Automated Design Architecture for 1-D Cellular Automata Using Quantum Cellular Automata

Vassilios A. Mardiris, *Member, IEEE*, Georgios Ch. Sirakoulis, *Member, IEEE*, and Ioannis G. Karafyllidis

Abstract—Cellular automata (CAs) have been widely used to model and simulate physical systems and processes. CAs have also been successfully used as a VLSI architecture that proved to be very efficient at least in terms of silicon-area utilization and clock-speed maximization. Quantum cellular automata (QCAs) as one of the promising emerging technologies for nanoscale and quantum computing circuit implementation, provides very high scale integration, very high switching frequency and extremely low power characteristics. In this paper we present a new automated design architecture and a tool, namely DATICAQ (Design Automation Tool of 1-D CAs using QCAs), that builds a bridge between 1-D CAs as models of physical systems and processes and 1-D QCAs as nanoelectronic architecture. The QCA implementation of CAs not only drives the already developed CAs circuits to the nanoelectronics era but improves their performance significantly. The inputs of the proposed architecture are CA dimensionality, size, local rule, and initial and boundary conditions imposed by the particular problem. DATICAQ produces as output the layout of the QCA implementation of the particular 1-D CA model. Simulations of CA models for zero and periodic boundary conditions and the corresponding QCA circuits showed that the CA models have been successfully implemented.

Index Terms—Cellular automata (CAs), quantum cellular automata (QCAs), design automation, nanoelectronic circuits

1 Introduction

Nanoelectronic circuits are expected to replace gradually the CMOS circuits in the years to come. Several possible nanotelectronic technologies have been proposed by researchers, such as carbon nanotubes, silicon nanowires, quantum-dot cellular automata (QCA), single-electron transistors (SETs) and circuits, resonant tunneling diodes, and single-molecule devices, as reported by ITRS [1].

Single electronics can be defined as the control of transport and position of a single electron or a small number of electrons in nanometer structures [2]. Significant research is being done on the field of modelling and simulation of SET devices and circuits [3], [4], [5], [6] which is accompanied by the development of an efficient single electron circuit theory [7].

QCA is a field-coupled computing approach. QCA cells change their states due to interactions with neighboring cells via electrostatic or magnetic fields. QCAs also offer a new computation and information representation method which is referred to as processing-in-wire. In terms of integration level, a few nanometer QCA cells can be fabricated using a self-assembly process [8], [9]. For these feature sizes the integration can reach densities of 10^{12} cells/cm^2 and the circuit switching frequency can be close to a terahertz [1]. QCAs were introduced in 1993 by Lent et al. [10] and since then several logic gates and circuits have been proposed such as the binary wire [11], the majority gate, AND, OR, NOT and XOR gates [12], bit-serial adder, full adder [13], [14], multiplier [15], multiplexer [16], [17], [18], flip-flop [19], [20], arithmetic logic units (ALU) [21] serial or parallel memories [22], [23] and cellular automata (CA) cells or models [17], [24]. Automated QCA layout tools were introduced in [25], [26].

Both individual QCA cells and multiple QCA cell arrangements, such as wires and majority logic circuits, have been fabricated and tested [27]. The major problem of the semiconductor and metallic QCA implementations is that they can only operate at cryogenic temperatures; nevertheless very recently a new fabrication method has been proposed which leads to QCA circuits operating at room temperature [74]. Research is also underway to fabricate molecular and magnetic QCAs [8], [28], [29]. These last implementations may also offer a solution to the aforementioned problem because they can operate at room temperature [30].

Cellular automata are models of physical systems, where space and time are discrete and interactions are local [31]. Prior and more recent works proved that CAs are very effective in simulating physical systems and solving scientific problems, because they can capture the essential features of systems where global behaviour emerges from the collective effect of simple components which interact locally [32], [33]. Furthermore, they can easily handle complex boundary and initial conditions, inhomogeneities and anisotropies [34]. Moreover, the CA approach is consistent with the modern notion of unified space-time. In computer science, space corresponds to memory and time to processing unit. In CA, memory (CA cell state) and processing unit (CA local rule) are inseparably related to a CA cell [35]. As a result, CAs have also been successfully used as a VLSI architecture and special computing machines have also been developed based on the CA architecture [36]. In terms of circuit design and layout, ease of mask generation,

- *V. A. Mardiris is with the Technological Educational Institute of Kavala, Kavala GR-65404, Greece. E-mail: mardiris@teikav.edu.gr.*
- *G. Ch. Sirakoulis and I. G. Karafyllidis are with the Department of Electrical and Computer Engineering, Democritus University of Thrace, Xanthi GR-67100, Greece. E-mail: {gsirak, ykar}@ee.duth.gr.*

Manuscript received 29 Sept. 2013; revised 23 June 2014; accepted 4 Aug. 2014. Date of publication 3 Nov. 2014; date of current version 12 Aug. 2015. Recommended for acceptance by K. Chakrabarty.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TC.2014.2366745

silicon-area utilization, and maximization of clock speed, CAs are perhaps one of the most suitable computational structures for VLSI realization [34], [35], [36], [37], [38].

More specifically, from circuit designing point of view, according to Toffoli [34], there are four main factors that determine the cost/performance ratio of an integrated circuit, namely, circuit design and layout, ease of mask generation, silicon-area utilization, and maximization of achievable clock speed; for a given technology, the latter is inversely proportional to the maximum length of the signal paths and this is also one of the critical factors in QCA circuit designing. CA circuit design reduces to the design of a single, relatively simple cell and layout is uniform. The whole mask for a large CA array (the cells with their internal connections as well as the interconnections between cells) can be generated by a repetitive procedure so no circuit area is wasted on long interconnection lines and because of local processing the length of critical paths is small when compared to others QCAs designs and implementations found in literature and is independent of the number of cells.

In the last two decades a wide variety of one-dimensional (1-D) CA applications have been proposed on several scientific fields like: simulation of physical systems [38], [40], [52], biological modelling involving models for self-reproduction, biological structures and DNA sequences [41], [42], image processing [35], [43], [44], language recognition [45], [46], computer architectures [47], fractals and chaos [48], [49], cryptography [50], computer networks [51], etc.

In this paper an automated cellular automata QCA design architecture and the corresponding tool, are presented for the first time. With the proposed architecture, CA rules and models can be implemented using QCAs, resulting in much better circuit characteristics in terms of area, clock frequencies and power consumption compared to the classical VLSI implementations. The QCA computational model introduces highly pipelined architectures with extremely high speeds. The information is encoded by the configuration of electrical charges inside the cells and not by electrical current flows, thus it has extremely lower power consumption compared to the classical CMOS technology [62]. For the proposed architecture the length of critical paths is kept to minimal values and is independent of the number of CA cells. Although the auto-generated design in this paper assumes metal-based QCA implementation, the underlying principles also apply to molecular QCA.

The proposed architecture uses several types of basic QCA configurable building blocks, that have been designed in order to be able to produce any 1-D CA model, by configuring and assembling them properly. All QCA building block designs are based on an universal 1-D CA cell QCA design, developed by the authors and presented in [37]. The proposed architecture can automatically design the QCA circuit that implements any 1-D CA model that comprises CA cells operating with the same or any variety (hybrid) of CA Wolfram rules [33]. The proposed tool, which will be called $DATICAQ$ (Design Automation Tool of 1-D CAs using QCAs), is a web-based application developed using the PHP programming language, and produces as output, the QCA design directly to its graphical interface. $DATICAQ$ can also generate the QCA Designer file enabling thus the immediate use of the well known $QCADesigner$ [57] in the case where the user wishes to use the produced QCA design as a part of a larger circuit.

The paper is organized as follows. The necessary background on CAs and QCAs is given in Section 2. The architecture of the automated design of 1-D CA with QCA as well as some application paradigms are presented in Section 3. In Section 4 the structure and the operation of $DATICAQ$ are also given. Finally, conclusions are drawn in Section 5.

2 CA and QCA Circuits

2.1 Cellular Automata

In this section a more formal definition of a CA will be presented [51]. In general, a CA requires:

1) a regular lattice of cells covering a portion of a d-dimensional space;
2) a set $\mathbf{C}(\vec{r},t) = (C_1(\vec{r},t), C_2(\vec{r},t), \ldots, C_m(\vec{r},t))$ of variables attached to each site $\vec{r}$ of the lattice giving the local state of each cell at the time $t = 0, 1, 2, \ldots$;
3) a rule $\mathbf{R}$=$(R_1, R_2, \ldots, R_m)$ which specifies the time evolution of the states $\mathbf{C}(\vec{r},t)$ in the following way: $\mathbf{C}_j(\vec{r}, t\ +1) = R_j(C(\vec{r},t), C(\vec{r}+\vec{\delta}_1,t), C(\vec{r}+\vec{\delta}_2,t), \ldots, C(\vec{r}+\vec{\delta}_q,t))$, where $\vec{r}+\vec{\delta}_k$ designate the cells belonging to a given neighborhood of cell $\vec{r}$.

In the above definition, the rule $\mathbf{R}$ is identical for all sites and it is applied simultaneously to each of them, leading to a synchronous dynamics. It is important to notice that the rule is homogeneous, i.e. it does not depend explicitly on the cell position $\vec{r}$. However, spatial (or even temporal) inhomogeneities can be introduced by having some cell states $C_j(\vec{r})$ constantly at 1, in some given locations of the lattice, to mark particular cells for which a different rule applies. Furthermore, in the above definition, the new state at time $t+1$ is only a function of the previous state at time t. It is sometimes necessary to have a longer memory and introduce a dependence on the states at time $t-1, t-2, \ldots, t-k$. Such a situation is already included in the definition, if one keeps a copy of the previous state in the current state.

The neighborhood of cell $\vec{r}$ is the spatial region in which a cell needs to search in its vicinity. In principle, there is no restriction to the size of the neighborhood, except that it is the same for all cells. However, in practice, it is often made up of adjacent cells only [53]. For two-dimensional (2-D) CA, two neighborhoods are often considered [52]: The von Neumann neighborhood, which consists of a central cell (the one which is to be updated) and its four geographical neighbors north, west, south and east. The Moore neighborhood contains, in addition, second nearest neighbors: northeast, northwest, southeast and southwest, that is a total of nine cells. In practice, when simulating a given CA rule, it is impossible to deal with an infinite lattice. The system must be finite and have boundaries. Clearly, a site belonging to the lattice boundary does not have the same neighborhood as other internal sites. In order to define the behavior of these sites, the neighborhood is extending for the sites at the boundary. Extending of the neighborhood leads to various types of boundary conditions such as periodic (or cyclic), fixed, adiabatic or reflection [52].

In this paper, we focus on 1-D CA with two possible states per cell, i.e., $C \in 0, 1$. In this case, the local transition

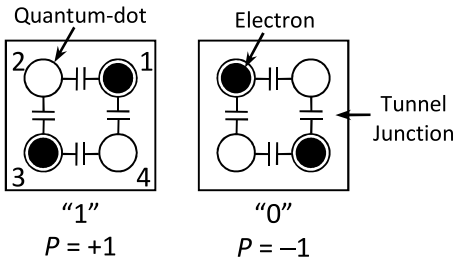

Fig. 1. The basic QCA cell.

rule R is a function $R : 0, 1^n \rightarrow 0, 1$ and the neighborhood size n is usually taken to be $n = 2\delta + 1$ such that:

$$C(r, t + 1) = R(C(r - \delta, t), \ldots, C(r, t), \ldots, C(r + \delta, t)), \quad (1)$$

where δ (positive integer) is a parameter, known as the radius, representing the standard 1-D cellular neighborhood. We shall furthermore limit ourselves to the $\delta = 1$ case, i.e., so-called elementary CA, for which the neighborhood size is $n = 3$:

$$\begin{aligned} &R : \{0, 1\}^3 \rightarrow \{0, 1\} \\ &C(r, t + 1) = R(C(r - \delta, t), C(r, t), C(r + \delta, t)). \end{aligned} \quad (2)$$

The domain of R is the set of all 2^3 3-tuples, which gives rise to $2^8 = 256$ distinct elementary rules. We will use Wolfram's decimal numbering convention for describing these rules [33], e.g. $R(111) = 1$, $R(110) = 0$, $R(101) = 1$, $R(100) = 1$, $R(011) = 1$, $R(010) = 0$, $R(001) = 0$, $R(000) = 0$, is denoted rule 184. For two-state CA a configuration of a size N grid at time t is a binary sequence $C(t)$. For finite-size grids, spatially periodic boundary conditions are frequently assumed, resulting in a circular grid; formally, this implies that cellular indices are computed modulus N.

Beyond the fact mentioned before that CAs have been successfully applied to many scientific fields, one of their most profound characteristics is related with their universality, as they represent Turing machines connected in parallel. In specific, von Neumann proved that a CA consisting of cells with four orthogonal neighbors and 29 possible states would be capable of simulating a universal Turing machine [31]. Recently, a Turing machine was explicitly implemented using a CA that executes the well known game of life by P. Rendell [66]. Most amazingly still, even 1-D CA can be universal. In the first place, Wolfram [65] gave an example of a 19-state universal 1-D next-nearest neighbor CA in which a block of 20 cells was used to represent each single cell in the CA being emulated. Later, Wolfram [65] proved even that two states and nearest neighbor rules are sufficient for providing universal computation in a 1-D CA; in particular, the rule 110 elementary CA is universal. The implementation of 1-D CA with two states per cell using basic QCA configurable building blocks can successfully lead to the implementation of Turing-like universal computing machines implemented in QCA architecture.

2.2 Quantum Cellular Automata

QCA is a nanoelectronic digital logic architecture in which information is stored as configurations of electron pairs in quantum dot arrays. QCA use arrays of coupled quantum dots to build Boolean logic functions and to perform useful computations. QCA take advantage of quantum mechanical effects to significantly reduce the size of digital circuits and operate at high speeds in very low power levels.

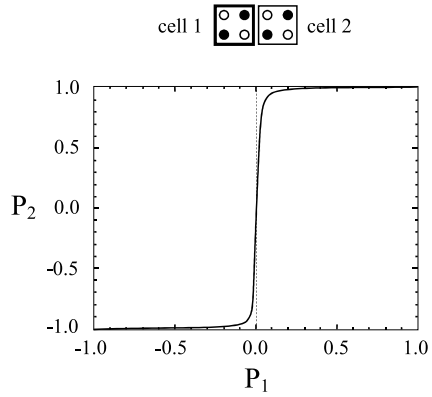

Fig. 2. Cell to cell response function for two neighboring QCA cells.

Conventional digital technologies use ranges of voltage or current to represent binary values. In contrast, QCA uses the position of electrons in quantum dots to represent binary values "0" and "1". QCA's primary advantages are the exceptionally high logic integration derived from the small size of dots, the simplicity and the notably low power-delay product [54].

The basic building block of QCA devices is the QCA cell presented in Fig. 1. It consists of four quantum dots in a square array coupled by tunnel barriers. The physical mechanisms for interactions between dots are the Coulomb interactions and quantum-mechanical tunneling. Electrons are able to tunnel between the dots, but cannot leave the cell. If two mobile electrons are placed in the cell, in the ground state and in absence of external electrostatic influence, Coulomb repulsion will force the electrons to dots on opposite corners [10]. The two possible charge configurations are presented in Fig. 1 and are corresponding to binary "1" and "0".

For an isolated cell, the two polarization states are energetically equivalent. However when a neighboring QCA cell is near, the equivalency breaks and only one of the two states becomes the cell ground state [10]. Polarization, P measures the extent to which the charge distribution is aligned along one of the diagonal axes. If we label the four dots from 1 to 4 anti-clockwise starting from the upper right dot of the cell, and assign ρ_i as the electron density of the ith dot, P is defined as:

$$P = \frac{(\rho_1 + \rho_3) - (\rho_2 + \rho_4)}{\rho_1 + \rho_2 + \rho_3 + \rho_4}. \quad (3)$$

The polarization of non-isolated cell is determined based on interaction between neighboring cells. Fig. 2 illustrates the cell to cell response function for two neighboring cells [54]. The figure's curve shows how the polarization P_2 of cell 2 changes for different values of the driver cell 1 polarization P_1.

The stability of a QCA circuit is based on the assumption that the system falls to the ground state every time it is excited by the inputs. However, this is not always guaranteed; so in this case the system settles in a metastable state, affecting the functionality of the design. This problem can

be solved by adiabatic switching [54]. In adiabatic switching the system is always kept in its instantaneous ground state by using a clocking scheme sequence of four periodic phases. QCA cells receive the clock signal through an electric field which can raise or lower the tunneling barrier between dots inside the cell. When the barrier is low the electrons can move from one dot to another according to the overall external electrostatic influence. In case of high barriers the electrons are locked inside the dots so the external fields can not change the state of the cell. The adiabatic switching clocking scheme consists of four phases: $Switch$, $Hold$, $Release$ and $Relax$.

At the beginning of the $Switch$ phase, the QCA cells in the zone are unpolarized since the cell tunneling barriers are low. During the switch phase the barriers are raised and the QCA cells become polarized according to the state of the cells that drive the zone. The driver cells must belong to a different clocking zone and specifically at the hold phase (90° phase difference, leading). Switch phase is the clock phase that the actual computation (or switching) occurs. At the end of switch phase, barriers are high enough to block any electron tunnelling and this locks the state of the cell. Next phase is the $Hold$ phase. During this phase the barriers remain high so the zone outputs can drive the inputs of the next clocking zone sub-circuit. The next zone (which is driven from our reference zone) must be at switch state (90° phase difference, leading). At the $Release$ phase, barriers are gradually lowered and finally cells are allowed to relax to an unpolarized state. Finally, during the fourth clock phase, the $Relax$ phase, cell barriers remain lowered and cells remain in an unpolarized state.

In order to use the adiabatic switching clocking scheme the QCA circuit must be partitioned into clocking zones in such a manner so that all cells in a clocking zone are controlled by the same clock. The clocking zone partitioning, is a crucial factor of the QCA design because the order of appearance of the four clocking zones in the circuit, controls the flow direction of the signals inside the QCA circuit.

A row of QCA cells acts like a wire usually called binary wire [11]. In QCA circuit designs of binary wires, inverters and three-input majority gates are the fundamental parts. The inverters are constructed with a fork structure and the majority gates with a cross structure [12]. Coplanar binary wire crossovers can also be implemented in QCA designs which is a very useful technique for designing more realizable circuits [12].

3 Proposed QCA Design Architecture

The objective of the proposed architecture is to implement a universal 1-D CA model circuit using QCA. To accomplish this objective, several versions of one CA cell QCA circuit implementations have been designed, which will be used as building tiles of the final CA model. As it will be shown all proposed versions of CA cell QCA implementations have exactly the same functional behaviour but they differ in signal manipulation in terms of interconnection and synchronization. Moreover it should be noticed that despite the fact that some designs found in literature as for example in [71] are using multi-layer wire crossovers, unfortunately till now there are still questions about how these multi-layer crossover designs can be realized in practice, since they require two overlapping active layers with via connections [72]. As a result, in the proposed QCA design architecture only coplanar crossovers are presented in all cases.

3.1 CA Cells Tile Design Principles

The 1-D CA functionality as described in the previous section, includes a row of CA cells in which every CA cell can communicate only with their two neighbours (west and east CA cell). Every CA cell has three inputs and one output. The output value of each CA cell at a given time is calculated from the previous time step output values of the two neighbouring cells and from the previous time step output value of the same cell, according to its Wolfram rule. For the CA cell designs of the proposed architecture, the mapping of every possible input value (coming from the neighbourhood cells) to its corresponding output, is implemented by an 8 to 1 multiplexer [17], [18]. The Wolfram rule the CA cell performs, is applied in its binary form, to the 8-bit input bus of the multiplexer and the 3-bit selection bus of the multiplexer is formed by: a) the output of the west neighbouring cell, b) the output of this cell and c) the output of the east neighbouring cell, ordered as mentioned, from most to least significant bit. So in all CA cell designs of the proposed architecture, the output signal of the 8 to 1 multiplexer is feedbacked to the 2nd bit of their selection bus and is leaded also to the neighbouring cells.

As it has been mentioned before the present state of each CA cell is evaluated according to the temporal previous states of their neighbouring cells. Because of that the design must have the initialization capability in order to start its operation. At the proposed architecture the initialization function is implemented by a 2 to 1 multiplexer across the feedback wire of the output signal. The multiplexer selects between the feedback and the initial value, to pass as input for the next CA cell calculation.

Very important factor for the CA cell designs to be functional correct at the proposed architecture, is that the output signals of the neighbouring cells must arrive simultaneously at the inputs of every CA cell. This is obvious and very simple to implement in conventional VLSI design, but not very easy to implement in a QCA design. The reason for that is that the signal propagation via interconnection binary wires in QCA technology, imports as much or even more delay than the processing elements of the design. So the delays imported by the processing elements and by the interconnection wires must be carefully summed for every parallel route, in order to check the synchronization of all the signals. Additionally a CA model is a parallel model in which the output states of the cells, all produced at the same time at every time step. So because of that, the input signals must not only be synchronized between the three CA cells of a local neighbourhood, but between all CA cells of the CA model. So because of that, the input signals must not only be synchronized between the three CA cells of a local neighbourhood, but between all CA cells of the CA model. This design requirement precludes the use of hybrid CA cell implementations [24] because in that case different ruled CA cells leads to QCA areas with significantly different dimensions and consequently the input signals mentioned above would not be synchronized.

At the proposed design architecture, every CA cell implementation includes three parallel routes. All three routes include the two multiplexer circuits described above so the delay inserted by the processing elements is equal for all routes. Consequently the synchronization problem is focused to equalize the delay inserted by the interconnection wires.

3.2 Routing and Placement

Generally the routing principles of QCA circuits are much different from these of the conventional VLSI circuits [70]. In QCA circuit design the processing by wire capability is widely used and many of the interconnection lines become processing elements for the circuit [63]. Processing by wire in QCAs, allows information manipulation while the signal propagates on a binary wire. A characteristic paradigm of processing by wire is presented in the inverter chain implementation [11].

Taking into account the previous statements, routing principles must be realized in combination with the placement of the design blocks which compose the QCA circuit. According to the proposed design architecture, every CA cell is implemented by a QCA design block. The whole CA model is constructed by placing all the QCA blocks (implementing CA cells) on specific places. The interconnection wires between these QCA blocks can easily destroy the synchronization of the whole circuit. To avoid this, no interconnection wires between blocks have been used. Special QCA blocks were designed so that they can fit one beside the other without using interconnecting wires. Additionally special care has been taken for the route of the feedback wire inside each CA block, because it must cross input and output wires of the cell, using the QCA coplanar crossing scheme [11]. It also must synchronize the output signal with the inputs receiving from the neighboring cells.

The block topology is strongly affected by the boundary conditions of the CA model. For zero boundary conditions, the inputs of the boundary CA cells which are not receiving signals from a neighbour are set to zero. In this case a N cell CA model is formed using three types of cell blocks which are placed forming a row. At this scheme the leftmost block named $bleft$, is taking inputs only from its right neighbour and puts a fixed zero value to its left neighbour input. The rightmost block named $bright$, is taking inputs only from its left neighbour and puts a fixed zero value to its right neighbour input. The remaining N-2 blocks named $down$, are taking inputs from their neighbours (Fig. 4). However in periodic boundary conditions the west input of the west boundary CA cell is evaluated from the output of the east boundary CA cell. The east input of the east boundary CA cell is evaluated from the output of the west boundary CA cell. In this case the CA cell blocks are placed forming a ring allowing thus the boundary CA cells to communicate as neighbours (Fig. 8).

3.3 QCA Design Robustness Issues

All CA model QCA designs, to be functional and robust, they must comply with a number of significant QCA design rules. Because of the propagation delays and power loss between cell-to-cell interactions, there should be a limit on the maximum signal propagation path length in each clock zone. Additionally, short wire lengths result in circuit operation in higher clock rates [14]. At nonzero Kelvin temperatures, higher operating temperatures lead to higher thermal fluctuations, which increase the probability for a kink to occur (a QCA cell to align differently from its expected polarization) [10], [55]. It has been proved [10], that the maximum QCA line length for kink-free operation is given by:

$$N \leq e^{\frac{E_k}{k_b T}}, \tag{4}$$

where E_k represents the energy required for a QCA cell to encounter a kink, k_b is the Boltzmann's constant and T is the temperature of operation. So it can be also implied that a QCA circuit having short wire lengths can operate without kinks at higher temperatures than a QCA circuit with longer wire lengths. Short wire lengths result to small clocking zones which may lead to more complicated clock distribution circuitry. In the proposed architecture the two-dimensional wave clocking scheme [17] is used to simplify clocking circuitry distribution using parallel clocking wires. The above clocking scheme also facilitates the clock distribution circuitry [17].

All QCA circuits implemented using the presented architecture have maximum wire length under 14 QCA cells much less than the acceptable limit of 28 QCA cells wire length for 1 THz clock rate proposed at [56], [64], so that taking into account equation 4 they can operate avoiding kinks at higher temperatures. Another design requirement concerning interactions between different signals from nearby wires is the minimum wire spacing. All designs proposed in this paper are following the minimum wire spacing limit of two (2) cells proposed in [64].

3.4 8 to 1 Multiplexer and CA Cell QCA Implementation

As have been mentioned in previous section, the 8 to 1 multiplexer design is the key component of the CA model QCA design architecture. The proposed QCA implementation of the 8 to 1 multiplexer with the feedback path forming the CA cell is illustrated in Fig 3. The signal processing flow inside the proposed QCA circuit starts from the upper left corner and propagates on a diagonal profile to the bottom right corner, where the output is produced implementing a 2-D clocking scheme [16]. The eight input signals of the multiplexer, corresponding to the 8-bit Wolfram rule, are crossing the circuit through horizontal binary wires and the selection bus input signals are crossing the circuit vertically through inverter chains forming an 8×3 wire grid. The QCA coplanar crossing scheme is taking place on every crossing point of the previous mentioned wires. Additionally next to every crossing point, an AND gate between the two crossing signals is implemented, in order for the selection signals to enable or disable the horizontal propagation of the input signals. Only one of the eight input signals is allowed to propagate horizontally to the right edge of the grid and this signal is forwarded to the multiplexer's output trough a sequence of OR gates. Then the output signal is forwarded through binary wires: (a) to selection bus least significant bit of the west neighbouring CA cell, (b) to selection bus most significant bit of the east neighbouring CA cell and (c) to the selection bus second bit of this CA cell. The

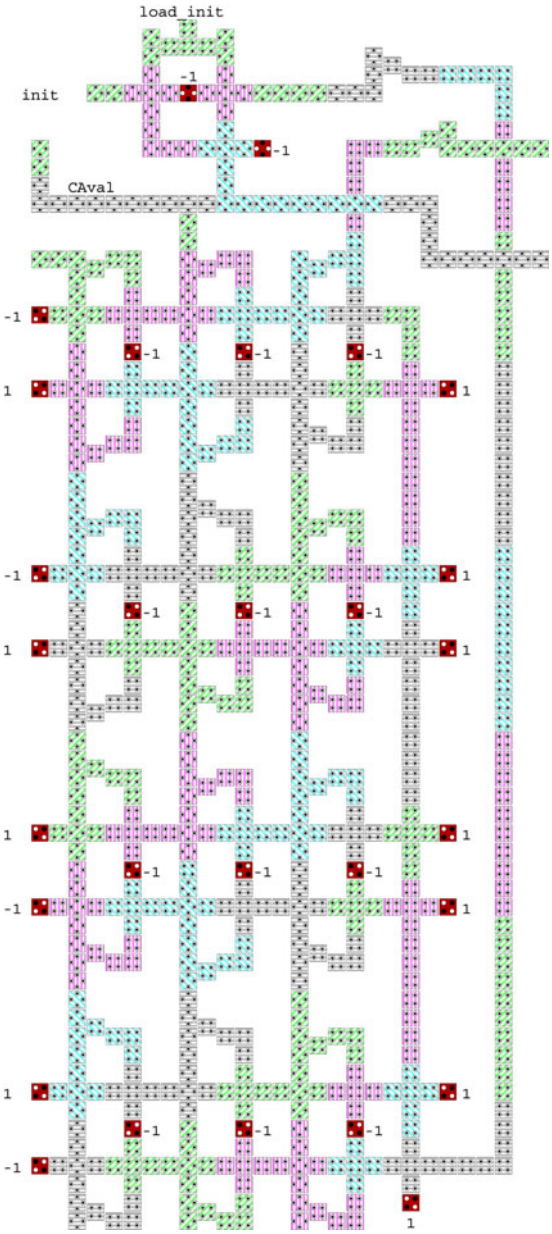

Fig. 3. QCA implementation of the basic CA cell which consists of an 8 to 1 multiplexer and the feedback path of its output to the CA cell's input through a 2 to 1 multiplexer.

signal in this feedback path is crossing through a 2 to 1 multiplexer. The selection bit (*load_init*) of the 2 to 1 multiplexer decides if the feedback signal or the CA cell initialization signal (*init*) will pass to the second bit input of the CA cell.

3.5 CA Cell Tiles QCA Implementations

In order to apply the proposed design architecture as described previously, several versions of the basic CA cell QCA implementations must be implemented. These implementations for zero and periodic boundary conditions CA model are presented below.

3.5.1 Zero Boundary Conditions

In case of zero boundary conditions, as have been mentioned before CA cell blocks are placed on a row and three types of CA cell blocks are used (Fig. 4). The leftmost block

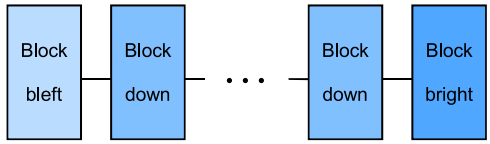

Fig. 4. Block diagram for the CA model implementation with zero boundary conditions.

named $bleft$ which is taking inputs only from its right neighbour. The rightmost named $bright$ which is taking inputs only from its left neighbour. And the block named $down$ which is taking inputs from both its neighbours. Every block includes a QCA circuit which is formed by an arrangement of QCA cells on the grid. In the proposed architecture, the QCA implementations of all three blocks have the same height in rows and communicate with each other only through row 53 and 59 on the grid counting from bottom to top. More specifically, the $down$ block receives the output of its left neighbour from the left edge of row 53 and sends its output to its right neighbour through right edge of row 53. The $down$ also receives the output of its right neighbour from the right edge of row 59 and sends its output to its left neighbour through left edge of row 59. The $bleft$ block receives only the output of its right neighbour from the right edge of row 59 and sends its output only to its right neighbour through right edge of row 53. And the $bright$ block receives only the output of its left neighbour from the left edge of row 53 and sends its output only to its left neighbour through left edge of row 59. Using this input/output signal arrangement it is obvious that if the blocks are placed on a horizontal row according to Fig. 4 the implemented CA cells will communicate successfully forming a CA model. The model outputs its next state every six clock cycles or 24 clock phases, which is the time every CA input signal needs to propagate to its neighbouring CA cell input. It should be stated that this propagation time is independent of the length of the CA model.

All blocks include the initialization input signals $init$ and *load_init* in order to set the initial state of the CA cell. The $init$ signal defines the value of the initial state and the *load_init* signal controls when the initialization will be done. The 8-bit CA rule is representing by the 8 fixed valued QCA cells on the left side of the block, which are evaluating the multiplexer's input bus.

Fig. 5 depicts the QCA implementations of the three blocks required as tiles by the proposed CA model QCA design architecture in the case of zero boundary conditions. All the QCA circuits are using a 2-D clocking scheme [16], [17], [18] propagating the signals from the left up to the right down design's corner on a diagonal profile as mentioned before. The QCA circuits were designed and simulated using the QCA Designer tool. In the environment of the previous mentioned tool, the overall QCA cell dimensions are defined to 18×18 nm^2, the dot diameter to 5 nm and the inter-cell distance to 2 nm.

According to the QCA Designer tool the design of the $bleft$ and $down$ blocks consists of 670 and 676 cells, respectively and they are both covering an area of 28×65.5 QCA grid cells or 558×1308 nm^2 that is approximately 0.73 μm^2. The $bright$ block design consists of 639 cells covering an area of 26×65.5 QCA grid cells or $518 \times 1{,}308$ nm^2 that is

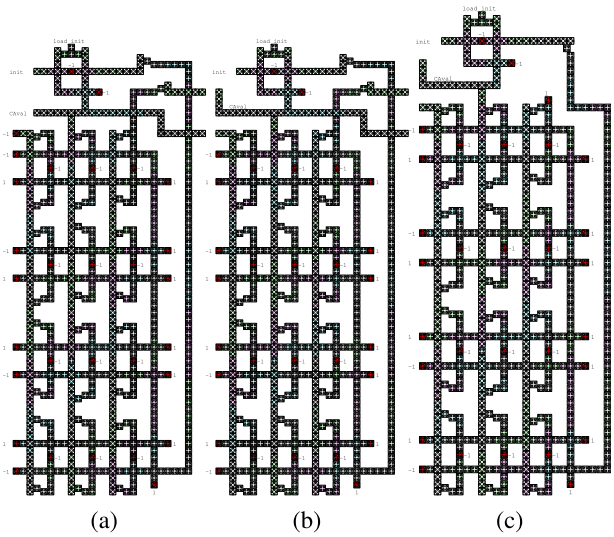

Fig. 5. QCA implementations of the three blocks used by the tool in the case of zero boundary conditions: a) $bleft$ block, b) $down$ block, c) $bright$ block.

approximately 0.68 μm^2. All QCA implementations of the proposed CA blocks were simulated exhaustively using the QCA Designer simulator to confirm their functional correctness. For the simulation, the Euler Method, coherence vector simulation engine of the QCADesigner tool is been used, with Relaxation time 1e^{-15} sec, Time step 1e^{-16} sec, Radius of Effect 80 nm and 1 THz clock rate, which are the proposed values of the Simulator. The simulation input vectors are generated automatically by the DATICAQ tool taking into account the data entered in the control panel of the user interface.

A paradigm of a QCA design for a five (5) cells CA model which uses rule 30, implemented according to the proposed architecture, is presented in Fig. 6. The resulting design consists of 3,337 cells covering an area of $2{,}758 \times 1{,}308$ nm^2 that is approximately 3.61 μm^2. It is covering an area of 138×65.5 grid cells and the ratio of the area covered by QCA cells to the overall area of the layout is about 0.369 more than $1/3$.

In general by using the here proposed architecture, for a QCA design implementing a CA model consisting of n number of CA cells, in case of zero boundary conditions, the area that it covers in grid cells can be calculated using the equations:

$$\begin{aligned} Circuit\ Area &= CircuitWidth \times CircuitHeight \\ &= (bleftWidth + (n-2) \times downWidth + brightWidth) \\ &\quad \times downHeight = (28 + (n-2) \times 26) \times 65.5 \end{aligned} \tag{5}$$

which eventually equals to:

$$Circuit\ Area = 1{,}834 \times n - 131 \tag{6}$$

and the number of the QCA cells of the design can be calculated using the equation:

$$\begin{aligned} &Number\ of\ QCA\ cells \\ &\quad = bleftCells + (n-2) \times downCells + brightCells \\ &\quad = 670 + (n-2) \times 676 + 639 \end{aligned} \tag{7}$$

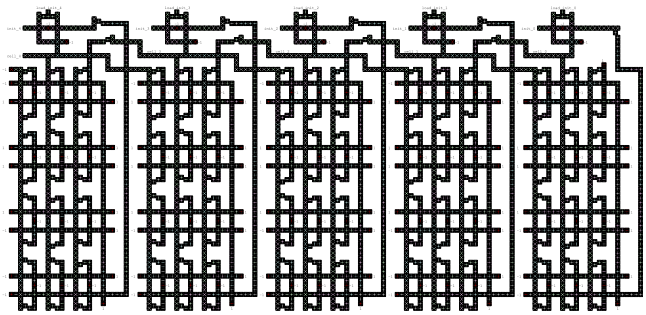

Fig. 6. The QCA design of a five cells zero boundary conditions CA model.

which results to:

$$Number\ of\ QCA\ cells = 676 \times n - 43. \tag{8}$$

The ratio (R) of the area covered by QCA cells to the overall area of the circuit layout can be calculated using the following equation:

$$R = (676 \times n - 43)/(1{,}834 \times n - 131), \tag{9}$$

where n is the number of the CA cells.

The simulation results of the design, acquired by the QCA Designer coherence vector simulation engine, are presented in Fig. 7. The input signals are presented with blue

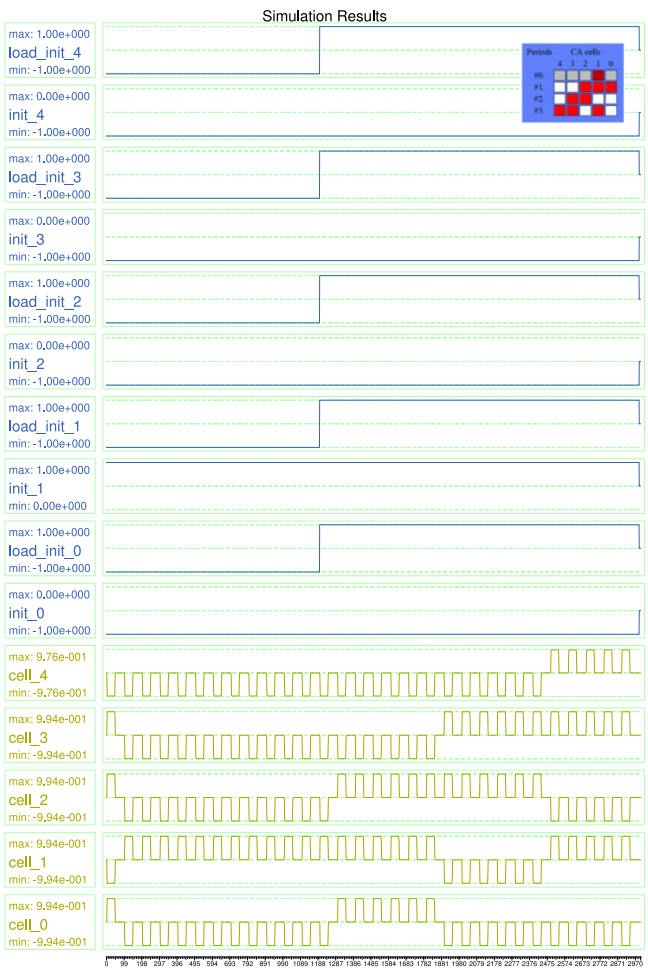

Fig. 7. Simulation results of the QCA circuit for a five cells CA model with zero boundary conditions.

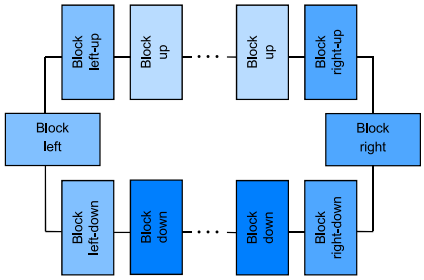

Fig. 8. The block placement diagram in case of periodic boundary conditions.

waveforms and the output signals with brown waveforms. In the beginning of the simulation, all the *load_init* signals are set to zero in order to set the initial state of the CA model 00010 using the values of the $init$ signals. At the next time step the *load_init* signals are set to one directing the circuit to operate in normal mode which is the CA evolution process with Wolfram rule 30. The circuit outputs the next state of the CA model every six clock cycles. At the right upper corner of the figure, a frame is appearing which presents the expected states of the corresponding 1-D CA model through the evolution process.

3.5.2 Periodic Boundary Conditions

In case of periodic boundary conditions the CA cell blocks are placed on a ring in such a way so that every CA cell communicates with its two neighbors. The block placement architecture used by the tool in the case of periodic boundary conditions is presented in Fig. 8. The ring shape is constructed by two curves (left and right) and two rows (up and down). The left curve is composed of the blocks $left$, $left\text{-}up$ and $left\text{-}down$ and the right curve is composed of the blocks $right$, $right\text{-}up$ and $right\text{-}down$ as shown in Fig. 8. The up row is composed of n identical blocks named up and the down row is composed of n identical blocked named $down$ and are exactly same with the $down$ blocks the system uses in the case of zero boundary conditions. The number of the blocks in up and down rows must be the same in order for the ring structure to be completed.

Because of the circuit request described in Section 3.2 about the synchronization of CA cells' output signals, the wires carrying the CA cells' input/output signals must be as short as possible. Consequently the communication ring formed by the CA cells must be as short as possible. In more details and in order for the architecture to comply with this requirement, the up-row blocks must provide their communication signals near their bottom side, the down-row blocks must provide its communication signals near their top side, the left-curve blocks must provide its communication signals near their right side and the right-curve blocks must provide their communication signals near their left side.

Implementing the above scheme in the proposed architecture, the QCA implementations of the blocks $left\text{-}down$, $down$, $right\text{-}down$, $left\text{-}up$, up and $right\text{-}up$ all have the same height in rows. The $down$ and $left\text{-}down$ blocks receive neighbour signals from row 53 left-side and row 59 right-side, respectively and send their signals through row 59 left-side and row 53 right-side counting from bottom to top, as described previously. The $right\text{-}down$ block receives neighbour signals from row 53 left-side and column 9 top-side and sends its signal through row 59 left-side and column 3 top-side, counting from bottom to top and from right to left. The up and $right\text{-}up$ blocks receive neighbour signals from row 59 left-side and row 53 right-side and send their signals through row 53 left-side and row 59 right-side counting from top to bottom. The $left\text{-}up$ block receives neighbour signals from row 53 right-side and column 9 bottom-side and sends its signal through row 59 right-side and column 3 bottom-side, counting from top to bottom and from left to right. Finally, the $left$ block receives neighbour signals from column 53 top-side and row 7 right-side and sends its signal through column 59 top-side and row 1 right-side, counting from bottom to top and from left to right, while the $right$ block receives neighbour signals from column 53 bottom-side and row 7 left-side and sends its signal through column 59 bottom-side and row 1 left-side, counting from top to bottom and from right to left, respectively. The model outputs its next state every six clock cycles or 24 clock phases, as in the case of zero boundary conditions and the propagation time is also independent from the length of the CA model.

All blocks include the initialization signals $init$ and *load_init* and the CA rule is represented by a row of fixed valued QCA cells setting the input bus of the 8 to 1 multiplexer as have been described before for the case of zero boundary conditions. Using the above input/output signal arrangement, if the blocks are placed on the ring schema of Fig. 8 they will communicate successfully forming a CA model.

The QCA implementations of the eight blocks required as tiles by the proposed CA model QCA design architecture in case of periodic boundary conditions are represented in Fig. 9. For all block implementations it can be seen that the feedback's path delay is 14 clock phases and is equal to the delay of paths carrying CA cell's output to the neighbouring cells. The QCA circuits were designed and simulated using the QCA Designer tool with the same parameters as in the case of zero boundary conditions. According to the QCA Designer tool the design of the tool the blocks $left$, $left\text{-}up$, $right$ and $right\text{-}down$ consist of 669, 665, 669 and 665 QCA cells, respectively and they are covering an area of 26×65.5 QCA grid cells or $518 \times 1{,}308$ nm^2 that is approximately 0.68 μm^2. The blocks up, $right\text{-}up$, $down$ and $left\text{-}down$, all consist of 675 QCA cells and they are covering an area of 28×65.5 QCA grid cells or $558 \times 1{,}308$ nm^2 that is approximately 0.68 μm^2. All QCA blocks were simulated successfully using the QCA Designer Simulator with the same parameters mentioned before.

Fig. 10 presents a QCA layout of a ten cell circuit that models CA rule 90 with periodic boundary conditions. The circuit consists of 6,718 cells covering an area of $4{,}458 \times 2{,}818$ nm^2 that is approximately 12.56 μm^2. It is covering an area of 223×141 grid cells and the ratio of the area covered by QCA cells to the overall area of the layout is about 0.214.

The simulation results of the aforementioned circuit design, acquired by the QCA Designer coherence vector simulation engine and in accordance with the previous ones mentioned in case of zero boundary conditions, are presented in Fig. 11. In this figure the upper eleven

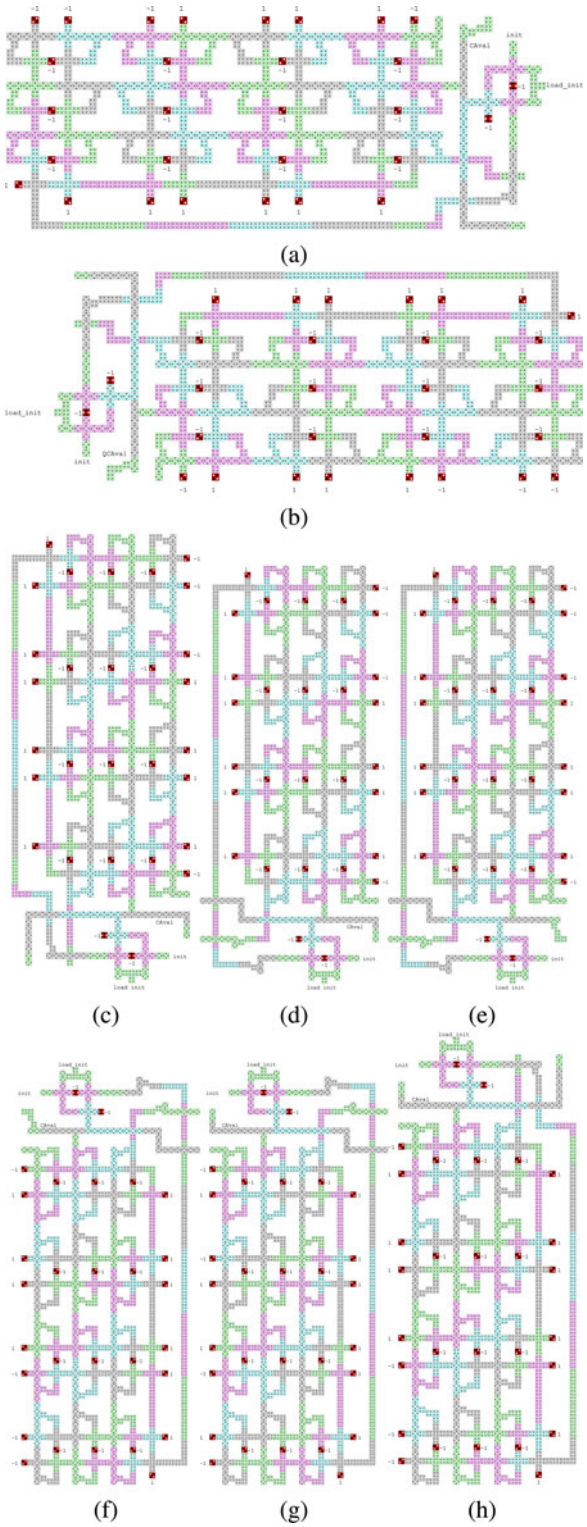

Fig. 9. QCA implementations of the three blocks used by the tool in case of zero boundary conditions: a) $left$ block, b) $right$ block, c) $left{-}up$ block, d) up block, e) $right{-}up$ block, f) $left{-}down$ block, g) $down$ block, h) $right{-}down$ block.

waveforms correspond to input signals, and the remaining ten waveforms, presented in different color, correspond to the 10 CAs' output signals. The four clock signals and the *load_init* signals are omitted intentionally for readability reasons. For the simulation all *load_init* signals have the same shape like the *load_init_0* which is presented.

At the simulation time, firstly all the *load_init* signals are set to zero in order to set the initial state of the CA model to 0000000100 using the values of the init signals from *load_init_9* (msb) to *load_init_0* (lsb). At the next time step the *load_init* signals are set to one directing the circuit to operate in normal mode which is the CA evolution process with Wolfram rule 90. The circuit gives the next state of the CA model every six clock cycles. After a new state appears on the outputs, the signals remain stuck at their current values until the next state arrives on circuit outputs. At the right upper corner of the Fig. 9, a frame is appearing which presents the expected states of the corresponding 1-D CA model through the evolution process.

In general, for a QCA design produced according to the proposed architecture and implementing a CA model with n CA cells in case of periodic boundary conditions, the area that it covers in grid cells can be calculated using the following equations:

$$\begin{aligned} Circuit\ Area &= CircuitWidth \times CircuitHeight \\ &= (left.leftup.CornerWidth + [(n-6)/2] \times upWidth \\ &\quad + rightup.rightCorner.Width) \times (upHeight \\ &\quad + blanc.spaceHeight + downHeight) \\ &= (76 + [(n-6)/2] \times 28 + 91) \times 65.5 + 10 + 65.5 \end{aligned} \tag{10}$$

which eventually equals to

$$Circuit\ Area = 1{,}974 \times n + 11{,}703. \tag{11}$$

The number of the QCA cells of the design can be calculated using the equations:

$$\begin{aligned} &Number\ of\ QCA\ cells \\ &\quad = leftCells + leftupCells + leftdownCells \\ &\qquad + [(n-6)/2] \times upCells + [(n-6)/2] \times downCells \\ &\qquad + rightCells + rightupCells + rightdownCells \\ &\quad = 669 + 665 + 675 + 670 + [(n-6)/2] \times 675 \\ &\qquad + 669 + 675 + 665 \end{aligned} \tag{12}$$

or in another words:

$$Number\ of\ QCA\ cells = 675 \times n - 32. \tag{13}$$

The ratio (R) of the area covered by QCA cells to the overall area of the circuit layout can be calculated using the equation:

$$R = (675 \times n - 32)/(1{,}974 \times n - 11{,}703), \tag{14}$$

where n is the number of the CA cells.

4 *DATICAQ*: Tool Description

The $DATICAQ$ is an automated CA design tool using QCA technology as an implementation medium. It can automatically create the QCA design layout of any binary 1-D CA. No manual designing or programming is required because the CA model is created and simulated using a user-friendly graphical user interface (GUI). The proposed tool creates the QCA design layout and simultaneously creates the QCA design simulation test vectors in order to make it easy for the

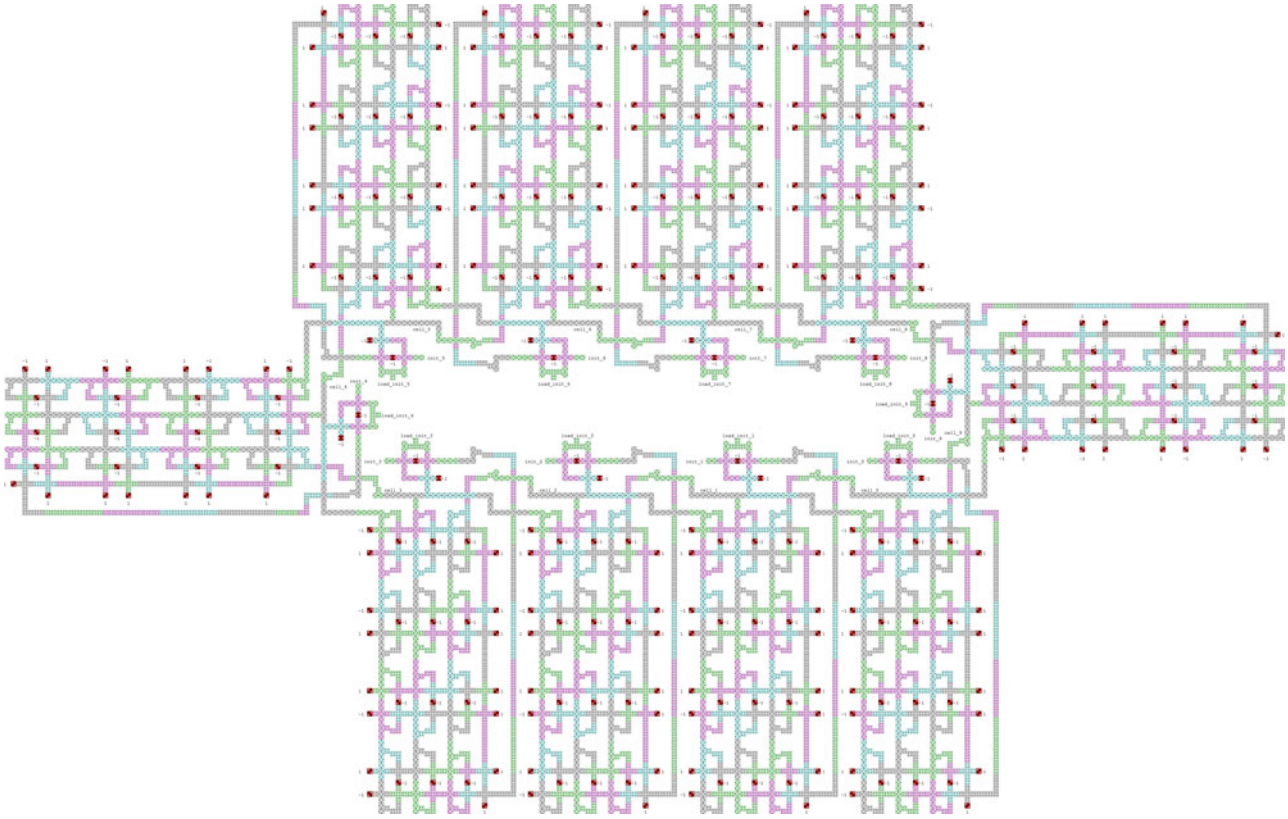

Fig. 10. The QCA design of a 10 cells periodic boundary conditions CA model.

tool user to compare the results of the simulated CA model to the results of the QCA design simulation. *DATICAQ* is a web-based application which has been implemented using the PHP programming language [73]. The proposed tool is open source and platform independent. The tool can run on any web server and is not requiring the installation of any

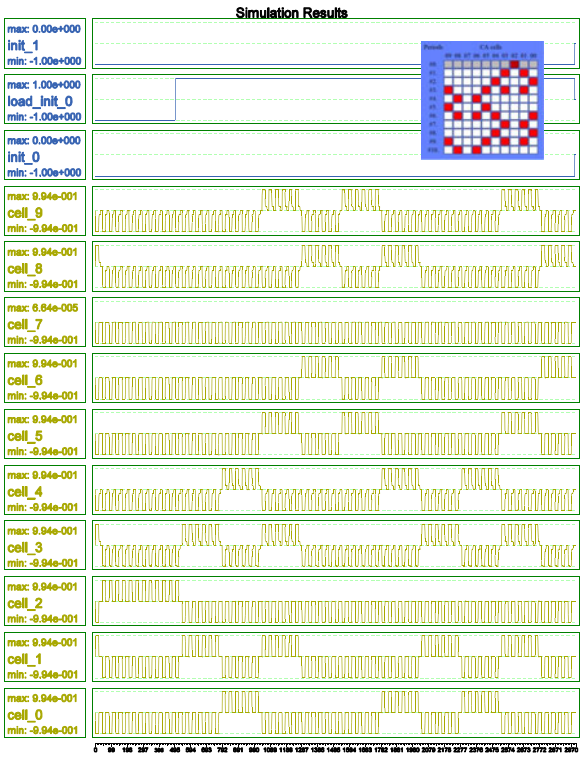

Fig. 11. Simulation results of the QCA circuit for a 10 CA cell model with periodic boundary conditions. Please notice that all the *init* signals, here omitted intentionally for readability reasons, are found at '0' with the exception of *init_2* which is found at '1'.

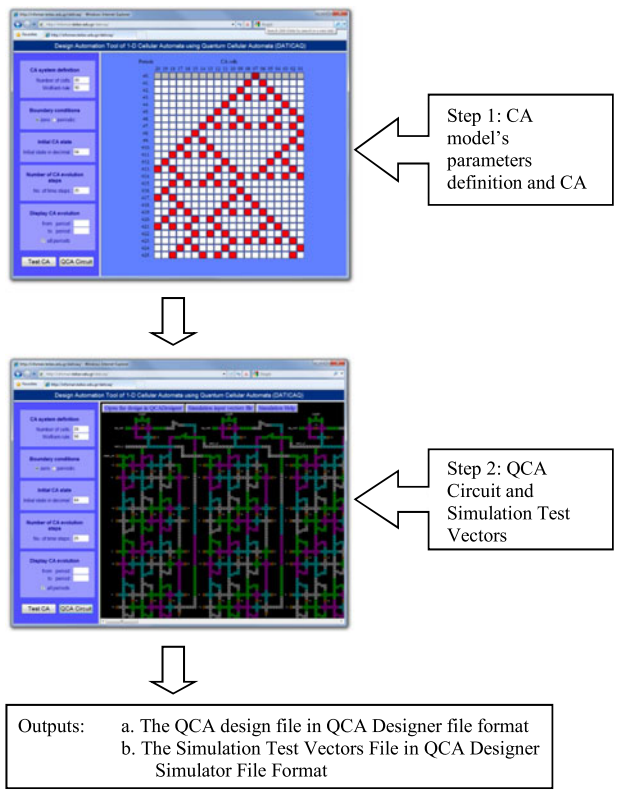

Fig. 12. Functionality block diagram of DATICAQ tool.

application. *DATICAQ* can be used via any computer connected to the Internet.

The proposed tool produces automated QCA designs in two steps as presented in Fig. 12. In the first step the requested CA model has to be defined and simulated in order to confirm the model functionality. In the second step the QCA design of CA's model is automatically produced and at the same time the test vectors for functional verification of the QCA design are produced. More specifically, in the first step the user must define the number of cells which compose the CA model to be designed, the $WolframRuleNumber$ [33] to be used for the CA evolution and the type of boundary conditions of the CA model. Furthermore, the initial state of CA cells, the number of evolution steps and the evolution period to be displayed must also be defined to enable the tool to simulate and display the simulation results of the requested CA model. By pressing Test CA button in the GUI, the simulation results of the defined CA model are presented in the right window of *DATICAQ*'s interface.

The simulation results are presented on a grid in which, the first row represents the initial state and the following rows present the time evolution of the CA model. In the grid presentation, value "0" corresponds to white color cells, while value "1" corresponds to red color cells. For readability reasons, the initial state of the simulation is presented in gray color.

The second step includes the main functionality of the tool which is the production of the automated QCA design. In order for the tool to produce the automated CA model's QCA circuit design, the previously presented architecture has been used.

The proposed tool can implement binary 1-D CA models using any Wolfram rule which is inputted by the tool's user in the first step of design's production. Because of this, CA cell block designs are implemented dynamically by a *PHP* function in order to produce adaptive CA cells using any requested rule. More specifically the designs of the different CA cell blocks are described in text files using a special format presented below. Every QCA cell in the design is located on one row in the text file. For every QCA cell, the following nine descriptors are used to describe their properties:

1) The horizontal position of the QCA cell from the left edge of the design, in nanometers.
2) The vertical position of the QCA cell from the top edge of the design, in nanometers.
3) A number from 0 to 3 representing one of the four clock zones $clk0, clk1, clk2$ or $clk3$, respectively, which controls the tunneling energy barriers of the cell.
4) The type of the QCA taking one of the following values: *QCAD_CELL_NORMAL* for a normal cell, *QCAD_CELL_INPUT* for an input cell, *QCAD_CELL_OUTPUT* for an output cell, *QCAD_CELL_FIXED* for a cell with fixed value.
5) The polarization for the QCA cells with fixed values. The value of this descriptor can be 0, 1 or $rule0$ to $rule7$. The value $rule0$ represents the least significant bit and the $rule7$ the most significant bit of the requested Wolfram's rule byte.
6) A flag value of 1 or 0 indicates if the cell is rotated by 45° or not.
7) A text value for the label of the QCA cell.
8) The horizontal position of the label.
9) The vertical position of the label.

The final QCA layout is presented in the right frame of the tool's interface and additionally three buttons appear on the top of the frame. By clicking on the first button, the QCA layout of the produced circuit can be downloaded in QCADesigner format [57]. With the second button, a stimuli file is produced containing a complete set of simulation vectors for the circuit, according to the initial CA state and the number of CA evolution step values that have been given by the user in the first step of the *DATICAQ* operation. Consequently, the stimuli file complies with the requirements of the QCADesigner Simulator [57] in order to provide the user with an easy way to simulate the produced circuit. By clicking on the third button appearing in the frame, the tool provides to the user a useful simulation instructions help window, in order to take advantage of the produced files without making any mistakes.

Finally, the realization of hybrid CA models [58] can be succeeded by creating a file which provides the mapping between CA cells and the corresponding rules. *DATICAQ* reads the mapping information from the file and builds the appropriate QCA circuit.

Comparing CA implementations using QCA to CA implementations using conventional CMOS technology we can be confident to say that QCA implementation exceeds CMOS implementation in terms of area and operation frequency [59]. The QCA implementation can operate near 1 THz against 0.35μm CMOS implementation which can operate near 120 MHz. The area covers a CA implementation using QCA is 0.68 to 0.73 and the area covers a CA implementation using 0.35 μm CMOS technology is near 100 μm [59]. Moreover, we have designed the analogous 1-D CA grid with rule 90 described in subsection 3.5.2 by using 0.13 μm CMOS technology and the area covered by this CA implementation in the specific CMOS technology is near 62 μm while its clock frequency is less than 1 GHz. The above superior results of CA design with QCA when compared with CMOS technology can be also confirmed with other CMOS CA implementations like the ones found in [60], [61]. Even more recent CA implementations of the past three years like a single CA crowd evacuation model in field programmable gate array (FPGAs) [68] by using an Altera Stratix device of 0.13 μm, or a dedicated FPGA CA processor for modeling wildfire by using Altera Stratix-V of 40 nm [69] or a massively parallel implementation for a floating-point-based CA in FPGA cluster [67] by using Xilinx Virtex-4 FPGAs does not provide any significant differences in terms of timing operation, while in area configuration due to FPGA special purposes they also came short when compared to the above QCA designs. In general, regarding the benefits in terms of calculated power dissipation for the molecular-scale QCA design like the one proposed in this paper, compared to existing and projected technology, these can be found in [62]. There the power dissipated per device versus propagation delay for QCA and its relation to complementary metal-oxide-semiconductor CMOS technology are given analytically. The upper bound

for the QCA region is based on the worst case scenario wherein every device performs a logically irreversible function and thus switches nonadiabatically, dissipating an amount of energy of order E_k during a switching period. While comparisons between proposed devices and existing devices should be made cautiously, these data indicate that QCA may offer a way to achieve the ultralow power dissipation required for molecular scales of integration.

5 Conclusions

In this paper a logic design methodology resulting to a new QCA design architecture and the corresponding design tool, are presented that can produce the QCA layout for any 1-D CA model. The example QCA layouts and the simulation results have been provided in terms of metal or semiconductor implementation logic. According to very recent work [74], this implementation logic seems to be fabricable at approximately 2×2 nm QCA cell dimensions using Silicon Atomic Quantum Dots and can operate at room temperature. In spite of that, the basic principles of the proposed methodology are compatible and can be easily applied on molecular or magnetic implementations, since the 2-D wave clocking scheme [16] that the proposed methodology uses, can be applied to these implementations. Most recently QCA research is focusing on molecular [75], [76] and magnetic implementations [77], [78]; as a result, in future work the presented methodology can be altered in order to produce designs based on new clocking schemes that have been proposed for molecular or magnetic implementations [79], [80], [81], [82]. The proposed architecture and the presented tool, named *DATICAQ* [73], offer the power of automated modelling and QCA implementation of CAs, while the latter is an interactive web-based tool hiding architecture and programming issues from the user. It can model dynamic complex phenomena and produce automatically the QCA layouts in QCADesigner format. The inputs to the presented architecture are the CA dimensionality, size, local rule, and the initial and boundary conditions imposed by the particular problem. Due to the friendly interface of the proposed systems no prior knowledge of QCA designing is required by the user. Research workers with different scientific backgrounds that use CAs and have no specific knowledge of either software programming or QCA designing are able to use *DATICAQ*. Furthermore, the proposed system helps the user to easily test several CAs and decide which one is appropriate for the problem in hand. Having in mind that 1-D CA and 2-D CA have been proven equally really efficient for VLSI design [58], [59], [60], [61], [67], [68], [69], [83], in our future work, research will be focused on finding possible extensions of the proposed architecture in order to produce 1-D CA models with larger neighborhoods as well as to introduce CA implementations in larger dimensions, i.e. in 2-D and 3-D CA models, enabling the depiction of several CA applications in modern nano-architectures.

References

[1] (2007). International Technology Roadmap for Semiconductors (ITRS). [Online]. Available: http://www.itrs.net

[2] C. Wasshuber, "About single electron devices and circuits," PhD dissertation, Faculty Elect. Eng. Inf. Technol., Tech. Univ. Wien, Wien, Germany, 1997.

[3] T. Oya, Y. Takahashi, M. Ikebe, T. Asai, and Y. Amemiya, "A single-electron circuit as a discrete dynamical system," *Superlattice Microstruct.*, vol. 34, pp. 253–258, 2003.

[4] I. G. Karafyllidis, "Design and simulation of a single-electron random-access memory array," *IEEE Trans. Circuits Syst. I, Fundam. Theory Appl.*, vol. 49, no. 9, pp. 1370–1375, Sep. 2002.

[5] J. M. Wang, "Simulation and design of nanocircuits with resonant tunnelling devices," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 54, no. 6, pp. 1293–1304, Jun. 2007.

[6] G. Zardalidis and I. G. Karafyllidis, "SECS: A new single-electron-circuit simulator," *IEEE Trans. Circuits Syst. I., Reg. Papers*, vol. 55, no. 9, pp. 2774–2784, Oct. 2008.

[7] J. Hoekstrag, "On circuit theories for single-electron tunneling devices," *IEEE Trans. Circuits Syst. I., Reg. Papers*, vol. 54, no. 11, pp. 2353–2359, Nov. 2007.

[8] C. Lent, B. Isaksen, and M. Lieberman, "Molecular quantum-dot cellular automata," *J. Amer. Chem. Soc.*, vol. 125, pp. 1056–1063, 2003.

[9] Y. Lu and C. S. Lent, "Theoretical study of molecular quantum dot cellular automata," *J. Comput. Electron.*, vol. 4, nos. 1/2, pp. 115–118, 2005.

[10] C. S. Lent, P. D. Tougaw, W. Porod, and G. H. Bernstein, "Quantum cellular automata," *Nanotechnology*, vol. 4, no. 1, pp. 49–57, Jan. 1993.

[11] C. S. Lent, P. D. Tougaw, "Lines of interacting quantum-dot-cells: A binary wire," *J. Appl. Phys.*, vol. 74, no. 10, pp. 6227–6233, 1993.

[12] P. D. Tougaw and C. S. Lent,"Logical devices implemented using quantum cellular automata," *J. Appl. Phys.*, vol. 75, no. 3, pp. 1818–1825, 1994.

[13] W. Wang, K. Walus, and G. A. Jullien, "Quantum-dot cellular automata adders," in *Proc. IEEE Int. Conf. Nanotechnol.*, Los Angeles, CA, USA, 2003, vol. 2, pp. 461–464.

[14] K. Kim, K. Wu, and R. Karri, "The robust QCA adder designs using composable QCA building blocks," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 26, no. 1, pp. 176–183, Jan. 2007.

[15] 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.

[16] V. Vankamamidi, M. Ottavi, and F. Lombardi, "Two-dimensional schemes for clocking/timing of QCA circuits," *IEEE Trans. Comput. Aided Design Integr. Circuits Syst.*, vol. 27, no. 1, pp. 34–44, Jan. 2008.

[17] V. A. Mardiris and I. G. Karafyllidis, "Design and simulation of modular 2^n to 1 quantum-dot cellular automata (QCA) multiplexers," *Int. J. Circ. Theor. Appl.*, vol. 38, no. 8, pp. 771–785, Oct. 2010.

[18] V. A. Mardiris and I. G. Karafyllidis,"Design and simulation of modular quantum-dot cellular automata multiplexers for memory accessing," *J. Circuits, Syst. Comput.*, vol. 19, no. 10, pp. 349–365, 2010.

[19] J. Huang, M. Momenzadeh, and F. Lombardi, "Analysis of missing and additional cell defects in sequential quantum-dot cellular automata," *Integr., the VLSI J.*, vol. 40, pp. 503–515, 2007.

[20] A. S. Shamsabadi, B. S. Ghahfarokhi, K. Zamanifar, and N. Movahedinia, "Applying inherent capabilities of quantum-dot cellular automata to design: D flip-flop case study," *J. Syst. Archit.*, vol. 55, no. 3, pp. 180–187, Mar. 2009.

[21] M. T. Niemier, M. J. Kontz, and P. M. Kogge, "A design of and design tools for a novel quantum dot based microprocessor," in *Proc. 37th Design Autom. Conf.*, Los Angeles, CA, USA, 2000, pp. 227–232.

[22] V. Vankamamidi, M. Ottavi, and F. Lombardi, "A serial memory by quantum-dot cellular automata (QCA)," *IEEE Trans. Comput.*, vol. 57, no. 8, pp. 606–618, May 2008.

[23] B. Taskin and Bo Hong, "Improving line-based QCA memory cell design through dual phase clocking," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 16, no. 12, pp. 1648–1656, Dec. 2008.

[24] M. Bubna, S. Mazumdar, S. Roy, and R. Mall, "Designing cellular automata structures using quantum-dot cellular automata," presented at the 14th Annu. IEEE Int. Conf. High Performance Comput., Goa, India, Dec. 2007.

[25] T. T. Teodosio, "Quantica: Arquitecturas e simulacao de operacao de dispositivos," MSc thesis, Dept. Electr. Comput. Eng., Tech. Univ. Lisbon, Lisbon, Portugal, Sep. 2007.

[26] R. Singhal, *Logic Realization Using Regular Structures in Quantum-Dot Cellular Automata (QCA)*. Ann Arbor, MI, USA: ProQuest/UMI Dissertation Publishing, 2012.

[27] A. Orlov, R. Kummamuru, R. Ramasubramaniam, C. Lent, G. Bernstein, and G. Snider, "Clocked quantum-dot cellular automata shift register," *Surface Sci.*, vol. 532, pp. 1193–1198, 2003.

[28] Y. Lu, M. Liu, and C. Lent, "Molecular quantum-dot cellular automata: From molecular structure to circuit dynamics," *J. Appl. Phys.*, vol. 102, no. 3, pp. 034311-1–034311-7, 2007.

[29] J. Jiao, G. Long, L. Rebbouh, F. Grandjean, A. Beatty, and T. Fehlner,"Properties of a mixed-valence (Fe II)2(Fe III)2 square cell for utilization in the quantum cellular automata paradigm for molecular electronics," *J. Amer. Chem. Soc.*, vol. 127, no. 50, pp. 17819–17831, 2005.

[30] M. Baseer Haider, J. L. Pitters, G. A. DiLabio, L. Livadaru, J. Y. Mutus, and R. A. Wolkow, "Controlled coupling and occupation of silicon atomic quantum dots at room temperature," *Phys. Rev. Lett.*, vol. 102, pp. 46805–46808, 2009.

[31] J. Von Neumann, *Theory of Self-Reproducing Automata*. Urbana, IL, USA: Univ. Illinois Press, 1966.

[32] R. P. Feynman, "Simulating physics with computers," *Int. J. Theor. Phys.*, vol. 21, nos. 6/7, pp. 467–488, 1982.

[33] S. Wolfram, *Theory and Applications of Cellular Automata*. Singapore: World Scientific, 1986.

[34] T. Tofolli, "Cellular automata as an alternative to (rather than an approximation of) differential equations in modelling physics," *Physica*, vol. 10d, pp. 117–127, 1984.

[35] G. Ch. Sirakoulis, I. Karafyllidis, A. Thanailakis, and V. Mardiris, "A methodology for VLSI implementation of cellular automata algorithms using VHDL," *Adv. Eng. Softw.*, vol. 32, pp. 189–202, 2001.

[36] G. Ch. Sirakoulis, I. Karafyllidis, and A. Thanailakis,"A CAD system for the construction and VLSI implementation of cellular automata algorithms using VHDL," *Microprocess. Microsyst.*, vol. 27, no. 8, pp. 381–396, 2003.

[37] V. A. Mardiris and I. G. Karafyllidis, "Universal cellular automaton cell using quantum cellular automata," *Electron. Lett.*, vol. 45, no. 12, pp. 607–609, Jun. 2009

[38] I. Karafyllidis, "Design of a dedicated parallel processor for the prediction of forest fire spreading using cellular automata and genetic algorithms," *Eng. Appl. Artif. Intell.*, vol. 17, no. 1, pp. 19–36, 2004.

[39] G. C. Sirakoulis, I. Karafyllidis, and A. Thanailakis, "A cellular automaton methodology for the simulation of integrated circuit fabrication processes," *Future Gener. Comput. Syst.*, vol. 18, no. 5, pp. 639–657, 2002.

[40] N. Margolus, T. Toffoli, and G. Vichniac, "Cellular automata supercomputers for fluid dynamics modeling," *Phys. Rev. Lett.*, vol. 56, no. 16, pp. 1694–1696, 1986.

[41] M. Arbib, "Simple self-reproducing universal automata," *Inf. Control*, vol. 9, pp. 177–189, 1966.

[42] C. Mizas, G. C. Sirakoulis, V. Mardiris, I. Karafyllidis, N. Glykos, and R. Sandaltzopoulos, "Reconstruction of DNA sequences using genetic algorithms and cellular automata: Towards mutation prediction?" *BioSystems*, vol. 92, no. 1, pp. 61–68, 2008.

[43] K. Preston, M. J. Duff, S. Levialdi, P. E. Norgren, and J. I. Toriwaki, "Basics of cellular logic with some applications in medical image processing," *Proc. IEEE*, vol. 67, no. 5, pp. 826–856, May 1979.

[44] L. Nalpantidis, A. Amanatiadis, G. Ch. Sirakoulis, and A. Gasteratos, "An efficient hierarchical matching algorithm for processing uncalibrated stereo vision images and its hardware architecture," *IET Image Process.*, vol. 5, pp. 481–492, 2011.

[45] A. R. Smith(III), "Real-time language recognition by one-dimensional cellular automata," *J. Comput. Syst. Sci.*, vol. 6, pp. 233–253, 1972.

[46] O. H. Ibarra, M. A. Palis, and S. M. Kim, "Fast parallel language recognition by cellular automata," *Theor. Comput. Sci.*, vol. 41, pp. 231–246, 1985.

[47] W. D. Hillis, "The connection machine: A computer architecture based on cellular automata," *Physica D*, vol. 10, pp. 213–238, 1984.

[48] S. Willson, "Computing fractal dimensions for additive cellular automata," *Physica D*, vol. 24, pp. 190–206, 1987.

[49] S. Takahashi, "Cellular automata and multi fractals: Dimensions spectra of linear cellular automata," *Physica D*, vol. 45, pp. 36–48, 1990.

[50] S. A. Chatzichristofis, D. A. Mitzias, G. Ch. Sirakoulis, and Y. S. Boutalis, "A novel cellular automata based technique for visual multimedia content encryption," *Opt. Commun.*, vol. 283, no. 21, pp. 4250–4260, Nov. 2010.

[51] V. Mardiris, G. C. Sirakoulis, C. Mizas, I. Karafyllidis, and A. Thanailakis, "A CAD system for modeling and simulation of computer networks using cellular automata," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 38, no. 2, pp. 253–264, Mar. 2008.

[52] B. Chopard and M. Droz, *Cellular Automata Modelling of Physical Systems*. Cambridge, U.K.: Cambridge Univ. Press, 1998.

[53] A. Adamatzky, *Identification of Cellular Automata*. New York, NY, USA: Taylor & Francis, 1994.

[54] I. Amlani, A. O. Orlov, G. H. Bernstein, C. S. Lent, and G. L. Snider, "Realization of a functional cell for quantum-dot cellular automata," *Science*, vol. 227, no. 5328, pp. 928–930, 1997.

[55] S. Bhanja and S. Sarkar, "Thermal switching error versus delay tradeoffs in clocked QCA circuits," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 16, no. 5, pp. 528–541, May 2008.

[56] K. Kim, K. Wu, and R. Karri, "Towards designing robust QCA architectures in the presence of sneak noise paths," in *Proc. Des., Autom. Test Eur.*, Munich, Germany, Mar. 7–11, 2005, pp. 1214–1219.

[57] K. Walus, T. J. Dysart, G. A. Jullien, and A. R. Budiman, "QCADesigner: A rapid design and simulation tool for quantum-dot cellular automata," *IEEE Trans. Nanotechnol.*, vol. 3, no. 1, pp. 26–31, Mar. 2004.

[58] I. Karafyllidis, I. Andreadis, P. Tsalides, and A. Thanailakis, "Non-linear hybrid cellular automata as pseudorandom pattern generators for VLSI systems," *VLSI Des.*, vol. 7, pp. 177–189, 1998.

[59] R. J. Chen and J. L. Lai, "VLSI implementation of the universal 2-D CAT/ICAT system," in *Proc. 11th IEEE Int. Conf. Electron., Circuits, Syst.*, Dec. 13–15, 2004, pp. 187–190.

[60] R. J. Chen, Y. T. Lai, and J. L. Lai, "Architecture design of the re-configurable 2-D von Neumann cellular automata for image encryption application," in *Proc. Int. Symp. Circuits Syst.*, Kobe, Japan, pp. 3059–3062, 2005.

[61] R. J. Chen, Y. T. Lai, and J. L. Lai, "Architecture design and VLSI hardware implementation of image encryption/decryption system using re-configurable 2D Von Neumann cellular automata," presented at the Int. Symp. Circuits Syst., Island of Kos, Greece, 2006.

[62] J. Timler and C. S. Lent, "Power gain and dissipation in quantum-dot cellular automata," *J. Appl. Phys.*, vol. 91, no. 2, pp. 823–831, 2002.

[63] M. T. Niemier and P. M. Kogge, "Problems in designing with QCAs: Layout = Timing," *Int. J. Circuit Theory Appl.*, vol. 29, no. 1, pp. 49–62, 2001.

[64] K. Kim, K. Wu, and R. Karri, "Towards designing robust QCA architectures in the presence of sneak noise paths," in *Proc. Des., Autom. Test Eur.*, vol. 2, Mar. 2005, pp. 1214–1219.

[65] S. Wolfram, *A New Kind of Science*. Champaign, IL, USA: Wolfram Media, 2002.

[66] A. Adamatzky, Ed.,*Collision Based Computing*. New York, NY, USA: Springer-Verlag, pp. 397–514, 2001.

[67] S. Murtaza, A. G. Hoekstra, and P. M. A. Sloot, "Cellular automata simulations on a FPGA cluster," *Int. J. High Performance Comput. Appl.*, vol. 25, no. 2, pp. 193–204, May 2011.

[68] I. G. Georgoudas, P. Kyriakos, G. C. Sirakoulis, and I. Andreadis, "An FPGA implemented cellular automaton crowd evacuation model inspired by the electrostatic-induced potential fields," *Microprocess. Microsyst.*, vol. 34, pp. 285–300, 2010.

[69] P. Progias and G. C. Sirakoulis, "An FPGA processor for modelling wildfire spreading," *Math. Comput. Modell.*, vol. 57 pp. 1436–1452, 2013.

[70] R. Zhang, P. Gupta, L. Zhong, and N. K. Jha, "Threshold network synthesis and optimization and its application to nanotechnologies," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 24, no. 1, pp. 107–118, Jan. 2005.

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

[72] H. Cho and E. Swartzlander, "Serial parallel multiplier design in quantum-dot cellular automata," in *Proc. 18th IEEE Symp. Comput. Arithmetic*, 2007, pp. 7–15.

[73] (2014). DATICAQ. [Online]. Available: http://infoman.teikav.edu.gr/daticaq

[74] R. A. Wolkow, L. Livadaru, J. Pitters, M. Taucer, P. Piva, M. Salomons, M. Cloutier, and B. V. C Martins, "Silicon atomic quantum dots enable beyond-CMOS electronics," in *Field-Coupled Nanocomputing Paradigms, Progress, and Perspectives*, Neal G. Anderson and S. Bhanja, Eds. New York, NY, USA: Springer, 2014.

[75] 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.
[76] A. Pulimeno, M. Graziano, A. Sanginario, V. Cauda, D. Demarchi, and G. Piccinini, "Bis-ferrocene molecular QCA wire: Ab initio simulations of fabrication driven fault tolerance," *IEEE Trans. Nanotechnol.*, vol. 12, no. 4, pp. 498–507, Jul. 2013.
[77] A. Orlov, A. Imre, G. Csaba, L. Ji, W. Porod, and G. H. Bernstein, "Magnetic quantum-dot cellular automata: Recent developments and prospects," *J. Nanoelectron. Optoelectron.*, vol. 3, pp. 1–14, 2008.
[78] M. Crocker, M. Niemier, and X. S. Hu, "A reconfigurable PLA architecture for nanomagnet logic," *ACM J. Emerg. Technol. Comput. Syst.*, vol. 8, no. 1, p. 1, Feb. 2012.
[79] F. Karim, K. Walus, and A. Ivanov, "Analysis of field-driven clocking for molecular quantum-dot cellular automata," *J. Comput. Electron.*, vol. 9, pp. 16–30, 2010.
[80] K. Hennessy and C. S. Lent, "Clocking of molecular quantum-dot cellular automata," *J. Vac. Sci. Technol. B*, vol. 19, no. 5, pp. 1752–1755, 2001.
[81] C. S. Lent and B. Isaksen, "Clocked Molecular Quantum-dot Cellular Automata," *IEEE Trans. Electron Devices*, vol. 50, no. 9, pp. 1890–1896, Sep. 2003.
[82] M. Graziano, M. Vacca, A. Chiolerio, and M. Zamboni, "A NCL-HDL snake-clock based magnetic QCA architecture," *IEEE Trans. Nanotechnol.*, vol. 10, no. 5, pp. 1141–1149, Sep. 2011.
[83] P. P. Chaudhuri, D. R. Chowdhury, S. Nandi, and S. Chattopadhyay, *Additive Cellular Automata: Theory and Applications*, vol. 1, New York, NY, USA: Wiley-IEEE Comput. Soc. Press, 1997.

Vassilios A. Mardiris received the Dipl Eng, MSc, and PhD degrees in electrical and computer engineering from the Democritus University of Thrace (DUTh), Greece, in 1995, 2005, and 2012, respectively. He joined Technological Educational Institute of Kavala in 2006 as a faculty member where he is currently an assistant professor at the Department of Business Administration, Technological Institute of Kavala, Greece. His current research emphasis is on modeling and simulation of nanoelectronics, quantum cellular automata, parallel architectures and computer arithmetic, bioinformatics and computer networks. In 2000, he received an award from Intel and IBM for the Low Power Design project LPGD ESPRIT #IV 25256. He is a member of the IEEE.

Georgios Ch. Sirakoulis (M'95) received the Dipl Eng and PhD degrees in electrical and computer engineering from the Democritus University of Thrace (DUTh), Greece, in 1996 and 2001, respectively. He is an associate professor at the Department of Electrical and Computer Engineering, DUTh. He has published more than 160 technical papers, he is co-editor of four books, co-author of eight book chapters and guest editor in eight special issues. He is an associate editor of *Microelectronics Journal*, *Journal of Applied Mathematics*, Recent Patents on Electrical Engineering, and Conference Papers in Computer Science. His current research emphasis is on electronic systems, cellular automata, memristors, quantum cellular automata, and unconventional computing. He is a member of the IEEE.

Ioannis G. Karafyllidis received the Dipl. Eng. and Ph.D. degrees in electrical engineering from the Aristotle University of Thessaloniki, Greece. In 1992 he joined the Department of Electrical and Computer Engineering, Democritus University of Thrace, Greece, as a faculty member, where he is currently a professor. His current research emphasis is on quantum computing, modeling and simulation of nanoelectronic devices and circuits, and biological networks modeling. He is a fellow of the Institute of Nanotechnology, a founding member of the American Academy of Nanomedicine and a member of the Technical Chamber of Greece (TEE).