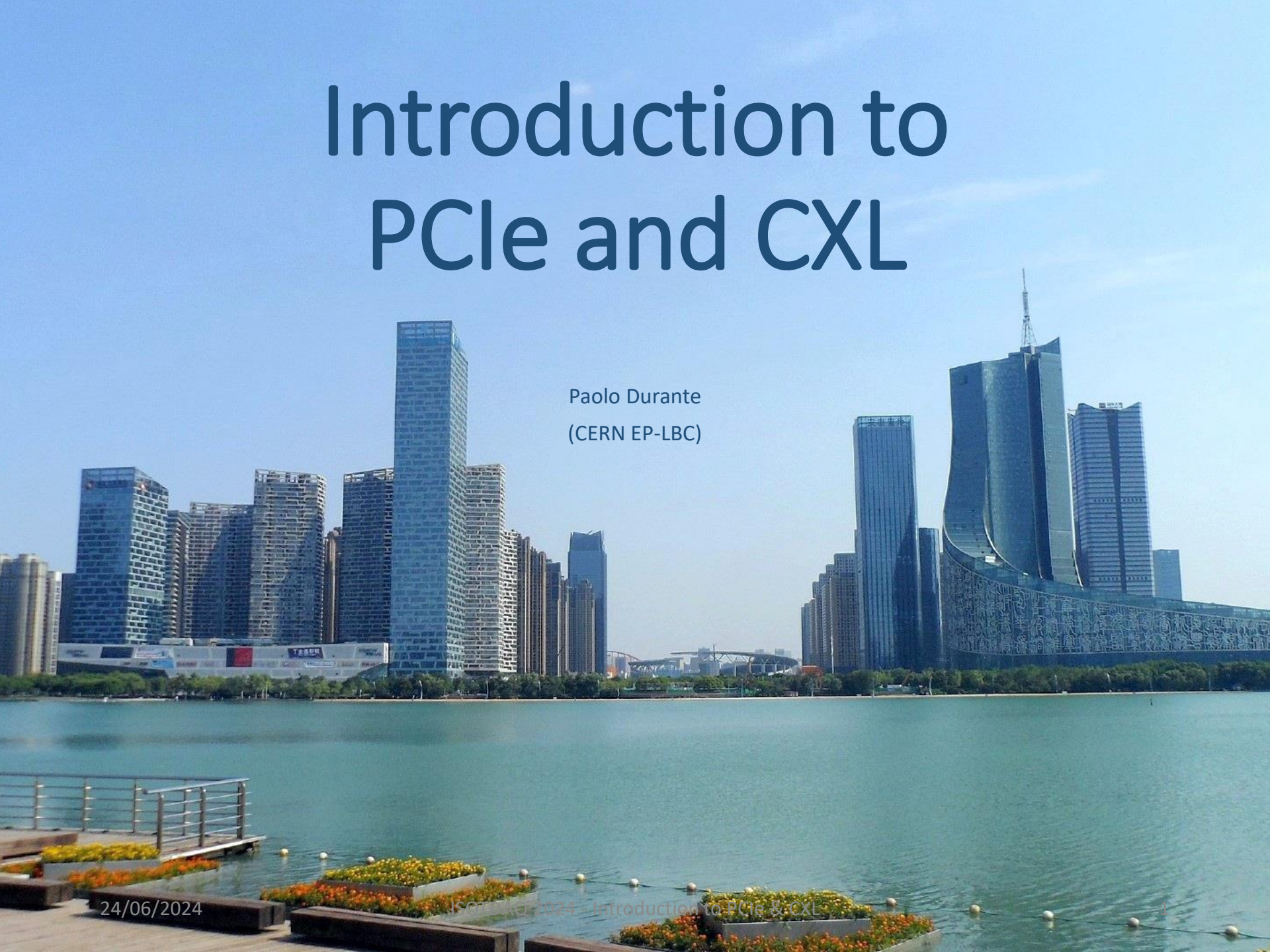
Introduction to PCIe and CXL



Paolo Durante

(CERN EP-LBC)

Where can you find PCI EXPRESS®?

PCI (Peripheral Component Interconnect) Express is a popular standard for high-speed computer expansion overseen by PCI-SIG (Special Interest Group)

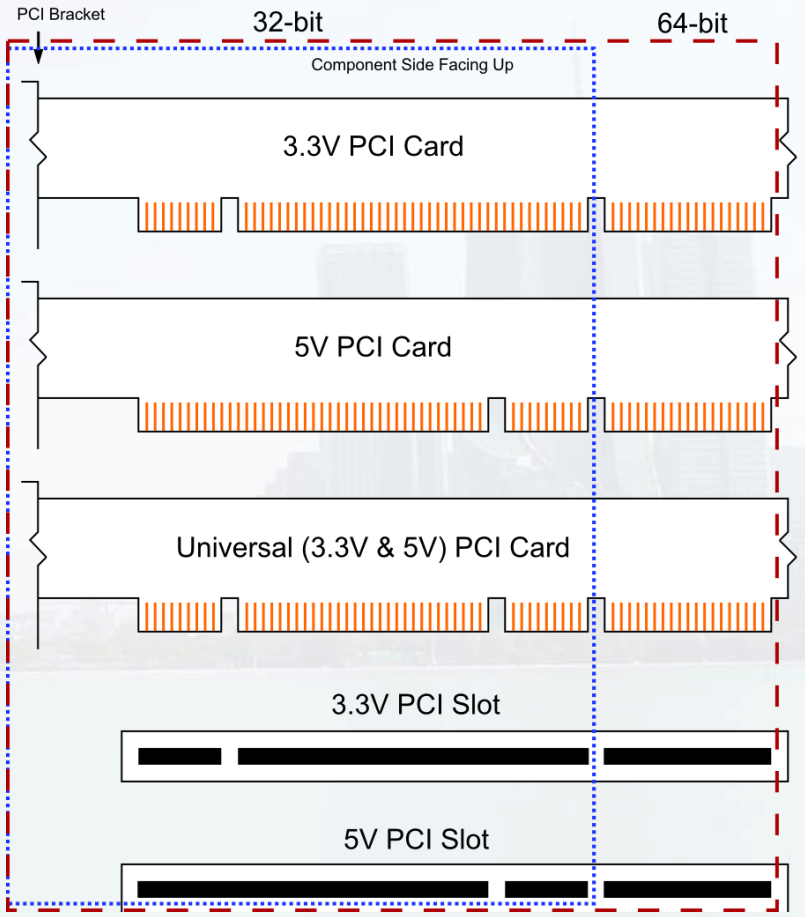
- PCIe interconnects can be present at all levels of your DAQ chain…
  - Readout boards
  - Storage media
  - Network interfaces
  - Compute accelerators (GPUs, FPGAs…)
- …and may be even more so in the future (with CXL)
  - Memory expanders
- Understanding your data acquisition system requires (some) level of understanding of PCI Express

What is this presentation about?

- **PCIe history and evolution**
- PCIe concepts
- PCIe layers
- PCIe practical aspects
- PCIe performance
- PCIe future roadmap

PCI (“conventional PCI”)

- 1992
- ***P**eripheral **C**omponent **I**nterconnect*
- Parallel Interface
- Bandwidth
  - 133 MB/s (~1.0 Gb/s) (32-bit@33 MHz)
  - 533 MB/s (~4.2 Gb/s) (64-bit@66 MHz)
- Plug-and-Play configuration (BARs)

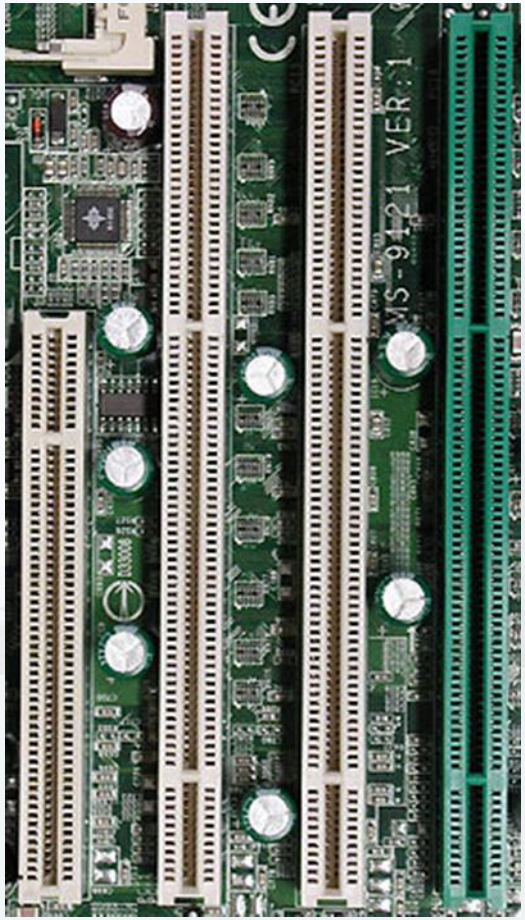


PCI example: ATLAS FILAR

- ~2003
- 4 optical channels
  - 160 MB/s (1.28 Gb/s)
- S-LINK protocol
  - 2 Altera FPGAs
- Burst-DMA over PCI
  - 3rd Altera FPGA
- 64-bit@66MHz PCI

PCI-X (“Extended PCI”)

- 1998
- PCI compatible
  - Hardware and software
  - Half-duplex bidirectional
- Higher bus efficiency
  - Split-responses
  - Message Signaled Interrupts
- Bandwidth
  - ≤ 1066 MB/s (~8.5 Gb/s) (64-bit@133 MHz)
  - 2133 MB/s (~17 Gb/s) (*PCI-X 266*)
  - 4266 MB/s (~34 Gb/s) (*PCI-X 533*)



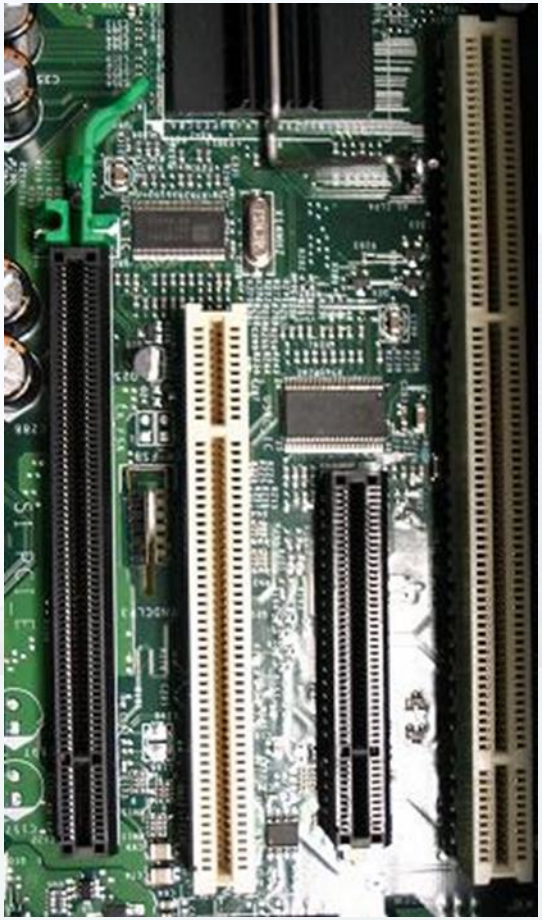
PCI-X example: CMS FEROL

- ~2011
- 4 SFP+ cages
  - 1x 10 Gb/s Ethernet
  - 3x SlinkXpress
- PCI-X interface to legacy FE (Slink64)
- Altera FPGA
- Simplex TCP-IP

PCI Express (PCIe)

- 2004
- PCI “inspired”
  - software, topology
- Serial interface
- Full-duplex bidirectional
- Bandwidth (Gen4)
  - x1: ≤2 GB/s (16 Gb/s) (in each direction)
  - x16: ≤32 GB/s (256 Gb/s) (in each direction)
- Still evolving
  - 1.0, 2.0, 3.0, 4.0, 5.0, 6.0…

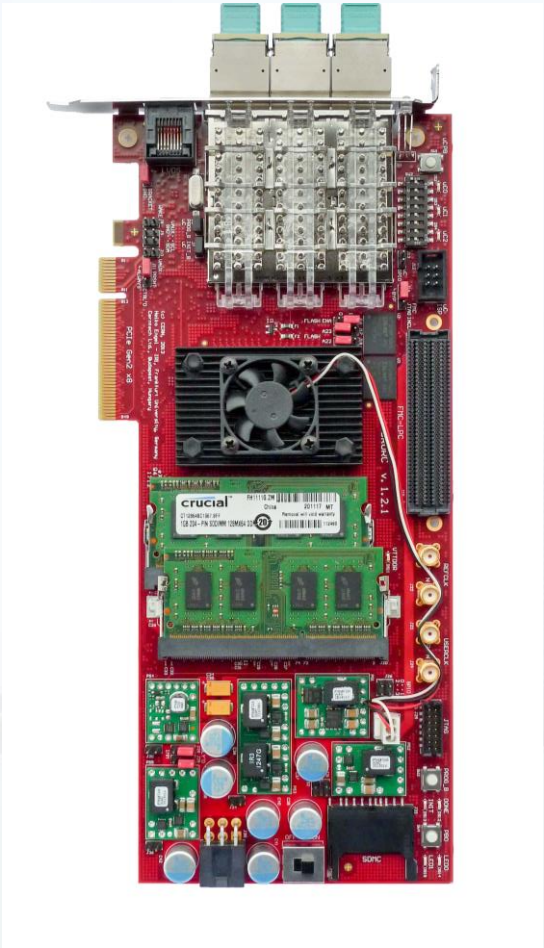
PCIe x16 | PCI | PCIe x8|PCI-X



PCIe example: ALICE C-RORC

- ~2014
- 3x QSFP
  - 36 channels
  - up to 6.6Gb/s/channel
- 2x DDR SO-DIMM
- XilinX Virtex-4 FPGA
- PCIe Gen2 x8

- Also used by ATLAS



PCIe example: LHCb TELL40

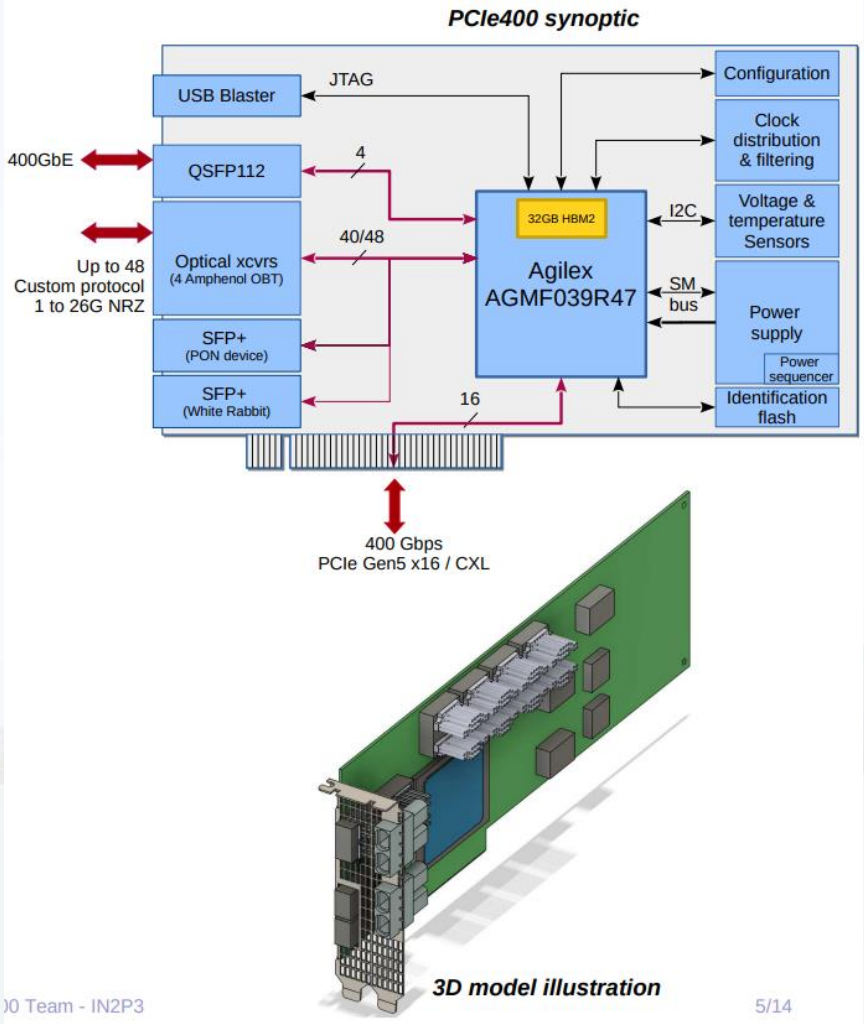
- Introduced for LHC Run3
- Currently in production
- ≤ 48 duplex optical links
  - GBT (3.2 Gb/s)
  - WideBus (4.48 Gb/s)
  - GWT (5.12 Gb/s)
- Altera Arria10 FPGA
- 110 Gb/s DMA
- PCIe 3.0 x16
- Also used by ALICE

PCIe example: ATLAS FELIX

- Introduced for LHC Run3
- ≤ 48 duplex optical links
- XilinX Ultrascale FPGA
- 2x DDR4 SO-DIMM
- PCIe 3.0 x16
- Wupper DMA (Open Source)
- Also used by DUNE

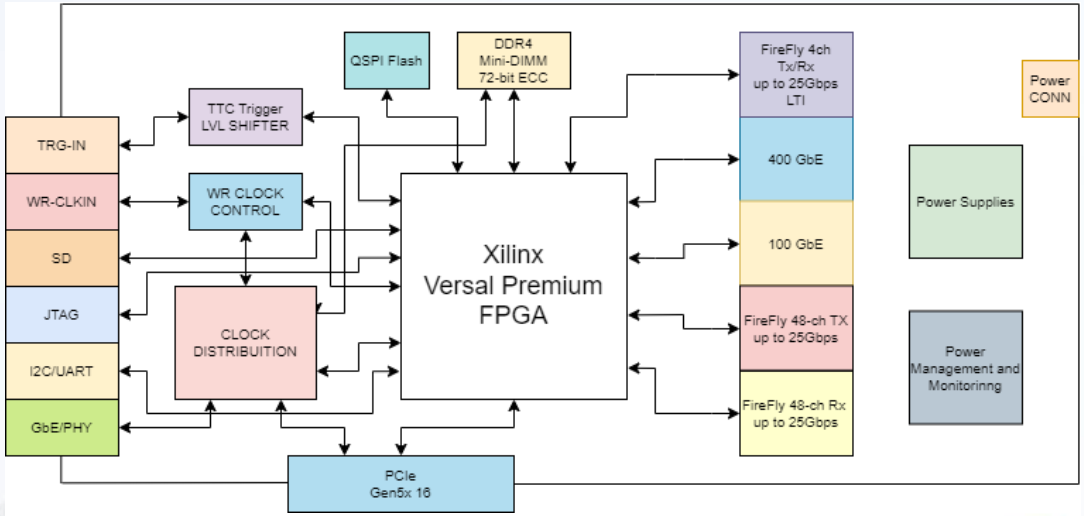
PCIe example: CPPM PCIe400

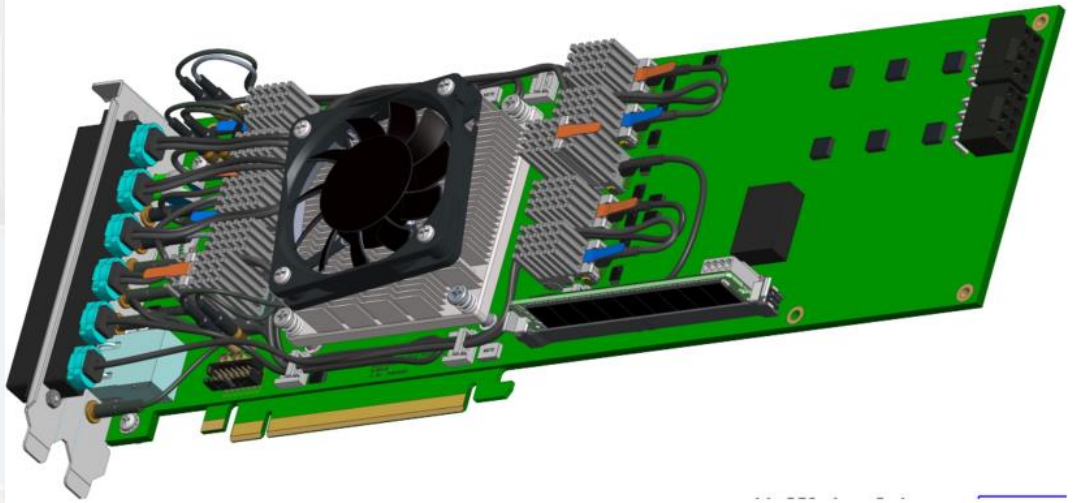
- PCIe Add in Card 3/4 length
- Agilex 7 M-series AGMF039R47A1E2V
  - Processing capabilities x8 - 12 compared to previous generation FPGA (Arria 10)
- 32 GiB HBM2e
- Up to 48x26Gbps NRZ for FE
- PCIe Gen 5 / CXL
- QSFP112 for 400GbE (experimental)
- 2 SFP+ for White Rabbit clock distribution or PON fast control
- High precision PLLs jitter <100fs RMS with phase control



PCIe example: BNL FLX-155

- FPGA: Xilinx Versal Premium XCVP1552
- PCIe Gen5 x16, 512 GT/s
- 48 FireFly data links @25 Gb/s
- LTI link
- 100/400 GbE
- DDR4
- GbE
- White Rabbit
- PetaLinux



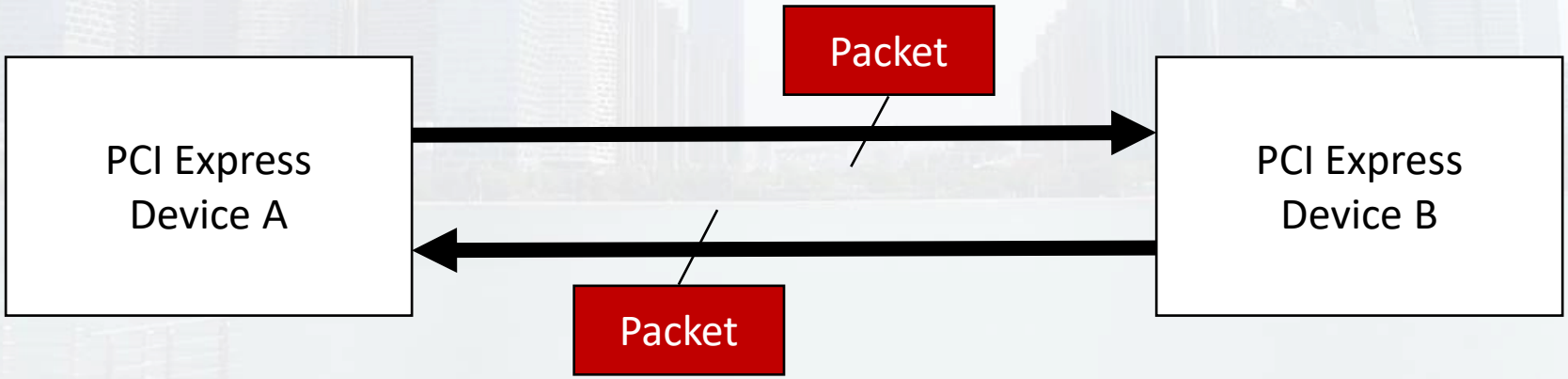


What is this presentation about?

- PCIe history and evolution
- **PCIe concepts**
- PCIe layers
- PCIe practical aspects
- PCIe performance
- PCIe future roadmap

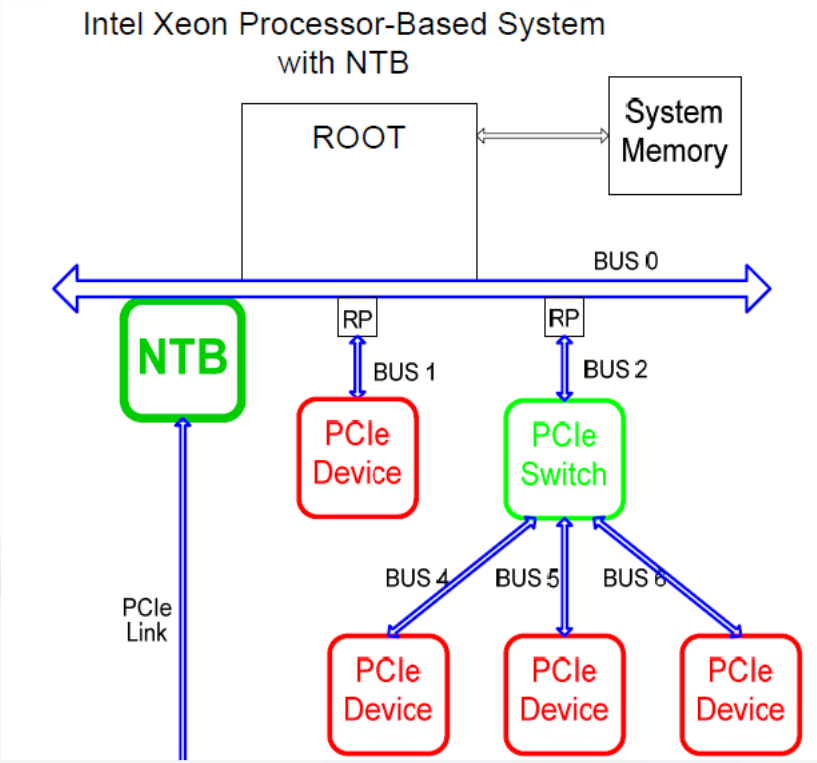
PCIe concepts – Packets

- Point-to-point connection
- “Serial” “bus” (fewer pins)
- Scalable link: **x1**, x2, **x4**, **x8**, x12, **x16**, x32
- Packet encapsulation



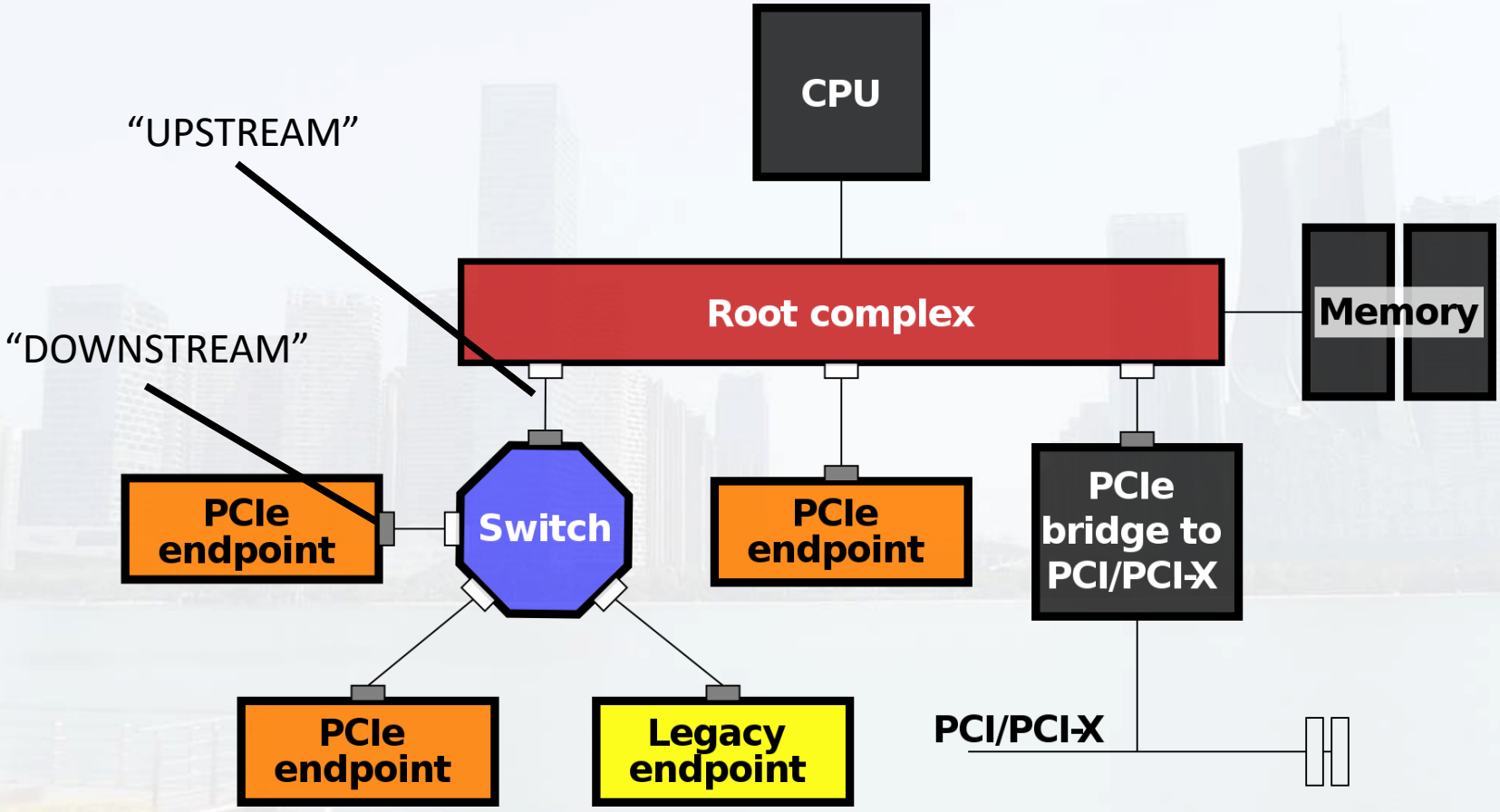
PCIe concepts – Root complex

- Connects the processor and memory subsystems to the PCIe fabric via a *Root Port*
- Generates and processes transactions with *Endpoints* on behalf of the processor



PCIe concepts – Topology

Relative to root – up is towards, down is away



PCIe concepts – BDF

"geographical addressing"

- **B**us : **D**evice . **F**unction
  - Form a hierarchy-based address
  - Multiple logical "Functions" allowed on one physical device
  - Bridges (PCI/PCI-X) form hierarchy
  - Switches (PCIe) form hierarchy

On linux: $ man lspci

```
$ lspci -tv
-+-[0000:ff]-+-08.0  Intel Corporation Xeon ...
 |           +-08.3  Intel Corporation Xeon ...
 |           +-08.4  Intel Corporation Xeon ...
 |           +-09.0  Intel Corporation Xeon ...
 |           ...
 +-[0000:80]-+-00.0-[81]--
 |           +-01.0-[82]--
 |           +-02.0-[83]----00.0  Intel Corporation Xeon Phi coprocessor 31S1
 |           +-03.0-[84]--
 |           +-03.2-[85]----00.0  Intel Corporation Xeon Phi coprocessor 31S1
 |           +-05.0  Intel Corporation Xeon E5/Core i7 Address Map, VTd_Misc, System Management
 |           +-05.2  Intel Corporation Xeon E5/Core i7 Control Status and Global Errors
 |           \-05.4  Intel Corporation Xeon E5/Core i7 I/O APIC
 +-[0000:7f]-+-08.0  Intel Corporation Xeon E5/Core i7 QPI Link 0
 |           +-08.3  Intel Corporation Xeon E5/Core i7 QPI Link Reut 0
 |           ...
 \-[0000:00]-+-00.0  Intel Corporation Xeon E5/Core i7 DMI2
             +-01.0-[01]--
             +-01.1-[02]--
             +-02.0-[03]----00.0  Intel Corporation Xeon Phi coprocessor 31S1
             +-03.0-[04]----00.0  Intel Corporation Xeon Phi coprocessor 31S1
             +-05.0  Intel Corporation Xeon E5/Core i7 Address Map, VTd_Misc, System Management
             +-05.2  Intel Corporation Xeon E5/Core i7 Control Status and Global Errors
             +-05.4  Intel Corporation Xeon E5/Core i7 I/O APIC
             +-11.0-[05]--+-00.0  Intel Corporation C602 chipset 4-Port SATA Storage Control Unit
             |            \-00.3  Intel Corporation C600/X79 series chipset SMBus Controller 0
             +-1c.0-[06]----00.0  Intel Corporation 82574L Gigabit Network Connection
             ...

00:00.0 Host bridge: Intel Corporation Xeon E5/Core i7 DMI2 (rev 07)
80:02.0 PCI bridge: Intel Corporation Xeon E5/Core i7 IIO PCI Express Root Port 2a (rev 07)
83:00.0 Co-processor: Intel Corporation Xeon Phi coprocessor 31S1 (rev 11)
```

Troubleshooting with lspci

- Device works but is “slow”
  - Link speed
  - Link width
  - MaxPayloadSize
  - Interrupts
  - Error flags
  - Look for bottlenecks upstream
- Device is “there” but driver fails to load
  - Unreadable config space
  - Unallocated BARs

Practical troubleshooting (1/3)

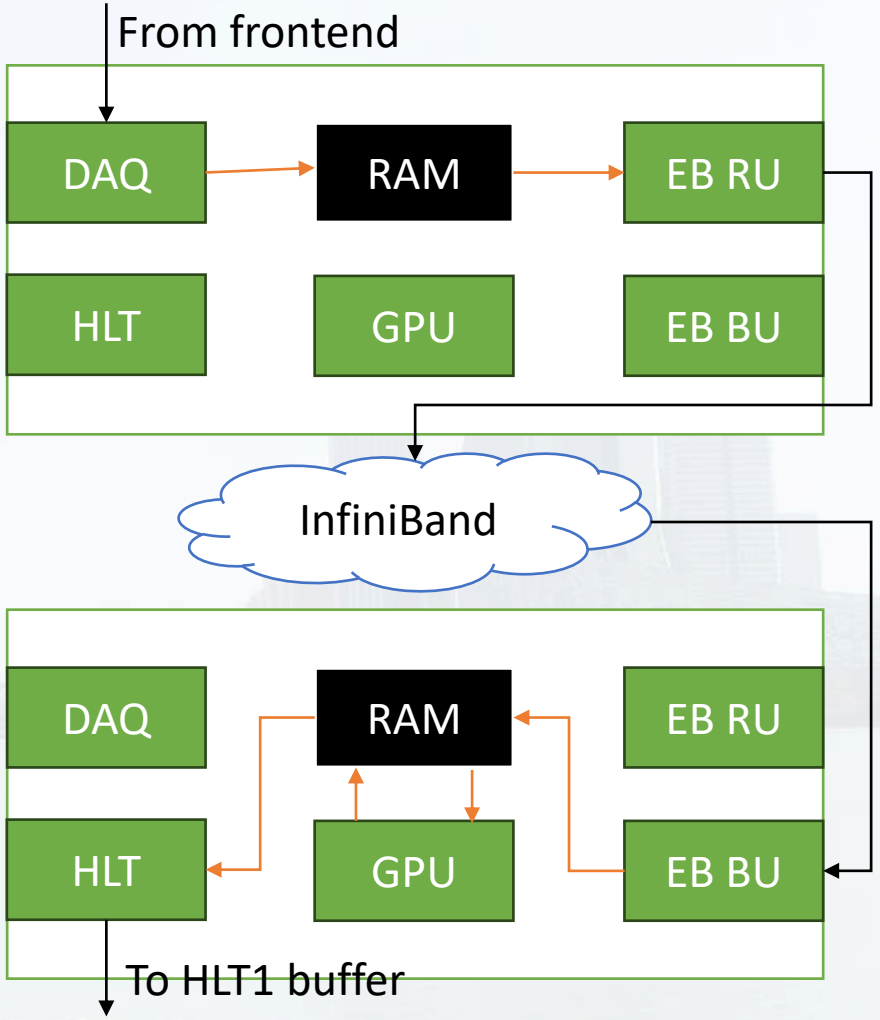
The LHCb experiment is composed of 6 subdetectors

Dataflow (in ~150 readout nodes):

- Readout board → memory (PCIe)
- Local memory → remote memory (PCIe + IB RDMA)
- Memory → GPU (PCIe)
- Memory → HLT1 buffer (PCIe + Ethernet)

The issue

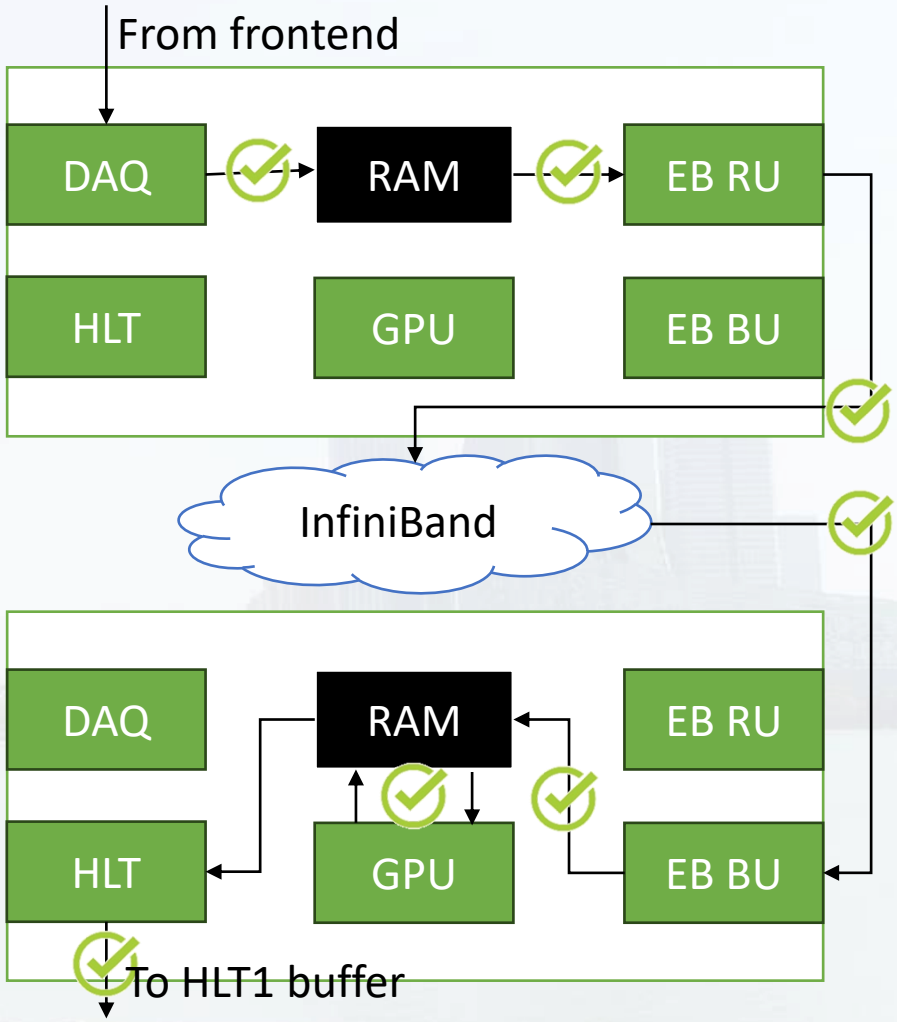
DAQ performs well with 5 subdetectors, throughput drops as soon as a sixth subdetector is added



Practical troubleshooting (2/3)

Check all connections

- DAQ → RAM (OK)
  - No backpressure
- EB RU → EB BU (OK)
  - Full ibwritebw throughput, no congestion
- RAM ↔ GPU (OK)
  - Nominal HLT rate, no backpressure
- HLT send → HLT recv (OK)
  - Full iperf throughput on each port of dual port NIC

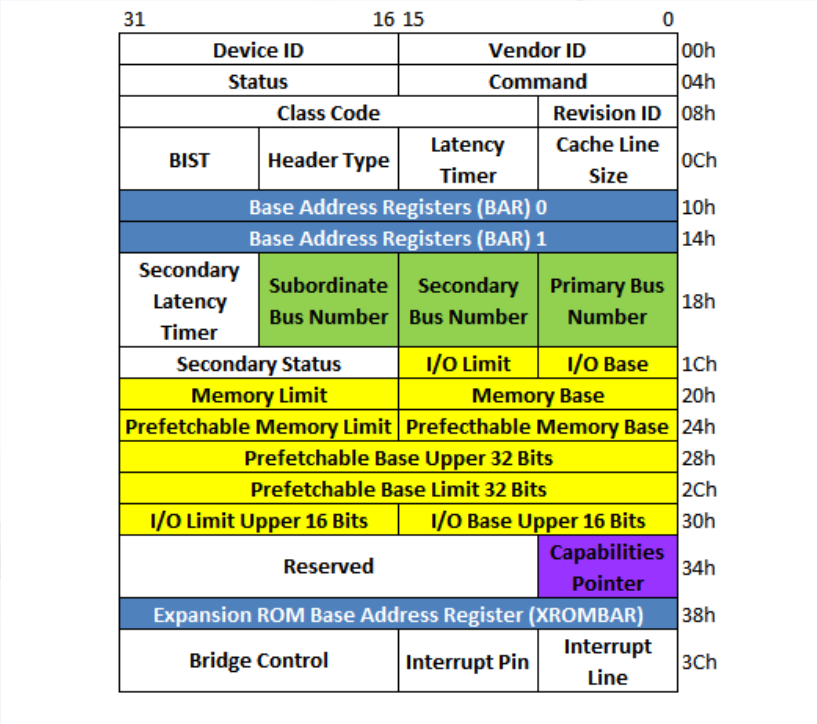


Practical troubleshooting (3/3)

```
$ sudo lspci | grep Ethernet
62:00.0 Ethernet controller: Intel Corporation I350 Gigabit Network Connection (rev 01)
62:00.1 Ethernet controller: Intel Corporation I350 Gigabit Network Connection (rev 01)
e1:00.0 Ethernet controller: Intel Corporation Ethernet Controller X550 (rev 01)
e1:00.1 Ethernet controller: Intel Corporation Ethernet Controller X550 (rev 01)

$ ifconfig
bond0: flags=5187<UP,BROADCAST,RUNNING,MASTER,MULTICAST> mtu 9000
inet 10.132.2.108 netmask 255.255.255.128 broadcast 10.132.2.127
ether b4:2e:99:ac:a7:94 txqueuelen 1000 (Ethernet)
RX packets 165258503 bytes 11423375211 (10.6 GiB)
RX errors 0 dropped 70093 overruns 0 frame 0
TX packets 5062760768 bytes 45520865620164 (41.4 TiB)
TX errors 0 dropped 2 overruns 0 carrier 0 collisions 0

enp225s0f0: flags=6211<UP,BROADCAST,RUNNING,SLAVE,MULTICAST> mtu 9000
ether b4:2e:99:ac:a7:94 txqueuelen 1000 (Ethernet)
RX packets 84474162 bytes 5875677732 (5.4 GiB)
RX errors 0 dropped 2 overruns 0 frame 0
TX packets 2529040640 bytes 22741702699689 (20.6 TiB)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

enp225s0f1: flags=6211<UP,BROADCAST,RUNNING,SLAVE,MULTICAST> mtu 9000
ether b4:2e:99:ac:a7:94 txqueuelen 1000 (Ethernet)
RX packets 80784342 bytes 5547697851 (5.1 GiB)
RX errors 0 dropped 3 overruns 0 frame 0
TX packets 2533720128 bytes 22779162920475 (20.7 TiB)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

$ sudo lspci -s e1:00 -vvv | grep LnkSta:
LnkSta: Speed 5GT/s (degraded), Width x4 (ok)
```

PCIe concepts – Address spaces

- Address spaces
  - Configuration (Bus/Device/Function)
  - Memory (64-bit)
  - I/O (32-bit)
- Configuration space
  - Base Address Registers (BARs) (32/64-bit)
  - Capabilities (linked list)
  - On linux: $ man setpci



31 — 16	15 — 0	
Device ID	Vendor ID	00h
Status	Command	04h
Class Code	Revision ID	08h
BIST / Header Type / Latency Timer	Cache Line Size	0Ch
Base Address Registers (BAR) 0		10h
Base Address Registers (BAR) 1		14h
Secondary Latency Timer / Subordinate Bus Number / Secondary Bus Number	Primary Bus Number	18h
Secondary Status	I/O Limit / I/O Base	1Ch
Memory Limit	Memory Base	20h
Prefetchable Memory Limit	Prefecthable Memory Base	24h
Prefetchable Base Upper 32 Bits		28h
Prefetchable Base Limit 32 Bits		2Ch
I/O Limit Upper 16 Bits	I/O Base Upper 16 Bits	30h
Reserved	Capabilities Pointer	34h
Expansion ROM Base Address Register (XROMBAR)		38h
Bridge Control / Interrupt Pin	Interrupt Line	3Ch

PCIe concepts – Memory & I/O

- Memory space maps cleanly to CPU semantics
  - 32-bits of address space initially
  - 64-bits introduced via Dual-Address Cycles (DAC)
    - Extra period of address time on PCI/PCI-X
    - 4DWORD header in PCI Express
  - Burstable (= Multiple DWORDs)
- I/O space maps cleanly to CPU semantics
  - 32-bits of address space
  - Non-burstable

PCIe concepts – Bus address

This is actually not specific to PCIe, but a generic reminder:

- Physical address: the address the CPU sends to the memory controller
- Virtual address: an indirect address created by the operating system, translated by the CPU to physical
- Bus address: an address understood by the devices connected to a specific bus
- On Linux, see: *pci_iomap*(), *remap_pfn_range*(), …

PCIe concepts – Bridges

Transparent

- Single root (or SR-IOV)
- Single address space
- Multiple downstreams (switch)
- Downstreams appear in the same topology
- Addresses are passed through unchanged

Non-Transparent

- Joins two independent topologies
- One root on each side
- Each side has its own address space
- Needs translation table
- Fault tolerance, “networking”, HPC

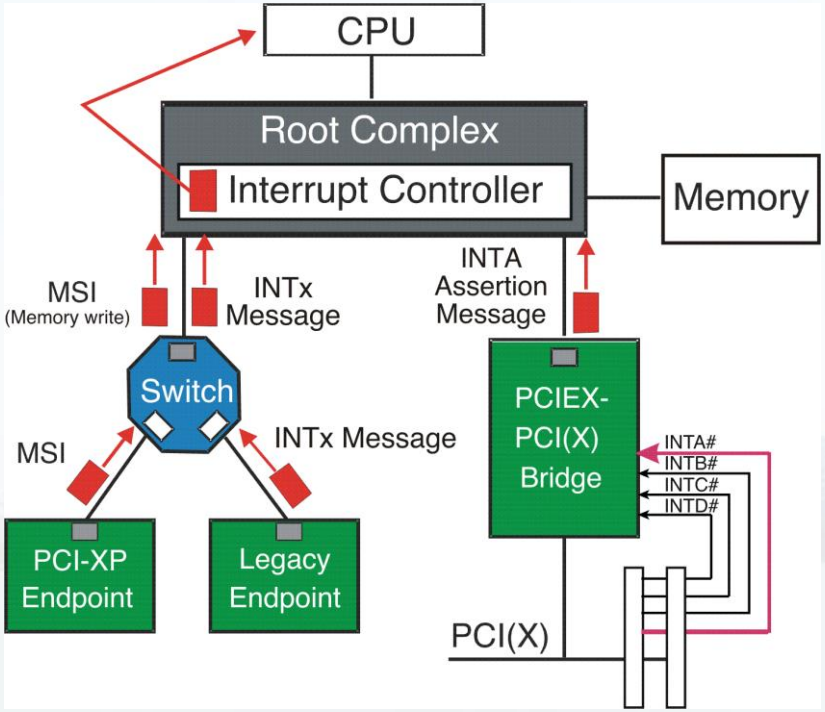
PCIe concepts – Interrupts

- PCI
  - INT*x*#
    - $x \in \{A, B, C, D\}$
  - Level sensitive
  - Can be mapped to CPU interrupt number
- PCIe
  - “Virtual Wire” emulation
  - Assert_INTx code
  - Deassert_INTx code

```
pci_read_config_byte(dev,
    PCI_INTERRUPT_PIN,
    &(...));


pci_read_config_byte(dev,
    PCI_INTERRUPT_LINE,
    &(...));


pci_enable_msi(dev);


request_irq(dev->irq, my_isr,
    IRQF_SHARED, devname,
    cookie);
```

PCIe concepts – MSI & MSI-X

- Based on messages (*MWr*)
- MSI uses one address with a variable data value indicating which “vector” is asserting
  - ≤ 32 per device (in theory)
- MSI-X uses a table of independent address and data pairs for each “vector”
  - ≤ 2048 per device (use affinity!)
- *Vector*: interrupt id



PCIe Gen1 (2003)

- Introduced at 2.5 GT/sec (32 Gb/s/d in x16)
- Signaling rate is also referred to as 2.5 GHz, 2.5 Gb/s
- 100 MHz reference clock
  - Eases synchronization between ends
  - Can use Spread Spectrum Clocking to reduce EMI
    - Optional, but nearly universal
- 8b/10b encoding used to provide DC balance and reduce “runs” of 0s or 1s which make clock recovery difficult
- Specification Revisions: 1.0, 1.0a, 1.1

PCIe Gen2 (2006)

- Speed doubled to 5 GT/sec (64 Gb/s/d in x16)
- Reference clock remains at 100 MHz
  - Lower jitter clock sources required vs 2.5 GT/sec
  - Generally higher quality clock generation/distribution required
- 8b/10b encoding continues to be used
- Specification Revisions: 2.0, 2.1
- Devices choosing to implement a maximum rate of 2.5 GT/sec can still be fully 2.x compliant

PCIe Gen3 (2010)

$$2 \times 5 = ?$$

PCIe Gen3 (2010)

2 x 5 = 8

- Speed “doubled” from 5 GT/sec (126 Gb/s/d in x16)
- More efficient encoding (20% → ~1%)
  - 8b/10b → 128b/130b
- 8 GT/sec electrical rate
  - 10 GT/sec required significant cost and complexity in channel, receiver design, etc.
- Reference clock remains at 100 MHz
- Backwards-compatible speed negotiation

PCIe Gen4 (2017)

$$2 \times 8 = ?$$

PCIe Gen4 (2017)

2 x 8 = 16

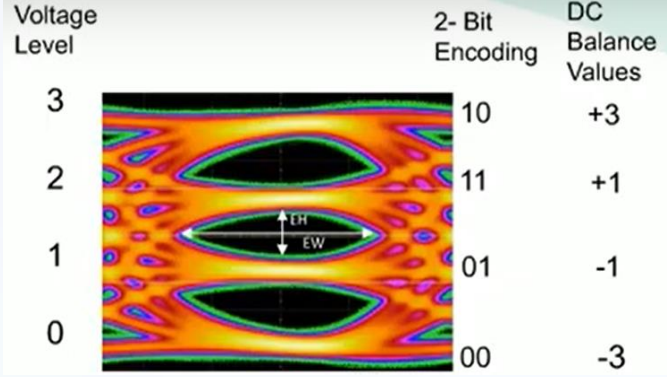
- Speed doubled from 8 GT/sec (252 Gb/s/d in x16)
- Same 128b/130b encoding
- 16 GT/sec electrical rate
  - Channel length: ≤ 10”/14”
  - Retimer mandatory for longer channels
  - More complex pre-amplification, equalization stages
- Reference clock remains at 100 MHz
- Backwards-compatible protocol negotiation and CEM spec

PCIe Gen5 (2019)

2 x 16 = 32

- Speed doubled from 16 GT/sec (504 Gb/s/d in x16)
- Same 128b/130b encoding (with small differences)
- 32 GT/sec electrical rate
  - Channel length: ≤ 10”/14”
  - Up to 2 retimers for longer channels
  - More complex pre-amplification, equalization stages
- Support for alternate protocols (see CXL)

PCIe Gen6 (2022)



2 x 32 = 64

- Speed doubled from 32 GT/sec (1024 Gb/s/d in x16)
- NRZ → PAM4 signaling
  - 2 bits per Unit Interval
  - Lower eye-height and width, much higher First Bit-Error Rate (FBER)
- Forward Error Correction (FEC)
  - Light-weight and low-latency (2ns) FEC for initial correction
  - CRC and link-level retry for larger errors
- Flow Control Unit (FLIT) encoding
  - Fixed-size and fixed(lower)-latency, compared to TLPs

PCIe Gen7 (2023)

PCI Express 7.0: Coming in 2027/2028 External Inbox ×

Dr. Ian Cutress from More Than Moore <morethanmoore@substack.co... 13 Jun 2023, 20:31 (5 days ago)

to xxxxxxxxxxx+morethanmoore

Open in app or online

PCI Express 7.0: Coming in 2027/2028

Initial Specification Now Live

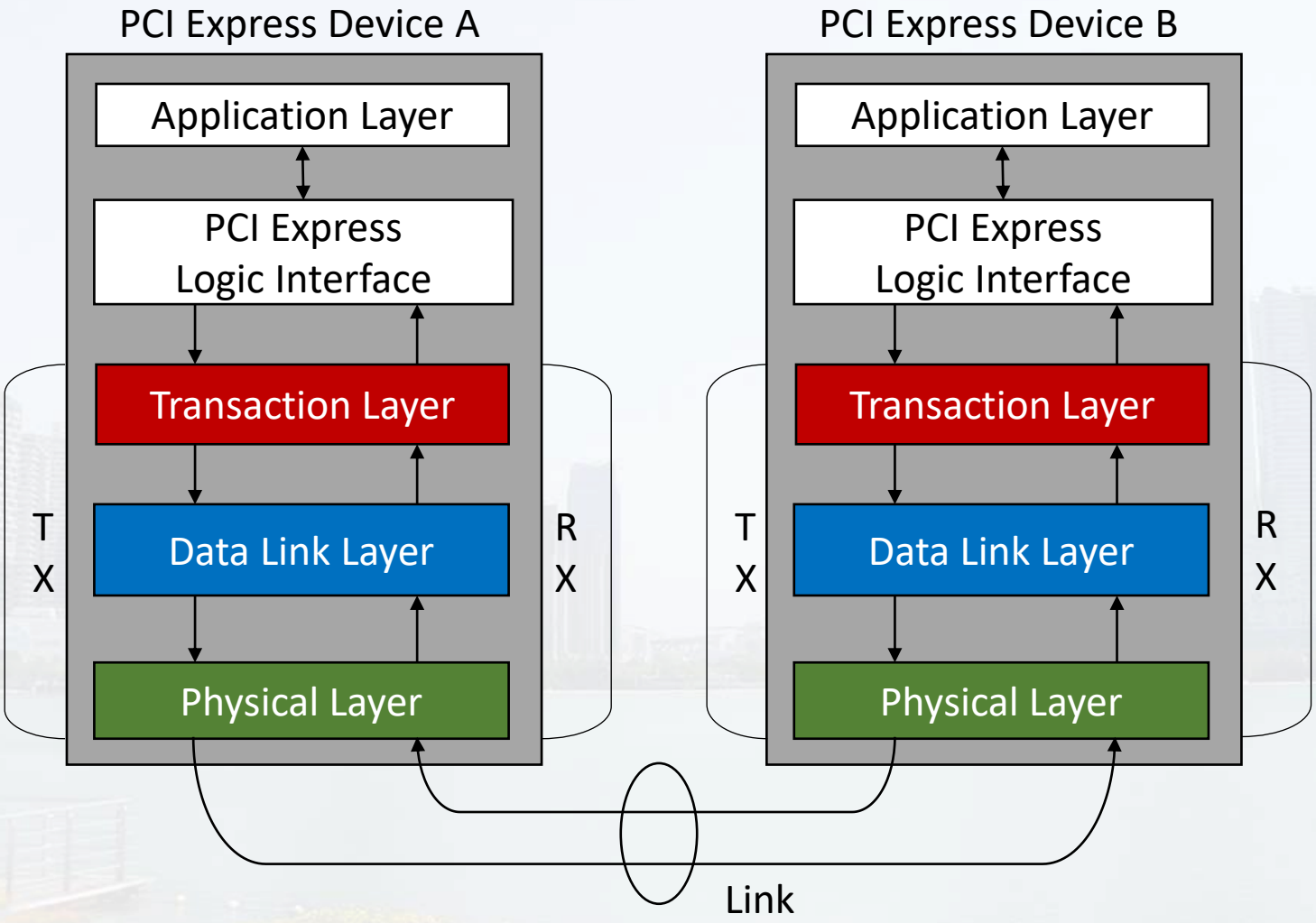
DR. IAN CUTRESS

JUN 13

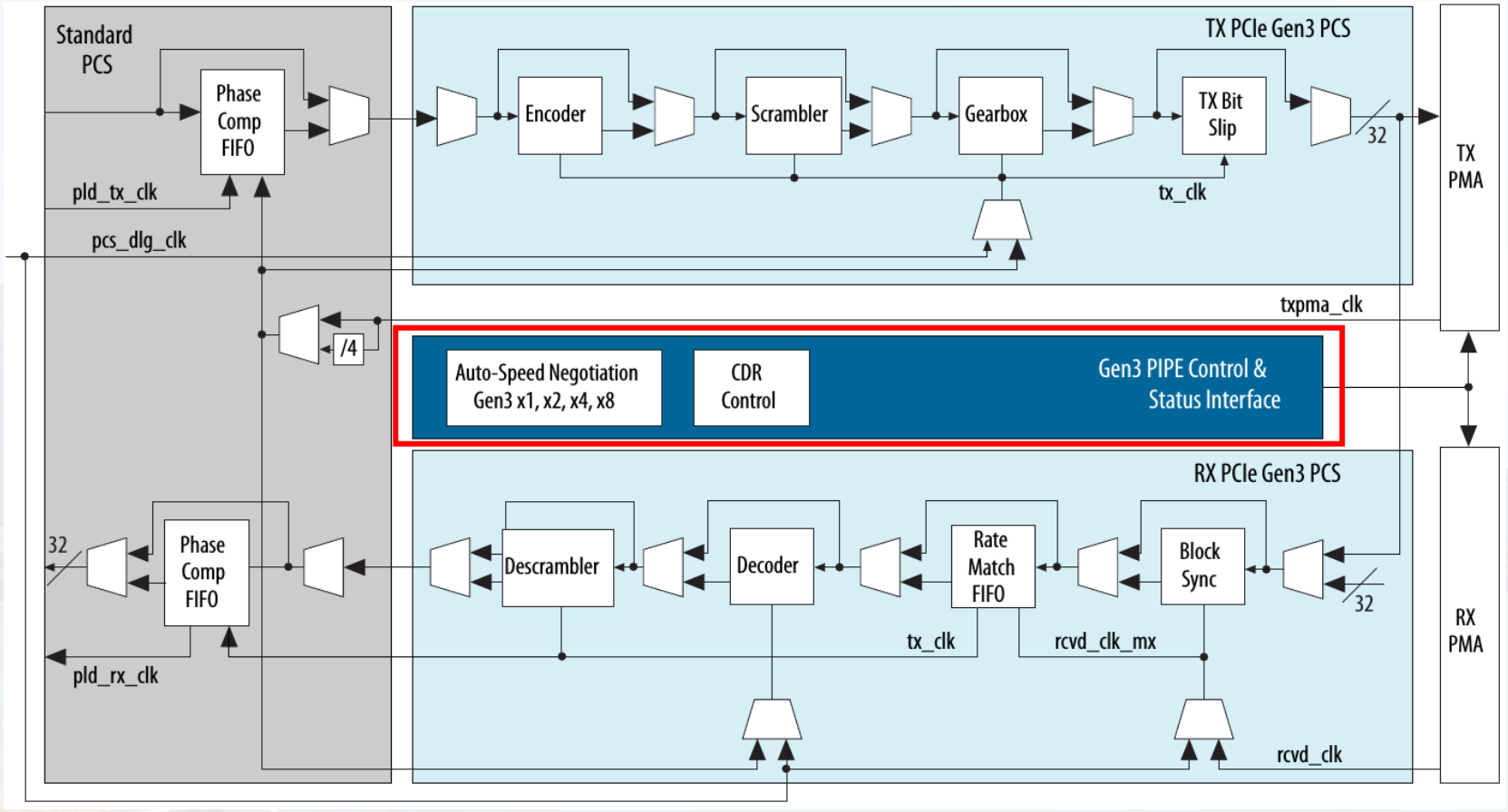
What is this presentation about?

- PCIe history and evolution
- PCIe concepts
- **PCIe layers**
- PCIe practical aspects
- PCIe performance
- PCIe future roadmap

PCIe – Protocol stack



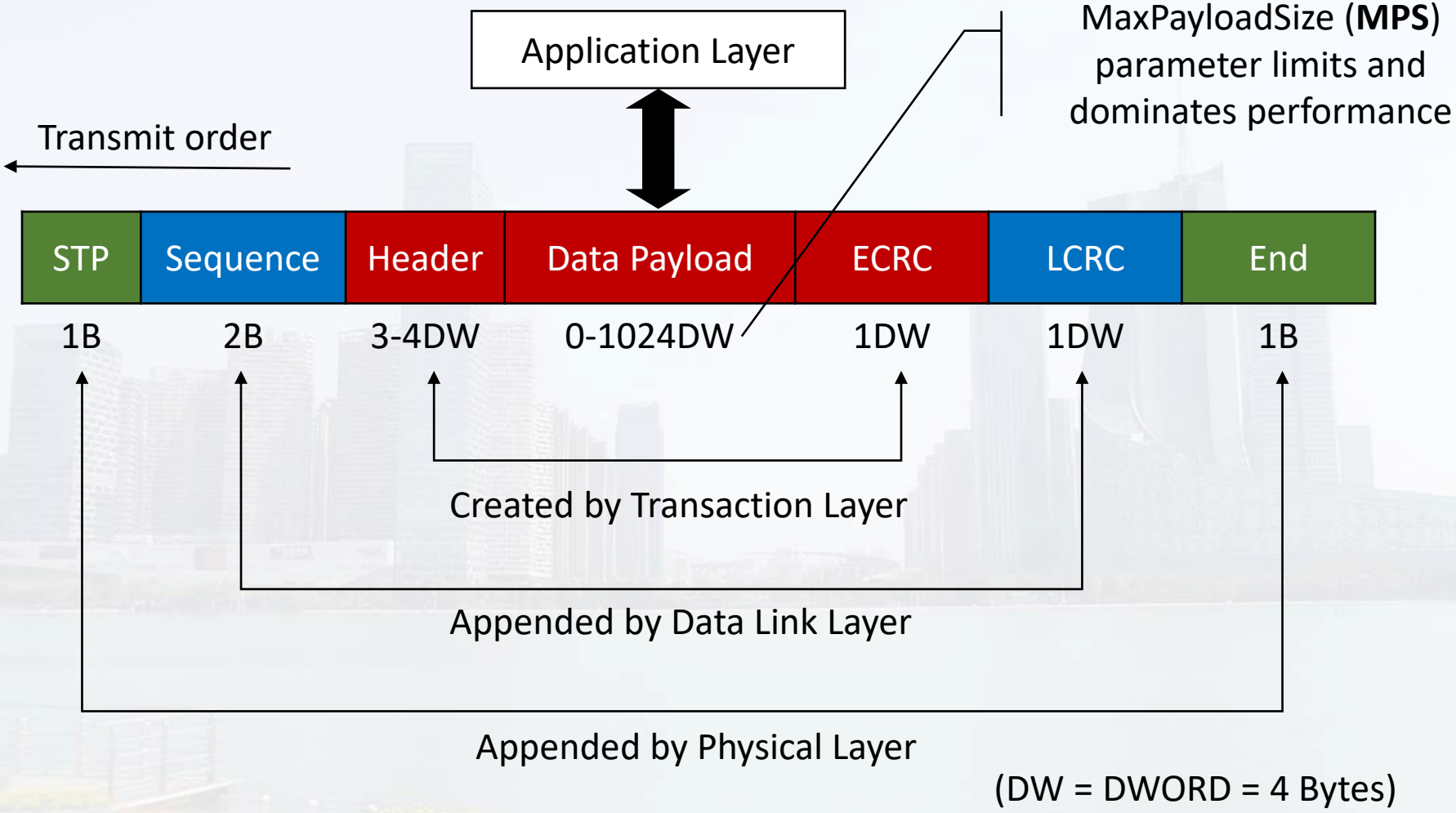
FPGA Hardened PCIe IP



PCIe – Transaction layer

- Four possible transaction types
  - **Memory Read** | **Memory Write**
    - Transfer data from or to a memory mapped location
    - Address routing
  - **IO Read** | **IO Write**
    - Transfer data from or to an IO location (on a legacy endpoint)
    - Address routing
  - **Config Read** | **Config Write**
    - Discover device capabilities, status, parameters
    - ID routing (BDF)
  - **Messages**
    - Event signaling

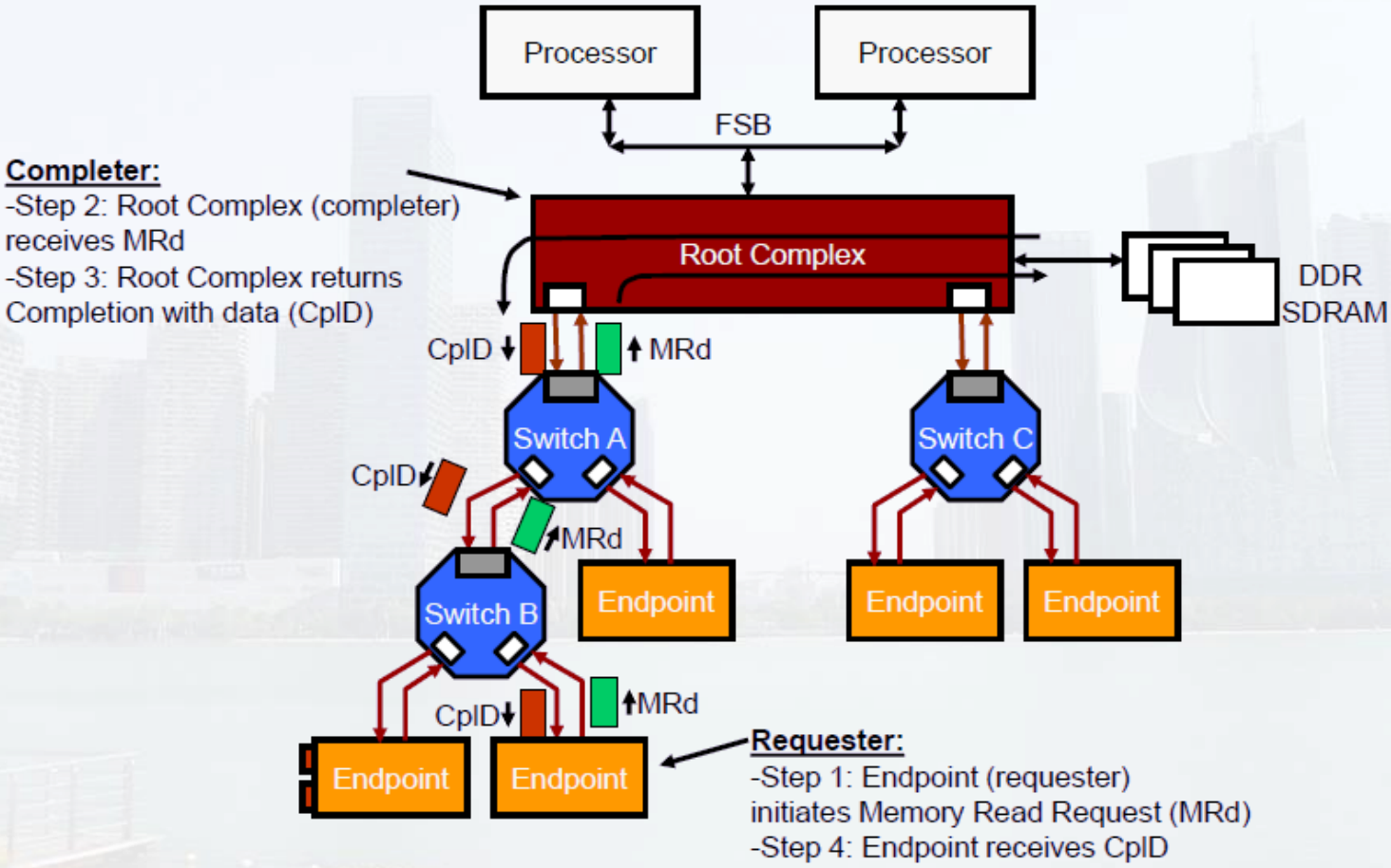
PCIe – TLP structure



PCIe – Split transaction model

- Posted transaction
  - Single TLP, no completion
- Non-posted transaction
  - Split transaction model
    - Requester initiates transaction (Requester ID + Tag)
      - Requester and Completer IDs encode the sender BDF
    - Completer executes transaction internally
    - Completer creates completion transaction (Cpl/CplD)
- Bus efficiency of Read is different (lower) wrt Write
  - Writes are posted while Reads are not

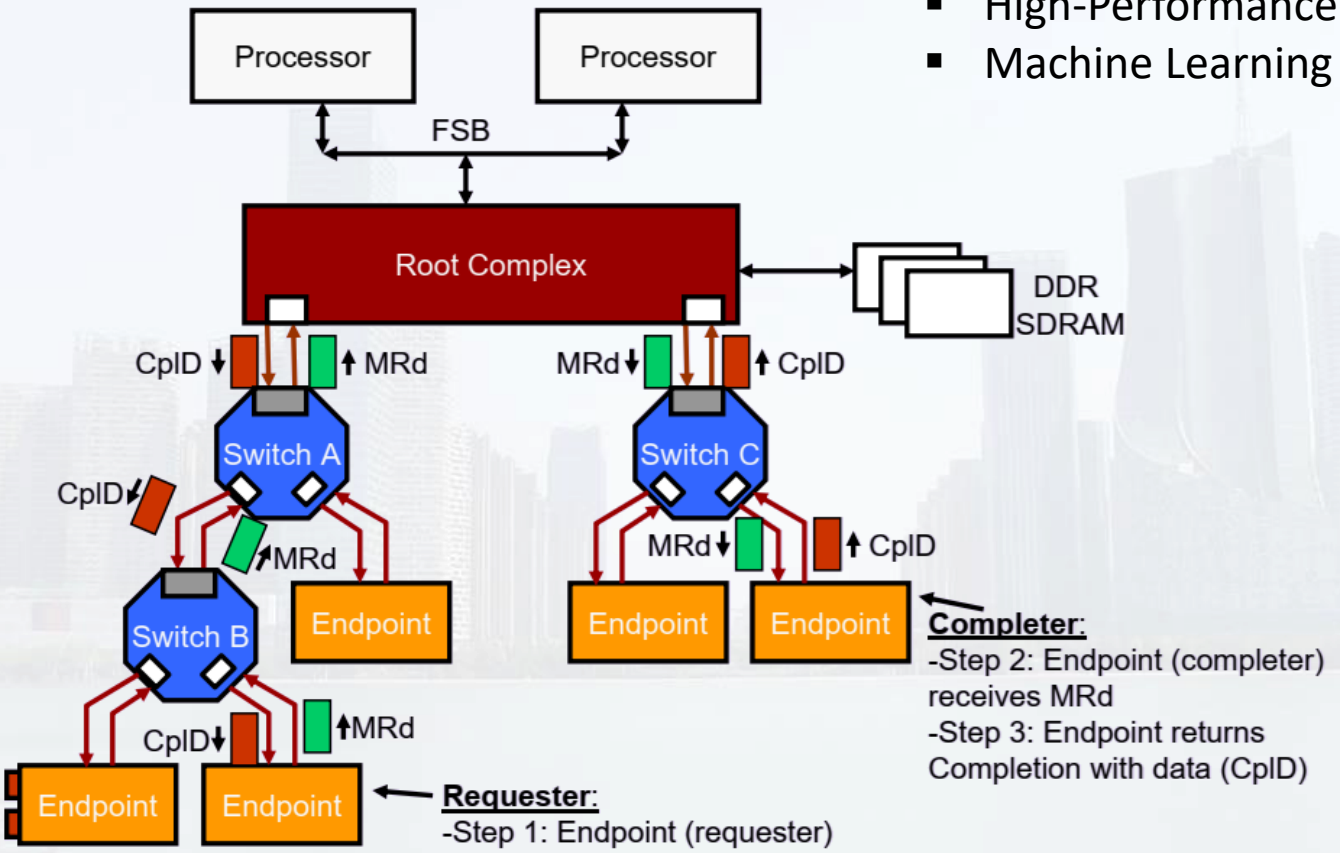
PCIe – DMA transaction



PCIe – Peer-to-Peer transaction

Typical use cases:

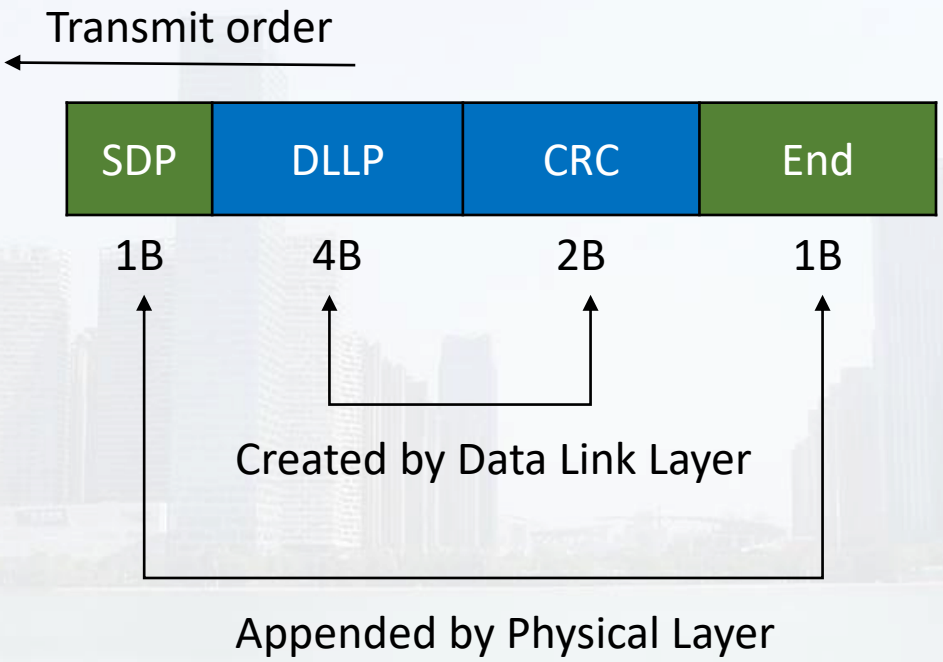
- High-Performance Computing
- Machine Learning



PCIe – Data Link Layer

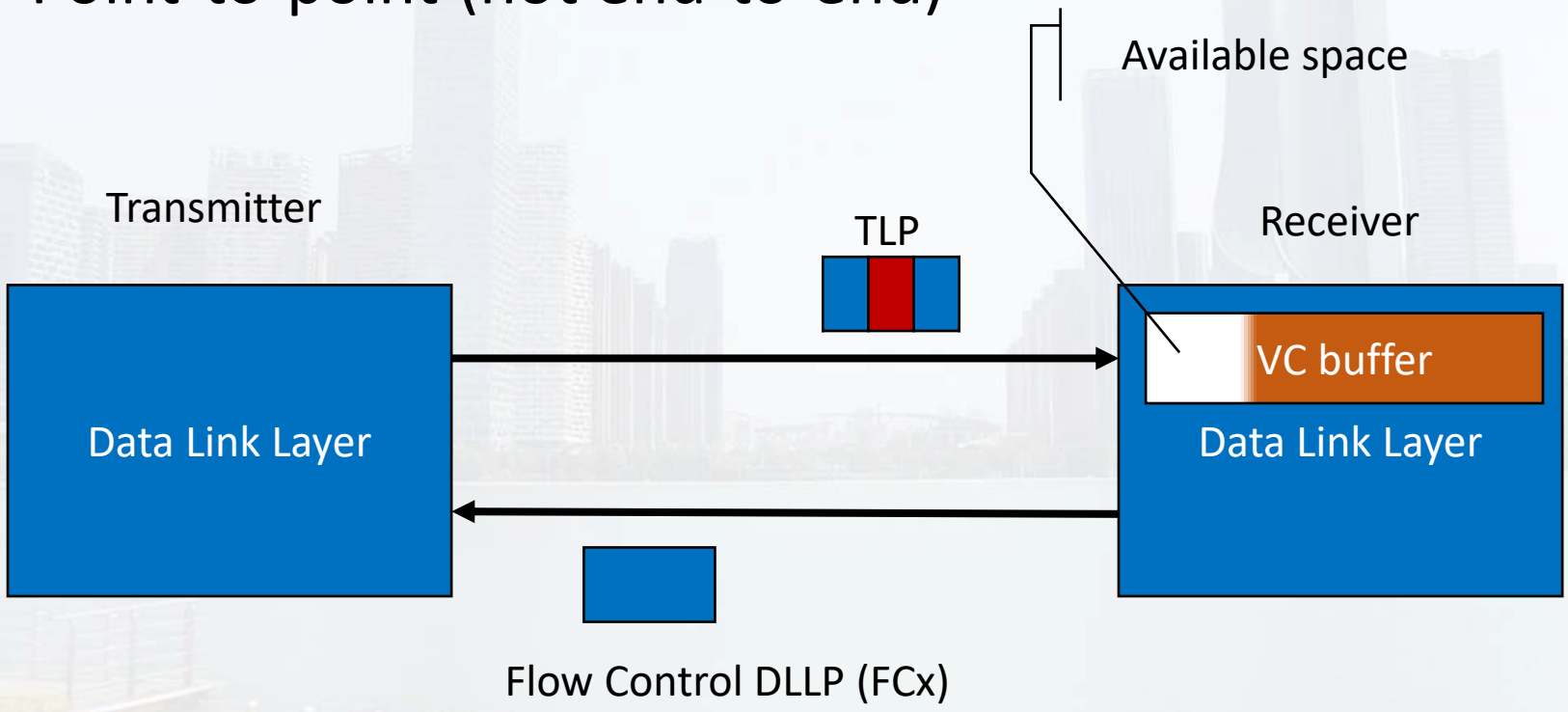
- ACK / NAK Packets
  - Error handling mechanism
- Flow Control Packets (FCPs)
  - Receiver sends FCPs (which are a type of DLLP) to provide the transmitter with **credits** so that it can transmit packets to the receiver
- Power Management Packets
- Vendor extensions
  - E.g.: CAPI, CCIX (memory coherency)

PCIe – DLLP structure



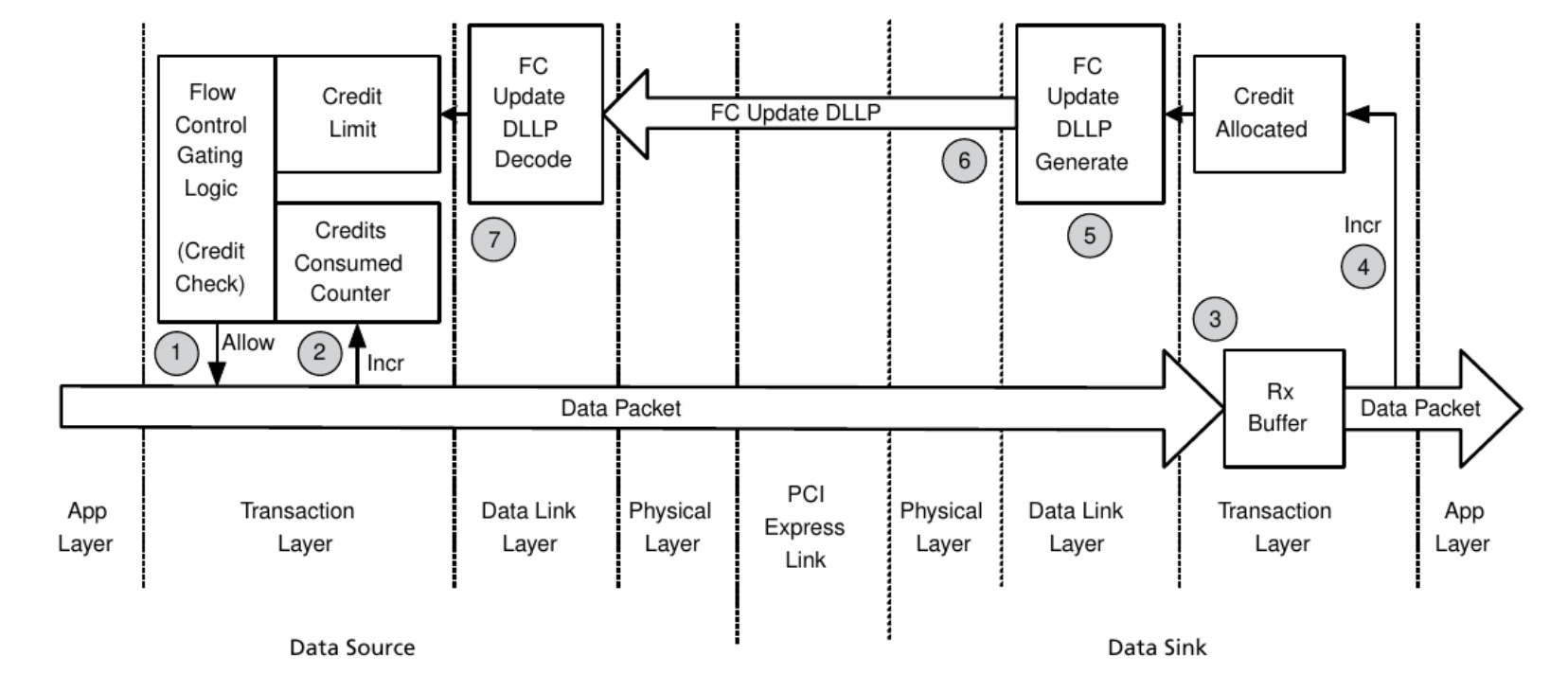
PCIe – Flow control

- Credit-based
- Point-to-point (not end-to-end)



PCIe – Flow Control Update Loop

"If the write requester sources the data as quickly as possible, and the completer consumes the data as quickly as possible, then the Flow Control Update loop may be the biggest determining factor in write throughput, after the actual bandwidth of the link." *(Intel)*



PCIe – RAS/QoS features

- Data Integrity and Error Handling
  - PCIe is RAS (Reliable, Available, Serviceable)
  - Data integrity at
    - link level (LCRC)
    - end-to-end (ECRC, optional)
- Virtual channels (VCs) and traffic classes (TCs) to support differentiated traffic or Quality of Service (QoS)
- In theory
  - Ability to define levels of service for packets of different TCs
  - 8 TCs and 8 VCs available
- In practice
  - Rarely more than 1 VC and 1 TC are implemented

PCIe – Error handling

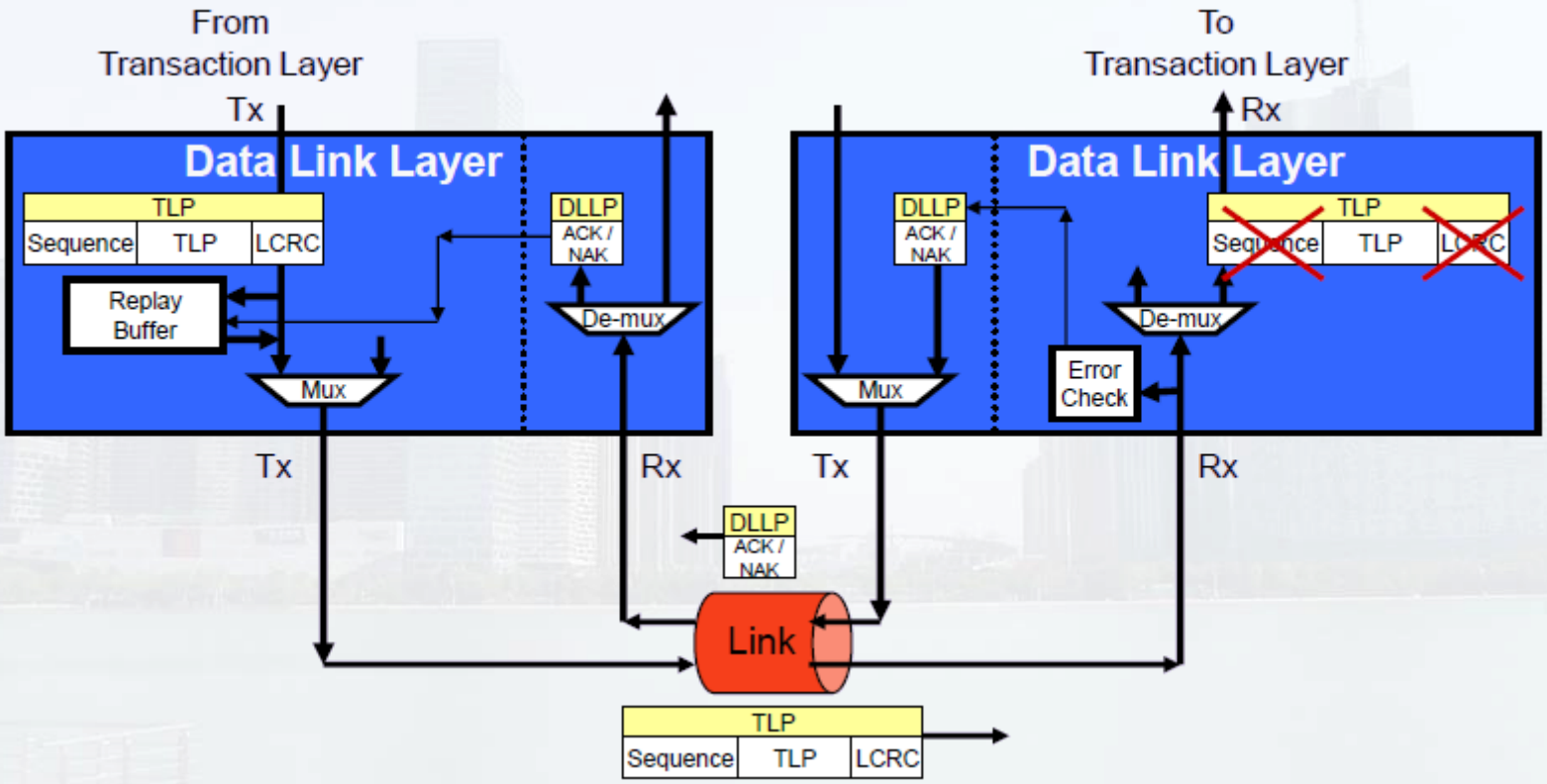
Correctable

- Recovery happens automatically in DLL
- Performance is degraded
- Example: LCRC error → automatic DLL retry
  (there is no forward error correction until PCIe Gen 6.0)

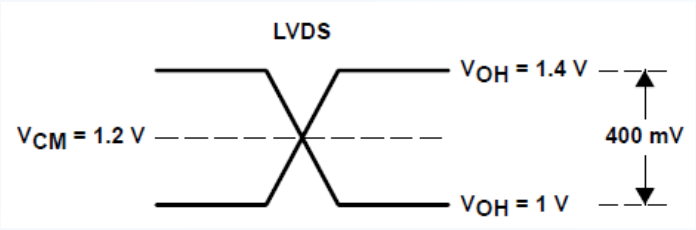
Uncorrectable

- **Fatal**
  - Platform-specific handling
- **Non-fatal**
  - Can be exposed to application layer and handled explicitly
- Can and do cause system deadlock / reset
- Recovery mechanisms are outside the spec
  - Example: failover for HA

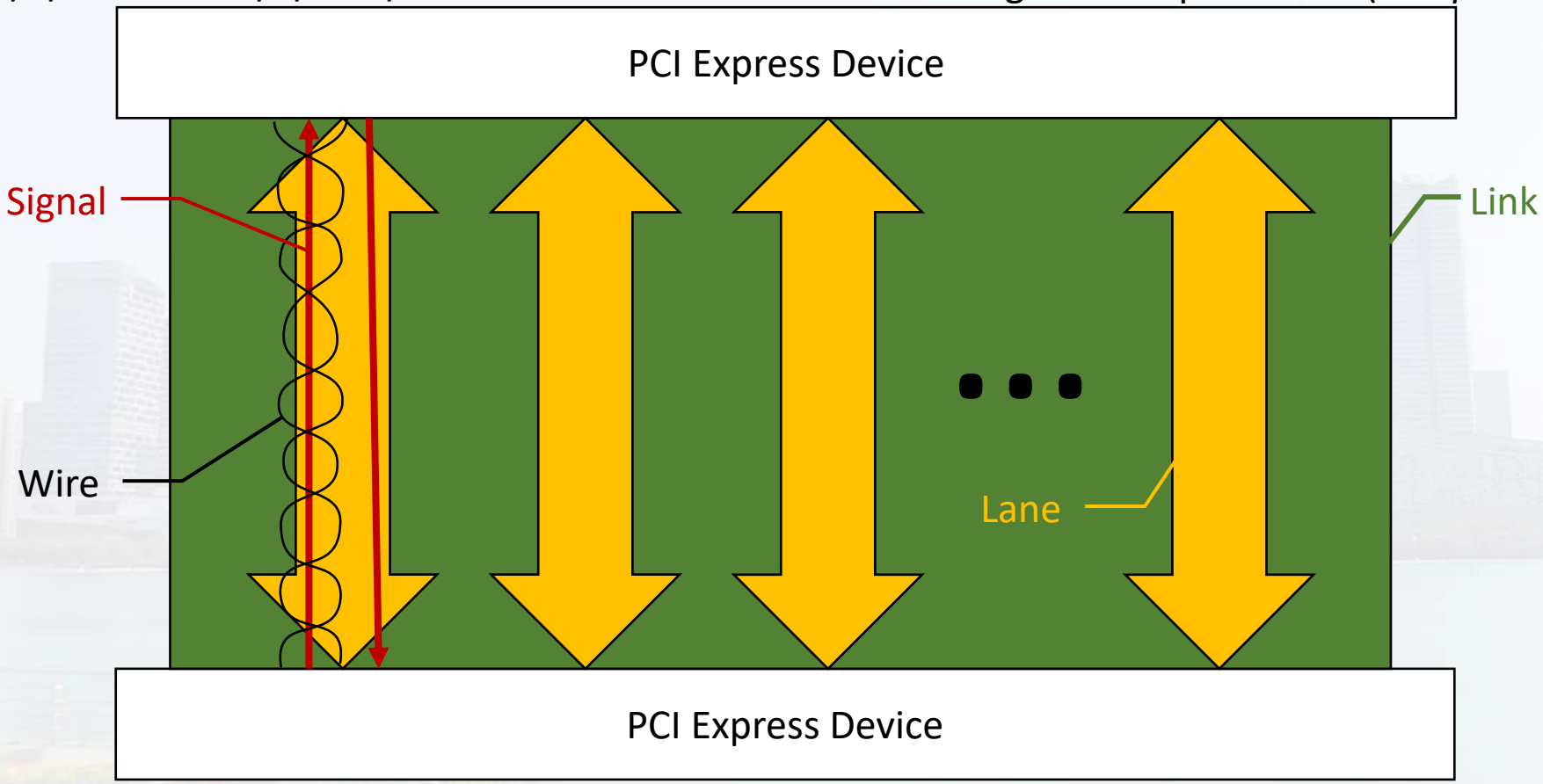
PCIe – ACK/NAK


From
Transaction Layer
Tx
Data Link Layer
TLP
Sequence
TLP
LCRC
Replay
Buffer
Mux
DLLP
ACK /
NAK
De-mux
Tx
Rx
To
Transaction Layer
Rx
Data Link Layer
DLLP
ACK /
NAK
TLP
Sequence
TLP
LCRC
De-mux
Error
Check
Mux
Tx
Rx
DLLP
ACK /
NAK
Link
TLP
Sequence
TLP
LCRC

PCIe – Physical layer



"While the lanes are not tightly synchronized, there is a limit to the lane to lane skew of 20/8/6 ns for 2.5/5/8 GT/s so the hardware buffers can re-align the striped data." (*Wikipedia*)



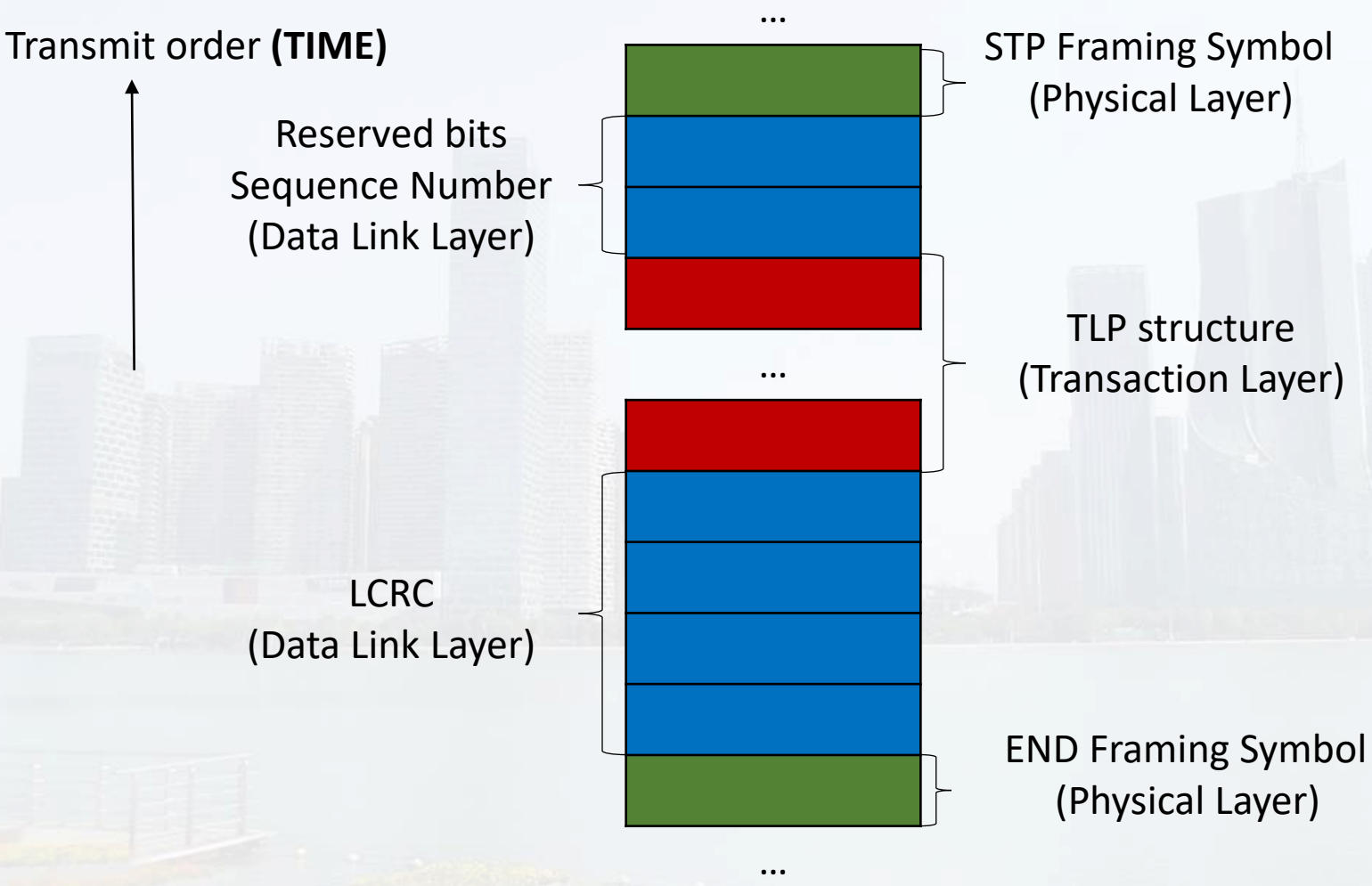
PCIe – Ordered-Set Structure

Six ordered sets are possible

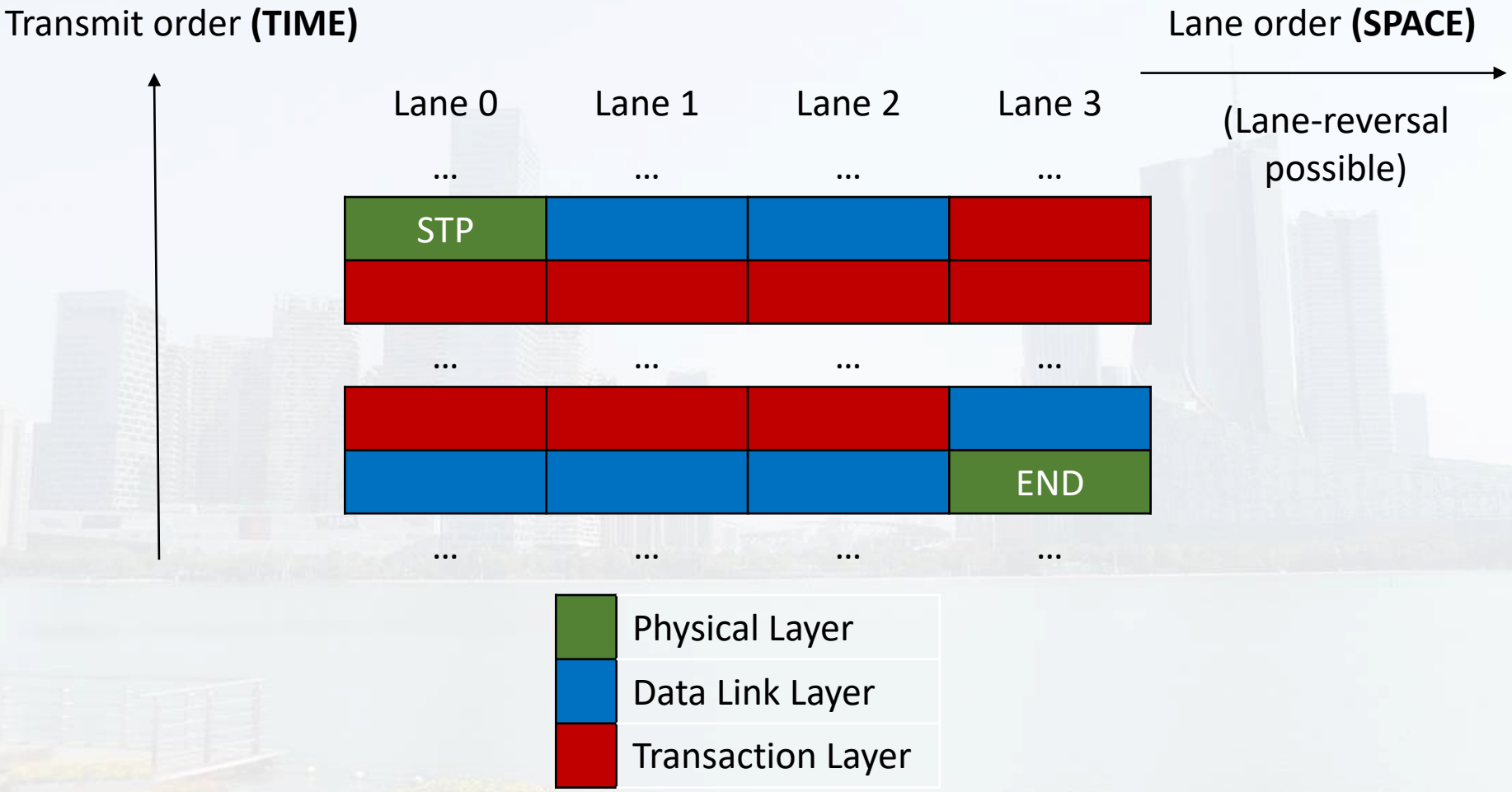
- Training Sequences (TS1, TS2): 1 COM + 15 TS
  - Used to de-skew between lanes
- SKIP: 1 COM + 3 SKP identifiers
  - Used to recalibrate receiver clock
- Fast Training Sequence (FTS): 1 COM + 3 FTS
  - Power management
- Electrical Idle (IDLE): 1 COM + 3 IDL
  - Transmitted continuously when no data
- Electrical Idle Exit (EIEOS): 16 characters (since 2.0)

character: 8 unscrambled bits

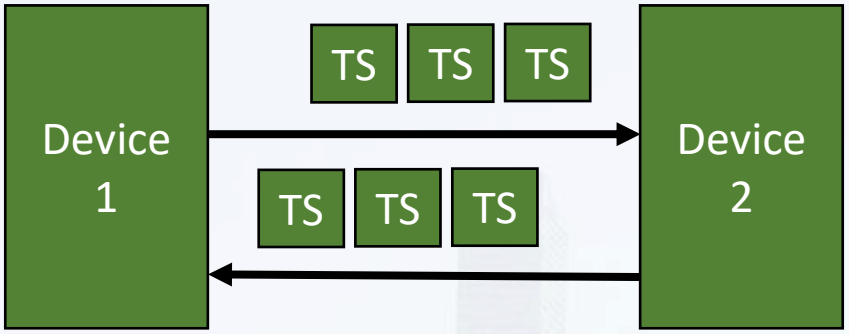
PCIe – Framing (x1)



PCIe – Framing (x4)



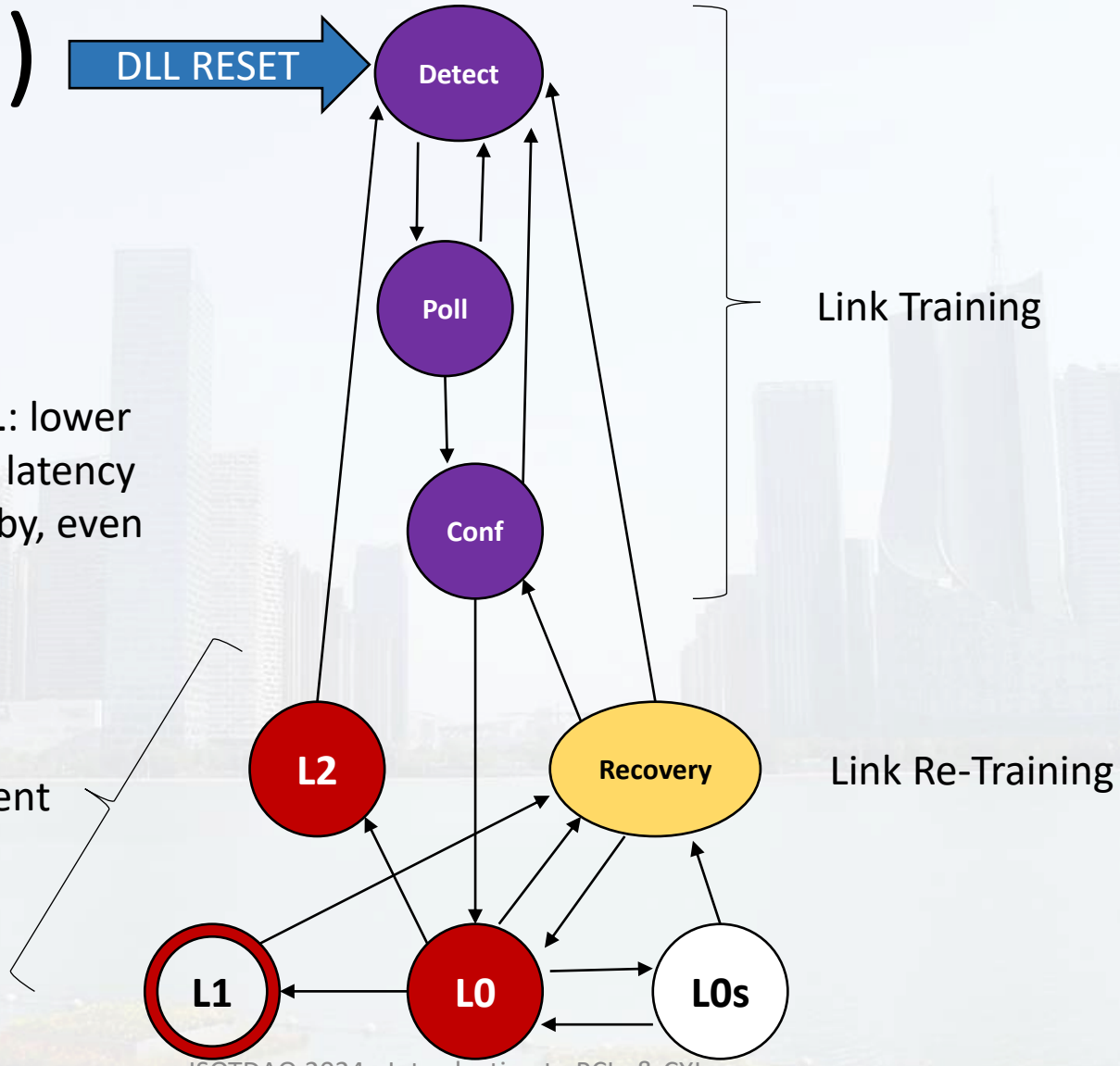
PCIe – Link training



- Lane polarity
- Link width / ordering
- Link equalization
  - Dynamic equalization!
- Link speed
- …

```
37181 ns  EP LTSSM State: RECOVERY.RCVRLOCK
37312 ns   RP PCI Express Link Status Register
(1881):
37312 ns     Negotiated Link Width: x8
37312 ns         Slot Clock Config: System
Reference Clock Used
37949 ns  EP LTSSM State: RECOVERY.RCVRCFG
38845 ns  RP LTSSM State: RECOVERY.RCVRCFG
41053 ns  RP LTSSM State: RECOVERY.SPEED
41309 ns  EP LTSSM State: RECOVERY.SPEED
43573 ns  EP LTSSM State: RECOVERY.RCVRLOCK
43765 ns  RP LTSSM State: RECOVERY.RCVRLOCK
43797 ns  RP LTSSM State: REC_EQULZ.PHASE0
43825 ns  RP LTSSM State: REC_EQULZ.PHASE1
44141 ns  EP LTSSM State: REC_EQULZ.PHASE0
44673 ns  EP LTSSM State: REC_EQULZ.PHASE1
44929 ns  RP LTSSM State: REC_EQULZ.DONE
44949 ns  RP LTSSM State: RECOVERY.RCVRLOCK
45209 ns  EP LTSSM State: REC_EQULZ.DONE
45229 ns  EP LTSSM State: RECOVERY.RCVRLOCK
45425 ns  EP LTSSM State: RECOVERY.RCVRCFG
45581 ns  RP LTSSM State: RECOVERY.RCVRCFG
45925 ns  RP LTSSM State: RECOVERY.IDLE
46073 ns  EP LTSSM State: RECOVERY.IDLE
46169 ns  EP LTSSM State: L0
46313 ns  RP LTSSM State: L0
47824 ns        Current Link Speed: 8.0GT/s
```

PCIe Link-Training State Machine (LTSSM)



- **L0**: active
- **L0 standby**, **L1**: lower power, higher latency
- **L2**: cold standby, even lower power
- **L3**: power off

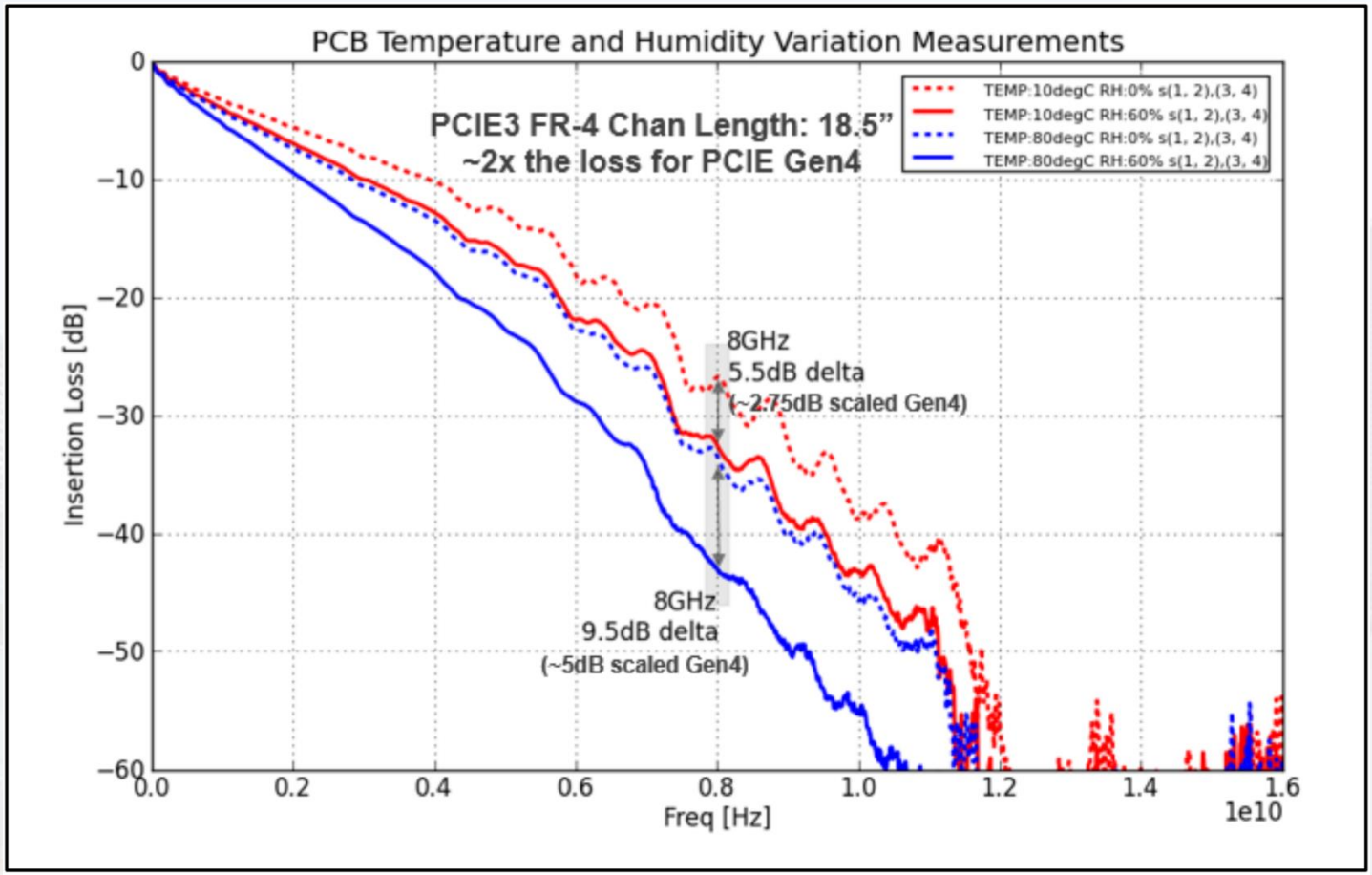
Simulate a PCIe link on your own!

- https://github.com/wyvernSemi/pcievhost
- http://www.anita-simulators.org.uk/wyvernsemi/articles/pci_express.pdf
- Written in C/Verilog
- Compatible with ModelSim (via DPI)
- Simulates link training, flow control, ACK/NAK, completions…

What is this presentation about?

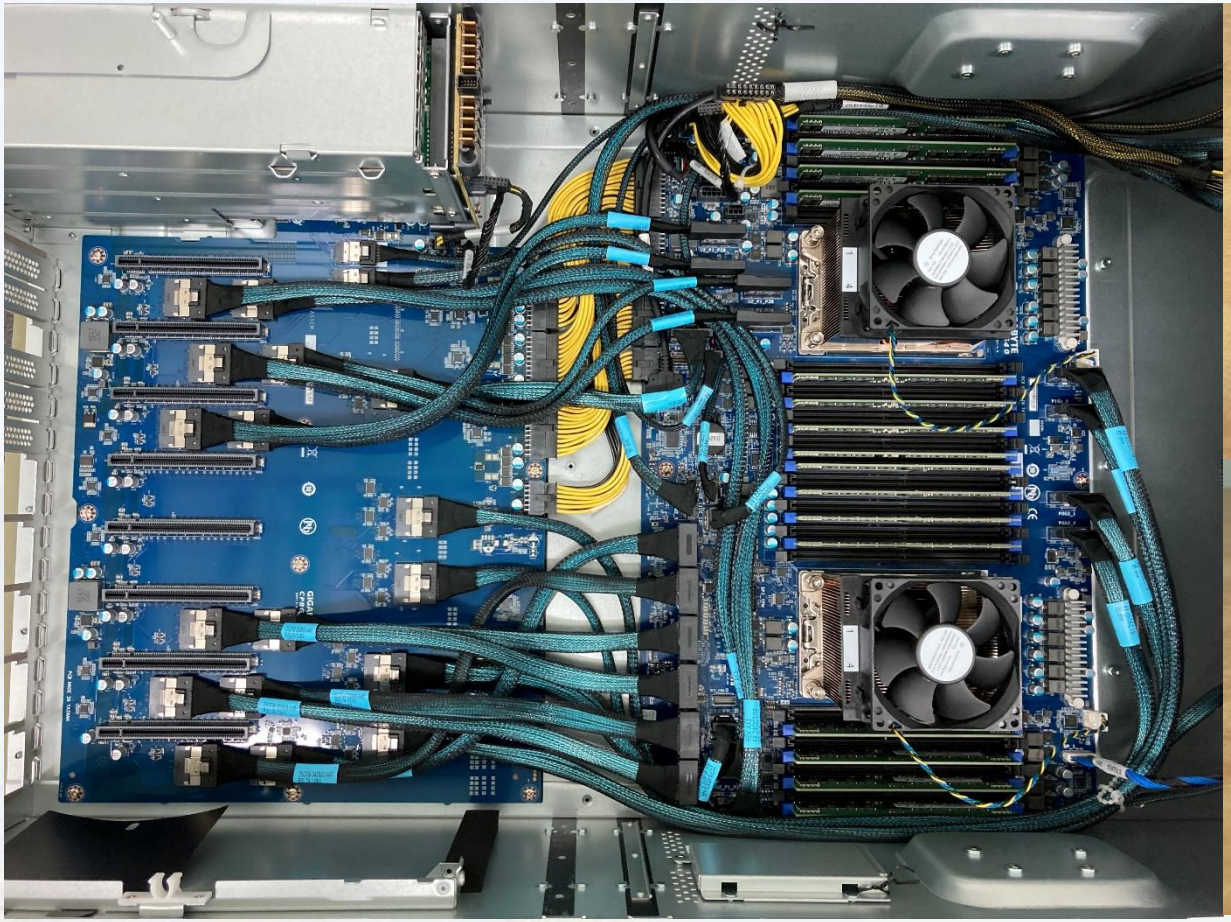
- History and evolution of PCIe
- PCIe concepts
- PCIe layers
- **PCIe practical aspects**
- PCIe performance
- PCIe future roadmap

PCIe link training
Signal integrity – Environment



PCIe link training
Signal integrity – Robustness

PCIe link training
Signal integrity – Connectors



Troubleshooting PCIe deployments at scale

If you run a large data acquisition system, and most of your I/O goes through PCIe links, you **have to** monitor all your endpoints and root ports

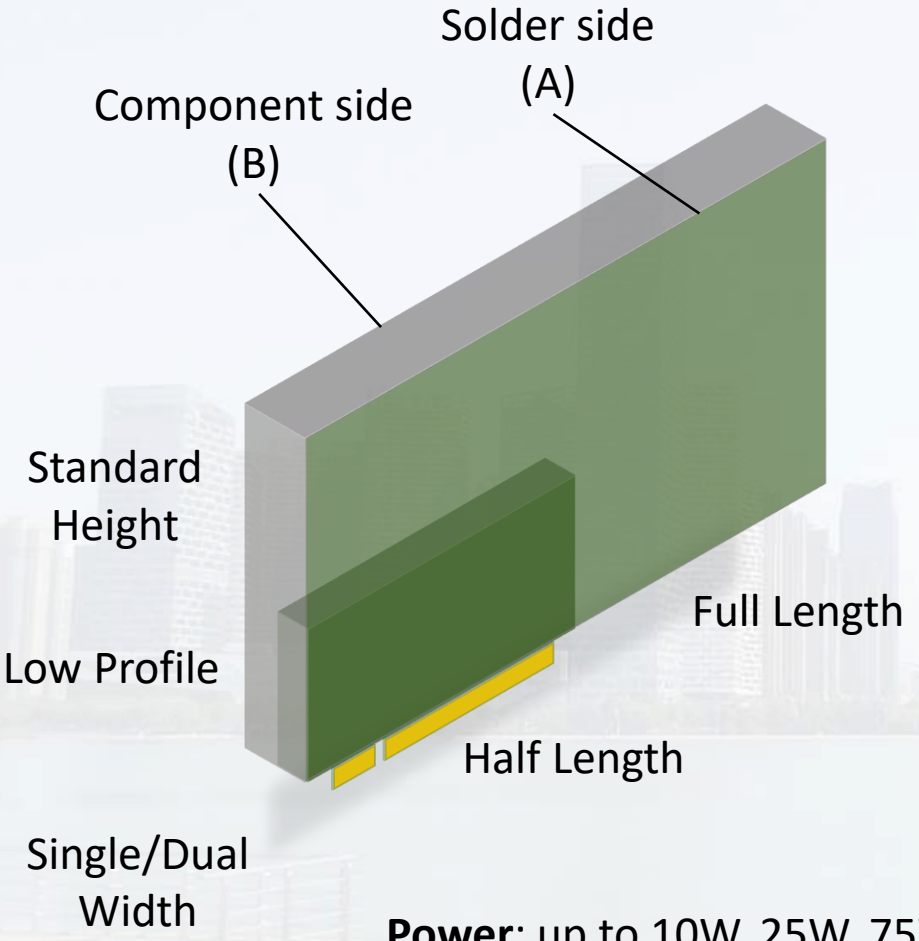
POSTED ON AUGUST 5, 2020 TO OPEN SOURCE

Pcicrawler: A Python-based command-line interface tool to debug PCI issues at scale

https://github.com/facebook/pcicrawler

https://engineering.fb.com/2020/08/05/open-source/pcicrawler

PCIe CEM Spec – AIC form factors



- Standard Height
  - 4.20” (106.7mm)
- Low Profile
  - 2.536” (64.4mm)
- Half Length (e.g.“HHHL”)
  - 6.6” (167.65mm)
- Full Length (e.g. “FHFL”)
  - 12.283” (312mm)

Power: up to 10W, 25W, 75W, 300W or 375W depending on form factor & optional extra power connectors

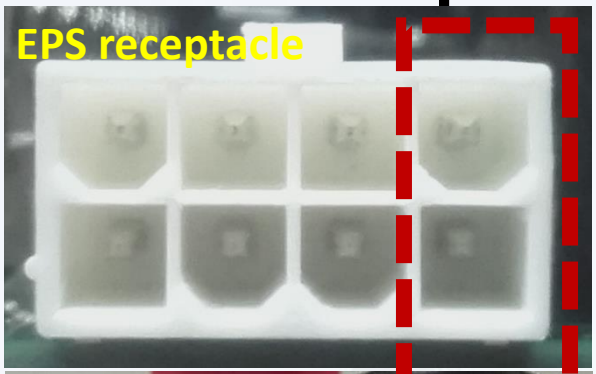
PCIe storage – More form factors

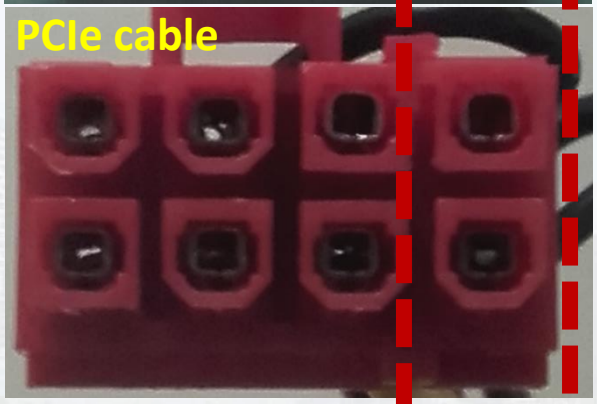
M.2
≤ 4 lanes

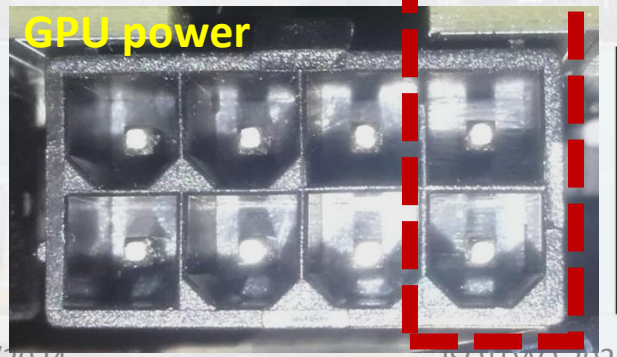
U.2
≤ 4 lanes

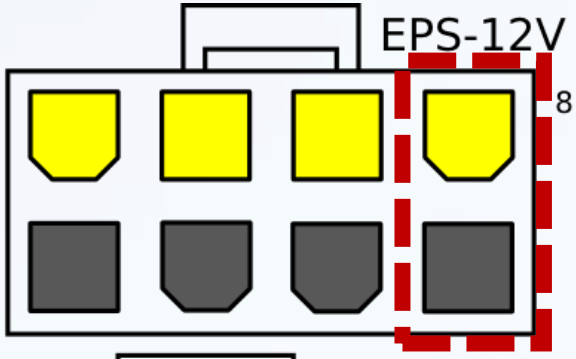
“ruler” (EDSFF, NGSFF) ≤ 8 lanes

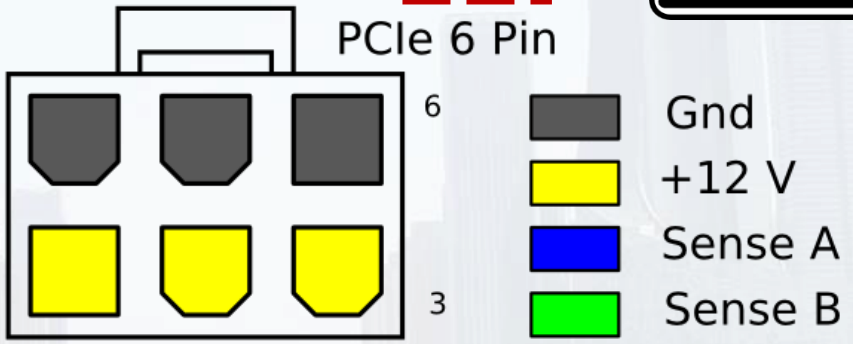
PCIe CEM Spec – Power Cables

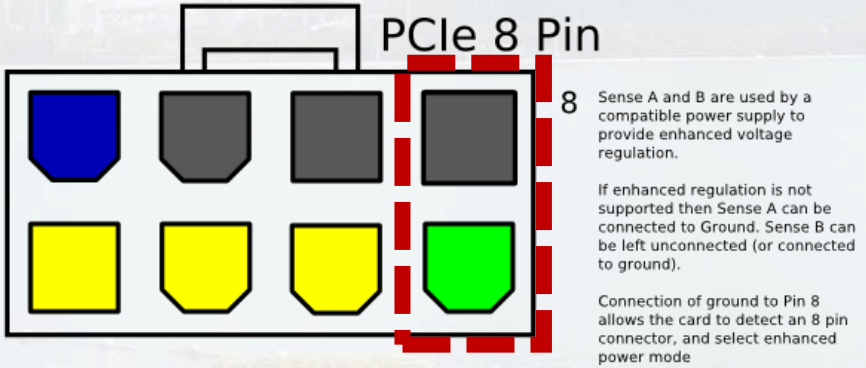












PCIe CEM Spec – Power Cables

Solutions Products Company AMD XILINX

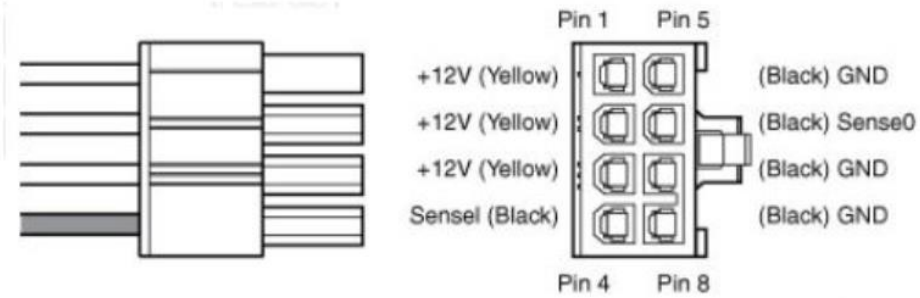
72298 - Alveo Data Center Accelerator Card - !CAUTION! Do not use EPS12V / ATX12V power source in place of PCI Express Auxiliary Power connector

Feb 16, 2023 · Knowledge

https://support.xilinx.com/s/article/72298?language=en_US

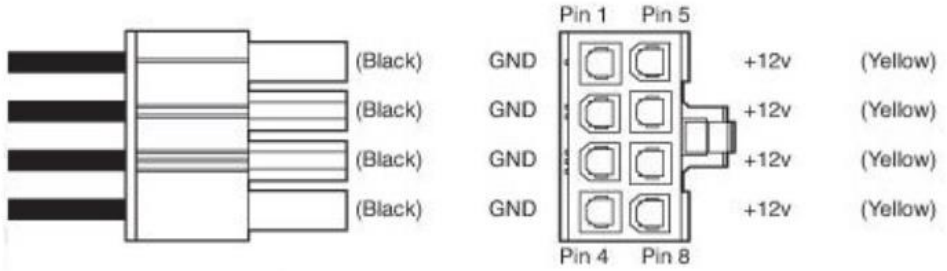
PCI Express Auxiliary 8-pin Power Cable pinout

For use with Alveo Data Center Accelerator Card



CPU 12V (EPS12V / ATX12V) 8-pin Power Cable pinout

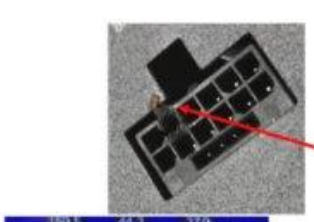
DO NOT USE with an Alveo Data Center Accelerator Card



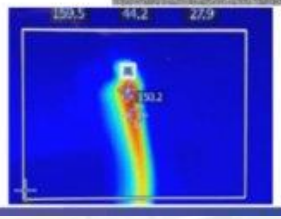
PCIe – 12VHPWR power failures

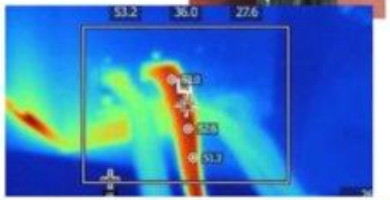
Observations

- Images below show overheating of the power connector at mating point. Multiple suppliers and designs have failed
- Cables with low cycles and without bend condition have not failed
- Failures observed on both rows of pins depending on load direction
- Hot spots observed @~2.5hrs, melting 10-30hrs
- Note - Also observed after high mating cycles ~40, straight plug w/o side load



Pin 11



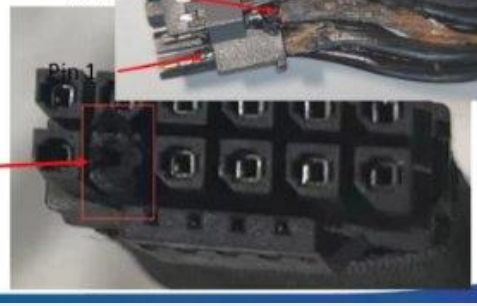


Pin 6

Pin 2

Pin 1

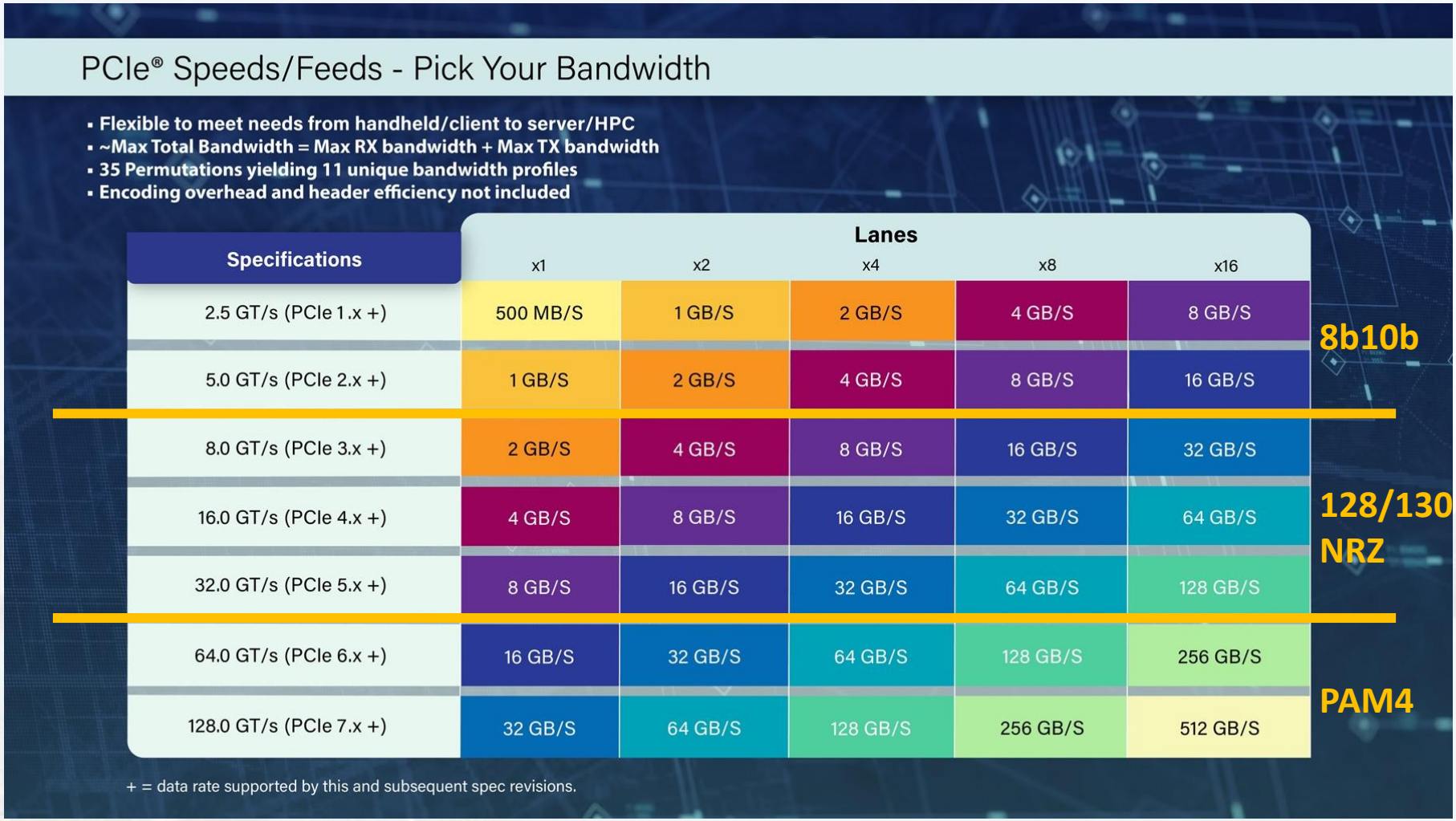
Pin 2



What is this presentation about?

- **PCIe history and evolution**
- PCIe concepts
- PCIe layers
- PCIe practical aspects
- **PCIe performance**
- PCIe future roadmap

PCIe – Theoretical data rates



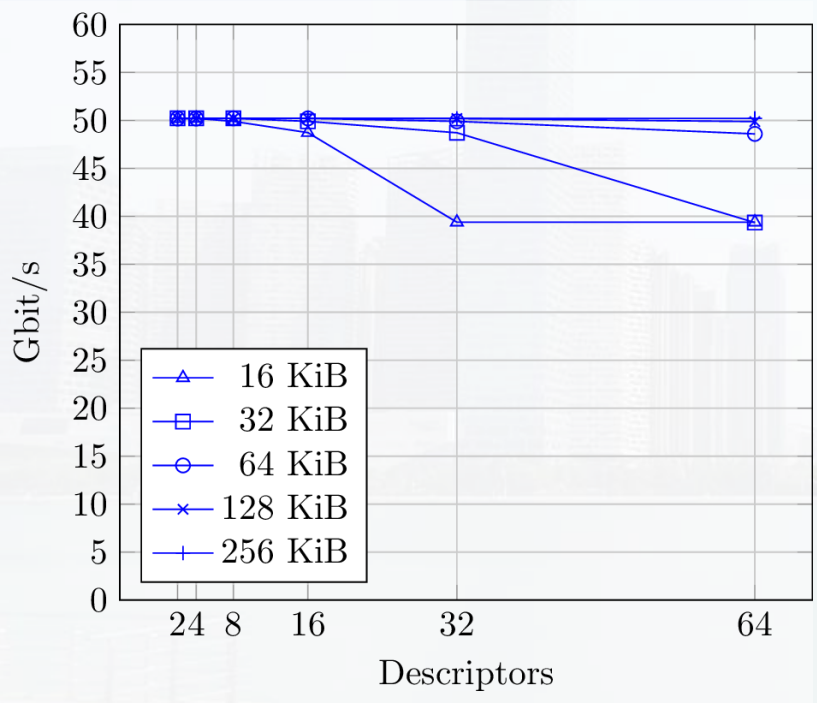
PCIe – Effective data rates

Theoretical bandwidth — Packet efficiency

- $\rho = \frac{Lane\ rate \times Lane\ width}{Encoding} \times \frac{MPS}{MPS+Headers}$
- Example: Gen2 x8, 128 Bytes MPS
  - $\rho = 40 \times 0.8 \times \frac{128}{128+24}$ = 32 x 0.84 = 26.9 Gb/s
- Example: Gen3 x8, 128 Bytes MPS
  - $\rho = 64 \times 0.98 \times \frac{128}{128+24}$ = 62.7 x 0.84 = 52.6 Gb/s
- Example: Gen3 x8, 256 Bytes MPS
  - $\rho = 64 \times 0.98 \times \frac{256}{256+24}$ = 62.7 x 0.91 = 57 Gb/s

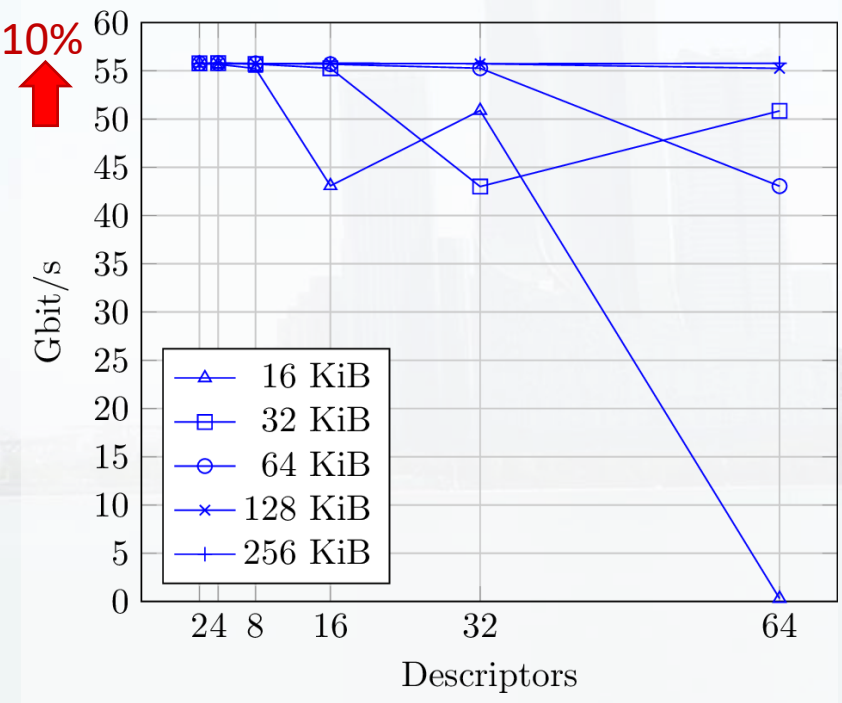
PCIe 3.0 x8 – DMA Performance

MPS = 128 Bytes

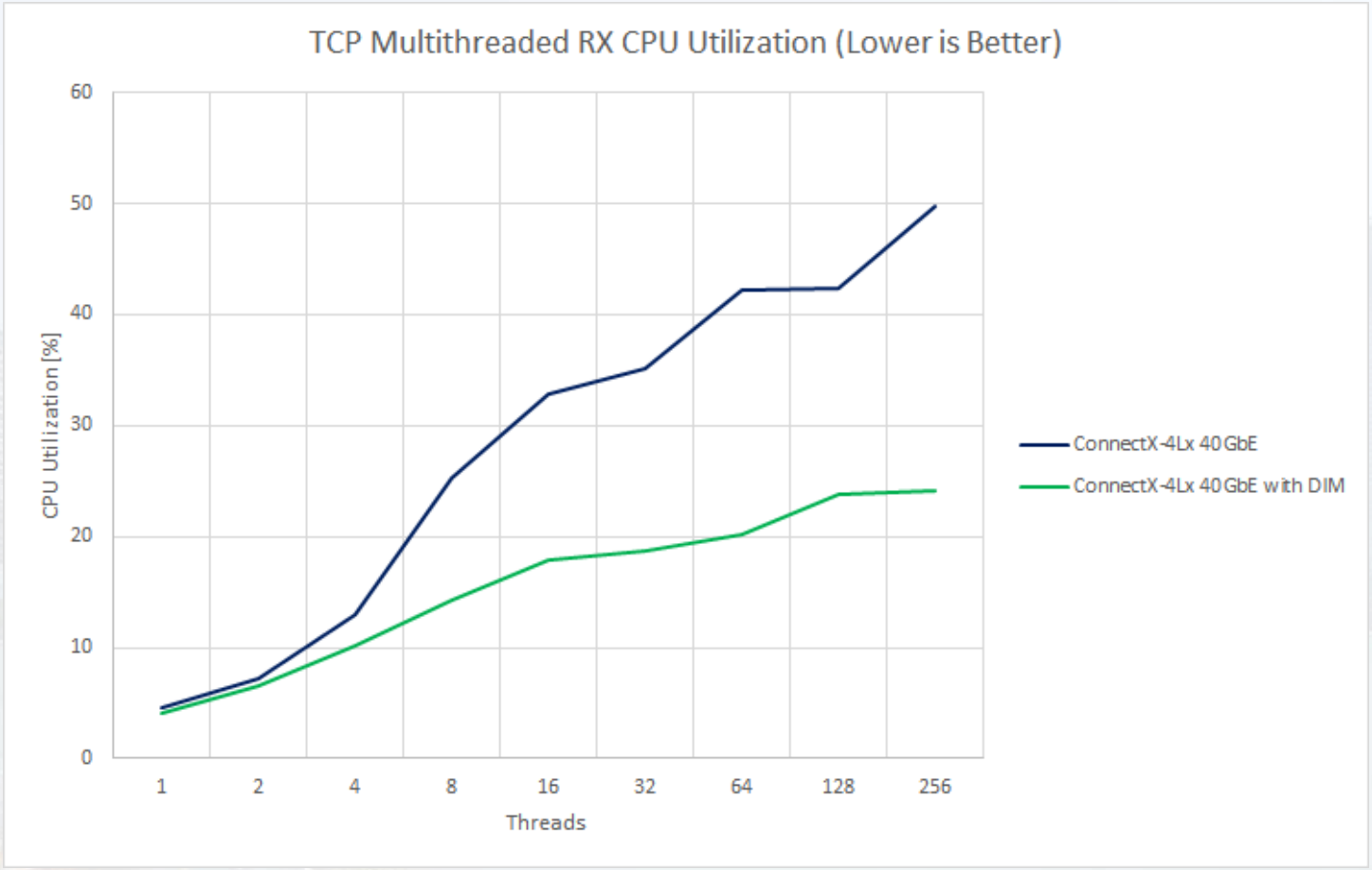


MPS = 256 Bytes

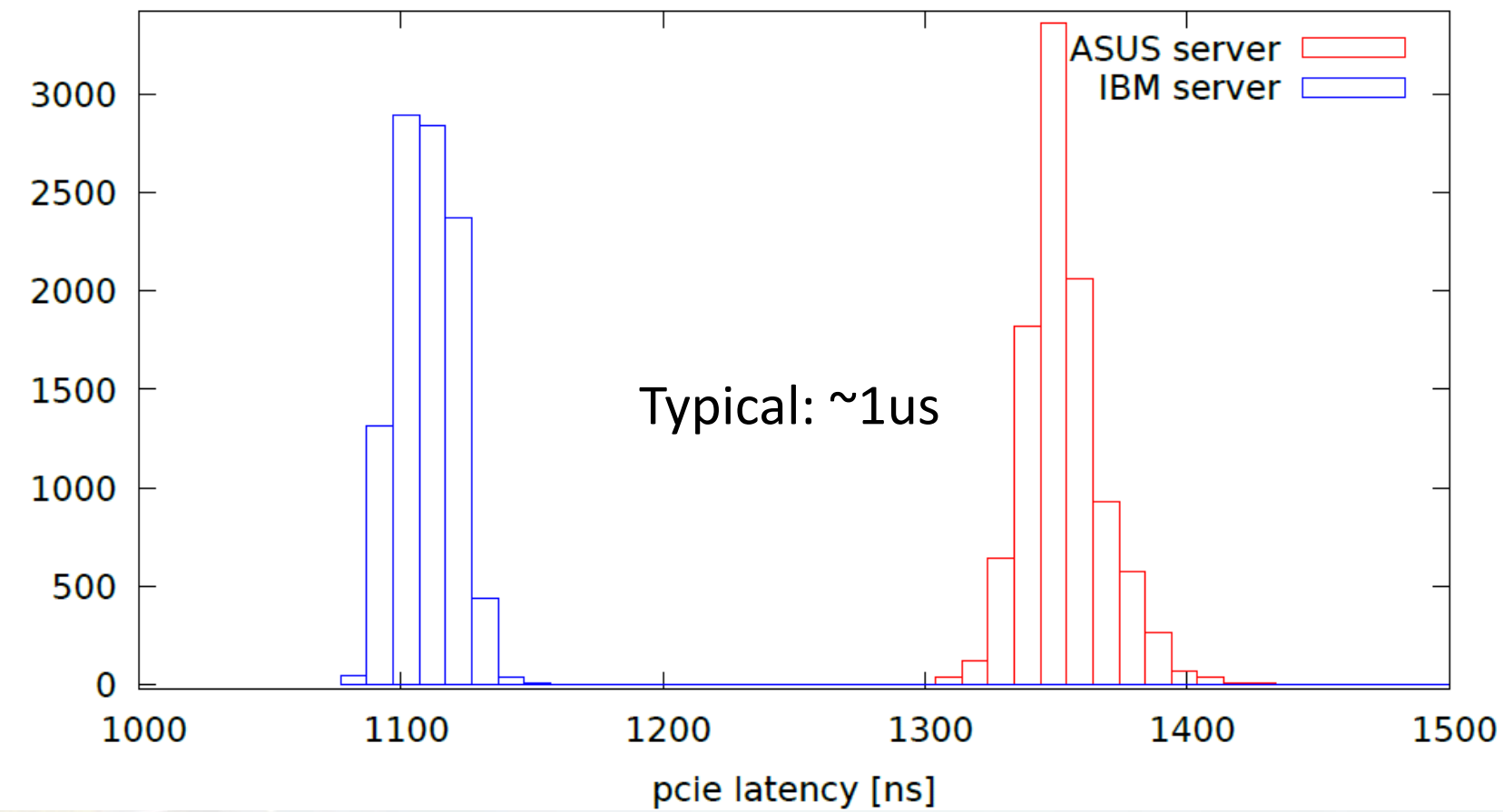
10%



PCIe performance – interrupt coalescing



PCIe performance – latency



NVMe performance

NVM Express (**NVMe**) is an interface specification for accessing a computer's non-volatile storage media attached via the PCIe bus.

NVM Express allows host hardware and software to fully exploit the levels of parallelism possible in modern SSDs.

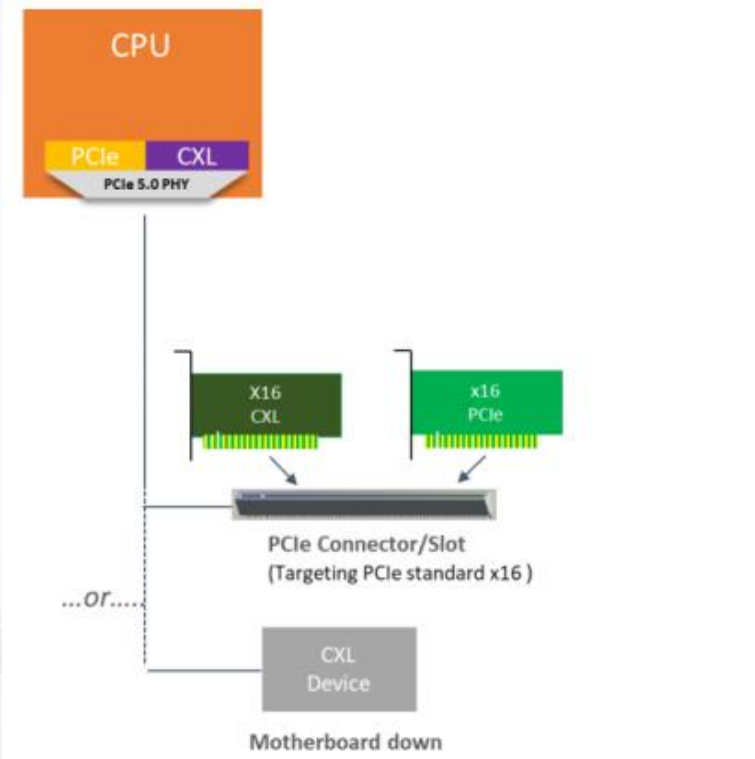
Specifications	4TB	2TB	1TB	500GB
Standard Model	ZP4000GM30013	ZP2000GM30013	ZP1000GM30013	ZP500GM30013
Interface	PCIe® Gen4 ×4 NVMe 1.4	PCIe Gen4 ×4 NVMe 1.4	PCIe Gen4 ×4 NVMe 1.4	PCIe Gen4 ×4 NVMe 1.4
NAND Flash Memory	3D TLC	3D TLC	3D TLC	3D TLC
Form Factor	M.2 2280-D2	M.2 2280-D2	M.2 2280-S2	M.2 2280-S2
Performance				
Sequential Read (Max, MB/s), 128KB[2]	7300	7300	7300	7000
Sequential Write (Max, MB/s), 128KB[2]	6900	6900	6000	3000
Random Read (Max, IOPS), 4KB QD32 T8[2]	1,000,000	1,000,000	800,000	400,000
Random Write (Max, IOPS), 4KB QD32 T8[2]	1,000,000	1,000,000	1,000,000	700,000

What is this presentation about?

- PCIe history and evolution
- PCIe concepts
- PCIe layers
- PCIe practical aspects
- PCIe performance
- **PCIe future roadmap**

What is Compute Express Link?

- Alternate protocol that runs across the standard PCIe physical layer
- 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 5.0
  - 8 Gbps in degraded mode



Compute Express Link has the benefit of supporting both standard PCIe devices as well as CXL devices – all on the same Link

CXL Consortium

- Alibaba, Cisco, Dell EMC, Facebook, Google, Hewlett Packard Enterprise, Huawei, Intel Corporation and Microsoft announced their intent to incorporate in March 2019.
- The Compute Express Link (CXL) Consortium and Gen-Z Consortium developed an execution of a Memorandum of Understanding (MOU), describing a mutual plan for collaboration between the two organizations in April 2020.

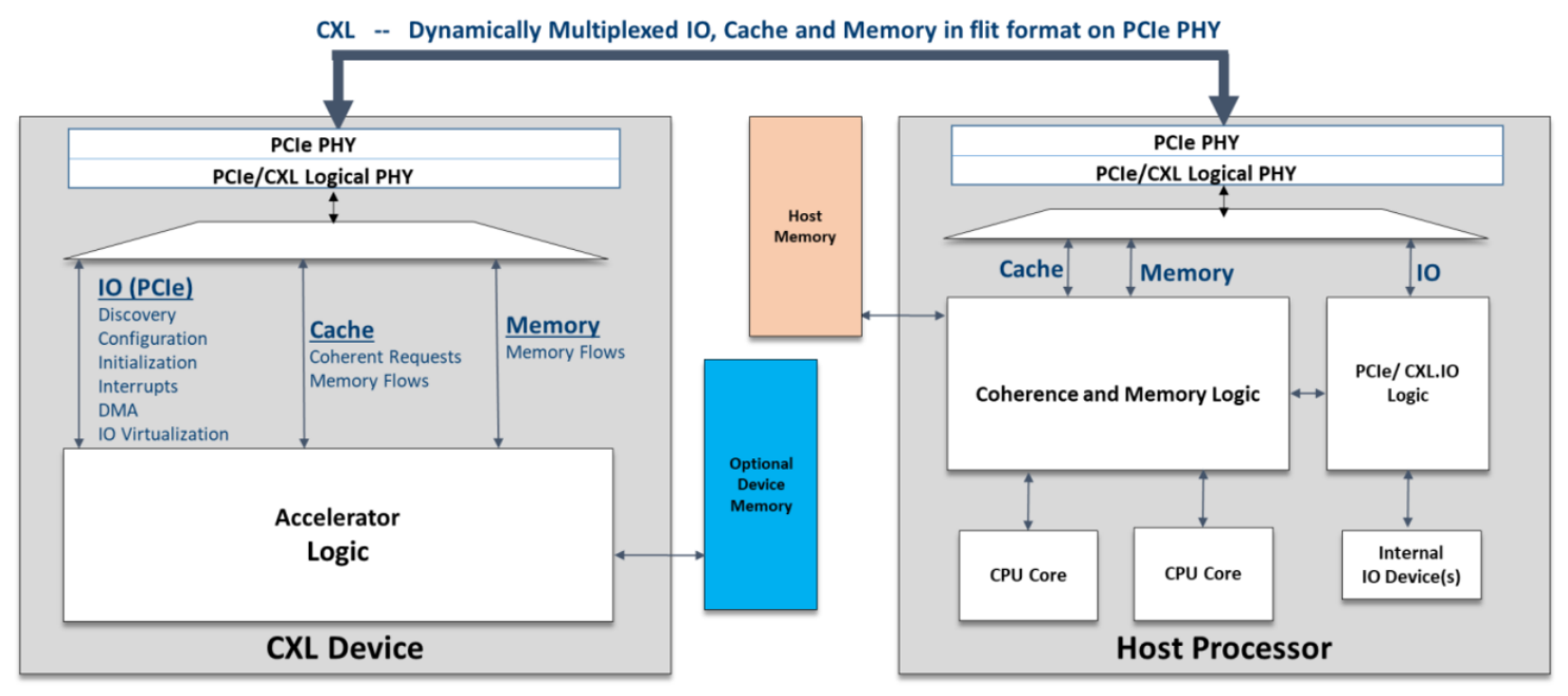
Why CXL?

Need a new class of interconnect for **heterogenous computing** and **disaggregation** usages:

- Efficient resource sharing
- Shared memory pools with efficient access mechanisms
- Enhanced movement of operands and results between accelerators and target devices
- Significant latency reduction to enable disaggregated memory

CXL – Dynamic Multiplexing

CXL multiplexes **three different protocols** at the PCIe PHY layer

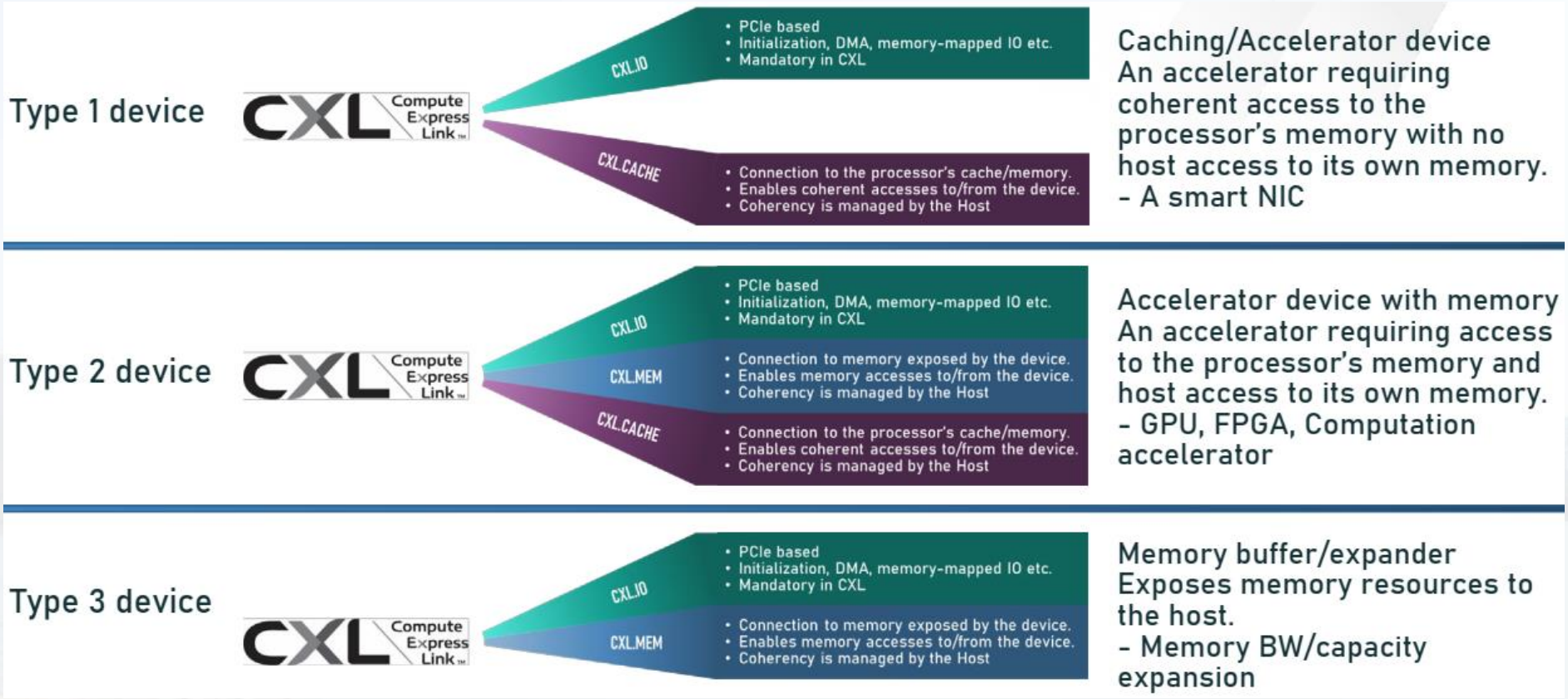


CXL protocols

- **cxl.io**
  - device discovery, configuration, initialization, I/O virtualization, and direct memory access (DMA)
- **cxl.cache**
  - enables a device to cache data from the host memory, employing a simple request and response protocol
  - the host processor manages coherency of data
- **cxl.memory**
  - allows a host processor to access memory attached to a CXL device

CXL device types

"Mix and match" protocols depending on application requirements

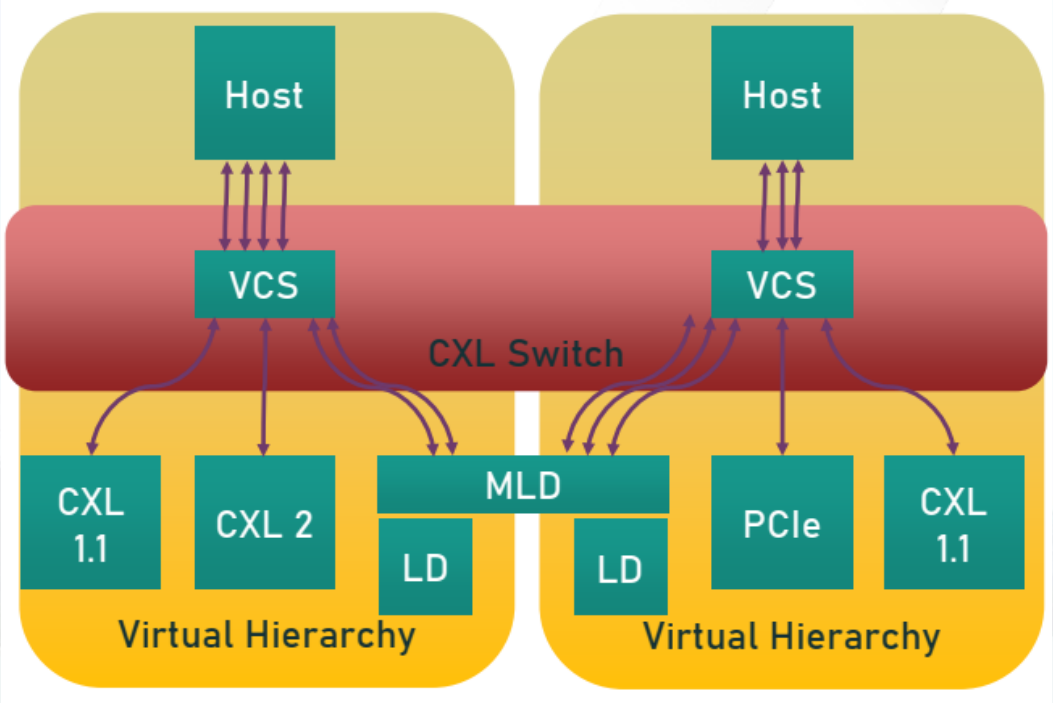


CXL evolution timeline

- CXL 1.0 – March 2019
  - enables device-level memory expansion and coherent acceleration modes
- CXL 1.1 – September 2019
- CXL 2.0 – November 2020
  - augments CXL 1.1 with enhanced fanout support and a variety of additional features
- CXL 3.0 – in the making
- CXL supporting platforms coming to market now

CXL 2.0 new features

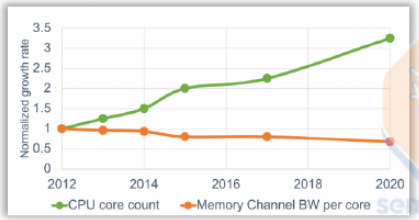
- CXL switches
  - Multiple host
  - Virtual hierarchies
- Multi-Logic devices
- Management
  - Fabric manager
  - Device allocation
  - QoS telemetry
  - Memory interleaving



CXL-attached memory

Summary of Data Center Memory Challenges

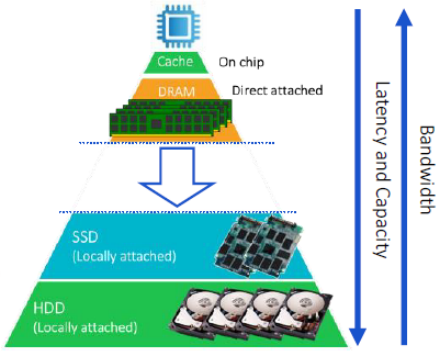
Decreasing memory bandwidth per core



OCP GLOBAL SUMMIT

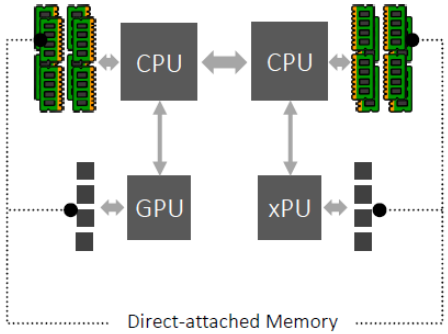
Source: Meta, OCP Summit Presentation Nov. '21

Huge latency and capacity gap



Server Memory Hierarchy

Stranded memory resources and low utilization



CXL-attached memory is the current focus of the CXL ecosystem

Data • Faster • Safer

4

Conclusions

- PCIe has a track record of 2x throughput improvements per generation
- PCIe has maintained backwards compatibility for decades
- PCIe has won the interconnect wars
  - Gen-Z has joined the CXL consortium
  - All CCIX consortium members have moved to CXL
  - CAPI never gained mindshare outside of IBM
- PCIe is proving suitable also for chip-to-chip interconnect
  - Universal Chiplet Interconnect Express (UCIe)
  - NVLink-C2C will be compatible with CXL
- However, the AI revolution has left PCIe behind
  - AI prefers low link counts at much higher rates to save chip shoreline area, minimizing latency is less of a concern
  - Proprietary alternatives: NVLink, AMD AFL, Google ICI, UltraEthernet…