HieIM: Highly Flexible In-Memory Computing using STT MRAM

Farhana Parveen, Zhezhi He, Shaahin Angizi, Deliang Fan*
Department of Electrical and Computer Engineering, University of Central Florida, Orlando, FL-32816, USA.
*Email: dfan@ucf.edu

Abstract— In this paper we propose a Highly Flexible In-Memory (HieIM) computing platform using STT MRAM, which can be leveraged to implement Boolean logic functions without sacrificing memory functionality. It could pre-process data within memory to further reduce power hungry long distance communication between memory and processing units as in Von-Neumann computing system. HieIM can implement all the Boolean logic functions (AND/NAND, OR/NOR, XOR/XNOR) between any two cells in the same memory array, thus overcoming the 'operand locality' problem in contemporary in-memory computing platform designs. To investigate the performance of HieIM, we test in-memory bulk bit-wise Boolean logic operations using different vector datasets, which shows $\sim 8\times$ energy saving and $\sim 5\times$ speedup compared to recent DRAM based in-memory computing platform. We further implement an in-memory data encryption engine design based on HieIM as another case study. With AES algorithm, it shows 51.5% and 68.9% lower energy consumption compared to CMOS-ASIC and CMOL based implementations, respectively.

I. Introduction

'Memory Wall' [1] bottleneck hinders the effectiveness of traditional Von-Neumann architecture specially for data intensive applications [2]. Due to separation of memory and processing units, massive data transfer between processor and memory creates several critical limitations, e.g., long memory access latency, significant congestion at I/Os, limited memory bandwidth and huge leakage power consumption. Hence, the need for efficient computing platform to support memory-oriented processing is the cynosure of recent big-data oriented research. To meet these demands, in-memory computing scheme [3–9] are proposed to calculate the intermediate results by pre-processing the data within memory before sending to the main processor, thus greatly reducing power hungry off-chip massive data flow.

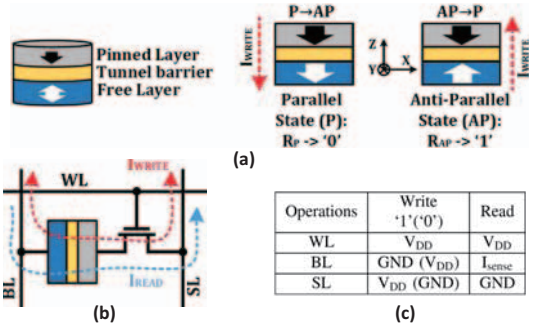
DRAM based in-memory data processing has been proposed in [5]. However, several inherent limitations of DRAM push them back from being an efficient in-memory computing solution. Among them- volatility, need for repeated refresh, destructive read/compute operation, slow charging and discharging etc. are the dominant ones. Additionally, incorporating logic functionality within memory increases complexity and cost of DRAM chip. In view of these, more efficient, compact and non-volatile in-memory computing platform that is capable of simultaneously offering density optimized memory and performance optimized computing facility, is one of the most important focus points for modern computing platform design.

Non-volatile Resistive RAM (ReRAM) [10, 11] and Phase Change RAM (PCRAM) [12, 13] offer more packing density ($\sim 2-4\times$) than DRAM, and hence appear to be competitive alternatives to DRAM. However, they have much slower and more power hungry operations than DRAM [11, 13]. Furthermore, PCRAMs wear out with each write [14].

Nowadays, Spin-Transfer Torque Magnetic RAM (STT-MRAM) [15] has come out to be a better alternative to DRAM technologies. Though it may not be much competitive to DRAM in terms of packing density and write performance, STT-MRAM has comparable read performance (latency and energy) with DRAM. More importantly, STT-MRAM provides two major advantages over DRAM: non-volatility and decoupled sensing and buffering [12]. While comparing with ReRAM and PCRAM, STT-MRAM shows better read/write performance and better endurance. Therefore, modern memory oriented research is leaning toward non-volatile memories, specially STT-MRAM. In early 2016, Everspin announced 256Mb STT-MRAM chips based on MTJ with interface speed similar to DRAM and was planning 1Gb chips in 2017 [16]. Toshiba and SK Hynix co-developed a 4-Gbit STT-MRAM chip prototype and demonstrated at IEDM 2017 [17]. Hence, in-memory computing using STT-MRAM without sacrificing memory capacity can pave a novel way to efficient computing paradigm. Recently, several research works proposed in-memory computing architectures [6, 8, 9] using STT-MRAM. However, these designs have a few limitations. First, none of these designs can perform computing (Boolean logic functions) between any two bits irrespective of their position in the memory array. They need to be stored either in the same bit-line or word-line, which is defined as *operand locality* in this work. Second, most of these works [8, 9] can implement only AND/OR logic function.

In this paper, we have proposed an efficient in-memory computing architecture using STT-MRAM, which can perform complete Boolean logic functions (AND/NAND, OR/NOR, XOR/XNOR) between any two bits stored in the same memory array, thus solving the 'operand locality' issue [18] of contemporary DRAM [5], SRAM [18] and other non-volatile memory based in-memory computing designs [7, 8]. To investigate the performance of our proposed design, we perform in-memory bulk bit-wise vector operation using different vector dataset [6], which shows $\sim 8\times$ energy saving and $\sim 5\times$ speedup compared to that in DRAM based in-memory computing platform [5]. We further employ the proposed HieIM to implement an in-memory data encryption engine design. With AES algo-

978-1-5090-0602-1/18/$31.00 ©2018 IEEE



Operations	Write '1'('0')	Read
WL	V_{DD}	V_{DD}
BL	GND (V_{DD})	I_{sense}
SL	V_{DD} (GND)	GND

Fig. 1. (a) Device structure of conventional magnetic tunnel junction in parallel- and anti-parallel states, with current- induced spin-transfer torque switching scheme, (b) Bit-cell schematic of 1T1R STT-MRAM bitcell, (c) Biasing conditions for STT-MRAM operations.

rithm, it shows 51.5% and 68.9% lower energy consumption compared to CMOS-ASIC and CMOL based implementations, respectively.

II. Spin-Transfer Torque Random-Access Memory

A typical Magnetic Tunnel Junction (MTJ) structure (Fig. 1a) consists of two ferromagnetic layers with a tunnel barrier sandwiched between them. Due to the Tunnelling Magneto Resistance (TMR) effect [19], the resistance of MTJ is high (low) when the magnetizations of two ferromagnetic layers are in anti-parallel (parallel) state. Thus, the data are stored as the magnetization direction in the free layer (FL), which could be flipped through current induced Spin-Transfer Torque (STT) [20]. Note that, MTJ with Perpendicular Magnetic Anisotropy (PMA) is used in this work.

1 Transistor 1 MTJ (1T1R) bit-cell is widely used in typical STT-MRAM design, as depicted in Fig.1b, which is controlled by Bit-Line (BL), Word-Line (WL) and Source Line (SL). The memory read and write biasing conditions are presented in Fig. 1c. For both memory read and write operations, WL is enabled, which activates the access transistor. Then, a voltage difference $-V_{DD}$ or $+V_{DD}$ is applied between the BL and SL, in order to write '1' or '0' respectively. For memory read, a sensing current (I_{sense}) is applied on the BL that generates a sensing voltage, which can be detected by a sense amplifier.

We have used the Landau-Lifshitz-Gilbert (LLG) equation [21] to model the STT-MRAM bitcell (MTJ) for circuit level simulation. The dimensions and parameters used for simulation are listed in Table I.

III. Proposed STT-MRAM based In-Memory Computing Architecture

The proposed STT-MRAM based highly flexible in-memory computing architecture can perform dual mode operation: memory mode and computing mode. Fig.2 shows the architecture of the proposed in-memory computing platform with a 3×3 STT-MRAM memory array. Here, each memory cell is designed using the 1T1R STT-MRAM bitcell structure described in section II. Each cell is associated with the Word Lines (WL), Bit Lines (BL) and Source Lines (SL). The WLs and BLs are externally controlled by the row decoder and column decoder respectively. Additionally, voltage drivers (VD) are connected to each BL and SL.

TABLE I
Simulation Parameters

Parameter	Value
Free layer dimension,$(W \times L \times t)_{FM}$	$65 \times 65 \times 2\ nm^3$
Polarization factor, P	0.4
Gilbert Damping Factor, α	0.007
Saturation Magnetization, M_s	$850\ kA/m$
Oxide thickness, t_{ox}	$1.2\ nm$
RA product, RA_p / TMR	$10.58\ \Omega \cdot \mu m^2$ / 171.2%
Supply voltage	$1\ V$
CMOS technology	$45\ nm$
STT-MRAM cell area	$48F^2$
Access transistor width	9F
Cell aspect Ratio	1.34

Fig. 2a shows the conventional STT-MRAM array with sense amplifier (SA). The BLs of the array are connected to SA to generate memory read output. We propose a new sensing circuit design using 5 Terminal (5T) Magnetic Domain Wall Motion (DWM) device [22] as an extension to the sense amplifier of STT-MRAM array to obtain the computing mode operation results (Fig. 2b). Here, both the memory mode and computing mode operations are described.

A. Memory Mode:

In memory mode, data are written into or read from the memory cells as described in section II and Fig. 1b.

Memory Write: To write data in a memory cell, the corresponding WL is activated using the row decoder. Then appropriate voltage difference (Fig. 1c) is applied to the corresponding BL and SL using the voltage drivers connected to them. The write current path through one STT-MRAM bitcell is shown in Fig. 1b.

Memory Read: To read data from a memory cell, the corresponding WL is activated using the row decoder and the corresponding BL is connected to the sense amplifier (SA) using the column decoder. Here the sense amplifier circuitry using StrongARM Latch [23] is shown in Fig. 2c. The read current path through one STT-MRAM bitcell is shown in Fig. 1b.

B. Computing Mode:

We propose a sensing circuit design using 5T DWM device [22], as an extension to the sense amplifier (SA) of memory array, to implement complete Boolean logic functions between any two cells in the memory array. The proposed sense amplifier extension circuitry is shown in Fig. 2b, which contains one 5T DWM device, one differential latch and 4 keys. All transistors associated with the 5T DWM device are designed to work in deep triode region by applying ΔV (100mV) across the drain and source ($V_{DS} \approx 100mV$) terminals. Hence, it will lead to ultra-small voltage drop and thus ultra-small power consumption. Fig. 3a shows transient micromagnetic simulation of the DWM strip with lateral currents of $\sim 48\mu A$ and $\sim 24\mu A$ from W- to W+ (electron flow is from W+ to W-). Please note, three notches have been inserted in the DWM strip to stabilize the DW position at W-, Mid and W+ positions inside it. It can be seen that the domain wall (DW) could be moved from W+ to W- (or middle) in the DWM nano-strip within $\sim 1ns$ by applying $\sim 48\mu A$ (or $\sim 24\mu A$) current from W- to W+.

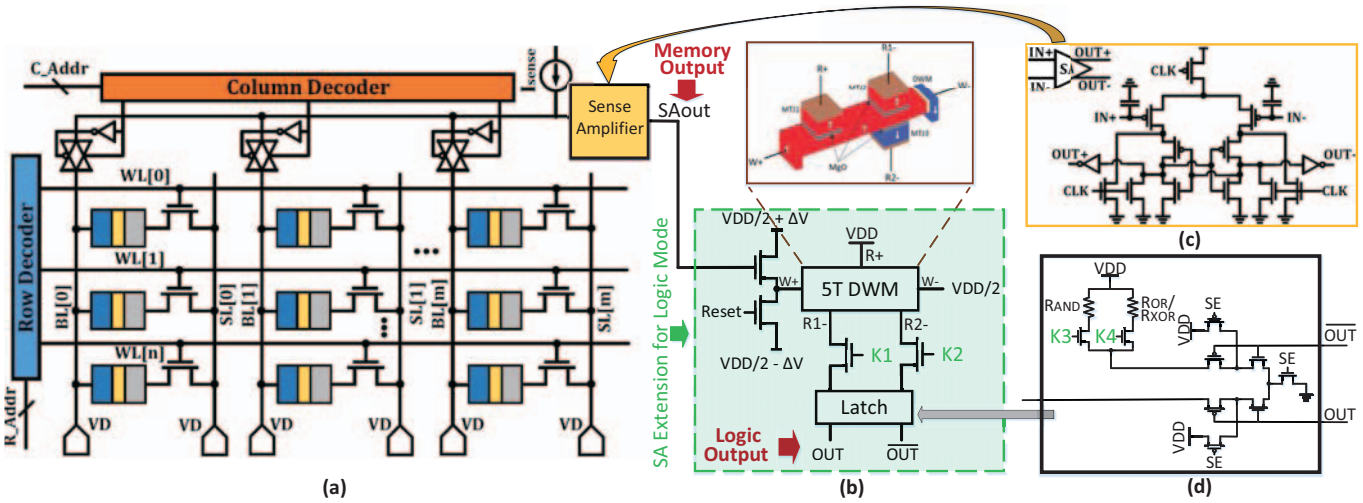


Fig. 2. (a) Proposed highly flexible dual mode in-memory computing architecture (HieIM), (b) Proposed sensing scheme for in-memory logic implementation, (c) Memory sense amplifier (StrongARM latch), (d) Differential Latch used in the proposed sensing circuit using the 5T device.

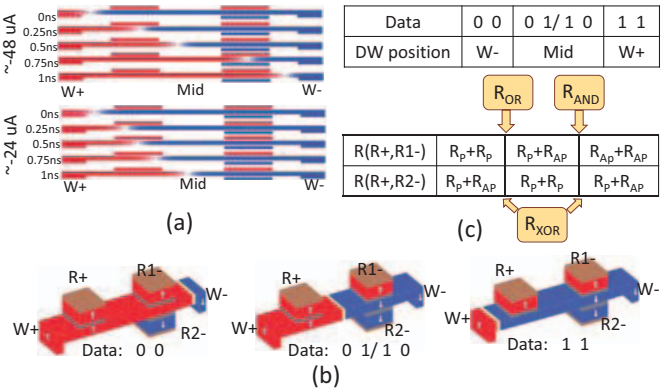


Fig. 3. (a)Micromagnetic simulation for flowing current from W+ to W-, (b) DW position within the 5T DWM device for different data combinations, (c) Resistance variation of the sensing current path through the 5T DWM device and selection of Reference MTJ value.

For a complete Boolean operation, the SA extension needs three subsequent stages- Reset, Compute and Sense. In Reset stage (Reset=1), the reset transistor (Fig. 2b) is turned on for 1ns. Then, a current of $\sim 48\mu A$ flows from W- to W+ terminals, which sets the Domain Wall (DW) back to its initial position at W- side.

In compute stage, for logic AND/OR/XOR between any two cells in the memory array, two operands stored in the memory array are read in two consecutive cycles using the sense amplifier (SA). The input transistor size of 5T DWM device is tuned in such a way that, if the SA output is '1', $\sim 24\mu A$ current will flow from W+ to W-. As per the micromagnetic simulations shown in Fig. 3a, this will move DW to the middle pinning site. On the other hand, if the SA output is '0', then no current will be injected to 5T DWM device, and hence the DW position will not change. Fig. 3b shows the final DW positions after loading two operands.

In sense stage, a small sensing current is injected through 5T DWM device from R+ to R1- (setting K1=1, K2=0) or from R+ to R2- (setting K1=0, K2=1) terminals based on required logic implementation. A differential latch (Fig. 2d) compares the sensing resistance of 5T DWM device with either of the two reference MTJs (R_{AND} or R_{OR}/R_{XOR}) by setting appropriate K3 and K4. Please note, R_{OR}=R_{XOR}. The value of R_{AND} (or R_{OR}, R_{XOR}) is in between $R_P + R_{AP}$ and $2R_{AP}$ (or $2R_P$). Table II shows the detailed logical breakdown of the sensing functionality of the proposed SA extension for AND/OR/XOR logic operations. Note that, NAND/NOR/XNOR could also be readily achieved at the differential output. Detailed implementation of every Boolean logic functions is described below.

TABLE II
IMPLEMENTATION OF COMPUTING MODE (AND/OR/XOR)

Data		1 1	0 1/ 1 0	0 0
DW Position		W+	Mid	W-
AND Keys: K1-K4=1010	R(R+,R1-)	$R_{AP}+R_{AP}$	R_P+R_{AP}	R_P+R_P
	compare to R_{AND}*	>	<	<
	OUT	1	0	0
OR Keys: K1-K4=1001	R(R+,R1-)	$R_{AP}+R_{AP}$	R_P+R_{AP}	R_P+R_P
	compare to R_{OR}**	>	>	<
	OUT	1	1	0
XOR Keys: K1-K4=0101	R(R+,R2-)	$R_{AP}+R_P$	R_P+R_P	R_P+R_{AP}
	compare to R_{XOR}**	>	<	>
	OUTbar	0	1	0

* R_{AND} = Between $2R_{AP}$ and $R_P + R_{AP}$
** R_{OR}/R_{XOR} = Between $2R_P$ and $R_P + R_{AP}$

AND/OR: For logic AND (OR) between any two cells in the memory array, sensing current is flowed from R+ to R1- terminal of the 5T DWM device using K1-K2 = 1-0. The reference MTJ, R_{AND} (or R_{OR}) is selected by setting K3-K4 = 1-0 (or 0-1). Hence, as shown in Table II, the latch output (OUT in Fig. 2d) will provide the logic AND (OR) operation result. Please note, logic NAND (NOR) operation result can be obtained from the differential output ($\overline{OUT}$) of the latch.

XOR: For logic XOR function between any two cells in the memory array, sensing current is flowed from R+ to R2- terminal of the 5T DWM device using K1-K2 = 0-1. Now, the reference MTJ-R_{XOR} is selected by setting K3-K4 = 0-1. Hence, per Table II, the differential latch output ($\overline{OUT}$ in Fig. 2d)

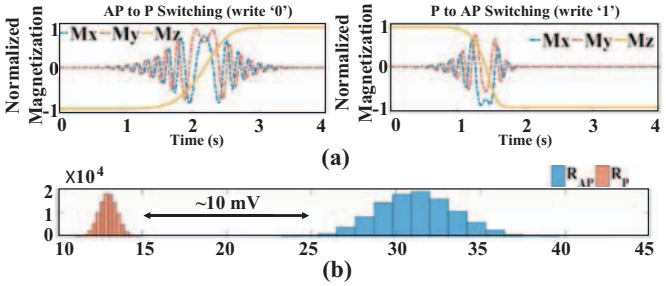


Fig. 4. (a) The transient plot of the normalized magnetization switching in x-, y- and z-axis, with provided STT-MRAM write scheme, (b) The Monte-Carlo simulation result of sense voltage (V_{sense}) distribution for (top) conventional memory read op- eration with single STT-MRAM.

will provide the logic XOR operation result. Please note, logic XNOR operation result can be obtained from the output (OUT) of the Latch.

IV. Memory Mode Performance Evaluation

We have performed the circuit level simulation of STT-MRAM in Cadence Spectre with 45nm NCSU PDK [24] as the CMOS library. The MTJ resistive model is obtained from the modeling approach described in II with device dimensions listed in Table I. Fig. 4a shows the normalized magnetization dynamics of free layer in x-, y- and z-axis, when performing the STT-MRAM write scheme that is described in Fig. 1c.

To evaluate the memory mode performance of HieIM at system level, we have configured the memory chip organization by dividing it into multiple Banks (Bank organization: total-4×4, active-1×1) consisting of multiple Mats (Mat organization: total-2×2, active-1×1). Each Mat includes multiple sub-arrays (sub-array size: 1024×512) organized in a H-tree routing manner. For the simulation, we have employed modified self-consistent NVSim [25] along with an in-house developed C++ code to verify the system level performance and to report the average latency, dynamic energy, leakage power and area. Table III tabulates and compares the performance of our design with two different memory arrays employed for in-memory processing (i.e. SRAM and DRAM) for a sample memory capacity of 4MB in 45nm process node.

According to Table III, the proposed HieIM memory model shows the least read dynamic energy in comparison to other designs. Besides, HieIM reduces the total leakage power compared to SRAM. Although, the proposed HieIM shows longer average latency compared to SRAM due to the longer write latency of magnetic memory storage. Moreover, the area overhead of the proposed STT-MRAM memory model is 24.86% more than DRAM but still 44.32% less than SRAM design. It is noteworthy that the first and foremost benefit of spintronic memories, compared to SRAM, is their non-volatility with al-

TABLE III
SRAM, DRAM and Proposed HieIM Memory Model Validation and Comparison for a Sample 4MB Memory

	SRAM		DRAM		HieIM	
Metrics	Write	Read	Write	Read	Write	Read
Average Latency (ns)	1.53		2.7		1.93	
Dynamic Energy (pJ)	297.46	312.56	967	1483	354.89	248.95
Leakage Power (mW)	5258		185.5		730.966	
Area (mm^2)	10.544		4.702		5.871	

TABLE IV
Performance of 2-input AND Gates

Performance	HieIM	DW Racetrack based [27]	MTJ based [28]	DW Racetrack based [29]	CMOS
Energy (fJ)	23.5	67.72	125.85	504.36	6.69
Speed (ns)	4	1.12	1.18	2.14	0.062

TABLE V
Performance Evaluation of FA Cells

Parameters	HieIM	HSM [30]	LPM [30]	Diode-GSHE [31]	CMOS [31]
Average Power (μW)	91.93	1354	85	15.6	49.4
Delay (ps)	26,000	269	877	10,000	1000

most 10 years' retention time [7].

Furthermore, in order to validate the variation tolerance of sense circuit, we have performed Monte-Carlo simulation with 100000 trials. A $\sigma = 5\%$ variation is added to the Resistance-Area product (RA_P) and a $\sigma = 10\%$ process variation is added to the TMR as done in [26]. The simulation result of sense voltage (V_{sense}) distributions in Fig. 4b shows the sense margin for conventional memory read. Such sense margin could be improved by increasing the sense current, but with a sacrifice of read operation energy efficiency.

V. Computing Mode Performance Evaluation

To evaluate the performance of HieIM for performing logic operations, AND/OR/XOR logic computation energy and latency have been measured. Comparison result between recent in-memory non-volatile Boolean AND gates and CMOS counterpart is shown in Table IV. It is seen that, in-memory AND operation can be carried out using our proposed design with 65.3% and 81.32% lower energy consumption than Domain-Wall (DW) Racetrack based [27] and MTJ based [28] in-memory non-volatile AND gate implementations, respectively. However, HieIM requires longer latency to compute the logic result than other designs [27–29]. However, this comes as a reasonable trade-off to the fact that, HieIM can implement all the Boolean logic functions (AND/OR/XOR) between any two cells within memory, overcoming operand locality issue.

We further investigate the performance of in-memory logic operations of HieIM by implementing in-memory Full Adder (FA). Comparison of non-in-memory CMOS based, diode-GSHE based [31] and Domain Wall nanomagnet based High-Speed mode (HSM) and Low-Power mode (LPM) [30] FA designs with that of HieIM is shown in Table V. Please note, full adder using HieIM is implemented using two half adders, whereas a half adder is designed using AND and XOR logic functions. The average power of FA using HieIM is comparable to that of LPM based FA design [30]. However, HieIM requires longer delay due to the read-and-write-back overhead of the intermediate results as a reasonable trade-off of the aforementioned in-memory operation.

Pros and Cons analysis of HieIM: As a critical assessment of our proposed highly flexible in-memory computing platform, following pros and cons can be identified-

Pros: HieIM 1) can be used both as a density-optimized non-volatile memory and performance-optimized in-memory computing platform; 2) can reduce power hungry long distance data communication between processor and memory by pre-processing raw data; 3) can perform any two input logic function (AND/OR/XOR) between any two bit-cells (i.e., operands) stored in the memory array, overcoming the operand locality issue [18] as in DRAM, SRAM and other recent non-volatile memory based in-memory computing platforms.

Cons: HieIM requires multiple cycles (4 cycles) to complete one Boolean logic computation, which is slower than other in-memory computing platforms while working in the computing mode.

VI. Case Studies

To evaluate the performance of the proposed STT-MRAM based in-memory computing platform- HieIM, two applications have been studied: 1) In-memory bulk bit-wise Boolean vector logic (AND/OR) operation and 2) In-memory data encryption using Advanced Encryption Standard (AES) [32].

A. In-memory Bulk Bitwise Boolean Vector Logic Operation:

To implement an in-memory bitwise vector computing platform, four different vector datasets [6] have been used. Here, a dataset '19-16-1s' refers to a vector dataset with vector length= 2^{19}, number of vectors= 2^{16}, and AND/OR operation is done between 2^1 rows, where 's' means sequentially. As our proposed design can perform logic operation between any two cells in the same sub-array, there is no restriction on data mapping. In this work, the data from two vectors have been mapped on two consecutive rows of a memory array (Fig. 5a).

Performance Evaluation: The performance of bulk bit-wise vector operation using our proposed architecture has been evaluated using similar simulation framework as described in section IV. Here, each compute (AND/OR) operation has been carried out using 4 consecutive clock cycles (1ns each) as described in section III. Performance of in-memory computing platform using DRAM [5] has also been evaluated using an in-house developed SPICE simulation platform incorporating $6F^2$ DRAM cell structure with 16fF cell capacitance [33] using 45nm technology node. Here, a complete compute operation between two bits needs three consecutive operation cycles-

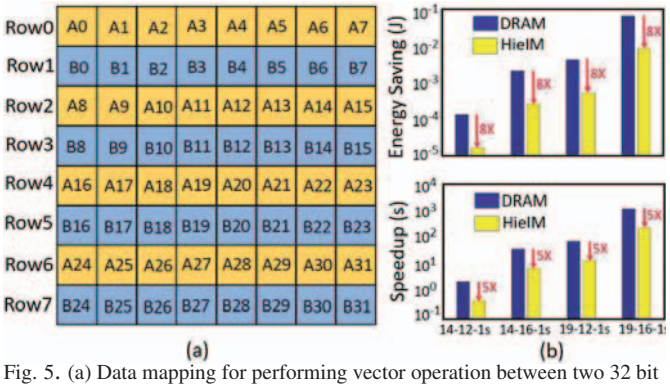


Fig. 5. (a) Data mapping for performing vector operation between two 32 bit vectors using an 8×8 STT-MRAM array (b) Energy saving and speedup for bulk-bitwise vector operation for different vector datasets compared to DRAM based in-memory computing platform.

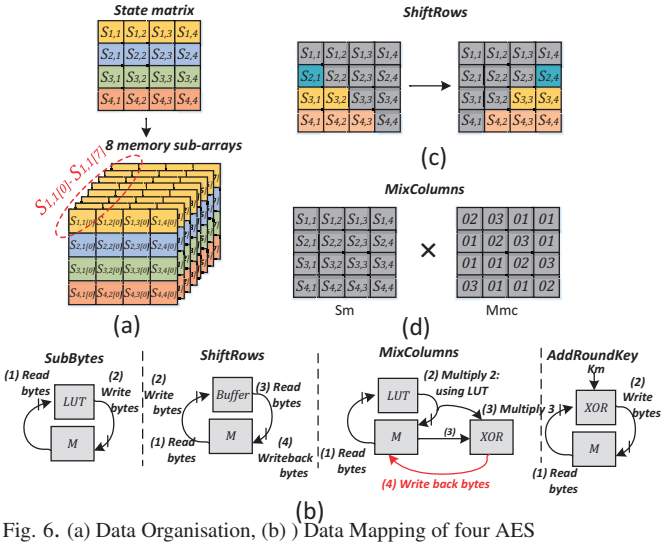


Fig. 6. (a) Data Organisation, (b)) Data Mapping of four AES transformations to the proposed in-memory processing platform, (c) schematic representation of ShiftRows transformation, (d) schematic representation of MixColumn transformation.

precharge, access and sense [5, 34], assuming data were already written in the memory array.

Fig. 5b shows energy saving and speedup of implementing in-memory bulk bitwise Boolean logic operation (AND/OR) for different vector datasets compared to DRAM based in-memory computing platform [5]. Here the in-memory computing platform using HieIM offers $\sim 8\times$ energy saving and $\sim 5\times$ speed up compared to that using DRAM based in-memory computing platform [5]. Again, the computation in DRAM is destructive, i.e. data stored in three DRAM cells associated with the computation are overwritten with the result of the logic operation. Whereas, the stored data are retained even after computing in our STT-MRAM based design.

B. In-memory Data Encryption Engine:

Advanced Encryption Standard (AES) [32] has been used to employ in-memory data encryption engine using HieIM. AES works on the standard input length of 16 bytes (128 bits) data organized in a 4×4 matrix (state matrix) while using three different key lengths (128, 192, & 256 bits) [35]. For 128-bit key length, AES encrypts the input data after 10 rounds of consecutive transformations [36]- e.g. SubBytes, ShiftRows, MixColumns, and AddRoundKey (Fig. 6).

Performance Evaluation: For evaluation of AES performance in general purpose processor (GPP), we have used similar method in [36] at 2GHz. AES C code is extracted from [37] and compiled, then cycle-accurate architecture simulator gem5 [38] is employed to take AES binary and system level processor power evaluating tool McPAT [39] is used to estimate power dissipation. For evaluation of AES in CMOS ASIC (1.133GHz), Synopsys Design Compiler tool is used. Here, the performance for all the platforms are listed in Table VI for 32nm technology. We have done fixed-voltage scaling of the results obtained from of our work to 32nm by using the appropriate scaling factor- which is $(1/S^2)$ for area and $(1/S)$ for energy [40], here S= L/32nm, where L=45nm. The device to architecture co-simulation results show that HieIM

TABLE VI
PERFORMANCE COMPARISON OF 128-BIT AES IMPLEMENTTAIONS

Platforms	Energy (nJ)	Cycles	Area (μm^2)
GPP [37]	460	2309	2.5e+6
ASIC [41]	6.6	336	4400
CMOL[42]	10.3	470	320
Baseline DW [36]	2.4	1022	78
Pipelined DW [36]	2.3	2652	83
Multi-issue DW [36]	2.7	1320	155
HieIM	3.2	1620	21.8

can achieve 51.5% and 68.9% lower energy consumption compared to CMOS-ASIC and CMOL based implementations, respectively. Furthermore, HielM occupies $\sim 3.5\times$ less area compared to baseline DW-AES [36], which requires lower number of cycles due to intrinsic shift operation and multi-bit data storage of DWM racetrack devices. However, racetrack device suffers from data corruption due to non-uniform DW velocities and huge Joule heating for large device dimension.

VII. CONCLUSION

In this paper we have proposed a dual-mode in-memory computing architecture using STT-MRAM array architecture which can perform any Boolean logic function (AND/NAND, OR/NOR, XOR/XNOR) between any two memory cells within the same sub-array. Extensive device, circuit and system level simulation have been carried out to evaluate the performance of the proposed in-memory computing platform in both memory mode and computing mode. In-memory bulk bitwise Boolean vector logic (AND/OR) operation for different vector datasets shows $\sim 8\times$ energy saving and $\sim 5\times$ speed up compared to that using DRAM based in-memory computing platform. We further have employed in-memory data encryption engine using AES algorithm, which shows 51.5% and 68.9% lower energy consumption compared to CMOS-ASIC and CMOL based implementations, respectively. To summarize, by adding several significant features as- non-volatility, in-memory logic operation with high data mapping flexibility, low dynamic power consumption, high packing density; our proposed design can thrive a new paradigm for future power efficient in-memory computing platform.

ACKNOWLEDGEMENT

This material is based upon work supported in part by the National Science Foundation under Grant No. 1740126.

REFERENCES

[1] W. A. Wulf et al. Hitting the memory wall: implications of the obvious. *ACM SIGARCH*, 23(1):20–24, 1995.
[2] P. Chi et al. Prime: a novel processing-in-memory architecture for neural network computation in reram-based main memory. In *ISCA*, volume 43, 2016.
[3] Y. Wang et al. An energy-efficient nonvolatile in-memory computing architecture for extreme learning machine by domain-wall nanowire devices. *IEEE TNANO*, 14(6):998–1012, 2015.
[4] J.-P. Wang et al. General structure for computational random access memory (cram), Dec. 2015. US Patent 9,224,447.
[5] V. Seshadri et al. Fast bulk bitwise and and or in dram. *IEEE Computer Architecture Letters*, 14(2):127–131, 2015.
[6] S. Li et al. Pinatubo: a processing-in-memory architecture for bulk bitwise operations in emerging non-volatile memories. In *DAC*, pages 1–6. IEEE, 2016.
[7] Y. Seo et al. High performance and energy-efficient on-chip cache using dual port (1r/1w) spin-orbit torque mram. *IEEE JETCAS*, 6(3):293–304, 2016.
[8] S. Jain et al. Computing in memory with spin-transfer torque magnetic ram. *arXiv preprint arXiv:1703.02118*, 2017.
[9] W. Kang et al. In-memory processing paradigm for bitwise logic operations in stt-mram. *IEEE Trans. Magn.*, 2017.
[10] S. Kvatinsky et al. Magic—memristor-aided logic. *IEEE TCAS II*, 61(11):895–899, 2014.
[11] B. a. Hudec. 3d resistive ram cell design for high-density storage class memory—a review. *Science China Information Sciences*, 59(6):061403, 2016.
[12] S. Raoux et al. Phase-change random access memory: a scalable technology. *IBM J Res Dev*, 52(4.5):465–479, 2008.
[13] B. C. Lee et al. Architecting phase change memory as a scalable dram alternative. In *ACM SIGARCH*, volume 37 of number 3, pages 2–13. ACM, 2009.
[14] E. Kultursay et al. Evaluating stt-ram as an energy-efficient main memory alternative. In *ISPASS*, pages 256–267. IEEE, 2013.
[15] E Chen et al. Advances and future prospects of spin-transfer torque random access memory. *IEEE Trans. Magn.*, 46(6):1873–1878, 2010.
[16] Everspin stt. 2016. URL: https://www.everspin.com/news/everspin-readies-industry%E2%80%99s-first-256mb-perpendicular-spin-torque-mram.
[17] S.-W. Chung and n. c. o. Supercomputing. 4gbit density stt-mram using perpendicular mtj realized with compact cell structure. In *IEDM*, pages 27–1. IEEE, 2016.
[18] S. Aga et al. Compute caches. In *hpca*, pages 481–492. IEEE, 2017.
[19] G Autes et al. Strong enhancement of the tunneling magnetoresistance by electron filtering in an fe/mgo/fe/gaas (001) junction. *Physical review letters*, 104(21):217202, 2010.
[20] X. Fong et al. Spin-transfer torque devices for logic and memory: prospects and perspectives. *IEEE TCAD*, 35(1):1–22, 2016.
[21] X. Fong et al. Knack: a hybrid spin-charge mixed-mode simulator for evaluating different genres of spin-transfer torque mram bit-cells. In *SISPAD*, pages 51–54. IEEE, 2011.
[22] F. Parveen et al. Hybrid polymorphic logic gate with 5-terminal magnetic domain wall motion device. In *ISVLSI*, pages 152–157. IEEE, 2017.
[23] Z. He and D. Fan. A low power current-mode flash adc with spin hall effect based multi-threshold comparator. In *ISLPED*, pages 314–319. ACM, 2016.
[24] Ncsu eda freepdk45. 2011. URL: http://www.eda.ncsu.edu/wiki/FreePDK45:Contents.
[25] X. Dong et al. Nvsim: a circuit-level performance, energy, and area model for emerging non-volatile memory. In *Emerging Memory Technologies*, pages 15–50. Springer, 2014.
[26] J.-W. Ryu et al. Self-adjusting sensing circuit without speed penalty for reliable stt-mram. *Electronics Letters*, 53(4):224–226, 2017.
[27] K. Huang et al. Magnetic domain-wall racetrack memory-based nonvolatile logic for low-power computing and fast run-time-reconfiguration. *IEEE TVLSI*, 24(9):2861–2872, 2016.
[28] K. Huang et al. Stt-mram based low power synchronous non-volatile logic with timing demultiplexing. In *NANOARCH*, pages 31–36. ACM, 2014.
[29] H.-P. Trinh et al. Magnetic adder based on racetrack memory. *IEEE TCAS I*, 60(6):1469–1477, 2013.
[30] A. Roohi et al. A tunable majority gate-based full adder using current-induced domain wall nanomagnets. *IEEE Trans. Magn.*, 52(8):1–7, 2016.
[31] Y. Zhang et al. Giant spin hall effect (gshe) logic design for low power application. In *DATE*, pages 1000–1005. IEEE, 2015.
[32] J. Daemen et al. *The design of Rijndael: AES — the Advanced Encryption Standard*. Springer-Verlag, 2002, page 238. ISBN: 3-540-42580-2.
[33] Y Yanagawa et al. In-substrate-bitline sense amplifier with array-noise-gating scheme for low-noise 4f 2 dram array operable at 10-ff cell capacitance. In *VLSIC*, pages 230–231. IEEE, 2011.
[34] D. T. Wang. *Modern dram memory systems: performance analysis and scheduling algorithm*. PhD thesis, Unversity of Maryland, 2005.
[35] N.-F. Standard. Announcing the advanced encryption standard (aes). *FIPSP*, 197, 2001.
[36] Y. Wang et al. Dw-aes: a domain-wall nanowire-based aes for high throughput and energy-efficient data encryption in non-volatile memory. *IEEE TIFS*, 11(11):2426–2440, 2016.
[37] K Malbrain. Byte-oriented-aes: a public domain byte-oriented implementation of aes in c, 2009.
[38] N. Binkert et al. The gem5 simulator. *SIGARCH*, 39:1–7, 2011.
[39] S. Li et al. Mcpat: an integrated power, area, and timing modeling framework for multicore and manycore architectures. In *MICRO*, pages 469–480. ACM, 2009.
[40] A. Stillmaker et al. Toward more accurate scaling estimates of cmos circuits from 180 nm to 22 nm. *VLSI Computation Lab, ECE Department, University of California, Davis, Tech. Rep. ECE-VCL-2011-4*, 4, 2011.
[41] S. Mathew et al. 340 mv–1.1 v, 289 gbps/w, 2090-gate nanoaes hardware accelerator with area-optimized encrypt/decrypt gf (2 4) 2 polynomials in 22 nm tri-gate cmos. *IEEE JSSC*, 50(4):1048–1058, 2015.
[42] Z Abid et al. Efficient cmol gate designs for cryptography applications. *IEEE TNANO*, 8:315–321, 2009.