Timing Verification of QCA Memory Architectures

Marco Ottavi[1], Luca Schiano[1], Salvatore Pontarelli[2], Vamsi Vankamamidi[1], and Fabrizio Lombardi[1]
[1] ECE Department, Northeastern University, Boston, (MA) 02115, USA
[2]Department of Electronic Engineering University of Rome, "Tor Vergata" Rome, 00133, Italy
Email: {mottavi,lschiano,spontare,vvankama,lombardi}@ece.neu.edu

***Abstract*— Quantum-dot Cellular Automata (QCA) provides a new functional paradigm for information processing and communication. In QCA the design of memories is substantially different from CMOS; several memory architectures have been proposed for QCA implementation. They have different logic and timing features in their operation. However, these architectures have not been fully verified due to limitations in current QCA design tools. This paper deals with the timing verification of three different memory architectures using simulation in HDL Verilog. Results are presented to confirm the viability and functional correctness of these memory architectures. This paper also shows that HDL based simulation is very effective for verification while allowing flexibility in modeling.**

I. Introduction

Quantum-dot Cellular Automata (QCA) is one of the most promising technologies to supersede the fundamental physical limits of CMOS at nano scale range [6]. QCA offers a new method of computation and information manipulation in which signal transfer and logic computation are combined in the so-called processing-in-wire paradigm. This paradigm permits to design digital circuits for arithmetic processing and storage by utilizing new logic and timing arrangements that exploit the unique features of QCA. Existing tools based on the computationally intensive solution of quantum equations have been developed to facilitate the design process. In particular, QCADesigner [9] has emerged as one of the most used tool for logic analysis at layout level. However, its complexity in execution and the limited flexibility in capturing novel system-level features of QCA (such as timing) have resulted in the inability to fully verify complex designs. An HDL Verilog library has been developed [4] to verify QCA memory architectures and their different operational features; this paper presents in detail the HDL based simulations that are used for verification of timing in the operations (read and write) of these memory architectures.

II. QCA Memory Architectures

A QCA based implementation of a memory architecture commonly requires the use of the so-called *memory-in-motion approach*, i.e. the stored information must be continuously moved through a set of QCA cells connected in a loop. In the technical literature, QCA based memories have been classified into parallel and serial architectures [3]. [1] has introduced the *serial loop approach* for the memory cell (and associated clocking scheme) shown in Fig. 1 (a); the four colors represent the zones clocked with different phases for quasi-adiabatic switching as commonly employed for metal implementation of QCA. When a zone is in the Hold phase (Zone A in figure), the bit value is retained; therefore for loops with N bits, $4N$ zones are needed in this architecture.

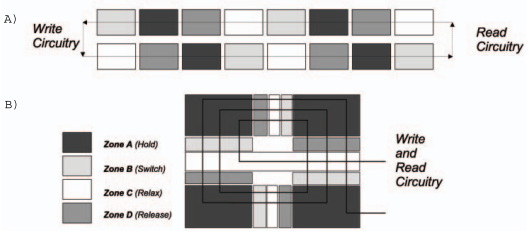


Fig. 1. Serial Memory arrangements (a): loop (b): spiral

[2] has introduced a *H-Memory architecture* with high density and uniform access time. The H-Memory has a complete binary tree structure with control circuitry at each node. The memory spirals (Fig. 1 (b)) are at the leaf nodes and account for a fixed number of clocking zones (four) with respect to memory size, i.e. an increase by four at every loop. Integration of logic and memory is accomplished in the layout, but the control circuitry and memory are still logically separate (similarly to a 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 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 relationship between memory size at each spiral and the cell count is not linear.

As shown in Fig. 2, [10] has proposed a *parallel memory architecture* for QCA, i.e. 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 170 QCA cells. The main disadvantage of this approach is the same as the one encountered in [1] namely, data in each memory cell is stored using a closed QCA wire loop (that is partitioned into four clocking

1-4244-0078-3/06/$20.00 (c) 2006 IEEE

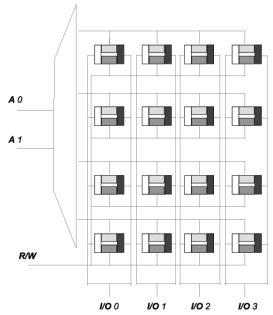
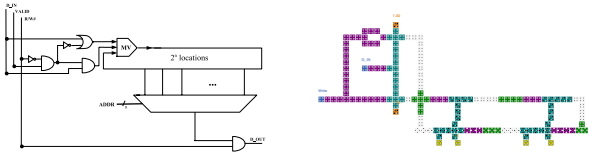


Fig. 2. Parallel Memory Arrangement



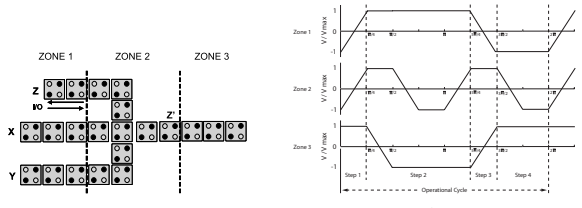
(a) (b)

Fig. 3. Loop implementation of an hybrid memory (a) Schematic Layout (b) QCA Layout (detail).

zones). Also, clocking zones (of very small dimension) cannot be shared between memory loops.

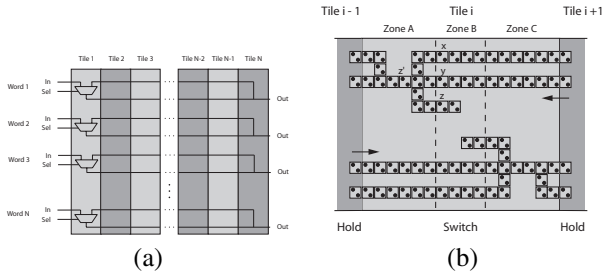
The *hybrid memory architecture* of [3] (shown in Fig. 3) can be considered as an evolution of the serial memory of [1]. It is referred to as "hybrid" because it operates on a serial write and parallel read. This characteristic permits to combine the low latency advantage of a parallel architecture with the low area requirement (and therefore high density) of a serial architecture. As a serial memory incurs in a slow access for both (write and read) operations, then this architecture uses a parallel read approach. In the hybrid memory 2^n bits are stored in a closed loop. An n-bit address selects the bits inside of the loop and a 2^n counter allows to synchronize the stored bits with the control circuitry. The write operation can incur in a delay of at most $(2^n - 1)$ clock cycles. Instead, the read operation of a specific address can be performed with no synchronization using an adder to offset the position of the bits due to the motion of the loop.

Differently from all previous architectures (based on a loop structure), the so-called *line based approach* exploits the Majority Voter (MV) as a memory element [7]. MV



(a) (b)

Fig. 4. a) QCA Line-Based-Memory cell (core layout). b) Clock signals.



(a) (b)

Fig. 5. Array in Line Based Serial Memory (a) Schematic Layout (b) Internal Tile

stores the value of a bit and propagates it as in a shift register. The clock scheme is modified to control the execution of the four phases of the cells (Hold, Switch, Release, Relax) for bidirectional mode of operation. Fig. 4 depicts the core of the QCA memory element and the clocking signals of a one-bit cell for a line based *parallel* memory. To account for the new clocking strategy, when the terminal Z acts as an input, Z' acts as output and vice-versa. If the inputs X and Y differ, the stored value is moved back and forth between Z and Z'; its value can be read when Zone 3 is in the Hold phase. If $X = Y = 1$ ($X = Y = 0$) the MV of Zone 2 forces the value 1 (0) to Z and Z' as for a write operation. The significant advantage of this architecture is that the clocking distribution circuitry is simplified, because the same clocking zones can be shared by different memory cells and all bit lines (columns) share the same clock signal for its column-wise distribution.

The memory cell of a line based *serial* architecture [7] relies on the bidirectional clocking scheme used for the parallel line based memory. This memory consists of two long horizontal wires connected together at both ends by two short vertical wires (Fig. 5), thus creating a loop for the memory-in-motion paradigm. The long horizontal wires are made by tiles with a structure similar to the core of the parallel line based memory. The number of tiles into which the memory loop is partitioned, determines the word size, while the number of stacked memory loops determines the memory size. To store data in the loops, the two horizontal wires must move bits in opposite directions. The line based design of serial memories allows to apply one single clock signal to the wires to propagate the signals from left to right and from right to left (thus symplifying the clocking structure). Clocking zones can be partitioned in columns, as shown in Fig. 5.a).

III. Timing Simulation

To provide a fast yet reliable verification tool for QCA memory architectures (as well as other circuits), a HDL Verilog library was generated [4] for a novel environment. The model of this environment relies on transforming the QCA layout of a circuit into equivalent (combinational)

functional blocks that are further transformed into Verilog modules; the size of each partitioned block also matches the size of a clocking zone in the QCA layout. Clocking is simulated by applying an extra control input (signal $zoneX[2]$, with $X \in (1..4)$) to the Verilog modules (to modify their capability to read and store the inputs). When a module is in the Switch phase ($zoneX = 01$), HDL allows the combinational function to operate on its inputs; while in the Hold phase ($zoneX$=10) it retains the latest attained value of the Switch phase (i.e. latching the inputs). When a module is in the Release and Relax phases ($zoneX$=11 and 00, respectively), the outputs are in high impedance (i.e. Hi-Z). Hereafter, results of the timing simulation of different memory architectures (as briefly reviewed in a previous section) are reported next for verification. As representative of different timing features, only the memory cells of the following memory architectures are cosidered: (1) the serial memory architecture [1], (2) the hybrid memory architecture [3], [5], (3) the line-based memory [8], [5]. For each of these architectures, the waveforms obtained by the Verilog simulation with the ModelSim [11] simulator are provided.

A. Loop-based Memory Cell

Fig. 1(a) shows the QCA implementation of a memory cell based on the "memory-in-motion loop" structure [1], [5]. Only the core of the memory cell is considered in the presented timing simulation, i.e. the addressing circuitry is not considered.

Clocking of the zones is derived from the same reference clock (a 50% duty cycle), with a phase displacement of $\frac{\pi}{2}$. Therefore, when Zone 1 is in Hold, Zone 2 is in Switch, Zone 3 is in Release and Zone 4 is in Relax. Fig. 6 shows the waveforms obtained by simulating the Verilog model with ModelSim [11]. In particular, after having initialized to zero the bit stored in the loop, the following repetitive sequence of operations takes place: (1) write a 1 (this event is referred to as "A" in Fig. 6); (2) store the previous value, allowing to perform a possible read in the same cycles ("B" in Fig. 6); (3) write a 0 (referred to as "D" in Fig. 6); (4) store the previous value

The information bit (d_in in Fig. 6) is latched when Zone 1 is in the Switch phase, and it enters the loop when Zone 4 is in the Switch phase. These four clock cycles have been taken into account (as an example) for the signal at the input interface circuitry. Furthermore, the output of the loop (out) has been arbitrarily assigned to a QCA device in the second clocking zone, so that two additional clock cycles must be added to the write latency. Therefore, the bit value 1 is written to out 6 clock cycles after the "write a 1" instruction is given to the serial memory (see "C" in Fig. 6). Similarly, 6 clock cycles after the "write a 0" instruction is given to the memory circuit bit value 0 is written to out (see "D" and "'E"' in Fig. 6, respectively).

B. Hybrid Memory Cell

Fig. 3 shows the QCA implementation of a hybrid memory (serial write/parallel read) [3]. As an example, four words ($N = 4$) are considered in the memory; therefore, four bits must be stored in the loop (and made accessible during a read operation).

Fig. 7 shows the waveforms obtained by ModelSim [11] for the Verilog model of the hybrid memory cell. The four bits stored in the loop are initialized to 0 and then serially written to a bit value of 1. The signal $bits\_stored$ consists of the concatenation of bits $out[0-3]$; the four bits belong to four different words and in this case for ease of presentation, they are read together. As shown in the waveforms, there is a latency prior to writing a bit in the memory (after the write signal is issued). In this simplified model (the addressing circuitry is not fully described as in [3]), the input data (d_in) requires three full clock cycles for propagation to the first memory cell bit, i.e. this corresponds to the distance in cycles between a Zone 1 in a Switch phase and the following Zone 4 in a Switch phase. To write the nth bit of the memory cell, $4(n-1)$ clock cycles must be added to this initial offset. As an example in Fig. 7, $3+3\times 4$=15 clock cycles are required to write the 4th bit of the memory (when $bits\_stored$ reaches the value "1111").

C. Line-Based Memory Cell

Fig. 4(a) shows the core cell (or tile) for realizing the storage element in a Line-Based memory. The clock signals for the required switching mechanism are shown in Fig. 4(b).

The resulting waveforms from ModelSim [11] are shown in Fig. 8. Signals $zone[1-3]$ refer to the corresponding clocking zones, while X and Y are the inputs, Z is the node in which the bit is stored after the MV, and out is the cell output. As shown in Fig. 8, the operations performed on a memory cell are as follows: (1) write a 1 as bit value (event referred in Fig. 8 as "A"); (2) retain the stored value ("B" in Fig. 8); (3) write a "0" ("C" in Fig. 8) ; (4) retain the stored value.

The output (out) is latched (and therefore ready for a possible read operation) when the signal Zone 3 goes in the Switch phase. Since the clocks connected to the different zones have different duty cycles, the clock signal of "zone 2" is considered as a reference (it was selected as it is the only signal with a 50% duty cycle). Therefore, the output is written with a bit of value 1 ("D" in Fig. 8) 6 clock cycles after the "write a 1" operation ("A"). Similarly, the memory is reset to a bit of value 0 ("E" in Fig. 8) after 6 clock cycles of latency (from event "C" in Fig. 8).

IV. Conclusion

In this paper, the timing features of different QCA memory architectures has been verified by utilizing an

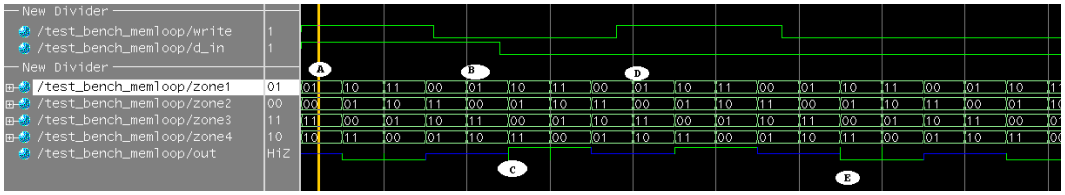


Fig. 6. Waveforms for the Loop-Based Memory model

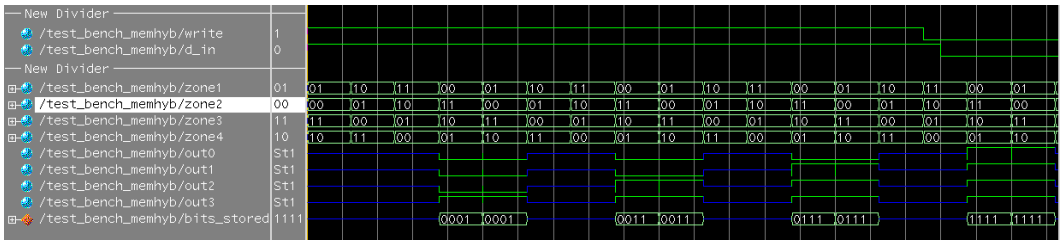


Fig. 7. Waveforms for the hybrid-memory model

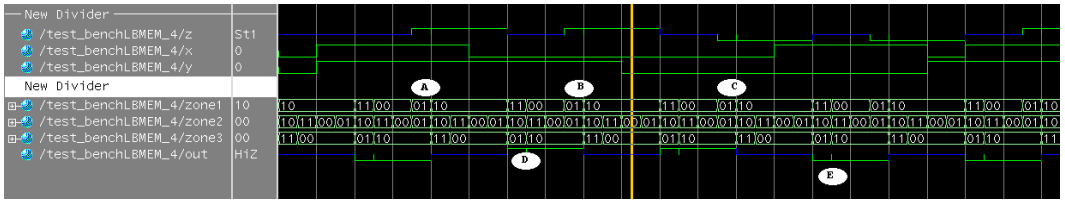


Fig. 8. Waveforms for the Line-Based memory model

HDL Verilog QCA library [4] that has been developed as part of a larger environment for Computer-Aided Design (CAD) of QCA circuits. HDL simulation permits timing verification of these memory architectures at reduced computational complexity. Flexibility for defining design features (such as organization of the clocking zones) that can not be changed using other simulation engines is also accomplished within simple models for the operation of the memories. This is particularly useful in the simulation of innovative QCA circuits (such as line based memories). Other simulation approaches [9] are rather inflexible when defining the clock scheme; in HDL, the clock signal can be easily modified within a very structured simulation environment.

REFERENCES

[1] D. Berzon and T. Fountain. A memory design in qcas using the squares formalism. In *Proceedings Ninth Great Lakes Symposium on VLSI*, pages 168–172, 1999.

[2] S. Frost, A. Rodrigues, A. Janiszewski, R. Rausch, and P. Kogge. Memory in motion: A study of storage structures in qca. In *First Workshop on Non-Silicon Computing*, 2002.

[3] M. Ottavi, S. Pontarelli, V. Vankamamidi, and F. Lombardi. Design of a qca memory with parallel read/serial write. In *Proceedings of 2005 IEEE Computer Society Annual Symposium on VLSI*.

[4] M. Ottavi, L. Schiano, D. Tougaw, and F. Lombardi. Hdlq: A hdl environment for qca design. In *Internal report, available on request*, 2005.

[5] M. Ottavi, V. Vankamamidi, S. Pontarelli, and F. Lombardi. Novel approaches to qca memory design. In *Proceedings of the 5th IEEE conference on Nanotechnology IEEE-NANO 2005*, pages 699–702, Nagoya (japan), Jul. 2005.

[6] R.Compano, L.Molenkamp, and D.J.Paul. Technology roadmap for nanoelectroincs. In *European Commission IST programme, Future and Emerging Technologies*.

[7] V. Vankamamidi, M. Ottavi, and F. Lombardi. Tile based design of a serial memory in qca. In *Proceedings of ACM Great Lakes Symposium on VLSI Chicago, Illinois, Apr. 2005*, pages 201–206.

[8] V. Vankamamidi, M. Ottavi, and F. Lombardi. A line-based parallel memory for qca implementation. In *IEEE Transactions on Nanotechnology*, volume 4, pages 690–698, November 2005.

[9] K. Walus. Qcadesigner homepage. In *http://www.qcadesigner.ca/index.html*, ATIPS Laboratory, University of Calgary, Calgary., 2004.

[10] K. Walus, A. Vetteth, G. Jullien, and V. Dimitrov. Ram design using quantum-dot cellular automata. In *Tech. Proc. 2003 Nanotechnology Conference*.

[11] Web-site. http:www.model.com.