presented by

Compute Express Link™ 2.0 Update

UEFI 2021 Virtual Plugfest

March 30, 2021

Mahesh Natu

Meet the Presenters

Mahesh Natu
Systems and Software WG Co-chair, CXL™ Consortium

Data Center Platform Architect, Intel Corporation

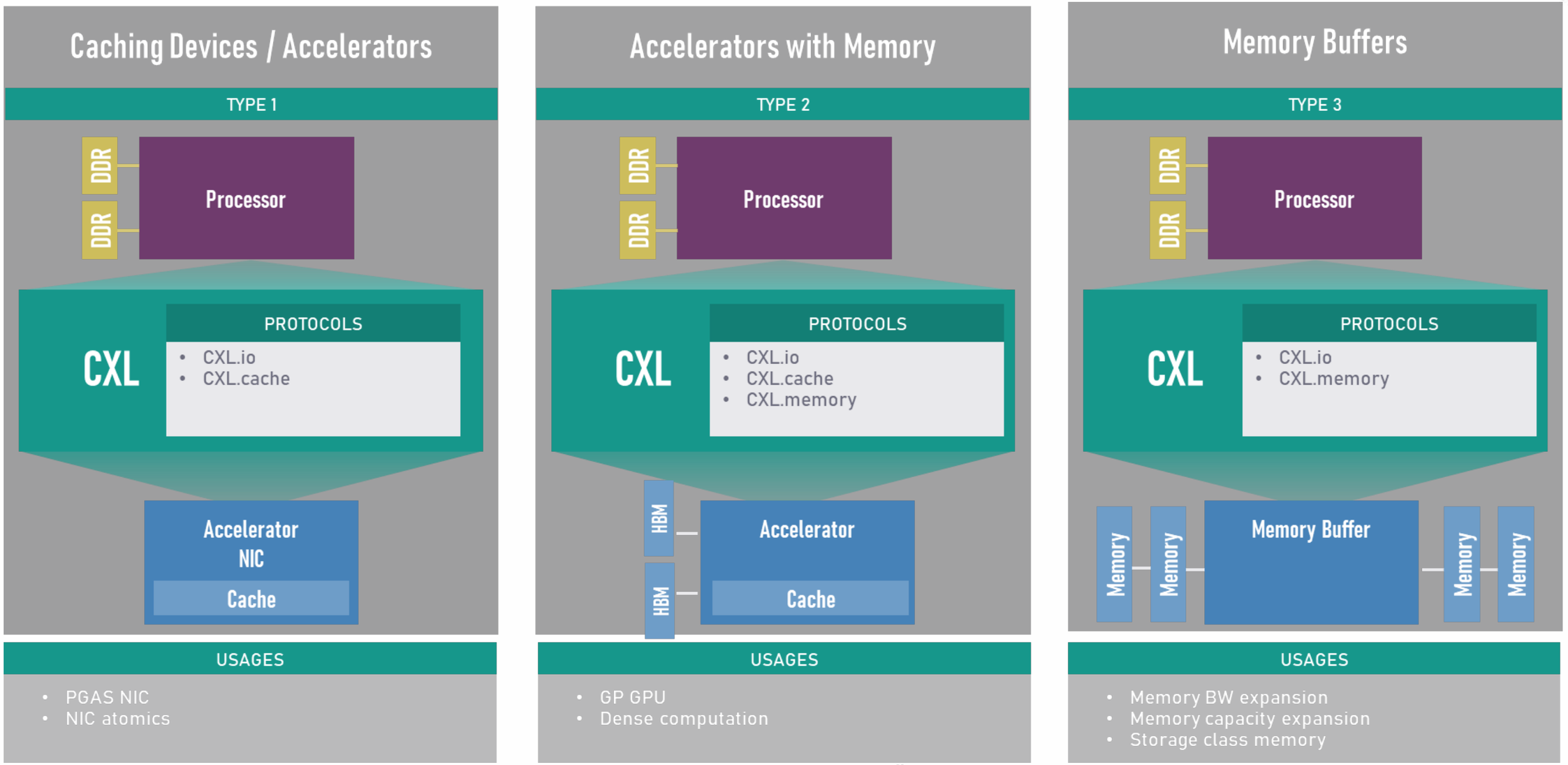
Agenda

- CXL™ Overview
  - CXL 2.0 additions
- CXL Topology Discovery
- Memory Expansion Usage
  - Hetero memory
  - Interleaving
  - Memory Device interface
  - Memory RAS
- Summary and Call to Action

What is CXL?

- Open industry standard for high bandwidth, low-latency interconnect
- Connectivity between host processor and accelerators/ memory device/ smart NIC
- Addresses high-performance computational workloads across AI, ML, HPC, and Comms segments
- Multiplex 3 protocols over PCIe® 5.0 PHY infrastructure
  - CXL.io – I/O semantics, similar to PCIe, mandatory
  - CXL.cache – Caching Semantics, optional
  - CXL.memory – Memory semantics, optional
- CXL spec is at https://www.computeexpresslink.org/download-the-specification

Usage Models

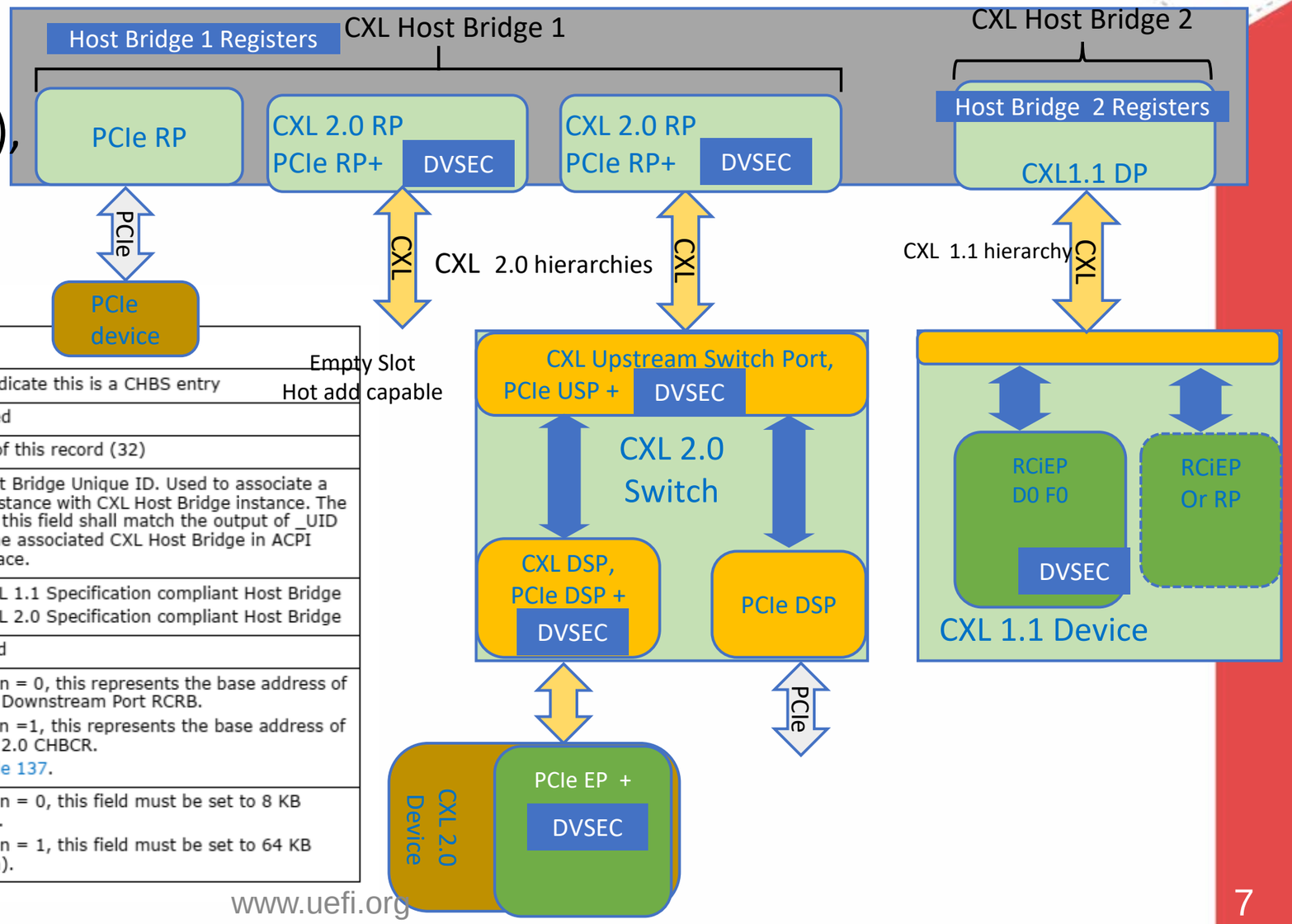


What Does CXL 2.0 Bring to the Table?

Feature	Description
CXL PCIe End-Point	CXL device to be discovered as PCIe Endpoint Support of CXL 1.1 devices directly connected to Root-Port or Downstream Switch Port
Switching	CXL Memory Fan-Out & Pooling with Interleaving CXL.Cache is direct routed between CPU and device with a single caching device within a hierarchy. Downstream port must be capable of being PCIe.
Resource Pooling	Memory Pooling for Type3 device – Multiple Logical Device (MLD), a single device to be pooled across 16 Virtual Hierarchies.
CXL.cachemem enhancements	Persistence (Global Persistence Flush), Managed Hot-Plug, Memory Error Reporting and QoS Telemetry
Security	Authentication, link integrity and Encryption
Software Infrastructure/ API	ACPI & UEFI changes to cover notification and management of CXL Ports and devices Standardized configuration register interface to Memory Devices including PMEM Standardized Memory Error Reporting CXL Switch API for a multi-host or memory pooled CXL switch configuration and management

CXL Discovery Flow – Step 1

- CXL Host Bridges registers can be discovered via CXL Early Discovery Table (CEDT), a new ACPI table.
- Defined in CXL Specification

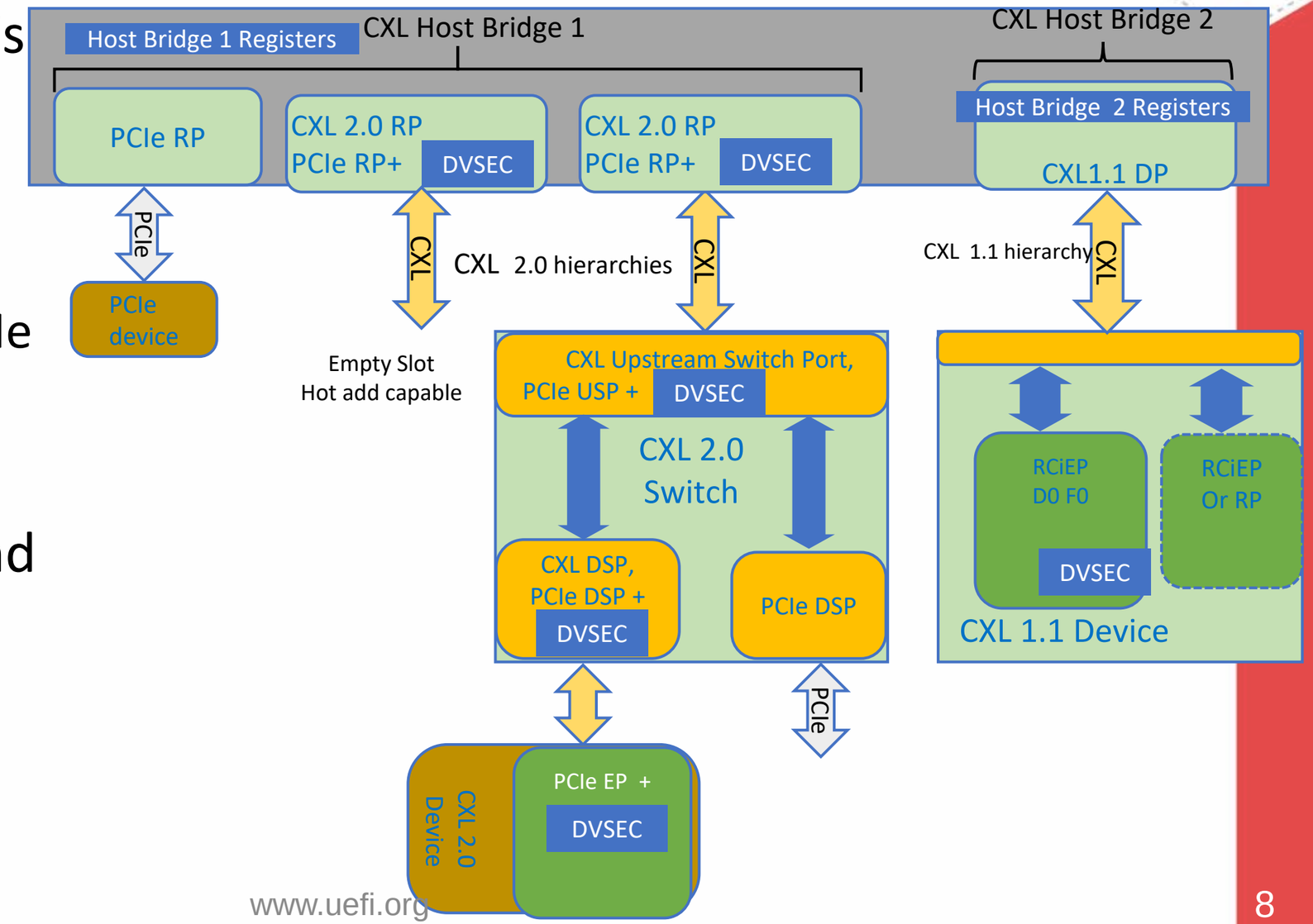


CHBS Structure

Field	Byte Length	Offset	
Type	1	0	=0 to indicate this is a CHBS entry
Reserved	1	1	`Reserved
Record Length	2	2	Length of this record (32)
UID	4	4	CXL Host Bridge Unique ID. Used to associate a CHBS instance with CXL Host Bridge instance. The value of this field shall match the output of _UID under the associated CXL Host Bridge in ACPI namespace.
CXL Version	4	8	00h: CXL 1.1 Specification compliant Host Bridge 01h: CXL 2.0 Specification compliant Host Bridge
Reserved	4	12	Reserved
Base	8	16	If Version = 0, this represents the base address of CXL 1.1 Downstream Port RCRB. If version =1, this represents the base address of the CXL 2.0 CHBCR. See Table 137.
Length	8	24	If Version = 0, this field must be set to 8 KB (2000h). If Version = 1, this field must be set to 64 KB (10000h).

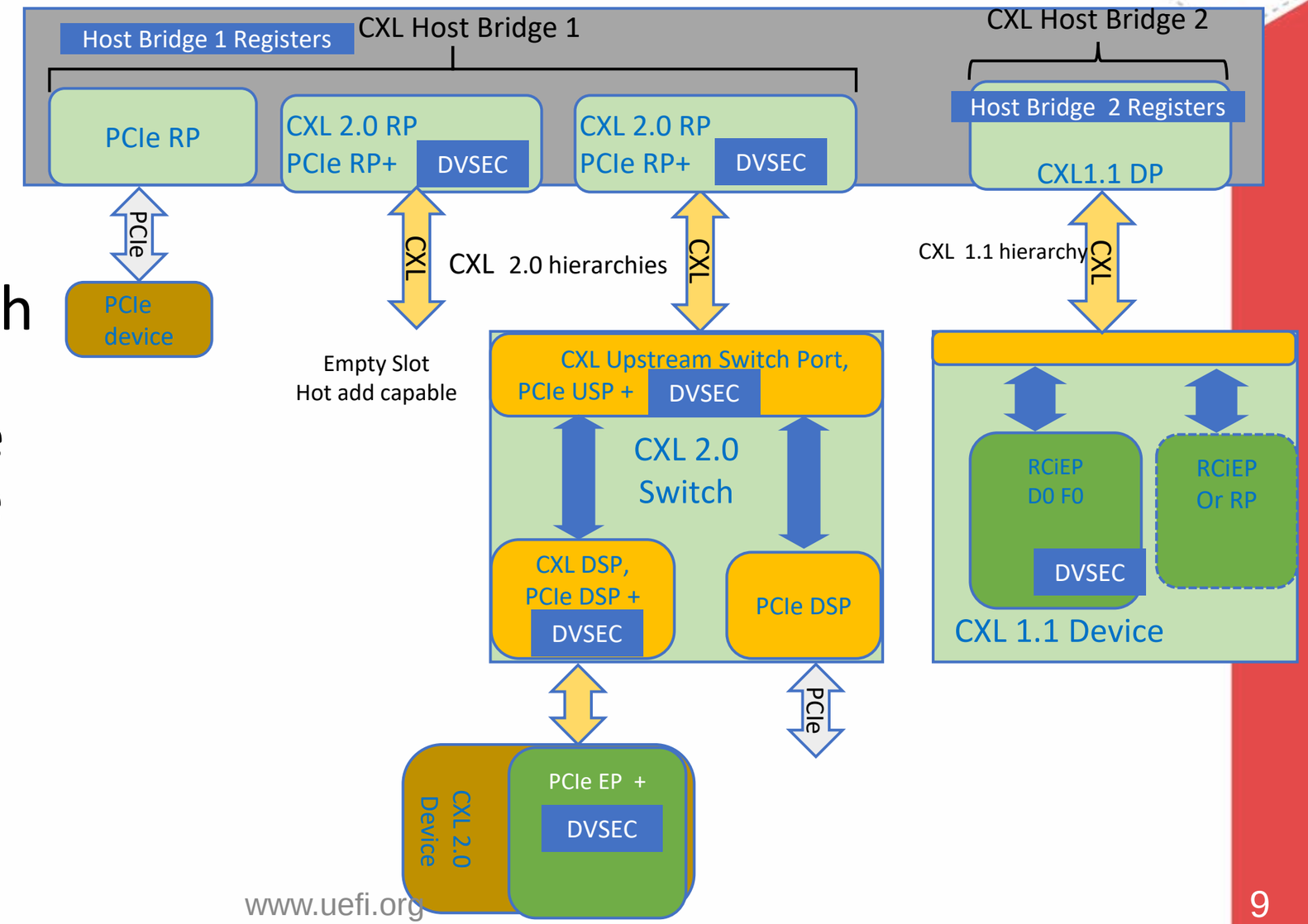
CXL Discovery Flow – Step 2

- Next level of discovery is based on ACPI Namespace
  - CXL HB Hardware ID="ACPI0016"
  - Compatibility ID of PCIe Host Bridge to enable enumeration by non-CXL enabled OSs
- Underneath, SW will find standard PCIe objects like _BBN, _CRS, PCIe _OSC and new objects like CXL _OSC



CXL Discovery Flow – Step 3

- The last leg of discovery uses standard PCIe enumeration
- CXL Root Ports, Switch Ports and Devices appear like their PCIe counterparts, but are decorated with CXL spec defined register blocks called DVSEC.



Namespace example

```
Device(CXL0)
{
    Name(_HID,EISAID("ACPI0016"))// CXL Host Bridge
    Name(_CID, Package(2){   EISAID(“PNP0A03),  EISAID("PNP0A08")}) // compatible with PCIe and PCI Host Bridges
    Name(_UID, 0)                    // Instance 0 of CXL HB, cross-reference with CEDT table entry to get HB register base
    Name(_BBN, ..)                   // Same as PCIe, enables PCIe SW to enumerate the tree
    Name(_CRS, ..)                   // Same as PCIe, enables PCIe SW to discover CXL.io resource assignment
   Method(_OSC,4)
    // CXL _OSC, identified by CXL GUID
     // PCIe OSC, identified by PCIe GUID, PCIe SW can enumerate
..
} // End CXL0
Device(CXL1)
{
   Name(_HID,EISAID("ACPI0016"))// CXL Host Bridge
    Name(_CID, Package(2){   EISAID(“PNP0A03),  EISAID("PNP0A08")}) // compatible with PCIe and PCI Host Bridges
    Name(_UID, 0)                    // Instance 1 of CXL HB, cross-reference with CEDT table entry to get HB register base
..
} // End CXL1
```

CXL _OSC

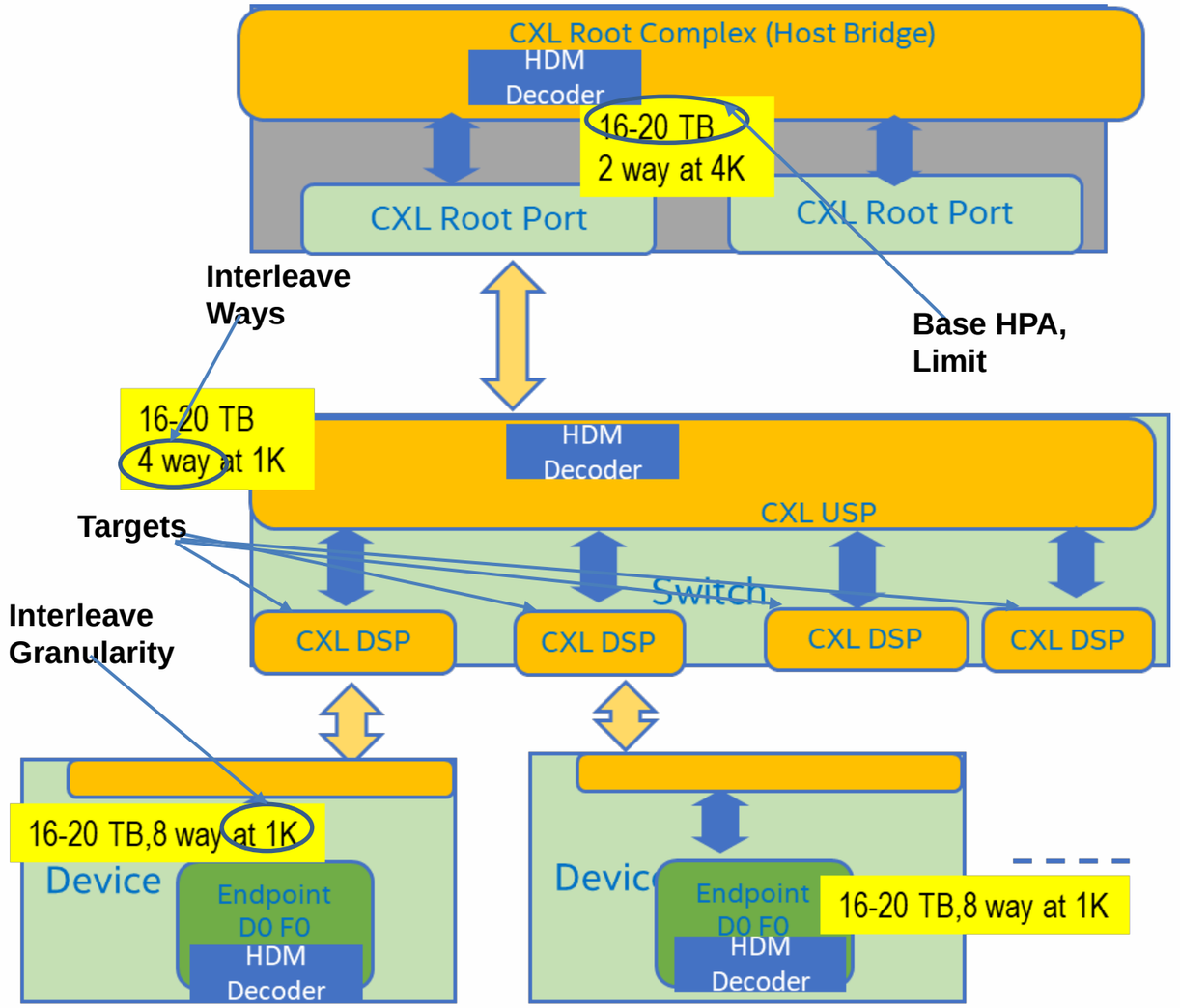
- PCIe relies on ACPI _OSC method to keep OS and Firmware in sync regarding management of PCIe capabilities
- CXL _OSC extends PCIe _OSC to cover CXL features
- Identified by new UUID
- Defined as a superset of PCIe _OSC
- _OSC Capability Buffer
  - First DWORD = Generic to _OSC, follows ACPI specification
  - 2nd DWORD = PCIe Support Field as defined by PCI Firmware Specification.
  - 3rd DWORD = PCIe Control Field as defined by PCI Firmware Specification (In/Out)
  - 4th DWORD = CXL Support Field, defined in CXL 2.0 Specification
  - 5th DWORD = CXL Control Field , defined in CXL 2.0 Specification (In/Out)
- If CXL _OSC is present, CXL aware OS evaluates it and ignores PCIe _OSC

Hetero Memory Attributes

- CXL systems are heterogenous by nature
- SRAT and HMAT models work great when system firmware has apriori knowledge of coherent components and the config is relatively static.
- CXL breaks both assumptions
  - Open ecosystem, fully PnP architecture with hot-plug support
- However, OS/VMMs still need SRAT and HMAT equivalent information to allocate memory optimally ..
- Coherent Device Attribute Table is the answer
  - Each coherent device reports its local latency/BW characteristics via DOE interface. This data structure is called CDAT.
  - System Firmware or OS/VMM stiches together CDAT obtained from each component to construct SRAT/HMAT equivalent structures.

CXL Memory Interleaving

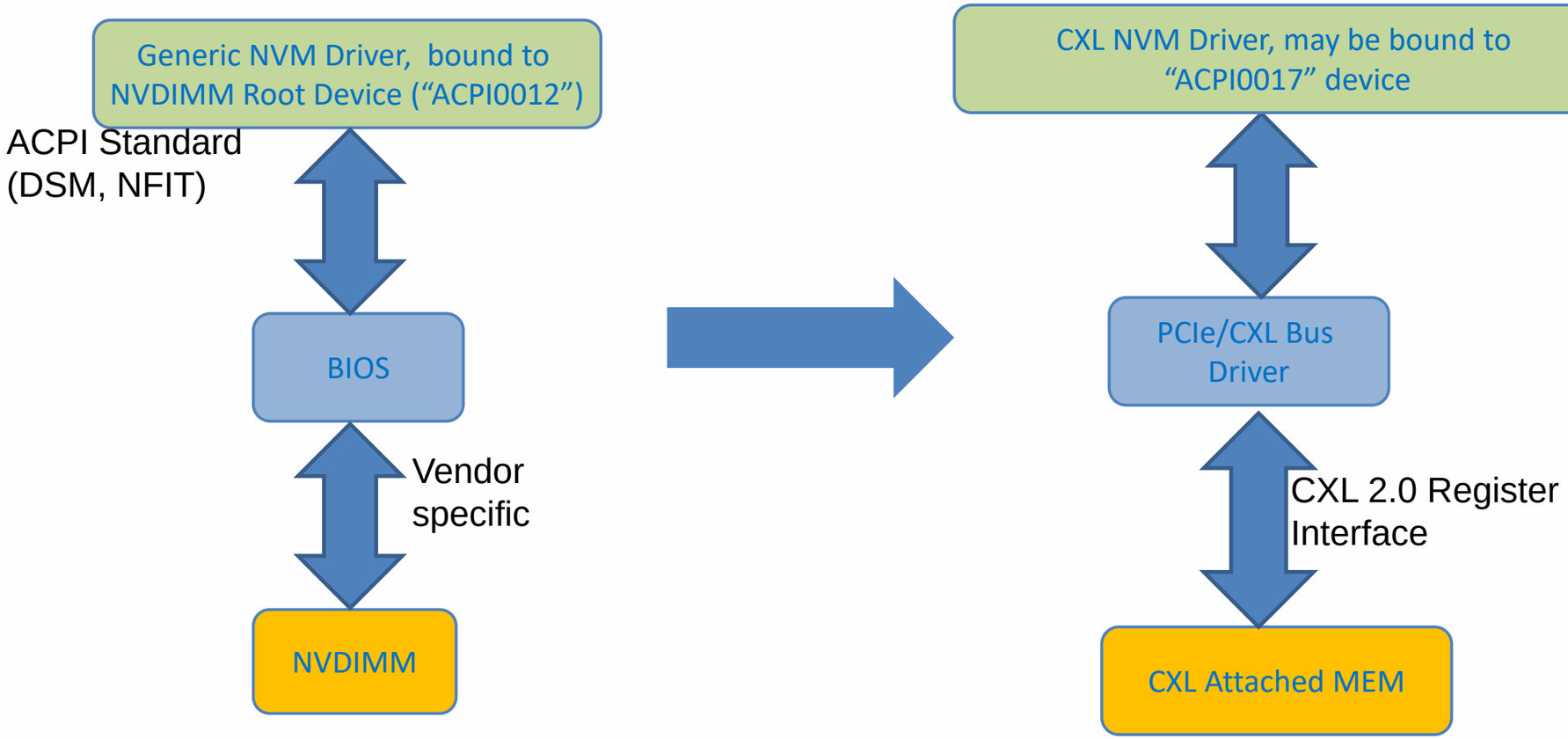
- CXL 2.0 memory devices may be interleaved for performance reasons
- An Interleave Set is identified by
  - Base HPA, Limit - Multiples of 256 MB
  - Interleave Way – 2, 4 or 8
  - Interleave Granularity - 2** (8,9,10, 11, 12, 13,14)
  - Targets (applicable to RC and USP)
- Configured via HDM Decoder registers in USP, RC and Device
  - USP and RC use these decoders to pick the target
  - Device uses these decoders to translate Host Physical Address (HPA) to Device Physical Address (DPA)
- Can be configured by UEFI or OS
  - OS: Lazy config or hot-plug



Memory Device Configuration Interface

- Type 3 devices, especially persistent memory devices, rely on system software for provisioning and management
- CXL 2.0 introduces a standard register interface for managing CXL attached memory devices
- Enables OSs to carry vendor-agnostic memory device driver
- Architecture Elements
  - Defined as number of discoverable Capabilities
  - Capabilities includes Device Status and standard mailboxes, accessed via MMIO registers
  - Standardized mailbox commands
  - Standardized Error log formats
  - Accommodates Firmware First Error Handling Model
- CXL does not have to rely on NFIT and NVDIMM _DSM methods

Comparison with NVDIMM Model



No impact to applications that consume memory

CXL 1.1 Memory Error Reporting

- CXL 1.1 Baseline
  - CXL device is expected to be self-sufficient and not rely on CPU for its RAS functionality
  - Any demand access to CXL.mem that results in uncorrected error will return poison
  - Non-demand errors (e.g. memory scrubber on the device) reported to device driver
- Limitations of above approach
  - Industry is interested in gaining more visibility into memory errors
  - Vendor specific driver is burdensome for memory vendors and OS vendors/distros
  - No firmware first support

CXL 2.0 Memory Error Reporting Improvements

- Standardized access to error logs
  - New register interface that enables host to access memory error records from the device directly
  - Standard error log structure, superset of Machine check banks
  - Multiple error queues in the device, one per error severity
  - Common interface for volatile and non-volatile memory
  - Allows software to keep track of correctable errors, poison generation, memory scrub detected errors etc. and take corrective actions
- Standardized signaling
  - OS signaling via standard MSI/MSI-X
  - Support for Firmware First and new CPER record type
  - Per severity masking
  - Leaky buckets for corrected errors

Summary and Call to Action

- CXL Consortium momentum continues to grow
  - 130+ members and growing
- CXL 2.0 introduces new features & usage models
  - Switching, pooling, persistent memory support, security
  - Fully backward compatible with CXL 1.1 and 1.0
  - Built in Compliance & Interop program
  - UEFI 2.9, ACPI 6.4 and CXL 2.0 specification comprehend CXL related UEFI/ACPI changes
- Call to action
  - Help drive CXL enhancements into UEFI and ACPI specifications
  - Get your firmware and software CXL ready
  - Join CXL Consortium

Questions?

Thanks for attending the UEFI 2021 Virtual Plugfest

For more information on UEFI Forum and UEFI Specifications, visit http://www.uefi.org

presented by