A Family of Stateful Memristor Gates for Complete Cascading Logic

Kyung Min Kim, *Member, IEEE*, and R. Stanley Williams, *Senior Member, IEEE*

***Abstract*— The conditional switching of memristors to execute stateful implication logic is an example of in-memory computation to potentially provide high energy efficiency and improved computation speed by avoiding the movement of data back and forth between a processing chip and memory and/or storage. Since the first demonstration of memristor implication logic, a significant goal has been to improve the logic cascading to make it more practical. Here, we describe and experimentally demonstrate nine symmetry-related Boolean logic operations by controlling conventional Ta/TaOx/Pt memristors integrated in a crossbar array with applied voltage pulses to perform conditional SET or RESET switching involving two or three devices, i.e., a particular device is switched depending on the state of another device. We introduce a family of four stateful two-memristor logic gates along with the copy and negation operations that enable two-input-one-output complete logic. In addition, we reveal five stateful three-memristor gates that eliminate the need for a separate data copy operation, decreasing the number of steps required for a particular task. The diversity of gates made available by simply applying coordinated sequences of voltages to a memristor crossbar memory significantly improves stateful logic computing efficiency compared to similar approaches that have been proposed.**

***Index Terms*— Logic circuits, Logic-in-memory, memristors, stateful logic.**

I. Introduction

AFTER a physical mechanism that exhibited the current-voltage 'pinched hysteresis loop' of the memristor mathematical model first formulated by Chua [1] was described in 2008 by Strukov *et al.* [2], researchers have invented various methods for computation and logic that utilize the nonlinear dynamical resistance switching characteristic of this fundamental circuit element [3], [4]. One notable approach was stateful implication logic [5]–[12], for which sequential Boolean logic operations are performed directly on bits stored in a memory array to perform logic operations without moving data to and from a processor chip. Eliminating data transfer operations between memory and processor can in principle provide highly energy-efficient computing, especially for applications in mobile devices and the internet of things (IoT) for which extreme computational speed may not be essential [13].

Manuscript received June 16, 2019; accepted July 2, 2019. Date of publication July 25, 2019; date of current version October 30, 2019. This paper was recommended by Associate Editor A. James. *(Corresponding author: R. Stanley Williams.)*

K. M. Kim is with the Department of Materials Science and Engineering, KAIST, Deajeon 34141, South Korea (e-mail: km.kim@kaist.ac.kr).

R. S. Williams is with the Department of Electrical and Computer Engineering, Texas A&M University, College Station, TX 77843 USA (e-mail: stan.williams@tamu.edu).

Digital Object Identifier 10.1109/TCSI.2019.2926811

Material implication (IMP) logic gates were realized with two memristors and one resistor, and are capable of synthesizing any other Boolean logic operation when complemented with a FALSE operation, i.e. an unconditional memristor RESET. It arose from the 'copy with inversion' operation first introduced by Kuekes et al. that was physically implemented by conditional resistive switching in a crossbar array, i.e. by applying appropriate voltages on the rows and columns of a crossbar such that the outcome of a resistive switching operation at one location depended on whether the resistance state at a different location was a high or low resistance [14]. The memristor-based IMP operation is expressed as $q' \leftarrow p$ IMP q, where p and q are two input bits and q' is the output bit, which replaces the second input bit. Although this approach is in principle capable of complete logic, the fact that one of the inputs is replaced with the output makes logic cascading using only stateful implication and FALSE difficult. For example, an XOR gate, which generates the sum output for a binary adder, can be executed by (p IMP q) IMP ((q IMP p) IMP 0). However, this compound operation requires that p and q be employed twice, so that both inputs need to be copied into other memristors in the crossbar in order for them to be re-used. Kvatinsky et al. proposed an improvement on this stateful logic approach that they called MAGIC (memristor-aided logic), which separates the output from the input cells and thus eliminates the need to copy inputs [7]. Adam *et al.* utilized a three-dimensional memristor crossbar array and demonstrated half-adder and full-adder operations composed of NAND and NOT instead of XOR and AND operations, where the former were executed without destroying the inputs [15]. In this way, the data overwriting problem can be avoided and logic cascading improved, but at the expense of requiring additional logic steps. In the original stateful logic approach introduced by Borghetti *et al.* [5], the $s \leftarrow p$ NAND q operation was executed by the unconditional RESET initialization of the target memristor ($s \leftarrow 0$), followed by two sequential IMP operations on q and s, and then p and s: p IMP (q IMP 0)), which required three memristors, only two of which were utilized during each step. Huang *et al.* developed stateful three-memristor logic operations [16], for which two memristor cells contained inputs that represented conditional states and the third memory bit was the output target. Thus, two IMP operations were merged and the total number of steps for a half-adder was reduced from fourteen to ten steps. They also showed that an AND gate was possible in two steps (the unconditional RESET initialization followed by conditional SET) in three-memristor stateful logic compared

to five steps for two-memristor operations: (p IMP (q IMP 0)) IMP 0.

By generalizing and combining the approaches above, we show here that there are actually four symmetry-related stateful two-memristor logic operations: the unconditional initialization of p (either TRUE or FALSE) with each of these outcomes followed by either a conditional SET or RESET of the bit q that depends on the state of p. For three-memristor gates, there are a total of eight possibilities: two conditional switching operations on s given four different initial states of p and q(TRUE TRUE, TRUE FALSE, FALSE TRUE and FALSE FALSE), which provide more logic gates that require fewer computational steps. Such stateful logic operates sequentially in the time domain on any bit addresses in a memristor array by connecting them to selected voltage sources from the system controller to define a particular gate, which is very different from conventional CMOS logic that utilizes a fixed spatial geometry of universal gates and latches. Accordingly, a larger selection of stateful logic gates should enable more efficient encoding and execution of a general computation. In addition, since the fundamental operations for stateful logic utilize fairly standard memory and control circuitry, this mode of computation is inherently reconfigurable and defect tolerant.

In this paper, we show diverse stateful logic gates that can be achieved by simply applying a coordinated sequence of voltages to the memristor array, which can significantly improve the computing efficiency. We describe the functioning of four symmetry-related two-memristor logic gates (IMP, OR, AND and NIMP) based on standard Ta/TaO_x/Pt memristors in a cross bar array [17], [18], and introduce the COPY and NOT operations required for logic cascading derived from these gates. In addition, we describe five realizable three-memristor gates: NOR, p NIMP q, q NIMP p, AND, and OR (underlined to distinguish them from the two-memristor gates) that can further reduce the number of discrete computational steps required for stateful logic. All these stateful logic gates are then experimentally demonstrated in a densely integrated 3×4 memristor crossbar. The performance and uniformity of the TaO_x memristors utilized in the crossbar here have been described elsewhere in the context of nonvolatile memory elements, with switching endurance greater than 10^9 SET/RESET cycles and highly accurate and reproducible resistance states demonstrated [19]–[22]. Finally, to demonstrate how a more complex operation can be composed using the family of stateful logic gates introduced here, we illustrate that a full adder operation is possible with less than or equal to thirteen sequential computational steps.

II. Stateful Two-Memristor Logic Gates

The two-memristor logic gates utilize two parallel resistive switches in a circuit with a series resistor. Figure 1a shows the basic circuit configuration of the components in a crossbar structure. Here, M_P and M_Q are the parallel memristors that store the logic inputs p and q, respectively, and R_R is the series resistor. The operating voltages V_P, V_Q, and V_R are applied to the bit lines of M_P and M_Q, and the word line connected to R_R, respectively, using circuitry appropriate for

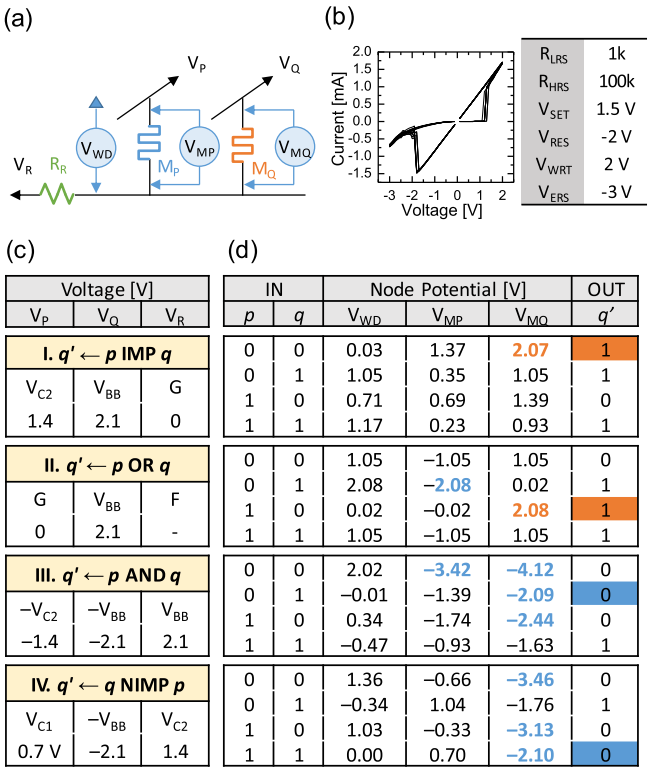


Fig. 1. Stateful two-memristor logic gates. (a) Schematic circuit diagram showing two parallel memristors and a series resistor that constitute the logic gates. (b) The resistance switching I-V curves of a Ta/TaO_x/Pt memristor and the switching parameters that implement the logic operations. (c) The applied voltage conditions for achieving the corresponding operations. (d) Calculated node potentials for the given conditioning states (p and q) and the resulting output (q'), with the colored bars indicating the bit that changes for each operation.

writing and erasing bits in a memory. In stateful two-memristor logic, the digital inputs are initially stored in M_P and M_Q, and the output is overwritten into one of the two memristors that compose the gate during the conditional logic operation. Here, the input bit that is replaced by the output is called the active bit, and the unchanged bit is called the passive bit. If p is the logic value of the passive bit and q is that of the active bit, the general logic operation is expressed as follows:

$$q' \leftarrow pFq \tag{1}$$

where p and q are the two input values, q' is the output overwritten on q, and F stands for a specific logic function.

The above stateful logic operation performs a conditional switching process on q depending on the initial values of p and q. For example, the IMP gate yields the conditional SET of q from $q = 0$ to $q' = 1$ only if p has a FALSE value ($p = 0$), where the low resistance state (LRS) and the high resistance state (HRS) of the memristor are assigned to the logic values 1 and 0, respectively. To set the stage for what follows, the operation of the stateful IMP gate is described briefly here. When V_P, V_Q, and V_R are applied as in Fig. 1a, depending on the state of M_P, the node voltage on M_Q will be approximately ($V_Q - V_P$) if M_P is in the LRS or ($V_Q - V_R$) if in the HRS. Therefore, by choosing the appropriate applied

voltage levels to make (V_Q– V_P) lower than the SET switching voltage and (V_Q– V_R) higher, M_Q will be SET only if M_P is in the HRS, which satisfies the definition $q' \leftarrow p$ IMP q.

The IMP gate changes the state of q corresponding to only one pair ($p = 0$ and $q = 0$) among four possible input states, so there should be three other symmetry-related two-memristor stateful logic operations that belong to the same family. To illustrate the potential gates, consider a Ta/TaO_x/Pt memristor. Figure 1b shows the memristive switching curves that implement the gates and summarizes the switching parameters. R_{LRS} and R_{HRS} are the resistance values in the LRS and HRS that correspond to the logic values 1 and 0, respectively. V_{SET} and V_{RST} are the transition voltages from the HRS to the LRS and from the LRS to the HRS, respectively. V_{WRT} and V_{ERS} are the unconditional writing and erasing voltages, respectively, which are higher in magnitude than V_{SET} and V_{RST}. For reliable operation of the logic gates, the following criteria need to be satisfied:

1) For a memristor to be unconditionally switched to the LRS or the HRS, it must be biased to a voltage amplitude higher than V_{WRT} or V_{ERS}, respectively, for guaranteed SET or RESET initialization of an input bit value for the logic gate.
2) For a memristor to remain in the LRS or HRS during a logic operation, the applied voltage magnitude must be lower than V_{SET} or V_{RES}, respectively, to prevent an unwanted switching event.

Therefore, for practical operation, no positive voltages between V_{SET} and V_{WRT} (from +1.5 V to +2.0 V in Fig. 1b) or negative voltages between V_{RES} and V_{ERS} (from −2 V to −3 V in Fig. 1b) should be applied to any memristors to avoid unintended or unstable switching.

With these guidelines, the required operating characteristics of stateful two-memristor logic operations can be calculated from experimentally measured parameters, given that the individual devices yield reproducible and uniform results within some error tolerance. To implement the full set of operations, four voltage levels are required: V_{BB} (the highest bit line voltage, set to 2.1 V here), V_{C1} (= 1/3 V_{BB}), V_{C2} (= 2/3 V_{BB}) and G (= 0 V). In this study, the value of R_R is fixed to R_{LRS} (∼1 kΩ), but other resistance values may be chosen in order to optimize logic performance or voltage margins. Also, the series resistance can be adjusted to compensate for the range of leakage currents that may exist for a particular system. Figure 1c shows the voltage input conditions to define each of the four logic gates. Figure 1d shows the calculated node potentials at the word line (V_{WD} in Fig. 1a), M_P and M_Q (V_{MP} and V_{MQ} in Fig. 1a, respectively) for the given input bits (p and q) and the voltage conditions given in Fig. 1c, and the resulting value of the output bit q'. In Fig. 1d, the node voltage amplitudes exceeding V_{WRT} (2 V) and V_{RES} (−2 V) are marked in red and blue bold fonts, respectively. The applied V_{WRT} (2 V) was chosen as described above to guarantee unconditional SET switching, whereas V_{RES} (−2 V) was chosen for the RESET switching baseline instead of V_{ERS} (−3 V), which requires some explanation. Consider a series connection of one memristor (M_P) in the LRS (chosen to be 1kΩ) and one resistor ($R_R = R_{LRS}$, i.e. also 1kΩ), as shown in Fig. 1a, and assume V_P and V_R, respectively, are applied at the nodes. The potential (V_P – V_R) is divided equally between M_P and R_R by the voltage divider effect. Resetting of M_P is triggered when (V_P – V_R)/2 becomes −2 V, which is the V_{RES} threshold. After M_P resets, its resistance is increased, which leads to a spontaneous increase of the node potential on M_P caused by the change of the voltage divider. If the resistance increase of M_P is just a factor of 3, which is a relatively small change for a TaO_x memristor, M_P can experience a potential of at least −3 V out of the total −4 V applied voltage. Because of the potential redistribution, the value of −2 V for V_{RES} is large enough to guarantee stable RESET switching.

The IMP gate from Ref. 5 is obtained by setting V_P, V_Q and V_R to V_{C2}, V_{BB} and G, respectively. In this configuration, the node voltage on M_Q exceeds the V_{WRT} only if $p = 0$ and $q = 0$; otherwise, both the node voltages on M_P and M_Q are suppressed below V_{SET}. Another conditional SET is possible that can change the input $q = 0$ to output $q' = 1$ only if $p = 1$, which is the definition of a logical OR gate. For this operation, V_P and V_Q are set to G and V_{BB}, respectively, while V_R is floated. This forms a series connection of M_P and M_Q through the word line, where the two memristors have opposite switching polarity. The potential difference between the two bit lines (V_P – V_Q ∼ V_{WRT}) is divided between the memristors in proportion to their resistance. If both states are equal, i.e. either $p = q = 0$ or $p = q = 1$, then V_{WRT} is equally distributed across the two memristors and thus neither can switch. If $p = 1$ and $q = 0$, most of V_{WRT} drops across M_Q, which will then switch from 0 to 1. If $p = 0$ and $q = 1$, V_{WRT} drops across M_P but with opposite polarity corresponding to reset switching, which is a null operation since M_P is already in the HRS.

Similarly, conditional RESET is also possible using appropriate voltage conditions to implement both AND and NIMP gates. The bias conditions and the node voltages for both gates are also shown in Figures 1c and 1d, respectively. For the AND gate, V_P, V_Q and V_R are set to –V_{C2}, –V_{BB} and V_{BB}, respectively. Then, as shown in Figure 1d, $q = 1$ is changed to $q' = 0$ only if $p = 0$. Although the reset voltage applied to the HRS has no effect on a logic operation, a voltage amplitude as high as −4.12 V may cause breakdown of the device. This is prevented by the self-limiting RESET switching behavior associated with the voltage divider effect [23]. For the NIMP gate, V_P, V_Q and V_R are set to V_{C1}, –V_{BB} and V_{C2}, respectively, which results in $q = 1$ being changed to $q' = 0$ only if $p = 1$. The voltage conditions shown in Figure 1c for demonstrating the conditional SET and RESET are easily realized by the circuit. Given the four symmetry-related gates IMP, OR, AND and NIMP, it is in principle possible to construct serial logic operations that require significantly fewer steps and thus take less time and energy to complete than if only stateful IMP is available.

III. Non-Destructive Two-Memristor Logic Cascading

Efficient logic cascading should satisfy two conditions: 1) the input bits should not be overwritten so that they can

(a)

$s' \leftarrow$ COPY q	M_Q	M_S
Step 1: $s \leftarrow 0$ (RESET)	q	0
Step 2 : $s' \leftarrow q$ OR s	q	q
Step 1: $s \leftarrow 1$ (SET)	q	1
Step 2 : $s' \leftarrow q$ AND s	q	q

(b)

$s' \leftarrow$ COPY $\neg q$	M_Q	M_S
Step 1: $s \leftarrow 0$ (RESET)	q	0
Step 2 : $s' \leftarrow q$ IMP s	q	$\neg q$
Step 1: $s \leftarrow 1$ (SET)	q	1
Step 2 : $s' \leftarrow s$ NIMP p	q	$\neg q$

Fig. 2. COPY and NOT gates. (a) two methods for achieving $s' \leftarrow$ COPY q. (b) two methods for achieving $s' \leftarrow$ COPY $\neg q$ (= NOT q).

be used multiple times, and 2) the output bit should be stored at a designated memory address for easy data access and subsequent logic operations. Therefore, the optimum condition for logic cascading can be expressed as

$$s \leftarrow p\mathrm{F}q \tag{2}$$

where p and q are the two inputs, and s is the third designated bit for recording the output. To achieve Eq. (2), a data copy operation for any original input bit to a target address is required to clone the active bit. The sequence using the copy operation followed by a two-memristor gate requires two steps:

1) Copy the datum in the active bit (e.g. q) to s, $s \leftarrow q$
2) Execute the logic operation using p and s, $s' \leftarrow p$ F s

In this way, the output of the logic operation between p and q can be recorded to s without destroying the active input q.

Figure 2a shows two ways to implement a COPY with memristor gates (OR or AND), depending on the initial value (HRS or LRS) of the target bit, s. Both sequences that copy the datum of q to s can be expressed as follows:

$$s' \leftarrow \text{COPY } q \text{ (or } s' \leftarrow q) \tag{3}$$

Similarly, the two-memristor gates IMP and NIMP can be used for negation after the SET or RESET initialization of s, respectively, and generate NOT q gates, which is shown in Fig. 2b. Both sequences that copy the negation of the datum of q to s can be expressed as follows:

$$s' \leftarrow \text{NOT } q \text{ (or } s' \leftarrow \neg q) \tag{4}$$

Consequently, the logic operation F is rendered non-destructive via the COPY (or NOT) operation followed by one of the two-memristor gates. Furthermore, the COPY operation can transfer data from either p or q to s, so this approach allows the outputs of the input-order-dependent gates (IMP and NIMP) to be recorded to the designated bit, s, which was not the case without the copy operation; i.e. the outputs of p IMP q and q IMP p were recorded to q' and p', respectively. Therefore, the COPY operation not only enables logic cascading but also facilitates the data handling.

In this fashion, thirteen out of the total of sixteen two-input Boolean logic operations can in principle be executed within three steps including initialization, COPY or NOT, and one of the four symmetry-related two-memristor gates. The TRUE and FALSE operations correspond to the unconditional SET and RESET memristor initializations, respectively, so both require only one step. The COPY p, COPY q, NOT p and NOT q operations require two steps, while p OR q (= q OR p), p AND q (= q AND p), p IMP q, q IMP p, p NIMP q and q NIMP p can be executed by copying either q or p to s followed by the application of the corresponding two-memristor gate, so they each require three steps. If the negation of p or q is copied to s using the NOT q operation, the two-memristor gates yield the following operations: p OR $\neg q$ and q OR $\neg p$ yield q IMP p and p IMP q, respectively; p AND $\neg q$ and q AND $\neg p$ yield p NIMP q and q NIMP p, respectively; both p IMP $\neg q$ and q IMP $\neg p$ yield p NAND q; and both p NIMP $\neg$ and q NIMP $\neg p$ yield p AND q. We thus observe the detailed symmetry relationships for this family of stateful memristor logic gates. The remaining three Boolean operations, NOR, EQUAL and XOR, require at least four memory cells using two-memristor gates. The NOR operation is equivalent to applying the OR gate first followed by NOT, which requires five steps (three steps for OR and two steps for NOT) according to the following sequence:$t' \leftarrow$ COPY q; $t'' \leftarrow p$OR t'; $s' \leftarrow$ NOT t''. The EQUAL and XOR operations each require seven steps: EQUAL corresponds to $s \leftarrow$ COPY q; $t \leftarrow$ COPY q; $s' \leftarrow (p$ OR $t)$ IMP $(p$ AND $s)$; and XOR to $s \leftarrow$ COPY q; $t \leftarrow$ COPY q; $s' \leftarrow (p$ OR $t)$ NIMP $(p$ AND $s)$, which are summarized in Figure 3.

In summary, the family of stateful two-memristor logic gates enable the COPY and NOT operations that make more efficient logic cascading possible. Moreover, they reduce the number of steps required for executing many serial logic operations. However, the complexity of executing two-memristor NOR, EQUAL and XOR logic operations provides an incentive to look at other gate structures and operating procedures that may be even more efficient.

IV. Stateful Three-Memristor Logic Gates

Three-memristor logic gates are also possible and potentially very useful. Figure 4a shows the schematic configuration for a three-memristor logic operation, for which three memristors (M_P, M_Q and M_S) with logic values p, q and s, respectively, are connected in parallel to share the same word line with one series resistor (R_R). The three-memristor logic operations are executed as follows:

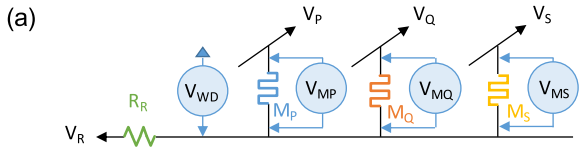
1) Initialize s; $s \leftarrow 0$ (RESET)
2) Execute $\underline{\mathrm{F}}$; $s' \leftarrow p\underline{\mathrm{F}}\, q$

where both p and q are passive input bits in this case, s is the active output bit, and $\underline{\mathrm{F}}$ (underlined F) stands for the specific logic operation performed by the manipulation of voltages in a crossbar containing three memristors that implement the gate.

Three-memristor operations provide more degrees of freedom and a larger number of potential gates. However, because of the practical upper limit to the voltage source magnitude, only some of the possibilities can be implemented within

	Input	
p	1 1 0 0	
q	1 0 1 0	
Gates	**Output**	**Two-memristor Gate Sequences**
TRUE	1 1 1 1	$s \leftarrow$ SET
p OR q	1 1 1 0	$s' \leftarrow$ COPY q; $s'' \leftarrow p$ OR s'
q IMP p	1 1 0 1	$s' \leftarrow$ COPY p; $s'' \leftarrow q$ IMP s'
p	1 1 0 0	$s' \leftarrow$ COPY p
p IMP q	1 0 1 1	$s' \leftarrow$ COPY q; $s'' \leftarrow p$ IMP s'
q	1 0 1 0	$s' \leftarrow$ COPY q
p EQUAL q	1 0 0 1	$s' \leftarrow$ COPY q; $t' \leftarrow$ COPY q; $t'' \leftarrow p$ OR t'; $s'' \leftarrow p$ AND s'; $s''' \leftarrow t''$ IMP s''
p AND q	1 0 0 0	$s' \leftarrow$ COPY q; $s'' \leftarrow$ p AND s'
p NAND q	0 1 1 1	$s' \leftarrow$ NOT q; $s'' \leftarrow$ p IMP s'
p XOR q	0 1 1 0	$s' \leftarrow$ COPY q; $t' \leftarrow$ COPY q; $s'' \leftarrow p$ AND s'; $t'' \leftarrow p$ OR t'; $s''' \leftarrow t''$ NIMP s''
NOT q	0 1 0 1	$s' \leftarrow$ NOT q;
p NIMP q	0 1 0 0	$s' \leftarrow$ COPY q; $s'' \leftarrow p$ NIMP s'
NOT p	0 0 1 1	$s' \leftarrow$ NOT p
q NIMP p	0 0 1 0	$s' \leftarrow$ COPY q; $s'' \leftarrow q$ NIMP s'
p NOR q	0 0 0 1	$t' \leftarrow$ COPY q; $t'' \leftarrow p$ OR t'; $s' \leftarrow$ NOT t''
FALSE	0 0 0 0	$s \leftarrow$ RESET

Fig. 3. Sequences for executing the 16 Boolean logic gates by stateful two-logic gates. Note that the COPY and NOT gates constitute two steps including one initialization step (either SET or RESET) and one two-memristor gate.



(b) (c)

Voltage [V]				IN			Node Potential [V]				OUT
V_P	V_Q	V_S	V_R	p	q	s	V_{WD}	V_{MP}	V_{MQ}	V_{MS}	s'
I. $s' \leftarrow p$ NOR q											
V_{C2}	V_{C2}	V_{BB}	G	0	0	0	0.05	1.35	1.35	2.05	1
1.4	1.4	2.1	0	0	1	0	0.71	0.69	0.69	1.39	0
				1	0	0	0.71	0.69	0.69	1.39	0
				1	1	0	0.94	0.46	0.46	1.16	0
II. $s' \leftarrow q$ NIMP p											
G	V_{C2}	V_{BB}	F	0	0	0	1.16	0.24	−1.16	0.94	0
0	1.4	2.1	-	0	1	0	0.03	1.37	−0.03	2.07	1
				1	0	0	1.39	0.01	−1.39	0.71	0
				1	1	0	0.71	0.69	−0.71	1.39	0
III. $s' \leftarrow p$ NIMP q											
V_{C2}	G	V_{BB}	F	0	0	0	1.16	−1.16	0.24	0.94	0
1.4	0	2.1	-	0	1	0	1.39	−1.39	0.01	0.71	0
				1	0	0	0.03	−0.03	1.37	2.07	1
				1	1	0	0.71	−0.71	0.69	1.39	0
IV. $s' \leftarrow p$ AND q											
$-V_{C2}$	$-V_{C2}$	V_{BB}'	V_{BB}	0	0	0	2.03	−3.43	−3.43	−0.23	0
−1.4	−1.4	1.8	2.1	0	1	0	0.35	−1.75	−1.75	1.45	0
				1	0	0	0.35	−1.75	−1.75	1.45	0
				1	1	0	-0.23	−1.17	−1.17	2.03	1
V. $s' \leftarrow p$ OR q											
G	G	V_{BB}	F	0	0	0	0.70	−0.70	−0.70	1.40	0
0	0	2.1	-	0	1	0	0.02	−0.02	−0.02	2.08	1
				1	0	0	0.02	−0.02	−0.02	2.08	1
				1	1	0	0.01	−0.01	−0.01	2.09	1

Fig. 4. Stateful three-memristor logic gates. (a) Schematic circuit diagram showing three parallel memristors and a series resistor that constitute the logic gates. (**b**) The applied voltage conditions for achieving the corresponding operations. (**c**) Calculated node potentials for the given conditioning states (p, q and s) and the resulting output (s'), with the colored bars indicating the bit(s) that change for each operation.

the constraints of the memory circuit. For the Ta/TaO_x/Pt memristor in Figure 1b, the five realizable operations are q NOR p, q NIMP p, p NIMP q, q AND p and q OR p. The first four gates are those that change the state of s from 0 to 1 only if the conditional values of p and q are (00), (01), (10) or (11), respectively. These four operations can be executed after initializing s to 0 by applying appropriate voltages to V_P, V_Q, V_S and V_R, for which the bias conditions are shown in Figure 4b. Figure 4c shows the node potentials at the given inputs and applied voltage conditions, and the resulting output at s'. In the first tables in Figures 4b and 4c, the conditional SET of M_S is possible only if both M_P and M_Q are in the HRS and by setting $V_P = V_Q = V_{C2}$, $V_S = V_{BB}$ and V_R = G, which results in the NOR gate. If at least one of p or q is in the LRS, V_P or V_Q can increase the word line potential (V_{WD}) and consequently decrease the node voltage on M_S (V_{MS}) below V_{SET}. In the second and third tables in Figures 4b and 4c, setting V_P = G and $V_Q = V_{C2}$ yields q NIMP p, and $V_P = V_{C2}$ and V_Q = G yields p NIMP q, providing that $V_S = V_{BB}$ and V_R is floating. Under these conditions, the node voltage on M_S exceeds V_{WRT} only if ($p = 0$ and $q = 1$) or ($p = 1$ and $q = 0$). In the fourth table in Figures 4b and 4c, the AND gate is possible by setting $V_P = V_Q = -V_{C2}$, $V_S = V'_{BB}$ and $V_R = V_{BB}$, where V'_{BB} is 1.8 V, which is an additional required voltage level. Finally, as shown in the fifth table in Figures 4b and 4c, the OR gate is also readily obtained by applying $V_P = V_Q$ = G, $V_S = V_{BB}$ and floating V_R (another possible condition for the OR gate is $V_P = V_Q = -V_{C1}$, $V_S = V_{BB}$ and $V_R = V_{C2}$).

The advantage of these three-memristor logic gates is that they further reduce the number of steps for sequential logic; OR, AND, NOR and NIMP can be executed without an explicit COPY or NOT step, and thus require only two steps each. In addition, the EQUAL and XOR operations can now be performed within three memristors and three steps. The EQUAL operation is achieved by executing NOR and AND gates in any sequence, and the XOR operation requires q NIMP p and p NIMP q operations after the RESET initialization of s. These are summarized in Figure 5. Figure 6 compares the number of sequential steps using two-memristor gates only and the optimum combination of two- and three-memristor gates required to synthesize all sixteen Boolean operations on two inputs, with the average number of steps being 3.1 and 2.2, respectively. This demonstrates that adopting the three-memristor gates can on average be ∼29 % more efficient in terms of the number of computational steps even before taking into consideration the decreased number of memristors required for a calculation.

V. Experimental Demonstration of The Basic Gates

The successful experimental operation of the stateful two- and three-memristor gates described above within an

	Input
p	1 1 0 0
q	1 0 1 0

Gates	Output	Three-memristor Gate Sequences
TRUE	1 1 1 1	-
p OR q	1 1 1 0	$s \leftarrow$ RESET; $s' \leftarrow p$ OR q
q IMP p	1 1 0 1	-
p	1 1 0 0	-
p IMP q	1 0 1 1	-
q	1 0 1 0	-
p EQUAL q	1 0 0 1	$s \leftarrow$ RESET; $s' \leftarrow p$ NOR q; $s'' \leftarrow p$ AND q
p AND q	1 0 0 0	$s \leftarrow$ RESET; $s' \leftarrow p$ AND q
p NAND q	0 1 1 1	-
p XOR q	0 1 1 0	$s \leftarrow$ RESET; $s' \leftarrow q$ NIMP p; $s'' \leftarrow p$ NIMP q
NOT q	0 1 0 1	-
p NIMP q	0 1 0 0	$s \leftarrow$ RESET s; $s' \leftarrow p$ NIMP q
NOT p	0 0 1 1	-
q NIMP p	0 0 1 0	$s \leftarrow$ RESET s; $s' \leftarrow q$ NIMP p
p NOR q	0 0 0 1	$s \leftarrow$ RESET s; $s' \leftarrow p$ NOR q
FALSE	0 0 0 0	$s \leftarrow$ RESET

Fig. 5. Sequences for executing the 16 Boolean logic gates by stateful three-memristor logic gates. Dash (−) means that the number of steps using three-memristor is not reduced compared to the two-memristor gate.

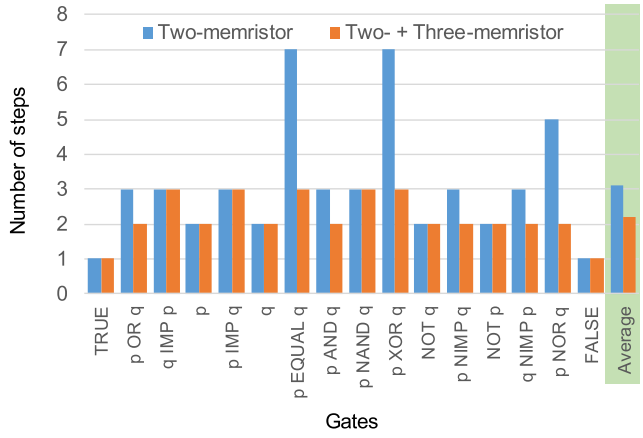


Fig. 6. Efficiency of two-memristor gates vs. combined two- and three-memristor gates. The number of computational steps for executing all 16 Boolean logic operations for two inputs using only the two-memristor gates (blue) and the optimal combination of two- and three-memristor gates (red) are plotted. The average values are compared on the right of the graph.

integrated 3 × 4 crossbar is summarized in Figure 7. This demonstration utilized a minimum sized but fully populated crossbar to show that the gate operations are experimentally feasible. For operations embedded in a larger crossbar, additional constraints and use of a different series resistance may be necessary to compensate for possible sneak path currents. Figure 7a shows the device configuration for the electrical testing where four bias voltages (V_P, V_Q, V_S, and V_R) can be applied to the four contacts independently. In each column of the array in Figure 7a, three memristors (M_P, M_Q, and M_S) sharing the same word line can be utilized

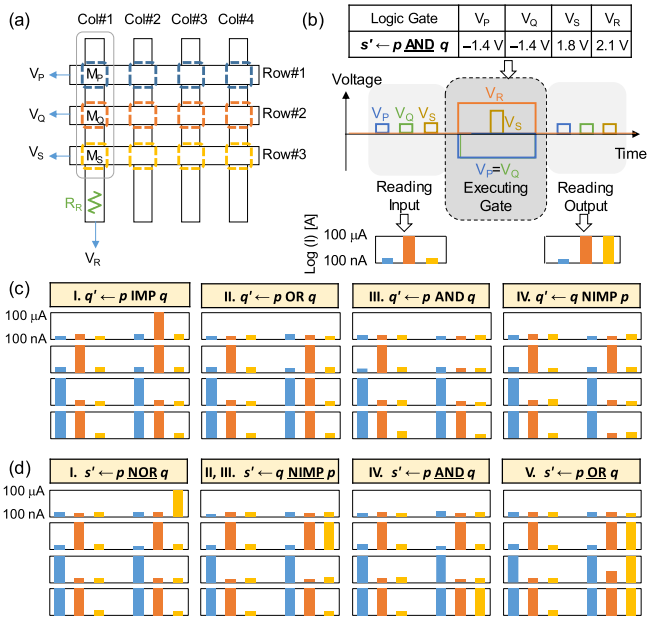


Fig. 7. Experimental validation of the two- and three-memristor gates. (a) a schematic of the 3 × 4 crossbar array prepared for the demonstration. Three bits sharing the same column compose one gate. The four columns are for four conditional inputs. (b) The voltage application process for reading input, executing the logic gate, and reading the output. (c) and (d) show the read data of logic inputs and outputs of two-memristor and three-memristor gates, respectively.

as a configurable stateful logic gate. The intrinsic resistance of the wire connecting each column to the respective contact pad determines R_R. Before implementing the logic gates, different input values [M_P, M_Q, M_S] were stored in each column: [000] in Col#1, [010] in Col#2, [100] in Col#3, and [110] in Col#4. The selected column was biased to V_R while others were floated. Figure 7b shows the biasing sequences on the four contacts to read the input data, implement the logic gate, and read the output data. For the input READs, a quasi-DC step-voltage with amplitude 0.2 V and width 100 μs was applied sequentially to V_P, V_Q, and V_S while V_R was grounded, and the resulting currents were measured to determine M_P, M_Q and M_S. In the logic execution step, the appropriate voltages were supplied for V_P and V_R for the two-memristor gates, or for V_P, V_Q, and V_R for the three-memristor gates. Initially, only DC voltages for the passive bit or bits were applied. Figure 7b shows the example of AND, which is the most complicated three-memristor gate; a constant −1.4 V was supplied for V_P and V_Q, and 2.1 V for V_R, as prescribed in Fig. 4d. Then, the bias for the active bit, V_S for the three-memristor gates (V_Q for the two-memristor gates), was ramped up from 0 V to 1.8 V for 100 μs, which triggered the AND operation. In this way, only one bias voltage required active control while the other biases were fixed, which allowed simple and reproducible gate operation in terms of the voltage timing and also demonstrated that conditional switching only occurred when all of the appropriate bias voltages were simultaneously applied to define the gate. After the logic operations, the output states were measured and the data were recorded. These steps were repeated over the four columns and

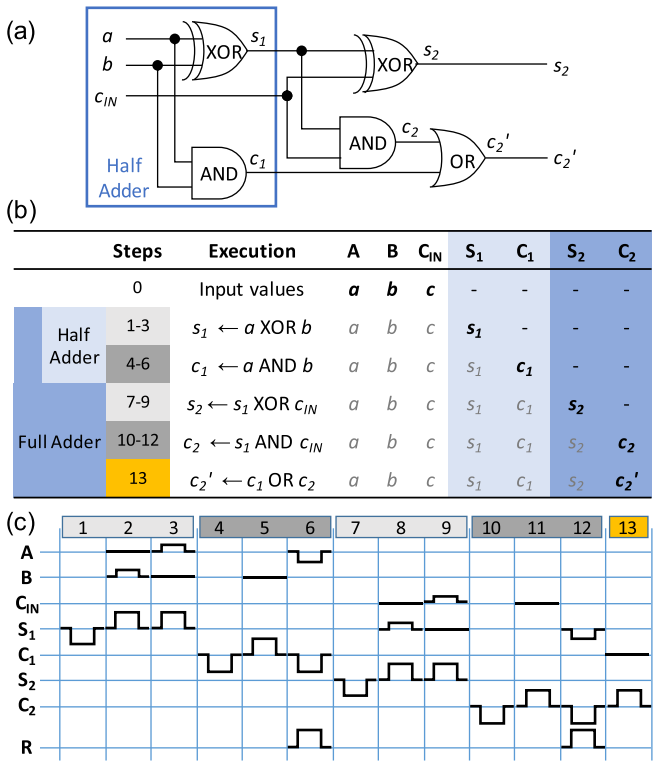


Fig. 8. Full adder execution utilizing the stateful logic gates. (a) The circuit diagram of the two-bit full adder. The half-adder portion is indicated by the blue rectangle. (b) A sequence for obtaining the sum and carry outputs for the full adder. Only 13 steps are required for this calculation, and further improvements are possible. (c) A signal timing diagram during the 13 steps for executing the full adder.

the resulting data were collected for each gate. Figure 7c summarizes the measured input and output currents for the four two-memristor gates, which confirmed they all performed as predicted in Fig. 1. The constant FALSE value in M_S is shown for reference in these cases. Figure 7d shows the input and output data for the three-memristor gates NOR, NIMP, AND and OR ($s' \leftarrow p$ NIMP q is not shown because it is essentially the same as $s' \leftarrow q$ NIMP p except for the order of p and q). These experiments demonstrated that the resistance states were uniform enough and the voltage margins within a tolerance that produced correct logic operations across the twelve memristors in the dense crossbar used for the experiments.

VI. Full Adder Execution

By combining both two-memristor and three-memristor gates, efficient logic cascading can be achieved. Here, we illustrate the logic sequences that would be required for half and full adders. Figure 8a shows the schematic full adder circuit composed of two XOR, two AND and one OR gates, where the half adder portion is indicated by a blue rectangle. The two inputs and the carry value are denoted by a, b and c_{IN}, respectively. Figure 8b and 8c show the logic steps and signal timing diagram for the full adder. The first two operations are $s_1 \leftarrow a$ XOR b and $c_1 \leftarrow a$ AND b, which produce the sum and carry outputs, respectively, and thus represents the half adder. Here, the three steps needed to implement the two-memristor AND have been selected over the two steps for the three-memristor AND because the latter requires an additional voltage level that adds complexity to the circuit. Even with this limitation, only 6 steps are required for the half-adder. The next operation is $s_2 \leftarrow s_1$ XOR c_{IN}, which yields the sum output of the full adder. The carry output of the full adder can be obtained by executing $c_2 \leftarrow s_1$ AND c_{IN} followed by $c_2' \leftarrow c_1$ OR c_2. Consequently, the sum and carry outputs of the full adder operations can be obtained within thirteen computational steps. If all the bits in an array are initialized to 0 as the default and the output of the logic operation is targeted to a previously unwritten region, the FALSE operations can be skipped and the required number of steps would be reduced to nine. If the three-memristor AND operation is available, the number of steps could be further reduced to seven. These are significant computational efficiency improvements over IMP-only stateful logic, which requires 35 steps (13 RESET initialization and 22 IMP steps) for executing a full adder [15].

VII. Conclusions

We have described a family of symmetry-related stateful logic operations possible by applying appropriate voltages on the bit and word lines of a dense crossbar memory that induce a conditional SET or RESET in one memristor depending on the state, FALSE or TRUE, of one or two others. Then we showed an experimental demonstration of four two-memristor and five three-memristor stateful logic gates using a densely integrated 3×4 Ta/TaO_x/Pt memristor crossbar. These stateful gates enable data cloning from any memory location to a targeted bit address in the same row or column. Combined with COPY and NOT operations, all sixteen two-input Boolean logic operations are theoretically possible within three computational steps using three memristors, which provides an efficient means for logic cascading. In the case of executing a two-bit full adder, the demonstrated approach could require as few as seven computational steps, which is a remarkable reduction compared to the 35 steps for the originally proposed stateful IMP logic.

We have thus shown that it is possible to perform complex compound Boolean logic operations on input data stored inside a nonvolatile memristor crossbar through the application of voltages to appropriate bit and word lines that then conditionally write the output to a predetermined memory cell. Given the serial nature of the operations, the absolute computational speed of such a system will be relatively low. However, for IoT or mobile applications where cost is critical, power is restricted or even intermittent, and the required computational throughput is low, memristor-based logic could be a convenient approach for limited computation. Since the stateful logic gates are actually reconfigurable combinations of memory cells, the data processing is inherently defect tolerant through the use of standard techniques for utilizing redundancy and wear leveling to compensate for manufacturing errors and circuit element failures.

References

[1] L. Chua, "Memristor-the missing circuit element," *IEEE Trans. Circuit Theory*, vol. CT-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] J. J. Yang, D. B. Strukov, and D. R. Stewart, "Memristive devices for computing," *Nature Nanotechnol.*, vol. 8, no. 1, pp. 13–24, Jan. 2013.
[4] D. S. Jeong, K. M. Kim, S. Kim, B. J. Choi, and C. S. Hwang, "Memristors for energy-efficient new computing paradigms," *Adv. Electron. Mater.*, vol. 2, no. 9, Sep. 2016, Art. no. 1600090.
[5] 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 implication," *Nature*, vol. 464, pp. 873–876, Apr. 2010.
[6] J. Reuben *et al.*, "Memristive logic: A framework for evaluation and comparison," in *Proc. 27th Int. Simulation Power Timing Modeling, Optim. Simulation (PATMOS)*, Sep. 2017, pp. 1–8.
[7] S. Kvatinsky *et al.*, "MAGIC—Memristor-aided logic," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 61, no. 11, pp. 895–899, Nov. 2014.
[8] S. Kvatinsky, G. Satat, N. Wald, 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) Syst.*, vol. 22, no. 10, pp. 2054–2066, Oct. 2014.
[9] N. Talati, S. Gupta, P. Mane, and S. Kvatinsky, "Logic design within memristive memories using memristor-aided loGIC (MAGIC)," *IEEE Trans. Nanotechnol.*, vol. 15, no. 4, pp. 635–650, Jul. 2016.
[10] K. M. Kim *et al.*, "Single-Cell stateful logic using a dual-bit memristor," *Phys. Status Solidi Rapid Res. Lett.*, vol. 13, no. 3, Mar. 2019, Art. no. 1800629.
[11] D. Ielmini and H. S. P. Wong, "In-memory computing with resistive switching devices," *Nature Electron.*, vol. 1, no. 6, pp. 333–343, 2018.
[12] Z. Sun, E. Ambrosi, A. Bricalli, and D. Ielmini, "Logic computing with stateful neural networks of resistive switches," *Adv. Mater.*, vol. 30, no. 38, Sep. 2018, Art. no. 1802554.
[13] C. S. Hwang, "Prospective of semiconductor memory devices: From memory system to materials," *Adv. Electron. Mater.*, vol. 1, no. 6, Jun. 2015, Art. no. 1400056.
[14] P. J. Kuekes, D. R. Stewart, and R. S. Williams, "The crossbar latch: Logic value storage, restoration, and inversion in crossbar circuits," *J. Appl. Phys.*, vol. 97, Feb. 2005, Art. no. 034301.
[15] G. C. Adam, B. D. Hoskins, M. Prezioso, and D. B. Strukov, "Optimized stateful material implication logic for three-dimensional data manipulation," *Nano Res.*, vol. 9, no. 12, pp. 3914–3923, Dec. 2016.
[16] P. Huang *et al.*, "Reconfigurable nonvolatile logic operations in resistance switching crossbar array for large-scale circuits," *Adv. Mater.*, vol. 28, no. 44, pp. 9758–9764, Nov. 2016.
[17] K. M. Kim *et al.*, "Voltage divider effect for the improvement of variability and endurance of TaO_x memristor," *Sci. Rep.*, vol. 6, Feb. 2016, Art. no. 020085.
[18] K. M. Kim *et al.*, "Low variability resistor–memristor circuit masking the actual memristor states," *Adv. Electron. Mater.*, vol. 1, no. 6, Jun. 2015, Art. no. 1500095.
[19] E. J. Merced-Grafals, N. Dávila, N. Ge, R. S. Williams, and J. P. Strachan, "Repeatable, accurate, and high speed multi-level programming of memristor 1T1R arrays for power efficient analog computing applications," *Nanotechnology*, vol. 27, no. 36, Aug. 2016, Art. no. 365202.
[20] M. Hu *et al.*, "Memristor-based analog computation and neural network classification with a dot product engine," *Adv. Mater.*, vol. 30, no. 9, Mar. 2018, Art. no. 1705914.
[21] F. Miao *et al.*, "Anatomy of a nanoscale conduction channel reveals the mechanism of a high-performance memristor," *Adv. Mater.*, vol. 23, no. 47, pp. 5633–5640, Nov. 2011.
[22] J. J. Yang *et al.*, "High switching endurance in TaO_x memristive devices," *Appl. Phys. Lett.*, vol. 97, no. 23, Dec. 2010, Art. no. 232102.
[23] K. M. Kim, S. R. Lee, S. Kim, M. Chang, and C. S. Hwang, "Self-limited switching in Ta_2O_5/TaO_x memristors exhibiting uniform multilevel changes in resistance," *Adv. Funct. Mater.*, vol. 25, no. 10, pp. 1527–1534, Mar. 2015.

Kyung Min Kim (M'08) was born in Seoul, South Korea, in 1981. He received the B.S. degree in materials science and engineering and the Ph.D. degree in nature science from Seoul National University, Seoul, in 2003 and 2008, respectively.

From 2008 to 2011, he was a Post-Doctoral Researcher with the Inter-University Semiconductor Research Center, Seoul National University. From 2011 to 2013, he was a Senior Engineer with the Samsung Advanced Institute of Technology, Samsung Electronics, South Korea. From 2014 to 2017, he was a Research Scientist with Hewlett Packard Labs, Hewlett Packard Enterprise. Since 2017, he has been an Associate Professor with the Material Science and Engineering Department, KAIST, Daejeon, South Korea. He has authored more than 50 articles, and more than 20 inventions. His research interests include memristor applications for the future computing such as in-memory computing devices and neuromorphic devices.

R. Stanley Williams (SM'08) received the B.A. degree in chemical physics from Rice University in 1974 and the Ph.D. degree in physical chemistry from U. C. Berkeley in 1978. He was a Member of Technical Staff with AT&T Bell Laboratories from 1978 to 1980, and a Faculty Member of the Chemistry Department, UCLA, from 1980 to 1995, and held various positions at HP Labs from 1995 to 2018. He is currently a Professor with the Electrical and Computer Engineering Department, Texas A&M University. Until, he holds more than 220 U.S. patents and has published more than 450 papers in reviewed scientific journals. His current research is in the areas of nonlinear dynamics and neuromorphic computation. He has been recognized for business, scientific, and academic achievement, including being named one of the top ten visionaries in the field of electronics by EETimes, the 2014 IEEE Outstanding Engineering Manager Award, the 2009 EETimes Innovator of the Year ACE Award, the 2007 Glenn T. Seaborg Medal for contributions to Chemistry, the 50th Anniversary Laureate Lecturer on Electrical and Optical Materials for the TMS, the 2004 Herman Bloch Medal for Industrial Research, the Inaugural Scientific American 50 Top Technology leaders in 2002, and the 2000 Julius Springer Award for *Applied Physics*.