Future of Memory: Massive, Diverse, Tightly Integrated with Compute – from Device to Software

Shuhan Liu, Robert M. Radway, Xinxin Wang, Jimin Kwon, Caroline Trippel, Philip Levis,
Subhasish Mitra, H.-S. Philip Wong, Stanford University, CA 94305, USA; E-mail: {shliu98, hspwong}@stanford.edu

***Abstract*—** A renaissance of memory research has created many memory technologies with various trade-offs. Yet, computing systems are bottlenecked by memory accesses. How do we unlock the use of these emerging memories to overcome this bottleneck? We envision systems consisting of massive memories that are diverse and tightly integrated with compute, from the device technology to the software levels.

I. Introduction

Future computing systems must do more with less: higher throughput at significantly lower energy than today. Otherwise, computing's energy demands can far exceed sustainable energy production (projections in [1], [2]). Workloads such as Artificial Intelligence/Machine Learning (AI/ML) require massive off-chip memory and are throttled by the "memory wall" – significant time and energy spent shuttling data between compute and memory chip(s) [3]. This memory wall worsens as semiconductor technologies face the "miniaturization wall" – the inability to gain device density in 2D due to physical limits and fabrication complexity [4]. We face these walls just as memory needs *explode* for AI/ML, big data, and networked systems [5]–[7]. Thus, the large demands on memory, the memory wall, and the miniaturization wall are three critical challenges for future computing systems.

While software generally *assumes* a random-access memory address space with uniform latency and bandwidth, software *use* of that memory is far from uniform. Von Neumann recognized this fact, stating that "*various parts of this memory have to perform functions which differ somewhat in their nature and considerably in their purpose"* [8]. While the current hardware memory hierarchy – SRAM, DRAM, Flash – is already diverse, these devices alone are insufficient to meet software needs. Instead of expecting new devices *to replace* existing memory, we must focus on integration of memory *with new capabilities* as a tool in our toolbox.

For logic circuits, the field-effect transistor (FET) reigns supreme: we assume that will continue to be the case. In contrast, for memory, an abundance of new and traditional devices use a variety of physical mechanisms and materials [9], [10]. Software and system architects typically want memory that is better in all attributes; instead, we should exploit the wide range of tradeoffs across technologies (Table I) because domain specificity offers high efficiency. Memory research currently often focuses on *optimizing individual attributes* (e.g., specific entries in device comparison tables, as in Table II). Instead, we must match sets of desired attributes derived from software use cases (Sec. II).

Beyond being massive and diverse, memory must be *tightly integrated* with compute. Von Neumann "*ideally … desire[d] an indefinitely large memory capacity"* with any "*word … immediately available … considerably shorter than … a fast electronic multiplier.*" [11] Similar desires hold true for energy. We envision tightly integrated – *both physically and architecturally* – compute-memory systems: memory matched to software, with abstractions to expose and exploit diverse memory attributes (Sec. II, Fig. 2).

Unable to achieve this capacity, and *"forced"* into "*a hierarchy of memories*" [11], the memory wall is often *incorrectly attributed* to the von Neumann architecture ([12], [13]). Physical implementation of memory hierarchy with latency-capacity tradeoffs has led to *memory sub-systems in separate chips* for technological, cost, and business reasons (e.g., SRAM, DRAM, Flash). However, it is conceivable to integrate massive, diverse memories onto compute chip(s); we must focus on system-level benefits this brings (Sec. III).

Thus, irrespective of the architecture – von Neumann or non-von Neumann, we must provide: (1) massive amounts of memory with (2) diverse functionality, (3) tightly integrated both physically and architecturally with compute (Fig. 1). We must (4) exploit this integration in the software stack. We focus on points (1)-(3). (4) is beyond this paper's scope.

II. Software Use Cases for Diverse Memories

We focus on compute and *memory*, with *bulk data storage* out of our scope. We consider three (of many) use cases (Fig. 3). Attributes include: (1) read/write access *frequency*; (2) read/write address *predictability* (e.g., *sequential* access by address order); and (3) data lifetime (i.e., time from variable creation to destruction). Based on such software use cases, diverse memory technologies degenerate into *bands* (Table III). These *bands* guide future optimizations in a vast, multi-dimensional design space (Fig. 4). For each *band*, we must optimize the relevant combination of attributes, rather than focusing on a sole few and neglecting others.

***Type A. Frequent Reads, Infrequent Writes, Predictable Accesses*:** Such software writes data rarely and reads written data many times, with predictable (often sequential) accesses, which enables data aggregation with write buffering and read pre-fetching. Examples include types of AI/ML inference weight memory and processor instruction caches (a few thousand instructions executed billions of times – Fig. 3). The write vs. read imbalance presents an opportunity to trade write costs (energy, latency, endurance) for better reads (energy, delay, retention) which must be co-optimized with density, technology choice (e.g., MRAM, RRAM, PCM), and encoding (e.g., multi-bit/cell). While reads are prioritized, some *minimal write characteristics* are required depending on system needs (e.g., hourly model updates for Large Language Models). For example, improving MRAM/RRAM endurance alone without improving write energy may not be sufficient

for energy-constrained systems (Fig. 5 (a)), as utilizing gained endurance cycles can come with a large energy cost.

***Type B. Frequent Writes, Few Reads per Write, Short Data Lifetime*:** In this use case, software writes data in (often sequential) blocks, and soon reads it (sometimes only once), discarding it thereafter (e.g., streaming I/O, AI/ML activations, and data analytics, with block sizes and write-to-read times of 10's of kB in 10's of µs [14], MBs in ms [15]–[18], and 1-128MBs in seconds [19], [20]). Short data lifetimes enable trading retention for speed, density, or energy efficiency (e.g., gain cells [21], [22], MRAM [23], and FeRAM [24]). Consider retention-speed trade-off curves in gain cell design [21], [22] (Fig. 5 (b)): design choice (V_{TH}) along this curve must be driven by the application. This curve can shift towards ideal corners by other design knobs (e.g., voltage), device improvement (sharper on/off transition or lower SS), or circuit innovations (e.g., hybrid gain cells [25]).

***Type C. Random Reads, Sequential Writes*:** Such software reads randomly (e.g., Zipf-like) but writes sequentially across contiguous blocks, e.g., buffers for a file system. For example, during memory defragmentation, randomly updated values are read and written sequentially back (e.g., in live data write logs). Random read delay can be hidden via multi-threading, but low read energy is critical. On-chip gain cells offer much lower access energy than today's off-chip DRAM.

III. Tight Integration with Compute

The physical layer of memory-compute integration is illustrated in Table IV. High bandwidth memory access requires high connection density between memory and compute [26]–[29]. While today's 2.5D/3D advanced packaging has connection densities that are far lower than on-chip integration, the pin pitch of packaging technology is projected to shrink to below 1 µm [30], [31], thus providing a continuum of 3D interconnects down to very fine-pitched (10's of nm via monolithic 3D integration). These high connection densities enable re-architecting the system to exploit potentially massively increased bandwidth [26], [27].

Compute-in-memory (CIM) architectures aim to reduce data movement by performing computations directly within memory arrays. While CIM performs matrix-vector multiplication with high parallelism [32], AI/ML models require additional operations, e.g., vector-vector multiplication for attention [17] and depth-wise separable convolution [18], that are challenging for CIM [33]. Programmability and flexibility are especially challenging as the memory/compute elements are fixed at design time.

Analog CIM non-volatile memory (NVM) device non-idealities (Fig. 6 (a)) degrade accuracy due to large device variations, weight programming non-linearity, limited on-off ratios, state drift, and array IR drop [34]. Even chip-in-the-loop fine-tuning fails to maintain software accuracy for RRAM-based CIM [35]. Analog CIM requires costly DACs and ADCs to convert/quantize data, limiting energy efficiency, throughput, and density [35], [36]. Small weight kernels have low array utilization and can't amortize these input and output peripheral circuits. Large weight kernels that exceed single memory array capacity require post-ADC digital accumulation, reducing energy efficiency [37], [38]. Analog CIM exhibits attractive array-level energy efficiency for low-precision operations (e.g., <10 fJ per 4-bit OP), but this diminishes rapidly with bit width (e.g., >100 fJ per 8-bit OP) due to extra ADC energy to overcome thermal noise: each extra bit of precision requires quadruple the capacitance and energy [37]. As CIM arrays get larger, ADCs are noise-limited, and amortization benefits saturate [37].

Digital CIM follows the same computing model as traditional digital systems, with *separate digital logic and (digital) memory elements*. Digital CIM provides full bit accuracy, avoiding analog CIM's SNR limitations. In some cases, logic elements are tightly integrated within the array itself (Fig. 6 (b)). To match memory cell pitch and achieve high density requires simple logic elements – for example, 6T SRAMs modified with a few added transistors to implement basic logic [39]–[41]. This can impede programmability for different workloads and increase runtime, e.g., bit-serial compact ALUs to replace large-area multipliers [39].

If AI/ML models are larger than on-chip capacity, naïvely relying on off-chip memory results in the usual memory wall – regardless of analog CIM, digital CIM or other architectures. The solution is to orchestrate AI/ML execution across a system of multiple chips without any off-chip memories, instead relying on the sum total of each chip's local on-chip memory/compute elements [16], [26], [42]. Combined with special mappings of the AI/ML model to the system to minimize inter-chip traffic by co-optimizing per-chip memory size, heterogeneous inter- and intra-chip interconnects, and idle power via fine-grained power management, such a system can create the illusion of a much larger chip, e.g., a *Dream Chip* with all compute/memory on-chip. Illusion is thus distinct from traditional parallel processing. Such *Illusion Systems* (Fig. 7) have been well established for digital accelerator AI/ML systems of various sizes (e.g., 8× larger than single chips) with system-level energy and execution time within 5% of corresponding Dream Chips [16], [42].

Case Studies – Digital AI/ML Accelerators with Tightly Integrated Memory: *Hybrid Gain Cell (HGC)* has 3.6× density and lower energy than high-current (HC) SRAM (Fig. 8). HGC integrated with *RRAM* (Fig. 9) [43] for AI/ML training and inference saves > 80% energy. Fig. 10 shows how RRAM-based system non-volatility can provide up to 9× energy benefits vs. traditional memory systems (e.g., SRAM, off-chip DRAM and NAND Flash), even with foundry RRAM macros' similar density to foundry SRAM [15], [16].

IV. Conclusion

Memory must evolve from a uniform, random-access view to a heterogeneous collection of different memories optimized for different uses [44]. Memory's diverse characteristic tradeoffs must be exposed via appropriate abstraction to software (e.g., endurance vs. retention, energy vs. latency) for end-to-end device-system-software optimization. Tight integration (monolithic, 3D, 2.5D) of new memory is key to large capacity, low latency, and high bandwidth. Demands *will still* outpace capacity; thus, proper co-design of multi-chip system communication, e.g., Illusion Systems, is critical.

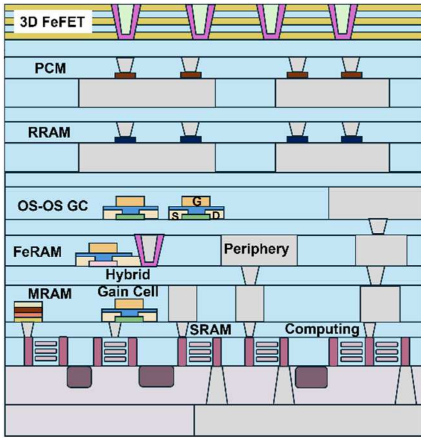
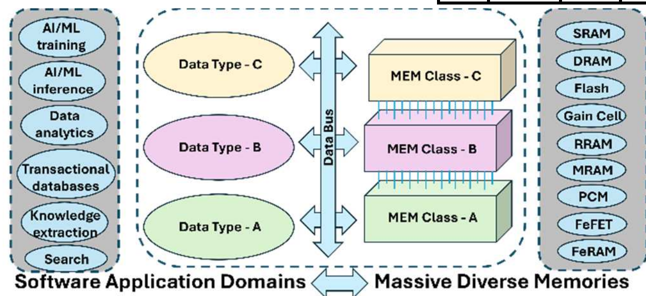


Fig.1: Future of memory: massive, diverse, and tightly integrated with compute.

Table I: Memory table based on device-level specifications that is driven by software use. Assume 1T1R structure for RRAM, MRAM, PCM with Si FEOL FET. Assume 3D NAND structure for FeFET. HGC: hybrid gain cell with Si read transistor and OS write transistor. OS-OS GC: 2T gain cell with both OSFETs for read and write. Standby power includes refresh power. Density is equivalent density after considering multiple layers. Possibility of single vs. multiple layer integration on-chip with logic.

	Read energy	Write energy	Standby power	Read latency	Write latency	Endurance	Retention	Density	Single layer	Multiple layers
High	FeRAM, DRAM Flash	RRAM, MRAM, PCM, Flash	DRAM	Flash	Flash	DRAM, SRAM, OS-OS GC, HGC	Flash, RRAM, MRAM, PCM, FeFET, FeRAM	3D Flash, 3D FeFET	SRAM	
Med.	RRAM, MRAM, PCM	DRAM, FeRAM	SRAM	RRAM, PCM, FeRAM, DRAM	FeRAM, DRAM, PCM, RRAM	FeRAM, MRAM	OS-OS GC, HGC	3D DRAM, 3D OS-OS GC	MRAM, PCM, RRAM, FeRAM, HGC	3D FeFET, OS-OS GC
Med.-Low	FeFET, OS-OS GC	FeFET	HGC, OS-OS GC	MRAM, OS-OS GC, FeFET	OS-OS GC, HGC, MRAM, FeFET	PCM, RRAM	DRAM	HGC, MRAM, RRAM, PCM, FeRAM	DRAM	
Low	SRAM, HGC	SRAM, HGC, OS-OS GC	RRAM, MRAM, PCM, FeFET, FeRAM, Flash	SRAM, HGC	SRAM	Flash, FeFET		SRAM	Flash	Flash, DRAM



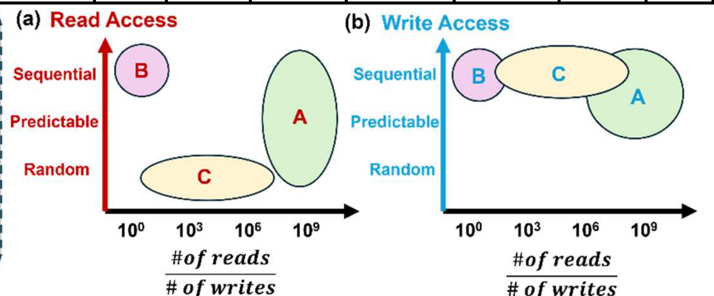


Fig. 2: Data-to-memory mapping by abstracting data types from software applications and memory classes from memory technologies. Memory classes are tightly integrated in the physical layer via a continuum of 3D vertical interconnects through monolithic integration or 2.5D/3D advanced packaging and allocated to different data types through data buses.

Fig. 3: Software use case data types categorized by read/write frequency and access pattern. For type A, Zsim [45] shows instruction cache misses $<5\times10^{-6}$ with thousands of instructions executed billions of times for each application run.

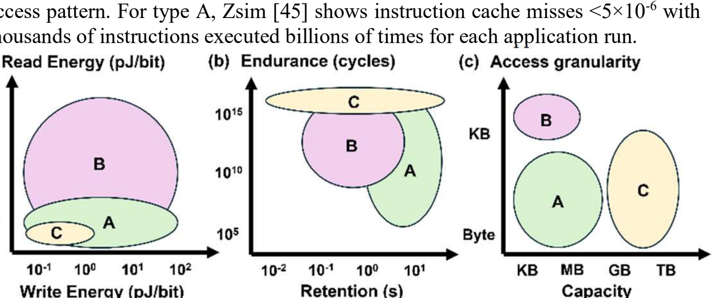


Fig. 4: (right) Domain-specific memory macro-level needs (i.e., including all peripheral circuits), derived from software use cases and system requirements. Three metrics sub-plots, where the relationships among memory types can be (a) subsets, (b) intersections, and (c) disjoint sets, exhibit the diverse and domain-specific needs for the memory types.

Table II: Traditional memory table with each attribute in isolation. Endur. = Endurance

	SRAM	DRAM	RRAM	MRAM
Energy	Low	Med	High	High
Speed	High	Med	Low	Low
Density	Low	Med	High	High
Endur.	High	High	Low	Med

Table III: Improvements needed for each memory technology to be used in the software use cases, based on state-of-the-art macro demonstrations. Integration can play a more important role than memory types, this is why 3D V-cache and DRAM fall into the same band, while OS-OS gain cell and HGC differ. We exclude 3D Flash and 3D FeFET from this table as they mainly target bulk data storage. Further research may realize CIM via these devices [46].

	SRAM	3D V-Cache	DRAM	OS-OS Gain Cell	Hybrid Gain Cell	RRAM	MRAM	PCM	FeRAM
A	Standby power	Read energy & speed & standby power	Read energy & speed & standby power	Read speed & capacity	Capacity	Write & read energy & endurance	Write & read energy	Write & read energy & endurance	Read speed & energy
B	Density	Standby power	Retention	Capacity	Capacity	Endurance & write energy	Write energy	Endurance & write energy	Read energy
C	Capacity	Read & write energy & speed	Read & write energy & speed	Capacity & read & write speed	Capacity & write speed	ALL	ALL	ALL	ALL

References (First Author, Venue, Year, for bevity) [1] N.C. Thompson, arxiv:2007.05558 [2] B.C. Lee, arxiv:2405.13858 [3] S. Williams, Comm. ACM '09 [4] H.-S. P. Wong, Proc. IEEE'20 [5] J. Sevilla, arxiv:2202.05924 [6] K. Kambatla, J. Parallel Distrib. Comput. '14 [7] H. Tataria, Proc. IEEE '21 [8] J. von Neumann, "First draft of a report on the EDVAC," 1945 [9] S. Yu, CRC Press '22. [10] H.-S. P. Wong, Nat. Nanotechnol. '15 [11] J. von Neumann, "Preliminary discussion of the logical design of an electronic computing instrument," 1946 [12] S. Basu, Proc. IEEE '19 [13] X. Zou, Sci. China Inf. Sci. '21[14] X. Jin, SOSP '17 [15] K. Prabhu, Symp. VLSI Circ. '24 [16] K. Prabhu, JSSC '22 [17] A. Vaswani, NeurIPS '17 [18] A.G. Howard, arXiv:1704.04861 [19] Apache Parquet [20] J. Peltenburg, ICFPT '20 [21] Shuhan Liu, IEDM '23 [22] Shuhan Liu, IEEE T-ED '24 [23] K.-F. Lin, ISSCC '24 [24] T. Francois, IEDM '21 [25] Shuhan Liu, Symp. VLSI Tech. '24 [26] R.M. Radway, IEDM '21 [27] T. Srimani, IEDM '23 [28] Y. H. Chen, ECTC '20 [29] M.-F. Chen, ECTC '19 [30] H.-J. Chia, ECTC '23 [31] W.-M. Wang, ECTC '24 [32] Q. Dong, ISSCC '20 [33] X. Wang, TCAS-II '22 [34] N. Lepri, IEEE T-ED '22 [35] W. Wan, Nature '22 [36] Q. Wang, IEDM '19 [37] B. Murmann, IEEE Trans. VLSI Syst. '21 [38] J. Yue, ISSCC '20 [39] H. Kim, JSSC '21 [40] Y.-D. Chih, ISSCC '21 [41] E.-J. Chang, Symp. VLSI '23 [42] R.M. Radway, Nat. Electron. '21 [43] Shuhan Liu, IEDM '24 (*to appear*) [44] P. Levis, https://dam.stanford.edu/ [45] D. Sanchez, Comput. Archit. News '13. [46] H.-T. Lue, IEDM '24 (*to appear*) [47] N. Pantano, ECTC '24 [48] A. Biswas, IEEE JSSC '19 [49] S. Xie, ISSCC '21 [50] X. Sun, IEEE TED '21 [51] S. Angizi, T-CADICS '19 [52] K. Zhang, ISSCC '24 [53] S.-Y. Wu, Symp. VLSI Tech. '09 [54] G. Gobieski, ASPLOS '19 [55] D. Garrett, ISSCC '23 [56] T.F. Wu, ISSCC '19 [57] S. Park, J. Syst. Arch. '11 [58] J. Wang, IEEE SSCL '21 [59] S. Lee, IEEE Access, '18

Table IV: Memories can be in the logic process itself (SRAM), in package (2.5D/3D chip stacking), monolithically 3D integrated in the back-end-of-line, or monolithically 3D integrated atop Si FETs in the front-end-of-line. Monolithic integration has higher connection pin density than in-package integration but requires fabrication at low temperatures. Off-chip package interfaces have limited bandwidth and significant latencies due to the macroscopic size and high capacitance of package pins and limited number of parallel connections. High-bandwidth memory (HBM) uses 5 – 50 μm-sized through-silicon vias (TSV) [26],[47] to bring logic and memory closer together, enabling high-bandwidth and lower-energy memory access. The interface protocols (e.g., DDR, HBM, PCIe, CXL, UCIe), are required as off-chip devices are fabricated by multiple entities, but can introduce energy, latency, and bandwidth overheads. These protocols are managed by software (from device drivers to operating systems), with minimum hardware support such as memory management units.

	SRAM	DRAM	HBM DRAM	Flash	Gain cell	RRAM	MRAM	PCM	FeFET	FeRAM
Integration	Logic process	In-package (e.g. DIMM)	In-package (3D stacked on logic chip) on interposer / substrate	In-package	Monolithic 3D in BEOL or Monolithic 3D in BEOL with Si FET FEOL	Monolithic 3D in BEOL with Si FET FEOL	Monolithic 3D in BEOL with Si FET FEOL	Monolithic 3D in BEOL with Si FET FEOL	Monolithic 3D in Si FEOL or monolithic 3D in BEOL	Monolithic 3D in BEOL with Si FET FEOL
Connection pin density	High	Low	Medium	Low	High	High	High	High	High	High
Application	Cache	Main memory	Main memory	Storage	Cache, main memory	Storage, main memory	Cache, main memory	Storage, main memory	Storage, main memory	Storage, main memory

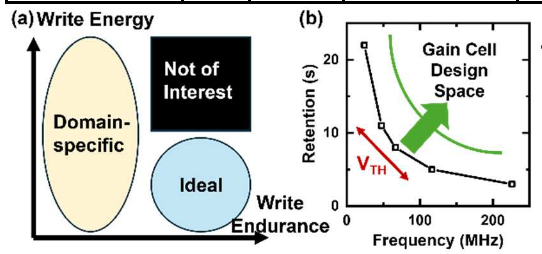


Fig. 5: (a) Write energy-endurance subspace divided into three regions: ideal, domain-specific, and not-of-interest. The high-endurance, high-energy corner is not of interest, as the high write energy discourages frequent writes. (b) Gain Cell retention-speed subspace with trade-off curve (black) [22] that arises from a limited transistor I_{ON}/I_{OFF} ratio. Curve shifts towards ideal corners (green) through other design knobs (e.g. voltage), device improvements like smaller SS and on/off transition region, or circuit innovations like hybrid gain cells.

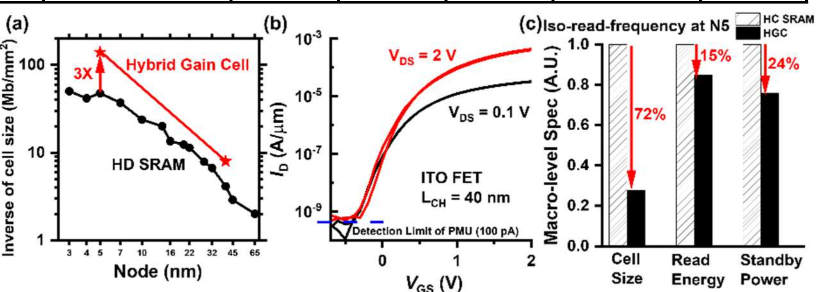


Fig. 8: (a) Hybrid gain cell has density 3× of HD SRAM and scalable to FinFET nodes with a cell size of ~1CGP×4MP, where CGP is the contacted gate pitch and MP is M0 pitch with an AB mask. The OSFET requires a modest channel length (L_{CH}) of 34 nm assuming CGP of 51 nm. (b) ITO FET with L_{CH} = 40 nm, experimentally demonstrated. (c) Cadence Spectre circuit simulation for hybrid gain cell vs. SRAM memory macro with the same peripheral circuit and array architecture at 5nm node. HGC has less area, read energy, and standby power compared to HC SRAM iso-frequency (2GHz) at the macro-level.

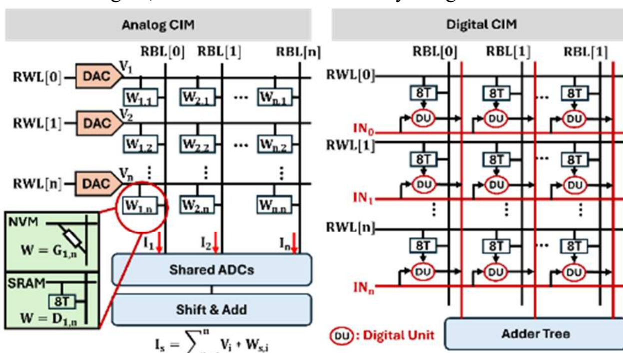


Fig. 6: CIM architectures can be broadly categorized into analog CIM or digital CIM. Analog CIM can be implemented with charge-based memories (SRAM [48], DRAM [49]) or resistance-based memories (RRAM [35], PCM [50], MRAM [51]). Analog CIM operates based on Ohm's Law and Kirchoff's law, with MVM results represented by the accumulated output current along bitline (BL) [35]. Digital CIM is typically implemented with charge-based memories with a localized computing unit that operates on the stored weight and the input activation signal. The resulting digital outputs are then summed using an adder tree to yield the final accumulated result [39]. The bitcell-wise digital unit can range from a simple logic gate (e.g., NOR [40]), to more sophisticated circuits with some area penalty (e.g., a NOR gate, a full adder and multiplexer [39]).

Acknowledgements: Department of Defense Microelectronics Commons California-Pacific-Northwest AI Hardware Hub, Eccalon/DoD, National Science Foundation (award number 2235329), SRC JUMP 2.0 CHIMES Center and PRISM Center, industry affiliates of the Stanford SystemX Alliance, Stanford Non-Volatile Memory Technology Research Initiative (NMTRI), and Stanford Differentiated Access Memory (DAM) programs. We thank graduate students, W.-C. (Harry) Chen and X, Wu, for editing and reviewing the manuscript, Prof. Boris Murmann for discussions, and Nvidia for providing resources for simulations. Present affiliations are: Jimin Kwon, UNIST, Korea; Robert M. Radway, University of Pennsylvania, USA.

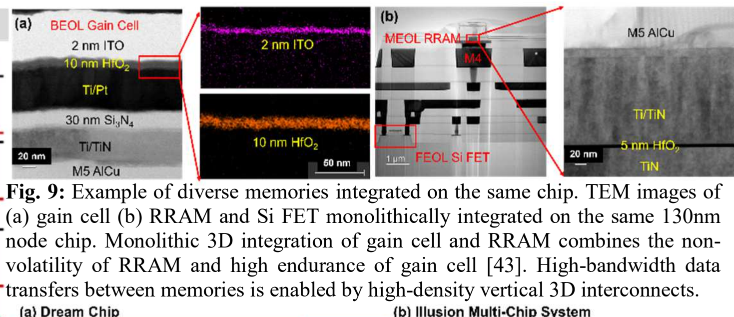


Fig. 9: Example of diverse memories integrated on the same chip. TEM images of (a) gain cell (b) RRAM and Si FET monolithically integrated on the same 130nm node chip. Monolithic 3D integration of gain cell and RRAM combines the non-volatility of RRAM and high endurance of gain cell [43]. High-bandwidth data transfers between memories is enabled by high-density vertical 3D interconnects.

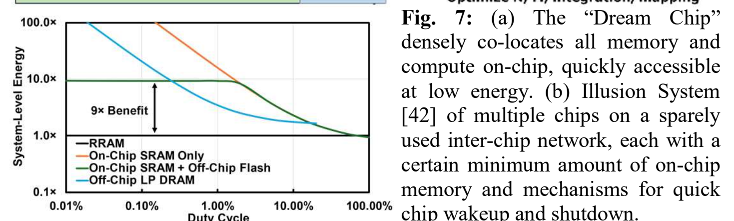


Fig. 7: (a) The "Dream Chip" densely co-locates all memory and compute on-chip, quickly accessible at low energy. (b) Illusion System [42] of multiple chips on a sparely used inter-chip network, each with a certain minimum amount of on-chip memory and mechanisms for quick chip wakeup and shutdown.

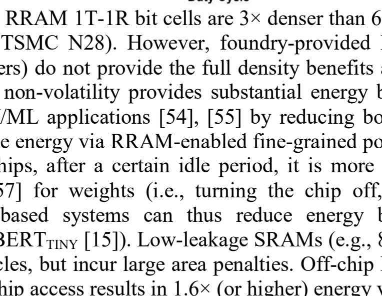


Fig. 10: RRAM 1T-1R bit cells are 3× denser than 6T SRAM (0.042 μm^2 [52] vs. 0.127 μm^2 [53] in TSMC N28). However, foundry-provided RRAM macros (including peripherals, controllers) do not provide the full density benefits at the system level [15], [16]. RRAM's on-chip non-volatility provides substantial energy benefits for infrequent and intermittent edge AI/ML applications [54], [55] by reducing both idle energy between inferences and inference energy via RRAM-enabled fine-grained power gating [15], [16], [56]. For SRAM-based chips, after a certain idle period, it is more energy-efficient to use backup NAND Flash [57] for weights (i.e., turning the chip off, re-fetching weights for new inputs). RRAM-based systems can thus reduce energy by 9× for AI/ML applications (e.g., MobileBERT$_{TINY}$ [15]). Low-leakage SRAMs (e.g., 8T SRAM [58]) shift this point to lower duty cycles, but incur large area penalties. Off-chip DRAM [59] has low refresh power, but its off-chip access results in 1.6× (or higher) energy vs. RRAM-based systems.