QCA memory with parallel read/serial write: design and analysis

M. Ottavi, S. Pontarelli, V. Vankamamidi, A. Salsano and F. Lombardi

Abstract: The authors present a novel memory architecture for quantum-dot cellular automata (QCA). The proposed architecture combines the advantage of reduced area of a serial memory with the reduced latency in the read operation of a parallel memory, hence its name is 'hybrid'. This hybrid architecture utilises a novel arrangement in the addressing circuitry of the QCA loop for implementing the memory-in-motion paradigm. An evaluation of latency and area is pursued. It is shown that while for a serial memory, latency is $O(N)$ (where N is the number of bits in a loop), a constant latency is accomplished in the hybrid memory. For area analysis, a novel characterisation that considers cells in the logic circuitry, interconnect in addition to the unused portion of the Cartesian plane (i.e. the whole geometric area encompassing the QCA layout), is proposed. This is referred to as the effective area. Different layouts of the hybrid memory are proposed to substantiate the reduction in effective area.

1 Introduction

Quantum-dot cellular automata (QCA) [1, 2] have been proposed for implementing high performance digital circuits with low power consumption, very high density and fast operational speed [3]. QCA represent an innovative technology at the nanometre scale to supersede today's VLSI CMOS-based devices that are foreseen to approach their physical limits in the next decade [4].

QCA implementations of basic logic gates as well as circuits (such as adders and FPGAs) have been proposed in the technical literature [5–11]. A unique feature of QCA-based circuits is the presence of clocking zones that allow for the semi-adiabatic switching of the cells [2, 12]. Hence, propagation of signals in a QCA circuit is not instantaneous, but it is delayed by a number of cycles equal to the number of clocking zones which the signal must traverse [13]. Signal propagation leads to pipelining as the obvious processing mode in QCA; however, this unique feature makes the implementation of a memory element particularly challenging. While still in infancy for planning the physical implementation of large memories, a preliminary study of QCA is useful to fully assess and exploit the unique characteristics of this technology. A QCA memory cell can be designed based on the memory-in-motion paradigm [8] in which the value of the stored data is moved through different cells arranged in a loop. This allows to store a value with no need of directly interfacing CMOS and QCA for generating the required clock signals (as dependent on the operation performed by the QCA circuit). The memory cells can be arranged in an array as a memory bank. Three architectures of QCA memory arrays have been proposed in the technical literature:

- The parallel architecture [11]: the memory is arranged similarly to a CMOS-based RAM (random access memory) array and each single memory element consists of a one bit loop.
- The serial architecture [14]: serially accessible words are stored into a loop that is functionally equivalent to a shift register.
- The H tree architecture [8]: the memory consists of small spirals, each containing a word, and arranged in a recursive tree structure.

This paper provides a detailed analysis of different figures of merit for a novel QCA memory architecture [15]. This architecture utilises a parallel read operation on multiple bit loops. The proposed memory arrangement is referred to as 'hybrid' owing to the different operational mechanisms for the two operations (read and write). This results in advantages over previous architectures: compared with the H tree and serial memories, the parallel read mechanism allows a reduced latency (an almost immediate access is accomplished); compared with a parallel arrangement, the proposed architecture incurs into a reduction in area, thus increasing the density. A novel figure of merit referred to as effective area, is introduced in the analysis. The effective area is the geometric area of the memory layout that includes the area of the QCA cells for the logic and interconnect circuits as well as the unused portion of the Cartesian plane (as required for avoiding avoid unwanted Coulombic interactions among cells).

The paper is organised as follows. Section 2 provides an overview of QCA, inclusive of its functional paradigms and basic logic blocks. In Section 3 QCA memory architectures, as proposed in the technical literature, are briefly reviewed. In Section 4, the architecture of the proposed memory cell and its control logic are described. In Section 5, the proposed memory architecture and QCA memory architectures previously presented in the technical literature are evaluated and compared with respect to latency and area. Section 6 outlines the conclusion.

IEE Proceedings online no. 20050094

doi:10.1049/ip-cds:20050094

Paper first received 4th April and in revised form 1st December 2005

M. Ottavi, V. Vankamamidi and F. Lombardi are with the Department of Electrical and Computer Engineering, Northeastern University, Boston, MA 02115, USA

S. Pontarelli and A. Salsano are with the Department of Electrical Engineering, University of Tor Vergata, Rome, Italy

E-mail: mottavi@ece.neu.edu

2 Review of QCA design

A detailed presentation of QCA design and architectures can be found in [2] and [16]. In QCA, the basic functional unit is the so-called cell; a QCA cell consists of a set of quantum dots with two extra electrons. A quantum dot is a region of the cell in which charge can localise; electrons can tunnel between dots. Figure 1 shows a schematic representation of a 4-dot QCA cell; the Coulombic interaction between the two mobile electrons tends to localise the particles in a diagonal pattern. Therefore, the 4-dot QCA cell can be in two stable polarisation states. These states can be used to encode binary information. The nature of the particles allows switching by quantum tunnelling the electrons from the two stable polarisation states. Two QCA cells interact by Coulombic forces; repulsion of charges of equal polarity forces electrons to be in dots of opposite position in a cell (i.e. along each diagonal). Once the polarisation of a QCA cell is fixed, the encoded binary information is transferred to adjacent cells as shown in Fig. 2 for the so-called QCA wire.

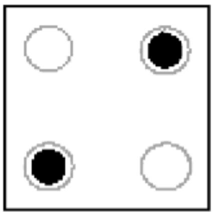

Fig. 1 *4-dot QCA cell*

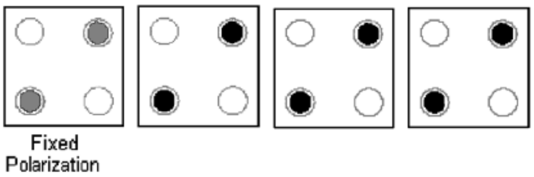

Fig. 2 *QCA wire*

The Coulombic interaction between adjacent cells allows also to implement logic circuits by simply changing the cell placement in the layout. In particular, the inverter can be realised as shown in Fig. 3. The binary information stored in cell 1 is transferred to cells 2 to 6. The electron pair in cell 7 interacts with its neighbouring cells 5 and 6 to reduce the Coulombic interaction and switching to the state with opposite polarisation (in this case, from $+1$ for logic 1 to -1 for logic 0). Another basic logic circuit that can be designed by QCA, is the *majority voter gate* (shown in Fig. 4).

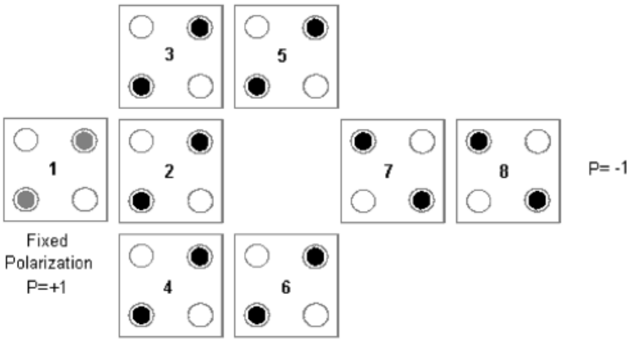

Fig. 3 *QCA inverter*

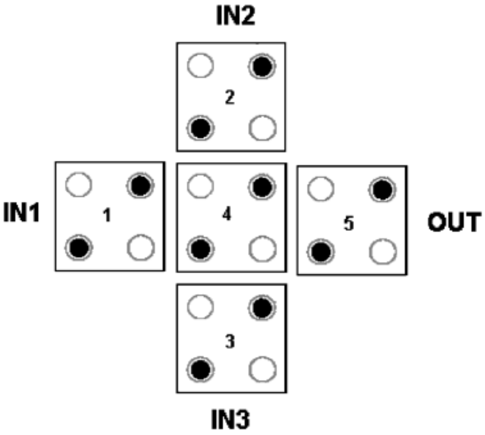

Fig. 4 *QCA majority voter*

This is effectively a logic gate with three inputs (cells $1, 2, 3$) and one output (cell 5). Cell 4 is the centre cell as computing element. The polarisation of this cell depends on the polarisation of the input cells and corresponds to the polarisation of the majority of the inputs. The information computed by cell 4 is transmitted to cell 5. The truth table for the majority gate is given in Table 1. The majority voter can be used to implement a two-input AND gate by fixing the polarisation of an input to logic value 0, and a two-input OR gate by fixing the polarisation of an input to logic value 1. Hence, the majority voter together with the inverter provide an universal logic set, because any combinational function can be implemented using these gates.

Table 1: Truth table of the QCA majority voter

Cell 1 (IN1)	Cell 2 (IN2)	Cell 3 (IN3)	Cell 5 (OUT)
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

Another unique feature of QCA cells is the ability to implement a coplanar wire-crossing, i.e. crossing of two wires (horizontal and vertical) on the same plane that is not possible in CMOS technology (Fig. 5). The vertical wire consists of cells rotated by 45 degrees, while the horizontal wire has the same configuration as in Fig. 2. The binary information on the vertical wire is inverted in every cell; it has been demonstrated that the signals along the vertical and horizontal wires do not interact.

A further feature of QCA is the clocking process; clocking of QCA circuits requires a completely different approach than CMOS. A clock provides both synchronisation and power gain to the QCA circuit [2]. The QCA clock is implemented by applying an E field that controls the potential barriers between quantum dots. The change in potential barriers allows to control the rate at which the electrons quantum mechanically tunnel between the dots in the QCA cell and therefore, the switching of its polarisation.

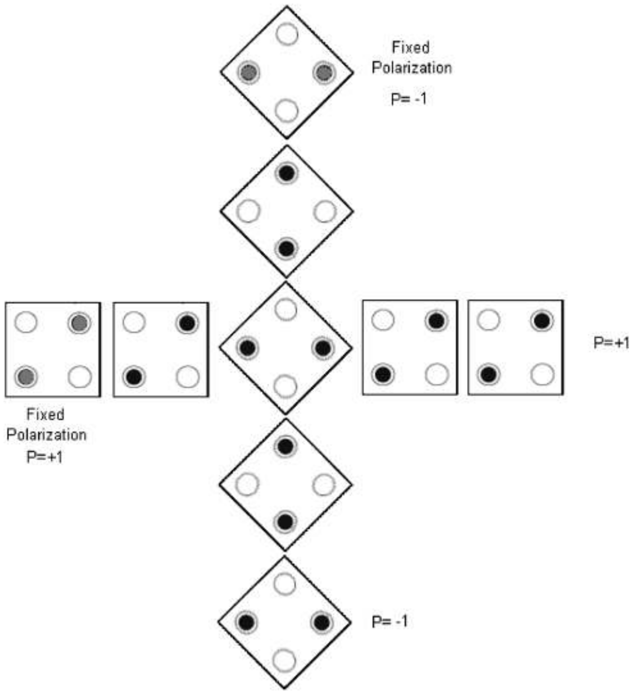

Fig. 5 *QCA coplanar wire-crossing*

When the clock signal (through the E field) is low, the potential barriers between dots are low because no polarisation exists ($P = 0$); so, the electrons move freely in the cell. When the clock signal (through the E field) is high, the potential barriers between dots are raised and electrons are localised, thus allowing the cell to be in one of the two stable states (i.e. the polarisation is $P = \pm 1$).

Clocking in QCA is implemented as follows: the QCA circuit layout is divided in adjacent zones that pertain to the four phases of the clock. Each of the clock signals is shifted by 90 degrees from the previous one as shown in Fig. 6. The slope of the transitions must be sufficiently small to maintain the cells near the ground state. This technique (known as quasi-adiabatic switching) has been proposed in [2]. In the first phase, the clock signal rises (the switch state) and the polarisation of the cell is dependent on the neighbouring cells. In the second phase (the hold state), the clock signal remains high, thus preventing the tunnelling of the electron pair and therefore, latching the information inside the cell. In this phase, the information can be used as input to the neighbouring cells that are in the switch state. In the third phase (the release state), the clock signal is lowered and the cell becomes unpolarised (i.e. $P = 0$). In the last phase (the relax state), the clock signal is kept low and the cell stays unpolarised.

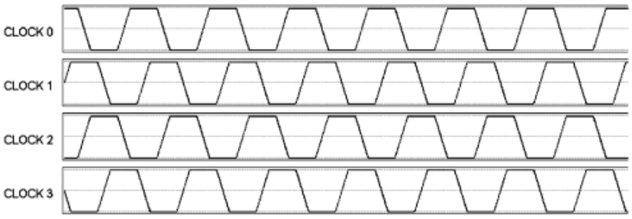

Fig. 6 *4-phase clock signals*

3 Memory in QCA

In this section, an overview of QCA memory architectures is presented. These architectures are based on the memory-in-motion paradigm [8] by which the value of the stored data is moved through different cells in a closed loop spanning at least four clocking zones. These architectures have different features, such as number of bits stored in a loop, access type (serial or parallel) and cell arrangement for the memory bank. Three architectures (parallel [11], serial [14] and H tree [8]) are considered in detail.

The parallel architecture of [11] is similar to a traditional CMOS-based memory architecture. The basic memory cell of this architecture is shown in Fig. 7. The data bit is stored in a loop, until the $WR/\overline{RD}$ control signal is low. When $WR/\overline{RD}$ is raised high, then the input bit is stored in the loop. The loop must be implemented using all zones of the four phase adiabatic switching technique for the clock, thus allowing the 'motion' of the stored bit. Starting from the basic cell, an array of memory cells can be implemented using the same architectures used in the design of CMOS memory banks. As for CMOS memories, this architecture is a 'true' random access memory; differently from the other QCA architectures, all bits of the memory array can be accessed at the same time.

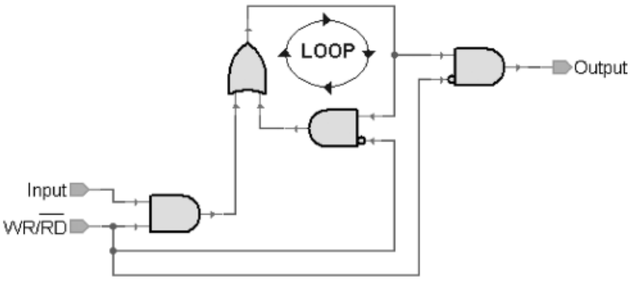

Fig. 7 *Basic cell of a parallel memory*

The serial architecture [14] is still based on a loop approach, but in this case, the loop is 'stretched' to store more than one bit. The control logic allows to synchronise the bit stored in the loop and for it to be addressable. A detailed description of the operation of the control logic can be found in [14]. This architecture allows to achieve a considerable reduction in area over a parallel implementation, because a single loop is used to store multiple bits. Owing to the serial access to the loop, the latency for the read/write operations is higher than in the parallel case. This may impact performance, because the latency of a serial QCA memory increases with the number of bits stored in a single loop.

As for the H tree memory, its significant feature is the implementation of the address decoding logic. This however, can be problematic for high density QCA memory designs. This memory utilises a recursive H structure, similar to the one commonly used to achieve zero-skew clock routing in systolic arrays; this memory is designed to have equal path length and uniform clocking zones. This architecture addresses some architectural problems present in previous QCA memory designs; however, its latency is extremely high owing to its serial operation. Moreover, the H tree memory utilises an addressing technique based on interleaved packets of data and address, thus requiring a customised design.

4 Proposed memory architecture

The proposed memory architecture can be considered as an evolution of the serial memory presented in [14]. It is referred to as 'hybrid' because it has serial write and parallel read capabilities. 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. The area required for its implementation is shown to be superior to other QCA architectures in terms of density. As a serial memory still incurs in slow access for both the write and read operations, so the proposed architecture uses a parallel read.

A block diagram of the proposed memory architecture is shown in Fig. 8. In this Figure, m loops of $2^n = N$ bits are arranged to form an m-bit word of $2^n = N$ locations (that can be accessed in parallel). Each loop has as inputs the n-bit address and the following additional signals: (1) the $R/W\#$ control signal that specifies if the loop is accessed by a write or read operation; (2) the serial data input D_{in}; (3) a $VALID$ control signal. The last signal is provided to each loop by the adder and allows the synchronisation of the write operation. The write operation must be performed serially on the loops and thus, the correct bit must be addressed. For both the read and write operations, addressing of the same bit (independently of the configuration of the shift register) requires the input address to be added to an offset (that is stored in a 2^n counter).

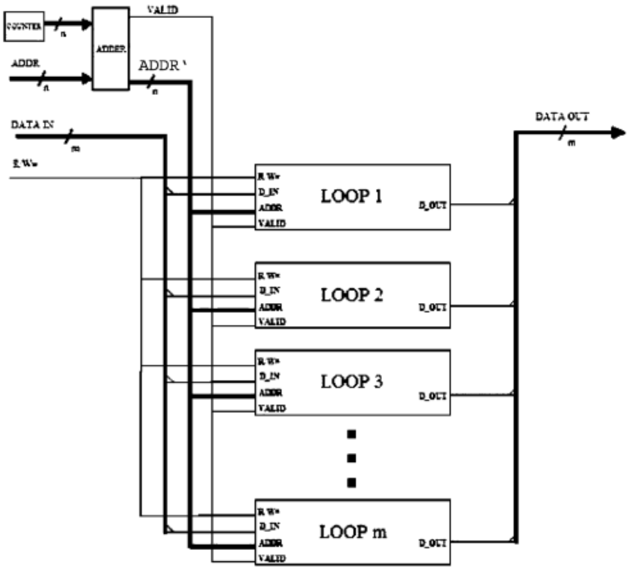

Fig. 8 *Block diagram of the hybrid QCA memory*

The operation of the hybrid memory can be described as follows: when a write is requested, this operation is performed provided the value of the 'biased' address $ADDR'$ is zero. When the NOR operation of the bits of the address is equal to one, then the write operation can incur in a delay of at most (2^n−1) clock cycles. If a read must be performed, the value of $ADDR$' is directly provided to the 2^n-to-1 demultiplexers of every loop, thus accomplishing an almost immediate (virtually zero delay) read operation for the addressed m-bits word.

The logic structure inside each loop is shown in Fig. 9, while its layout is shown in Fig. 13. The inputs of the loops are the m bits $ADDR'$, D_{in}, $VALID$ and the $R/W\#$ signals, while the output is the D_{out} signal. The write logic circuitry provides the inputs to the majority voter (MV) to either change the value of the stored information (placing the same new data at two of the three inputs), or leave it unchanged (placing a 0 and 1 at the two inputs). The logic gates of the write circuitry are realised using two inverters (similar to the one shown in Fig. 3) and three majority voters with a fixed polarised input. This circuitry can be identified in the lower right corner of the layout of Fig. 13. The read logic circuitry corresponds to the multiplexer shown in Fig. 9. Differently from CMOS design, in which these circuits are realized using tri-state or pass-transistor logic gates, in QCA they must be realised as a network of AND/OR gates. The multiplexer is therefore realised by initially decoding the address bits (A1 and A0 in Fig. 10) using an 1-out-of-2^n decoder that selects the addressed bit. This bit is an input of an OR gate, while the non-selected bits are masked to 0 prior to reaching the inputs of the same OR gate. Therefore, the output of the OR gate (i.e. DOUT) represents the output of the multiplexer too. To reduce the area, the circuit shown in Fig. 10 is employed.

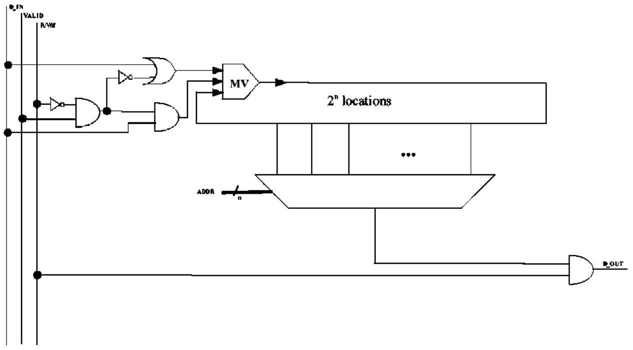

Fig. 9 *Loop implementatin of the hybrid memory*

As previously stated, the layout of the hybrid memory is derived from a serial memory with the addition of an output read circuitry. This permits to estimate the area of a serial memory cell to be half of the area of the hybrid memory. Moreover, differently from the hybrid memory,

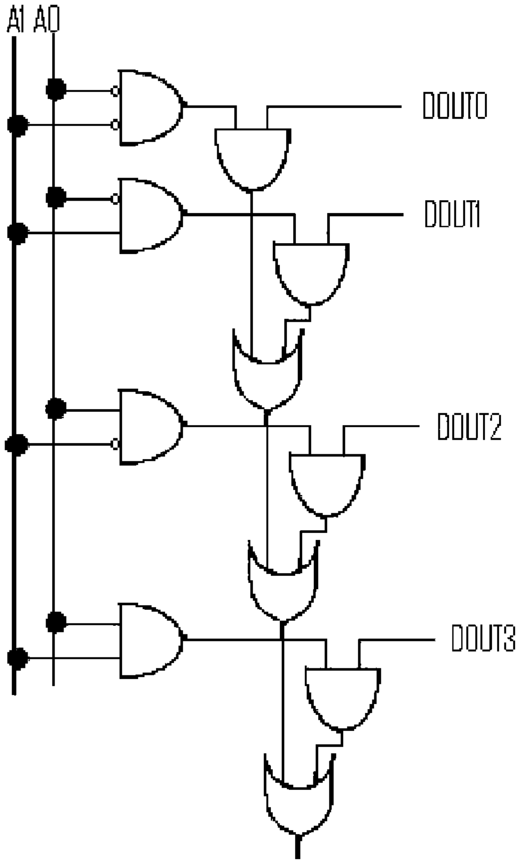

Fig. 10 *Read circuitry of the hybrid memory*

the area of a serial cell increases linearly with the number of stored bits and therefore the area/bit ratio is independent of the number of bits stored in the loop.

5 Comparison

In this section, the hybrid architecture is evaluated and compared with previously proposed QCA memories. The comparison metrics are the read/write latency (compared to a serial memory) and area (compared to a parallel implementation). Table 2 shows the qualitative features of the proposed memory architecture as well as the serial, parallel and H tree architectures discussed in detail in a previous section.

Table 2: Comparison

	Advantages	Disadvantages
Serial memory	Density (but SQUARES 160×120)/16 bits	Read/write latency
Parallel memory	Read/write latency	Area density (small size)
H tree memory	Density	Read/write latency Unconventional access mode
Hybrid memory	Read latency density	Write latency

In particular, the following features will be analysed in details.

- For the read operation, the proposed memory improves over a serial memory, because it has a virtually zero latency (which is comparable to a parallel memory). For the write operation, the proposed architecture has the same performance as a serial memory with a loop of equal size. The difference in latency for the two operations (read and write) suggests that the proposed hybrid architecture is better suited to applications in which memories are seldom written and often read (such as a PROM).
- The area of the proposed hybrid memory is reduced compared with a parallel architecture owing to the presence of decoding logic circuitry in each loop; however, some modifications in the layout can be implemented to allow sharing of duplicated resources between loops. The proposed hybrid memory allows to store more than one bit per loop, thus reducing the number of required loops (as compared to a parallel memory). The number of stored bits (per unit area) in the proposed memory improves over a parallel architecture for small loop size. For large loop size, the complexity of the decoding circuits reduces such improvement. The complexity of the decoding circuitry grows linearly with the number of address bits and therefore logarithmically with the number of bits stored in the loop. Thus, it can be shown that a loop storing up to 32 bits is still more area efficient than a parallel memory. Large memories can be generated by using an array of hybrid memory cells. Addressing in the hybrid memory is accomplished on a three-level arrangement. The first part of the address selects a row, the second part selects a column (like in standard CMOS RAM), while the last part (i.e. the least significant bits) is given as input address to the selected hybrid cell. The area comparison given in this section assumes that both parallel and hybrid memories are arranged in an array of identical cells. The decoding circuitry for row and column addressing is identical in all architectures. Moreover as in standard CMOS memories, the row and column decoding circuits represent only a small part of the area of a large memory; so, only the area of a single cell of the memory array is considered (in large memories the cell array occupies close to 95% of the chip area).

5.1 Latency

The proposed hybrid architecture combines the area efficiency of a serial memory [14] with a reduction in latency for the read operation (that is performed in parallel). An estimate of the latency is found by evaluating the complexity of the algorithm that implements the steps involved in the read and write operations. The read/write procedures in a single memory loop are reported in algorithmic form, i.e. Algorithm 1 for the serial architecture and Algorithm 2 for the proposed hybrid architecture. In Algorithm 1, the read and write operations of a serial memory are effectively a serial search of a list of N elements, where $N = 2^n$ is the number of addressable locations in the memory loops. This has a complexity (and therefore a latency) of $O(N)$ as requiring $N/2$ accesses on average. Algorithm 2 shows the read and write operations for the proposed hybrid memory; in this case, the read and write operations are different. By avoiding the loop and on average $N/2$ clock cycles for the read operation, a constant latency (immediate access with no dependency on the dimension of the loop) is encountered. The write and read latencies are different in the proposed hybrid memory. While the write latency increases linearly with the number of stored bits per loop (i.e. N), the read latency is constant. As for performance, these features compare quite closely to non-volatile (NV) memories. From ITRS [17] the foreseen timing performance of NV memories in 2018 (at a 25 nm feature size) are: read latency (the read cycle time) of 12 ns and program or write time (per cell) of 1 ms.

In QCA, the delay is related to the maximum clock rate. For molecular QCA (as likely implementation technology of QCA), the switching time is determined by the time it takes for a single electron to move across a molecule. By considering energy dissipation, it has been shown [18] that this results in QCA architectures which could operate at a density of 10^{12} devices/cm^2, an operating frequency of 100 GHz and a power dissipation of 100 W/cm^2 (i.e. a switching period of 10 ps). Therefore assuming (in the worst case) single-cell clocking zones and signals extracted from the loop traversing at least 100 cells, then the constant parallel read delay of the proposed memory architecture is 1 ns. For write latency (based on the assumptions discussed above), the worst case in a loop of N bits is $N \cdot 10$ ps. In this case, QCA will outperform CMOS-based NV memories in 2018.

Algorithm 1: *Serial memory*

```
Read Write access procedure to a serial memory
input: $ADDR$ Address;
       $DATA_{IN}$ Input Data;
       $R/\bar{W}$ counter;
output: $DATA_{OUT}$ Output Data
$i$ counter;
Loop[1 … N] memory loop;
begin
```

```
    while $ADDR \neq i$ do
        $i = i + 1$
    end
    if $R/\bar{W} = 0$ then
        $Loop(i) = DATA_{IN}$
    else
        $DATA_{OUT} = Loop(i)$
    end
end
```

5.2 Area overhead

The area of a QCA-based architecture is related to the number of QCA cells needed for its implementation. In many previous works on QCA design, circuit complexity was evaluated only by the number of QCA cells needed to realise the required functionality. However, differently from a transistor-based design, QCA cells realise the desired logic functions in addition to the interconnect. Moreover, such analysis must also take into account the area that is left unused in the layout. The unused portion of the layout is needed to limit the Coulombic interaction between QCA cells in different parts of the circuit. Therefore, computation of the area must take into account the geometric area occupied by the whole QCA circuit, i.e. the sum of the area occupied by the cells (implementing logic functions and interconnect) and the unused portion. This total area is referred to as the effective area.

Algorithm 2: *Hybrid memory*

```
Read Write access procedure to a hybrid memory
input: $ADDR$ Address;
       $DATA_{IN}$ Input Data;
       $R/\bar{W}$ counter;
output: $DATA_{OUT}$ Output Data
$i$ counter;
Loop[1 ... N] memory loop;
$ADD'$ modified address;
begin
    if $R/\bar{W} = 1$ then
        $ADD' = ADD + i$
        $DATA_{OUT} = Loop(i)$
    else
        while $ADDR \neq i$ do
            $i = i + 1$
            $Loop(i) = DATA_{IN}$
        end
    end
end
```

For example, consider Fig. 11; a simple majority voter (with interconnect) consists of 13 cells; however the effective area of the circuit is twice as much, i.e. a square of $5 \times 5 = 25$ cells must be instantiated once the circuit is placed on the layout. In the analysis, the effective area is used for evaluating and comparing the proposed memory architecture. Let d be the lateral dimension of a QCA cell and n_x and n_y be the count of cells in the x and y dimensions of the planar layout, then the effective area of a QCA layout is given by

$$A = (n_x d) \cdot (n_y d) = A_u + A_{un} \qquad (1)$$

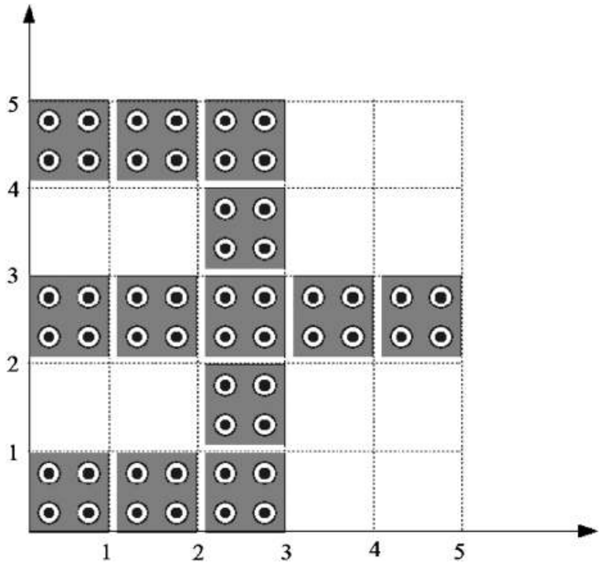

Fig. 11 *Effective area of a majority voter with interconnect*

where A_u (A_{un}) is the used (unused) portion of the area occupied by the QCA cells in the logic and interconnect circuits.

Consider initially a 1 × 4 memory; two layouts can be compared.

- Consider the layout of a 1 × 4 parallel architecture as shown in Fig. 12.

The effective area of this layout has lateral dimensions of 82 and 55 QCA cells. Therefore from (1) its effective area is $A = 0.455\,\mu m^2$ for $d = 10\,nm$. From Fig. 12 it is evident that a large area is used to realise the address decoding logic and four loops are required (each with a read/write circuitry).

- Figure 13 shows the proposed hybrid memory, thus achieving a more compact layout. In this case, the effective area has dimensions of 53 and 28 QCA cells. Hence, for $d = 10\,nm$, $A = 0.148\,\mu m^2$. The layout of the hybrid memory is only 32% of the layout of a parallel memory architecture. The effective area is reduced owing to the following reasons: (1) the loop structure is realised in a smaller area; (2) the control circuitry is shared between the four bits.

Consider next the case of a 4 × 4 memory. In this case, the hybrid memory architecture can be extended to a layout

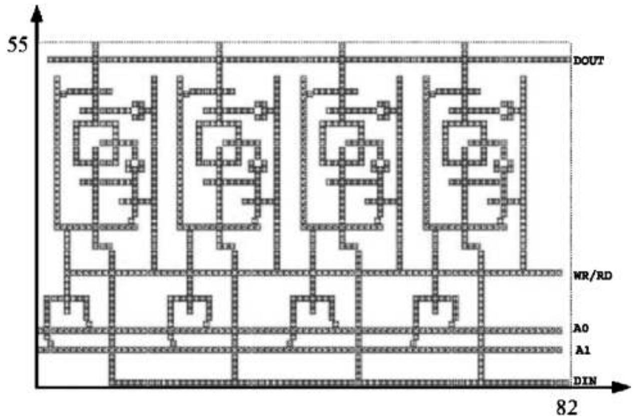

Fig. 12 *Layout of a 1 × 4 parallel memory*

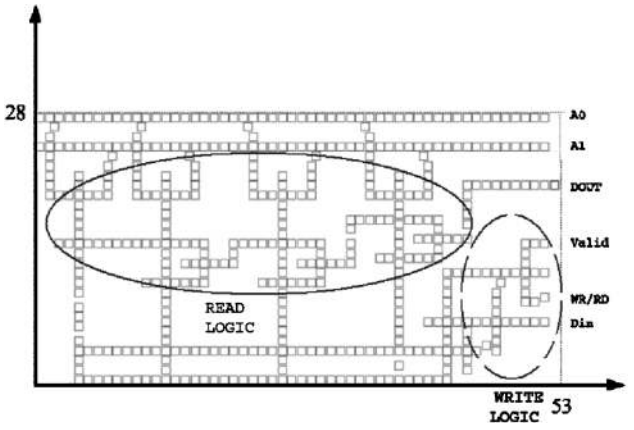

Fig. 13 *Layout of 1 × 4 hybrid memory*

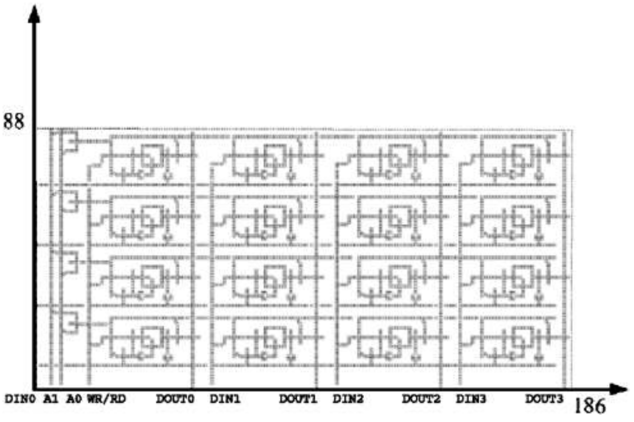

Fig. 14 *Layout of 4 × 4 parallel memory*

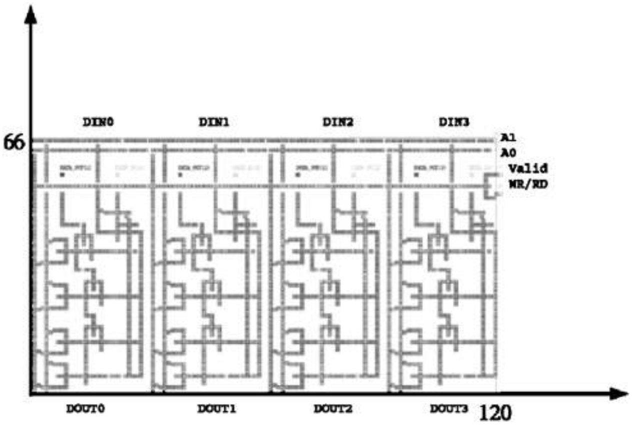

Fig. 15 *Layout of a 4 × 4 hybrid memory*

with an H tree structure. Therefore, three cases can be considered for comparison purposes.

- Figure 14 shows the layout of a 4 × 4 parallel memory. The dimensions of the effective area are 88 and 186 cells such that for $d = 10\,\text{nm}$, $A = 1.63\,\mu\text{m}^2$.
- Figure 15 shows the layout of 4 × 4 hybrid memory generated by modular replication of the layout in Fig. 8. The dimensions of the effective area are 66 and 120 cells and $A = 0.792\,\mu\text{m}^2$. Note that the effective area of the 4 × 4 memory is larger than four times the area of a 1 × 4 memory because the control signals must be routed to the loops (that in this case are longer).
- A new arrangement can be used in the design of the hybrid memory; this consists of sharing the signal routing resources, thus reducing the area. Figure 16 shows a H tree-based 4 × 4 hybrid memory whose effective area has dimensions given by 130 and 52. Therefore for $d = 10\,\text{nm}$, $A = 0.676\,\mu\text{m}^2$. The reduction in area is due to sharing of the interconnections in the array and its modular replication in implementing larger memory architectures.

Table 3 summarises the effective areas of the parallel, hybrid and serial memories analysed and compared in this

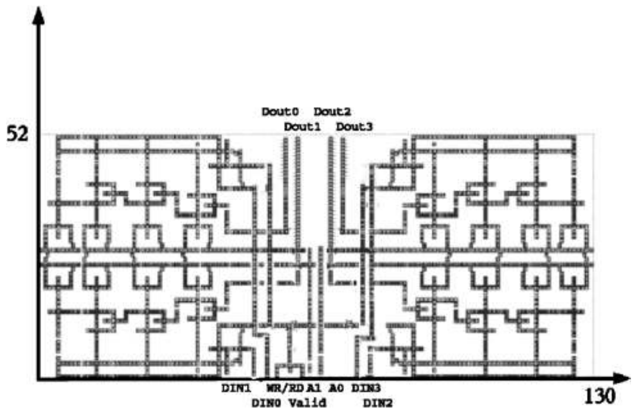

Fig. 16 *H tree structure layout of a 4 × 4 hybrid memory*

Table 3: Area comparison

	A_{hyb} (μm^2)	A_{par} (μm^2) same size	A_{ser} (μm^2) same size	R
1 × 4	0.148	0.455	0.074	32%
4 × 4	1.36	1.63	0.296	80%
4 × 4 H tree	0.676	1.63	0.296	41%

Table 4: Comparison of A_u and A_{un} for the proposed layouts

	A (μm^2)	A_u (μm^2)	A_{un} (μm^2)	A_u (%)	A_{un} (%)
1 × 4 SER	0.074	0.021	0.053	28	72
1 × 4 PAR	0.451	0.124	0.326	27	73
1 × 4 SER	0.296	0.084	0.212	28	72
1 × 4 HYB	0.148	0.0481	0.1	32.5	67.5
1 × 4 PAR	1.63	0.392	1.24	24	76
1 × 4 HYB	0.792	0.233	0.558	29.5	70.5
1 × 4 H tree	0.676	0.183	0.49	27.1	72.9

paper. Moreover, the ratio $R = A_{hyb}/A_{par}$ (where A_{hyb} (A_{par}) is the effective area of a hybrid memory architecture (parallel memory)) is also reported in the Table. By increasing the size, the improvement in area of an hybrid memory is reduced due to the decoding circuitry for each loop.

As a final remark, Table 4 shows the ratios of A_u and A_{un} for the previously introduced layouts. A_u varies from 24% to 32.5% i.e. A_{un} accounts for the larger portion of the effective area.

Different densities can be realised for QCA-based memories: from 1 GBits/cm^2 to 5 GBits/cm^2 for metal QCA and 10 GBits/cm^2 to 50 GBits/cm^2 for molecular QCA (at $d = 3$ nm). Based on these densities and the comparison analysis presented previously, the performance metrics outlined in [17] are well within the reach of QCA. Moreover, QCA may also substantially improve over the predictions for current technologies, such as CMOS.

6 Conclusions

In this paper, a novel hybrid memory architecture for QCA implementation has been proposed. The hybrid nature of this architecture refers to the parallel read operation on multiple bit loops, while still having a serial write. The different operational mechanisms for the operations (read and write) result in two advantages over previous architectures: (i) the parallel read mechanism allows a reduced latency (an almost immediate access is accomplished) compared with $O(N)$ latency for serial memories (where N is the number of bits in a loop); (ii) a reduction in area is accomplished compared with a parallel memory, thus increasing density (this is possible owing to sharing the interconnects, the reduced control logic and a different loop implementation). The hybrid memory is best suited in applications in which the memory is often read and seldom written (such as a PROM).

A novel figure of merit referred to as effective area has been introduced in the analysis. The effective area is defined as the geometric area of the memory which includes the area of the QCA cells for the logic and interconnect circuits in addition to the unused portion of the Cartesian plane (that separates the QCA cells to avoid unwanted Coulombic interactions among them).

7 References

1 Lent, C.S., Tougaw, P.D., and Porod, W.: 'Quantum cellular automata: the physics of computing with arrays of quantum dot molecules'. PhysComp 94: Proc. Workshop on Physics and Computing, IEEE Computer Society Press, 1994

2 Lent, C.S., and Tougaw, P.D.: 'A device architecture for computing with quantum dots', *Proc. IEEE.*, 1997, **85**, pp. 541–557

3 Timler, J., and Lent, C.S.: 'Power gain and dissipation in quantum-dot cellular automata', *J. Appl. Phys.*, 2002, **91**, pp. 823–831

4 Compano, R., Molenkamp, L., Paul, D.J.: 'Technology roadmap for nanoelectroincs', in European Commission IST programme, Future and Emerging Technologies

5 Niemier, M.T., and Kogge, P.M.: 'Logic in wire: using quantum dots to implement a microprocessor'. Int. Conf. on Electronics, Circuits, and Systems (ICECS '99), Cyprus, September 1999

6 Amlani, I., Orlov, A., Toth, G., Bernstein, G.H., Lent, C.S., and Snider, G.L.: 'Digital logic gate using quantum-dot cellular automata', *Science*, 1999, **284**, pp. 289–291

7 Niemier, M.T., Rodrigues, A.F., and Kogge, P.M.: 'A potentially implementable FPGA for quantum dot cellular automata'. Workshop on Non-Silicon Computation (NSC-1), Boston, MA, USA, 2002

8 Frost, S.E., Rodrigues, A.F., Janiszewski, A.W., Rausch, R.T., and Kogge, P.M.: 'Memory in motion: a study of storage structures in QCA'. First Workshop on Non-Silicon Computing, 2002, Vol. 2

9 Tougaw, P.D., and Lent, C.S.: 'A potentially implementable FPGA for quantum dot cellular automata'. Workshop on Non-Silicon Computation (NSC-1), Boston, MA, 1994

10 Dimitrov, V.S., Jullien, G.A., and Walus, K.: 'Quantum-dot cellular automata carry-look-ahead adder and barrel shifter'. IEEE Emerging Telecommunications Technologies Conf., 2002

11 Walus, K., Vetteth, A., Jullien, G.A., and Dimitrov, V.S.: 'RAM design using quantum-dot cellular automata'. Technical Proc. 2003 Nanotechnology Conf. and Trade Show, March 2003, Vol. 2, pp. 160–163

12 Tougaw, P.D., and Lent, C.S.: 'Dynamic behavior of quantum cellular automata', *J. Appl. Phys.*, 1996, **80**, pp. 4722–4736

13 Niemier, M.T., and Kogge, P.M.: 'Problems in designing with QCAS: layout = timing', *Int. J. Circuit Theory Appl.*, 2001, **29**, (1), pp. 49–62

14 Berzon, T.J., and Fountain, D.: 'A memory design in QCAS using the squares formalism'. Proc. Ninth Great Lakes Symp. VLSI '00, March 1999, pp. 166–169

15 Ottavi, M., Pontarelli, S., Vankamamidi, V., and Lombardi, F.: 'Design of a QCA memory with parallel read/serial write'. Proc. IEEE Computer Society Ann. Symp. on VLSI, May 2005, pp. 292–294

16 Blair, E.P., and Lent, C.S.: 'Quantum-dot cellular automata: an architecture for molecular computing'. Int. Conf. on Simulation of Semiconductor Processes and Devices SISPAD 2003, Sept. 2003, Vol. 2, pp. 14–18

17 'International Technology roadmap', 2003, http://public.itrs.net/

18 Timler, J., and Lent, C.S.: 'Maxwell's demon and quantum-dot cellular automata', *J. Appl. Phys.*, 2003, **94**, pp. 1050–1060