Power-efficient Spiking Neuromorphic Designs using CMOS and Emerging Devices

by

Ziru Li

Department of Electrical and Computer Engineering
Duke University

Defense Date: April 3, 2024

Approved:

Hai Li, Supervisor

James Morizio

Lisa Wills

Shimeng Yu

Yiran Chen

Dissertation submitted in partial fulfillment of the requirements for the degree of
Doctor of Philosophy in the Department of Electrical and Computer Engineering
in the Graduate School of Duke University
2024

ABSTRACT

Power-efficient Spiking Neuromorphic Designs using CMOS and Emerging Devices

by

Ziru Li

Department of Electrical and Computer Engineering
Duke University

Defense Date: April 3, 2024

Approved:

Hai Li, Supervisor

James Morizio

Lisa Wills

Shimeng Yu

Yiran Chen

An abstract of a dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the Department of Electrical and Computer Engineering
in the Graduate School of Duke University
2024

Abstract

The artificial intelligence (AI) algorithms have played critical roles in a variety of application scenarios in our daily life. The size of state-of-the-art large-scale AI models widely adopted in different domains have been proliferating to tens of billions of parameters [4]. The dedicated AI hardware tailored for data-intensive and computation-intensive AI algorithms consume tremendous power due to data transmission of model parameters and massive computation. The solutions to boosting the power efficiency of AI hardware are two-fold. On the one hand, continuous research efforts have been paid to search for more efficient computing paradigms for neural networks. For instance, the bio-inspired neuromorphic computing paradigm stems from the investigation of the natural neural system. The neuromorphic spiking-neural-networks (SNNs) emulate the human brain which transmits information efficiently through spike events. On the other hand, hardware designers have been seeking architecture- and circuit-level solutions to reducing the memory access and computation costs. Processing-in-memory (PIM) paradigm, which is one of the promising solutions, eliminates the power and latency of data transmission by performing data operations directly within the memory.

In this dissertation, my research work on power-efficient neuromorphic designs will be introduced. These neuromorphic designs harness the spike-based data processing and in-memory-computing paradigm. With the help of architecture-level techniques and dedicated circuits with CMOS and emerging memory devices, the proposed designs achieve significant improvement in terms of power efficiency and performance.

Contents

List of Tables

List of Figures

Acknowledgements

First and foremost I would like to express my deepest gratitude to Professor Hai (Helen) Li and her co-PI of the lab Professor Yiran Chen, who mentored me in the past 5 years, for their invaluable guidance and support during my PhD program. I would like to express my gratitude to the members of my dissertation committee, Prof. James Morizio, Prof. Lisa Wu Wills, and Prof. Shimeng Yu, for their insightful comments and suggestions.

Furthermore, I am grateful to my collaborators, including Bonan, Qilin, Brady, Shiyu, Yitu, Bing, and my lab mates, including Qing, Linghao, Zhiyao, Fan, Xiaoxuan, Ximing, Huanrui, Ang, Edward and Jingyang. They have inspired and assisted me with their insightful ideas and precious advice.

Last but not least, I owe an immense debt of gratitude to my parents for their unwavering love, understanding, and encouragement throughout this academic journey. Their unwavering support has been the cornerstone of my success.

1. Introduction

Neuromorphic hardware platforms have shown great potential in efficiently performing data-intensive and computation-intensive machine learning tasks. These hardware designs implement the brain-inspired spiking-neural-networks (SNNs), which are regarded as the third-generation neural networks [39]. Different from the conventional frame-driven deep-neural-networks (DNNs), SNNs introduce the time dimension to the computation. In SNNs, the data transmitted between different layers of neurons are in the form of spike events through time, and the computation occurs during the generation and reception of spike events. This event-driven feature enables SNNs to hold the potential of achieving lower power while implemented by the dedicated hardware.

1.1 Spiking-neural-networks

SNNs mimic the human neural systems with a set of synapse and neuron abstractions that incorporate various time-based encoding schemes into their dynamic models. For example, rate-coded SNNs use the firing rate of spike trains to represent the strength of data transmission between neurons. A larger number of spikes indicates stronger activation. Based on the generation and measurement of rate-coded spike trains, this encoding scheme can be classified into count-based coding, population-based, Poisson rate coding [2], etc. In the SNN abstraction, the rate-coding scheme requires a relatively large amount of spike events and long duration of the time window to maintain sufficient precision, since the information is lost when averaged among the intervals between multiple spike events [68]. Alternatively, in the temporal-coding scheme, the information is represented by the firing times of spike events. The temporal-coding scheme significantly reduces the required number of spikes and overall time duration, thus speeding up the computation and saving energy. For instance, an extremely simple temporal-coding scheme called time-to-first-spike (TTFS) [56] encodes the datum to the firing time of a single spike within the time window. An earlier-fired spike represents a stronger activation. The detailed inference process of a TTFS-based SNN is formulated in Section 2.1.

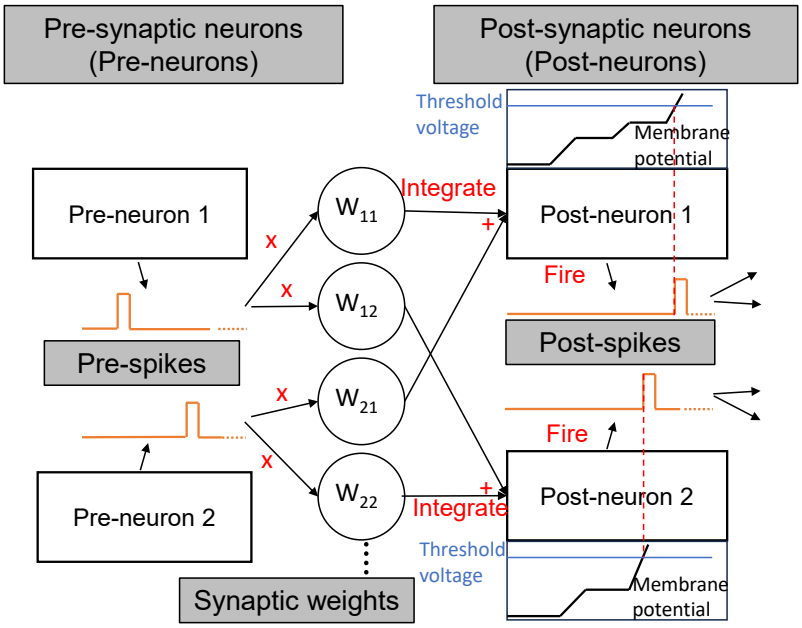


FIGURE 1.1: A Time-to-first-spike SNN with integrate-and-fire neurons.

Besides various encoding schemes, the SNN abstraction also includes different models to mimic the neuron dynamics. The integrate-and-fire (IF) model is one of the classic neuron models. When pre-synaptic neurons transmit spikes to a post-synaptic neuron, the spikes are first multiplied by the synaptic weights. Then, the post-synaptic neuron integrates the modulated spikes on its membrane potential. When the membrane potential reaches the threshold voltage, the post-synaptic neuron fires a spike and resets its membrane potential to the resting voltage. Thus, the neuron membrane is modeled as a constant capacitor that integrates the voltage from input spiking currents. FIGURE 1.1 shows an example of a TTFS-based SNN with IF neurons. Based on the standard IF model, a leaky IF model takes the leakage of the membrane capacitor into account by adding a leakage current to the model. The leaky IF model can be formulated as:

$$I(t) = C_m \frac{dV_m(t)}{dt} + \frac{V_m(t) - V_{rest}}{R_m} \quad (V_m(t) < V_{th}), \tag{1.1}$$

where $V_m(t)$ is the membrane potential, C_m is the capacitance of the neuron membrane, R_m is the resistance on the leakage path, V_{rest} is the resting voltage and V_{th} is the threshold voltage. Variants of the IF model include IF model with adaption [24], integrate-and-fire-or-

burst model [52], resonate-and-fire model [23], quadratic IF neuron [12], etc. Furthermore, there exist more complex neuron models such as the well-known Hodgkin–Huxley (HH) model [19]. HH model is formed by 4 equations and tens of parameters, which is much more complicated than the IF-based neuron models. Therefore, the IF model is widely adopted in SNN hardware for the sake of hardware implementation simplicity. For example, the IF neuron can be easily implemented by an analog integrate-and-fire circuit (IFC) [37].

1.2 CMOS neuromorphic hardware platforms

Built with different data encoding schemes, the spiking neuromorphic systems, i.e., the specific hardware that emulates SNN models, can improve energy-efficiency and computation parallelism by leveraging asynchronous circuits that work in the event-driven manner. These designs can thus eliminate power-hungry global clock networks [1]. Among CMOS-based practices, digital neuromorphic systems, such as GoldenGate [41], TrueNorth [1], SpiNNaker [16] and Loihi [9], have a good programmability to support multiple encoding schemes, including rate-coding and temporal-coding. However, they approximate the continuous neuron dynamics and spike propagation in discrete time steps, which is not biological plausible and degrades the inference accuracy. Mixed-signal neuromorphic systems such as BrainScaleS [47] and Neurogrid [3] are built with analog neuron circuits. Power-hungry digital-to-analog/analog-to-digital converters (DACs & ADCs) are required as the interface between the SRAM-based synapses and analog neuron circuits. Confronted with the limitations of those neuromorphic systems using conventional digital and analog CMOS technologies, researchers introduce new technologies and architectures to vastly boost the computational efficiency.

1.3 PIM-based spiking neuromorphic designs

Emerging memory devices like resistive random-access-memory (ReRAM) with the small cell area and low power consumption shows advantages in neuromorphic hardware implementations. Moreover, the synapse-alike behavior of ReRAM enables the analog synapse, freeing the digital/analog interface between neurons and synapses in the mixed-

signal neuromorphic systems. The implementation of both the unsupervised spike-timing-dependency-plasticity (STDP) and supervised learning rules with ReRAM devices have been investigated [53, 20, 29, 51]. However, there lacks concrete circuit implementation of a power-efficient ReRAM-based neuromorphic hardware for SNN inference.

The ReRAM technology provides a PIM solution to the power efficient neuromorphic hardware. In a typical ReRAM-based PIM spiking neuromorphic hardware design [37, 66], ReRAM cells are organized in a crossbar manner. The synapse weights are programmed to the conductance of the ReRAM cells, while the input data are fed into the horizontal word-lines of the ReRAM crossbar in the form of wordline voltages. The output modulated data are generated along the vertical bitlines in the form of bitline currents, which represent the multiply-and-accumulate (MAC) results between input voltages and ReRAM cell conductance. ReRAM crossbars act as both data storage and processing elements, eliminating the power consumption of weight data transmission and conversion. The analog computation principle also boosts the energy efficiency of MAC computation. Furthermore, the crossbar structure enables the input voltages to be shared by multiple ReRAM cells along the same wordline, increasing the computation parallelism and benefiting the overall throughput. As a result, the PIM-based spiking neuromorphic design become a strong candidate of power-efficient AI hardware.

Despite the high potential of PIM in boosting the energy efficiency, there is a lack of concrete circuit designs of PIM-based spiking neuromorphic hardware for supporting the inference of SNNs, especially temporal-coded SNNs. Besides the overall energy efficiency and performance, the inference robustness to the circuit process variations is another key criteria that need to be considered. The process variations of emerging memory devices inject noises to the conductance, inducing erroneous activation strength and firing time of post-synaptic spikes, and thus, degrading the inference accuracy [7]. This demands for techniques to mitigate the impact of process variations and improve the inference robustness of the PIM-based spiking neuromorphic design.

In order to maximize the efficiency gain of the emerging neuromorphic computing

paradigm, in this report, we propose a series of cross-layer solutions of PIM-based spiking neuromorphic designs using CMOS and emerging devices. To be more specific, the main research works presented in this report are as follows:

1. Chapter 3 introduces ReSiPE [31], a ReRAM-based single-spiking PIM processing engine tailored for the conventional frame-driven DNN inference. ReSiPE harnesses the concept of TTFS to minimize the computation duration and boost the energy efficiency.
2. Chapter 4 introduces ASTERS [32], a ReRAM-based neuromorphic design to conduct the Temporal-coded SNN inference. In addition to the PIM-based processing engine design, we also propose two techniques, including "Multi-Level Firing Threshold Adjustment" to mitigate the impact of ReRAM process variations, and “Timing Threshold Adjustment” to further speed up the computation.
3. Chapter 5 introduces SpikeSen [34], a low-latency in-sensor-intelligence neuromorphic vision system using CMOS-based neuromorphic spiking neurons. SpikeSen processes the photocurrents from sensors locally and executes the computation in the frequency domain to reduce the data transmission and conversion and speed up the computation.
4. Chapter 6 presents TFSRAM, an SRAM-based PIM neuromorphic chip for TTFS-based SNN inference in 65nm technology node.

2. Background

In this chapter, the preliminary knowledge of software and hardware substrates of the presented research works is provided.

2.1 Temporal-coded SNNs

As briefly introduced in Section 1.1, temporal-coded SNNs encode the activation strength to the firing time of spikes. Given a TTFS-based SNN depicted in FIGURE 1.1, the inference process is as follows:

$$C_{m,j}\frac{dV_{m,j}(t)}{dt} = \Sigma_{i=0}^{N-1} w_{ij}\delta(t-t_i) \quad (V_{m,j}(t) < V_{th}) \tag{2.1}$$

where w_{ij} is the weight value of the synaptic connection between the $i-th$ pre-synaptic neuron and the $j-th$ post-synaptic neuron, N is the total number of neurons that have synaptic connections with the $j-th$ post-synaptic neuron, $V_{m,j}(t)$ and $C_{m,j}$ are the membrane potential and membrane capacitance of the $j-th$ post-synaptic neuron, t_i is the spike time of the $i-th$ pre-synaptic neuron. The input activation strength is encoded as t_i. The output activation strength is represented by the spike time of the post-synaptic neuron t_j, which is the time when $V_{m,j}(t_j) = V_{th}$. After the post-synaptic spike is fired at $t = t_j$, $V_{m,j}(t)$ is reset to the resting voltage.

An exponentially-decaying spike waveform is more bio-plausible than the binary spike δ(t-t_i) [70]. Substituting the pre-synaptic spike in Equation 2.2 with an exponentially-decaying waveform, the inference process becomes:

$$\begin{aligned} C_{m,j}\frac{dV_{m,j}(t)}{dt} &= \Sigma_{i=0}^{N-1} w_{ij}g(t) \quad (V_{m,j}(t) < V_{th}) \\ g(t) &= \begin{cases} 0 & 0 \leqslant t < t_i \\ 1 - exp(-\frac{t-t_i}{\tau_s}) & t \geqslant t_i \end{cases} \end{aligned} \tag{2.2}$$

where τ_s is the decaying time constant of spike. There are two observations from this equation. First, an earlier pre-synaptic spike with spike time t_i can contribute to the membrane potential increase of the $j-th$ post-synaptic neuron if $t_i <= t_j$ in a TTFS-based

SNN. The later pre-synaptic spikes that arrive after t_j have no impact on the output post-synaptic spike time t_j. Second, the neuron will enter the idle state after it fires the spike ($t > t_j$). These unique features in a TTFS-based SNN can be leveraged to further reduce the unnecessary power consumption and computation latency in an SNN hardware.

2.2 Prior PIM-based spiking neuromorphic designs

In this section, several representative PIM-based spiking neuromorphic designs are introduced. These designs include both emerging ReRAM-based PIM designs and CMOS-based PIM designs using SRAM.

Wang *et al.* [58] introduced a ReRAM-based SNN system for real time classification. Their neuron circuit added a load resistor and a current mirror to the bitline output, which may elevate the bitline voltage when the bitline current increases and thus introduce non-linearity to the product of pre-synaptic spikes and synapses. Besides, they did not provide the method of mapping the inhibitory synapses to the ReRAM crossbar. The synaptic strength in a spiking neural network can be positive or negative, which determines whether the pre-spike should be excited or inhibited. The negative inhibitory synapses are necessary because they enable the network to differentiate its response to various input patterns [22].

Wijesinghe *et al.* [60] introduced a system-level implementation of SNNs with stochastic neurons and performed the simulation with IBM 45nm technology node. However, its synapse mapping scheme to map the negative inhibitory synapses involved negative voltage input and required additional circuits to reverse the pre-synaptic spikes, leading to extra hardware overhead.

Yan *et al.* [66] proposed a ReRAM-based process engine that leverages rate-coded spike computation in 150nm technology node. The neuron circuit in this design contains a current amplifier to hold a stable bitline voltage and increase the linearity of MAC computation. However, the current amplifiers consistently consume mW power during the entire inference time window, leading to unnecessary energy consumption during the idle state of the neurons.

Kim *et al.* [28] proposed an SRAM-based neuromorphic processor to support rate-coded SNNs in 28nm technology node. The chip stores 4-bit/8-bit synaptic weights in the 8T-SRAM arrays, takes in input spikes along the read wordlines and accumulates the membrane potential from the weighted spikes along the read bitlines. The multi-bit integration is implemented by capacitor-DAC adders which incur large area overhead.

3. ReRAM-based Single-Spiking Processing-In-Memory Engine for DNN Inference

PIM designs that leverage emerging nanotechnologies like ReRAM have demonstrated enormous potential in accelerating deep learning applications due to high energy efficiency and integration density. The common approach of existing ReRAM-based PIM designs is to encode data into either voltage levels with assist of power-thirsty analog/digital conversion circuits, or spike series by sacrificing computing latency. In this chapter, we introduce ReSiPE, a ReRAM-based Single-spiking PIM Engine for DNN inference, which uses the arrival time of a single spike to represent a data. This encoding scheme is inspired by the TTFS encoding scheme in SNNs. We analyze how to encode data into a set of single spikes and develop circuit to realize the matrix-based computation. The proposed design can minimize the spike numbers, shorten the computation period, and thus dramatically improve the energy efficiency. Our simulation results show that ReSiPE achieves 67.1% power reduction and 1.97× power efficiency improvement compared to rate-coding-based ReRAM PIM designs at comparable area, throughput and accuracy.[1]

3.1 Introduction

PIM is regarded as an effective solution for deploying DNNs. In conventional ReRAM-based PIM paradigms, the elements of a matrix are mapped to the conductance values of the corresponding cells in a ReRAM crossbar. When supplying an input vector to the wordlines, the signals along bitlines form the computed output vector. Such a multiple-input multiple-output (MIMO) operation realizes the MVM computation [21]. A common implementation is the level-based design [48], which represents the input data in the form of analog voltages and collects the bitline currents/voltages as output. Such designs usually require complex digital/analog and analog/digital converters (DAC and ADC) to facilitate the signal transition, resulting in high area and energy overhead. Moreover, the level-based design need constantly supply the input signals during the entire MVM computation period,

[1] Ziru Li, Bonan Yan, and Hai Li. "ReSiPE: ReRAM-based single-spiking processing-in-memory engine". In: 2020 57th ACM/IEEE Design Automation Conference (DAC). IEEE. 2020, pp. 1–6. © 2020 IEEE.

incurring high power consumption.

Alternatively, the rate-coding based designs [37, 54, 66] encode data into a series of spikes and use the frequency of spike series to record the data strength. The peripheral circuitry in these designs can be simplified, presenting advantages in area and energy efficiency. However, the rate-coding based designs suffer from the quantization errors and thus usually prolong the computing period for ensuring satisfactory performance (e.g., classification accuracy).

The data format schemes of both level-based designs and rate-coding based designs are not energy-friendly. When values of inputs increase, the power consumption will accordingly increase due to either the rise of voltage amplitude in level-based designs or the increase of spike numbers in rate-coding based designs. The key insight to fix this disadvantage is to decouple the power consumption with data representation. Instead of adopting spike series in [66, 37], we propose *single-spiking data format*, which uses only a single spike for a datum and denotes its value with the time duration from the beginning of the period to the arrival of the spike. ReRAM crossbar does not consume much power except during the short computation period.

Furthermore, to support the MVM computation based on the single-spiking data format, on ReRAM crossbar structure, we introduce ReSiPE, a ReRAM-based Single-spiking PIM Engine. The key contributions are summarized as follows:

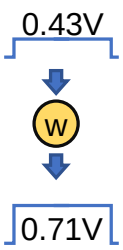
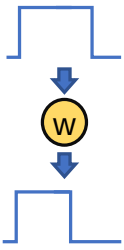
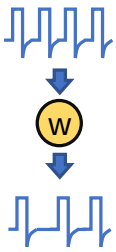
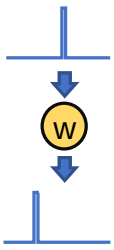
1. We recognize the energy bottleneck of ReRAM-based PIMs and propose to explore the unique single-spiking data format in PIM design.
2. We design the circuitry to support single-spiking multiply-accumulate (MAC) scheme and analyze its feasibility. We then generalize the approach to MVM operation at architectural level and propose ReSiPE.
3. We evaluate the efficiency of ReSiPE in terms of power consumption, throughput, area and accuracy based on several typical neural network benchmarks. Our analysis shows that ReSiPE can achieve 67.1% power reduction and 1.97× power efficiency improvement compared to the state-of-the-art rate-coding based ReRAM PIM designs.

The rest of the chapter is organized as follows. Section 3.2 summarizes state-of-the-art ReRAM-based PIM designs based on various data formats. We describe the single-spiking data format and our proposed ReRAM-based single-spiking PIM engine in Section 3.3. Section 3.4 elaborates detailed evaluation results of ReSiPE. At the end, we conclulde this chapter in Section 3.5.

3.2 Related works

As summarized in TABLE 3.1, we categorize the existing ReRAM-based PIM designs for DNNs into three classes based on the data forms.

Table 3.1: Various Data Formats in ReRAM PIM Designs

Data Format	Level	PWM	Spike		
			Rate Coding	Temporal Coding	This work
Shape					
Interface Circuit	DAC & ADC	Pulse Modulator	Spike Modulator	Neuron Circuit	ReSiPE
Non-zero Voltage Applying Duration	Long	Medium	Medium	Medium	Short
In/Out Scale	Same	Same	Different	Same	Same
Latency	Fast	Medium	Medium	Slow	Medium

Level-based designs: Data are modulated as voltage potentials and fed to ReRAM array [8, 21]. Power- and area-hungry interface DAC and ADC circuits are necessary in this type of processing engines to convert input signals and produce output data. Inputs are supplied to ReRAM crossbars and fully occupy the entire computation period.

PWM-based design: Jiang *et al.* [25] proposed to encode information into the time-domain pulse width by using pulse width modulation (PWM), in order to better utilize the computation period. The work still requires ADC to generate output data.

Spike-based design: Prior spike-based ReRAM PIM designs usually adopt *temporal coding* or *rate coding*. Temporal coding refers to the relative spike timing between pre- and post-synapses. Special-shaped spikes are carefully designed to fulfill certain functions, e.g. spike-timing-dependent plasticity (STDP). So the peripheral circuitry design usually is complicated. The enriched functionality provided by the temporal coding can largely reduce the power consumption but result in very long latency to accurately emulate neural-alike dynamics [46]. As a compromise, rate coding scheme encodes data into the spike frequency [37, 66, 54]. The circuit is hence largely simplified. However, repetitive spike generation hinders further enhancement of energy efficiency.

3.3 The ReSiPE Design

In this work, we aim at a novel ReRAM-based PIM design for higher power efficiency, area and performance. More specific, our proposed ReSiPE is based on the single-spiking data format and leverage the timing information for data representation and processing.

3.3.1 Single-spiking Data Format

FIGURE 3.1 depicts the signal relations of two sequential layers when applying the single-spiking data format. The MVM computation in each layer is accomplished in two stages with identical latency. Each stage is regarded as a *full-scale time slice*, or a *slice* in the following context.

As illustrated, an input of layer n is given in the first *slice* (S1). The duration from the beginning of S1 to the rising edge of the spike (t_n) is counted as the input value. At the end of S1, the computation stage kicks in with a short period of δt, during which the ReRAM crossbar will perform the MVM computation. The output of layer n will be generated in the second *slice* (S2), which can be directly used as the input of its subsequent layer $n+1$. In this way, the operation across different layers can be realized in pipeline form.

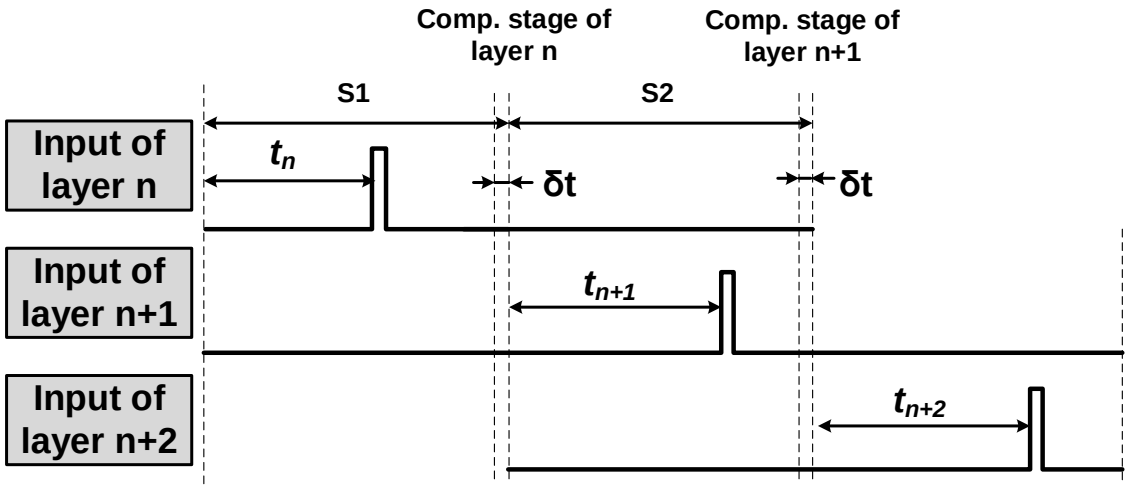


FIGURE 3.1: An illustration of the single-spiking data format.

The proposed single-spiking data format has a few advantages. First, the input is represented by the arrival time of the spike, independent of spike width and shape. Second, the MVM computation is carried out in the short computation stage, consequently lowering the energy consumption incurred by ReRAM crossbar. In addition, the identical format of input and output signals simplifies the peripheral circuity design, removing power-thirsty readout circuits and ADCs.

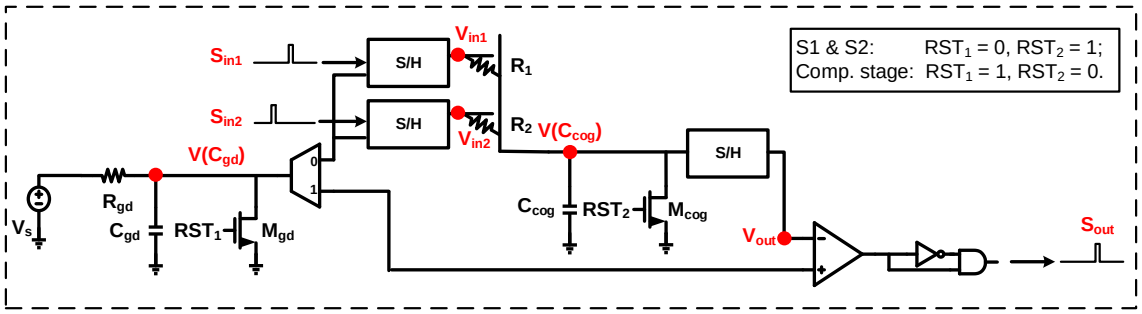


FIGURE 3.2: The schematic of a simplified circuit to implement single-spiking MAC.

3.3.2 Design Principle of Single-spiking MAC

As a novel form of data representation, the single-spiking data format simplifies an input or output signal as a single spike and use the timing information to express its

value. The following challenge is how to enable the MVM operations based on timing information on ReRAM-based PIM designs. Peripheral circuitry of ReRAM crossbar needs to re-designed to address this challenge and support the computation in the new data format. In this section, we will demonstrate our proposed circuit implementation to execute MAC operations, and consequently support MVM computation in the single-spiking data format scheme.

We elaborate the design concept and operation based on the simplified circuit in FIGURE 3.2. Controlled by two control signals, RST_1 and RST_2, the entire operation is divided into three steps: S1, computation, and S2. In S1 and S2, $RST_1 = 0$ and $RST_2 = 1$, the capacitor C_{gd} is charged; in the computation stage, $RST_1 = 0$ and $RST_2 = 1$, the capacitor C_{cog} is charged.

S1: Transit input timing to voltage. The capacitor C_{gd} is used as the timing reference. At the beginning of each slice, we start to charge it from 0V through a constant voltage source V_s. $V(C_{gd})$ continues to rise up until the computation stage, during which M_{gd} turns on and quickly discharges $V(C_{gd})$ to 0V. In S1, $V(C_{gd})$ is fed into the sample and hold (S/H) circuits associated with input signals to the ReRAM crossbar. When an input spike, e.g., S_{in1}, comes in with the arrival time of t_{in1}, $V(C_{gd})$ at the moment will be captured, such as

$$V_{in1} = V_s \left(1 - e^{-\frac{t_{in1}}{R_{gd}C_{gd}}}\right) \approx \frac{V_s t_{in1}}{R_{gd}C_{gd}}. \tag{3.1}$$

In this way, t_{in1}, the timing information of S_{in1}, is converted into V_{in1}, the voltage to the ReRaM cells. The approximation in Eq. (3.1) is satisfied when $\frac{t_{in1}}{R_{gd}C_{gd}} \ll 1$, which means that the charging of C_{gd} can be regarded as a linear process. If t_{in1} is relatively large, the non-linearity will be induced in the charging process. We will discuss the impact of this non-linearity in Section 3.3.4.

Computation. The short-time computation stage with a duration of δt happens at the end of S1. In the computation stage, the capacitor C_{cog} is charged simultaneously by both V_{in1} and V_{in2}, respectively via R_1 and R_2. In this process, the equivalent voltage source and

resistance can be calculated as:

$$V_{eq} = \frac{V_{in1}G_1 + V_{in2}G_2}{G_1 + G_2} \text{ and } R_{eq} = \frac{1}{G_1 + G_2}, \tag{3.2}$$

where G_1 and G_2 refer to the conductance of the two ReRAM cells R_1 and R_2, respectively. $V(C_{cog})$ is sampled at the end of the computation stage and can be calculated as:

$$V_{out} = V_{eq}\left(1 - exp^{-\frac{\delta t}{R_{eq}C_{cog}}}\right) \approx \frac{V_{eq}\delta t}{R_{eq}C_{cog}}. \tag{3.3}$$

The approximation in Eq. (3.3) is valid when the charging of C_{cog} can be regarded as a linear process. When R_{eq} is too small, the condition cannot be satisfied. We will discuss the impact of the non-linearity on the computation results in Section 3.3.4.

S2: Transit V_{out} to output spiking timing. In S2, C_{gd} follows the same charging and discharging operation as in S1. Different from S1, however, $V(C_{gd})$ is supplied to the comparator and compared with V_{out}. When $V(C_{gd})$ surpasses V_{out}, the comparator will generate a rising signal. The subsequent inverter is used to generate the pulse width delay and the AND gate produces the falling edge to form an output spike. The duration from the beginning of S2 to the presence of the generated spike is denoted by t_{out}, which can be calculated by using the following function:

$$V_{out} = V_s\left(1 - e^{-\frac{t_{out}}{R_{gd}C_{gd}}}\right) \approx \frac{V_s t_{out}}{R_{gd}C_{gd}}. \tag{3.4}$$

By combining Eqs (3.2)∼(3.4), we can present t_{out} in the format of the MAC result of G_1, G_2 and t_{in1}, t_{in2}:

$$t_{out} = \frac{\delta t}{C_{cog}}(t_{in1}G_1 + t_{in2}G_2). \tag{3.5}$$

At such, the computation result in voltage format is transferred into the time information of a single-spiking signal.

FIGURE 3.3 shows the simulation results of the proposed design. Here, the length of a full-scale time slice is set to 100ns. So performing the entire process including S1 and S2

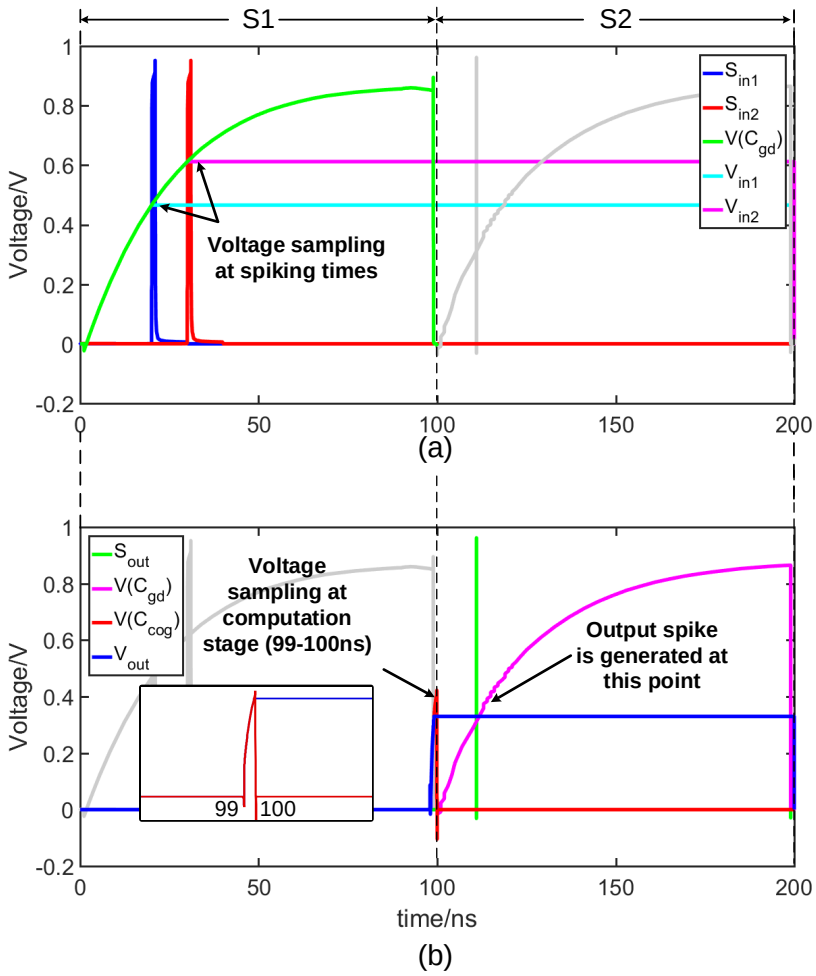


FIGURE 3.3: The circuit simulation. (a) The active waveform in S1; (b) The active waveform in the computation stage and S2.

takes 200ns. The computation stage with δt = 1ns occurs at the end of S1, i.e., during 99ns $\sim$ 100ns.

3.3.3 The Overview of ReSiPE Design

We use the simplified example in FIGURE 3.2 to elaborate how to implement the MAC operation with two input signals in the single-spiking data format. In this subsection, we generalize it to the MVM operation and propose a ReRAM-based single-spiking PIM engine, namely, ReSiPE.

FIGURE 3.4 presents the overview of ReSiPE microarchitecture. It consists of two

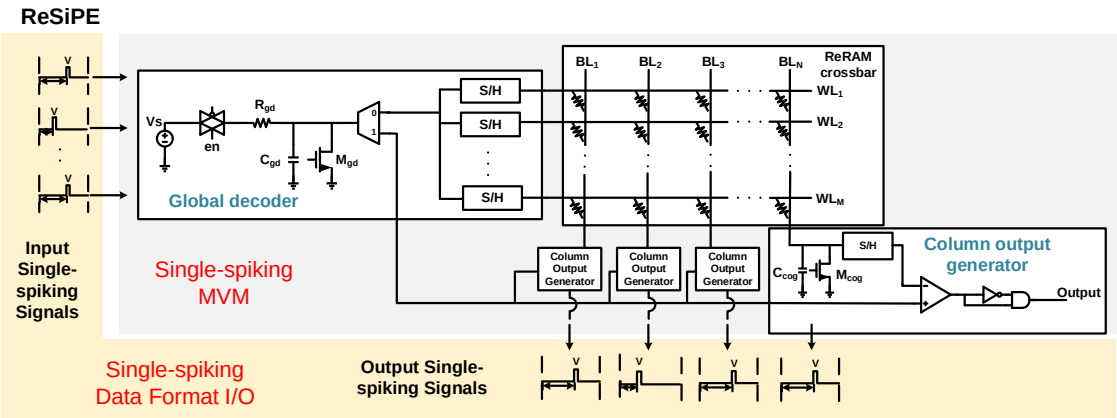


FIGURE 3.4: An overview of ReSiPE architecture.

main components—the I/O in single-spiking data format and the implementation of single-spiking MVM. ReSiPE supports I/O based on single-spiking data format, which has been hereinbefore interpreted. Here, we will focus on the implementation of single-spiking MVM. We append two additional modules to the ReRAM crossbar, which are a global decoder (GD) and a column output generator (COG) cluster. These two modules are derived from the example circuit proposed in Section 3.3.2.

Global decoder (GD): A ReRAM crossbar array is associated with one GD module. Its main function is decoding input single-spiking signals to wordline (WL) voltage levels of ReRAM crossbar. During S1, GD takes in input single-spike signals, produces the input voltages according to the arrival time of the spikes, and drives the ReRAM crossbar during the instant computation stage.

Column output generator (COG) cluster: A COG cluster consists of multiple COG modules, each of which is assigned to a bitline (BL) of the ReRAM crossbar. During the computation stage, a COG module samples the voltage that represents the computation result, and generates an output signal in the single-spiking data format during S2.

Single-spike MVM: Following the two-stage execution discussed previously, the ReRAM crossbar performs MVM operations with signals in single-spiking data format. Similar to the analysis in Section 3.3.2, for a ReRAM crossbar with M wordlines, the jth bitline

generates a spike with the timing approximating to

$$t_{out,j} = \frac{\delta t}{C_{cog}} \sum_{i=1}^{M} t_{in,i} G_i. \tag{3.6}$$

The output signals of bitlines will then be passed to the sequential layer.

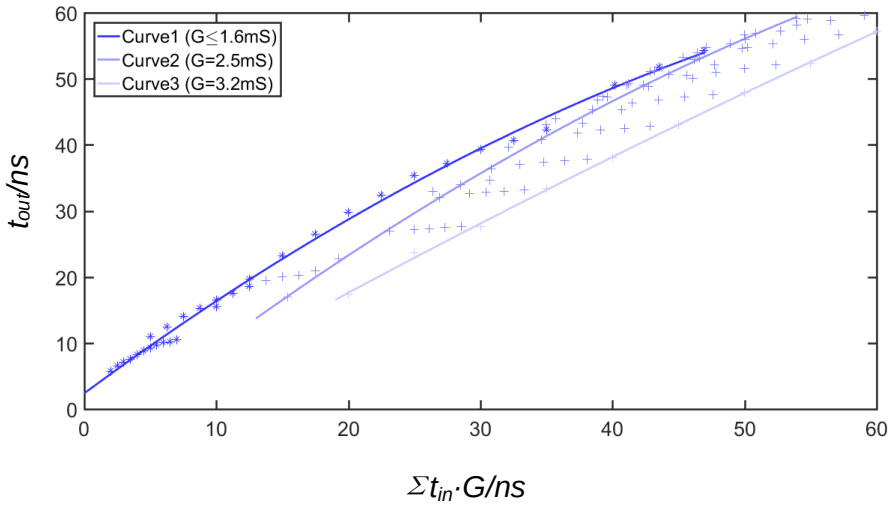


FIGURE 3.5: Input-output characterization: the MVM computing results t_{out} versus the input strength $t_{in}G$.

3.3.4 Computational Result Analysis

In this subsection, we present our simulation result and show that ReSiPE can correctly execute single-spiking MVM. The simulation was implemented on Cadence Virtuoso 6.1.6, using TSMC 65nm PDK. The size of the ReRAM crossbar was 32×32. We adopted the 1T1R structure of ReRAM cells. The low resistance state (LRS) was set to 10kΩ and the high resistance state (HRS) was set to 1MΩ. Both C_{gd} and C_{cog} were 100fF. V_s was set to 1V and R_{gd} was set to 100kΩ. The length of a full-scale time slice is 100ns and the pulse width of a single spike is set to 1ns.

FIGURE 3.5 shows the relation of the actual MVM simulation results and the input strength using 100 random sample points of different t_{in} and G. Here, the $x-$axis is the

the input strength of $\sum t_{in}G$ and the $y-$axis represents values of t_{out} that ReSiPE actually acquires. The total G varies from $0.32\text{mS} \sim 3.2\text{mS}$ and t_{in} varies from $10\text{ns} \sim 80\text{ns}$.

Our result indicates that t_{out} is affected by the non-linearity of $V(C_{cog})$ and $V(C_{gd})$ so we further investigate it. Here, `Curve 1` is the fitting curve of the sample points with total $G \leqslant 1.6\text{mS}$. Non-linearity of $V(C_{gd})$ is due to the large input t_{in}, which can be observed along `Curve 1` with the increase of $\sum t_{in}G$. The extent of the non-linearity is subtle, because C_{gd} is used for calibration in both S1 and S2, which partially cancels out the effect.

Non-linearity of $V(C_{cog})$ is due to the large total G of one column of ReRAM crossbar. It can be seen that the sample points in light blue color below `Curve 1` belong to the cases where the total conductance is larger than 1.6mS. The charging process of C_{cog} in the computation stage approaches to saturation quickly and thus t_{out} is smaller than the linear calculation, especially at big t_{in}. To better illustrate the scenario, we highlight `Curve 2` and `Curve 3` that fit the sample points with $G = 2.5\text{mS}$ and $G = 3.2\text{mS}$, respectively.

The simulation result leads to the conclusion that t_{out} can represents the MVM results of t_{in} and G with slight non-linearity under the condition that the total $G \leqslant 1.6\text{mS}$. This condition can be satisfied by setting the resistance range of ReRAM cells from $50\text{k}\Omega$ to $1\text{M}\Omega$ [67, 26], although the slight non-linearity still exists. We will investigate the impact of this slight non-linearity on the accuracy of the neural networks in Section 3.4.3.

3.4 Experimental Results

3.4.1 Experiment Setup

We evaluate the proposed ReSiPE engine in terms of power efficiency, latency and accuracy. We implement it with Cadence Virtuoso using TSMC 65nm process design kit (PDK). The size of ReRAM crossbar was set to 32×32. We adopted the one-transistor-one-ReRAM (1T1R) structure of ReRAM cells in simulation [8]. Calibrated with the clock frequency of 1GHz, the length of a slice is 100ns and the length of the computation stage is set to 1ns. The same circuitry parameters in Section 3.3.4 are adopted.

3.4.2 Power, Latency and Area Comparison

We compare the power, power efficiency, computation latency and area of our proposed ReSiPE with several representative PIM designs, as summarized in TABLE 3.2. The comparison counterparts were carefully selected to cover different data formats, including the level-based [42, 8], the PWM-based [25], and rate-coding based [66, 37].

Table 3.2: Comparison with recent works

Type	**Single-spiking**	Level-based	Level-based	Rate-based	Rate-based	PWM-based
Work	**This work**	[42]	[8]	[66]	[37]	[25]
Synapse	**1T1R**	1T1R	1T1R	1T1R	1T1R	1T1R
Technology	**65nm**	180nm	65nm	150nm	130nm	130nm
Power (mW)	**0.50**	15.8	N/A	1.52	3.08	2.455
Energy efficiency (TOPS/W)	**40.8**	20.7	16.95	16.9	N/A	0.82
Layer per layer (ns)	**100**	N/A	14.8	200	100	320
Area (kF^2)	**8.9**	N/A	60.42	37.15	10.37	N/A

1) Power efficiency: We test the power efficiency of the processing engines under the same array sizes with ReRAM devices are fully utilized. ReSiPE shows 1.97×, 2.41× and 49.76× higher power efficiency than the representative level-based, rate-coding and PWM design, respectively. This improvement originates from the simplification of mixed-signal interface circuit, the reduction of spike numbers and the elimination of the time-domain drivers.

According to our estimation, the major component of the power consumption comes from the COG cluster, because the capacitor C_{gd} needs charging during every slice. Experimental results show that the COG cluster contributes to 98.1% of the entire power consumption. Future technology scaling that enables smaller Metal-Insulator-Metal capacitors in COG clusters would bring further energy reduction.

2) Latency and area: Previous rate-coding based and PWM-based designs require a relatively long computing latency to transfer a spike series or a modulated pulse. Due to the elaborated data format scheme, ReSiPE shortens the computing latency 50% and 68.8% than rate-coding and PWM-based baselines, respectively. It doesn't improve com-

puting speed much compared to the level-based baselines as these designs utilize high-speed DAC/ADC to speedup the computation.

We estimate the areas by scaling all the design to the same ReRAM crossbar size. ReSiPE can dramatically reduce the design area than other counterparts. Experimental results demonstrate that ReSiPE saves 14.2% and 85.3% area compared to rate coding-based and level-based baselines, respectively. The reason why ReSiPE considerably reduces the area compared to level-based designs lies in the elimination of area-consuming DAC/ADCs [43]. The analysis also indicates a trade-off between computing latency and area to achieve higher throughput. For example, to compete with level-based designs in terms of throughput, we may increase the ReSiPE numbers to improve the parallelism level. FIGURE 3.6 shows the estimated throughput of such an approach. Under same area budget and ReSiPE provides much higher throughput than other designs.

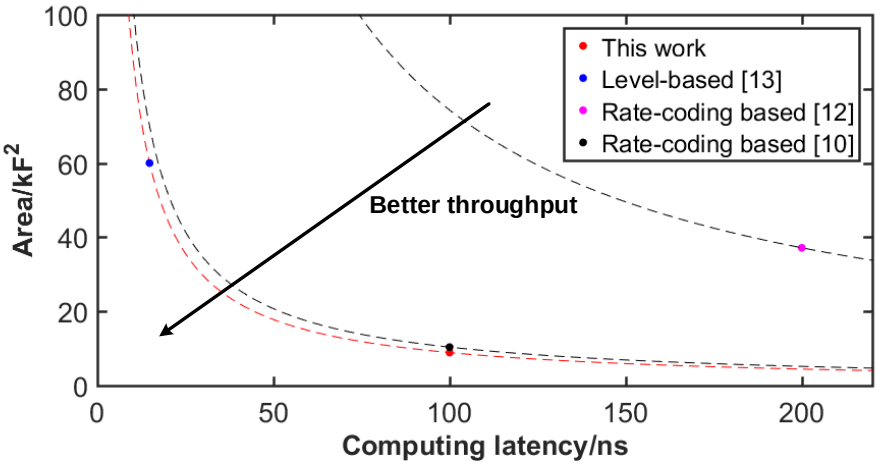


FIGURE 3.6: The tradeoff between computing latency and design area. Dashed lines imply the overall throughput under certain latency and area constraint.

3.4.3 Accuracy Analysis

We adopted six pretrained networks to evaluate accuracy:

- MLP-1: 1-layer perceptron network on MNIST;
- MLP-2: 2-layer perceptron network on MNIST;

- CNN-1: 4-layer LeNet on MNIST;
- CNN-2: AlexNet on Cifar-10;
- CNN-3: VGG16 on Cifar-10; and
- CNN-4: VGG19 on Cifar-10.

The pretrained networks are mapped to the circuitry implementation to acquire the final classification accuracy. The non-linearity of circuit design principle, discussed in Section 3.3.4, is taken into account. The impact of process variation (PV) of ReRAM cells is also investigated, which follows the normal distribution according to [36, 38]. In the experiment, we adopt the process variation with the standard deviation $\sigma = 0$, 5%, 10%, 15%, 20%.

As shown in FIGURE 3.7, the classification accuracy drop of the case $\sigma = 0$ compared to the ideal accuracy is caused by the non-linearity presented in Section 3.3.4.This slightly non-linearity leads to less than 2.5% accuracy drop. The device variation of 20% leads to 1% $\sim$ 15% accuracy drop. The impact of PV is more significant in more complex neural networks model, which is more sensitive to the variation. Nevertheless, the accuracy is still competitive.

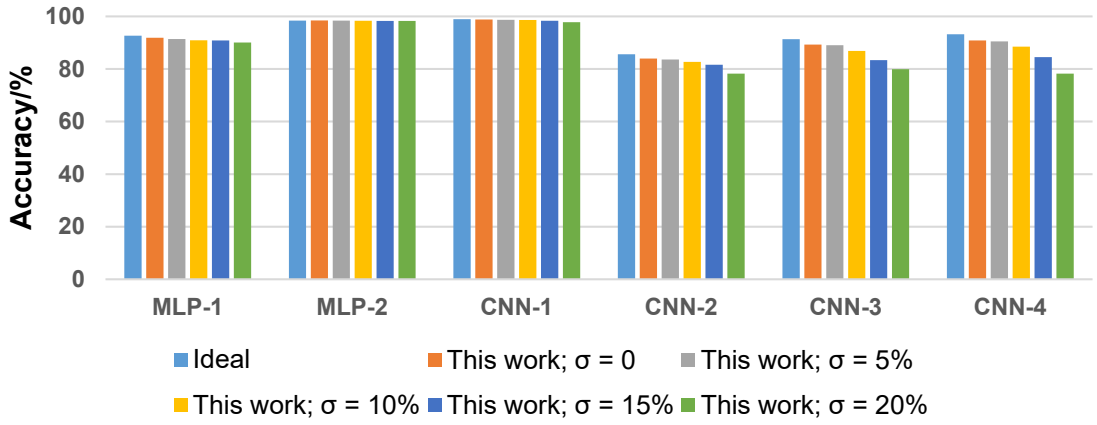


FIGURE 3.7: Accuracy comparison between models considering PV.

3.5 Conclusion

In this chapter, we propose a novel ReRAM-based single-spiking processing-in-memory engine called ReSiPE, aiming at improving the power efficiency of ReRAM-based PIM designs. The existing level-based designs involve constant voltage input during computation, while rate-coding based designs require repetitive spike generation. Both of them need additional ADCs or IFCs to conduct data format conversion. Motivated by these challenges, ReSiPE features the single-spiking data format and MVM schemes, which represents both input and output data in single-spiking signals and executes MVM computation. Experiments show that ReSiPE achieves 67.1% power reduction with competitive area, throughput and accuracy. With our proposed single-spiking data format, post-spike latency could be potentially reduced by multi-layer pipelining. ReSiPE is hence open to future microarchitecture optimization toward better layer-wise computing latency.

4. ReRAM-based Spike-timing Neuromorphic Design for Temporal-coded SNN Inference

The TTFS encoding scheme has been introduced to SNNs due to its extremely simple data representation. In the last chapter, ReSiPE adopts the single-spike encoding scheme, which is inspired by TTFS, for DNN computation to greatly reduce the spike number and computation period. However, the frame-driven DNNs cannot fulfill the potential of TTFS because the computation is still driven by fixed clock cycle repeatedly. When applying TTFS to SNNs, the data format of a single spike event allows the possibility of eliminating unnecessary idle period without the restriction of fixed clock cycle. The idea will be discussed in this chapter.

Although TTFS-based SNNs enable the spiking neuromorphic hardware to further enhance the energy efficiency, there is a lack of concrete circuit designs of PIM-based neuromorphic hardware for supporting it. In this chapter, we propose ASTERS, a ReRAM-based neuromorphic design to conduct the TTFS-based SNN inference in a PIM manner. Different from ReSiPE, ASTERS is a dedicated SNN processing engine that harnesses the event-driven computation instead of the frame-driven DNN computation. We devise peripheral circuits and the synapse mapping scheme for ASTERS. A timing threshold scheme is utilized to eliminate unnecessary spikes and speed up the inference. We also propose a multi-level firing threshold adjustment scheme to mitigate the impact of ReRAM device process variations and elevate the inference accuracy. Integration of the two techniques enables the efficient and robust TTFS neuromorphic design. Experimental results show that our proposed ASTERS achieves more than 34.7% energy savings compared to existing spiking neuromorphic designs, while maintaining 90.1% accuracy under the process variations with a 20% standard deviation.[1]

[1] Ziru Li et al. "ASTERS: adaptable threshold spike-timing neuromorphic design with twin-column ReRAM synapses". In: Proceedings of the 59th ACM/IEEE Design Automation Conference. 2022, pp. 1099–1104. © 2022 Association for Computing Machinery. https://doi.org/10.1145/3489517.3530591

4.1 Introduction

In the last chapter, we focus on building a spike-based DNN macro with ReRAM-based PIM. However, accelerators for DNNs cannot provide satisfactory energy efficiency for low-power applications, such as portable and Internet of Things (IoT) devices, due to its synchronous computation driven by high frequency clock and intensive data transmission. The SNN offers an alternative *event-driven* solution for high energy efficiency, since it features asynchronous computation and extremely sparse data format. As introduced in Section 1.1, there are several ways to modulate information on spike events. Generally, fewer spike events generated during execution imply less energy consumption. The rate-coding scheme is also adopted in SNNs, in which data are carried by the spike frequency [6, 10]. A spike train with a higher frequency propagates a stronger signal to its destination neuron. Usually a large amount of spike events are needed to convey a datum via averaging the temporal information of the spike events. Another limitation of rate-coding-based networks is that the inference accuracy is significantly affected by the parameters such as the firing threshold [68]. Temporal coding scheme, which encodes data to spike times, creates a new path to achieve high efficiency by enabling the data representation with sparse spikes [17]. Particularly, the TTFS encoding scheme [56] uses the arrival time of the first spike in a time window to represent data and reduces the spike number to just one with extremely low energy consumption.

The spiking neuromorphic systems, either digital [41, 1, 16, 9] or mixed-signal [47, 3] designs, boost the power-efficiency by leveraging the event-driven characteristics of SNNs. Besides the CMOS spiking neuromophic systems, ReRAM-based spiking neuromorphic systems also emerges as a PIM solution. There have been rate-coding-based DNN process engine (PE) designs [37, 66], which count the spike numbers in fixed-time windows to decode the data. These designs suffer from the large quantization errors caused by the averaging-distributed spikes of the rate-coding scheme. In contrast, ReSiPE [31] is a single-spiking ReRAM-based DNN PE, which adopted TTFS encoding by taking the arrival time of a

single spike to represent a datum. In ReSiPE, pre- and post-synaptic spikes are located in separate time windows and the computation is synchronous, which results in increasing latency when executing multi-layer networks. These neuromorphic designs avoid the conventional power-hungry DACs & ADCs [30] and thus boost the energy efficiency. However, they operate with a regular computation latency for each layer, and hence, are not able to flexibly scale down the computation latency and make full use of the event-driven characteristics of SNNs for even faster inference.

Despite the high potential of TTFS-based SNNs in boosting information processing efficiency, there is a lack of concrete circuit designs of ReRAM-based neuromorphic hardware for supporting it. *Energy efficiency* of the ReRAM-based TTFS neuromorphic design and *inference robustness* to the ReRAM device process variations are the two key criteria that need to be considered. In TTFS-based SNNs, a neuron generates only one spike in each time window, and remains idle till the end of the time window. Besides, because the data strength is encoded to the arrival time of the spike, the later spike generation is less significant and can be removed without much affecting the overall activation propagation. Making full use of such characteristics of TTFS-based SNNs in the hardware design can further improve the energy efficiency of the spiking neuromorphic design. The ReRAM device process variations inject noises to the conductance, inducing erroneous activation strength and firing time of post-spikes, and thus, degrading the inference accuracy [7]. This demands for techniques to mitigate the impact of ReRAM device process variations and improve the inference robustness of the TTFS neuromorphic design. We summarized the following questions when implemented SNNs with ReRAM crossbars:

- How to map the excitatory and inhibitory synapses to the ReRAM crossbar array?
- How to leverage the event-driven property of SNNs to reduce energy consumption as most neurons are in the idle state?
- How to mitigate the impact of ReRAM device process variations on the accuracy of SNN computation?

In order to maximize the efficiency gain of the emerging neuromorphic computing

paradigm, in this chapter, we propose ASTERS, the **A**daptable threshold **S**pike-timing neuromorphic design with **T**win-column **Re**RAM **S**ynapses. ASTERS is a circuit scheme for efficient SNN inference with two SNN threshold adjusting techniques that can be applied on the proposed hardware platform. More specific, the main contributions of this work include:

1. We use twin-column excitatory/inhibitory synaptic weight mapping scheme (contrast to the conventional row-wise mapping scheme) and design a brand-new set of spike-timing axon and neuron circuits for ReRAM crossbar array. To our best knowledge, ASTERS is the first ReRAM-based neuromorphic design that elaborates a complete set of peripheral circuits to compute TTFS-SNNs and leverages the event-driven characteristics to speed up the computation.
2. Regarding the widely-concerned process variation of ReRAM nano-devices, we propose a multi-level "firing threshold adjustment" method that effectively recovers the inference accuracy degradation with only 1.5% hardware overhead.
3. We also devise the "timing threshold adjustment" method, a design-automation technique specifically for the proposed post-synaptic neuron circuits. It removes a considerable amount of unnecessary spike generation between adjacent neural network layers and thus speeds up the inference.

We implement the circuits with the TSMC 65nm process design kits (PDK) and verify by simulations. The simulation shows that ASTERS achieves at least 34.7% energy reduction compared to existing neuromorphic systems [47, 41, 3, 9] at the same technology node with the same ReRAM device characteristics. Our design also presents an outstanding resilience to variations by maintaining 90.1% accuracy under the process variations with a 20% standard deviation.

4.2 Comparison with Prior ReRAM-based Spiking Neuromorphic Designs

As depicted in the top row of FIGURE 4.1, the conventional DNN spiking neuromorphic designs [66, 31] adopt the clock-driven dataflow. The excitatory and inhibitory synapses are

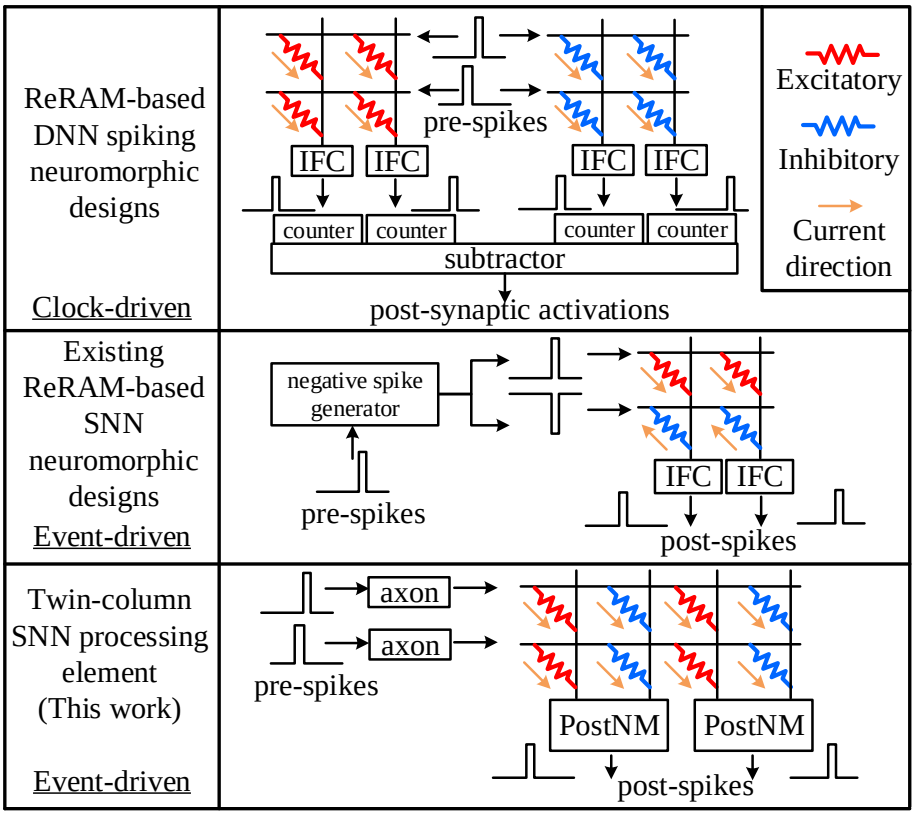


FIGURE 4.1: Comparison between ReRAM-based spiking neuromorphic designs for DNNs (top) and SNNs (middle and bottom).

mapped to different spatial positions of ReRAM arrays. The computation with positive and negative synaptic weights are performed independently, and the post-synaptic activations are calculated from the difference of the bitline outputs at the clock edge. This is suitable for the clock-driven DNN hardware because it has clock edges to trigger continuous frame-based computation. The differential operation occurs at the end of each frame computation. However, in the event-driven SNN hardware, the post-synaptic activations are accumulated throughout the time window. Hence, the activations from the excitatory synapses should be subtracted by those from the inhibitory synapses every moment and accumulated by the post-neuron modules, making clock-driven dataflow infeasible. In the existing ReRAM-based SNN neuromorphic designs [60, 59] mentioned in Section 2.2, the excitatory and inhibitory synapses are mapped to adjacent rows to form a differential pair, and input positive and negative spike voltages are fed into the two rows respectively, as depicted in

FIGURE 4.1 (middle), requiring additional circuits to reverse the pre-synaptic spikes.

4.3 Twin-Column SNN PE

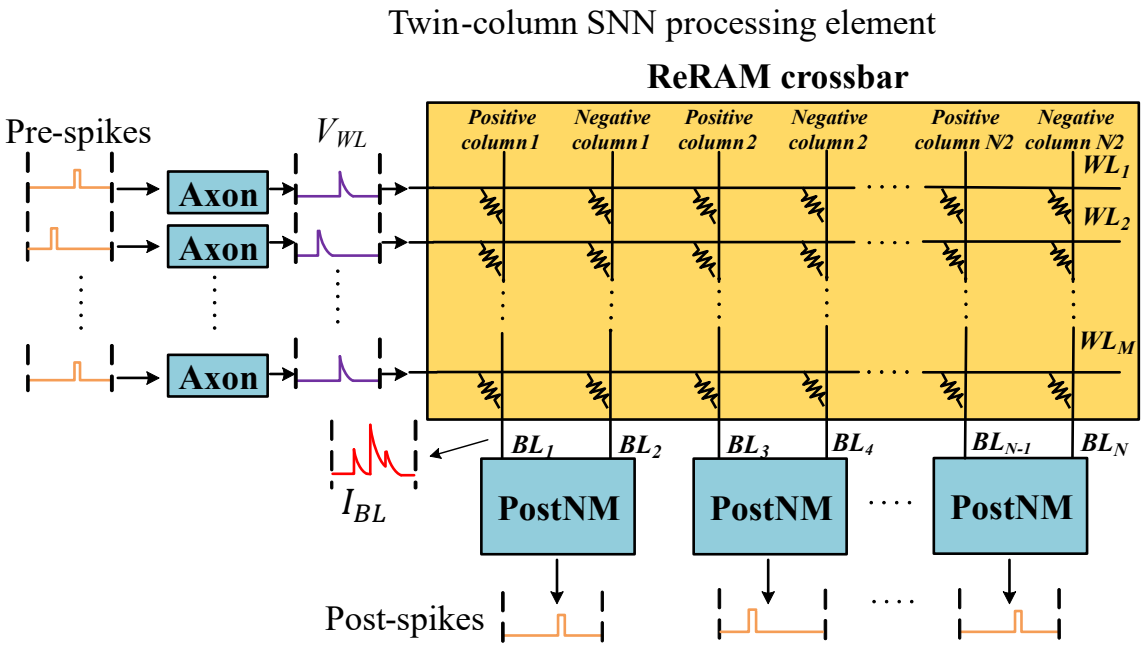


FIGURE 4.2: Circuit components of our twin-column SNN processing element.

This section and following section will describe the proposed cross-layer solution to realizing TTFS-based computation. First, we introduce the improved basic circuit components together with the synapse mapping scheme (Section 4.3.1-4.3.3). Then, we propose two novel computer-aided design methods, "multi-level threshold adjustment scheme" (Section 4.4.1) and "timing threshold adjustment scheme" (Section 4.4.2), to improve the inference robustness and energy efficiency of the TTFS neuromorphic design, respectively.

4.3.1 Fundamental Circuit Components

FIGURE 4.2 presents the overview of our twin-column SNN processing element. The fundamental circuit components to a TTFS-based SNN comprise of a set of axon modules to generate exponentially-decaying spike waveform, the ReRAM synaptic crossbar, and a set of post-neuron modules to implement IF neurons.

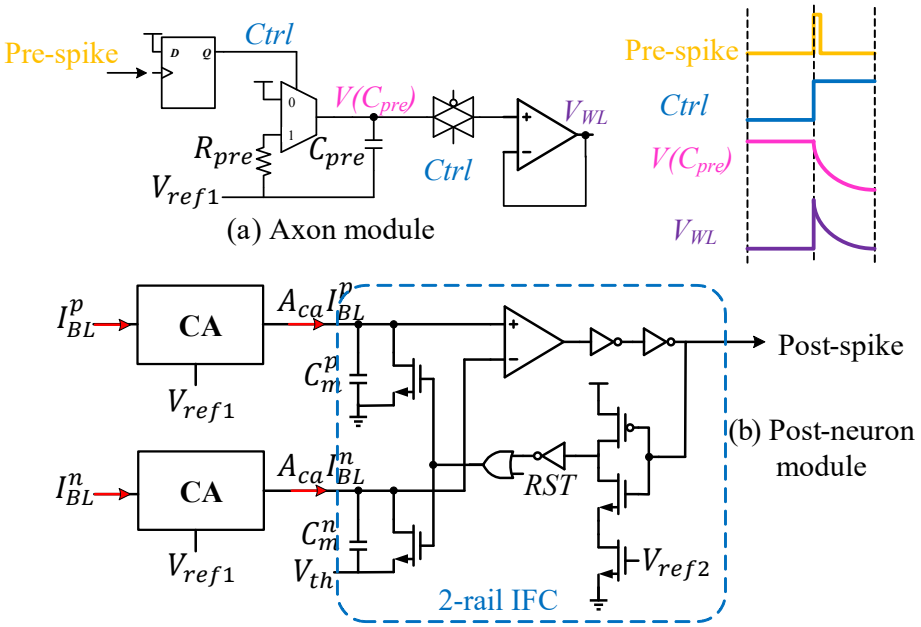


FIGURE 4.3: (a) The axon module design. (b) The 2-rail post-neuron module design.

4.3.1.1 Axon module

FIGURE 4.3(a) depicts the axon module. It works as a low-pass filter to process the pre-spikes. The core is a resistor-capacitor pair of R_{pre} and C_{pre} determining the time constant τ_s of the exponentially-decaying waveform. The axon module will generate an exponentially-decaying voltage signal with a bias of V_{ref1} at the arrival of the pre-spike t_i, formulated as:

$$V_{WL,i}(t) = (V_{DD} - V_{ref1})\theta(t - t_i)\exp\left(-\frac{t - t_i}{R_{pre}C_{pre}}\right) + V_{ref1} \tag{4.1}$$

The bias voltage V_{ref1} is enforced because the following post-neuron module needs to hold the bitline voltage as V_{ref1} (explained in Sec 4.3.1.3), and the ReRAM crossbar wordline voltages need to rise accordingly.

4.3.1.2 ReRAM crossbar

The synapses stored in the ReRAM crossbar modulate the input wordline voltages and generate CuBa activations in the form of bitline currents. Assume that the conductance of the ReRAM cells along bitline j connected to a post-neuron module are $G_{i,j}$,

$i = 0, 1, \cdots, M-1$, where M is the total number of the synapses connected to a post-synaptic neuron. The bitline voltage $V_{BL,j}$ is held as V_{ref1} by the post-neuron module. The strength of CuBa activation is calculated as:

$$\begin{aligned} I_{BL,j} &= \sum_{i=0}^{N-1} G_{i,j}(V_{WL,i}(t) - V_{BL,j}) \\ &= \sum_{i=0}^{N-1} G_{i,j}(V_{DD} - V_{ref1})\theta(t - t_i)\exp(-\tfrac{t-t_i}{R_{pre}C_{pre}}) \end{aligned} \tag{4.2}$$

Note that there are excitatory and inhibitory synapses in SNN models. We need a novel synapse mapping scheme and corresponding neuron circuit designs to support it, which results in our twin-column synapse mapping scheme.

4.3.1.3 Post-neuron module

The bitline currents are collected by the post-neuron modules. This module implements the standard integrate and fire (IF) neuron model—integrating the CuBa activation and firing a single post-spike in each time window. The post-neuron module includes two sub-modules: current amplifier (CA) and integrate-and-fire circuit (IFC). The design in [66] uses a CA to keep the bitline voltage to a stable value. Here, we adopt a similar design to hold $V_{BL,j} = V_{ref1}$, so that the increase of the membrane potential will not affect the linearity of the synaptic connection. By calibrating the circuit parameters, a current gain A_{ca} can be derived from CA module. It will also introduce a delay of t_{delay} to the activation. The activation from CA is then passed to the IFC to accumulate the voltage on a capacitor. Post-spikes are fired once the voltage exceeds a preset threshold voltage V_{th}.

4.3.2 Twin-Column Synapse

In our proposed twin-column SNN processing element, we use the two adjacent columns of ReRAM bitcells to represent the excitatory and inhibitory synapses. These twin columns connect to the same post-neuron module. The difference of I_{BL} of these two twin columns, namely I_{BL}^{p} and I_{BL}^{n}, is the overall CuBA activation that flows into the post-neuron module. The novel post-neuron module computes the difference of the complementary twin-column currents and execute neural dynamics. Each post-neuron module contains two CA along the

twin columns, and one 2-rail IFC. The 2-rail IFC module, shown in FIGURE 4.3(b), contains two capacitors C_m^p and C_m^n with identical capacitance C_m. The two capacitors accumulate membrane potentials from I_{BL}^p and I_{BL}^n, respectively. C_m^p is connected to the ground while C_m^n is biased at V_{th}. Therefore, V_m^p, the voltage of C_m^p, represents the potential accumulated by I_{BL}^p from 0; while V_m^n, the voltage of C_m^n, corresponds to the potential accumulated by I_{BL}^n from V_{th}. A comparator detects whether V_m^p is larger than V_m^n. Once detected, it generates a rising voltage step output, indicating the overall potential accumulated by $I_{BL}^p - I_{BL}^n$ is larger than V_{th}. Subsequently, a following inverter discharges the membrane capacitors (C_m^p and C_m^n) and resets the membrane potentials (V_m^p and V_m^n) with a delay controlled by V_{ref2}. This results in an output post-spike with controllable pulse width.

The neural dynamics performed in the post-neuron module is formulated as:

$$\begin{aligned} u &= V_m^p(t) - (V_m^n(t) - V_{th}); \\ V_m^p(t) &= \textstyle\int_0^t \frac{1}{C_m} A_{ca} I_{BL}^p(t - t_{delay})dt; \\ V_m^n(t) &= V_{th} + \textstyle\int_0^t \frac{1}{C_m} A_{ca} I_{BL}^n(t - t_{delay})dt; \\ C_m \tfrac{du}{dt} &= A_{ca}[I_{BL}^p(t - t_{delay}) - I_{BL}^n(t - t_{delay})]. \end{aligned} \tag{4.3}$$

Eq. (4.3) resembles Eq. (2.2), indicating that the twin-column PE is able to compute the IF neural dynamics.

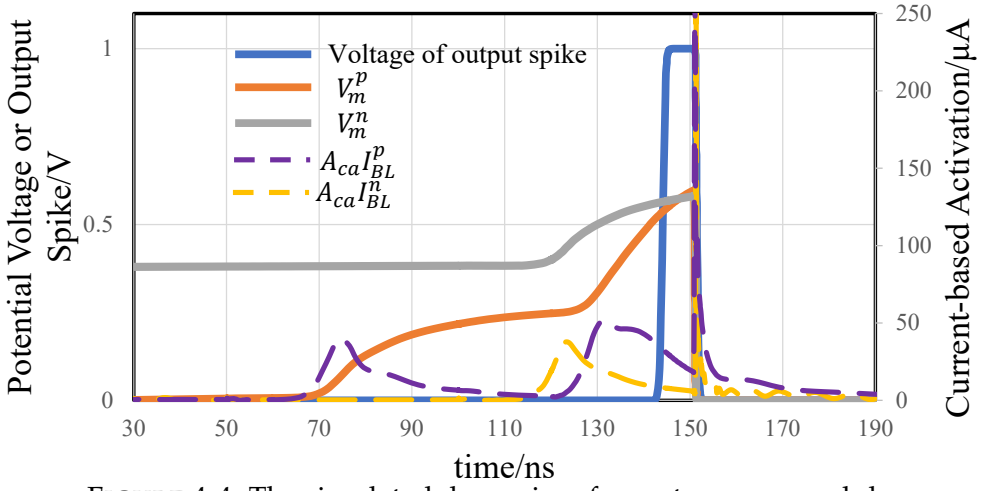


FIGURE 4.4: The simulated dynamics of a post-neuron module.

FIGURE 4.4 shows the simulation waveform of the post-neuron module, validating the function of twin-column PE. The key circuitry parameters are summarized in Table 4.1. $A_{ca}I_{BL}^{p}$ denotes the CA output current from the positive column, and $A_{ca}I_{BL}^{n}$ is the corresponding one from the negative column. In this example, $A_{ca}I_{BL}^{p}$ has two spikes at $74ns$ and $131ns$ taken in by the 2-rail IFC; there is one spike of $A_{ca}I_{BL}^{n}$ at $123ns$. V_m^p is accumulated by $A_{ca}I_{BL}^{p}$ from 0, while V_m^n is accumulated by $A_{ca}I_{BL}^{n}$ from $V_{th} = 400\text{mV}$. When $V_m^p(t)$ exceeds $V_m^n(t)$ ($\sim 150ns$), the IFC generates an output post-spike. Note that glitches occur to the inputs of the IFC when the output spike is generated. Only the first spike matters in the TTFS encoding, these glitches do not affect the propagation of the post-spike.

Table 4.1: Circuitry Parameters in Simulation.

Symbol	Value	Symbol	Value
R_{on}/R_{off}	50k■/1M■	ReRAM crossbar size	32×32
C_{pre}	2.43pF	V_{DD}	1V
R_{pre}	1.32k■	$V_{th,0}$	400mV
C_m^p	2.69pF	V_{ref1}	100mV
C_m^n	2.69pF	V_{ref2}	150mV

4.3.3 Power-Gating of Post-Neuron Module

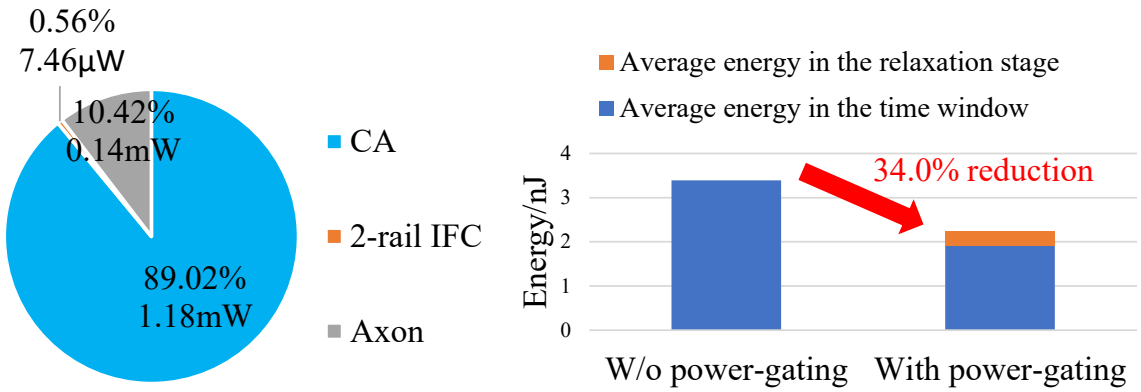


FIGURE 4.5: (a) Power breakdown of computing peripheral circuits; (b) Power-gating technique saves energy per iteration.

FIGURE 4.5(a) shows the power breakdown of the peripheral circuits. The CA dominates the overall power consumption owing to the internal static bias currents. After a post-spike is fired, the post-neuron module enters a refractory period, during which the

neuron cannot be fired, until the end of this compute iteration. Here we use the power-gating technique to reduce the power consumption during the refractory period. Once the post-spike is fired, the gating transistor will be turned off until the next compute iteration starts.

To achieve this, an extra relaxation stage is inserted in the control sequence to ensure the supply voltage becomes stable after the power-gating transistor switches on. We simulate the effect of power-gating to the post-neuron module with SPICE. The parasitic capacitance and resistance of the power-mesh with the estimated area of 16 post-neuron modules are extracted, corresponding to a 32×32 ReRAM crossbar. Based on the simulation results, a $200ns$ relaxation stage is needed if 16 post-neuron modules are switched on simultaneously. The energy consumption of one post-neuron module during the relaxation stage is $320.7pJ$, smaller than the energy reduction by power gating. For example, when the time window size is $2.56\mu s$, the average neuron firing time when doing MNIST classification is $1.31\mu s$. In this application, power-gating saves about 34.0% energy for one post-neuron module on average, as shown in FIGURE 4.5(b).

4.4 Firing/Timing Threshold Adjust

Aiming to further improve the efficiency and robustness of the twin-column SNN PE, we propose two schemes, i.e. multi-level firing threshold adjustment (MFTA) and timing threshold adjustment (TTA). MFTA and TTA are orthogonal to each other and can be simultaneously applied to ASTERS.

4.4.1 Multi-level Firing Threshold Adjustment

ReRAM device process variations are inevitable. They impose noises to the CuBa activation, affect the post-spike firing times, and subsequently degrade the computational accuracy. This problem can be effectively solved by calibrating firing thresholds V_{th} to correct the firing times. When the post-spike of a post-neuron is fired earlier than expected due to the ReRAM process variations, we deliberately raise its V_{th} to postpone the post-spikes, and vice versa.

MFTA is applied to SNN processing element right after deploying the pre-trained SNN model to the ReRAM crossbars. In MFTA, V_{th} of each post-neuron module can be selected from a preset group of evenly-distributed discrete values $\{V_{th,1}, V_{th,2}, ..., V_{th,2N}\}$ rather than continuous variables.

MFTA is applied to SNN processing element only *once* for each SNN deployment before iterative workloads begin. A set of input test patterns are fed to adjust the thresholds. MFTA starts from $V_{th} = V_{th,N}$, which is the expected global threshold voltage in the pre-trained SNN model. Before MFTA starts, the expected firing time of the *n-th* neuron in Layer l for the *i-th* test input, denoted as $t_{exp}(i, l, n)$, is derived from the pre-trained SNN and stored in the cache.

Algorithm 1 shows the principle of our proposed MFTA scheme. For each test input $Input_i$, MFTA starts from the first layer ($l = 1$). The threshold voltage of all the neurons in the first layer will be adjusted according to the difference between $t_{exp}(i, 1, n)$ and the actual post-spike time $t(i, 1, n)$. If $t(i, 1, n) > t_{exp}(i, 1, n)$, the threshold voltage of the *n-th* neuron lowers one discrete step of preset threshold values; otherwise it rises up to the next higher preset threshold value. Through experiments, we observe that it takes no more than 10 iterations in most cases to converge into an optimal threshold value. When all the neuron thresholds in the SNN model have been adjusted, the next input test pattern will be taken in. By repeatedly conducting MFTA with different inputs, the thresholds are adjusted to proper levels to compensate for the firing time deviation caused by process variations.

Apparently, how to feed input test patterns for threshold adjustment is critical for MFTA scheme. By setting C (adjustment number allowed by each neuron), Algorithm 1 guarantees that all the firing thresholds can be involved in the adjustment. This method also prevents unnecessary threshold changes and dynamically schedules the number of required test inputs.

Based on the proposed MFTA method, we implement the circuit for MFTA control logic at TSMC 65nm technology node. The block diagram is depicted in FIGURE 4.6 (1). The circuit results in only 1.5% energy overhead for our SNN processing element. The efficacy

Algorithm 1: MFTA scheme

Input: Target firing time of neurons $t_{exp}(i,l,n)$
Input: No. of MFTA each neuron should complete C
Output: The threshold of each neuron $V_{th}(l,n)$
No. of MFTA each neuron has completed $c(l,n) = 0$;
The initial threshold of each neuron $V_{th}(l,n) = V_{th,N}$;
while *Any* $c(l,n) < C$ **do**
  for $(l = 0; l < L; l++)$ **do**
    Convergence detector $CD(l,n) = 0$;
    Value indicating last adjustment result $r(l,n)$;
    while $CD(l,n) = 0$ **do**
      $t(i,l,n) =$ **TTFS** $(Input_i)$ $//Input_i$ is processed;
      if $t(i,l,n) < t_{exp}(i,l,n)$ *and* $c(l,n) < C$ **then**
        $V_{th}(l,n)++$; $c(l,n)++$;
        if $r(l,n) = 0$ **then** $CD(l,n) = 1$;
        else $r(l,n) = 1$;
      else if $t(i,l,n) > t_{exp}(i,l,n)$ *and* $c(l,n) < C$ **then**
        $V_{th}(l,n)--$; $c(l,n)++$;
        if $r(l,n) = 1$ **then** $CD(l,n) = 1$;
        else $r(l,n) = 0$;

of MFTA scheme is given in Section 4.5.2.

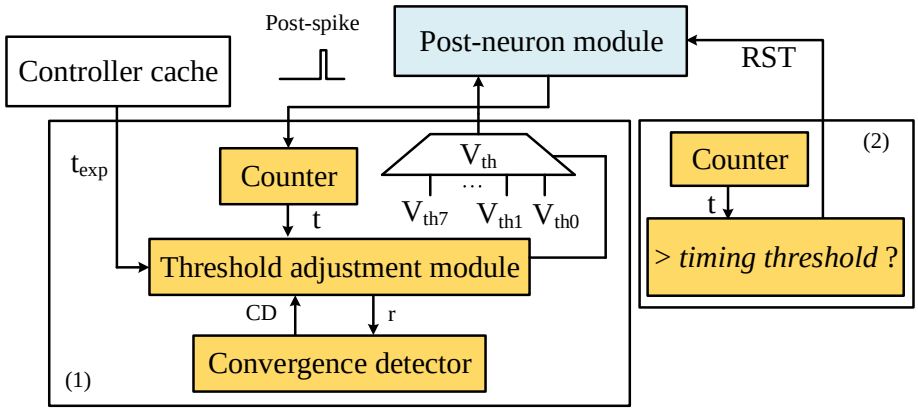


FIGURE 4.6: The block diagram of (1) the multi-level firing threshold adjustment module and (2) the timing threshold adjustment controller.

4.4.2 Timing Threshold Adjustment

While MFTA improves the resilience against device process variations, TTA scheme speeds up the inference and enables further energy savings. In each TTFS-SNN layer, neurons fire spikes earlier, representing stronger activations; while other neurons fire spikes later and contribute less to the membrane potential accumulation of the neurons in the

subsequent layers. We propose TTA scheme for our neuromorphic designs by taking advantage of the event-driven nature of TTFS-based SNN models. A *timing threshold* will be set. During each iteration, the post-spikes that fire before the *timing threshold* will be propagated. At the *timing threshold*, the activation propagation of all the neurons will be ceased, and the next iteration starts.

TTA scheme cuts the duration of iterations by allowing only a portion of neurons in each layer to propagate their spikes. It helps reduce the energy consumption as it shortens the time windows and eliminates unnecessary spike generation. On the other hand, the inference accuracy could be affected due to the loss of some post-spikes, but the degradation is marginal because the propagated strong activation has contributed to the majority of the destination neurons' potential. We will discuss the detailed trade-off between the benefits of TTA scheme and the accuracy degradation in Section 4.5.3.

4.5 Experiments

4.5.1 Experimental Setup

The simulation of ASTERS is conducted with TSMC 65nm PDK. Each ReRAM crossbar array is 32×32. Gaussian-distributed noises from the experimental research [49, 36] are injected into the ReRAM conductance to verify the performance of MFTA scheme.

We observe that complex SNN models, e.g. deep convolutional spiking neural networks (DCSNN), often mix the origins of the accuracy degradation (if any) with the algorithmic reason or the hardware reason. And the basic operations are the same regardless the model size. Therefore, we only present our investigation for the hardware on the well-studied TTFS-SNN perceptrons on the MNIST dataset. As long as the mechanisms of deeper TTFS-SNN models are fully investigated, more works can be conducted with the same computing principles. The pre-trained SNN model has 400 hidden-layer neurons and 10 output neurons. Its ideal inference accuracy without considering process variations is 95.4%. For the standard IF neuron module, an axon time constant of $\tau_s = 20ns$ and a $2.56\mu s$ time window are adopted. The average inference energy consumption is calculated

as the arithmetic mean of the energy consumption of the MNIST test set.

4.5.2 Accuracy Analysis of MFTA

MFTA scheme is to rescue the inference accuracy from ReRAM device process variations. The efficacy of MFTA lies in the two key parameters: C and $2N$.

Dependency on C: C is the iteration number of threshold adjustment neurons should complete. A larger C means that each firing threshold is involved in more adjustments triggered by more input testing patterns. To quantitatively analyze this, the simulation injects process variations with different standard deviation $\sigma = 10\%$, 20%, 30% and 40% to the ReRAM conductance. For each process variation setup, we conduct 50 experiments and obtain the average inference accuracy. We set the number of available threshold levels $2N = 16$ for this set of experiments.

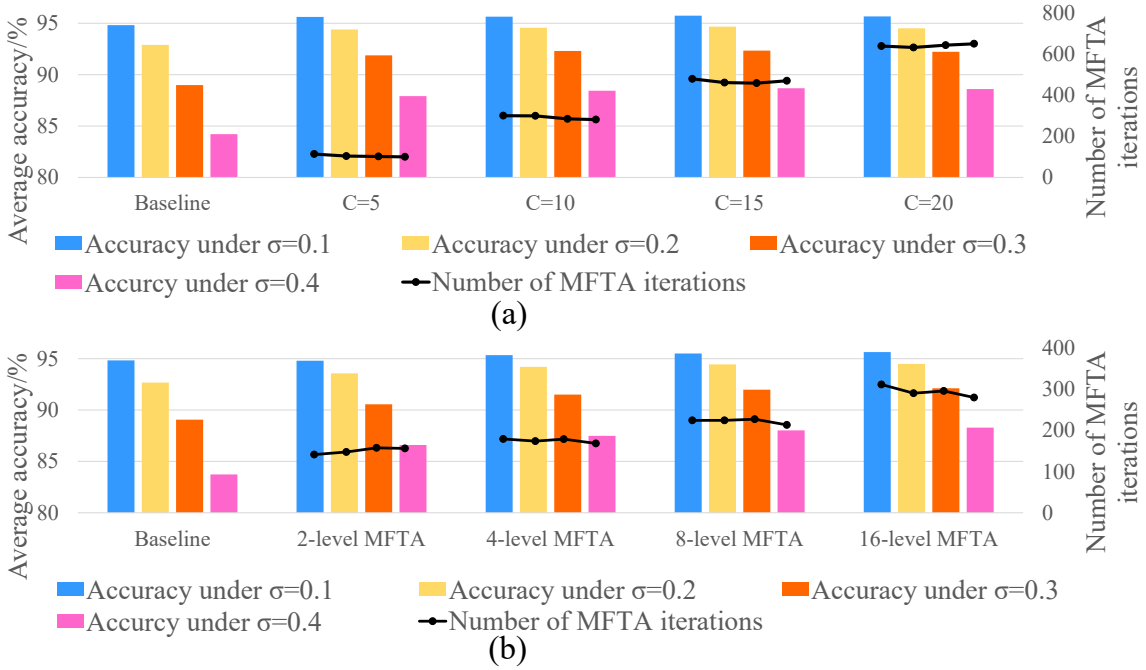


FIGURE 4.7: Tests of (a) C and (b) $2N$ affecting the recovering inference accuracy (higher is better) and how many iterations that MFTA needs (lower is better).

FIGURE 4.7(a) presents the results. The "baseline" group refers to the inference accuracy with the process variation injection and without applying MFTA. As C increases, the iteration number of MFTA (black dots) increases in an approximately linear relation,

and shows no correlation to the process variations. This trend validates that C is the knob to control the MFTA iterations. A larger C induces more iterations which imply higher compensation overhead.

The inference accuracy (bars) is boosted by applying MFTA compared to the baseline. With a larger C, MFTA leads to better accuracy given the process variation, and there is a clear saturation of accuracy recovery as C keeps increasing. Therefore, there is a sweet point that balances the iteration number and inference accuracy. For our pre-trained model, we set $C = 10$ for both satisfactory accuracy recovery and relatively small hardware costs.

Dependency on $2N$: $2N$ is the number of available levels for V_{th} candidates. A larger $2N$ value means finer granularity of MFTA scheme, projecting to more precise MFTA tuning. Yet more iterations are required to complete MFTA since the step change of the threshold voltage in each iteration is smaller.

The initial global voltage threshold of MFTA is set to 400mV based on the original circuit-level configuration, and the range of the available threshold levels is set to $260\text{mV} \sim 560\text{mV}$. FIGURE 4.7(b) shows how the average accuracy and total MFTA iteration number change with the number of threshold levels under different process variation setups. We select the number of threshold levels to be 2 (160mV interval), 4 (80mV interval), 8 (40mV interval), 16 (20mV interval). The average accuracy after applying MFTA increases with more threshold levels from the baseline accuracy. Under these four process variations σ, the accuracy after applying MFTA increases significantly when the number of threshold levels jumps from 2 to 4, whereas gains only marginally in the cases over 4. Besides, the numbers of iterations to complete MFTA in all the process variation cases increase from ~ 150 to ~ 300. This validates the trend that finer granularity of MFTA levels ($2N$) would cause more iterations. In this set of experiments, the results also show that the number of iterations to complete MFTA has no correlation to process variations. Based on these observations, we set the default $2N$ parameter as 4 levels to achieve satisfactory accuracy recovery as well as small hardware cost.

FIGURE 4.8 shows the Monte Carlo simulation to validate the efficacy of MFTA scheme.

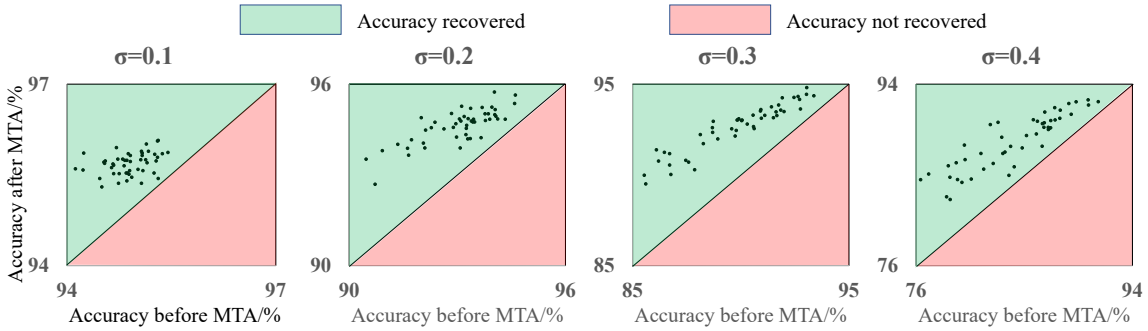


FIGURE 4.8: Accuracy recovered by MFTA under various process variations. x- and y-axis are the percentage accuracy before and after applying MFTA, respectively.

It is the result of 50 experiments under different process variation levels, including the accuracy before threshold adjustment (x-axis) and the accuracy after threshold adjustment (y-axis), under the aforementioned MFTA configuration. As shown, all the experiments under $\sigma = 10\% \sim 40\%$ obtain accuracy improvement. On average, ASTERS remains at 95.3% accuracy under 10% variations, and at 94.2% accuracy under 20% variations.

4.5.3 Trade-off in TTA

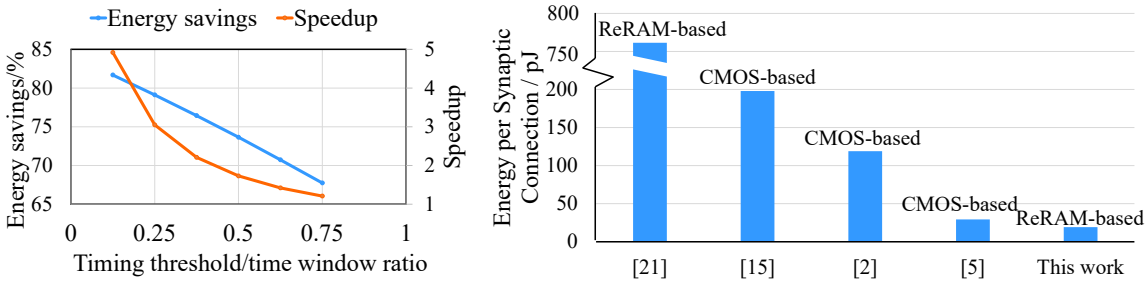


FIGURE 4.9: (a) Energy savings and speedup vs. *timing threshold*. (b) The energy comparison with ReRAM-based and CMOS-based counterparts.

We investigate the trade-off between the energy savings and speedup of TTA scheme and the accuracy degradation under different *timing threshold* setups. FIGURE 4.9(a) shows the trade-off between energy savings and speedup. Notably, the x-axis is the ratio of timing threshold to time window size. The baseline is the proposed twin-column PE without TTA. Smaller *timing threshold* results in faster inference as well as better energy savings. FIGURE 4.10 shows the accuracy degradation induced by TTA. When the *timing*

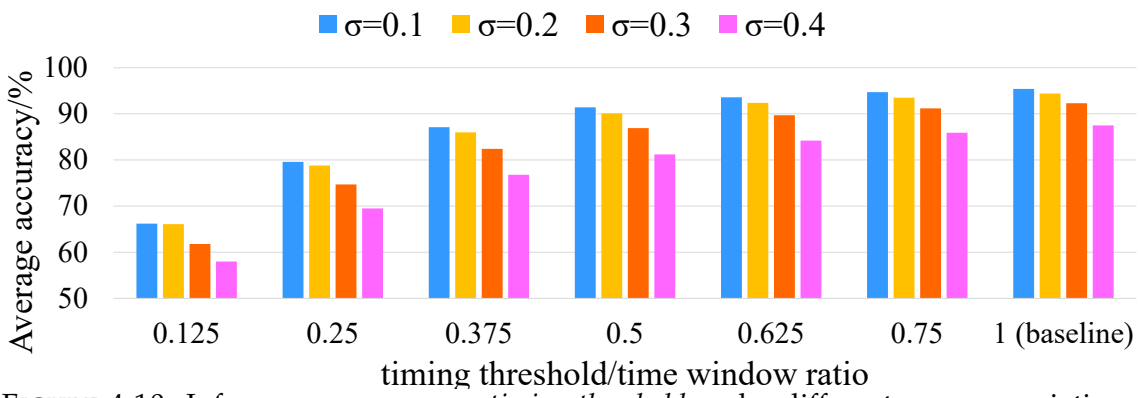


FIGURE 4.10: Inference accuracy vs. *timing threshold* under different process variations. Notably, the x-axis is the ratio of timing threshold to time window size.

threshold is half of the *time window size*, ASTERS remains above 91.4% accuracy under 10% variations, and above 90.1% accuracy under 20% variations. Under this *timing threshold* setup, more than 1.7× speedup and 73.7% energy savings can be achieved. As such, we determine the default setup of "*timing threshold* = 0.5× time window size = 1.28μs" for the proposed TTA scheme.

4.5.4 Energy & Speed

FIGURE 4.9(b) presents the energy comparison of ASTERS with SoTA ReRAM-based spiking neuromorphic designs [60] and CMOS-based spiking neuromorphic design counterparts [47, 3, 9]. To be fair, the energy results are scaled to the 180 nm-1.8 V process. We adopt the same TTA scheme setup as in Section 4.5.3. In order to make the metric independent from SNN models, we choose the "energy per synaptic activation" as the figure of merit in the comparison, which represents the average energy consumption for one synaptic connection [3]. The inter-crossbar data communication in all the counterparts is not taken into account because we focus on evaluating our proposed macro-level circuits. FIGURE 4.9(b) indicates that ASTERS achieves 97.5% energy reduction versus the ReRAM-based baseline [60]. It achieves over 83.9% energy reduction compared to the CMOS-based mixed-signal baselines [47, 3] and 34.7% energy reduction compared to the digital baseline design [9]. The energy reduction mainly comes from the energy efficient circuit designs and TTA scheme. The inference latency of ASTERS is 1.48μs per frame,

including a 200ns relaxation stage for power-gating.

4.6 Conclusion

Facing the fast-evolving field of neuromorphic computing hardware design, this work aims to enable the highly efficient spike encoding scheme such as TTFS with promising emerging memory technologies. We propose and evaluate ASTERS, an adaptable threshold spike-timing neuromorphic design with twin-column ReRAM synapses. Specifically for the TTFS encoding scheme and the proposed novel hardware substrate, we introduce the MFTA methodology to improve the resilience against ReRAM device process variations and the TTA methodology to exploit the advantage of event-driven SNN execution models. These two methods enable highly efficient execution of SNN inference. Experiment results show that ASTERS achieves more than 34.7% energy savings compared to existing spiking neuromorphic designs. It can also obtain 90.1% accuracy even under 20% process variations.

5. Low-Latency In-Sensor-Intelligence Design With Neuromorphic Spiking Neurons

Edge computing vision systems bring the computational power and intelligence closer to the data source. By performing cognitive tasks directly on edge devices, these systems enable real-time analysis, decision-making, and action without the need for constant data transmission to a centralized server. This not only reduces latency and bandwidth requirements but also enhances privacy and security by processing sensitive data locally. Edge computing vision systems have demonstrated great potential of enabling rapid, decentralized, and intelligent visual processing. At the core of edge computing vision systems are embedded processors for image processing computation. ReSiPE and ASTERS introduced in the last two chapters play the role of spiking neuromorphic processors for conventional DNNs and SNNs, providing an energy solution to the efficient image processing. Besides, the spike-based data format is suitable for the dynamic vision sensor (DVS) that captures the change of light environments and generates spike events instead of generating image frames. A complete vision system cannot perform computation without perception. Another critical component is the image sensor that generates raw visual data for the subsequent processors to execute analysis and recognition tasks. No matter whether conventional frame cameras or DVSs are utilized, the inevitable data transmission between sensors and processors induce latency and bandwidth issues that limit the overall performance of vision systems.

In-sensor-processing (ISP) paradigm has been exploited in state-of-the-art vision system designs to pave the way towards power-efficient sensing and processing. The redundant data transmission between sensors and processors is significantly minimized by local computation within each pixel. However, existing ISP designs suffer from limited frame rates and degraded fill factors. In this chapter, we introduce a low-latency in-sensor-intelligence neuromorphic vision system using neuromorphic spiking neurons, namely SpikeSen. SpikeSen directly operates on the photocurrents and executes the computation in the frequency domain, reducing the long exposure time and speeding up the computation. Experiments

show that SpikeSen can achieve more than 6.1× computation speedup compared to existing ISP designs with competitive energy consumption per pixel.[1]

5.1 Introduction

The proliferating edge computing techniques greatly reduce the energy consumption and latency of data transmission by moving computation resources close to source data in low-power real-time applications such as Internet-of-Things (IoT), portable devices, robots, etc. Vision systems that incorporate image sensors and processors are critical components in edge computing devices. The limited power budget and latency requirement have triggered a series of near-sensor-processing (NSP) vision system designs for image processing applications [11, 35, 64, 5]. These designs place low-power processors near the image sensor on a single chip to reduce the physical distance between the raw image data and the processing elements. Nevertheless, the power-hungry analog-to-digital converters (ADCs) between sensors and processors still hinder the vision systems from pursuing higher energy efficiency and performance.

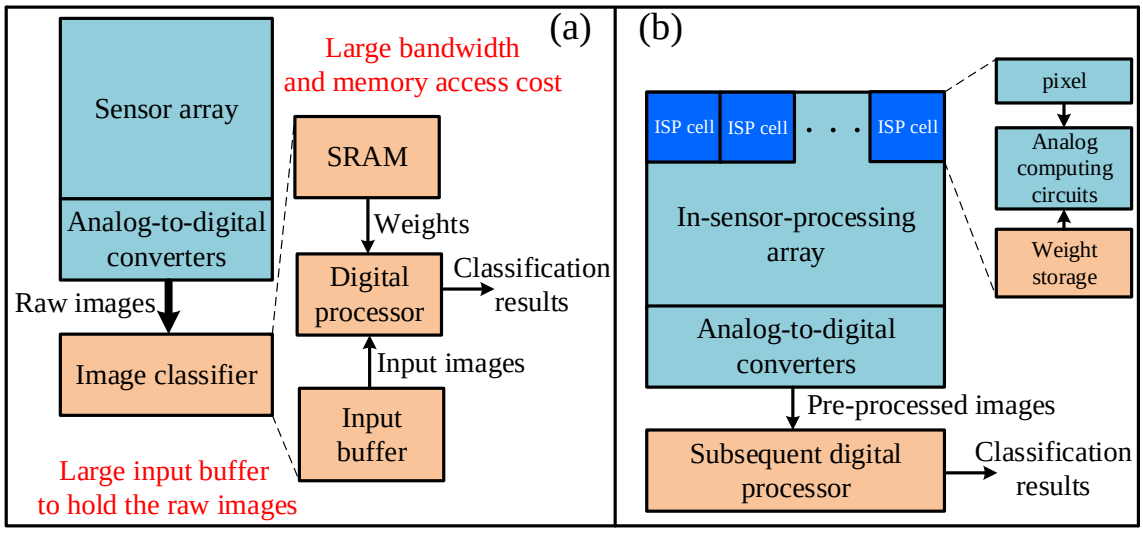


FIGURE 5.1: Conventional vision system design (a) and ISP design (b).

ISP paradigm recently gains attention owing to its capability of building ultra-low-power

[1] Ziru Li et al. "Spikesen: Low-latency in-sensor-intelligence design with neuromor- phic spiking neurons". In: IEEE Transactions on Circuits and Systems II: Express Briefs (2023). © 2023 IEEE.

vision systems targeted on always-on image processing applications. In the ISP designs [61, 63, 62, 40, 44], processing elements are integrated within the pixels of the CMOS sensors. As shwon in FIGURE 5.1, the processing element directly fetches the analog outputs from sensor pixels and performs image pre-processing computation locally, such as multi-layer perceptrons (MLPs), support vector machines (SVMs), or the 1^{st} layer of binary neural networks (BNNs). This design concept reduces the considerable power of analog-to-digital conversion, saves memory access cost and bandwidth between sensors and processors, and avoids the need for large buffers for raw images. Nevertheless, unresolved drawbacks still exist in prior ISP designs, including the degraded fill factor and the long exposure time.

Advances in neuromorphic engineering, inspired by the computation mechanisms of human brains, have led to a generation of large-scale spike-based processors for cognitive computing. Representative neuromorphic designs [3, 1, 9] perform spike-based computation in which the data are represented by the temporal information of spike trains. With simple synapse and neuron circuit implementations, neuromorphic computing hardware becomes a promising solution to pursuing higher power efficiency. Conventional spike-based image sensors [57, 50] transforms the photocurrent values to the spike frequency or timing in each pixel circuit, but they fail to implement computation associating multiple pixels. Developing a scheme which benefits from the direct integration of ISP and neuromorphic computing hardware becomes a valuable research topic.

To overcome the challenges in existing ISP vision systems, we propose a low-latency in-sensor-intelligence design with neuromorphic spiking neurons, namely SpikeSen. To our best knowledge, SpikeSen is the first ISP design that directly processes the photocurrents and computes the 1^{st} layer of BNNs in the spike frequency domain. The main contributions in this chapter include:

1. We propose a spike-based computing pixel (SCP) and SCP string circuit that leverages the CMOS-based neuromorphic spiking neuron and capacitive synaptic weight to process the photocurrent locally within each pixel and accumulate the partial results from adjacent pixels.

2. Based on the proposed SCP circuit substrate, we design SpikeSen, a low-latency ISP vision sensor with neuromorphic spiking neurons. SpikeSen executes convolution in the frequency domain with extremely low latency.
3. We optimize the mapping scheme and control flow of SpikeSen with a novel sub-SCP structure. The parallel operations of multiple sub-SCPs further simplify the control flow and speed up the computation.

5.2 Challenges in In-sensor-processing Designs

In the conventional vision systems, the data transmission dominates the overall energy consumption and latency. As reported in [18], nearly half of the overall energy consumption of a vision system is caused by accessing the model parameters and the input images. This power bottleneck triggers the NSP and ISP design paradigm that integrates the processing units, memory, and sensing pixel array on a signal chip to reduce the physical distance between processing units and data, including the raw image data and the model parameters.

There exist NSP designs [11, 64, 5] that deploy the processors close to the pixel array. ISP designs [44, 63, 62] go further by fusing the processing units with the pixels. Unresolved drawbacks exist in the prior CMOS-based ISP designs. First, the local analog processing element deployed within each pixel contains several memory cells to store model parameters, and thus the fill factor is degraded. For example, in [63], a register is added in each processing circuit to store weights. In [62], an SRAM macro that contains 16 SRAM cells is included in each in-pixel processing circuit. Second, the computation in the prior designs requires a long exposure time to accumulate photocurrents, which leads to a long computation latency. Third, the prior designs still require power-consuming ADCs to convert analog outputs to the digital domain.

5.3 Neuromorphic vision sensor design

In this section, we introduce SpikeSen from the basic circuit substrate to the top architecture. We start from the unit cell in SpikeSen named spike-based computing pixel (SCP) in Section 5.3.1. Then we introduce how to build the interconnects between differ-

ent SCPs in Section 5.3.2. Based on the proposed circuit structure, we present our proposed spike-based neuromorphic vision sensor design in Section 5.3.3.

5.3.1 Spike-based computing pixel

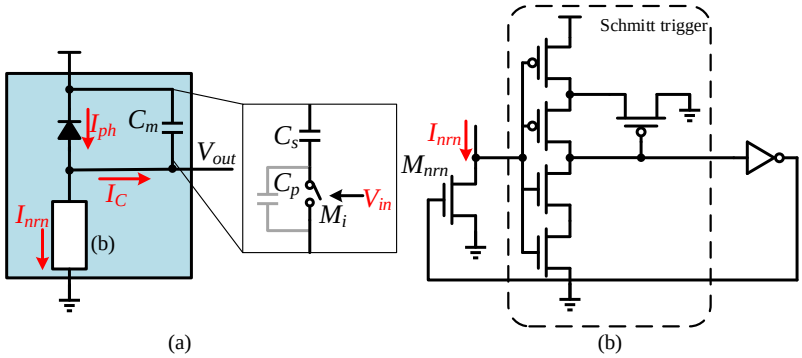


FIGURE 5.2: (a) The structure of our proposed spike-based computing pixel (SCP). (b) The structure of the neuromorphic spiking neuron in SCP.

The proposed SCP computes the product between the pixel value in the format of photocurrent and the weight value programmed to a capacitor. FIGURE 5.2(a) depicts the structure of our proposed SCP. In each SCP, the "programmable capacitor" C_m is in parallel with the photodiode (PD) and in series with the spiking neuron. The spike neuron consists of a transistor M_{nrn} in parallel with a Schmitt trigger that controls the gate voltage of M_{nrn}. We adopt the Schmitt trigger designed in [15] due to its simplicity, as shown in FIGURE 5.2(b).

Principle of SCP: Assume the current flowing through the spike neuron is I_{nrn}, the photocurrent is I_{ph} and the current charging C_m is I_C. One can derive the equations:

$$I_{nrn} + I_C = I_{ph}; I_C = C_m \frac{dV_{out}}{dt}. \tag{5.1}$$

Assume the high/low threshold voltages of the Schmitt trigger in the spiking neuron are V_{thH} and V_{thL}, respectively. When V_{out} increases from 0 and is lower than V_{thH}, M_{nrn} is cut off. I_{nrn} is much smaller than I_{ph}, leading to $I_C = I_{ph}$. I_C keeps charging C_m until $V_{out} = V_{thH}$, toggling the output of the Schmitt trigger. M_{nrn} is then switched

on with $I_{nrn} >> I_{ph}$, leading to a large discharging current $I_C = -I_{nrn}$ that decreases V_{out} to V_{thL} instantly, based on Equation (5.1). Then M_{nrn} is turned off again. This charging/discharging loop generates the oscillating V_{out}, and its frequency is proportional to I_{ph} since the charging phase ($I_C = I_{ph}$) is dominant in the charging/discharging loop.

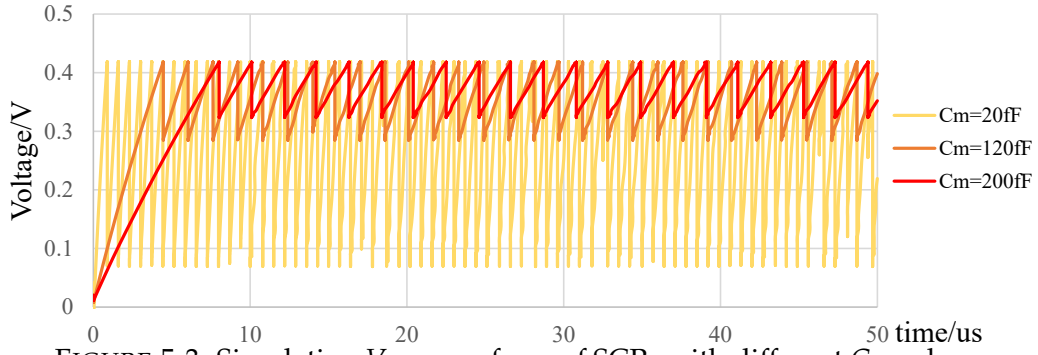


FIGURE 5.3: Simulation V_{out} waveform of SCPs with different C_m values.

In an SCP, the output oscillating frequency is not only proportional to the input photocurrent I_{ph}, but also inversely proportional to the capacitance C_m since $\frac{dV_{out}}{dt} = \frac{I_C}{C_m}$ during the charging phase. As shown in FIGURE 5.3, larger C_m leads to smaller V_{out} oscillation frequency. In our SCP design shown in FIGURE 5.2(a), C_m consists of a transistor switch in series with a capacitor C_s and in parallel with a small capacitor C_p. The binary weight is represented by the gate voltage of the transistor switch M_i, denoted as V_{in}. When V_{in} is low and M_i is switched off, C_p leads to low capacitance state (LCS) of C_m. When M_i is switched on, C_p is bypassed and C_s leads to high capacitance state (HCS). By programming the weight to $\frac{1}{C_m}$, the output oscillating frequency of SCP will represent the product of the photocurrent strength and the weight.

5.3.2 SCP string for MAC operation

The proposed SCP implements multiplication within each pixel unit in the frequency domain. Prior ISP designs [63, 62] require a long exposure time ($\sim 1ms$) to accumulate the photocurrents from multiple cells on the capacitors and convert the voltages to the digital domain as the computation results. Substituting the slow photocurrent accumulation with high-frequency oscillation greatly shortens the computation latency and boosts the perfor-

mance. However, the connections between multiple pixels are also necessary to accumulate the multiplication results from different pixels.

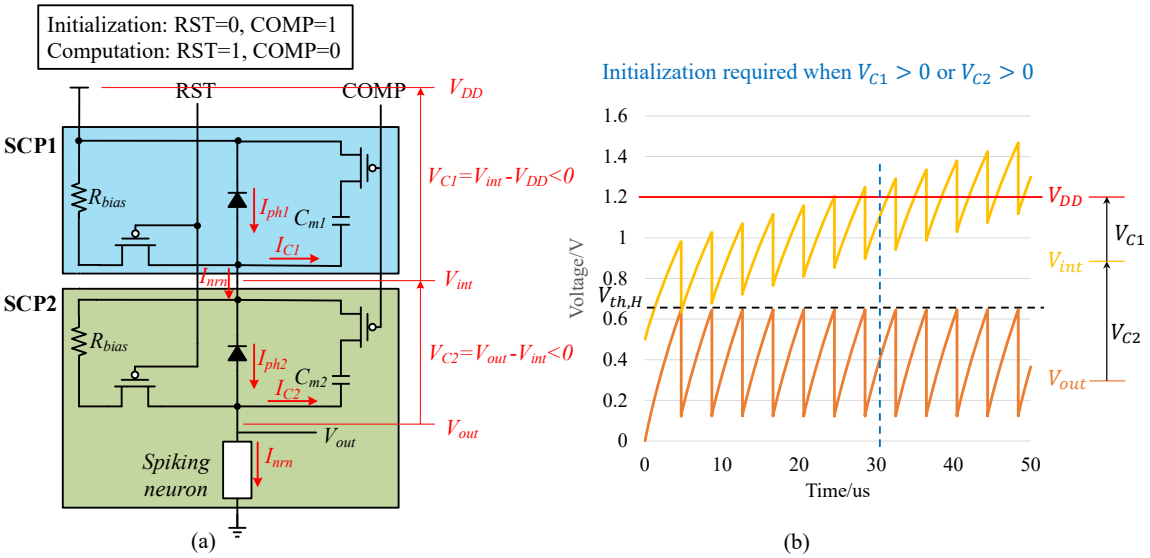


FIGURE 5.4: (a) The structure of a 2-cell SCP string for MAC operation. (b) The simulation result of a 2-cell SCP string oscillating waveforms.

To support multiply-and-accumulate (MAC) operations with our basic SCP design, we propose a structure called SCP string. FIGURE 5.4(a) shows the structure of a 2-cell SCP string that consists of two basic SCP cells (SCP1 and SCP2) in series as a simplified example. SCP1 is connected to the power supply V_{DD}. The output is generated by the spiking neuron in SCP2. There is an additional initialization circuit including two bias resistors R_{bias}, which will be interpreted hereinafter. Since these two SCPs are connected in series, the following equations can be derived:

$$\begin{aligned} &I_{nrn} + I_{C1} = I_{ph1}; I_{C1} = C_{m1}\frac{dV_{C1}}{dt}; \\ &I_{nrn} + I_{C2} = I_{ph2}; I_{C2} = C_{m2}\frac{dV_{C2}}{dt}; \\ &\frac{dV_{out}}{dt} = \frac{dV_{C1}}{dt} + \frac{dV_{C2}}{dt}; \end{aligned} \tag{5.2}$$

where V_{C1} and V_{C2} are the voltages across two SCPs. The series connection enables the oscillation to be modulated by both SCPs, i.e. $\frac{dV_{out}}{dt} = \frac{I_{C1}}{C_{m1}} + \frac{I_{C2}}{C_{m2}}$.

The oscillation behavior in the SCP string is similar to the oscillation in a single SCP. When V_{out} increases from 0V, I_{nrn} is much smaller than I_{ph1} and I_{ph2}. Thus $I_{C1} = I_{ph1}$

and $I_{C2} = I_{ph2}$, leading to $\frac{dV_{out}}{dt} = \frac{I_{ph1}}{C_{m1}} + \frac{I_{ph2}}{C_{m2}}$. In this way, the increasing speed of V_{out} can represent the MAC result $\frac{I_{ph1}}{C_{m1}} + \frac{I_{ph2}}{C_{m2}}$. When V_{out} reaches the threshold voltage of the Schmitt trigger V_{thH}, I_{nrn} becomes much larger than I_{ph1} and I_{ph2}, decreasing V_{out} instantaneously. The oscillation frequency is proportional to the MAC result between the input vector $[I_{ph1}\ I_{ph2}]$ and the weight vector $[\frac{1}{C_{m1}}\ \frac{1}{C_{m2}}]$. Following the same design principle, an n-cell SCP string that consists of n basic SCP cells in series can compute the MAC results of n-element input vectors and weight vectors. The SCP string enables the accumulation of the products calculated in multiple SCP cells in the frequency domain. The SCP cells are connected in series so that the voltage changes of all the SCPs are summed up. This overall voltage change is reflected by the change of output voltage V_{out}. Thus, the oscillation frequency of V_{out} represents the overall MAC result.

Weight representation and bias cancellation: As interpreted hereinbefore, the weight in each SCP is represented by $\frac{1}{C_m}$. However, for the implementation of binary neural networks, even if the -1 weights are represented by $\frac{1}{C_{m,HCS}}$ where $C_{m,HCS}$ is the high capacitance state value, they still positively contribute to the output oscillation frequency. To cancel this positive weight bias and enable ± 1 weights, an additional cycle is required to program all the capacitance values inversely. The final output should be the difference between the output frequency in the computation stage and the output frequency in this *bias cancellation* stage, i.e.

$$\begin{aligned} Output &= f_{comp} - f_{bc} \\ &= \sum_{n=1}^{N} I_{ph,n} \frac{1}{C_{m,n}} - \sum_{n=1}^{N} I_{ph,n} \frac{1}{C'_{m,n}} \\ &= \sum_{n=1}^{N} I_{ph,n} (\frac{1}{C_{m,n}} - \frac{1}{C'_{m,n}}), \end{aligned} \tag{5.3}$$

where $C_{m,n}$ and $C'_{m,n}$ are C_m values in the $n-th$ SCP in the SCP string during the computation stage and *bias cancellation* stage, respectively. $C'_{m,n}$ is $C_{m,HCS}$ while $C_{m,n}$ is $C_{m,LCS}$, or vice versa. In this case, $+1$ weights are represented by $\frac{1}{C_{m,LCS}} - \frac{1}{C'_{m,HCS}}$ and -1 weights are

represented by $\frac{1}{C_{m,HCS}} - \frac{1}{C'_{m,LCS}}$. Such a differential computing paradigm also contributes to better noise immunity.

The initialization of SCP string: The PDs should work in reverse bias. In an SCP string, we need to guarantee that the PD in each SCP is in reverse bias, which means $V_{C1} < V_{bias} < 0$ and $V_{C2} < V_{bias} < 0$. Hence, a bias resistor R_{bias} is added in parallel with PD and C_m in each SCP. The voltages of the internal nodes between adjacent SCPs are distributed evenly between V_{DD} and ground by the bias resistor string during the initialization stage, reversely biasing all the PDs. FIGURE 5.4(b) shows the simulation result that manifests the oscillation behavior of the internal node voltage in a 2-cell SCP string in FIGURE 5.4(a). Both the output voltage V_{out} and the internal node voltage V_{int} oscillate at the same frequency. It can be observed that V_{int} gradually approaches V_{DD} when the oscillation proceeds, leading to decreasing bias voltage of the PD in SCP1. To guarantee the reverse bias of PDs throughout the operation, a periodic initialization is required to prevent any two internal node voltages in the SCP string from intersecting each other. The initialization period depends on the values of the photocurrent I_{ph} and C_m, as well as the total voltage supply of the SCP string V_{DD}.

5.3.3 Spike-based neuromorphic vision sensor

5.3.3.1 Fundamental SpikeSen architecture

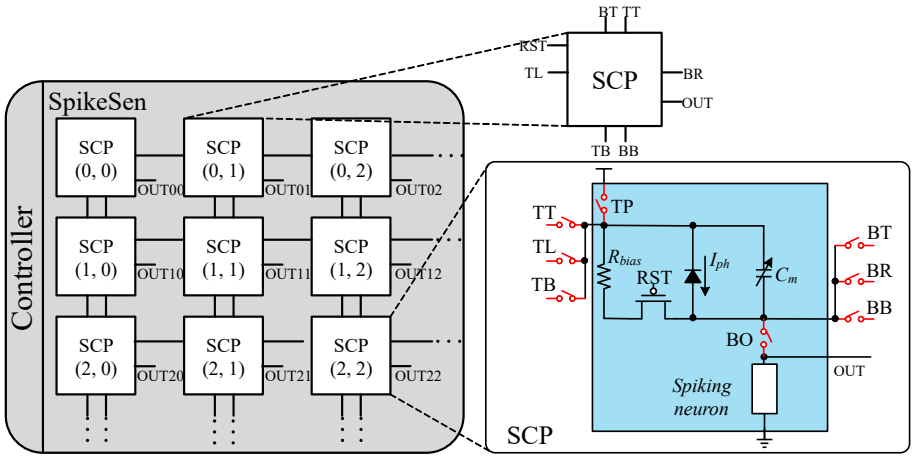
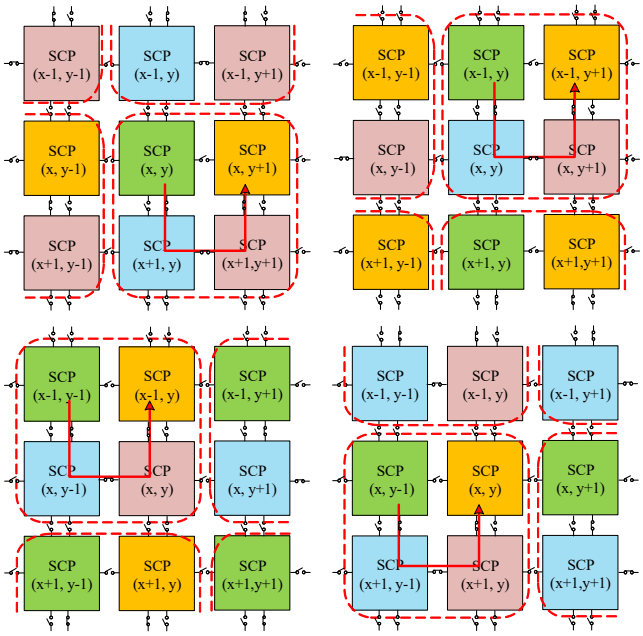


FIGURE 5.5: The overall architecture of SpikeSen.



MODE 1: Multiplied by w(0,0)

MODE 2: Multiplied by w(1,0)

MODE 3: Multiplied by w(0,1)

MODE 4: Multiplied by w(1,1)

(b)

	Mode 1	Mode 2	Mode 3	Mode 4
TP	ON	OFF	OFF	OFF
TT	OFF	ON	OFF	OFF
TL	OFF	OFF	OFF	ON
TB	OFF	OFF	ON	OFF
BT	OFF	OFF	OFF	ON
BR	OFF	ON	OFF	OFF
BB	ON	OFF	OFF	OFF
BO	OFF	OFF	ON	OFF

FIGURE 5.6: (a) The LRCC formation to perform a 2×2 convolution. (b) The ON/OFF state of the 8 interconnects in each SCP under the 4 interconnect modes.

The fundamental architecture overview of SpikeSen is shown in FIGURE 5.5. To enable the formation of different SCP strings within SpikeSen, interconnects between adjacent SCP cells are necessary. Take 2×2 convolution as an example. A 4-cell SCP string executes the MAC result in a 2×2 window. We adopt a "Left-to-right column-by-column (LRCC)" interconnect scheme to form an SCP string with 4 SCPs, as shown by the red arrows in FIGURE 5.6(a). In each 4-cell SCP string, the 4 SCP cells have 4 different interconnect modes and modulate the photocurrents with different weights. For example, the top-left SCP is connected to V_{DD} and the bottom-left SCP; the top-right SCP is connected to the bottom-right SCP and output the MAC results. In total, 8 controllable interconnects in each SCP are required to form the LRCC SCP string, as highlighted in FIGURE 3.4(a). Note that when the adjacent SCPs are connected, the interconnects are merged. For example, the TL of SCP(x, y+1) is the same interconnect as the BR of SCP(x, y). FIGURE 5.6(b) shows the ON/OFF state of the 8 interconnects under the 4 interconnect modes. Note that the 8 interconnects can support larger convolution kernels under our LRCC interconnect scheme. The difference lies in how to control the interconnects according to the position of

each SCP in the SCP string.

To slide multiple kernels on the image and generate multiple output feature map channels, C binary weights are programmed to C_m in each SCP, where C is the output channel number. These C binary weights are at the same position of different kernels in a BNN layer. C output feature map channels are generated sequentially. Since for each output channel, the control flow is identical, we will concentrate on the single-channel convolution hereinafter.

5.3.3.2 Sub-SCP structure optimization

In the fundamental SpikeSen architecture, the entire SCP array is divided into multiple SCP strings that work in parallel with the same kernel. When one cycle of computation is completed, the interconnect mode of each SCP should be switched to form different SCP strings, which is equivalent to sliding the kernel over the image. Besides, the weights are also supposed to be shifted on the SCP array. The total cycle number of a single-channel convolution computation depends on the number of interconnect modes, e.g. 4 cycles for 2 $\times$ 2 convolution. Both the interconnect mode and the weight of each SCP need switching when one cycle of computation is completed.

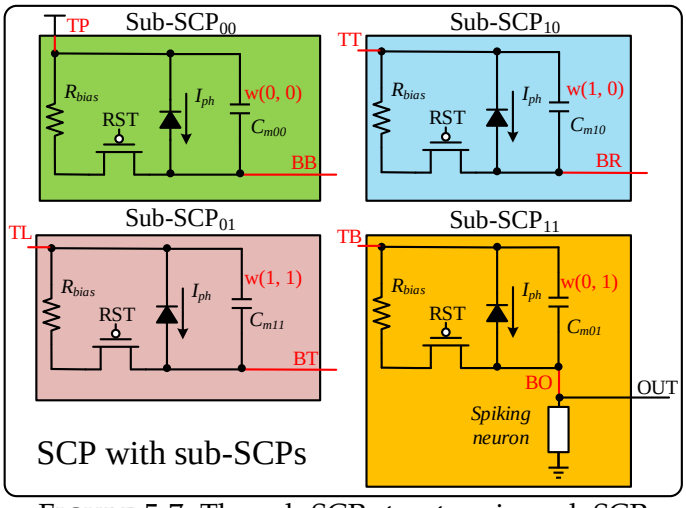


FIGURE 5.7: The sub-SCP structure in each SCP.

The controllable interconnects using CMOS switches inject additional noises to the

oscillating voltage output and degrade the computation accuracy. Therefore, we propose a novel sub-SCP structure of SCP that can implement the kernel sliding in a more efficient way than the fundamental SCP in FIGURE 3.4(a). The proposed sub-SCP structure of an SCP is shown in FIGURE 5.7. For a 2×2 convolution, the original SCP is divided into four sub-SCPs, and each sub-SCP contains two interconnects, one programmable C_m and one PD, respectively. All the four sub-SCPs in one SCP sense the same image region, and work simultaneously under one of the four interconnect modes respectively. The spiking neuron is added to the sub-SCP at the output side of the SCP string.

The benefits from the sub-SCP design are two-fold. First, since the interconnects are distributed to different sub-SCPs, no CMOS switches are needed to control the ON/OFF state of the interconnects. The noises caused by CMOS switches are avoided. Second, the weights in one convolution kernel are mapped to different sub-SCPs of a SCP. Since the sub-SCPs work in parallel under different interconnect modes, the computation of a single-channel convolution is completed in one cycle instead of multiple cycles in the fundamental architecture, thus reducing the total computation latency and boosting the throughput.

5.4 Experiments

5.4.1 Experiment setup

We perform the post-layout simulation of the proposed circuits in Cadence Virtuoso with TSMC 65nm PDK. The size of the SCP array is set to 32×32. In the evaluated structure, each SCP contains 4 sub-SCPs to perform a 2×2 convolution layer. Since the oscillation frequency of spiking neuron reaches above $2MHz$, the computation stage is set to $5us$ for each output channel, and the total computation latency of each output channel is $10us$ including the *bias cancellation* stage. The initialization period is set to $5us$.

5.4.2 Computational accuracy analysis

FIGURE 5.8 shows the relationship between the oscillation frequency of a 4-cell SCP string and the expected MAC results after *bias cancellation*. Here we simulate with 4-cell SCP string to verify the computation results of a 2×2 convolution. In the simulation,

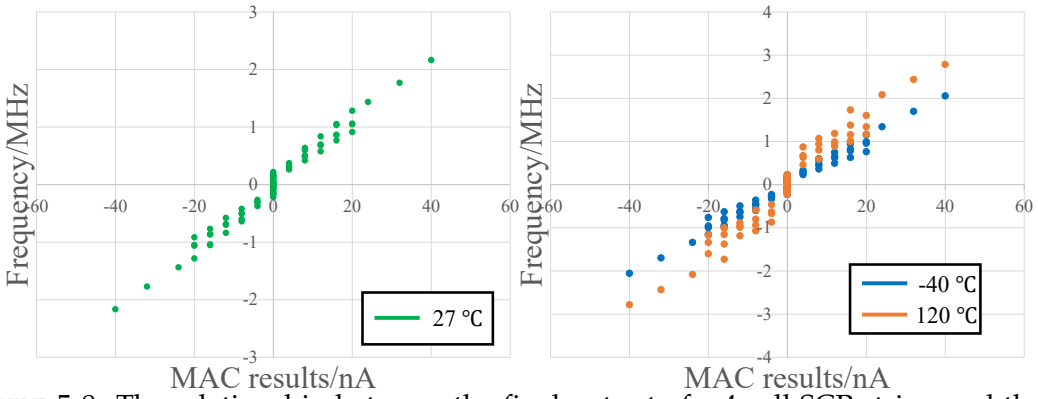


FIGURE 5.8: The relationship between the final output of a 4-cell SCP string and the expected MAC results after *bias cancellation*.

LCS/HCS are set to 20fF/200fF. The photocurrents range from $0 \sim 10nA$ with 6 levels. The initialization circuit is also included in the simulation. Process variations with a standard deviation of 10% are injected to C_m in each SCP. A $\pm 5\%$ voltage fluctuation is applied to the power supply of the SCP string. We perform Monte Carlo simulation with 160 data points under 27°C, 80 data points under -40°C and 80 data points under 120°C. The expected MAC results of 4 capacitive weights and 4 photocurrent-based inputs are depicted by the X-axis. The final output frequency depicted by the Y-axis is linearly proportional to the expected MAC results, indicating the capability of the proposed SCP structure to perform MAC operations. The oscillation frequency of SpikeSen output increases with the temperature. The linearity and immunity to process variations are degraded under high temperature (120°C). The measured SNR under room temperature is 18.2dB.

5.4.3 Energy, performance and area comparison

Table 5.1 shows the comparison between SpikeSen and existing NSP designs [64, 5] and ISP designs [44, 63, 62]. We scale the latency of each design according to a 2×2 convolution layer with 6 output channels. We select energy per frame per pixel as the Figure of Merit for fair energy efficiency comparison and scale the data to $65nm/1.2V$ node.

SpikeSen achieves at least $6.1\times$ latency reduction over existing NSP designs [64, 5] and ISP designs [44, 63, 62]. SpikeSen consumes 22.06pJ pixel-wise energy per frame, which achieves at least 66.8% reduction compared to the digital and mixed-signal ISP counter-

Table 5.1: Comparison with existing NSP and ISP designs

Work	**This work**	[44]	[64]	[5]	[63]	[62]
Process	$65nm$	$180nm$	$90nm/40nm$	$65nm$	$65nm$	$180nm$
Voltage (V)	**2.5/1**	3.3/1.8	3.3/2.9/1.8/1.1	2.5/0.5∼0.8	1.2	0.8∼1.8
Array size	32×32	64×64	1296×976	320×240	32×32	32×32
Pixel size ($\mu m \times \mu m$)	30×30	28.8×28.8	3.5×3.5	7×7	20×20	35×35
Fill factor	28.4%	18.32%	N/A	N/A	14%	9.14%
Latency	$60us$	N/A	N/A	N/A	$368us$	$9.6ms$
FoM[a].	**22.06**	14.1	66.5	73.7	13	∼0.2

[a] $FoM = P_{pixel} \times t_{frame}$, where P_{pixel} represents the average power of SCP and t_{frame} represents the latency of each frame ($60us$). The unit of FoM is pJ/pixel·frame

parts [64, 5]. The extremely low computation latency of SpikeSen stems from the high-frequency-domain computation performed by SCPs. Because the oscillation is generated by charging and discharging the capacitor, SpikeSen also features minor static power dissipation during the oscillation. SpikeSen does not defeat [63, 62] in terms of energy per pixel. However, it should be noted that these designs generate binary MAC results by sensing the differential voltage outputs. To achieve competitive precision with SpikeSen, they require power-hungry ADCs to sense differential voltage as well as analog peripherals that assist the computation, leading to considerable power consumption and area overhead.

5.5 Conclusion

In this chapter, we propose SpikeSen, a low-latency in-sensor-intelligence vision sensor design with neuromorphic spiking neurons. SpikeSen harnesses the novel spike-based computing pixel design and performs the computation in the frequency domain. SpikeSen significantly outperforms prior designs with more than 6.1× computation speedup and competitive energy consumption.

6. Fabrication of an SRAM-based PIM Neuromorphic Processing Engine for TTFS-based SNN Inference

ASTERS demonstates a ReRAM-based PIM solution to a neuromorphic processing engine for TTFS-based SNN inference. Compared the ReRAM-based PIM, SRAM-based PIM demonstrates less vulerability to the device process variations, whereas the density of weight storage degrades. While migrating the “twin-column synapse" design concept of ASTERS to an SRAM-based design, the critical issue is how to map multi-bit synaptic weights to the SRAM array.

This chapter presents a PIM-based neuromorphic processing engine for TTFS-based SNN inference, which comprises a 64x64 8T-SRAM array for synapse storage and current-based post-neuron circuits. We propose to map the positive and negative synaptic weights to adjacent columns in the SRAM array and accumulate the membrane potential in the dedicated post-neuron circuits. Featuring the power-efficient 2-rail post-neuron circuit design, the proposed processing engine achieves the energy efficiency 249.8 TOPS/W under 8-bit inputs and signed 4-bit weights. We fabricate TFSRAM processing engine in 65nm technology node and provide measurement results in this chapter.

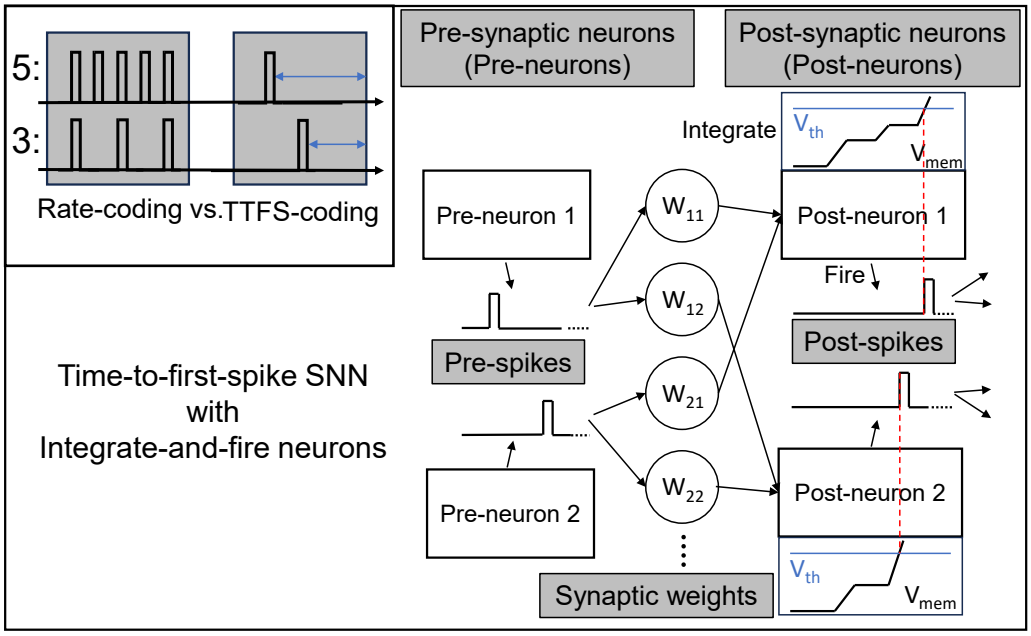


FIGURE 6.1: Time-to-first-spike SNN with integrate-and-fire neurons.

6.1 Introduction

As, indicated hereinbefore, prior PIM SNN neuromorphic chips [66, 28], which utilize the firing rates of spike trains to represent data, suffer from prolonged time window for spike counting to maintain sufficient accuracy. Temporal-coded SNNs, especially TTFS-based SNNs, outperform rate-coded SNNs by using the firing time of fewer spike events (down to a single spike in TTFS-based SNNs) to convey equivalent information efficiently. In particular, TTFS coding is applied to SNNs to reduce the spike number to one. We show the structure of a TTFS-based SNN we want to implement again in FIGURE 6.1.

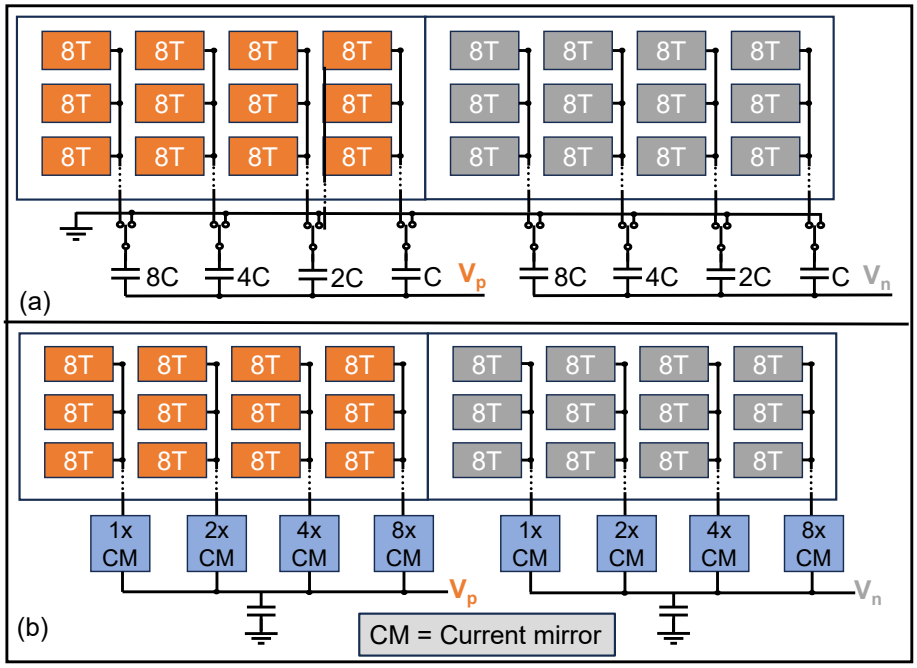


FIGURE 6.2: Comparison between (a) prior capacitor-based neurmorphic macros [28] and (b) current-based TFSRAM.

In this chapter, we propose TFSRAM, a TTFS PIM processing engine which comprises a 64x64 8T-SRAM array for twin-column synapse storage, and DAC-free current-based post-neuron circuits to implement the IF neuron model. The proposed engine achieves 249.8 TOPS/W energy efficiency with three features: (1) Twin-column synapse mapping scheme and current-based computation to eliminate the need of additional circuits to reverse the pre-synaptic spikes for negative weights [59], as well as area-consuming capacitor DAC

for V_{mem} accumulation (FIGURE 6.2); (2) A power-efficient 2-rail post-neuron circuit to implement the V_{mem} subtraction throughout the time window; (3) A threshold voltage (V_{th}) scaling specially designed for temporal-coded SNNs to reconfigurably extend the dynamic range of V_{mem}.

6.2 SRAM-based TTFS neuromorphic engine

FIGURE 6.3 shows the design overview of TFSRAM macro. The main components of TFSRAM include a 64×64 SRAM array with read/write interface, 64 spike generators to generate pre-synaptic spikes, 8 post-neuron circuits, and spike timing counters to capture the spike time of post-synaptic spikes. The SRAM array contains a 64×8 8-bit weight matrix.

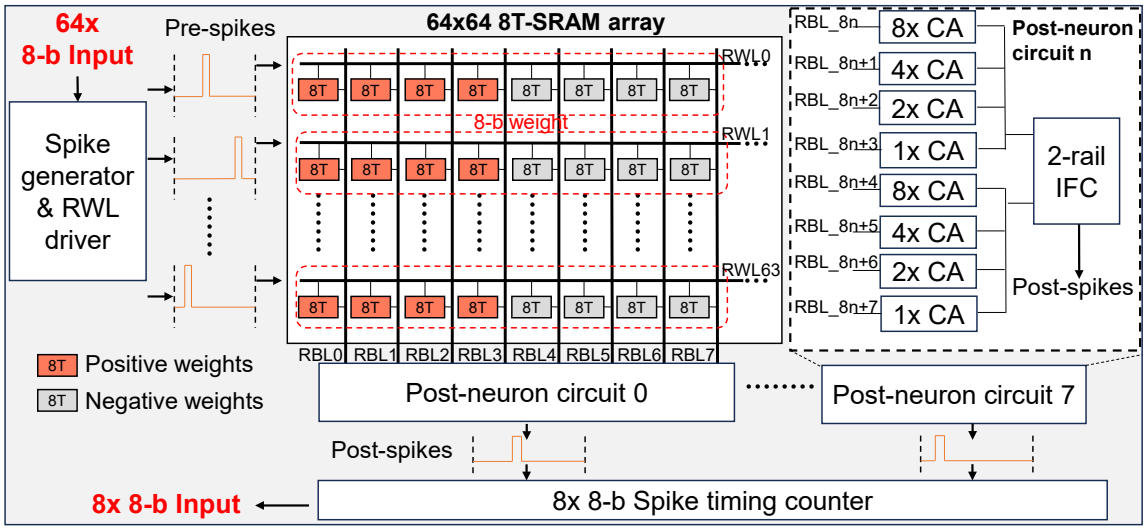


FIGURE 6.3: Design overview of TFSRAM.

Twin-column synapse mapping: One 8-bit synaptic weight is partitioned to a 4-bit positive part and a 4-bit negative part, and programmed to the SRAM cells in adjacent 4 read bitlines (RBLs) along the same read wordline (RWL), respectively. The positive and negative RBLs form a differential pair. When the pre-synaptic spikes are fed to the RWLs, spiking currents are generated along the RBLs from the '1' cells. Each 8 RBLs belonging to the same synaptic weight are connected to the same post-neuron circuit.

2-rail post-neuron circuit: The post-neuron circuit (FIGURE 6.4) computes the difference of the spiking currents from twin-column synapses and implements neuron dynamics. Each post-neuron module contains 8 current mirrors with proportional gains along the 8 RBLs, and one 2-rail IF circuit (IFC). The structure of 2-rail IFC in TFSRAM is identical with the one in ASTERS, introduced in Section 4.3.2. The 2-rail IFC contains two capacitors C_p and C_n with identical capacitance. The current mirrors from the positive part are connected to C_p, while the current mirrors from the negative part are connected to C_n. C_p accumulates membrane potential ($V_{mem,p}$) from 0V, while C_n accumulates membrane potential ($V_{mem,n}$) from V_{th}. Once $V_{mem,p}$ is larger than $V_{mem,n}$, the output post-synaptic spike (V_{out}) is generated with its pulse width controlled by V_{ref}. The differential membrane potential on C_p/C_n will then be reset to $0V/V_{th}$. The transient simulation waveform is shown in FIGURE 6.4 under V_{th}=400mV and V_{ref}=150mV. The post-synaptic spikes are converted to the digital domain by the subsequent 8-bit spike timing counters.

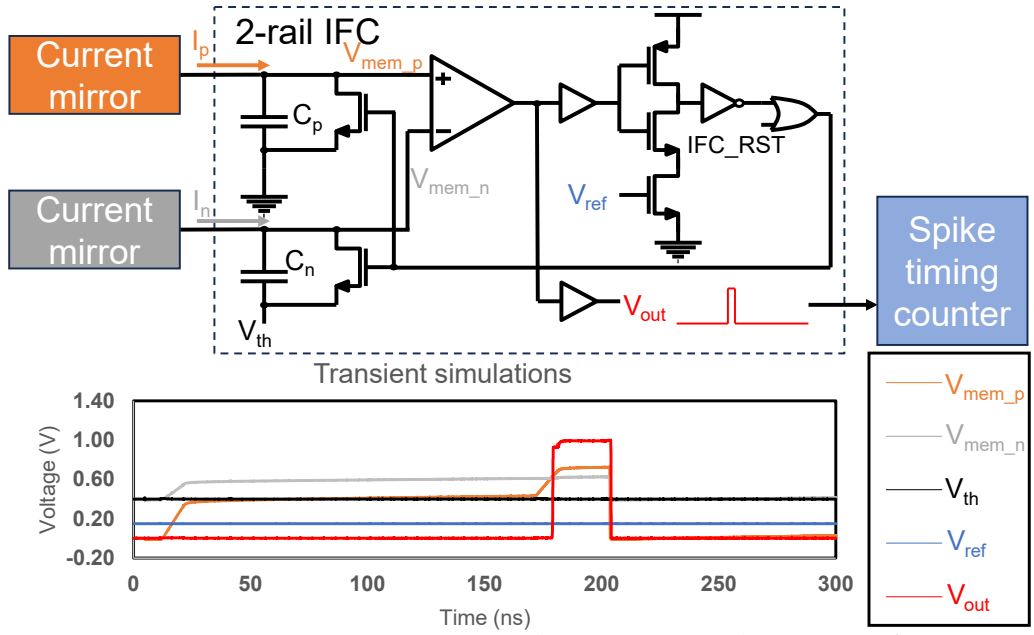


FIGURE 6.4: Post-neuron circuit and transient simulation waveform.

Threshold voltage scaling: In the 2-rail IFC, the equivalent V_{mem} cannot surpass $V_{DD} - V_{th}$. To expand the dynamic range of V_{mem}, we introduce the threshold voltage

scaling (TVS) mechanism specifically tailored for temporal-coded SNNs. The spike timing counter takes in a control signal named vth_scale to decides which spike from each neuron within the inference time window is regarded as the “first spike", whose spike timing is counted as the output. During the inference, V_{mem} remains increasing towards the threshold voltage to trigger post-synaptic spikes and being reset to 0. Therefore, the n-th post-synaptic spike indicates that the membrane potential has increased to the equivalent n-fold V_{th} (FIGURE 6.5).

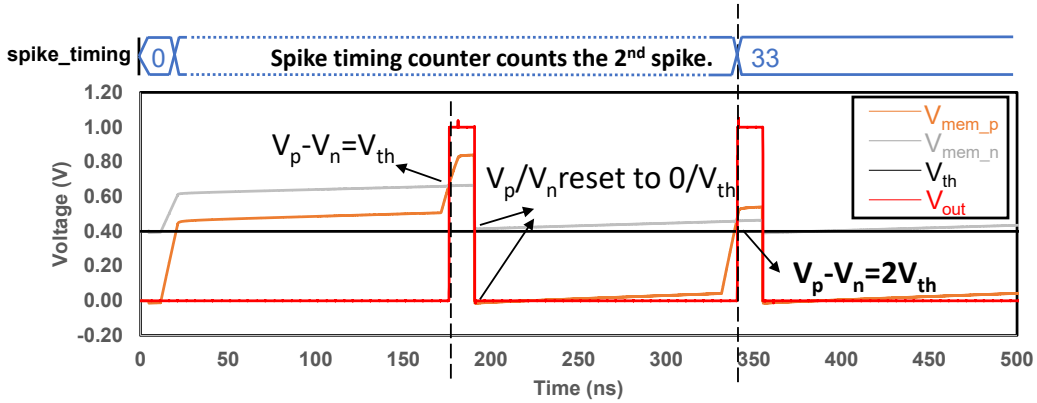
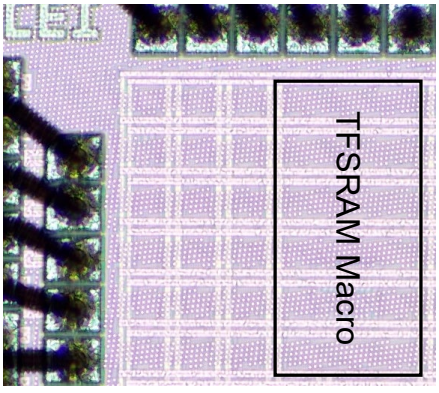


FIGURE 6.5: Threshold voltage scaling increases the dynamic range of V_{mem}.

6.3 Implementation results

FIGURE 6.6 depicts the die photo and chip summary of TFSRAM, fabricated in 65nm CMOS process. We measured the chip with the chip testing platform built on Xilinx



Technology	**65nm**
Voltage	**1V**
Clock Frequency	**100MHz**
Time Window	**2.56μs**
Die Area	**0.98mm x 0.98mm**
Macro Area	**336μm x 106μm**
Macro CIM Capacity	**4Kb**
Macro Power	**93.1μW @ 0.65V** **406.7μW @ 1.0V**

FIGURE 6.6: Die photo and summary of TFSRAM.

PYNQ-Z2 FPGA board for the control and clock signal generation. The chip is packaged by surface mount technology (SMT), and thus we use a socket to easily assemble the chip to the self-made printed circuit board (PCB) (Figure 6.7). To access the input and output registers of the TFSRAM macro, the I2C slave module is placed around the macro in the chip to support data communication under I2C protocol. The tasks of the FPGA include passing input data to the chip and fetching output data from the chip through I2C, and generating clock and reset signals for the chip.

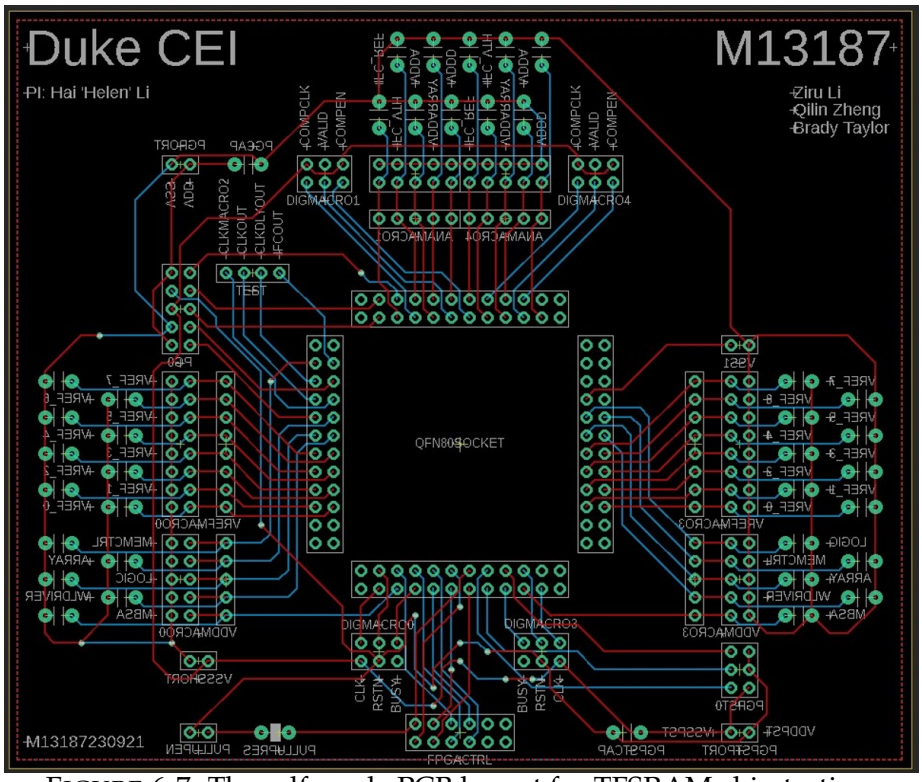


FIGURE 6.7: The self-made PCB layout for TFSRAM chip testing.

Under the condition of 1V/100MHz, the TFSRAM macro consumes $406.7\mu W$ and $98.2\mu W$ during computation and during idle state, respectively. FIGURE 6.8 shows the measured power breakdown and area breakdown of the TFSRAM macro. We also measured the macro power under the voltage supply from 0.5V to 1V. Tested with a pre-trained 2-layer TTFS-based SNN on Sklearn's 8×8 digit dataset, TFSRAM generates correct outputs while applying the voltage over 0.65V. Table 6.1 shows the comparison results with

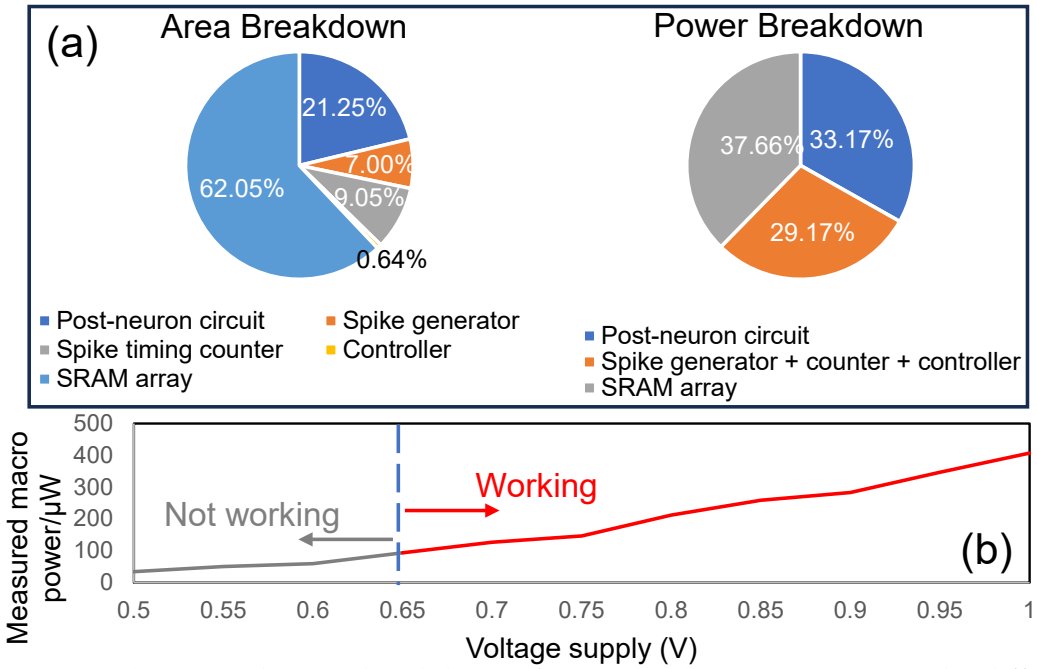


FIGURE 6.8: (a) Area and power breakdown of TFSRAM. (b) Macro power under different voltage supply.

prior spike-based PIM neuromorphic macros. TFSRAM achieves the energy efficiency of 249.8 TOPS/W and the area efficiency of 2.85 TOPS/mm^2 under 8-bit inputs and signed 4-bit weights.

Table 6.1: Comparison with prior spike-based PIM macros

Work	**This work**	VLSI'19 [1]	VLSI'22 [2]	ASSCC'22 [4]	TCSI'24 [5]
Process	**65nm**	150nm	28nm	65nm	40nm
Computing cell	**8T-SRAM**	1T1R RRAM	8T-SRAM	6T-SRAM	SOT-MRAM
Voltage (V)	**1**	-	1.1	-	-
Clock frequency (MHz)	**100**	50	200	-	-
Macro size (bit)	**64x64**	256x256	64x160	32Kb	512x512
Macro area (mm^2)	**0.036**	-	0.048	0.25	~0.064
Die area (mm^2)	**0.96**	-	2.91	-	-
Input precision (bit)	**1-8**	1-8	1-8	6-12 spikes	2
Weight precision (bit)	**4**	1	1/4/8	1.5b	4
Output precision (bit)	**8**	8	8	-	-
Macro power (mW)	**0.41**	1.52	15.8	0.56	-
Energy efficiency (TOPS/W)	**249.8** (8b, 4b)	16.9 (1b, 8b)	413.8 (4b, 4b)	290 Equivalent (2.8, 1.5b)	17.56 (2b, 4b)

7. Conclusion

My research is targeting at building dedicated spiking neuromorphic systems with CMOS/ReRAM-hybrid technology. These involve circuit designs for the inference of SOTA DNN/SNN algorithms, and hardware-software co-designs to improve the power efficiency, performance and robustness.

First, we recognize the potential of the spike-based data format, especially temporal-coding, in efficiently boosting the energy efficiency of ReRAM-based PIM designs. This results in ReSiPE, a novel ReRAM-based single-spiking processing engine for DNN inference, that outperforms prior level-based and rate-based ReRAM PIMs in terms of energy efficiency.

Second, we recognize that SNNs demonstrate superior energy efficiency compared to conventional DNNs due to their event-driven nature, and thus hold promise for power-constrained applications such as edge computing and IoT devices. Returning the single-spiking paradigm, a.k.a. TTFS scheme, back to the SNN further advances its capabilities of efficient neural processing. To support TTFS-based SNNs, we propose ASTERS, an adaptable threshold ReRAM-based PIM engine tailored for TTFS-based SNN inference with techniques to improve its efficiency and robustness against ReRAM process variations.

Third, we recognize that a complete neuromorphic vision system not only contains neuromorphic processors, but also requires image sensors to generate sensory data. The data communication between sensors and processors bottleneck the overall performance of neuromorphic vision systems. To eliminate the redundant data transmission in spike-based neuromorphic vision system, we present SpikeSen, a low-latency in-sensor-intelligence vision sensor design with neuromorphic spiking neurons, which directly performs the computation on photocurrents within sensor pixels in the spike frequency domain.

Last but not least, we pave along the way of ASTERS towards a fabricated PIM-based neuromorphic processing engine for TTFS-based SNN inference. We choose to utilize the SRAM-based PIM for higher robustness. The fabricated TFSRAM macro features multi-bit twin-column synapse mapping, current-based post-neuron circuit design and threshold

voltage scaling. Measured results show that TFSRAM achieves the energy efficiency of 249.8 TOPS/W and the area efficiency of 2.85 TOPS/mm^2 under 8-bit inputs and signed 4-bit weights.

The neuromorphic hardware designs presented in this report harness the spike-based data representation and in-memory-computing paradigm. These two design paradigms orthogonally contribute to neuromorphic computing systems that enable real-time and energy-efficient implementation of cognitive tasks on power-constrained AI hardware. I believe the co-design of hardware and algorithms tailored for PIM-based neuromorphic computing systems will unlock unprecedented levels of efficiency and performance, paving the way forward to the next-generation AI hardware that closely approaches the capabilities of the human brain while operating at a fraction of energy consumption.

The problems that need investigating with regard to the PIM-based neuromorphic design are summarized in the following subsections.

7.0.1 Enabling unsupervised learning

The neuromorphic designs proposed in this dissertation focus on the efficient inference of temporal-coded SNNs. Another critical advantage of temporal-coded SNNs over conventional DNNs is the capability of exploiting temporal dynamics and performing unsupervised learning rules such as STDP. Compared to the supervised learning algorithms, unsupervised learning rules eliminate the costs of gradient computation and data communication of back-propagation and enable local learning.

As demonstrated in Section 1.3, implementations of SNN unsupervised learning with ReRAM synapses have been explored in a series of research work [53, 20, 29, 51, 59]. While introducing these learning mechanisms to the ReRAM array in our proposed neuromorphic designs, the key challenge is how to capture the spatiotemporal information of spike events from the inference and program the synaptic arrays accordingly. An efficient solution should take full advantages of both the local learning capability (from the algorithm perspective) and the parallelism of the crossbar structure (from the hardware perspective) to avoid

unnecessary data caching, conversion and transmission. For example, we are targeting at unsupervised learning support on the basis of ReRAM-based spike-timing neumorphic macros like ASTERS. The methodology and circuit implementation to translate the firing time of the single spikes from the post-neuron circuits to the programming voltages of the ReRAM crossbars is the major issue to be resolved. Enabling unsupervised learning in our PIM-based spiking neuromorphic designs is a promising direction that will benefit the edge computing AI hardware in online learning and model fine-tuning tasks.

7.0.2 Exploring efficient inter-PE data communication

The proposed neuromorphic processing engine designs are often organized as computing nodes in a large-scale neuromorphic computing system. How to deploy different partitions of workload to these computing nodes and how to manage the data communication between these computing nodes are valuable questions that need addressing in the future research. For example, the NeuroSim model [45] adopts the H-tree structure for the inter-PE routing. The Loihi neuromorphic chip [9] uses the mesh-based network-on-chip to transmit spikes between different neuromorphic cores. In each time step, the synchronous mechanism needs to guarantee that all the spikes have been propagated to their destinations before moving forward to the next time step. This violates the asynchronous computation feature of SNNs because it still requires a high frequency clock to route and synchronize spike transmission. Some more aggressive interconnect topologies like the optical circuit switching (OCS) of TPU V4 supercomputer even bring the inter-chip interconnect into 3-D dimension [27] to gain connectivity and boost performance. It is worthy to explore the optimal inter-PE data communication schemes specially for the event-driven SNNs when building a neuromorphic computing system with the proposed PE designs.

Bibliography

[1] Filipp Akopyan et al. “Truenorth: Design and tool flow of a 65 mw 1 million neuron programmable neurosynaptic chip”. In: *IEEE transactions on computer-aided design of integrated circuits and systems* 34.10 (2015), pp. 1537–1557.

[2] Daniel Auge et al. “A survey of encoding techniques for signal processing in spiking neural networks”. In: *Neural Processing Letters* 53.6 (2021), pp. 4693–4710.

[3] Ben Varkey Benjamin et al. “Neurogrid: A mixed-analog-digital multichip system for large-scale neural simulations”. In: *Proceedings of the IEEE* 102.5 (2014), pp. 699–716.

[4] Stella Biderman et al. “Pythia: A suite for analyzing large language models across training and scaling”. In: *International Conference on Machine Learning*. PMLR. 2023, pp. 2397–2430.

[5] Kyeongryeol Bong et al. “A low-power convolutional neural network face recognition processor and a CIS integrated with always-on face detector”. In: *IEEE Journal of Solid-State Circuits* 53.1 (2017), pp. 115–123.

[6] Yongqiang Cao, Yang Chen, and Deepak Khosla. “Spiking deep convolutional neural networks for energy-efficient object recognition”. In: *International Journal of Computer Vision* 113 (2015), pp. 54–66.

[7] Gouranga Charan et al. “Accurate inference with inaccurate RRAM devices: Statistical data, model transfer, and on-line adaptation”. In: *2020 57th ACM/IEEE Design Automation Conference (DAC)*. IEEE. 2020, pp. 1–6.

[8] Wei-Hao 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: *2018 IEEE International Solid-State Circuits Conference-(ISSCC)*. IEEE. 2018, pp. 494–496.

[9] Mike Davies et al. “Loihi: A neuromorphic manycore processor with on-chip learning”. In: *Ieee Micro* 38.1 (2018), pp. 82–99.

[10] Peter U Diehl et al. “Conversion of artificial recurrent neural networks to spiking neural networks for low-power neuromorphic hardware”. In: *2016 IEEE International Conference on Rebooting Computing (ICRC)*. IEEE. 2016, pp. 1–8.

[11] Zidong Du et al. “ShiDianNao: Shifting vision processing closer to the sensor”. In: *Proceedings of the 42nd Annual International Symposium on Computer Architecture*. 2015, pp. 92–104.

[12] Bard Ermentrout. “Type I membranes, phase resetting curves, and synchrony”. In: *Neural computation* 8.5 (1996), pp. 979–1001.

[13] Zichen Fan et al. “Red: A reram-based deconvolution accelerator”. In: *2019 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE. 2019, pp. 1763–1768.

[14] Haowen Fang et al. “Neuromorphic algorithm-hardware codesign for temporal pattern learning”. In: *2021 58th ACM/IEEE Design Automation Conference (DAC)*. IEEE. 2021, pp. 361–366.

[15] IM Filanovsky and H Baltes. “CMOS Schmitt trigger design”. In: *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications* 41.1 (1994), pp. 46–49.

[16] Steve B Furber et al. “The spinnaker project”. In: *Proceedings of the IEEE* 102.5 (2014), pp. 652–665.

[17] Wulfram Gerstner et al. “A neuronal learning rule for sub-millisecond temporal coding”. In: *Nature* 383.6595 (1996), pp. 76–78.

[18] Sujan Kumar Gonugondla, Mingu Kang, and Naresh Shanbhag. “A 42pJ/decision 3.12 TOPS/W robust in-memory machine learning classifier with on-chip training”. In: *2018 IEEE International Solid-State Circuits Conference-(ISSCC)*. IEEE. 2018, pp. 490–492.

[19] Alan L Hodgkin and Andrew F Huxley. “A quantitative description of membrane current and its application to conduction and excitation in nerve”. In: *The Journal of physiology* 117.4 (1952), p. 500.

[20] Miao Hu et al. “A compact memristor-based dynamic synapse for spiking neural networks”. In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 36.8 (2016), pp. 1353–1366.

[21] Miao Hu et al. “Hardware realization of BSB recall function using memristor crossbar arrays”. In: *Proceedings of the 49th annual design automation conference*. 2012, pp. 498–503.

[22] Taras Iakymchuk et al. “Simplified spiking neural network architecture and STDP learning algorithm applied to image classification”. In: *EURASIP Journal on Image and Video Processing* 2015 (2015), pp. 1–11.

[23] Eugene M Izhikevich. “Resonate-and-fire neurons”. In: *Neural networks* 14.6-7 (2001), pp. 883–894.

[24] Eugene M Izhikevich. "Which model to use for cortical spiking neurons?" In: *IEEE transactions on neural networks* 15.5 (2004), pp. 1063–1070.

[25] Hao Jiang et al. "Pulse-width modulation based dot-product engine for neuromorphic computing system using memristor crossbar array". In: *2018 IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE. 2018, pp. 1–4.

[26] Sung Hyun Jo, Kuk-Hwan Kim, and Wei Lu. "High-density crossbar arrays based on a Si memristive system". In: *Nano letters* 9.2 (2009), pp. 870–874.

[27] Norm Jouppi et al. "Tpu v4: An optically reconfigurable supercomputer for machine learning with hardware support for embeddings". In: *Proceedings of the 50th Annual International Symposium on Computer Architecture*. 2023, pp. 1–14.

[28] Sangyeob Kim et al. "Neuro-cim: A 310.4 tops/w neuromorphic computing-in-memory processor with low wl/bl activity and digital-analog mixed-mode neuron firing". In: *2022 IEEE Symposium on VLSI Technology and Circuits (VLSI Technology and Circuits)*. IEEE. 2022, pp. 38–39.

[29] S Lashkare et al. "PCMO-based RRAM and NPN bipolar selector as synapse for energy efficient STDP". In: *IEEE Electron Device Letters* 38.9 (2017), pp. 1212–1215.

[30] Weitao Li et al. "Timely: Pushing data movements and interfaces in pim accelerators towards local and in time domain". In: *2020 ACM/IEEE 47th Annual International Symposium on Computer Architecture (ISCA)*. IEEE. 2020, pp. 832–845.

[31] Ziru Li, Bonan Yan, and Hai Li. "ReSiPE: ReRAM-based single-spiking processing-in-memory engine". In: *2020 57th ACM/IEEE Design Automation Conference (DAC)*. IEEE. 2020, pp. 1–6.

[32] Ziru Li et al. "ASTERS: adaptable threshold spike-timing neuromorphic design with twin-column ReRAM synapses". In: *Proceedings of the 59th ACM/IEEE Design Automation Conference*. 2022, pp. 1099–1104.

[33] Ziru Li et al. "RED: A ReRAM-based efficient accelerator for deconvolutional computation". In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 39.12 (2020), pp. 4736–4747.

[34] Ziru Li et al. "Spikesen: Low-latency in-sensor-intelligence design with neuromorphic spiking neurons". In: *IEEE Transactions on Circuits and Systems II: Express Briefs* (2023).

[35] Robert LiKamWa et al. "Redeye: analog convnet image sensor architecture for continuous mobile vision". In: *ACM SIGARCH Computer Architecture News* 44.3 (2016), pp. 255–266.

[36] Meng-Yao Lin et al. “DL-RSIM: A simulation framework to enable reliable ReRAM-based accelerators for deep learning”. In: *2018 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. IEEE. 2018, pp. 1–8.

[37] Chenchen Liu et al. “A spiking neuromorphic design with resistive crossbar”. In: *Proceedings of the 52nd Annual Design Automation Conference*. 2015, pp. 1–6.

[38] Yun Long, Xueyuan She, and Saibal Mukhopadhyay. “Design of reliable DNN accelerator with un-reliable ReRAM”. In: *2019 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE. 2019, pp. 1769–1774.

[39] Wolfgang Maass. “Networks of spiking neurons: the third generation of neural network models”. In: *Neural networks* 10.9 (1997), pp. 1659–1671.

[40] Lukas Mennel et al. “Ultrafast machine vision with 2D material neural network image sensors”. In: *Nature* 579.7797 (2020), pp. 62–66.

[41] Paul Merolla et al. “A digital neurosynaptic core using embedded crossbar memory with 45pJ per spike in 45nm”. In: *2011 IEEE custom integrated circuits conference (CICC)*. IEEE. 2011, pp. 1–4.

[42] Reiji Mochida et al. “A 4M synapses integrated analog ReRAM based 66.5 TOPS/W neural-network processor with cell current controlled writing and flexible network architecture”. In: *2018 IEEE Symposium on VLSI Technology*. IEEE. 2018, pp. 175–176.

[43] Kenichi Ohhata. “A 2.3-mW, 950-MHz, 8-bit fully-time-based subranging ADC using highly-linear dynamic VTC”. In: *2018 IEEE Symposium on VLSI Circuits*. IEEE. 2018, pp. 95–96.

[44] Seokjun Park et al. “7.2 243.3 pJ/pixel bio-inspired time-stamp-based 2D optic flow sensor for artificial compound eyes”. In: *2014 IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC)*. IEEE. 2014, pp. 126–127.

[45] Xiaochen Peng et al. “DNN+ NeuroSim V2. 0: An end-to-end benchmarking framework for compute-in-memory accelerators for on-chip training”. In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 40.11 (2020), pp. 2306–2319.

[46] Mirko Prezioso et al. “Self-adaptive spike-time-dependent plasticity of metal-oxide memristors”. In: *Scientific reports* 6.1 (2016), p. 21331.

[47] Johannes Schemmel et al. “A wafer-scale neuromorphic hardware system for large-scale neural modeling”. In: *2010 IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE. 2010, pp. 1947–1950.

[48] Ali Shafiee et al. “ISAAC: A convolutional neural network accelerator with in-situ analog arithmetic in crossbars”. In: *ACM SIGARCH Computer Architecture News* 44.3 (2016), pp. 14–26.

[49] Xueyuan She, Yun Long, and Saibal Mukhopadhyay. “Improving robustness of reram-based spiking neural network accelerator with stochastic spike-timing-dependent-plasticity”. In: *2019 International Joint Conference on Neural Networks (IJCNN)*. IEEE. 2019, pp. 1–8.

[50] Chen Shoushun and Amine Bermak. “Arbitrated time-to-first spike CMOS image sensor with on-chip histogram equalization”. In: *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 15.3 (2007), pp. 346–357.

[51] Amar Shrestha et al. “Stable spike-timing dependent plasticity rule for multilayer unsupervised and supervised learning”. In: *2017 international joint conference on neural networks (IJCNN)*. IEEE. 2017, pp. 1999–2006.

[52] Gregory D Smith et al. “Fourier analysis of sinusoidally driven thalamocortical relay neurons and a minimal integrate-and-fire-or-burst model”. In: *Journal of neurophysiology* 83.1 (2000), pp. 588–610.

[53] Greg S Snider. “Spike-timing-dependent learning in memristive nanodevices”. In: *2008 IEEE international symposium on nanoscale architectures*. Ieee. 2008, pp. 85–92.

[54] Linghao Song et al. “Pipelayer: A pipelined reram-based accelerator for deep learning”. In: *2017 IEEE international symposium on high performance computer architecture (HPCA)*. IEEE. 2017, pp. 541–552.

[55] Brady Taylor et al. “Processing-in-memory technology for machine learning: From basic to asic”. In: *IEEE Transactions on Circuits and Systems II: Express Briefs* 69.6 (2022), pp. 2598–2603.

[56] Simon Thorpe, Arnaud Delorme, and Rufin Van Rullen. “Spike-based strategies for rapid processing”. In: *Neural networks* 14.6-7 (2001), pp. 715–725.

[57] Xiuling Wang, Winnifred Wong, and Richard Hornsey. “A high dynamic range CMOS image sensor with inpixel light-to-frequency conversion”. In: *IEEE Transactions on electron devices* 53.12 (2006), pp. 2988–2992.

[58] Yu Wang et al. “Energy efficient RRAM spiking neural network for real time classification”. In: *Proceedings of the 25th edition on Great Lakes Symposium on VLSI*. 2015, pp. 189–194.

[59] Zhongrui Wang et al. “Fully memristive neural networks for pattern classification with unsupervised learning”. In: *Nature Electronics* 1.2 (2018), pp. 137–145.

[60] Parami Wijesinghe et al. "An all-memristor deep spiking neural computing system: A step toward realizing the low-power stochastic brain". In: *IEEE Transactions on Emerging Topics in Computational Intelligence* 2.5 (2018), pp. 345–358.

[61] Chen Xu et al. "5.1 A stacked global-shutter CMOS imager with SC-type hybrid-GS pixel and self-knee point calibration single frame HDR and on-chip binarization algorithm for smart vision applications". In: *2019 IEEE International Solid-State Circuits Conference-(ISSCC)*. IEEE. 2019, pp. 94–96.

[62] Han Xu et al. "Senputing: An ultra-low-power always-on vision perception chip featuring the deep fusion of sensing and computing". In: *IEEE Transactions on Circuits and Systems I: Regular Papers* 69.1 (2021), pp. 232–243.

[63] Han Xu et al. "Utilizing direct photocurrent computation and 2D kernel scheduling to improve in-sensor-processing efficiency". In: *2020 57th ACM/IEEE Design Automation Conference (DAC)*. IEEE. 2020, pp. 1–6.

[64] Tomohiro Yamazaki et al. "4.9 A 1ms high-speed vision chip with 3D-stacked 140GOPS column-parallel PEs for spatio-temporal image processing". In: *2017 IEEE International Solid-State Circuits Conference (ISSCC)*. 2017, pp. 82–83. DOI: 10.1109/ISSCC.2017.7870271.

[65] Bonan Yan et al. "Neuromorphic Computing Systems with Emerging Nonvolatile Memories: A Circuits and Systems Perspective". In: *2020 International Symposium on VLSI Technology, Systems and Applications (VLSI-TSA)*. IEEE. 2020, pp. 122–123.

[66] Bonan Yan et al. "RRAM-based spiking nonvolatile computing-in-memory processing engine with precision-configurable in situ nonlinear activation". In: *2019 Symposium on VLSI Technology*. IEEE. 2019, T86–T87.

[67] Shimeng Yu, Yi Wu, and H-S Philip Wong. "Investigating the switching dynamics and multilevel capability of bipolar metal oxide resistive switching memory". In: *Applied Physics Letters* 98.10 (2011).

[68] Lei Zhang et al. "Tdsnn: From deep neural networks to deep spike neural networks with temporal-coding". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 33. 01. 2019, pp. 1319–1326.

[69] Qilin Zheng et al. "Accelerating Sparse Attention with a Reconfigurable Non-volatile Processing-In-Memory Architecture". In: *2023 60th ACM/IEEE Design Automation Conference (DAC)*. IEEE. 2023, pp. 1–6.

[70] Shibo Zhou and Xiaohua Li. "Spiking neural networks with single-spike temporal-coded neurons for network intrusion detection". In: *2020 25th International Conference on Pattern Recognition (ICPR)*. IEEE. 2021, pp. 8148–8155.

Biography

Ziru Li received the Bachelor of Engineering (2019) in Electronic Engineering from Tsinghua University, Beijing, China. He started pursuing his Ph.D. degree in Electrical and Computer Engineering at Duke University in 2019. His research interests mainly focus on mixed-signal circuit and system design for neuromorphic computing.

Ziru Li has authored or co-authored 12 papers in top-tier conferences and journals, including DAC [31, 32, 14, 69], TCAS-II [34, 55], TCAD [33], DATE [13], ISVLSI [65], etc. He is serving as a reviewer of IEEE journals including TCAD, TCAS-I, TCAS-II, TVLSI, TC, ACM journal JETC.