SDC
STORAGE DEVELOPER CONFERENCE
SNIA ■ SANTA CLARA, 2014

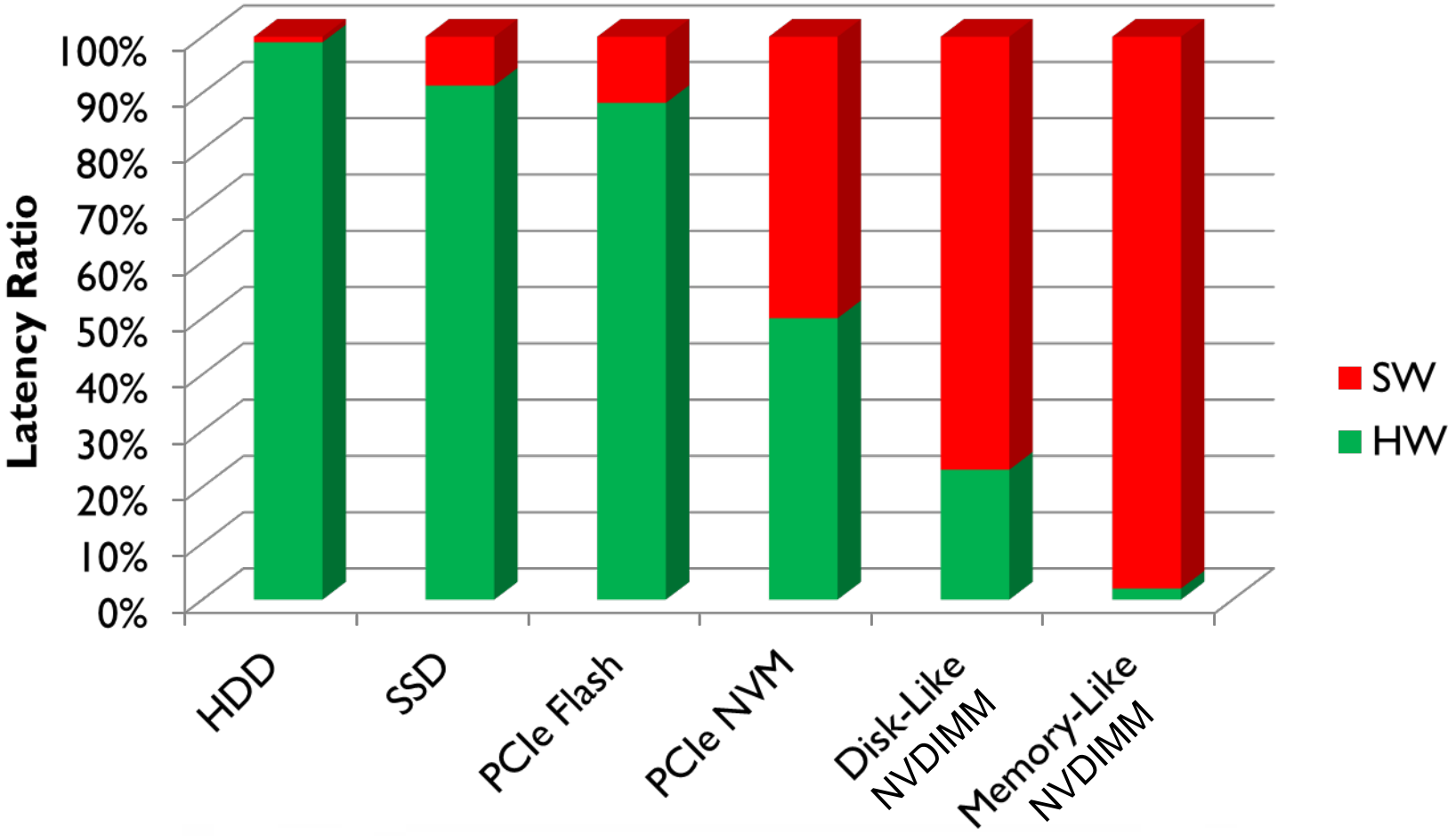
RDMA Requirements for High Availability in the NVM Programming Model

Doug Voigt
HP

Agenda

- NVM Programming Model Motivation
- NVM Programming Model Overview
- Remote Access for High Availability
- RDMA Requirements for High Availability

In NVM technologies, HW is getting ahead of SW … again!

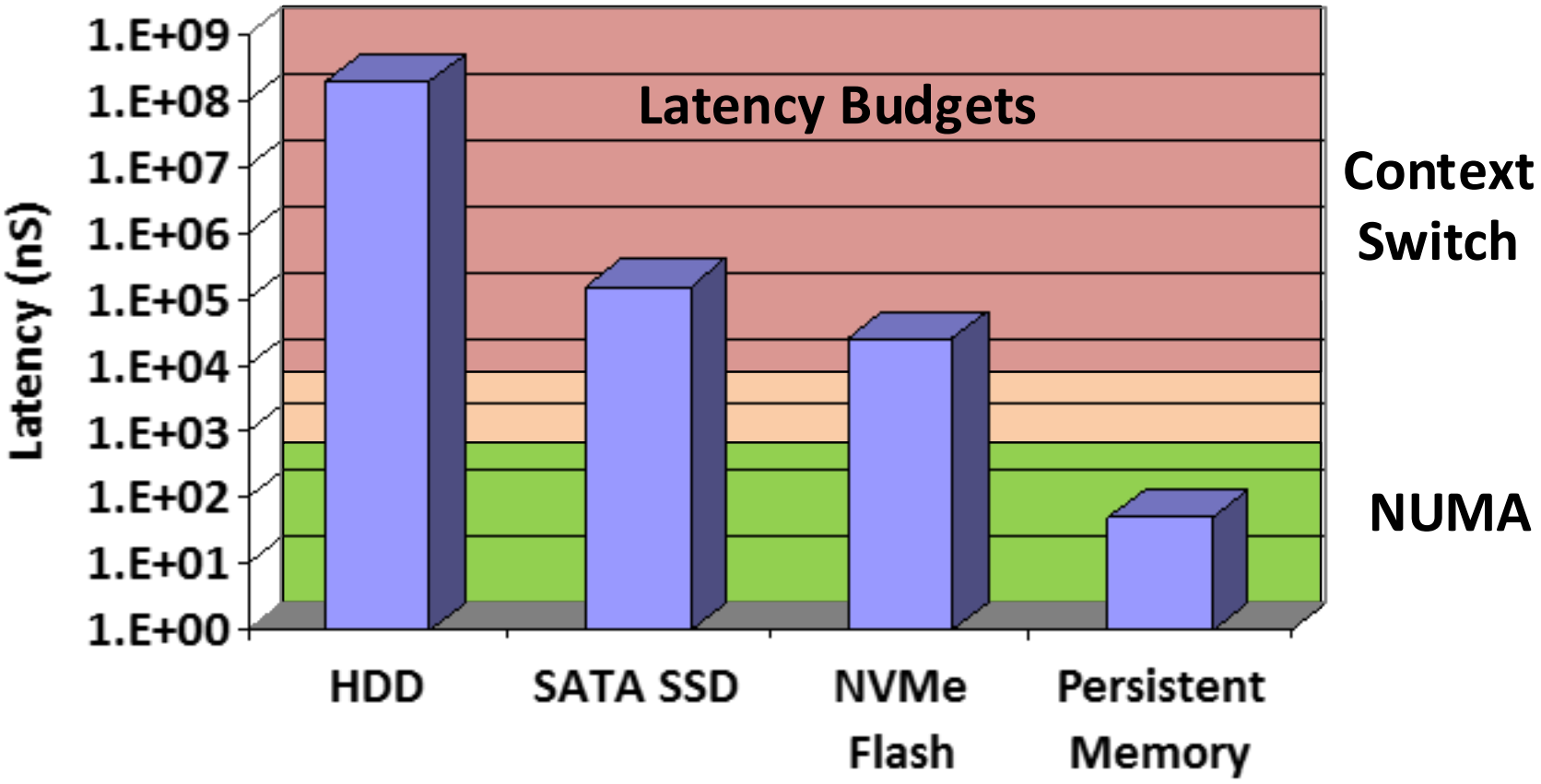


Data from Intel and HP Presentations at NVM Summit 2014

NVM - Non Volatile Memory

- Any type of non-volatile memory including:
  - Block Accessible
    - Disk form factor SSDs (SATA, SCSI, etc.)
    - NVM PCIe cards (NVMe), SCSI-Express, SATA Express,
    - Vendor-specific
  - Byte Accessible
    - Emerging persistent memory (PM) hardware (e.g., NVDIMMs)
- Persistent Memory:
  - Future technologies bring memory-like access for NVM devices,
  - Access time comparable to RAM
  - PM Brings Storage to Memory Slots
    - Option for software to skip copying data from memory to disks
    - No need for specialized "I/O".

New NVM technologies have disruptively low latency



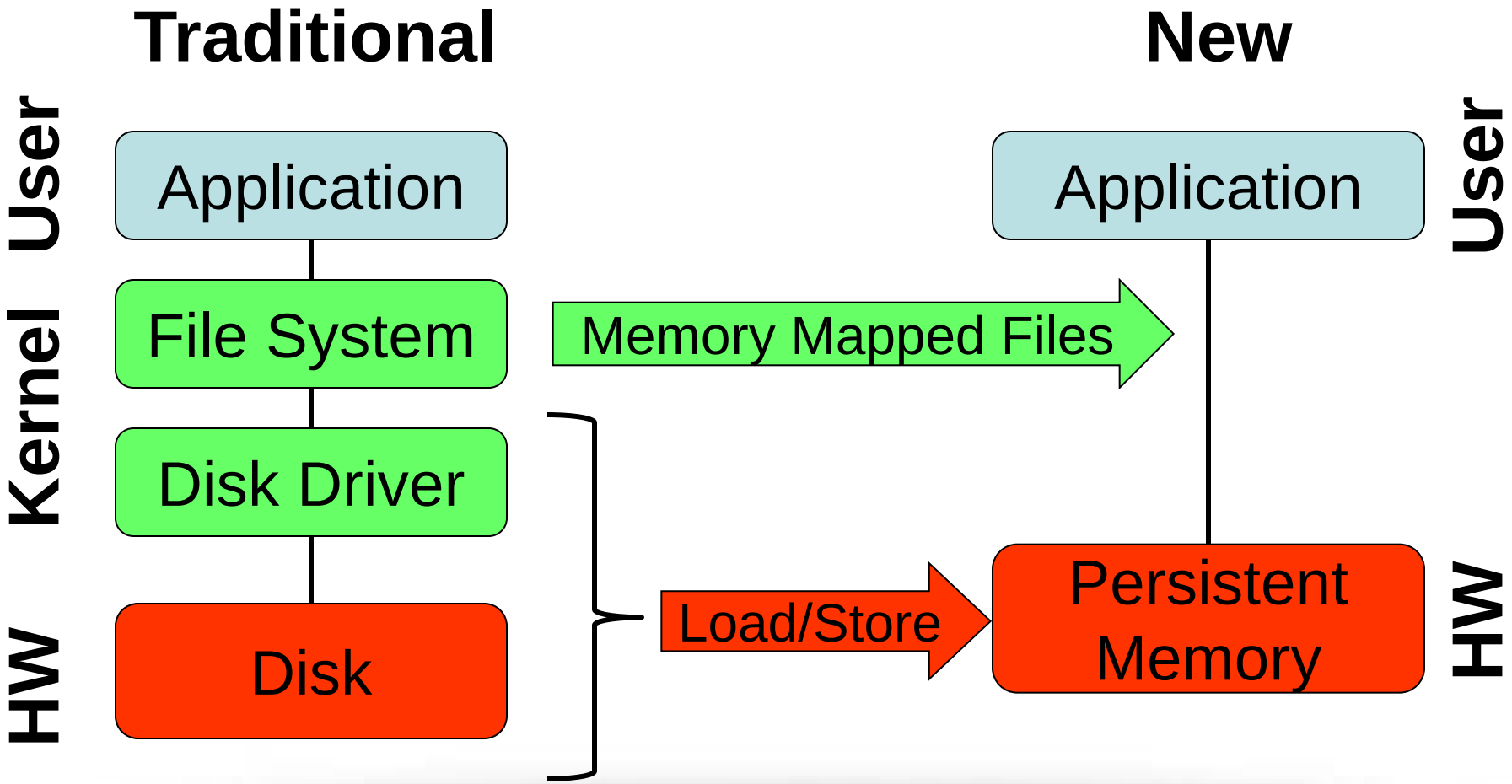
Typical NUMA Range: 0-200 nS

Typical context switch range: above 2-3 uS

Need for A New Programming Model

- Current programming model
  - Data records are created in volatile memory
    - Memory operations
  - Copied to HDD or SSD to make them persistent
    - I/O operations
- Opportunities provided by NVM devices
  - Software to skip the steps that copy data from memory to disks.
  - Software can take advantage of the unique capabilities of both persistent memory and flash NVM
- Need for a new programming model
  - Application writes persistent data directly to NVM which can be treated just like RAM

Eliminate File System Latency with Memory Mapped Files



Creators of the NVM Programming Model

- NVM Programming Technical Work Group is chartered to:
  - Deliver specifications describing the behavior of a common set of software interfaces that provide access to NVM
  - Encourage a common ecosystem for NVM-enabled software without limiting the ability to innovate

SNIA NVM Programming Model

- Version 1 approved by SNIA in December 2013
  - Downloadable by anyone
- Expose new block and file features to applications
  - Atomicity capability and granularity
  - Thin provisioning management
- Use memory mapped files for persistent memory
  - Existing abstraction that can act as a bridge
  - Limits the scope of application re-invention
  - Open source implementations available for incremental innovation (e.g. PMFS)
- Programming Model, not API
  - Described in terms of attributes, actions and use cases
  - Implementations map actions and attributes to API's

The Four Modes

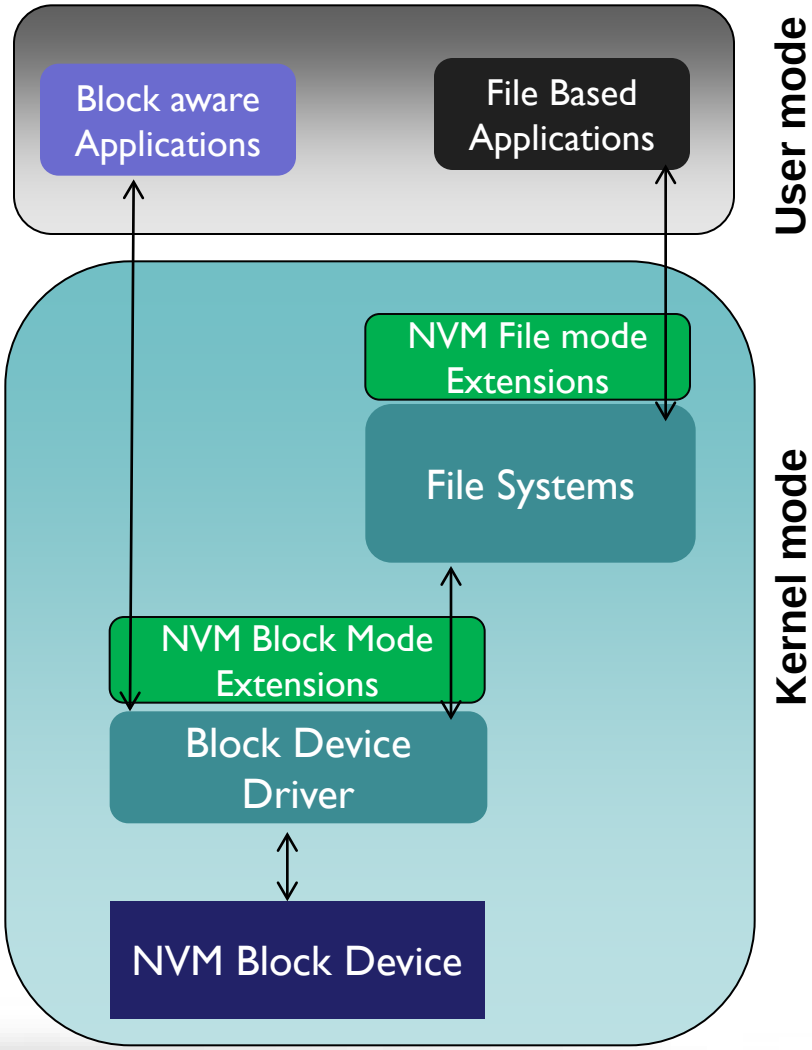
Block Mode Innovation
- Atomics
- Access hints
- NVM-oriented operations

Emerging NVM Technologies
- Memory mapping
- User mode flush
- Error recovery

	Traditional	Persistent Memory
User View	NVM.FILE	NVM.PM.FILE
Kernel Protected	NVM.BLOCK	NVM.PM.VOLUME
Media Type	Disk Drive	Persistent Memory

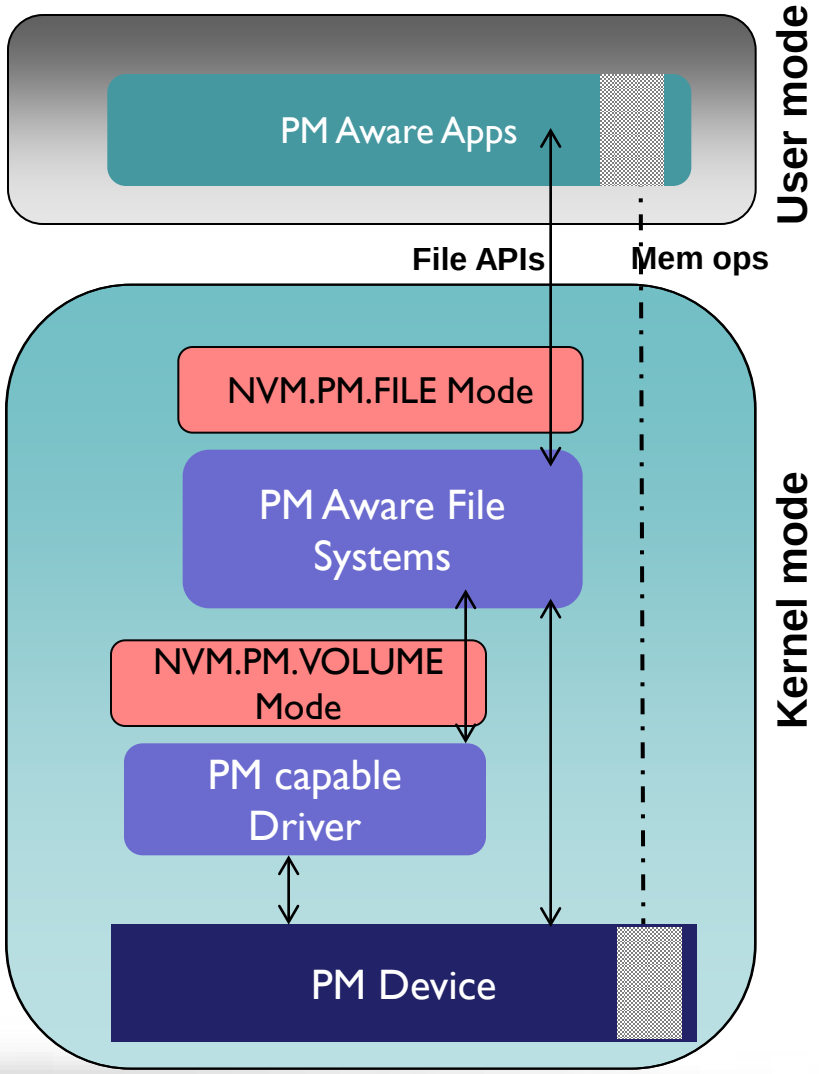
File and Block Mode Extensions

- NVM.BLOCK Mode
  - Targeted for file systems and block-aware applications
  - Atomic writes
  - Length and alignment granularities
  - Thin provisioning management
- NVM.FILE Mode
  - Targeted for file based apps
  - Discovery and use of atomic write features
  - Discovery of granularities

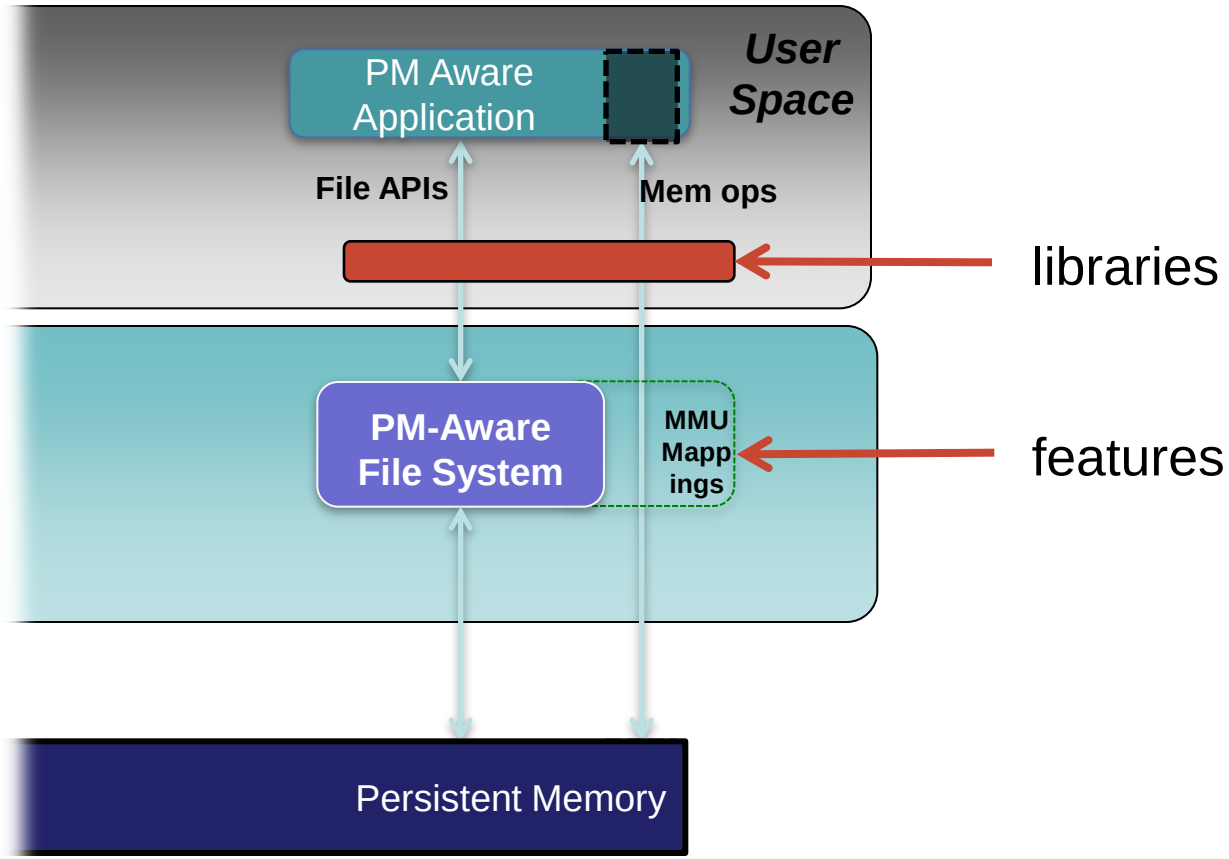


Persistent Memory (PM) Modes

- NVM.PM.VOLUME Mode
  - Software abstraction for persistent memory hardware
  - Address ranges
  - Thin provisioning management
- NVM.PM.FILE Mode
  - Application behavior for accessing PM
  - Mapping PM files to application address space
  - Syncing PM files
  - Error recovery



Building on the Basic PM Model



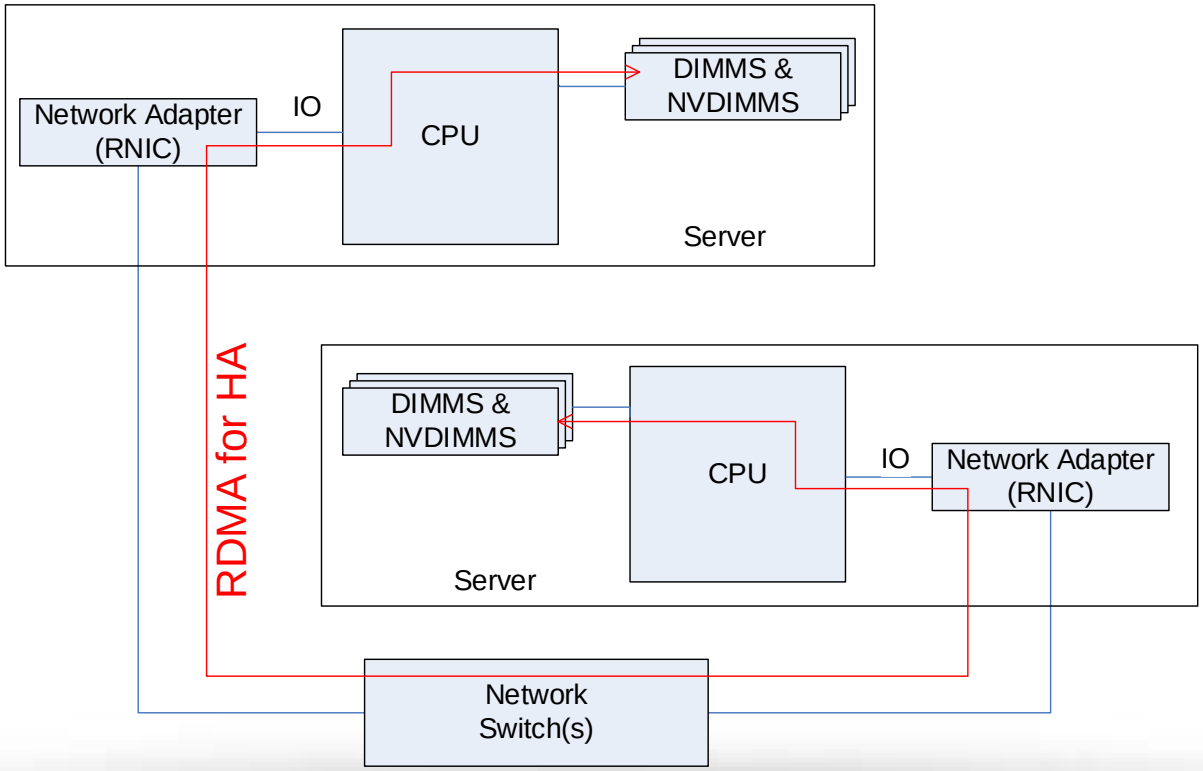
- NVM.PM.FILE programming model “surfaces” PM to application
- Refine presentation with additional libraries that evolve into language extensions
- Add compatible functionality to PM file systems

NVM.PM.FILE MAP, SYNC

- Map
  - Associates memory addresses with open file
  - Caller may request specific address
- Sync
  - Flush CPU cache for indicated range
  - Additional Sync types
    - Optimized Flush – multiple ranges from user space
    - Optimized Flush and Verify – Optimized flush with read back from media

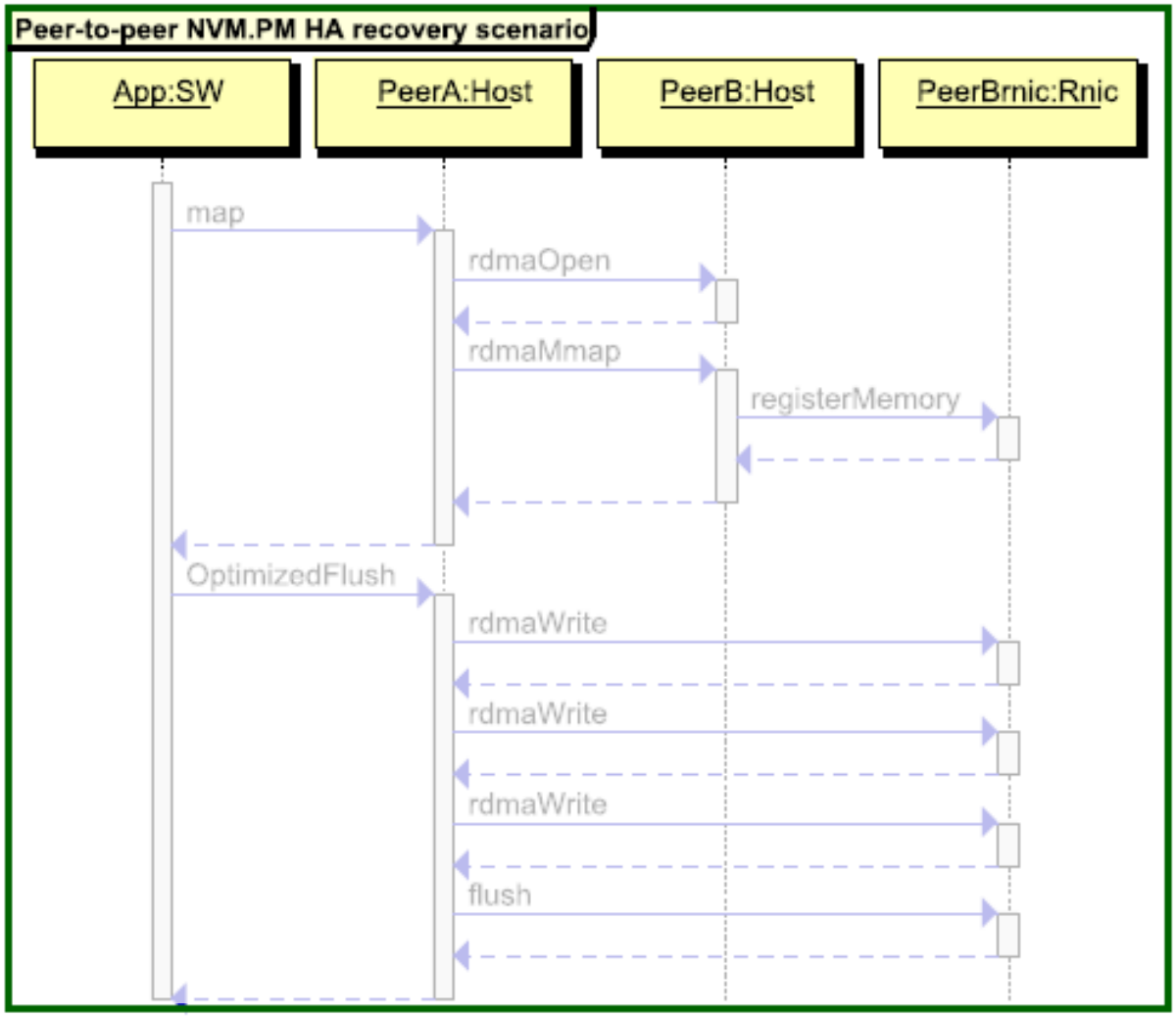
Remote Access to PM for High Availability

RDMA copy from local to networked persistent memory
for high availability memory mapped files
built on NVM.PM.FILE version 1 programming model



RDMA Flow for HA Optimized Flush

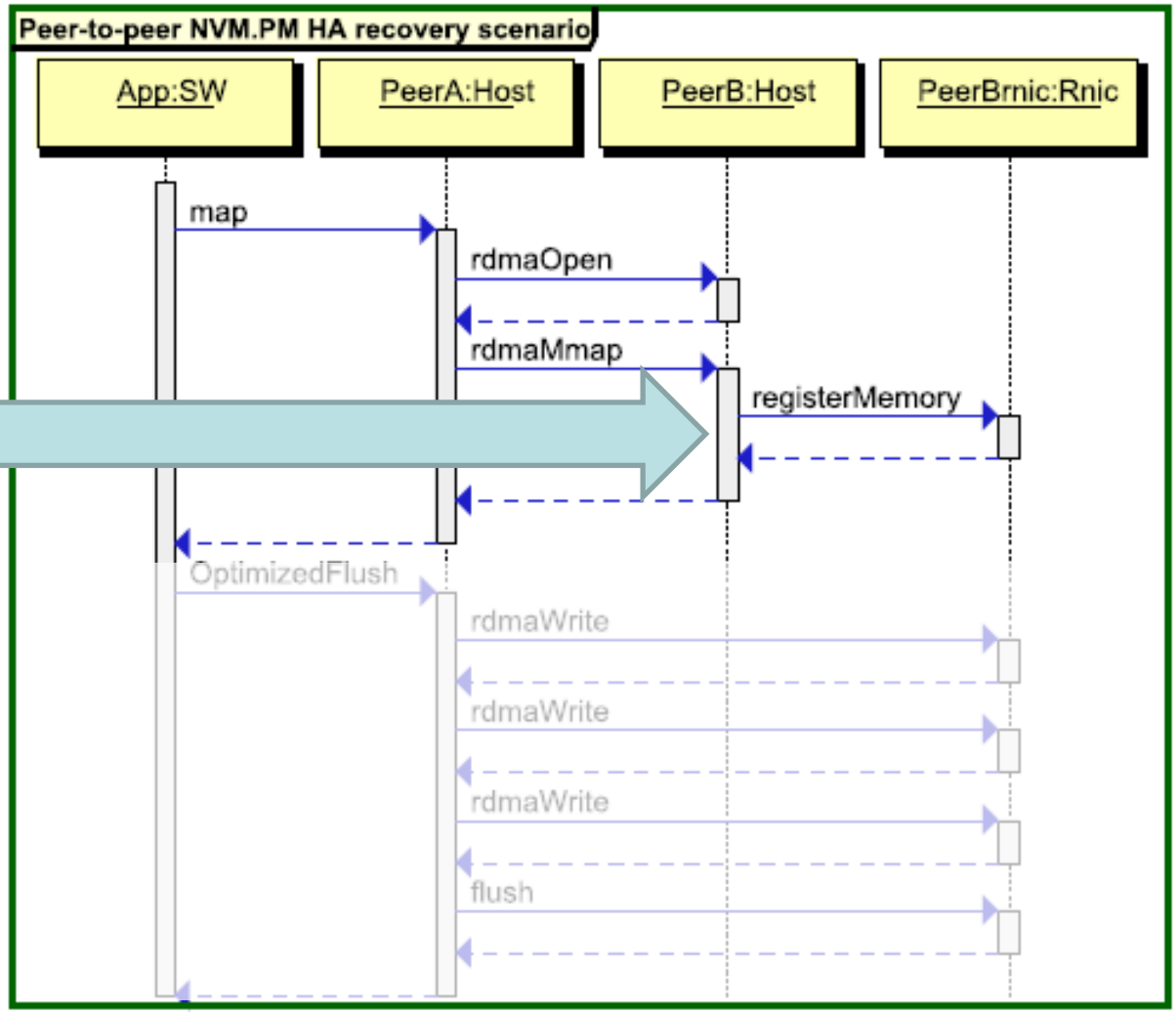
Sequence Diagram actors:
PM aware application
2 hosts mirroring PM
RDMA Adapters (Rnic)



RDMA Flow for HA Optimized Flush

Sequence Diagram actors:
PM aware application
2 hosts mirroring PM
RDMA Adapters (Rnic)

Map triggers RDMA Registration

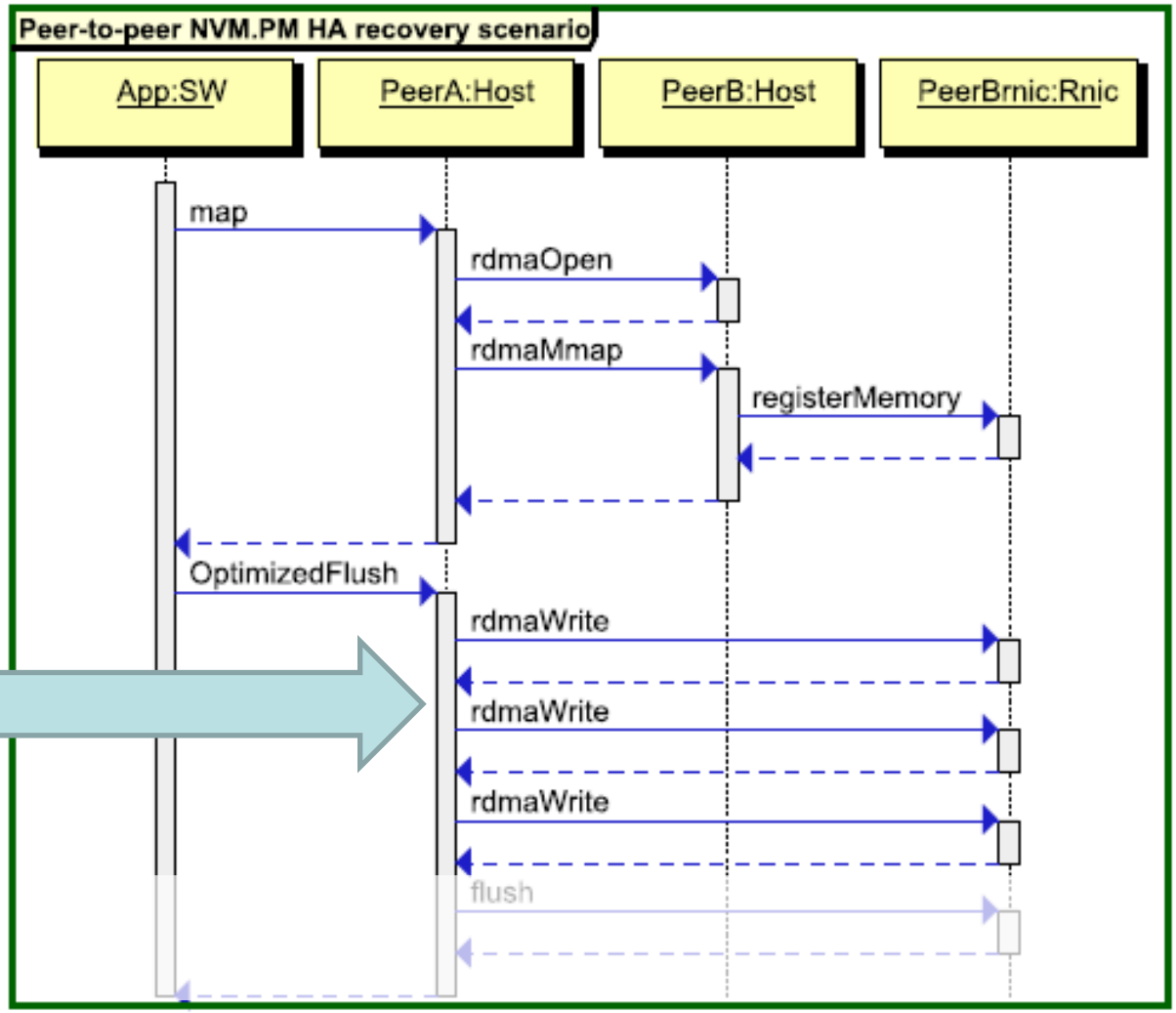


RDMA Flow for HA Optimized Flush

Sequence Diagram actors:
PM aware application
2 hosts mirroring PM
RDMA Adapters (Rnic)

Map triggers RDMA Registration

Optimized Flush triggers dis-contiguous RDMA writes



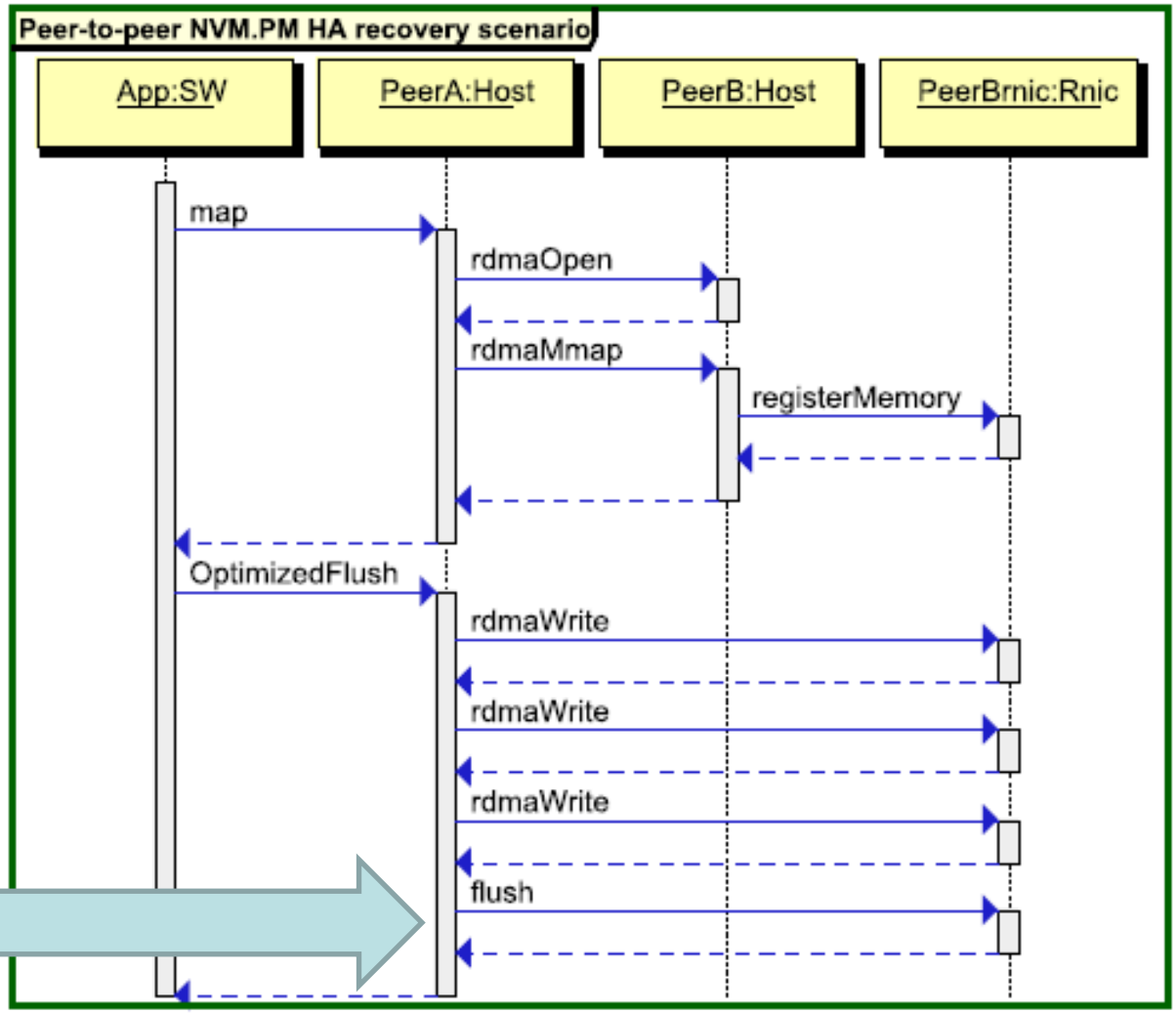
RDMA Flow for HA Optimized Flush

Sequence Diagram actors:
PM aware application
2 hosts mirroring PM
RDMA Adapters (Rnic)

Map triggers RDMA Registration

Optimized Flush triggers dis-contiguous RDMA writes

Flush to guarantee durability and HA



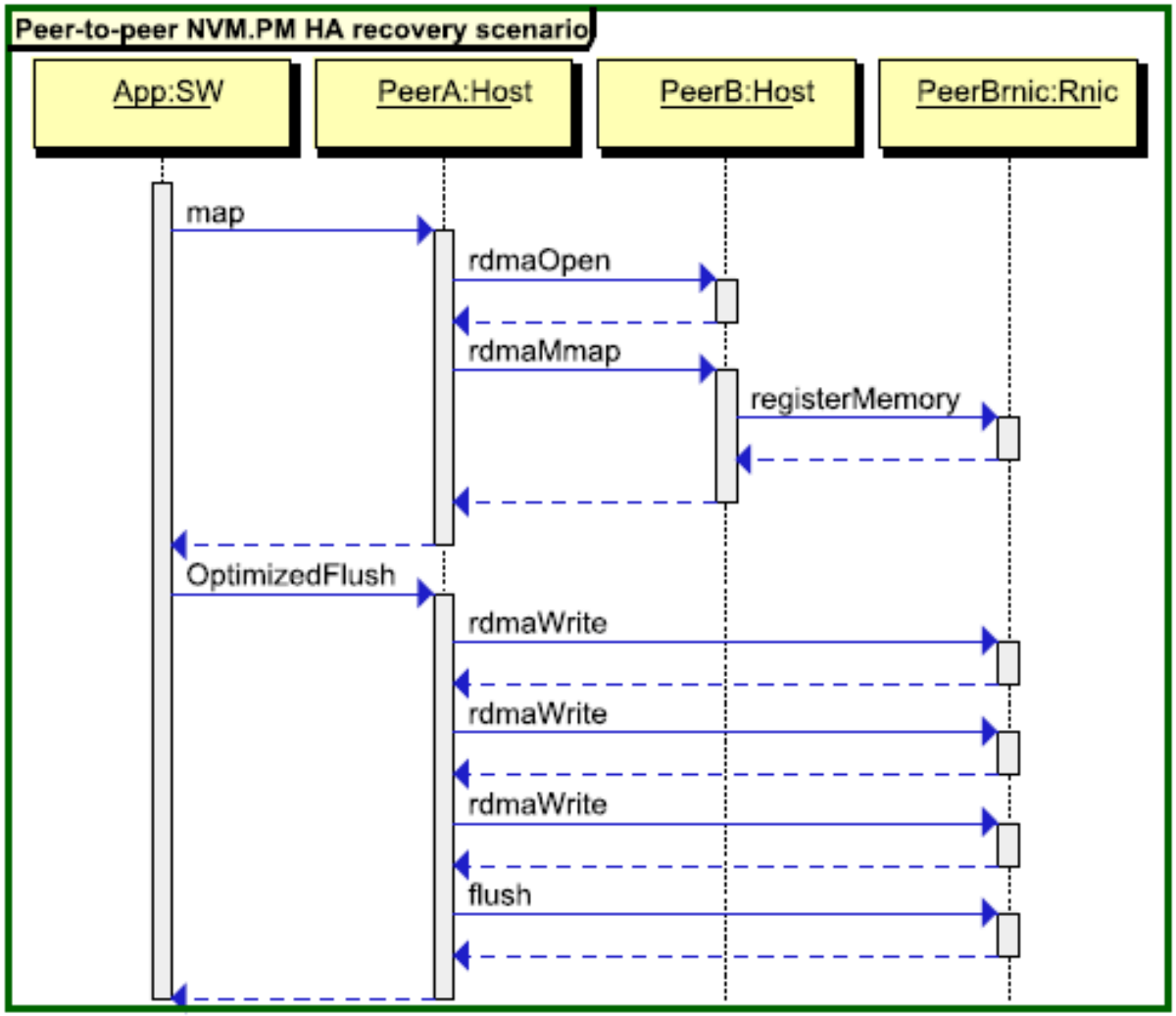
RDMA Flow for HA Optimized Flush

Sequence Diagram actors:
PM aware application
2 hosts mirroring PM
RDMA Adapters (Rnic)

Map triggers RDMA Registration

Optimized Flush triggers dis-contiguous RDMA writes

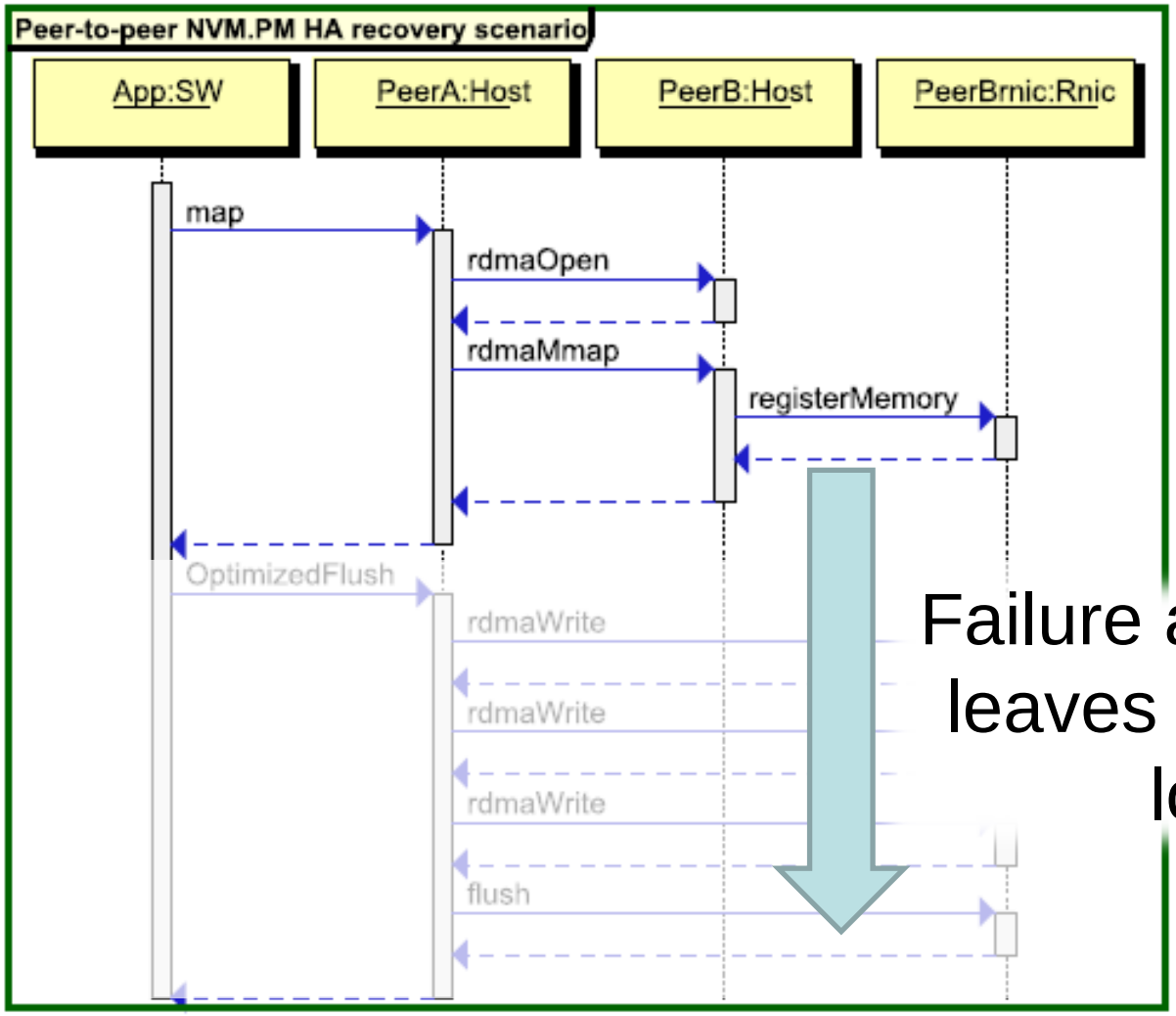
Flush to guarantee durability and HA



RDMA Security for HA PM

- Security depends on trust relationship
  - Between RDMA participants
  - Avoiding denial of service
- Client/Server
  - Server does not trust client
  - Server always orchestrates RDMA
- Peer to Peer
  - Peers replicate data
  - Trust derived from common file system
  - Reduces round trips

Recover from failure



Failure after registration leaves Rnic resources locked up

RDMA Requirements for HA PM

- Assurance of remote durability
- Efficient multiple byte range access
- Efficient write security given addressing correlated with file system
- Resource recovery and hardware fencing after failure

Resources

- SNIA NVM Programming Portal
  - Current information and resources
  - Links to approved specifications
  - http://snia.org/forums/sssi/nvmp
- NVM Programming Model Version 1
  - http://snia.org/tech_activities/standards/curr_standards/npm
- Tutorials and Presentations
  - *The Impact of the NVM Programming Model* (Andy Rudoff)
    - http://www.snia.org/sites/default/files2/SDC2013/presentations/GeneralSession/AndyRudoff_Impact_NVM.pdf
  - *SNIA NVM Programming Model Tutorial* (Paul von Behren)
    - http://www.snia.org/sites/default/education/tutorials/2013/fall/Storage-Solid_State_Storage/Paul-von_Behren_SNIA_NVM_Programming_Model.pdf

RDMA Requirements for High Availability in the NVM Programming Model

Q&A