True In-memory Computing with the CRAM: From Technology to Applications

Zamshed I. Chowdhury, Salonik Resch, Masoud Zabihi, Zhengyang Zhao, Thomas Peterson, Mahendra DC
Ulya R. Karpuzcu, Jian-Ping Wang, Sachin Sapatnekar
University of Minnesota, Twin Cities

***Abstract*—Traditional Von Neumann computing is falling apart in the era of exploding data volumes as the overhead of data transfer becomes forbidding. Instead, it is more energy-efficient to fuse compute capability with memory where the data reside. Emerging spintronic technologies show remarkable versatility for the tight integration of logic and memory. In this presentation, we introduce Computational RAM (CRAM), a novel high-density, reconfigurable spintronic in-memory substrate, and survey several years of progress in developing the CRAM concept across the system stack from the device level all the way to applications.**

I. Fusing Computation and Memory

Classical computing platforms are not optimized for efficient data transfer, which complicates large-scale data analytics in the presence of exponentially growing data volumes. Imbalanced technology scaling further exacerbates this situation by rendering data communication, and not computation, a critical bottleneck [5]. Specialization in hardware cannot help in this case unless conducted in a data-centric manner.

Tight integration of compute capability into the memory, *Processing in memory (PIM)*, is especially promising as the overhead of data transfer becomes forbidding at scale. The rich design space for PIM spans full-fledged processors and co-processors residing in memory [6]. Until the emergence of 3D-stacking, however, the incompatibility of the state-of-the-art logic and memory technologies prevented practical prototype designs. Still, 3D-stacking can only achieve *near memory processing, NMP* [1], [2], [8]. The main challenge remains to be fusing computation and memory without violating array regularity.

Emerging spintronic technologies show remarkable versatility for the tight integration of logic and memory. This talk covers a high-density, reconfigurable spintronic in-memory compute substrate, Computational RAM (CRAM) [10]. The basic idea is to add compute capability to the magnetic tunnel junction (MTJ) based memory cell [7], [12], without breaking the array regularity. Thereby each memory cell can participate in gate-level computation as an input or as an output. Computation is not disruptive, i.e., memory cells acting as gate inputs do not loose their stored values. This idea is equally applicable to Spin-Torque-Transfer (STT) and Spin-Orbit-Torque (SOT) based technologies.

CRAM can implement different types of basic Boolean gates to form a functionally complete set, therefore there is no fundamental limit to the types of computation. If implemented using SOT (STT), each column (row) in an CRAM array can have only one active gate at a time, however, computation in all columns (rows) can proceed in parallel. CRAM provides *true* in-memory computing by reconfiguring cells within the memory array to implement logic functions. As all cells in the array are identical, inputs and outputs to logic gates do not need to be confined to a specific physical location in the array. In other words, CRAM can intiate computation at any location in the memory array on demand.

II. Operation Principle

Without loss of generality, we will next use an STT-based CRAM as a running example. In its most basic form, a CRAM array is essentially a 2D magneto-resistive RAM (MRAM). When compared to the standard 1T(ransistor)1M(TJ) MRAM cell, however, each CRAM cell features an additional transistor T_L (Fig.1(a)), which acts as a switch between memory and logic configurations. A CRAM cell can operate as a regular MRAM memory cell or serve as an input/output to a logic gate.

Each MTJ consists of two layers of ferromagnets, termed as pinned and free layers, separated by a thin insulator. The magnetic spin orientation of the pinned layer is fixed; of the free layer, controllable. Changing the orientation of the free layer entails passing a (polarized) current through the MTJ, where the current direction sets the orientation. The relative orientation of the free layer with respect to the pinned layer, i.e., anti-parallel (AP) or parallel (P), gives rise to two distinct MTJ resistance levels, i.e., R_{high} and R_{low}, which encode logic 1 and 0, respectively. As resistance levels represent logic states, Fig.1 depicts each MTJ by its resistance.

Memory Configuration: The dashed components in Fig.1(a) capture all add-ons to the standard MRAM memory cell, in order to support logic functions. When the array is configured as memory, the *Logic Bit Line (LBL)* is set to 0 to turn the switch T_L off, and thereby to disconnect the cells from the *Logic Line (LL)*. In this case, the array becomes equivalent to a standard MRAM array. In the following, we detail the configuration for various memory operations (where *LBL* is always set to 0).

- Data retention: The *Word Line (WL)* is set to 0 to isolate the cells and to prevent current flow through the MTJs.
- Read: *WL* is set to 1, to connect each MTJ to its *Bit Select Line (BSL)* and *Memory Bit Line (MBL)*. A small voltage pulse applied between *BSL* and *MBL* induces a current through the MTJ, which is a function of the resistance level (i.e., logic state), and which in turn a sense amplifier attached to *BSL* captures.
- Write: *WL* is set to 1, to connect each MTJ to its *BSL* and *MBL*. A large enough voltage pulse (in the order of the supply voltage) is applied between *BSL* and *MBL* to induce a large enough current through the MTJ to change the spin orientation of the free layer.

Logic Configuration: *LL* connects all cells participating in computation, on a per row basis. Such cells may act as logic gate inputs or outputs. For each CRAM cell participating in computation, *WL* is set to 0 to disconnect its MTJ from *MBL*. Instead, *LBL* is set to 1 to turn the switch T_L on, which in turn connects the MTJ to the *LL*.

As an example, Fig.1(b) demonstrates the formation of a two input logic gate in the array, where cells labeled by "0", "1", and "2" correspond to the inputs In_0, In_1, and the output Out, respectively. Fig.1(c) depicts the equivalent circuit: *BSL* of the output, BSL_2, is grounded; while *BSL* of the two inputs,

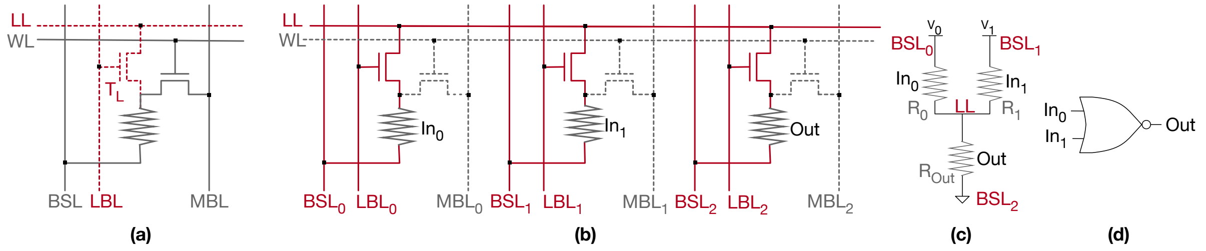

Fig. 1: (a) CRAM cell; (b) 2-input gate formation in the array; (c), (d) 2-input NOR gate circuit equivalents.

In_0	In_1	Out	$I_{Out} = I_0 + I_1$	
0 (R_{low})	0 (R_{low})	1	I_{00}	$> I_{crit}$
0 (R_{low})	1 (R_{high})	0	I_{01}	$< I_{crit}$
1 (R_{high})	0 (R_{low})	0	$I_{10} = I_{01}$	$< I_{crit}$
1 (R_{high})	1 (R_{high})	0	I_{11}	$< I_{crit}$

TABLE I: 2-input NOR truth table (Out pre-set = 0).

BSL_0 and BSL_1, are set to voltages V_0 and V_1. The values of V_0 and V_1 determine the currents through the input MTJs, I_0 and I_1, as a function of their resistance values R_0 and R_1 (i.e., logic states). $I_{Out} = I_0 + I_1$ flows through the output resistance R_{Out}. If I_{Out} is higher than the critical MTJ switching current I_{crit}, it will change the free layer orientation of Out's MTJ, and thereby, the logic state of Out. Otherwise, Out will keep its previous state.

We can easily expand this example to more than two inputs. The key observation is that we can change the logic state of the output as a *function* of the logic states of the inputs, within the array. And voltages at *BSL*s of the inputs dictate how such *function*s would look like.

Continuing with the example from Fig.1(b)/(c), let us try to implement an universal, 2-input NOR gate. Table I provides the truth table. Out would be 0 in this case for all input combinations but $In_0 = 0$, $In_1 = 0$, which incurs the lowest R_0 and R_1, and hence, the highest $I_{Out} = I_0 + I_1$. Let us refer to this value of I_{Out} as I_{00}, following Table I. Accordingly, if we pre-set Out to 0 (before computation starts), and determine V_0 and V_1 such that I_{00} does exceed I_{crit}, while both I_{11} and $I_{01} = I_{10}$ does not, Out would not switch from (its pre-set value) 0 to 1, for all input combinations but $In_0 = 0$, $In_1 = 0$.

As Boolean gates of practical importance (such as NOR) are commutative, a single voltage level at the *BSL*s of the inputs suffices to define a specific logic functionality. Each voltage level can serve as a signature for a specific logic gate. Accordingly, in the above example, $V_0 = V_1$ applies, and its value simply follows from Kirchoff's Laws, where R_{high}, R_{low}, and I_{crit} represent technology dependent constants. While NOR gate is universal, we can implement different types of logic gates following a similar methodology for mapping the corresponding truth tables to the CRAM array.

STT-based CRAM can perform only one type of logic function in a row, at a time, however, multiple rows can perform the very same logic function in parallel, on the same set of columns. In other words, CRAM supports a special form of *SIMD (single instruction multiple data)* parallelism, where *instruction* translates into *logic gate/operation*; and *data*, into *input cells in each row, across multiple rows, which span the very same columns*.

III. Design Space Exploration Across the System Stack

CRAM leverages spintronics, a promising technology due to its robustness, high endurance, and trajectory towards fast improvement [11]. The CRAM idea was first introduced at the device-level in [10] with pointers to circuit-level implementation. Next came a system-level study for STT-based CRAM [3], followed by a cross-level analysis to bridge CRAM technology, circuit implementations, and operation scheduling [14]. Then, in [13], we redesigned CRAM around a new MTJ based on the spin-Hall effect (SHE), providing greatly improved energy efficiency. Using both designs, we recently demonstrated effectiveness of CRAM in accelerating Binary Neural Networks [9] and pattern matching at scale which lies at the core of emerging genomics applications [4].

Acknowledgements

This work was supported in part by the DARPA Non-Volatile Logic program, NSF SPX Award CCF-1725420, and by C-SPIN, one of the six SRC STARnet Centers, sponsored by MARCO and DARPA.

References

[1] "Hybrid Bandwidth Memory (HBM)," http://www.amd.com/en-us/innovations/software-technologies/hbm.

[2] "Hybrid Memory Cube (HMC)," http://www.hotchips.org/wp-content/uploads/hc_archives/hc23/HC23.18.3-memory-FPGA/HC23.18.320-HybridCube-Pawlowski-Micron.pdf.

[3] Z. Chowdhury *et al.*, "Efficient in-memory processing using spintronics," *IEEE Computer Architecture Letters*, vol. 17, no. 1, pp. 42–46, 2018.

[4] Z. Chowdhury *et al.*, "Spintronic in-memory pattern matching using computational ram (cram)," *IEEE Journal on Exploratory Solid-State Computational Devices and Circuits*, November 2019.

[5] M. Horowitz, "Computing's Energy Problem (and What We Can Do About It)," *Keynote at ISSCC*, February 2014.

[6] G. H. Loh *et al.*, "A Processing in Memory Taxonomy and a Case For Studying Fixed-function PIM," in *Workshop on Near-Data Processing in conjunction with MICRO*, 2013.

[7] A. Lyle *et al.*, "Direct Communication between Magnetic Tunnel Junctions for Nonvolatile Logic Fanout Architecture," *Applied Physics Letters*, vol. 97, no. 152504, 2010.

[8] R. Nair *et al.*, "Active Memory Cube: A Processing-in-Memory Architecture for Exascale Systems," *IBM Journal of R.&D.*, vol. 59, no. 2/3, 2015.

[9] S. Resch, S. K. Khatamifard, Z. I. Chowdhury, M. Zabihi, Z. Zhao, J.-P. Wang, S. S. Sapatnekar, and U. R. Karpuzcu, "Pimball: Binary neural networks in spintronic memory," *ACM TACO*, vol. 16, no. 4, pp. 41:1–41:26, Oct. 2019.

[10] J.-P. Wang *et al.*, "General Structure for Computational Random Access Memory (CRAM)," 2015, US Patent 9224447 B2.

[11] J.-P. Wang *et al.*, "A pathway to enable exponential scaling for the beyond-cmos era: Invited," in *DAC*, 2017.

[12] J. Wang *et al.*, "Programmable Spintronics Logic Device Based on a Magnetic Tunnel Junction Element," *Applied Physics Letters*, vol. 97, no. 10D509, 2005.

[13] M. Zabihi *et al.*, "Using spin-hall mtjs to build an energy-efficient in-memory computation platform," in *ISQED*, March 2019.

[14] M. Zabihi *et al.*, "In-memory processing on the spintronic cram: From hardware design to application mapping," *IEEE Transactions on Computers*, 2018.