High-Throughput In-Memory Computing for Binary Deep Neural Networks With Monolithically Integrated RRAM and 90-nm CMOS

Shihui Yin, *Graduate Student Member*, Xiaoyu Sun, *Graduate Student Member, IEEE*, Shimeng Yu, *Senior Member, IEEE*, and Jae-sun Seo, *Senior Member, IEEE*

***Abstract*— Deep neural network (DNN) hardware designs have been bottlenecked by conventional memories, such as SRAM due to density, leakage, and parallel computing challenges. Resistive devices can address the density and volatility issues but have been limited by peripheral circuit integration. In this work, we present a resistive RAM (RRAM)-based in-memory computing (IMC) design, which is fabricated in 90-nm CMOS with monolithic integration of RRAM devices. We integrated a 128 × 64 RRAM array with CMOS peripheral circuits, including row/column decoders and flash analog-to-digital converters (ADCs), which collectively become a core component for scalable RRAM-based IMC for large DNNs. To maximize IMC parallelism, we assert all 128 wordlines of the RRAM array simultaneously, perform analog computing along the bitlines, and digitize the bitline voltages using ADCs. The resistance distribution of low-resistance states is tightened by an iterative write-verify scheme. Prototype chip measurements demonstrate high binary DNN accuracy of 98.5% for MNIST and 83.5% for CIFAR-10 data sets, with 24 TOPS/W and 158 GOPS. This represents 22.3× and 10.1× improvements in throughput and energy–delay product (EDP), respectively, compared with the state-of-the-art literature, which can enable intelligent functionalities for area-/energy-constrained edge computing devices.**

***Index Terms*— Deep neural networks (DNNs), in-memory computing (IMC), monolithic integration, nonvolatile memory (NVM), resistive RAM (RRAM).**

Manuscript received March 28, 2020; revised June 18, 2020 and July 23, 2020; accepted August 3, 2020. Date of publication August 19, 2020; date of current version September 22, 2020. This work was supported in part by NSF-SRC-E2CDA under Contract 2018-NC-2762B; in part by NSF under Grant 1652866, Grant 1715443, and Grant 1740225; in part by JUMP C-BRIC; and in part by JUMP ASCENT (SRC Program sponsored by the Defense Advanced Research Projects Agency (DARPA)). The review of this article was arranged by Editor T.-H. Kim. *(Corresponding author: Jae-sun Seo.)*

Shihui Yin and Jae-sun Seo are with the School of Electrical, Computer and Energy Engineering, Arizona State University, Tempe, AZ 85287 USA (e-mail: shimeng.yu@ece.gatech.edu; jaesun.seo@asu.edu).

Xiaoyu Sun and Shimeng Yu are with the School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA 30332 USA.

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TED.2020.3015178

I. Introduction

DEEP neural networks (DNNs) have been very successful in large-scale recognition and classification tasks [1]–[5], while state-of-the-art deep learning algorithms tend to present very deep and large network models [1]–[3]. This poses significant challenges for embedded hardware implementations [6], [7] in terms of computation, memory, and communication. To address this on the algorithm side, recent works aggressively lowered the precision to the extreme where both the weights and neuron activations are binarized to $+1$ or -1 [8], [9] for inference, such that the multiplication between weights and activations becomes XNOR operation and accumulation becomes bitcounting of bitwise XNOR values. Such binarized neural network (BNN) algorithms largely reduce the computational complexity and weight memory requirement.

On the hardware side, a number of application-specific integrated circuit (ASIC) solutions in CMOS [10], [11] were presented, but data storage and communication became the bottleneck for energy-efficient computing [10]. Although SRAM technology followed CMOS scaling well [12], SRAM density ($\sim$150 F^2 per bitcell) and on-chip SRAM capacity (a few MB) are insufficient to hold a large number of DNN parameters (even with binary precision), leakage current is undesirable, and parallelism is limited due to row-by-row operation [13].

As an alternative hardware platform, emerging resistive devices have been proposed for dense weight storage and parallel neural computing for matrix–vector multiplications [14]–[20]. However, a number of limitations still exist for resistive RAMs (RRAMs) for practical large-scale neural computing due to (device-level nonidealities (e.g., variability and endurance), inefficiency in representing/multiplying negative weights, and monolithic integration of RRAMs and CMOS peripheral circuits.

Due to these limitations, the literature on RRAM-based DNN hardware has mostly implemented simpler multilayer perceptrons (MLPs) [17], [19], with limited implementation of mainstream convolutional neural networks (CNNs). In addition, a number of RRAM works were demonstrated without proper peripheral circuitries monolithically integrated into the

same technology [17], while peripheral circuits can dominate the chip area [21]. An RRAM macro with multilevel sense amplifiers in 55-nm CMOS was recently presented [22], but a relatively low accuracy of 81.83% for the CIFAR-10 data set was reported with binary/ternary precision, and only 9 (out of 256) rows are asserted simultaneously, limiting high parallelism.

In this work, we address such limitations in RRAM-based in-memory computing (IMC) for DNNs/CNNs. We adhere to binary RRAM devices (low-/high-resistance states (LRS/HRS) with high ON/OFF ratio) and one-transistor-one-resistor (1T1R) structure for robustness against noise/variability and ease for integration. Using binary RRAM devices, we present new RRAM bitcell/array designs that can efficiently map XNOR functionality with binarized (+1 and −1) weights/neurons and are suitable for IMC of binarized DNNs.

This work builds upon our preliminary work [23], where the bitcell design is common. However, [23] is a simulation-only work in 65-nm CMOS using ideal resistor models for the RRAM devices. In this work, we report the implementation results of the prototype chip we fabricated in monolithically integrated 90-nm CMOS and RRAM technology, with full peripheral circuits for the 128 × 64 RRAM macro, including analog-to-digital converters (ADCs) with more robust quantization scheme, row/column decoder, wordline (WL) drivers, column multiplexers, level shifters, and scan circuits. The 128 × 64 RRAM array that we integrated with CMOS peripheral circuits, including row/column decoders and flash ADCs, could collectively be a core component for large-scale RRAM-based IMC.

In another preliminary work [24], we presented high-level architecture exploration together with RRAM design space, algorithm techniques, and NeuroSim [25] tool evaluation. On the other hand, this work focuses more on the device optimization and device-circuit codesign, where we present the RRAM device programming optimization and results, prototype chip design, IMC measurements, power/energy characterizations, and DNN accuracy. Based on chip measurement results, we demonstrate deep CNNs for CIFAR-10 [26] and MLPs for MNIST [27] data sets with high classification accuracy and energy efficiency.

II. XNOR-RRAM Macro Design and Optimization

A. XNOR-RRAM Bitcell Design

Conventional binary RRAMs cannot effectively represent the positive and negative weight values (+1 and −1) in recent BNNs [8], [9] because the LRS and HRS values of binary RRAM devices are both positive. In addition, the activation/weight value combinations of +1/+1 and −1/−1 should result in the same effective resistance [see Fig. 1(c)]. To that end, for XNOR-Net [8] type of BNNs, we proposed to use the "XNOR-RRAM" bitcell design in [23], which was a preliminary simulation study with ideal RRAM device models and variability-prone current-mode sense amplifiers [28]. In this work, we implemented the XNOR-RRAM prototype chip in Winbond's 90 nm nonvolatile memory (NVM) technology [29], employed more robust voltage-mode sense

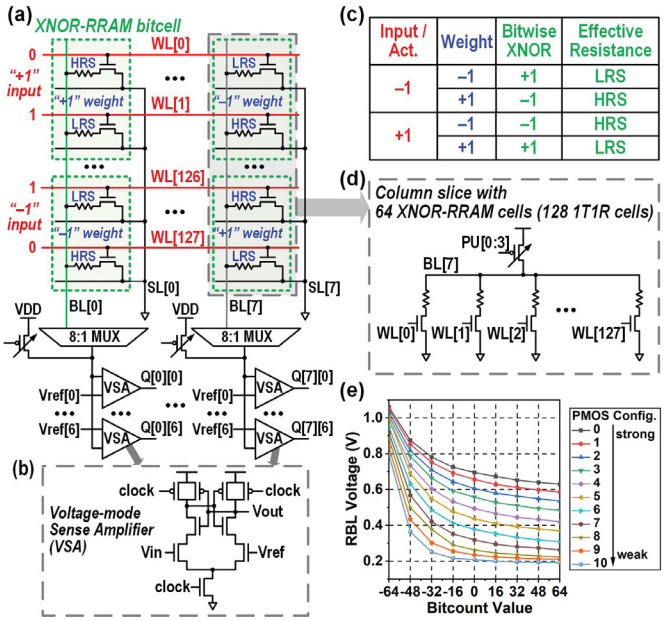

Fig. 1. (a) Column schematic with XNOR-RRAM cells. (b) VSA schematic for the flash ADC. (c) XNOR-RRAM cell operation. (d) Resistive divider between pMOS header and 64 parallel XNOR-RRAM cells. (e) Column measurements with different pMOS strengths.

amplifiers (VSAs) for the flash ADC, and integrated all peripheral circuits necessary for IMC (row/column decoders and multiplexers) as well as device programming (level shifters). As shown in Fig. 1(a), the XNOR-RRAM cell involves differential RRAM cells and differential WLs. Fig. 1(a) shows the XNOR-RRAM cell that consists of two 1T1R cells. The binary activations are mapped onto the differential WLs, and the binary weights are mapped onto the HRS/LRS values of XNOR-RRAM cells. By asserting all WLs of the RRAM array simultaneously, all cells in the same column are computed in parallel, implementing binary multiply-and-accumulate (bMAC) computations. The 128 × 64.1T1R array effectively represents 64 × 64 XNOR-RRAM cells. The area of the 1T1R bitcell that we used is ∼0.5 μm × 0.5 μm (∼31 F^2), and hence, one XNOR-RRAM cell area is ∼62 F^2.

B. Proposed In-RRAM Computing and Read Disturb

The proposed IMC with XNOR-RRAM array [see Fig. 1(a)] features a static pMOS header and parallel XNOR-RRAM cells that perform bitwise XNOR operations [see Fig. 1(d)]. A static pMOS header, the strength of which is digitally configurable, pulls up the read bitline (RBL) voltage. The RRAM cells in the same column pull down the RBL voltage in parallel. Depending on how many cells with high WL voltage are in LRS or HRS, a static resistive divider is formed between the pMOS head and the pull-down path based on the parallel RRAM cells. As more RRAM cells are in LRS (higher bitcount value from the algorithm), RBL voltage will be lower.

The measured transfer function with different pMOS header strengths [see Fig. 1(e)] shows that a stronger pMOS increases the RBL voltage for the same bitcount value. We achieved

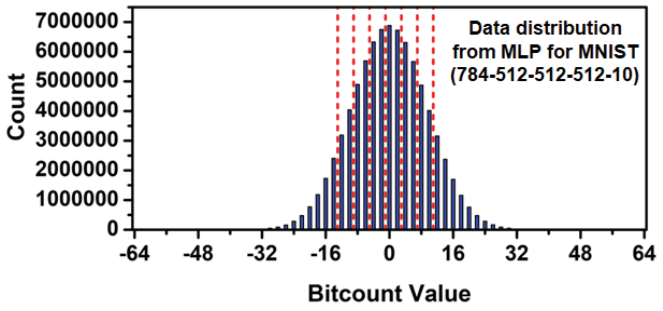

Fig. 2. Based on the bitcount distribution from DNN workload, linear quantization within a confined range is performed.

better DNN accuracies with transfer curves that have the steepest slope around the bitcount value of 0 since the flash ADC reference voltages could be separated further compared with the cases where the transfer curves have more gradual slope. Therefore, we chose to use the medium-strength pMOS configuration of 4 or 5 [in Fig. 1(e)] for our DNN workloads.

Although it has been reported that high RBL voltage can cause read disturb issues in RRAMs [30], read disturb is largely prevented in our XNOR-RRAM design for two reasons. First, Fig. 1(e) shows that a relatively high RBL voltage of >0.6 V only occurs for bitcount values smaller than −32. In this range, there is only <0.046% data according to the bitcount distribution in Fig. 2. Second, we experimentally observed that RRAM cells whose HRS resistance is larger than 1 MΩ are stable and are not susceptible to read disturb issues even with high RBL voltages of >0.6 V. On the other hand, we did observe that the outlier HRS cells with <1 MΩ resistance can experience read disturb with high RBL voltages. However, our RRAM device programming results (see Fig. 6) show that only <1% of the programmed HRS cells exhibit less than 1-MΩ resistance. Considering these two reasons, the probability that read disturb will occur becomes extremely low (e.g., <0.00046*0.01) in our XNOR-RRAM array.

C. ADC Design and Optimization

Each VSA compares the RBL voltage of the selected column with a reference voltage (V_{ref}). Seven V_{ref}'s of an ADC are calibrated for the eight columns that the ADC is connected to. By running the 784-512-512-512-10 MLP for MNIST, we first characterized the distribution of ideal bitcount values that should be obtained from XNOR-RRAM arrays. As shown in Fig. 2, the bitcount data distributions are highly centered around 0. Based on this data distribution, out of the possible bitcount range between −64 and +64, the reference bitcount values are chosen in a confined range between −15 and 13, where most data resides. After experimenting different candidates, the optimal seven reference bitcount values we chose are: −13, −9, −5, −1, 3, 7, and 11 (red dashed lines in Fig. 2).

In comparison, we performed software simulation with ideal quantization (no ADC offset) by using linear quantization for the full range of bitcount values from −64 to +64. For 3-, 4-, and 5-bit ADCs with "full-range" linear quantization, we obtained the CNN accuracies of 45.56%, 85.84%, and 88.59% for CIFAR-10. For 3-bit ADC with the "confined-range" linear quantization, we achieved 86.70% accuracy, which is even higher than that of 4-bit ADC with full-range quantization.

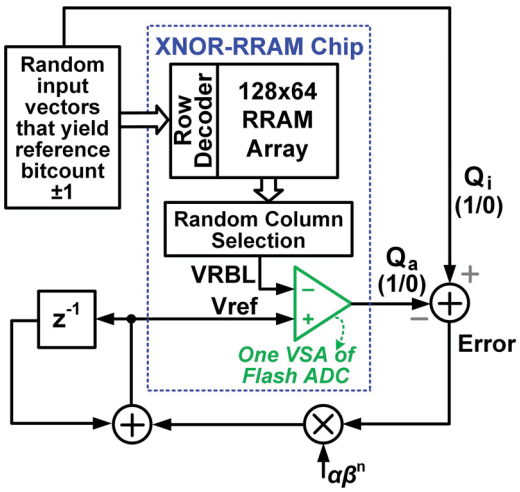

Fig. 3. ADC reference voltage calibration flow.

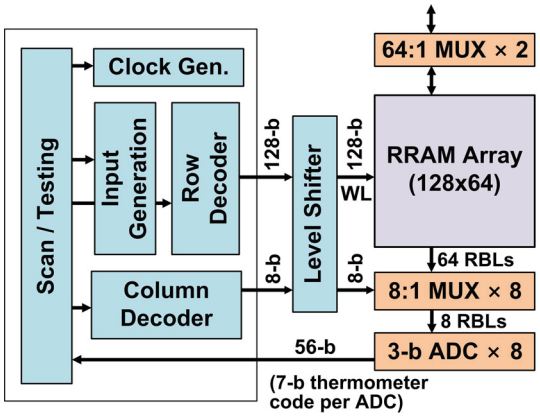

Fig. 4. Top-level block diagram of the prototype chip.

On the other hand, compared with nonlinear quantization schemes [23], [31], the proposed confined/linear quantization scheme simplifies the ensuing accumulation of ADC outputs (partial sums) and also increases the smallest V_{ref} difference for the adjacent sense amplifiers in the flash ADC.

An automatic algorithm (see Fig. 3) is employed to determine the optimal set of V_{ref}'s for the flash ADCs, to compensate for circuit nonideal factors, such as RRAM resistance variation and comparator offset/noise. For each reference bitcount value, we randomly generate 1000 input vectors for the adjacent bitcount1 values. V_{ref} is increased (or decreased if the correction amount is negative) by $\alpha\beta^n \times (Q_i - Q_a)$, where Q_a is the actual ADC output, Q_i is the ideal ADC output, α is initial correction step size (e.g., 5 mV), β is a scaling factor (e.g., 0.995) that is less than 1, and n is the iteration index.

D. XNOR-RRAM Macro Design and Operation

As shown in Fig. 4, the XNOR-RRAM prototype chip includes a 128 × 64.1T1R array, row decoder, level shifter,

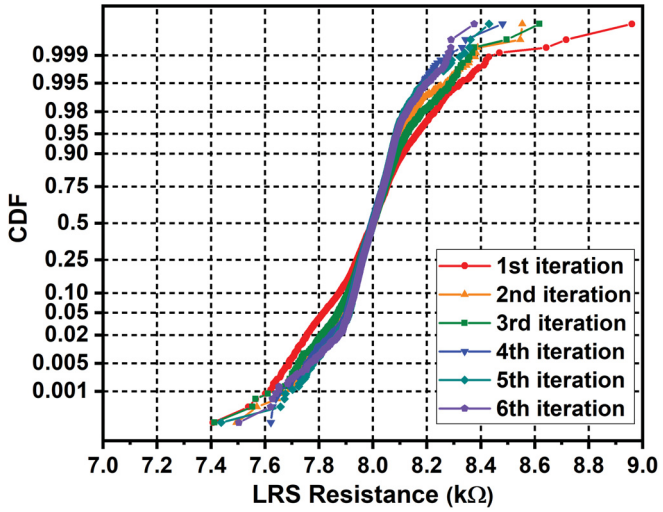

Fig. 5. As we repeatedly program the RRAM array with write-verify iterations, LRS distribution becomes further tightened.

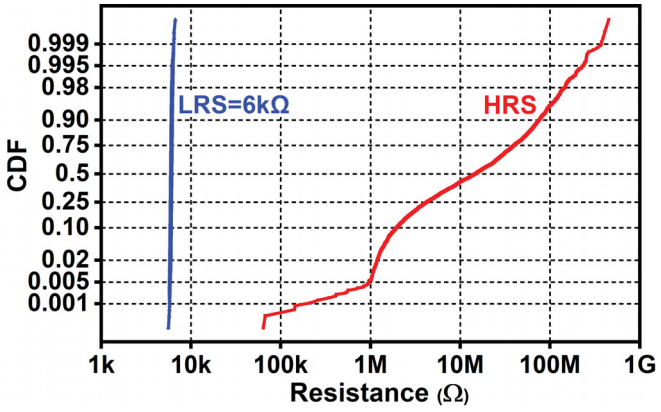

Fig. 6. Programming results and distributions of RRAM devices for XNOR-RRAM array.

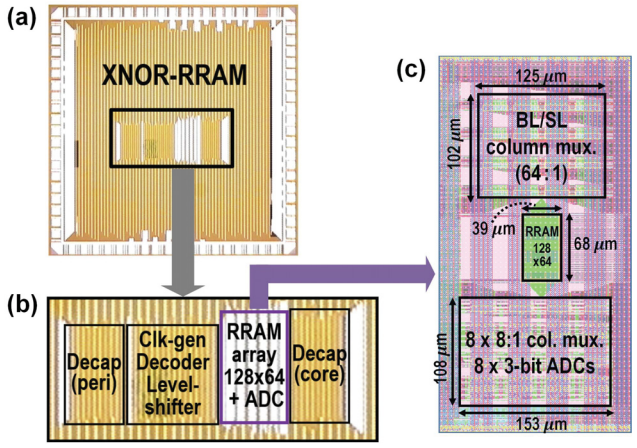

Fig. 7. (a) Pad-limited prototype chip micrograph. (b) Core area consisting of RRAM array and CMOS peripheral circuits. (c) Layout and dimensions of RRAM array, multiplexers, and flash ADC.

eight 8-to-1 column multiplexers, eight 3-bit flash ADCs, and two 64-to-1 column decoders for RRAM cell-level programming. To make a balance between area and throughput, we share one flash ADC by eight columns. Since XNOR-RRAM array has 64 columns, there are a total of eight flash ADCs.

For the functionality test, a 64-bit input vector is fed through a scan chain and the ADC outputs can be read out through the scan chain. For power measurement, random 64-bit input vectors are generated by linear-feedback shift register (LFSR) every eight cycles. The row decoder has two modes of operation: 1) it asserts all differential WL signals simultaneously for bMAC operations or 2) it generates one-hot WL signals for cell-level programming.

E. LRS and HRS Programming

For our application of mapping BNNs onto the XNOR-RRAM array, tightening LRS distribution is very important because the column current will be dominated by current through LRS cells. To that end, we set the target LRS resistance to be in a tight range of 5.9–6.1 kΩ. To achieve this, we apply an aggressive write-verify scheme. First, we set the initial gate voltage to 2.3 V and apply a 100-ns SET pulse with an amplitude of 2.1 V. If the resistance after SET is lower than the lower bound, i.e., 5.9 kΩ, a 200-ns RESET pulse with an amplitude of 3.8 V and a gate voltage of 4.0 V is applied to the RRAM cell followed with a SET pulse with a 0.05-V lower gate voltage; if the resistance after SET is higher than the upper bound, i.e., 6.1 kΩ, a RESET pulse is applied to the RRAM cell followed with a SET pulse with a 0.05-V higher gate voltage. We repeated the previous steps for up to ten times until the LRS resistance falls in the target range. In Fig. 5, we show how the LRS distributions changed after iterative write-verify operations.

For HRS, we set the target HRS resistance value to be above 1 MΩ. To achieve this, we apply a 200-ns RESET pulse with an amplitude of 3.8 V and a gate voltage of 4 V to the RRAM cell and repeat applying the same RESET pulse up to ten times until the resistance value is greater than 1 MΩ.

Fig. 6 shows the final RRAM device programming results of LRS and HRS distribution for the 128 × 64 array, where 4096 RRAM cells are programmed in LRS and 4096 RRAM cells in HRS. Less than 1% of HRS resistance values are lower than 1 MΩ, and more than 99% of LRS resistance values are in the range of 5.7–6.3 kΩ.

The resistance values are read at 0.2 V by a source measurement unit (SMU). Although we go through up to ten times of SET/RESET operations for the initial programming, since we will not reprogram the weights often for DNN inference applications, the endurance of $>10^5$ cycles reported by Winbond [29] is sufficient.

III. Measurement Results

We designed and fabricated a prototype chip with Winbond's embedded RRAM technology [29], which monolithically integrates 90-nm CMOS and RRAM between M1 and M2. The pad-limited prototype chip micrograph is shown in Fig. 7(a), and Fig. 7(b) shows the core area. Eight ADCs (shared among 64 columns) and eight column

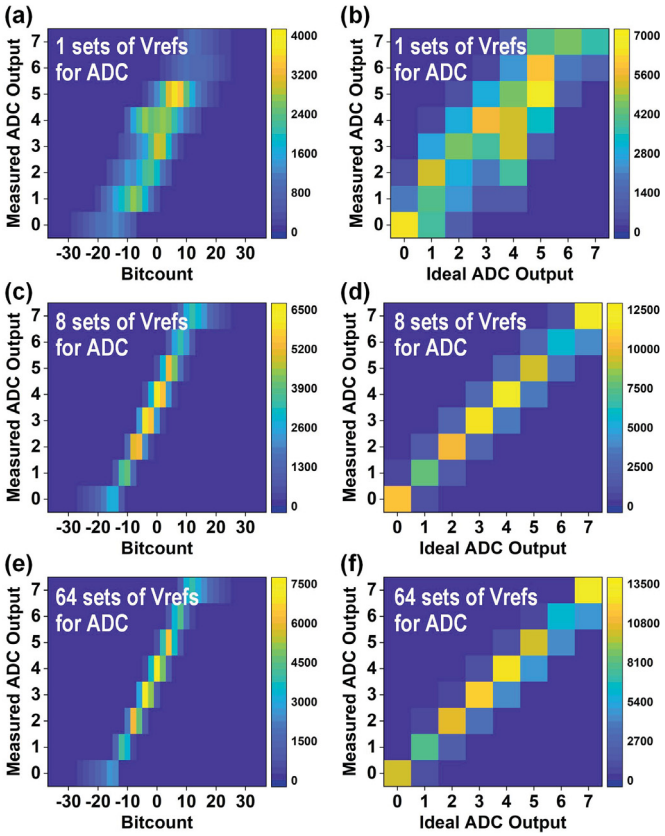

Fig. 8. Measured ADC results are compared with ideal bMAC results and ideal ADC outputs. (a) and (b) Single set of V_{ref}'s calibrated for all eight ADCs. (c) and (d) V_{ref}'s of each ADC are calibrated. (e) and (f) V_{ref}'s for each column are calibrated.

multiplexers occupy 20% and 12% area of the XNOR-RRAM core, respectively [see Fig. 7(c)].

A. In-Memory Computing Measurements

In Fig. 1(e), the measurement results of a single column were shown for RBL voltage against ideal bitcount values. This RBL voltage needs to be digitized with the ADC. We investigated three different V_{ref} schemes for the flash ADC: 1) one set of V_{ref}'s for the entire eight ADCs of the testchip; 2) eight sets of V_{ref}'s for eight ADCs (one set per ADC); and 3) 64 sets of V_{ref}'s for 64 columns (one set per column).

For these three schemes, Fig. 8 shows the comparison of measured ADC output against the bitcount values from the BNN algorithm as well as the ideal ADC output. We first programmed the XNOR-RRAM with a 64×64 weight submatrix from the trained BNN for MNIST using the aforementioned write-verify scheme; 2000 64-bit binary test vectors were then presented to XNOR-RRAM, to perform bMAC computations and obtain the 2000×64 ADC outputs. In total, 128 000 pairs of measured ADC outputs and target bitcount values are used to estimate the joint distribution.

The 2-D histograms in Fig. 8 shows how accurately the XNOR-RRAM array computes and quantizes the bMAC values. It can be seen that the bitcount values and the ADC output show an expected linear relationship. Fig. 8(a) and (b) shows that using only one set of V_{ref}'s without offset calibration can result in large variations in the ADC output. However, if each ADC has its own V_{ref}'s [see Fig. 8(c) and (d)] or exhibits offset cancellation capability, the ADC output resides in a tight range for each bMAC value. If each column has its own V_{ref}'s [see Fig. 8(e) and (f)], there is only a minor difference compared with the results in Fig. 8(c) and (d).

As an initial prototype chip design, please note that our VSA and ADC design did not include offset compensation circuits. If our VSA/ADC had employed offset cancellation circuits typically accompanied in sense amplifier designs [32], then all ADCs in our XNOR-RRAM macro would be able to use the same V_{ref}'s, enabling the IMC design with higher practicality.

In Section III-B, we first present a "direct mapping" method where we map a small MLP directly onto our XNOR-RRAM chip by programming the RRAM chip multiple times for each MLP layer and measuring IMC results. In Section III-C, we present the large BNN accuracy characterization results, where we employ a "sampling" method that leverages the aforementioned 128 000 measurement results, samples the ADC output for each 64-input bMAC of the large BNN, and digitally simulates the accumulation of partial sums (ADC outputs) and non-MAC operations.

B. DNN Accuracy Measurement With Direct Mapping

For the "direct mapping" experiment, we use a simple MLP of 784-64-64-10 for the MNIST data set. If we include batch normalization (BN), we achieve 95% accuracy, but to perform direct measurement without additional non-MAC operations, we trained the 784-64-64-10 MLP without BN, whose software accuracy is 90.8%. Similar to how the original BNN algorithms [8], [9] still use multibit precision for the primary input, note that the 784 input neurons in this simple MLP also exhibit 8-bit precision. To that end, we used digital simulation until the first hidden layer of 64 neurons, and then, we mapped the second layer of 64-64 and the third layer of 64-10 directly with our XNOR-RRAM chip. Since these second/third layers are small and can fit in our XNOR-RRAM chip, we programmed our chip two times for the second and third layers, and the outputs of the chip measurement from the second-layer programming are directly conveyed as the input activations applied for the vector–matrix multiplication (VMM) for the third-layer chip measurement. Using the aforementioned direct mapping method, the actual chip's measured accuracy is 90.05%.

If we use the "sampling" method described in Section III-C, the estimated hardware accuracy for the same 784-64-64-10 MLP is 90.01%, which is close to the actual hardware accuracy obtained by direct mapping. This shows that our hardware accuracy characterized by measurement-based sampling (see Section III-C) well represents the actual hardware accuracy.

C. DNN Accuracy Characterization With Measurement-Based Sampling

With these three schemes, we benchmarked the accuracy for deep BNNs for MNIST and CIFAR-10 data sets [see Fig. 9(a)]. For MNIST, we used an MLP with a structure of 784-512-512-512-10. For CIFAR-10, we employed a CNN

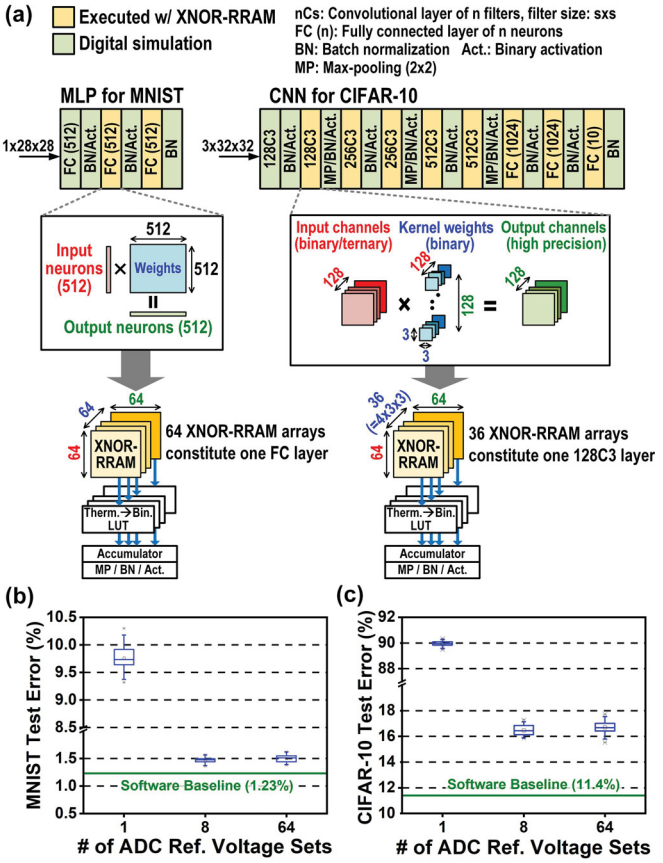

Fig. 9. Evaluation of deep BNNs. (a) BNN weights are mapped onto parallel XNOR-RRAM arrays. Test errors are evaluated on (b) MLP for MNIST and (c) CNN for CIFAR-10.

with six convolution layers and three fully connected layers [9]. Fully connected and convolution layers of MLP and CNN are mapped onto multiple XNOR-RRAM instances, where the convolution kernels and weight matrices of deep BNNs are divided into 64 × 64 weight submatrices, where VMMs of 64 input activations and 64×64 weights are mapped with IMC operations of our XNOR-RRAM array. As shown in Fig. 9(a), one fully connected layer of the MLP for MNIST with 512 input activations and 512 output activations will be mapped onto 64 XNOR-RRAM arrays, and one convolution layer of the CNN for CIFAR-10 that has 128 input channels and 128 output channels with 3 × 3 kernels will be mapped onto 36 XNOR-RRAM arrays. Weights for different input channels are stored on different rows, weights for different output channels are stored on different columns, and weights within each convolution kernel (e.g., 9 = 3 × 3) are stored in different XNOR-RRAM macros [see Fig. 9(a)].

Subsequently, the partial MAC results from different XNOR-RRAM macros are accumulated via digital simulation. For each 64-input bMAC value (partial sum) of a deep BNN, we randomly sample the ADC output distribution from the 128 000 measurement results (see Section III-A). It should be noted that analog IMC occurs only inside the XNOR-RRAM array, and the inputs (activations) and outputs (partial sums) of the XNOR-RRAM array are all digital. Since the partial sum accumulation for the final sum is all done in digital fashion, there will not be any accuracy degradation outside of the XNOR-RRAM array for the entire DNN.

To evaluate deep BNNs with a single array in the prototype chip, we ran software emulation based on the conditional probability distribution of measured ADC outputs from 2000 random test vectors (see Fig. 8). The partial sums of 64-bit inputs and 64 × 64 weight submatrices are first stochastically quantized to 3-bit according to the measured conditional probability distribution (see Fig. 8). Subsequently, the accumulation of partial sums and non-MAC operations, such as BN, max-pooling, and activation, is performed in digital simulation with high fixed-point precision.

The test error values obtained from 20 runs with different random seeds are summarized in box plots in Fig. 9(b) and (c), with the three different V_{ref} schemes for MNIST MLP and CIFAR-10 CNN, respectively. The redline, box top edge, box bottom edge, top bar, and bottom bar represent the mean, 75th percentile, 25th percentile, maximum, and minimum of the 20 data points.

Compared to the scheme with a single set of V_{ref}'s for the ADC (without offset calibration), using eight sets of V_{ref}'s for eight ADCs show considerable improvement in both MNIST and CIFAR-10 accuracies. This would be largely due to the local mismatch of the ADC, which can be compensated by offset cancellation schemes typically employed in ADC designs [32]. On the other hand, the accuracy values for the scheme using 64 sets of V_{ref}'s are hardly different to those using eight sets of V_{ref}'s. This means that column-by-column variation is small and does not affect the accuracy noticeably.

Using eight sets of V_{ref}'s, XNOR-RRAM achieves 98.5% classification accuracy for MNIST (software baseline: 98.7%) and achieves 83.5% accuracy for CIFAR-10 (software baseline: 88.6%) data sets. The accuracy degradation of CNN for CIFAR-10 occurs due to limited ADC precision (CNN baseline with ideal quantization leads to 86.70% in simulation) and small separation in adjacent V_{ref}'s of ADC (caused by the gradual slope of RBL transfer curve). These could be improved by employing an ADC with higher precision [33] (trading off ADC area and power) or asserting a less number of rows [22] in parallel to reduce the dynamic range (trading off latency or energy efficiency).

D. Performance and Energy Characterization

We measured the power of the prototype chip under different power supply voltages (1.2 down to 0.9 V) for the pMOS pull-up and ADC. As we lower the pMOS pull-up power supply voltage, the current of the voltage dividers decreases, reducing the total power and improving the energy efficiency, as shown in Fig. 10. However, as we reduce the power supply voltage, the ADC sensing margin reduces, degrading the accuracy on BNN benchmarks. For example, for MNIST MLP, the accuracy degrades to 97.28% when the power supply voltage is 0.9 V, and for CIFAR-10 CNN, the accuracy degrades to 80.65% when the power supply voltage is 1.1 V. With all 128 rows and 8 columns (8 ADCs shared among 64 columns) asserted and computed simultaneously in each cycle, the 128 × 64 XNOR-RRAM array achieves

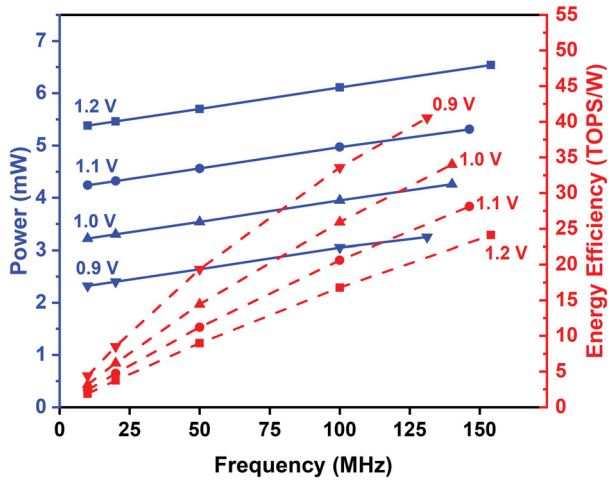

Fig. 10. Power and energy measurement results with voltage/frequency scaling.

a high throughput of 157.6 GOPS and an energy efficiency of 24.1 TOPS/W at 1.2-V supply.

E. Comparison to Prior In-RRAM Computing Work

Table I shows the comparison with the closest prior work implemented in 55-nm CMOS with embedded RRAM [22]. We defined throughput as "(number of operations)/(read IMC delay) × (number of parallel computing columns among 64 columns)." Since [22] only turns on nine rows in one cycle and each ADC connects from positive weight column and negative weight column, there are 18 (=9×2) MAC operations per ADC evaluation. In our design, we turn on all 128 rows, and since two 1T1R cells represent one weight, 64 MAC operations are performed per ADC evaluation. The read IMC delay is reported as 10.2 ns for [22] and ours is 6.5 ns. The column multiplexing ratio mentioned in [22] was 32:1, while our work's column multiplexing ratio is 8:1. In other words, our work computes 4× more columns in parallel, for a given array size. Therefore, our throughput per 128 × 64 array is 22.3× better than that of [22].

For IMC, turning on more rows typically requires ADCs with higher precision due to a higher dynamic range of the MAC results. However, we turn on all 128 rows and achieve better binary DNN accuracy than [22] even with lower precision ADC (3-bit ADC in our work versus 4-bit ADC in [22]), aided by both the confined-range linear quantization (see Fig. 2) and the ADC V_{ref} optimization (see Fig. 3).

In Table I, we reported a figure of merit (FoM) that is the product of energy efficiency (TOPS/W) and throughput per 128 × 64 array (GOPS), which effectively represents the inverse of energy–delay product (EDP). EDP is a well-known metric that reflects a balance between energy and performance for computer systems. Our work achieves 10.1× higher FoM compared with that of [22]. These improvements will be even higher if we normalize the CMOS technology (55 nm [22] versus 90 nm for our work).

Exhibiting high throughput and low EDP is essential for performance-critical or real-time embedded systems (e.g., autonomous driving, natural language processing, and real-time machine translation), and our IMC technique becomes very suitable for such latency-/energy-/area-constrained artificial intelligence systems.

TABLE I
COMPARISON WITH RECENT IN-RRAM COMPUTING WORK

	Xue et al. [22]		This work
CMOS Technology	55nm		90nm
Sub-array size	256×512b		128×64b
Operating voltage	1V (0.9-1.1V)		1.2V
# of rows turned on simultaneously	9		128
Column multiplexing ratio	32:1		8:1
# of operations per ADC operation	36		128
Precision (bits) activation / weight / output	A:1 / W:ternary / O:4	A:2 / W:3 / O:4	A:1 / W:1 / O:3
Energy-Efficiency (TOPS/W)	53.17	21.9	24.1
Read IMC delay (ns) [a]	10.2	14.6	6.5
Throughput (GOPS) per 128x64 array [b]	7.06	4.94	**157.6 (22.3X higher)**
FoM (Energy-Efficiency × Throughput / 128x64 array) [c]	375.4	108.2	**3,798.2 (10.1X higher)**
CIFAR-10 accuracy	81.83%	88.52%	83.5%

[a] Read IMC delay represents the delay to perform one IMC (along the column) including the ADC operation.
[b] (number of operations) / (read IMC delay) × (number of parallel-computing columns among 64 columns)
[c] FoM effectively represents the inverse of energy-delay product (EDP), a well-known metric that balances energy and performance requirements.

IV. CONCLUSION AND DISCUSSION

We demonstrated an energy-efficient IMC XNOR-RRAM array, which turns on all differential WLs simultaneously and performs analog MAC computation along the bitlines. By monolithically integrating flash ADCs and 90-nm CMOS peripheral circuits with RRAM arrays, we demonstrate the scalability of XNOR-RRAM toward large-scale DNNs. XNOR-RRAM prototype chip measurements and extracted simulations demonstrate a high energy efficiency of 24 TOPS/W, a high throughput of 157.6 GOPS, and a high classification accuracy of 98.5% and 83.5% for the MNIST and CIFAR-10 data sets, respectively. Our work achieves 22.3× and 10.1× improvements in throughput and EDP, respectively, compared with those of the state-of-the-art literature.

ADCs generally incur a large overhead for IMC especially with dense NVMs, as also reported by prior works [34], [35]. Unlike SRAM, the RRAM column pitch is less, which makes the core even more dominated by the peripheral circuits. We used flash ADC where area will exponentially increase with bit precision; therefore, a possible tradeoff is to use more compact successive-approximation-register (SAR) ADC [22], [36], while allowing longer latency, in order to reduce the area overhead.

In our XNOR-RRAM design, further energy-efficiency improvement is largely governed by the LRS resistance of the RRAM technology. Higher energy efficiency could be achieved by RRAM technologies with higher LRS resistance values [24], while this consequently will reduce the ON/OFF ratio. Our current XNOR-RRAM only supports binarized DNNs (both activations and weights have +1 or −1 values),

but multibit precision DNNs that lead to higher accuracy could be supported by bit-serial operation and additional digital peripheral circuits [37] and/or digital-to-analog (DAC) converters [33] while sacrificing energy efficiency. However, the core IMC technology with the proposed 2T2R cell design and peripheral ADC can be applied generally to any given RRAM technology and RRAM arrays.

Acknowledgment

The authors are grateful for chip fabrication support by Winbond Electronics.

References

[1] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778, doi: 10.1109/CVPR.2016.90.

[2] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 4700–4708, doi: 10.1109/CVPR.2017.243.

[3] W. Xiong, L. Wu, F. Alleva, J. Droppo, X. Huang, and A. Stolcke, "The microsoft 2017 conversational speech recognition system," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Apr. 2018, pp. 5934–5938, doi: 10.1109/ICASSP.2018.8461870.

[4] X. Xu *et al.*, "Scaling for edge inference of deep neural networks," *Nature Electron.*, vol. 1, no. 4, pp. 216–222, 2018, doi: 10.1038/s41928-018-0059-3.

[5] D. Silver *et al.*, "Mastering the game of go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, pp. 484–489, Jan. 2016, doi: 10.1038/nature16961.

[6] B. Wu, A. Wan, F. Iandola, P. H. Jin, and K. Keutzer, "SqueezeDet: Unified, small, low power fully convolutional neural networks for real-time object detection for autonomous driving," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jul. 2017, pp. 129–137, doi: 10.1109/CVPRW.2017.60.

[7] S. Yin *et al.*, "A 1.06-μW smart ECG processor in 65-nm CMOS for real-time biometric authentication and personal cardiac monitoring," *IEEE J. Solid-State Circuits*, vol. 54, no. 8, pp. 2316–2326, Aug. 2019, doi: 10.1109/JSSC.2019.2912304.

[8] M. Rastegari, V. Ordonez, J. Redmon, and A. Farhadi, "XNOR-Net: ImageNet classification using binary convolutional neural networks," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2016, pp. 525–542, doi: 10.1007/978-3-319-46493-0_32.

[9] I. Hubara, M. Courbariaux, D. Soudry, R. El-Yaniv, and Y. Bengio, "Binarized neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 4107–4115.

[10] Y.-H. Chen, T. Krishna, J. S. Emer, and V. Sze, "Eyeriss: An energy-efficient reconfigurable accelerator for deep convolutional neural networks," *IEEE J. Solid-State Circuits*, vol. 52, no. 1, pp. 127–138, Jan. 2017, doi: 10.1109/JSSC.2016.2616357.

[11] B. Moons, R. Uytterhoeven, W. Dehaene, and M. Verhelst, "Envision: A 0.26-to-10TOPS/W subword-parallel dynamic-voltage-accuracy-frequency-scalable convolutional neural network processor in 28 nm FDSOI," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2017, pp. 246–247, doi: 10.1109/ISSCC.2017.7870353.

[12] M. T. Bohr and I. A. Young, "CMOS scaling trends and beyond," *IEEE Micro*, vol. 37, no. 6, pp. 20–29, Nov. 2017, doi: 10.1109/MM.2017.4241347.

[13] S.-Y. Wu *et al.*, "A 7 nm CMOS platform technology featuring 4th generation FinFET transistors with a 0.027 μm^2 high density 6-T SRAM cell for mobile SoC applications," in *IEDM Tech. Dig.*, Dec. 2017, pp. 2–6, doi: 10.1109/IEDM.2016.7838333.

[14] S. Yu, "Neuro-inspired computing with emerging nonvolatile memorys," *Proc. IEEE*, vol. 106, no. 2, pp. 260–285, Feb. 2018, doi: 10.1109/JPROC.2018.2790840.

[15] S. Choi *et al.*, "SiGe epitaxial memory for neuromorphic computing with reproducible high performance based on engineered dislocations," *Nature Mater.*, vol. 17, no. 4, pp. 335–340, Apr. 2018, doi: 10.1038/s41563-017-0001-5.

[16] D. Ielmini and H.-S.-P. Wong, "In-memory computing with resistive switching devices," *Nature Electron.*, vol. 1, no. 6, pp. 333–343, Jun. 2018, doi: 10.1038/s41928-018-0092-2.

[17] F. M. Bayat, M. Prezioso, B. Chakrabarti, H. Nili, I. Kataeva, and D. Strukov, "Implementation of multilayer perceptron network with highly uniform passive memristive crossbar circuits," *Nature Commun.*, vol. 9, no. 1, pp. 1–7, Dec. 2018, doi: 10.1038/s41467-018-04482-4.

[18] C. Li *et al.*, "Analogue signal and image processing with large memristor crossbars," *Nature Electron.*, vol. 1, no. 1, pp. 52–59, Jan. 2018, doi: 10.1038/s41928-017-0002-z.

[19] C. Li *et al.*, "Efficient and self-adaptive *in-situ* learning in multilayer memristor neural networks," *Nature Commun.*, vol. 9, no. 1, pp. 1–8, Dec. 2018, doi: 10.1038/s41467-018-04484-2.

[20] B. Li, L. Song, F. Chen, X. Qian, Y. Chen, and H. H. Li, "ReRAM-based accelerator for deep learning," in *Proc. Design, Automat. Test Eur. Conf. Exhib. (DATE)*, Mar. 2018, pp. 815–820, doi: 10.23919/DATE.2018.8342118.

[21] D. Kadetotad, P.-Y. Chen, Y. Cao, S. Yu, and J. Seo, "Peripheral circuit design considerations of neuro-inspired architectures," in *Neuro-Inspired Computing Using Resistive Synaptic Devices*, S. Yu, Ed. Cham, Switzerland: Springer, 2017, pp. 167–182, doi: 10.1007/978-3-319-54313-0_9.

[22] C.-X. Xue *et al.*, "Embedded 1-Mb ReRAM-based computing-in-memory macro with multibit input and weight for CNN-based AI edge processors," *IEEE J. Solid-State Circuits*, vol. 55, no. 1, pp. 203–215, Jan. 2020, doi: 10.1109/JSSC.2019.2951363.

[23] 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," in *Proc. Design, Automat. Test Eur. Conf. Exhib. (DATE)*, Mar. 2018, pp. 1423–1428, doi: 10.23919/DATE.2018.8342235.

[24] S. Yin *et al.*, "Monolithically integrated RRAM- and CMOS-based in-memory computing optimizations for efficient deep learning," *IEEE Micro*, vol. 39, no. 6, pp. 54–63, Nov. 2019, doi: 10.1109/MM.2019.2943047.

[25] P.-Y. Chen, X. Peng, and S. Yu, "NeuroSim: A circuit-level macro model for benchmarking neuro-inspired architectures in online learning," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 37, no. 12, pp. 3067–3080, Dec. 2018, doi: 10.1109/TCAD.2018.2789723.

[26] A. Krizhevsky, "Learning multiple layers of features from tiny images," M.S. thesis, Univ. Toronto, Toronto, ON, Canada, 2009, doi: 10.1.1.222.9220.

[27] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998, doi: 10.1109/5.726791.

[28] B. Mohammad, P. Dadabhoy, K. Lin, and P. Bassett, "Comparative study of current mode and voltage mode sense amplifier used for 28 nm SRAM," in *Proc. 24th Int. Conf. Microelectron. (ICM)*, Dec. 2012, pp. 1–6, doi: 10.1109/ICM.2012.6471396.

[29] C. Ho *et al.*, "Integrated HfO_2-RRAM to achieve highly reliable, greener, faster, cost-effective, and scaled devices," in *IEDM Tech. Dig.*, Dec. 2017, pp. 2–6, doi: 10.1109/IEDM.2017.8268314.

[30] Y. S. Chen *et al.*, "Highly scalable hafnium oxide memory with improvements of resistive distribution and read disturb immunity," in *IEDM Tech. Dig.*, Dec. 2009, pp. 1–4, doi: 10.1109/IEDM.2009.5424411.

[31] Z. Jiang, S. Yin, M. Seok, and J.-S. Seo, "XNOR-SRAM: In-memory computing SRAM macro for binary/ternary deep neural networks," in *Proc. IEEE Symp. VLSI Technol.*, Jun. 2018, pp. 173–174, doi: 10.1109/VLSIT.2018.8510687.

[32] C.-Y. Chen, M. Q. Le, and K. Y. Kim, "A low power 6-bit flash ADC with reference voltage and common-mode calibration," *IEEE J. Solid-State Circuits*, vol. 44, no. 4, pp. 1041–1046, Apr. 2009, doi: 10.1109/JSSC.2009.2014701.

[33] P. Chi *et al.*, "PRIME: A novel processing-in-memory architecture for neural network computation in ReRAM-based main memory," in *Proc. ACM/IEEE 43rd Annu. Int. Symp. Comput. Archit. (ISCA)*, Jun. 2016, pp. 27–39, doi: 10.1109/ISCA.2016.13.

[34] T. Gokmen and Y. Vlasov, "Acceleration of deep neural network training with resistive cross-point devices: Design considerations," *Frontiers Neurosci.*, vol. 10, p. 333, Jul. 2016, doi: 10.3389/fnins.2016.00333.

[35] A. Shafiee *et al.*, "ISAAC: A convolutional neural network accelerator with *in-situ* analog arithmetic in crossbars," in *Proc. ACM/IEEE 43rd Annu. Int. Symp. Comput. Archit. (ISCA)*, Jun. 2016, pp. 14–26, doi: 10.1109/ISCA.2016.12.

[36] Q. Wang, Y. Kim, and P. Li, "Neuromorphic processors with memristive synapses: Synaptic interface and architectural exploration," *ACM J. Emerg. Technol. Comput. Syst.*, vol. 12, no. 4, pp. 1–22, Jul. 2016, doi: 10.1145/2894756.

[37] J. Wang *et al.*, "A compute SRAM with bit-serial integer/floating-point operations for programmable in-memory vector acceleration," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2019, pp. 224–226, doi: 10.1109/ISSCC.2019.8662419.