

# THOR - A Neuromorphic Processor with 7.29G TSOP<sup>2</sup>/mm<sup>2</sup>Js Energy-Throughput Efficiency

Mayank Senapati<sup>1</sup>, Manil Dev Gomony<sup>1</sup>, Sherif Eissa<sup>1</sup>, Charlotte Frenkel<sup>2</sup>, and Henk Corporaal<sup>1</sup>

<sup>1</sup>Eindhoven University of Technology

<sup>2</sup>Delft University of Technology

**Abstract**—Neuromorphic computing using biologically inspired Spiking Neural Networks (SNNs) is a promising solution to meet Energy-Throughput (ET) efficiency needed for edge computing devices. Neuromorphic hardware architectures that emulate SNNs in analog/mixed-signal domains have been proposed to achieve order-of-magnitude higher energy efficiency than all-digital architectures, however at the expense of limited scalability, susceptibility to noise, complex verification, and poor flexibility. On the other hand, state-of-the-art digital neuromorphic architectures focus either on achieving high energy efficiency (Joules/synaptic operation (SOP)) or throughput efficiency (SOPs/second/area), resulting in poor ET efficiency. In this work, we present THOR, an all-digital neuromorphic processor with a novel memory hierarchy and neuron update architecture that addresses both energy consumption and throughput bottlenecks. We implemented THOR in 28nm FDSOI CMOS technology and our post-layout results demonstrate an ET efficiency of 7.29G TSOP<sup>2</sup>/mm<sup>2</sup>Js at 0.9V, 400 MHz, which represents a 3X improvement over state-of-the-art digital neuromorphic processors.

## I. INTRODUCTION

Neuromorphic computing using biologically inspired Spiking Neural Networks (SNN) has arisen as a new paradigm that can accommodate energy and throughput requirements of edge AI processing [1]. Neuromorphic hardware aims to emulate human brain operations and offers various advantages over traditional systems, including sparse low-power computation and highly scalable parallel processing [2]. Energy efficiency (Joules/synaptic operation (SOP)) and throughput efficiency (SOPs/second/area) are the two key metrics to evaluate a Neuromorphic architecture for edge AI applications. We combine these two metrics into one single figure of merit called *Energy-Throughput (ET)* efficiency in terms of GSOP<sup>2</sup>/mm<sup>2</sup>Js to efficiently capture the trade-off between them and to fairly compare the different Neuromorphic architectures. Digital neuromorphic architectures have made considerable progress in recent years, however, little focus has been given to optimizing ET efficiency.

Based on our analysis of energy consumption, silicon area usage and throughput of the state-of-the-art all-digital neuromorphic processors [3]–[7], we identify the following challenges that needs to be tackled for achieving the highest ET efficiency: (1) The synapse memory, which holds the individual states and parameters of the synapses is typically very large and contributes significantly to the overall energy consumption (and area usage) as can be seen in Figure 1, which shows the energy consumption breakdown in different components of the processor.

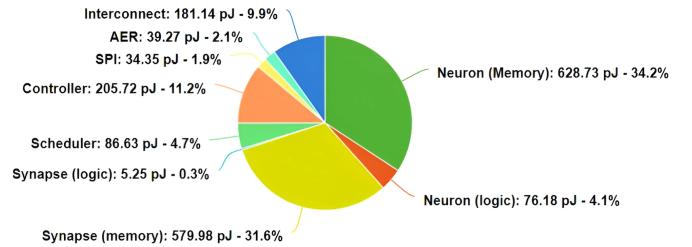


Figure 1: Energy breakdown of a state-of-the-art neuromorphic processor [3] running MNIST for a single neuron event (using technology library 28nm FDSOI at 0.9V@100MHz). Different components of the architecture are explained in section III.

SRAMs are typically used as on-chip memory in most of the all-digital neuromorphic architectures. However, SRAMs comes with different configurations in terms of number of banks, IO or word width, depth, internal multiplexing factor etc, requiring an extensive design space exploration. In addition, Standard Cell Memories (SCMs) [8] are becoming increasingly popular as a substitute to relatively smaller sized SRAMs due to high energy efficiency despite the poor area efficiency. Optimizing the energy consumption and area usage of synapse memory requires an extensive analysis of memory hierarchy for the synapse memory including different memory architectures and types. (2) The neuron and synapse memories are idle between successive accesses, which contributes to a significant amount of idle energy consumption and wastage of expensive on-chip memory bandwidth. This requires a novel architecture with efficient time multiplexing and pipelining of operations. (3) The scheduler is designed with fixed number of neurons to be processed in parallel, which is a limiting factor for scaling up the architecture for increased throughput. To address these limitations in the state-of-the-art neuromorphic processing architectures and to achieve the highest ET efficiency, this paper contributes the following:

- 1) A neuromorphic processor THOR with a novel architecture for neuron update including a parallel neuron update scheme in the neuron event, a multi-threaded scheduler for an increased throughput, and a detailed analysis on the impact of parallelism on the energy consumption. (section IV)
- 2) A detailed analysis of the memory hierarchy using multiple memory types and configurations. Based on our analysis we present the memory selection for THOR

- with configuration options of the different parameters (number of banks, IO or word width, depth, internal multiplexing factor etc). (section V)
- 3) We perform post-layout implementation of THOR in 28nm FDSOI CMOS technology and show a high ET efficiency of 7.29 GSOP<sup>2</sup>/mm<sup>2</sup>J<sub>s</sub> at 0.9V, 400 MHz). (section VI)

We review state-of-the-art architectures in section II and relevant background information in section III. In section IV, we present THOR's architecture followed by an energy exploration of different design choices and memory hierarchy in section V where we make our design choices. Finally, we present our implementation results in section VI and make a comparison with state-of-the-art architectures and conclude our paper with section VII.

## II. RELATED WORK

A variety of all-digital neuromorphic compute architectures have been proposed in the past. Truenorth [9], Loihi [10] and Spinnaker [11] represent very large scale neuromorphic architectures with multiple cores aimed at flexibility or programmability. Several multi-core architectures [4]–[7], achieve low energy consumption. For example, [7] relies on asynchronous circuits [12] to wake memories and logic while [4] uses routing circuits that rely on spike-driven communication to keep energy consumption to a minimum. FPGA based neuromorphic architectures [13]–[15], have been proposed as well to allow re-configurability. These architectures prioritize flexibility and programmability over energy efficiency. Architectures designed for low power embedded applications [3]–[5], [7], [16], [17] achieve low energy and area usage, but run with a slow clock, achieving low throughput. For example, in [17] event driven processing with asynchronous components acts as a bottleneck. While [18] implements a time multiplexed neuron ALU with event driven clock and power gating to achieve high energy efficiency on an always-on architecture. [5] proposes an always-on architecture with event driven clock gating and Globally Asynchronous Locally Synchronous (GALS) architecture which achieves 2.1pJ/SOP at 0.5V .ODIN [3] uses high density SRAM memory along with a time multiplexed neuron ALU but suffers from poor ET efficiency. To summarize, state-of-the-art neuromorphic architectures with high flexibility have low energy and area efficiency while energy efficient architectures suffer from low throughput and/or area efficiency. In this paper, we aim to maximize ET efficiency of digital neuromorphic architecture by performing an extensive analysis of energy, area and throughput bottlenecks.

## III. BACKGROUND

This section first introduces the Spiking Neural Networks (SNN) that is the class of artificial neural network supported by THOR and then the baseline hardware architecture of a digital neuromorphic processor.

### A. Spiking Neural Networks

Spiking Neural Networks (SNN) are a class of deep learning models which attempt to mimic biological nervous systems.

SNNs provide several advantages over traditional Artificial Neural Networks (ANN). The event driven nature of these networks encourages sparse computation which contributes to low power consumption. Since these networks are based on biological models, they are suitable candidates for biologically inspired online learning. The Leaky Integrate-and-fire (LIF) neuron, as shown in Figure 2, is a commonly used neuron model. When a spike arrives on a synapse, it triggers an input current into the post-synaptic neuron, which is integrated as voltage called a Synaptic Operation (SOP). The voltage of the neuron leaks according to a time constant. If the neuron reaches a certain threshold voltage, it fires an output spike and resets its voltage to a resting state.

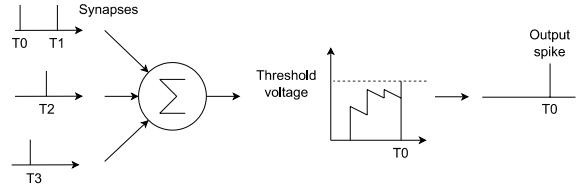


Figure 2: Leaky Integrate and fire (LIF) neuron model. Each spike arrives at a particular synapse with a corresponding synaptic weight, which increases the membrane potential. When the membrane potential reaches a certain level, it spikes and then returns to initial value.

### B. ODIN Baseline Architecture

We selected the state-of-the-art digital architecture ODIN presented in [3] as our baseline architecture, that consists of an LIF neuron core, synapse core, scheduler and peripherals as shown in Figure 3. We reduce the original baseline by removing support for Izhikevich neurons [19]. On-chip memories are used to store the individual states of neurons and synapses in the neuron and synapse cores, respectively. A Scheduler manages the neuron and synaptic updates. Each of the 256 neurons has a fan-in of 256 online-learning synapses, to emulate a fully connected 256x256 crossbar. An Address-event representation (AER) interface handles input and output events off-chip [20]. ODIN implements online learning with Spike-driven synaptic plasticity (SDSP) [21] and two operations: *Synapse event*: Triggers one specific synaptic operation (SOP) and *Neuron event*: updates all 256 neurons with 256 SOPs, according to a source neuron id, by time-multiplexing, as shown in Figure 4. Each event takes 2 cycles per SOP and influences online learning. Each word in the synapse memory contains 8 synapses. Hence, synapse memory access takes place every 8 SOPs.

## IV. THOR ARCHITECTURE

In this section, we first present the top-level THOR architecture and then the three main building blocks: neuron core, synapse core and multi-threaded spike scheduler with novel improvements.

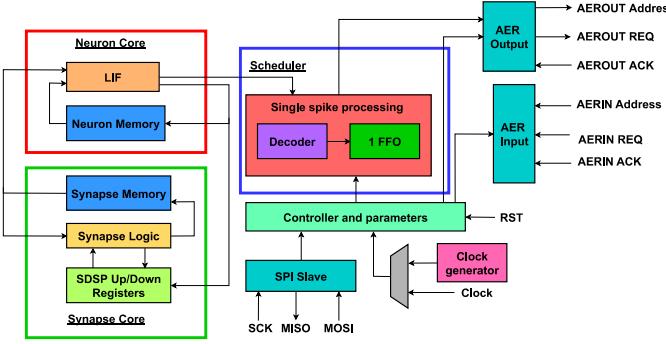


Figure 3: Baseline ODIN Architecture after removing the logic for supporting Izhikevich neuron model consists of neuron core, synapse core and spike scheduler.

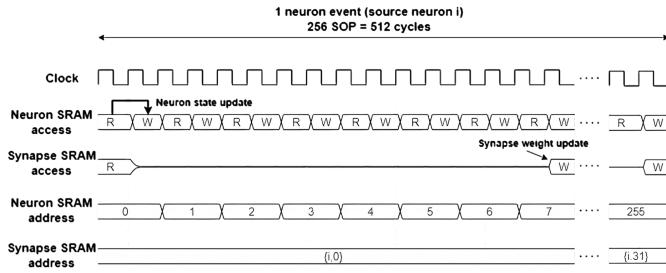


Figure 4: Neuron event in ODIN showing the access patterns of neuron and synapse SRAMs [3]

#### A. Top level architecture

Figure 5 shows the top level architecture of THOR. The main components are the neuron core, synapse core, and multi-threaded scheduler that consists of dedicated schedulers for output and input spikes. THOR implements an all-to-all  $N$  neuron network structure similar to the baseline ODIN architecture with the same SDSP online learning rules. Neuron and synapse memories are accessed and configured externally through the SPI interface. Input events are handled by an AER input block which is part of the controller. The input and output spike schedulers handle propagation of spikes on-chip and off-chip respectively.

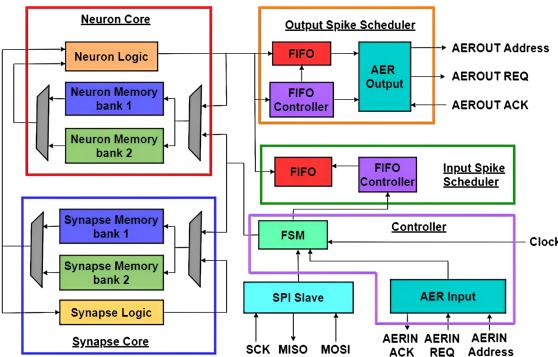


Figure 5: THOR - Top level Architecture. Detailed Scheduler architecture in Figure 7

As the neuron event is a fundamental operation in SNNs, we propose an architecture to achieve a *high-throughput neuron event* for higher area and throughput efficiency. To achieve the high-throughput neuron event, THOR contains two banks of neuron and synapse memory where each bank supports  $P$ -wide memory reads and writes. It also contains  $P$  parallel neuron and synapse logic units. Read and write operations are interleaved between the two banks to achieve high utilization of logic units. Figure 6 shows the timing diagram of our modified neuron event where  $P$  SOPs and memory updates are executed in parallel in a pipeline. While each  $P$  SOPs take two clock cycles, we leverage our two-bank memory architecture to achieve  $P$  operation every cycle with interleaved memory access. During our modified *neuron event*, all memories and logic units are fully utilized, resulting in high throughput and low leakage.

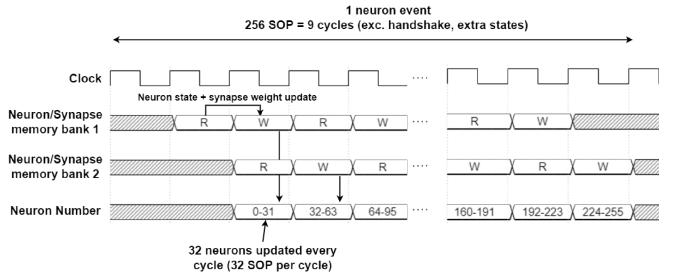


Figure 6: Neuron event timing diagram of THOR processing 256 SOPs in 9 clock cycles. We have selected the configuration of 32 neurons updates in parallel based on our analysis presented in section V.

#### B. Neuron Core

Our neuron core consists of two interleaved neuron memory banks and  $P$  LIF neuron logic units. The state of an LIF neuron is stored in 7 bytes as shown in Table I. The LIF neuron logic consists of (1) a state update block for integration and firing. (2) a calcium update block to implement SDSP learning. Each memory bank is implemented using 7 sub-banks, one for each byte of neuron state (Table I). Each sub-bank has word size of  $P$  bytes and consists of  $N/2P$  entries. The leakage and threshold memory write circuits are gated during inference, as they are only configured during initialization. Furthermore, the calcium information banks can be gated and disabled if online learning is not being used.

Byte	Information	Status - Online learning disabled	Status - Online learning enabled
0	Membrane Potential	Read/Write	Read/Write
1	Leakage	Read-only	Read-only
2	Threshold	Read-only	Read-only
3-6	Calcium information	Unused	Byte 4 - Read/Write Byte 3,5,6 - Read-only

Table I: Neuron state breakdown (more details in [3]); 7 bytes/neuron.

#### C. Synapse Core

The synapse core consists of logic units and memory banks. The logic blocks are responsible for updating triggered synapses. Updates are a function of the post-synaptic

membrane potential and calcium information. We implement P parallel neuron processing units in the neuron core for parallel processing.

#### D. Multi-threaded Scheduler

We implement a multi-threaded scheduler consisting of two parallel independent hardware schedulers to handle internal and outgoing spikes. The architecture of both schedulers is identical and shown in Figure 7 (a). They consist of a FIFO, and a controller which executes an FSM independent of the main controller. The schedulers receive a P-bit spike vector from the neuron core each cycle during a neuron event, denoting which neurons have spiked, and an offset ( $\log_2 N$  bits) to indicate neuron starting address. Whenever a spike is detected in the input, the spike vector and offset are pushed into the FIFO.

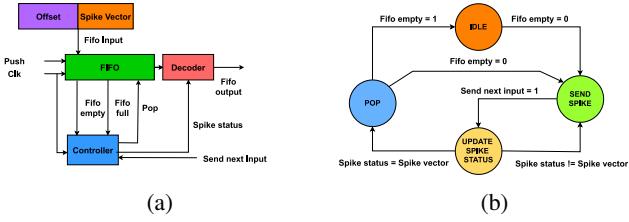


Figure 7: (a) Multi-threaded scheduler block diagram. The FIFO output and "Send next input" signals are connected to the AER output interface and to the controller for the output and input schedulers respectively.(b) Multi-threaded scheduler finite state machine (FSM).

The FSM used by the controller is shown in Figure 7 (b). While the FIFO is not empty, the scheduler decodes an output spike from the first spike vector in the queue. Output spikes are only triggered by a signal "Send next input" from the controller or the AER output, for input and output schedulers respectively, to control traffic. A dedicated status register tracks when a spike vector has been exhausted to pop it from the queue. The FIFO has a depth of  $N/P$  entries, which is sufficient to handle all scenarios. Furthermore, having parallel schedulers operating independent results in higher throughput as it prevents pipeline stalls.

#### V. MEMORY HIERARCHY OPTIMIZATION

We consider two memory types: Static Random Access Memory (SRAM) and Standard Cell Memory (SCM). SRAMs are a common choice for on chip memories and are primarily defined in terms of word size, number of words and mux factor. SCMs consist of arrays of latches or flip-flops and a readout circuit which can be built out of multiplexers, gates or tristate buffers [8]. In this section, we explore the efficiency of different memory hierarchies with different parallelism schemes. We first compare SRAMs and SCMs for different bank sizes and then for different frequencies and degrees of parallelism. We conclude the section with our design choices based on analysis results.

#### A. SCM vs SRAMs

Prior studies have shown that larger SRAMs are area efficient than SCMs, however, the latter offers better area efficiency for smaller sizes. Moreover, the energy consumption of SRAM and SCMs have not yet compared in detail for different bank sizes, multiplexing factor and I/O width. We compare the energy efficiency of SCMs and SRAMs for 32-bit and 64-bit wide word sizes and different bank sizes as shown in Figure 8. Our analysis shows that SRAM favors larger memory sizes while SCMs do not scale well with memory size due to the increasing overhead logic which increases leakage power. SCM performance improves for larger word sizes for a fixed memory size as having less entries in SCM memory reduces the complexity of the decoder and multiplexer logic. The results from Figure 8 show that for specific memory and word size combinations, SCMs can have better energy efficiency than SRAMs. Furthermore, the use of low leakage libraries and optimal bank sizes improved the energy efficiency of SCMs. However, SCMs have one order-of-magnitude lower energy efficiency compared to SRAMs for large memory sizes.

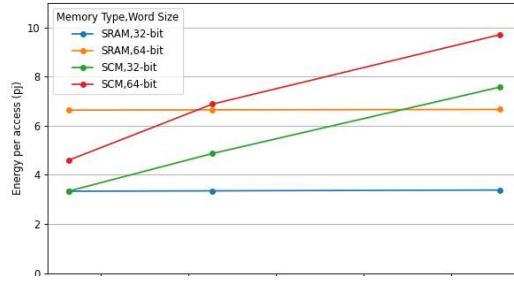


Figure 8: Energy analysis of SCM and SRAM for different word sizes and different memory sizes (0.9V, 100MHz) in 28nm FDSOI technology. SCM benefits from smaller memory sizes because the periphery circuitry is less complex and power hungry than that of an SRAM.

#### B. Synapse memory hierarchy

To explore different memory hierarchies with different bank sizes and degree of parallelism, we define our synapse memory structure in a generalized manner. Let  $N$  be the number of neurons,  $S$  be the memory bank size in bits, and  $P$  be the degree of neuron update parallelism in our design. The synapse memory consists of  $N^2$  synapses arranged in a crossbar architecture. The total size of the synapse memory is  $4N^2$  bits and the I/O word width is  $4P$  bits, as we store synapses in 4 bits (weights). Figure 9 shows the general architecture of the synapse memory. For a given bank size ( $S$ ), both memory types consist of  $4N^2/S$  bank rows. However, as our analysis was limited by 32-bit word size SRAM macros, we have to partition each SRAM row into 32-bit wide banks ( $4P/32$  banks), while for SCMs we only have 1 bank per row as we could adjust the SCM word size freely. Defining the memory hierarchy in a generalized manner allow us to create a parameterized synapse memory with SRAM macros of a specific size. The external address decoders and readout circuit

are synthesized with standard cells to estimate the energy and area overhead for multiplexing. We compare SCMs and SRAMs for our energy analysis in subsection V-A.

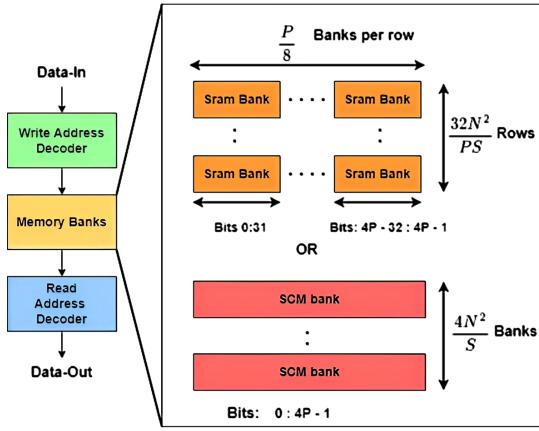


Figure 9: Synapse Memory Architecture.  $N$  = Number of neurons,  $P$  = Degree of parallelism,  $S$  = Bank size in bits

### C. Parallelism exploration

In this section, we analyze the impact of memory choices with respect to different parallel designs for THOR. We implement designs with different amount of parallel neurons and different operating frequencies. The designs are implemented in 28nm technology with 0.9V target voltage, and synthesized with Cadence Genus.

The power numbers are drawn from post-synthesis reports and include the overhead of AER handshakes and controller state change. The *Energy per synaptic operation*  $E_{sop}$  is considered as a metric of energy efficiency. We consider the scenario where the chip is saturated with neuron events. We calculate the Energy per neuron event using equation  $E_{SOP} = (N_{cycles} \times T_{cycle} \times P_{avg})/N$ , where  $N_{cycles}$  is the number of cycles for a single neuron event,  $N$  is the number of neurons, and  $P_{avg}$  is the average power. Figure 10 shows the  $E_{SOP}$  of different designs with different degrees of parallelism, different synapse memory technologies (synapse memory size corresponding to 64K synapses), and operating at different frequencies. At lower frequencies, SCM memories suffer from high leakage power. This can be improved using low leakage libraries. Based on the results from Figure 10, we chose SCM memories with 32 parallel neuron updates for our implementation as it provides the highest energy efficiency. Note that although the energy efficiency of SRAM is very close to the SCM and yet provides a better area efficiency, we selected SCM in the final implementation to support Voltage-Frequency Scaling (which will be limited by SRAMs) in our future work.

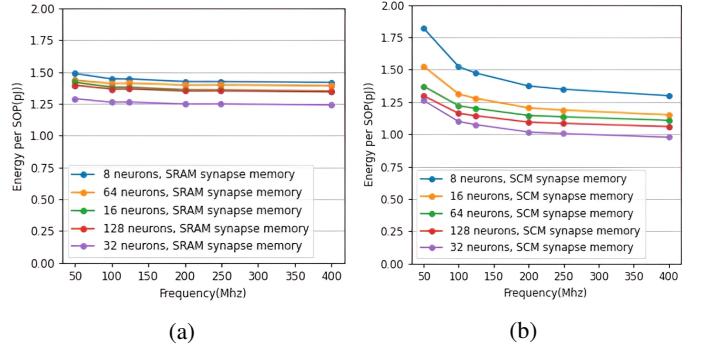


Figure 10:  $E_{sop}$  for THOR with SRAM (a) and SCM (b) (0.9V) shows that SCM based synapse memories suffer from high leakage power due to the large area of synapse memory.

## VI. RESULTS AND COMPARISON

We implemented THOR with 32 physical neurons in 28nm FDSOI technology (Figure 11) at 400 MHz and 0.9V, with 4KB and 64KB for the neuron and synapse memories, respectively. The post-layout netlist energy breakdown is reported in Figure 12. All blocks have input and clock gating to reduce idle dynamic energy.

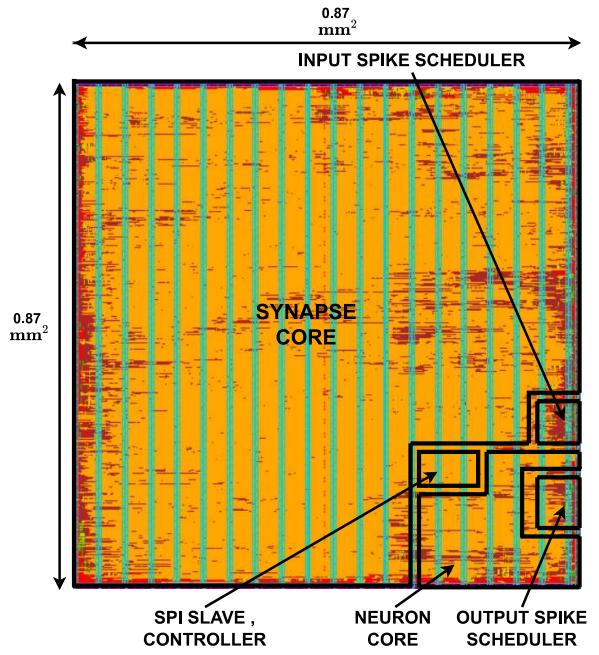


Figure 11: THOR post-synthesis layout. Chip area ( $870\mu m \times 870\mu m$ ) is mostly occupied by the SCM synapse memories.

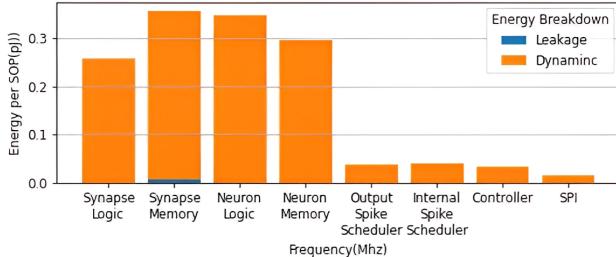


Figure 12: Post-layout Energy Breakdown of a neuron event (0.9V, 400MHz).

As Figure 12 shows, the memories make significant contributions to THOR’s energy consumption. We use multiple threshold voltage libraries to reduce leakage consumption. For a P-degree of parallelism, the multiplexing circuit of neuron memory must handle 7P bytes of neuron information. One can improve the energy efficiency further with the use of low leakage libraries and voltage-frequency scaling, especially for the synapse memories.

A summary of THOR’s performance compared to state-of-the-art all-digital designs is shown in Table II.  $\mu$ Brain outperforms others in  $E_{sop}$  and Energy-Area efficiency, as it uses asynchronous design techniques and operates at extreme low clock frequency (in the Hz range) to reduce the dynamic power consumption. However, THOR has comparable  $E_{sop}$  even with the use of synchronous logic. Moreover, THOR does uses SCM instead of SRAM to allow voltage-frequency scaling for improved energy performance. The massive parallel multi-core architectures of [6] and [4] achieve high throughput, however, they have low energy efficiency. By combining all the metrics Energy consumption per SOP, Area and Throughput together, THOR outperforms state-of-the-art all-digital neuromorphic architectures by at least 3X.

## VII. CONCLUSIONS

We presented THOR, an all-digital neuromorphic processor with novel architecture for neuron update including a parallel neuron update scheme in the neuron event and a multi-threaded scheduler that solves the energy and throughput bottlenecks in state-of-the-art processors. We performed an energy analysis of different memory types and configurations and devised the optimal memory hierarchy for neuron and synapse memory. We implemented THOR in 28nm FDSOI CMOS technology and demonstrated a single core THOR with an area of  $0.77 \text{ mm}^2$  and an Energy-Area-Throughput efficiency of  $7.37 \text{ GSOP}^2/\text{mm}^2\text{Js}$  at 0.9V and 400 MHz, with a 3X improvement of ET efficiency compared to state-of-the-art digital neuromorphic processors.

## REFERENCES

- [1] S. Narayanan *et al.*, “Spinalflow: An architecture and dataflow tailored for spiking neural networks,” in *2020 ACM/IEEE 47th Annual International Symposium on Computer Architecture (ISCA)*, 2020, pp. 349–362.
- [2] M. Pfeiffer and T. Pfeil, “Deep learning with spiking neurons: Opportunities and challenges,” *Frontiers in Neuroscience*, vol. 12, 2018.
- [3] C. Frenkel *et al.*, “A 0.086-mm, 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*, vol. 13, no. 1, pp. 145–158, 2018.
- [4] Y. Kuang *et al.*, “A 64k-neuron 64m-1b-synapse 2.64 pj/sop neuromorphic chip with all memory on chip for spike-based models in 65nm cmos,” *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 68, no. 7, pp. 2655–2659, 2021.
- [5] M. Wong *et al.*, “A 2.1 pj/sop 40nm snn accelerator featuring on-chip transfer learning using delta stdp,” in *ESSDERC 2021-IEEE 51st European Solid-State Device Research Conference*, 2021, pp. 95–98.
- [6] G. K. Chen *et al.*, “A 4096-neuron 1m-synapse 3.8-pj/sop spiking neural network with on-chip stdp learning and sparse weights in 10-nm finfet cmos,” *IEEE JSSC*, vol. 54, no. 4, pp. 992–1002, 2018.
- [7] J. Zhang *et al.*, “A 28nm configurable asynchronous snn accelerator with energy-efficient learning,” in *2021 27th IEEE International Symposium on Asynchronous Circuits and Systems*. IEEE, 2021, pp. 34–39.
- [8] A. Teman *et al.*, “Power, area, and performance optimization of standard cell memory arrays through controlled placement,” *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, vol. 21, no. 4, pp. 1–25, 2016.
- [9] F. Akopyan *et al.*, “Truenorth: Design and tool flow of a 65 mw 1 million neuron programmable neurosynaptic chip,” *IEEE transactions on computer-aided design of integrated circuits and systems*, vol. 34, no. 10, pp. 1537–1557, 2015.
- [10] M. Davies *et al.*, “Loihi: A neuromorphic manycore processor with on-chip learning,” *Ieee Micro*, vol. 38, no. 1, pp. 82–99, 2018.
- [11] E. Painkras *et al.*, “Spinnaker: A multi-core system-on-chip for massively-parallel neural net simulation,” in *Proceedings of the IEEE 2012 Custom Integrated Circuits Conference*. IEEE, 2012, pp. 1–4.
- [12] A. Peeters *et al.*, “Click elements: An implementation style for data-driven compilation,” in *2010 IEEE Symposium on Asynchronous Circuits and Systems*. IEEE, 2010, pp. 3–14.
- [13] X. Huang *et al.*, “Spiking neural network based low-power radioisotope identification using fpga,” in *2020 27th IEEE International Conference on Electronics, Circuits and Systems (ICECS)*. IEEE, 2020, pp. 1–4.
- [14] J. P. Mitchell *et al.*, “A small, low cost event-driven architecture for spiking neural networks on fpgas,” in *International Conference on Neuromorphic Systems 2020*, 2020, pp. 1–4.
- [15] H. Irmak *et al.*, “A dynamic reconfigurable architecture for hybrid spiking and convolutional fpga-based neural network designs,” *Journal of Low Power Electronics and Applications*, vol. 11, no. 3, p. 32, 2021.
- [16] H. Lee *et al.*, “Neuroengine: a hardware-based event-driven simulation system for advanced brain-inspired computing,” in *Proceedings of the 26th ACM International Conference on Architectural Support for Programming Languages and Operating Systems*, 2021, pp. 975–989.
- [17] J. Stuijt *et al.*, “μbrain: An event-driven and fully synthesizable architecture for spiking neural networks,” *Frontiers in neuroscience*, vol. 15, p. 538, 2021.
- [18] D. Wang *et al.*, “Always-on, sub-300-nw, event-driven spiking neural network based on spike-driven clock-generation and clock-and-power-gating for an ultra-low-power intelligent device,” in *2020 IEEE Asian Solid-State Circuits Conference (A-SSCC)*. IEEE, 2020, pp. 1–4.
- [19] C. Frenkel, J.-D. Legat, and D. Bol, “A compact phenomenological digital neuron implementing the 20 izhikevich behaviors,” in *2017 IEEE Biomedical Circuits and Systems Conference (BioCAS)*, 2017, pp. 1–4.
- [20] K. Boahen, “Point-to-point connectivity between neuromorphic chips using address events,” *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 47, no. 5, pp. 416–434, 2000.
- [21] J. M. Brader *et al.*, “Learning real-world stimuli in a neural network with spike-driven synaptic dynamics,” *Neural computation*, vol. 19, no. 11, pp. 2881–2912, 2007.

	<b>mu-brain [17]</b>	<b>Wang [18]</b>	<b>Kuang [4]</b>	<b>Wong [5]</b>	<b>Chen [6]</b>	<b>Zhang [7]</b>	<b>ODIN [3]</b>	<b>THOR</b>
<b>Circuit type</b>	Asynchronous	Synchronous	Synchronous	Synchronous	Synchronous	Asynchronous	Synchronous	Synchronous
<b>Technology (nm)</b>	40	65	65	40	10	28	28	28
<b>Total area (mm<sup>2</sup>)</b>	1.42	1.99	89.48	14.57	1.28	0.52	0.086	0.77
<b>Number of cores</b>	1	1	64	44	64	1	1	1
<b>Neurons</b>	336	650	64K	11K	64K	256	256	256
<b>Synapses</b>	37K	67K	64M-total	2.8M	64M	131K	65K	65K
<b>Online learning</b>	No	No	No	Yes	Yes	Yes	Yes	Yes
<b>Energy per SOP (pJ/SOP)</b>	0.627 @ 1.1V	1.5 @ 70Khz, 0.5V	2.64 @ 24MHz, 0.89V 4.60 @ 192MHz, 1.20V	2.1 @ 12.5MHz, 0.5V 9.5 @ 160MHz, 1V	3.8 @ 105MHz, 0.52 V 8.3 @ 506MHz, 0.9V	3.97 @ 6.7MHz, 0.8V	8.40 @ 75MHz, 0.55V	1.40 @ 400MHz, 0.9V
<b>Throughput (SOP/s)</b>	16 K @ 1.4Hz input rate	260 K @ 0.5V, 70Khz	12.29G @ 192MHz, 1,20V	0.28G @ 12.5MHz, 0.5V 1.93G @ 160MHz, 1V	5.16G @ 105MHz, 0.52V 25.11G @ 506MHz, 0.9V	0.01G @ 6.7MHz, 0.8V	37.5M @ 75MHz, 0.55V	7.84G @ 400MHz, 0.9V
<b>Energy-Throughput (ET) (TSOP<sup>2</sup>/mm<sup>2</sup>J s)</b>	17.92K @ 1.4hz input rate	88.4K @ 0.5V, 70Khz	0.049G @ 192MHz, 1,20V	0.009G @ 12.5MHz, 0.5V 0.014G @ 160MHz, 1V	1.03G @ 105MHz, 0.52V 2.25G @ 506MHz, 0.9V	0.005G @ 6.7MHz, 0.8V	51.9M @ 75MHz, 0.55V	<b>7.29G</b> @ 400MHz, 0.9V

Table II: Comparison of THOR with state-of-the-art neuromorphic processors. THOR outperforms the state-of-the-art architectures by a factor of over 3X in terms of ET efficiency.