Spintronic Processing Unit in Spin Transfer Torque Magnetic Random Access Memory

He Zhang, Wang Kang, *Member, IEEE*, Kaihua Cao, Bi Wu, Youguang Zhang, *Member, IEEE*, and Weisheng Zhao, *Fellow, IEEE*

***Abstract*— Recently, exploiting emerging nonvolatile memories to implement the process-in-memory (PIM) paradigm have shown great potential to address the von Neumann bottleneck and have attracted extensive research and development. In this paper, we present a novel PIM platform—spintronic processing unit (SPU), within spin transfer toque magnetic random access memory (STT-MRAM). This energy-efficient and reconfigurable PIM platform can perform different tasks—data storage and logic computing—using the same physical fabric that is programmable at the finest grain, i.e., the individual memory cell level, without the need to move data outside the memory fabric. The proposed SPU works just like a typical memory and all the logic functions are achieved through regular memory-like write and read operations with minimal modifications. The functionality and performance are evaluated via hybrid circuit simulations under the 40-nm process technology node. Our proposed SPU is expected to be a feasible PIM platform in the near future, owing to the increasing maturity of STT-MRAM.**

***Index Terms*— Process-in-memory (PIM), spin transfer toque magnetic random access memory (STT-MRAM), spintronic processing unit (SPU), von Neumann bottleneck.**

I. Introduction

OVER the past decades, there is an explosive growth in data size, especially in current big data and machine learning applications. However, regarding conventional von Neumann architecture, the overhead of the data communication between the processor and the memory units results in huge performance degradation and energy consumption, called von Neumann bottleneck [1], [2]. In order to overcome this bottleneck, a hopeful approach is to embed processing capability into the memory, termed processing-in-memory (PIM) [3]–[6], which has attracted considerable interests. Nevertheless, in the past years, such promising technology could not render satisfactory product, owing to the design and fabrication complexity.

Recently, utilizing emerging nonvolatile memories (NVMs), such as resistive random access memory, phase change RAM, and spin transfer torque magnetic RAM (STT-MRAM), to implement PIM shows great potential and attracts extensive attention due to the inherent capability to realize logic functions [7]–[14]. In particular, STT-MRAM could be a promising NVM candidate for PIM platform, owing to its high endurance, high density, high speed, and low power. In a PIM platform, one of the most important keys is to build an effective logic paradigm. To date, a number of studies are under research and development. They can be divided into three primary categories. The first one is a sensing-based approach [12], [15]. The major problem of this approach is that it can only achieve some specific logic functions. The second one is implication logic [13], [16]–[19]. It generally requires multiple memory cells connected together. Logic functions can then be achieved by supplying different voltage pulses to the memory cells. This method can do achieve a complete set of logic functions, however, on the cost of iteration. The last one is called sequential logic [14], [20]–[21], which achieves logic functions by a series of write cycles plus a read cycle. This approach can also achieve a complete set of logic functions, again, on the cost of iterative write energy and delay.

In this paper, we propose an efficient PIM paradigm, termed spintronic processing unit (SPU), within STT-MRAM. The STT-MRAM considered here is based on the typical one transistor plus one magnetic tunnel junction (1T1MTJ) cell structure, which is the mainstream cell structure for product. Interestingly, each 1T1MTJ cell can also be reconfigured into a logic gate, i.e., SPU, based on the input operands. The proposed SPU works just like a typical memory cell and all the logic functions are achieved via regular memory-like write and read operations with minimal modifications. Therefore, we are able to perform different tasks—data storage and logic computing—using the same physical fabric that is programmable at the finest grain, that is, at the individual memory cell level, without the need to move data outside the memory fabric. With the rapid maturity and commercialization of STT-MRAM, this paper can pave the way towards the application of STT-MRAM for feasible finer-grained PIM platform. In summary, our primary contributions are as follows:

1) We proposed a novel PIM paradigm with 1T1MTJ cell structure. This paradigm belongs to the sequential logic type. Through our optimization, it can realize

Manuscript received December 7, 2018; revised January 19, 2019; accepted February 6, 2019. Date of publication February 26, 2019; date of current version March 22, 2019. This work was supported in part by the National Natural Science Foundation of China under Grant 61871008 and Grant 61571023 and in part by the International Mobility Project under Grant B16001. The review of this paper was arranged by Editor A. M. P. Anantram. *(Corresponding authors: Wang Kang; Weisheng Zhao.)*

The authors are with the Fert Beijing Institute, BDBC, Beihang University, Beijing 100191, China, and also with the School of Microelectronics, Beihang University, Beijing 100191, China (e-mail: wang.kang@buaa.edu.cn; weisheng.zhao@buaa.edu.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TED.2019.2898391

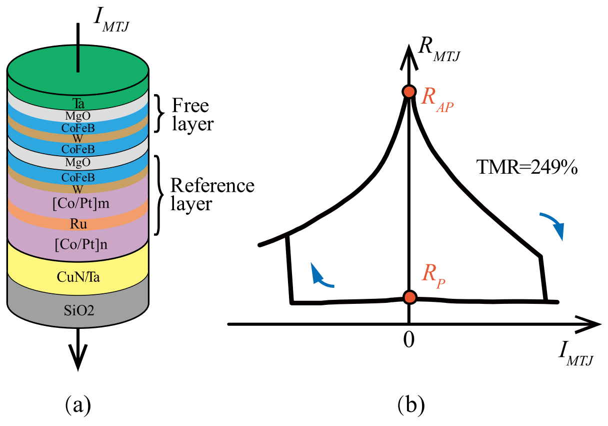


Fig. 1. (a) Structure of the p-MTJ stack with MgO/CoFeB/W/CoFeB/MgO free layer and W bridging layer. Co/Pt multilayers are synthetic antiferromagnetic layers for bottom pinning. (b) STT switching measured by dc current sweep. Arrows show the perpendicular magnetization transitions from AP to P states or the opposite situation [22].

different logic functions within one or two read/write cycles.

2) We implemented the SPU paradigm in a typical 1T1MTJ-based STT-MRAM with a small modification for the array structure. Only a transmission gate is added on each word line (WL) to control the access signal.

3) We proposed an architecture for the SPU and the CPU. Based on the conventional von Neuman architecture, we added a memory control unit in the main memory, to translate the computing commands into read or write operations in the PIM platform.

The remainder of this paper is organized as follows. Section II presents a brief introduction of the MTJ and describes the principle, implementation and architecture of the proposed SPU within STT-MRAM. The simulation results are shown in Section III. Finally, Section IV concludes this paper.

II. Processing in 1T1MTJ-Based STT-MRAM

A. Introduction of MTJ

In recent years, MTJ has drawn widely attentions due to its fascinating prospect. Among them, the perpendicular anisotropy-based MTJ (p-MTJ) has great potential for reducing power consumption and downscaling the device size. Fig. 1(a) shows the structure of a recently reported p-MTJ stack from our group. It mainly contains a free layer, reference layer, and oxide layer. The magnetization direction of free layer can be altered freely, representing the data to be stored, while the magnetization direction of the reference layer is fixed. Fig. 1(b) shows the STT-induced switching of the MTJ with dc current sweep. In this MTJ structure, the tunnel magnetoresistance ratio can reach 249% [22], which is rather preferable for high-speed memory-read and PIM operations.

B. Principle of Reconfigurable Logic Functions

As shown in Fig. 2(a), a typical 1T1MTJ cell consists of an access transistor connected in series with an MTJ device [23], [24]. The cells are arranged into an array through the bit line (BL), WL, and source line (SL), as shown in Fig. 3(a). Write and read operations can be achieved by supplying corresponding voltages across the target BL and SL, when the access transistor is ON by activating the WL. Regarding logic operations with the 1T1MTJ cell, the first step is to choose the input operands of the logic functions. We define the bias voltage on the access transistor (V_{WL}), the initial content stored in the MTJ and the write voltage across the BL and SL ($V_{\mathrm{BL-SL}}$) as the logic input operands A, B, and C. In particular, if $V_{\mathrm{WL}} = \mathrm{GND}$, it represents a logical input "0" ($A = 0$), means that the transistor is off and the MTJ is not accessible; otherwise if $V_{\mathrm{WL}} = V_{\mathrm{DD}}$, the logical input is "1" ($A = 1$) and the transistor is ON. Furthermore, we assume that a low-resistance state and a high-resistance state of the MTJ device represent data "0" and "1" ($B = 0$ and $B = 1$), respectively. For signal C, we define $C = 0$ when $V_{\mathrm{BL-SL}}$ attempts to write "0" ($V_{\mathrm{BL}} = V_{\mathrm{DD}}$, $V_{\mathrm{SL}} = \mathrm{GND}$), and $C = 1$ when it attempts to write "1" ($V_{\mathrm{BL}} = \mathrm{GND}$ and $V_{\mathrm{SL}} = V_{\mathrm{DD}}$) into the memory cell. Based on the above configurations, we can obtain a state transition diagram of the 1T1MTJ cell depending on different combinations of A, B, and C, as shown in Fig. 2(b). The diagram shows that the transition occurs only in two conditions: $B = 0$ ($A = 1$, $C = 1$) $\rightarrow$ $B = 1$ and $B = 1$ ($A = 1$, $C = 0$) $\rightarrow$ $B = 0$. A truth table can then be achieved as shown in Fig. 2(c), where B_i represents the initial content stored in the MTJ device, while B_{i+1} denotes the final logic computing result stored in the MTJ device. After a Karnaugh analysis on the truth table, we can obtain the following describing the above relationship of these operands, expressed as:

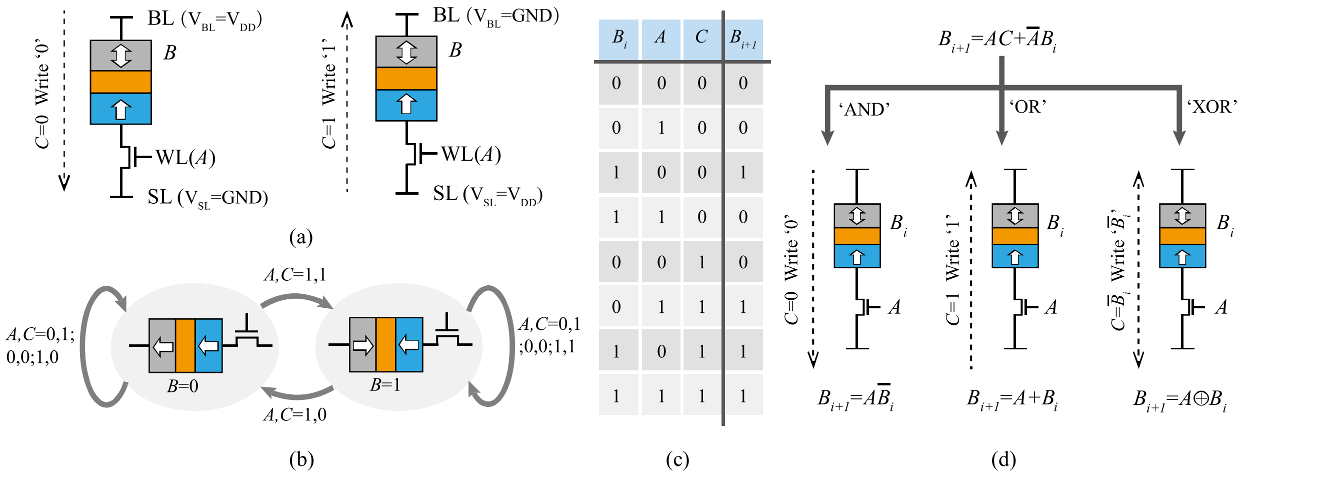
$$B_{i+1} = AC + \bar{A}B_i. \tag{1}$$

Equation (1) is the core rule of the reconfigurable SPU. As shown in Fig. 2(d), here, we choose C as the control signal that determines the type of logic function, and the other two signals are logic input operands. In specific, if $C = 0$, the equation can be simplified to $B_{i+1} = \bar{A}B_i$, which performs an "AND" operation between $\bar{A}$ and B_i; otherwise, if $C = 1$, the simplified equation is $B_{i+1} = A + B_i$, which performs a "OR" operation between A and B_i. Furthermore, if we set $C = \bar{B}_i$, the equation can be transformed into $B_{i+1} = A\bar{B}_i + \bar{A}B_i = A \oplus B_i$, which performs an "XOR" operation between A and B_i. Tables I and II show the expression and truth table for the three reconfigurable Boolean logic functions. All other Boolean logic functions can be realized similarly with the 1T1MTJ cell. Thereby, each 1T1MTJ cell can be reconfigured into either a memory cell or an SPU via regular memory-like write and read operations.

As shown above, different logic functions can be performed in single-1T1MTJ memory cell. Besides, different with the sensing-based approach and the implication logic, we do not require to move the data together, therefore, sample logic functions can be realized within one write operation (one more read operation for "NOR"), when suppose the data are already loaded to the memory. However, this operation will cover the original data in the memory cell. Therefore, if the original data should be remained, it need to be copied to another memory cell, which causes an extra write and read operation.

C. Implementation of the SPU in 1T1MTJ STT-MRAM

Next, we illustrate how to implement the SPU paradigm in a typical 1T1MTJ STT-MRAM. Fig. 3(a) shows a modified array structure of an STT-MRAM, which adds a transmission gate and an NMOS transistor on each WL, compared to a typical STT-MRAM array. The sensing circuit here utilizes



B_i	A	C	B_{i+1}
0	0	0	0
0	1	0	0
1	0	0	1
1	1	0	0
0	0	1	0
0	1	1	1
1	0	1	1
1	1	1	1

Fig. 2. (a) Bit-cell structure of the 1T1MTJ-based STT-MRAM. Here, the bias voltage on the access transistor, the initial content in the MTJ, and the write voltage across the BL and SL are defined as the logic input operands *A*, *B*, and *C*, respectively. (b) State transition diagram of the 1T1MTJ cell with different combinations of *A*, *B*, and *C*. (c) Truth table depending on the three input logic operands, here, B_{i+1} represents the logic computing output. (d) Principle of implementing reconfigurable logic functions. A logic expression ($B_{i+1} = AC + \bar{A}B_i$) can be obtained from the truth table. By assigning different *C*, we can get "AND," "OR," and "XOR" functions.

a typical precharge sensing amplifier of STT-MRAM [25], because the realization of the proposed PIM paradigm has no special requires for the sensing circuit. The transmission gate is used for controlling the transfer of the access signal, and the transistor is utilized to ensure the WL is on 0 voltage potential when the transmission gate is OFF. In this configuration, we can control the STT-MRAM array to work either in a regular data storage mode or in a logic computing mode. If the array works in the data storage mode, the transmission gates are always ON. The signal A is set as V_{DD}. The signal B_i is the initial data stored in the MTJ. The signal C from the BL is the target data that waits to be written into the MTJ. Otherwise, if the array works in the logic computing mode, A can be treated as an external logic input operand from the WL, which can be either "0" or "1" owing to the transmission gate. The signal B_i is the initial data stored in the MTJ. The logic function is controlled by the signal C from the BL. The final MTJ state B_{i+1} depends on the combination of A, B, and C, as described above (see also Fig. 2).

Furthermore, by utilizing multiple arrays in STT-MRAM, multibit logic operations can be achieved in a highly parallel structure, as shown in Fig. 2(b). Assume that there are X blocks, the C[X] is the data waited to write in it, and the A[X] is the other logic operand. These blocks are independent between each other, therefore the B[X], which is selected by the address, can perform different logic functions in different bits simultaneously, according to C[X]. In addition, for an entire memory, it consists of many this kind of multiple arrays, therefore, the parallelism can be improved further. In summary, the memory can achieve parallel computing by seeing each block as a computing unit.

Besides, based on the proposed modified array structure, more complex logic functions can be achieved. Fig. 4 shows the implementation of a 1-bit full adder in the memory array. The X and Y are the addends, the Z_i and Z_{i+1} are the input and output carry, respectively, when the S is the output sum. The R_1R_3 are the registers in the memory. The "LOG" instruction represents the specific logic write operation. It consists of three operands, the first one is signal A, the second one is the signal C, and the last one is the address of the target memory cell. As shown, by utilizing three memory cells and three registers in the memory, the 1-bit full adder can be achieved within five read operations and five write operations. The results are in-situ stored in these memory cells.

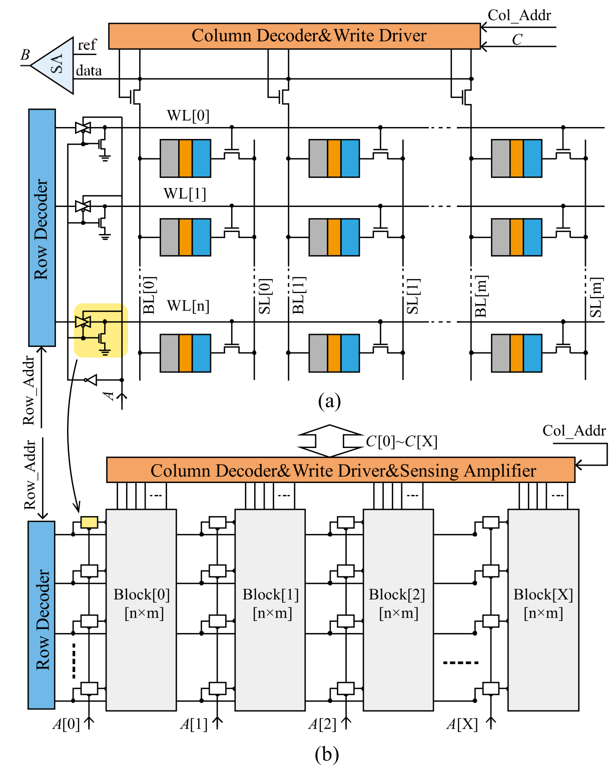


Fig. 3. (a) Modified array structure of a typical 1T1TMTJ-based STT-MRAM, which adds a transmission gate on each WL, compared with a typical STT-MRAM array. The signal *A* from the WL controls the access of the cell, and the signal *C* from the BL is the target data that waits to be written into the target MTJ. (b) Multiple arrays for multibit logic computing. Each block has an independent input signal *A*. Therefore, it can achieve X-bits logic operation, simultaneously.

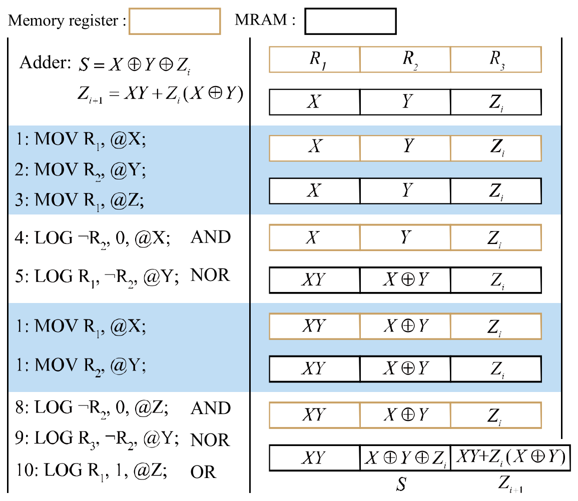
D. Configuration of a PIM Machine

As shown in Fig. 5(a), the convention von Neuman architecture is composed by a central processing unit (CPU) and main memory. Besides, the CPU consists of the control unit

TABLE I

EXPRESSIONS AND OPERATIONS OF THE RECONFIGURABLE BOOLEAN LOGIC FUNCTIONS

Logic Function	Expression	Actual Input		Control
		A	B_i	C
"AND"	$p \cdot q$	$\bar{p}$	q	0
"OR"	$p + q$	p	q	1
"XOR"	$\bar{p} \cdot q + p \cdot \bar{q}$	p	q	$\bar{q}$



Memory register :
MRAM :
Adder: $S = X \oplus Y \oplus Z_i$
$Z_{i-1} = XY + Z_i(X \oplus Y)$
R_1 R_2 R_3
X Y Z_i
1: MOV R$_1$, @X;
2: MOV R$_2$, @Y;
3: MOV R$_1$, @Z;
4: LOG ¬R$_2$, 0, @X; AND
5: LOG R$_1$, ¬R$_2$, @Y; NOR
1: MOV R$_1$, @X;
1: MOV R$_2$, @Y;
8: LOG ¬R$_2$, 0, @Z; AND
9: LOG R$_3$, ¬R$_2$, @Y; NOR
10: LOG R$_1$, 1, @Z; OR
XY $X \oplus Y$ Z_i
XY $X \oplus Y \oplus Z_i$ $XY + Z_i(X \oplus Y)$
S Z_{i-1}

Fig. 4. Implementation of a 1-bit full adder based on the modified array structure. The X and Y are the addends, the Z_i and Z_{i+1} are the input and output carry, respectively, when S is the output sum. The R$_1$R$_3$ are the registers in the memory. The "LOG" instruction represents the specific logic write operation. It consists of three operands, the first one is signal A, the second one is the signal C and the last one is the address of the target memory cell.

and the arithmetic logical unit (ALU). For a complete cycle, first, the instruction is fetched from the main memory to the control unit. Then, the control unit decodes the instruction into commands which can be performed by the ALU. Finally, the ALU executes these commands. It reads data from the main memory, calculates them, and stores the computing results back.

However, in order to realize the PIM through the proposed SPU, the configuration of the memory and the CPU needs some modifications. First of all, it needs to be declared that we speculate the SPU is more suitable for memory intensive tasks, because the compute-intensive tasks will offset the advantage of reducing data transfer due to the delay and energy consumption of write operation. Based on this premise, Fig. 5(b) shows a new master-slave architecture which can better serve our proposed SPU. It still consists of two parts: the CPU and the SPU. For the CPU, we keep the original composition (control unit and ALU). On the other hand, the SPU is composed by a memory control unit and the modified memory array which is shown in Fig. 3. The memory control unit is responsible for translating computing commands into write operations. The working process is as follows. First, the instruction is fetched from the main memory to the control unit and decodes the instruction into commands. Then, if this task is marked with compute intensive, it will be completed by the ALU, just like in a typical von Neuman architecture. Otherwise, if the task is marked with memory intensive, the commands will be sent

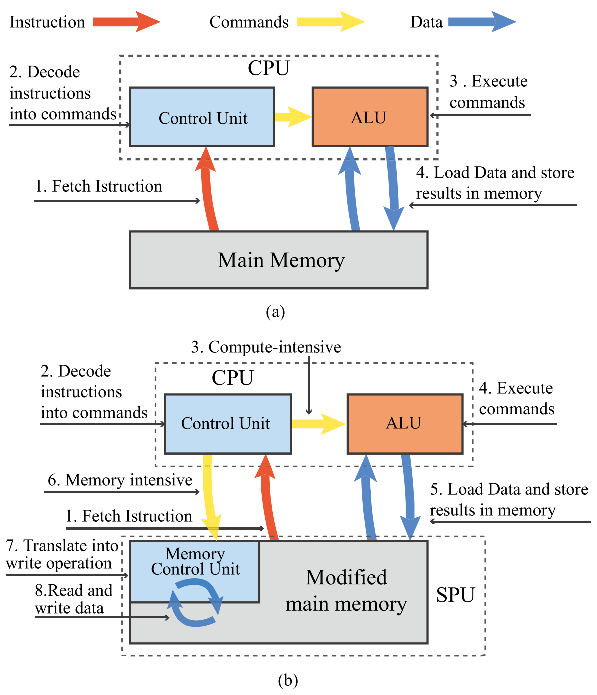


Fig. 5. (a) Configuration of the conventional von Neuman architecture. (b) Possible PIM architecture based on the proposed SPU. The SPU is composed by a memory control unit and the modified memory array, when the CPU is basically unchanged. This architecture can take full advantage of the SPU by classifying the computing tasks (compute intensive or memory intensive) and send them to the ALU and SPU, respectively.

back to the memory control unit. Then, the memory control unit translates these commands into a series of write and read operations. Finally, the task is achieved in the modified main memory, and the results are in-situ stored in it. It should be noted that the software support (out of the scope of this paper) is required to distribute different tasks to either CUP or SPU in the proposed architecture.

III. RESULTS AND DISCUSSION

The simulation was performed using a 40-nm CMOS design kit and a compact MTJ model [26]. Hybrid CMOS/MTJ circuits were designed and simulated to evaluate the functionality and performance of the proposed SPU in STT-MRAM. Some key parameters are shown in Table III. Fig. 6 shows the transient simulation waveforms of the "AND," "OR," and "XOR" logic functions, respectively. It consists of the state of the MTJ (i.e., signal B), the voltage waveforms of the signal A and signal C. Here, 0 V/1.5 V represents logic value "0"/"1" for A, and +1.5 V/−1.5 V represents logic value "0"/"1" for C. As can be seen, the proposed SPU can perform these three logic functions within one single-write operation via the memory-like operations. The performance of "AND," "OR," and "XOR" are shown in Table V. As shown, the power consumptions of different logic functions are not the same, because the write currents are different when the MTJ is in different states, and the average durations of these two kinds of currents are different for those logic functions. For the logic operation speed, the "AND" and "OR" operations take as long as a write operation, when the "XOR" operation requires a longer time, because of the extra read cycle.

Besides, we verified the function of the modified main memory, as shown in Fig. 7. It contains the wave forms of four

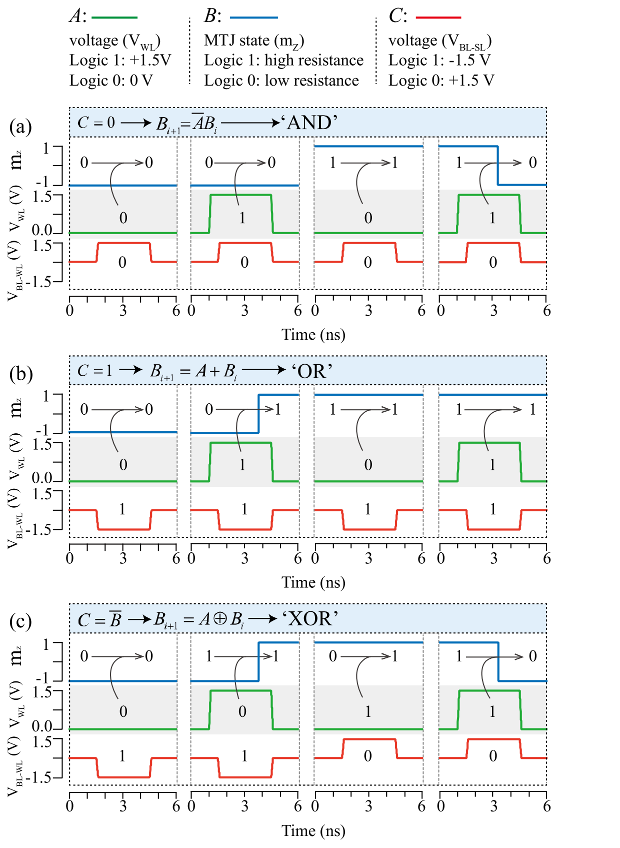


Fig. 6. Transient simulation waveforms of the proposed SPU within 1T1MTJ STT-MRAM based on hybrid MTJ/CMOS circuit simulation. (a) AND, (b) OR, and (c) XOR functions, respectively.

TABLE II
TRUTH TABLE OF THE THREE BOOLEAN LOGIC FUNCTIONS

Logic Function	Input p	Input q	Output	Actual Input A	Actual Input B_i	Control C
"AND"	0	0	0	1	0	0
	0	1	0	1	1	0
	1	0	0	0	0	0
	1	1	1	0	1	0
"OR"	0	0	0	0	0	1
	0	1	1	0	1	1
	1	0	1	1	0	1
	1	1	1	1	1	1
"XOR"	0	0	0	0	0	1
	0	1	1	0	1	0
	1	0	1	1	0	1
	1	1	0	1	1	0

selected MTJs, and their corresponding signals A and C. The SPU here can switch between the data storage mode and the logic computing mode freely. For the area consumption, take a 16 × 256 memory array as an example. Our modified array only has a 5.9% increase, compared with the conventional memory array. The speed and power consumption for the AND, OR, and XOR functions are shown in Table IV.

In order to evaluate our work more objectively, a comparison with other NVM-based PIM logic paradigms are shown in Table V. The "steps" is calculated in the situation below: the two logic operands are in different places in the memory and the computing result requires still be stored in the memory. As shown, compared with other logic paradigms, the proposed PIM paradigm utilizes STT-MRAM, which has a more mature technology. Besides, it has lower steps, with the

TABLE III
SIMULATION PARAMETER

Parameters	Default Value
Free layer dimension, $(W \times L \times t)_{FL}$	40 × 40 × 1.3 nm
Anisotropy field, H_K	1.433×10^3 A/m
Saturation magnetization, M_S	15.8×10^3 A/m
Polarization factor, P	0.52
Oxide layer thickness, t_{ox}	0.85 nm
Gilbert Damping Factor, α	0.027
Resistance area product, RA_p	$5\ \Omega \cdot \mu m^2$
TMR ratio at zero bias	200%
Temperature, T	300 K
Supply voltage	1.5 V
CMOS technology	40nm

TABLE IV
PERFORMANCE OF DIFFERENT LOGIC FUNCTIONS

Logic function	Average power consumption	Logic operation speed
AND	323.5 fJ	6 ns
OR	109.5 fJ	6 ns
XOR	278.9 fJ	10 ns

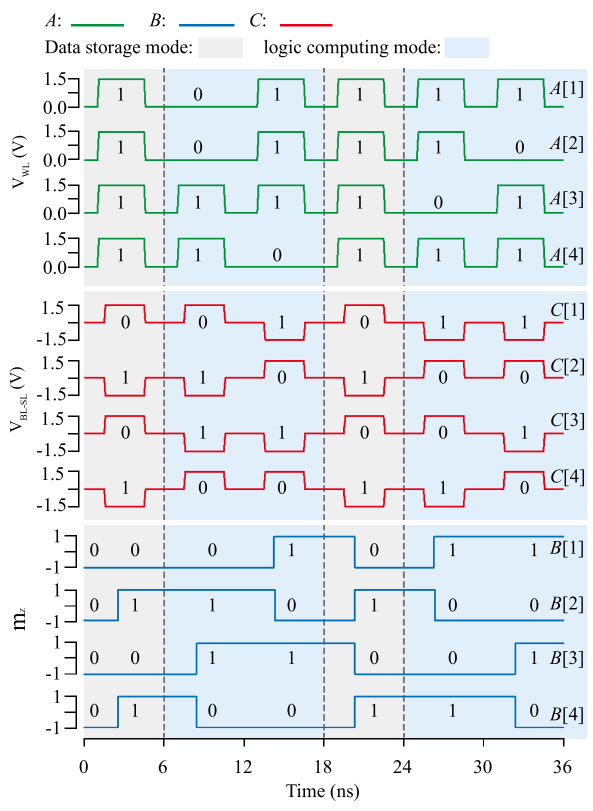


Fig. 7. Wave forms of the modified main memory. It contains the signals A and C and the states of four selected MTJs. The SPU here switches between the data storage mode and the logic computing mode.

TABLE V
COMPARISON OF VARIOUS PROCESS IN MEMORY LOGIC PARADIGM

Methods	Ref	Devices	Steps	Issues
Sensing-based	[15]	2 ReRAM	3 read + 3 write	Costly mapping Read reliability
	[12]	2 MTJ	3 read + 3 write	
Implication Logic	[13]	3 MTJ	2 read + 4 write	Costly mapping Write reliability
	[16]	3 ReRAM	2 read + 4 write	
Sequential logic	[14]	1T1R	2 read + 2 write	Need initialize
	[21]	1 ReRAM	2 read + 2 write	
Our work		1T1MTJ	1/2 read + 1/2 write	Original data covered or one more write operation required

drawback of covering the original data. As mentioned, if the original data need to be remained, the "steps" will rise to two read and two write operations. However, many application scenarios do not need to remain the original data, such as some image processing and encryption. Therefore, the proposed PIM paradigm will still have a bright prospect.

IV. CONCLUSION

We proposed an efficient and reconfigurable PIM platform—SPU—in typical 1T1MTJ-based STT-MRAM. The proposed SPU utilized the memory cell at the finest grain to perform logic computing task through regular memory-like write and read operations with minimal modifications. Our hybrid CMOS/MTJ circuit simulations verified the functionality and performance of the proposed design. It can be argued that this paper is a natural evolution of the PIM paradigm, by moving towards finer-grained and highly parallel structures. As STT-MRAM has recently been a commercialized product, our proposed SPU is expected to be a practical PIM platform.

REFERENCES

[1] N. S. Kim *et al.*, "Leakage current: Moore's law meets static power," *Computer*, vol. 36, no. 12, pp. 68–75, Dec. 2003. doi: 10.1109/MC.2003.1250885.

[2] W. A. Wulf and S. A. McKee, "Hitting the memory wall: Implications of the obvious," *ACM SIGARCH Design Comput. Archit. News*, vol. 23, no. 1, pp. 20–24, Mar. 1995. doi: 10.1145/216585.216588.

[3] J. Ahn, S. Yoo, O. Mutlu, and K. Choi, "PIM-enabled instructions: A low-overhead, locality-aware processing-in-memory architecture," in *Proc. ACM/IEEE 42nd Annu. Int. Symp. Comput. Archit. (ISCA)*, Jun. 2015, pp. 336–348. doi: 10.1145/2749469.2750385.

[4] Y. Xie, "Future memory and interconnect technologies," in *Proc. Design Automat. Test Eur. Conf. Exhib. (DATE)*, Mar. 2013, pp. 964–969. doi: 10.7873/DATE.2013.202.

[5] M. Gao, G. Ayers, and C. Kozyrakis, "Practical near-data processing for in-memory analytics frameworks," in *Proc. Int. Conf. Parallel Archit. Compilation (PACT)*, Oct. 2015, pp. 113–124. doi: 10.1109/PACT.2015.22.

[6] B. C. Lee, E. Ipek, O. Mutlu, and D. Burger, "Architecting phase change memory as a scalable dram alternative," in *Proc. 36th Annu. Int. Symp. Comput. Archit. News*, vol. 37, no. 3, Jun. 2009, pp. 2–13. doi: 10.1145/1555815.1555758.

[7] D. Reis, M. Niemier, and X. S. Hu, "Computing in memory with FEFETs," in *Proc. ACM ISLPED*, Jul. 2018, p. 24. doi: 10.1145/3218603.3218640.

[8] S. Li, C. Xu, Q. Zou, J. Zhao, Y. Lu, and Y. Xie, "Pinatubo: A processing-in-memory architecture for bulk bitwise operations in emerging non-volatile memories," in *Proc. ACM/EDAC/IEEE DAC Design Automat. Conf.*, Jun. 2016, pp. 1–6. doi: 10.1145/2897937.2898064.

[9] H. Zhang, W. Kang, L. Wang, K. L. Wang, and W. Zhao, "Stateful reconfigurable logic via a single-voltage-gated spin hall-effect driven magnetic tunnel junction in a Spintronic memory," *IEEE Trans. Electron Devices*, vol. 64, no. 10, pp. 4295–4301, Oct. 2017. doi: 10.1109/TED.2017.2726544.

[10] G. C. Adam, B. D. Hoskins, M. Prezioso, and D. B. Strukov, "Optimized stateful material implication logic for three-dimensional data manipulation," *Nano Res.*, vol. 9, no. 12, pp. 3914–3923, Dec. 2016. doi: 10.1007/s12274-016-1260-1.

[11] J. Lee, D. I. Suh, and W. Park, "The universal magnetic tunnel junction logic gates representing 16 binary Boolean logic operations," *J. Appl. Phys.*, vol. 117, no. 17, Apr. 2015, Art. no. 17D717. doi: 10.1063/1.4916806.

[12] Z. He, S. Angizi, and D. Fan, "Exploring STT-MRAM based in-memory computing paradigm with application of image edge extraction," in *Proc. IEEE Int. Conf. Comput. Design (ICCD)*, Nov. 2017, pp. 439–446. doi: 10.1109/ICCD.2017.78.

[13] Z. Chowdhury *et al.*, "Efficient in-memory processing using spintronics," *IEEE Comput. Archit. Lett.*, vol. 17, no. 1, pp. 42–46, Jan./Jun. 2018. doi: 10.1109/LCA.2017.2751042.

[14] Z. R. Wang *et al.*, "Functionally complete boolean logic in 1T1R resistive random access memory," *IEEE Electron Device Lett.*, vol. 38, no. 2, pp. 179–182, Feb. 2017. doi: 10.1109/LED.2016.2645946.

[15] L. Gao, F. Alibart, and D. B. Strukov, "Programmable CMOS/memristor threshold logic," *IEEE Trans. Nanotechnol.*, vol. 12, no. 2, pp. 115–119, Mar. 2013. doi: 10.1109/TNANO.2013.2241075.

[16] P. Huang *et al.*, "Reconfigurable nonvolatile logic operations in resistance switching crossbar array for large-scale circuits," *Adv. Mater.*, vol. 28, no. 44, pp. 9758–9764, Nov. 2016. doi: 10.1002/adma.201602418.

[17] H. Li *et al.*, "A learnable parallel processing architecture towards unity of memory and computing," *Sci. Rep.*, vol. 5, Aug. 2015, Art. no. 13330. doi: 10.1038/srep13330.

[18] H. A. Du Nguyen, L. Xie, M. Taouil, R. Nane, S. Hamdioui, and K. Bertels, "On the implementation of computation-in-memory parallel adder," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 25, no. 8, pp. 2206–2219, Aug. 2017. doi: 10.1109/TVLSI.2017.2690571.

[19] K. Cao *et al.*, "In-memory direct processing based on nanoscale perpendicular magnetic tunnel junctions," *Nanoscale*, vol. 10, no. 45, pp. 21225–21230, Oct. 2018. doi: 10.1039/C8NR05928D.

[20] M. Soeken, P. E. Gaillardon, S. Shirinzadeh, R. Drechsler, and G. D. Micheli, "A PLiM computer for the Internet of Things," *Computer*, vol. 50, no. 6, pp. 35–40, Jun. 2017. doi: 10.1109/MC.2017.173.

[21] E. Linn, R. Rosezin, S. Tappertzhofen, U. Böttger, and R. Waser, "Beyond von Neumann–logic operations in passive crossbar arrays alongside memory operations," *Nanotechnology*, vol. 23, no. 30, Aug. 2012, Art. no. 30520. doi: 10.1088/0957-4484/23/30/305205.

[22] M. Wang *et al.*, "Current-induced magnetization switching in atom-thick tungsten engineered perpendicular magnetic tunnel junctions with large tunnel magnetoresistance," *Nat. Commun.*, vol. 9, no. 1, p. 671, Feb. 2018. doi: 10.1038/s41467-018-03140-z.

[23] M. Durlam *et al.*, "A 1-Mbit MRAM based on 1T1MTJ bit cell integrated with copper interconnects," *IEEE J. Solid-State Circuit*, vol. 38, no. 5, pp. 769–773, May 2003. doi: 10.1109/JSSC.2003.810048.

[24] K. Rho *et al.*, "A 4Gb LPDDR2 STT-MRAM with compact 9F2 1T1MTJ cell and hierarchical bitline architecture," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2017, pp. 396–397. doi: 10.1109/ISSCC.2017.7870428.

[25] W. Kang *et al.*, "Separated precharge sensing amplifier for deep submicrometer MTJ/CMOS hybrid logic circuits," *IEEE Trans. Magn.*, vol. 50, no. 6, Jun. 2014, Art. no. 3400305. doi: 10.1109/TMAG.2013.2297393.

[26] Y. Zhang *et al.*, "Compact model of Subvolume MTJ and its design application at Nanoscale technology nodes," *IEEE Trans. Electron Devices*, vol. 62, no. 6, pp. 2048–2055, Jun. 2015. doi: 10.1109/TED.2015.2414721.