(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
25 October 2018 (25.10.2018)

W!P O ¦ PCT

(10) International Publication Number
WO 2018/193338 Al

(51) International Patent Classification:
G11C 13/00 (2006.01) *H03K 19/20* (2006.01)
G11C 7/10 (2006.01)

(21) International Application Number:
PCT/IB20 18/052524

(22) International Filing Date:
10 April 2018 (10.04.2018)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
62/487,758 20 April 2017 (20.04.2017) US

(71) Applicant: **KING ABDULLAH UNIVERSITY OF SCIENCE AND TECHNOLOGY** [SA/SA]; 4700 King Abdullah University of Science and Technology, Thuwal, 23955-6900 (SA).

(72) Inventors: **NAOUS, Rawan;** 4700 King Abdullah University of Science and Technology, Thuwal, 23955-6900 (SA). **SALAMA, Khaled Nabil;** 4700 King Abdullah University of Science and Technology, Thuwal, 23955-6900 (SA).

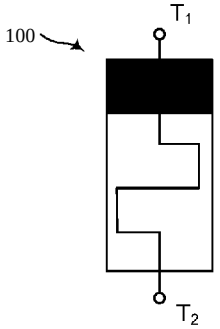
(81) Designated States *(unless otherwise indicated, for every kind of national protection available):* AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JO, JP, KE, KG, KH, KN, KP, KR, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available):* ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Declarations under Rule 4.17:
— *as to applicant's entitlement to apply for and be granted a patent (Rule 4.1 7(H))*
— *of inventorship (Rule 4.1 7(iv))*

Published:
— *with international search report (Art. 21(3))*
— *in black and white; the international application as filed contained color or greyscale and is available for download from PATENTSCOPE*

(54) Title: STOCHASTIC MEMRISTOR LOGIC DEVICES



101

! T1	T2	Voltage	State (z)
! 0	0	0	i z
! 0	¦	-Y	! OFF
! I	0	V	ON
! ¦	1	0	j z

FIG. 1A

(57) Abstract: In accordance with the present disclosure, one embodiment includes a memristor that is caused to be in a particular resistance state by a voltage applied across terminals of the memristor. A first logical input and a second logical input that are below a threshold voltage of the memristor are applied to a first terminal of the memristor. A first control input and a second control input are applied to a second terminal of the memristor. A logical output is determined based on a resistance state of the memristor.

WO 2 18/193338 A1 ®

STOCHASTIC MEMRISTOR LOGIC DEVICES

BACKGROUND

[0001] Resistive random access memory (ReRAM), or the memristor, is one of the prominent devices of emerging non-volatile memory technologies. Conventional logic design often prioritizes the repeatability and accuracy of an operation. This practice pushes the use of the available resources up to its limits and can lead to overdesigning and costly error correcting schemes to ensure precision. However, when dealing with applications where the notion of accuracy is not formally set, extra design margins can be added. A tradeoff arises between the level of accuracy and other performance metrics such as energy efficiency and delay. Thus far, the memristor has been integrated into applications that target the current systems' repeatability and accuracy constraints.

BRIEF DESCRIPTION OF THE DRAWINGS

[0002] Many aspects of the present disclosure can be better understood with reference to the following drawings. The components in the drawings are not necessarily to scale, emphasis instead being placed upon clearly illustrating the principles of the present disclosure. Moreover, in the drawings, like reference numerals designate corresponding parts throughout the several views.

[0003] FIG. 1A is an illustration of an example of a memristor and an example truth table of the memristor in accordance with the present disclosure.

[0004] FIG. IB illustrates a number of example graphs of performance characteristics of the memristor of FIG. 1A in accordance with the present disclosure.

[0005] FIG. 2 illustration of an example of a memristor logic device in accordance with the present disclosure.

[0006] FIGS. 3A-3C are illustrations and graphs that demonstrate the operation of a memristor logic device as a NAND gate in accordance with the present disclosure.

[0007] FIGS. 4A-4C are illustrations and graphs that demonstrate the operation of a memristor logic device as an AND gate in accordance with the present disclosure.

[0008] FIGS. 5A-5C are illustrations and graphs that demonstrate the operation of a memristor logic device as an OR gate in accordance with the present disclosure.

[0009] FIGS. 6A-6C are illustrations and graphs that demonstrate the operation of a memristor logic device as a NOR gate in accordance with the present disclosure.

[001 0] FIGS. 7A-7C are illustrations and graphs that demonstrate the operation of a memristor logic device as a half-adder in accordance with the present disclosure.

[001 1] FIGS. 8A-8E are illustrations and graphs that demonstrate the operation of a memristor logic device as a full adder in accordance with the present disclosure.

DETAILED DESCRIPTION

[001 2] Embodiments of the present disclosure involve approximate computing based on stochastic memristor logic devices. It combines emerging technologies and unconventional computing approaches. Moreover, it introduces a circuit-based technique to achieve approximation and adaptability.

[001 3] As discussed above, a tradeoff arises between the level of accuracy and other performance metrics such as energy efficiency and delay. As disclosed herein, a memristor can be used for variable or stochastic switching between two resistive states respectively. It offers opportunities to benefit from the intrinsic variations in investigating alternative approaches. This feature can increase performance and provide for reconfiguration and adaptability of arithmetic components and building blocks for image processing applications, quantum computing, and further applications.

[0014] The memristor can be used for realizing Boolean operators. Lowering the applied voltage levels shifts the memristor into a stochastic mode of operation. Variable switching behavior in this mode can lead to a probabilistic output for logic gates. approximate computing with memristor-based logic gates.

[0015] The principles disclosed herein relate to approximate computing based on stochastic memristive elements or memristor-based logic gates, along with the probabilistic analysis of the underlying behavior, and its impact on the overall performance. Moreover, it further introduces a circuit based technique to achieve approximation and adaptability. It essentially builds on the variability of the memristor that is related to the amplitude of the applied voltage along with the time period. This allows for configuring the accuracy, the energy level, and delay based on the design requirements and available resources. Hence, incorporating these elements within digital logic designs, the gates would then behave in a probabilistic manner. With a mathematical distribution for the switching events, and a sequential logic approach, a formulation for the output of the gates and its corresponding cascades can be established. This accounts for the probabilistic feature of the gates and the corresponding arithmetic block structure.

[0016] With reference to FIG. 1A, depicted is a memristor 100 with two terminals, a terminal Ti and a terminal T_2. A truth table 101 shows an example of the operation of the memristor 100, as will be discussed further below. The memristor 100 can be described as a non-linear circuit element that exhibits a change in its resistance based on the applied input bias. It is a non-volatile component that retains the value upon the removal of the input excitation. Two main variants are available with continuous and threshold-based behavior. The former undergoes a change with every level of voltage applied. The later on the other hand, does not show any change in case the applied voltage is below a set threshold, which primarily makes this type of devices suitable for the digital applications.

[0017] FIG. IB shows graphs 102, 104, 106, and 108, which exhibit examples of performance characteristics of the memristor 100. The graph 102 shows a hysteresis that relates the voltage and current through the memristor 100.

[001 8] Under the criteria of threshold based devices, several models are available in the literature that characterize this behavior. Either in direct fitting to the physical characteristics or mere abstraction to capture the threshold mechanism. For the modeling and simulation a bipolar model can be used. It offers a reliable and simple systematic approach to the resistance change and the intermediate state equations described as

$$/ = x^{-1} V_M \quad (1)$$

$$\frac{dx}{dt} = f(\dot{V}_M) W(x, V_M) \quad (2)$$

Where x represents the resistance of the memristor. The functions f(V M̀) and W(X,VM) model the threshold property and the window function respectively as follows

$$f(V_M) = \beta (V_M - 0.5)[¾ + V_t \setminus - \setminus V_M - V_t \setminus] \quad (3)$$

$$W(x, V_M) = e(V_M) 0 (R_{OFF} - x) + \theta(- v_M) \theta(\chi - R_{0N}) \quad (4)$$

[0019] Θ corresponds to the step function applied within the window function to confine the range of the resistance values between the high ROEF and low RON resistance states respectively. The rate of change is controlled by the parameter β, and the threshold voltage is set by V_t.

[0020] Aside from the generic modeling, the material characteristics play an important role in the underlying mechanisms of the memristor. The interfacial interactions of the internal composition can pose an added level of the operation, mainly the variability. In that perspective, different devices can show a stochastic feature, particularly in the switching between the two resistance states. In an abstraction scheme, the switching can be seen as an event occurring due to a set of input characteristics. The memristor operation can have a switching mechanism between resistance states that can be described or

approximated by a mathematical distribution such as the Poisson. To add this variable operation into the available models, modification of the physical equations governing the state and the ionic interactions needs to be established. However, in one approach can portray the corresponding behavior by mapping the change into the threshold voltage. That is, the threshold voltage is not represented as a hard value but rather varies according to the input states in the sub-threshold regime. Important parameters in the variability can be the input voltage and the time period. The smaller the voltage applied, the longer it takes for the switching to occur, and vice versa. This can be shown by the graphs 106 and 108, which show switching probability as a function of voltage and time, respectively. The average time τ for the device to switch and the applied voltage are exponentially related

$$log_{10}(\tau) = aV + \epsilon \quad (5)$$

With a and e being fitting parameters that are determined based on the device fabrication and material used. The values can be extracted as -2.67 and 5.43 accordingly. Moreover, the probability of switching within a time interval t was found to be following a Poisson distribution.

$$P_s(t) = 1 - e^{-t/\tau} \quad (6)$$

[0021] The graph 104 shows the resultant outcome of the stochastic process as a characteristic hysteresis with added switching points below the threshold. Nonetheless, despite this variability, the memristor can remain suitable for logic operations as it can toggle between two stable states. The ON state corresponding to the set operation with moving to **RON**, and the OFF state with moving to **ROFF**- Current sensing can be used to measure the output of the memristor. Accordingly, the low resistance state shows a high current that maps into digital '1 ', and the high resistance state gives a low current that

corresponds to digital '0'. Probabilistic analysis and formulation can be used to incorporate the probabilistic element into the digital computing realm, as discussed below. Several designs can be for the memristor 100 in logic design. For example, the imply operator can be the basis for Boolean gates. Where two memristors and a load resistor are used to sequentially and conditionally set/reset output according to q' ← p(imp)q. A CMOS-like logic can also be used, with polarized memristors resembling pmos and nmos transistors connected to form the complete set of the logic operators. A third approach builds on sequential logic, with several cycles needed to perform a logic operation such as a Boolean operation. In spite of the added cycles, only a single memristor is needed to perform 14 out of 16 logic operators. For example, AND, NAND, OR, and NOR can be performed with 3 cycles including an initialization cycle and two logical cycles that apply logical inputs to the terminals. Moreover, it all can be done within the compact memristive crossbar memory which paves the way for in-memory computing and further innovations as well. The basics of the sequential logic operations along with the added probabilistic outcomes are discussed further below.

[0022] The principles of operation of the sequential logic approach can use the two terminals of the memristor 100 as the inputs of any logic operator. Building on a bipolar switching, where opposite polarities are used for the SET and RESET of the memristor state of the memristor 100. A positive voltage, greater than the threshold, across the terminals Ti and T2 *(e.g.,* $Ti\text{-}T_2 > Vt$) would move the memristor into the ON state or '1', where as a negative voltage would shift it back to the OFF state or 'O'. In this approach, no change is reflected in the memristor state (z) in case the terminals T1 and T2 are at the same potential.

[0023] Referring back to FIG. 1A, the truth table 101 shows this behavior. In order to realize a boolean operation, a sequence of cycles can be applied. For the logic gate,

three cycles can be used to achieve the correct output of the operation. The first cycle can be considered an initialization cycle, where the memristor is put in a predetermined state, that would also facilitate the attainment of the required operation for the applied input parameters or logical inputs. For example, a logical T can be applied to Ti, and a logical '0' can be applied to T2 as an initialization stage, where Ti is greater than T_2. In other cases, Ti can be set less than T2 as an initialization stage. In some cases, T_1-T2 can be greater than the threshold V_t for some operational versions. Considering a device with set/reset threshold voltage of ± 4:6V, the applied voltage can be ± 5V to ensure proper switching and state change. However, once a stochasticity mode is in place, lower voltages *(e.g.* below the threshold voltage Vt) can trigger the switching to a stable state that would not change unless the reverse polarity voltage is applied. Hence, the gates can then behave in a probabilistic manner, but allow for better energy efficiency. Thereby, giving rise to the concept of approximate computing. The analysis of the gates under probabilistic operation is discussed further below.

[0024] Referring now to FIG. 2, a memristor logic device 200 is shown, that can perform logic and can also be referred to as a logic gate. The memristor logic device 200 has a memristor 100 and an output circuit 203. The memristor 100 has terminals Ti and T2 that can be used to control a state of the memristor 100. The memristor logic device 200 can perform different logic operations by applying a series of inputs or input cycles to the terminal Ti and T2, which can be considered terminals of the memristor logic device 200. The inputs can be applied by circuitry, such as an input circuit that includes passive and active circuit elements or components. Such an input circuit can control the timing an sequencing of logical inputs that can be applied to the terminals such as a particular voltage, a ground, or an open circuit. The output circuit 203 can give an output at an output terminal of the memristor logic device 200. The output circuit 203 can, for example, use

current sensing to determine a state of the memristor 100 after each of a number of inputs to the terminal Ti and T_2. The state that is sensed can be utilized to generate a probabilistic logical output, that can in some cases be used in another logic device such as another memristor logic device *(e.g.,* memristor logic devices can be cascaded). In some cases, the output circuit 203 can include passive circuit elements and/or active circuit elements such as op-amps.

[0025] Based on the sequential logic highlighted earlier, a single memristor behavior governs the output values. Hence, under sub-threshold excitation the memristor can be characterized by a probability of switching $P_s(t)$. With the incorporation of this variability, a set of assumptions can be used for analysis to ensure a common ground and to provide a simple overview of the expected outcome.

1) The initialization stage is deterministic

2) The inputs p and q to the logic gates are deterministic

3) The operation of the memristor is probabilistic with a generic switching probability $P_s(t)$

[0026] Thus, instead of having a certain correct output, the values are masked with the probability of switching and the sequence of applied input parameters. For each entry in the truth table, depending on the input setting, the certainty of having a correct output can be characterized by P_{out}. For each gate the output for every entry in the truth table can hold a probability to be correct. This probability can be determined by the switching probability of the memristor and the sequence of input application.

[0027] Moving to FIG. 3A, shown is a NAND gate 303, and a memristor logic device 200 that is configured to operate as a NAND gate. A series of control inputs along with logical inputs can be applied to the terminals Ti and T_2 such that the memristor logic device 200 operates as a NAND gate. For example, a sequence of inputs can be applied to the memristor logic device 200 at the terminals Ti and T_2, including an initialization cycle and

two logical cycles. Note that in some cases the logical cycles can include logical inputs p and q being input sequentially to the memristor logic device 200 at one of the terminals Ti and T2 *(e.g.,* in a particular order), while a specific set of control inputs are applied sequentially to the other one of the terminals Ti and T_2.

[0028] In NAND operation, the first cycle can be an initialization cycle and can apply control input 1, to Ti and control input 0 to T_2. The second cycle can apply a control input 0 to Ti and a logical input q to T_2. The third cycle can apply a control input 1 to Ti and a logical input p to T_2. The state of the memristor 100 of the memristor logic device 200 once the third cycle is performed can cause a probabilistic logical output of the memristor logic device 200. In some cases, the initialization cycle control input T can be deterministic. Also, the voltage of the control input '1' can be greater than the threshold Vt for the initialization cycle. In some cases, the logical inputs and/or control inputs after the initialization cycle can be a lower voltage *(e.g.,* lower than Vt).

[0029] FIG. 3B includes a table 312 that shows one example of inputs to Ti and T2 that can cause the memristor logic device 200 to operate as a NAND gate. For example, the three cycles applied to Ti can be (1, 0, 1) respectively, where the second two cycles (0,1) after the initialization input can be referred to as control inputs that can cause the memristor logic device 200 to operate as a NAND gate. The three cycles applied to T2 can be (0, q, p) respectively, where q and p are logical inputs to the memristor logic device 200. The state of the memristor 100 of the memristor logic device 200 once the third cycle is performed can cause a probabilistic logical output of the memristor logic device 200. The probabilistic output of the memristor logic device 200 operating as a NAND gate for each set of logical inputs can be described by P_{out}, as shown in the equations in table 315.

[0030] Details in the analysis for a NAND gate and all remaining gates can follow [[the]] a similar construct and flow, but can have their own set of probabilities and

consequent behavior. For the input pair (p, q), four different combinations are available. For (p, q) = (0, 0), the first cycle can put the memristor in the ON state by applying control input 1, to Ti and control input 0 to T_2. The second cycle will not have an effect on the state of the memristor as a zero potential can put across its terminals. For the third cycle, a positive potential may not have an effect in changing the state as it is already in the ON state. Thus, having the pair (0, 0) can always produce a correct output, and P(l) = 1. For the pair (0,1), to get a correct one at the output, the memristor should switch to the ON state in the third cycle or it should not switch in the second and third cycles. The probability of getting the correct out for (0,1) is then

$$Po,i(X) = \boldsymbol{P_s(t) + (1 - p_s(t))^2}$$
$$P_{0,1}(1) = 1 - P_s(t) + P_s^2(t) \quad (7)$$

[0031] For the pair (1, 0), zero potential can be applied in the second and third cycles. This can move the initialization state directly to the output, so P(l) = 1. For the pair (1, 1), the second cycle can determine whether the output is correct or not, as the third cycle has zero potential and the previous state would then impact the output. Thus, to have a correct '0' at the output, the memristor can switch to the OFF state in the second cycle. Thus, P(0) = Ps(t). The overall probabilities of the NAND gate to get the '0' and the T correct at the output for the pair (1,1) are

$$P_{out}(0) = P_S(t)$$
$$\boldsymbol{PoutW} = \frac{1}{3}[3 - P_s(t) + P_s^2(t)] \quad (8)$$

[0032] In FIG. 3C, graphs 318 and 321 show the output probabilities for $P_{ou}t(l)$ and Pout(0), respectively. In each graph, the hard line represents the theoretical probabilistic analysis presented earlier. The points represent the simulation verification with the memristor element. The three cycle operation of the NAND gate can be applied on the memristor in sequence. All the truth table entries can be tested for 100 times each. The

simulation can be run over a period of 1ms and the voltage ranged between 2V and 4V accordingly. The number of matches can be counted for each run and the overall probabilities can be calculated based on the expected outcome for the gate. In addition to the single bit probabilities, the behavior of the gate can be further quantified with the notion of accuracy or precision. It can be specified as

$$Accuracy = \frac{N_0}{N}\mathbf{P}_{out}\mathbf{(0)} + \frac{N}{N}P_{0Ut}(X) \quad (9)$$

[0033] where N corresponds to the total number of entries in a particular truth table. Ni is the number of entries with the output set to '1', and No is the number of entries with the output set to '0'. The accuracy for the NAND operator can be calculated as

$$Accuracy(NAND) = \frac{1}{4}[3 + P_s^2(t)] \quad (10)$$

[0034] Graph 324 shows the accuracy of the NAND gate with respect to the switching probability of the memristor. The theoretical precision attained can be verified with simulation of the memristor for 100 times for the all of the input pairs, and recording the output states. The NAND gates shows high accuracy starting from 75%. It is due to the deterministic initialization state that shifts the ON state to the output in two of the possible gate input combination. Hence, a large space for saving in terms of simple logic operation is possible with the scaling of the voltage and benefiting from the stochastic feature of the memristor 100. A similar analysis is conducted for all the remaining gates that require 3 cycles for operation, *i.e.* AND, OR, and NOR. When cascading the gates to form an arithmetic device that performs an arithmetic operation, a probabilistic dimension is added into the analysis and requires the accommodation of further parameters into account.

[0035] FIG. 4A shows an AND gate 403, with logical inputs p and q. FIG. 4A also shows the memristor logic device 200 configured to operate as an AND gate. A series of control inputs along with logical inputs can be applied to the terminals Ti and T2 such that

the memristor logic device 200 operates as an AND gate. For example, the first cycle can be an initialization cycle and can apply control input 1, to Ti and control input 0 to T_2. The second cycle can apply a logical input p to Ti and a control input 1 to T_2. The third cycle can apply a logical input q to Ti and a control input 1 to T_2. The state of the memristor 100 of the memristor logic device 200 once the third cycle is performed can cause a probabilistic logical output of the memristor logic device 200.

[0036] FIG. 4B includes a table 412 that shows one example of inputs to Ti and T2 that can cause the memristor logic device 200 to operate as an AND gate. For example, the three cycles applied to Ti can be (1, p, q) respectively, where p and q are the logical inputs to the AND gate. The three cycles applied to T2 can be (0, 1, 1) respectively, where 1,1 can be control inputs that cause the memristor logic device 200 to operate as an AND gate. The first cycle *(e.g.,* where Ti=l, and T2=0) can be referred to as an initialization cycle or initialization inputs. The state of the memristor 100 of the memristor logic device 200 once the third cycle is performed can cause a probabilistic logical output of the memristor logic device 200. The probabilistic output of the memristor logic device 200 operating as an AND gate for each set of logical inputs can be described by P_{out}, as shown in the equations in table 415.

[0037] FIG. 4C shows graphs 418, 421, and 424. Graphs 418 and 421 show the output probabilities for $P_{out}(l)$ and $P_{out}(0)$, respectively. In each graph, the hard line represents the theoretical probabilistic analysis presented earlier. The points represent the simulation verification with the memristor element. Graph 424 shows the accuracy of the AND gate with respect to the switching probability of the memristor.

[0038] FIG. 5A shows an OR gate 503, with logical inputs p and q. The memristor logic device 200 is shown with inputs configured to cause it to operate as an OR gate. A series of control inputs along with logical inputs can be applied to the terminals Ti and T2

such that the memristor logic device 200 operates as an OR gate. For example, the first cycle can be an initialization cycle and can apply control input 1, to Ti and control input 0 to T_2. The second cycle can apply a control input 0 to Ti and a logical input q to T_2. The third cycle can apply a control input 1 to Ti and a logical input p to T_2. The state of the memristor 100 of the memristor logic device 200 once the third cycle is performed can cause a probabilistic logical output of the memristor logic device 200.

[0039] FIG. 5B includes a table 512 that shows one example of inputs to Ti and T2 that can cause the memristor logic device 200 to operate as an OR gate. For example, the three cycles applied to Ti can be (1, p, q) respectively, where p and q are the logical inputs to the OR gate. The three cycles applied to T2 can be (0, 1, 0) respectively, where the second two cycles (1,0) can be control inputs that cause the memristor logic device 200 to operate as an OR gate. The first cycle *(e.g.,* where Ti=l, and T2=0) can be referred to as an initialization cycle or initialization inputs. The state of the memristor 100 of the memristor logic device 200 once the third cycle is performed can cause a probabilistic logical output of the memristor logic device 200. The probabilistic output of the memristor logic device 200 operating as an OR gate for each set of logical inputs can be described by P_{out}, as shown in the equations in table 515.

[0040] FIG. 5C shows graphs 518, 521, and 524. Graphs 518 and 521 show the output probabilities for $P_{out}(1)$ and $P_{out}(0)$, respectively. In each graph, the hard line represents the theoretical probabilistic analysis presented earlier. The points represent the simulation verification with the memristor element. Graph 524 shows the accuracy of the OR gate with respect to the switching probability of the memristor.

[0041] FIG. 6A shows a NOR gate 603, with logical inputs p and q. The memristor logic device 200 is shown with inputs configured to cause it to operate as a NOR gate. A series of control inputs along with logical inputs can be applied to the terminals Ti and T2

such that the memristor logic device 200 operates as a NOR gate. For example, the first cycle can be an initialization cycle and can apply control input 1, to Ti and control input 0 to T_2. The second cycle can apply a control input 0 to Ti and a logical input q to T_2. The third cycle can apply a control input 0 to Ti and a logical input p to T_2. The state of the memristor 100 of the memristor logic device 200 once the third cycle is performed can cause a probabilistic logical output of the memristor logic device 200.

[0042] FIG. 6B includes a table 612 that shows one example of inputs to Ti and T2 that can cause the memristor logic device 200 to operate as a NOR gate. For example, the three cycles applied to Ti can be (1, 0, 0) respectively, where the second two cycles (0,0) can be control inputs that cause the memristor logic device 200 to operate as a NOR gate. The three cycles applied to T2 can be (0, q, p) respectively, where p and q are the logical inputs to the NOR gate. The first cycle *(e.g.,* where Ti=l, and T2=0) can be referred to as an initialization cycle or initialization inputs. The state of the memristor 100 of the memristor logic device 200 once the third cycle is performed can cause a probabilistic logical output of the memristor logic device 200. The probabilistic output of the memristor logic device 200 operating as an OR gate for each set of logical inputs can be described by P_{out}, as shown in the equations in table 615.

[0043] FIG. 6C shows graphs 618, 621, and 624. Graphs 618 and 621 show the output probabilities for $P_{out}(1)$ and $P_{out}(0)$, respectively. In each graph, the hard line represents the theoretical probabilistic analysis presented earlier. The points represent the simulation verification with the memristor element. Graph 624 shows the accuracy of the NOR gate with respect to the switching probability of the memristor. Basic arithmetic operations can build on the main logic gates that can be formed by a combination of the principles disclosed herein.

[0044] FIG. 7A shows a half-adder 703, and a corresponding memristor half-adder 704. The memristor half-adder 704 is an arithmetic device that includes a memristor OR gate 705, a memristor NAND gate 706, and a memristor AND gate 707 composing an XOR gate. The memristor half-adder performs a half-adder operation, which is an arithmetic operation. For incorporating the stochastic memristor-based gates into an arithmetic operator, a basic building block can be a half adder. It can be mainly composed of an XOR gate for calculating the sum bit, comprising the OR gate 705, the NAND gate 706, and the AND gate 707. Another AND gate is used for the carry bit, as shown in the schematic of the half-adder 703. The logic equations for the output bits So and Co are

$$S_0 = p \ (XOR) \ q \tag{11}$$

$$C_0 = p \ (AND) \ q \tag{12}$$

However, as illustrated earlier, only 4 out of the 6 logic gates operators are possible with a single memristor. The remaining gates of XOR and XNOR can be attained by a combination of the basic gates. For example, The XOR is obtained by applying the OR gate 705, and the NAND gate 706 to the inputs p and q and then putting the results back into the AND gate 707. The equation of the operation is

$$p(XOR)q = (p(OR)q) \ AND \ (p \ (NAND) q) \tag{13}$$

[0045] So, in order to analyze the behavior of this XOR gate, in this example the assumption of having deterministic inputs p and q may not hold valid. Especially at the input of the AND gate used to calculate the sum. For cascaded logic both inputs of the second stage logic can be probabilistic. The stochastic behavior of the memristor, and the gates in the first cycle can impose different input conditions on the following gates.

[0046] Note that because the logic is sequential, the cycles applied to each memristor gate can be designed to be staggered for proper sequential operation. For example, the third cycle ***(e.g.*** the output cycle) of the OR gate 705 may not complete at the same time

as the third cycle of the NAND gate 706. Instead, they can complete sequentially, such that one of the gate outputs can be the logical 'p' and the other gate output can be the sequentially applied logical 'q' input to the AND gate 707. In some cases, there can also be circuits that apply the control input '1' to the top terminal of the AND gate 707 before the p and q inputs are received sequentially from the OR gate 705 and the NAND gate 706. In other cases, all inputs to the AND gate 707 can be probabilistic.

[0047] Further to the analysis of the AND gate, a distinct set of output probabilities can be formulated along with the precision of the sum and carry bit. The input probabilities can be assigned as $P_{in}(1)$ and $P_{in}(0)$. The probabilities for the inputs p and q are considered the same as they are coming from the outputs of the OR and NAND gates in the first stage that can exhibit similar probabilistic operation in terms of the output probabilities for '1' and '0'. The output probabilities for the AND gate with probabilistic inputs are

$$P_{out}(0) = \frac{1}{3}(3 + (2Pinif) - P_{in}(X) - 2)P_s(t) + (1 - P_{in}(1) - P_{in}^2(0) + P\&(0)P_{in}(l))P_s^2(t)) \quad (14)$$

$$P_{out}(1) = Pin(1)P_s(t) + (p_{in}(i) - i)p_{in}(i)p_s(t) \quad (15)$$

[0048] FIG. 7B shows a truth table 709 for a half adder. The sum bit So can have two entries for '0' and two entries for T . Thereby, the accuracy or precision of the sum bit is specified as

$$Accuracy\{S_0) = \frac{i}{2}(P_{out}(0) + P_{out}(l)) \quad (16)$$

[0049] For the carry bit Co, the accuracy can be that of the deterministic input AND gate that was shown earlier in Figure 5. It can be primarily dependent on the switching probability Ps(t) as its inputs can be deterministic.

$$Accuracy(C_0) = \frac{1}{4}(1 + 4P_s(t) - P_s^2(t)) \quad (17)$$

[0050] The theoretical analysis can also be verified with simulation of the gates based on the stochastic memristor.

[0051] FIG. 7C includes graphs 718, 721, and 724. Graph 718 shows the accuracy measure in response to the level of voltage applied across the terminals of the memristor. Savings up to 18% in the voltage can have a minor impact on the accuracy, which can decrease drastically with lower values. FIG. 721 shows the accuracy as a function of the delay for a voltage of 3.7V and 3V respectively. As depicted, the accuracy of operation of the adder can increase with larger application time overlying the probabilistic behavior of the memristor. A compromise can arise between the delay and the accuracy achieved for a particular input voltage. This feature of approximate computing can be highly efficient with higher bit adders, especially with the added saving per block. The graph 724 shows the accuracy for the sum and carry bit in the theoretical and simulation forms. As noticed, the performance of the sum bit starts off at 50% as two cases are present for the high bit and since the AND gate produces accurate '1'. The accuracy of the carry bit can start with a lower performance than the sum, but can catch up at a switching probability of 0.3, and can have better performance onward.

[0052] FIG. 8A shows a schematic of a full adder 803. In the half-adder, the XOR gate initiated with deterministic inputs. Thus, only the last AND gate required analysis with probabilistic inputs. However, the arithmetic block can be extended to include the carry in bit Cin, and an additional XOR operation can be performed to calculate the sum Si. Whereas the carry requires two AND operations and an OR to get the final carry out

Ci. FIG. 8B shows a truth table 812 for the full adder. The equations for the output bits can be specified as

$$S_1 = (p \ (XOR)\ q) \ XOR \ C_{in} \quad (18)$$

$$C_1 = (p \ (AND)\ q) \ OR \ ((p \ (XOR)\ q) \ AND \ C_{in}) \quad (19)$$

[0053] The calculation of the output bits passes through several stages of gates. Hence, the inputs are not deterministic as the cascade induces the notion of variability or probabilistic outcomes. In a similar fashion to the analysis for the gates provided in the previous section, the output probabilities and the corresponding accuracy for each of the logic gates can be formulated based on non-deterministic input parameters. These equations can be tested with the output accuracy of the sum and carry bits.

[0054] FIG. 8C includes graphs 818 and 821, showing results of this analytical approach for SI and CI, respectively. The simulation can include the theoretical formulation based on the probabilities of the inputs and its consequent relation and cascades. In addition, a system level simulation of the memristor element-based structure can be conducted. A 100 simulation runs for the full architecture were conducted, and the resultant output can be noted in regards to the expected output behavior for all the entries in the truth table 812.

[0055] Moreover, there can be a relatively small degradation for the full adder in comparison to the half adder. This is due to the fact that both bits So and Si can be outputs of an XOR gate, but the first can be with deterministic inputs and the latter can be with probabilistic inputs. Thus, in this case the stochasticity can have a negative impact on performance. However, when comparing the carry bits Co and Ci, the performance can be enhanced with the added probabilistic operation. As shown, the accuracy of the full adder can have better performance up until around 0.3 probability of switching, after which, the accuracy for the half adder carry can show faster improvements.

[0056] FIG. 8D includes a graph 825 that shows the sum accuracy plot with respect to the switching probability for various numbers of bits. To have a general overview of the system performance under several cascades, an N-bit adder can output characteristics as shown. A ripple-carry adder structure can be adopted. In some cases, degradation of performance can be saturated after 4 cascades. That is the accuracy of the sum bit, and similarly for the carry bit, can show a consistent behavior for adders with 4 bits and higher. That is, the effect of probabilistic components can fade or be balanced with the internal operation of the probabilistic internal adder blocks.

[0057] FIG. 8E includes a graph 828 and a graph 831. These graphs show examples of the configurable accuracy, delay, and energy relationship of the present disclosure. The graph 828 shows an example of accuracy as a function of voltage applied, for a number of time delays. The graph 831 shows an example of input voltage as a function of applied time period, for a number of precision measures. Depending on the applied time period, a 100% precision of operation can be achieved at a lower nominal voltage at the expense of longer delay. Fixed accuracy levels for a range of voltages and time can allow for diverse options in the design space allocation.

[0058] Although embodiments have been described herein in detail, the descriptions are by way of example. The features of the embodiments described herein are representative and, in alternative embodiments, certain features and elements may be added or omitted. Additionally, modifications to aspects of the embodiments described herein may be made by those skilled in the art without departing from the spirit and scope of the present invention defined in the following claims, the scope of which are to be accorded the broadest interpretation so as to encompass modifications and equivalent structures.

CLAIMS

Therefore, at least the following is claimed:

1. A method comprising:

initializing a memristor by applying, in a first cycle, a voltage across a first terminal and a second terminal of the memristor, wherein initializing the memristor causes the memristor to be in a particular resistance state;

applying, in a second cycle, a first logical input to the first terminal of the memristor and a first control input to the second terminal of the memristor, the first logical input being below a threshold voltage of the memristor;

applying, in a third cycle, a second logical input to the first terminal of the memristor and a second control input to the second terminal of the memristor, the second logical input being below the threshold voltage of the memristor; and

determining a logical output of a memristor logic device comprising the memristor based on a resistance state of the memristor once the second logical input is applied.

2. The method of claim 1, wherein the memristor logic device is performing a logic operation that is at least one of an AND operation, an OR operation, a NAND, or a NOR operation at least in part by applying the first cycle, the second cycle, and the third cycle.

3. The method of claim 1, further comprising determining a probability that the logical output is a correct logical output for a logic operation being performed by the memristor logic device.

4. The method of claim 1, wherein determining the logical output further comprises current sensing to detect the resistance state of the memristor.

5. The method of claim 1, wherein the memristor logic device is a first memristor logic device, and further comprising

applying the logical output of the first memristor logic device to an input of another a second memristor logic device; and

performing an arithmetic operation using an arithmetic device comprising a plurality of memristor logic devices including the first_memristor logic device and the second memristor logic device.

6. The method of claim 5, wherein performing the arithmetic operation further comprises:

generating an arithmetic output of the arithmetic device; and

determining a probability that the arithmetic output is a correct arithmetic output of the arithmetic operation.

7. An apparatus comprising:

a memristor comprising a first terminal, a second terminal, and an output terminal;

an input circuit that applies a sequence of inputs to the first terminal and a corresponding sequence of inputs to the second terminal in a plurality of cycles comprising

an initialization cycle that causes the memristor to be in a particular state; and

at least one logic cycle that applies a logical input to the first terminal and a control input to the second terminal, wherein a voltage across the first terminal and the second terminal remains below a threshold voltage of the memristor for the at least one logic cycle; and

wherein a logical output at the output terminal is based on a resistance state of the memristor once the sequence of inputs to the first terminal and the corresponding sequence of inputs to the second terminal are applied.

8. The apparatus of claim 7, wherein the memristor logic device is configured to perform a logic operation that is at least one of an AND operation, an OR operation, a NAND, or a NOR operation based at least in part on the at least one logic cycle.

9. The apparatus of claim 7, further comprising:

output circuitry that determines a probability that the logical output is a correct logical output for a logic operation being performed by the apparatus.

10. The apparatus of claim 7, further comprising an output circuit that performs current sensing to detect the resistance state of the memristor and generates the logical output at the output terminal.

11. The apparatus of claim 7, wherein the memristor is a first memristor, and further comprising:

an arithmetic device that performs an arithmetic operation, the arithmetic device comprising the first memristor configured to perform a first logic operation; and

at least one second memristor coupled to an output of the first memristor and configured to perform a second logic operation.

12. The apparatus of claim 11, further comprising output circuitry configured to:

determine an arithmetic output of the arithmetic device; and

determine a probability that the arithmetic output is a correct arithmetic output.

13. A logic device, comprising:

a memristor comprising a first terminal, a second terminal, and an output terminal;

input circuitry coupled to the first terminal, and the second terminal, the input circuitry configured to apply a sequence of inputs to the memristor to cause the memristor to operate as the logic device, wherein the sequence of inputs comprises:

an initialization cycle that initializes a state of the memristor; and

at least a first logic cycle that that includes applying a logical input to the first terminal and a control input to the second terminal, wherein a voltage across the first terminal and the second terminal remains below a threshold voltage of the memristor for the at least one logic cycle; and

output circuitry configured to determine a logical output at the output terminal based on a resistance state of the memristor once the sequence of inputs are applied.

14. The logic device of claim 13, wherein the logic device is configured to perform a logic operation that is at least one of an AND operation, an OR operation, a NAND, or a NOR operation based at least in part on the at least one logic cycle.

15. The logic device of claim 13, wherein the output circuitry is further configured to:

determine a probability that the logical output is a correct logical output for a logic operation being performed by the memristor logic device.

16. The logic device of claim 13, wherein the output circuitry is further configured to:

perform current sensing to detect the resistance state of the memristor and generates the logical output at the output terminal.

17. The logic device of claim 13, wherein the logic device is configured to operate as a NAND gate, and wherein the sequence of inputs comprises:

the initialization cycle comprising applying a control input of 1 to the first terminal and a control input of 0 to the second terminal;

a first logic cycle comprising applying a control input of 0 to the first terminal and a logical input of q to the second terminal; and

a second logic cycle comprising applying a control input of 1 to the first terminal and a logical input of p to the second terminal.

18. The logic device of claim 13, wherein the logic device is configured to operate as a AND gate, and wherein the sequence of inputs comprises:

the initialization cycle comprising applying a control input of 1 to the first terminal and a control input of 0 to the second terminal;

a first logic cycle comprising applying a logical input of p to the first terminal and a control input of 1 to the second terminal; and

a second logic cycle comprising applying a logical input of q to the first terminal and a contrail input of 1 to the second terminal.

19. The logic device of claim 13, wherein the logic device is configured to operate as a OR gate, and wherein the sequence of inputs comprises:

the initialization cycle comprising applying a control input of 1 to the first terminal and a control input of 0 to the second terminal;

a first logic cycle comprising applying a control input of 0 to the first terminal and a logical input of q to the second terminal; and

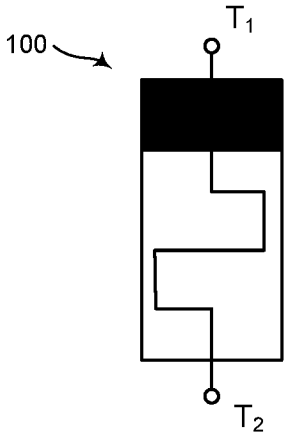
a second logic cycle comprising applying a control input of 1 to the first terminal and a logical input of p to the second terminal.

20. The logic device of claim 13, wherein the logic device is configured to operate as a NOR gate, and wherein the sequence of inputs comprises:

the initialization cycle comprising applying a control input of 1 to the first terminal and a control input of 0 to the second terminal;

a first logic cycle comprising applying a control input of 0 to the first terminal and a logical input of q to the second terminal; and

a second logic cycle comprising applying a control input of 0 to the first terminal and a logical input of p to the second terminal.



101

T_1	T_2	Voltage	State (z)
0	0	0	z
0	1	-V	OFF
1	0	V	ON
1	1	0	z

FIG. 1A

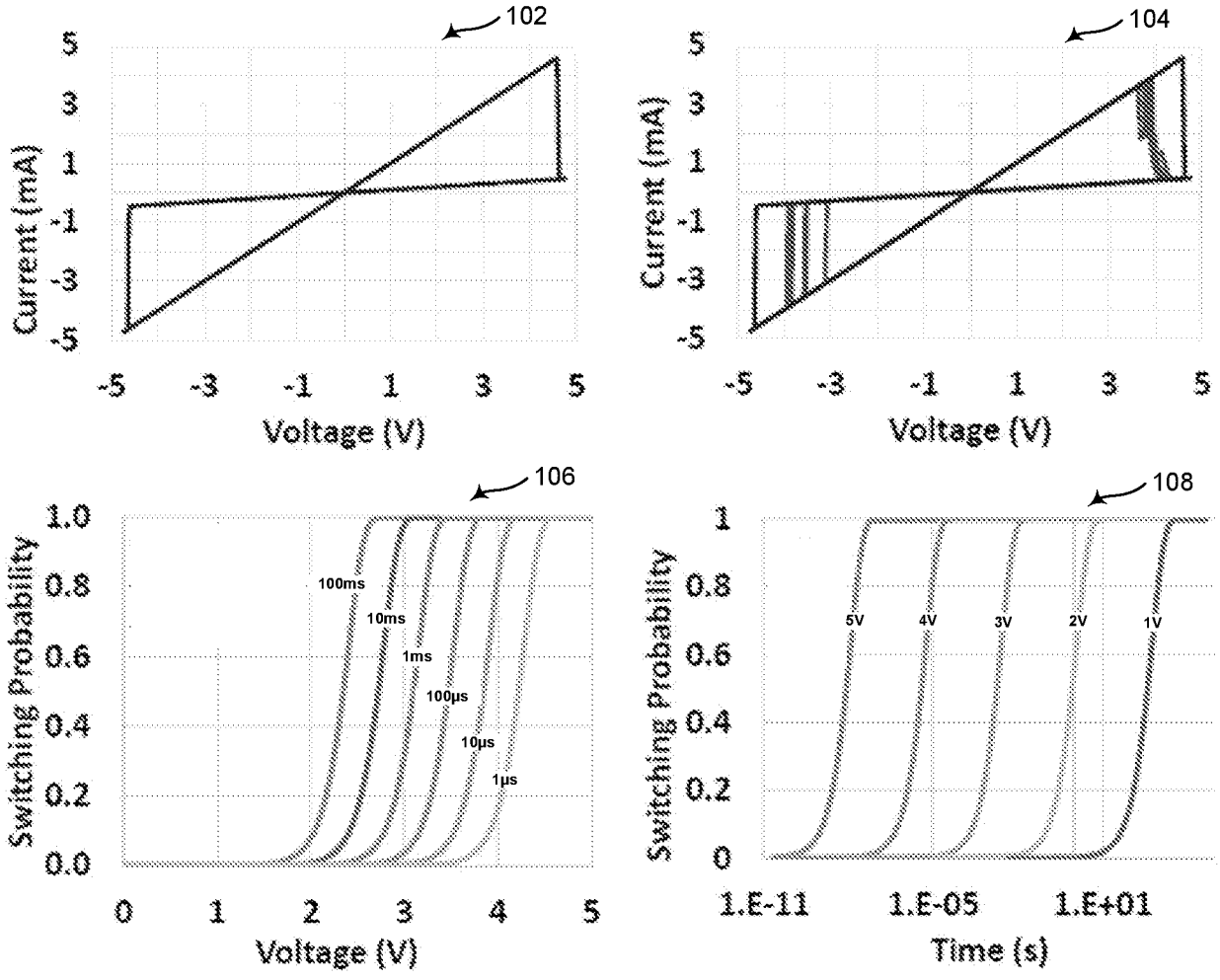


FIG. 1B

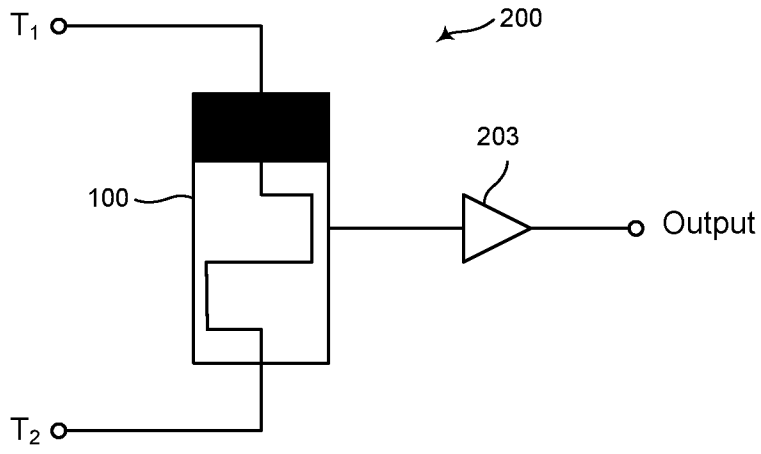


FIG. 2

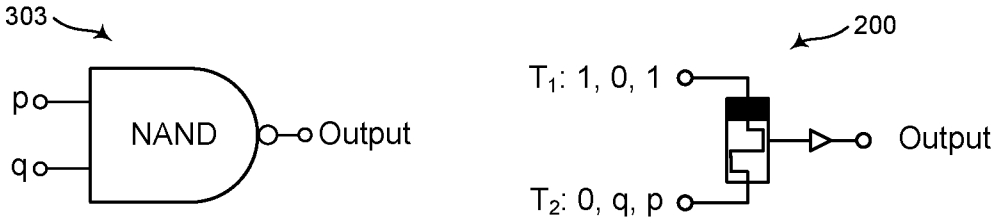


FIG. 3A

312

	Cycle 1	Cycle 2	Cycle 3
T_1	'1'	'0'	'1'
T_2	'0'	q	p

315

p	q	State (z_1)	State (z_2)	State (z_3)	Output	P_{out}
0	0	ON	ON	ON	'1'	$P_1 = 1$
0	1	ON	OFF	ON	'1'	$P_1 = 1 - P_s(t) + P_s^2(t)$
1	0	ON	ON	ON	'1'	$P_1 = 1$
1	1	ON	OFF	OFF	'0'	$P_0 = P_s(t)$

FIG. 3B

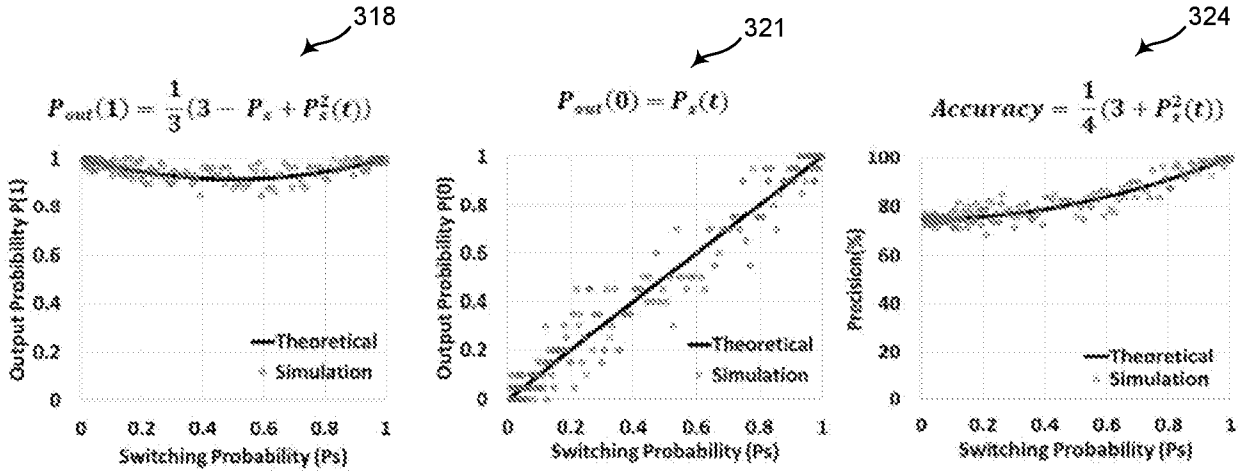


FIG. 3C

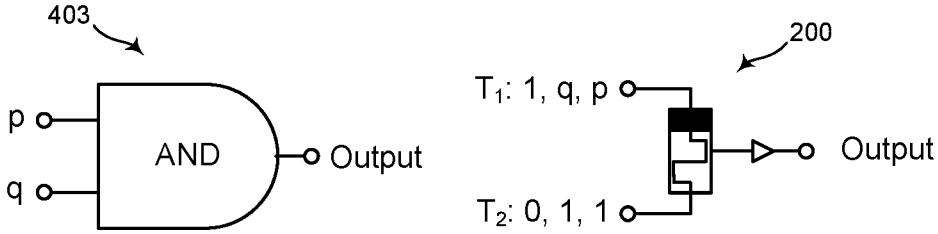


FIG. 4A

412

	Cycle 1	Cycle 2	Cycle 3
T_1	'1'	p	q
T_2	'0'	'1'	'1'

415

p	q	State (z_1)	State (z_2)	State (z_3)	Output	P_{out}
0	0	ON	OFF	OFF	'0'	$P_0 = 2\,P_s(t)-P_s^2(t)$
0	1	ON	OFF	OFF	'0'	$P_0 = P_s(t)$
1	0	ON	ON	OFF	'0'	$P_0 = P_s(t)$
1	1	ON	ON	ON	'1'	$P_1 = 1$

FIG. 4B

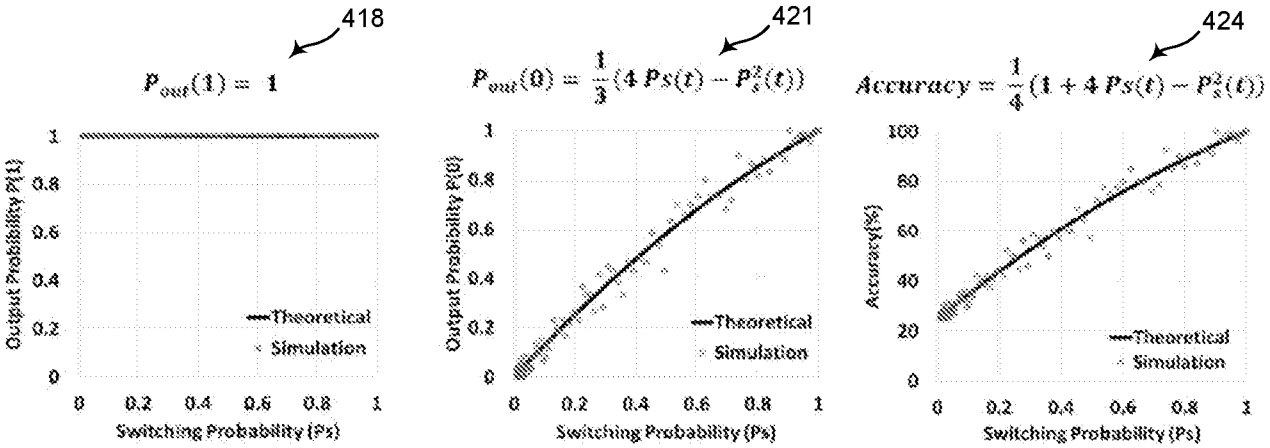


FIG. 4C

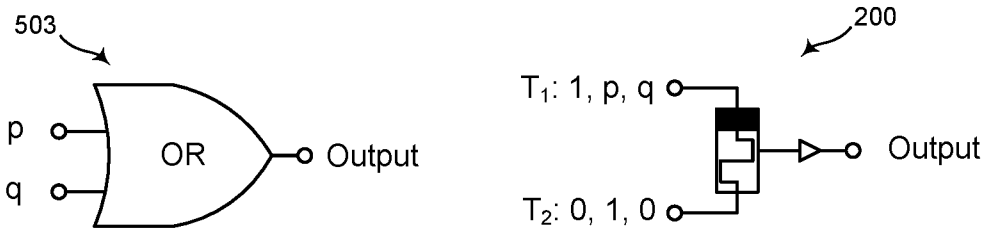


FIG. 5A

OR

512

	Cycle 1	Cycle 2	Cycle 3
T_1	'1'	p	q
T_2	'0'	'1'	'0'

515

p	q	State (z_1)	State (z_2)	State (z_3)	Output	P_{out}
0	0	ON	OFF	OFF	'0'	$P_0 = P_s(t)$
0	1	ON	OFF	ON	'1'	$P_1 = 1 - P_s(t) + P_s^2(t)$
1	0	ON	ON	ON	'1'	$P_1 = 1$
1	1	ON	ON	ON	'1'	$P_1 = 1$

FIG. 5B

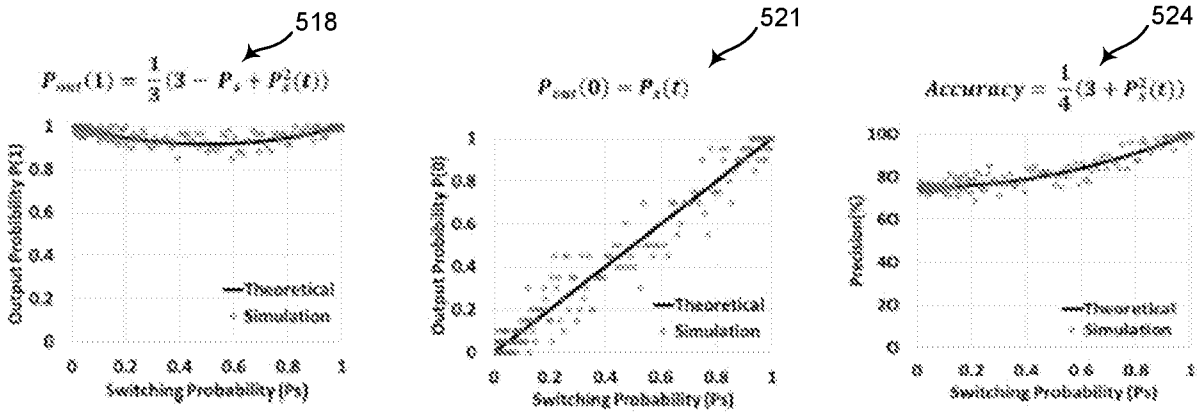


FIG. 5C

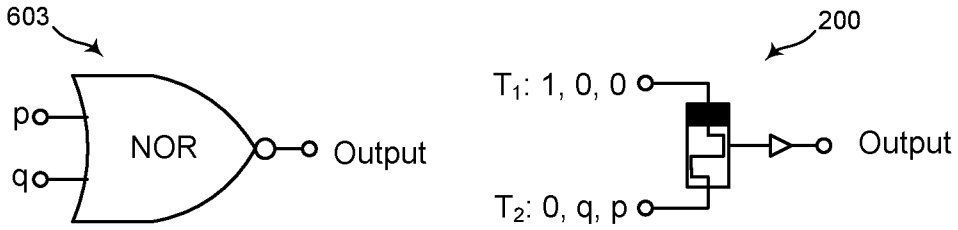


FIG. 6A

612

	Cycle 1	Cycle 2	Cycle 3
T_1	'1'	'0'	'0'
T_2	'0'	q	p

615

p	q	State (z_1)	State (z_2)	State (z_3)	Output	P_{out}
0	0	ON	ON	ON	'1'	$P_o = 1$
0	1	ON	OFF	OFF	'0'	$P_o = P_s(t)$
1	0	ON	ON	OFF	'0'	$P_o = P_s(t)$
1	1	ON	OFF	OFF	'0'	$P_o = 2\,P_s(t) - P_s^2(t)$

FIG. 6B

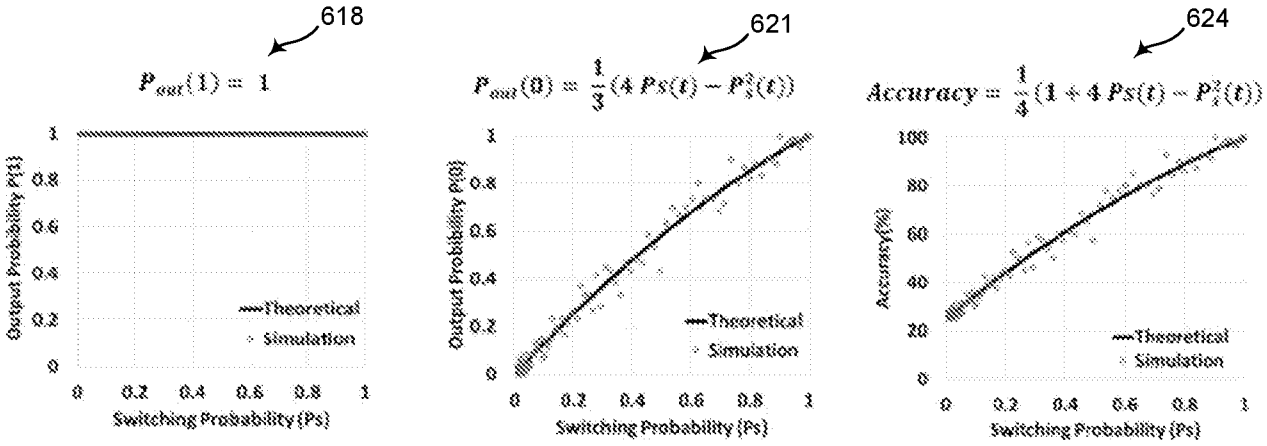


FIG. 6C

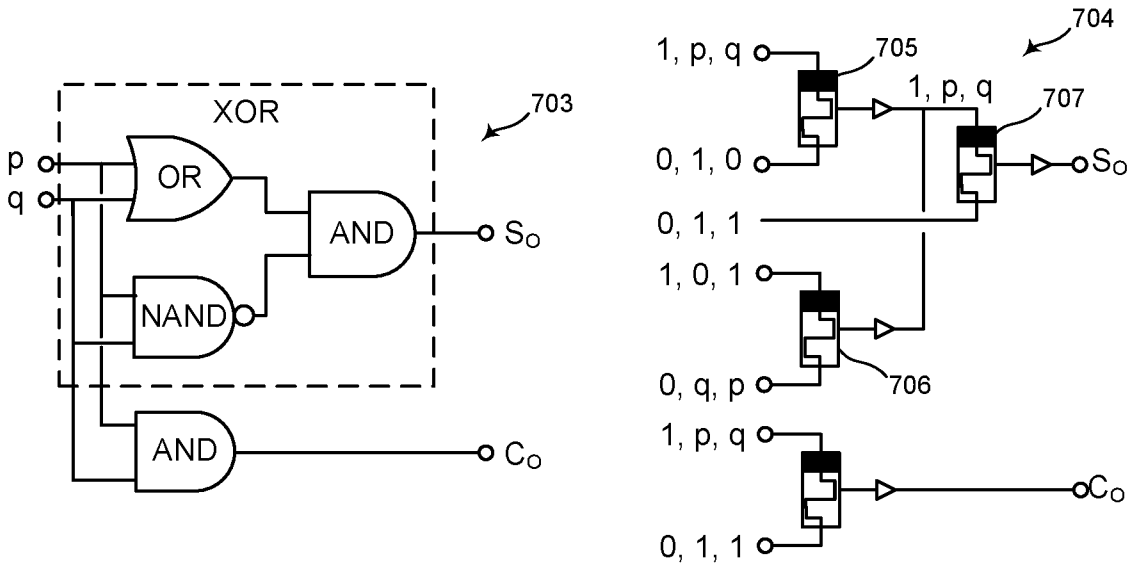


FIG. 7A

709

p	q	Sum (S_o)	Carry (C_o)
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

FIG. 7B

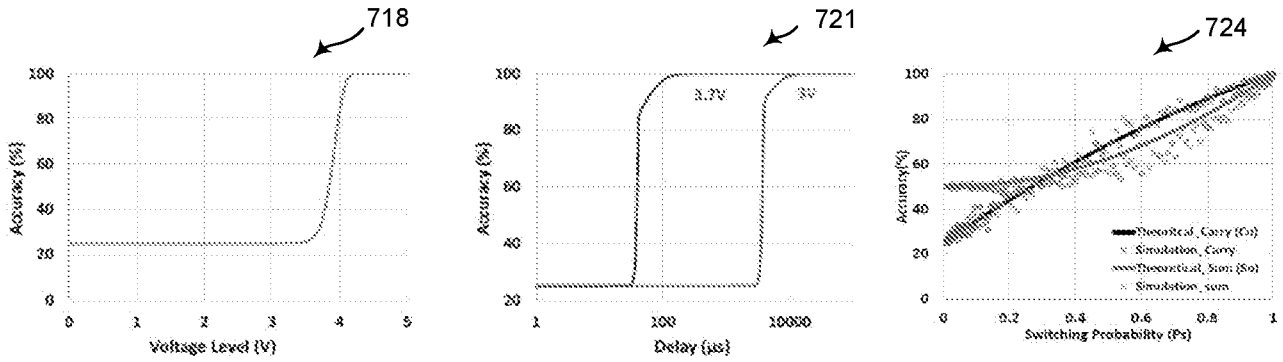


FIG. 7C

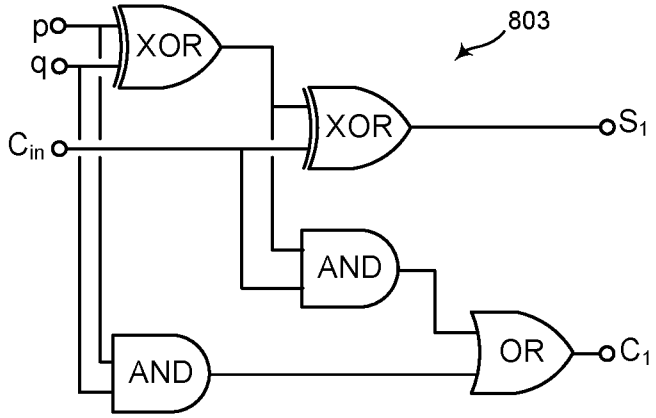


FIG. 8A

812

C_{in}	p	q	Sum (S_1)	Carry (C_1)
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

FIG. 8B

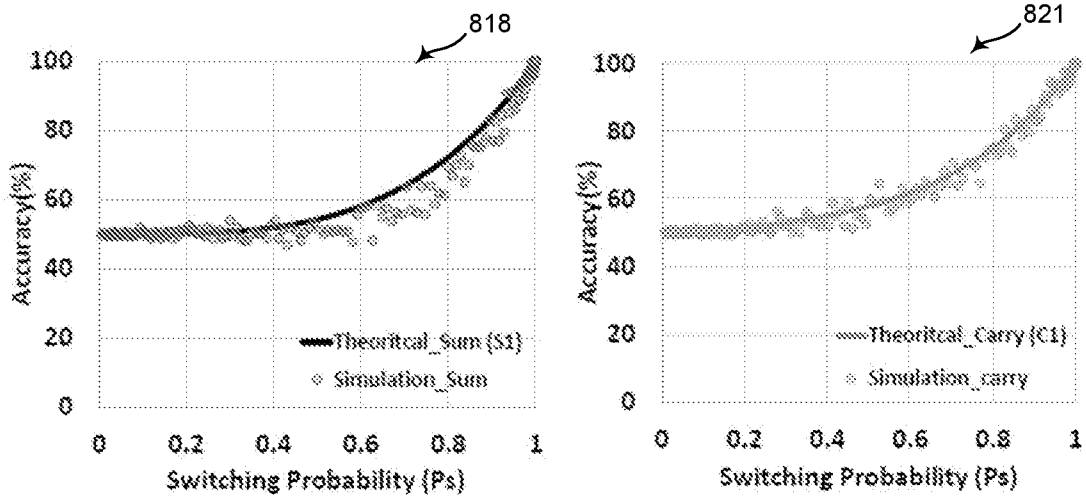


FIG. 8C

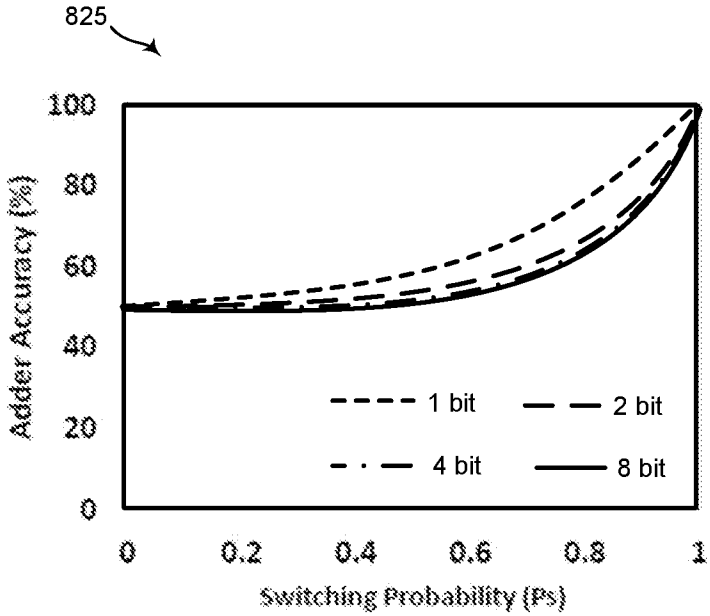

825
100
80
60
40
20
0
Adder Accuracy (%)
1 bit
2 bit
4 bit
8 bit
0
0.2
0.4
0.6
0.8
1
Switching Probability (Ps)

FIG. 8D

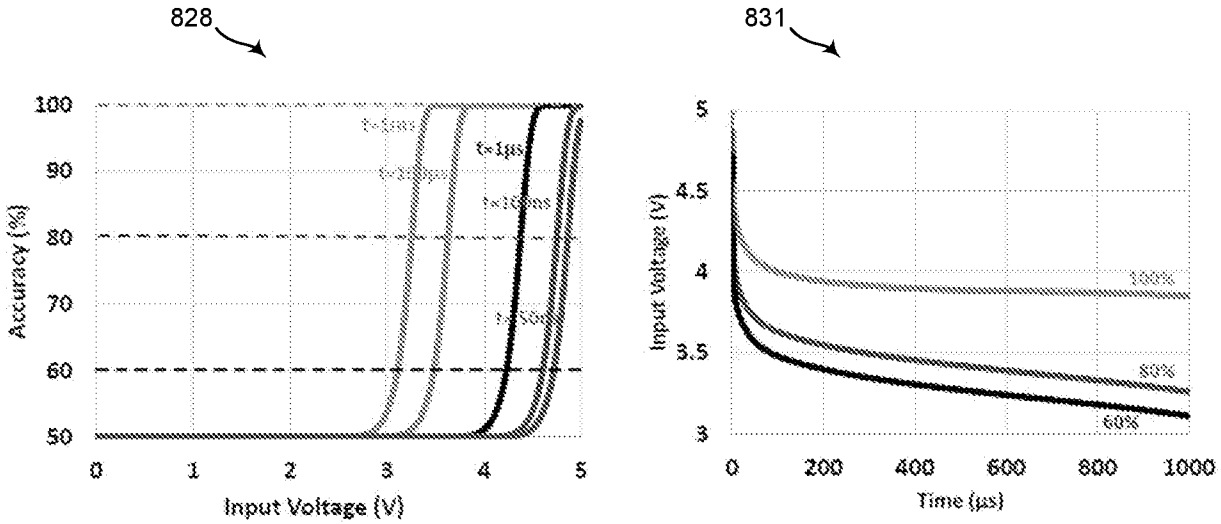

828
100
90
80
70
60
50
Accuracy (%)
0
1
2
3
4
5
Input Voltage (V)
831
5
4.5
4
3.5
3
Input Voltage (V)
100%
80%
60%
0
200
400
600
800
1000
Time (µs)

FIG. 8E

INTERNATIONAL SEARCH REPORT

International application No
PCT/IB2018/052524

A. CLASSIFICATION OF SUBJECT MATTER
INV. G11C13/00 G11C7/10 H03K19/20
ADD.

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
G11C H03K

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
EPO-Internal , WPI Data

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	ANNE SIEMON ET AL: "Realization of Boolean Logic Functionality Using Redox-Based Memristive Devices", ADVANCED FUNCTIONAL MATERIALS, vol. 25, no. 40, 1 October 2015 (2015-10-01), pages 6414-6423, XP055494332, DE ISSN: 1616-301X, DOI: 10.1002/adfm.201500865	1,4,5,7,, 10, 11. 13, 16
Y	the whole document	2,8, 14, 16-20
A		3,6,9, 12, 15

A	US 2010/220523 A1 (MODHA DHARMENDRA S [US] ET AL) 2 September 2010 (2010-09-02) paragraphs [0026] - [0032]	1-20

	-/-	

[X] Further documents are listed in the continuation of Box C. [X] See patent family annex.

* Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier application or patent but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) orwhich is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
27 July 2018	03/08/2018

Name and mailing address of the ISA/	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Fax: (+31-70) 340-3016	Czarik, Damien

2

Form PCT/ISA/210 (second sheet) (April 2005)

C(Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	LINN E ET AL: "Paper; Beyond von Neumann logi c operati ons i n passi ve crossbar arrays al ongsi de memory operations;Beyond von Neumann l ogi c operati ons i n passi ve crossbar arrays al ongside memory operations" , NANOTECHNOLOGY, IOP, BRISTOL, GB, vol . 23, no. 30, 10 July 2012 (2012-07-10) , page 3052Q5 , XP02Q227293, ISSN : 0957-4484, D0I : 10. 1088/0957-4484/23/30/305205 the whole document -----	2 ,8, 14, 16-20
A	YI LI ET AL: "Real i zation of Functi onal Compl ete Stateful Bool ean Logi c i n Memr i stive Crossbar" , ACS APPLI ED MATERIALS & INTERFACES, vol . 8 , no. 50, 21 December 2016 (2016-12-21) , pages 34559-34567, XP055494341 , US ISSN : 1944-8244, D0I : 10. 1021/acsami .6bll465 the whole document -----	1-20

2

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No
PCT/IB2018/052524

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2010220523 A1	02-09-2010	NONE	