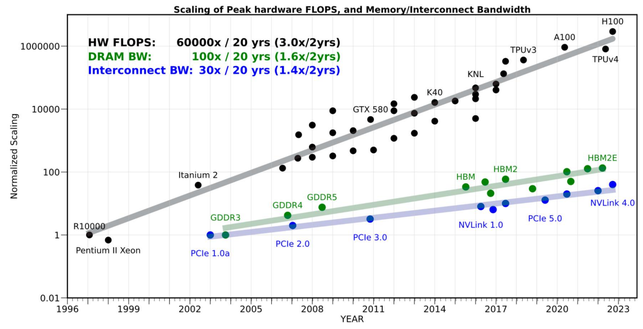
A Preliminary Comparative Analysis of Face-to-Face Bonded Long-Term RAM (LtRAM) and SRAM Chiplets vs. Off-Chip HBM for Agentic Inference

Samuel A. Dayo, Alex Aranburu, Victor J.B. Jung, Colin H. Drewes, Shridhar Mukund**
Prof. Robert M. Radway, Prof. Christos Kozyrakis, Prof. Luca Benini, Prof. H.S Philip Wong, Prof. Subhasish Mitra

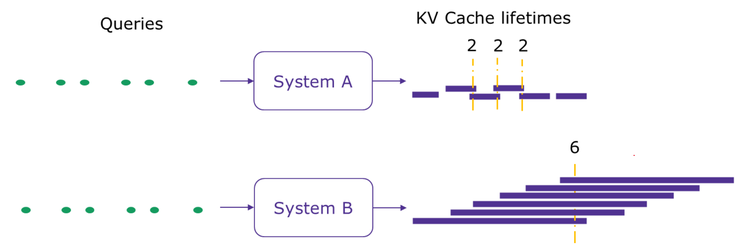
**EMD Electronics*

Motivations: Memory-Bandwidth Bottleneck



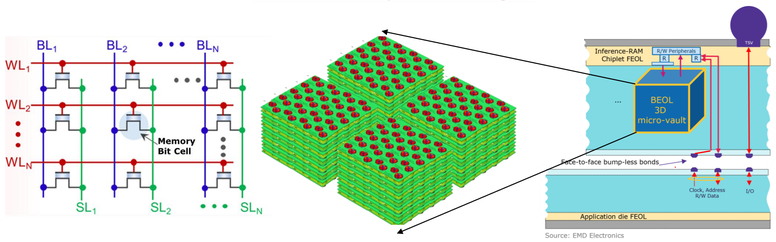
A. Gholami *et al.*, 'AI and the Memory Wall', 2024

Motivations: Reduce Active Memory Footprint



Long-term RAM (LtRAM)

LtRAM Chiplet Design:



Source: EMD Electronics

LtRAM Chiplet Summary:

Feature	Specification
Density & Granularity	0.32 GB/mm²; 1250 ports/mm²; 32 KB Banks
Area-scalable Bandwidth	< 2.32 TB/s/mm² (scalable from 40 mm² to wafer-scale)
Read Latency	< 8ns (4-stage pipelined random-access, < 2ns per stage)
Read Energy	~0.01 pJ/b
Read Voltage	< 500 mV
Write Latency	< 16 ns (8-cycle write)
Write Energy	0.04 – 0.06 pJ/b
Write Voltage	1.5 V
Write Endurance	10^{12} cycles

LtRAM Chiplet Density and Energy Benefits:

$$\frac{\textit{Memory Bandwidth}}{mm^2} \geq \frac{\textit{Max MACOPs/sec}}{mm^2} \geq \left(\frac{\textit{Memory Capacity (Weights + KV Cache)}}{mm^2}\right)\downarrow * \left(\frac{\textit{Number of Tokens/sec}}{mm^2}\right)\uparrow$$

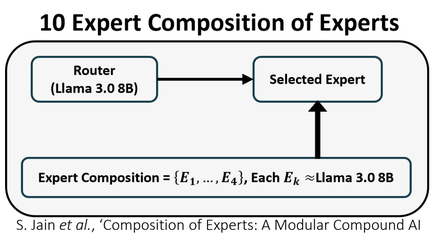
Spatially Scalable Key Performance Indicators (KPIs)	On-Chip F2F Hybrid Bonded Memory Type (AI Accelerator Chip)			Off-Chip HMB
	DRAM Wide I/O ports	SRAM	Inference RAM	Perimeter Limited 4 per 800 mm ² 1.6 TB/sec per HBM
Port Density (ports/mm²)	Very Low §~0.01 §6 per 800 mm ²	Low ~83 32KB Macro	High 1250 32KB Macro	V Very Low 0.005
Data Granularity (bytes/transaction)	Coarse 2,048	Fine 4, flexible	Fine 4, flexible	Coarse 2,048
Memory Density (Usable Gb/mm²)	High §~0.2	Very Low ~0.025	Higher 0.32	Highest §~0.2 x 12
Bandwidth Density (TB/sec per mm²)	Low §0.016	Medium ~0.116	High 2.32	Very Low 0.008
Read Energy (pJ/bit)	High ∅~1	Low ~0.01	Low 0.01	Very High ∅20

* Winbond DRAM CUBE
∂ On the Model of Computation, William Dally, 2022
§ Best guess based of known granularity limited by peripheral overhead and Serdes interface

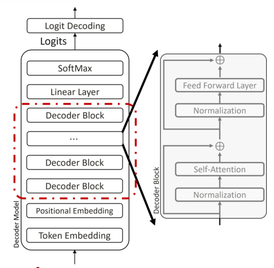
Quantitative Analysis Framework

Agentic Workload Modeling:

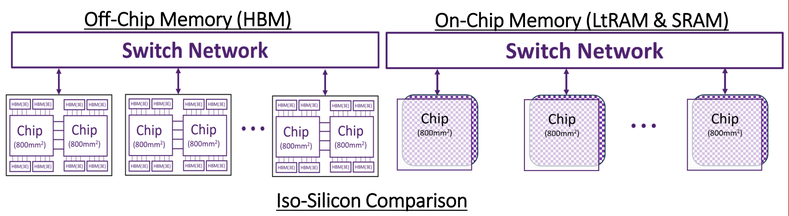
10 Expert Composition of Experts



S. Jain *et al.*, 'Composition of Experts: A Modular Compound AI System Leveraging Large Language Models', 2024

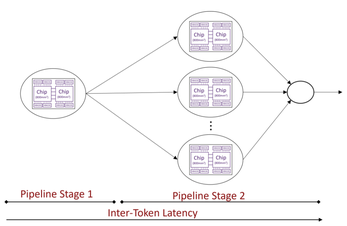


Memory-Compute Integration:



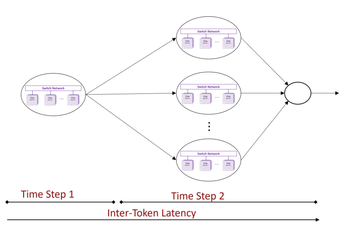
Iso-Silicon Comparison

Composition of Experts Execution Modes:



Expert Pipelining

- Dedicated expert per package
- Concurrent full batch execution
- Queries held during batch collation (overhead)



Time Partitioning

- Experts interleaved across packages (load balanced)
- Pipelined model execution, single query per stage
- Queries streamed in and launched (no overhead)

Quantitative Analysis Results

Performance Metrics:

Query Response Time

- Time to complete one inference query (I + O)

Aggregate Token Throughput

- Tokens processed per second (I + O)

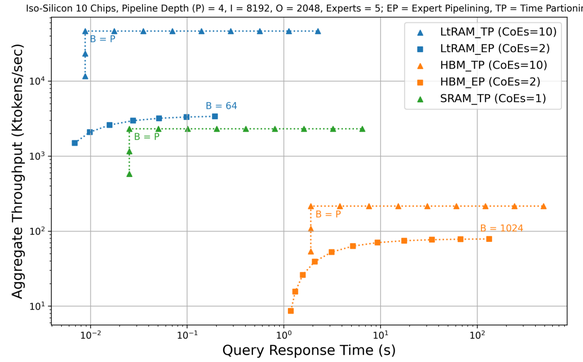
Memory Capacity per Query Throughput

- Total memory (KV cache + weights) needed per query/sec

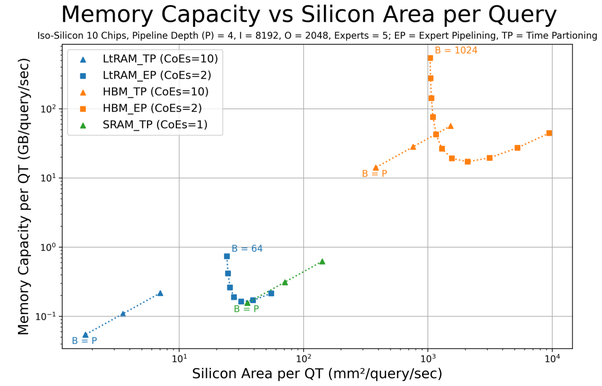
Silicon Area per Query Throughput

- Total accelerator area (mm²) needed per query/sec

Aggregate Throughput vs Query Response Time



Memory Capacity vs Silicon Area per Query



Long-Term RAM – Key Benefits

1. **Compute-Bound Query Processing**
   - Enables sustained throughput scaling at near iso-latency
   - Unlocks peak compute utilization across pipeline depths
2. **Compact Active Memory Footprint**
   - Reduced per-query memory footprint
   - Multi-query concurrency with minimal memory overhead
3. **Throughput-Dense Silicon**
   - Near-linear throughput scaling with chip count
   - Iso-throughput targets with significantly fewer accelerators

BEOL-Compatible IGZO FeFET Nanodendrite Enabling Sequence Decoding for Dendrocentric Computing

Hugo J.-Y. Chen[1], Chi-Hsin Huang[1], Kwabena Boahen[1,2,3], and H.-S. Philip Wong[1]

[1]Department of Electrical Engineering, [2]Department of Bioengineering, [3]Department of Computer Science,
Stanford University, Stanford, CA, USA

Background knowledge & Motivation

- **Goal: New energy-efficient AI hardware for neuromorphic computing**
- Accelerate data retrieval in language model

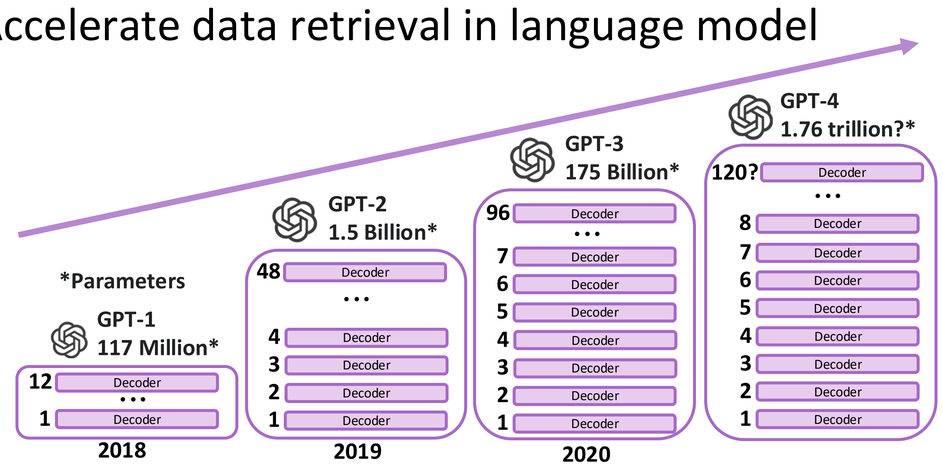
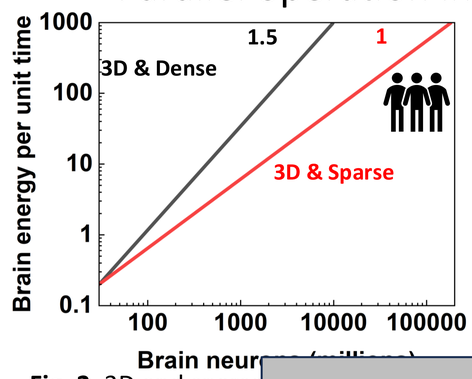


Fig.1. The scaling of parameters and decoder layers in large language model [1]

Run GPT on the edge

- Needs a 15X faster processor and a 170X bigger battery to run GPT-3*
  - High memory density
  - Parallel operation in 3D chip

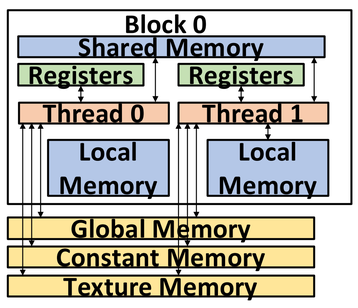
*compared to today's smartphone



- Heat generates as volume: n^3
- Heat dissipates as surface: n^2
  → scales as power 1.5 → overheat → series operation
- Applied pulse to m out of n wires:
- Keep m constant, increase n → activity scales as 1/n
- Heat generates as volume: n^2
- Heat dissipates as surface: n^2
  → scales as power 1 → parallel operation

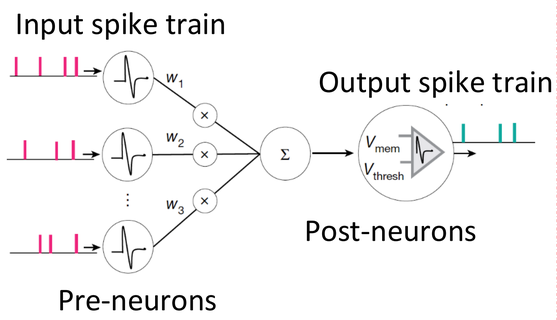
Fig. 2. 3D and sparse

Synaptocentric (GPU) [3]



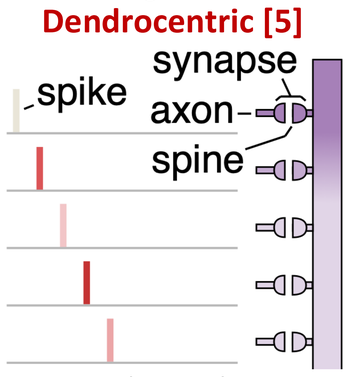
- Matrix multiplication
  - 2D chip
  - Synchronized clock

Axocentric (SNN) [4]



- Time dependent
  - Event driven
  - Run in parallel but overheat

Dendrocentric [5]



- Sequence dependent
  - 3D chip
  - Run in parallel w/o overh

Fig. 3. AI hardware. Synaptocentric, Axocentric, and Dendrocentric

(preliminary unpublished results)

Experimental results

(preliminary unpublished results)

Conclusions & References

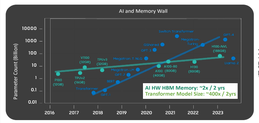
(preliminary unpublished results)

GEMMA: Gain Cell Embedded DRAM-based MAMBA-like Model Accelerator

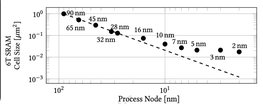
Hery Shin (*hshin24@stanford.edu*), Thierry Tambe (*ttambe@stanford.edu*)

Motivation

1. GCRAM: An Emerging Solution for the Growing On-Chip Memory Demands of ML Applications



[On-chip memory requirement for AI workloads [1]]

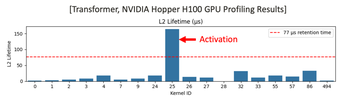


[End of SRAM scaling [2]]

	SRAM	Short-term RAM
Structure	6T	2T or 3T gain cells
Benefits	Fast, easy to integrate, low static power	Dense, low energy
Drawbacks	Sparse	Short retention times, expensive refreshes, active research
Uses	Fast R/W caches	Fast write-and-read operations

[Alternative memory technology]

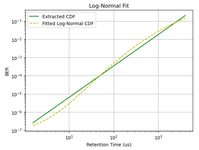
2. Analysis of Kernels Based on Data Lifetime



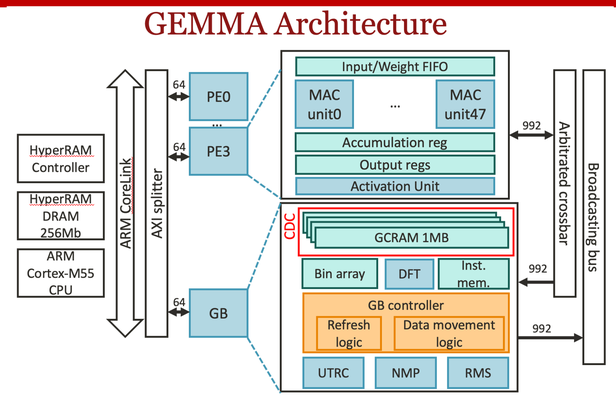
- Kernels such as nonlinear functions, residual layers are categorized as generating long-lived data [3].

3. Worst-case Retention-based Static Refresh Policy

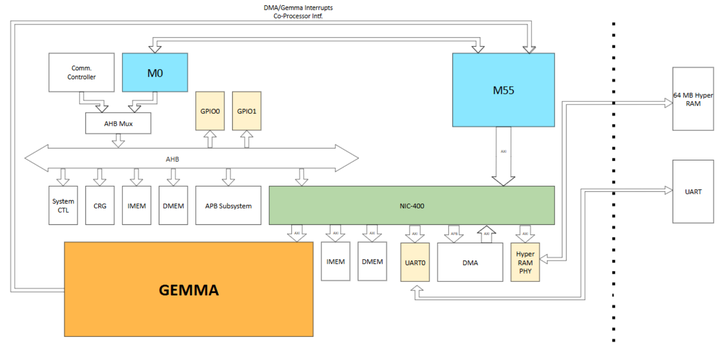
- Over 99% of cells exhibit retention times longer than the worst-case refresh period [4].
- Opportunities to minimize refresh overhead for both long-lived and short-lived data



Overall Architecture



[GEMMA Accelerator Schematic]



[SoC Integration Schematic]

1. SIMD Architecture

- Output stationary GEMV PEs

2. Microscaling-Adaptive Floating Point Hybrid Data Format

- Compress weight and activation to ~5 bits per element

3. Near Memory Processor in GB & Activation Units in PEs

- Reduce the data movement between GB and PEs
- Reduce memory access through kernel fusion

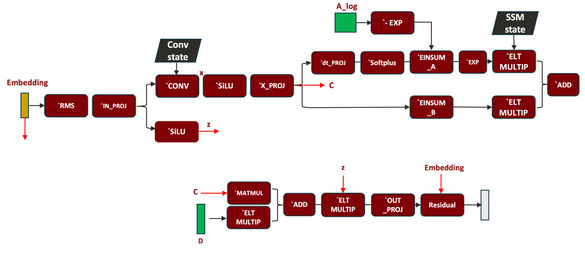
4. Clock Domain Crossing between EDRAM and Accelerator

- Reduce data lifetime of activations

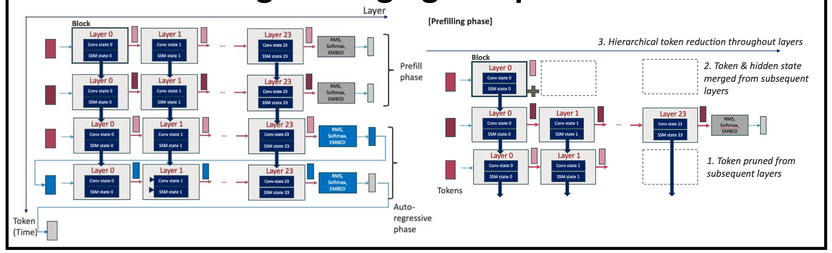
Data Lifetime-Aware HW & SW Co-Design

1. Data Lifetime-Aware Scheduling

- Kernel categorization based on data lifetime



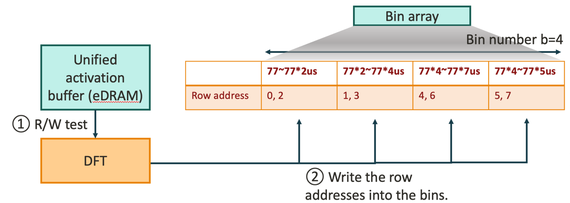
2. Token Pruning & Merging Compression



Fine-grained Refresh Policy

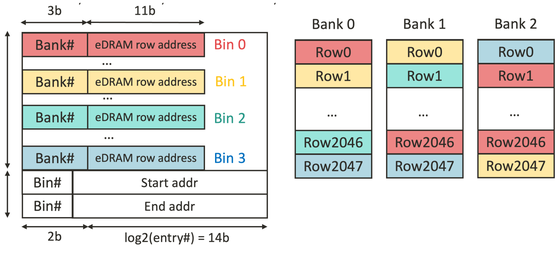
1. Design For Test (DFT)

- Row-wise retention estimation before runtime

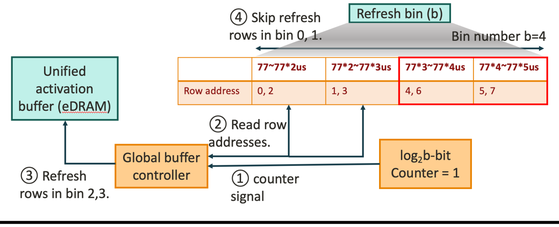


2. Row-wise Refresh With Bin Array

- Virtual-physical address translation book



3. GCRAM Metadata



- Metadata for each GCRAM row: lifetime indicator + error correction code
- Long-lived data mapped onto long retention time rows.
- Short-lived data can skip its refresh or be overwritten after its lifetime.

Conclusions & References

- Seamlessly integrate Si-based 3T gain cell RAM (GCRAM) into state space model accelerators to increase on-chip memory density.
- Challenge of Si-GCRAM: Refresh overhead due to short retention time
- **Data Lifetime minimizing HW&SW co-design**: Various techniques to shorten data lifetime of stored weight/activation by clock domain crossing, scheduling, token compression, etc.
- **Retention-aware Fine-grained Refresh Policy**: Row-wise refresh management reduces refresh overhead and enable lifetime-dependent data mapping in the GCRAM.

[1] A. Gholami, *et al*., "AI and Memory Wall," IEEE MICRO, Mar. 2024.
[2] K. Zhang, "1.1 Semiconductor Industry: Present & Future," in 2024 IEEE International Solid-State Circuits Conference (ISSCC), Feb. 2024.
[3] P. Li, T. Tambe, "GainSight: Profile-Guided Design for Heterogeneous On-Chip Memories in Next-Generation AI Accelerators," IEEE MICRO
[4] RAAAM

GainSight: Application-Guided Profiling for Composing Heterogeneous On-Chip Memories in AI Hardware Accelerators

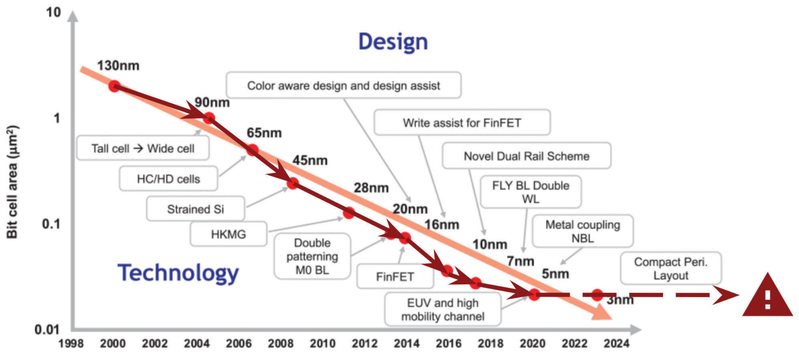
Peijing Li (peli@stanford.edu)**, Thierry Tambe** (ttambe@stanford.edu)
Department of Electrical Engineering, Stanford University, Stanford, CA, United States

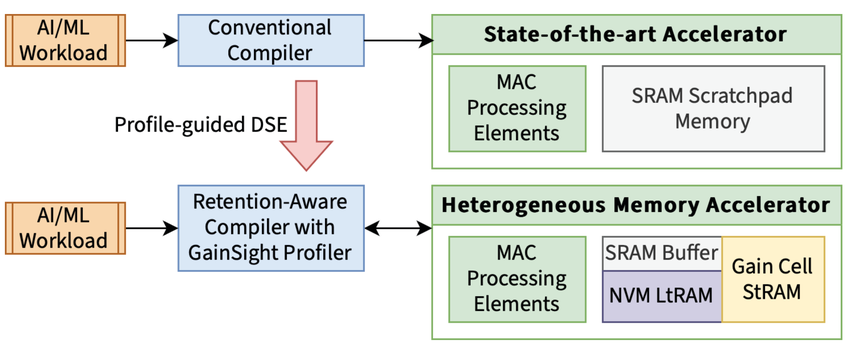
Abstract -- *Visit our homepage on gainsight.stanford.edu!*

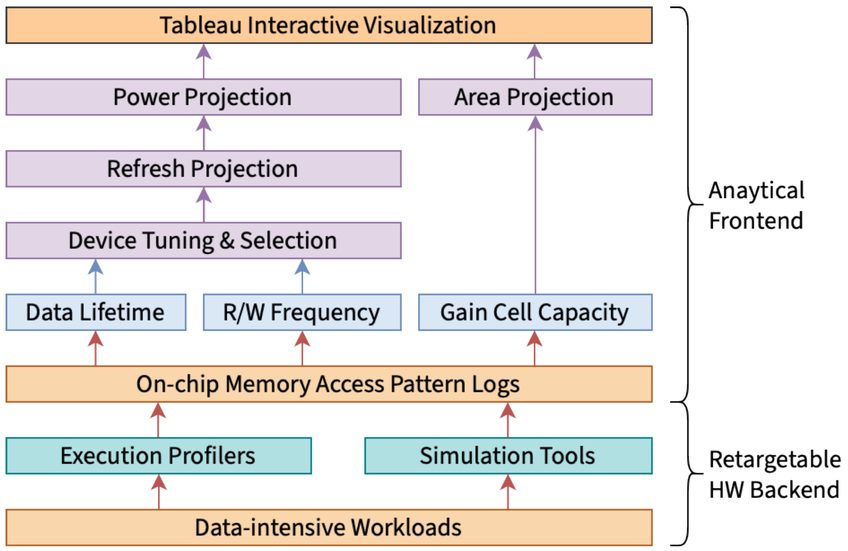
As AI workloads drive soaring memory requirements, there is a need for higher-density on-chip memory for domain-specific accelerators that goes beyond what current SRAM technology can provide. We motivate that algorithms and application behavior should guide the composition of heterogeneous on-chip memories. However, there has been little work in factoring dynamic application profiles into such design decisions. We present GainSight, a profiling framework that analyzes fine-grained memory access patterns and computes data lifetimes in domain-specific accelerators. By combining instrumentation and simulation across retargetable hardware backends, GainSight aligns heterogeneous memory designs with workload-specific traffic and lifetime metrics.

Introduction

- Motivation: the "memory capacity wall" for transformer-based AI models; the stopping of SRAM scaling
- Long and short-lived data call for heterogeneous on-chip memory
- Devices such as gain cell random access memory (GCRAM) are promising candidates for short-lived data, but their short retention times and refresh costs pose challenges.
- Need to quantitatively correlate dynamic data lifetime profiles of workloads with characteristics of long and short-term memory
- **We conceived the GainSight Profiler to tackle these challenges, to analyze the memory access patterns and data lifetimes of software running on domain-specific accelerators, in order to inform the design of heterogeneous on-chip memories.**







Retargetable Hardware Backend

- Acquire fine-grained, on-chip memory access patterns for a physical or simulated architecture
- Currently implemented simulators for NVIDIA H100 GPUs (AccelSim) and systolic arrays (SCALE-Sim-v2)
- Exploit workload sampling to reduce program sizes to speed up cycle-accurate simulation

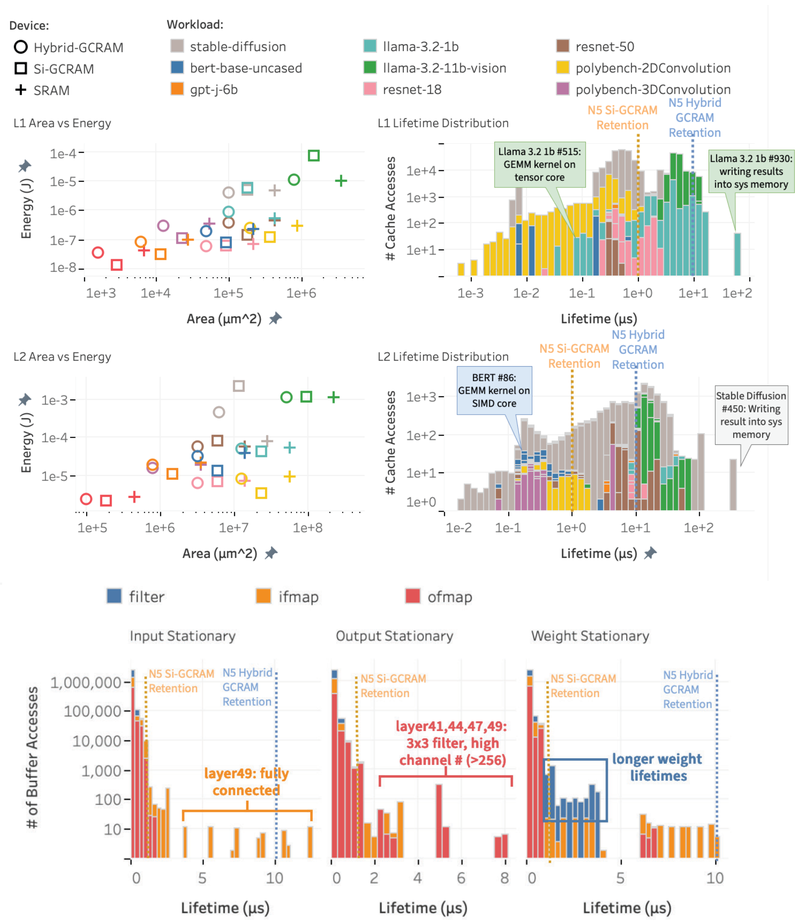
Analytical Frontend

- Consume memory traces generated by backend
- Calculate address-level data lifetimes, read/write frquencies, and on-chip memory capacity utilization
- Visualize lifetime distribution and compare with silicon gain cell retention times in Tableau
- Project number of refreshes, area, and total energy using calculated memory statistics for devices of SRAM, fully Si-Si GCRAM, and hybrid Si-ITO GCRAM.

Case Study

Insights from profiling MLPerf Inference and PolyBench workloads using simulated GPUs and systolic arrays:

1. 40% of L1 and 18% of L2 GPU cache accesses, and 79% of systolic array scratchpad accesses across profiled workloads are short-lived and suitable for Si-GCRAM
2. Si-GCRAM reduces active energy by 11-28% compared to SRAM;
3. Up to 90% of GPU cache fetches are never reused, highlighting inefficiencies in terms of cache pollution.



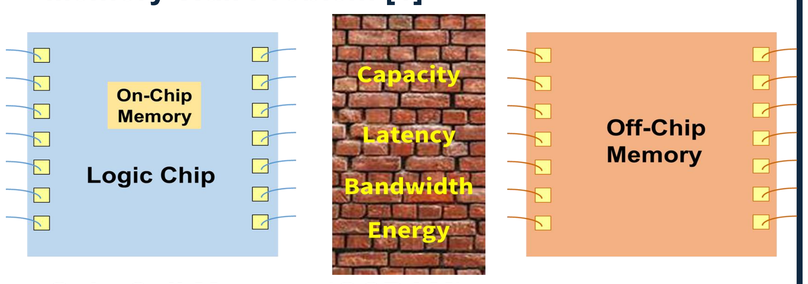
OpenGCRAM: An Open-Source Gain Cell Compiler Enabling Design-Space Exploration for AI Workloads

Xinxin Wang[1], Lixian Yan[1], Shuhan Liu[1], Luke Upton[1], Zhuoqi Cai[1], Yiming Tan[1], Shengman Li[1], Peijing Li[1], Jesse Cirimelli-Low[2], Thierry Tambe[1], Matthew Guthaus[2], and H.-S. Philip Wong[1]

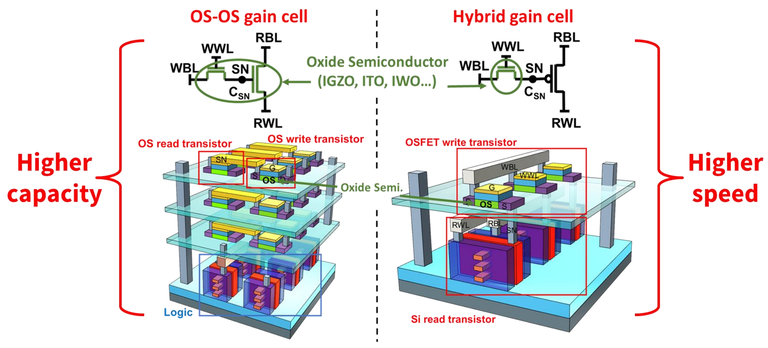
[1]Stanford University [2]University of California, Santa Cruz

I. Motivation:

- **Memory Wall Problem [1]**



- **Gain Cell Memory (GCRAM)**
  - ***vs SRAM: High density, high bandwidth, ultra-low leakage [2]***

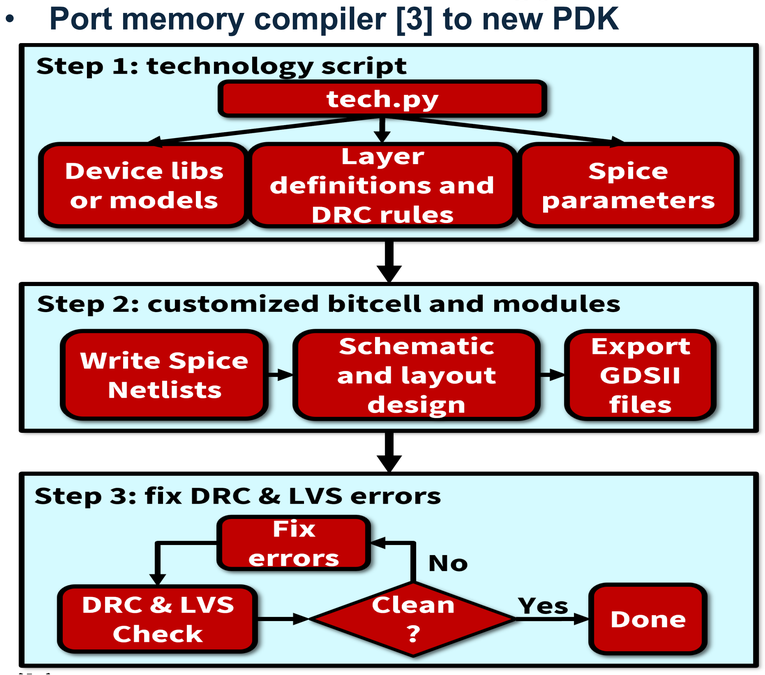


To enable fast, accurate, customizable, optimized Gain Cell bank generation and performance simulation targeting for AI workloads:

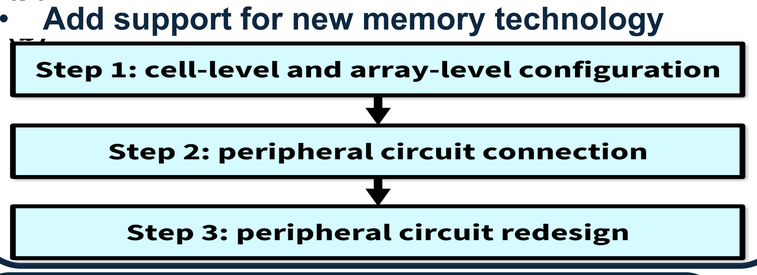
We need a Gain Cell compiler

II. Methodology:

- **Port memory compiler [3] to new PDK**

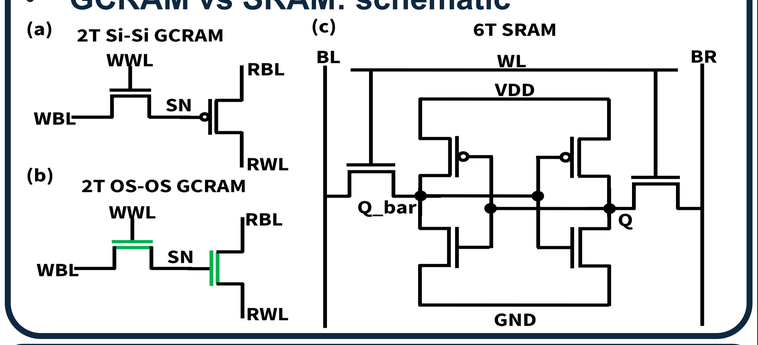


- **Add support for new memory technology**



III. OpenGCRAM

- **GCRAM vs SRAM: schematic**

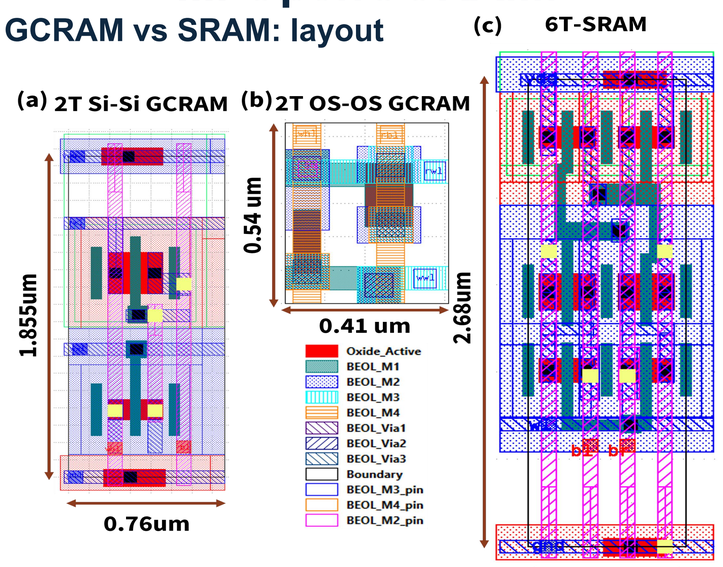


Acknowledgement: This work was supported in part by SRC JUMP 2.0 PRISM and CHIMES centers, Stanford SystemX Alliance, DAM project, and NMTRI.

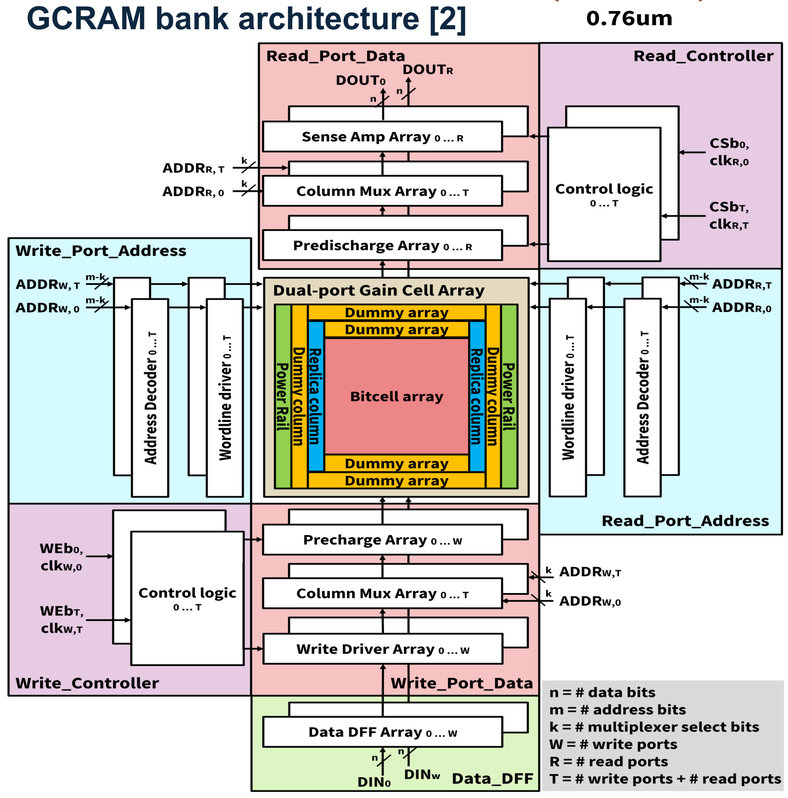
References: [1] A. Gholami, Micro 2024; [2] S. Liu, IEDM 2023; [3] M. R. Guthaus, et al., ICCAD 2016; [4] P. Li, et al., arXiv:2504.14866

III. OpenGCRAM:

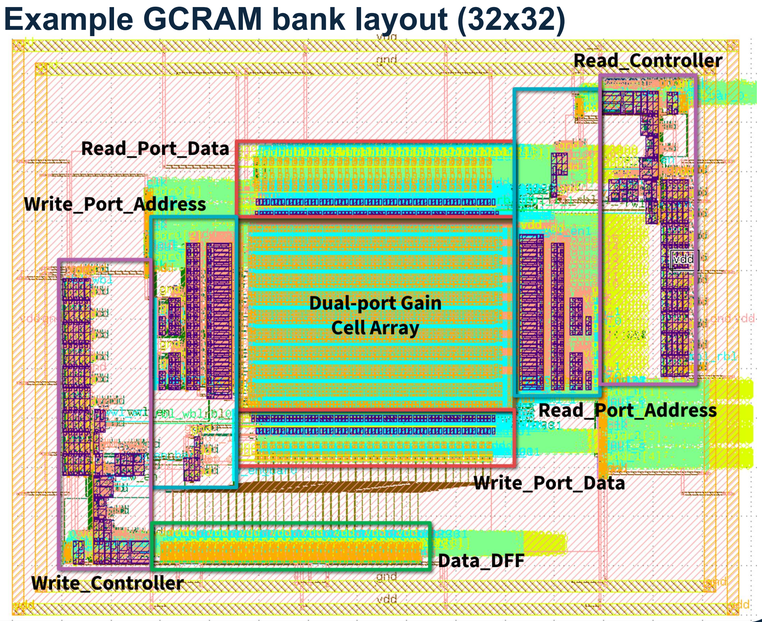
- **GCRAM vs SRAM: layout**



- **GCRAM bank architecture [2]**

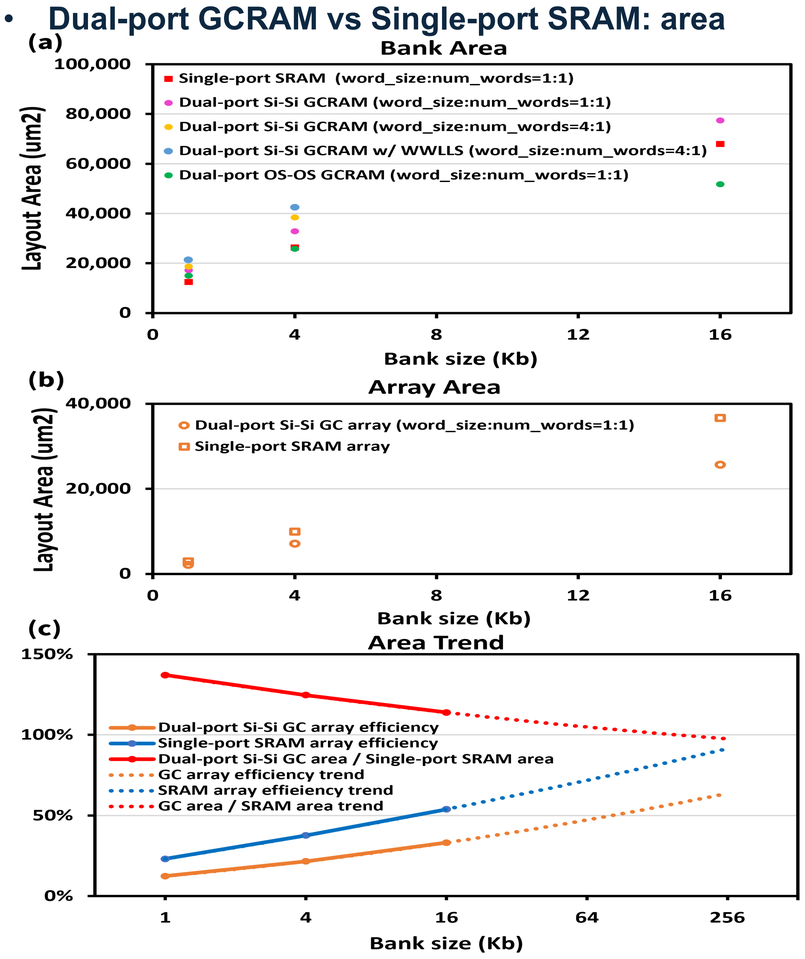


- **Example GCRAM bank layout (32x32)**

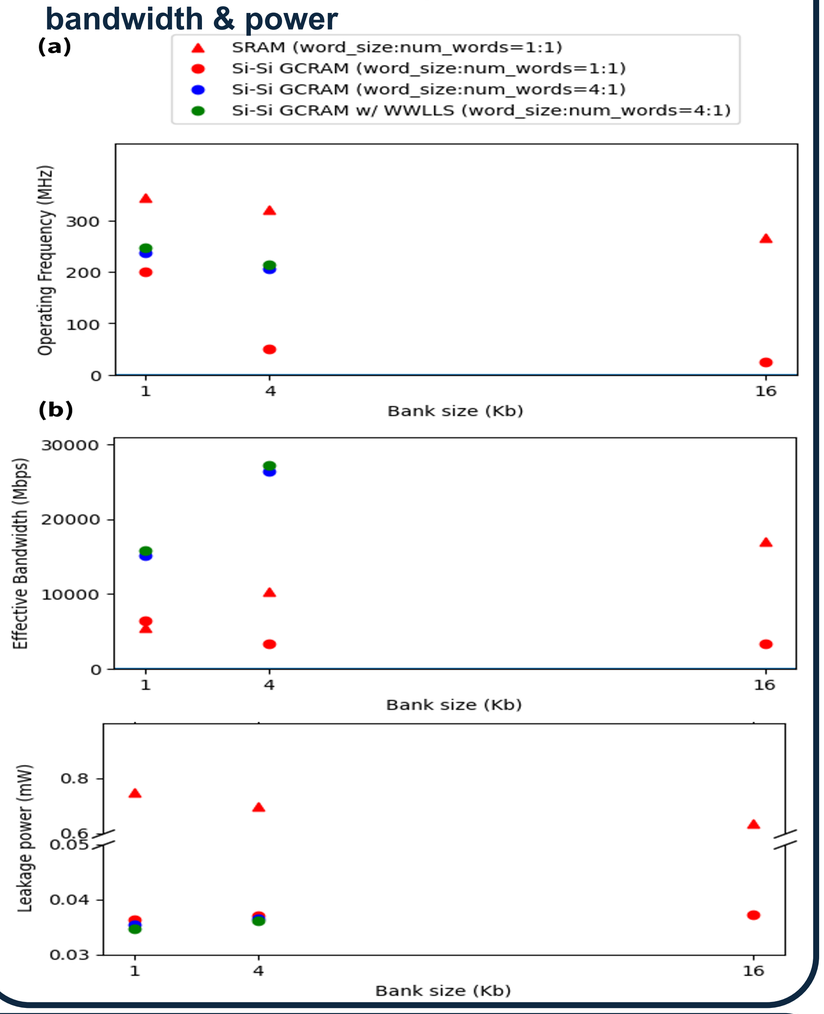


III. Results:

- **Dual-port GCRAM vs Single-port SRAM: area**

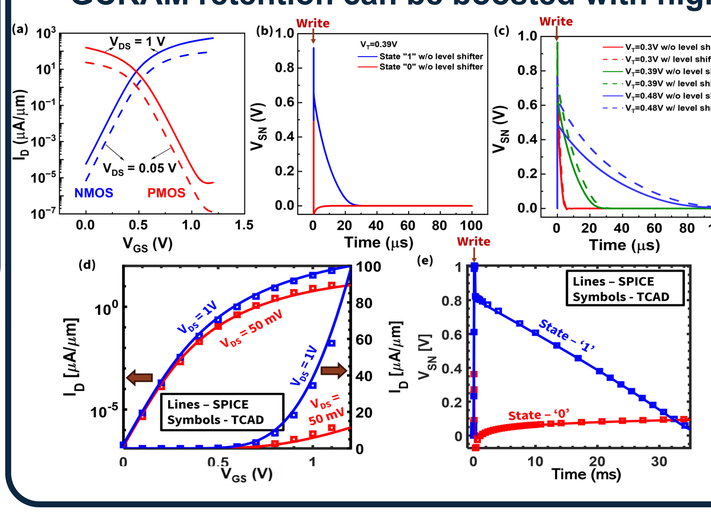


- **Dual-port GCRAM vs Single-port SRAM: freq, bandwidth & power**

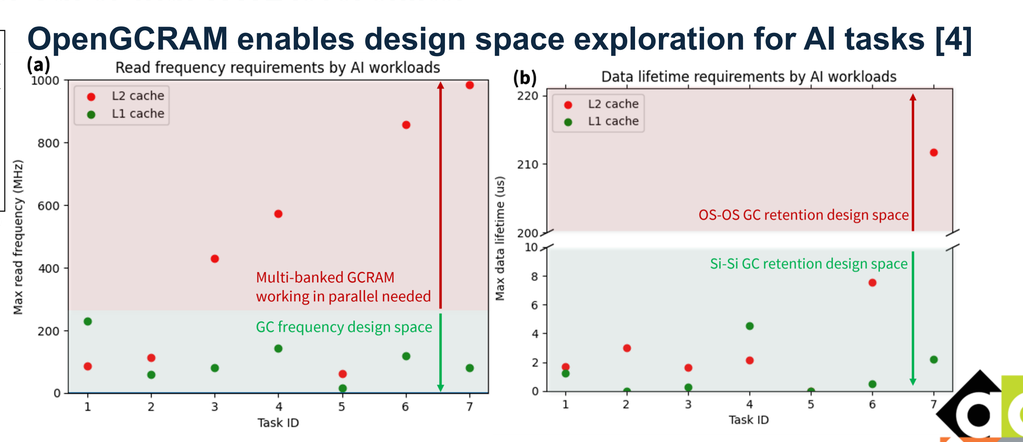


IV. Design Space Exploration:

- **GCRAM retention can be boosted with higher Vth or with WWL level shifter**



- **OpenGCRAM enables design space exploration for AI tasks [4]**



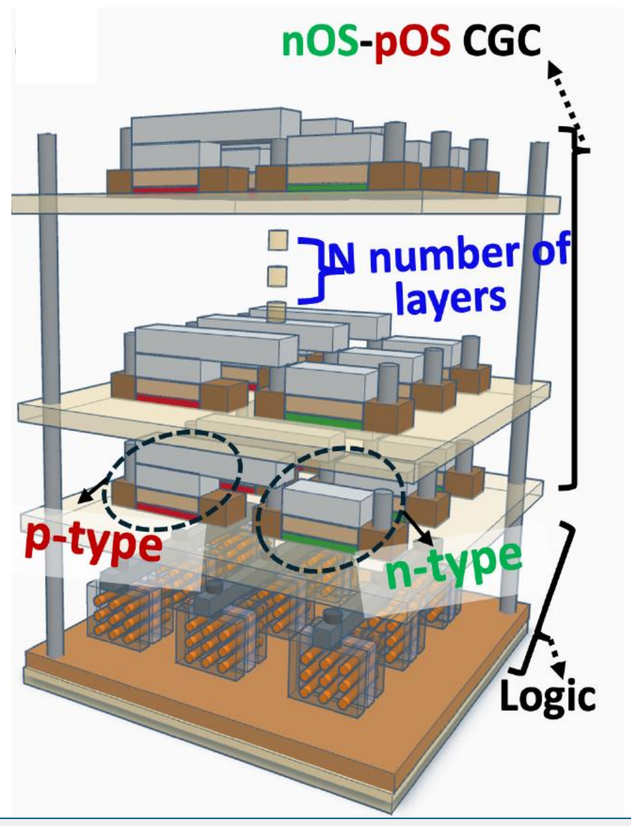
N-P Oxide Semiconductor Complementary Gain Cell (CGC) Memory

F. F. Athena, E. Ambrosi, K. Jana, Ch. H. Wu, J. Hartono, Y. M. Lee, C. C. Kuo, S. Liu, B. Saini, C. C. Wang, C. F. Hsu, G. Zeevi, X. Wang, J. Kang, E. Pop, T. Y. Lee, P. C. McIntyre, H.-S. P. Wong, X. Y. Bao

(contact: fathena@stanford.edu) PI: H.-S. Philip Wong

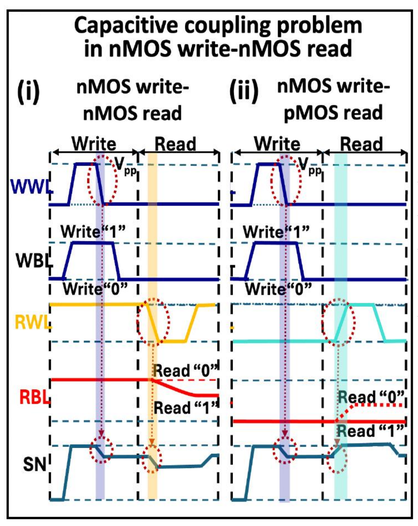
Objectives

- Complementary GC (CGC) , with its n- and p-type OS configuration, results in high density, good retention, CMOS compatibility, low capacitive coupling, and adequate read speeds
- CGC with complementary back-end-of-line technology enables future monolithic multiple layers 3D integration



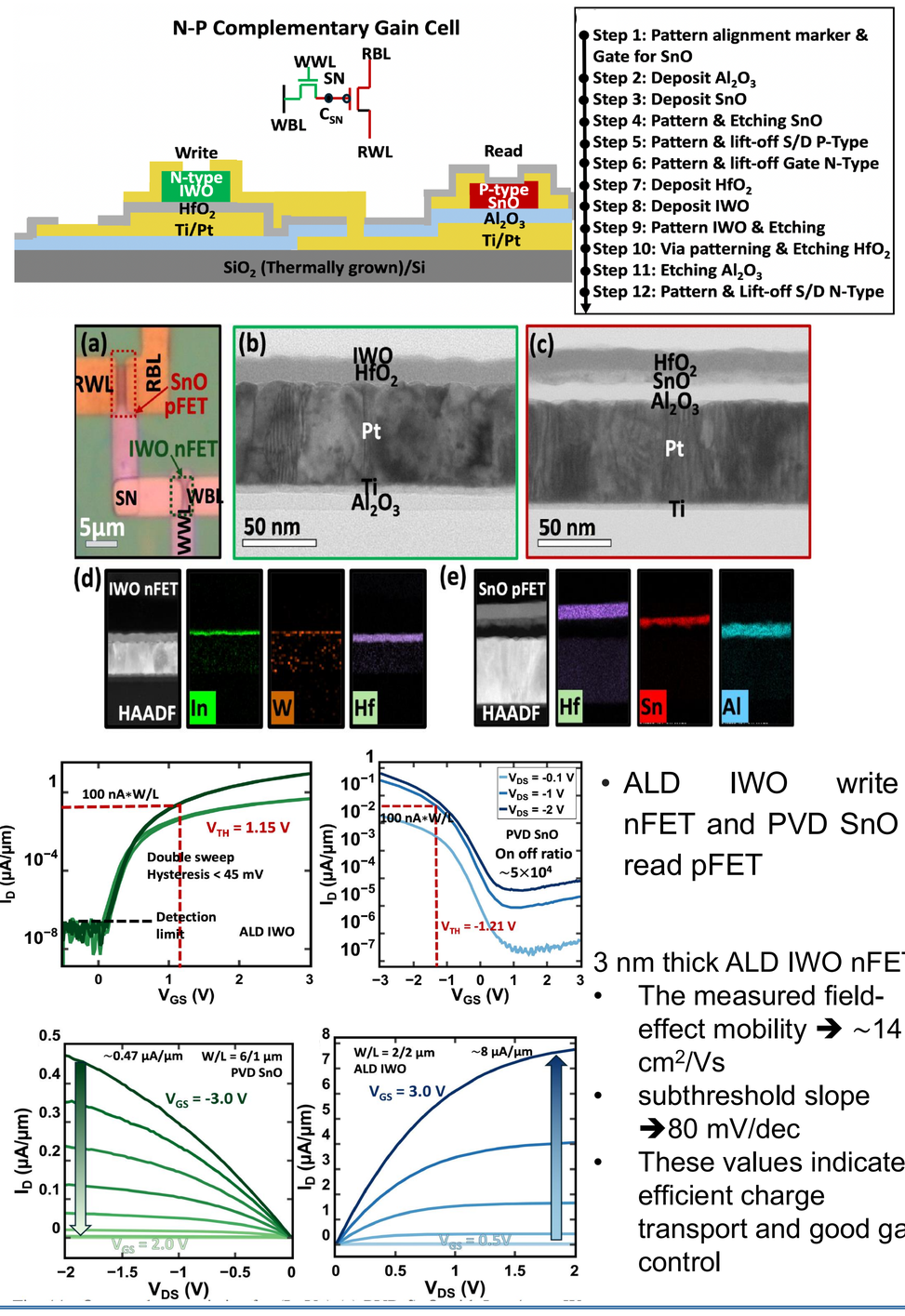
GC Comparison

nOS-nOS	nOS-pSi (HGC)	nOS-pOS (CGC)
✓ CMOS compatible ✓ Monolithic (N-layer) integration × Issue with capacitive coupling × Poor speed-retention tradeoff	✓ CMOS compatible × FEOL Si FET for read ➔ multiple memory layers not possible ✓ No capacitive coupling problem ✓ Better speed-retention tradeoff	✓ CMOS compatible ✓ Monolithic multiple-layer integration possible ✓ Good p-OSFET performance is needed ✓ No capacitive coupling problem ✓ Better speed-retention tradeoff



- ✓CMOS Compatible
- ✓Monolithic 3D
- ✓No capacitive coupling
- ✓Better speed-retention tradeoff

Technical Approach

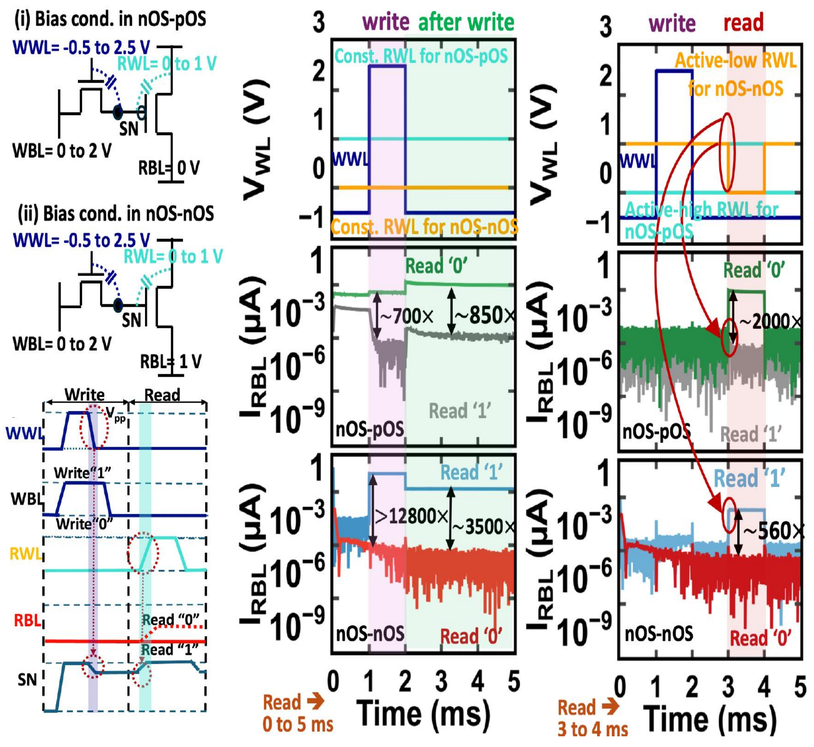


- ALD IWO write nFET and PVD SnO read pFET

3 nm thick ALD IWO nFET

- The measured field-effect mobility ➔ ~14 cm^2/Vs
- subthreshold slope ➔80 mV/dec
- These values indicate efficient charge transport and good gate control

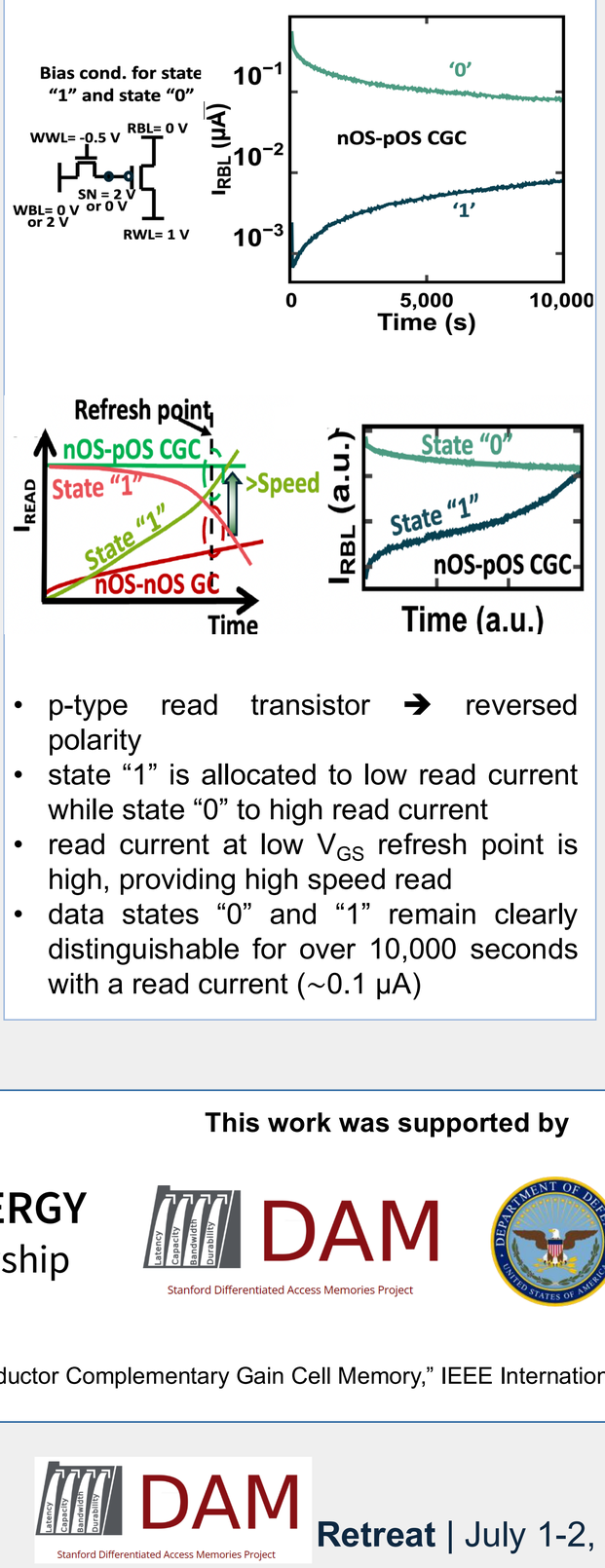
Counterbalanced Capacitive Coupling in CGC



RWL-SN capacitive coupling

- causes further degradation in nOS-nOS GC
- improves/ retains the margin between state "0" and state "1" for nOS-pOS CGC (performing read only from 3 ms to 4 ms)

Optimized State Allocation



- p-type read transistor ➔ reversed polarity
- state "1" is allocated to low read current while state "0" to high read current
- read current at low V_{GS} refresh point is high, providing high speed read
- data states "0" and "1" remain clearly distinguishable for over 10,000 seconds with a read current (~0.1 µA)

Key Accomplishments, Showstoppers & Next Steps

- **Key results and metrics**:
  - Counteracting capacitive coupling
  - P-type read enhances voltage sensing speed and sense margin
  - Better speed-retention tradeoff
  - 10,000 s retention, with state '0' as a high-current state
- **Grand challenge applications**:
  - M3D integrated high-density on-chip memory
- **Next Steps**:
  - Improve SS for p-type OS
  - Variation control
  - Multi-layer integration and scalability to advanced nodes

This work was supported by

Stanford | ENERGY Precourt Institute for Energy

Stanford | ENERGY Postdoctoral Fellowship

DAM Stanford Differentiated Access Memories Project

Stanford | Non-Volatile Memory Technology Research Initiative (NMTRI)

Stanford | SystemX Alliance

CHIMES Center for Heterogeneous Integration of Micro Electronic Systems

Publications: F. F. Athena et al., "First Demonstration of an N-P Oxide Semiconductor Complementary Gain Cell Memory," IEEE International Electron Devices Meeting, 2024 (SRC ID: 397536)

Conserving Memory Bandwidth in SmartNICs with Flow-Based Memory Addressing

Agur Adams, Colin Drewes, David Shim, and Ben O'Keefe
Advised by Phil Levis and Zakir Durumeric

Problem

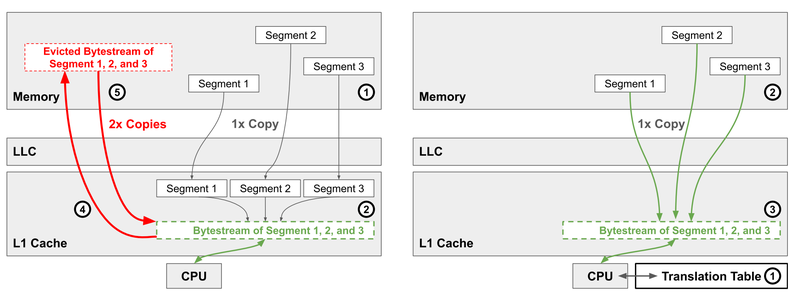
SmartNICs cannot easily perform complex network analysis at line rate due to limited memory bandwidth.

Network line rates have significantly increased:
- Increased speeds has led to **reduced visibility**
- High traffic volumes **overwhelms** most analysis tools

Complex analysis at line rates ≥ 100 Gbps requires on-NIC processing:
- Switches and routers support only simple analysis
- Specialized hardware can be costly or hard to program

SmartNICs have limited memory bandwidth relative to their line rates:
- Copying data to reassemble byte streams is expensive
- Careful cache management required

	Cores	NIC	DRAM	DRAM BW per core	NIC BW per core	Ratio
Google Cloud C3 Sapphire Rapids	176	200 Gbps	2x 8-ch DDR5	3.49 GB/s	0.14 GB/s	**24.93**
BlueField-3 SmartNIC DDR5	16	400 Gbps	2-ch DDR5	5.60 GB/s	3.13 GB/s	**1.79**



Goals

We aim to create a SmartNIC packet processing pipeline capable of arbitrary network analysis on reassembled TCP bytestreams at 200 Gbps using flow-based addressing.

adamsas@stanford.edu

Solution

Using **flow-based addressing**, a SmartNIC can make the payloads of a series of packets addressable as a contiguous region of memory without a copy.

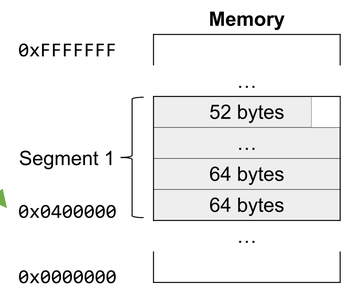
The SmartNIC populates and manages a translation table of all transport-layer segments of a flow.

Example: 4015 bytes received in three TCP segments (assuming a maximum segment size of 1460 bytes)

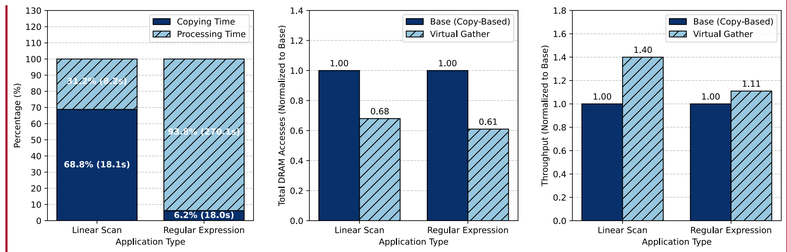
Translation Table

Addr	Len	Off
0x0400000	0x5B4	0x5B4
0x0C00000	0x5B4	0xB68
0x1000000	0x447	**0xFAF**

Last Off	0xB68	
+	Len	0x447
New Off		**0xFAF**



Impact

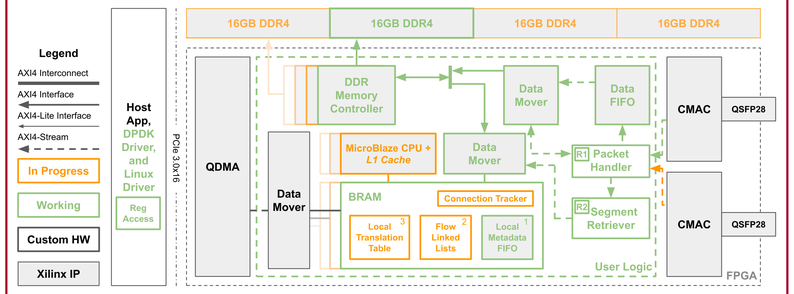


Preliminary Results: To quantify the cost of copying overhead on packet processing and the benefits of our optimization, we emulated the translation table in software and measured the time spent on copying versus application processing in two network applications: *linear scan* and *regular expression* matching.

- DRAM access reduction: 32.2% (linear) / 39.1% (regex)
- Throughput increase: 40% (linear) / 11% (regex)

Design

We are building out the proposed flow-based addressing scheme on an AMD Alveo U250 FPGA-based SmartNIC. Key aspects of the design include:
- Large number of RISC-V CPUs in parallel for flexible network analysis
- TCP connection tracking for flow-level awareness and timeouts
- Linked-list assembly for on-demand memory allocation



Challenges

How do you insert and manage the table at line rate (e.g., fragmentation, garbage collection, etc.)?
- We observed approximately 22,000 new flows a second (~145 Gbps) on the Stanford Campus network.
- We estimate ~32 ns between packets at 200 Gbps.

How large do you make the table?
- ~80% of all flows on the Stanford Campus network are less than 1100 bytes long, which means a translation table with just 18 entries could support most flows.

Should you use virtual or physical addresses in the table?
- We use physical addresses in our FPGA implementation but are considering both options.

How does the translation table interact with the cache?
- Could use the translation table to prefetch for the SmartNIC CPU's cache