JOURNAL OF NANO- AND ELECTRONIC PHYSICS
Vol. **9** No 1, 01021(5pp) (2017)

ЖУРНАЛ НАНО- ТА ЕЛЕКТРОННОЇ ФІЗИКИ
Том **9** № 1, 01021(5cc) (2017)

A Processing in Memory Realization Using Quantum Dot Cellular Automata (QCA): Proposal and Implementation

P.P. Chougule[1], B. Sen[2], R. Mukherjee[2], P.S. Patil[1], R.K. Kamat[3], T.D. Dongale[1,*]

[1] *Computational Electronics and Nanoscience Research Laboratory, School of Nanoscience and Biotechnology, Shivaji University, Kolhapur- 416004 India*
[2] *Department of Computer Science and Engineering, NIT, Durgapur-713209 India*
[3] *Embedded System and VLSI Research Laboratory, Department of Electronics, Shivaji University, Kolhapur-416004 India*

(Received 13 November 2016; revised manuscript received 07 February 2017; published online 20 February 2017)

Processing in Memory (PIM) is a computing paradigm that promises enormous gain in processing speed by eradicating latencies in the typical von Neumann architecture. It has gained popularity owing to its throughput by embedding storage and computation of data in a single unit. We portray implementation of Akers array architecture endowed with PIM computation using Quantum-dot Cellular Automata (QCA). We present the proof of concept of PIM with its realization in the QCA designer paradigm. We illustrate implementation of Ex-OR gate with the help of QCA based Akers Array and put forth many interesting potential possibilities.

Keywords: QCA, Akers logical array, Processing in memory, Computer architecture.

DOI: 10.21272/jnep.9(1).01021 PACS numbers: 03.67.Lx, 85.35.Be, 07.05.Tp

1. INRODUCTION

Disparity posed by the processing time by the processor versus data access speed is a bottleneck of the von Neumann architecture which has apparent more in the recent backdrop of increasing requirement of computing power. Researchers are striving hard to address this bottleneck using various methods such as induction of cache memory, implementation of branch predictor algorithms, realization of morphware and configware [1]. In recent years, Processing in Memory (PIM) architecture has gained popularity in the wake of increasingly complex application domains such as big data analytics, machine learning, soft computing and other emerging computing paradigms [2-3]. As against the requirement of two separate units for processing and storing of the data in the conventional von Neumann architecture, PIM architecture accomplishes the same only with one integrated unit. Therefore PIM emerges as the winner in terms of speed, feature size and power consumption. Scholarly literature reveals various methods to implement the PIM architecture [4-7]. In this paper we present implementation of PIM using combination of Quantum Dot Cellular Automata (QCA) and Akers array.

The Akers rectangular logic arrays, first proposed in 1972 by S. B. Akers [8] has a great developmental history. However the same has been less traversed for the purpose of PIM computation. This kind of architecture has enormous potential with the implementation possibilities through nanotechnology which can be referred to as 'nanoscale PIM'. The obvious advantages such as reduction in feature size leads to augmenting the computing speed as well as significant reduction the power consumption. In this regards, QCA is a promising and reliable technique as described by so many papers as the future electronics [9-11]. Since the Akers array is known for its PIM capabilities, we have integrated the same with QCA so as to synergize their wherewithal for improving the computing metrics. Realization of the same is depicted in this paper in a software environment of QCA designer suite.

Rest of the paper is organized as follows, after general introduction; second section proposes the QCA Akers logic array. This is followed by evaluation of primitive logic cell and utilizing the same for forming an Ex-OR gate. Throughout the paper simulation results are presented. At the end, conclusion and future work is reported.

2. BACKGROUND DETAILS

This section explains basic background related to QCA and Akers logic array. This is followed by the details of cell structure and functional operation.

2.1 Quantum Dot Cellular Automata

QCA is a cellular automata of square cells, where each cell consist of four quantum dots and two electrons localized in it. Quantum dots are placed at the corners of QCA cell. The electrons are free to tunnel between two adjacent quantum dots. But due to the electrostatic (van der Waals' force) repulsion force, electrons will get localized in diagonal quantum dots. So, there will be only two probabilities of electrons position in QCA cell. First diagonal position is considered as '0' and second diagonal position is considered as '1' as shown in fig. 1 (a). Depending on these considerations different circuits can be designed using QCA. These kinds of circuits are depending on the interaction between bistable QCA cells.

The reliability of the QCA system is depends upon clocking and each QCA system is depends upon four clocks, viz. clock 0, clock 1, clock 2 and clock 3 as shown in Fig. 1(b). Every clock has four phases i.e. switch

* tdd.snst@unishivaji.ac.in

phase, hold phase, release phase and relax phase. When clock '0' will be in switch phase, clock '1' will be in hold phase, clock '2' in release phase and clock '3' in relax phase. In switch and release phase electrons are allowed to tunnel between adjacent quantum dots and attend maximum stable place. In hold phase electrons are not allowed to tunnel. They are strictly localized and adjacent cell doesn't get affected by neighboring cells. In relax phase, there are no electrons in any quantum dots, hence this kind of clocking makes QCA a strong candidate for future electronics.

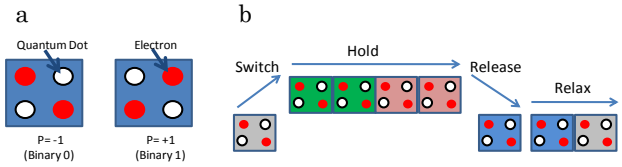

Fig. 1 – (a) QCA cell structure. It consists of four quantum dots and two electrons. The output depends upon the diagonal position of electrons. (b) Clocks in QCA and its four phases

2.2 Akers Logic Array

Akers logic array is proposed by famous mathematician S.B. Akers in 1972. It is two dimensional logic array based on rectangular grids. Every logic cell in the original design has three inputs and its output exhibits following function: [8]

$$F(X,\ Y,\ Z)\ = X + YZ \qquad (1)$$

In his famous publication, Akers had given four alternative equations which can be used for In-memory computation. From that four equations, we are using above eq. (1) for proposed architecture. The primitive cells of Akers logic array consist of three inputs '*X*', '*Y*' and '*Z*' and depending upon these inputs it provide output '*F*'. This output acts as an input for neighbor cells as illustrated in Fig. 2(a). Fig. 2(b) represents typical 3×3 Akers logic array.

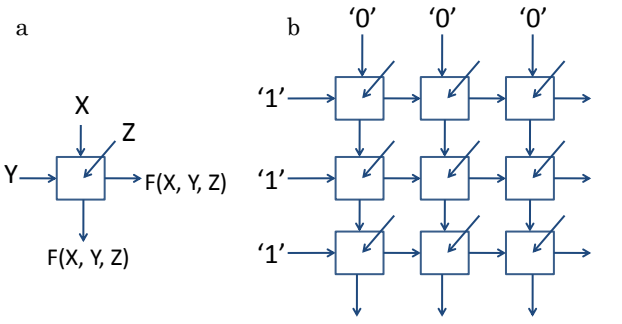

Fig. 2 – Akers array (a) A logic cell with three inputs *X*, *Y*, and *Z* and two identical outputs (b) Akers 3×3 array structure

Here '*X*' and '*Y*' are fixed inputs and '*Z*' is controlled input i.e. the output of each cell is completely depends upon '*Z*'. The '*Z*' input is virtual input and it stores its value in every cell. This indicate that, every cell stores a single bit of memory and each cells output act as input for neighboring cells. The execution of Boolean function is performed by arranging the cells in a specific manner. We have final output at a lower rightmost cell of an array. We propose to use the input variable '*Z*' of Akers logic array to store the internal state of a modified QCA Akkers cell. In this regards the inputs of the executed Boolean function are treated as the stored data in QCA cell. This facilitates the modified QCA Akers logic array to multiplex the functions viz. processing and storing the data. Using this functionality in the form of primitive, implementation of PIM is possible as depicted in the remainder of the paper. We further elaborate the structure and operations of logic cells in the following section.

3. EVALUATION OF PRIMITIVE LOGIC CELL

The basis of the proposed logic cell functionality is the eq. (1). The primitive logic cell circuit structure is shown in Fig. 3 (a). We have simulated the output of the above said logic structure using QCA Designer suite which is an freeware developed with the research effort by the Walus Group at the University of British Columbia for creating, designing and simulating designs based on Quantum Dot Cellular Automata (QCA) [12]. The digital simulation engine, one of the modules of the suite simulates the cells on the basis of null, logic 1 and logic 0 along with the appropriate clock stimulus, the system iterates till the convergence in terms of stable state is reached [13]. The design is simulated as bi-stable simulation with the parameters set as default in the simulation suite. The simulation outcome is shown in Fig. 3 (b). The inputs of cell '*X*' and '*Y*' are given as fixed input i.e. zero and one respectively. The control input '*Z*' is used for storing the logical state of QCA cell i.e. Qz. By using these control inputs the desired output from the Akers logic cell is derived. The designed circuit of primitive logic cell satisfies the output of Akers logic cell previously described in eq. 1.

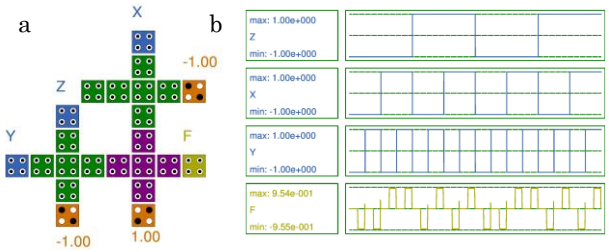

Fig. 3 – Logic cell. (a) Proposed primitive logic cell using QCA. (b) Its simulation results which gives output of eq. (1)

After establishing the functioning of the primitive logic cell and its confirmation through simulation we could implement basic logic gates by using the primitive logic cell. Since the logic gates can be used to form the complex logic systems, the intent is to evaluate their functionality. We exemplify here implementation of Ex-OR gate owing to its hybrid nature.

4. IMPLEMENTATION DETAILS OF PIM ARCHITECTURE

In order to evaluate modified QCA Akers logic array, we consider Ex-OR gate as a representative candidate. The two input Ex-OR gate using QCA Akers logic array is shown in Fig. 4. This Ex-OR gate structure is same as original Akers arrays Ex-OR gate but differ-

ence lies in the basic cell structure. One of the major drawbacks of the conventional Akers logic Ex-OR gate is that, the number of unit cells increases exponentially with the increase in the inputs [14]. As against the usage of primitive logic cell implemented herein results in considerably miniature size of QCA cells and therefore serves as the most apt choice for the next generation computing architecture.

Two input QCA Akers logic Ex-OR gate consists of four Akers cells, as shown in Fig. 4(a). Two QCA cells circuits are connected in parallel manner as shown in Fig. 4 (b). The QCA Akers two input Ex-OR gate thus formed, was simulated in QCA Designer suite. The simulation parameters are given in Table 1. Fixed inputs as per the Akers logic have been provided as shown in Fig. 4(b), while various combinations were instantiated at 'A' and 'B'. The electrostatic QCA does not store data itself hence we have provided an external memory block for each input [15-16]. This is a square memory block which consists of two clocks and the difference between the input and output cell also have two clocks. So, even if we change the signal, earlier input will be stored within memory block for two more clocks. Therefore, the signal can be stored in these blocks for longer time. In this kind of structure, an input will go for computation in main circuit and at same time that input will be get stored in the memory block.

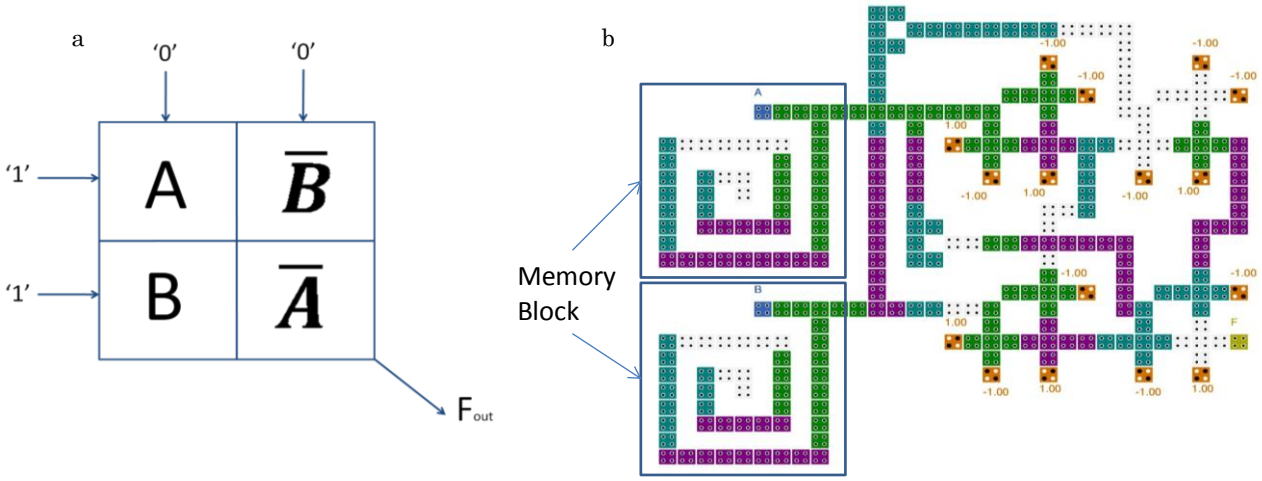

Fig. 4 – Two input Ex-OR gate. (a) Original Akers two input Ex-OR gate. (b) QCA Akers two input Ex-OR gate

Table 1 – Simulation parameters set in the QCA designer suite

Parameters	Values
Temperature	1 K
Relaxation Time	1.000000e-015 s
Time step	1.000000e-015 s
Clock High	9.800000e-022 J
Clock low	3.800000e-023 J
Clock Shift	0.000000e+000
Clock Amplitude factor	2.000000
Radius of effect	80.000000 nm
Relative permittivity	12.900000
Layer separation	11.500000 nm
Convergence Tolerance	0.001000
Number of samples	128000
Maximum Integrations per sample	100
Parameters	Values
Temperature	1 K

Simulation output is shown in Fig. 5 confirms the Ex-OR behavior. Thus the QCA based Akers logic array exhibits Ex-OR logic functions. Moreover the control signal of QCA multiplexes the logic functionality along with the memory operation. Thus this implementation serves as one of the fundamental building blocks for the PIM architecture.

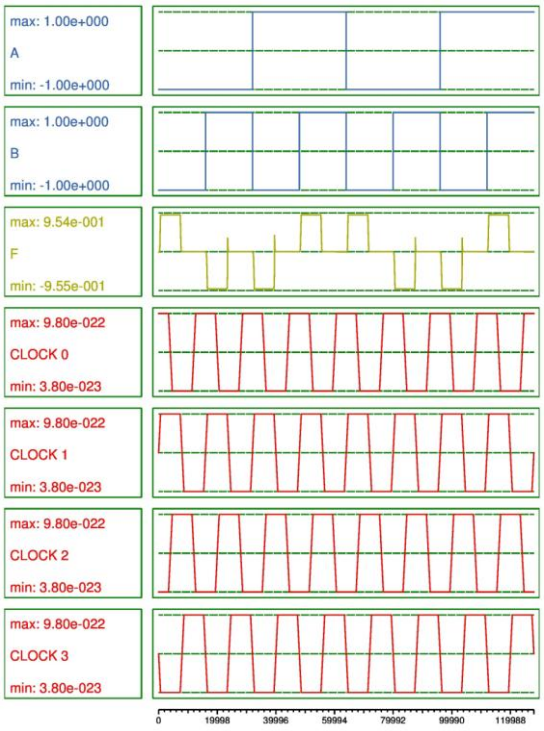

Fig. 5 – Simulation results of QCA Akers two input XOR gate

The power consumption of proposed architecture is much less than the conventional CMOS circuits. The power analyses of XOR gate at different kink energy (Ek) values are shown in Fig. 6. The different cells consuming different energy and these power dissipation values are very less compared to the conventional XOR gate [17]. QCA cells behavior is highly depend on the temperature and kink energy. The kink energy is related to the surface science. As per the Terrace ledge kink model, the energy required to remove an atom from the surface depends on the number of bonds to other surface atoms which must be broken. The polarization result of proposed architecture is shown in Fig. 7. This polarization helps the circuit to keep signal strength strong. Whenever the clock changes, the polarization also changes and it keeps signal strong.

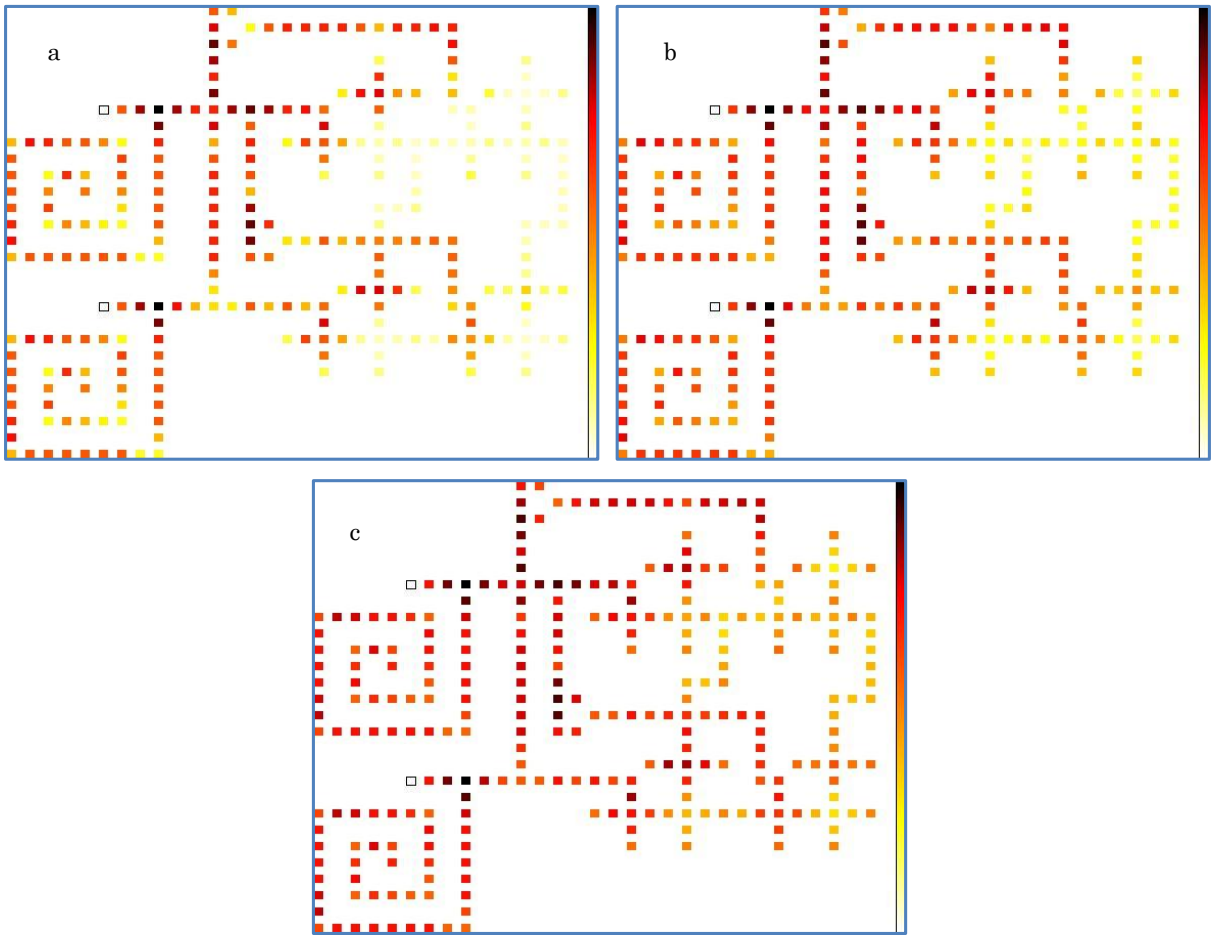

Fig. 6 – Power dissipation analysis of QCA Akers XOR gate (a) At Ek 0.5 meV (b) At Ek 1.0 meV (c) At Ek 1.5 meV

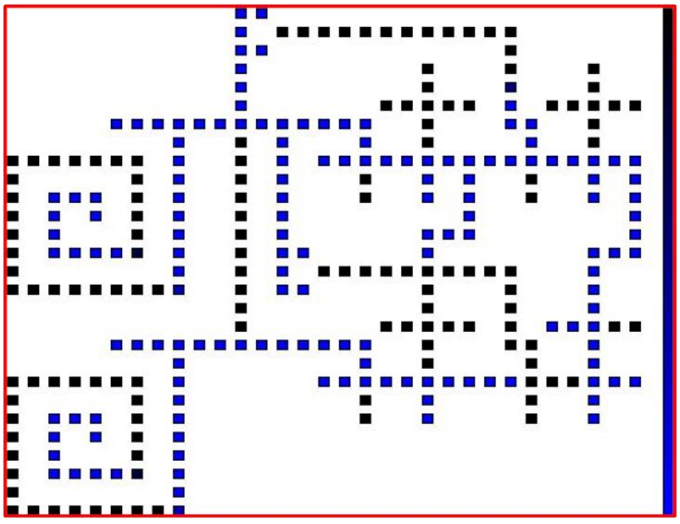

Fig. 7 – Polarization analysis of QCA Akers XOR gate

5. CONCLUSION AND FUTURE WORK

The PIM architecture is implemented (for first time) by utilizing the computational power of QCA and Akers logic array. We proposed to multiplex this array as a memory and also for performing various Boolean operations. This unique attribute makes it a competitive candidate useful for the PIM computing applications. The Ex-OR gate realization was converged at much less complexity with 184 cells, 19 clock zones and with a footprint of 0.23 μm² which is significantly smaller than its CMOS counterpart [17]. Table 2 compares the power dissipation at different value of the kink energy. The kink energy is directly associated with the energy cost of the cells. The power dissipation is way smaller than the conventional CMOS counterpart in which it is of the order of few mWs. Thus the combination of QCA and Akers array provides many additional benefits over the conventional CMOS design. Moreover their synergic integration leads to the design with reduction in the power consumption and feature size, with improvement in the speed. We are in a process of extending the design to form a reconfigurable microprocessor.

Table 2 – Energy analysis of XOR gate

Parameters	E_k = 0.5 (meV)	E_k = 1.0 (meV)	E_k = 1.5 (meV)
Max Kink Energy	0.00148 meV	0.00148 meV	0.00148 meV
Max Energy dissipation of circuit	0.75863 meV	0.84063 meV	0.96062 meV
Max Energy dissipation vector	03	03	03
Average Energy dissipation of circuit	0.41697meV	0.55105 meV	0.71821 meV
Max Energy dissipation among all cells	0.00740 meV	0.00729 meV	0.00736 meV
Max Energy dissipation vector	13	21	21
Min Energy dissipation of circuit	0.09050 meV	0.27342 meV	0.48326 meV
Min Energy dissipation vector	11	11	11
Average Leakage Energy dissipation	0.09114 meV	0.27342 meV	0.48566 meV
Average Switching Energy Dissipation	0.32583 meV	0.27763 meV	0.23255 meV

REFERENCES

1. J. Becker, R. Hartenstein, *J. Syst. Arch.* **49**, 127 (2003).
2. D.P. Zhang, N. Jayasena, A. Lyashevsky, J. Greathouse, M. Meswani, M. Nutter, M. Ignatowski, *Proceedings of the ACM SIGPLAN Workshop on Memory Systems Performance and Correctness*, 1 (2013).
3. J. Torrellas, *International Conference on Computer Design*, 3 (2012).
4. J. Ahn, S. Yoo, O. Mutlu, K. Choi, *ACM/IEEE 42nd Annual International Symposium on Computer Architecture* (ISCA), 336 (2015).
5. M. Eshaghian-Wilner, *Bio-inspired and nanoscale integrated computing* (N. J.: Wiley: 2009).
6. C. Fu, D. Wen, X. Wang, X. Yang, *J. Syst. Arch.* **56**, 384 (2010).
7. R. Nair, S. Antao, C. Bertolli, P. Bose, J. Brunheroto, T. Chen, C. Cher, C. Costa, J. Doi, C. Evangelinos, B. Fleischer, T. Fox, D. Gallo, L. Grinberg, J. Gunnels, A. Jacob, P. Jacob, H. Jacobson, T. Karkhanis, C. Kim, J. Moreno, J. O'Brien, M. Ohmacht, Y. Park, D. Prener, B. Rosenburg, K. Ryu, O. Sallenave, M. Serrano, P. Siegl, K. Sugavanam Z. Sura, *IBM J. Res. Dev.* **59**(2-3), (2015).
8. S.B. Akers, *IEEE Trans. Comput.* **21**, 848 (1972).
9. G.L. Snider, A.O. Orlov, I. Amlani, X. Zuo, G. Bernstein, C. Lent, J. Merz, W. Porod, *J. Vac. Sci. Technol. A* **17**, 1394 (1999).
10. I. Amlani, A. Orlov, G. Toth, G. Bernstein, C. Lent, G. Snider, *Science* **284**, 289 (1999).
11. E. Yaakobi, A. Jiang, J. Bruck, *IEEE International Symposium on Information Theory Proceedings (ISIT),* pp. 2369 (2013).
12. K. Walus, T. Dysart, G. Jullien, R. Budiman, *IEEE T. Nanotech.* **3**, 26 (2004).
13. V. Mardiris, I. Karafyllidis, *J. Circuit Syst. Comp.* **19**, 349 (2010).
14. Y. Levy, J. Bruck, Y. Cassuto, E. Friedman, A. Kolodny, E. Yaakobi, S. Kvatinsky, *Microelectr. J.* **45**, 1429 (2014).
15. E. Ganesh, and V. Krishnan, *Int. J. Recent Trends in Eng. Tech.* **2**, 83 (2009).
16. P.P. Chougule, B. Sen, T.D. Dongale, arXiv:1607.05065.
17. Y. Mortazavi, *Digital Electronics Course Report*, 1 (2004).