Proceedings of the 11th World Congress on Electrical Engineering and Computer Systems and Sciences (EECSS'25)
Paris, France - August, 2025
Paper No. EEE 126
DOI: 10.11159/eee25.126

A Fully Programmable Memristor-based Processing-in-Memory Architecture

Raqibul Hasan
Center for Computational and Data Sciences
Independent University, Bangladesh
Dhaka, Bangladesh
raqibul.hasan@iub.edu.bd

Abstract **-** Power consumption and device variability are becoming critical challenges in processor design. Memristor-based architectures are emerging as a promising alternative to traditional CMOS technology. Specialized architectures offer greater energy efficiency than general-purpose systems. However, designing application-specific computing systems can be costly. Reconfigurable architectures present a viable solution by delivering high performance and low power consumption at a more reasonable cost. This paper proposes a reconfigurable architecture based on memristor-crossbar tiles. In a tile, there is on average one-quarter of a transistor per memristor for storage, which gives a significant area benefit compared to a 1T1M crossbar tile. A single tile array could be configured for any of the following operations: memory, search, neural network, and Boolean function implementation. For these operations, we leverage the computation in memory feature of the memristor crossbar.

Keywords**:** memristor crossbar, processing-in-memory, memory, neural network, search, Boolean function

1. Introduction

Big data and artificial intelligence (AI) will be the dominating applications in the next decade. Numerous systems—such as surveillance, autonomous vehicles, and pattern recognition in camera—rely heavily on image processing tasks. Neural networks are extensively employed for pattern recognition, signal processing, and image processing applications [1–3]. In big data applications, content search is one of the dominant operations.

General-purpose computing systems are designed to support a wide range of applications. However, the extensive flexibility they offer often comes at the cost of reduced energy efficiency. Power consumption and device variability have become critical concerns in modern processor design [5]. In response, researchers are exploring various strategies to develop more energy-efficient processors. These efforts include architectures for approximate computing that leverage techniques such as dynamic voltage scaling, dynamic precision control, and inexact hardware implementations [6,7]. Additionally, application-specific architectures have been proposed for targeted domains like signal and video processing [7]. Despite their performance and power efficiency benefits, designing such specialized systems can be costly. Reconfigurable architectures offer a compelling alternative, combining high performance, low power consumption with cost-effectiveness.

The concept of the memristor was first introduced in 1971 by Dr. Leon Chua [8]. Since then, various research teams have demonstrated memristive behaviour using different materials. A TiO_2 based memristor device published in [9], exhibits a high on-state resistance ($R_{ON} \approx 125$ kΩ) and a large resistance ratio ($R_{OFF}/R_{ON} \approx 1000$). The resistance state of a memristor device can be changed by applying a voltage greater than a threshold across the device [9,10,11]. The specific device presented in [9] has a threshold voltage of approximately 4V. Memristors can be arranged in a dense grid structure known as a crossbar, with the schematic and layout illustrated in Fig. 1.

The memristor crossbar structure enables highly efficient parallel evaluation of multiply-add operations in the analog domain. A memristor crossbar has the sneak path current problem which makes accessing a single memristor challenging. Yue et al. proposed a reconfigurable architecture based on 1T1R (one transistor, one RRAM) crossbars for memory and computation purposes [12]. A 1T1R crossbar has a significant area overhead compared to a 0T1M (0 transistor, 1 memristor) crossbar.

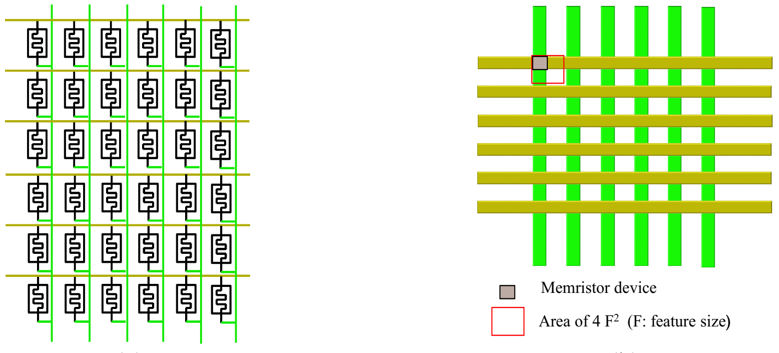

Fig. 1: (a) Memristor crossbar schematic and (b) memristor crossbar layout.

This paper presents a reconfigurable architecture based on memristor crossbar tiles. To minimize sneak path currents, we employ 8×8 memristor crossbar arrays, utilizing the memristor device described in [9]. In a tile, there ison average ¼ transistor per memristor for storage. A single tile could be configured for any of the following operations: memory, search, neural network, and Boolean function implementation.

In traditional computing systems, data transfer accounts for a substantial portion of overall power consumption. Studies on various multicore systems have reported that 30% to 40% of system power is consumed by the on-chip network [13]. In the proposed system, data is processed near its physical location. Consequently, the proposed memristor-based architecture can achieve a significant reduction in data transfer energy.

The rest of the paper is organized as follows: Section 2 describes the related works in the area. Section 3 describes the proposed processing-in-memory architecture. Sections 4, 5, 6, and 7 describe the memory, neuron, search, and Boolean function implementation in the proposed architecture respectively. Finally, in Section 8, we conclude our work.

2. Related Works

Sun et al. designed a 1T1M cell-based memory array and proposed a parallel testing method for the memory system [14]. Yue et al. proposed a reconfigurable architecture based on 1T1R (one transistor, one RRAM) crossbars for memory and computation purposes [12]. Yakopcic et al. designed a memristor-based cache memory and evaluated its system level impact [15].

Several studies examined memristor-based synapse, neuron, and neural network designs. Zhang et al. [16] designed a hybrid spiking neuron circuit combining memristor, and CMOS devices. They demonstrated a fully hardware execution of a spiking neural network based on the hybrid neuron circuit. Alibart et al. demonstrated ex-situ and in-situ training of memristor crossbar-based linear classifiers using the perceptron learning rule [18]. They have not examined the training of non-linearly separable problems and have not provided details of the ex-situ and in-situ training circuits. IBM designed and fabricated a 64-core processing in-memory chip based on phase-change memory device for the inference operation of deep neural networks [17]. They utilized ex-situ training and ADC circuit in the system.

Soudry et al. [19] proposed in-situ training of memristor crossbar neural networks based on the gradient descent learning rule. They implemented a synapse using two transistors and one memristor. Work in [20] designed an on-chip training system for multi-layer neural networks based on the back-propagation learning algorithm. They implemented the synaptic weights in memristor crossbar arrays and used ADC, DAC circuits for the training operations.

Kim et al. designed a CMOS circuit for a ternary content addressable memory [21]. Work in [22] demonstrated a TCAM cell that utilized 1 transistor and 1 RRAM for each bit-cell. Works in [23] demonstrated a memristor-based content search circuit using approximately 4 memristors for each bit-cell. Xiao et al. designed an STT-RAM based nonvolatile lookup table (LUT) for Boolean function implementation [24]. Work in [25] proposed a design methodology for logic circuit implementation using memristor crossbars. Their methodology optimized the design of logic functions and automatically mapped them onto the memristor crossbar.

3. Proposed Architecture

Fig. 2 shows the overall architecture of the proposed memristor tile array based reconfigurable system. The memristor tile arrays are connected through static routing elements. A configurable element in the system has a memristor crossbar array, input buffer, output buffer, and control logic circuits. Fig. 3 shows the block diagram of a memristor based tile array. The control logic generates appropriate control signals (e.g., the pass transistor control signal in Fig. 6) for a desired operation. A single tile could be configured for any of the following operations: memory, search, neural network, and Boolean function implementation based on the application requirement.

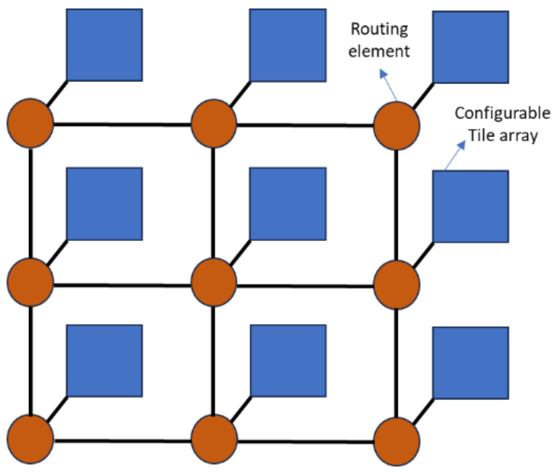

Fig. 2: Proposed memristor tile array-based reconfigurable multi-purpose architecture.

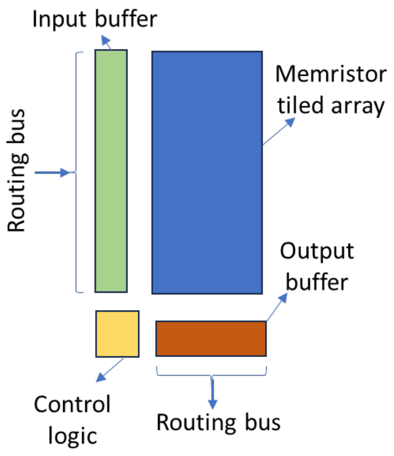

Fig. 3: Memristor crossbar based tile array along with input buffer, output buffer, and control circuit block.

SRAM-based static routing is utilized to facilitate re-programmability in the switches [26]. Fig. 4 shows the routing switch design. Fig. 5 shows the block diagram of a memristor crossbar-based tile array, while Fig. 6 shows the schematic of the tile array. Each memristor tile is consisting of an 8×8 memristor crossbar and pass transistors connected to the rows and columns. A tile array is consisting of 4 memristor tiles. Configuring the pass transistors of the tile, we can dynamically make the size of the crossbar as 8×8, 16×8, 24×8, or 32×8.

The main reason behind this tiled design is to limit the sneak path current for the memory operation. A large memristor crossbar produces more sneak path currents, which introduces error in the memory read operation. The comparators at the bottom of the tile arrays work as sensing circuits that provide output of the function implemented in the tile array. In an 8×8 memristor tile, we have 64 memristors and 16 transistors for isolation. Hence, on average, we have ¼ transistor for each memristor in a tile.

4. Memory Operation

The proposed system uses a memristor in the crossbar to store a binary bit. The memory operation in the memristor-based tiled array is similar to the technique used in [15]. The memristor conductance value σ_{on} represents bit 1, while the conductance value σ_{off} represents bit 0. Sneak path currents in a memristor crossbar may produce error in the read operation. Sneak path current problem becomes worst when all the memristors in the crossbar are in the σ_{on} state. We used 8x8 memristor tile arrays to implement a memory. For memory read operation, one tile is accessed at a time from a tile array, selecting the appropriate pass transistors. This limits the sneak-path currents and makes sure that the data is read correctly. In Fig. 6, we use the value of the R_s resistor of 0.4M ohms to increase the read noise margin. In an 8×8 memristor crossbar, the worst case read voltage for data bit 1 is 110.15mV (voltage drop across R_s in Fig. 6). While the worst case read voltage for data bit 0 is

46.09mV. These voltage levels demonstrate that we can read data from an 8×8 memristor array without any error. For the memristor device published in [9], a 16×16 tile was producing read errors.

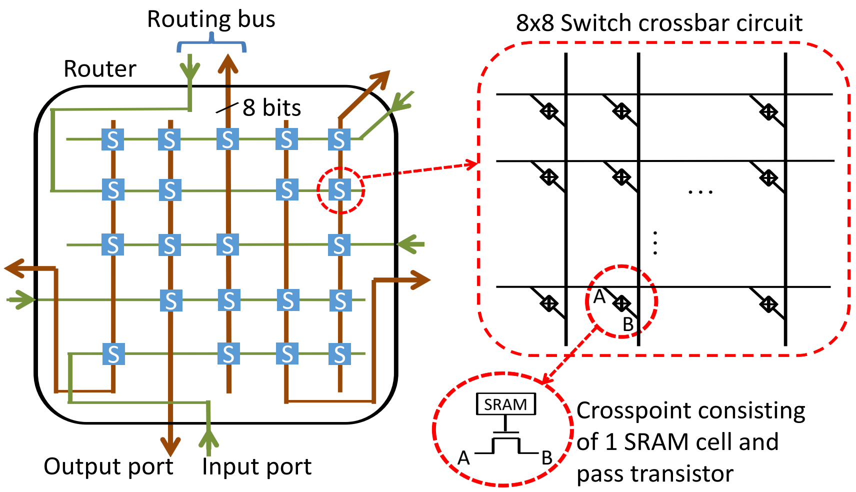
Routing bus
Router
8 bits
S
Output port
Input port
8x8 Switch crossbar circuit
A
B
SRAM
A
B
Crosspoint consisting of 1 SRAM cell and pass transistor

Fig. 4: SRAM-based static routing switch. Each blue square (S) in the left part of the figure represents the 8x8 SRAM based switch shown in the right (assuming an 8-bit network bus).

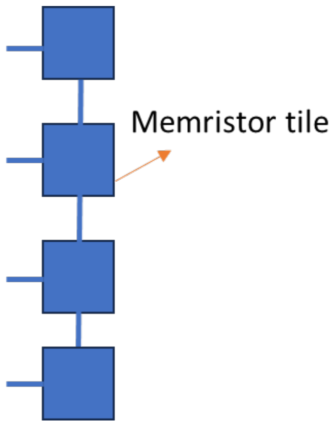

Fig. 5: The block diagram of a memristor crossbar-based tile array.

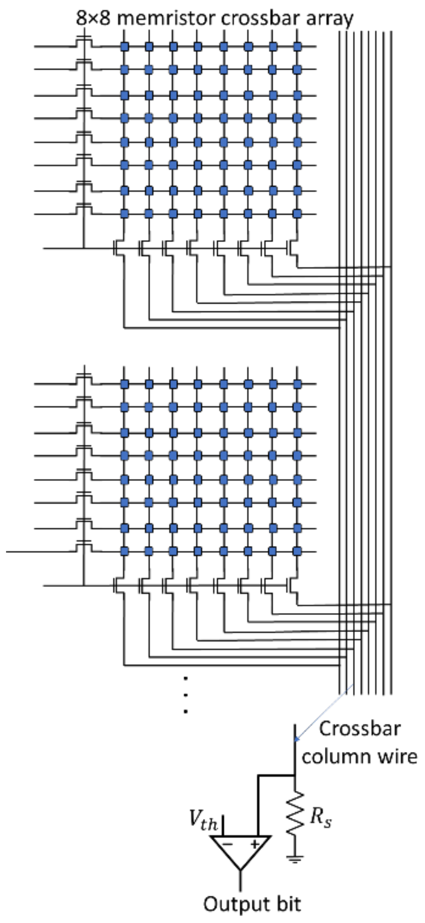

Fig. 6: The schematic of a memristor crossbar-based tile array.

In the proposed system, for memory implementation, a tile array stores 1 block of data, which is 32 byte. A memory address is divided into three parts: block number, tile number, and tile offset. To access data from a memristor tile based memory, the block address is decoded using memristor tile circuits. This task is similar to the implementation of a Boolean function. The implementation of a Boolean function on memristor tile array is explained in Section 7. Based on the tile number and tile offset, data from the appropriate tile row is accessed.

5. Neuron Operation

Neural networks are widely used for classification applications. Neurons are the building blocks of a neural network. A neuron performs the following two operations: i) dot product operation on its inputs and weights, and evaluation of a nonlinear function on the dot product. Fig. 7 shows a memristor based neuron circuit. For each input, both the data and its complemented form are applied to the neuron circuit. In this design, logic low is represented by -1V, and logic high is represented by 1V. The bottom of the crossbar column is connected to the gate input of the comparator. Note that no current flows through the gate of a MOS transistor. Applying Kirchhoff's current law, the potential at the bottom of the crossbar column is expressed by Eq. (1). In this neuron circuit, the synaptic weight corresponding to the input x_1 is (σ_{1a} - σ_{1b}). The output of the comparator at the bottom of the circuit represents the neuron output. This neuron circuit implements a threshold activation function.

$$V_{col} = \frac{x_1(\sigma_{1,a} - \sigma_{1,b}) + x_2(\sigma_{2,a} - \sigma_{2,b}) + \dots + x_n(\sigma_{n,a} - \sigma_{n,b})}{\sigma_{1,a} + \sigma_{1,b} + \sigma_{2,a} + \sigma_{2,b} + \dots + \sigma_{n,a} + \sigma_{n,b}} \qquad (1)$$

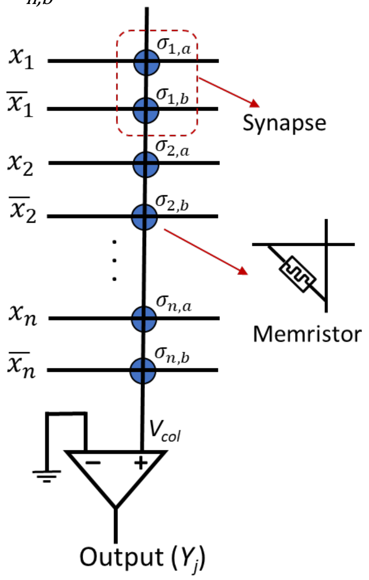

Fig. 7: Memristor-based neuron circuit. $x_1, x_2, \dots, x_n$ are the inputs and y_j is the output.

A neural network must be trained before it can be utilized. In ex-situ training [27], this process is carried out in software, and the final trained weight values are then programmed into physical memristor crossbars. This method eliminates the need for on-chip training circuitry, thereby reducing hardware complexity. Programming a memristor within a crossbar to either its R_{ON} or R_{OFF} resistance state is relatively straightforward—it involves applying a voltage of the correct polarity across the device for a sufficient duration, without requiring iterative write cycles or intermediate read operations during programming [32]. When a neural network is trained using only three discrete weight levels, it is referred to as a Ternary Neural Network (TNN). The study in [33] demonstrated a memristor-based neural network utilizing ternary weights. The work in [28] showed the implementation of memristor-based neural networks using only the two extreme resistance states (R_{ON} and R_{OFF}), while still achieving higher weight precision. These types of neural networks could be implemented in the proposed system.

6. Search Operation

Content search is a common operation in many big data applications. This operation is similar to the string search or string matching operation. For string matching, we want to find the appearance of an input string in a set of reference strings. Fig. 8 shows the circuit for matching a single content based on the technique described in [23]. In this circuit, only two conductance levels for the memristors are used (σ_{on}, σ_{off}). Same as the neuron circuit, this design represents logic low by -1V, and logic high by 1V.

This circuit essentially computes the dot product between the applied input voltages and the memristor conductances within the crossbar array, which we refer to as weights. To match an input string or pattern of length n, each weight corresponding to a normal (non-complemented) bit in the string is assigned the maximum positive value, $w=(1/R_{on}-1/R_{off})$ or equivalently (σ_{on} - σ_{off}). In contrast, weights corresponding to complemented bits are assigned the maximum negative value, $-w$. Therefore, for a perfect match, the dot product equals nw. If there is a single-bit mismatch, the result becomes $(n-2)w$. To correctly identify a matching input, the dot product is compared against a threshold value of $(n-1)w$. The circuit shown in Fig. 8 carries out this operation.

For a matching input,

$$V_{col} = \frac{n(\sigma_{on} - \sigma_{off}) - (n-1)(\sigma_{on} - \sigma_{off})}{(2n-1)(\sigma_{on} + \sigma_{off})}$$

Or, $$V_{col} = \frac{(\sigma_{on} - \sigma_{off})}{(2n-1)(\sigma_{on} + \sigma_{off})} \quad (2)$$

For an input with 1-bit mismatch from the reference string/pattern,

$$V_{col} = \frac{(n-2)(\sigma_{on} - \sigma_{off}) - (n-1)(\sigma_{on} - \sigma_{off})}{(2n-1)(\sigma_{on} + \sigma_{off})}$$

Or, $$V_{col} = \frac{-(\sigma_{on} - \sigma_{off})}{(2n-1)(\sigma_{on} + \sigma_{off})} \quad (3)$$

V_{ref} of the comparator is connected to ground and hence it is able to provide the correct string matching result as output. Fig. 9 shows the memristor crossbar-based multiple string matching circuit proposed in [23]. In general, for matching m strings of n-bit length, the circuit in Fig. 9 utilizes a $(4n-2) \times m$ memristor crossbar. Content search operation can be performed in the proposed system based on the technique described above.

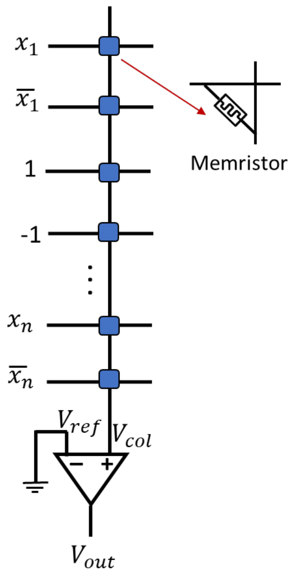

Fig. 8: Memristor-based circuit for matching a single string described in [23].

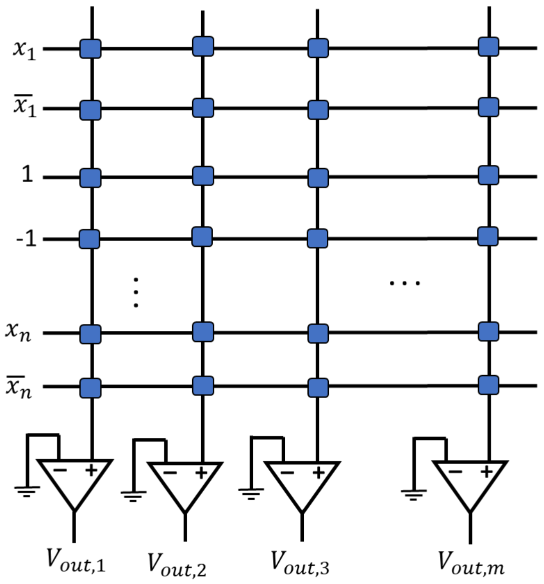

Fig. 9: Memristor crossbar-based string matching circuit described in [23].

7. Boolean Function Implementation

A Boolean function can be implemented as a sum of minterms. Each mean term can be implemented in the memristor crossbar circuit based on the technique used for the search operation. In this approach, each minterm is considered as a reference string/pattern. Implementation of the logical OR operation is also similar to the string matching circuit. Consider the weight $w=(\sigma_{on} - \sigma_{off})$ for each input. If all the n inputs are low, the corresponding dot product will be $-nw$. If at least one input is high, then the corresponding dot product will be greater than or equal to $-(n-2)w$. Thresholding the dot product at $-(n-1)w$ will provide the OR function output. Fig. 10 shows the implementation of a 3-input OR function based on a memristor crossbar circuit.

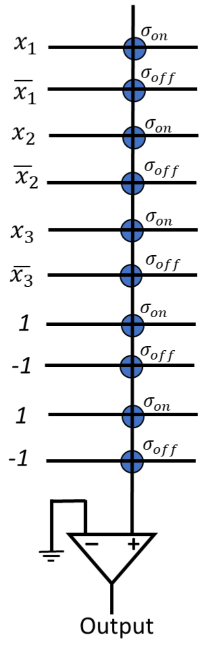

Fig. 10: Memristor-based 3 input OR function circuit.

4. Conclusion

This paper proposes a reconfigurable architecture based on memristor-crossbar tile. In a tile, there is on average ¼ transistor for each memristor for storage, which gives a significant area benefit compared to a 1T1M crossbar tile. A single tile array could be configured for any of the following four operations: memory, search, neural network, and Boolean function implementation. For these operations, we leverage the computation in memory feature of the memristor crossbar. Our future work will compare the area and power consumption of the proposed system with those of an equivalent digital system.

Acknowledgements

This project is supported by the Independent University, Bangladesh (IUB).

References

[1] Chiroma, H., Abdullahi, U. A., Alarood, A. A., Gabralla, L. A., Rana, N., Shuib, L., ... & Herawan, T. (2018). Progress on artificial neural networks for big data analytics: a survey. IEEE Access, 7, 70535-70551.

[2] Abiodun, O. I., Jantan, A., Omolara, A. E., Dada, K. V., Umar, A. M., Linus, O. U., ... & Kiru, M. U. (2019). Comprehensive review of artificial neural network applications to pattern recognition. IEEE access, 7, 158820-158846.

[3] Egmont-Petersen, M., de Ridder, D., & Handels, H. (2002). Image processing with neural networks—a review. Pattern recognition, 35(10), 2279-2301.

[4] H. Esmaeilzadeh, E. Blem, R. Amant, K. Sankaralingam, and D. Burger, "Dark silicon and the end of multicore scaling," in Proceeding of the 38th Annual International Symposium on Computer Architecture, pp. 365–376, 2011.

[5] V. K. Chippa, D. Mohapatra, A. Raghunathan, K. Roy, and S. T. Chakradhar, "Scalable effort hardware design: exploiting algorithmic resilience for energy efficiency," In Proceedings of the 47th Design Automation Conference. ACM, New York, NY, USA, 2010.

[6] S. Venkataramani, V. K. Chippa, S. T. Chakradhar, K. Roy, and A. Raghunathan, "Quality programmable vector processors for approximate computing," In Proceedings of the 46th Annual IEEE/ACM International Symposium on Microarchitecture, USA, 2013.

[7] Knag, P., Kim, J. K., Chen, T., & Zhang, Z. (2015). A sparse coding neural network ASIC with on-chip learning for feature extraction and encoding. IEEE Journal of Solid-State Circuits, 50(4), 1070-1079.

[8] Chua, L. (1971). Memristor-the missing circuit element. IEEE Transactions on circuit theory, 18(5), 507-519.

[9] Lu, W., Kim, K. H., Chang, T., & Gaba, S. (2011, January). Two-terminal resistive switches (memristors) for memory and logic applications. In 16th Asia and South Pacific Design Automation Conference (ASP-DAC 2011) (pp. 217-223). IEEE.

[10] Yu, S., Wu, Y., & Wong, H. S. P. (2011). Investigating the switching dynamics and multilevel capability of bipolar metal oxide resistive switching memory. Applied Physics Letters, 98(10).

[11] Dong, X., Xu, C., Xie, Y., & Jouppi, N. P. (2012). Nvsim: A circuit-level performance, energy, and area model for emerging nonvolatile memory. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 31(7), 994-1007.

[12] Zha, Y., Nowak, E., & Li, J. (2020). Liquid silicon: A nonvolatile fully programmable processing-in-memory processor with monolithically integrated ReRAM. IEEE Journal of Solid-State Circuits, 55(4), 908-919.

[13] Moscibroda, T., & Mutlu, O. (2009, June). A case for bufferless routing in on-chip networks. In Proceedings of the 36th annual international symposium on Computer architecture (pp. 196-207).

[14] Sun, L., Zheng, N., Zhang, T., & Mazumder, P. (2018). Fault modeling and parallel testing for 1T1M memory array. IEEE Transactions on Nanotechnology, 17(3), 437-451.

[15] Yakopcic, C., Hasan, R., & Taha, T. M. (2015). Hybrid crossbar architecture for a memristor based cache. Microelectronics Journal, 46(11), 1020-1032.

[16] Zhang, X., Lu, J., Wang, Z., Wang, R., Wei, J., Shi, T., Dou, C., Wu, Z., Zhu, J., Shang, D. and Xing, G., 2021. Hybrid memristor-CMOS neurons for in-situ learning in fully hardware memristive spiking neural networks. Science Bulletin, 66(16), pp.1624-1633.

[17] Le Gallo, M., Khaddam-Aljameh, R., Stanisavljevic, M., Vasilopoulos, A., Kersting, B., Dazzi, M., Karunaratne, G., Brändli, M., Singh, A., Mueller, S.M. and Büchel, J., 2023. A 64-core mixed-signal in-memory compute chip based on phase-change memory for deep neural network inference. Nature Electronics, 6(9), pp.680-693.

[18] Alibart, F., Zamanidoost, E., & Strukov, D. B. (2013). Pattern classification by memristive crossbar circuits using ex situ and in situ training. Nature communications, 4(1), 2072.

[19] Soudry, D., Di Castro, D., Gal, A., Kolodny, A., & Kvatinsky, S. (2015). Memristor-based multilayer neural networks with online gradient descent training. IEEE transactions on neural networks and learning systems, 26(10), 2408-2421.

[20] Hasan, R., Taha, T.M. and Yakopcic, C., 2017. On-chip training of memristor crossbar based multi-layer neural networks. Microelectronics journal, 66, pp.31-40.

[21] Kim, Y. D., Ahn, H. S., Park, J. Y., Kim, S., & Jeong, D. K. (2006, February). A storage-and power-efficient range-matching TCAM for packet classification. In 2006 IEEE International Solid State Circuits Conference-Digest of Technical Papers (pp. 587-596). IEEE.

[22] Zheng, L., Shin, S., Lloyd, S., Gokhale, M., Kim, K., & Kang, S. M. (2016, May). RRAM-based TCAMs for pattern search. In 2016 IEEE International Symposium on Circuits and Systems (ISCAS) (pp. 1382-1385). IEEE.

[23] Yakopcic, C., Bontupalli, V., Hasan, R., Mountain, D., & Taha, T. M. (2017). Self-biasing memristor crossbar used for string matching and ternary content-addressable memory implementation. Electronics Letters, 53(7), 463-465.

[24] Liu, X., Deng, E., Zhang, H., Zhang, Y., Pan, B., & Kang, W. (2022). Novel nonvolatile lookup table design based on voltage-controlled spin orbit torque memory. IEEE Transactions on Electron Devices, 69(4), 1677-1682.

[25] Xie, L., Du Nguyen, H. A., Taouil, M., Hamdioui, S., & Bertels, K. (2015, October). Fast boolean logic mapped on memristor crossbar. In 2015 33rd IEEE International Conference on Computer Design (ICCD) (pp. 335-342). IEEE.

[26] Chow, P., Seo, S. O., Rose, J., Chung, K., Paez-Monzon, G., & Rahardja, I. (1999). The design of an SRAM-based field-programmable gate array. I. Architecture. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 7(2), 191-197.

[27] Boquet, G., Macias, E., Morell, A., Serrano, J., Miranda, E., & Vicario, J. L. (2021, January). Offline training for memristor-based neural networks. In 2020 28th European Signal Processing Conference (EUSIPCO) (pp. 1547-1551). IEEE.

[28] Hasan, R. (2023, December). Robust Ex-situ Training of Memristor Crossbar-based Neural Network with Limited Precision Weights. In Proceedings of the 18th ACM International Symposium on Nanoscale Architectures (pp. 1-7).

[29] Li, F., Liu, B., Wang, X., Zhang, B., & Yan, J. (2016). Ternary weight networks. arXiv preprint arXiv:1605.04711.