Podcast 12

Compute-in-Memory Processing Exploration

[speaker1]: Welcome back to the show. Today we're continuing our exploration of compute-in-memory processing — and why this model of computing behaves very differently from CPUs and GPUs, and is especially beneficial for edge AI.

[speaker2]: Right, and I think a good place to expand the discussion is to step back and look at the spectrum of processor architectures. Because Gemini-II doesn't really fit into the usual categories people are used to.

[speaker1]: Most listeners intuitively understand CPUs. They're the ultimate generalists. Your laptop can render documents, compile code, stream video, run spreadsheets, play games — all on the same hardware.

[speaker2]: That flexibility comes from architectural choices. CPUs devote a lot of silicon to varied resources along with their arithmetic logic units that help across many workloads. Workstations also incorporate a lot of DRAM that enable handling different types of applications and the required data. This flexibility comes at a price: general purpose computers are not optimized for price, power or size for any single workload.

[speaker1]: If we move up the specialization curve then we come to digital signal processors. DSPs were designed to offload arithmetic logic units of heavy math, especially multiplications and MAC operations.

[speaker2]: And GPUs evolved from that idea. They trade flexibility for MAC throughput. They're phenomenal at matrix arithmetic but you wouldn't use a GPU as a replacement for a general-purpose CPU.

[speaker1]: Exactly. Your GPU won't run your spreadsheet or control your robot. It's a specialist engine.

[speaker2]: And ASICs push specialization even further. If you design hardware for a single workload, you can drastically lower power and raise performance — but you lose almost all flexibility.

[speaker1]: So the deeper the optimization, the narrower the workload envelope.

[speaker2]: What's fascinating about GSI Technology's Gemini-I and Gemini-II is that they don't follow that traditional lineage.

[speaker1]: They're digital compute-in-memory architectures performing computation directly inside memory using bit-slice processing and in-memory logic — not analog current summation, and not conventional von Neumann execution.

[speaker2]: Which means they occupy a very unusual position in the design space. More flexible than ASICs and GPUs. Yet able to outperform GPUs for certain data-dominated workloads.

[speaker1]: And like GPUs, they operate as co-processors. They still rely on a CPU host — but the computational model is radically different.

[speaker2]: Yes, Compute-in-memory applies operations to data across the entire memory array simultaneously, minimizing data movement — the dominant bottleneck in modern AI workloads. This is especially important at the edge, where bandwidth and power are constrained and you are processing an immediate input data set or stream.

[speaker1]: Gemini's bit-processor structure is key here. In standard SRAM, activating multiple word lines causes collisions — essentially invalid behavior.

[speaker2]: But Gemini-II's compute structure is designed to exploit that electrical behavior to produce Boolean functions. What would be garbage in normal memory becomes logical functions.

[speaker1]: So each bit line acts like a RISC-style Boolean processor.

[speaker2]: Now let's connect that to arithmetic. CPUs accelerate math with ALUs. DSPs and GPUs extend that with arrays of dedicated multipliers and MAC units.

[speaker1]: Gemini-II builds arithmetic differently. Dual-port SRAM cells implement AND and XOR operations, which can be coded to any function directly inside memory.

[speaker2]: And once you have AND/XOR plus addition, multiplication becomes a sequence of Boolean operations and accumulation.

[speaker1]: One of the most underappreciated ideas here is bit-slice flexibility. Values exist as bit planes rather than fixed-width words.

[speaker2]: Precision becomes dynamic. You can run 1-bit operations, 2-bit, 3-bit, 4-bit, traditional words, or extremely wide representations like INT2048 simply by allocating bit planes.

[speaker1]: Which allows flexibility on math frameworks — something not possible to achieve in fixed datapath machines. It also allows the programmer explicit tradeoffs between accuracy, performance, and energy.

[speaker2]: There are one million bit-line processors in the Gemini-II, each with a 1-bit full adder and access to 768megabit of Register Memory. All operating at sub-nanosecond cycles.

[speaker1]: A MAC operation decomposes into predictable full-adder counts: (n × m + k) operations, depending on operand widths and accumulator size.

[speaker2]: And crucially, these operations execute where the data resides.

[speaker1]: In matrix multiplication, bit-line processors compute dot products independently. Matrix tiles load once into compute memory.

[speaker2]: For each MAC step, new operand values are broadcast, and accumulators expand to avoid overflow.

[speaker1]: Broadcast cost scales with bit width, not matrix size — a subtle but powerful distinction.

[speaker2]: Performing 128K MACs in parallel on one core completes in just a few dozen cycles.

[speaker1]: Unsigned and signed variants differ slightly, but the numbers stay remarkably small because everything is local and parallel.

[speaker2]: And Gemini-II supports multi-core parallelism providing multi-threaded compute-in-memory operations.

[speaker1]: Compare that with GPUs like NVIDIA Jetson AGX Orin or Jetson AGX Thor. Tensor Cores are incredibly fast, but they operate on register fragments, which forces a tiling strategy. When matrices exceed shared memory or cache capacity, data must shuttle repeatedly between DRAM and compute units.

[speaker2]: Even advanced features like Blackwell's Tensor Memory Accelerator reduce overhead but don't eliminate multi-pass memory traffic. And TTFT — Time To First Token — is dominated by exactly this behavior.

[speaker1]: Gemini-II can achieve single-pass memory behavior for large edge models. Load A once, stream B once, write C once. GPUs duplicate DRAM reads because tiles must be reloaded across K-dimension blocks. Transformer inference is essentially a cascade of matrix multiplications — projections, attention, and MLP layers.

[speaker2]: Which means external memory bandwidth frequently becomes the true limiter at the edge. This is where compute-in-memory becomes more than a performance story. It's about enabling real-time awareness within tight power envelopes. For workloads like Gemma-3 12B, early tokens often drive decisions — identifying anomalies, hazards, or environmental changes.

[speaker1]: Demonstrations have shown low second count responses to multimodal VLMs at roughly 30 watts — squarely within the range of deployable edge systems.

[speaker2]: So across the processor spectrum, Gemini-II represents a different axis of optimization. Not just faster math, but fundamentally less data movement.

[speaker1]: Compute where the data lives., with significantly less data movement.

[speaker2]: And when data movement dominates cost — which is increasingly common — that architectural shift becomes decisive.

[speaker1]: That wraps up today's extended deep dive.

[speaker2]: Thanks for listening, and we'll see you next time.