Hardware software co-design for leveraging STDP in a memristive neuroprocessor

IOP Publishing Neuromorph. Comput. Eng. 4 (2024) 024010 https://doi.org/10.1088/2634-4386/ad462b

NEUROMORPHIC
Computing and Engineering

OPEN ACCESS

RECEIVED
31 December 2023

REVISED
17 April 2024

ACCEPTED FOR PUBLICATION
1 May 2024

PUBLISHED
22 May 2024

PAPER

Hardware software co-design for leveraging STDP in a memristive neuroprocessor

Nishith N Chakraborty*, Shelah O Ameli, Hritom Das, Catherine D Schuman and Garrett S Rose

Department of Electrical Engineering & Computer Science, University of Tennessee, Knoxville, 1520 Middle Dr, Knoxville, TN 37996, United States of America
* Author to whom any correspondence should be addressed.

E-mail: nchakra1@vols.utk.edu, oameli@vols.utk.edu, hdas@utk.edu, cschuman@utk.edu and garose@utk.edu

Keywords: memristor, synapse, neuron, spike-timing-dependent plasticity, static, temporal, reservoir computing

Abstract

In neuromorphic computing, different learning mechanisms are being widely adopted to improve the performance of a specific application. Among these techniques, spike-timing-dependent plasticity (STDP) stands out as one of the most favored. STDP is simply managed by the temporal information of an event, which is biologically inspired. However, most of the prior works on STDP are focused on circuit implementation or software simulation for performance evaluation. Previous works also lack a comparative analysis of the performances of different STDP implementations. This study aims to provide a comprehensive assessment of STDP, centering on the performance across various applications such as classification (static and temporal datasets), control, and reservoir computing. Different applications necessitate distinct STDP configurations to achieve optimal performance with the neuroprocessor. Additionally, this work introduces an application-specific integrated circuit design of STDP circuitry. The design is based on current-controlled memristive synapse principles and utilizes 65 nm CMOS technology from IBM. The detailed presentation includes circuitry specifics, layout, and performance parameters such as energy consumption and design area.

1. Introduction

Driven by the efficiency of the human brain, neuromorphic systems are increasingly becoming favored for their effectiveness, adaptive signal processing capabilities, and minimal power consumption [1]. The main objective is to identify a computational framework that surpasses traditional Von Neumann-based computer architectures in terms of scaling, energy efficiency, and space utilization, particularly in data-intensive or resource-constrained applications. Hardware implementations of spiking neural networks (SNNs) have shown promise in this regard [2]. These systems comprise interconnected neurons and synapses that play a pivotal role in cognitive decision-making and learning processes influenced by external input.

Synapses employ diverse learning approaches, encompassing both supervised and unsupervised methods, with the latter gaining prominence in machine learning applications like automated labeling and object detection [3]. Spike-timing-dependent plasticity (STDP) represents one such technique for synaptic plasticity and learning, wherein it modulates the synaptic strength between neurons by considering the timing disparity in their output spikes. If the pre-neuron spike precedes the post-neuron spike, synaptic potentiation occurs, whereas the reverse leads to synaptic depression. The extent of weight change is directly proportional to the time gap between the spikes [4].

STDP adjusts the strength of synapses according to activity patterns, playing a crucial role in memory formation and information processing [5]. Additionally, it finds utility in artificial neural networks, guiding the training of models to replicate the behavior of biological neurons and synapses [6]. This enables the recognition of patterns and the execution of tasks akin to biological systems [7]. Moreover, SNNs equipped with STDP have showcased notable enhancements in tasks related to image, object, time-series, and language

processing [8]. Consequently, STDP holds significance in both neuroscience and artificial intelligence research.

STDP has been integrated into hardware through various means, including digital circuits [9, 10], analog circuits [11], and memristor circuits [12]. Among these, memristors stand out as promising choices for implementing STDP due to their adaptable weight storage capabilities [13]. A memristor is a two-terminal device that demonstrates analog memory properties by enabling resistance switching. Altering the resistance of a memristor is achieved by applying a voltage surpassing a specific threshold across its terminals. With compatibility with CMOS technology and non-volatile characteristics, memristors prove to be well-suited for analog computation [13, 14].

Some recent works illustrate the use of STDP in a memristive SNN. Experimental demonstration of coincidence detection using an SNN is found in [15], where the SNN consists of passively integrated metal-oxide memristive synapses connected to analog leaky-integrate-and-fire (LIF) silicon neurons in a crossbar. Through the utilization of STDP learning, the network achieves a robust capability in identifying coincidences by preferentially enhancing the synaptic strengths associated with the synchronized inputs. Wang *et al* [16] uses a diffusive memristor based on silver nanoparticles in a dielectric film to employ the stochastic LIF dynamics, along with nonvolatile memristive synapses with STDP functionality to design an SNN to experimentally demonstrate unsupervised synaptic weight updating and pattern classification. Kim *et al* [17] reports an SNN implementation with CMOS-integrated 2T1R phase-change memory devices. Ambrogio *et al* [18] documents the experimental STDP behaviors observed in hafnium-oxide (HfO_2) memristors and leverages these findings to model an SNN utilizing 1T1R discrete memristors. Matsukatova *et al* [19] addresses a major bottleneck stemming from the memristor device variations and proposes a combination of reservoir computing (RC) using volatile polyaniline memristive devices and spiking neuromorphic systems using nonvolatile parylene memristors for readout layer with STDP learning to achieve greater robustness to device variability.

In this work, we design STDP circuits for a current-controlled synapse based on HfO_2 memristors. The memristor possesses a resistance range spanning from a few kΩ to over 150 kΩ [13]. However, in our design, we exclusively utilize the low resistance states (LRSs) for their increased reliability, while steering clear of the highly variable high resistance states (HRSs) [20]. By regulating the current during the *SET* operation, we can effectively program the memristors into distinct low resistance states, addressing challenges associated with variability and limited resolution [21].

Despite the effectiveness of memristors as synapses, there is a significant requirement for optimizations in CMOS-memristor co-design. Furthermore, as the size and complexity of SNN-based applications and learning algorithms continue to grow, achieving application-specific optimization for neuromorphic hardware in terms of accuracy, latency, and energy consumption becomes intricate and time-consuming. Consequently, there is an increasing demand for an extensible simulation framework capable of conducting architectural explorations with SNNs. This approach to hardware-software co-design leverages the synergy between hardware and software to optimize and/or meet design constraints such as cost, performance, and power for the final product [22].

In this work, we employ the TENNLab Neuromorphic Computing Software Framework [23], which offers a unified interface for hardware implementations as well as a diverse range of algorithms and applications. This framework is founded on a broad model of SNNs, providing flexibility in the physical realization of the neural network. SNNs tailored for specific applications are constructed through a blend of supervised learning, employing genetic algorithms such as evolutionary optimization (EO), and unsupervised learning through STDP. The physical implementation of the SNN is achieved using RAVENS [24].

This paper makes the following significant contributions:

(a) Design of a novel STDP circuitry for a current-controlled memristive synapse.
(b) Integration of the proposed circuit behavior into an SNN model.
(c) Hardware-software co-design approach to evaluate the performance of the proposed STDP circuitry for classification and control applications using a neuromorphic framework.
(d) RC is also evaluated with our proposed STDP circuitry to benchmark the effectiveness of STDP in a neuroprocessor.

The rest of the paper is organized as follows. Section 2 provides an overview of the STDP circuitry design along with the memristive synapse. Section 3 illustrates the integration of the memristive STDP with a neuroprocessor for its performance evaluation in classification and control applications with a software framework. A comparative analysis with prior work is shown in section 4. Finally, the paper concludes in section 5. Additional results are provided in the appendix section.

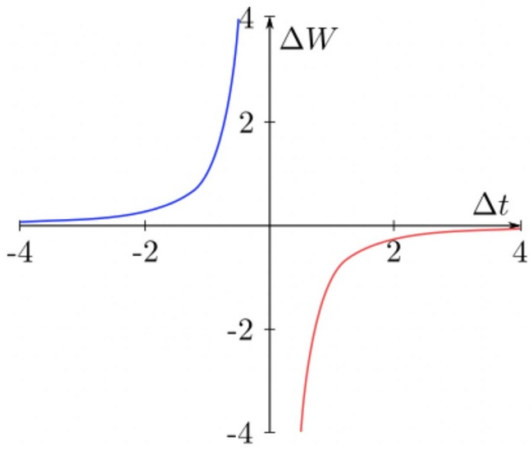


Figure 1. A typical STDP Curve illustrating exponential weight changes as the temporal proximity to the firing events increases.

2. STDP circuitry for the memristive synapse

The mathematical expression for the weight change (ΔW) in STDP can be expressed through the following equation,

$$\Delta W = \begin{cases} A_+ \cdot \exp\left(-\frac{\Delta t}{\tau_+}\right) & \text{if } \Delta t < 0 \\ -A_- \cdot \exp\left(-\frac{\Delta t}{\tau_-}\right) & \text{if } \Delta t > 0. \end{cases} \quad (1)$$

Here, Δt represents the time gap between pre- and post-synaptic spikes. If Δt is negative, the synaptic weight undergoes potentiation with a decay rate determined by τ_+; conversely, if Δt is positive, the synaptic weight undergoes depression with a decay rate determined by τ_-. The parameters A_+ and A_- are positive constants that govern the extent of potentiation and depression, respectively. Figure 1 illustrates the weight change graphically with the changing temporal difference of firing events.

In STDP, a weight update occurs when pre- and post-synaptic spikes are within a certain time window with one another. This is referred to as the STDP time window. This simply means if a pre-synaptic spike occurs before or after a certain time (in this work, clock cycles, each clock cycle having a period of 1 μs) of a post-synaptic spike, it leads to a weight change.

The STDP circuitry is designed for a HfO_2-based memristive synapse. The synapse with the STDP circuit is designed to work with an integrate-and-fire (I&F) neuron [25]. It is a mixed-signal design that uses analog current integration, followed by digital spike generation. The neuron uses a clock to generate a digital spike. The whole SNN design is synchronous, so we use a clock to manage spike timing in the local network. Further, the spikes are pulses, driven by synchronous elements like flip-flops. Thus, we are not shaping the spikes to mimic biology. This is by design as binary pulses can be transmitted efficiently through a digital network on chip.

For STDP, the synchronous nature of our spikes means that we are counting the number of clock cycles difference between pre- and post-synaptic spikes. In addition, there can be an overlap in the form of a post-synaptic spike that is generated in the same cycle as the pre-synaptic spike. In this one case where there is overlap, the difference is simply 0. So, the Δt of interest will always be a number of clock cycles, which will be dependent on the clock frequency. However, the Δt can vary based on the clock frequency.

This section delves into the circuit implementation of the synapse along with the reference generation circuit that executes current-control, along with the STDP circuitry [13, 26–28].

2.1. Current-controlled synapse circuit

Figure 2 illustrates the proposed HfO_2-based memristive synapse. The synapse takes an input spike or a pre-synaptic spike and converts it into a synaptic output current based on its weight value. In this particular design topology, instead of using the entire resistance range of the memristor, only the low-resistance states are used as weights, where the weight is inversely proportional to the resistance of the memristor. This design

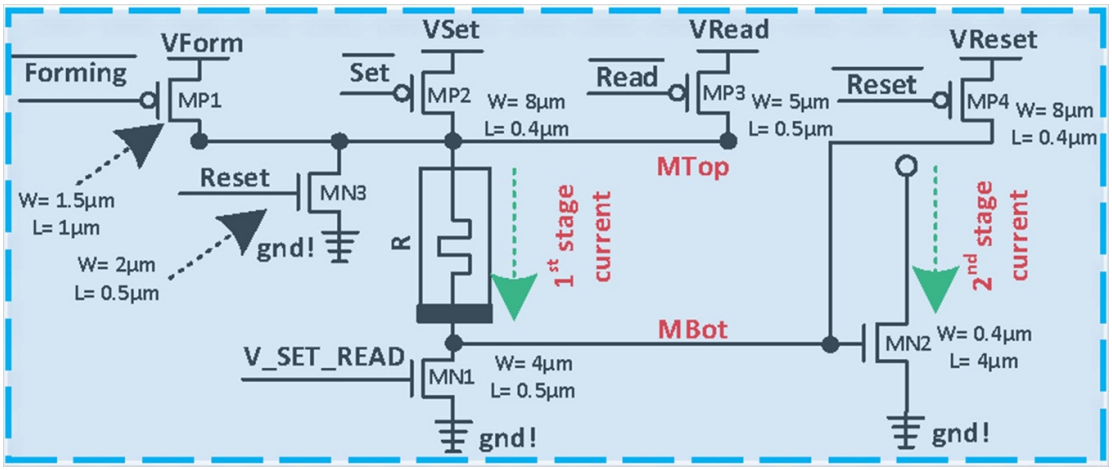


Figure 2. Circuit depicting the synapse consisting of memristor and connected transistors. Seven devices (3-NMOS, 4-PMOS) with a memristor are utilized to perform four operations such as *FORMING*, *RESET*, *SET*, and *READ*. Each of the operations uses one pair of devices. M_{N1} device is shared for *FORMING*, *SET*, and *READ* operations.

was engineered taking into account considerations for reducing both area and power while optimizing the use of the memristor.

To use only the LRS region of the memristor, certain operations are needed to be performed. The first one is *FORMING*, which is a one-time process that creates a metallic filament when a high voltage is applied across the memristor terminals. This operation sets the memristor resistance to a low-resistance state. However, to specifically program it to a particular value, a *RESET* operation is needed, which resets the memristor to a high-resistance state. Finally, a *SET* operation brings the resistance to a low-resistance state within the operating region of the synapse. During the *READ* operation, the synapse works as a weighted input to the post-synaptic neuron based on pre-synaptic input. In this design, thick-oxide transistors are used as the devices connected to the memristors need to operate in a high-voltage region.

For a standalone synapse, seven high-voltage transistors are used, including four p-type transistors and three n-type transistors. Each operation has its own p-type transistor. On the contrary, only *RESET* and *READ* operations have their dedicated n-type transistors. The remaining n-type transistor M_{N1} is shared for the *FORMING*, *SET*, and *READ* operations. A detailed timing diagram of a synapse is presented in our prior works [20, 29].

After fabrication, a one-time *FORMING* operation is performed on the synapse. During this operation, M_{P1} is activated to bring *MTop* to *VForm* (3.3 V), and *V_SET_READ* is set to 1 V to bring the voltage at *MBot* close to ground. The saturation current of M_{N1} restricts the current flowing through the forming path.

After completing the *FORMING* process, a *RESET* is necessary to program the device into a new low-resistance state. This *RESET* process is digitally managed through the utilization of transistors M_{P4} and M_{N3}. These transistors direct *MTop* to the ground and guide *MBot* towards *VReset*, inducing the device to shift to a high-resistance state.

Ultimately, to program the resistance within the operational range of the synapse, the *SET* operation is employed. This is achieved by activating M_{P2} and maintaining *V_SET_READ* within the range of 0.8 V and 1.11 V, thereby programming a resistance between 18 kΩ and 5 kΩ.

During the *READ* mode, the synapse transforms the input from the pre-synaptic neuron into a current using the resistance of the memristor. This is achieved by engaging M_{P3} and configuring *V_SET_READ* to 0.6 V. The gate voltage at M_{N1} induces a current through the memristor, and the resulting output current (2nd stage current) is determined by the gate voltage at M_{N2}, which fluctuates according to the resistance value.

2.2. Reference generation circuit

Given that the transistor M_{N1} in figure 2 is employed for *FORMING*, *SET*, and *READ* operations, precise control of the gate voltage *V_SET_READ* at M_{N1} is crucial. To accomplish this, a *V_SET_READ* generation circuit, depicted in figure 3, is designed. Upon the arrival of the pre-synaptic spike, the *READ* operation takes place, and *V_SET_READ* is established at 0.6 V using M_{P3}, M_{P4}, and M_{N1}, with *readref* set to 0.6 V. In the *FORMING* and *SET* operations, transistors M_{P1}, M_{P2}, and M_{N1} are utilized. The gate voltage at M_{P2}, denoted

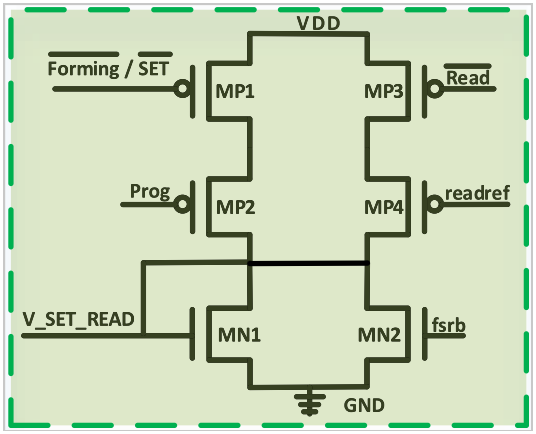


Figure 3. Current control circuit to generate *V_SET_READ* for the synapse.

as *Prog*, determines the *V_SET_READ* for *FORMING* and *SET* operations. When the synapse is not engaged in any of these operations, *V_SET_READ* is brought to 0 by the signal *fsrb* at the gate of M_{N2}.

2.3. STDP circuitry

When a spike occurs, the synapse translates it into a post-synaptic current based on the memristor's conductance. However, a single spike is not enough to change the conductance in long-term plasticity mechanisms such as STDP. For a synaptic update in STDP, both pre- and post-synaptic spikes are needed within a certain time window of each other. However, in previous work, we have explored a short-term plasticity mechanism for the synapse, where a single spike leads to a temporary change in the memristor's conductance [26].

Implementing STDP on the previously mentioned synapse involves leveraging the memristor's resistance-switching mechanism. Short reset pulses can be employed to induce a gradual rise in resistance [30]. However, it is important to note that this approach exhibits significant non-linearity, and the adjustment in weight is not directly proportional to the pulse width. This is due to the occurrence of current limiting when resistance increases [13, 26, 31].

The adjustable programming voltage, *V_SET_READ*, allows independent reduction of the memristor's resistance without requiring a reset. However, increasing the resistance in a similar manner is not feasible. Given that STDP implementation relies on resistance-switching, the unidirectional nature of this synapse's switching presents a challenge. Consequently, a two-step approach is adopted for resistance switching. When a weight update is needed, the synapse resets the memristor to a HRS and subsequently utilizes a specified set programming voltage to program the resistance to a new LRS.

For its implementation, the STDP circuit is constructed with two sub-circuits. The programming voltage generation circuit (colored blue in figure 4) is responsible for generating the programming voltage and regulating the *Prog* signal in figure 3, subsequently yielding *V_SET_READ*. The digital pulse generation circuit (yellow block in figure 4) generates *Reset* and *Set* pulses when a weight update is necessary. These two blocks act as STDP implementation control. The block diagram for the STDP circuit is illustrated in figure 4. The blocks for memristive synapse (green block in figure 4) and reference generation (brown block in figure 4) refer to figures 2 and 3 respectively. The detailed circuit diagram for the programming voltage generation and pulse generation blocks are shown in figures 5 and 6 respectively.

Figure 5 shows the circuit that controls the signal *Prog* in figure 3. Using this circuit *V_SET_READ* in figure 2 is programmed between 1.1 V to 0.8 V that sets the synaptic resistance between 5 kΩ and 18 kΩ. It is a capacitor-based design where a single capacitor (C_1) is used to store the charge corresponding to a particular synaptic resistance. Two other capacitors (C_2 and C_3) are used to dictate the depression and potentiation operations respectively [27].

Initially, the capacitor C_1 is charged to *Vn_Set*, which is used to initially set the memristor's resistance using *Vinit_Reset* and *Vinit_Set*. After that, this charge *Vn_Set* is preserved until a weight update condition is fulfilled. At the same time capacitors C_2 and C_3 are reset to *VDD* and ground respectively.

For depression operation where the post-synaptic spike precedes the pre-synaptic spike Post, *Vpost* is reset to the ground by discharging the capacitor C_2. Following the *Post* going low, transistor M_{P4} slowly charges C_2. When the pre-synaptic spike *Pre* arrives, C_1 is charged depending on the value of *Vpost*, the

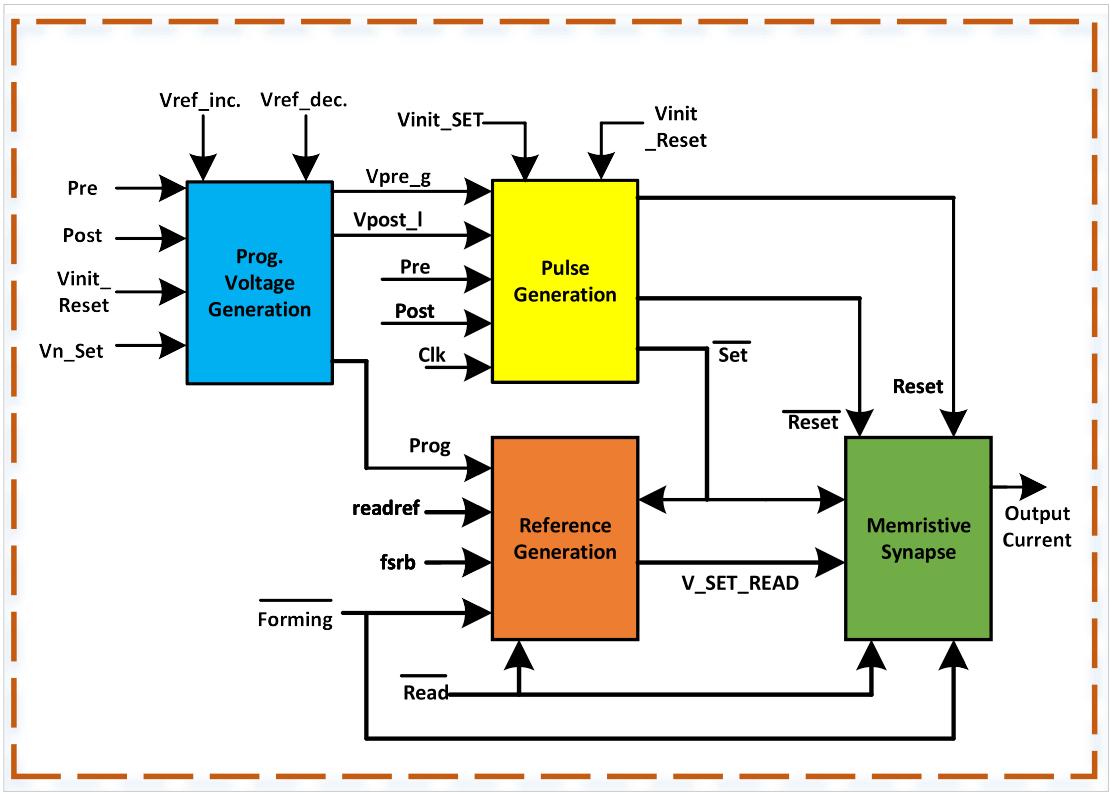


Figure 4. Block diagram illustrating the STDP implementation.

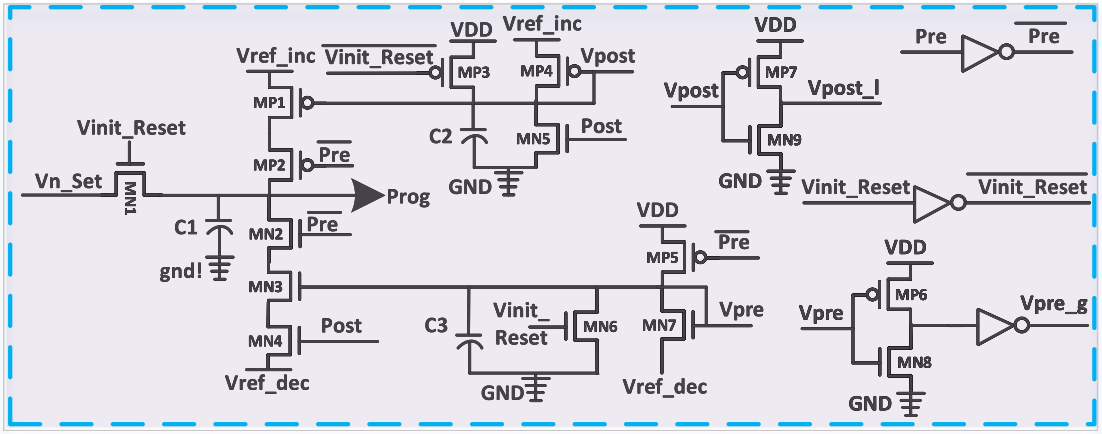


Figure 5. Programming voltage generation circuit for synaptic weight update.

voltage at C_2. This charging of C_1 increases *Prog*, in turn decreasing weight. With the increase in time difference between *Pre* and *Post*, the amount of charge increase at C_1 decreases, decreasing the amount in weight change.

The potentiation operation follows a similar strategy when the pre-synaptic spike occurs before the post-synaptic spike. When *Pre* arrives, capacitor C_3 is charged to *VDD*. After *Pre* goes back to 0, transistor M_{N7} slowly discharges C_3. With the arrival of *Post*, C_1 is discharged depending on the value of *Vpre*, the voltage at C_3. The synaptic weight is potentiated by this decrease in *Prog*. Similar to the depression operation, the increase in time difference between spikes decreases the amount of weight change.

To set the STDP time window, *Vpre* and *Vpost* signals are passed through specially sized inverters ($(W/L)_{MP6} = 120$ nm/200 nm, $(W/L)_{MN8} = 1\ \mu$m/60 nm, $(W/L)_{MP7} = 120$ nm/3μm, $(W/L)_{MN9} =$ 3 μm/60 nm) to generate *Vpre_g* and *Vpost_l*. As *Vpre* goes below 0.45 V, potentiation stops, and depression stops as *Vpost* goes above 0.25 V. This sets the STDP time window around 11 μs in both directions.

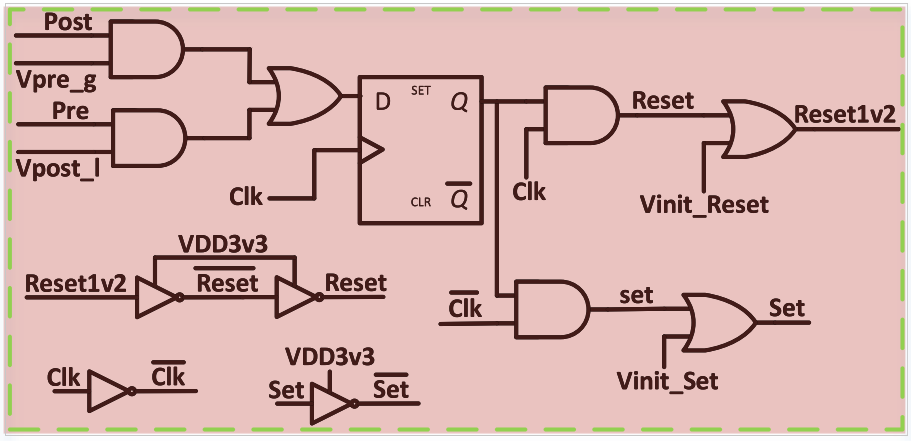


Figure 6. Digital *Reset* and *Set* pulse generation circuit.

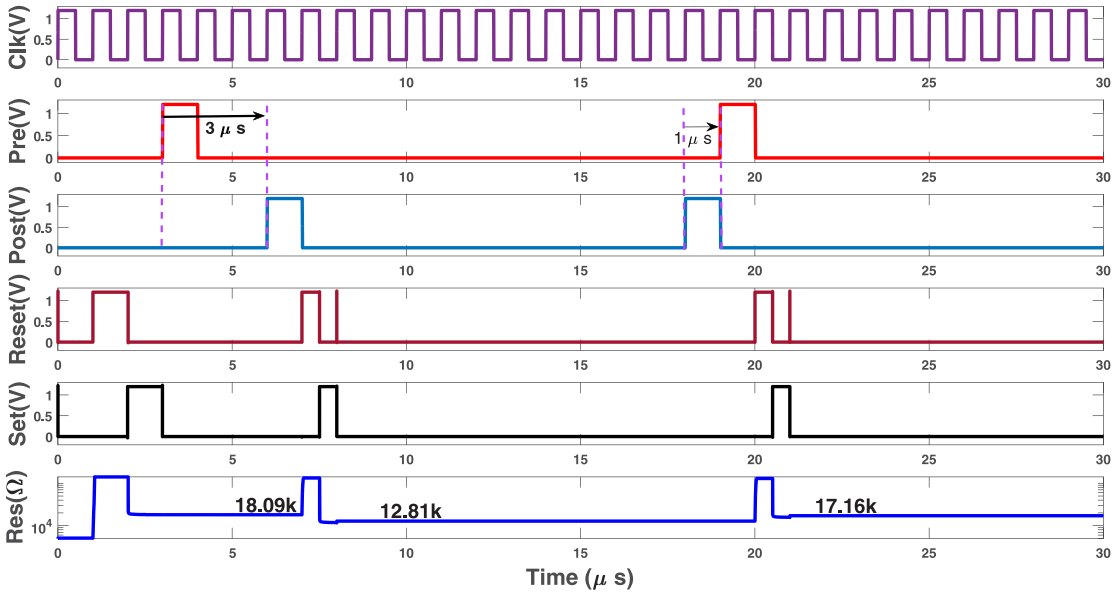


Figure 7. Timing diagram verifying the STDP functionality of the proposed circuit. The clock has a period of 1 μs. Pre and Post define the pre- and post-synaptic spikes. Res shows the resistance change according to the STDP rule.

The circuit in figure 6 generates *Reset* and *Set* pulses when a weight update condition is fulfilled. Upon the occurrence of a pre- or post-synaptic spike, this circuit checks if the spike is within the STDP time window. This is done by taking *Vpre_g* and *Vpost_l* from figure 5 and passing them through AND gates with the spikes. If the weight update condition is met, the clock cycle after the spike is divided into a *reset* and a *set* pulses, which are used in OR gates with *Vinit_Reset* and *Vinit_Set* to produce *Reset* and *Set* pulses for the synaptic weight update. These signals are converted to 3.3 V pulses using specially sized inverters, as the synapse uses these pulses [27].

2.4. Simulation results

A 65 nm CMOS process and a memristor model from [29] are used to perform simulations for functional verification of the proposed circuit.

2.4.1. Functional verification

For the simulation, the clock frequency is kept at 1 MHz. All the capacitors have 626*fF* capacitance. The values of the reference voltages *VForm*, *VReset*, *VSet*, and *VRead* are kept at 3.3 V, 2 V, 2 V and 1.2 V respectively. We also used *readref* (0.6 V), *Vref_inc* (0.6 V), and *Vref_dec* (0.1 V).

Figure 7 shows the functional verification of the STDP circuit. From the *Res* subplot, we see that after a *RESET*, the synaptic resistance is set at 18.09 kΩ using a *Vn_Set* of 0.6 V. At 3 μs, a pre-synaptic spike (*Pre*) is

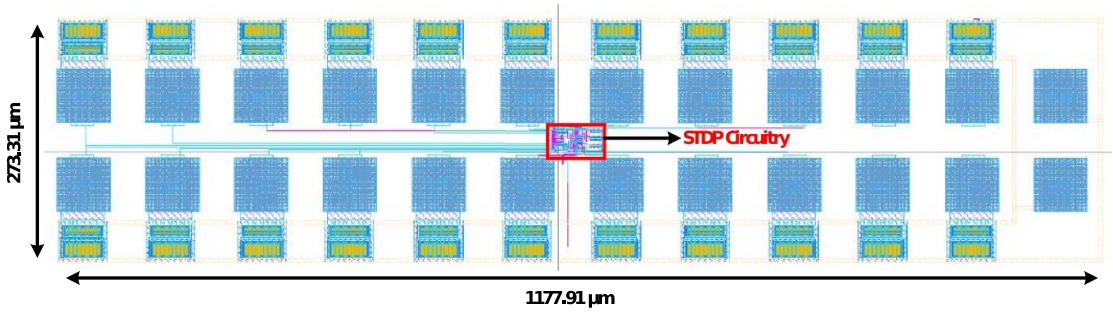


Figure 8. Layout of the test structure with proposed STDP circuitry. This is a 2 × 12 test structure with ESDs. The centered rectangular-shaped design is the STDP circuitry.

observed, succeeded by a post-synaptic spike (*Post*) at 6 μs. For this work, STDP time window is 11 μs. Since the time gap falls within the STDP time window for potentiation, the memristor's resistance decreases to 12.81 kΩ, thereby increasing the weight. This weight augmentation involves a two-step process: initially resetting the memristor and subsequently configuring it to a low-resistance state during the clock cycle following *Post*. Likewise, a depression operation is demonstrated when a post-synaptic spike occurs at 18 μs, promptly followed by a pre-synaptic spike. This sequence leads to a resistance increase to 17.16 kΩ.

Figure 8 shows the layout of the test structure containing the STDP circuitry with a 2 × 12 pin-pad structure. The circuits take up an area of 1436 μm^2 without the pin-pads, and 321 933.64 μm^2 with the pin-pads. The potentiation and depression operations consume an average power of 115.9 μW and 121 μW respectively without the pin-pads. With pin-pads, the test structure takes 164.4 μW power on average for depression operation, and for potentiation operation, the average power consumption is 156.9 μW.

The data reported in the paper is for a single memristor synapse circuit with STDP control. The number of memristors needed for an application depends on the optimum network generated by the evolutionary optimization algorithm (EONS), as discussed in section 3. As each synapse has one memristor, depending on the number of synapses in the generated network, the number of memristors needed can be calculated. The hardware cost consists of the area and power of the synapses and neurons required by the network. The average power consumption of the neuron is 166.70 μW [25], and it takes up an area of 2350 μm^2 [32].

2.4.2. STDP curve

Figure 9 illustrates how the weight change is dependent on the time interval between neuron spikes for different initial resistances. The initiation time difference between the pre-synaptic and post-synaptic spikes determines whether the change is potentiation (for negative time differences) or depression (for positive differences). The weight is inversely proportional to the memristor's resistance. For different initial resistance values programmed using different *Vn_Set* in figure 5, extensive simulations were performed using a 65 nm CMOS process in Cadence Spectre, and the change in resistance was recorded. This change was then converted to the percentage of the initial weight of the memristor.

The most significant resistance change occurs when the time difference is low, gradually diminishing as the time gap increases. The figures are consistent with (1) and a typical STDP graph shown in figure 1. However, biological STDP might vary from the ideal STDP curve shown in 1 depending on the initial weight. For our circuit, depression and potentiation are defined by the charging and discharging of capacitors C_2 and C_3 in figure 5. The charging and discharging follow an RC circuit topology where the resistors are implemented by MOSFET devices to save the area needed by large resistors. Due to that, certain non-linearity is induced in the charging and discharging process, leading to a slightly non-ideal STDP behavior. This is a trade-off between chip area and non-linearity. Using large resistors instead of MOSFETs will result in a more ideal STDP behavior.

In our circuit, the time difference between spikes is multiple of the number of clock cycles. However, the way that the STDP circuit works in figure 5 is completely analog in nature, meaning the value of *Prog* depends on the absolute time difference between pre- and post-synaptic spikes, thus leading to a continuous decrease in weight change as the time difference increases. So, although we are currently sampling the spikes every microsecond, if we change the clock frequency, it could be sampled with a higher or lower time period. But it will not change the amount of weight change for the absolute time differences between the spikes.

The four subplots also show the dependence of weight change on the value of initial resistance. At the lower resistance level shown in subplot (a), depression operation is stronger, and potentiation is weak. It is because the initial resistance is 6.99 kΩ, where the range of the synaptic resistance is between 5 kΩ and 18 kΩ.

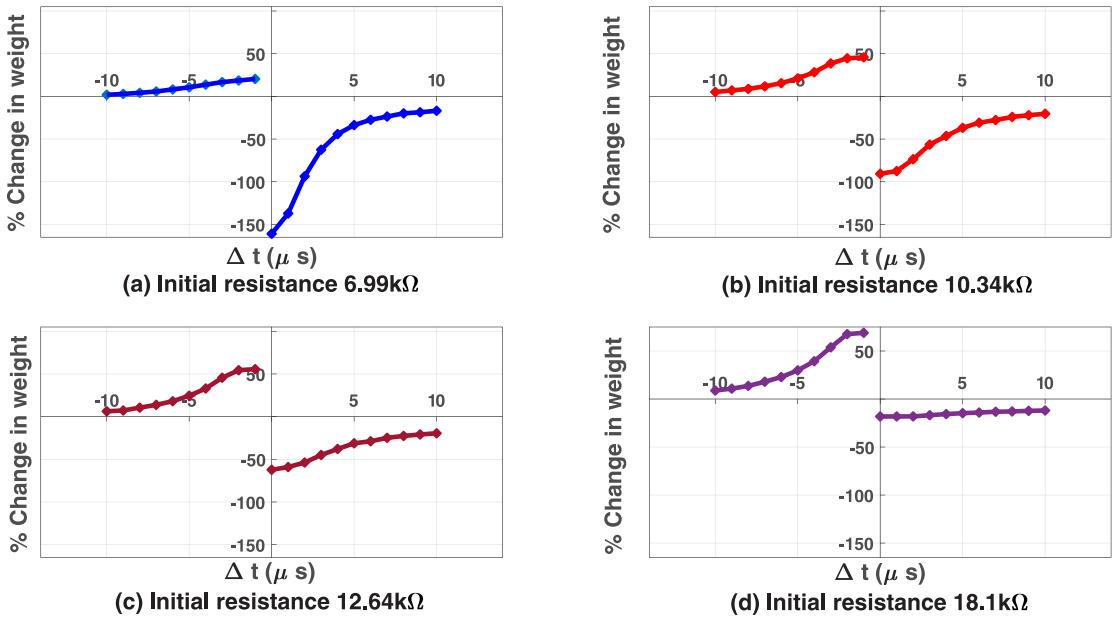


Figure 9. STDP curves for different initial resistances.

As a result, the resistance has more room to go upwards, causing stronger depression. As we increase the initial resistance, the STDP curve becomes more symmetric, denoting similar potentiation and depression rates, as shown in subplot (c). As we further increase the resistance to 18.1 kΩ, the potentiation operation becomes stronger, and depression rates become negligible (shown in subplot (d)).

3. Leveraging STDP in neuroprocessor

In this section, we evaluate the performance of the Memristive STDP Circuitry on a number of classification and control applications. We describe the components of our approach, including the training algorithms, the hardware implementation, and details of the applications environment.

3.1. Training algorithms

In this section, we describe the training approaches we used in generating the SNNs.

3.1.1. Evolutionary optimization

Evolutionary approaches have proven highly effective in designing and generating SNNs, thus in this work we employ an evolutionary training algorithm called Evolutionary Optimization for Neuromorphic systems (EONS) [33]. EONS optimizes parameters of the network like number of neurons, number of synapses, connection between neurons and parameters of the neurons and synapses over time. Schuman *et al* [34] EONS uses the traditional evolutionary approach, it randomly initializes a parent network of a given population size, and then evaluates that network using a fitness score. It uses tournament selection and mutation to produce a child population. In the next epoch, this child population becomes the parent and the process is repeated for a number of epochs. The fitness function is usually specific to the application being solved, in this work for the classification tasks, accuracy is used as the fitness, for the control tasks, the fitness score is the reward associated with completing the task.

3.1.2. Reservoir computing

A RC systems consist of a highly recurrent neural network known as a reservoir that maps input into a high-dimensional space and a readout layer that transforms the high-dimensional states within the reservoir. The reservoir layer's operation relies on the dynamics of spiking neurons and their interconnections, rather than memristors. We leverage sparse recurrent connections with synaptic delays between spiking neurons, and synaptic plasticity to enable the reservoir to capture and retain information over longer time periods. The readout layer is typically trained with a simple linear regression method [35]. Several studies have shown RC as an effective method for temporal/sequential data processing [36, 37]. In this work, we use two approaches to generate the reservoir architecture; the first is a fixed, randomly initialized SNN as the reservoir which acts as the feature extractor for the input data. The reservoir has 100 hidden neurons and 20

Table 1. STDP variations in RAVENS.

	ΔW	
Δt	Default STDP	No STDP
−4	1	0
−3	2	0
−2	2	0
−1	3	0
0	4	0
1	−4	0
2	−2	0
3	−1	0

reservoir output neurons that feed into the readout layer, we use a 10% connectivity within the reservoir. We use a simple linear classifier [38] to train the readout layer. For the second reservoir, we use EONS to optimize the internal structure of the reservoir [the number of neurons and synapses, and their connectivity], we retain 20 neurons and a linear classifier for the reservoir readout layer.

3.2. Hardware implementation

We employed RAVENS for our neuromorphic hardware implementation [24, 39]. RAVENS is a neuroprocessor for SNNs which has implementations in CPU Simulation, FPGA, and micro-controllers. RAVENS includes features such as I&F neuron model, linear charge leakage, axonal delay, adjustable resting potential, and programmable absolute and relative refractory period. Readers may refer to [39] for a detailed overview of the RAVENS neuroprocessor. The feature of RAVENS we target in this work is its STDP implementation. RAVENS allows for a lookup table that determines the synapse's weight change between neurons based on the pre-synaptic and post-synaptic spike times. Given the pre-synaptic neuron fire time t_{pre}, and the post-synaptic neuron fire time t_{post}, we calculate the time change as:

$$\Delta t = t_{pre} - t_{post}. \tag{2}$$

Then given Δt, we index into the STDP table to get the weight change ΔW on the synapse connecting the pre-synaptic and post-synaptic neurons.

Table 1 shows the STDP variations we evaluate in this work. Default STDP (DefSTDP) are the default values used for an 8-length STDP table, No STDP simply means STDP is turned off and is not used to adjust the synapse weights. Memristive STDP (MemSTDP) is the implementation of the Memristive STDP Circuitry described in section 2. According to table 2, the device generates 21 resistance values between 21 kΩ and 5 kΩ during simulation from an initial resistance of 10.3 kΩ. To ensure compatibility with the RAVENS lookup table, linear interpolation is employed to scale these resistance values to fall within the range of −10 to 10 representing the STDP weight update values.

$$\Delta W = \text{stdpMin} + \left(\frac{\text{resVal} - \text{resMin}}{\text{resMax} - \text{resMin}} * (\text{stdpMax} - \text{stdpMin}) \right). \tag{3}$$

Here, resVal is an experimentally derived resistance value within the range 21 kΩ and 5 kΩ, and ΔW refers to the corresponding STDP weight update value within the range of −10 to 10, representing the minimum and maximum STDP update values.

3.3. System software: TENNLab

To interface between the training algorithms, neuroprocessors, and classification and control applications we use the TENNLab Neuromorphic Computing Software Framework [23]. The TENNLab framework provides a common interface to multiple hardware implementations and a variety of algorithms and applications, it has both C++ and Python instances, in this work we use the Python version which comes with an integration to train SNNs for OpenAI gym environments. We also leverage the input encoding and output decoding [40] approaches in the TENNLab framework for encoding our data into input spikes and decoding output spikes into classification labels or control actions.

3.4. Experimental setup

To evaluate the performance of the memristive STDP circuitry, we conducted experiments applying it to classification and control applications. We compared its effectiveness against the default STDP and no STDP scenarios in the RAVENS neuroprocessor. The evaluation included training and testing networks using EO

Table 2. Memristive STDP circuitry in RAVENS.

	ΔW	
Δt	Resistance (kΩ)	stdp value
−10	12.4	0
−9	12.6	0
−8	12.8	0
−7	13.2	0
−6	13.5	0
−5	14.1	1
−4	15.1	2
−3	16.1	3
−2	17.9	6
−1	19.3	7
0	19.7	8
1	5.6	−9
2	5.7	−9
3	6.3	−8
4	7.4	−6
5	8.1	−6
6	8.7	−5
7	9.1	−4
8	9.4	−4
9	9.6	−4
10	9.8	−3

Table 3. Experiments summary.

No.	Task	Training algorithm	Test subjects	Networks generated	Total tests
I	Classification	EONS	4 Datasets	200, 100	2100
II	Classification	Fixed Reservoir, Evolved Reservoir	3 Datasets	200	1800
III	Control	EONS	4 Applications	20	320

and RC algorithms. For the classification problem, we trained 100 networks using the EEG Emotion dataset [41, 42] and 200 networks each for Iris, Wine and breast cancer datasets [38, 43]. It is worth noting that the EEG Emotion dataset is temporal, while the other three datasets are static. All classification datasets were trained using an evolutionary approach (EONS) and the two RC approaches (a fixed reservoir network, and an evolved reservoir network). Additionally, for each control application, we trained 20 networks using EONS. Both the evolutionary and reservoir approaches involved random initialization, resulting in varied network performance. Consequently, we generated multiple networks for each test case, yielding a total of 4200 different trained networks across all test cases. A summary of all conducted experiments is illustrated in table 3. Furthermore, detailed information about the classification datasets and control applications considered for the performance evaluation of STDP in the Neuro-Processor is provided in tables 4 and 5.

3.4.1. Classification

For the classification problems, the primary objective was to maximize the accuracy on the training set. Each network begins its simulation by processing a data sample, and ultimately producing a classification label. This label is determined either through a network trained by EONS or via the readout layer of a reservoir. Each dataset is split into training and testing samples, with the training dataset used for network training and the test dataset employed for subsequent evaluation. We evaluate three simple datasets from the UCI machine learning repository [43], and one timeseries dataset [42]. Table 4 provides detailed features of each dataset.

3.4.2. Control applications

Control applications involve the implementation of algorithms or strategies aimed at regulating and manipulating system behavior to achieve predefined fitness scores or optimize specific performance criteria. Within these applications, each action is assigned a reward value, determining its contribution to the overall control task objective. The cumulative reward obtained from executing multiple actions to achieve the task is referred to as the fitness score. These applications commonly utilize feedback mechanisms and decision-making processes to steer the system towards desired outcomes, thereby contributing to the overall

Table 4. Classification datasets.

Dataset	Instances	Features	Outputs	Time-steps
Iris	150	4	[0,1,2]	—
Wine	178	13	[0,1,2]	—
Breast cancer	569	30	[0,1]	—
EEG emotion	2132	2	[0,1,2]	750

Table 5. Control applications.

Application	Observations	Actions
Polebalance	4	2
Bowman	9	2,3
Cart-Pole	4	2
Lunar Lander	8	4

control and optimization of system behavior. We evaluate four control applications, which are outlined below, and detail their respective observations and actions in table 5.

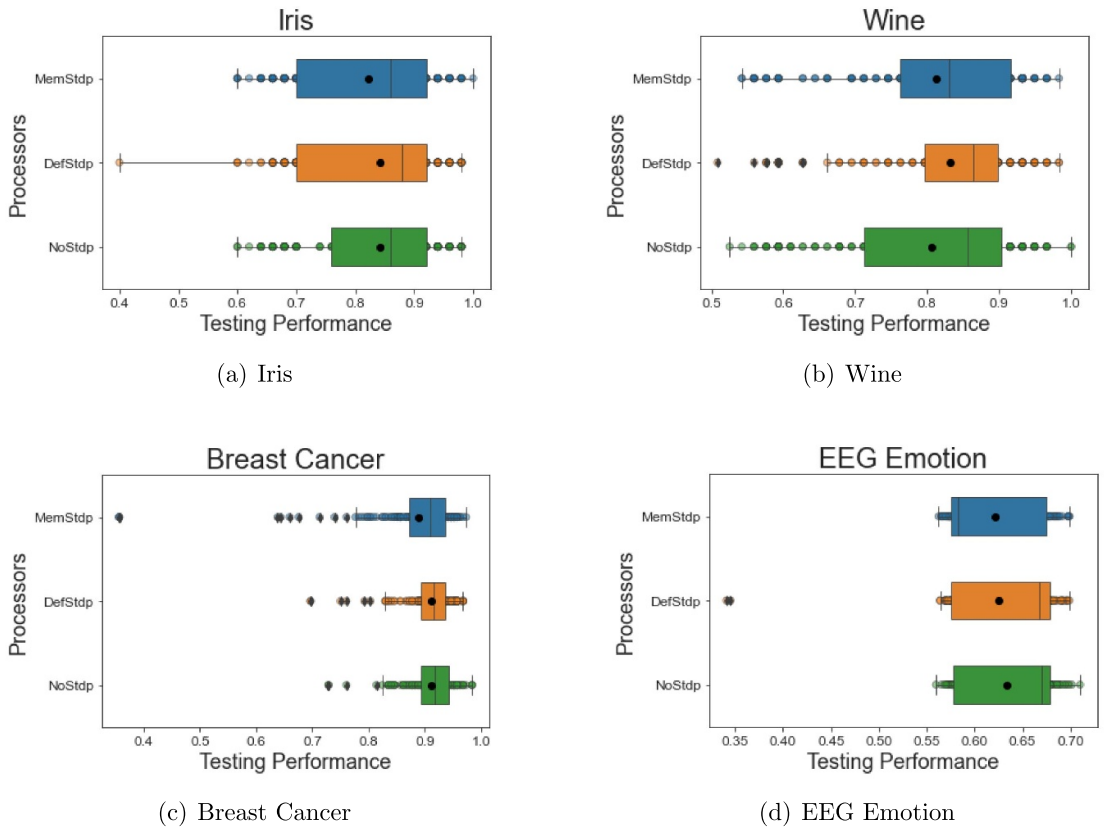
- TENNLab Control Applications: these applications were internally developed for TENNLab control baseline evaluations. They were all developed using C++.
  - Pole Balance: in this application, imagine a virtual scenario with a cart positioned on a track, supporting a standing pole. Your choices are limited to pushing the cart either to the left or the right. The aim is to navigate the cart strategically, preventing collisions with the environment's walls and maintaining the pole's upright position. Your ultimate goal is to extend the time the cart avoids hitting the wall while ensuring the pole remains stable, for a cumulative duration of up to five minutes.
  - Bowman: in this application, envision a bowman situated at the bottom of the screen, tasked with moving his bow across the playing field in both clockwise and counterclockwise directions to target birds in flight. Your actions involve determining the direction to move the bow and when to shoot arrows. The primary goal is to optimize the number of birds successfully shot while concurrently minimizing the usage of arrows.
- OpenAI Gym Applications: these applications were developed and maintained by OpenAI Gym [44].
  - CartPole-v1: this is the OpenAI version of the pole balance application discussed earlier. In this application, there's a cart accompanied by a standing pole in an upright position. The challenge is to navigate the cart horizontally while maintaining the pole's upright stance. The available actions involve striking the pole either from the left or the right.
  - Lunar-Lander-v2: the aim of this application is to oversee the descent of a spaceship toward the surface. The actions include activating the left, right, or main engine, or refraining from firing any engine.

3.5. Results

In this section, we go over the results of the training algorithms and discuss the SNNs that performed the best in our simulation. We take a closer look at the outcomes, examining how the different hardware implementations fared and highlighting the configurations of SNNs that stood out during the simulation.

3.5.1. Classification

(i) EONS: figure 10 shows the performance of the EONS-trained network on the classification tasks. Three static and one temporal dataset are considered to evaluate the impact of STDP in neuroprocessor. The accuracy is scaled from 0 to 1, where the highest value denotes the better performance of the processor. Figure 10(a) shows the testing performance of the Iris dataset with three different test cases such as (i) blue: neuroprocessor with memristive-Mem-STDP, (ii) orange: neuroprocessor with default or digital-Def-STDP, and (iii) green: neuroprocessor without-No-STDP. According to figure 10(a), the MemStdp shows optimized accuracy distributions compared to DefStdp. However, the NoStdp configuration shows better performance than others. Figure 10(b) shows the testing performance of the Wine dataset, where DefStdp shows optimized data distributions compared to other test cases. In addition, MemStdp shows better performance distribution compared to NoStdp. Figure 10(c) shows the testing performance on the breast cancer dataset. DefStdp performed better than other test cases.



(a) Iris (b) Wine

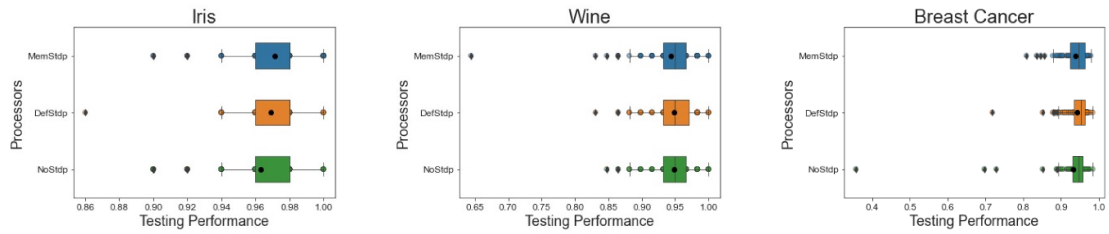
(c) Breast Cancer (d) EEG Emotion

Figure 10. Performance evaluations of EONS-trained networks on classification dataset are illustrated. The mean accuracy for each dataset is shown as a black dot. (a) shows the testing accuracy of three different processors with analog STDP (MemStdp), default or digital STDP (DefStdp), and without STDP (NoStdp). (b) and (c) show the testing performance of the Wine and breast cancer datasets respectively. (d) shows the testing performance of the EEG Emotion, which is a temporal dataset.

Finally, the testing performance on EEG Emotion figure 10(d) shows, that the performance of all test cases is similar with a similar data distribution. From the results, different datasets exhibit enhanced performance with specific design configurations, emphasizing the need for tailored solutions. The memristive device consistently achieves comparable accuracies across all datasets, showcasing its efficacy in comparison to other processor types.

(ii) Evolved Reservoir: in figure 11, we thoroughly examined how EONS evolved reservoir networks performed across various classification datasets. We evaluated their accuracy on three static datasets (Iris, Wine and breast cancer), both in training and testing scenarios, providing a detailed understanding of their capabilities. We exclude the temporal dataset from this experiment due to resource constraints. Looking specifically at the Iris dataset figure 11(a), each neuroprocessor shows commendable performance. MemSTDP stands out with an impressive mean accuracy of around 98%, slightly surpassing DefSTDP and NoSTDP, which achieve approximately 97% and 96%, respectively. While these differences are subtle, it is important to highlight the overall consistency across the three neuroprocessors. Discussing the Wine and breast cancer datasets (figures 11(b) and (c)), we notice a similar trend. Each neuroprocessor demonstrates comparable accuracy distributions with similar maximum and minimum accuracies. This consistency emphasizes the adaptability of STDP implementations in the evolved reservoirs, showcasing their reliability across different static datasets in classification tasks.

(iii) Randomly Initialized Reservoir: the outcomes for the randomly initialized reservoir are illustrated in figure 12. We conducted tests on three static datasets and the EEG Emotion temporal datasets. Again, the performance of each neuroprocessor varies across these datasets. In the case of both the Iris and Wine datasets, we observe that NoSTDP slightly outperforms Default or Digital-Def-STDP and Memrisitive-Mem-STDP. Specifically, in figure 12(a), all neuroprocessors exhibit mean accuracies of approximately 85%. For the Wine dataset (figure 12(b)), NoSTDP outperforms the other processors with a mean accuracy of 63%, DefSTDP achieves a mean accuracy of 57%, while MemSTDP has an



(a) Iris (b) Wine (c) Breast Cancer

Figure 11. Performance assessment of EONS Evolved Reservoir Networks on classification datasets. The mean accuracy for each dataset is shown as a black dot.

accuracy of 55%. For the breast cancer dataset shown in figure 12(c), networks generated for MemSTDP shows better accuracy distributions, when compared to the DefSTDP and NoSTDP Neuroprocessors. MemSTDP networks show a maximum accuracy of 90%, a minimum accuracy of 65% and a mean accuracy of 75%. Both DefSTDP and NoSTDP show maximum and minimum accuracies of ≈90% and 58% respectively. All networks across each neuroprocessor achieve a mean accuracy of ≈75%. Finally, the EEG dataset figure 12(d) shows a wide accuracy distribution across each processor, with NoSTDP slightly outperforming the others with a max accuracy of 70%, a minimum accuracy of 35% and a mean accuracy of 75%.

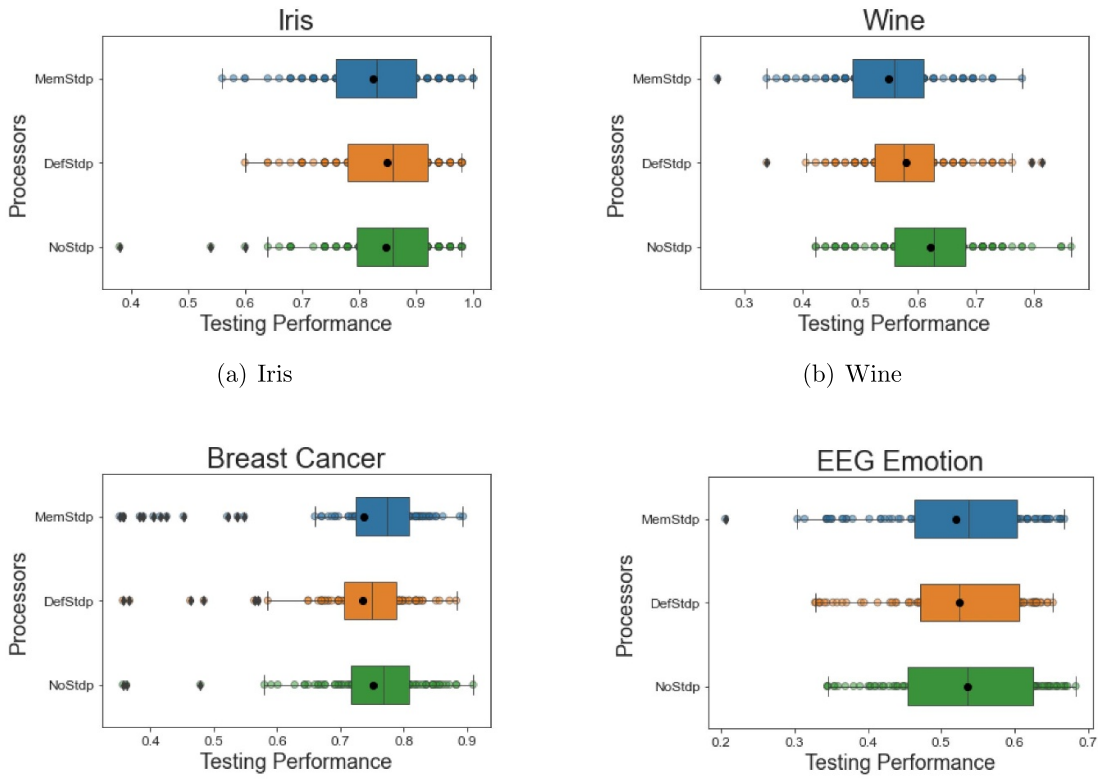
In summary, looking at how different training algorithms performed on four different datasets for our classification task, we noticed some common patterns. The best performing STDP neuroprocessor appears to be contingent upon the specific dataset. Notably, the memristive STDP consistently demonstrates comparable accuracies across all scenarios and, in certain instances, even outperforms alternative STDP implementations, including some cases where it produces networks with more consistent performances as indicated by a narrower distribution in figures 10(b) and (d). These results underscore the nuanced relationship between STDP implementations and dataset characteristics, emphasizing the importance of tailoring the approach to the specific requirements of the task at hand.

3.5.2. Control

We utilized the Evolutionary Optimization (EONS) training algorithm to evaluate the performance of three distinct neuroprocessors in control applications. The results, illustrated in figure 13, showcase the performance of networks generated by EONS. For each application, the fitness score refers to a reward associated with control actions, as defined in section 3.4.2. The final fitness score is the overall reward obtained from performing a series of actions to achieve the control task objective. Similar to the classification tasks, we conducted three different test cases: (i) blue represents the neuroprocessor with memristive-Mem-STDP, (ii) orange for the neuroprocessor with default or digital-Def-STDP, and (iii) green for the neuroprocessor without -No- STDP.

The outcomes of the control applications present an intriguing and diverse narrative, with each neuroprocessor exhibiting unique behaviors based on the specific application. In figures 13(a) and (d), we observe that the inclusion of STDP negatively impacts the performance of the application. However, for NoSTDP, there is a widely varied distribution for both the Polebalance and Lunar Lander applications. In contrast, MemSTDP and DefSTDP demonstrate a narrower distribution of fitness scores across all networks. Concerning CartPole (figure 13(c)), each neuroprocessor performs relatively well, with DefSTDP showing less variation across the fitness of other networks. MemSTDP achieves a maximum fitness of 45, outperforming DefSTDP, and the mean fitness score across each neuroprocessor is approximately 35. For Bowman (figure 13(b)), the inclusion or exclusion of STDP results in similar fitness levels. Notably, networks without STDP exhibit a less varied distribution, indicating that more networks in this category have fitness scores within the Interquartile Range (IQR) of 0.40 to 0.55. In contrast, other processors show a more diverse distribution, with an IQR of 0.26 to 0.55 for MemSTDP and an IQR of 0.35 to 0.58.

As observed in section 3.5.1, the results suggest that various implementations of STDP in neuromorphic hardware may offer superior performance in specific applications compared to others.



(a) Iris (b) Wine

(c) Breast Cancer (d) EEG Emotion

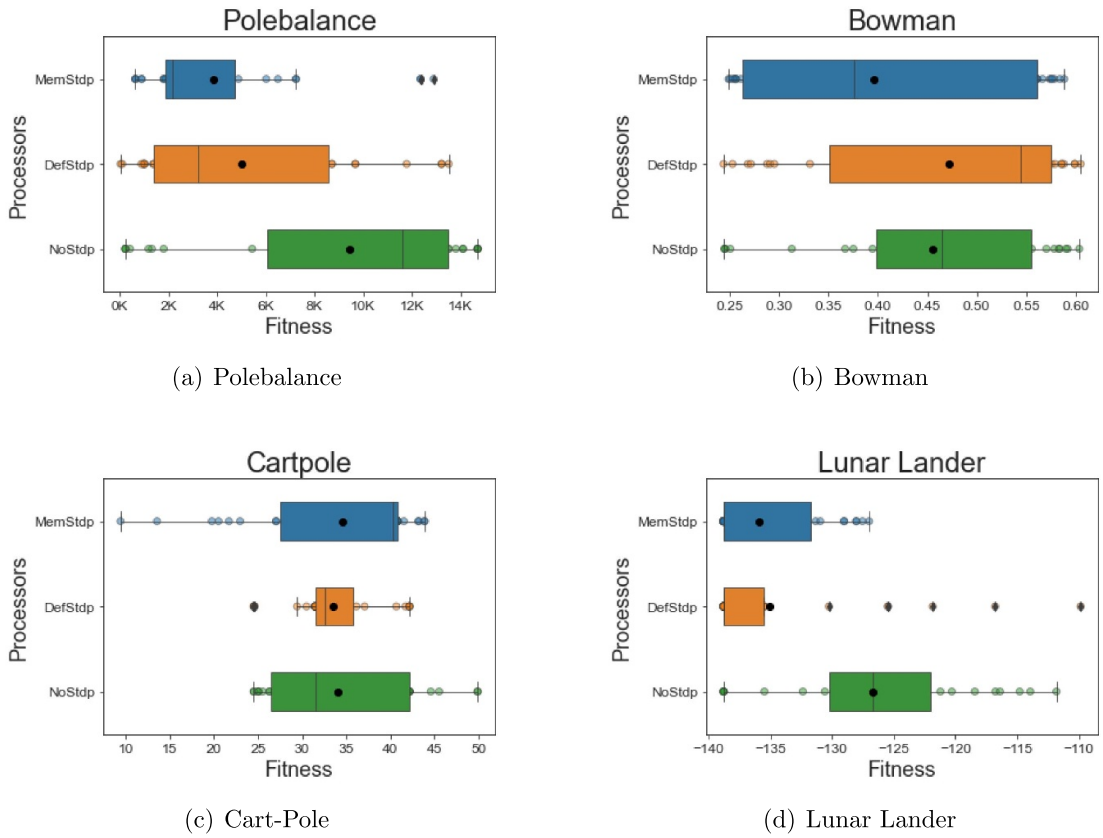
Figure 12. Performance evaluations of Randomly-Initialized Reservoir Networks on classification datasets are illustrated. The mean accuracy for each dataset is shown as a black dot.

4. Comparison with prior work

In this work, different classification and control tasks are observed to optimize the hardware design for better performance. Hardware design is laborious and time-consuming. The performance of applications mostly depends on the efficiency of hardware design. Due to that, a software-hardware co-design approach can optimize the hardware design effort with a better performance of applications. Here, three test cases are considered to determine the best design scenario for a specific application. Table 6 shows a comparison with prior works. Most of the prior work utilized the default STDP curve to emulate the characteristics of STDP in a software framework or hardware platform. J. P. Mitchell *et al* presents a work to utilize the STDP for learning, where they only include the default STDP with control and classification problems. They did not include any temporal dataset and RC to evaluate the performance of STDP [2]. Another research group presented the utilization of default STDP with static and temporal dataset evaluation [45]. However, there was no RC for evaluating STDP in a neuromorphic setup. Wang *et al* presented a memristive-based co-design for AI to leverage the STDP for static and temporal applications [46]. There was no analysis on control application and RC.

In [47], researchers consider a rat's brain for emulation of STDP to deal with static and temporal datasets. Another research group proposed a hardware-software co-design for AI systems, where they utilized STDP for learning. Here, they evaluate the performance of static datasets to observe the performance of the system with STDP. However, the temporal dataset and RC were missing in their analysis to evaluate the effect of STDP. In [48], the authors presented a work on a framework with default STDP characteristics. Classification and control tasks were considered to evaluate the performance of STDP mechanism.

Finally, our work considered three test cases such as a neuromorphic framework RAVENS without STDP, with default STDP, and with memristive STDP. In addition, static and temporal datasets are considered for the evaluation. As we know, biological components like synapses and neurons are utilizing temporal data for computation. Due to that, our analysis considers the temporal dataset to get more effective results with the STDP mechanism. Nowadays, RC also getting attention for its low maintenance costs via sparse connectivity and a single output layer, which is built with a simple memristive crossbar. All possible structures like traditional classification tasks and control tasks with EONS are considered here. Moreover, RC is also



(a) Polebalance (b) Bowman

(c) Cart-Pole (d) Lunar Lander

Figure 13. Performance of EONS-trained networks on control applications. The mean accuracy for each dataset is shown as a black dot.

Table 6. Performance validation of STDP with prior works.

Reference	[2]	[45]	[46]	[47]	[48]	[49]	This work
Default STDP	Yes	Yes	No	No	Yes	Yes	Yes
Memristive STDP	No	No	Yes	No (Rat's brain)	No	No	Yes (HfO2-based)
Technology	—	—	—	—	—	—	65 nm CMOS
Static dataset	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Temporal dataset	No	Yes	Yes	Yes	No	No	Yes
Application: classification	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Application: control	Yes	No	No	No	No	Yes	Yes
Reservoir computing	No	No	No	No	No	No	Yes

considered for the validation of STDP in neuromorphic systems. An application-specific integrated circuit version of the STDP is considered, which is built with HfO_2-based memristive devices and 65 nm CMOS technology from IBM 10LPe. Different tasks show their best performance with different design combinations. A dynamically adaptable design can select or deselect STDP blocks to get the best performance from the neuromorphic system.

5. Conclusion and future work

In this work, a current-controlled memristive STDP is utilized to evaluate the performance of applications in neuroprocessor. Current controlled synaptic STDP is more reliable and has lower energy compared to other design techniques. STDP is designed and evaluated in the Cadence Spectre with 65 nm CMOS technology. Then a software framework called RAVENS is trained and optimized based on EONS, where the behavioral model of STDP is incorporated into the framework. Three different test cases of STDP are observed with various applications. According to the observation, different application shows better performance with different design options. Based on the evaluation result, we are optimizing the hardware design to reduce the design costs of the silicon area and labor. Even we can predetermine the necessity of STDP for particular applications with software frameworks. As a future work, our designed chip will be tested for more

applications to get a better insight into whether the STDP is useful for a particular application or not. If the STDP is useful then, what kind of STDP is beneficial: default or memristive version. It is also possible to calculate the latency and energy from cadence and incorporate them with the event in the software framework to calculate the total energy and latency for each task. The software framework will be updated with additional parameters in future work.

Data availability statement

All data that support the findings of this study are included within the article (and any supplementary files).

Acknowledgments

This material is based in part on research sponsored by Air Force Research Laboratory under Agreement FA8750-21-1-1018. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of Air Force Research Laboratory or the U.S. Government. This material is based upon work supported by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research, under Award Number DE-SC0022566.

Appendix. Summary of results

Table A1. EONS generated networks for classification.

Device	Dataset	Testing accuracy	Neurons	Synapse
Memristive STDP	Iris	0.82	12.23	14.61
	Wine	0.81	29.51	21.74
	Breast cancer	0.89	62.03	22.2
	EEG emotion	0.62	23.69	21.43
Default (Digital) STDP	Iris	0.84	12.41	15.4
	Wine	0.83	29.52	22.24
	Breast cancer	0.91	62.08	23.85
	EEG emotion	0.62	23.63	21.05
No STDP	Iris	0.84	12.19	15.39
	Wine	0.80	29.4	21.57
	Breast cancer	0.91	62.04	23.22
	EEG emotion	0.63	23.52	20.12

Table A2. Randomly-initialized reservoir networks for classification.

Device	Dataset	Testing accuracy	Neurons	Synapse
Memristive STDP	Iris	0.82	128	1620.36
	Wine	0.55	146	2112.06
	Breast cancer	0.74	180	3212.32
	EEG emotion	0.52	140	1941.23
Default (Digital) STDP	Iris	0.85	128	1620.36
	Wine	0.58	146	2112.06
	Breast cancer	0.73	180	3212.32
	EEG emotion	0.52	140	1941.23
No STDP	Iris	0.84	128	1620.36
	Wine	0.62	146	2112.06
	Breast cancer	0.75	180	3212.32
	EEG emotion	0.53	140	1941.23

Table A3. Evolved reservoir networks for classification.

Device	Dataset	Testing accuracy	Neurons	Synapse
Memristive STDP	Iris	0.97	28.3	19.52
	Wine	0.94	46.21	21.58
	Breast cancer	0.94	80.08	27
Default (Digital) STDP	Iris	0.97	28.37	18.88
	Wine	0.95	46.07	20.31
	Breast cancer	0.94	80.08	27.56
No STDP	Iris	0.96	28.33	18.98
	Wine	0.95	46.11	21.58
	Breast cancer	0.93	80.09	28.29

Table A4. EONS generated networks for control.

Device	Application	Fitness	Neurons	Synapse
Memristive STDP	Polebalance	3858.61	12.1	17.6
	Bowman	0.4	23.53	21
	Cart-Pole_v1	34.57	11	12.43
	Lunar-Lander_V2	−135.95	20.17	7.2
Default (Digital) STDP	Polebalance	4990.98	11.6	15.27
	Bowman	0.47	23.53	20.37
	Cart-Pole_v1	33.51	10.93	11.6
	Lunar-Lander_V2	−135.01	20.1	7.07
No STDP	Polebalance	9421.23	12	17.2
	Bowman	0.45	23.8	22.73
	Cart-Pole_v1	34	11.33	12.36
	Lunar-Lander_V2	−126.67	20.43	14.97

ORCID iDs

Nishith N Chakraborty https://orcid.org/0009-0003-0287-0042
Shelah O Ameli https://orcid.org/0009-0006-1832-8516
Hritom Das https://orcid.org/0000-0003-2548-8754
Catherine D Schuman https://orcid.org/0000-0002-4264-8097
Garrett S Rose https://orcid.org/0000-0003-3070-4087

References

[1] Parmar V, Kingra S K and Suri M 2018 *J. Phys. D: Appl. Phys.* **51** 454004
[2] Mitchell J P, Bruer G, Dean M E, Plank J S, Rose G S and Schuman C D 2017 Neon: neuromorphic control for autonomous robotic navigation *2017 IEEE Int. Symp. on Robotics and Intelligent Sensors (IRIS)* pp 136–42
[3] Nair V and Clark J 2004 An unsupervised, online learning framework for moving object detection *Proc. 2004 IEEE Computer Society Conf. on Computer Vision and Pattern Recognition, 2004. CVPR 2004.* vol 2 p II
[4] Caporale N and Dan Y 2008 *Annu. Rev. Neurosci.* **31** 25–46
[5] Sjöström P J and Häusser M 2006 *Neuron* **51** 227–38
[6] Maass W, Natschläger T and Markram H 2002 *Neural Comput.* **14** 2531–60
[7] Goodfellow I, Bengio Y and Courville A 2016 *Deep Learning* (MIT Press)
[8] Das A, Pradhapan P, Groenendaal W, Adiraju P, Rajan R T, Catthoor F, Schaafsma S, Krichmar J L, Dutt N and Van Hoof C 2018 *Neural Netw.* **99** 134–47
[9] Foshie A Z, Chakraborty N N, Murray J J, Fowler T J, Ara Shawkat M S and Rose G S 2021 A multi-context neural core design for reconfigurable neuromorphic arrays *2021 IEEE Computer Society Annual Symp. on VLSI (ISVLSI)* pp 67–72
[10] Cassidy A, Andreou A G and Georgiou J 2011 A combinational digital logic approach to STDP *2011 IEEE Int. Symp. of Circuits and Systems (ISCAS)* pp 673–6
[11] Tanaka H, Morie T and Aihara K 2009 *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.* **E92.A** 1690–8
[12] Linares-Barranco B, Serrano-Gotarredona T, Camuñas-Mesa L, Perez-Carrasco J, Zamarreño-Ramos C and Masquelier T 2011 *Front. Neurosci.* **5** 26
[13] Weiss R, Das H, Chakraborty N N and Rose G S 2022 STDP based online learning for a current-controlled memristive synapse *2022 IEEE 65th Int. Midwest Symp. on Circuits and Systems (MWSCAS)* pp 1–4
[14] Mannan Z I, Kim H and Chua L 2021 *Sensors* **21** 644
[15] Prezioso M, Mahmoodi M R, Bayat F M, Nili H, Kim H, Vincent A and Strukov D B 2018 *Nat. Commun.* **9** 5311
[16] Wang Z *et al* 2018 *Nat. Electron.* **1** 137–45

[17] Kim S *et al* 2015 Nvm neuromorphic core with 64k-cell (256-by-256) phase change memory synaptic array with on-chip neuron circuits for continuous in-situ learning *2015 IEEE Int. Electron Devices Meeting (IEDM)* pp 17.1.1–4
[18] Ambrogio S, Balatti S, Milo V, Carboni R, Wang Z-Q, Calderoni A, Ramaswamy N and Ielmini D 2016 *IEEE Trans. Electron Devices* **63** 1508–15
[19] Matsukatova A N *et al* 2023 *Adv. Intell. Syst.* **5** 2200407
[20] Das H, Febbo R D, Rizzo C P, Chakraborty N N, Plank J S and Rose G S 2023 *IEEE J. Emerg. Sel. Top. Circuits Syst.* **13** 889–900
[21] Payvand M, Demirag Y, Dalgaty T, Vianello E and Indiveri G 2020 Analog weight updates with compliance current modulation of binary rerams for on-chip learning *2020 IEEE Int. Symp. on Circuits and Systems (ISCAS)* pp 1–5
[22] Teich J 2012 *Proc. IEEE* **100** 1411–30
[23] Plank J S, Schuman C D, Bruer G, Dean M E and Rose G S 2018 *IEEE Lett. Comput. Soc.* **1** 17–20
[24] Foshie A Z, Rizzo C, Das H, Zheng C, Plank J S and Rose G S 2022 Benchmark comparisons of spike-based reconfigurable neuroprocessor architectures for control applications *GLSVLSI—Great Lakes Symp. on VLSI.* (ACM) pp 383–6
[25] Chakraborty N N, Rose G S and Kao M H 2022 Programmable refractory period implementations in a mixed-signal integrate-and-fire neuron *2022 IEEE Int. Symp. on Circuits and Systems (ISCAS)* pp 770–4
[26] Chakraborty N N, Das H and Rose G S 2023 A mixed-signal short-term plasticity implementation for a current-controlled memristive synapse *Proc. Great Lakes Symp. on VLSI 2023, GLSVLSI'23 (New York, NY, USA)* (Association for Computing Machinery) pp 179–82
[27] Chakraborty N N, Das H and Rose G S 2023 Spike-timing-dependent plasticity for a hafnium-oxide memristive synapse *2023 IEEE 66th Int. Midwest Symp. on Circuits and Systems (MWSCAS)* pp 463–7
[28] Chakraborty N N, Das H and Rose G S 2023 Spike-driven synaptic plasticity for a memristive neuromorphic core *2023 IEEE 66th Int. Midwest Symp. on Circuits and Systems (MWSCAS)* pp 644–8
[29] Das H, Rathore M, Febbo R, Liehr M, Cady N C and Rose G S 2023 Rfam: reset-failure-aware-model for hfo2-based memristor to enhance the reliability of neuromorphic design *Proc. Great Lakes Symp. on VLSI 2023, GLSVLSI'23, (New York, NY, USA)* (Association for Computing Machinery) pp 281–6
[30] Beckmann K, Olin-Ammentorp W, Chakma G, Amer S, Rose G S, Hobbs C, Nostrand J V, Rodgers M and Cady N C 2020 *ACM J. Emerg. Technol. Comput. Syst.* **16** 1–18
[31] Das H, Febbo R D, Tushar S N B, Chakraborty N N, Liehr M, Cady N C and Rose G S 2023 *IEEE Trans. Circuits Syst.* I **70** 4804–15
[32] Chakraborty N N, Das H and Rose G S 2023 Homeostatic plasticity in a leaky integrate and fire neuron using tunable leak *2023 IEEE 66th Int. Midwest Symp. on Circuits and Systems (MWSCAS)* pp 738–42
[33] Schuman C D, Mitchell J P, Patton R M, Potok T E and Plank J S 2020 Evolutionary optimization for neuromorphic systems *Proc. 2020 Annual Neuro-Inspired Computational Elements Workshop* pp 1–9
[34] Schuman C *et al* 2022 *Neuromorph. Comput. Eng.* **2** 014002
[35] Tanaka G, Yamane T, Héroux J B, Nakane R, Kanazawa N, Takeda S, Numata H, Nakano D and Hirose A 2019 *Neural Netw.* **115** 100–23
[36] Jaeger H and Haas H 2004 *Science* **304** 78–80
[37] Lukoševičius M and Jaeger H 2009 *Comput. Sci. Rev.* **3** 127–49
[38] Pedregosa F *et al* 2011 *J. Mach. Learn. Res.* **12** 2825–30 (arXiv:1201.0490)
[39] Foshie A Z, Plank J S, Rose G S and Schuman C D 2023 Functional specification of the ravens neuroprocessor (arXiv:2307.15232)
[40] Schuman C D, Plank J S, Bruer G and Anantharaj J 2019 Non-traditional input encoding schemes for spiking neuromorphic systems *IJCNN: The Int. Joint Conf. on Neural Networks* pp 1–10
[41] Bird J J, Faria D R, Manso L J, Ekárt A and Buckingham C D 2019 *Complexity* **2019** 1–14
[42] Bird J J, Ekart A, Buckingham C D and Faria D R 2019 Mental emotional sentiment classification with an eeg-based brain-machine interface *Proc. International Conf. on Digital Image and Signal Processing (DISP'19)*
[43] Asuncion A and Newman D 2007 Uci machine learning repository
[44] Brockman G, Cheung V, Pettersson L, Schneider J, Schulman J, Tang J, and Zaremba W 2016 Openai gym (arXiv:1606.01540)
[45] Shahsavari M, Thomas D, Brown A and Luk W 2021 Neuromorphic design using reward-based STDP learning on event-based reconfigurable cluster architecture *Int. Conf. on Neuromorphic Systems 2021, (New York, NY, USA)* (Association for Computing Machinery)
[46] Wang W, Song W, Yao P, Li Y, Van Nostrand J, Qiu Q, Ielmini D and Yang J J 2020 *iScience* **23** 101809
[47] Lu S and Sengupta A 2022 Hybrid neuromorphic systems: an algorithm-application-hardware-neuroscience co-design perspective: invited special session paper *2022 IEEE 4th Int. Conf. on Artificial Intelligence Circuits and Systems (AICAS)* pp 210–3
[48] Varshika M, Mishra A K, Kandasamy N and Das A 2023 Hardware-software co-design for on-chip learning in ai systems *Proc. 28th Asia and South Pacific Design Automation Conf.* pp 624–31
[49] Plank J S, Rose G S, Dean M E, Schuman C D and Cady N C 2017 A unified hardware/software co-design framework for neuromorphic computing devices and applications *2017 IEEE Int. Conf. on Rebooting Computing (ICRC)* (IEEE) pp 1–8