SpikeSynth: Energy-Efficient Adaptive Analog Printed Spiking Neural Networks

Priyanjana Pal[1*], Alexander Studt[1*], Tara Gheshlaghi[1], Michael Hefenbrock[2], Michael Beigl[1], and Mehdi Tahoori[1]
[1]*Karlsruhe Institute of Technology,* [2]*RevoAI GmbH*
[1]{priyanjana.pal, alexander.studt, tara.gheshlaghi, michael.beigl, mehdi.tahoori}@kit.edu, [2]michael.hefenbrock@revoai.de

***Abstract*—Biologically-inspired Spiking Neural Networks (SNNs) have emerged as a promising avenue toward energy-efficient neuromorphic computing, particularly in edge applications such as soft robotics, wearable health monitors, and IoT devices. Printed Electronics (PE), offering advantages of ultra-low cost fabrication and mechanical flexibility, present a viable platform to realize such neuromorphic systems at scale. However, designing adaptable and efficient spiking circuits that meet the unique constraints of PE applications remains a challenge. To address this, we propose a novel analog spiking neuromorphic circuit with a learnable spike generator (LSG). Unlike fixed-threshold models, our generator adapts spike timing dynamics during training, enabling better task-specific performance. To optimize for ultra-low power consumption on resource-constrained platforms, we further introduce a robustness-aware training framework that further minimizes the energy consumption adaptively. Simulation results across 13 benchmarks demonstrate an average 57.6% power reduction for the LSG while improving the average classification accuracy by 8%, area and energy reduction by 89% and 28.7% respectively compared to the state-of-the-art printed analog spiking neural networks (P-SNNs).**

I. Introduction

The evolution of modern electronics is increasingly driven by the demand for energy-efficient systems capable of performing cognitive tasks through brain-inspired neuromorphic computing. Applications such as soft robotics, on-body health monitoring, edge intelligence, smart packaging and large-scale IoT networks as shown in Fig. 1[1] require electronic platforms that are flexible, lightweight, and cost-effective. Unfortunately, conventional silicon technologies—though powerful—are fundamentally constrained by their rigid substrates, high-temperature processing, and expensive fabrication infrastructure, making them unsuitable for many of these emerging use cases [1].

In this regard, printed electronics (PE) has emerged as a ultra-low-cost fabrication technology that overcomes these challenges through additive manufacturing of electronic circuits and components directly onto flexible substrates [1]. This approach eliminates the need for complex lithography and etching steps, drastically reducing both production time and costs [2]. Moreover, PE supports the integration of electronics into flexible, stretchable, and even foldable materials, enabling seamless embedding into fabrics, plastics, and other unconventional carriers. Such characteristics are especially valuable for applications with tight mechanical constraints or disposability requirements, making PE a compelling candidate for the next generation of intelligent and form-adaptive devices [3, 4].

Fig. 1. Target application domains of printed electronics: smart bandages, smart food packaging, RFID tags and smart milk carton etc [3, 4].

Traditional computing systems based on von Neumann architecture are increasingly facing fundamental challenges when applied to cognitive tasks requiring perception, learning, and real-time decision-making. These limitations—especially in terms of energy consumption, parallelism, and adaptability—have sparked growing interest in alternative models inspired by the human brain [5]. Among the most promising developments are the Spiking Neural Networks (SNNs), which form the foundation of neuromorphic computing [6, 7].

To address these challenges, researchers have turned to brain-inspired neuromorphic computing paradigms, particularly SNNs [8]. Unlike conventional artificial neural networks (ANNs), which process information using continuous-valued signals and synchronous updates [9], SNNs operate using discrete electrical pulses or "spikes" to encode and transmit information. This biological inspiration enables SNNs to perform cognitive tasks with very high energy efficiency and lower computational complexity [10]. Such characteristics align ideally with the constraints and opportunities of PE, providing a compelling rationale for integrating SNN-based neuromorphic computing. Specifically, analog SNNs inherently offer ultra-low-power operation and compact implementations, making them highly suitable for the low-power, resource-constrained environments typically targeted by PE applications [11].

The decision to select an appropriate paradigm for SNN, either digital or analog, remains a crucial decision in neuromorphic system design. Digital SNNs are built using standard digital circuits and are known for their reliability, especially in noisy environments. They are also easier to scale up for large networks and can be designed using existing digital tools [12, 13]. On the other hand, analog SNNs work by mimicking how real neurons and synapses behave using continuous electrical signals. These designs can be much more energy-efficient and use less chip area, which is helpful in systems

*Authors contributed equally to this work.
[1]Images are generated by the DALL-E AI tool.

where power and space are limited, making them a promising choice for PE applications with strict energy constraints. Analog SNNs also process data in real-time and are naturally suited for tasks where quick, energy-efficient responses are important.

Leveraging the benefits offered by PE and neuromorphic computing, prior studies successfully realized the implementation of analog printed spiking neurons on organic substrates [14, 15]. Additionally, researchers presented adaptations to the architectures and training algorithms for analog Printed Artificial Neural Networks (P-ANNs) [16]. Also, they introduced training algorithms that account for variation, aging effects, fault sensitivity and power considerations [17]–[20]. Although prior work has demonstrated the feasibility of analog spiking neuron circuits using PE and even proposed architecture-level and algorithmic adaptations for P-ANNs [14]–[16], most implementations rely on fixed spike generation mechanisms. In contrast, our work introduces a programmable and learnable spike generation module, where the temporal firing behavior of the neuron is parameterized and co-optimized during training.

This learnable spike generator (LSG) provides a novel analog neuron architecture from conventional designs by allowing the neuron to adapt its temporal dynamics in response to both task objectives and circuit-level constraints. Furthermore, we incorporate robustness-aware training not only on the weights, but within the spike generation process itself, accounting for device-level non-idealities such as process variation or aging [17, 18, 21]. To the best of our knowledge, this is the first demonstration of a learnable analog SNN implemented in PE. In this work, we propose an analog printed spiking neuron circuit design and its associated learning algorithm for neural network computation. In short, the contributions are:

- We propose a novel spiking neuron architecture with a *learnable and trainable spike generator*, enabling the neuron to adapt its temporal firing dynamics during learning.
- We incorporate *robustness-aware design methodology* into the spike generation process itself, accounting for process variation, and training with bounded hardware constraints.

Through simulations on 13 benchmarks, we demonstrate that our design achieves a **57.6%** average reduction in learnable spike-generator (LSG) power, **89%** area reduction, achieves classification accuracy of **83%**, and a **28.7%** reduction in inference energy compared to non-adaptive P-SNNs.

The rest of this paper is structured as follows: Sec. II, provides the background of this work. Sec. III proposes the learnable printed analog spiking neuron design and develops a learning algorithm to solve specific classification tasks. Sec. IV introduces the robustness-aware training of the P-LSNN considering variations of the parameters. In Sec. V, we evaluate the effectiveness of the proposed models on benchmark datasets, compare our results with the existing printed neuromorphic circuits, and discuss its application. Finally, Sec. VI concludes this paper.

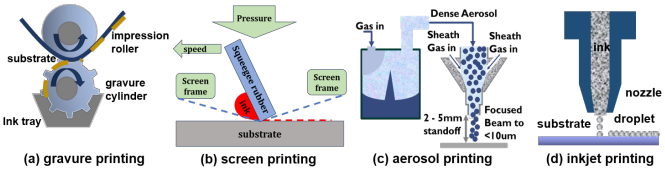


Fig. 2. Schematic of (a) gravure printing (b) screen printing; (c) aerosol printing and (d) inkjet printing [26]

II. Preliminaries

A. Printed Electronics (PE)

PE refers to a fabrication technology which is based on printing processes, such as jet-printing, screen- or gravure-printing [22]. Due to the simple manufacturing process as well as low equipment costs, ultra low-cost electronic circuits can be fabricated, much cheaper compared to silicon-based processes, which require expensive foundries even for semiconductor manufacturing foundries [23] with older technology nodes. Moreover, electronics on flexible substrates are enabled by using contact-less printing methods such as inkjet-printers in combination with highly optimized functional inks such as conductive, semi-conductive and non-conductive materials. From these inks, organic [24] or oxide-based [25] transistors can be built. While organic materials are easy to be processed, they have lower environmental stability. On the other side, oxide-based inks have excellent conductivity and environmental stability, but are harder to be printed and suffer from impurities due to surfactants [22].

TABLE I
Key properties and parameters of different printing methods

Property/Parameter	Gravure	Screen	Aerosol	Inkjet
Throughput (m^2/s)	3–60	2–3	0.01–0.1	0.01–0.5
Resolution (lines/cm)	20–400	50	10–100	60–250
Printing Speed (m/min)	100–1000	10–15	5–50	15–500

Printing technologies, as shown in Fig. 2, are broadly categorized into two main types, each tailored for specific applications and material requirements: (i) contact printing and (ii) non-contact printing. Contact printing includes: (a) replication printing, exemplified by gravure printing (illustrated in Fig. 2 (a)), which is optimized for high-volume production; and Fig. 2 (b) screen printing, which entails considerable manufacturing time and costs and yields relatively low resolution. In contrast, non-contact printing involves: Fig. 2 (c) aerosol printing, ideal for lower-volume production due to its slower printing speeds and higher maintenance demands; and Fig. 2 (d) jet printing, with inkjet printing as an example, for customized fabrication of electronic circuits in smaller quantities. The key properties and parameters of the different printing techniques are summarized in Tab. I.

However, PE utilizes either additive or subtractive manufacturing methods. As shown in Fig. 3 (a), the subtractive process alternates between deposition and etching, similar to traditional silicon-based methods and is more expensive due to specialized processing, costly equipment, and infrastructure. Conversely, the additive process as in Fig. 3 (b) involves only deposition steps, layering materials to create transistors, passive components, and interconnects. Although fully additive PE typically operate slower, have larger feature sizes, and exhibit

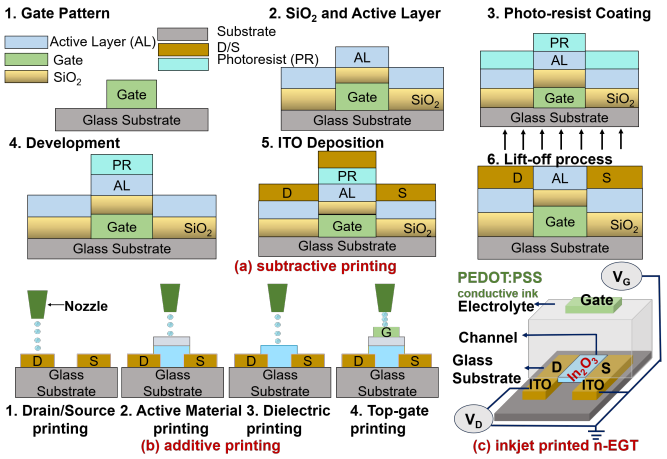


Fig. 3. Schematic of (a) subtractive process (b) additive manufacturing process (c) inkjet n-type electrolyte-gated (n-EGT) transistor.

greater variability than their subtractive counterparts, they offer significant cost advantages particularly appealing for the ultra-low-cost electronics.

Most state-of-the-art inkjet-printed field-effect transistors rely on organic semiconductors structured as channels between source and drain electrodes. Typically, organic FETs use P-type materials exhibiting low field-effect mobility [27] and require high operating voltages (≥ 25 V), making them unsuitable for low-power PE applications for energy harvesting or small batteries. In contrast, inorganic oxide semiconductors employing N-type electrolyte-gated transistors (nEGTs, Fig. 3 (c)) provide higher electron mobility, enabling sub-1 V operation ideal for energy-constrained scenarios [28, 29].

B. Printed Artificial Neural Networks (P-ANNs)

P-ANNs are analog computing architectures designed to replicate the core operations of ANNs using low-cost, large-area fabrication techniques. These systems can operate directly on analog sensory inputs, bypassing the need for digitization and enabling compact, energy-aware edge computing [16]. Weighted summations are typically performed using printed resistor crossbar arrays, negative weights are handled using analog inverters, and nonlinear activation functions such as tanh or ReLU are implemented through analog primitives—all within a fully analog domain. Recent advancements have extended the capabilities of printed architectures by introducing printed temporal filters [30], thereby allowing P-ANNs to handle time-varying signals to some extent. However, despite this progress, P-ANNs remain inherently dense and continuously active, lacking the sparse, event-driven computation model that characterizes SNNs. This continuous activity leads to higher static power consumption, especially in scenarios involving sparse or infrequent inputs. Additionally, the analog weights and nonlinearities in P-ANNs are typically fixed post-fabrication, limiting their adaptability and reconfigurability in changing environments or across different tasks.

C. Printed Spiking Neural Networks (P-SNNs)

P-SNNs offer a biologically inspired computing paradigm well-suited for next-generation flexible and energy-efficient electronic systems. Modeled after the brain's communication

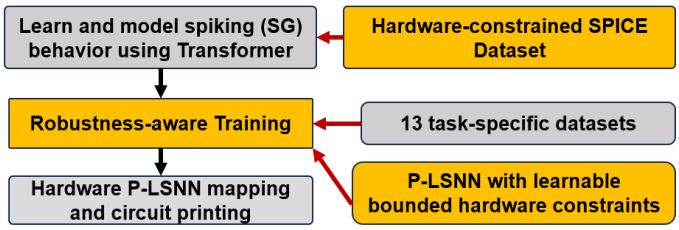


Fig. 4. Proposed flow for an on-demand robust printed spiking neural network design given a specification of a desired functionality realized through robustness-aware P-LSNN training.

mechanism, SNNs utilize discrete voltage spikes to transmit information, mimicking the behavior of biological neurons. A typical spiking neuron comprises three components: the dendrite, which receives external stimuli; the soma, which accumulates input signals and triggers spikes based on membrane potential thresholds; and the axon, which conveys the output signal to downstream neurons. In PE, these components are realized using printed transistors [31], capacitors, and resistive elements, allowing the entire neuron circuit to be fabricated on flexible, ultra-low-cost substrates. Unlike traditional ANNs, which rely on continuous signal propagation and synchronous updates, printed SNNs operate in an asynchronous manner, significantly reducing power consumption and enabling real-time signal processing at the edge [32].

D. Low-Power and Energy-Efficient Neuromorphic Design

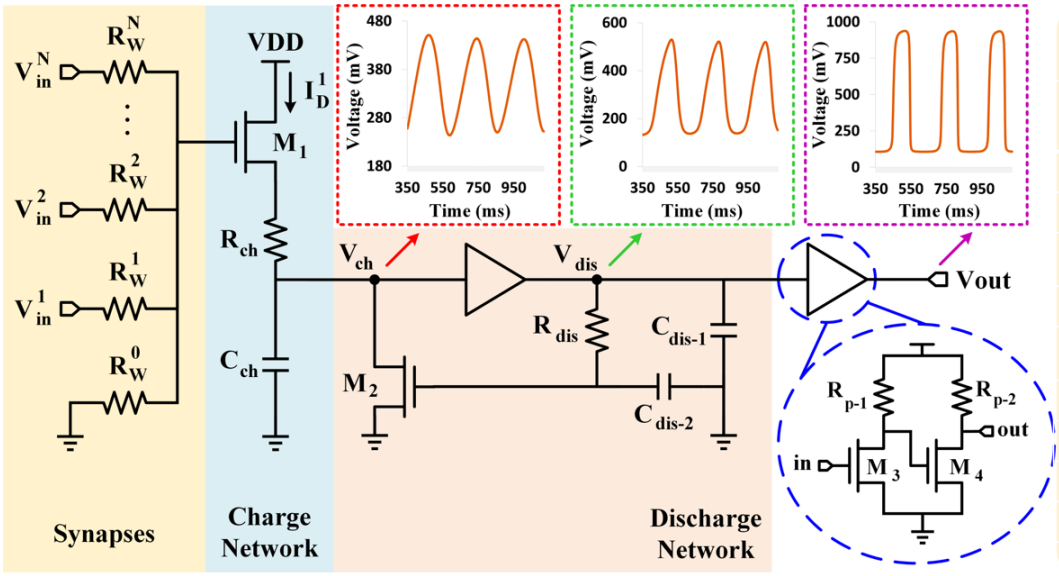
Although neuromorphic computing has already been proven to be significantly power and energy-efficient compared to conventional approaches [33], ongoing research aims to further reduce the power consumption of these neuromorphic circuits. For instance, [34] developed a novel device to decrease the power required for the activation functions. The work of [35] utilized hardware-software co-design to optimize circuit structure for data flow in the computing process. Regarding computational paradigms, numerous silicon circuits have adopted brain-inspired SNN to minimize power in analog, digital, or mixed-signal by integrating synaptic inputs, generating action potentials, and transmitting them along neuron axons to connect with post-synaptic terminals [10, 12, 36].

III. Proposed Learnable Printed Spiking Neural Networks (P-LSNNs)

The proposed P-LSNN begins with the design of a physically realizable spiking neuron circuit, followed by a Transformer-based surrogate model that enables differentiable training. Once the spiking network is trained using this surrogate, the learned parameters are systematically mapped back to corresponding analog circuit components, completing the design loop as illustrated in Fig. 4.

A. Implementation of Learnable Printed Spiking Neuron

The architecture of the proposed Learnable Printed Spiking Neuron is organized into three key stages: synaptic input, charge accumulation, and reset-discharge control. The synaptic stage interfaces with external stimuli and mimics biological synapses by modulating inputs based on assigned weights. The subsequent charge network governs the membrane potential's integration over time, while the reset and discharge network



Feasible design space

Design parameters	min	max
$R_{ch}\,(k\Omega)$	20	50
$R_{dis}\,(k\Omega)$	20	50
$C_{ch}\,(nF)$	10	100
$C_{dis\text{-}1}(nF)$	10	100
$C_{dis\text{-}2}(nF)$	10	100
$R_{p\text{-}1}(k\Omega)$	20	50
$R_{p\text{-}2}(k\Omega)$	20	50
$W_{M1,M2,M3}(\mu m)$	600	1000
$L_{M1,M2,M3}(\mu m)$	40	100

Fig. 5. Circuit level implementation of proposed printed learnable spiking neuron which includes three stages: Synapses, Charge Network, and Reset and Discharge Network. The feasible design space for the learnable design parameters are tabulated in the right side of the figure.

manages spike generation and membrane reset, collectively forming a trainable spike-generation mechanism.

a) Synapses: Synapses act as the interface between presynaptic and postsynaptic activity, emulating biological signal transmission. In our circuit, presynaptic neurons are modeled as voltage input sources. Each input is weighted via a resistor crossbar network, enabling scalable and tunable input processing. This crossbar structure facilitates parameter learnability, allowing the neuron's response to adapt during training. Applying nodal analysis, the synaptic current reaching the neuron's core is modeled as the weighted sum of all input voltages:

$$\frac{V_g^1}{R_w^0} + \frac{V_g^1 - V_{in}^1}{R_w^1} + \cdots + \frac{V_g^1 - V_{in}^N}{R_w^N} = 0. \tag{1}$$

Here, V_{g}^1 denotes the gate voltage of transistor M_1, while V_{in}^1 represents the input voltage applied to the neuron. The resistor R_{w}^0 forms part of a voltage divider, establishing the input biasing condition. The resistors R_{w}^1 through R_{w}^N encode the synaptic weights of N presynaptic neurons, enabling weighted summation in the analog domain. These resistance values are tunable, allowing the circuit to adapt its response based on learned synaptic strengths.

To support both excitatory and inhibitory connections, an additional subcircuit is included to handle negative weights, following the strategy proposed in [16]. For the computation of the optimal gate voltage V_{g}^1 corresponding to the learned parameters, please refer to the detailed calculation procedure in Sec. III-B.

b) Learnable Spike-generator (LSG): As illustrated in Fig. 5, the LSG converts a weighted sum of incoming synaptic voltages into a precisely timed train of output spikes. Its *input* is at the gate voltage V_g^1 of transistor M_1, which encodes the weighted sum of N synaptic inputs V_{in}^i via resistive weights R_w^i (as described in Sec. III-B2); whereas its *output* is a spiking waveform V_{out} whose timing (delay, width, and inter-spike interval) encodes the learned integration of those inputs. As shown in Fig. 5, the LSG implements an event-driven transfer characteristics and integrates the charge until an internal node $V_{\mathrm{ch}}(t)$ exceeds the threshold voltage of M_2, triggers a firing event, then resets and enters a short refractory period before repeating. The overall LSG circuit consists of i) charge network, ii) thresholding and iii) reset and discharge network.

- **Charge Network:** The components R_{ch} and C_{ch} together form a passive RC network that generates a delayed voltage signal V_{in} across the capacitor C_{ch}. This delay is influenced by both the time constant $R_{\mathrm{ch}}C_{\mathrm{ch}}$ and the frequency of incoming spikes. The gate voltage of transistor M_1, denoted as V_{g}^1, plays a critical role by regulating the drain current I_{D}^1, thereby dynamically adjusting the charging behavior of C_{ch}. This modulation directly influences the timing and frequency of spike generation, making the system responsive to learnable parameters.
- **Thresholding:** The voltage signal across C_{ch} is amplified using a two-stage inverter-based amplifier. The first inverter, composed of an N-type pull-down transistor and a resistive pull-up network, provides initial amplification but introduces a 180° phase shift. To correct this inversion and stabilize the signal, a second inverter stage is cascaded. The amplifier output is subsequently connected to two cascaded RC networks: the first comprising R_{p-2} and C_{dis-1}, and the second composed of R_{dis} and C_{dis-2}. These networks introduce phase delays essential for sustaining oscillatory behavior in the spiking neuron.
- **Reset and Discharge Network:** The final capacitor, C_{dis-2}, modulates the gate voltage V_{g}^2 of transistor M_2. When V_{g}^2 exceeds the threshold voltage of M_2, current I_{D}^2 is activated, discharging C_{ch} and thereby initiating a reset. Meanwhile, as synaptic inputs continue to inject charge through the weighted resistor network, C_{ch} gradually recharges, enabling periodic spiking behavior. This cycle supports adaptive spiking dynamics, with timing influenced by both circuit-level parameters and learnable input conditions. While the first amplification stage increases the signal amplitude at V_{in}, the resulting output voltage

V_{out} may still be insufficient to drive subsequent neurons. To address this, a second amplification stage is included, boosting the final output voltage $V_{\text{out}}^{'}$ to approach the supply level ($\sim$ 1 V), ensuring reliable transmission to postsynaptic neural circuits.

B. Modeling and training of P-LSNN

By interconnecting multiple printed spiking neurons, the behavior of a SNN can be effectively realized in hardware, enabling a wide range of neuromorphic computations. To understand the architecture, however, it is essential to tailor the circuit-level parameters—such as the conductances in the resistor crossbar that encode synaptic weights—to the requirements of specific target tasks. Achieving this requires the formulation of a learnable optimization framework for the P-LSNN, allowing the component values to be systematically adjusted through task-driven training objective.

1) Modeling of P-LSNN: The behavior of the printed LSG circuit is modeled using a transformer-based architecture that predicts the output voltage time series $V_{\text{out}}(t)$ as a function of the input voltage $V_{\text{in}}(t)$ and a set of hardware parameters expressed as the vector $\boldsymbol{\upsilon}$ where each component, such as the resistances or width and length of the transistors, is restricted with $\upsilon_i^{\min} \leq \upsilon_i \leq \upsilon_i^{\max}$. These parameters are constrained within physically realizable ranges to reflect the functionality within PE.

To respect hardware constraints, each unconstrained parameter $\zeta_i \in \mathbb{R}$ is passed through a scaled $\tanh$ function. The output of the surrogate model is given by:

$$\text{LSG}\left(V_{\text{in}}(t), \boldsymbol{\upsilon}\right) \tag{2}$$

where the components are given by

$$\upsilon_i = \frac{\upsilon_i^{\max} + \upsilon_i^{\min}}{2} + \frac{\upsilon_i^{\max} - \upsilon_i^{\min}}{2} \tanh(\zeta_i) \tag{3}$$

and $[\upsilon_i^{\min}, \upsilon_i^{\max}]$ defines the hardware-feasible interval for parameter υ_i. These transformed values are provided as input to the surrogate model. The input voltage is given by the crossbar:

$$V_{\text{in}}(t) = \sum_{n=1}^{N} V_{\text{in}}^n(t)\left(w_n \cdot \mathbb{1}_{\theta_n \geq 0}\right) + \text{inv}(V_{\text{in}}^n(t))\left(w_n \cdot \mathbb{1}_{\theta_n < 0}\right). \tag{4}$$

Here θ_n represents the learnable parameter where the absolute value is the crossbar conductance, i.e. $g_n = |\theta_n|$ and $\mathbb{1}_{\{\cdot\}}$ is the indicator function that returns 1 if its condition is met and is otherwise 0 and $\text{inv}(\cdot)$ denotes the negative function that is realized by the negation circuit. The weights w_n are given by the conductances normalized by all conductances $w_n = \frac{g_n}{\sum_m g_m}$.

2) Surrogate model for the LSG with hardware constraints: To facilitate gradient-based training through backpropagation [37], it is essential to construct a fully differentiable model that accurately captures the transfer characteristics of the printed spike-generator circuit. This work introduces a novel training methodology for P-LSNNs by integrating a differentiable, hardware-agnostic surrogate model directly into the optimization pipeline. Unlike prior approaches [32] that relied on surrogate modeling for behavioral approximation, we leverage the Transformer architecture to enable end-to-end gradient-based learning of both network and circuit-level parameters.

The proposed methodology treats these hardware parameters as trainable entities bounded within limits, allowing for simultaneous optimization of functional accuracy and physical realizability. By embedding the Transformer-based surrogate into the training loop, the model captures the dynamic response of the spike generator circuit from input voltages $V_{\text{in}}(t)$ to output voltages $V_{\text{out}}(t)$, while maintaining differentiability for backpropagation [37]. This closed-loop, hardware-constrained training flow allows for co-design of the neural architecture and its physical implementation, enabling adaptive and task-optimized behavior in printed spiking neuromorphic systems. Transformer architectures have demonstrated exceptional versatility across a wide range of domains, forming the backbone of state-of-the-art models like BERT [38] and GPT [39]. In this work, we leverage their sequence modeling capabilities to construct a fully learnable and differentiable approximation of the printed LSG's temporal dynamics.

Algorithm 1 End-to-End Training of P-LSNN with LSG Surrogate Model.

Require: Training dataset $\mathcal{D} = \{\boldsymbol{x}, \boldsymbol{y}\}$
Require: SPICE simulation dataset $\mathcal{S}$
Train surrogate model LSG on $\mathcal{S}$ using MSE loss
Freeze surrogate model parameters
Initialize network weights $\boldsymbol{\theta}$ and LSG parameters $\boldsymbol{\zeta}$
while lr $>$ lr_{min} **do**
  for each $(x, y) \in \mathcal{D}$ **do**
    Generate temporal input sequence $x^{(t)}$
    Run forward pass through P-LSNN that includes:
      Compute crossbar voltage V_{in} using equation (4)
      Compute LSG output V_{out} using equation (2)
    Compute loss with equation (5) or (7) (without or with robustness-aware training respectively)
    Backpropagate loss w.r.t. $\boldsymbol{\theta}$ and $\boldsymbol{\zeta}$
    Update parameters using Adam optimizer
  end for
  if Patience $>$ 100 **then**
    lr $\leftarrow$ lr/2
  end if
end while

To generate the dataset for training the LSG surrogate model, we conducted over 64K SPICE simulations using the well-established Printed Process Design Kit (P-PDK) [40]. Each simulation was executed for a duration of 15 ms with a temporal resolution of 10 μs to capture fine-grained transient behavior of the circuit. The input voltage $V_{\text{in}}(t)$ was varied across multiple regimes to ensure that the surrogate model could generalize to both static and dynamic spiking conditions. The following input categories were considered:

- Constant voltages, ranging from 0 V to 2 V in steps of 0.2 V, which serve as baseline inputs for characterizing stable responses;

- Cascaded neuron outputs, where the output from a first neuron is used as the input to a second neuron, thereby modeling signal propagation through a multi-layer spiking network;
- A set of harmonic input voltage signals with varying amplitudes (A), DC offsets (B), frequencies (f from 0 Hz to 5 Hz), and phase shifts (0 to 2π), intended to emulate dynamic and noisy spiking activity.

Each simulation instance is parameterized by range of learnable parameters as tabulated in Fig. 5. These parameters are constrained within the intended functionality of the spike generator and serve as the input features to the surrogate model. This comprehensive SPICE dataset uses more than ten times the size of that used in previous studies [32], which did not account for learnable hardware parameters. This enables accurate modeling of the spike-generator circuit under diverse operating conditions. A Transformer model with 3 encoder layers, 3 attention heads, and an embedding size of 48 achieved the best performance during evaluation, resulting in a mean squared error (MSE) of $7.3 \cdot 10^{-3}$ on the test set.

3) Training of the P-LSNN: After pretraining, the surrogate model is frozen and used to replace the LSG circuit in the full P-LSNN. The additional parameters ζ_i are now treated as independently trainable variables for each LSG circuit. These are optimized together with the rest of the P-LSNN using gradient descent.

The P-LSNN receives temporally extended input signals and produces spike-based outputs over time. To promote consistent prediction, the cross-entropy loss is computed at every time step and averaged across the sequence, which leads to this optimization objective:

$$\underset{\boldsymbol{\theta},\boldsymbol{\zeta}}{\text{minimize}} \frac{1}{T} \sum_{t=0}^{T} L\left(\boldsymbol{x}_t, \boldsymbol{y}, \boldsymbol{\theta}, \boldsymbol{\zeta}\right), \tag{5}$$

where $\boldsymbol{x} \in \mathbb{R}^{B \times T}$ is the input data series with batch size B, $\boldsymbol{y} \in \mathbb{R}^{B}$ denotes the corresponding classes, $\boldsymbol{\theta}$ is the vector of all the learnable parameters from the crossbar conductances and $\boldsymbol{\zeta}$ is the vector of the learnable hardware parameters of the LSG circuit. Since all components, including the LSG model with constrained v_i, are differentiable, the full system supports end-to-end training.

This methodology enables full backpropagation-based training of P-LSNNs by introducing a differentiable surrogate for the LSG circuit, parameterized by constrained learnable variables, allowing both the circuit-level parameters and network weights to be jointly optimized in a hardware-aware adaptive manner. The entire end-to-end training of the P-LSNN is shown in algorithm 1.

IV. Robustness-aware P-LSNN Training

In PE, the parameters associated with hardware components are highly susceptible to process-induced variations. These variations arise due to factors such as ink spreading, droplet jetting irregularities, and satellite droplet wetting on the substrate [42]. Specifically, printed n-type electrolyte-gated transistors (n-EGTs) exhibit variability across multiple fabrication stages—including channel, dielectric, and top-gate formation—resulting in non-Gaussian distributions for electrical and structural properties [40].

TABLE II
Accuracy Comparison Across Different Printed Nueromorphic Architectures across 13 benchmark datasets

Dataset	SNN[1]	P-ANN[2] [41]	P-SNN[3] [32]	P-LSNN[4]
Acute.	1.00	0.999 ± 0.002	1.000 ± 0.000	1.000 ± 0.000
Bal.	0.84	0.893 ± 0.020	0.430 ± 0.000	0.819 ± 0.073
Breast.	0.98	0.915 ± 0.020	0.975 ± 0.000	0.966 ± 0.007
Cardio.	0.84	0.824 ± 0.014	0.747 ± 0.000	0.755 ± 0.015
En(y1)	1.00	0.918 ± 0.014	0.848 ± 0.000	0.848 ± 0.000
En(y2)	0.98	0.892 ± 0.010	0.860 ± 0.000	0.869 ± 0.026
Iris	1.00	0.958 ± 0.009	0.710 ± 0.171	0.946 ± 0.019
Mamm.	0.82	0.789 ± 0.006	0.810 ± 0.007	0.810 ± 0.013
Pen.	0.38	0.371 ± 0.034	0.400 ± 0.070	0.536 ± 0.060
Seed.	1.00	0.900 ± 0.015	0.837 ± 0.070	0.915 ± 0.013
Tic-Tac.	1.00	0.818 ± 0.004	0.789 ± 0.019	0.873 ± 0.125
Vert.(2 cl.)	0.85	0.768 ± 0.038	0.635 ± 0.000	0.635 ± 0.000
Vert(3 cl.)	0.83	0.819 ± 0.004	0.683 ± 0.016	0.778 ± 0.057
Average	**0.886**	**0.836 ± 0.015**	**0.748 ± 0.027**	**0.830 ± 0.031**

[1] SNN: Software Spiking Neural Network [2] P-ANN: Printed Artificial Neural Network [3] P-SNN: Printed Spiking Neural Network [4] P-LSNN: Printed Learnable Spiking Neural Network.

To account for these fabrication non-idealities during training, we model both $\tilde{\boldsymbol{\theta}}$ and $\tilde{\boldsymbol{\zeta}}$ as random variables drawn from distributions $p(\tilde{\boldsymbol{\theta}})$ and $p(\tilde{\boldsymbol{\zeta}})$, respectively. $\tilde{\boldsymbol{\zeta}}$ represents the parameters for the LSG circuit and $\tilde{\boldsymbol{\theta}}$ the remaining parameters in the P-LSNN. These distributions capture the statistical nature of variation introduced during the printing process. Consequently, the training objective is defined over these distributions:

$$\underset{\tilde{\boldsymbol{\theta}},\tilde{\boldsymbol{\zeta}}}{\text{minimize}}\, L(\mathcal{D}, \tilde{\boldsymbol{\theta}}, \tilde{\boldsymbol{\zeta}}), \tag{6}$$

where the parameters $\tilde{\boldsymbol{\theta}}$ and $\tilde{\boldsymbol{\zeta}}$ are resampled from their respective distributions at each forward pass, allowing the model to encounter diverse perturbations over the course of training. Here, $\mathcal{D} = \{\boldsymbol{x}, \boldsymbol{y}\}$ denotes the training dataset, and $L(\cdot)$ is the task-specific loss function (e.g., cross-entropy [43]).

To make the formulation differentiable and trainable via gradient-based methods, we employ a reparameterization trick [44], expressing the random variables as:

$$\tilde{\boldsymbol{\theta}} = \boldsymbol{\theta} \odot \boldsymbol{\varepsilon}_\theta, \quad \tilde{\boldsymbol{\zeta}} = \boldsymbol{\zeta} \odot \boldsymbol{\varepsilon}_\zeta,$$

where $\boldsymbol{\theta}$ and $\boldsymbol{\zeta}$ are the nominal, learnable parameters, and $\boldsymbol{\varepsilon}_\theta \sim p(\boldsymbol{\varepsilon}_\theta)$ and $\boldsymbol{\varepsilon}_\zeta \sim p(\boldsymbol{\varepsilon}_\zeta)$ represent multiplicative noise sampled from the variation distributions, which in our case are uniform. Substituting this into the loss yields modified training objective:

$$\underset{\boldsymbol{\theta},\boldsymbol{\zeta}}{\text{minimize}}\, L(\mathcal{D}, \boldsymbol{\theta} \odot \boldsymbol{\varepsilon}_\theta, \boldsymbol{\zeta} \odot \boldsymbol{\varepsilon}_\zeta). \tag{7}$$

V. Evaluation Framework

To assess the effectiveness of the proposed P-LSNN with learnable spike-generation capabilities, we first designed the spike-generator incorporating synaptic inputs, and implemented

TABLE III
ROBUSTNESS-AWARE ACCURACY ACROSS MODELS WITH $\pm$ 10% COMPONENT VARIATION ACROSS 13 BENCHMARK DATASETS

Dataset	P-ANN [41]	P-SNN [32]	P-LSNN
Acute.	1.000 ± 0.012	1.000 ± 0.000	0.999 ± 0.007
Bal.	0.877 ± 0.008	0.457 ± 0.049	0.548 ± 0.147
Breast.	0.931 ± 0.039	0.974 ± 0.030	0.958 ± 0.016
Cardio.	0.763 ± 0.002	0.747 ± 0.000	0.757 ± 0.017
En(y1)	0.847 ± 0.012	0.857 ± 0.012	0.883 ± 0.052
En(y2)	0.867 ± 0.026	0.841 ± 0.026	0.854 ± 0.055
Iris	0.843 ± 0.045	0.637 ± 0.068	0.909 ± 0.076
Mamm.	0.766 ± 0.053	0.808 ± 0.010	0.808 ± 0.011
Pen.	0.548 ± 0.047	0.486 ± 0.058	0.466 ± 0.078
Seed.	0.820 ± 0.041	0.709 ± 0.042	0.856 ± 0.092
Tic-Tac.	0.660 ± 0.017	0.763 ± 0.026	0.804 ± 0.119
Vert.(2 cl.)	0.661 ± 0.000	0.635 ± 0.000	0.635 ± 0.000
Vert(3 cl.)	0.634 ± 0.075	0.677 ± 0.012	0.683 ± 0.050
Average	$\mathbf{0.786 \pm 0.029}$	$\mathbf{0.738 \pm 0.024}$	$\mathbf{0.781 \pm 0.055}$

the complete training pipeline using PyTorch [45].[2] We performed a comparative evaluation of our learnable P-LSNN against prior works [16, 32] and SNNs based on benchmark frameworks [46].

A. Experiment Setup

1) Circuit Setup: To evaulate our proposed approach, we utilized the well-established n-electrolyte-gated transistor (EGT) P-PDK [40] to design both the synapses and the learnable spike-generator. To evaluate their spiking behavior, we conducted more than 60K SPICE simulations to understand the circuit spike-timing behavior w.r.t the components variation in Cadence Virtuoso[3], as shown in Fig. 5.

2) Training and Evaluation Setup:

a) Datasets and Hyperparameter choices: To evaluate the proposed P-LSNN we used 13 benchmark datasets which are aligned to the applications in PE. We used a datasplit of 60%, 20% and 20% for the training, validation and test set, respectively. To optimize the model parameters including the learnable hardware parameters ζ_i we used the ADAM [47] algorithm with an inital learning rate of 0.1. This initial learning rate was halved if the validation accuracy showed no further improvement for 100 epochs. This process was repeated 10 times until a sufficiently small learning rate was reached. To ensure to reduce the sensitivity to initialization, the process was executed 10 times with random seeds from 0 to 9, ensuring the reliability of the resulting solution.

b) Baselines: We employed another two approaches as the baselines of P-SNNs to validate the major motivation of this work, that is, energy saving. P-ANNs [16] with the same topologies as P-SNNs are trained on the corresponding datasets. Additionally, considering the target computing paradigm of this work, P-SNN is compared with its hardware-agnostic counterpart by conducting training on SNNs with the leaky-integration-fire mechanism [46].

[2]https://github.com/Neuromophic-Computing/SpikeSynth

[3]https://www.cadence.com/en_US/home.html

TABLE IV
COMPARISON OF POWER, AREA, AND TRAINING TIME WITH THE EXISTING METHODS ON 13 BENCHMARK DATASETS.

Dataset	Power (mW)					Area (cm²)			Training Time (hr)		
	P-ANN [41]	P-SNN [32]		P-LSNN		P-ANN [41]	P-SNN [32]	P-LSNN	P-ANN [41]	P-SNN [32]	P-LSNN
	Total[1]	SG[2]	Total[1]	LSG[2]	Total[1]						
Acute.	6.20	0.85	1.744	0.389	1.223	1.65	12.51	1.36	0.55	4.75	5.27
Bal.	4.50	1.02	1.694	0.426	1.186	2.68	15.03	1.80	1.69	9.85	16.17
Breast.	11.00	0.85	1.189	0.411	0.664	1.76	12.50	1.70	1.39	16.46	15.98
Cardio.	18.50	1.02	3.767	0.353	2.560	2.38	15.06	2.02	1.98	29.02	35.20
En(y1)	7.70	1.02	1.680	0.466	1.138	2.28	15.07	1.71	1.48	19.69	24.42
En(y2)	7.98	1.02	1.424	0.455	1.119	1.88	15.03	1.90	1.38	21.45	21.20
Iris	5.50	1.02	1.778	0.382	1.353	2.27	15.01	1.81	0.54	4.31	5.43
Mamm.	6.54	0.85	1.410	0.382	0.981	2.13	12.52	1.66	1.19	13.42	14.65
Pen.	11.12	2.21	4.730	0.878	3.083	5.51	32.54	1.71	3.27	32.69	40.45
Seed.	7.89	1.02	1.693	0.410	1.376	2.62	15.07	1.78	0.78	5.71	5.57
Tic-Tac.	11.10	0.85	1.506	0.366	1.109	1.99	12.53	1.40	1.26	13.78	13.57
Vert.(2cl.)	5.65	0.85	1.362	0.422	1.084	2.38	12.57	1.30	0.78	7.72	7.53
Vert(3cl.)	5.75	1.02	1.697	0.392	1.427	2.38	15.01	1.53	0.92	7.21	7.59
Average	**8.42**	**1.04**	**1.974**	**0.44↓**	**1.407↓**	**2.46**	**15.41**	**1.67↓**	**1.32**	**14.31**	**16.39**

[1] Total: Total Power [2] SG: Spike-generator power [3] LSG: Learnable SG power

3) Accuracy Calculation: For a robust estimation of accuracy, we selected the top-k models ($k = 3$) from 10 independent training runs with different random seeds. The average accuracy and standard deviation were then calculated from these models on the test set. For robustness-aware training, we injected noise onto the weights to emulate imprecise hardware components. Since the evaluation results on the test set are not deterministic, we sample each test dataset 10 times per model in the robustness case and average the results.

4) Power Calculation: To evaluate the efficiency of the proposed P-LSNN architecture, we estimated the total power consumption by summing the contributions of three distinct components: the crossbar array, the negation circuit (inverter), and the spike-generation (LSG) circuit. Each component is modeled based on its specific electrical behavior to ensure accurate and hardware-relevant power estimates. To estimate the power of the crossbar and the negation circuit we used similar methods as in prior works [41]. Estimating the power consumption of the LSG circuit is more complex due to its temporal and nonlinear behavior. To address this, we trained a feed-forward neural network to predict the LSG power based on the learned hardware parameters ζ_i. This power model was trained on a dataset of SPICE simulations with a wide range of parameter combinations and input patterns. The data was split into a training (70%), validation (15%) and test (15%) set. After hyperparameter optimization we trained a feed-forward network with 3 hidden layers (256, 128, 64) until the validation accuracy did not change for 100 epochs. The resulting test loss (MSE) was $7.0 \cdot 10^{-6}$.

B. Results

Tab. II reports per-dataset classification accuracy under standard (non–robustness-aware) training for four models: the reference SNN, P-ANN, baseline P-SNN, and proposed P-LSNN. Averaged across all datasets, the SNN achieves 88.6 % accuracy, followed by P-ANN at 83.6 %, P-LSNN at 83.0 %, and P-SNN at 74.8 %. To assess robustness under hardware variability, we trained and evaluated each model with $\pm 10\%$ random weight perturbations (Tab. III). Under these perturbations, P-ANN achieves 78.6 %, P-LSNN 78.1 %, and P-SNN 73.8 %.

Fig. 6 plots energy consumption in millijoules (mJ) for P-ANN, P-SNN, and P-LSNN across all datasets: P-ANN ranges from 13.50 mJ (Bal) to 55.50 mJ (Cardio); P-SNN ranges from 3.57 mJ (Breast) to 14.19 mJ (Pen); and P-LSNN ranges from 1.99 mJ (Breast) to 9.25 mJ (Pen).

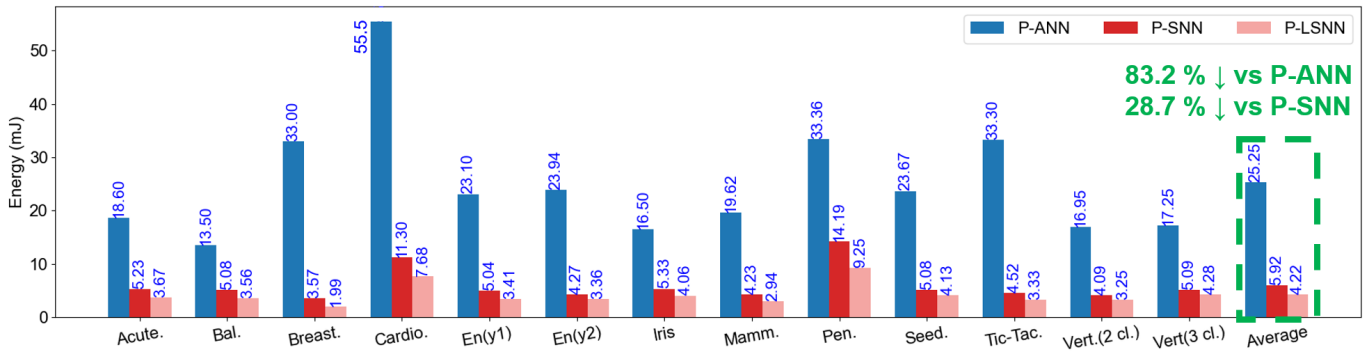


Fig. 6. Comparison of energy (mJ) utilization for P-ANN [41], P-SNN [32] and proposed P-LSNN across 13 benchmark datasets.

Tab. IV compares power consumption (mW), area (cm^2), and training time (hr) for P-ANN, P-SNN, and P-LSNN across all datasets. On average, P-ANN consumes 8.42 mW, occupies 2.46 cm^2, and requires 1.32 hr of training; P-SNN consumes 1.97 mW, occupies 15.41 cm^2, and requires 14.31 hr; and P-LSNN consumes 1.41 mW, occupies 1.67 cm^2, and requires 16.39 hr. Additionally, on average, the LSG circuit's power drops from 1.04 mW in the printed SG [32] to 0.44 mW.

C. Discussion

The comparison between the proposed P-LSNN architecture and the baseline P-SNN model demonstrates notable improvements in accuracy, area, and energy efficiency. Incorporating learnable adaptive parameters into the LSG circuit significantly enhances performance across several datasets. In particular, certain benchmarks exhibit substantial accuracy gains, while others maintain performance levels comparable to the baseline.

On average, the P-LSNN achieves an 8.2 % increase in classification accuracy relative to the P-SNN model. In addition to improved accuracy, the P-LSNN also shows a considerable power reduction. Specifically, the average energy utilization of the P-LSNN (as shown in Fig. 6) is reduced by 28.7 % compared to P-SNN and by 83.2 % when compared to the P-ANN baseline [41]. Our experiments also reveal that, on average, our proposed LSG reduced the power consumption by 57.6% compared to previous works [32] resulting in a highly energy-efficient printed spiking neuromorphic circuit. Also, the large capacitors ($10\,\mu F$) in the P-SNNs [32] significantly increased area requirements. By reducing capacitor values from $10\,\mu F$ to $10 - 100\,nF$ range, the proposed P-LSNN's average circuit area is significantly reduced by 89%, which is much smaller than the P-SNN and even smaller than the P-ANN ($2.46\,cm^2$). Thus, the remaining trade-off for the P-LSNN model is the increased training time—averaging 16.39 hours, compared to 1.32 hours for the P-ANN and 14.31 hours for the P-SNN. Despite this longer training duration, the considerable gains in power efficiency, reduced area footprint, and improved robustness justify the additional training cost.

Furthermore, the robustness was also evaluated under 10% process variation by injecting noise into the network weights and learnable circuit components. As expected, both P-SNN and P-LSNN models exhibited a slight degradation in performance. The P-SNN showed an average drop of 1.0%, while the P-LSNN experienced 4.9% reduction. This is attributed to the fact that, for P-SNN, the SG circuit components were excluded from noise injection, whereas in the P-LSNN, all learnable parameters were subject to variation. Despite this, the average accuracy of the proposed approach under variation remained 3.3% higher than that of the baseline without variation, underscoring the robustness and generalization capacity of the proposed approach.

Overall, the integration of adaptive parameters into the LSG not only improves accuracy and power efficiency but also improves the circuit adaptability with respect to the spike-timing events and also offers resilience against practical variability, making P-LSNN a promising candidate for ultra-low-power adaptive neuromorphic applications.

VI. Conclusion

Designing spiking circuits that are both energy-efficient and adaptable is especially difficult on printed electronics, whose low-cost processes introduce wide device variations and tight power constraints. To address this challenge, we proposed an analog printed spiking neural network with a learnable spike generator trained under a robustness-aware framework.

Across thirteen public benchmarks, the learnable spike generator lowers its own average power by 57.6 % relative to prior spike-generation circuits. Furthermore, the learnable printed spiking neural network trims energy per inference by 28.7 % compared with the non-adaptive printed spiking neural network and raises average classification accuracy from 75 % to 83 %. These gains come with a reduction in area, making the design more compact, and only a small increase in offline training time, which remains acceptable for ultra-low-power printed hardware. Under a ± 10 % process-variation test, the proposed network still outperforms the baseline by 3.3 % without variation, showing that the learned circuit parameters generalise beyond nominal conditions.

Future work may integrate online learning for real-time adaptation to changing conditions and extend the framework to fully event-driven, asynchronous operation, further enhancing the practicality of printed spiking systems for always-on edge and IoT applications.

Acknowledgment

This work has been supported by the European Research Council (ERC) (Grant No. 101052764) and the European High Performance Computing Joint Undertaking (HammerHAI, Grant No. 101234027).

REFERENCES

[1] L. Meng *et al.*, "Soft and flexible material-based affinity sensors," *Biotechnology advances*, vol. 39, p. 107398, 2020.

[2] J. Chang *et al.*, "Challenges of Printed Electronics on Flexible Substrates," in *2012 IEEE 55th international midwest symposium on circuits and systems (MWSCAS)*. IEEE, 2012, pp. 582–585.

[3] L. Wang and J. Liu, "Ink spraying based liquid metal printed electronics for directly making smart home appliances," *ECS Journal of Solid State Science and Technology*, vol. 4, no. 4, p. P3057, feb 2015.

[4] A. McLister, J. McHugh, J. Cundell, and J. Davis, "New developments in smart bandage technologies for wound diagnostics," *Advanced Materials*, vol. 28, no. 27, pp. 5732–5737, 2016.

[5] J. Backus, "Can programming be liberated from the von neumann style? a functional style and its algebra of programs," *Communications of the ACM*, vol. 21, no. 8, pp. 613–641, 1978.

[6] E.-I. Vatajelu, G. Di Natale, and L. Anghel, "Special session: Reliability of hardware-implemented spiking neural networks (snn)," in *2019 IEEE 37th VLSI Test Symposium (VTS)*, 2019, pp. 1–8.

[7] M. Eslaminia and S. L. Beux, "A shared synapse architecture for all-optical spiking neural networks," in *2023 21st IEEE Interregional NEWCAS Conference (NEWCAS)*, 2023, pp. 1–5.

[8] A. Lebanov, M. V. Lopez, F. De Roose, N. P. Papadopoulos, G. Indiveri, A. Rubino, M. Payvand, S. Smout, M. Willegems, F. Catthoor, J. Genoe, P. Heremans, and K. Myny, "Flexible unipolar igzo transistor-based integrate and fire neurons for spiking neuromorphic applications," *IEEE TBioCAS*, vol. 18, no. 1, pp. 200–214, 2024.

[9] J. Misra and I. Saha, "Artificial neural networks in hardware: A survey of two decades of progress," *Neurocomputing*, vol. 74, no. 1, pp. 239–255, 2010, artificial Brains.

[10] A. Hazan and E. Ezra Tsur, "Neuromorphic neural engineering framework-inspired online continuous learning with analog circuitry," *Applied Sciences*, vol. 12, no. 9, 2022.

[11] Z. Cui, *Printed electronics: materials, technologies and applications*. John Wiley & Sons, 2016.

[12] D. Hu, X. Zhang, Z. Xu, S. Ferrari, and P. Mazumder, "Digital implementation of a spiking neural network (snn) capable of spike-timing-dependent plasticity (stdp) learning," in *14th IEEE International Conference on Nanotechnology*. IEEE, 2014, pp. 873–876.

[13] M. S. Asghar, S. Arslan, and H. Kim, "A low-power spiking neural network chip based on a compact lif neuron and binary exponential charge injector synapse circuits," *Sensors*, vol. 21, no. 13, p. 4462, 2021.

[14] J. S. Chang, A. F. Facchetti, and R. Reuss, "A circuits and systems perspective of organic/printed electronics: Review, challenges, and contemporary and emerging design approaches," *IEEE Journal on emerging and selected topics in circuits and systems*, vol. 7, no. 1, pp. 7–26, 2017.

[15] V. Tischler *et al.*, "An integrate-and-fire neuron circuit made from printed organic field-effect transistors," *Organic Electronics*, vol. 113, p. 106685, 2023.

[16] D. D. Weller *et al.*, "Realization and Training of an Inverter-based Printed Neuromorphic Computing System," *Scientific reports*, vol. 11, no. 1, pp. 1–13, 2021.

[17] H. Zhao *et al.*, "Aging-Aware Training for Printed Neuromorphic Circuits," in *IEEE/ACM ICCAD*, 2022.

[18] ——, "Power-aware training for energy-efficient printed neuromorphic circuits," in *42nd IEEE/ACM ICCAD*, 2023.

[19] P. Pal *et al.*, "Print-safe: Printed ultra-low-cost electronic x-design with scalable adaptive fault endurance," *ACM Transactions on Embedded Computing Systems*, pp. 1–1, 2025.

[20] P. Pal, F. Afentaki, H. Zhao, G. Saglam, M. Hefenbrock, G. Zervakis, M. Beigl, and M. B. Tahoori, "Fault sensitivity analysis of printed bespoke multilayer perceptron classifiers," in *2024 IEEE European Test Symposium (ETS)*. IEEE, 2024, pp. 1–6.

[21] P. Pal, H. Zhao, T. Gheshlaghi, M. Hefenbrock, M. Beigl, and M. B. Tahoori, *Neural Architecture Search for Highly Bespoke Robust Printed Neuromorphic Circuits*. New York, NY, USA: Association for Computing Machinery, 2025. [Online]. Available: https://doi.org/10.1145/3676536.3676812

[22] Z. Cui, *Printed electronics: materials, technologies and applications*. John Wiley & Sons, 2016.

[23] J. Handy, "Why are computer chips so expensive?" 2014.

[24] S. Chung *et al.*, "All-inkjet-printed organic thin-film transistor inverter on flexible plastic substrate," *IEEE EDL*, vol. 32, no. 8, pp. 1134–1136, 2011.

[25] F. Shao *et al.*, "Recent progress on jet printing of oxide-based thin film transistors," *Journal of Physics D: Applied Physics*, vol. 52, no. 14, p. 143002, 2019.

[26] P. Pal *et al.*, "Efficient analog error correction for printed unary-encoded computing," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pp. 1–1, 2025.

[27] S. K. Garlapati *et al.*, "Ink-Jet Printed CMOS Electronics from Oxide Semiconductors," *Small*, vol. 11, no. 29, pp. 3591–3596, 2015.

[28] F. Rasheed, "Compact modeling and physical design automation of inkjet-printed electronics technology," Ph.D. dissertation, Dissertation, Karlsruhe, Karlsruher Institut für Technologie (KIT), 2020.

[29] C. Marques *et al.*, "Digital power and performance analysis of inkjet printed ring oscillators based on electrolyte-gated oxide electronics," *Applied Physics Letters*, vol. 111, no. 10, p. 102103, 09 2017.

[30] T. Gheshlaghi *et al.*, "Adapt-pnc: Mitigating device variability and sensor noise in printed neuromorphic circuits with so adaptive learnable filters," in *2025 Design Automation and Test in Europe (DATE)*, 2025, pp. 1–6.

[31] F. Rasheed *et al.*, "Variability modeling for printed inorganic electrolyte-gated transistors and circuits," *IEEE TED*, vol. 66, no. 1, pp. 146–152, 2018.

[32] P. Pal *et al.*, "Analog printed spiking neuromorphic circuit," in *2024 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2024, pp. 1–6.

[33] B. Li *et al.*, "Build Reliable and Efficient Neuromorphic Design with Memristor Technology," in *24th ASPDAC Proceedings*, 2019, pp. 224–229.

[34] V. Vadde *et al.*, "Power Efficient ReLU Design for Neuromorphic Computing Using Spin Hall Effect," *arXiv preprint arXiv:2303.06463*, 2023.

[35] Y.-H. Chen *et al.*, "Eyeriss: A Spatial Architecture for Energy-Efficient Dataflow for Convolutional Neural Networks," *ACM SIGARCH computer architecture news*, vol. 44, no. 3, pp. 367–379, 2016.

[36] Z. Zhai *et al.*, "Temporal-rate encoding to realize unary positional representation in spiking neural systems," *CoRR*, vol. abs/2012.09912, 2020.

[37] Y. Chauvin and D. E. Rumelhart, *Backpropagation: Theory, Architectures, and Applications*. Psychology press, 2013.

[38] J. Devlin *et al.*, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[39] L. Ouyang *et al.*, "Training Language Models to Follow Instructions with Human Feedback," *Advances in Neural Information Processing Systems*, vol. 35, pp. 27 730–27 744, 2022.

[40] F. Rasheed *et al.*, "Variability Modeling for Printed Inorganic Electrolyte-gated Transistors and Circuits," *IEEE transactions on electron devices*, vol. 66, no. 1, pp. 146–152, 2018.

[41] H. Zhao *et al.*, "Power-Aware Training for Energy-Efficient Printed Neuromorphic Circuits," in *42nd IEEE/ACM ICCAD*, 2023.

[42] E. Sowade *et al.*, "All-inkjet-printed thin-film transistors: manufacturing process reliability by root cause analysis," *Scientific reports*, vol. 6, no. 1, p. 33490, 2016.

[43] A. Mao *et al.*, "Cross-entropy Loss Functions: Theoretical Analysis and Applications," in *ICML*. PMLR, 2023, pp. 23 803–23 828.

[44] D. P. Kingma and M. Welling, "Auto-Encoding Variational Bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[45] A. Paszke *et al.*, "Pytorch: An Imperative Style, High-performance Deep Learning Library," in *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 2019, pp. 8024–8035.

[46] J. K. Eshraghian *et al.*, "Training Spiking Neural Networks Using Lessons from Deep Learning," *arXiv preprint arXiv:2109.12894*, 2021.

[47] D. P. Kingma *et al.*, "Adam: A Method for Stochastic Optimization," *arXiv preprint arXiv:1412.6980*, 2014.