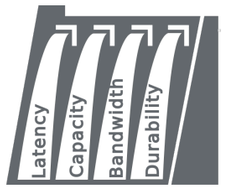
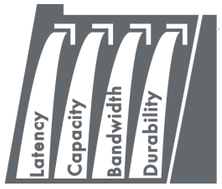
DAM: Differentiated Access Memory

Philip Levis and Caroline Trippel
DAM/MemoryDAX Second Winter Workshop
January 21, 2026



DAM



MemoryDAX

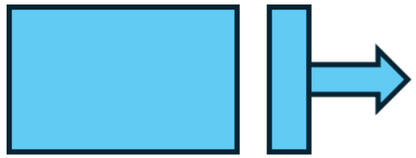
In One Slide

- Year 2 of a 7 year research project to overcome current computing limitations through heterogeneous memories
- Memory is the limiting factor in computing today
  - Significant performance, energy, and cost improvements require transformative changes to memory
  - Memory needs to specialize: differentiated access memories (DAM)
- Differentiated access memories raise many open research questions
  - Which application patterns can leverage differentiation?
  - How will software use the memories (explicit vs. implicit placement)?
  - Which memories and tradeoffs (energy, density, latency, retention, endurance, etc.)?
  - How will these memories be packaged and composed in larger systems?

Application Needs Vary Greatly

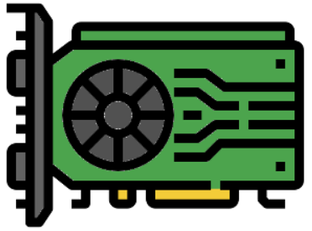
Data Analytics

Streams of data
Write-once, read-once
Filters (scans)
Joins (random access)



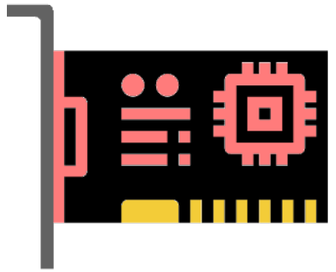
Append-Mostly Databases

Write once
Mostly append
Read many times
Scans
Random access



Machine Learning Accelerator

Blocked operations
Sparse accesses
Read multiple times
Write many times
Read/write
Throughput (training)
Latency (inference)



High-Speed Networking

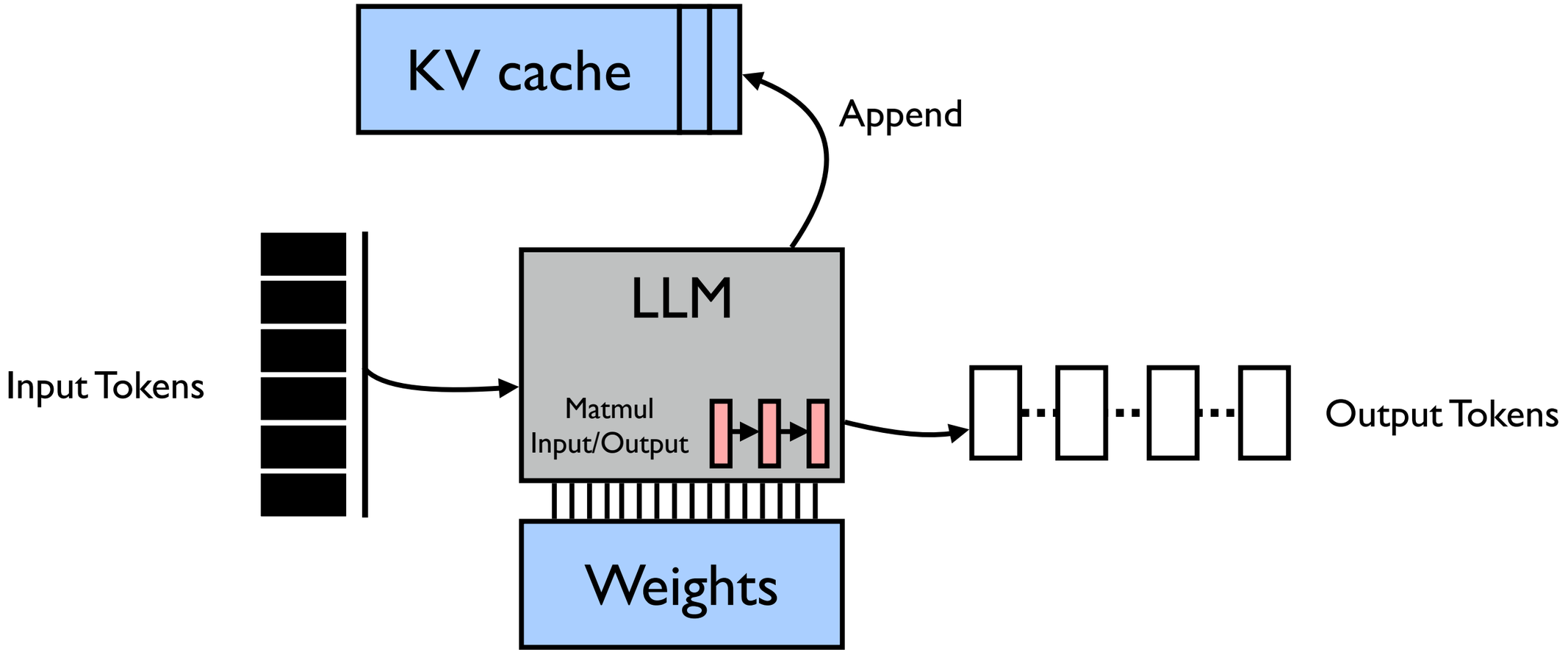
Ultra-low latency
Header processing
Packet-oriented
Read once
Write once

Actually Many Kinds of Memory...

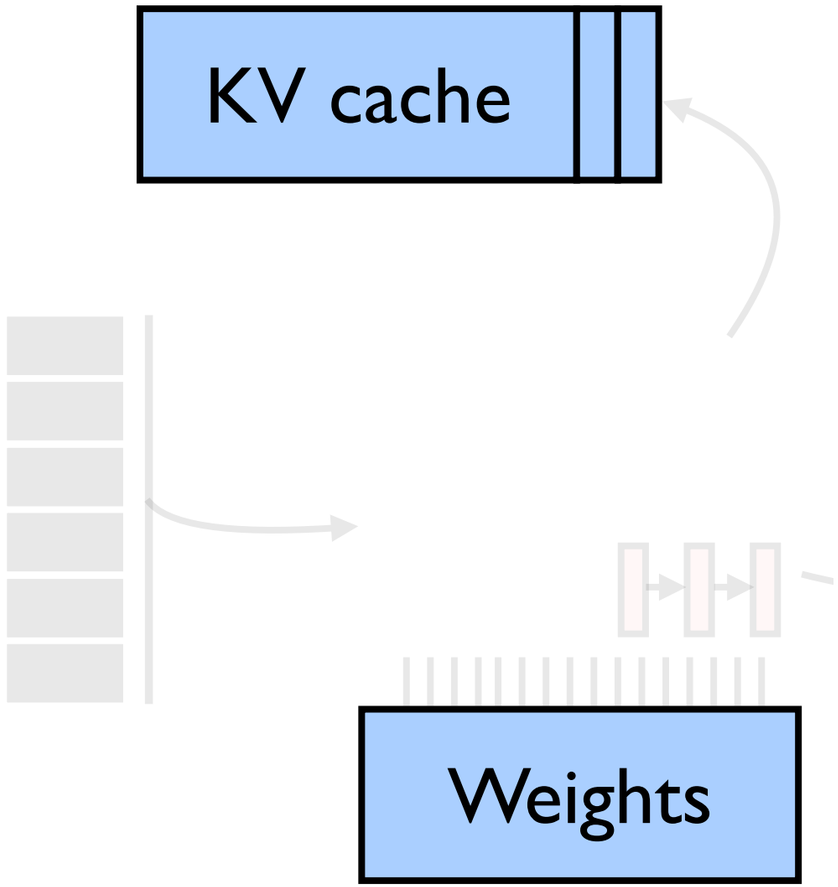
DRAM
SRAM
MRAM
RRAM
FRAM
PCM
Flash
GC
HGC
FeFet
OS-OS

	Energy/power (active)		Energy/ power (standby)	Access time, latency		endurance	retention	Density (capacity)	On-logic chip integration	
	read	Write		read	Write				One layer (possible)	Multiple layers for density
High	RRAM, MRAM, PCM, FeRAM,	RRAM, MRAM, PCM, Flash	DRAM	Flash	Flash	DRAM, SRAM, OS-OS GC, HGC	Flash, RRAM, MRAM, PCM, FeFET, FeRAM	Flash, FeFET	MRAM, PCM, RRAM, FeRAM,	FeFET, OS-OS GC
Medium	DRAM	DRAM, FeRAM	SRAM	RRAM, PCM, FeFET, FeRAM	RRAM, PCM, FeFET, FeRAM	FeRAM, MRAM	OS-OS GC, HGC	DRAM, FeRAM, OS-OS GC	DRAM	
Medium low	FeFET, OS-OS GC	FeFET	HGC, OS-OS GC	DRAM, MRAM, OS-OS GC	DRAM, OS-OS GC, HGC	PCM, RRAM	DRAM	HGC, MRAM, RRAM, PCM,		
low	SRAM, HGC	SRAM, HGC, OS-OS GC	RRAM, MRAM, PCM, FeFET, FeRAM, Flash	SRAM, HGC	SRAM	Flash, FeFET		SRAM	Flash	Flash, DRAM

Zooming In: LLM Inference

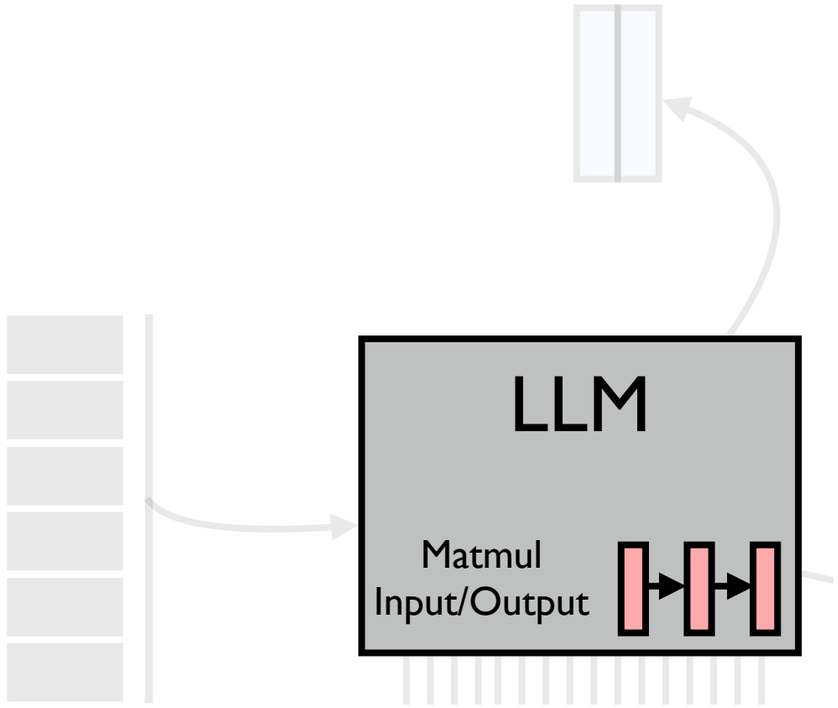


Write Rarely, Read Often



- Both the KV cache and model weights are read heavy
  - Read the cache N times for N output tokens, append cache entries
  - Write parameters on model load the model, read each time it executes

Write Once, Read Once



- Activations: inputs and outputs to matrix multiplications
  - Write the values once
  - Quickly read them once

Outcome of Year One

- We can distill many common software patterns and memory types into two broad classes
- Application data
  - Written very rarely: read-only (code, inference model weights), read-mostly (network data stores, databases, KV caches), "cold" data,
  - Written and read very frequently: caches, scratchpads, inference activations, computational kernel scratchpads,
- Memory types
  - Non-volatile memories: expensive and limited writes, but higher density, lower read energy, and high retention
  - Dynamic memories: require refreshes if not accessed, but higher density, lower read and write energies

Five Types of Memory

Structure

Benefits

Drawbacks

Uses

SRAM

	SRAM
Structure	6T
Benefits	Fast Easy to integrate Low static power
Drawbacks	Sparse
Uses	Fast read/write caches

DRAM

	SRAM	DRAM
Structure	6T	1T1C
Benefits	Fast Easy to integrate Low static power	Dense
Drawbacks	Sparse	Hard to integrate High power
Uses	Fast read/write caches	Large, random-access RW data

Block Flash

	SRAM	DRAM	Block Flash
Structure	6T	1T1C	1G
Benefits	Fast Easy to integrate Low static power	Dense	HUGE Capacity
Drawbacks	Sparse	Hard to integrate High power	No logic Low endurance Expensive, slow erases Block access Low bandwidth
Uses	Fast read/write caches	Large, random-access RW data	Large, read-mostly data

Long-term RAM (LtRAM)

	SRAM	DRAM	Block Flash	LtRAM (long-term RAM)
Structure	6T	1T1C	1G	FeRAM, MRAM, RRAM
Benefits	Fast Easy to integrate Low static power	Dense	HUGE Capacity	Dense Low Read Energy
Drawbacks	Sparse	Hard to integrate High power	No logic Low endurance Expensive, slow erases Block access Low bandwidth	Writes are slow and high energy Limited endurance
Uses	Fast read/write caches	Large, random-access RW data	Large, read-mostly data	Write rarely

Short-term RAM (StRAM)

	SRAM	DRAM	Block Flash	LtRAM (long-term RAM)	StRAM (short-term RAM)
Structure	6T	1T1C	1G	FeRAM, MRAM, RRAM	Gain Cells (2T, 3T)
Benefits	Fast Easy to integrate Low static power	Dense	HUGE Capacity	Dense Low Read Energy	Dense Low Energy
Drawbacks	Sparse	Hard to integrate High power	No logic Low endurance Expensive, slow erases Block access Low bandwidth	Writes are slow and high energy Limited endurance	Active research Refresh power
Uses	Fast read/write caches	Large, random-access RW data	Large, read-mostly data	Write rarely	Write-and-read

LtRAM: Long-term RAM

- Stores data for seconds to days
- High read:write ratio
- Lower read energy than DRAM, higher write energy
- Often non-volatile
- Server uses: copy-on-write memory caches, code pages, cold memory
- Inference uses: model weights, KV caches
- Example technologies: Flash, MRAM, RRAM, FeRAM, 3DXP
- On-die, in-package, or off-package with compute

StRAM: Short-term RAM

- Stores data for microseconds to seconds
- Write:read ratio $\approx 1:1$
- Higher density than SRAM
- Lower write energy than SRAM, tunable retention
- Server uses: on-CPU caches, DMA memory, queues and buffers
- Inference uses: model activations, program variables
- Technology: gain-cell RAM (GCRAM)
- Integrated on-die with compute

Outcome of Year One

- We can distill many common software patterns and memory types into two broad classes
- Long-term RAM (LtRAM) for written-rarely, long-lived data
- Short-term RAM (StRAM) for frequently accessed data

Agenda in Year Two: Develop LtRAM and StRAM

Developing LtRAM and StRAM

- How do we build StRAM and LtRAM?
  - Today: talk by Shuhan Liu on "Gain Cells and StRAM: Past, Present, and Future"
  - Today: talk by Xinxin Wang on "Gain Cell Compiler"

Developing LtRAM and StRAM

- How do we build StRAM and LtRAM?
  - Today: talk by Shuhan Liu on "Gain Cells and StRAM: Past, Present, and Future"
  - Today: talk by Xinxin Wang on "Gain Cell Compiler"
- How do we integrate StRAM and LtRAM with processors and use it well?
  - Today: talk by Sam Dayo on "LtRAM in AI Accelerators"
  - Today: talk by Colin Drewes on "Optimal 3D Integration with MIQP"

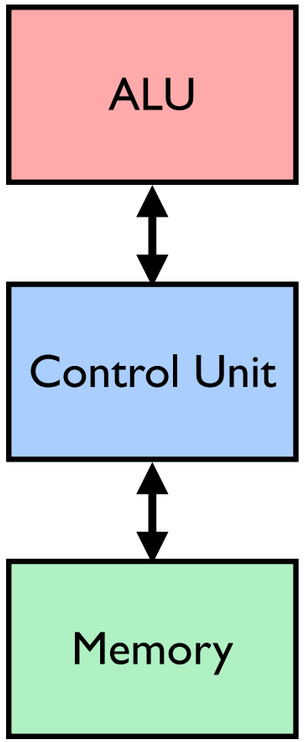
Developing LtRAM and StRAM

- How do we build StRAM and LtRAM?
  - Today: talk by Shuhan Liu on "Gain Cells and StRAM: Past, Present, and Future"
  - Today: talk by Xinxin Wang on "Gain Cell Compiler"
- How do we integrate StRAM and LtRAM with processors and use it well?
  - Today: talk by Sam Dayo on "LtRAM in AI Accelerators"
  - Today: talk by Colin Drewes on "Optimal 3D Integration with MIQP"
- How do provision and allocate different types of memory?
  - Today: talk by Thierry Tambe on "Memory and Data Lifetimes"

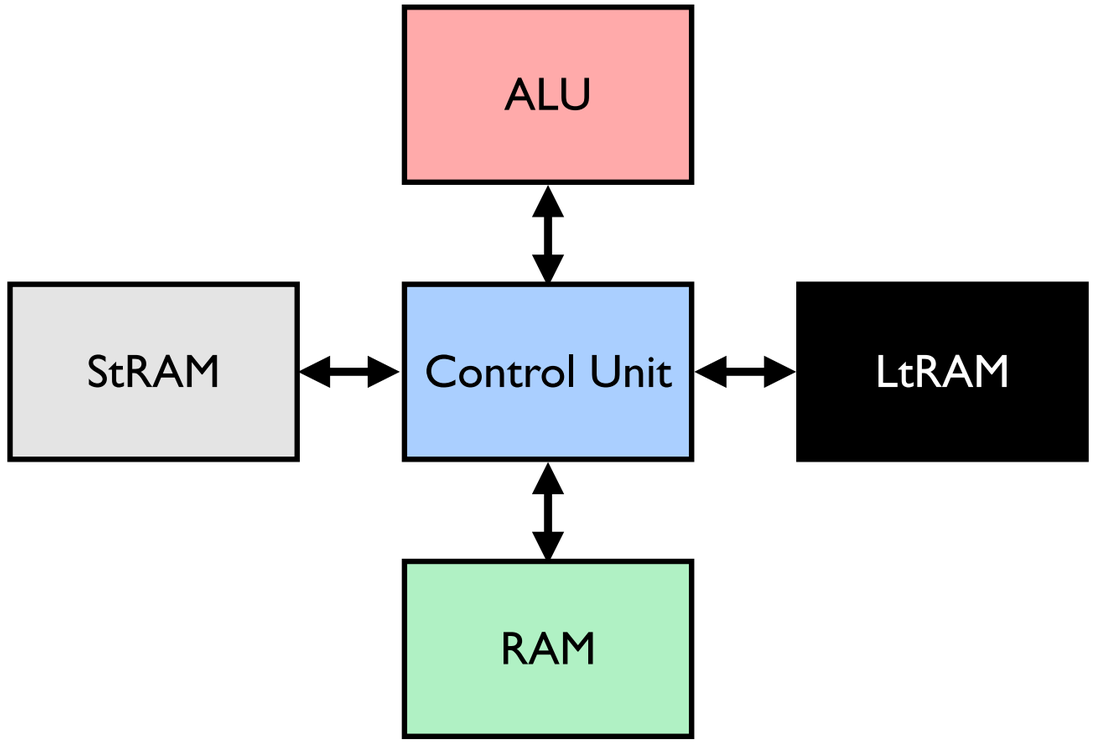
Developing LtRAM and StRAM

- How do we build StRAM and LtRAM?
  - Today: talk by Shuhan Liu on "Gain Cells and StRAM: Past, Present, and Future"
  - Today: talk by Xinxin Wang on "Gain Cell Compiler"
- How do we integrate StRAM and LtRAM with processors and use it well?
  - Today: talk by Sam Dayo on "LtRAM in AI Accelerators"
  - Today: talk by Colin Drewes on "Optimal 3D Integration with MIQP"
- How do provision and allocate different types of memory?
  - Today: talk by Thierry Tambe on "Memory and Data Lifetimes"
- How should hardware support different types and uses of memory?
  - Today: talk by Chun Deng on "Hardware Support for Server Memory Placement"
  - Today: talk by Agur Adams on "Packets are not Pages: Flow-Based Addressing Conserves Memory Bandwidth"

Where We Are Going



von Neumann
Architecture



DAM
Architecture

Today's Schedule

9:00	Project Review: Philip Levis and Caroline Trippel
9:15	Memory and Data Lifetimes: Thierry Tambe
9:45	GainCells and StRAM: Past, Present, and Future: Shuhan Liu
10:15	LtRAM in AI Accelerators: Sam Dayo
10:45	Break
11:00	Optimal 3D Integration with MIQP: Colin Drewes
11:30	Hardware Support for Server Memory Placement: Chun Deng
12:00	Cantor then Lunch
13:30	Challenges on the Way to AGI: John Hennessy
14:30	Panel on Leading Challenges with Memory: Industrial Guests
15:30	High Density Pixel Circuit Design with Gain Cell Memory: Xinxin Wang
15:45	Packets are Not Pages: Flow-Based Addressing Conserves Memory Bandwidth: Agur Adams
16:00	Differentiated Access Memories 2026: Everyone
16:30	Social Discussion

Thank You!