Stateful In-Memory Logic System and Its Practical Implementation in a TaO$_x$-Based Bipolar-Type Memristive Crossbar Array

Young Seok Kim, Myeong Won Son, Hanchan Song, Juseong Park, Jangho An, Jae Bum Jeon, Geun Young Kim, Seoil Son, and Kyung Min Kim*

Memristive stateful logic enables energy- and cost-efficient in-memory computing, which is desirable for edge computing in the coming Internet of Things (IoT) era. Researchers have recently developed various stateful logic gates and have shown viable computing applications based on ideal memristive characteristics. However, few studies have demonstrated a system-level in-memory computing operation that can address the practical issues affecting device realization. Herein, a practically viable stateful logic device based on a 1-transistor–1-memristor (1T1M) array structure is proposed, considering the inherently stochastic memristor characteristics. Details on how to select the viable stateful logic gates in a given memristor are shown, and as an example of logic cascading, they are implemented in a device to operate a multibit carry look-ahead adder. Then, an in-memory computing layout that can perform all of the computing functions—data storing, transferring, and executing—inside the memory, addressing data traffic issues, is suggested. Finally, a software/hardware mixed stateful logic emulator that can virtually mimic array-level in-memory computing hardware based on cell-level memristive characteristics is demonstrated.

Memristors are expected to enable a new paradigm for future energy-efficient computing, beyond complementary metal-oxide-semiconductor technology.$^{[1,2]}$ Jeong et al. and Zidan et al. highlighted memristors as promising candidates for three applications: emerging memory, in-memory computing, and neuromorphic computing devices. In 2018, Ielmini and Wong further emphasized the importance of memristors for in-memory computing technology.$^{[3]}$ An in-memory logic device eliminates the data in and out (I/O) operation between logic and memory devices and thus helps solve the data bottleneck problem, while significantly reducing energy consumption. It is also essential to develop a one-chip computer, which is preferred for the Internet of Things (IoT) applications and edge computing, where low cost and low-energy consumption are considered more important than high performance.$^{[4,5]}$

The nonvolatile latching characteristic of the memristor enables Boolean logic operations inside the memory. Because Borghetti et al. first proposed the material implication (IMP) gate in a crossbar array in 2010,$^{[6]}$ various logic gates$^{[7–24]}$ and their optimized cascading methods for realizing adders$^{[25–30]}$ have been demonstrated. They included two input gates, as well as multiple input gates, to perform more complicated gates efficiently. The most noticeable achievement was reported by Sun et al. in 2018. In their work, the carry and sum calculations of a full adder were executed in just two steps by accessing four and five input cells together. They called this concept of accessing multiple cells sequentially in a discrete time dimension to perform complicated calculations efficiently a “stateful neural network.”

Although these gates and optimized operation algorithms are innovative, the technologies still have issues which limit them from being fully implemented in a device. Here, we consider two critical factors which are mandatory for device implementation. First, the logic operation must be 100% successful, even considering the switching voltage variation. In a memory operation, where all cells are independently accessed, guaranteed switching can be achieved by applying programming (writing and erasing) voltage amplitudes that are higher than the switching (set and reset) threshold.$^{[31,32]}$ The maximum voltage amplitude is not limited because one is not concerned with other issues, such as a breakdown. In the logic operation, however, multiples cells are selected at the same time, and they are dependent on each other. The increase in programming voltage amplitudes on the target cell may lead to unwanted programming of the other cells. This means there is a maximum limit to the logic operation voltage amplitudes, as well as a minimum, and this makes the operating voltage margin much tighter than...
in a memory operation. If the variation in the programming voltage of a certain gate is more severe than the required operating voltage margin, the gate will not be practically available. Nevertheless, there have been few studies that consider the switching voltage variation into the logic operation.

Second, a compact and viable data manipulation method is required. In previous studies of cascading logic gates, it was assumed that the inputs were well aligned to the designed gate operation. However, in a 2D or 3D array, the data may exist at any location on the array. Furthermore, the target address of the output may be occupied by other data. In such cases, the data first have to be relocated before performing the specific logic gate, which is possible via a data copy or transfer manipulation.

In this study, we propose a practically viable in-memory logic device and system. We first collect all possible stateful logic gates and systematically evaluate their practical validity, considering the device variations from experiments. We present a detailed methodology for evaluating the validity of the logic gates. The results showed that only five gates (NOT, NOR, IMP, OR, and BUFFER) were valid in our TaOx-based 1-transistor–1-memristor (1T1M) array device, considering its statistical variation. Then, we show a compact multibit carry look-ahead adder (CLA) operation, utilizing the practically available gates, enhanced by parallel logic operation. As a demonstration, we develop an in-memory computing emulator, where a software-based controller is used to operate the 1T1M array hardware, and successfully demonstrate the CLA operation. Finally, we propose a crossbar array layout composed of memory, bus, and logic sections for organized data manipulation, to implement in-memory computing.

Figure 1a shows a conceptual schematic of the stateful neural network using the McCulloch–Pitts neuron model proposed by Sun et al.19,34 In the schematic, the inputs are conductance \( G_i \), which represents the state of the memristor cells \( i = A, B, \) and \( C \), and a series resistor \( i = R \). Then, the weight values \( W \) go into the machine. In the crossbar array device, the form of the weight value is a voltage, where a current value is \( G_i \) times the applied voltage \( V_i \). Inside the machine, the currents are summed by Kirchhoff’s current law, and the actual voltage potential across each memristor cell \( V_i = V_i - V_{NODE} \) is determined accordingly, where \( V_{NODE} \) is a node potential of the opposite terminal of \( V_i \) which can be expressed by \( V_{NODE} = \sum_i V_i G_i / \sum_i G_i \).

Then, if any \( V_i \) on the memristor exceeds the switching threshold voltage \( \theta \), the corresponding memristor cell is switched, and its conductance is updated as a logical output. Here, the logic gates can be categorized depending on whether the cell is set switched or reset switched, into SET-logic gates and RESET logic gates. Figure 1a shows the initial “0” state (or false) of \( C \), which is updated to the “1” state (or true) by the SET-logic gates.

Figure 1. Logic-gate evaluation procedure considering the variation of memristor. a) A conventional stateful neural network model that does not consider the variation in switching threshold voltage (upper panel) and a proposed model that considers (lower panel). b) Typical switching voltage distribution for bipolar memristors. c) Switching voltage distributions normalized to \( V_{SET,\text{Max}} \) for SET-logic and to \( V_{RES,\text{Min}} \) for RESET-logic. The unit values are highlighted in yellow. d) Definition of parameters given in part (c). e) Voltage conditions need to be considered for possible cases in SET-logic and RESET-logic. f) Schematic of circuit for implementing IMP gate. States P and Q refer the logical values of two memristors. g) Truth tables and required voltage conditions corresponding to each case in the truth tables before and after the IMP gate operation.
In the conventional scheme (top panel of Figure 1a), the switching voltage is assumed to be a constant value; therefore, the logical output is deterministic once \( \nu \) is obtained. However, in reality, the switching voltages are inherently stochastic.\(^{[15-18]}\) In this regard, the bottom panel in Figure 1a is more realistic. The \( \theta \) is distributed and the output is a problematic function of the degree of \( \theta \) variation.\(^{[19,39]}\) Then, for guaranteed gate operation, \( \nu \) should be higher than the maximum amplitude of \( \theta \) variation. Therefore, for deterministic gate operation, the acceptable value of \( \theta \) variation should be systemically examined for practical use in the stateful logic gate.

Here, we establish a procedure to evaluate the degree of immunity of the logic gates against \( \theta \) variation. To accomplish that, we first redefine several parameters related to the switching voltages and their distribution. Figure 1b shows the switching voltage distribution of a typical bipolar memristor. It also corresponds to the characteristic of the device used in this study. Here, the set voltage \( (V_{\text{SET}}) \) is the threshold voltage needed to change a high resistance state (HRS) to a low resistance state (LRS) where the HRS and LRS represent logical “0” and “1” values, respectively. The set switching is drastic, so an applied voltage exceeding \( V_{\text{SET}} \) can directly change the HRS to the LRS. The reset voltage \( (V_{\text{RES}}) \) is the threshold voltage needed to change the LRS to the HRS. Unlike the set switching, the reset switching of our device is gradual, so a complete reset switching is possible by applying a clear voltage \( (V_{\text{CLR}}) \), which is lower than the \( V_{\text{RES}} \). If the reset switching is abrupt in other memristors, \( V_{\text{CLR}} \) is simply equal to \( V_{\text{RES}} \). In different types of memristors, other cases are possible; either the set switching can be gradual or the reset switching can be drastic. In those cases, the distribution model can be applied with a simple modification. The maximum and minimum values and their distribution are shown for each switching voltage.

For further simplification and generalization, the switching voltages are normalized to the \( V_{\text{SET,max}} \) and \( V_{\text{RES,min}} \) for the SET- and RESET-logic gates evaluation, respectively, and only the important parameters are shown in Figure 1d. For example, in the SET-logic (the upper panel in Figure 1c), \( \Delta_{\text{SET}} \) is defined by \( \left[ (V_{\text{SET,max}} - V_{\text{SET,min}})/V_{\text{SET,max}} \right] \) and \( \rho_{\text{SET}} \) is by \( (V_{\text{RES,max}}/V_{\text{SET,max}}) \). Similarly, in the RESET-logic (the lower panel in Figure 1c), \( \Delta_{\text{RES}} \) and \( \rho_{\text{RES}} \) are defined by \( \left[ (V_{\text{RES,min}} - V_{\text{RES,max}})/V_{\text{RES,min}} \right] \) and \( (V_{\text{CLR,min}}/V_{\text{RES,min}}) \), respectively. In addition, \( \chi_{\text{CLR}} \) is defined by \( (V_{\text{CLR.min}}/V_{\text{RES.min}}) \). These parameters are shown in Figure 1d.

For a successful logic operation, the following conditions should be satisfied. In the SET-logic, 1) for the cells maintaining the 0 state, the normalized potential on them \( (\nu) \) should be lower than \( (1 - \Delta_{\text{SET}}) \) (i.e., \( \nuM < 1 - \Delta_{\text{SET}} \) for \( 0 \rightarrow 0 \)); 2) for the cells maintaining the 1 state, the \( \nu \) should be higher than \( (\rho_{\text{RES}} \Delta_{\text{SET}}) \) (i.e., \( \nuM > \rho_{\text{RES}} \Delta_{\text{SET}} \) for \( 1 \rightarrow 1 \)); and 3) for the output cell changing from 0 to 1, \( \nuM \) should be higher than \( 1 \) (i.e., \( \nuM > 1 \) for \( 0 \rightarrow 1 \)). The case for the state changing from 1 to 0 can be neglected because there is no such case in the SET-logic. In the RESET-logic, 1) for the cells maintaining the 0 state, \( \nuM \) should be lower than \( (\rho_{\text{SET}} \Delta_{\text{SET}}) \) (i.e., \( \nuM < \rho_{\text{SET}} \Delta_{\text{SET}} \) for \( 0 \rightarrow 0 \)); 2) for the cells maintaining the 1 state, \( \nuM \) should be higher than \( -1 + \Delta_{\text{RES}} \) (i.e., \( \nuM > -1 + \Delta_{\text{RES}} \) for \( 1 \rightarrow 1 \)); and 3) for the output cell changing from 1 to 0, \( \nuM \) should be lower than \( -1 \) before switching, and it should be lower than \( -\chi_{\text{CLR}} \) after switching (i.e., \( \nuM < -1 \) before switching and \( \nuM < -\chi_{\text{CLR}} \) after switching for \( 1 \rightarrow 0 \)). The case for the state changing from 0 to 1 can be neglected. These conditions are shown in Figure 1e.

Then, the voltage condition rules in Figure 1e can be applied to each gate. Figure 1f,g shows an example of the application of the logic rules in Figure 1e to the IMP gate by Borghetti et al.\(^{[6]}\) Figure 1f shows the logic unit with two memristors. Here, \( \nuP \) and \( \nuQ \) are the actual voltage potentials on two memristors, \( M_P \) and \( M_Q \). Figure 1g shows 16 voltage conditions for every possible input and output case during the IMP gate operation. For successful IMP gate operation, all 16 conditions should be satisfied. If the duplicated inequalities are neglected, only eight conditions (yellow-colored cells) have to be considered.

In the inequalities, there are two variables \( (\Delta_{\text{SET}} \text{ and } \rho_{\text{RES}}) \) in the SET-logic and three \( (\Delta_{\text{RES}}, \rho_{\text{SET}}, \text{ and } \chi_{\text{CLR}}) \) in the RESET-logic. To solve the inequalities, the maximum value of \( \Delta_{\text{SET}} \) (or \( \Delta_{\text{RES}} \)) is calculated as a function of \( \rho_{\text{RES}} \) (or \( \rho_{\text{SET}} \) and \( \chi_{\text{CLR}} \)). Once the maximum value of \( \Delta_{\text{SET}} \) (or \( \Delta_{\text{RES}} \)) is obtained, the required \( \nuP \) and \( \nuQ \) can be obtained accordingly. (A more detailed calculation process is shown in Supplementary Note 1 and Figure S1 and S5, Supporting Information.)

Here, the maximum value of \( \Delta_{\text{SET}} \) (or \( \Delta_{\text{RES}} \)) is defined as a “variation tolerance factor” (VTF) which shows how much the logic gate is tolerant against variation in switching voltage. For example, in a specific SET-logic gate, the logic gate is practically viable only if the experimentally obtained \( \Delta_{\text{SET}} \) of a certain memristor is smaller than the VTF of the logic gate. As such, by comparing the VTF value and the experimentally obtained \( \Delta_{\text{SET}} \) for the SET-logic (or \( \Delta_{\text{RES}} \) for the RESET-logic), the viability of the logic gates can be intuitively determined. Taking the IMP gate, for example, the solution indicates that VTF is 0.333, and is independent of \( \rho_{\text{RES}} \). Therefore, the gate is viable if the device has a \( \Delta_{\text{SET}} \) lower than 0.333.

In this way, the VTF values of all logic gates can be obtained. The lists of SET-logic and RESET-logic gates are shown in Figure 2a,e, respectively.\(^{[6,7,17-22]}\) The first digit in the gate name refers to the number of cells required to execute the gate. The “-” at the ending refers to the floating of \( V_R \) during gate operation. At “2IMP” gate, for example, two cells are inputs, and one of them is an output cell, which is set switched only if both cells are in the HRS. At “3NOR” gate, for another example, two cells are inputs, and another cell is assigned to the output, which is set switched only if all input cells are in the HRS.

Truth tables of all logic gates are shown in Figure S6, Supporting Information. The listed logic gates can be executed in a single step in the bipolar memristor array. Figure 2b-d shows the VTF values of the SET-logic gates for \( \rho_{\text{RES}} = 1, 0.5, \) and 0.2, respectively. The insets show I-V curve shapes corresponding to the \( \rho_{\text{RES}} \) values.

When \( \rho_{\text{RES}} = 1 \), all of the VTF values are positive; therefore, all logic gates are possible if no variation in switching voltage is considered (\( \Delta_{\text{SET}} = 0 \)). However, because \( \Delta_{\text{SET}} \) cannot be 0, the possible logic gates are dependent on the \( \Delta_{\text{SET}} \) value: if \( \Delta_{\text{SET}} < 0.111 \), all gates are possible; if \( 0.111 < \Delta_{\text{SET}} < 0.25 \), two gates (5SUM and 3NAND) are impossible; if \( 0.25 < \Delta_{\text{SET}} < 0.333 \), six gates are impossible (5SUM, 3NIMP, 3RIMP, 3NIMP, 3IMP, and 3NAND), and so on. When \( \rho_{\text{RES}} = 0.5 \) as in Figure 2c, the VTF values are generally smaller than those in Figure 2b. When \( \rho_{\text{RES}} = 0.2 \) as in
Figure 2d, most of the logic gates except four gates (3NAND, 2NOT, 3NOR, and 2IMP) are impossible because the smaller \( \rho_{RES} \) value may cause unwanted reset switching. The \( \rho_{RES} \) dependence suggests that the memristor with the higher \( \rho_{RES} \) can synthesize more gates at a given \( \Delta_{SET} \).

Similarly, Figure 2f–h shows the VTF values for the RESET-logic gates for \( \rho_{SET} = 2 \) and \( \chi_{CLR} = 1 \), \( \rho_{SET} = 1 \) and \( \chi_{CLR} = 1 \), and \( \rho_{SET} = 2 \) and \( \chi_{CLR} = 2 \), respectively. The insets also show the corresponding \( I-V \) curve shapes. Practically impossible gates are shown in red. Vertical lines represent some \( \Delta_{SET} \) values for SET-logic or \( \Delta_{RES} \) values for RESET-logic. (\( \Delta_{SET} \) or \( \Delta_{RES} \) = 0, 0.111 or 0.1, 0.25, 0.333). In the charts, by comparing the VTF values and the \( \Delta_{SET} \) or \( \Delta_{RES} \) of the device, the practically viable gates can be intuitively determined at a given memristor.

In both configurations, by applying appropriate voltages on the word and bit lines, a specific gate operation is possible without disturbing other cells, when the gate is closed. Also, in the 1T1M structure, the transistor acts as a current limiter (i.e., a series resistor) so that the reliability of the device is ensured. Moreover, the transistor can work as an in-cell series resistor component that can increase the absolute value of \( V_{RES} \), and, thus, it is beneficial for the SET-logic gates but harmful to the RESET-logic gates.

Figure 3d shows our 1T1M device structure with a Ta/TaOx/Pt memristor. The inset shows a top-view SEM image of the device. Figure 3e shows the \( V_{CLR} \), \( V_{RES} \), and \( V_{SET} \) distributions obtained from 50 consecutive cycles where the maximum and minimum values of the \( V_{CLR} \), \( V_{RES} \), and \( V_{SET} \) were \([-2.25, -1.1]\), \([-1.05, -0.9]\), and \([1.45, 1.85]\), respectively. Here, the voltage sweep rate was fixed to 2 V s\(^{-1}\). Once the voltage sweep rate in the voltage sweep mode (or the pulse width in the pulse mode) is changed, the switching voltage amplitude and distribution may be affected, as shown in Figure S7, Supporting Information. In our device, the faster sweep rate
caused the lower switching voltage amplitude and the higher voltage variation.\textsuperscript{45,46} Therefore, the selection of the sweep rate is crucial for the practical stateful logic operation. The abrupt set switching and the gradual reset switching shown in Figure 3e is a representative characteristic of TaO\textsubscript{x}-based devices.\textsuperscript{47,48}

Then, to enhance the statistical reliability of the data from a limited number of datasets, \( V_{\text{RES}} \) and the \( V_{\text{CLR}} \) were fitted with the Weibull distribution,\textsuperscript{49} and \( V_{\text{SET}} \) with the normal distribution. The details of the distribution fitting are shown in Figure S8, Supporting Information. From the distribution fitting, 99.5\% confidence intervals were \([-2.406, -0.871]\) for the \( V_{\text{CLR}} \), \([-1.086, -0.888]\) for the \( V_{\text{RES}} \), and \([1.338, 1.986]\) for the \( V_{\text{SET}} \). These intervals were set to the maximum and minimum switching voltages shown in Figure 1b, which gave \( \Delta_{\text{SET}} \) and \( \Delta_{\text{RES}} \) of 0.326 and 0.182, respectively. Also, the normalized parameters were \( \rho_{\text{RES}} = 0.447 \), \( \rho_{\text{SET}} = 1.232 \), and \( \chi_{\text{CLR}} = 2.216 \). From these parameters, the VTF values of the SET-logic and the RESET-logic can be obtained, which are shown in Figure 3f,g, respectively. The logic gates in blue color are the viable logic gates satisfying VTF > \( \Delta_{\text{SET}} \) or \( \Delta_{\text{RES}} \). A comparison of the experimentally obtained successful gate operation results of 3NOR and 3NAND of specific devices with a higher \( \Delta_{\text{SET}} \) value of 0.383 and 0.340, respectively, and the theoretical success rate. For each input state, the success rate was obtained from 50 trials.

Figure 3. VTF values of 1T1M array device and verification of the gate viability. a) A top-view SEM image of a 1T1M array (scale bar: 20 \( \mu \text{m} \)). b,c) Configurations of a shared-wordline logic unit (SWLU) and a shared-bitline logic unit (SBLU). For the SWLU operation, only one shared gate is open, whereas, for the SBLU, multiple gates are open. d) A schematic device structure of the 1T1M device. Inset shows a top-view layout of a 1T1M cell (scale bar: 10 \( \mu \text{m} \)). e) Switching voltage distributions of a single 1T1M device. The inset table summarizes the parameters of this device. The inset I–V curve shows the original switching curve data from 50 measurements. Each line of the graph stands for a fitting line for each threshold voltage distribution, where the distribution of the \( V_{\text{SET}} \) fits the normal distribution, and \( V_{\text{RES}} \) and \( V_{\text{CLR}} \) fit the Weibull distribution. f,g) The VTF values of SET-logic and RESET-logic of the device obtained from the derived parameters. The logic gates in blue color are the viable logic gates satisfying VTF > \( \Delta_{\text{SET}} \) or \( \Delta_{\text{RES}} \). h) A comparison of the experimentally obtained successful gate operation results of 3NOR and 3NAND of specific devices with a higher \( \Delta_{\text{SET}} \) value of 0.383 and 0.340, respectively, and the theoretical success rate.
The raw data from the experiments are shown in Figure S9 and S10, Supporting Information.

Figure 4a shows the CLA, one of the multibit adder types developed, to provide a faster and more efficient operation than a ripple carry adder (RCA). Unlike the RCA, where the carry and sum bits are calculated sequentially, the CLA carries out the calculation in parallel; the carry-out values of all bits are obtained first, and then, the sum values are calculated later. For the CLA operation, the circuit requires a carry look-ahead generator to calculate the carry-out values at the same time. This is complicated but can improve the calculation speed dramatically. The parallel operating capability of the stateful logic gate makes it capable of implementing the CLA without any additional circuits.

To execute the n-bit CLA operation (where \( n = 4 \)), a 5 \( \times \) 13 crossbar matrix was prepared and the cells were programmed, as shown in Figure 4b, which corresponds to the initialization step. Here, the dashed vertical lines, solid vertical lines, and horizontal lines refer to the gate, word, and bit lines, respectively. In the matrix, four sets (\( n = 1, 2, 3, 4 \)) of addend and augend (\( A_n, B_n \)) are located at a 4 \( \times \) 2 crossbar matrix at the left side of the array. Then, the carry-in input (\( C_m \)) is located at the top of the third column. Other cells in the third column were initialized to the LRS where the carry-out values were to be programmed by the RESET-logic. All other cells from the fourth column to the end were initialized to the HRS. This data arrangement allows the most efficient CLA operation, fulfilled by a parallel gate operation.

Once the cells are ready to process, the first step is to calculate all carry-out values. Figure 4c shows the most optimized carry look-ahead generator designed for the stateful logic operation. It utilizes three practically available gates: the 2NOT and 3NOR gates of SET logic, and the 2BUFFER gate of the RESET logic. In the carry look-ahead generator, the carry-out values (\( C_2 \)) of the inputs (\( A_n, B_n, \) and \( C_n \)) are stored at the (3, \( n + 1 \)) position in the (x, y)-coordinate, where the left-top corner is defined as (1, 1). For that, the first step is to calculate \( C_2 \) and store the datum at (4, 1). This can be expanded to \( A_1 \times B_1 + B_1 \times C_1 + C_1 \times A_1 \) by De Morgan's law. Because the 3NOR gate can store the output of \( A_1 \times B_1 \) to any cell and the summation (+) of the logic can be simply achieved by overwriting the data on the same cell, the \( C_2 \) output can be recorded at the (4, 1) position by executing the 3NOR gate three times: \( C_2 \leftarrow 3\text{NOR}(A_1, B_1),\) \( C_2 \leftarrow 3\text{NOR}(B_1, C_1),\) and \( C_2 \leftarrow 3\text{NOR}(C_1, A_1).\) Detailed explanations of the solving process and overwriting are shown in Figure S11, Supporting Information. Then, \( C_2 \) can be stored at the (4, 2) position by flipping the datum in (4, 1) using the 2NOT gate. Finally, \( C_2 \) can be copied from the (4, 2) position to the (3, 2) position by executing the 2BUFFER gate. Consequently, the \( A_2, B_2, \) and
C₂ values can be lined up in the second row with five steps. In this way, the carry-out value can be consecutively stored in the next column, and 5 \( n \) steps are required to store all of the carry-out values of the \( n \)-bit CLA.

Once the carry-out values are lined up in the third column, the sum values can be calculated in parallel. Figure 4d shows the sum generator. The sum calculation is equal to \((A \oplus B) \oplus C\) and it can be re-expressed to \((A \odot B) \odot C\), where \( A \odot B \) is equal to \((A + A + B) \oplus (B + A + B)\) (\( \oplus \): XOR gate, \( \odot \): XNOR gate). The solving process is also shown in Figure S11, Supporting Information. Consequently, the sum calculation is possible by repeating the 3NOR gate eight times. Also, the 3NOR gate can be performed over multiple rows in parallel because the biasing of every bit line allows every row to work independently. Therefore, the sum values of \( n \) inputs can be calculated together.

In the sum generator, steps 1–4 correspond to the first XNOR operation and steps 5–8 to the second XNOR operation. As a result, the sum results are stored at the last columns in eight steps. Combining the carry-out and sum calculation, \( n \)-bit CLA requires \( 5n + 8 \) steps. Table S1, Supporting Information, shows the number of cells and steps for executing \( n \)-bit full adder compared with previous technologies. Although our work is not the most efficient, it is the only practically feasible one among the technologies.

The aforementioned \( n \)-bit CLA operation assumed that the input data were well-organized, as shown in Figure 4b, before performing the designed sequence. However, because the data are physically scattered over the memory array, they need to be physically relocated to the designed location. In this regard, we suggest a new in-memory computing device layout design. Figure 5a schematically shows the proposed crossbar layout. The array is composed of memory, logic, and bus sections. All of the sections are physically identical, but their roles are distinguishable. In the memory section, the data are stored permanently. In the logic section, the data are not used for permanent data recording but are used temporally as a cache memory only for the logic operation. Also, considering that the stateful logic operation is executed along a 1D direction, a bus section is assigned to buffer the data during data manipulation. Then, the original data at any location in the memory section can be copied to the bus section along the vertical direction and to the target column along the horizontal direction inside the bus section. In the end, they are copied to the final position in the logic section. Once the data are copied to the defined location, the \( n \)-bit CLA calculation can be executed. Afterward, the output can be returned to the memory section in reverse sequence.

To conduct a hardware-based CLA demonstration, it requires at least 51 individual voltage sources for basing the gate, bit, and word lines of the 5 × 13 size of the 1T1M array independently. Due to a lack of hardware resources, instead, we developed a software-assisted in-memory computing emulator. Figure 5b
shows an emulating algorithm for one gate operation. The emulator is virtually mimicking the 1T1M crossbar structure and makes it possible to apply any voltage on the gate, bit, and word lines of the crossbar. Once the array is biased, the emulator calculates all the voltage potentials on the memristors \( v_{i,j} \) where \( i = [1:5] \) and \( j = [1:13] \) in the array. Then, \( v_{i,j} \) is applied to the memristor hardware and the response by \( v_{i,j} \) is obtained. Then, the outputs are returned to the emulator and updated, and \( v_{i,j} \) is recalculated. This process is repeated until every cell reaches a nonswitching condition. Once the steady state is obtained, one logical step is finished.

Figure 5c shows an initialized bitmap for a 4-bit CLA operation where the digital inputs are 1101 and 1001 and the carry-in is 0. Here, the darker cell and brighter cell refer to the HRS and LRS, respectively. Figure 5d shows the bitmap after performing the voltage sequence shown in Figure 4. A detailed step-by-step snapshot of the procedure can be found in Figure S12 and S13. Supporting Information. A full video of the 4-bit CLA operation procedure from the emulator can also be found in the Supporting Information.

In this study, we demonstrated a 4-bit CLA operation using a 1T1M array with the help of a software-based logic controller. For the successful demonstration, we systematically selected practically viable stateful logic gates that would stochastically guarantee a complete logic operation. Although our device was not commercially optimized in that device uniformity was not highly assured, we found that five gates were executable, and this allowed n-bit CLA operation in 5n + 8 computational steps with a \( (n + 1) \times 13 \) size array. To implement the stateful logic system, we proposed a new memory layout concept composed of memory, bus, and logic sections which were physically identical but functionally divided. This allowed effective data manipulation, avoiding the data traffic issue.

For the next advance of the stateful logic technology, some issues should be addressed. First, improving switching uniformity was found to be the most crucial factor for better energy and time efficiency. If the device uniformity can be improved so that 5SUM and 4CARRY gates become possible, the total number of steps for the full adder execution can be reduced to 2n. Second, an unwanted switching of memristor can occur after a long time due to the accumulation of the voltage stress or the nonlinear switching dynamics of the memristor.[52] For resolving the problem, one may insert a periodic refresh step that can eliminate the lurking error factor. Third, the increase in line resistance in the high-density device may cause device-to-device variation and worsen the success rate of the gate operation. The line resistance is a serial parasitic resistance component such that it can be compensated if \( R_S \) is controllable. Moreover, the control of \( R_S \) itself can enhance the functionality of the stateful logic.[20] Therefore, the next stateful logic technology should find a way on how to control the \( R_S \) value systematically.

The proposed in-memory computing system is not the most energy efficient, nor has the highest computing performance. The first virtue of the device is its low cost and small size, which is desirable for the cost-cutting computing of the IoT era, rather than performance-oriented computing. Considering the structural similarity between 1T1M and DRAM, whose price is about <1 dollar per 1 GB, we expect that a fully functional computing device can be integrated on a chip for less than a dollar, which would open a new computing market in the future.

**Experimental Section**

### Device Fabrication

A device with a minimum 2.8 μm transistor channel width with two levels of metallization for the gate, source, drain, and ground contacts was fabricated using a commercial foundry. A memristor cell with a Ta/TaOx/Pt structure was integrated at the top of the device in a lab. Its top (50 nm-thick Ta) and bottom (55 nm-thick Pt) electrodes were evaporated and patterned by e-beam lithography followed by a lift-off, and the 15 nm-thick TaOx layer was sputtered at Ar and O2 mixed ambient with 75% of O2 partial pressure.

### Electrical Measurement

Each device was measured using a probe station and semiconductor parameter analyzer (Keithley 4200A-SCS). To measure the characteristics of the 1T1M device, 1.2 V was applied to the transistor gate, and the current was obtained by sweeping the voltage on the top electrode, which was directly connected to the memristor. At this time, the bottom electrode and transistor body were connected to ground.

**Supporting Information**

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

**Acknowledgements**

This research was supported by the Ministry of Trade, Industry & Energy (MOTIE; grant nos. 20003655 and 20003789) and Korea Semiconductor Research Consortium (KSRC) support program for the development of the future semiconductor device. [Correction added after publication March 11, 2020: In Figure 2a and e, the reference numbers listed were wrong and have been corrected.]

**Conflict of Interest**

The authors declare no conflict of interest.

**Keywords**

in-memory computing, memristors, resistive switching memory, stateful logic, variations

---

[1] D. S. Jeong, K. M. Kim, S. Kim, B. J. Choi, C. S. Hwang, *Adv. Electron. Mater.* **2016**, *2*, 1.
[2] M. A. Zidan, J. P. Strachan, W. D. Lu, *Nat. Electron.* **2018**, *1*, 22.
[3] D. Ielmini, H. S. P. Wong, *Nat. Electron.* **2018**, *1*, 333.
[4] H. Wu, P. Yao, B. Gao, H. Qian, *Nat. Electron.* **2018**, *1*, 8.
[5] W.-H. Chen, C. Dou, K.-X. Li, W.-Y. Lin, P.-Y. Li, J.-H. Huang, J.-H. Wang, W.-C. Wei, C.-X. Xue, Y.-C. Chiu, Y.-C. King, C.-J. Lin, R.-S. Liu, C.-C. Hsieh, K.-T. Tang, J. J. Yang, M.-S. Ho, M.-F. Chang, *Nat. Electron.* **2019**, *2*, 420.
[6] J. Borghetti, G. S. Snider, P. J. Kuekes, J. J. Yang, D. R. Stewart, R. S. Williams, *Nature* **2010**, *464*, 873.
