Energy-efficient SNN Architecture using 3nm FinFET Multiport SRAM-based CIM with Online Learning

Huijbregts, Lucas; Liu, Hsiao Hsuan; Detterer, Paul; Hamdioui, Said; Yousefzadeh, Amirreza; Bishnoi, Rajendra

DOI
10.1145/3649329.3656514

Publication date
2024

Document Version
Final published version

Published in
Proceedings of the 61st ACM/IEEE Design Automation Conference, DAC 2024

ESAM: Energy-efficient SNN Architecture using 3nm FinFET Multiport SRAM-based CIM with Online Learning

Lucas Huijbregts[1,2], Liu Hsiao-Hsuan[2], Paul Detterer[2], Said Hamdioui[1], Amirreza Yousefzadeh[3,2], Rajendra Bishnoi[1]
[1]Delft University of Technology, [2]IMEC, [3]University of Twente
{l.c.a.huijbregts,s.hamdioui,r.k.bishnoi}@tudelft.nl,samantha.liu@imec.be,paul.detterer@imec.nl,a.yousefzadeh@utwente.nl

ABSTRACT

Current Artificial Intelligence (AI) computation systems face challenges, primarily from the memory-wall issue, limiting overall system-level performance, especially for Edge devices with constrained battery budgets, such as smartphones, wearables, and Internet-of-Things sensor systems. In this paper, we propose a new SRAM-based Compute-In-Memory (CIM) accelerator optimized for Spiking Neural Networks (SNNs) Inference. Our proposed architecture employs a multiport SRAM design with multiple decoupled Read ports to enhance the throughput and Transposable Read-Write ports to facilitate online learning. Furthermore, we develop an Arbiter circuit for efficient data-processing and port allocations during the computation. Results for a 128×128 array in 3nm FinFET technology demonstrate a 3.1× improvement in speed and a 2.2× enhancement in energy efficiency with our proposed multiport SRAM design compared to the traditional single-port design. At system-level, a throughput of 44 MInf/s at 607 pJ/Inf and 29mW is achieved.

ACKNOWLEDGMENTS

This work is partially funded by the European Union, DAIS (Grant No. 101007273), CONVOLVE (Grant No. 101070374) and NEUROKIT2E (Grant No. 101112268).

1 INTRODUCTION

The demand for Artificial Intelligence (AI) applications to operate on battery-powered Edge devices, such as smartphones, wearables, and diverse Internet-of-Things (IoT) systems, is rapidly growing. These devices handle substantial amounts of data that require processing through AI algorithms. As transmitting raw data can be energy-intensive and introduce higher latency and privacy risks, there is an increasing demand for the (partial) execution of AI applications directly on Edge devices. The primary solution being explored for low-power Edge AI involves neuromorphic computing and the utilization of *Spiking Neural Networks* (SNNs). This approach leverages large-scale parallel operations, integrating advanced techniques like *Computation In-Memory* (CIM), event-based computation, and the reduction of parameter precision [1]. The main challenge for neuromorphic accelerators involves improving the speed and efficiency of *Multiply-and-Accumulate* (MAC) operations.

In digital CIM, synaptic weights are stored in *Static Random Access Memories* (SRAM), where MAC operations are predominantly conducted through either Adder Trees and their variants [2–5] or *Multiplication In-Memory with sequential accumulation in the SRAM Periphery* (CIM-P) [6–9]. Adder Trees allow enhanced parallelism but come at the price of disrupting the SRAM structure and introducing considerable hardware overhead. In contrast, SRAM-based CIM-P designs minimize hardware overhead and efficiently leverage SNN sparsity, albeit with the trade-off of reduced parallelism in the pre-synaptic neuron dimension. This is because, in typical SRAM digital designs, only one row can be accessed at a time, resulting in the capability for only one pre-synaptic neuron to fire per timestep. Implementations like [9] strive to address this concern through approximate computing, yet this compromises classification accuracy. Another challenge with CIM-P involves spike arbitration; ensuring that only one spike enters the SRAM per timestep. Generally, these arbitration systems are extensive and demand multiple clock cycles per spike [6]. Additionally, on-chip learning is a popular practice for SNNs, enabling adaptability to dynamic environments and training with smaller datasets. On-chip learning is performed much more efficiently with transposable SRAM, facilitating access to cells in both row- and column-wise directions [7, 10–13]. However, the majority of methods either necessitate additional hardware components in the SRAM array, adversely impact cell stability, or lead to slow and high-power Read/Write operations. Hence, there is a decisive need for a cost-effective solution for SRAM-based CIM that can deliver a high degree of parallelism without compromising the classification accuracy and which enables efficient on-chip learning.

In this paper, we propose an SRAM-based SNN accelerator using CIM for low-power Edge AI applications, called ESAM. To solve the aforementioned problems with the state-of-the-art we implement:

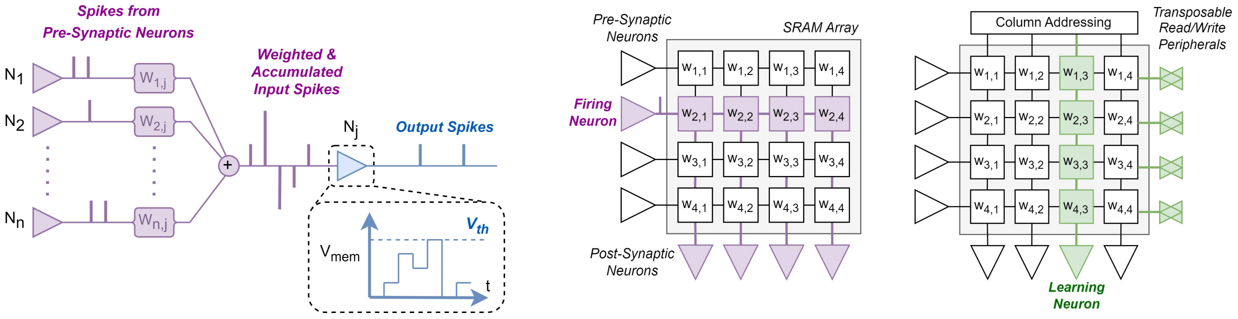
- Enhanced pre-synaptic neuron parallelism through novel multiport SRAM cells for synaptic weight storage, each equipped with multiple decoupled Read ports;
- A novel multiport Arbiter circuit designed to assign input spikes to their designated port channels;
- Transposable Read/Write access to the multiport SRAM array, facilitating online learning.

The multiport SRAM cell as well as Arbiter circuit and other peripheral circuits are designed using IMEC's 3nm FinFET technology. Simulation results for a 128×128 array show that our proposed SNN architecture can improve SNN computation speed and energy-efficiency by 3.1× and 2.2× respectively, in comparison to the standard single port SRAM design. Employing the proposed design at system-level and executing a handwritten digit classification application achieves a throughput of 44 MInf/s at just 607 pJ/Inf, while consuming 29 mW of power.

The rest of this paper is organized as follows: Section 2 presents the basics of SNNs, as well as the concept of Transposable crossbar memories. Section 3 presents the proposed accelerator architecture with different SRAM cell options, after which Section 4 does evaluations. Finally, Section 5 concludes the paper.

DAC '24, June 23–27, 2024, San Francisco, CA, USA

ACM ISBN 979-8-4007-0601-1/24/06.
https://doi.org/10.1145/3649329.3656514


Spikes from
Pre-Synaptic Neurons
N_1
N_2
N_n
$W_{1,j}$
$W_{2,j}$
$W_{n,j}$
Weighted &
Accumulated
Input Spikes
+
N_j
Output Spikes
V_{th}
V_{mem}
t
Pre-Synaptic
Neurons
SRAM Array
Firing
Neuron
$w_{1,1}$ $w_{1,2}$ $w_{1,3}$ $w_{1,4}$
$w_{2,1}$ $w_{2,2}$ $w_{2,3}$ $w_{2,4}$
$w_{3,1}$ $w_{3,2}$ $w_{3,3}$ $w_{3,4}$
$w_{4,1}$ $w_{4,2}$ $w_{4,3}$ $w_{4,4}$
Post-Synaptic
Neurons
Column Addressing
Transposable
Read/Write
Peripherals
$w_{1,1}$ $w_{1,2}$ $w_{1,3}$ $w_{1,4}$
$w_{2,1}$ $w_{2,2}$ $w_{2,3}$ $w_{2,4}$
$w_{3,1}$ $w_{3,2}$ $w_{3,3}$ $w_{3,4}$
$w_{4,1}$ $w_{4,2}$ $w_{4,3}$ $w_{4,4}$
Learning
Neuron

(a) Illustration of SNN working principle (b) Row-wise access for Inference (c) Column-wise access for learning

Figure 1: Illustration of working principle of Spiking Neural Network, mapping to memory crossbar and traditional row-wise SRAM access versus Transposable column-wise SRAM access.

2 BACKGROUND

2.1 Digital In-memory for Spiking Neural Networks

Spiking Neural Networks (SNNs) represent a significant advancement from traditional Neural Networks [14] by introducing a more biologically inspired model, where neurons communicate through discrete spikes, enhancing the temporal aspects of information processing and enabling more efficient modeling of complex patterns in dynamic data. These spikes contribute to an increased information-carrying capacity and facilitate event-based computing, thereby minimizing energy consumption. Figure 1(a) shows a visual example of an SNN. The Figure shows an *Integrate-and-Fire* (IF) neuron, which accumulates weighted spikes as its membrane potential (V_{mem}) and sends out a spike when V_{mem} exceeds its threshold potential; $V_{mem} \geq V_{th}$. The digital implementation of such SNNs offers advantages including enhanced energy efficiency, parallel processing capabilities, and compatibility for hardware acceleration, rendering them particularly suitable for real-time, low-power applications in domains like Edge computing. In digital *Computational-In-Memory* (CIM), synaptic weights are mapped to a memory crossbar, as shown in Figure 1(b). Here, *Multiply-and-Accumulate* (MAC) operations are mainly conducted through Adder Tree-based designs [2–5] or through *sequential accumulation in the SRAM Periphery* (CIM-P) [6–9].

2.2 Transposable SRAM for On-Chip Learning

On-chip learning is a prevalent practice in SNNs, allowing the network to continuously learn even after deployment, ensuring adaptability to changing environments. For efficient on-chip learning, it is crucial to have access to the synapse weights in both the pre-synaptic and post-synaptic dimensions [16–18]. Inference necessitates reading in the pre-synaptic dimension, corresponding to a memory row, as indicated in Figure 1(b). Contrarily, learning in SNNs typically occurs when particular conditions arise in the post-synaptic neuron. Hence, weight updates should be applied to all synapses preceding the learning neuron, associated with a memory column, as depicted in Figure 1(c). Overall, row-wise Read access and column-wise Read/Write access are essential for SNN Inference and on-chip learning, respectively. SRAM with access in both the row-wise and column-wise directions is called *Transposable* SRAM.

3 PROPOSED ESAM TECHNIQUE FOR CIM ARCHITECTURE

In this section, we present our proposed SNN accelerator. We first provide an overview of the complete ESAM (Energy-efficient SNN Architecture for Multiport SRAM) system. Subsequently, we provide a more detailed explanation of the innovative subsystem components: the bitcell design, the Arbiter, and the Neuron design.

3.1 Architecture Overview

As discussed, SRAM-based Computational-In-Memory (CIM) employing Adder Trees to perform the MAC operation results in substantial hardware overhead. Conversely, CIM-P designs reduce hardware overhead and efficiently leverage the sparsity of SNNs; however, they suffer from significantly lower parallelism. The issue lies in the fact that conventional SRAM-based CIM-P designs access memory rows one by one to execute the MAC operation. To enhance the parallelism in the CIM-P architecture, we enable multiple simultaneous accesses to the array through novel multiport processing configurations. This improvement requires addressing input encoding, considering the random nature of input spikes, and managing the accumulation process during MAC operations. To tackle these aspects, we introduce an innovative Arbiter-based encoding scheme and a novel Neuron design capable of handling multiple accesses. Additionally, these multiport capabilities simplify the Transposable arrangements necessary for on-chip learning.

Figure 2 presents a Macro-level overview of the architecture, illustrating the composition of a single CIM-P Tile. Each Tile executes the MAC operation, involving incoming input spikes and the synapse weights stored in the SRAM array. Multiple Tiles can be integrated to constitute a multi-layer neural network. To create a fully connected neural network, Tiles can be cascaded directly, and spikes are transmitted fully in parallel as binary pulses between the Tiles, negating the need for decoding or routing the spikes. Each Tile contains three novel components, highlighted in Figure 2: the Arbiter, the SRAM cells, and the Neuron Array. The Arbiter processes requests for sending spikes and accommodates up to p

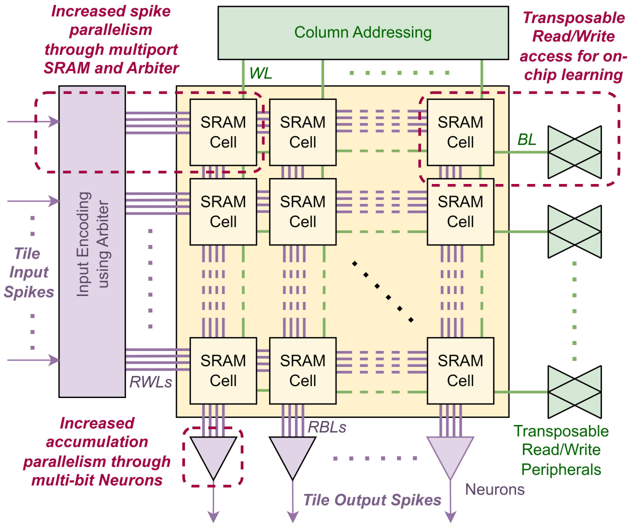

Increased spike parallelism through multiport SRAM and Arbiter
Column Addressing
Transposable Read/Write access for on-chip learning
WL
SRAM Cell
BL
Tile Input Spikes
Input Encoding using Arbiter
RWLs
RBLs
Increased accumulation parallelism through multi-bit Neurons
Tile Output Spikes
Neurons
Transposable Read/Write Peripherals

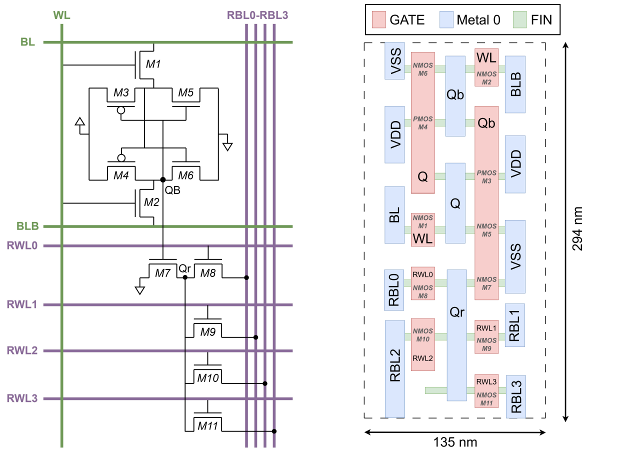
Figure 2: Overview of proposed Macro Architecture. Green indicates the Transposed Read/Write access, with the Inference Read access in Purple.

(p = 4 in Figure 2) spike requests per clock cycle, resulting in the activation of corresponding Inference Wordlines (RWLs, depicted in purple). The Read outputs are then conveyed on the Inference Bitlines (RBLs), which then proceed to the Neurons. At the Neurons, these outputs are added together in parallel, after which they determine changes in the membrane potentials V_{mem}. Moreover, as depicted in Figure 2 in green, the Transposed Wordlines (WL) and Bitlines (BL) enable column-wise Read/Write access to the SRAM array. This notably decreases the number of operations required to update a column of synapse weights, thereby enhancing the overall efficiency of on-chip learning.

3.2 Transposable Multiport SRAM cell-based Synapse

To enhance parallelism and improve the throughput of the architecture, we implement increased accessibility through a multiport SRAM cell design featuring several Read ports capable of processing multiple spikes in parallel. Furthermore, to enhance the efficiency of online learning, we require additional column-wise accessibility. Therefore, to fulfill these requirements, we propose a new multiport cell design that includes one dedicated column-wise Read/Write port and several row-wise Read ports.

Figure 3(a) shows the schematic of our proposed one-Read/Write and four-Read (1RW+4R) SRAM cell. The original 6T SRAM cell, comprising transistors M1-M6, remains unchanged; however, in contrast to common convention, its Word Line (WL) runs vertically, while its Bit Line (BL) and Bit Line Bar (BLB) run horizontally, enabling column-wise Read/Write access. M7 provides access to the cell content via the inverted cell content node QB. Through M7, Qr is discharged to VSS if QB is '1', meaning it effectively mirrors the content Q of the cell. M8-M11 then provide access to Qr to one of the added Bitlines RBL0-RBL3, depending on which of the added Wordlines RWL0-RWL3 is set to '1'. To perform a Read operation, an added Bitline, RBLx, is precharged to V_{prech}. When the corresponding WLx is driven to '1', the corresponding access transistor conducts. Depending on QB, RBLx is then either kept at V_{prech} or discharged to VSS, which is then sensed.



(a) Cell schematic

(b) Cell layout (using IMEC's 3nm FinFET)

Figure 3: Schematic and layout of proposed five-Read and one-Write (1RW+4R) SRAM Cell.

By connecting M7 only with its Gate to the content of the cell, a decoupled port is formed. This minimizes the influence of the added ports to the stability of the cell. It also enables us to scale V_{prech} to lower values than VDD with almost negligible impact on the cell stability. This results in power savings at the cost of slower precharging. Figure 3 presents the 1RW+4R cell, which has four dedicated decoupled Read ports and one standard Read/Write port. Similar to this, we have also designed a 1RW+1R, 1RW+2R, and 1RW+3R, each having one, two, and three decoupled Read ports, respectively, alongside one standard Read/Write port. These cells can be inferred from the structure in Figure 3(a) by removing Bitlines, Wordlines, and access transistors. The layout of the 1RW+4R cell is shown in Figure 3(b).

To sense the BL/BLB of the Transposed port, we have employed the traditional voltage-based differential Sense Amplifier and used row-muxing by a factor of four to match the SRAM row pitch. For RBL0-RBL3 single ended Reading, we have employed cascaded inverter-based Sense Amplifiers to be able to match the pitch of the SRAM columns, which deliver a slightly slower readout result than traditional Sense Amplifiers.

3.3 Arbiter Design

The Arbiter design is implemented to manage spike requests to maximize port utilization and process a higher number of spikes to enhance the overall parallelism of the CIM-P architecture. The Arbiter takes a spike Request vector R as input. The vector contains '1's which indicate for which SRAM Wordlines there are pending spike requests. Figure 4(a) shows the structure of the 4-port Arbiter, built out of four cascaded 1-port Arbiters. The 1-port Arbiter is implemented as a Fixed Priority Encoder for this work. This Priority Encoder is highlighted in Figure 4(b). It is built of a string of identical logic-based subblocks, highlighted in 4(c).

The Priority Encoder takes R and selects the leftmost '1' in the vector, creating one-hot Grant vector G. The signal $s[n]$ is used to

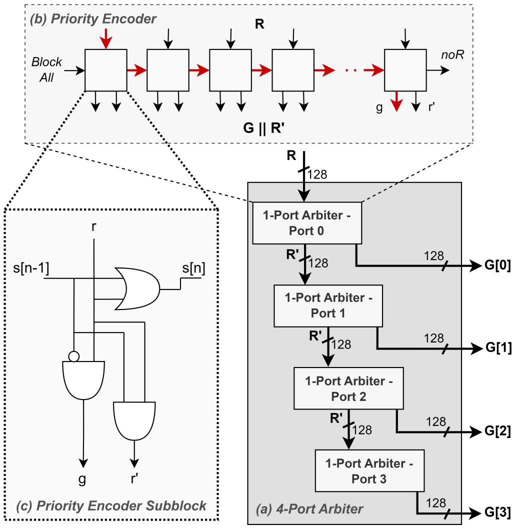


Figure 4: The proposed logic-based 4-Port Arbiter based on four cascaded 1-Port Arbiters.

block any requests further to the right of the selection. If R does not contain any spike, *noR* is made '1'. The vector R' is the same as R except the selected spike is masked out. These non-granted spikes R' are passed to subsequent 1-port Arbiters in a cascaded fashion to extend the system to multiple ports, generating multiple G-vectors within a single clock cycle.

In principle the Priority Encoder can be used as an Arbiter by itself, as shown in Figure 4. However, for a full SRAM array of > 128 rows, this results in an excessively long critical path inside the Priority Encoder, highlighted in Red in Figure 4(b). To mitigate this, in practice the 1-port Arbiter is not implemented as a singular Priority Encoder, but instead by combining multiple shorter Priority Encoders in a tree structure. Multiple shorter base Priority Encoders process the actual request vector R, while higher-level Priority Encoders in turn arbitrate amongst these base Priority Encoders. Note that both the base and higher-level Priority Encoders still utilize the same structure as presented in Figure 4(b). At the cost of 8.0% area overhead this reduces the critical path from >1100ps to <800ps for the 128-wide, 4-port Arbiter.

3.4 Neuron Design

Figure 5 shows a high-level overview of the Neuron subcomponent. For this work an Integrate-and-Fire (IF) neuron was chosen, as the test setup involves a time-static classification task. The Neurons take in the sensed data from the p multiport Bitlines of the SRAM arrays, represented by {'1'/'0'}. For every Bitline, a validity flag is used to indicate which memory ports were actually used in every clock cycle. This ensures an unused port is not erroneously read as a '1' and added to the membrane potential. The valid Bitlines are decoded to {'+1'/'-1'}, added together, and then added to the m-bit V_{mem} register. The V_{th} of every individual Neuron is stored in its own t-bit register. When all input spikes of a Tile have been served by the Arbiter, *R_empty* becomes '1', which enables the comparison of V_{mem} and V_{th}. If $V_{mem} \geq V_{th}$, Neuron output register r is set to '1' to indicate the request to send a spike to the subsequent Tile.

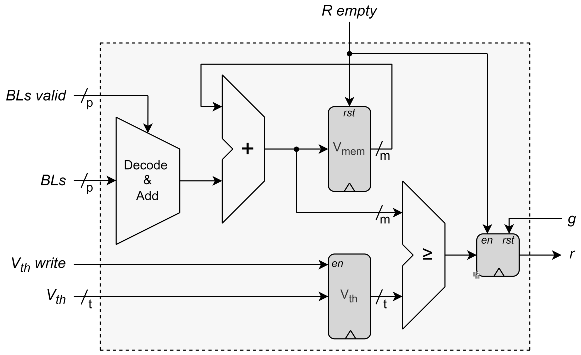


Figure 5: Proposed Neuron for ESAM Architecture.

Simultaneously, V_{mem} is reset to zero to start accumulating spikes again. If the Neuron's spike request r is granted (g = '1'), r is reset to '0'.

4 RESULTS

4.1 Experimental Setup

Table 1 summarizes the experimental setup utilized in this work. Note that in order to improve the SRAM Write operation, the *Negative BL* (NBL) assist technique [19] is used, which creates a lowered voltage ($V_{WD} < VSS$) on the complementary Bitline to force the cell to the desired state. This technique is necessary due to the high parasitics at smaller technology nodes. The required V_{WD} is determined for various array sizes, and if it is necessary for V_{WD} to be $< -400mV$, the array size is considered non-valid for implementation due to low expected yield [19]. This restriction limits the array size to $\leq$ 128 rows and columns for all cell designs.

The synthesis results, combined with the SRAM Macro outcomes, are utilized to simulate the network on a spike-by-spike basis in Python and determine the timing, power, and energy at the system-level.

4.2 Circuit-Level Evaluation

From layout, we have observed that the area of standard 6T is $0.01512\mu m^2$ [20]. The areas of the 1RW+1R, 1RW+2R, 1RW+3R and 1RW+4R cells are 1.5×, 1.875×, 2.25× and 2.625× larger respectively. We explored the possibility of adding 5+ ports, but only 4 Bitlines could match the pitch of the 4-port cell. Adding another port would require further widening of the cell, increasing the area by 87.5% of the 6T cell, making it too area-inefficient.

Table 1: Experimental Setup Details

Parameter	Specification
Technology Node	IMEC 3nm FinFet
Supply Voltage	700mV (V_{prech} = 500mV, for single-ended)
Simulation	Cadence Spectre
Synthesis	Cadence Genus
Parasitics Extraction	Calibre PEX + Line Geometries and Node Datasheets
Process Variation	±3σ
SRAM Target Cell	Worst-case Cell/Row/Column

Figure 6 shows the time and energy measurements for Writing to the cell and Reading from the cell using the Transposed port (WL and BL/BLB). Write time is the delay between the start of the Write process and the cell content flipping to 90% of its intended value. Read time is the delay between the Wordline being driven and the data output of the Sense Amplifier flipping. Additionally, Write energy is the energy consumed during the Write time, while Read energy is the energy consumed during a full clock cycle, which includes precharging of the BL/BLB [21].

As expected, both the Write and Read operation results scale with the addition of ports due to the parasitics that these ports introduce. The effect is stronger for the Write operation, as the parasitics also require a lower value of V_{WD} when more ports are added, increasing the voltage differential and consequently the power consumption. Note also that when just one extra Inference Port is added, there is an immediate and significant increase in both Write and Read times of the Transposed port. This is because the WL wire in the proposed cells is narrower and thus more resistive, which is necessary due to the new RBL0-RBL3 that have to be routed in the same metal layer.

Figure 7 demonstrates the relationship between access time and energy consumption for different V_{prech} levels and different numbers of ports in the SRAM. The results are shown for full utilization of all available ports, meaning that if a cell has p ports, p Read operations are performed. Total access time is calculated as the sum of the precharge time and the Read time.

- **The effect of precharge voltage:** As Figure 7 indicates, there is a trade-off between access time and energy. We select $V_{prech} = 500mV$, as it leads to a reduction of at least 43% in energy consumption at the cost of at most 19% higher access time for all port numbers. Lowering V_{prech} from $500mV$ to $400mV$ saves up to 10% more energy for 1- and 2-port designs. However, for 3- and 4-port designs energy consumption actually increases due to much slower precharging.
- **The effect of the number of Inference ports:** Adding extra Inference ports increases the parallelism and reduces the average access time. However, the average access energy starts increasing after adding the fourth port. This increase in energy consumption per port can be explained by the increased cell size that creates more parasitics. These results support our earlier claim that increasing the ports to more than four would not be beneficial due to the excessive area and resulting parasitics.

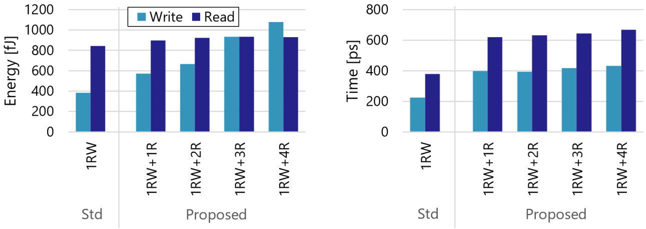


Figure 6: Write and Read Energies and Timings via Transposed port for different types of SRAM Cells.

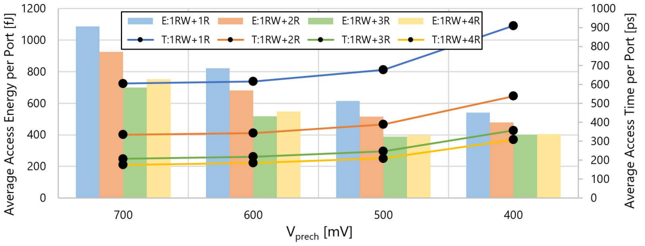


Figure 7: Illustration of the average access energy and time per number of ports for different V_{prech} values and various ports for 128×128 arrays.

4.3 Timing Evaluation

To calculate the system clock period, we need to determine the duration of each pipeline stage. Table 2 shows the durations of each stage for the various cell designs, including slack. The longest of the two stages determines the minimum clock period for each design. The Table shows that the critical path of the Arbiter does not scale with added ports. As a result, with more added ports the SRAM Read + Neuron accumulation stage becomes the bottleneck.

4.4 System-Level Evaluations

4.4.1 Online Learning. Online learning becomes significantly more efficient with Transposable memory access. Without the presence of the Transposed ports, reading and writing all the weights in an array of 128×128 6T (1RW) SRAM cells would require 2×128 clock cycles, consuming 257.8 ns and 157 pJ. For our proposed SRAM cells, the synaptic weights of a post-synaptic Neuron can be read and written in just 2×4 cycles, where the factor 4 is due to the use of 4-to-1 MUXs. The 4-port cell, identified as the worst performer in the Transposed Read/Write port configuration, operates with a clock period of 1.2 ns. This means it requires only 9.9 ns (26.0× less) and 8.04 ns (19.5× less) to read and write a full column.

4.4.2 Inference. To evaluate the system's Inference performance, we have created a Fully Connected Binary-SNN network for MNIST digit classification by placing multiple Tiles in sequence. The network has a structure of *768:256:256:256:10*. In case a layer exceeds the maximum SRAM array size, multiple SRAM arrays are used in a single Tile. Each SRAM has its own 128-wide Arbiter. This increases parallelism by another factor; a 256-wide layer will have two p-port Arbiters, meaning up to $2 \times p$ spikes can be selected per clock cycle. In order to reduce the input images from 784 to 768 pixels, a 2×2 set of pixels is removed from every corner of the images. This ensures that the first layer corresponds to exactly 6×128 inputs. We have trained the network as a *Binary Neural Network* (BNN) with a sign activation function and per-neuron biases. The BNN is then converted to a Binary-SNN with per-neuron thresholds, as described in [15]. The resulting accuracy achieved by the network is 97.64%.

Table 2: Time required for each stage in the pipeline for different SRAM cells, with the longest of the two stages indicating the clock period for each cell (1RW is the standard 6T cell).

	1RW	1RW+1R	1RW+2R	1RW+3R	1RW+4R
Arbiter	1.01ns	1.01ns	1.04ns	1.03ns	1.01ns
SRAM + Neuron	0.69ns	1.08ns	1.18ns	1.14ns	1.23ns

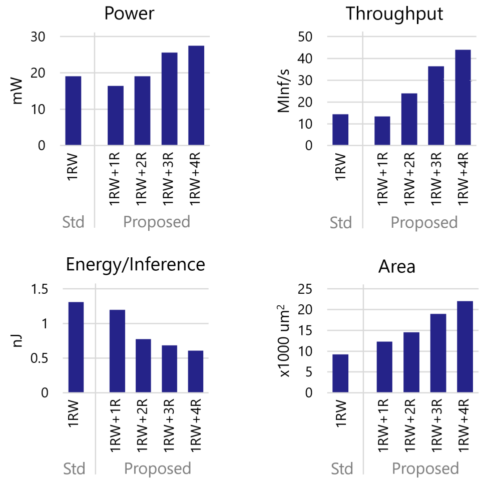


Figure 8: System-level comparison between using standard 6T (1RW) SRAM Cell and our proposed cells.

Figure 8 provides a comparison of system-level power, performance, energy, and area for the five discussed SRAM cell options. Generally, the power of the system increases with the number of added ports. However, the system's power implemented with the standard 1RW cells is higher than that of the 1RW+1R and 1RW+2R cells. This is due to the active power savings from the voltage scaling of V_{prech} for the decoupled ports. When comparing the 1RW and 1RW+1R cells, throughput decreases slightly, as the effective parallelism is the same, but read operations for the 1RW+1R cell are slower due to the added parasitics. However, at 2+ added ports, the increased parallelism compensates for this. Additionally, with every added port, the overall energy/Inference decreases significantly due to the increased spike throughput. The main downside of the multiport design is that the area of the design using the 1RW+4R cell is 2.4× larger than when using the standard 1RW cell.

Table 3 compares the 1RW+4R cell with state-of-the-art SNN accelerators aimed at ultra-low-power applications. We achieved typical overall power consumption, but significantly improve energy/Inference and throughput. Note that the presented implementation is biased heavily towards high throughput. For applications that have lower throughput demands, a lower VDD, lower clock frequency, and HVT transistors can be utilized to significantly reduce power consumption, while maintaining similar energy/Inference.

5 CONCLUSION

In our research paper, we presented an SRAM-based CIM accelerator designed for SNNs in IMEC's 3nm FinFET. The synthesis results indicate that our system exhibits high classification accuracy, while also reducing system-level power consumption and increasing the number of Inferences per second by four orders of magnitude. These accomplishments were made possible through the use of a new multiport SRAM cell that enables enhanced parallelism and voltage scaling, an improved spike arbitration system, a low-cost communication fabric, and technology scaling which is made possible by the fully digital design.

Table 3: Comparison of the 1RW+4R-based system with state-of-the-art Small-scale SNN Accelerators.

	[6]	[9]	[10]	*This Work*
Technology [nm]	65	10	65	3
Neuron Count	650	4096	1K	778
Synapse Count	67K	1M	256K	330K
Activation Bit Width	6	1	–	1
Weight Bit Width	1	7	5	1
Transposable	No	No	Yes	Yes
Clock Frequency	70kHz	506MHz	100MHz	810MHz
MNIST				
Power	305nW	196mW*	53mW	29.0mW
Accuracy [%]	97.6	97.9	97.2	97.6
Throughput [inf/s]	2	6250	20	44M
Energy/Inf [nJ]	195	1000	–	0.607

* Inferred from SOP/s/mm^2, Area, and pJ/SOP

REFERENCES

[1] Y. Lee, P. Tsung, and M. Wu, "Techology trend of edge AI," in VLSI-DAT 2018, pp. 1–2.
[2] H. Fujiwara et al. "A 5-nm 254-TOPS/W 221-TOPS/mm2 Fully-Digital Computing-in-Memory Macro Supporting Wide-Range Dynamic-Voltage-Frequency Scaling and Simultaneous MAC and Write Operations," in ISSCC 2022, pp 1–3.
[3] Y. Chih et al. "An 89TOPS/W and 16.3TOPS/mm2 All-Digital SRAM-Based Full-Precision Compute-In Memory Macro in 22nm for Machine-Learning Edge Applications," in ISSCC 2021, pp. 252–254.
[4] D. Wanq et al. "All-Digital Full-Precision In-SRAM Computing with Reduction Tree for Energy-Efficient MAC Operations," in ICTA 2022, pp. 150–151.
[5] D. Wang et al. "DIMC: 2219TOPS/W 2569F2/b Digital In-Memory Computing Macro in 28nm Based on Approximate Arithmetic Hardware," in ISSCC 2022, pp. 266–268.
[6] 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 A-SSCC 2020, pp. 1–4.
[7] J. Seo et al. "A 45nm CMOS neuromorphic chip with a scalable architecture for learning in networks of spiking neurons," in CICC 2011, pp. 1–3.
[8] D. Kim et al. "MONETA: A Processing-In-Memory-Based Hardware Platform for the Hybrid Convolutional Spiking Neural Network With Online Learning," in Frontiers in Neuroscience 16, 2022.
[9] G. 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," in JSSC 54.4, 2019, pp. 992–1002.
[10] J. Kim et al. "Efficient Synapse Memory Structure for Reconfigurable Digital Neuromorphic Hardware," in Frontiers in Neuroscience 12, 2018.
[11] J. Wang et al. "A Compute SRAM with Bit-Serial Integer/Floating-Point Operations for Programmable In-Memory Vector Acceleration," in ISSCC 2019, pp. 224–226.
[12] S. K. Bose and A. Basu, "A 389TOPS/W, 1262fps at 1Meps Region Proposal Integrated Circuit for Neuromorphic Vision Sensors in 65nm CMOS," in A-SSCC 2021, pp. 1–3.
[13] Z. Lin et al. "Two-Direction In-Memory Computing Based on 10T SRAM With Horizontal and Vertical Decoupled Read Ports," in JSSC 56.9, 2021, pp. 2832–2944.
[14] W. Maass, "Networks of spiking neurons: The third generation of neural network models," in Neural Networks 10.9, 1997, pp. 1659–1671.
[15] H. Kim, H. Oh, and J. Kim, "Energy-efficient XNOR-free In-Memory BNN Accelerator with Input Distribution Regularization," in ICCAD 2020, pp. 1–9.
[16] A. Yousefzadeh et al. "On practical issues for stochastic STDP hardware with 1-bit synaptic weights," in Frontiers in Neuroscience 12, 2018.
[17] A. Yousefzadeh et al. "Hardware implementation of convolutional STDP for on-line visual feature learning," in ISCAS 2017, pp. 1–4.
[18] A. Rostami et al. "E-prop on SpiNNaker 2: Exploring online learning in spiking RNNs on neuromorphic hardware," in Frontiers in Neuroscience 16, 2022.
[19] H.-H. Liu et al. "Extended Methodology to Determine SRAM Write Margin in Resistance-Dominated Technology Node," in TED 69.6, 2022, pp. 3113–3117.
[20] H.-H. Liu et al. "DTCO of sequential and monolithic CFET SRAM," in SPIE 12495, 2023.
[21] H.-H. Liu et al. "CFET SRAM DTCO, Interconnect Guideline, and Benchmark for CMOS Scaling," in TED 70.3, 2023, pp. 883–890.