Current-Sensing Efficient Adder for Processing-in-Memory Design

Joonseop Sim, Mohsen Imani, Woojin Choi, Yeseong Kim and Tajana Rosing
University of California San Diego
{j7sim, moimani, woc015, yek048, tajana}@ucsd.edu

This is based on paper [1], presented at International Symposium on Quality Electronic Design (ISQED) 2018. You can access to the full paper at:
https://ieeexplore.ieee.org/abstract/document/8357265

I. Introduction

Processing in memory (PIM) is a promising way to address the data movement issue by processing data inside the memory, thus improving both performance and energy efficiency [2]–[4]. Prior PIM techniques support limited basic functionalities such as basic bitwise operations (AND, OR, and IMP). Although several techniques have been proposed to perform addition and multiplication in NVM architectures [5]–[8], they execute these functions by combining multiple boolean operations (IMP, NOT, NOR). Therefore, they are inherently slow due to their multi-cycle operation as well as slow processing speed.

In this paper, we propose a high performance and low cost PIM architecture based on the latch-up effect of thyristors, enabling single-cycle addition (ADD), and significantly improving the performance of multiplication (MUL). Our design requires no additional cell array for processing, hence can be an excellent candidate for the storage class memory which has been considered as the main application of memristor-based products.

II. Proposed Design

A. Thyristor Latch-Up

We exploit a vertical PNPN structure commonly referred to a *thyristor* [9]. Fig. 1a shows the structure used in our design. The structure has three P-N junctions and is equivalent to two cross-coupled bipolar junction transistors (BJTs) as shown in Fig. 1a. This structure has a short-circuit path, often referred as *latch-up* in CMOS design. When one of the two BJTs gets forward biased, it feeds the base of the other BJT. This positive feedback increases the current until it saturates to I_{short}.

Fig. 1b shows the voltage and the current behaviors of the structure. In the initial state, a thyristor has a high resistance (2MΩ in our experimental setup). When the voltage across the device (V_{AB}) is increased, the device keeps the high resistance state until V_{AB} reaches the latch-up voltage (V_{LU}). Latch-up occurs at V_{LU} and the current through the cell (i.e., from A to B in Fig. 1a) abruptly increases until the applied bias turns back to the latch-down voltage (V_{LD}). In order to restore the thyristor device resistance to the original state, a reverse bias, V_{RC}, should be applied to V_{AB}. It moves the minor carriers out from the base regions, and the device is set to the initial state again. In the rest of the paper, we call this recovery state as the *write back* step.

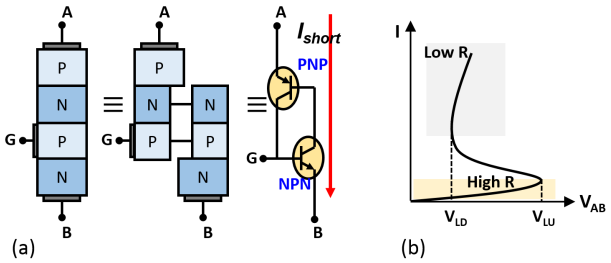


Fig. 1. (a) Thyristor Structure and (b) Voltage-Current Behavior

B. Sensing Circuit Design for Addition

We propose a new sensing circuit, which exploits the thyristor latch-up effect, to enable ADD operation in a cycle. Fig. 2 is the schematic of the proposed design. The design consists of two parts: the current mirror and adder. Once three rows of a memory block corresponding to the input values (A, B, C_{in}) are activated, the total current from the activated rows denoted as I_{BL} is delivered to the selected BL. The current mirror circuit in Fig. 2 copies the I_{BL} to I_1 and I_2 and delivers them to *Carry Out* (C_{out}) and *Sum* branches, respectively. Our design computes the outputs by distinguishing the total current amount reached to the sensing circuit.

Fig. 3a presents the truth table of a full adder. The sum is the exclusive-or (XOR) result for the three inputs and the carry out is the majority function of the inputs. The three inputs are interchangeable in that the order of them does not affect the output. In the memsistor devices, the amount of the bitline current is the combination of one R_{on} and two R_{off}. Based on this characteristic, there are four different cases depending on the current amount, I_{000}, I_{100}, I_{110} and I_{111}, according to the number of high (0) and low (1) resistances in the memristors of activated rows.

Fig. 3b shows how the proposed circuit distinguishes the four current regions to create the desired C_{out} and *Sum*. In our circuit design, there are three major voltage nodes (i.e., V_1, V_2, and V_3 shown in Fig. 2) whose potential determine the final outputs of the *Sum* and C_{out} by the following digital logic gates. The voltage of each node is transferred as a function of the current in the selected bitline. V_{THR} is a threshold value which determines whether an input potential is interpreted as a logic 0 or 1 (i.e., any value less than V_{THR} is 0 and any value above V_{THR} is 1.) Let R_{thy} be the resistance of the thyristor. Then, the electric potentials at V_1, V_2 and V_3 are represented by $V_1 = I_1 \cdot (2R)$, $V_2 = I_2 \cdot (3R)$ and $V_3 = I_2 \cdot (2R \cdot R_{thy})/(2R + R_{thy})$, respectively.

As for the carry-out function, V_1 node has higher electric potential than V_{THR} in cases of I_{110}, I_{111} when it delivers a logic 1 to C_{out} through two inverters which strengthen the

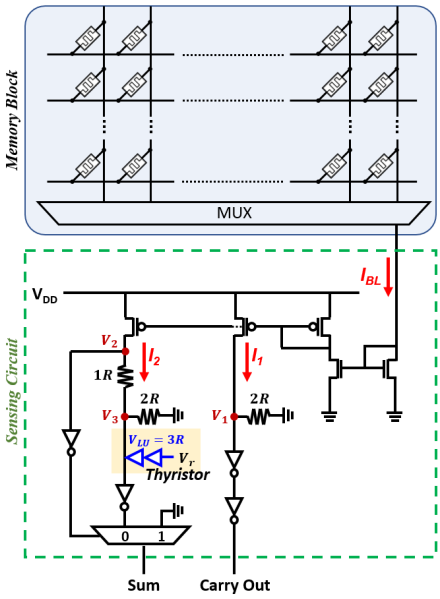
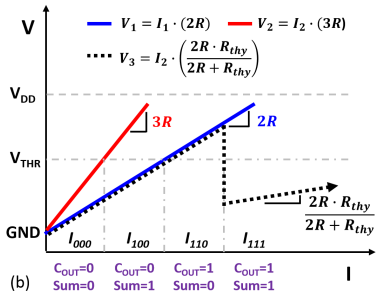


Fig. 2. Proposed Sensing Circuit for Addition

A	B	C_{IN}	C_{OUT}	Sum	I_{BL}
0	0	0	0	0	I_{000}
0	0	1	0	1	I_{100}
0	1	0	0	1	
1	0	0	0	1	
0	1	1	1	0	I_{110}
1	0	1	1	0	
1	1	0	1	0	
1	1	1	1	1	I_{111}

(a)



(b)

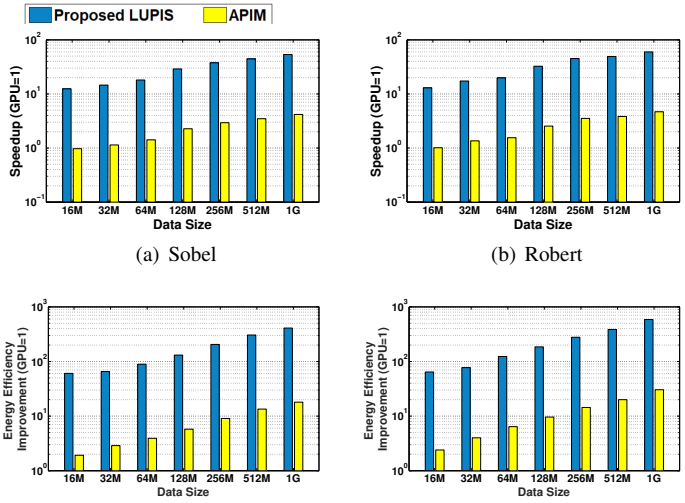
Fig. 3. Voltage Transfer as a Input Current (I_{BL})

signal. For the I_{000} and I_{100} cases, V_1 node has lower potential than V_{THR} and C_{out} presents a logic 0.

The *Sum* function uses branches where V_2 and V_3 nodes are located. As shown in Fig. 3b, V_2 node has lower potential than V_{THR} only in case of I_{000} and its inverted logic 1 is delivered to the MUX, pulling down the *Sum* potential to the ground, i.e., logic 0. In the opposite cases, i.e., I_{100}, I_{110} and I_{111}, V_2 has a logic 1, and the MUX delivers the data from the connection where V_3 is located, so that V_3 decides the outputs for the three cases. For I_{100} and I_{110} case, the V_3 shows either logic 0 or logic 1 depending on the input current in a similar way of V_1, since the thyristor is not activated in this region. However, once the current increases and reaches the I_{111} range, the latch-up occurs in the thyristor device, and thus the electric potential of V_3 abruptly falls below V_{THR}, making *Sum* logic 1. With this logic, our design is able to complete all *Sum* and C_{out} results. Once the final outputs are generated, we reset the thyristor for the next cycle, by invoking the write-back procedure to turn the thyristor resistance back to the original state. In case of N-bit addition, C_{out} is written back to the memory and is used to calculate the results for the next bit.

III. Experimental Results

There are several applications which can benefit by the PIM-based addition and multiplication. Fig. 4 shows the speedup and energy efficiency improvement of, i) the proposed design and ii) a state-of-the-art PIM technique, APIM [7], over the AMD GPU core. The results present that our proposed design can achieve significantly better energy and performance efficiency. Apart from the superior efficiency improvement, our evaluation also shows that LUPIS energy consumption increases linearly with the data set size, since the PIM capability can highly hide the cost of data movements. In contrast, the energy and execution time of the GPU case do not scale linearly with the data size, as the larger dataset requires higher costs for the data movements before processing. To sum up, our design can achieve up to 62× speedup and 484× lower energy consumption than the GPU architecture. As compared to APIM, the results present 12.7× and 20.9× higher efficiency for speedup and energy respectively.



(a) Sobel (b) Robert

(c) Sobel (d) Robert

Fig. 4. Speedup and Energy Efficiency of Proposed LUPIS and APIM [7] over Different Applications.

Acknowledgements

This work was partially supported by CRISP, one of six centers in JUMP, an SRC program sponsored by DARPA, and also NSF grants #1730158 and #1527034. Joonseop Sim is partially supported by a Ph.D. fellowship from SK Hynix Inc.

References

[1] J. Sim *et al.*, "Lupis: Latch-up based ultra efficient processing in-memory system," in *Quality Electronic Design (ISQED), 2018 19th International Symposium on*, IEEE, 2018.

[2] G. H. Loh *et al.*, "A processing in memory taxonomy and a case for studying fixed-function pim," in *Workshop on Near-Data Processing (WoNDP)*, 2013.

[3] A. M. Aly *et al.*, "M3: Stream processing on main-memory mapreduce," in *Data Engineering (ICDE), 2012 IEEE 28th International Conference on*, pp. 1253–1256, IEEE, 2012.

[4] M. S. Razlighi *et al.*, "Looknn: Neural network with no multiplication," in *DATE*, pp. 1775–1780, IEEE, 2017.

[5] E. Lehtonen *et al.*, "Stateful implication logic with memristors," in *Proceedings of the 2009 IEEE/ACM ISNA*, IEEE Computer Society, 2009.

[6] S. Kvatinsky *et al.*, "Memristor-based material implication (imply) logic: Design principles and methodologies," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 2014.

[7] M. Imani *et al.*, "Ultra-efficient processing in-memory for data intensive applications," in *Proceedings of the 54th Annual Design Automation Conference 2017*.

[8] M. Imani *et al.*, "Resistive configurable associative memory for approximate computing," in *DATE, 2016*, IEEE, 2016.

[9] X. Tong *et al.*, "Two-terminal vertical memory cell for cross-point static random access memory applications," *JVST B*, 2014.