An On-Chip Learning Accelerator for Spiking Neural Networks using STT-RAM Crossbar Arrays

Shruti R. Kulkarni*, Shihui Yin†, Jae-sun Seo† and Bipin Rajendran‡
*Department of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark, NJ 07102, USA.
†School of Electrical, Computer and Energy Engineering, Arizona State University, Tempe, AZ 85287, USA.
‡Department of Engineering, King's College London, Strand, London, WC2R 2LS, UK.
Email: bipin.rajendran@kcl.ac.uk

Abstract—**In this work, we present a scheme for implementing learning on a digital non-volatile memory (NVM) based hardware accelerator for Spiking Neural Networks (SNNs). Our design estimates across three prominent non-volatile memories - Phase Change Memory (PCM), Resistive RAM (RRAM), and Spin Transfer Torque RAM (STT-RAM) show that the STT-RAM arrays enable at least 2$\times$ higher throughput compared to the other two memory technologies. We discuss the design and the signal communication framework through the STT-RAM crossbar array for training and inference in SNNs. Each STT-RAM cell in the array stores a single bit value. Our neurosynaptic computational core consists of the memory crossbar array and its read/write peripheral circuitry and the digital logic for the spiking neurons, weight update computations, spike router, and decoder for incoming spike packets. Our STT-RAM based design shows ∼20$\times$ higher performance per unit Watt per unit area compared to conventional SRAM based design, making it a promising learning platform for realizing systems with significant area and power limitations.**

Index Terms—**Neuromorphic hardware, Spiking Neural Networks, crossbar arrays, STT-RAM**

I. Introduction

Deep Neural Networks (DNNs), inspired by the architecture of the brain, have become the state-of-the-art in carrying out various cognitive processing tasks, especially those that are needed at the edge [1]–[3]. However, training these networks requires huge computational resources and takes long time periods as the problem complexity grows [4]. While several research studies have demonstrated neural network model optimizations that can be carried out to realize them on energy constrained platforms [5], SNNs, inspired by the event-driven computation in the brain, have been shown to perform similar tasks with less computations than current DNNs [6].

In order to accelerate DNNs and SNNs, there have been several demonstrations of high throughput hardware accelerators [7]–[14]. Most of these accelerator architectures are based on minimizing the data transfer bottleneck between processor and storage units, which limits the performance of conventional von Neumann machines. However, these accelerators are based on conventional SRAM memory, which are limited in the memory density that can be achieved on-chip.

Emerging nanoscale Non-Volatile Memory (NVM) devices have been shown to be suitable for in-memory computing based on crossbar array architectures for designing neural network accelerators [15]–[17], which offer higher on-chip memory density, thereby reducing the number of off-array data transfers. The main challenge in accelerators having analog storage NVMs is that the memristive devices are sensitive to process variations and peripheral circuit noise [18].

In this paper, we describe the design of a generic crossbar array for realizing inference and training in SNNs. We use NVM devices as binary storage units, i.e. storing two resistive states of high ('0') or low ('1'). This scheme avoids the use of expensive data converters at the array periphery and the neuronal dynamics can be realized with conventional digital CMOS designs. It also avoids the issues pertaining to the conductance variability of these NVM devices as seen in analog memories. We evaluate our design for SNN inference across three different NVM devices, namely STT-RAM, PCM, and RRAM. The performance of the STT-RAM based design shows nearly 2$\times$ and 5$\times$ improvement over PCM and RRAM designs, respectively.

While 32-bit single precision floating point representation is commonly used by most digital computing platforms today, it has been shown that significant energy and speed benefits can be achieved in the hardware if the computation can be carried in lower bit-precision [19]. Several works for DNNs have demonstrated inference engines with 1-bit to 8-bit fixed-point formats [20]–[22]. Low-precision implementations are beneficial especially for embedded and edge devices, which run with a limited power budget and memory capacity. Training neural networks with lower precision is a challenging problem, since the range of values required by the gradients spans over 5 to 6 orders of magnitude, which cannot be supported in the low-precision fixed point representation. There have been several efforts in training DNNs with lower precision of 8-bit and 16-bit floating point representation by applying algorithmic modifications to network training such as transfer learning, stochastic rounding, etc. to achieve the baseline 32-bit floating-point accuracy [23]–[25]. We use 16-bit floating point representation in our work for realizing the on-chip learning accelerator. We compare our STT-RAM based hardware accelerator design with an equivalent SRAM based design and show that STT-RAM design achieves nearly 20$\times$ higher throughput per unit power and per unit area.

This paper is organized as follows. Section II gives a

This work was supported in part by the Semiconductor Research Corporation and Cisco.

brief background on spiking neural network models and a modified back-propagation learning rule used in this work. This model was first discussed in [26], which also presented a CMOS-based inference engine design. In Section III, we present our SNN inference accelerator design and compare the performance of three prominent NVM technologies - STT-RAM, RRAM, and PCM. We then present the schemes for realizing back-propagation based learning on the NVM crossbar array in Section IV. Section V presents the overall performance projections of our proposed architecture for carrying out forward pass, back-propagation, and weight updates. Finally, Section VI concludes the work and presents some future directions.

II. Binary Activation Spiking Neural Networks

We use the Binary Activation SNN (BASNN) model described in [26] to design our NVM based hardware accelerator. This network model employs gradient descent based learning rule to adjust the network weights and is one of the simplest SNN models to realize on hardware. It has been demonstrated to achieve close to the state-of-the-art SNN performance on the MNIST dataset, with the best test accuracy being 99.4% with a convolutional network. This algorithm uses a modified spiking neuron model with binary activation function, where the spike output (a_j^k) of a neuron j in layer k is given as:

$$a_j^k(t) = y_b\left(\sum_{i=1}^{N} a_i^{k-1}(t) w_{j,i}^k + b_j^k\right), \tag{1}$$

where y_b is the threshold function, with $y_b(x) = 1$ only if $x > \theta$ and θ is the spiking threshold for the membrane potential [26]. The term within the brackets in (1) represents the spiking neuron's membrane potential for the time-step t. The use of a straight-through estimator makes the neuronal function differentiable during training [27]. The neuron's activation derivative is:

$$a_j^{k\prime}(v_j^k(t)) = \begin{cases} \frac{1}{2\theta}, 0 \le v_j^k(t) \le 2\theta \\ 0, \text{ otherwise} \end{cases} \tag{2}$$

The process of training follows the conventional gradient descent based weight update. This requires evaluating the gradients of the loss function ($\mathcal{L}(\mathbf{w})$) with respect to the network parameters ($\mathbf{w}$ and $\mathbf{b}$) using the chain rule for derivatives. Finally, the weight update term which is proportional to the loss gradient ($\eta \partial \mathcal{L}/\partial \mathbf{w}^k$) is added to the current set of weights for each layer. The loss function $\mathcal{L}$ adopted for training is the squared hinge loss [26]. The process of weight update involves the following set of expressions to be evaluated. Weight update for each layer k, is given as,

$$\Delta \mathbf{w}^k = \eta \boldsymbol{\delta}^k \times (\mathbf{a}^{k-1})^T \tag{3}$$

Here, $\mathbf{a}^{k-1}$ is the vector of spikes output from the previous layer $k-1$. The error gradient is represented by the vector $\boldsymbol{\delta}^k$, which is iteratively calculated for each layer starting from the last layer $L = k+1$ as,

$$\boldsymbol{\delta}^k = \left((\mathbf{w}^{k+1})^T \times \boldsymbol{\delta}^{k+1}\right) \circ \mathbf{a}^{k\prime} \tag{4}$$

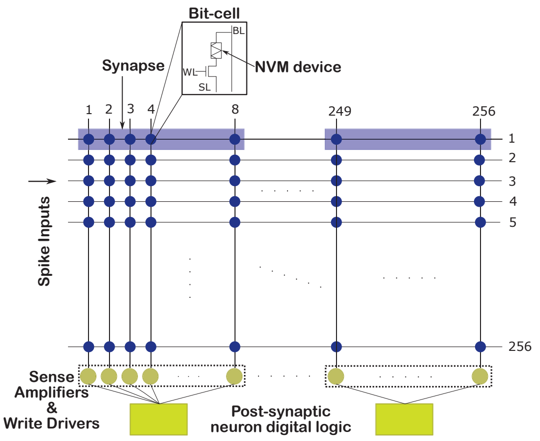
Bit-cell
BL
WL
SL
Synapse
NVM device
1
2
3
4
8
249
256
1
2
3
4
5
256
Spike Inputs
Sense Amplifiers & Write Drivers
Post-synaptic neuron digital logic

Fig. 1. Neuro-synaptic crossbar array based hardware with 256 inputs lines, 32 output neurons and 8-bit synapses. Each output layer neuron on the post-synaptic side of the array is connected to 8 bitlines and can access the associated devices for the selected row (wordline).

For the last layer L, the gradient term is evaluated as,

$$\boldsymbol{\delta}^L = \mathcal{L}' \circ (-\mathbf{S}^d) \tag{5}$$

Here, $\mathbf{S}^d$ is the desired set of spikes at the output layer of the network.

As demonstrated in [26], the BASNN fully-connected feed-forward network, with two hidden layers of 256 required only 7-bit signed fixed-point synaptic precision to achieve the floating-point baseline accuracy of 98.0% for the MNIST dataset. We now discuss the design of a crossbar array based architecture for accelerating SNN inference with 8 bits of fixed-point precision for the weights, as introduced in [28].

III. Memory Array Design

Fig. 1 shows a single neuro-synaptic core with a crossbar array of 256×256 NVM devices. The high resistance state (HRS) of the device represents binary '1' and low resistance state (LRS) represents '0'. Each of these arrays can be tiled together to realize larger and deeper networks for inference, similar to some of the earlier designs [12], [15]. The input spikes are applied to the respective rows and the post-synaptic digital neurons are interfaced along the columns of the array. The memory read and write peripheral circuits are also placed along the columns (bitlines) of the array. Each synaptic weight in our inference neurosynaptic core is represented in 8-bit fixed-precision. Thus, as seen in Fig. 1, eight columns of the array are connected to a single post-synaptic neuron. This particular core can support 32 post-synaptic neurons. The rows receiving a spike are read in a sequential manner in every memory read cycle.

We evaluate the performance of a single neurosynaptic core for inference with three different NVM devices - RRAM, PCM, and STT-RAM. For each of these devices, we built

TABLE I
READ AND WRITE CIRCUITS FOR NVM DEVICES DESIGNED IN 65 NM NODE

Design parameters	PCM	STT-RAM	RRAM
R_{SET} (Ω)	$10,000$	3000	$60,000$
R_{RESET} (Ω)	$1000,000$	6000	$500,000$
Read			
Area (μm^2)	21.18	41	21.18
HRS read power (μW)	2.95	1.42	2.34
LRS read power (μW)	3.07	1.51	2.67
Read latency (ns)	10	5	25
Write			
Area (μm^2)	134	324	134
HRS write power (μW)	777.2	1034	373.7
LRS write power (μW)	317.3	1027	382.98
Write latency (ns)	150	7	80

TABLE II
EVALUATION OF NEUROSYNAPTIC CORE FOR INFERENCE ACROSS THREE DIFFERENT NVM TECHNOLOGIES

Parameters	STT-RAM	PCM	RRAM
Read latency (ns)	5	10	25
Memory clock frequency (MHz)	100	50	20
Memory access power (mW)	0.53	0.92	0.79
Digital logic power (mW)	1.46	1.46	1.46
Total power (mW)	1.98	2.38	2.25
Total bit cell area (8 KB) (mm^2)	0.02	0.02	0.02
Memory peripheral area (mm^2)	0.095	0.040	0.040
Digital logic area (mm^2)	0.02	0.02	0.02
Total core area (mm^2)	0.13	0.08	0.08
GSOPS	3.2	1.6	0.64
GSOPS/W	1610	673	285
GSOPS/mm^2	24.62	20	8
GSOPS/W/mm^2	12385	8412	3562

compact models [29], [30] and their corresponding peripheral memory access circuits with specifications of the latest reported experimental devices [31]–[34]. Each NVM bit cell considered for this design has an area of 29F^2 in the 1T-1R configuration [35]–[38]. Table I lists the area and power numbers for the read and write circuits for each of the NVM devices. The read sense amplifiers are designed as either current mode or voltage mode. In the case of voltage mode sensing, a voltage drop across a pre-charged bitline indicates if the state is a '0' or not. In case of a current mode sensing, the current flowing through the device is compared with that flowing through a reference resistance cell. For the STT-RAM device, as the on-off ratio is low ($\sim$2), a current mode sensing is used. For the other two devices, which have a higher on-off ratio of $\sim$10, the voltage mode sensing is adopted.

To evaluate the designs with these three NVM arrays, we choose the performance metric of Synaptic Operations per Second (SOPS), or Giga SOPS (GSOPS) as introduced in [12]. One synaptic operation for inference involves reading an 8-bit synaptic weight and updating the respective post-synaptic neuronal membrane potential. Table II lists the performance metrics for the three different NVM-based designs. We also present the normalized metrics with respect to power (GSOPS/W), area (GSOPS/mm^2), and both area and power (GSOPS/W/mm^2). The metrics are evaluated for neurosynaptic core designed with the memory cells and peripheral logic at 65 nm node. As can be seen from Table II, the STT-RAM core has nearly 2$\times$ and 5$\times$ higher throughput per unit Watt compared to PCM and RRAM based neurosynaptic cores, respectively. The main reason for the performance improvement in the STT-RAM array is its low read latency, allowing the memory access to take place at higher frequency compared to PCM and RRAM. Hence, we use the STT-RAM memory arrays to design the learning accelerator for SNNs.

IV. LEARNING ACCELERATOR DESIGN

The NVM based neurosynaptic core described in Section III is extended to support on-chip spike based learning with STT-RAM arrays. We trained a fully-connected network with two hidden layers, each having 256 neurons for the MNIST dataset. We used 16-bit floating point (also called half-precision or FP-16) representation to train our network [39]. Our network obtained a test accuracy of 97.25% on the MNIST dataset after training for 500 epochs. We also used a dropout of 0.1 in the input layer and 0.2 in the hidden layers. To implement stochastic gradient descent, we employed a batch size of 1 for our design to avoid the need to store the intermediate variables of the size of the network parameters outside of the array. In [26], a fully-connected network with two hidden layers each with 1024 neurons in the 32-bit floating point precision reports a test accuracy of 98.7%, which used the Adam optimizer and a batch size of 100.

Fig. 2 shows our scheme for realizing the learning computations on the crossbar. It involves two phases, one for evaluating the gradients δ and the other for performing the weight updates. The forward pass evaluates the neuronal spike (a^k), membrane potentials (v^k), and the activation gradients

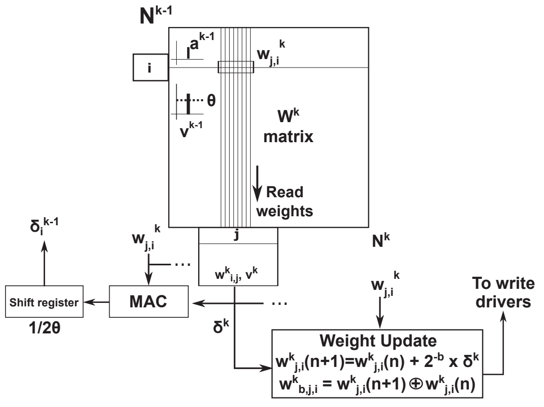

Fig. 2. Scheme for performing back-propagation of error gradients and weight update. During the backward pass, when a spike is received on the row of the array, the corresponding weights are read into the post-synaptic logic and new weights are computed. The flipped bits between the original and new weights are then programmed back into the array through write drivers. The error gradient back-propagation happens whenever the pre-synaptic neuron has non-zero activation gradient.

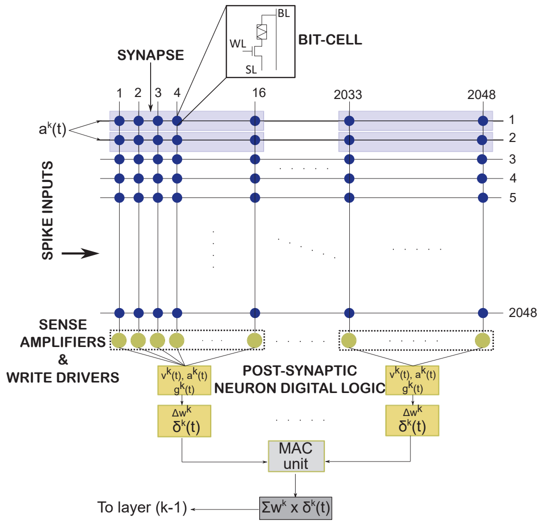

Fig. 3. A 2048 × 2048 crossbar array supporting access to 2048 inputs and 128 outputs at a time. Each synaptic weight is represented by 16 devices on a row. The peripheral digital logic consists of blocks to update the neuron membrane potential, the error derivative δ, and the weight update terms Δw.

(g^k). For each pre-synaptic neuron i, synaptic weights on the i^{th} row of the crossbar are read and stored in local registers of the post-synaptic neurons. If the pre-synaptic activation gradient is non-zero ($g^{k-1} \neq 0$ or if $v_i^{k-1} \in [0, 2\theta]$), we then compute the δ values as the output of a MAC (multiply and accumulate) unit as,

$$\delta_i^{k-1} = \frac{1}{2\theta} \sum_j w_{j,i}^k . \delta_j^k \tag{6}$$

The weight update term $\Delta \mathbf{w}$ is evaluated only if there is a pre-synaptic spike, i.e., if $a_i^{k-1} \neq 0$. The weights on the corresponding row, the $w_{j,i}$ values are read in the peripheral registers and the update terms Δw values are added to each of the synaptic weights as,

$$w_{j,i}(n+1) = w_{j,i}(n) + \Delta w_{j,i} \tag{7}$$

The bits flipped from the read weight values and the newly updated values are compared by performing a bit-wise XOR as $\mathbf{w}_{flip} = \mathbf{w}(n+1) \oplus \mathbf{w}(n)$, and then the write driver programs the specific bits to their required final states. The two steps of updating the weights on a row and evaluating δ values are repeated sequentially over all the wordlines which have received an input spike.

We used a larger array with 2048×2048 elements to realize the STT-RAM on-chip learning accelerator (Fig. 3). From interconnect parameters [40], we estimate the resistance of the bitline with 2048 bit cells as $R_{BL} = 9.5\,\text{k}\Omega$ and capacitance as $C_{BL} = 146\,\text{fF}$ at the 65 nm node. Thus, the RC wire delay of $\sim$ 1.4 ns is less than the minimum pulse width needed to read or write (2 to 10 ns) to a single device, making our large array design feasible. Each synaptic weight for a neuron has 16 STT-RAM bit cells. This array can support 128 post-synaptic neurons and 2048 inputs. The digital logic blocks at the array periphery consist of a 16-bit floating point adder, a MAC unit, and an XOR block to identify the flipped bits in the weights and biases. Additionally, the digital logic also includes the spike decoders and controllers needed for accessing the memory for read and write.

We designed the digital logic with FP-16 compute units in Verilog and synthesized them using the Synopsys Design Compiler tool to obtain the area and power estimates of each of these blocks at 65 nm node. Table III presents the synthesized area and power estimates for the different digital logic blocks (FP-16 neuron, MAC unit, spike decoder, router and controller). The area and power for the spike router is taken as 10% of that of the neuronal design based on previous published works [41]. We compare our design with an SRAM memory block of the same capacity as the STT-RAM array of 512 KB. The DESTINY tool is used to obtain the estimates for the SRAM memory block of 512 KB [42]. Similar to the SRAM cell used in the inference design, we use a bit cell of size $150F^2$ in DESTINY [26]. Table IV presents the read and write peripheral power for SRAM and STT-RAM. We estimate the area of the memory array as the sum of bit cell area (2048×2048 bit cells each having an area of $29F^2$) and the total area of the designed peripheral read/write circuits. Based on the maximum latency of SRAM access and bandwidth reported by DESTINY, we set the memory operating clock frequency at 250 MHz. The STT-RAM array that we designed can operate at 100 MHz based on the timing requirements of its peripheral circuits. The total area (post-synthesis) of the complete STT-RAM design is 1.83 mm^2, whereas the area of the SRAM design is 7.27 mm^2.

TABLE III
POST-SYNTHESIS AREA AND POWER NUMBERS FOR DIGITAL LOGIC BLOCKS IN THE NEUROSYNAPTIC CORE AT 65 NM NODE

Blocks	Area (mm^2)	Power (mW)
Neuron Logic (Forward pass)	0.358	20.49
Spike Router	0.036	2.05
Controller and decoder	0.11	6.15
MAC and weight update blocks	0.039	18.21
Total	0.543	46.9

TABLE IV
COMPARISON OF SRAM AND STT-RAM TECHNOLOGIES OF 512 KB CAPACITY USED IN NEUROSYNAPTIC CORE AT 65 NM NODE

Design Parameters	SRAM (DESTINY)	STT-RAM
Operating Frequency (MHz)	250	100
Read Power (mW)	529.5	27.29
Write Power (mW)	555.25	760.19
Memory bit cell area (mm^2)	6.29	1.22
Peripheral area (mm^2)	0.44	0.17

V. PERFORMANCE ANALYSIS

Our digital logic blocks synthesized with 65 nm library cells can operate at 500 MHz, without incurring any timing

TABLE V
AVERAGE SPIKE STATISTICS PER LAYER PER CORE IN THE SNN TO TRAIN WITH MNIST DATASET.

Network Statistics	**SRAM**	**STT-RAM**
Memory clock (MHz)	250	100
Incoming Spikes $\mathbf{a}^{k-1}$	20	20
Incoming gradients $\mathbf{g}^{k-1}$	95	95
Forward Pass		
No. of synaptic reads	2560	2560
Memory clock cycles for read	20	20
Power (mW)	580	78
Back-propagation		
No. of synaptic reads	5664	5664
Memory clock cycles for MAC	2833	1152
Power (mW)	489	26
Weight Update		
No. of writes	1180	1180
Memory clock cycles for write	20	160
Power (mW)	91	123
Overall GSOPS		
Synaptic Operations (SOPs)	9404	9404
Total # of Memory clock cycles	2853	1172
GSOPS	0.82	0.80

violation. The arrival of a spike on the input wordline enables the read of all the bit cells on a row in the memory array. For the STT-RAM array, the logic (running at 500 MHz) operates 5× faster than the memory (operating at 100 MHz), while for the SRAM memory (operating at 250 MHz), the logic clock is 2× faster than memory. The neuronal updates, as well as the evaluations of δ and Δw values take place in the digital logic. The back-propagation modules used for computing δs perform multiplication of δ_i^k and $w_{i,j}^k$ across all the output neurons i, of layer k, and hence need more clock cycles than the memory read/write per row j. The total cycles needed for writing to the weights through write drivers and computing error back-propagation δ^{k-1} is decided based on the maximum of the cycles needed to write to the memory array and to perform the MAC operation as the two stages can be parallelized. For the STT-RAM array, we make use of only two write drivers per 16-bit synapse, to limit the STT-RAM write driver power requirement and hence, each write in the STT-RAM array requires 8 memory clock cycles. To measure the performance, we compute the synaptic operations during the forward pass, back-propagation, and weight update, based on the spike and gradient statistics collected while training the network in software emulation with half-precision representation. Our neurosynaptic core can be used to realize one layer of the SNN with maximum of 128 neurons. For realizing layers with 256 neurons, two of such cores can be interfaced to the previous layer output.

We measure the performance of a single core, by measuring the time required to perform the synaptic reads for neuronal potential update (forward pass), synaptic reads for evaluating the MAC (for δ) and finally the synaptic reads and writes during the weight update stage (using Δw). Table V lists the average number of synaptic operations during different stages of learning in the neurosynaptic core.

Using the listed statistics we estimate the GSOPS (Giga Synaptic Operations per second) for the neurosynaptic core. Table VI presents the performance comparison of the SRAM and STT-RAM designs. While the SRAM can be operated at a higher frequency than the STT-RAM, the overall throughput is limited by the MAC logic which takes more cycles per memory access, hence, we do not see a significant difference in the GSOPS in the two designs. As we have accounted for the network activity during training, the number of synaptic writes is significantly smaller than the number of synaptic reads (in this example, we have $1,180$ writes compared to $8,224$ reads in both forward and backward passes per image). This translates to smaller overall energy requirement for the STT-RAM core, as STT-RAM memory read energy is significantly less than that of the SRAM memory (by $\sim$7×). Hence, considering the total power in the design for the forward and backward passes, the STT-RAM core has $\sim$5× higher GSOPS/W compared to that of the SRAM core. Normalizing the performance with respect to the core area, it can be seen that STT-RAM core has nearly 20× higher GSOPS/W/mm^2 than SRAM core.

TABLE VI
PERFORMANCE COMPARISON BETWEEN SRAM AND STT-RAM DESIGNS

Design	GSOPS	GSOPS/W	GSOPS/W/mm^2
SRAM design	0.82	0.71	0.09
STT-RAM	0.80	3.5	1.93

VI. CONCLUSION AND FUTURE WORK

We have designed a learning accelerator based on NVM crossbar arrays that implements learning for SNNs using 16-bit floating point representation. Overall, the STT-RAM core performs nearly 20× better than the equivalent SRAM core in terms of GSOPS/W/mm^2, due to its lower read energy and smaller bit cell area, when considering the network statistics for training with the MNIST dataset. Our design's throughput is limited by the number of cycles required by the MAC unit. Introducing more parallelization in the MAC logic can improve the overall performance of the design, which could be a future direction to this work. In future work, network training under novel bit-representation schemes such as bfloat as well as reduced precision could be studied and hardware-aware optimization strategies during training could be explored to improve network accuracy.

ACKNOWLEDGMENT

B.R. gratefully acknowledges support from IBM through a Faculty Award and support from NJIT where he was employed at the time of this work.

REFERENCES

[1] C. Szegedy *et al.*, "Inception-v4, inception-resnet and the impact of residual connections on learning." in *AAAI*, vol. 4, 2017, p. 12.

[2] D. Amodei, S. Ananthanarayanan, R. Anubhai, J. Bai, E. Battenberg, C. Case, J. Casper, B. Catanzaro, Q. Cheng, G. Chen *et al.*, "Deep speech 2: End-to-end speech recognition in english and mandarin," in *International Conference on Machine Learning (ICML)*, 2016, pp. 173–182.

[3] H. Greenspan, B. van Ginneken, and R. M. Summers, "Guest editorial deep learning in medical imaging: Overview and future promise of an exciting new technique," *IEEE Transactions on Medical Imaging*, vol. 35, no. 5, pp. 1153–1159, May 2016.

[4] F. Iandola, "Exploring the design space of deep convolutional neural networks at large scale," *arXiv preprint arXiv:1612.06519*, 2016.

[5] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.

[6] J. H. Lee, T. Delbruck, and M. Pfeiffer, "Training Deep Spiking Neural Networks Using Backpropagation," *Frontiers in Neuroscience*, vol. 10, p. 508, 2016.

[7] Y. Chen, T. Krishna, J. Emer, and V. Sze, "Eyeriss: An energy-efficient reconfigurable accelerator for deep convolutional neural networks," in *IEEE International Solid-State Circuits Conference (ISSCC)*, Jan 2016, pp. 262–263.

[8] S. Kim, T. Gokmen, H. Lee, and W. E. Haensch, "Analog cmos-based resistive processing unit for deep neural network training," in *IEEE International Midwest Symposium on Circuits and Systems (MWSCAS)*, Aug 2017, pp. 422–425.

[9] D. Han, J. Lee, J. Lee, S. Choi, and H. Yoo, "A 141.4 mw low-power online deep neural network training processor for real-time object tracking in mobile devices," in *IEEE International Symposium on Circuits and Systems (ISCAS)*, May 2018, pp. 1–5.

[10] J. Lee, C. Kim, S. Kang, D. Shin, S. Kim, and H. Yoo, "UNPU: A 50.6TOPS/W unified deep neural network accelerator with 1b-to-16b fully-variable weight bit-precision," in *IEEE International Solid - State Circuits Conference - (ISSCC)*, Feb 2018, pp. 218–220.

[11] G. Desoli, N. Chawla, T. Boesch, S. Singh, E. Guidetti, F. D. Ambroggi, T. Majo, P. Zambotti, M. Ayodhyawasi, H. Singh, and N. Aggarwal, "A 2.9 TOPS/W deep convolutional neural network SoC in FD-SOI 28nm for intelligent embedded systems," in *IEEE International Solid-State Circuits Conference (ISSCC)*, Feb 2017, pp. 238–239.

[12] P. A. Merolla, J. V. Arthur, R. Alvarez-Icaza, A. S. Cassidy, J. Sawada, F. Akopyan, B. L. Jackson, N. Imam, C. Guo, Y. Nakamura, B. Brezzo, I. Vo, S. K. Esser, R. Appuswamy, B. Taba, A. Amir, M. D. Flickner, W. P. Risk, R. Manohar, and D. S. Modha, "A million spiking-neuron integrated circuit with a scalable communication network and interface," *Science*, vol. 345, no. 6197, pp. 668–673, 2014.

[13] M. Davies *et al.*, "Loihi: A neuromorphic manycore processor with on-chip learning," *IEEE Micro*, vol. 38, no. 1, pp. 82–99, January 2018.

[14] J. Pei, L. Deng, S. Song, M. Zhao, Y. Zhang, S. Wu, G. Wang, Z. Zou, Z. Wu, W. He *et al.*, "Towards artificial general intelligence with hybrid tianjic chip architecture," *Nature*, vol. 572, no. 7767, p. 106, 2019.

[15] T. Gokmen and Y. Vlasov, "Acceleration of deep neural network training with resistive cross-point devices: Design considerations," *Frontiers in Neuroscience*, vol. 10, p. 333, 2016. [Online]. Available: https://www.frontiersin.org/article/10.3389/fnins.2016.00333

[16] I. Boybat, M. Le Gallo, S. Nandakumar, T. Moraitis, T. Parnell, T. Tuma, B. Rajendran, Y. Leblebici, A. Sebastian, and E. Eleftheriou, "Neuromorphic computing with multi-memristive synapses," *Nature Communications*, vol. 9, no. 1, p. 2514, 2018.

[17] P. Narayanan, A. Fumarola, L. L. Sanches, K. Hosokawa, S. C. Lewis, R. M. Shelby, and G. W. Burr, "Toward on-chip acceleration of the backpropagation algorithm using nonvolatile memory," *IBM Journal of Research and Development*, vol. 61, no. 4/5, pp. 11–1, 2017.

[18] M. Cheng, L. Xia, Z. Zhu, Y. Cai, Y. Xie, Y. Wang, and H. Yang, "Time:a training-in-memory architecture for rram-based deep neural networks," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pp. 1–1, 2018.

[19] M. Horowitz, "1.1 computing's energy problem (and what we can do about it)," in *IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC)*, Feb 2014, pp. 10–14.

[20] M. Rastegari, V. Ordonez, J. Redmon, and A. Farhadi, "XNOR-Net: imagenet classification using binary convolutional neural networks," in *European Conference on Computer Vision*. Springer, 2016, pp. 525–542.

[21] D. Lin, S. Talathi, and S. Annapureddy, "Fixed point quantization of deep convolutional networks," in *International Conference on Machine Learning (ICML)*, 2016, pp. 2849–2858.

[22] I. Hubara, M. Courbariaux, D. Soudry, R. El-Yaniv, and Y. Bengio, "Binarized neural networks," in *Advances in Neural Information Processing Systems*, 2016, pp. 4107–4115.

[23] O. Valery, P. Liu, and J.-J. Wu, "Low precision deep learning training on mobile heterogeneous platform," in *IEEE Euromicro International Conference on Parallel, Distributed and Network-based Processing (PDP)*, 2018, pp. 109–117.

[24] N. Wang, J. Choi, D. Brand, C.-Y. Chen, and K. Gopalakrishnan, "Training deep neural networks with 8-bit floating point numbers," in *Advances in Neural Information Processing Systems*, 2018, pp. 7675–7684.

[25] N. Mellempudi, S. Srinivasan, D. Das, and B. Kaul, "Mixed precision training with 8-bit floating point," *arXiv preprint arXiv:1905.12334*, 2019.

[26] S. Yin, S. K. Venkataramanaiah, G. K. Chen, R. Krishnamurthy, Y. Cao, C. Chakrabarti, and J. Seo, "Algorithm and hardware design of discrete-time spiking neural networks based on back propagation with binary activations," in *IEEE Biomedical Circuits and Systems Conference (BioCAS)*, Oct 2017, pp. 1–5.

[27] Y. Bengio, N. Léonard, and A. Courville, "Estimating or propagating gradients through stochastic neurons for conditional computation," *arXiv preprint arXiv:1308.3432*, 2013.

[28] S. R. Kulkarni, D. V. Kadetotad, S. Yin, J.-s. Seo, and B. Rajendran, "Neuromorphic Hardware Accelerator for SNN Inference based on STT-RAM Crossbar Arrays," iEEE International Conference on Electronics, Circuits and Systems (ICECS), 2019, (in press).

[29] S. R. Kulkarni, D. V. Kadetotad, J.-s. Seo, and B. Rajendran, "Well-posed Verilog-A compact model for phase change memory," in *IEEE International Conference on Simulation of Semiconductor Processes and Devices (SISPAD)*, 2018, pp. 369–373.

[30] S. R. Kulkarni, D. V. Kadetotad, S. Yin, J.-s. Seo, and B. Rajendran, "Compact modelling of non-volatile memory devices incorporating reliability characteristics," in *SRC TECHCON*, 2019.

[31] H. Noguchi *et al.*, "4Mb STT-MRAM-based cache with memory-access-aware power optimization and write-verify-write / read-modify-write scheme," in *IEEE International Solid-State Circuits Conference (ISSCC)*, Jan 2016, pp. 132–133.

[32] S. Kang, W. Y. Cho, B.-H. Cho, K.-J. Lee, C.-S. Lee, H.-R. Oh, B.-G. Choi, Q. Wang, H.-J. Kim, M.-H. Park *et al.*, "A 0.1-μm 1.8-V 256-Mb phase-change random access memory (PRAM) with 66-MHz synchronous burst-read operation," *IEEE Journal of Solid-State Circuits*, vol. 42, no. 1, pp. 210–218, 2007.

[33] Q. K. Trinh, S. Ruocco, and M. Alioto, "Voltage scaled STT-MRAMs towards minimum-energy write access," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 6, no. 3, pp. 305–318, 2016.

[34] Y. Kim, S. K. Gupta, S. P. Park, G. Panagopoulos, and K. Roy, "Write-optimized reliable design of STT MRAM," in *ACM/IEEE International Symposium on Low Power Electronics and Design*, 2012, pp. 3–8.

[35] M.-F. Chang, K.-F. Lin, C.-H. Chuang, L.-Y. Huang, T.-F. Chien, S.-S. Sheu, K.-L. Su, H.-Y. Lee, F. T. Chen, C.-H. Lien *et al.*, "Circuit design challenges and trends in read sensing schemes for resistive-type emerging nonvolatile memory," in *IEEE International Conference on Solid-State and Integrated Circuit Technology (ICSICT)*, 2012, pp. 1–4.

[36] C. J. Lin *et al.*, "45nm low power CMOS logic compatible embedded STT MRAM utilizing a reverse-connection 1T/1MTJ cell," in *IEEE International Electron Devices Meeting (IEDM)*, Dec 2009, pp. 1–4.

[37] Y. Jin, M. Shihab, and M. Jung, "Area, power, and latency considerations of stt-mram to substitute for main memory," in *International Symposium on Computer Architecture (ISCA)*, 2014.

[38] H. Cai, W. Kang, Y. Wang, L. Naviner, J. Yang, and W. Zhao, "High performance mram with spin-transfer-torque and voltage-controlled magnetic anisotropy effects," *Applied Sciences*, vol. 7, no. 9, p. 929, 2017.

[39] "Ieee standard for floating-point arithmetic," *IEEE Std. 754-2008*, pp. 1–70, Aug 2008.

[40] A. A. Vyas *et al.*, "International roadmap of devices and systems 2017 edition: More Moore." *The International Roadmap for Devices and Systems: 2017*, 2018.

[41] S. Moradi and R. Manohar, "The impact of on-chip communication on memory technologies for neuromorphic systems," *Journal of Physics D: Applied Physics*, vol. 52, no. 1, p. 014003, 2018.

[42] M. Poremba, S. Mittal, D. Li, J. S. Vetter, and Y. Xie, "DESTINY: A tool for modeling emerging 3D NVM and eDRAM caches," in *Design, Automation Test in Europe Conference Exhibition (DATE)*, March 2015, pp. 1543–1546.