Benchmark of RRAM based Architectures for Dot-Product Computation

Xiaochen Peng[1], and Shimeng Yu[2]
[1]School of Electrical, Computing and Energy Engineering, Arizona State University, Tempe, 85281, AZ
[2]School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, 30332, GA
Email: shimeng.yu@ece.gatech.edu

***Abstract*—Memory array architecture based on emerging non-volatile memory devices have been proposed for on-chip acceleration of dot-product computation in neural networks. As recent advances in machine learning have shown that precision reduction is a useful technique to reduce the computation and memory storage, it is desired to evaluate their hardware cost. In this paper, we use a circuit-level macro model, i.e. NeuroSim, to benchmark the circuit-level performance metrics, such as chip area, latency, and dynamic energy for the XNOR-RRAM and conventional 8-bit RRAM architectures. Both architectures are implemented to process the dot-product operation of a 512×512 synaptic matrix in sequential row-by-row and parallel read-out fashion separately. The simulation results are based on RRAM models and 32nm CMOS PDK, the energy-efficiency of the parallel XNOR-RRAM architecture could achieve 311 TOPS/W, showing at least ~15× and ~621× improvement compared to the parallel and sequential conventional 8-bit RRAM architectures respectively.**

***Keywords*—non-volatile memory, machine learning, hardware accelerator, neuromorphic computing**

I. Introduction

Recent advances in deep neural networks (DNN) have achieved remarkable success in various area, including speech and image recognition. However, such applications that implemented with conventional CPUs/GPUs or FPGAs which are based on traditional von Neumann architecture are no longer adequate for the state-of-the-art DNN, due to the high requirement of bandwidth, memory storage capacity and power consumption for the data communication during weighted sum and weight update. Thus, the ultimate goal of hardware accelerator design for neural networks is to efficiently implement the entire learning algorithms on-chip to achieve significant computation speed-up and low power consumption.

Lots of effort has been made to design large-scale neuromorphic hardware accelerators in recent years, e.g. TPU [1], TrueNorth [2], Eyeriss [3], etc. These systems are custom-designed based on CMOS ASIC technology, and could operate more efficiently in terms of speed and power consumption. However, the synaptic matrix are normally stored in 6-transistor or 8-transistor SRAM arrays which are not area-efficient, since typically one single SRAM cell could occupy $100F^2 \sim 200F^2$ (F is the lithography feature size). Thus, researchers have proposed the crossbar array based on resistive random access memory (RRAM) to implement synaptic arrays. RRAM is not only area-efficient (with size $4F^2 \sim 12F^2$ per cell), but can also achieve multilevel per bit by exploiting the multi-conductance-state, which makes it attractive to store "analog" synaptic weights with higher density [4].

Several RRAM-based hardware accelerators for DNN such as ISAAC [5] and PRIME [6] have been proposed to implement "analog" weight matrix (16-bit synaptic weights in ISAAC and 8-bit synaptic weights in PRIME), it was suggested that such architectures could potentially achieve significant improvement in throughput, energy, and computational density comparing with CMOS ASIC designs at the same technology node. Meanwhile, deep learning researchers have shown that Binary Neural Network (BNN) [7] could achieve satisfying classification accuracy on many representative image datasets, such as MNIST, CIFAR-10 and ImageNet. Thus, correspondingly the XNOR-RRAM architecture [8] has been proposed to replace high-precision dot-product operations with bit-wise XNOR and bit-counting operations. It could help to dramatically decrease the hardware resources and significantly improve the computational energy-efficiency since the synaptic weights and neuron activations are binarized to "-1" and "+1".

In this work, we utilize a circuit-level macro model, i.e. NeuroSim [9] to benchmark the conventional 8-bit RRAM and XNOR-RRAM architectures in sequential row-by-row and parallel read-out fashion separately. The case study is to process the dot-product operation of a 512×512 synaptic matrix, it means that the weight matrix is too large to be stored in one single RRAM array, which may cause slow-accessing and extra energy consumption. Thus, array partitioning [10] is desired to improve the overall architecture performance by paralleling the computation efficiently.

The rest of the paper is organized as follows: Section II introduces the background of RRAM-based synaptic array. In section III, we describe the conventional 8-bit RRAM and XNOR-RRAM architectures in sequential row-by-row and parallel read-out fashion. Section IV discusses the benchmark results of the architectures mentioned above with the support of NeuroSim. Finally section V concludes the paper.

II. Background

Fig. 1 shows the basic functional unit of a RRAM-based accelerator, which is called pseudo-crossbar array [11]. As one type of non-volatile memory that stores information by changing cell resistances, every RRAM cell in the pseudo-

978-1-5386-8240-1/18/$31.00 ©2018 IEEE

crossbar array can be used to represent the elements of a matrix. The input vector is represented as voltage inputs of the bit-lines (BLs), such that the dot-product value will be the current passing through the RRAM cells that sharing one sense-line (SL), and can be obtained by a current (or voltage) current sense amplifier (CSA) or analog to digital converter (ADC).

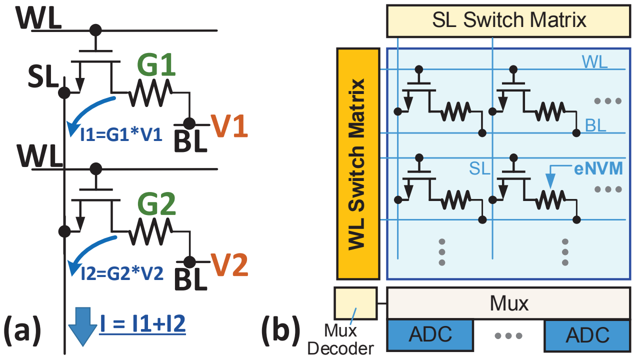

Fig. 1. (a) Using a sense-line (SL) to perform sum of dot-products. (b) Pesudo-crossbar array with Switch Matrix, MUX+MUX Decoder and ADC.

This pseudo-crossbar array can naturally perform "analog" matrix-vector multiplication. To address this, we can either use a high-precision DAC to provide the input voltages with multiple voltage levels, or we can represent the inputs as multiple sequential pulse cycles with a single fixed voltage level. We employed the multiple-cycle design as it does not introduce the distortion to the dot-product due to the nonlinear I-V characteristics of RRAM [12]. The "analog" synaptic weights can be represented by multi-bit RRAM cells, however it is impractical to represent a high-precision synaptic weight (e.g. 16-bit) by a single RRAM cell, thus we can use several multi-bit RRAM cells in a row to represent a high-precision synaptic weight (e.g. use 4× 4-bit RRAM cells to represent a 16-bit synaptic weight).

In this work, we chose to implement an 8-bit neuron activation by 8-bit sequential voltages, representing LSB to MSB of the fixed-point input, which means to get one dot-product result, we need to process at least 8 cycles. At first cycle, the first partial sum will be read out by the ADC and stored in the register, at following cycles, the partial sums will be shifted and accumulated to the earlier results that stored in the register to get the final dot-product. To investigate how the number of RRAM bits affect corresponding circuit-level performance, we benchmarked three cases, i.e. to represent the 8-bit synaptic weights with 8× 1-bit RRAM cells, 4× 2-bit RRAM cells and 2× 4-bit RRAM cells separately.

III. RRAM-BASED ARCHITECTURE DESIGN

In this work, we implemented four architectures, including sequential and parallel XNOR-RRAM, sequential and parallel conventional 8-bit RRAM architectures as shown in Fig. 2.

For the XNOR-RRAM [8], the neuron activations and synaptic weights are binarized to "+1" and "-1". One synaptic weight is represented by two RRAM cells in column. Synaptic weight "-1" is represented by such RRAM pair where the top one is high resistance state (HRS) and bottom one is low resistance state (LRS), i.e. (HRS; LRS), inversely synaptic weight "+1" is represented as a (LRS; HRS) RRAM pair. Similarly, neuron activations "-1" and "+1" are also represented respectively as input voltage pairs (0; 1) and (1; 0). Thus, it is straight-forward to read out row-by-row sequentially, while sensing current passing through LRS represents "+1", and sensing current passing through HRS represents "-1". As Fig. 2 (a) shows, the partial sums will be accumulated by adders and stored in registers till reading out all the rows, then the total weighted-sum will be binarized by the digital comparators. When it is parallel read-out, the current flowing through the SLs (I_{SL}) is dependent on the combination of WL input patterns and RRAM cell patterns, since the LRS cells will dominate the total current, it can be considered that I_{SL} is proportional to the number of LRS cells along the column. Thus, the reference (I_{ref}) of ADC for parallel mode should be set to the current value when half of the activated cells are LRS in the column. For example, if

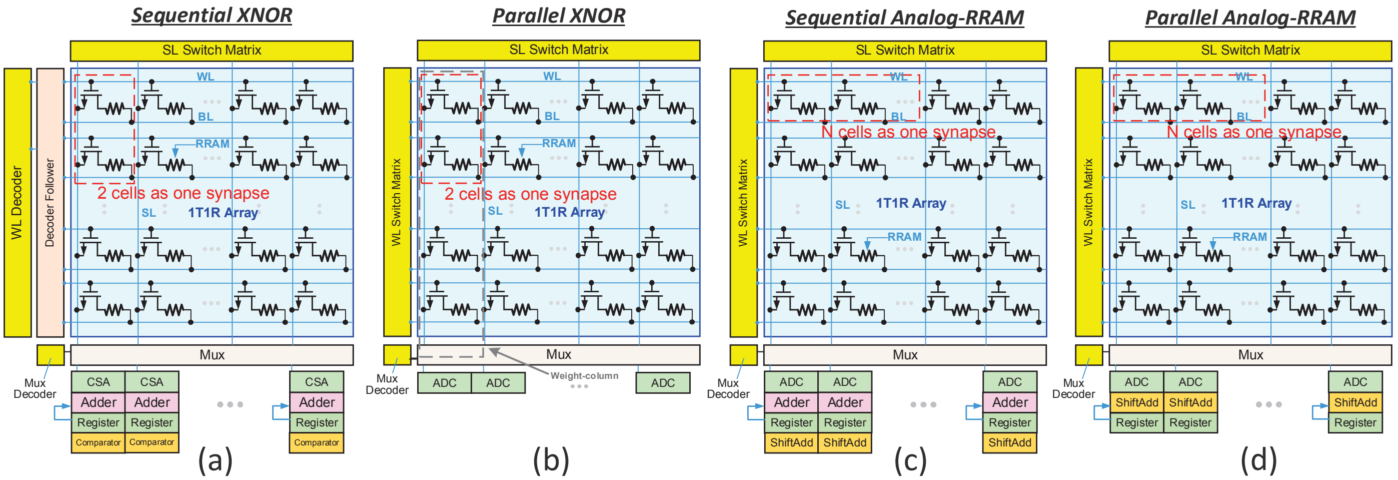

Fig. 2. The diagram of (a) sequential XNOR-RRAM architecture; (b) parallel XNOR-RRAM architecture; (c) conventional sequential 8-bit RRAM architecture; (d) conventional parallel 8-bit RRAM architecture.

there are more "+1" dot-products than "-1", the current I_{SL} is larger than I_{ref}, and ADC will give out "+1".

For the conventional 8-bit RRAM architectures, the fixed-point neuron activations are mapped as 8-bit sequential voltages, which means it needs 8 cycles to get the total weighted-sum. The synaptic weights are implemented as 8× 1-bit RRAM cells (or 4× 2-bit cells, 2× 4-bit cells) in a row, representing from the LSB to MSB. Fig. 2 (c) shows the sequential mode, where the adders and registers are used for row-by-row sequential read-out and accumulation. After reading the whole array, the partial sum will go through the shift-add modules, which are used to shift and accumulate the partial sums of the 8-bit sequential input voltages during 8 cycles. One should notice that, when we use several multi-bit RRAM cells to form the 8-bit synaptic weights, i.e. 4× 2-bit cells or 2× 4-bit cells, correspondingly, the ADC should also be set to multi-bit precision, i.e. either 2-bit or 4-bit ADC, to guarantee the computation accuracy. Similarly, Fig. 2 (d) shows the parallel architecture, which does not need adders and registers for row-by-row accumulation, instead, it requires high-precision ADC to read out the weighted-sum of one column, which is capable of truncating the weighted-sum to a reasonable precision and guarantee the computation accuracy.

When the matrix size is larger (e.g. 512×512), the corresponding peripheral circuits will be heavier, it will take longer time to process a larger array, and more importantly, it will require an ADC with much higher precision for parallel read-out, which is impractical for circuit design. Thus, array partition is a promising solution to save the area, latency and energy. In this way, the entire matrix can be implemented by several sub-arrays, and the output of each sub-array is just the partial weighted-sum. While all the sub-arrays can operate in parallel simultaneously, the partial weighted-sum will go through some adder trees to get the total weighted-sum. Such that, even though the matrix size is much larger, the total latency is still quite short, i.e. the sum of latency for single sub-array operation and the latency of adder trees.

IV. Simulation setup and results

To benchmark the RRAM-based architectures mentioned above, we customized a circuit-level macro model called NeuroSim [9] to estimate the circuit-level performance, including area, latency, dynamic energy and leakage power. The case study is to process a 512×512 dot-product operation, while we assumed various design options from device level to circuit level.

Table I shows the key parameters of simulation. Considering the array partitioning, we fixed sub-array size to be 256×256, and the partial weighted-sums of each sub-array will be summed up by the adder trees at the end. For different architecture and design options (e.g. RRAM cell precision, or ADC precision) which will affect the output precision of each sub-array, the number of bits of corresponding adder trees should also be different. Since the ADC layout area is relatively larger than a single RRAM cell, it is impractical to implement an ADC to each column, thus we assumed that every 8 columns sharing 1 ADC, which helps to save the chip area, but as a trade-off, we need to process 8 cycles to read out all the columns.

TABLE I. SIMULATION PARAMETERS

Parameters	Values
Sub-array size	256×256
Technology node	32 nm
RRAM resistance	100 kΩ / 10 MΩ
Number of columns share 1 ADC	8
Read activity	50%
Read voltage	0.5 V
ADC precision of parallel XNOR	4-bit
CSA precision of sequential XNOR	1-bit
ADC precision of parallel_1bit_cell	7-bit
ADC precision of parallel_2bit_cell	8-bit
ADC precision of parallel_4bit_cell	9-bit
CSA precision of sequential_1bit_cell	1-bit
ADC precision of sequential_2bit_cell	2-bit
ADC precision of sequential_4bit_cell	4-bit

Table II summarizes the benchmark results for 8 cases, including sequential and parallel XNOR-RRAM; sequential conventional 8-bit RRAM architectures implemented by 1-bit, 2-bit and 4-bit RRAM cells separately; parallel conventional 8-bit RRAM architectures implemented by 1-bit, 2-bit and 4-bit RRAM cells separately. It shows that the parallel XNOR-RRAM greatly reduces the latency by ~176× and the energy-efficiency could be improved by >37× compared to the sequential XNOR-RRAM.

For the parallel conventional 8-bit RRAM architectures, as we use higher-precision RRAM cells to implement the synaptic weights, the actual total number of RRAM cells becomes less, consequently the peripheral circuits are less. Thus, it shows that when we use 4-bit RRAM cells, the circuit-level performance is the best among the three cases, the energy-efficiency can achieve ~20 TOPS/W. However, compared to the parallel XNOR-RRAM, the area is ~5× larger and energy-efficiency is ~15× lower.

Although the sequential conventional 8-bit RRAM architectures are more area-efficient compared to the parallel ones (because ADCs dominant the area while sequential ones have much lower-precision ADCs), the latency and energy-efficiency are still much worse since sequential ones need extra time and energy to process row-by-row. It should be noticed that, different from the parallel modes, when the number of bits of the RRAM cell becomes higher, the energy-efficiency becomes worse (0.5>0.27>0.17 TOPS/W). This is because the higher-bit RRAM cells required higher-precision ADCs (which consume higher energy) while the ADCs dominant the total energy.

TABLE II. BENCHMARK OF ARCHITECTURES FOR 512×512 DOT-PRODUCT

	Area (μm^2)	Latency (ns)	Energy (pJ)	TOPS/ W
Parallel-XNOR	36618.4	30.5	840.3	**311.95**
Sequential-XNOR	31030.3	5293.4	31560.5	**8.31**
Parallel-1bit-cell	322246.7	742.3	50213.7	**5.22**
Parallel-2bit-cell	213853.8	428.1	25382.7	**10.33**
Parallel-4bit-cell	188375.4	272.9	13095.7	**20.02**
Sequential-1bit-cell	154802.2	16155.5	522544.7	**0.50**
Sequential-2bit-cell	89557.0	15806.9	987954.7	**0.27**
Sequential-4bit-cell	67342.0	15912.0	1550660.5	**0.17**

V. CONCLUSION

In this paper, we benchmarked XNOR-RRAM and conventional 8-bit RRAM architectures in sequential row-by-row and parallel read-out fashion. To analyze the impact of RRAM precision, we have also set several cases, i.e. use 1-bit, 2-bit and 4-bit RRAM cells to form 8-bit synaptic weights separately. We used NeuroSim as a handy tool to estimate the hardware performance, i.e. area, latency and dynamic energy to investigate the trade-offs among those architectures. The simulation result shows that the parallel XNOR-RRAM is capable to achieve the best performance, while the energy-efficiency to process a 512×512 dot-product operation is ~311 TOPS/W at 32nm PDK, which shows at least ~15× and ~621× improvement compared to the parallel and sequential conventional 8-bit RRAM architectures respectively.

ACKNOWLEDGEMENT

This work is supported in part by NSF-CCF-1552687, and NSF/SRC E2CDA program with NSF-CCF-1740225 and SRC Contract 2018-NC-2762.

REFERENCES

[1] N. P. Jouppi, et al. , "In-datacenter performance analysis of a tensor processing unit," *ACM/IEEE International Symposium on Computer Architecture (ISCA)*, 2017.

[2] F. Akopyan et al., "TrueNorth: design and tool flow of a 65 mW 1 million neuron programmable neurosynaptic chip," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 34, no. 10, pp. 1537-1557, Oct. 2015.

[3] Y. H. Chen, J. Emer and V. Sze, "Eyeriss: A spatial architecture for energy-efficient dataflow for convolutional neural networks," *ACM/IEEE International Symposium on Computer Architecture (ISCA)*, pp. 367-379, 2016.

[4] S. Yu, P.-Y. Chen, Y. Cao, L. Xia, Y. Wang, and H. Wu, "Scaling-up resistive synaptic arrays for neuro-inspired architecture: Challenges and prospect," *IEEE International Electron Devices Meeting (IEDM)*, pp. 451-454, 2015.

[5] A. Shafiee et al., "ISAAC: A convolutional neural network accelerator with in-situ analog arithmetic in crossbars," *ACM/IEEE International Symposium on Computer Architecture (ISCA)*, Seoul, 2016, pp. 14-26.

[6] P. Chi et al., "PRIME: A novel processing-in-memory architecture for neural network computation in ReRAM-based main memory," *ACM/IEEE International Symposium on Computer Architecture (ISCA)*, pp. 27-39, 2016.

[7] M. Courbariaux, et al., "Binarized neural network: Training deep neural networks with weights and activations constrained to+ 1 or-1," arXiv: 1602.02830, 2016.

[8] X. Sun, S. Yin, X. Peng, R. Liu, J.-s. Seo and S. Yu, "XNOR-RRAM: A scalable and parallel resistive synaptic architecture for binary neural networks," *ACM/IEEE Design, Automation & Test in Europe Conference (DATE)*, pp. 1423-1428, 2018.

[9] P.-Y. Chen, X. Peng, S. Yu, "NeuroSim+: An integrated device-to-algorithm framework for benchmarking synaptic devices and array architectures," *IEEE International Electron Devices Meeting (IEDM)* 2017.

[10] P.-Y. Chen and S. Yu, "Partition SRAM and RRAM based synaptic arrays for neuro-inspired computing," *IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 2310-2313, 2016.

[11] S. Yu, "Neuro-inspired computing with emerging non-volatile memory," *Proc. IEEE*, vol. 106, no. 2, pp. 260-285, 2018.

[12] P.-Y. Chen, D. Kadetotad, Z. Xu, A. Mohanty, B. Lin, J. Ye, S. Vrudhula, J.-S. Seo, Y. Cao, S. Yu, "Technology-design co-optimization of resistive cross-point array for accelerating learning algorithms on chip," *ACM/IEEE Design, Automation & Test in Europe (DATE)*, 2015.