A Line-Based Parallel Memory for QCA Implementation

Vamsi Vankamamidi, Marco Ottavi, *Member, IEEE*, and Fabrizio Lombardi, *Senior Member, IEEE*

***Abstract*—Quantum-dot cellular automata (QCA) has been widely advocated as a new device architecture for nanotechnology. Using QCA, the innovative design of digital systems can be achieved by exploiting the so-called capability of processing-in-wire, i.e., signal manipulation proceeds at the same time as propagation. QCA systems require low power together with the potential for high density and regularity. These features make QCA an attractive technology for manufacturing memories in which the in-wire paradigm can be exploited for storage purposes. This paper proposes a novel parallel memory architecture for QCA implementation. This architecture is based on storing information on a QCA line by changing the direction of signal flow among three clocking zones. Timing of these zones requires two additional clocks to implement a four-step process for reading/writing data to the memory. Its operation has been verified by simulation. It is shown that the requirements for clocking, number of zones, as well as the underlying CMOS circuitry are significantly reduced compared with previous QCA parallel architectures.**

***Index Terms*—Emerging technologies, memory architecture, quantum-dot cellular automata (QCA), quantum computing.**

I. Introduction

IN THE past few decades, the exponential scaling in feature size and increase in processing power have been successfully achieved by very large scale integration (VLSI) using mainly CMOS technology; however, there is substantial evidence [12] that emerging technologies (mostly based at nanoscale ranges) will be required to supersede the fundamental physical limits of CMOS devices. Among these new technologies, quantum-dot cellular automata (QCA) gives a solution at nanoscale and offers a new method of computation and information transformation. Interconnections for signal transfer are used for logic computation and manipulation by which the so-called processing-in-wire paradigm is accomplished. Micro-sized QCA devices have been fabricated with metal cells which operate at 50 mK [11] (i.e., cryogenic). In terms of feature size, a QCA cell of few a nanometers has been fabricated through a molecular implementation by a self-assembly process. Different devices and circuits have been proposed for QCA implementation. These include a carry look-ahead adder, a barrel shifter, microprocessors, and field-programmable gate arrays (FPGAs) [4], [5], [10], [14], [16].

The work in [11] reported an experimental demonstration of a metal QCA cell; such a device consists of four metal dots connected with tunnel junctions and capacitors. Experiments have confirmed that switching of a single electron in a double-dot cell can control the position of a single electron in another double-dot cell. Basic logic behavior with these cells has been demonstrated in [1] through a majority voter (MV) as a primitive block for QCA design. It is also stated that room-temperature operation requires QCA cells to be fabricated in the range of 1–5 nm. Some possible realizations of molecular QCA have been proposed; progress toward making QCA molecules and establishing the attachment chemistry for a substrate compatible with QCA have been reported.

Clocking is an important feature for QCA. Signal propagation is accomplished along serial timing zones using the one-dimensional (1-D) technique of [8]. This 1-D arrangement results from the four phases required for correctly operating the QCA cells. A trapezoid clocking scheme has been proposed in [9] to provide feedback paths, while generating processing-in-wire capabilities of QCA designs. As shown in [8], QCA has many desirable features for processing; clocking and timing can be adjusted as functions of the cells in a Cartesian layout with low power (power gain has been demonstrated by clocking of the cells), high density, and regularity.

Different circuits and systems can be designed using QCA. A system that is well suited to this technology is the QCA implementation of large memories. However, large-memory designs in QCA present unique characteristics due to their architectural structure (such as the tournament bracket in cell placement). Moreover, sequential circuits and memory elements cannot be directly mapped into QCA using the same criteria of traditional CMOS technology.

Traditionally, two types of memory architectures can be distinguished: parallel and serial architectures. A parallel architecture offers the substantial advantage of low latency because, at each memory cell, only one data bit is stored, so there is no delay in that bit reaching the Read/Write circuitry. In a serial design, multiple bits are stored in each memory cell and share the Read/Write circuitry, thus resulting in a delay proportional to the word size. In CMOS, random access memory (RAM) is usually designed using parallel architectures in which the select/control signal reaches all memory cells (MC) in a row (thus forming a word) during the same clock cycle. This results in an output which is read simultaneously. The one-bit-per-memory cell in QCA reduces latency, but the replication of Read/Write circuitry for each memory bit increases hardware count (QCA-cell, Control-cell, Clocking-zone). Therefore, the parallel architecture provides faster operation of memory at a reduced density.

The objective of this paper is to propose a novel parallel memory architecture which is amenable to QCA implementation. This architecture utilizes an arrangement in the memory

Manuscript received April 11, 2005; revised June 22, 2005.

The authors are with the Electrical and Computer Engineering Department, Northeastern University, Boston MA 0215 USA (e-mail: vvankama@ece.neu.edu; mottavi@ece.neu.edu; lombardi@ece.neu.edu).

Digital Object Identifier 10.1109/TNANO.2005.858589

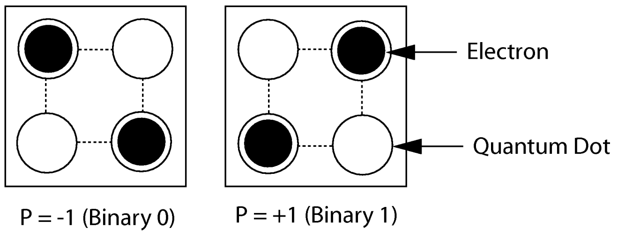

Fig. 1. Binary behavior of a QCA cell.

cell design by which storage is achieved by moving data back and forth along a line of QCA cells. This line-based arrangement results in substantial savings in the number of zones and underlying circuitry's complexity for clocking the QCA memory. However, the proposed architecture requires two additional clocking signals as the line-based operation of the memory cell needs three zones and a four-step process whose timing is different from the one commonly used for adiabatic switching.

This paper is organized as follows. Section II provides a brief review of QCA. Memory architectures for QCA as proposed in the technical literature are described in Section III. The memory design proposed in this paper is presented in Section IV. Section V describes the clocking and timing mechanisms of the proposed parallel memory. Section VI presents the logic evaluation of the proposed memory architecture using a QCA simulator. Discussion and comparison between parallel QCA memories are given in Section VII. Finally, in Section VIII, the conclusions are drawn.

II. Review of QCA

QCA is a new device architecture which is amenable to the nanometer scale [8]. QCA stores logic states not as voltage levels, but based on the position of individual electrons [14]. A quantum cell can be viewed as a set of four charge *containers or dots*, positioned at the corners of a square cell.

Computation is realized by the Coulombic interaction of extra electrons in quantum dots. Each quantum dot is a nanometer-scaled square with wells at each corner of the cell. The two extra electrons which are present in each cell can quantum mechanically tunnel between wells, but they cannot tunnel out of the cell. *Electron repulsion* causes the extra electrons to occupy diagonally opposite wells. These two electron configurations help to *encode* binary information in the cells. Fig. 1 shows a QCA cell and the Boolean nature of the *polarization* for its two electron configurations.

The unique feature of QCA-based designs is that logic states are not stored in voltage levels as in conventional electronics, but they are represented by the *position* of individual electrons. Unlike conventional CMOS circuits in which information is transferred from one place to another by electron transport [14], QCA operates by the Coulombic interaction that connects the state of one cell to the state of its neighbors. This results in a technology in which *information transfer* (interconnection) is the same as *information transformation* (logic manipulation).

QCA cells can be designed to realize a binary wire, an inverter, and an MV. The basic logic gate in QCA is the MV. The MV with logic function $\mathrm{MV}(A, B, C) = AB + AC + BC$ can be realized by only five QCA cells. Logic AND and OR functions can be implemented from an MV by setting one input (the programming input) permanently to 0 and 1, respectively. Let a control cell be defined as the cell controlling the behavior of the MV as AND or OR logic gate. Cells which are positioned adjacent to each other tend to align and produce a binary wire, while cells positioned diagonally from each other align in opposite fashion and produce logic complementation (i.e., an inverter). These fundamental QCA devices provide a complete functional set of logic gates.

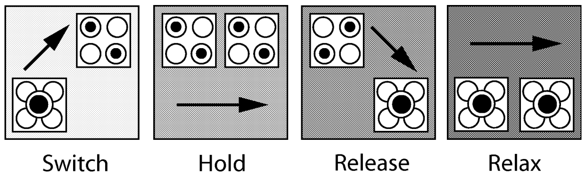

Fig. 2. QCA clock phases.

In traditional electronic systems, timing is controlled through a reference signal (i.e., the clock), however, timing in QCA is accomplished by clocking in four distinct and periodic phases for adiabatic switching [8]. A QCA circuit is partitioned into *serial* (one-dimensional) zones, and each zone is maintained in a phase. For QCA, the clock phases are Switch, Hold, Release, and Relax. Fig. 2 depicts a cell in its *four clock phases*.

Timing zones of a QCA circuit or system are arranged by following the periodic execution of these four clock phases. Zones in the Hold phase are followed by zones in the Switch, Release, and Relax phases. This clocking mechanism provides inherent pipelining and multibit information transfer for QCA. As a zone in the Hold phase is followed by a zone in the Switch phase (and preceded by a zone in the Relax phase), then computation in QCA is strictly 1-D (i.e., unidirectional and consistent with signal propagation). Currently, QCA circuits and systems follow the clocking zone partition scheme of [8]. Designs are partitioned into multiple clocking zones only along one dimension (say the X axis), thus effectively creating *columns* (as *zones*). Clocking and pipelining require designs to maintain sets of four adjacent zones at any time (as according to the four phases, i.e., Switch, Hold, Release, and Relax). For the four phases, clocking to a zone (and the design as a whole) is applied through an underlying CMOS circuitry by a signal, as shown in Fig. 3 [6]. Such a signal generates the required electric field to modulate the tunneling barrier of all cells in the zone (adiabatic switching). To maintain zones in sets of four phases, four CMOS controlled wires (carrying the signal of Fig. 3) are required. The phase of each signal is shifted by $\pi/2$.

III. Review of QCA Memories

The design of memories must first consider clocking and timing as important features for QCA operation. The use of an adiabatic switching technique as commonly employed for QCA circuits requires a four-phased clocking signal which is supplied by CMOS wires buried under the QCA circuitry for modulating the electric field.

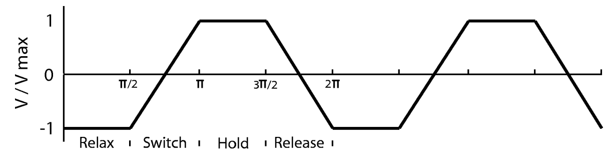

Fig. 3. Four-phased signal for clocking zones in QCA, adiabatic switching.

For quasi-adiabatic operation of a cell, the four phases are Relax, Switch, Hold, and Release. During the Relax phase, there is no interdot barrier and a cell remains unpolarized. During the Switch phase, the interdot barrier is slowly raised and a cell attains a definitive polarity under the influence of its neighbors. In the Hold phase, barriers are high and a cell retains its polarity. Finally, in the Relax phase, barriers are lowered and a cell loses its polarity. As for timing of QCA circuits, they are partitioned into multiple clocking zones, and all cells in a zone are clocked according to this periodic four-phased signal. Therefore, a straightforward approach to implement a memory by QCA is to maintain a cell (zone) in the Hold phase as long as its value must be retained for storage. The main problem with this rather obvious approach is the requirement of an explicit control of the CMOS clock signal from the decoder (which is implemented in QCA). Also, the transfer of signals from QCA to CMOS requires a complicated sensing process using sophisticated electrometers. For a truly QCA-based implementation, memory must be kept in motion, i.e., the memory state has to be continuously moved through a set of QCA cells connected in a loop partitioned into four clocking zones, and, at any given time, one of them is in the Hold phase to retain the information.

The authors of [2] have made an early attempt to design a QCA memory using the so-called SQUARES formalism. The basic principle of this technique is to define a set of equally sized blocks, each performing a basic function in QCA (as either logic or interconnect). These blocks can then be tiled together to design more complex QCA circuits. The obvious advantage of this technique is the ease in the geometric layout; also, this formalism allows a design to be highly modular. However, as the blocks are of standard size (in SQUARES a 5×5 grid is used), a substantial unutilized area appears in each block, thus causing spatial redundancy and lower density in the overall design. The memory designed using SQUARES is a serial architecture. Each memory cell is a closed-loop QCA wire which is partitioned into multiple clocking zones equal to four times the number of bits stored in the loop. This creates a large number of clocking zones even for a modest memory size, thus requiring a considerable amount of CMOS circuitry to generate the clocking signals. Finally, additional control circuitry (such as comparators) must be utilized to make the memory bit-addressable. This results in a quite high hardware penalty per memory cell.

Researchers at Notre Dame University have introduced the H-Memory architecture [5] whose main objectives are high density and uniform access time. The H-Memory has a complete binary tree structure with control circuitry at each node; as the memory spirals are at the leaf nodes, an integration of logic and memory is accomplished in the layout, but the control circuitry and memory are logically separate (similar to CMOS design). However, unlike conventional designs, control and data bits are serialized. The bit stream enters the memory structure at the root node and traverses down the tree by utilizing one control bit for routing at every node in the path. The architectural choice of dealing with serial bit streams results also in rather complex control logic for QCA. The router at each internal node has ten gates and six feedback loops; each loop requires four clocking zones for its implementation. The circuitry at the leaf nodes (i.e., the memory cells) requires 11 gates per node. Also, the memory cell at each leaf node is a spiral allowing storage of several bits, while sharing clocking zones between multiple loops. In this design, the memory size at each spiral and the cell count do not have a linear relationship; each outer loop has an increasing diameter, thus requiring more QCA cells for its implementation (although its storage capacity remains constant).

The authors of [16] have proposed a conventional parallel memory architecture (such as encountered in CMOS-based RAM design) for QCA, i.e., by storing one bit at each memory cell. The single-bit memory cells allow the design of a simple Read/Write circuitry; each memory cell is implemented using 158 QCA cells and the Select signals are separately generated using decoders. The main disadvantage of this approach is the same as that encountered in [2]; namely, data in each memory cell is stored using a closed QCA wire loop (which is partitioned into four clocking zones). Also, clocking zones cannot be shared between memory loops and their dimensions are very small. Therefore, the memory design requires a large number of clocking zones, thus complicating the underlying CMOS circuitry for providing the required clocking signals.

IV. Proposed QCA Memory Design

In this paper, a fundamentally different design of a parallel QCA memory is introduced. This architecture is based on a novel logic arrangement for the MV, namely, the wires to an MV can behave differently (either as input or output) in time depending on the clock phase in which they are operative. This arrangement combined with a new clocking strategy overcomes the limitation of a traditional unidirectional flow of logic signals in QCA. The new arrangement of the MV is exploited in the design of a parallel memory architecture for QCA by which the number of clocking zones for implementing the memory is independent of its size. This is accomplished by sharing zones among all memory cells in a column. A further advantage of this approach is a reduction in the CMOS circuitry to provide the clock signals. The hardware requirements for the Read/Write control logic are comparable to [16]. Also, the Read/Write control logic is very simple compared to other designs in the literature [2], [5]. The only additional cost with respect to [16] is that the design requires two additional clock signals whose Hold/Relax times are different from the original (equally timed four-phased) clock signal. The basic principle of the proposed approach is to store bits by moving them back and forth in QCA lines, hence this technique is referred to as line-based. The design of a one-bit memory architecture is shown in Fig. 4.

The proposed line-based QCA memory cell requires three consecutive clocking zones forming a timing row. At any given time, at least one of the three zones is in Hold phase, such that the memory state is retained. Whenever zone 3 is in the Hold phase, the memory state can be read out based on the values of

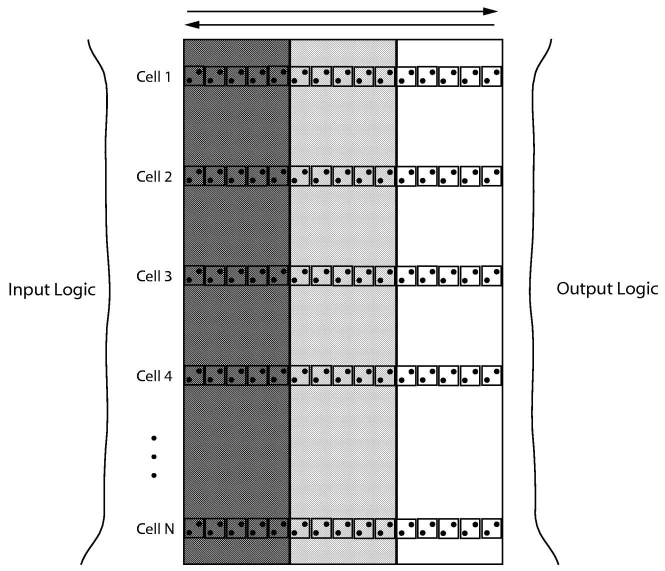

Fig. 4. Line-based memory.

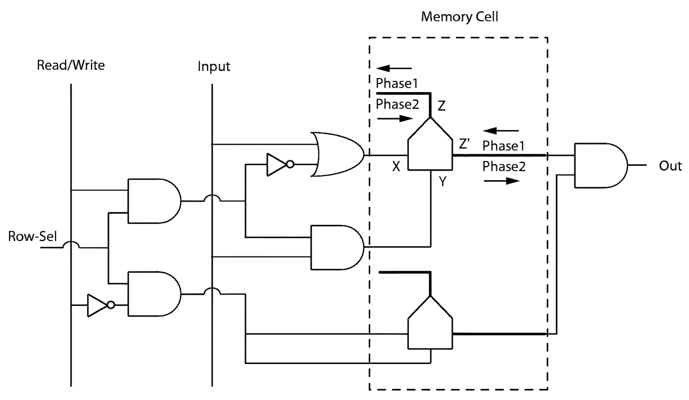

Fig. 5. QCA memory cell with input and output logic circuitry.

the Select signals; whenever zone 1 is in the Switch phase, then a new input state can be written to the memory cell. Multiplexing between the current value and new input value is controlled by the Select signal; this is performed by the MV in zone 2 and the systematic switching of the clocking zones. Such multiplexing by the MV is possible because the wires to the MV behave differently at specific times, i.e., either as input or output depending on the clock phase they are in.

Fig. 5 shows the schematic diagram of a complete QCA line-based memory cell with its Read/Write control logic. Fig. 6 shows the QCA implementation along with the clocking zones. The Read/Write control logic on the input circuitry of the QCA cell consists of four gates. Two of these gates are used to determine the memory operation by ANDing the Read/Write control signals with the Row Select signal. The other two gates are used to duplicate the memory input signal whenever the Write Control signal is high, i.e., the MV in zone 2 is majority dominated and its output is equal to the (new memory) input. When the Write Control signal is low, the outputs of these gates are

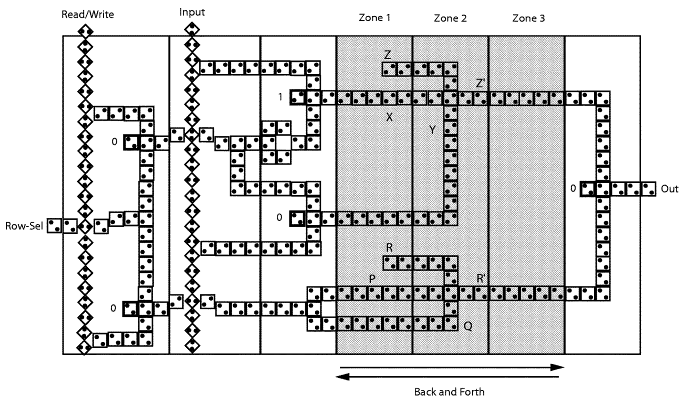

Fig. 6. Multiplexer circuitry to one cell of line-based memory.

different (zero and one, respectively), such that they have no influence on the operation of the MV, i.e., the output of the MV follows the third input which corresponds to the current memory value.

The output circuitry of the memory cell consists of one gate to read the memory state depending on the value of the Control signal. As the Read control signal must be moved to the output circuitry of the memory cell, duplication is required to allow domination of the output of the MV and the transfer of its value. An MV is required for implementing this transfer due to the clocking process which changes the direction of signal propagation. Therefore, the operation of the proposed QCA memory is determined by the following four steps.

Step 1) The inputs and zone 3 of the memory cell are in the Hold phase, while zones 1 and 2 are in the Switch phase. Therefore, the QCA wires (indicated by X, Y and Z′) are inputs to the MV in the Write path; Z is an output. Depending on the Write Control signal, the output Z is either the new memory input or the old (current) memory state. For the MV in the Read path, P, Q, and R′ are inputs and R is an output. The output R is always equal to the inputs P and Q which correspond to the Read Control signal.

Step 2) During this step, the inputs and zone 3 of the memory cell are in the Relax phase. Zone 1 is in the Hold phase and zone 2 is in the Switch phase. As zone 1 is in the Hold phase, zone 2 is in the Switch phase and zone 3 is in the Relax phase, then the previously defined outputs Z and R now become inputs to the MV and the previously defined inputs Z′ and R′ become the new outputs. The input values Z and R are transferred to the outputs Z′ and R′ because the other two inputs of the MVs have no influence or are equal to Z and R.

Step 3) Zones 1–3 are in the Relax, Hold, and Switch phases, respectively. Thus, a new multiplexed memory state and the Read Control signal are transferred to zone 3.

Step 4) During this step, zone 3 is in the Hold phase and the new memory state is read at the Out cell (depending on the value of the Read Control signal).

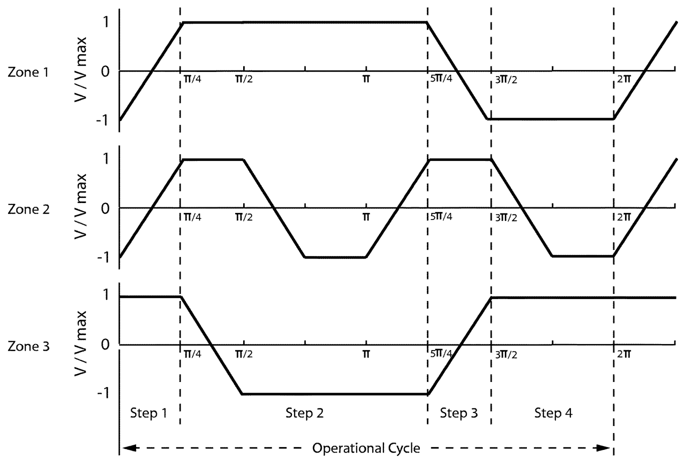

Fig. 7. Clocking signals for the three zones.

V. Clocking Considerations

Metal QCA designs and architectures presented in the technical literature are clocked through a single four-phased clock signal. Designs are partitioned into clocking zones; the clock signal for adjacent QCA zones is phase-shifted by $\pi/2$ such that a concatenation of sets of four adjacent zones is allowed as basic mode of operation for logic propagation in the QCA circuit. However, in the proposed memory architecture, two of the three zones for the memory cell are in the Switch phase at the same time. Similarly, they are in the Hold and Relax phases simultaneously. The period of the Hold and Relax phases is different for the three clocking zones. Therefore, the three zones will have to be clocked by separate signals through the underlying CMOS circuitry.

Fig. 7 shows the periodic signals required to clock the three zones of the memory cell. In Step 1), the clocking signals of zones 1 and 2 are in the Switch phase and zone 3 is in the Hold phase. This allows the new memory input value and the old memory state to be voted to write a new memory state. In Step 2), the new memory state in zone 1 is transferred to zone 2; so, the clock signals for zones 1–3 must be in the Hold, Switch, and Relax phases, respectively. However, zone 2 was in the Switch phase in Step 1) and the Switch phase cannot be followed yet by another Switch phase. Therefore, zone 1 has to be in the Hold phase long enough for the clock signal of zone 2 to transition through the remaining phases and return to the Switch phase, at which time a new memory state is transferred. In Step 3), the memory state is transferred to zone 3; this requires all three zones to be in the Release, Hold, and Switch phases, respectively. In Step 4), zone 3 is in the Hold phase and the memory state can be read out. Zone 3 must be in the Hold phase long enough to allow zones 1 and 2 to cycle through their remaining phases and be in the Switch phase when the memory operation returns to Step 1. Zone 3 is in the Hold phase during Step 1) of the memory operation.

The clocking signals of all three zones of a memory cell are periodic (the frequency of zones 1 and 3 is half of zone 2). All four steps of the clock signal for zone 2 are of the same duration, i.e., the same as the global signal used to clock the remaining parts of the QCA design. Therefore, the proposed memory design requires two additional signals for clocking. The input and output control circuitry is a simple combinational logic with no feedback loop and, therefore, it can be clocked using conventional QCA schemes and its four-equally-phased clock.

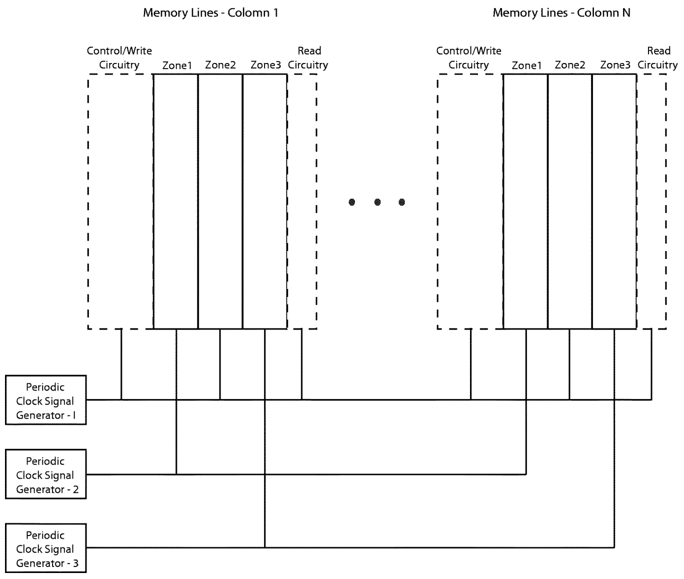
Memory Lines - Colomn 1
Memory Lines - Colomn N
Control/Write Circuitry
Zone1
Zone2
Zone3
Read Circuitry
Periodic Clock Signal Generator - I
Periodic Clock Signal Generator - 2
Periodic Clock Signal Generator - 3

Fig. 8. Underlying CMOS circuitry for clocking the parallel QCA memory.

Fig. 8 shows the CMOS circuitry required to supply the clocking signals to the proposed QCA memory design. Three periodic wave generators are used to obtain the required QCA clock signals. One of the signals is the conventional QCA clock signal (four-phased, equally timed) for the control circuitry (Read, Write), zone 2 of the memory cells as well as the remaining parts of the QCA design. The other two clock signals which have unequal durations (as required to obtain logic propagation), are used for zones 1 and 3 of the memory cells. All columns of the memory cells function identically and the clock signals from the three wave generators are used for all of them. As observed in Fig. 8, the additional complexity for clocking is accounted for generating the two additional signals. Routing of these signals is simplified due to the regularity of the proposed design and its clocking zones.

VI. Simulation

QCADesigner [15] provides a design and simulation environment for QCA circuits. It has multiple simulation engines with CAD capabilities. However, in QCADesigner clocking can only be simulated using the four-phased signal of Fig. 3, thus signal propagation is unidirectional. The same clocking arrangement is also available in AQUINAS [3]. The memory cell design proposed in this paper requires multiple clock signals with different Hold and Relax times and bidirectional signal propagation.

Therefore, some modification were made to simulate the proposed memory cell using QCADesigner for the logic verification of the proposed memory design, A time-to-space (TS) transformation is utilized; this consists of duplicating the circuit (space domain) and operating it along a single direction as equivalent to the memory cell with the two (back-and-forth)

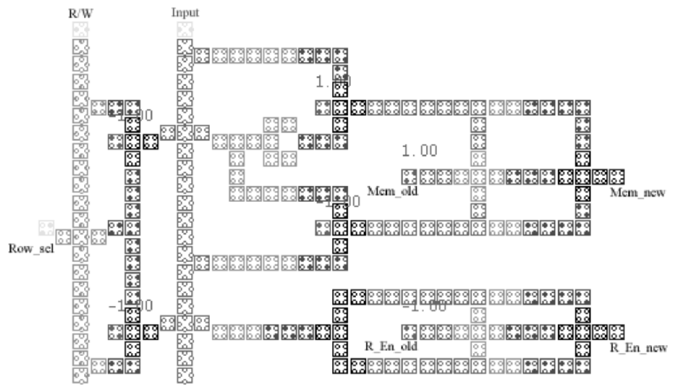

Fig. 9. Simulated circuit of proposed memory cell by ST transformation using QCA designer.

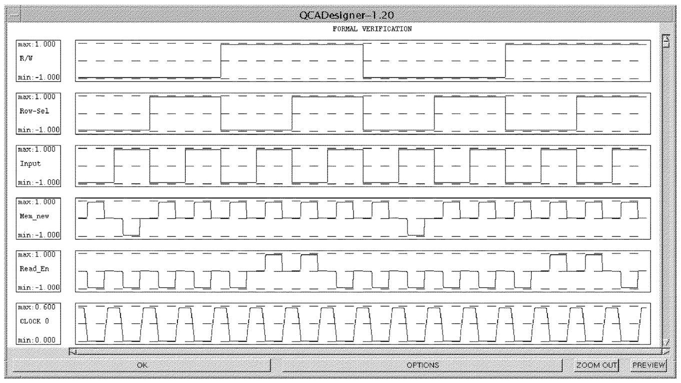

Fig. 10. Simulation waveforms for proposed memory cell.

data movements over one operational cycle (time domain). This transformation can be generalized to the simulation of the memory cell over multiple operational cycles; in this case the cascade arrangement known as the 1-D iterative logic array (ILA) must be utilized. Each of the blocks of the ILA corresponds to the memory cell in a data movement. This approach was originally proposed in [7] for testing sequential circuits.

Fig. 9 shows the design (as equivalent to the proposed memory cell) for simulating one operational cycle. It should be observed that, instead of using a single MV for both (back-and-forth) data movements, two MVs (space domain) are employed and data are moved in only one direction. Simulation has been performed using the bistable engine of QCADesigner with cells dimensions of 18 nm and dot size of 5 nm. Fig. 10 shows the waveforms obtained by simulating this design as equivalent to the operation of a memory cell over one operational cycle. When the R/W signal and the Row-Sel are high, then Mem_New takes the value of Input data (after a delay), whereas at all other times it is equal to Mem_Old (which is fixed at logic 1). Similarly, the Read signal is enabled when the R/W signal is low and the Row-Sel signal is high. Simulation has therefore shown that the logic and timing features of the proposed memory design execute correctly and are compatible with adiabatic switching for QCA implementation.

Finally, the four ports (legs) of the MV in the memory cell design of Fig. 6 have different cell lengths. This occurs because the memory designs presented in this paper and [16] are tailored to QCA implementation and the 1-D clocking scheme causes slightly uneven lengths for the signals to the MVs. The work in [13] has discussed the effects due to different lengths in the cell lines connected to an MV and proposed a solution to the Schrödinger equation using correlation terms for the correct calculation of the output polarization. In the proposed design, within a clocking zone the signal lines to the MVs can be adjusted to have nearly equal length by increasing the size of the memory cell. Moreover, the effects due to uneven lengths could be mitigated by increasing the clock period, therefore providing sufficient time for the QCA cells to attain their true ground state [8]. However as verified by the previously presented simulation results, this is not a problem in the proposed design.

VII. Discussion and Comparison

The architecture presented in the previous section has provided a new QCA design for parallel memory. At cell level, the set of three clocking zones (as required for the line-based implementation of the memory cell) is applicable to all cells that are in the same column. This two-dimensional (2-D) parallel architecture is similar in many respects to the 2-D architecture presented by [16], although the implementation of the memory cell is radically different. Therefore, this parallel memory architecture can be used for comparison purposes to evaluate the proposed memory architecture.

The memory design presented in [16] consists of a closed QCA wire loop; in each memory cell, data is retained by continuously moving it in the wire loop. This mechanism requires the wire loop to be partitioned into four clocking zones, each zone is clocked by a signal with a $\pi/2$ phase difference with the signal of the adjacent zone. As a result, one of the four clocking zones is always in the Hold phase and data is retained in the wire loop. Apart from the clocking zones for the Read/Write logic, each memory cell requires four clocking zones. This directly translates to an increase in the underlying CMOS circuitry to provide the correct clocking signals to the zones. Also, the dimensions of these clocking zones (of width equal to that of a single QCA cell and length equal to that of a few cells) make clocking extremely difficult, if not infeasible. Each memory cell in this design requires a total of seven AND/OR logic gates. To reduce an MV to an AND/OR gate, the control input must be permanently set to either logic 1 or 0. The implementation of these control cells encounters an additional complexity because their polarity must be coerced to a fixed value. Moreover, each memory cell in this design requires seven of such control cells.

The main advantage of the parallel architecture presented in this paper is the sharing of the clocking zones between all memory cells in a column of the 2-D memory design. Therefore, the number of clocking zones for holding data is only dependent on the number of columns (word-size), i.e., it is independent of the number of rows (memory-size). Also, since clocking zones are shared, their dimensions are well suited to underlying clocking circuitry for clocking. As the basic principle of the proposed architecture is to keep memory in motion by moving data back and forth in a QCA line (rather than circulating it in a loop), a modification to the clocking process is required. In this case, the use of a 1-D QCA signal to clock all zones is rather restrictive. To reverse the direction of signal

TABLE I
COMPARISON OF PARALLEL QCA MEMORY ARCHITECTURES

Characteristics	**Loop-Based**	**Line-Based**
No.of QCA Cells per Memory Cell	~ 173	~ 233
No.of QCA Control Cells per Memory Cell	7	5
No.of Zones (Z)	4 per memory loop	3 for all memory lines in a column
CMOS Circuitry	Complex	Moderate
Clocking of Zones	Difficult	Simple

flow as required by the proposed architecture, clock signals with longer Relax/Hold times have to be used; this implies that two more clock signals (in addition to the conventional clock signal) are required. Table I summarizes the characteristics of the two parallel QCA memory architectures which have been compared.

As for density, Fig. 11 shows the projected memory density for DRAM using CMOS technology and parallel memory architectures (the proposed architecture is given by the "Memory Line" curve, while the architecture of [16] is given by the "Memory Loop" curve). DRAM density projections for CMOS are obtained from [12]. When calculating the density of a QCA memory, cell sizes of 1 and 5 nm were assumed for either molecular or metal implementation. The memory-loop architecture of [16] requires an area of $25d \times 40d$ QCA cells per one memory cell, whereas the architecture proposed in this paper takes an area of $25d \times 45d$ QCA cells (where d is the inter-dot distance). Overhead for additional control circuitry (such as memory decoder and routing of signals) is included for both architectures. This area (cost) model includes any unused space within the 2-D layout of a design. For example, the area for the memory cell in Fig. 6 is the product of the number of cells along the X axis, the number of cells along the Y axis, and the dimension of each cell. It is assumed that the underlying CMOS circuitry for clocking does not create any additional overhead when calculating the 2-D area. From Fig. 11, it is evident that parallel QCA memory architectures even with a metal implementation (5 nm) will result in a memory density that CMOS technology will be able to match only after some years. By using a molecular implementation (in the 1-nm range), the large density offered by QCA for memory designs is further evidenced by values well beyond the range of CMOS technology. The loop-based method of [16] offers a density

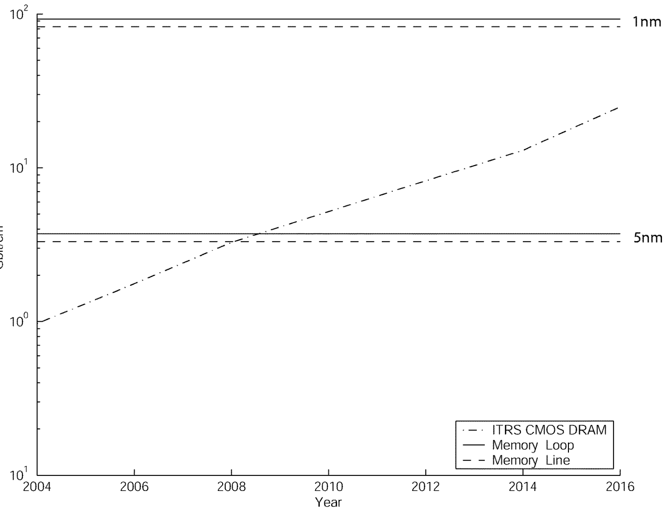

Fig. 11. Density comparison between CMOS DRAM and parallel QCA memory architectures.

that is slightly higher than that of the line-based architecture of this paper.

However, when considering the number of clocking zones (and therefore the operating frequency for switching) the advantage of the proposed line-based architecture is substantial; for the architecture of [16], four clocking zones are required to implement the memory loop for each bit stored in memory. For a line-based memory, only three clocking zones are required for all memory cells in a column of the 2-D memory array (independent of the number of rows). Therefore, in a line-based QCA memory architecture, the number of clocking zones depends only on the word width whereas in a loop-based QCA architecture it depends on both word width and the number of words. Fig. 12 shows the number of clocking zones versus number

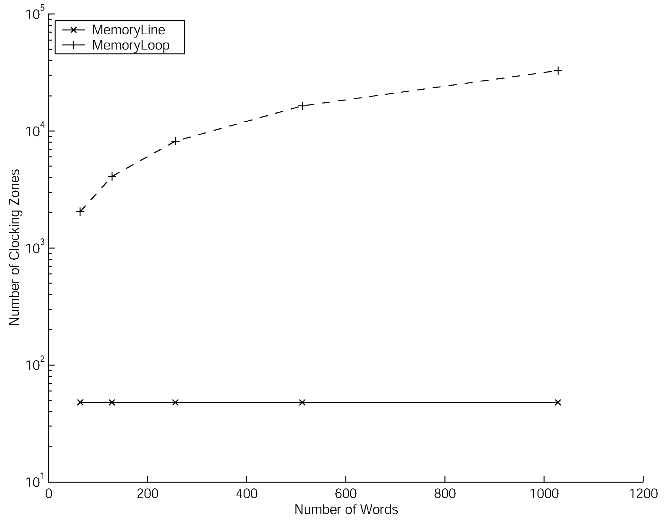

Fig. 12. Comparison in the number of clocking zones for parallel QCA memory architectures.

of words for the two parallel QCA architectures compared in this paper. The difference between these two memory architectures is in orders of magnitude, and it increases with the number of words.

VIII. Conclusion

This paper has presented a novel parallel memory architecture which exploits the so-called capability of processing-in-wire; in this architecture, storage is accomplished by using a QCA line as basic configuration for a memory cell. In the proposed memory cell, data are moved back and forth along a line, thus implementing a bidirectional signal flow which overcomes the traditional arrangement of loop-based QCA memory design. It has been shown that this process requires three clocking zones, thus resulting in adiabatic switching with two additional clocks. These clocks are used to implement a four-step process for reading/writing data to the memory. Using a time-to-space transformation, the operation of the memory cell in this architecture has been verified using QCADesigner as a simulation tool. It is shown that clocking requirements, number of zones, as well as the underlying CMOS circuitry complexity are significantly reduced compared with previous QCA parallel architectures.

References

[1] I. Amlani, A. Orlov, G. Toth, C. Lent, G. Bernstein, and G. L. Snider, "Digital logic gate using quantum-dot cellular automata," *Science*, vol. 284, pp. 289–291.

[2] D. Berzon and T. Fountain, "A memory design in qcas using the squares formalism," in *Proc. 9th Great Lakes Symp. VLSI*, 1999, pp. 168–172.

[3] E. Blair and C. Lent, "An architecture for molecular computing using quantum-dot celluar automata," in *Proc. 3rd IEEE Conf. Nanotechnology*, vol. 1, San Francisco, CA, 2003, pp. 12–14.

[4] A. Vetteth, K. Walus, V. S. Dimitrov, and G. A. Jullien, "Quantum-dot cellular automata carry-look-ahead adder and barrel shifter," presented at the IEEE Emerging Telecommunications Technologies Conf., Dallas, TX, Sep. 2002, 2-I-4.

[5] S. Frost, A. Rodrigues, A. Janiszewski, R. Rausch, and P. Kogge, "Memory in motion: a study of storage structures in qca," in *Proc. 1st Workshop on Non-Silicon Computing*, 2002. [Online]. Available: http://www.cs.cmu-edu/~phoenix/nsc1/.

[6] K. Hennessy and C. Lent, "Clocking of molecular quantum-dot cellular automata," *J. Vac. Sci. Technol. B*, vol. 19, no. 5, pp. 1752–1755, 2001.

[7] F. C. Hennie, "Space-time transformations," in *Finite-State Models For Logical Machines*. New York: Wiley, 1968, pp. 415–445.

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

[9] M. Niemier and P. Kogge, "Problems in designing with qcas: layout=timing," *Int. J. Circuit Theory Applicat.*, vol. 29, no. 1, pp. 49–62, Mar. 2001.

[10] M. Niemier, A. Rodrigues, and P. Kogge, "A potentially implementable fpga for quantum dot cellular automata," in *Proc. 1st Workshop on Non-Silicon Computation*, Boston, MA, 2002. [Online]. Available: http://www.cs.cmu-edu/~phoenix/nsc1/.

[11] A. Orlov, I. Amlani, G. Bernstein, C. Lent, and G. Snider, "Realization of a functional cell for quantum-dot cellular automata," *Science*, vol. 277, pp. 928–931, 1997.

[12] *Technology Roadmap for nanoelectroincs, Eur. Commission, IST Programme, Future and Emerging Technologies*, R. Compañó, L. Molenkamp, and D. J. Paul, Eds., 2000. [Online]. Available: http://www.cordis.lu/esprit/src/melna-rm.htm.

[13] G. Toth and C. S. Lent, "The role of correlation in the operation of quantum-dot cellular automata," *J. Appl. Phys.*, vol. 89, pp. 7943–7953, 2001.

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

[15] Qcadesigner Homepage, K. Walus. (2004). *http://www.qcadesigner.ca/index.html* [Online]

[16] K. Walus, A. Vetteth, G. Jullien, and V. Dimitrov, "Ram design using quantum-dot cellular automata," in *Tech. Proc. Nanotechnology Conf. and Trade Show*, vol. 2, 2003, pp. 160–163.

Vamsi Vankamamidi received the B.S. degree in computer engineering from Bombay University, Bombay, India, in 2000 and the M.S. degree in electrical engineering and computer science from Toledo University, Toledo, OH, in 2001. He is currently working toward the Ph.D. degree in computer engineering at Northeastern University, Boston, MA.

As part of his dissertation, he is working on quantum-dot cellular automata (QCA), a nanoscale device architecture to supersede conventional silicon-based technology. His research interests include novel circuit designs, memory architectures, and nanoscale computing devices.

Marco Ottavi (M'04) received the Laurea degree in electronic engineering from the University of Rome "La Sapienza," Rome, Italy, in 1999 and the Ph.D. degree in microelectronic and telecommunications engineering from the University of Rome "Tor Vergata," Rome, in 2004.

In 2000, he was with ULISSE Consortium, Rome, as a Designer of digital systems for space applications. In 2003, he was a Visiting Research Assistant with the Electrical and Computer Engineering Department, Northeastern University, Boston, MA, and where, since 2004, he has been a Postdoctoral Research Associate. His research interests include yield and reliability modeling, fault-tolerant architectures, and online testing and design of nanoscale circuits and systems.

Fabrizio Lombardi (M'81–SM'02) received the B.Sc. degree (Hons.) in electronic engineering from the University of Essex, Essex, U.K., in 1977, the M.S. degree in microwaves and modern optics and the Diploma in microwave engineering from University College London, London, U.K., both in 1978, and the Ph.D. degree from the University of London, London, U.K., in 1982.

He currently holds the International Test Conference (ITC) Endowed Professorship at Northeastern University, Boston, MA, where, from 1998 to 2004, he served as Chair of the Department of Electrical and Computer Engineering. Prior to Northeastern University, he was a faculty member with Texas Tech University, Lubbock, TX, the University of Colorado-Boulder, and Texas A&M University, College Station. His research interests are testing and design of digital systems, quantum and nano computing, ATE systems, configurable/network computing, defect tolerance, and computer-aided design VLSI. He has extensively published in these areas and edited six books.

Dr. Lombardi has been the recipient of many professional awards, including the Visiting Fellowship at the British Columbia Advanced System Institute, University of Victoria, BC, Canada (1988), twice the Texas Experimental Engineering Station Research Fellowship (1991–1992, 1997–1998), the Halliburton Professorship (1995), the Outstanding Engineering Research Award at Northeastern University (2004), and an International Research Award from the Ministry of Science and Education of Japan (1993–1999). He was the recipient of the 1985/1986 Research Initiation Award from the IEEE/Engineering Foundation and a Silver Quill Award from Motorola-Austin (1996). He was an Associate Editor (1996–2000) for the IEEE TRANSACTIONS ON COMPUTERS and a Distinguished Visitor of the IEEE Computer Society (1990–1993 and 2001–2004). Since 2000, he has been the Associate Editor-In-Chief of the IEEE TRANSACTIONS ON COMPUTERS and an Associate Editor of the *IEEE Design and Test Magazine.* Since 2004, he has served as the Chair of the Committee on "Nanotechnology Devices and Systems" of the Test Technology Technical Council of the IEEE. He has been involved in organizing many international symposia, conferences, and workshops sponsored by professional organizations as well as guest editor of Special Issues in archival journals and magazines such as the IEEE TRANSACTIONS ON COMPUTERS, the IEEE TRANSACTIONS ON INSTRUMENTATION AND MEASUREMENT, the *IEEE Micro Magazine,* and the *IEEE Design & Test Magazine.* He is the Founding General Chair of the IEEE Symposium on Network Computing and Applications.