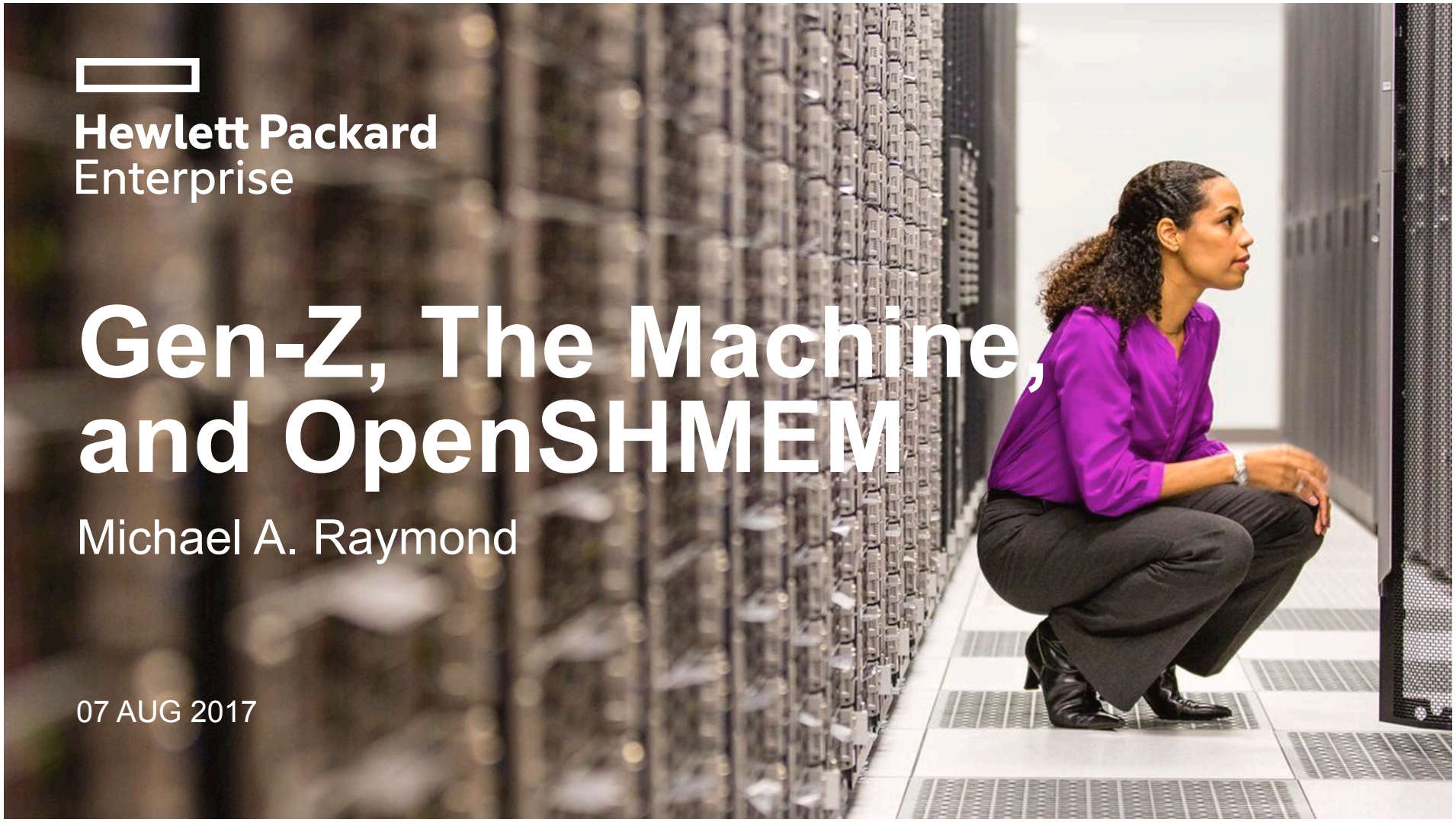


Gen-Z, The Machine, and OpenSHMEM

Michael A. Raymond

07 AUG 2017

Agenda

– Mental models

– A merged programming model

– Related hardware trends

– The Gen-Z project

– Future work

Message Passing (e.g. MPI)

– The dominant HPC programming model

– Usable on almost any type of hardware

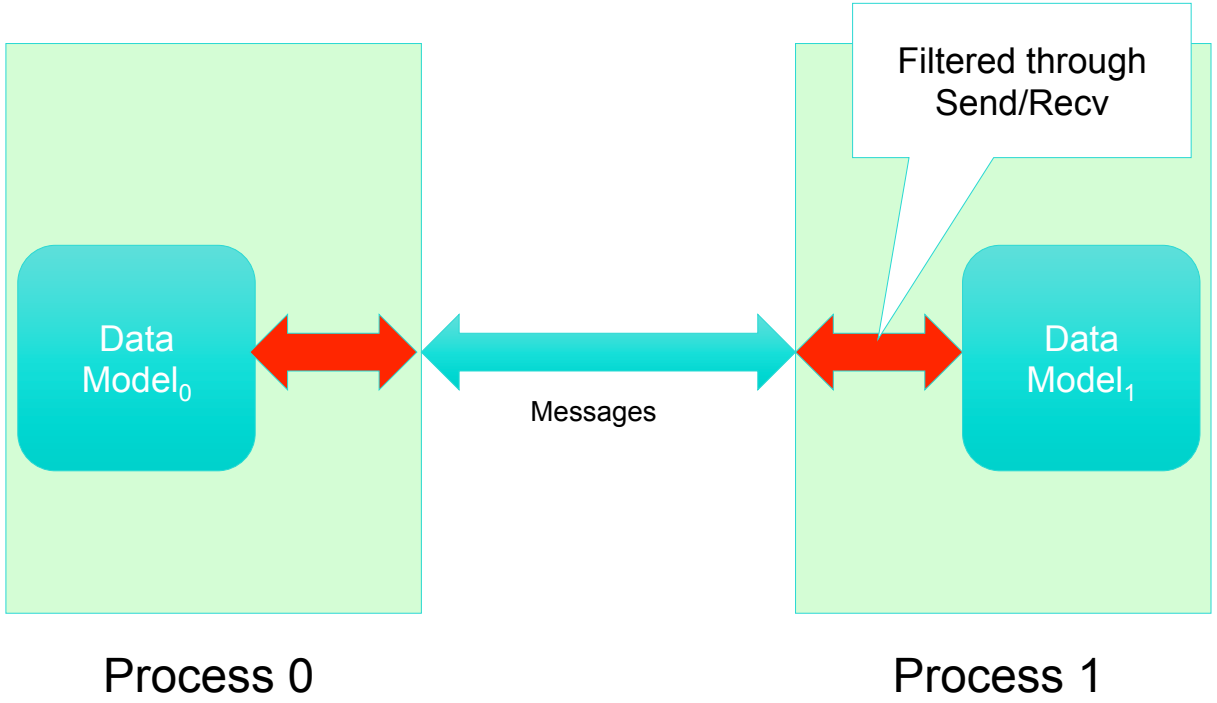
– Makes data exchanges easy to reason about (most of the time)

– Every data transfer must have an active sender(s) and an active receiver(s)

– With some application domains, there is no way to predict who you will need to exchange data with

– Synchronization is implicit

Message Passing (Abstracted)



Remote Memory Access (e.g. OpenSHMEM)

– Processes do Puts / Gets / Atomic Memory Operations (AMOs) to each other's address spaces

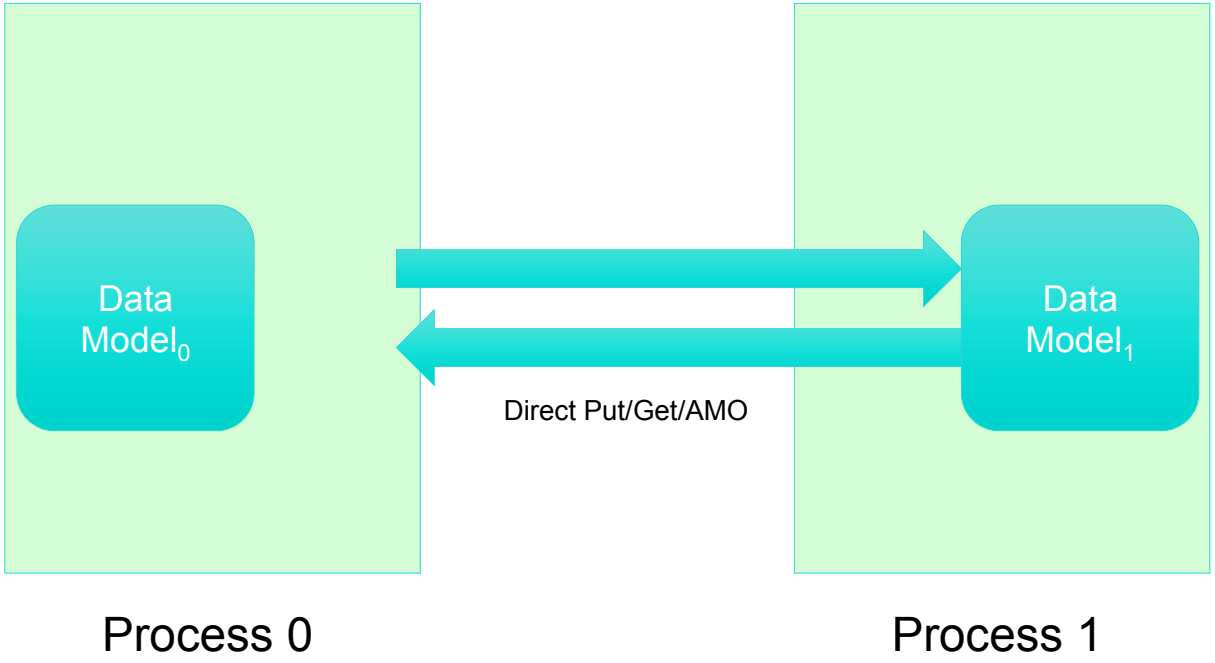
– Implemented with hardware RDMA (in most cases)

– Only requires an active initiating process

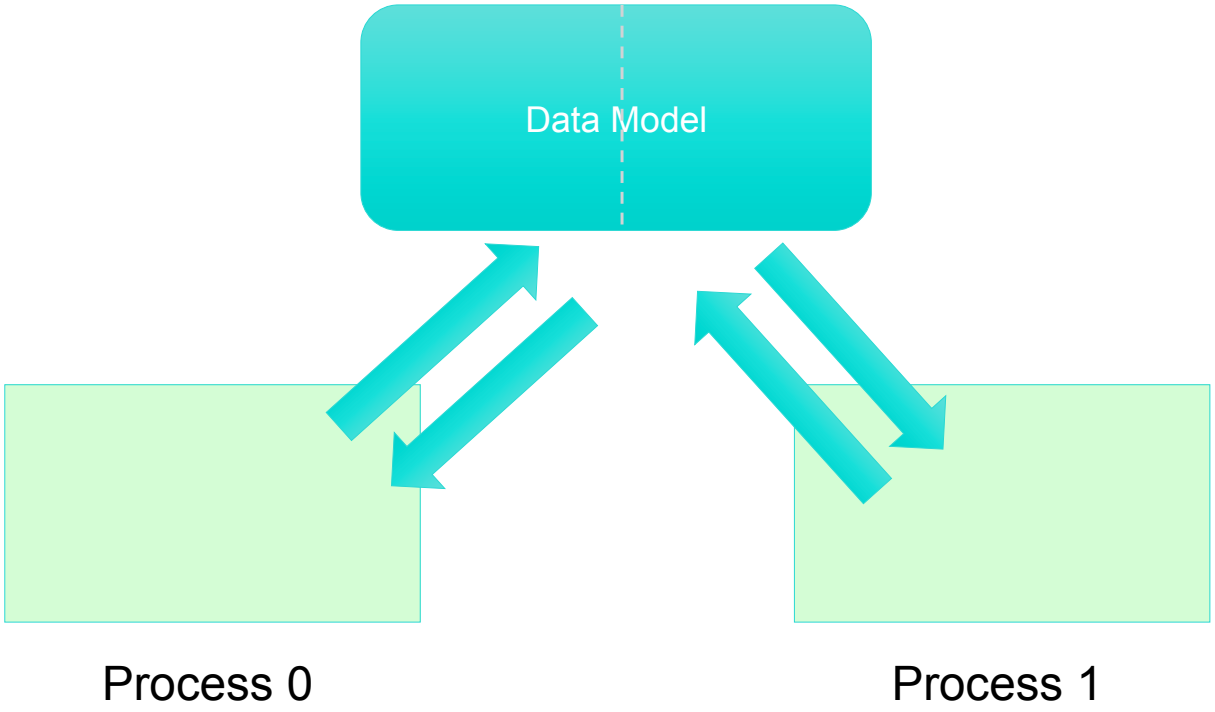
– Makes irregular data exchanges easier to accomplish

– Synchronization is explicit

Remote Memory Access (Abstracted)



Taken further…



Direct Put/Get/AMO

Borrowing Ideas

Prior OpenSHMEM Work

– User-Defined Spaces: Welch et al in Proceedings of PGAS 2014
  – Proposed “spaces” for symmetric heaps amongst a subset of the PEs
  – Saves memory and reduces synchronization
– Work by Cray and Intel on an interface for selecting where to locate the symmetric heap
  – Discussed at the FEB 2017 Face to Face Meeting in Atlanta
  – Presented an interface for selecting on what type(s) of memory to locate the symmetric heap
  – For systems with hierarchical memory / different types of memory
  – Gives explicit control to the application

Recent HPC Middleware Designs

– E.g. Map Reduce / Hadoop, Spark, Cassandra, Pregel / Giraph

Pros

– Highly scalable, only require TCP/IP, expect frequent node failures

– Fault tolerant

– Can change the number of workers

– “Applets”

Cons

– Expect little communication between workers / shards

– Generally don’t take advantage of high performance networking features

Put Them All Together

– A / The disaggregated symmetric heap persists **external** to the worker processes

– Not constrained to the (limited) RAM on the compute hosts

– If a PE exits, the others retain access to this heap

– The number of PEs accessing the heap can change during their lifetime

– Different applications can simultaneously access the heap

– Within the limits of Consistency, Availability, Partition tolerance (CAP) theory, the heap has some degree of fault tolerance

How is this different than a shared DB / File system?

– Direct byte-level addressability of the raw data in the heap

– Accessibility through high performance RDMA methods

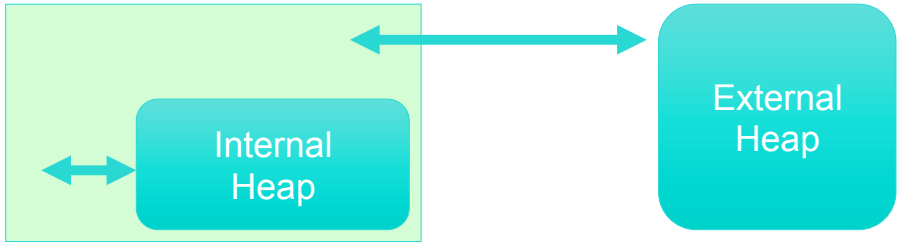
– Can use the same OpenSHMEM methods as with other heap(s)

  – May need to pass an object specifying which heap to access

– No built in transactions

Programming Challenges: Performance

- Location matters
- External transfers are slower
- Do we keep a copy of the data locally and “publish” when ready?
- Do we also use a traditional symmetric heap?
  - How would be divide up the data model?
- What’s the best way for a Hierarchical Storage Manager to manage the external heap?



Programming Challenges: Security, Coherency, and Fault Tolerance

– Need to control access to the external heap
  – Starts to look like a filesystem management issue
– Need to manage allocation within the external memory devices
– How do we detect, protect against, and manage failed updates to the external heap?
  – Double buffering for fault tolerance?
  – Do OpenSHMEM programs need to start doing transactions?

Hardware Trends

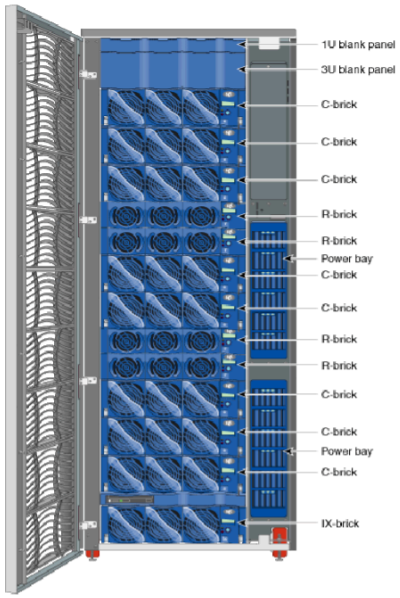
SGI M-brick

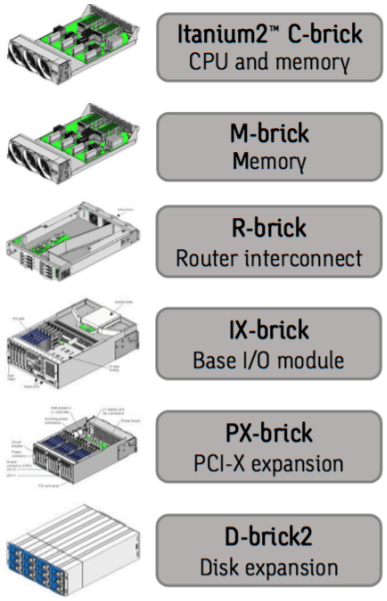
– Some customers had enough compute, but needed extra memory

– M-bricks were SGI Origin3000 and Altix nodes with extra memory and no CPUs

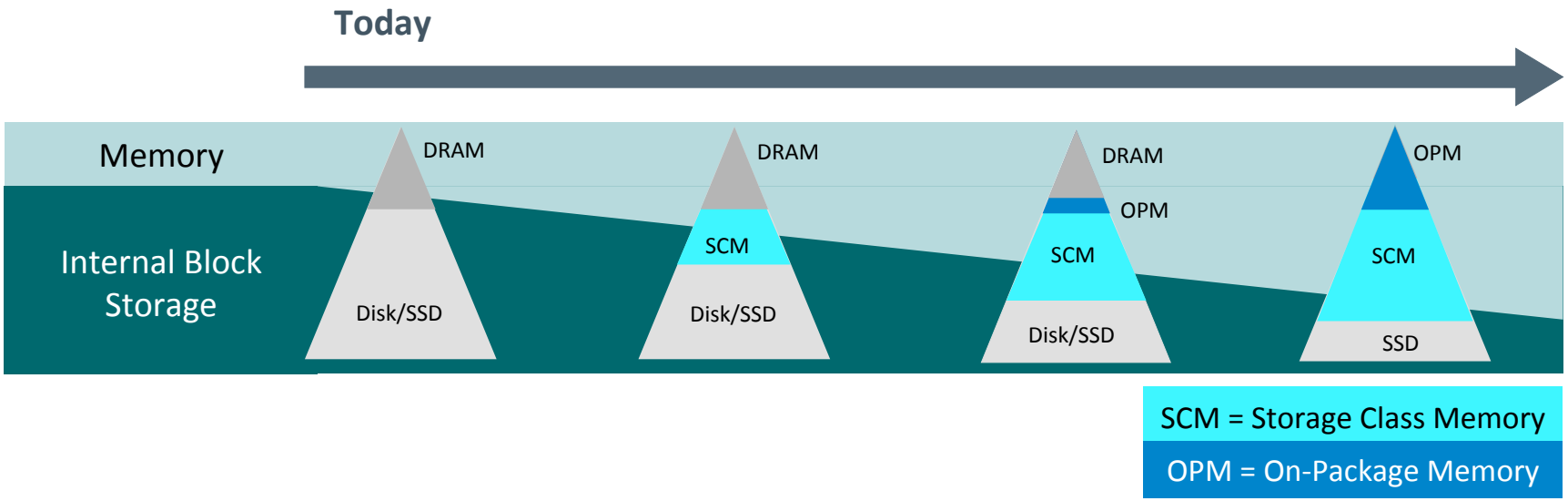
– Saved customers money by not having to pay for unneeded CPUs

– A larger working set size was worth the cost of off-node memory accesses



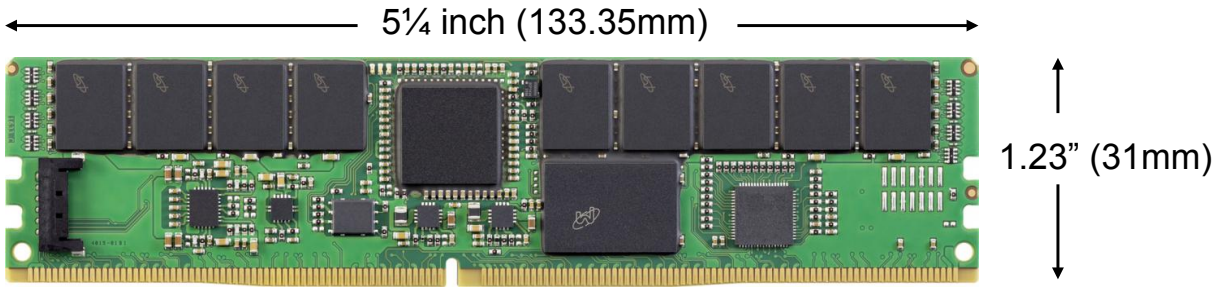


Memory/Storage Convergence: The Media Revolution



Memory Semantics will be pervasive in Volatile ***AND*** Non-Volatile Storage as these technologies continue to converge.

SCM on the Memory Bus



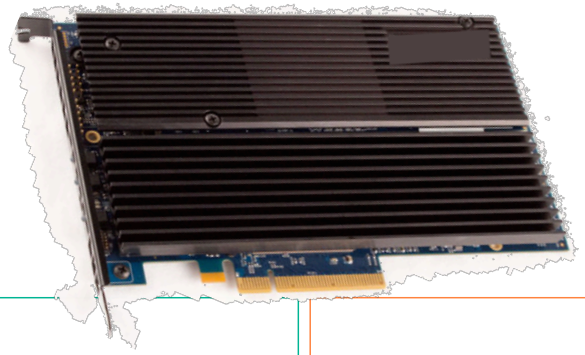
Pros

- Low latency, high bandwidth
- Memory semantic interface (Load/Store)
- Standard mechanical format (DIMM)
- Standard electrical interface
- Environmental conditions are well known

Cons

- Each NVDIMM consumes a Memory Slot
- DIMM format constrains capacity
- Constrained electrical interface
  - Single Ended (slower than differential)
  - Directly connected (pin intensive)
- Close proximity to CPU = thermal & placement challenges

SCM on the PCIe Bus



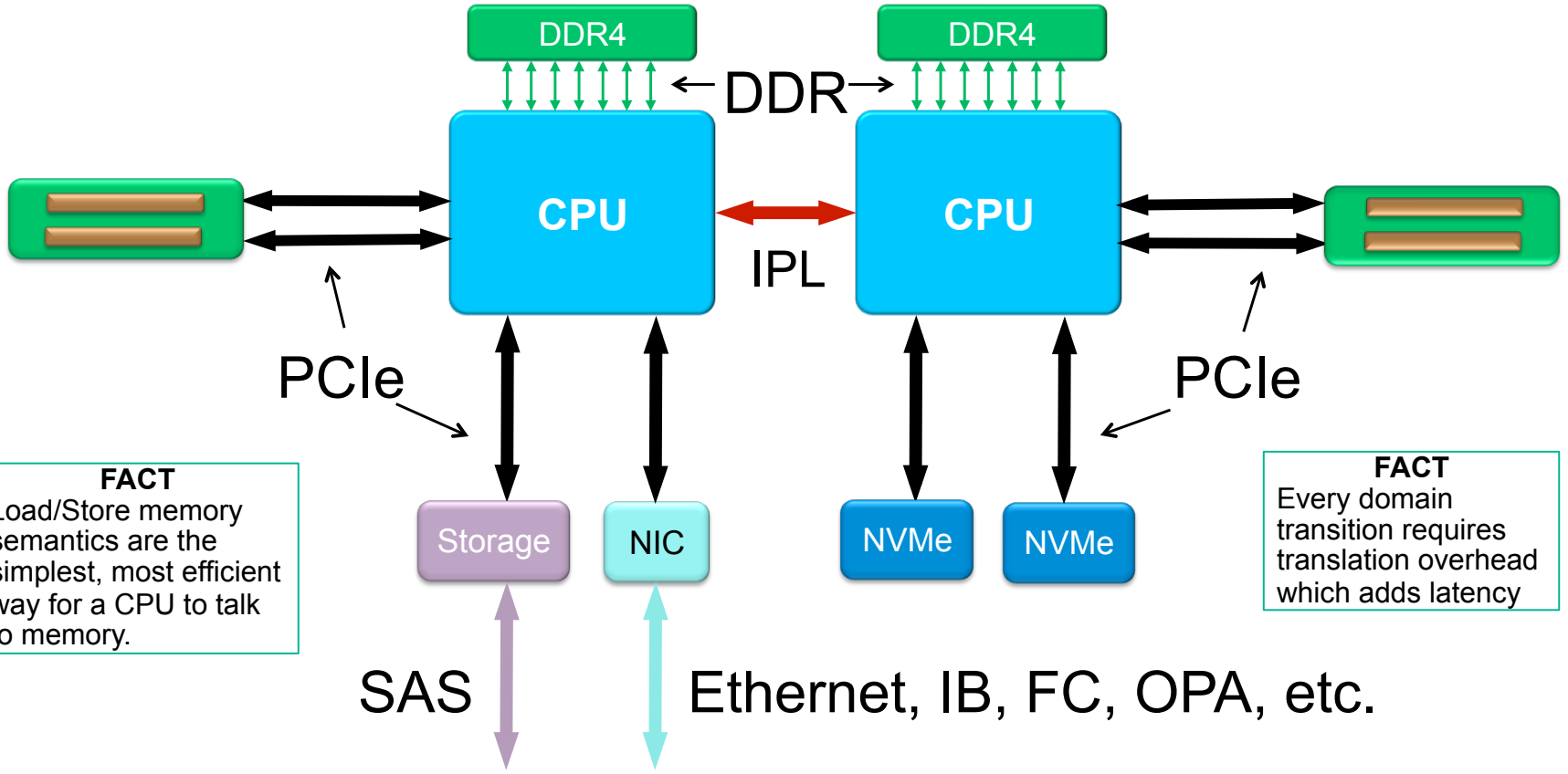
Pros

- Good enough bandwidth/latency
- Standard mechanical format
- Standard electrical interface
- Programming interface well known
- Environmental conditions are well known

Cons

- Performance not keeping pace up CPU
  - 4GT/s, 8GT, 16GT/s (2003 – 2017, 14 years)
  - Bandwidth: PCIe x16 ~ < 1 memory channel (16GB/s)
- Asymmetric architecture (1x root, Nx endpoints)
- Architecture does not scale beyond the server
- Limited to 256 components per fabric
- No multi-path support

Typical 2 socket architecture – 8 possible interconnect types



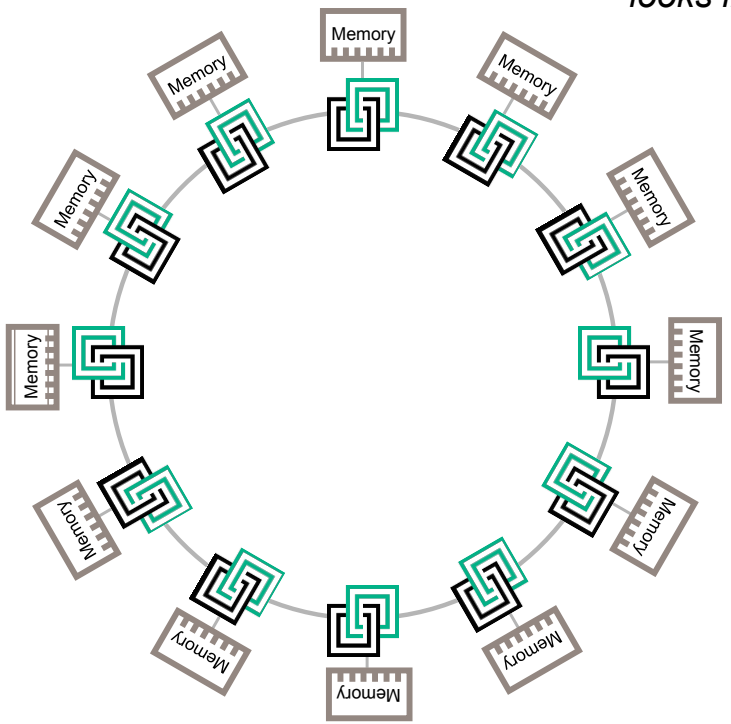
FACT
Load/Store memory semantics are the simplest, most efficient way for a CPU to talk to memory.

FACT
Every domain transition requires translation overhead which adds latency

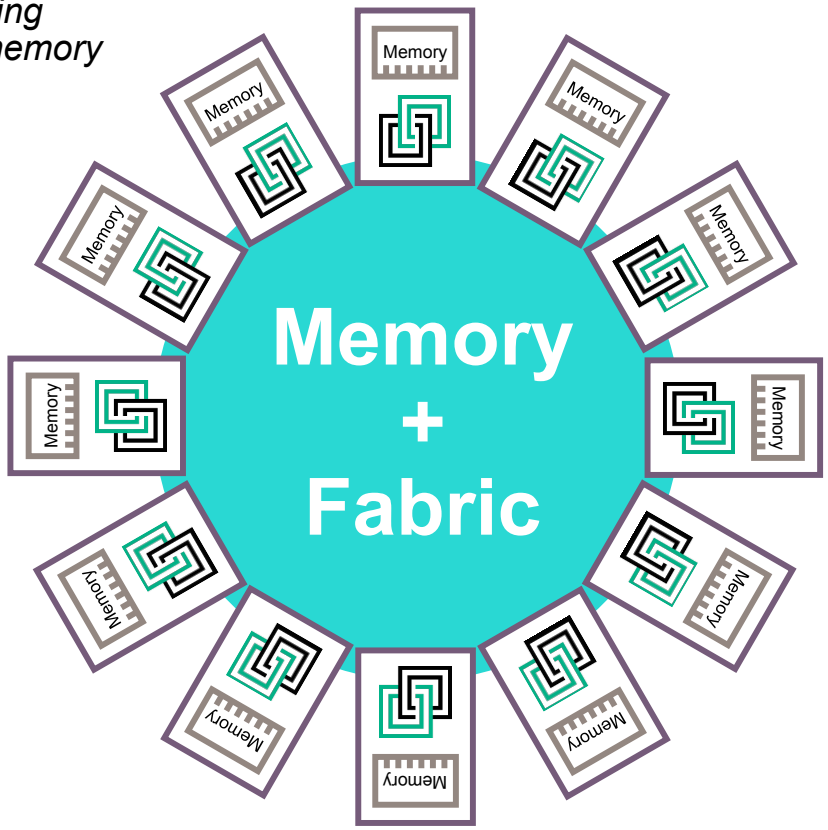
Hewlett Packard Enterprise

HPE & Memory Driven Computing

Everything looks like memory



From processor-centric computing…



…to Memory-Driven Computing

Hewlett Packard Enterprise

Driving Principles: Memory-Driven Computing

- A new and **open memory-semantics system protocol**
- **Co-packaged high-performance DRAM** for the application working set
- **Fabric-attached nonvolatile memory** converging memory and storage
- **High-radix router interconnect topology,** using direct memory access
- **Silicon photonics** integrating micro-rings, logic and fiber attach in 3D package

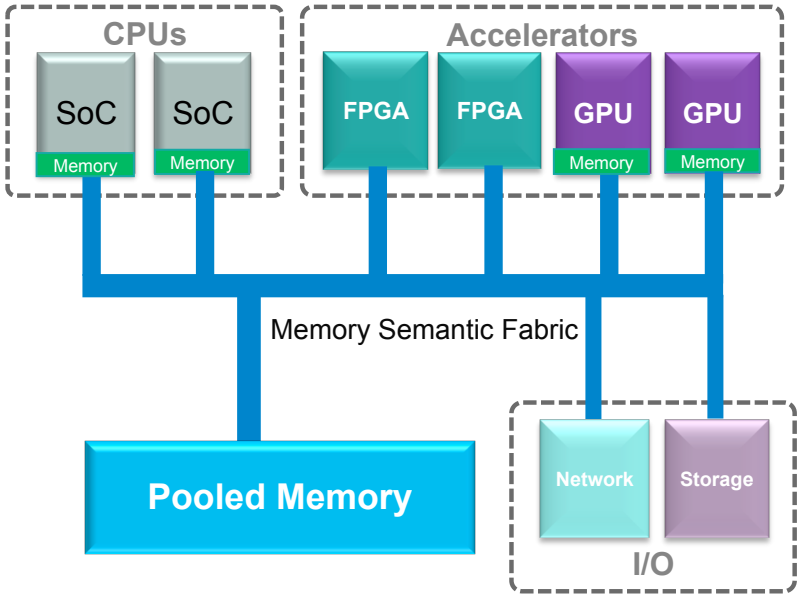
Memory Semantic Fabric

Communication at the speed of memory

What is a Memory Semantic Fabric?

- A communication protocol that speaks the same language the CPU speaks: load/ store, put/get, and read/write

- Connectivity that extends beyond the server to the rack and data center



Hewlett Packard Enterprise

Open: Broad Industry & Device Support

GEN Z Consortium Members

ALPHA DATA, AMD, Amphenol, ARM, BROADCOM, CAVIUM, CRAY, DELL EMC, FIT, Hewlett Packard Enterprise, HUAWEI, IBM, IDT, Lenovo, LOTES, Mellanox TECHNOLOGIES, Micron, Microsemi, NOKIA, PLDA GROUP, redhat, SAMSUNG, SEAGATE, SK hynix, Western Digital, XILINX

- Components
- Intellectual Property
- Connectors
- Subsystems
- Systems
- Software

GEN Z Component Categories

CPUs

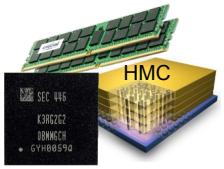
GPUs

Special Purpose Processors

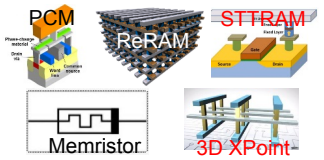
Storage Controllers

Switches/ Gateways

FPGAs



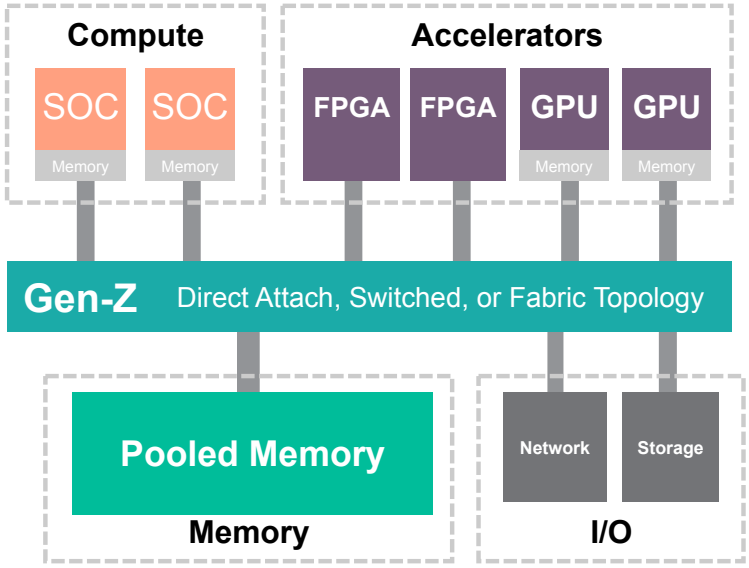
Volatile Memory



Storage Class Memory

HPE Exascale strategy centered around Gen-Z

High Bandwidth Low Latency	– Memory Semantics – simple Reads and Writes – From tens to several hundred GB/s of bandwidth – Sub-100 ns load-to-use memory latency
Advanced Workloads & Technologies	– Real time analytics – Enables data centric and hybrid computing – Scalable memory pools for in memory applications – Abstracts media interface from SoC to unlock new media innovation
Secure Compatible Economical	– Provides end-to-end secure connectivity from node level to rack scale – Supports unmodified OS for SW compatibility – Graduated implementation from simple, low cost to highly capable and robust – Leverages high-volume IEEE physical layers and broad, deep industry ecosystem



Hewlett Packard Enterprise

HPE’s The Machine

– Announced The Machine concept in 2014

– Prototype announced in May 2017

– 1280 cores

– 160 TB of RAM

– Gen-Z implemented on FPGAs

Hewlett Packard Enterprise

Future Work

Additional OpenSHMEM Needs

– Methods to create, destroy, search for, and query external symmetric heaps

– A method to specify which heap to use for transfers

– Research into fault tolerance with persistent direct-access memory

– Ideas and examples of how to use these external heaps and get good performance

Questions?