A 4096-Neuron 1M-Synapse 3.8-pJ/SOP Spiking Neural Network With On-Chip STDP Learning and Sparse Weights in 10-nm FinFET CMOS

Gregory K. Chen, *Member, IEEE*, Raghavan Kumar, *Member, IEEE*, H. Ekin Sumbul, *Member, IEEE*, Phil C. Knag, *Member, IEEE*, and Ram K. Krishnamurthy, *Fellow, IEEE*

***Abstract*— A reconfigurable 4096-neuron, 1M-synapse chip in 10-nm FinFET CMOS is developed to accelerate inference and learning for many classes of spiking neural networks (SNNs). The SNN features digital circuits for leaky integrate and fire neuron models, on-chip spike-timing-dependent plasticity (STDP) learning, and high-fan-out multicast spike communication. Structured fine-grained weight sparsity reduces synapse memory by up to 16× with less than 2% overhead for storing connections. Approximate computing co-optimizes the dropping flow control and benefits from algorithmic noise to process spatiotemporal spike patterns with up to 9.4× lower energy. The SNN achieves a peak throughput of 25.2 GSOP/s at 0.9 V, peak energy efficiency of 3.8 pJ/SOP at 525 mV, and 2.3-μW/neuron operation at 450 mV. On-chip unsupervised STDP trains a spiking restricted Boltzmann machine to de-noise Modified National Institute of Standards and Technology (MNIST) digits and to reconstruct natural scene images with RMSE of 0.036. Near-threshold operation, in conjunction with temporal and spatial sparsity, reduces energy by 17.4× to 1.0-μJ/classification in a 236 × 20 feed-forward network that is trained to classify MNIST digits using supervised STDP. A binary-activation multilayer perceptron with 50% sparse weights is trained offline with error backpropagation to classify MNIST digits with 97.9% accuracy at 1.7-μJ/classification.**

***Index Terms*— Near-threshold voltage circuits, neuromorphic computing, spike-timing-dependent plasticity (STDP), spiking neural networks (SNNs), weight sparsity.**

I. Introduction

SPIKING neural networks (SNNs) draw inspiration from the brain to improve the capabilities and energy efficiency of machine learning. The mammalian cortex achieves cognition that is unmatched by artificial intelligence, all in a volume of ∼1 L with a power of ∼20 W [1]. Meanwhile, the best efforts at simulating a brain-scale computer with ∼100 billion neurons and ∼100 trillion synapses requires supercomputers using megawatts of power to perform calculations at a fraction of the rate of biology [2]. SNNs apply our current knowledge of neuroanatomy to approach the energy efficiency of biology by using temporally sparse spiking communication, spatially sparse connections, reduced precision, and approximate computation. Simultaneously, they harness insights in neurophysiology on temporal spike coding and biologically plausible learning to emulate the intelligence of the brain.

SNNs and neuromorphic processors leverage temporal coding to efficiently communicate data between populations of neurons as spatiotemporal spike patterns [3]. Encoding data in both spike location and time achieve high information coding density, while still employing reduced-precision data and computation [4]. SNNs are comprised many parallel neuron processing elements, which follow spiking neuron models such leaky integrate and fire (LIF). LIF is a simple and well-studied model that integrates inputs, over time, from many fan-in neurons, onto a state variable called the membrane potential. Efficient SNN processing requires scalable, reconfigurable, and high-fan-in connectivity between neurons.

Learning rules such as spike-timing-dependent plasticity (STDP) underlie the brain's learning mechanisms and have the ability to train SNNs with higher computational efficiency than the conventional methods [5]. Networks trained with STDP reinforce spike patterns that relate to previously encountered stimuli and filter out anomalous spiking activity. Biologically plausible operation necessitates that STDP performs event-driven learning using only locally available data. The learning rule updates synaptic weights between pairs of neurons based on the neurons' relative spike timing. Circuit implementation of STDP hinges upon efficient storage and access of neuron spike history and synapse weights.

Several previous works emulate a brain-scale neural network on a computer system in order to better understand both massively parallel computing and brain function. Wong *et al.* [2] process 530B neurons and 137T synapses at 1542× slower than real time on a 96-rack supercomputer. Painkras *et al.* [6] perform computations for 1B neurons and 1T synapses on a computer system comprised many 1-W chip multiprocessors.

Multiple existing solutions implement ASICs to accelerate inference in an SNN with offline training. Merolla *et al.* [7] integrate 1M neurons on a chip, each with 256 1-b input synapses. Benjamin *et al.* [8] implement analog synapses with the lowest overhead in a shared-dendrite configuration that precludes synaptic plasticity. Other works implement on-chip learning using techniques other than STDP. Knag *et al.* [9] implement a two-layer network on LIF neurons that are trained on-chip with the SAILnet sparse coding algorithm.

Manuscript received August 16, 2018; revised November 14, 2018; accepted November 23, 2018. Date of publication December 25, 2018; date of current version March 25, 2019. This paper was approved by Guest Editor Ken Takeuchi. *(Corresponding author: Gregory K. Chen.)*

The authors are with the Circuit Research Lab, Intel Corporation, Hillsboro, OR 97229 USA (e-mail: gregory.k.chen@intel.com).

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.2018.2884901

A number of recent chips implement on-chip learning with STDP. Cruz-Albrecht *et al.* [10] simulated training on a chip with weights stored in post-processed integrated memristors. The chip connects 576 neurons with a compiled programmable routing fabric and connectivity stored in SRAM. Seo *et al.* [11] implement STDP for fully recurrent 256-neuron chips with 1- and 4-b synapses stored in a transposable SRAM. Stochastic STDP is performed for 1-b weights by comparing spike history counters to a 15-b pseudorandom linear-feedback shift register (LFSR) value.

This project builds on previous work by improving the energy efficiency of SNN inference and training through scalable, reconfigurable processing of spatially and temporally sparse events. Low-latency high-fan-out communication is achieved with multicast spike delivery on a network on chip (NoC). The multicast communication allows STDP circuits to efficiently gather spike history from distributed locations across the chip to calculate weight updates.

Synaptic connections in the brain are sparse to reduce resource utilization. Likewise, this chip leverages sparse weights to reduce the SNN model size. Sparse synapses require less weight memory, fewer computations, and lower data transfer energy. Fine-grained structured sparsity is used to reduce the amount of memory required for storing connections, while still achieving high accuracy. Weight sparsity results in graceful accuracy degradation compared to dense connectivity since neural networks are often overparameterized [12]. Structured sparsity enables hardware-algorithm co-design by grouping synapses to make them easier to store and access [13], [14]. Fine-grained sparsity reduces the correlation in connections between groups of neurons, which has been shown to improve machine learning performance in liquid-state machines [15], [16].

The brain is observed to be a noisy environment, with trial-to-trial variability for the same stimuli. In machine learning, noise is known to act as a regularization technique during training, to prevent overfitting. Blank-out noise masks out synapses through multiplication with a Bernoulli process and has been shown as an effective tool for improving network generalization [17]. Stochastic operation also implements neural sampling during inference to process uncertainty with ambiguous inputs [18]. Similarly, this paper co-optimizes algorithmic noise with energy efficiency using approximate computing and a dropping flow control. These techniques are shown in a 4k neuron, 1M synapse SNN chip in 10-nm FinFET CMOS. The design is measured to achieve the highest reported SNN inference and training energy efficiency. SNN operation is demonstrated for three workloads: 1) image reconstruction using a restricted Boltzmann machine (RBM) trained with unsupervised STDP; 2) classification using an SNN trained with supervised STDP; and 3) classification using a binary-activation neural network (BNN) trained with error-backpropagation.

The rest of this paper is organized as follows. Section II describes the SNN operation and circuit optimizations. Section III shows the measurement results including circuit results and three workload examples. Section IV discusses future implications of this paper. Section V concludes this paper.

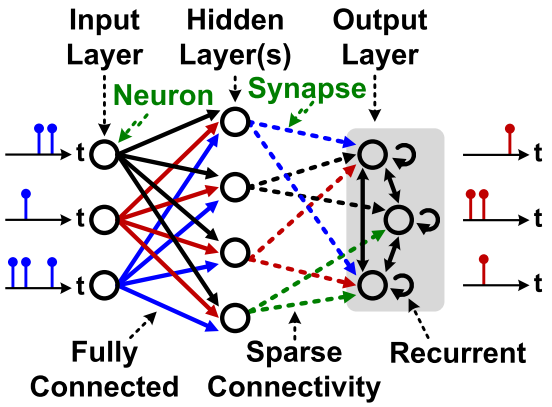


Fig. 1. SNN topology and terminology.

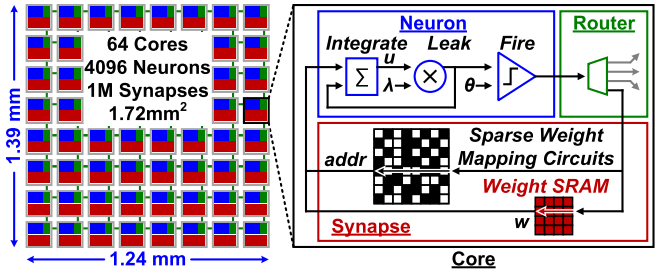


Fig. 2. SNN chip overview.

II. Spiking Neural Network Operation

SNNs use neuromorphic neuron models and learning rules to solve machine learning problems. They are comprised a network of neuron processing elements that are connected with weighted synapses (Fig. 1). The processing elements sum weighted inputs from many fan-in neurons and generate a temporal sequence of 1-b output spikes. The synaptic connections between layers of neurons can be dense or sparse. During training, SNNs learn to complete machine learning tasks by looking at examples. They examine training data and perform incremental weight updates to minimize a loss function or reconstruction error. During inference, the trained SNN maps inputs to the desired outputs with a certain accuracy or error.

The SNN chip contains 4k digital LIF neurons and 1M synapses, divided into 64 cores, and connected by an NoC (Fig. 2). Within a core the neuron block stores state and performs LIF calculations for 64 neurons. The synapse block contains 16k synapse weights stored in SRAM, computes the network's structured sparse connectivity, and delivers weights to the target neurons. Each SNN neuron is mapped directly to one of the cores on the test chip and remains stationary for the duration of the SNN operation. Synapses are similarly directly mapped to an address location in the weight SRAM. All communication is performed by sending spikes between these blocks on the NoC. Neuron spikes are multicast to synapse groups across cores using NoC routers that implement programmable SNN connectivity. Core-to-core connection is explicitly mapped onto the chip by the user, and connections within a core are either fully connected or controlled by the weight sparsity generation circuits. Recurrent connections

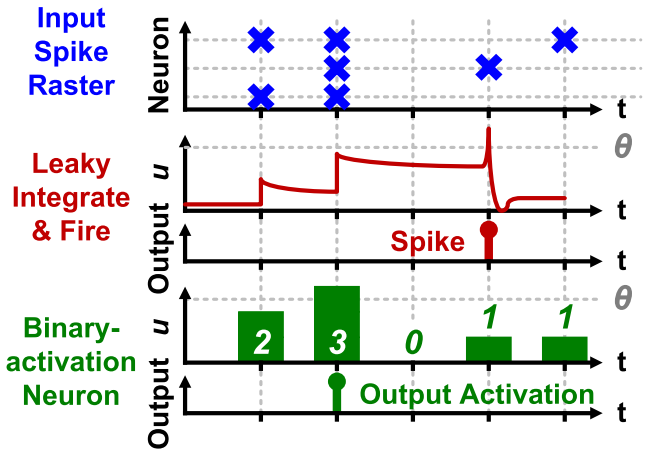


Fig. 3. LIF neurons integrate weighted spikes onto the membrane potential and generate an output spike when the net effect of inputs over time exceeds a threshold.

send spikes through the synapse core back to their source neuron core. The energy breakdown between components is 49% synapse memory and 51% in the remaining logic for computations and storing neuron state.

A. Spiking Neuron Operation

SNNs perform temporal processing by integrating weighted spike inputs across time onto the membrane potential. This state variable is retained across time steps with leakage to prioritize recent inputs. The equation governing LIF neuron state is the following:

$$u(t+1) = \lambda u(t) + \sum \text{sw} + B + I. \tag{1}$$

In (1), $u(t)$ denotes the membrane potential at time t, λ is a leak factor with $0 < \lambda < 1$, s is an input spike, w is the spike's weight, B is a trainable bias, and I is a primary input to the SNN.

The LIF response to an example input spike raster is shown in Fig. 3. Spikes arrive from many input neurons (y-axis), at different times (x-axis) with time discretized into time steps. Each weighted input is accumulated onto the membrane potential, denoted as u, and retained across time steps. LIF neurons spike when the net effect of all input spikes across time causes the membrane potential to exceed a threshold (θ). This is in contrast to binary-activation neurons, which are equivalent to LIF where membrane potentials (partial sums) are reset at the end of every time step.

At each time step, the neuron performs integration by multiplying a weight matrix with a sparse spike vector (Fig. 4). Spikes are delivered to the synapses using address event representation (AER), as a list of addresses for neurons that spike [19]. AER does not send any data for non-spiking neurons to reduce bandwidth requirements at low spike rates. Weights are accessed for neurons that spike only, reducing the number of computations and memory reads. The weights are integrated onto the target neurons' membrane potentials. The neuron state is split into two 64-entry register files (RFs), with 20 b for weight integration and 32 b in the fire RF for time step updates.

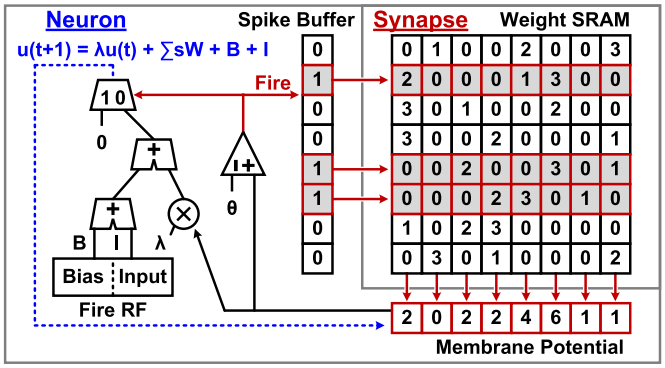


Fig. 4. Neuron circuits multiply the weight matrix with the sparse spike vector and for integrate the result into the membrane potential RF.

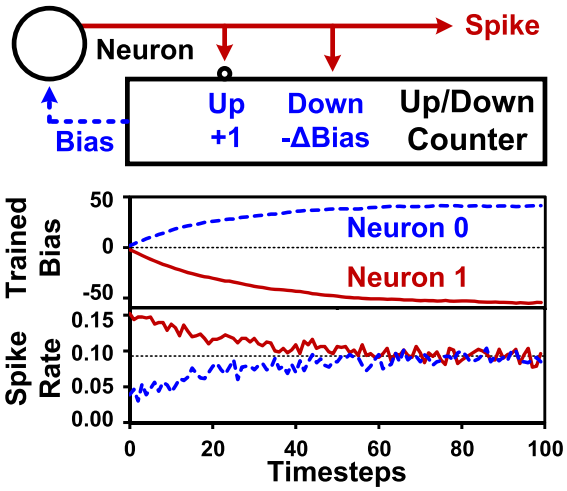


Fig. 5. Homeostasis sets the average spike rate with negative bias feedback.

The neuron group performs leak and fire operations once per time step. Each neuron compares its membrane potential to a threshold value. If the potential exceeds the threshold, the neuron spikes and its potential is reset to 0. Otherwise, the potential leaks to make the effect of input spikes fade over time. The neuron then adds in a trainable bias and the SNN's primary inputs. The bias implements intrinsic plasticity homeostasis to set average spike rate (Fig. 5). The homeostasis circuit monitors spiking activity and applies negative feedback to the bias using an up/down counter with programmable decrements.

B. STDP

SNNs are trained with STDP to recognize spatiotemporal spike patterns by learning from relative spike timing between pairs of neurons. The primary form of STDP is unsupervised, meaning that training occurs during any operation without requiring the calculation of a loss function or teacher signal. Spike patterns are input to the SNN and permanent weight updates are performed based on the following pairwise rule. Long-term potentiation (LTP) increments weights when spike timing follows the direction of spike propagation, and thus, the spikes are likely to be causal (Fig. 6). Oppositely, long-term depression (LTD) decrements weights when spike timing

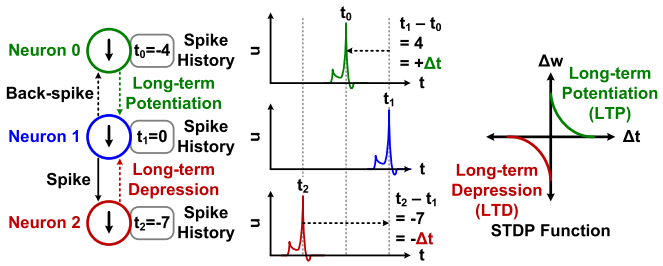


Fig. 6. STDP overview.

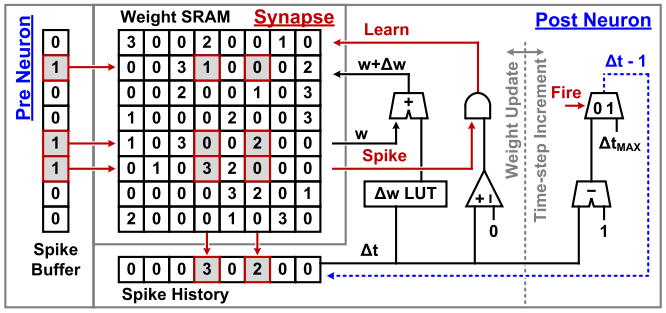


Fig. 7. STDP circuits, spike history counters, and weight LUT for event-driven update of synapses connecting neurons that fire in close succession.

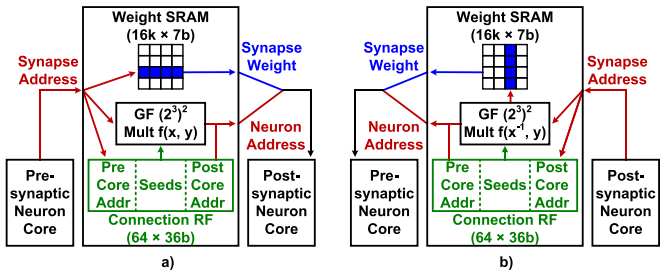


Fig. 8. Structured connectivity enables a low-memory method of generating sparse weights. A permutation function calculates a target neuron for each synapse. (a) Forward and (b) reverse pathways are implemented with an invertible permutation function to perform on-chip STDP learning.

flows against spike propagation, and thus, the spikes are likely uncorrelated. The weight update value is exponential with spike time, making spikes that are coincident have a stronger effect on learning, whereas spikes that are far apart in time have less or no effect. Synapse weights have a 7-b resolution, allowing for slower learning rates and improved accuracy, while still having a low-memory footprint.

On-chip, the event-driven STDP update check is performed when a neuron spike, by collecting the spike history of each connected neuron in the forward and backward directions. LTD and LTP are performed in symmetrical ways, using forward and back-spikes. Back-spikes implement LTP but do not participate in weight integration except in the case of RBMs and other undirected SNNs.

To implement STDP, each neuron keeps track of its own spike history (Δt) (Fig. 7). This history is retrieved by forward and back spikes to calculate weight updates. The spike history is initialized to a maximum value when the neuron spikes. This determines the time duration of the STDP function. Then, the spike history counts down the number of time steps since its previous spike, saturating at 0 to indicate that the neuron will not participate in learning at this time. If the spike history is non-zero, then upon receiving a spike, the neuron indexes a lookup table (ΔW LUT) with the spike history value to retrieve a weight update. It then updates the corresponding weight in the synapse memory. Fig. 7 shows the example of LTD. LTP is performed in the same way with the role of the pre-/post-neurons and the polarity of the weight update reversed.

C. Sparsity

In addition to temporally sparse spikes, the SNN chip implements spatially sparse weights. Model size reduction is critical since weights take up 56% of chip area. The technique first sparsifies the SNN using a fine-grained function. Then, the network is trained while enforcing the fixed connectivity. This method reduces the memory requirement for storing sparse connectivity by obviating storage of target neuron addresses for each synapse. The required connection data includes pointers for mapping between cores and seeds to set connections within a core (Fig. 8). The 64 × 36 b connection RF requires less than 2% overhead compared to the 16 k × 7 b weight SRAM. For a 75% sparse SNN, structured sparsity uses 2.7× less memory than storing neuron address pointers, and 3.9× less memory compared to a dense implementation.

The sparse connections are generated prior to training time and recalculated at runtime by the synapse group using a Galois field (GF) permutation function. The 6-b GF function multiplies the synapse memory address with a user-defined seed in a finite field to calculate the target neuron within the destination core. Different pseudorandom connectivities are generated by changing the seed values. The function is reversible, enabling a backwards path way via multiplication with the inverse of the seed. This implements back-spikes for STDP learning and undirected synapses for generative networks such as RBMs. The permutation function inputs are mapped to a composite-field of $GF(2^3)^2$ from $GF(2^6)$, performing 3 b instead of 6-b logic to enable area and energy-efficient computation [20]. The GF function is defined by the pair of polynomials: $x^3 + x + 1$ (ground field) and $x^2 + x + 1$ (extension field). This polynomial pair yielded the lowest area for the GF multiplier after analyzing all the possible composite-field mapping polynomials.

The SNN chip generates a programmable level of fine-grained sparsity, ranging from fully connected to 15/16 sparse (Fig. 9). The GF operation is repeated until the user-defined sparsity level is met. Structured sparsity reduces the number of memory accesses and computations, allowing both processing to complete faster and decreasing energy by up to 15× per time step with 16× fewer weights (Fig. 10).

D. Stochastic Synapses

The SNN chip uses an approximate computing technique that employs time step acceleration to implement stochastic synapses and co-optimize throughput and noise. The method controls the number of clock cycles allocated to process each algorithm time step. Allocating one clock cycle for every possible spike in the time step is wasteful since SNN activity

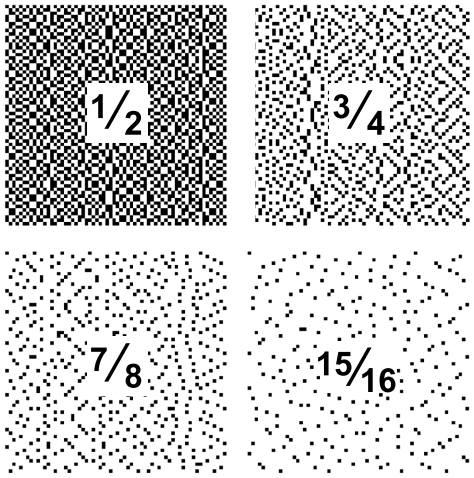


Fig. 9. Example 64 × 64 connectivity maps with user-defined sparsity.

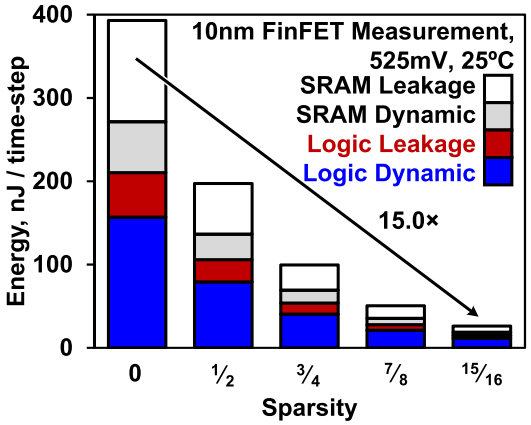


Fig. 10. Energy per time step savings for increasing weight sparsity.

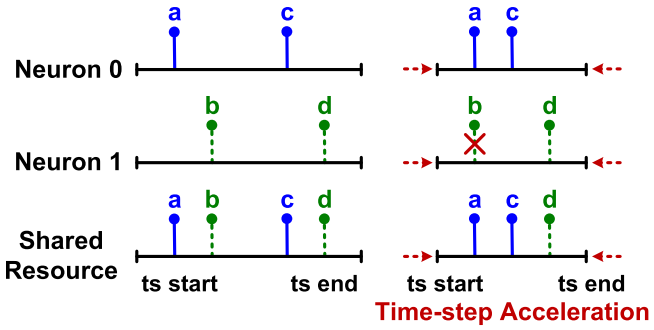


Fig. 11. Approximate computing with time step acceleration reduces the number of clock cycles required to execute each time step and generates algorithmic noise with selective dropping of spikes at bandwidth bottlenecks.

is sparse and would result in performance loss and a leakage energy cost (Fig. 11). Time step acceleration reduces the number of clock cycles per time step. This forces some dynamically calculated spike onto shared resources in the same clock cycle. At these bandwidth bottlenecks, the SNN drops a controlled number of spikes to inject noise into the network. Randomizing the clock cycle within a time step that a spike is processed improves noise characteristics in the SNN. At the algorithm level, spikes in the same time step are concurrent, resulting in no change to nominal SNN operation.

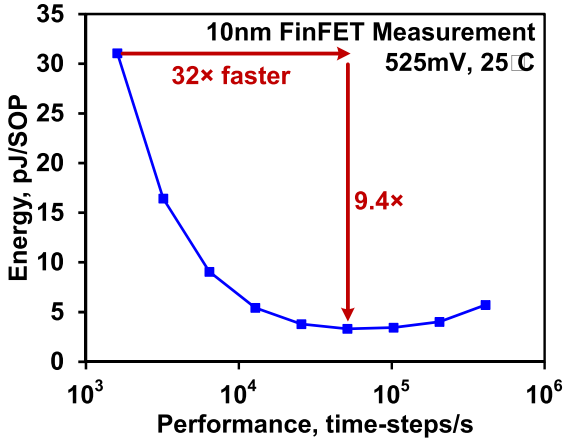


Fig. 12. Energy and throughput improvements with approximate computing.

Time step acceleration is compared with scheduled execution, where each synaptic operation (SOP) is allocated one clock cycle. For a scheduled SNN, the number of clock cycles scales with the allowed neuron fan-in, since the maximum number of operations increases. Compared to a scheduled SNN with fan-in of 1024, time step acceleration allows arbitrary fan-in and simultaneously increases throughput by 32× and decreases energy by 9.4× (Fig. 12).

Time step acceleration is employed in the NoC with a dropping flow control. When spike packets compete for a router output port, all but one are dropped, generating noise. The NoC has a flattened butterfly topology to make the noise generation more uniform as a function of the packet source and destination. In this two-ary six-flat, routers connect to all other routers with an Hamming distance of 1 in their addresses. Since each bit flip between the destination core and local router addresses corresponds to a productive hop, route computation is performed one bit at a time. Candidate hops are generated through bitwise comparison of the local and destination addresses (Fig. 13). The NoC selects from these candidate hops randomly and take different routes between the same source and destination cores to improve noise characteristics (Fig. 14). A pseudo-random number generator (PRNG) using a combination of LFSRs and rule 30 cellular automata [21] selects from among the candidate hops and forward spike packets to the corresponding router output port.

The NoC delivers multicast spikes to reduce serialization latency. Multicast route computation calculates a spanning tree of destination cores in a distributed manner across the NoC. To achieve this, each bit in the address for a set of destination cores uses a 2-b symbol. Each symbol can be 0, 1, or "∗," with "∗" indicating that cores with either 0 or 1 at that bit location in their addresses are destinations. When processing the "∗" symbol in a router, spike packets branch to multiple output ports and continue to different subsets of the destination cores (Fig. 15). Multicast operation reduces spike delivery energy by up to 3× over unicast routing.

III. Measurements

The 1.72-mm^2 SNN chip [22] is implemented in 10-nm FinFET CMOS [23] (Fig. 16). 4k neurons and 1M synapses are distributed in 64 cores. The area breakdown for the core

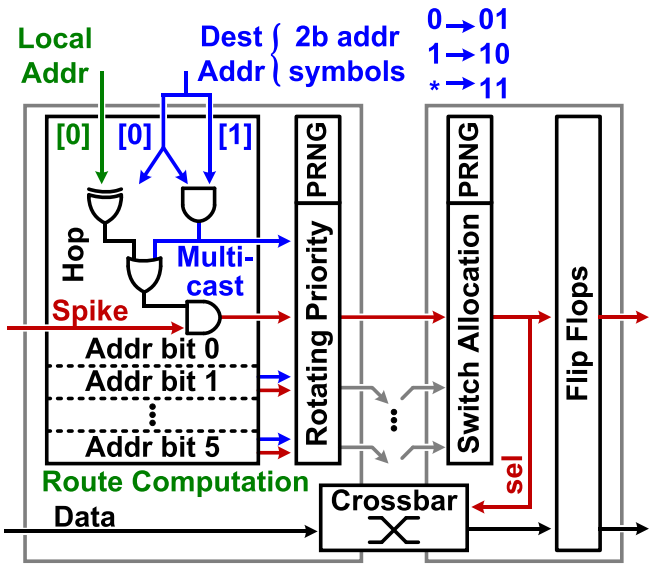


Fig. 13. NoC router schematic including multicast route computation circuits.

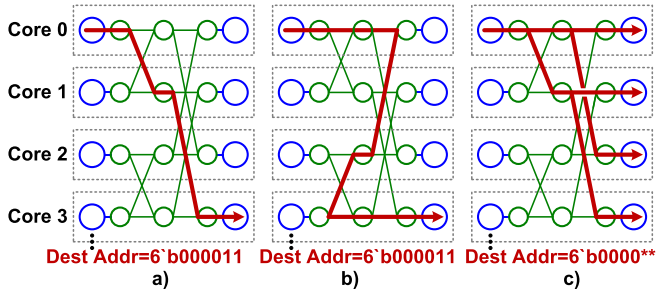


Fig. 14. Example spike delivery paths for randomized routing and multicast. (a) Fewer hops. (b) Path diversity. (c) Multicast.

```
for i in rand_order(dest_addr): #process bits in random order

    if local_addr[i]==dest_addr[i]: # bit matches
        continue # process next bit

    else if local_addr[i]!=dest_addr[i]: # bit does not match
        next_addr = local_addr
        next_addr[i] = dest_addr[i] # flip bit to match dest
        send_packet_toward_next_addr() # forward packet
        break # end route computation

    else if dest_addr[i]== '*': # wildcard bit
        # branch 0
        next_addr = local_addr
        next_addr[i] = !local_addr[i] # flip bit to !local value
        send_packet_toward_next_addr() # forward packet
        # branch 1
        dest_addr[i] = local_addr[i] # resolve '*' to local value
        continue # process next bit
```

Fig. 15. Pseudocode for multicast routing on the flattened butterfly NoC.

is 67% synapses, 18% neuron, and 15% router. The SRAM alone for the synapses contributes 56% of the chip area, 49% of the energy, 23% in dynamic energy, and 26% in leakage. The SNN places weights in a separate SRAM power domain to allow independent voltage scaling. Power gating unused

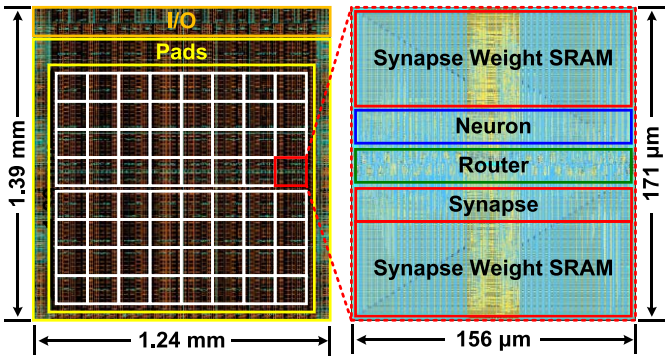


Fig. 16. 10-nm FinFET chip micrograph and core layout detail.

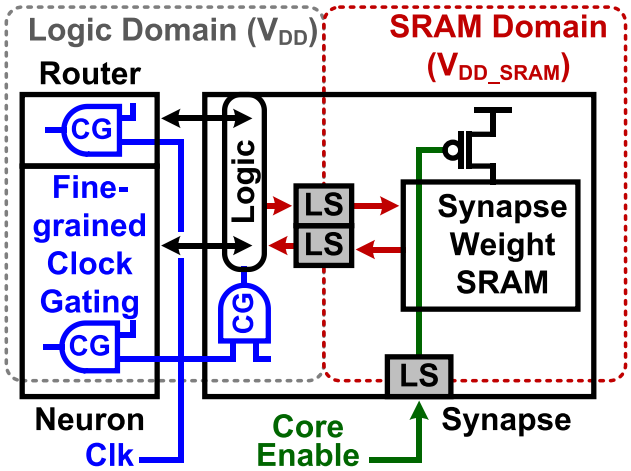


Fig. 17. Power and clock domains for SNN chip.

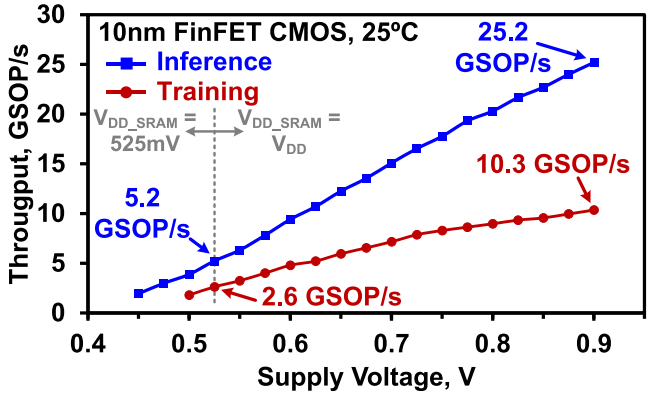


Fig. 18. Measured SNN throughput for inference and training.

weight cores reduces leakage by up to 2.7× (Fig. 17). Clock gating unused portions of neuron state reduces RF clock power by 2.4×. Fine-grained clock gating lowers datapath clock power by 35%.

The chip throughput (Fig. 18) and energy efficiency (Fig. 19) are measured as a function of the supply voltage for both training and inference. Each SOP delivers a spike through a unique synapse. Training uses both forward and back-spikes, but only forward spikes are counted as SOPs. At the energy optimal point of 525 mV, inference throughput reaches 5.2 GSOP/s at 3.8 pJ/SOP. Performance scales up

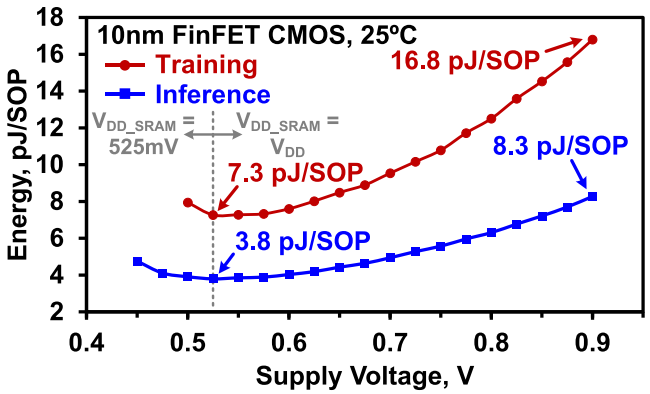


Fig. 19. Measured SNN energy efficiency for inference and training.

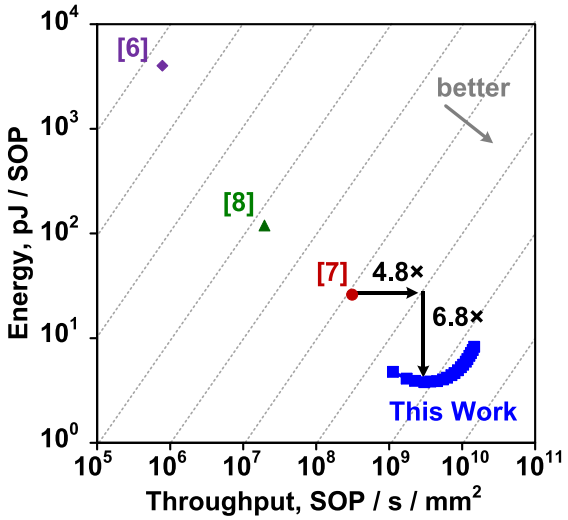


Fig. 20. Throughput and energy comparison with the previous SNNs.

TABLE I
COMPARISON WITH THE PREVIOUS SNNS

Project	This Work		[6]	[7]	[8]
Process	10nm		130nm	28nm	180nm
Area, mm^2	1.72		102	430	168
Synapse Bits	7b		-	1b	13b
Learning	Yes		Yes	No	No
Sparsity	Yes		Yes	No	No
Voltage	525mV	0.9V	1.2V	0.775V	1.8V
Freq., MHz	105	506	180	-	-
SOP/s/mm^2	3.0G	14.6G	784k	624M	19.6M
pJ/SOP	3.8	8.3	4000	26	119

to 25.2 GSOP/s at 0.9 V. Power scales down to 2.3 μW/neuron at the minimum operating voltage of 450 mV, with SRAM at 525 mV. Compared to the previous SNNs, this paper simultaneously achieves 4.8× higher throughput and 6.8× lower energy at 525 mV (Fig. 20) (Table I).

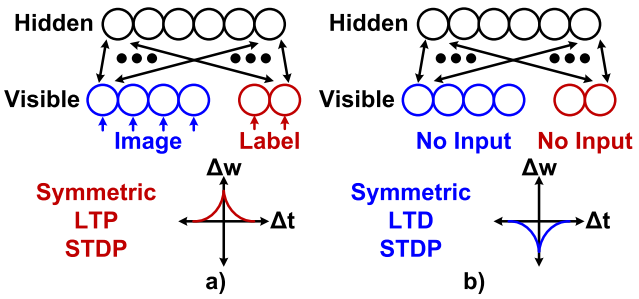


Fig. 21. Training spiking RBMs with unsupervised symmetric STDP. (a) Data phase. (b) Reconstruction phase.

A. RBM With Unsupervised STDP

A fully connected spiking RBM is mapped to the SNN chip and trained with unsupervised symmetric STDP for reconstruction tasks (Fig. 21). RBMs have undirected synapse connections between two layers of neurons, visible and hidden. The mapping of the RBM to the chip allows for a maximum size of 1024 × 1024 neurons. Visible and hidden neurons occupy 32 cores each and are mapped to the chip in a checkerboard pattern. Undirected connections are implemented by mapping spikes from both the visible and hidden layers to the same synapse addresses. Spikes from the visible neurons are implemented as forward spikes, and hidden spikes are backward so that the same synapses can communicate with two post-synaptic neuron cores as described in Section II-C.

Event-driven contrastive divergence training operates in two alternating phases [24]. In the data phase, pixel intensities from a natural scene image or Modified National Institute of Standards and Technology (MNIST) digits are input to visible-layer neurons as 8-b values and converted to spike rate on-chip using a Poisson spike generator. The spike patterns generated by this input are strengthened with symmetric LTP. In the reconstruction phase, the input is removed and spike patterns that do not relate to the training data are depressed. The RBM is trained on 8 × 8 patches from an original 1024 × 1024 pixel image, and the reconstructed image is back-projected from the hidden neurons with an RMSE of 0.036 (Fig. 22). The RBM forms a model of its expected inputs enabling de-noising. De-noising is helpful for real-world visual recognition problems with noise or occlusions. Anomalies are removed and missing components are filled in for test-set corrupted digits after training on 50 k training-set MNIST images (Fig. 23).

B. Feed-Forward Network With Supervised STDP

Supervised STDP is implemented using a reward signal to perform MNIST classification. Reward-modulated STDP uses the spike rate of a neuron or other supervisory signal to modulate the polarity of STDP weight updates (Fig. 24) [25]. Tying the reward to the desired output class maps input MNIST features to the correct digit. MNIST digits are pre-processed using 4 5 × 5 convolutional Gabor filters and then undergo 3 × 3 pooling. The resulting 8-b values are directly added into the membrane potential of the input neurons. On-chip, a 236 × 20 feed-forward SNN is mapped to 4 cores

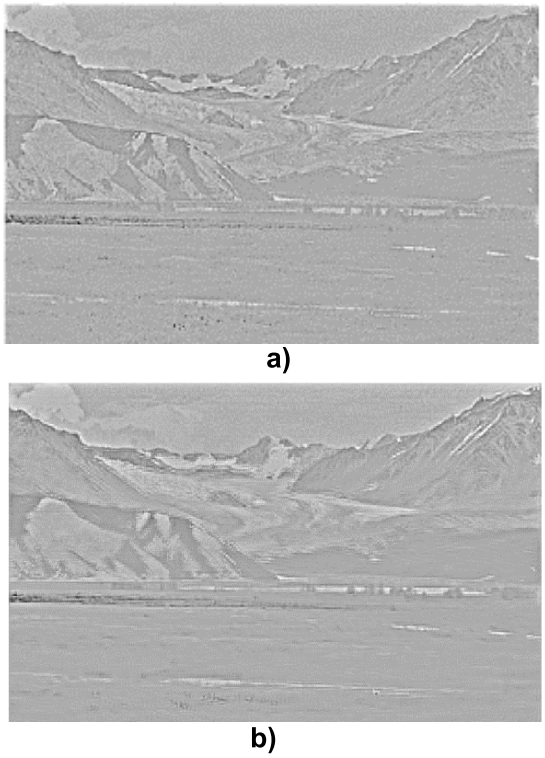


Fig. 22. Image reconstruction from 8 × 8 patches of original image. (a) Original image. (b) Reconstructed image, RMSE 0.036.

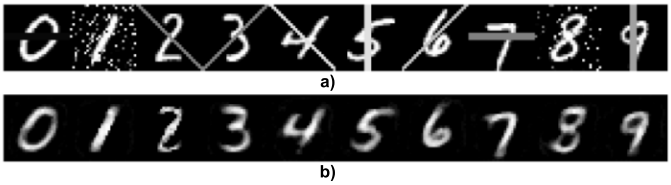


Fig. 23. De-noising of corrupted MNIST digits. (a) Corrupted images. (b) De-noised images.

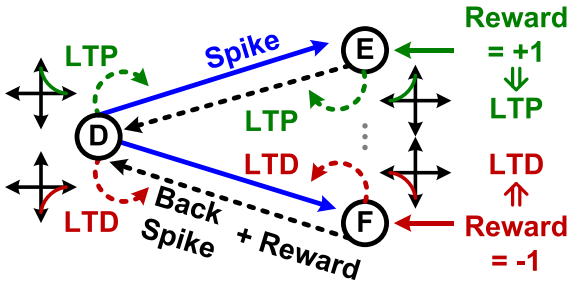
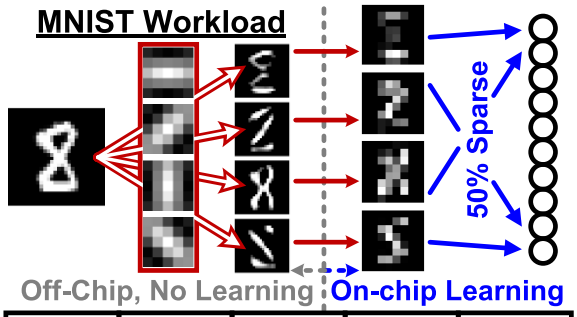


Fig. 24. Supervised training by modulating STDP polarity.

and classifies the digits with 89% accuracy for full connectivity and 88% accuracy with 50% weight sparsity (Fig. 25). Spiking operation to skip zero-valued activations, 50% sparse connectivity, power gating, and voltage scaling combine to reduce energy by 17.4× to 1.0-μJ/classification with a throughput of 6250 classifications/s (Fig. 26).

C. Binary Multilayer Perceptron With Deep Learning

A binary-activation multilayer perceptron (MLP) is trained to classify MNIST digits (Fig. 27). Binary inputs are generated



MNIST	Gabor	Pool	Input	Output
28x28	4@5×5	3x3	236	20

Fig. 25. MNIST classification with a 236 × 20 feed-forward SNN.

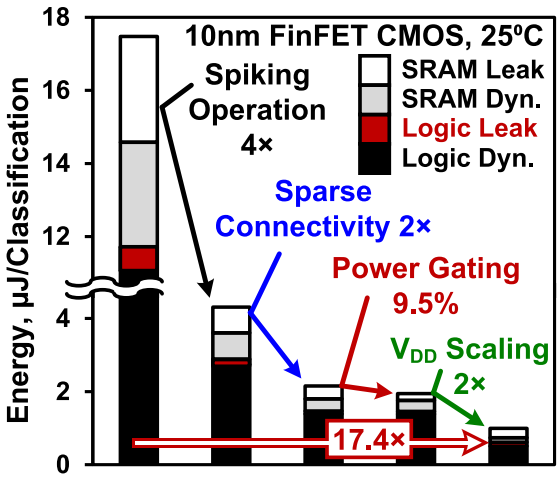


Fig. 26. Energy savings from approximate computing, sparse weights, power-gating unused weights, and supply voltage scaling.

by thresholding the MNIST pixel intensity. The network is trained offline using error-backpropagation, where the derivative of the binary-activation function is estimated with a straight through estimator (Fig. 28) [26], [27]. Sparsity is enforced throughout offline training by zeroing updates to sparsified weights representing no connection. Deep learning on a three-layer 784 × 1024 × 512 × 10 MLP with 7-b weights results in 98.60% accuracy for full connectivity and 98.56% accuracy when 50% of weights are removed at random and the network is re-trained from scratch. An MLP with 50% structured sparsity in all layers is mapped to the SNN chip. Connections between the neuron layers use 32, 16, and 2 synapse cores, respectively, which are physically mapped from left to right on the chip. The on-chip SNN achieves an accuracy of 98.15% with worst-case time step duration. Stochastic time step acceleration increases throughput by 8× and reduces energy by 7.3× to 1.7-μJ/classification, with a graceful accuracy degradation to 97.90% (Fig. 29). The approximate computing technique results in an activation sparsity of 80% and synapse drop rate of 23.5%. The MLP accuracy versus drop rate closely tracks ideal blank-out noise generated by multiplying the MLP weights with a randomly generated mask (Fig. 30).

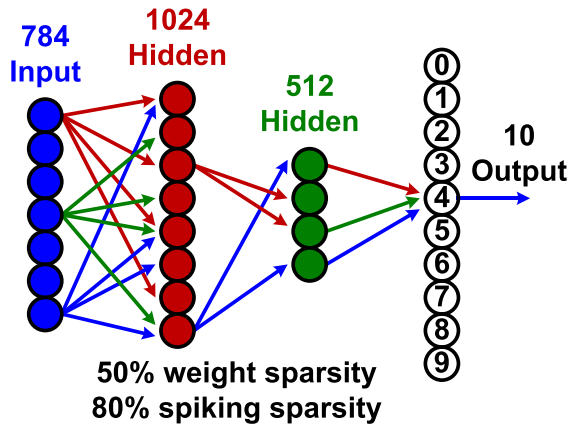


Fig. 27. Binary-activation MLP with 50% sparse weights.

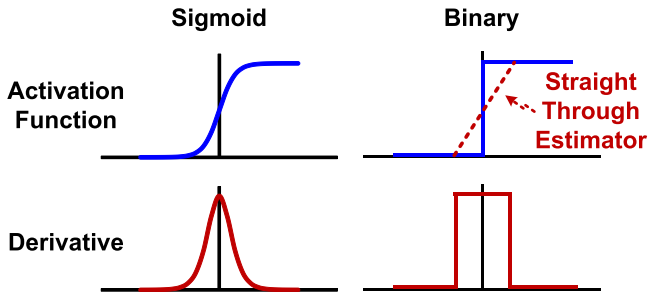


Fig. 28. Backpropagation with binary activations and derivative estimation.

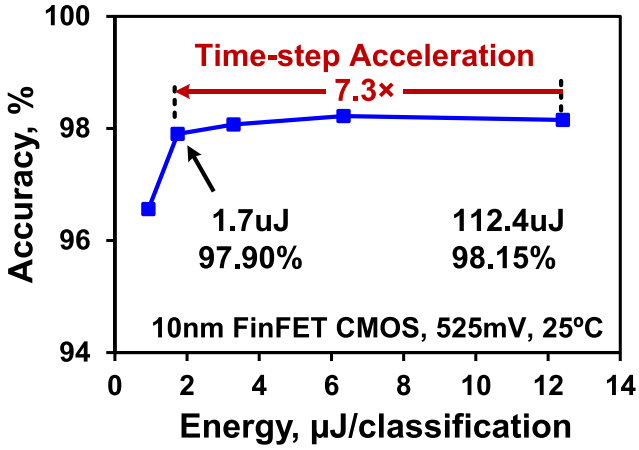


Fig. 29. Energy improvements with approximate computing.

IV. Discussion and Related Works

This project demonstrates advances in the rapidly evolving field of SNNs, and efforts continue to increase energy efficiency and improve learning with neuromorphic techniques. The two fundamental limiters to neural network scalability and energy efficiency are data storage and movement. SNNs mitigate some of these effects by communicating with 1-b spikes and using low-precision sparse weights. The SNN chip further reduces data movement using compute near memory, with persistent weights in each core and an option to integrate the weight values locally. Since each core is identical, a larger version of the chip can be fabricated by stamping out more cores and expanding of the NoC address space.

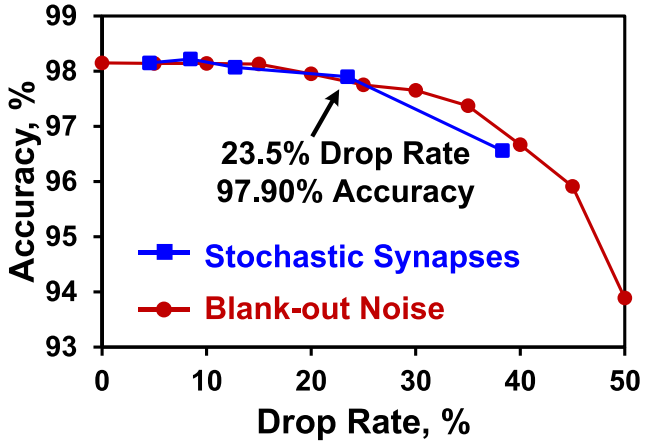


Fig. 30. Accuracy with stochastic synapses versus blank-out noise.

Additional techniques are required to map SNNs that exceed the memory capacity of the chip. Options include addition of DRAM to stream data to and from SRAM, or a dataflow architecture with communication between identical SNN chips [28].

Some of the same features that make SNNs energy efficient have been explored in deep neural networks (DNNs). In this paper, we discussed training DNNs with binary activations. Likewise, research is being conducted on binary or ternary weights which leverage low-precision computational units for higher energy efficiency [26], [29]. Another brain-like feature under study is weight sparsity for model size reduction. Energy for neural network operation is dominated by data movement, making model reduction critical for reducing off-chip energy [30]. Han *et al.* [12] prune DNN connections based on repeated training and removal of unimportant small-valued weights. Davies *et al.* [5] support sparsity in an SNN by storing synapses with target neuron indices. Several other works implement systolic arrays of multiply accumulate units to operate on matrices in compressed sparse row format [31], [32]. Mao *et al.* [13] constrain sparsity to the vector and kernel level for more regular operation requiring $\sim 2\times$ fewer memory references. Wen *et al.* [14] introduce group sparsity, which results in dense matrix operations after unrolling the convolution.

Improvements in the cognitive capabilities of SNNs will be critical for expanding the impact and practicality of neuromorphic computing. Some existing works enhance SNN capabilities with more accurate and detailed neuro-synaptic models and connectivity. In addition to pairwise STDP, Davies *et al.* [5] implement triplet STDP and show 96% MNIST classification accuracy using supervised STDP with temporally coded images. Brader *et al.* [33] report a supervised STDP rule that monitors calcium concentration to enable rate-based learning during periods of high spiking activity. Diehl and Cook [34] trains a winner-take-all network with lateral inhibition using unsupervised STDP.

Other proposals aim to connect the dots between the current understanding of neurobiology and recent advances in deep learning. Bengio *et al.* [35] formulate a bio-plausible version of error backpropagation based on a variant of STDP where

weight updates are proportional to the temporal derivative of the neuron membrane potential when a spike arrives. Bohté *et al.* [36] derive backpropagation for temporal coding with single spikes. Lee *et al.* [37] perform backpropagation directly on neuron by treating membrane potentials as differentiable signals in a deep SNN. Cao *et al.* [38] tailor CNN features for conversion to a rate-coded SNN with lower loss of accuracy.

V. Conclusion

This paper presents high-fan-out spiking communication and on-chip STDP in a scalable and reconfigurable 10-nm FinFET SNN. Near-threshold voltage circuit optimizations, power gating, and clock gating are employed to achieve 3.8-pJ/SOP energy efficiency. The chip demonstrates fine-grained structured sparsity to reduce weight memory by up to 16×, with less than 2% memory overhead for storing connections while achieving high workload accuracy. This implementation co-optimizes circuit bandwidth and algorithmic noise using a dropping flow control to accelerate time step computations and achieve unparalleled energy efficiency.

Acknowledgment

The authors would like to thank J. Tschanz, M. Haycock, M. Mayberry, M. Davies, N. Srinivasa, H. Wang, J. Held, A. Malavasi, S. Kale, S. Hsu, A. Agarwal, M. Anders, and H. Kaul for their valuable discussions and encouragement.

References

[1] C. D. James, "Toward exascale computing through neuromorphic approaches," Dept. Biosensors, Nanomater., Sandia Nat. Lab., Albuquerque, NM, USA, Tech. Rep. SAND2010-6312, 2010.

[2] T. M. Wong *et al.*, "10^{14}," IBM Res. Division, Almaden Res. Center, San Jose, CA, USA, Tech. Rep. RJ10502, 2012.

[3] C. Mead, "Neuromorphic electronic systems," *Proc. IEEE*, vol. 78, no. 10, pp. 1629–1636, Oct. 1990.

[4] W. Maass, "Networks of spiking neurons: The third generation of neural network models," *Neural Netw.*, vol. 10, no. 9, pp. 1659–1671, Dec. 1997.

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

[6] E. Painkras *et al.*, "SpiNNaker: A 1-W 18-core system-on-chip for massively-parallel neural network simulation," *IEEE J. Solid-State Circuits*, vol. 48, no. 8, pp. 1943–1953, Aug. 2013.

[7] P. A. Merolla *et al.*, "A million spiking-neuron integrated circuit with a scalable communication network and interface," *Science*, vol. 345, no. 6197, pp. 668–673, Aug. 2014.

[8] B. V. Benjamin *et al.*, "Neurogrid: A mixed-analog-digital multichip system for large-scale neural simulations," *Proc. IEEE*, vol. 102, no. 5, pp. 699–716, May 2014.

[9] P. Knag, J. K. Kim, T. Chen, and Z. Zhang, "A sparse coding neural network ASIC with on-chip learning for feature extraction and encoding," *IEEE J. Solid-State Circuits*, vol. 50, no. 4, pp. 1070–1079, Apr. 2015.

[10] J. M. Cruz-Albrecht, T. Derosier, and N. Srinivasa, "A scalable neural chip with synaptic electronics using CMOS integrated memristors," *Nanotechnology*, vol. 24, no. 38, p. 384011, Sep. 2013.

[11] J. Seo *et al.*, "A 45nm CMOS neuromorphic chip with a scalable architecture for learning in networks of spiking neurons," in *Proc. IEEE CICC*, Sep. 2011, pp. 1–4.

[12] S. Han, J. Pool, J. Tran, and W. Dally, "Learning both weights and connections for efficient neural network," *Proc. NIPS*, 2015, pp. 1135–1143.

[13] H. Mao *et al.* (Jun. 2017). "Exploring the regularity of sparse structure in convolutional neural networks." [Online]. Available: https://arxiv.org/abs/1705.08922.

[14] W. Wen, C. Wu, Y. Wang, Y. Chen, and H. Li, "Learning structured sparsity in deep neural networks," *Proc. NIPS*, 2016, pp. 2074–2082.

[15] W. Maass, "Motivation, theory, and applications of liquid state machines," in *Computability Context Comput. Logic Real World*, London, U.K.: Imperial College Press, 2011, pp. 275–296.

[16] M. Lukoševičius and H. Jaeger, "Reservoir computing approaches to recurrent neural network training," *Comput. Sci. Rev.*, vol. 3, no. 3, pp. 127–149, Aug. 2009.

[17] E. O. Neftci, B. U. Pedroni, S. Joshi, M. Al-Shedivat, and G. Cauwenberghs, "Stochastic synapses enable efficient brain-inspired learning machines," *Front. Neurosci.*, vol. 10, p. 241, Jan. 2016.

[18] L. Buesing, J. Bill, B. Nessler, and W. Maass, "Neural dynamics as sampling: A model for stochastic computation in recurrent networks of spiking neurons," *PLoS Comput. Biol.*, vol. 7, no. 11, p. e1002211, Nov. 2011.

[19] J. P. Lazzaro, J. Wawrzynek, M. Mahowald, M. Sivilotti, and D. Gillespie, "Silicon auditory processors as computer peripherals," *IEEE Trans. Neural Netw.*, vol. 4, no. 3, pp. 523–528, May 1993.

[20] S. Mathew *et al.*, "340 mV–1.1 V, 289 Gbps/W, 2090-gate NanoAES hardware accelerator with area-optimized encrypt/decrypt GF$(2^4)^2$ polynomials in 22 nm tri-gate CMOS," *IEEE J. Solid-State Circuits*, vol. 50, no. 4, pp. 1048–1058, Apr. 2015.

[21] S. Wolfram, "Random sequence generation by cellular automata," *Adv. Appl. Math.*, vol. 7, pp. 123–169, 1986.

[22] G. K. Chen, R. Kumar, H. E. Sumbul, P. C. Knag, and R. K. Krishnamurthy, "A 4096-neuron 1M-synapse 3.8PJ/SOP spiking neural network with on-chip STDP learning and sparse weights in 10 NM FinFET CMOS," in *Proc. IEEE Symp. VLSI Circuits*, Jun. 2018, pp. 225–256.

[23] C. Auth *et al.*, "A 10 nm high performance and low-power CMOS technology featuring 3rd generation FinFET transistors, self-aligned quad patterning, contact over active gate and cobalt local interconnects," in *IEDM Tech. Dig.*, Dec. 2017, pp. 29.1.1–29.1.4.

[24] E. Neftci, S. Das, B. Pedroni, K. Kreutz-Delgado, and G. Cauwenberghs, "Event-driven contrastive divergence for spiking neuromorphic systems," *Front. Neurosci.*, vol. 7, p. 272, Jan. 2014.

[25] R. Legenstein, D. Pecevski, and W. Maass, "Theoretical analysis of learning with reward-modulated spike-timing-dependent-plasticity," in *Proc. NIPS*, Dec. 2007, pp. 881–888.

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

[27] S. Yin *et al.*, "Algorithm and hardware design of discrete-time spiking neural networks based on back propagation with binary activations," in *Proc. IEEE Biomed. Circuits Syst. Conf.*, Oct. 2017, pp. 1–5.

[28] C. Nicol. (Aug. 2017). *A Dataflow Processing Chip for Training Deep Neural Networks*. [Online]. Available: https://www.hotchips.org/wp-content/uploads/hc_archives/hc29/HC29.22-Tuesday-Pub/HC29.22.60-NeuralNet1-Pub/HC29.22.610-Dataflow-Deep-Nicol-Wave-07012017.pdf

[29] M. Rastegari, V. Ordonez, J. Redmon, and A. Farhadi, "XNOR-Net: ImageNet classification using binary convolutional neural networks," *Eur. Conf. Comput. Vis.*, 2016, pp. 525–542.

[30] Y. 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.

[31] A. Parashar *et al.*, "SCNN: An accelerator for compressed-sparse convolutional neural networks," in *Proc. 44th Annu. Int. Symp. Comput. Archit.*, Jun 2017, pp. 27–40.

[32] M. Anders *et al.*, "2.9 TOPS/W reconfigurable dense/sparse matrix-multiply accelerator with unified INT8/INTI6/FP16 datapath in 14 NM tri-gate CMOS," in *Proc. IEEE VLSI*, Jun. 2018, pp. 39–40.

[33] J. M. Brader, W. Senn, and S. Fusi, "Learning real-world stimuli in a neural network with spike-driven synaptic dynamics," *Neural Comput.*, vol. 19, no. 11, pp. 2881–2912, Nov. 2007.

[34] P. U. Diehl and M. Cook, "Unsupervised learning of digit recognition using spike-timing-dependent plasticity," *Front. Comput. Neurosci.*, vol. 9, p. 99, Aug. 2015.

[35] Y. Bengio, D.-H. Lee, J. Bornschein, T. Mesnar, and Z. Lin. (Aug. 2016). "Towards biologically plausible deep learning." [Oline]. Available: https://arxiv.org/abs/1502.04156.

[36] S. M. Bohté, J. N. Kok, and J. A. Poutré, "Error-backpropagation in temporally encoded networks of spiking neurons," *Neurocomputing*, vol. 48, nos. 1–4, pp. 17–37, Oct. 2002.

[37] J. H. Lee, T. Delbruck, and M. Pfeiffer, "Training deep spiking neural networks using backpropagation," *Front. Neurosci.*, vol. 10, p. 508, Nov. 2016.
[38] Y. Cao, Y. Chen, and D. Khosla, "Spiking deep convolutional neural networks for energy-efficient object recognition," *Int. J. Comput. Vis.*, vol. 113, no. 1, pp. 54–66, May 2014.

Gregory K. Chen (M'06) received the B.S., M.S., and Ph.D. degrees in electrical engineering from the University of Michigan, Ann Arbor, MI, USA, in 2006, 2009, and 2011, respectively.

He is currently a Researcher with the Circuit Research Lab, Intel Corporation, Hillsboro, OR, USA. His research interests include neuromorphic computing, network on chips (NoCs), deep learning, and ultra-low-power computing.

Raghavan Kumar (M'13) received the M.S. and Ph.D. degrees in electrical engineering from the University of Massachusetts, Amherst, MA, USA, in 2012 and 2015, respectively.

He is currently a researcher with the Circuit Research Lab, Intel Corporation, Hillsboro, OR, USA. His research interests include hardware security, machine learning, and high-performance and low-power datapath circuits.

H. Ekin Sumbul (M'09) received the B.S. degree in electronics engineering from Sabanci University, Istanbul, Turkey, in 2010, and the Ph.D. degree in electrical and computer engineering from Carnegie Mellon University, Pittsburgh, PA, USA, in 2015.

He is currently a Research Scientist with the Circuit Research Lab, Intel Corporation, Hillsboro, OR, USA. His research interests include neuromorphic computing, neural networks and deep learning, and low-power and high-performance in-memory and near-memory processing systems.

Phil C. Knag (M'10) received the B.S. degree in computer engineering and the M.S. and Ph.D. degrees in electrical engineering from the University of Michigan, Ann Arbor, MI, USA, in 2010, 2012, and 2015, respectively.

He is currently a Researcher with the Circuit Research Lab, Intel Corporation, Hillsboro, OR, USA. His research interests include neuromorphic computing, machine learning, and energy efficient circuits.

Ram K. Krishnamurthy (M'98–SM'03–F'11) received the B.E. degree in electrical engineering from the National Institute of Technology, Trichy, India, in 1993, the M.S. degree in electrical and computer engineering from the State University of New York, Buffalo, NY, USA, in 1994, and the Ph.D. degree in electrical and computer engineering from Carnegie Mellon University, Pittsburgh, PA, USA, in 1997.

Since 1997, he has been with Intel Corporation, Hillsboro, OR, USA, where he is currently a Senior Principal Engineer with the Circuits Research Lab, where he also the heads the high-performance and low-voltage circuits research group. In this role, he leads research in high-performance, energy-efficient, and low-voltage circuits for microprocessors and system-on-a-chips (SoCs), and has made contributions to the circuit design of various generations of Intel products, including Intel Itanium, Pentium4, Core, Atom, and Xeon line of microprocessors and SoCs. He also serves as an Adjunct Faculty of the Electrical and Computer Engineering Department, Oregon State University, Corvallis, OR, USA, where he taught advanced very large scale integration (VLSI) design. He has authored over 200 papers and three book chapters on high-performance energy-efficient circuits. He has filed over 200 patents (125 issued).

Dr. Krishnamurthy is a Board Member of the Industry Advisory Board for the State University of New York. He was a recipient of the IEEE International Solid State Circuits Conference Distinguished Technical Paper Award, the IEEE European Solid-State Circuits Conference Best Paper Award, the Outstanding Industry Mentor Award from Semiconductor Research Corporation (SRC), the Intel awards for most patents filed and most patents issued, the Intel Labs Gordon Moore Award, the Alumni Recognition Award from Carnegie Mellon University, the Distinguished Alumni Award from the State University of New York, the MIT Technology Review's TR35 Innovator Award, recognized as a top ISSCC Paper Contributor, and the two Intel Achievement Awards for pioneering the first 64b Sparse-Tree ALU Technology and the first Advanced Encryption Standard Accelerator on Intel products. He serves as the Chair of the SRC Technical Advisory Board for circuits, a Guest Editor of the IEEE JOURNAL OF SOLID-STATE CIRCUITS, an Associate Editor of the IEEE TRANSACTIONS ON VLSI SYSTEMS, and the Technical Program Committees of ISSCC, CICC, and SOCC conferences. He served as the Technical Program Chair and the General Chair of the IEEE International Systems-on-Chip Conference and presently serves on the Conference's Steering Committee. He is a Distinguished Lecturer of the IEEE Solid-State Circuits Society.