SCIENCE CHINA
Information Sciences

• LETTER •

August 2021, Vol. 64 189401:1–189401:2
https://doi.org/10.1007/s11432-019-2938-8

Reconfigurable logic circuit design for stateful Boolean logic computing

Li LUO[1,4], Zhekang DONG[2], Xiaofang HU[3], Lidan WANG[1,4] & Shukai DUAN[3,4]*

[1]*College of Electronic and Information Engineering, Southwest University, Chongqing* 400715*, China;*
[2]*School of Electronics and Information, Hangzhou Dianzi University, Hangzhou* 310018*, China;*
[3]*College of Artificial Intelligence, Southwest University, Chongqing* 400715*, China;*
[4]*Brain-inspired Computing & Intelligent Control of Chongqing Key Lab, Chongqing* 400715*, China*

Received 26 December 2019/Revised 2 March 2020/Accepted 28 May 2020/Published online 19 March 2021

Citation Luo L, Dong Z K, Hu X F, et al. Reconfigurable logic circuit design for stateful Boolean logic computing. Sci China Inf Sci, 2021, 64(8): 189401, https://doi.org/10.1007/s11432-019-2938-8

Dear editor,

The conventional von Neumann computer system has the memory wall problem [1], which limits the computation speed and causes high energy and latency of the computing system. To realize computing systems able to process massive data and complex computing tasks with high efficiency, a key and viable approach is in-memory computing. Memristor-based stateful logic design is a promising candidate for realizing the in-memory logic computing [2].

There are considerable interests in memristor-based stateful Boolean logic in recent years. Material implication logic is achieved with memristors and resistors [3]. Separate memristors are required for the input and output of memristor-aided logic [4]. Four stateful two-memristor logic gates and five stateful three-memristor gates are introduced in [5]. The reconfigurable logic design is presented in [6], and the inputs of logic gates are denoted by four physical quantities, rather than just the nonvolatile resistance state.

This study proposes a novel reconfigurable circuit performing sixteen stateful Boolean logic operations. The input and output variables of the proposed logic gates are denoted by nonvolatile resistance states, indicating that the logic result is stored *in situ* and able to participate in subsequent operations, which is suitable for the logic cascading and conducive to the realization of complex computations.

Stateful NOR logic design. Figure 1(a) shows the circuit schematic for stateful logic gates, where P and Q are input memristors, M is the load memristor, Y is the output memristor, and S is the carbon nanotude filed-effect transistor. The resistance range of memristor is $[R_{\rm L}, R_{\rm H}]$, where $R_{\rm L} = 50\ \Omega$, $R_{\rm H} = 1000\ \Omega$. Operating voltages $V_{\rm P}$, $V_{\rm Q}$, $V_{\rm M}$, $V_{\rm Y}$, and $V_{\rm S}$ are applied to P, Q, M, Y, and S, respectively.

To operate the circuit as stateful logic gates, inputs p and q are the resistance states of P and Q, respectively. The output y is the final resistance state of Y. Before logic operations, P and Q are initialized to desired states $R_{\rm L}$ or $R_{\rm H}$, and M and Y are set to $R_{\rm L}$, where $R_{\rm L}$ and $R_{\rm H}$ are assigned to logic 1 and 0, respectively. When performing a logic operation, operating voltages are applied simultaneously, and NOR is realized by setting appropriate voltages, specifically, $V_{\rm P} = V_{\rm Q} = V_{\rm C}$, $V_{\rm M} = V_{\rm Y} =$ G (grounded), $V_{\rm S} = V_{\rm R}$.

The selection of $V_{\rm C}$ and $V_{\rm R}$ is critical to correctly perform logic operations. The value of $V_{\rm C}$ should be selected in a range that the voltage across each input memristor and the voltage across the load memristor are both between the threshold voltages $V_{\rm L}$ and $V_{\rm H}$, to support the logic operation while keeping unchanged of inputs during the logic operation. Meanwhile, the magnitude of $V_{\rm R}$ should be larger than $V_{\rm H}$ to switch Y to $R_{\rm H}$.

The voltage $V_{\rm S}$ is gated to Y, and the state of S is determined by the node voltage V_n at the node n. When the potential V_n of the node n exceeds $V_{\rm ST}$, S is turned on (equivalent to a closed switch). Therefore, the voltage drop across Y approximates to $V_{\rm R}$, and Y is switched to $R_{\rm H}$. If V_n is lower than $V_{\rm ST}$, S is turned off (equivalent to an open switch). As a result, the voltage drop across Y approximates to 0, keeping Y at $R_{\rm L}$. Based on Kirchhoff's law, we can get

$$\frac{V_{\rm P}-V_n}{R_{\rm P}}+\frac{V_{\rm Q}-V_n}{R_{\rm Q}}+\frac{V_{\rm M}-V_n}{R_{\rm M}}=0, \qquad (1)$$

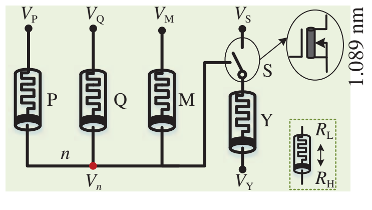
where $R_{\rm P}$, $R_{\rm Q}$, and $R_{\rm M}$ are the resistances of P, Q, and M, respectively.

Based on (1), $V_{\rm P} = V_{\rm Q} = V_{\rm C}$, and $V_{\rm M} =$ G, V_n can be calculated by

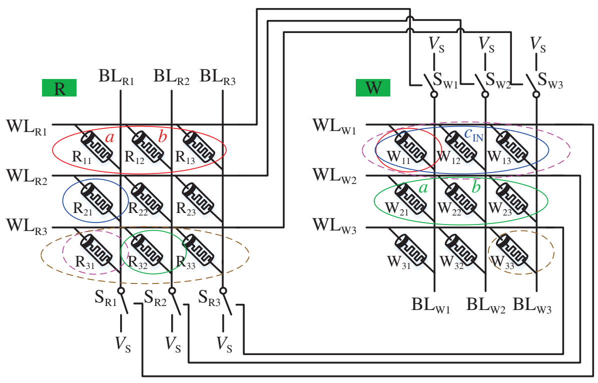
$$V_n=\frac{R_{\rm M}(R_{\rm P}+R_{\rm Q})}{(R_{\rm P}+R_{\rm Q})\cdot R_{\rm M}+R_{\rm P}\cdot R_{\rm Q}}\cdot V_{\rm C}. \qquad (2)$$

According to the specific states of two-input Boolean logic and (2), node voltages are $0.09V_{\rm C}$, $0.51V_{\rm C}$, $0.51V_{\rm C}$, and $0.67V_{\rm C}$, respectively, for input combinations '00', '01', '10', and '11'. To perform NOR, the threshold voltage $V_{\rm ST}$ of S should satisfy $0.09V_{\rm C} < V_{\rm ST} < 0.51V_{\rm C}$. Here, we chose $V_{\rm ST}$ to be $0.4V_{\rm C}$. In this way, when inputs are both logic 0, the magnitude of V_n is less than $V_{\rm ST}$, and then S remains open, which keeps the logic state of Y unchanged. For all other inputs, the magnitude of V_n is greater than $V_{\rm ST}$, thus S is closed, making the logic state of Y to be logic 0.

*Corresponding author (email: duansk@swu.edu.cn)



(a)



(b)

Figure 1 (Color online) (a) Schematic circuit diagram for stateful logic operations; (b) full adder.

Remaining stateful Boolean logic. By tuning operating voltages V_P, V_Q, and V_M, and fixing other biases V_S (V_R) and V_Y (G), sixteen Boolean operations can be realized. That is, by simply changing the voltages applied to P, Q, and M, the same circuit is able to execute different logic operations. For instance, NAND can be realized by setting $V_P = V_Q = 0.7V_C$, $V_M = V_Y =$ G, and $V_S = V_R$. From (1), the node voltage V_n of NAND gate is

$$V_n = \frac{0.7R_M(R_P + R_Q)}{(R_P + R_Q)\cdot R_M + R_P \cdot R_Q}\cdot V_C. \tag{3}$$

Based on (3), the node voltages are $0.06V_C$, $0.35V_C$, $0.35V_C$, and $0.46V_C$, respectively, for inputs '00', '01', '10', and '11'. Because of $V_{ST} = 0.4V_C$, when inputs are both logic 1, the node voltage V_n exceeds V_{ST}, and then S is closed, which switches the logic state of Y from logic 1 to 0. For all other input combinations, V_n is suppressed below V_{ST}, thus S remains open, keeping the logic state of Y to be logic 1. As a result, this configuration is capable of performing NAND. The logic function reconfiguration of the circuit for NOR and NAND operations is illustrated in Appendix A.

For complete 16 Boolean logic operations, the specific applied voltage assignment, the node voltage formula, and the corresponding node voltage and output under different input combinations are summarized in Appendix B. Furthermore, the impact of resistance variation on the logic operation is discussed in Appendix C.

Full adder. The one-bit full adder implemented by using the bidirectional crossbar arrays is shown in Figure 1(b), where the memristors in the same color ellipses can be used as a stateful logic gate. The one-bit full adder contains three inputs (i.e., addend a, summand b, and carry-in c_{IN}) and two outputs (summary d and carry-out c_O). The logic functions of d and c_O can be expressed as $d = a \oplus b \oplus c_{IN}$ and $c_O = (a \oplus b)\cdot c_{IN} + a\cdot b$, respectively, where $\oplus$, $+$, and $\cdot$ denotes XOR, OR, and AND logic operations, respectively.

To correctly perform logic operations, when the crossbar array is used for input branches, selected and unselected rows are floated and grounded, respectively. While the crossbar array is used for output branches, selected and unselected rows are grounded and floated, respectively. Before executing logic operation, the input a is stored into R_{11} and W_{21}, the input b is stored in R_{12} and W_{22}, the input c_{IN} is stored in W_{12}, and other memristors are set to logic 1.

In the first two steps, the XOR between a and b is realized, and the logic output (i.e., $a \oplus b$) is stored in W_{11}. In next two steps, the logic operation $d = a \oplus b \oplus c_{IN}$ is executed, and the result is stored into R_{21}. So after step 4, the sum output d is obtained. In step 5, the AND between $a \oplus b$ and c_{IN} is executed, whose result is stored in R_{31}. In Step 6, AND is performed, and data in R_{32} is changed to be $a \cdot b$. In the final step, c_O can be gained by realizing the OR between the logic values in R_{31} and R_{32}, that is, $c_O = (a \oplus b)\cdot c_{IN} + a\cdot b$ is gained and stored in W_{33}. As a result, the sum and carry outputs of a full adder are obtained with seven logic operation steps. The logic operation steps and corresponding applied voltage levels for half adder and full adder is listed in Appendix D. In addition, simulation results are shown in Appendix E.

Conclusion. A reconfigurable stateful logic design is presented, which is able to perform complete 16 Boolean logic operations with the same circuit topology, different than other memristive stateful logic designs. Then the full adder are realized by using the presented stateful logic gates. The entire scheme opens up a new approach for the fusion of computation and memory to develop beyond von Neumann computer architectures.

Acknowledgements This work was supported by National Key Research and Development Program of China (Grant Nos. 2018YFB1306600, 2018YFB1306604), National Natural Science Foundation of China (Grant Nos. 61672436, 61601376, 61976246), and Fundamental Research Funds for the Provincial Universities (Grant No. GK199900299012-010).

Supporting information Appendixes A–E. The supporting information is available online at info.scichina.com and link.springer.com. The supporting materials are published as submitted, without typesetting or editing. The responsibility for scientific accuracy and content remains entirely with the authors.

References

1 Borkar S, Chien A A. The future of microprocessors. Commun ACM, 2011, 54: 67–77
2 Li Y, Zhou Y X, Wang Z R, et al. Memcomputing: fusion of memory and computing. Sci China Inf Sci, 2018, 61: 060424
3 Borghetti J, Snider G S, Kuekes P J, et al. 'Memristive' switches enable 'stateful' logic operations via material implication. Nature, 2010, 464: 873–876
4 Kvatinsky S, Belousov D, Liman S, et al. MAGIC — memristor-aided logic. IEEE Trans Circ Syst II, 2014, 61: 895–899
5 Kim K M, Williams R S. A family of stateful memristor gates for complete cascading logic. IEEE Trans Circ Syst I, 2019, 66: 4348–4355
6 Hu S Y, Li Y, Cheng L, et al. Reconfigurable Boolean logic in memristive crossbar: the principle and implementation. IEEE Electron Dev Lett, 2019, 40: 200–203