Flash Memory Summit

CXL™ Switching

Presenter: Tam Do, Technical Staff Engineer

Microchip Technology Inc.

MICROCHIP

CXL™ Overview

- Using PCI Express® (PCIe®) physical and electrical interface standard
- Enables efficient high-bandwidth interface between CPU, memory and accelerators
- Maintains memory coherency between CPU memory space and memory on CXL-attached devices that enables resource sharing, memory disaggregation with memory pooling and sharing
- Open industry standard starting with CXL 1.0 to 3.0

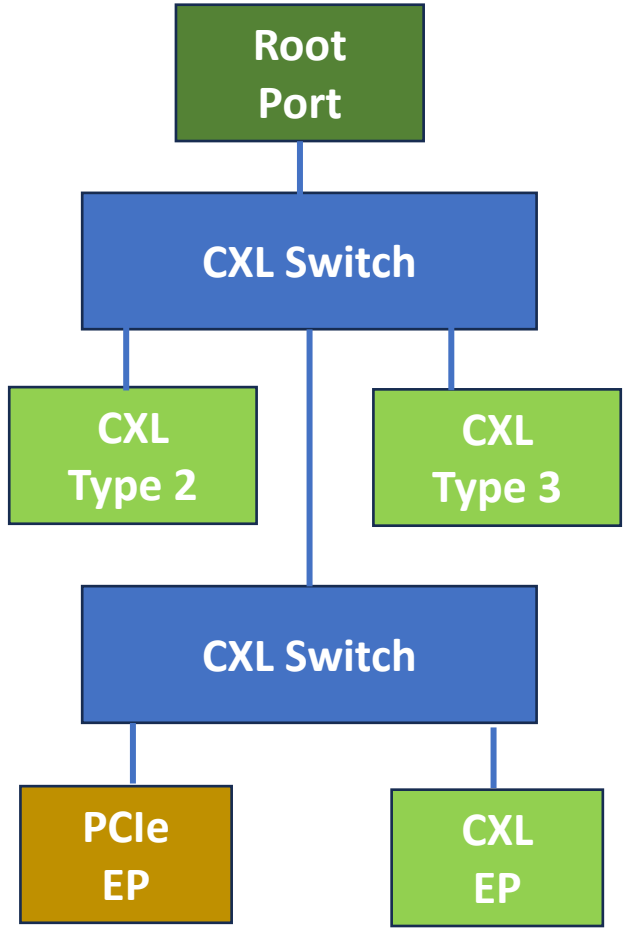
MICROCHIP

CXL™ Protocols

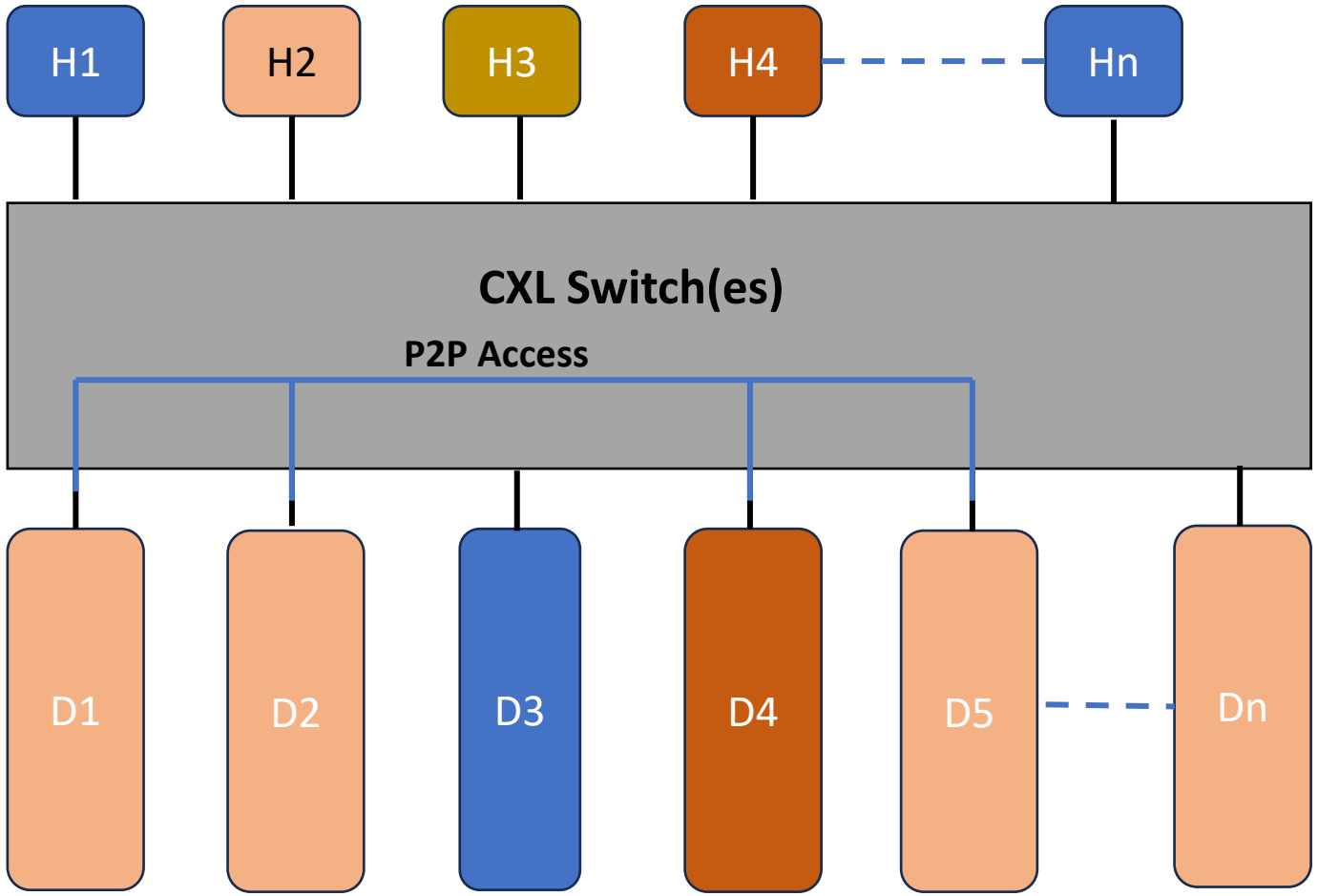
- CXL.io:
  - This protocol is *functionally* equivalent to the PCIe® protocol—and utilizes the broad industry adoption and familiarity of PCIe. As the foundational communication protocol, CXL.io is versatile and addresses a wide range of use cases.
- CXL.cache:
  - This protocol, which is designed for more specific applications, enables accelerators to efficiently access and cache host memory for optimized performance.
- CXL.memory:
  - This protocol enables a host, such as a processor, to access device-attached memory using load/store commands.

Flexible CXL™ Switching

- Connects multiple hosts and memory devices
- Provides low latency, high-bandwidth interfaces between various CXL devices, such as CPUs, GPUs and memory modules
- It enables these devices to share data through multiple host platforms
- Supports downstream PCIe® links



CXL™ 3.0: Device to Device Connectivity



- CXL 3.0 enables **non-tree topologies and peer-to-peer communication (P2P)** within a virtual hierarchy of devices
- PCIe® is **tree topology**

CXL™ 2.0 and 3.0 Switch Comparison

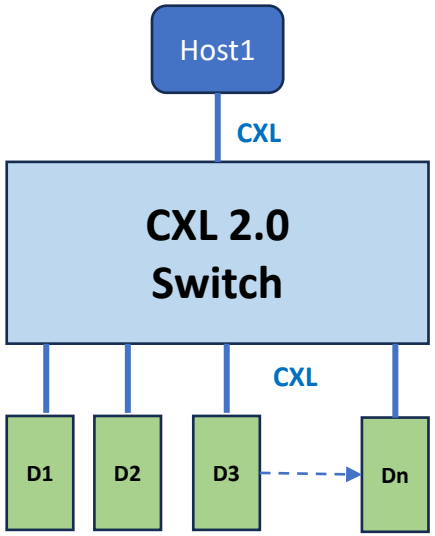
Flash Memory Summit

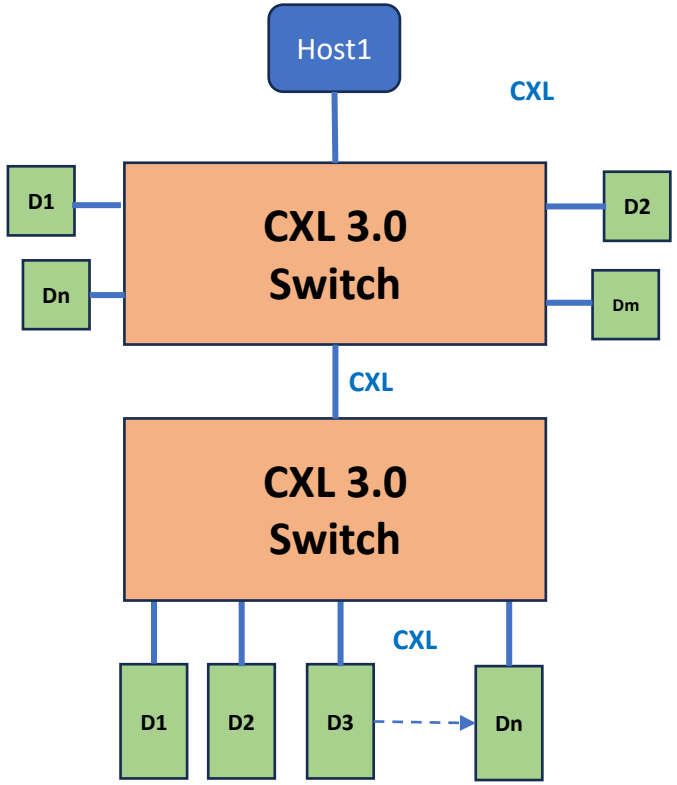
- CXL 2.0
  - Supports single level
  - Fan-out and multi-host
  - Superset of PCIe® switch
  - Enables memory expansion
  - Native PCIe DS links
  - Virtual CXL Switch (VCS) hierarchy: a host is isolated from other hosts
  - Supports pooling and sharing through the switch
- CXL 3.0
  - Superset of CXL 2.0
  - Supports multiple levels (cascade)
  - Fanout to all device types
  - Each host's root port can connect to multiple device types
  - Fabric interconnect with leaf switches

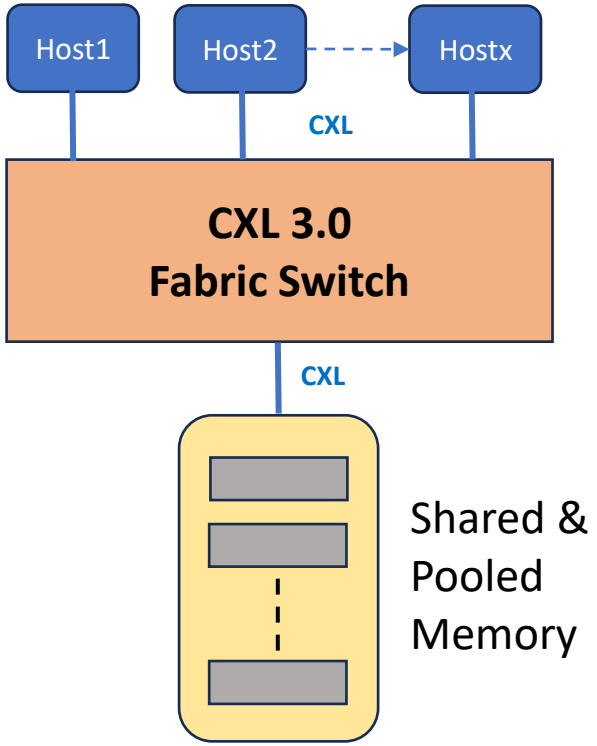
MICROCHIP

CXL™ 2.0 & 3.0 Switch Comparison Diagram

Flash Memory Summit







MICROCHIP

Flash Memory Summit

Memory Sharing/Pooling in CXL™ Ecosystem

Presenter: Chetana Kaushik, Principal Applications Engineer

Microchip Technology Inc.

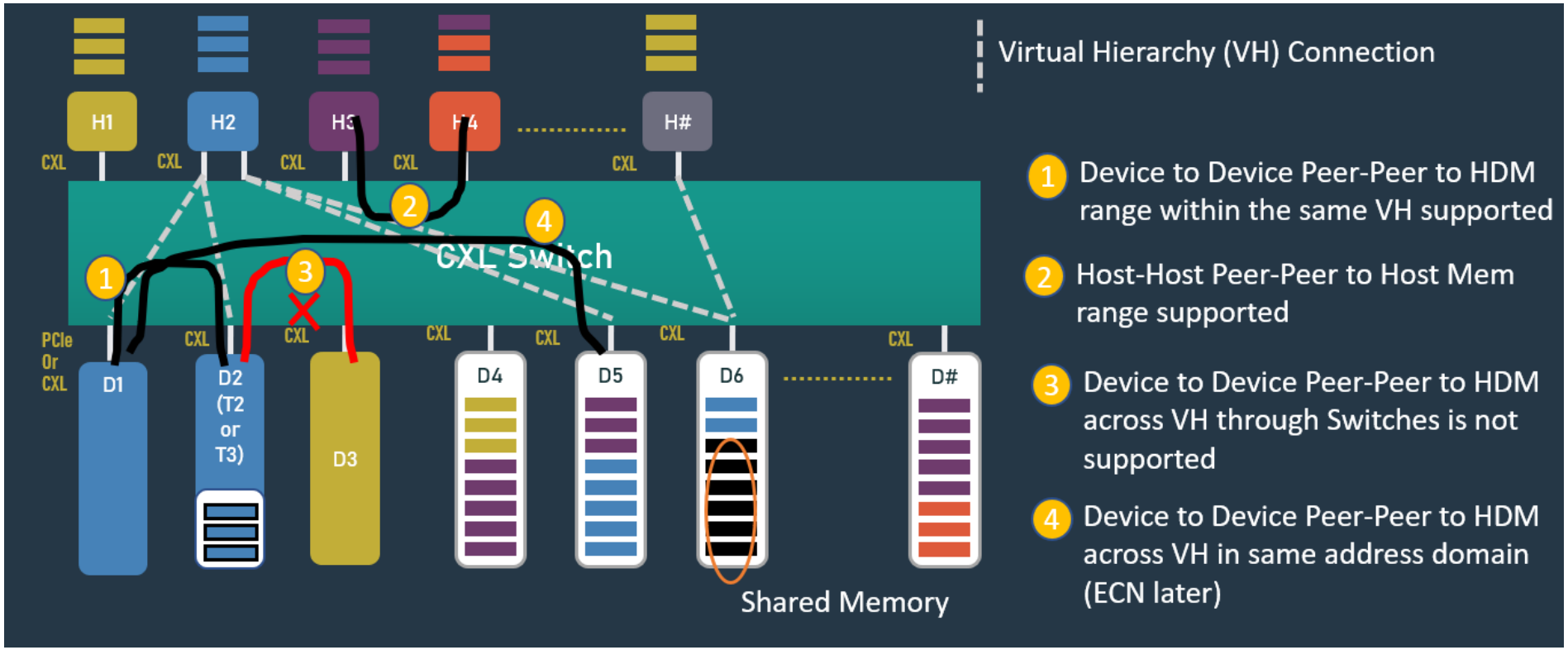
MICROCHIP

CXL™ Devices

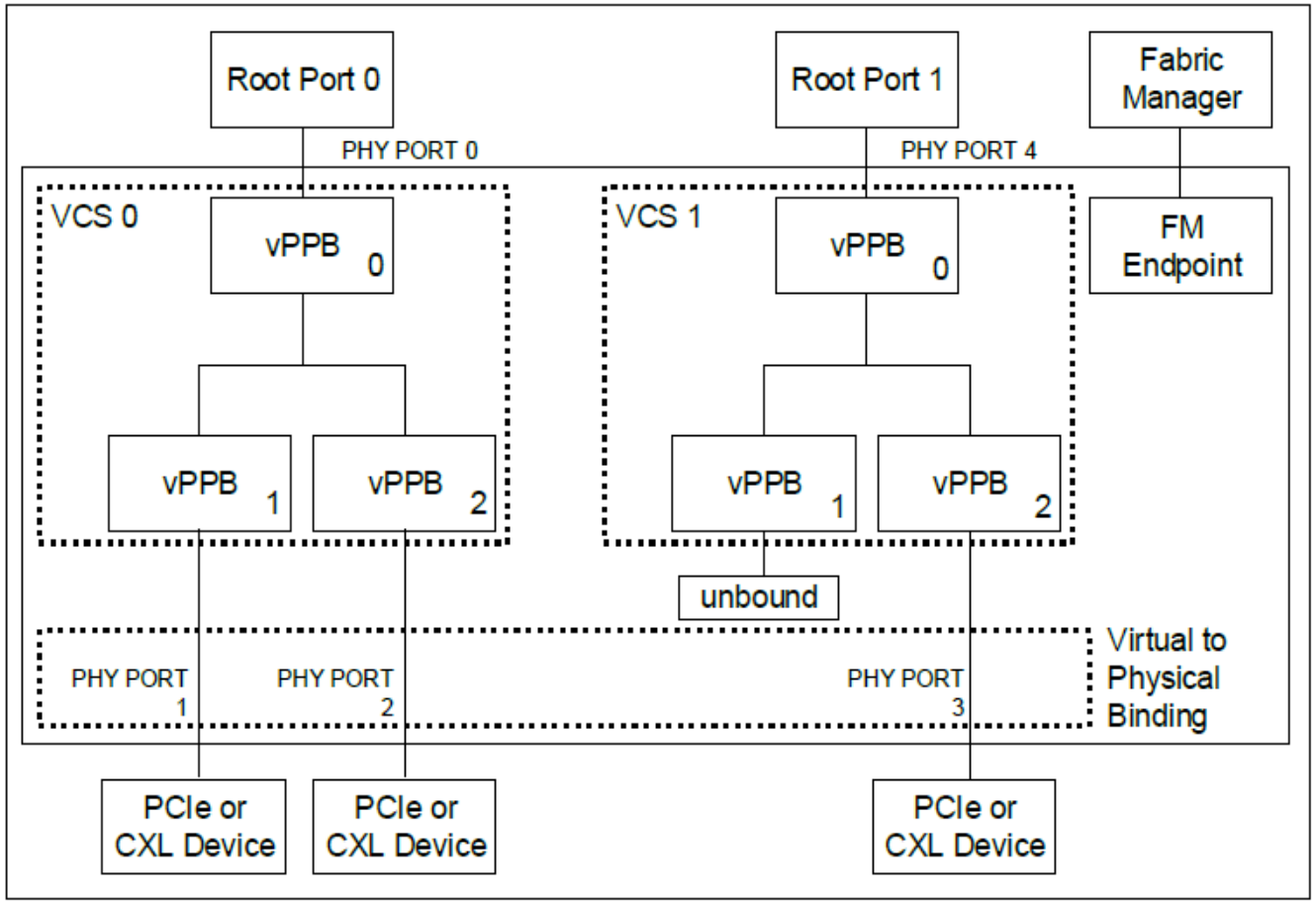
- CXL supports different types of devices that can be interconnected using CXL interface like CPU, GPU, ASIC, FPGA, Accelerators, etc.
- Type 1
  - Processor device with shareable cache memory
  - Uses .io and .cache protocols
- Type 2
  - A Type 1 device and has sharable data memory
  - Uses .io, .cache and .mem protocols
- Type 3
  - Memory expansion
  - Uses .io, .mem protocols

MICROCHIP

Memory Pooling



Binding/Unbinding Devices



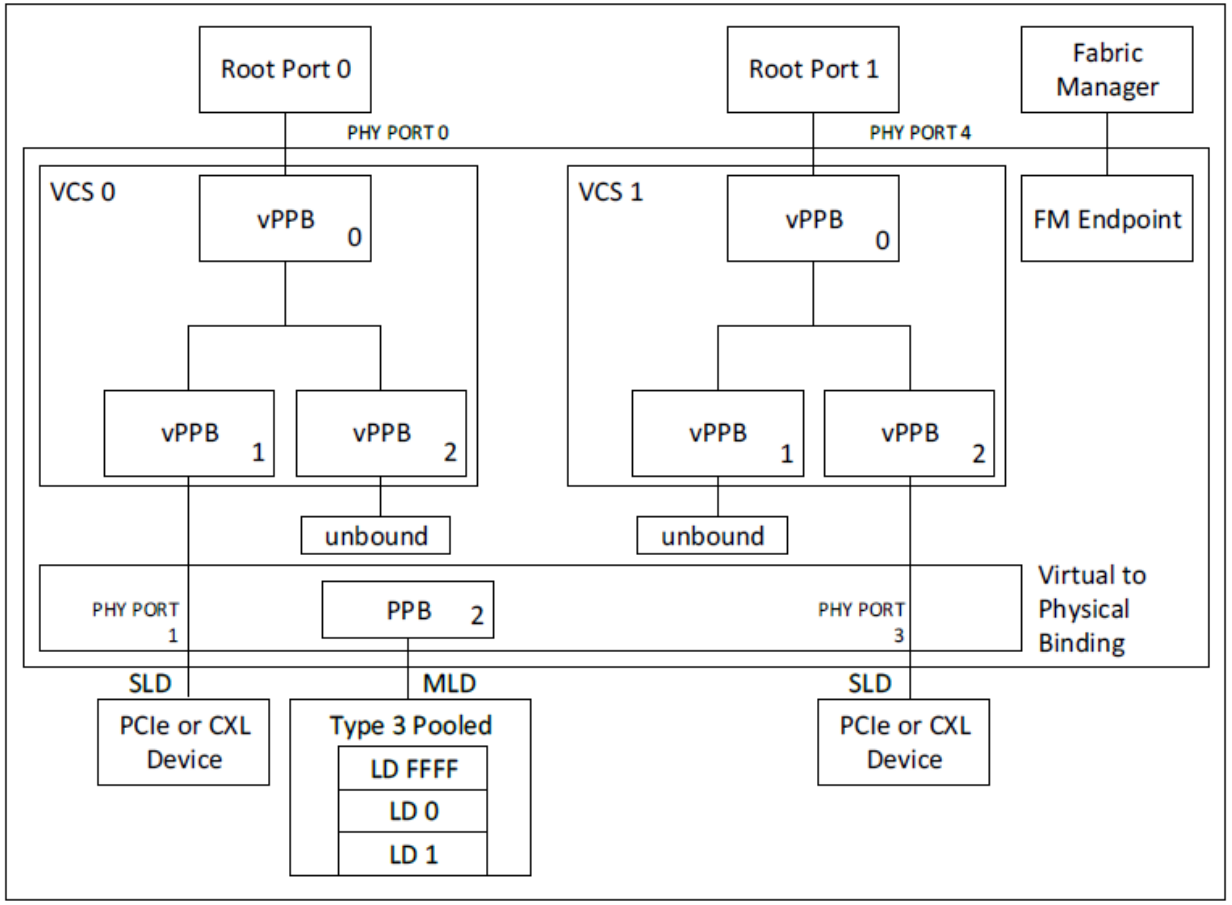
- VCS – Virtual CXL™ Switch
- vPPB – Virtual PCI®-to-PCI Bridge
- FM – Fabric Manager

Binding and Unbinding of a Pooled Device

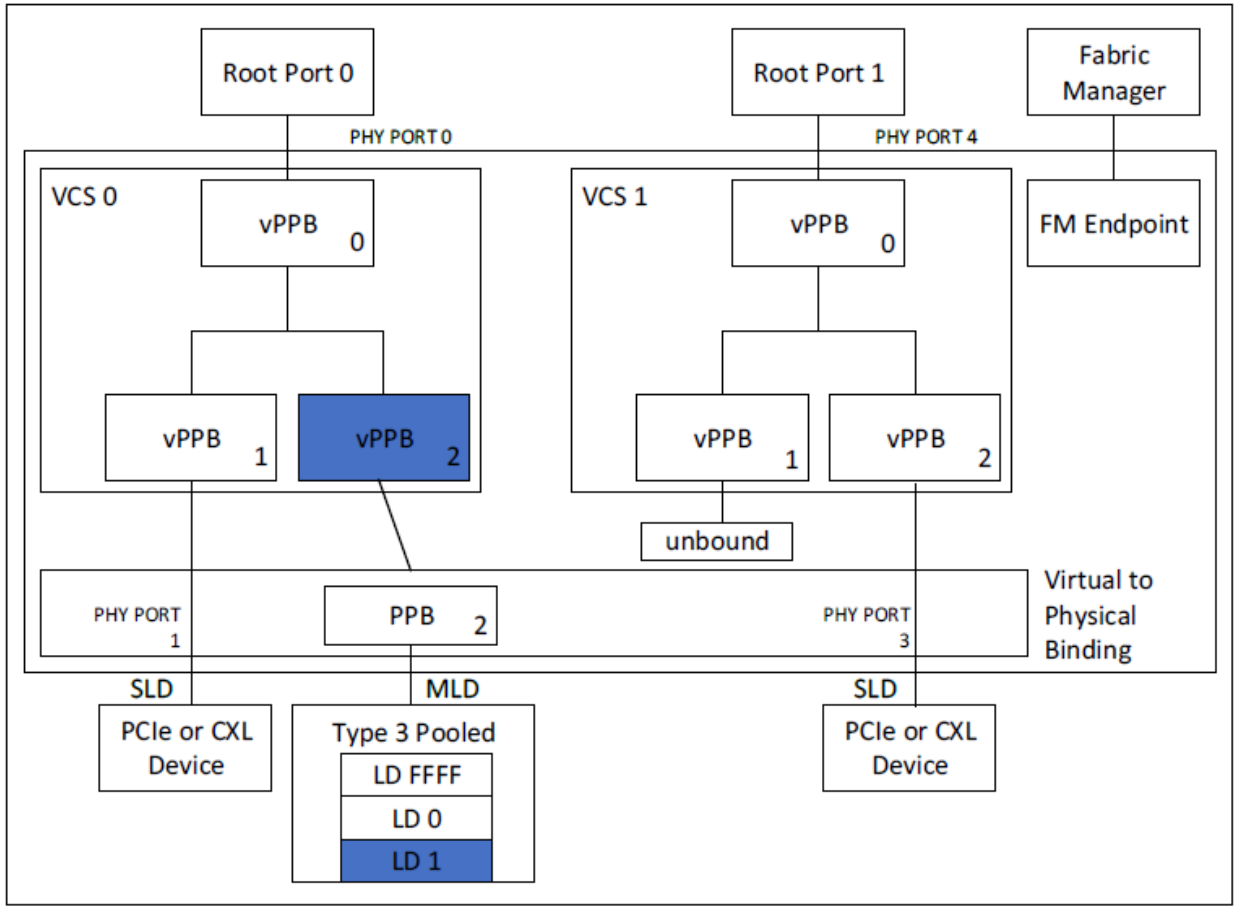
- The physical link cannot be impacted by binding and unbinding of a Logical Device within an MLD component.
- The physical PPB for an MLD port is always owned by the FM. The FM is responsible for port link control, AER, DPC, etc., and manages it using the FM API.
- The FM may need to manage the pooled device to change memory allocations, enable the LD, etc.

MICROCHIP

Example of a CXL Switch before Binding of LDs within Pooled Device



Example of a CXL Switch after Binding of LD-ID 1 within Pooled Device



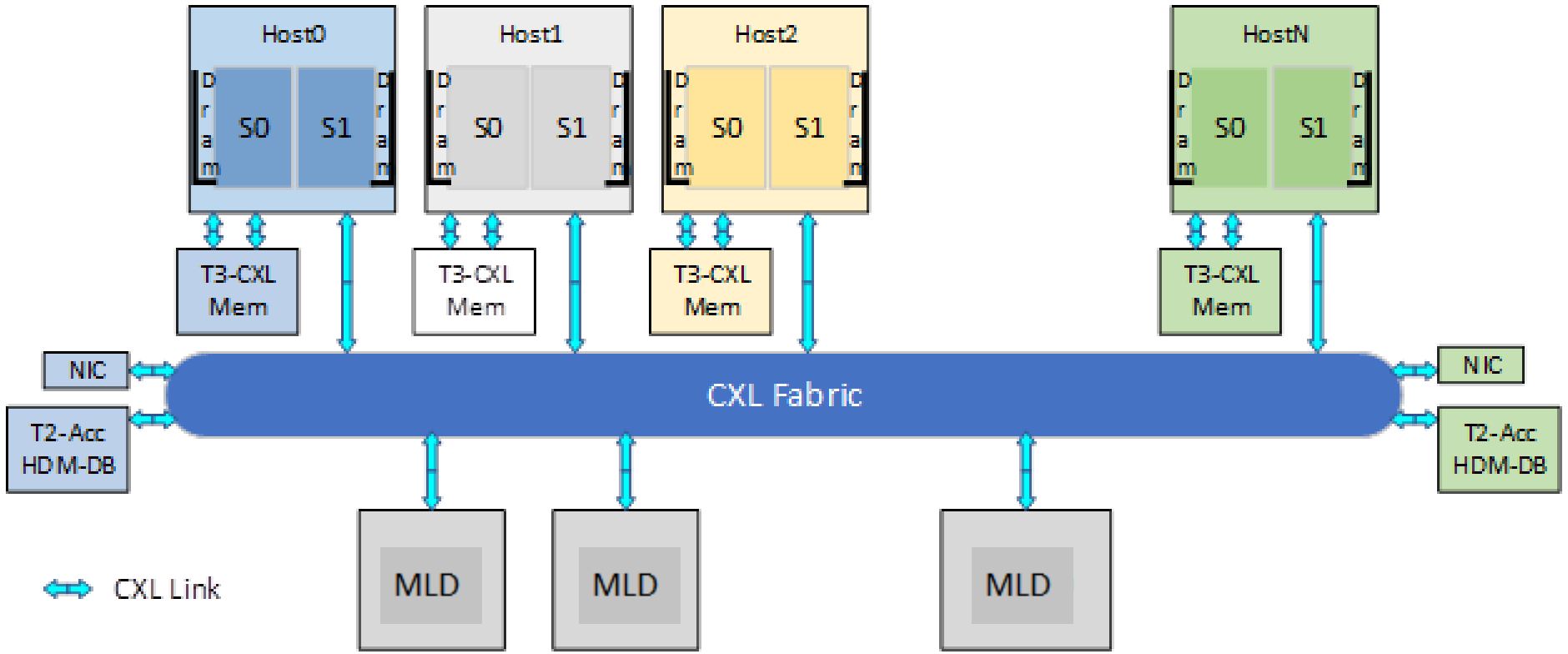
Allocating Memory Types

A host domain may include multiple tiers of memory

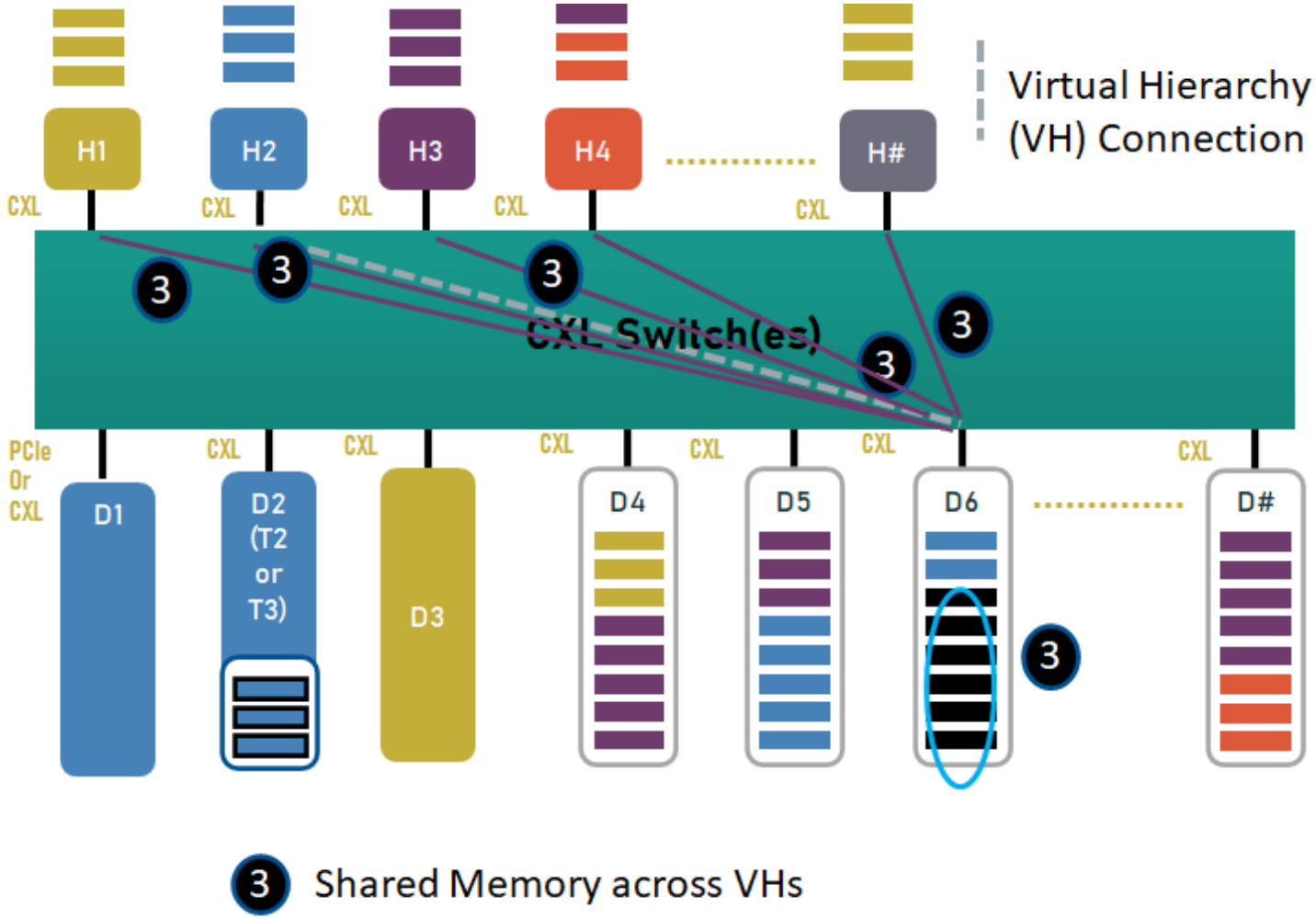
- Memory natively attached to a host (e.g., DDR, HBM, etc.)
- Device memory attached to a host CXL™ link
- Device memory attached to a host through CXL switches

- A CXL fabric may be composed of many host domains and global family of devices which is accessible by all hosts and peer devices within the fabric

CXL™ Fabric Example with Multiple Host Domains and Memory Types



Memory Sharing



.cache and Memory Sharing

- **D**evice coherent management (HDM-**D**)
- **B**ack-invalidate snoop coherence management (HDM-D**B**)