ESTU: Enabling Spiking Transformers on Ultra-Low-Power FPGAs

Gianluca Leone, Paola Busia, Mauro Orrú, Luigi Raffo, *Member, IEEE*, and Paolo Meloni, *Member, IEEE*

***Abstract*—Spiking Neural Networks (SNNs) are computational efficient algorithms that have proven to be very effective on edge applications. Recent efforts of the scientific community have investigated the integration of the attention layer into Spiking Neural Networks to close the accuracy gap with non-spiking approaches. As effective, hardware implementations of spiking transformers have remained limited to high-end FPGA platforms so far, with power requirements incompatible for low-power applications. This work introduces ESTU, a compact spiking transformer hardware architecture tailored for edge deployment. ESTU has been designed with resource minimization in mind in order to fit in power and resource constrained FPGAs, as the Lattice iCE40UP5K. We validated ESTU on available Spiking Transformer models at the State of the Art and on a sEMG classification application addressing NinaPro DB-5, demonstrating improved accuracy of over 2% compared to the results of a vanilla SNN, fully unlocking spiking transformers capability at the edge at the cost of 3.76 mW during inference.**

***Index Terms*—Spiking transformers, spiking neural networks, tiny transformers, edge AI, FPGA, low power.**

I. Introduction

SPIKING Neural Networks (SNNs) have emerged as a promising solution for ultra-efficient AI-inference on portable computing platforms [1]. Remarkable results can be achieved on neuromorphic ASICs [2], [3] or when the event-driven computing paradigm is emulated on FPGAs [4]. The efficiency of SNNs stems from their biologically inspired design: 1) Neurons process spikes, enabling the exploitation of spatio-temporal event sparsity; 2) As spikes are binary, multiply-and-accumulate operations in conventional artificial neural networks can be replaced with additions; 3) Moreover, neural memory enables efficient processing of time series while reducing memory footprint [5], [6]. Because of such advantages, a promising research line aims at deploying SNNs on low-power FPGAs, as an easy-access and adaptable target complying with the needs of low-power wearable computing.

Received 6 August 2025; revised 5 September 2025 and 10 October 2025; accepted 24 October 2025. Date of publication 27 October 2025; date of current version 26 November 2025. This work was supported in part by the Key Digital Technologies Joint Undertaking (KDT JU) in EdgeAI "Edge AI Technologies for Optimized Performance Embedded Processing" Project under Grant 101097300; in part by European Union's Horizon 2020 Research and Innovation Program under Grant 101140052 (H2TRAIN); and in part by the NextGenerationEU Mission 4, Component 2, Investment 1.5, Project METBIOTEL–Innovation Ecosystem ECS 0000024 in Rome Technopole Spoke 1 and Spoke 6 under Grant CUP B83C22002820006. This brief was recommended by Associate Editor A. Yan. *(Corresponding author: Gianluca Leone.)*

The authors are with the Department of Electrical and Electronic Engineering (DIEE), University of Cagliari, 09123 Cagliari, Italy (e-mail: gianluca.leone94@unica.it).

Digital Object Identifier 10.1109/TCSII.2025.3626209

However, SNNs' accuracy still lags behind traditional AI methods, despite significant research efforts aimed at closing this gap [7]. A groundbreaking contribution in this direction is Spikformer [8], which surpasses the accuracy of conventional SNN models by combining SNNs with transformers. The core innovation is a lightweight Spiking Self-Attention (SSA), which employs binary representations for the Query (Q), Key (K), and Value (V) projections and eliminates the softmax.

Several recent works have explored the implementation of spiking transformer architectures on FPGAs [9], [10], [11], [12], [13]. Some of these studies do not disclose architectural implementation details [9], limiting reproducibility and evaluation. Some works, such as [10], focus on the SSA layer performing a direct one-to-one mapping of matrix multiplications on a Xilinx Zynq 7000 platform, leveraging the binary nature of spikes but entirely neglecting sparsity exploitation. Similarly, the HLS-based design in [11], deployed on a high-end Xilinx Alveo U280 card, employs a large number of DSPs to implement adder trees, achieving a very high throughput of 28.98 TOPS while consuming 70.93 W, but still fails in leveraging sparsity.

In contrast, other works accounted for spike sparsity. The architecture in [12], implemented on a Xilinx Virtex Ultra FPGA, encodes the positions of active spikes in both the Q and K matrices and iterates only over non-zero elements, comparing their indices. Finally, [13] is the only work explicitly aimed at edge deployment. It targets a Xilinx ZCU102 FPGA with a sparsity-aware architecture for large-scale spiking language models and reports the lowest power consumption among the reviewed works, requiring 1.2 W.

Unfortunately, all these works present power/resource budget inaccessible for severely power-constrained applications.

In this work, we aim to bring the advantages of spiking transformer models directly to near-sensor processing, a domain traditionally dominated by microcontrollers. To this end, we introduce ESTU,[1] an open-source[2] ultra-low-power spiking transformer hardware architecture optimized for low-power resource-constrained FPGAs. ESTU exploits hardware reuse, as different transformer layers are executed on the same hardware, reprogrammed by means of microcode instructions. Thus, ESTU can execute spiking transformers for near-sensor data analysis on a tiny device, matching the potential of FPGA-based execution with highly-constrained power/energy/cost budgets. The contribution of this work is threefold:

[1]ESTU: northwest wind in Sardinian.

[2]https://github.com/EOLAB-2025/ESTU.git

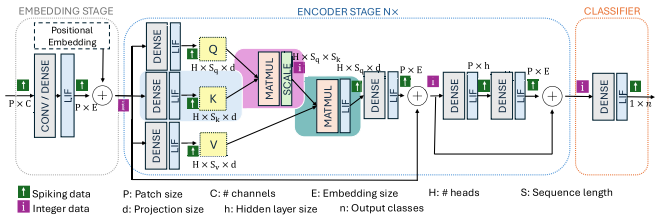

Fig. 1. General architecture of a spiking transformer model.

- ESTU is the first low-power spiking transformer hardware architecture tailored for end-to-end sensor data analysis, operating within a power envelope of just 3.76 mW;
- ESTU puts together a microcode-programmable and sparsity-aware architecture, a RISC-V softcore, and configurable encoding and decoding slots, ensuring self-sufficiency and flexibility across diverse domains.
- We demonstrated that tiny spiking transformer models, such as those deployable on ESTU, can improve the classification accuracy over vanilla SNNs on an sEMG use case, and showed the flexibility of ESTU by hosting two additional models available at the state of the art.

II. Spiking Transformers

Figure 1 illustrates a generic spiking transformer architecture based on the lightweight SSA mechanism introduced in [8]. The architecture is inspired by the Vision Transformer model, whose computational workload is unevenly distributed among three processing stages: the *embedding stage*, where the input image/signal is organized into a sequence of patches/tokens, the *encoder stage*, where the attention mechanism is applied, and the final *classifier stage*.

The most relevant contribution to the workload is represented by the *encoder stage*, and in particular by the Multi-Head Attention layer. As demonstrated in [8] and [14], spiking implementations of the attention can neglect the impact of the softmax operation and are reduced to:

$$Attention(\boldsymbol{Q},\boldsymbol{K},\boldsymbol{V}) = \frac{QK^T}{scale}\boldsymbol{V} \tag{1}$$

where Q, K, and V represent different binarized projections of the input sequence, on a certain number of independent heads H. The length of the sequence S_x is usually the same for the three projections; however, recent works have introduced online mechanisms where the current instance of Q is evaluated against longer sequences of K and V, providing memory of the previously processed inputs, such as in [15].

It is worth noting that, as spiking transformers represent a relatively new class of models, it is expected that the underlying algorithms, particularly the structure and ordering of attention mechanisms, or the particular embedding approach based on the input to be processed, will continue to evolve. Therefore, in this work, we abstracted the model into a sequence of basic operators shown in Table I, whose different combinations and supported input sizes enable the execution of a wide range of transformer models. This operator-level decomposition forms the foundation of ESTU design, enabling flexible execution at the edge.

TABLE I
Description of Supported Operations

Operation	Description
`Dense(spike)`	Dense spiking layers
`Dense(int)`	Dense spiking layers with integer inputs
`Sum(spike, spike)`	Element-wise sums between binary arrays
`Sum(spike, int)`	Element-wise sums between binary and non-binary input arrays
`Mul(spike, spike)`	Matrix multiplications between spikes
`Mul(spike, int)`	Matrix multiplications between integers and spikes

III. Hardware Architecture

The system architecture, shown in Fig. 2, includes:

- Configurable encoding slot used to preprocess the sensor data and obtain spike trains;
- Micro-code configurable accelerator core (ESTU), in charge of executing sparse spiking workload;
- Configurable decoding slot capable of translating output spike trains into classification outcome;
- RISC-V softcore to handle input/output data flow, power management, and architecture parameters.

While the proposed design is, in principle, portable to any reconfigurable platform, current deployment is on a Lattice iCE40UP5K FPGA. This is part of the overall ESTU design strategy, aimed at ultra-low power consumption: (a) focus on hardware resource minimization, enabling deployment on resource- and energy-constrained devices. We implemented an operator fusion strategy, designing ESTU's datapath so that the logic in common between operators could be shared. (b) Architectural support for sparsity exploitation, easily and flexibly exploitable to minimize energy consumption by setting the system in idle mode in the absence of events.

A. Estu Architecture

ESTU is based on a microcode-programmable computational core, whose hardware structures are reused to execute all the operations in Table I. Microcode is stored in a dedicated instruction memory. Inside the core, spike and integer memories buffer intermediate results locally. The instruction and spike memories are mapped onto multiple BRAM macros, each of 4 Kbit, while the integer memory leverages larger Single-Port RAMs (SPRAMs), each providing 16Kx16 bits. Memory buffers are connected by means of a configurable interconnect to the processing logic in ESTU, which includes:

- A bank of 16 `AND`gates involved in spike-weight and spike-spike multiplications;
- Two multipliers used for `Dense(int)`operation;
- An addition block including 2 16-bit adders used for `Dense(spike,int)`and `Sum`, a 2-input 20-bit accumulator, used by `Dense`and `Mul`, and a 16-input pop-counter used in `Mul(spike, spike)`.

Any operation result can be routed directly to the spike or integer memories, or passed through a Leaky Integrate-and-Fire (LIF) module, which binarizes the result according to:

$$\begin{aligned} v(t) &= \alpha v(t-1) + (1-\alpha)i(t); \\ s(t) &= v(t) > \theta; \\ v(t) &= v(t) \cdot (1 - s(t)) \end{aligned} \tag{2}$$

With $i(t)$ the input current, $v(.)$ the LIF neuron *membrane potential*, α the *decay*, and θ the *spike threshold*. This neuron module can compute neuron dynamics for all the neurons in

TABLE II
IMPACT OF SPIKE GROUPING ON SPARSITY AND STACK MEMORY

Group size	Software sparsity	Exploited sparsity	Stack size [bit]	BRAMs on iCE40
1	0.95	0.95	179 Kb	44 (147%)
2	0.95	0.90	83 Kb	22 (73%)
4	0.95	0.82	38 Kb	10 (33%)
8	0.95	0.66	18 Kb	5 (17%)

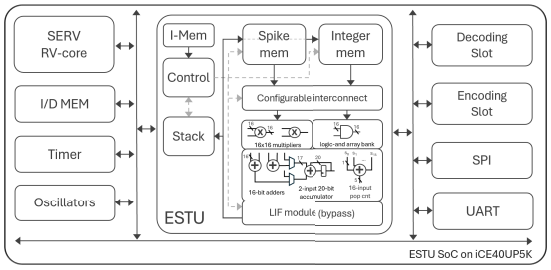

Fig. 2. Overview of the ESTU SoC architecture.

the transformer topology. The overall computational core can be reused to execute all the transformer layers, according to the schedule required for the inference execution.

B. Event Sparsity Exploitation

ESTU includes a stack memory that, when possible, keeps track of neuron activity, to limit fetching and processing only when spikes are emitted. However, when spikes are processed in parallel, sparsity exploitation is more complex, as it leads to random memory access patterns that hardly match the organization of memory macros in the FPGA.

Thus, to fit in the Lattice iCE40UP5K FPGA, featuring 30 4-Kb BRAMs, we adopt the following strategy: spikes are grouped in blocks of four, and the stack memory stores only pointers to groups that contain at least one active element. This lowers memory requirements for the stack, at the price of reducing the effective sparsity that can be exploited by ESTU. This trade-off is shown in Table II for an SSA configuration with 4 attention heads, embedded tokens with 16 dimensions, 200 time steps, and a sparsity of 0.95.

C. Flexibility and Microcode

Different flavours of spiking transformers can be deployed on ESTU by changing the microcode and defining different settings and sequences of operators. Microcode instructions are 169 bits wide and indicate opcodes, sources and destination addresses, input and output sizes, stack address, and LIF parameters. Figure 3 shows how different operators are executed in succession, using the processing of one of the attention heads as an example, highlighted in Figure 1.

At every step, the source data and the output results are read and written from the local spike and integer memories and routed towards the processing elements through the configurable interconnect module. Depending on the operation at hand, different paths can be activated. Memory addresses must be set in the microcode to implement the connections between the operators. The input of the attention layer *Attn input* in Figure 3 is processed by a `Dense(int,int)` layer that reads both its input from the integer memory and employs two DSP-based multipliers and an accumulator to evaluate the LIF

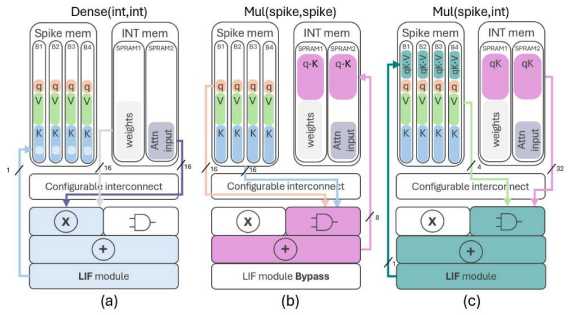

Fig. 3. Data flow of the three operations highlighted in Figure 1: (a) `Dense(int,int)`; (b) `Mul(spike,spike)`; (c) `Mul(spike,int)`.

neuron input current. The output is stored in the spike memory in the K matrix region. Both the spike and integer memories are multi-banked to enable independent parallel accesses to multiple inputs and to boost throughput, depending on the operation at hand. The following `Mul(spike,spike)` activates both the spike memory read ports and bulky process 16 pairs of spikes through the and-gate array and the pop-counter, bypasses the LIF module, and writes the result in the integer memory. Finally, the `Mul(spike,int)` reads from two different memories, LIF-binarizes the output of the matrix product, and writes the results in spike memory.

With this organization, ESTU supports different spiking attention schemes, such as, for instance, the one proposed in [16], featuring a different order of matrix multiplications.

D. Encoding/Decoding Slots and System-Level Control

ESTU is enclosed by configurable encoding and decoding slots, which serve as flexible interfaces between the spiking domain and the specific application context. On the input side, the encoding slot continuously converts sensor data streams into spikes, and on the output side, the decoding slot aligns with the target evaluation method, e.g., spike rate evaluation, to generate interpretable outcomes.

To maintain the minimalistic footprint of the accelerator design, the system integrates SERV,[3] a bit-serial RISC-V core that enables software control over several memory-mapped peripherals, such as SPI and UART interfaces, used to stream input/output, as well as to load the model during start-up.

Moreover, SERV grants control over power management. The unpredictable sparsity in the encoded data forces the system to be clocked for the worst-case scenario. Optimizing idle power consumption is therefore a key aspect of deploying neuromorphic algorithms on accelerators at the edge. To address this issue, we introduced a stand-by mode, fully controlled by SERV, in which the on-chip high-frequency oscillator is turned off, implementing global clock gating for the entire system. The wake-up mechanism is handled by a timer, driven by an additional on-chip low-frequency oscillator.

IV. RESULTS

A. Hardware Implementation, Performance and Limitations

Logic synthesis and implementation were carried out using an open-source architecture-neutral FPGA framework called OSS CAD Suite,[4] including Yosys Open SYnthesis Suite

[3]https://github.com/olofk/serv
[4]https://github.com/YosysHQ/oss-cad-suite-build

TABLE III
RESOURCE REQUIRMENT

LC	BRAM	SPRAM	DSP	Clock [MHz]
4,301 (81%)	30 (100%)	2 (50%)	6 (75%)	21

TABLE IV
THROUGHPUT PER OPERATOR SUPPORTED BY ESTU

Operator	ESTU [op/cycle]	SW-based [op/cycle]	Operator	ESTU [op/cycle]	SW-based [op/cycle]
Dense(spike)	4	1.17	Mul(spike,spike)	16	1.03
Dense(int)	2	1.17	Mul(spike,int)	4	1.03
Sum(spike, spike)	2	1	LIF	1	0.03
Sum(spike,int)	2	1	–	–	–

(Yosys) for logic synthesis, and Netxtpnr for place and route. Table III summarizes the overall resource requirements of the proposed architecture in the target Lattice iCE40UP5K FPGA while clocked at its maximum clock speed of 21 MHz. This maximum frequency stems partly from deploying on a resource-constrained 5k-LUT FPGA, which is easily congested, and partly from the use of an open-source toolchain that ensures accessibility, though optimizing less aggressively than commercial tools.

Table IV summarizes the throughput of each supported operator, expressed as the number of input pairs processed per clock cycle. ESTU supports the execution of the above operators, along with LIF neuron integration, using fixed-point arithmetic. The total inference time for a specific spiking transformer model executed on ESTU depends on the distribution of operator types, on the sparsity, and on the size of the input data, as shown by Equation 3:

$$T_{\text{inf}} = \sum_{i \in \text{Operators}} \frac{\text{Sparsity}_i \times \text{Data}_i}{\text{Throughput}_i} \tag{3}$$

Each term in the sum corresponds to the contribution of a specific operator type, accounting for the number of operations (after sparsity is applied), and the operator-specific throughput.

Finally, the maximum supported model size is constrained by the current memory allocations:

- Integer memory: the combined size of attention and synaptic weights must be less than 1 Mb.
- Spike memory: the stored spikes must not exceed 32 Kb.
- Potential memory: supports up to 768 LIF neurons.

To assess the benefits of on-FPGA execution, despite the limited frequency, Table IV reports the throughput evalutated for the same operators (using the sEMG use-case in Section IV-B) on a RISC-V based microcontoller[5] [17] in a similar power envelope. This experiment shows that the parallelism and the hardware-based flow/addressing/loop control enabled by ESTU significantly increase efficiency. Such improvement results in around halving the energy consumption for the considered case.

B. Benchmarking Models

Table V reports the performance of ESTU on benchmark spiking transformer models, highlighting the achievable processing time with respect to real-time constraints. The model in [15] is for fast prosthetic hand control, while the one in [14] targets epileptic seizure detection and prediction.

[5]The execution setup is with a single core, -O3 compiler optimization, 8-bit quantization, 65 MHz clock frequency, corresponding to 10 mW power consumption. Reported throughput for RISC-V is best-case, as sparsity checks are disabled, neglecting the related overhead.

TABLE V
BENCHMARKING ON SOA SPIKING TRANSFORMER MODELS

Use case	Head	Patch	Sequence	Constraint	Latency	Sparsity
sEMG [15]	8	32	150	0.5 ms	0.47 ms	60%
EEG [14]	1	8	24	3.91 ms	3.81 ms	70%
sEMG [our]	4	64	200	5 ms	0.65 ms	91%

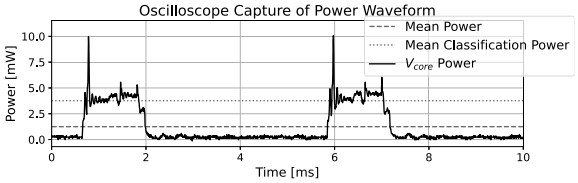

Fig. 4. Oscilloscope capture of instantaneous power consumption at 12 MHz.

Moreover, to provide a direct estimation of the benefits deriving from enabling attention mechanism in SNNs, we validated ESTU on a model trained to address the NinaPro DB-5 sEMG gesture classification problem [18]. We extracted spiking features out of the raw sEMG signal by applying the delta modulation algorithm as in [19]. Then, we used the SpikingJelly framework to train a spiking transformer with the same structure reported in Fig 1, with dense-based embedding and a single encoder stage inspired by the online attention implementation proposed in [15]: it features 4 attention heads, a memory of 200 time steps representing the sequence length of K and V, and projection size 16. Each time step processes spikes from 32 input channels, with an embedding size of 64.

We achieved a classification accuracy of 87.21%, over 2% higher than the one achievable with a comparable training procedure by the vanilla SNN model presented in [19]. Inference execution on the test set demonstrated 91% of the neurons are, on average, inactive. This sparsity level can be effectively exploited when running inference on ESTU, resulting in a 29% speedup, compared to the non-sparse required execution time. To enable deployment, the parameters of the model were post-training uniform-scale-quantized to 8-bit integer precision, whereas a 20-bit fixed-point representation was exploited for membrane potentials. This solution has a minimal impact on classification accuracy, with a small drop to 86.97%.

For the estimation of the execution time of our sEMG gesture recognition model we considered the system clocked at its maximum frequency of 21 MHz and accounted for the sparsity observed on the test set, while for the other models in Table V we report the minimum value of sparsity needed to meet the real-time constraint. The benchmarked models differ in terms of the number of attention heads, token embedding dimensions, and temporal resolution (i.e., the sequence length). Despite these differences, all models fit and can be executed by ESTU in real-time.

C. Power Consumption

We measured both active and standby power through a shunt resistor soldered on the V_{CORE} track of the iCEBreaker evaluation board, obtaining 3.76 mW and 0.23 mW, respectively, as shown in Figure 4.

TABLE VI
COMPARISON WITH THE STATE OF THE ART

Work	Platform	Node [nm]	Logic cells	DSP	RAM [MB]	Throughput [GOPS]	Power [W]	Energy Efficiency [GOPS/J]
Our work	iCE40UP5K	40	4.3k	6	0.08	0.192	0.004	51.06
[9]	VU37P	16	-	-	-	70.11	3.55	19.75
[10]	Zynq-7000	28	-	-	-	-	1.47	-
[11]	U280	16	503k	7,249	23.3	28,980	70.93	408.57
[12]	Virtex Ultra Scale	20	453k	-	3.5	307.2	12	25.6
[13]	ZCU102	16	90k	0	0.7	986.3	1.2	758.6

For these measurements, the system was clocked at 12 MHz, which is sufficient to meet real-time requirements in the sEMG worst-case scenario, resulting in 4.28 μJ per inference (with a 29 % saving thanks to sparsity exploitation) and an average power consumption of 1.03 mW.

V. COMPARISON WITH THE STATE OF THE ART

Table VI presents a comparative analysis of FPGA-based spiking transformer accelerators in the literature. Notably, most of these implementations do not prioritize real-time sensor data analysis, focusing on maximizing throughput on vision-datasets [10], [11], [12], [13] targeting larger FPGAs, which results in significantly higher power consumption.

Some of these works completely neglect sparsity [10], [11] or only focus on specific operators [10]. None of these approaches can be ported to small reconfigurable devices, as they do not reuse processing logic or buffers and simply instantiate multiple components for the different operators, resulting in significant memory and logic overhead. In contrast, ESTU approach, based on reuse and microcode, is, at the same time, the only one supporting reprogrammability and usable in low-power FPGAs, achieving 3.76 mW power dissipation in inference. It serves the purpose for portable/wearable near-sensor processing, while [13], although aimed at edge deployment, reports a power consumption of 1.2 W, 319 times higher.

Even though ESTU's peak performance is the lowest in the table, it was sufficient for typical SNNs with SSA as shown in Section IV-B. Nonetheless, despite the overhead of control logic on such a tiny accelerator, our energy efficiency is still 2.6 times higher than [9] and 2 times higher than [12]. High-end and high-cost 16 nm platforms enable significantly higher efficiency (14.86× [13] and 8× [11]), implementing a much higher degree of parallelism. However, literature reports for such platforms values of quiescent power exceeding 0.6 W [20] and 25 W [21] respectively, 2 and 3 orders of magnitude higher than ESTU's inference power, indicating that they target application domains where ultra-low-power is not a primary concern.

VI. CONCLUSION

We presented ESTU, an end-to-end spiking transformer hardware architecture tailored for low-end, low-power FPGAs. ESTU enhances flexibility and reusability by integrating a microcode-programmable accelerator, a RISC-V softcore, and two customizable encoding and decoding slots.

To the best of our knowledge, ESTU is the first architecture to enable spiking attention at the edge, achieving a power consumption of just 3.76 mW, paving the way for deploying accurate spiking models in deeply embedded systems.

REFERENCES

[1] R. Li et al., "Real-time sub-milliwatt epilepsy detection implemented on a spiking neural network edge inference processor," *Comput. Biol. Med.*, vol. 183, Dec. 2024, Art. no. 109225.

[2] G. Tang et al., "SENECA: Building a fully digital neuromorphic processor, design trade-offs and challenges," *Frontiers Neurosci.*, vol. 17, 2023, Art. no. 1187252, doi: 10.3389/fnins.2023.1187252.

[3] A. Vitale, E. Donati, R. Germann, and M. Magno, "Neuromorphic edge computing for biomedical applications: Gesture classification using EMG signals," *IEEE Sensors J.*, vol. 22, no. 20, pp. 19490–19499, Oct. 2022.

[4] G. Leone, M. Antonio Scrugli, L. Badas, L. Martis, L. Raffo, and P. Meloni, "SYNtzulu: A tiny RISC-V-controlled SNN processor for real-time sensor data analysis on low-power FPGAs," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 72, no. 2, pp. 790–801, Feb. 2025.

[5] P. Busia, G. Leone, A. Matticola, L. Raffo, and P. Meloni, "Wearable epilepsy seizure detection on FPGA with spiking neural networks," *IEEE Trans. Biomed. Circuits Syst.*, early access, May 30, 2025, doi: 10.1109/TBCAS.2025.3575327.

[6] L. Martis, G. Leone, L. Raffo, and P. Meloni, "Low-power FPGA-based spiking neural networks for real-time decoding of intracortical neural activity," *IEEE Sensors J.*, vol. 24, no. 24, pp. 42448–42459, Dec. 2024.

[7] E. O. Neftci, H. Mostafa, and F. Zenke, "Surrogate gradient learning in spiking neural networks: Bringing the power of gradient-based optimization to spiking neural networks," *IEEE Signal Process. Mag.*, vol. 36, no. 6, pp. 51–63, Nov. 2019.

[8] Z. Zhou et al., "Spikformer: When spiking neural network meets transformer," 2022, *arXiv:2209.15425*.

[9] H. Vishwamith, M. Isik, and I. C. Dikmen, "HPCNeuroNet: Advancing neuromorphic audio signal processing with transformer-enhanced spiking neural networks," in *Proc. 4th Interdiscipl. Conf. Electrics Comput. (INTCEC)*, Jun. 2024, pp. 1–7.

[10] Z. Song, P. Katti, O. Simeone, and B. Rajendran, "Stochastic spiking attention: Accelerating attention with stochastic computing in spiking networks," in *Proc. IEEE 6th Int. Conf. AI Circuits Syst. (AICAS)*, Apr. 2024, pp. 31–35.

[11] Y. Gao et al., "Advancing neuromorphic architecture toward emerging spiking neural network on FPGA," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 44, no. 9, pp. 3465–3478, Sep. 2025.

[12] Z. Li, W. Mao, S. Zhang, Q. Dong, and Z. Wang, "An efficient sparse hardware accelerator for spike-driven transformer," in *Proc. IEEE Asia–Pacific Conf. Appl. Electromagn. (APACE)*, Dec. 2024, pp. 250–253.

[13] C. Du, Q. Wen, Z. Wei, and H. Zhang, "Energy efficient spike transformer accelerator at the edge," *Intell. Mar. Technol. Syst.*, vol. 2, no. 1, p. 24, Sep. 2024.

[14] Q. Chen, C. Sun, C. Gao, and S.-C. Liu, "Epilepsy seizure detection and prediction using an approximate spiking convolutional transformer," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 2024, pp. 1–5.

[15] N. Leroux, J. Finkbeiner, and E. Neftci, "Online transformers with spiking neurons for fast prosthetic hand control," in *Proc. IEEE Biomed. Circuits Syst. Conf. (BioCAS)*, Oct. 2023, pp. 1–6.

[16] M. Yao et al., "Spike-driven transformer v2: Meta spiking neural network architecture inspiring the design of next-generation neuromorphic chips," 2024, *arXiv:2404.03663*.

[17] E. Flamand et al., "GAP-8: A RISC-V SoC for AI at the edge of the IoT," in *Proc. IEEE 29th Int. Conf. Appl.-Specific Syst., Archit. Processors (ASAP)*, Jul. 2018, pp. 1–4.

[18] S. Pizzolato, L. Tagliapietra, M. Cognolato, M. Reggiani, H. Müller, and M. Atzori, "Comparison of six electromyography acquisition setups on hand movement classification tasks," *PLoS ONE*, vol. 12, no. 10, Oct. 2017, Art. no. e0186132.

[19] M. A. Scrugli, G. Leone, P. Busia, L. Raffo, and P. Meloni, "Real-time sEMG processing with spiking neural networks on a low-power 5K-LUT FPGA," *IEEE Trans. Biomed. Circuits Syst.*, vol. 19, no. 1, pp. 68–81, Feb. 2025.

[20] Y. Kim, H. Kim, N. Yadav, S. Li, and K. K. Choi, "Low-power RTL code generation for advanced CNN algorithms toward object detection in autonomous vehicles," *Electronics*, vol. 9, no. 3, p. 478, Mar. 2020.

[21] A. Wadood, A. Lu, K. Zhang, and Z. Fang, "FORC: A high-throughput streaming FPGA accelerator for optimized row columnar file decoders in big data engines," in *Proc. 34th Int. Conf. Field-Program. Log. Appl. (FPL)*, Sep. 2024, pp. 318–324.