MAGIC—Memristor-Aided Logic

Shahar Kvatinsky, *Student Member, IEEE*, Dmitry Belousov, Slavik Liman, Guy Satat, *Student Member, IEEE*, Nimrod Wald, Eby G. Friedman, *Fellow, IEEE*, Avinoam Kolodny, *Senior Member, IEEE*, and Uri C. Weiser, *Fellow, IEEE*

***Abstract*—Memristors are passive components with a varying resistance that depends on the previous voltage applied across the device. While memristors are naturally used as memory, memristors can also be used for other applications, including logic circuits. In this brief, a memristor-only logic family, i.e., memristor-aided logic (MAGIC), is presented. In each MAGIC logic gate, memristors serve as an input with previously stored data, and an additional memristor serves as an output. The topology of a MAGIC NOR gate is similar to the structure of a common memristor-based crossbar memory array. A MAGIC NOR gate can therefore be placed within memory, providing opportunities for novel non-von Neumann computer architectures. Other MAGIC gates also exist (e.g., AND, OR, NOT, and NAND) and are described in this brief.**

***Index Terms*—IMPLY, in-memory computing, memristive systems, memristor, memristor-aided logic (MAGIC), stateful logic.**

I. Introduction

In recent years, the concept of a memristor, originally proposed by Leon Chua in 1971, has generated renewed interest. In [1], Chua proposed a fourth fundamental component in addition to the three already well-known fundamental electronic components: resistor, capacitor, and inductor. In [13], Chua and Kang extended the theory of memristors to memristive systems. Memristors and memristive devices are two-terminal devices, where the resistance of the device is changed by the electrical current, as shown in Fig. 1. The resistance of the memristor is bounded by a minimum resistance R_{ON} and a maximum resistance R_{OFF}. In this brief, for simplicity, the terms memristor and memristive device are used interchangeably [14].

For almost 40 years, the concept of a memristor was just theoretical. In 2008, Hewlett Packard Laboratories [2] connected the physical phenomenon of resistance switching in TiO_2 to memristors. Since 2008, several possible applications of memristors have been presented. Nonvolatile emerging memory technologies, including resistive RAM (RRAM) and spin–transfer torque magnetoresistive RAM, can be considered as memristors [14]. The primary application of memristors has been memory [3], [16], [22], where the resistance serves to store data. Another interesting and new application is memristor-based logic [4], [5], [9]–[12], [17], [18].

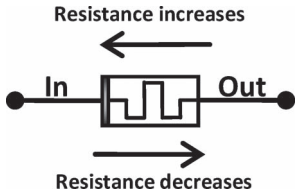

Fig. 1. Memristor symbol. The polarity of the memristor is represented by a thick black line. When current flows into the device (the upper arrow), the resistance of the device increases. When current flows out of the device (the lower arrow), the resistance of the device decreases.

Stateful logic, including material implication (IMPLY), as a memristor-based logic gate is presented in [5], [9], [11], [17], [22], and [23]. The proposed stateful logic is built within a memristive crossbar array, the most commonly used memristive memory structure (particularly for RRAM). The stored data within the memristors are the input and output of the logic gate. This method, however, requires sequential voltage activation at different locations within the circuit. Furthermore, with IMPLY, the result is stored by one of the inputs and not by a dedicated output memristor. Stateful logic also requires additional circuit components (for example, a controller and an additional resistor within each row of the crossbar), dissipates high power, has high computational complexity, and requires complicated control circuitry.

In this brief, memristor-aided logic (MAGIC)—a method for memristive-only logic—is presented. This method does not require a complicated structure and enables stable evaluation of the gate function. Stable evaluation is achieved by applying a single voltage pulse at the gateway of the circuit. MAGIC NOR gates can also be fabricated within a crossbar, enabling computing within memory.

II. Operating Principle of MAGIC

MAGIC requires only memristors within the logic gates. The logical state in a MAGIC gate is represented as a resistance, where the high and low resistances are considered, respectively, as logical zero and one (for simplicity, the resistance of logical zero and logical one is considered, respectively, as R_{OFF} and R_{ON}). The inputs and output of the logic gates are the logical states of the memristors. Unlike an IMPLY logic gate, separate memristors are required for the input and output. The inputs

Manuscript received April 24, 2014; revised July 1, 2014; accepted September 3, 2014. Date of publication September 11, 2014; date of current version November 1, 2014. This work was supported in part by the Hasso Plattner Institute, by the Advanced Circuit Research Center at the Technion, by the Intel Collaborative Research Institute for Computational Intelligence (ICRI-CI), and by the U.S.–Israel Binational Science Foundation under Grant 2012139. This brief was recommended by Associate Editor T. S. Gotarredona.

S. Kvatinsky, D. Belousov, S. Liman, A. Kolodny, and U. C. Weiser are with the Department of Electrical Engineering, Technion—Israel Institute of Technology, Haifa 32000, Israel (e-mail: skva@tx.technion.ac.il).

G. Satat is with the MIT Media Laboratory, Massachusetts Institute of Technology, Cambridge, MA 02139 USA.

N. Wald is with Qualcomm, Inc., Haifa 3190500, Israel.

E. G. Friedman is with the Department of Electrical Engineering and Computer Engineering, University of Rochester, Rochester, NY 14627 USA.

Color versions of one or more of the figures in this brief are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCSII.2014.2357292

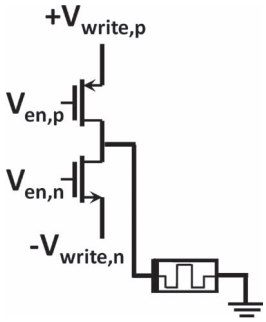

Fig. 2. Example of placing a memristor into the initialization stage. This example is similar to writing a configurable analog circuit [19]. A different initialization scheme is presented in Section III for MAGIC within memory.

of the MAGIC gates are the initial logical state of the input memristors, and the output is the final logical state of the memristor. This concept is demonstrated in Fig. 3(a) for a MAGIC NOR gate, where the initial resistances of memristors $in1$ and $in2$ are the inputs of the gate, and the final resistance of memristor *out* is the output of the gate.

Operation of a MAGIC gate consists of two sequential stages. The first stage initializes the output memristor to a known logical state. In the second stage of operation, a voltage V_0 is applied across the logic gate. While applying V_0, the voltage across the output memristor depends upon the logical state of the input and output memristors. The nonlinear characteristics of the memristor, namely, the threshold currents or voltages [8], are exploited to maintain correct operation. For specific input combinations, the voltage is sufficient to change the logical state of the output memristor, i.e., the memristor voltage/current is greater than the threshold voltage/current, whereas for other input combinations, the output remains at the initialize state, i.e., the memristor voltage/current is below the threshold current or voltage. Note that, in certain cases, full switching is not achievable for memristors with a threshold current.

Initialization of the memristors can be achieved in several ways. For example, it is possible to use a similar topology as the circuit used in [19] for configurable memristive analog circuits, as shown in Fig. 2. For MAGIC gates within memory (as described in Section III), initialization is achieved as a regular write operation within the memory cells.

In the next section, the basic MAGIC NOR gate is described. Additional MAGIC gates for different Boolean functions based on the MAGIC topology are also available and described in Section VI.

III. MAGIC NOR Gate

A two-input NOR gate consists of two input memristors ($in1$, $in2$) connected in parallel and an additional memristor (*out*) as the output. A schematic of a two-input NOR gate is shown in Fig. 3(a). The initial execution step includes writing a low resistance into the output memristor (initialization to logical one) and, if necessary, writing the input value into memristors $in1$ and $in2$. In the final execution step, the evaluation is achieved by applying a voltage pulse V_0 at the gateway of the logic gate [the gateway as shown in Fig. 3(a)].

The applied voltage produces a current that passes through the circuit and appears at memristor *out*. For the case where both

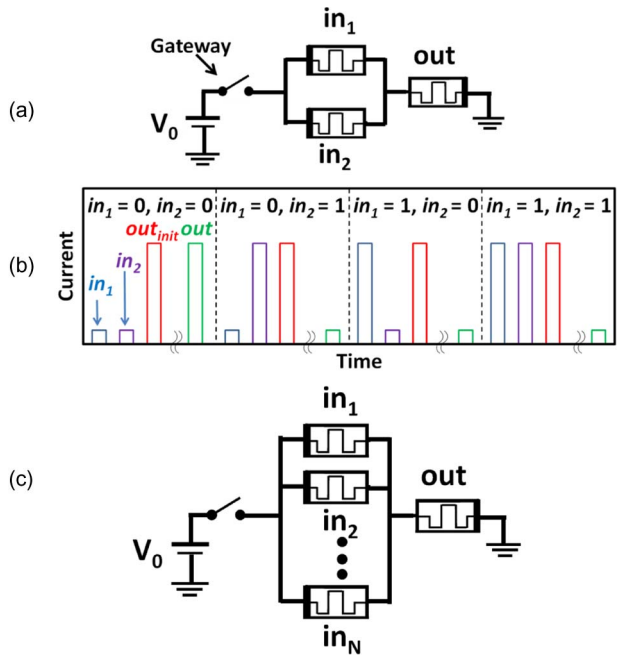

Fig. 3. MAGIC NOR. (a) Schematic of a two-input NOR gate. The logic gate consists of two input memristors $in1$ and $in2$ and an output memristor *out*. During execution, a voltage V_0 is applied at the gateway of the circuit (marked by an arrow). (b) Simulations of a two-input NOR gate for all input combinations. The different curves show the currents read from each memristor prior to execution and after applying V_0. (c) Schematic of an N-input NOR.

input memristors are logical zero (high resistance), the voltage/current of the output memristor is lower than the memristor threshold voltage/current. Hence, the logical state of the output memristor does not change and remains at logical one. For all other input combinations, the voltage/current is greater than the memristor threshold voltage/current. The logical state of the output memristor for these input combinations switches to logical zero. The behavior of the MAGIC NOR gate is shown in Fig. 3(b).

Assume a memristor with voltage thresholds of $V_{T,\mathrm{ON}}$ and $V_{T,\mathrm{OFF}}$. For correct circuit behavior, the voltage at the output memristor is lower than $V_{T,\mathrm{OFF}}$ when both inputs are logical zero. For all other input combinations, the voltage across the output memristor should be greater than $V_{T,\mathrm{OFF}}$. The minimum voltage at the output memristor greater than $V_{T,\mathrm{OFF}}$ is achieved when one input is logical one and the other input is logical zero. Combining the cases where the voltage at the output memristor is above and below the threshold voltage leads to a design constraint on the applied voltage V_0. Assuming $R_{\mathrm{OFF}} \gg R_{\mathrm{ON}}$, the constraint is

$$2V_{T,\mathrm{OFF}} < V_0 < \frac{R_{\mathrm{OFF}}}{2R_{\mathrm{ON}}} \cdot V_{T,\mathrm{OFF}}. \tag{1}$$

When an input memristor is logical zero, the operation of a MAGIC NOR can be destructive, changing the input to logical one during execution. To eliminate destroying the input, the voltage across the input memristor needs to be below the threshold voltage $V_{T,\mathrm{ON}}$. The maximum applied voltage for a two-input NOR gate is therefore

$$V_0 < \min\left[\frac{R_{\mathrm{OFF}}}{2R_{\mathrm{ON}}} \cdot V_{T,\mathrm{OFF}}, |V_{T,\mathrm{ON}}|\right]. \tag{2}$$

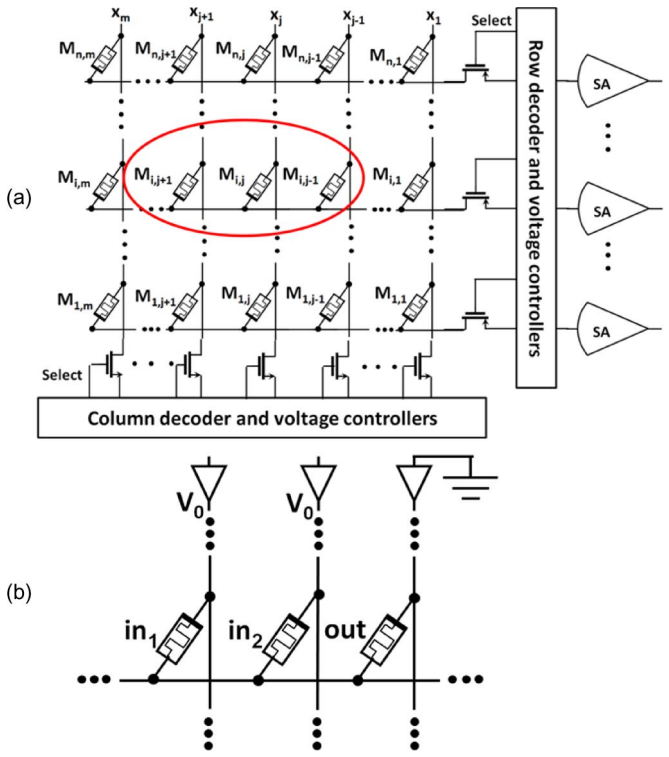

Fig. 4. MAGIC NOR gate within a crossbar array. (a) Schematic of a memristive crossbar structure. A two-input NOR gate is achieved in row i, where $in1$ and $in2$ are, respectively, in columns $j+1$ and j, and *out* is in column $j-1$, as marked by an oval. (b) Schematic of a two-input NOR gate within a crossbar array. The voltage at the gateway V_0 is the applied voltage at columns j and $j+1$, whereas column $j-1$ is connected to ground. Note that the schematic is identical to the figure shown in Fig. 3(a).

Multiple-input (three or more) NOR logic gates can also be produced in a similar manner, as shown in Fig. 3(c). For χ input memristors, the design constraints are

$$\frac{V_{T,\mathrm{OFF}}}{R_{\mathrm{ON}}}\cdot\left[R_{\mathrm{ON}}+\left(\frac{R_{\mathrm{OFF}}}{\chi-1}\right)\|R_{\mathrm{ON}}\right]<V_0<V_{T,\mathrm{OFF}}\cdot\left[1+\frac{R_{\mathrm{OFF}}}{\chi R_{\mathrm{ON}}}\right]. \quad (3)$$

For nondestructive operation of a χ-input NOR, the maximum applied voltage is

$$V_0<\min\left[V_{T,\mathrm{OFF}}\cdot\left(1+\frac{R_{\mathrm{OFF}}}{\chi R_{\mathrm{ON}}}\right),\left(1+\frac{\chi R_{\mathrm{ON}}}{R_{\mathrm{OFF}}}\right)\cdot|V_{T,\mathrm{ON}}|\right]. \quad (4)$$

IV. MAGIC Within a Crossbar Array

RRAM commonly utilizes a crossbar structure. The crossbar structure enables dense memory of $4F^2$, where F is the feature size. Memristive-only logic gates within a memristive crossbar array reduce power and provide an opportunity for novel non-von Neumann architectures, where the logical operations are executed within the memory [11]. When performing logic within the memory, the input is the stored data within the memristors, and the output is the stored data after execution. Initialization of the input and output is achieved as a regular memory write operation, and sensing the result is achieved as a regular memory read operation.

To integrate a memristive-only logic gate within a crossbar array, two requirements need to be satisfied: the structure and connections of the logic gate should be placed within a crossbar array, and the logical state of the logic gate is represented as a resistance, as in a memristive memory. A MAGIC NOR gate fulfills both of these requirements. The structure of a memristive crossbar array and a two-input MAGIC NOR gate within a crossbar is shown in Fig. 4.

While other memristive stateful logic gates can also be integrated within a memristive crossbar array [11], these logic gates require an additional resistor within each row of the crossbar array. Additionally, unlike the NOR Boolean operation, the IMPLY operation is not logically complete and requires the operation of FALSE (writing a logical zero to a memristor). A comparison between memristive IMPLY and MAGIC is listed in Table I.

TABLE I
COMPARISON BETWEEN IMPLY AND MAGIC

	IMPLY [11]	MAGIC
No. of voltages	2 (V_{SET}, V_{COND})	1 (V_0)
Separate input and output	No	Yes
Basic functions	IMPLY (+ FALSE)	OR, AND, NOR, NAND, NOT
No. of memristors for NOR/NAND	3 (+ a resistor)	3
No. of clock cycles for NOR/NAND	4	2
Within memory	Yes	Yes (for NOR)
Logically complete	Requires FALSE	Yes (NOR, NAND)

TABLE II
MEMRISTOR PARAMETERS (FOR VTEAM MODEL [20])

k_{on}	-216.2 m/sec
k_{off}	0.091 m/sec
$V_{T,ON}$	-1.5 volt
$V_{T,OFF}$	0.3 volt
x_{on}	0
x_{off}	3 nm
α_{on}	4
α_{off}	4
R_{ON}	1 kΩ
R_{OFF}	300 kΩ

V. Evaluation and Design Considerations for a MAGIC NOR Gate

The speed of a MAGIC NOR gate is evaluated in SPICE for a 0.18-μm CMOS process. A memristor model, the VTEAM model [20], which extends the TEAM model with a threshold voltage, is used with a Biolek window function [8]. The parameters of the memristors are chosen to produce a switching time of 1 ns for a voltage pulse of 1 V for RESET and 2 V for SET and to fit practical devices, as reported in [21]. The parameters of the circuit simulations are listed in Table II.

The behavior and speed of a MAGIC NOR gate for different values of V_0 are shown in Fig. 5. To evaluate the delay of the logic gate, the slowest input case is considered. The delay of a MAGIC NOR gate is determined from an input combinations of {1,0} or {0,1}.

From (1) and (2), V_0 can vary from 0.6 to 1.5 V for the parameters listed in Table II. As shown in Fig. 5(b), increasing the applied voltage V_0 decreases the delay of the logic gate. For V_0 at 1 V, the delay of the logic gate is 1.3 ns, an increase of 30% as compared to the switching time of a single memristor.

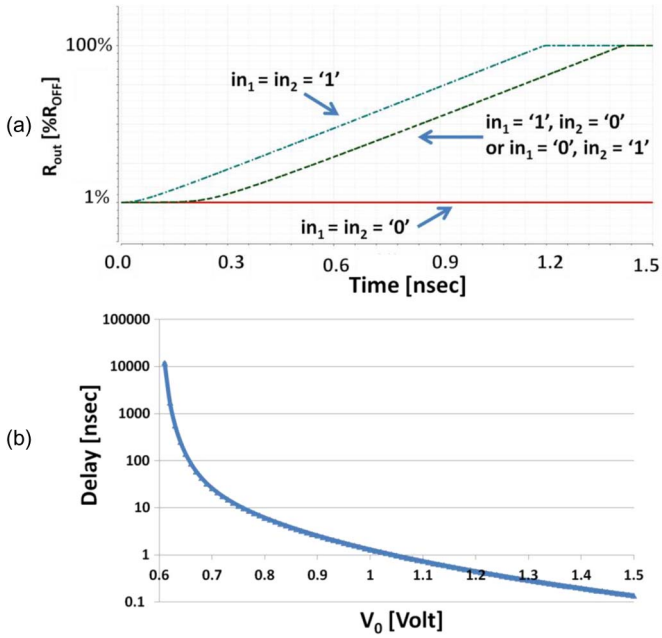

Fig. 5. SPICE simulations of a two-input MAGIC NOR gate. (a) Output memristor for different input combinations, $V_0 = 1$ V. The delay is evaluated as the time to switch the output memristor to logical zero when one input is logical one and the other input is logical zero (dashed line). (b) Delay for different values of V_0.

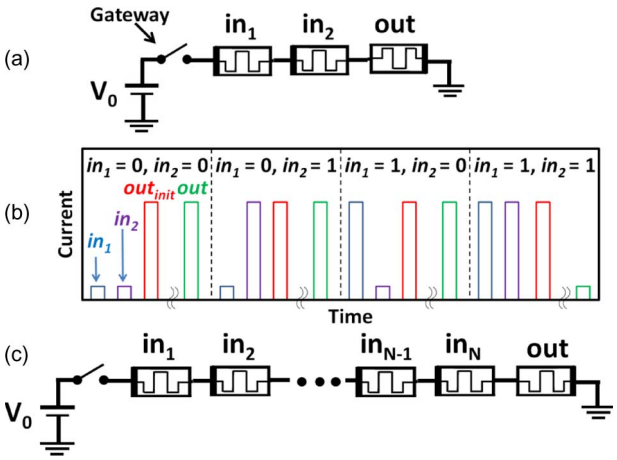

Fig. 6. MAGIC NAND gate. (a) Schematic of a two-input NAND gate. The logic gate consists of two input memristors $in1$ and $in2$ and an output memristor *out*. During execution, a voltage V_0 is applied at the gateway of the circuit. (b) Simulation results for a two-input NAND gate for all input combinations. The different curves exhibit the currents of each memristor prior to execution and after applying V_0. (c) Schematic of an N-input NAND gate.

VI. Additional MAGIC Gates

With the same design principles described in Section II, additional Boolean functions can added to the MAGIC family. The additional MAGIC gates described in this section are not placed within a crossbar array (except for the NOT gate) but can be used as standalone logic.

Connecting the input memristors in series with the same topology as the MAGIC NOR gate produces a NAND gate, as shown in Fig. 6. OR and AND logic gates have a similar structure as, respectively, NOR and NAND, except for the opposite polarity of the output memristor *out*. Unlike NAND and NOR, *out* is initialized to logical zero prior to execution. The schematic and

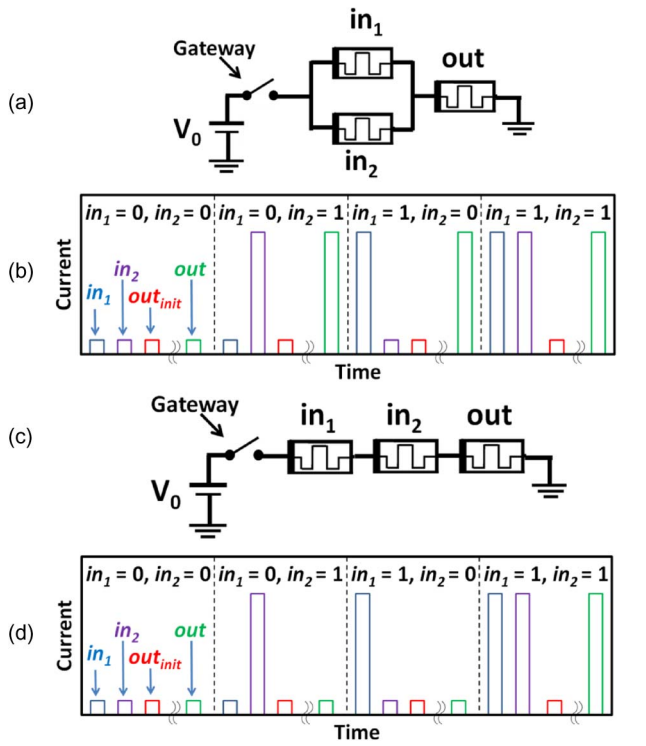

Fig. 7. MAGIC OR and AND gates. The structure of the logic gates is similar to MAGIC NOR and NAND gates. The output memristor *out* is connected with the same polarity as the input memristors and is initialized to logical zero. (a) Schematic of a two-input OR gate. (b) Simulation results for a two-input OR gate for all input combinations. (c) Schematic of a two-input AND gate. (d) Simulation results for a two-input AND gate for all input combinations.

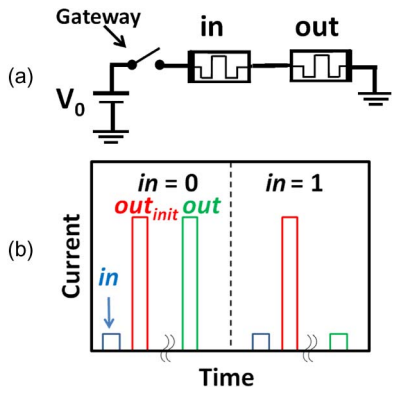

Fig. 8. MAGIC NOT gate. (a) Schematic of a NOT gate. The logic gate consists of an input memristor *in* and an output memristor *out*. (b) Simulation results for a NOT gate.

behavior of the OR and AND MAGIC gates are shown in Fig. 7. Similar to the MAGIC NOR and NAND gates, multi-input logic gates are also possible for the MAGIC OR and AND gates.

A MAGIC NOT gate (inverter) consists of an input memristor *in* and an output memristor *out*. The memristors are connected in series with an opposite polarity in a complementary memristor structure (or complementary resistive switches) [15], as shown in Fig. 8(a). In the first stage of execution, the output memristor is initialized to logical one. When applying V_0 at the gateway of the circuit, the voltage divider between *in* and *out* determines whether the resistance of the output memristor changes. For the case where *in* is logical zero, the voltage across *out* is below the threshold voltage, and the logical state

TABLE III
SUMMARY OF MAGIC GATES

Function	Design Constraints	Design Constraints – Multiple Inputs	Within Crossbar?
NOR	$2V_{T,OFF} < V_0 < \min\left[\frac{R_{OFF}}{2R_{ON}} \cdot V_{T,OFF}, \lvert V_{T,ON}\rvert\right]$	$\frac{V_{T,OFF}}{R_{ON}} \cdot \left[R_{ON} + \left(\frac{R_{OFF}}{\chi - 1}\right) \parallel R_{ON}\right] < V_0 < \min\left[V_{T,OFF} \cdot \left(1 + \frac{R_{OFF}}{\chi R_{ON}}\right), \left(1 + \frac{\chi R_{ON}}{R_{OFF}}\right) \cdot \lvert V_{T,ON}\rvert\right]$	Yes
NAND	$3V_{T,OFF} < V_0 < \min\left[\lvert V_{T,ON}\rvert, \left(2 + \frac{R_{OFF}}{R_{ON}}\right) \cdot V_{T,OFF}\right]$	$(\chi + 1) \cdot V_{T,OFF} < V_0 < \min\left[\lvert V_{T,ON}\rvert \cdot \left(1 + \frac{\chi R_{ON}}{R_{OFF}}\right), \left(\chi + \frac{R_{OFF}}{R_{ON}}\right) \cdot V_{T,OFF}\right]$	No
OR	$V_{T,ON} < V_0 < 1.5V_{T,ON}$	$V_{T,ON} < V_0 < \left(1 + \frac{1}{\chi}\right) V_{T,ON}$	No
AND	$V_{T,ON} < V_0 < 2V_{T,ON}$	$\left(1 + \frac{R_{ON}}{R_{OFF}}\chi\right) V_{T,ON} < V_0 < \left(2 + \frac{R_{ON}}{R_{OFF}}(\chi - 1)\right) V_{T,ON}$	No
NOT	$2V_{T,OFF} < V_0 < \frac{R_{OFF}}{R_{ON}} \cdot \min\left(V_{T,OFF}, \lvert V_{T,ON}\rvert\right)$	-	Yes

of *out* remains logical one, as desired. Note that in this case, the voltage at *in* is relatively high, and the logical state at *in* may therefore be switched to logical zero. Hence, the MAGIC NOT operation can be destructive to the input unless the applied voltage at memristor *in* is below $V_{T,\mathrm{ON}}$. For the case where *in* is logical one, the voltage across memristor *out* is sufficient to switch the logical state of *out* (greater than the threshold voltage) to logical zero. Simulation results for a NOT gate are shown in Fig. 8(b). A summary of several MAGIC gates, including the design constraints, is listed in Table III. For linear memristors and memristors without a threshold voltage (or current), similar design issues as those reported for the IMPLY gate [9], [11] need to be considered (e.g., the state drift phenomenon), complicating the design procedure.

VII. CONCLUSION

MAGIC, a novel method for memristor-based logic, is presented in this brief. Five basic logic functions, namely, NOT, AND, NAND, NOR, and OR, use simple connections among memristors, where the number of memristors is equal to the number of inputs plus one additional memristor at the output.

Only one applied voltage controls these logic gates, different than other memristor-based stateful logic. Unlike the IMPLY gate, the input and output in MAGIC are separated, and the output is written to a dedicated memristor. The use of MAGIC NOR gates within a memristive crossbar array can lead to more efficient systems in terms of performance and power consumption and to novel non-von Neumann architectures.

REFERENCES

[1] L. O. Chua, "Memristor—The missing circuit element," *IEEE Trans. Circuit Theory*, vol. 18, no. 5, pp. 507–519, Sep. 1971.

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

[3] Y. Ho, G. M. Huang, and P. Li, "Nonvolatile memristor memory: Device characteristics and design implications," in *Proc. IEEE Int. Conf. Comput.-Aided Des.*, Nov. 2009, pp. 485–490.

[4] Q. Xia *et al.*, "Memristor–CMOS hybrid integrated circuits for reconfigurable logic," *Nano Lett.*, vol. 9, no. 10, pp. 3640–3645, Oct. 2009.

[5] J. Borghetti *et al.*, "Memristive switches enable 'stateful' logic operations via material implication," *Nature*, vol. 464, pp. 873–876, Apr. 2010.

[6] E. Lehtonen, J. H. Poikonen, and M. Laiho, "Two memristors suffice to compute all Boolean functions," *Electron. Lett.*, vol. 46, no. 3, pp. 239–240, Feb. 2010.

[7] M. D. Pickett *et al.*, "Switching dynamics in titanium dioxide memristive devices," *J. Appl. Phys.*, vol. 106, no. 7, pp. 074508-1–074508-6, Oct. 2009.

[8] S. Kvatinsky, E. G. Friedman, A. Kolodny, and U. C. Weiser, "TEAM: ThrEshold adaptive memristor model," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 60, no. 1, pp. 211–221, Jan. 2013.

[9] S. Kvatinsky, E. G. Friedman, A. Kolodny, and U. C. Weiser, "Memristor-based IMPLY logic gate design procedure," in *Proc. IEEE Int. Conf. Comput. Des.*, Oct. 2011, pp. 142–147.

[10] S. Kvatinsky *et al.*, "MRL—Memristor ratioed logic for hybrid CMOS–memristor circuits," *IEEE Trans. Nanotechnol.*, to be published.

[11] S. Kvatinsky, E. G. Friedman, A. Kolodny, and U. C. Weiser, "Memristor-based material implication (IMPLY) logic: Design principles and methodologies," *IEEE Trans. Very Large Scale Integr. (VLSI)*, vol. 22, no. 10, pp. 2054–2066, Oct. 2013.

[12] S. Kvatinsky *et al.*, "MRL—Memristor ratioed logic," in *Proc. Int. Cellular Nanoscale Netw. Appl.*, Aug. 2012, pp. 1–6.

[13] L. O. Chua and S. M. Kang, "Memristive devices and systems," *Proc. IEEE*, vol. 64, no. 2, pp. 209–223, Feb. 1976.

[14] L. O. Chua, "Resistance switching memories are memristors," *Appl. Phys. A, Mater. Sci. Process.*, vol. 102, no. 4, pp. 765–783, Mar. 2011.

[15] E. Linn, R. Rosezin, C. Kügeler, and R. Waser, "Complementary resistive switches for passive nanocrossbar memories," *Nat. Mater.*, vol. 9, no. 5, pp. 403–406, May 2010.

[16] R. Patel, S. Kvatinsky, E. G. Friedman, and A. Kolodny, "Multistate register based on resistive RAM," *IEEE Trans. Very Large Scale Integr. (VLSI)*, to be published.

[17] E. Lehtonen and M. Laiho, "Stateful implication logic with memristors," in *Proc. IEEE/ACM Int. Symp. Nanoscale Archit.*, Jul. 2009, pp. 33–36.

[18] Y. Levy *et al.*, "Logic operation in memory using a memristive Akers array," *Microelectron. J.*, to be published.

[19] Y. V. Pershin and M. Di Ventra, "Practical approach to programmable analog circuits with memristors," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 57, no. 8, pp. 1857–1864, Aug. 2010.

[20] S. Kvatinsky, M. Ramadan, E. G. Friedman, and A. Kolodny, "VTEAM—A General Model for Voltage Controlled Memristors," CCIT Tech. Rep. #856, Apr. 2014.

[21] J. J. Yang, D. B. Strukov, and D. R. Stewart, "Memristive devices for computing," *Nat. Nanotechnol.*, vol. 8, pp. 13–24, Jan. 2013.

[22] K. Kim, S. Shin, and S.-M. S. Kang, "Field programmable stateful logic array," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 30, no. 12, pp. 1800–1813, Dec. 2011.

[23] E. Lehtonen, J. H. Poikonen, and M. Laiho, "Applications and limitations of memristive implication logic," in *Proc. Int. Workshop CNNA*, Aug. 2012, pp. 1–6.