Memristor-based Memory: The Sneak Paths Problem and Solutions

Mohammed Affan Zidan\textsuperscript{a}, Hossam Aly Hassan Fahmy\textsuperscript{b}, Muhammad Mustafa Hussain\textsuperscript{a}, Khaled Nabil Salama\textsuperscript{a}

\textsuperscript{a}Electrical Engineering, King Abdullah University of Science and Technology (KAUST), Thuwal 23955-6900, Saudi Arabia
\textsuperscript{b}Electronics and Communication Department, Faculty of Engineering, Cairo University, Cairo, Egypt

Abstract
In this paper, we investigate the read operation of memristor-based memories. We analyze the sneak paths problem and provide a noise margin metric to compare the various solutions proposed in the literature. We also analyze the power consumption associated with these solutions. Moreover, we study the effect of the aspect ratio of the memory array on the sneak paths. Finally, we introduce a new technique for solving the sneak paths problem by gating the memory cell using a three-terminal memistor device.

Keywords: Nanotechnology, Memory, Memory Array, Memristor, Sneak Paths

1. Introduction
Memristors (memory resistors) offer a promising alternative to conventional memory devices. According to the International Technology Road-map for Semiconductors (ITRS), current memory technologies (DRAM, SRAM, and NAND Flash) will soon be facing design challenges related to their continued scaling-down [1]. Memristors are considered to be a very good candidate for future memory devices when compared to other emerging technologies such as Magnetoresistive RAM (MRAM) and Phase Change RAM (PCM/PCRAM) [2]. The main advantage these emerging technologies share is the property of retaining data after bias removal. Moreover, memristor-based memories have many unique advantages including very high density compared to other memory technologies and hard disk drives.

The memristor is a nonlinear resistor which changes its state relative to the net charge (or net electric flux) passing through its two terminals. It saves its state after an electrical bias is removed. The memristor (M), which was described by Chua in 1971 [3], is generally thought of as the fourth of the two-terminal basic passive elements, alongside the resistor (R), capacitor (C), and inductor (L). The first reported passive implementation of the memristor was the TiO\textsubscript{2}-based device introduced by HP in 2008 [4]. Recently, devices based on different materials have been introduced [5–9]. In addition, several models for memristors has been introduced [10–16]. Since the first reported use of the memristor, it received a significant of attention in the research community. In addition to be used as a memory element [2, 17–26], the memristor has found many applications in oscillators [27–30], logic and arithmetic circuits [31, 32], programmable analog circuits [33, 34], and in modeling and emulation of natural phenomena [35, 36].

One of the main challenges facing the memristor at the circuit and architecture level is the sneak paths problem. In this paper we introduce a new way to analyze the sneak paths using normalized noise margins. Our analysis is based on simulations for different memory array sizes, data sets, and architectures using the models presented in [17]. Moreover, we study the effect of the aspect ratio of the memory array on the sneak paths. Finally, a new method is introduce for solving the sneak paths based on a new gating technique by using three-terminal memistor device as a gate for the memristor memory cell.

The following section discusses the main concept of the memristor-based memory. The Section 3 describes the sneak paths analysis, and Section 4 summarized the main solutions for the sneak paths that have been described in the literature. Then, the new proposed solution is given in Section 5.

2. Memristor-Based Memory
Memristor-based memories are fabricated as a high-density crossbar architecture. Memristor devices are located at each intersection between two bars, as shown in Fig. 1. Typical memristor-based memories do not use transistors for cell gating. The advantage of these devices is that they have a retainable memory and a very high density compared to other storage devices.
2.1. Writing Operation

Data are stored in the memristor in the form of its resistance value, where each of the limiting resistances $R_{\text{off}}$ and $R_{\text{on}}$ are assigned to the two Boolean values ‘0’ and ‘1’. $R_{\text{off}}$ and $R_{\text{on}}$ are the maximum and minimum resistances of the device, respectively. Writing one of these values is simply done by passing current through the cell of interest until the memristor’s resistance saturates. The saturation value ($R_{\text{on}}$ or $R_{\text{off}}$ ) depends on the direction of the writing current. Even this simple writing operation could consume considerable of energy, depending on the values of the memristor’s resistances.

2.2. Reading Operation

While writing to the memristor is a straightforward operation, reading is more challenging. In the memristor memory array, we are trying to sense a cell resistance merged in a complete resistive structure. This could be compared to the problem of finding a needle in a haystack. Moreover, the reading operation itself could be destructive to the cell data, depending on the device properties.

2.3. Multilevel Memory

Multilevel memory is one promising application for the memristor device. Using such a technique would enormously increase the density of memristor-based memory, but would also reduce the noise margin significantly. The current proposed techniques for building binary memristor-based memory suffer from many problems that could be fatal for the multilevel memory. Some researchers believe 1M is insufficient for building multilevel memory, and that 1M1T (one memristor and one transistor) or 1M1D (one memristor and one diode) are needed [42]. We believe that addressing the current challenges facing the binary memristor-based memory will directly solve the multilevel memory problems.

3. Sneak Paths Analysis

Sneak paths are undesired paths for current, parallel to the intended path. The source of the sneak paths is the fact that the crossbar architecture is based on the memristor as the only memory element, without gating. Fig. 2a shows an array with a simple voltage divider and its equivalent circuit. The figure shows the ideal case in which the current flows from the source to the ground passing through only the desired cell at the intersection between the activated column and row. Unfortunately this is not the real case as shown in Fig. 2b. The current flows through many sneak paths beside the desired one. These paths act as an unknown parallel resistance to the desired cell resistance as shown in Fig. 2b. What makes the sneak paths problem harder to solve is the fact that the paths depend on the content of the memory. This is due to the fact that the current will sneak with more intensity through the paths with smaller resistance, which is memory content dependent.

The added resistance of the sneak paths significantly narrows the noise margin and reduces the maximum possible size of a memristor array. To study the effect of the sneak paths on the noise margin, we simulate memristor-based memory arrays of

Table 1: Detailed comparison between memristor-based memory, traditional memories, and other emerging memories according to the 2011 ITRS report [1]. The abbreviations used are: T – transistor, C – capacitor, R – resistor, and D – diode. The bold font indicates the best value per row.

<table>
<thead>
<tr>
<th>Feature Size (nm)</th>
<th>36-65</th>
<th>45</th>
<th>90</th>
<th>22</th>
<th>0.14</th>
<th>1.2</th>
<th>12</th>
<th>154 - 309</th>
</tr>
</thead>
<tbody>
<tr>
<td>Density (Gbit/cm²)</td>
<td>0.8 - 13</td>
<td>0.4</td>
<td>1.2</td>
<td>52</td>
<td>45</td>
<td>35</td>
<td>12</td>
<td>&lt;50</td>
</tr>
<tr>
<td>Read Time (ns)</td>
<td>2-10</td>
<td>0.2</td>
<td>15</td>
<td>100</td>
<td>65</td>
<td>35</td>
<td>100</td>
<td>0.3</td>
</tr>
<tr>
<td>Write Time (ns)</td>
<td>2-10</td>
<td>0.2</td>
<td>10⁷</td>
<td>10⁶</td>
<td>65</td>
<td>35</td>
<td>100</td>
<td>0.3</td>
</tr>
<tr>
<td>Retention Time</td>
<td>4-64 ms</td>
<td>N/A</td>
<td>10 years</td>
<td>10 years</td>
<td>10 years</td>
<td>&gt;10 years</td>
<td>&gt;10 years</td>
<td>&gt;10 years</td>
</tr>
</tbody>
</table>
different sizes and with different data sets. The sets are selected to reflect both the worst and best cases for the memory content. The worst case for the sneak paths is a memory full of “ones” since the effect of the sneak paths becomes more dominant as their resistance decreases. On the other hand the “all zeros” case is the best case condition since all the sneak paths are made of $R_{off}$ resistances in series. In addition to the previous cases, checkerered cases typical of real data and interleaved rows (or columns) are also used. These cases are considered as normal test cases since ones and zeros are present in equal numbers and are uniformly distributed. The simulation result is independent of the location of the cell in the array if we neglect the rows’ and columns’ pad resistances. We can interpret the array as a complete sphere, since connecting the terminals of each row or column will not introduce any change to the equivalent circuit as shown in Fig. 3. Hence all the cell locations are equivalent from the sneak paths point of view. All the simulations were made on Cadence Virtuoso 6 with a reading voltage ($V_R$) of ‘1V’ and a load resistance ($R_L$) of 19.76MΩ.

As our basic building block, we used the model in [17] for the memristor as a memory cell, such that:

$$I_M = k_x \sinh(aV)$$  \hspace{1cm} (1)

where $a$ and $k_x$ are constants. The reported feasible values in [17] for the constants are $a = 3^{-1}$ and $k_{on} = 10^{-8}$A and $k_{off} = 10^{-11}$A for the ON and OFF states of the device respectively.

In conventional CMOS circuits, there are two regions defined for accepted values of ONEs and ZEROs [43], as shown in Fig. 4. For typical CMOS circuits, the perfect ONE has the value of $V_{OH} = V_{dd}$ and the perfect ZERO is $V_{OL} = GND$. However, the circuit can tolerate shift in values of the input of certain bounds called noise margins.

In the case of memristor memory sensing, the value detected for ONE or ZERO depends on the severity of the sneak-path noise. For a given data pattern stored in the memory, the difference between voltages values representing ONE and ZERO at the target cell is a perfect measure for the sneak-path effect.

We define a total noise margin as the region between the ON and the OFF states, such that:

$$\Delta = V_{One} - V_{Zero}$$  \hspace{1cm} (2)

where $V_{one}$ and $V_{zero}$ are values for the sensing circuit output (voltage divider) in the case of ONE and ZERO stored in the desires cell. This value is proportional to the noise margin such that is equal to twice the margin if no guard region is assigned.

Figure 3: Two equivalent arrays from the sneak paths point of view.

Figure 4: Traditinal CMOS noise margines, where $V_{oh}$ and $V_{ol}$ are the maximum and the minimum voltage outputs of the circuit respectively. NMH abbreviates Noise Margin High and NML abbreviates Noise Margin Low.
To study the effect of the array size on the sneak-paths we defined a normalized value, where the $\Delta$ is compared to its best case, as:

$$
\Delta' = \frac{\Delta_{\text{Array}}}{\Delta_{\text{Device}}}
$$

where $\Delta_{\text{Device}}$ is the case of one device used (the best case) while $\Delta_{\text{Array}}$ is for the array case. $\Delta_{\text{Array}}$ is highly dependent on the data stored in the memory as shown later.

### 3.1. Floating Array

The basic structure for a memory array is to leave the unused array terminals floating. Simulation results for the floating memristor array are shown in Fig. 5a. The figure shows $\Delta'$ versus the array size for four different data sets. The simulations show that the noise margins of both the “all ones” and the “interleaved” cases almost vanish at a very small array size of 4kbit. At the array size of 16kbit, $\Delta'$ reaches a negligible value of 0.00145 and 0.00323 for the “all ones” and the “interleaved” cases respectively. This shows how the sneak paths affect the noise margins and consequently limit the maximum capacity of the array. On the other hand, the noise margin for the best case condition is almost unaffected by the array size. The reason for this is that the large $R_{\text{off}}/R_{\text{on}}$ ratio of $10^3$ makes sneak paths of resistances $R_{\text{off}}$ in series ineffective.

### 3.2. Grounded Array

Grounding the unselected rows and columns might be considered as a mean of preventing sneak paths. In [20] the equivalent circuits for all the possibilities of grounding the floating terminals are given. None of the four possibilities of: 1) floating rows and columns, 2) floating rows and grounded columns, 3) grounded rows and floating columns, and 4) grounded rows and columns, could solve the sneak paths problem. The idea behind grounding the floating terminals is to provide paths for the sneaking current to the ground rather than the sense circuit. However, part of the current will still find its path to the ground through the load resistance. Effectively grounding rows or columns or both will move part of the unknown resistance of the sneak paths so that it is parallel to the total resistance of the equivalent circuit instead of $R_M$, which does not solve the problem.

While grounding the array’s floating terminals does not solve the sneak paths problem, it does marginally improve the noise margin. Fig. 5b-d shows the simulation results for $\Delta'$ versus the array size for the grounded terminals cases. For the grounded rows and columns case, the simulations show that the noise margin still vanishes as the array size increases but at a slower rate than in the floating terminals case. At an array size of 16kbit, $\Delta'$ reaches a negligible values of 0.052 and 0.096 for the “all ones” and the “interleaved” cases respectively, but on the other hand these values are higher than for the floating terminals case. The two cases of grounding either columns or rows show slightly better results. This is due to fact that grounding both rows and columns at the same time is equivalent to short circuit any element not in the selected row or column. Since each of

![Figure 5: Noise margins ($\Delta'$) versus the array size containing four different data sets.](image)
The main disadvantage of the grounded technique is the huge power consumption for the reading operation compared to the floating terminals case. Fig. 6 shows a logarithmic plot of the average power consumption for the cases of floating and grounded terminals. The figure shows the enormous increase in power consumption for the grounded terminals case compared to the floating terminals case. Fig. 7 shows that the increase in power consumption is much more than the increase in noise margin. At an array size of 16kbit, an average power of 12.77μW is consumed in the grounded terminals case, compared to 48.88nW in the case of floating terminals, i.e. power consumption in the grounded terminals case is higher by a factor of 261.2. This level of power consumption makes the grounded terminals solution impractical. Also, at large array sizes, the improvement in the noise margin is impractical since the margin almost vanishes.

4. Sneak Paths Solutions

In this section the main solutions proposed in the literature for the sneak paths are discussed.

4.1. Multistage Reading

This method was introduced in [17] by the HP Labs team. Their technique attempts to overcome the sneak paths problem using a straightforward, but long, algorithm. The reading procedure is given as: 1) perform current measurement for the target cell, 2) put the target cell in the OFF state, and perform current measurement for the target cell, 3) put the target cell in the ON state, and perform current measurement for the target cell, 4) compare the measured currents to determine the state of the cell, and 5) return the memory cell to its (assumed) original state. This sensing algorithm requires a large amount of time and also a large sensing circuit (three sample-and-hold circuits, a voltage comparator, voltage divider, and the control circuit). This technique will also be inefficient for the narrow noise margins at large array sizes, since the effect of sneak paths will dominate and the resistance value of the target cell will be negligible. According to the simulations shown in Fig. 5a, ∆′ could be as low as 0.00145 for an array size of only 4kbit.

4.2. Unfolded Architecture

This solution is presented in [20], and is based on having a separate column for each memristor, as shown in Fig. 8. While this solution eliminates the sneak paths problem, it enormously reduces the memory density. The decreased density can be defined as:

\[ D_{uf} = \frac{D_o}{\text{no. of rows}} \]  

where \( D_{uf} \) is the new density of the unfolded architecture and \( D_o \) is the original density. An array with only one row will also eliminate the effect of sneak paths while occupying much less area than the unfolded architecture.
4.3. Diode Gating

One of the proposed solutions for the sneak paths is to add a diode to each memory cell [20], producing a new cell of one diode and one memristor (1D1M), as shown in Fig. 9. Such a strategy would eliminate sneak paths. According to [44], adding diodes to the array will increase the delay of the system by adding capacitive loads and diode threshold voltages will decrease the output swing. However, the major problem facing such a strategy is that it will block the writing process in the native array structure, since writing to a memristor requires two different polarities. In [45] a 3D array structure is provided to enable the write operation with a diode present. In this technique, each cell will contain one programming element, two diodes, and four connecting crossbars. While this technique allows the write operation, it consumes more area per cell. In addition, the 3D alignment for four bars may reduce the array density significantly. Finally, it is not clear that the new structure containing four bars will still eliminate the sneak paths.

4.4. Transistor Gating

Using large transistors for gating the memristor will solve the sneak paths problem. On the other hand this method will ruin the high memristor-memory density, since the gating transistor’s size is much larger than that of the memristor. Although using small devices will reduce the sneak paths it will not eliminate it. This is due to the fact that the recently introduced small transistors are considered to act as leaky valves. Moreover, these devices with relatively high OFF current will increase the static power component significantly. Finally, it should be mentioned that one of the major issues of using transistor gating is its limitation to the 3D stacking of memristor arrays. In [46] an array of one transistor and one memristor (1T1M) is reported. They report a gating transistor of 10\(\mu\)m channel length and 200\(\mu\)m channel width. Moreover, two wires are required for driving each cell; one for the transistor and one for the memristor.

4.5. Complimentary Memristors

In this technique two complimentary memristors are used in the memory cell, so that their total resistance is always \(R_{\text{on}} + R_{\text{off}}\), as introduced in [47]. Having always a high resistance cell reduces the sneak-path current significantly. In this method, one is distinguished from the zero by the orientation of the desired cell, \([R_{\text{on}}, R_{\text{off}}]\) or \([R_{\text{off}}, R_{\text{on}}]\). Therefore, a complex reading technique is required. Moreover, the system will not take full advantage of having high \(R_{\text{off}}/R_{\text{on}}\) device.

4.6. Using Memristors Nonlinearity

The voltage drop on the desired cell is higher than any of the sneak-path elements, since the shortest sneak-path will contain at least three series memristors. In [48], a high nonlinear device is reported, such that \(I(V/2) \approx I(V)/100\) at \(V \approx 1V\). This very useful property will significantly reduce the sneak paths current relative to the desired cell current, and will consistently reduce the sneak-path effect by a high factor. This solution also will not be practical for large memory array.

4.7. AC Sense

Instead of using regular DC signal an AC signal is used for sensing the data stored in the desired cell, as introduced in [49]. This technique uses load capacitance at the input of the sense amplifier to implement a low pass filter, as shown in Fig. 10. The response of the filter is mainly based on the resistive value of the desired cell. However, this method adds extra complexity for the memory system, since AC input and sensing are required. Moreover, this method will not be as effective for large arrays.

![Figure 10](image-url)  
Figure 10: Simple memory array showing the added column capacitors for the AC sense.

5. Array Aspect Ratio

In this section we study the effect of the aspect ratio on the performance of memristor array. Non-unity aspect ratio could be thought of as a helping method towards a sneak-paths free memory. The aspect ratio of an array is defined as its number of columns to the number of rows. Normal square arrays have aspect ratios of unity. The aspect ratio of the memory array is one of the main parameters which could be used to limit the effect of the sneak paths. A memory with one row or one column will not suffer from sneak paths at all, since there will be only one path for the current as shown in Fig. 11. As the aspect ratio approaches unity, the possibilities for sneak paths increase and \(\Delta'\) decreases. An unbalanced aspect ratio structure could
be fabricated in a square area by folding the array in a zigzag shape. However, the main cost of using an aspect ratio other than one is the increase in the required area for selection and sensing circuitry. This area could be given as:

\[ \text{Sense Circuit Area} = \sqrt{S} \left[ \frac{\theta}{\sqrt{A}} + \rho \sqrt{A} \right] \]  

(5)

where \( S \) is the array size, \( A \) is the aspect ratio, \( \theta \) is the column cell area, and \( \rho \) is the row cell area.

Fig. 12 shows the simulation results for the noise margin \( \Delta' \) versus the aspect ratio for different array sizes. The simulations are made for array structures with both floating and grounded terminals with the “all ones” data set (the worst case). In general the simulations show that the minimum values for \( \Delta' \) occur at an aspect ratio of one. Conversely, the sneak paths effect disappears completely for the cases of a one-row or one-column array. By comparing Fig. 12a and Fig. 12b we can see that the rate of decay of the noise margin with aspect ratio is much faster in the floating case than the grounded one. For the grounded arrays, the noise margin depends mainly on the number of rows rather than the aspect ratio. This is due to the fact that the current sneaks mainly through the columns near the active cell, regardless of the total number of columns.


Memristors can be considered better gates compared to transistors or diodes, since they can be characterized by having very high OFF resistance with much smaller area. In [17], memristor devices are reported to have \( R_{off} = 1 \text{G}\Omega \). Moreover, memristors are not intruder species to the memory array, compared to transistors and diodes. However, it is not possible to write on either the gate memristor or the data memristor separately, given a device with high ON/OFF ratios of more than one hundred. Also, trying to introduce extra rows or columns to enable separate writing will return us to the initial point, where the sneak paths are dominant.

The three-terminal memistor device captures both of the advantages of the memristor and transistor as a gate device. This device was introduced prior to the memristor in 1960 by Widrow [50]. The memistor is a three-terminal device where

![Figure 13: Structure of the proposed memristor gated array, where an example of selected cell is shown.](image)

![Figure 12: Noise margin (\( \Delta' \)) versus aspect ratio for different array sizes.](image)
the resistance between two terminals is controlled using the third one, in the same analogy of transistor but with a memory effect. In other words, the resistance of the device is controlled by time integral of the current on the third terminal and not the instantaneous current as the case of transistor, as stated be in [50, 51]. This means that there is no need to keep an active bias on the third terminal to keep the device ON (or OFF). The memistor will retain its ON or OFF state after removing the bias from the third terminal. One of the advantages of this bias-less switching is the very low static power consumed. Memistors can inherit the high ON/OFF ratio and small footprint of memristors and the high controllability of transistor by having a third terminal. Fig. 13 shows the structure of the memistor gated array, where each memory cell is gated with a memistor device. Extra columns are required for programming the memistor gate. It is a assumed that the memistor has the same ON/OFF values of the data cell. At the desired cell the gate device is turned ON and all the other gates are turned OFF, which is how the desired cell is selected. Therefore, all the sneak paths will contain at least three series high resistances. This will shift the operation of the memory to work equivalently to the best case scenario, where all the sneak paths are made of OFF devices.

All the unselected cells will have a total resistance higher than \( R_{off} \). The resistance of the selected cell, with open gate, will depend mainly on the data device resistance. This resistance will be either \( 2R_{on} \) or \( R_{off}+R_{on} \), with a very high ON/OFF ratio. Our proposed method has a major advantage over the complementary memristors technique that the desired cell has much higher ratio between its ON and OFF states. In complementary structures the total resistance is always \( (R_{on} + R_{off}) \), in all of the cases. Based on that, our proposed solution has higher average signal to noise ratio. Moreover, the \( R_{off}/R_{on} \) ratio of the device is directly reflected on desired cell state values.

Fig. 14 show the noise margins for the proposed technique and the normal array with floating terminals versus the array size for the worst case data set “all ones”. It appears clearly that the memistor gating has a significant impact on the sneak paths effect, where the worst case of the memistor gating is almost as the best case of the normal array with floating terminals. For 16kbit array the memistor gating architecture noise margins is 619.5x compared to the normal array with floating terminals. In the same direction, the worst case power consumption is significantly decreased as shown in Fig. 15. For the worst case with the memory filled with “all ones”, average power consumption was reduced more than five time for an array size of 16kbit. This ratio increases as the array size increases. Finally, in worth mentioning that while miniaturizing the three-terminal memistor as a single device did not pass the same long path as the two-terminal memristor, we believe that its great advantages as a gating device will motivate the fabrication community for creating better memistor devices.

7. Conclusion

We reviewed the main introduced solutions for the memristors sneak-paths problem. We proposed a new technique for analyzing the sneak paths problem facing memristor-based memory arrays. The new analysis is accompanied by simulations on Cadence Virtuoso 6 for different memory array sizes, data sets, and architectures. We also studied the memory array aspect ratio effect on the sneak paths. Finally, we introduced an new solution for the problem based on using the three-terminal memistor device.

Acknowledgments

The authors would like to thank Dr. Amro Elshurafa for his reviews and valuable suggestions.

References

[1] International technology roadmap for semiconductors. URL http://www.itrs.net/
[26] G. Huang, Y. Ho, P. Li, Memristor system properties and its design applications to circuits such as nonvolatile memristor memories, in: International Conference on Communications, Circuits and Systems (ICCCAS’10), 2010, pp. 805–810.