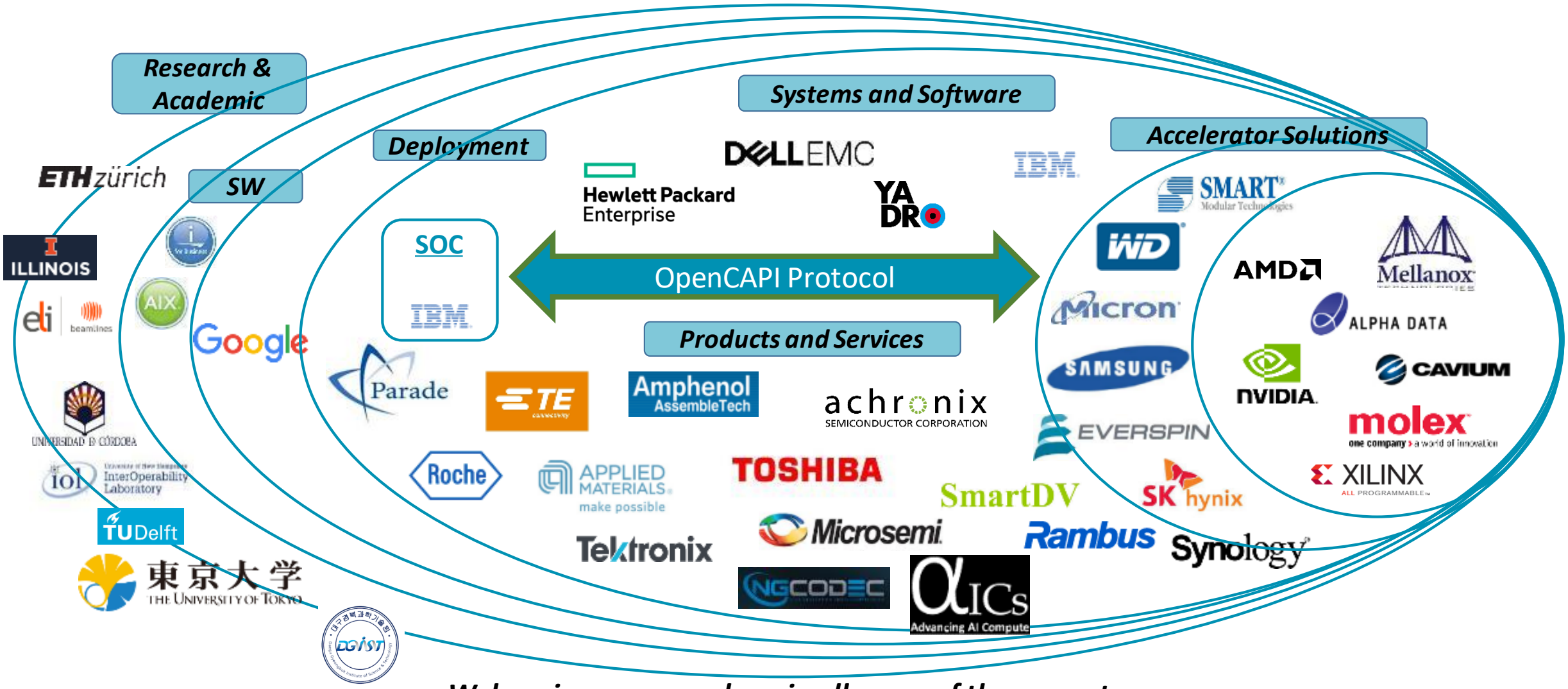
Industry Collaboration and Innovation

A data-centric approach to server design

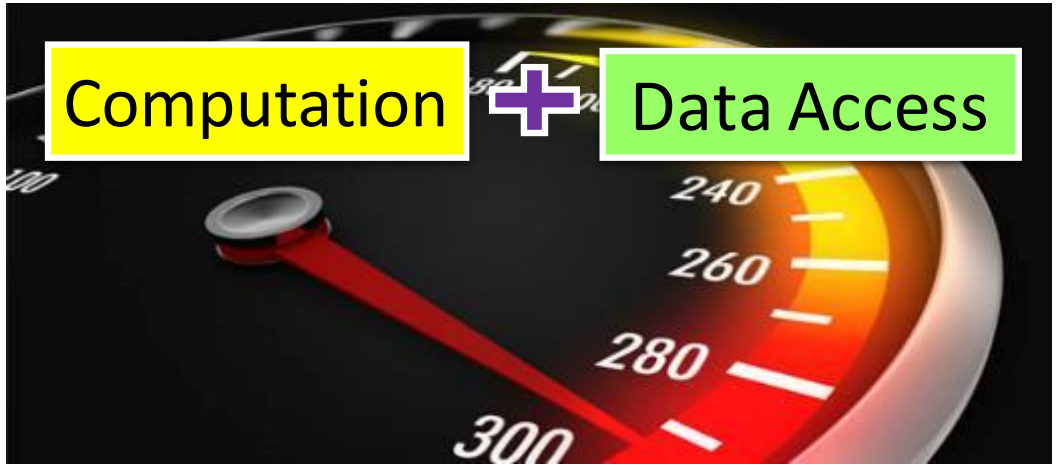
Cross Industry Collaboration and Innovation



Welcoming new members in all areas of the ecosystem

What's Driving the Creation of a High Perf. Bus

- Historical silicon technology improvements out of steam
  - More cores on a processor help but you'll never have enough; especially for today's emerging workloads (analytics, artificial intelligence, machine learning, real time analysis, etc.)
- New advanced memory technologies are changing the economics of computing
- Companies realizing the need to off load the microprocessors from routine algorithms to meet demand and improve system performance → ***Accelerated Computing***



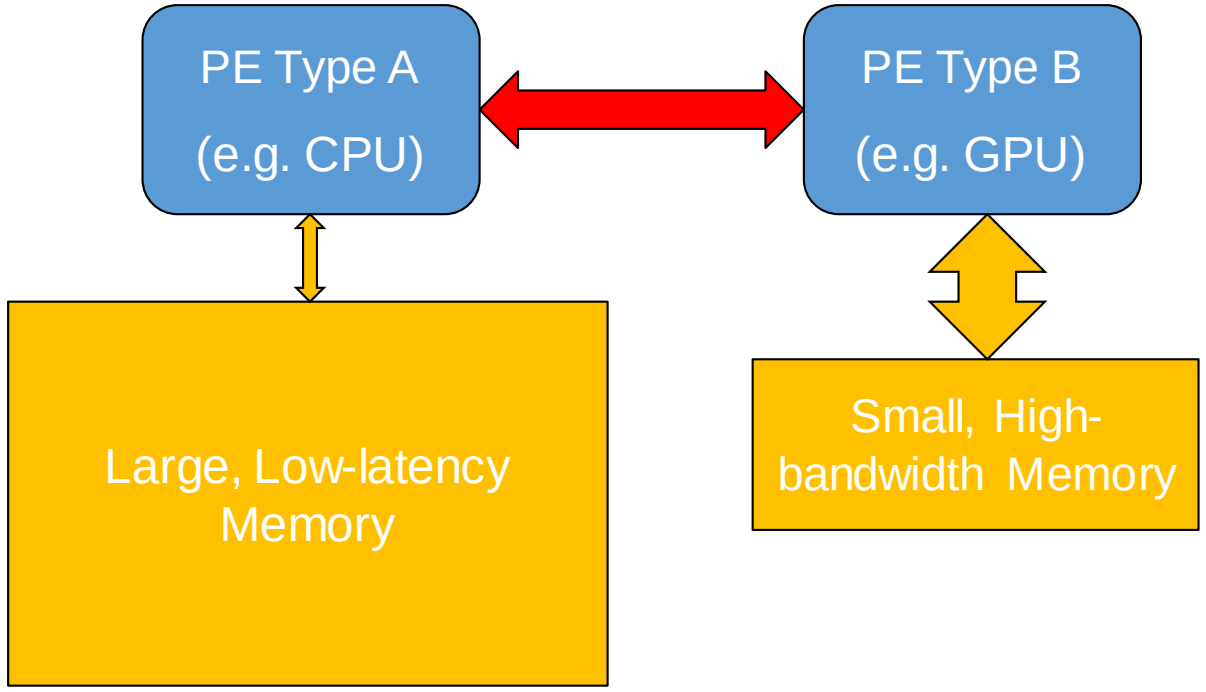
Accelerators Require High B/W Interconnects

Heterogeneous systems are attractive for efficient performance

- Each part of an application runs on the best compute location

But there are performance and programmability challenges
Desire a highly-capable interconnect between PEs

- Low-latency communication & high data bandwidth
- Fine-grained + bulk data transfers
- Consistent, unified view of memory
- Hardware cache coherence & atomic operations



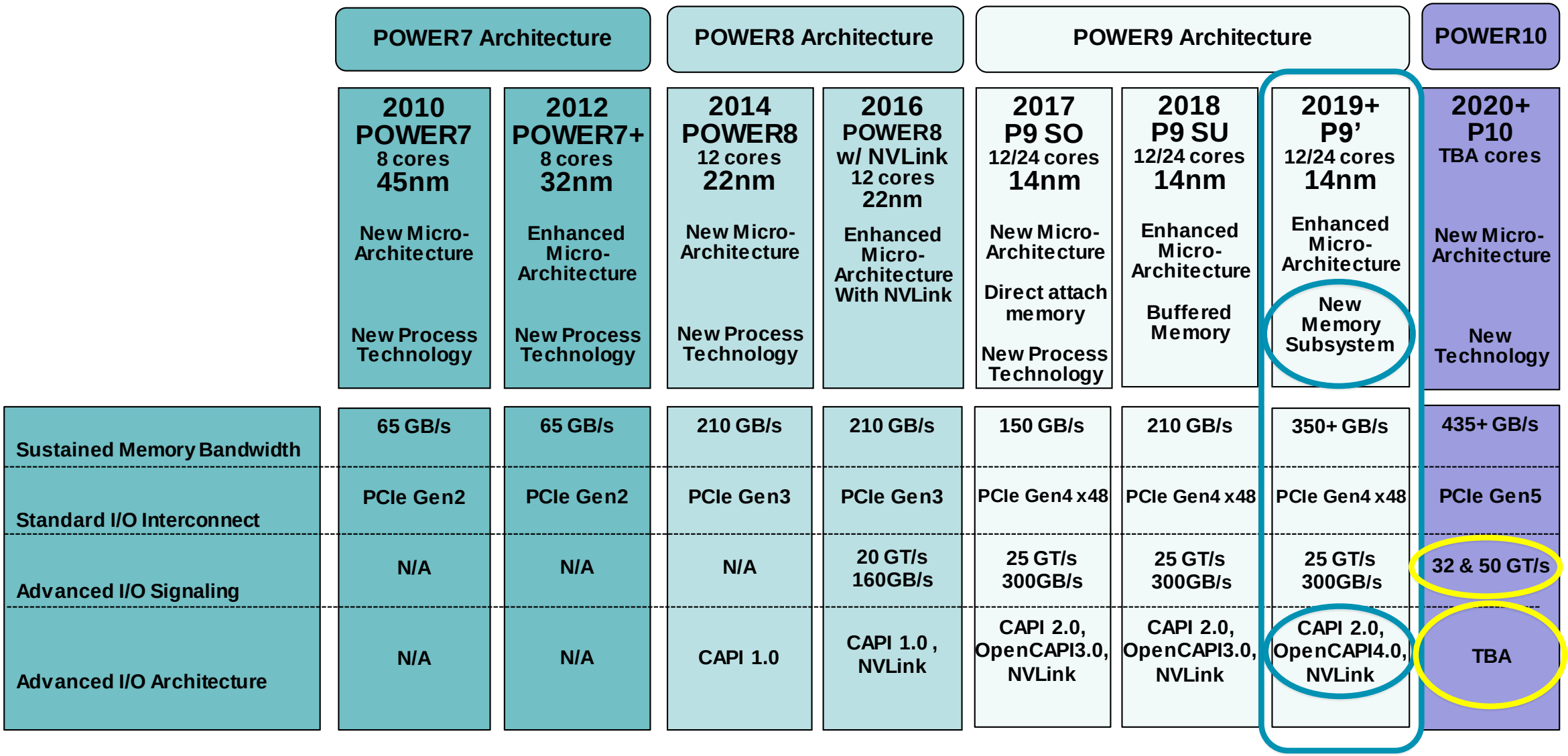
Accelerator Processor Interface Requirements

- **Industry driving these attributes:**
  - High performance (move a lot of data quickly; bandwidth, latency)
  - Introduction of device coherency requirements
  - Able to interface with complex Storage and Memory solutions
  - Fulfill various accelerator form factors (e.g., GPUs, FPGAs, ASICs)
  - Need to be architecture agnostic to enable the ecosystem growth and adaption

Why OpenCAPI and what is it?

- OpenCAPI is a new ‘bottom’s up’ IO standard
- Key Attributes of OpenCAPI 3.0
  - *Open IO Standard – Choice for developers and others to contribute and grow an ecosystem*
  - *Coherent interface – Microprocessor memory, accelerator and caches share the same memory space*
  - *Architecture agnostic – Capable going beyond Power Architecture*
    - *Not tied to Power – Architecture Agnostic*
  - *High performance – No OS/Hypervisor/FW Overhead for Low Latency and High Bandwidth*
  - *Ease of programing*
  - *Ease of implementation with minimal accelerator design overhead*
  - *Ideal for accelerated computing and SCM including various form factors (FPGA, GPU, ASIC, TPU, etc.)*
  - *Optimized for within a single system node*
  - *Supports heterogeneous environment – Use Cases*
- OpenCAPI 3.1
  - *Applies OpenCAPI technology for use of standard DRAM off the microprocessor*
  - *Based on an Open Memory Interface (OMI)*
  - *Further tuned for extreme lower latency*

Proposed POWER Processor Technology and I/O Roadmap



Statement of Direction, Subject to Change

POWER9 Processor

New Core Microarchitecture

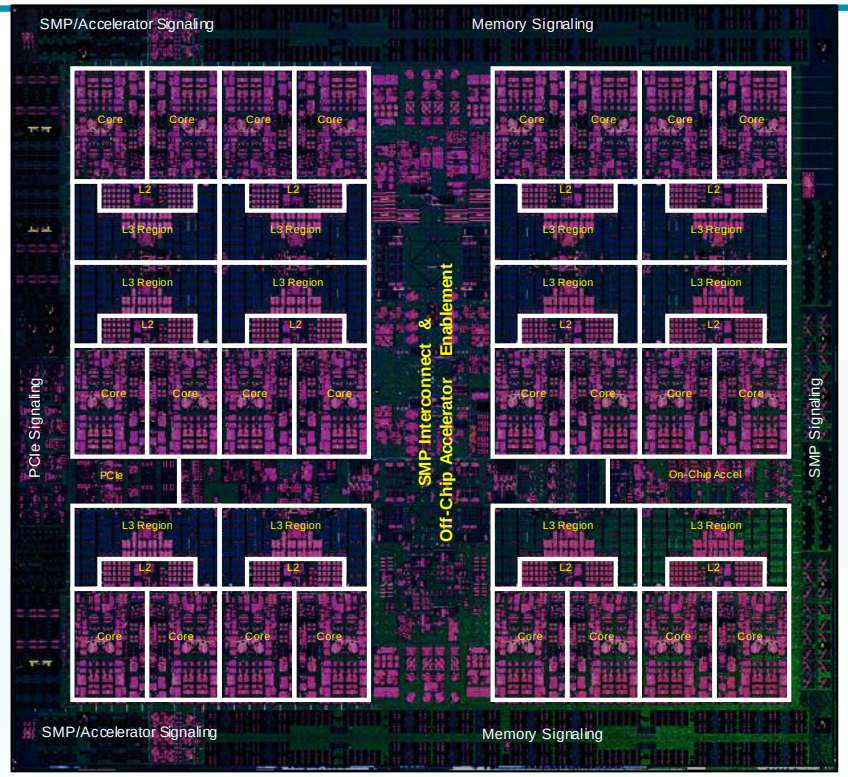
- Stronger thread performance
- Efficient agile pipeline
- POWER ISA v3.0

Enhanced Cache Hierarchy

- 120MB NUCA L3 architecture
- 12 x 20-way associative regions
- Advanced replacement policies
- Fed by 7 TB/s on-chip bandwidth

Cloud + Virtualization Innovation

- Quality of service assists
- New interrupt architecture
- Energy Scale (Workload optimized frequency)



14nm finFET Semiconductor Process

- Improved device performance and reduced energy
- 17 layer metal stack and eDRAM
- 8.0 billion transistors

Leadership Hardware Acceleration Platform

- Enhanced on-chip acceleration
- Nvidia NVLink 2.0: High bandwidth and advanced new features (25G)
- CAPI 2.0: Coherent accelerator and storage attach (PCIe G4)
- OpenCAPI: Improved latency and bandwidth, open interface (25G)

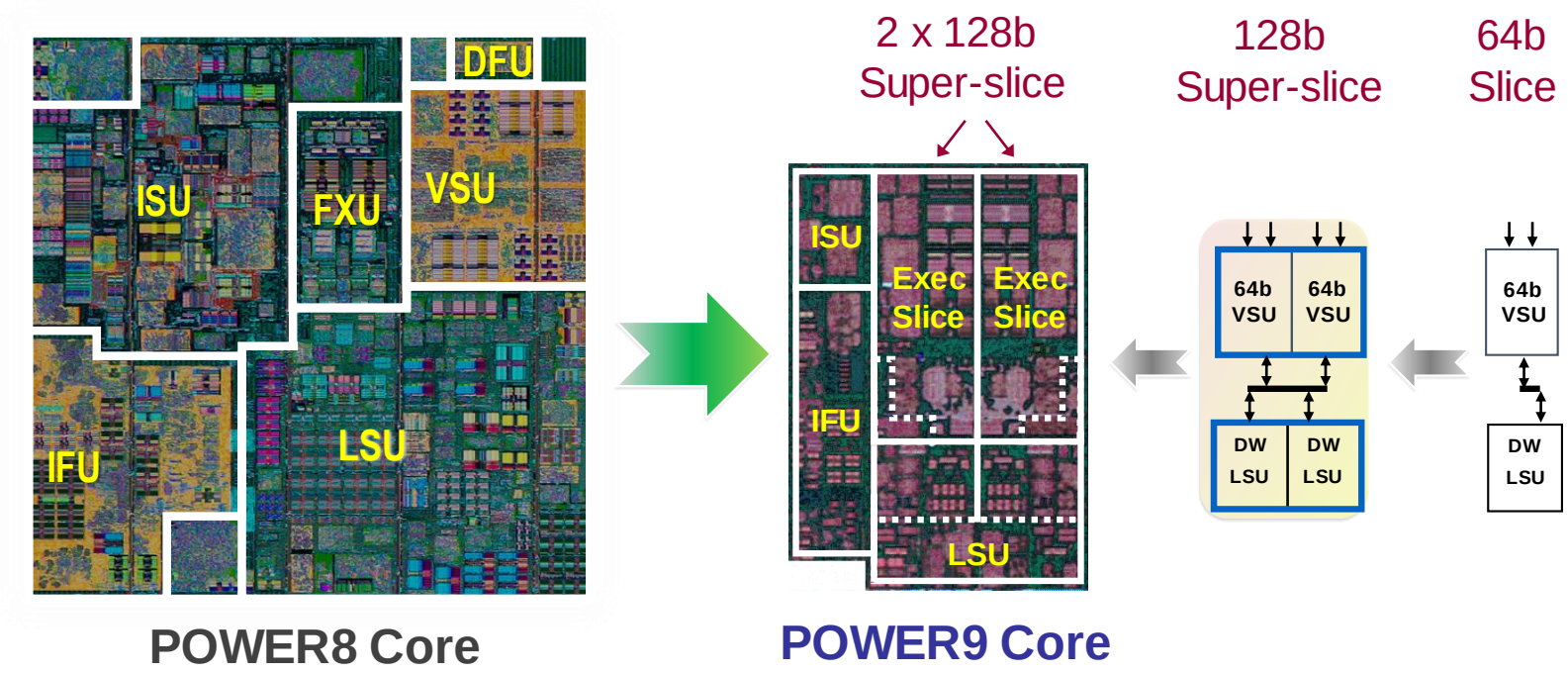
State of the Art I/O Subsystem

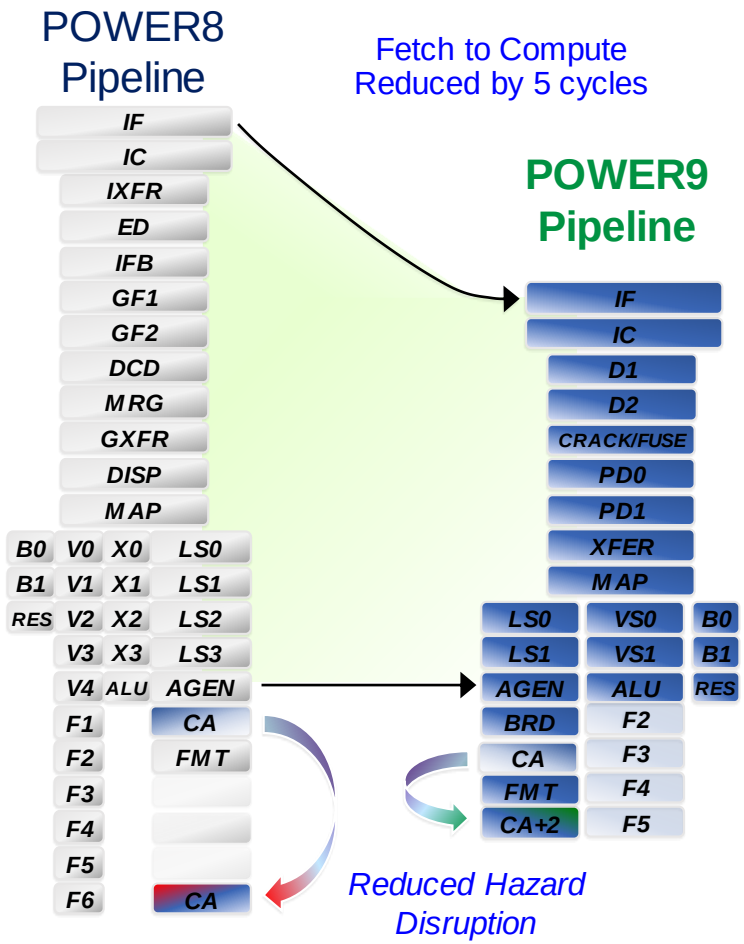
- PCIe Gen4 – 48 lanes

High Bandwidth Signaling Technology

- 16 GT/s interface
  - Local SMP
- 25 GT/s Common Link interface
  - Accelerator, remote SMP

POWER9 Core Microarchitecture





Re-factored Core Provides Improved Efficiency & Workload Alignment

- Enhanced pipeline efficiency with modular execution and intelligent pipeline control
- Increased pipeline utilization with symmetric data-type engines: Fixed, Float, 128b, SIMD
- Shared compute resource optimizes data-type interchange

POWER9 High Bandwidth I/O

- ***PCIe Gen4***
- ***CAPI 2.0***
- ***NVLink 2.0***
- ***OpenCAPI 3.0***

Silicon Die
Various packages
(scale-out, scale-up)

8 and 16Gbps PHY
Protocols Supported
- PCIe Gen3 x16 and PCIe Gen4 x8
- CAPI 2.0 on PCIe Gen4

PCIe Gen4

P9 25Gbs

25Gbps PHY
Protocols Supported
- OpenCAPI 3.0
- NVLink 2.0

OpenCAPI vs I/O Device Driver – Because minimizing SW Path Length is crucial for performance

Typical I/O Model Flow using Device Driver

Total ~13µs for data prep

DD Call → Copy or Pin Source Data → MMIO Notify Accelerator → Acceleration → Poll / Intr Completion → Copy or Unpin Result Data → Return From DD Completion

300 Instructions, 10,000 Instructions → 7.9µs

Application Dependent, but Equal to below

3,000 Instructions, 1,000 Instructions, 1,000 Instructions → 4.9µs

Flow with a OpenCAPI Model

Total 0.36µs

Shared Memory Notify Accelerator → Acceleration → Shared memory completion or fast thread wake up

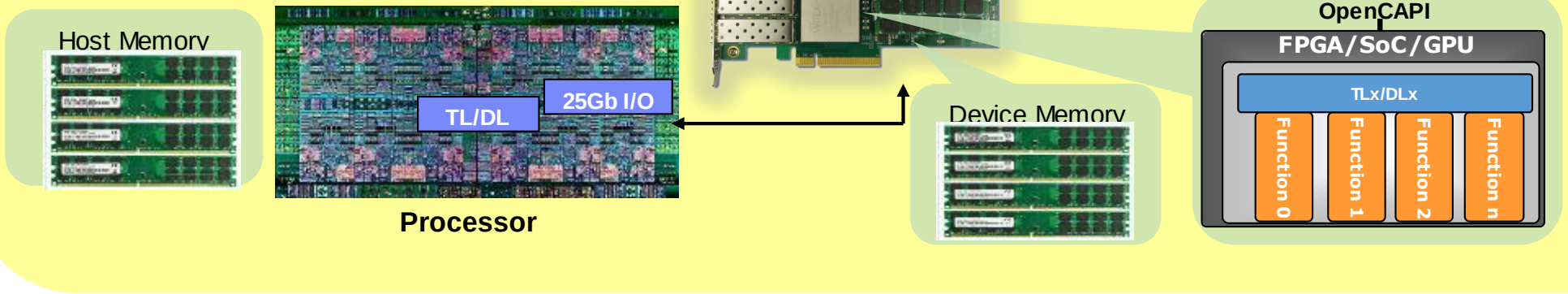
400 Instructions

0.3µs

Application Dependent, but Equal to above

100 Instructions

0.06µs



Use Cases - A True Heterogeneous Architecture Built Upon OpenCAPI

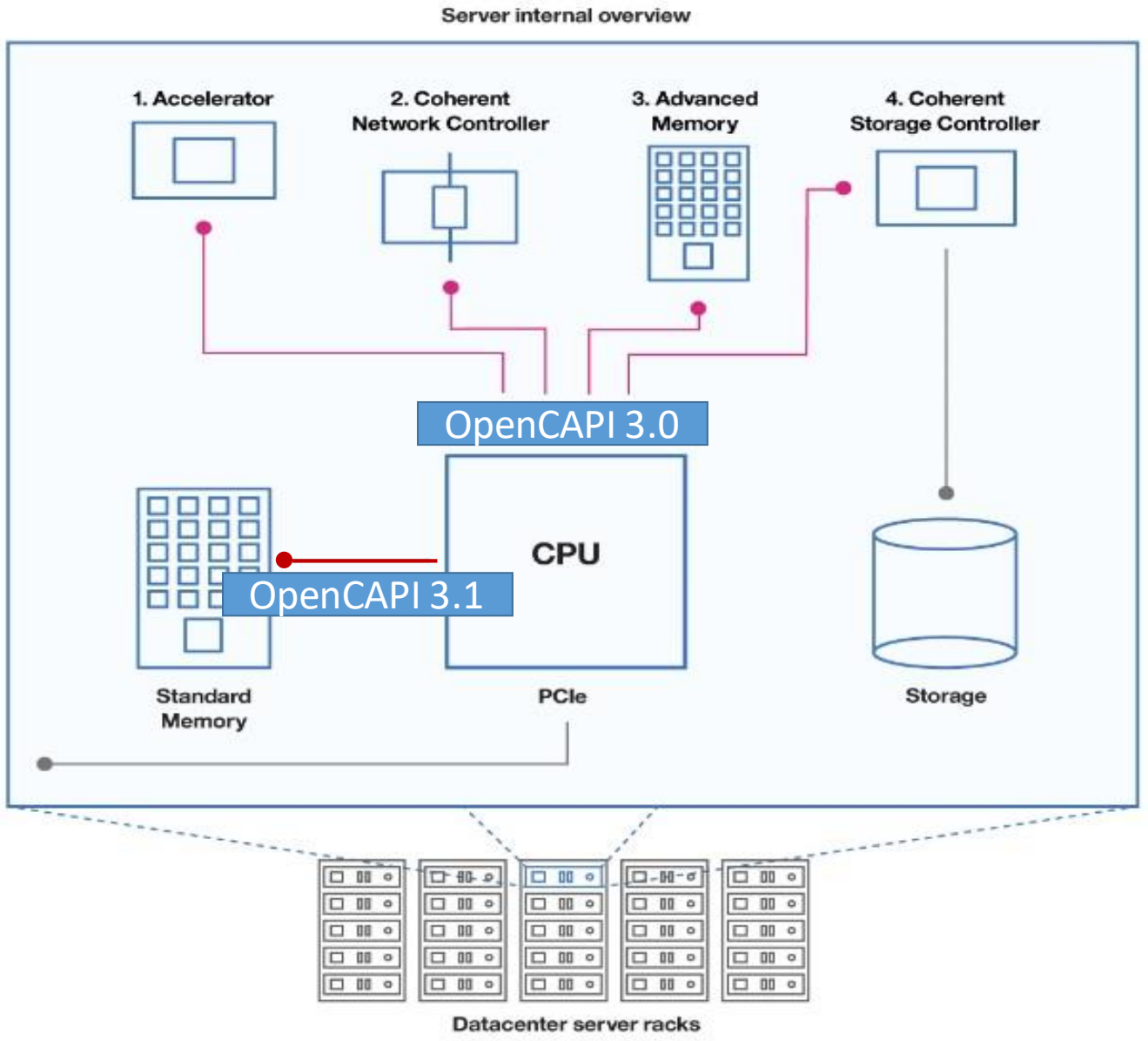
OpenCAPI

1. **Accelerators:** The performance, virtual addressing and coherence capabilities allow FPGA and ASIC accelerators to behave as if they were integrated into a custom microprocessor.

2. **Coherent Network Controller:** OpenCAPI provides the bandwidth that will be needed to support rapidly increasing network speeds Network controllers based on virtual addressing can eliminate software overhead without the programming complexity usually associated with user-level networking protocols.

3. **Advanced Memory:** OpenCAPI allows system designers to take full advantage of emerging memory technologies to change the economics of the datacenter.

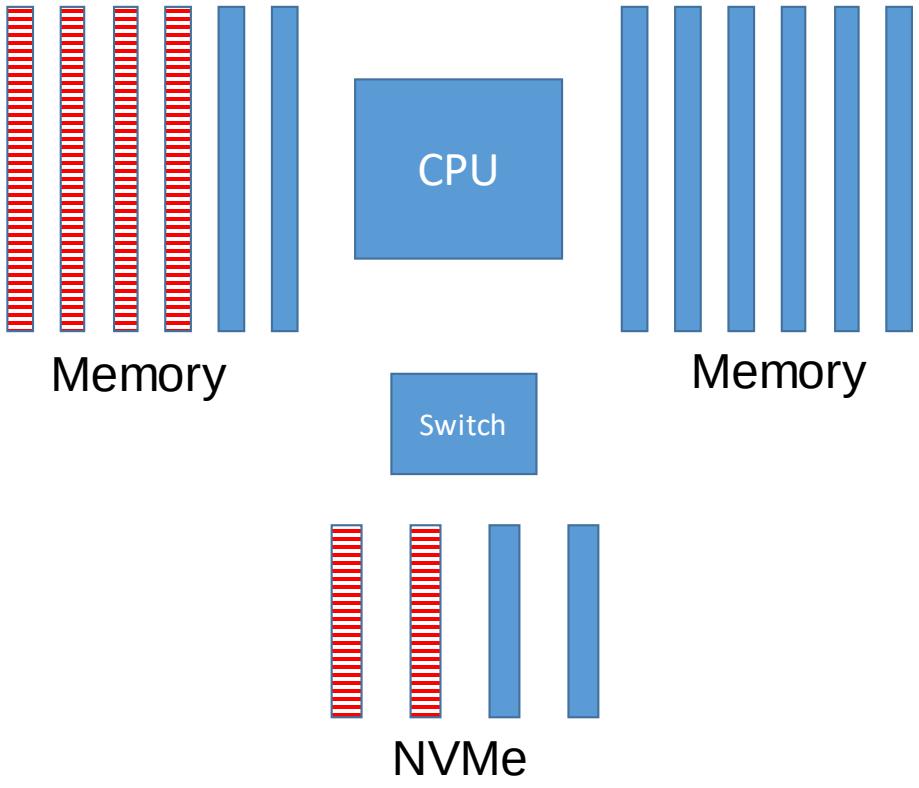
4. **Coherent Storage Controller:** OpenCAPI allows storage controllers to bypass kernel software overhead, enabling extreme IOPS performance without wasting valuable CPU cycles.



OpenCAPI specifications are downloadable from the website at www.opencapi.org

- **Register**
- **Download**

Industry Usage of SCM



Industry initiatives put SCM where it's easy

- Easier from hardware development perspective
- Very limiting for end users
- Allows interfaces to be used as control point

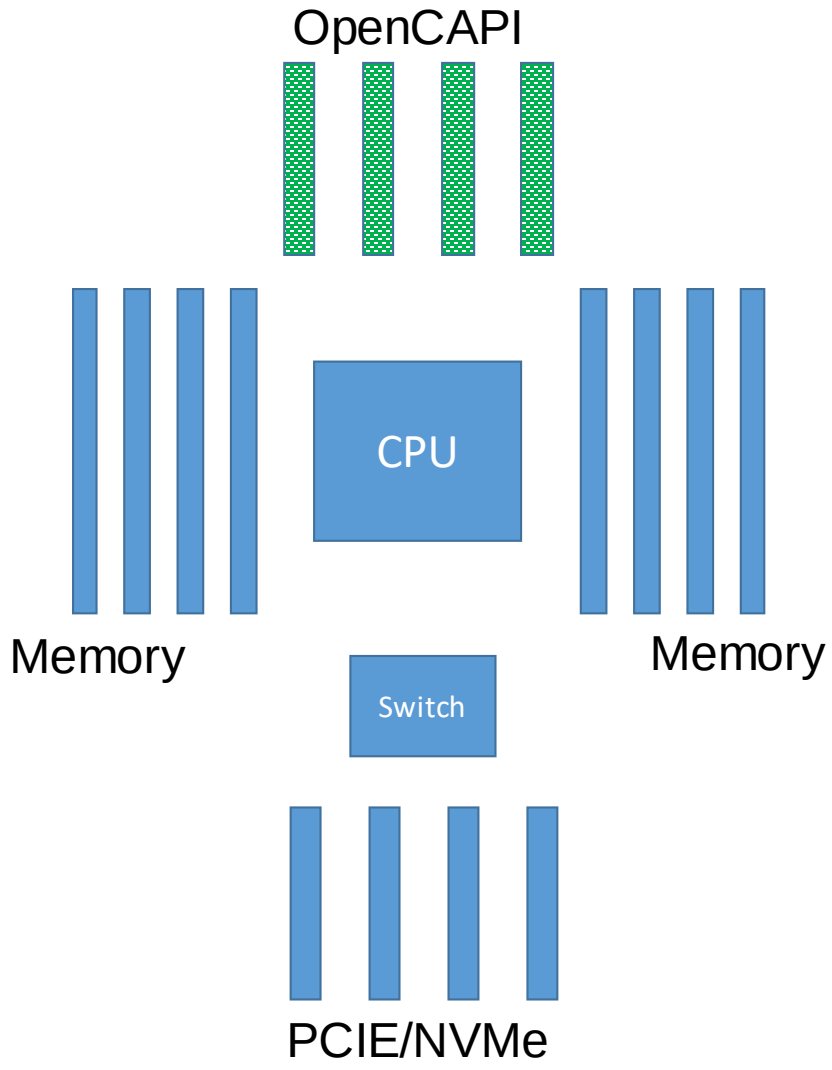
SCM accessed via NVMe:

- **Latency advantage reduced by software stack**
- **Bandwidth limited by PCIe infrastructure**
- **IOPs limited by CPUs consumed to run NVMe stack**

SCM accessed over DDR memory buses:

- **Load/Store access model improves over NVMe**
  - **Eliminates CPU cycles spent on NVMe stack**
  - **Reduces latency by removing SW pathlength**
- **Creates DRAM vs SCM configuration tradeoffs**
  - **Capacity and bandwidth spread across 2 tiers**
  - **Both tiers are sub-optimized**

OpenCAPI & SCM



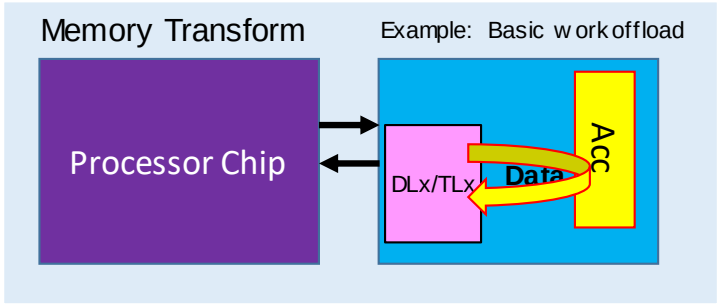
Advantages of OpenCAPI attach for SCM:

Range of Access Semantics
- User-Mode block transfer (like CAPI Flash)
- Memory-like Near Storage

Optimized Latency
- Low Latency hardware path
- Elimination or reduction of software pathlength

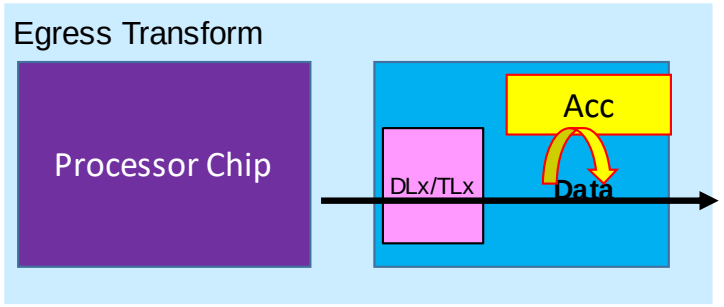
Tiered Memory solutions without compromise
- Full DDR DRAM capacity and bandwidth
- Up to 100GB/sec per socket for SCM access

Acceleration Paradigms with Great Performance

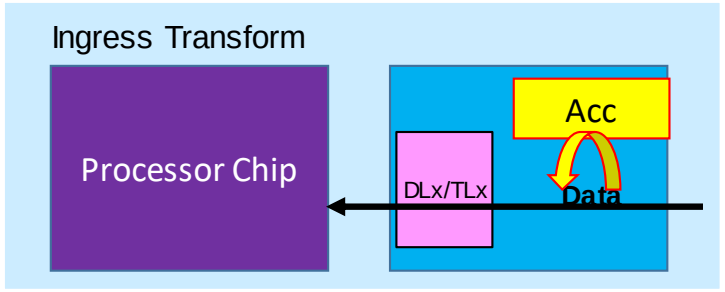


OpenCAPI is ideal for acceleration due to Bandwidth to/from accelerators, best of breed latency, and flexibility of an Open architecture

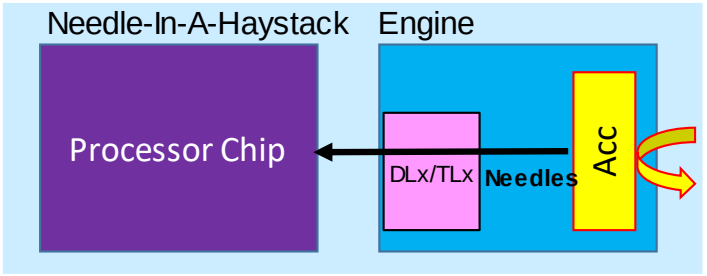
Examples: Machine or Deep Learning such as Natural Language processing, sentiment analysis or other Actionable Intelligence using OpenCAPI attached memory



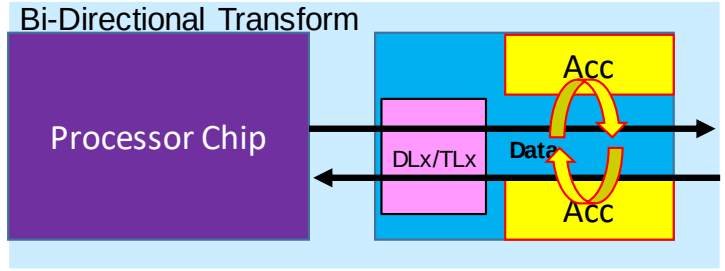
Examples: Encryption, Compression, Erasure prior to delivering data to the network or storage



Examples: Video Analytics, Network Security, Deep Packet Inspection, Data Plane Accelerator, Video Encoding (H.265), High Frequency Trading etc



Examples: Database searches, joins, intersections, merges
Only the Needles are sent to the processor



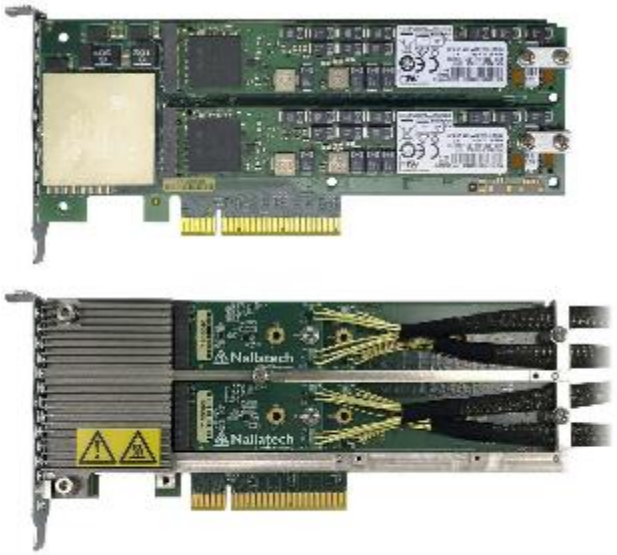
Examples: NoSQL such as Neo4J with Graph Node Traversals, etc

Why do I care about Virtual Addressing?

- **An OpenCAPI device operates in the virtual address spaces of the applications that it supports**
  - Eliminates kernel and device driver software overhead
  - Allows device to operate on application memory without kernel-level data copies/pinned pages
  - Simplifies programming effort to integrate accelerators into applications
  - Culmination => Improves Accelerator Performance

- **The Virtual-to-Physical Address Translation occurs in the host CPU**
  - Reduces design complexity of OpenCAPI accelerator development
  - Makes it easier to ensure **interoperability between OpenCAPI devices and different CPU architectures**
  - **Security** - Since the OpenCAPI device never has access to a physical address, this **eliminates the possibility of a defective or malicious device accessing memory locations belonging to the kernel or other applications that it is not authorized to access**

OpenCAPI and CAPI2 Adapters

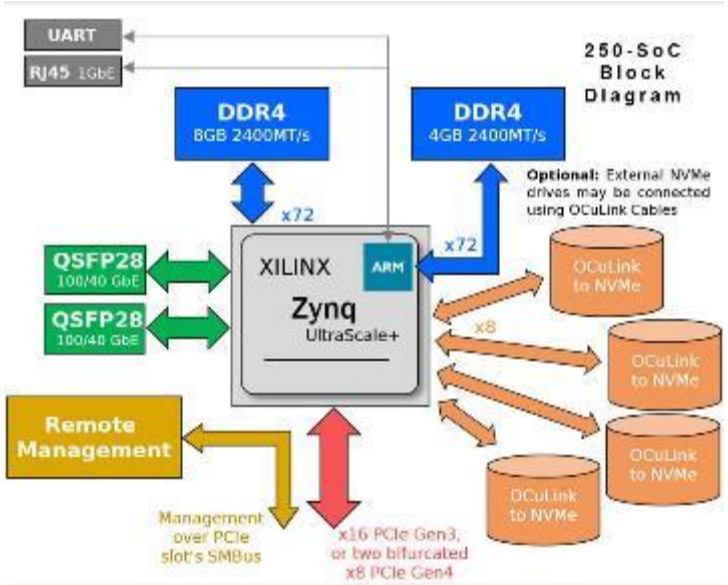


Nallatech 250S+

Storage Expansion

- Xilinx US+ KU15P FPGA
- 4 GB DDR4
- PCIe Gen4 x8 and CAPI2
- 4x M.2 Slots
- M.2 to MiniSAS or Oculink for U.2 drive support

CAPI Flash API, Accelerated DB, Burst Buffer



Nallatech 250-SoC

Multipurpose Converged Network / Storage

- Xilinx Zynq US+ ZU19EG FPGA
- 8/16 GB DDR4, 4/8 GB DDR4 ARM
- PCIe Gen4 x8 or Gen3 x16, CAPI2
- 4 x8 Oculink Ports support NVMe, Network, or OpenCAPI
- 2 100Gb QSFP28 Cages

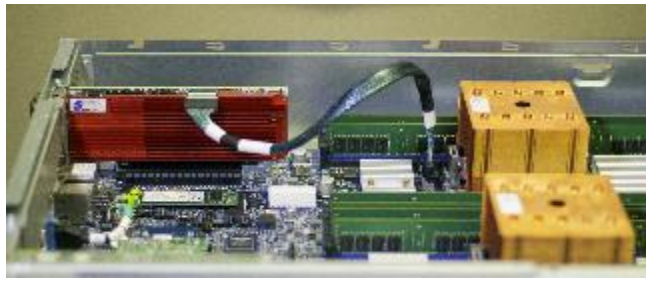
Mellanox Innova2

Network + FPGA

- Xilinx US+ KU15P FPGA
- Mellanox CX5 NIC
- 16 GB DDR4
- PCIe Gen4 x8
- 2 25Gb SFP Cages
- X8 25Gb/s OpenCAPI Support

Network Acceleration (NFV, Packet Classification), Security Acceleration

OpenCAPI and CAPI2 Adapters



AlphaData ADM-9V3

High Performance Reconfigurable Computing

- Xilinx US+ VU3P FPGA
- 16 / 32 GB DDR4
- PCIe Gen3 x16 or Gen4 x8 and CAPI2
- 2 QSFP28 Cages
- X8 25Gb/s OpenCAPI SlimSAS

Data Center, Network Accel, HPC, HFT

AlphaData ADM-9H7

Large FPGA with 8GB HBM

- Xilinx US+ VU37P FPGA + HBM
- 8GB High Bandwidth Memory
- PCIe Gen4 x8 or Gen3 x16, CAPI2
- 2 x8 25 Gb/s OpenCAPI Ports (support up to 50 GB/s)
- 4 100Gb QSFP28 Cages

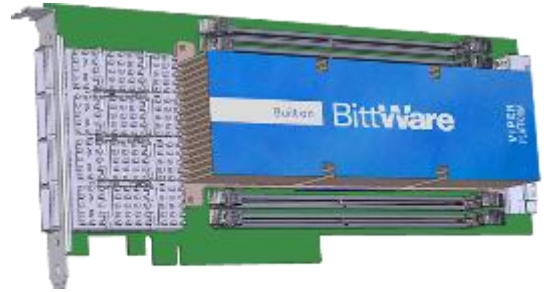
AlphaData ADM-9H3

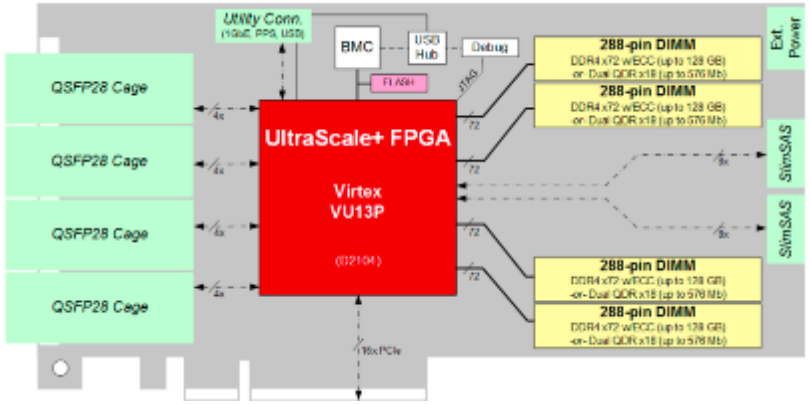
Large FPGA with 8GB HBM

- Xilinx Virtex US+ VU33P-3 FPGA + HBM
- 8GB High Bandwidth Memory
- PCIe Gen4 x8 or Gen3 x16, CAPI2
- 1 x8 25 Gb/s OpenCAPI Ports (support up to 50 GB/s)
- 2 100Gb QSFP28 Cages

ML/DL, Inference, System Modeling, HPC

PROVISIONAL





Bittware XUPVV4

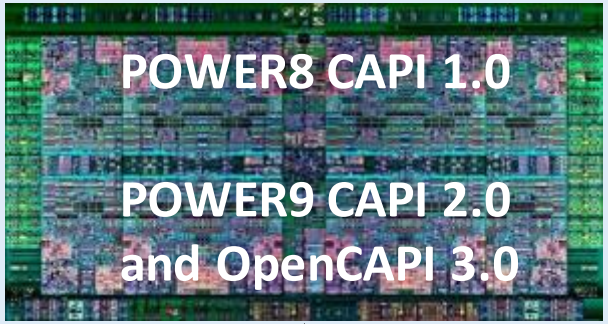
Massive FPGA

- Xilinx US+ VU13P FPGA
- 4 288-pin DIMM Slots, DDR4 or Dual QDR
- Up to 512GB DDR4
- PCIe Gen3 x16, CAPI2 Capable
- 4 100Gb QSFP28 Cages
- 2 x8 25Gb/s OpenCAPI Support

Optimized for Thermal Performance for Large acceleration in the Data Center

CAPI and OpenCAPI Performance

	CAPI 1.0 PCIE Gen3 x8 Measured BW @8Gb/s	CAPI 2.0 PCIE Gen4 x8 Measured BW @16Gb/s	OpenCAPI 3.0 25 Gb/s x8 Measured BW @25Gb/s
128B DMA Read	3.81 GB/s	12.57 GB/s	**22.1 GB/s**
128B DMA Write	4.16 GB/s	11.85 GB/s	**21.6 GB/s**
256B DMA Read	N/A	13.94 GB/s	**22.1 GB/s**
256B DMA Write	N/A	14.04 GB/s	**22.0 GB/s**



Xilinx

KU60/VU3P FPGA

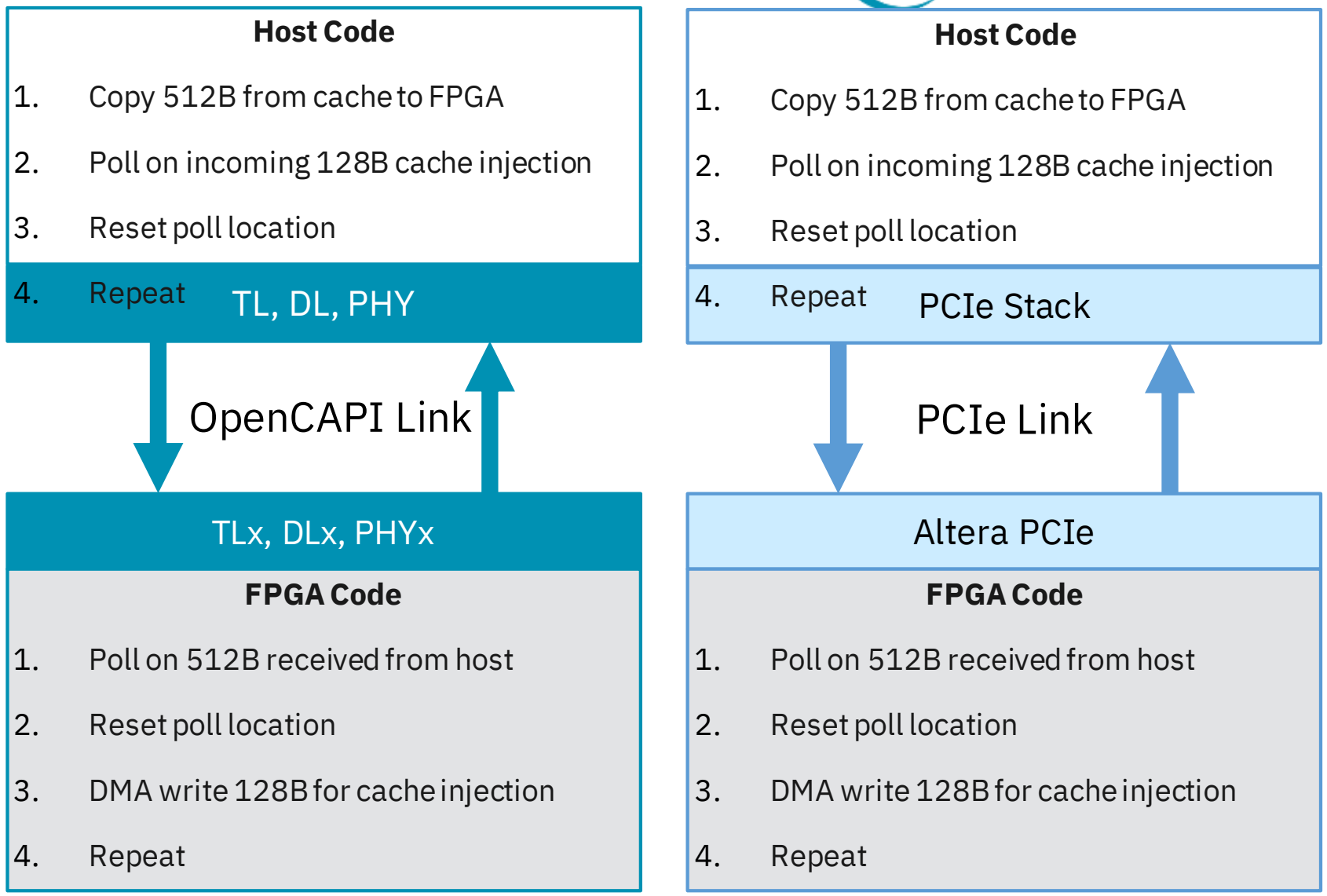
POWER8
Introduced in 2013

POWER9
Second Generation

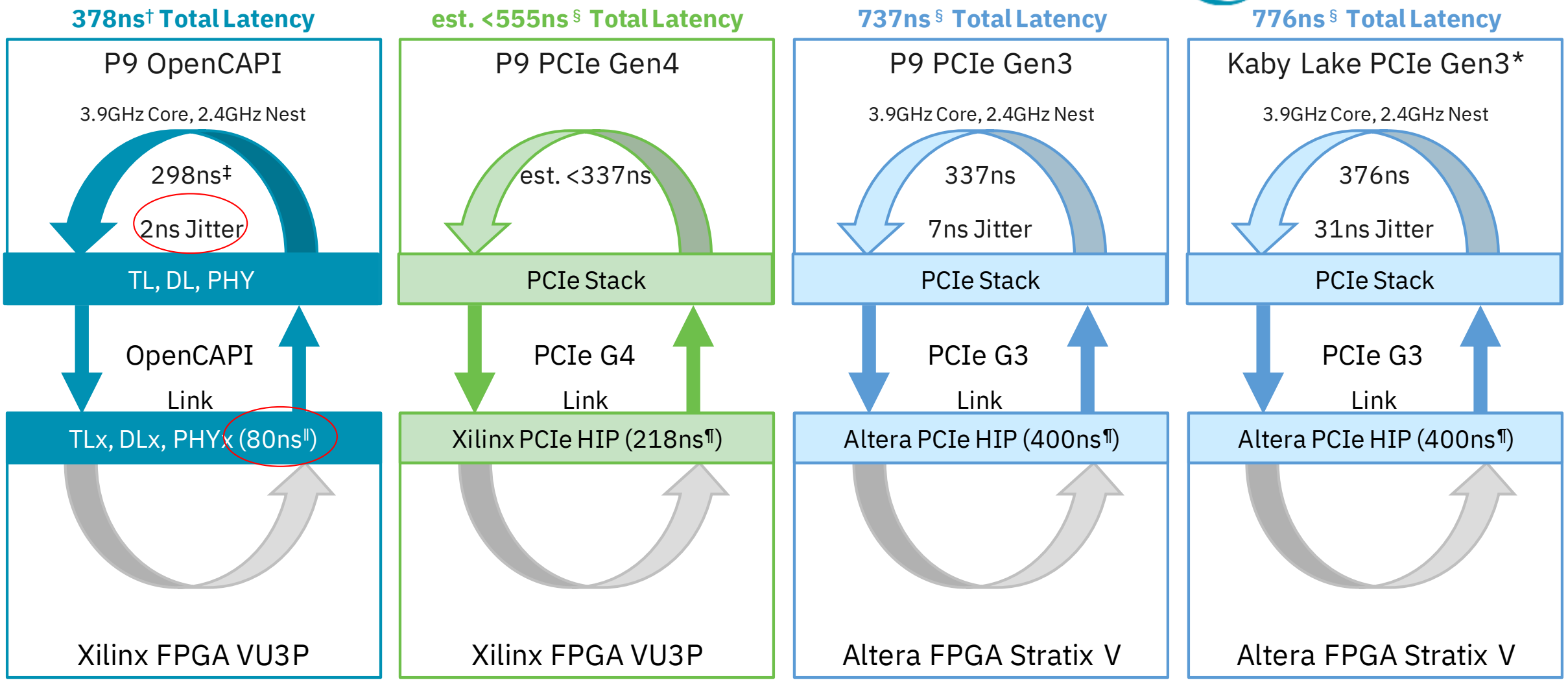
POWER9
Open Architecture with a Clean Slate Focused on Bandwidth and Latency

Latency Pingpong Test

- Simple workload created to simulate communication between system and attached FPGA
- Bus traffic recorded with protocol analyzer and PowerBus traces
- Response times and statistics calculated



Latency Test Results



* Intel Core i7 7700 Quad-Core 3.6GHz (4.2GHz TurboBoost)

† Derived from round-trip time minus simulated FPGA app time

‡ Derived from round-trip time minus simulated FPGA app time and simulated FPGA TLx/DLx/PHYx time

§ Derived from measured CPU turnaround time plus vendor provided HIP latency

‖ Derived from simulation

¶ Vendor provided latency statistic

CAPI SNAP

What if…

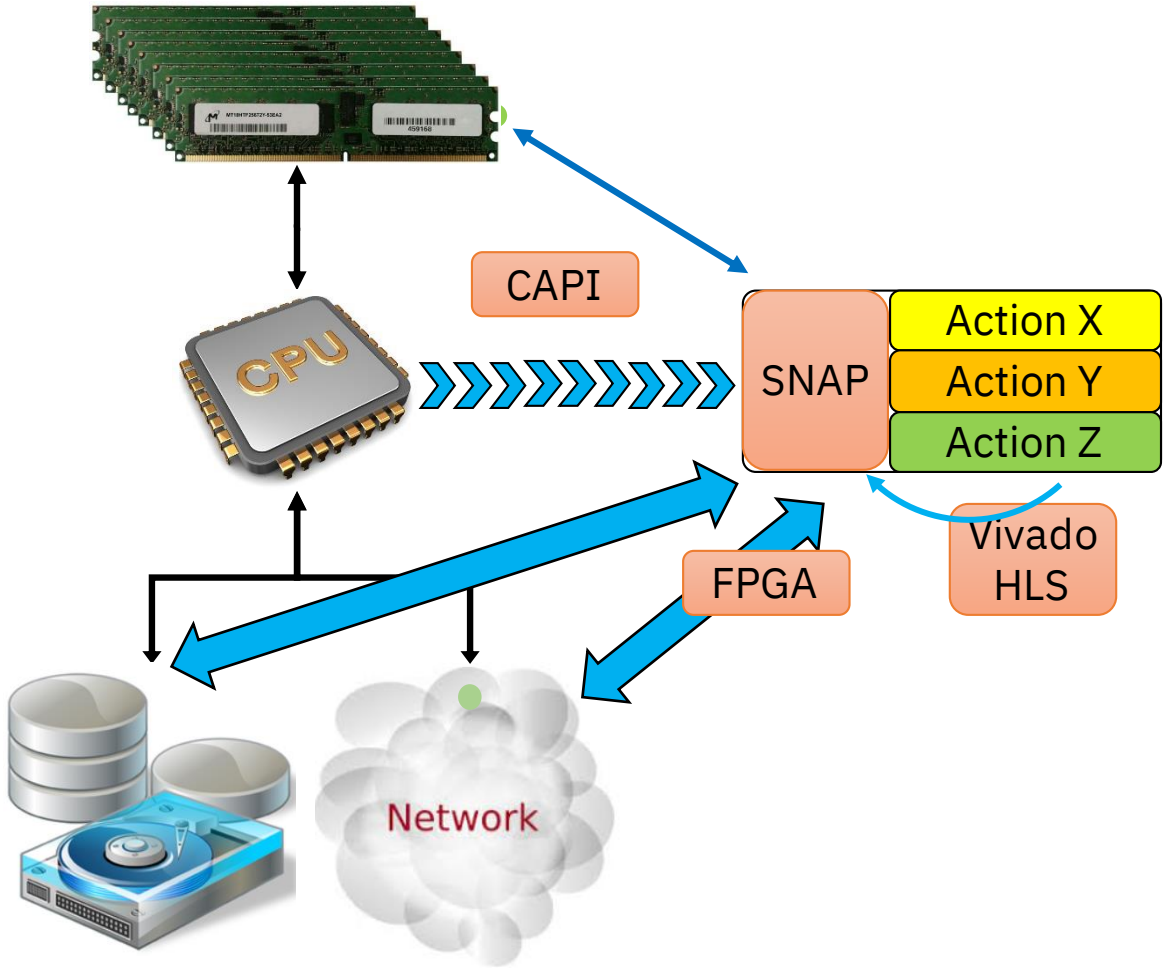
…you could easily program your FPGA using C/C++?
…and get 10x performance* in a few days?
…while operating on data flowing to the server?

	PCI-E FPGA	CAPI FPGA	CAPI SNAP
Target Customer	Computer Engineers	Computer Engineers	Programmers
Development time	3-6 Months	3-6 Months	Days
Software Integration	PCI-E Device Driver	LibCXL	Simple API
Source Code	VHDL, Verilog, OpenCL	VHDL, Verilog, OpenCL	C/C++, Go
Coherency, Security	None	POWER + PSL	POWER + PSL

- Targeted for programmers and computer engineers writing RTL
- SNAP framework manages all of the data flow to enable any user to focus on their core computational algorithm to quickly create accelerated IP.

* Compared to running the same C/C++ in software

The CAPI – SNAP concept



CAPI — FPGA becomes a peer of the CPU
➔ Action **directly** accesses host memory

+

SNAP — Manage server threads and actions
Manage access to IOs (memory, network)
➔ Action **easily** accesses resources

+

FPGA — Gives on-demand compute capabilities
Gives direct IOs access (storage, network)
➔ Action **directly** accesses external resources

+

Vivado HLS — Compile Action written in C/C++ code
Optimize code to get performance
➔ Action code **can be ported efficiently**

=

Best way to **offload/accelerate** a C/ C++ code with :
- Quick porting
- Minimum change in code
- Better performance than CPU

CAPI SNAP Availability

- Today
  - CAPI 1.0 (POWER8)
  - CAPI 2.0 (POWER9)
  - OpenCAPI 3.0 (POWER)

https://github.com/open-power/snap