# Computing with Nanoscale Memory: Model and Architecture

*(Invited Paper)*

Somnath Paul and Swarup Bhunia
Department of Electrical Engineering and Computer Science
Case Western Reserve University, Cleveland, OH-44106, USA
E-mail: {sxp190, skb21}@case.edu

***Abstract*—Emerging nanoscale devices hold tremendous potential in terms of integration density, low power operation and switching speed. Unlike CMOS devices, however, majority of these devices are not suitable for implementing cascaded, irregular logic structure. On the other hand, dense and periodic structures of most emerging nanodevices as well as their bi-stable nature make them amenable to large high-density memory array design. Moreover, self-assembly of many nanostructures is efficient for a bottom-up system design flow. Hence, reconfigurable computing paradigms that use memory as underlying computing element, appear promising for these devices. In this paper, first we study nanoscale FPGA, which extends conventional spatial CMOS FPGA architecture using nanoscale memory and interconnect. Next, we focus on a time-multiplexed memory based computing paradigm that employs two-dimensional memory for improved performance, integration density and resource usage.**

## I. Introduction

In the quest of a potential alternative to CMOS at the end of its roadmap [1], multitude of research efforts have been directed towards investigating novel devices with interesting and unique switching characteristics. An extensive review of these devices can be found in [1] and [2]. These emerging nanoscale devices can be classified into two broad types:

- Memory devices: Some of the promising memory devices include a) ferroelectric random access memory (FeRAM) [3], b) spin torque transfer random access memory (STTRAM) [4], c) phase change random access memory (PCRAM) [5], d) nanoelectromechanical memory (NEMM) [6], and e) molecular memory [7]. Unlike conventional CMOS-based static and dynamic random access memory, these memory devices are suitable for non-volatile operation, which is attractive in many applications including reconfigurable computing frameworks.
- Logic and information processing devices: These array of devices include a) single-electron transistors (SET) [8], b) carbon nanotube field effect transistor (CNTFET) [9], c) semiconductor nano-wires, d) quantum-dot cellular automata (QCA) [10], and e) molecular devices such as molecular diodes [11,12,13].

Most of these emerging nanodevices hold tremendous potential in terms of integration density (in the order of $10^{10}$ devices/$cm^2$) [23], low power operation and higher switching speed. In spite of differences in the working principle of these individual devices, they bear the following common characteristics:

- Since nanodevices have limited gain [9, 13] and interconnections with three-terminal transistors using bottom-up approach is challenging, CMOS based level restoring interface circuitry are typically required for realization of logic circuits. Such interleaved nano-CMOS hybrid circuit reduces the overall integration density and complicates the fabrication process.
- The bottom-up self-assembly techniques are not suitable in realizing arbitrarily complex layout patterns often required for realizing irregular cascaded logic structures. Instead, they are more amenable towards the fabrication of periodic regular structures.
- They have high integration density. However, they typically suffer from high defect rate (e.g. 10% or more of the devices may be defective [34] compared to less than 100 parts per million for CMOS). Furthermore, any small variation in atomic scale is likely to manifest significantly in device parameters.

The above characteristics suggest that transforming these nanodevices into nanocomputing framework may require non-traditional computing model and architecture. Instead of using nanodevices as ‘better switches’ in existing ‘switch based’ implementations of logic functions, different approaches are needed to fully exploit the benefits of nanodevices while addressing the challenges, such as high defect rate and lack of gain. Reconfigurable computing frameworks appear attractive for these devices and have been extensively investigated [11, 15, 16, 20]. The reason as outlined in [1], is that emerging nanodevices (e.g. molecular crossbars) are particularly suitable for the design of dense and regular memory fabric rather than irregular CMOS-like multi-level logic structure. Memory preserves the density advantage offered by the emerging nanodevices. Since many nanodevices need to be interfaced with CMOS logic, a dense nanoscale memory structure can substantially reduce the CMOS overhead per memory cell. Moreover, memory structures realized using nanodevices have very well defined CMOS interfaces [16] thus facilitating CMOS-nano hybridization.

Reconfigurability facilitates mapping arbitrary applications in a generic periodic fabric using bottom-up manufacturing. It

978-1-4244-4958-3/09/$25.00©2009 IEEE

also helps to alleviate the effects of manufacturing defects. It is well-known that redundancy or error-correction based defect tolerance schemes for random logic incur significant overhead. However, due to the regular structure of the memory, high defect rates can be tolerated by either re-mapping to non-defective locations [26], or by the use of redundant rows and columns [24] or by using error correction schemes [25].

In this paper, we review reconfigurable nanocomputing frameworks using nanodevices. In particular, we focus on two major computing paradigms that use nanoscale memory:

1) Purely spatial computing framework or *nanoFPGA*: Conventional FPGAs fall into this category. The architecture primarily consists of one-dimensional lookup tables (LUTs) for storing function response, which communicate via a programmable interconnect network with other LUTs. Realization of both LUT [21, 27] and interconnect network [14] using nanoscale devices have been widely investigated. We will discuss several nanoFPGA architectures and analyze the issue of defect tolerance in the context of nanoFPGA.
2) Time-multiplexed computing framework: Reconfigurable computing presented in [15] and [22] are examples of this computing model. The main idea is to perform the major share of the computation in local computing elements before communicating with other computing elements. Computation in each of the computing elements is performed over multiple cycles by storing function response in a large two-dimensional memory array. It can drastically minimize the overhead for programmable interconnects between the components and hence, improves performance. The dense and periodic structures of most emerging nanodevices as well as their bi-stable nature make them amenable to large high-density memory array design. High-density memory arrays are inherently more tolerable to defects as redundancy can be implemented more efficiently. The devices with high gain are required only for peripheral circuits and do not need to be integrated within the array. Hence, reconfigurable computing paradigms that can take advantage of high-density nanoscale memory to reduce the overhead of programmable interconnect, can be effective for improving integration density, performance and resource usage.

The rest of the paper is organized as follows. In section II, we discuss the use of an emerging nanoscale memory devices, such as STTRAM and NEMS for building conventional FPGA architectures. We also discuss a defect-aware mapping scheme for improving the yield. In section III, we discuss a time-multiplexed memory based computing (MBC) model with associated architecture which offers higher density and higher performance compared to a nanoFPGA model. We also describe a scheme for reducing the memory requirement in a MBC framework by using content addressable memory. Section IV discusses the promises and challenges associated with memory based architectures using emerging nanodevices.

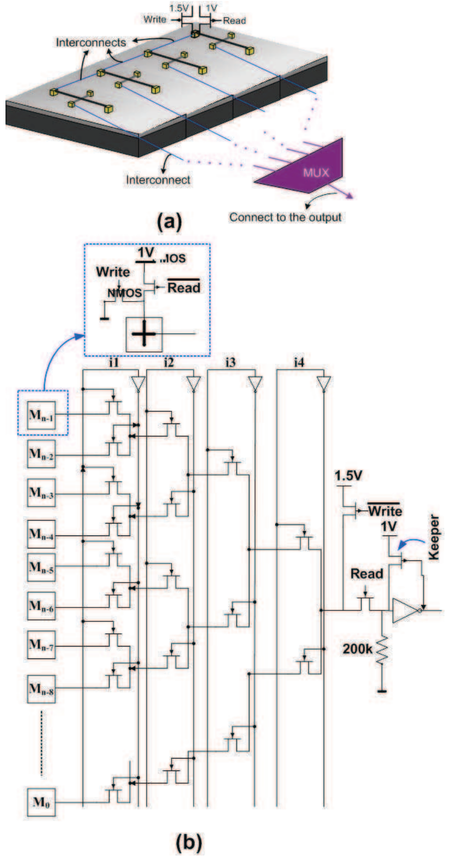

Fig. 1. a) 3-D view of a 2X1 NEMS-CMOS LUT.; b) Transistor-level schematic view of a NEMS-CMOS LUT. The decoding circuit is realized using a CMOS-based multiplexer-tree [27].

## II. nanoFPGA

Conventional FPGAs follow a purely spatial computing model, where functions are evaluated by spatially programmed connection of configurable processing elements. In such a model, each compute/connection resource is dedicated to single function. Researchers have explored possible extension of this model to the nano domain, through use of emerging nanodevices to realize the LUTs or connection blocks in a nanoFPGA framework. Some nanodevices which have been investigated to realize non-volatile FPGAs are: a) nanoelectromechanical switches (NEMS) [27], b) molecular diodes [16] and c) STTRAM [28-29]. The principal idea in all of these approaches is to realize non-volatile configuration bits of the FPGA LUTs using novel memory cell. Many of these nanodevices can be modeled as bistable resistors [23-30], where the high resistance state is used to represent a logic ‘0’ and a low resistance state to represent logic ‘1’ or vice versa. It is worth noting that the resistance states of these bistable devices are only configured at the time of application mapping to the FPGA. Since reconfigurable platforms typically undergo reconfiguration very infrequently, the power overhead associated with the high write current requirement for many of the nanodevices [30] may not be a major concern

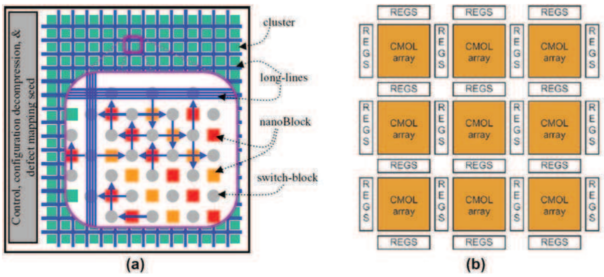

Fig. 2. a) Layout of nanofabric with a partial blowup of a single cluster and some of the adjacent long-lines [11]; b) A macro-array of CMOL FPGA arrays interleaved with CMOS registers [31].

in most applications. In [27], the authors have attempted to realize a LUT based conventional FPGA architecture using carbon nanotube (CNT) nanoelectromechanical switches. Fig. 1 shows the schematic of the CMOS-NEMS hybrid FPGA proposed in [27], which interfaces CMOS multiplexer with CNT-NEMS memory cells. Some nanodevices (e.g. molecular diodes) can be used to realize programmable logic arrays (PLAs) efficiently. In these cases, significant reduction can be achieved in the memory required to map a given application [16]. An alternate nanoFPGA framework resembling cell-based FPGA design was proposed in [21]. These two ideas are illustrated in Fig. 2. It can be noted that due to the high density of the nanodevices, small 1-D LUTs similar to CMOS FPGAs may not be efficient in terms of area and other design parameters, since the overhead for interfacing these small LUTs with CMOS circuitry can be significant [21]. Thus it has been proposed in [16], to employ dense 2-D memory array as LUTs or PLAs to construct a spatial computing framework for the nanodevices. Note that such memory based spatial computing model has also been investigated in the context of conventional SRAM based reconfigurable platforms for improving performance. [36].

### A. *STTRAM based FPGA design*

STTRAM is being considered as the next generation universal memory [30] due to its high speed of operation along with unlimited endurance and high integration density. In [28], researchers have attempted to replace the CMOS SRAM configuration bits with STTRAM magnetic tunneling junction (MTJ) devices. However, the dynamic sensing scheme for reading the LUT content as presented in [28] is inappropriate for a spatial computing framework such as FPGA. The reason as pointed out in [29] is that the time of evaluation of a particular configurable logic block (CLB) depends on the delay of the timing path on which the CLB lies. A more appropriate static sensing scheme was described in [29]. Fig. 3 shows the structure of the CLB in the non-volatile FPGA framework as presented in [29]. Application of efficient circuit/architecture/application mapping co-optimization techniques in CMOS-STTRAM hybrid FPGA can be effective to improve its area, delay and power over conventional CMOS FPGA [29]. This observation, coupled with its non-volatile property, makes STTRAM a very promising candidate for nanoFPGA design.

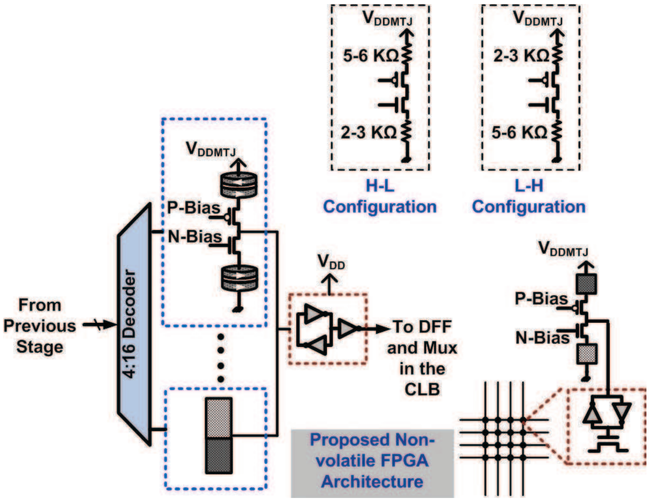

Fig. 3. Scheme for integrating STTRAM in the CLB and programmable interconnects in a CMOS-STTRAM hybrid FPGA architecture [29].

***Preferential mapping for power reduction:*** Preferential application mapping is a concept that exploits the asymmetry in MTJ state for power reduction in CMOS-STTRAM hybrid FPGA. As explained in [29], the read '1' scenario from the CLB consumes more power over read '0' in the CMOS-STTRAM hybrid FPGA design. Hence, during the application mapping process, the mapping tool attempts to maximize the number of logic '0's stored in the LUT. For standard benchmark circuits, such a mapping approach has been shown to achieve considerable reduction in the dynamic and static power per CLB.

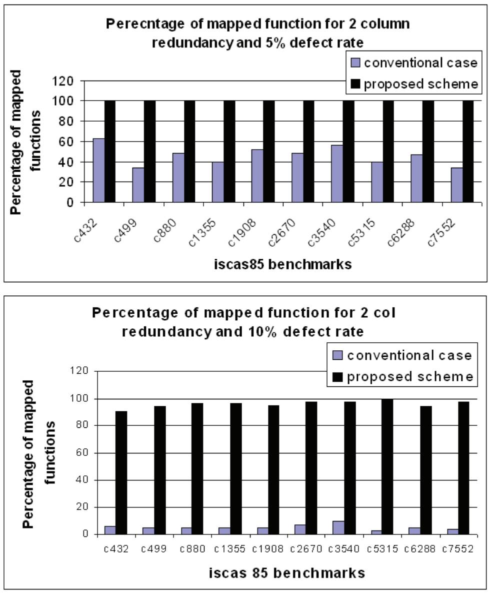

Fig. 4. Variation in percentage of mapped function with defect rate [24].

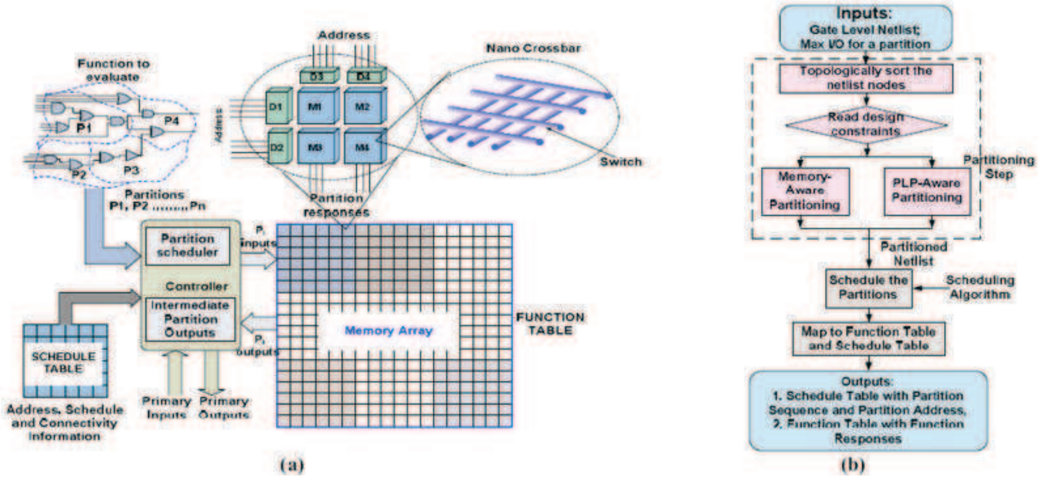

Fig. 5. a) Overall memory based computation scheme. A multi input/output logic function to be evaluated is partitioned and the partitions are stored into memory arrays (realized with nanoscale devices using a set of small memory modules). A controller performs the tasks of partition evaluation in topological order and handling of intermediate partition outputs; b) Design flow for MBC [15].

### B. *Defect tolerance in nanoFPGA designs*

Extensive research [32-35] has been performed to address defect tolerance in computing architectures for different nanoscale devices. Most of these investigations rely on a previously generated "defect map" which captures the location of defective devices. This defect map is then used to configure around the defective devices and map the target application to the functional devices. Such defect tolerance can be achieved for both logic blocks as well as interconnects in the nanoFPGA framework. A different approach, as presented in [24], takes advantage of the defects which can be modeled as permanent stuck-at faults. Memory cells with such defects can be used to realize logic '0' or '1' in a LUT. Many nanodevices are expected to have such defects. The proposed approach requires a defect map that specifies the locations with stuck-at defects. The possibility of using the defective memory cells to realize boolean logic function can improve the yield over techniques which map circuits to only working devices. Even for high defect rates, such a mapping technique has been shown to outperform redundancy based approaches. Fig. 4 shows that for high defect rate scenarios, the mapping procedure presented in [24] can significantly improve the yield over conventional redundancy based techniques. A major challenge in this approach is, however, to characterize defects efficiently and find irreversible stuck-at type ones during defect map generation.

## III. Memory based computing

In contrast to the purely spatial reconfigurable computing frameworks of a nanoFPGA, researchers have investigated an alternative temporal computing framework [15, 22]. Unlike nanoFPGA, where each compute resource is dedicated to single function, such a model reuses a resource over multiple cycles during evaluation. The primary component in the programmable computing element is a dense 2-D memory array. The target application is partitioned into multi-input multi-output LUTs which are then mapped to the memory arrays. We refer to such reconfigurable platforms as memory based computing (MBC) platforms. Fig. 5 illustrates the MBC framework proposed in [15] along with the design flow. Each computing element of the MBC platform has four major components: a) *Function Table* which consists of a dense 2-D memory array organized into multiple banks. Multi-input multi-output LUTs storing the responses for the target circuit are mapped to this 2-D memory array; b) intermediate registers which stores the intermediate partition outputs; c) local interconnect architecture which selects the values stored in the intermediate registers and forms the LUT address; c) *Schedule Table*, a small memory array which holds the partitions with their input-output address. The function table can be implemented with nanodevices while the other components form a CMOS controller that interfaces with the nanoscale memory array. The controller evaluates the partitions in topological fashion over multiple cycles. Output from one computing element is transmitted to another through a programmable interconnect framework similar to conventional FPGA.

The MBC model provides the following major advantages for the emerging nanodevices:

- Using dense 2-D memory array, MBC preserves the density advantage of nenodevices. CMOS interfacing logic in MBC is localized and separate, minimizing the requirement of interface hardware and potentially facilitating CMOS-nano hybridization process.
- Dense memory arrays facilitate large partitions to be mapped in the MBC framework. Larger partition size leads to higher performance and minimizes the requirement for programmable interconnects. As illustrated in Fig. 6, the reduction in global routing resources compared to conventional FPGA architecture is significant. Since interconnect performance does not scale as well as

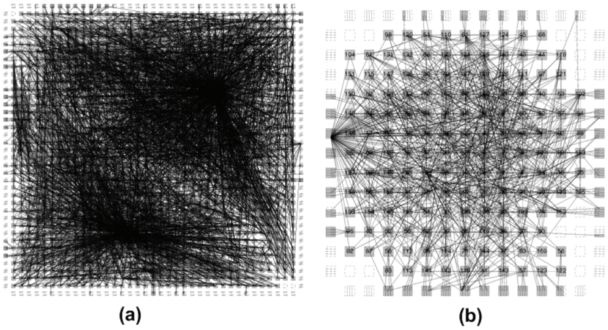

Fig. 6. Interconnect requirement for the sequential benchmark *s38417* when mapped to a) 65nm CMOS FPGA and b) MBC frameworks [37].

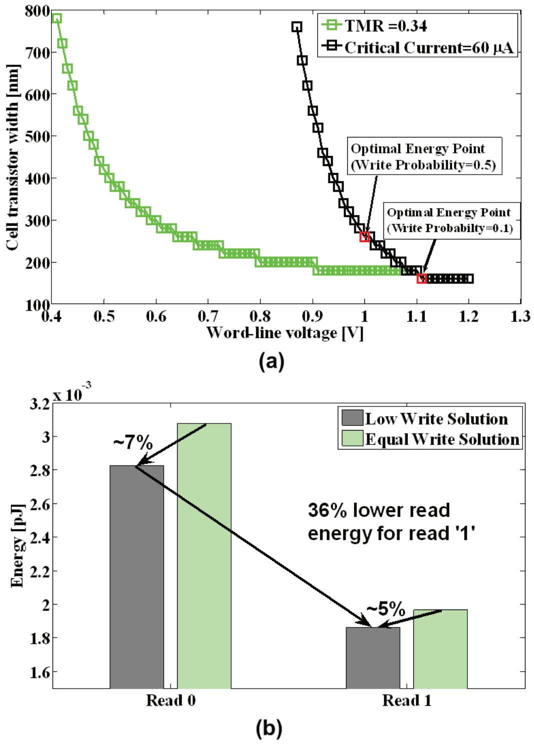

Fig. 7. a) Design of STTRAM cell for MBC framework to achieve optimal read energy; b) Read energy for a cell storing logic '0' and '1' [37].

logic/memory performance with technology, MBC leads to better technology scalability of performance compared to a completely spatial framework such as FPGA.

The advantages of the MBC framework over a nanoFPGA platform has been reported for two emerging nanodevices, namely molecular crossbar [15] and STTRAM [37]. With device models available from the literature, architecture level simulations were performed for standard benchmark circuits for both nanoFPGA and MBC frameworks. These simulations suggest that considerable improvement in performance and energy-delay product (EDP) can be achieved for a time-multiplexed computing model with the nanodevices configured as a dense memory array.

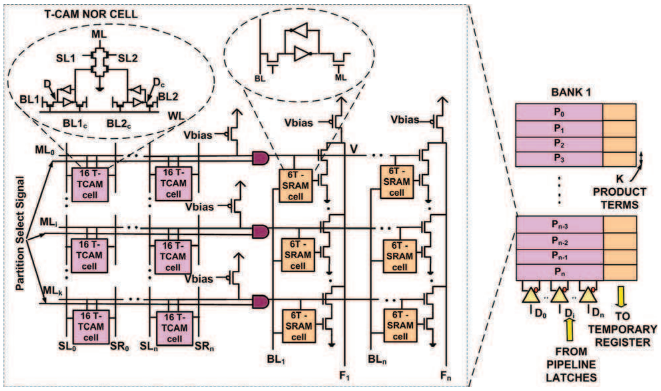

Fig. 8. TCAM based realization of the partitions in a MBC framework [19].

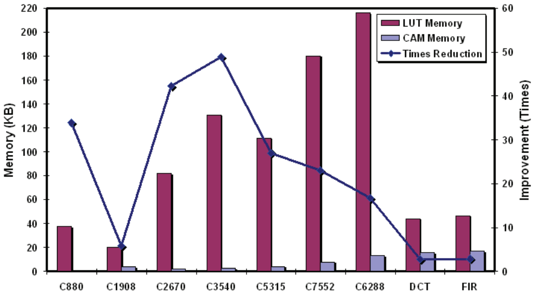

Fig. 9. Comparison of memory requirements between LUT and CAM based implementations [19].

## *A. Preferential Design for MBC*

Similar to nanoFPGA, a preferential mapping algorithm which exploits the asymmetry in logic '0' and logic '1' storage with respect to read energy can be used to improve the EDP in a MBC framework [37]. Circuit simulations show that a read '1' operation from the MTJ cell in a STTRAM memory dissipates lesser energy compared to a read '0' operation. Hence, LUT contents in the function table can be skewed to contain more logic '1' than logic '0'. Moreover, noting the read-dominant memory access pattern in the MBC framework, one can select the wordline voltage as well as size of the transistor in the MTJ cell that optimizes read performance and energy while affecting write operation, which are infrequent. Fig. 7 shows the circuit level optimizations which exploit the read dominated access pattern in the MBC framework to arrive at an energy efficient MTJ cell design. The asymmetry between read '0' and read '1' energy as shown in Fig. 7(b) is exploited by an appropriate application mapping technique proposed in [37].

## *B. Memory reduction in a MBC framework*

The concept of PLA-based logic evaluation can be exploited in MBC to reduce the memory requirement drastically [19]. The basic idea is to represent a function in the LUT in terms of its support or input space instead of output space. The function is evaluated for a given input by searching the input space and returning logic '1' on a match and logic

‘0’ otherwise. Hence, we need to have a content addressable memory (CAM) to store the function response. In order to store optimized representation of a function, we need to consider storing don’t care (‘x’) conditions e.g. a 4-input OR gate can be represented by four terms: 1xxx, x1xx, xx1x, xxx1 (compared to 16 in LUT-based representation). In order to store ‘x’ values, we can use Ternary-CAM (TCAM), as illustrated in Fig. 8. Fig. 9 shows the exponential savings in memory requirement achieved by this approach over a LUT-based implementation. Although the viability of such a framework was verified for SRAM, we note that the same concept can be extended to nanodevices, which are amenable to CAM design. The possibility for realizing CAM and TCAM using STTRAM has already been reported in [38]. Moreover, it is worth noting that the emerging nanodevices which hold promise for implementing 3 or multi-valued logic can further reduce the number of bits required to implement the CAM. For example, a 3-valued CAM can reduce the memory requirement as shown in Fig. 9 by half.

## IV. Discussion and Summary

Memory based computing models and architectures appear very promising for the emerging nanodevices. Both a purely spatial computing framework like conventional FPGA as well as a time-multiplexed computing platform can leverage on nanoscale memory structures. The principal advantages of using nanoscale memory for computing are: reconfigurability, non-volatility, defect tolerance and high integration density. Compared to nanoFPGA, time-multiplexed memory based computing that uses dense 2-D memory array can be more effective in preserving the high integration density of nanoscale memory and improving performance due to large reduction in programmable interconnect resources. Circuit/architecture/application mapping co-optimization approaches for both nanoFPGA and MBC that exploit the properties of both the memory technology as well as the computing model can significantly improve the energy-delay product. For example, exploiting the read-dominant access pattern in MBC, an STTRAM cell can be optimized for read energy, while affecting infrequent write operation. On the other hand, a preferential mapping policy that skews the LUT content towards ‘0’ over ‘1’ can considerably save energy in STTRAM FPGA. There are, however, number of challenges associated with memory based reconfigurable architectures for emerging nanodevices. They include deriving optimal memory organization for a nanodevice, minimizing CMOS-nano hybridization overhead and efficient defect map generation.

## References

[1] ITRS 2007: Process Integration, Devices and Structures, [Online] http://www.itrs.net/Links/2007ITRS/2007_Chapters/2007_PIDS.pdf.

[2] ITRS 2008 Update, [Online] http://www.itrs.net/Links/2008ITRS/Update/2008_Update.pdf.

[3] K. Kim and Y.J. Song, “Current and future high-density FRAM technology”, *Integrated Ferroelectrics* Vol.61, pp. 3-15, 2004.

[4] T.W. Andre, J.J. Nahas, C.K. Subramanian et al., “A 4-Mb 0.18-$\mu$m 1T1MTJ toggle MRAM with balanced three input sensing scheme and locally mirrored unidirectional write drivers”, *ISSCC*, 2004.

[5] W.Y. Cho, B.H. Cho, B.G. Choi et al, “A 0.18-$\mu$m 3.0-V 64-Mb nonvolatile phase-transition random access memory (PRAM)”, *EEE Journal of Solid-State Circuits*, Vol. 40, pp. 293-300, 2005.

[6] T. Rueckes et al. “Carbon nanotube-Based Nonvolatile Random Access Memory for Molecular Computing” *Science* 2000.

[7] W. Wu et al., “One-kilobit cross-bar molecular memory circuits at 30-nm half-pitch fabricated by nanoimprint lithography” *Appl. Phys* 2005.

[8] K. Uchida, “Programmable Single-Electron Transistor Logic for Low-Power Si-LSI”, *ISSCC* 2002.

[9] A. Batchtold et al., “Logic Circuits using Carbon Nanotube Transistors”, *Science*, Vol. 294, pp 1317-1320, 2001.

[10] M. Ottavi et al., “Design of a QCA Memory with Parallel Read/Serial Write”, *IEEE Computer Society Annual Symp. on VLSI*, 2005.

[11] S.C. Goldstein et al., “NanoFabrics: Spatial Computing Using Molecular Electronics”, *ISCA*, 2001.

[12] A. DeHon “Array-Based Architecture for FET-Based Nanoscale Electronics”, *IEEE Trans. Nanotechnol* Vol 2, No. 1, 2003.

[13] P.J. Kuekes, D.R. Stewart, R. S. Williams. “The Crossbar Latch: Logic Value Storage, Restoration and Inversion in Crossbar Circuits”, *J. Appl. Phys* Vol. 93, 2005.

[14] G.S. Snider and R.S. Williams, “Nano/CMOS architectures using a field-programmable nanowire interconnect”, *Nanotechnology*, 2007.

[15] S. Paul and S. Bhunia, “MBARC: A Scalable Memory Based Reconfigurable Computing Framework for Nanoscale Devices”, *ASPDAC*, 2008.

[16] M.M. Ziegler and M.R. Stan, “CMOS/Nano Co-Design for Crossbar-Based Molecular Electronic System”, *IEEE Trans. on Nanotechnology*, Vol. 2, 2003.

[17] [Online]: http://www.altera.com/products/devices/stratix3/st3-index.jsp

[18] C. He et al., “Scalable Defect Mapping and Configuration of Memory-Based Nanofabrics”, *Intl. Workshop HLDVT*, 2005.

[19] S. Paul and S. Bhunia, “Reconfigurable computing using content addressable memory for improved performance and resource usage”, *DAC*, 2008.

[20] A. Dehon et al., “Hybrid CMOS/nanoelectronic digital circuits: devices, architectures, and design automation”, *(ICCAD*, 2005.

[21] K.K. Likharev and D.B. Strukov, “Prospects for the Development of Digital CMOL Circuits”, *IEEE NanoArch*, 2007.

[22] D. Jones and D.M. Lewis, “A time-multiplexed FPGA architecture for logic evaluation”, *IEEE CICC*, 1995.

[23] M.R. Stan et al, “Molecular Electronics: From Devices and Interconnect to Circuits and Architecture”, *Proceedings of the IEEE*, Vol. 91, pp. 1940-1957, 2003.

[24] S. Paul et al., “Defect-Aware Configurable Computing in Nanoscale Crossbar for Improved Yield”, *IOLTS*, 2007.

[25] J.G. Brown et al., “CAEN-BIST: testing the nanofabric”, *ITC*, 2004.

[26] M.B. Tahoori, “A mapping algorithm for defect-tolerance of reconfigurable nano-architectures”, *ICCAD*, 2005.

[27] Y. Zhou et al., “Low power FPGA design using hybrid CMOS-NEMS approach”, *ISLPED*, 2007.

[28] W. Zhao et al., “Integration of Spin-RAM technology in FPGA circuits”, *ICSICT*, 2006.

[29] S. Paul et al., “Hybrid CMOS-STTRAM non-volatile FPGA: design challenges and optimization approaches”, *ICCAD*, 2008.

[30] M. Hosomi et al, “A Novel Nonvolatile Memory with Spin Torque Transfer Magnetization Switching: Spin-RAM”, *IEDM*, 2006.

[31] D.B. Strukov and K.K. Likharev, “CMOL FPGA: a reconfigurable architecture for hybrid digital circuits with two-terminal nanodevices”, *Nanotechnology*, 2005.

[32] A. Agarwal et al., “Fault tolerant placement and defect reconfiguration for nano-FPGAs”, *ICCAD*, 2008.

[33] M.B. Tahoori and S. Mitra, “Defect and Fault Tolerance of Reconfigurable Molecular Computing”, *FCCM*, 2004.

[34] M. Mishra and S.C. Goldstein, “Defect Tolerance at the End of the Roadmap”, *Proc. ITC*, 2003.

[35] Z. Wang and K. Chakarborty, “Built-in Self-test and Defect Tolerance in Molecular Electronics-based Nanofabrics”, *JETTA*, Vol. 23, 2007.

[36] S.J.E. Wilton, “Embedded Memory in FPGAs: Recent Research Results”, *IEEE Conf. on Comm., Computers and Signal Proc.*, 1999.

[37] S. Paul et al, “Nanoscale Reconfigurable Computing Using Non-Volatile 2-D STTRAM Array ”, *IEEE Conference on Nanotechnology*, 2009.

[38] W. Xu et al, “spin-transfer torque magnetoresistive content addressable memory (CAM) cell structure design with enhanced search noise margin”, *ISCAS*, 2008.