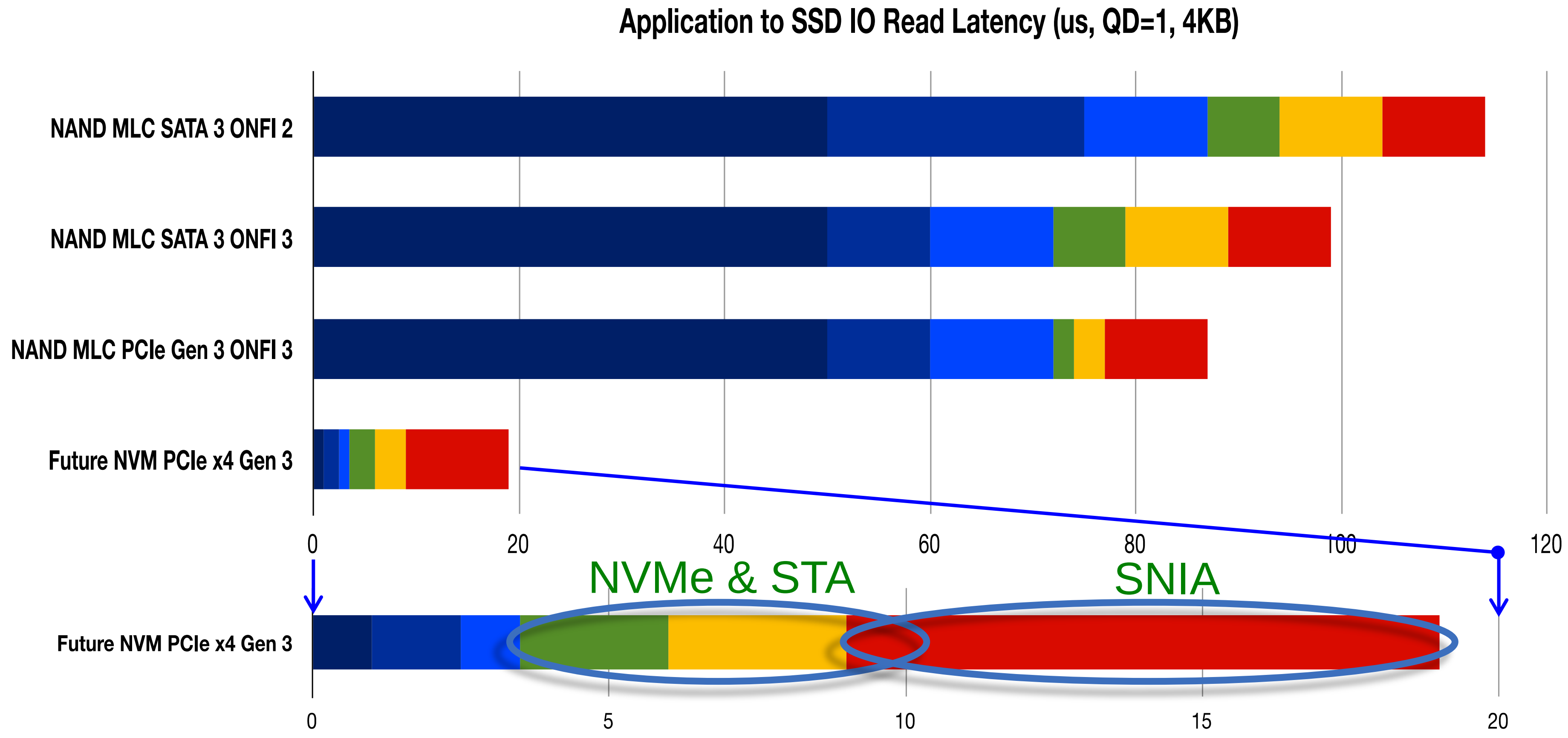
The SNIA NVM Programming Model

#OFADevWorkshop

Opportunities with Next Generation NVM



NVM Express/SCSI Express: Optimized storage interconnect & driver
SNIA NVM Programming TWG: Optimized system & application software

From Jim Pappas, Intel, SNIA NVM Summit

SNIA NVM Programming Model Version 1

- Approved by SNIA in December 2014
  - Downloadable by anyone
- Expose new features of block and file to applications
  - Atomicity capability and granularity
  - Thin provisioning management
- Use of memory mapped files for persistent memory
  - Existing abstraction that can act as a bridge to higher value from persistent memory
  - Limits the scope of re-invention from an application point of view
  - Open source implementations already available for incremental innovation (e.g. PMFS)
- Programming Model, not API
  - Describes behaviors in terms of actions
  - Facilitates discovery of capabilities using attributes
  - Usage illustrated as Use Cases
  - Implementations map actions and attributes to API elements

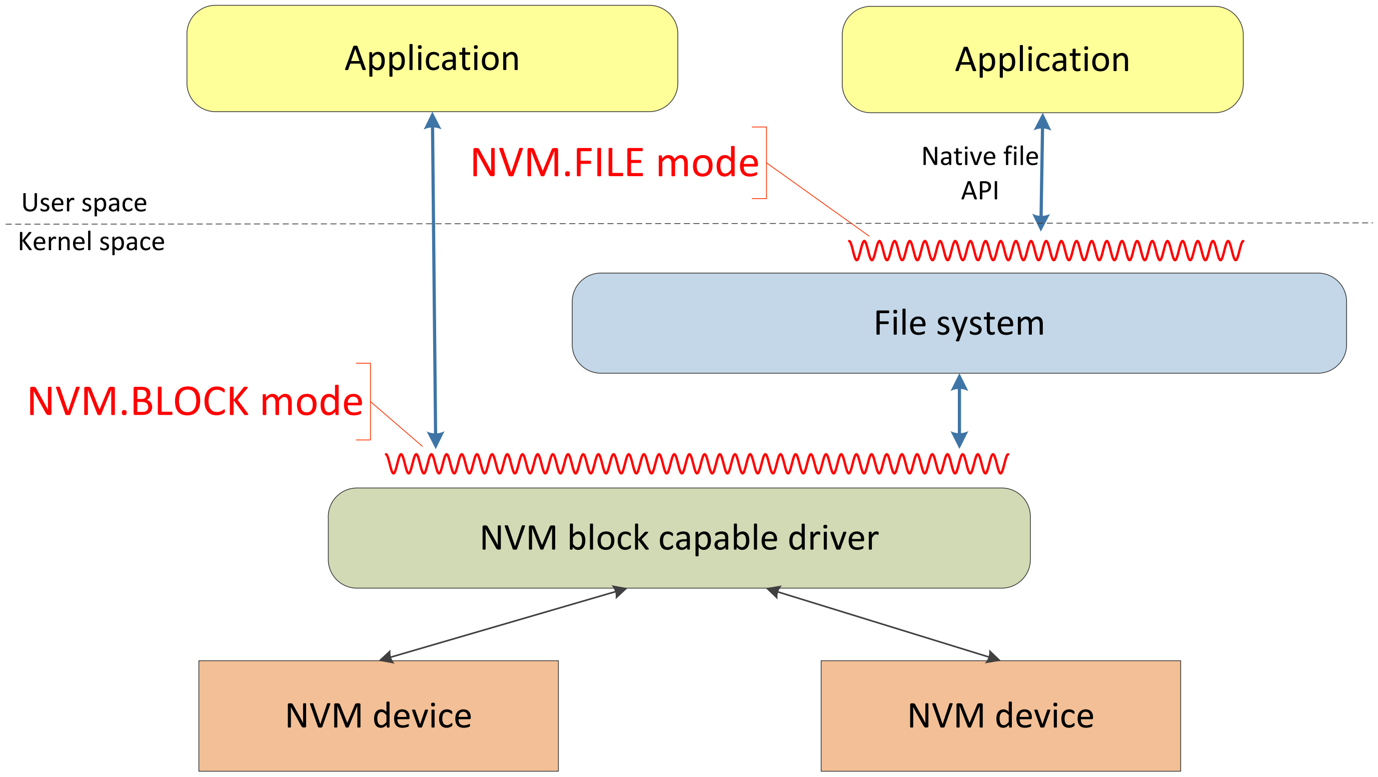
Conventional Block and File Modes

BLOCK mode describes extensions:

- Atomic write features
- Granularities (length, alignment)
- Thin Provisioning Management

FILE mode describes extensions:

- Discovery and use of atomic write features
- The discovery of granularities (length, alignment characteristics)



Memory Mapping in NVM.FILE mode uses volatile pages and writes them to disk or SSD

Persistent Memory (PM)

- Is
  - Byte addressable from programmer's point of view
  - Load/Store access
  - Memory-like performance (stalls CPU loads)
  - DMA-able including RDMA
- Is Not
  - Prone to unexpected latencies
    - Demand paging
    - Page Cache
  - Durable until data is flushed
- Think Battery Backed RAM

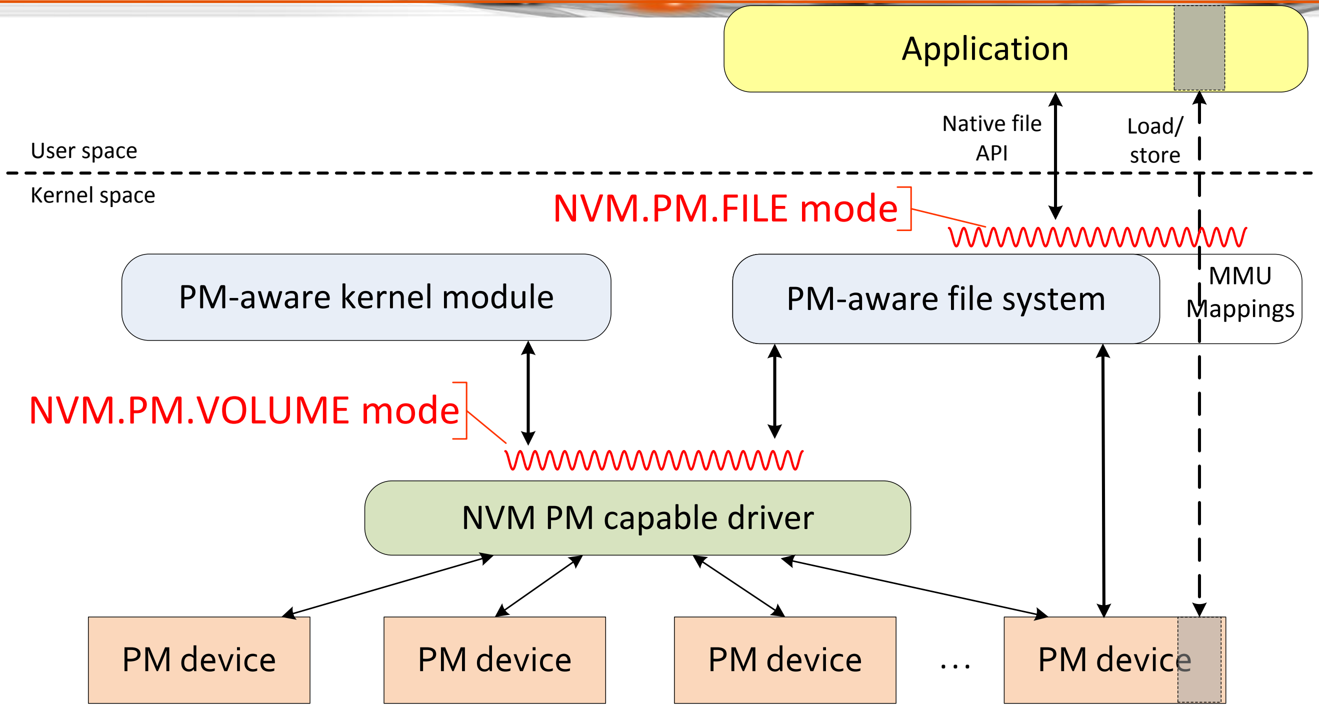
Persistent Memory Modes

NVM.PM.VOLUME mode provides a software abstraction to OS components for Persistent Memory (PM) hardware:

- List of physical address ranges for each PM volume
- Thin provisioning management

NVM.PM.FILE mode describes the behavior for applications accessing persistent memory including:

- mapping PM files (or subsets of files) to virtual memory addresses
- syncing portions of PM files to the persistence domain



Memory Mapping in NVM.PM.FILE mode enables direct access to persistent memory using CPU instructions

Expected usage of PM modes

- Uses for NVM.PM.VOLUME
  - Kernel modules
  - PM aware file systems
  - Storage stack components
- Uses for NVM.PM.File
  - Applications
    - Persistent datasets, directly addressable, no DRAM footprint
    - Persistent caches (warm cache effect)
  - Reconnect-able blobs of persistence
    - Naming
    - Permissions

Beyond V1: Investigating new work items

Three new work items are under investigation

1) Software hints
   - Application usage, access patterns
   - Optimization based on discovered device attributes
   - Present hints emerging in standards (SCSI, NVMe) to applications
2) Atomic transactional behavior
   - Add atomicity and recovery to programming model
   - Not addressed by current sync semantics
3) Remote access
   - Disaggregated memory
   - RDMA direct to NVM
   - High availability, clustering, capacity expansion use cases

RDMA Challenge

- Use case:
  - RDMA copy from local to remote persistent memory
  - for high availability memory mapped files
  - built on NVM.PM.FILE from version 1 programming model
- Requirements:
  - Assurance of remote durability (remote sync?)
  - Efficient byte range access (scatter gather RDMA?)
  - Efficient addressing (late binding without connection teardown?)
  - Efficient write security given fixed addressing in file context
  - Resource recovery and hardware fencing after failure

Summary

- The NVM Programming Model is aligning the industry
  - Gaining common terminology
  - Not forcing specific APIs
  - http://snia.org/forums/sssi/nvmp
- What are we doing with it?
  - PM models expose it
    - Linux PMFS at https://github.com/linux-pmfs
  - New PM models build on existing ones
    - Linux Pmem Examples: https://github.com/pmem/linux-examples
    - New TWG work items
- Emerging technologies will drive increasing work in this area as cost comes down

Thank You

OPENFABRICS ALLIANCE

10TH ANNUAL INTERNATIONAL DEVELOPER WORKSHOP