# SAFER: Sparsity Integrated Compute-in-Memory AI Accelerator with a Fused Dot-Product Engine and a RISC-V CPU

Amitesh Sridharan*, Asmer Hamid Ali*, Yongjae Lee*, Anupreetham Anupreetham†, Yaotian Liu*, Jeff Zhang*, Jae-sun Seo†, Deliang Fan*

*Arizona State University, Tempe, AZ, USA. Email: {asridh25, dfan}@asu.edu

†Cornell Tech, New York, NY, USA. Email: {js3528}@cornell.edu

***Abstract*—We present a sparsity-aware in-SRAM multiply-and-accumulate (MAC) accelerator with a fused dot-product engine (SAFE) and a RISC-V CPU (SAFER). For the first time, we implement a unified dot-product compute methodology in Compute-in-memory (CIM) circuits vastly reducing the hardware footprint for simultaneously supporting both floating point (FP) and integer (INT) MACs. Additionally, we integrate various $N$:$M$ sparsity formats allowing the CIM macro to store and operate exclusively on compressed non-zero weights. We also tightly integrate a 32-bit RISC-V CPU to SAFE for efficient data-movement across chip. The 28nm SAFER prototype achieves a peak energy efficiency of 105.7 TOPS/W (78.9 TOPS/W) and 79.9 TOPS/W (63 TOPS/W) in the macro (chip) level for FP8 and INT8 workloads respectively. SAFER also achieves a memory footprint reduction proportional to sparsity through compressed storage, vastly reducing the macro count required for large AI models. For our proposed figure of merit which accounts for PPA along with memory footprint, and for this FoM SAFER improves current SoTA CIMs by 13.8× for FP8 workloads.**

***Index Terms*—Digital in-memory computing, Multiply and accumulate, Floating-point and Integer acceleration, RISC-V.**

## I. Introduction

Digital CIM architectures have a successful track record in performing efficient matrix multiplications (MM) for AI workloads, thanks to their high memory bandwidth and tight compute-memory coupling. There have been a plethora of CIM designs targeting various hardware and software features to enable efficient processing of deep neural networks (DNNs).

**Sparsity** is one such feature and has been widely adopted in various DNNs. DNN models are getting larger and is becoming more challenging to fit them on chip. This is further exacerbated by the fact that current CIMs have poor storage density due to integration of compute logic circuits. Weight sparsity achieved through pruning in DNNs, offers significant memory footprint reduction when paired with compression. Compressed storage can also reduce memory accesses to CIMs when mapping large DNN workloads. Prior CIM works have explored this: [1] uses bitmaps for compression but requires external compute and complex dense-format conversion via butterfly multiplexers. SP-IMC [2] adopts compressed sparse column (CSC) with simplified decode hardware but suffers from low MAC utilization. SAFE aims to address these drawbacks by offering a simple decode mechanism for various sparsity ratios and full utilization of the compute circuits.

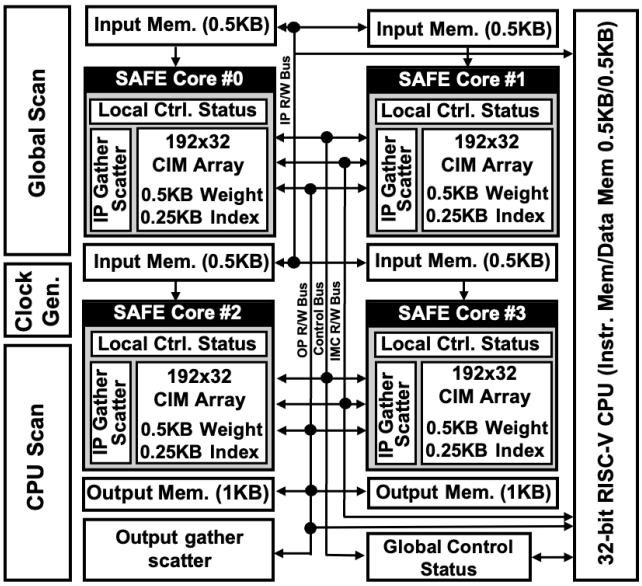


Fig. 1. SAFER chip architecture.

**Data-type re-configurability** is another feature that enables support for various DNNs. Supporting different number formats such as integer (INT) and floating-point (FP) precision in CIM hardware is expensive and needs careful hardware reuse. Recent CIM works have explored this, but schemes in [3], [4] trade off compute accuracy in FP arithmetic for hardware complexity, whereas [5] has no accuracy drop but incurs large hardware overhead. SAFE, for the first time, explores a fused-dot product (FSD) approach in CIM, which was previously only employed in ASICs [6], [7]. This method can reduce the cost of rounding and normalization for large vector-vector MACs [6]. Each column in CIM arrays typically performs large vector-vector MACs, therefore FSD can naturally fit well with CIM array design. Additionally, FSD scheme uses fixed-point adders for accumulations. Adder trees occupy a large footprint in digital CIMs, which can be amortized by using the same adder trees in FSD for both INT and FP.

**MAC configuration flexibility** is another key feature that requires attention as there are a variety of AI workloads all requiring different MAC structures. Self-attention layer in large language models (LLMs) differs from a convolution layer and even convolutions vary from layer to layer in different

TABLE I
RISC-V INSTRUCTION SET EXTENSIONS.

| Instruction Type | Function |
|---|---|
| **RV32IM Base** | **All ratified instructions** |
| **Load store instructions** *New opcode for each instruction, honors the same funct field for byte,1/2 byte, word. R-Type: Loads to CPU reg, stores to CPU D-mem. | **Load/Store IMC** |
| | **Load/Store IP mem** |
| | **Load/Store OP mem** |
| | **Load/Store global CSR** |
| **Floating point Extensions** *(R-type and I-type extension) | **FP8 E4M3/E5M2 8-vector addition** |
| | **FP8 E4M3/E5M2 multiply** |
| | **FP8 E4M3/E5M2 Single add** |
| | **32-bit INT 8-vector addition** |
| **Copy instructions** *Moves data from non CPU memory to desired location *(R-type and I-type extension) | **Copy desired IP mem location to OP mem location and vice versa** |

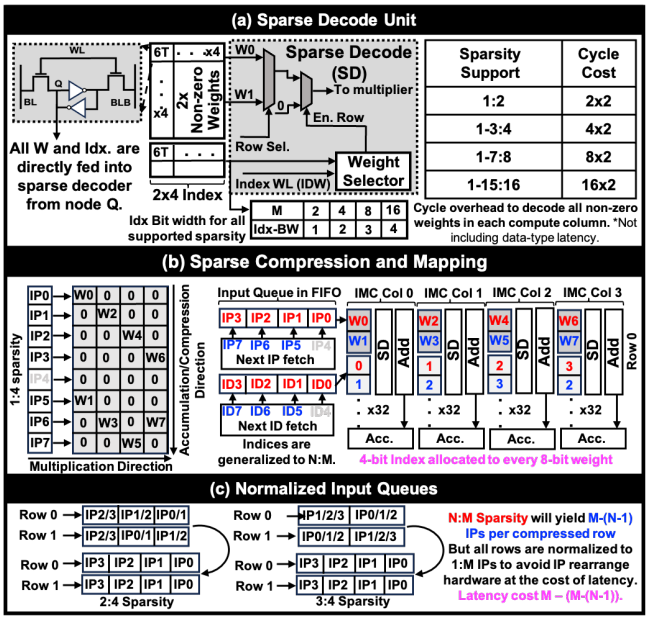


Fig. 2. (a) Sparse decode unit, (b) sparse compression and mapping, (c) reducing hardware complexity by normalizing input queue length.

DNNs. SAFER integrates a custom RISC-V CPU to cast weights and activations across all memories on chip to help with MAC reconfigurability. We also augment the RISC-V CPU with vector additions to help with post accumulation of partial sums for large matrices.

## II. ARCHITECTURE AND OPERATION

### A. SAFER Chip Architecture

Fig. 1 shows the SAFER chip architecture, consisting of four SAFE cores, input/output (IP/OP) buffers, control logic, and OP gather-scatter unit. To enable parallel processing, each SAFE core is allocated a dedicated 0.5KB IP buffer for feeding unique IPs. The OPs/partial-sums from all SAFE cores are collected and transferred to two 1KB OP buffers via the OP gather-scatter unit. We incorporate a custom 32-bit single-cycle RISC-V CPU to aid with the address calculations for data movement between IP/OP buffers and weights (Ws) in the CIM array. This is done by augmenting the base RV32-IM instruction set with additional load/store instructions. These additional instructions provide support for a variety of MAC

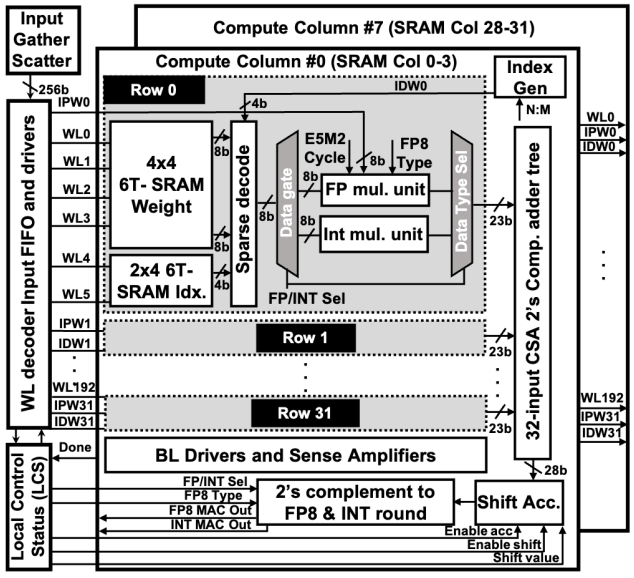


Fig. 3. SAFE core and macro architecture.

configurations through uni/multi-casting Ws/OPs/IPs to any/all of the SAFE cores. The CPU also monitors the status of all SAFE cores through global control status registers. Additionally, we include FP8 (E4M3 and E5M2) and INT32 vector addition to the CPU to enable partial-sum aggregations from SAFE macros for large matrices. Table I provides the list of all supported instructions.

### B. SAFE Macro Architecture

Fig. 3 illustrates a single SAFE core. Each core has a 192×32 CIM array which is broken into eight FSD "compute columns" (CCs) for compressed W storage and sparse FP8/INT8 MAC operations. Each core also has an input gather-scatter (IGS) which retrieves IPs and distributes them to the CCs through the IP FIFOs. The CIM array also has WL-decoders, BL-drivers, and sense amplifiers to facilitate row-by-row read and writes. The control logic manages compute modes and sparsity ratios. Status registers track MAC count and manage IP requests via the IGS.

*1) Compute Column (CC):* Each CC in a SAFE macro contains 192×4 6T-SRAM bits, out of which 128×4 is used for W storage and 64×4 for Idx storage. The CC also includes a 32-vector FSD MAC unit, supported by a backup SRAM with 64 storage locations for 8-bit weights. This backup SRAM allows a new set of weights to be written to the CIM array while the current set is actively used for computation. CC is divided into 32 rows, where each row has 4×4 bits for W storage and 2×4-bits for Idx storage. These Ws and Idxs are sent to a sparse decode unit which filters two 8-bit Ws using the two 4-bit Idxs into one weight (or 0) and this weight is sent to the multipliers. There is an individual FP8 and INT8 multiplier to handle both data-types. The outputs from the multiplier is sent to a 32-input 2's complement adder tree. The multipliers are data-gated to save power between different data types. Both data types share the same adder tree as the accumulation data format is normalized between the two. The outputs from the adder tree is then fed into a shift accumulator (SA). The shift portion of the SA can be enabled by the control

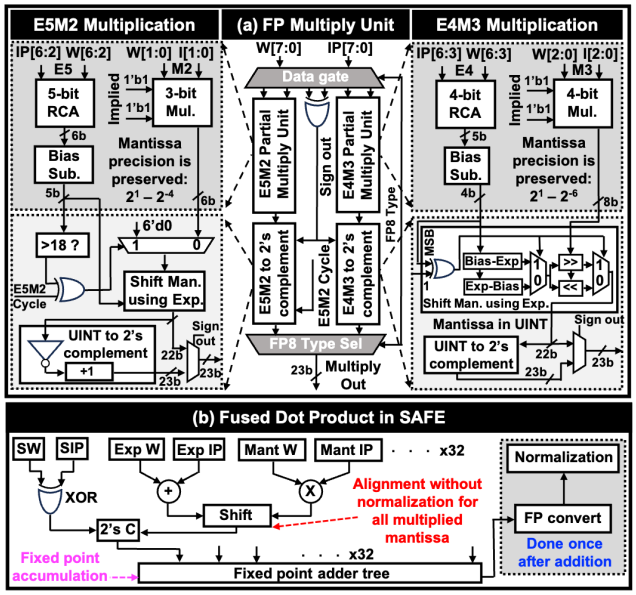


Fig. 4. (a) Floating-point multiply unit and (b) fused dot-product in SAFE.

logic depending on the data type. Now the 2's complement partial-sum from the SA can optionally converted back to FP.

*2) Sparse Decode Unit (SD):* As shown in Fig. 2, SAFE's sparse decode method supports various $N$:$M$ sparsity formats. It adopts a compressed sparse row (CSR) compression scheme, storing an index for each non-zero weight. The index bit-width is determined by the maximum supported $M$ in $N$:$M$. All supported $N$:$M$ ratios and their corresponding bit-width is shown in Fig. 2(a). The stored indices specify which weights the IPs must be multiplied with before accumulation. Each row in a CC has an Idx word line (IDWL); as IPs are streamed into the macro via the IPWs, the corresponding indices are simultaneously streamed through the IDWLs and the W selector uses these indices to determine whether the corresponding weight needs multiplication from the streamed-in IPs. This is required because when a matrix is compressed along the accumulation direction, it breaks the multiplication structure. Not all Ws need to be multiplied with the streamed-in IPs. We also time-multiplex sparsity, this is because there is only a single IPW for every W, if a 1:4 sparsity is implemented then for each CIM row, four IPs need to be streamed-in along with 4 Idxs and will vastly increase the routing resources. The sparse compression and mapping mechanism is shown in Fig. 2(b). For $N$:$M$ sparsity where $N{\neq}1$, each row would require $M{-}(N{-}1)$ IPs, and the IPs required in each row are not in-order. To avoid more hardware for input re-arrangement, we stream-in all $M$ IPs regardless of $N$. This normalizes the cycle count to $M$, as described in Fig. 2(c).

*3) FP8 fused dot-product (FSD) and INT8 MAC:* The FSD method supports vector-vector MACs for FP data-types, as shown in Fig. 4(b). FP multiplications begin with exponent addition and mantissa multiplication; instead of rounding, the mantissa's precision is preserved and is shifted by the exponent and mapped to a 2's complement number line. For E4M3, the number line spans $\pm 2^{9} - 2^{-12}$ (23 bits) and for E5M2 the number line spans $\pm 2^{16} - 2^{-18}$ (36 bits). Now that the

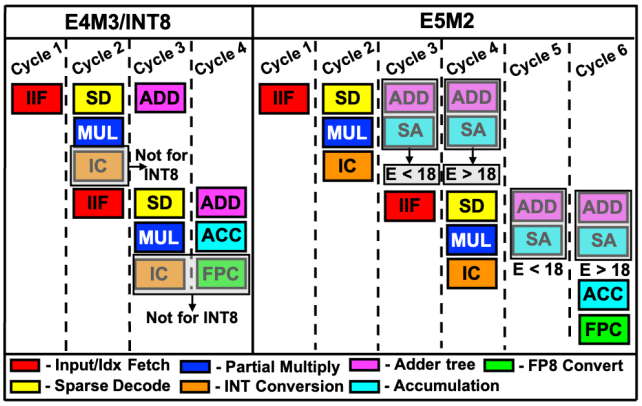


Fig. 5. Pipeline diagram for a 1:2 sparse workload.

vectors are in 2's complement format, they can be accumulated using a fixed-point adder. The adder-tree bit-width is set to 23 bits instead of 36 bits because of area constraints. To support E5M2, we time-multiplex the adder hardware over two cycles. The first cycle is used to handle all exponents below 18 and the next cycle is used for exponents above 18 and the shift accumulator performs shift and accumulate compuutation of the two partial sums. Fig. 4(a) shows the detailed FP multiply units in SAFE. To support INT8, an 8-bit integer multiplier is added to the pipeline after the SD stage, while the adder hardware is reused this is shown in Fig. 3. Fig. 5 illustrates SAFE's pipeline for all supported data types.

## III. Chip Measurements and Results

SAFER is prototyped in 28nm CMOS. It occupies $0.95mm^2$, and each SAFE core occupies $0.15mm^2$. Fig. 7 shows the power/area breakdown for a SAFE core. FP8 MACs consume less power and area, compared to INT8 MACs. This is due to the wide multiplier unit for INT8 and the 2's complement version of an FP8 number is very sparse and reduces the overall activity factor of the adder tree. SAFER can operate at 0.57-1.2V, reaching a $F_{max}$ of 141 MHz@0.57V and 815 MHz@1.2V. Fig. 6(a) shows the voltage-frequency scaling measurements for both FP8 and INT MACs, where the CPU was set to perform NoP instruction. All measurements were done at 28$^o$C. For INT8 workloads, we use an IP toggle rate of 50% and bit-wise W sparsity of 50%. FP8 workloads use randomly generated numbers for both IPs and Ws within the representable range. The SAFE(R) achieves a peak energy efficiency of 105.7 (78.9), 78.8 (59) and 79.9 (63) TFLOPS/W for FP8 E4M3, FP8 E5M2, and INT8 MACs respectively.

To quantify how well various sparsity translates to memory footprint, we map a ResNet-18 model ($\sim$11M parameters) trained for CIFAR-100 dataset for various 1:$M$ sparsity. With 1:16 sparsity, only 91 SAFE macros are required for the entire model, which marks 15.5$\times$ savings in macro area, as shown in Fig. 6(b). We also validate the accuracy for ResNet-18 (DNN) and a Llama-2-7b (LLM) model for FP8 and INT8 under various sparsity to demonstrate the practical need for different $N$:$M$ sparsity, as shown in Table II. To account for reduced memory footprint due to compressed storage, we devise a figure of merit (FoM) of TOPS/W/$mm^2$*Weights/Byte, which takes energy efficiency, chip area and memory size in bytes for

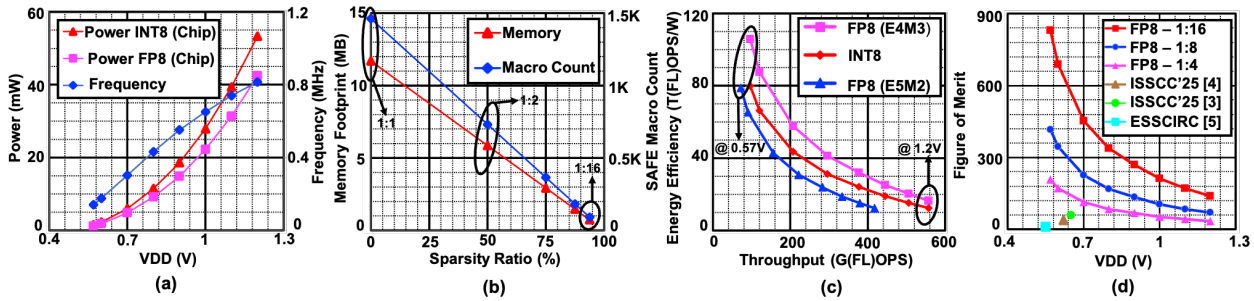


Fig. 6. (a) Power and frequency scaling, (b) sparsity savings, (c) throughput and energy efficiency scaling, (d) figure of merit.

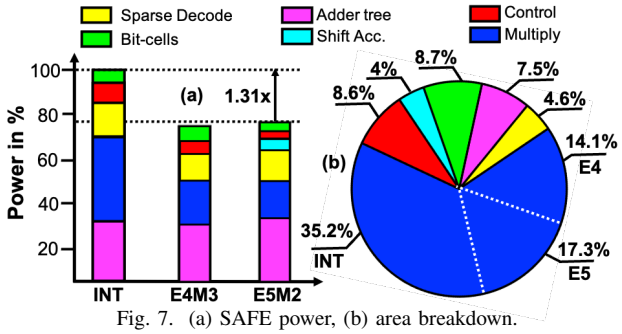


Fig. 7. (a) SAFE power, (b) area breakdown.

| | |
|---|---|
| Macro area | 0.15 mm² |
| Die area | 1.52 mm² |
| Channels | 32-IP & 8-OP/core |
| Alignment | One-shot post multiplication |
| Peak T(FL)OPS\W | 16-105.7(E4M3)<br>12-78.8 (E5M2)<br>12.4-79.9(INT8) |
| Peak G(FL)OPS | 24-139 INT8/E4M3<br>18-104(E5M2) |

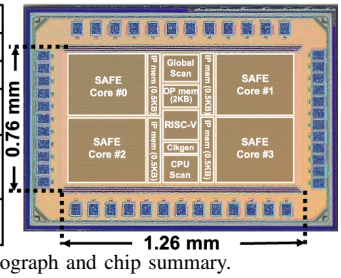


Fig. 8. Die-micrograph and chip summary.

8-bit weight parameters of various sparsity ratios in the CIM array. Table III and Fig. 6(d) show that SAFER achieves 13.8× improvements in this FoM for FP8 workloads, compared to prior SoTA digital CIMs.

## IV. Conclusion

In this work, we prototype SAFER, an in-SRAM sparse FSD-based CIM processor, which integrates a custom RISC-V CPU enabling support for a variety of AI models from DNNs to LLMs. Through FSD, SAFE enables FP8 MACs with minimal overhead while also supporting hardware sharing between FP8 and INT8 formats. Measurement results show that SAFE achieves close to SoTA TOPS/W and TOPS/$mm^2$ while maintaining full accuracy for all MAC workloads. SAFE also implements sparsity in the form of compressed storage, achieving memory footprint reduction proportional to sparsity. For the proposed FoM, SAFER shows 13.8× improvement compared to SoTA digital CIMs for FP8 workloads.

## V. Acknowledgement

This work is supported in part by the NSF under Grant No. 2314591, No. 2505326, No. 2528723, No. 2528767 and the CoCoSys Center in JUMP 2.0, an SRC Program by DARPA.

TABLE II
AI Model Accuracy for Various Sparsity Ratios.

| Model/Dataset | ResNet -18 on CIFAR-100 | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Act/W Precision | INT8/INT8 | | | | FP8/FP8 | | | |
| Sparsity ratio | 1:1 | 1:4 | 1:8 | 1:16 | 1:1 | 1:4 | 1:8 | 1:16 |
| Model accuracy (%) | 76 | 75.7 | 75.3 | 73.9 | 76.5 | 76.5 | 76.2 | 74.8 |
| Model | Llama-2-7b-hf (Post trained, not fine tuned) | | | | | | | |
| Act/W Precision | FP8/FP8 | | | | | | | |
| Performance Score | Wikitext2 (Lower is better) | | | | BoolQ 0-shot (Higher is better) | | | |
| Sparsity ratio | 1:1 | 2:4 | 4:8 | 8:16 | 1:1 | 2:4 | 4:8 | 8:16 |
| Model accuracy score | 5.1 | 12 | 8.3 | 7.2 | 0.77 | 0.67 | 0.72 | 0.74 |

TABLE III
Comparison with SoTA Fully Digital CIM Works.

| | | This work | ISSCC'25 [4] | ISSCC'25 [3] | CICC'24 [2] | ESSCIRC' 23 [5] |
|---|---|---|---|---|---|---|
| Technology | | **28nm** | 28nm | 28nm | 28nm | 28nm |
| Voltage (V) | | **0.57-1.2** | 0.62-0.9 | 0.65-0.9 | 0.57-1.18 | 0.55-1.2 |
| Frequency (MHz) | | **141-815** | 100-525 | 153-400 | 201-1160 | 650 |
| SRAM Cell | | **6T Logic** | 10T | 6T | 10T | 8T/14T |
| Macro size | | **24Kb (16KbW)** | 224Kb | 32Kb | 8Kb (4KbW) | 4Kb |
| Weight/Input Precision | | **INT8,FP8 (E4/5M3/2)** | FP16/8, INT8 | INT4/8 FP8,BF16 | INT4/8 | FP8 |
| Accuracy Loss | | **No** | Yes | Yes | No | No |
| Sparsity Support | | **CSR/N:M (50%-99%)** | Zero skip dynamic | None | CSC/N:M (1%-99.9%) | None |
| Energy Efficiency T(FL)OPS/W | FP8* | **16-105.7(PIM) 13-78.9(Chip)** | 99.7 | 192.3 | -- | 12.1 |
| | INT8 | **12.4-79.9(PIM) 10-63(Chip)** | 115 | 71.4 | 57.7 | -- |
| Compute density T(FL)OPS/mm² | FP8* | **0.3-1.79&** | 3.3 | 3.23 | -- | 0.94 |
| | INT8 | **0.24-1.3#** | 3.81 | 2.34 | 1.2 | -- |
| Figure of Merit$: T(FL)OPS/W/ mm2* Ws/byte, where W is an 8-bit weight | FP8* | **8.6-51.9(1:1) 34.7-207(1:4) 138-831(1:16)** | 37.42 For all sparsity | 59.53 For all sparsity | -- | 29.2 For all sparsity |
| | INT8 | **6.8-41.8(1:1) 27.4-167(1:4) 109-670(1:16)** | 164.3 For all sparsity | 84.1 For all sparsity | 57.7(1:1) 230.68(1:4) 922.7(1:16) | -- |

TFLOPS Calculation: IP Channels x OP Channels/Latency . *FP8 metrics are shown for E4M3.
&INT and E5M2 Multiplier area subtracted for accurate comparison. #FP8 Multiplier area subtracted.
$FoM = Energy efficiency/die area * # of 8-bit weights/byte

## References

[1] S. Liu *et al.*, "16.2 A 28nm 53.8TOPS/W 8b Sparse Transformer Accelerator with In-Memory Butterfly Zero Skipper for Unstructured-Pruned NN and CIM-Based Local-Attention-Reusable Engine," in *IEEE ISSCC*, 2023.

[2] A. Sridharan *et al.*, "SP-IMC: A Sparsity Aware In-Memory-Computing Macro in 28nm CMOS with Configurable Sparse Representation for Highly Sparse DNN Workloads," in *IEEE CICC*, 2024.

[3] Y. Yiyang *et al.*, "14.5 A 28nm 192.3TFLOPS/W Accurate/Approximate Dual-Mode-Transpose Digital 6T-SRAM CIM Macro for Floating-Point Edge Training and Inference," in *IEEE ISSCC*, 2025.

[4] Z. Yue *et al.*, "14.4 A 51.6TFLOPs/W Full-Datapath CIM Macro Approaching Sparsity Bound and $<2^{-30}$ Loss for Compound AI," in *IEEE ISSCC*, 2025.

[5] J. Saikia *et al.*, "FP-IMC: A 28nm All-Digital Configurable Floating-Point In-Memory Computing Macro," in *IEEE ESSCIRC*, 2023.

[6] B. Hickmann *et al.*, "Intel Nervana Neural Network Processor-T (NNP-T) Fused Floating Point Many-Term Dot Product," in *IEEE ARITH*, 2020.

[7] J.-S. Park *et al.*, "A Multi-Mode 8K-MAC HW-Utilization-Aware Neural Processing Unit with a Unified Multi-Precision Datapath in 4nm Flagship Mobile SoC," in *IEEE ISSCC*, 2022.