

# Breaking the Memory Wall: Performance Impacts of 333-eDRAM Integration in Tensor Accelerators

Rishi Purohit  
rishipur@mit.edu

Dandi Desta  
dandides@mit.edu

## I. INTRODUCTION

Transformer accelerators are increasingly used as more specialized, lower-latency alternatives to GPUs on specific workloads, but they are increasingly limited, especially from an energy perspective, by data movement bottlenecks. For long-context GPT-style workloads, large attention intermediates and model tensors can exceed on-chip SRAM capacity, forcing accesses to lower levels of the memory hierarchy and increasing energy substantially.

This project asks whether a new type of embedded DRAM — 333-eDRAM — can improve that bottleneck in a TPU v4-inspired accelerator. Rather than redesigning the entire accelerator, we focus on the memory hierarchy: specifically, whether replacing on-chip SRAM buffers with denser 333-eDRAM can reduce off-chip traffic enough to improve Energy-Delay Product (EDP).

The motivation is that SRAM is fast but area-expensive. Under a fixed area budget, its 6T cell limits how much capacity can be placed on chip. eDRAM is denser, but conventional designs suffer from high leakage and refresh overhead. 333-eDRAM is promising because it combines IGZO FETs for low leakage, CNFETs for strong read current, and silicon CMOS for peripheral logic, aiming to preserve density advantages while reducing the drawbacks that have limited prior eDRAM approaches [2].

Our central hypothesis is that 333-eDRAM is most useful when deployed selectively. In particular, we expect it to be most effective in the global buffer, where additional capacity can keep large reused tensors on chip, while SRAM remains better suited for latency-sensitive local buffers near the compute array.

To evaluate this idea, we modeled four memory configurations for a TPU-like accelerator: an all-SRAM baseline, local-buffer 333-eDRAM, global-buffer 333-eDRAM, and all-333-eDRAM. We analyzed these designs on GPT-style workloads in AccelForge, focusing on how energy and reuse behavior change with sequence length and einsum fusion. The key question is not whether 333-eDRAM should replace SRAM everywhere, but where its extra density provides the most value [11], [13]–[16].

Our results support three main takeaways. First, 333-eDRAM is most effective when used selectively in the global buffer rather than in the local buffer, because the global buffer benefits more from added capacity and tensor reuse

while the local buffer is more sensitive to peripheral overhead, showing that new memory technologies may be best used not as a substitute for but as a complement to existing technologies. Second, the largest gains appear at intermediate sequence lengths where SRAM begins to spill large attention intermediates such as  $QK$  and  $QK_{\text{softmax}}$ , but 333-eDRAM can still keep them on chip and preserve reuse, showing that workload size and structure is a key mediator of the performance gains. Third, einsum fusion amplifies the benefit of 333-eDRAM by allowing these intermediates to remain live in on-chip storage instead of being repeatedly materialized in main memory, showing that performance is quite sensitive to choice of mapping. Across all experiments, the dominant effect is on energy, which experiences significant reductions of up to 16.4%, rather than latency, which changes only marginally.

## II. BACKGROUND AND RELATED WORK

Prior work motivates this paper from three directions: accelerator memory bottlenecks, dense embedded-memory alternatives to SRAM, and emerging 3T/IGZO-based eDRAM designs. Our contribution is to connect these threads at the architecture level for a TPU-like accelerator running GPT-style workloads.

### A. Memory Hierarchies and Off-Chip Traffic

Modern machine learning accelerators depend heavily on data reuse, so memory-hierarchy design is often as important as raw compute throughput. TPU v4 is a natural baseline for this study because it is a production-scale accelerator whose efficiency still depends on keeping operands and intermediates close to the compute fabric [1]. More broadly, design-space studies have shown that limited on-chip storage can make off-chip DRAM a dominant energy cost; for example, Li et al. report that off-chip DRAM can account for a large fraction of system energy in DNN accelerators, motivating denser on-chip memory technologies that reduce main-memory traffic [5]. This motivation is especially strong for Transformer workloads, where attention intermediates grow rapidly with sequence length.

### B. SRAM, eDRAM, and Gain-Cell Memories

SRAM remains the standard on-chip buffer technology because it is fast and does not require refresh, but its 6T cell is area-expensive. eDRAM offers higher density, which can reduce off-chip accesses by increasing effective on-chip

capacity. RANA [3] showed both the promise and the difficulty of this approach: eDRAM-backed buffers can improve accelerator energy, but refresh and retention behavior must be handled carefully at the architecture level. Gain-cell eDRAM work further supports the density argument. Hwang and Kwon showed that a 3T eDRAM cell can occupy only 54% of the area of a comparable 6T SRAM cell, making 3T-class embedded memories attractive for capacity-constrained accelerator buffers [4]. At the same time, both RANA and later eDRAM-based compute-in-memory work emphasize that denser cells do not automatically translate into better systems; access cost, retention, and peripheral overhead remain important [3], [6].

Yu et al. [20] describe a 4T2C eDRAM cell designed for compute-in-memory (CiM) that can hold a ternary — rather than binary — cell state while using less area per cell than traditional SRAM. However, their results were generated using simulations of 65nm technology, which is quite a ways from state-of-the-art and may obscure issues of parasitic capacitance.

### C. IGZO-Based and Heterogeneous eDRAM Designs

Recent work has explored oxide-channel and hybrid-device memories to improve the retention and efficiency of dense embedded memories. IGZO-based eDRAM and CIM designs use the low leakage of oxide-channel devices to extend retention and reduce refresh pressure, making them attractive for back-end-of-line integration above logic [7], [8]. Other recent accelerator-oriented designs, including eDRAM digital CIM and Transformer accelerators built around 3T-eDRAM-class memories, show that these devices are becoming increasingly relevant for neural workloads [6], [9]. Within this space, 333-eDRAM is particularly interesting because it combines IGZO for retention, CNFETs for read performance, and silicon CMOS for peripheral logic in a heterogeneous monolithic 3D design. Kong et al. showed that this structure improves device-level EDP and EADP relative to silicon eDRAM, suggesting that it is a strong candidate for dense on-chip storage [2].

### D. Relation to This Work

Our work differs from prior eDRAM and CIM papers in two ways. First, we study 333-eDRAM as a *memory-hierarchy* technology rather than as a standalone bit cell or a compute-in-memory macro. We preserve a TPU-like datapath and ask where this denser memory should be placed: in latency-critical local buffers, larger global buffers, or both. Second, we target GPT-style attention workloads, where the main question is whether increased on-chip capacity preserves reuse of large intermediates and avoids spill traffic as sequence length grows. In this sense, our work is closest in spirit to prior studies showing that memory choice should depend on data lifetime, access frequency, and hierarchy level rather than on a universal SRAM replacement strategy [5], [10]. The specific contribution here is to evaluate whether the device-level advantages of 333-eDRAM translate into system-level gains for Transformer acceleration, and under what placement and workload conditions those gains appear.

## III. MOTIVATION

Our project studies whether replacing on-chip SRAM with 333-eDRAM improves the efficiency of a TPU-like accelerator on transformer attention workloads. The core motivation is that attention moves large intermediate tensors through the memory hierarchy, so performance depends not only on compute throughput but also on on-chip capacity, access cost, and data movement energy. Because 333-eDRAM is denser than SRAM while remaining on chip, it is a natural candidate for storing larger activations and intermediates close to the MAC array.

By way of exposition, Table I summarizes the memory characteristics used for a 112 mm<sup>2</sup> global-buffer comparison. The key difference is that 333-eDRAM provides much higher capacity in the same footprint, while keeping read and write costs in a similar range. However, it is important to note that read and write latencies and energies *are* higher for the 333-eDRAM cache, so there is a clear tradeoff here. Nevertheless, the increased capacity has significant benefits in terms of potential for tensor reuse.

We study this question using an AccelForge model of a TPU-style accelerator running a GPT-3-like attention layer without KV caching [11], [13]–[16]. This workload stresses the on-chip memory system because it materializes large tensors such as  $QK$  and repeatedly moves intermediates through the local and global buffers as sequence length increases.

Architecturally, we focus on the two main on-chip memory structures: the local buffer and the global buffer. We replace SRAM with 333-eDRAM in one or both of these structures while keeping the rest of the architecture fixed, which isolates the effect of memory technology from unrelated mapping changes. Our goal is to determine where added memory density matters most, and what tradeoff it creates in energy and latency.

Our hypothesis is that 333-eDRAM helps primarily by increasing effective on-chip capacity. If more of the working set remains resident in the global buffer, large intermediates can be reused instead of spilled to main memory. By contrast, the local buffer is more latency-sensitive and may benefit less because peripheral overhead matters more there.

To evaluate this idea, we built a HWComponents model of 333-eDRAM using latency, energy, and area data from the ArrayCharacterization flow in MemSysExplorer [12], [13], [17]. Because this characterization flow builds on DESTINY-style memory modeling, we cite DESTINY as the underlying memory-array modeling infrastructure for SRAM, eDRAM, and emerging memory technologies [18], [19]. We then modified an AccelForge TPU v4-style architecture specification by replacing the normally SRAM-based buffers with denser 333-eDRAM and measuring the resulting performance on GPT-style attention workloads [11], [13]–[16].

## IV. EXPERIMENT SETUP

We evaluate the architecture with three experiments, all run in AccelForge on a GPT-3-like attention workload without KV caching [11], [13]–[16]. Across all experiments, the compute datapath and overall organization remain fixed; the main

TABLE I  
MEMORY CHARACTERISTICS FOR A 112 MM<sup>2</sup> GLOBAL BUFFER.

| Metric            | SRAM                   | 333-eDRAM              |
|-------------------|------------------------|------------------------|
| Read latency (s)  | $7.81 \times 10^{-12}$ | $4.15 \times 10^{-11}$ |
| Read energy (J)   | $2.41 \times 10^{-10}$ | $3.70 \times 10^{-10}$ |
| Write latency (s) | $1.56 \times 10^{-11}$ | $4.05 \times 10^{-11}$ |
| Write energy (J)  | $3.02 \times 10^{-10}$ | $4.00 \times 10^{-10}$ |
| Leak power (W)    | 5.60                   | 1.74                   |
| Capacity          | 128 MB                 | 512 MB                 |

variable is whether the on-chip buffers use SRAM or 333-eDRAM under the same area budget.

#### A. Experiment 1: Buffer Replacement Study

We first compare four configurations: an all-SRAM baseline, local-buffer-only 333-eDRAM, global-buffer-only 333-eDRAM, and all-333-eDRAM. This experiment isolates which storage level benefits most from the new memory technology.

We set up this experiment by using the existing TPU v4 architecture definition available through AccelForge [11], [13]–[16], but modified it to include leakage power for the local and global buffers, estimated from a 50 mW/mm<sup>2</sup> baseline for 6T SRAM. We then calculated, for each of the local and global buffers, the capacity of 333-eDRAM that would yield an area the same as the original SRAM buffers. We obtained values of 512MB and 32MB, respectively (rounding to the nearest power of two and not going over; in any case the areas were quite close to the originals).

#### B. Experiment 2: Sequence-Length Scaling and Capacity Cut-offs

We next scale sequence length to stress on-chip capacity. This experiment identifies when large quadratically scaling intermediates, especially  $QK$  and  $QK_{\text{softmax}}$ , stop fitting in the global buffer for SRAM and for 333-eDRAM. The goal is to pinpoint the regime where extra density changes reuse behavior and reduces spill traffic.

This involved changing the number of tokens in the Accelforge workload definition for GPT 3 without KV caching. We only focus on the control and the 333 eDRAM global buffer, consistent with the results of Experiment 1.

#### C. Experiment 3: Effect of Einsum Fusion

Finally, we study the interaction between memory technology and einsum fusion. With fusion enabled, intermediates can remain in on-chip buffers instead of being explicitly materialized in main memory. Comparing fused and unfused mappings for SRAM and 333-eDRAM shows whether added capacity amplifies the benefit of fusion.

We simulated lack of fusion by using the TensorHolder facility within the AccelForge TPU definition and used the AccelForge mapping infrastructure to compare fused and unfused schedules [11], [14]–[16]. We specified that the AV, Z, V, K, and Q tensors (the intermediates) should be kept within main memory using the `keep` flag on TPU MainMemory.

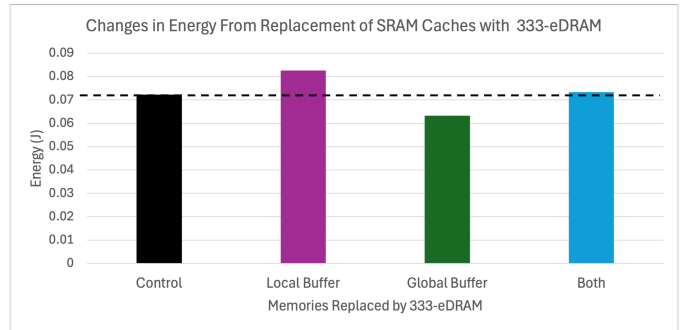


Fig. 1. Energy performance on the GPT 3 (no KV) workload on a TPU with the local buffer, global buffer, both, and neither replaced by 333-eDRAM.

## V. RESULTS AND DISCUSSION

The results across all three experiments point to a fairly consistent conclusion: for this workload, 333-eDRAM is most useful when it is used to expand the *global buffer*, not when it is used uniformly throughout the memory hierarchy. The main effect is a change in data movement energy accruing from changes in data reuse behavior rather than latency changes. In particular, the denser 333-eDRAM global buffer allows large attention intermediates to stay on chip over a wider range of sequence lengths, which reduces accesses to main memory and improves overall energy efficiency.

We broadly interpret our results through the lens of reuse. In attention, tensors such as  $QK$  and  $QK_{\text{softmax}}$  are expensive not only because they are large, but because they are referenced repeatedly by subsequent operations. If those tensors fit in the global buffer, the accelerator can exploit temporal reuse by serving many accesses from on-chip storage. If they do not fit, the mapping must either spill them to main memory or recompute/fetch them in smaller pieces, both of which increase energy. The benefit of 333-eDRAM therefore comes less from making any single access dramatically cheaper and more from changing from where repeated accesses are serviced.

One striking result is that latency hardly changes even when energy does; all changes are below 3%. The data show that the local and global memories are not on the critical path often enough for their modest access-time differences to translate into large end-to-end runtime changes; the MAC dominates latency because the accelerator overlaps most buffer accesses with ongoing tensor computation.

#### A. Experiment 1: Buffer Replacement Study

Experiment 1 compares four memory configurations: the all-SRAM baseline, local-buffer-only 333-eDRAM, global-buffer-only 333-eDRAM, and an all-333-eDRAM design. Results are shown in Fig. 1. The main takeaway is consistent with our hypothesis: the global buffer is the right place to deploy 333-eDRAM. Replacing the global buffer yields the best improvement in energy, with a 12.5% decrease (10.5% EDP) relative to the all-SRAM baseline, while replacing only the local buffer actually harms energy, resulting in a 14.3% *increase* in both energy and EDP relative to the control.

This difference follows directly from the role of each buffer in the accelerator. The global buffer is large, shared across more of the computation, and responsible for holding tensors that are reused over longer windows. That makes density especially valuable there: increasing capacity increases the chance that large intermediates remain resident long enough to be consumed by subsequent einsums without returning to main memory. As a result, the global buffer is where 333-eDRAM can translate extra bits per unit area into materially higher reuse.

On the other hand, the local buffer is much smaller and sits closer to the compute array, where low access overhead matters more than raw density. Because it already holds only relatively small working sets, increasing its capacity does not unlock the same level of additional reuse. At the same time, 333-eDRAM incurs nontrivial peripheral overhead. ArrayCharacterization results from the MemSysExplorer/DESTINY-based modeling flow indicate that a meaningful fraction of the access cost comes not from the storage cell itself but from the surrounding peripheral circuitry, including sensing and interface logic [17]–[19]. Furthermore, at a per-cell level, it is true that 333-eDRAM has both higher latency and energy, so if the density gains are not as important, the calculus does not work out in 333-eDRAM’s favor.

This helps explain why a hybrid design emerges as the best architectural choice. SRAM remains well matched to latency-critical local storage, while 333-eDRAM is better used where capacity and reuse dominate.

### B. Experiment 2: Sequence-Length Scaling and Capacity Cutoffs

Experiment 2 is the central result of the paper, and the results are shown in Fig. 2. As sequence length increases, the energy benefit of the 333-eDRAM global buffer is not uniform; instead, it is concentrated over a specific range of sequence lengths where on-chip capacity is the deciding factor. This behavior is exactly what we would expect for attention. The key intermediates, especially  $QK$  and  $QK_{\text{softmax}}$ , scale quadratically with sequence length and are accessed frequently enough that keeping them in the global buffer provides substantial reuse. When they fit on chip, repeated accesses are cheap; when they no longer fit, the mapping must spill them and the energy cost rises sharply.

The figure shows two important cutoff points. The first is a lower spill point, visible in the inset, where the SRAM global buffer can no longer hold the full  $QK$  and  $QK_{\text{softmax}}$  tensors. That transition occurs at a input sequence length (number of tokens) between 128 and 136. Beyond that point, the SRAM loses the ability to keep those quadratically growing tensors resident in the global buffer, so reuse drops and memory traffic increases. The second cutoff appears later in the main figure, between 1280 and 1536 tokens, where even the denser 333-eDRAM global buffer eventually runs out of space.

The interval between the SRAM spillover cutoff and the 333-eDRAM spillover cutoff is the most interesting regime in the experiment because it isolates the value of increased

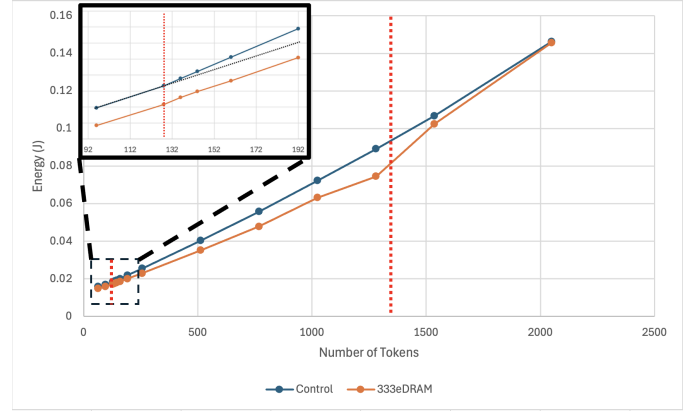


Fig. 2. Energy performance with and without a 333-eDRAM global buffer as sequence length increases. The inset highlights the lower spill point where the SRAM global buffer can no longer retain the quadratically scaling  $QK$  and  $QK_{\text{softmax}}$  intermediates, while the main plot shows the higher-sequence-length cutoff for the 333-eDRAM buffer. The dashed line is a guide showing the linear trend to highlight divergence.

on-chip density. In this middle region, the 333-eDRAM design achieves a substantial energy advantage of up to 16.4% (seq. len 1280). 333-eDRAM is still able to keep the large attention intermediates on chip, while SRAM is not. Once SRAM begins spilling, every subsequent reuse opportunity for those tensors becomes more expensive. The 333-eDRAM design, by contrast, continues to serve those accesses from the global buffer, preserving reuse and avoiding many lower-level transfers. This is exactly the kind of workload regime where denser on-chip memory matters most.

The mapping files make this story even clearer. Comparing the mappings in Fig. 3, we see that at the representative intermediate sequence length of 1024, the 333-eDRAM configuration keeps both  $QK$  and  $QK_{\text{softmax}}$  in the global buffer long enough to support reuse across the downstream stages of attention. The SRAM mapping cannot do that. Instead, because its global buffer is smaller at the same area budget, those tensors must be partitioned or spilled, which breaks the clean reuse pattern and forces additional movement through the hierarchy.

Leakage power also contributes to this result. Because 333-eDRAM uses a 3T cell rather than the 6T cell minimum typical of SRAM, it benefits from lower leak power of 1.74 W rather than 5.60 W (estimated based on a 50 mW/mm<sup>2</sup> baseline for 6T SRAM). That said, leakage is better understood as a supporting factor rather than the entire explanation. The dominant mechanism is still reuse: lower leakage helps, but the major win comes from the fact that denser storage changes the point at which large reused tensors stop fitting on chip.

### C. Experiment 3: Effect of Einsum Fusion

Experiment 3 studies the interaction between memory technology and einsum fusion. Fusion is especially relevant here because it directly changes whether intermediate tensors are materialized in memory at all. When adjacent einsum-style operations are fused, the producer and consumer can often

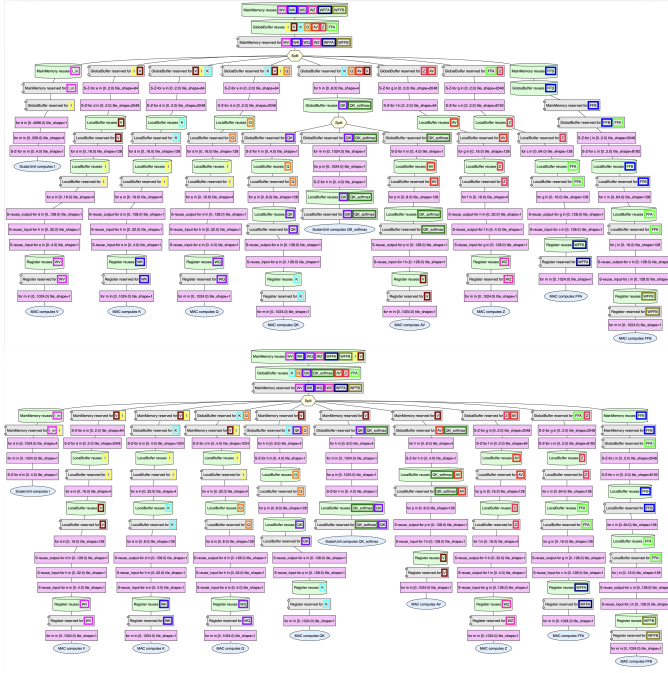


Fig. 3. Mappings for Experiment 2 for control (above) and 333-eDRAM global buffer (below) architectures. The 333-eDRAM global buffer, by being significantly more dense (though equal area) is able to reuse the  $QK$  and  $QK_{softmax}$  tensors while the control spills them out to main memory.

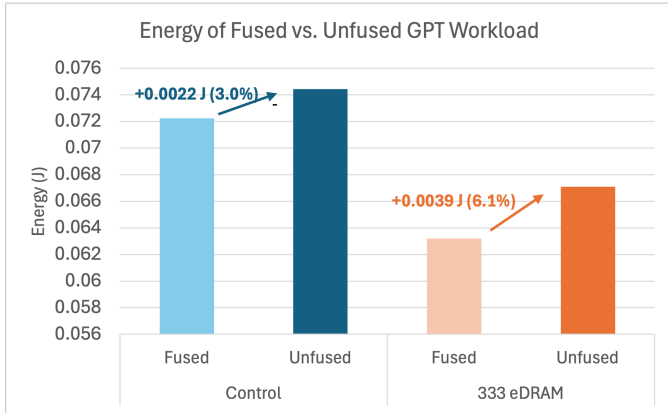


Fig. 4. Energy performance for a TPU running a GPT3-style workload with fusion and no fusion. Fusion is defined as forcing the AV, Z, V, K, and Q tensors to reside in main memory, where these tensors are defined as they are in the standard AccellForge GPT3 workload.

be scheduled so that intermediate results remain in on-chip buffers and are consumed soon after they are produced. In the best case, this avoids writing large temporaries out to main memory entirely. When fusion is disabled, those same intermediates are explicitly materialized, stored, and later reloaded, increasing both memory traffic and energy.

The results, shown in Fig. 4, highlight that fusion helps both SRAM and 333-eDRAM, but the benefit is especially pronounced for the 333-eDRAM configuration, with a relative improvement of 6.1% for 333-eDRAM compared with 3.0% for SRAM. This difference has an intuitive explanation. Fusion

is most valuable when the architecture has enough on-chip capacity to hold the produced intermediates until they are consumed. Since 333-eDRAM enlarges the global buffer, it creates exactly the conditions under which fusion can pay off. Said differently, fusion creates the opportunity to avoid repeated pulls of intermediate data from main memory, and 333-eDRAM provides the storage density and thus capacity needed to capitalize on that opportunity.

This is also why disabling fusion hurts the 333-eDRAM design more in relative terms. Without fusion, the architecture is forced to commit intermediates to memory explicitly, which removes one of the biggest advantages of a larger global buffer. The 333-eDRAM design still retains some benefit from lower leakage and higher density, but it can no longer fully exploit the “keep the tensor live on chip” behavior that makes it so effective in the fused case. The SRAM baseline also suffers when fusion is removed, but because its capacity was already limiting reuse, it has less to lose.

The results indicate that a denser memory hierarchy is most useful when the mapping and fusion strategy actually expose reuse opportunities. Conversely, mapping with fusion is most valuable when the hardware can hold onto the fused intermediates long enough for the consumer to use them. In our experiments, the strongest outcomes occur when those two pieces align: 333-eDRAM increases effective global-buffer capacity, and fusion ensures that this capacity is used to retain intermediates rather than repeatedly materializing them from main memory.

#### D. Increasing 333-eDRAM Port Width

We also explored increasing the port width in the hope of improving bandwidth further. We intuited that if the global buffer could deliver more data per access, perhaps it would better support the bandwidth demands of attention and even make it more attractive to place some otherwise stationary weight tensors in 333-eDRAM-backed storage. However, this modification did not improve performance.

The ArrayCharacterization output from the MemSysExplorer/DESTINY-based modeling flow (not shown due to space) suggests why [17]–[19]. Increasing port width raises the cost of the peripheral circuitry substantially, and that extra peripheral energy offsets the gain from the higher bandwidth.

#### E. Summary

Overall, the results across our experiments support a fairly specific but generally applicable conclusion. 333-eDRAM is most effective when it is used selectively in the global buffer of a TPU-like accelerator running memory-intensive attention workloads. Its benefit comes primarily from increasing on-chip capacity and preserving reuse of large intermediates, not from dramatically reducing latency. That is why the strongest gains appear in the sequence-length range where SRAM begins to spill but 333-eDRAM does not, and why fusion magnifies the advantage by allowing those same intermediates to remain live on chip rather than being materialized repeatedly.

## VI. LLM USAGE

We used Gemini for helping us set up the libraries at the beginning; we ran into issues with package managers, etc. as well as forking the repo and getting the commits to work when repos are embedded within each other. We also used Gemini to help proofread and for assistance with converting the print statements from the code into a structured table for analysis.

We used Claude for making syntactic edits to config files; i.e., there were many config files we had to change so we gave the base file to Claude and described the changes, and it created duplicates populated with the correct info. We used Gemini to find typos in the final report.

AI was not at all helpful for anything technically oriented because it seems as though it still doesn't know much about what goes into optimizing an accelerator. Though it has a vague understanding of reuse, mapping, and fusion, it doesn't hold up to any rigor. For this reason, we didn't use AI for any of the Python coding aspects.

## VII. CODE AVAILABILITY

Our repo is available at <https://github.com/rishipurohit/hwcomponents-333e>. The relevant notebook is `component_energy_area.ipynb`.

## REFERENCES

- [1] N. Jouppi, G. Kurian, S. Li, P. Ma, R. Nagarajan, L. Nai, N. Patil, S. Subramanian, A. Swing, B. Towles, C. Young, X. Zhou, Z. Zhou, and D. A. Patterson, "TPU v4: An Optically Reconfigurable Supercomputer for Machine Learning with Hardware Support for Embeddings," in *Proceedings of the 50th Annual International Symposium on Computer Architecture (ISCA)*, 2023, Article 82, pp. 1–14, doi: 10.1145/3579371.3589350.
- [2] D. Kong, S. Prakash, J. Kufel, G. Kyriazidis, Y. Omri, D. Verity, E. Ozer, V. Janapa Reddi, and G. Hills, "333-eDRAM: 3T Embedded DRAM Leveraging Monolithic 3D Integration of 3 Transistor Types: IGZO, Carbon Nanotube and Silicon FETs," in *Proceedings of the 62nd ACM/IEEE Design Automation Conference (DAC)*, 2025, doi: 10.1109/DAC63849.2025.11132950.
- [3] F. Tu, W. Wu, S. Yin, L. Liu, and S. Wei, "RANA: Towards Efficient Neural Acceleration with Refresh-Optimized Embedded DRAM," in *Proceedings of the 45th Annual International Symposium on Computer Architecture (ISCA)*, 2018, pp. 340–352, doi: 10.1109/ISCA.2018.00037.
- [4] M.-E. Hwang and S. Kwon, "A 0.94  $\mu$ W 611 KHz In-Situ Logic Operation in Embedded DRAM Memory Arrays in 90 nm CMOS," *Electronics*, vol. 8, no. 8, Article 865, Aug. 2019, doi: 10.3390/electronics8080865.
- [5] H. Li, M. Bhargava, P. N. Whatmough, and H.-S. P. Wong, "On-Chip Memory Technology Design Space Explorations for Mobile Deep Neural Network Accelerators," in *Proceedings of the 56th ACM/IEEE Design Automation Conference (DAC)*, 2019, pp. 1–6, doi: 10.1145/3316781.3317874.
- [6] I. Lee, J. Lee, J. Jang, and J.-J. Kim, "An eDRAM Digital In-Memory Neural Network Accelerator for High-Throughput and Extended Data Retention Time," in *Proceedings of the Design, Automation & Test in Europe Conference (DATE)*, 2025.
- [7] S. R. S. Raman, S. Xie, and J. P. Kulkarni, "IGZO CIM: Enabling In-Memory Computations Using Multilevel Capacitorless Indium-Gallium-Zinc-Oxide-Based Embedded DRAM Technology," *IEEE Journal on Exploratory Solid-State Computational Devices and Circuits*, vol. 8, no. 1, pp. 35–44, 2022, doi: 10.1109/JXCDC.2022.3188366.
- [8] M. Kim, Y. Kim, and J.-J. Kim, "Compute-in-Memory Array Design Using Stacked Hybrid IGZO/Si eDRAM Cells," in *Proceedings of the Design, Automation & Test in Europe Conference (DATE)*, 2025.
- [9] D. Kim, I.-J. Jung, G. Ko, S. Ham, C. D. Manh, G. Yang, and J.-Y. Kim, "D3TA: 38.9 TOPS/W Transformer Accelerator with Dual-Port 3T-eDRAM Digital Compute-In-Memory using HyperAttention and Triple-Sparsity-Handling," in *Proceedings of the IEEE European Solid-State Electronics Research Conference (ESSERC)*, 2025, doi: 10.1109/ESSERC66193.2025.11214050.
- [10] Y. Luo, Y.-C. Luo, and S. Yu, "A FeRAM Based Volatile/Non-volatile Dual-mode Buffer Memory for Deep Neural Network Training."
- [11] T. Andrulis and M. Gilbert, "AccelForge," software repository, MIT License. [Online]. Available: <https://github.com/Accelergy-Project/accelforge>. Accessed: May 1, 2026.
- [12] T. Andrulis, "HWComponents," software repository, MIT License. [Online]. Available: <https://github.com/Accelergy-Project/hwcomponents>. Accessed: May 1, 2026.
- [13] T. Andrulis, J. S. Emer, and V. Sze, "CiMLoop: A Flexible, Accurate, and Fast Compute-In-Memory Modeling Tool," in *Proc. IEEE Int. Symp. Performance Analysis of Systems and Software (ISPASS)*, 2024, pp. 10–23, doi: 10.1109/ISPASS61541.2024.00012.
- [14] T. Andrulis, M. Gilbert, V. Sze, and J. S. Emer, "Fast and Fusiect: An Optimal Fusion-Aware Mapper for Accelerator Modeling and Evaluation," arXiv:2602.15166, 2026. [Online]. Available: <https://arxiv.org/abs/2602.15166>.
- [15] M. Gilbert, Y. N. Wu, A. Parashar, V. Sze, and J. S. Emer, "Loop-Tree: Enabling Exploration of Fused-layer Dataflow Accelerators," in *Proc. IEEE Int. Symp. Performance Analysis of Systems and Software (ISPASS)*, 2023, pp. 316–318, doi: 10.1109/ISPASS57527.2023.00038.
- [16] M. Gilbert, T. Andrulis, V. Sze, and J. S. Emer, "The Turbo-Charged Mapper: Fast and Optimal Mapping for Accelerator Modeling and Evaluation," arXiv:2602.15172, 2026. [Online]. Available: <https://arxiv.org/abs/2602.15172>.
- [17] MemSysExplorer Project, "MemSysExplorer," Tufts University, Harvard University, and Amherst College. [Online]. Available: <https://msx.ece.tufts.edu/>. Accessed: May 1, 2026.
- [18] M. Poremba, S. Mittal, D. Li, J. S. Vetter, and Y. Xie, "DESTINY: A Tool for Modeling Emerging 3D NVM and eDRAM Caches," in *Proc. Design, Automation & Test in Europe Conference and Exhibition (DATE)*, 2015, pp. 1543–1546, doi: 10.7873/DATE.2015.0733.
- [19] S. Mittal, R. Wang, and J. S. Vetter, "DESTINY: A Comprehensive Tool with 3D and Multi-Level Cell Memory Modeling Capability," *Journal of Low Power Electronics and Applications*, vol. 7, no. 3, Article 23, 2017, doi: 10.3390/jlpea7030023.
- [20] C. Yu, T. Yoo, H. Kim, T. T. -H. Kim, K. C. T. Chuan and B. Kim, "A Logic-Compatible eDRAM Compute-In-Memory With Embedded ADCs for Processing Neural Networks," in *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 68, no. 2, pp. 667–679, Feb. 2021, doi: 10.1109/TCSI.2020.3036209.