2020

Ultra-low Power Circuits and Architectures for Neuromorphic Computing Accelerators with Emerging TFETs and ReRAMs

Jie Lin
University of Central Florida

ULTRA-LOW POWER CIRCUITS AND ARCHITECTURES FOR NEUROMORPHIC COMPUTING ACCELERATORS WITH EMERGING TFETS AND RERAMS

by

JIE LIN
B.S. University of Electronic Science and Technology of China, 2006
M.S. University of Electronic Science and Technology of China, 2009

A dissertation submitted in partial fulfilment of the requirements
for the degree of Doctor of Philosophy
in the Department of Electrical and Computer Engineering
in the College of Engineering and Computer Science
at the University of Central Florida
Orlando, Florida

Spring Term
2020

Major Professor: Jiann-Shiun Yuan

ABSTRACT

Neuromorphic computing using post-CMOS technologies is gaining increasing popularity due to its promising potential to resolve the power constraints in Von-Neumann machine and its similarity to the operation of the real human brain.

To design the ultra-low voltage and ultra-low power analog-to-digital converters (ADCs) for the neuromorphic computing systems, we explore advantages of tunnel field effect transistor (TFET) analog-to-digital converters (ADCs) on energy efficiency and temperature stability. A fully-differential SAR ADC is designed using 20 nm TFET technology with doubled input swing and controlled comparator input common-mode voltage.

To further increase the resolution of the ADC, we design an energy efficient 12-bit noise shaping (NS) successive-approximation register (SAR) ADC. The 2nd-order noise shaping architecture with multiple feed-forward paths is adopted and analyzed to optimize system design parameters. By utilizing tunnel field effect transistors (TFETs), the $\Delta\Sigma$ SAR is realized under an ultra-low supply voltage VDD with high energy efficiency.

The stochastic neuron is a key for event-based probabilistic neural networks. We propose a stochastic neuron using a metal-oxide resistive random-access memory (ReRAM). The ReRAM's conducting filament with built-in stochasticity is used to mimic the neuron's membrane capacitor, which temporally integrates input spikes. A capacitor-less neuron circuit is designed, laid out, and simulated. The output spiking train of the neuron obeys the Poisson distribution.

Based on the ReRAM based neuron, we propose a scalable and reconfigurable architecture that exploits the ReRAM-based neurons for deep Spiking Neural Networks (SNNs). In prior publications, neurons were implemented using dedicated analog or digital circuits that are not area and energy

efficient. In our work, for the first time, we address the scaling and power bottlenecks of neuromorphic architecture by utilizing a single one-transistor-one-ReRAM (1T1R) cell to emulate the neuron. We show that the ReRAM-based neurons can be integrated within the synaptic crossbar to build extremely dense Process Element (PE)–spiking neural network in memory array–with high throughput. We provide microarchitecture and circuit designs to enable the deep spiking neural network computing in memory with an insignificant area overhead.

To my wife Feiyan Mu, my parents Wendu Lin and Wenping Liu

ACKNOWLEDGMENTS

It is the most precious period that I studied for my Ph.D. degree at the University of Central Florida (UCF) in Orlando. I want to thank every person who has ever helped and inspired me during the four years' studying. Also, I want to thank my family members' love and support, all the professors' teaching and guidance, and all my supervisors' direction and encouragement.

First, I would like to especially thank my advisor Dr. Jiann-Shiun Yuan for his guidance and supervision during my Ph.D. study. Dr. Yuan leads and improves the Secure CMOS Design Lab at UCF that supports and offers the necessary material to me. Dr. Yuan is very professional and knowledgeable in neuromorphic computing and circuit design, and he always gives me good suggestions, advises, and directions for research. Thanks to Dr. Yuan, I am always on the right track for my research work. With Dr. Yuan's inspiration and encouragement, I can finish my research projects and publications on time and make my dissertation much better.

After that, I would like to thank all my dissertation committee members. Thank Dr.Yuan to serve my committee chair. Thank Dr. Kalpathy B. Sundaram in UCF, Dr. Mingjie Lin in UCF, Dr. Rickard Ewetz in UCF, and Dr. Zixi (Jack) Cheng in UCF to be willing serving my dissertation committee members. And thanks for all their time spending and professional suggestions.

Besides, I would like to appreciate the all wonderful people in our Secure CMOS Design Lab at UCF, I would express my gratitude to all my lab-mates for their generous help and daily company, including Yu Bi, Qutaiba Alasad, Shayan Taheri, Wen Yang, and Milad Salem.

Finally, I would like to give special thanks to my wife Feiyan Mu, for her love and accompany. I also appreciate my family and my parents, Wendu Lin and Wenping Liu. Thanks for their support and consideration.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1: INTRODUCTION

1.1 Motivation

1.1.1 Neuromorphic Computing Systems

Neuromorphic computing systems consist of electronic circuits and devices that mimic the biological nervous systems [2, 3]. The circuits are typically designed using mixed-mode analog/digital circuits with complementary metal-oxide-semiconductor (CMOS) transistors. Neuromorphic computing systems process information using energy-efficient asynchronous and event-driven methods [4]. Neuromorphic computing systems are often more adaptive, fault-tolerant, and flexible than traditional computing systems [5]. Conventional computing systems utilize Von-Neumann architecture with one or more central processing units physically, with separated the main memory, which leads to a critical bottleneck in today's computing system—the memory wall [6]. On the other hand, neuromorphic computing systems are characterized by near memory computing: the synapses, as well as the neurons of the neural network are implemented in the same location to perform both information storage and nonlinear computation. Therefore, neuromorphic computing paradigms offer an attractive solution for implementing alternative non-Von-Neumann architectures with advanced and emerging technologies [5].

This neuromorphic computing paradigm is now being researched by an increasing number of research groups. Many recent works try to use new materials and nanotechnologies for building nanoscale devices that can emulate some of the properties of the biological neurons and synapses [7–11]. At the system level, remarkable brain-inspired neuromorphic computing platforms have been developed to perform pattern recognition and machine learning tasks [12] and for fast simulation of biology neuro-system models [13]. All researches show orders of energy

efficiency and throughput improvement over traditional computers.

1.1.2 The Needs and Challenges for Ultra-low Power and Ultra-low Voltage Data Converters

Most of the computation of neuromorphic computing system is completed in analog domain. However, input/output data and intermediate data being transmitting between computing cores are always in digital domain. Thus data converters are essential building blocks in neuromorphic computing systems. Unfortunately, data converters, especially analog-to-digital converters (ADCs) consumes significant amount of power and silicon area. For example, in ISAAC architecture, the ADCs consumes 58% of the total power and 30% of the total area [14]. Moreover, many of the neuromorphic systems work under ultra-low supply voltage below 0.5 V [4, 5]. Thus, ultra-low voltage and low-power operation is desirable for ADCs in neuromorphic computing systems.

The charge redistribution successive-approximation-register (SAR) ADC is suited for low-voltage and low-power operation. In particular, SAR ADCs can be deployed for wide frequency range because of its simple analog circuitry and the high precision of capacitor matching in modern semiconductor processes [15–19]. However, using CMOS technology, ADCs with advanced process can operate at high speed whereas their minimum power consumption is limited by static leakage power at low frequency. Moreover, the leakage current will degrade the accuracy of capacitor based feedback digital-to-analog converters (DACs). The problem of leakage current becomes worse as process scales. For instance, the of leakage current of the 16-nm CMOS transistors is about 100 x higher than that of 40-nm CMOS. Therefore, we can not obtain enough power reduction by just utilizing process scale down.

The other issue related to the low voltage operating of the advanced CMOS process is the decrease of the on resistance in switches in the ADCs. During operation, the input analog signal is sampled when the clock signal is high. To reduce the on-resistance of the switch especially for

the input sample and hold (S/H) switches, bootstrapped circuits [20] are commonly used. However, the added circuity to implement bootstrapped will significantly increase the power and area consumption.

1.1.3 Recent Progress of Neuromorphic Circuits and Architectures

The human brain's cognitive power emerges from noisy, imprecise and unreliable components. This realization has motivated scientists and engineers to explore event-based probabilistic neural networks [21, 22]. Emulation of biological neuronal dynamics, including leaky-integrate-and-fire (LIF) neurological stochasticity and event-driven design, is fundamental for the implementation of such systems. The LIF is often emulated in silicon [7–9]. However, in those designs capacitors are leveraged to perform the integration and store the membrane potential. This makes the neuron layout difficult to scale with technology, and hinders the integration capability of the neuromorphic hardware.

In addition to the deterministic neurological dynamics, the value of noise as a resource for neural computation had already been addressed in the context of artificial neural networks [23, 24]. The stochasticity of biological process mainly emerges from the unreliability of synapses and stochastic openings and closings of membrane channels [24]. However, for the sake of mathematical simplicity, noise is usually projected into the spike generation process of the presynaptic neuron [24]. In this way, noise in the network can be modeled by a stochastical firing neuron [25]. The common approach to add stochasticity to a deterministic neuron is to introduce uncorrelated background noise into every neuron [9]. This degrades power efficiency and limits scalability. In an event-based neural network, only the arrival of events will trigger processing; otherwise the system remains silent and consumes little power. Thus, all the neurons can adapt their speed according to input spikes [21], leading to superior performance per watt (PPW) metric.

Recently, research has focused on exploiting the emerging nanoscale devices to emulate the dynamics of neural systems [26–28]. Due to non-volatility, high scalability, and energy efficiency, the memristor crossbar is a promising candidate and has been used to simulate the synapses [28, 29]. Additionally, due to their successful application as memory cells, researchers have proposed in-memory neural networks based on memristors [30, 31] that have the potential for much higher speed and lower energy consumption than today's Von Neumann computer architecture. In 2016, Tuma et al. [22] used the phase configuration of chalcogenide-based phase-change material as the membrane potential of the neuron. Also, conductive bridge memory (CBRAM) was used to mimic the neuron in [32, 33]. In [22], however, the magnitude of reset pulse is high (6 V), which can increase the power consumption and cause reliability issues. In [32], a negative voltage source is needed to reset the CBRAM device and thus increase the design complication. In [33], a 10 pF capacitor is used as the membrane, making the neuron hard to scale with the evolution of technology. Moreover, the research of integrating the memristor based neurons with the memristor synapses is still not yet fully explored.

Although deterministic neurons can be used to implement stochastic behavior based on exploiting network-generated variability [34], this approach is less attractive due to the lack of direct control over stochastic properties of such networks (e.g., the input rate dependent stochastic spiking train cannot be straightforwardly mapped onto such network, and can be only approximated [9]). Moreover, Poissonian statistics generated by a network itself is usually only stable for a limited input range, if no additional noise generation mechanism is used [9, 35].

Among all the current memory technologies, metal-oxide ReRAM attracts much attention, because of its compatibility with conventional semiconductor processes and low write energy. In [36], M. Lee et al. have demonstrated that the ReRAM have superior cycling endurance of over 10^{12}, reducing the risk of reading error caused by the gradually decreased off-state resistance after long-time operation. Here, for the first time, we introduce an artificial neuron that uses the metal-oxide

ReRAM device to realize integration-and-fire with stochastic dynamics. The actual membrane potential of the neuron is stored as the length of the conducting filament, therefore the membrane capacitor is omitted in our design. The random generation and migration of oxygen vacancies result in built-in stochastic behavior. We propose a behavioral model that include the cycle-to-cycle and device-to-device variations of the ReRAM for fast behavioral simulation. We design a ReRAM-based neural circuit and examine the area, process variation effect, and power consumption of the circuit.

Spiking neural network (SNN) is regarded as the third generation neural network for its potential in improving the latency and energy efficiency of the deep neural network (DNN) [37]. On the other hand, traditional Von Neumann architecture suffers from "memory wall" [6], where moving data and operating the memory buses is much more expensive than computing itself [38], especially for the data and computing intensive neural network tasks [39, 40]. In-memory computing architectures are promising solutions to address this problem by moving computing to memory. By designing processing units inside/near the memory array, in-memory architectures dramatically diminish the overhead of data movements [27, 41]. So, it is natural to combine in-memory computing and SNN to introduce a new generation of compact and energy efficient neural network architecture.

The emerging non-volatile memory (NVM), such as phase changing memory (PCM) [42], spin-transfer torque magnetic random access memory (STT-MRAM) [43], and resistive random access memory (ReRAM) [44], have long attracted researcher's attention because the crossbar structure is very suitable for designing the synapse arrays [10, 11]. The NVM synapses provide high throughput and small silicon estate occupation [45], especially when high-density 3D stacking is employed [46]. In [30], Aayush Ankit et al. propose RESPARC, the first reconfigurable in-memory SNN architecture built on NVM crossbars. RESPARC uses analog neurons to accumulate spike current from the NVM crossbar and programmable switches to transfer spiking packets. However,

the implementation of analog neurons that process the output current of the crossbar are relatively complex and hinders seamless integration with highly dense synaptic arrays.

Moreover, even with the aggressive scaling of technology, realizing the capacitance densities measured in biological neuronal membranes ($\sim$ 10fF/μm^2) is challenging [47]. Most recently, researchers begin to explore the possibility to use NVMs as the neurons [47–49]. With the help of NVM neurons, it is possible to embed all computing in an SNN—multiply-accumulate operation (MAC) of synaptic signals and integrate-and-fire (IF) of membrane potential—in the NVM memory arrays.

1.2 Chapter Outline

This dissertation will focus on ultra-low power circuits and architectures for neuromorphic computing tasks. Chapter 1 give a brief introduction on the ultra-low power neuromorphic computing and recent progress on the neuromorphic circuits and architectures. Chapter 2 proposes a ultra-low power SAR ADC to convert the output analog neuromorphic signal back to digital. Chapter 3 increase the signal-to-noise ratio of the proposed ADC to 76 dB using noise shaping technology. Chapter 4 introduce a ReRAM based artificial neuron that will function as the integrate-and-fire membrane with very little area and power consumption. Chapter 5 shows a low power architecture utilizing the ReRAM based neuron to perform machine learning workloads. Finally, chapter 6 draws the conclusion of this dissertation.

CHAPTER 2: ULTRA-LOW POWER ADC DESIGN USING EMERGING TUNNELING FIELD EFFECT TRANSISTORS

[1]In the past few years, with the advancement of power-constrained applications, such as energy harvesting systems, wireless sensor networks, and biomedical implants, there has been a growing interest in temperature stable ultra-low-power designs. As an important module that links the digital core to the real world, the energy efficiency and temperature stability to perform analog to digital conversion (ADC) have profound impact on the overall system performance. For low-resolution ADCs, where noise floor is no longer the power limiting factor, the energy efficiency is limited by CMOS technology for its 60 mV/dec sub-threshold slope (SS) [50]. In addition, the sensitivity of sub-threshold circuit to temperature increases when comparing to circuit working in strong inversion, which may introduce large temperature drift that even cause the circuit fail to function properly. Tunneling field effect transistor (TFET), with its superior IDS-VGS characteristic and weaker temperature dependency, is considered a strong candidate for ultra-low-power and ultra-low-VDD design. Moreover, TFET is a standard CMOS process flow compatible device [51]. Therefore, it can be potentially utilized to address the growing challenge for the CMOS technology for ultra-low power ADC design.

The goal of this chapter is to explore and utilize the advantages of 20-nm TFET device characteristics to design energy efficient and low temperature variation successive approximation register (SAR) ADC operating under ultra-low supply voltage VDD. We explore the design, analysis, performance and temperature stability of the TFET 6-bit successive approximation register (SAR) ADC. The SAR ADC topology is chosen due to its outstanding energy efficiency with low to mod-

[1]This chapter was published as Lin, Jie, and Jiann-Shiun Yuan. "Ultra-low power successive approximation analog-to-digital converter using emerging tunnel field effect transistor technology." *Journal of Low Power Electronics* 12.3 (2016): 218-226.

erate resolution and medium sampling rate [52], where power dissipation of the ADC is mainly limited by technology [50]. We also investigate the temperature variation of the TFET SAR ADC from -55°C to 125°C. This chapter is organized as follows. Section II presents the advantage of the TFET transistors over CMOS counterparts in terms of energy efficiency and temperature efficiency. Section III presents circuit implementation of ultra-low power SAR ADC. Section IV presents the energy and performance evaluation of the SAR ADC. The conclusion is provided in Section IV to summarize our results

2.1 TFET'S Advantages on Ultra-Low Power ADC Design

2.1.1 Challenges in Ultra-Low-Power Low Temperature Variation Design on CMOS Technology

In mixed signal design, due to the limited number of transistors, the static energy (transistor leakage) is negligible. And the dynamic energy of the digital part can be expressed as:

$$E_D = \frac{1}{2}\alpha C V_{DD}^2 \tag{2.1}$$

where α is the oversampling ratio. Differing from digital circuits, the power dissipation of analog part of the circuit is limited by noise floor and signal bandwidth. For an ideal class B operation, the power dissipation of analog circuit is [50]:

$$E_A = \beta \frac{1}{V_{DD}} k_B T \cdot SNR \cdot \frac{1}{g_m/I_{DS}} \tag{2.2}$$

where k_B is the Boltzmann's constant; T is the absolute temperature; β is a constant related to circuit topology and SNR is signal-to-noise ratio. From [50] it is noted that when the resolution

is low (SNDR<60 dB), the energy is mainly limited by the characteristic of CMOS technology, more specifically, by the ratio of g_m/I_{DS}, which can be written as [51]:

$$\frac{g_m}{I_{DS}} = \frac{\ln(10)}{SS} \tag{2.3}$$

For CMOS transistors, $SS = \ln(10)k_B/q \approx 60\text{mV/dec}$, corresponding to a g_m/I_{DS} ratio of 38.3 V^{-1} at room temperature. As a result, energy efficiency of low resolution ADCs is limited by SS of CMOS technology.

2.1.2 Energy Efficiency of TFET

TFETs utilize a gate voltage to control the band-to-band tunneling across a p-n junction [53]. The cross-section and energy band diagrams of n-channel TFET in OFF and ON states are shown in Fig. 2.1a and 2.1b. From Fig. 2.1, when zero bias voltage is applied to the gate of the TFET, the conduct band minimum of the channel E_C is above the valence band maximum of the source E_V. Thus the band-to-band tunneling is shut down and the device is off. When a bias voltage is applied to the gate of the transistor, the conduction band of the channel is shifted down. A tunneling window, V_{TW}, will be created if EC is below E_V. As a result, electrons in the source will tunnel into the channel and the device is on.

Using Kane-Sze tunneling formula, the drain current of TFET can be modeled as

$$I_D = afV_{TW}\xi e^{-b/\xi} \tag{2.4}$$

where where a, b, and f are coefficients determined by the martial properties; $\xi = \xi_0(1 + \gamma_1 V_{DS} + \gamma_2 V_{GS})$, γ_1 and γ_2 being constants, is the average electrical field. As a result, the SS of TFET and

be expressed as:

$$SS_T = \ln(10) I_{DS} \frac{\partial I_{DS}}{\partial V_{GS}}^{-1} = \ln(10) \left[\frac{1}{V_{TW}} \frac{dV_{TW}}{dV_{GS}} + \frac{\xi + b}{\xi^2} \frac{d\xi}{dV_{GS}} \right]^{-1} \tag{2.5}$$

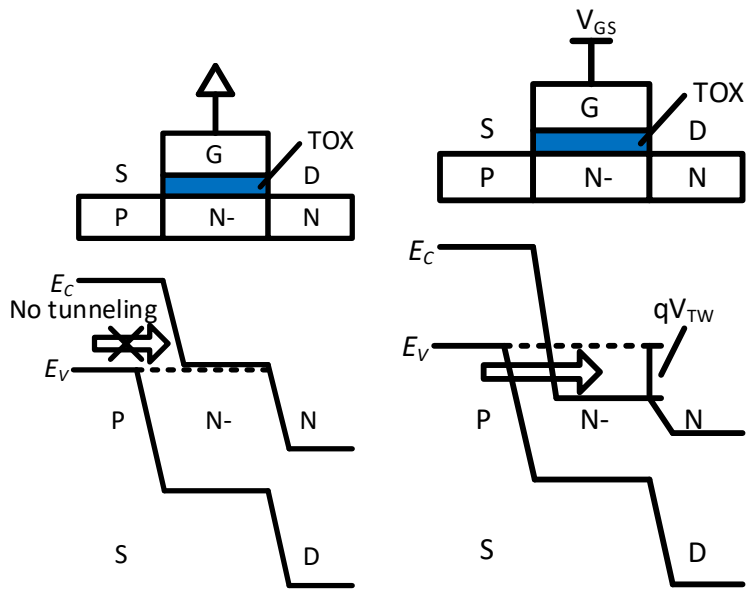


Figure 2.1: Cross-section and energy band diagram of a TFET when the device is biased in (a) OFF (b) ON state

With a thin and/or high k gate dielectric, the gate-source voltage can directly control the tunneling window, i.e., $dV_{TW}/dV_{GS} \approx 1$. As a result, the first term in Eq. 2.5 decreases with the V_{GS}. Also, based on definition of ξ, the second term of Eq. 2.5 decreases with the V_{GS} too. Hence, TFET is able to overcome the 60 mV/dec SS when VDD is low, and thus can achieve a g_m/I_{DS} ratio well above 40 V^{-1}. Moreover, GaSb-InAs heterojunction with lower bandgap to silicon is employed [53] to improve tunneling probability. As a result, g_m/I_{DS} ratio can be further improved.

For an ADC with median or low SNR, the performance is not limited by noise. Thus, sampling frequency becomes an important parameter to measure circuit performance. TFET impact on ADC

energy efficiency with a fixed sampling frequency is explored as follows: according to Eq. 2.1, VDD is a major input for ADC energy efficiency. Consequently, it is highly important to reduce the SS and maintain higher drive current because it eventually leads to lower supply voltage and hence the low power dissipation. A comparison of I-V characteristic of TFET and 20 nm CMOS transistor predicted by the model equations is shown in Fig. 2.2.

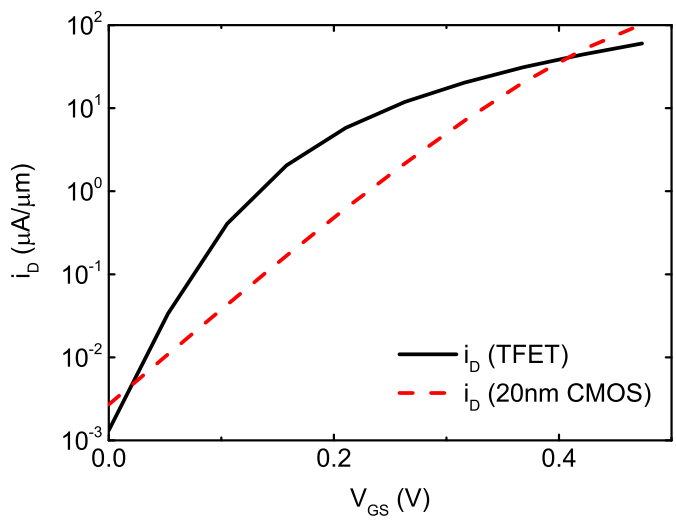


Figure 2.2: I-V characteristic comparison between TFET and CMOS transistor

From Fig. 2.2, within the sub-threshold region, TFET shows higher on-current per channel width, leading to a lower delay. Consequently, with the same sampling frequency, TFET based ADCs can work under lower VDD, which shows an improved energy efficiency over that of CMOS technology.

For an ADC with median or low SNR, Eq. 2.2 can be reduced according to [50]:

$$E_A = A\frac{1}{g_m/I_{DS}} \tag{2.6}$$

where A is a constant. Eq. 2.6 shows that g_m/I_{DS} ratio is another major constraint to the energy of the ADC. Allowing the transistor to work in the sub-threshold region will make the transistor

to approach the maximum g_m/I_{DS} ratio. However, lowering VDD will also increase the transistor delay and hence lower the transit frequency (f_T) of the transistor, limiting the sampling frequency of the ADC. A trade-off between f_T and g_m/I_{DS} is simulated and the results is shown in Fig. 2.3. To make a fair comparison, the dimension of both TFET and CMOS transistors are set the same as 40 nm $\times$ 20 nm.

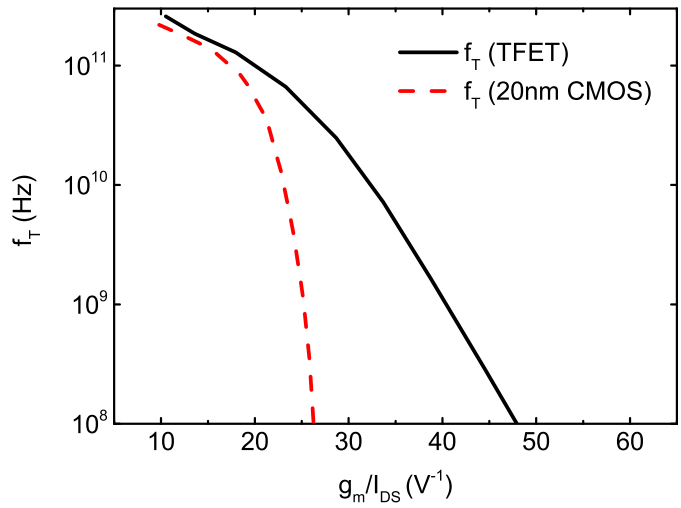


Figure 2.3: Tradeoff between g_m/I_{DS} and f_T in 20 nm TFET and CMOS technologies

From Fig. 2.3, for a fixed f_T and hence a fixed sampling frequency, TFET transistors deliver higher g_m/I_{DS} ratio, particularly in median or low frequency region ($f_T < 5$ GHz). In order to achieve f_T = 5 GHz, a 20nm CMOS device need to be biased such that $g_m/I_{DS} \approx 22$ V^{-1}. With TFET transistors, 5 GHz f_T can be achieved with a $g_m/I_{DS} \approx 37$ V^{-1}, indicating a 40% energy reduction, according to Eq. 2.2.

2.1.3 The Temperature Dependence of TFET

In the OFF state (V_{GS} or $V_{GS} < V_{th}$) I_{DS} is constituted by a leakage current, and its temperature variation is mainly contributed by thermal generation in the depletion region [54, 55], and the

temperature dependence of I_{DS} can be modeled as:

$$I_{DS}(T) = I_{DS_0} + \alpha(T - T_0) \tag{2.7}$$

where α is a constant and T_0 = 300 K. In the ON state, where $V_{GS} > V_{th}$, I_{DS} increases as temperature gets higher due to reduced energy band. In this situation [54], I_{DS} can be expressed as:

$$I_{DS}(T) = I_{DS_0}\sqrt{\frac{T}{T_0}} \tag{2.8}$$

The look up table based model in [53] did not include temperature effect of TEFT. To account for the temperature variation of on and off current, combining Eq. 2.7 and Eq. 2.8 together, the I_{DS} of model in [53] can to modified as:

$$I_{DS} = I_{DS_0}\frac{1}{1 + e^{-C(V_{GS}-V_{th})}} + [I_{DS_0} + \alpha(T - T_0)]\frac{1}{1 + e^{-C(V_{GS}-V_{th})}} \tag{2.9}$$

where C is a constant and I_{DS_0} is I_{DS} at 300K. With the help of sigmoid function, we integrate both temperature dependence of on current and off current of TFET into the model. An extra simulation is performed to verify the temperature effect of TFET. We generate series of I_{DS}-V_{GS} curve using the modified model in a temperature ranging from -55°C to 125°C, as shown in Fig. 2.4.

2.2 TFET Based SAR ADC Design

A key difference between TFET and CMOS transistors is TFET transistor's is unidirectional conduction. As a result, some components need to be modified to keep the circuit function properly. Transmitting gates and D flip flops (DFF) are main blocks in digital part of the ADC.

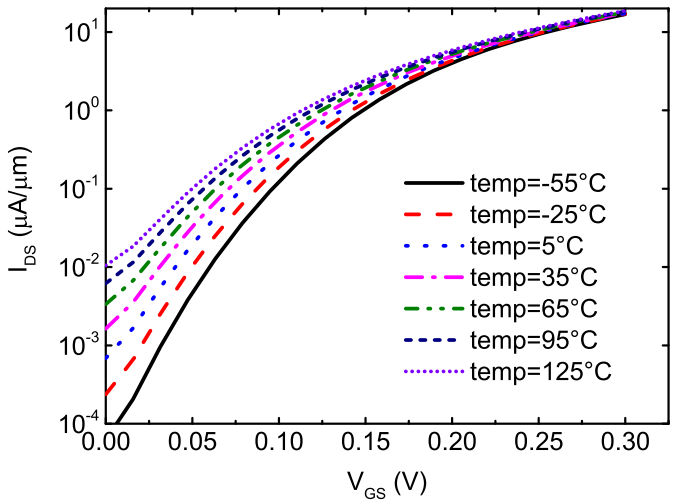


Figure 2.4: I_{DS}-V_{GS} characteristic of TFET

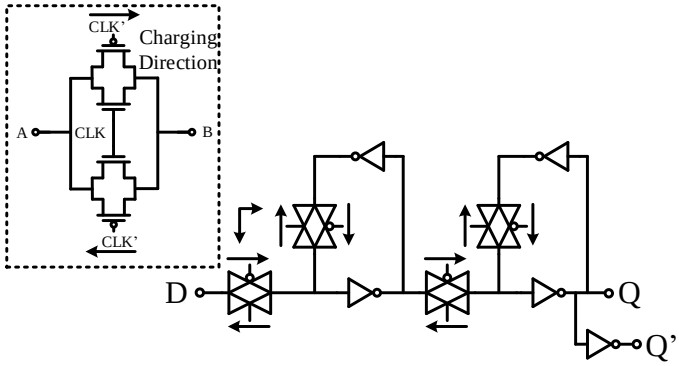


Figure 2.5: TFET version of transmitting gate and TFET version of DFF

Consequently, TFET version transmitting gates and DFFs need to be modified to address for the unidirectional conduction of TFET itself. In Fig. 2.5 two transmitting with opposite charging direction were put in parallel to perform bi-directional conduction. A comparison 20 nm CMOS and TFET transmitting gate as a sampling switch is shown in Fig. 2.6 and Fig. 2.7. A 10 kHz, rail-to-rail sinusoidal wave at VDD of 0.3V is sampled by CMOS and TFET transmitting gate, respectively, and the output signal is shown in Fig. 2.6.

Because that CMOS transistors' large Ron in sub-threshold region, CMOS switch fails to track the input signal.

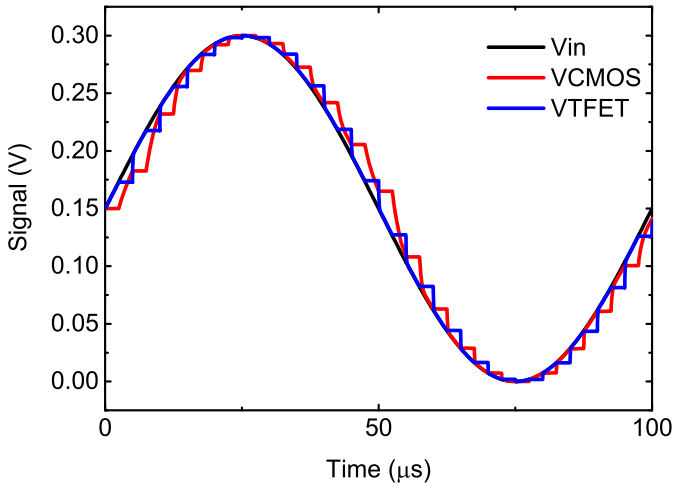
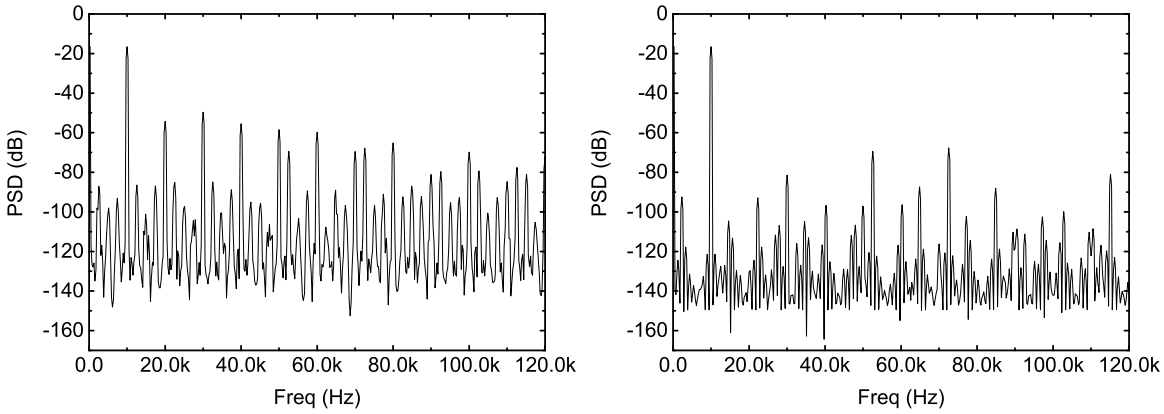


Figure 2.6: Comparison between CMOS and TFET transmitting gate as a sampling switch



(a) (b)

Figure 2.7: (a) output spectrum of CMOS switch and (b) output spectrum of TFET switch

Especially when input voltage is at the middle between VDD and GND rails, and will produce large harmonic distortion in output signal, as shown in Fig. 2.7a. To address this problem, the gate drive of the CMOS switch need to be boosted using complex bootstrapped switching schemes. On the other hand, TFET transistors have larger g_m/I_{DS} ratio and consequently a smaller Ron than CMOS counterparts. Therefore, TFET switches can track the input signal precisely without boosting the gate drive of the switch, resulting a 30 dB lower 3rd harmonic distortion than its CMOS counterpart, as shown in Fig. 2.7b.

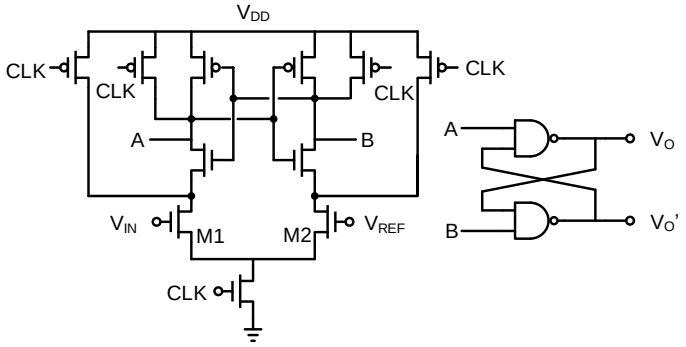


Figure 2.8: Schematic of the comparator omitting the tail current source

A single stage dynamic comparator is used due to its zero static power and high speed. Under ultra-low VDD, the tail current source is omitted to account for the shrinking voltage headroom, as shown in Fig. 2.8. When signal CLK is low, the comparator is resetting and signal "A" and "B" are connected to VDD. When CLK is toggling, the input transistor M1 and M2 will generate a differential current to trigger the latch. It can be observed that the comparator has no static power dissipation.

A 6-bit SAR ADC topology for VDD ranging from 0.3 V to 0.5 V was proposed [52]. However, the single ended structure has poor immunity to power supply noise and common mode level drafting. As a result, to guarantee the performance, other circuits such as low noise LDO and precise voltage reference are needed to provide supply and bias to the ADC, which increase the power dissipation. To address the influence of supply noise and common mode level drafting, a fully differential structure is introduced. Fully differential structure can also provide twice the input and output swing of the ADC/DAC than that in [52], which further improve the immunity of the supply noise by 6 dB. Moreover, fully differential structure can cancel even order distortion, which greatly improve ENOB of the ADC.

Fig. 2.9 shows principal blocks of the 6-bits SAR ADC including the DAC, comparator and control logic. In Fig. 10, $C_i = 2C_{i+1}$; $C_6 = C_C = 5$ fF, and the total capacitance used in the DAC is 640

fF.

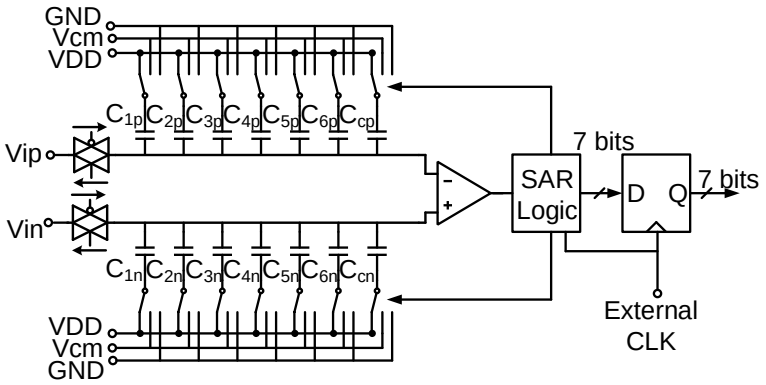
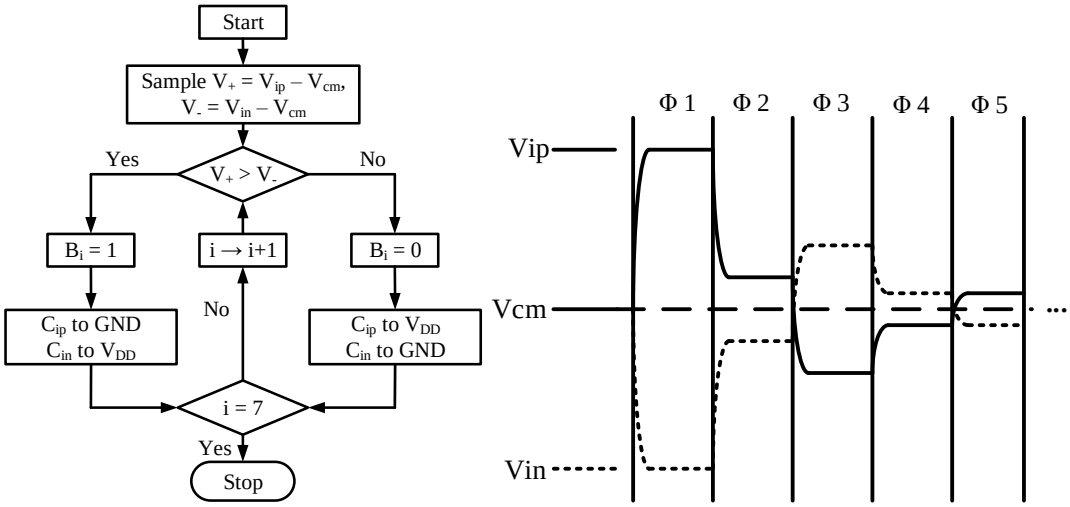


Figure 2.9: The principal blocks of the 6-bits SAR ADC

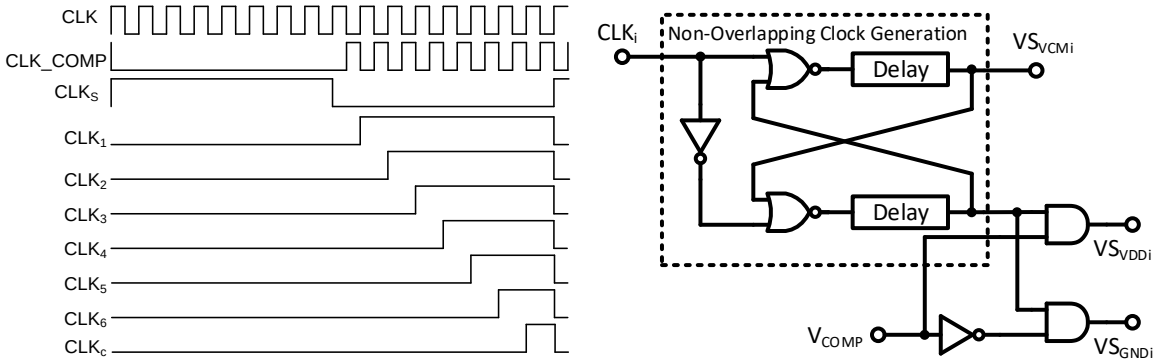
To make the maximum utilization of the supply voltage, the positive and negative voltage reference are VDD and GND, respectively, and V_{CM} is set to be VDD/2. Due to fully differential operation, noise on supply voltage can be cancelled and the circuit that generate VCM can be coarse thus reduce area and power dissipation of supporting circuit of the ADC. The input signal is sampled via TFET switches shown in Fig. 2.5. The feedback switches are also implemented with TFET switches to switch among VDD, GND and common level voltage V_{CM}. The comparator designed in Fig. 2.8 is adopted in Fig. 2.9 to perform low power signal comparison and generate decision signal to control SAR logic circuit. SAR logic module, which is shown in Fig. 2.11b, comprises TFET based logic gates, and generates the clock of all sampling switches and feedback switches. Finally, the output is saved by TFET based DFFs. The fully differential switch procedure is shown in Fig. 2.10a and the ideal DAC output is shown in Fig. 2.10b.

The clock scheme generated by SAR logic is shown in Fig. 2.11a. CLK is the external clock; CLK_COMP is the clock that triggers the comparator; CLKS is the sampling clock and CLK_i is the clock that control the feedback switch of C_i, as shown in Fig. 2.11b. The sampling period is eight clock cycles thus there is enough time for the sampling circuit to settle. When sampling clock is high, the comparator is disabled and all capacitor's bottom plate is connected to V_{CM}.



(a) (b)

Figure 2.10: (a) Fully differential switching process, and (b) ideal DAC output Voltage



(a) (b)

Figure 2.11: (a) timing diagram of the ADC, and (b) SAR logic schematic

When the sampling clock becomes low, the top plate of capacitor array is isolated and the comparator begins to compare the voltage on them. CLK_i will become high after i-th decision is made and switch the bottom plate of C_i to VDD or GND. In Fig. 2.11b, CLK_i is feed into a non-overlapping clock generation module to ensure that the bottom plate of the capacitor Ci will not connected to both V_{CM} and VDD/GND simultaneously. Signal VS_{VCMi}, VS_{VDDi} and VS_{GNDi} are the control signal for switches connect bottom plate of capacitor C_i to V_{CM}, VDD and GND, respectively. V_{COMP} is the output voltage of the comparator and can determine the bottom plate of Ci is switched to

VDD or GND.

2.3 Simulation Results

Transistor-level simulation and analysis of the TEFT based ADC is performed using Cadence Spectre with modified VerilogA TFET transistor model. A 20 nm CMOS based ADC is also designed by replacing all TFET transistors by 20 nm CMOS transistors (20nm PTM-MG spice model [56]). This CMOS based ADC is also simulated using Cadence Spectre to compare the performance of TFET and CMOS technology. The full range input to the ADC are two sinusoid waves of peak-to-peak value of VDD and the phase difference is 180°, making the differential mode peak-to-peak value of full range input signal 2VDD. The minimum TFET transistor length is 20 nm. Both the VerilogA model for TFET and PTM-MG model include parasitic capacitance [53], [56]. To explore the TFET benefits in subthreshold region, the typical supply voltage is set to be 0.3V. The typical temperature is 25 °C. Power dissipation of the ADC is measured in terms of energy, which is defined as Energy = Power / Sampling Frequency.

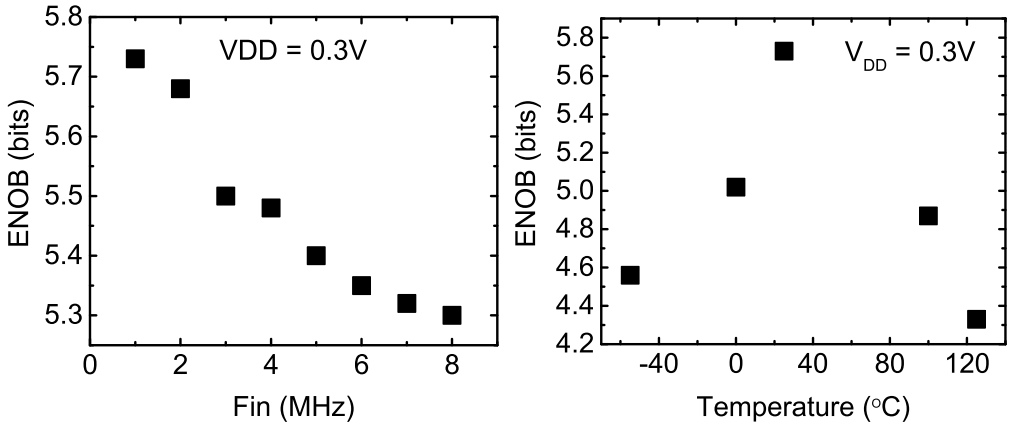


Figure 2.12: (a) ENOB vs. input frequency, and (b) ENOB vs. temperature

Fig. 2.12a shows the effective number of bits (ENOB) with respect to input frequency. A continuous degradation of ENOB with respect to in frequency is observed due to increased nonlinearity and electrical and/or quantization noise. When input signal is a 1 MHz sinusoid wave with peak-to-peak voltage of 0.3 V, the ADC can obtain an ENOB of 5.7 bits. If the frequency of the sinusoid wave is increased to 8 MHz, the ENOB will decrease to 5.3 bits. Fig. 2.12b shows the ENOB with respect to temperature. The frequency of input sinusoidal wave is 1M Hz. When temperature is 125°C, the ENOB is reduced from 5.7 bits to 4.3 bits because of reduced charge mobility. When temperature is -55°C, the ENOB reduced to 4.5 bits due to the increased leakage current and threshold voltage. The ENOB variation in temperature ranging from -55°C to 125°C is 24%.

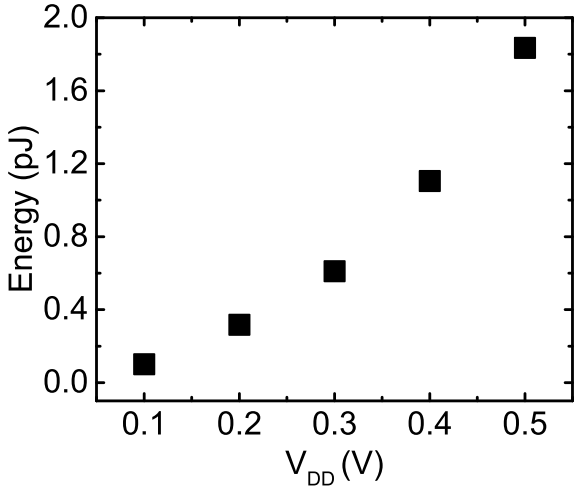
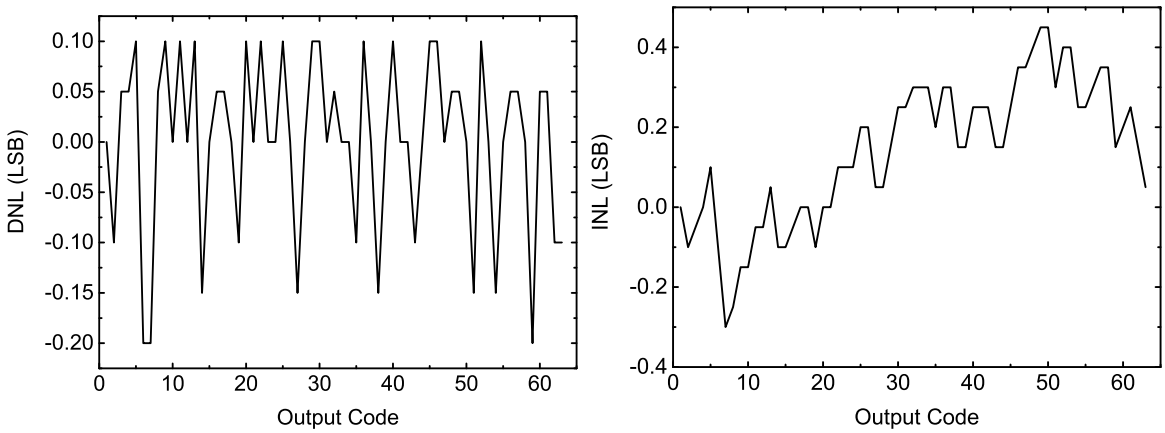


Figure 2.13: Energy vs VDD

Fig. 2.13 shows the energy of the ADC with respect to VDD. The ADC energy is obtained by setting sampling frequency to the maximum that the comparator can make correct decision. The ADC energy increases almost quadatically versus VDD. The ADC can achieve an energy of 0.1 pJ at the supply voltage of 0.1 V. When VDD increases to 0.5V, the ADC energy will increase to 1.8 pJ.

Also, INL and DNL of the ADC is simulated at typical conditions, And the simulation result is

shown in Fig. 2.14.



(a) (b)

Figure 2.14: (a) DNL and (b) INL of the ADC

Fig. 2.14 shows the differential nonlinearity (DNL) (Fig. 2.14a) and the integral nonlinearity (INL) (Fig. 2.14b) performance. The SAR ADC has the maximum $+0.1/-0.2$ LSB DNL and $+0.4/-0.3$ LSB INL.

To compare the performance of TFET and 20 nm CMOS technology, both TFET based ADC and CMOS based ADC are simulated for ENOB and energy under the same condition.

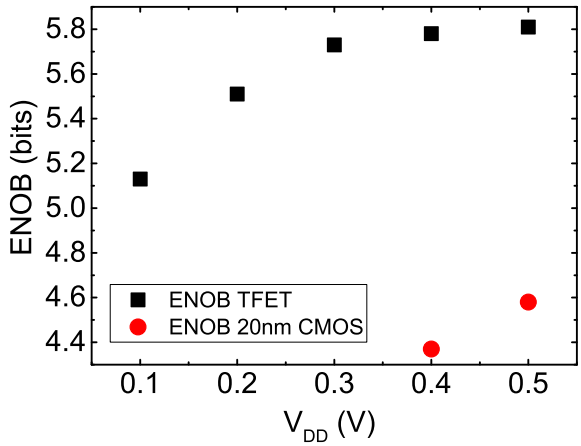


Figure 2.15: ENOB vs VDD of TFET and CMOS based ADCs

Fig. 2.15 shows the ENOB of both TFET based ADC and CMOS based ADC with respect to VDD. When VDD is 0.1 V, the ENOB of the TEFT based ADC is 5.1 bits. The ENOB increase rapidly with the VDD and achieves 5.7 bits when VDD is 0.3 V. When VDD is above 0.3 V, the ENOB become saturated and become 5.8 bits when VDD is 0.5 V. Because the large on resistance of CMOS transistor, the CMOS based ADC cannot work with VDD under 0.4V. For VDD is 0.4 V and 0.5 V, the ENOB of CMOS based ADC is 4.4 bits and 4.8 bits, respectively. The difference is made by the nonlinearity introduced by sampling switches, as shown in Fig. 2.6 and Fig. 2.7.

Fig. 2.16 shows the energy of the both ADCs with respect to temperature. At -55°C, the energy of the TFET based ADC will increase due to increased sub-threshold leakage current. On the other hand, at 125°C, the energy of the TFET based ADC will also increase due to reduced carrier mobility. At 25°C the energy of TFET based ADC is 0.61 pJ. The energy variation in temperature ranging from -55°C to 125°C is 31.5%. On the other hand, CMOS based ADC also suffer from increased sub-threshold leakage current at low temperature and reduce carrier mobility at high temperature. At 25°C the energy of CMOS based ADC is 0.95 pJ, 55% larger than TFET based ADC. The energy variation in temperature for CMOS ADC ranging from -55°C to 125°C is 81%.

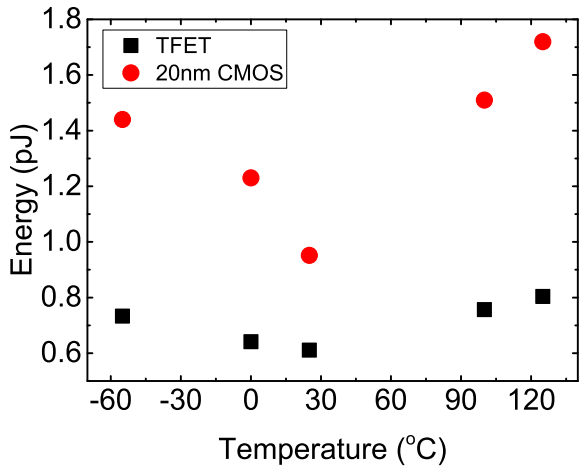


Figure 2.16: Energy vs. temperature

Finally, Comparison between TFET and reported CMOS ADCs [57] is made, as shown in Fig. 2.17. From Fig. 2.17, the TFET based SAR ADC can achieve energy 3 times lower than state of art CMOS ADCs.

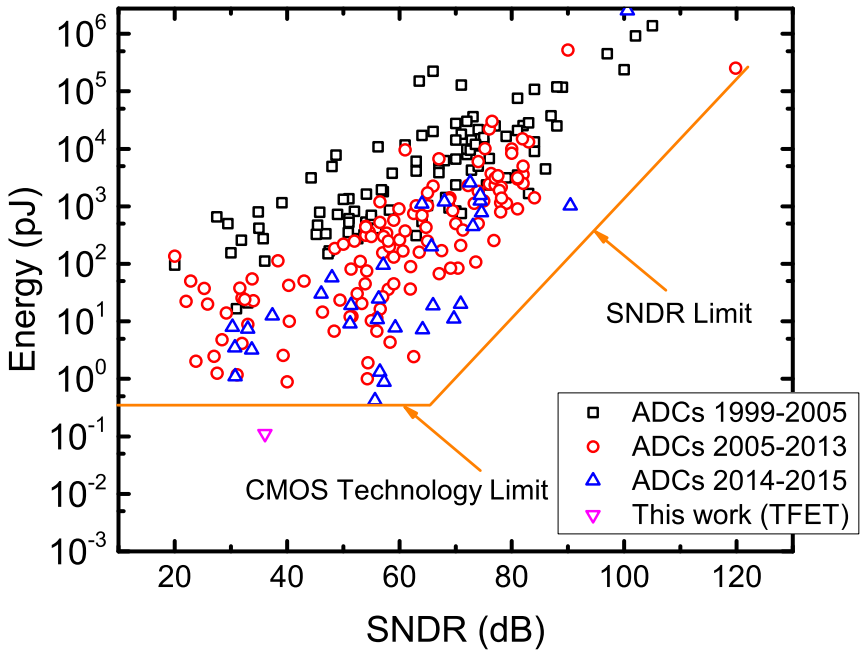


Figure 2.17: ADC energy comparison between CMOS and TFET

2.4 Conclusion

A TFET based 6-bit SAR ADC is designed and evaluated. The ADC can achieve an ENOB of 5.7 to 5.3 with input frequency ranging from 1 to 8 MHz, and The ENOB variation in temperature ranging from -55°C to 125°C is 24%. Under the VDD of 0.1V, energy is 0.1 pJ with 31.5% variation with temperature ranging from -55°C to 125°C.

CHAPTER 3: A 12-BIT ULTRA-LOW VOLTAGE NOISE SHAPING SAR ADC USING EMERGING TUNNELING FETS

[1]During the era of Internet of Things (IoT), tremendous sensor nodes are self-powered and they impose great challenges for circuit designers. Analog-to-Digital converters (ADCs) are essential functional block for sensor interfaces that digitize the analog signal from the sensor for subsequent digital processors. Most of the power supply of the sensor nodes—the harvesting devices such as solar cells—can only generate extremely low output voltage that is usually under 0.5 V, and limited power. Therefore, ultra low-voltage and low-power operation is inevitable for wireless sensor nodes [58]. The output of the sensor usually need to be digitized by a ADC with moderate resolution (10–12 bits) and bandwidth (1–1000 kHz). Moreover, output signal of the signal is also usually small, even at the level of micro volts [59]. In these applications, ADCs are the most critical and power hungry blocks. Among various ADC architectures, successive approximation register (SAR) ADC shows a better power efficiency [58, 60, 61]. Furthermore, SAR ADC can be opamp-free and thus easily benefit from technology advancing and can have no static power. These reasons arouse many researches on exploring SAR ADC in depth.

However, the accuracy of SAR ADC is hard to reach resolution over than 10 bits due to fundamental and related second-order effects. As with all ADCs, KT/C noise limits sampling accuracy. For moderate resolution ADCs, the minimum capacitance to achieve sufficient low sampling noise is usually larger than that the required capacitance needed to achieve adequate matching. In addition, the number of unit capacitance increase exponentially with the accuracy of the ADC, leaves great difficulty for layout matching and parasitic reduction. To Solve this problem, a significant advan-

[1]This chapter was published as Lin, Jie, and Jiann-Shiun Yuan. "A 12-bit ultra-low voltage noise shaping successive-approximation register analogto-digital converter using emerging TFETs." *Journal of Low Power Electronics* 13.3 (2017): 497-510.

tageous method is to use oversampling because it attenuates noise KT/C noise. But without noise shaping, oversampling is usually unattractive. Noise shaping has been efficiently demonstrated in SAR ADCs [60–62]. Those works only shape noise to the first order noise transfer function, with the set-back of limited attenuation at low frequency and less degree of freedom in parameter design.

In this work, for the first time, we proposed a 2nd order noise shaping SAR (NSSAR) ADC that provides 20 dB more attenuation than the first counterparts. By optimize the design parameters of the ADC, the KT/C noise generated by the integrators are reduced, leading to a reduced area and power consumption. Under ultra-low VDD, energy efficiency is limited by CMOS technology for its 60 mV/dec sub-threshold slope (SS) [63]. As a result, tunneling field effect transistor (TFET), with its superior IDS-VGS characteristic, is strong candidate for ultra-low-power and ultra-low-VDD design. Using the TFETs we can realize the 2nd order NSSAR and achieve an over 10 bits' accuracy that not reported before under ultra-low VDD.

The chapter is organized as follows. In Section II, the architecture of the proposed NSSAR ADC is introduced and analyzed. Section III describes the modeling of the TFETs. Section IV discuss circuit design using TFETs for the NSSAR. The simulation results and comparison with previous works are presented in Section V. Finally, the conclusions are drawn in Section V.

3.1 Second-Order Noise Shaped SAR (NSSAR) ADC Design

3.1.1 System Design

The SAR ADC can be treated as a zero order sigma-delta modulator without any form of noise shaping. As a result, noise shaping can be realized by insert filters into the signal path [61, 62]. In principle, the only active block necessary in a passive loop filter is the comparator of the SAR

ADC and avoid other active circuits (amplifiers or buffers). Thus the passive loop filter is a suitable choice for ultra-low power, ultra-low VDD operation. Due to the feedback path of the ADC is primarily defined by the SAR algorithm, feed-forward sigma-delta architectures are promising candidacies to for NSSAR ADCs. Moreover, since the input signal to the loop filter is only the shaped quantization noise, the requirements on the linearity of the loop filter is greatly reduced. As a result, the influence of parasitic capacitance in the passive integrator is addressed by the feed-forward architecture. The signal-flew diagram of a first order NSSAR ADC [61] is shown Fig. 3.1.

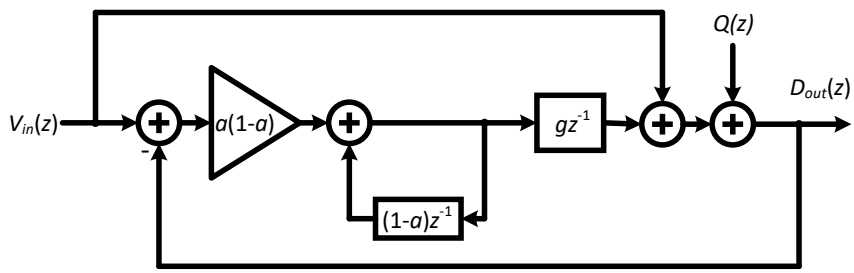


Figure 3.1: Signal flew diagram of a first order NSSAR ADC

From Fig. 3.1 The transfer function of the first order NSSAR ADC and the stability condition for g are [61]

$$D_{out}(z) = V_{in}(z) + \frac{1-(1-a)z^{-1}}{1-(1-a)(1-ga)z^{-1}}Q(z) \tag{3.1}$$

and

$$-\frac{1}{1-a} < g < \frac{2-a}{a(1-a)} \tag{3.2}$$

where a is the noise leakage caused by limited DC gain of the passive integrator and g is a constant that used to control the location of the pole in the noise-transfer-function (NTF). From Eq. 3.1 and Eq. 3.2 the minimum DC gain of NTF while keeping the system stable is $a/2$. Then, for the first order modulator, the only way to decrease the DC gain of NTF is to increase the DC gain of the passive integrator, at the cost of excessive area and power consumption.

An alternative way to lower the DC gain of NTF is to increase the order of noise shaping in the NTF. Based on the feed-forward architecture, we propose a second order NSSAR ADC with the signal-flew diagram shown in Fig. 3.2. In Fig. 3.2, another passive integrator is insert between the quantizer and the first integrator. Theoretically, using this architecture, any order of noise shaping can be realized. However, parasitic capacitors in the switches will dramatically increase the leakage in the integrator in the later stages. Thus, we chose to only use 2nd order architecture. We kept the feed-forward path from the input to the quantizer and the single feedback path from the digital output to the input that relax the linearity requirement for the loop filter and make the system level design more correspond to circuity realization based on the SAR ADC. Moreover, 2nd order architecture provides more degree of freedom to for systemic optimization in the system design phase.

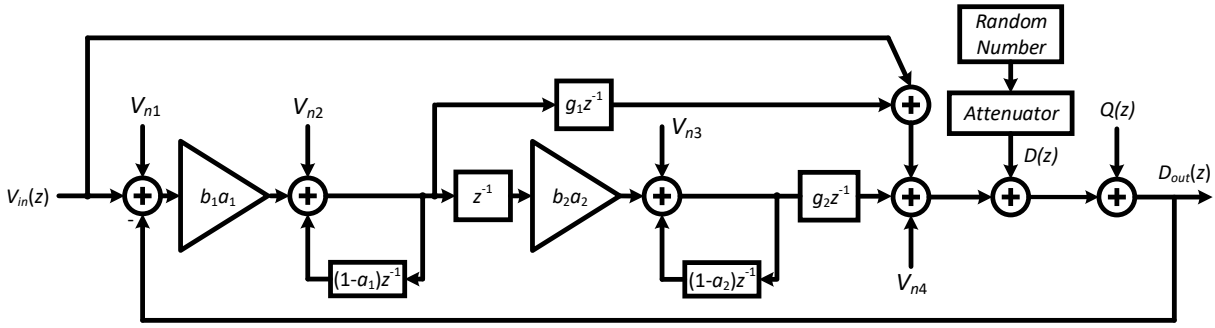


Figure 3.2: Signal flew diagram of the second order NSSAR ADC propose in this paper

From Fig. 3.2, the transfer function of the 2nd order NSSAR ADC is

$$D_{out}(z) = V_{in}(z) + \frac{[1-(1-a_1)z^{-1}]\,[1-(1-a_2)z^{-1}]}{1+Az^{-1}+Bz^{-2}}\,[Q(z)+D(z)] \tag{3.3}$$

where Where $Q(z)$ is the quantization noise, $D(z)$ is the dither signal, which we will mention later,

and A and B are

$$\begin{aligned} A &= -2 + a_1 + a_2 + a_1 b_1 g_1 \\ B &= 1 - a_1 - a_2 + a_1 a_2 - (a_1 b_1 - a_1 a_2 b_1) g_1 + a_1 a_2 b_1 b_2 g_2 \end{aligned} \tag{3.4}$$

Thus, the NTF has two zeros and two pols as

$$\begin{aligned} Z_1 &= 1 - a_1 \\ Z_2 &= 1 - a_2 \\ P_1 &= \frac{-A - \sqrt{A^2 - 4B}}{2} \\ P_2 &= \frac{-A + \sqrt{A^2 - 4B}}{2} \end{aligned} \tag{3.5}$$

The stability condition for the NTF is then $|P_1| < 1$ and $|P_2| < 1$. Moreover, from Eq. 3.3 to 3.5, when $A = 2$ and $B = 1$, the system has the minimum DC gain of NTF while being stable. Under this condition, the DC gain of the NTF is $a_1 a_2/4$, improved by a factor of $a_2/2$ comparing to the first order NSSAR. From Eq. 3.5, we can summarize a procedure to determine parameter g_1 and g_2.

1. First, determine the location of the poles as P_1 and P_2.
2. Second, using $A = -(P_1 + P_2)$ to get the value of A and hence calculate the value of g_1 using Eq. 3.4.
3. Third, using Eq. 3.5 to calculate B and using Eq. 3.4 to obtain the value of g_2.

Moreover, from Eq. 3.4 and Eq. 3.5, a rule of thumb can be summarized that to make max utilize of the poles that can obtain a larger attenuation to the $Q(z)$, while keeping system stable, parameter g1 and g2 need to be select as large as possible.

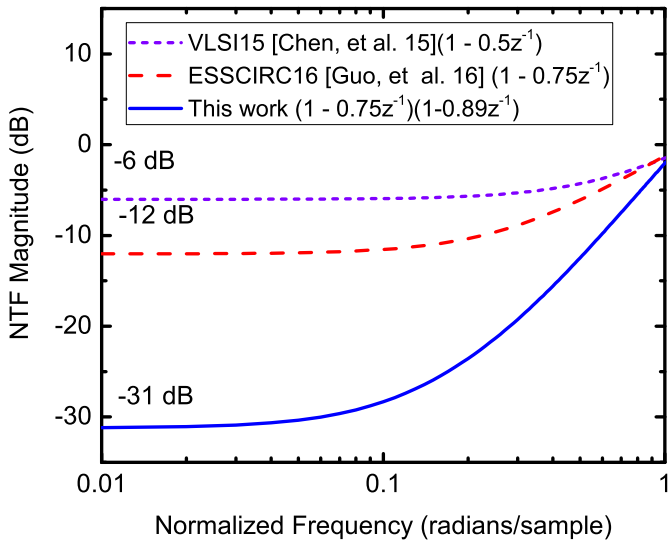


Figure 3.3: Comparison of NTF of this work with previous publication

In Fig. 3.3, we plot the magnitude response of NTF of 2nd NSSAR (all poles are at origin point) with the parameter that $a_1 = 0.11$, $a_2 = 0.25$, and compared it with NTF of previous work. The 2nd NSSAR provides an extra 19 dB of attenuation at low frequency comparing to previous first order works, indicating an increase of ENOB of over 2.8 bits.

System level simulation of the 1st and 2nd order NSSAR is performed using matlab. We use a 6-bit quantizer in the simulation to model the original 6-bit SAR ADC. To manifest the influence of limited cycles, we set the input level to be 0.01 V_{ref}. The parameters of the ADCs are shown in Table 3.1. From Eq. 3.1 and Eq. 3.3 to Eq. 3.5 the zeros and poles of the transfer function are $Z_{1st} = 0.8$, $P_{1st} = 0$, $Z_{2nd1} = 0.8$, $Z_{2nd2} = 0.89$, $P_{2nd1} = -0.58$ and $P_{2nd2} = 0.02$. Therefore, the attenuation of the NTF at low frequency will increase by a factor of $|P_1 - 1||P_2 - 1|$, which is 3.9 dB. Moreover, poles are far from unit disk, providing a save margin for system stability.

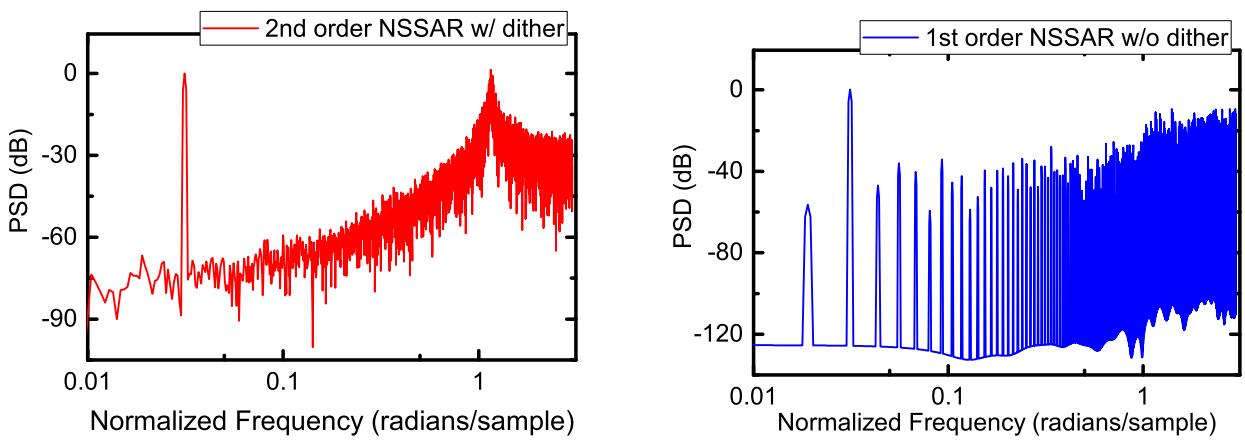
Fig. 3.4 shows the output power spectrum density (PSD) using matlab. The magnitude is normalized with respect to the input signal. The output of the 1st order NSSAR (see Fig. 3.4b) shows series of discrete peaks in the frequency spectrum caused by limited cycles. As mentioned before,

this peaked noise spectrum is problematic in certain applications such as artificial cochlear.

Table 3.1: Parameters Used in System Simulation

Order	Symbol	Quantity
First	a	0.25
	g	4
Second	a_1	0.11
	a_2	0.25
	b_1	0.8
	b_2	0.88
	g_1	25
	g_2	50

On the other hand, with the dither and 2nd noise shape, the weak signal can be detecting with reasonable signal to noise ratio which is over 60 dB.



(a) (b)

Figure 3.4: (a) output PSD of the 2nd order NSSAR and (b) of 1st order NSSAR

3.1.2 Circuit Realization of the NSSAR

Based on the principle of the proposed transfer function, a NSSAR ADC was implemented. In order to address the input swing limited by the ultra-low VDD, fully differential is adopted to

double the input swing at the cost of more silicon area occupied.

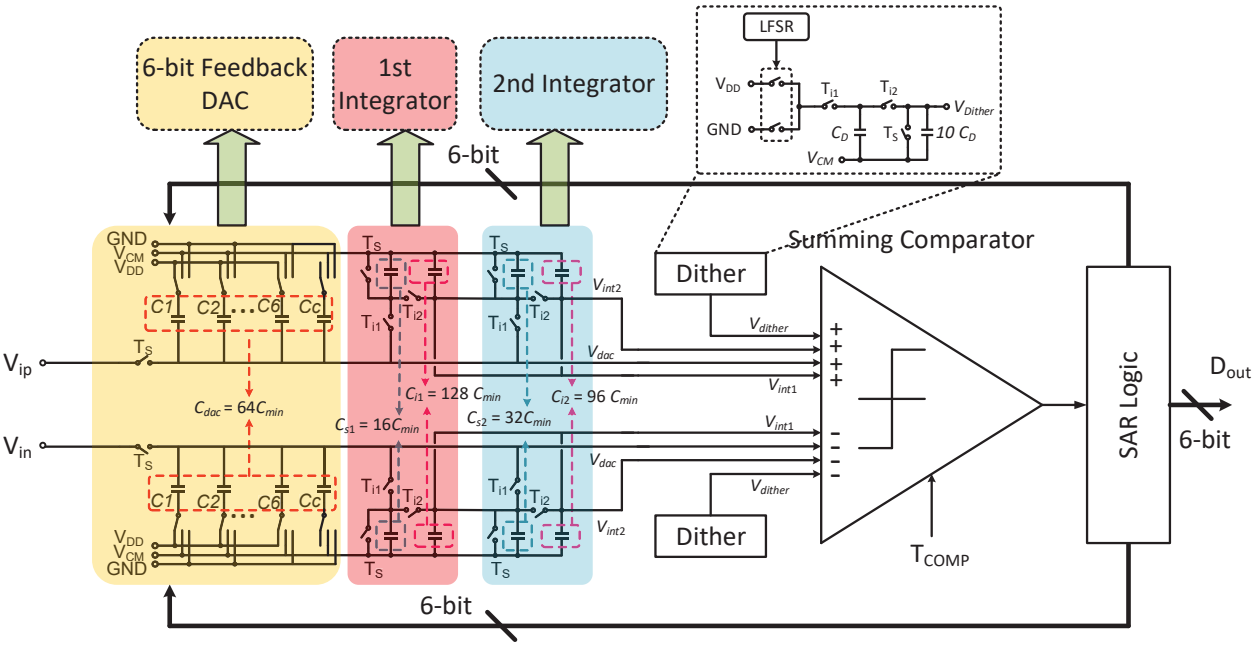


Figure 3.5: Schematic of the 2nd order noise shaped SAR ADC with dither injection

The designed ADC comprises a 6-bit SAR ADC [64] and a second order passive integrator. we make one extra switching of the DAC array C_C based on the final comparator decision so that the residue is based on the full resolution of digital estimation [62]. Moreover, quantizer and the feedback DAC use the same capacitor array in the NSSAR ADC, henceforth, the DAC mismatch error transfer function (ETF) is always 1 and the mismatch error can be easily estimated and calibrated in digital domain [61]. The applied coefficients are the same as those used in Fig. 3.2, and the sampling frequency is 1.38 MHz with the maximum input bandwidth of 43.1 kHz. The oversampling ratio (OSR) is 16. The schematic of the ADCs is shown in Fig. 3.5.

The schematic of the two-stage state integrator is shown in Fig. 3.6. To ensure that the integrator works properly, the sampling capacitors are reset every operation cycle. To derive the transfer function of integrator, firstly consider the passive integrator comprising C_{dac}, C_{s1}, C_{i1} and corre-

sponding switches, and we have the follow equation:

$$V_{out1st}[n] = \frac{C_{i1}}{C_{i2} + Cs1} V_{out1st}[n-1] + \frac{C_{s1}}{C_{i1} + C_{s1}} \frac{C_{dac}}{C_{dac} + C_{s1}} V_{in}[n-0.5] \tag{3.6}$$

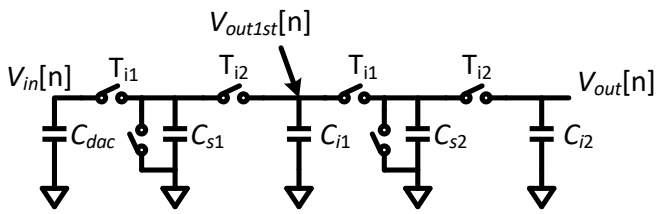


Figure 3.6: Schematic of two-stage integrator

Consider that the sampling frequency is much higher than the signal bandwidth, we can safely assume that $V_{in}[n-0.5] = V_{in}[n]$. Consequently,

$$\frac{V_{out1st}(z)}{V_{in}(z)} = \frac{b_1 a_1}{1 - (1 - a_1) z^{-1}} \tag{3.7}$$

where $a_1 = C_{s1}/(C_{i1} + C_{s1})$ and $b_1 = C_{dac}/(C_{s1} + C_{dac})$. In our design, we set $C_{i1} = 2C_{dac}$. And the sampling capacitor $C_{s1} = 1/8Ci1$. Thus it follows that $a_1 = 0.11$ and $b_1 = 0.8$, the same as shown in Table 3.1. The same as Eq. 3.6, we get the equation for the second stage integrator as

$$V_{out}[n] = \frac{C_{i2}}{C_{i2} + C_{s2}} V_{out}[n-1] + \frac{C_{s2}}{C_{i2} + C_{s2}} \frac{C_{i1}}{C_{i1} + C_{s1}} V_{out1st}[n-0.5] \tag{3.8}$$

Also, it is safe to assume that $V_{out1st}[n-0.5] = V_{out1st}[n]$ still holds. Using Eq. 3.8, we can obtain that

$$\frac{V_{out}(z)}{V_{out1st}(z)} = -\frac{b_2 a_2}{1 - (1 - a_2) z^{-1}} \tag{3.9}$$

where $a_2 = C_{s2}/(C_{i2} + C_{s2})$ and $b_2 = C_{i1}/(C_{s2} + C_{i1})$. To obtain the value in Table 3.1, we set $C_{s2} = 2C_{s1}$, $C_{i2} = 0.75C_{i1}$. Henceforth, $a_2 = 0.11$, $b_2 = 0.8$. Henceforth, the NTF shown in Fig. 3.4 is synthesized by the two-stage passive integrator and the capacitor ratio we used.

In [61], W. Guo and N. Sun proposed the noise analysis of the 1st order NSSAR ADC. In our work, we extend their work from investigating the noise performance under specified system parameters, $g = 1/a$, to more general system parameters. Moreover, combined with analysis in the previous subsection, we provide a detailed optimization for system parameters that leads to better performance of the 2nd order NSSAR. In Fig. 3.2, V_{n1} is the KT/C from noise of the DAC capacitors, V_{n2} is the noise sampled by C_{i1}, V_{n3} is the noise sampled by C_{i2} and V_{n4} is the input referred noise of the summing comparator. Using Eq. 3.7 and Eq. 3.9, expression for V_{n1} to V_{n3} can be written as

$$\begin{aligned}
\overline{V}_{n1}^2 &= \frac{KT}{C_{dac}} \\
\overline{V}_{n2}^2 &= \frac{a_1 KT}{c_{i1}} + \frac{a_1^2 b_1^2}{1-b_1}\frac{KT}{C_{dac}} = \frac{a_1(0.5 - 0.5b_1 + a_1 b_1^2)}{1-b_1}\frac{KT}{C_{dac}} \\
\overline{V}_{n3}^2 &= \frac{a_2 KT}{c_{i2}} + \frac{a_2^2 b_2^2}{1-b_2}\frac{KT}{C_{i1}} = \frac{a_2(0.66 - 0.66b_2 + 0.5a_2 b_2^2)}{1-b_2}\frac{KT}{C_{dac}}
\end{aligned} \tag{3.10}$$

where K is the Bozeman constant and T is the absolute temperature. Using Fig. 3.2, we can derive the transfer function of the ADC including noise source as

$$\begin{aligned}
V_{out}(z) = V_{in}(z) &+ V_{n1}(z) \\
&+ \frac{g1\left[1-(1-a_1-a_1a_2b_1b_2g_2/g_1)z^{-1}\right]z^{-1}}{1+Az^{-1}+Bz^{-2}}V_{n2}(z) \\
&+ \frac{g_2\left[1-(1-a_2)z^{-1}\right]z^{-1}}{1+Az^{-1}+Bz^{-2}}V_{n3}(z) \\
&+ \frac{\left[1-(1-a_1)z^{-1}\right]\left[1-(1-a_2)z^{-1}\right]}{1+Az^{-1}+Bz^{-2}}\left[Q(z)+D(z)+V_{n4}\right]
\end{aligned} \tag{3.11}$$

In Eq. 3.11 V_{n4} is shaped by the same NTF of the quantization noise. Therefore, it can be omitted during circuit design. V_{n1} is directly added to the input signal and not shaped by the loop filter. The NTF of V_{n2} and V_{n3} are closely related to parameter g_1 and g_2 and, unlike the NTF of the quantization noise, the larger the two parameter, the larger the NTFs are. Hence, the value needs carefully tuned to achieve an optimal signal to noise ratio for the ADC. To accomplish that, we use four quantities to illustrate the influence of the g_1 and g_2 on the performance of the ADC. The

first one is $|P_1 - 1||P_2 - 1|$. As we mentioned in pervious subsection, $|P_1 - 1||P_2 - 1|$ shows how much the will poles affect the attenuation of the quantization noise. Fig. 3.7a shows the variation of $|P_1 - 1||P_2 - 1|$ versus the value of g_1 and g_2. As we predicted, when g_1 and g_2 become larger, the attenuation coming out of the poles increases, displaying a darker color in Fig. 3.7a.

The second one is the NTF for the integrator noise, V_{n2} and V_{n3}, to the output of the ADC, which can be written as

$$NTF_i = \sqrt{\alpha_{n2} NTF_{n2}^2 + \alpha_{n3} NTF_{n2}^3} \tag{3.12}$$

where

$$\begin{aligned} \alpha_{n2} &= \frac{a_1(0.5 - 0.5b_1 + a_1 b_1^2)}{1 - b_1} \\ \alpha_{n3} &= \frac{a_2(0.66 - 0.66b_2 + 0.5a_2 b_2^2)}{1 - b_2} \\ NTF_{n2} &= \frac{g_1\left[1 - (1 - a_1 - a_1 a_2 b_1 b_2 g_2/g_1)z^{-1}\right] z^{-1}}{1 + Az^{-1} + Bz^{-2}} \\ NTF_{n3} &= \frac{g_2\left[1 - (1 - a_2)z^{-1}\right] z^{-1}}{1 + Az^{-1} + Bz^{-2}} \end{aligned} \tag{3.13}$$

From Eq. 3.10 to Eq. 3.13 it can be observed that NTF_i is proportional the square of capacitance used in the ADC and hence the square of area and power consumption. As a result, it is critical to reduce the NTF_i to improve the power and area efficiency of the ADC, particularly in ultra-low VDD and ultra-low power. Fig. 3.7b shows the value of NTF_i at DC versus g1 and g2. When g1 and g2 are small, the color in Fig. 3.7b is darker, revealing a decreasing in NTF_i.

The last two quantities are $|P_1|$ and $|P_2|$, which gives the criterion whether the system is stable. In previous subsection, we show that to make sure that the system is stable, it should be satisfied that $|P_1| < 1$ and $|P_2| < 1$. In Fig. 3.7c and Fig. 3.7d, we plot the value of $|P_1|$ and $|P_2|$ changing with g_1 and g_2. Also, we highlight the boundary where the system is stable. To ensure the system is stable, the data point for system parameters must locate near the darkest area in the figures to guarantee a sufficient safe margin. Combining the four figures of merits, the optimized g_1 and g_2

can be chosen by putting the data point in the dark portion in Fig. 3.7a to Fig. 3.7d. In our work we set $g_1 = 25$ and $g_2 = 50$, as the red triangle in Fig. 3.7.

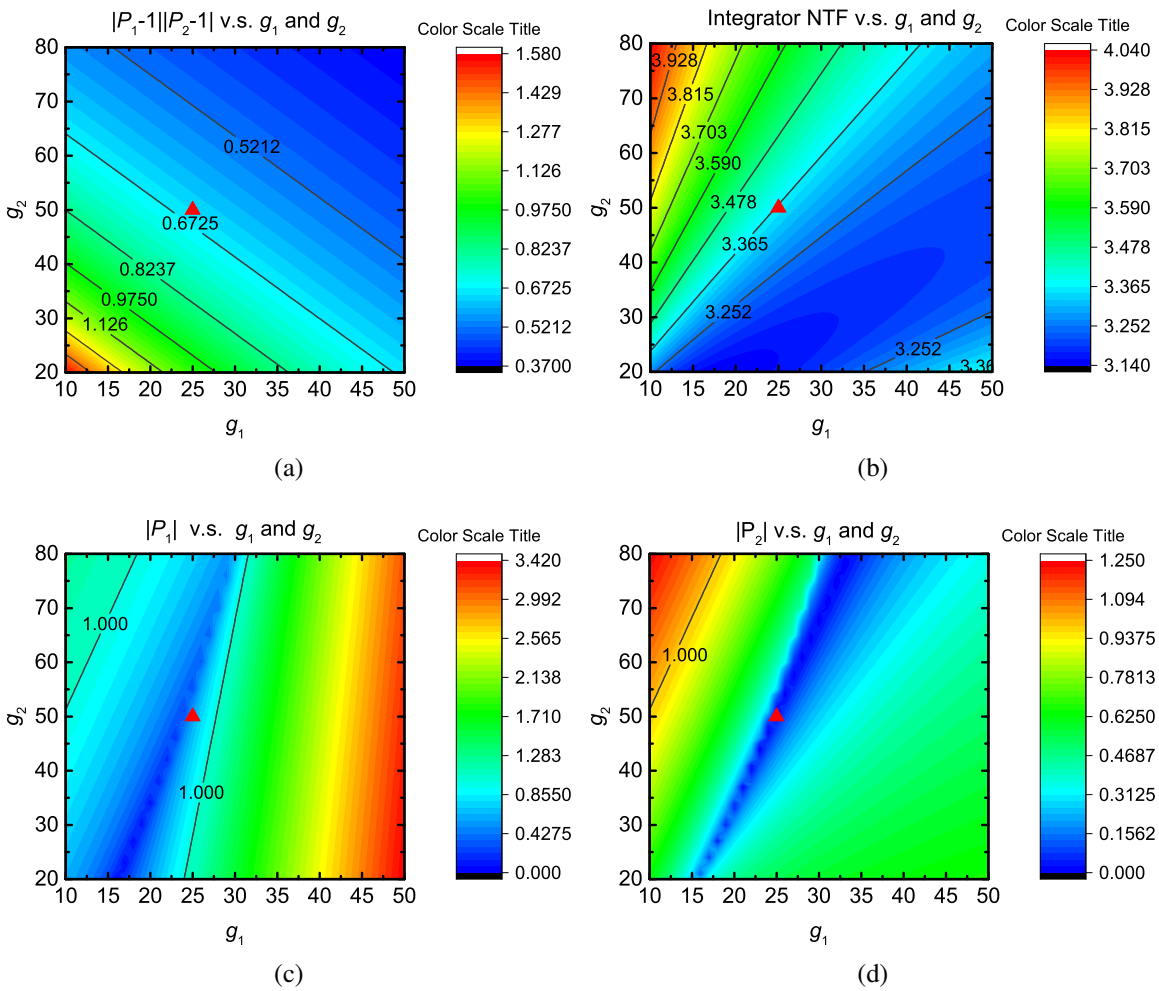


Figure 3.7: Figures of merits to choose g_1 and g_2. (a) $|P_1 - 1||P_2 - 1|$, (b) NTF of the integrator, (c) $|P_1|$ and (d) $|P_2|$. The red triangle show the selection of g_1 and g_2 that $g_1 = 25$, $g_2 = 50$.

To determine the value of capacitance used in the NSSAR, we utilized the NTF_i. So, the total electrical noise at the output of the ADC is

$$\overline{V}_{tot}^2 = (1 + NTF_i^2)\frac{KT}{C_{dac}} \tag{3.14}$$

The number of bits and the dynamic range of the converter are related by [65]

$$\frac{V_s^2}{\overline{V}_{tot}^2} = \frac{(2V_{ref})^2/2}{(1+NTF_i)KT/C} = \frac{\frac{3}{2}(2^{ENOB}-1)^2}{OSR} \tag{3.15}$$

where V_s is the signal, V_{ref} is the reference voltage. Using Eq. 3.15, and letting ENOB = 12, we can calculate the C_{dac} is 444.3 fF. Henceforth, We set C_{dac} = 480 fF in this design.

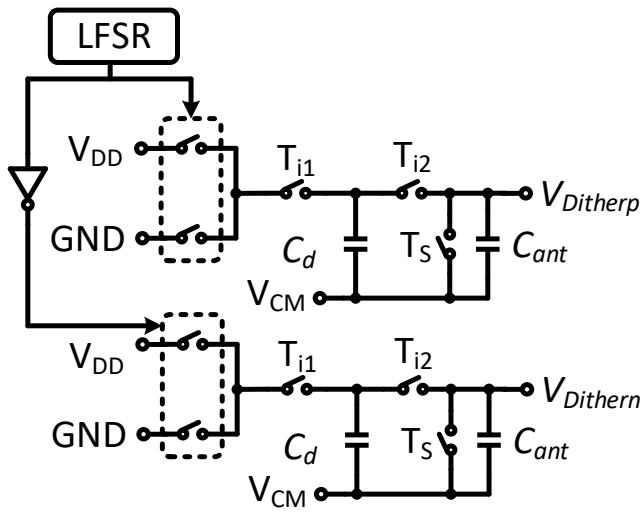


Figure 3.8: Schematic of dither circuit

Limited-cycles is problematic phenomenon that rises in the sigma-delta when input level is low [66, 67]. More seriously, when ADCs are applied in audio applications, limit cycles can result in audible artifacts. To eliminate the influence of limited cycle, a dither circuit is added into the signal path as shown in Fig. 3.2. Different from ordinary paradigms, an attenuator is employed to reduce the amplitude of the dither signal to accommodate the limited low frequency attenuation of the NTF. Given that the DC gain of NTF of the 2nd order NSSAR is around 30dB, to ensure that the ADC has an ENOB over 12-bits, the attenuation factor is -40 dB in our design. Fig. 3.8 shows a detailed schematic of the dither circuit.

Although self-dither technology is proposed by previous works [68], the dither is somehow related to the input signal and the randomness of the dither may be jeopardized. Consequently, in Fig.

3.8, A linear feedback shift register (LFSR) provides the pseudo-random number to switch the C_d between VDD and GND. An attenuation capacitor $C_{ant} = 10C_d$ is connected in parallel with the Cd. Due to charge sharing the magnitude of the dither voltage is reducing to 0.09 V_{dither}. Another 20 dB attenuation is realized in the summing comparator, which will be discussed in the following section.

3.2 Implementation of the Building Blocks Using TFETS

Simply replacement of CMOS transistors with TFETs does not necessarily lead to a performance improvement. This is caused by the unique behaviors of TFETs, such as ambipolarity and asymmetry as mentioned in the Chapter 2. In following subsections, we describe the implementation of key building blocks of the ADC with TFET.

3.2.1 Clock Generating and SAR Logic Design Using TFETs

In the circuit schematic, the clock generating circuit and SAR logic block is the main digital block of the circuit generating the control bits according the output of the comparator. The asynchronous clocking paradigm proposed in [69] could optimize the comparing interval and lead a faster operation speed. However, the control signal is determined by the comparator state. As a result, we use synchronous clock scheme that is easy to generate extra control signal for passive integrators when comparator is not working. The clock generate circuit is shown in Fig. 3.9.

In Fig. 3.9, clk_{ext} is the external clock with the frequency of 25 MHz. Signal clk_{sar}, clk_{int1} and clk_{int2} are generated based on clk_{ext} using a 4-bit counter and corresponding combinational logic. A non-overlapping clock generating block is employed here to guarantee that the sampling switch can integrating switch will on be on at the same time.

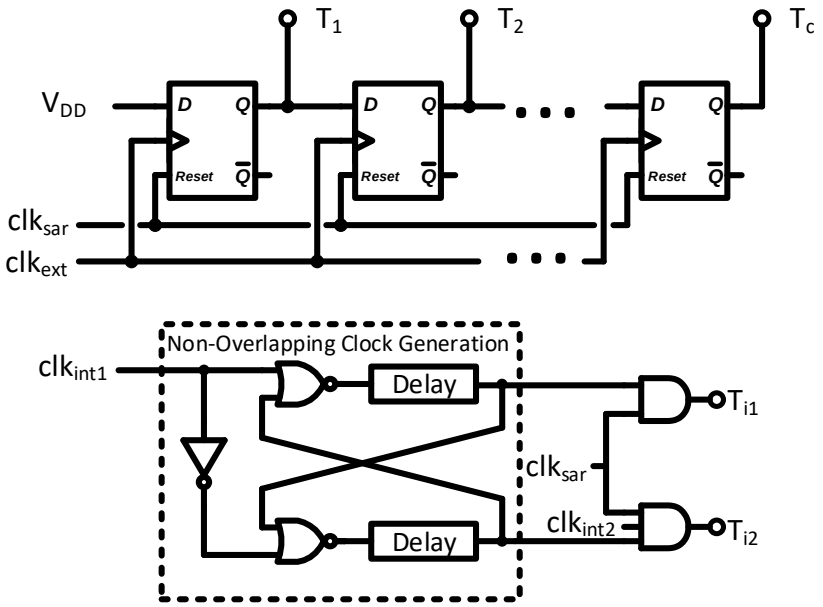


Figure 3.9: Schematic of clock generating circuit

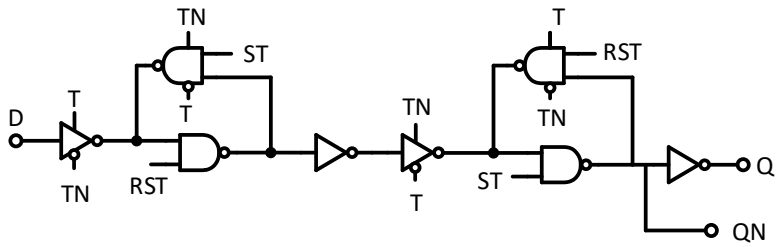


Figure 3.10: Schematic of TFET based DFF

The logical gates in Fig. 3.9 can be realized by replacing the CMOS transistors with TFETs in conventional logic gates. However, we redesign the D Flip-Flop (DFF) that using clocked inverters instead of transmission gate to avoid the effect of ambipolar and asymmetry of TFETs. The schematic of the TFET based DFF is shown in Fig. 3.10.

Fig. 3.11a and Fig. 3.11b shows the switch process and time diagram of the 2nd order noised shaped SAR ADC. In Fig. 3.11a, the width of sampling pulse is 8 clk_{ext} cycles, which is 320 nS. And the time for successive approximation process is also 8 clk_{ext} cycles. Finally, the operating time of the passive integrator is 2 clk_{ext} cycles, which makes the whole operating period of the

NSSAR 18 clk_{ext} cycles.

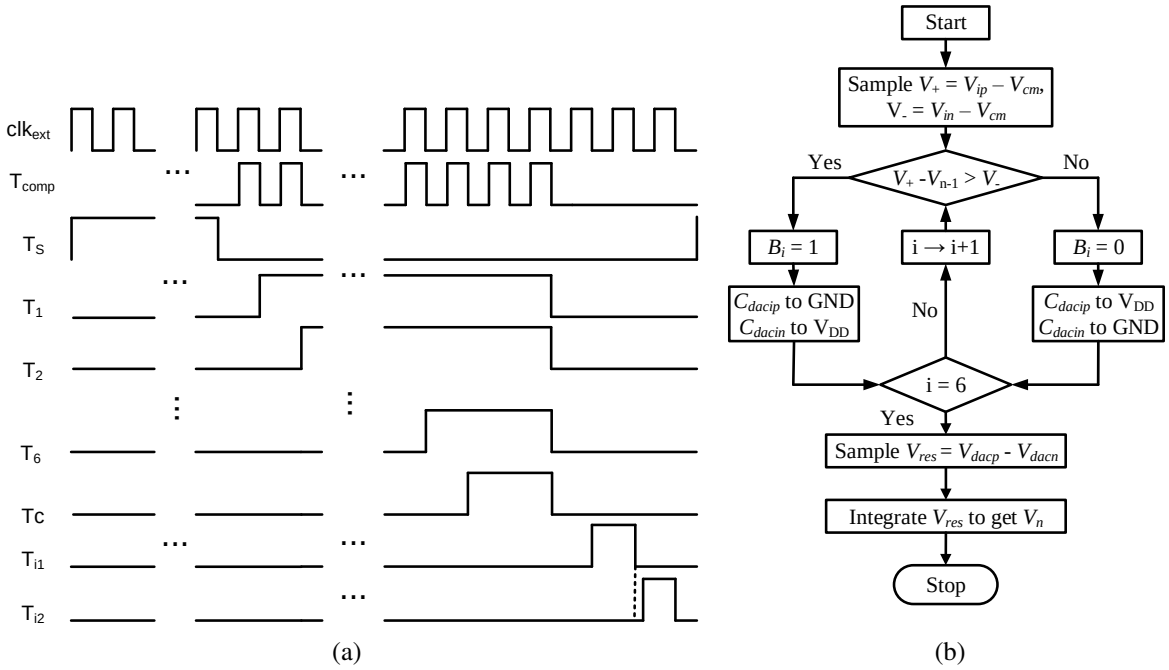


Figure 3.11: (a) Timing diagram and (b)switching process of the ADC

In Fig. 3.11b, V_{n-1} is the voltage stored in the integration capacitor from the last operation cycle and be expressed as:

$$V_{n-1} = g_1 V_{int1} + g_2 V_{int2} \tag{3.16}$$

From Fig. 3.11, when sampling clock T_S falls low, the external clock is applied to the comparator to perform success approximation. Then, the ADC will output a 6-bit code and the differential residue voltage of the DAC, V_{dac}, will be integrated onto the integration capacitors as V_{int1} and V_{int2} for the next operation, respectively. The integration operation is controlled by the non-overlapping clock signal T_{i1} and T_{i2}.

Fig. 3.12 shows the DAC control logic.

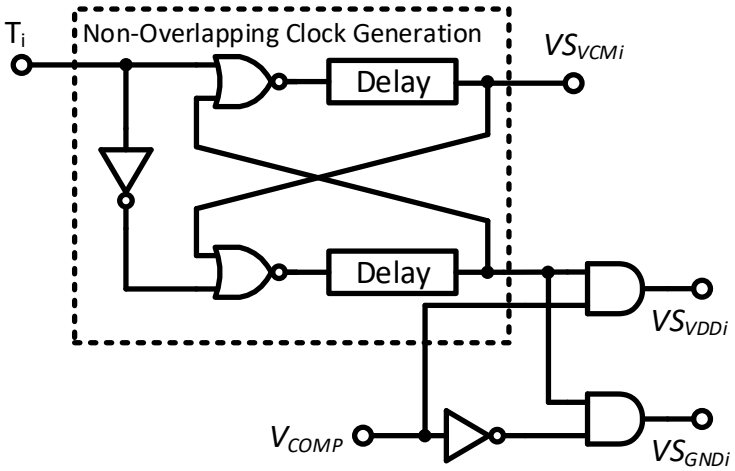
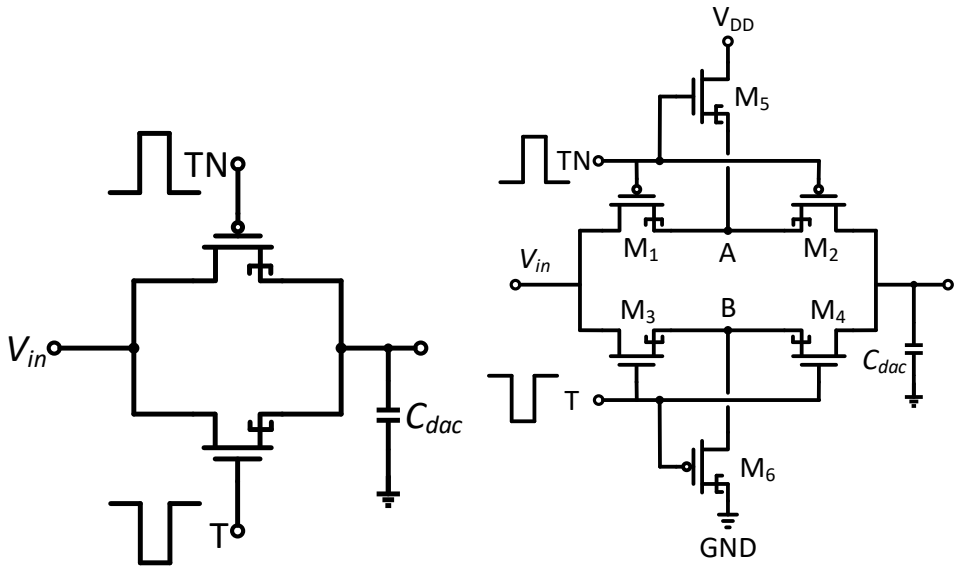


Figure 3.12: Schematic of DAC control logic

At the rising edge of T_i, a non-overlapping circuit pulls the control signal VS_{VCMi} to GND to cut the ith capacitor in the DAC from V_{CM}. In the same time, according to the output of the comparator V_{COMP}, the capacitor will be connected to VDD or GND under the control of signal VS_{VDDi} or VS_{GNDi}. Non-overlapping of signal VS_{VCMi} and VS_{VDDi}/VS_{GNDi} avoids multiple switches be on at the same time, which causes excessive current consumption.

3.2.2 TFETS Based Sampling Switches Design

In Fig. 3.5, input sample-and-hold (S/H) of the ADC samples the input analog signal to all DAC capacitors during the sampling phase and isolate the input node and comparator during successive approximation process, which forms the feed-forward path in Fig. 3.2. Thus, the input switch must have small on-resistance in the sampling mode and low leakage current in holding mode for all input values to ensure the linearity of the sampling circuit. The complementary switch (C-switch) with an PTFET and NTFET in parallel, as shown in Fig. 3.13a, is a promising candidate because it guarantees low on resistance with input ranging from rail-to-rail, and reduces the influence of asymmetric effect [70].

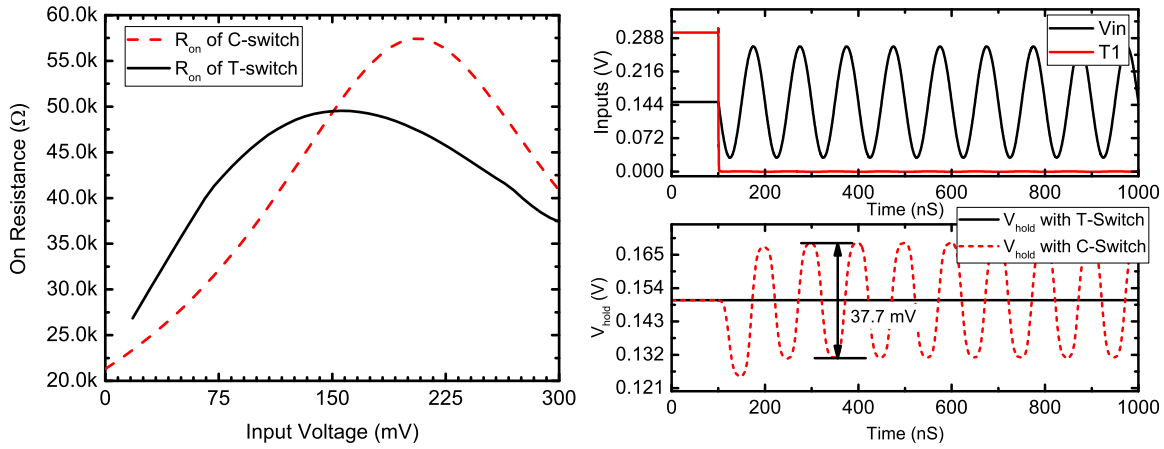


(a) (b)

Figure 3.13: Schematic of (a) C-switch and (b) T-switch

However, due to the effect of ambipolarity, leakage current will vary dramatically with the input signal in the holding mode (the switches are off). To address this problem, we employed a T-switch, as shown in Fig. 3.13b. As in Fig. 3.13b, when clock signal TN is high and T is low, the switch works in hold mode. TFETs M1-M4 are off and M5 and M6 are on. As a result, both PTFET's source are tied to VDD and both NTFET's source are tied to GND and the VGS of those transistors are "0". At the moment when M1-M4 is turning on, the voltage at net "A" and net "B" will guarantee that the VDS of NTFETs is greater than "0" and the VDS of PTFETs is less than "0", thus minimize the effect of asymmetry. A drawback of T-switch is its area occupation—to maintain low on resistance the width of M1-M4 is twice as their counterparts in C-switch.

Transistor level simulation is performed on both switch to verify their specifications. Transistor dimension of TFETs in C-switch and T-switch are 400 nm / 20 nm and 800 nm / 20 nm, respectively. Fig. 3.14a shows the on resistance of both transistors when VDD is 0.3V. Within the whole input swing, the on resistance of both switches range from 20 kΩ to 55 kΩ.



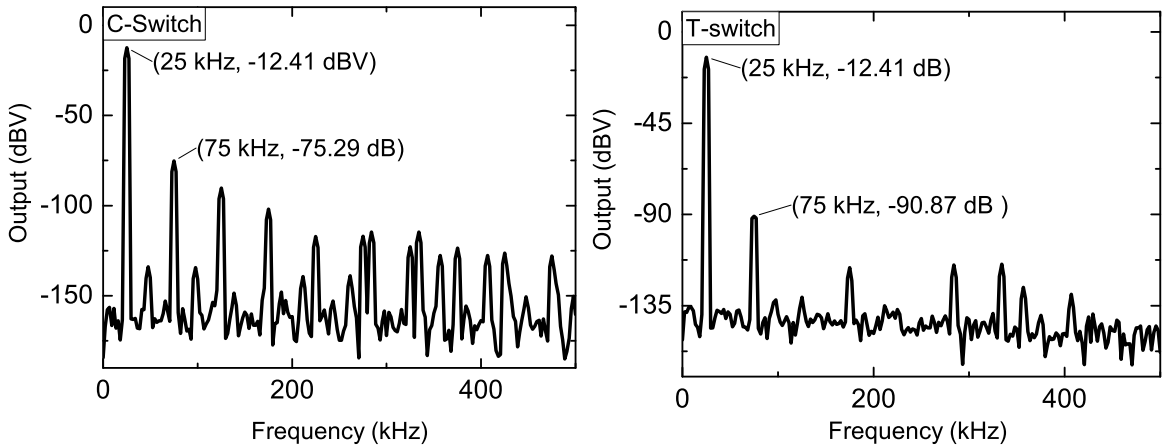
(a) (b)

Figure 3.14: On resistance of C-switch and T-switch (b) isolation of C-switch and T-switch

Consequently, they produce a time constant from 9.6 nS to 26.4 nS combined with the C_{dac}, which is sufficient small comparing to the sampling period (320 nS).

The reduction of ambipolar current because of zero VGS is shown in Fig 3.14b where clock signal goes off at 100nS and at the same time a 240 mV_{pp}, 25 kHz sinusoidal wave is applied to the input V_{in}. With the zero V_{GS} the ambipolar current is almost zero and results in a good isolation of the switch in holding mode. On the other hand, for C-switch, ambipolar current is proportional to the input voltage, and lead to a leakage of 37.7 mV_{pp} from input to the voltage on top plate of C_{dac}.

Moreover, the C-switch and T-switch are used to sample a differential sinusoid signal to Cdac with frequency of 25 kHz and amplitude of 2×240 mV_{pp} with the sampling frequency of 1.041 MHz. The spectrum of voltage of on top plate of C_{dac} is shown in Fig. 3.15a and 3.15b, for C-switch and T-switch, respectively.



(a) (b)

Figure 3.15: (a) output spectrum of C-switch and (b) output spectrum of T-switch

3.2.3 Summing Comparator Design Using TFET

Fig. 3.16 shows the schematic of the dynamic comparator. We design TFETs version of the comparator by substitute CMOS transistors with corresponding TFETs. Four input pairs are employed to implement the weighted summing in Fig. 3.2 and Fig. 3.5. As demonstrated in Eq. 3.11, the offset and input referred noise will be attenuated by the NTF. Hence, we use minimum length of the transistors, i.e. 20 nm, to save the area. The unit dimension of input transistors is 200 nm / 20 nm. Then, the input of the comparator is

$$V_{COMP} = V_{dac} + 25V_{int1} + 50V_{int2} + 0.1V_{dither} \tag{3.17}$$

The output of the comparator is stored by a SR latch which is not shown in the figure. When T_{COMP} is low, the output of the comparator is VDD and the SR latch will keep the previous value. At rising edge of T_{COMP}, the differential current produced by the 4 input pairs will trigger the regenerative feedback loop comprising M1-M4 and change the value stored in the SR latch.

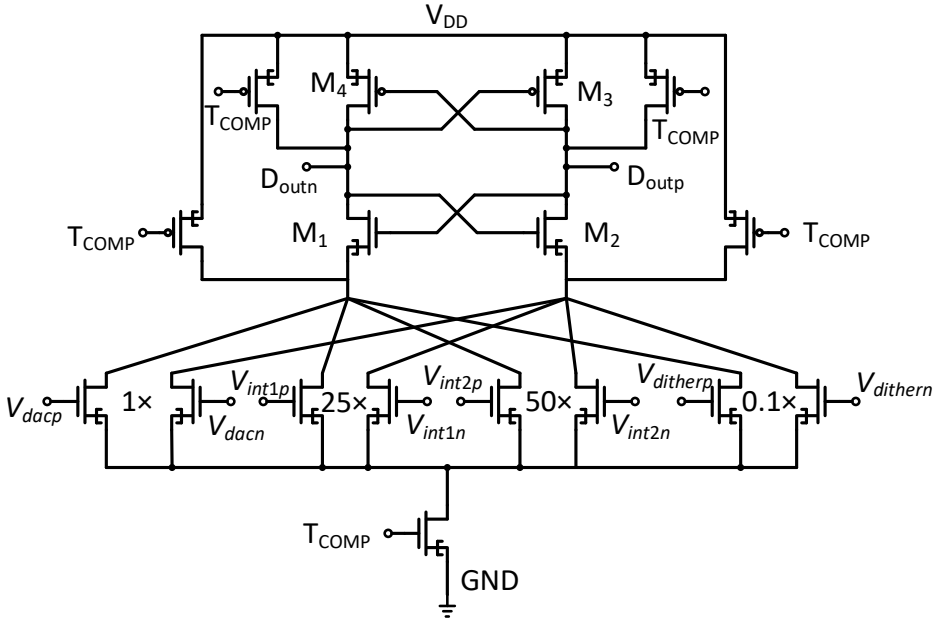


Figure 3.16: Schematic of comparator

3.2.4 Feedback DAC Design

The feedback DAC is implemented with a binary-scaled charge-redistribution topology. Digital bits (B_1–B_6) from DAC control logic drive the bottom of capacitors either to VDD or to GND to produce the output. According to Eq. 3.15, to meet the requirement for KT / C noise, we set the total capacitance of one side of the DAC as 480 fF. Consequently, the unit capacitance of the DAC is $C_{dac} / 2^6 = 7.5$ fF. The capacitor can be realized using metal–oxide–metal sandwich structure, as shown in [69].

3.3 Simulation Results

Transistor-level simulation and analysis of the TEFT based NSSAR ADC is performed using Cadence Spectre with transient noise simulation module. with modified VerilogA TFET transistor model. The minimum TFET transistor length is 20 nm.

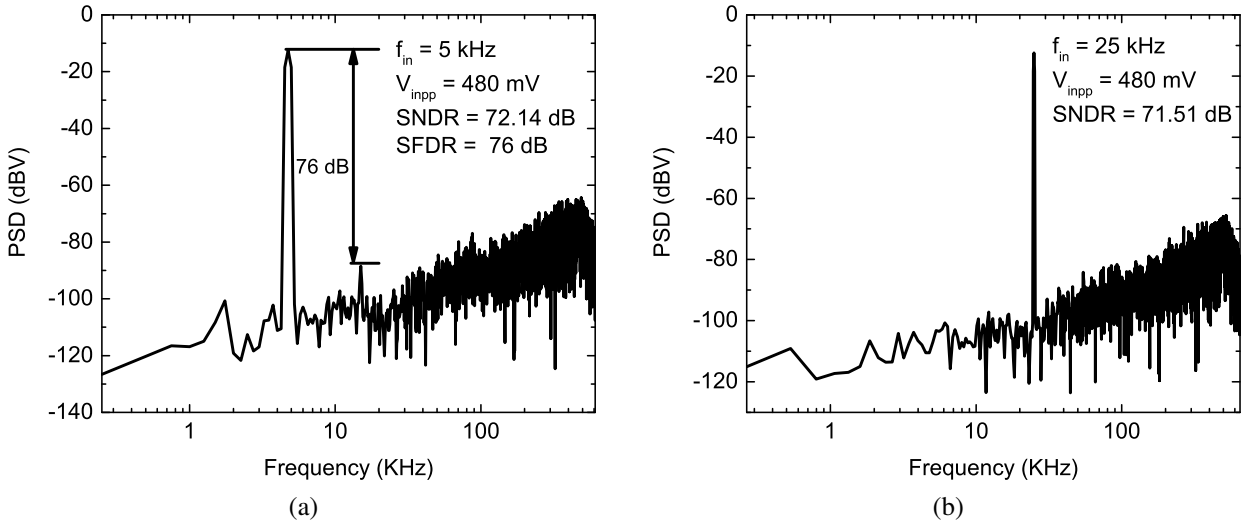


Figure 3.17: Output PSD of the NSSAR ADC when input frequency is (a) 5 kHz and (b) 25 kHz

To explore the TFET benefits in subthreshold region, the typical supply voltage is set to be 0.3 V. The typical temperature is 25 °C. Under the normal condition, a 25 MHz external clock is used for the ADC. As a result, according to Fig. 3.11a, the sampling frequency is 1.38 MHz.

Fig. 3.17 depicts the output PSD of the ADC when input is a 5 kHz and 25 kHz, 480 mV_{pp} differential sinusoidal signal. The simulated SNDR for the 5 kHz input signal is 72.14 dB and the SFDR is 76 dB. Consequently, the ENOB for the 5 kHz input signal is 11.69 bits. The harmonics of the 25 kHz input fall out of Nyquist frequency and submerges in the shaped noise. As a result, we did not get the data on it. The SNDR for the 25 kHz input is 71.51 dB and the ENOB is 11.58 bits.

We also disabled the second stage integrator and the dither circuit to verify the impact of number of orders of the NTF and dither circuit. Fig. 3.18 shows the output PSD of the first order NSSAR with dither circuit disabled. The input signal is again a 25 kHz, 480 mV_{pp} differential sinusoidal wave. Due to limited cycles, the noise is concentrated into several discrete peaks, which can be problematic in audio applications.

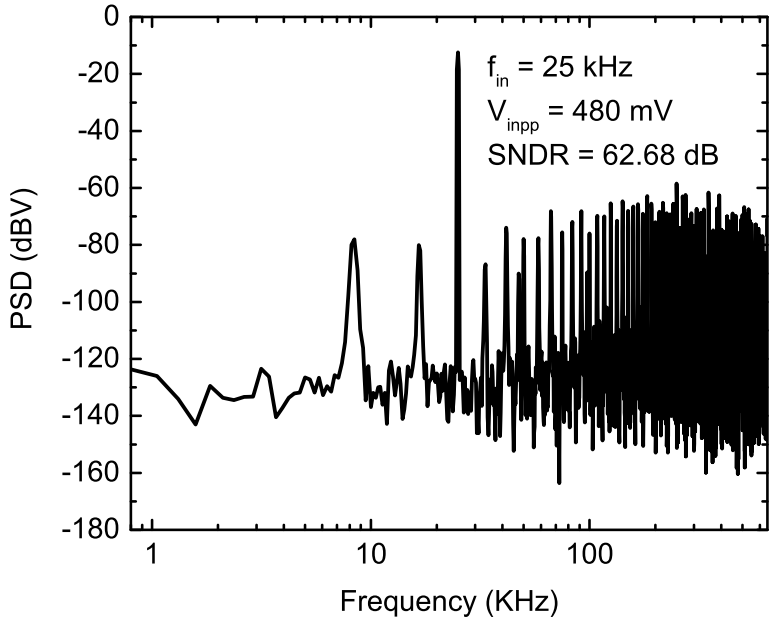


Figure 3.18: output PSD of 1st order modulator without dither circuit

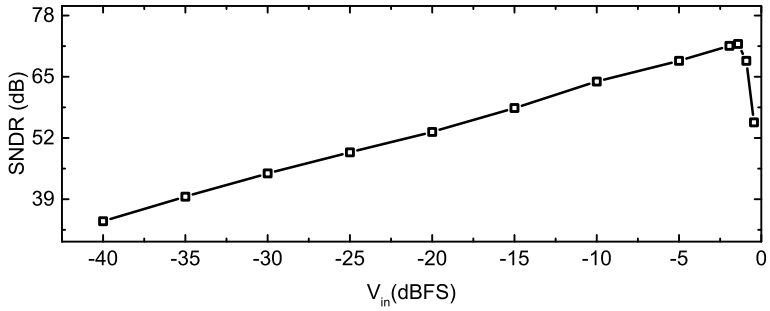


Figure 3.19: SNDR v.s. input amplitude

The SNDR is 62.68 dB, 9 dB lower than that in 2nd order NSSAR.

Fig. 3.19 shows the SNDR performance versus input amplitude at frequency of 25 kHz. The measured peak SNDR of the ADC is 71.98 dB when input is 0.51 V_{PP}. When input further increases, the SNDR will drop dramatically, due to the limitation of linearity of input S/H. Fig. 3.20 plots SNDR versus input frequency. As the input frequency increases, the SNDR reduces slightly. When the sampling frequency is 1.38 MHz, the total power of the ADC is 0.94 μW. The power consumption break-down is shown in Fig. 3.21.

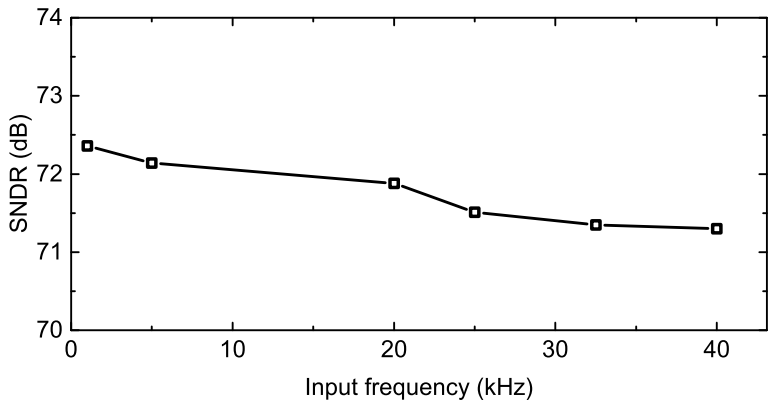


Figure 3.20: SNDR v.s. input frequency

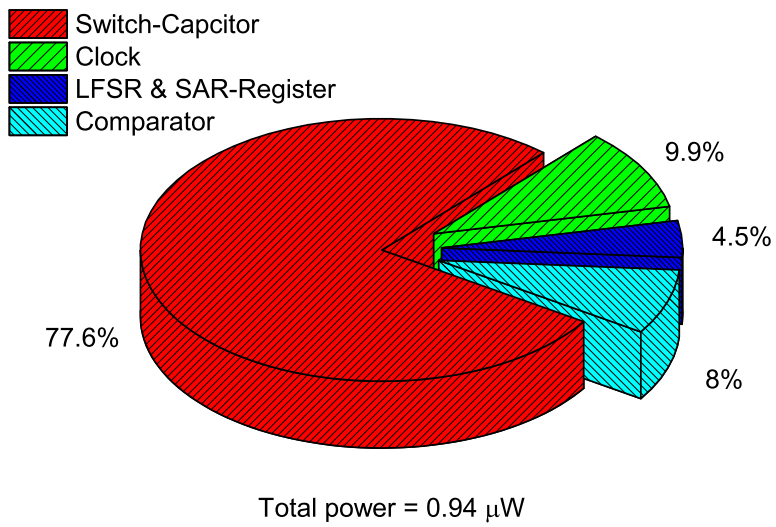


Figure 3.21: Power consumption break-down of the ADC

From Fig. 3.21, most of the power (77.6%) is dissipated on switched capacitor circuits indicating that the power consumption is limited by KT/C noise.

Performances of the ADC are summarized and compared to previous works and shown in Table 3.2. Schreier FOM, (see Eq. (22)), is also employed to give the comprehensive figure of merit of the ADCs .

$$FOM = SNDR + 10\log_{10}\left(\frac{BW}{power}\right) \tag{3.18}$$

Table 3.2: Performance Summary and Comparisons

References	[15]	[17]	[16]	[19]	[18]	This work
Technology	90 nm	90 nm	180 nm	90 nm	90 nm	20 nm TFETs
VDD (V)	0.3	0.3	0.3	0.35	0.3	0.3
Bandwidth (kHz)	125	45	2.5	150	300	43.4
Power (nW)	52	35	15.9	285	187	940
ENOB (bits)	8.63	8.38	8.77	8.91	9.46	11.67
SNDR (dB)	53.71	52.2	54.55	55.5	58.7	71.98
Schreier FOM (dB)	177.5	173.3	166.5	172	180.7	178.7

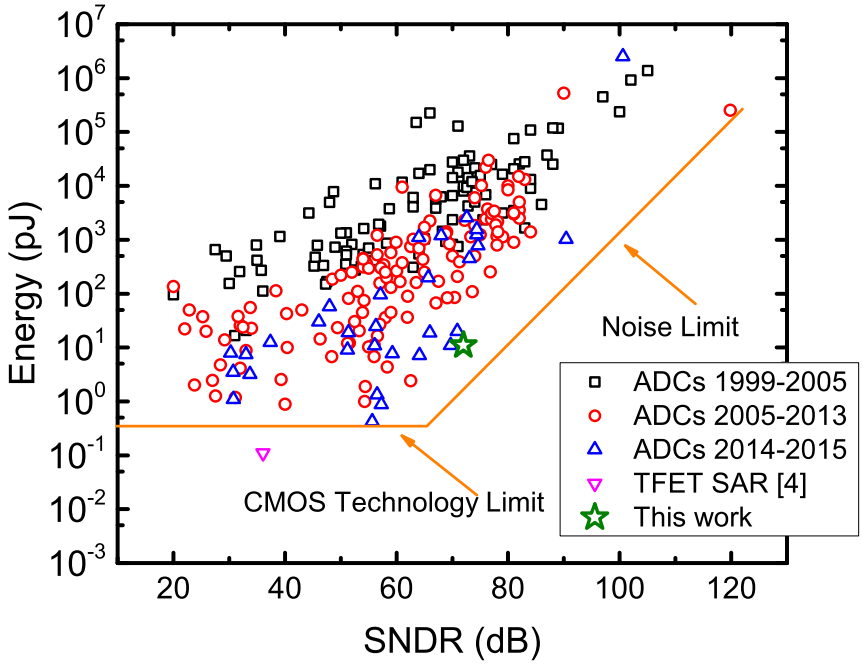


Figure 3.22: ADC energy comparison between CMOS and TFET

In Fig. 3.22, We compare the energy consumption of our design and ADCs reported in 2016 IEEE International Solid- State Circuits Conference (ISSCC) and VLSI Symposia (VLSI) during last two decades. From Fig. 3.22, our design is approaching the noise limit, which has a high consistency with high portion of switch-capacitor power consumption shown in Fig. 3.21.

3.4 Conclusion

This chapter for the first time presents a 12-bit SAR ADC with ultra-low 0.3V VDD using emerging TFETs technology that suitable for IoT applications. The proposed 2nd order NSSAR architecture is analyzed with respect of NTF, system stability and KT/C noise. Optimization method is introduced in to trade-off between contradict system specifications. Emerging TFET is employ in our design to replace CMOS counterparts for superior performance at sub-threshold region. With the optimized 2nd noise shaping and the higher SS of TFETs, the ENOB is effectively increase and the power consumption is mainly limited by noise performance. ADC achieves ENOB of 11.67 bits and Schreier FOM of 178.7 dB, one of the highest among recent works.

CHAPTER 4: ANALYSIS AND SIMULATION OF CAPACITOR-LESS RERAM-BASED STOCHASTIC NEURONS FOR IN-MEMORY SPIKING NEURAL NETWORK

[1]We propose the Neural Array—a modified one-transistor-one-ReRAM (1T1R) crossbar that integrates our ReRAM neurons with ReRAM synapses to form a compact and energy efficient in-memory neural network. To the authors' knowledge, this is the first effort to integrate the ReRAM-based neurons with ReRAM crossbar-based synapses. We utilize the ternary weight, which is limited to {-1, 0, 1}, to omit the need for weight generation circuit, such as ADC-DAC [71] with ultra-low power circuit implementation [49, 72, 73], and pulse-width-modulation (PWM) [22]. Thus we simplify the design and lower the power consumption. Moreover, we show that weight ternarization can reduce the effect of device mismatch on network accuracy.

Subsequently, we employ the neuron into a deep belief network (DBN) with noisy rectified linear unit (NReLU). We develop a simple algorithm to ternarize the weight of a trained DBN. We show that the accuracy loss of the ternarization is negligible in our algorithm. To reduce the impact of process variation on the classification accuracy, we model the mismatch of neurons as unknown "noise" of the network and set "noise" to zero during ternarization, and thus the weight ternarization provides robustness to the network against device mismatch. Moreover, we utilize the sparsity obtained from the ternarization to reduce the device usage, which will further reduce the area and power consumption. We train the DBN on MNIST dataset [74] and simulate the spiking DBN in MATLAB. We also show that the accuracy of ReRAM neuron-based DBN is robust against the ReRAM process variation effect.

[1]This chapter was published as Lin, Jie, and Jiann-Shiun Yuan. "Analysis and simulation of capacitor-less ReRAM-based stochastic neurons for the in-memory spiking neural network." IEEE transactions on biomedical circuits and systems 12, no. 5 (2018): 1004-1017.

This chapter is organized as follows. Section II describes the underlying device physics of ReRAM and how to mimic the integrate-and-fire behavior with the growth of conducting filament. The control circuit design of the ReRAM-based stochastic neuron is also included in this Section. In Section 4.2, the simulation results of the ReRAM-based neuron are presented. In Section 4.3, a DBN with NReLU on MNIST data set is trained. We ternarize and reorder the weight matrices so that they are optimized and can be mapped from the NReLU DBN to a DBN of ReRAM neurons. We also analyze the influence of the device mismatch on classification accuracy and show that the weight ternarization can reduce the effect of device mismatch. Finally, the conclusion is given in Section 4.4.

4.1 ReRAM-Based Stochastic Neuron

The basic device structure and the underlying physics involved for the ReRAM relies upon the formation and rupture of the nanoscale conductive oxygen vacancies [75]. In the set process, oxygen ions drift to the anode and create conductive oxygen vacancies through the oxide, thus reducing the resistance of the device from a High Resistance State (HRS) to a Low Resistance State (LRS). Inversely, in the the reset process, the oxygen ions move back to recombine with the vacancies, resulting in the transition from a LRS to a HRS [75]. This process is modeled by growth or rupture of one dimensional Conductive Filament (CF). The gap distance, $g(t)$, between the tip of the filament and the electrode, controls the ReRAM voltage-current curve through trap-assisted-tunneling [26].

An artificial spike-based neuron consists of inputs (dendrites), the computation element (soma) and the output (axon) as shown in Fig. 4.1. The key computational element is the soma that integrates the input spikes to the membrane potential and generates a firing event.

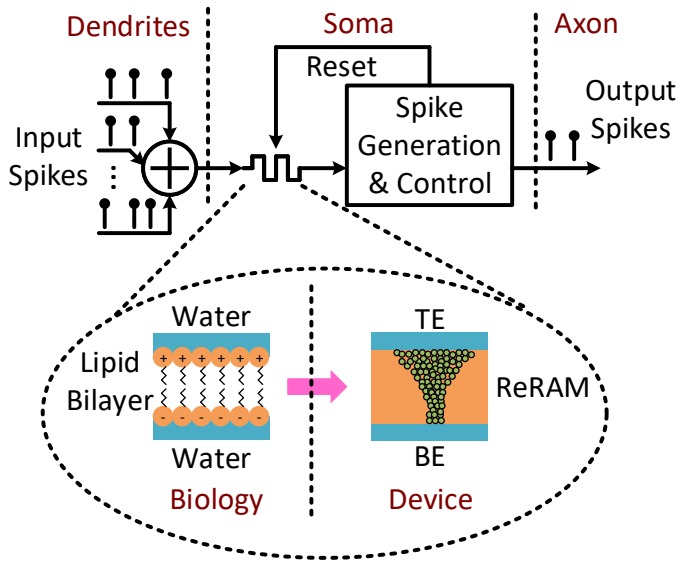


Figure 4.1: Stochastic neuron based on a ReRAM device that consists of inputs (dendrites), the computation part (soma) and the output (axon). The dendrites may be connected to multiple synapses interfacing with other neurons in a network. The key computational element is the neuron membrane which is emulated by a ReRAM device. It integrates the input spikes in terms of the growth of conduct filament. The thresholding and spiking generation are performed by a simple electrical circuit. Because of their inherent nanometer-scale dimensions and native stochasticity, these ReRAM devices are able to implement large and dense populations of neurons for neuromorphic computation.

In a generic Integrate-and-Fire neuron, the membrane potential u is determined by the differential equation [76].

$$\frac{du(t)}{dt} = \frac{1}{\tau}\left[F(u) + G(u)I\right] \tag{4.1}$$

In Eq. 4.1, τ is the time constant of the neuron, $F(u)$ is the "leaking" term and accounts for imperfections in the cell membrane that lead to leakage of the accumulated charge, and $G(u)$ is the input resistance term. The membrane potential dynamically evolves, due to the input current. Whenever the membrane potential reaches a certain threshold θ, the neuron fires and $u(t)$ is reset to its initial value. The neuron dynamic will resume its operation after a refractory period.

We leverage the similarity between the CF growth and the evolution of the membrane potential for implementation of the soma in Fig. 4.1 with a ReRAM device and the spike generation and reset

control circuit.

In the ReRAM-based neuron, we emulate the neuron features using the configuration of conductive filament in a ReRAM as:

$$u_r(t) = g_{max} - g(t) \tag{4.2}$$

where g_{max} is the maximum gap distance. Jiang et al. [75] have modeled the evolution of the the gap distance by applying Arrhenius law and the probability for oxygen ions to overcome the migration barriers, E_A. Thus, the evolution dynamic of the $u_r(t)$ with respect to the applied voltage V is:

$$\frac{du_r(t)}{dt} = v_0 \exp\left(-\frac{qE_A}{kT}\right) \sinh\left(\frac{\gamma a_0 q}{T_{OX} kT} V\right) \tag{4.3}$$

where T_{OX} is the oxide thickness, a_0 is the hopping site distance, and γ is the field local enhancement factor in terms of polarizability of the material [77]. From Eq. (4.3), the ReRAM-based neuron preforms the temporal integration of the input signal the same way that the generic LIF neuron model of Eq. (4.1) does.

In addition to the deterministic dynamics, the generation and migration of oxygen vacancies, which are determined by the kinetic energy of the ions, is inherently random [75]. As a result, the activation of the ReRAM based neuron is stochastic as well.

To verify the I-V characteristics of the ReRAM, we use a modified Verilog-A model [75] with the key parameters shown in Table 4.1. These parameters can be obtained by fitting the experimental data in [27] with the model predictions. Fig. 4.2 depicts the simulated current-voltage characteristics of the ReRAM driven by a DC voltage source. One can observe that the set/reset threshold for the device is around 1.1V/1.25V, respectively. Thus, to ensure the ReRAM device is fully reset, we set the magnitude of both setting and resetting voltages to 1.3 V. Moreover, it can be seen from the exp term in Eq. (4.3) that an increase in temperature caused by voltage pulse injected

into the ReRAM will further increase $u(t)$ and then result in more current generated. This positive feedback between $u(t)$ and temperature causes an abrupt setting of ReRAM devices [28]. The set threshold also shows stochasticity in Fig. 4.2.

Table 4.1: Simulation parameters for ReRAM

Parameters	Value[1]
Oxide thickness (T_{OX})	5 nm
Minimum gap distance (g_{min})	0.1 nm
Maximum gap distance (g_{max})	1.7 nm
Thermal resistance (R_{th})	2100 K/W
Velocity-dependent attempt-to-escape freq. (γ_0)	16
Activation energy for vacancy generation (E_A)	0.6 eV
Threshold temperature for significant random variations (T_{crit})	450 K
Average switching distance parameter (g_0)	0.27 nm
Average switching voltage parameter (V_0)	0.43 V
Average switching current parameter (I_0)	61.45 μA
Temperature fitting parameter (T_{smth})	500 K
Distance fitting parameter (δ_{go})	0.002 nm

[1] The the 6σ variation of the parameters is 10%.

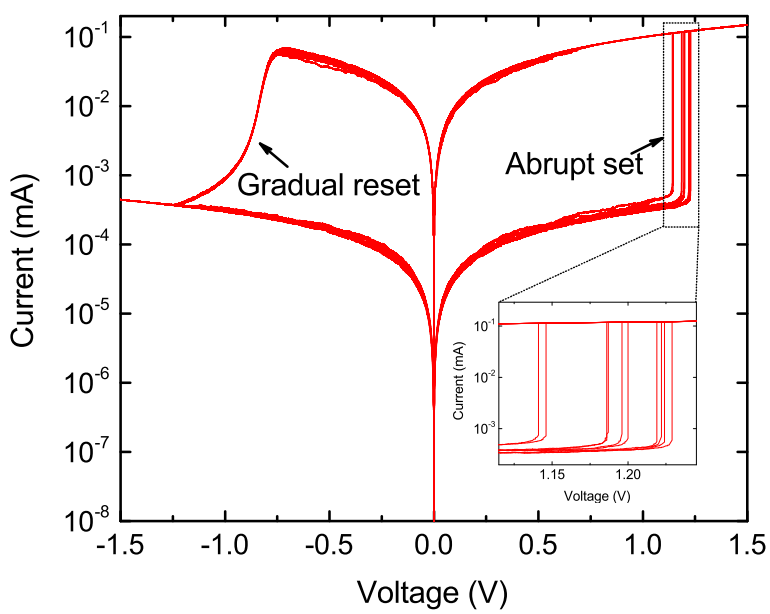


Figure 4.2: I-V characteristics of the ReRAM model which shows its abrupt setting and gradual resetting.

Thus, the ReRAM based neuron is different from the generic LIF neuron in three aspects:

1. the missing leakage term $F(u)$,

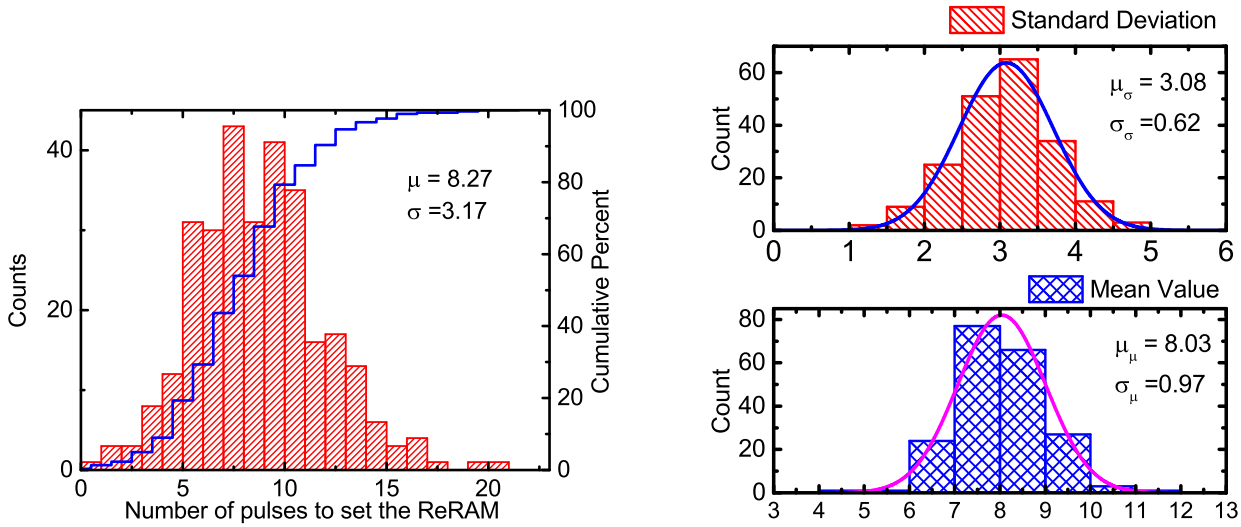
2. the nonlinearity of the hyperbolic $\sinh()$ function in Eq. (4.3), and

3. the random activation and device mismatch of the ReRAM device.

The first issue is addressed in recent research by mapping leak-less Integrate-and-Fire (IF) neuron to the Rectified Linear Unit (ReLU) used in the Convolutional Neural Network (CNN) in [37, 78–80]. Near lossless classification accuracy has been reported in the literature, which has shown the validation of using leak-less IF neurons in large-scale spiking neural networks. In our work, we utilize this mapping paradigm to explore the computing capacity of the ReRAM-based neuron, instead of approximating the accuracy of the biological process.

Due to the nonlinearity in Eq. (4.3), it is difficult to control the growth rate of the conductance filament with the amplitude of input voltage pulses. As a result, we keep the amplitude of the voltage constant through a voltage reference. In [22] the authors used the pulse-width-modulated input to emulate the weight of the synapses. However, considering the large amount of synapses used in the neural network, inclusion of the PWM generation circuit, such as the one reported in [81], will consume excessive area and power. So in our design, we keep the input pulse width in the same layer of the neural network constant, which limits our synapse weight $\boldsymbol{W}$ to $\{-1, 0, 1\}$. We prove that the information loss due to limited synapse weight precision will only cause minor classification accuracy loss as shown in Section 4.3. Also note, because pre-synaptic event is binary, the input to the neuron is equal to the difference between the number of excitatory input spikes ($\boldsymbol{W}$ = 1) and inhibitory input spikes ($\boldsymbol{W}$ = -1). Thus, the precision of membrane potential is limited to integer numbers. We utilize this limited precision to reduce the influence of the mismatch source of the neuron activation, such as process variation of the ReRAM device. Therefore, no delicate calibration is needed in our design. Only a few global parameters are determined infrequently. Thus the design complexity is greatly reduced. Detailed discussion and simulation results are given in

Section 4.3.



(a) (b)

Figure 4.3: (a) The probability of set switching is simulated for one set of the model parameters in Table 4.1 for 300 trials. In each trial, the height of the set pulse is 1.3 V and the width of the set pulse is 10 ns. We change the random seed in every trail. We also calculate the mean value μ and the standard deviation σ of the pulses needed to set the device. (b) 200 different sets of devices parameters with 6σ variation of 10% are measured in the way described in (a). The mean value μ (lower plot) and standard deviation σ (upper plot) for these 200 devices are presented with another histogram graph.

To obtain the statistics for both activation stochasticity and device-to-device variation, we simulated the pulse numbers required for triggering the set transition (with a fixed 10 ns pulse width) during 300 trials with different random seeds in one device and repeated such simulation for 200 different devices with 6σ parameter variation of 10%, which is actually the worst case scenario found for stable memristor devices noted in the literature [28, 82, 83]. Fig. 4.3 shows the simulated statistical distribution of the ReRAM set triggered by the input spiking train. In Fig. 4.3a, for one particular ReRAM, the number of input pulses need to set the device roughly follows a Gaussian distribution with a mean value μ of 8.27 and standard derivation σ of 3.17. We also show the cumulative distribution function (CDF) of the distribution, which we used to model the ReRAM device for behavioral level simulation. In Fig. 4.3b, the distribution of the μ and σ process variability

is shown. Due to the process variation, the average voltage pulse to set the ReRAM is centered around 8.03 with a standard deviation about 0.97. Also, most devices have a σ of 3.08 with a standard deviation of about 0.62.

We propose a simple behavioral level model of the ReRAM neuron that can be easily embedded into machine learning codes and can be applied to different types of memristors with stochastic accumulative-and-set process. Moreover, we include the process variation in the model to account for device mismatch. Here, we assume that the set process of the memristors roughly follows Gaussian distribution. Using the CDF shown in Fig. 4.3b, the output of the neuron is:

$$y = \begin{cases} 1 & \text{with probability } P = \Phi(n) \\ 0 & \text{with probability } 1 - P \end{cases} \tag{4.4}$$

$$n = \begin{cases} 0 & y = 1 \\ n' + (n_e - n_i) & y = 0 \end{cases} \tag{4.5}$$

where $n_e - n_i$ is the difference between the number of excitatory input spikes and inhibitory input spikes, n' is the n from the last simulation step, and $\Phi(n)$ is the CDF of Gaussian distribution with process variation:

$$\Phi(n) = \frac{1}{2}\left[1 + \text{erf}\left(\frac{n - \mu_\mu + N(0, \sigma_\mu)}{(\mu_\sigma + N(0, \sigma_\sigma))\sqrt{2}}\right)\right] \tag{4.6}$$

where $N(0, x)$ is a Gaussian distribution with zero mean and variance x. $N(0, x)$ will be sampled only once at the initialization of the program to model mismatch caused by process variation, as shown in Fig. 4.3b. Compared to the circuit level model proposed by R. Naous et al. in [84], our model is independent of technology and can be applied to other memristor devices with Gaussian

set process.

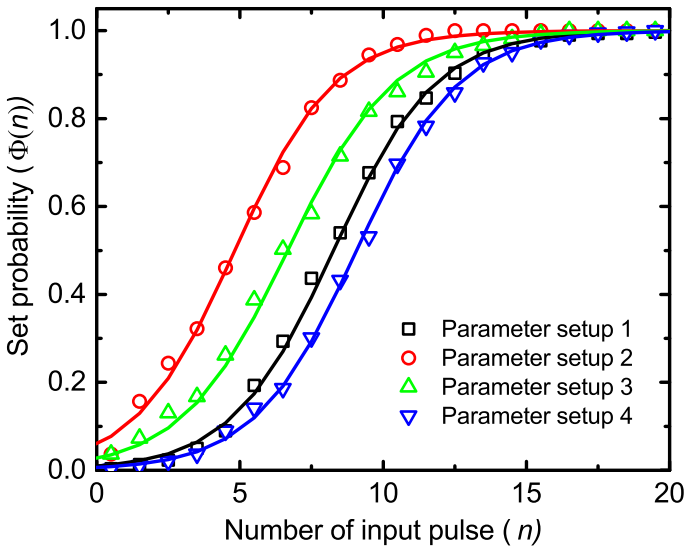


Figure 4.4: Model predictions (solid lines) of the simulated set CDF. We employ 4 groups of parameters in Table 4.1 and calculate the μ and σ for each parameter for the ReRAM to be substituted into Eq. 4.4. The behavioral level model can be applied to various of stochastic memristor devices with Gaussian set process.

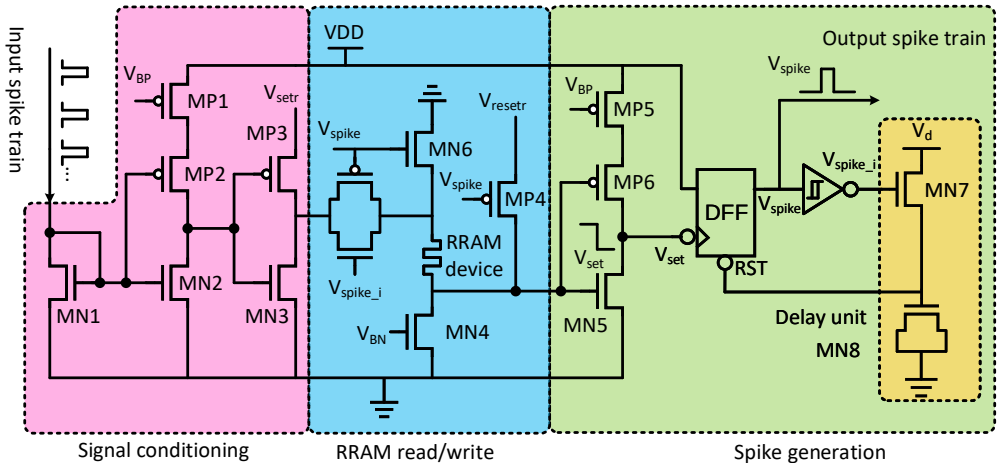
Moreover, it is compatible with the time-stepped behavioral level simulation [80] for large scale networks. The only necessary calibration is to change the data of μ_μ, μ_σ, σ_μ and σ_σ through device characterization.

To verify our model, we simulate the set process of the ReRAM with four groups of parameters in Table 4.1, and compare the simulated CDF $\Phi(n)$ (see Fig. 4.3a) and the model prediction from Eq. (4.6). One should note that the mismatch is now known, therefore we calculate the μ and σ for each scenario and apply them to Eq. (4.6) respectively. Fig. 4.4 shows that the prediction of the behavioral level model (solid line) fits perfectly to the device simulation. As a result, we use this behavioral level model in our MATLAB code in Section 4.3.

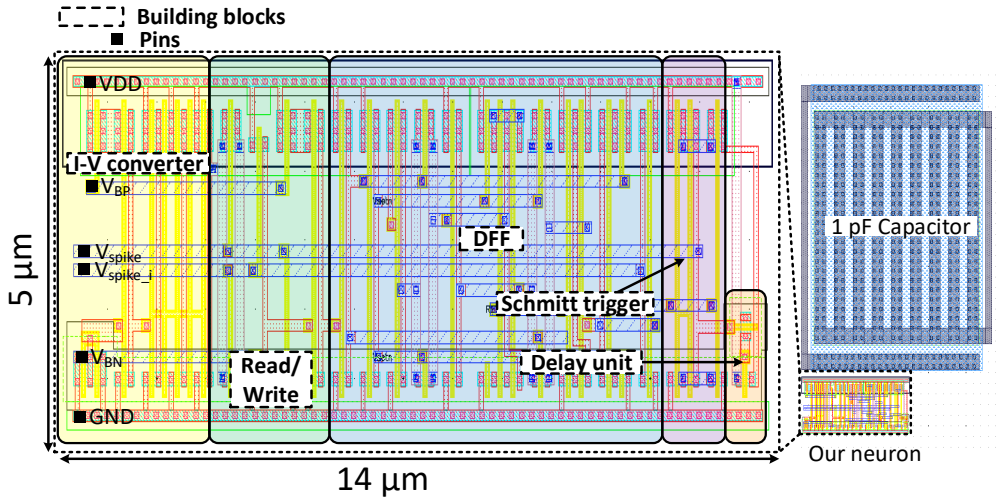
4.1.1 Spike Generation and Control Circuit Design

Since we use the ReRAM device to replace the membrane capacitor, our neuron does not use any capacitor and is thus more area efficient than those of capacitor-based designs [7–9]. Moreover, the abrupt set of the ReRAM (shown in Fig. 4.2) makes the firing easily detectable and eliminates the need for high precision comparators. We design a simple artificial soma that detects the membrane potential $u_r(t)$, generates the output spikes and resets the ReRAM device. As shown in Fig. 4.5a, the artificial soma is composed of three main function blocks: current-voltage converter (MN1-MN3 and MP1-MP3), ReRAM with an asynchronous read/write circuit (MN4, MN5, MP5 and MP6), and a pulse generation module formed by a D-type flip-flop (DFF), a Schmitt trigger and a delay unit.

As mentioned in the previous section, a voltage reference is needed to provide a fixed amplitude to the setting voltage pulses. Hence, we use transistors MN1-MN3 and MP1-MP3 to perform current to voltage conversion and signal conditioning. V_{setr} is the reference voltage to set the ReRAM and is equal to 1.3 V in our design. A transistor MN4 is connected in series with the ReRAM device, forming a voltage divider consisting of the ReRAM and the on resistance of MN4. Initially, the ReRAM is in high resistance state. As shown in Fig. 4.2, the resistance of the ReRAM is much larger than the on resistance of MN4. Thus, the setting voltage pulse is mainly applied on the ReRAM due to voltage division. After accumulating some input spikes, the ReRAM will switch to the low resistance state, as shown in Fig. 4.2. Then the next input spike will generate a falling edge to trigger the D-FF. Afterward, the ReRAM is disconnected from the input and will be reset through transistors MN6 and MP4 for the next period of integration. A delay unit generates a delayed reset signal (RST in Fig. 4.5a) to the DFF. Therefore, a voltage pulse is produced by the feedback loop, comprised of the DFF and the delay unit. The output pulse width is controlled by the delay unit, through which we can tune the spike pulse in the neural network layer effectively.



(a)



(b)

Figure 4.5: (a) Schematic of the ReRAM based stochastic neuron. (b) Left: Layout of the neuron. Right: area comparison between our neuron and an 1 pF capacitor.

Finally, to avoid the meta-stable state of the feedback loop, we utilize the hysteresis of the Schmitt trigger and insert it between the DFF and the delay unit.

To realize a capacitor-less design and a controllable delay, we propose the simple two-transistor (MN7 and MN8) implementation that utilizes the gate leakage current I_l of the deep sub-micron

transistor [85]:

$$I_l = WLA\frac{V_d^2}{T_{OX}^2}\exp\left\{-\frac{B\left[1-\left(1-\frac{V_d}{\phi_{OX}}\right)^{\frac{3}{2}}\right]T_{OX}}{V_d}\right\} \tag{4.7}$$

where A and B are technology parameters, ϕ_{OX} is the barrier height for electrons in the conduction band, V_d is the voltage as seen in Fig. 4.5a). Note that the delay $t_d = (V_d - V_{thres})/SR$, where V_{thres} is the threshold voltage to reset the DFF and SR is the slew rate:

$$SR = \frac{I_l}{C_{OX}} = \frac{I_l T_{OX}}{WL\epsilon_e\epsilon_r} \tag{4.8}$$

where ϵ_e and ϵ_r are the permittivity of a vacuum and the relative permittivity, respectively. Thus, by substituting Eq. (4.7) into Eq. (4.8), we observe that the delay t_d is exclusively determined by the voltage V_d and T_{OX}. Therefore, we choose MN8 as a minimum size transistor in our design.

The proposed circuit has been designed and simulated using 65 nm CMOS technology. The layout of the CMOS part of the neuron, including signal conditioning, ReRAM read/write, and spike generation and reset, is shown in Fig. 4.5b with the area of 14×5 μm^2. ReRAMs in a crossbar array can achieve the smallest theoretical size of $4F^2$ [29], where F is the feature size. Hence, the area of ReRAM device is negligible and not shown in Fig. 4.5b. To estimate the area comparison between our design and capacitor based designs [7–9], an 1pF Metal-Insulator-Metal (MIM) capacitor is placed next to our neuron. Using the top two metal layers from the same 65nm CMOS technology, the capacitor consumes 25×26 μm^2 in the layout, which is more than 9 times larger than our neuron size.

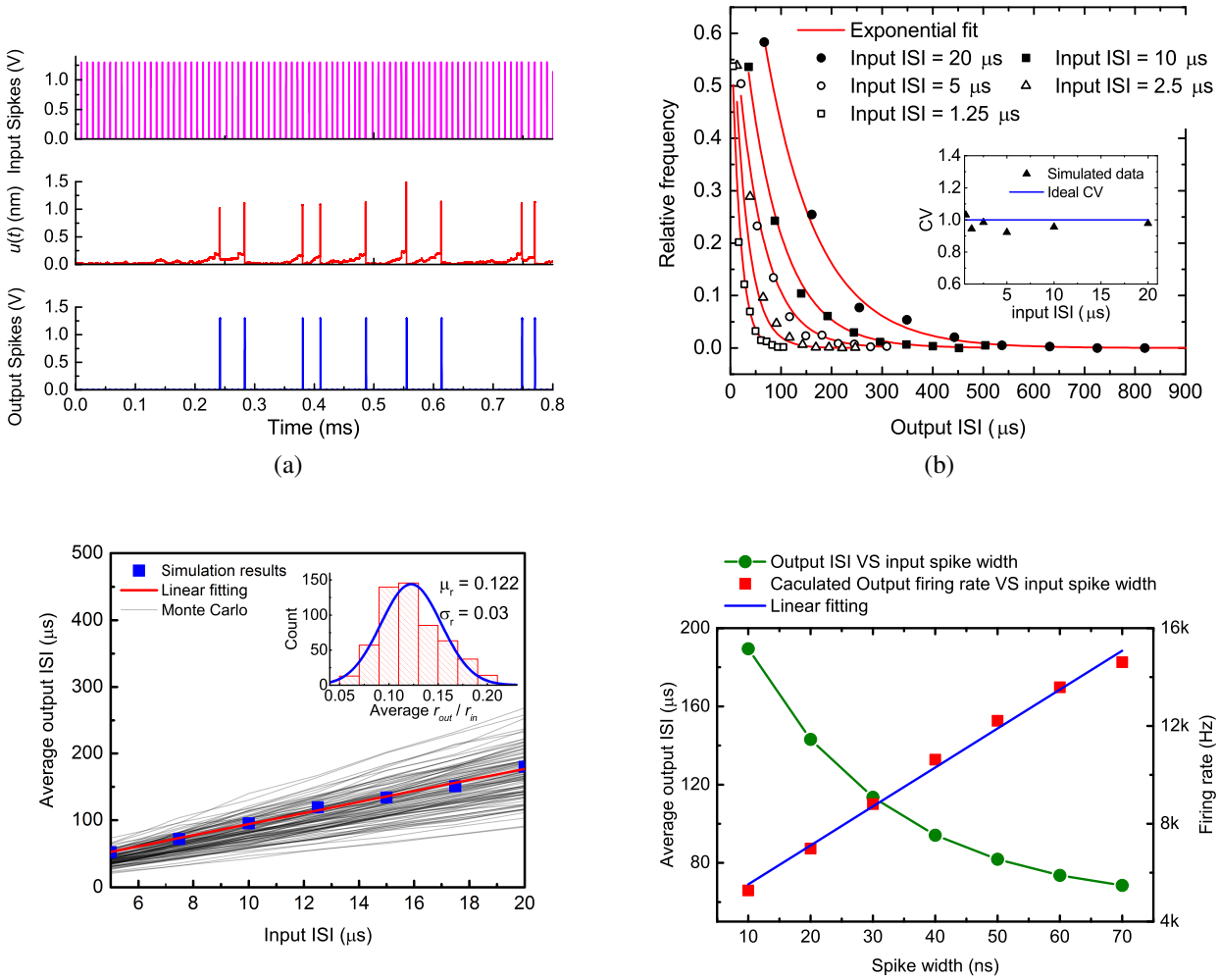


Figure 4.6: (a) Simulation results of the membrane potential $u(t)$ (the second row) and output spikes (the third row) versus input spikes (the first row). (b) Distribution of interspike intervals in a neuron. For different input interspike intervals (ISIs) from 1.25 μs to 20 μ, we record the distribution of output ISIs and using an exponential fit to approximate its probability density function. The inset is coefficient of variance (CV) with reference to an ideal Poisson (CV=1). The ISI distribution and CV show that the output spiking train follows the Poisson process. (c) The average output ISI versus input ISI and its statistics distribution. Inset: we run 100 Monte Carlo simulations for the distribution of the ratio between the average output firing rate (r_{our}) and the input firing rate (r_{in}). (d) Average output interspike interval of the neuron under different input spike widths. The firing rate is Firing_rate $= 1/$Average_ISI. Linear fitting is also performed to verify the linearity.

4.2 Results and Discussion

In our simulation, Berkeley Short-channel IGFET Model (BSIM4) [86] is used for the CMOS transistor modeling. To ensure the ReRAM could be fully reset, V_{setr} and V_{resetr} are set to 1.3 V. The supply voltage VDD for the circuit is 1.3 V. The detailed values of the supply and bias voltage sources are shown in Table 4.2.

Table 4.2: Bias and supply voltage sources

Name	Value
Supply voltage (VDD)	1.3 V
Setting voltage reference (V_{setr})	1.3 V
Resetting voltage reference (V_{resetr})	1.3 V
Bias voltage for PMOS (V_{BP})	1.0 V
Bias voltage for NMOS (V_{BN})	0.5 V
Voltage reference for delay unit (V_d)	1.3 V

To show the trajectory of membrane potential evolution and its relation to input spikes, a spike train with a rate of 100 spikes/ms is applied to the neuron. As seen in Fig. 4.6a, the membrane potential $u_r(t)$ evolves by integrating the input spikes. Stochastic process is also added to the evolution of the membrane potential and results in the stochasticity of the firing. We have shown the distribution of firing probability in Fig. 4.3. When the $u_r(t)$ approaches the threshold, the spike generation module is triggered to fire an output spike as shown in the third row of Fig. 4.6a. At the same time, the ReRAM is reset and ready to process another input spike train. No global clock is needed to read or write the ReRAM, leading to a fully asynchronous operation. Thus, our artificial soma is suitable for event-based systems.

To embed the soma circuit into the stochastic neural networks, it is important for us to investigate the output inter-spike interval (ISI) distribution. We feed the soma circuit in Fig. 4.5a with input spike trains with a fixed pulse width of 10 ns and ISIs of 1.25 μs, 2.5 μs, 5 μs, 10 μs and 20 μs. The

simulation duration is 50 ms, and all output spikes in 50 ms are recorded. As a result, we recorded 2826, 1583, 941, 607, and 390 spike events for each input spike train respectively. The distribution of the output ISI is shown in Fig. 4.6b. An important observation is that the output ISIs follow an approximately exponential distribution, coinciding with the distribution that has been reported by Sung Hyun et al. in [87]. In addition, the inset in Fig. 4.6b shows that the coefficient of variance (CV) of the output ISIs is close to 1. We also count the number of spike events in every 1 ms for each input spike train and run the Kolmogorov–Smirnov test (KS test) on them with the significance level of 0.05. Table 4.3 shows the output significance (p-value) and the expected value λ. As shown, all p-values are well above the threshold significance of 0.05, which implies that the output spike train of the neuron is a Poisson process. Since the Poisson spike train is the input for most spiking neural networks [9, 21, 26, 28], a ReRAM-based neuron design is perfect for the input layers of those neural networks. We also simulate the process variation of both the ReRAM and the CMOS transistors with 10% variation of the parameters in Table 4.1 and the Monte Carlo models of the BSIM4 for 100 samples. The 6σ variation of the expected value and the p-value is also shown in Table 4.3, which shows a large mismatch within the neurons. We will show in Section 4.3 that mismatch can be filtered out by ternarization of synapses weights.

Table 4.3: KS test for spike counts of each input spiking train.

Input ISI [1]	λ (6σ variation)	Significance (6σ variation)[2]
1.25 μs	56.52 (29.95)	0.19 (0.02)
2.5 μs	31.66 (16.78)	0.37 (0.05)
5 μs	18.82 (9.97)	0.48 (0.04)
10 μs	12.14 (6.43)	0.23 (0.01)
20 μs	7.8 (4.13)	0.13 (0.03)

[1] The simulation during is 50 ms and the sampling interval is 1 ms.
[2] The threshold significance is 0.05.

We use the widely accepted rate coding scheme for neuromorphic computing [7] to examine the

output and input relationship for the proposed circuit. With the rate coding, information is encoded by spike rate in the network and the spike width and height do not carry any information. Fig. 4.6c shows the average output ISI with respect to the input ISI of the ReRAM-based neuron. The pulse width of the input spike is 10 ns and the input ISIs spread from 5 μs to 20 μs. From Fig. 4.6c, the ReRAM-based neuron exhibits a linear transfer function with add-on noise, which is similar to the characteristics of the Noisy Rectified Linear Units used in the deep belief network. We unitize this feature to map the DBN to IF network and achieve near-lossless accuracy. The mapping process is described in detail in Section 4.3. Note that the fitting curve does not pass the origin. Instead, it has an offset because the ReRAM has a chance to set by a single input spike and there will be no ISI for this scenario. We ran 100 Monte Carlo simulations and depict all the output ISIs in Fig. 4.6c to show the impact of process variation. We calculate the output firing rate using $r = 1/\text{ISI}$ for every input ISI and divide the r_{out} rate by corresponding r_{in}. By doing so, we obtain 600 data samples from the Monte Carlo simulation. Because of the technology drift, the average value (μ_r) and standard deviation (σ_r) of ratio of r_{out} and r_{in} are 0.122 and 0.03, respectively.

From Eq. (4.7), the CMOS process variation effect can also change the output spike-width of the ReRAM neuron, which is the input spike width of the succeeding neuron. Note that the neuron in the same layer can share a control and spike generation circuit (see Section 5), thus the change of the input spike width is global to all neurons in the same layer. Fig. 4.6d depicts how the input spike width will affect the output spike frequency. We fix the input ISI to 10 μs (i.e. fix the input frequency to 100 kHz), and record the average output ISI with respect to different input spike widths. Then we can calculate the output firing rate with $Firing\_rate = 1/Average\_ISI$. We also perform a linear fitting for the input spike width and the output firing rate and show the fitting curve in Fig. 4.6d. The output firing rate exhibits good linearity with the input spike width, which proves that the firing rate variation due to input spike width can be easily compensated by tuning the V_d of the firing neuron circuit.

Finally, we report the power dissipation. For over 1,000 firing events, the average energy for the ReRAM device to fire a 10 ns spike, including the energy to set and reset the ReRAM, is 2.14 pJ, and the average power for the whole circuit is 1.289 μW.

4.3 Application and Optimization of ReRAM Neurons in Deep Belief Network

We verify that ReRAM-based neurons are suitable for stochastic spike-based in-memory machine learning through building a deep belief network with five layers of Restricted Bolzmann Machine (RBM), and training it on the MNIST handwriting database [74]. The training set consists of 60,000 individual handwritten digits, each labeled 0-9 for the individual 28×28 pixel gray scale images. The testing set includes 10,000 individual handwritten digits.

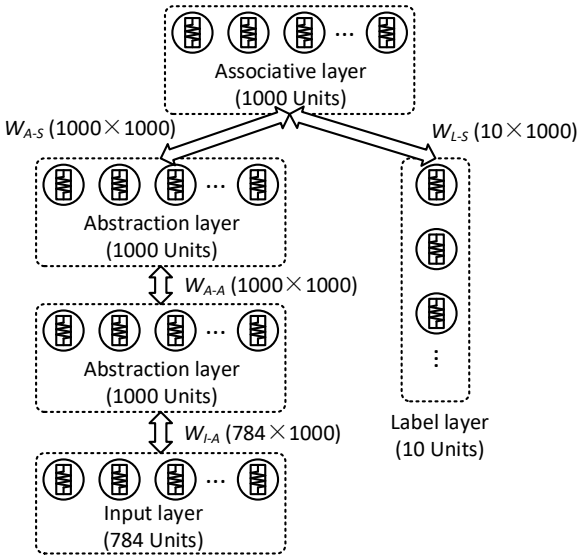


Figure 4.7: Architecture of the DBN for handwritten digit recognition. The connections between layers represent the weights of a RBM.

The DBN is comprised of an input layer of 784 visual units (corresponding to the pixels of 28×28 input images), two 1000-unit "Feature Abstraction Layers" that abstract the feature of the input data, a 10-unit "Label Layer" with units corresponding to the 10 digit-classes and a 1000-unit

"Association Layer" that associates the extracted feature with the "Label Layer", as shown in Fig. 4.7.

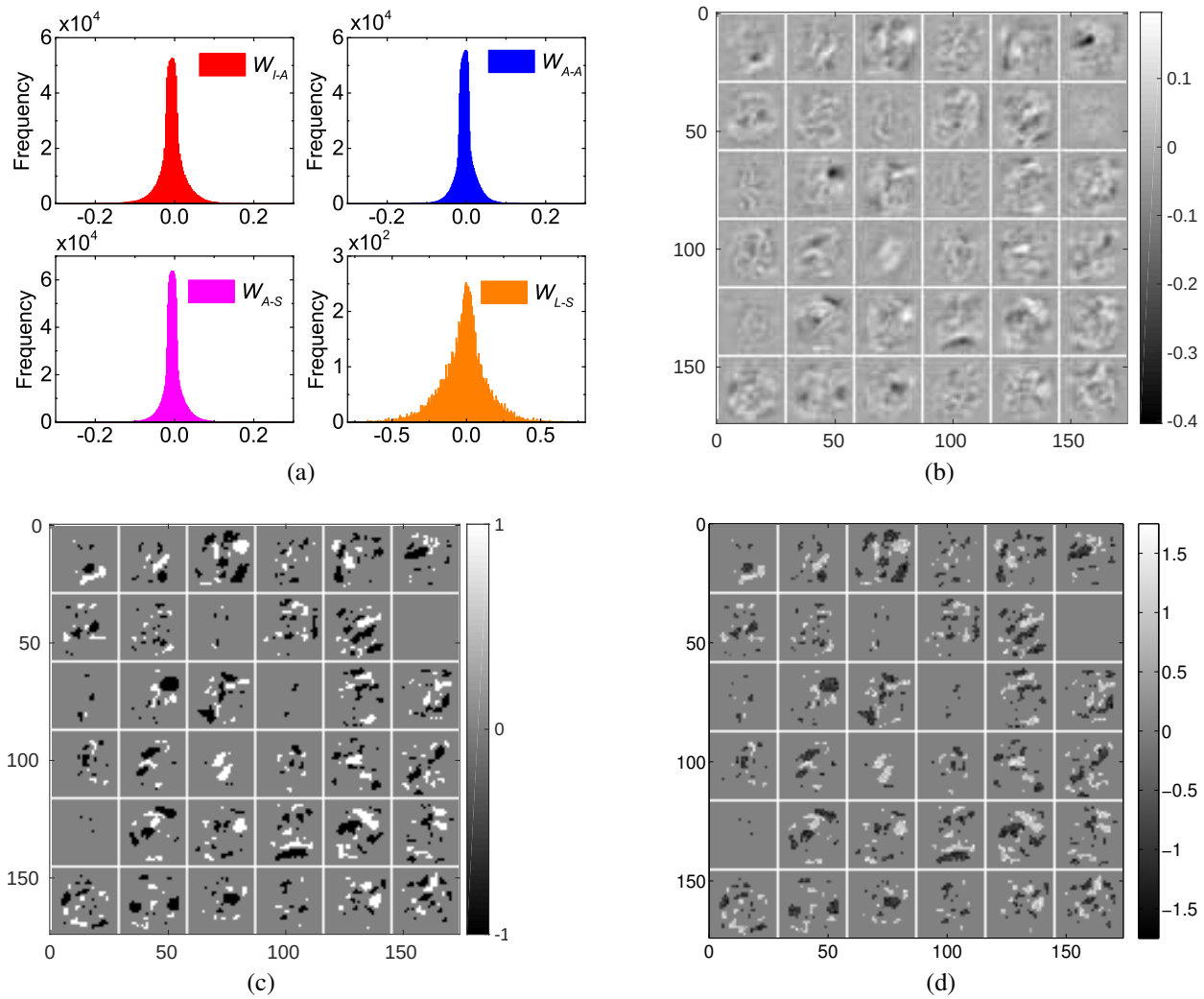


Figure 4.8: (a) Distribution of trained weight. A large portion of the weight is close to zero, meaning that the information carried is not very important and can be omitted. (b) Visualization of the $\boldsymbol{W_{I\text{-}A}}$ learned by a subset of neurons in the abstraction layer for 28×28 images in the MNIST data set. Each image shows the vector of weights feeding into one neuron in the first abstract layer. (c) Ternarized weight of Fig. 4.8b. Dark area correspond to $W_{I\text{-}A,ij} = -1$, light area means $W_{I\text{-}A,ij} = 1$ and grey area means $W_{I\text{-}A,ij} = 0$. (d) The influence of the mismatch of neurons on the ternarized feature. The features extracted are maintained with variation on the intensity. More importantly, the noise remains zero after mismatch is added.

We label the synapses weight between the input layer and the first feature abstraction layer as "$\boldsymbol{W_{I\text{-}A}}$", the weight between the first and second feature abstraction layers as "$\boldsymbol{W_{A\text{-}A}}$", the weight between the second feature abstraction layer and association layer as "$\boldsymbol{W_{A\text{-}S}}$", and the weight between the the association layer and label layer as "$\boldsymbol{W_{L\text{-}S}}$".

4.3.1 Emulate the Noisy Rectified Linear Unit with the ReRAM Neuron

The noiseless IF neuron has been utilized to replace the rectified linear unit (ReLU) in the convolutional neural network in [37, 78–80] obtaining near-lossless accuracy. In [80], Rueckauer et al. have reported a one-to-one correspondence between an ReLU unit and a SNN neuron. This proves that the spike rate of the neuron is proportional to the ReLU activation. Note that due to the absence of leakage and refractory period, the time step value of the simulation is not important as long as it is much larger than the pulse width of the spikes.

Similarly, we leverage the ReRAM neuron for the DBN architecture shown in Fig. 4.7 by examining the relationship of the DBN using NReLU [23] and the spiking DBN. First, the output firing rate ($1/$ISI) of the neuron is linearly proportional to the input firing rate, as shown in Fig. 4.6c. Note that now the weights are ternarized and the input of the neuron is the difference between the number of excitatory input spikes and inhibitory input spikes. Moreover, if the input is negative (i.e. there are more inhibitory input spikes), the neuron will not fire at all. Thus, the NReLU activation: $a = \max(0, x + N(0, \sigma(x)))$ resembles a firing rate approximation of an IF stochastic neuron with no refractory period. Second, for classification tasks, only the neuron with maximum firing rates in the output label layer is recognized as the inferred label. Thus the absolute firing rate of each neuron is not of importance and the overall rate can be scaled by a constant factor. Finally, it is difficult to provide the bias to each spiking neuron layer; therefore, the relative scale of the synapse weights to each other and the firing probability of the neuron are the only parameters that

matter. As a result, we use the following procedure to convert trained DBNs with NReLU into IF based DBNs with device mismatch:

1. Use NReLUs for all units of the network.
2. Fix the bias to zero throughout training.
3. Perform weight ternarization and reorder (see Section 4.3.2) the trained synapses to obtain weights in the IF based DBN.
4. Use the behavioral level model of the ReRAM neuron obtained in Eq. (4.4) and (4.6) to replace all NReLU units in the network.
5. Use the data from Fig. 4.3b to determine the mean value μ_μ and μ_σ.
6. Use the data from Fig. 4.3b to perform one-time Gaussian sampling to determine the mismatch $(N(0, \sigma_\mu)$ and $N(0, \sigma_\sigma))$ of the ReRAM devices.

Note that from Fig. 4.6d, the CMOS technology drift will cause the transfer function of the neuron to change. However, the firing rate can be scaled by changing V_d. Furthermore, because the control circuit can be shared among the neurons, the CMOS process variation is global to the network. Therefore the accuracy will only have minor degradation due to CMOS technology drift (see Section 4.3.3).

4.3.2 Weight Ternarization

Human and animal studies show that mammalian brains undergo massive synaptic pruning during childhood that removes synapses under a certain threshold during puberty [88]. Inspired by this phenomenon, we propose ternarization of the weights in our neural network. Ternarization of the

weights is crucial to implement our design in terms of 1) overcoming the nonlinearity between the filament growth rate and the voltage magnitude, 2) reducing the influence of overfitting and ReRAM mismatch, and 3) reducing the complexity of the design so that we can reorder the weight matrix to utilize the sparsity of weight matrices.

After training, Fig. 4.8a is the distribution of the trained weights in double precision. One can observe that a large portion of the weights are close to zero, meaning that the corresponding synaptic connections are very weak and can be omitted. Therefore, we ternarize each weight to be constrained to $\{-1, 0, 1\}$ with two thresholds t_i^{lo} and t_i^{hi}, where i is the layer index. The original double precision weights $\boldsymbol{W_{I\text{-}A}}$, $\boldsymbol{W_{A\text{-}A}}$, $\boldsymbol{W_{A\text{-}S}}$, and $\boldsymbol{W_{L\text{-}S}}$ can be ternarized as follows:

$$w_{ij,ter} = \begin{cases} -1, & \text{if } w_{ij} < t_i^{lo}. \\ 0, & \text{if } t_i^{lo} \leq w_{ij} \leq t_i^{hi}. \\ 1, & \text{if } w_{ij} > t_i^{hi}. \end{cases} \tag{4.9}$$

where $w_{ij,ter}$ is the ternarized entry of the weight matrix and w_{ij} is the entry in double precision. From Eq. (4.9) the choice of the threshold t_i^{lo} and t_i^{hi} is a trade-off between loss of information and sparsity of the weight matrix. Moreover, Fig. 4.8a shows that the distribution of the trained full-precision weights is roughly symmetrical with respect to zero. As a result, we have $t_i^{hi} = -t_i^{lo} = A_{ti}\sigma_{W_i}$ where σ_{W_i} is the standard deviation of the weight matrix $\boldsymbol{W_i}$ and A_{ti} is the tuning parameter.

More importantly, the ternarization can be considered as a regularization with the threshold of A_t against overfitting—weights learned from training set contain irrelevant information or "noise". Fig. 4.8b and Fig. 4.8c illustrate the comparison of part of the weight matrix $\boldsymbol{W_{I\text{-}A}}$ before and after ternarization where $A_t = 1.6$. One can see that the first abstract layer learned some information through training as shown in Fig. 4.8b. By ternarization in Fig. 4.8c, the clear part (relevant

information) in Fig. 4.8b is kept and normalized to $\{-1, 1\}$ and the blur part ("noise") in Fig. 4.8b is omitted and set to zero.

Also, if we consider the mismatch of the neurons as a unknown "noise" of the network model that will only be sampled once during initialization, the ternarization can also reduce the impact of the mismatch. Consider a membrane potential of a neuron with the index i in the lth layer in a simulation time step:

$$u_{ri}^{l} = \sum_{i,j} W_{ij} \left[\mu(r_{j,out}^{l-1}) + N(0, \sigma(r_{j,out}^{l-1}))\right] \tag{4.10}$$

where $\mu(r_{j,out}^{l-1})$ is the average firing rate of the neuron j in the $(l-1)^{th}$ layer and $N(0, \sigma(r_{j,out}^{l-1})$ is Gaussian noise with zero mean and variance of $r_{j,out}^{l-1}$ that models the process variation (see Fig. 4.6c). Note that the $N(0, \sigma(r_{j,out}^{l-1})$ will be sampled only once at the network initialization. Eq. 4.10 can be reorganized as:

$$u_{ri}^{l} = \sum_{i,j} W_{ij}[1 + N(0, \text{CV}(r_{j,out}^{l-1}))]\mu(r_{j,out}^{l-1}) \tag{4.11}$$

where CV() is the Coefficient of Variation and can be expressed as:

$$\text{CV}(r_{j,out}^{l-1}) = \frac{\sigma(r_{j,out}^{l-1})}{\mu(r_{j,out}^{l-1})} = \frac{\sigma(r_{j,out}^{l-1})/r_{j,in}^{l-1}}{\mu(r_{j,out}^{l-1})/r_{j,in}^{l-1}} = \frac{\sigma_r}{\mu_r} \tag{4.12}$$

where μ_r and σ_r are defined in Fig. 4.6c and $r_{j,in}^{l-1}$ is the input spike rate of the neuron. Using the results shown in Fig. 4.6c, we set $\text{CV}(r_j^{l-1}) = 0.24$.

From Eq. (4.11), the mismatch of the neurons can be integrated into the corresponding columns of the weight matrix. Therefore, due to the distinct weight value gained from the ternarization, the pattern of the feature learned from the training dataset (see Fig. 4.8c) will not be affected by the

mismatch. Only the intensity of features is changed. Moreover, the "noise" is set to zero during ternarization already, thus they will remain zero when mismatch is added. This keeps overfitting error regularized. Fig. 4.8d is an example of the feature intensity variation due to neuron mismatch. From Fig. 4.8d, the feature pattern is unchanged and no noise due to the mismatch is added, which indicates minor accuracy loss caused by neuron mismatch. We verify this conclusion by simulating the behavioral level model (Eq. (4.4)) in Section 4.3.3.

Also note from Fig. 4.8c that the ternarized weight matrix is sparse. We utilize the sparsity to reduce area and power consumption of the hardware with the extended Cuthill-McKee algorithm proposed by Cui and Qiu [89]. This algorithm leverages the linear transformation to effectively break down any matrices into all-zero sub-blocks. The all-zero blocks do not require hardware resource, and thus we can save power and area consumption through them. We map the reordered matrix to the Process Element (PE) in Chapter 5 with array size of 32×32 and 64×64, respectively.

After the reordering, the only change in the Neural Array to accommodate the matrix reorder is to change the input and output connections of the 1T1R crossbar [89]. Table 4.4 summarizes sparsity of the original weight matrices and the reduction of device usage in terms of the number of 1T1R ReRAM cells used. The size of the crossbar is set to either 32×32 or 64×64. In total metrics reordering reduces the number of ReRAM cells used by 36% and 33%, respectively.

Table 4.4: Percentage of Reduction in Device Usage (Number of ReRAM Cells Used)

	Size	Sparsity	1T1R Used (% of Reduction)	
			32×32	64×64
$\boldsymbol{W_{I\text{-}A}}$	784 k	89%	515 k (35%)	538 k (32%)
$\boldsymbol{W_{A\text{-}A}}$	1,000 k	89%	656 k (35%)	677 k (33%)
$\boldsymbol{W_{A\text{-}S}}$	1,000 k	89%	622 k (38%)	658 k (35%)
$\boldsymbol{W_{L\text{-}S}}$	10 k	87%	5 k (45%)	5 k (45%)
Total	2,794 k	89%	1,800 k (36%)	1,879 k (33%)

4.3.3 Classification Accuracy

We test the spiking DBN with both the ternarized weight matrices, and the method discussed in Section 4.3.1 for mapping DBN with NReLU to IF DBN in MATLAB. We train the single RBM with the contrastive divergence (CD) algorithm [90]. And we train the DBN with a greedy layer-wise method proposed by Bengio et al. in [91], which trains one RBM at a time and continues until the last RBM is trained. The 60,000 MNIST training set [74] is used for training. We partition the original 60,000 training set into two sets of 50,000 data points for training and 10,000 data points for validation.

The training parameters of the network are shown in Table 4.5. To map the NReLU to the spiking neuron, we employ the model in Eq. (4.4) and Eq. (4.6), where the parameters μ_μ, μ_σ, σ_μ and σ_μ are obtained from Fig. 4.3b. We perform the inference using this model, where the intensity values of the MNIST images were normalized to values between 0 and 1. For inference, we feed the 10,000 testing set to the network to get the classification accuracy. We generate Poisson distributed spiking trains for each image pixel, with firing rates proportional to the pixel's intensity value. The simulation time step is 1 ms. To verify the impact of A_t, we perform inference on the validation data set with A_t from 1.0 to 2.0. Fig. 4.9 shows the classification error rate as a function of A_t. The base line is the full precision weight network without mismatch of the neurons (σ_μ and σ_σ are zero), which is theoretically the highest accuracy we can obtain. When A_t is too large, too much information is ignored and the error rate increases because of the biased network. On the other hand, when we reduce the value of A_t to lower than 1.4 the error rate will also go up. This is because when A_t is too small, too many irrelevant features or noise learned by the layer is maintained. This causes the overfitting error of the network to increase. To get the optimized performance, we choose $A_t = 1.6$.

To show the accuracy degradation due to weight ternarization, NReLU to ReRAM mapping and

device mismatching, we simulate on the MNIST testing set with four networks. These networks are a NReLU network with full precision weights (base line), a NReLU network with ternary weights (for weight ternarization), an IF network with full precision weights (for device mismatch), and an IF network with ternary weights (for device mismatch and weight ternarization). Table 4.6 shows the classification accuracy of the four scenarios. The performance loss caused by ternarization is negligible—under 1%. However, due to the extra overfitting error introduced by mismatch between ReRAM devices, the accuracy loss that directly maps the NReLU with IF neurons is significant (over 5%). Fortunately, we can use the weight ternarization as the regularization against the mismatch and reduce the accuracy to roughly 1% using the ternarized weight.

Table 4.5: Train Parameters

Parameters	Value	Parameters	Value
Learning rate	0.1	Batch size	100
Number of epochs	20	Momentum	0.5
Number of neurons	3794	Number of synapses	2,794,000

Table 4.6: Classification performance on the MNIST test set of the DBN, the ternarized DBN and the spiking DBN

Neuron model	Weight Precision	Accuracy (%)
NReLU	Full Precision	95.86
NReLU	Ternarized	95.29
IF	Full Precision	90.23
IF	Ternarized	94.7

One drawback of the spike based neural network is that it needs multiple simulation steps to converge to its final accuracy. This leads to multiple reading of the weights from memory, low throughput and high power consumption [37, 80]. Thus we depict the time domain accuracy curve in Fig. 4.10. We multiply μ_μ, σ_μ, σ_μ and σ_σ with a coefficient c_i to make the first-order approximation of the neuron transfer function change regarding the global input parameters, such as input pulse voltage and pulse width. It can be seen that the global parameters of the network will have a major impact on the convergence time.

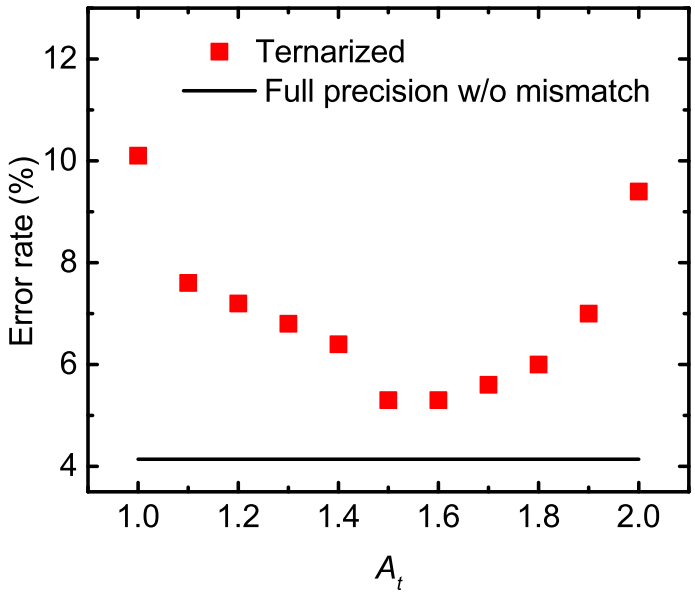


Figure 4.9: Classification error rate as a function of A_t. Large A_t will filter too much information learned by the layer and increase the error rate. On the other hand, small A_t will keep too much minor information and amplify their magnitude to the major information, leading to dropping of accuracy. The base line is the accuracy of the double precision weight.

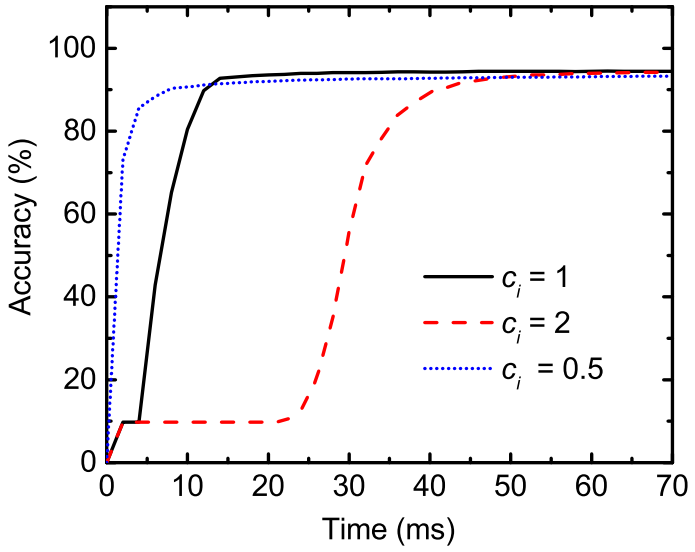


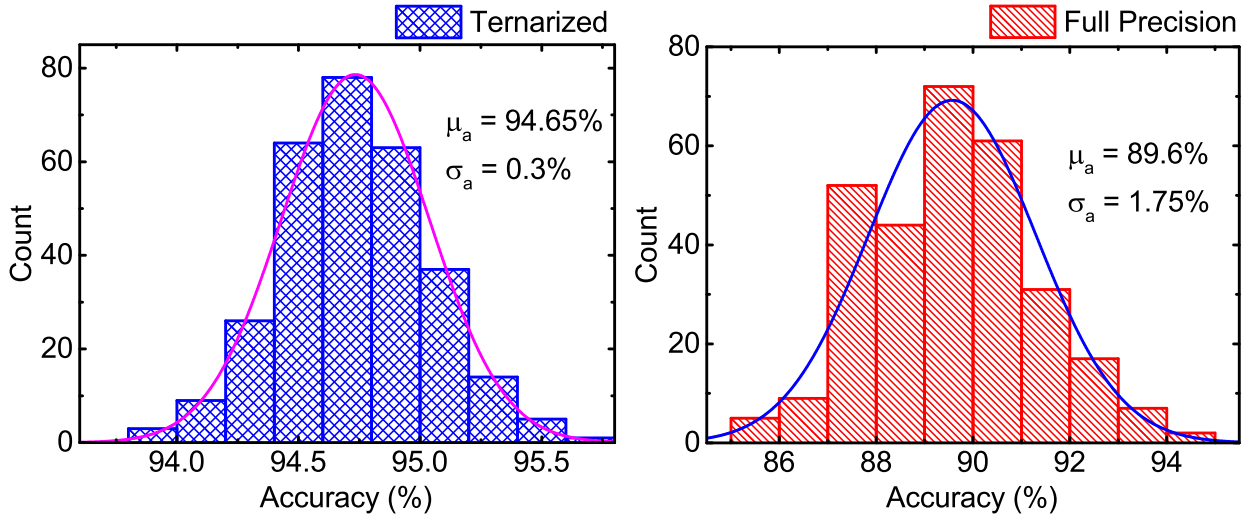
Figure 4.10: Time to first output spike and performance based on the first output spike. All 10,000 MNIST test examples were presented to the spiking DBN for 70 ms. We change the parameters of the neuron globally by multiplying the μ_μ, σ_μ, σ_μ and σ_σ with a coefficient c_i

This is because when the input ISI/output ISI is high (when voltage pulse height or pulse width is low), more input spikes are needed to activate the neuron and thus the time needed for the neurons in the label layer to get activated increases. However, some information will be lost when the neuron is too sensitive and cause the final accuracy to drop (see the curve with $c_i = 0.5$). Therefore, the global parameters of the network need careful selection to balance the accuracy and throughput/power consumption.

The model in Eq. (4.4) will change the distribution of the parameters of the neurons with each initialization. We utilize this method to generate 300 neuron networks and perform inference on them to examine the impact of process variation on accuracy. We also perform the inference on full precision networks to illustrate the robustness of the ternarized network against device mismatch. Fig 4.11a shows the accuracy distribution of the ternarized networks. The ternarized networks show good average accuracy (μ_a = 94.64%) and are very robust against process variability (σ_a = 0.3%). On the other hand, process variation reduces the accuracy of the full precision network to 89.6% (see Fig. 15b) and its standard deviation is 1.75%, which is much higher than that of the ternarized network.

4.4 Conclusion

A capacitor-less stochastic neuron circuit using the change of a conductive filament to represent the membrane potential has been designed. Using the 65nm CMOS technology node, the circuit is 14×5 μm^2 in size and consumes 1.28 μW on average power dissipation. The output spikes follow the Poisson distribution and can be used in stochastic spiking networks. We propose the Neural Array that integrates the ReRAM-based neurons with ReRAM synapses. To mitigate the implementation complexity of analog full precision synapses and depress the error raised by device mismatch, we use the ternary weight for the synapses.



(a) (b)

Figure 4.11: (a) The ternarized networks show good average accuracy (μ_a = 94.64%) and is very robust with σ_a = 0.3%. (b) The full precision network is more influenced by the device mismatch. The μ_a is degraded to 89.6% and the σ_a is 1.75%.

Compared to the 1T1R ReRAM memory crossbar, the Neural Array has a smaller area overhead of 0.74% and a power overhead of 13.35%. We develop a simple algorithm using only one parameter A_t to ternarize the weight. The accuracy degradation, due to ternarization and mapping, is about 1%. Furthermore, we utilize the sparsity from the ternarization and reorder the weight matrices with the extended Cuthill-McKee algorithm to reduce the device usage for the synapses. The number of ReRAM cells used is reduced by 36% and 33%, indicating smaller area usage and lower power consumption. The neural network is robust against the ReRAM process variation effect. The change of transfer function of the ReRAM will only affect the converge time.

CHAPTER 5: A SCALABLE AND RECONFIGURABLE IN-MEMORY ARCHITECTURE FOR TERNARY DEEP SPIKING NEURAL NETWORK WITH RERAM BASED NEURONS

[1]In this chapter, we introduce the Process Element (PE), a modified memory array that utilizes the one-transistor-one-ReRAM (1T1R) based neurons and synapses to implement spiking neural network in memory. The PE comprises one route table to store route information, one synapse array and one neuron array with modified periphery circuit to allow the 1T1R synapses to receive the input spikes and deliver them to 1T1R neurons for membrane potential updating. The PE has the same scalability as the memory array with a limited circuit add-on. Based on the PE, we propose a three-layer architecture for deep spiking neural networks. The bottom layer is built upon the PEs with a control unit to send the spike to targeted neurons. Several PEs are connected to communicate with each other through a local router to form the second layer of the architecture, Process Element Matrix (PEM). All PEMs, along with memory matrices, are connected to a global router to assemble the topmost layer of the architecture. As a result, the architecture is highly scalable by changing the size of the PE.

To achieve full-reconfigurability, we use the address event representation (AER) handshake sequence [92] as the communication protocol within the architecture and design the routing scheme that supports both intra-PEM and inter-PEM routing. Different from the programmable switch paradigm [30], AER is a bus-based communication scheme that can be easily integrated into the memory bus controller [92, 93]. It also provides connections between arbitrary PEs in our architecture (the programmable switches can only connect adjacent PEs). The architecture is fully reconfigurable by just changing the data stored in the route tables.

[1]This chapter was published as Lin, Jie, and Jiann-Shiun Yuan. "A scalable and reconfigurable in-memory architecture for ternary deep spiking neural network with ReRAM based neurons." *Neurocomputing* 375 (2020): 102-112.

In summary, our key contributions are:

- We employ the ReRAM based neurons as the computing components to update the membrane potential and integrate the neurons within a ReRAM memory array of synapses and route tables and propose the PE—a spiking-neural-network-in-memory array. To the author's best knowledge, this is the first attempt to merge all the computing and information of the SNN into a memory array.

- We further propose a reconfigurable and scalable architecture based on the PE and present an AER routing scheme to accommodate different network topologies. We extend AER spike communication from a flat architecture to a fractal hierarchy which includes the intra-PEM routing and the inter-PEM routing. This extension is critical in scaling up SNN systems towards levels of modern DNN models.

- To address the design challenges in ReRAM based architectures, which we discuss in detail in Sec. 5.1.3, we train our SNN model with ternary weights and show the robustness of ternary SNN under device variation and random setup process.

- We evaluate our architecture with different spiking network topologies (spiking Resnet (SResnet) and spiking Squeezenet (SSqueez)) on two datasets (MNIST, CIFAR-10) and compare our architecture's performance with a digital/analog neuron baseline.

The rest of the chapter is organized as follows. Section II introduces the related background and the challenges of our work. Section III describes the design of the PE with modification of the peripheral circuits. Section IV proposes the PEM architecture design, especially the intra-PEM and inter-PEM communication. Section V uses case studies of two data sets (MNIST and CIFAR-10) on two spiking neural network examples to analyze the accuracy, overhead, and energy consumption of the architecture. Finally, Section VI draws the conclusion.

5.1 Preliminaries and Challenges

5.1.1 Deep Spiking Neural Network

In SNNs, the input is encoded as spike trains and involve spike-based (0/1) information transfer between neurons. At a particular instant, each spike is propagated through the layers of the network while the neurons accumulate the spikes over time until the membrane potential exceeds the threshold, causing the neuron to fire. In [37, 94], the authors map pre-trained deep neural networks (DNNs) to SNNs to perform low energy cognitive tasks. Also, in [94], Rueckauer et al. realize some auxiliary layers in DSNN, such as Batch Normalization (BN), Max pooling and Softmax, which dramatically reduces the gap of accuracy between DNN and DSNN. Another advantage of the direct mapping method is that it utilizes a simple IF neuron model, which can be implemented with ReRAM based neurons. In our work, two deep SNN topologies are evaluated—a very deep Resnet [95] and a condensed Squeezenet [96].

5.1.2 ReRAM Based Neuron

The basic device structure and the underlying physics involved for the ReRAM relies upon the formation and rupture of the nanoscale conductive oxygen vacancies [75]. For the neuron, the neuron membrane potential can be emulated with the growth of conductive filament in a ReRAM [48,49]. We define the membrane potential $u(t)$ with the filament length $u(t) = g_{max} - g(t)$, where g_{max} is the maximum gap distance and $g(t)$ is the gap distance in real time. Moreover, positive feedback between $u(t)$ and temperature causes an abrupt setting of ReRAM devices [28] and can be used to imitate the integrate-and-fire model. As shown in Fig. 5.1a, following successive applications of the voltage pulses on the top electrode (TE), $u(t)$ (the red pillar) progressively increases, enabling the temporal integration of spikes in the ReRAM device. When $u(t)$ exceeds

the threshold, the ReRAM based neuron is "firing", and then a reset voltage will be applied to the bottom electrode to reset the ReRAM to the initial condition. We redraw Fig. 4.6a here as Fig. 5.1b to show the functionality of the ReRAM based artificial neuron. The second row shows the evolution of the artificial membrane potential with the input spikes. When the $u(t)$ exceeds the threshold, the filament length will increase abruptly and is detected by the spiking circuit. Thus a voltage spike (in the third row) will be generated and injected into the on-chip network. The activation function in Eq. (4.4) and Eq. (4.5) can be approximated with a binary stochastic neurons (BSN) [97] with the activation function

$$\text{BSN}(a) = 1_z < sigm(a) \tag{5.1}$$

where 1_x is the indicator function on the truth value of x and $z \sim U[0,1]$, and $sigm(x)$ is the Sigmoid function. Therefore, we use the BSNs to model the stochastic behavior of the ReRAM neurons.

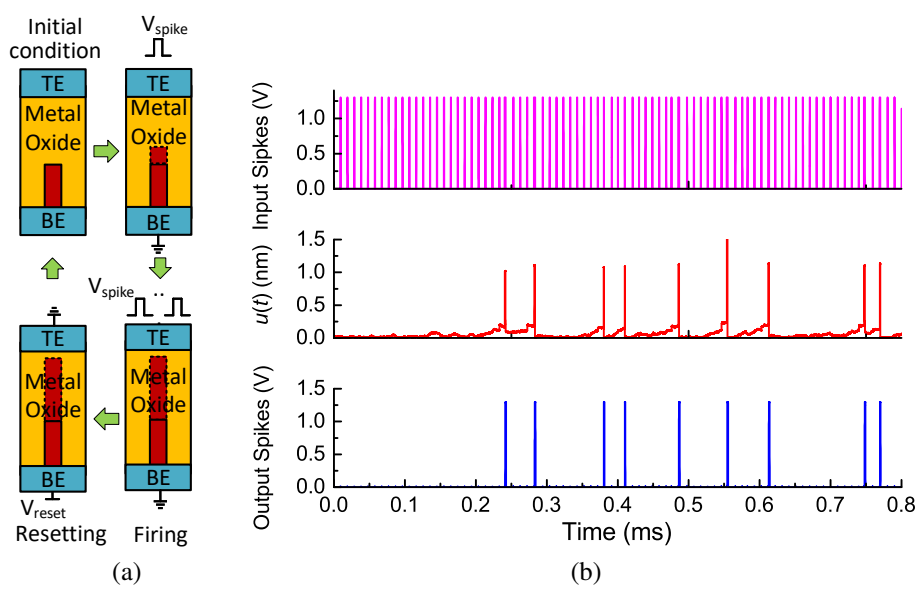


Figure 5.1: (a) A demonstration of integrating, firing and resetting of ReRAM based neuron. (b) Simulation results of the membrane potential $u(t)$ (the second row) and output spikes (the third row) versus input spikes (the first row).

5.1.3 Design Challenges

Compared with the analog and digital neurons, some challenges exist in the ReRAM based neuron design, such as the nonlinear response [48], large process variation [26, 48] and stochastic setup process [28, 48, 49]. Also, limited bit levels of ReRAM [98] hinder the precision of ReRAM synapses. Therefore, incorporating high precision of data and weights in SNNs becomes the main challenge for NVM implementation. Researchers in the field of machine learning have demonstrated that ternary networks achieve satisfying recognition accuracy on ImageNet dataset [99, 100]. Ternary networks use ternary weights $w \in \{w_n, 0, w_p\}$ when processing the forward propagation. It provides a promising solution to break the high precision limits in NVM SNN accelerator design. From [48], the impact of process variation is negligible when ternary weight is utilized. Also, the stochasticity can be modeled with device parameters using Eq. (4.4) and Eq. (4.5) and added into the IF function during training to reduce the accuracy loss [48]. Therefore, it will contribute a lot to energy efficiency if achieving a well-trained network model with ternary level weight parameters and feature maps.

5.2 Process Element (PE) Design

The ternary SNNs can to address the design challenges of ReRAM-based neurons. Moreover, the low precision 2-bit weight can simplify the hardware design and reduce overhead, e.g. the ADC/DACs [20, 101] and pulse-width-modulation (PWM) circuity [47] with high overhead can be removed. Based on these observations, we propose the PE, a novel ReRAM-memory array implementing spiking neurons, ternary synapses and routing information that connects the neurons to form the spiking network. The fundamental building block of the PE is the 1T1R cell. Fig. 5.2 shows the configuration of a 1T1R cell that functions as a synapse (up), and a neuron (down). For the synapse implementation, a pulse voltage is applied to the gate of the selection transistor of the

1T1R cell (also the word line (WL) of the memory cell). The presynaptic spike level V_{pre} is set to $V_{DD}/2$ to save power and avoid to write the ReRAMs accidentally. V_{spike} is the voltage pulse generated by the spike generator (see Sec. 5.3.2.1) that turns on the select transistor for a duration equaling the pulse width to develop or rapture the conductive filament and to reset the ReRAM.

The neuron cell is a modified version of the synapse cell which allows a reverse voltage pulse on the ReRAM electrodes under the control of signal *RESET* to perform the reset of the ReRAM. In Fig. 5.2, V_{setr} and V_{resetr} are the two voltage sources to develop the conducting filament and reset the ReRAM after firing. The resistance of the ReRAM is always positive. Therefore, we use a similar manner as in [47] that employs a reversed voltage pulse to partially reset the ReRAM device to mimic membrane depression causing by a negative postsynaptic signal. V_{inhr} is the voltage to perform depression of the neuron by partially resetting the ReRAM. Signals pos and neg are the sign of the postsynaptic signals. The post-synaptic pulse V_{post} controls the select transistor of the ReRAM cell. The additional switches in the 1T1R neuron cell will not introduce the much overhead to our architecture because the number of neurons is much less (over two orders of magnitudes) than the number of synapses.

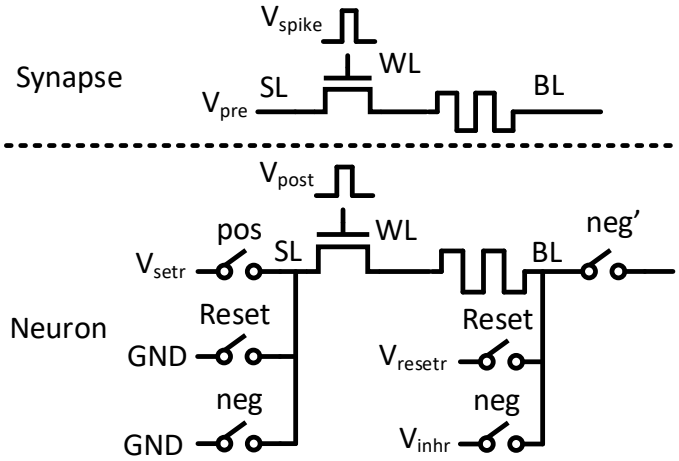


Figure 5.2: Signal of WL, SL and BL of the synapse (up) and neuron (down).

Fig. 5.3 shows the detailed design of the PE. In Fig. 5.3, the whole array is divided into three

groups—two 2D arrays for the route table and the synapses respectively, and one 1D array for the neurons. The route table array and synapses array share the same WL, and all three groups share the same word line (WL) and WorD line Decoder (WDD). One neuron in the PE is corresponding to one row in the route table that stores the destination of the event address it generates and a column in the synapse array which is the fan in of the neuron. The neural network can be easily scaled up by just enlarging the size of the array linearly. We use two ReRAM synapses to map one ternary weight. As a result, the ratio between the number of 1T1R synapses and the number of 1T1R neurons is $2R$ where R is the row of the PE.

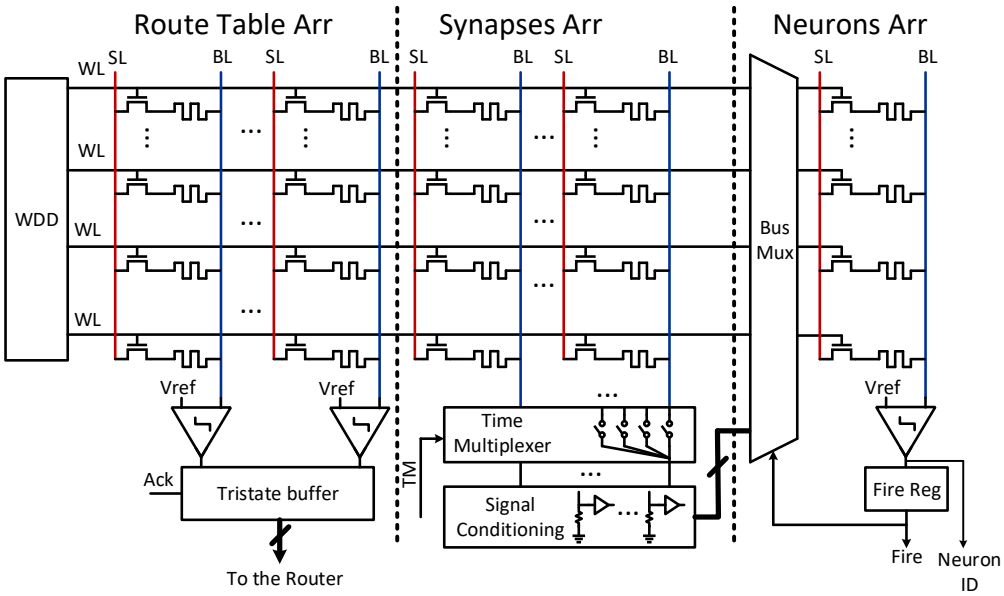


Figure 5.3: Schematic of the PE. The memory array is divided into three groups: route table array, synapses array, and neurons array. They have the same WL, BL and SL structures as the traditional memory array, and therefore, the same scalability as the memory array.

The PE receives the input spikes by connecting the selection line (SL) to voltage source V_{pre}, as shown in Fig. 5.2, and sent spikes to the WL through the bus multiplexer. A signal conditioning circuitry is connected to the bit line (BL) of the 1T1R synapses that amplifies the level of the post-synaptic spike to V_{DD} to drive the selection transistors of 1T1R neurons through the bus multiplexer. Thus, in our design, the WL of the synapses emulates the axons of the sending neurons,

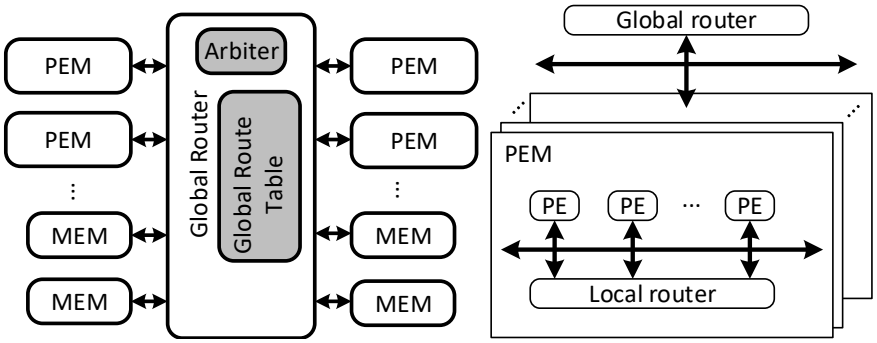
which transfers pre-synaptic spikes while the BL is the dendrites of the receiving neurons which transfers the post-synaptic spikes. Therefore, in our later discussion, we call the WL address of the synapses "axon ID". Every column of 1T1R synapses composes the fan-in of a 1T1R neuron with a physical interconnection between them to maximize the throughput. In case that the fan-in of some neurons, such as the ones in the fully-connected layer of the neural network, exceeds the crossbar size, we employ a time-multiplexing (TM) scheme to allow the neuron to receive the spikes from multiple BLs of the synapses array. When the fan-in exceeds the limit, the neuron can receive the spikes from the next three synapse columns through the time multiplexer, making the maximum fan-in $4\times$ of the physical fan-in.

With the development of conductive filament of the ReRAM, the 1T1R neuron can integrate the input spikes until it reaches the firing threshold. When the neuron fires, the input spike will trigger the sense amplifier (SA) to output an "1". Note that this signal is used to control the readout of the firing ReRAM and initiate the handshake communication only. Development of conductive filament and reset of ReRAM is controlled by V_{spike} and the spike generator. The fire reg will store this signal and toggle the bus multiplexer to connect the WL of the 1T1R neuron to the WDD. Sequentially, the resistance state of ReRAMs in the neurons vector is read out through the memory read circuit to identify the firing neuron's ID. We add an extra tristate buffer, through which the route table array is connected to the local bus that connects adjacent PEs to the router.

5.3 Reconfigurable in-Memory Architecture for Deep Spiking Neural Network

The topmost level among the three reconfigurable hierarchies of the in-memory architecture is shown in Fig. 5.4a. As shown in 5.4a, the MEMs are memory matrices that only have data storage capability to store the input AER packets and the global route table. Their micro-architecture and circuit designs are similar to the design in [102]. The process element matrices (PEMs) com-

prise of matrices of the PE in Fig. 5.3 and their associative periphery circuits. The MEMs and PEMs communicate with each other through a global router, and a global route table encodes the addresses. A global arbiter is employed to resolve contention for channel access.



(a) (b)

Figure 5.4: (a) Topmost level of the in-memory architecture. (b) Routing hierarchy.

5.3.1 Routing Scheme

AER protocol was introduced as an efficient means for point-to-point (P2P) communication of neural spike events between arrays of neurons, in which addresses of neurons are communicated over a shared digital bus, whenever they fire [93]. AER supports event-driven operation, which is the major advantage of SNNs over DNNs, through a four-phase handshake sequence. Moreover, AER is scalable and can be used to build large-scale network-on-chip [92, 93].

The routing scheme of our architecture comprises intra-PEM routing, which transfers address events within the same PEM and the inter-PEM routing, which transfers the address events throughout the whole architecture. As shown in 5.4b, the address events are partitioned in a hierarchically optimal manner. The local routing in each PE is performed in parallel. Each local router recognizes the event by a local address, unique to the presynaptic neuron and routes the event to parents (the global router), children and/or siblings (the local routers of other PEMs) as needed, based on the

connectivity at that level of spatial scale. It utilizes the local route table as a low latency address look-up to identify the next destinations of the event. When an event arrives the local arbiter, the local router will decide whether the addresses are physically located within this PEM. If the recipients are in this PEM, the addresses are sent to the WDD to finish this routing. If the destination is in a different PEM, the targeting information is placed on the global bus, and the global route table will be looked up to find out the target PEMs.

Table 5.1 shows the layout of the AER packet. In our design, every firing neuron's destination is encoded through the local route table in the PE and then sent to a local router. At this moment, the AER packet is 21 bits, including the 1-bit header that indicates whether the event is routing within the same PEM or send to the global bus to other PEMs, the 10-bit fan-out information and the 10-bit axon ID. The fan-out is the number of target neurons that the address event will be sent to, and axon ID is the address of the targeting WL. Henceforth, the total number of routable neuron in each PEM is $2^{10} = 1024$ with the max fan-out of 1024. If the header indicating local/global routing is "1", the event will be sent to the global router, and another 6-bit header (PEM ID) is added to the packet through the global route table to identify the ID of the PEMs that connected to the global bus. As a result, the total routable neurons in our architecture are 64K. Finally, a parity bit is added to the packet. The total number-of-bit of the AER packet is 28.

Table 5.1: AER packet layout

Bits	Description
0	Parity bit
1-6	PEM ID
7	Header of local/global routing
8-9	Time multiplexing config
10-17	Fan out
18-19	PE ID
20-27	Axon ID

5.3.2 PEM-Configurable and Scalable Process Element Matrix

A PEM is composed of multiple PEs tied together to the local router and the arbiter tree. The design goal for PEM is to support both storage and computation with a minimum area overhead. To achieve this goal, we maximize the reuse of peripheral circuits for both storage and computation. Fig. 5.5 includes four PEs, each of which has PE with a 256×21 route table array, a 256×512 synapses array and a 256×1 neurons array. A predecoder of conventional in-memory architecture [103] is used for PE selection. One spike generator (Fig. 5.5 ①) is shared by all PEs that converts the incoming address event into a voltage pulse and generates the voltage pulse to reset the neurons. A PE Control Unit (PECU) (Fig. 5.5 ②) provides operation and input/output control. The local router (Fig. 5.5 ③) supports complex interconnect between neurons within the same PEM and between different PEMs. The data communication follows the four-phase handshake protocol of AER, which is shown in Fig. 5.5 ④. We also use the handshake signals Req and Ack to control the spike generator and finite state machine (FSM) in the PECU and the local router. To prevent the packet loss when multiple neurons want to send a spike event simultaneously, we employ an arbiter tree to set up a queuing mechanism [92] to allow neurons to wait for their turn.

5.3.2.1 Spike Generator

In order to convert the digital address event signal to the voltage pulse that can set and reset the ReRAM, we utilize the circuit we proposed in our previous works [48,49] as the spike generator, as shown in Fig. 5.5 ①. The spike generator is triggered by the enable circuity under two conditions. When signals Req and Ack are both high, which means the targeting synapses have been selected, the spike generator sends a pre-synaptic pulse to this synapse. When signals Reset and Ack are both high, which means the neuron is ready to be reset, the spike generator sends a post-synaptic pulse to the neuron to reset the ReRAM.

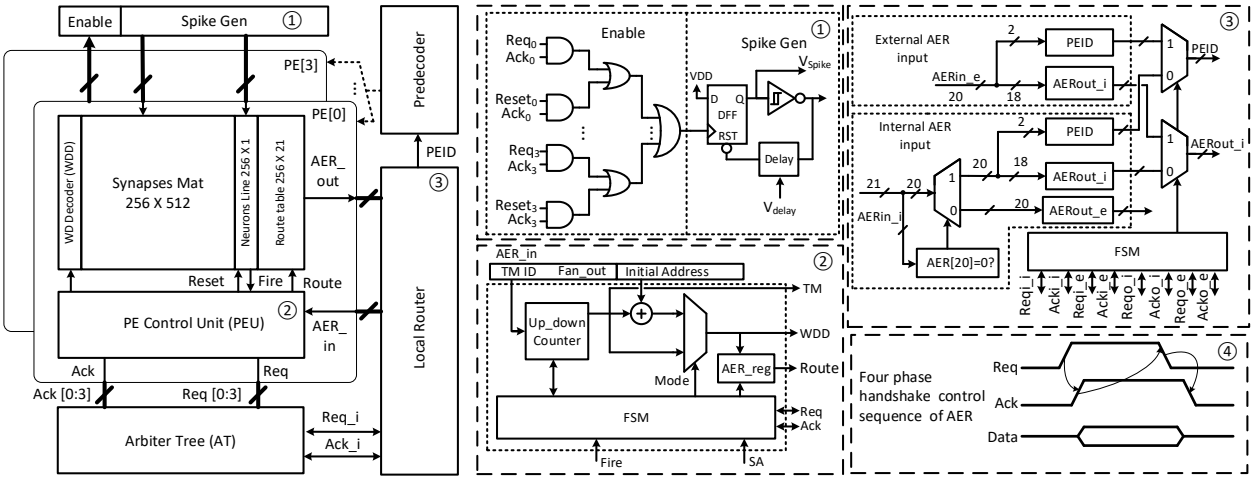


Figure 5.5: Left: Process Element Matrix structure. Right: functional blocks in PEM and the four-phase handshake sequence. ① Spike generator to convert the spike events into voltage pulses; ② Process element control unit that controls the operation of the PE; ③ Local router that routes within the PE and between PEs in the PEM; ④ The four-phase sequence.

The specific functionality of control signals is shown in the following section. The enable tree will generate a rising edge to trigger the D-FF. A delay unit generates a delayed reset signal to the D-FF. Hence, a voltage pulse is produced by this feedback loop comprising the D-FF and the delay unit. The pulse width of the output spike is determined by a voltage controlled delay line (VCLD). Since the energy of the pulse train is proportional to the pulse width, less wide pulses are needed to set the ReRAM than narrow pulses. Thus, by tuning the pulse width, we can set up the number of pulses that can trigger the artificial neuron, which effectively set up the threshold of the neuron. We employ rate coding scheme in our architecture for its simplicity and straightforward circuit implementation.

5.3.2.2 PE Control Unit

Fig. 5.5 ② shows the PECU design. We reuse the counter that heavily operated to reduce the overhead. The PECU controls the input/output and spike generation of the PE. Also, the PECU

controls the set/reset of the ReRAM based neuron. Moreover, the PECU conducts the four-phase handshake sequence shown in Fig. 5.5 ④. Fig. 5.6 shows the simplified finite state machine (FSM) chart of the PECU. In Fig. 5.6, when receiving the input AER packet (the signal Req is high), the down-counter will generate the targeting columns of the synapses and send the values to the selection line decoder (SLD), according to the fan out bits in the packets. Thus the targeting synapses will be connected to V_{pre}, as shown in Fig. 5.2. Also, one axon register will store the information of the axon ID bits and send them to the word line decoder (WDD). If all the synapses are selected (at this time the counter = 0), the PECU will set the signal Ack high and trigger the spiking generating circuit. If the time multiplexing (TM) is needed, the TM bits will also be sent to a counter to control the TM switches.

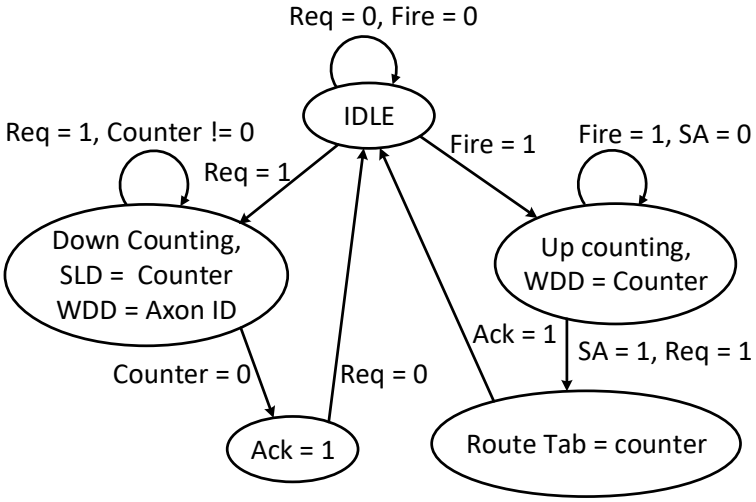


Figure 5.6: Simplified FSM chart of the PECU. The loop on the left side is the working procedure of the PECU to fetch the input AER packet and send the voltage spike to the recipient neurons. The loop on the right side shows the output process that the PECU detects a neuron is firing and generates an output AER packet and sends it to the local router.

When one of the neurons fires, the Fire signal is high. PECU will start an up counter and begin to read out the ReRAMs in the neurons array. Then, the corresponding row in the route table will be read out and sent to the local router. Then, the PECU will set Req to 1 to start the receiving sequence of the subsequent neurons. Finally, the PECU will reset the Req signal to end the

handshake.

5.3.2.3 The Router

As shown in Fig. 5.5 ③, the local router allows the neuron to send the spike event within the PEM and between PEMs. The local router has two AER input ports for internal routing (intra-PEM) and external routing (inter-PEM), respectively. Also, the local router has four pairs of handshake signals (Req and Ack) to fulfill the handshake sequence. To minimize the delay, we set that the intra-PEM routing has higher priority than inter-PEM routing, i.e., the internal routing request will be handled first.

Fig. 5.7 shows the handshake signal flow in all routing situations, while the number denotes the order of the phases. For the internal routing mode, the route first exams the Header of local/global routing (H bit). If the address event is going to the same PEM, then the AER packet is parsed, and the PE ID is sent to the predecoder and the other parts of the packets are forward to the PECU of the recipient PE. Also, the Req signal is sent to the PECU FSM to start the handshake sequence. If the packet is sent to other PEMs, the packet is sent to the global bus, and a 6-bit PEM ID will be added to the packet through the global route table. For external routing, the AER will be parsed directly and sent to the according predecoder and PECU. The design of the global router is the same as the external part of the local router, except that the PEM ID will be extracted instead of PE ID.

5.4 Results and Discussion

In this Section, we present the results of various experiments that demonstrate the benefits and the effectiveness of the proposed architecture in exploring the design space of 1T1R ReRAM arrays

for deep SNN applications.

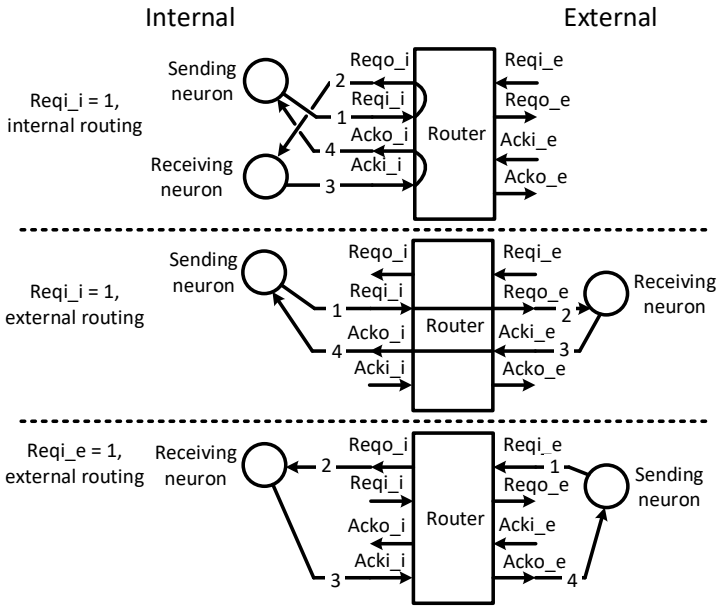


Figure 5.7: Handshake signal flow of the router in three routing situations: internal neuron to internal neuron, internal neuron to external neuron and external neuron to internal neuron. The numbers indicate the phase of the handshake protocol. The router also pass the handshake signals (Req and Ack) to modules that are communicating.

We convert the input images into spike trains with rate code. For every input image, the input activation is sampled as Poisson spike trains with rates proportional to the pixel intensities [104].

5.4.1 Ternarization of SNNs

In [48], the design challenges of the ReRAM-based neurons are addressed by the ternarization of the spiking deep believe networks (SDBN). In this section, we extend the results of [48] by ternarizing spiking Resnet and spiking Squeezenet to facilitate mapping the SNNs to our architecture. We construct the SNN model by direct mapping the pre-trained DNNs to SNNs [94]. We use the following procedure to convert trained DNNs into IF based SNNs: 1) Use Binary Stochastic Neurons (BSNs) [97] for all units of the network to mimic the stochastic setup of the ReRAM shown in

Eq. (4.4) and (4.5), 2) Fix the bias to zero throughout training, 3) Implement special SNN layers, such as Batch Normalization, Max pooling and Softmax, and 4) Perform weight ternarization and normalization to train synapses and obtain weights in the DSNN.

5.4.1.1 Weight Ternarization

We ternarize each weight to $\{-1, 0, 1\}$ with two thresholds: t_i^{lo} and t_i^{hi}. The choice of the t_i^{lo} and the t_i^{hi} is a trade-off between loss of information and sparsity of the weight matrix [100]. To optimize the t_i^{lo} and t_i^{hi}, C. Zhu et al. [100] solved the optimization problem of minimizing L2 distance between the float-point weight matrix W^i and the ternary weight matrix W_{ter}^i. The resulting thresholds are $t_i^{hi} = -t_i^{lo} = \frac{0.7}{n}\sum_j w_j^i$, where n is the total number of elements in the W^i. However, most of the network models are over-parameterized [105], leading to overly conservative threshold choices. In [105], S. Han et al. pruned the connections between layers during training and reduced the number of weights by $9\times$ while keeping the original accuracy. Therefore, it is feasible to optimize the W_{ter}^i with a small portion of entries in W^i. As a result, we propose the new threshold equation as

$$t_i^{hi} = -t_i^{lo} = \frac{0.7}{n}\sum_j w_j^i\big|_{w_j^i > w_{th}} \tag{5.2}$$

where w_{th} is the threshold below which the entry is pruned. We perform an experiment of SResnet on CIFAR-10 dataset to fine-tune w_{th}. We compare our results with the one using C. Zhu et al.'s threshold equation [100] in Table 5.2.

Table 5.2: Influence of Pruning on Ternarization

	With Pruning (Ours)	Without Pruning [100]
Weight Sparsity	11%	49%
Accuracy	92.11%	88.53%
Connections	0.48 M	2.12 M
Neurons	13.8 k	13.8 k

From Table 5.2, with the pruning, we can increase the weight sparsity to nearly $5\times$, which significantly reduces the number of computation. Moreover, the accuracy is improved by 3.6% using the pruning because that overfitting is suppressed by pruning the connections.

5.4.1.2 Threshold Normalization

In the DSNNs, the firing rate of the neurons is restricted to the range of [0, $rmax$], whereas BSNs in DNNs do not have such constraints. Weight normalization avoids the approximation errors due to too low or too high firing rates [37, 94]. However, with the ternarization, the weights are limited to $\{$-1, 0, 1$\}$, which is not normalizable. Thus we multiply the threshold with the normalization constant to effectively normalize the weights. To show the effectiveness of threshold normalization, we exam the activation function of the IF neuron after weight normalization:

$$s_i = \begin{cases} 1, & \text{if } \frac{1}{a_n}\sum_j s_j w_{i,j} < V_{th}. \\ 0, & \text{if others.} \end{cases} \tag{5.3}$$

where s_i, s_j are the spikes generated in the input and output layers, respectively, and a_n is the normalization coefficient. Eq. (5.3) is equivalent to

$$s_i = \begin{cases} 1, & \text{if } \sum_j s_j w_{i,j} < a_n V_{th}. \\ 0, & \text{if others.} \end{cases} \tag{5.4}$$

which normalizes the weight $w_{i,j}$ by a_n.

5.4.1.3 Implementation of Auxiliary Layers

The auxiliary layers of the SNNs, such as Batch Normalization, Max pooling, and Softmax, play critical roles to increase the accuracy of the SNN. The implementations of those special layers are proposed in recent literature [94, 106]. We modify those designs to realize the special layers for the ternary SNNs. The Batch Normalization (BN) layer transforms the inputs to achieve zero-mean and unit variance through the relation $BN[x] = \frac{\gamma}{\sigma}(x - \mu) + \beta$ where mean μ, variance σ, and the two learned parameters β and γ are all obtained during training [107]. The transformation can be integrated into the weights after training, thereby eliminating the need to compute the normalization repeatedly [94]. Specifically, we set $\hat{w}_{i,j} = \frac{\gamma}{\sigma} w_{i,j}$ for the scaling part of BN, and add a constant $\hat{b}_i = \beta - \frac{\gamma}{\sigma}\mu$ or account for the offset part of BN, which can be presented with an input spike train with constant rate.

In [94], Rueckauer et al. implement the spiking max pooling with a pooling gate that is parameterized by the history of the input neurons, which will raise the memory and computation cost. To reduce the memory and computation usage, in our design, we employ the max pooling gate proposed by Orchard et al. [106] that judges the neurons fires first having the maximal response to the stimulus in that sampling time step. This judgment is based on two main observations. Firstly, a stronger stimulus will generate spike trains at a higher rate. Secondly, the stronger the neuron's input weights are correlated with the spatial pattern of incoming spikes, the faster the membrane potential will increase, leading to faster firing. Thus the max pooling gate will only propagate the spikes from the neurons that fire first during one sampling time step.

To implement the Softmax layer, we use counters instead of ReRAMs to integrate the input spikes and compute the Softmax on each counter's output. Usually, the Softmax layer only appeared as the last layer of the neural network, and the number of neurons in that layer is limited (which is equal to the number of classification categories). Thus we can assume that the overhead of inserting

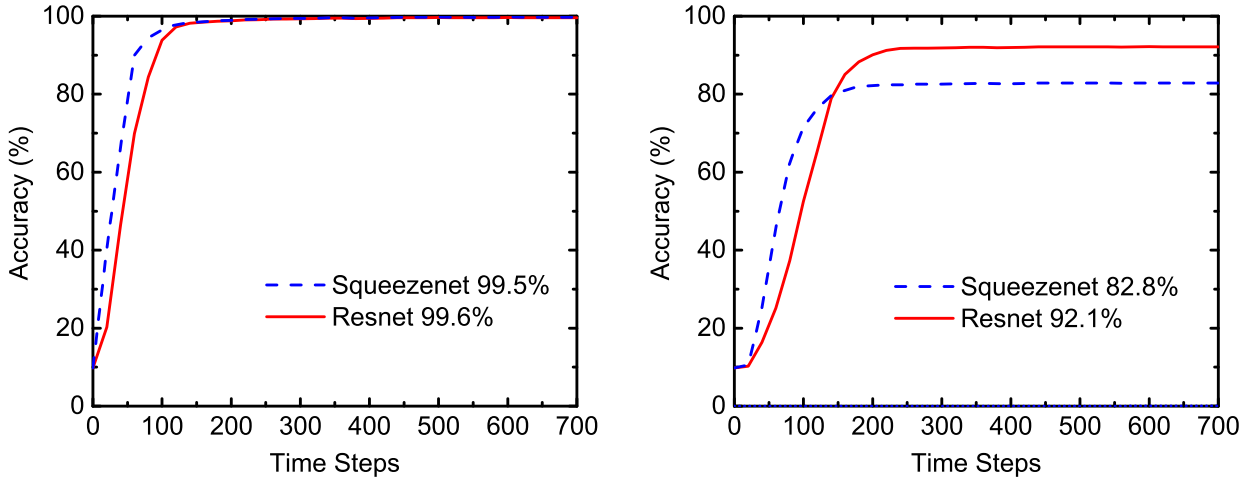
of spiking Softmax layers is insignificant.

5.4.1.4 Accuracy

We evaluate the performance on MNIST [108] and CIFAR-10 [109] datasets. The SNNs and DNNs were implemented in pytorch [110]. On the MNIST dataset, both DNN Resnet and Squeezenet generate the validation accuracy of 99.6%. And the mapped DSNN's validation accuracy achieves 99.6% for Resnet and 99.5% for Squeezenet (see Fig. 5.8a). The difference between DNN and DSNN is negligible because the accuracy of deep network models saturates for MNIST dataset. On the CIFAR-10 dataset, the baselines for DNN Resnet and Squeezenet are trained with validation accuracy of 92.6% and 83.5%, respectively. Our converted DSNN yields the accuracy of 92.1% for the SResnet and 82.8% for the SSqueeze. The accuracy losses of the DNN-DSNN mapping are 0.5% and 0.7%, respectively (see Fig. 5.8b). It is noticeable that the Squeezenet converges faster than the Resnet, indicating that the Squeezenet has a lower latency because of its relatively shallower architecture than that of the Resnet. We also check the influence of the process variation of the ReRAM device. We add Gaussian process variation with 0 mean and 20% standard deviation to every ReRAM device. The results are listed and compared with the ideal device in Table 5.3. From Table 5.3, the influence of the process variation is suppressed by the low precision weights, which coincides the results in [48].

5.4.2 Architecture Evaluation Methodology

Our architecture consists of ReRAM arrays and CMOS peripherals. We build a cross-layer evaluation platform to simulate our architecture. For a ReRAM device, the Verilog-A model based on the equations in [75] is employed for the SPICE simulation with key parameters shown in Table 4.1.



(a) (b)

Figure 5.8: (a) Validation accuracy of SResnet (solid line) and SSqueeze (dashed line) on MNIST dataset versus the number of time steps. (b) Validation accuracy of SResnet (solid line) and SSqueez (dashed line) on CIFAR-10 dataset versus the number of time steps.

Table 5.3: Summary of Number of Operations

Network	Dataset	Process Variation	Accuracy
SResnet	MNIST	0%	99.6%
SResnet	MNIST	20%	99.4%
SResnet	CIFAR-10	0%	92.1%
SResnet	CIFAR-10	20%	91.5%
SSqueeze	MNIST	0%	99.5%
SSqueeze	MNIST	20%	99.5%
SSqueeze	CIFAR-10	0%	82.8%
SSqueeze	CIFAR-10	20%	81.9%

The peripheral circuit consisting of PECUs and local routers is implemented at the Register Transfer Level in Verilog HDL and mapped to 45nm technology with NCSU FreePDK™ and Synopsys Design Compiler. Synopsys Power Compiler is used to estimate the energy consumption. The analog part of the circuit such as the spike generator is simulated using HSPICE and laid out using Cadence Virtuoso Layout Suite. The ReRAM array is modeled using an in-house modified version of NVSim [111] that adds area and power of additional blocks, including tristate buffer, time multiplexer, signal conditioning circuit, and fire register, to the original model. Table 5.4 lists the

simulation parameters and the implementation metrics for our architecture. In Table 5.5, we also summarize the area, power, and delay of the peripheral circuits, including PECUs, local roaters, arbiters in the arbiter tree, tri-state buffers, and spike generators, in a single PEM.

Table 5.4: Parameters and Metrics

Parameters	Value	Parameters	Value
Feature Size	45 nm	VDD	1.6 V
No. of PEMs	64	PEM Size	4
No. of Neurons	64 k	No. of Synapses	32 M
Route Table Capacity	1.34 Mbit	Maximum Fan-in	1024
Gate Count	100288	Area	1.21 mm^2
Frequency	100 MHz	Power	59 mW

Table 5.5: Design Parameters of the Peripheral Circuit of the PEM

Circuit Name	Quantity	Area (μm^2)	Power (μW)	Delay (ns)
PECUs	4	635	32.5	0.33
Arbiters	7	198	20	0.58
Router	1	1340	129	0.69
Tri-state Buffers	4	119	2.63	0.13
Spike Generator	1	128	46	0.98

We do not consider the training phase of the SNN and the energy expended in programming the synapses arrays, because the training is done infrequently at the cloud or computer clusters. On the other hand, the testing or evaluation phase tends to be involved much more frequently. Hence, we evaluate our architecture for the more critical testing phase. Our benchmark comprises the SResnet and SSqueeze we obtained for the MNIST dataset and the CIFAR-10 dataset. The SNNs were trained using the ternarized synapses accommodate the design challenges of ReRAM-based neurons. Table 5.6 shows the benchmark details.

In the evaluated SNNs, except for the input neurons and the counters in the Softmax layer, each neuron has 21-bit route information that is stored in the route table. Thus the total size of the route table is $n \times 21$ bit, where n is the number of neurons in the SNN, excluding the neurons in the first

layer and the counters that constitute the Softmax layer. We summarize the size of route tables of the evaluated SNNs in Table 5.7. The size of the global route table is $m \times 6$ bit, where m is the number of the PEMs used in the SNN. It is easy to observe that the size of the global route table is negligible compared to the local route table, thereby contributing little area and power overhead.

Table 5.6: Benchmark Details

Dataset	SNN Type	Layers	Neurons	Synapses
MNIST	SResnet	18	8.5 k	0.32 M
MNIST	SSqueeze	10	4.2 k	0.18 M
CIFAR-10	SResnet	18	13.8 k	0.48 M
CIFAR-10	SSqueeze	10	6.2 k	0.31 M

Table 5.7: Route Table Size

Dataset	SNN Type	Route table size
MNIST	SResnet	167 kbit
MNIST	SSqueeze	85.6 kbit
CIFAR-10	SResnet	285.6 kbit
CIFAR-10	SSqueeze	126.4 kbit

5.4.3 Performance Evaluation

We compare our architecture with several counterparts. The baseline is a CPU-only configuration simulated using gem5 [112]. The parameters of the CPU is shown in Table 5.8 for simulation. We also evaluate the Prime—an in-memory computing solution [41], with the parameters shown in Table 5.9. The architecture in [41] is no optimized to operate SNNs. As a result, we replace the Sigmoid activation function in [41] with a digital neuron proposed in [103] in the simulation.

Fig. 5.9 compares the energy savings and performance speedups obtained per classification for our architecture over the CPU baseline and Prime on our benchmarks.

The energy saving and the process speed is normalized to the baseline.

Table 5.8: Configuration of CPU

Parts	Parameters
Processor	4 cores; Out-of-order
L1 cache	64 KB; Private 4-way
L2 cache	256 KB ; Private 8-way
L3 cache	24 MB ; Shared
Main Memory	32 GB; 533 MHz IO bus

Table 5.9: Configuration of Prime

Parts	Parameters
Crossbar size	256×256
Speed	100 MHz
Number of SAs	8 per crossbar
Activation Func	Integrate-and-Fire

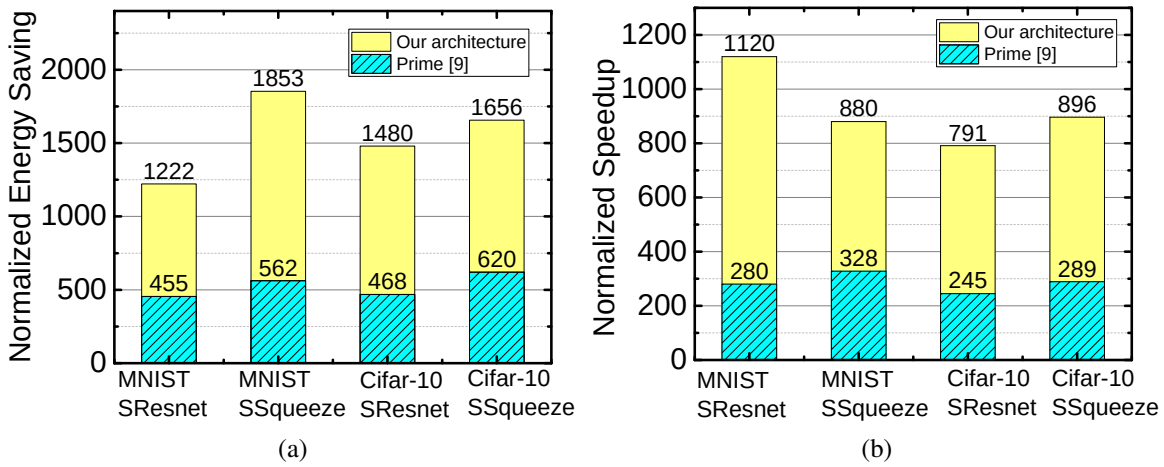


Figure 5.9: Energy Saving and Performance Speedup comparison of our architecture versus CPU baseline and Prime in-memory computing architecture per classification.

As shown in Fig. 5.9a, our architecture provides significant energy savings between 1222× and 1853× (three orders of magnitude improvement). Our architecture also shows over 3× energy improvement of in-memory computing counterparts. For the speed performance, the benefit is between 791× to 1120× and over two times of the Prime. Hence, our architecture efficiently accelerates both network models.

5.4.4 Overhead Evaluation

On the overhead evaluation, we do not consider the connected memory matrix. The ratio of MEM and PEM in the system is the trade-off between computing power and energy consumption. We use the original settings in NVSim [111] for a ReRAM array as our baseline. We model the baseline ReRAM array with the same number of 1T1R cells as our PE. As the cost of enabling in-memory SNN computing and spike event communication, the PEMs incurs 42% area overhead and 127% power overhead. Fig. 5.10 shows the breakdown of the area and power overhead in PEM. The area and power overhead due to computation (involving PECU and spike generator) is 20% and 48.1%, respectively. On the other hand, the blocks that control the spike event communication introduce 21% area overhead and 77% power overhead.

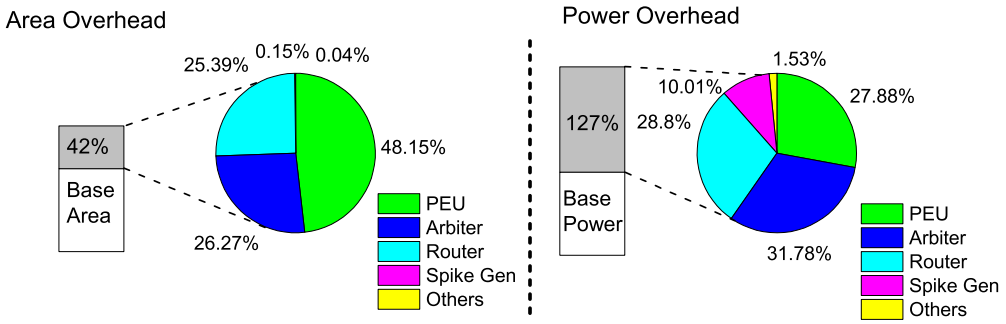


Figure 5.10: Left: the area overhead breakdown. Right: the power overhead breakdown

To evaluate the area and power overhead of the AER routing scheme, we change the baseline configuration to incorporate the operation of the synapses and neurons in the PE, as well as the PECU and spike generator that support the handshake protocol and provide the physical spike signal. The overhead comprises the area and power consumption of the 1T1R cells composing the routing table, the arbiter tree, and the local and global router. We show the overhead breakdown in Fig. 5.11. The total area and power overhead related to AER addressing is 12% and 48%, respectively. Extra 1T1R cells that form the route table introduce 15.96% of the area overhead and 5.66% of the power overhead. On the other hand, the control logic, including the router and the

arbiter, contributes to 84.03% of the area overhead and 94.34% of the power overhead, respectively.

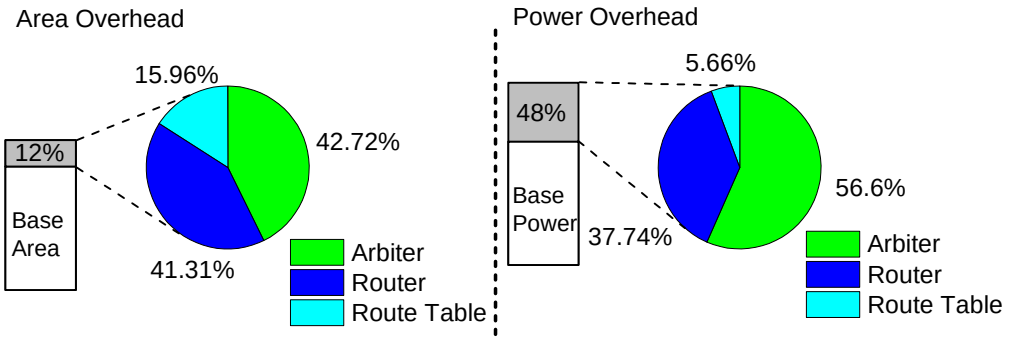


Figure 5.11: Left: the area overhead breakdown. Right:the power overhead breakdown

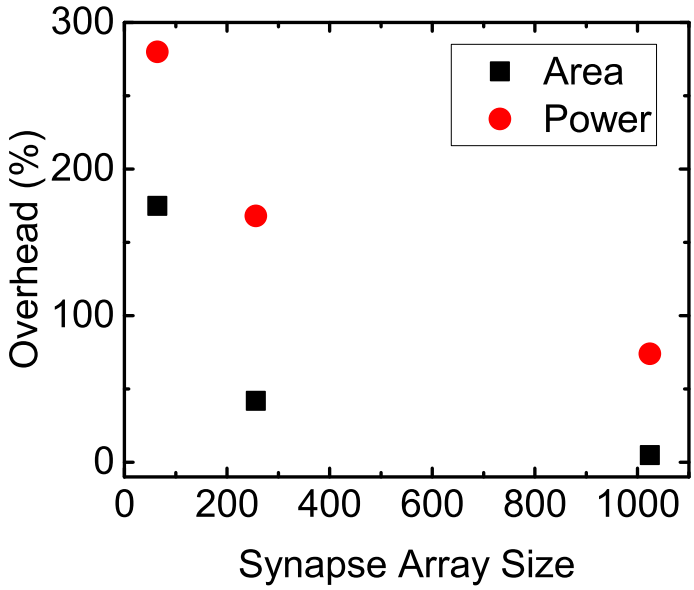


Figure 5.12: Change of overhead with the size of neurons vector

5.4.5 Comparison Between Varying Synapses Arrays

Fig. 5.12 shows the change in area and power overhead when the size of Synapses array is 64, 256 and 1024 while the number of neurons in one PE is kept the same. From Fig. 5.12, the overhead will increase when the size of the vector decreases owing to the increasing amount of the periphery circuit. However, keep raising the synapses array will lead to low utilization of the synapses on-

chip, particularly for the sparse connected CNN. For example, the average synapse utilization of a 64 synapse array for the CIFAR-10 CNN application is around 40%. This number will decrease to less than 3% when the synapse array size increases to 1024.

5.5 Conclusion

In this chapter, we propose the PE, a memory array based spiking neural network implementation. We develop a reconfigurable hierarchy that efficiently implements SNNs of different topologies using the address event representation protocol. Additionally, our evaluation on MNIST and CIFAR-10 datasets for SResnet and SSquessze shows that our architecture is a promising architecture to implement SNNs in-memory and providing good results on performance improvement and energy saving with reasonable overhead.

CHAPTER 6: SUMMARY AND OUTLOOK

As described in this dissertation, by leveraging the potential of emerging device, we designed ultra-low power circuits and architecture for neuromorphic computing systems. TFET devices has a much steeper slope than the CMOS counterparts in the subthreshold region. Thus, It can be used to design high performance circuit working under low supply voltage to increase the energy efficient of the circuit. Also, the intrinsic physics of ReRAM technologies with biological primitives provides new opportunities to develop efficient neuromorphic systems.

In chapter 2, we design a TFET based SAR ADC that can work at the supply voltage under 0.3 V with resolution of over 5 bits. This result is already pass the technology limitation of the CMOS ADCS by a large margin. The ADC can achieve an ENOB of 5.7 to 5.3 with input frequency ranging from 1 to 8 MHz, and The ENOB variation in temperature ranging from -55°C to 125°C is 24%. Under the VDD of 0.1V, energy is 0.1 pJ with 31.5% variation with temperature ranging from -55°C to 125°C.

To increase the resolution of the ADC to meet the requirement of the neuromorphic system, we change the structure of the ADC and add noise shaping into the signal path. We present the noise-shaped SAR ADC in Chapter 3, with the optimized 2nd noise shaping and the higher SS of TFETs, the ENOB is effectively increase and the power consumption is mainly limited by noise performance. ADC achieves ENOB of 11.67 bits and Schreier FOM of 178.7 dB, one of the highest among recent works.

The next circuit that is critical in the neuromorphic system is the artificial neuron that performs integrate-and-fire as the calculation unit in the system. By explore the physical feature of the ReRAM, we design the capacitor-less neuron using ReRAM in chapter 4. Using the 65nm CMOS technology node, the circuit is 14×5 μm^2 in size and consumes 1.28 μW on average power

dissipation. The output spikes follow the Poisson distribution and can be used in stochastic spiking networks. We use the ternary weight for the synapses. Compared to the 1T1R ReRAM memory crossbar, the Neural Array has a smaller area overhead of 0.74% and a power overhead of 13.35%.

Finally, we propose the low power neuromorphic architecture in chapter 5. We propose the process element (PE) to integrate ReRAM based neurons and synapses. We develop a reconfigurable hierarchy that efficiently implements SNNs of different topologies using the address event representation protocol. Simulation results show that our architecture provides significant energy savings between 1222× and 1853× (three orders of magnitude improvement). Our architecture also shows over 3× energy improvement of in-memory computing counterparts. For the speed performance, the benefit is between 791× to 1120× and over two times of the Prime. Hence, our architecture efficiently accelerates both network models.

APPENDIX: COPYRIGHT PERMISSION LETTERS OF IEEE

IEEE Requesting permission to reuse content from an IEEE publication

Analysis and Simulation of Capacitor-Less ReRAM-Based Stochastic Neurons for the in-Memory Spiking Neural Network

Author: Jie Lin

Publication: Biomedical Circuits and Systems, IEEE Transactions on

Publisher: IEEE

Date: Oct. 2018

Thesis / Dissertation Reuse

The IEEE does not require individuals working on a thesis to obtain a formal reuse license, however, you may print out this statement to be used as a permission grant:

Requirements to be followed when using any portion (e.g., figure, graph, table, or textual material) of an IEEE copyrighted paper in a thesis:

1) In the case of textual material (e.g., using short quotes or referring to the work within these papers) users must give full credit to the original source (author, paper, publication) followed by the IEEE copyright line © 2011 IEEE.
2) In the case of illustrations or tabular material, we require that the copyright line © [Year of original publication] IEEE appear prominently with each reprinted figure and/or table.
3) If a substantial portion of the original paper is to be used, and if you are not the senior author, also obtain the senior author's approval.

Requirements to be followed when using an entire IEEE copyrighted paper in a thesis:

1) The following IEEE copyright/ credit notice should be placed prominently in the references: © [year of original publication] IEEE. Reprinted, with permission, from [author names, paper title, IEEE publication title, and month/year of publication]
2) Only the accepted version of an IEEE copyrighted paper can be used when posting the paper or your thesis on-line.
3) In placing the thesis on the author's university website, please display the following message in a prominent place on the website: In reference to IEEE copyrighted material which is used with permission in this thesis, the IEEE does not endorse any of [university/educational entity's name goes here]'s products or services. Internal or personal use of this material is permitted. If interested in reprinting/republishing IEEE copyrighted material for advertising or promotional purposes or for creating new collective works for resale or redistribution, please go to http://www.ieee.org/publications_standards/publications/rights/rights_link.html to learn how to obtain a License from RightsLink.

If applicable, University Microfilms and/or ProQuest Library, or the Archives of Canada may supply single copies of the dissertation.

BACK CLOSE

LIST OF REFERENCES

[1] J. Lin and J.-S. Yuan, "A scalable and reconfigurable in-memory architecture for ternary deep spiking neural network with ReRAM based neurons," *Neurocomputing*, vol. 375, pp. 102 – 112, 2020. [Online]. Available: `http://www.sciencedirect.com/science/article/pii/S0925231219313566`

[2] C. Mead, "Neuromorphic electronic systems," *Proceedings of the IEEE*, vol. 78, no. 10, pp. 1629–1636, Oct 1990.

[3] R. Douglas, M. Mahowald, and C. Mead, "Neuromorphic analogue vlsi," *Annual review of neuroscience*, vol. 18, no. 1, pp. 255–281, 1995.

[4] E. Chicca, F. Stefanini, C. Bartolozzi, and G. Indiveri, "Neuromorphic electronic circuits for building autonomous cognitive systems," *Proceedings of the IEEE*, vol. 102, no. 9, pp. 1367–1388, Sep. 2014.

[5] G. Indiveri and S. Liu, "Memory and information processing in neuromorphic systems," *Proceedings of the IEEE*, vol. 103, no. 8, pp. 1379–1397, Aug 2015.

[6] W. A. Wulf and S. A. McKee, "Hitting the memory wall: implications of the obvious," *ACM SIGARCH computer architecture news*, vol. 23, no. 1, pp. 20–24, 1995.

[7] C. Zhao, B. T. Wysocki, C. D. Thiem, N. R. McDonald, J. Li, L. Liu, and Y. Yi, "Energy efficient spiking temporal encoder design for neuromorphic computing systems," *IEEE Transactions on Multi-Scale Computing Systems*, vol. 2, no. 4, pp. 265–276, Oct. 2016.

[8] G. Indiveri, B. Linares-Barranco, T. J. Hamilton, A. Van Schaik, R. Etienne-Cummings, T. Delbruck, S.-C. Liu, P. Dudek, P. Häfliger, S. Renaud *et al.*, "Neuromorphic silicon neuron circuits," *Frontiers in Neuroscience*, vol. 5, p. 73, 2011.

[9] M. Al-Shedivat, R. Naous, G. Cauwenberghs, and K. N. Salama, "Memristors empower spiking neurons with stochasticity," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 5, no. 2, pp. 242–253, June 2015.

[10] B. Liu, H. Li, Y. Chen, X. Li, T. Huang, Q. Wu, and M. Barnell, "Reduction and IR-drop compensations techniques for reliable neuromorphic computing systems," in *2014 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, Nov 2014, pp. 63–70.

[11] X. Qiao, X. Cao, H. Yang, L. Song, and H. Li, "Atomlayer: A universal reram-based cnn accelerator with atomic layer computation," in *Proceedings of the 55th Annual Design Automation Conference*, ser. DAC '18. New York, NY, USA: ACM, 2018, pp. 103:1–103:6.

[12] P. A. Merolla, J. V. Arthur, R. Alvarez-Icaza, A. S. Cassidy, J. Sawada, F. Akopyan, B. L. Jackson, N. Imam, C. Guo, Y. Nakamura *et al.*, "A million spiking-neuron integrated circuit with a scalable communication network and interface," *Science*, vol. 345, no. 6197, pp. 668–673, 2014.

[13] S. B. Furber, F. Galluppi, S. Temple, and L. A. Plana, "The spinnaker project," *Proceedings of the IEEE*, vol. 102, no. 5, pp. 652–665, 2014.

[14] A. Shafiee, A. Nag, N. Muralimanohar, R. Balasubramonian, J. P. Strachan, M. Hu, R. S. Williams, and V. Srikumar, "ISAAC: A convolutional neural network accelerator with in-situ analog arithmetic in crossbars," *ACM SIGARCH Computer Architecture News*, vol. 44, no. 3, pp. 14–26, 2016.

[15] S. Hsieh and C. Hsieh, "A 0.3v 0.705fj/conversion-step 10-bit sar adc with shifted monotonie switching scheme in 90nm cmos," in *2016 IEEE International Symposium on Circuits and Systems (ISCAS)*, May 2016, pp. 2899–2899.

[16] C. Hsieh and S. Liu, "A 0.3v 10bit 7.3fj/conversion-step sar adc in 0.18μm cmos," in *2014 IEEE Asian Solid-State Circuits Conference (A-SSCC)*, Nov 2014, pp. 325–328.

[17] J. Lin and C. Hsieh, "A 0.3 v 10-bit 1.17 f sar adc with merge and split switching in 90 nm cmos," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 62, no. 1, pp. 70–79, Jan 2015.

[18] S. Hsieh and C. Hsieh, "A 0.44-fj/conversion-step 11-bit 600-ks/s sar adc with semi-resting dac," *IEEE Journal of Solid-State Circuits*, vol. 53, no. 9, pp. 2595–2603, Sep. 2018.

[19] P. Lee, J. Lin, and C. Hsieh, "A 0.4 v 1.94 fj/conversion-step 10 bit 750 ks/s sar adc with input-range-adaptive switching," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 63, no. 12, pp. 2149–2157, Dec 2016.

[20] J. Lin and J. S. Yuan, "A 12-bit ultra-low voltage noise shaping successive-approximation register analogto-digital converter using emerging TFETs," *Journal of Low Power Electronics*, vol. 13, no. 3, pp. 497–510, 2017.

[21] P. O'Connor, D. Neil, S.-C. Liu, T. Delbruck, and M. Pfeiffer, "Real-time classification and sensor fusion with a spiking deep belief network," *Frontiers in Neuroscience*, vol. 7, p. 178, 2013.

[22] T. Tuma, A. Pantazi, M. Le Gallo, A. Sebastian, and E. Eleftheriou, "Stochastic phase-change neurons," *Nature Nanotechnology*, vol. 11, no. 8, pp. 693–699, 2016.

[23] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ser. ICML'10. USA: Omnipress, 2010, pp. 807–814.

[24] W. Maass, "Noise as a resource for computation and learning in networks of spiking neurons," *Proceedings of the IEEE*, vol. 102, no. 5, pp. 860–880, May 2014.

[25] R. Jolivet, A. Rauch, H.-R. Lüscher, and W. Gerstner, "Predicting spike timing of neocortical pyramidal neurons by simple threshold models," *Journal of Computational Neuroscience*, vol. 21, no. 1, pp. 35–49, Aug 2006.

[26] S. Yu, X. Guan, and H. S. P. Wong, "On the stochastic nature of resistive switching in metal oxide RRAM: Physical modeling, Monte Carlo simulation, and experimental characterization," in *2011 International Electron Devices Meeting*, Dec. 2011, pp. 17.3.1–17.3.4.

[27] H. Li, T. F. Wu, A. Rahimi, K.-S. Li, M. Rusch, C.-H. Lin, J.-L. Hsu, M. M. Sabry, S. B. Eryilmaz, J. Sohn *et al.*, "Hyperdimensional computing with 3D VRRAM in-memory kernels: Device-architecture co-design for energy-efficient, error-resilient language recognition," in *2016 IEEE International Electron Devices Meeting (IEDM)*, Dec. 2016, pp. 16.1.1–16.1.4.

[28] S. Yu, B. Gao, Z. Fang, H. Yu, J. Kang, and H.-S. P. Wong, "Stochastic learning in oxide binary synaptic device for neuromorphic computing," *Frontiers in Neuroscience*, vol. 7, p. 186, 2013.

[29] H.-S. P. Wong, H.-Y. Lee, S. Yu, Y.-S. Chen, Y. Wu, P.-S. Chen, B. Lee, F. T. Chen, and M.-J. Tsai, "Metal-oxide RRAM," *Proceedings of the IEEE*, vol. 100, no. 6, pp. 1951–1970, June 2012.

[30] A. Ankit, A. Sengupta, P. Panda, and K. Roy, "Resparc: A reconfigurable and energy-efficient architecture with memristive crossbars for deep spiking neural networks," in *Proceedings of the 54th Annual Design Automation Conference 2017*, ser. DAC '17. New York, NY, USA: ACM, 2017, pp. 27:1–27:6.

[31] S. Li, C. Xu, Q. Zou, J. Zhao, Y. Lu, and Y. Xie, "Pinatubo: A processing-in-memory architecture for bulk bitwise operations in emerging non-volatile memories," in *2016 53nd ACM/EDAC/IEEE Design Automation Conference (DAC)*, June 2016, pp. 1–6.

[32] M.-W. Kwon, S. Kim, M.-H. Kim, J. Park, H. Kim, S. Hwang, and B.-G. Park, "Integrate-and-fire (I&F) neuron circuit using resistive-switching random access memory (RRAM)," *Journal of Nanoscience and Nanotechnology*, vol. 17, no. 5, pp. 3038–3041, Jan. 2017.

[33] G. Palma, M. Suri, D. Querlioz, E. Vianello, and B. De Salvo, "Stochastic neuron design using conductive bridge ram," in *2013 IEEE/ACM International Symposium on Nanoscale Architectures (NANOARCH)*, July 2013, pp. 95–100.

[34] E. Chicca and S. Fusi, "Stochastic synaptic plasticity in deterministic avlsi networks of spiking neurons," in *Proceedings of the World Congress on Neuroinformatics*, 2001.

[35] R. Moreno-Bote, "Poisson-like spiking in circuits with probabilistic synapses," *PLOS Computational Biology*, vol. 10, no. 7, pp. 1–13, 07 2014. [Online]. Available: `https://doi.org/10.1371/journal.pcbi.1003522`

[36] M.-J. Lee, C. B. Lee, D. Lee, S. R. Lee, M. Chang, J. H. Hur, Y.-B. Kim, C.-J. Kim, D. H. Seo, S. Seo *et al.*, "A fast, high-endurance and scalable non-volatile memory device made from asymmetric Ta_2O_{5-x}/TaO_{2-x} bilayer structures," *Nature materials*, vol. 10, no. 8, pp. 625–630, 2011.

[37] P. U. Diehl, D. Neil, J. Binas, M. Cook, S. C. Liu, and M. Pfeiffer, "Fast-classifying, high-accuracy spiking deep networks through weight and threshold balancing," in *2015 International Joint Conference on Neural Networks (IJCNN)*, July 2015, pp. 1–8.

[38] S. W. Keckler, W. J. Dally, B. Khailany, M. Garland, and D. Glasco, "Gpus and the future of parallel computing," *IEEE Micro*, vol. 31, no. 5, pp. 7–17, Sept 2011.

[39] H. Li and Z. Zhang, "Mining the intrinsic trends of CO_2 solubility in blended solutions," *Journal of CO_2 Utilization*, vol. 26, pp. 496–502, 2018.

[40] H. Li, D. Yan, Z. Zhang, and E. Lichtfouse, "Prediction of CO_2 absorption by physical solvents using a chemoinformatics-based machine learning model," *Environmental Chemistry Letters*, pp. 1–8, 2019.

[41] P. Chi, S. Li, C. Xu, T. Zhang, J. Zhao, Y. Liu, Y. Wang, and Y. Xie, "Prime: A novel processing-in-memory architecture for neural network computation in reram-based main memory," in *2016 ACM/IEEE 43rd Annual International Symposium on Computer Architecture (ISCA)*, June 2016, pp. 27–39.

[42] B. C. Lee, E. Ipek, O. Mutlu, and D. Burger, "Architecting phase change memory as a scalable dram alternative," in *Proceedings of the 36th Annual International Symposium on Computer Architecture*, ser. ISCA '09. New York, NY, USA: ACM, 2009, pp. 2–13.

[43] D. Apalkov, A. Khvalkovskiy, S. Watts, V. Nikitin, X. Tang, D. Lottis, K. Moon, X. Luo, E. Chen, A. Ong *et al.*, "Spin-transfer torque magnetic random access memory (stt-mram)," *ACM Journal on Emerging Technologies in Computing Systems (JETC)*, vol. 9, no. 2, p. 13, 2013.

[44] H. Akinaga and H. Shima, "Resistive random access memory (reram) based on metal oxides," *Proceedings of the IEEE*, vol. 98, no. 12, pp. 2237–2251, 2010.

[45] H. Li, K.-S. Li, C.-H. Lin, J.-L. Hsu, W.-C. Chiu, M.-C. Chen, T.-T. Wu, J. Sohn, S. B. Eryilmaz, J.-M. Shieh *et al.*, "Four-layer 3d vertical rram integrated with finfet as a versatile computing unit for brain-inspired cognitive information processing," in *2016 IEEE Symposium on VLSI Technology*, June 2016, pp. 1–2.

[46] W. Huangfu, S. Li, X. Hu, and Y. Xie, "Radar: A 3d-reram based dna alignment accelerator architecture," in *Proceedings of the 55th Annual Design Automation Conference*, ser. DAC '18. New York, NY, USA: ACM, 2018, pp. 59:1–59:6.

[47] T. Tuma, A. Pantazi, M. Le Gallo, A. Sebastian, and E. Eleftheriou, "Stochastic phase-change neurons," *Nature nanotechnology*, vol. 11, no. 8, pp. 693–699, 2016.

[48] J. Lin and J. Yuan, "Analysis and simulation of capacitor-less reram-based stochastic neurons for the in-memory spiking neural network," *IEEE Transactions on Biomedical Circuits and Systems*, pp. 1–14, 2018.

[49] J. Lin and J. Yuan, "Capacitor-less rram-based stochastic neuron for event-based unsupervised learning," in *2017 IEEE Biomedical Circuits and Systems Conference (BioCAS)*, Oct 2017, pp. 1–4.

[50] L. S. Y. Wong, S. Hossain, A. Ta, J. Edvinsson, D. H. Rivas, and H. Naas, "A very low-power cmos mixed-signal ic for implantable pacemaker applications," *IEEE Journal of Solid-State Circuits*, vol. 39, no. 12, pp. 2446–2456, Dec 2004.

[51] R. Gandhi, Z. Chen, N. Singh, K. Banerjee, and S. Lee, "Cmos-compatible vertical-silicon-nanowire gate-all-around p-type tunneling fets with$\leq$ 50-mv/decade subthreshold swing," *IEEE Electron Device Letters*, vol. 32, no. 11, pp. 1504–1506, Nov 2011.

[52] M. S. Kim, H. Liu, X. Li, S. Datta, and V. Narayanan, "A steep-slope tunnel fet based sar analog-to-digital converter," *IEEE Transactions on Electron Devices*, vol. 61, no. 11, pp. 3661–3667, Nov 2014.

[53] H. Lu, J. W. Kim, D. Esseni, and A. Seabaugh, "Continuous semiempirical model for the current-voltage characteristics of tunnel fets," in *2014 15th International Conference on Ultimate Integration on Silicon (ULIS)*, April 2014, pp. 25–28.

[54] O. F. Shoron, S. A. Siddiqui, A. Zubair, and Q. Khosru, "A simple physically based model of temperature effect on drain current for nanoscale tfet," in *2010 IEEE International Conference of Electron Devices and Solid-State Circuits (EDSSC)*. IEEE, 2010, pp. 1–4.

[55] P.-F. Guo, L.-T. Yang, Y. Yang, L. Fan, G.-Q. Han, G. S. Samudra, and Y.-C. Yeo, "Tunneling field-effect transistor: Effect of strain and temperature on tunneling current," *IEEE Electron Device Letters*, vol. 30, no. 9, pp. 981–983, 2009.

[56] W. Zhao and Y. Cao, "Predictive technology model for nano-CMOS design exploration," *ACM Journal on Emerging Technologies in Computing Systems (JETC)*, vol. 3, no. 1, p. 1, 2007.

[57] B. Murmann, "Adc performance survey 1997–2015," 2015. [Online]. Available: `http://web.Stanford.edu/~murmann/adcsurvey.html`

[58] L. S. Y. Wong, S. Hossain, A. Ta, J. Edvinsson, D. H. Rivas, and H. Naas, "A very low-power cmos mixed-signal ic for implantable pacemaker applications," *IEEE Journal of Solid-State Circuits*, vol. 39, no. 12, pp. 2446–2456, Dec 2004.

[59] R. R. Harrison, P. T. Watkins, R. J. Kier, R. O. Lovejoy, D. J. Black, B. Greger, and F. Solzbacher, "A low-power integrated circuit for a wireless 100-electrode neural recording system," *IEEE Journal of Solid-State Circuits*, vol. 42, no. 1, pp. 123–133, Jan 2007.

[60] Z. Chen, M. Miyahara, and A. Matsuzawa, "A 9.35-enob, 14.8 fj/conv.-step fully-passive noise-shaping sar adc," in *2015 Symposium on VLSI Circuits (VLSI Circuits)*, June 2015, pp. C64–C65.

[61] W. Guo and N. Sun, "A 12b-enob 61μW noise-shaping sar adc with a passive integrator," in *ESSCIRC Conference 2016: 42nd European Solid-State Circuits Conference*, Sep. 2016, pp. 405–408.

[62] J. A. Fredenburg and M. P. Flynn, "A 90-ms/s 11-mhz-bandwidth 62-db sndr noise-shaping sar adc," *IEEE Journal of Solid-State Circuits*, vol. 47, no. 12, pp. 2898–2904, Dec 2012.

[63] H. Lu, D. Esseni, and A. Seabaugh, “Universal analytic model for tunnel fet circuit simulation,” *Solid-State Electronics*, vol. 108, pp. 110–117, 2015.

[64] J. Lin and J. S. Yuan, “Ultra-low power successive approximation analog-to-digital converter using emerging tunnel field effect transistor technology,” *Journal of Low Power Electronics*, vol. 12, no. 3, pp. 218–226, 2016.

[65] A. Marques, V. Peluso, M. Steyaert, and W. Sansen, “Analysis of the trade-off between bandwidth, resolution, and power in /spl delta//spl sigma/ analog to digital converters,” in *1998 IEEE International Conference on Electronics, Circuits and Systems. Surfing the Waves of Science and Technology (Cat. No.98EX196)*, vol. 2, Sep. 1998, pp. 153–156 vol.2.

[66] S. I. Mann and D. P. Taylor, “Limit cycle behavior in the double-loop bandpass /spl sigma/-/spl delta/ a/d converter,” *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 46, no. 8, pp. 1086–1089, Aug 1999.

[67] D. Reefman, J. Reiss, E. Janssen, and M. Sandler, “Description of limit cycles in sigma-delta modulators,” *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 52, no. 6, pp. 1211–1223, June 2005.

[68] L. He, L. Jin, J. Yang, F. Lin, L. Yao, and X. Jiang, “Self-dithering technique for high-resolution sar adc design,” *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 62, no. 12, pp. 1124–1128, Dec 2015.

[69] C. Liu, S. Chang, G. Huang, and Y. Lin, “A 10-bit 50-ms/s sar adc with a monotonic capacitor switching procedure,” *IEEE Journal of Solid-State Circuits*, vol. 45, no. 4, pp. 731–740, April 2010.

[70] R. Pandey, S. Mookerjea, and S. Datta, "Opportunities and challenges of tunnel fets," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 63, no. 12, pp. 2128–2138, Dec 2016.

[71] M. Hu, J. P. Strachan, Z. Li, E. M. Grafals, N. Davila, C. Graves, S. Lam, N. Ge, J. J. Yang, and R. S. Williams, "Dot-product engine for neuromorphic computing: Programming 1t1m crossbar to accelerate matrix-vector multiplication," in *2016 53nd ACM/EDAC/IEEE Design Automation Conference (DAC)*, June 2016, pp. 1–6.

[72] J. Lin and J. Yuan, "Ultra-low power successive approximation analog-to-digital converter using emerging tunnel field effect transistor technology," *Journal of Low Power Electronics*, vol. 12, no. 3, pp. 218–226, 2016.

[73] ——, "A 300 mv, 6-bit ultra-low power sar adc," in *Solid-State and Integrated Circuit Technology (ICSICT), 2016 13th IEEE International Conference on*. IEEE, 2016, pp. 713–715.

[74] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[75] Z. Jiang, S. Yu, Y. Wu, J. H. Engel, X. Guan, and H.-S. P. Wong, "Verilog-A compact model for oxide-based resistive random access memory (RRAM)," in *2014 International Conference on Simulation of Semiconductor Processes and Devices (SISPAD)*, Sept. 2014, pp. 41–44.

[76] L. F. Abbott and C. van Vreeswijk, "Asynchronous states in networks of pulse-coupled oscillators," *Phys. Rev. E*, vol. 48, pp. 1483–1490, Aug 1993.

[77] J. McPherson, J.-Y. Kim, A. Shanware, and H. Mogul, "Thermochemical description of dielectric breakdown in high dielectric constant materials," *Applied Physics Letters*, vol. 82, no. 13, pp. 2121–2123, 2003.

[78] Y. Cao, Y. Chen, and D. Khosla, "Spiking deep convolutional neural networks for energy-efficient object recognition," *International Journal of Computer Vision*, vol. 113, no. 1, pp. 54–66, May 2015.

[79] A. Sengupta, Y. Ye, R. Wang, C. Liu, and K. Roy, "Going deeper in spiking neural networks: Vgg and residual architectures," *arXiv preprint arXiv:1802.02627*, 2018.

[80] B. Rueckauer, I.-A. Lungu, Y. Hu, M. Pfeiffer, and S.-C. Liu, "Conversion of continuous-valued deep networks to efficient event-driven networks for image classification," *Frontiers in Neuroscience*, vol. 11, p. 682, 2017. [Online]. Available: `https://www.frontiersin.org/article/10.3389/fnins.2017.00682`

[81] C. Posch, D. Matolin, and R. Wohlgenannt, "A QVGA 143 db dynamic range frame-free PWM image sensor with lossless pixel-level video compression and time-domain CDS," *IEEE Journal of Solid-State Circuits*, vol. 46, no. 1, pp. 259–275, Jan 2011.

[82] Z. Fang, H. Y. Yu, W. J. Liu, Z. R. Wang, X. A. Tran, B. Gao, and J. F. Kang, "Temperature instability of resistive switching on HfO_x-based rram devices," *IEEE Electron Device Letters*, vol. 31, no. 5, pp. 476–478, May 2010.

[83] P. Pouyan, E. Amat, and A. Rubio, "Reliability challenges in design of memristive memories," in *2014 5th European Workshop on CMOS Variability (VARI)*, Sept 2014, pp. 1–6.

[84] R. Naous, M. Al-Shedivat, and K. N. Salama, "Stochasticity modeling in memristors," *IEEE Transactions on Nanotechnology*, vol. 15, no. 1, pp. 15–28, Jan 2016.

[85] K. Roy, S. Mukhopadhyay, and H. Mahmoodi-Meimand, "Leakage current mechanisms and leakage reduction techniques in deep-submicrometer cmos circuits," *Proceedings of the IEEE*, vol. 91, no. 2, pp. 305–327, Feb. 2003.

[86] X. Xi, M. Dunga, J. He, W. Liu, K. M. Cao, X. Jin, J. J. Ou, M. Chan, A. M. Niknejad, C. Hu *et al.*, "Bsim4. 3.0 mosfet model," *UC Berkley*, 2003.

[87] S. H. Jo, K.-H. Kim, and W. Lu, "Programmable resistance switching in nanoscale two-terminal devices," *Nano Letters*, vol. 9, no. 1, pp. 496–500, 2009, PMID: 19113891.

[88] G. Chechik, I. Meilijson, and E. Ruppin, "Neuronal regulation: A mechanism for synaptic pruning during brain maturation," *Neural Computation*, vol. 11, no. 8, pp. 2061–2080, 1999.

[89] J. Cui and Q. Qiu, "Towards memristor based accelerator for sparse matrix vector multiplication," in *2016 IEEE International Symposium on Circuits and Systems (ISCAS)*, May 2016, pp. 121–124.

[90] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Comput.*, vol. 18, no. 7, pp. 1527–1554, Jul. 2006.

[91] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, "Greedy layer-wise training of deep networks," in *Proceedings of the 19th International Conference on Neural Information Processing Systems*, ser. NIPS'06. Cambridge, MA, USA: MIT Press, 2006, pp. 153–160.

[92] K. A. Boahen, "Point-to-point connectivity between neuromorphic chips using address events," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 47, no. 5, pp. 416–434, May 2000.

[93] J. Park, T. Yu, S. Joshi, C. Maier, and G. Cauwenberghs, "Hierarchical address event routing for reconfigurable large-scale neuromorphic systems," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 10, pp. 2408–2422, Oct 2017.

[94] B. Rueckauer, I.-A. Lungu, Y. Hu, M. Pfeiffer, and S.-C. Liu, "Conversion of continuous-valued deep networks to efficient event-driven networks for image classification," *Frontiers in Neuroscience*, vol. 11, p. 682, 2017.

[95] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[96] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "Squeezenet: Alexnet-level accuracy with 50x fewer parameters and¡ 0.5 mb model size," `https://arxiv.org/abs/1602.07360`, 2016, [Online; accessed 1-July-2018].

[97] Y. Bengio, N. Léonard, and A. Courville, "Estimating or propagating gradients through stochastic neurons for conditional computation," `https://arxiv.org/abs/1308.3432`, 2013, [Online; accessed 1-July-2018].

[98] F. Alibart, L. Gao, B. D. Hoskins, and D. B. Strukov, "High precision tuning of state for memristive devices by adaptable variation-tolerant algorithm," *Nanotechnology*, vol. 23, no. 7, p. 075201, 2012.

[99] A. Zhou, A. Yao, Y. Guo, L. Xu, and Y. Chen, "Incremental network quantization: Towards lossless cnns with low-precision weights," `https://arxiv.org/abs/1702.03044`, 2017, [Online; accessed 4-July-2018].

[100] C. Zhu, S. Han, H. Mao, and W. J. Dally, "Trained ternary quantization," `https://arxiv.org/abs/1612.01064`, 2016, [Online; accessed 8-July-2018].

[101] B. Li, L. Xia, P. Gu, Y. Wang, and H. Yang, "Merging the interface: Power, area and accuracy co-optimization for rram crossbar-based mixed-signal computing system," in *2015 52nd ACM/EDAC/IEEE Design Automation Conference (DAC)*, June 2015, pp. 1–6.

[102] A. Kawahara, R. Azuma, Y. Ikeda, K. Kawai, Y. Katoh, Y. Hayakawa, K. Tsuji, S. Yoneda, A. Himeno, K. Shimakawa *et al.*, "An 8 Mb multi-layered cross-point reram macro with 443

MB/s write throughput," *IEEE Journal of Solid-State Circuits*, vol. 48, no. 1, pp. 178–185, Jan 2013.

[103] F. Akopyan, J. Sawada, A. Cassidy, R. Alvarez-Icaza, J. Arthur, P. Merolla, N. Imam, Y. Nakamura, P. Datta, G. J. Nam, B. Taba, M. Beakes, B. Brezzo, J. B. Kuang, R. Manohar, W. P. Risk, B. Jackson, and D. S. Modha, "Truenorth: Design and tool flow of a 65 mw 1 million neuron programmable neurosynaptic chip," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 34, no. 10, pp. 1537–1557, Oct 2015.

[104] P. U. Diehl and M. Cook, "Unsupervised learning of digit recognition using spike-timing-dependent plasticity," *Frontiers in computational neuroscience*, vol. 9, p. 99, 2015.

[105] S. Han, J. Pool, J. Tran, and W. J. Dally, "Learning both weights and connections for efficient neural networks," in *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, ser. NIPS'15. Cambridge, MA, USA: MIT Press, 2015, pp. 1135–1143.

[106] G. Orchard, C. Meyer, R. Etienne-Cummings, C. Posch, N. Thakor, and R. Benosman, "Hfirst: a temporal approach to object recognition," *IEEE transactions on pattern analysis and machine intelligence*, vol. 37, no. 10, pp. 2028–2040, 2015.

[107] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proceedings of the 32Nd International Conference on International Conference on Machine Learning - Volume 37*, ser. ICML'15. JMLR.org, 2015, pp. 448–456. [Online]. Available: `http://dl.acm.org/citation.cfm?id=3045118.3045167`

[108] Y. Lecun *et al.*, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov 1998.

[109] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," Citeseer, Tech. Rep., 2009.

[110] A. Paszke *et al.*, "Automatic differentiation in pytorch," in *NIPS-W*, 2017, pp. 1–4.

[111] X. Dong, C. Xu, Y. Xie, and N. P. Jouppi, "Nvsim: A circuit-level performance, energy, and area model for emerging nonvolatile memory," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 31, no. 7, pp. 994–1007, July 2012.

[112] N. Binkert, B. Beckmann, G. Black, S. K. Reinhardt, A. Saidi, A. Basu, J. Hestness, D. R. Hower, T. Krishna, S. Sardashti *et al.*, "The gem5 simulator," *ACM SIGARCH Computer Architecture News*, vol. 39, no. 2, pp. 1–7, 2011.