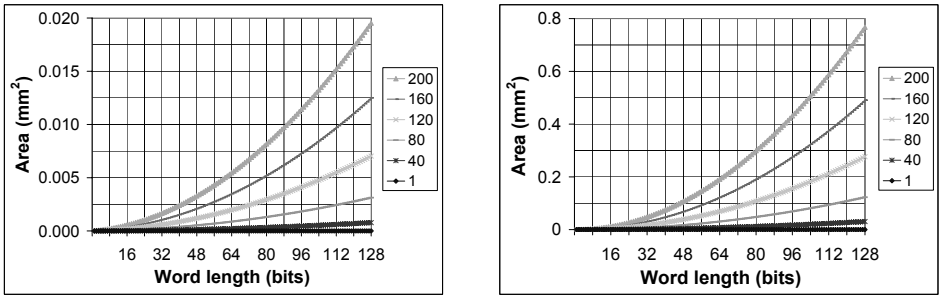
radius of effect 80 nm, relative permittivity 12.9, layer separation 11.5 nm, Euler method, and randomized simulation order.

7 Area

The wiring was found to contribute heavily to the area of the arithmetic units. The previous proposals of the serial-parallel multiplier [13, 19] promised linearly increasing area compared to the length of the operand word, but this was accomplished by leaving out the wires needed to distribute the parallel operand across the unit, originating from a compact bus. As this wiring is usually necessary, we included it in our analysis, optimizing for latency and area, under the single-layer technology constraint. The practical serial-parallel design occupies quadratic area, and the percentage of wiring dominates from 16-bit to larger units, reaching 90% in the 128-bit case as shown in Fig. 9(a). Also the array multiplier grows quadratically, but it has equally fast growing terms for both the active area and the wiring, the overhead settling to a constant 40%, as shown in Fig 9(b).

The array multiplier grows much faster than the serial-parallel unit (even without the linear advantage): a 16-bit array is 20 times as large as the corresponding serial-parallel unit, and the ratio of the areas settles asymptotically, from 128-bit units upwards, to the array being about 40 times as large.

The above area relations hold for the QCA multipliers on different implementation technologies, but the area (nm^2) grows also quadratically with the technology specific QCA cell width (nm). The unit cell sizes 0–10 nm correspond to predicted molecular and 10–200 nm to semiconductor QCA technologies, resulting in the feature size specific areas shown in Fig. 10.



(a) (b)

Fig. 10. The areas for some implementation technologies (according to cell width in nanometers): a) serial-parallel multiplier, and b) array multiplier.

8 Performance

The latency of both multipliers grows linearly with the operand word, but there is tremendous difference in the throughput. The array multiplier produces a new result at each clock cycle, achieving a constant throughput, while the serial-parallel unit suffers a rapid decrease, as the throughput is proportional to the inverse of the word length. Table 1 summarizes the structural characteristics of the designs.

An important measure of cost efficiency is, how many multiplication results we can obtain per clock cycle, for a unit circuit area invested in the multiplier. The throughput per area metric decreases faster than inversely to the square of the word length, as shown in Fig. 11(a). On very small word lengths (2–4 bit operands) the designs have the same efficiency, but with larger operands, the array multiplier is linearly better: a 128-bit array produces 6.5 times more results than a serial-parallel unit, per clock cycle and unit area, as shown in Fig. 11(b).

Table 1. Asymptotic comparison of n-bit multiplier structures.

Design	**Latency**	**Throughput**	**Area**
Array Multiplier [10]	$4n-1$	1	$1100n^2$
Serial-Parallel Multiplier following [13, 19]	$3n+2$	$1/(2n)$	$26n^2$

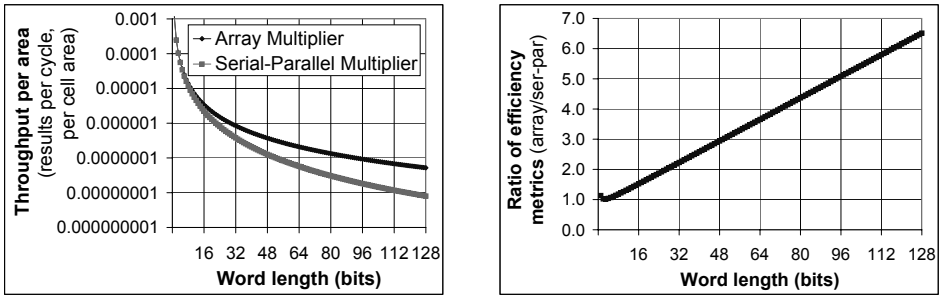


Fig. 11. The performance-area efficiency of the multipliers: a) throughput per area unit, and b) the ratio of the efficiency metrics (array per serial-parallel).

9 Power

The thermal noise floor sets a requirement for a signal to have an energy content of at least $E_{sig} = 100k_BT$ (where k_B is the Bolzmann's constant, and T the temperature in kelvins), to obtain a decent error probability of 3.72×10^{-44} [30]. However, there is no fundamental need to dissipate this energy on every step of computation, like the conventional technologies do; the laws of physics require a dissipation of about 140 times smaller energy, and *only* when a bit of information is lost [11]. On QCA, most of the signal energy can be transferred from cell to cell and re-used, making the unavoidable erasure dissipation a significant factor. This ultimate limit of energy efficiency can be reached, since there is no need to move electron currents and charge circuit capacitances. [20]

The power analysis presented here is based on the thermodynamical requirement of heat generation in the active QCA layer, which is expected to dominate the power, but also the underlying clock network, possibly implemented with CMOS technology, will be a source of dissipation. The clock network should be charged adiabatically, and the generator could be placed off-chip to ease cooling. [12]

9.1 Bit erasure energy

The Landauer's principle [11], resulting from a thermodynamic consideration of energy and entropy of the system, declares, that losing a bit of information about the system state leads inevitably to dissipating $E_{dis} = k_BT\ln 2$ of energy into the environment (computed in this paper at the room temperature). The circuits presented in our study are based on irreversible logic, losing information at each step of computation, and their lowest power limits are set by this law of nature, when

operated with the normal Landauer-type clocking [3, 31].

The multipliers discard information in each functional unit, at each clock cycle. A combination of a two-input AND-gate and a full adder compresses 16 distinct input combinations into four different output values (in quite an unbalanced way, most information lost when seven input cases are mapped onto a single output case), losing three bits of system state in one multiplier cell. This is luckily not the real case, because the input operand bits a_i and b_j of these units are either held intact for the whole computation or fed with the outputs to the next functional unit. As a by-product of the multiplier organization, these bits are available for reversing the computation, and the multiplier cell loses only one bit of information. At the end of an n-bit multiplication, also the original input operands must be discarded, erasing $2n$ bits. The total number of bit erasures is directly, and the energy efficiency inversely, proportional to the square of the word length, as shown in Fig. 12.

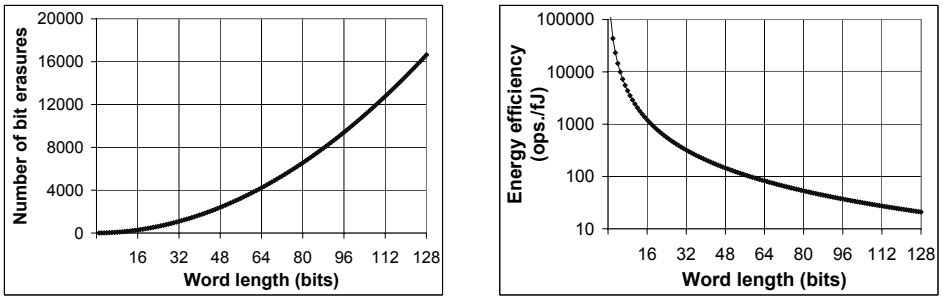


Fig. 12. The bit erasures in n-bit multiplication, when the operand bits are available with intermediate outputs: a) total number, and b) resulting energy efficiency (multiplications per unit energy).

9.2 Power density

The place and timing of the bit erasures in an n-bit multiplication is different for each design, in principle affecting the expected power density, but in practice, evened out by the array computing n multiplications in parallel, while the serial-parallel unit computes a single multiplication. There is also a difference in the hardware utilization, as the array can run with 100% of the functional units computing all the time, while the serial-parallel structure reaches a utilization of about 75%. This is due to the fact that the last pipeline stages of the bit-serial approach form a bottleneck, forcing the previous stages to stall or compute with zero inputs.

The minimum reachable power density is found by normalizing the total erasure energy with the computation time and the area of the multiplier. The metric

does not depend on the word length of the unit, only on the structure and the operating frequency; the array has a smaller value than the serial-parallel unit, on the same clock rate. Implementation specific power densities for various molecular technologies are shown in Fig. 13.

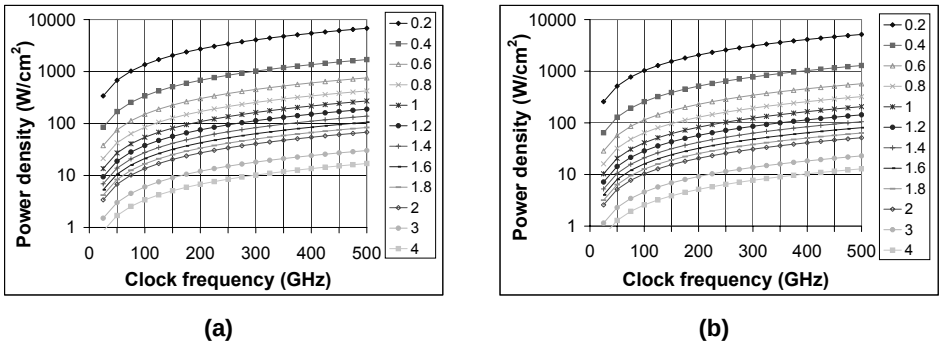


Fig. 13. The power density of the designs on molecular technologies (according to the cell width in nanometers): a) serial-parallel multiplier, and b) array multiplier.

The power density is highly dependent on the cell size of the technology, limiting the maximum operating frequency, as we can easily cool only about 100 watts of heat per square centimeter. This is a problem with molecular implementations, while the more coarse technologies are limited by other issues. In view of heat generation, a coarse featured technology might be more feasible than the smallest possible; the maximum clock frequencies of the multipliers, for various molecular QCA technologies, are shown in Fig. 14.

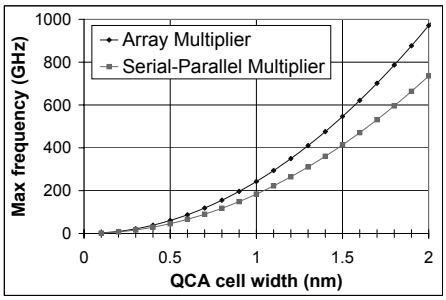


Fig. 14. The maximum operating frequency of the multiplier structures on various molecular QCA technologies, under the cooling restriction of 100 W/cm^2.

In our area optimized implementations, the array multiplier enables higher operating frequencies than the serial-parallel multiplier, even with the bit-serial structure having several idle cycles reducing the power density. On a 1 nm cell technology, the maximum frequency of the array is 240 GHz, while the serial-parallel

structure reaches 180 GHz. Although these are very high frequencies compared to conventional technologies, they are quite restricting limits on a nanotechnology, which is predicted to have switching in the terahertz regime.

10 Conclusion

The nanotechnology multipliers described in this paper are free of the noise problems found in most previous arithmetic structures on QCA, and the correct operation has been verified with the most reliable tool currently available. The circuits have inevitable quadratically wiring overhead, resulting in the massive array multiplier having the best area efficiency, in addition to the best performance. The operating frequencies are limited by the power dissipation of discarding information, governed by the Landauer's principle, and the restrictions found under irreversible operation are much tighter, than the predicted switching speeds of molecular QCA technologies.

Our work is one of the first attempts to design and evaluate feasible arithmetic units on the very promising QCA nanotechnology, which has several problems hindering large scale manufacturing. The fundamental issue still to be addressed on all levels of design is the presence of faults and defects inherent to the molecular technologies, requiring redundancy to increase the robustness of the circuits. We expect wires to scale easily using modular redundancy, but constructing logic gates is much more demanding (a promising approach is characterized in [32]). The actual manufacturing processes are very early in their development; a feasible approach might be based on a combination of high-resolution electron beam lithography and DNA nano-patterning [33].

The power dissipation of complete QCA circuits, including the clock distribution and generation, has to be analyzed in more detail, as we get to know more of the parameters of the implementation technologies. Since QCA reaches the fundamental limits of irreversible computing, the role of reversibility will certainly increase on every level of design work.

References

[1] C. Lent, "Molecular quantum-dot cellular automata," in *Proc. IEEE Workshop Signal Processing Systems Design and Implementation (SIPS)*, Banff, AB, Canada, Oct. 2–4, 2006, keynote talk.

[2] C. Lent, P. Tougaw, and W. Porod, "Quantum cellular automata: the physics of computing with arrays of quantum dot molecules," in *Proc. Workshop Physics and Computation (PhysComp)*, Dallas, TX, Nov. 17–20, 1994, pp. 5–13.

[3] C. Lent and P. Tougaw, "A device architecture for computing with quantum dots," *Proc. IEEE*, vol. 85, no. 4, pp. 541–557, Apr. 1997.

[4] G. Snider, A. Orlov, I. Amlani, G. Bernstein, C. Lent, J. Merz, and W. Porod, "Quantum-dot cellular automata," in *Dig. Papers of Microprocesses and Nanotechnology Conf.*, Yokohama, Japan, July 6–8, 1999, pp. 90–91.

[5] A. Orlov, R. Kummamuru, R. Ramasubramaniam, C. Lent, G. Bernstein, and G. Snider, "Clocked quantum-dot cellular automata devices: experimental studies," in *Proc. IEEE Conf. Nanotechnology (NANO)*, Maui, HI, Oct. 28–30, 2001, pp. 425–430.

[6] R. 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 Trans. Electron Devices*, vol. 50, no. 9, pp. 1906–1913, Sept. 2003.

[7] K. Kim, K. Wu, and R. Karri, "The robust QCA adder designs using composable QCA building blocks," *IEEE Trans. Computer-Aided Design*, vol. 26, no. 1, pp. 176–183, Jan. 2007.

[8] ——, "Towards designing robust QCA architectures in the presence of sneak noise paths," in *Proc. Design, Automation and Test in Europe (DATE)*, Messe Munich, Germany, Mar. 7–11, 2005, pp. 1214–1219.

[9] I. Hänninen and J. Takala, "Robust adders based on quantum-dot cellular automata," in *Proc. IEEE Int. Conf. Application-Specific Systems, Architectures and Processors (ASAP)*, Montréal, QC, Canada, July 8–11, 2007, pp. 391–396.

[10] ——, "Pipelined array multiplier based on quantum-dot cellular automata," in *Proc. European Conf. Circuit Theory and Design (ECCTD)*, Seville, Spain, Aug. 26–30, 2007, pp. 938–941.

[11] R. Landauer, "Irreversibility and heat generation in the computing process," *IBM J. Res. Dev.*, vol. 5, pp. 183–191, 1961.

[12] C. Lent, M. Liu, and Y. Lu, "Bennett clocking of quantum-dot cellular automata and the limits to binary logic scaling," *Nanotechnology*, vol. 17, no. 16, pp. 4240–4251, Aug. 2006.

[13] K. Walus, G. Jullien, and V. Dimitrow, "Computer arithmetic structures for quantum cellular automata," in *Conf. Rec. 37th Asilomar Conf. Signals, Systems and Computers*, Pacific Grove, CA, Nov. 9–12, 2003, pp. 1435–1439.

[14] W. Wang, K. Walus, and G. Jullien, "Quantum-dot cellular automata adders," in *Proc. IEEE Conf. Nanotechnology (NANO)*, San Francisco, CA, Aug. 11–14, 2003, pp. 461–464.

[15] R. Zhang, K. Walus, W. Wang, and G. Jullien, "Performance comparison of quantum-dot cellular automata adders," in *IEEE Int. Symp. Circuits and Systems (ISCAS)*, Kobe, Japan, May 23–26, 2005, pp. 2522–2526.

[16] H. Cho and E. Swartzlander, "Adder designs and analyses for qauntum-dot cellular automata," *IEEE Trans. Nanotechnol.*, vol. 6, no. 3, pp. 374–383, May 2007.

[17] A. Vetteth, K. Walus, V. Dimitrov, and G. Jullien, "Quantum-dot cellular automata carry-look-ahead adder and barrel shifter," in *Proc. IEEE Conf. Emerging Telecommunications Technologies*, Dallas, TX, Sept. 23–24, 2002.

[18] J. Janulis, P. Tougaw, S. Henderson, and E. Johnson, "Serial bit-stream analysis using quantum-dot cellular automata," *IEEE Trans. Nanotechnol.*, vol. 3, no. 1, pp. 158–164, Mar. 2004.

[19] H. Cho and E. Swartzlander, "Serial parallel multiplier design in quantum-dot cellular automata," in *Proc. IEEE Symp. Computer Arithmetic (ARITH)*, Montepellier, France, June 25–27, 2007, pp. 7–15.

[20] J. Timler and C. Lent, "Maxwell's demon and quantum-dot cellular automata," *J. Appl. Phys.*, vol. 94, pp. 1050–1060, July 2003.

[21] L. Bonci and M. Macucci, "Numerical investigation of energy dissipation in quantum cellular automaton circuits," in *Proc. European Conf. Circuit Theory and Design (ECCTD)*, Cork, Ireland, Aug. 29–Sep. 1 2005, pp. II/239–II/242.

[22] ——, "Analysis of power dissipation in clocked quantum cellular automaton circuits," in *Proc. European Solid State Circuits Conf. (ESSCIRC)*, Montreux, Switzerland, Sept. 18–22, 2006, pp. 58–61.

[23] J. Huang, X. Ma, and F. Lombardi, "Energy analysis of QCA circuits for reversible computing," in *Proc. IEEE Conf. Nanotechnology (NANO)*, Westin Cincinnati, OH, July 17–20, 2006, pp. 39–42.

[24] S. Srivastava, S. Sarkar, and S. Bhanja, "Power dissipation bounds and models for quantum-dot cellular automata circuits," in *Proc. IEEE Conf. Nanotechnology (NANO)*, Westin Cincinnati, OH, July 17–20, 2006, pp. 375–378.

[25] Y. Wang and M. Lieberman, "Thermodynamic behavior of molecular-scale quantum-dot cellular automata QCA wires and logic devices," *IEEE Trans. Nanotechnol.*, vol. 3, no. 3, pp. 368–376, Sept. 2004.

[26] Z. Patitz, N. Park, M. Choi, and F. Meyer, "Qca-based majority gate design under radius of effect-induced faults," in *Proc. IEEE Int. Symp. Defect and Fault Tolerance in VLSI Systems (DFT)*, Monterey, CA, Oct. 3–5, 2005, pp. 217–225.

[27] J. Rabaey, *Digital Integrated Circuits, A Design Perspective*, ser. Electronics and VLSI. Upper Saddle River, NJ: Prentice Hall, 1996, ch. 7.4.

[28] K. Walus and G. Jullien, "Design tools for an emerging SoC technology: quantum-dot cellular automata," *Proc. IEEE*, vol. 94, no. 6, pp. 1225–1244, June 2006.

[29] (2007) QCADesigner website. University of Calgary ATIPS Laboratory. [Online]. Available: http://www.qcadesigner.ca

[30] M. Frank, "Introduction to reversible computing: Motivation, progress, and challenges," in *Proc. ACM Int. Conf. Computing Frontiers (CF)*, Ischia, Italy, May 4–6, 2005, pp. 385–390.

[31] E. Blair and C. Lent, "Quantum-dot cellular automata: an architecture for molecular computing," in *Proc. Int. Conf. Simulation of Semiconductor Processes and Devices (SISPAD)*, Boston, MA, Sept. 3–5, 2003, pp. 14–18.

[32] J. Huang, M. Momenzadeh, and F. Lombardi, "On the tolerance to manufacturing defects in molecular QCA tiles for processing-by-wire," *J. Electron. Testing*, vol. 23, no. 2–3, pp. 163–174, June 2007.

[33] W. Hu, K. Sarveswaran, M. Lieberman, and G. Bernstein, "High-resolution electron beam lithography and DNA nano-patterning for molecular QCA," *IEEE Trans. Nanotechnol.*, vol. 4, no. 3, pp. 312–316, May 2005.