2016 13th International Conference on Electrical Engineering, Computing Science and Automatic Control (CCE), Mexico, City. Mexico

Fully Digital Write-in Scheme for Multi-Bit Memristive Storage

Nima Taherinejad, Sai Manoj P. D., Michael Rathmair, and Axel Jantsch
Institute of Computer Technology
Faculty of Electrical Engineering and Information Technology
TU Wien
Gusshausstrasse 27-29, 1040 Vienna, Austria
Email: {nima.taherinejad, sai.dinakarrao, michael.rathmair, axel.jantsch}@tuwien.ac.at

***Abstract*—Memristors have been used in various applications, including single- and multi-bit storage units. The non-linear voltage-current relation in memristors is often seen as a problem, necessitating complex circuits and methods for a reliable write-in. In this paper, we take advantage of this phenomenon for storing more than one bit of information in a single memristor using digital bit streams. First, we demonstrate how two bits of information can be stored and read back from a single memristor unit. Then, we propose encoding schemes that can enhance the reliability of digitally writing two and three bits of data in a single memristor. To verify the reliability of this method for multi-bit data storage, we have run simulations based on the most prominent simulation models available.**

I. Introduction

Electrical engineers are quite familiar with circuits which include the three basic passive circuit elements - resistor, capacitor and inductor. However, for the first time, in 1971 Leon Chua proposed a fourth circuit element, describing the relation between charge (q) and flux (ϕ) [1]. The resistance of the so called Memristor device, depends on the total charge passed through the device [1], [2]. The device characteristics of the memristor open the floor for many new fields of application [3]–[7]. One of the most important applications is, as expected, using it as a memory element. For example, [8]–[10] and others have invested considerable effort in studying the design of memristor based memory units. The main advantage of using memristors is given by the fact that in a modern chip, the number of transistors required to store data (e.g. in an SRAM) has a significant -and increasing- impact on the total transistor count [11]. In consequence, implementation of new memory architectures by using memristors would decrease the total amount of device leakage current dramatically [9].

Since HP developed the first passive memristor in 2008 [12], there have been works on single-bit [8]–[10], [13] and multi-bit [14], [15] memory storages. The unique $\phi - q$ characteristic of memeristors however, leads to a nonlinear $v - i$ characteristic which makes it difficult to determine the pulse size for achieving a certain state [14], [15]. Therefore, it has been seen as a problem for multilevel memory designs [14], [15]. To overcome this problem, researchers have developed methods, some fairly complicated, which use analog circuits and op-amps for both reading and writing [14], [15]. Using op-amps and analog circuits for read-out seems to be inevitable [8], [14], [15]. However, as we will present, the aforementioned characteristic can be taken advantage of, to store more than one bit in memristors, using circuits which are compatible with digital designs. These circuits are hence less complex. Although digital bit streams have been used for setting synaptic values [16], the exact stored value and recovery of the stored value are not of high importance in such applications. In memory applications, on the other hand, this is of paramount importance. Therefore, in this paper we discuss the reliability of the storage and read-out, as well as how encoding can help improving these parameters.

The rest of this paper is organized as the following: In the next section, we briefly review the aforementioned unique characteristics of memristor and show how it can be taken advantage of, in order to store more than one bit in each memristor. In Section III, we present the digital writing method for storing two bits of data on a single memristor, as well as the read-out circuit. Then, in Section IV, we show the results of our simulations which confirm the feasibility of the proposed approach. In Section V, we improve the reliability of the storage, and the number of stored bits, using different proposed encoding schemes. Finally, we conclude the paper in Section VI, which entails our plan for future works as well.

II. Theorem

It is a well established phenomenon in the literature, that the resistance of a memristor depends on the charge flown through it [1], [17]. This can be modeled in various ways, one of the most prominent of which is [17]:

$$R(q(t)) = R_{\text{off}} + \frac{R_{\text{on}} - R_{\text{off}}}{e^{-4k_m(q(t)+q_0)} + 1}. \qquad (1)$$

where R_{off} and R_{on} are the maximum and minimum resistance of the memristor, $q(t)$ is the charge flown through the memristor with the initial value of q_0. Finally, k_m is a constant which represents physical characteristics of the device such as doping of the semiconductor and its size.

According to (1), not only the resistance, but also changes of the resistance in a memristor due to identical voltage pulses depend on its current state [14], [18]. This characteristic seen as a hardship [14], [15] can be turned into an opportunity based on the following theorem.

Theorem: *Distinct output resistances correspond to distinct input patterns, which allows storage of more than one bit in a single memristor.*

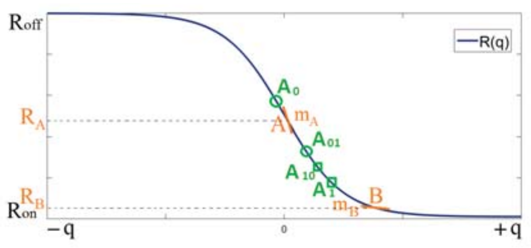

Fig. 1. Resistance of a memristor as a function of the charge stored in it.

Short Proof[1]***:*** Assume two distinct states A and B, seen in Fig. 1, with unequal resistance of R_x, where $x \subset \{A,B\}$. Now, we are interested in finding out the ratio of resistance change at these two points, due to the application of an input voltage pulse with the amplitude of V and width of T. Based on Ohm's law we have:

$$V = R(q(t))\frac{dq}{dt} \tag{2}$$

where $R(q(t))$ of the memristor is given by (1). Solving this equation is rather complicated and out of the scope of this paper. Therefore, in a simplified manner, we estimate the changes of resistance at each point by its slope, m_x. In other words, in the vicinity of point x, we estimate (1) by:

$$R(q) = -m_x q + R_x. \tag{3}$$

By substituting (3) into (2), and integrating over the period, T, we have [18]

$$\begin{aligned} &\rightarrow \quad V dt = R dq = (-m_x q + R_x) dq \\ &\rightarrow \quad \int_0^T V dt = \int_0^{dq_x} (-m_x q + R_x) dq \\ &\rightarrow \quad VT = \tfrac{-m_x}{2}(dq_x)^2 + R_x dq_x. \end{aligned} \tag{4}$$

Solving (4) we obtain

$$dq_x = \frac{R_x}{m_x} K_x, \tag{5}$$

where

$$K_x = 1 + \sqrt{1 - \frac{2m_x VT}{R_x^2}}. \tag{6}$$

On the other hand, we can easily infer from (3) that $\frac{dR_x}{dq_x} = -m_x$. Based on which, by calculating the ratio of resistance change due to charge change at point A and B is

$$\frac{dR_\text{A}}{dq_\text{A}} / \frac{dR_\text{B}}{dq_\text{B}} = \frac{m_\text{A}}{m_\text{B}} \rightarrow \frac{dR_\text{A}}{dR_\text{B}} = \frac{R_\text{A} K_\text{A}}{R_\text{B} K_\text{B}}. \tag{7}$$

This shows how the change of resistance of memristors at different stages due to identical pulses depends on its state (resistance) at the time. This ratio could be further simplified and approximated to the resistance ratio only [18].

Now, assuming that a positive pulse presents "1" and a negative pulse presents "0", if a memristor is fed by a positive input pulse followed by a negative input pulse ("10"), it will reach a different state ("A_{10}"), compared to the case where the negative pulse is applied before the positive pulse ("01" and hence A_{01}). The rationale behind it, is that the resistance change due to an initial negative pulse (A to A_0) is different compared to the resistance change due to the same pulse, once the state of memristor has changed due to a previous positive pulse (A_1 to A_{10}). The same holds for changes due to positive pulses in two different states (A_0 to A_{01} versus A to A_1). Therefore, we can conclude that, since the states of the memristor after application of "01" and "10" are different, based on the state of the memristor, its original input can be retrieved [18]. This will be verified in the next section.

[1]The full proof is provided in [18]. We kindly ask the readers to refer to that paper for more information on details.

TABLE I. Data to be stored on the Memristor, resulting states, and expected un-encoded read-outs.

Data	R_mem (kΩ)	Exp. Read-out
"00"	5.213	"000"
"01"	5.108	"001"
"10"	4.890	"011"
"11"	4.778	"111"

III. Write and Read

Based on the aforementioned concept, in this section we present our read and write method and simulate them in Section IV, to test the feasibility of successfully recovering a data value which was stored through digital streaming. For the array architecture, isolation and access to the memory cell, a 1T1R architecture similar to [15] is assumed. Since, the op-amp (comparator) is the crucial part of the circuit determining the feasibility and reliability of this implementation, we have used a model of an existing off-the-shelf op-amp, namely LT1012, which takes into account non-ideal characteristics such as off-set.

A. Write-in Method

As previously discussed in order to distinguish the "1" input and "0" input, respectively, positive and negative pulses are used, which can be implemented through simple switches to reference voltages. This eliminates the need for the complex circuit proposed in [14] or the DAC used in [15]. The magnitude of these pulses for the current experiment is 0.5V and the pulse-widths are 10ms. The inputs were applied based on the most-significant-bit first and the least-significant-bit last.

Table I shows the data to be stored and the resulting state of the memristor (resistance) due to these inputs. This table shows that, as expected, different input data patterns lead to different states for the memristor. Similar to [15], the thermometer read-out codes at this column can be later converted to the normal binary representation (similar to the left-most column) using simple digital logic circuits. Only one instance of such a circuit will be necessary for each memory block containing hundreds to thousands of memristor cells. Therefore, the cost of the additional hardware is negligible.

B. Read-out Method

For retrieving the stored data, [8] uses an op-amp based circuit, [15] uses an ADC, and [14] uses a circuit inspired by ADCs. In this paper, as shown in Fig. 2, we also use a circuit inspired by flash ADCs. In this circuit, after the inputs are applied and the data is stored in the memristor, the input source is disconnected. Then, after applying a controlled

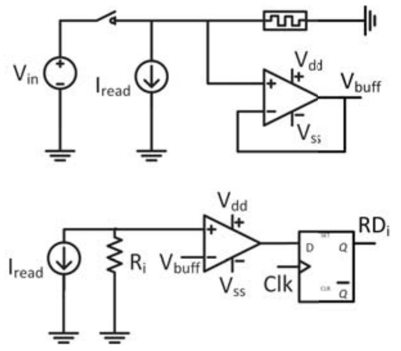

Fig. 2. For reading out, V_{buff} is fed to three comparators connected to D Flip-flops which provide RD_i bits (for simplicity, only one set is depicted).

current pulse (here a rectangular pulse with the magnitude of 1μA and width of 1ms), the resulting voltage across the memristor is buffered and fed to three comparator circuits (only one of which is shown at the bottom of Fig. 2, for simplicity). The voltage across the memristor is then compared with the voltage produced by applying a similar current pulse to predetermined resistance values. If the voltage across the predetermined resistors (R_i) is greater than the voltage read from the memristor, the comparator will flip to high output and a "1" will be stored in the respective flip-flop output (RD_i). This implies that the resistance of the cell is below the respective resistor, R_i.

We note that the read-out current pulse is chosen to be significantly smaller than input pulses, so that it would have a smaller effect on the state of the memristor. Nevertheless, however small this effect is, it needs to be compensated for. Therefore, similar to [8], [15], by applying another controlled pulse in the reverse direction, this effect can be compensated for. The advantage of using current pulses, compared to the voltage pulses used in [8], [15], is that the effect of positive and negative current pulses on the resistance change will be similar whereas as proved in Section II, it is not the case for voltage pulses. That is, a negative voltage pulse, due to inherent characteristics of the memristor and its non-linear voltage-current relationship, cannot precisely compensate a positive voltage pulse and will leave residual extra charges (positive or negative). Therefore, state of memristor needs to be checked and more often refreshed to compensate the residual charges as well.

Using current sources, as proposed here, the complex method of read-out effect elimination is reduced to an automatic compensation through a revers current pulse, which is considerably simpler than the one used in [14]. Automatic refreshing may lead to extra power consumption, however, it eliminates the state-read operation or counting process for the refreshing proposed in [15]. Therefore, it is going to be simpler. We note that this technique may not prove efficient for all types of memristor technologies (e.g. for non-TiO_2 memristors), or scenarios (e.g. for power constrained applications). Therefore, in order to compensate for the destructive effect of uncompensated read-outs (for power critical applications or where this technique is not efficient), other scenarios, as discussed in [15], can be employed to compensate for the read-out effect.

IV. Simulations

In order to evaluate the practicality of the proposed approach in storing data in memristors, we have simulated the proposed circuit in LTspice. We have used the most prominent model (namely, Biolek's [17]) of TiO_2 memristors, described by (1). The following values were used for the model parameters (as given by the authors [17]); $R_{on} = 100\Omega$, $R_{off} = 10k\Omega$, $k_m = 10000$, and $R_{init} = 5k\Omega$. Input signals last 10ms and 1ms, respectively for input voltages and read-out current pulses. The amplitudes are ± 0.5V for "1" and "0" input pulses and -1μA for read-out current pulses. The value of R_i resistors used in this set of simulations are 5170, 5130, and 5010Ω for $i = 1,2,$ and 3 respectively.

Comparators and their precision play a crucial role in practical implementation of the read-out circuit, which determine to what level the small differences between states can be distinguished. To account for their non-idealities, we have used the model for an off-the-shelf op-amp, namely LT1012 (with 25μV maximum offset). This ascertains that the assumed values for parameters such as offset, are realistic and accounted for, i.e., this circuit can -in practice also- perform as expected.

Fig. 3 shows the result of simulation for a sample case, namely "01" input. In this figure, the blue curve shows the voltage across the memristor (namely V_{in}) and the black curve shows the RD_1 output. For simplicity, RD_2 and RD_3 which as expected remained zero during the whole simulation are not shown. As we can see, shortly after applying the read-out current at 3.1 secs (zoomed-in area) the output of RD_1 turns to "1".

This and other simulations for all outputs satisfied the expected outputs as shown in Table I, where the expected outputs represent "RD_3 & RD_2 & RD_1" combination. Once these values are obtained, different inputs are distinguished properly and with simple logic circuits these outputs can be turned into binary values once again. This proof of concept confirms the feasibility of digital streaming approach for storing, and successfully retrieving two bits of information on a single memristor.

V. Encoding and Three-Bit Storage

In this section, we present the impact of the range of the memristor and the encoding of data on the reliability of the read-out.

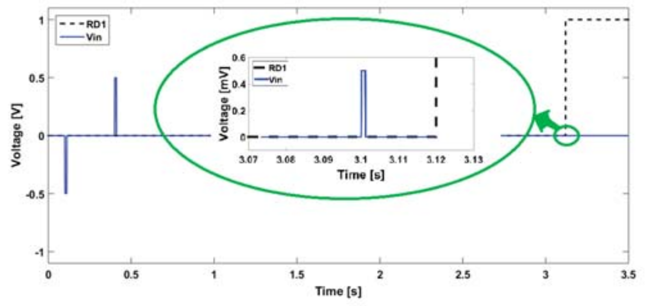

Fig. 3. Simulation results for "01" input. Shortly after applying the read-out current at 3.1s (zoomed in), the output of RD_1 flips to "1" representing the stored data. RD_0 and RD_2 remained zero and are not shown for simplicity.

A. Exploration of the Memristance Range

Trying to store three bits or more in a single memristor can have a negative impact on the reliability of the read-out, due to smaller differences between resistances at different states. Expanding the range of the memristor (between R_{on} and R_{off}), helps in reducing the relative sensitivity. Although, the relative difference in memristance between inputs with the same number of ones or zeros can be thus improved, this improvement is not necessarily proportional to the expansion of the range.

For example, in a memristor with a range of 100Ω-10kΩ, if the difference between "10" and "01" is 10Ω, in a memristor with a range of 1kΩ-100kΩ, the difference -according to our simulations- can be 17Ω. This is validated in our simulations, where two memristors are considered: memristor 1 (M_1) with the range of 0.1-10kΩ and memristor 2 (M_2) with the range of 0.5-100kΩ. The pulse width is set to 100ms, and the amplitudes are set to ±0.5V for "1" and "0" inputs. Initial state of memristor is assumed to be 5kΩ. Table II presents the memristance value for different 3-bit data patterns.

TABLE II. RESISTANCE FOR MEMRISTORS WITH DIFFERENT RANGES OF MEMRISTANCE. M_1: 0.1-10$k\Omega$ AND M_2: 0.5-100$k\Omega$.

Input	R_{M_1} (kΩ)	R_{M_2} (kΩ)
"000"	7.2453	47.2428
"001"	5.8991	28.9629
"010"	5.8997	28.9710
"011"	3.9089	5.0033
"100"	5.8990	28.9600
"101"	3.9078	4.9898
"110"	3.9088	5.0014
"111"	1.0396	0.5000

For instance, the memristance difference between "011" and "101" is 1Ω for M_1, whereas it is 14Ω for M_2. This indicates that inputs with the same number of ones or zeros can be differentiated easier in the memristor with a wider resistance range.

Although in this way the difference is increased, it is yet small and requires complex complementary circuits, such as the one used in [14], to set the values precisely. It also needs precise comparators, such as the one assumed in [15], to read the values out. To alleviate the problem of close resistance at different states, and in order to improve reliability without using analog write-in circuits or extra circuits with extensive complexity for the read-out operation, we propose two encoding schemes. These techniques lead to a significantly higher relative memristance difference between inputs with the same number of ones and zeros. As the number of memristors are not increased with encoding, the area of the memory remains constant, except for the addition of an encoder for the whole system (which is a memory block with hundreds to thousands of memory cells). Therefore, the area used for the encoder is considerably negligible.

B. Uniform Input Encoding

In the first scheme, we employ a simple encoding method. In this technique, we append the most significant bit (MSB) to the end of the input, as shown in Fig. 4. In other words, input "100" is encoded as "100**1**". The simulation setup is the same as stated values earlier in Section IV, with memristor M_1 chosen for simulation, and initial memristance set to 5kΩ. Table III and IV show the effect of encoding for 2-bit input and 3-bit input data, respectively.

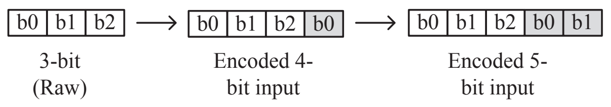

Fig. 4. Uniform Input Encoding of 3-bit input data to 4-bit and 5-bit data.

Table III presents the memristance for raw input data and encoded data. Originally, the memristance difference between "01" and "10" is nearly 1Ω, whereas when encoded to 3-bits, the relative memristance is nearly 1.9kΩ. As such, larger implementation errors (such as inaccuracy of reference values) and variations (of fabrication, implementation or between different cells) can be tolerated. In [15], a potential of up to 6% physical variation in fabrication process is observed and in [14], 10% memristance variation is assumed. However, given that the minimum distance between the memristance states after the proposed encoding is 19% (whereas it was 0.2% before), a flawless operation despite those variations is expected.

TABLE III. MEMRISTANCE FOR 2-BIT INPUT WITH UNIFORM ENCODING.

Input	Raw (kΩ)	Encoded value (kΩ)	Encoding Scheme
"00"	6.6384	7.2453	"00**0**"
"01"	5.0005	5.8998	"01**0**"
"10"	4.99955	3.9079	"10**1**"
"11"	2.5919	1.0396	"11**1**"

Similarly, the effect of encoding on 3-bit data is presented in Table IV. Here, we show the memristance when the 3-bit data is encoded to 4-bit, and 5-bit (N.B., in this case first *two* MSB bits are appended to the end of the data). From Table IV, we can see that the memristance difference between the two inputs of "001" and "100" for the raw (3-bit), encoded to 4-bit and encoded to 5-bit data, is respectively 0.1Ω, 1.64kΩ and 1.4kΩ. However, if we notice, the relative memristance difference between "001", and "010" when encoded to 4-bit, is <1Ω. This is because the appended bits for both are the same. This difference is larger between these two inputs in the case of 5-bit encoded data. Simply because the appended bits in this case are not the same anymore. Nonetheless, in some instances the values are still significantly close to each other which endangers a reliable retrieval of the stored data. Hence, further improvements as it follows are necessary.

C. Non-Uniform Encoding

To address the problem of close values and similar appended bits in the uniform encoding scenario, we propose another encoding scheme, shown in Table V. In this scenario, appendices were selected not based on the two MSBs, but rather such that the distance between all states is maximized. To this end, some bits are appended with two bits, some with one, and some with none.

As we see in Table V, using the new encoding scheme the minimum difference between all inputs is increased to 905.6Ω,

TABLE IV. MEMRISTANCE FOR 3-BIT INPUT WITH TWO DIFFERENT UNIFORM ENCODING SCHEMES.

Input	Raw (kΩ)	Encoded to 4-bit (kΩ)	Encoding Scheme	Encoded to 5-bit (kΩ)	Encoding Scheme
“000”	7.2453	7.7431	“000**0**”	8.1511	“000**00**”
“001”	5.8991	6.6384	“001**0**”	7.2451	“001**00**”
“010”	5.8997	6.6389	“010**0**”	5.9002	“010**01**”
“011”	3.9089	5.0004	“011**0**”	3.9095	“011**01**”
“100”	5.8990	4.9996	“100**1**”	5.8968	“100**10**”
“101”	3.9078	2.5906	“101**1**”	3.9073	“101**10**”
“110”	3.9088	2.5919	“110**1**”	1.0404	“110**11**”
“111”	1.0396	0.1000	“111**1**”	0.1000	“111**11**”

whereas it was previously only 0.5Ω in the 4-bit encoding scheme and 2.2Ω in the 5-bit encoding scheme shown in Table IV. This minimum distance being always larger than 11%, shows that potential physical and fabrication variations mentioned in [14], [15] can be tolerated by the system. Hence, using this non-uniform encoding scheme, three bits of data can be digitally streamed to a single memristor and the stored values can be reliably recovered using the read-out circuit presented in Section III.

VI. CONCLUSION

In this paper after a brief introduction we discussed the specific theory which allows us to use the non-linear behavior of memristors to our advantage for storing more than one bit of data in memristors via a simple bit streaming process. This process is compatible with digital circuits and in contrast to existing methods does not require analog or mixed-signal circuits and scenarios previously used for write-in process. Next, we presented the details of writing and reading-out methods which can be implemented using compact circuits with low complexity and requirement. The proposed method has smaller complexity compared to other works in the literature, and a smaller footprint and most importantly, it uses only digital circuits for writing in values in a memristor.

Further, we simulated the proposed methods successfully, thus demonstrating the feasibility of the proposed approach. The most crucial part of a practical implementation is the precision and real-world constraints of the op-amp used in the read-out circuit. Hence, to ascertain the feasibility of our approach, models of an off-the-shelf available op-amp, namely LT1012, were used in the simulations which take into account the non-idealities such as the off-set.

We then proposed two encoding techniques, with which the distance of values stored in the memristor is increased in order to ascertain a more reliable storage and retrieval of two- and three-bit values in a single memristor, despite the potential implementation inaccuracies and cell to cell variations. This technique being compatible with digital circuits, does not require any additional complex analog or mixed-signal circuits, which are currently used in the literature to increase the reliability of the storage and retrieval.

TABLE V. MEMRISTANCE FOR 3-BIT INPUT WITH NON UNIFORM ENCODING, WHERE “-” PRESENTS NO INPUT APPLICATION.

Input	Raw (kΩ)	Encoded Value (kΩ)	Encoding Scheme
“000”	7.2453	8.1511	“000**00**”
“001”	5.8991	5.8991	“001**--**”
“010”	5.8997	7.2455	“010**00**”
“011”	3.9089	3.9089	“011**--**”
“100”	5.8990	4.9996	“100**1-**”
“101”	3.9078	2.5906	“101**1-**”
“110”	3.9088	1.0404	“110**11**”
“111”	1.0396	0.1000	“111**1-**”

A. *Future Works*

We hope that in future we can expand this method by further evaluating the related issues for the memory system, and simulating the feasibility of faster storage and retrieval of data as well as larger number of bits in a single memristor. Moreover, by obtaining a memristor, we plan to implement the proposed circuit and verify the proposed approaches in practice as well.

Last but not least, we will explore applications of memristors in domains such as learning and approximate computing. In such applications, not only retrieving the exact value of stored data is not necessary, but often it burdens the system with extra computational loads. We believe, in such applications, by storing more data in memristors in a simple digital-compatible fashion such as proposed, and an approximate retrieval of data, learning systems can benefit from a smaller memory footprint and perform faster, with virtually no loss of performance.

REFERENCES

[1] L. Chua, “Memristor-the missing circuit element,” *IEEE Transactions on Circuit Theory,*, vol. 18, no. 5, pp. 507–519, Sep. 1971.

[2] ——, “Resistance switching memories are memristors,” *Applied Physics A*, vol. 102, no. 4, pp. 765–783, 2011.

[3] Y. V. Pershin, S. L. Fontaine, and M. D. Ventra, “Memristive model of amoeba learning,” *Physical Review E*, vol. 80, pp. 1–6, 2009.

[4] J. Borghetti, G. S. Snider, P. J. Kuekes, J. J. Yang, D. R. Stewart, and R. S. Williams, “Memristive switches enable stateful logic operations via material implications,” *Nature*, vol. 464, pp. 873–876, 2010.

[5] Y. Pershin and M. Di Ventra, “Neuromorphic, digital, and quantum computation with memory circuit elements,” *Proceedings of the IEEE*, vol. 100, no. 6, pp. 2071–2080, Jun. 2012.

[6] V. Hongal, R. Kotikalapudi, and M. Choi, “Design, test, and repair of mlut (memristor look-up table) based asynchronous nanowire reconfigurable crossbar architecture,” *IEEE Journal on Emerging and Selected Topics in Circuits and Systems,*, vol. 4, no. 4, pp. 427–437, Dec. 2014.

[7] Y. Levy, J. Bruck, Y. Cassuto, E. G. Friedman, A. Kolodny, E. Yaakobi, and S. Kvatinsky, “Logic operations in memory using a memristive akers array,” *Elsevier*, vol. 45, pp. 873–876, Nov. 2014.

[8] Y. Ho, G. Huang, and P. Li, “Dynamical properties and design analysis for nonvolatile memristor memories,” *IEEE Transactions on Circuits and Systems I: Regular Papers,*, vol. 58, no. 4, pp. 724–736, April 2011.

[9] B. Mohammad, D. Homouz, and H. Elgabra, “Robust hybrid memristor-CMOS memory: Modeling and design,” *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 21, no. 11, pp. 2069–2079, Nov 2013.

[10] V. Baghel and S. Akashe, “Low power memristor based 7T SRAM using MTCMOS technique,” in *2015 Fifth International Conference on Advanced Computing Communication Technologies (ACCT)*, Feb 2015, pp. 222–226.

[11] *International Technology Roadmap for Semiconductors - System Drivers*, 2011th ed., ITRS Technology Working Group, 2011.

[12] D. B. Strukov, G. S. Snider, D. R. Stewart, and R. S. Williams, “The missing memristor found,” *Nature*, vol. 453, pp. 80–83, 2008.

[13] D. Niu, Y. Chen, and Y. Xie, "Low-power dual-element memristor based memory design," in *2010 ACM/IEEE International Symposium on Low-Power Electronics and Design (ISLPED),*, Aug. 2010, pp. 25–30.

[14] H. Kim, M. P. Sah, C. Yang, and L. O. Chua, "Memristor-based multilevel memory," in *2010 12th International Workshop on Cellular Nanoscale Networks and Their Applications (CNNA)*, Feb 2010, pp. 1–6.

[15] M. Zangeneh and A. Joshi, "Design and optimization of nonvolatile multibit 1T1R resistive RAM," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 22, no. 8, pp. 1815–1828, Aug 2014.

[16] A. Rothenbuhler, T. Tran, E. H. B. Smith, V. Saxena, and K. A. Campbell, "Reconfigurable threshold logic gates using memristive devices," *Journal of Low Power Electronics and Applications*, vol. 3, no. 2, p. 174, 2013.

[17] D. Biolek, M. Di Ventra, and Y. V. Pershin, "Reliable SPICE Simulations of Memristors, Memcapacitors and Meminductors," *ArXiv e-prints*, Jul. 2013.

[18] N. Taherinejad, P. D. Sai Manoj, and A. Jantsch, "Memristors' potential for multi-bit storage and pattern learning," *Proceedings of the IEEE European Modelling Symposium (EMS) Conference*, Oct. 2015.