Zircon: A 16nm Heterogeneous CGRA SoC for Accelerating Dense and Sparse ML

Yuchen Mei*, Michael Oduoza*, Po-Han Chen, Maxwell Strange, Zhouhua Xie, Bo Wun Cheng, Kalhan Koul, Kartik Prabhu, Jeffrey Yu, Sai Gautham Ravipati, Mark Horowitz, and Priyanka Raina
*Dept. of Electrical Engineering, Stanford University, Stanford, CA, United States; * Equal Contribution*
Corresponding Author Email: {yuchenm, mcoduoza}@stanford.edu

***Abstract*—Machine learning accelerators face a growing mismatch between rapidly evolving models and fixed-function hardware. Dense ML ASICs achieve high efficiency on GEMMs but perform poorly on non-linear and sparse kernels, while homogeneous coarse-grained reconfigurable arrays (CGRAs) provide flexibility at the cost of compute density and utilization. We present Zircon, the first fabricated heterogeneous CGRA SoC that combines a specialized array, optimized for dense GEMMs and convolutions, with a flexible array designed for non-GEMM and sparse ML operations. Across dense and sparse ML workloads, Zircon achieves up to 34.5× and 33.4× better energy-delay product (EDP), respectively, than homogeneous CGRAs, while matching the EDP of fixed-function ML ASICs with substantially greater flexibility.**

***Index Terms*—Heterogeneous CGRA, machine learning acceleration, reconfigurable computing**

I. Introduction

Modern machine learning (ML) models, such as transformer-based language models, evolve rapidly as new training techniques, architectural motifs, and non-linear operators emerge. In contrast, fixed-function ML accelerators are optimized around a narrow set of dense kernels, primarily general matrix multiplication (GEMM) [1]–[4]. While this specialization enables high efficiency, non-GEMM components such as activation functions, normalization layers, attention mechanisms, transpose, and sparse tensor operations are often executed inefficiently or offloaded to a CPU, degrading end-to-end performance.

Stand-alone coarse-grained reconfigurable arrays (CGRAs) offer adaptability as algorithms change, but homogeneous CGRAs suffer from low compute density and routing congestion on large GEMMs, and poor spatial and temporal utilization on sparse and irregular workloads [6], [7]. These issues worsen as sparsity becomes increasingly prevalent in modern ML models. Beyond hardware limitations, flexible CGRAs also lack the software infrastructure needed to map the growing diversity of non-GEMM operators in a fusion-aware manner, making it difficult to track and support emerging workloads without significant manual re-engineering. Together, these trends expose a fundamental gap in the accelerator design space (Fig. 1): dense ML ASICs deliver efficiency without adaptability, homogeneous CGRAs provide flexibility without density, and reconfigurable accelerators broadly suffer from low utilization on sparse tasks and lack the full-stack support needed to deploy emerging non-GEMM ML operations.

We thank SRC PRISM Center, NSF 2238006, Intel HIP, Sloan, AHA, PORTAL, SystemX, and Samsung for funding, and Intel for chip fabrication.

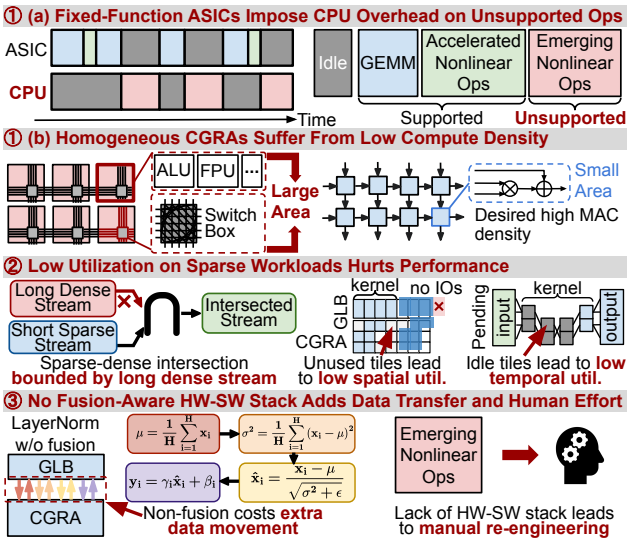


Fig. 1. *The ML accelerator design gap.* Zircon targets these previously unaddressed challenges, combining high efficiency with adaptability.

To address this gap, we present Zircon (Fig. 2), the first fabricated *heterogeneous* CGRA SoC. Zircon makes three contributions: (1) a heterogeneous CGRA architecture integrating a specialized array (SA) for dense GEMMs and convolutions with a flexible array (FA) for sparse tensor algebra and non-GEMM ML operations, enabling 14.2× and 1.36× higher multiply-accumulate (MAC) and memory density; (2) sparse acceleration hardware, including a locator primitive, sparse tile pipelining, and sparse tile unrolling, that improves the array utilization and sparse ML EDP by up to 9× and 33.4×; and (3) a hardware-software stack for mapping dense non-GEMM ML operators onto the FA, enabling composability, fusion, and forward support for emerging workloads, improving non-GEMM ML operator performance by up to 2.4×.

II. Heterogeneous SoC Architecture

Zircon's heterogeneous SoC architecture (Fig. 2) is designed to strike a balance between compute density and flexibility by integrating two arrays with complementary roles, enabling efficient acceleration across dense and sparse ML workloads. It consists of a flexible array (FA), a specialized array (SA),

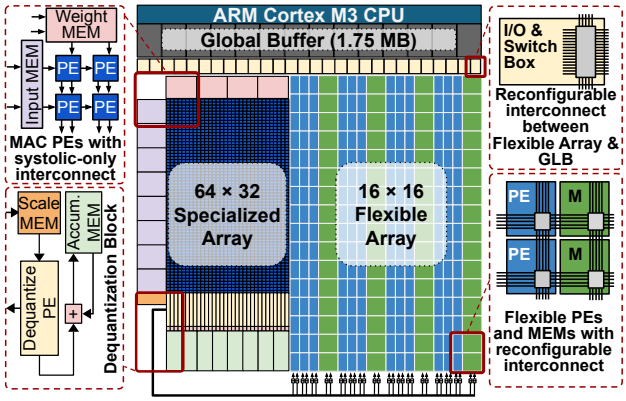


Fig. 2. *Zircon heterogeneous CGRA SoC.* Zircon integrates a specialized array (SA) for dense GEMMs, a flexible array (FA) for sparse and non-linear ML operations, a tiled global buffer (GLB), and an Arm Cortex-M3 controller.

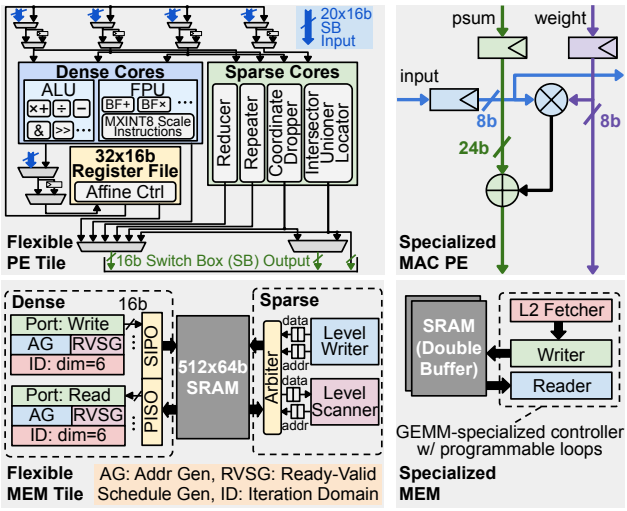


Fig. 3. *Tile-level flexibility vs. specialization.* By specializing FA components for dense GEMMs, the SA fits 2048 MAC PEs and 262 KB of SRAM in the same region the FA would fit 144 PEs and 192 KB of SRAM, with 14.2× and 1.36× higher MAC and memory density respectively.

a 1.75 MB global buffer (GLB), and an Arm Cortex-M3 CPU. The GLB has 14 tiles, each with 128KB of SRAM and dedicated load and store units.

The flexible array is a distributed CGRA composed of processing element (PE) and memory (MEM) tiles connected by a reconfigurable interconnect. Flexible PEs have dense cores supporting BF16 and INT16 arithmetic and sparse cores implementing the sparse abstract machine (SAM) [8] compute primitives. Flexible MEM tiles include 4 KB of SRAM, configurable address generators, and SAM storage primitives. These PEs and MEMs enable support for non-GEMM ML operators and sparse tensor algebra with reconfigurable dataflow.

The specialized array is derived from the FA and is specialized for GEMMs and convolutions with MAC-only PEs and a systolic interconnect, reducing PE area and enabling 14× higher MAC density. Similarly, specialized MEM

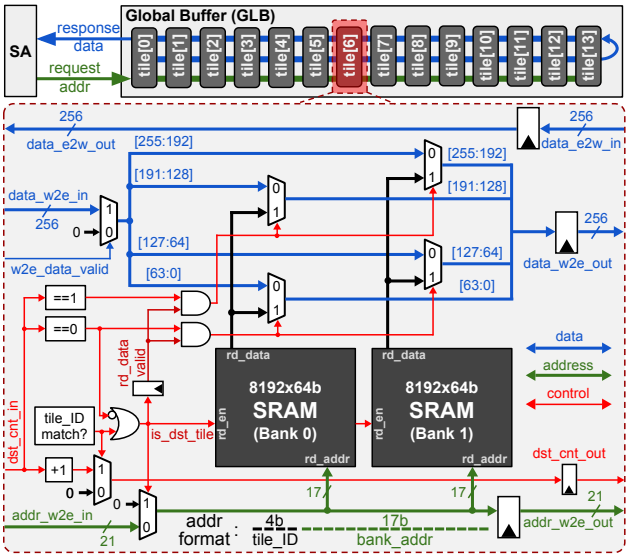


Fig. 4. *Flexible global memory access.* A ring interconnect allows the SA to access any GLB tile, avoiding the area and energy overheads of crossbars or global broadcasts. A 256b word is fetched from two neighboring GLB tiles.

tiles simplify address generation for GEMM access patterns, achieving 1.36× higher density. Fig. 3 shows the flexible and specialized PEs and MEMs. To balance inference energy efficiency and accuracy, the SA also supports MXINT8 microscaling through dedicated dequantization and accumulation PEs (Fig. 2, bottom-left).

Zircon enables low-overhead composition of GEMM and non-GEMM kernels by streaming SA outputs directly into the FA through FIFO-based ready-valid interfaces (Fig. 2, bottom), avoiding intermediate writes to global memory. Both arrays require flexible, placement-independent access to global memory. Zircon employs a ring interconnect between the SA and GLB, which gives the SA uniform-latency access to all GLB tiles (Fig. 4). Requests traverse through the ring until they reach the target tiles, which then send the data back to the SA along the rest of the loop. The FA can access any GLB tile via the configurable I/O switch boxes shown in Fig. 2. Zircon further increases FA-GLB bandwidth by aggregating data across multiple routing tracks, achieving 4× higher bandwidth than prior CGRAs [6], [7]. This prevents the SA from being back-pressured by the FA due to limited FA-GLB bandwidth and ensures rate matching between compute and memory across the heterogeneous SoC.

III. Sparse Acceleration Hardware Optimizations

Sparse-dense matrix multiplication (SpMM) is a key kernel in sparse ML networks such as graph convolutional networks (GCNs). In prior CGRAs, SpMM performance is constrained by dense-stream traversal during elementwise merge intersection [7]. However, the intersection is redundant for SpMM, as all elements in the sparse operand are present in the output, essentially making the intersection operation a no-op (Fig. 5, left). Zircon addresses this limitation with a new locator prim-

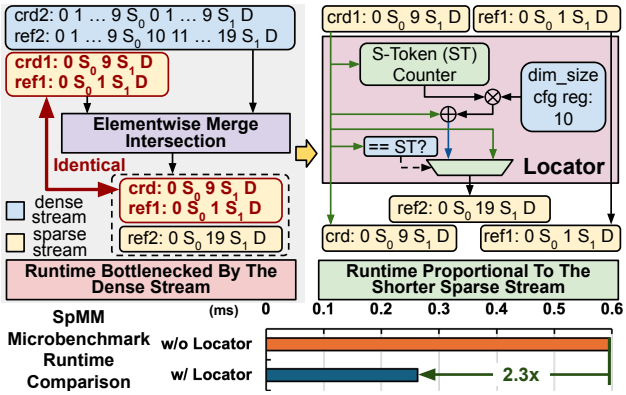


Fig. 5. *Locator primitive in the FA PE tile for sparse-dense intersection.* The locator enables iterate-locate style intersection, eliminating dense-stream bottlenecks in SpMM and yielding 2.3× speedup over prior CGRA approaches.

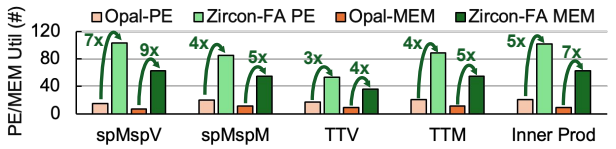


Fig. 6. *Improving sparse CGRA utilization.* Sparse tile unrolling increases array utilization by up to 9× compared to a state-of-the-art CGRA, Opal [7].

itive in the FA PE, which computes dense operand addresses using the sparse coordinates and the dense dimension. It iteratively offsets the address by the dense dimension when traversing a new sparse fiber, while forwarding the sparse coordinates and memory locations to the output (Fig. 5, right). This eliminates dense-stream traversal, yielding up to 2.3× speedup and driving most of the EDP improvement on GCN.

Sparse workloads also suffer from low utilization in prior CGRAs due to their inability to fully utilize global memory bandwidth and overlap global memory accesses with compute. Zircon integrates sparse tile pipelining and unrolling [9] to address temporal and spatial underutilization. Sparse tile pipelining uses double-buffered level storage in the FA MEM with decoupled reads and writes, so consecutive sparse tiles can overlap in time. Sparse tile unrolling replicates the compute graph across FA regions, time-multiplexes GLB input streams, and arbitrates out-of-order outputs into reserved GLB regions. Together, these two techniques improve FA utilization by up to 9× on sparse workloads (Fig. 6).

IV. Hardware-Software Stack for Dense Non-GEMM ML Operations

Previous CGRAs lack systematic support for dense non-GEMM ML operations, which typically leads to heavy manual effort in kernel implementation and poor support for complex operator schedules. Complex non-GEMM operators, such as LayerNorm and Softmax, are often executed as multiple separate passes in prior CGRAs and ASICs [1], [2], [6], increasing intermediate data movement and degrading performance.

Zircon introduces a hardware-software stack for dense non-GEMM ML operations based on a Halide frontend, which de-

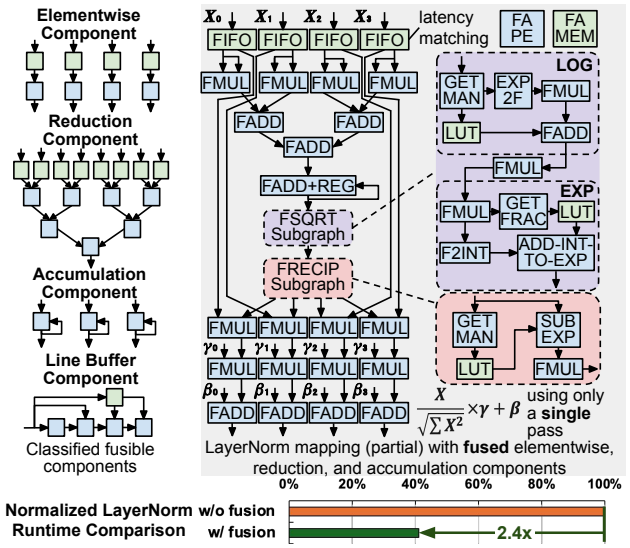


Fig. 7. *Composable mapping of non-GEMM ML operators.* The Halide-based compiler decomposes non-linear operators into fusible elementwise, reduction, accumulation, and line buffer components, enabling efficient parallel and pipelined execution of the operators on the FA.

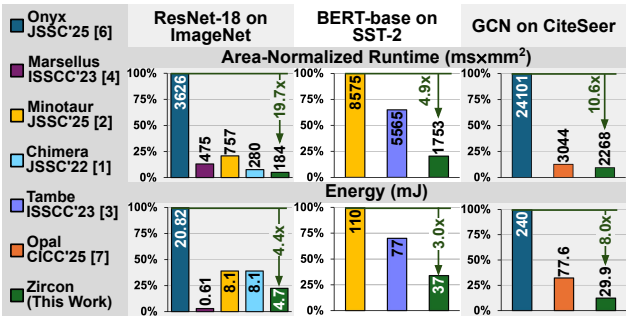


Fig. 8. *Application-level ML comparison.* Zircon achieves 34.5× lower EDP than a homogeneous CGRA [6] on dense ML models and up to 33.4× lower EDP on an end-to-end sparse ML network. Area is normalized by node with $area \times (16nm/node)^2$. Bars are normalized and annotated with data labels.

composes operators into reusable FA building blocks: elementwise, reduction, accumulation, and line buffer components (Fig. 7, left). The stack then groups the decomposed blocks into fusible patterns and executes them in a single pass. To enable such fusion, the stack generates FA MEM tile schedules that buffer intermediate results within the array, eliminating the need for intermediate writes to the GLB. The MEM tile schedules also enable delay matching to reduce stalls and increase throughput. Overall, our software stack enables fusion across non-GEMM operators, reducing data movement and significantly improving performance. Fig. 7 illustrates part of a LayerNorm mapping using the building blocks. It achieves a 2.4× runtime improvement over non-fused LayerNorm.

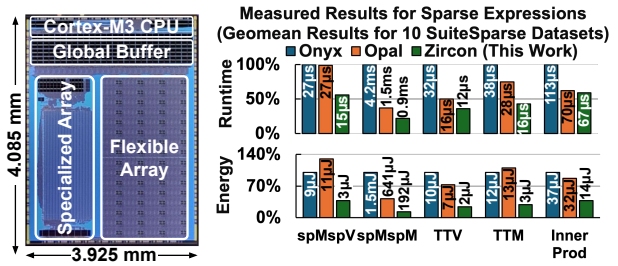
V. Results

Zircon is fabricated in Intel 16 (Fig. 9a). We evaluate it on a range of dense and sparse ML workloads. Fig. 8 and Table I report full-model results for ResNet-18, BERT-base,

TABLE I
COMPARISON WITH STATE-OF-THE-ART CGRAS AND ASICS.

			Minotaur JSSC'25 [2]	Tambe ISSCC'23 [3]	LUT-SSM ISSCC'26 [5]	Onyx JSSC'25 [6]	Opal CICC'25 [7]	**Zircon (This Work)**
Architecture			DNN ASIC	Transformer ASIC	LLM ASIC	SoC w/ CGRA	SoC w/ CGRA	SoC w/ CGRA
Node/Area			40 nm/65.6 mm^2	12 nm/4.6 mm^2	28 nm/16 mm^2	12 nm/23 mm^2	16 nm/16 mm^2	16 nm/16 mm^2
Voltage/Max Frequency			0.9 V/100 MHz	0.62-1.0 V/717 MHz	1.0 V/200 MHz	0.78 V/970 MHz	0.85 V/720 MHz	0.84-1.2 V/200 MHz
Data Precision			Posit8	FP4, FP8	FP16, INT1-4	INT16, BF16	INT16, BF16	INT16, BF16, MXINT8
TOPS			0.026-0.051	0.37-0.73	5.408	0.57	–	0.70⑤
TOPS/W			0.43-0.50	8.24-18.1	99.3	0.76	–	1.37⑤
Dense and Sparse ML Benchmarks	ResNet-18①	Runtime	72.1 ms	–	–	89 ms	–	**11.5 ms**
		ImageNet Accuracy	70.2%	–	–	68.9%	–	69.3%
	BERT-base②	Runtime	817 ms	682 ms	–	135.3 ms⑥	160.4 ms⑥	**109.5 ms**
		SST-2 Accuracy	91.9%	90.3%	–	–	–	90.2%
	Large Language Model	Prefill Runtime	–	–	Mamba-1.4B④ 74750 ms⑧	Llama3.2-1B③ 7870 ms⑥⑦	Llama3.2-1B③ 9332 ms⑥⑦	**Llama3.2-1B③ 2802 ms⑦**
		Accuracy and Loss	–	–	14.02 (+2.5) ⑨	–	–	**5.95 (+0.03)⑩**
	Sparse GCN	Runtime	–	–	–	589 ms	190 ms	**142 ms**

① ResNet-18 input size: $(3, 224, 224)$. ② BERT-base sequence length: 128. ③ Llama3.2-1B sequence length: 512. ④ Mamba-1.4B sequence length: 1024. ⑤ From a ResNet-18 downsampling (MXINT8) + quantization layer (BF16) @(0.85V, 170MHz). ⑥ Scaled from conv. Results include GEMMs only. ⑦ Excluding EMA. ⑧ Including EMA. ⑨ Perplexity on Wikitext-2. Baseline based on FP16 act. and weights. ⑩ Baseline based on BF16 act. and weights.



(a) (b)

Fig. 9. (a) *Zircon die photo.* (b) *Sparse kernel performance.* Sparse tile pipelining and unrolling improve sparse kernel performance by up to 1.8×, eliminating utilization collapse in prior CGRAs [6], [7]. Bars are normalized to Onyx, with data labels showing absolute values.

Llama3.2-1B, and GCN. In GCN, dense linear layers are mapped to the SA and sparse layers to the FA, aligning each layer with the appropriate array. Across these workloads, Zircon achieves up to 34.5× lower EDP than homogeneous CGRAs [6], [7]. Fig. 9b presents results for sparse expression benchmarks, featuring five tensor kernels commonly used in sparse applications. Tile pipelining and unrolling improve kernel performance by up to 1.8×, eliminating the utilization collapse observed in prior CGRAs. Table I compares Zircon against prior work. Relative to prior CGRAs, Zircon achieves the highest reported throughput and energy efficiency and delivers up to 7.7× lower runtime on ML models, at comparable accuracy. Compared with prior ASICs, Zircon shows up to 7.5× lower runtime, while maintaining similar task accuracy. Zircon demonstrates that heterogeneous CGRAs can deliver ASIC-class efficiency on dense ML while sustaining high utilization on sparse and non-GEMM workloads, providing a scalable path for future ML accelerators.

REFERENCES

[1] K. Prabhu, A. Gural, Z. F. Khan, R. M. Radway, M. Giordano, K. Koul, R. Doshi, J. W. Kustin, T. Liu, G. B. Lopes *et al.*, "CHIMERA: A 0.92-TOPS, 2.2-TOPS/W edge AI accelerator with 2-MByte on-chip foundry resistive RAM for efficient training and inference," *IEEE Journal of Solid-State Circuits*, vol. 57, no. 4, pp. 1013–1026, 2022.

[2] K. Prabhu, R. M. Radway, J. Yu, K. Bartolone, M. Giordano, F. Pedding-haus, Y. Urman, W.-S. Khwa, Y.-D. Chih, M.-F. Chang *et al.*, "Minotaur: A posit-based 0.42–0.50-TOPS/W edge transformer inference and training accelerator," *IEEE Journal of Solid-State Circuits*, 2025.

[3] T. Tambe, J. Zhang, C. Hooper, T. Jia, P. N. Whatmough, J. Zuckerman, M. C. Dos Santos, E. J. Loscalzo, D. Giri, K. Shepard *et al.*, "22.9 A 12nm 18.1 TFLOPs/W sparse transformer processor with entropy-based early exit, mixed-precision predication and fine-grained power management," in *ISSCC*. IEEE, 2023, pp. 342–344.

[4] F. Conti, D. Rossi, G. Paulin, A. Garofalo, A. Di Mauro, G. Rutishauer, G. marco Ottavi, M. Eggimann, H. Okuhara, V. Huard *et al.*, "22.1 A 12.4 TOPS/W@ 136GOPS AI-IoT system-on-chip with 16 RISC-V, 2-to-8b precision-scalable DNN acceleration and 30%-boost adaptive body biasing," in *ISSCC*. IEEE, 2023, pp. 21–23.

[5] S. Yoo, D. Kam, G. Park, S. Kwon, D. Lee, and Y. Lee, "31.7 LUT-SSM: A 99.3TFLOPS/W LUT-based state-space model accelerator using energy-efficient element-wise layer fusion and LUT-friendly weight-only quantization," in *ISSCC*. IEEE, 2026, pp. 544–545.

[6] K. Koul, O. Hsu, Y. Mei, S. G. Ravipati, M. Strange, J. Melchert, A. Carsello, T. Kong, P.-H. Chen, H. Ke *et al.*, "Onyx: A 12-nm programmable accelerator for dense and sparse applications," *IEEE Journal of Solid-State Circuits*, 2025.

[7] P.-H. Chen, B. W. Cheng, M. Oduoza, Z. Xie, K. Koul, S. G. Ravipati, Y. Mei, R. Lu, A. Carsello, M. Horowitz *et al.*, "Opal: A 16nm coarse-grained reconfigurable array for full sparse ML applications," in *CICC*. IEEE, 2025, pp. 1–3.

[8] O. Hsu, M. Strange, R. Sharma, J. Won, K. Olukotun, J. S. Emer, M. A. Horowitz, and F. Kjølstad, "The sparse abstract machine," in *Proceedings of the 28th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 3*, 2023, pp. 710–726.

[9] K. Koul, Z. Xie, M. Strange, S. G. Ravipati, B. W. Cheng, O. Hsu, P.-H. Chen, M. Horowitz, F. Kjolstad, and P. Raina, "Designing programmable accelerators for sparse tensor algebra," *IEEE Micro*, vol. 45, no. 3, pp. 58–65, 2025.