Systolic-Array Spiking Neural Accelerators with Dynamic Heterogeneous Voltage Regulation

Jeong-Jun Lee
Electrical and Computer Engineering
University of California
Santa Barbara, CA, United States
jeong-jun@ucsb.edu

Jianhao Chen
Electrical and Computer Engineering
Texas A&M University
College Station, TX, United States
chenjh@tamu.edu

Wenrui Zhang
Electrical and Computer Engineering
University of California
Santa Barbara, CA, United States
wenruizhang@ucsb.edu

Peng Li
Electrical and Computer Engineering
University of California
Santa Barbara, CA, United States
lip@ucsb.edu

***Abstract*—Spiking neural networks (SNNs) have emerged as a new generation of neural networks, presenting a brain-inspired event-driven model with advantages in spatiotemporal information processing. Due to the need for high power consumption of compute-intensive neural accelerators, adequate power delivery network (PDN) design is a key requirement to ensure power efficiency and integrity. However, PDN design for SNN accelerators has not been extensively studied despite its great potential benefit in energy efficiency.**

In this paper, we present the first study on dynamic heterogeneous voltage regulation (HVR) for spiking neural accelerators to maximize system energy efficiency while ensuring power integrity. We propose a novel sparse-workload-aware dynamic PDN control policy, which enables high energy efficiency of sparse spiking computation on a systolic array. By exploring sparse inputs and all-or-none nature of spiking computations for PDN control, we explore different types of PDNs to accelerate spiking convolutional neural networks (S-CNNs) trained with the dynamic vision sensor (DVS) gesture dataset. Furthermore, we demonstrate various power gating schemes to further optimize the proposed PDN architecture, which leads to a more than a three-fold reduction in total energy overhead for spiking neural computations on systolic array-based accelerators.

***Index Terms*—spiking neural networks, power delivery networks, systolic arrays, voltage regulation**

I. Introduction

As a brain-inspired computing model, spiking neural networks (SNNs) have emerged as a new generation of neural network models [5], [16], [17], [21], [30]. The all-or-none nature, event-driven operation, and inherent spatiotemporal characteristics of SNNs support a variety of network models, including convolutional neural networks, which have been widely adopted in vision-oriented tasks. Especially, spiking-CNNs, or simply S-CNNs, are well-suited for processing complex spatiotemporal data such as image recognition, pattern recognition, and gesture recognition, with their inherent temporal aspect.

Despite strong demand and potential, hardware acceleration of spiking computation has been proposed by only a few previous works, such as [2], [9], [27], [28] and neuromorphic chips including IBM's TrueNorth [3] and Intel's Loihi [12]. Furthermore, the aforementioned works offer an architecture for general spiking models rather than a specific dataflow or mapping strategy of given tasks.

While the basic accelerator architecture is one key research focus for energy-efficient spiking neural computation acceleration, design of efficient power delivery networks (PDNs) is also a major challenge for an optimization of the overall system energy efficiency [18], [22], [31]. Huge power loss due to power delivery failure may lead to a significant degradation in overall energy efficiency, and cause timing error due to decreased power integrity. For large-scale systems, including multi-core processors and system-on-ship (SoC), various voltage regulation schemes in the PDN benefit the energy efficiency and power delivery integrity of the system in supporting computationally-intensive workloads [7]. While there exist a few works to incorporate optimized power delivery in the PDN for computationally-intensive neural networks [7], [25], [29], little work has been devoted to SNN architectures. More specifically, PDN design optimization for SNNs, considering the all-or-none nature of spiking activities and inherent spatiotemporal characteristics, has not been explored.

A recent SNN architecture work proposed holistic reconfigurable dataflow optimization for S-CNNs, using variable tiling, specifically focusing on the positioning of the temporal dimension. [20] proposed a temporal parallel processing method developed for spiking computation on the systolic array where the time dimension is mapped in parallel for time-batched computation, sidestepping temporally sequential operation in conventional SNNs. By efficiently handling partial sums and filters with structured mapping methodology, [20]

This material is based upon work supported by the National Science Foundation under Grant No. 2000851 and No. 1948201. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

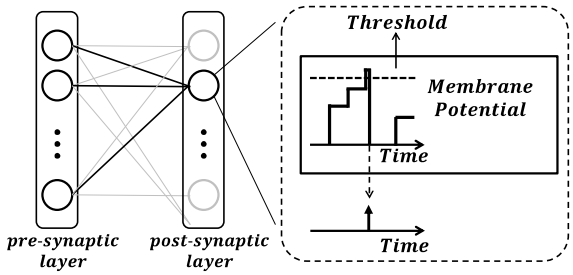

Fig. 1. Schematic representation of spiking computation.

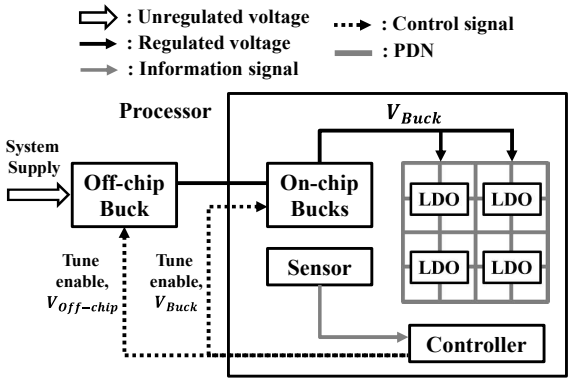

Fig. 2. Schematic representation of three-stage hybrid voltage regulation architecture.

achieved orders of magnitude of performance improvement for accelerating deep Alexnet and VGG-16.

We are motivated by this approach, i.e., temporal parallel processing, which enables independent processing between columns and coordinates with the power domain control. By extending the temporal parallel processing concept with predetermined sparse input information and heterogeneous voltage regulation (HVR) schemes, we propose a HVR based sparse-workload-aware PDN architecture for spiking neural computation on systolic array-based SNN accelerators.

The main contributions of this work are:

- We present HVR based PDN architecture and control policy for spiking neural computation on systolic array accelerators, which applies to S-CNN models.
- We investigate how the dataflow, which enables temporal parallel processing, coordinates with power gating based on the sparse nature of the spiking models, and propose a sparse-workload-aware dynamic PDN control policy.
- We show how the PDN architecture, especially the two-/three-stage network, dynamic PDN control policy, and various gating schemes reduce the energy overhead of the SNN accelerators.

This work presents the first work on HVR based PDN architecture and control for spiking neural computation on systolic array-based accelerators. We demonstrate various gating schemes with different PDN architectures to boost energy efficiency for a real-world application with the dynamic vision sensor (DVS) gesture dataset [4]. The proposed techniques deliver more than a three-fold reduction in total energy overhead.

II. Background

A. Spiking Neural Computation

Apart from traditional neural network models, SNNs offer a viable solution to the deployment of energy-efficient hardware design with two unique aspects.

1) Data representation: All data types in non-spiking, conventional networks, are multi-bit data. As in [10], [11], [13], [15], [24], 8 to 16-bit precision is the most frequently applied data representation for all types of data, i.e., input feature map (IFmap), filter and output feature map (OFmap). On the other hand, SNNs communicate via binary-valued input and output spikes as shown in Fig. 1.

2) Temporal processing: Temporal processing in spiking models is another key unique aspect. SNNs explicitly models the input into all-or-none binary values across multiple time points. As shown in Fig. 1, a spiking neuron integrates presynaptic spike inputs and conditionally generates spike output only if the membrane potential exceeds firing threshold of a neuron. Depending on leaky behavior, we classify the neuron as integrate-and-fire (IF) [6] or leaky integrate-and-fire (LIF) neuron model [14], which are the two most widely adopted neuron models.

B. Systolic Arrays

Systolic arrays have been a widely-adopted choice with their advantages in high compute density, low hardware complexity, and simple data distribution with neighbor-to-neighbor connections. 2-D systolic array offers spatial-locality and data reuse in both horizontal and vertical directions, i.e., top-to-bottom and left-to-right, lend themselves as an efficient architecture model for processing compute-intensive tasks such as neural network accelerators [19], [23], [26].

C. Spiking Neural Computations on a Systolic Array

Mapping complex spatiotemporal spiking neural computations on the systolic array are non-trivial with its unique nature discussed in Section II-A. [20] proposed holistic reconfigurable dataflow optimization for S-CNNs, especially focusing on the added time dimension. Among various dataflow types, temporal parallel processing method to perform computation across multiple time points concurrently suggests an energy-efficient dataflow strategy while overcoming the limitation of stereotyped temporally-sequential computation in spiking models.

Based upon [20], we follow the dataflow with temporal parallel processing as shown in Fig. 3, where **O**, **I** and **W** are the matrices of the output feature maps (OFmaps), input feature maps (IFmaps), and filters, respectively. U is a given stride size, and f is a spike generation function. We optimize the power delivery network upon aforementioned dataflow as will be discussed in later section.

Algorithm 1: Pseudocode for mapping S-CNNs

Input: IFMAP: $\mathbf{I}[T][C][H][H]$, FILTER: $\mathbf{W}[M][C][R][R]$
Output: OFMAP: $\mathbf{O}[T][M][E][E]$

```
/* For all OFMAP pixels                                              */
for x, y=0; x, y<E; x++, y++ do
    /* For all Output channels and Timesteps                         */
    for m̄=0; m̄ < ⌈M/A_h⌉ do
        for t̄=0; t̄ < ⌈T/A_w⌉ do
            /* For all Input channels                                */
            for c=0; c<C; c++ do
                Parallel/* Array operation                           */
                for t, m=0; t, m< A_w, A_h, t++, m++ do
                    /* For all elements of unrolled filter           */
                    for i, j=0; i, j<R; i++, j++ do
                        /* Partial sum                               */
                        P[t+t̄A_w][m+m̄A_h][x][y][i][j][c] =
                        I[t+t̄A_w][c][Ux+i][Uy+j] ×W[m+m̄A_h][c][i][j]
                        /* Synaptic input integration                */
                        P[t][m][x][y] += P[t][m][x][y][i][j][c]
                    end
                end
            end
            /* Membrane potential update                             */
            P'[t][m][x][y] = P[t][m][x][y] + P_F[t − 1][m][x][y]
            /* Conditional spike generation                          */
            O[t][m][x][y] = f(P'[t][m][x][y])
            if O[t][n][m][x][y] = 1 then
                P_F[t][n][m][x][y] = 0                    // Reset
            end
            else
                P_F[t][n][m][x][y] = P'[t][n][m][x][y]
            end
        end
    end
end
```

H: IFmap width / height, E: OFmap width / height,
R: Filter width / height, C: Number of IFmap channels,
M: Number of OFmap channels, T: Number of time steps,
A_w, A_h: Array width / height
$0 \le x, y < E,\ E = (H - R + U)/U,\ 0 \le c < C,\ 0 \le m < M,\ 0 \le t < T,\ 0 \le n < N$

Fig. 3. Pseudo algorithm of the temporal parallel processing dataflow.

D. Voltage Regulation for Neural Network

Power delivery network (PDN) is becoming an increasingly complex task to match stringent power requirement of power supply to on-chip devices [29]. To ensure an adequate supply of power with efficiency and integrity, various types of PDN architectures have been explored. For example, [25], [29] proposed heterogeneous voltage regulation (HVR) with three-stage PDN which consists of off-chip buck converters, on-ship buck converters and low-dropout VRs (LDOs). Compared to conventional two-stage PDNs, three-stage PDN better supports workload aware power management of machine learning tasks and reduces energy dissipation with rich heterogeneity and tunability, as shown in Fig. 2.

With rapidly growing applications, neural network accelerators require a large amount of power to support computationally-intensive data processing. Thus, power delivery is one of the major challenges for a variety of processor system designs, including server-class, multi-core processors and systems-on-chips (SoCs). [7] proposed dynamic power delivery network (PDN) control policy for systolic array deep neural network (DNN) accelerators. Recognizing the need for efficiency and integrity of power delivery to DNN accelerators, [7] proposed an optimized PDN architecture with dynamic control and power gating method.

We adopt and extend the three-stage PDN architecture for spiking computations in this paper. Importantly to note that, we apply sparse-workload information and fine-grained power control with various gating schemes, which is not considered in [7].

III. Sparse-workload-aware Voltage Regulation

Mapping the spiking computations on a systolic array with pre-determined variable tiling strategy, or simply dataflow has significant impacts on energy dissipation but less extensively explored [20]. Furthermore, there exists a dearth of comprehensive approaches that capture the exact interactions in data movement/computation and introduces application-specific optimization for given tasks. Apart from the energy dissipation of the accelerator, another key aspect of the system's overall energy efficiency is power delivery. As discussed previously, PDN design offers potential benefits in power saving for computationally-intensive workloads such as neural network accelerators.

Recognizing the potential advantage to employ PDN design optimization for spiking neural computation, we propose a method to jointly coordinate the dataflow in the spiking neural accelerator with the power control policy. Our key observation is that the spiking activities in SNNs can be interpreted as a sparse matrix, which opens up the opportunity for an efficient PDN control aligned for the systolic array.

A. Sparsity in SNNs

As introduced in the previous section, SNNs models the input into binary values across multiple time points. As a result, handling sparsity is a key challenge for real-world applications in SNNs considering binary input matrix. Fig. 5 shows a normalized average firing rate of the neurons in the first convolution layer (CONV1) for DVS gesture dataset using 300 timesteps. On average, most of the neurons remain silent while the number of neurons exponentially decreases as the firing rate increases. For example, there exist only 6.04 neurons out of 65,536 neurons on average, which fires 150 times across 300 timesteps, for 1,463 input samples.

B. Proposed PDN architecture

Recognizing the need for handling sparsity, we propose a sparse-workload-aware, dynamic PDN control policy for S-CNNs. As shown in Fig. 3, we follow the dataflow in [20], which enables temporal parallel processing and simplified control of columns in an array for a targeted OFmap pixel. While this dataflow specifies data scheduling via variable tiling, we propose in-depth PDN control policy optimization considering data sparsity, as shown in Fig. 4.

According to Fig. 3 and Fig. 4, the systolic array computes a partial result of a specific pixel in OFmap. For a given pixel of OFmap, the array conducts parallel processing both in the horizontal and vertical axis, i.e., through multiple time

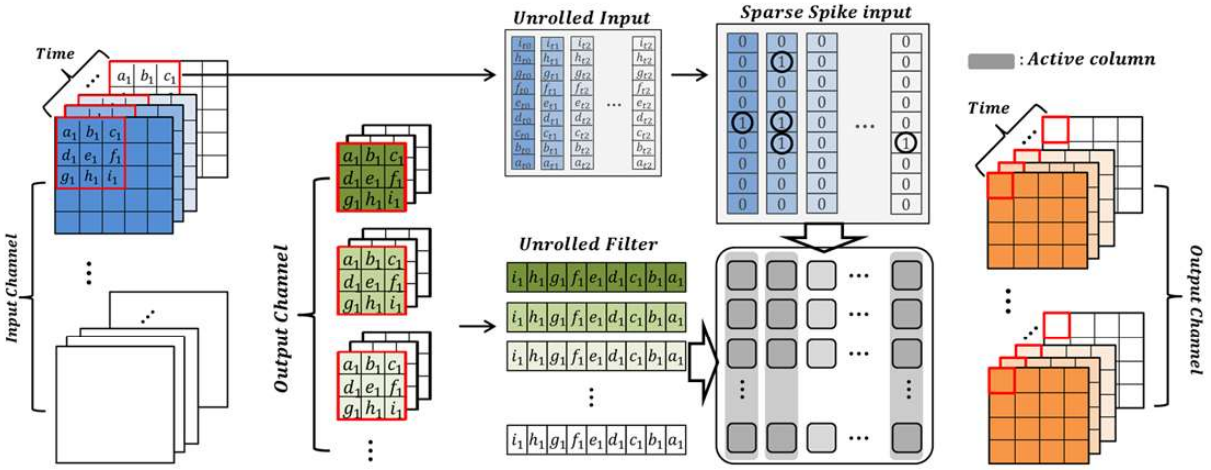

Fig. 4. Schematic representation of the temporal parallel processing dataflow and sparse IFmap.

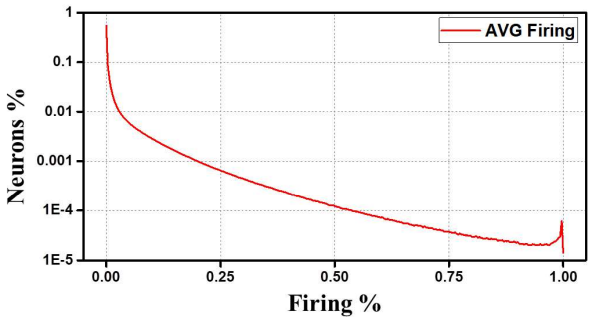

Fig. 5. Normalized average firing activity rates of the neurons in first convolution layer of DVS gesture dataset. The percentage of neurons and firing rates are normalized to the total number of neurons, and the total number of timesteps, respectively.

points (row-wise), and across different channels (column-wise). Importantly, this dataflow enables to compute each column independently as partial sums are not required from other columns (time points). We can separate the computation between different time points, for the most computationally-intensive step, i.e., feedforward synaptic input integration step. This opens up the opportunity to separate computations across columns, which enables the formation of independent power domains to support few grouped columns in a systolic array. More specifically, columns without nonzero input elements become inactive throughout the array iteration, as shown in Fig. 4. In most cases of array iteration, power savings can be brought by sparse nature of spiking activities as discussed in the earlier section.

However, blindly employing power shut down for columns may lead to an inaccurate outcome. Important to note that, columns that are inactive for a specific array iteration and columns that are unused for entire array iterations should be classified. If we power-off the inactive columns at a specific array iteration, membrane potential values are not maintained and hence causes wrong result for input integration step at the next iteration. It requires the power network to have more fine-grained control for ensuring minimum power supply to maintain the membrane potential.

To tackle the above issue, we propose various control policies in different types of PDN architecture, which will be discussed in the following section.

IV. Evaluation Methodology

To support various PDN architecture and power domain control with gating schemes, we consider a 65nm 100MHz systolic array accelerator with an array size of 256×256 as identical to Google's TPU [1]. Also, we adopt CACTI [8] for the evaluation of the energy dissipation which is widely-adopted, an architecture level power modeling framework [7], [20], [23], [26]. We specify the power control characterization of the accelerator-based upon sparse data for DVS gesture inference.

A. PDN Architectures

We power the accelerator with two different PDN architectures, i.e., two-stage PDN and three-stage PDN, with and without dynamic control. The two-stage PDN architecture consists of off-chip buck converters and on-chip buck converters. Compared to a single-stage PDN architecture, the two-stage PDN improves the quality of power delivery and supports a multicore system much faster [29]. However, the response time of on-chip buck converters may still limit the PDN performance, which is tackled by three-stage PDN architecture. The three-stage PDN architecture includes an additional network of distributed on-chip LDO, which enables to drive power domains.

Dynamic control policy adjusts active voltage regulators dynamically where static control fixes the number of active voltage regulators. With or without using dynamic control policy, we combine the two different PDN architecture with control policy and compare four different types of PDN:

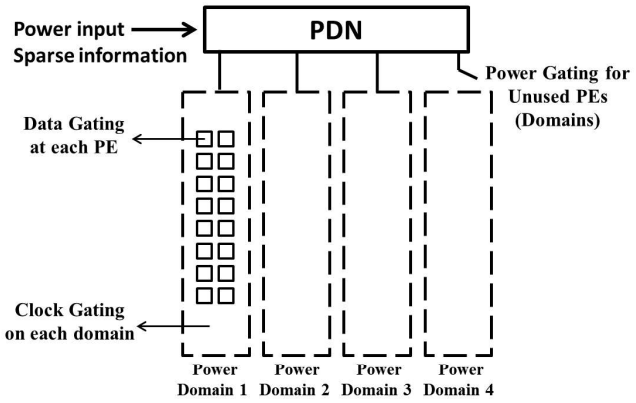

Fig. 6. An overview of PDN architecture and gating strategies applied for dynamic control policy.

TABLE I
AN OVERVIEW OF THE GATING TECHNIQUES IN PDN ARCHITECTURES.

PDN Gating	Description
Data gating	Skip computation in each PE, if the input is zero
Clock gating	Use minimum power level to maintain scratch pad memory to store membrane potential. When inputs for a specific column are all zeros, PDN lowers the power level for a corresponding power domain while computation is not performed
Power gating	Power-off the unused power domains, which is mainly due to unmatched dimension between array size and input variable

2-stage static, 2-stage dynamic, 3-stage static and 3-stage dynamic.

B. Dynamic Control Policy with Gating Strategy

We apply three different gating strategies for dynamic control policy of PDN as summarized in Table I.

1) Data gating: As presented in [11], we can significantly reduce energy dissipation by skipping the computation in each PE whenever the input is zero. For a computationally-intensive neural network application, data gating presents a high-efficient performance for energy reduction at each processing unit level.

2) Clock gating: During each systolic array processing iteration, there exist active columns to perform a computation for non-zero inputs as shown in Fig. 4. However, the remaining columns only require a minimum power level to maintain the membrane potential in its local memory, i.e., scratchpad memory. Fortunately, predictability from the pre-calculated sparse information of workload ensures successful control of each clock domain.

3) Power gating: Mapping variables into a systolic array results in dimension mismatch. Whenever the dimension is less than the width or height of the array, corresponding columns or rows of the array remains unused, which leads to incorporate power gating strategy. For example, mapping the time variable with 300 timesteps to 256×256 introduce the remaining 44 columns after the first array iteration.

TABLE II
AN OVERVIEW OF THE APPLIED GATING SCHEMES.

Gating scheme	Description
Scheme *A*	Data gating only.
Scheme *B*	Data gating + clock gating.
Scheme *C*	Data gating + clock gating + power gating.

TABLE III
AN OVERVIEW OF THE NETWORK STRUCTURE.

Layer	Number of channels	Number of neurons
CONV1	64	32X32
CONV2	128	32X32
CONV3	256	16X16

We demonstrate the results of different PDN architectures based on the three gating schemes as shown in II, among various combinations of gating strategies.

C. DVS gesture dataset

We present the result based on a practical DVS gesture which is a dataset for real-time gesture recognition [4]. The dataset consists of 1463 input samples of 11 different hand gestures. We adopt the most-widely adopted rate-based coding scheme and convert each input sample into a 300 timestep binary matrix. This work is primarily focused on convolution layers in network structure as summarized in Table III. Based on a trained network, we employ 1,463 input samples to evaluate the energy consumption of systolic array system in different PDN architectures with various gating schemes. Note that, we utilize the spiking activities for DVS gesture inference to manage the proposed gating schemes.

V. RESULTS

We perform a comprehensive evaluation of energy efficiency in different PDN architectures and gating schemes based upon the setups described in Section IV. We break down the PDN architecture energy consumption into five different energy overhead sources. Throughout this section, we denote those sources, i.e., LDO regulators, processor, the off-chip buck converters, the package, and the on-chip buck converters, as ldo, processor, off-buck, pkg and on-buck, respectively.

A. Dynamic control policy with single power domain

In Fig. 7, four different PDN types are denoted as 2-dynamic, 2-static, 3-dynamic and 3-static where each PDN represents two-stage PDN with and without dynamic control, and three-stage PDN with and without dynamic control, respectively. The results are normalized to energy dissipation of two-stage PDN with static control at each convolution layer. In general, three-stage PDNs enables fine-grained control of on-chip buck converters since three-stage PDNs further incorporates additional stage of distributed LDOs. However, this requires additional energy consumption for LDOs where there exists a clear benefit considering the energy consumption of on-chip buck converters. By leveraging the dynamic adaptation of the sparse workloads, dynamic control policy presents better

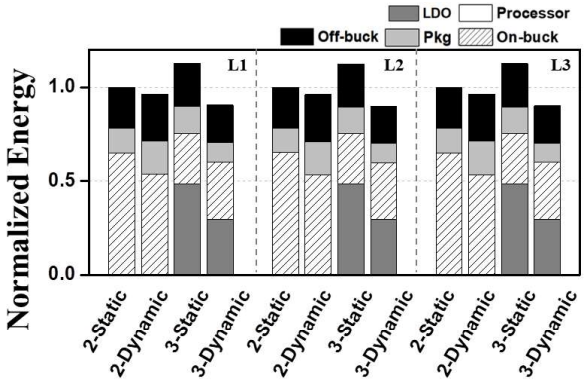

Fig. 7. Normalized energy consumption of different PDNs with a single power domain in three convolution layers. Four different PDN types are described in Section V-A.

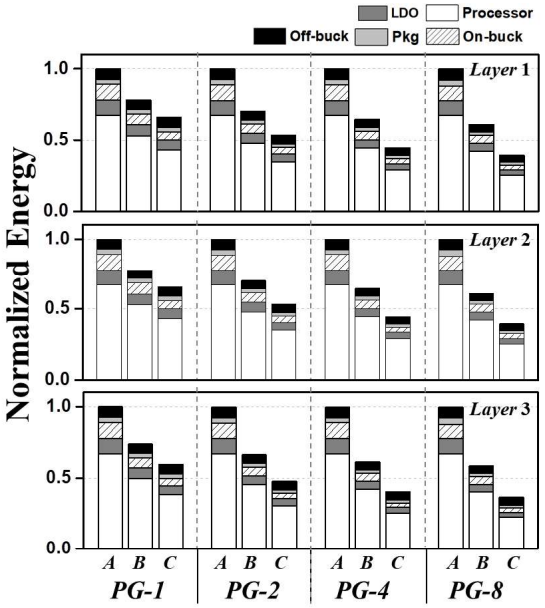

Fig. 8. Normalized energy consumption of 3-stage PDNs with different numbers of power domain in three convolution layers. Gating scheme A, B and C follows the combinations of gating techniques represented in Table II, respectively.

performance for both two-stage and three-stage PDNs, as shown in Fig. 7. As in Fig. 7, our key observation is that the three-stage PDN with dynamic control policy improves the energy consumption in all three convolution layers, on average for 1,463 input samples of the DVS gesture dataset.

B. Gating scheme with multiple power domains

We demonstrate further benefits brought by the proposed gating schemes in Table II. We compare different gating schemes based upon the most efficient PDN architecture, three-stage PDN with dynamic control policy, as previously discussed. Furthermore, we explore the granularity of each gating schemes, in terms of the number of power domains. We denote the PDN using a number of i power domains as PG-i in Fig. 8.

For each layer in Fig. 8, the results are normalized to the energy dissipation of three-stage, single power-domain PDN with gating scheme *A*. We observe clear benefits from schemes A to C, as inactive columns originated from zero-inputs are optimized with clock gating, and unused power domains due to mapping mismatch between variable and array size, are blocked by power gating. More advanced from simple data gating, which skips trivial computation for zero input at PE level [11], the proposed gating scheme dynamically adopts the sparsity incurred in spiking neural computation and thus efficiently manages leakage power.

In all three convolution layers, the proposed gating scheme reduces overall energy consumption, in particular with the energy savings in processor, on-chip buck converters and LDOs. Among PDNs with different gating schemes and the number of power domains, the proposed power gating technique can achieve more than a three-fold reduction in energy dissipation.

VI. Conclusion

This work is motivated by a dearth of an efficient power delivery network to support SNN accelerators. Based on previously proposed dataflow which enables temporal parallel processing of S-CNNs on systolic array, we recognize the presence of sparsity in spiking computations and introduce a sparse-workload-aware PDN architecture, which employs gating control with respect to the input information.

We demonstrate how the various PDN architectures and gating schemes can be applied on systolic array accelerator to improve energy efficiency. The proposed techniques delivers more than a three-fold reduction in total energy overhead for accelerating dynamic vision sensor (DVS) gesture dataset.

Acknowledgment

This material is based upon work supported by the National Science Foundation under Grant No. 2000851 and No. 1948201. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

[1] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard *et al.*, "Tensorflow: A system for large-scale machine learning," in *12th {USENIX} symposium on operating systems design and implementation ({OSDI} 16)*, 2016, pp. 265–283.

[2] A. Afifi, A. Ayatollahi, and F. Raissi, "Implementation of biologically plausible spiking neural network models on the memristor crossbar-based cmos/nano circuits," in *2009 European Conference on Circuit Theory and Design*. IEEE, 2009, pp. 563–566.

[3] F. Akopyan, J. Sawada, A. Cassidy, R. Alvarez-Icaza, J. Arthur, P. Merolla, N. Imam, Y. Nakamura, P. Datta, G.-J. Nam *et al.*, "Truenorth: Design and tool flow of a 65 mw 1 million neuron programmable neurosynaptic chip," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 34, no. 10, pp. 1537–1557, 2015.

[4] A. Amir, B. Taba, D. Berg, T. Melano, J. McKinstry, C. Di Nolfo, T. Nayak, A. Andreopoulos, G. Garreau, M. Mendoza *et al.*, "A low power, fully event-based gesture recognition system," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 7243–7252.

[5] G. Bellec, D. Salaj, A. Subramoney, R. Legenstein, and W. Maass, "Long short-term memory and learning-to-learn in networks of spiking neurons," in *Advances in Neural Information Processing Systems*, 2018, pp. 787–797.

[6] A. N. Burkitt, "A review of the integrate-and-fire neuron model: I. homogeneous synaptic input," *Biological cybernetics*, vol. 95, no. 1, pp. 1–19, 2006.

[7] J. Chen, J. Riad, E. Sánchez-Sinencio, and P. Li, "Dynamic heterogeneous voltage regulation for systolic array-based dnn accelerators," in *2020 IEEE 38th International Conference on Computer Design (ICCD)*. IEEE, 2020, pp. 486–493.

[8] K. Chen, S. Li, N. Muralimanohar, J. H. Ahn, J. B. Brockman, and N. P. Jouppi, "Cacti-3dd: Architecture-level modeling for 3d die-stacked dram main memory," in *2012 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2012, pp. 33–38.

[9] L. Chen, C. Li, T. Huang, Y. Chen, and X. Wang, "Memristor crossbar-based unsupervised image learning," *Neural Computing and Applications*, vol. 25, no. 2, pp. 393–400, 2014.

[10] T. Chen, Z. Du, N. Sun, J. Wang, C. Wu, Y. Chen, and O. Temam, "Diannao: A small-footprint high-throughput accelerator for ubiquitous machine-learning," in *Proceedings of the 19th International Conference on Architectural Support for Programming Languages and Operating Systems*, ser. ASPLOS '14. New York, NY, USA: Association for Computing Machinery, 2014, p. 269–284. [Online]. Available: https://doi.org/10.1145/2541940.2541967

[11] Y.-H. Chen, T. Krishna, J. S. Emer, and V. Sze, "Eyeriss: An energy-efficient reconfigurable accelerator for deep convolutional neural networks," *IEEE journal of solid-state circuits*, vol. 52, no. 1, pp. 127–138, 2016.

[12] M. Davies, N. Srinivasa, T.-H. Lin, G. Chinya, Y. Cao, S. H. Choday, G. Dimou, P. Joshi, N. Imam, S. Jain *et al.*, "Loihi: A neuromorphic manycore processor with on-chip learning," *IEEE Micro*, vol. 38, no. 1, pp. 82–99, 2018.

[13] Z. Du, R. Fasthuber, T. Chen, P. Ienne, L. Li, T. Luo, X. Feng, Y. Chen, and O. Temam, "Shidiannao: Shifting vision processing closer to the sensor," in *2015 ACM/IEEE 42nd Annual International Symposium on Computer Architecture (ISCA)*, 2015, pp. 92–104.

[14] W. Gerstner and W. M. Kistler, *Spiking neuron models: Single neurons, populations, plasticity*. Cambridge university press, 2002.

[15] V. Gokhale, J. Jin, A. Dundar, B. Martini, and E. Culurciello, "A 240 g-ops/s mobile coprocessor for deep neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2014, pp. 682–687.

[16] Y. Jin, Y. Liu, and P. Li, "Sso-lsm: A sparse and self-organizing architecture for liquid state machine based neural processors," in *2016 IEEE/ACM International Symposium on Nanoscale Architectures (NANOARCH)*. IEEE, 2016, pp. 55–60.

[17] Y. Jin, W. Zhang, and P. Li, "Hybrid macro/micro level backpropagation for training deep spiking neural networks," *arXiv preprint arXiv:1805.07866*, 2018.

[18] W. Kim, M. S. Gupta, G.-Y. Wei, and D. Brooks, "System level analysis of fast, per-core dvfs using on-chip switching regulators," in *2008 IEEE 14th International Symposium on High Performance Computer Architecture*. IEEE, 2008, pp. 123–134.

[19] H. Kung, B. McDanel, and S. Q. Zhang, "Packing sparse convolutional neural networks for efficient systolic array implementations: Column combining under joint optimization," in *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems*, 2019, pp. 821–834.

[20] J.-J. Lee and P. Li, "Reconfigurable dataflow optimization for spatiotemporal spiking neural computation on systolic array accelerators," in *2020 IEEE 38th International Conference on Computer Design (ICCD)*. IEEE, 2020, pp. 57–64.

[21] J. H. Lee, T. Delbruck, and M. Pfeiffer, "Training deep spiking neural networks using backpropagation," *Frontiers in neuroscience*, vol. 10, p. 508, 2016.

[22] H. Li, J. Xu, Z. Wang, P. Yang, R. K. Maeda, and Z. Tian, "Adaptive power delivery system management for many-core processors with on/off-chip voltage regulators," in *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2017*. IEEE, 2017, pp. 1265–1268.

[23] S. Lym and M. Erez, "Flexsa: Flexible systolic array architecture for efficient pruned dnn model training," *arXiv preprint arXiv:2004.13027*, 2020.

[24] M. Peemen, A. A. Setio, B. Mesman, and H. Corporaal, "Memory-centric accelerator design for convolutional neural networks," in *2013 IEEE 31st International Conference on Computer Design (ICCD)*. IEEE, 2013, pp. 13–19.

[25] J. Riad, J. Chen, E. Sánchez-Sinencio, and P. Li, "Variation-aware heterogeneous voltage regulation for multi-core systems-on-a-chip with on-chip machine learning," in *2020 21st International Symposium on Quality Electronic Design (ISQED)*. IEEE, 2020, pp. 190–194.

[26] A. Samajdar, Y. Zhu, P. Whatmough, M. Mattina, and T. Krishna, "Scale-sim: Systolic cnn accelerator simulator," *arXiv preprint arXiv:1811.02883*, 2018.

[27] S.-Q. Wang, L. Wang, Y. Deng, Z.-J. Yang, S.-S. Guo, Z.-Y. Kang, Y.-F. Guo, and W.-X. Xu, "Sies: A novel implementation of spiking convolutional neural network inference engine on field-programmable gate array," *Journal of Computer Science and Technology*, vol. 35, pp. 475–489, 2020.

[28] X. Wu, Y. Wang, H. Tang, and R. Yan, "A structure–time parallel implementation of spike-based deep learning," *Neural Networks*, vol. 113, pp. 72–78, 2019.

[29] X. Zhan, J. Chen, E. Sánchez-Sinencio, and P. Li, "Power management for multicore processors via heterogeneous voltage regulation and machine learning enabled adaptation," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 27, no. 11, pp. 2641–2654, 2019.

[30] W. Zhang and P. Li, "Spike-train level backpropagation for training deep recurrent spiking neural networks," in *Advances in Neural Information Processing Systems*, 2019, pp. 7802–7813.

[31] V. Zyuban, J. Friedrich, D. M. Dreps, J. Pille, D. W. Plass, P. J. Restle, Z. T. Deniz, M. M. Ziegler, S. Chu, S. Islam *et al.*, "Ibm power8 circuit design and energy optimization," *IBM Journal of Research and Development*, vol. 59, no. 1, pp. 9–1, 2015.