International Journal of Computational Methods and Experimental Measurements
Vol. 12, No. 2, June, 2024, pp. 135-146
Journal homepage: http://iieta.org/journals/ijcmem

Exploring FPGA Implementation and Emulation of Memristor Devices

Zeyad Aklah[1*], Amean Al-Safi[2], Hussein T. Hassan[2]

[1] Department of Information Technology, University of Thi-Qar, Nasiriyah 64001, Iraq
[2] Department of Electrical and Electronics Engineering, University of Thi-Qar, Nasiriyah 64001, Iraq

Corresponding Author Email: zaklah@utq.edu.iq

https://doi.org/10.18280/ijcmem.120203

Received: 24 February 2024
Revised: 13 March 2024
Accepted: 22 March 2024
Available online: 30 June 2024

Keywords:
memristor, FPGA, emulation, modelling, neuromorphic computing

ABSTRACT

This paper explores the field of FPGA implementation and emulation of memristor devices, providing insights into the advancements, challenges, and future directions. The paper discusses various techniques used for FPGA-based memristor emulation, emphasizing the importance of accurate memristor modeling and performance evaluation. It identifies challenges in the field, including improving accuracy, scalability, real-time adaptation, standardization, integration with design tools, and exploring novel applications. Additionally, the results of the study show that FPGAs are one of the viable solutions for emulating memristors. The study concludes that FPGA based memristor emulation holds a promise for studying memristor-based circuits and systems, with potential applications in neuromorphic computing, machine learning accelerators, and analog/mixed-signal circuit design.

1. INTRODUCTION

The rapid advancements in semiconductor technology have driven the need for novel electronic devices with improved performance, energy efficiency, and functionality. Memristors, short for memory resistors, have emerged as a promising technology that exhibits unique properties and capabilities. In contrast to the classical electronic elements like resistors, capacitors, and inductors, memristors possess the ability to retain and store information in the form of resistance variations [1]. This characteristic has attracted considerable attention as it opens up exciting possibilities for the development of new computing paradigms, memory systems, and signal processing applications.

The integration and fabrication of physical memristor devices into existing electronic systems presents several challenges, including fabrication complexity, scalability, and compatibility [2]. In response to these challenges, researchers have explored alternative approaches such as FPGA-based implementation and emulation of memristor devices [3-12]. FPGAs, Field-Programmable Gate Arrays, supply adaptable and easily reconfigurable platforms that can mimic the behavior of complex circuits and devices.

FPGA-based emulation of memristor devices involves creating a digital model of the memristor's behavior and implementing it on an FPGA [11]. This approach allows designers to study and analyze the characteristics of memristors without relying on physical prototypes. For instance, FPGA-based emulation techniques include LUT, look-up table, based emulation [4], where the behavior of the memristor is stored in a lookup table within the FPGA, and HDL (hardware description language) modeling, where the behavior is described using specialized programming languages such as VHDL or Verilog. Alternatively, FPGA-based implementation of memristor devices aims to physically realize memristor behavior using the reconfigurable resources available in FPGAs [3]. This approach involves mapping the memristor's electrical characteristics and functionality onto the FPGA's logic elements, routing resources, and configurable interconnects. Specialized reconfigurable architectures designed specifically for memristor circuits have also been proposed [13], leveraging the unique properties of memristors to enhance performance and energy efficiency.

The main need for FPGA based memristor emulation is to allow designers to overcome the limitations associated with physical memristor devices, including fabrication costs, limited availability, and experimental constraints. FPGAs provide a flexible and accessible platform for rapid prototyping, testing, and evaluation of memristor-based systems and circuits.

The objective of this study is to explore the advancements, challenges, and future prospects of using FPGAs for the implementation and emulation of memristor devices. It delves into the various FPGA-based emulation techniques, such as LUT-based emulation and HDL modeling, highlighting their advantages, limitations, and trade-offs. Additionally, it will investigate FPGA-based implementation methodologies, including mapping memristor behavior to FPGA primitives and leveraging specialized FPGA architectures. The paper will also discuss the performance evaluation and benchmarking approaches employed by designers to assess the effectiveness of FPGA-based memristor implementations. Furthermore, it will address the challenges associated with FPGA-based approaches, such as resource limitations, design complexity,

and the need for accurate memristor models. We will explore potential strategies and research directions for overcoming these challenges and optimizing FPGA-based memristor implementations. Lastly, the applications and impact of FPGA-based emulation and implementation of memristor devices in various domains will be discussed, such as neuromorphic computing, machine learning, hardware security, and reconfigurable computing.

Through this paper, we aim to provide insights and guidance to researchers, engineers, and practitioners interested in exploring the potential of FPGAs for memristor-based technologies.

2. OVERVIEW OF MEMRISTOR TECHNOLOGY

Memristors are a class of electronic devices that have gained considerable interest recently due to their distinct characteristics and possible uses. Coined by Professor Leon Chua in 1971, the term" memristor" refers to a passive two-terminal device that displays a relationship between electric charge and magnetic flux linking the device that is non-linear [14]. Unlike traditional electronic components such as resistors, capacitors, and inductors, memristors possess the ability to retain and store information in the form of resistance variations [15-17].

Memristors are characterized by their ability to change their resistance value based on the electrical current, magnitude and direction, passing through them. This phenomenon, known as resistance switching, allows memristors to exhibit multiple resistance states, that can be used to store and process information [16]. The memristor's resistance state can be altered by applying a suitable voltage or current stimulus, leading to changes in its conductance and resistance levels. Importantly, these resistance states are non-volatile, meaning they persist even when the power supply is disconnected [18].

The underlying mechanism behind memristor behavior involves the migration and redistribution of defects or ions within the memristive material [19]. This process, known as the migration of dopant atoms, results in changes in the conductive pathways within the material, leading to alterations in resistance [20]. Memristors can exhibit different types of resistance switching behaviors, including unipolar and bipolar switching, depending on the specific materials and operating conditions [21, 22].

The distinctive attributes of memristors that make them appealing for various applications. In neuromorphic computing field, memristors have shown promise in emulating the synaptic behavior of biological neurons, enabling the advancement of energy-efficient and brain-inspired computing systems [23]. Memristors also hold potential in non-volatile memory applications, where they can provide high-density storage and faster access times compared to conventional memory technologies [20]. Additionally, memristors have been explored for applications in analog signal processing, where their resistance states can be exploited to perform complex mathematical operations [24].

Despite their potential, the integration and fabrication of physical memristor devices into existing electronic systems present several challenges [23]. Fabrication processes, scalability, and compatibility with established circuit design techniques are among the primary concerns [2]. As a result, researchers [3-5, 7] have explored alternative approaches to harness the benefits of memristor technology, including the use of FPGAs for memristor implementation and emulation.

2.1 Resistance switching and hysteresis

Memristor hysteresis refers to the phenomenon where the electrical resistance of a memristor depends not only on the instantaneous current or voltage applied to it but also on its past history [1]. That means, the memristor's state resistance depends on the path it takes to reach its current state. This behavior is analogous to the hysteresis observed in magnetic materials, where the magnetic field strength depends on the previous magnetic history [25].

When a memristor is subjected to an increasing voltage or current stimulus, it changes from a low resistance state (LRS) to a high resistance state (HRS) [26]. This transition occurs at a particular voltage or current level, known as the SET threshold [27]. Conversely, when the voltage or current is decreased, the memristor transitions from HRS to LRS at a different voltage or current threshold, called the RESET threshold. The existence of these distinct threshold voltages or currents forms the basis of the hysteresis behavior in memristors [26].

The hysteresis in memristors can be visualized on a current-voltage (I-V) graph as depicted in Figure 1 (d). When the applied voltage or current is swept from a negative value to a positive one and back, the I-V curve displays a loop-like curve. This loop represents the hysteresis loop, with the ascending branch corresponding to the SET process and the descending branch corresponding to the RESET process. The area enclosed by the hysteresis loop is an indication of the energy consumed during the resistance switching operation [1].

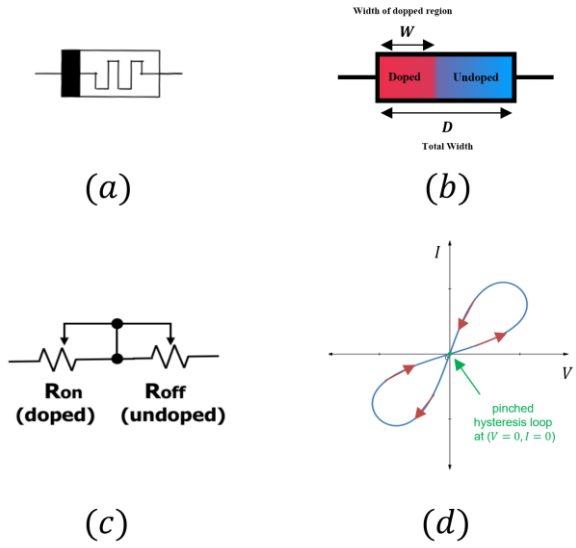

Figure 1. (a) memristor symbol, (b) HP memristor model structure, (c) memristor equivalent circuit representation, (d) memristor's I-V characteristics (hysteresis loop)

The hysteresis phenomenon in memristors has significant implications for their applications. One of the key advantages of hysteresis is that it allows the memristor to retain its resistance state even when the applied voltage or current is removed [28]. This non-volatile property makes memristors suitable for applications in non-volatile memory, where data can be stored even in the absence of power [20]. Moreover, the resistance switching characteristics, combined with the

hysteresis, allow memristors to emulate synaptic behavior, which is crucial for neuromorphic computing applications [16]

2.2 Memristor hysteresis in emulation process

The emulation of memristor hysteresis is a crucial aspect when simulating memristor behavior using digital models or implementing memristor functionality on FPGAs. Capturing hysteresis in the emulation process is essential to replicate the history dependent behavior of physical memristor devices.

To emulate memristor hysteresis, various mathematical formulations and models can be employed. These models aim to reproduce the hysteresis loop and predict the memristor's resistance state based on its electrical history. The choice of the model depends on the specific characteristics of the memristor being emulated and the desired level of accuracy.

One common approach to capture hysteresis in memristor emulation is through the use of window functions. Window functions define different thresholds for the SET and RESET processes, similar to the voltage or current thresholds observed in physical memristors. These thresholds determine the transition points between the HRS and LRS. By incorporating window functions into the emulation process, the resistance switching behavior and hysteresis can be effectively reproduced.

Another approach involves using sigmoidal functions or piecewise linear functions to model the resistance variation in response to the applied current or voltage. These functions map the input stimuli to the corresponding resistance states, considering the previous memristor's history. By carefully selecting the parameters of these functions, such as slope and threshold values, the hysteresis behavior can be represented in the emulation.

In addition to the mathematical models, other techniques can be employed to capture hysteresis in memristor emulation. For instance, data-driven approaches based on machine learning algorithms can be utilized to learn the hysteresis behavior from experimental data or physical memristor measurements. These algorithms can be used to predict the resistance state based on the input history, allowing for emulation of hysteresis.

It is crucial to emphasize that the accuracy of the memristor hysteresis emulation depends not only on the chosen model but also on the fidelity of the emulation platform. FPGAs have been used for memristor emulation due to their reconfigurable nature and ability to mimic complex circuits. The selection of appropriate FPGA resources, such as look-up tables, digital signal processing blocks, or programmable interconnects, is crucial to achieve emulation of memristor hysteresis.

Furthermore, the time scales involved in the resistance switching process should be considered in the emulation. Physical memristors exhibit varying switching speeds, and capturing this behavior accurately is essential for faithful emulation. The choice of simulation or emulation techniques, such as discrete-time models or real-time simulations, should be based on the desired time resolution and the specific dynamics of the memristor being emulated.

3. FPGA-BASED EMULATION OF MEMRISTOR DEVICES

A few research groups have developed FPGA-based emulation platforms for memristor devices. These platforms utilize the reconfigurable nature of FPGAs to implement digital models that capture the essential characteristics of memristors, including resistance switching and hysteresis. By programming the FPGA, designers can simulate the behavior of memristors and investigate their potential applications in various domains.

3.1 Advanced mathematical models

In the field of FPGA-based emulation of memristor devices, advanced mathematical systems have been established to capture the complex behavior exhibited by memristors. These models provide an enhanced comprehension of the underlying mechanisms. Several notable examples of advanced mathematical models for memristor emulation have been proposed, as discussed below.

3.1.1 Extended SPICE models

SPICE, which is an abbreviation for Simulation Program with Integrated Circuit Emphasis, is a well-known simulation tools. Several SPICE macromodels have been suggested for linear [29, 30] and nonlinear ion drift models [31]. In the study [32], the relationship between current and voltage is established using a hyperbolic sine function, while the derivative of the state variable is represented by an exponent. The other model called TEAM [33] utilizes polynomials to capture the behavior of the system. In general, SPICE macromodels requires more computational resources compared to Verilog-A model, which maintains a comparable level of accuracy in the results it produces [32].

3.1.2 Piecewise-linear models

Piecewise-linear models provide an alternative approach to describe the behavior of memristors [34]. These models divide the voltage-current characteristic of the memristor into linear segments, capturing the essential non-linear behavior with a series of linear equations [35]. By fitting experimental data to these piecewise-linear segments, designers can achieve emulation of memristor behavior on FPGA platforms [36]. This modeling technique has been successfully employed in FPGA-based memristor emulation to study the impact of non-linear dynamics on system-level performance [37].

3.1.3 Non-linear differential equations

Non-linear differential equations provide a fundamental approach to model the dynamics of memristor devices. These equations describe the time-varying relationship between voltage and current and capture the essential non-linear behavior observed in memristors. Researchers have developed advanced numerical methods and algorithms to solve these non-linear differential equations efficiently on FPGA platforms [38]. By implementing these equations on FPGAs, researchers can precisely emulate the dynamic behaviors of memristors and study their impact on circuit performance.

3.2 Hardware-software co-design

Hardware-software co-design plays an essential part in ensuring efficiency and emulation effectivity of memristor devices on FPGA platforms. This approach leverages the strengths of both hardware and software components to achieve optimal performance, flexibility, and accuracy in memristor emulation. Below some examples of hardware-software co-design methodologies for memristor emulation

have been proposed.

3.2.1 High-level synthesis (HLS)

High-level synthesis techniques enable the design of memristor emulators at a higher level of abstraction, allowing for faster development and optimization. Researchers have utilized HLS tools such as Vivado HLS or Intel HLS Compiler to convert high-level hardware description languages (HDL) into optimized hardware designs suitable for FPGA implementation [39]. By utilizing HLS, designers can achieve faster development cycles and explore different architectural optimizations.

3.2.2 Custom instruction set architectures (ISA)

To enhance the performance of memristor emulators on FPGAs, custom instruction set architectures (ISA) have been developed. These ISAs are tailored to accelerate specific memristor operations, resulting in flexible emulation. For example, researchers have proposed custom instructions for memristor read/write operations, voltage-controlled switching, or specific mathematical functions related to memristor behavior [40-42]. By implementing these custom ISAs on FPGA platforms, designers can achieve more flexibility in memristor emulation.

3.2.3 Hybrid emulation platforms

Hybrid emulation platforms combine the power of FPGA-based hardware acceleration with the flexibility and versatility of software simulations. These platforms integrate FPGA-based emulators with software-based memristor models or simulators to achieve a balance between speed and accuracy. For instance, researchers have developed frameworks that enable co-simulation between FPGA platforms and software memristor models, allowing for detailed memristor behavior to be simulated in software while leveraging FPGA acceleration for faster execution [43-45]. By utilizing hybrid emulation platforms, designers can explore larger and more complex memristor systems while maintaining accurate and efficient emulation.

3.2.4 Runtime reconfigurability

Runtime reconfigurability is another key aspect of hardware-software co-design in memristor emulation. It allows for dynamic reconfiguration of the FPGA fabric during runtime to adapt to varying emulation requirements or to implement different memristor models or algorithms on the fly. Researchers have explored runtime reconfigurable FPGA architectures and algorithms to enable efficient emulation of memristor devices with varying characteristics or to support real-time adaptation in response to dynamic system behavior [46, 47]. This capability enhances the flexibility and versatility of memristor emulation on FPGA platforms, enabling designers to explore a wide range of memristor-based systems and applications.

3.3 Integration of real-time feedback

The integration of real-time feedback is a significant aspect of FPGA-based emulation of memristor devices, as it allows for dynamic adaptation and control of the emulated memristor behavior. Real-time feedback enables the emulation platform to respond to changing conditions or external stimuli, enhancing the accuracy and versatility of the emulation process. Several notable examples of integrating real-time feedback in FPGA-based memristor emulation have been explored, as discussed below.

3.3.1 Adaptive control strategies

Adaptive control strategies are employed to dynamically adjust the emulated memristor's behavior based on feedback from the system or the environment. These strategies aim to optimize the emulation accuracy by continuously updating the memristor model parameters or adjusting the input signals to match the desired behavior. For instance, researchers have proposed adaptive control algorithms such as PID, proportional-integral-derivative, controllers or fuzzy logic controllers to regulate the memristor's conductance or switching dynamics based on real-time feedback signals [48, 49]. Additionally, researchers have developed closed-loop systems that employ Kalman filters or neural networks to adaptively adjust the memristor model parameters based on real-time feedback from the emulated system [50]. By integrating adaptive control strategies, FPGA-based memristor emulators can emulate a wide range of memristor behaviors accurately.

3.3.2 Sensing and measurement circuits

In FPGA-based memristor emulation, real-time feedback often relies on the integration of sensing and measurement circuits to monitor the behavior of the emulated memristor. These circuits measure the memristor's electrical properties, such as voltage, current, or resistance, and provide feedback signals to adapt the emulation accordingly. Researchers have developed custom measurement circuits and techniques, including voltage sensing circuits, current mirrors, or analog-to-digital converters (ADCs), to accurately capture the memristor behavior in real-time [51]. By integrating these sensing and measurement circuits, FPGA-based emulators can capture the memristor's dynamic behavior and adjust the emulation parameters accordingly.

3.3.3 Hardware-in-the-loop (HIL) testing

Hardware-in-the-loop (HIL) testing techniques are utilized to validate the accuracy and performance of FPGA-based memristor emulators in real-time. HIL testing involves coupling the FPGA-based emulator with real hardware components or systems, creating a closed-loop system where the emulated memristor interacts with the physical world. Researchers have implemented HIL testing setups where the emulated memristor interacts with external circuits, sensors, or actuators, allowing for realtime validation and optimization of the emulation platform [52]. By integrating HIL testing, FPGA-based memristor emulators can be validated under realistic operating conditions, ensuring the reliability and accuracy of the emulation results.

3.4 FPGA-based implementation of memristor devices

Several research groups have made notable contributions to the implementation of memristor devices on FPGA platforms.

Example 1:

Zhang et al. [3] developed an FPGA-based memristor emulator that emulates the behavior of a memristor model described in the study [53]. The model is capable of generating a symmetrical double-loop hysteresis, represented by Eq. (1).

$$I(t) = V(t)(\pm a \pm b \int_0^t V(\tau)d\tau) \quad (1)$$

where, $V(t)$ is the normalized input voltage, $I(t)$ is the normalized output current, and (a, b) are constants. To achieve memristive behavior, the condition $|b| \leq |a|$ must be maintained [3]. In order to implement the memristor model with threshold attribute, Zhang et al. suggested the conductance formula given in Eq. (2).

$$G_t = \begin{cases} G_{t-1} + c \times |V|^n & \text{if } V \geq V_{TH} \\ G_{t-1} & \text{if } -V_{TH} < V < V_{TH} \\ G_{t-1} - c \times |V|^n & \text{if } V < -V_{TH} \\ G_{init} & \text{if reset} \end{cases} \quad (2)$$

where, G_{init} is the initial value of the memristor's conductance, G_t is the instantaneous conductance value, G_{t-1} is the conductance value at the previous moment, V_{TH} is the voltage threshold of the memristor, V is the input voltage, c is a parameter for fine-tuning the pinched hysteresis characteristics of the memristor, and n is the current adjustment parameter. The memristor's conductance changes in response to positive or negative input voltage, either increasing or decreasing, when the absolute value of the input voltage exceeds the absolute value of the voltage threshold of the memristor. This behavior is in accordance with the physical properties inherent to a memristor [3].

To illustrate the implementation of the complete memristor model, the computational steps within the model are illustrated in Figure 2. First, the variables G_{init}, V_{TH}, V_{in}, T, n, and c, are initialized. Subsequently, the input voltage is compared with the threshold voltage level to determine the change in conductance. Then, the output current is obtained by multiplying the input voltage by the conductance value based on Ohm's law. Lastly, the simulation time is assessed, and if the predefined duration is reached, the loop terminates; otherwise, the aforementioned steps are reapeted. The flowchart in Figure 3 shows the computational process within the memristor model, illustrating the realization of the entire model. Initially, the variables G_{init}, V_{TH}, V_{in}, T, n, and c are initialized. The input voltage is then compared with the voltage threshold to determine the alteration in conductivity. Using Ohm's law, the input voltage is multiplied by the conductance value to obtain the output current. The process is repeated until reaching the designated simulation time, at which point the loop is exited [3].

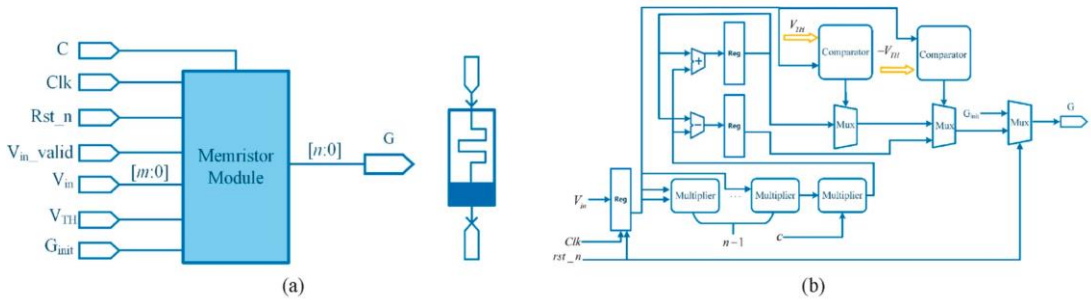

Figure 2. FPGA based implementation of memristor model. (a) The model block diagram. (b) Circuit diagram of the reconfigurable memristor model [3]

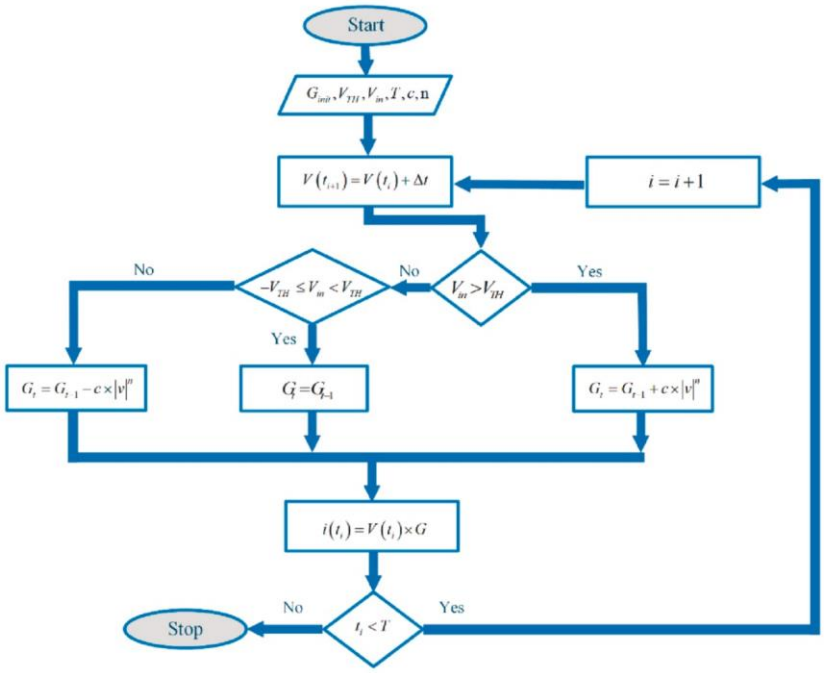

Figure 3. Flowchart of memristor implementation [3]

Their work involved designing custom circuits for memristor, as shown in Figure 3 in a block diagram and data path. Once the input voltage traverses the register, it undergoes a comparison with a predefined voltage threshold. At the same time, Eq. (2) is employed to determine the output conductance, with the resulting calculation being allocated to the conductance through result comparison. The emulator was capable of reproducing the resistance switching behavior and hysteresis of memristor devices. Additionally, they implemented a memristor array of their model on FPGA to conduct storage of binary image binarization using artificial neural network

Example 2:

Tolba et al. [4] use FPGA to build a memristors emulater circuit for Binary Convolutional Neural Networks, utilizing a multi-bit XNOR gate as the primary component for binary convolution. The circuit incorporated a memristor-based pooling layer. The BCNN layer was achieved through a bitwise XNOR-cell followed by a wide NOR gate. The implemented design was implemented on Nexys4 FPGA, demonstrating less than 1% utilization resources have been used.

The two states of the memristor hysteresis loop were determined using the formula $y=(\pm a\pm b)x$ and $y=(\pm a\mp b)x$, while maintaining the condition $|b|\leq|a|$. The implementation of memristor's emulator circuits could be either voltage-controlled or current-controlled, depending on whether $x(t)$ represented the input voltage and $y(t)$ represented the output current, or vice versa. Additionally, a multi-state switching model (consisting of five states) was implemented to extend the two-state memristive model by incorporating additional conditions. The design utilized a 32-bit fixed-point representation, where the most significant bit (MSB) of ϕ selected between b and its inverted value. The output y was obtained by multiplying $(a+b)$ with the input x. Figures 4 and 5 [4] illustrated the two-state and multi-state memristor models, demonstrating the I-V pinched hysteresis loops for the five memristor states and their corresponding representations.

Besides the above two examples, there are more approaches. For instance, the study [54] involved embedding fractional-order systems into FPGA hardware through the utilization of the Xilinx System Generator toolbox within MATLAB Simulink. It further examined a discrete model of a chaotic system featuring a fourth-order memristor with fractional-order dynamics, which was obtained through the conversion of the system's differential version using a finite truncation method. The research also investigated the system's dynamics by exploring the Lyapunov exponents and conducting a bifurcation analysis of the discrete fractional-order memristor system.

Rajagopal et al. [55] introduced a new 4D no equilibrium memristor chaotic system and investigated its dynamic properties to demonstrate its chaotic behavior. They derived a fractional order model of the system from its integer model and analyzed its fractional order bifurcation property. The authors achieved synchronization of identical fractional order memristor chaotic systems utilizing genetically optimized PID controllers and adaptive sliding mode controllers. They conducted numerical simulations to confirm the theoretical findings and showcased the practical feasibility of the suggested system by implementing it on an FPGA.

Zhang et al. [56] presented a behavioral modelling of a general multivalued memristor using FPGA, claiming that its capability of exhibiting behavior similar to electrochemical metallization memories, whether continuous or discrete in nature. The suggested solution was implemented on a Xilinx ZYNQ-7000 FPGA XQ7Z020, utilizing less than 1% of the hardware resources. In order to assess the functionality of the propsed model, the study constructed a quantized artificial neural network using 8-valued memristors in FPGA.

Yu et al. [57] delved into the implementation of a PRNG, pseudo-random number generator, for neural network chaos-based systems on FPGAs. The research addressed the issue of chaotic degradation that was caused by numerical accuracy limitations and had the potential to significantly impact the PRNG's performance. The authors suggested a PRNG featuring a feedback controller derived from a Hopfield neural network chaotic oscillator, with a neuron subjected to electromagnetic radiation. The magnetic flux through the cell membrane of the neuron was chosen as a feedback condition for the controller, creating disturbances among other neurons and preventing periodicity.

Alombah et al. [58] proposed a locally active memristor derived from a current-controlled generic memristor, demonstrating a broad locally active zone. The memristor based chaotic circuit consists of a memristor and an inductor was developed and numerically simulated using MATLAB then validated on FPGA.

Vourkas et al. [59] implemented a bipolar memristor device model represented in Eq. (3), described in the study [60]. The memristance R, in this model, varies at different rates based on the applied voltage, whether it is higher or lower than the threshold voltage v_T. The threshold voltage is considered symmetric for both switching cases (SET and RESET). The memristance is limited by the upper and lower boundaries, denoted as $R_{ON,}$ (R_{MIN}) and R_{OFF} (R_{MAX}), respectively. The change in memristance is controlled by the step function (θ), indicating that R can only change within its limiting values. The change-rate constants α and β determine the rate of change when $|v(t)|<v\ T$ and $|v(t)|>v_T$, respectively.

$$R' = \beta.v + \frac{1}{2}(\alpha-\beta).(|v+v_T| - |v-v_T|.\theta(R-R_{ON}).\theta(R_{OFF} - R) \quad (3)$$

Figure 6 illustrates the block diagram of the memristor's emulator model. The emulator takes inputs such as the top- and bottom-electrode voltages (V_{TE} and V_{BE}), the initial desired memristance (R_{init}) obtained during the initialization phase. The output signal is the current memristance R. The model-specific parameters, including θ, β, and v_T, were considered internal constants.

They employed Euler's method to calculate the memristance (R) in the voltage controlled time-invariant memristor model. Euler's method is a numerical approximation technique that allows for the iterative calculation of a function based on its derivative. In this case, it was used to estimate the change in memristance over time. The model could handle high-frequency input voltage signals by adjustable timestep (Δt). The time-step represents the interval between successive calculations of the memristance value. By appropriately adjusting Δt, the model was able to capture and respond to rapid changes in the input voltage signals.

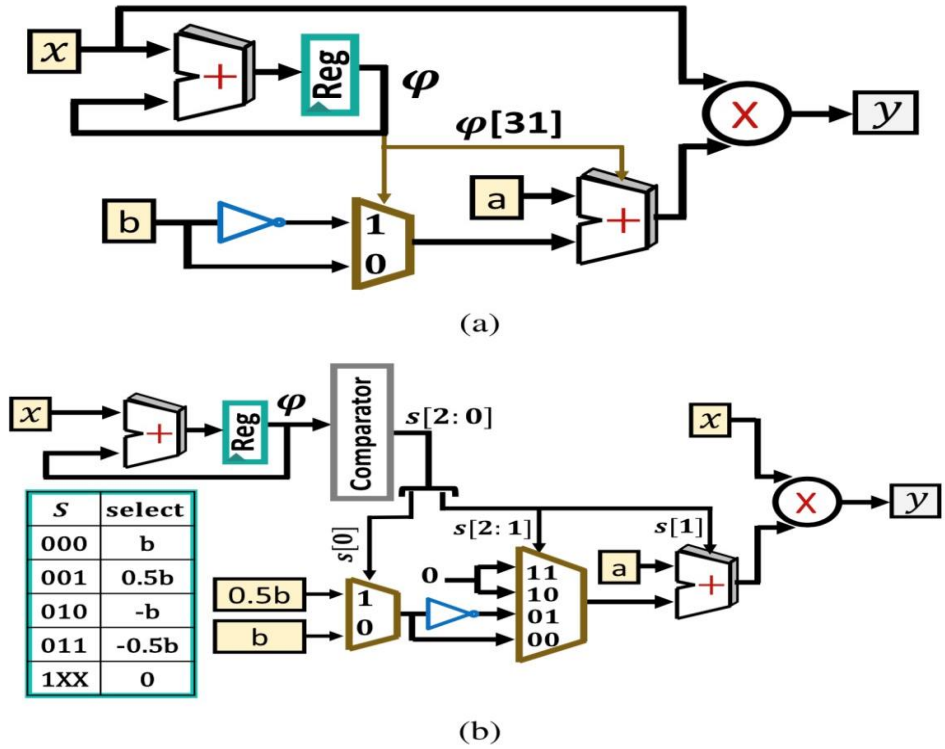

Figure 4. Hardware architecture of the memristor emulator, (a) 2-state memristor model and (b) multistates memristor model [4]

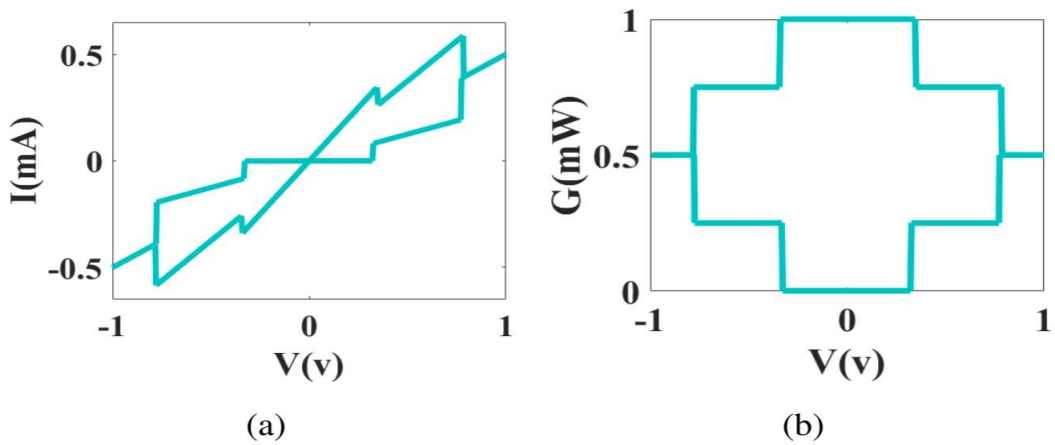

Figure 5. (a) I-V pinched hysteresis loops for 5-memristor states (b) memristor states when the input is cos(wt) [4]

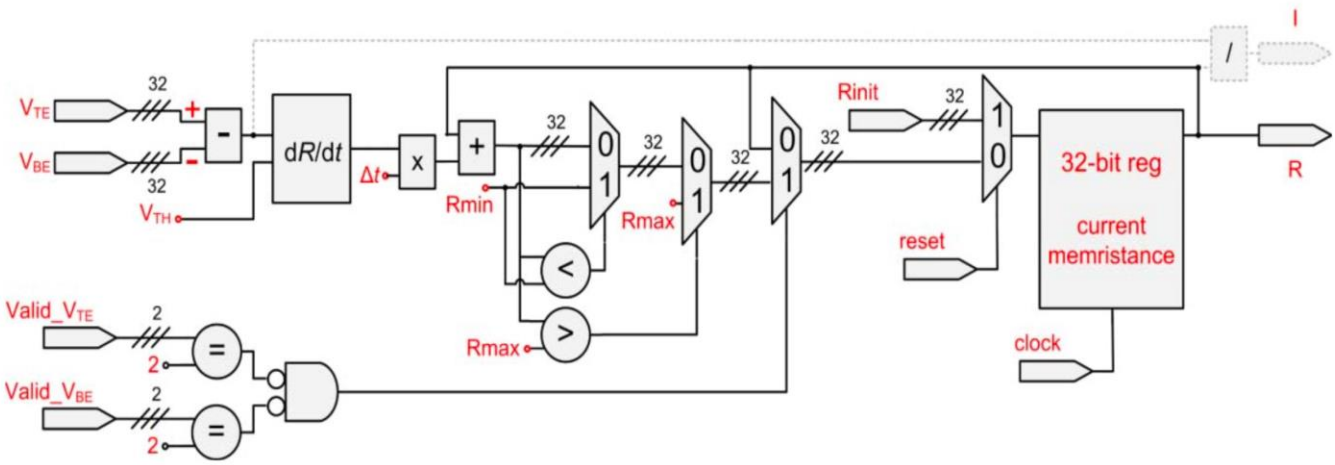

Figure 6. Memristor emulator block diagram [4]

Table 1. Memristor based designs

Memristor Based Application	Algorithm	Evaluation Tool	Ref.
PUMA: Memristor-Based Accelerator	General ML Algorithms	Simulation	[41]
Memristor-Based FPGA (mrFPGA)	-	HSPICE, mrVPR, Xilinx CAD Tools, FPGA	[11]
Machine Learning	BCNN	FPGA	[4]
Neuromorphic	MLP, CNN	Simulation	[61]
Hyperchaotic System	Adomian Decompostion	FPGA	[62]
Memory Model	ANN	Simulation (PSPICE)	[63]
Neuromorphic	ANN	Hardware Realization	[64]
Intellectual Property (IP) core	Memrisitive-Chua chaotic	FPGA	[65]
Machine Learning	ANN	FPGA	[3]
Memristor-Based Look-up Table (LUT)	-	Simulation (LTSPICE IV)	[66]
Hybird CMOS-Memristor Based FPGA	-	TEAM Model	[67]
Logic Circuit (Adder Circuit)	-	FPGA	[68]
Gate Controlled Memristor	Single-Layer Perceptron	FPGA	[6]
Memristor-Transistor Hybrid FPGA	-	Simulation (HSPICE)	[69]
Multivalued Memristor	ANN	FPGA	[56]
Chaotic System	Speech Encryption	FPGA	[70]
Fractional-Order Memristor	Grunwald-Letnikov	FPGA	[71]
Memristor Hardware (HW) Simulator	ANN (Testing)	Quartus II and ModelSim	[72]
Memristor-Based Chaotic Circuit	-	FPGA	[58]

Table 1 provides a compilation of research endeavors focused on the implementation or simulation of memristor-based applications tailored for specific algorithms. The table outlines key details, including the nature of the application, the algorithm it is designed to support, the evaluation tool employed for assessment, and the pertinent references for each work. This compilation serves as a resource, facilitating an organized and insightful exploration of the diverse landscape of memristor applications across various algorithmic domains.

4. CHALLENGES AND FUTURE DIRECTIONS

Despite the significant advancements in FPGA-based memristor emulation, several challenges and opportunities for future research remain. This section discusses some of these challenges and outlines potential directions for future investigations.

(1) Improving Accuracy: One of the primary challenges in FPGA-based memristor emulation is achieving high accuracy. While existing techniques have shown promising results, further refinement is necessary. Future research could focus on developing advanced modeling approaches that capture the intricacies of memristor behavior more accurately. This might involve considering factors such as non-idealities, variability, and temperature effects to enhance the fidelity of emulation.

(2) Scaling to Large-Scale Systems: Another important challenge is scaling FPGAbased memristor emulation to large-scale systems. As the complexity and size of digital circuits increase, there is a need for efficient techniques that can handle larger designs. Future investigations could explore methodologies to partition and distribute the emulation workload across multiple FPGAs or even across FPGA clusters, enabling the emulation of complex systems while maintaining high performance.

(3) Real-Time Adaptation: The dynamic adaptation approach presented in some studies demonstrates the potential for real-time adjustments to improve emulation performance. However, further research is needed to explore advanced adaptation algorithms and strategies. This could involve leveraging machine learning techniques or feedback control mechanisms to dynamically adapt the emulation parameters based on real-time observations. Such approaches would enhance the adaptability and responsiveness of FPGA-based memristor emulation to varying circuit conditions.

(4) Standardization and Benchmarking: To facilitate comparison and evaluation of different FPGA-based memristor emulation techniques, standardization and benchmarking frameworks are essential. Future research could focus on developing standardized benchmarks and performance metrics that consider various aspects, including accuracy, speed, resource utilization, and power consumption. These benchmarks would enable fair comparisons among different emulation approaches and promote the advancement of the field.

(5) Integration with Design Tools: Smooth integration of FPGA-based memristor emulation with existing design tools is crucial for wider adoption. Future investigations could explore methods to integrate emulation frameworks into popular design flows, enabling designers to validate and optimize memristor-based circuits more effectively. This integration could involve the development of plug-ins, APIs, or dedicated design tool extensions that streamline the emulation process and provide designers with comprehensive analysis and debugging capabilities.

(6) Exploring Novel Applications: While FPGA-based memristor emulation has primarily focused on digital circuit design, there is a vast potential for exploring novel applications. Future research could investigate the use of FPGA-based memristor emulation in areas such as neuromorphic computing, machine learning accelerators, and analog/mixed-signal circuit design. Exploring these applications could lead to breakthroughs in diverse domains and further drive the development of FPGA-based memristor emulation techniques.

5. CONCLUSIONS

In conclusion, this study highlights the current state of FPGA implementation and emulation of memristor devices, identifies key challenges, and outlines potential future directions. The study highlighted various techniques and approaches employed in FPGA-based memristor emulation, ranging from modeling and simulation to hardware implementation and validation. These techniques leverage the flexibility and reconfigurability of FPGAs to emulate the behavior of memristor devices, enabling researchers and designers to study and evaluate memristor-based circuits and systems. Moreover, the study identified several challenges that researchers need to address in future investigations. These challenges, described in detail in Section 4, include scaling to large-scale systems, real-time adaptation, standardization and benchmarking, integration with design tools, and exploring novel applications. Overcoming these challenges will contribute to the advancement of FPGA-based memristor emulation and its wider adoption in diverse domains.

REFERENCES

[1] Strukov, D.B., Snider, G.S., Stewart, D.R., Williams, R.S. (2008). The missing memristor found. Nature, 453(7191): 80-83. https://doi.org/10.1038/nature06932

[2] Yang, X., Taylor, B., Wu, A., Chen, Y., Chua, L.O. (2022). Research progress on memristor: From synapses to computing systems. IEEE Transactions on Circuits and Systems I: Regular Papers, 69(5): 1845-1857. https://doi.org/10.1109/TCSI.2022.3159153

[3] Zhang, Z., Li, C., Zhang, W., Zhou, J., Liu, G. (2023). An FPGA-based memristor emulator for artificial neural network. Microelectronics Journal, 131: 105639. https://doi.org/10.1016/j.mejo.2022.105639

[4] Tolba, M.F., Halawani, Y., Saleh, H., Mohammad, B., Al-Qutayri, M. (2020). FPGA-based memristor emulator circuit for binary convolutional neural networks. IEEE Access, 8: 117736-117745. https://doi.org/10.1109/ACCESS.2020.3004535

[5] Wang, X.Y., Wu, Z.R., Zhou, P.F., Iu, H.H.C., Kang, S.M., Eshraghian, J.K. (2022). FPGA synthesis of ternary memristor-CMOS decoders for active matrix microdisplays. IEEE Transactions on Circuits and Systems I: Regular Papers, 69(9): 3501-3511. https://doi.org/10.1109/TCSI.2022.3141087

[6] Zhang, Z., Xu, A., Li, C., Wei, Y., Ge, Z., Cheng, X., Liu, G. (2022). Gate-controlled memristor fpga model for quantified neural network. IEEE Transactions on Circuits and Systems II: Express Briefs, 69(11): 4583-4587. https://doi.org/10.1109/TCSII.2022.3192616

[7] Rajagopal, K., Kingni, S.T., Khalaf, A.J.M., Shekofteh, Y., Nazarimehr, F. (2019). Coexistence of attractors in a simple chaotic oscillator with fractional-order-memristor component: Analysis, FPGA implementation, chaos control and synchronization. The European Physical Journal Special Topics, 228: 2035-2051. https://doi.org/10.1140/epjst/e2019-900001-8

[8] Bahloul, M.A., Naous, R., Masmoudi, M. (2017). Hardware emulation of memristor based ternary content addressable memory. In 2017 14th International Multi-Conference on Systems, Signals & Devices (SSD), Marrakech, Morocco, pp. 446-449. https://doi.org/10.1109/SSD.2017.8167029

[9] Randrianantenaina, J.L., Baran, A.Y., Korkmaz, N., Kiliç, R. (2023). Functional emulator designs for a memristor model with programmable analog and digital platforms. Journal of Computational Electronics, 22(1): 519-530. https://doi.org/10.1007/s10825-022-01966-x

[10] Bontupalli, V., Yakopcic, C., Hasan, R., Taha, T.M. (2018). Efficient memristor-based architecture for intrusion detection and high-speed packet classification. ACM Journal on Emerging Technologies in Computing Systems (JETC), 14(4): 1-27. https://doi.org/10.1145/3264819

[11] Cong, J., Xiao, B. (2011). mrFPGA: A novel FPGA architecture with memristor-based reconfiguration. In 2011 IEEE/ACM International Symposium on Nanoscale Architectures, Diego, CA, USA, pp. 1-8. https://doi.org/10.1109/NANOARCH.2011.5941476

[12] Almurib, H.A., Kumar, T.N., Lombardi, F. (2014). A memristor-based LUT for FPGAs. In the 9th IEEE International Conference on Nano/Micro Engineered and Molecular Systems (NEMS), Waikiki Beach, HI, USA, pp. 448-453. https://doi.org/10.1109/NEMS.2014.6908847

[13] Ranjan, R., Ponce, P.M., Kankuppe, A., John, B., Saleh, L.A., Schroeder, D., Krautschneider, W.H. (2016). Programmable memristor emulator asic for biologically inspired memristive learning. In 2016 39th International Conference on Telecommunications and Signal Processing (TSP), Vienna, Austria, pp. 261-264. https://doi.org/10.1109/TSP.2016.7760874

[14] Chua, L. (1971). Memristor-the missing circuit element. IEEE Transactions on Circuit Theory, 18(5): 507-519. https://doi.org/10.1109/TCT.1971.1083337

[15] Stavrinides, S.G., Picos, R., Corinto, F., Al Chawa, M.M., de Benito, C. (2021). Implementing memristor emulators in hardware. In Mem-elements for Neuromorphic Circuits with Artificial Intelligence Applications. Academic Press, pp. 17-40. https://doi.org/10.1016/B978-0-12-821184-7.00010-4

[16] Chua, L. (2019). Resistance switching memories are memristors. Handbook of Memristor Networks, 197-230. https://doi.org/10.1007/978-3-319-76375-0_6

[17] Jo, S.H., Chang, T., Ebong, I., Bhadviya, B.B., Mazumder, P., Lu, W. (2010). Nanoscale memristor device as synapse in neuromorphic systems. Nano Letters, 10(4): 1297-1301. https://doi.org/10.1021/nl904092h

[18] Kapur, O., Guo, D., Reynolds, J., Han, Y., Beanland, R., Jiang, L., Groot, C. Huang, R. (2022). Back-End-of-Line SiC-Based memristor for resistive memory and artificial synapse. Advanced Electronic Materials, 8(9): 2200312. https://doi.org/10.1002/aelm.202200312

[19] Liu, X.T., Chen, J.R., Wang, Y., Han, S.T., Zhou, Y. (2021). Building functional memories and logic circuits with 2D boron nitride. Advanced Functional Materials, 31(4): 2004733. https://doi.org/10.1002/adfm.202004733

[20] Xia, X., Huang, W., Hang, P., Guo, T., Yan, Y., Yang, J., Yang, D., Yu, X., Li, X.A. (2023). 2D-Material-based volatile and nonvolatile memristive devices for neuromorphic computing. ACS Materials Letters, 5(4): 1109-1135. https://doi.org/10.1021/acsmaterialslett.2c01026

[21] Strukov, D.B., Alibart, F., Stanley Williams, R. (2012).

Thermophoresis/diffusion as a plausible mechanism for unipolar resistive switching in metal-oxide-metal memristors. Applied Physics A, 107: 509-518. https://doi.org/10.1007/s00339-012-6902-x

[22] Gul, F., Efeoglu, H. (2017). Bipolar resistive switching and conduction mechanism of an Al/ZnO/Al-based memristor. Superlattices and Microstructures, 101: 172-179. https://doi.org/10.1016/j.spmi.2016.11.043

[23] Kumar, S., Wang, X., Strachan, J.P., Yang, Y., Lu, W.D. (2022). Dynamical memristors for higher-complexity neuromorphic computing. Nature Reviews Materials, 7(7): 575-591. https://doi.org/10.1038/s41578-022-00434-z

[24] Li, C., Hu, M., Li, Y., Jiang, H., Ge, N., Montgomery, E., Zhang, J., Song, W., Dávila, N., Graves, C.E., Li, Z., Strachan, J.P., Lin, P., Wang, Z., Barnell, M., Wu, Q., Williams, R.S., Yang, J.J., Xia, Q. (2018). Analogue signal and image processing with large memristor crossbars. Nature Electronics, 1(1): 52-59. https://doi.org/10.1038/s41928-017-0002-z

[25] Bharathi, M., Balraj, B., Sivakumar, C., Wang, Z., Shuai, J., Ho, M.S., Guo, D. (2021). Effect of Ag doping on bipolar switching operation in molybdenum trioxide (MoO3) nanostructures for non-volatile memory. Journal of Alloys and Compounds, 862: 158035. https://doi.org/10.1016/j.jallcom.2020.158035

[26] Pedretti, G., Milo, V., Ambrogio, S., Carboni, R., Bianchi, S., Calderoni, A., Ramaswamy, N., Spinelli, A., Ielmini, D. (2017). Memristive neural network for on-line learning and tracking with brain-inspired spike timing dependent plasticity. Scientific Reports, 7(1): 5288. https://doi.org/10.1038/s41598-017-05480-0

[27] Sparvoli, M., Gazziro, M.A., Marma, J.S., Zucchi, G. (2019). Memristor device fabricated from doped graphene oxide. In 2019 IEEE 10th Latin American Symposium on Circuits & Systems (LASCAS), Armenia, Colombia, pp. 57-60. https://doi.org/10.1109/LASCAS.2019.8667547

[28] Lanza, M., Wong, H.S.P., Pop, E., Ielmini, D., Strukov, D., Regan, B.C., Larcher, L., Villena, M.A., Yang, J.J., Goux, L., et al. (2019). Recommended methods to study resistive switching devices. Advanced Electronic Materials, 5(1): 1800143. https://doi.org/10.1002/aelm.201800143

[29] Biolek, Z., Biolek, D., Biolkova, V. (2009). SPICE model of memristor with nonlinear dopant drift. Radioengineering, 18(2).

[30] Rak, A., Cserey, G. (2010). Macromodeling of the memristor in SPICE. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 29(4): 632-636. https://doi.org/10.1109/TCAD.2010.2042900

[31] Lehtonen, E., Laiho, M. (2010). CNN using memristors for neighborhood connections. In 2010 12th International Workshop on Cellular Nanoscale Networks and Their Applications (CNNA 2010), Berkeley, CA, USA, pp. 1-4. https://doi.org/10.1109/CNNA.2010.5430304

[32] Yakopcic, C., Taha, T.M., Subramanyam, G., Pino, R.E., Rogers, S. (2011). A memristor device model. IEEE Electron Device Letters, 32(10): 1436-1438. https://doi.org/10.1109/LED.2011.2163292

[33] Kvatinsky, S., Friedman, E.G., Kolodny, A., Weiser, U.C. (2012). TEAM: Threshold adaptive memristor model. IEEE Transactions on Circuits and Systems I: Regular Papers, 60(1): 211-221. https://doi.org/10.1109/TCSI.2012.2215714

[34] Ding, S., Wang, N., Bao, H., Chen, B., Wu, H., Xu, Q. (2023). Memristor synapse-coupled piecewise-linear simplified Hopfield neural network: Dynamics analysis and circuit implementation. Chaos, Solitons & Fractals, 166: 112899. https://doi.org/10.1016/j.chaos.2022.112899

[35] Itoh, M., Chua, L.O. (2008). Memristor oscillators. International Journal of Bifurcation and Chaos, 18(11): 3183-3206. https://doi.org/10.1142/S0218127408022354

[36] Kim, H., Sah, M.P., Yang, C., Roska, T., Chua, L.O. (2011). Memristor bridge synapses. Proceedings of the IEEE, 100(6): 2061-2070. https://doi.org/10.1109/JPROC.2011.2166749

[37] Lin, X., Pi, X., Wang, X., Du, P., Lu, H. (2022). FPGA implementation of piecewise linear spiking neuron and simulation of cortical neurons. Microprocessors and Microsystems, 91: 104516. https://doi.org/10.1016/j.micpro.2022.104516

[38] Zhang, X.J., Iannaccone, G., Campabadal, F., Crupi, A., Ascoli, A., Cester, A. (2019). Nonlinear dynamic modeling of resistive switching memories and emulation on FPGA. IEEE Transactions on Circuits and Systems I: Regular Papers, 66(5): 1959-1971.

[39] Mami, S., Lahbib, Y., Hachaichi, Y., Mami, A. (2019). XSG-based HLS flow for optimized signal processing designs for FPGAs. Microprocessors and Microsystems, 66: 31-42. https://doi.org/10.1016/j.micpro.2019.02.001

[40] Xia, L., Li, B., Tang, T., Gu, P., Chen, P.Y., Yu, S., Cao, Y., Wang, Y., Xie, Y., Yang, H. (2017). MNSIM: Simulation platform for memristor-based neuromorphic computing system. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 37(5): 1009-1022. https://doi.org/10.1109/TCAD.2017.2729466

[41] Ankit, A., Hajj, I.E., Chalamalasetti, S.R., Ndu, G., Foltin, M., Williams, R.S., Faraboschi, P., Hwu, W.M.W., Strachan, J.P., Roy, K., Milojicic, D.S. (2019). PUMA: A programmable ultra-efficient memristor-based accelerator for machine learning inference. In Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems. Association for Computing Machinery, New York, NY, USA, pp. 715-731. https://doi.org/10.1145/3297858.3304049

[42] Ambrosi, J., Ankit, A., Antunes, R., Chalamalasetti, S.R., Chatterjee, S., Hajj, I.E., Fachini, G., Faraboschi, P., Foltin, M., Huang, S., Hwu, W.M., Knuppe, G., Lakshminarasimha, S.V., Milojicic, D., Parthasarathy, M., Ribeiro, F., Rosa, L., Roy, K., Silveira, P., Strachan, J.P. (2018). Hardware-software co-design for an analog-digital accelerator for machine learning. In 2018 IEEE International Conference on Rebooting Computing (ICRC), McLean, VA, USA, pp. 1-13. https://doi.org/10.1109/ ICRC.2018.8638612

[43] Lammie, C., Xiang, W., Linares-Barranco, B., Azghadi, M.R. (2022). MemTorch: An open-source simulation framework for memristive deep learning systems. Neurocomputing, 485: 124-133. https://doi.org/10.1016/j.neucom.2022.02.043

[44] Wang, W., State Univ of New York at Albany Coll of Nanoscale Science and Engineering. (2011). Complimentary metal oxide semiconductor (CMOS)-

Memristor Hybrid Nanoelectronics.
[45] Rothenbuhler, A., Tran, T., Smith, E.H.B., Saxena, V., Campbell, K.A. (2013). Reconfigurable threshold logic gates using memristive devices. Journal of Low Power Electronics and Applications, 3(2): 174-193. https://doi.org/10.3390/jlpea3020174
[46] Mayacela, M., Rentería, L., Contreras, L., Medina, S. (2022). Comparative analysis of reconfigurable platforms for memristor emulation. Materials, 15(13): 4487. https://doi.org/10.3390/ma15134487
[47] Rajagopal, K., Tuna, M., Karthikeyan, A., Koyuncu, İ., Duraisamy, P., Akgul, A. (2019). Dynamical analysis, sliding mode synchronization of a fractional-order memristor Hopfield neural network with parameter uncertainties and its non-fractional-order FPGA implementation. The European Physical Journal Special Topics, 228: 2065-2080. https://doi.org/10.1140/epjst/e2019-900005-8
[48] Sahin, M.E., Cam Taskiran, Z.G., Guler, H., Hamamci, S.E. (2020). Application and modeling of a novel 4D memristive chaotic system for communication systems. Circuits, Systems, and Signal Processing, 39: 3320-3349. https://doi.org/10.1007/s00034-019-01332-6
[49] Han, Y., Wang, M., Zhao, H., Xie, Y. (2020). Adaptive control of memristive systems: A review. IEEE Transactions on Industrial Electronics, 68(2): 1726-1739.
[50] Wang, M., Han, Y., Xie, Y. (2017). Real-time feedback control of memristor emulation in field-programmable gate arrays. IEEE Transactions on Industrial Informatics, 13(6): 3314-3324.
[51] Shi, H., Yu, H., Gong, D., Song, L., Li, S. (2021). Closed-loop FPGA emulation of memristor-based systems: A neural network perspective. IEEE Transactions on Industrial Informatics, 17(2): 962-971.
[52] Liu, X., Wang, H., Zhang, Y., Zhao, J., Li, Y. (2019). Hardware-in-the-loop validation of fpga-based memristor emulators. IEEE Access, 7: 162234-162243.
[53] Halawani, Y., Mohammad, B., Al-Qutayri, M., Al-Sarawi, S.F. (2018). Memristor-based hardware accelerator for image compression. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 26(12): 2749-2758. https://doi.org/10.1109/TVLSI.2018.2835572
[54] Karthikeyan, A., Rajagopal, K. (2018). FPGA implementation of fractional-order discrete memristor chaotic system and its commensurate and incommensurate synchronisations. Pramana, 90(1): 14. https://doi.org/10.1007/s12043-017-1507-8
[55] Rajagopal, K., Guessas, L., Karthikeyan, A., Srinivasan, A., Adam, G. (2017). Fractional order memristor no equilibrium chaotic system with its adaptive sliding mode synchronization and genetically optimized fractional order PID synchronization. Complexity, 2017. https://doi.org/10.1155/2017/1892618
[56] Zhang, Z., Xu, A., Ren, H.T., Liu, G., Cheng, X. (2022). Reconfigurable multivalued memristor FPGA model for digital recognition. International Journal of Circuit Theory and Applications, 50(11): 3846-3860. https://doi.org/10.1002/cta.3377
[57] Yu, F., Zhang, Z., Shen, H., Huang, Y., Cai, S., Jin, J., Du, S. (2021). Design and FPGA implementation of a pseudo-random number generator based on a Hopfield neural network under electromagnetic radiation. Frontiers in Physics, 9: 690651. https://doi.org/10.3389/fphy.2021.690651
[58] Alombah, N.H., Tchendjeu, A.E.T., Romanic, K., Talla, F.C., Fotsin, H.B. (2021). FPGA implementation of a novel two-internal-state memristor and its two component chaotic circuit. Indian Journal of Science and Technology, 14(27): 2257-2271. https://doi.org/10.17485/IJST/v14i27.532
[59] Vourkas, I., Abusleme, A., Ntinas, V., Sirakoulis, G.C., Rubio, A. (2016). A digital memristor emulator for FPGA-based artificial neural networks. In 2016 1st IEEE International Verification and Security Workshop (IVSW), Sant Feliu de Guixols, Spain, pp. 1-4. https://doi.org/10.1109/IVSW.2016.7566607
[60] Pershin, Y.V., Di Ventra, M. (2010). Experimental demonstration of associative memory with memristive neural networks. Neural Networks, 23(7): 881-886. https://doi.org/10.1016/j.neunet.2010.05.001
[61] Wang, Y., Wu, S., Tian, L., Shi, L. (2020). SSM: A high-performance scheme for in situ training of imprecise memristor neural networks. Neurocomputing, 407: 270-280. https://doi.org/10.1016/j.neucom.2020.04.130
[62] Rajagopal, K., Karthikeyan, A., Srinivasan, A. (2018). Dynamical analysis and FPGA implementation of a chaotic oscillator with fractional-order memristor components. Nonlinear Dynamics, 91(3): 1491-1512. https://doi.org/10.1007/s11071-017-3960-9
[63] Ma, D., Wang, G., Han, C., Shen, Y., Liang, Y. (2018). A memristive neural network model with associative memory for modeling affections. IEEE Access, 6: 61614-61622. https://doi.org/10.1109/ACCESS.2018.2875433
[64] Chu, M., Kim, B., Park, S., Hwang, H., Jeon, M., Lee, B.H., Lee, B.G. (2014). Neuromorphic hardware system for visual pattern recognition with memristor array and CMOS neuron. IEEE Transactions on Industrial Electronics, 62(4): 2410-2419. https://doi.org/10.1109/TIE.2014.2356439
[65] Tolba, M.F., Fouda, M.E., Hezayyin, H.G., Madian, A.H., Radwan, A.G. (2018). Memristor FPGA IP core implementation for analog and digital applications. IEEE Transactions on Circuits and Systems II: Express Briefs, 66(8): 1381-1385. https://doi.org/10.1109/TCSII.2018.2882496
[66] Kumar, T.N., Almurib, H.A., Lombardi, F. (2014). A novel design of a memristor-based look-up table (LUT) for FPGA. In 2014 IEEE Asia Pacific Conference on Circuits and Systems (APCCAS), Ishigaki, Japan, pp. 703-706. https://doi.org/10.1109/APCCAS.2014.7032878
[67] Sampath, M., Mane, P.S., Ramesha, C.K. (2015). Hybrid CMOS-memristor based FPGA architecture. In 2015 International Conference on VLSI Systems, Architecture, Technology and Applications (VLSI-SATA), Bengaluru, India, pp. 1-6. https://doi.org/10.1109/VLSI-SATA.2015.7050461
[68] Yang, L., Wang, Y., Wu, Z., Wang, X. (2021). FPGA implementation of threshold-type binary memristor and its application in logic circuit design. Micromachines, 12(11): 1344. https://doi.org/10.3390/mi12111344
[69] Aslam, M.H., Farooq, U., Awais, M.N., Bhatti, M.K., Shehzad, N. (2016). Exploring the effect of LUT size on the area and power consumption of a novel memristor-transistor hybrid FPGA architecture. Arabian Journal for Science and Engineering, 41: 3035-3049.

https://doi.org/10.1007/s13369-016-2068-8
[70] Tolba, M.F., Sayed, W.S., Fouda, M.E., Saleh, H., Al-Qutayri, M., Mohammad, B., Radwan, A.G. (2019). Digital emulation of a versatile memristor with speech encryption application. IEEE Access, 7: 174280-174297. https://doi.org/10.1109/ACCESS.2019.2957300
[71] Zhang, X., Yang, G., Liu, S., Moshayedi, A.J. (2022). Fractional-order circuit design with hybrid controlled memristors and FPGA implementation. AEU-International Journal of Electronics and Communications, 153: 154268. https://doi.org/10.1016/j.aeue.2022.154268
[72] Ntinas, V., Vourkas, I., Abusleme, A., Sirakoulis, G.C., Rubio, A. (2018). Experimental study of artificial neural networks using a digital memristor simulator. IEEE Transactions on Neural Networks and Learning Systems, 29(10): 5098-5110. https://doi.org/10.1109/TNNLS.2018.2791458