A Multilevel Cell STT-MRAM-Based Computing In-Memory Accelerator for Binary Convolutional Neural Network

Article *in* IEEE Transactions on Magnetics · July 2018

DOI: 10.1109/TMAG.2018.2848625

Yinglin Zhao
Beihang University (BUAA)
10 PUBLICATIONS **34** CITATIONS

SEE PROFILE

Wang Kang
Beihang University (BUAA)
162 PUBLICATIONS **2,421** CITATIONS

SEE PROFILE

Shouyi Yin
Tsinghua University
276 PUBLICATIONS **1,769** CITATIONS

SEE PROFILE

Weisheng ZHAO
Beihang University (BUAA)
685 PUBLICATIONS **9,145** CITATIONS

SEE PROFILE

A Multilevel Cell STT-MRAM-Based Computing In-Memory Accelerator for Binary Convolutional Neural Network

Yu Pan[1], Peng Ouyang[1,2], Yinglin Zhao[1], Wang Kang[1,2], Shouyi Yin[3], Youguang Zhang[1], Weisheng Zhao[1,2], and Shaojun Wei[3]

[1]BDBC, School of Electronics and Information Engineering, Beihang University, Beijing 100083, China
[2]Beihang University Hefei Innovation Research Institute, Heifei 230013, China
[3]Institute of Microelectronics, Tsinghua University, Beijing 100084, China

Due to additive operation's dominated computation and simplified network in binary convolutional neural network (BCNN), it is promising for Internet of Things scenarios which demand ultralow power consumption. By means of fully exploiting the in-memory computing advantages and low current consumption design using multilevel cell (MLC) spin-toque transfer magnetic random access memory (STT-MRAM), this paper proposes an MLC-STT-computing in-memory-based computing in-memory architecture to achieve convolutional operation for BCNN to further reduce the power consumption. Simulation results show that compared with the resistive random access memory (RRAM)- and spin orbit torque-STT-MRAM-based counterparts, the architecture proposed in this paper reduces power consumption by ∼35× and 59% in Modified National Institute of Standards and Technology data set, respectively.

Index Terms—**Binary convolutional neural network (BCNN), full-add operation, in-memory computing, multilevel cell (MLC) spin-toque transfer magnetic random access memory (STT MRAM).**

I. Introduction

CONVOLUTIONAL neural networks (CNNs), which can provide humane intelligence such as object recognition and speech recognition, are widely used in Internet of Things (IoT), which demands small storage area and low computing resources. It is a challenge for both software and hardware technologies. Binary CNN (BCNN) is an excellent way to solve this issue with small precision loss [1], [2]. By binarizing inputs and weights, BCNN greatly reduces storage overhead and computational burden and transforms complex convolution operations into additions, which shows great potential in IoT application. However, for the implementation of IoT applications, especially wearable applications, current ways of using static random-access memory (SRAM) that separate computing and storing still have problems of low power consumption and low area overhead, since a lot of data handling causes energy costs.

Nonvolatile memory (NVM) technologies such as resistive random access memory (RRAM) [3], phase-change memory [4], and spin-toque transfer magnetic random access memory (STT-MRAM) [5] are promising substitutes for SRAM to reduce standby power. Among these NVMs, STT-MRAM is most preferable due to its unique characteristics like high density and near-zero power leakage [6]. And multilevel cell STT-MRAM (MLC-STT-MRAM) is proposed in [7], making it possible to store two bits in one cell. Since data communicating in current design between memory and processors consume much energy and accessing time, further improvement of system performance is limited. In-memory computing is an effective way to solve this issue. Taking advantage of STT-MRAM, memory architectures in [8] and [9] make it possible to compute in memory efficiently and can work as NVM and reconfigurable in-memory logic simultaneously without add-on logic circuits to memory chip.

In this paper, we design an MLC-STT-computing in-memory (CIM)-based computing in-memory accelerator for BCNN to further reduce the power consumption. When realizing the add operation, different from the designs in [8] and [9], in which two addends are stored in two different bit-cells of the same column within an STT-CIM and enable both word lines (WLs) to be connected with the drain of nMOSs, our design stores two bits in one cell and achieves logic and add operation in one cell between the two bits that is benefited from our proposed modified sensing circuit. Therefore, this design needs less number of nMOS transistors and has lower current consumption when executing in-memory computing. It greatly reduces the number of power-hungry and long-distance data communications. Testing the convolutional layers of BCNN on Modified National Institute of Standards and Technology (MNIST) data set on this design, the simulation results show that our accelerator which removes lots of long-distance and power-hungry data communications between the computing unit and the memory unit greatly reduces the energy consumption and is more promising for the ultralow power occasions when compared with the state of the arts.

II. Proposed MLC-STT-CIM Structure

A. MLC-STT-MRAM Cell

Fig. 1(a) shows an MLC-STT-MRAM cell structure with a small MTJ and a large MTJ that store bit information [7]. An MTJ consists of two ferromagnetic layers and a tunneling layer between them. One of the ferromagnetic layers is called the reference layer with fixed magnetization direction, and the

Manuscript received March 9, 2018; revised May 31, 2018; accepted June 11, 2018. Date of publication July 4, 2018; date of current version October 17, 2018. (*Yu Pan, Peng Ouyang, and Yinglin Zhao contributed equally to this work*). Corresponding author: P. Ouyang (e-mail: oyp2017@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/TMAG.2018.2848625

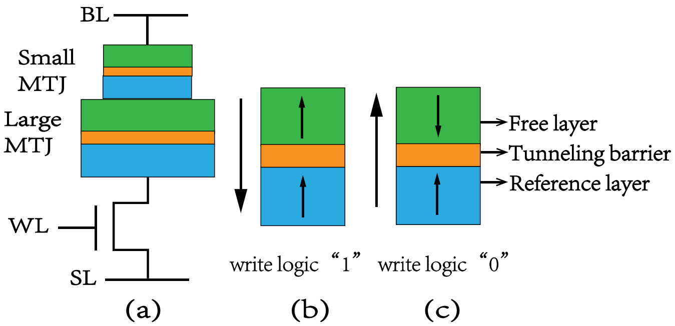

Fig. 1. (a) MLC-STT-MRAM cell structure. (b) MTJ with parallel construction. (c) MTJ with antiparallel construction.

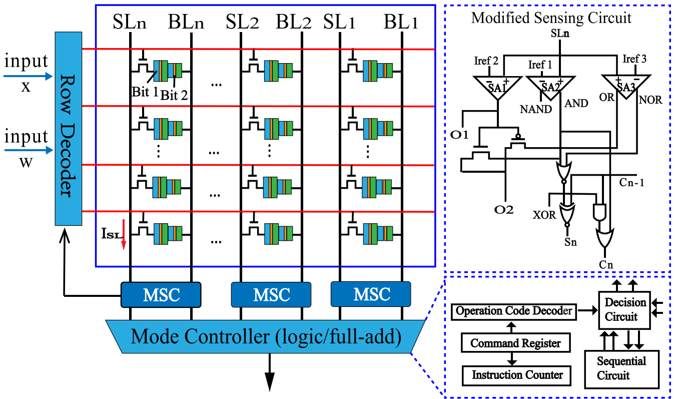

Fig. 2. Proposed MLC-STT-CIM architecture.

other is called the free layer whose magnetization direction can be changed by applying a current bigger than critical switching current. When applying positive voltage between BL and SL, the two ferromagnetic layers will have parallel magnetization direction, as shown in Fig. 1(b), and the MTJ shows low resistance (R_P) representing logic "1." In the opposite case shown in Fig. 1(c), the resistance of MTJ (R_{AP}) is higher and it represents logic "0." Hence, the two MTJs in one cell have four resistance states ($R_{\text{AP-AP}}$, $R_{\text{AP-}P}$, $R_{P\text{-AP}}$, $R_{P\text{-}P}$) that can represent a two-bit data, respectively.

B. MLC-STT-CIM Structure

As shown in Fig. 2, the MLC-STT-CIM structure is proposed to perform in-memory computing. Modified sensing circuit (MSC) is designed for logic and full-add operation, and mode controller decides which operation the architecture works in. As mentioned in Section II-A, different two-bit data in a MLC-STT-CIM cells are presented by different resistance values. According to Ohm's law, the current of SL I_{SL} has four possible values correspondingly. Assume that the first data are stored in the large MTJ and the other one is stored in the small MTJ, and $R_{\text{AP-}P} > R_{P\text{-AP}}$. Relationship between cell resistances, logic data, and I_{SL} is presented in Table I. Connect I_{SL} to our MSC shown in Fig. 2, we can realize both the memory and the computing operation. Relationships of size between reference currents I_{ref} and I_{SLs} are shown in Table I. Next, we will go into more details about how our MLC-STT-CIM works.

C. Working Mechanism

Based on the structure mentioned above and under the controller of mode controller, the MLC-STT-CIM works in three modes including memory mode, logic mode, and full-add mode.

TABLE I
RELATIONSHIP BETWEEN CELL RESISTANCES, LOGIC DATA, AND I_{SL}. RELATIONSHIP BETWEEN REFERENCE CURRENTS AND I_{SLs}

Resistance	Logic data	I_{SL}
$R_{AP\text{-}AP}$	00	I_{00}
$R_{AP\text{-}P}$	01	I_{01}
$R_{P\text{-}AP}$	10	I_{10}
$R_{P\text{-}P}$	11	I_{11}
$R_{AP\text{-}AP} > R_{AP\text{-}P} > R_{P\text{-}AP} > R_{P\text{-}P}$		
$I_{00} < I_{01} < I_{10} < I_{10}$		
$I_{00} <$ Iref3 $< I_{01} <$ Iref2 $< I_{10} <$ Iref1 $< I_{11}$		
$I_{00} <$ Iref3 $< I_{01} <$ Iref2 $< I_{10} <$ Iref1 $< I_{11}$		

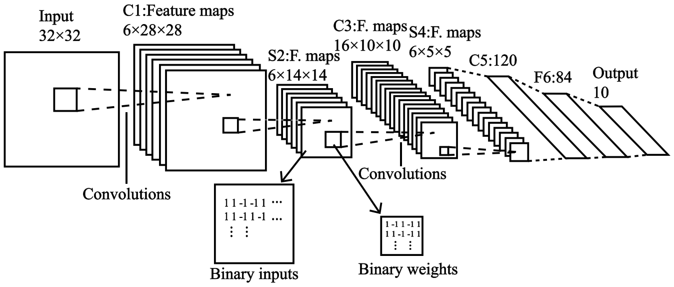

Fig. 3. XNOR-Net topology used in this paper.

1) Memory Mode: To write a bit into STT-MRAM cell, a suitable current should be used to pass through MTJ to change resistance of MTJ. Applying a small current to the cell, we can read the bit by comparing the sensing current to a reference signal. For MLC-STT-CIM cell, when WLs are enabled, we can write the bits to the cell with a two-step writing scheme: a large current is used to change the magnetic orientation of the large MTJ to write the first bit named bit1 as denoted in Fig. 2. And based on the second bit named bit2 in small MTJ, a small current is used to switch the state of small MTJ if necessary. To read these bits, we connect current of source line to the positive input of the sense amplifiers: SA1, SA2, and SA3, respectively. Connecting I_{ref2} to the negative input of SA1 to read the first bit from output port O1, the second bit signal will be sensed from output port O2. Different from the traditional MLC-STT-MRAM-based design, whose read operation is also a two-step procedure [7], this design can read two bits at a time, which saves accessing time.

2) Logic Mode: Except sensing the second bit in memory mode, SA2 can realize logic AND and NAND. As mentioned above, only I_{11} is larger than I_{ref1}. In other words, only both MTJs are in the P configuration (both store logic "1"), which leads to an output of logic "1" ("0") at the positive (negative) output of SA2, while all other cases lead to logic "0" ("1"). Thus, the positive and negative outputs of SA2 evaluate the logic AND and NAND of the values stored in the enabled bit-cells. Obviously, A OR (NOR) operation can be realized at the positive (negative) terminal of SA3, and an XOR operation can be realized when feeding the AND output of SA2 and

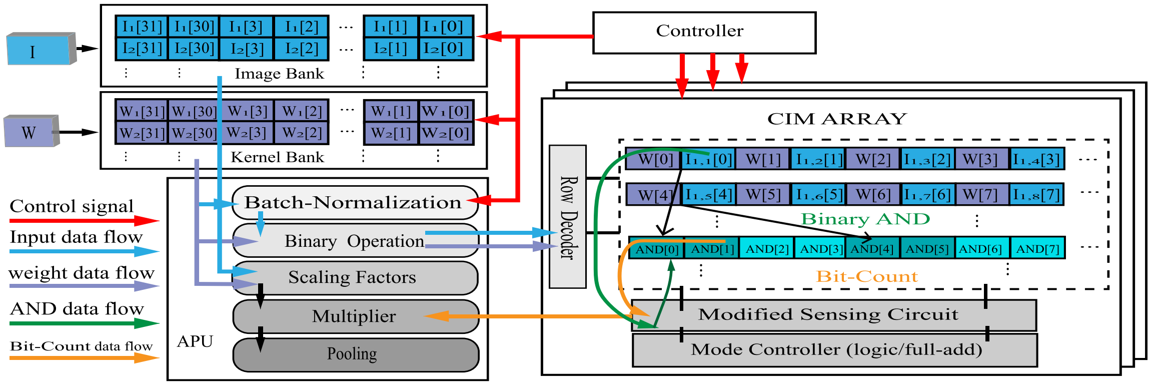
I
W
I₁[31] I₁[30] I₁[3] I₁[2] ··· I₁[1] I₁[0]
I₂[31] I₂[30] I₂[3] I₂[2] ··· I₂[1] I₂[0]
Image Bank
W₁[31] W₁[30] W₁[3] W₁[2] ··· W₁[1] W₁[0]
W₂[31] W₂[30] W₂[3] W₂[2] ··· W₂[1] W₂[0]
Kernel Bank
Controller
CIM ARRAY
Row Decoder
W[0] I₁.₁[0] W[1] I₁.₂[1] W[2] I₁.₃[2] W[3] I₁.₄[3]
W[4] I₁.₅[4] W[5] I₁.₆[5] W[6] I₁.₇[6] W[7] I₁.₈[7]
Binary AND
AND[0] AND[1] AND[2] AND[3] AND[4] AND[5] AND[6] AND[7]
Bit-Count
Modified Sensing Circuit
Mode Controller (logic/full-add)
Control signal
Input data flow
weight data flow
AND data flow
Bit-Count data flow
APU
Batch-Normalization
Binary Operation
Scaling Factors
Multiplier
Pooling

Fig. 4. Proposed accelerator for BCNN and the main compute flow for convolutional layers of BCNN.

the NOR output of SA3 to a complementary metal oxide semiconductor (CMOS) NOR gate.

3) Full-Add Mode: Suppose words A_n and B_n are stored in large MTJ and small MTJ in a cell within an MLC-STT-CIM array, and we wish to compute the full-add logic function. According to the following algorithm shown in (1) and (2), the sum S_n and the carry out C_n can be computed using A_n XOR B_n and A_n AND B_n, in addition to C_{n-1}. We can see that ADD operation in terms of the outputs of bitwise operations, AND, and XOR. Three additional logic gates are required to enable this computation. In this mode, for example, take the computing of 4-bit width-based adder, the small energy of 3.4 pJ is consumed, since the add operation is performed within one cell

$$S_n = A_n \wedge B_n \wedge C_n \tag{1}$$

$$C_n = ((A_n \wedge B_n)\,\&\,C_{n-1})|(A_n \& B_n). \tag{2}$$

III. Proposed Accelerator for BCNN

A. Binary Convolutional Neural Network

In Fig. 3, the XNF-Net [1] is adopted to perform our fully connected layers. Convolutional layer is used to extract features and fully connected layer is in principle the same as the traditional multi-layer perceptron neural network, which classifies the extracted features. In XNOR-Net, fully connected layers are converted into convolutional layers, and both the weights and the inputs are approximated with binary values (“+1” or “−1”). In circuit level, we denote “+1” and “−1” by logic “1” and “0,” respectively. The convolutional operation of binary inputs $I(B)$ and weights $W(B)$ is expressed as follows, which can be fully implemented in memory with our MLC-STT-CIM described above:

$$I * W = \mathrm{Bit} - \mathrm{Count}(I(B)\&W(B)). \tag{3}$$

B. Proposed BCNN Accelerator

Fig. 4 shows the proposed accelerator for BCNN and the main computing flow for convolutional layers of BCNN. Input I and W tensors are stored in Image Banks and Kernel Banks to prepare for the next calculations. Batch normalization, binary operation, scaling factors, multiplier, and pooling are organized in auxiliary process unit. Our MLC-STT-CIM is responsible for the convolutional operations. All these are controlled by global controller.

The detailed calculation process is as follows. First, inputs are transferred to batch normalization to perform batch normalization that will guarantee small information loss. Second, normalized inputs and weights are binarized in binary operation with sign function. Then binarized inputs $I(B)$ and weights $W(B)$ are transmitted to the proposed CIM ARRAY to perform in memory operation. One of the advantages of CNN is weight sharing, so in CIM ARRAY, we store weights in large MTJs and corresponding inputs in small MTJs to support bitwise AND within one cell. Green line represents data flow of binary AND operation and the AND results from MSCs are written into CIM ARRAY directly. Orange line represents bit-count operation data flow and using full-add function we realize bit-count in MSC. At the same time, tensors of I and W are sent to scaling factors to calculate α and K. Finally, convolutional results and scaling factors α and K are conveyed to multiplier to finish the calculations in convolutional layer. Next, max pooling is executed in pooling. In this design, convolutional computation no longer needs the operations of data read and write, which saves a large quantity of communication energy.

IV. Experimental Results

A. Methodology

Testing on MNIST data set, we adopt XNOR-Net [1] to evaluate this paper. The operation parameters of XNOR-Net are shown in Table II. The main MTJ parameters: diameters of large MTJ and small MTJ are 60 and 40 nm, tunneling barrier thickness is 0.85 nm, and TMR = 1.7. We got the current and voltage parameters with 45 nm CMOS technology in Cadence Virtuoso when our MLC-STT-CIM perform full-add operation. System-level performance, energy, and cycle time of the proposed accelerator is estimated by modified CACTI [10] and NVSim [11]. Meanwhile, we adopt state of the arts [3], [9] which use RRAM and spin orbit torque (SOT)-MRAM to perform the comparisons. The corresponding results are shown in Fig. 5 and Tables II and III.

B. Experimental Results

First, the area breakdown of MLC-STT-CIM and its SOT-MRAM counterpart [9] is shown in Fig. 5. Due to the high density of MLC-STT-MRAM, the cell array contributes only 66% of area overhead. Design in [9] composes not

TABLE II

XNOR-NET PARAMETERS OF CONVOLUTIONAL LAYERS AND CORRESPONDING ENERGY COMPARISON OF THE STATES OF THE ARTS

Layer	Kernel number	The number of convolution operations	Number of ANDs	Computation of Bit Count/ Bit-Count number	Energy consumption of RRAM[3]		Energy consumption of SOT-MRAM[9]		Energy consumption of our design	
					Layer (uJ)	Per operation (nJ)	Layer (uJ)	Per operation (nJ)	Layer (uJ)	Per operation (nJ)
C1	6	4704	117600	25/4704	9.92	2.1	0.67	0.14	0.278	0.059
C3	16	1600	40000	25/1600	3.37	2.1	0.229	0.14	0.094	0.059
C5	120	120	3000	25/120	0.25	2.1	0.017	0.14	0.007	0.059
F6	84	84	84	0	0.01	0.11	0.001	0.008	0.0003	0.003
Total	-	-	-	-	13.55	-	0.92	-	0.38	-

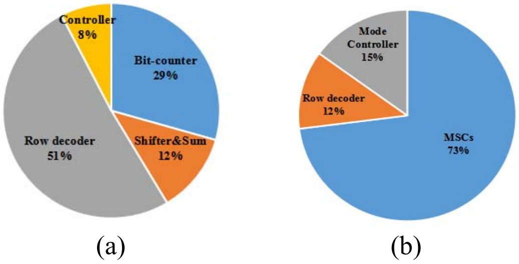

Fig. 5. (a) Breakdown of area overhead SOT-MRAM counterpart [9]. (b) Breakdown of area overhead of our design.

TABLE III

EVALUATION OF FULL ADDER WITH DIFFERENT BIT WIDTHS

Width of adder	Energy(pJ)	Area(F^2)	Execution time (ns)
4-bit	3.4	624	9.4
8-bit	7.3	1248	14.2
16-bit	16.4	2496	23.8
32-bit	40	4992	43

only sensory circuits in subarray, but also bit-counter and Shifer&Sum for pooling operation, and the area breakdown of them is only 40%. In our design, MSCs offer 73% of area overhead of add-on circuit as they are needed in every column, and there are three comparators, two MOS transistors, and four logic gates to finish in-memory logic operation or full-add operation in each MSC. And the area breakdown of row decoder of [9] is larger than that of our design because its subarray size is 1024 × 256, which is larger than ours.

Besides, one of the superiorities of our design is that it can achieve full-add operation in memory. Table III shows the energy, area, and execution time when using different widths of adders. From the simulation results, we got that as the width increases, the execution time also increases because of the increase of operands, yet it decreases relatively on each bit on average, as we write all the addends in parallel. As we can see, this design can realize 4-bit full-add with only 3.4 pJ energy and 9.4 ns latency, which is prominent to meet the demand of low power and small delay time in IoT.

Furthermore, we applied it in MNIST data set and compared it with the state of the arts in terms of energy. Simulation results of convolution layers are shown in Table II. In BCNN, the energy consumption of our design is only 0.38 μJ, which is ∼48× and ∼35× less than CNN-RRAM and BCNN-RRAM, respectively [3], and lower 59% than SOT-MRAM [9]. This design achieves prominent improvement when compared with RRAM [3]; the main reasons are that the limits of RRAM themselves limit the subarray size, which causes energy overhead in matrix splitting and the peripheral circuits consume most of the energy (85% of energy overhead). SOT-MRAM [9] has a very large subarray size of 1024 × 256, which greatly improves parallelism and it does not have add-on peripheral circuits, which is the same as our design. However, energy overhead of our design is also lower than SOT-MRAM [9], which proves that our design exhibits the advantages of MLC-STT-MRAM fully. Benefited from the MLC-STT-MRAM design, we achieve AND operation in one cell rather than enable two WLs to operate in two multiple cells, which reduces operation current. The cycle time of our design is 27.24 ns and it will reduce significantly if we expand the subarray size to increase parallelism. What is more, according to MLC-STT-CIM's characteristic of compute in memory to reduce communicate energy, the larger the data set, the higher the energy efficiency will be achieved.

V. CONCLUSION

In this paper, we propose an MLC-STT-CIM architecture to achieve in-memory logic and full-add operation. When executing 4-bit full-add, it can achieve 3.4 pJ energy consumption with an execution time of 9.4 ns. Applying this design to accelerate BCNN, the simulation results show that it reduces power consumption by ∼35× and 59% in MNIST data set, respectively, when compared with the state of the arts.

ACKNOWLEDGMENT

This work was supported in part by the National Science Foundation for Young Scientists of China under Grant KZ73030401, in part by Beihang University Hefei Innovation Research Institute through Beihang Hefei Science City Project under Grant 01-0004-01, and in part by Special Foundation of Beijing Municipal Science and Technology Commission under Grant Z161100000216149.

REFERENCES

[1] M. Rastegari, V. Ordonez, J. Redmon, and A. Farhadi. (2016). "XNOR-Net: ImageNet classification using binary convolutional neural networks." [Online]. Available: https://arxiv.org/abs/1603.05279

[2] M. Courbariaux, I. Hubara, D. Soudry, R. El-Yaniv, and Y. Bengio. (2016). "Binarized neural networks: Training deep neural networks with weights and activations constrained to +1 or −1." [Online]. Available: https://arxiv.org/abs/1602.02830

[3] T. Tang, L. Xia, B. Li, Y. Wang, and H. Yang, "Binary convolutional neural network on RRAM," in *Proc. IEEE 22nd ASP-DAC*, Jan. 2017, pp. 782–787.

[4] B. C. Lee, E. Ipek, O. Mutlu, and D. Burger, "Architecting phase change memory as a scalable DRAM alternative," *SIGARCH Comput. Archit. News*, vol. 37, no. 3, pp. 2–13, Jun. 2009.

[5] Y. Kim, S. K. Gupta, S. P. Park, G. Panagopoulos, and K. Roy, "Write-optimized reliable design of STT MRAM," in *Proc. ACM/IEEE ISLPED*, Aug. 2012, pp. 3–8.

[6] R. Dorrance, F. Ren, Y. Toriyama, A. A. Hafez, C.-K. K. Yang, and D. Markovic, "Scalability and design-space analysis of a 1T-1MTJ memory cell for STT-RAMs," *IEEE Trans. Electron Devices*, vol. 59, no. 4, pp. 878–887, Apr. 2012.

[7] Y. Zhang, L. Zhang, W. Wen, G. Sun, and Y. Chen, "Multi-level cell STT-RAM: Is it realistic or just a dream?" in *Proc. ICCAD*, Nov. 2012, pp. 526–532.

[8] S. Jain, A. Ranjan, K. Roy, and A. Raghunathan. (2017). "Computing in memory with spin-transfer torque magnetic RAM." [Online]. Available: https://arxiv.org/abs/1703.02118

[9] S. Angizi, Z. He, F. Parveen, and D. Fan, "IMCE: Energy-efficient bit-wise in-memory convolution engine for deep neural network," in *Proc. ASP-DAC*, 2018, pp. 111–116.

[10] N. Muralimanohar, R. Balasubramonian, and N. Jouppi, "Optimizing NUCA organizations and wiring alternatives for large caches with CACTI 6.0," in *Proc. IEEE/ACM Int. Symp. Microarchitecture*, Dec. 2007, pp. 3–14.

[11] X. Dong, C. Xu, Y. Xie, and N. P. Jouppi, "NVSim: A circuit-level performance, energy, and area model for emerging nonvolatile memory," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 31, no. 7, pp. 994–1007, Jul. 2012.