Tri-Stack-PIM: A Heterogeneous Multi-Die 3D-Integrated Processing-in-Memory Architecture for Efficient LLM Inference

Shuao Jia
Beijing University of Posts and Telecommunications
Beijing, China
pdxqjiashuao@bupt.edu.cn

Zichao Ling
Beijing University of Posts and Telecommunications
Beijing, China
lingzichao@bupt.edu.cn

Chen Bai
Fudan University
Shanghai, China
cbai@fudan.edu.cn

Kang Zhao
Beijing University of Posts and Telecommunications
Beijing, China
zhaokang@bupt.edu.cn

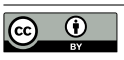
Jianwang Zhai*
Beijing University of Posts and Telecommunications
Beijing, China
zhaijw@bupt.edu.cn

Abstract

Large language model (LLM) inference during autoregressive decoding is fundamentally memory-bandwidth-bound. Existing three-dimensional (3D) stacked processing-in-memory (PIM) architectures alleviate the off-chip bandwidth wall, yet remain constrained by several limitations: physical layer (PHY), controllers, and direct memory access (DMA) engines compete with arithmetic units for advanced-node base-die area; hybrid-bonding keep-out zones fragment the base-die floorplan and impede physical design; compute tiles built in a dynamic random access memory (DRAM) process are restricted to low logic density, capping achievable throughput; and nonlinear operations offloaded to the host reintroduce a communication bottleneck.

We present Tri-Stack-PIM, a heterogeneous three-die architecture that resolves these tensions by dedicating a top DRAM die to storage, a middle buffer die on a mature process node to PHYs, DMA engines, and near-memory compute units, and a bottom base die on an advanced node exclusively to single-instruction-multiple-data (SIMD) multiply-accumulate (MAC) arrays. We further introduce a Unified Buffer-Aware Ready-based (UBAR) hardware out-of-order micro-operation scheduler that enforces cross-die dependency correctness at runtime while maximizing issue-slot utilization through a multi-term priority function, recovering up to 90% of the cycles that an in-order policy would forfeit. A joint architecture-dataflow co-exploration identifies Pareto-optimal configurations that reduce the energy-delay product (EDP) by 33-66× over naïve parameter selection. Evaluated across multi-head, grouped-query, and multi-query attention (MHA, GQA, MQA) mechanisms with batch sizes 1-32, Tri-Stack-PIM achieves geometric-mean speedups of 4.95×, 2.57×, and 1.68× across all three attention mechanisms and batch sizes over a graphics processing unit (GPU) baseline, AttAcc, and H^2-LLM, respectively.

*Corresponding author. This work is supported by the National Natural Science Foundation of China (No. 62404021), the Beijing Natural Science Foundation (No. L2603021), and the Fundamental Research Funds for the BUPT (No. 2025AI4S20).



This work is licensed under a Creative Commons Attribution 4.0 International License.
ICCAD '26, San Jose, CA, USA
© 2026 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-2873-0/2026/11
https://doi.org/10.1145/3831252.3834022

CCS Concepts

• **Hardware → 3D integrated circuits**.

Keywords

3D Integration, System-Technology Co-Optimization

ACM Reference Format:
Shuao Jia, Zichao Ling, Chen Bai, Kang Zhao, and Jianwang Zhai. 2026. Tri-Stack-PIM: A Heterogeneous Multi-Die 3D-Integrated Processing-in-Memory Architecture for Efficient LLM Inference. In *IEEE/ACM International Conference on Computer-Aided Design (ICCAD '26), November 08–12, 2026, San Jose, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3831252.3834022

1 Introduction

LLMs are now deployed across a wide range of applications, such as conversational assistants, code generation, document summarization, retrieval-augmented question answering, and agentic workflows. During autoregressive decoding, generating each token requires fetching the full set of model weights along with a key-value (KV) cache that grows with sequence length; as a result, model throughput is mainly bound by memory bandwidth.

3D-stacked PIM is one promising solution to this bottleneck, and two complementary hardware trends make it practical. First, near-memory and in-memory computing move bandwidth-sensitive computation adjacent to—or directly inside—the DRAM banks, eliminating off-chip traffic and exploiting the abundant internal bandwidth of the memory array [12, 16, 19–21, 24, 32]. Second, hybrid bonding (HB) provides dense, low-parasitic die-to-die interconnects that deliver far higher inter-die bandwidth and lower transfer energy than conventional microbump-based stacking, allowing logic and memory dies to be integrated at fine pitch [5, 25, 40, 41]. Together, these two advances enable practical 3D-stacked architectures for LLM decoding.

Prior LLM-oriented PIM systems have explored this direction along several paradigms. First, DRAM-process PIM [20, 32] places compute logic directly inside the DRAM array. This maximizes proximity to the data, but is fundamentally constrained by the low logic density achievable in a DRAM process, which limits the arithmetic capability that can be integrated. Second, microbump-based 3D-stacked designs [16] typically retain operations other than general matrix multiplication (GEMM) on the host processor. This reintroduces host-device data movement, so performance is again bottlenecked by the

Table 1: Comparison of existing designs and our Tri-Stack-PIM. Columns indicate support for advanced-node compute logic (Adv.-Node Compute), in-stack softmax execution (In-Stack Softmax), compute/interface physical decoupling (C/I Decoupling), and hardware-dynamic micro-operation scheduling (Dyn. Sched.).

Work	Adv.-Node Compute	In-Stack Softmax	C/I Decoupling	Dyn. Sched.
DRAM-process PIM				
TransPIM [47]	✗	✓	✗	✗
AttAcc [32]	✗	✓	✗	✗
NeuPIMs [16]	✗	✗	✗	✗
IANUS [34]	✗	✗	✗	✗
SpecPIM [23]	✗	✗	✗	✓
Logic-die / HB-based 3D-stacked				
Duplex [44]	✓	✓	✗	✓
H^2-LLM [21]	✓	✗	✗	✗
Tri-Stack-PIM	✓	✓	✓	✓

bandwidth-limited host-device link that PIM is meant to eliminate. Third, HB has recently matured into a practical die-to-die integration technology [7, 15, 18], enabling a more flexible partitioning. HB-based 3D-stacked systems [21, 22, 41] stack a dedicated compute die—fabricated in a high-performance logic process—directly beneath the DRAM die, and connect the two through dense HB interconnects that offer high inter-die bandwidth. Decoupling the compute die from the DRAM process removes the logic-density limitation of the first paradigm, while the HB link avoids the host-device bottleneck of the second. This flexibility, however, comes at a cost: the HB bonding pads impose keep-out zones on the base die, fragment the logic die into many small regions, increasing placement and routing complexity for compute logic [22].

As Table 1 shows, no prior design closes all four gaps at once. DRAM-process PIM is capped by the low logic density of the DRAM process; host-side handling of nonlinear (non-GEMM) operations reintroduces the bandwidth-limited host-device bottleneck that PIM should remove; interface logic and control remain interleaved with arithmetic on the same compute-dedicated die; and none supports dynamic, cross-die scheduling at runtime. This last coupling is the most costly. When wide HB interfaces, host-facing PHYs, and control share one logic die with dense compute engines, they compete for the advanced-node silicon and routing meant for arithmetic. The cost is twofold: the consumed silicon, and the keep-out zones that HB bonding pads impose on the base die—fragmenting routable area and tightening place-and-route. This pressure only intensifies as interface width and operator complexity scale. Closing these gaps is not a matter of incremental refinement—a better neural processing unit (NPU) microarchitecture or a faster host link leaves the underlying coupling intact. What the industry needs is a fundamentally new, forward-looking 3D architecture.

We present *Tri-Stack-PIM*, a heterogeneous three-die 3D-stacked PIM architecture that resolves the above bottlenecks by physically disaggregating storage, interfacing, and computation across three specialized dies. A top DRAM die provides high-capacity storage; a middle Buffer Die on a mature node handles interfacing, control, and nonlinear computation; and a bottom Base Die on an advanced node is reserved exclusively for dense matrix computation. The Buffer Die

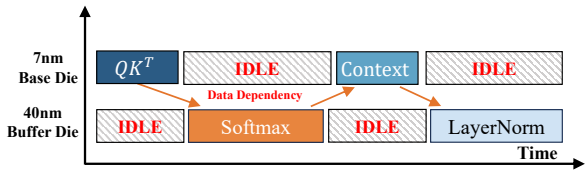


Figure 1: Per-channel execution trace of attention in Tri-Stack-PIM.

is the key enabler, and it pays off in two distinct ways. First, it restores a clean floorplan for the Base Die. To sustain full bandwidth, HB partitions the DRAM die into hundreds of independently-accessed banks; bonding their input/output (I/O) pads directly to the compute die below would scatter bond points across its entire area, shattering the floorplan into small, disjoint regions and sharply inflating placement-and-routing complexity for the compute logic [22]. Inserted between the two, the Buffer Die aggregates the per-bank I/Os and re-exposes them through a regularized interface, leaving the Base Die a contiguous canvas for arithmetic—at negligible cost to delivered bandwidth. Second, it turns the freed mature-node area into an active execution layer. Aggregation consumes only a fraction of the Buffer Die—so what should fill the rest? We use it to pull every fixed-function block off the critical compute die: the host-facing PHYs, the HB controller, the DMA engines, the softmax and reduction engines (SREs) that execute nonlinear operators, and the channel-local dynamic scheduler. This is a deliberate technology-function match. High-speed I/O, control, and datapath glue scale poorly with process node, so hosting them on a mature die costs almost nothing in performance while reclaiming scarce advanced-node silicon for the one thing that truly rewards a leading-edge process: compute density. Better still, co-locating the nonlinear operators here keeps all non-GEMM work inside the stack, eliminating the host round-trips that throttle prior designs.

The main contributions of this work are:

- We show that coupling interface, control, and compute on a single logic die throttles compute provisioning as HB width scales—both by consuming advanced-node silicon and by fragmenting the base die's floorplan—and propose a heterogeneous three-die partition that separates DRAM storage, mature-node interface/control, and advanced-node dense compute.
- We design a UBAR-based scheduler on the Buffer Die that tracks dependency readiness, DRAM timing, and backend occupancy to issue micro-operations out of order, reclaiming cross-die idle cycles that static scheduling cannot.
- We develop a coupled search over dataflow choices and hardware parameters that finds markedly better-balanced operating points than tuning either in isolation.
- Across the evaluated workloads, Tri-Stack-PIM achieves geometric-mean speedups of 4.95× over GPU-only, 2.57× over AttAcc, and 1.68× over H^2-LLM.

2 Motivation

As planar scaling yields diminishing returns [15], 3D integration has become the primary architectural path for sustaining performance growth. Within this paradigm, HB constitutes the most

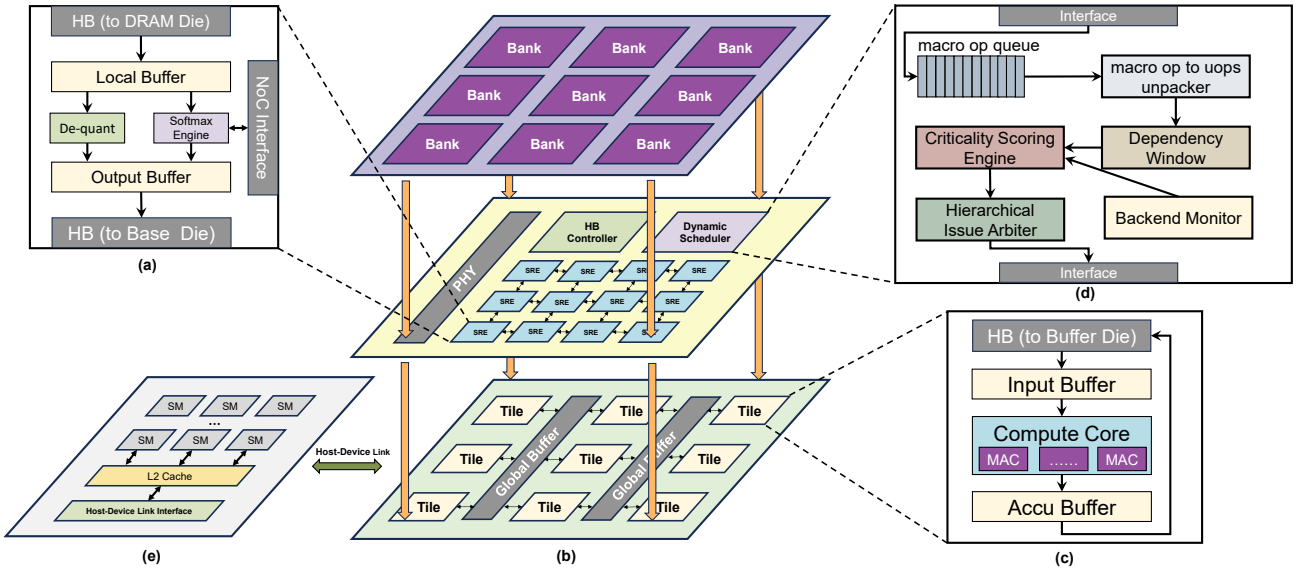


Figure 2: Tri-Stack-PIM overview. The top DRAM die stores weights and KV cache. The middle Buffer Die hosts host-facing PHY, HB controller, DMA, dynamic scheduler, and SREs. The bottom Base Die concentrates advanced-node silicon on compute tiles with local static random access memory (SRAM) buffers.

aggressive realization: by transcending the pitch constraints of conventional microbumps [25, 46], HB achieves sub-micron interconnect density that unlocks vertical memory bandwidth at a granularity fundamentally unattainable through prior 3D stacking approaches—transforming the economics of memory-bound computation.

However, state-of-the-art hybrid-bonded 3D PIM architectures remain subject to several practical constraints. First, because each DRAM bank bonds vertically to the base die through a dedicated HB interface, the proliferation of banks induces a commensurate proliferation of keep-out zones on the logic tier, progressively fragmenting the floorplan and penalizing placement-and-routing closure. Second, the co-integration of host-facing high-speed I/O, and control infrastructure consumes non-trivial silicon area on the base die, eroding the budget available for compute logic.

From an algorithmic standpoint, the arithmetic demand imposed on the logic die during decoding is intensifying. The widespread adoption of GQA/MQA [1, 36] means that a single KV-cache fetch is mathematically broadcast across G query heads (e.g., G=8 in LLaMA-2-70B), while batched inference further amplifies effective arithmetic intensity in proportion to $G \times B$, where B denotes batch size. Exploiting this reuse, however, demands a dedicated on-die buffer to stage KV heads across the broadcast and batching dimensions. Consequently, surrendering an excessive fraction of the already-constrained logic-die silicon to non-compute structures directly strangles the compute-engine budget and forfeits the very arithmetic density that modern attention mechanisms could otherwise sustain.

Consequently, existing HB-enabled PIM systems encounter a fundamental tension: the logic die is squeezed simultaneously from both the physical and algorithmic directions—per-bank keep-out zones and host-facing I/O infrastructure fragment and consume its already finite silicon, while GQA/MQA broadcast reuse ($G \times B$ arithmetic intensity) demands precisely the dense compute fabric that such overheads displace.

These limitations motivate our heterogeneous three-die stacking strategy. The maturation of multi-layer hybrid bonding and fine-pitch through-silicon-via (TSV) interconnects in industrial packaging [15, 22, 39, 40] now renders such multi-tier 3D integration manufacturable. We leverage these advances to interpose a mature-node buffer die between the compute and DRAM tiers. The benefits are twofold. First, redistributing all DRAM-facing hybrid-bond I/Os onto the buffer tier—rather than the logic die—eliminates the per-bank keep-out zones that would otherwise fragment the compute floorplan and inflate placement-and-routing complexity. Second, what should occupy the residual buffer-die area? We make the decision to co-locate the high-speed I/O PHYs, the nonlinear-acceleration unit, and controllers on this die. The rationale is economic: analog/mixed-signal circuits gain little from dimensional scaling, and advanced FinFET nodes actively worsen noise, variability, and voltage-headroom margins for PHY designs [33]. A mature planar node (e.g., 40/45 nm) thus provides a superior substrate—lower cost, simpler process, well-characterized models—while liberated advanced-node silicon is reclaimed exclusively for computation-heavy engines.

We therefore propose Tri-Stack-PIM (as shown in Figure 1), a heterogeneous three-die stacked architecture: the top DRAM dies furnish storage capacity; the sandwiched mature-node buffer die hosts high-speed I/O PHYs and memory-proximal low-arithmetic operators; and the bottom advanced-node base die dedicates its silicon exclusively to dense arithmetic.

However, this architecture introduces a critical scheduling challenge. Because the Base Die embodies the costliest advanced-node silicon in the stack, its utilization is the first-order determinant of attention throughput. Yet strict data dependencies between Base-Die GEMMs and Buffer-Die Softmax inherently expose the pipeline to Base-Die stalls absent effective scheduling (Figure 1).

A static software schedule is the first step: a compiler constructs a macro-operation directed acyclic graph (DAG), determines operator placement, and overlaps independent operators across dies [2, 10, 11]. This macro-instruction granularity marks the exact boundary of compile-time schedulability. Yet each macro-instruction decodes into numerous micro-operations whose readiness depends on runtime state-DRAM timing, die-to-die link congestion, heterogeneous-unit availability-none statically predictable. A static schedule is therefore brittle, and streaming compiler-generated μop sequences over the host link would trigger a control-traffic explosion. Inspired by the CDC 6600 scoreboard [37], we adopt a hardware-dynamic out-of-order μop scheduler that resolves these hazards at runtime.

Moreover, μops contend simultaneously for heterogeneous resources; mere hazard detection for correctness alone is insufficient without scheduling quality. The scheduler must perform prioritized multi-resource arbitration that maximizes utilization. We therefore introduce UBAR, a micro-scheduler that couples out-of-order issue with quality-driven multi-resource arbitration to dynamically orchestrate μop execution.

In summary, we propose Tri-Stack-PIM—a heterogeneous three-die architecture interposing a mature-node buffer die between DRAM and an advanced-node compute die to consolidate I/O and mixed-signal functions off-chip while reclaiming compute silicon exclusively for dense arithmetic—and UBAR, a hardware-dynamic out-of-order micro-scheduler coupling hazard resolution with quality-driven multi-resource arbitration to orchestrate μop execution where static scheduling cannot.

3 Tri-Stack-PIM Architecture

3.1 Design Principle

Tri-Stack-PIM is a command-driven dataflow near-memory platform. It contains no program counter, no instruction cache, and no branch predictor; execution advances by dependency readiness within an offline-compiled DAG. This maximizes arithmetic silicon on the base die. Three principles underlie the design. (1) *Data Proximity.* Rather than incrementally optimizing GPU/NPU architecture, Tri-Stack-PIM attacks the memory wall at its root: each DRAM bank connects via hybrid bonding directly to a dedicated compute tile, delivering ultra-high bandwidth at ultra-low latency. (2) *Separation of Compute and Control.* In 2 die stacked designs, host-facing I/O, controllers, and scheduling logic compete for area and routing resources with the compute fabric. Tri-Stack-PIM resolves this by concentrating all non-compute functions onto the buffer die (Figure 2), leaving the base die's entire area budget for arithmetic. This yields two advantages. First, the buffer die is fabricated on a mature node well suited to mixed-signal PHYs and control datapaths, while the base die targets an advanced node that maximizes MAC density per unit area. Second, it eliminates routing congestion on base die. (3) *Hybrid Static/Dynamic Scheduling.* Programming this near-memory platform presents a fundamental tension: a multiple-instruction multiple-data (MIMD) model demands per-tile firmware of prohibitive complexity; a single-instruction multiple-thread (SIMT) model requires a full compiler backend mapping attention kernels onto lockstep lanes. Tri-Stack-PIM sidesteps both extremes with a clear division of scheduling responsibility. The host compiler owns macro-level decisions—emitting a statically scheduled DAG of macro-instructions that encodes operator type, operand placement, and inter-operator dependencies—while UBAR resolves all μop-level ordering at runtime, exploiting dynamic state invisible to any compile-time analysis.

Figure 2 illustrates the overall architecture. The top DRAM die provides storage; the bottom base die contains one compute tile per bank—each equipped with SIMD lanes—mapped one-to-one to the DRAM banks above via hybrid bonds. The middle buffer die integrates the host-facing PHY, UBAR scheduler, DMA engine, and SRE units for near-memory softmax and reduction.

3.2 Execution Model

3.2.1 Hierarchical Instruction Set Architecture. The host compiler emits macro-instructions, each encoding a coarse-grained operation together with its data-dependency constraints. Four macro types suffice to express the complete attention dataflow: (1) `LAUNCH_GEMM`—a tiled matrix–vector multiply distributed across DRAM banks and base-die compute tiles, carrying a `repeat_count` field that encodes the GQA query-head fan-out over the fetched KV tile; (2) `SRE_SOFTMAX`—softmax accumulation and partial-sum reduction executed in the buffer-die SRE; (3) `DMA_H2D`—host-to-device tile transfer; (4) `MODE_SWITCH`—a global barrier that serializes execution phases. Every macro carries a compiler-assigned tag together with a predecessor list (`pre_tag`), collectively forming a static inter-macro dependency DAG; a successor may issue only after all of its predecessors have retired. Because macros belonging to independent GQA groups carry no cross-group dependencies, they may be dispatched concurrently across channels.

Upon arrival at the per-channel controller, each macro-instruction is decoded into micro-operations (μops) and posted to a hardware scoreboard. The controller maintains no program counter: issue eligibility is purely predicate-driven—a μop becomes ready when its outstanding dependency count reaches zero and the requisite execution resources are available.

3.2.2 Attention Dataflow. A single attention-head computation traverses four stages. ① The controller issues `ACTAB/RDAB` commands (All-Bank Activate and Read-All-Banks) to the DRAM die; each column burst delivers a KV-cache tile through the hybrid-bond interface, throttled by DRAM timing (t_{RCD}, $t_{\mathrm{CCD,AB}}$, t_{RP}, refresh). ② The tile lands in a base-die compute tile's local SRAM and is consumed by an SIMD MAC array; compute latency per burst is $\lceil (W_{\mathrm{HB}}/b_{\mathrm{quant}} \times r)/N_{\mathrm{lanes}} \rceil$ cycles, where W_{HB} is the HB data width, b_{quant} the quantization width, r the GQA repeat count, and N_{lanes} the SIMD lanes. ③ Partial query-key (QK) dot-product results are forwarded upward through the HB interface to per-bank SREs on the buffer die; when the head's hidden dimension is sharded across multiple banks, each SRE relays its partial scores over an on-die network-on-chip (NoC) to a geometrically elected master SRE for cross-bank reduction prior to softmax. ④ The master SRE executes softmax in a single streaming pass—doing maximum, exponential rescaling, and denominator accumulation. The normalized attention weights are then forwarded back through the HB interface to the base-die compute tiles, which execute a second `LAUNCH_GEMM` phase—fetching V-cache tiles from DRAM and accumulating the weighted value sum—to produce the final per-head context vector.

3.2.3 Base-Die Compute Tile Each compute tile is purely data-triggered (Figure 2c). It comprises three structures: (1) a local SRAM buffer absorbing incoming bursts, with backpressure to the scheduler when full; (2) an N_{lanes}-wide fully pipelined SIMD MAC array processing

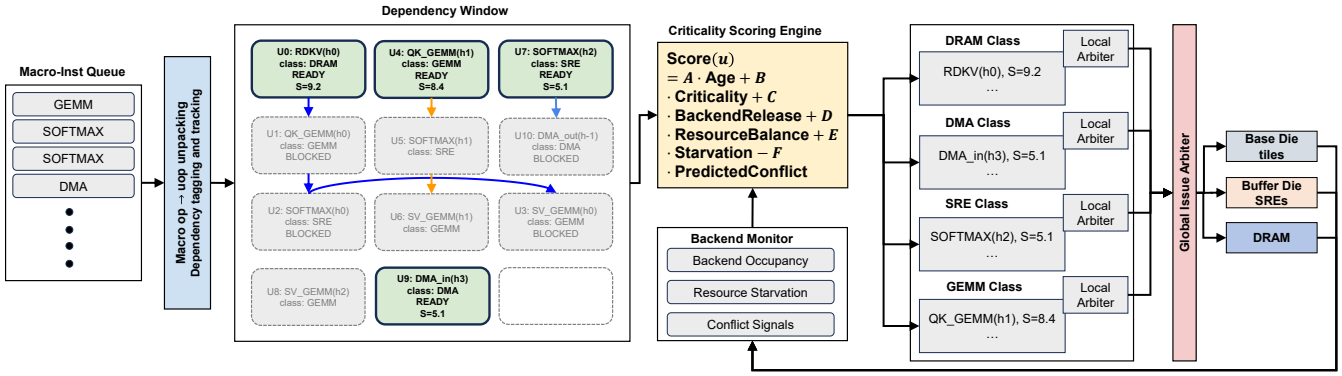


Figure 3: Dynamic scheduling pipeline on the Buffer Die: software emits macro-instructions that are unpacked into dependency-tracked micro-operation windows; a two-stage arbiter selects ready micro-operations using the UBAR priority score computed from runtime counters.

one element per lane per cycle, serving both the QK and score-value (SV) `LAUNCH_GEMM` phases sequentially; and (3) a ready-tick register that serializes successive bursts to the same tile, enforcing the constraint that the MAC array cannot accept a new chunk until the prior one retires.

3.2.4 Buffer-Die SRE. As shown in Figure 2(a), the SRE resides on the buffer die, where the mature node provides ample area at low cost and eliminates the >1 µs host round-trip that would otherwise be incurred by offloading softmax to the host. One SRE is instantiated per bank, with all SREs within a channel interconnected by a 2D-mesh NoC. Each SRE comprises a local SRAM buffer for staging partial-result tiles arriving from the base die, a reduction unit for cross-bank accumulation, and a softmax unit that performs running-max tracking, exponential rescaling, and denominator accumulation in a single streaming pass. Execution proceeds in four phases: ① base-die tiles traverse the hybrid-bonding interface into each co-located SRE buffer; ② when cross-bank reduction is required, a master SRE elected at the geometric centroid gathers tiles across the mesh (per-hop latency: 2 cycles; total transfer cost $\lceil tile_bytes/BW_{\text{NoC}} \rceil + h \cdot t_{\text{hop}}$); ③ the master accumulates the gathered partials into a single score vector; ④ softmax normalizes the result and writes the attention weights back for the subsequent SV-GEMM.

3.3 UBAR Dynamic Scheduler.

A single macro-instruction decomposes into tens to hundreds of µops whose completion latencies depend on runtime information that are fundamentally unpredictable at compile time. A naïve in-order policy enforces strict head-of-line ordering: whenever the oldest ready µop is denied issue due to a timing or resource constraint, all younger µops stall irrespective of their individual eligibility, squandering a substantial fraction of available issue slots under high concurrency. To overcome this limitation, we adopt out-of-order µop scheduling. To further improve issue quality, UBAR incorporates a priority-scoring mechanism.

Tri-Stack-PIM employs a two-level abstraction (Figure 3). The host emits coarse-grained macro-instructions with compile-time dependency edges (`tag/pre_tag`); a per-channel hardware unpacker expands each macro into fine-grained µops posted on a scoreboard with predecessor counts. A µop becomes ready when its dependency count reaches zero, entering the scheduling window visible to the arbiter.

The UBAR scheduler assigns each ready µop u a composite priority score:

$$S(u) = \alpha\, T_{\text{age}} + \beta\, T_{\text{crit}} + \gamma\, T_{\text{back}} + \delta\, T_{\text{bal}} + \epsilon\, T_{\text{starv}} - \zeta\, T_{\text{conf}}, \quad (1)$$

where T_{age} increases linearly with residence time and saturates at a configurable horizon, thereby favoring older µops without the unbounded growth that would reduce the scheduler to first-in-first-out (FIFO) order. T_{crit} encodes a static critical-path class assigned at µop creation time (Barrier > Softmax > GEMM > DRAM > DMA). T_{back} monitors the fill ratio of the base-die GEMM input FIFO and activates only when occupancy drops below a critical threshold, elevating the priority of DRAM-read µops capable of replenishing the data-starved compute pipeline—this constitutes the primary mechanism for breaking head-of-line blocking, and accordingly carries the largest default weight ($\gamma = 5\times$ the baseline). T_{bal} tracks per-class cumulative issue counts and penalizes over-represented classes, preserving issue diversity across heterogeneous resource types. T_{starv} provides an independent anti-starvation guarantee: it remains identically zero until a patience threshold is exceeded, then ramps to full scale at a saturation bound, ensuring bounded worst-case latency through a mechanism distinct from the smooth aging of T_{age}. T_{conf} is the sole subtractive term; it queries the DRAM timing model's per-bank open-row state and applies a full-scale penalty to row-buffer-missing DRAM-access µops, steering issuance away from predicted bank conflicts.

As shown in Figure 3, each channel issues µops through a two-stage arbiter designed to split the critical selection path for timing closure. In the first stage, ready µops are grouped into four priority classes and the highest-scoring entry in each class is elected as class champion; in the second stage, a global argmax selects the overall winner from among the four champions, falling through to the next-ranked champion on a resource-check failure rather than stalling. This decomposition bounds the selection fan-in to four regardless of scoreboard depth and closes the arbiter within a single buffer-die cycle.

Table 2: Joint design space. Each parameter interacts with at least one parameter in a different domain.

Domain	parameters	Choices
Dataflow	KV bank mapping	Round-robin, sequence-shard ×{1, 2, 4, 8}
Base Die	SIMD lanes (N_{lanes})	{32, 64, 128, 256, 512}
	Clock frequency	0.5–1.5 GHz (11 steps)
	Input buffer capacity	32 KB–8 MB (7 steps, log-spaced)
Buffer Die	Clock frequency	0.1–1.0 GHz (10 steps)
	SRE buffer capacity	32 KB–8 MB (7 steps, log-spaced)
Interconnect	HB data width (W_{HB})	{128, 256, 512, 1024, 2048, 4096} bits

3.4 Joint Architecture–Dataflow Co-Exploration (DSE)

The design parameters are tightly coupled: widening the hybrid-bonding I/O increases the per-read data volume yet yields no speedup once the SIMD array ceases to be memory-bound; conversely, widening the SIMD array elevates peak throughput only if the dataflow and workload characteristics keep enough tiles in flight to saturate it—that is, only if sufficient parallelism is exposed. Because no parameter can be tuned in isolation, a joint co-exploration of architecture and dataflow is necessary.

Table 2 enumerates seven parameters spanning four domains. Notably, the dataflow choice denotes the head-to-bank mapping, which determines how attention heads are distributed across DRAM banks and channels, thereby governing both inter-bank parallelism and row-buffer reuse. The arithmetic intensity of the attention kernel shifts with the group ratio $G = N_{\text{heads}}/N_{\text{KV}}$: MHA ($G$=1) is predominantly memory-bound, MQA (G=16) is predominantly compute-bound, and GQA (G=4) stresses both axes simultaneously. We therefore explore three representative models spanning these regimes—OPT-6.7B (MHA), PaLM-8B (MQA), and LLaMA3-8B (GQA). We employ multi-objective Bayesian optimization via Ax/BoTorch [6, 29], jointly minimizing latency, die cost, and power. Hardware block sizes and power modeling are detailed in Section 4.1. The resulting Pareto fronts are analyzed in §4.

4 Evaluation

4.1 Experimental Setup

We evaluate three models that span the attention-mechanism spectrum—OPT-6.7B (MHA, 32/32 heads), LLaMA3-8B (GQA, 32/8 heads), and PaLM-8B (MQA, 16/1 heads) [8, 13, 45]—across batch sizes {1, 4, 8, 16, 32} and (prompt, generation)-length pairs (128, 64), (512, 256), (1024, 64), and (1024, 16). The Base-Die compute tile is synthesized with the ASAP7 process design kit (PDK) [9] (0.5–1.5 GHz, 16–256 MACs/tile; 73.1–4614.5 μm^2/block, 0.15–41.2 fJ/MAC). The Buffer-Die SRE is synthesized with NanGate 45 nm PDK [27] (0.1–1.0 GHz; 0.47 mm^2, 1.1–9.5 pJ/softmax op). On-chip SRAMs are modeled with FinCACTI [35]: a 64 KB buffer occupies 0.116 mm^2 at 5.27 pJ/access (7 nm) versus 0.305 mm^2 at 10.55 pJ/access (45 nm). Die-to-die link energy (0.88 pJ/bit) is derived from the HB model of H^2-LLM [21]; Our Cycle-accurate simulation extends Ramulator 2 [26].

We compare against three baselines. GPU-Only models a representative edge GPU backed by 4-channel LPDDR5X with 30.7 GB/s effective bandwidth and 1.0 TFLOPS effective FP16 compute, evaluated via the analytical model open-sourced by AttAcc [32]. AttAcc [32] is simulated cycle-accurately on the same LPDDR5X assumption using its bank-group-level PIM configuration (128 GB/s internal bandwidth, 0.128 TFLOPS). H^2-LLM [21] is evaluated on our cycle-accurate simulator under its published parameters mapped to the same LPDDR organization for a controlled comparison. All experiments are executed on an Intel Xeon Platinum 8375C server at 2.90 GHz.

4.2 Performance Analysis

Since our work targets the attention mechanism rather than host-device orchestration, we report *per-token average attention latency*—aggregated over all 32 decoder layers—as the metric. The base die operates at 1.3 GHz and the buffer die at 1.0 GHz uniformly across GPU-only, AttAcc, H^2-LLM, and Tri-Stack-PIM; unless otherwise specified, the default SIMD width is 64 lanes for Tri-Stack-PIM and 32 lanes for H^2-LLM. Figure 4 presents the resulting attention latency across all workload configurations. Tri-Stack-PIM achieves geometric-mean speedups of 4.95× over GPU-only, 2.57× over AttAcc, and 1.68× over H^2-LLM.

Across all configurations, Tri-Stack-PIM achieves geometric-mean speedups of 14.15×/7.03× over GPU-only, 4.50×/2.12× over AttAcc, and 1.16×/2.05× over H^2-LLM on the MHA (OPT-6.7B) and GQA (LLaMA3-8B) workloads, respectively. The MHA gains are expectedly larger: at G=1 the workload is strictly memory-bound, leaving the GPU baseline bandwidth-starved and maximizing the benefit of near-memory compute. A similar disparity appears against AttAcc, whose absence of base-die SRAM reuse incurs redundant DRAM accesses that penalize the memory-intensive MHA most acutely.

On the MQA workload (PaLM-8B), Tri-Stack-PIM (SIMD = 256) achieves geometric-mean speedups of 3.58× over GPU-only, 5.20× over AttAcc, and 5.82× over H^2-LLM. Three observations emerge. First, AttAcc underperforms the GPU in every configuration: serving 16 query heads through bank-group-level PIM without a reuse buffer incurs four redundant KV scans—one per bank group—whereas the GPU amortizes a single KV fetch via L2 reuse. Integrating on-die SRAM is precluded by the DRAM process, which lacks the metal stack required for dense 6T cells. Second, H^2-LLM also trails GPU-only: its host-offloaded softmax introduces a per-head synchronization barrier bounded by host-device bandwidth, and, absent cross-tier functional disaggregation, the base-die area constrains the SIMD datapath to 32 lanes—leaving the system compute-starved relative to available DRAM throughput. Third, Tri-Stack-PIM at 64 SIMD lanes falls below AttAcc because the MQA case is compute-bound; however, the base-die area budget permits scaling to 256 lanes, at which point Tri-Stack-PIM surpasses all baselines decisively.

4.3 UBAR Dynamic Scheduler Effectiveness

We isolate UBAR's contribution along two orthogonal axes: (a) the latency reduction from out-of-order issue alone—reordering μops beyond strict in-order—and (b) the incremental gain of UBAR's multi-term scoring function over a naïve age-priority out-of-order (OoO) baseline.

4.3.1 Recoverable Scheduling Slack under In-Order Issue. Figure 5 quantifies the fraction of scoreboard-active cycles during which the In-Order stalls on a blocked head-of-line μop while at least one ready μop exists elsewhere.

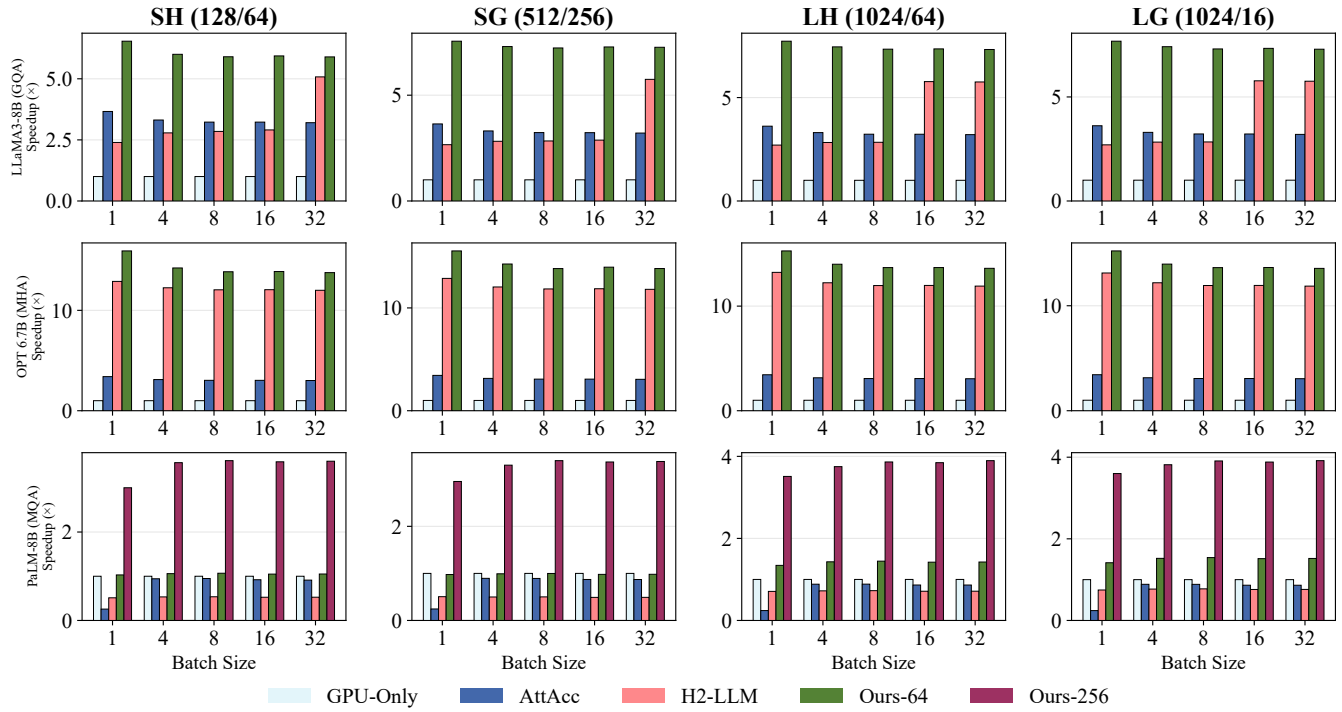


Figure 4: Per-token attention speedup across all evaluated workloads. Tri-Stack-PIM consistently outperforms the GPU-only, AttAcc, and H^2-LLM baselines.

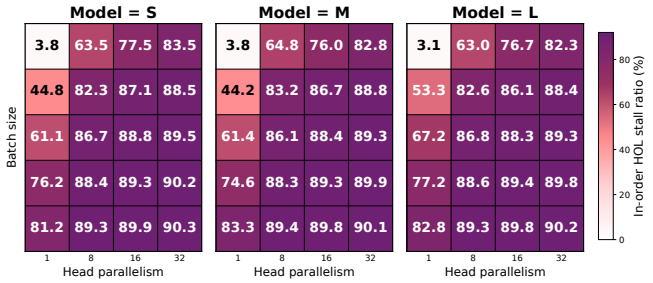


Figure 5: Fraction of cycles where in-order issue would stall on a blocked head uop despite a ready non-head uop being available. This recoverable slack is what the UBAR dynamic scheduler exploits.

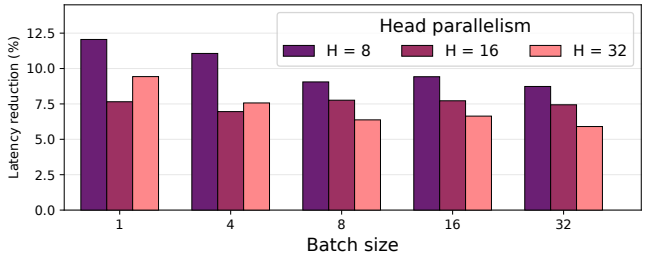


Figure 6: Latency reduction of UBAR over age-first OoO, grouped by batch size and head parallelism (averaged across model sizes S/M/L).

Results find that head parallelism and batch depth independently create scheduling opportunity, and both saturate rapidly. With H=1, B=1, the scheduler degenerates to a serial chain: only 3.5% of active cycles exhibit recoverable slack. Raising either axis alone suffices to unlock substantial opportunity: H=1, B=8 yields 63% slack through cross-request interleaving; B=1, H=8 yields 64% through intra-request head diversity. Once both axes exceed a modest threshold ($B \geq 8$, $H \geq 8$), the head-of-line stall ratio saturates at 86–90% regardless of model size, indicating that the in-order controller wastes nearly nine of every ten schedulable cycles on a conflict a ready-aware policy could bypass. This near-universal saturation—insensitive to model size once concurrency is sufficient—establishes OoO issue not as a corner-case optimization but as an architectural necessity for any PIM controller serving multi-head batched inference.

4.3.2 Contribution of UBAR's Multi-Term Scoring over OoO Alone. To isolate the contribution of quality-driven arbitration beyond mere out-of-order issue, we compare UBAR against an age-first OoO baseline that zeroes all scoring weights except timestamp—effectively an oldest-ready-first policy. Both schedulers share identical hardware parameters (SIMD = 32, HB = 2048 b, base die 500 MHz, buffer die 1.0 GHz, 1 MB+4 MB buffers) on a single-channel stress mapping with 8 ch × 16 banks. We sweep Model ∈ {S(2k), M(4k), L(8k)}, Batch ∈ {1,4,8,16,32}, and Heads ∈ {8,16,32} in MHA mode.

Figure 6 reports UBAR's latency reduction over age-first across all configurations. Three observations emerge. (1) *Scoring outweighs reordering.* The geometric-mean scoring gain is 9.4%, exceeding the 7.2% from OoO reordering alone; naïve oldest-first captures less than half the total 17.2% UBAR advantage. (2) *Gain peaks under low head parallelism.* At $H = 8$ the mean scoring gain reaches 11.7%, versus 8.2% at $H = 32$. (3) *Age-first can degrade performance.* At (S, $B = 1$,

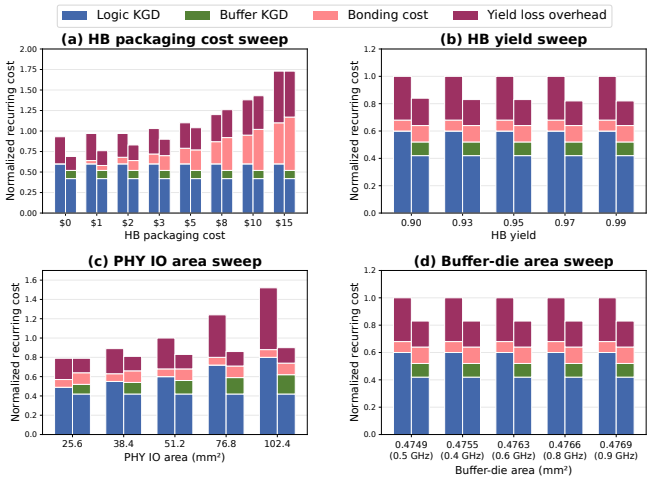


Figure 7: Yield-aware recurring-cost breakdown of Tri-Stack-PIM versus a two-die baseline, showing the tradeoff between reduced advanced-node logic cost and the added Buffer-Die and bonding overhead.

$H = 8$), age-first is 2.1% slower than in-order first-come-first-served (FCFS).

4.4 Design-Space Exploration Analysis

We find that the Pareto-optimal sweet spots reduce attention latency by 28.5×, 17.1×, and 39.8× for GQA, MHA, and MQA over the worst-case configuration, while compressing area by 130–95× and power by 1.5–7.6×. The resulting 33–66× EDP improvement confirms that the DSE simultaneously optimizes for speed and energy efficiency.

Takeaway 1: Arithmetic intensity dictates the effective HB bandwidth requirement. Widening the hybrid bond from 128 b to 4096 b yields 12× and 4.8× median latency reductions for MHA and GQA, but only 2× for MQA. MHA and GQA are memory-bound, so wider HB translates directly into proportional throughput gains. MQA, however, is compute-bound rendering additional bandwidth ineffective against this sequential bottleneck. **Takeaway 2: Dataflow strategy must be co-designed with the attention pattern.** At fixed HB width, mapping alone induces 2.3× (MHA) to 9.6× (MQA) latency variation. GQA favors contiguous hidden-dimension sharding to co-locate shared KV for concurrent multi-head access; MHA benefits from round-robin sequence sharding that distributes independent KV pairs across banks for maximum read parallelism; MQA prefers snake hidden-dimension sharding to spread computation across processing elements (PEs), exploiting SIMD parallelism on its single shared KV head. **Takeaway 3: SIMD lanes only pays off under compute-bound regimes.** For MQA, doubling SIMD lanes from 32 to 256 yields a near-linear 4.5× speedup, as the single serialized KV head leaves ample arithmetic work per byte fetched. For MHA and GQA, wider SIMD provides negligible benefit (< 1.1×) because the bottleneck is DRAM bandwidth—widening the HB bus, not the datapath, is the correct lever.

4.5 Overhead Analysis

Cost Analysis. We adopt the cost model of AccelStack [4]. Since the DRAM die is identical across both designs, it cancels in the comparison; we report only recurring manufacturing cost, excluding non-recurring engineering (NRE) expenditures. The per-unit cost thus reduces to

$$C_{\mathrm{RE}} = \frac{C_{\mathrm{logic}}}{Y_{\mathrm{logic}}} + \frac{C_{\mathrm{buffer}}}{Y_{\mathrm{buffer}}} + C_{\mathrm{package}}, \tag{2}$$

where each die yield Y follows the negative-binomial model $Y = Y_{\mathrm{wafer}}\,(1 + AD_0/\alpha)^{-\alpha}$, and C_{package} aggregates hybrid-bonding step cost and yield loss across all stacking interfaces. Defect densities are set to $D_0 = 0.10\,\mathrm{cm}^{-2}$ (7 nm) and $0.05\,\mathrm{cm}^{-2}$ (45 nm) [38]. The host-facing PHY occupies 3.2 mm^2 per channel and is modeled as node-invariant. Relocating this 51 mm^2 of interface silicon from the 7 nm base die (wafer cost \$9,346) to the 45 nm buffer die (wafer cost \$2,000) yields a 4.7× reduction in per-unit wafer cost for the PHY alone.

Figure 7 sweeps four parameters—hybrid-bonding cost per step, bonding yield, host-facing PHY area, and buffer-die area—to delineate the recurring-cost trade-off between Tri-Stack-PIM and an H^2-LLM-like two-die stacked design.

Overall, Tri-Stack-PIM remains cost-favorable provided per-step hybrid-bonding overhead stays below approximately \$5; beyond that threshold the additional bonding step erodes the saving (7a). Bonding yield, by contrast, exerts only marginal influence: across $Y_{\mathrm{bond}} \in [0.90, 0.99]$ total cost shifts by less than 4% (7b). The dominant lever is PHY I/O area (7c): each additional square millimeter costs 4.7× more at 7 nm than at 45 nm, so the cost advantage widens monotonically with PHY size. Conversely, enlarging the buffer die raises $C_{\mathrm{buffer}}/Y_{\mathrm{buffer}}$ at 45 nm and narrows the gap (Figure 7 (d)).

One cost factor lies outside this model. In a conventional two-die stack the base die is flip-chip bonded face-down to the interposer, presenting the host-facing PHY with a direct, well-characterized channel. Tri-Stack-PIM relocates the PHY to the buffer die, inserting an additional hybrid-bond interface and TSV tier into the host signal path. Accommodating this altered channel would necessitate PHY intellectual property (IP) re-design—an NRE and qualification expense that Eq. (2) does not capture. We leave quantification of this overhead to future work.

Thermal Analysis. High temperatures degrade DRAM retention, which raises the refresh rate and erodes effective bandwidth; reliable 3D-stacked design therefore requires the peak temperature to stay below 85 °C [42, 43]. We perform a steady-state thermal simulation with HotSpot-7.0 in detailed_3D grid mode [14, 17], modeling the full three-die stack (DRAM / buffer / base, each 50 µm thinned Si) with a 64×128 spatial grid per layer. We construct a per-block floorplan and calculate each block's power at maximum utilization; the host-facing Peripheral Component Interconnect Express (PCIe) Gen5 PHY, the dominant heat source on the buffer die, dissipates up to 3.58 W per stack (64 GB/s × 7 pJ/b [3, 28]). The total stack power under full load is 11.97 W. We model the vertical heat resistance of the HB interface with an thermal conductivity of 1.625 $\mathrm{W\,m^{-1}\,K^{-1}}$ [30]. For the heat sink, we evaluate two schemes: advanced liquid cooling (0.05 $\mathrm{K\,W^{-1}}$) and microfluidic cooling (0.01 $\mathrm{K\,W^{-1}}$) [31]. Under both, all three dies remain well below 85 °C: Peak temperatures (DRAM / buffer / base) are 63.74 / 63.75 / 62.63 °C with liquid cooling and 61.90 / 61.91 / 60.76 °C with microfluidic cooling.

5 Conclusion

We present Tri-Stack-PIM, a heterogeneous 3-die stacked architecture for LLM inference. Disaggregating DRAM, interface/control/SRE logic, and compute across three dies decouples interface scaling from

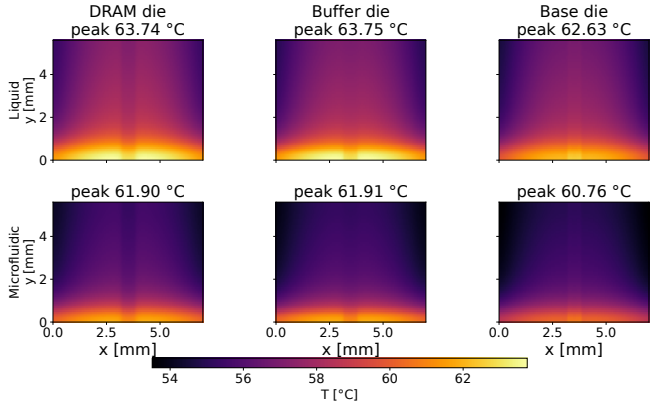


Figure 8: Steady-state temperature maps of the three-die stack. *Top*: advanced liquid cooling. *Bottom*: microfluidic cooling. Both remain 22 K below the 85 °C retention limit.

compute provisioning. A dynamic scheduler on the Buffer Die reclaims head-of-line stall cycles via the UBAR priority policy, while a joint co-exploration yields balanced configurations and design insights. Tri-Stack-PIM achieves geometric-mean speedups of 4.95×, 2.57×, and 1.68× over GPU-only, AttAcc, and H^2-LLM, respectively.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (No. 62404021), the Beijing Natural Science Foundation (No. L2603021), the Fundamental Research Funds for the BUPT (No. 2025AI4S20), the Research Initiation Project for Introduced Talents of BUPT (Nos. 510124058 and 510224062), and the Innovation Research Fund of BUPT.

References

[1] Joshua Ainslie, James Lee-Thorp, Michiel De Jong, Yury Zemlyanskiy, Federico Lebrón, and Sumit Sanghai. 2023. GQA: Training generalized multi-query transformer models from multi-head checkpoints. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. 4895–4901.

[2] Arvind and R.S. Nikhil. 1990. Executing a program on the MIT tagged-token dataflow architecture. *IEEE Trans. Comput.* 39, 3 (1990), 300–318.

[3] Woorham Bae and Sung-Yong Cho. 2021. A 1.93-pJ/bit PCI Express Gen4 PHY transmitter with on-chip supply regulators in 28 nm CMOS. *Electronics* 10, 1 (2021), 68. doi:10.3390/electronics10010068

[4] Chen Bai, Xin Fan, Zhenhua Zhu, Wei Zhang, and Yuan Xie. 2025. AccelStack: A cost-driven analysis of 3D-stacked LLM accelerators. In *2025 IEEE/ACM International Conference On Computer Aided Design (ICCAD)*. IEEE, 1–9.

[5] Fujun Bai, Xiping Jiang, Song Wang, Bing Yu, Jie Tan, Fengguo Zuo, Chunjuan Wang, Fan Wang, Xiaodong Long, Guoqing Yu, Ni Fu, Qiannan Li, Hua Li, Kexin Wang, Huifu Duan, Liang Bai, Xuerong Jia, Jin Li, Mei Li, Zhengwen Wang, Sheng Hu, Jun Zhou, Qiong Zhan, Peng Sun, Daohong Yang, Cheichan Kau, David Yang, Ching-Sung Ho, Hongbin Sun, Hangbing Lv, Ming Liu, Yi Kang, and Qiwei Ren. 2020. A stacked embedded DRAM array for LPDDR4/4X using hybrid bonding 3D integration with 34GB/s/1Gb 0.88 pJ/b logic-to-memory interface. In *2020 IEEE International Electron Devices Meeting (IEDM)*. IEEE, 6–6.

[6] Maximilian Balandat, Brian Karrer, Daniel R. Jiang, Samuel Daulton, Benjamin Letham, Andrew Gordon Wilson, and Eytan Bakshy. 2020. BoTorch: A framework for efficient Monte-Carlo Bayesian optimization. In *Advances in Neural Information Processing Systems 33 (NeurIPS)*. https://arxiv.org/abs/1910.06403

[7] Soon Aik Chew, Joeri De Vos, and Eric Beyne. 2024. Wafer-to-wafer hybrid bonding at 400-nm interconnect pitch. *Nature Reviews Electrical Engineering* 1 (2024), 73–74. doi:10.1038/s44287-024-00019-8

[8] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. 2023. PaLM: Scaling language modeling with pathways. *Journal of Machine Learning Research* 24, 240 (2023), 1–113.

[9] Lawrence T Clark, Vinay Vashishtha, Lucian Shifren, Aditya Gujja, Saurabh Sinha, Brian Cline, Chandarasekaran Ramamurthy, and Greg Yeric. 2016. ASAP7: A 7-nm FinFET predictive process design kit. *Microelectronics Journal* 53 (2016), 105–115.

[10] Jack B Dennis and David P Misunas. 1974. A preliminary architecture for a basic data-flow processor. In *Proceedings of the 2nd Annual Symposium on Computer Architecture*. 126–132.

[11] Joseph A Fisher. 1983. Very long instruction word architectures and the ELI-512. In *Proceedings of the 10th Annual International Symposium on Computer Architecture*. 140–150.

[12] Juan Gómez-Luna, Izzat El Hajj, Ivan Fernandez, Christina Giannoula, Geraldo F. Oliveira, and Onur Mutlu. 2022. Benchmarking a new paradigm: Experimental analysis and characterization of a real processing-in-memory system. *IEEE Access* 10 (2022), 52565–52608. doi:10.1109/ACCESS.2022.3174101

[13] Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurelien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Roziere, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, Danny Wyatt, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Francisco Guzmán, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Govind Thattai, Graeme Nail, Gregoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel Kloumann, Ishan Misra, Ivan Evtimov, Jack Zhang, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Karthik Prasad, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, Khalid El-Arini, Krithika Iyer, Kshitiz Malik, Kuenley Chiu, Kunal Bhalla, Kushal Lakhotia, Lauren Rantala-Yeary, Laurens van der Maaten, Lawrence Chen, Liang Tan, Liz Jenkins, Louis Martin, Lovish Madaan, Lubo Malo, Lukas Blecher, Lukas Landzaat, Luke de Oliveira, Madeline Muzzi, Mahesh Pasupuleti, Mannat Singh, Manohar Paluri, Marcin Kardas, Maria Tsimpoukelli, Mathew Oldham, Mathieu Rita, Maya Pavlova, Melanie Kambadur, Mike Lewis, Min Si, Mitesh Kumar Singh, Mona Hassan, Naman Goyal, Narjes Torabi, Nikolay Bashlykov, Nikolay Bogoychev, Niladri Chatterji, Ning Zhang, Olivier Duchenne, Onur Çelebi, Patrick Alrassy, Pengchuan Zhang, Pengwei Li, Petar Vasic, Peter Weng, Prajjwal Bhargava, Pratik Dubal, Praveen Krishnan, Punit Singh Koura, Puxin Xu, Qing He, Qingxiao Dong, Ragavan Srinivasan, Raj Ganapathy, Ramon Calderer, Ricardo Silveira Cabral, Robert Stojnic, Roberta Raileanu, Rohan Maheswari, Rohit Girdhar, Rohit Patel, Romain Sauvestre, Ronnie Polidoro, Roshan Sumbaly, Ross Taylor, Ruan Silva, Rui Hou, Rui Wang, Saghar Hosseini, Sahana Chennabasappa, Sanjay Singh, Sean Bell, Seohyun Sonia Kim, Sergey Edunov, Shaoliang Nie, Sharan Narang, Sharath Raparthy, Sheng Shen, Shengye Wan, Shruti Bhosale, Shun Zhang, Simon Vandenhende, Soumya Batra, Spencer Whitman, Sten Sootla, Stephane Collot, Suchin Gururangan, Sydney Borodinsky, Tamar Herman, Tara Fowler, Tarek Sheasha, Thomas Georgiou, Thomas Scialom, Tobias Speckbacher, Todor Mihaylov, Tong Xiao, Ujjwal Karn, Vedanuj Goswami, Vibhor Gupta, Vignesh Ramanathan, Viktor Kerkez, Vincent Gonguet, Virginie Do, Vish Vogeti, Vítor Albiero, Vladan Petrovic, Weiwei Chu, Wenhan Xiong, Wenyin Fu, Whitney Meers, Xavier Martinet, Xiaodong Wang, Xiaofang Wang, Xiaoqing Ellen Tan, Xide Xia, Xinfeng Xie, Xuchao Jia, Xuewei Wang, Yaelle Goldschlag, Yashesh Gaur, Yasmine Babaei, Yi Wen, Yiwen Song, Yuchen Zhang, Yue Li, Yuning Mao, Zacharie Delpierre Coudert, Zheng Yan, Zhengxing Chen, Zoe Papakipos, Aaditya Singh, Aayushi Srivastava, Abha Jain, Adam Kelsey, Adam Shajnfeld, Adithya Gangidi, Adolfo Victoria, Ahuva Goldstand, Ajay Menon, Ajay Sharma, Alex Boesenberg, Alexei Baevski, Allie Feinstein, Amanda Kallet, Amit Sangani, Amos Teo, Anam Yunus, Andrei Lupu, Andres Alvarado, Andrew Caples, Andrew Gu, Andrew Ho, Andrew Poulton, Andrew Ryan, Ankit Ramchandani, Annie Dong, Annie Franco, Anuj Goyal, Aparajita Saraf, Arkabandhu Chowdhury, Ashley Gabriel, Ashwin Bharambe, Assaf Eisenman, Azadeh Yazdan, Beau James, Ben Maurer, Benjamin Leonhardi, Bernie Huang, Beth Loyd, Beto De Paola, Bhargavi Paranjape, Bing Liu, Bo Wu, Boyu Ni, Braden Hancock, Bram Wasti, Brandon Spence, Brani Stojkovic, Brian Gamido, Britt Montalvo, Carl Parker, Carly Burton, Catalina Mejia, Ce Liu, Changhan Wang, Changkyu Kim, Chao Zhou, Chester Hu, Ching-Hsiang Chu,

Chris Cai, Chris Tindal, Christoph Feichtenhofer, Cynthia Gao, Damon Civin, Dana Beaty, Daniel Kreymer, Daniel Li, David Adkins, David Xu, Davide Testuggine, Delia David, Devi Parikh, Diana Liskovich, Didem Foss, Dingkang Wang, Duc Le, Dustin Holland, Edward Dowling, Eissa Jamil, Elaine Montgomery, Eleonora Presani, Emily Hahn, Emily Wood, Eric-Tuan Le, Erik Brinkman, Esteban Arcaute, Evan Dunbar, Evan Smothers, Fei Sun, Felix Kreuk, Feng Tian, Filippos Kokkinos, Firat Ozgenel, Francesco Caggioni, Frank Kanayet, Frank Seide, Gabriela Medina Florez, Gabriella Schwarz, Gada Badeer, Georgia Swee, Gil Halpern, Grant Herman, Grigory Sizov, Guangyi, Zhang, Guna Lakshminarayanan, Hakan Inan, Hamid Shojanazeri, Han Zou, Hannah Wang, Hanwen Zha, Haroun Habeeb, Harrison Rudolph, Helen Suk, Henry Aspegren, Hunter Goldman, Hongyuan Zhan, Ibrahim Damlaj, Igor Molybog, Igor Tufanov, Ilias Leontiadis, Irina-Elena Veliche, Itai Gat, Jake Weissman, James Geboski, James Kohli, Janice Lam, Japhet Asher, Jean-Baptiste Gaya, Jeff Marcus, Jeff Tang, Jennifer Chan, Jenny Zhen, Jeremy Reizenstein, Jeremy Teboul, Jessica Zhong, Jian Jin, Jingyi Yang, Joe Cummings, Jon Carvill, Jon Shepard, Jonathan McPhie, Jonathan Torres, Josh Ginsburg, Junjie Wang, Kai Wu, Kam Hou U, Karan Saxena, Kartikay Khandelwal, Katayoun Zand, Kathy Matosich, Kaushik Veeraraghavan, Kelly Michelena, Keqian Li, Kiran Jagadeesh, Kun Huang, Kunal Chawla, Kyle Huang, Lailin Chen, Lakshya Garg, Lavender A, Leandro Silva, Lee Bell, Lei Zhang, Liangpeng Guo, Licheng Yu, Liron Moshkovich, Luca Wehrstedt, Madian Khabsa, Manav Avalani, Manish Bhatt, Martynas Mankus, Matan Hasson, Matthew Lennie, Matthias Reso, Maxim Groshev, Maxim Naumov, Maya Lathi, Meghan Keneally, Miao Liu, Michael L. Seltzer, Michal Valko, Michelle Restrepo, Mihir Patel, Mik Vyatskov, Mikayel Samvelyan, Mike Clark, Mike Macey, Mike Wang, Miquel Jubert Hermoso, Mo Metanat, Mohammad Rastegari, Munish Bansal, Nandhini Santhanam, Natascha Parks, Natasha White, Navyata Bawa, Nayan Singhal, Nick Egebo, Nicolas Usunier, Nikhil Mehta, Nikolay Pavlovich Laptev, Ning Dong, Norman Cheng, Oleg Chernoguz, Olivia Hart, Omkar Salpekar, Ozlem Kalinli, Parkin Kent, Parth Parekh, Paul Saab, Pavan Balaji, Pedro Rittner, Philip Bontrager, Pierre Roux, Piotr Dollar, Polina Zvyagina, Prashant Ratanchandani, Pritish Yuvraj, Qian Liang, Rachad Alao, Rachel Rodriguez, Rafi Ayub, Raghotham Murthy, Raghu Nayani, Rahul Mitra, Rangaprabhu Parthasarathy, Raymond Li, Rebekkah Hogan, Robin Battey, Rocky Wang, Russ Howes, Ruty Rinott, Sachin Mehta, Sachin Siby, Sai Jayesh Bondu, Samyak Datta, Sara Chugh, Sara Hunt, Sargun Dhillon, Sasha Sidorov, Satadru Pan, Saurabh Mahajan, Saurabh Verma, Seiji Yamamoto, Sharadh Ramaswamy, Shaun Lindsay, Shaun Lindsay, Sheng Feng, Shenghao Lin, Shengxin Cindy Zha, Shishir Patil, Shiva Shankar, Shuqiang Zhang, Shuqiang Zhang, Sinong Wang, Sneha Agarwal, Soji Sajuyigbe, Soumith Chintala, Stephanie Max, Stephen Chen, Steve Kehoe, Steve Satterfield, Sudarshan Govindaprasad, Sumit Gupta, Summer Deng, Sungmin Cho, Sunny Virk, Suraj Subramanian, Sy Choudhury, Sydney Goldman, Tal Remez, Tamar Glaser, Tamara Best, Thilo Koehler, Thomas Robinson, Tianhe Li, Tianjun Zhang, Tim Matthews, Timothy Chou, Tzook Shaked, Varun Vontimitta, Victoria Ajayi, Victoria Montanez, Vijai Mohan, Vinay Satish Kumar, Vishal Mangla, Vlad Ionescu, Vlad Poenaru, Vlad Tiberiu Mihailescu, Vladimir Ivanov, Wei Li, Wenchen Wang, Wenwen Jiang, Wes Bouaziz, Will Constable, Xiaocheng Tang, Xiaojian Wu, Xiaolan Wang, Xilun Wu, Xinbo Gao, Yaniv Kleinman, Yanjun Chen, Ye Hu, Ye Jia, Ye Qi, Yenda Li, Yilin Zhang, Ying Zhang, Yossi Adi, Youngjin Nam, Wang, Yu Zhao, Yuchen Hao, Yundi Qian, Yunlu Li, Yuzi He, Zach Rait, Zachary DeVito, Zef Rosnbrick, Zhaoduo Wen, Zhenyu Yang, Zhiwei Zhao, and Zhiyu Ma. 2024. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783* (2024).

[14] Jun-Han Han, Kevin Skadron, et al. 2021. HotSpot 7.0: Enhanced compact thermal modeling for 3D-IC cooling solution exploration. ACM SIGDA University Demonstration at DAC.

[15] Tingbo He. 2026. The Tau (τ) scaling law: Exploration and practice of new paths in semiconductors. Keynote, IEEE International Symposium on Circuits and Systems (ISCAS), Shanghai, China. Huawei Technologies. Available: https://www.huawei.com/en/news/2026/5/ieee-iscas-tau-scaling.

[16] Guseul Heo, Sangyeop Lee, Jaehong Cho, Hyunmin Choi, Sanghyeon Lee, Hyungkyu Ham, Gwangsun Kim, Divya Mahajan, and Jongse Park. 2024. NeuPIMS: NPU-PIM heterogeneous acceleration for batched LLM inferencing. In *Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 3*. 722–737.

[17] Wei Huang, S. Ghosh, S. Velusamy, K. Sankaranarayanan, Kevin Skadron, and M.R. Stan. 2006. HotSpot: A compact thermal modeling methodology for early-stage VLSI design. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 14, 5 (2006), 501–513.

[18] Y. Kagawa, N. Fujii, K. Aoyagi, Y. Kobayashi, S. Nishi, N. Todaka, S. Takeshita, J. Taura, H. Takahashi, Y. Nishimura, K. Tatani, M. Kawamura, H. Nakayama, T. Nagano, K. Ohno, H. Iwamoto, S. Kadomura, and T. Hirayama. 2016. Novel stacked CMOS image sensor with advanced Cu2Cu hybrid bonding. In *2016 IEEE International Electron Devices Meeting (IEDM)*. IEEE, 8.4.1–8.4.4. doi:10.1109/IEDM.2016.7838375

[19] Jin Hyun Kim, Yuhwan Ro, Jinin So, Sukhan Lee, Shin-haeng Kang, YeonGon Cho, Hyeonsu Kim, Byeongho Kim, Kyungsoo Kim, Sangsoo Park, Jin-Seong Kim, Sanghoon Cha, Won-Jo Lee, Jin Jung, Jong-Geon Lee, Jieun Lee, JoonHo Song, Seungwon Lee, Jeonghyeon Cho, Jaehoon Yu, and Kyomin Sohn. 2023. Samsung PIM/PNM for transformer-based AI: Energy efficiency on PIM/PNM cluster. In *2023 IEEE Hot Chips 35 Symposium (HCS)*. IEEE Computer Society, 1–31.

[20] Daehan Kwon, Seongju Lee, Kyuyoung Kim, Sanghoon Oh, Joonhong Park, Gi-Moon Hong, Dongyoon Ka, Kyudong Hwang, Jeongje Park, Kyeongpil Kang, Jungyeon Kim, Junyeol Jeon, Nahsung Kim, Yongkee Kwon, Vladimir Kornijcuk, Woojae Shin, Jongsoon Won, Minkyu Lee, Hyunha Joo, Haerang Choi, Guhyun Kim, Byeongju An, Jaewook Lee, Donguc Ko, Younggun Jun, Ilwoong Kim, Choungki Song, Ilkon Kim, Chanwook Park, Seho Kim, Chunseok Jeong, Euicheol Lim, Dongkyun Kim, Jieun Jang, Il Park, Junhyun Chun, and Joohwan Cho. 2022. A 1y-nm 1.25 V 8Gb 16Gb/s/pin GDDR6-based accelerator-in-memory supporting 1TFlops MAC operation and various activation functions for deep learning application. *IEEE Journal of Solid-State Circuits* 58, 1 (2022), 291–302.

[21] Cong Li, Yihan Yin, Xintong Wu, Jingchen Zhu, Zhutianya Gao, Dimin Niu, Qiang Wu, Xin Si, Yuan Xie, Chen Zhang, and Guangyu Sun. 2025. H2-LLM: Hardware-dataflow co-exploration for heterogeneous hybrid-bonding-based low-batch LLM inference. In *Proceedings of the 52nd Annual International Symposium on Computer Architecture*. 194–210.

[22] Cong Li, Yihan Yin, Chenhao Xue, Zhao Wang, Fujun Bai, Yixin Guo, Xiping Jiang, Qiang Wu, Yuan Xie, and Guangyu Sun. 2026. Hardware-software co-design for 3D-DRAM-based LLM serving accelerator. *arXiv preprint arXiv:2603.04797* (2026).

[23] Cong Li, Zhe Zhou, Size Zheng, Jiaxi Zhang, Yun Liang, and Guangyu Sun. 2024. SpecPIM: Accelerating speculative inference on PIM-enabled system via architecture-dataflow co-exploration. In *Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 3*. 950–965.

[24] Ke Liu, Xuan Zhang, Jinin So, Jong-Geon Lee, Shin-Haeng Kang, Sukhan Lee, Songyi Han, YeonGon Cho, Jin Hyun Kim, Yongsuk Kwon, KyungSoo Kim, Jin Jung, Ilkwon Yun, Sung Joo Park, Hyunsun Park, Joonho Song, Jeonghyeon Cho, Kyomin Sohn, Nam Sung Kim, and Hsien-Hsin S. Lee. 2021. Near-memory processing in action: Accelerating personalized recommendation with AxDIMM. *IEEE Micro* 42, 1 (2021), 116–127. doi:10.1109/MM.2021.3097700

[25] Mei-Chien Lu. 2024. Hybrid bonding for ultra-high-density interconnect. *Journal of Electronic Packaging* 146, 3 (2024), 030802.

[26] Haocong Luo, Yahya Can Tuğrul, F Nisa Bostancı, Ataberk Olgun, A Giray Yağlıkçı, and Onur Mutlu. 2023. Ramulator 2.0: A modern, modular, and extensible DRAM simulator. *IEEE Computer Architecture Letters* 23, 1 (2023), 112–116.

[27] Nangate Inc. 2011. Nangate Open Cell Library. Version 1.3, standard-cell library for the FreePDK45 technology, accessed for research use.

[28] NVIDIA Corporation. 2022. NVIDIA opens NVLink for custom silicon integration. Press Release. https://nvidianews.nvidia.com/news/nvidia-opens-nvlink-for-custom-silicon-integration

[29] Miles Olson, Elizabeth Santorella, Louis C. Tiao, Sait Cakmak, Mia Garrard, Samuel Daulton, Zhiyuan Jerry Lin, Sebastian Ament, Bernard Beckerman, Eric Onofrey, Paschal Igusti, Cristian Lara, Benjamin Letham, Cesar Cardoso, Shiyun Sunny Shen, Andy Chenyuan Lin, Matthew Grange, Elena Kashtelyan, David Eriksson, Maximilian Balandat, and Eytan Bakshy. 2025. Ax: A platform for adaptive experimentation. In *Proceedings of the 2nd AutoML Conference (JMLR W&CP)*, Vol. 293. https://proceedings.mlr.press/v293/olson25a.html

[30] Herman Oprins, Vladimir Cherman, Tomas Webers, Soon-Wook Kim, Joeri de Vos, Geert Van der Plas, and Eric Beyne. 2020. 3D wafer-to-wafer bonding thermal resistance comparison: Hybrid Cu/dielectric bonding versus dielectric via-last bonding. In *2020 19th IEEE Intersociety Conference on Thermal and Thermomechanical Phenomena in Electronic Systems (ITherm)*. IEEE, 219–228.

[31] Yue Pan, Zihan Xia, Po-Kai Hsu, Lanxiang Hu, Hyungyo Kim, Janak Sharda, Minxuan Zhou, Nam Sung Kim, Shimeng Yu, Tajana Rosing, and Mingu Kang. 2025. Stratum: System-hardware co-design with tiered monolithic 3D-stackable DRAM for efficient MoE serving. In *Proceedings of the 58th IEEE/ACM International Symposium on Microarchitecture*. 1–17.

[32] Jaehyun Park, Jaewan Choi, Kwanhee Kyung, Michael Jaemin Kim, Yongsuk Kwon, Nam Sung Kim, and Jung Ho Ahn. 2024. AttAcc! Unleashing the power of PIM for batched transformer-based generative model inference. In *Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2*. 103–119.

[33] Behzad Razavi. 2002. CMOS technology characterization for analog and RF design. *IEEE Journal of Solid-State Circuits* 34, 3 (2002), 268–276.

[34] Minseok Seo, Xuan Truong Nguyen, Seok Joong Hwang, Yongkee Kwon, Guhyun Kim, Chanwook Park, Ilkon Kim, Jaehan Park, Jeongbin Kim, Woojae Shin, Jongsoon Won, Haerang Choi, Kyuyoung Kim, Daehan Kwon, Chunseok Jeong, Sangheon Lee, Yongseok Choi, Wooseok Byun, Seungcheol Baek, Hyuk-Jae Lee, and John Kim. 2024. Ianus: Integrated accelerator based on NPU-PIM unified memory system. In *Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 3*. 545–560.

[35] Alireza Shafaei, Yanzhi Wang, Xue Lin, and Massoud Pedram. 2014. FinCACTI: Architectural analysis and modeling of caches with deeply-scaled FinFET devices. In *2014 IEEE Computer Society Annual Symposium on VLSI*. IEEE, 290–295.

[36] Noam Shazeer. 2019. Fast transformer decoding: One write-head is all you need. *arXiv preprint arXiv:1911.02150* (2019).

[37] James E Thornton. 1964. Parallel operation in the Control Data 6600. In *Proceedings of the October 27–29, 1964, Fall Joint Computer Conference, Part II: Very High Speed Computer Systems*. 33–40.

[38] Vik's Newsletter. 2024. How foundries calculate die yield. https://www.viksnewsletter.com/p/how-foundries-calculate-die-yield. Accessed: 2026-07-11.

[39] Song Wang, Yixin Guo, Wenkai Tao, Xuerong Jia, Fujun Bai, Jie Tan, Yukun Wang, Liang Bai, Fengguo Guo, Qian Liu, Jing Li, Peng Yin, Fan Liu, Jiawei Liu, Xiaodong Long, Yutong Han, Zonghao Yu, Meng Cheng, Sheng Chen, and Xiping Jiang. 2025. A 3D unified analysis method (3D-UAM) for wafer-on-wafer stacked near-memory structure. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* (2025).

[40] Song Wang, Bing Yu, Wenwu Xiao, Fujun Bai, Xiaodong Long, Liang Bai, Xuerong Jia, Fengguo Zuo, Jie Tan, Yixin Guo, Peng Sun, Jun Zhou, Qiong Zhan, Sheng Hu, Yu Zhou, Yi Kang, Qiwei Ren, and Xiping Jiang. 2023. A 135 GBps/Gbit 0.66 pJ/bit stacked embedded DRAM with multilayer arrays by fine pitch hybrid bonding and mini-TSV. In *2023 IEEE Symposium on VLSI Technology and Circuits (VLSI Technology and Circuits)*. IEEE, 1–2.

[41] John Wuu, Rahul Agarwal, Michael Ciraula, Carl Dietz, Brett Johnson, Dave Johnson, Russell Schreiber, Raja Swaminathan, Will Walker, and Samuel Naffziger. 2022. 3D V-Cache: The implementation of a hybrid-bonded 64MB stacked cache for a 7nm x86-64 CPU. In *2022 IEEE International Solid-State Circuits Conference (ISSCC)*, Vol. 65. IEEE, 428–429.

[42] Zhiheng Yue, Huizheng Wang, Jiahao Fang, Jinyi Deng, Guangyang Lu, Fengbin Tu, Ruiqi Guo, Yuxuan Li, Yubin Qin, Yang Wang, Chao Li, Huiming Han, Shaojun Wei, Yang Hu, and Shouyi Yin. 2024. Exploiting similarity opportunities of emerging vision AI models on hybrid bonding architecture. In *2024 ACM/IEEE 51st Annual International Symposium on Computer Architecture (ISCA)*. IEEE, 396–409.

[43] Zhiheng Yue, Yang Wang, Chao Li, Shaojun Wei, Yang Hu, and Shouyi Yin. 2025. 3D-PATH: A hierarchy LUT processing-in-memory accelerator with thermal-aware hybrid bonding integration. In *Proceedings of the 58th IEEE/ACM International Symposium on Microarchitecture*. 78–93.

[44] Sungmin Yun, Kwanhee Kyung, Juhwan Cho, Jaewan Choi, Jongmin Kim, Byeongho Kim, Sukhan Lee, Kyomin Sohn, and Jung Ho Ahn. 2024. Duplex: A device for large language models with mixture of experts, grouped query attention, and continuous batching. In *2024 57th IEEE/ACM International Symposium on Microarchitecture (MICRO)*. IEEE, 1429–1443.

[45] Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. 2022. OPT: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068* (2022).

[46] Anqi Zhou, Yu Zhang, Fei Ding, Ziqi Lian, Renxi Jin, Yudong Yang, Qidong Wang, and Liqiang Cao. 2024. Research progress of hybrid bonding technology for three-dimensional integration. *Microelectronics Reliability* 155 (2024), 115372.

[47] Minxuan Zhou, Weihong Xu, Jaeyoung Kang, and Tajana Rosing. 2022. TransPIM: A memory-based acceleration via software-hardware co-design for transformer. In *2022 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*. IEEE, 1071–1085.