An Always-On 3.8 μJ/86% CIFAR-10 Mixed-Signal Binary CNN Processor With All Memory on Chip in 28-nm CMOS

Daniel Bankman, *Student Member, IEEE*, Lita Yang, *Student Member, IEEE*, Bert Moons, *Student Member, IEEE*, Marian Verhelst, *Senior Member, IEEE*, and Boris Murmann, *Fellow, IEEE*

***Abstract*— The trend of pushing inference from cloud to edge due to concerns of latency, bandwidth, and privacy has created demand for energy-efficient neural network hardware. This paper presents a mixed-signal binary convolutional neural network (CNN) processor for always-on inference applications that achieves 3.8 μJ/classification at 86% accuracy on the CIFAR-10 image classification data set. The goal of this paper is to establish the minimum-energy point for the representative CIFAR-10 inference task, using the available design tradeoffs. The BinaryNet algorithm for training neural networks with weights and activations constrained to +1 and −1 drastically simplifies multiplications to XNOR and allows integrating all memory on-chip. A weight-stationary, data-parallel architecture with input reuse amortizes memory access across many computations, leaving wide vector summation as the remaining energy bottleneck. This design features an energy-efficient switched-capacitor (SC) neuron that addresses this challenge, employing a 1024-bit thermometer-coded capacitive digital-to-analog converter (CDAC) section for summing pointwise products of CNN filter weights and activations and a 9-bit binary-weighted section for adding the filter bias. The design occupies 6 mm^2 in 28-nm CMOS, contains 328 kB of on-chip SRAM, operates at 237 frames/s (FPS), and consumes 0.9 mW from 0.6 V/0.8 V supplies. The corresponding energy per classification (3.8 μJ) amounts to a 40× improvement over the previous low-energy benchmark on CIFAR-10, achieved in part by sacrificing some programmability. The SC neuron array is 12.9× more energy efficient than a synthesized digital implementation, which amounts to a 4× advantage in system-level energy per classification.**

***Index Terms*— Binarized neural networks, deep learning, mixed-signal processing, near-memory computing, switched-capacitor (SC).**

Manuscript received April 30, 2018; revised July 10, 2018 and August 15, 2018; accepted August 21, 2018. Date of publication October 3, 2018; date of current version January 14, 2019. This paper was approved by Guest Editor Masato Motomura. This work was supported by MARCO and DARPA through Systems on Nanoscale Information fabriCs (SONIC)—one of the six STARnet Centers. *(Corresponding author: Daniel Bankman.)*

D. Bankman, L. Yang, and B. Murmann are with the Department of Electrical Engineering, Stanford University, Stanford, CA 94305 USA (e-mail: dbankman@stanford.edu).

B. Moons and M. Verhelst are with the Department of Electrical Engineering, ESAT–MICAS, KU Leuven, 3001 Leuven, Belgium.

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

Digital Object Identifier 10.1109/JSSC.2018.2869150

I. Introduction

RECENT advancements in machine learning algorithms, hardware, and data sets have made it possible to train deep neural networks (DNNs) with many hidden layers [1], [2]. DNNs have achieved state-of-the-art performance on tasks such as visual recognition and speech recognition, and have been deployed in cloud-based services such as language translation and photo search [3], [4]. Due to concerns of latency, network bandwidth, and privacy, there now exists a trend to push DNNs from cloud to edge, for inference applications such as keyword spotting, face detection, and image classification [5]–[7].

The key building block of DNNs is the multiply–accumulate (MAC) operation. A significant challenge arises from the fact that DNNs perform millions to billions of MAC operations per inference. For this reason, commercial GPUs have become the standard hardware for deep learning. An embedded GPU system-on-module has been developed for inference at the edge [8], but consumes several watts of power, making it difficult to deploy in always-on applications. Given that the energy cost of memory access is considerably higher than that of MAC operations, prior work has demonstrated dedicated DNN accelerators that aim to minimize data movement [9], [10]. The Eyeriss architecture maps CNN computations onto a spatial array in a manner that minimizes access to energy-hungry off-chip DRAM, but does not eliminate it completely due to the high complexity of the ImageNet task [11]. With energy per classification at the millijoule level, edge deployment remains a challenge.

This design targets an inference task of moderate complexity, the CIFAR-10 image classification data set [12]. A strong tradeoff exists between energy and programmability, and between energy and accuracy. The design objective of this work is to minimize energy while permitting the sacrifice of some programmability and accuracy. Nonetheless, this design remains semi-programmable and achieves accuracy on CIFAR-10 competitive with other recent hardware implementations. To our knowledge, this design achieves the lowest energy per classification on CIFAR-10 reported to date.

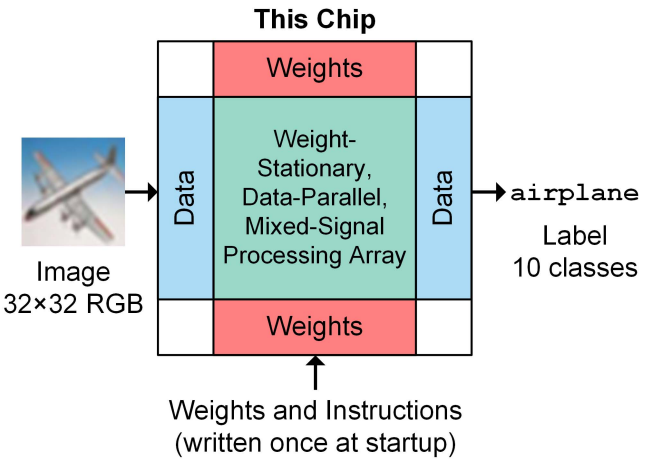

Fig. 1. Chip overview. Our design contains a weight-stationary, data-parallel, mixed-signal processing array as well as the surrounding memory required to time multiplex the array for processing deep CNNs.

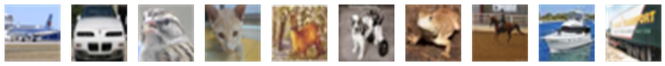

Fig. 2. The CIFAR-10 data set contains 10 categories of 32×32 RBG images: airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck.

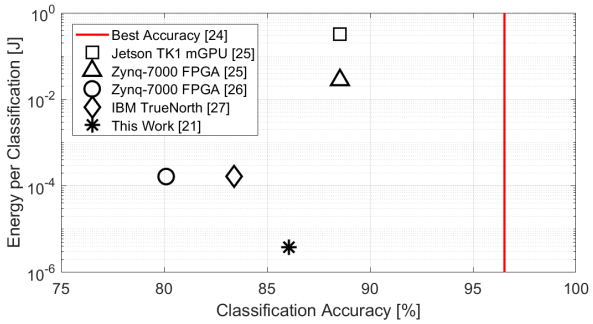

Fig. 3. Energy-accuracy tradeoff in CNN hardware running CIFAR-10 image classification.

For the CIFAR-10 problem size, algorithmic and architectural solutions are readily applied to overcome the memory energy bottleneck [13], [14]. The BinaryNet algorithm for training neural networks with weights and activations constrained to $+1$ and -1 drastically simplifies multiplications to XNOR and allows integrating all memory on-chip [13]. Inspired by [14], we demonstrate a weight-stationary, data-parallel architecture with input reuse that amortizes memory access across many computations. We address BinaryNet's remaining challenge of wide vector summation using mixed-signal processing [15]. The employed switched-capacitor (SC) neuron improves energy efficiency over the equivalent digital adder tree while preserving system-level classification accuracy. Our proof-of-concept mixed-signal binary CNN processor (depicted in Fig. 1) performs a complete image classification on the chip.

The computing in memory (CIM) approach has recently been explored as a means of overcoming the memory energy bottleneck. Zhang *et al.* [16] describe how the conventional von Neumann and CIM architectures scale in bandwidth and energy as a function of the total amount of the data required for a computation, in order to illustrate the advantages of the CIM approach. These advantages apply to the general class of weight-stationary, data-parallel processing arrays with each cell containing both memory and compute elements. This class of processing arrays subsumes the CIM designs in [16]– [19], as well as the SC neuron array with weight local memory latches described in this paper. However, the recent CIM designs do not realize a complete DNN on chip, nor do they address the issue of dataflow into and out of the processing array (with the exception of [19], which cascades two arrays to process two DNN layers). Restricting the circuit implementation of the processing array to high-density RAM results in poor matching between the unit elements involved in analog addition along a column (e.g., SRAM cells). The CIM design of [16] achieves sub-nanojoule energy per classification, but like the rest of the aforementioned CIM designs, its functionality has only been demonstrated on the low-complexity MNIST handwritten digit recognition task [20]. In contrast, the mixed-signal binary CNN processor presented in this paper contains the surrounding memory required to time multiplex the SC neuron array for processing deep CNNs, thereby realizing a complete neural network with image-to-label functionality on chip.

The remainder of this paper expands on our conference contribution [21] and is organized as follows. Section II describes the co-design of the binary CNN topology and the chip architecture. Section III presents the circuit implementation of the SC neuron, and the behavioral Monte Carlo simulation technique used to predict CNN classification accuracy in the presence of circuit nonidealities. Section IV presents measurement results. Section V compares the mixed-signal binary CNN processor of this work to the synthesized digital implementation presented in [22], which uses the same baseline architecture but adds a degree of flexibility to the CNN topologies that can be processed. In addition, the SC neuron is compared to a hand-designed digital neuron in terms of simulation results. Section VI concludes the paper.

II. System Architecture

This work uses the CIFAR-10 image classification data set as a driving example of an inference task for always-on edge devices. Fig. 2 shows examples of the 10 categories of 32×32 RGB images, illustrating that due to the small image size, the category can be difficult to determine. Human accuracy on the CIFAR-10 data set is 94% [23]. The best accuracy reported on CIFAR-10 at this time is 96.53%, achieved using a deep CNN with floating-point precision [24].

The original BinaryNet topology for CIFAR-10 image classification achieved 88.60% accuracy, 7.93% lower than the best reported [13], [24]. However, Zhao *et al.* [25] demonstrated that the field-programmable gate array (FPGA) implementation of the original BinaryNet translates this accuracy loss to an energy efficiency gain with respect to an embedded GPU, reducing energy per classification from hundreds to tens of millijoules. Fig. 3 shows several energy accuracy

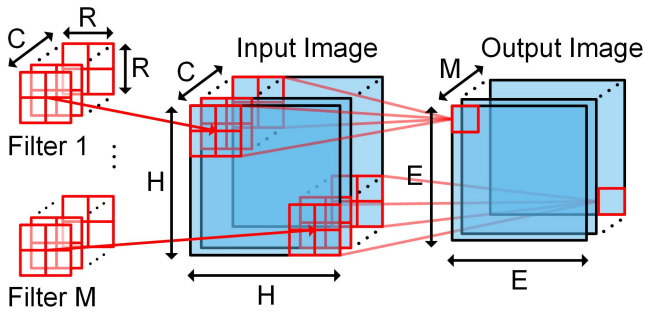

Fig. 4. General multi-filter, multi-channel convolution. Filter size R, number of channels C, number of filters M, input image size H, output image size E.

points achieved by GPU, FPGA, and custom hardware running CIFAR-10 image classification [21], [26], [27].

In this work, we make several modifications to the original BinaryNet topology in order to simplify the logic and interconnect at the interface between memory and compute, resulting in a binary CNN topology which we call *CMOS-inspired*. This topology achieves 86.05% accuracy on CIFAR-10 (2.55% lower than the original [13]), but enables energy per classification of 3.8 μJ in the mixed-signal binary CNN processor. This energy-accuracy point resides on the Pareto-optimal frontier for CIFAR-10 (Fig. 3), achieving 40× lower energy than the previous state of the art [27]. This is accomplished through hardware specialization, which sacrifices some programmability but still provides a practical CNN topology that can be trained for other lightweight applications. These applications include audio keyword spotting using spectrograms as input, and more generally always-on wake-up detection that serves to turn on more accurate and/or more programmable inference hardware when necessary.

A. Binary CNN Topology

In general, each layer of a CNN performs a multi-filter, multi-channel convolution, depicted in Fig. 4. Consider the case of a single filter ($M = 1$). Each channel of the filter is convolved with the corresponding channel of the input image. Channelwise results are then summed to form a single-channel output image. The filter also has a bias term that is added to the output. With multiple filters ($M > 1$), each filter is applied independently to the input image and produces one channel of the output image. Hence, the output image has M channels. One channel of one pixel of a hidden layer input image is the output of one neuron in the previous layer, also called an activation.

The *CMOS-inspired* binary CNN topology is described in Table I. It uses only valid convolutions (no zero padding). Each convolution is followed immediately by batch normalization, which scales pixel intensities to have zero mean and unit variance [28]. After training, batch normalization becomes an affine transformation. Each filter has batch normalization parameters in addition to its weights and bias. Batch normalization helps to train neural networks with saturating activation functions (such as the sgn function) [13], [28]. After batch normalization, output image pixels are binarized. Max-pooling

TABLE I
BINARY CNN WITH REGULAR STRUCTURE

Layer	Type	H	E	R	C	M	Stride
1	convolution	32	31	2	256	256	1
2	convolution	31	30	2	256	256	1
3	convolution	30	29	2	256	256	1
4	convolution	29	28	2	256	256	1
4p	max pooling	28	14	2	-	-	2
5	convolution	14	13	2	256	256	1
6	convolution	13	12	2	256	256	1
6p	max pooling	12	6	2	-	-	2
7	convolution	6	5	2	256	256	1
8	convolution	5	4	2	256	256	1
9	fully-connected	4096 - to - 10					

layers downsample each channel of an image independently, by discarding all but the activation with maximum intensity in a set of non-overlapping 2×2 patches [1].

The *CMOS-inspired* binary CNN differs from the original BinaryNet topology as follows:

1) Structural Regularity: By enforcing structural regularity on the CNN topology, we ensure high utilization of each path through the interface logic between memory and compute, thereby making efficient use of the dynamic energy dissipated in the physical design. Each CNN layer uses strictly 2×2 filters ($R = 2$), 256 channels ($C = 256$), and 256 filters ($M = 256$). The circuit-level implications of this structural regularity are short wires and arrayed, low fan-out de-multiplexers at the interface between memory and compute.

2) No Hidden Fully Connected Layers: Fully connected (FC) layers do not permit weight reuse, hence they can dominate the memory requirements of a CNN. The original BinaryNet topology required 1.67 MB of weight memory, with 558 kB needed for six CNN layers and 1.13 MB needed for three FC layers. The binary CNN of this work uses only one FC layer, which outputs the category label. It requires 261.5 kB of weight memory: 256 kB for eight CNN layers and 5.5 kB for one FC layer.

3) Max-Pooling After Binarization: The original BinaryNet topology placed max-pooling layers immediately following convolutions, before batch normalization and binarization. This required max-pooling to operate over floating point values (due to the filter bias, which has floating point precision in the original BinaryNet [13]). Alternatively, the binary CNN of this work places max-pooling layers after binarization, such that max-pooling operates over binary activations. In this manner, only binary activations need to be manipulated and stored outside the processing element that computes them. With binary inputs, the 2×2 max-pooling operation reduces to a four-way logical OR [26].

In summary, the *CMOS-inspired* topology follows the goal of minimizing energy while permitting the sacrifice of some programmability and accuracy. Restricting the filter size to 2×2 and the number of filters and channels to 256 minimizes the path loading between memory and compute. Using only a single FC layer at the network output reduces the required

weight memory capacity by 5.1×. The *CMOS-inspired* topology achieves 86.05% accuracy on CIFAR-10 (2.55% lower than the original [13]), but enables energy per classification of 3.8 μJ, 40× lower than the previous low-energy benchmark (Fig. 3).

B. Filter Computation and Dedicated Hardware Neuron

One filter computation includes the dot product between an $R \times R \times C$ image patch x and an $R \times R \times C$ filter w, the addition of the filter bias b, and the affine transformation required for batch normalization parameterized by μ, σ, γ, β, and ϵ. The sgn function is applied to the result, producing the activation z

$$z = \text{sgn}\left(\left(\left(\sum_{i=0}^{N-1} w_i x_i + b - \mu\right)\frac{\gamma}{\sqrt{\sigma^2+\epsilon}} + \beta\right)\right). \quad (1)$$

In this expression, w_i, x_i, and z take on values $+1$ or -1 as in [13], $N = R \times R \times C = 1024$ is the number of pointwise weight-activation products, and the remaining terms are floating point. Taking advantage of the fact that the argument of the sgn function can be scaled by a positive number without affecting its output, we absorb the batch normalization parameters into b and the sign of γ into each w_i [25], [26]. The filter computation is simplified to

$$z = \text{sgn}\left(\sum_{i=0}^{N-1} w_i x_i + b\right). \quad (2)$$

The filter bias b is then converted from floating point to a 9-bit sign-magnitude integer. Because the sgn function is itself a quantizer, only the clipping of b introduces error.

Each filter computation is completed inside a dedicated hardware neuron. In this manner, partial results do not need to be written to or loaded from memory. The neuron contains XNOR gates that perform pointwise multiplications between the $2 \times 2 \times 256$ filter and image patch. In the mixed-signal binary CNN processor, addition is implemented using charge redistribution. The summation over 1024 pointwise products and bias is completed in the analog domain. Only the polarity of the result, the 1-bit neuron activation, needs to be resolved by a voltage comparator.

C. Top-Level Architecture

Fig. 5(a) shows the top-level block diagram of the mixed-signal binary CNN processor. The architecture was inspired by [14], which showed that data parallelism and parameter reuse together result in a significant reduction in the number of memory accesses required to perform a multi-filter, multi-channel convolution. The input to the chip is a 32×32 RGB image, and the output is a 4-bit category label. The weight-stationary, data-parallel neuron array performs filter computations for up to eight CNN layers. In the final stage of the neural network, the category label is computed by an FC layer that performs MAC operations digitally. Up to 10 FC layer neurons can be accommodated, each using XNOR multiplication followed by a 13-bit wide sequential accumulator. This provides sufficient precision to determine the index of the FC layer neuron with the maximum activation, which represents the category label. The processor features a customized instruction set for CNN layers, max-pooling layers, FC layers, and input–output actions. Instructions indicating the layer types, sizes, and ordering are programmed at startup

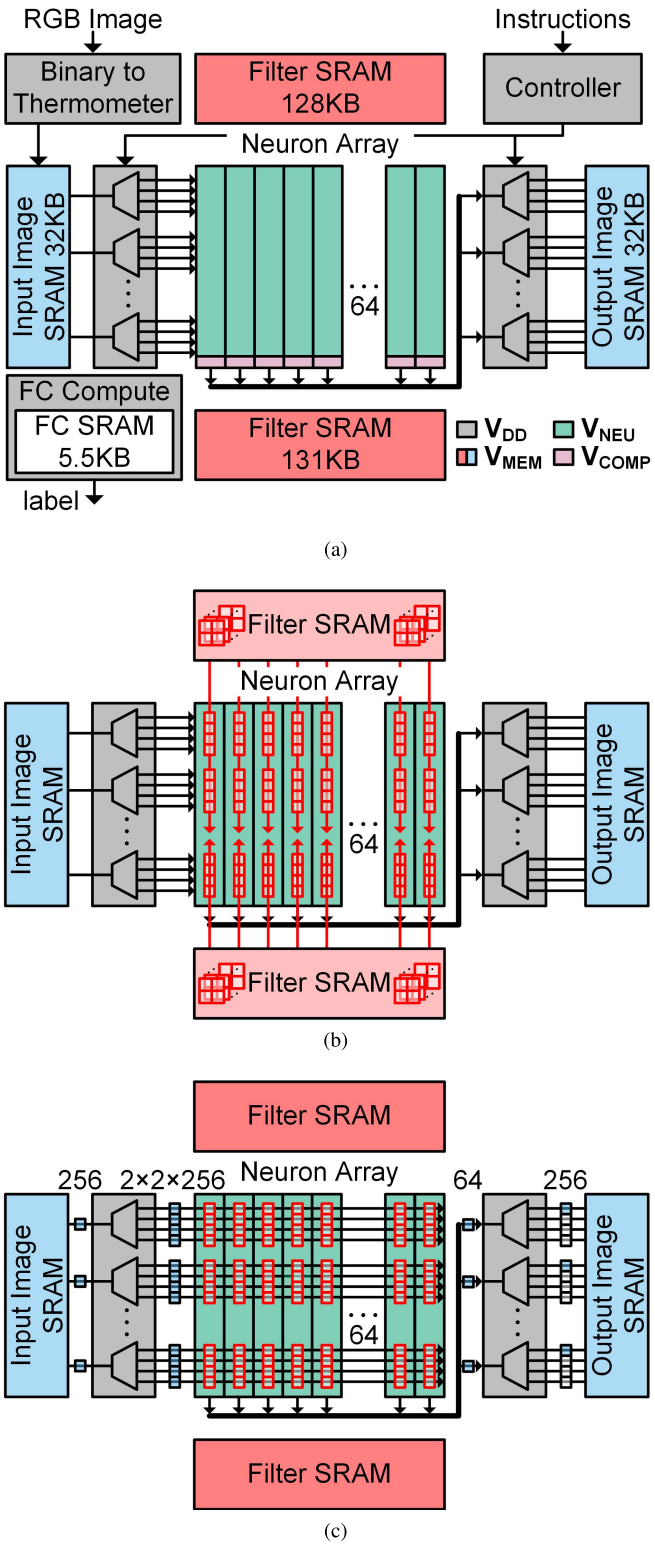

Fig. 5. (a) Top-level block diagram. (b) Weights are transferred from filter SRAM to neuron array weight local memory latches. (c) 64 neurons process the same image patch simultaneously.

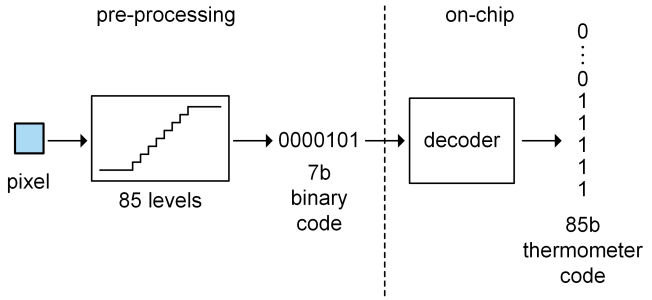

Fig. 6. Each color channel of each pixel is quantized to 85 levels, read into the chip as a 7-bit binary code, and subsequently decoded into an 85-bit thermometer code.

over a Serial Peripheral Interface. The number of CNN layers is programmable, up to a maximum of eight. Voltage supplies for digital logic and memory, V_{DD} and V_{MEM}, are separated to allow independent voltage scaling. A 0.6-V combined digital supply and analog reference V_{NEU} powers the SC neuron array, and a 0.8-V supply V_{COMP} powers the voltage comparators resolving neuron outputs.

Each color channel of each pixel of the 32×32 RGB image is quantized to 85 levels and read into the chip as a 7-bit binary code. On chip, these binary codes are converted to 85-bit thermometer codes (Fig. 6) and stacked (with 1 bit of zero padding) to form a 256-channel pixel. The resulting $32 \times 32 \times 256$ input image is directly compatible with the binary CNN, allowing the input layer to utilize the same dedicated hardware neuron (or in software, the same XOR, NOT, and popcount instructions) as the rest of the network. This is a solution to the first layer issue pointed out in [13]. Alternatively, a separate first layer input image SRAM sized at 2.6 kB could store the 32×32 RGB image with 7 bits per color channel, which would consume less energy to write and read. However, the overall savings in energy per classification would be limited by the fact that the first layer contains only 25% of the filter computations in the network in Table I. Hence, we opt for the simplicity of hardware reuse and avoid the datapath and control overhead of an additional memory block.

Filter SRAM banks are located on the north and south sides of the neuron array and are written once at startup. With strictly $2 \times 2 \times 256$ filters, 256 kB of filter SRAM is required to store weights for eight CNN layers. The south filter SRAM bank includes an additional 3 kB for storing filter biases. Weights for the FC layer are stored in a separate 5.5-kB SRAM bank. Image SRAM banks are located on the east and west sides of the neuron array, and have the capacity to store the input and output images for one CNN layer at a time. Image SRAM banks alternate roles between input and output from layer to layer in a ping-pong fashion. Each image SRAM bank is sized at 32 kB to accommodate the largest possible CNN layer input—the $32 \times 32 \times 256$ thermometer-coded RGB image.

D. Convolution Dataflow

Before a convolution begins, filter weights are transferred from filter SRAM into neuron array weight local memory latches, as shown in Fig. 5(b). Weights are stationary throughout the duration of a convolution, which amortizes the energy cost of reading them from SRAM across all filter computations until the next convolution begins. Image SRAM banks are 256 bits wide, with one word representing one 256-channel pixel, as shown in Fig. 5(c). The input DEMUX block interfaces between image SRAM, which loads a 256-channel pixel, and the neuron array, which receives a $2 \times 2 \times 256$ patch. The data-parallel array of 64 neurons processes the same image patch simultaneously, amortizing the energy cost of reading the patch across 64 filter computations. In order to process 256 filters per layer with 64 hardware neurons, the neuron array must be four-way time-multiplexed. This is accomplished by splitting the 256 filters into four filter groups, and repeating convolution over the input image for each filter group. The output DEMUX block manages writing 64 channels at a time to a 256-channel pixel in output image SRAM.

E. Datapath Microarchitecture

Fig. 7 shows how sliced processing translates to reduced path loading. The input DEMUX block is implemented as an array of 256, 1-to-4 de-multiplexers with output registers. Each pixel of the input image can be reused in the processing of two overlapping patches, amortizing the input image SRAM read energy per filter computation by $2\times$. A 2×2 crossbar interchanges pixel pairs at the neuron array input as required by sliding window convolution. Filter weights are transferred over a 4 bit per neuron bus, split into north and south halves to reduce loading of weight transfers by $2\times$. To minimize logic and interconnect at the neuron array to memory interface, each neuron processes the same four filters and writes to the same four output channels in each CNN layer. For example, neuron 0 always processes filters 0 to 3, and neuron 63 always processes filters 252 to 255. The output DEMUX block is implemented as an array of 1-to-4 de-multiplexers, which minimizes path loading between the neuron array and output image SRAM. Max pooling occurs incrementally during convolution by first reading a 256-channel pixel from output image SRAM, and then writing back the logical OR of the 64 channels corresponding to the current filter group with the 64-channel neuron array output. Each path through the de-multiplexers is utilized in each CNN layer. In this manner, the *CMOS-inspired* binary CNN topology with a fixed filter size makes efficient use of the dynamic energy consumed in the physical design.

This design uses strictly 2×2 filters. However, a network with filter size fixed at 1×1 or 3×3 can utilize a similar datapath microarchitecture. With 1×1 filters, the filter SRAM capacity decreases to 64 kB, the input DEMUX block is no longer necessary, and only one input image SRAM read is required per filter computation. With 3×3 filters, the filter SRAM capacity increases to 576 kB, the input DEMUX block becomes an array of 256, 1-to-9 de-multiplexers, and three input image SRAM reads are required per filter computation.

III. Switched-Capacitor Neuron Array

BinaryNet makes multiplications trivial by reducing them to XNOR, but addition remains a challenge due to the large

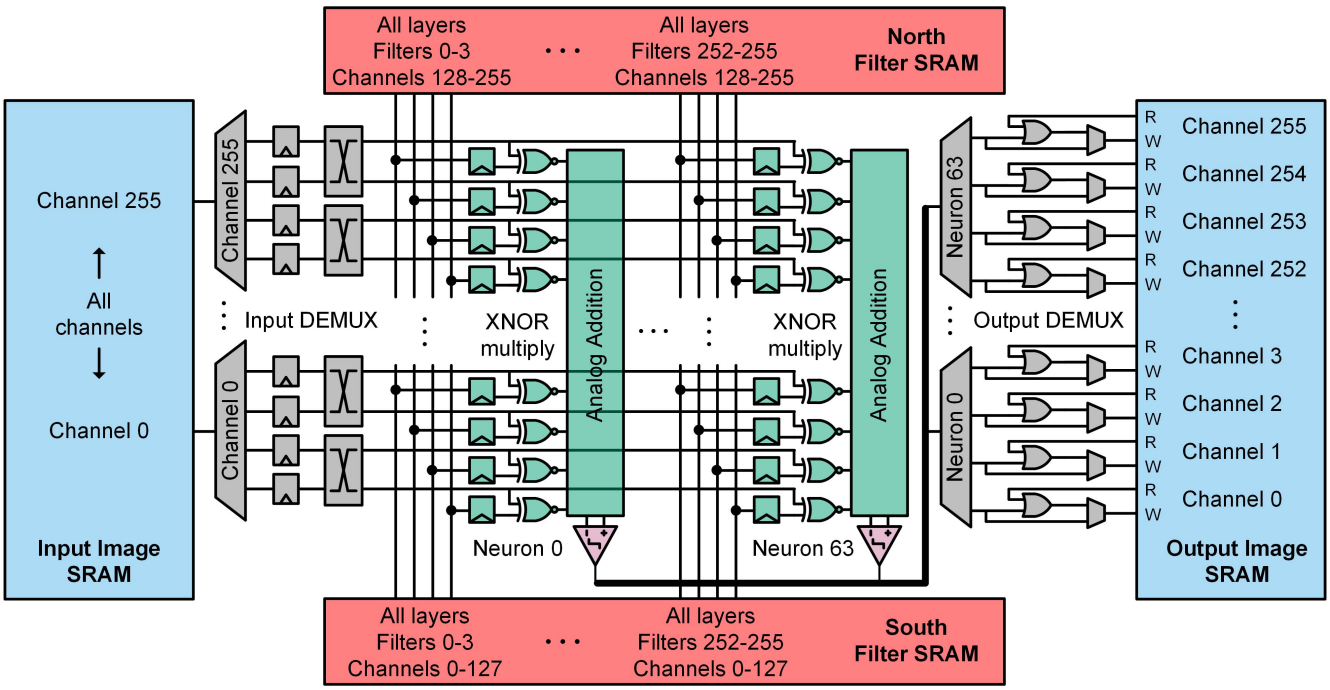

Fig. 7. Microarchitecture of neuron array datapath.

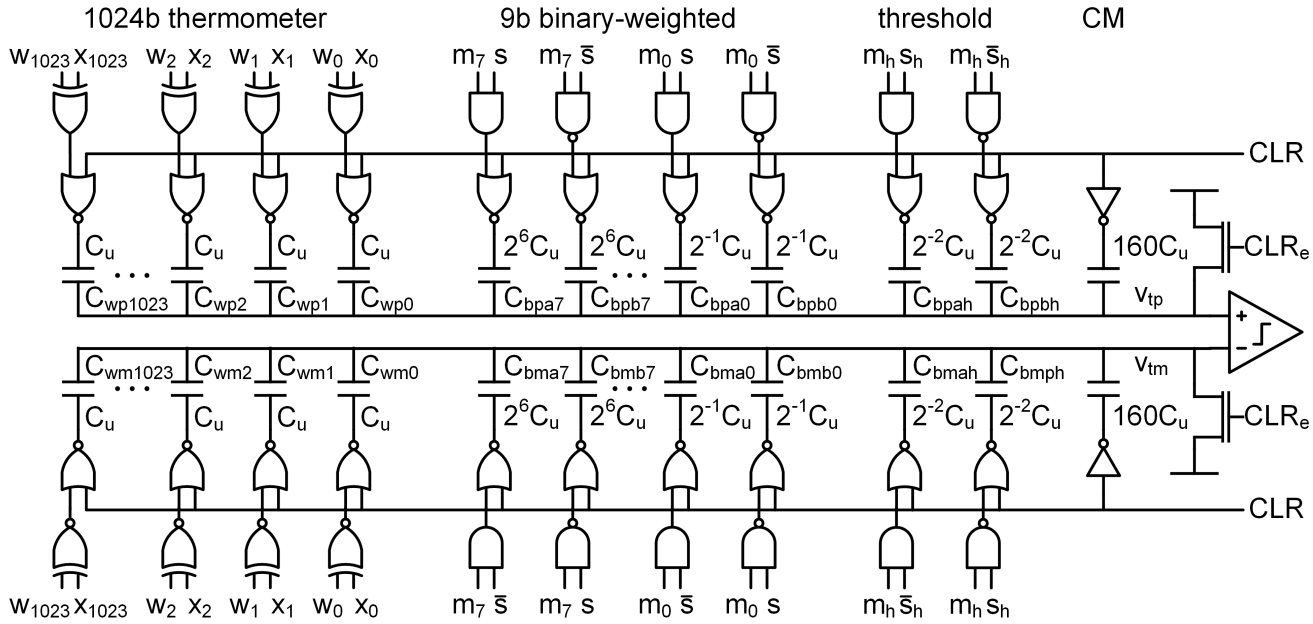

Fig. 8. SC neuron performing wide vector summation via charge redistribution. The 1-bit neuron output is resolved with a single comparator decision.

number of inputs per neuron. The dedicated hardware neuron of this design must compute its output z according to the expression in (2). Because the binary CNN can tolerate a certain number of erroneous neuron outputs without misclassifying the input image, an opportunity exists to improve energy efficiency by computing z with an internally analog, externally digital SC neuron.

A. Ideal Neuron Operation

Fig. 8 shows the SC neuron schematic, which computes z exactly according to (2) under ideal operation. The filter bias b is represented as a 9-bit sign-magnitude integer as follows:

$$b = (-1)^{s} \sum_{i=0}^{B-2} 2^{i} m_i \tag{3}$$

where the bias word length $B = 9$, s is the sign bit, and m_i are the magnitude bits. The SC neuron uses a differential CDAC with four sections: a 1024-bit thermometer section for summing pointwise products of filter weights and activations, a 9-bit binary-weighted section for adding the filter bias, a threshold section, and a common-mode (CM) setting section.

The CDAC unit element is a metal-oxide-metal (MOM) fringe capacitor. In the SC neuron layout, each unit capacitor pair sits on a standard cell row with its drivers. Due to the number of wire tracks per standard cell height, one extra MOM fringe capacitor finger is left unused for every two unit capacitors. These fingers are re-purposed as the CM section and switched to V_{DD} to slightly lift the comparator input common-mode voltage. The binary-weighted section is symmetrically switched to maintain a constant common-mode voltage into the comparator [29]. A dynamic double-tail comparator resolves the 1-bit neuron output [30].

Before convolution begins, all capacitors are discharged by asserting and de-asserting the signals CLR and CLR_e. To prevent drawing excessive charge from the supply, the bottom plate nodes are discharged by asserting CLR before the top plate is discharged via CLR_e. To prevent asymmetric charge injection, the top plate switches are turned off before the bottom plate voltages resume their values set by the neuron inputs. The frequency of CLR operations is dictated by leakage at the top plate node. During convolution, a CLR operation is executed every 100 to 300 cycles, resulting in a negligible contribution to energy per classification.

With zero charge on the top plate nodes, the differential voltage $v_{\mathrm{td}} = v_{\mathrm{tp}} - v_{\mathrm{tm}}$ can be expressed as a superposition of capacitive dividers

$$\begin{aligned}\frac{v_{\mathrm{td}}}{V_{\mathrm{DD}}} = &\sum_{i=0}^{N-1}\left(\frac{C_{wp,i}}{C_{p,\mathrm{tot}}}\left[\mathrm{w}_i \oplus \mathrm{x}_i\right] - \frac{C_{wm,i}}{C_{m,\mathrm{tot}}}\left[\overline{\mathrm{w}_i \oplus \mathrm{x}_i}\right]\right)\\ &+ \sum_{i=0}^{B-2}\left(\frac{C_{bpa,i}}{C_{p,\mathrm{tot}}}\left[\overline{\mathrm{m}_i \mathrm{s}}\right] + \frac{C_{bpb,i}}{C_{p,\mathrm{tot}}}\left[\mathrm{m}_i \overline{\mathrm{s}}\right]\right)\\ &- \sum_{i=0}^{B-2}\left(\frac{C_{bma,i}}{C_{m,\mathrm{tot}}}\left[\overline{\mathrm{m}_i \overline{\mathrm{s}}}\right] + \frac{C_{bmb,i}}{C_{m,\mathrm{tot}}}\left[\mathrm{m}_i \mathrm{s}\right]\right). \end{aligned} \tag{4}$$

In this expression, the non-italicized w_i and x_i take on values 0 and 1 (corresponding to -1 and $+1$ in (2)). The Boolean expressions in brackets are treated as 0 or 1 integers multiplying the capacitive ratios, and $C_{p,\mathrm{tot}}$ and $C_{m,\mathrm{tot}}$ denote the total capacitance in the positive and negative differential halves. We have omitted the threshold section (assumed $\mathrm{m}_h = 0$) for brevity. In the ideal case, the capacitors take exactly the values labeled in Fig. 8, and the expression in (4) reduces to

$$\frac{v_{\mathrm{td}}}{V_{\mathrm{DD}}} = \frac{C_u}{C_{\mathrm{tot}}}\left(\sum_{i=0}^{N-1} w_i x_i + b\right). \tag{5}$$

We define the CDAC step size and full-scale range as follows

$$v_{\mathrm{LSB}} = \frac{C_u}{C_{\mathrm{tot}}} V_{\mathrm{DD}} \tag{6}$$

$$V_{\mathrm{FS}} = \frac{(N + 2^{B-1} - 1)C_u}{C_{\mathrm{tot}}} V_{\mathrm{DD}}. \tag{7}$$

An ideal comparator senses the polarity of v_{td}, which is exactly equal to z in (2). The purpose of the threshold section is to subtract $1/2$ from the integer term inside the parenthesis in (5), which defines how the sgn function treats an identically zero input. This allows simulation of the ideal SC neuron to produce ideal results, but becomes irrelevant in the presence of comparator noise and offset.

B. Impact of Circuit Nonidealities

Three circuit nonidealities in the SC neuron can affect the classification accuracy of the binary CNN. These are digital-to-analog converter (DAC) unit capacitor mismatch, comparator offset, and comparator noise. In the presence of unit capacitor mismatch, the capacitive ratios in (4) do not reduce to a perfect factor of C_u/C_{tot} as in (5). Due to unit capacitor mismatch and comparator offset, the SC neuron effectively applies a different filter, with weights w_{eff} and bias b_{eff} determined by the actual capacitor values and comparator offset voltage. Unlike unit capacitor mismatch and comparator offset that are random but static, comparator noise v_n randomizes every decision independently. The differential voltage can be expressed as

$$\frac{v_{\mathrm{td}}}{V_{\mathrm{DD}}} = \sum_{i=0}^{N-1} w_{\mathrm{eff},i} x_i + b_{\mathrm{eff}} + \frac{v_n}{V_{\mathrm{DD}}} \tag{8}$$

where the actual capacitor values and comparator offset determine w_{eff} and b_{eff}

$$w_{\mathrm{eff},i} = \left(\frac{C_{wp,i}}{2C_{p,\mathrm{tot}}} + \frac{C_{wm,i}}{2C_{m,\mathrm{tot}}}\right) w_i \tag{9}$$

$$\begin{aligned} b_{\mathrm{eff}} = &\sum_{i=0}^{N-1}\left(\frac{C_{wp,i}}{2C_{p,\mathrm{tot}}} - \frac{C_{wm,i}}{2C_{m,\mathrm{tot}}}\right)\\ &+ \sum_{i=0}^{B-2}\left(\frac{C_{bpa,i}}{C_{p,\mathrm{tot}}}\left[\overline{\mathrm{m}_i \mathrm{s}}\right] + \frac{C_{bpb,i}}{C_{p,\mathrm{tot}}}\left[\mathrm{m}_i \overline{\mathrm{s}}\right]\right)\\ &- \sum_{i=0}^{B-2}\left(\frac{C_{bma,i}}{C_{m,\mathrm{tot}}}\left[\overline{\mathrm{m}_i \overline{\mathrm{s}}}\right] + \frac{C_{bmb,i}}{C_{m,\mathrm{tot}}}\left[\mathrm{m}_i \mathrm{s}\right]\right)\\ &+ \frac{v_{\mathrm{os}}}{V_{\mathrm{DD}}}. \end{aligned} \tag{10}$$

C. Offset Calibration at Startup

With 9 bits allocated to the filter bias, sufficient range exists for offset correction using the binary-weighted DAC section, without the introduction of an additional trim-DAC. The comparator offset v_{os} is digitized at startup, stored in a local register, and subtracted from the bias loaded from filter SRAM during weight transfer using saturating arithmetic [31]. Behavioral Monte Carlo simulations show that this greatly relaxes the amount of comparator offset that can be tolerated without degrading system-level classification accuracy. With calibration, the tolerable offset is limited by the resulting clipping of the filter bias. In environments where large temperature changes may induce significant offset drift, calibration can be performed periodically (e.g., once per second) at negligible cost in average energy per classification and throughput.

D. Behavioral Monte Carlo Simulation

Based on the error model of (8)–(10), we now present a method for simulating system-level classification accuracy in the presence of circuit nonidealities. This method determines the amount of capacitor mismatch and comparator offset and

noise that the binary CNN can tolerate without accuracy degradation, which specifies the design requirements on the analog signal path.

The statistical parameters of the error model are unit capacitor coefficient of variation σ_u/C_u and the comparator offset and noise standard deviations normalized to the supply voltage, σ_{os}/V_{DD} and σ_n/V_{DD}. Given the statistical parameters, we generate random values for every capacitor and every comparator offset in a model of the 64-neuron array. The set of CNN filters processed by each neuron is predetermined by the architecture. Hence, it is straightforward to transform the weights and biases of each filter into their effective values according to the randomly generated capacitances and offset of the neuron that will process that filter. Then, the transformed CNN performs inference on a set of test images. Each filter computation is performed using the expression in (8), which requires generating and adding a random noise term with standard deviation σ_n/V_{DD} to model comparator noise. This exhaustive method captures not only mismatch between capacitors in a DAC but also mismatch between neurons in the array.

Fig. 9 plots classification accuracy over a range of the statistical parameters, for each nonideality acting alone. Ideally, the binary CNN in Table I achieves 86.05% classification accuracy on the CIFAR-10 test set. At the design point $\sigma_u/C_u = 0.85\%$, $\sigma_{os}/V_{FS} = 1.0\%$ (13 LSB), and $\sigma_n/V_{FS} = 0.1\%$ (1.3 LSB), the classification accuracy with all nonidealities acting together is 86%. This design point has $2\times$ margin in each of the three statistical parameters, meaning that increasing σ_u/C_u, σ_{os}/V_{FS}, and σ_n/V_{FS} each by $2\times$ with all nonidealities acting together results in 86% classification accuracy. We take this design point as a specification for the analog signal path.

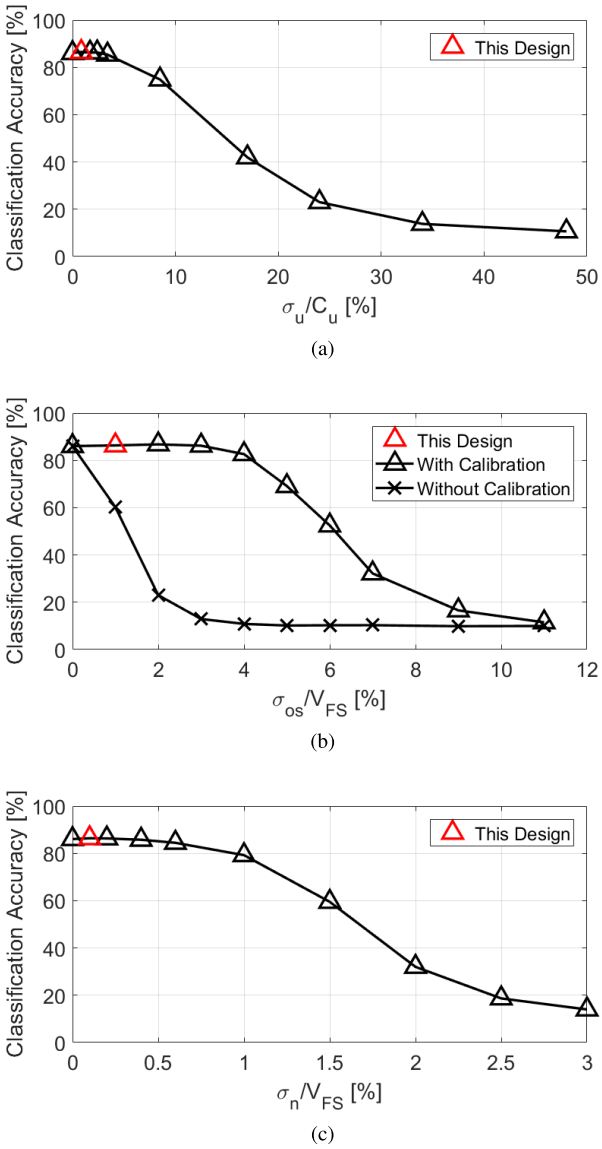

Fig. 9. Behavioral Monte Carlo simulation shows that the design point $\sigma_u/C_u = 0.85\%$, $\sigma_{os}/V_{FS} = 1.0\%$, and $\sigma_n/V_{FS} = 0.1\%$ achieves 86% classification accuracy.

E. Synapse Logic

We refer to the unit cell of the weight-stationary, data-parallel neuron array as a "synapse", shown in Fig. 10. The synapse must at least contain a weight latch and an XNOR gate. In this design, activation signals are driven horizontally in both directions, due to the ping pong style architecture. For simplicity and robustness, we use two separate wires, with one active and one held low during the processing of each CNN layer. An OR gate inside the synapse merges the signals on the two wires. In the SC neuron, the CLR signal is necessary to zero the charge on the top plate node, and additionally data gates the CDAC during weight transfers to prevent it from consuming dynamic energy without computing a useful result.

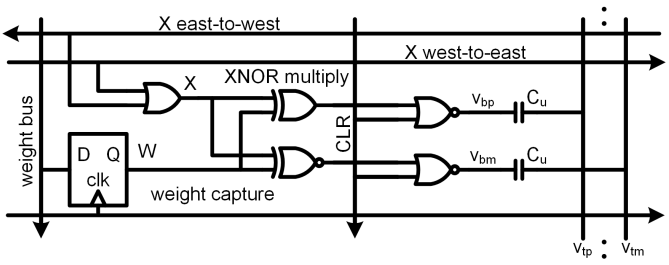

Fig. 10. SC neuron synapse contains a weight latch and logic gates for driving the CDAC's two differential halves. One synapse (including the unit capacitor pair) occupies one standard cell row.

F. MOM Fringe Capacitive DAC

Fig. 11 shows a single unit element of the CDAC. We assume a Pelgrom coefficient $A = 0.85\% \times \sqrt{1\text{fF}}$ based on [32] and calculate the coefficient of variation as follows:

$$\frac{\sigma_u}{C_u} = \frac{A}{\sqrt{C_u}}. \tag{11}$$

Hence, a 1 fF unit capacitance is used to achieve the coefficient of variation specified by behavioral Monte Carlo $\sigma_u/C_u = 0.85\%$. Top plate parasitic capacitances C_{p1} and C_{p2} attenuate signal swing at v_{td}, lowering V_{FS}, and tightening the requirement on the absolute comparator input-referred noise and offset RMS voltages. In SPICE simulation of the SC

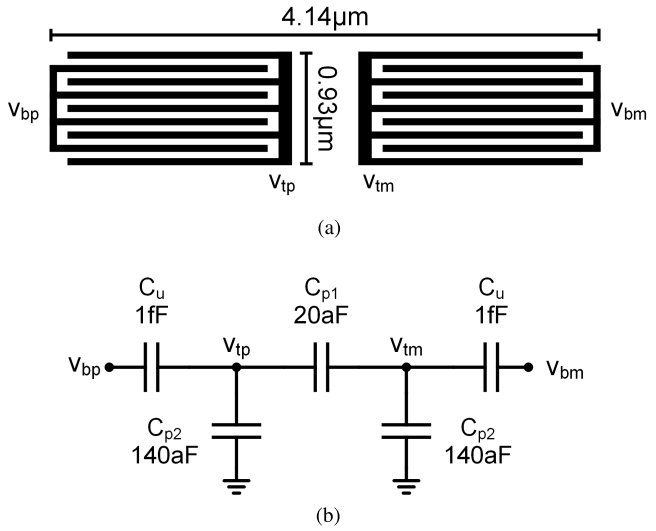

Fig. 11. (a) 1fF unit capacitor layout drawn to scale. (b) Model with top plate parasitics.

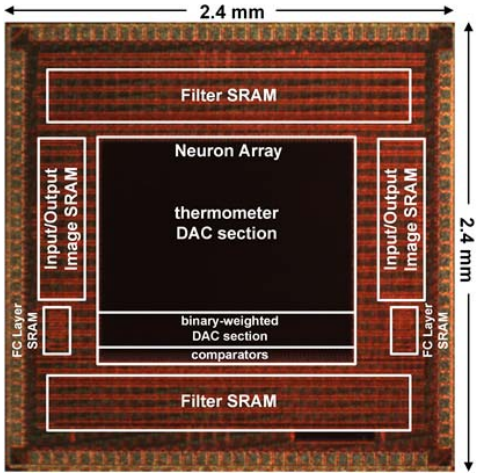

Fig. 12. Die photo.

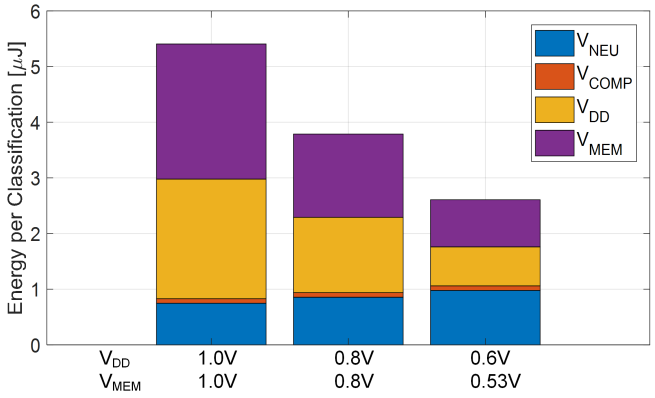

Fig. 13. Measured energy per classification breakdown.

neuron, $V_{\mathrm{FS}} = 460$ mV, which sets the comparator input-referred noise and offset requirements at $\sigma_n = 460$ μV and $\sigma_{\mathrm{os}} = 4.6$ mV. With greater than 1 pF of capacitance in the CDAC, the noise voltage sampled during the CLR operation can be safely neglected.

G. Dynamic Double-Tail Comparator

The dynamic double-tail comparator was selected for its energy efficient fully dynamic operation (no static pre-amp) and its low-voltage compatibility [30]. High transconductance efficiency and low threshold voltage mismatch make it possible to meet the noise and offset requirements with low switched capacitance, leading to a simulated energy per decision of 85 fJ. Section IV shows that this translates to 9% of measured neuron array energy per classification, and 2% of overall core energy per classification.

IV. Experimental Results

The prototype IC was fabricated in TSMC's high-performance low-leakage (HPL) 28-nm CMOS process. Fig. 12 shows a micrograph of the 6 mm ^{2} die. 328 kB of on-chip SRAM was implemented using standard macros. The chip uses a single clock domain. All measurements were taken at room temperature.

Fig. 13 shows the energy breakdown versus V_{DD} and V_{MEM}. The supply V_{MEM} powers image and filter SRAM banks. V_{NEU} serves as a combined digital supply and analog reference for the neuron array, and V_{COMP} powers comparators. The supply V_{DD} powers everything else on chip, including the control finite-state machine (FSM), input–output DEMUX blocks, FC layer weight storing SRAMs, and FC layer digital MAC logic. From post-layout simulation, we estimate that the FC layer consumes less than 2% of the core energy per classification. At nominal supply voltages $V_{\mathrm{DD}} = V_{\mathrm{MEM}} = 1.0$ V, the chip operates up to 380 frames/s (FPS) with $f_{\mathrm{CLK}} = 16$ MHz and achieves 5.4 μJ/classification. Lowering V_{DD} and V_{MEM} to 0.8 V leads to 3.8 μJ/classification (1.43× reduction) at 237 FPS with $f_{\mathrm{CLK}} = 10$ MHz. At this operating point, the neuron array consumes 0.86 μJ/classification from the 0.6-V supply V_{NEU}, and the comparators consume 81.2 nJ/classification from the 0.8-V supply V_{COMP}. Dividing by 983 040 comparator decisions per classification, we obtain 82.6 fJ/decision. Not included in these energy figures is the 1.8-V chip I/O energy, which amounts to 0.43 μJ/classification (a small fraction of the core energy). The energy-efficiency of the 1.8-V chip I/O interface is 13 pJ/bit.

For the purpose of measurement, classification accuracy is defined as the fraction of images classified correctly during one run through the CIFAR-10 test set. Measurements over multiple runs on multiple chips were used to collect statistics, such as the mean and standard deviation of classification accuracy. Fig. 14 shows a histogram of classification accuracy measured over 10 different chips, 180 runs each through the 10 000 image CIFAR-10 test set. These measurements were taken with $V_{\mathrm{DD}} = V_{\mathrm{MEM}} = 0.8$ V, and do not capture process- or aging-related variations. The mean classification accuracy is 86.05%, the same as observed in a perfect digital model of the binary CNN described in Table I. The standard deviation is 0.40%, with spread solely caused by noise and mismatch in the SC neuron array (which can notably lead

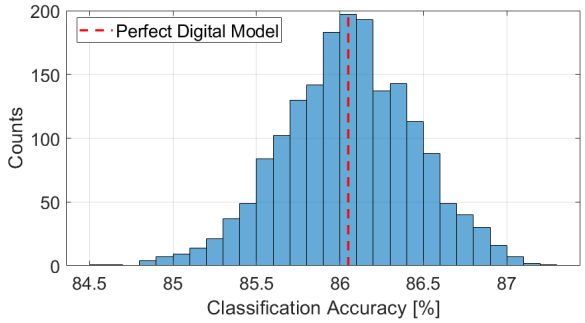

Fig. 14. Measured classification accuracy over 10 chips, 180 runs each through the 10 000 image CIFAR-10 test set.

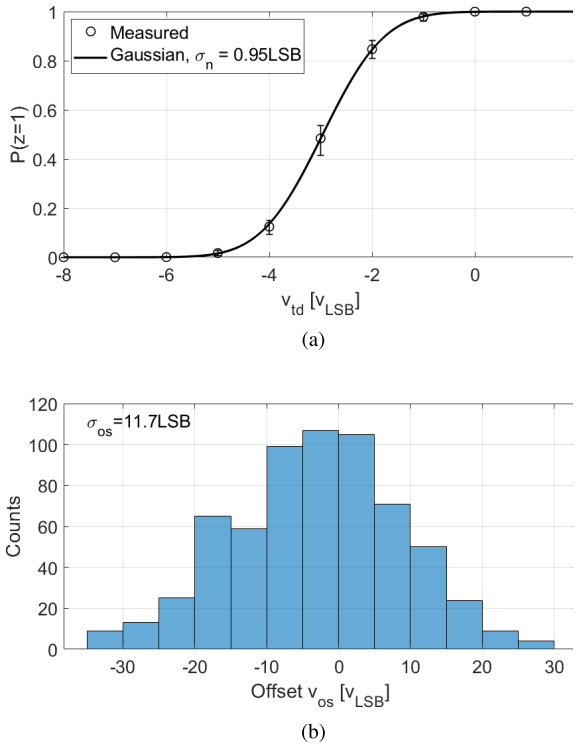

Fig. 15. (a) Gaussian CDF fit extracts the noise and offset from measurements of a single comparator. (b) Measured offset over 10 chips, 64 comparators per chip.

to a higher classification accuracy than in the perfect digital model). The 95% confidence interval in mean classification accuracy is 86.03% to 86.07%.

At $V_{DD} = V_{MEM} = 0.8$ V, digital logic and SRAM macros are error free. To explore further energy savings in the presence of bit errors, we reduced V_{DD} to 0.6 V and set V_{MEM} to 0.53, 0.52, 0.51, and 0.50 V. The mean accuracy degrades to 85.7%, 85.2%, 84.2%, and 82.5%, respectively. At $V_{DD} = 0.6$ V and $V_{MEM} = 0.53$ V (borderline practical), the chip consumes 2.61 μJ/classification, a 2.1$\times$ reduction versus nominal 1.0-V supplies. At this operating point, the frame rate is 36 FPS with $f_{CLK} = 1.5$ MHz. A complete characterization of classification accuracy in the presence of bit errors and the resulting energy savings is presented in [33]. From the energy breakdown in Fig. 13, we see that neuron array energy

TABLE II
COMPARISON TO STATE OF THE ART

	This work	IBM TrueNorth [27]	VLSI '17 [34]	VLSI '17 [35]
Technology	28nm	28nm	65nm	40nm
Algorithm	CNN	CNN	DNN	LCA
Dataset	CIFAR-10	CIFAR-10	MNIST	MNIST
Weight bits Activation bits	1 1	1.6 1	1.6 1	4 1
Supply [V]	0.6, 0.8	1.0	0.55 - 1.0	0.9
Classification Accuracy [%]	86.05	83.41	90.1	88
Energy per Classification [μJ]	3.79	164	0.28 - 0.73	0.050
Power [mW]	0.899	204.4	50 -600	87
Frame Rate [FPS]	237	1249	820K - 3280K	1.7M
Arithmetic Energy Efficiency	532 1b-TOPS/W	-	6.0 - 2.3 TOPS/W	3.43 TOPS/W

increases due to leakage at the lower FPS imposed by voltage scaling. However, this increase is small compared to the logic and memory savings.

Comparator noise was measured in units of LSB by applying a ramp input using the CDAC and counting the number of logic 1s resolved in a sequence of 1024 decisions at each input voltage. Fig. 15(a) shows the resulting $P(z = 1)$ versus v_{td} curve for a single comparator and the Gaussian CDF fit, indicating a noise standard deviation of 0.95 LSB. Error bars show the minimum and maximum of the proportion of logic 1s over 10 repetitions of the measurement. Averaging over 10 chips, 64 comparators per chip, we find $\sigma_n = 0.97$ LSB, which meets the 1.3 LSB requirement specified by behavioral Monte Carlo. The comparator's offset voltage in units of LSB is negative the mean of the Gaussian CDF fit. For the particular comparator of Fig. 15(a), the offset voltage equals 2.96 LSB. Fig. 15(b) shows a histogram of offset voltages over 10 chips, 64 comparators per chip. The offset standard deviation $\sigma_{os} = 11.7$ LSB, which meets the 13 LSB requirement specified by behavioral Monte Carlo.

Table II compares this paper with prior art. To our knowledge, TrueNorth [27] is the only dedicated neural network chip for which post-silicon energy per classification and accuracy on CIFAR-10 have been reported to date. Furthermore, TrueNorth previously set the state of the art for minimum energy on CIFAR-10, at 164 μJ/classification. Our intent in comparing with TrueNorth is simply to report an advancement in energy per classification. This advancement stems from specializing the mixed-signal binary CNN processor to the *CMOS-inspired* topology, which sacrifices some programmability. The designs in [34] and [35] are provided in Table II for context, illustrating that lower energy per classification can be attained on the MNIST handwritten digit data set, an inference task of significantly lower complexity. The binarized DNN accelerator in [34] has all memory on-chip, but because fully-connected neural networks do not reuse weights, the energy

TABLE III
MIXED-SIGNAL VERSUS DIGITAL COMPARISON

	This Work [21]		BinarEye [22]			
Design Style	Mixed-Signal		RTL-to-GDSII Digital			Hand-Designed Digital
Results	Measured		Measured			Estimated
Technology	28nm CMOS, HPL 1P_8M, 4X2Y1Z		28nm CMOS, HPL 1P_8M, 5X1Z1U			28nm CMOS, HPL 1P_8M, 4X2Y1Z
Core Area [mm^2]	4.6		1.4			-
Neuron Array Area [mm^2]	1.6		0.85			1.7
Number of Filters, Number of Channels	256, 256		256, 256	128, 128	64, 64	256, 256
Supply V_{NEU} [V]	0.6	0.6	0.72	0.72	0.72	0.6
Supply V_{COMP} [V]	0.8	0.8	-	-	-	-
Supply V_{DD} [V]	0.8	0.6	-	-	-	0.8
Supply V_{MEM} [V]	0.8	0.53	0.7	0.7	0.7	0.8
9-layer CNN CIFAR-10 Classification Accuracy [%]	86.05	85.69	86	82	76	86.05
9-layer CNN CIFAR-10 Core Energy per Classification [μJ]	3.79	2.61	15	3.8	1.1	6.7
9-layer CNN CIFAR-10 Neuron Array Energy per Classification [μJ]	0.94	1.06	12.1	3.1	0.84	3.9
Clock Frequency [MHz]	10	1.5	9.5	10.6	13.5	10
Throughput [FPS]	237	36	238	945	3791	237
Power [mW]	0.899	0.094	3.9	3.7	4.2	1.6
Arithmetic Throughput [1b-GOPS]	478	72	478	478	478	478
Arithmetic Energy-Efficiency [1b-TOPS/W]	532	772	134	132	122	299

cost of an SRAM bit load is attached to each XNOR operation. However, this accelerator has the advantage that its processing-in-memory (PIM) module can be densely tiled and used to process DNN layers in a pipelined manner. The design in [35] implements the Locally Competitive Algorithm (LCA) using an array of 512 internally analog neurons with fully digital I/O, and exhibits low energy but does so at relatively low accuracy on the MNIST data set. However, the design in [35] is distinguished by its capability of on-chip learning.

V. COMPARISON OF MIXED-SIGNAL AND DIGITAL IMPLEMENTATIONS

In this section, we compare the SC neuron array to its equivalent digital implementation, in order to quantify the advantage of mixed-signal processing. In [36], it was shown that analog filters possess greater energy efficiency than digital in the low-SNR regime. Although neural networks share the same MAC building block as filters, the applicability of this result is limited to mixed-signal processing due to the requirement for dense digital storage. In [15], it was shown that the energy overhead of switches becomes dominant at low SNR for one instance of mixed-signal processing for neural networks, making the advantage of purely analog processing unrealizable. Although the energy of the SC neuron array is limited by its digital gates, it nonetheless achieves an energy savings of 12.9× relative to a synthesized digital neuron array, and an energy savings of 4.2× relative to a hand-designed digital neuron array. This is attributed to the low effective switched capacitance of the CDAC and the low energy per decision of the comparator, made possible by precise matching of MOM fringe capacitors, high transconductance efficiency, and low threshold voltage mismatch.

A. BinarEye

The BinarEye chip presented in [22] has the same baseline architecture as this work, but uses a digital neuron array and was synthesized with an RTL-to-GDSII standard cell flow. BinarEye adds the flexibility to modulate the number of filters and channels in the CNN, which provides an extra system-level knob to increase throughput and decrease energy consumption at the cost of classification accuracy. Whereas this design uses strictly $2 \times 2 \times 256$ filters, BinarEye can be programmed to use filters of size $2 \times 2 \times 256$, $2 \times 2 \times 128$, or $2 \times 2 \times 64$. Table III compares the mixed-signal binary CNN processor of this work with BinarEye, and Fig. 16 plots the measured core energy per classification versus classification accuracy for both designs at several operating points. The third column of Table III shows the estimates extrapolated from post-layout simulation of a single hand-designed digital neuron, which we address in Section V-B.

Whereas this design consumes 3.8 μJ/classification at 86.05% accuracy on CIFAR-10, BinarEye consumes 15 μJ/classification at the same accuracy and throughput.[1] This energy difference is attributed to the SC neuron array, which consumes 12.9× lower energy per classification than the synthesized, digital neuron array of BinarEye.[2] However, BinarEye can scale energy consumption quadratically with

[1]With the SC neuron array, convolution must be periodically paused to zero the charge on the CDAC top plate node (CLR operation). As a result, the mixed-signal design takes 42 000 clock cycles per classification, as opposed to 40 000 cycles achieved by BinarEye which does not require CLR operations.

[2]Because BinarEye uses a single power domain for the neuron array and control/datapath logic, the neuron array energy per classification reported for BinarEye in Table III is based on a combination of measurements and simulation.

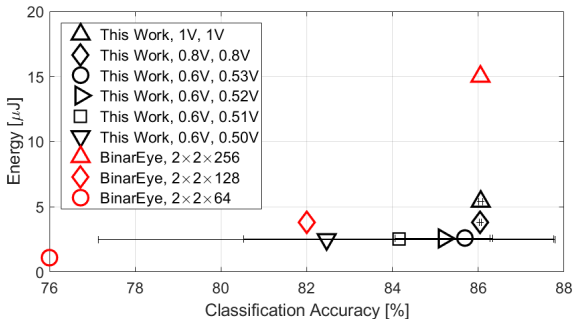

Fig. 16. Measured energy-accuracy tradeoff for this work (supply voltage scaling) and BinarEye (neuron width scaling).

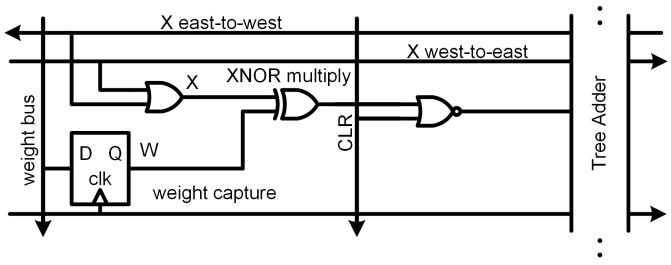

Fig. 17. Digital neuron synapse.

neuron width (by dividing each neuron into parallel sub-neurons) if lower accuracy is permitted. Using $2\times2\times64$ filters, BinarEye reduces energy per classification to 1.1 μJ and increases throughput to 3791 FPS, at 76% classification accuracy on CIFAR-10. This design consumes greater silicon area than BinarEye, leading to higher die cost. The available chip area was limited for the BinarEye tape-out, which adversely affected routability. This resulted in additional vertical wiring and buffering, and led to the neuron array energy being dominated by wires and buffers. The larger chip area available for the mixed-signal design allowed us to enforce a strict grid structure on the neuron array layout, thereby minimizing the length of array-traversing wires.

In this work, we explored circuit techniques that lower energy consumption at the cost of random classification errors due to the underlying circuit nonidealities, such as thermal noise and threshold voltage mismatch. The energy per classification of 3.8 μJ is measured at $V_{\mathrm{DD}} = 0.8$ V and $V_{\mathrm{MEM}} = 0.8$ V, where the 95% confidence interval in mean classification accuracy is 86.03% to 86.07%. For the memory voltage overscaling measurements, the large error bars are due to SRAM V_{MIN} variations across 10 chips [33]. BinarEye uses supply voltages $V_{\mathrm{DD}} = 0.72$ V and $V_{\mathrm{MEM}} = 0.7$ V, where compute and memory are both error free. By modulating the filter size, BinarEye lowers energy at the cost of accuracy in a deterministic way.

B. Hand-Designed Digital Neuron

The SC neuron array of this paper consumes 12.9× lower energy per classification than the synthesized digital neuron array of BinarEye for two reasons: 1) it performs addition via charge redistribution and 2) it has a full custom design and layout. To isolate the energy improvement of mixed-signal processing versus digital processing, as opposed to the energy improvement of hand-designed circuits versus synthesized circuits, we additionally compare against a hand-designed digital neuron in this section. The hand-designed digital neuron performs the same arithmetic function specified by (2), but it differs from the SC neuron of Fig. 8 in that the CDAC has been replaced by a Wallace tree adder. The hand-designed digital neuron is wider than the SC neuron due to the adder (29 μm vs 21 μm), but without the binary-weighted CDAC section, it is vertically shorter (922 μm vs 1210 μm). The hand-designed digital neuron synapse shown in Fig. 17 only needs to drive a single input of the tree adder, rather than two differential halves of the CDAC. However, the CLR signal serves the same data-gating function as in the SC neuron, so the NOR gate driver is retained.

At a single node in a static CMOS digital circuit, the energy consumed per cycle is

$$E_{\mathrm{cyc}} = \alpha C_{\mathrm{sw}} V_{\mathrm{DD}}^2 + P_{\mathrm{leak}} T_{\mathrm{clk}} \tag{12}$$

where α is the activity factor (the probability of a 0-to-1 transition), C_{sw} is the switched capacitance at the node, P_{leak} is the leakage power of the driver, and T_{clk} is the clock period. To characterize the energy consumption in a complete circuit, we apply stimuli over a range of α and perform a linear fit, the slope of which is $C_{\mathrm{sw}} V_{\mathrm{DD}}^2$, where C_{sw} is the effective switched capacitance of the complete circuit.

Fig. 18 shows the energy per cycle versus input activity factor in post-layout simulation for the SC and hand-designed digital neurons. All weight latches are set to logic 1, and an I.I.D. Bernoulli stimulus is applied to each activation input with probability p varying from 0.01 to 0.99. In this manner, $\alpha = p(1-p)$ is swept from 0.0099 to 0.25. Due to the Wallace tree adder, energy per cycle in the digital neuron is weakly nonlinear in activity factor. For the CDAC, it can be shown that the expected energy per cycle under I.I.D. Bernoulli stimulus in the thermometer-coded section is

$$\mathbf{E}[E_{\mathrm{cyc}}] = \left(1 - \frac{C_u}{C_{\mathrm{tot}}}\right) \alpha N C_u V_{\mathrm{DD}}^2. \tag{13}$$

This explains the close linear fit for energy per cycle in the SC neuron versus activity factor. The effective switched capacitance of the SC neuron is 18.1 pF, 2.9× lower than that of the digital neuron. Only 12.7% is contributed by the CDAC, with the remaining capacitance contributed by the synapse logic gates.

The neuron array energy per classification has four components: 1) filter computations; 2) weight transfers; 3) CLR operations; and 4) leakage energy over the idle time. The above-mentioned results show how the neuron's effective C_{sw} for filter computations is extracted from simulation. We additionally extract the neuron's effective C_{sw} for weight transfers, and simulate the energy of a CLR operation and the leakage power. Based on these quantities and the measured SC neuron array energy per classification, we estimate an activity factor $\alpha = 0.059$. Then, we use the simulated energy values at this activity factor to break down the measured energy per

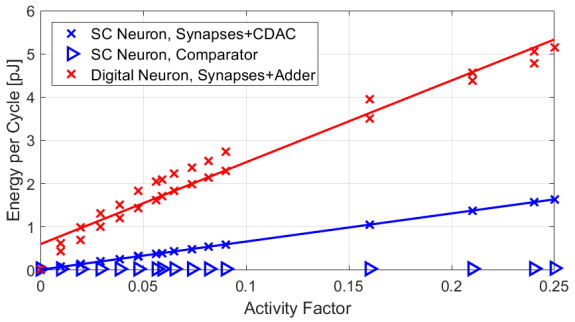

Fig. 18. Simulated energy per cycle versus activity factor.

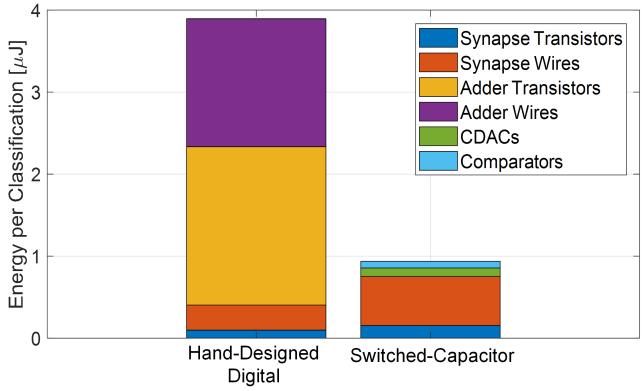

Fig. 19. Simulated neuron array energy per classification breakdown.

classification of the SC neuron array into its components and to estimate the energy per classification of a hand-designed digital neuron array. Because filter computations account for more than 94% of the energy in both cases, we neglect the other components in the following bar graph. Fig. 19 shows the energy per classification broken down by transistors, wires, CDACs, and comparators.

The energy per classification measured for the SC neuron array is 4.2× lower than that estimated for a hand-designed digital neuron array. Fig. 19 shows that the SC neuron array energy is dominated by wiring in the synapse. Due to the additional gates required to drive two differential CDAC halves, the SC synapse energy is 1.9× greater than the digital synapse energy. However, this is outweighed by the fact that the CDACs and comparators together consume 19× lower energy than the equivalent Wallace tree adders.

The parameters of the 28-nm technology make it possible for the CDACs and comparators to meet the error requirements of BinaryNet at low energy. Precise matching between MOM fringe capacitors allows a 1-fF unit capacitance, reducing the energy per classification of the CDACs to 104 nJ. High transconductance efficiency and low threshold voltage mismatch reduce the energy per classification of the comparators to 81.2 nJ while maintaining a noise and offset of 0.97 LSB and 11.7 LSB, respectively. In comparison, the estimated energy per classification of the digital static CMOS Wallace tree adders is 3.49 μJ.

C. Discussion

Together with [22], this study explores three circuit implementations of the same baseline architecture: synthesized digital, hand-designed digital, and mixed-signal. The RTL-to-GDSII flow used in [22] results in the fastest design time and smallest silicon area. With the RTL-to-GDSII flow, the baseline architecture is readily extended to achieve a wider, more flexible tradeoff in the energy-accuracy plane (Fig. 16). However, the top-down place and route flow does not fully take advantage of CNN structural regularity. A full-custom design and layout, whether mixed-signal or digital, allows the length of array-traversing wires to be minimized, thus lowering the energy consumption. The hand-designed digital neuron array consumes 3.1× lower energy than the synthesized digital neuron array in [22], translating to an overall 2.2× reduction in energy per classification at the system level. However, the hand-designed digital neuron array consumes the largest silicon area, owing to the width of the Wallace tree adder. The SC neuron array consumes 4.2× lower energy than the hand-designed digital neuron array, translating to an overall 1.8× reduction in energy per classification at the system level. The mixed-signal implementation has the lowest energy consumption and achieves the same mean classification accuracy, but has a classification accuracy standard deviation of 0.40% due to noise and mismatch in the SC neuron array. In summary, each of the three circuit implementations occupies a distinct point in the tradeoff space of energy, accuracy, throughput, die cost, and design time.

VI. Conclusion

We have presented a mixed-signal binary CNN processor that uses strictly on-chip memory, amortizes access across many computations, and performs addition via charge redistribution. The BinaryNet algorithm drastically simplifies multiplications to XNOR and reduces memory requirements [13]. The *CMOS-inspired* binary CNN topology incurs a 2.55% accuracy loss, but reduces weight memory from 1.67 MB to 261.5 kB and minimizes path loading at the interface between memory and compute. The energy cost of memory access is amortized across many computations using a weight-stationary, data-parallel architecture with input reuse [14]. The remaining energy bottleneck is wide vector summation, which we address using an energy efficient SC neuron. This design is distinguished from the recent CIM designs of [17]– [19] by the fact that it contains the surrounding memory required to time-multiplex the SC neuron array for processing deep CNNs, thereby realizing a complete neural network with image-to-label functionality on chip. Given the error requirements of the binary CNN and the parameters of the 28-nm technology, the SC neuron array operates at 12.9× lower energy than the synthesized digital neuron array in [22], leading to an overall energy savings of 4×. The mixed-signal binary CNN processor achieves 3.8 μJ/classification on the CIFAR-10 data set, a 40× improvement over the previous low energy benchmark.

Acknowledgment

Silicon fabrication was provided by the TSMC University Shuttle Program. The authors would like to thank S. Liao (TSMC) for design support. They would also like to thank Mentor Graphics for providing access to the Analog FastSPICE simulator.

REFERENCES

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst. 25*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds. Red Hook, NY, USA: Curran Associates, 2012, pp. 1097–1105. [Online]. Available: http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf

[2] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, pp. 436–444, May 2015, doi: 10.1038/nature14539.

[3] K. He, X. Zhang, S. Ren, and J. Sun. (2015). "Deep residual learning for image recognition." [Online]. Available: https://arxiv.org/abs/1512.03385

[4] A. Hannun *et al.* (Dec. 2014). "Deep Speech: Scaling up end-to-end speech recognition." [Online]. Available: https://arxiv.org/abs/1412.5567

[5] Siri Team, "Hey Siri: An on-device DNN-powered voice trigger for apple's personal assistant," *Apple Mach. Learn. J.*, vol. 1, no. 6, Oct. 2017. [Online]. Available: https://machinelearning.apple.com/2017/10/01/hey-siri.html

[6] Computer Vision Machine Learning Team, "An on-device deep neural network for face detection," *Apple Mach. Learn. J.*, vol. 1, no. 7, Nov. 2017. [Online]. Available: https://machinelearning.apple.com/2017/11/16/face-detection.html

[7] A. G. Howard *et al.* (2017). "MobileNets: Efficient convolutional neural networks for mobile vision applications." [Online]. Available: https://arxiv.org/abs/1704.04861

[8] D. Franklin. (Mar. 2017). *Nvidia Jetson TX2 Delivers Twice the Intelligence to the Edge*. [Online]. Available: https://devblogs.nvidia.com/jetson-tx2-delivers-twice-intelligence-edge/

[9] M. Horowitz, "Computing's energy problem (and what we can do about it)," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2014, pp. 10–14.

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

[11] O. Russakovsky *et al.*, "ImageNet large scale visual recognition challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, Dec. 2015.

[12] A. Krizhevsky. (2009). *Learning Multiple Layers of Features From Tiny Images*. [Online]. Available: https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf

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

[14] C. Zhang, P. Li, G. Sun, Y. Guan, B. Xiao, and J. Cong, "Optimizing FPGA-based accelerator design for deep convolutional neural networks," in *Proc. ACM/SIGDA Int. Symp. Field-Program. Gate Arrays*, New York, NY, USA, 2015, pp. 161–170, doi: 10.1145/2684746.2689060.

[15] B. Murmann, D. Bankman, E. Chai, D. Miyashita, and L. Yang, "Mixed-signal circuits for embedded machine-learning applications," in *Proc. IEEE Asilomar Conf. Signals, Syst. Comput.*, Nov. 2015, pp. 1341–1345.

[16] J. Zhang, Z. Wang, and N. Verma, "In-memory computation of a machine-learning classifier in a standard 6T SRAM array," *IEEE J. Solid-State Circuits*, vol. 52, no. 4, pp. 915–924, Apr. 2017.

[17] A. Biswas and A. P. Chandrakasan, "Conv-RAM: An energy-efficient SRAM with embedded convolution computation for low-power CNN-based machine learning applications," in *IEEE ISSCC Dig. Tech. Papers*, Feb. 2018, pp. 488–490.

[18] W. H. Chen *et al.*, "A 65nm 1Mb nonvolatile computing-in-memory ReRAM macro with sub-16ns multiply-and-accumulate for binary DNN AI edge processors," in *IEEE ISSCC Dig. Tech. Papers*, Feb. 2018, pp. 494–496.

[19] W. S. Khwa *et al.*, "A 65nm 4Kb algorithm-dependent computing-in-memory SRAM unit-macro with 2.3ns and 55.8TOPS/W fully parallel product-sum operation for binary DNN edge processors," in *IEEE ISSCC Dig. Tech. Papers*, Feb. 2018, pp. 496–498.

[20] 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.

[21] D. Bankman, L. Yang, B. Moons, M. Verhelst, and B. Murmann, "An always-on 3.8μj/86% CIFAR-10 mixed-signal binary CNN processor with all memory on chip in 28nm CMOS," in *IEEE ISSCC Dig. Tech. Papers*, Feb. 2018, pp. 222–224.

[22] B. Moons, D. Bankman, L. Yang, B. Murmann, and M. Verhelst, "BinarEye: An always-on energy-accuracy-scalable binary CNN processor with all memory on chip in 28nm CMOS," in *Proc. Custom Integr. Circuits Conf. (CICC)*, Apr. 2018, pp. 1–4.

[23] A. Karpathy. (2011). *Lessons Learned From Manually Classifying CIFAR-10*. [Online]. Available: http://karpathy.github.io/2011/04/27/manually-classifying-cifar10/

[24] B. Graham. (2014). "Fractional max-pooling." [Online]. Available: https://arxiv.org/abs/1412.6071

[25] R. Zhao *et al.*, "Accelerating binarized convolutional neural networks with software-programmable FPGAs," in *Proc. 2017 ACM/SIGDA Int. Symp. Field-Program. Gate Arrays*, New York, NY, USA, 2017, pp. 15–24, doi: 10.1145/3020078.3021741.

[26] Y. Umuroglu *et al.* (2016). "FINN: A framework for fast, scalable binarized neural network inference." [Online]. Available: https://arxiv.org/abs/1612.07119

[27] S. K. Esser *et al.*, "Convolutional networks for fast, energy-efficient neuromorphic computing," *Proc. Nat. Acad. Sci. USA*, vol. 113, no. 41, pp. 11441–11446, 2016. [Online]. Available: http://www.pnas.org/content/113/41/11441

[28] S. Ioffe and C. Szegedy. (Feb. 2015). "Batch normalization: Accelerating deep network training by reducing internal covariate shift." [Online]. Available: https://arxiv.org/abs/1502.03167

[29] L. Kull *et al.*, "A 3.1 mW 8b 1.2 GS/s single-channel asynchronous SAR ADC with alternate comparators for enhanced speed in 32 nm digital SOI CMOS," *IEEE J. Solid-State Circuits*, vol. 48, no. 12, pp. 3049–3058, Dec. 2013.

[30] D. Schinkel, E. Mensink, E. Klumperink, E. van Tuijl, and B. Nauta, "A double-tail latch-type voltage sense amplifier with 18ps setup+hold time," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, San Francisco, CA, USA, Feb. 2007, pp. 314–605.

[31] M. El-Chammas and B. Murmann, "A 12-GS/s 81-mW 5-bit time-interleaved flash ADC with background timing skew calibration," *IEEE J. Solid-State Circuits*, vol. 46, no. 4, pp. 838–847, Apr. 2011.

[32] V. Tripathi and B. Murmann, "Mismatch characterization of small metal fringe capacitors," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 61, no. 8, pp. 2236–2242, Aug. 2014.

[33] L. Yang, D. Bankman, B. Moons, M. Verhelst, and B. Murmann, "Bit error tolerance of a CIFAR-10 binarized convolutional neural network processor," in *Proc. ISCAS*, May 2018, pp. 1–5.

[34] K. Ando *et al.*, "BRein memory: A 13-layer 4.2 K neuron/0.8 M synapse binary/ternary reconfigurable in-memory deep neural network accelerator in 65 nm CMOS," in *Proc. Symp. VLSI Circuits*, Jun. 2017, pp. C24–C25.

[35] F. N. Buhler, P. Brown, J. Li, T. Chen, Z. Zhang, and M. P. Flynn, "A 3.43TOPS/W 48.9pJ/pixel 50.1nJ/classification 512 analog neuron sparse coding neural network with on-chip learning and classification in 40nm CMOS," in *Proc. Symp. VLSI Circuits*, Jun. 2017, pp. C30–C31.

[36] E. A. Vittoz, "Future of analog in the VLSI environment," in *Proc. IEEE Int. Symp. Circuits Syst.*, vol. 2, May 1990, pp. 1372–1375.

Daniel Bankman (S'13) received the B.S. degree in electrical engineering from the Massachusetts Institute of Technology, Cambridge, MA, USA, in 2012, and the M.S. degree from Stanford University, Stanford, CA, USA, in 2015, where he is currently pursuing the Ph.D. degree, with a focus on mixed-signal processing for machine learning.

He held various internship positions with Analog Devices, Wilmington, MA, USA, and Intel, San Diego, CA, USA. His current research interests include algorithms, architectures, and circuits for energy-efficient learning and inference in smart devices.

Mr. Bankman was a recipient of the Texas Instruments Stanford Graduate Fellowship in 2012, the Numerical Technologies Founders Prize in 2013, and the SONIC John von Neumann Student Research Award in 2015 and 2017.

Lita Yang (S'10) received the B.S. degree (Hons.) in electrical engineering from the California Institute of Technology (Caltech), Pasadena, CA, USA, in 2012, and the M.S. degree in electrical engineering from Stanford University, Stanford, CA, USA, in 2015, where she is currently pursuing the Ph.D. degree, under the supervision of Dr. B. Murmann.

She held various internship positions with Oracle Labs, Redwood Shores, CA, USA, in 2012, Qualcomm Corporate Research and Development, San Diego, CA, USA, in 2014, and TSMC, Hsinchu, Taiwan, in 2015. Her current research interests include energy-efficient hardware for machine learning and computer vision applications, especially low-energy memory that exploits the error resilience of convolutional neural networks (CNNs) on image classification tasks.

Ms. Yang was a recipient of the StarNET SRC John von Neumann Student Research Award for Excellence in Architectures Research in 2015 and 2017.

Bert Moons (S'13) received the M.S. and Ph.D. degrees in electrical engineering from KU Leuven, Leuven, Belgium, in 2013 and 2018, respectively.

He was an IWT-Funded Research Assistant with ESAT–MICAS, KU Leuven, where he was involved in Ph.D. research on energy-scalable and run-time adaptable digital architectures and circuits for embedded deep-learning applications. He was a Visiting Research Student with the Murmann Mixed-Signal Group, Stanford University, Stanford, CA, USA. He is currently a Hardware Architect with Synopsys, Leuven, where he is involved in DesignWare EV6x Embedded Vision and Deep Learning processors. He has authored more than 15 conference and journal publications.

Dr. Moons was a recipient of the SSCS Pre-Doctoral Achievement Award in 2018.

Marian Verhelst (S'01–M'09–SM'14) received the Ph.D. degree *(cum ultima laude)* from KU Leuven, Leuven, Belgium, in 2008.

In 2005, she was a Visiting Scholar with the Berkeley Wireless Research Center (BWRC), University of California at Berkeley, Berkeley, CA, USA. From 2008 to 2011, she was a Research Scientist with the Radio Integration Research Lab, Intel Labs, Hillsboro, OR, USA. She is currently an Associate Professor with the ESAT–MICAS, KU Leuven. Her current research interests include embedded machine learning, hardware accelerators, self-adaptive circuits and systems, and low-power sensing and processing for the Internet-of-Things (IoT). She has authored or co-authored over 100 papers in conferences and journals.

Dr. Verhelst was a member of the Young Academy of Belgium and the STEM Advisory Committee to the Flemish Government. She is currently a member of the DATE Conference Executive Committee, the ESSCIRC and International Solid-State Circuits Conference (ISSCC) TPCs, and the ISSCC Executive Committee. She was a Distinguished Lecturer with the IEEE SSCS. She was a recipient of the Prestigious ERC Starting Grant from the European Union. She served as an Associate Editor for the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS (TCAS)-II and the IEEE JOURNAL OF SOLID-STATE CIRCUITS (JSSC).

Boris Murmann (S'99–M'03–SM'09–F'15) received the Dipl.-Ing. (FH) degree in communications engineering from Fachhochschule Dieburg, Dieburg, Germany, in 1994, the M.S. degree in electrical engineering from Santa Clara University, Santa Clara, CA, USA, in 1999, and the Ph.D. degree in electrical engineering from the University of California at Berkeley, Berkeley, CA, USA, in 2003.

From 1994 to 1997, he was with Neutron Mikrolektronik GmbH, Hanau, Germany, where he was involved in the development of low-power and smart-power application-specified integrated circuits (ASICs) in automotive CMOS technology. Since 2004, he has been with the Department of Electrical Engineering, Stanford University, Stanford, CA, USA, where he is currently a Full Professor. His current research interests include the area of mixed-signal integrated circuit design, with special emphasis on data converters, sensor interfaces, and circuits for embedded machine learning.

Dr. Murmann served as the Data Converter Subcommittee Chair and 2017 Program Chair of the IEEE International Solid-State Circuits Conference (ISSCC). He was a co-recipient of the Best Student Paper Award at the Very Large-Scale Integration (VLSI) Circuits Symposium in 2008 and a recipient of the Best Invited Paper Award at the IEEE Custom Integrated Circuits Conference (CICC) in 2008, the Agilent Early Career Professor Award in 2009, and the Friedrich Wilhelm Bessel Research Award in 2012. He served as an Associate Editor for the IEEE JOURNAL OF SOLID-STATE CIRCUITS.