A Memristor-Based Learning Engine for Synaptic Trace-Based Online Learning

Wang, Deyu; Xu, Jiawei; Li, Feng; Zhang, Lianhao; Cao, Chengwei; Stathis, Dimitrios; Lansner, Anders; Hemani, Ahmed; Zheng, Li-Rong; Zou, Zhuo

Published in:
IEEE Transactions on Biomedical Circuits and Systems

Link to article, DOI:
10.1109/TBCAS.2023.3291021

Publication date:
2023

Document Version
Peer reviewed version

A Memristor-Based Learning Engine for Synaptic Trace-Based Online Learning

Deyu Wang, Jiawei Xu, *Member, IEEE,* Feng Li, Lianhao Zhang, Chengwei Cao, Dimitrios Stathis, Anders Lansner, Ahmed Hemani, Li-Rong Zheng, *Senior Member, IEEE,* and Zhuo Zou, *Senior Member, IEEE*

***Abstract*—The memristor has been extensively used to facilitate the synaptic online learning of brain-inspired spiking neural networks (SNNs). However, the current memristor-based work can not support the widely used yet sophisticated trace-based learning rules, including the trace-based Spike-Timing-Dependent Plasticity (STDP) and the Bayesian Confidence Propagation Neural Network (BCPNN) learning rules. This paper proposes a learning engine to implement trace-based online learning, consisting of memristor-based blocks and analog computing blocks. The memristor is used to mimic the synaptic trace dynamics by exploiting the nonlinear physical property of the device. The analog computing blocks are used for the addition, multiplication, logarithmic and integral operations. By organizing these building blocks, a reconfigurable learning engine is architected and realized to simulate the STDP and BCPNN online learning rules, using memristors and 180 nm analog CMOS technology. The results show that the proposed learning engine can achieve energy consumption of 10.61 pJ and 51.49 pJ per synaptic update for the STDP and BCPNN learning rules, respectively, with a 147.03× and 93.61× reduction compared to the 180 nm ASIC counterparts, and also a 9.39× and 5.63× reduction compared to the 40 nm ASIC counterparts. Compared with the state-of-the-art work of Loihi and eBrainII, the learning engine can reduce the energy per synaptic update by 11.31× and 13.13× for trace-based STDP and BCPNN learning rules, respectively.**

***Index Terms*—Memristor, learning engine, trace dynamics, online learning, spike-timing-dependent plasticity (STDP), Bayesian confidence propagation neural network (BCPNN), spiking neural network (SNN).**

I. Introduction

This work was supported in part by the Key-Area Research and Development Program of Guangdong Province (2021B0909060002); in part by the NSFC-STINT joint project (Grant 62011530132); in part by the National Natural Science Foundation of China under Grant 62076066 and Grant 92164301; in part by the Shanghai Municipal Science and Technology Major Project under Grant 2021SHZDZX0103. *(Corresponding authors: Zhuo Zou; Li-Rong Zheng; Jiawei Xu.)*

Deyu Wang, Feng Li, Chengwei Cao, and Zhuo Zou are with the State Key Laboratory of Integrated Chips and Systems, School of Information Science and Tehchnology, Fudan University, Shanghai 200433, China (e-mail: deyuwang19@fudan.edu.cn; zhuo@fudan.edu.cn).

Jiawei Xu is with the Guangdong Institute of Intelligence Science and Technology, Zhuhai, Guangdong 519115, China, and also with the School of Electrical Engineering and Computer Science, KTH Royal Institute of Technology, Stockholm, Swden (e-mail: xujiawei@gdiist.cn).

Lianhao Zhang is with the Department of Electrical Engineering, Technical University of Denmark, Kongens Lyngby, Denmark.

Dimitrios Stathis, Anders Lansner, and Ahmed Hemani are with the School of Electrical Engineering and Computer Science, KTH Royal Institute of Technology, Stockholm, Sweden.

Lirong Zheng is with the School of Information Science and Technology, Fudan University, Shanghai 200437, China, and also with the Guangdong Institute of Intelligence Science and Technology, Zhuhai, Guangdong 519115, China (e-mail: lrzheng@fudan.edu.cn).

IN the recent years, the remarkable development of the Internet of Things (IoT) has led to a significant growth of information at the edge, requiring time- and energy-efficient online processing on terminal devices [1]. Inspired by the biological nervous system, the spiking neural network (SNN) has gained great interest due to its potential to address spatiotemporal information. Featuring event-driven processing, SNNs offer a power-saving computing paradigm for IoT and edge computing. In addition, the Hebbian learning rules of SNNs allow for unsupervised online learning from unlabeled data [2]. Furthermore, the synaptic dynamics exhibited in the Hebbian learning of SNNs have also been leveraged and widely used in neuromorphic applications [3]. Nonetheless, the scaling of complementary metal-oxide-semiconductor (CMOS) technology gives rise to issues related to energy consumption and storage, particularly in the implementation of online learning.

The memristor provides a possibility to alleviate the aforementioned issues by fusing computation and storage [4]. The memristor has various characteristics such as biomimetic conductance adjustability, nonvolatility, and compatibility with the current CMOS process. Especially, the hysteretic behavior of the memristor enables it to represent multiple values, which resembles biological synapses. Therefore, the memristor is a promising candidate to realize synaptic online learning of SNNs and numerous studies have been conducted. In [5], a hybrid CMOS-memristive circuit was proposed to implement the triplet STDP learning algorithm. In [6], the suppression triplet STDP learning rule was realized using second-order memristors. A memristor-based circuit is proposed, based on the mechanism of biological unsupervised nonassociative learning [7]. Recently, a multi-parameter control circuit based on the forgetting memristor was proposed, where the STDP learning mechanism is implemented [8]. Furthermore, various memristor-based SNNs that support online learning have been presented. In [9], a simplified spike-timing-dependent plasticity (STDP) learning rule was proposed to carry out learning tasks in SNNs using memristors. An STDP-based SNN with multi-memristive synapses was proposed for unsupervised learning of temporal correlations in [10]. Another study presented a memristor-based SNN that utilizes nonlinear weight update in its memristor synapses based on STDP [11]. In [12], a SNN hardware was constructed using an optimized memristor-based synapse model, which employs the BCM mechanism. However, the current memristor-based SNNs mostly adopt relatively simple synaptic learning rules such as simplified STDP. They can not support the more biologically plausible yet sophisticated trace-based learning

rules, such as trace-based variants of STDP and the Bayesian confidence propagation neural network (BCPNN) learning rules.

Trace-based learning rules refer to the unsupervised online learning rules that involve the synaptic trace variables into the learning process, which better describe the synaptic plasticity discovered in the human brain [13], [14]. Therefore, trace-based learning rules have been widely used in neuromorphic applications [15]–[21]. Recently, several energy-efficient implementations for trace-based online learning have been proposed. A neuromorphic SNN processor named Loihi was presented, which can support programmable synaptic learning rules including trace-based pairwise and triplet STDP [22]. A real-time custom ASIC implementation called eBrainII was developed, where the BCPNN learning rule is employed for synaptic online learning [23]. An FPGA-based SNN processor was proposed, where the trace-based STDP is adopted as the online learning algorithm [24]. Despite their good performance and energy efficiency, these works still encounter bottlenecks in the aspects of storage and memory access of synaptic state variables, restricted by the memory wall of the von Neumann architecture. As a device fusing computation and storage, the memristor offers the potential for in-memory computation of trace variables [25]. Recently, the inherent PCM conductance drift was exploited to realize the eligibility trace in reinforcement learning [26]. Besides, we studied how to map the computation of synaptic trace variables to a memristor model to implement trace-based learning rules [27]–[29].

In this work, we propose a reconfigurable learning engine for the energy-efficient implementation of trace-based online learning. Especially, the learning engine enables in-memory computation for trace dynamics by exploiting the nonlinear physical property of memristors. This paper advances our previous work in [29] and presents three new contributions as follows:

- This paper proposes a learning engine for generalized trace-based learning rules of spiking neural networks. The learning engine consists of memristor-based blocks and analog computing blocks, thus supporting the computation of the nonlinear decay of trace variables and other required calculation operations of trace-based learning.
- A novel reconfigurable design is presented in the learning engine, where the building blocks can be reused to support diverse trace-based learning rules. The flexibility and reconfigurability of the design are validated by implementing the trace-based STDP and the BCPNN learning rules at the circuit level.
- This paper evaluates the performance and energy efficiency of the proposed learning engine. The results show that the learning engine can achieve energy consumption of 10.61 pJ and 51.49 pJ per synaptic update for the STDP and BCPNN learning rules, respectively, with a 147.03× and 93.61× reduction compared to the 180 nm ASIC counterparts, and also a 9.39× and 5.63× reduction compared to the 40 nm ASIC counterparts. Compared with the state-of-the-art work, the learning engine can reduce the energy per synaptic update by 11.31× and 13.13× for trace-based STDP and BCPNN learning rules, respectively.

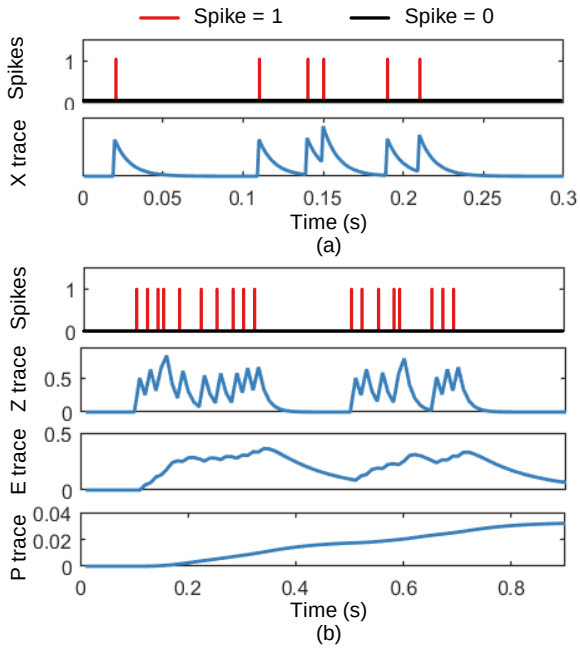

Fig. 1. (a) The spike train and the corresponding trace variable. (b) Three key traces of the BCPNN learning rule, including Z trace, E trace, and P trace. Z trace is the low-pass filtered output of the spike train, E trace is the low-pass filtered output of Z trace, and P trace is the low-pass filtered output of E trace.

The rest of this paper is organized as follows. Section II introduces the background knowledge about trace-based learning and memristor nonlinearity. Section III presents the memristor-based learning engine for trace-based online learning. Section IV presents the circuit model and design of the building blocks of the learning engine. Section V reports the experimental results, evaluates the performance and energy efficiency, and compares the proposed learning engine with other counterparts. Section VI concludes the paper.

II. Preliminary

A. Trace-based Learning

In spiking neural networks, information is encoded and processed through sequences of spikes. The precise timing of these spikes is utilized to modulate the strength of synaptic connections. In [13], it was discovered that the occurrence of pre- or post-synaptic spikes does not directly cause an immediate change in synaptic weights. Instead, spikes may trigger the update of hidden internal variables that are closely associated with synaptic plasticity. Therefore, the internal variable called trace is introduced. The trace variable increases when there is a spike and decays when there is no spike, as defined below:

$$\frac{dx}{dt} = -\frac{x}{\tau} + k \cdot s \tag{1}$$

Where x represents the trace variable and s denotes the spike. The constant k determines the rising amplitude, and the time constant τ controls the decay rate. The dynamics of a trace variable is visualized in Fig. 1 (a).

The trace variable plays a crucial role in emulating synaptic dynamics and is a fundamental component of trace-based

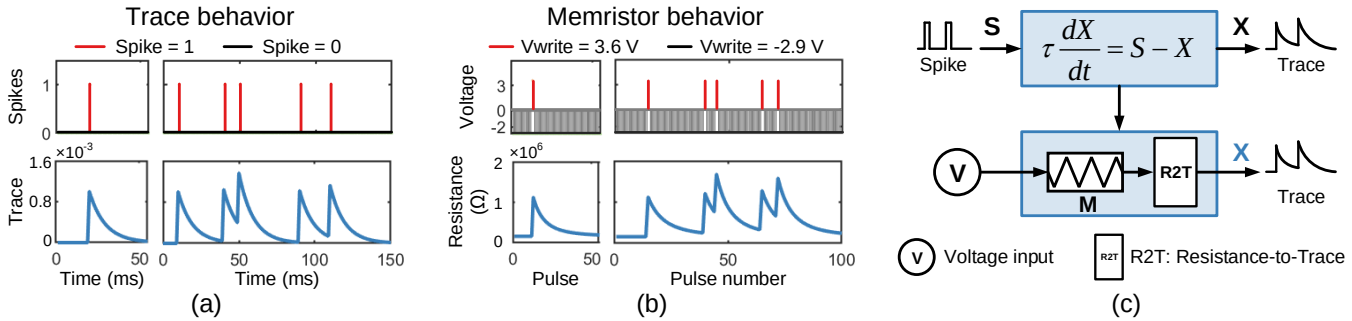

Fig. 2. (a) The dynamics of the trace variable. (b) The resistance of the memristor when applied with consecutive voltage pulses. (c) The schematic of the memristor-based block to transform the incoming spike train to the trace variable.

learning rules, such as the widely used trace-based STDP variants and the more sophisticated BCPNN learning rule.

1) Trace-based STDP: STDP is a Hebbian online learning rule, where the strength of synaptic weights is adjusted according to the timing of pre- and post-synaptic spikes [30]. To date, multiple variants of STDP have been studied and proposed based on the classic STDP model. In [13], a pairwise STDP and a triplet STDP are implemented with synaptic trace variables, which can give a more biologically plausible description of the STDP mechanism of the human brain.

The trace-based variants of the STDP learning rule have been widely adopted in neuromorphic applications. For instance, it was employed in an SNN for digit recognition, which achieved high classification performance on the MNIST benchmark [15]. Another work proposed a self-organizing SNN for unsupervised learning tasks by combining trace-based STDP with the self-organizing map (SOM) algorithm [16]. Additionally, a spiking recurrent neural network was presented for classifying electromyography (EMG) gestures, utilizing the trace-based STDP and a soft winner-take-all mechanism [17].

So far, a variety of digital neuromorphic implementations have been proposed, where the trace-based STDP learning rule is adopted for online learning. The Loihi processor developed by Intel can support trace-based pairwise and triplet STDP with its programmable learning engine [22]. An FPGA-based SNN processor was presented for image classification, and the trace-based STDP is employed for the online learning process [24]. Recently, a neuromorphic core for edge systems was presented, which can support various on-chip learning methods, including the variants of trace-based STDP [31].

2) BCPNN Learning Rule: The BCPNN learning rule is derived from Bayes' rule [32] and has clear links to biological synaptic plasticity processes [14]. In the spike-based version of this learning rule, three key traces are introduced to keep track of the dynamics of the pre-, post-synaptic, and coincident synaptic activities [33], as visualized in Fig. 1 (b). The synaptic weight is thus modulated according to the trace variables.

The BCPNN learning rule has been leveraged to handle pattern recognition tasks in an unsupervised manner. Competitive classification performance on the MNIST and Fashion-MNIST benchmarks can be achieved, with an accuracy of 98.6% and 88.9% on test sets, respectively [18]. Moreover, the BCPNN learning rule has been used extensively in detailed spiking models of brain-like cognitive capabilities such as associative memory [19], working memory [20], and episodic memory [21].

The BCPNN learning rule has been utilized to construct cortex-like neural networks. The neural networks have been implemented in high-performance computers, such as clusters [19], and GPUs [34]. Recently, a custom ASIC implementation called eBrainII was developed, and the BCPNN learning rule is employed for the online learning process [23].

B. Memristor Nonlinearity

Although the current digital neuromorphic implementations for trace-based online learning have demonstrated good performance and energy efficiency, they still suffer from bottlenecks in the storage and memory access of synaptic state variables. As an emerging device fusing computation and storage, the memristor provides a possibility to relieve these issues. However, the current memristor-based work can only support simple online learning rules like the simplified STDP. They can not deal with the more sophisticated trace-based learning rules that involve the nonlinear decay of trace variables.

The memristor exhibits rich hysteretic current-voltage behavior, which is commonly observed in various nanoscale electronic devices. Nonetheless, there can be considerable nonlinearities in the ionic transport process, due to the electric fields generated by voltage inputs in nanoscale devices. This phenomenon is known as nonlinear dopant drift or nonlinearity [35]. The resistance of the memristor changes in a nonlinear manner when applied with external voltage pulses, which is similar to the nonlinear decay of trace variables. Therefore, it is motivated that the memristor nonlinearity can be exploited to emulate the nonlinear decay of trace variables.

III. Learning Engine

In this work, we propose a generalized learning engine for trace-based online learning. The learning engine is comprised of memristor-based blocks and analog computing blocks, which perform the necessary calculations. Especially, the memristor's behavior is utilized to simulate the nonlinear decay of the synaptic trace variable. Moreover, the learning engine features a reconfigurable architecture, allowing it to support both the trace-based pairwise STDP and the BCPNN learning rules.

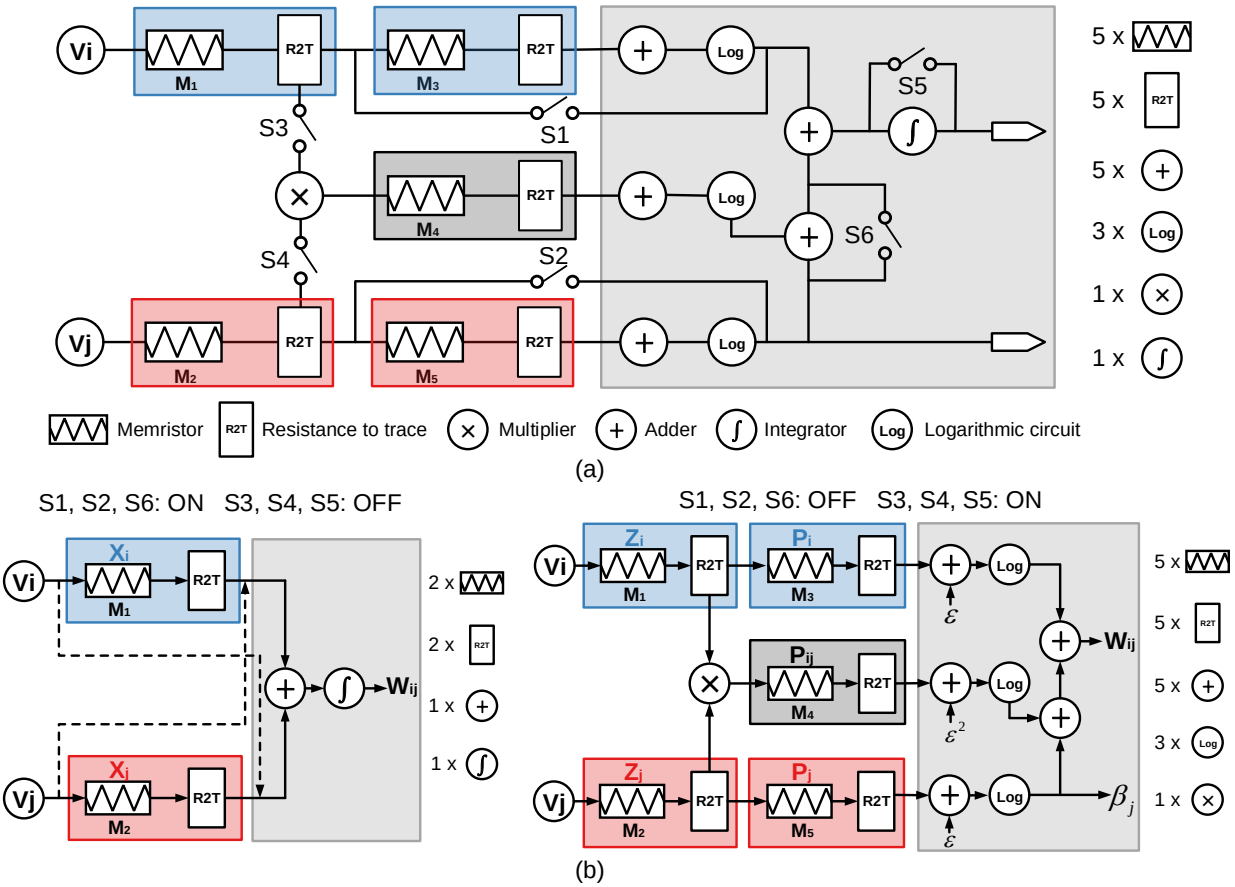

Fig. 3. (a) The reconfigurable architecture of the learning engine, composed of memristor-based building blocks and analog computing blocks. (b) Two working modes of the learning engine, i.e., the trace-based pairwise STDP and the BCPNN learning rule.

A. Synaptic Trace with Memristor

The behaviors of the trace variable and the memristor resistance are illustrated in Fig. 2 (a) and (b), respectively. The trace variable increases in response to spikes and decays between them, while the memristor resistance increases in response to positive voltage pulses and decreases in response to negative voltage pulses. This suggests that the consistency between the dynamics of the trace and the behavior of the memristor resistance can be utilized to compute trace variables.

Fig. 2 (c) presents the schematic of a memristor-based block, which is designed for the computation of trace variables. Since the trace variable changes with the incoming spike sequences, the memristor resistance is stimulated with external voltage pulses. In this way, the iteration of the trace variable can be represented with the memristor resistance. The different rise amplitudes and decay rates of the traces (represented by the constant k and the time constant τ in formula (1)) can be correspondingly achieved by adjusting the amplitude of voltage pulses applied across the memristor. In addition, a resistance-to-trace (R2T) module is needed to convert the resistance to the trace value at a specific mapping scale. The details of the mapping process between the memristor resistance and the trace variable can be referred to in our previous work [29].

B. Reconfigurable Architecture of the Learning Engine

Fig. 3 (a) depicts the reconfigurable architecture of the learning engine. The learning engine comprises two types of building blocks: the memristor-based block and the analog computing blocks. The memristor-based block, consisting of a memristor element and a R2T module, is responsible for computing trace variables. Each memristor-based block represents a trace variable in the trace-based learning rules. The analog computing blocks of the learning engine handle the other calculation operations, including addition, multiplication, logarithmic, and integral operations. The circuit model and design of these building blocks will be elaborated in Section IV.

The memristor-based block and analog computing blocks can support the required operations for trace-based online learning. Therefore, a range of trace-based learning rules in SNNs can be implemented by cascading the building blocks. In this work, the proposed learning engine is equipped with five memristor-based blocks, five adders, three logarithmic modules, a multiplier, and an integrator. By organizing and configuring these building blocks, the learning engine can support both the trace-based pairwise STDP and the BCPNN learning rules.

The learning engine achieves reconfigurability by switching the data path to support different learning rules. As shown

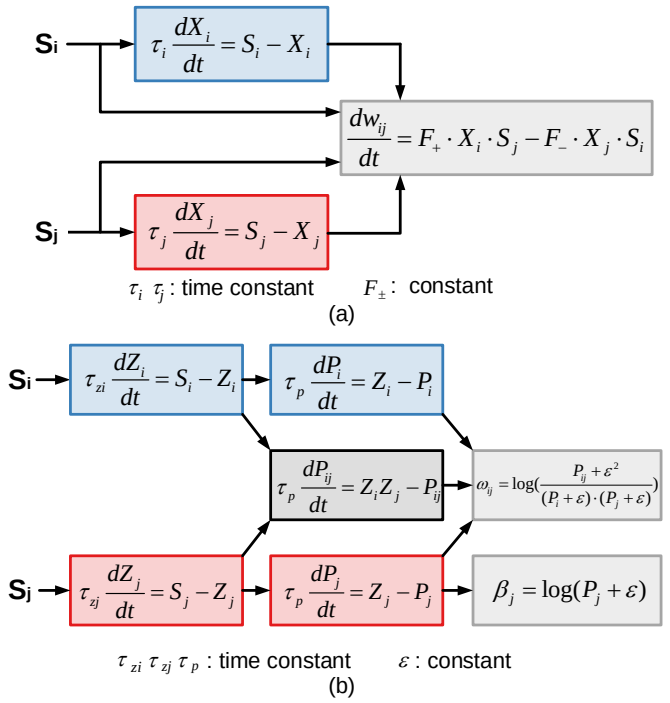

Fig. 4. (a) The iteration process of the trace-based pairwise STDP, including two traces (X_i and X_j). (b) The iteration process of the simplified BCPNN learning rule, including five traces (Z_i, Z_j, P_i, P_j, and P_{ij}). The E traces in the original BCPNN learning rule are mainly used for delayed reinforcement learning and can be omitted in many cases.

in Fig. 3 (b), the learning engine can be configured into two working modes. Switches S1, S2, S3, S4, S5, and S6 control the switching between the two modes. When switches S1, S2, and S6 are turned on and switches S3, S4, and S5 are turned off, the learning engine functions for the trace-based pairwise STDP. On the other hand, when switches S3, S4, and S5 are turned on and switches S1, S2, and S6 are turned off, the learning engine functions for the BCPNN learning rule.

If the learning engine is configured for trace-based pairwise STDP, the incoming pre- and post-synaptic spike sequences S_i and S_j, which are represented with voltage pulses, drive the iteration of two synaptic traces (X_i and X_j) with time constants τ_i and τ_j, respectively, as illustrated in Fig. 4 (a). This requires the use of two memristor-based blocks to represent the trace variables. Additionally, an adder and an integrator are necessary to update the synaptic weight W_{ij} based on the two traces.

When the engine is configured for the BCPNN learning rule, the spike trains S_i and S_j drive pre- and post-synaptic Z traces (Z_i and Z_j) with corresponding time constant τ_{zi} and τ_{zj}, respectively. These Z traces in turn drive the three P traces (P_i, P_j, and P_{ij}) following the same kind of dynamics with the time constant τ_p, as shown in Fig. 4 (b). Therefore, the iteration of the BCPNN learning rule requires five memristor-based blocks to represent the five trace variables. For the P_{ij} trace, the input to the block is the product of the outputs of Z_i and Z_j, thus requiring a multiplier. Besides, five adders and three logarithmic modules are needed to calculate the weight W_{ij} and the bias β_j based on the trace variables.

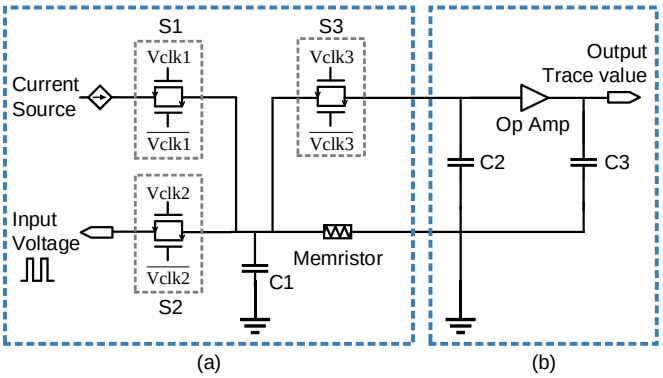

Fig. 5. The circuit diagram of the memristor-based block. (a) The memristor element is stimulated and the resistance of the memristor is sampled. (b) The resistance-to-trace module is used to convert the resistance to a trace value.

In this work, the most commonly used trace-based pairwise STDP and the simplified BCPNN learning rule without E traces are implemented. The reconfigurability of the learning engine allows for the reuse of the same computing blocks for both two configurations. Moreover, it should be noted that the other STDP variants such as the triplet STDP containing three traces and the original BCPNN learning rule with eight traces, can also be supported by increasing the number of building blocks inside the learning engine. The learning engine's building blocks are scalable and reconfigurable, allowing for the support of a wider range of trace-based learning rules of SNNs.

IV. Ciruit Model and Design

In this section, the memristor-based block and the analog computing blocks of the learning engine are modeled and designed at the circuit level.

A. Memristor-Based Block

In the memristor-based block, the memristor element is constructed with a SPICE model in Verilog-A. In addition, peripheral circuits are needed to sample the resistance of the memristor and transform it into a trace value.

1) Verilog-A Memristor Model: In this work, the VTEAM model [36] combined with an improved window function based on [27] is adopted to characterize the behavior of the memristor.

The VTEAM model is a widely used SPICE-level model for memristor devices. It is compatible with various window functions and exhibits great flexibility in simulating the non-linearity of memristors, which is defined as follows:

$$\frac{dx(t)}{dt} = \begin{cases} k_{\text{off}} \cdot (\frac{v(t)}{v_{\text{off}}} - 1)^{\alpha_{\text{off}}} \cdot f(x(t)), & 0 < v_{\text{off}} < v \\ 0, & v_{\text{on}} < v < v_{\text{off}} \\ k_{\text{on}} \cdot (\frac{v(t)}{v_{\text{on}}} - 1)^{\alpha_{\text{on}}} \cdot f(x(t)), & v < v_{\text{on}} < 0 \end{cases}$$
$$R(t) = R_{\text{on}} + (R_{\text{off}} - R_{\text{on}}) \cdot x(t)$$
$$v(t) = R(t) \cdot i(t) \quad (2)$$

Here, $x(t)$ denotes an internal state variable, $v(t)$ represents the voltage applied to the memristor, whereas $i(t)$ indicates the current. $R(t)$ stands for the resistance, and $f(x)$ is the

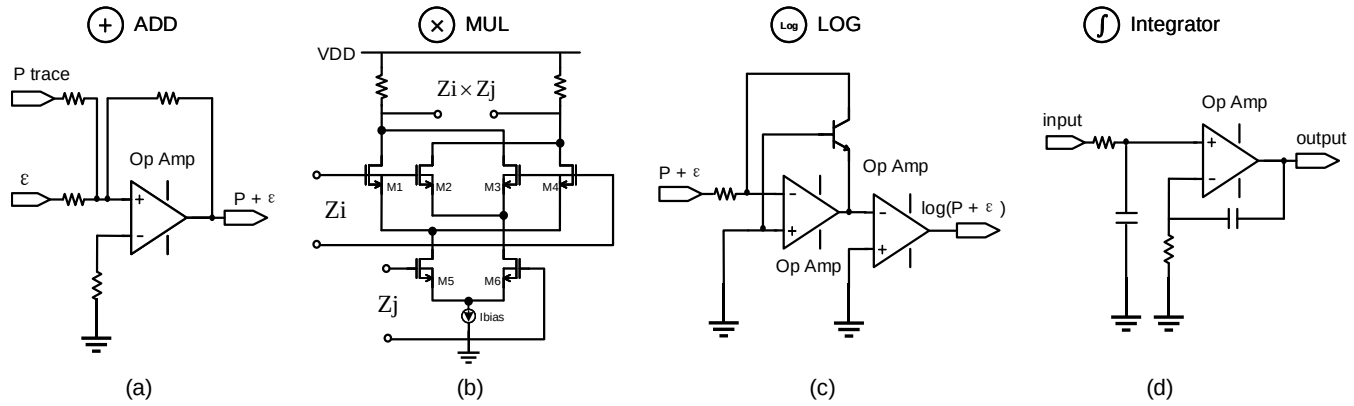

Fig. 6. The circuit diagram of the analog computing blocks: (a) The adder. (b) The multiplier. (c) The logarithmic module. (d) The integrator.

window function. The threshold voltages of the memristor are designated as $v_{\rm on}$ and $v_{\rm off}$, while $R_{\rm on}$ and $R_{\rm off}$ refer to the minimum and maximum resistance of the memristor. The parameters $k_{\rm on}$, $k_{\rm off}$, $\alpha_{\rm on}$ and $\alpha_{\rm off}$ are adjustable parameters.

To emulate the nonlinearity of memristor devices, the window function is introduced as an essential part of the memristor model. Recently, we proposed a flexible yet concise window function to mimic the memristor nonlinearity [27]. In this work, we improve the window function proposed in [27], by introducing a tuning function $p(i)$ to replace the previous tuning parameter p. With this enhancement, the window function can more accurately capture the varying nonlinearities of physical memristors as their resistance increases or decreases, by applying different parameters $p_{\rm off}$ and $p_{\rm on}$ under different current directions.

The improved window function $f(x)$ is provided as follows:

$$
\begin{aligned}
f(x) &= j[\mathrm{sgn}(-i)\cdot(x-1)+\mathrm{stp}(-i)]^{p(i)} \\
\mathrm{sgn}(i) &= \begin{cases} 1, i \geq 0 \\ -1, i < 0 \end{cases} \\
\mathrm{stp}(i) &= \begin{cases} 1, i \geq 0 \\ 0, i < 0 \end{cases} \\
p(i) &= \begin{cases} p_{\rm off}, i \geq 0 \\ p_{\rm on}, i < 0 \end{cases}
\end{aligned}
\tag{3}
$$

where i is the current, j is a tuning parameter, and $p(i)$ is a tuning function. The tuning function $p(i)$ determines the decrease rate of the window function near the boundaries. The nonlinearity is weakened when $p_{\rm off}$ or $p_{\rm on}$ approaches 0.

2) Circuit Design: The circuit diagram for the memristor-based block is depicted in Fig. 5. The resistance of the memristor is stimulated and sampled in Fig. 5 (a). The memristor element is constructed using the Verilog-A SPICE model described earlier. The input voltage represents the incoming spike train, with positive voltage indicating a spike and negative voltage indicating no spike. Switches S1, S2, and S3 are controlled by voltages Vclk1, Vclk2, and Vclk3, respectively, to switch between the circuit's two operating states. In the holding state, switch S2 is on while S1 and S3 are off, and the voltage source excites the memristor, changing its resistance. In the sampling state, switches S1 and S3 are on while switch S2 is off, and a constant current from the current source passes through the memristor to convert its resistance into a voltage value, which is stored in capacitor C1.

Fig. 5 (b) is used to transform the sampled memristor resistance into an expected trace value. In the sampling state, when switches S1 and S3 are on, and switch S2 is off, the memristor resistance is sampled and stored as a voltage value in capacitor C1. Since switch S3 is on, the voltage stored in capacitor C2 is equal to the voltage stored in capacitor C1. Next, the operational amplifier amplifies the voltage stored in capacitor C2, and the resulting voltage is stored in capacitor C3 as the input voltage for the next-stage circuit.

B. Analog Computing Blocks

In addition to the memristor-based block, several analog computing blocks are needed to perform additional computation operations for trace-based learning, including the addition, multiplication, logarithmic and integral operations.

1) Adder: Fig. 6 (a) presents the diagram of the non-inverting adder circuit. As an example, the sampling voltage of the P trace is added with the constant parameter ε.

2) Multiplier: As shown in Fig. 6 (b), the multiplier is implemented based on the classic Gilbert cell. The sampling voltage of Z_i trace is multiplied by the sampling voltage of Z_j trace. The aspect ratios W/L for $M1$, $M2$, $M3$, $M4$ are 1μm/0.18μm, while the aspect ratios W/L for $M5$, $M6$ are 2μm/0.18μm. In addition, the bias voltages Vdc for Z_i and Z_j are 1.5 V and 1.3 V respectively.

3) Logarithmic Module: Fig. 6 (c) presents the diagram of the logarithmic circuit. The input is the sum of the P trace and the constant parameter ε, which is then logarithmically calculated through a triode and an operational amplifier.

4) Integrator: Fig. 6 (d) presents the diagram of the integrator. The two capacitors both have a capacitance of 1 pF.

The learning engine proposed here consists of multiple analog computing blocks, including five adders, a multiplier, three logarithmic modules, and an integrator. To enable trace-based STDP, an adder, and an integrator are utilized. On the other hand, for the BCPNN learning rule, five adders, a multiplier, and three logarithmic modules are required.

TABLE I
FITTING PHYSICAL MEMRISTOR DEVICES WITH THE IMPROVED MEMRISTOR MODEL

Memristor	Parameters Setting (j = 1)											RMSE		
	α_{off}	α_{on}	v_{off}[V]	v_{on}[V]	$R_{\text{off}}[\Omega]$	$R_{\text{on}}[\Omega]$	$k_{\text{off}}[\text{s}^{-1}]$	$k_{\text{on}}[\text{s}^{-1}]$	p_{off}	p_{on}	dt	AICAS'22 [29]	This Work	Reduction
Ferroelectric [37]	5	5	1.4	-2.0	1.4×10^{7}	1.6×10^{5}	1.6×10^{5}	-1.3×10^{9}	1.48	1.79	100 ns	8.2×10^{5}	7.2×10^{5}	12.3%
STO-based [6]	2	2	1.3	-1.3	2.0×10^{9}	2.9×10^{8}	1.2×10^{5}	-1.2×10^{6}	4.50	2.52	10 μs	3.0×10^{7}	7.9×10^{6}	73.3%
NiO-based [38]	1	1	0.1	-0.1	3.2×10^{4}	2.6×10^{4}	7.4	-11.1	1.22	5.16	2 ms	393.8	86.5	78.0%

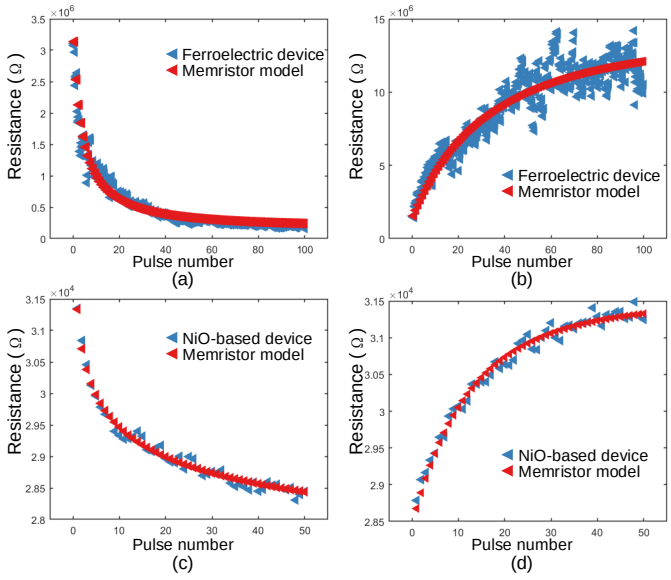

Fig. 7. The fitting results of two physical memristor devices with the memristor model. (a) The ferroelectric memristor is applied with consecutive negative voltage pulses. (b) The ferroelectric memristor is applied with consecutive positive voltage pulses. (c) The NiO-based memristor is applied with consecutive negative voltage pulses. (d) The NiO-based memristor is applied with consecutive positive voltage pulses.

V. EXPERIMENTAL RESULTS

We perform various experiments to validate and evaluate the proposed learning engine. First, we verify the behavioral consistency between the Verilog-A memristor model with real devices. Then we perform SPICE-level simulations to verify the feasibility of the learning engine. The energy efficiency of the learning engine is also evaluated. Furthermore, we conduct comparative tests on different computing platforms to demonstrate the advantages of the learning engine. In particular, we perform ASIC designs for the STDP and BCPNN learning rules and synthesize the RTL designs under the process of 180 nm and 40 nm, respectively. Moreover, the learning engine is also compared with the state-of-the-art works. Finally, we present the discussion on device variation issues and our follow-up work in the future.

A. Parameters for Memristor SPICE Model

The memristor element of the memristor-based block is constructed with a memristor SPICE model in Verilog-A. To mimic the real memristor behavior, the parameters for the memristor model are obtained by fitting several physical memristor devices [6], [37], [38].

To fit the experimental data of physical devices, the root mean square error (RMSE) is used as the optimization metric, as defined below:

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\hat{y}_k - y_k)^2} \tag{4}$$

Here, N denotes the overall number of samples, y_k represents the kth sample of the reference resistance of physical devices, while $\hat{y}_k$ denotes the kth sample of the resistance fitted with the memristor model.

According to the tuning resistance by consecutive voltage pulses reported in [6], [37], [38], a set of parameters is selected to fit the memristor model to the experimental data. To obtain the optimized set of parameters, the RMSE is minimized using gradient descent [39] and simulated annealing algorithms [40]. Especially, the parameters v_{off}, v_{on}, R_{off}, R_{on} are established based on the experimental data reported in [6], [37], [38], respectively. The timestep dt is determined based on the reported pulse duration. The parameters j, α_{off}, and α_{on} are manually set to demonstrate similarity. The fitting procedure is then iterated on k_{off}, k_{on}, p_{off}, and p_{on} to minimize the error function given in formula (4).

The optimized parameters of the memristor model for three physical memristor devices are presented in Table I. For the ferroelectric memristor, a minimum RMSE of 7.2×10^{5} is attained, which reduces the RMSE by 12.3% compared with our previous work in [29]. For the STO-based memristor, a minimum RMSE of 7.9×10^{6} is achieved, with a reduction of RMSE by 73.3%. For the NiO-based memristor, a minimum RMSE of 86.5 is achieved, with a reduction of RMSE by 78%. The fitting results of the ferroelectric memristor and the NiO-based memristor are also visualized in Fig. 7. The results show that the improved memristor model has a decent fitting effect on the physical memristor devices. In the subsequent experiments, the set of fitting parameters for the ferroelectric memristor device is adopted to implement the memristor element in SPICE-level simulations.

B. SPICE Simulation of Trace-based Learning Rules

To validate the feasibility of the learning engine for trace-based online learning, we performed simulations at the circuit level. Specifically, both the trace-based pairwise STDP and the BCPNN learning rules are simulated in the SPICE simulation environment. To ensure the reliability of the memristor-based building block and maintain compatibility between memristors and analog circuits, we adopt the 180 nm process to implement

TABLE II
SPICE SIMULATION RESULTS OF TRACE-BASED LEARNING RULES

	STDP			BCPNN						
	X_i	X_j	w_{ij}	Z_i	Z_j	P_i	P_j	P_{ij}	w_{ij}	β_j
Mean Error	4.01×10^{-5}	3.47×10^{-5}	1.53×10^{-10}	1.96×10^{-2}	1.47×10^{-2}	1.35×10^{-3}	1.18×10^{-3}	4.15×10^{-5}	4.39×10^{-2}	2.5×10^{-2}
Max Error	2.36×10^{-4}	2.34×10^{-4}	4.75×10^{-10}	1.6×10^{-1}	1.59×10^{-1}	3.65×10^{-3}	3.17×10^{-3}	3.65×10^{-3}	6.59×10^{-1}	7.67×10^{-2}
RMSE	5.76×10^{-5}	4.37×10^{-5}	1.95×10^{-10}	3.43×10^{-2}	2.96×10^{-2}	1.64×10^{-3}	1.56×10^{-3}	1.39×10^{-4}	6.77×10^{-2}	3.29×10^{-2}
RRMSE	3.82%	2.85%	1.5%	11.94%	10.33%	1.82%	2.35%	3.29%	2.06%	1.62%
CC	98.71%	98.98%	99.86%	94.56%	95.32%	99.83%	99.81%	99.72%	99.85%	99.92%

the learning engine, which has been mainly employed in recent works of CMOS-memristor hybrid designs [41], [42].

During the SPICE simulation, the memristor-based block of the learning engine is implemented with the circuit presented in Fig. 5. The memristor element is constructed using the memristor SPICE model in Verilog-A with the fitting parameters obtained from the ferroelectric memristor in Table I. The analog computing blocks are implemented with the analog circuits in Fig. 6, using 180 nm analog CMOS technology. By configuring and reusing the building blocks, both the trace-based pairwise STDP and the BCPNN learning rule can be implemented with the learning engine.

In the simulation, voltage pulses with different amplitudes are used to represent a set of incoming pre- and post-synaptic spike trains. The time per synaptic update can be adjusted by applying voltage pulses with different widths. In this work, the timestep for each synaptic update operation is set to be 100 ns based on the voltage pulse width reported in [37]. The SPICE simulation step is 10 ps, therefore 10,000 SPICE simulation steps are needed to simulate one timestep. The total simulation time is 100 μs, which includes 1000 timesteps in total. During the simulation, the instantaneous current and voltage of key nodes are recorded and exported for accuracy and power analysis. Given the same incoming pre- and post-synaptic spike trains, the SPICE simulation results are compared with the reference model of the trace-based learning rules and error analysis is conducted. During the analysis, the average error, maximum error, RMSE, relative RMSE (RRMSE), and correlation coefficient (CC) are used as the main evaluation metrics. Specifically, the RRMSE normalizes the RMSE with the peak-to-peak amplitude of the reference data, while the CC reflects the correlation between the simulation results and the reference data.

As shown in Table II, the simulation results demonstrate a high degree of fit with the reference model. The nonlinear decay of the trace variables can be achieved with high accuracy. The correlation coefficient of the synaptic weights w_{ij} of the trace-based pairwise STDP and the BCPNN learning rules reach more than 99%, and the relative RMSE is less than 3%. The above results prove that the proposed learning engine can implement the trace-based learning rules with a high level of precision at the circuit level.

TABLE III
ENERGY CONSUMPTION OF THE LEARNING ENGINE FOR EACH SYNAPTIC UPDATE OF TRACE-BASED LEARNING RULES

		STDP	BCPNN
Memristor (pJ)		3.28	4.91
Analog circuits (pJ)	R2T module	2.45	6.12
	Adder	1.22	6.12
	Multiplier	-	27
	Logarithmic module	-	7.34
	Integrator	3.66	-
Total (pJ)		10.61	51.49

C. Evaluation of Energy Efficiency

In this work, power and energy reduction is the main goal. We take advantage of the nonlinear property of physical memristor devices to mimic synaptic dynamics, which can avoid sophisticated nonlinear computation compared with digital implementations. In the current hardware implementations of SNNs, the energy per synaptic operation (SOP) is often used to evaluate the energy efficiency [43], [44]. However, the definition of SOP is not unified in these implementations and is not applicable to this work. Therefore, we employ the energy per synaptic update as the evaluation metric, which is defined as the energy consumed for a complete weight update process.

The total energy consumption of the learning engine is composed of two parts, the memristor part, and the analog circuit part. To evaluate the power consumption of the memristor part, the instantaneous voltage and instantaneous current of the memristor nodes in the circuit are measured and recorded. The power consumption of the memristors for each synaptic update is then calculated by accumulating the instantaneous power. The power consumption of the analog circuit is calculated according to the operating voltage and current of the circuit modules.

Table III exhibits the energy consumed by the learning engine for every synaptic update of trace-based learning rules. In the working mode for the trace-based pairwise STDP, it takes 100 ns for each synaptic weight update, and the consumed energy is 10.61 pJ in total. The detailed power breakdown of the learning engine is visualized in Fig. 8 (a).

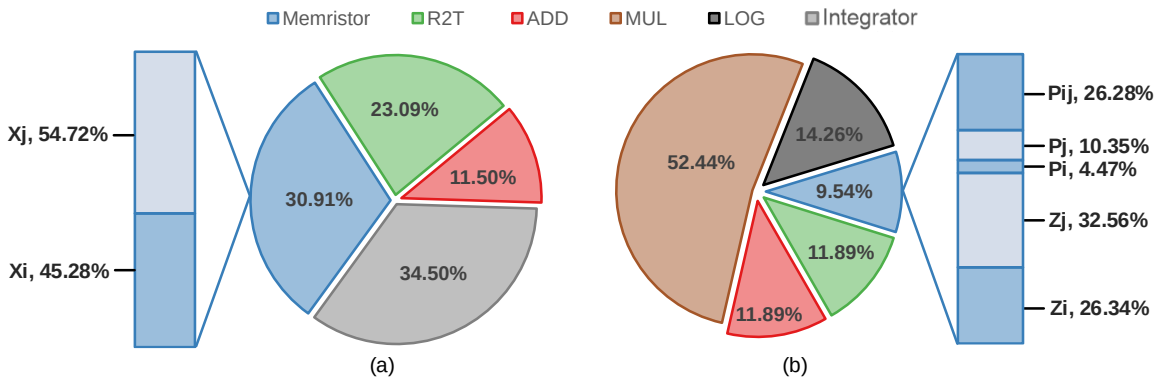

Fig. 8. (a) The power breakdown of the learning engine in the learning mode for the trace-based pairwise STDP. The memristor part is composed of two memristor nodes, representing two traces X_i and X_j. (b) The power breakdown of the learning engine in the learning mode for the BCPNN learning rule. The memristor part is composed of five memristor nodes, representing five traces Z_i, Z_j, P_i, P_j, and P_{ij}.

The memristor part accounts for approximately 30% of the total power consumption. Due to the difference between the pre- and post-synaptic spike trains S_i and S_j, the power consumption of the two memristor nodes that represent the two traces X_i and X_j also varies slightly.

When the learning engine works for the BCPNN learning rule, it takes 100 ns for each synaptic update, and the energy consumption is 51.49 pJ in total, as presented in Table III. The memristor part is responsible for less than 10% of the total power consumption, as demonstrated in Fig. 8 (b). The power consumption of the five memristors, used to represent the five traces Z_i, Z_j, P_i, P_j, and P_{ij}, varies due to the different voltage pulses applied. The analog computing blocks account for most of the power consumption, of which the multiplier accounts for more than 50%.

D. Comparative Experiments on Different Platforms

To demonstrate the advantages of the proposed learning engine in terms of performance and energy efficiency, comprehensive comparative experiments are performed on different computing platforms where trace-based learning rules are implemented. The synaptic update processes of the trace-based STDP and the BCPNN learning rules are both implemented on the CPU, the Raspberry Pi, and the FPGA and ASIC design, respectively. As shown in Table IV, for each implementation, the time consumption, average power, and energy consumption are measured and evaluated. The energy-delay product is also provided as a reference.

1) Implementation on CPU: As a general-purpose processor, the CPU is often employed for brain-like simulations. In this work, the CPU experiment is conducted on the Intel Core i7-9750H processor. The processor has six cores and its base frequency is 2.60 GHz (max turbo frequency is 4.5 GHz), along with 12 MB Intel Smart Cache. The operating system used is Microsoft Windows 10.0.19044. The synaptic update processes of the trace-based STDP and the BCPNN learning rules are implemented with Python in Anaconda Prompt, respectively.

The total running time of the simulation is recorded, and the real-time power of the CPU is measured with the HWiNFO monitor, as depicted in Fig. 9 (a). However, the operating frequency of the CPU may fluctuate significantly in a short period, which can affect the accuracy of the measurement results. Therefore, to mitigate the effect of CPU operating frequency fluctuations, the measurement results are obtained by averaging 1,000,000 tests. As shown in Table IV, for the trace-based pairwise STDP, it takes 1.33 μs on average to complete a synaptic update process, with an average energy consumption of 18.06 μJ. In terms of the BCPNN learning rule, it takes 3.26 μs on average to complete a synaptic update process, with an average energy consumption of 52.43 μJ.

2) Implementation on Raspberry Pi: As an embedded device, the Raspberry Pi can be used to train neural networks at the edge. The experiment on the Raspberry platform is performed using the Raspberry Pi 4B, equipped with a 1.5GHz ARM Cortex-A72 processor and 4GB LPDDR4 memory. The synaptic update processes of the trace-based STDP and the BCPNN learning rules are separately implemented using Python in a Linux environment.

Similarly, the simulation time and average power consumption of the Raspberry Pi are measured. To mitigate the effect of fluctuations in the working frequency of the device, the measurement results are also obtained by averaging 1,000,000 tests. As shown in Fig. 9 (b), the real-time power of the Raspberry Pi 4B is measured with the POWER-Z USB tester and exported to the laptop for further analysis. As shown in Table IV, for the trace-based pairwise STDP, the average time taken to complete a synaptic update process is 3.61 μs, with an average energy consumption of 3.48 μJ. As for the BCPNN learning rule, the average time taken to perform a synaptic update is 8.67 μs, with an average energy consumption of 8.29 μJ.

3) Implementation on FPGA and ASIC: The FPGA is a programmable hardware platform that provides great parallelism and flexibility. Besides, the ASIC design can deliver better performance and energy efficiency, thanks to their specialized and optimized circuitry.

Specifically, the trace-based pairwise STDP and the BCPNN learning rules are implemented with Verilog HDL, respectively. The full functional verification is performed, and the RTL designs are implemented on the Xilinx xc7a100tfgg484-

TABLE IV
IMPLEMENTING TRACE-BASED LEARNING RULES ON DIFFERENT COMPUTING PLATFORMS

	Platform	CPU	Raspberry Pi	FPGA	ASIC (180-nm digital)	ASIC (40-nm digital)	Memristor-based (180-nm analog)
STDP	Time per synaptic update	1.33 μs	3.61 μs	80 ns	80 ns	80 ns	100 ns
	Average power	13.8891 W	0.9656 W	121 mW	19.5 mW	1.25 mW	0.106 mW
	Energy per synaptic update	18.06 μJ	3.48 μJ	9.68 nJ	1.56 nJ	99.68 pJ	10.61 pJ
	Energy-delay product	24.02 $\mu J \cdot \mu s$	12.56 $\mu J \cdot \mu s$	0.77 $\mu J \cdot ns$	124.8 $nJ \cdot ns$	7.97 $nJ \cdot ns$	1.06 $nJ \cdot ns$
BCPNN	Time per synaptic update	3.26 μs	8.67 μs	130 ns	130 ns	130 ns	100 ns
	Average power	15.8933 W	0.9531 W	271 mW	37 mW	2.23 mW	0.51 mW
	Energy per synaptic update	52.43 μJ	8.29 μJ	35.23 nJ	4.82 nJ	290 pJ	51.49 pJ
	Energy-delay product	170.92 $\mu J \cdot \mu s$	71.87 $\mu J \cdot \mu s$	4.58 $\mu J \cdot ns$	626.6 $nJ \cdot ns$	37.7 $nJ \cdot ns$	5.1 $nJ \cdot ns$

Fig. 9. (a) Implement trace-based learning rules on the CPU and measure the real-time power with the HWiNFO monitor. (b) Implement trace-based learning rules on Raspberry Pi 4B and measure the real-time power with the POWER-Z USB tester. (c) Implement trace-based learning rules on the Xilinx xc7a100tfgg484-2 FPGA device. (d) Implement trace-based learning rules on ASIC and analyze the power consumption with Primetime given fixed incoming spike trains.

2 FPGA device, as depicted in Fig. 9 (c). Since the learning engine adopts 180 nm analog CMOS technology for analog circuits, the digital ASIC design is also synthesized in the 180 nm process as a fair comparison. The RTL designs are synthesized using Synopsys Design Compiler under typical case corner at a clock frequency of 100 MHz. The obtained gate.v, .sdc. .sdf and .spf files, together with the vcd files recording the switching activities are imported into the Prime-time for power analysis, as shown in Fig. 9 (d). Furthermore, the energy consumption for the memory access of the synaptic trace variables that are stored in the SRAM memory is also considered.

To implement the trace-based pairwise STDP, it takes 80 ns to perform a weight update process. It consumes 1.56 nJ for each synaptic update, which is 147.03 times the energy consumption of the learning engine, as shown in Table IV. In terms of the BCPNN learning rule, it takes 130 ns for each weight update. The energy consumption per synaptic update is 4.82 nJ, which is 93.61 times that of the learning engine.

As a comparison, the ASIC design is also synthesized using the more advanced 40 nm technology at 100 MHz. As shown in Table IV, for the pairwise STDP, the energy consumption per synaptic update is 99.68 pJ, which is 9.39 times that of the learning engine. As for the BCPNN learning rule, the energy for each synaptic update is 290 pJ, which is 5.63 times that of the learning engine.

It should be noted that the time taken for each synaptic update process in these implementations can be adjusted by varying their working frequency, which in turn would affect their average power. However, their energy consumption per synaptic update would still be in the same range, that's why we employ the energy per synaptic update as the main evaluation metric in this work.

E. Comparison with the State-of-the-art Work

Recently, several hardware implementations for trace-based online learning have been presented [22]–[24], which exhibit excellent energy efficiency. Table V compares the proposed learning engine with the state-of-art works that support trace-based learning rules. Since these works are oriented to different application scenarios, their reported results are normalized as the energy consumption per synaptic update to achieve a fair comparison.

TABLE V
COMPARISON WITH THE STATE-OF-THE-ART WORK

	TCASI'21 [24]	Loihi [22]	eBrainII [23]	This work
Implementation	FPGA	ASIC	ASIC	Memristor-based
Technology	-	14 nm	28 nm	180 nm
Trace-based Learning	STDP	STDP	BCPNN	STDP BCPNN
Energy per Synaptic Update	297 pJ (STDP)	120 pJ (STDP)	676 pJ (BCPNN)	**10.61 pJ** (STDP) **51.49 pJ** (BCPNN)

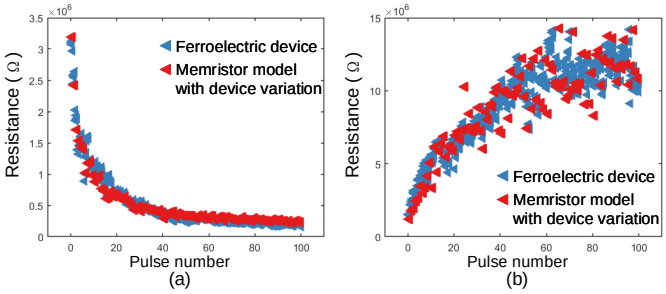

Fig. 10. Fitting the ferroelectric memristor using the memristor model with device variations. (a) The ferroelectric memristor is applied with consecutive negative voltage pulses. (b) The ferroelectric memristor is applied with consecutive positive voltage pulses.

The FPGA-based SNN processor employs trace-based STDP for the online learning of each image in the MNIST task. The reported energy consumption per synapse for the online learning of each image is 0.297 μJ/image [24]. The learning of each image is achieved by encoding the image into a spike train with a timestep length of 1000. After normalization, the energy consumption for each synaptic update is 297 pJ. For Loihi, the energy per synaptic update (pairwise STDP) is 120 pJ, which is extracted from pre-silicon SDF and SPICE simulations, in accordance with early post-silicon characterization [22]. In the eBrainII work, a lazy-update mechanism is employed in the synaptic update process [23]. To update the 101 cells of a synaptic matrix, the energy consumption is 68.27 nJ, and the average energy for each synaptic cell is 676 pJ. Compared with these works, the proposed learning engine can achieve a 27.99×, 11.31×, and 13.13× reduction of energy consumption for the synaptic update of trace-based STDP and the BCPNN learning rules, respectively.

F. Discussion on Device Variations and Future Work

In this paper, we focus on the design of the proposed learning engine for synaptic trace-based online learning and evaluating the feasibility and energy efficiency of the design. However, in reality, memristor-based structures tend to suffer from device variations, including device-to-device (D2D) variations and cycle-to-cycle (C2C) variations [45], [46]. D2D variations are spatial variations where the same voltage pulse leads to varying resistances among different memristor devices. On the other hand, C2C variations are temporal and describe changes in the behavior of a single device over time. Several works have studied the impact of device variations on the performance of spiking neural networks. In [47], an SNN simulation was conducted, revealing the network's resilience to device variations. Another study also confirmed the robustness of SNNs against device variations, using a model of the double-gate MOSFET device [48].

In this section, we model device variations by introducing Gaussian noise into the resistance $R(t)$ of the memristor model in formula (2) based on the method in [46]. The memristor resistance with device variation $R^{'}(t)$ is represented as:

$$R^{'}(t) = R(t) \times (1 + N(\sigma)) \tag{5}$$

Here, the mean of the Gaussian noise $N(\sigma)$ is 0, and the standard deviation is σ. The value of the σ is 0.1544, which is calculated based on the variation observed between the real memristor resistance and the memristor model. The fitting results of the ferroelectric memristor device using the memristor model added with Gaussian noise is visualized in Fig. 10. Taking the device variation into consideration, we performed experiments at the network level. To be specific, an SNN network based on the trace-based pairwise STDP, containing 625 excitatory neurons and inhibitory neurons, is trained and tested on the MNIST dataset. Despite the presence of device variations, the accuracy of a network-level MNIST task using memristor-based STDP remains almost the same ($92.67 \pm 0.47\%$). The network-level results show that thanks to unsupervised online learning, the SNNs adopting trace-based learning rules demonstrate good robustness against the device variations in typical physical memristors.

To clarify, this paper concentrates on the design and evaluation of the proposed learning engine. Therefore, we only conduct a basic modeling experiment of device variation.

In the future, there are several areas that can be explored for further study. Firstly, a more comprehensive and systematic analysis of the impact of device variations on the proposed design can be conducted, including both the device-to-device and cycle-to-cycle variations. Additionally, it would be valuable to investigate the influence of process scaling on the functionality of the learning engine. Furthermore, the architectural design that integrates the proposed learning engine into large-scale neuromorphic chips can be explored.

VI. CONCLUSION

In this work, a reconfigurable learning engine is proposed to implement synaptic trace-based online learning. The learning engine consists of memristor-based blocks and analog computing blocks. Based on these building blocks, the learning engine is architected and realized using memristors and 180 nm analog CMOS technology. Two trace-based learning rules, i.e., the trace-based STDP and the BCPNN learning rules, are implemented with the learning engine. The performance and energy efficiency of the learning engine is evaluated. The proposed learning engine can achieve energy consumption of 10.61 pJ and 51.49 pJ per synaptic update for the STDP and BCPNN learning rules, respectively, with a 147.03× and

93.61× reduction compared to the 180 nm ASIC counterparts, and also a 9.39× and 5.63× reduction compared to the 40 nm ASIC counterparts. Compared with the state-of-the-art work, the learning engine can reduce the energy per synaptic update by 11.31× and 13.13× for trace-based STDP and BCPNN learning rules, respectively. Furthermore, this paper verifies the feasibility and efficiency of the proposed learning engine for online learning at the synaptic level, which lays the foundation for network-level applications and provides the possibility to realize biological brain-like intelligence.

REFERENCES

[1] Y. Li *et al.*, "Oxide-Based Electrolyte-Gated Transistors for Spatiotemporal Information Processing," *Advanced Materials*, vol. 32, no. 47, p. 2003018, 2020.

[2] R. V. W. Putra and M. Shafique, "FSpiNN: An Optimization Framework for Memory-Efficient and Energy-Efficient Spiking Neural Networks," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 39, no. 11, pp. 3601–3613, 2020.

[3] K. Roy, A. Jaiswal, and P. Panda, "Towards spike-based machine intelligence with neuromorphic computing," *Nature*, vol. 575, no. 7784, pp. 607–617, 2019.

[4] X. Yang, B. Taylor, A. Wu, Y. Chen, and L. O. Chua, "Research progress on memristor: From synapses to computing systems," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 69, no. 5, pp. 1845–1857, 2022.

[5] M. R. Azghadi, B. Linares-Barranco, D. Abbott, and P. H. Leong, "A hybrid CMOS-memristor neuromorphic synapse," *IEEE transactions on biomedical circuits and systems*, vol. 11, no. 2, pp. 434–445, 2016.

[6] R. Yang, H.-M. Huang, Q.-H. Hong, X.-B. Yin, Z.-H. Tan, T. Shi, Y.-X. Zhou, X.-S. Miao, X.-P. Wang, S.-B. Mi *et al.*, "Synaptic Suppression Triplet-STDP Learning Rule Realized in Second-Order Memristors," *Advanced Functional Materials*, vol. 28, no. 5, p. 1704455, 2018.

[7] Q. Hong, R. Yan, C. Wang, and J. Sun, "Memristive circuit implementation of biological nonassociative learning mechanism and its applications," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 14, no. 5, pp. 1036–1050, 2020.

[8] W. Zhou, S. Wen, Y. Liu, L. Liu, X. Liu, and L. Chen, "Forgetting memristor based STDP learning circuit for neural networks," *Neural Networks*, vol. 158, pp. 293–304, 2023.

[9] N. Zheng and P. Mazumder, "Learning in memristor crossbar-based spiking neural networks through modulation of weight-dependent spike-timing-dependent plasticity," *IEEE Transactions on Nanotechnology*, vol. 17, no. 3, pp. 520–532, 2018.

[10] I. Boybat, M. Le Gallo, S. Nandakumar, T. Moraitis, T. Parnell, T. Tuma, B. Rajendran, Y. Leblebici, A. Sebastian, and E. Eleftheriou, "Neuromorphic computing with multi-memristive synapses," *Nature communications*, vol. 9, no. 1, pp. 1–12, 2018.

[11] T. Kim, S. Hu, J. Kim, J. Y. Kwak, J. Park, S. Lee, I. Kim, J.-K. Park, and Y. Jeong, "Spiking neural network (SNN) with memristor synapses having non-linear weight update," *Frontiers in computational neuroscience*, vol. 15, p. 22, 2021.

[12] Y. Huang, J. Liu, J. Harkin, L. McDaid, and Y. Luo, "An memristor-based synapse implementation using BCM learning rule," *Neurocomputing*, vol. 423, pp. 336–342, 2021.

[13] A. Morrison, M. Diesmann, and W. Gerstner, "Phenomenological models of synaptic plasticity based on spike timing," *Biological cybernetics*, vol. 98, no. 6, pp. 459–478, 2008.

[14] P. J. Tully, M. H. Hennig, and A. Lansner, "Synaptic and nonsynaptic plasticity approximating probabilistic inference," *Frontiers in synaptic neuroscience*, vol. 6, p. 8, 2014.

[15] 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.

[16] H. Hazan, D. Saunders, D. T. Sanghavi, H. Siegelmann, and R. Kozma, "Unsupervised learning with self-organizing spiking neural networks," in *2018 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2018, pp. 1–6.

[17] Y. Ma, B. Chen, P. Ren, N. Zheng, G. Indiveri, and E. Donati, "EMG-based gestures classification using a mixed-signal neuromorphic processing system," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 10, no. 4, pp. 578–587, 2020.

[18] N. B. Ravichandran, A. Lansner, and P. Herman, "Brain-like approaches to unsupervised learning of hidden representations-a comparative study," in *International Conference on Artificial Neural Networks*. Springer, 2021, pp. 162–173.

[19] C. Johansson and A. Lansner, "Towards cortex sized artificial neural systems," *Neural Networks*, vol. 20, no. 1, pp. 48–61, 2007.

[20] F. Fiebig, P. Herman, and A. Lansner, "An indexing theory for working memory based on fast Hebbian plasticity," *eneuro*, vol. 7, no. 2, 2020.

[21] N. Chrysanthidis, F. Fiebig, A. Lansner, and P. Herman, "Traces of Semantization, from Episodic to Semantic Memory in a Spiking Cortical Network Model," *Eneuro*, vol. 9, no. 4, 2022.

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

[23] D. Stathis, C. Sudarshan, Y. Yang, M. Jung, C. Weis, A. Hemani, A. Lansner, and N. Wehn, "eBrainII: a 3 kW realtime custom 3D DRAM integrated ASIC implementation of a biologically plausible model of a human scale cortex," *Journal of Signal Processing Systems*, vol. 92, no. 11, pp. 1323–1343, 2020.

[24] S. Li, Z. Zhang, R. Mao, J. Xiao, L. Chang, and J. Zhou, "A fast and energy-efficient SNN processor with adaptive clock/event-driven computation scheme and online learning," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 68, no. 4, pp. 1543–1552, 2021.

[25] B. Yan, Y. Yang, and R. Huang, "Memristive dynamics enabled neuromorphic computing systems," *SCIENCE CHINA Information Sciences*, 2023, online, URL: https://www.sciengine.com/SCIS/doi/10.1007/s11432-023-3739-0.

[26] Y. Lu, X. Li, B. Yan, L. Yan, T. Zhang, Z. Song, R. Huang, and Y. Yang, "In-Memory Realization of Eligibility Traces Based on Conductance Drift of Phase Change Memory for Energy-Efficient Reinforcement Learning," *Advanced Materials*, vol. 34, no. 6, p. 2107811, 2022.

[27] J. Xu, D. Wang, F. Li, L. Zhang, D. Stathis, Y. Yang, Y. Jin, A. Lansner, A. Hemani, Z. Zou *et al.*, "A Memristor Model with Concise Window Function for Spiking Brain-Inspired Computation," in *2021 IEEE 3rd International Conference on Artificial Intelligence Circuits and Systems (AICAS)*. IEEE, 2021, pp. 1–4.

[28] D. Wang, J. Xu, D. Stathis, L. Zhang, F. Li, A. Lansner, A. Hemani, Y. Yang, P. Herman, and Z. Zou, "Mapping the BCPNN Learning Rule to a Memristor Model," *Frontiers in Neuroscience*, vol. 15, 2021.

[29] D. Wang, J. Xu, F. Li, L. Zhang, Y. Wang, A. Lansner, A. Hemani, L.-R. Zheng, and Z. Zou, "Memristor-Based In-Circuit Computation for Trace-Based STDP," in *2022 IEEE 4th International Conference on Artificial Intelligence Circuits and Systems (AICAS)*. IEEE, 2022, pp. 1–4.

[30] N. Caporale and Y. Dan, "Spike timing–dependent plasticity: a Hebbian learning rule," *Annu. Rev. Neurosci.*, vol. 31, pp. 25–46, 2008.

[31] H. Wang, Z. He, T. Wang, J. He, X. Zhou, Y. Wang, L. Liu, N. Wu, M. Tian, and C. Shi, "TripleBrain: A Compact Neuromorphic Hardware Core with Fast On-Chip Self-Organizing and Reinforcement Spike-Timing Dependent Plasticity," *IEEE Transactions on Biomedical Circuits and Systems*, 2022.

[32] A. Lansner and Ö. Ekeberg, "A one-layer feedback artificial neural network with a Bayesian learning rule," *International journal of neural systems*, vol. 1, no. 01, pp. 77–87, 1989.

[33] B. Vogginger, R. Schüffny, A. Lansner, L. Cederström, J. Partzsch, and S. Höppner, "Reducing the computational footprint for real-time BCPNN learning," *Frontiers in neuroscience*, vol. 9, p. 2, 2015.

[34] Y. Yang, D. Stathis, R. Jordão, A. Hemani, and A. Lansner, "Optimizing BCPNN Learning Rule for Memory Access," *Frontiers in Neuroscience*, vol. 14, p. 878, 2020.

[35] D. B. Strukov, G. S. Snider, D. R. Stewart, and R. S. Williams, "The missing memristor found," *nature*, vol. 453, no. 7191, pp. 80–83, 2008.

[36] S. Kvatinsky, M. Ramadan, E. G. Friedman, and A. Kolodny, "VTEAM: A general model for voltage-controlled memristors," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 62, no. 8, pp. 786–790, 2015.

[37] A. Chanthbouala, V. Garcia, R. O. Cherifi, K. Bouzehouane, S. Fusil, X. Moya, S. Xavier, H. Yamada, C. Deranlot, N. D. Mathur *et al.*, "A ferroelectric memristor," *Nature materials*, vol. 11, no. 10, pp. 860–864, 2012.

[38] Y. Li, J. Chu, W. Duan, G. Cai, X. Fan, X. Wang, G. Wang, and Y. Pei, "Analog and digital bipolar resistive switching in solution-combustion-processed NiO memristor," *ACS applied materials & interfaces*, vol. 10, no. 29, pp. 24 598–24 606, 2018.

[39] S. Ruder, "An overview of gradient descent optimization algorithms," *arXiv preprint arXiv:1609.04747*, 2016.

[40] S. P. Brooks and B. J. Morgan, "Optimization using simulated annealing," *Journal of the Royal Statistical Society: Series D (The Statistician)*, vol. 44, no. 2, pp. 241–257, 1995.

[41] J. Ahmadi-Farsani, S. Ricci, S. Hashemkhani, D. Ielmini, B. Linares-Barranco, and T. Serrano-Gotarredona, "A CMOS–memristor hybrid system for implementing stochastic binary spike timing-dependent plasticity," *Philosophical Transactions of the Royal Society A*, vol. 380, no. 2228, p. 20210018, 2022.

[42] W. Wang, L. Danial, Y. Li, E. Herbelin, E. Pikhay, Y. Roizin, B. Hoffer, Z. Wang, and S. Kvatinsky, "A memristive deep belief neural network based on silicon synapses," *Nature Electronics*, pp. 1–11, 2022.

[43] E. Stromatias, D. Neil, F. Galluppi, M. Pfeiffer, S.-C. Liu, and S. Furber, "Scalable energy-efficient, low-latency implementations of trained spiking deep belief networks on spinnaker," in *2015 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2015, pp. 1–8.

[44] C. Frenkel, M. Lefebvre, J.-D. Legat, and D. Bol, "A 0.086-mm^2 12.7-pJ/SOP 64k-synapse 256-neuron online-learning digital spiking neuromorphic processor in 28-nm CMOS," *IEEE transactions on biomedical circuits and systems*, vol. 13, no. 1, pp. 145–158, 2018.

[45] S. Batool, M. Idrees, S.-R. Zhang, S.-T. Han, and Y. Zhou, "Novel charm of 2D materials engineering in memristor: when electronics encounter layered morphology," *Nanoscale Horizons*, vol. 7, no. 5, pp. 480–507, 2022.

[46] L. Li, Z. Zhou, N. Bai, T. Wang, K.-H. Xue, H. Sun, Q. He, W. Cheng, and X. Miao, "Naive Bayes classifier based on memristor nonlinear conductance," *Microelectronics Journal*, vol. 129, p. 105574, 2022.

[47] D. Querlioz, O. Bichler, P. Dollfus, and C. Gamrat, "Immunity to device variations in a spiking neural network with memristive nanodevices," *IEEE transactions on nanotechnology*, vol. 12, no. 3, pp. 288–295, 2013.

[48] S. Y. Woo, K.-B. Choi, J. Kim, W.-M. Kang, C.-H. Kim, Y.-T. Seo, J.-H. Bae, B.-G. Park, and J.-H. Lee, "Implementation of homeostasis functionality in neuron circuit using double-gate device for spiking neural network," *Solid-State Electronics*, vol. 165, p. 107741, 2020.