Proceedings of CONF-FMCE 2026 Symposium: Smart City and Infrastructure Engineering
DOI: 10.54254/2755-2721/2026.CH37367

Spiking Neural Network Chips for Low-Power Real-Time Edge Intelligence

Leyang Qin

Ulink College of Shanghai, Shanghai, China
Leyang.qin.0086@ulink.cn

Abstract. Edge devices require short response time and low active and idle power, yet conventional neural-network hardware repeatedly moves weights and activations between memory and processing units. Spiking neural network (SNN) chips address this problem with event-driven communication, persistent neuron state, and local synaptic storage. This review first explains the signal path of an SNN chip through four connected modules: spike encoding, neuron computation, synaptic learning, and output decoding. It then compares three implementation routes: digital many-core platforms, mixed-signal or sensor-integrated platforms, and memristive in-memory hardware. Representative systems include TrueNorth, Loihi, Tianjic, Darwin3, ODIN, DYNAP-SE, and Speck. The comparison shows that no single metric is enough. Large digital chips provide scale and programmability but still incur memory and routing costs; mixed-signal and sensor-integrated designs reduce unnecessary activity but require attention to device mismatch and input-event rate; memristive arrays reduce weight movement but remain sensitive to variation and peripheral overhead. Meaningful evaluation therefore requires accuracy, timestep count, event activity, latency, active and idle power, and a clear measurement boundary. These criteria help identify when SNN hardware offers a practical advantage for low-power real-time edge intelligence.

Keywords: spiking neural networks, neuromorphic computing, edge AI, machine learning

1. Introduction

Edge nodes face power, response-time, and memory limits at the same time. An industrial monitor or wearable may remain idle for long periods yet still need to react within a few milliseconds. For such a system, idle power matters alongside the energy and latency of active inference [1]. Conventional artificial-neural-network (ANN) accelerators achieve strong accuracy, but repeated movement of weights and activations can account for a substantial share of energy on a small device [2]. This operating regime favors architectures that keep state local and limit data movement while the input is quiet.

Spiking neural networks (SNNs) communicate with discrete events and maintain a time-dependent neuron state [3]. In many digital implementations, a received spike triggers an addition or table lookup instead of a dense multiply-accumulate operation [2]. Event-driven control can also gate inactive cores and routers. The saving, however, depends on the firing rate, fan-out, encoding

© 2026 The Authors. This is an open access article distributed under the terms of the Creative Commons Attribution License 4.0 (https://creativecommons.org/licenses/by/4.0/).

DOI: 10.54254/2755-2721/2026.CH37367

method, and inference duration; sustained activity or a long rate-coding window can still generate substantial memory and packet traffic.

Existing SNN hardware can first be divided by its underlying circuit implementation. Fully digital platforms use CMOS logic and memory for neuron updates, synaptic state, and event routing. TrueNorth demonstrates large-scale integration, Loihi adds programmable on-chip learning, Tianjic supports both Artificial Neuron Network (ANN) and SNN operators, and Darwin3 uses an instruction-set architecture with compressed connectivity [4-7]. Digital implementation offers reproducible behavior and mature design tools, but synaptic-memory access and packet routing can still dominate cost when networks or event rates grow.

A second route combines continuous-time or sensor-level processing with digital communication. DYNAP-SE implements neuron and synapse dynamics in mixed-signal circuits and uses asynchronous AER routing, whereas Speck places an event camera and asynchronous digital SNN processor on the same die [8, 9]. These platforms can avoid periodic work when the input is quiet, but their measured benefit depends on circuit mismatch, bias or calibration conditions, scene activity, and whether sensor power is included.

The third route stores synaptic weights as conductance states in memristive arrays and performs accumulation through the array response. This organization can reduce weight movement and directly support in-memory computation [2]. Its present limitations are different from those of CMOS-only chips: device drift, write noise, array non-idealities, and data-conversion or control circuits can narrow the system-level gain. The three routes therefore should not be ranked by neuron count or pJ/SOP alone; each trades implementation maturity, programmability, integration scope, and energy in a different way.

Although SNN-chip research has produced several mature prototypes, three bottlenecks remain. First, low firing activity does not guarantee low latency or low energy when inference uses many timesteps. Second, mixed-signal and emerging-memory devices introduce variation and peripheral overhead that are often absent from device-level efficiency figures. Third, workloads, operation definitions, and measurement boundaries differ across papers, which makes headline comparisons unreliable. This review therefore examines SNN chips for low-power real-time edge use and emphasizes the conditions under which their advantages can be interpreted.

The remainder of the paper is organized as follows. Section 2 explains the operating principle of an SNN chip through spike encoding, neuron computation, synaptic learning, and output decoding. Section 3 reviews representative digital, mixed-signal or sensor-integrated, and memristive implementations. Section 4 discusses temporal cost, device non-idealities, peripheral overhead, and evaluation practice. Section 5 concludes the review.

2. Operating principle of SNN chip

An SNN chip can be understood as a signal path with four connected modules. The encoder converts an input into spike events; synapses store connectivity and scale those events; neurons integrate the resulting input and emit new spikes; and a decoder converts the output activity into a task decision. Learning determines the synaptic weights used along this path. Separating the four modules makes it possible to relate an algorithmic choice directly to memory access, event traffic, latency, and circuit cost [2, 3].

DOI: 10.54254/2755-2721/2026.CH37367

2.1. Spike encoding

Spike encoding determines how input information is represented in time. Rate coding represents a value by the number or frequency of spikes within an inference window, while temporal schemes use quantities such as the first-spike time. An event sensor provides a third case: it generates asynchronous events directly from changes in the physical signal instead of first assembling a dense frame. The encoding choice affects both information fidelity and the traffic presented to the chip [2]. Let T denote the number of inferences timesteps, r the average input-event rate per timestep, and F the average mapped fan-out of an event. Before multicast reuse and pruning are considered, the number of requested synaptic accesses scales approximately with T*r*F. A larger T can improve the resolution of a rate-coded estimate, but it also extends decision latency and creates more opportunities for state-memory access and routing. For this reason, an accuracy value is difficult to interpret unless the encoding method, T, and the resulting latency are reported together.

2.2. Neuron computation

Spiking neurons are the basic computing units of an SNN, and the leaky integrate-and-fire (LIF) model is widely used in hardware because it represents temporal integration with only a small amount of state. One continuous-time form of the model is given in equation (1):

$$\tau_m \frac{dV_m(t)}{dt} = -(V_m(t) - V_{rest}) + R_m I(t) \tag{1}$$

Here, $V_m(t)$ is the membrane potential, V_{rest} is the resting potential, τ_m is the membrane time constant, R_m is the membrane resistance, and $I(t)$ is the input current. When $V_m(t)$ reaches or exceeds the threshold V_{th}, the neuron emits a spike and its membrane state is reset. Digital cores usually store this state in registers or local Static Random-Access Memory (SRAM), whereas mixed-signal neurons may represent it as charge on a capacitor [3].

A received AER packet identifies the active input or synaptic row. The core then reads the relevant weights and membrane state, performs the update, writes the new state, and checks the firing condition. If a spike is generated, arbitration and multicast routing send the event to its destinations.

2.3. Synaptic learning

A synapse stores the connection strength applied when a presynaptic spike arrives. During inference, the weight may remain fixed; during learning, it is updated according to an offline or local rule. ANN-to-SNN conversion calibrates a trained ANN for spike-rate inference and often requires a relatively long T. Surrogate-gradient training treats the spike function with a differentiable approximation during off-chip training and can target shorter inference windows. Spike-timing-dependent plasticity (STDP), by contrast, changes a local weight according to the relative timing of pre- and postsynaptic spikes and is suitable for adaptation or smaller on-chip learning tasks [2].

DOI: 10.54254/2755-2721/2026.CH37367

2.4. Output decoding

Output decoding converts the activity of the final SNN layer into a task result. A rate-coded classifier may count output spikes over T and select the neuron with the largest count. A time-to-first-spike decoder instead selects the earliest output event, while a membrane-potential decoder reads the final neuron state even if no output spike occurs. The first method can trade a longer window for a more stable count; the second reduces nominal decision time but is sensitive to timing variation; the third adds a state-readout requirement. The reported latency and energy therefore depend on both the neuron dynamics and the chosen decoder [2].

Taken together, the four modules form one causal chain: encoding sets event density, synaptic organization determines weight access, neuron dynamics determine state updates and spike generation, and decoding determines when enough evidence has accumulated for a decision. A hardware comparison should keep these choices next to task accuracy, event count, T, active and idle power, and the boundary of the reported measurement.

3. Representative SNN hardware

3.1. Digital platforms

Representative digital SNN platforms span application-specific arrays and programmable many-core systems. TrueNorth's 28 nm chip achieves scale through a highly customized array of time-multiplexed neurosynaptic cores that store neuron state and synaptic configuration on chip; network training and parameter preparation are performed externally [4]. This reduces the control and storage needed for online updates. Its 63 mW video result is tied to a stated input resolution and frame rate, which is useful context for implementation studies. Reproducing the same power on another task would still depend on preprocessing, mapped connectivity, activity, and the required accuracy.

Loihi includes programmable local-learning rules and supports experiments in plasticity, online adaptation, and sparse graph computation [5]. The additional state and control broaden the range of mappings that can be studied on the chip. Tianjic takes another route by providing functional units for ANN and SNN computation [6]. In a heterogeneous edge workload, dense layers can remain in an ANN representation while temporal functions use spikes, which avoids imposing one encoding on every stage of the application.

Darwin3 addresses the storage and execution of large sparse networks. Its ten principal neuromorphic instructions, mesh of computing nodes, and compressed connection data support up to 2.35 million neurons [7]. The reported improvements include a code-density increase of as much as 28.3 times and mapped-memory reductions ranging from 6.8 to 200.8 times for convolutional SNNs. ODIN represents a much smaller capacity target: 256 neurons and 64k synapses with online learning occupy 0.086 mm^2 in 28 nm CMOS, and the paper reports 12.7 pJ/SOP [10]. Together, the two designs show how connectivity storage and learning circuitry change with the intended scale.

SpiNNaker follows a more programmable approach. It uses ARM cores and multicast packet routing to simulate a broad range of neuron and synapse models in real time [11]. The platform prioritizes model flexibility and software control, so its energy and real-time capacity depend on partitioning, routing-table use, and the workload assigned to each processor. When a study requires custom neuron dynamics, that programmability may matter more than the efficiency of a fixed-function datapath.

DOI: 10.54254/2755-2721/2026.CH37367

3.2. Mixed-signal and sensor-integrated platforms

DYNAP-SE combines subthreshold analog neuron and synapse circuits with asynchronous digital AER communication [9]. Each of its four cores contains 256 adaptive-exponential neurons, and content-addressable memory stores incoming connectivity. Continuous-time circuit dynamics remove the periodic numerical update used in a synchronous digital simulation. The effect is attractive for slowly varying or event-based inputs, although the operating range is set by bias conditions and device characteristics.

Mixed-signal neurons exhibit device mismatch in thresholds, time constants, and synaptic responses. In DYNAP-SE, the neuron and synapse operating points are set by programmable on-chip, temperature-compensated bias generators [9]. Mixed-signal energy results should therefore be accompanied by the resulting task accuracy and, where possible, its spread across devices; calibration effort should also be reported when calibration is used. Sensory classification and a tightly controlled feedback loop can tolerate different levels of variation.

Speck places a 128 × 128 dynamic vision array and 328k integrate-and-fire units on the same die. The processor resting power is reported as 0.42 mW. On the evaluated event datasets, the lowest reported real-time processor-only power for a sample was 0.70 mW; those measurements excluded the DVS power [8]. Changes at the sensor are passed directly into the asynchronous processing chain. For scenes with limited motion, this reduces input traffic and avoids frame formation before inference. The benefit will vary with scene activity, since a rapidly changing image produces a denser event stream.

3.3. Memristive and emerging-device implementations

Memristive crossbars store weights as conductances and accumulate column current during a read. Wang et al. implemented both neuronal and synaptic functions with memristive devices in an unsupervised pattern classifier [12]. A later fatigue-STDP array assigns short- and long-term behavior to an interfacial dynamic memristor and a hafnia-based one-transistor-one-memristor cell, respectively [13]. Both studies demonstrate learning and state behavior in physical devices, but their array sizes and peripheral assumptions differ from those of commercial-scale processors.

Two-dimensional materials, including α-In_2Se_3, MoS_2, and h-BN, have been investigated for multilevel conductance and vertically stacked devices [14]. Most results are still obtained from individual devices or relatively small arrays. At larger dimensions, selector requirements, line resistance, write verification, conversion circuits, and defective cells can influence both energy and accuracy [14, 15]. Those contributions are not captured by a device switching-energy figure alone.

For the present comparison, emerging-device results are grouped by the functions demonstrated and the level of integration reached. A processor-level assessment would also need array yield, endurance over the intended update schedule, peripheral energy, mapped-task accuracy after calibration, and measurements across multiple samples. The available device evidence is important, but the system-level data remain uneven.

4. Limitations and outlook

4.1. Temporal length and event activity

Sparse activity does not guarantee short or low-energy inference. A converted SNN may fire infrequently in each step and still execute a large total number of events over tens or hundreds of

DOI: 10.54254/2755-2721/2026.CH37367

steps. During that interval, neuron state is revisited, routers remain available, and the final decision is delayed. This affects reactive control as well as battery-powered sensing. A more informative report gives accuracy, energy, and latency over several values of T rather than selecting one firing window [2].

Reaching a short inference window usually requires decisions at both the algorithm and hardware levels. Training can account for the target weight precision, fan-in and fan-out limits, available local memory, and congestion estimates. Event penalties reduce traffic, while early-exit rules let individual samples finish before the maximum timestep. Noise injection is useful when the model will be mapped to mixed-signal circuits. Speck's attention-based dynamic-computing method is one example in which the amount of processing changes with the input [8]. Measured event counts are still needed, because a shorter nominal window does not always produce less communication.

4.2. Variability and peripheral overhead

Analog and resistive implementations introduce errors between training and deployment. Mixed-signal neuron parameters vary, and resistive memories exhibit drift, asymmetric updates, endurance limits, and disturbance of neighboring states. As arrays grow, wire resistance and unintended current paths alter the effective weights. Peripheral conversion and readout circuits can also materially affect area and energy [2, 15]. Two-dimensional materials introduce further manufacturing questions involving wafer-scale growth, layer thickness, defects, and CMOS integration [14]. Estimates of final accuracy should include these measured effects.

Several mitigation techniques are already used. Selectors reduce unintended currents; write-verify procedures and reference cells improve programming stability; differential encoding and redundancy tolerate faults at an area cost. Device-aware training can incorporate measured variation into the weight update. In a near-term system, digital circuits are likely to retain routing, control, and error handling, with analog or resistive arrays used for selected operations. The fatigue-STDP work provides an example of assigning different physical devices to two learning time scales [13].

4.3. Reporting and benchmark selection

At minimum, a comparison should state the dataset and accuracy, input encoding, timestep count, measured event rate, batch size, end-to-end latency, active and idle power, technology node, and on-chip memory. It should also say whether sensing, preprocessing, routing, and I/O fall inside the power measurement. For pJ/SOP, the counted operation needs an explicit definition. Because workloads, process nodes, activity levels, and measurement methods differ, these quantities should be interpreted within a clearly stated measurement boundary rather than used as stand-alone ranking criteria.

Benchmark choice influences the conclusions. Keyword spotting, event-camera gestures, tactile signals, anomaly detection, and obstacle avoidance contain temporal structure and often impose a response deadline. Dense static-image classification can be useful for checking accuracy and capacity, although it does not exercise idle behavior or sensor-level event sparsity in the same way. A useful benchmark set would include both types and report the event statistics for each mapped network.

DOI: 10.54254/2755-2721/2026.CH37367

5. Conclusion

This review has connected the operating principle of an SNN chip to the evidence reported for representative hardware. Encoding determines event density; neuron and synapse organization determines state and weight access; learning constrains the mapped model; and decoding determines when a decision is available. Digital platforms provide mature integration, scale, and programmability. Mixed-signal and sensor-integrated platforms reduce periodic or frame-based activity but require careful treatment of mismatch, calibration, scene activity, and measurement scope. Memristive implementations reduce weight movement and support in-memory accumulation, although device and array non-idealities remain important. These routes solve different parts of the edge-computing problem and do not support a single technology ranking. Future evaluations should report the encoding and decoder, timestep count, event activity, task accuracy, end-to-end latency, active and idle power, and the components included in the measurement. For mixed-signal and emerging devices, calibration effort, peripheral power, yield, and variation across chips should also be stated.

References

[1] Shi, W., Cao, J., Zhang, Q., et al. (2016). Edge computing: Vision and challenges. *IEEE Internet of Things Journal, 3*, 637–646.
[2] Roy, K., Jaiswal, A., & Panda, P. (2019). Towards spike-based machine intelligence with neuromorphic computing. *Nature, 575*, 607–617.
[3] Maass, W. (1997). Networks of spiking neurons: The third generation of neural network models. *Neural Networks, 10*, 1659–1671.
[4] Merolla, P. A., Arthur, J. V., Alvarez-Icaza, R., et al. (2014). A million spiking-neuron integrated circuit with a scalable communication network and interface. *Science, 345*, 668–673.
[5] Davies, M., Srinivasa, N., Lin, T. H., et al. (2018). Loihi: A neuromorphic manycore processor with on-chip learning. *IEEE Micro, 38*, 82–99.
[6] Pei, J., Deng, L., Song, S., et al. (2019). Towards artificial general intelligence with hybrid Tianjic chip architecture. *Nature, 572*, 106–111.
[7] Ma, D., Jin, X., Sun, S., et al. (2024). Darwin3: A large-scale neuromorphic chip with a novel ISA and on-chip learning. *National Science Review, 11*, nwae102.
[8] Yao, M., Richter, O., Zhao, G., et al. (2024). Spike-based dynamic computing with asynchronous sensing-computing neuromorphic chip. *Nature Communications, 15*, 4464.
[9] Moradi, S., Qiao, N., Stefanini, F., et al. (2018). A scalable multicore architecture with heterogeneous memory structures for dynamic neuromorphic asynchronous processors (DYNAPs). *IEEE Transactions on Biomedical Circuits and Systems, 12*, 106–122.
[10] Frenkel, C., Lefebvre, M., Legat, J. D., et al. (2019). A 0.086-mm² 12.7-pJ/SOP 64k-synapse 256-neuron online-learning digital spiking neuromorphic processor in 28-nm CMOS. *IEEE Transactions on Biomedical Circuits and Systems, 13*, 145–158.
[11] Furber, S. B., Galluppi, F., Temple, S., et al. (2014). The SpiNNaker project. *Proceedings of the IEEE, 102*, 652–665.
[12] Wang, Z., Joshi, S., Savel'ev, S., et al. (2018). Fully memristive neural networks for pattern classification with unsupervised learning. *Nature Electronics, 1*, 137–145.
[13] Dang, B., Zhang, T., Meng, F., et al. (2026). Spiking neural networks with fatigue spike-timing-dependent plasticity learning using hybrid memristor arrays. *Nature Electronics, 9*, 213–224.
[14] Park, S., Naqi, M., Lee, N., et al. (2024). Recent advancements in 2D material-based memristor technology toward neuromorphic computing. *Micromachines, 15*, 1451.
[15] Ielmini, D., & Wong, H.-S. P. (2018). In-memory computing with resistive switching devices. *Nature Electronics, 1*, 333–343.