Towards Memristor-based Reconfigurable FFT Architecture

Khaled Alhaj Ali[1,4], Mostafa Rizk[3], Amer Baghdadi[1], Jean-Philippe Diguet[2], and Jalal Jomaah[4]

[1]IMT Atlantique, CNRS Lab-STICC, UBL, F-29238 Brest, France
[2]Lab-STICC CNRS UMR 6285, Université de Bretagne-Sud, Lorient, France.
[3]School of Engineering, Lebanese International University, Beirut, Lebanon.
[4]Physics Department, Faculty of Sciences, Lebanese University, Beirut, Lebanon.

***Abstract*—The emerging non-volatile memory technologies open novel perspectives for original highly-efficient and flexible architecture design. Dense integration of non-volatile memristors close to logic gates and circuit wires allows for new dimensions of reconfiguration and low power design. In this paper we present a review of the recent contributions that have investigated the use of memristors for different applications in digital design. Based on this review, and considering the widely used fast Fourier transform (FFT), we investigate for the first time a novel memristor-based reconfigurable FFT architecture. The proposed original architecture allows an efficient support of any combination of radix-2 and radix-3 butterflies. Scalability is ensured through a 2D mesh based topology and fast reconfiguration can be achieved using memristor-based interconnections.**

I. Introduction

The recent development of new non-volatile memory technologies based on the memristor concept has triggered many efforts to explore their potential usage in different application domains. This novel type of two terminal nano-scale elements presents very fast switching characteristics, non-volatile dense storage capacity, and low power consumption. Main part of conducted efforts aims to exploit them for establishing a unified and efficient memory system replacing current flash and CMOS-based memories. On the other hand, the possibility to integrate memristors on top of CMOS logic gates allows for new design ideas based on close combination and interaction between memory and computation. This introduces new opportunities of efficient reconfiguration, high performance, and low power design.

Designing flexible architectures, which can adapt dynamically to application needs, bring great advantages in terms of energy efficiency and performances. Such flexibility is required at processing, interconnect, and memory levels. Current technologies are inefficient for highly self-adaptive systems, due to the cost of reconfiguration, including delay and power consumption. Memory accesses, predominant in many applications, constitute a real bottleneck. Flexibility is particularly required in digital communication and multimedia applications where new standards and multiple service modes are continuously emerging, with strengthen requirements in terms of performance and energy efficiency. Fast Fourier transform (FFT) is one of the main and widely used components in such applications. Designing reconfigurable FFT architectures that can support efficiently different input data sizes associated with different radices are of high interest for these applications [1][2].

In this paper we present a review of the recent contributions that have investigated the use of memristors for different applications in digital design. Based on this review, we investigate a novel memristor-based reconfigurable FFT architecture (mrFFT). To the best of the authors' knowledge, this is the first memristor-based FFT design in the literature. The proposed original architecture allows the efficient support of many configurations with different FFT sizes. It is constituted of multiple mixed-radix butterfly (MBF) blocks which are arranged in a 2D mesh topology with memristor-based interconnections and a block containing the required first-in first-out FIFOs and multipliers. The interconnections are reconfigured by changing the states of the memristors through a dedicated controller.

The rest of the paper is organized as follows. Section II introduces the memristor concept and presents a review of the recent contributions that have investigated the use of memristors for different applications in digital design. Section III describes the proposed memristor-based reconfigurable FFT architecture. Finally, Section IV concludes the paper.

II. Memristor: concept and applications

The existence of the memristor, the fourth fundamental passive circuit element, was theoretically predicted in 1971 by Chua, but not experimentally validated until 2008 by HP Labs [3].

A. Background

A memristor is essentially a non-volatile nanoscale programmable resistor. It is memory resistor whose resistance, or memristance, is changed by applying a voltage across the device. Chua has defined the memristor as a previously missing relation between the flux Φ and the charge q and therefore yielding the defining relation: $M(q) = d\Phi/dq$ [4]. Memristor is characterized by its pinched hysteresis loop as the current-voltage (I-V) characteristic illustrated in Figure 1(a). The shape of the hysteresis loop depends on the amplitude and frequency of its input voltage [5]. This phenomenon shows that its instantaneous resistance can be modulated between two levels (R_{on}, R_{off}) where $R_{on} < R_{off}$ just by feeding it with a positive or negative bias. The bias value should be greater than a threshold voltage (V_{th}). The last resistance value is maintained even after it is power deprived.

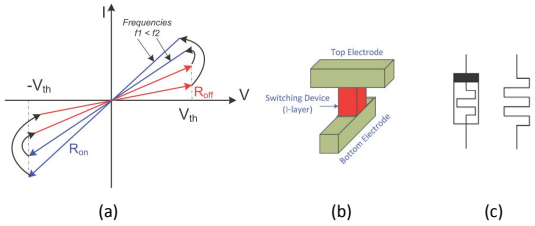

Fig. 1: Memristor: (a) Hysteresis loop, (b) Structure, (c) Symbol [5]

Therefore, memristors can be used to store binary data in the form of resistance which makes it a good candidate to non-volatile memories. On the other hand, its switching characteristics open new opportunities for the implementation of logic gates and neuromorphic synapses [6]. Figure 1(b) shows the structure of a typical memristor device consisting of two metallic electrodes that sandwich a thin dielectric insulating layer. Figure 1(c) depicts the two symbols typically used to denote a memristor, where the black square represents the positive terminal.

B. Memristors for memories

In general, any 2-terminal non-volatile memory (NVM) device based on resistance switching is called memristor [5]. Many technological variants are explored in this context. Resistive RAM (RRAM) is one of the most promising candidates for the next generation NVM. They show in fact an excellent scalability (below 10 nm), high switching speed (below 300 ps), large resistance ratio R_{off}/R_{on}, low power operability and long endurance. Figure 2(a) presents RRAM with a topology of crossbar array which is considered as a simple and dense structure. Writing into these arrays is simply done by selecting the horizontal and vertical nanowires corresponding to the target bit cell and applying the bias to write 0 or 1 corresponding to R_{off} and

978-1-5386-4049-4/17/$31.00 ©2017 IEEE

R_{on} , respectively. Reading can be achieved by sending a current to the target cell and measuring it with a dedicated sensing circuit which can then decide its binary state. However, in some cases, this can induce a leakage current named *sneak paths* as shown in Figure 2(b) that flows in the selected lines but through the unselected memristor. Many strategies are proposed to overcome this issue. In [7], the authors use the read-half-select method that can minimize this phenomenon.

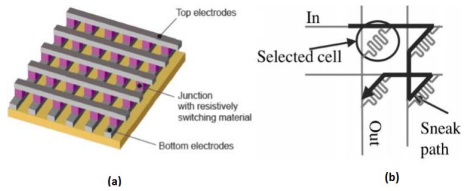

Fig. 2: RRAM: (a)Memristive crossbar,(b) sneak path phenomenon[8]

C. Memristors for logic gates

Several memristor-based logic designs have been proposed in the recent literature.

Memristor aided logic (MAGIC): MAGIC is a memristor-only logic family that supports boolean functions [9]. In each logic gate, memristors serve as inputs with previously stored data, and an additional memristor serves as an output. MAGIC can be applied in memristor-based crossbar in case of NOR gate [9]. Figure 3(a) shows the 2NOR1 MAGIC gate with input (In1,In2) and output (out) memristors.

Memristor ratioed logic (MRL): MRL is a hybrid CMOS-memristive logic family [10]. In this family, OR and AND logic gates are based on memristive devices, and CMOS inverters are added to provide a complete logic structure and signal restoration. Figure 3(b) illustrate an example of the NOR gate built with MRL design.

Memristor-based material implication IMPLY gate: In the IMPLY gate, each memristor is used as an input/output of the computational logic element [3]. As shown in Figure 3(c), the gate consists of two memristors (p, q), one resistor R_G, voltage sources V_{COND} and V_{SET}. In this case, $R_{ON} < R_G < R_{OFF}$ and $|V_{COND}| < |V_{SET}|$, the initial memristances of p and q represent the input to the gate, while the output is written into memristor q after applying V_{COND} and V_{SET} simultaneously. The truth table is shown in Figure 3(c) where $p \longmapsto q = p' + q$.

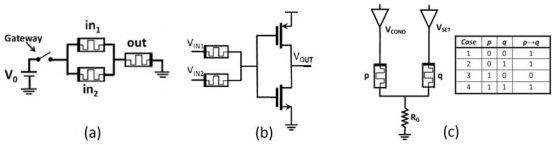

Fig. 3: Memristive logic gates: (a) MAGIC, (b) MRL, (c) IMPLY [5]

D. Application designs based on memristors

Several recent contributions have investigated the use of memristors for different applications in digital design. A summary of some relevant contributions is provided in this section.

Multistate register based on resistive RAM (MPR) [8]: Figure 4 provides a schematic of the MPR cell which is similar to a D-flip-flop.

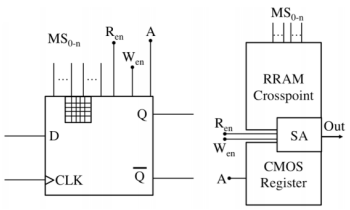

Fig. 4: Multistate register element (MPR [8])

The main idea is that MPR stores multiple data bits in an RRAM cross-point implemented on top of the CMOS latch. In this case, one bit is active in the MPR while the other bits are idle. Any of these bits can be activated in one clock cycle after MPR is enabled in the reading mode.

Memristor-based multithreading [11]: Based on MPR, a novel micro-architecture of a continuous flow multithreading (CFMT) is proposed in the literature. This architecture can be used for processors that run multiple threads on a pipeline unit as illustrated in Figure 5. In CFMT, an MPR holds the states of the different threads within the execution pipeline stages, where only one thread is active at a time. In this case, switching penalty like "flushing delays" are avoided. Moreover, it is possible to completely disable the non-volatile memristor layer to save the power related to idle threads.

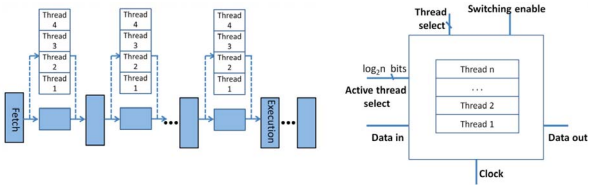

Fig. 5: Continuous flow multithreading (CFMT) [11]

Memristor-based Look-Up Table (LUT) for low energy operation of FPGA [12]: Memristor-based LUT for FPGA has been proposed as shown in Figure 6 in which the memristors are connected in separate columns. Therefore, the prevalent problems associated with nano-crossbars (such as the write half-select and the sneak paths) are not encountered. Simulation results in [12] show reduced power consumption for read and write operations and reduced delay compared to SRAM LUTs. On the other hand, memristor-based LUTs can be turned OFF completely during standby mode.

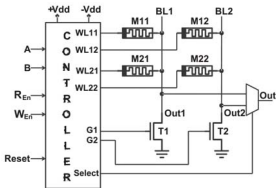

Fig. 6: Two-input LUT block based on memristors [12]

FPGA architecture with memristor-based reconfiguration [13]: The switching characteristics of memristors allow to build programmable interconnects. The mrFPGA topology proposed in [13] is shown in Figure 7 where the switching blocks (SB) and connection blocks (CB) are fabricated above the logic blocks (LB) saving in this way 5.18× area, 1.63× power, and 2.28× speedup.

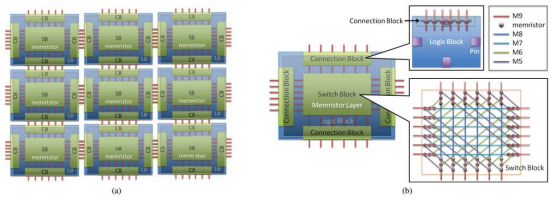

Fig. 7: FPGA architecture with memristor-based reconfiguration [13]

Memristive Memory Processing Unit (MPU) [14]: In-memory processing concept is proposed to reduce the data transfer between memory and processor, so reducing the effect of the "memory wall" and consequently increasing performance. In this approach, a memristive-based MPU is proposed for in-memory processing based on the above presented MAGIC NOR logic. The proposed MPU add processing capability to the RRAM based on MAGIC NOR logic. It can dynamically change the function of the memory between processing and storage, as illustrated in Figure 8

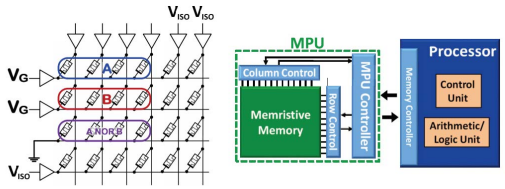

Fig. 8: In-memory computation using MAGIC NOR [14]

Based on this review of memristor-based design applications in the literature, we propose a novel memristor-based reconfigurable FFT architecture that will be detailed in the next section.

III. Proposed mrFFT design

Fast Fourier transform is a widely used and popular component in digital communication and multimedia applications. In case of a power of 2 FFT size, radix-2 butterfly is typically used [15]. Different radix butterflies are also proposed either to support different FFT sizes (not power of 2) and/or to reduce the computational complexity. If the application requires to have different radices, typical inefficient implementations duplicate the hardware resources.

We consider in this paper the recent contribution proposed in [1] which defines a flexible FFT algorithm to support 48 different configurations with an FFT size up to 2187. This flexibility is achieved through a reconfigurable processing element that implements six combinations of radix-2 and radix-3 butterflies (radices 2, 2^2, 2^3, 3, 3^2, 2×3).

Based on this algorithm and flexibility requirements, we propose a novel memristor-based reconfigurable FFT architecture. A mixed-radix butterfly (MBF) is designed to support radix-2 and radix-3 butterflies. The efficient pipelined single-path delay feedback (SDF) implementation of the FFT is adopted. The FFT size N determines the number and type (radix) of the pipelined FFT stages to be concatenated. The proposed architecture is presented below.

A. Mixed-radix butterfly: MBF

In fact, the radix-3 butterfly can be decomposed into 3 butterflies and one real multiplier k as shown in Figure 9(b). This decomposition allows the proposal of the mixed-radix butterfly as illustrated in Figure 9(c).

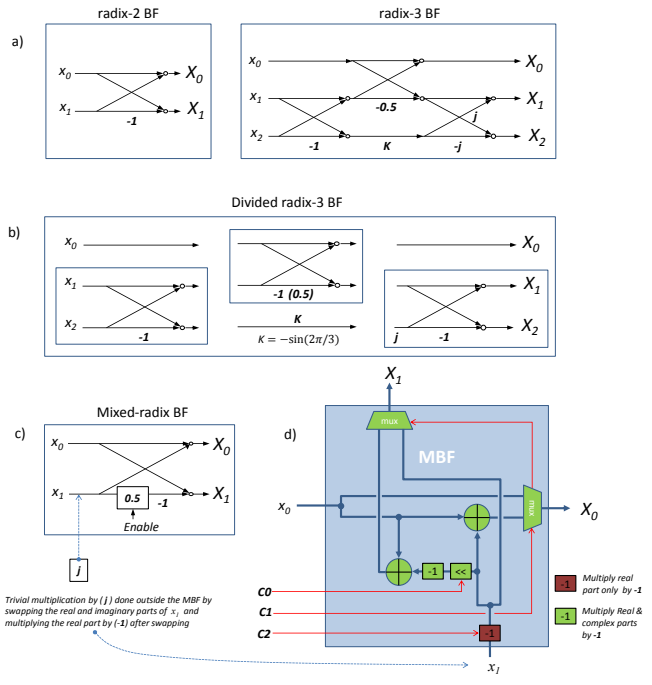

Fig. 9: Mixed-radix butterfly

The MBF can be configured to execute a radix-2 butterfly or any part of the radix-3 butterfly. The trivial multiplication by j is done outside the MBF by simply swapping the real and imaginary parts of the corresponding input. Figure 9(d) describes the associated hardware implementation of MBF.

B. mrFFT hardware architecture

Supporting radix-2 and radix-3 butterflies implies an FFT size N with prime factorization $N = 2^p 3^q$. Considering the example of $N = 864 = 2^5 3^3$. N can be rewritten in the form $N_{864} = (2^2 2^2 2)(3^2 3)$. This means that the FFT needs: 2 stages of radix-2^2, 1 stage of radix-3^2, 1 stage of radix-2 and 1 stage of radix-3. However, considering all possible cases for FFT sizes less than a predefined value N_{MAX} requires a highly flexible system that can concatenate any number and type of FFT stages. In order to address this issue, we propose the novel architecture illustrated in Figure 10. In this architecture, the MBFs are arranged in a 2D mesh topology with memristor-based interconnections. The interconnections are reconfigured by changing the states of the memristors connecting: (1) each horizontal wire (HW) with vertical one (VW), (2) each HW or VW with an MBF, and (3) each HW or VW with the block containing FIFOs and multipliers (FM).

Changing the states of these memristors is the role of the selection blocks (SB). SB act like writing into a conventional memristive crossbar array. They select a single memristor by setting the ground voltage V_G at one end and V^+ or V^- at the other end. SB blocks are controlled by a dedicated controller shown in the upper right corner of the architecture.

In the design shown in Figure 10, the number of the MBFs needed to support an FFT of size $N = 2^p 3^q$ is $N_{MBF} = p + 3q$. This is due to the fact that 3 MBFs are needed for each radix-3 butterfly. Therefore, for an FFT with $N_{MAX} = 2187$, the proposed architecture requires $N_{MBF} = 0 + 3(7) = 21$ MBF blocks. In this case, the 48 supported FFT configurations are listed in boldface in Table I. On the other hand, the number of additional wires needed between blocks depends on the number of feedback FIFOs and multipliers used.

C. FIFOs and multipliers

For an FFT of size $N = 2^p 3^q$, FIFOs' sizes must be in the form of 2^p and 3^q only. The number of allocated FIFOs satisfies the relation $N_F = p + 2q$ which attains its maximum at $N_F = 14$ for the 48 configurations considered in Table I. In these 14 FIFOs, the maximum required size corresponds to $2^{11}/2 = 1024$. Analyzing the reuse opportunities over the 48 FFT configurations, and considering that two FIFOs are required for each radix-3 butterfly, one of the two copies needed of the proposed FIFO sizes are listed in Table II.

TABLE I: Supported 48 FFT configurations with $N = 2^p 3^q$

p \ q	0	1	2	3	4	5	6	7
0	**1**	**3**	**9**	**27**	**81**	**243**	**729**	**2187**
1	**2**	**6**	**18**	**54**	**162**	**486**	**1458**	4374
2	**4**	**12**	**36**	**108**	**324**	**972**	2916	8748
3	**8**	**24**	**72**	**216**	**648**	**1944**	5832	17496
4	**16**	**48**	**144**	**432**	**1296**	3888	11664	34992
5	**32**	**96**	**288**	**864**	2592	7776	23328	69984
6	**64**	**192**	**576**	**1728**	5184	15552	46656	139968
7	**128**	**384**	**1152**	3456	10368	31104	93312	279936
8	**256**	**768**	2304	6912	20736	62208	186624	559872
9	**512**	**1536**	4608	13824	41472	124416	373248	1119744
10	**1024**	3072	9216	27648	82944	248832	746496	2239488
11	**2048**	6144	18432	55296	165888	497664	1492992	4478976

TABLE II: Optimized sizes of the 2 x 7 required FIFOs

FIFO Size	16	32	81	128	256	729	1024

TABLE III: Number of required complex multipliers

	Complex multiplication by k	Complex multipliers
N_{2048}	0	6
N_{2187}	7	7

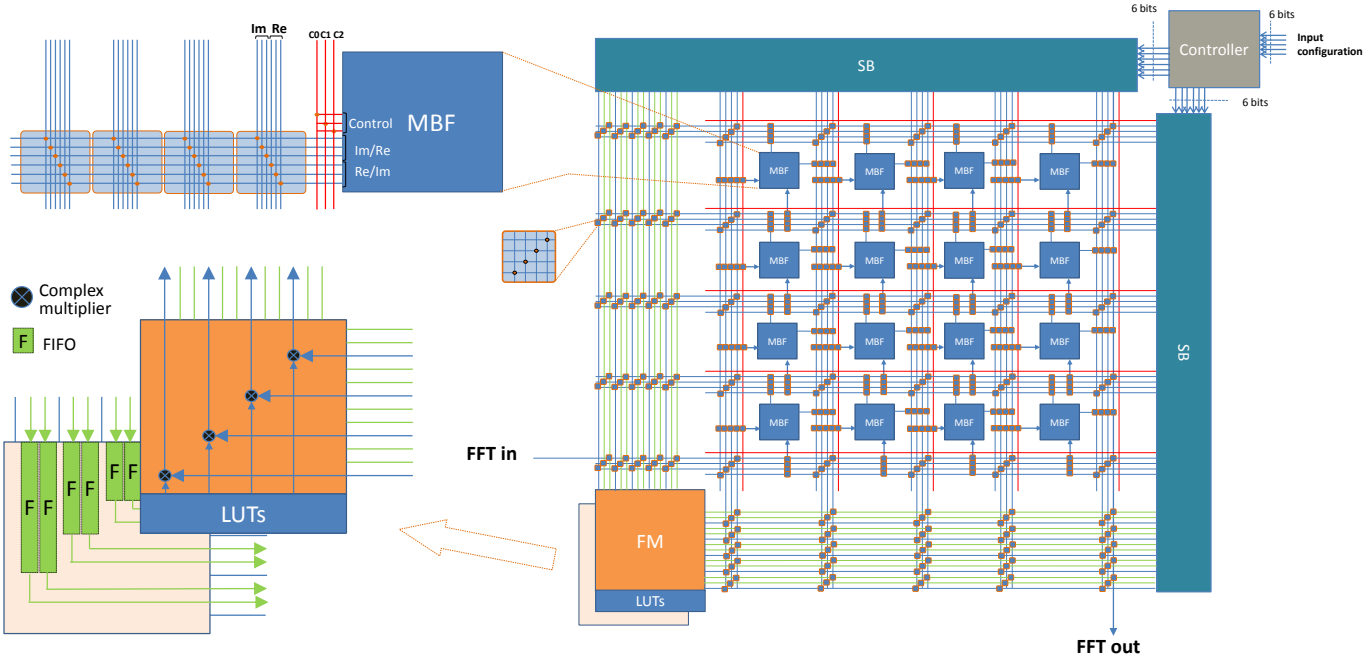

Fig. 10: Proposed mrFFT architecture

As for the multipliers, it is noticed that only one complex multiplier is required by each pipelined FFT stage whatever the type of the used radix [1]. On the other side, multipliers by k are only utilized in case of radix-3 stages. By considering the worst cases at $N_{2048} = 2^{11}$ and $N_{2187} = 3^7$, the results in Table III show that 14 complex multipliers are enough for all of the 48 supported FFT configurations including the multiplication by k (Figure 9). Figure 10 illustrates, with a simplified example, the distribution of the multipliers and FIFOs in the FM block. Regarding twiddle factors which are used in the multiplication operations, they are stored in a Look-Up-Table (LUT) in the FM block.

The proposed original architecture allows for efficient reconfiguration and hardware usage. It is fully pipelined and executes one sample per clock cycle, as conventional SDF based FFT implementations. However, it allows mixing any combination of radix-2 and radix-3 butterflies. Switching from one FFT configuration to another is determined by the size of the scalable 2D mesh topology. In fact, only one clock cycle is required to reconfigure a complete row of MBF blocks. Furthermore, the non-volatile memristor-based switching characteristics imply reduced power consumption.

IV. Conclusion

In this paper, we presented a review of recent state-of-the-art contributions that have investigated the use of memristors for different applications in digital design. Based on this review, a novel memristor-based reconfigurable FFT architecture is proposed for the first time. The proposed scalable architecture enables the efficient support of many configurations with different FFT sizes. The MBF blocks integrated in a 2D mesh topology with memristor-based interconnections allow for original and optimized hardware reuse. Fast reconfiguration is achieved as only one clock cycle is required to reconfigure a complete row of MBF blocks. Future work will consider the hardware implementation of the proposed architecture and the evaluation of energy savings.

References

[1] X.-Y. Shih, Y.-Q. Liu, and H.-R. Chou, "48-Mode Reconfigurable Design of SDF FFT Hardware Architecture Using Radix-3 2 and Radix-2 3 Design Approaches," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 64, no. 6, pp. 1456–1467, 2017.

[2] J. Nadal, C. Abdel Nour, A. Baghdadi, and H. Lin, "Hardware prototyping of FBMC/OQAM baseband for 5G mobile communication systems," in *IEEE International Symposium on Rapid System Prototyping (RSP)*, New Delhi, India, Oct. 2014, pp. 135 – 141.

[3] S. Kvatinsky, G. Satat, N. Wald, E. G. Friedman, A. Kolodny, and U. C. Weiser, "Memristor-based material implication (IMPLY) logic: Design principles and methodologies," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 22, no. 10, pp. 2054–2066, 2014.

[4] L. Chua, "Memristor-the missing circuit element," *IEEE Transactions on circuit theory*, vol. 18, no. 5, pp. 507–519, 1971.

[5] S. Hamdioui, S. Kvatinsky, G. Cauwenberghs, L. Xie, N. Wald, S. Joshi, H. M. Elsayed, H. Corporaal, and K. Bertels, "Memristor for computing: Myth or reality?" in *Proceedings of the IEEE/ACM Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2017, pp. 722–731.

[6] S. P. Adhikari, C. Yang, H. Kim, and L. O. Chua, "Memristor bridge synapse-based neural network and its learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 23, no. 9, pp. 1426–1435, 2012.

[7] M. S. Qureshi, M. Pickett, F. Miao, and J. P. Strachan, "CMOS interface circuits for reading and writing memristor crossbar array," in *Proceedings of the IEEE international symposium on Circuits and systems (ISCAS)*, 2011, pp. 2954–2957.

[8] R. Patel, S. Kvatinsky, E. G. Friedman, and A. Kolodny, "Multistate register based on resistive RAM," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 23, no. 9, pp. 1750–1759, 2015.

[9] S. Kvatinsky, D. Belousov, S. Liman, G. Satat, N. Wald, E. G. Friedman, A. Kolodny, and U. C. Weiser, "MAGIC - Memristor-aided logic," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 61, no. 11, pp. 895–899, 2014.

[10] S. Kvatinsky, N. Wald, G. Satat, A. Kolodny, U. C. Weiser, and E. G. Friedman, "MRL - memristor ratioed logic," in *Proceedings of the 13th International Workshop on Cellular Nanoscale Networks and Their Applications (CNNA)*, 2012, pp. 1–6.

[11] S. Kvatinsky, Y. H. Nacson, Y. Etsion, E. G. Friedman, A. Kolodny, and U. C. Weiser, "Memristor-Based Multithreading." *Computer Architecture Letters*, vol. 13, no. 1, pp. 41–44, 2014.

[12] T. N. Kumar, H. A. Almurib, and F. Lombardi, "Design of a memristor-based look-up table (LUT) for low-energy operation of FPGAs," *Integration, the VLSI Journal*, vol. 55, pp. 1–11, 2016.

[13] J. Cong and B. Xiao, "mrFPGA: A novel FPGA architecture with memristor-based reconfiguration," in *Proceedings of the IEEE/ACM International Symposium on Nanoscale Architectures*, 2011, pp. 1–8.

[14] R. B. Hur and S. Kvatinsky, "Memristive memory processing unit (MPU) controller for in-memory processing," in *Proceedings of the IEEE International Conference on the Science of Electrical Engineering (ICSEE)*, 2016, pp. 1–5.

[15] S. He and M. Torkelson, "A new approach to pipeline FFT processor," in *Proceedings of the 10th International Parallel Processing Symposium (IPPS)*, 1996, pp. 766–770.