Chapter

Spiking Neural Encoding and Hardware Implementations for Neuromorphic Computing

Honghao Zheng and Yang Cindy Yi

Abstract

Due to the high requirements of the computational power of modern data-intensive applications, the traditional von Neumann structure and neuromorphic computing structure started to play complementary roles in the area of computing. Thus, neuromorphic computing architectures have attracted much attention with high data capacity and power efficiency. In this chapter, the basic concept of neuromorphic computing is discussed, including spiking codes and neurons. The spiking encoder can transfer analog signals to spike signals, thus avoiding using power-consuming analog-to-digital converters. Comparisons of training accuracy and robustness of neural codes are carried out, and the circuit implementations of the spiking temporal encoders are briefly introduced. The encoding schemes are evaluated on the PyTorch platform with the most common datasets, such as Modified National Institute of Standards and Technology (MNIST), Canadian Institute for Advanced Research, 10 classes (CIFAR-10), and The Street View House Numbers (SVHN). From the result, the multiplexing temporal code has shown high data capacity, robustness, and low training error. It achieves at least 6.4% more accuracy than other state-of-the-art works using other encoding schemes.

Keywords: analog/mixed-signal integrated circuit (IC) design, neuromorphic computing, neural spike encoding, multiplexing temporal encoder, gamma alignment

1. Introduction

Since the late 1980s, researchers have been paying attention to neuromorphic computing [1]. Researchers have noticed that by mimicking the biological neural systems with either software or hardware implementations, computing systems' data capacity and power efficiency can be greatly improved. With the limitations of conventional von Neumann structures appealing to the progress of application requirements, the self-training mechanism of neuromorphic computing structures has attracted more and more attention, both from the industrial and academic areas. For instance, neuromorphic computing systems can more efficiently process data-intensive tasks like speech processing and image recognition [2–4] due to the nature of parallel computing. Besides that, the comparison between the conventional structure and neuromorphic chips has shown that the power consumed by neuromorphic computing systems is also lower than traditional von Neumann structures and it is achieved by the parallelism and distributed processing as well as event-driven processing nature of neuromorphic computing. For example, the IBM TrueNorth chip is very power efficient when processing recognition applications [5]. It only consumes less than 3 watts of power. Compared with conventional central processing units (CPUs) or graphics processing units (GPUs), it has shown great improvements. Although GPU has more small specialized cores than CPU, it still consumes tens or even hundreds of watts for the same tasks. Therefore, neuromorphic computing systems, especially those realized by application-specific integrated circuits (ASICs), have demonstrated their superiority in both learning capability and power efficiency aspects.

Among all the types of artificial neural networks (ANNs), one special type is the so-called spiking neural network (SNN) [6–9]. Inspired by the signal transformation in biological neural networks, researchers realized that information could be transmitted in the form of spikes in neural systems [10]. A neuron's function is to receive stimulus and output impulses. It is formed of four main parts, dendrites, soma, axons, and synapses. A dendrite receives the stimulus and transmits it to soma. Soma is the central computing unit in the biological neural network and generates an output signal when the input exceeds a threshold voltage. Each neuron has a specific threshold voltage, and when the output signal is generated, the soma fires a spike to the axon, which transmits the output signal to the synapse. Finally, the impulse will be conveyed through the synapse to the subsequent neurons. With such property, neurons in the network can stay silent unless triggered by coming spikes. In this way, the operating power consumption can be greatly saved. To convert information to the form of spikes, several encoding schemes have been investigated [11, 12]. This research started a few decades ago. These encoding schemes can be categorized into two main types, rate encoding and temporal encoding [13]. The rate encoding scheme represents that information is converted to the number of spikes in one spike train [14]. The spike rate in this encoding scheme means the number of spikes in one encoding window. The larger the input is, the higher the spike rate will be in the corresponding encoding window. This encoding scheme is very straightforward to implement and is one of the most commonly used encoding schemes. On the other hand, temporal encoding takes account of the number of spikes in the encoding window and utilizes the temporal property of the spike train [15]. With the different temporal properties used, the temporal encoding scheme can be more deeply divided into several types, like the Time-to-First-Spike (TTFS) [16], the Interspike Interval (ISI) [17], and the Phase of firing encoding [18].

With the advancement of neuroscience, researchers noticed a special encoding scheme in biological neural systems. This encoding scheme can integrate multiple encoding schemes that operate in different time scales [19]. This kind of encoding scheme is called multiplexing encoding [20]. For instance, ISI encoding can be integrated with phase encoding, forming the multiplexing ISI-phase encoding. Compared with just one encoding scheme, multiplexing encoding has various advantages, including high data capacity and high robustness, especially in noisy environments. The advantages and disadvantages of these encoding schemes are summarized in **Table 1**. For example, rate encoding is not only easier to implement than other schemes but it also has lower data capacity. Temporal encoding schemes not only have higher data capacity than rate encoding but also have lower robustness compared with multiplexing encoding.

Encoding scheme	Rate	Temporal	Multiplexing
Advantage	Straightforward Easy to implement	Higher data capacity	Highest data capacity High robustness
Disadvantage	Low data capacity	Low robustness against noise	High complexity High power and area cost

Table 1.
Advantages and disadvantages of different encoding schemes.

A literature review of those encoding schemes is also carried out. This literature review provides a concise overview of key studies and advancements in the area of spike neural encoding.

In Rolls and Treves [21], the authors have carried out a quantitative analysis of information in the realm of neural encoding. The researchers noticed the existence of the firing rate encoding in the short time window. In the aspect of quantitative analysis, more information is encoded by the rate encoding scheme rather than temporal encoding. In rate code, neurons have been found to be able to take synaptic weighted sums of the inputs for training purposes.

In Auge et al. [22], the authors have summarized the theoretical foundation as well as the applications of encoding schemes. It includes both the rate encoding and the temporal encoding schemes. They concluded that the rate encoding has high robustness since it does not rely on the precise firing timing of spikes. They also noticed that temporal codes have been shown to have higher information capacity, faster reaction times, and higher transmission speeds.

In Kayser et al. [23], researchers have verified the hypothesis that different codes might be employed concurrently and provide complementary stimulus information. They also quantified the information encoded in the auditory cortex of animals and found that multiplexing those codes together will provide a much higher information level. What's more, the authors also found that the multiplexing codes with phase of firing code are very much robust to sensory noise added to the stimulus.

In this chapter, a deeper discussion of these mentioned encoding schemes will be presented in Section 2. Section 3 will discuss the ASIC implementations of these encoding schemes and their simulation results. Lastly, the training results of these encoding schemes working with some common datasets and the hardware testbench of the multiplexing temporal encoder will both be illustrated in Section 4.

2. Neural encoding schemes

As mentioned in Section 1, the neural encoding schemes represent the different ways the input signals get converted to spike signals in spiking neural networks. Researchers have put much effort into finding various encoding schemes that utilize different properties of spike trains in the SNN [24]. The most straightforward and, thus, the first discovered encoding scheme is rate encoding [21]. It uses the number of spikes in one spike train to typify the input information. **Figure 1(a)** shows that the input stimulus is transferred to the firing rate during the sampling window. Therefore, as long as the numbers of spikes are the same, two different spike trains still stand for the same input signal. The simplicity of this code leads to its common use in nowadays applications. For instance, the Intel Loihi chip, which is a neuromorphic

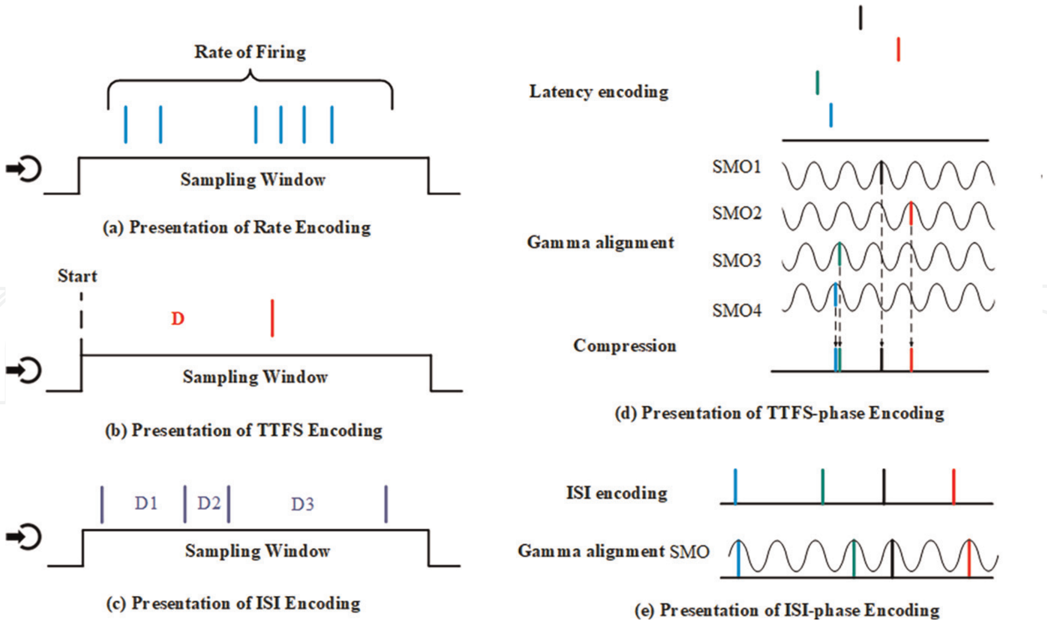

Rate of Firing
Sampling Window
(a) Presentation of Rate Encoding
Start
D
Sampling Window
(b) Presentation of TTFS Encoding
D1
D2
D3
Sampling Window
(c) Presentation of ISI Encoding
Latency encoding
SMO1
SMO2
SMO3
SMO4
Gamma alignment
Compression
(d) Presentation of TTFS-phase Encoding
ISI encoding
Gamma alignment SMO
(e) Presentation of ISI-phase Encoding

Figure 1.
Examples of encoding scheme. (a) Presentation of rate encoding. Input stimulus is transferred to the firing rate in the encoding window. (b) Presentation of TTFS encoding. Input stimulus is converted to the time difference between the onset of the window and the first spike. (c) Presentation of ISI encoding. Input information is transferred to the time intervals of spikes.

research test chip designed by Intel Labs, uses an asynchronous SNN to implement adaptive self-modifying event-driven fine-grained parallel computations used to implement learning and inference with high efficiency. It utilizes rate encoding in its neural network and has been evaluated through lots of applications such as adaptive robot arm control and drone motor control [25, 26]. It only consumes less than 1 watt of power while maintaining a good operating speed. Moreover, the Tianjic chip also implements rate encoders in the neural network and achieves high accuracy for pattern recognition applications [27, 28]. However, this simple encoding scheme has its disadvantages. First, rate encoding has a relatively lower data capacity than other encoding schemes since it only utilizes the number of spikes in a spike train but ignores the temporal patterns of the spike train. Second, the low data capacity also caused the low robustness of the encoding scheme against noises and errors. Since one spike train only represents one input information, any mistakes of the spike train will lead to an inaccurate result.

To overcome these drawbacks, other encoding schemes that can use other properties of spike trains are proposed for the spike encoding process. Temporal patterns, which mean the different timings of spikes in the spike train, are the most used aspects for encoding [29]. Therefore, a large category of neural code is called temporal encoding, which employs both the spike number and the temporal pattern of the spike train for stimulus. Among these temporal codes, three are the most widely used, the TTFS code, the ISI code, and the phase of firing code.

The time of the first spike encoding, also known as latency encoding, is the most basic temporal encoding scheme [22, 30]. Just as literal, the TTFS encoding converts the input information to the time difference between the onset of the sampling window and the first spike. Since the only useful spike is the first one, for energy efficiency, normally there is only one spike in the spike train for TTFS encoding, as demonstrated in **Figure 1(b)**. Since the onset of sampling windows is often defined by external references, the precision of the encoding process is very much dependent on the accuracy of external signals. Any variation in the external source could affect the performance of the encoder [31]. Another shortcoming of the TTFS encoding scheme is that its robustness is low. With the property that only one spike is effective, even only one mistake in the TTFS-encoded spike train could cause enormous errors in the final output of the encoding process. Thus, the function of the TTFS encoder is not robust against even minor noise or error.

Another neural code is proposed to overcome the disadvantages of the TTFS code, the ISI code. As demonstrated in **Figure 1(c)**, instead of being converted to the time difference between the onset and the first spike, the input stimuli are converted to time intervals of spikes [15]. Unlike latency encoding, ISI encoding utilizes the spikes as the internal reference frame for each other, thus avoiding the dependence on the external references. As discussed in the previous paragraph, one main drawback of latency encoding is its relatively low data capacity. However, since the ISI code has multiple spikes in one sampling window, it can transfer more information than latency encoding. There are two types of ISI encoders, the parallel structure and the iteration structure, introduced in Zhao et al. [30]. The parallel encoder, the simpler type, could convey information faster but maintains fewer spikes in one encoding window. On the contrary, the iteration encoder generates more spikes in the sampling window but also takes more time. Both structures' spike numbers relate to the number of neurons in the encoder. The parallel structure has this relation:

$$N_S = N, \tag{1}$$

where N and N_S are the number of spikes in one sampling window and the number of neurons in the encoder. The iteration structure has an exponential relation:

$$N_S = 2^{N-1}. \tag{2}$$

From Eqs. 1 and 2, we can notice that the iteration structure has more spikes when the encoder has more than two neurons. Thus, when looking for a high data capacity and high robustness, the iteration encoder is a promising candidate [17].

Besides relying on the number of spikes and intervals between spikes, information can also be conveyed as relative position on internal reference frames. The internal reference frames are called the subthreshold membrane oscillation (SMO). The SMO can replace the external reference frame and thus overcome the precision issue. What's more, with the help of SMO, the phase of firing encoding scheme can be implemented in neuromorphic computing systems [32–34]. In the phase encoding, input signals are transferred to the phase of SMO. When the SMO arrives at this phase, one spike will be fired. The math model of the SMO can be written as follows:

$$SMO_i = A \cdot \cos(\omega T + \phi_i), \tag{3}$$

where A, ω, and ϕ present the amplitude of the SMO, the angular velocity of the signal, and the start phase of the sine SMO signal, respectively.

To further improve the performance of neuromorphic computing systems, another encoding mechanism has been investigated by researchers. First found in biological neural systems, the multiplexing encoding schemes combine multiple neural codes, especially the ones with different time scales, to have higher data capacity [23]. Each independent encoding scheme can carry a certain amount of information. For instance, in ISI-phase encoding, the ISI encoding scheme and phase of firing encoding

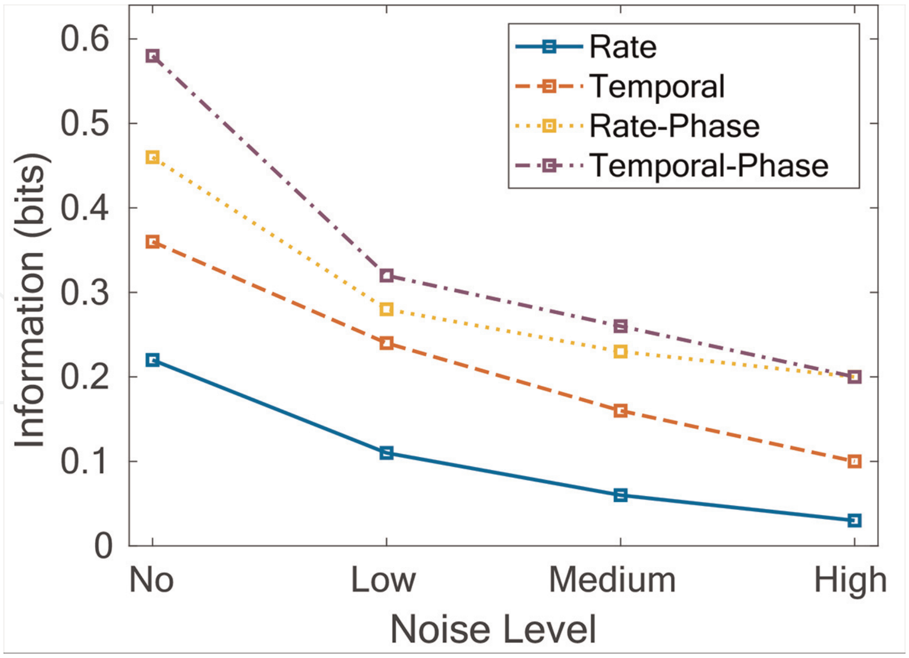


Figure 2.
Information in codes for different noise levels. The blue line indicates the information carried in the rate code with different noise levels. The orange line represents the information level of the temporal encoding scheme. Similarly, the yellow line means the information in the rate and phase of firing multiplexed encoding scheme, and the purple line demonstrates the information carried in the temporal and phase of firing multiplexed code.

scheme carry their own information. After the multiplexing process, the two parts of information are combined and transferred within one sampling time window. Therefore, with multiplexing encoding, the same amount of information can be conveyed within a shorter sampling window and thus increase the data transmission rate [35].

The multiplexing encoding schemes are more robust than the other ones. Experiments have been conducted to quantify the data density in the different neural codes with different levels of input sensory noise [23], as shown in **Figure 2**. From the figure, it is noticeable that although the information from all the encoding schemes decreases with the increase in noise level, the multiplexing encoding schemes always keep the highest data density. Moreover, we can also notice that temporal encoding keeps more information than rate encoding; thus, temporal-phase encoding also has higher data capacity than rate-phase encoding. With the result discussed above, it is proved that the multiplexing encoding scheme is more robust against noisy environments and the high data capacity in all noise levels helps with data transferring from noisy input to spike signals.

The multiplexing encoding requires two separate steps to transfer the input signals to multiplexing encoded spikes [36]. The first one is the encoding process that transferred the analog inputs to differently encoded spikes. For example, if rate encoding is utilized, the encoded spikes are in the form of spike trains with different numbers of spikes. For TTFS encoding, the encoded spike is normally in the form of a single spike. As for the ISI code, the spikes are spike trains with the same number of spikes but different temporal patterns.

After the encoding process, the spikes need to be shifted to meet the phase encoding mechanism [37, 38]. This step is called the gamma alignment step. In this step, the already generated spikes are shifted to the next immediate local maximum of SMOs. The relationship between the original spikes and the shifted spikes can be expressed as

$$P'_{\tau} = P_t, \tag{4}$$

where t is the timing of the original spike and τ is the next immediate local maximum.

As depicted in **Figure 1(d)**, the TTFS-encoded spikes are processed by the gamma alignment step and become the TTFS-phase encoded spikes. In this figure, the spikes are divided into four different channels. Each channel has its corresponding SMO. The SMOs have the same amplitude and angular velocity, and their phases follow this relationship:

$$\phi_i = \phi_0 + (i-1)\frac{2\pi}{N}, \tag{5}$$

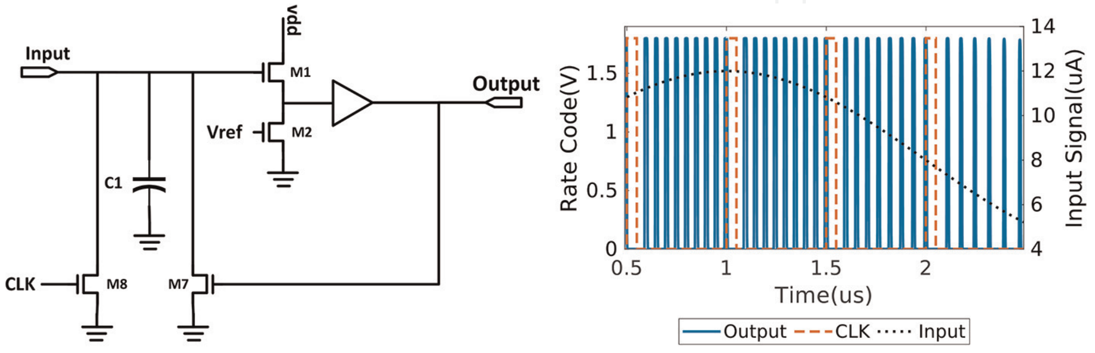
where i means the ith channel and N represents the total number of channels. As for the ISI-phase encoding scheme, since only one channel in **Figure 1(e)** exists, the ISI-coded spikes are shifted to their immediately following local maximums of the same SMO [39].

3. Circuit implementations of the neural encoders

To utilize the various encoding schemes in the application-specific integrated circuit (ASIC) of neural network systems, the circuit implementations of these neural encoders need to be investigated. This section will discuss the circuit implementations of these different encoders and their simulation results with analog sinusoidal current input.

3.1 Rate encoder

The schematic of the rate encoder is demonstrated in **Figure 3(a)**. After the clock signal CLK resets the voltage across the membrane capacitor C1 with the switch transistor M8, an encoding window begins. The voltage across the membrane capacitor C1 increases when the input current is injected. When the membrane voltage exceeds the reference voltage Vref, a spike will be fired through the buffer. The fired spike will also trigger the switch transistor M7 to bring the membrane voltage back to



(a) Circuit schematic of rate encoder

(b) Simulation result of rate encoder

Figure 3.
Circuit schematic and simulation result of rate encoder.

the ground so that the integration process will start over. Thus, the relation between the input current and the spike numbers can be written as:

$$N = \frac{P}{T} = \frac{P}{\frac{C_m V_{ref}}{I_{in}}} = \frac{P I_{in}}{C_m V_{ref}}, \tag{6}$$

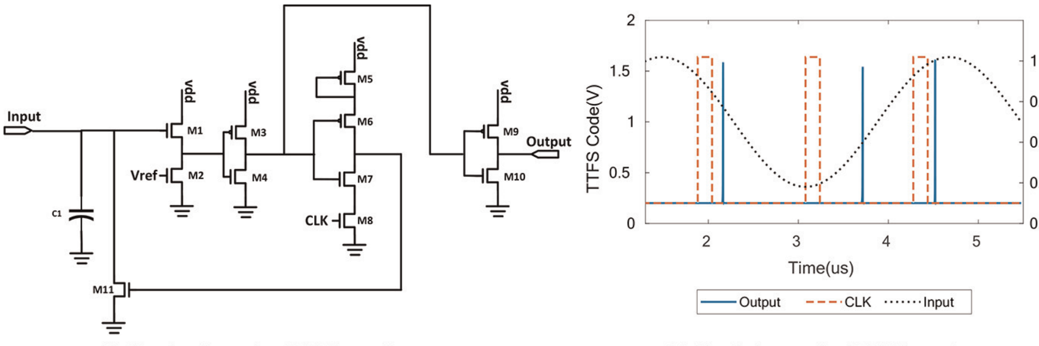
where N, P, and T represent the spike number, encoding window period, and integration time of one spike. I_{in}, C_m, and V_{ref} mean the input current, membrane capacitance, and the reference voltage. The formula shows that the number of spikes in the sampling window and the input current have a linear relationship. From **Figure 3(b)**, a similar relationship can be observed. When the input current is high, there are more spikes in the sampling window; when the input current is relatively smaller, there are fewer spikes in the sampling window.

3.2 TTFS encoder

The schematic of the TTFS encoder is depicted in **Figure 4(a)** [40]. The charge integration mechanism starts after the CLK signal resets the membrane voltage with switch transistor M11. Along with the voltage across the membrane capacitor C1 increasing, the voltage at the source of the transistor M1 will increase at a rate controlled by Vref. When the source voltage of M1 exceeds the threshold voltage of the inverter composing M3 and M4, the output will be digitally high. Almost immediately after the output becomes digital high, the four-transistor clock-controlled inverter also gives a digital high feedback signal to the switch M11 so that the membrane voltage goes back to the ground. Thus, at the output of the encoder, there will only be a spike instead of a square wave at digital high. Moreover, until the next CLK signal, the feedback signal will always be at the high voltage, so there will only be one spike in one sampling window. The time difference between the onset and the spike can be written as:

$$T = \frac{C_m V_{ref}}{I_{in}}. \tag{7}$$

Figure 4(b) shows that the time difference is inversely proportional to the input current. The larger the input is, the closer the spike is to the CLK signal.



(a) Circuit schematic of TTFS encoder

(b) Simulation result of TTFS encoder

Figure 4.
Circuit schematic and simulation result of TTFS encoder.

3.3 ISI encoder

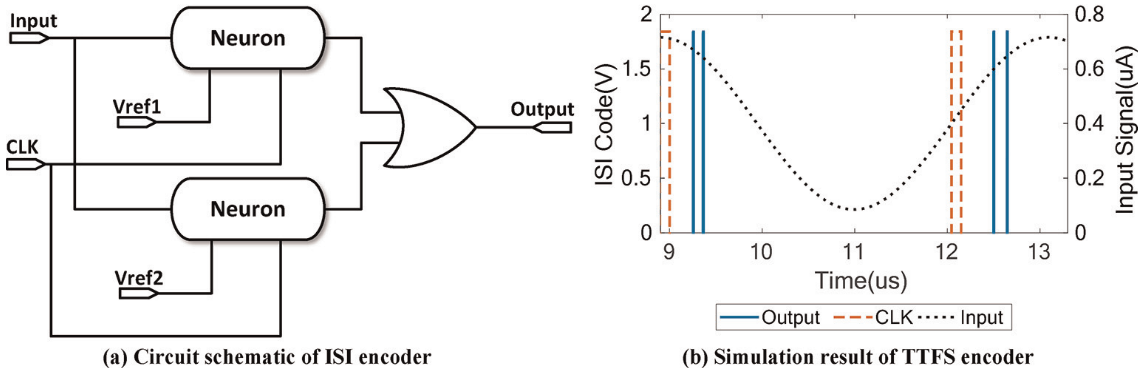
As mentioned in Section 2, the ISI encoder has two different structures. In this section, we will talk in detail about the parallel structure. Although more neurons in the ISI encoder lead to more spikes in one encoding window, it also leads to higher power consumption and a larger design area. Thus, the two-neuron parallel structure ISI encoder circuit will be discussed in this section. The schematic of the encoder is demonstrated in **Figure 5(a)**. The two neurons utilize the same CLK signal and have the same encoding window. The input currents of the two neurons are the same, so the charge integration rates of the neurons are the same. The only difference between the neurons is that they have different reference voltages so that the neurons will fire spike at different times. Afterward, an OR gate is implemented to integrate the two spikes into a two-spike train. With that, the input information is converted to the time intervals of spikes, which can be expressed as:

$$D = T_2 - T_1 = \frac{C_m\left(V_{ref2} - V_{ref1}\right)}{I_{in}}. \tag{8}$$

The simulation result of the ISI encoder is illustrated in **Figure 5(b)**. It is noticeable that when the input is smaller, the time interval of the spike is larger and vice versa. Thus, this encoder has fulfilled the mathematical relation of the information conversion.

3.4 TTFS-phase encoder

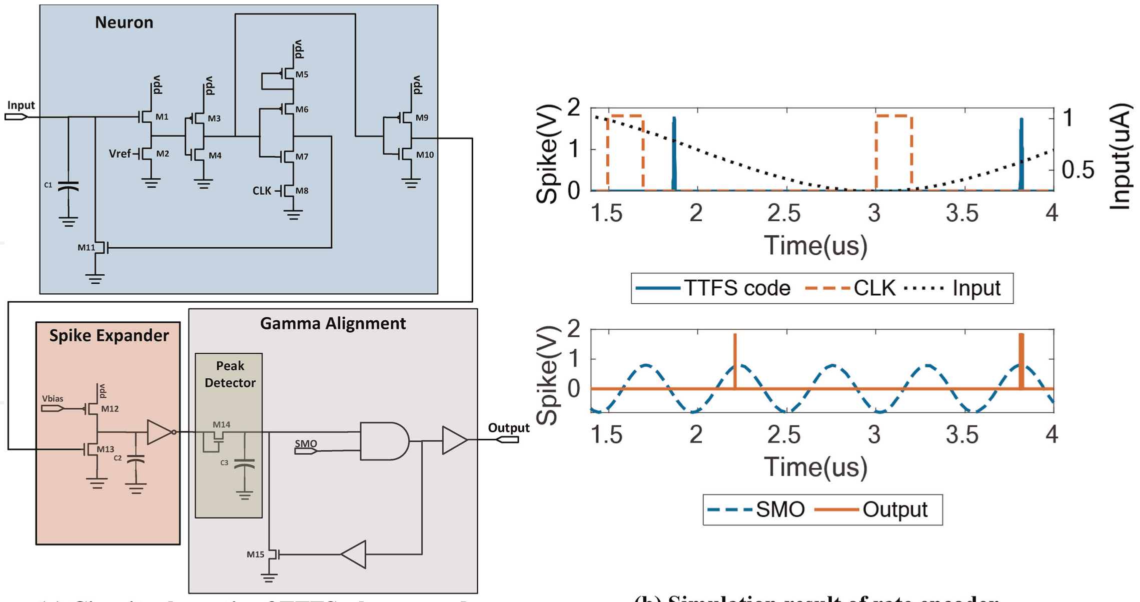
As discussed in Section 2, the TTFS-phase encoding scheme shifts the TTFS-encoded spikes to the immediate local maximum of their corresponding SMOs. Since there is only one channel in our design, as shown in **Figure 6(a)**, the TTFS-phase encoder utilizes only one SMO [37]. To carry out the spike-shifting process, a gamma alignment block is implemented. Inside the gamma alignment block, a peak detector captures and holds the coming spike. The spike voltage will be held across the capacitor with a diode-connected transistor. After that, when the local maximum of the SMO arrives, a spike will be fired by the AND gate and outputted after being stabilized by a buffer. Meanwhile, the spike will trigger the switch transistor and bring the captured voltage back to the ground until the next spike comes.



(a) Circuit schematic of ISI encoder

(b) Simulation result of TTFS encoder

Figure 5.
Circuit schematic and simulation result of ISI encoder.

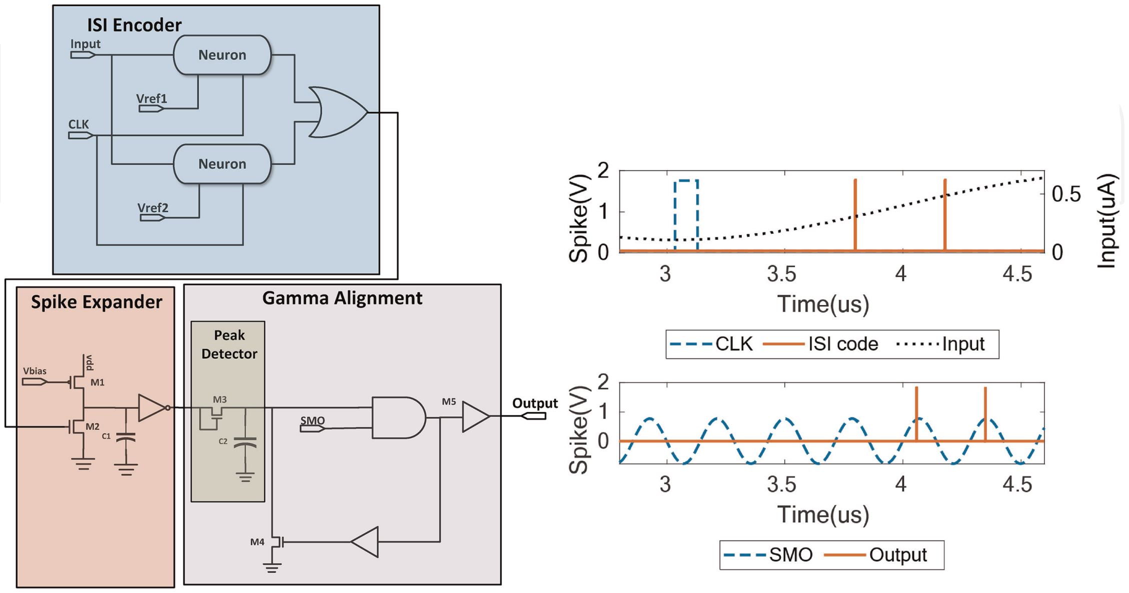


(a) Circuit schematic of TTFS-phase encoder

(b) Simulation result of rate encoder

Figure 6.
Circuit schematic and simulation result of TTFS-phase encoder.

The peak detector can only detect spikes that last longer than 10 nanoseconds (ns). However, the TTFS neuron can only output 1 ns spikes. Thus, a spike expander, as demonstrated in **Figure 6(a)**, is designed to extend the width of spikes. With Vbias controlling the charging rate on the capacitor, the width of spikes can be adjusted without changing the capacitor and thus save a lot of design area. **Figure 6(b)** illustrates the signal flows in the TTFS-phase encoder. The top panel of the figure represents the TTFS encoding function, while the bottom panel depicts the gamma alignment process. After being encoded by the TTFS neuron, the current signal is converted to spikes, and in the gamma alignment block, the TTFS spikes are moved to the next local maximum of the SMO.



(a) Circuit schematic of ISI-phase encoder

(b) Simulation result of TTFS encoder

Figure 7.
Circuit schematic and simulation result of ISI-phase encoder.

3.5 ISI-phase encoder

Similar to the TTFS-phase encoder, the ISI-phase encoder is designed with an ISI encoder with one spike expander and one gamma alignment block, as shown in **Figure 7**. With the spike expander, the output spikes of the ISI encoder have a width of over 10 ns and thus can be captured by the peak detector. To shift the expanded spikes to the local maximum of the SMO, the gamma alignment block is implemented for the spike train in one sampling window. With the two neurons in the ISI encoder, there are two spikes in one encoding window, and thus the SMO frequency should be higher; otherwise, the two spikes in the same encoding window will possibly be moved to the same local maximum to leave only one spike in the sampling window.

4. Encoder performance analysis

4.1 Performance comparison of encoding schemes

We have implemented SNNs in Python to compare the performance of these various encoding schemes, especially the classification accuracy for popular datasets. For instance, the MNIST dataset [41], the CIFAR-10 dataset [42], and the SVHN dataset [43] are utilized in the comparison for the accuracy of the above-mentioned encoding schemes. In all, 60,000 data points served as the training samples, and 10,000 were testing samples in the MNIST dataset, 50,000 samples were training ones, and 10,000 samples were testing ones. As for the SVHN samples, 73,257 are for training and 26,032 are for testing. The process of verifying encoding scheme performance can be divided into three steps. The first step is to design encoders to convert the datasets to spikes. The second is to build neural networks with these spike encoders and tune the neural networks according to the encoding schemes and datasets to get the desired output. The last step is to run the simulation and compare the accuracy achieved by the various codes.

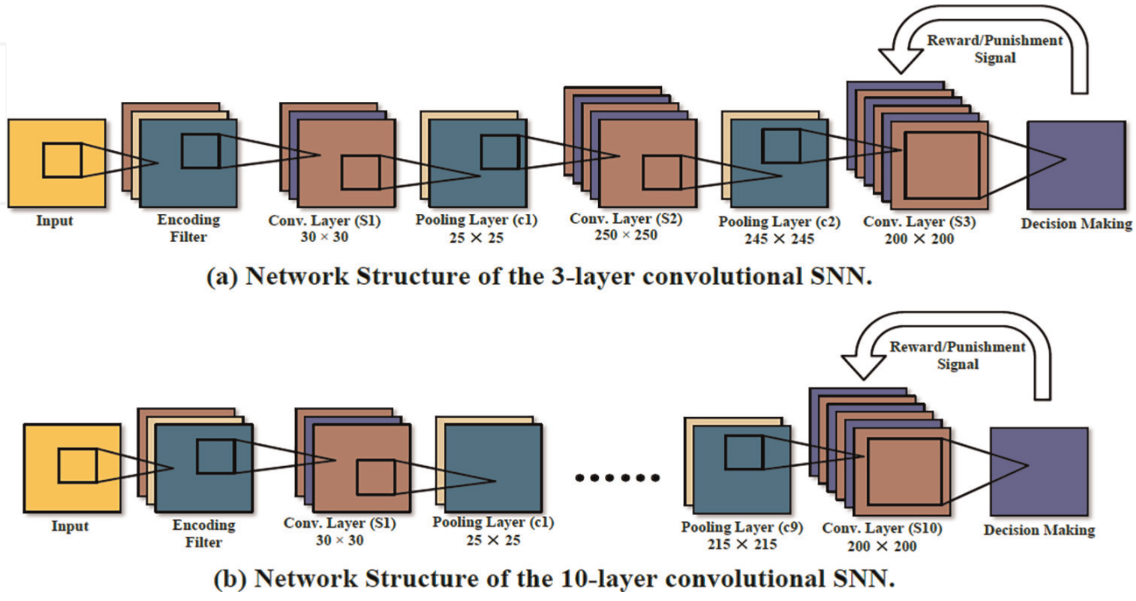
First, the rate encoder needs to be designed. Due to the nature of the rate encoding that the spiking number in the sampling window has a linear proportional relation with the input signal amplitude, the rate encoding is implemented in Python that outputs spike numbers within the range of 0 to 16. A larger input pixel value leads to a greater number of spikes in one spike train.

Second, the TTFS encoder is implemented to realize the inversely proportional relation of the input pixel value and the first spike time. One more thing to notice is that only the pixel values larger than the threshold can be transferred to spikes. This property maps the neuron functionality more closely. Similarly, the ISI encoder is implemented with multiple TTFS encoders with different thresholds. With those encoders, spikes with different timings are outputted. Those spikes have time intervals also inversely proportional to the input value. For both the TTFS and ISI encoders, the input pixel values are first linearly assigned to the range of 0 to 8 and then transferred to spike trains.

As for the multiplexing neural encoders, the corresponding encoders, the TTFS and ISI encoders, need to be added with the gamma alignment process. Realizing gamma alignment in Python means updating the spike times of the TTFS and ISI encoder outputs with the numbers of an arithmetic progression. The various frequencies of the SMO can be achieved by changing the common difference of the arithmetic progression. Thus, the TTFS-phase encoder and ISI-phase encoder are achieved with different SMO frequencies.

The datasets utilized in this experiment are transferred into different architectures with the help of the TTFS-phase and ISI-phase encoders. Certain training neural networks need to be designed and tuned accordingly to verify the training accuracy of these encoding schemes. These networks are implemented with a PyTorch package for spiking neural networks, SpykeTorch [44]. The two parameters are the threshold of neurons and the size of each layer. The threshold values of neurons are 15 for the first layer and 10 for the rest of the layers. For the MNIST dataset, a 3-layer convolutional neural network is implemented. Due to the datasets' complexity, a deeper neural network is implemented for the CIFAR-10 and SVHN datasets to get the desired results. This network is a 10-layer convolutional neural network. The neurons in both networks are all leaky integrate-and-fire (LIF) neuron models. The neuron model's parameters are: initial $V_m = 0$, $E_L = 0$, $C_m = 100pF$, $R_m = 10kOhm$, $V_{reset} = 0$, and $\tau_m = 1ms$. What's more, the synapses in the SNNs simply take into account the weights and output of neurons and provide excitation current. They do not give any specific influence themselves. The spike-timing-dependent plasticity (STDP) is the training algorithm used in these neural networks since the spiking neural network cannot be trained with backpropagation and is most commonly trained with the STDP training rule. As demonstrated in **Figure 8**, the two convolutional SNNs contain a decision-maker layer to provide reward/punishment signal as part of the reinforcement STDP learning rule. The neurons between the convolutional layers are connected in the relation of N:1. N equals the size of the kernel in the former convolutional layer. Neurons in convolutional layers are organized into local receptive fields that slide across the input data, and they share weights to efficiently learn and capture local patterns.

This simulation is performed with a 12 GB NVIDIA Tesla K80 GPU and 13 GB RAM on the Google Colab platform. For the sake of fairness, the encoding schemes are compared between the other state-of-the-art works and this work, as shown in **Table 2**. For conciseness, the results without reference are from this work. This work


Reward/Punishment Signal
Input
Encoding Filter
Conv. Layer (S1) 30 × 30
Pooling Layer (c1) 25 × 25
Conv. Layer (S2) 250 × 250
Pooling Layer (c2) 245 × 245
Conv. Layer (S3) 200 × 200
Decision Making

(a) Network Structure of the 3-layer convolutional SNN.

Reward/Punishment Signal
Input
Encoding Filter
Conv. Layer (S1) 30 × 30
Pooling Layer (c1) 25 × 25
Pooling Layer (c9) 215 × 215
Conv. Layer (S10) 200 × 200
Decision Making

(b) Network Structure of the 10-layer convolutional SNN.

Figure 8.
(a) Network structure of the 3-layer convolutional SNN. (b) Network structure of the 10-layer convolutional SNN.

Encoder type	Rate	TTFS	ISI	TTFS-phase	ISI-phase
MNIST	[45] – 83.0%	[45] – 85.0%	[45] – 90.0%	91.8%	93.8%
CIFAR-10	[46] – 79.7%	[47] – 77.4%	[48] – 83.7%	77.9%	83.8%
SVHN	[49] – 75.0%	[50] – 82.1%	82.8%	82.5%	86.4%

Table 2.
Performance comparison of code-level encoders with the MNIST, CIFAR-10, and SVHN datasets.

achieves 91.8% of accuracy and 93.78% of accuracy for the MNIST dataset. Compared with other works, the multiplexing encoding schemes classify the dataset with at most 10.78% higher accuracy. The ISI-phase encoding gets 83.83% of accuracy for the CIFAR-10 dataset, while the other works at most get 83.71% of accuracy. It classifies the dataset at most 6.4% more accurately than other neural codes. When it comes to the SVHN dataset, the multiplexing encoding has got even more desired results, especially ISI-phase encoding. It has achieved 86.4% of testing accuracy while the rate encoder only gets to 75% of accuracy. The 11.4% of accuracy difference has proved the superiority of the multiplexing encoding schemes. With these comparison results, we notice that the multiplexing encoding, although more complex than other encoding schemes since it requires one more processing step, yields the highest training accuracy for the commonly used datasets. If considering the complexity of the state-of--the-art networks is often higher than that given in this work, the multiplexing encoding can have even higher accuracy for image classification applications. Thus, the multiplexing encoding schemes have the capability to convert datasets into a more classifiable structure and get better training performance for the whole system.

5. Conclusions

In this chapter, we discussed different encoding schemes and the advantages and disadvantages of each spiking neural code. It shows that the rate encoding is straightforward but has low data capacity, the temporal codes have higher data capacity but are not robust against noise, and the multiplexing encoding schemes not only have both high data capacity and high robustness but also have high complexity and thus great power and area cost. Moreover, the mechanisms of the neural encoding schemes are also explained. The input signals are converted to different properties of spike trains in sampling windows. For instance, the signals are converted to the time intervals between the spikes for the ISI encoding, and the inputs are transferred to the spike number for the rate encoding. To utilize these encoding schemes in the analog circuit neural systems, the circuit implementations of these schemes are introduced as long as the mathematical models of the neural codes are employed. To the best of our knowledge, the ISI, TTFS-phase, and ISI-phase encoders proposed by our group are the first IC implementations. We have also built neural networks with different encoders to compare their performance when working with some commonly used image classification datasets. For fairness, we compared our group's performance of multiplexing encoders with those of the state-of-the-art works. For MNIST, the multiplexing encoder achieves 10.78% higher accuracy. For CIFAR-10, the ISI-phase encoder can classify the images 6.4% more accurately. The ISI-phase encoder gets 11.4% of higher accuracy than other works for SVHN. These comparison results have shown that although the multiplexing encoding may require more power and area, it has the potential to achieve better training performance for the whole system.

As for future work, our group is going to implement spike neural networks as well as those above-mentioned encoding schemes in the Neural Simulation Tool (NEST) simulator [51]. With such a simulator, a more detailed and more realistic simulation of SNN can be carried out and there will be more convincing evidence that multiplexing encoding schemes achieve higher data capacity and robustness than rate or temporal codes alone. What's more, we will also investigate the various training algorithms for SNNs, including STDP, spike-based backpropagation, and ANN-SNN conversion. We will dig into the cooperation of different encoding schemes with various training algorithms and find out the most suitable one for the multiplexing encoder. The hardware implementation difficulties of the training algorithms will also be considered as part of the tradeoff.

Author details

Honghao Zheng* and Yang Cindy Yi
The Bradley Department of Electrical and Computer Engineering, Virginia Tech, USA

*Address all correspondence to: zhenghh@vt.edu

References

[1] Mead C. Neuromorphic electronic systems. Proceedings of the IEEE. 1990; **78**(10):1629-1636

[2] Bai K, Yi Y. Opening the "black box" of silicon chip design in neuromorphic computing [Internet]. In: Bio-Inspired Technology. London, UK: IntechOpen; 2019. DOI: 10.5772/intechopen.83832

[3] Bai K, Yi Y. DFR: An energy-efficient analog delay feedback reservoir computing system for brain-inspired computing. ACM Journal on Emerging Technologies in Computing Systems (JETC). 2018;**14**(4):1-22

[4] Hamedani K, Zhou Z, Bai K, Liu L. The novel applications of deep reservoir computing in cyber-security and wireless communication [Internet]. In: Intelligent System and Computing. London, UK: IntechOpen; 2020. DOI: 10.5772/intechopen.89328

[5] Akopyan F, Sawada J, Cassidy A, Alvarez-Icaza R, Arthur J, Merolla P, et al. Truenorth: Design and tool flow of a 65 mw 1 million neuron programmable neurosynaptic chip. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems. 2015; **34**(10):1537-1557

[6] Hamedani K, Liu L, Hu S, Ashdown J, Wu J, Yi Y. Detecting dynamic attacks in smart grids using reservoir computing: A spiking delayed feedback reservoir based approach. IEEE Transactions on Emerging Topics in Computational Intelligence. 2019;**4**(3): 253-264

[7] Bai K, Li J, Hamedani K, Yi Y. Enabling an new era of brain-inspired computing: Energy-efficient spiking neural network with ring topology. In: 2018 55th ACM/ESDA/IEEE Design Automation Conference (DAC). New York, NY: IEEE; 24 Jun 2018. pp. 1-6

[8] Bai K, Bradley YY. A path to energy-efficient spiking delayed feedback reservoir computing system for brain-inspired neuromorphic processors. In: 2018 19th International Symposium on Quality Electronic Design (ISQED). Santa Clara, CA: IEEE; 13 Mar 2018. pp. 322-328

[9] Hamedani K, Liu L, Liu S, He H, Yi Y. Deep spiking delayed feedback reservoirs and its application in spectrum sensing of MIMO-OFDM dynamic spectrum sharing. Proceedings of the AAAI Conference on Artificial Intelligence. 2020;**34**(02):1292-1299

[10] Ghosh-Dastidar S, Adeli H. Spiking neural networks. International Journal of Neural Systems. 2009;**19**(04): 295-308

[11] Zhao C, Li J, Yi Y. Making neural encoding robust and energy efficient: An advanced analog temporal encoder for brain-inspired computing systems. In: 2016 IEEE/ACM International Conference on Computer-Aided Design (ICCAD). Austin, TX: ACM; 7 Nov 2016. pp. 1-6

[12] Zhao C, Danesh W, Wysocki BT, Yi Y. Neuromorphic encoding system design with chaos based CMOS analog neuron. In: 2015 IEEE Symposium on Computational Intelligence for Security and Defense Applications (CISDA). Verona, NY: IEEE; 26 May 2015. pp. 1-6

[13] Zhao C, Hamedani K, Li J, Yi Y. Analog spike-timing-dependent resistive crossbar design for brain inspired computing. IEEE Journal on Emerging and Selected Topics in Circuits and Systems. 2017;**8**(1):38-50

[14] Cullen KE. The neural encoding of self-motion. Current Opinion in Neurobiology. 2011;**21**(4):587-595

[15] Zhao C, Wysocki BT, Liu Y, Thiem CD, McDonald NR, Yi Y. Spike-time-dependent encoding for neuromorphic processors. ACM Journal on Emerging Technologies in Computing Systems (JETC). 2015;**12**(3): 1-21

[16] Nomura O, Sakemi Y, Hosomi T, Morie T. Robustness of spiking neural networks based on time-to-first-spike encoding against adversarial attacks. IEEE Transactions on Circuits and Systems II: Express Briefs. 2022;**69**(9): 3640-3644

[17] Zhao C, Yi Y, Li J, Fu X, Liu L. Interspike-interval-based analog spike-time-dependent encoder for neuromorphic processors. IEEE Transactions on Very Large Scale Integration (VLSI) Systems. 2017;**25**(8): 2193-2205

[18] Montemurro MA, Rasch MJ, Murayama Y, Logothetis NK, Panzeri S. Phase-of-firing coding of natural visual stimuli in primary visual cortex. Current Biology. 2008;**18**(5):375-380

[19] Panzeri S, Brunel N, Logothetis NK, Kayser C. Sensory neural codes using multiplexed temporal scales. Trends in Neurosciences. 2010;**33**(3):111-120

[20] Lankarany M, Al-Basha D, Ratté S, Prescott SA. Differentially synchronized spiking enables multiplexed neural coding. National Academy of Sciences of the United States of America. 2019; **116**(20):10097-10102

[21] Rolls ET, Treves A. The neuronal encoding of information in the brain. Progress in Neurobiology. 2011;**95**(3): 448-490

[22] Auge D, Hille J, Mueller E, Knoll A. A survey of encoding techniques for signal processing in spiking neural networks. Neural Processing Letters. 2021;**53**(6):4693-4710

[23] Kayser C, Montemurro MA, Logothetis NK, Panzeri S. Spike-phase coding boosts and stabilizes information carried by spatial and temporal spike patterns. Neuron. 2009;**61**(4):597-608

[24] Yi Y. Analog Integrated Circuit Design for Spike Time Dependent Encoder and Reservoir in Reservoir Computing Processors. Lawrence, United States: University of Kansas Center for Research, Inc.; 1 Jan 2018

[25] Davies M, Srinivasa N, Lin TH, Chinya G, Cao Y, Choday SH, et al. Loihi: A neuromorphic manycore processor with on-chip learning. IEEE Micro. 2018;**38**(1):82-99

[26] Davies M, Wild A, Orchard G, Sandamirskaya Y, Guerra GA, Joshi P, et al. Advancing neuromorphic computing with loihi: A survey of results and outlook. Proceedings of the IEEE. 2021;**109**(5):911-934

[27] Deng L, Wang G, Li G, Li S, Liang L, Zhu M, et al. Tianjic: A unified and scalable chip bridging spike-based and continuous neural computation. IEEE Journal of Solid-State Circuits. 2020; **55**(8):2228-2246

[28] Pei J, Deng L, Song S, Zhao M, Zhang Y, Wu S, et al. Towards artificial general intelligence with hybrid Tianjic chip architecture. Nature. 2019; **572**(7767):106-111

[29] Zhao C, An Q, Bai K, Wysocki B, Thiem C, Liu L, et al. Energy efficient temporal spatial information processing circuits based on stdp and spike iteration. IEEE Transactions on Circuits

and Systems II: Express Briefs. 2019; **67**(10):1715-1719

[30] Zhao C, Wysocki BT, Thiem CD, McDonald NR, Li J, Liu L, et al. Energy efficient spiking temporal encoder design for neuromorphic computing systems. IEEE Transactions on Multi-Scale Computing Systems. 2016;**2**(4): 265-276

[31] Rueckauer B, Liu SC. Conversion of analog to spiking neural networks using sparse temporal coding. In: 2018 IEEE International Symposium on Circuits and Systems (ISCAS). Florence, Italy: IEEE; 27 May 2018. pp. 1-5

[32] Paraskevopoulou SE, Constandinou TG. A sub-1μW neural spike-peak detection and spike-count rate encoding circuit. In: 2011 IEEE Biomedical Circuits and Systems Conference (BioCAS). San Diego, CA: IEEE; 10 Nov 2011. pp. 29-32

[33] Masquelier T, Hugues E, Deco G, Thorpe SJ. Oscillations, phase-of-firing coding, and spike timing-dependent plasticity: an efficient learning scheme. Journal of Neuroscience. 2009;**29**(43): 13484-13493

[34] Cattani A, Einevoll GT, Panzeri S. Phase-of-firing code. arXiv preprint arXiv:1504.03954. Apr 15 2015

[35] Akam T, Kullmann DM. Oscillatory multiplexing of population codes for selective communication in the mammalian brain. Nature Reviews Neuroscience. 2014;**15**(2):111-122

[36] Nadasdy Z. Information encoding and reconstruction from the phase of action potentials. Frontiers in systems neuroscience. 2009;**3**:6

[37] Zheng H, Mohammadi N, Bai K, Yi Y. Low-power analog and mixed-signal ic design of multiplexing neural encoder in neuromorphic computing. In: 2021 22nd International Symposium on Quality Electronic Design (ISQED). Santa Clara, CA: IEEE; 7 Apr 2021. pp. 154-159

[38] Arriandiaga A, Portillo E, Espinosa-Ramos JI, Kasabov NK. Pulsewidth Modulation-Based Algorithm for Spike Phase Encoding and Decoding of Time-Dependent Analog Data. IEEE Transactions on Neural Networks and Learning Systems. 2019;**31**(10): 3920-3931

[39] Zheng H, Anderson J, Yi Y. Approaching the area of neuromorphic computing circuit and system design. In: 2021 12th International Green and Sustainable Computing Conference (IGSC). Pullman, WA: IEEE; 18 Oct 2021. pp. 1-8

[40] Bai K, An Q, Liu L, Yi Y. A training-efficient hybrid-structured deep neural network with reconfigurable memristive synapses. IEEE Transactions on Very Large Scale Integration (VLSI) Systems. 2019;**28**(1):62-75

[41] Deng L. The mnist database of handwritten digit images for machine learning research [best of the web]. IEEE Signal Processing Magazine. 2012;**29**(6): 141-142

[42] Krizhevsky A, Hinton G. Learning multiple layers of features from tiny images

[43] Netzer Y, Wang T, Coates A, Bissacco A, Wu B, Ng AY. Reading digits in natural images with unsupervised feature learning

[44] Mozafari M, Ganjtabesh M, Nowzari-Dalini A, Masquelier T. Spyketorch: Efficient simulation of convolutional spiking neural networks

with at most one spike per neuron. Frontiers in Neuroscience. 2019;**13**:625

[45] Nowshin F, Yi Y. Memristor-based deep spiking neural network with a computing-in-memory architecture. In: 2022 23rd International Symposium on Quality Electronic Design (ISQED). Santa Clara, CA: IEEE; 6 Apr 2022. pp. 1-6

[46] Nguyen VT, Trinh QK, Zhang R, Nakashima Y. STT-BSNN: An In-Memory Deep Binary Spiking Neural Network Based on STT-MRAM. IEEE Access. 2021;**9**:151373-151385

[47] Cao Y, Chen Y, Khosla D. Spiking deep convolutional neural networks for energy-efficient object recognition. International Journal of Computer Vision. 2015;**113**(1):54-66

[48] Park S, Kim S, Choe H, Yoon S. Fast and efficient information transmission with burst spikes in deep spiking neural networks. In: 2019 56th ACM/IEEE Design Automation Conference (DAC). Las Vegas, NV: IEEE; 2 Jun 2019. pp. 1-6

[49] Wang Z, Liu J, Ma Y, Chen B, Zheng N, Ren P. Perturbation of spike timing benefits neural network performance on similarity search. IEEE Transactions on Neural Networks and Learning Systems. Sep 2022;**33**(9): 4361-4372

[50] Ma C, Yan R, Yu Z, Yu Q. Deep spike learning with local classifiers. IEEE Transactions on Cybernetics. May 2023; **53**(5):3363-3375

[51] Plesser HE, Diesmann M, Gewaltig MO, Morrison A. NEST: the Neural Simulation Tool. In: Jaeger, D., Jung, R. (eds) Encyclopedia of Computational Neuroscience. New York, NY: Springer; 2015. DOI: 10.1007/978-1-4614-6675-8_258