

Big Data Feeds Big Compute through Memory

Siamak Tavallaei

Sr. Principal Engineer

Samsung Semiconductor Inc. (SSI)

DSA, PPT, Systems Architecture



AI LEADERSHIP
FORUM

September 2026

Copyright © 2026

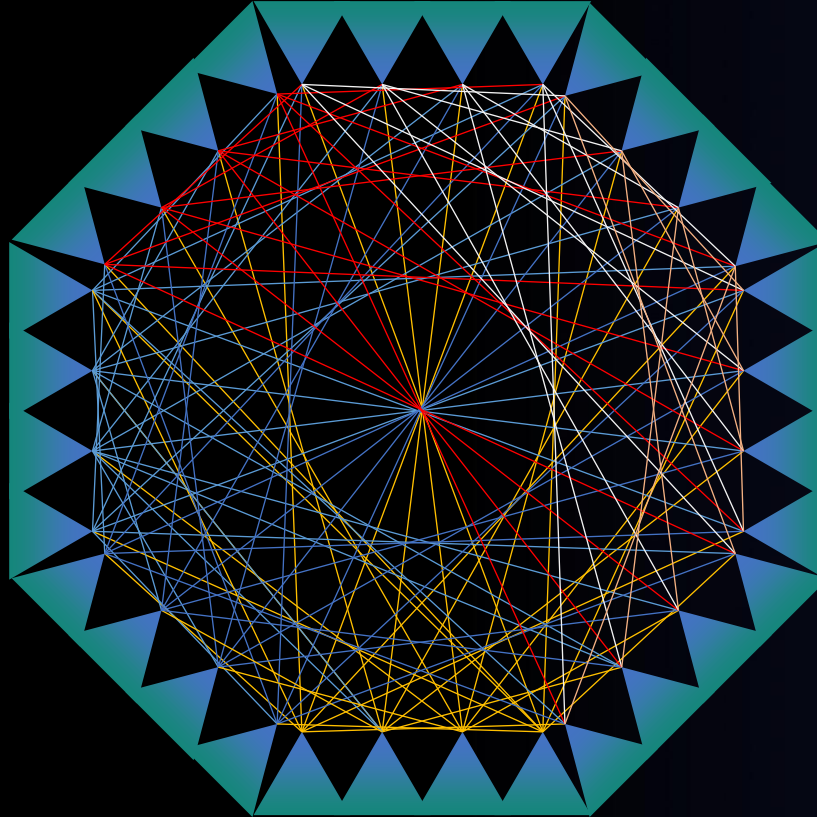
Premise

The future of AI isn't just about faster **compute**; it's about faster **data-movement** enabled by **smarter memory** to accelerate the next generation of autonomous agents

Standardization is key to help suppliers benefit from their investments:
Win/Win

Standard hardware will encourage broader software development: a positive feedback loop to garner higher **RoI**

Space & Time
an energy-bounded continuum



Scale Natively
Then, Scale Out
to Super Clusters (SC)

With given Energy,
We keep trading off
Time
&
Space

Memory

Provides

Space!

That's what

People

Want!

Interconnect

helps data move

Fast!

That's

absolutely

Necessary!

Work expands to fill all the available Time*

*Parkinson's Law: "Work expands so as to fill the time available for its completion."

Compute **grows** to fill all the allocated Memory

The bigger the Memory space,
the higher the Compute!

Trending Landscape

Big Data accelerated the **storage** technology: NAND and SSDs growth

- Lots of labeled data
- **Human**-generated data
- Permanent storage (hard to replace people)

Big Compute is running AI/ML

- **Machine**-generated data (large **scratchpads** for working sets and space for long-term memory)

We need faster, **temporary** storage space

- Memory (DRAM)

It's okay to regenerate data: ephemeral

- Time and Energy **trade-off**

Samsung Research Team
developed a few solutions
and conducted a series of experiments
to explore the new landscape

Boosting CXL[®] Performance for AI

CXL Webinar: [Vertical Optimization of the CXL Stack - Compute Express Link](#)

- **Latency Focused** : S-CHMU (Samsung CXL **H**ot-**p**age **M**onitoring **U**nit) – Device-level Page Migration Technology
- **Bandwidth Focused** : CXL-PNM (**P**rocessing **N**ear **M**emory): pre- and post-processing data @ CXL Memory side
- **Capacity Focused** : Pangea (CXL **P**ooling **S**W) – Increase effective memory capacity by accessing stranded memory

S-CHMU

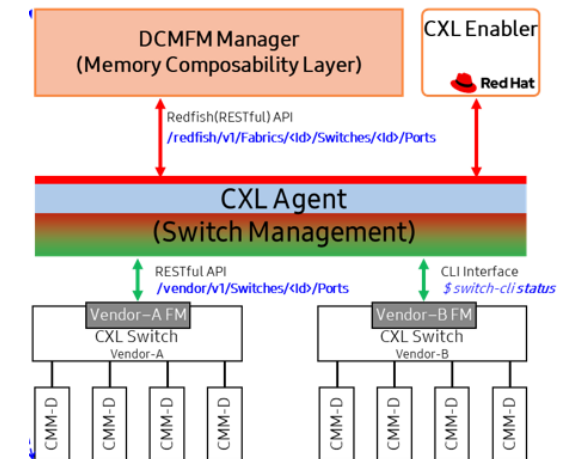
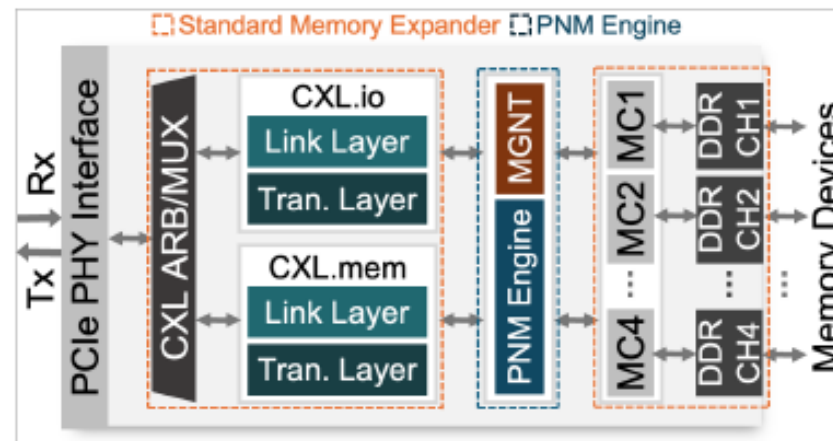
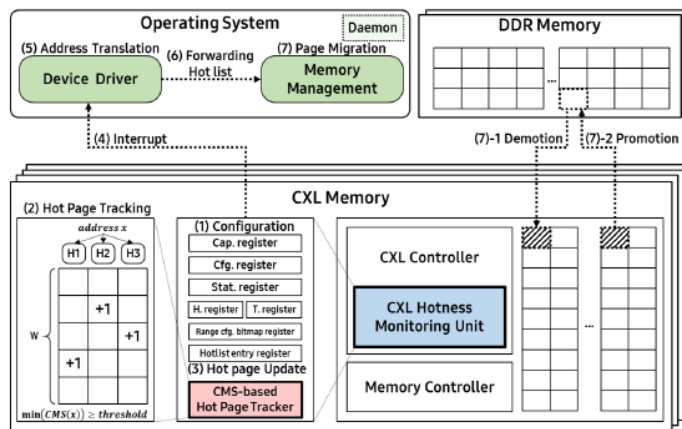
Latency: 40% reduction

CXL-PNM

Bandwidth : 5x

Pangea V2.0

Capacity: 50%



What did we learn?

Big Data

Feeds

Big Compute

Data-movement happens between memory!



As we transition to Agentic AI, the bottleneck has shifted.

Trending AI workloads are limited by data-movement-- not compute!

- **Data-movement** is through **memory** devices, memory protocols, and memory-optimized transports!
- Memory is the "place" to keep data while in **transit** and when **staged** prior to **execution**!

Transit

Stage

Execution

A practical example of how Big Data meets Big Compute!

Compute will grow to the extent we allocate memory to it!

Recent research on **RAG** pipelines illustrates the value of traditional **search** and **database** techniques feeding the emerging AI/ML compute engines. (Retrieval Augmented Generation)

We are familiar with the value of **large memory footprints** for **database** and **search** engines.

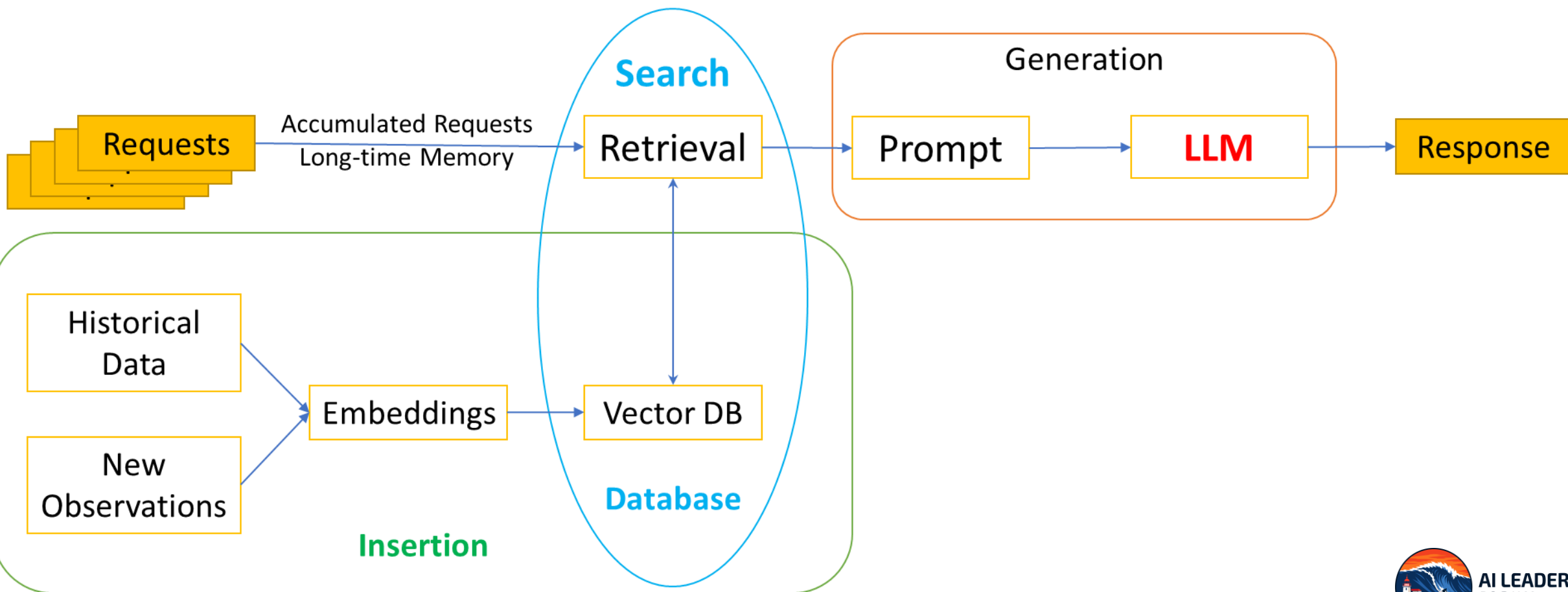
Caching processed data helps reduce data-movement and re-execution

Request

Pre-fill

Decode

Response

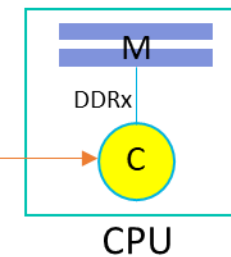
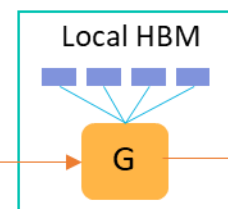
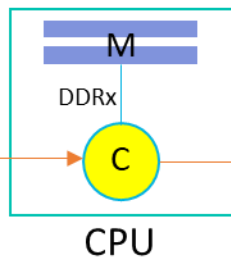
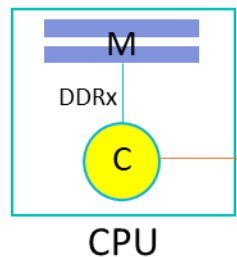


Request

Pre-fill

Decode

Response



xLink

Search

Generation

Requests

Accumulated Requests
Long-time Memory

Retrieval

Prompt

LLM

Response

Historical
Data

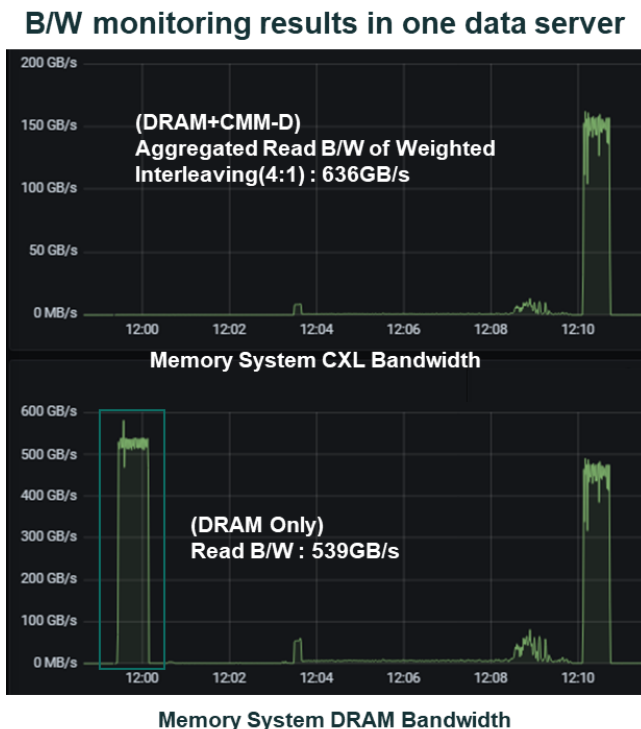
New
Observations

Embeddings

Vector DB

Database

Insertion



Different **pipeline phases** require different amounts of memory.

- Instead of overprovisioning memory for **peak demands**, memory-**pooling** helps increase the size of models we can support while increasing compute resource utilization.
- Compute nodes may **inflate** or **deflate** their memory footprint based on individual RAG pipeline memory and compute utilization.

CPUs
have become a major factor
in AI growth!

Traditional Search and Database engines
feed data from **memory**
to the AI/ML machines

Search and Database engines
are the software **Link**
Between
user requests and AI/ML engines

CPUs
are the hardware **Link**
Between
users and AI/ML engines

CXL[®]

is the **Link**

Between

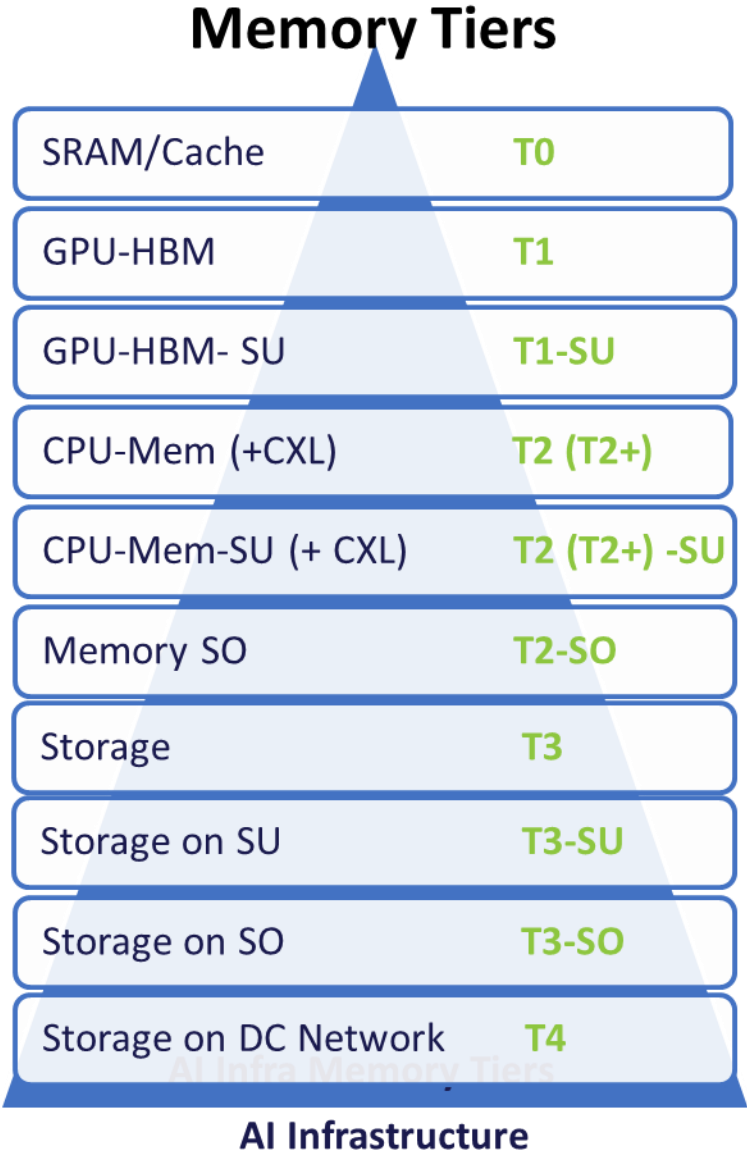
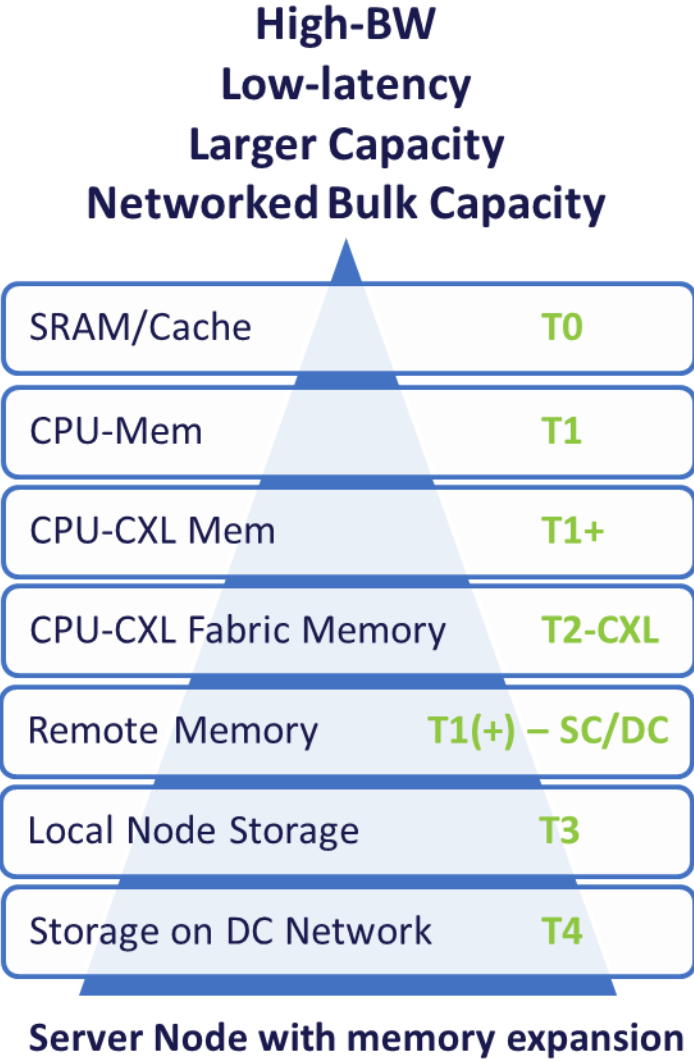
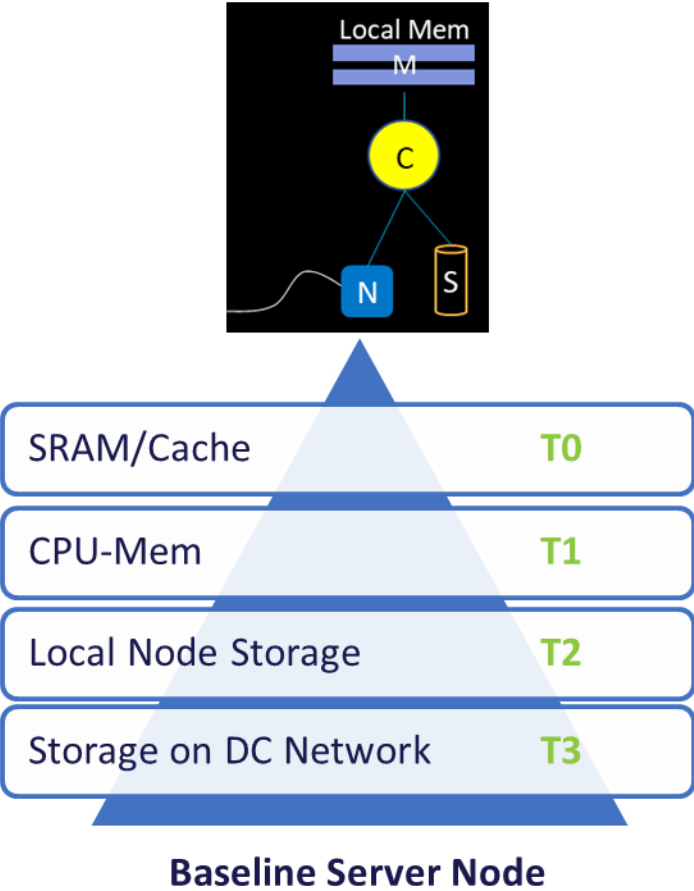
Bulk Memory and AI/ML engines

Every CPU

has several

CXL[®] Ports

Sever-centric Memory-tier vs. AI Infra Memory-tier Pyramids within **Scale-up** (SU) and **Scale-out** (SO) Fabrics



Increase Capacity at each Tier

SRAM/Cache	T0
GPU-HBM	T1
GPU-HBM- SU	T1-SU
CPU-Mem (+CXL)	T2 (T2+)
CPU-Mem-SU (+ CXL)	T2 (T2+)-SU
Memory SO	T2-SO
Storage	T3
Storage on SU	T3-SU
Storage on SO	T3-SO
Storage on DC Network	T4

AI Infra Memory Tiers

AI Infrastructure

Expand Capacity at each Tier

Deliver the expected Bandwidth

How do we physically realize this
memory hierarchy?

Bridges between Scale-up Fabrics (xLink) and CXL-enabled Memory/Storage

Opportunity for **NAND** and **DRAM**

- AI/ML needs lots of memory and storage
- Memory-tiering is a solution
- Suitable software programming model is essential
- Suitable memory and storage attachment points to the scale-up fabric is necessary

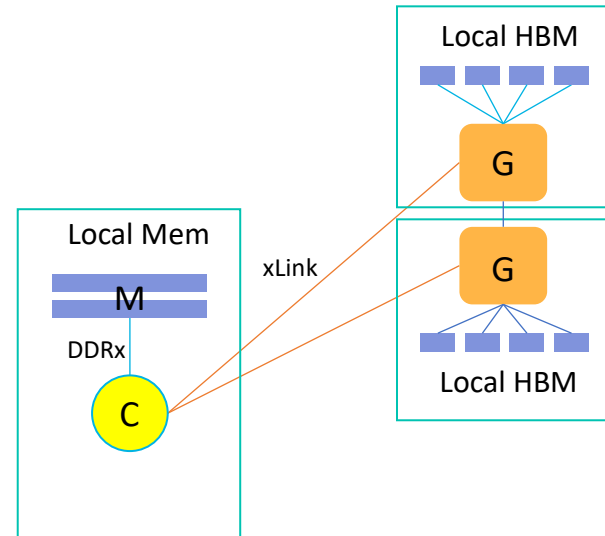
Scale-up natively,
then **bridge**
onto **standard** components
via Hybrid Switches

A GPU / xPU Node

Depicted: interconnecting dual-ported devices to represent multi-ported ones

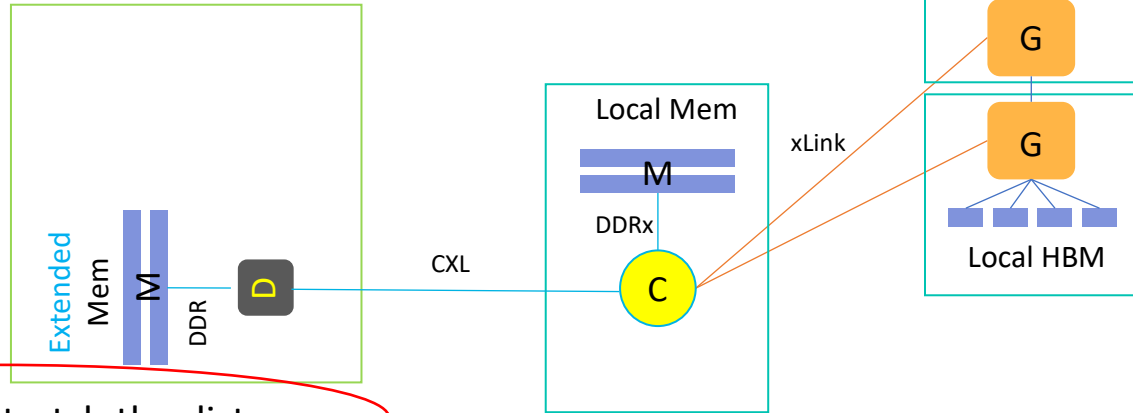
Scale-up using native interconnects and protocols

G: GPU, xPU
C: CPU
M: Local memory
S: NVMe SSD
N: NIC

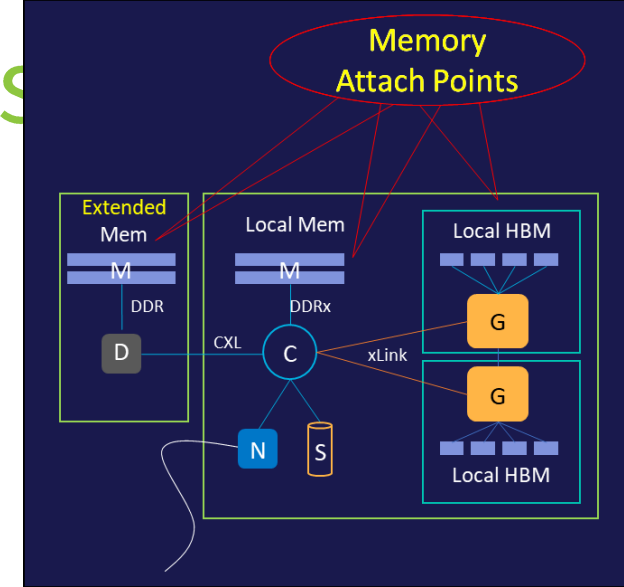


Bulk Memory directly-connected to CUPs

G: GPU, xPU
C: CPU
M: Local memory
S: NVMe SSD
N: NIC
D: CXL Device



Extended Memory: stretch the distance

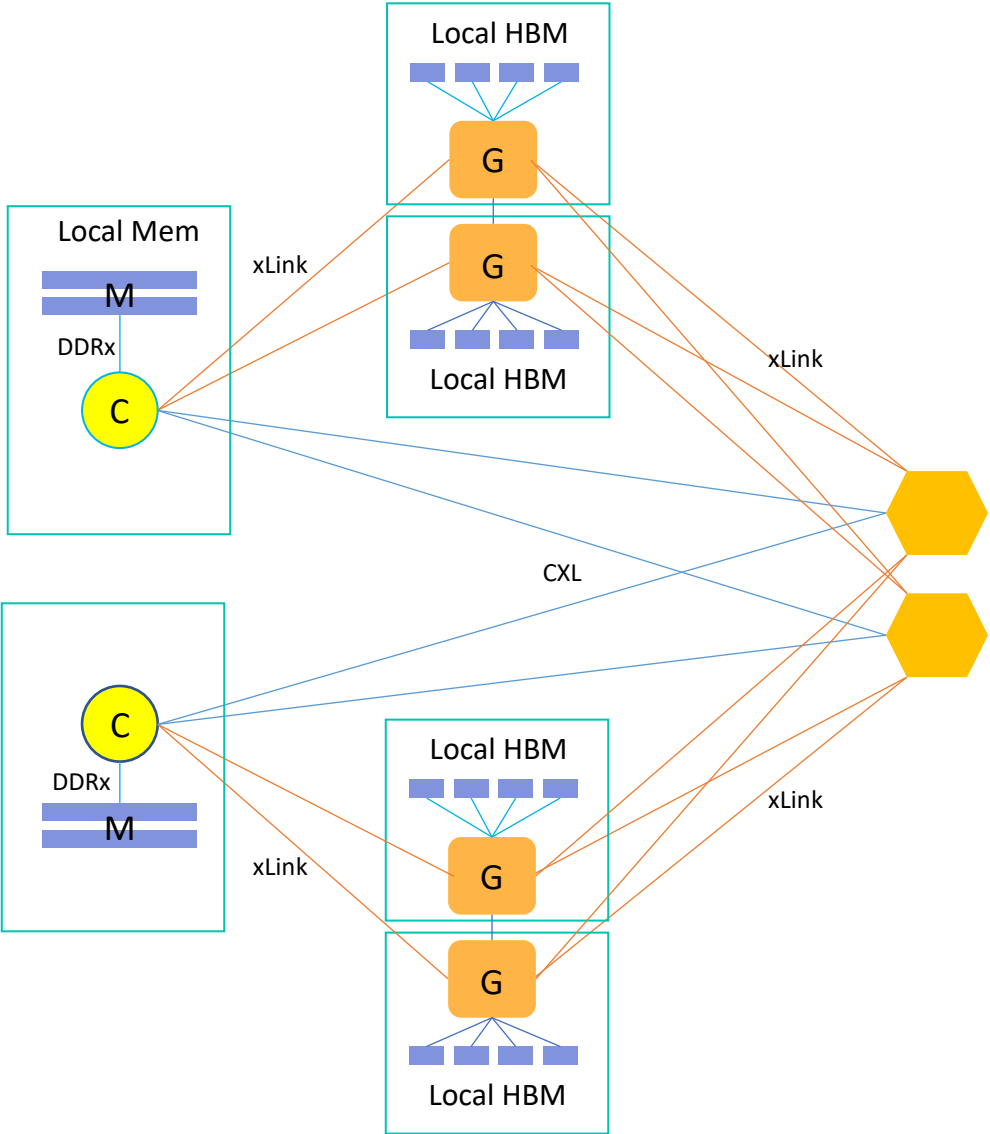


Scaling Up Compute via Fabric Topologies

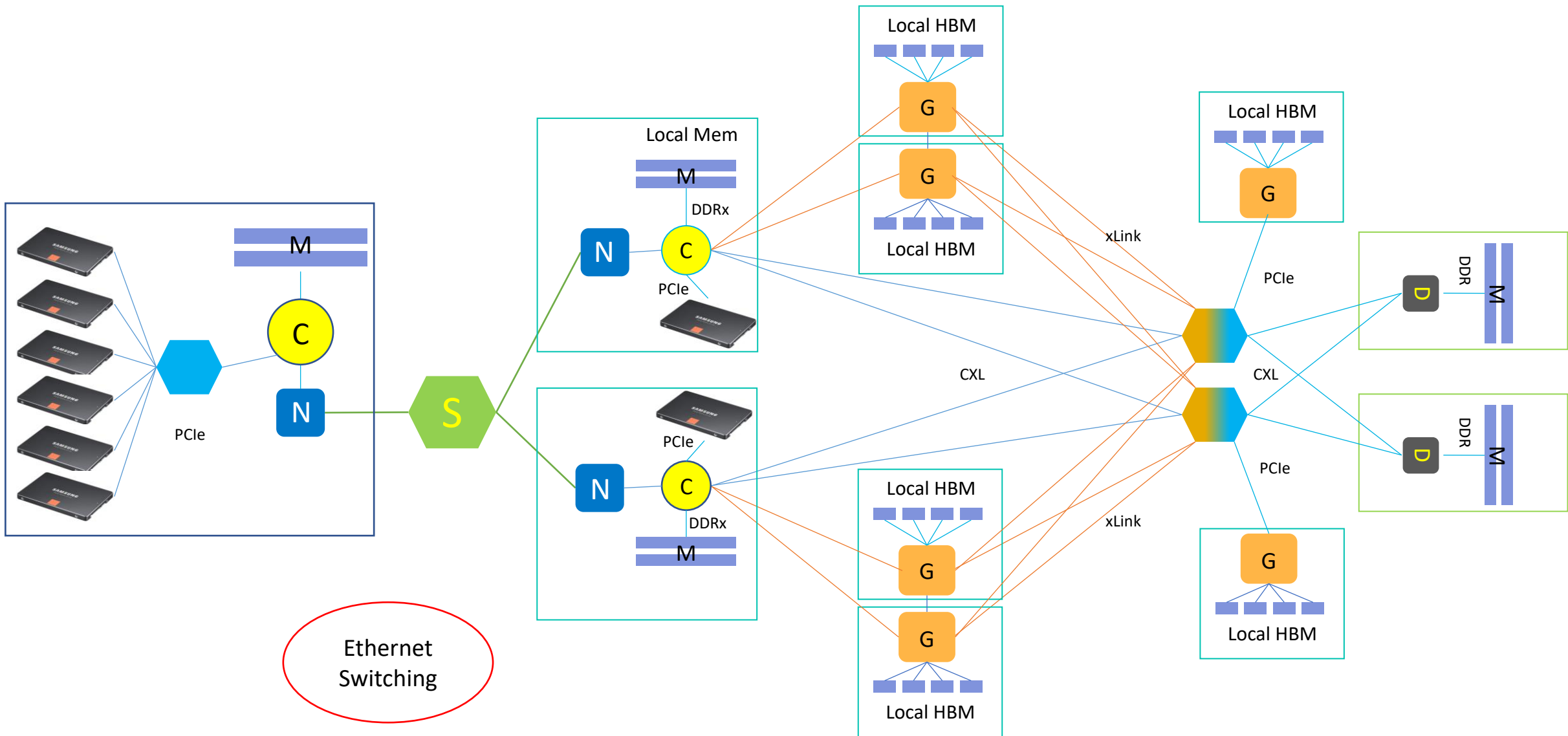
G: GPU, xPU
C: CPU
M: Local memory
S: NVMe SSD
N: NIC
D: CXL Device

Extended Memory
Expanded Memory

xLink Switching



Expanding Virtual Memory via “network-connected” SSDs and RDMA



A simple template for various topologies

Depicted: interconnecting dual-ported devices to represent multi-ported ones

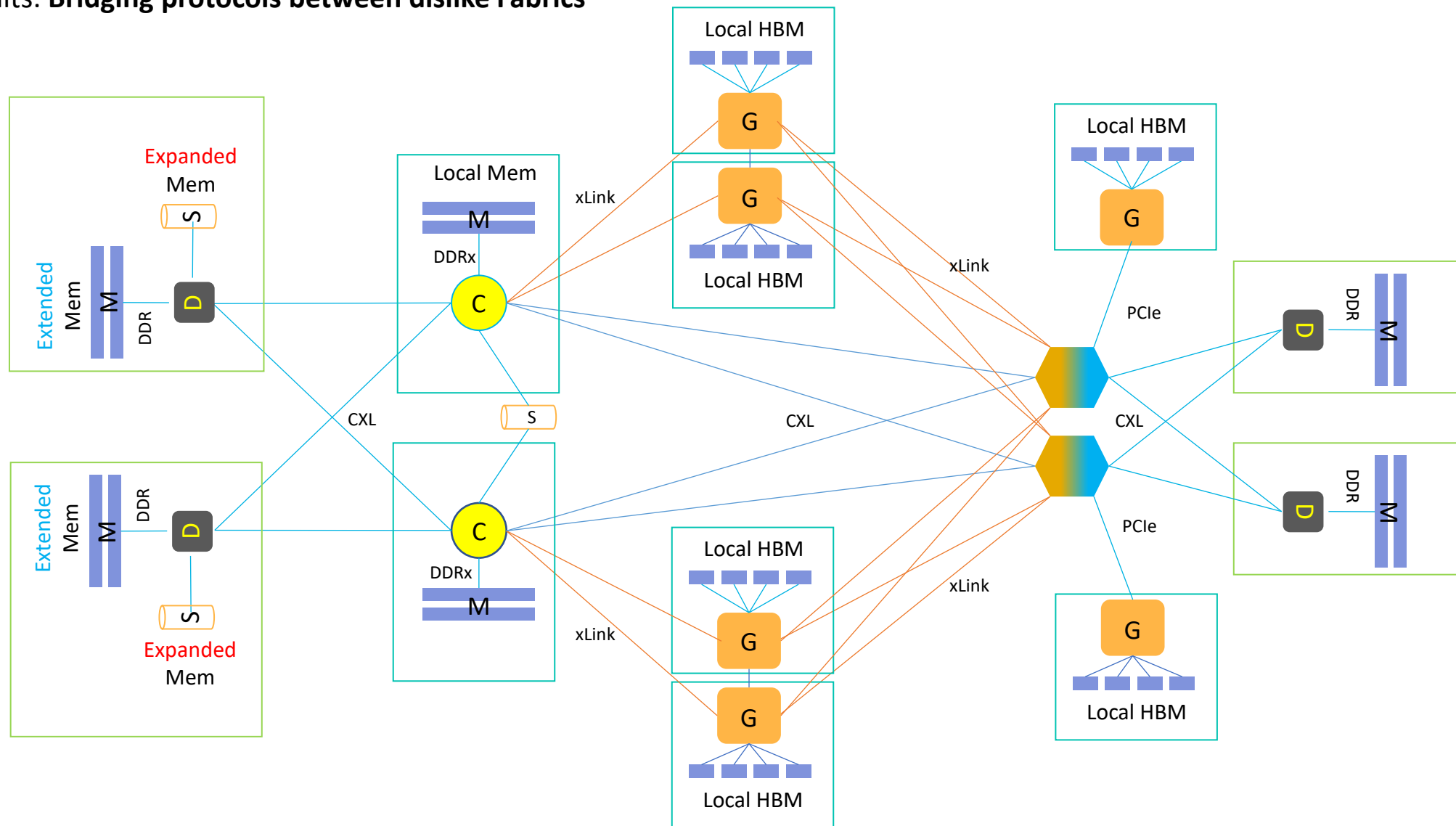
Scale-up using native interconnects and protocols plus pooling/sharing using Fabric techniques

Hybrid Switching Units: Bridging protocols between dislike Fabrics

G: GPU, xPU
C: CPU
M: Local memory
S: NVMe SSD
N: NIC
D: CXL Device

Extended Memory
Expanded Memory

PCIe Switching
CXL Switching
xLink Switching
Hybrid Bridging



How do we
physically realize this?

Will we need
a **customized** memory controller
for every situation?

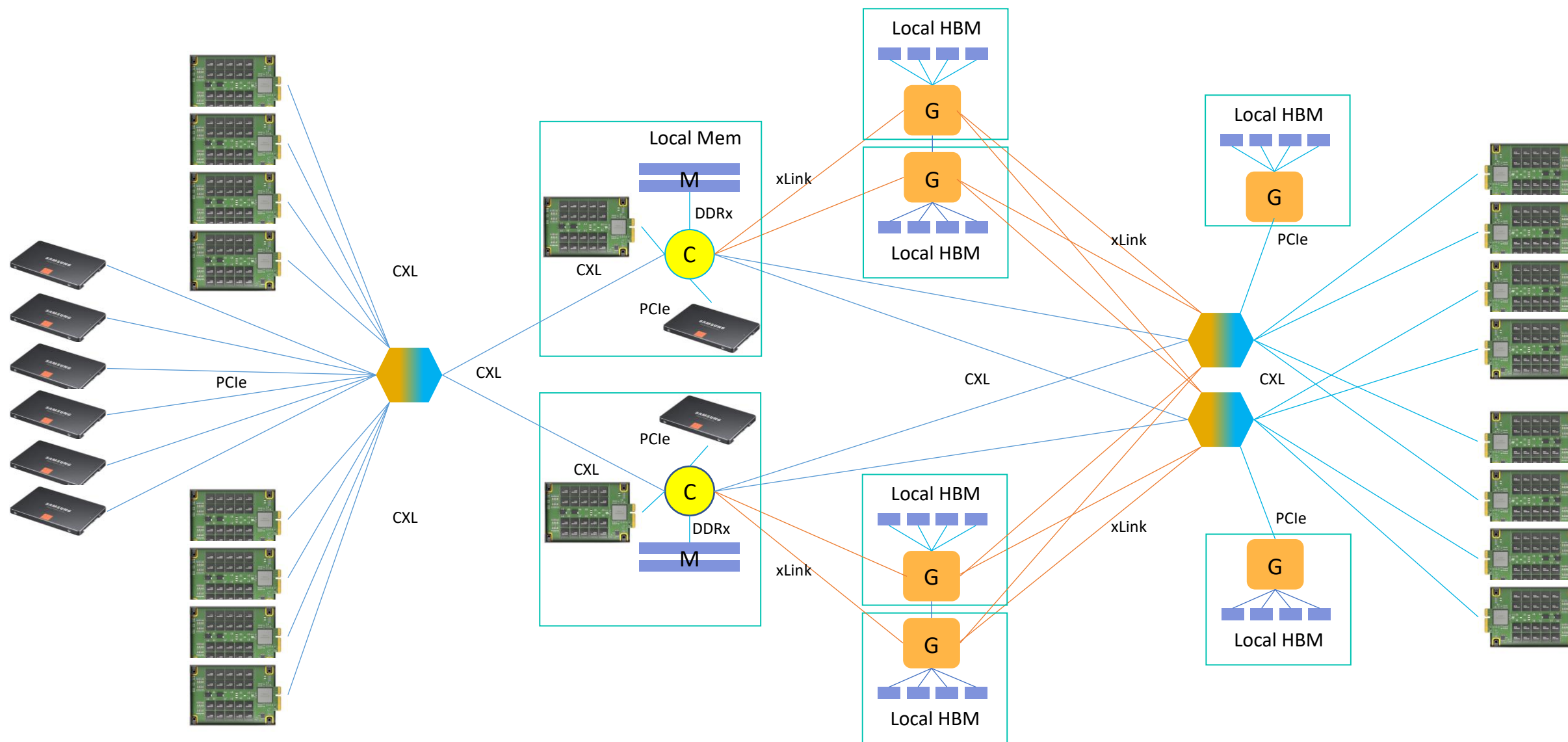
Or

will a

standard memory interconnect

do the job?

CXL Memory serves various use-cases!



Common

Physical Layer (for bulk memory and IO)

Protocol for interoperable Controllers

Form Factors for DRAM and NAND

CXL[®] is Foundational!

Backward-compatible!

Investment Protection
(RoI)

Win / Win

Takeaways

- Standardization is key to help suppliers benefit from their investments: **Win/win**
- Standard hardware will encourage more software development: a positive feedback loop to garner **RoI**
- Each major CPU includes several CXL Root Ports: **Prevalence**

Growth Path Enabler

Bridging between specialized scale-up fabrics and standard CXL Memory