SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

October 2020, Vol. 63 202401:1–202401:8
https://doi.org/10.1007/s11432-020-2866-0

Efficient 16 Boolean logic and arithmetic based on bipolar oxide memristors

Rui YUAN[1], Mingyuan MA[2], Liying XU[1], Zhenhua ZHU[2], Qingxi DUAN[1], Teng ZHANG[1], Yu ZHU[2], Yu WANG[2], Ru HUANG[1,3,4*] & Yuchao YANG[1,3,4*]

[1]*Key Laboratory of Microelectronic Devices and Circuits (MOE), Department of Micro/Nanoelectronics, Peking University, Beijing* 100871*, China;*
[2]*Department of Electronic Engineering, Beijing National Research Center for Information Science and Technology, Tsinghua University, Beijing* 100084*, China;*
[3]*Center for Brain Inspired Chips, Academy for Artificial Intelligence, Peking University, Beijing* 100871*, China;*
[4]*Frontiers Science Center for Nano-optoelectronics, Peking University, Beijing* 100871*, China*

Received 15 March 2020/Revised 25 March 2020/Accepted 8 April 2020/Published online 26 August 2020

Abstract The physically separated memory and logic units in traditional von Neumann computers place essential limits on the performance and cause increased energy consumption, and hence in-memory computing is required to overcome this bottleneck. Here, a new bipolar memristor based in-memory logic approach is proposed, which is capable of achieving all 16 possible Boolean logic functions in a single device in less than 3 steps. This approach does not require initialization and can facilitate logic cascading, and the calculation taking place in-situ is showcased by 1-bit full adder and 2-bit multiplier with improved efficiency, thus showing a great prospect in future in-memory computing architecture.

Keywords memristor, nonvolatile logic, in-memory computing, full adder, multiplier

Citation Yuan R, Ma M Y, Xu L Y, et al. Efficient 16 Boolean logic and arithmetic based on bipolar oxide memristors. Sci China Inf Sci, 2020, 63(10): 202401, https://doi.org/10.1007/s11432-020-2866-0

1 Introduction

As the scaling of CMOS devices approaches their physical limits, Moore's law is approaching its end [1]. The development of future computing technology requires innovative materials, devices, and architectures [2–5]. While the performance of traditional computers is strongly limited by the von Neumann bottleneck, in-memory logic can fundamentally address the overhead caused by data movements between logic and memory and thus achieve high efficiency [6]. Recently, memristor based in-memory computing has dawn extensive attention [7–10], due to its high speed, high scalability, and low energy consumption [11–15]. Memristor is a simple metal-insulator-metal sandwich structure, and their resistances can be switched between two digital states, low resistance state and high resistance state [16–18], laying the foundation for logical operations. Memristors have demonstrated rich logic and arithmetic functionalities in recent years [8, 19–34]. Memristor based nonvolatile logic can usually be divided into two classes-stateful and non-stateful logic, according to the physical quantities of input and output variables. For stateful logic, such as IMPLY [7] and MAGIC [29], both input and output variables are represented by the same physical quantity, i.e., resistance state. However, for non-stateful logic [8, 10], the voltage signals applied on the devices are usually adopted as inputs, while the resistance state usually serves as the output, hence featuring different physical quantities. In consideration of the input variables p and

* Corresponding author (email: ruhuang@pku.edu.cn, yuchaoyang@pku.edu.cn)

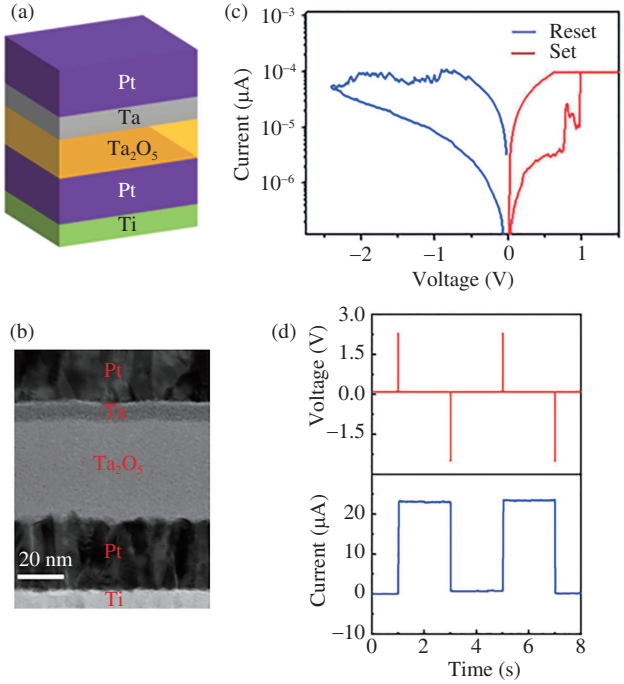


Figure 1 (Color online) (a) Schematic of the Pt/Ta/Ta_2O_5/Pt/Ti device structure. (b) TEM image of the Pt/Ta/Ta_2O_5/Pt/Ti structure. Scale bar: 20 nm. (c) Typical resistive switching characteristics of the Pt/Ta/Ta_2O_5/Pt/Ti devices. (d) Current response under voltage pulse. Constant read voltage of 0.1 V was applied to the TE terminal.

q, both inputs are represented by the same physical quantity, namely, resistance state and voltage for stateful and non-stateful logic, respectively, while the output variable is always defined as the resistance state. Detailed logic definitions and approaches thus decide the overall performance in terms of area, latency, and logic cascading [7, 8, 10, 29].

In this study, we propose and demonstrate an efficient logic approach based on bipolar oxide memristors, where the two input variables are represented by different physical quantities, that is, the device resistance and the voltage applied onto one terminal of the memristor. This allows to implement all 16 Boolean logic functions using 1 device in less than 3 steps. Further experimental demonstrations on 1-bit full adder and 2-bit multiplier show high efficiency in area and latency compared with existing approaches, indicating the potential of this method in next-generation in-memory computing.

2 Realization of Boolean logic

The structure of the memristor used in this study includes Pt/Ta as the top electrode, Ta_2O_5 as the switch layer, and Pt/Ti as the bottom electrode, as schematically illustrated in Figure 1(a). Figure 1(b) shows transmission electron microscopy (TEM) image of the device, where the stacking structure can be clearly observed. In this study, a Ta_2O_5 thickness of 30 nm was adopted, and the Pt/Ta/Ta_2O_5/Pt/Ti device showed bipolar switching, as shown by the current-voltage (I-V) characteristics of the Pt/Ta/Ta_2O_5/Pt/Ti device in Figure 1(c). Figure 1(d) further shows the current response of the device under voltage pulses with 100 ns width, where a +2.3 V voltage pulse is applied to switch device from high resistance state (HRS) to low resistance state (LRS) and a −2.5V voltage can switch it from LRS back to HRS. We can see that the Pt/Ta/Ta_2O_5/Pt/Ti device has two stable resistance states, which is crucial for memristor based nonvolatile logic operations.

We now turn to the implementation of in-memory logic using the above Pt/Ta/Ta_2O_5/Pt/Ti devices. Here, we only use 1 single cell as the logic gate. The initial resistance state and the voltage applied on one electrode represent the inputs p and q. The output is represented by final resistance state, and the voltage applied on the other electrode depends on the logic function to be performed (Figure 2(a)).

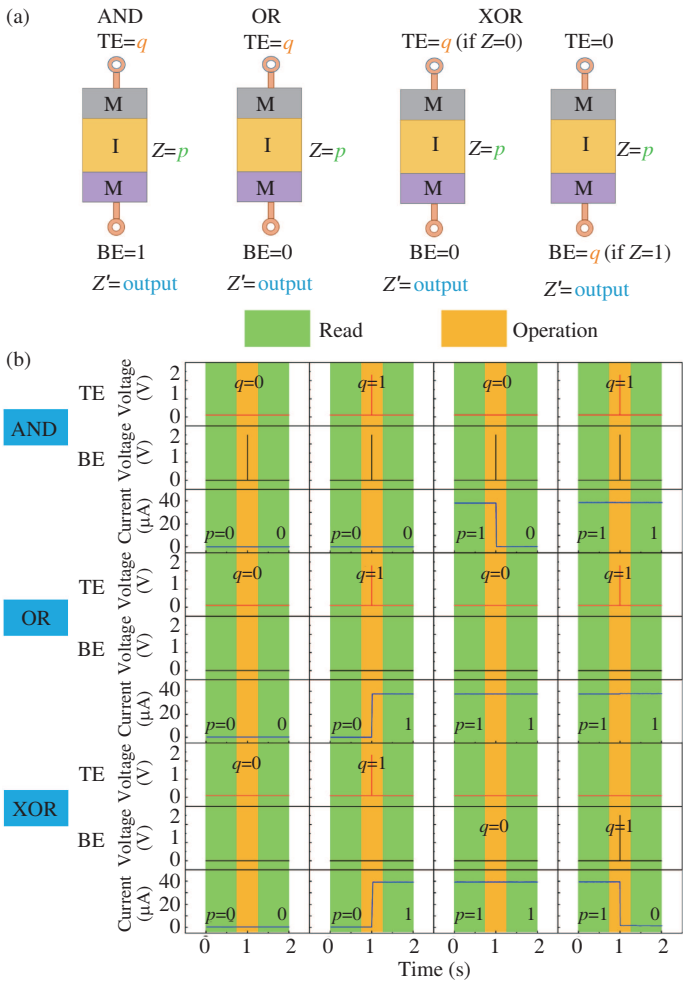


Figure 2 (Color online) (a) Schematic diagram of implementing three logical operations. (b) Experimental demonstration of AND, OR, and XOR in pulse measurements. The initial read operation (green background) is to know the initial resistance state and verify the correctness of the logic operation. In practical applications, only XOR logic needs to read the initial resistance state to determine the voltage input port. Constant read voltage of 0.1 V was applied to the TE terminal during the entire logic operations. $V_{\rm set}$ added to the TE is 2.3 V, 100 ns and $V_{\rm reset}$ added to the BE is 2.5 V, 100 ns.

High voltage and LRS represent logic "1", and vice versa. As examples, we experimentally demonstrate three logic functions AND, OR, and XOR in Figure 2, and all 16 Boolean logic functions can be achieved following the operations in Table 1.

Figure 2(a) shows the schematic diagram of implementing three logic functions. To implement AND logic function, initial resistance state serves as input p, and the voltage applied to top electrode (TE) is defined as input q. The output is represented by final resistance state Z', while bottom electrode (BE) is placed at high voltage. Only when $p = 1$ and $q = 1$, the final resistance state keeps in LRS which satisfies the equation $Z' = ZT_1$. Similarly, OR logic function can be implemented by grounding the BE. To realize XOR function, firstly we need to read the initial resistance state as a selection signal. If $Z = 0$, the voltage applied to TE is chosen as input q, and BE is grounded. On the contrary, the voltage applied to BE represents input q and TE is grounded, when $Z = 1$. The experimental results are shown in the Figure 2(b), demonstrating the correctness of this logical approach. As a result, realizing AND and OR functions only needs 1 write operation, while XOR logic function needs 1 read and 1 write operations in a single device, which is more efficient compared with existing bipolar memristor based logic approaches [35, 36]. Table 1 shows the implementation method of all Boolean logic. It is worth pointing out that the implementation of the logic function in bold font needs to read out the resistance state as a selection signal, and the additional circuit overhead for this needs to be optimized.

Table 1 Complete 16 Boolean logic implementations

Logic	Input		Cycle 1			Cycle 2			Output	Logic	Input		Cycle 1			Cycle 2			Output
	p	q	Z	TE	BE	Z_m	TE	BE	Z'		p	q	Z	TE	BE	Z_m	TE	BE	Z'
TURE	0	0		1	0				1	**XOR**	0	0	p	q	0				0
	0	1		1	0				1		0	1	p	q	0				1
	1	0		1	0				1		1	0	p	0	q				1
	1	1		1	0				1		1	1	p	0	q				0
FALSE	0	0		0	1				0	**XNOR**	0	0	p	1	q				1
	0	1		0	1				0		0	1	p	1	q				0
	1	0		0	1				0		1	0	p	q	1				0
	1	1		0	1				0		1	1	p	q	1				1
p	0	0	p						0	**NAND**	0	0	p	q	1	0	1	0	1
	0	1	p						0		0	1	p	q	1	0	1	0	1
	1	0	p						1		1	0	p	q	1	0	1	0	1
	1	1	p						1		1	1	p	q	1	1	0	1	0
q	0	0		0	1	0	q	0	0	**NOR**	0	0	p	q	0	0	1	0	1
	0	1		0	1	0	q	0	1		0	1	p	q	0	1	0	1	0
	1	0		0	1	0	q	0	0		1	0	p	q	0	1	0	1	0
	1	1		0	1	0	q	0	1		1	1	p	q	0	1	0	1	0
NOT p	0	0	p	1	0				1	RIMP	0	0	p	1	q				1
	0	1	p	1	0				1		0	1	p	1	q				0
	1	0	p	0	1				0		1	0	p	1	q				1
	1	1	p	0	1				0		1	1	p	1	q				1
NOT q	0	0	p	1	q				1	IMP	0	0	q	1	p				1
	0	1	p	1	q				0		0	1	q	1	p				1
	1	0	p	0	q				1		1	0	q	1	p				0
	1	1	p	0	q				0		1	1	q	1	p				1
AND	0	0	p	q	1				0	NIMP	0	0	p	0	q				0
	0	1	p	q	1				0		0	1	p	0	q				0
	1	0	p	q	1				0		1	0	p	0	q				1
	1	1	p	q	1				1		1	1	p	0	q				0
OR	0	0	p	q	0				0	RNIMP	0	0	q	0	p				0
	0	1	p	q	0				1		0	1	q	0	p				1
	1	0	p	q	0				1		1	0	q	0	p				0
	1	1	p	q	0				1		1	1	q	0	p				0

3 Realization of full adders and multipliers

Full adders and multipliers play important roles in modern computer systems and are the basis of computing units. Here a 1-bit full adder and a 2-bit multiplier are experimentally demonstrated. There are three inputs (addend A, summand B, and carry-in C_i) and two outputs (summary S and carry-out C_o) for 1-bit full adder. The logic functions can be expressed as

$$S = A \oplus B \oplus C_i, \tag{1}$$

$$C_o = (A \oplus B)C_i + AB. \tag{2}$$

Figure 3(a) shows the operating sequence of the 1-bit binary full adder, where only 3 devices and up to 6 steps are required. The first device calculates the C_o and the S is calculated by the second device. The third device calculates intermediate values. It is worthwhile pointing out that the resistance state needs to be read out when XOR logic is done in steps 2–5. The first step is the writing of data A, and the remaining steps are logical calculations. The calculation process diagram for the typical input combination (1+1+1) is shown in Figure 3(b). Since the result of the calculation is a resistance state,

(a)

Operating sequence	Step \ Device	1	2	3
A	1	A	A	A
Read A	2	Read A	Read A	–
$A\oplus B$, AB	3	$A\oplus B$	$A\oplus B$	AB
Read $(A\oplus B)$	4	–	Read $A\oplus B$	–
$(A\oplus B)C_i$, $A\oplus B\oplus C_i$, Read AB	5	$(A\oplus B)C_i$	$A\oplus B\oplus C_i$	Read AB
$(A\oplus B)C_i$+AB	6	$(A\oplus B)C_i$+AB	–	–

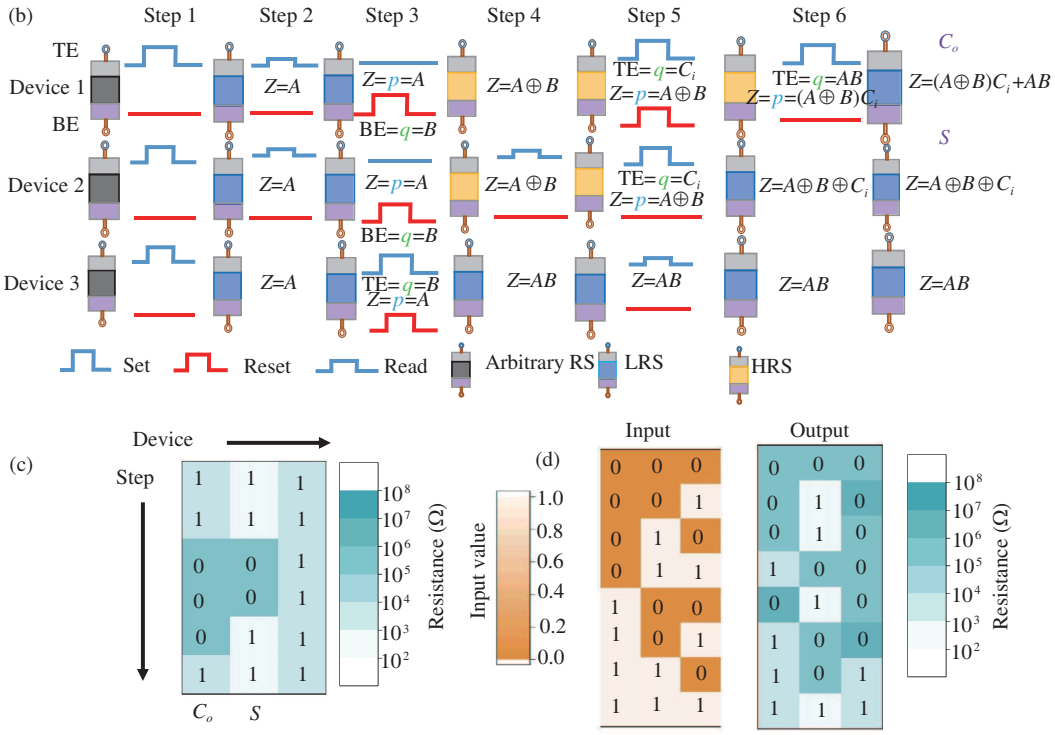


Figure 3 (Color online) (a) Operating sequence of the 1-bit binary full adder; (b) calculation process diagram for the typical input combination (1+1+1); (c) resistance evolution of three devices for input combination (1+1+1); (d) experimental results for all 8 possible input combinations.

Table 2 Comparison of XOR logic operation and 1-bit full adder schemes

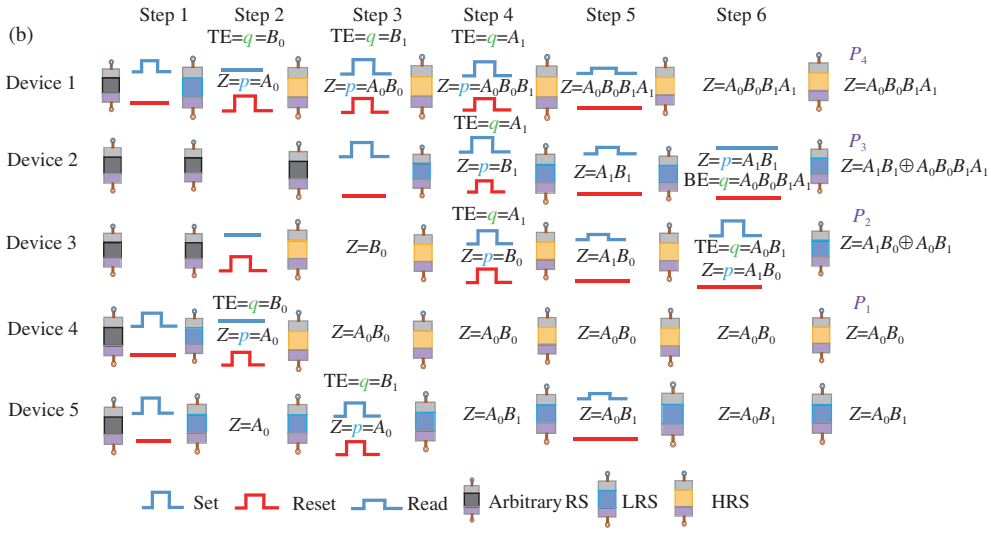
Reference	XOR		1-bit full adder	
	Device number	Step	Device number	Step
[7, 35]	4	6	8	27
[36]	5	13	6	29
[37]	–	–	6	35
[27]	4	4	9	10
[28]	–	–	16	10
[10]	1	2	5	8
This study	1	2	3	6

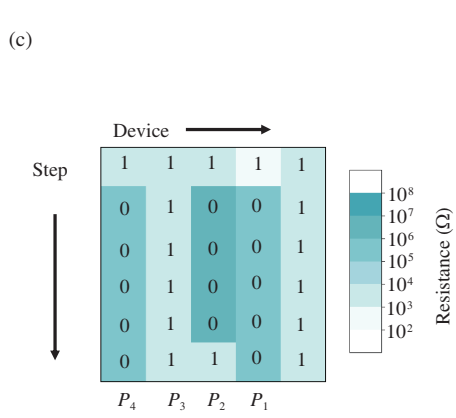
it can be directly used as the next input, reducing the conversion of voltage and resistance which will help reduce the number of calculation steps. Another reason for the reduction in calculation steps is the efficient implementation of XOR logic. Table 2 summarizes the number of devices and steps required to implement XOR and one-bit full adders for different logic schemes. The state evolutions of all devices for input combination (1+1+1) are shown in Figure 3(c). Figure 3(d) shows the experimental results for all 8 input combinations, where correct results have been successfully obtained in all cases.

Furthermore, a 2-bit multiplier is experimentally demonstrated using only 5 devices and up to 6 steps.

(a)

Operating sequence	Step \ Device	1	2	3	4	5
A_0	1	A_0	–	–	A_0	A_0
A_0B_0, B_0	2	A_0B_0	–	B_0	A_0B_0	–
$A_0B_0B_1$, B_1, A_0B_1	3	$A_0B_0B_1$	B_1	–	–	A_0B_1
$A_0B_0A_1B_1$, A_1B_1, A_1B_0	4	$A_0B_0A_1B_1$	A_1B_1	A_1B_0	–	–
Read $A_0B_0A_1B_1$, Read A_1B_1, Read A_1B_0, ReadA_0B_1	5	Read $A_0B_0A_1B_1$	Read A_1B_1	Read A_1B_0	–	Read A_0B_1
$A_1B_1\oplus A_0B_0A_1B_1$, $A_1B_0\oplus A_0B_1$	6	–	$A_1B_1\oplus A_0B_0$ A_1B_1	$A_1B_0\oplus A_0B_1$	–	–





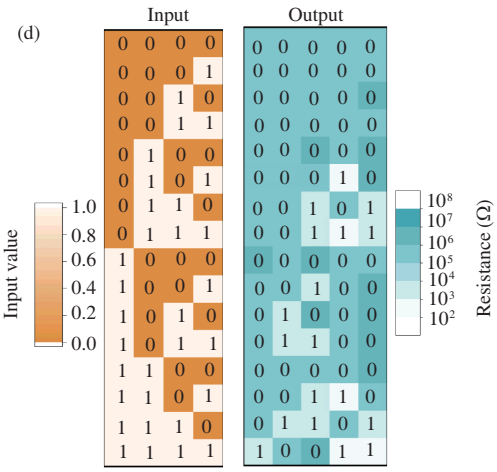


Figure 4 (Color online) (a) Operating sequence of the 2-bit multiplier; (b) calculation process diagram for the typical input combination (11×10); (c) resistance evolution of five devices for input combination (11×10); (d) experimental results for all 16 possible input combinations.

A_1A_0, B_1B_0 are the two inputs of the multiplier, and $P_4P_3P_2P_1$ are the 4-bit output of the 2-bit multiplier, which can be expressed as

$$P1 = A_0B_0, \tag{3}$$

$$P2 = A_1B_0 \oplus A_0B_1, \tag{4}$$

$$P3 = A_1B_1 \oplus A_0B_0A_1B_1, \tag{5}$$

$$P4 = A_0B_0A_1B_1. \tag{6}$$

The operating sequence of the 2-bit multiplier is showed in Figure 4(a), where the output $P_4P_3P_2P_1$ are calculated by the first 4 devices. The resistance state needs to be read out when XOR logic is done in steps 5 and 6. The entire process is divided into data writing, reading, and performing logical operations. Calculation process diagram for the typical input combination (11×10) and the corresponding resistance evolution is shown in Figure 4(b) and (c). Figure 3(d) further shows the experimental results for all 16 possible input combinations.

To the best of our knowledge, this is the first experimental implementation of 2-bit multiplier using memristors, and the devices and steps required to realize full adder and multiplier in this study are more efficient compared with previous memristor logic approaches [10, 27, 28, 35–37]. The in-situ calculation overwrites the input, deciding that less devices are needed. Moreover, this approach does not require initialization and can facilitate logic cascading, because the output is a resistance state, it can be used as the input of the next stage, reducing the number of voltage and resistance conversions. This logic approach is also compatible with the current mainstream nonvolatile memory structure 1T1R [38], showing a great prospect in future in-memory computing.

4 Conclusion

In this study, we have proposed an efficient in-memory logic approach which is capable of implementing all 16 Boolean logic functions in 1 single cell without initialization. The improved efficiency of the proposed approach is experimentally showcased by 1-bit full adder and 2-bit multiplier. This approach is also compatible with the current mainstream nonvolatile memory structure 1T1R, paving a new way for future in-memory computing.

Acknowledgements This work was supported by National Key R&D Program of China (Grant No. 2017YFA0207600), National Natural Science Foundation of China (Grant Nos. 61925401, 61674006, 61927901, 61421005), and the 111 Project (Grant No. B18001). Yuchao YANG acknowledges the support from Beijing Academy of Artificial Intelligence (BAAI) and the Tencent Foundation through the Xplorer Prize.

References

1 Waldrop M M. The chips are down for Moore's law. Nature, 2016, 530: 144–147
2 Ventra M D, Pershin Y V. The parallel approach. Nat Phys, 2013, 9: 200–202
3 Cassinerio M, Ciocchini N, Ielmini D. Logic computation in phase change materials by threshold and memory switching. Adv Mater, 2013, 25: 5975–5980
4 Merolla P A, Arthur J V, Alvarez-Icaza R, et al. A million spiking-neuron integrated circuit with a scalable communication network and interface. Science, 2014, 345: 668–673
5 Prezioso M, Merrikh-Bayat F, Hoskins B D, et al. Training and operation of an integrated neuromorphic network based on metal-oxide memristors. Nature, 2015, 521: 61–64
6 Li S C, Xu C, Zou Q S, et al. Pinatubo: a processing-in-memory architecture for bulk bitwise operations in emerging non-volatile memories. In: Proceedings of the 53rd Annual Design Automation Conference, Austin Texas, 2016
7 Borghetti J, Snider G S, Kuekes P J, et al. 'Memristive' switches enable 'stateful' logic operations via material implication. Nature, 2010, 464: 873–876
8 Linn E, Rosezin R, Tappertzhofen S, et al. Beyond von Neumann-logic operations in passive crossbar arrays alongside memory operations. Nanotechnology, 2012, 23: 305205
9 Yang Y C, Yin M H, Yu Z Z, et al. Multifunctional nanoionic devices enabling simultaneous heterosynaptic plasticity and efficient in-memory Boolean logic. Adv Electron Mater, 2017, 3: 1700032
10 Xu L Y, Yuan R, Zhu Z H, et al. Memristor-based efficient in-memory logic for cryptologic and arithmetic applications. Adv Mater Technol, 2019, 4: 1900212
11 Goswami S, Matula A J, Rath S P, et al. Robust resistive memory devices using solution-processable metal-coordinated azo aromatics. Nat Mater, 2017, 16: 1216–1224
12 Wong H S P, Salahuddin S. Memory leads the way to better computing. Nat Nanotech, 2015, 10: 191–194
13 Hwang C S. Prospective of semiconductor memory devices: from memory system to materials. Adv Electron Mater, 2015, 1: 1400056
14 Ma W, Zidan M A, Lu W D. Neuromorphic computing with memristive devices. Sci China Inf Sci, 2018, 61: 060422
15 Yang Y C, Huang R. Probing memristive switching in nanoionic devices. Nat Electron, 2018, 1: 274–287
16 Yang J J, Pickett M D, Li X, et al. Memristive switching mechanism for metal/oxide/metal nanodevices. Nat Nanotech, 2008, 3: 429–433

17 Kim K M, Jeong D S, Hwang C S, et al. Nanofilamentary resistive switching in binary oxide system; a review on the present status and outlook. Nanotechnology, 2011, 22: 254002
18 Choi B J, Jeong D S, Kim S K, et al. Resistive switching mechanism of TiO_2 thin films grown by atomic-layer deposition. J Appl Phys, 2005, 98: 033715
19 Ielmini D, Wong H S P. In-memory computing with resistive switching devices. Nat Electron, 2018, 1: 333–343
20 Vourkas I, Sirakoulis G C. Emerging memristor-based logic circuit design approaches: a review. IEEE Circ Syst Mag, 2016, 16: 15–30
21 Li H T, Gao B, Chen Z W, et al. A learnable parallel processing architecture towards unity of memory and computing. Sci Rep, 2015, 5: 13330
22 Serb A, Khiat A, Prodromakis T. Seamlessly fused digital-analogue reconfigurable computing using memristors. Nat Commun, 2018, 9: 2170
23 You T G, Shuai Y, Luo W B, et al. Exploiting memristive $BiFeO_3$ bilayer structures for compact sequential logics. Adv Funct Mater, 2014, 24: 3357–3365
24 Breuer T, Siemon A, Linn E, et al. A HfO_2-based complementary switching crossbar adder. Adv Electron Mater, 2015, 1: 1500138
25 Siemon A, Breuer T, Aslam N, et al. Realization of Boolean logic functionality using redox-based memristive devices. Adv Funct Mater, 2015, 25: 6414–6423
26 Gao S, Zeng F, Wang M J, et al. Implementation of complete Boolean logic functions in single complementary resistive switch. Sci Rep, 2015, 5: 15467
27 Huang P, Kang J F, Zhao Y D, et al. Reconfigurable nonvolatile logic operations in resistance switching crossbar array for large-scale circuits. Adv Mater, 2016, 28: 9758–9764
28 Chen B, Cai F X, Zhou J T, et al. Efficient in-memory computing architecture based on crossbar arrays. In: Proceedings of IEEE International Electron Devices Meeting (IEDM), Washington, 2015
29 Kvatinsky S, Belousov D, Liman S, et al. MAGIC-memristor-aided logic. IEEE Trans Circ Syst II, 2014, 61: 895–899
30 Shen W S, Huang P, Fan M Q, et al. Stateful logic operations in one-transistor-one- resistor resistive random access memory array. IEEE Electron Device Lett, 2019, 40: 1538–1541
31 Liu G Z, Zheng L J, Wang G Y, et al. A carry lookahead adder based on hybrid CMOS-memristor logic circuit. IEEE Access, 2019, 7: 43691–43696
32 Xu N, Park T G, Kim H J, et al. A stateful logic family based on a new logic primitive circuit composed of two antiparallel bipolar memristors. Adv Intell Syst, 2020, 2: 1900082
33 Yang H, Duan S K, Dong Z K, et al. A memristor-CMOS-based general-logic circuit and its applications (in Chinese). Sci Sin Inform, 2020, 50: 289–302
34 Wald N, Kvatinsky S. Understanding the influence of device, circuit and environmental variations on real processing in memristive memory using memristor aided logic. MicroElectron J, 2019, 86: 22–33
35 Cheng L, Zhang M Y, Li Y, et al. Reprogrammable logic in memristive crossbar for in-memory computing. J Phys D-Appl Phys, 2017, 50: 505102
36 Kvatinsky S, Satat G, Wald N, et al. Memristor-based material implication (IMPLY) logic: design principles and methodologies. IEEE Trans VLSI Syst, 2014, 22: 2054–2066
37 Adam G C, Hoskins B D, Prezioso M, et al. Optimized stateful material implication logic for three-dimensional data manipulation. Nano Res, 2016, 9: 3914–3923
38 Chen W H, Lin W J, Lai L Y, et al. A 16 Mb dual-mode ReRAM macro with sub-14 ns computing-in-memory and memory functions enabled by self-write termination scheme. In: Proceedings of IEEE International Electron Devices Meeting (IEDM), San Francisco, 2017