encyclopedia

MDPI

Entry

Memristor-Based Logic Circuits: Concept and Prospect

Zhipeng Liao [1], Abusaleh Jabir [2], Guangyu Xue [1], Zohreh Hajiabadi [1], Vihirthan Ravitharan [1], Sridhar Chandrasekaran [3] and Firman M. Simanjuntak [1,4,*]

[1] School of Electronics & Computer Science, University of Southampton, Southampton SO17 1BJ, UK
[2] School of Engineering, Computing, and Mathematics, Oxford Brookes University, Oxford OX3 0BP, UK
[3] School of Electronics Engineering, Vellore Institute of Technology, Chennai 600127, India
[4] ECS Centre for Neuromorphic Technologies, University of Southampton, Southampton SO17 1BJ, UK
* Correspondence: f.m.simanjuntak@soton.ac.uk

Definition

As computational demands from artificial intelligence increase, traditional von Neumann architectures face memory wall bottlenecks while CMOS technology approaches physical limitations defined by Moore's law and Dennard scaling. This entry examines memristor-based logic circuits as potential solutions to overcome these computing constraints. We systematically analyse various memristor-based logic implementation approaches by reviewing and comparing stateful methods, including Material Implication (IMPLY) and Memristor-Aided Logic (MAGIC), as well as non-stateful techniques such as Memristor-Ratioed Logic (MRL) and Scouting Logic (SL). Each approach demonstrates unique capabilities for implementing fundamental logic operations, with progressive improvements in efficiency documented across multiple research studies. We discuss that while memristor-based logic shows promise, significant limitations exist: crossbar arrays can only integrate specific gates (NOT, NOR), MAGIC architecture struggles with multilevel cascade and large fan-out circuits, IMPLY requires numerous operational steps, and MRL faces challenges integrating CMOS-based inverters into crossbars.

Keywords: memristor; logic gates; IMPLY; MAGIC; MRL; SL

1. Introduction

With the intelligent revolution sweeping the world, we have entered the era of artificial intelligence in the 21st century. AI machines that use fully neural network models may require high computational complexity and computational power [1]. The improvement of hardware computing power relies on its logic gates, where its innovation is strongly dependent on CMOS architecture and fabrication; however, classical logic gate designs cannot keep up with Moore's law [2] and Dennard scaling [3]. Moreover, today's von Neumann computer architecture suffers from a data transmission bottleneck, where the computing speed is further limited by how fast the data can flow back and forth between the storage unit and the computing unit [4]. Such frequent data movement is accompanied by considerable energy consumption and is therefore inefficient [5]. The emergence of non-von Neumann architectures, such as quantum and neuromorphic computing, offers a promising solution to overcome the limitations of today's computing speed and power efficiency [4,6].

Nevertheless, these emerging computer architectures still require logic gates to run the peripheral circuitry, for which today's classical logic gate designs are often not suitable to support low-power computing hardware, especially when they need to be operated in

Check for updates

Academic Editor: Raffaele Barretta

Received: 8 April 2026
Revised: 22 June 2026
Accepted: 10 August 2026
Published: 1 September 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

extreme environments, e.g., cryogenic and high-energy radiation ambient; this is due to transistors not performing well in these environments [7,8]. These considerations have stimulated interest in emerging switching devices, especially memristors, for constructing compact and low-power logic circuits with reduced transistor overhead [6]. The memristor was originally postulated by Chua as the fourth fundamental passive circuit element relating charge and flux [9], and was later physically realised in nanoscale resistive-switching devices [10], which triggered extensive research into memristor-based memory, neuromorphic hardware, and logic architectures [6,11].

Among all arithmetic logic operations, addition serves as the foundation for all computational operations. Both subtraction and multiplication can be achieved through the concatenation of adders or by taking the two's complement, while division can be constructed using the intersections of subtracters and adders. Further, in the binary domain, addition can be broken down into multiple logical operations, allowing the use of logic gates to build an adder. Modern digital systems commonly use two well-defined states in CMOS circuits to represent logic 1 and logic 0, and then construct arithmetic modules based on Boolean operations. In memristor devices, logic states can similarly be encoded by the high-resistance state (HRS) and low-resistance state (LRS), creating the possibility of implementing logic using resistance switching rather than only transistor-based voltage switching [12,13].

In 2012, S. Kvatinsky et al. [12], proposed the construction of OR and AND gates using two different states of high and low resistance of a memristor to represent logic 1 and logic 0. Since then, several major families of memristor-based logic have been developed. From a circuit perspective, these schemes can be broadly divided into stateful logic, where logical operands and results are stored as resistance states in the devices, and non-stateful logic, where the output is represented by voltage or current and interpreted by peripheral circuitry or sensing structures. Representative stateful families include IMPLY logic and MAGIC, whereas representative non-stateful or read-based families include MRL and Scouting Logic. More recent studies have further extended the field from primitive gate demonstrations to experimentally validated and reconfigurable logic-in-memory and arithmetic implementations [14–18]. Therefore, this entry discusses the conceptual development of memristor-based logic circuits, compares their representative operating styles, and summarises their limitations and future prospects.

2. Operation Mechanism

2.1. Memristor Device Models and Switching Dynamics

Since this article surveys logic schemes reported by various research groups, the underlying circuit analyses in the literature are based on diverse memristor models. To improve physical clarity, ensure reproducibility, and establish a quantitative baseline for the logic operations discussed in this survey, it is essential to explicitly define the representative mathematical modelling frameworks.

At the circuit level, a memristor device is commonly described by a state-dependent current-voltage relation and an internal state variable x:

$$I(t) = G(x, V, t)V(t) \tag{1}$$

$$\frac{dx}{dt} = F(x, V, t) \tag{2}$$

where x is the internal state variable (typically representing the normalised conductive filament length, oxygen vacancy concentration, or dopant boundary), and G is the state-dependent conductance. Early compact formulations utilised linear or nonlinear ion-drift models combined with window functions to constrain the state evolution near the physical

boundaries of the device [19]. For instance, the nonlinear dopant drift model employs a window function $F(x) = 1 - (2x - 1)^{2p}$ to suppress unphysical state changes at the boundaries [19].

However, standard ion-drift models do not explicitly define a "no-switching" voltage zone. For logic circuit analysis—where state preservation during read or cascaded operations is critical—threshold-type models (such as the VTEAM model or threshold SPICE models) are the standard reference frameworks [20,21]. In this survey, qualitative statements regarding voltage division (e.g., "the divided voltage is insufficient to change the state") are strictly interpreted using a generic threshold-based mathematical framework. The nonlinear switching dynamics can be expressed as a piecewise function:

$$\frac{dx}{dt} = \begin{cases} k_{set}\left(\frac{V(t)}{V_{set}} - 1\right)^{\alpha_{set}} f_{set}(x), \ V \geq V_{set} \\ 0, \ V_{reset} < V < V_{set} \\ k_{reset}\left(\frac{V(t)}{V_{reset}} - 1\right)^{\alpha_{reset}} f_{reset}(x), \ V_{reset} \leq V \end{cases} \tag{3}$$

where V_{set} and V_{reset} denote the positive SET and negative $RESET$ threshold voltages, respectively. The parameters k_{set} and k_{reset} determine the switching rates, while α_{set} and α_{reset} explicitly define the nonlinearity dynamics of the switching process, capturing the exponential or high-order polynomial dependence of ionic drift on the applied electric field. The functions $f_{set}(x)$ and $f_{reset}(x)$ are boundary conditions.

For typical oxide-based bipolar resistive switching memory (RRAM) devices referenced in logic circuit designs, specific parameters for the switching thresholds generally fall within the following representative ranges: $V_{set} \approx 0.5$ V to 1.5 V and $V_{reset} \approx -0.5$ V to -1.5 V, depending on the specific material stack (e.g., HfOx, TaOx) and compliance current settings.

Based on this mathematical framework, the condition for a voltage division to be considered "insufficient to change the state" is not a universal, device-independent assumption, but a rigorous, reproducible criterion. Specifically, a logic operation is non-disturbing if and only if the voltage across the memristor, V_M, satisfies the voltage constraint:

$$V_{reset} < V_M < V_{set} \tag{4}$$

and the accumulated state change over the applied pulse width T_p is negligible:

$$|\Delta x| = \left| \int_0^{T_p} \frac{dx}{dt} dt \right| \ll 1 \tag{5}$$

Beyond basic threshold models, evaluating the robustness of memristor-based logic requires more comprehensive physical models that account for Joule heating, temperature-dependent ionic mobility, and stochastic filamentary dynamics [22–24]. For realistic circuit simulations, the Stanford-PKU compact model and its variability-aware extensions are widely adopted [25,26]. These advanced models incorporate cycle-to-cycle (C2C) and device-to-device (D2D) variations in the high-resistance state (HRS), low-resistance state (LRS), and threshold voltages [24,27]. Consequently, practical logic architectures must be designed with sufficient voltage margins to tolerate these statistical variations, ensuring that the inequality $V_{reset} < V_M < V_{set}$ holds true even under worst-case variability scenarios.

2.2. Stateful

Stateful logic operation is a concept where devices themselves both store logical values within their state and perform logic operations on them, with the results also stored back in the devices. This contrasts with the traditional approach of separating storage and

computation for logic operations. Memristor devices are well-suited for stateful logic operations. For example, a true/1 (false/0) logic value can be stored (encoded) in its HRS (LRS), etc. These devices can also be configured to perform specific logic operations, with the results stored back in the memristor device. Based on this concept, various memristor-based stateful logic systems have been proposed.

2.2.1. Material Implication (IMPLY)

The concept of implication in logic (A → B) states that if proposition A implies proposition B, it means that the truth of proposition A implies the truth of proposition B. However, the truth of proposition B does not necessarily mean the truth of proposition A. The truth table for implied logic is shown in Figure 1 [28]. The memristor exhibits a natural advantage in implementing implication operations, and its implementation method is much simpler compared to those based on CMOS. In 2009 [29], Lehtonen and Laiho utilised two memristors, denoted as p and q, to represent events A and B. The resulting operational output of the implication operation was then stored in memristor q′. In this system, the high resistance state of the memristor represents logic 0 (open state), while the low resistance state represents logic 1 (closed state). The resistance value of the ground resistor RG falls between the on and off resistances of the memristor. Voltages Vcond (V_P) and Vset (V_Q) are individually applied to memristor p and q, respectively. When p is in a high resistance state (logic 0), because the voltage of Vreset is greater than the voltage of Vcond, and the partial voltage of p is less than the voltage the reset requires, the state of p is unchanged. Due to Roff being much greater than the ground resistor RG, the voltage of RG is very close to the ground, so the voltage of q is approximately Vset, setting the state of q to logical 1. If the state of q is logical 0, then its state will be set to 1; if the state of q is logical 1, it will not change. When p is in a low-resistance state (logic 1), the ground resistor RG is much larger than the memristor on-resistance. As a result, the voltage of the resistor RG is close to Vcond, while the voltage of the memristor p is close to 0. Thus, the state of p remains unchanged. On the contrary, the partial voltage of q is Vset-Vcond, which is lower than the voltage necessary to reach the set state. As a result, the state of q remains unchanged. This method introduces a serial adder that utilises implication logic [29]. To produce the result, it necessitates $3n + 3$ memristors (where n is the number of input data bits) and 89 steps.

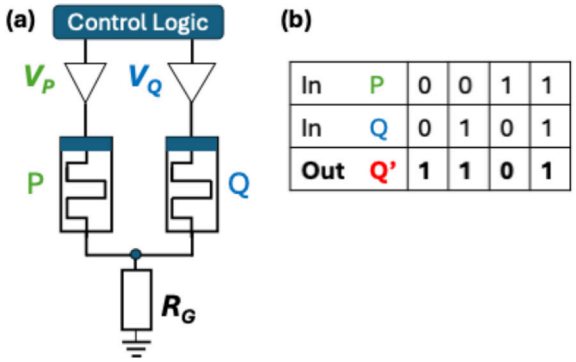

Figure 1. Material Implication with memristors (**a**) IMPLY gate design and its truth table (**b**), adapted from [28] (the triangle symbols are tri-state voltage drivers, connecting the memristor terminals to the required operation voltages, such as V_{cond} and V_{set}, or place the terminal in a high-impedance state when it is not actively driven).

In 2010, J. Borghetti et al. [13] from HP Labs proposed that implication operations could be employed to carry out arbitrary logical operations, indicating that a comprehensive set of logical operations could be achieved by constructing implication operations

using memristors. They demonstrated the feasibility of constructing all arithmetic logic operations based on implied logic operations. In 2014, Kvatinsky et al. [30] introduced a new method for implementing a memristor-based full adder, significantly reducing the required number of operation steps. In this work, exclusive OR (XOR) logic was constructed using a sequence of Material Implication operations together with initialization steps. Since XOR is a non-monotonic Boolean function, it cannot be realised by a single implication operation; instead, auxiliary memristors (function memristors M_1, M_2) are employed to store intermediate states during the operation sequence [13]. More specifically, the method first initialises the auxiliary memristors, then generates implication-based intermediate terms from the two input operands, and finally combines these stored intermediate states to obtain the XOR output. In this way, the XOR function is realised as a stepwise stateful operation rather than as a single physical gate. Based on this IMPLY-based XOR construction, the Sum and Carry-out expressions of the full adder were reformulated, reducing the total number of steps to 29, with 13 steps required for the XOR operation [30].

In 2017, Rohani and Taherinejad [31] proposed a novel method to further reduce the steps required and the number of memristors. This method simplifies the number of implication operations necessary to implement the XOR operation using ten equivalent implication formulas. This approach further decreases the number of steps for the one-bit adder to 22. In 2018, Karimi and Rezai [32] proposed a method to increase the coincidence degree of calculating the Sum and Carry while further reducing the extra operations required to calculate the different parts. This further reduces the total number of steps to 21, which is now the minimum number of steps required for the implication-based logic adder implementation.

However, optimizing a single-bit adder is only the first step toward practical in-memory arithmetic. Subsequent studies demonstrate that IMPLY-based design quickly expanded from simple serial step-minimization to parallelization, scalability, and application-specific adaptations. In 2020, Rohani et al. proposed a semi-parallel IMPLY full adder that was significantly faster than earlier serial designs, while Radakovits et al. introduced a semi-serial IMPLY adder to build a memristor multiplier, highlighting the critical trade-off between latency and hardware cost [33,34]. By 2023, Kaushik and Bodapati extended IMPLY logic to high-speed conditional-carry and carry-select adders for n-bit in-memory arithmetic [35]. Concurrently, optimization efforts expanded into energy-efficient approximate computing, as demonstrated by Fatemieh et al., who proposed approximate serial IMPLY-based full adders for error-tolerant applications [36]. More recently, in 2025, Seiler and co-workers further improved serial IMPLY adders in terms of speed, energy, and robustness against non-idealities [37,38]. As indicated by a recent paper by Liu and Chu, the synthesis and optimization methods for implication logic have reached a high level of maturity [39].

2.2.2. Memristor-Aided Logic (MAGIC)

In 2014, Kvatinsky et al. [40] proposed a method that does not require a complicated structure and enables stable evaluation of the gate function to solve the high computational complexity of IMPLY logic, shown in Figure 2.

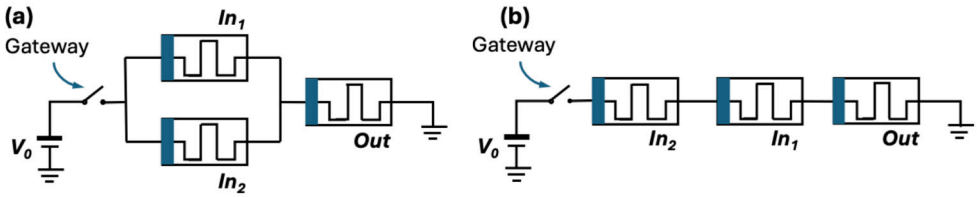

Figure 2. (**a**) OR gate, (**b**) AND gate, adapted with permission from [40].

Unlike the IMPLY logic gate, the MAGIC gate requires separate input and output memristors and only uses one applied voltage to control the logic gate. The value of each input memristor represents the initial input state, and the output memristor represents the result of the final calculation. The MAGIC gate is divided into two stages. In the first stage, the output memristor is initialised to the 1 logic state (low-resistance state), and the input value is applied to the input memristor. In the second stage, a voltage V_0 is used to flow through the entire logic gate, and the voltage division between the output memristor and the input memristor determines the final logic state of the gate.

In 2016, Talati et al. [41] proposed an adder based on a crossbar MAGIC gate. As mentioned in paper [40] above, only MAGIC gates with NOT and NOR logic can be arranged into crossbar arrays. Hence, Talati et al. reconstructed the calculation formulas for the Sum and Carry of the adder, using only NOT and NOR operations to execute the calculations. In 2022, Jha, C.K. et al. [42] discovered the most suitable adder architecture for crossbar by combining the adder and crossbar arrays of various architectures. They compared seven adder architectures: Ripple Carry (RC), Carry-Lookahead (CL), Ladner-Fischer (LF), Kogge-Stone (KS), Brent-Kung (BK), Carry Skip (CK), and Serial Prefix (SE). Among these architectures, the SE adder demonstrates the most efficient computation in terms of requiring the fewest gates and the lowest latency within the traditional crossbar architecture.

An important intermediate step in the development of MAGIC was its experimental validation. In particular, in 2020, Hoffer et al. showed that the robustness of MAGIC depends strongly on the set-to-reset threshold ratio of the underlying memristor technology, and proposed modified MAGIC gates for devices that do not satisfy the conventional assumption [16]. Furthermore, in 2024, A. Bende et al. experimentally verified MAGIC OR and NOT in a 1T1R TaOx RRAM crossbar, pointing out that the initialization stage dominates the total energy consumption [43]. More recently, the research focus has shifted toward architecture-level and practical implementation issues. In 2025, Nabipour et al. performed a unified benchmarking of multi-bit MAGIC-ReRAM arithmetic circuits and showed that parallel mapping can greatly reduce latency in crossbar-based adders and multipliers [44]. In 2026, Lalchhandama et al. further examined the practicality of MAGIC NOR in 1T1M crossbar arrays, emphasizing sneak-path-free implementation constraints and scalable mapping strategies for in-memory computing [45].

2.3. Non-Stateful

In contrast to stateful logic applications, memristors have also been shown to be effective in traditionally performing logic operations, e.g., where the logic value is encoded as a voltage/current level. To this end, work on both memristor exclusive architecture and a hybrid of memristor and traditional MOSFET/CMOS architecture is reported.

2.3.1. Memristor-Ratioed Logic (MRL)

Unlike the mechanisms of IMPLY and MAGIC gates based on the principle of resistance, their output signals are the states of memristors and require a read voltage for detection. In contrast, the output results of MRL are voltage values and do not need additional output components. In 2012, Kvatinsky et al. [12] proposed a hybrid circuit structure of memristor and CMOS, called MRL, to reduce the number of elements compared to pure CMOS. They suggested that we could utilise CMOS for constructing the inverter, while the OR gate and AND gate are implemented using memristors (as depicted in Figure 3). In the case of OR gates and AND gates, if the input voltage is all logic 0 or all logic 1, the memristor inside the gate circuit will have no current flowing through it, and V out is equal to Vin. When the two inputs are different, for an AND gate, the memristor closer to logic 0 is turned on, and the output is 0; for an OR gate, the memristor closer to logic 1 is turned

on, and the output is 1. However, because passive memristor networks cannot provide signal restoration or robust inversion, MRL relies on standard CMOS inverters to achieve logical completeness (as depicted in Figure 3) [12,46].

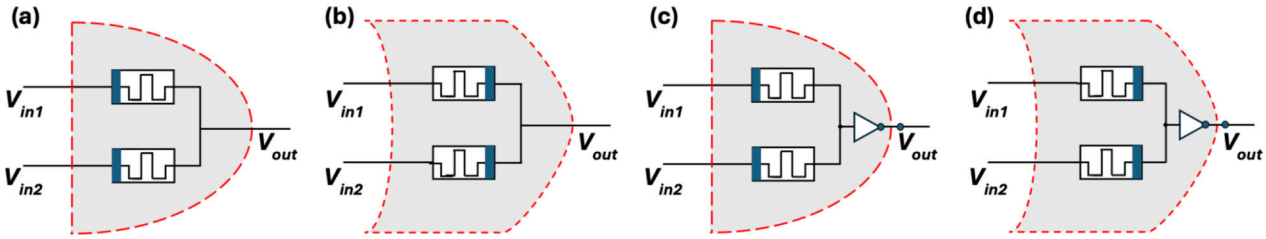

Figure 3. Schematic of MRL (**a**) AND gate, (**b**) OR gate, (**c**) NAND gate, and (**d**) NOR gate, adapted with permission from [46].

Building upon these fundamental gates, subsequent research expanded the logical completeness and application scope of the MRL architecture. X. Xu et al. (2017) [47] proposed hybrid memristor-MOS XOR and XNOR gates based on the MRL architecture, further enhancing the logical completeness of MRL. N. Mirzaie et al. (2018) [48] suggested that instead of using the crossbar structure, we could build the logic gate directly on the MRL structure and then use the logic gate to build the full adder. This can be applied in the three-dimensional pipeline ADC utilizing TSV/design optimization, where the MRL occupies 67.6% and 75% less silicon area. In 2019, Ali et al. [49], based on a crossbar array, proposed a novel adder using MRL to construct logic gates.

In the same year, X. Yang et al. [50] proposed a pure memristor structure to build an XOR gate (4M). Furthermore, to achieve compatibility between memristors and CMOS circuits, the 1T-4M and 3T-4M models were designed, as depicted in Figure 4, and the output difference voltage signal of the memristor module drives the MOS to output logic 1 or logic 0. They also demonstrated a more general implementation of the AND and OR logic operations in terms of the MIN and MAX operations, respectively, over the post-algebraic [51]. These operations are the analogue equivalents of the corresponding digital operations.

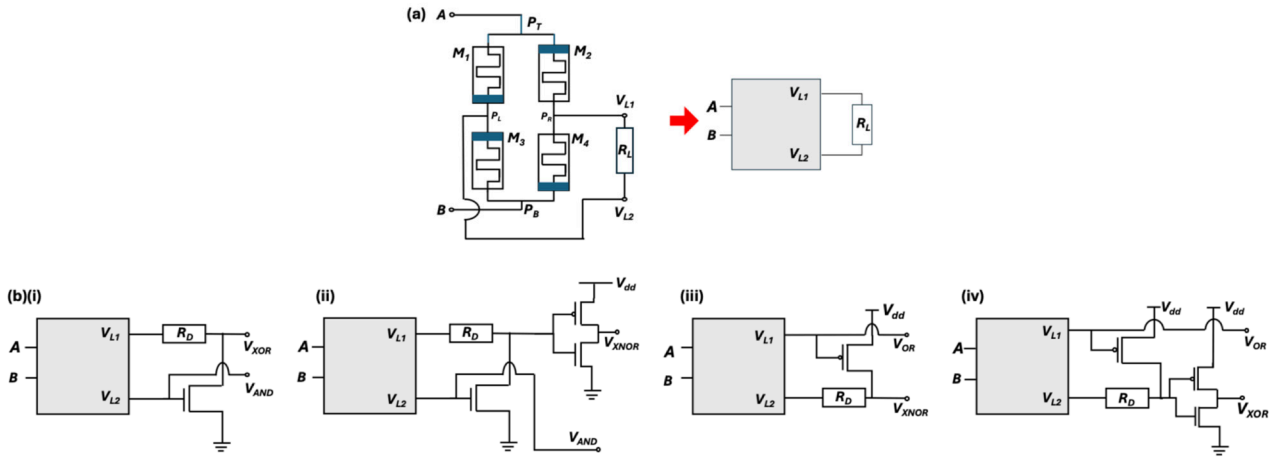

Figure 4. (**a**). Pure memristor XOR structure; (**b**) (**i**) unbuffered (weak) 1T-4M XOR, (**ii**) buffered 3T-4M XNOR, (**iii**) unbuffered (weak) 1T-4M XNOR, and (**iv**) buffered 3T-4M XOR, adapted with permission from [50].

2.3.2. Scouting Logic (SL)

In 2017, Xie L et al. [52] proposed a novel logic design architecture called Scouting Logic, as shown in Figure 5. The current through the equivalent gate input resistance

or the voltage drop over it is compared with a reference signal. The proposed design is implemented using two types of sense amplifiers (SA): current- and voltage-based. In this logic structure, the reference voltage or current can be flexibly adjusted so that the same memristor circuit presents a different logic gate operation.

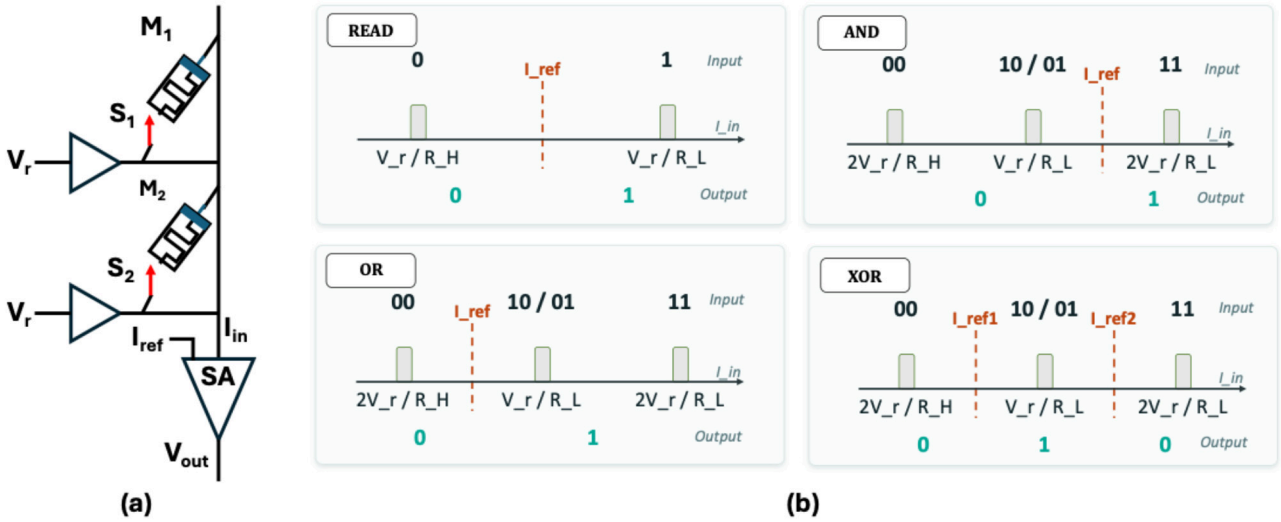

Figure 5. Scouting Logic architecture (**a**) memory, and (**b**) references of primitive operations, adapted with permission from [52].

Its core structure and operating principle use a two-input logic gate using only memristor reads, without any state switching (Figure 5a). It depicts two input memristors, whose resistive states encode the logic inputs (0/1), connected such that a read voltage (Vr) produces an input-dependent current Iin = Vr/Req (where Req is the equivalent resistance for each input pattern, like 00, 01, 10, 11), which flows into an SA. A reference branch provides a fixed reference current Iref, and the SA compares Iin against Iref to output logic 1 or 0, enabling the same hardware to realise different functions (AND, OR, XOR, etc.) in a single read step by appropriately selecting the reference and memristor state mapping [52]. Thereafter, the reference current selection strategy is used to realise different two-input Scouting Logic (SL) gates (OR, AND, XOR) using the current-sensing amplifier (CSA) (Figure 5b), where the Iin depends on the Req of the two parallel input memristors (M1//M2), yielding three distinct current levels: 2Vr/RH (for 00 input, both high-resistance RH), Vr/RL (for 10/01 inputs, one low-resistance RL), and 2Vr/RL (for 11 input, both RL). By positioning the Iref between these levels or using dual references straddling Vr/RL for XOR (output 1 on 10/01), the SA comparison produces the desired logic function in a single read step without memristor switching [52]. This architecture is useful for embedded non-volatile memory elements, such as memristor circuits for computing-in-memory (CIM). The SA in scout logic is usually a variety of ADCs that convert the analogue output of multilevel cell [53] or single-level cell [54] into a digital signal in an analogue CIM. For example, different reference memristors can be used to determine whether a memristor in a memristor array is logic 1 or logic 0 [55].

3. Challenges and Prospects

Although the crossbar array is the most feasible configuration to achieve an ultra-high-density chip [56], it has several limitations in constructing logic gates. Specifically, only NOT gates and NOR gates can be integrated into the crossbar array. Furthermore, integrating the inverter built into CMOS in the logic gate of the MRL architecture into the crossbar proves to be challenging and has limitations [49]. As a result, only part of the logic

gate can be implemented, necessitating the conversion of the Sum and Carry formulas of the adder, leading to additional processing steps and increased time consumption. Despite its simplicity and efficiency, the MAGIC architecture exhibits limitations in supporting multilevel cascaded logic and large fan-out circuits, which may affect its applicability in more complex logic systems [57]. The IMPLY architecture requires a considerably large number of steps to implement the adder operation gradually, consuming significantly more time compared to the traditional CMOS structure adder.

Hybrid memristor–CMOS/transistor gates could reduce silicon area and power, but at the cost of added design complexity, device-level non-idealities, and often with tighter noise margin constraints [58]. Hybrid gates can replace multi-transistor stacks (e.g., 6 T AND, 10 T XOR) with small memristor networks plus just a few CMOS devices, significantly reducing the number of MOSFETs on silicon. Table 1 compares memristor–transistor hybrid and pure transistor logic gate designs proposed in the literature. The comparisons summarised in this entry are extracted from the original literature and were not re-simulated under a unified SPICE or Verilog-A testbench. Memristors can be manufactured down to the nanoscale at the BEOL process and placed above CMOS [59]. In the specific context of heterogeneous FPGA building blocks, Farooq et al. [58] reported that hybrid memristor–transistor blocks can be 25–60% smaller than transistor-only blocks. Nevertheless, the fabrication of these hybrid gate circuits is often limited by the lithography layers and interposer complexity, which could require extra periphery and eventually reduce the optimum achievable theoretical area density.

Table 1. The typical two-input logic gate designs proposed in the literature.

Gate Design	Memristor Count	Transistor Count	Ref
AND	0	6	[60]
	2	0	[61,62]
	2–3	4–6	[63]
NAND	0	8–10	[60]
	2	2	[61,62]
	3	6–8	[64]
OR	0	6	[60]
	2	0	[61]
	2	2	[62]
	3	6–8	[64]
NOR	0	4	[60]
	2	2	[61,62]
	3	6–8	[64]
XOR	0	8–10	[60]
	2	2	[46]
	4	2	[62]
	6	2	[65,66]
	4	3	[47]
	1	4	[67]
	6	4	[61,68]
XNOR	2	2	[46]
	≥4	2	[61]
	3	3	[69]
	4	3	[47]
	4	4	[62]

Making memristor networks have logic functions reduces dynamic switching activity in CMOS, and thus can save more power, especially in larger combinational blocks and CIM arrays [61,70]. Moreover, memristors are non-volatile memories that do not require static and standby power, since they retain state without refresh, and the crossbar arrays can be powered down when idle [59,71]. However, hybrid gates often suffer from degraded logic levels and reduced noise margins: memristor voltage dividers or threshold networks can produce intermediate or attenuated levels, so extra CMOS buffers or inverters are frequently inserted to restore full swings, which adds area, delay, and some static power back into the design [62,67]. The resistive nature and parasitics of memristors and crossbars can limit speed due to the RC time constants, meaning that hybrid gates are often not faster than well-optimised CMOS [71–73].

Hybrid gates enable a more straightforward application of weighted threshold logic, compact multi-input gates, and logic-in-memory operations where the same memristor array serves as both storage and computation unit, whereas these operations are difficult or expensive in pure CMOS architecture [64,71]. For example, memristor–CMOS threshold and MeMOS designs can implement n-input AND/OR or XOR/XNOR functions with fewer transistors than equivalent pure CMOS designs, while also allowing weights to be reprogrammed, enabling logic, routing, or even neuromorphic tasks within the same unit [61,64,67]. The problem, however, is that such reconfigurability complicates testing and verification; logic correctness now depends on programmed resistance values, which make the circuits very sensitive to resistance drift, device-to-device and cycle-to-cycle variation, and programming noise; thus, calibration, error-tolerant architectures, or redundancy are often needed, which increase transistor count and overhead [59,71,72]. Memristors with high ON/OFF ratios and long retention could help make hybrid logic more robust [6,58,71].

Recent works highlight the implementation of crossbar; although lab-scale prototypes and small logic blocks are promising, scaling to high-yield is still hindered by sneak-path current issues [72]. Fortunately, instead of using a transistor as a select device, a diode selector can be used to solve the sneak path in the crossbar, maintaining the feature size (4F2) [56]. The prospect of memristor–transistor hybrids becomes more evident when logic and memory need to be tightly coupled (e.g., in memory computing [59], neuromorphic synapses [74] and reconfigurable systems with compact arithmetic like CLAs or ECC blocks) [59], where reductions in CMOS count and on-chip interconnect can outweigh the cost of extra circuitry and variability management, such as using extra buffering [70,72]. Thus, the hybrid designs bring notable benefits due to these unique features—nonvolatility, scalability, and analogue programmability—that can be easily exploited at the circuit or system level.

Author Contributions: Conceptualization, Z.L. and F.M.S.; methodology, Z.L. and A.J.; validation, Z.L. and A.J.; formal analysis, Z.L.; investigation, Z.L.; resources, F.M.S.; data curation, Z.L.; writing—original draft preparation, Z.L.; writing—review and editing, F.M.S., A.J., Z.H., S.C. and V.R.; visualization, Z.L. and G.X.; supervision, F.M.S.; project administration, F.M.S.; funding acquisition, F.M.S. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by MSCA EC Grant Agreement No. 224 (101029535-MENESIS), Royal Society Grant RG/R2/232206-SiMSANeC, and UKRI Engineering and Physical Sciences Research Council Grant No. UKRI3222-SPICA.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: No new data were created or analyzed in this study. Data sharing is not applicable to this article.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

ADC	Analogue-to-Digital Converter
CIM	Computing-In-Memory
IMPLY	Material Implication
MAGIC	Memristor-Aided Logic
MRL	Memristor-Ratioed Logic
SL	Scouting Logic
LRS	Low-Resistance State
HRS	High-Resistance State
SA	Sense Amplifiers
RC	Ripple Carry
CL	Carry–Lookahead
LF	Ladner–Fischer
KS	Kogge-Stone
BK	Brent–Kung
CK	Carry Skip
SE	Serial Prefix
CSA	Current-Sensing Amplifier

References

1. Ye, J.; Wang, W.; Shi, C.; Cui, X.; Qu, W.; Lin, P.; Yu, X.; Cheng, R.; Wu, H.; Chen, B. An overhead-reduced, efficient, fully analog neural-network computing hardware. *Sci. Adv.* **2025**, *11*, eadv7555. [CrossRef]
2. Moore, G.E. Cramming more components onto integrated circuits. *Proc. IEEE* **1998**, *86*, 82–85. [CrossRef]
3. Dennard, R.H.; Gaensslen, F.H.; Hwa-Nien, Y.U.; Leo Rideout, V.; Bassous, E.; Leblanc, A.R. Design of Ion-Implanted MOSFETs with Very Small Physical Dimensions. *Proc. IEEE* **1999**, *87*, 668–678. [CrossRef]
4. Di Ventra, M.; Pershin, Y.V. The parallel approach. *Nat. Phys.* **2013**, *9*, 200–202. [CrossRef]
5. Sze, V.; Chen, Y.H.; Yang, T.J.; Emer, J.S. Efficient Processing of Deep Neural Networks: A Tutorial and Survey. *Proc. IEEE* **2017**, *105*, 2295–2329. [CrossRef]
6. Xiao, Y.; Jiang, B.; Zhang, Z.; Ke, S.; Jin, Y.; Wen, X.; Ye, C. A review of memristor: Material and structure design, device performance, applications and prospects. *Sci. Technol. Adv. Mater.* **2023**, *24*, 2162323. [CrossRef]
7. Zhao, Q.-T.; Han, Y.; Han, H.-C.; Schreiber, L.R.; Lee, T.-E.; Chiang, H.-L.; Radu, I.; Enz, C.; Grützmacher, D.; Stampfer, C.; et al. Ultra-low-power cryogenic complementary metal oxide semiconductor technology. *Nat. Rev. Electr. Eng.* **2025**, *2*, 277–290. [CrossRef]
8. Prinzie, J.; Simanjuntak, F.M.; Leroux, P.; Prodromakis, T. Low-power electronic technologies for harsh radiation environments. *Nat. Electron.* **2021**, *4*, 243–253. [CrossRef]
9. Chua, L. Memristor-The missing circuit element. *IEEE Trans. Circuit Theory* **1971**, *18*, 507–519. [CrossRef]
10. Strukov, D.B.; Snider, G.S.; Stewart, D.R.; Williams, R.S. The missing memristor found. *Nature* **2008**, *453*, 80–83. [CrossRef]
11. Mannocci, P.; Farronato, M.; Lepri, N.; Cattaneo, L.; Glukhov, A.; Sun, Z.; Ielmini, D. In-memory computing with emerging memory devices: Status and outlook. *APL Mach. Learn.* **2023**, *1*, 010902. [CrossRef]
12. Kvatinsky, S.; Wald, N.; Satat, G.; Kolodny, A.; Weiser, U.C.; Friedman, E.G. MRL-Memristor Ratioed Logic. In Proceedings of the 2012 13th International Workshop on Cellular Nanoscale Networks and Their Applications, Turin, Italy, 29–31 August 2012; pp. 1–6.
13. Borghetti, J.; Snider, G.S.; Kuekes, P.J.; Yang, J.J.; Stewart, D.R.; Williams, R.S. Memristive switches enable stateful logic operations via material implication. *Nature* **2010**, *464*, 873–876. [CrossRef]
14. Wang, Z.; Li, Y.; Su, Y.; Zhou, Y.; Cheng, L.; Chang, T.-C.; Xue, K.-H.; Sze, S.M.; Miao, X. Efficient Implementation of Boolean and Full-Adder Functions with 1T1R RRAMs for Beyond Von Neumann In-Memory Computing. *IEEE Trans. Electron Devices* **2018**, *65*, 4659–4666. [CrossRef]
15. Cheng, L.; Li, Y.; Yin, K.; Hu, S.; Su, Y.; Jin, M.; Wang, Z.; Chang, T.; Miao, X. Functional Demonstration of a Memristive Arithmetic Logic Unit (MemALU) for In-Memory Computing. *Adv. Funct. Mater.* **2019**, *29*, 1905660. [CrossRef]

16. Hoffer, B.; Rana, V.; Menzel, S.; Waser, R. Experimental Demonstration of Memristor-Aided Logic (MAGIC) Using Valence Change Memory (VCM). *IEEE Trans. Electron Devices* **2020**, *67*, 3115–3122. [CrossRef]
17. Song, Y.; Wang, X.; Wu, Q.; Yang, F.; Wang, C.; Wang, M. Reconfigurable and Efficient Implementation of 16 Boolean Logics and Full-Adder Functions with Memristor Crossbar for Beyond von Neumann In-Memory Computing. *Adv. Sci.* **2022**, *9*, 2200036. [CrossRef]
18. Youn, S.; Lee, J.; Kim, S.; Park, J.; Kim, K.; Kim, H. Programmable Threshold Logic Implementations in a Memristor Crossbar Array. *Nano Lett.* **2024**, *24*, 3581–3589. [CrossRef]
19. Biolek, Z.; Biolek, D.; Biolková, V. SPICE model of memristor with nonlinear dopant drift. *Radioengineering* **2009**, *18*, 210–214.
20. Pershin, Y.V.; Ventra, M.D.I. SPICE model of memristive devices with threshold. *arXiv* **2012**, arXiv:1204.2600.
21. Kvatinsky, S.; Ramadan, M.; Friedman, E.G.; Kolodny, A. VTEAM: A General Model for Voltage-Controlled Memristors. *IEEE Trans. Circuits Syst. II Express Briefs* **2015**, *62*, 786–790. [CrossRef]
22. Kim, S.; Choi, S.; Lu, W. Comprehensive Physical Model of Dynamic Resistive Switching in an Oxide Memristor. *ACS Nano* **2014**, *8*, 2369–2376. [CrossRef]
23. Ielmini, D.; Milo, V. Physics-based modeling approaches of resistive switching devices for memory and in-memory computing applications. *J. Comput. Electron.* **2017**, *16*, 1121–1143. [CrossRef]
24. Chandrasekaran, S.; Hua, C.; Gao, S.; Xue, G.; Wen, J.; Jabir, A.; Georgiadou, D.G.; Hajiabadi, Z.; Liao, Z.; Simanjuntak, F.M. Memristor-Based Logic Circuits: Gate Designs and Cycle-To-Cycle Reliability. *IET Circuits Devices Syst.* **2026**, *2026*, 9915488. [CrossRef]
25. Li, H.; Jiang, Z.; Huang, P.; Wu, Y.; Chen, H.-Y.; Gao, B.; Liu, X.Y.; Kang, J.F.; Wong, H.-S.P. Variation-Aware, Reliability-Emphasized Design and Optimization of RRAM using SPICE Model. In Proceedings of the Design, Automation & Test in Europe Conference & Exhibition (DATE), Grenoble, France, 9–13 March 2015.
26. Reuben, J.; Biglari, M.; Fey, D. Incorporating Variability of Resistive RAM in Circuit Simulations Using the Stanford–PKU Model. *IEEE Trans. Nanotechnol.* **2020**, *19*, 508–518. [CrossRef]
27. Bengel, C.; Siemon, A.; Cuppers, F.; Hoffmann-Eifert, S.; Hardtdegen, A.; von Witzleben, M.; Hellmich, L.; Waser, R.; Menzel, S. Variability-Aware Modeling of Filamentary Oxide-Based Bipolar Resistive Switching Cells Using SPICE Level Compact Models. *IEEE Trans. Circuits Syst. I Regul. Pap.* **2020**, *67*, 4618–4630. [CrossRef]
28. Zanotti, T.; Pavan, P.; Puglisi, F.M. Multi-Input Logic-in-Memory for Ultra-Low Power Non-Von Neumann Computing. *Micromachines* **2021**, *12*, 1243. [CrossRef]
29. Lehtonen, E.; Laiho, M. Stateful implication logic with memristors. In *2009 IEEE/ACM International Symposium on Nanoscale Architectures*; IEEE: New York, NY, USA, 2009; pp. 33–36.
30. Kvatinsky, S.; Satat, G.; Wald, N.; Friedman, E.G.; Kolodny, A.; Weiser, U.C. Memristor-Based Material Implication (IMPLY) Logic: Design Principles and Methodologies. *IEEE Trans. Very Large Scale Integr. Syst.* **2014**, *22*, 2054–2066. [CrossRef]
31. Rohani, S.G.; Taherinejad, N. An improved algorithm for IMPLY logic based memristive Full-adder. In Proceedings of the Canadian Conference on Electrical and Computer Engineering, Windsor, ON, Canada, 30 April–3 May 2017.
32. Karimi, A.; Rezai, A. Novel design for a memristor-based full adder using a new IMPLY logic approach. *J. Comput. Electron.* **2018**, *17*, 1303–1314. [CrossRef]
33. Rohani, S.G.; Taherinejad, N.; Radakovits, D. A Semiparallel Full-Adder in IMPLY Logic. *IEEE Trans. Very Large Scale Integr. Syst.* **2020**, *28*, 297–301. [CrossRef]
34. Radakovits, D.; Taherinejad, N.; Cai, M.; Member, S.; Delaroche, T.; Mirabbasi, S. A Memristive Multiplier Using Semi-Serial IMPLY-Based Adder. In *IEEE Transactions on Circuits and Systems I: Regular Papers*; IEEE: New York, NY, USA, 2020; Volume 67, pp. 1495–1506. [CrossRef]
35. Kaushik, N.; Member, G.S.; Bodapati, S. Imply-based high-speed conditional carry and carry select adders for in-memory computing. *IEEE Trans. Nanotechnol.* **2023**, *22*, 280–290. [CrossRef]
36. Fatemieh, S.E.; Reshadinezhad, M.R.; Taherinejad, N. Fast and Compact Serial IMPLY-Based Approximate Full Adders Applied. *IEEE J. Emerg. Sel. Top. Circuits Syst.* **2023**, *13*, 175–188. [CrossRef]
37. Seiler, F.; Taherinejad, N. An Improved Serial IMPLY Adder Algorithm for Efficient Neural Network Applications. In Proceedings of the 2025 IEEE 16th Latin America Symposium on Circuits and Systems (LASCAS), Bento Gonçalves, Brazil, 25–28 February 2025; Volume 1, pp. 1–5. [CrossRef]
38. Seiler, F.; Taherinejad, N. An Efficient Robust Serial IMPLY-based In-Memristor Adder. In Proceedings of the 2025 Cross-Disciplinary Conference on Memory-Centric Computing (CCMCC), Dresden, Germany, 8–10 October 2025; pp. 1–7. [CrossRef]
39. Liu, T.; Chu, Z. Implication logic synthesis and optimization methods for memristor-based logic circuits. *Microelectron. J.* **2025**, *157*, 106553. [CrossRef]
40. Kvatinsky, S.; Belousov, D.; Liman, S.; Satat, G.; Wald, N.; Friedman, E.G.; Kolodny, A.; Weiser, U.C. MAGIC-Memristor-aided logic. *IEEE Trans. Circuits Syst. II Express Briefs* **2014**, *61*, 895–899. [CrossRef]

41. Talati, N.; Gupta, S.; Mane, P.; Kvatinsky, S. Logic design within memristive memories using memristor-aided loGIC (MAGIC). *IEEE Trans. Nanotechnol.* **2016**, *15*, 635–650. [CrossRef]
42. Jha, C.K.; Mahzoon, A.; Drechsler, R. Investigating Various Adder Architectures for Digital In-Memory Computing Using MAGIC-based Memristor Design Style. In Proceedings of the 2022 IEEE International Conference on Emerging Electronics (ICEE), Bengaluru, India, 11–14 December 2022; pp. 1–4.
43. Bende, A.; Singh, S.; Jha, C.K.; Kempen, T.; Cüppers, F.; Bengel, C.; Zambanini, A.; Nielinger, D.; Patkar, S.; Drechsler, R.; et al. Experimental Validation of Memristor-Aided Logic Using 1T1R TaO x RRAM Crossbar Array. In Proceedings of the 2024 37th International Conference on VLSI Design and 2024 23rd International Conference on Embedded Systems (VLSID), Kolkata, India, 6–10 January 2024; pp. 565–570.
44. Nabipour, S.; Shirinzadeh, F.; Datta, K.; Kole, A.; Shirinzadeh, S.; Drechsler, R. Performance Evaluation of MAGIC-ReRAM Arithmetic Circuits for Low-Latency In-Memory Computing. In Proceedings of the 2025 IEEE Nordic Circuits and Systems Conference (NorCAS), Riga, Latvia, 28–29 October 2025; pp. 1–7. [CrossRef]
45. Lalchhandama, F.; Datta, K.; Drechsler, R.; Chakraborty, S.; Sengupta, I. Practicality of MAGIC NOR in a 1T1M Crossbar Array For In-Memory Computing. *J. Circuits Syst. Comput.* **2026**, *31*. [CrossRef]
46. Yr, A.; Das, S.; Trivedi, G. Design of Area-Efficient Memristor-Based XOR and XNOR Gates. In Proceedings of the 2025 35th International Conference Radioelektronika (RADIOELEKTRONIKA), Hnanice, Czech Republic, 12–14 May 2025; pp. 1–6.
47. Xu, X.; Cui, X.; Luo, M.; Lin, Q.; Luo, Y.; Zhou, Y. Design of hybrid memristor-MOS XOR and XNOR logic gates. In Proceedings of the 2017 International Conference on Electron Devices and Solid-State Circuits (EDSSC), Hsinchu, Taiwan, 18–20 October 2017; pp. 1–2.
48. Mirzaie, N.; Alzahmi, A.; Shamsi, H.; Byun, G.S. Three-Dimensional Pipeline ADC Utilizing TSV/Design Optimization and Memristor Ratioed Logic. *IEEE Trans. Very Large Scale Integr. Syst.* **2018**, *26*, 2619–2627. [CrossRef]
49. Ali, K.A.; Rizk, M.; Baghdadi, A.; Diguet, J.P.; Jomaah, J. MRL crossbar-based full adder design. In *2019 26th IEEE International Conference on Electronics, Circuits and Systems, ICECS 2019*; IEEE: New York, NY, USA, 2019; pp. 674–677.
50. Yang, X.; Adeyemo, A.; Bala, A.; Jabir, A. Novel techniques for memristive multifunction logic design. *Integration* **2019**, *65*, 219–230. [CrossRef]
51. Dubrova, E.; Jiang, Y.; Brayton, R.K. Minimization of Multiple-Valued Functions in Post Algebra. In Proceedings of the IWLS 2001 (International Workshop on Logic Synthesis), Lake Tahoe, CA, USA, 12–15 June 2001; Available online: https://people.kth.se/ dubrova/PAPERS/IWLS01.pdf (accessed on 6 April 2026).
52. Xie, L.; Nguyen, H.A.D.; Yu, J.; Kaichouhi, A.; Taouil, M.; Alfailakawi, M.; Hamdioui, S. Scouting Logic: A Novel Memristor-Based Logic Design for Resistive Computing. In Proceedings of the IEEE Computer Society Annual Symposium on VLSI, ISVLSI, Bochum, Germany, 3–5 July 2017.
53. Wang, D.; Xu, L.; Liu, S.; Li, Z.; Chen, Y.; He, W.; Li, X.; Sun, Y. TL-nvSRAM-CIM: Ultra-High-Density Three-Level ReRAM-Assisted Computing-in-nvSRAM with DC-Power Free Restore and Ternary MAC Operations. In Proceedings of the 2023 IEEE/ACM International Conference on Computer Aided Design (ICCAD), San Francisco, CA, USA, 29 October–2 November 2023; pp. 1–9.
54. Chen, W.H.; Li, K.X.; Lin, W.Y.; Hsu, K.H.; Li, P.Y.; Yang, C.H.; Xue, C.X.; Yang, E.Y.; Chen, Y.K.; Chang, Y.S.; et al. A 65nm 1Mb nonvolatile computing-in-memory ReRAM macro with sub-16ns multiply-and-accumulate for binary DNN AI edge processors. In *Digest of Technical Papers—IEEE International Solid-State Circuits Conference*; IEEE: New York, NY, USA, 2018; Volume 61.
55. Wang, X.; He, J.; Shao, K.; Zheng, J.; Tian, F.; Cheng, T.K.-T.; Tsui, C.-Y. ReSCIM: Variation-Resilient High Weight-Loading Bandwidth In-Memory Computation Based on Fine-Grained Hybrid Integration of Multi-Level ReRAM and SRAM Cells. In *Proceedings of the 43rd IEEE/ACM International Conference on Computer-Aided Design*; ACM: New York, NY, USA, 2024; pp. 1–9.
56. Salahuddin, S.; Ni, K.; Datta, S. The era of hyper-scaling in electronics. *Nat. Electron.* **2018**, *1*, 442–450. [CrossRef]
57. Liu, G.; Shen, S.; Jin, P.; Wang, G.; Liang, Y. Design of Memristor-Based Combinational Logic Circuits. *Circuits Syst. Signal Process.* **2021**, *40*, 5825–5846. [CrossRef]
58. Farooq, U.; Aslam, M.H.; Usman, M. On the comparison of memristor-transistor hybrid and transistor-only heterogeneous FPGAs. *J. King Saud Univ.-Comput. Inf. Sci.* **2019**, *31*, 514–527. [CrossRef]
59. Xia, Q.; Robinett, W.; Cumbie, M.W.; Banerjee, N.; Cardinali, T.J.; Yang, J.J.; Wu, W.; Li, X.; Tong, W.M.; Strukov, D.B.; et al. Memristor−CMOS Hybrid Integrated Circuits for Reconfigurable Logic. *Nano Lett.* **2009**, *9*, 3640–3645. [CrossRef]
60. Gajda, Z.; Sekanina, L. Reducing the number of transistors in digital circuits using gate-level evolutionary design. In *Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation*; ACM: New York, NY, USA, 2007; pp. 245–252.
61. Singh, T. Hybrid Memristor-CMOS (MeMOS) based Logic Gates and Adder Circuits. *arXiv* **2015**, arXiv:1506.06735.
62. Wan Mohamad Sharif, W.M.H.; Md Idros, M.F.; Mutalib Al-Junid, S.A.; Osman, F.N.; Abdul Razak, A.H.; Halim, A.K.; Adib Harun, M. Hybrid memristor-CMOS implementation of logic gates design using LTSpice. *Int. J. Electr. Comput. Eng.* **2021**, *11*, 2003. [CrossRef]

63. Gao, L.; Alibart, F.; Strukov, D.B. Programmable CMOS/Memristor Threshold Logic. *IEEE Trans. Nanotechnol.* **2013**, *12*, 115–119. [CrossRef]
64. Papandroulidakis, G.; Serb, A.; Khiat, A.; Merrett, G.V.; Prodromakis, T. Practical Implementation of Memristor-Based Threshold Logic Gates. *IEEE Trans. Circuits Syst. I Regul. Pap.* **2019**, *66*, 3041–3051. [CrossRef]
65. Guckert, L.; Swartzlander, E. Optimized memristor-based ripple carry adders. In *2016 50th Asilomar Conference on Signals, Systems and Computers*; IEEE: New York, NY, USA, 2016; pp. 1575–1579.
66. Khalid, M.; Mukhtar, S.; Siddique, M.J.; Ahmed, S.F. Memristor Based Full Adder Circuit for Better Performance. *Trans. Electr. Electron. Mater.* **2019**, *20*, 403–410. [CrossRef]
67. Zhou, Y.; Li, Y.; Xu, L.; Zhong, S.; Xu, R.; Miao, X. A hybrid memristor-CMOS XOR gate for nonvolatile logic computation. *Phys. Status Solidi* **2016**, *213*, 1050–1054. [CrossRef]
68. Emara, A.S.; Madian, A.H.; Amer, H.H.; Amer, S.H.; Abdelhalim, M.B. Testing of memristor ratioed logic (MRL) XOR gate. In Proceedings of the 2016 28th International Conference on Microelectronics (ICM), Giza, Egypt, 17–20 December 2016; pp. 181–184.
69. Singh, A. Memristor based XNOR for high speed area efficient 1-bit full adder. In Proceedings of the 2017 International Conference on Computing, Communication and Automation (ICCCA), Greater Noida, India, 5–6 May 2017; pp. 1549–1553.
70. Pokharia, M.; Prasad, K.; Hegde, R.; Mekie, J. Hybrid CMOS-Memristor Logic for Boosting the Power-Efficiency in Error Tolerant Applications. In *2024 37th International Conference on VLSI Design and 2024 23rd International Conference on Embedded Systems (VLSID)*; IEEE: New York, NY, USA, 2024; pp. 431–436.
71. Chen, Q.; Lu, L.; Meng, J.; Xu, M.; Wang, T. Advances of Emerging Memristors for In-Memory Computing Applications. *Research* **2025**, *8*, 0916. [CrossRef]
72. Kumar, P.; Zhu, K.; Gao, X.; Wang, S.-D.; Lanza, M.; Thakur, C.S. Hybrid architecture based on two-dimensional memristor crossbar array and CMOS integrated circuit for edge computing. *npj 2D Mater. Appl.* **2022**, *6*, 8. [CrossRef]
73. Cho, K.; Lee, S.-J.; Eshraghian, K. Memristor-CMOS logic and digital computational components. *Microelectron. J.* **2015**, *46*, 214–220. [CrossRef]
74. Lim, J.G.; Park, S.; Lee, S.M.; Jeong, Y.; Kim, J.; Lee, S.; Park, J.; Hwang, G.W.; Lee, K.-S.; Park, S.; et al. Hybrid CMOS-Memristor synapse circuits for implementing Ca ion-based plasticity model. *Sci. Rep.* **2024**, *14*, 17915. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.