Accelerate Everything.

Successfully Deploying Persistent Memory and Acceleration via Compute Express Link!

Stephen Bates, Chief Technology Officer, PIRL 2019

It's all about the software. Until you reach the limits of the hardware. Then it's all about the hardware [1].

[1] some geek, 2017.

it's [Software] an afterthought in most cases [of hardware standardization]. Usually to the detriment of adoption..... ***[2].***

[2] some geek, 2019.

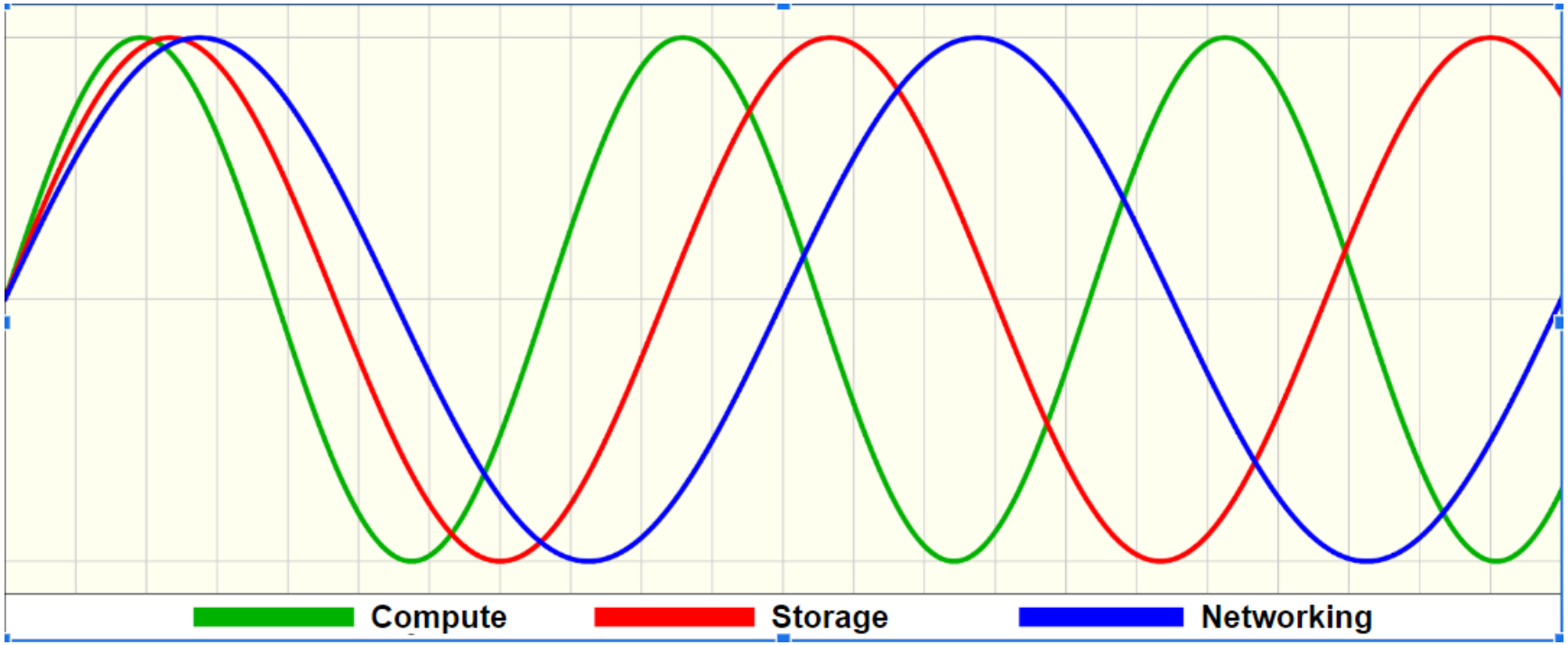
We've Come a Long Way, Baby!

1955: 5MB, 1 million USD

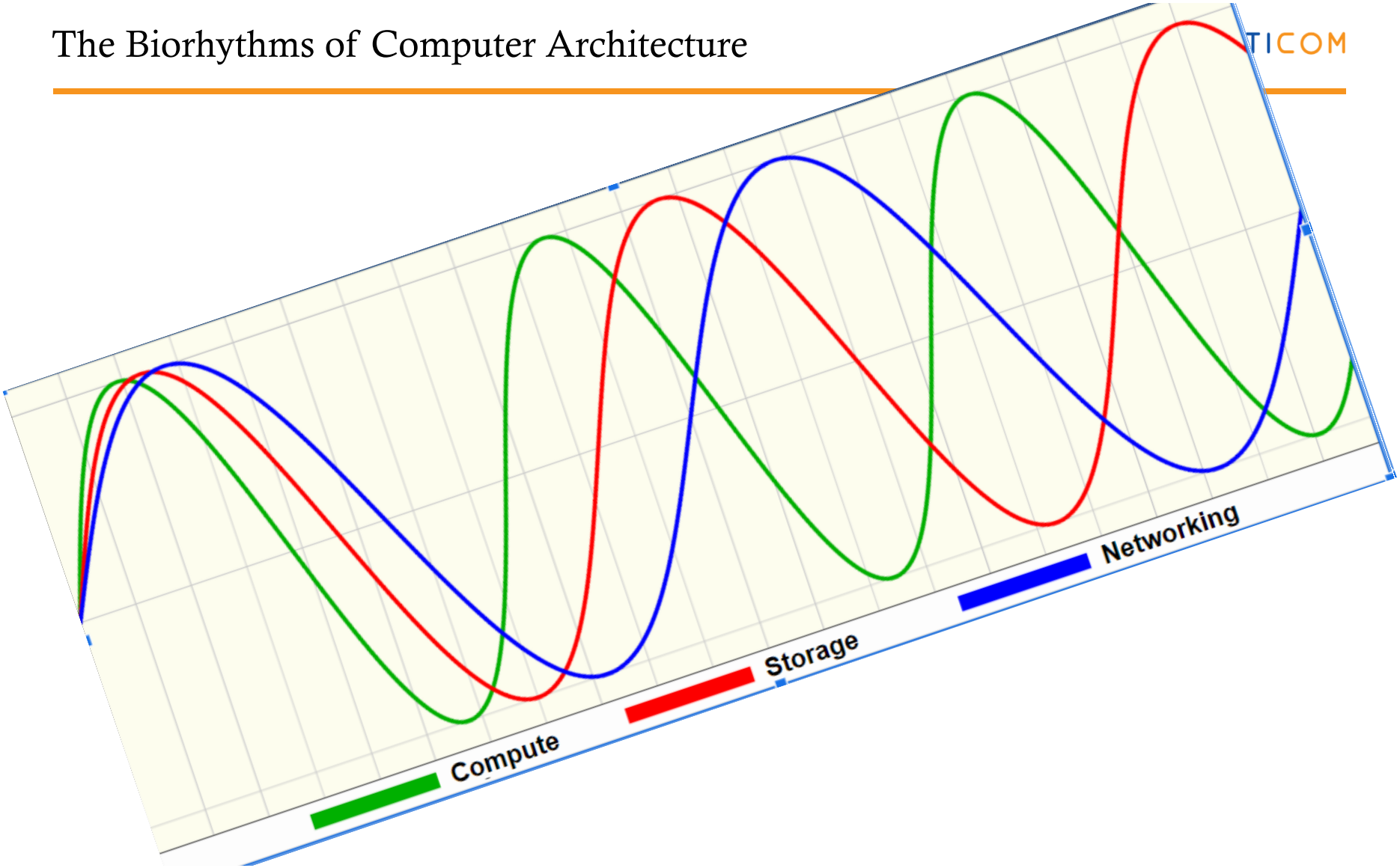
2018: 1TB (1000000MB), 500 USD

About a Billion times improvement in $/GB in 65 years

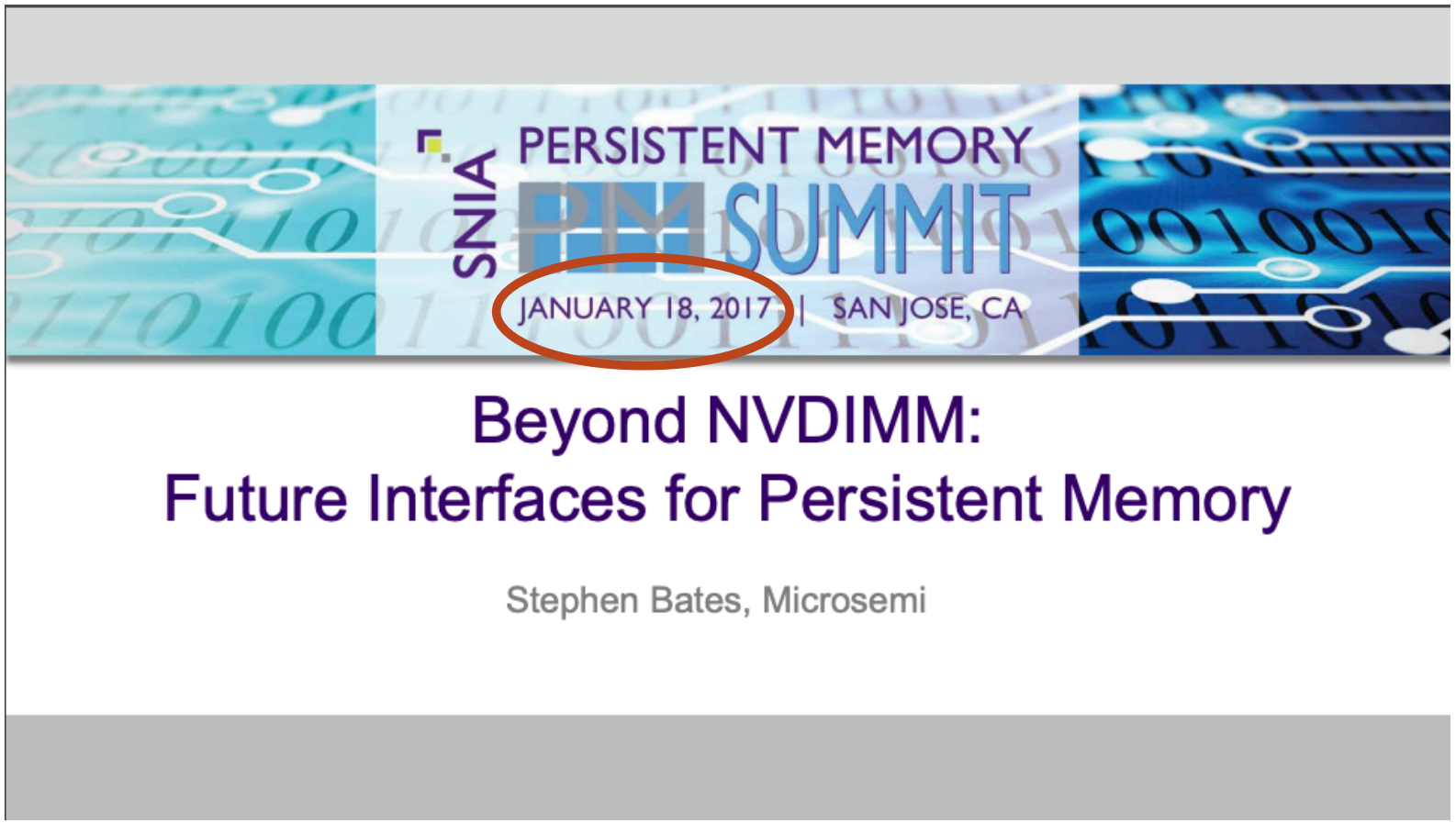
The Biorhythms of Computer Architecture



The Biorhythms of Computer Architecture



We've Been At This For Some Time….



Persistent Memory (PM)

Low Latency

Memory Semantics

Storage Features

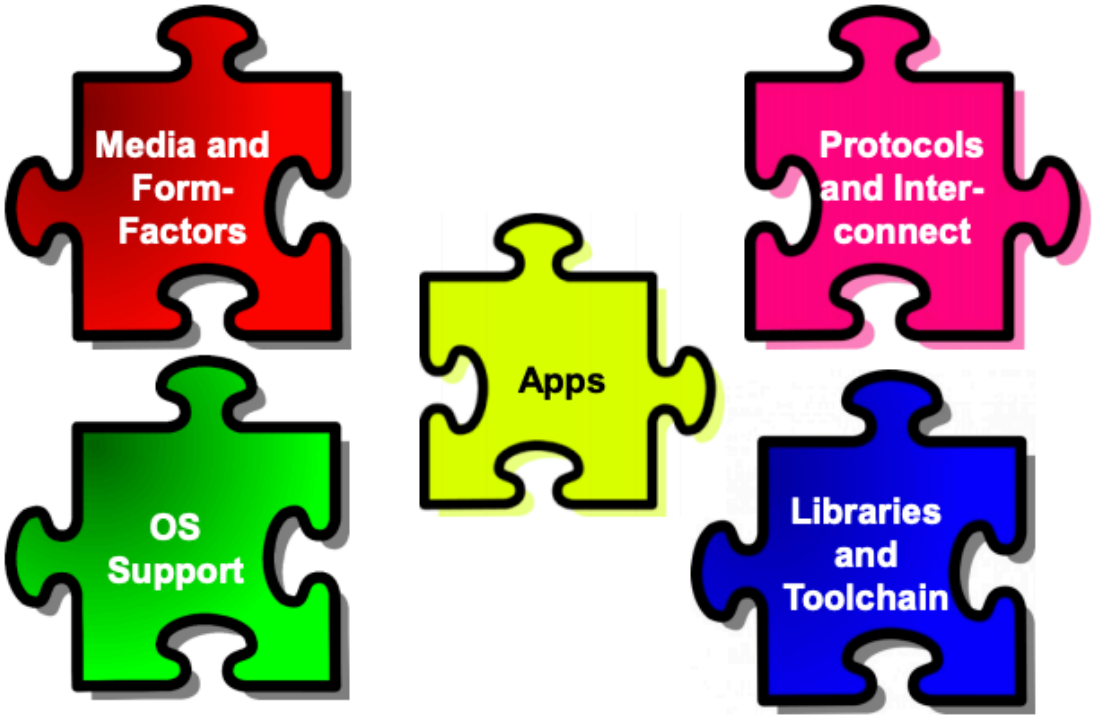
Throughput easy; latency hard

Throughput is easy

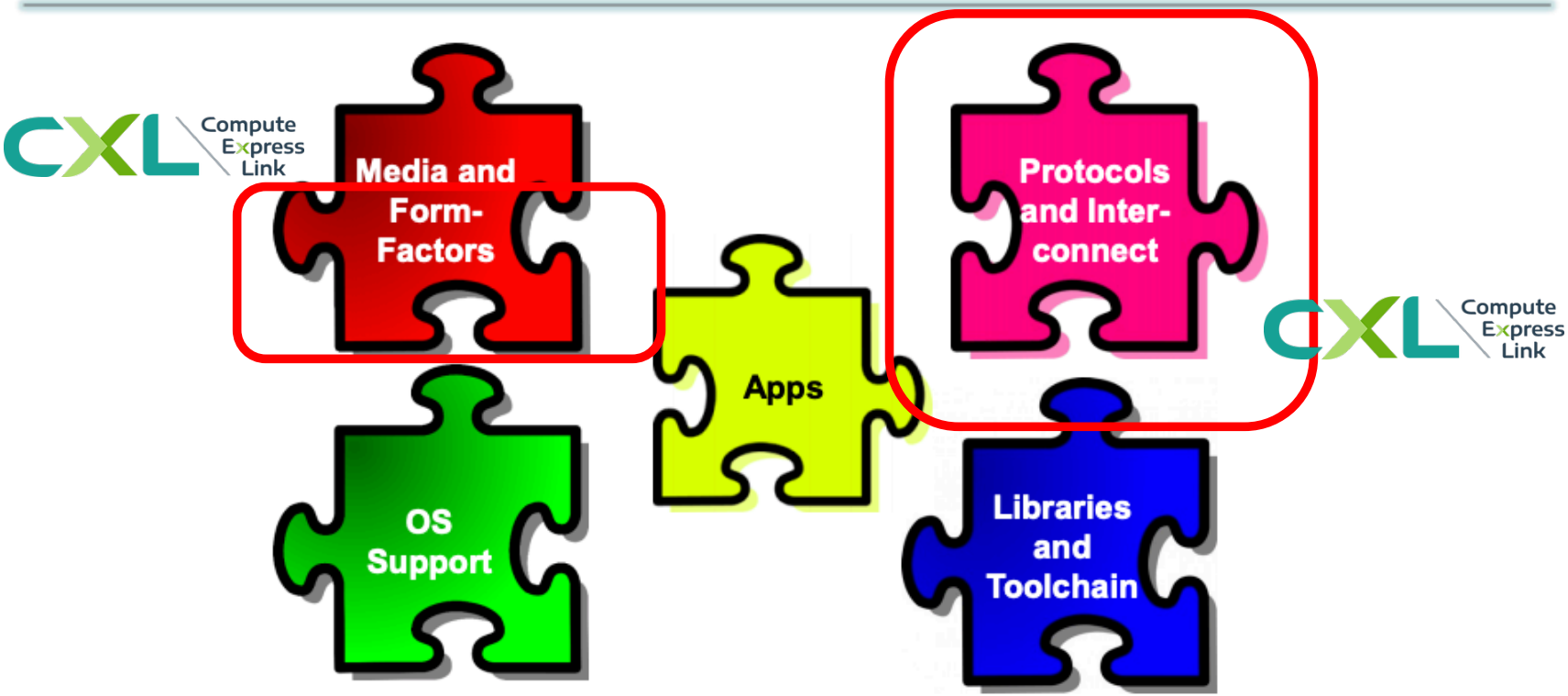
Latency is hard

Throughput is an engineering problem; latency is a physics problem!

What is Needed?



What is Needed?

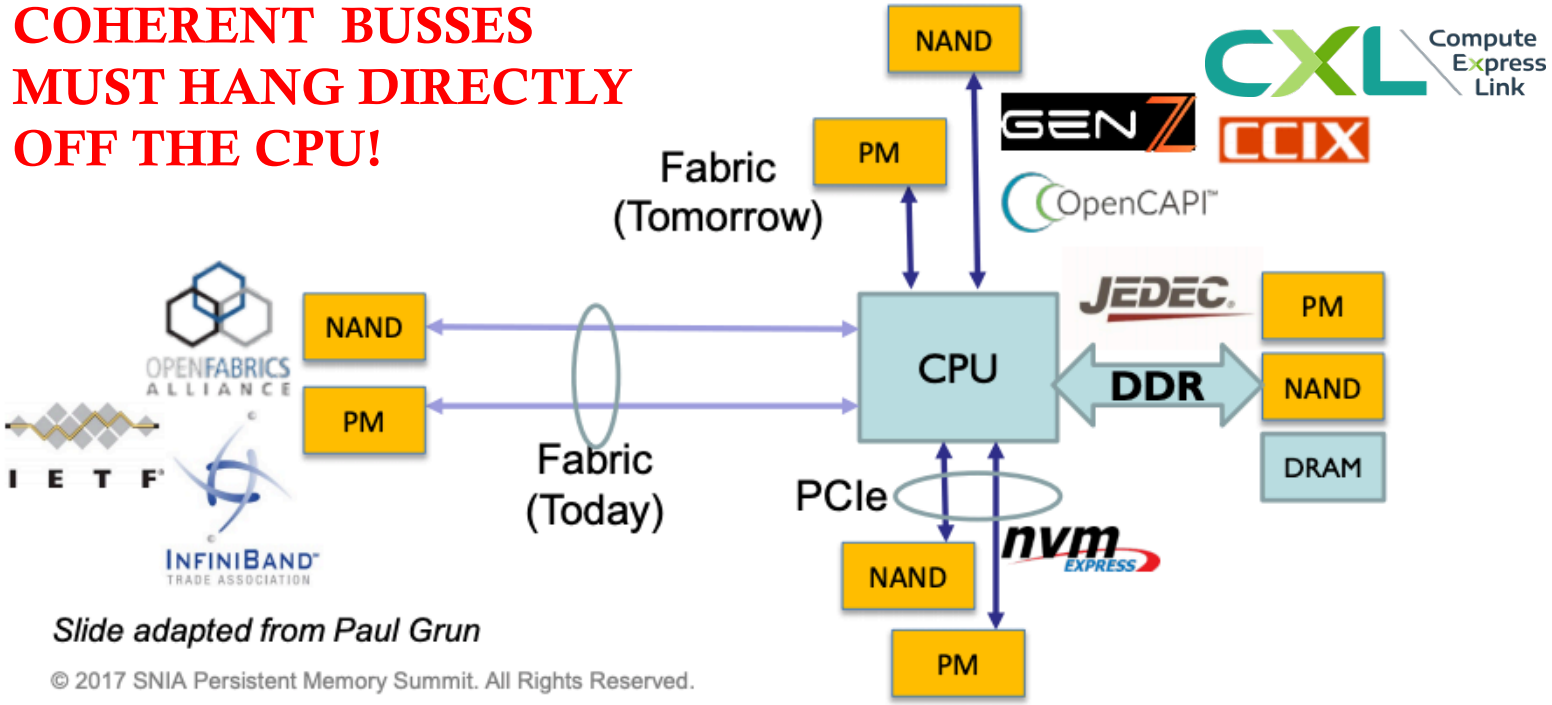


Where does PM sit?

(Answer – anywhere it wants to)

SNIA PERSISTENT MEMORY PM SUMMIT
JANUARY 18, 2017 | SAN JOSE, CA

COHERENT BUSSES MUST HANG DIRECTLY OFF THE CPU!



Slide adapted from Paul Grun

Coming Soon to a Cinema Near You!

GEN Z

A New Fabric

featuring
Optional coherency
NVMe support
Scale

Coming in 2020

CCIX

The ARMpire Strikes Back

featuring
Off the CPU bus
Accelerator support
Cache coherency
Scale?

Coming Soon??

OpenCAPI

The Return of the Big Blue

featuring
Off the CPU bus
Accelerator support
Cache coherency

Now Showing in Select Cinemas

BUT WHAT CAN I SEE TODAY???

Coming Soon to a Cinema Near You!

GEN Z

A New Fabric

featuring
Optional coherency
NVMe support
Scale

Coming in 2020

CCIX

The ARMpire Strikes Back

featuring
Off the CPU bus
Accelerator support
Cache coherency
Scale?

Coming Soon??

OpenCAPI

The Return of the Big Blue

featuring
Off the CPU bus
Accelerator support
Cache coherency

Now Showing in Select Cinemas

BUT WHAT CAN I SEE TODAY???

CXL Compute Express Link

Broad Industry support for CXL

CXL consortium - Currently 75 companies and growing

www.computeexpresslink.org

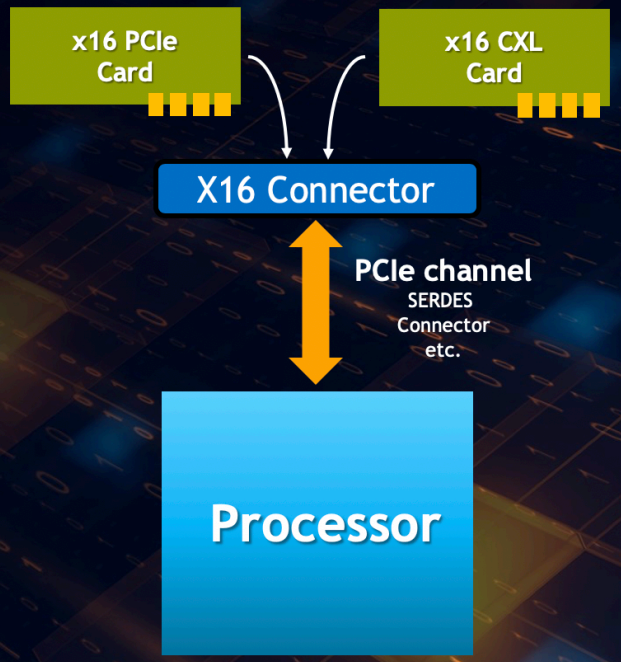
CXL Compute Express Link

Last Week!

- All the CPU vendors I care about are now CXL members.
- Same cannot be said for OpenCAPI, CCIX or Genz
- Remember, coherent buses MUST come directly out of the CPU!

What is CXL?

- CXL is an alternate protocol that runs across the standard PCIe physical layer
- CXL uses a flexible processor port that can auto-negotiate to either the standard PCIe transaction protocol or the alternate CXL transaction protocols
- First generation CXL aligns to 32 Gbps PCIe Gen5
- CXL usages expected to be key driver for an aggressive timeline to PCIe Gen6



CXL Compute Express Link

Let's break that down:

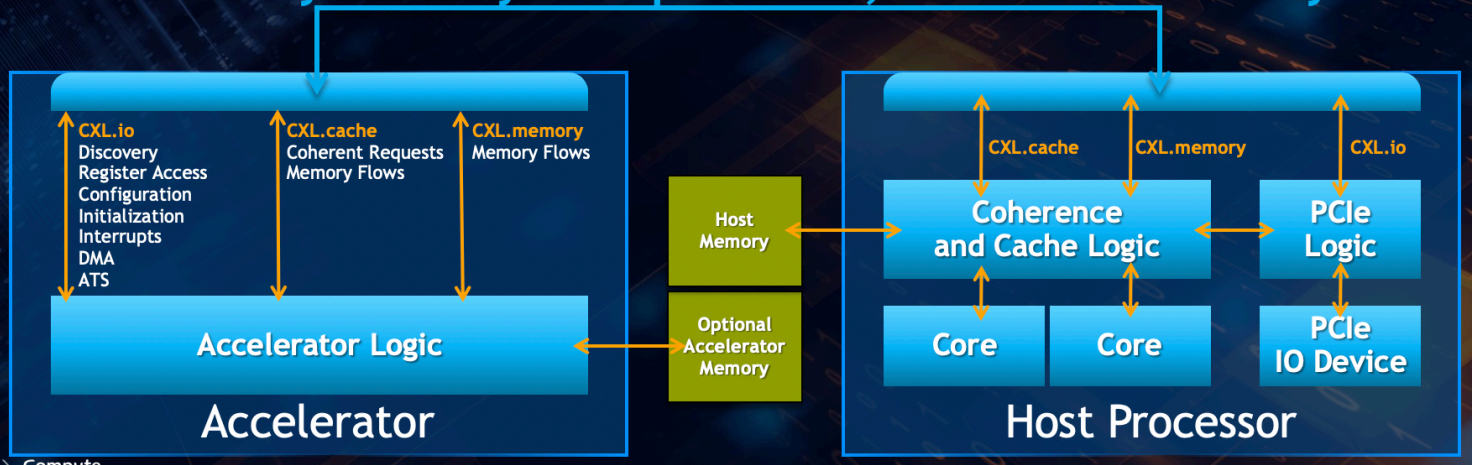
- PCIe 5.0 based. Links can be switches via UEFI and perhaps even at run-time (MEMORY_HOTPLUG anyone?).
- PCIe connectors (and form-factors) same as CXL connectors (and form-factors). As well as Add-In-Cards we can do things with storage form-factors like U.2 and EDSFF.
- Management buses that also connect to PCIe devices can also connect to CXL devices (I2C, SMBUS). Useful for management.
- Can tie into other frameworks like ACPI (for HMAT for example) and RedFISH/SwordFISH for remote management of CXL enabled servers.

CXL Protocols

The CXL transaction layer is comprised of 3 dynamically multiplexed sub-protocols on a single link:

- CXL.io - Discovery, configuration, register access, interrupts, etc.
- CXL.cache - Device access to processor memory
- CXL.memory - Processor access to device attached memory

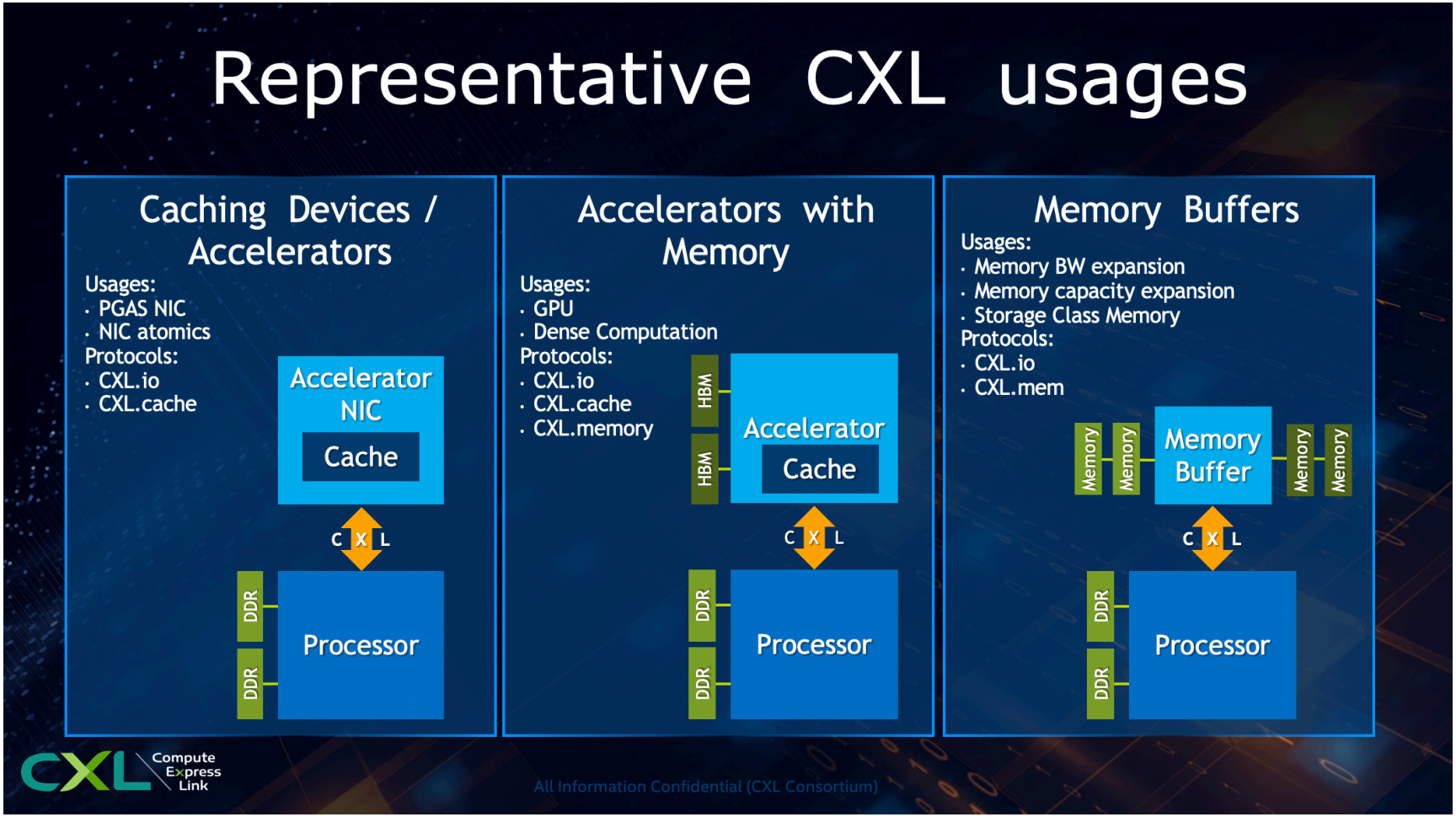
CXL - Dynamically Multiplexed IO, Cache and Memory

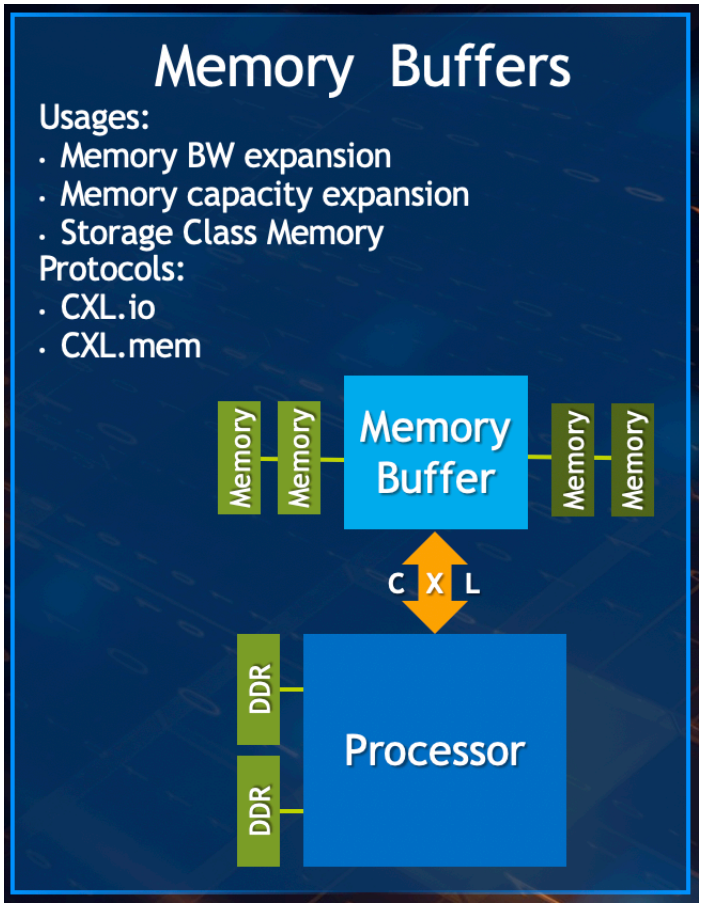


CXL Compute Express Link

Let's break that down. Three protocols on one physical layer:

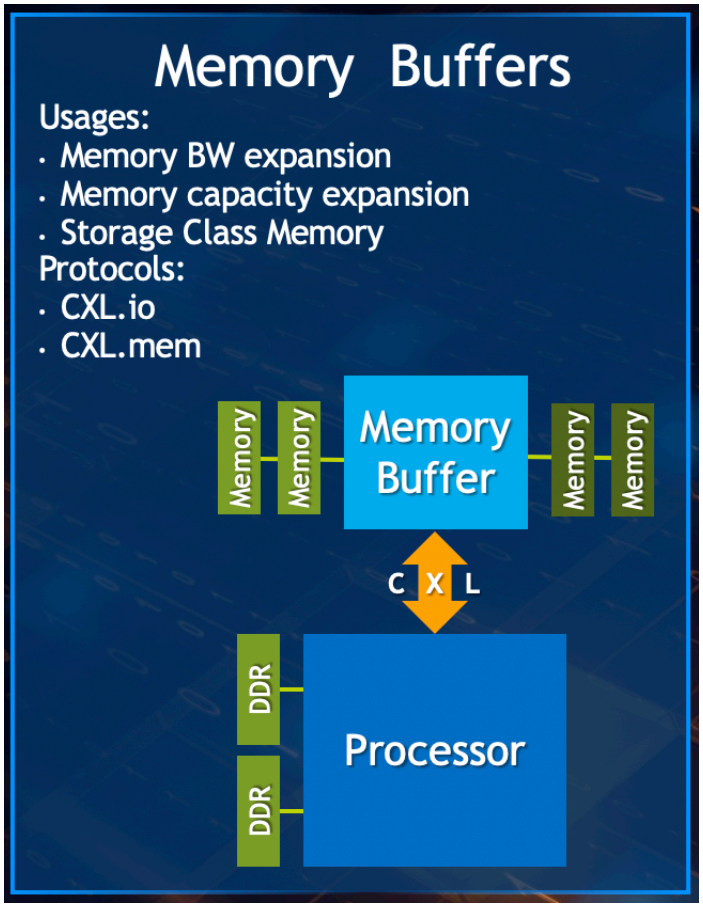
- **CXL.io**: This is PCIe Gen 5.0. All PCIe services will just work!
  - DMA
  - Interrupts (MSI/MSIX)
  - SR-IOV, ACS, ATS etc. for virtualization
  - NVM Express!!!??? – We will come back to this
- **CXL.mem**: This is the protocol by which the host CPU accesses (persistent) memory on the CXL device.
- **CXL.cache**: The is the protocol by which the CXL device accesses host memory (useful for accelerators, not covered here today).


Representative CXL usages
Caching Devices / Accelerators
Usages:
· PGAS NIC
· NIC atomics
Protocols:
· CXL.io
· CXL.cache
Accelerator
NIC
Cache
C X L
DDR
DDR
Processor
Accelerators with Memory
Usages:
· GPU
· Dense Computation
Protocols:
· CXL.io
· CXL.cache
· CXL.memory
HBM
HBM
Accelerator
Cache
C X L
DDR
DDR
Processor
Memory Buffers
Usages:
· Memory BW expansion
· Memory capacity expansion
· Storage Class Memory
Protocols:
· CXL.io
· CXL.mem
Memory
Memory
Memory Buffer
Memory
Memory
C X L
DDR
DDR
Processor
CXL Compute Express Link
All Information Confidential (CXL Consortium)



Let's break that down. Consider the right-most model:

- Essentially a NVDIMM but no longer constrained by the physical and electrical requirements of DDR and DIMMs.
- Since the form-factors are PCIe we have more options around the shape, power and heat of these solutions.
- The CXL.io allows for discovery, configuration and management (we can write a PCIe driver for these devices).
- We can put a DMA engine on the Memory Buffer and program that via PCIe to do data movement for us.
- No longer consuming DIMM slots or channels. Save all that capacity and bandwidth for standard DRAM.



Let's break that down. Consider the right-most model:

- Since CXL.io is PCIe we can write a PCIe driver for the memory buffer chip.
- If we add a DMA engine to the memory buffer chip we can program it via the driver.
- We might want to add other administration and performance related commands we can pass between processor and memory buffer chip.
- **We already have a great PCIe-based protocol for doing all this!**

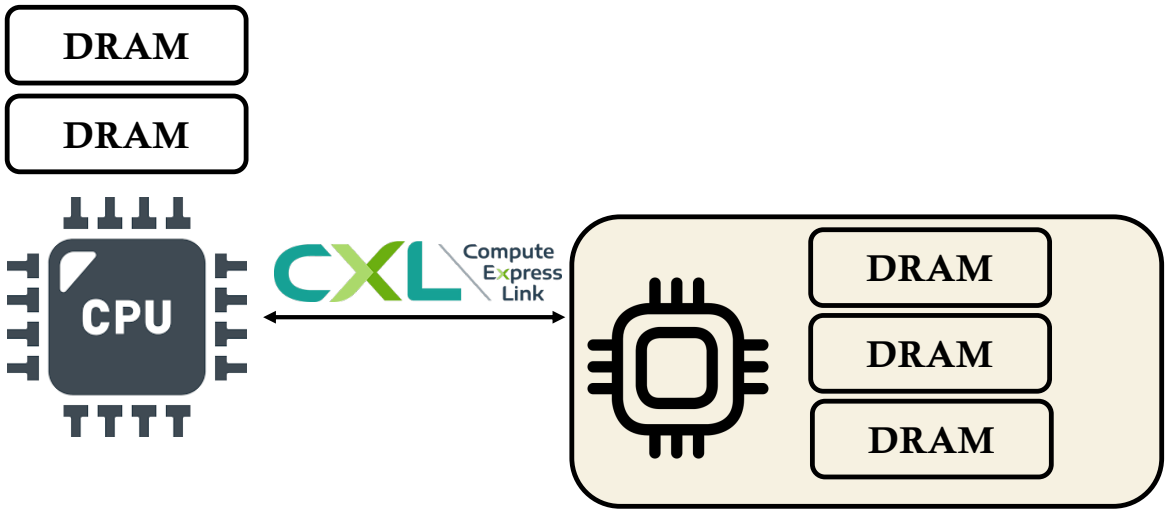
DDR NVDIMM vs CXL NVDIMM

Attribute	DDR	CXL	Comment
Form-factor	DIMM	Many	CXL has many form-factor options
DMA	No	Yes	CXL allows placement on DMA engine on device. Can be programmed via PCIe driver.
HW Virtualization	No	SR-IOV	NVDIMM can be virtualized via software which impacts performance.
Management	SMBus and MMIO	SMBus and MMIO and CXL.io	If we adopt NVMe for CXL devices we can use NVMe Management Interface (NVMe-MI).
Latency	Very Low	Low	Until we get hardware it is hard to get comparative numbers for NVDIMM vs CXL.mem to the same memory types (e.g. 3DXP).
Throughput	19GB/s	64GB/s	NVDIMM is 64 bits @ 2400MT/s/channel. CXL is (upto) 16 lanes of PCIe Gen 5 in each direction.

Linux Support for CXL

- (Persistent) Memory discovery will be done via ACPI. This can include Heterogenous Memory Attribute Tables (HMAT) to describe properties of the memory.
- The discovered memory will be added to the physical memory pool.
- We can control how and who this memory is used by to some extent by the numactl framework.
- *If* the CXL device has a DMA engine and accelerator(s) these can be programmed via a PCIe driver (perhaps NVM Express).

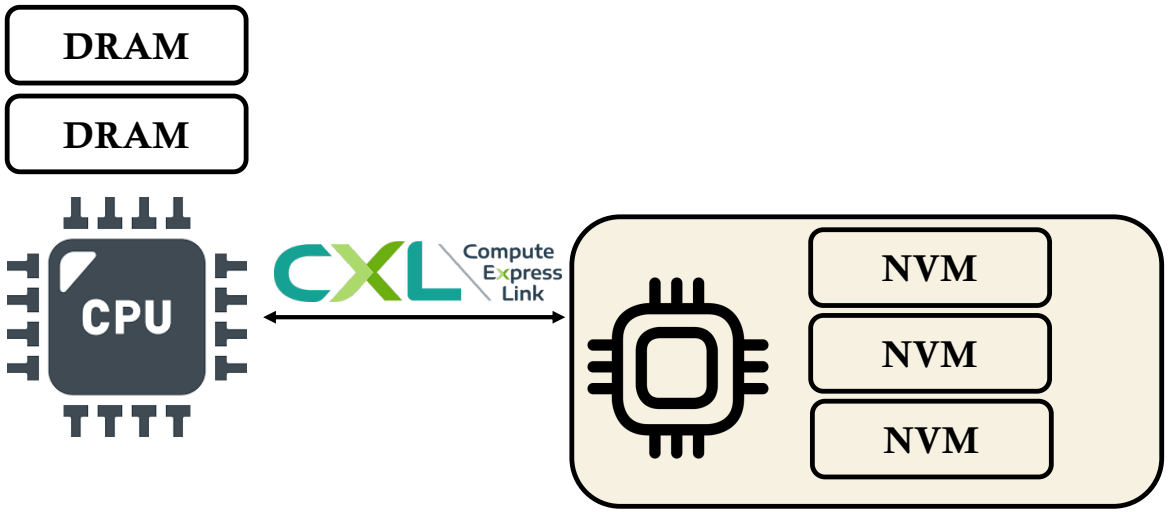
Use Case: Volatile Memory Expansion



Volatile Memory Expansion:

- Very high memory capacity systems.
- Reduces the need to scale out just for memory capacity.
- Performance of CXL.mem latency is TBD and platform specific.
- A DMA engine on CXL device could assist with data movement.

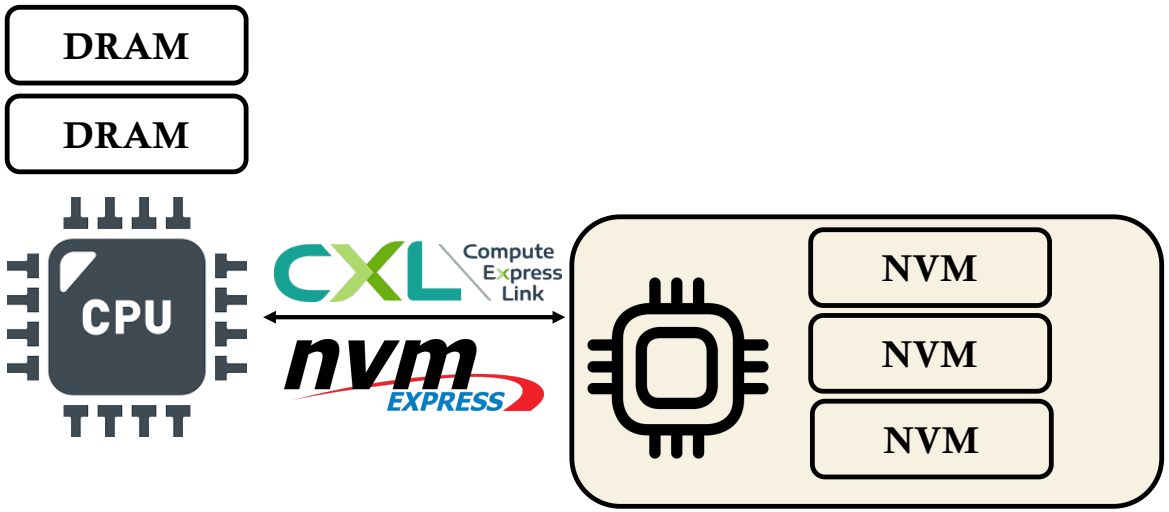
Use Case: CXL-Based NVDIMM



CXL-based NVDIMM:

- Use all the DIMM slots for DRAM, not NVDIMM.
- NVM can be managed by controller chip if needed.
- A lot more flexibility on form-factor, power etc than DDR based NVDIMM.
- A DMA engine on CXL device could assist with data movement.
- Can also be used just to expand volatile memory capacity.

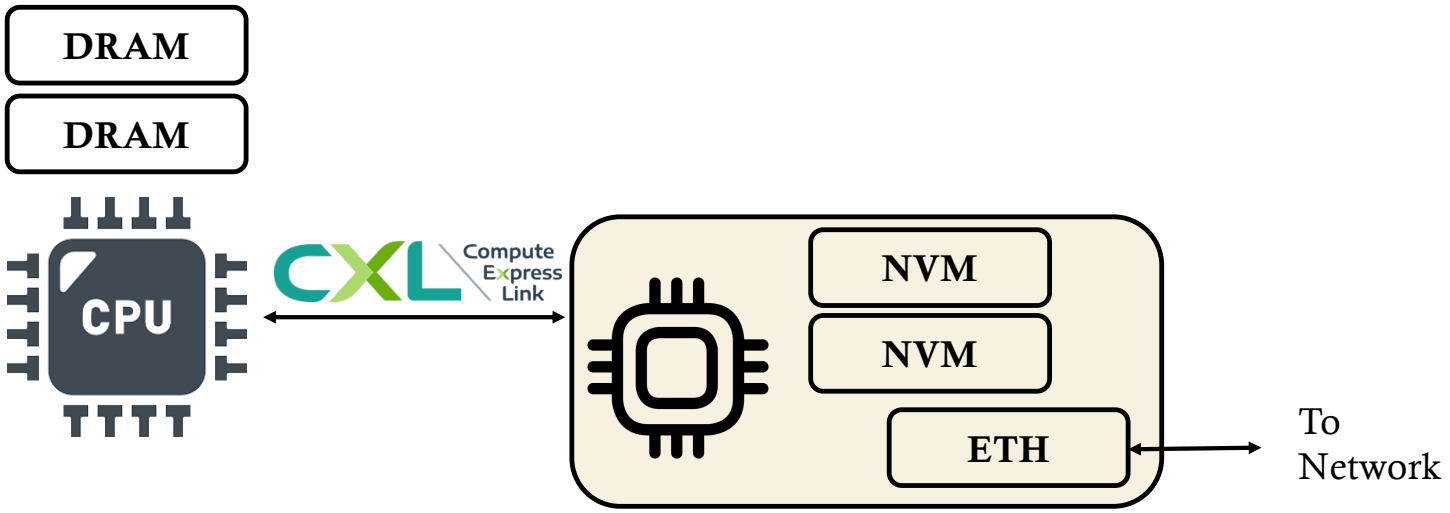
Use Case: CXL-Based NVDIMM + Accelerator



CXL-based NVDIMM+Accelerator:

- Controller chip includes compute functions (e.g. AI, search, graph database)
- Controller chip can be programmed via PCIe driver (e.g. NVMe).
- NVM can still be exposed to host and accessed via CXL.mem (volatile or persistent)

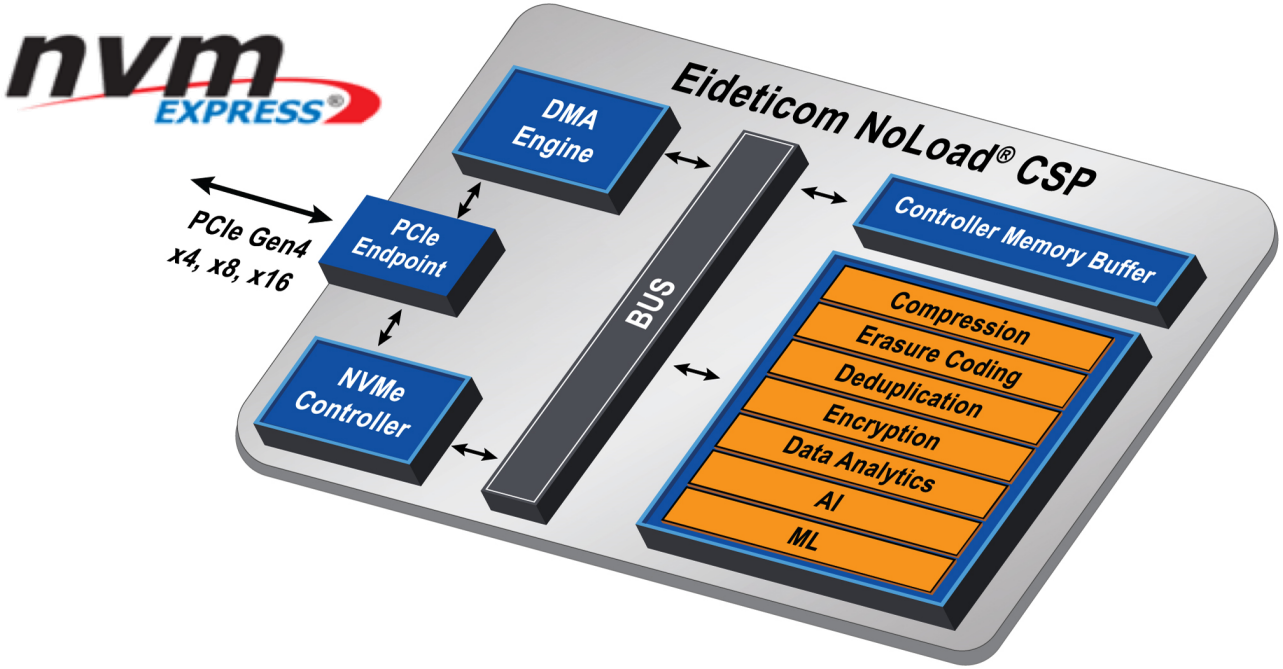
Use Case: CXL-Based NVDIMM + Remote PMEM



CXL-based NVDIMM+NIC:

- Controller chip includes network functions
- Controller chip can be programmed via PCIe driver.
- NVM can still be exposed to host and accessed via CXL.mem (volatile or persistent)
- Memory can be exposed to CPU that is actually fetched in from across the network
- Can be combined with previous example to add compute too!

NoLoad® NVMe-based Computational Storage Processor



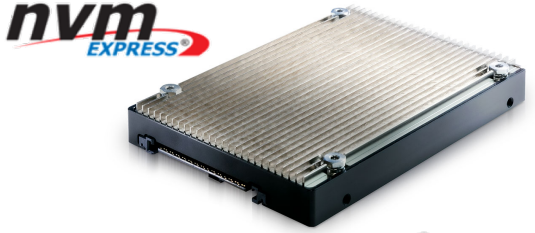
Best-In-Class Storage and Analytic Acceleration delivered via an NVMe-based Computational Storage Processor.

NoLoad® CSP – Hardware Platforms

Available Now

NoLoad® CSP U.2

- Standard U.2 NVMe form-factor: Utilizing SFF-8639 connector
- BittWare 250-U2



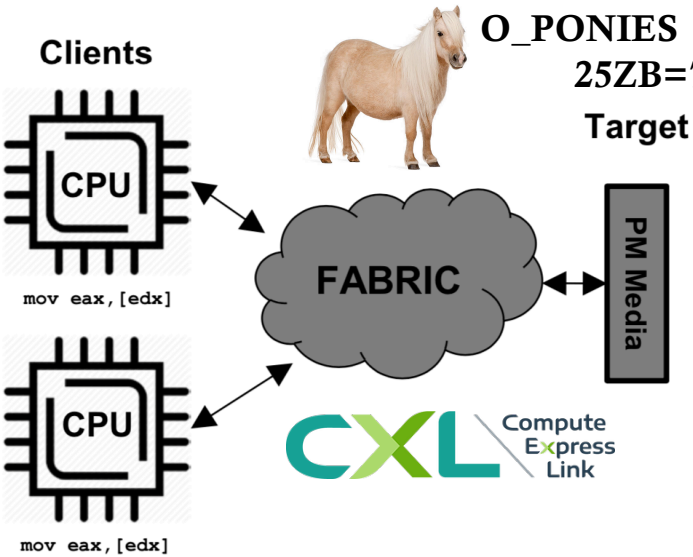
NoLoad® CSP Alveo

- Standard GPU form-factor: x16 PCIe
- Deployed on Xilinx Alveo U200, 250 or U280

CXL features for 2.0:

- Improved throughput and latency (PCIe Gen6).
- Switching via (standard) PCIe switches
- Memory pooling (allowing multiple hosts to connect to a pool of (persistent) memory.

The Holy Grail of PMoF



O_PONIES

25ZB=74.4 bits please

Target

Loads and stores on a client CPU affect Persistent Memory across the fabric!

The knights that say "c"!

We are a loooong way from here!

Conclusions

- CXL may finally be bringing some clarity to the “Star Wars” of open, coherent buses.
- Minimal software changes needed to deploy (persistent) memory on CXL.
- Adding acceleration and remote PM both possible.
- We all get a pony!

Thank You!

Eideticom HQ
3553 31st NW,
Calgary, AB,
Canada T2L 2K7

Eideticom (Bay Area)
168 South Park,
San Francisco, CA 94107
USA

www.eideticom.com

Contact: sales@eideticom.com