SNIA PERSISTENT MEMORY + SUMMIT 2021 COMPUTATIONAL STORAGE

FROM DATACENTER TO EDGE : VIRTUAL EVENT
APRIL 21-22, 2021

Persistent Memory in CXL

Andy Rudoff, PMem SW Architect
Chet Douglas, Principal Engineer
Tiffany Kasanicky, Principal Engineer

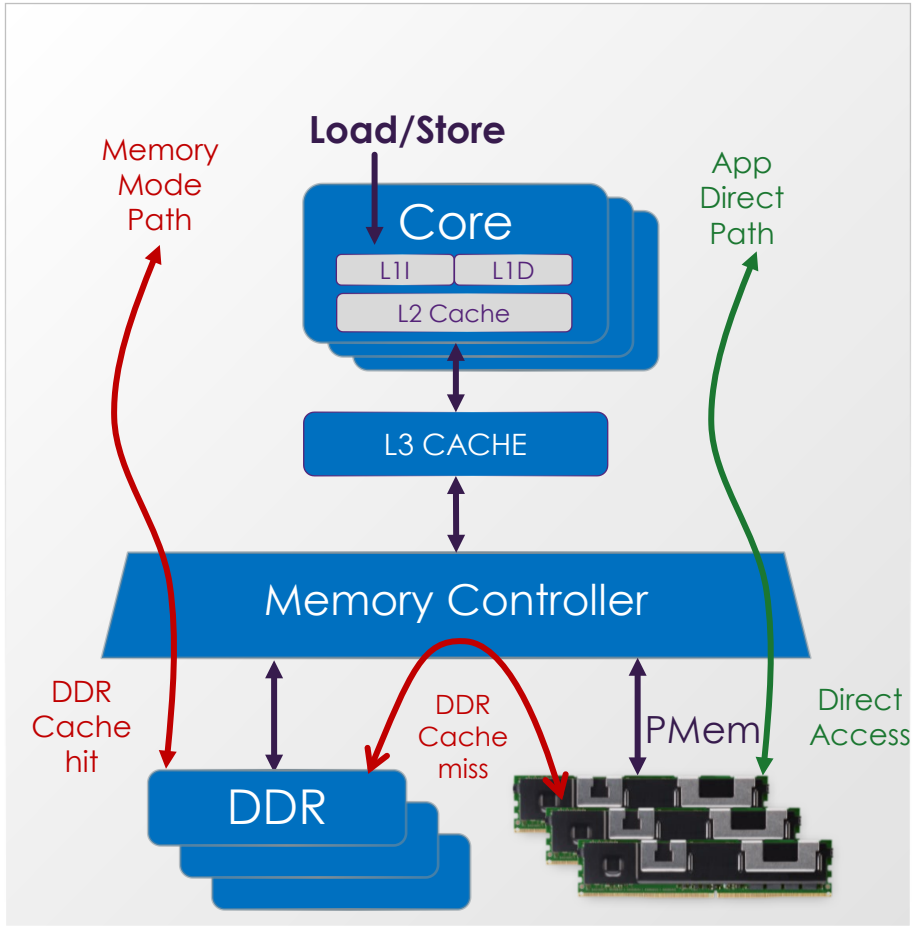
Intel Corporation

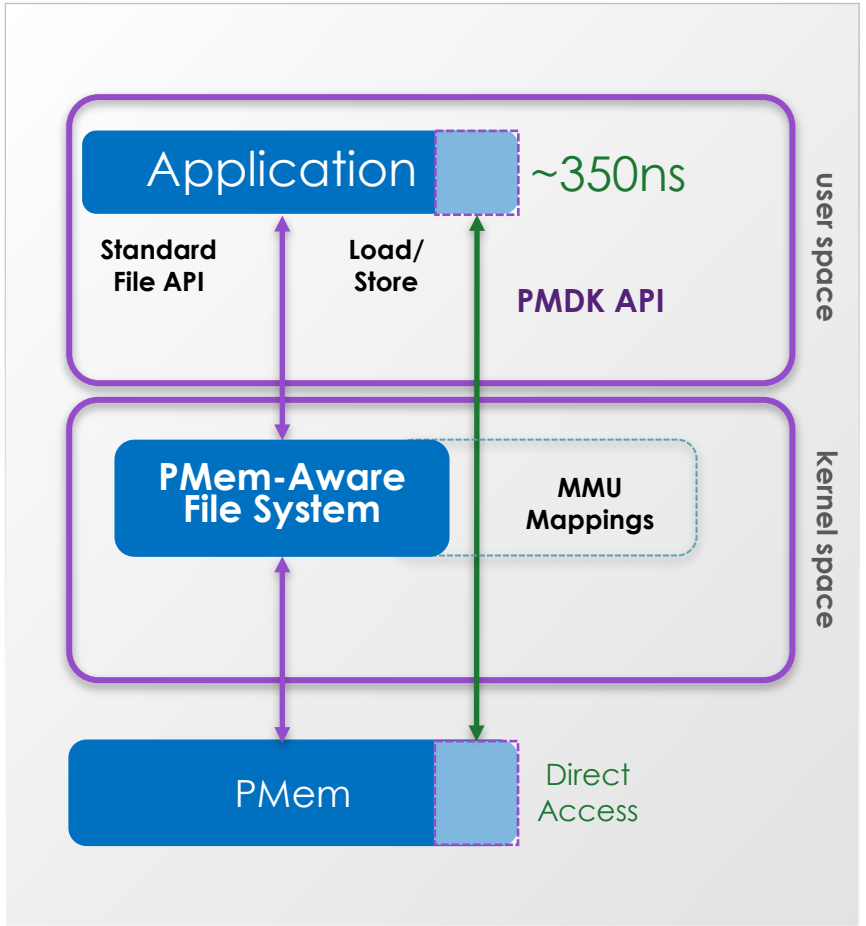
Persistent Memory Today

Connecting to the Memory Bus

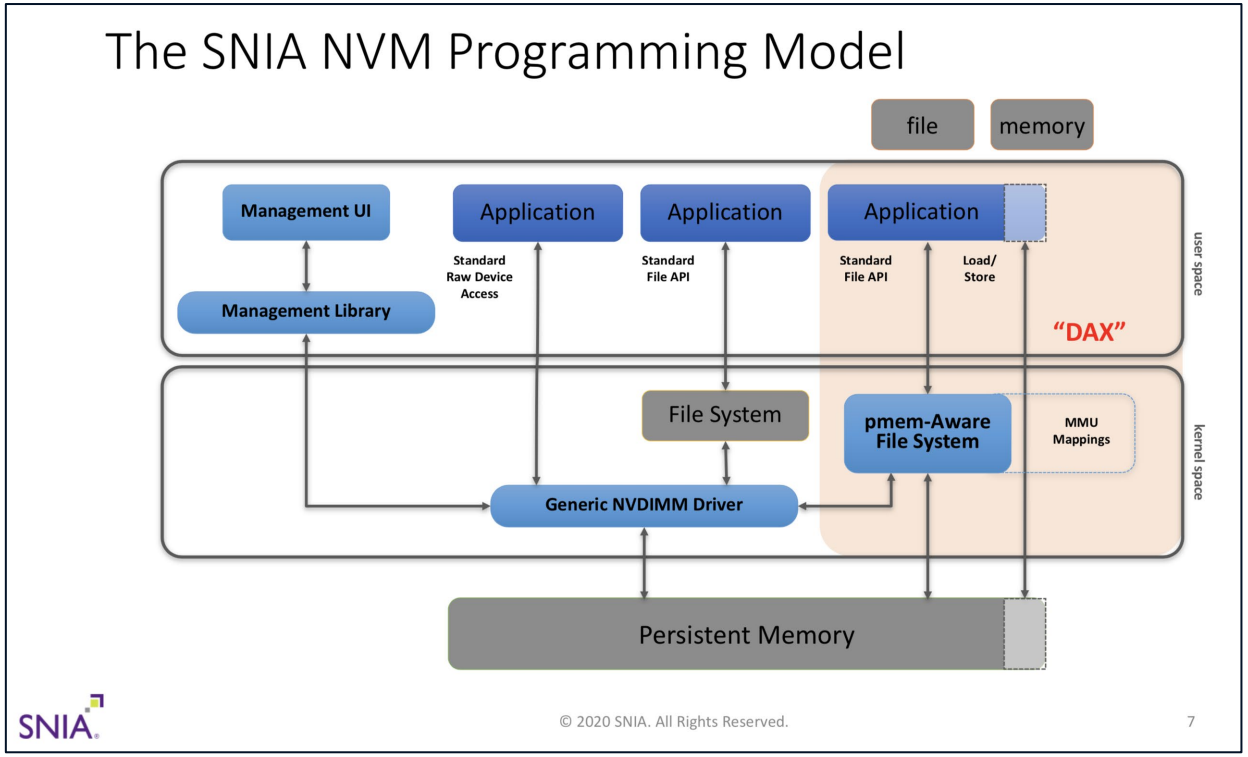
Connecting to the Memory Bus

Intel's Approach for Optane PMem

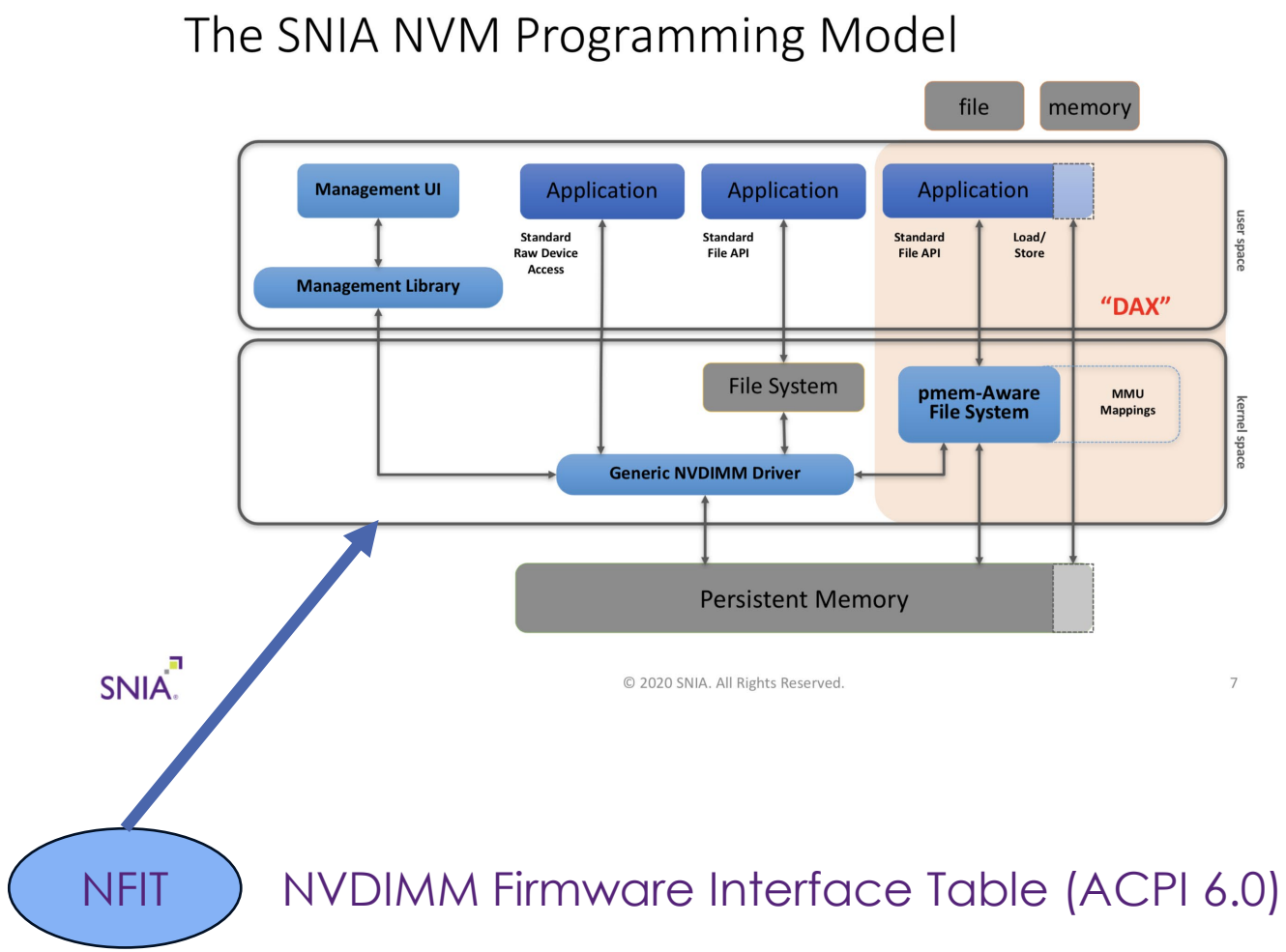




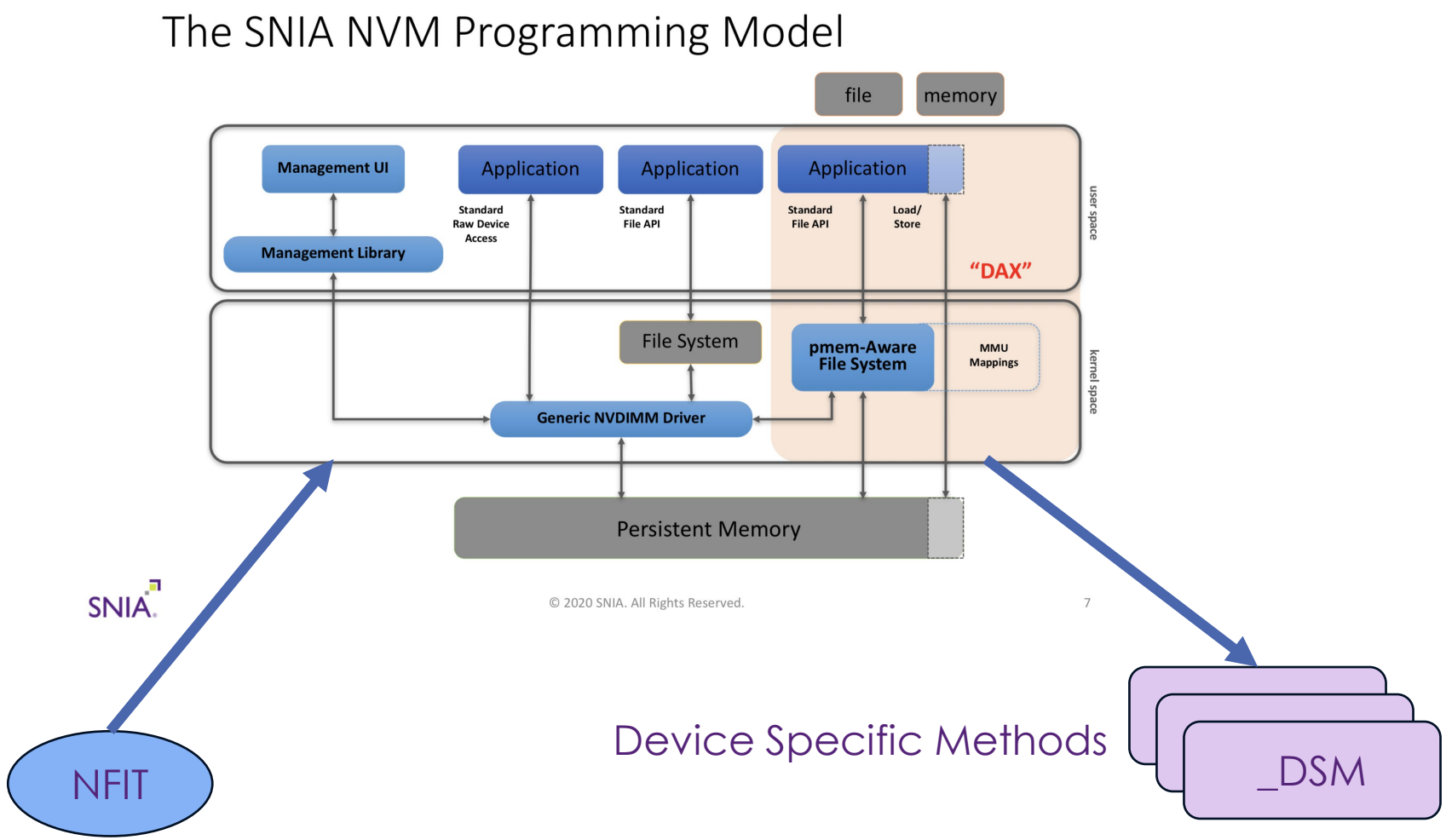
The ACPI "NVDIMM" Framework



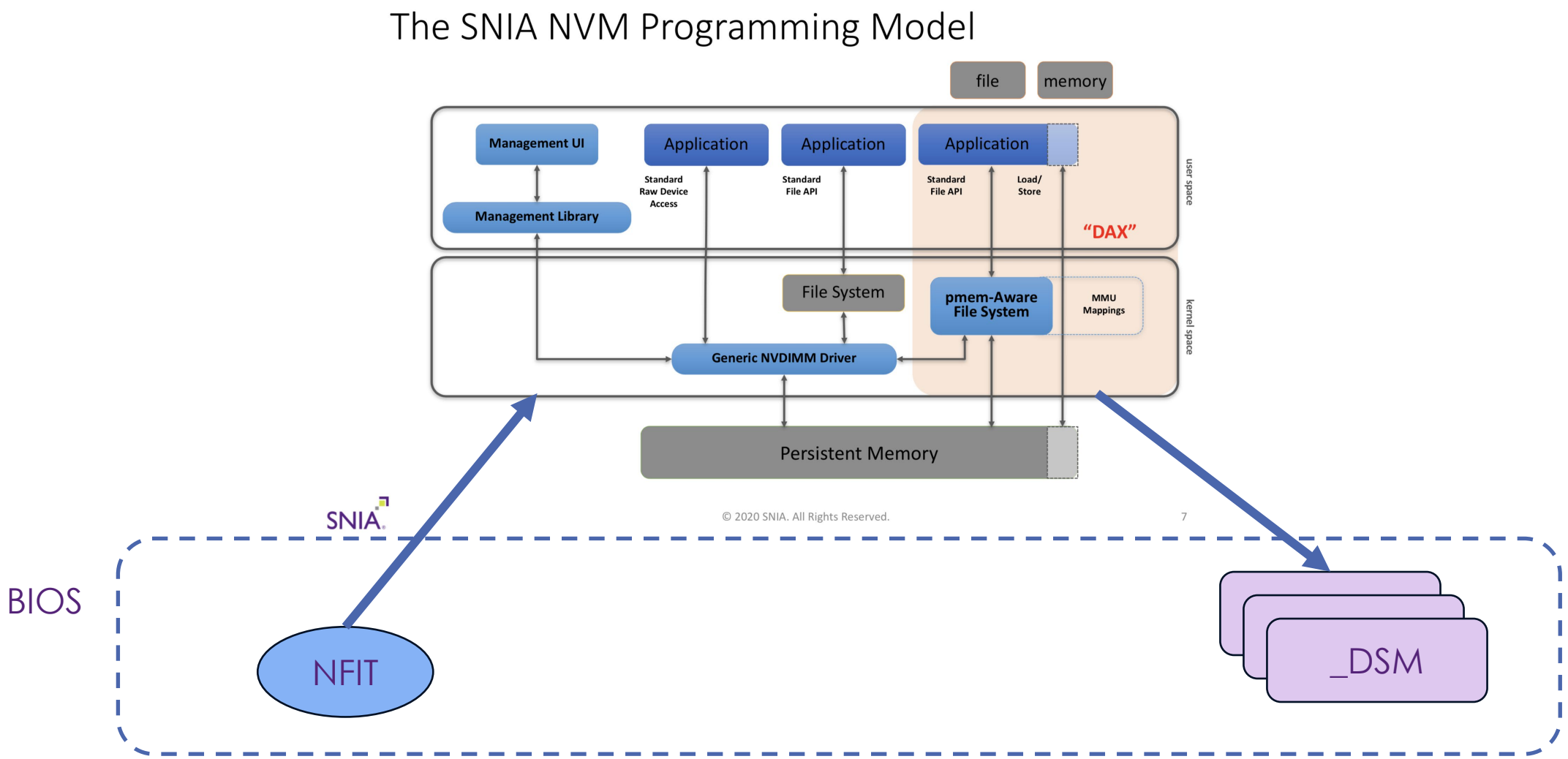
The ACPI "NVDIMM" Framework



The ACPI "NVDIMM" Framework



The ACPI "NVDIMM" Framework



Learnings from ACPI Based Approach

Pros

- ACPI NFIT Unified NVDIMM-N and Intel's Optane PMem
  - Helped get enabling upstream early
- _DSMs allowed a generic kernel implementation
  - Differences abstracted away by _DSMs
- Mechanism has evolved gracefully
  - Fairly small additions, few errata

Cons

- ACPI dynamic support doesn't scale
  - Hot plug challenging
  - Meant for small number of empty sockets
- _DSM complexity hard to maintain
  - Bug fixes, additions logistical challenges
  - Virtually impossible to support multiple devices
  - No generic BIOS

Compute Express Link

A New Class of Interconnect

Compute Express Link

- CXL 1.1
  - June 2019
- CXL 2.0
  - Nov 2020
  - 1.1 Compatible
  - Adds pmem
- computeexpresslink.org

Introducing CXL

CXL Compute Express Link

- 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
  - Heterogeneous processing: scalar, vector, matrix, spatial architectures spanning CPU, GPU, FPGA
  - Memory device connectivity
  - PCIe PHY completely leveraged with additional latency optimization
  - Dynamic multiplexing of 3 protocols
- Based on PCIe® 5.0 PHY infrastructure
  - Leverages channel, retimers, PHY, Logical, Protocols
  - CXL.io – I/O semantics, similar to PCIe - mandatory
  - CXL.cache – Caching Semantics – optional
  - CXL.memory – Memory semantics - optional

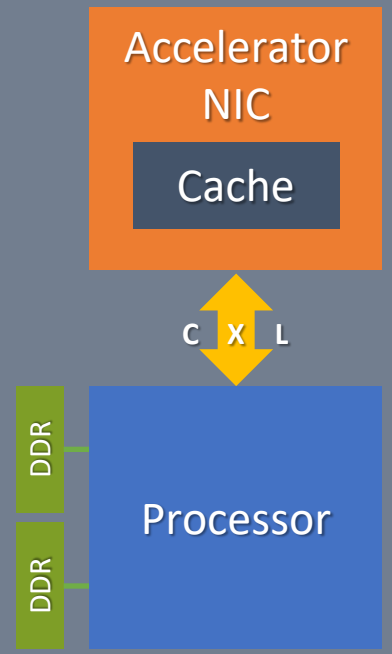
Our Focus

Representative CXL Usages

Caching Devices / Accelerators

Usages:
- PGAS NIC
- NIC atomics

Protocols:
- CXL.io
- CXL.cache

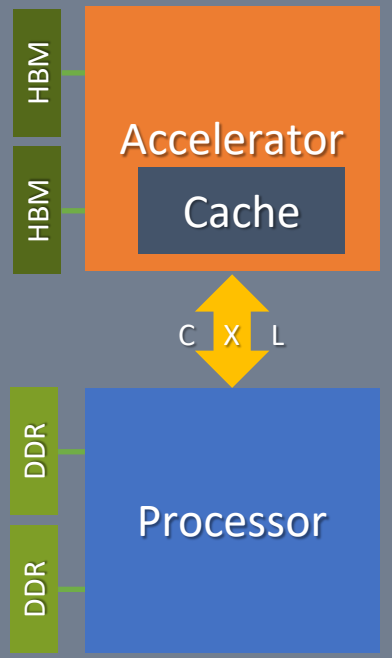


(Type 1 Device)

Accelerators with Memory

Usages:
- GPU
- FPGA
- Dense Computation

Protocols:
- CXL.io
- CXL.cache
- CXL.memory

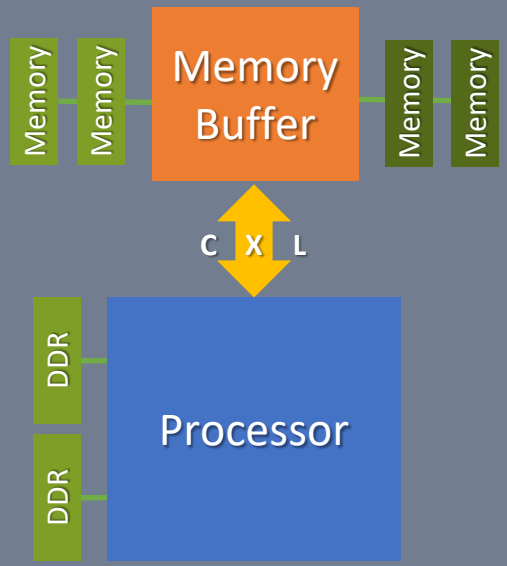


(Type 2 Device)

Memory Buffers

Usages:
- Memory BW expansion
- Memory capacity expansion
- 2LM

Protocols:
- CXL.io
- CXL.mem



(Type 3 Device)

Where PMem Fits In

Caching Devices / Accelerators

Usages:
- PGAS NIC
- NIC atomics

Protocols:
- CXL.io
- CXL.cache

(Type 1 Device)

Accelerators with Memory

Usages:
- GPU
- FPGA
- Dense Computation

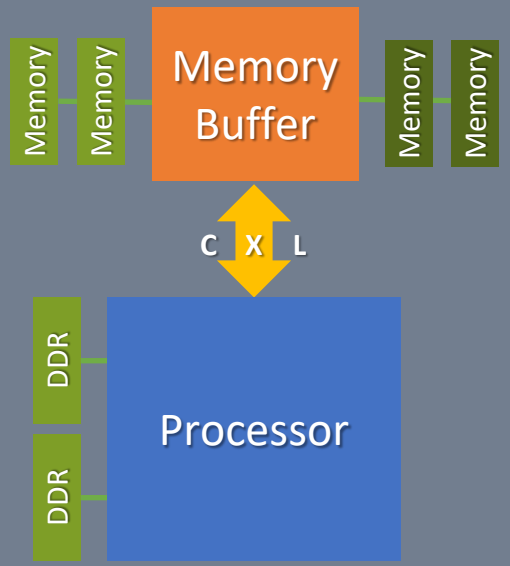
Protocols:
- CXL.io
- CXL.cache
- CXL.memory

(Type 2 Device)

Memory Buffers

Usages:
- Memory BW expansion
- Memory capacity expansion
- 2LM

Protocols:
- CXL.io
- CXL.mem



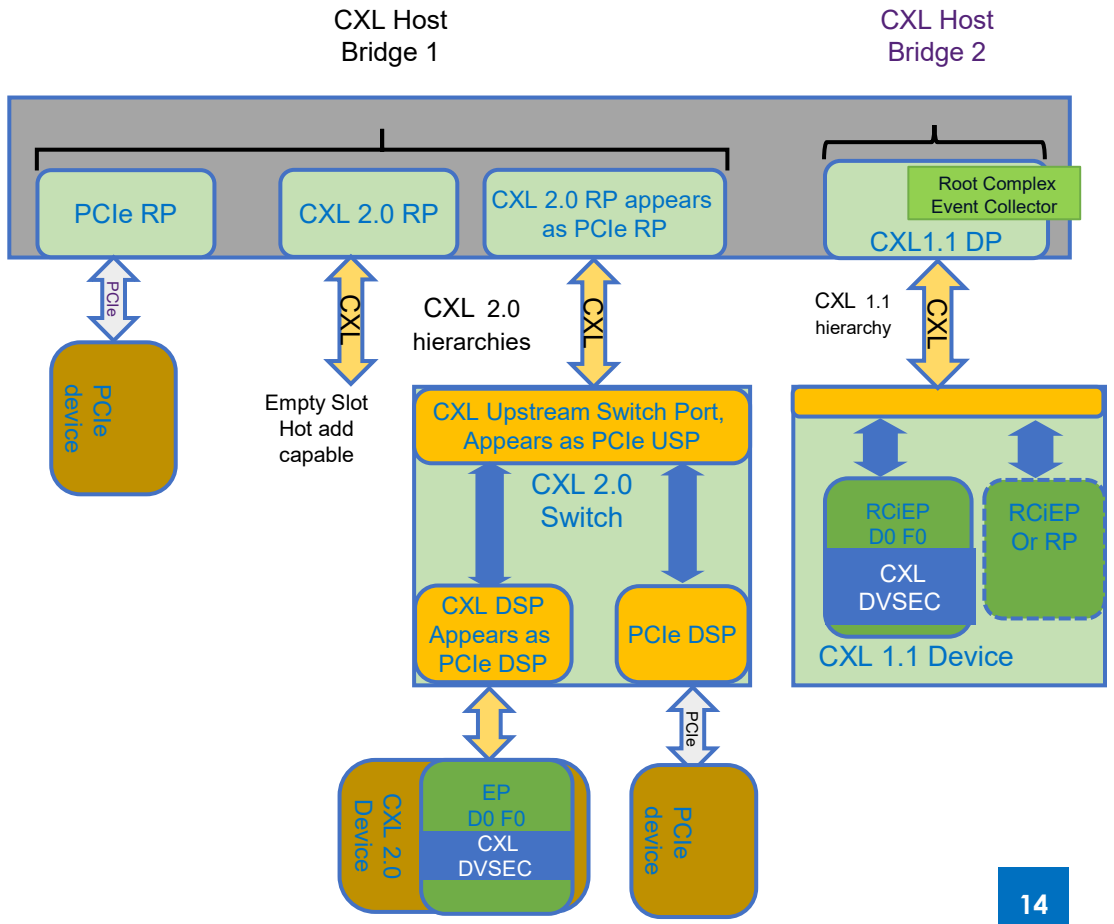
(Type 3 Device)

Adding PMem to CXL

The CXL 2.0 Specification

CXL 2.0 Changes for PMem

- Most changes should apply to all memory types
  - Minimize PMem-specific changes, rest apply to volatile memory too
- PCIe enumeration
  - NFIT isn't used for CXL devices (they aren't NVDIMMs!)
  - Leverage PCIe frameworks, including hot plug
- MMIO registers
  - Mailbox interface, etc.
- Command Interface
  - Was vendor-private for NVDIMMs
- Driver Writers Guide (delivered as separate doc)
- Minor changes to external specs like ACPI/UEFI



Mailbox Commands

- Was vendor private
- Standards are a double-edged sword
  - Generic Drivers
  - Committee visit for every change
- Learnings from NVDIMMs helped
  - Leverage what worked
  - Fix pain points
- Some commands apply to all CXL →

CXL Device Command Opcodes

Opcode					Required*	Input Payload Size (B)	Output Payload Size (B)
Command Set Bits[15:8]		Command Bits[7:0]		Combined Opcode			
01h	Events	00h	Get Event Records (Section 8.2.9.1.2)	0100h	M	1	20h+
		01h	Clear Event Records (Section 8.2.9.1.3)	0101h	M	8+	0
		02h	Get Event Interrupt Policy (Section 8.2.9.1.4)	0102h	M	0	4
		03h	Set Event Interrupt Policy (Section 8.2.9.1.5)	0103h	M	4	0
02h	Firmware Update	00h	Get FW Info (Section 8.2.9.2.1)	0200h	O	0	50h
		01h	Transfer FW (Section 8.2.9.2.2)	0201h	O	80h+	0
		02h	Activate FW (Section 8.2.9.2.3)	0202h	O	2	0
03h	Timestamp	00h	Get Timestamp (Section 8.2.9.3.1)	0300h	O	0	8
		01h	Set Timestamp (Section 8.2.9.3.2)	0301h	O	8	0
04h	Logs	00h	Get Supported Logs (Section 8.2.9.4.1)	0400h	M	0	8+
		01h	Get Log (Section 8.2.9.4.2)	0401h	M	18h	0+

Memory Device Commands

- Most added commands →
- _DSMs are gone
  - OS uses mailbox directly
  - BIOS too
- Much complexity moved:
  - From BIOS to OS
- Allows generic:
  - BIOS
  - OS Drivers

CXL Memory Device Command Opcodes

Opcode					Required	Input Payload Size (B)	Output Payload Size (B)
Command Set Bits[15:8]		Command Bits[7:0]		Combined Opcode			
40h	Identify	00h	Identify Memory Device (Section 8.2.9.5.1.1)	4000h	M	0	43h
41h	Capacity Config and Label Storage	00h	Get Partition Info (Section 8.2.9.5.2.1)	4100h	O	0	20h
		01h	Set Partition Info (Section 8.2.9.5.2.2)	4101h	O	0Ah	0
		02h	Get LSA (Section 8.2.9.5.2.3)	4102h	PM	8	0+
		03h	Set LSA (Section 8.2.9.5.2.4)	4103h	PM	8+	0
42h	Health Info and Alerts	00h	Get Health Info (Section 8.2.9.5.3.1)	4200h	M	0	12h
		01h	Get Alert Configuration (Section 8.2.9.5.3.2)	4201h	M	0	10h
		02h	Set Alert Configuration (Section 8.2.9.5.3.3)	4202h	M	0Ch	0
		03h	Get Shutdown State (Section 8.2.9.5.3.4)	4203h	PM	0	1
		04h	Set Shutdown State (Section 8.2.9.5.3.5)	4204h	PM	1	0
43h	Media and Poison Mgmt	00h	Get Poison List (Section 8.2.9.5.4.1)	4300h	PM	10h	20h+
		01h	Inject Poison (Section 8.2.9.5.4.2)	4301h	O	8	0
		02h	Clear Poison (Section 8.2.9.5.4.3)	4302h	O	48h	0
		03h	Get Scan Media Capabilities (Section 8.2.9.5.4.4)	4303h	PM	10h	4
		04h	Scan Media (Section 8.2.9.5.4.5)	4304h	PM	11h	0
		05h	Get Scan Media Results (Section 8.2.9.5.4.6)	4305h	PM	0	20h+
44h	Sanitize	00h	Sanitize (Section 8.2.9.5.5.1)	4400h	O	0	0
		01h	Secure Erase (Section 8.2.9.5.5.2)	4401h	O	0	0

Mandatory

Mandatory for PMem

...not the full table

Example: Identify Memory Device

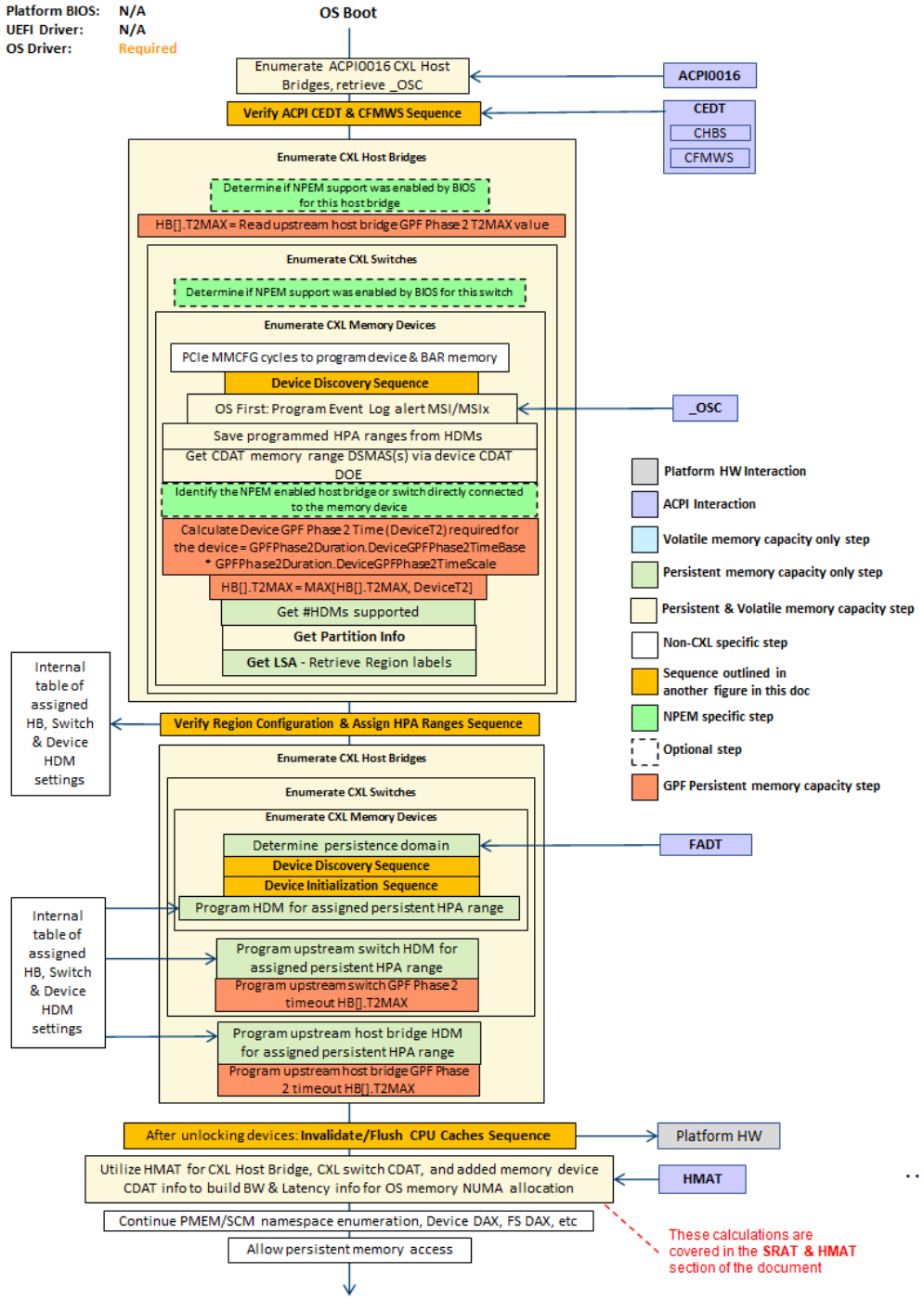
Identify Memory Device Output Payload

Byte Offset	Length	Description
0	10h	**FW Revision:** Contains the revision of the active FW formatted as an ASCII string. This is the same information that may be retrieved with the Get FW Info command.
10h	8	**Total Capacity:** This field indicates the total usable capacity of the device. Expressed in multiples of 256 MB. Total device usable capacity is divided between volatile only capacity, persistent only capacity, and capacity that can be either volatile or persistent. Total Capacity shall be greater than or equal to the sum of Volatile Only Capacity and Persistent Only Capacity.
18h	8	**Volatile Only Capacity:** This field indicates the total usable capacity of the device that may only be used as volatile memory. Expressed in multiples of 256 MB.
20h	8	**Persistent Only Capacity:** This field indicates the total usable capacity of the device that may only be used as persistent memory. Expressed in multiples of 256 MB.
28h	8	**Partition Alignment**: If the device has capacity that may be used either as volatile memory or persistent memory, this field indicates the partition alignment size. Expressed in multiples of 256 MB. Partitionable capacity is equal to Total Capacity - Volatile Only Capacity - Persistent Only Capacity. If 0, the device doesn't support partitioning the capacity into both volatile and persistent capacity.
		Informational Event Log Size: The number of events the device can

...not the full table

Driver Writers Guide

- CXL 2.0 Spec defines commands
  - Can be terse, spec language
- Platforms decide some of the details
  - Example: BIOS on Intel platforms
- Document flows, algorithms
- Published on intel.com



…see the Guide for details

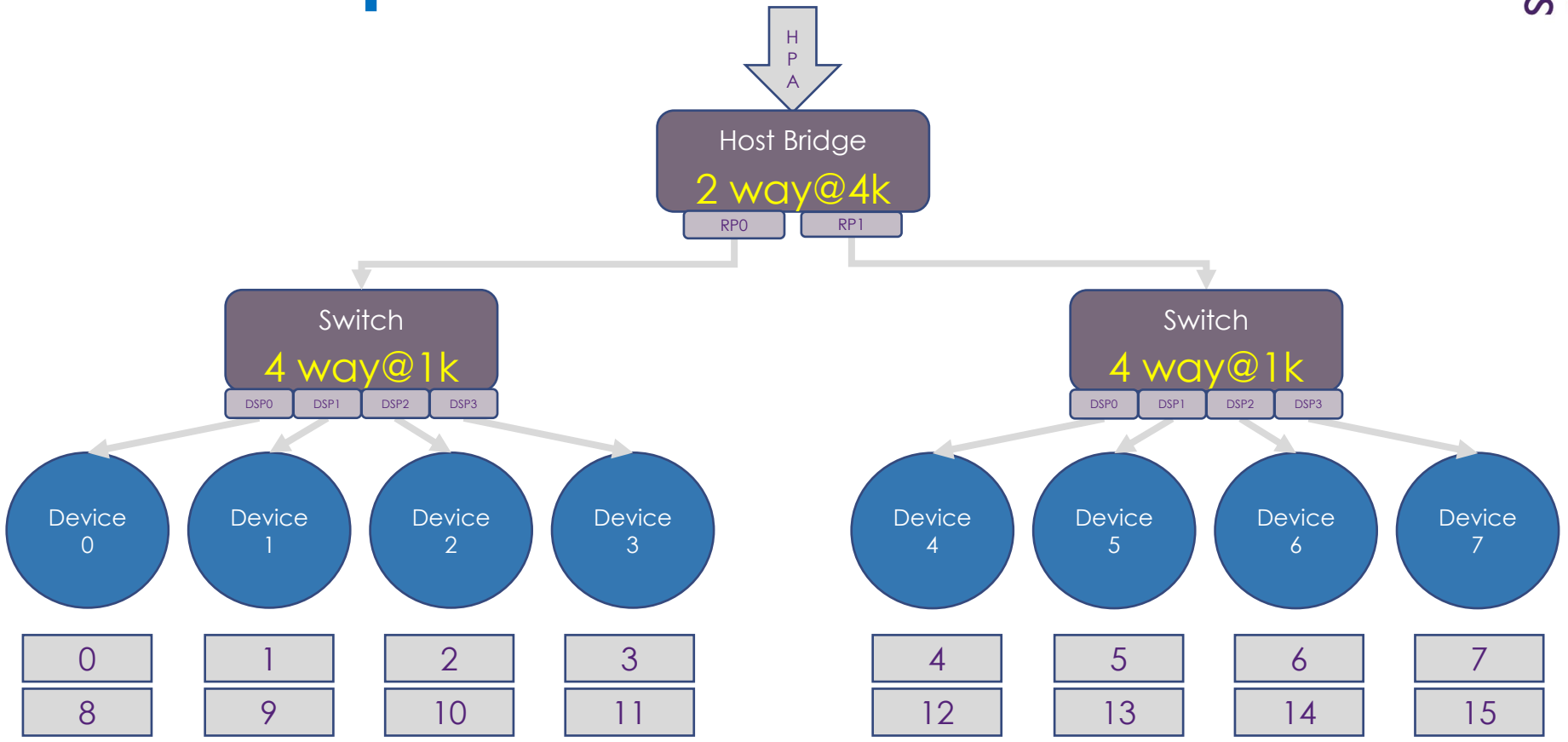
The Hard Stuff

Where we spent much of our time

Interleaving

- HDM Decoders
  - Allow interleaving across devices
  - New to PCIe: interleave sets
- Important concept for PMem
  - For volatile memory, changing the interleave may impact performance
  - For PMem, changing the interleave loses your data
- Label Storage Area
  - Defined in CXL 2.0 spec
  - Provides region (interleave set) and namespace configuration

Label Example

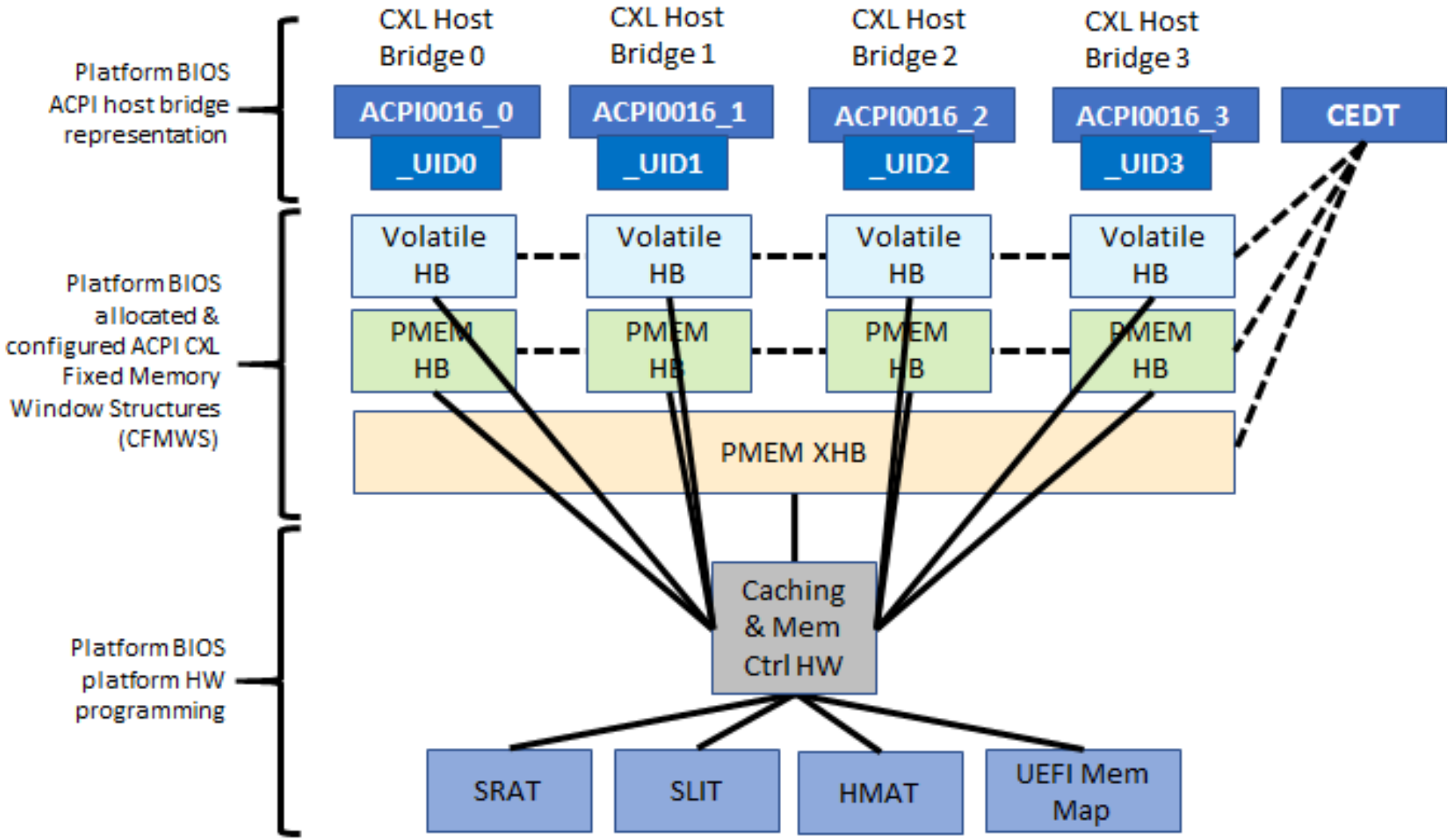


Label		Label		Label		Label	
N	8	N	8	N	8	N	8
Pos	0	Pos	1	Pos	2	Pos	3
IG	1k	IG	1k	IG	1k	IG	1k

Label		Label		Label		Label	
N	8	N	8	N	8	N	8
Pos	4	Pos	5	Pos	6	Pos	7
IG	1k	IG	1k	IG	1k	IG	1k

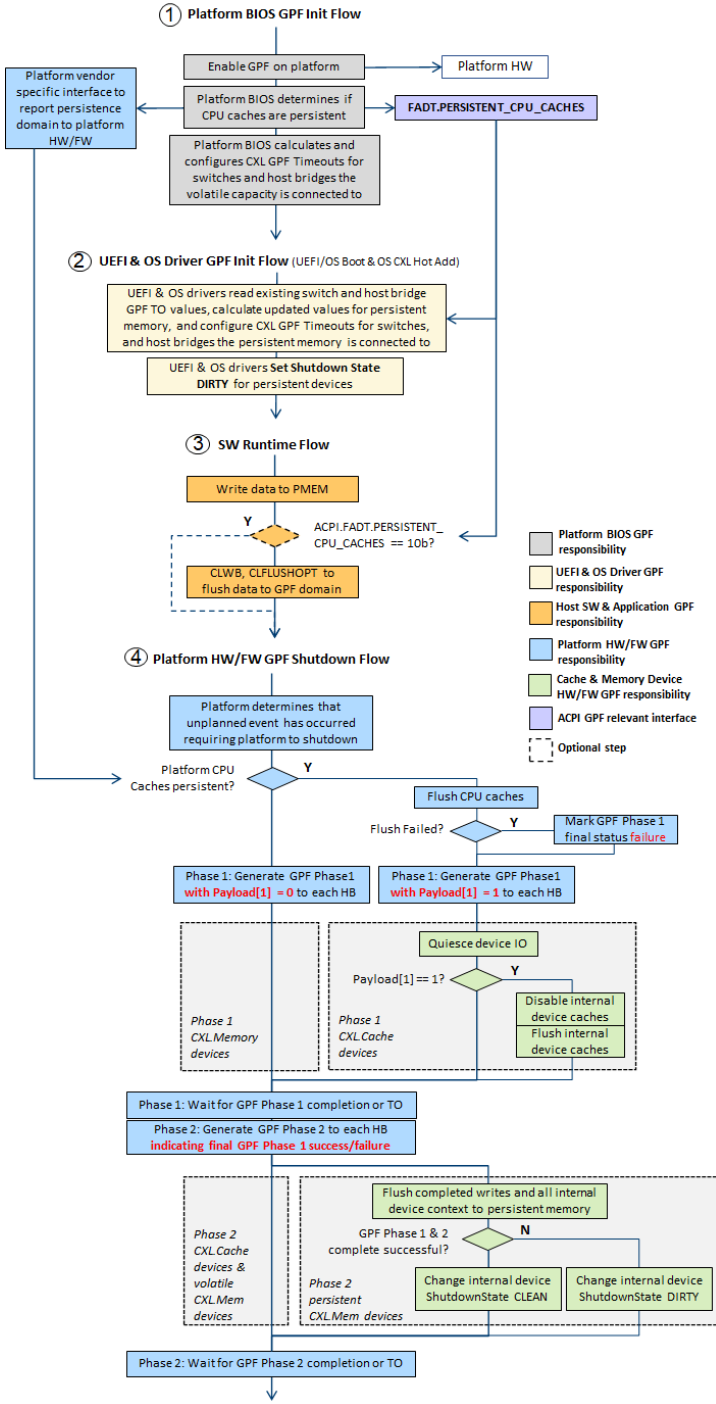
Hot Plug

- Handled by OS
  - No BIOS in flow
- Uses “windows”
  - BIOS provided
- Flow used for PMem
  - Except boot dev



Flush-on-fail To Persistence

- Global Persistent Flush (GPF)
  - Analogous to ADR or eADR with NVDIMMs
- Simple when it works
  - Application just relies on it
- More complex when it fails
  - Dirty Shutdown Count
  - Already part of the programming model
  - Code written for NVDIMM still works correctly
    - Provided the model was followed correctly



...see the Guide for details

Software Enabling

Current State and Future Work

The Good News

- Momentum around CXL is Huge
  - NVDIMM enabling was a few of us in a room, updating ACPI
  - CXL is dozens of highly-engaged companies, including all major OSVs
- Preliminary generic Type 3 CXL Driver already upstream in Linux
  - Orchestrated by NVDIMM framework maintainer Dan Williams
- QEMU patches emulating CXL Type 3 devices posted
  - Written by Ben Widawsky, provided a development platform for the driver
- Cross-company, cross committee collaboration on specifications
  - Prevents messy collisions from different implementation decisions

The Good News

- Momentum around CXL is Huge
  - NVDIMM enabling was a few of us in a room, updating ACPI
  - CXL is dozens of highly-engaged companies, including all major OSVs
- Preliminary generic Type 3 CXL Driver already upstream in Linux
  - Orchestrated by NVDIMM framework maintainer Dan Williams
- QEMU patches emulating CXL Type 3 devices posted
  - Written by Ben Widawsky, provided a development platform for the driver
- Cross-company, cross committee collaboration on specifications
  - Prevents messy collisions from different implementation decisions
- Generic BIOS code and OS drivers!

The Challenges

- Memory bus, ACPI, NFIT was essentially done
  - OS versions that support NVDIMMs, continue to work across generations
- CXL requires a new set of drivers
  - Drivers do new things, like group into interleave sets
  - Not clear if drivers can be added to older kernels
- NVDIMM tooling → CXL tooling
  - Example: ndctl command on Linux begats cxl command

Summary

- The programming model remains the same
  - Applications written to the SNIA programming model continue to work
- CXL offers:
  - Moving PMem off the memory bus
  - Scalability (all types of memory)
  - Flexibility
- PMem on CXL specified as of CXL 2.0, published last November
  - OS enabling is emerging

Thank you

Please visit www.snia.org/pm-summit for presentations