A 65nm 5 TOPS/W Digital CIM Accelerator with Reconfigurable Precision and Temporal Pipelining for Spiking Neural Networks

Deepika Sharma, Shubham Negi, Trishit Dutta, Amogh Agrawal, Kaushik Roy
Department of Electrical and Computer Engineering, Purdue University, West Lafayette, USA
{sharm444, snegi, tdutta, agrawa64, kaushik}@purdue.edu

***Abstract*—Spiking Neural Networks (SNNs) have the inherent ability to process highly sparse event data from dynamic vision sensors (DVS). However, many existing SNN accelerators support only fixed network architectures, limited data precision, and fail to efficiently handle membrane potential dynamics and varying input sparsity. This work presents SpiDR, a reconfigurable digital compute-in-memory (CIM) SNN accelerator that addresses these limitations through: (1) fused weight-Vmem macros for reduced data movement, (2) staggered layout and reconfigurable peripherals for variable precision support, (3) zero-skipping to exploit unstructured input sparsity, and (4) asynchronous pipelining to maintain throughput despite variable computation times. Fabricated in TSMC's 65nm CMOS technology, SpiDR achieves up to 5 TOPS/W at 95% input sparsity and supports diverse event-based workloads including gesture recognition and optical flow estimation. SpiDR achieves $2\times$ to $1000\times$ higher area efficiency and up to $7.6\times$ higher energy efficiency compared to prior designs.**

***Index Terms*—spiking neural networks, compute-in-memory, event-based vision, DVS, optical flow**

I. Introduction

Dynamic Vision Sensors (DVS) have introduced a paradigm shift in machine vision from conventional frame-based to more efficient event-based sensing, which only transmits changes in pixel intensity [1]. This sensing paradigm results in lower data redundancy and higher temporal resolution, which benefit dynamic applications such as vision-based autonomous navigation. Spiking Neural Networks (SNNs), inspired by biological neural dynamics, offer a promising approach for processing the sparse, temporal events from DVS. However, general-purpose processors like CPUs and GPUs suffer from inefficiencies when executing SNNs due to their synchronous execution model, high data movement, and underutilization of processing units under sparse activity.

To address these limitations, several SNN accelerators have been proposed [2]–[6]. However, current SNN accelerators have several constraints, such as fixed network architectures, neuron models, or limited bit-precision support, restricting their widespread adoption.

This work was partially supported by IARPA MicroE4AI program, DARPA/SRC JUMP CoCoSys, the U.S DoE and Intel Corp.

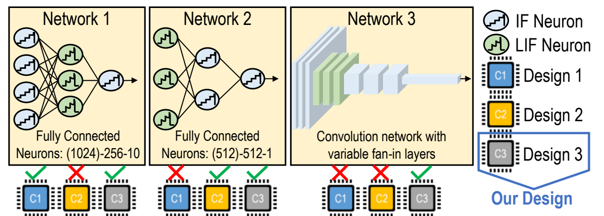


Fig. 1. Lack of reconfigurability in existing SNN accelerators.

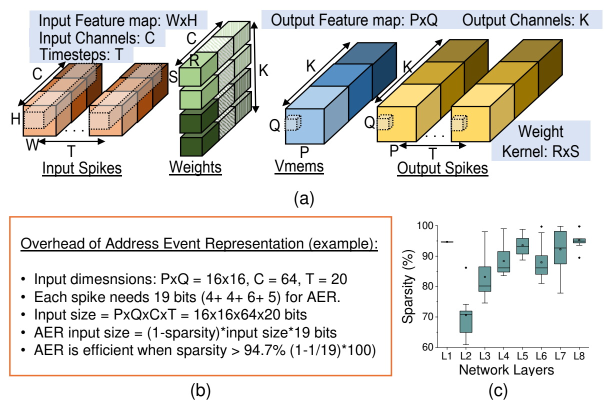


Fig. 2. (a) An example spiking convolution layer. (b) Overhead of using AER for a spiking convolution layer. (c) Layerwise input sparsity distribution for the optical flow estimation network used in our evaluation.

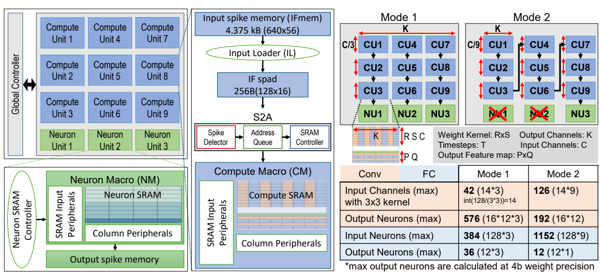
Moreover, a critical bottleneck in many existing designs is the inefficient handling of neuron membrane potentials (Vmems), which accumulate over multiple timesteps and require high-precision storage and update operations.

Additionally, many SNN accelerators rely on sparse optimizations, such as Address Event Representation (AER), which compress input spikes but become inefficient when input sparsity drops below a threshold. In practice, input sparsity varies widely across layers and workloads, making such optimizations unreliable. Fig. 2(a-b), highlight the AER overhead with an example spiking convolution layer, and Fig. 2(c) presents the layerwise input sparsity variation in our optical flow estimation network, illustrating the need for hardware that gracefully handles variable sparsity.

In response to these challenges, this work presents SpiDR, a reconfigurable SNN inference core with the following features:

- Digital CIM macros with fused weight/Vmem storage and support for three weight/Vmem precision modes (4/7, 6/11, and 8/15 bits).
- Zero-skipping to leverage unstructured input sparsity across a range of sparsity levels.
- Asynchronous handshaking to maintain an efficient computation pipeline under varying spike activity.
- Reconfigurable operating modes with timestep pipelining to reduce data movement for different layer sizes.

II. Proposed SNN Core Architecture



(a) Proposed SNN core architecture (b) Reconfigurable parallelism

Fig. 3. System architecture and reconfigurable parallelism.

Fig. 3(a) outlines the proposed SNN core with 9 compute units (CU) and 3 neuron units (NU). Compute units accumulate weights into Vmems based on spikes, featuring an input spike memory (IFmem), input loader (IL), input scratchpad buffer (IFspad), spike-to-address converter (S2A), and a CIM compute macro (CM). The IL transfers data from IFmem to IFspad, aligning for convolution (Conv) or fully connected (FC) layer operations. The S2A converts IFspad spikes into weight and Vmem row addresses for in-memory weight to Vmem accumulation in the CM. NUs receive partial Vmems from CUs and perform the required neuron operations (e.g., accumulation and threshold comparison for Integrate and Fire (IF) neuron) using the CIM neuron macro (NM) coordinated by a neuron SRAM controller.

A. Mapping SNN layers on SpiDR

Each macro computes partial Vmems for multiple output neurons concurrently. We use a weight stationary mapping strategy where K is mapped along the macro's column dimension and spatial input dimensions (RxSxC) over which a partial Vmem value is accumulated are mapped across the rows of a compute macro. Our design also eliminates off-chip movement of partial Vmems by mapping the entire input fan-in (RxSxC for Conv and the number of input neurons for FC layers) onto the chip. This is achieved by introducing reconfigurable operating modes and pipelining timesteps across compute units.

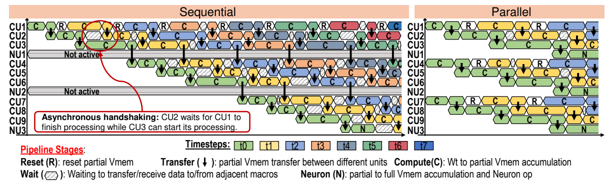


Fig. 4. Timestep pipelining with asynchronous handshaking. Each unit proceeds independently based on data readiness. For example, CU2 processes timestep t1, forwards partial Vmems to CU3, and waits for CU1 to complete t2. Meanwhile, CU3 can start processing t1.

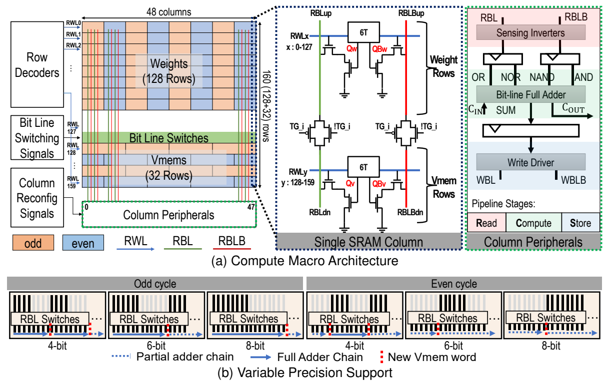


Fig. 5. (a) Compute macro architecture with fused weight and Vmem arrays. RBL switches connect bitlines for in-memory accumulation. (b) Column peripheral configurations for different weight/Vmem precisions (4/7, 6/11, 8/15). For example, in a 4-bit odd-cycle operation, weights are added to odd Vmem rows by enabling RBL switches at bitline groups (0–3), (8–11), ..., (40–43).

1) Reconfigurable operating modes: SpiDR supports two operating modes (Fig. 3(b)) based on input fan-in. Mode 1 handles smaller fan-ins ($\leq 128 \times 3$) using three parallel pipelines, each with three compute macros and one neuron macro, for high parallelism. Mode 2 supports larger fan-ins (up to 9×128) by using all nine compute macros and a single neuron macro, reducing parallelism but eliminating off-chip partial Vmem transfers.

2) Timestep Pipelining: SpiDR pipelines timesteps across compute and neuron units using an asynchronous handshaking mechanism (Fig. 4) to accommodate sparsity-dependent variation in compute latency. Each unit begins processing as soon as data is available, avoiding stalls that would be required in a synchronous pipeline. Input channels are evenly distributed across macros to minimize idle time.

B. Macro Design

Fig. 5(a) shows the compute macro, an enhanced version of [7], featuring RBL switches (transmission gates) to connect weight and Vmem bitlines. This eliminates the need for extra read wordlines used in [7] and enables support for multiple bit precisions (4/7, 6/11, 8/15) via programmable column peripherals. SpiDR's compute macro achieves a $1.8\times$ improvement in area efficiency over [7] and RBL switches account for only 0.9% of the macro's area.

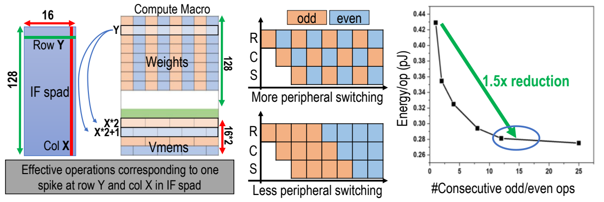
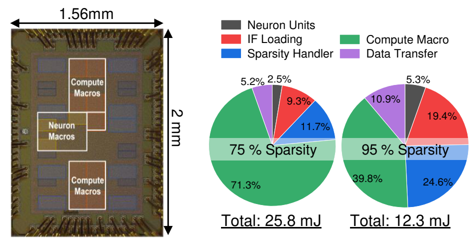


Fig. 6. Analyzing and reducing the peripheral switching overhead. Each column in the IFspad maps to two Vmem rows to support higher Vmem precision. Grouping consecutive even or odd operations reduces peripheral switching, lowering energy by 1.5×.



(a) Chip Photo (b) Energy Distribution

Fig. 7. Die photo and component-wise energy distribution.

TABLE I
CHIP SUMMARY

Technology	65 nm CMOS					
Core size	0.93mm x 1.07mm					
Die size	1.56mm x 2mm					
On chip SRAM	Total: 52.08 kB IMC Macros: 9.7 kB Excluding input spike memories*: 12.7 kB					
Supply Voltage	0.9V - 1.2V					
Frequency	50MHz - 150MHz					
Weight/Vmem Precision	4/7-bit, 6/11-bit, 8/15-bit					
	@50MHz, 0.9V			@150MHz, 1V		
Power Consumption (mW)	4.9			18		
Weight Precision	**4b**	**6b**	**8b**	**4b**	**6b**	**8b**
Energy Efficiency (TOPS/W) @95% sparsity	5	3.34	2.5	4.09	2.73	2.04
Throughput (GOPS) @95% sparsity	24.54	16.36	12.27	73.59	49.06	36.80

*Large input spike memories are not required if SpiDR is used as part of a larger system and data is streamed in continuously.

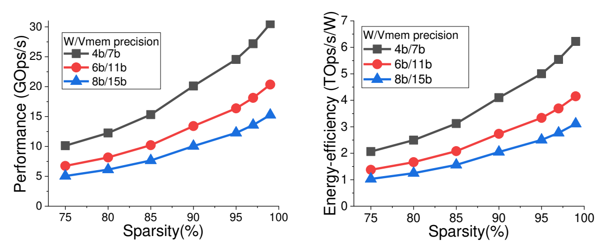


Fig. 8. Peak performance and energy efficiency trend with changing sparsity and bit precision. Both performance and energy efficiency increase with reducing precision and increasing sparsity as a result of increased parallelism and skipping ineffectual computations.

Adding one row of weights to Vmems consists of three pipeline stages: *Read*: one weight and one Vmem RWL are activated, generating the NOR and AND outputs at RBL and RBLB that are sensed and latched. (2) *Compute*: these outputs and C_{IN} from the adjacent column generate SUM and C_{OUT} bits. (3) *Store*: SUM is written to the appropriate Vmem row. The neuron macros share a similar design, with 72 rows (32 for partial Vmems, 32 for full Vmems, and the rest for the neuron threshold and leak values). Fig. 5(b) shows RBL switch configurations to support different precisions.

1) Reducing Peripheral Switching: In the 128×16 IFspad, each row (Y) corresponds to a weight row, and each column (X) corresponds to two Vmem rows (one even and one odd, to accommodate higher Vmem precision) in the compute macro (Fig. 6). Each spike at position (Y, X) triggers two operations: adding even-indexed weights from row Y to Vmems at row $X * 2$, and adding odd-indexed weights from row Y to Vmems at row $X * 2 + 1$. We segregate even and odd operations to minimize switching overhead, substantially reducing energy per operation. For example, switching peripherals after 15 consecutive even or odd operations, rather than after every cycle, results in a $1.5\times$ reduction in energy per operation. This insight guides the design of the spike-to-address converter (S2A).

C. Spike-to-address converter (S2A)

The S2A enables sparse, energy-efficient execution by converting binary input spikes into row addresses for in-memory accumulation. Each row of the IFspad, representing weight row Y, is scanned using a low-complexity spike detector (trailing zero detector) to determine the Vmem address (X). These (Y, X) tuples are stored in an address queue and used by the SRAM controller to issue in-memory accumulation commands to the compute macro. To reduce the even-odd switching frequency, we employ an even-odd ping-pong FIFO for the queue. After reading the address tuple from the even FIFO queue and performing the corresponding accumulation, it is written to the odd FIFO queue. The operating mode switches from even to odd once the odd FIFO is full or the even FIFO is empty. The depth of the even and odd FIFOs is set to 16, as no significant energy savings are observed for larger FIFOs (Fig. 6).

III. EVALUATION

Fig. 7(a) shows the die micrograph and Fig. 7(b) presents the component-wise energy distribution of the proposed SNN core at 75% and 95% input sparsity. Remarkably, even at high sparsity, CIM macros dominate energy consumption, indicating that the overhead of control logic is not overpowering actual computations. Furthermore, data movement is only a small fraction of the total energy consumption in SpiDR. Table I summarizes the chip specifications.

Fig. 8 presents the measured peak performance (GOPS) and energy efficiency (TOPS/W) of SpiDR across varying bit precisions and input sparsity levels. We evaluated SpiDR on two event-driven tasks: gesture

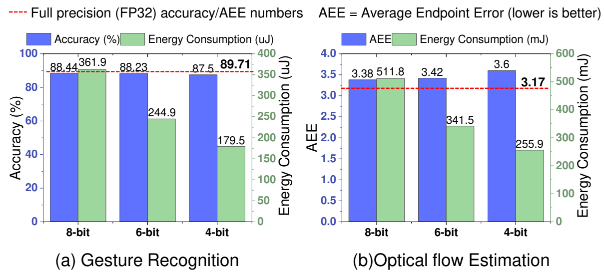


Fig. 9. Accuracy and energy consumption trade-off at different weight precisions for gesture recognition and optical flow estimation. Average endpoint error (AEE) is commonly used as a performance metric for optical flow estimation as opposed to accuracy.

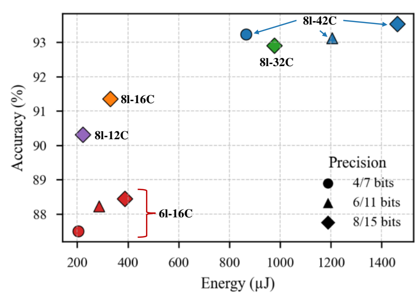


Fig. 10. Accuracy-energy trade-off for various network and precision configurations for gesture recognition at 150 MHz and 1 V. The marker color encodes the network configuration (layers and channels).

recognition using the IBM DVS gestures dataset [8] and optical flow estimation using the DSEC flow dataset [9]. Table II lists the network architectures, and Fig. 9 presents the corresponding energy-accuracy trade-offs at 50MHz, 0.9V. Since SpiDR is a digital CIM design, hardware implementation incurs no accuracy degradation. Additionally, Fig. 10 demonstrates SpiDR's flexibility in navigating the accuracy-energy space across a range of network depths, widths, and precision levels.

Table III compares SpiDR with recent digital SNN accelerators. While prior asynchronous designs with fixed network architectures [2]–[4] achieve low power on fixed tasks, their lack of reconfigurability limits their use. In contrast, SpiDR supports variable precisions, neuron models, and network topologies, enabling broader applicability. Compared using normalized energy (GSOPS/W) and area (SOPS/mm^2) efficiency at 65nm, 0.9V, and

TABLE II
NETWORK DETAILS

Application	Input size	Timesteps	Input layer	Intermediate layers	Output layer
Optical flow estimation	288x384	10	Conv (2,32)	6*Conv (32,32)	Conv (32,2)
Gesture recognition[a]	64x64	20	Conv (2,16)	4*Conv (16,16)	FC (64,11)

Conv (Input,Output Channels), FC (Input,Output Neurons)
[a]2x2 maxpool with stride 2 after every two intermed. conv layers.

TABLE III
COMPARISON WITH DIGITAL SNN ACCELERATORS

Parameter	SpiDR (This Work)	ANP-I ISSCC'23 [2]	ReckOn ISSCC'22 [3]	μBrain Frontiers'21 [4]	SD Training ISSCC'19 [5]	Sparse IMC ISSCC'24 [6]
Technology Supply (V) Freq. (MHz) Area[+] (mm^2)	65 nm CMOS 0.9 - 1.2 50 - 150 3.12 (0.99)	28nm CMOS 0.56 – 0.9 40 – 210 1.63 (1.27)	28nm FDSOI 0.5 – 0.8 13-115 0.86 (0.45)	40nm CMOS 1.1 - 2.68 (1.42)	65 nm CMOS 0.8 20 (10.08)	22 nm CMOS 0.55-0.9 51-280 2.83 (2.23)
Target Domain	SNN	SNN	Spiking RNN	SNN	SNN	SNN
Neuron Model	IF, LIF, RMP	hidden:LIF, out:IF	hidden:LIF, out:IF	IF[a]	IF	LIF, Residue[b]
Neurons	in: 1152[c] out: 576[c]	(1024)-512-10	(256)-256-16	(256)-64-16	2x200(hidden)	512[d]
Compute type	Digital CIM	Async. Digital	Async Digital	Async Digital	Digital	Digital CIM
Weight Prec. Vmem Prec.	4 / 6 / 8 7 / 11 / 15	hidden: 8, op: 10 -	8 16	4 7	- 8	4/8 -
Power Energy Eff. (Reported)	4.9 mW 2.5 - 5 TOPS/W @50MHz, 0.9V	2.91 mW 1.5 pJ/SOP @40MHz, 0.56V	80 μW 5.3 pJ/SOP @13MHz, 0.5V	73 μW 26 pJ/SOP MNIST @1.1V	23.6 mW 3.42 TOPS/W @20MHz 0.8V	524 μW 3.87 pJ/SOP @0.55V
Energy Eff. [GSOPS/W] (65nm, 0.9V, 4b)	5000[e] (250)[f]	222.37	50.17	35.36	5404[e] (270)[f]	33
Area Eff. [SOPS/mm^2] (65nm, 0.9V, 4b)	1.24×10^9	2.19×10^8	3.84×10^6	1.12×10^6	6.35×10^8	2.6×10^6
Tasks and Datasets	IBM DVS gest. classif. Optical flow estimation on DSEC-flow	MNIST classif. IBM DVS gesture classif. Keyword spotting (N-TIDIGIT, SeNic)	IBM DVS gesture classif. Keyword spotting (Spiking Heidelberg Digits) Navigation (delayed cue integration)	MNIST classif. Radar hand gest. classif. (customized dataset)	MNIST classif.	MNIST classif. IBM DVS gesture classif.
DVS gesture Accuracy	93.54%	92.0%	87.3%	-	-	94%
Flexible Network Arch.	Yes	No	No	No	No	Yes
Modified Training	No	Yes	Yes	No	Yes	No
Sparsity Support	Supports unstructured input sparsity	Yes (event-driven)	Yes (event-driven)	Yes (event-driven)	Yes (propagate only spikes)	Yes

[+]die area (core area) where provided, [a]LIF can be used with an external clock, [b]unclear terminology, [c]maximum input and output neurons in mode 2 and mode 1 respectively, [d]assuming 16-bit membrane potential (1KB/16-bit), [e]GOPS are reported, [f]assuming 95% sparsity or 5% spike rate

4b weights, SpiDR achieves the highest area efficiency (1.24×10^9 SOPS/mm^2) and very high energy efficiency, only slightly below [5]. However, [5] only reports results for a simple MNIST classification task while SpiDR is evaluated on more complex event-driven workloads. Overall, SpiDR achieves $2\times$ to $1000\times$ higher area efficiency and up to $7.6\times$ higher energy efficiency over existing digital SNN accelerators. Moreover, SpiDR does not require any modifications in the training methodology beyond quantization.

These results demonstrate that SpiDR offers a compelling balance of flexibility, performance, and efficiency for diverse event-driven inference workloads.

REFERENCES

[1] C. Brandli *et al.*, "A 240× 180 130 db 3 μs latency global shutter spatiotemporal vision sensor," *JSSC*, vol. 49, no. 10, 2014.

[2] J. Zhang *et al.*, "22.6 anp-i: A 28nm 1.5 pj/sop asynchronous spiking neural network processor enabling sub-o. 1 μj/sample on-chip learning for edge-ai applications," in *ISSCC*, 2023.

[3] C. Frenkel *et al.*, "Reckon: A 28nm sub-mm2 task-agnostic spiking recurrent neural network processor enabling on-chip learning over second-long timescales," in *ISSCC*, vol. 65, 2022.

[4] J. Stuijt *et al.*, "μbrain: An event-driven and fully synthesizable architecture for spiking neural networks," *Frontiers in neuroscience*, vol. 15, 2021.

[5] J. Park *et al.*, "7.6 a 65nm 236.5 nj/classification neuromorphic processor with 7.5% energy overhead on-chip learning using direct spike-only feedback," in *ISSCC*, 2019.

[6] Y. Liu *et al.*, "30.2 a 22nm 0.26 nw/synapse spike-driven spiking neural network processing unit using time-step-first dataflow and sparsity-adaptive in-memory computing," in *ISSCC*, 2024.

[7] A. Agrawal *et al.*, "Impulse: A 65-nm digital compute-in-memory macro with fused weights and membrane potential for spike-based sequential learning tasks," *SSCL*, vol. 4, 2021.

[8] A. Amir *et al.*, "A low power, fully event-based gesture recognition system," in *CVPR*, 2017.

[9] M. Gehrig *et al.*, "Dsec: A stereo event camera dataset for driving scenarios," *RAL*, vol. 6, no. 3, 2021.