Using Spin-Hall MTJs to Build an Energy-Efficient In-memory Computation Platform

Masoud Zabihi[1], Zhengyang Zhao[1], Mahendra D. C.[2], Zamshed I. Chowdhury[1], Salonik Resch[1], Thomas Peterson[2], Ulya R. Karpuzcu[1], Jian-Ping Wang[1], Sachin S. Sapatnekar[1]
[1]Department of Electrical and Computer Engineering, University of Minnesota, Minneapolis, MN 55455.
[2]School of Physics and Astronomy, University of Minnesota, Minneapolis, MN 55455.
Email: sachin@umn.edu

Abstract

We present the Spin Hall Effect (SHE) Computational Random Access Memory (CRAM) for in-memory computation, incorporating considerations at the device, gate, and functional levels. For two specific applications (2-D convolution and neuromorphic digit recognition), we show that SHE-CRAM is 3× faster and has over 4× lower energy than a prior STT-based CRAM implementation, and is over 2000× faster and at least 130× more energy-efficient than state-of-the-art near-memory processing.

Keywords: Spintronics, In-memory computing, SHE-CRAM, Nonvolatile memory, Neuromorphic computing

1. Introduction

Trends in the computational structure and data footprint of emerging applications prompt the need for a significant departure from traditional CPU-centric computing [1]. First, in the big data era, the cost (in terms of energy and latency) of transporting data from memory to the processor is prohibitive. Communication energy dominates computation energy; even the cleverest latency-hiding techniques cannot conceal their overhead. Second, since general-purpose processors are inefficient for emerging applications, there is a trend towards specialized accelerators, tailored for efficient execution of specific classes of applications. However, even these structures can suffer from memory bottlenecks.

An effective way to overcome these bottlenecks is to embed compute capability into the main memory, allowing distributed processing of data at the source and obviating the need for intensive energy-hungry communication, through

- true in-memory computing uses the memory array to perform computations through simple reconfigurations.
- near-memory computing places computational units at the periphery of memory for fast data access [2,3].

Post-CMOS technologies open the door to new architectures for in-memory computation. The Computational Random Access Memory (CRAM) architecture [4–6] is a true in-memory computing substrate where a memory array is dynamically reconfigured to perform computations. This architecture has been illustrated on the spin transfer torque (STT) magnetic tunnel junction (MTJ): individual MTJs are relatively slow and power-hungry compared to CMOS-based devices, but these drawbacks are compensated by their ability to perform true in-memory processing, which leads to large savings in communication energy to the periphery of the array or beyond, which more than "pay for" STT-MTJ drawbacks.

To further improve CRAM efficiency, this paper uses a novel 3-terminal MTJ, whose write mechanism is based on the spin-Hall effect (SHE). The SHE-MTJ delivers improved speed and energy-efficiency over the traditional 2-terminal STT-MTJ [13], and recent research on novel spin-Hall materials, e.g., sputtered $BiSe_x$ [8], which experimentally demonstrates very high spin-Hall angles, will lead to further gains over today's SHE-MTJs. Moreover, the separation of write and read paths in the 3-terminal SHE-MTJ makes this device more reliable than the STT-MTJ [7].

However, due to differences between the SHE-MTJ and the STT-MTJ (e.g., in the number of terminals and in the operation mechanism), building a SHE-CRAM is more complex than simply replacing STT-MTJs in the STT-CRAM with SHE-MTJs. In this work, we show how the STT-CRAM architecture must be altered for SHE-MTJs, and that these changes require alterations to operation scheduling schemes. We propose a double-write-line array structure for the SHE-CRAM, and present new data placement and scheduling methods for the implementation of computational blocks in the SHE-CRAM. By evaluating computations on representative applications, we show that, in comparison with the STT-based CRAM, the SHE-CRAM demonstrates an overall improvement in latency and energy.

2. SHE-CRAM structure

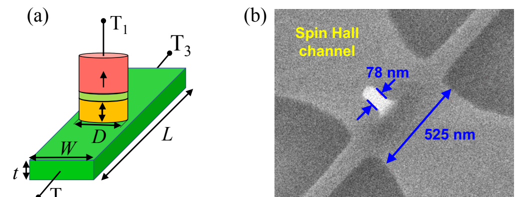

Fig. 1. (a) Schematic of a 3-terminal SHE-MTJ. (b) SEM image taken at 60° from the perpendicular direction showing an MTJ pillar (with resist) fabricated on the W spin Hall channel.

The structure of 3-terminal SHE-MTJ is shown in Fig. 1(a). It is composed of a conventional perpendicular MTJ (pMTJ) stack seated on a spin-Hall (SH) channel, where the free layer of MTJ is directly contacted with the channel. Depending on the free layer orientation, the MTJ can have one of two resistance states: parallel (R_P, logic '0'), and anti-parallel (R_{AP}, logic '1'), where $R_{AP} > R_P$. The free layer orientation is controlled by the direction of the current through the SH channel (between T_2 and T_3 in Fig. 1(a)). Due to the SHE, when the current density exceeds the threshold current density J_{SHE}, the magnetization of the free layer of the MTJ is set according to the current direction [9]. The read operation measures the current between T_1 and T_3, which depends on the MTJ state. Fig. 1(b) shows a fabricated SHE-

MTJ with a diameter of 78 nm over a 525 nm wide SH channel; miniaturization to ~10nm geometries is possible.

Fig. 2 shows the architecture of the SHE-CRAM array, which can operate in memory or logic mode. At the bitcell level, this structure is quite different from the STT-CRAM. The 2T1MTJ bitcell accommodates the 3-terminal SHE-MTJ: each cell has one SHE-MTJ with two terminals gated by access transistors. Each row has two select lines (SLs), ESL and OSL – which select the even and odd columns, respectively – and a logic line (LL); each column has a read and write word line (WLR, WLW). At the array level, the arrangement of wires must accommodate the connections required by the 3-terminal SHE-MTJ. Conventionally, the word line in a memory is drawn as a horizontal line, but we show a rotated array where the word lines run vertically. We make this choice for convenience so that we can compactly show the sequence of computations in later figures.

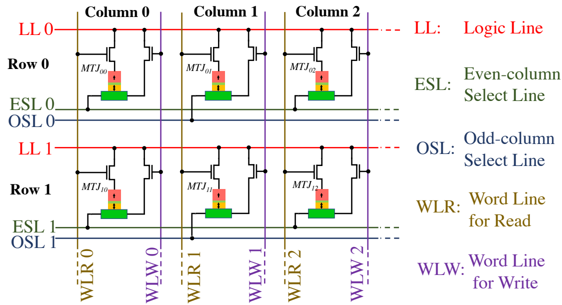

Fig. 2. Overall structure of the SHE-CRAM.

In *memory write mode* (Fig. 3(a)), the transistor connected to WLR is off, WLW is high, turning on the write access transistor, and the SL is either positive or negative (i.e., one of two current directions is applied) depending on whether a 0 or 1 is to be written. This current through the SH channel writes to the MTJ. In *memory read mode* (Fig. 3(b)), WLR is set high to turn the read transistor on. A current is passed through the MTJ between LL and the SL to sense its resistance, i.e., the memory state, by connecting the SL to a sense amplifier.

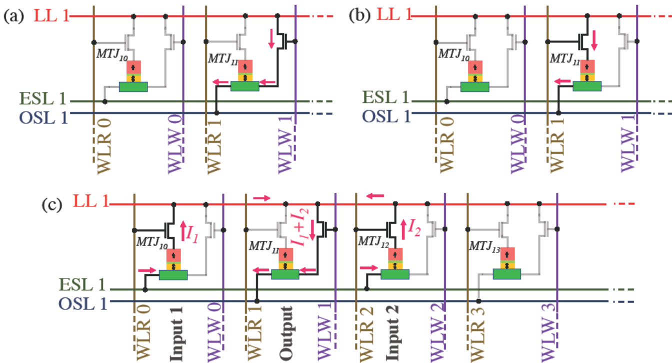

Fig. 3. Current flow during: (a) memory write operation, (b) memory read operation, and (c) logic mode.

In *logic mode* (Fig. 3(c)), a logic operation can be performed between cells in a CRAM row. For input cells, transistors connected to their WLR lines are turned on, and for the output cell, the WLW access transistor is turned on to allow current to flow through the spin-Hall channel. The LL is left floating, the SL for the inputs is set to a specified voltage, and the SL for the output is grounded. This implies that a current, whose value depends on the states of the inputs, passes through the spin-Hall channel of the output MTJ.

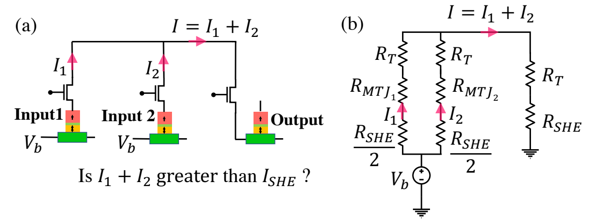

Fig. 4. (a) Performing a logic operation in a row of SHE-CRAM, and (b) the equivalent circuit model.

Fig. 4(a) isolates the part of the array involved in a logic operation with two inputs, and shows its equivalent circuit in Fig. 4(b), where the resistor values depend on the state variables (MTJ resistances) and transistor resistances. Before the computation starts, the output MTJ is initialized to a preset value. By applying bias voltage V_b across the input and output cell SLs, current I_1+I_2 flows through the spin-Hall channel of the output, where the magnitude of each current depends on the input MTJ state (i.e., resistance). If $I_1+I_2 > I_{SHE}$, where I_{SHE} is the SHE threshold current, then depending on the current direction, a value is written to the output MTJ state; otherwise, the preset output state remains intact. As explained in Sec. 3B, by appropriately choosing the voltages and output preset, different logic functions can be implemented.

Note that in the logic mode, all input operands must be in even-numbered columns, and the output must be in an odd-numbered column – or vice versa. This is unlike the STT-CRAM, where no such limitation is necessary, and is a consequence of the 3-terminal structure of the SHE-MTJ.

The three modes – memory read/write and logic mode – are summarized in Table 1.

Table 1: Status of lines and transistors in the SHE-CRAM during memory and logic modes.

Operation		***WLW***	***WLR***	***Transistor connected to WLW***	***Transistor connected to WLR***	***Active ESL***	***Active OSL***	***LL***
Memory Mode	Write	High	Low	ON	OFF	Even column	Odd column	Active
	Read	Low	High	OFF	ON	Even column	Odd column	Active
Logic Mode	Input Cells	Low	High	OFF	ON	Any column	Any column	Float
	Output Cells	High	Low	ON	OFF			

3. SHE-CRAM details

Table 2 defines the parameters of the SHE-MTJ and provides typical values, to be used in the rest of this paper. The dimensions of the SHE-MTJ in Table 2 are appropriately chosen to (a) provide an optimal margin window (see next sections), (b) provide a low I_{SHE}, and (c) avoid unwanted STT switching during logic operations.

A. Device-level design

The specifications of the SHE-MTJ in the SHE-CRAM are shown in Table 2. For our evaluation, the novel sputtered $BiSe_x$ is used as the SH channel, due to its high spin-Hall

Table 2: SHE-MTJ specifications.

Parameters	Value
MTJ type	CoFeB/MgO p-MTJ
Spin Hall channel material	Sputtered $BiSe_x$ [8]
MTJ diameter (D)	10 nm
Spin Hall channel length (L)	30 nm
Spin Hall channel width (W)	15 nm
Spin Hall channel thickness (t)	4 nm
Spin Hall channel sheet resistance (R_{Sheet})	32 kΩ
Spin Hall channel resistance (R_{SHE})	64 kΩ
MTJ RA product	20 Ω·μm²
MTJ TMR ratio	100%
MTJ Parallel resistance (R_P)	253.97 kΩ
MTJ Anti-parallel resistance (R_{AP})	507.94 kΩ
STT critical current density (J_{STT})	5×10^6 A/cm²
SHE threshold current density (J_{SHE})	5×10^6 A/cm² [8][13]
STT threshold current (I_{STT})	3.9 μA
SHE threshold current (I_{SHE})	3 μA
SHE pulse width (t_{SHE})	1 ns [10]
Transistor Resistance (R_T)	1 kΩ

efficiency [8]. Fig. 5 demonstrates the SHE switching of such a structure which requires a very low switching current density. The device is a micron-size Hall bar, which is composed of $BiSe_x$ (5nm) / Ta (0.5 nm) as the SH channel and CoFeB (0.6nm) /Gd (1.2nm) /CoFeB(1.1nm) as the magnetic layer. The easy-axis of the magnetic layer is along the out-of-plane direction. Two magnetization states (up or down, corresponding to the positive or negative Hall resistance) are revealed from the loop in Fig. 5(a). The magnetization can be switched between the two states by injecting a current through the SH channel, as shown in Fig. 5(b). The threshold switching current density J_{SHE} is determined to be 4.4×10^5 A/cm², which is two orders lower than normal spin-Hall structures with metal like Ta, W, or Pt as the SH channel. In Table 2, J_{SHE} is set to 5×10^6 A/cm², based on [13]. Note that although an external magnetic field is applied to assist spin-Hall switching in Fig. 5(b), the external field is not necessary under field-free strategies [11][12]. Note that the choice for L, W, and t is based on an optimization described in Sec. 3C.

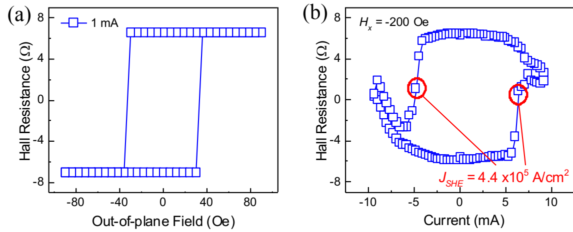

Fig. 5. Demonstration of SHE switching with ultra-low J_{SHE} in a Hall bar device [8]. The SH layer is composed of $BiSe_x$ (5) /Ta (0.5), and the perpendicular magnetic layer is composed of CoFeB (0.6) /Gd (1.2) /CoFeB (1.1) (all thicknesses in nm). (a) The out-of-plane hysteresis loop showing the two magnetization states of the device. (b) SHE switching loop of the device with a very low switching current.

B. Gate-level design

In logic mode, the configuration of the SHE-CRAM into various gate types is controlled by two factors: (a) output preset value, (b) bias voltage, V_b (Fig. 4(a)). By modeling the current path of each gate as in Fig. 4(b), we can determine the conditions for implementing each gate type. The voltage V_b applied across the MTJ interconnections in logic mode falls across ESL and OSL. This voltage, applied across $\left(\frac{R_{SHE}}{2} + R_{MTJ_1} + R_T\right) || \left(\frac{R_{SHE}}{2} + R_{MTJ_2} + R_T\right)$ in series with $(R_{SHE} + R_T)$, is shown in Fig. 4(b). Here, "||" represents the equivalent resistance of resistors in parallel. For the configuration in Fig. 4(b), the current I through the logic line is

$$I = \frac{V_b}{\left[\left(\frac{R_{SHE}}{2}+R_{MTJ_1}+R_T\right)||\left(\frac{R_{SHE}}{2}+R_{MTJ_2}+R_T\right)\right]+R_3}, \quad (1)$$

If V_b is too low, $I < I_{SHE}$, and the current is insufficient to switch the output; if it is too high, $I > I_{SHE}$, and the output is switched regardless of the input state.

The resistance of the MTJ may take on one of two values, R_P or R_{AP}. For conciseness, we define R_1, R_2, and R_3 as:

$$R_1 = \frac{R_{SHE}}{2} + R_P + R_T \quad (2)$$

$$R_2 = \frac{R_{SHE}}{2} + R_{AP} + R_T \quad (3)$$

$$R_3 = R_{SHE} + R_T \quad (4)$$

Consider the case where the gate in Fig. 4(a) is used to implement a 2-input AND gate. For each of the input states (00 through 11), we can calculate the currents flowing through the spin-Hall channel of the output MTJ as:

$$I_{00} = \frac{V_b}{\frac{R_1}{2}+R_3} \quad (5)$$

$$I_{01} = I_{10} = \frac{V_b}{(R_1||R_2)+R_3} \quad (6)$$

$$I_{11} = \frac{V_b}{\frac{R_2}{2}+R_3} \quad (7)$$

For the AND gate the preset output value is 1. For correct AND operation, we must choose V_b appropriately so that $I_{00} > I_{SHE}$ and $I_{01} = I_{10} > I_{SHE}$ (i.e., both cases, the preset output is switched to 0), and $I_{11} < I_{SHE}$ (i.e., the output stays at 1).

Since $R_P < R_{AP}$, $R_1 < R_2$. Therefore, from eq. (5)–(7),

$$I_{11} < I_{01} = I_{10} < I_{00}. \quad (8)$$

Thus, if we chose V_b to be large enough so that $I_{01} = I_{10} > I_{SHE}$, then $I_{00} > I_{SHE}$ must always be true. From eq. (6), the following constraint must be obeyed.

$$V_b > \left((R_1||R_2) + R_3\right)I_{SHE} \quad (9)$$

However, to ensure the correctness of the 11 input case, V_b cannot be too large. Specifically, from eq. (7), it is required that $I_{11} < I_{SHE}$, which leads to the second constraint,

$$V_b < \left(\frac{R_2}{2} + R_3\right)I_{SHE}. \quad (10)$$

These two constraints limit the range of V_b for the AND gate. A NAND gate is identical to the AND, except that a preset value of 0 is used; the range of V_b is identical to the AND.

Similar constraints can be derived for other logic gates, and the bias voltage ranges to implement other gates can be calculated similarly. Table 3 summarizes the bias voltage ranges and the preset value for various gate types using the parameters of Table 2 for the SHE.

For each gate, we can define Noise Margin (NM) of V_b, which is defined as [6]:

$$NM = \frac{V_{max}-V_{min}}{V_{mid}}; \; V_{mid} = \frac{(V_{max}+V_{min})}{2} \quad (11)$$

Table 3: Bias voltage ranges, and output preset value.

Gate	Preset	Closed form formula for bias voltage range	Numerical value (Volt)
NOT	0	$(R_1+R_3)I_{SHE} < V_b < (R_2+R_3)I_{SHE}$	1.065 – 1.827
Buffer	1		
NAND	0	$\left(\frac{R_1R_2}{R_1+R_2}+R_3\right)I_{SHE} < V_b < \left(\frac{R_2}{2}+R_3\right)I_{SHE}$	0.768 – 1.017
AND	1		
NOR	0	$\left(\frac{R_1}{2}+R_3\right)I_{SHE} < V_b < \left(\frac{R_1R_2}{R_1+R_2}+R_3\right)I_{SHE}$	0.636 – 0.768
OR	1		
$\overline{\text{MAJ3}}$	0	$\left(\frac{R_1R_2}{R_1+2R_2}+R_3\right)I_{SHE} < V_b < \left(\frac{R_1R_2}{2R_1+R_2}+R_3\right)I_{SHE}$	0.546 – 0.624
MAJ3	1		
$\overline{\text{MAJ5}}$	0	$\left(\frac{R_1R_2}{2R_1+3R_2}+R_3\right)I_{SHE} < V_b < \left(\frac{R_1R_2}{3R_1+2R_2}+R_3\right)I_{SHE}$	0.418 – 0.446
MAJ5	1		

where V_{max} and V_{min} are, respectively, the upper and lower limits on V_b, and V_{mid} is the midpoint of the bias voltage range. To maximize noise immunity, we chose V_{mid} as the actual applied voltage. The energy, E, dissipated by each gate, is

$$E = V_{mid} I_{SHE} t_{SHE}. \quad (12)$$

Using the values in Table 3, the NM and energy for various SHE-CRAM based logic implementations are computed. We compare the noise margin and energy of logic operations in the STT-CRAM for today's STT-MTJs as reported in [6], and the SHE-CRAM. From Fig. 6, the SHE-CRAM always results in higher noise margins compared to STT-CRAM. This can be attributed to the fact that the resistances (R_{MTJ}) associated with the logic inputs are significantly higher than the resistance R_{SHE} associated with the output, which provides a larger allowable interval for V_b. In contrast, the inputs and outputs for the STT-CRAM are both correspond to MTJ resistances. A comparison of energy-efficiency shows that in all cases, the SHE-CRAM has lower switching current and faster switching time than the STT-CRAM, resulting in better E (Fig. 7).

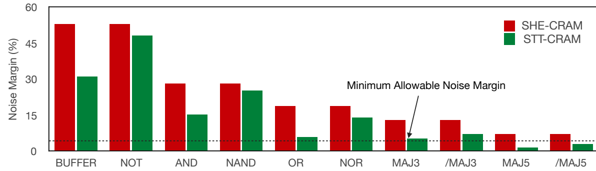

Fig. 6. Comparison of noise margin between gates implemented using STT-CRAM and SHE-CRAM.

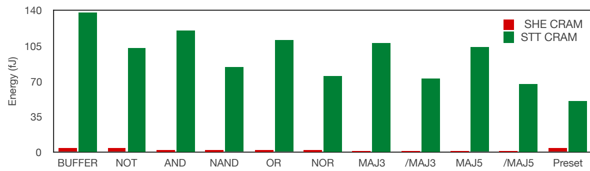

Fig. 7. Comparison of energy between gates implemented using STT-CRAM and SHE-CRAM.

C. *Optimization of spin-Hall channel dimensions*

To further improve device performance, we can optimize the dimensions of spin-Hall channel in the SHE-MTJ device with respect to NM and E. The spin-Hall channel resistance is

$$R_{SHE} = R_{Sheet}(L/W) \quad (13)$$

where $L \geq W$. For a NAND (or AND) gate, from eq. (11),

$$NM_{NAND} = \frac{R_2\left(1-\frac{R_1}{R_1+R_2}\right)}{\frac{R_2}{2}+\frac{R_1R_2}{R_1+R_2}+2R_3}. \quad (14)$$

Similarly, energy for the implementation of a NAND (or AND) gate can be rewritten as:

$$E_{NAND} = (WtJ_{SHE})^2\left(\frac{R_2}{4}+\frac{R_1R_2}{2(R_1+R_2)}+R_3\right)t_{SHE} \quad (15)$$

In Fig. 8, the corresponding noise margin and energy of a NAND (or AND) gate is shown. In Fig. 8(a), by reducing the length to width ratio (*L/W*) of the SH channel, R_{SHE} decreases. In each case, the optimal V_b that maximizes the noise margin NM is found as the midpoint of the allowable interval of V_b. While NM depends on R_P and R_{AP} as well as R_{SHE}, it can be shown (by examining the sensitivity of NM to R_{SHE}) that NM is most sensitive to the reduction in R_{SHE} (details omitted due to space limitations). This causes NM to decrease with increasing (*L/W*). Increasing the channel thickness t reduces R_{sheet}, thus decreasing R_{SHE}: as before, this increases NM.

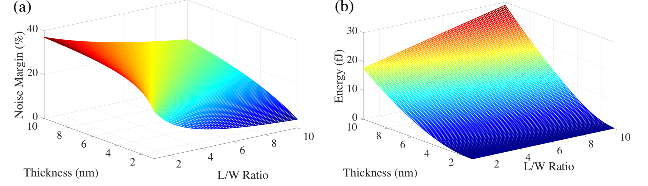

Fig. 8. Impact of SHM geometry on NM and energy.

In Fig. 8(b), by increasing L/W (or t), the energy increases. To maximize noise margin and minimize energy, L/W should be as small as possible (due to fabrication considerations the ratio is considered 2 rather than 1). For the choice of t, a balance between NM and energy must be found. Although a larger thickness increases NM, it increases the energy. As a compromise, based on Fig. 8, we choose a near middle point of t = 4 nm (providing 32% energy improvement with 3% degradation in NM compared to the middle point of 5 nm).

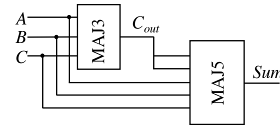

Fig. 9. FA based on majority logic, where $C_{out} = MAJ3(A,B,C)$ and $Sum = MAJ5(A,B,C,C_{out},C_{out})$.

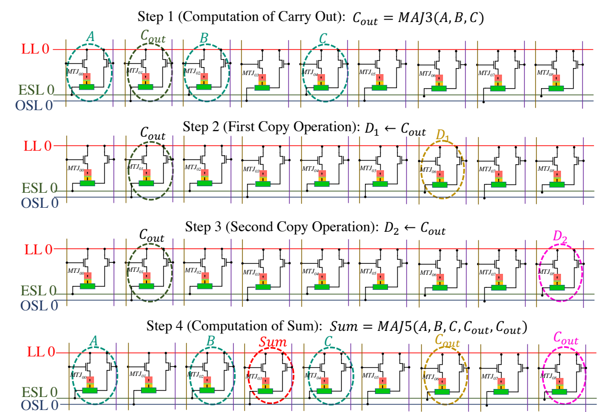

Fig. 10. Four required steps for the implementation of the FA based on majority logic in a row.

D. *Functional-level design*

Full adders: The original STT-CRAM [4] built a NAND based implementation of full adder (FA) using 9 steps. Using majority logic one can implement a FA, as shown in Fig. 9, and this requires only 3 steps [5]. STT-CRAM technology has very limited NM for majority gates; in contrast, the NM in

SHE-CRAM is sufficiently high that majority implementations are realistic. However, SHE-CRAM array in Fig. 2 is limited by the fact that all input operands must be in even columns, and the output must be in an odd column, or vice versa. This affects multi-step operations where some intermediate results, which act as operands for the next step, may be in even columns, while others may be in odd columns. This requires additional steps to move some operands.

Fig. 10 shows that the implementation of a majority logic based FA in a row of the SHE-CRAM requires 4 steps. In step 1, $C_{out} \leftarrow MAJ3(A, B, C)$ is calculated: the inputs are in even columns (0, 2, 4) and the output is in odd column 1. In steps 2 and 3, C_{out} is copied, $D \leftarrow BUFFER(C_{out})$, to two different even-numbered columns (6 and 8). Finally, in step 4, with all operands in even-numbered columns, we compute $Sum \leftarrow MAJ5(A, B, C, C_{out}, C_{out})$.

Note that due to the SHE-CRAM structure, C_{out} computed in step 1 cannot be used directly for computation of Sum and must be copied twice to proper locations at Step 2 and Step 3, meaning that this operation requires 4 steps, unlike the STT-CRAM, which would require 3 steps; however, as stated earlier, SHE-CRAM provides better *NM* than STT-CRAM.

Multibit adders: Using the majority logic based FA, we show the implementation of a 4-bit ripple carry adder (Fig. 11), with the computations scheduled as shown in Fig. 12. At step 1, C_1 is generated in row 0. At step 2, C_1 is transferred to row 1.

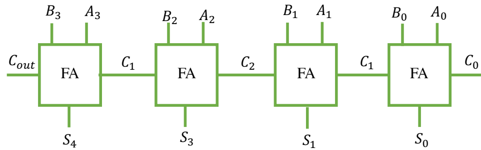

Fig. 11. 4-bit ripple carry adder using 4 FAs.

Similarly, the generated Carrys from the second FA (implemented in row 1) and third FA (implemented in row 2) are transferred to rows 2 and 3 at steps 4 and 6, respectively. Once all Carrys are generated in their corresponding rows, we can copy Carrys twice to proper locations (D_1 to D_8), and then compute Sums (recall that input operands are required to be in all-even or all-odd columns). We transfer the Carry from one row to its adjacent row using inter-row switches (Fig. 13).

Step	1	2	3	4	5	6	7	8	9	10
Row 0	C_1							D_1	D_5	S_0
Row 1		C_1	C_2					D_2	D_6	S_1
Row 2				C_2	C_3			D_3	D_7	S_2
Row 3						C_3	C_{out}	D_4	D_8	S_3

Fig. 12. Scheduling table for a 4-bit ripple carry adder.

Fig. 14 shows the data layout of the 4-bit ripple carry adder at the end of step 10. The location of each cell can be specified by (Row number, Column number). Initially, 4-bit numbers $A_3A_2A_1A_0$ and $B_3B_2B_1B_0$ are stored in (0 to 3, 0) and (0 to 3, 2), respectively, and Carry-in C_0 is stored in (0, 4). At step 1, C_1 is calculated in (0, 1), and at step 2 it is transferred to (1, 4).

Similarly, C_2 and C_3 are generated and transferred between step 3 and 6. At step 7, C_{out} is calculated in (3, 1). The content of (0 to 3, 1) is then copied to (0 to 3, 6) and (0 to 3, 8) based on the abovementioned schedule. Finally, at step 10, S_0, S_1, S_2, and S_3 are calculated in (0 to 3, 3).

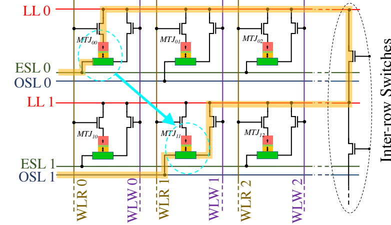

Fig. 13. Inter-row transfer between cells in two adjacent rows (shown by the blue arrow) using switches inserted between rows. The current path is highlighted in orange.

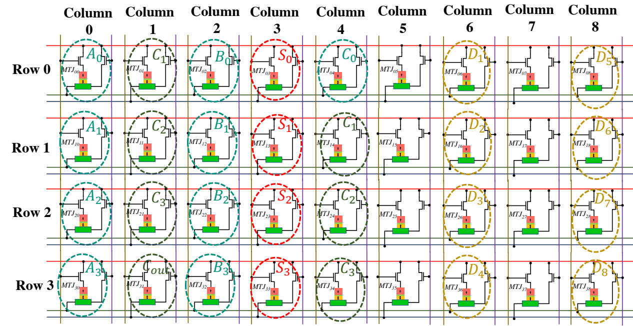

Fig. 14. Data layout of the SHE-CRAM, implementing the 4-bit ripple carry adder, at the end of step 10.

The execution time is determined by counting the number of steps and multiplying them by the logic delay for a majority function, which is dominated by the MTJ switching time. The energy is calculated by considering numbers of gates and their corresponding energy (Table 4). The dominant energy component of this implementation is related to the output presetting of gates (see Fig. 15).

Table 4: Counts of gates and their corresponding energy values for the calculation of the energy required for the implementation of the 4-bit ripple carry adder.

	BUFFER	/MAJ3	/MAJ5	PRESET	Total Energy (fJ)
Number of gates	11	4	4	19	
Energy/gate (fJ)	4.34	1.76	1.30	3.74	
Total Energy (fJ)	47.74	7.04	5.12	71.06	130.96

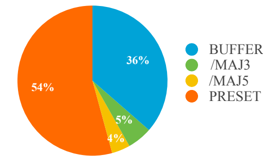

Fig. 15. Energy distribution for the implementation of 4-bit ripple carry adder using SHE-CRAM. Energy for preset is the dominant component.

More complex building blocks: Similar principles can be used to implement structures such as multipliers and dot products, which can be integrated to implement applications using SHE-CRAM; details are omitted due to space limitations.

4. Application-level analysis

To benchmark SHE-CRAM performance at the application level, we study its performance when it is deployed on two

applications that were analyzed for the STT-CRAM in [6]: (a) 2-D convolution, where a 512×512 image is filtered using a 3×3 filter, and (b) neuromorphic digit recognition using 10K testing images in the MNIST database.

For both applications, we compare the energy and execution time using SHE-CRAM, STT-CRAM, and a near-memory processing (NMP) system (representative of current state-of-the-art). The NMP system places a processor at the periphery of a memory, and is superior to a system in which data is fetched from memory to processor (or coprocessor) [1][14][15]. Also, note that in evaluations of STT-CRAM and SHE-CRAM, the effect of peripheral circuitry is considered.

The results of the comparison are presented in Table 5. SHE-CRAM outperforms STT-CRAM in both execution time and energy, and both SHE-CRAM and STT-CRAM beat the NMP system in term of energy and execution time. In both applications, SHE-CRAM is at least 4× more energy efficient, and 3× faster than STT-CRAM. For 2-D convolution, SHE-CRAM is over 2000× faster, and 130× more energy-efficient than an NMP system. The corresponding numbers for the neuromorphic application are over 4000× and 190×, respectively.

The improvements in SHE-CRAM over the STT-CRAM can be attributed to the speed and energy-efficiency of the SHE-MTJ device. Note that the ratio of speed improvement is almost the same as the 3× improvement of the SHE-MTJ over the STT-MTJ, but the energy improvement is less than the ratio of STT-MTJ to SHE-MTJ switching energy, primarily because of the significant energy overhead of the peripheral driver circuitry of the memory array. Using larger subarrays in the memory can provide up to 25% energy improvements, while degrading the speedup from 3× to just over 2×.

The superiority of both CRAMs over the NMP system can be attributed to the low memory access time of the in-memory computing paradigm, and high levels of available parallelism in CRAM. In contrast, in the NMP system, the energy and execution time consists of two components: (a) fetching data from the memory unit, and (b) processing data in processor units. We can have maximum parallelism in a NMP systems by using multiple processor units and latency hiding techniques, but energy and execution time cost of fetching data from the memory are a severe bottleneck. This bottleneck does not exist in the CRAM due to data locality.

5. Conclusion

SHE-CRAM leverages the speed and efficiency of the 3-terminal SHE device, and we demonstrate a new in-memory computing architecture using this device. We propose a design method which contains consideration in device, gate, and functional levels. At the device level, the 3-terminal SHE-MTJ integrated with highly efficient spin-Hall material is served as the unit cell of CRAM. At the gate level, we show that energy and noise margin of implementation of a gate using SHE-CRAM is always superior to those of STT-CRAM. Moreover, we optimize the dimensions of the spin-Hall channel with respect to the noise margin and the implementation energy of a gate. At the functional level, we illustrate how a FA can be implemented in SHE-CRAM, principles that can be extended to more complex structures. Finally, at the application level, we have analyzed the SHE-CRAM performance for 2-D convolution and neuromorphic digit recognition. We show a large improvement in speed and energy over both the STT-CRAM and a NMP system.

Table 5: Comparison between execution time and energy of NMP, SHE-CRAM, and STT-CRAM. The CMOS-based NMP data is based on the calculations in [6].

Application	Parameters	NMP	STT-CRAM	SHE-CRAM
2-D Convolution	Execution Time	144.4 μs	231 ns	63 ns
	Energy	388.6 μJ	16.5 μJ	2.9 μJ
Digit Recognition	Execution Time	1.96 ms	1105 ns	408 ns
	Energy	2.57 mJ	63.8 μJ	13.5 μJ

6. References

[1] S. W. Keckler, *et al.*, “GPUs and the future of parallel computing,” *IEEE Micro*, vol. 31, pp. 7–17, 11 2011.

[2] J. T. Pawlowski, "Hybrid memory cube (HMC),” in *Proceedings of the IEEE Hot Chips Symposium*, 2011.

[3] J. Macri, “AMD’s next generation GPU and high bandwidth memory architecture: FURY,” in *Proceedings of the IEEE Hot Chips Symposium*, 2015.

[4] J.-P. Wang and J. D. Harms, “General structure for computational random access memory (CRAM),” US Patent 9224447 B2, Dec. 29 2015.

[5] Z. Chowdhury, *et al.*, “Efficient in-memory processing using spintronics,” *IEEE Computer Architecture Letters*, vol. 17, pp. 42-46, 2017

[6] M. Zabihi, *et al.*, “In-memory processing on the spintronic CRAM: From hardware design to application mapping,” *IEEE Transactions on Computers*, Early Access, published on 20 July 2018.

[7] G. Prenat, *et al.*, “Ultra-fast and high-reliability SOT-MRAM: From cache replacement to normally- off computing,” *IEEE Transactions on Multi-Scale Computing Systems*, vol. 2, pp. 49–60, 2016.

[8] Mahendra D. C. *et al.,* “Room-temperature high spin-orbit torque due to quantum confinement in sputtered $Bi_xSe_{(1-x)}$ films,” *Nature Materials*, vol. 17, pp. 800-807, 2018.

[9] L. Liu, *et al.*, “Spin-torque switching with the giant spin Hall effect of tantalum,” *Science*, vol. 336, Issue 6081, pp. 555-558, 2012.

[10] K. Garello *et al.*, “Ultrafast magnetization switching by spin–orbit torques,” *Applied Physics Letters*, vol. 105, 212402, 2014.

[11] S. Fukami, *et al.*, “Magnetization switching by spin–orbit torque in an antiferromagnet–ferromagnet bilayer system,” *Nature Materials*, vol. 15, pp. 535–541, 2016.

[12] Z. Zhao, *et al.*, “External-field-free spin Hall switching of perpendicular magnetic nanopillar with a dipole-coupled composite structure”, arXiv:1603.09624, 2017.

[13] C. Zhang, *et al.*, “Spin-orbit torque induced magnetization switching in nano-scale Ta/CoFeB/MgO,” *Applied Physics Letters*, vol 107, 012401, 2015.

[14] M. Horowitz, “Computing’s energy problem (and what we can do about it),” in *Proceedings of the IEEE International Solid-State Circuits Conference*, pp. 10–14, 2014.

[15] J. Jeddeloh and B. Keeth, “Hybrid memory cube new DRAM architecture increases density and performance,” in *Proceedings of the IEEE International Symposium on VLSI Technology*, pp. 87–88, 2012.