A Stateful Logic Family Based on a New Logic Primitive Circuit Composed of Two Antiparallel Bipolar Memristors

Nuo Xu, Tae Gyun Park, Hae Jin Kim, Xinglong Shao, Kyung Jean Yoon, Tae Hyung Park, Liang Fang, Kyung Min Kim,* and Cheol Seong Hwang*

Stateful logic enables highly energy-efficient computation because the time and energy consumption for data transfer between the memory and the processing units in the traditional computation system can significantly be saved due to the combined functionalities of logic operation and nonvolatile memory. The logic primitive circuit, usually composed of resistive switching memory or memristor units, is the fundamental and kernel element to build a stateful logic family. However, the current stateful logic primitive circuits cannot be configured between the devices from the adjacent layers of the 3D crossbar array (CBA) which largely limited its application in high-density and capacity memory arrays. Herein, a new stateful logic primitive circuit is presented based on the structure of two antiparallel bipolar memristors. The presented logic primitive circuit enables a complete set of stateful logic operation and is compatible with a 3D CBA. The working principle and validity of the logic design were experimentally demonstrated by either a real TiO₂-based memristive circuit or the HSPICE simulation. Furthermore, a space-time-wise cascading method was demonstrated by XOR and the full adder functions based on a CBA, and its merits were further elucidated through comparison with different logic families with various cascading methods.

1. Introduction

Ever since the publication of Turing’s seminal paper on “Entscheidungsproblem” in 1936, where the concept of the automatic general computing machine was first introduced,[1] the processor and memory in a computer have been individually developed with a higher emphasis on the processor.[2] While information technology (IT) has enjoyed the flourish of such technological improvements during the past ≈80 years, it is time to reconsider the conventional paradigm, which can be well recognized based on the early ending of Dennard scaling in 2005 and on the currently nearly ending Moore’s law.[3] A clear demand in this reconsideration is caused by the fact that the power consumption of the current computation system will increase to an unsustainable level due to the physical limitation of the feature size shrinking of the current workhorse device—complementary metal oxide semiconductor field-effect transistor (CMOSFET).[4] Moreover, the conventional CMOS-based computation system is confronted with an inherent bottleneck from von Neumann architecture whose paradigm of separating memory and processor enormously consumes the energy and time in data transfer between the memory and processing unit, which is also known as the “von Neumann bottleneck.”[2,5] Therefore, exploiting a novel device to construct the new computation system which may solve the von Neumann bottleneck problem is critically important to maintain the IT innovation.

Memristor, the fourth basic circuit element predicted by Chua and experimentally demonstrated using a TiO₂ resistive switching device by Chua and coworkers and Strukov et al., is a promising candidate to construct the expectable in-memory computing system, owing to its characteristic of resistance-type information carrier.[6–8] In memristor-based in-memory computing systems, the computation unit and memory are merged in the same block so that the shorter data movement shows a

Dr. N. Xu, Prof. L. Fang
Institute for Quantum Information & State Key Laboratory of High Performance Computing, and Microelectronics and Microprocessor Institute
College of Computer
National University of Defense Technology
Changsha, Hunan 410073, China

E-mail: km.kim@kaist.ac.kr

© 2019 The Authors. Published by WILEY-VCH Verlag GmbH & Co. KGaA, Weinheim. This is an open access article under the terms of the Creative Commons Attribution License, which permits use, distribution and reproduction in any medium, provided the original work is properly cited.

DOI: 10.1002/aisy.201900082

Dr. N. Xu, T. G. Park, Dr. H. J. Kim, X. Shao, Dr. K. J. Yoon, Dr. T. H. Park, Prof. C. S. Hwang
Department of Materials Science and Engineering and Inter-University Semiconductor Research Center
Seoul National University
Seoul 151-744, Republic of Korea
E-mail: cheolsh@snu.ac.kr

Prof. K. M. Kim
Department of Materials Science and Engineering
Korea Advanced Institute of Science and Technology
Daejeon 34141, Republic of Korea
E-mail: km.kim@kaist.ac.kr
possible improvement in the computation efficiency.\cite{9,10} The activities for applying the memristor to in-memory computation system can be divided into two mainstreams. One of them can be regarded as analog-type in-memory computing in which the vector–matrix multiplication is used as the primitive to perform the computation.\cite{11,12} The other corresponds to logic in memory (LIM), in which the computation primitives are the basic logic operations (or gatings).\cite{13–20} The former seems to attract more extensive attention because performing the vector–matrix multiplication by physical means (crossbar array [CBA]) based on Kirchhoff’s law is a great replacement for digital operations using the conventional CMOS circuit for machine learning and deep learning.\cite{21–26} Nonetheless, the LIM approach becomes more appealing because it inherits the extensive methodology of digital logic with fine-grained logic operations.

In general, the core of the memristor-based LIM approaches is a certain conditional resistive switching process controlled by the fundamental Kirchhoff’s current and voltage laws. The logic operation is achieved by mapping the logic variables of input and output to the circuit variables of the resistance, voltage, or current of the memristor. According to the symmetry of the logic variables, the current memristor-based LIM strategies can be subdivided into two categories of “memristor sequential logic” and “stateful logic.”\cite{19} The logic inputs of voltage and logic outputs of resistance (or current) are the characteristics of the memristor sequential logic. In this case, one of the research goals is achieving different (as many as possible) logic operations in only one memristor, utilizing the underlying mathematical relationships.\cite{17–19,27–29} However, such asymmetrical logic variables, i.e., inputs are voltage and outputs are resistance, incur a cumbersome problem in logic cascading. The resistance-type output must be transferred to the voltage-type input for the next operation.\cite{10} Fortunately, such a drawback barely exists in stateful logic in which the symmetric resistance-type logic variables, i.e., both inputs and outputs are resistances, are used.\cite{13,30–32} This work belongs to the stateful logic with the flexible cascading property. Cascading in a logic circuit is as important as the logic gate itself in any type of logic operation, as almost all the logic operations must be combined into a complicated configuration both in spatial and in time-wise directions. Efficient and flexible cascading is indispensable for an efficient LIM.

The basic element of stateful logic is the logic primitive circuit which could achieve one or several logic operations constituting a complete set of logic.\cite{13} The more complicated functions can be achieved by cascading these logic operations through (time-spatial) reconfiguration of the logic primitive circuit, as discussed in greater detail later. The logic primitive circuit determines the characteristic of the corresponding stateful logic system and can be used to classify the different stateful logic families. So far, several stateful logic primitive circuits have been proposed to build different stateful logic families. Figure 1a shows the several stateful logic primitive circuits. The material implication (IMP) gate constructed by a pair of parallel memristors with a series resistor ($R_s$) is the first proposed stateful logic primitive circuit which represents a class of IMP-based stateful logic works.\cite{13,14–16} However, the two memristor-based IMP logic primitive circuits had an inherent problem of losing input during the IMP operation because one of the logic input memristors is reused by the logic output, as shown in Figure 1a. Therefore, the cascading process should be carefully designed to avoid such an undesirable effect.\cite{23–27} A common method to avoid such a problem is to use the high-level NAND operation as the primitive for logic cascading,\cite{38,39} where the output can be placed into the third memristor and will not affect the logic inputs. However, this method requires additional operation steps to store (transfer) the output of the IMP gate to another memristor. In contrast, the other two stateful logic primitives (shown in Figure 1a) do not have this problem. Both the memristor-aided logic (MAGIC)\cite{14} and the three memristors stateful logic (TMSL)\cite{31} have three memristors in their logic primitives which can represent the logic inputs and outputs, separately.

Combining the CBA, these logic primitive circuits can be easily configured and cascaded by applying the corresponding control signals, as shown in Figure 1b. Each of them can form a complete logic computation system because the basic logic operations from their logic primitive circuits are all the complete sets of logic. However, it is difficult (maybe impossible) to select the best primitive circuit because the universal evaluation criteria for the logic circuit synthesis\cite{40,41} and cascading methods are yet to be determined at present. In this regard, exploiting the new logic primitive circuits to build the new stateful logic families is timely and important. While some of the previously suggested logic primitives have proven their potential to build a complete stateful logic family, the logic primitive circuits that could be favorably compatible with the 3D CBA, which may greatly facilitate the computation efficiency, are still missing. As shown in Figure 1c, all the presently available logic primitive circuits are related to the parallel connection of two memristors, which is suitable for configuring the devices locating on the same plane of the 3D CBA. However, they can hardly be configured between the devices locating on the different layers or planes. The two antiparallel bipolar memristors (APBM), suggested in this work, can be arranged to any locations within the bulk of the 3D CBA by flexibly combining the different memristors to configure the logic primitive circuit.

The logic primitive circuit can provide six basic operations, implying four two-input logic functions by applying different voltages on them. After placing the circuit in an array, an appropriate combination of time-wise and space-wise logic cascading was achieved. The suggested stateful logic method is highly efficient in terms of its high speed, chip area, and energy savings. Due to the inherent nonvolatile performance of the involved memristors, the data are retained within the circuit even when the power is turned off. Eliminating the burdens of data storage and reloading could be a possible solution to the von Neumann bottleneck problem. Therefore, the suggested APBM can be another breakthrough in the stateful logic circuit.

2. Logic Primitive Circuit and Basic Logic Operations

The schematic configuration of the suggested logic primitive circuit is shown in Figure 2a. Two bipolar memristors ($M_1$ and $M_2$) are antiparallel connected and are then connected to the ground (GND) through an $R_s$. Similar to the other stateful logic primitive circuits, the logic inputs are the two resistances of the two antiparallel-connected memristors, and the logic output
is one of the resistances of them after a conditional switching which is controlled by the applied voltage \( V_A \) and \( R_S \). The detailed mapping processes are as follows.

For the sake of description, the direction from the voltage source to GND is defined as a positive direction. It is assumed that the SET switching (switching from the high-resistance state [HRS] to the low-resistance state [LRS]) of \( M_1 \) occurs in the positive voltage direction, whereas the SET switching of \( M_2 \) occurs in the negative voltage direction. For the sake of simplicity, the absolute values of the SET voltage \( V_{SET} \) and RESET voltage \( V_{RESET} \) are first assumed to be identical to the threshold voltage \( V_{THS} \) to introduce the logic design concept. When the voltage across the subcircuit of \( M_1/M_2 \) \( (V_M) \) is higher than \( V_{THH} \), \( M_1 \) (M2) will be switched from HRS (LRS) to LRS (HRS), whereas \( M_1 \) (M2) will switch from LRS (HRS) to HRS (LRS) when \( V_M \) is lower than \( -V_{THH} \).

The logic states 0 and 1 correspond to the HRS and LRS of the memristors, respectively, and four different logic states (“00,” “01,” “10,” and “11”) can be defined in the subcircuit of \( M_1/M_2 \) depending on the status of \( M_1 \) and \( M_2 \). The state of the subcircuit is switched among the four states by applying the \( V_A \) pulses shown in Figure 2b. For example, when the state of \( M_1/M_2 \) is “00” (both \( M_1 \) and \( M_2 \) are in HRS), the transition “00” \( \rightarrow \) “10” will occur when \( V_M \) is larger than \( V_{THH} \); whereas the transition “00” \( \rightarrow \) “01” will occur in the condition where \( V_M \) is lower than \( -V_{THH} \). Then, six transitions (“00” \( \rightarrow \) “10,” “01” \( \rightarrow \) “10,” “11” \( \rightarrow \) “10,” “00” \( \rightarrow \) “01,” “10” \( \rightarrow \) “01,” and “11” \( \rightarrow \) “01”) are possible when \( V_M \) meets the corresponding conditional values.

When the entire logic primitive circuit is considered, the six transitions are controlled by \( V_A \) and \( R_S \) based on the relationship of \( V_M = \frac{R_A}{R_S+R_A} \times V_A \) (\( R_S \) is the total resistance value of subcircuit \( M_1/M_2 \) which is deduced from Kirchhoff’s laws. Figure 2c shows the variations in the required \( V_A \) for each transition (in Figure 2b) as a function of \( R_S \), where the red, blue, and green lines in the upper half of the graph correspond to the boundaries of the “00” \( \rightarrow \) “10,” “01” \( \rightarrow \) “10,” and “11” \( \rightarrow \) “10” transitions, respectively. Similarly, the three lines in the lower half of the graph define the boundaries of the other three transitions. The detailed process of boundary calculation is included in Supporting Information (“Online SI-I, Logic Operation Boundary Calculation”). When \( (V_A, R_S) \) is higher (lower) than a specific transition line in the upper (lower) half of the graph, specific transition occurs. For example, when \( (V_A, R_S) \) locates in the region (4), the specific transition combination of “00” \( \rightarrow \) “10” and “01” \( \rightarrow \) “10” will occur. The state of \( M_1/M_2 \) will go into “11” \( \rightarrow \) “10” will occur when \( V_M \) is lower than \( -V_{THH} \); whereas the transition “00” \( \rightarrow \) “01” will occur in the condition where \( V_M \) is lower than \( -V_{THH} \). Then, six transitions (“00” \( \rightarrow \) “10,” “01” \( \rightarrow \) “10,” “11” \( \rightarrow \) “10,” “00” \( \rightarrow \) “01,” “10” \( \rightarrow \) “01,” and “11” \( \rightarrow \) “01”) are possible when \( V_M \) meets the corresponding conditional values.

When the entire logic primitive circuit is considered, the six transitions are controlled by \( V_A \) and \( R_S \) based on the relationship of \( V_M = \frac{R_A}{R_S+R_A} \times V_A \) (\( R_S \) is the total resistance value of subcircuit \( M_1/M_2 \) which is deduced from Kirchhoff’s laws. Figure 2c shows the variations in the required \( V_A \) for each transition (in Figure 2b) as a function of \( R_S \), where the red, blue, and green lines in the upper half of the graph correspond to the boundaries of the “00” \( \rightarrow \) “10,” “01” \( \rightarrow \) “10,” and “11” \( \rightarrow \) “10” transitions, respectively. Similarly, the three lines in the lower half of the graph define the boundaries of the other three transitions. The detailed process of boundary calculation is included in Supporting Information (“Online SI-I, Logic Operation Boundary Calculation”). When \( (V_A, R_S) \) is higher (lower) than a specific transition line in the upper (lower) half of the graph, specific transition occurs. For example, when \( (V_A, R_S) \) locates in the region (4), the specific transition combination of “00” \( \rightarrow \) “10” and “01” \( \rightarrow \) “10” will occur. The state of \( M_1/M_2 \) will go into

![Figure 1](image-url)
if its original state is "00" or "01"; otherwise, the state of \( M_1/M_2 \) will be unchanged.

The six transition lines in total can generate six available transition regions in the \( V_A \) versus \( R_S \) graph, as shown in Figure 2c, corresponding to six different transition combinations. Particularly, when the value of \( R_S \) is fixed, the different transition combinations will be directly determined by six different values of \( V_{A_m} \), as shown in Figure 2c. These transition combinations can be interpreted with six different operations implying four logic functions between the original and final states of \( M_1 \) and \( M_2 \), as shown in Figure 2d. The symbols \( p \) and \( p' \) are used to represent an initial and a final state of \( M_1 \), respectively, and q

Figure 2. Logic primitive circuit and basic logic operations. a) Schematic of the APBM circuit and ideal I–V curves of the two bipolar memristors within this circuit. b) State transfer diagram of the APBM circuit. The two bipolar memristors in the antiparallel circuit are shown as a state machine with two bits. Thus, the four states (00), (01), (10), and (11) can be obtained, and state transition may occur when the potential difference \( V_{M_1} \) exceeds a certain threshold value. c) Graph of the logic operation boundaries. Each line corresponds to a transition process in (b). Each section corresponds to a different unit logic operation. Six basic operations can be achieved with six different voltages and a fixed series resistance \( R_S \). d) The truth table of the six basic operations implying four logic functions of IMP, AND, TRUE, and FALSE. They correspond to the six sections in (c). e) HSPICE simulation results of the six basic operations. The two voltage pulse signals with black and red colors represent the real-time signals of applying input voltage pulse \( V_{A_m} \) and voltage difference \( V_{M_1} \), respectively. The real-time resistance values of the orange and green colors are the resistance values of memristors \( M_1 \) and \( M_2 \) (read at 0.1 V), along with the change in the application of voltage \( V_{A_m} \). Six basic operations are triggered by six input voltage pulse signals and a fixed resistance of 4 kΩ. The six voltage pulse signals are \(-2.2 \, V/10 \, ns \) for IMP\(_q\), \(2.2 \, V/10 \, ns \) for IMP\(_p\), \(4 \, V/10 \, ns \) for AND\(_p\), \(-4 \, V/10 \, ns \) for AND\(_q\), \(4 \, V/10 \, ns \) for AND\(_q\), \(-7 \, V/10 \, ns \) for FT\(_p\), and \(7 \, V/10 \, ns \) for TF\(_p\).
and \( q' \) have the same meaning for \( M_2 \). Accordingly, the logic functions can be obtained with \( (p, q) \) being the logic inputs and \( (p', q') \) being the logic outputs. For example, when \( V_\lambda \) is \( V_\lambda \), a pair of logic functions \( (p' = 1, q' = pq) \) is achieved based on the specific transition combination of “00” \( \rightarrow \) “10” and “01” \( \rightarrow \) “10.” Therefore, this operation is named “AND-\( q \).” Figure 2d shows all six basic operations. Among them, “\( FT_{-p \_q} \)” and “\( TF_{-p \_q} \)” provide the logic functions of “FALSE” and “TRUE” which can be used to initialize the device [write “0” ("1") or “0” ("1") to \( M_1 \) and \( M_2 \), respectively]; the other four operations provide the logic functions of “IMP” and “AND” which can be used as the logic operations. It will be demonstrated later that the six basic operations (implying four logic functions) are sufficient to construct a complete Boolean logic.

Alternatively, the six basic operations can be achieved by varying the \( R_3 \) into three values and the \( V_\alpha \) into two values. The variable \( R_3 \) can be achieved from subthreshold metal-oxide-semiconductor-field-effect transistor (MOSFET) or discrete \( R_3 \) connections, as discussed in Supporting Information (“Online SI-II, Variable \( R_3 \)”). As both approaches (variable \( V_\lambda \) and variable \( R_3 \)) produce identical principles of the logic operation, the variable \( V_\lambda \) case is mainly focused on in this article. The detailed circuit configuration including driving peripheral circuits, however, may be different for the aforementioned two cases.

In Figure 2, the electrical behaviors of the symmetric bipolar resistance switching (BRS), which has the same absolute values of \( V_{\text{SET}} \) and \( V_{\text{RESET}} \), were assumed for the sake of simplicity although it is difficult to achieve in a real memristive system.

Moreover, the two resistances \( R_{\text{HRS}} \) or \( R_{\text{LRS}} \) of the real memristors even from the same fabrication process can hardly be identical,

suggesting that the logic family proposed in Figure 2 may not be experimentally realized. It can be demonstrated, however, that the six basic operations can still be feasibly obtained even for the case of \([V_{\text{SET}}] > [V_{\text{RESET}}]\) and \([V_{\text{SET}}] < \) \([V_{\text{RESET}}] \) and \( R_{\text{HRS}} \neq R_{\text{LRS}} \) or \( R_{\text{HRS}} = R_{\text{LRS}} \) whose details are included in the Supporting Information (“Online SI-III, Discussion on the Resistance Switching Parameters of the Memristor”).

The functional correctness of the suggested APBM logic primitive circuit was verified by the HSPICE simulations with a compact Verilog-A-based memristor model, which relies on the dynamics of 1D conductive filament growth/dissolution in the oxide layer.

The model reflects three nonideal effects from the real device: asymmetric switching voltages, gradual reset process, and nonlinear conductance. Figure 2e shows the simulation results of achieving the six basic operations of four logic functions in the suggested APBM circuit. In the simulation process, an \( R_3 \) of 4 kΩ is chosen, and the voltage pulses with −2.2 V/10 ns, −2.2 V/10 ns, −4 V/10 ns, 4 V/10 ns, −7 V/10 ns, and 7 V/10 ns, which correspond to the transition voltages from V1 to V6, are used to trigger the operations IMP_\( q \), IMP_p, AND_p, AND_q, FT_\( p \_q \), and TF_\( p \_q \), respectively. The detailed procedure to determine the \( V_\alpha \) and \( R_3 \) values from the assumed device behavior is described in the Supporting Information (“Online SI-IV, Derivation of Logic Operation Boundary Conditions Based on the Realistic 1D Conductive Filament ReRAM Model”).

To confirm further the feasibility of the proposed logic primitive circuit, the actually fabricated Au/Pt/TiO_2/Ti/Pt memristors were used to construct an APBM circuit to verify the derived state transition relations. Although the test is performed on the APBM circuit constructed by the discrete devices rather than the integrated 3D CBA, they are sufficient for a proof-of-concept verification in this work. The detailed device characteristics and the test methods are included in the Supporting Information (“Online SI-V, Experimental Verification of the APBM Circuit Using the Fabricated Au/Pt/TiO_2/Ti/Pt Memristors”). Figure 3 shows the experimental results of the eight state transitions of the Au-/Pt/TiO_2/Ti/Pt-based APBM circuit. Demonstrating the eight state transitions using the device is successful as it is proposed. Based on these transition relations, the operations implying logic functions can be achieved by applying the corresponding \( V_\lambda \) and adopting the \( R_3 \) which are derived by the procedure explained in SI-III, Supporting Information.

3. Logic Cascading and Complete Boolean Logic

For the implementation of the arbitrary logic and the computation functions, the suitable hardware platforms of memristors are first required to configure the logic primitive circuit. As mentioned earlier, the 3D stacked CBA is the most convenient platform for configuring APBM circuit, so it is mostly concentrated in this study. The memristors located in the two adjacent layers and that share a common bit line (BL) naturally comprise the antiparallel configuration in the 3D structure. An APBM circuit can be configured by applying voltage to two word lines (WLs) from the adjacent layers in between with a GND-applied common BL, as shown in Figure 4a. The two adjacent layers can be defined as P region and Q region, respectively, based on the positive and negative connections of cells. Especially, when the group of common BLs is connected to GND, the memristors selected by the two adjacent WLs can be configured as a group of APBM circuits which supports the parallel operations, as shown in Figure 4b. Although the detailed discussion of improving the parallelism is out of the scope of this work, it should be noted that the parallel operation is very important for stateful logic computation since the nature of the stateful logic is the time-dimensional cascading. (It will be discussed in detail later.) In-depth discussion about this issue will be dealt with in the following works. With conventional 3D CBA, the APBM circuit can only be configured between the memristors from the two adjacent layers. In fact, another configuration for the APBM circuit could be achieved between the memristors in the same layer of the CBA as well, if the cells in the same layer are placed specially based on the special fabrication or operation method (“Online SI-VI, Two types of planar CBAs to configure the APBM circuit,” Supporting Information).

In this article, the adverse effects from the well-known sneak current and wire resistance issues, which may have slightly different aspects from the standard high-density ReRAM, are not dealt with in detail. Instead, suppression of the sneak current by the adoption of an appropriate selector and low-enough wire resistance by adopting sufficiently thick wires are basically asserted. However, in this new scenario, selector and a
memristor should be considered as a single aggregate to extract the switching voltages and resistances to determine the logic trigger conditions.

All the 16 basic two-input logic operations (or gates) can be achieved by any of the three CBAs, i.e., the entire stateful logic family can be built based on the APBM logic primitive circuit. In the APBM stateful logic family, NAND operation is used as the primitive operation for cascading to avoid the loss of input problem. The following parts introduce the methodologies of implementing the NAND operation between the two data in two arbitrary positions of the platforms and performing the complicated functions using the NAND operation.

The input and output data in the suggested platforms of Figure 4 and Figure S9, Supporting Information, could be one of the four patterns: (i) the two input memristors are both in the P region; (ii) the two input memristors are both in the Q region; (iii) the two input memristors are in different regions, and the output is saved in the P region; and (iv) the two input memristors are in different regions, and the output is saved in the Q region. NAND operation can be achieved in any of the four cases.

Figure 5a shows case (i). The input values, a and b, are stored in the memristors A and B in the P region, and the output value, s', is achieved by the three steps saved in the memristor S in the Q region. The NAND function \( s = \overline{a \cdot b} \) is implemented by cascading two IMP functions which are triggered by reasonably configuring two of the three memristors into the APBM circuit. Figure 5b shows the HSPICE simulation results of this NAND operation (NAND_I). A similar method can be used to achieve the NAND operation with the data pattern of case (ii) (NAND_II). (See, Supporting Information, "Online SI-VII-i, NAND_II: Two input memristors are both in the Q region.") A lower number of operation steps than the just-described NAND_I and NAND_II can be used when one of the two inputs is constantly 1, which implies the NOT operation. A detailed discussion (NAND_I \((A, 1)\) or NAND_II \((B, 1)\)) is included in Supporting Information ("Online SI-VII-ii, NOT operation").

To implement the NAND operation with the data patterns of cases (iii) and (iv), the two input data must be moved to the same region first, which can be performed via "move operation," as shown in Figure 6. It is achieved by executing an AND function \((\text{datum} \cdot \text{datum})\) through the "AND_p" or "AND_q" operation. Figure 6a shows the procedure of moving datum from the Q to the P region. The original datum is stored in memristor B in the Q region and can be moved to memristor S in the P region within two steps of achieving the initialization process and
AND operation by performing the “TF_pq” and the “AND_p” operation through sequentially configuring the memristors into the APBM circuit. It should be noted that this operation could not keep the datum in the original memristor (after the move operation, the datum in B will be changed as “1”), which is the reason why it is called “move” not “copy.” The HSPICE
simulation result of this “move operation” (MOVE_I) is shown in Figure 6b. The process of moving datum from the P to the Q region (MOVE_II) using different basic logic operations (“FT_pq” and “AND_q”) is included in the Supporting Information (“Online SI-VII-iii, MOVE_II: Move datum from P to Q region”). Combining one of the two move operations (MOVE_I or MOVE_II) and one of the two NAND operations (NAND_I or NAND_II), the NAND function can be implemented in the data patterns of cases (iii) and (iv) because both cases (iii) and (iv) can be converted to (i) and (ii) based on the move operations, as shown in Figure 7.

XOR is a common logic operation in the modern digital circuit system. Here, the implementation of a binary XOR operation is an example to demonstrate the probability of achieving any binary Boolean logic function by cascading the NAND operations. Figure 8 shows a possible method of achieving the binary XOR operation (XOR_I) by configuring the APBM circuit among the five positive and four negative memristors allocated on one of the suggested hardware platforms. The binary “XOR” function is implemented by cascading binary NAND functions based on the expression \( A \oplus B = \overline{A \overline{B} A \overline{B}} \). The output is stored in the memristor P5 after ten steps. Figure 8b shows the corresponding voltage schemes for the involved steps and data manipulation results for achieving XOR_I. Here, the initialization steps of these NAND operations are not considered for the total number of steps based on the assumption that the large enough CBA is used. In this context, all the initialization processes can be achieved before the execution of the computing process: setting the necessary number of “0” and “1” memristors in the P and Q regions by triggering the two initialization operations (“FT_pq” and “TF_pq”) or the simple SET or RESET operation for these devices, respectively. However, it should be noted that the size of the actual array is limited, which makes the cells have to be reused in the cascading (computation) process. In this case, the initialization processes cannot be performed before the computing process and have to be carefully considered in steps when the cascading and reusing is needed.

In this method, the two logic inputs and all the intermediate results of the component NAND operations are saved in the target memristors, which can be used as the logic inputs for the subsequent logic processes. For example, in Figure 8b, the two logic inputs \( A \) and \( B \) and even the intermediate results \( \overline{A}, \overline{B}, \overline{A \overline{B}}, \overline{A \overline{B} A \overline{B}} \) are all saved in different memristors. They can be used again as the logic inputs for the subsequent logic operations, which may reduce the total number of logic computing steps and may realize energy-efficient computing at the system level. However, if the XOR operation is considered solely, it will not be necessary to save all the intermediate results, which can allow an even more efficient XOR operation of using only six operation steps and five memristors (three positive memristors and two negative memristors), as shown in Figure 9.

Although the XOR function is implemented by sequentially combining the NAND functions based on the disassembled XOR expression \( A \oplus B = \overline{A \overline{B} A \overline{B}} \), several intermediate sub-NAND functions can be implemented through the use of the simple operations (e.g., “AND_p,” “AND_q,” “IMP_q,” and “IMP_p”) by utilizing the intermediate results rather than performing a standard NAND operation. For example, when \( \overline{A} \) is generated in Q1 (step 1), \( \overline{A \overline{B}} \) is not obtained from inputs \( A \) and \( B \) through NAND_I, but it can be directly obtained from the logic operation of “IMP_q” (step 4) and can replace \( \overline{A} \) in Q1. While this is
simpler than the previous case, it also means that $\overline{A}$ is lost after this process. $\overline{A}$, however, is a necessary input for achieving $\overline{A} \overline{B}$ through NAND_II. Therefore, $\overline{A} \overline{B}$ should be performed before $\overline{A}$ is lost, which can be achieved by executing step 3 prior to step 4. This can be understood from the overlapping of different NAND operations, as shown in Figure 9b. In general, a certain binary logic operation can be shown as the sequential combinations of several sub-binary NAND operations. Thus, during the process of performing this binary logic operation, the outputs of a certain sub-operation (intermediate steps, namely "sub-NAND operations") may be the inputs of the subsequent other sub-NAND functions. Then, those subsequent sub-NAND operations can start from the intermediate steps rather than the initialized steps. This method can reduce the total number of operational steps and the necessary number of memristors. This benefit can be attributed to the specific cascading method in APBM stateful logic family, which will be discussed in Section 4 in detail.

The similar cascading method can be extended to execute any complicated multi-input (more than two inputs) logic computing task based on the principle that an arbitrarily complicated combined logic operation can be disassembled into combinations of several binary NAND functions. The one-bit full adder is demonstrated as an example of this multi-input compound logic operation, as shown in the Supporting Information (“Online SI-VIII, Scheme of One-Bit Full Adder”).

4. Discussion of Three Logic Cascading Methods

Finally, the suggested stateful APBM logic family is compared with the current CMOS combinational circuit-based logic family, in which the logic functions are achieved by combining the logic primitive circuit in the space dimension, and the stateful dual-bit memristor (SDBM) family, where the logic functions are achieved by cascading the logic operations in the same logic primitive circuit in the time dimension, which was recently...
Figure 7. Continued.

Figure 8. Schematic of achieving the XOR operation through the APBM circuit (XOR_1). a) Schematic of a potential circuit configuration to achieve the operation XOR_1. Five memristors from the P region and four memristors from the Q region are needed. b) Computation procedures, scheme of voltage signals, and results of data manipulation of XOR_1. Please refer to the video of “XOR_1” in Supporting Information to see the detailed cascading process.
suggested by some of the authors.\cite{15} Stateful logic is an inherently time-sequential logic. If the cascading of the logic functions is flexible only along the time dimension, the calculation efficiency might not be very competitive, and the driving circuitry must be quite costly despite its high spatial efficiency. Therefore, it can be regarded as the time-wise logic circuit. The CMOS combinational logic circuit, in contrast, exerts opposite features; it has very high time-wise efficiency but the space-wise efficiency must be far from the optimal. So, it can be regarded as the space-wise logic circuit. The cascading method of the suggested APBM family is in between the two extremes, so it may be regarded as "space–time-wise."

Figure 10 shows three kinds of schemes for achieving a simple logic function of "XOR" and one-bit full adder using the CMOS combinational logic circuit (mostly space wise), the SDBM logic circuit (mostly time wise), and the APBM logic circuit (space–time wise), respectively. The quantitative comparisons of the three kinds of circuits are shown in Table 1 (for the XOR function) and 2 (for one-bit full adder), respectively. The feature size "F" is used as a unit to evaluate the area cost. The time of one voltage pulse is used as a time unit to evaluate the time cost. For simplicity, the pulse widths in the CMOS combinational logic circuit and two kinds of memristor-based stateful logic circuits are assumed to be identical, and the pulse widths for the writing and reading operations are assumed to be identical as well. It should be noted that the time units for CMOS and stateful logic have quite distinctive implications due to the completely different operational principles. Also, the area cost of the stateful logic should not only be considered in the specific devices for a certain stateful logic function (e.g., full adder) but the whole CBA because of the space reconfiguration of the stateful logic. Therefore, the comparison between the CMOS combinational logic circuit and the stateful logic circuit in this section is just for the purpose to illustrate the characteristics of stateful logic. A more fair comparison for the two kinds of logic schemes needs to consider the overall area, time and the power of the complex computation systems rather than just the logic circuit.\cite{10} The real benefit in performance (speed and energy) of the stateful logic compared with CMOS will not come from a single function operation but from the in-memory computation ability and parallel computation.

The cascading characteristics of the three different logic schemes can be understood from the comparisons in their implementations of XOR. A schematic of the diagram of the CMOS combinational XOR gate is shown in Figure 10a, where "XOR" is achieved by cascading several NOT and NAND circuits (which can be regarded as the logic primitive circuit). All these logic primitive circuits are achieved by predesigned "CMOS gates" arranged in space. In this case, the output is instantaneously achieved (when the gate delay is neglected) at a location spatially distinct from that of the input nodes when the two inputs are assigned to A and B without any additional operational step. Thus, this is space-wise cascading. As shown in Table 1, this property makes the computation extremely fast as the time cost of the CMOS XOR gate is only 1 time unit and causes large-area consumption (e.g., 18 MOSFETs), resulting in an area cost of $108F^2$.

Figure 10b shows an SDBM-based XOR operation, which exhibits a property of time-wise cascading.\cite{15} Only one dual-bit memristor is used to implement a binary "XOR" function, meaning that the area cost is only $4F^2$, which is a very competitive feature of this logic implementation. However, the XOR function is achieved by cascading several basic logic operations in a logic primitive circuit (1M-1R), such as "$IMP_{pq}$," "$AND_q$," "$F_q$," and "$IMP_p$," and four steps are necessary (see Figure 10b-ii).
causing a time cost of 4 time units. The two inputs are induced by the resistance states of the embedded unipolar resistance switching and BRS memristive parts. The data always remain within one of the two memristor components and are changed with time, along with the sequential logic operations. This must be extremely appealing in terms of space, but the speed of this

Figure 10. Schematic of three kinds of logic families with different logic cascading properties. a,d) The schematic diagrams of the conventional CMOS combinational “XOR” gate and CMOS combinational circuit-based one-bit full adder. b,e) The schematics and computation processes of an SDBM-based “XOR” operation and one-bit full adder. c,f) The schematics and computation processes of the APBM-based XOR operations and one-bit full adder.
Comparison of the one-bit full adder procedures of three circuit structures.

| Method               | CMOS       | SDBM       | APBM       |
|----------------------|------------|------------|------------|
| cascading method     | Space wise | Time wise  | Space + time |
| Area cost            | 108F² (18 MOSFETs) | 4F² (one dual-bit memristor) | 20F² (five bipolar memristors) |
| Time cost [time units] | 1         | 4          | 6          |

“F” represents the feature size determined associated with each technology node. For a single memristor in a 2D CBA, the area cost is estimated as 4F²; for a single MOSFET, the area cost is estimated as 6F²; “Time unit”: one voltage pulse is defined as a time unit.

Table 2. Comparison of the one-bit full adder procedures of three circuit structures.

| Method               | CMOS       | SDBM       | APBM       |
|----------------------|------------|------------|------------|
| cascading method     | Space wise | Time wise  | Space + time |
| Area cost            | 324F² (54 MOSFETs) | 12F² (three dual-bit memristors) | 36F² (nine bipolar memristors) |
| Time cost [time units] | 1         | 27 (6 time units for read; 6 time units for write; 15 time units for logic) | 14 |
| Loss of inputs       | Yes        | Yes        | No         |
| Technical maturity   | Easy       | Hard       | Moderate   |

“F” represents the feature size determined associated with each technology node. For a single memristor in a 2D CBA, the area cost is estimated as 4F²; for a single MOSFET, the area cost is estimated as 6F²; “Time unit”: one voltage pulse is defined as a time unit.

circuit may be less effective, especially in the case of performing several complicated computation tasks. The highly combined gate functions require the repeated reuse of the inputs or certain intermediate results, as in the case of the one-bit full adder, which will be discussed later. In contrast, the cascading scheme of the suggested APBM logic family may be regarded as having a property in between the two extremes. Figure 10c is a representative demonstration of an APBM-based XOR, where the appropriate combination of multiple memristors (space-wise cascading) and logic sequences (time-wise cascading) infers an efficient computing performance. In fact, although the circuit structure of the logic primitive is fixed (two antiparallel bipolar memristors connected with an R), the specific bipolar memristors used in each step can be changed. This can be understood as the spatial reconfiguration characteristic of the APBM circuit. The property of the “space + time” configuration of cascading makes the logic data manipulation occur not only along the time dimension but also the spatial dimension, which provides flexibility in designing the circuit based on the reconfigurable logic framework. In Table 1, the area and time costs of the APBM XOR (space–time-wise cascading) are 20F² and 6 time units, respectively, which are in between the values of the CMOS XOR gate (space-wise cascading) and the SDBM XOR operation (time-wise cascading).

By comparing the three kinds of XOR circuits, it is shown that the stateful logic shows the far different cascading method compared with CMOS combinational logic circuit. Such a difference reflects the changes in area and time.

The features of the two kinds of stateful logic families (with two different cascading properties) can be further revealed through a complicated computation task of the one-bit full adder. Figure 10d shows a schematic of achieving a complicated computation task: one-bit full adder by cascading different CMOS gates. Two “XOR” gates, two “AND” gates, and one “OR” gate are arranged in the spatial dimension, and the two outputs can be instantaneously obtained when the three inputs a, b, and c are assigned. Thus, the CMOS adder still takes two extremes: the largest area cost of 324F² and the smallest time cost of 1 time unit. The disadvantage of the SDBM logic family (time-wise cascading), however, can be obviously shown in Figure 10e. Six stages are necessary to achieve a function of one-bit full adder using three dual-bit memristors. The three dual-bit memristors correspond to the area cost of 12F² (as shown in Table 2). For each dual-bit memristor, the unipolar part (p-bit) is used to save the logic computing result, and the bipolar part (q-bit) is used as a buffer to provide a port for logic cascading. Each stage includes several smaller steps. As the cascading of the logic functions of the SDBM logic is flexible only in the time dimension (two inputs can be linked only through a dual-bit memristor, by assigning the two inputs to the two bits, respectively), additional read and write operations are necessary for transferring data into the other dual-bit memristor, to finish the spatial cascading of the logic primitives. Such an operation method largely increases the time cost to 27 time units, as shown in Table 2. In contrast, the logic primitive circuit in the APBM logic family could show space-wise efficiency, whereas the sacrifice in the time-wise efficiency was mitigated. For example, in Figure 10f, nine memristors (36F² area cost) and 14 time units are required to achieve the one-bit full adder function. No additional read and write operations are required for moving the data because the two inputs are automatically loaded when the two memristors with input data are selected to configure the APBM circuit. Such features save the many operation steps for performing a complex computation task that needs to repeatedly reuse the inputs or immediate results. Thus, another conclusion can be obtained; the logic family with the logic primitive circuit of supporting space–time-wise cascading may be an optimized concept for the data manipulation of stateful logic considering both the area and time costs.

Furthermore, more advantages of the suggested APBM logic family can be identified when the two other items of “loss of inputs” and “technical maturity” are considered, as shown in Table 2. Due to the volatility of the CMOS circuit, the input and output data are all lost when the power is switched off. Thus, the data had to be saved in another place (nonvolatile memory) and reloaded again along with the computing process, which increases the overall power consumption. This problem is intrinsically avoided in the stateful logic because the input and
output data are saved in the logic device itself. The consumption in power, area, and time to perform a computation, however, may be different for the different stateful logic families. The SDBM logic family cannot save the inputs because the inputs must be replaced by the output. Thus, additional time and area costs are necessary to compensate for this loss, which may increase power consumption. In contrast, the APBM logic family can avoid the loss of inputs by saving the output in another memristor through the spatial reconfiguration of the logic primitive circuit. Therefore, no additional time or area cost is needed for saving the inputs, which may lead to high energy efficiency. In the aspect of technical maturity, although the two stateful logic systems cannot be competitive with the currently industrialized CMOS technology, the potential of achieving computation in the memory of these two stateful logic circuits shows high prospects in the low-power-consumption application, which may also promote the industrialization of the memristor. In addition, it will be easier to fabricate APBM logic circuits than SDBM logic circuits because the well-known high-performance memristors, such as the HfO$_2$- and TaO$_x$-based bipolar resistive switching materials, can be used in the APBM circuit, whereas the dual-bit technology material is limited to TiO$_2$, which is known as a comparably low-performance material.

5. Conclusions

In conclusion, a new stateful logic primitive circuit consisting of two antiparallel bipolar memristors is proposed to build a 3D crossbar-compatible stateful logic family. Each logic operation generated from the APBM circuit can be operated with a single voltage source and a single value of $R_x$. Six basic logic operations—“IMP$_{pq}$,” “IMP$_{p}$,” “AND$_{p}$,” “AND$_{q}$,” “FT$_{pq}$,” and “TF$_{pq}$”—were generated by applying six different voltage signals to the logic primitive circuit. The binary NAND function was implemented for all the four input patterns based on the six aforementioned basic operations, which can then be used as a high-level primitive to achieve more complicated functions, such as XOR and one-bit full adder, demonstrating the logic completeness of the suggested scheme. Compared with the current CMOS combinational logic family, in which the logic primitive circuit only supports the space-wise cascading, and the SDBM logic family, in which the logic primitive circuit only supports the time-wise cascading, the present APBM logic family combines the space- and time-wise aspects of general logic cascading. This must provide the field with unprecedented flexibility for designing a new nonvolatile logic circuit with extremely low power consumption. In addition, the computation efficiency is expected to improve further as the proposed logic primitive circuit overcomes the limitation of the configuration between the devices from the adjacent layers of the 3D CBA. Furthermore, the suggested logic primitive circuit can be fundamentally implemented by any BRS device, even with asymmetric switching properties with respect to the SET and RESET voltages as well as different resistance values, and by devices with non-ideal characteristics like a gradual reset and nonlinear conductance. In general, the stateful logic circuits require multiple operation steps and smaller footprint compared with the CMOS-based logic circuits and are fundamentally an LIM, so they are suitable for the edge devices where space and energy efficiencies matter more than time efficiency, i.e., high performance. The APBM suggested in this work may offer an efficient compromise between the purely time-wise stateful logic and purely space-wise CMOS logic circuits.

6. Experimental Section

To fabricate the memristor, 50 nm platinum (Pt) as a bottom electrode (BE) was deposited on the Ti/SiO$_2$/Si wafer using electron beam evaporation, where 10 nm titanium (Ti) was used as an adhesion layer. Then, the 35 nm TiO$_2$ thin film was fabricated via radio frequency (RF) magnetron sputtering with 150 W RF power in 13 mTorr reactive working pressure in 20% O$_2$/80% Ar gas ambient at room temperature, using a 3” TiO target. About 100 nm Au and 50 nm Pt were deposited in situ as top electrode (TE) using electron beam evaporation. All the electrodes were patterned by the photolithography method and followed by a lift-off process.

The electrical measurement was carried out in the pulse-type voltage application (with the HP 81110A pulse generator [PG] as the voltage-programmed current source and the Tektronix 684C oscilloscope [OSC] as the current monitor) and in the voltage sweep mode using a semiconductor parameter analyzer (HP 4154B) to conduct the electroforming process. All the bias voltage was applied to the TE, whereas the BE was grounded. The antiparallel connection of two devices was conducted using the switching circuit designed by a printed circuit board (PCB). The mathematical tool “MATLAB” was used to calculate the logic operation conditions. The circuit simulation tool “HSPICE” was used for simulation of the logic operations.

Supporting Information

Supporting Information is available from the Wiley Online Library or from the author.

Acknowledgements

N.X. and T.G.P. contributed equally to this work. This work was supported by Future Semiconductor Device Technology Development Program (20003655) through the Ministry of Trade, Industry, & Energy (MOTIE, Korea) [project name: Development of self rectifying resistance switching materials and devices for selectorless crossbar application]. N.X. thanks China Scholarship Council (CSC, file no. 201503170323) for financial support.

Conflict of Interest

The authors declare no conflict of interest.

Keywords

antiparallel memristors, logic cascading, stateful logics, 3D crossbar arrays

Received: July 23, 2019
Revised: October 22, 2019
Published online: December 9, 2019

[1] A. M. Turing, Proc. London Math. Soc. 1937, 2, 230.
[2] C. Carvalho, in Proc. of IEEE Int. Conf. on Control and Automation, IEEE, Montreal 2002, pp. 27–34.
[3] K. M. Bresniker, S. Singhal, R. S. Williams, Computer 2015, 48, 44.
