Persistent Memory, NVM Programming Model, and NVDIMMs

Presented at the 34th International Conference on Massive Storage Systems and Technology (MSST 2018)
May 15, 2018

Contents

- Persistent Memory Overview
- NVM Programming Model
- NVDIMM
- Conclusions

Today's Presenter, Tom Coughlin

Tom Coughlin, President, Coughlin Associates is a widely respected digital storage analyst as well as a business and technology consultant. He has over 35 years in the data storage industry with engineering and management positions at high profile companies.

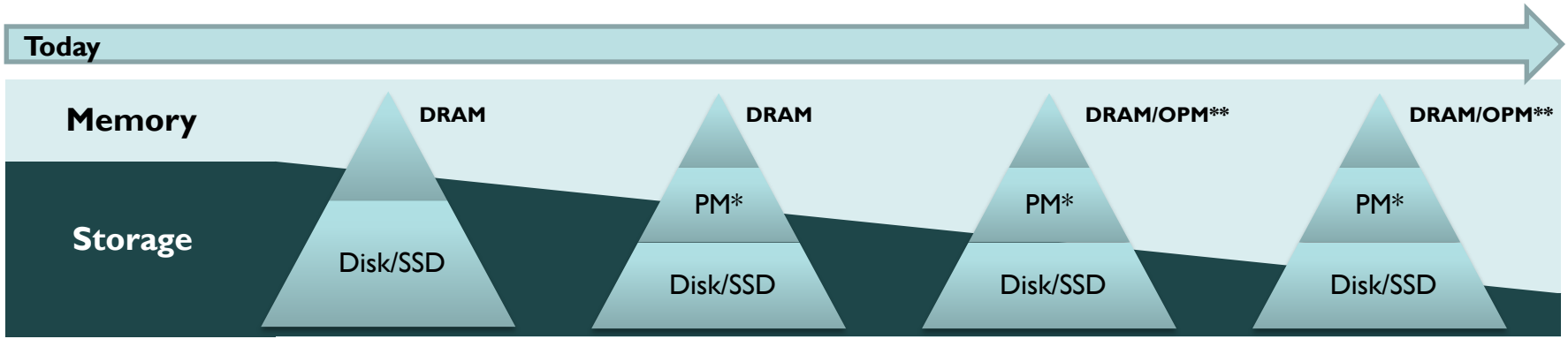
Dr. Coughlin has many publications and six patents to his credit. Tom is also the author of Digital Storage in Consumer Electronics: The Essential Guide, which is now in its second edition with Springer. Coughlin Associates provides market and technology analysis as well as Data Storage Technical and Business Consulting services. Tom publishes the *Digital Storage Technology Newsletter, the Media and Entertainment Storage Report*, the *Emerging Non-Volatile Memory Report* and other industry reports. Tom is also a regular contributor on digital storage for Forbes.com and other blogs.

Tom is active with SMPTE (Journal article writer and Conference Program Committee), SNIA (including a founder of the SNIA SSSI), the IEEE, (he is past Chair of the IEEE Public Visibility Committee, Past Director for IEEE Region 6, President Elect for IEEE USA and active in the Consumer Electronics Society) and other professional organizations. Tom is the founder and organizer of the Annual Storage Visions Conference (www.storagevisions.com as well as the Creative Storage Conference (www.creativestorage.org). He was the general chairman of the annual Flash Memory Summit for 10 years. He is a Fellow of the IEEE and a member of the Consultants Network of Silicon Valley (CNSV). For more information on Tom Coughlin and his publications and activities go to www.tomcoughlin.com

Persistent Memory

Memory & Storage Convergence

- **Volatile and non-volatile technologies are continuing to converge**



*PM = Persistent Memory

**OPM = On-Package Memory

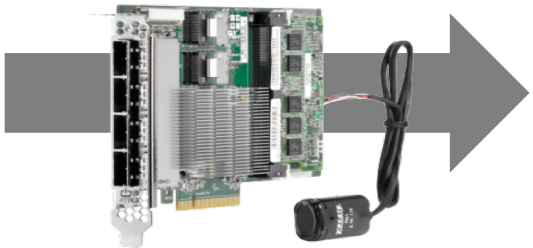
New and Emerging Memory Technologies

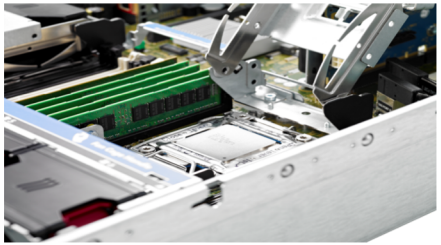
HMC	3DXPoint™ Memory	Low Latency NAND
HBM	MRAM	
RRAM	PCM	Managed DRAM

Source: Gen-Z Consortium 2016

Persistent Memory (PM) Vision

Persistent Memory Brings Storage





Fast
Like Memory

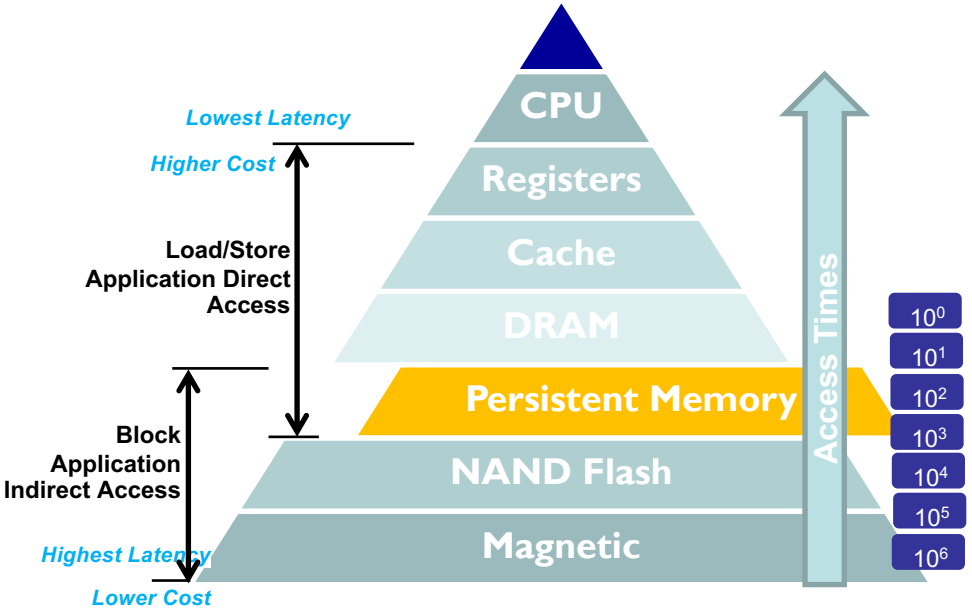
Persistent
Like Storage

To Memory Slots

- For system acceleration
- For real-time data capture, analysis and intelligent response

Persistent Memory

- Bridges the gap between DRAM and Flash
- Dramatically increases system performance
- Enables a fundamental change in computing architecture
- Apps, middleware and OSs are no longer bound by file system overhead in order to run persistent transactions

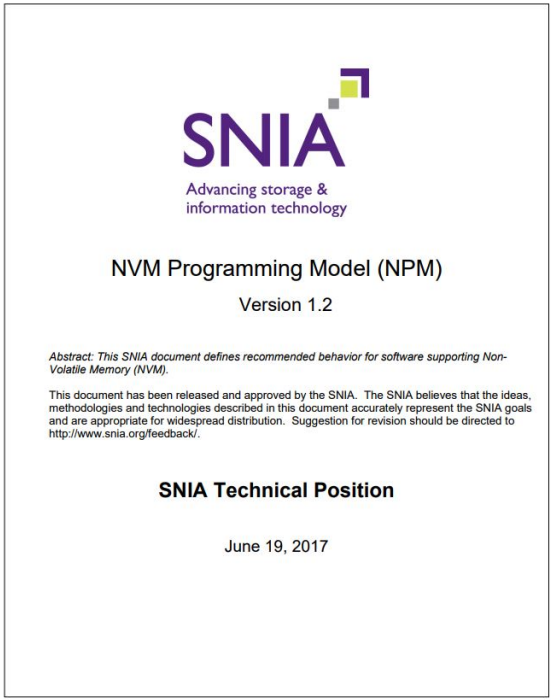


NVM Programming Model –

Writing Applications for Persistent Memory

Role of the NVM Programming Model

- Rally the industry around a view of Persistent Memory that is:
  - Application centric
  - Vendor neutral
  - Achievable today
  - Beyond storage
    - Applications
    - Memory
    - Networking
    - Processors


SNIA
Advancing storage & information technology

NVM Programming Model (NPM)

Version 1.2

Abstract: This SNIA document defines recommended behavior for software supporting Non-Volatile Memory (NVM).

This document has been released and approved by the SNIA. The SNIA believes that the ideas, methodologies and technologies described in this document accurately represent the SNIA goals and are appropriate for widespread distribution. Suggestion for revision should be directed to http://www.snia.org/feedback/.

SNIA Technical Position

June 19, 2017

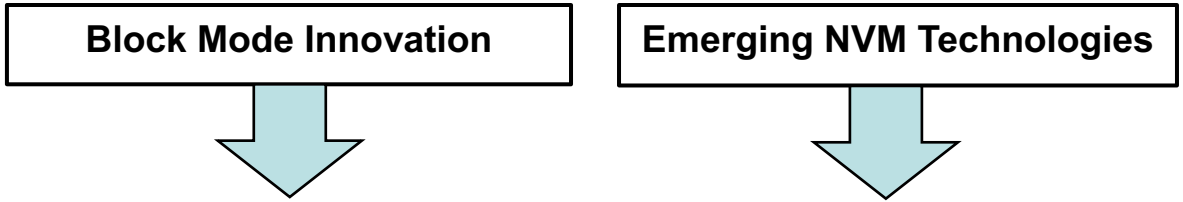
SNIA NVM Programming TWG - Mission

- Accelerate the availability of software that enables Persistent Memory hardware.
  - Hardware includes SSD's and PM
  - Software spans applications and OS's
- Create the NVM Programming Model
  - Describes application visible behaviors
  - Allows API's to align with OS's
  - Exposes opportunities in networks and processors

SNIA NVM Programming Model

- Version 1.2 approved by SNIA in June 2017
  - http://www.snia.org/tech_activities/standards/curr_standards/npm
- Expose new block and file features 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
  - Limits the scope of application re-invention
  - Open source implementations available
- Programming Model, not API
  - Described in terms of attributes, actions and use cases
  - Implementations map actions and attributes to API's

The NVM Programming Model Has 4 Modes



	IO	Persistent Memory
User View	NVM.FILE	NVM.PM.FILE
Kernel Protected	NVM.BLOCK	NVM.PM.VOLUME
Media Type	Disk Drive	Persistent Memory
NVDIMM	Disk-Like	Memory-Like

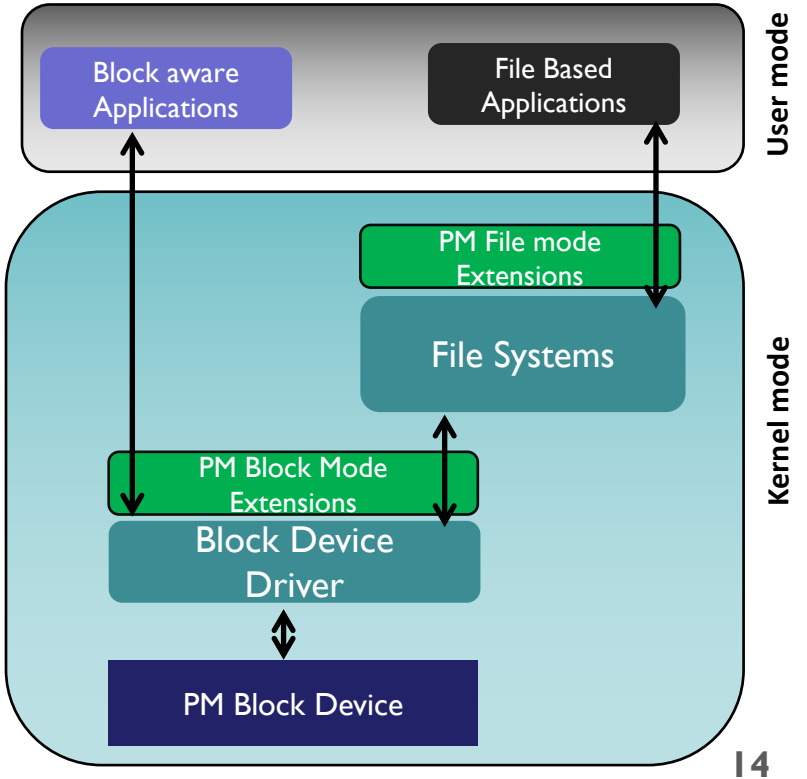
The current version (1.2) of the specification is available at
http://www.snia.org/tech_activities/standards/curr_standards/npm

Programming Model Modes

- Block and File modes use IO
  - Data is read or written using RAM buffers
  - Software controls how to wait (context switch or poll)
  - Status is explicitly checked by software
- Volume and PM modes enable Load/Store
  - Data is loaded into or stored from processor registers
  - Processor makes software wait for data during instruction
  - No status checking – errors generate exceptions

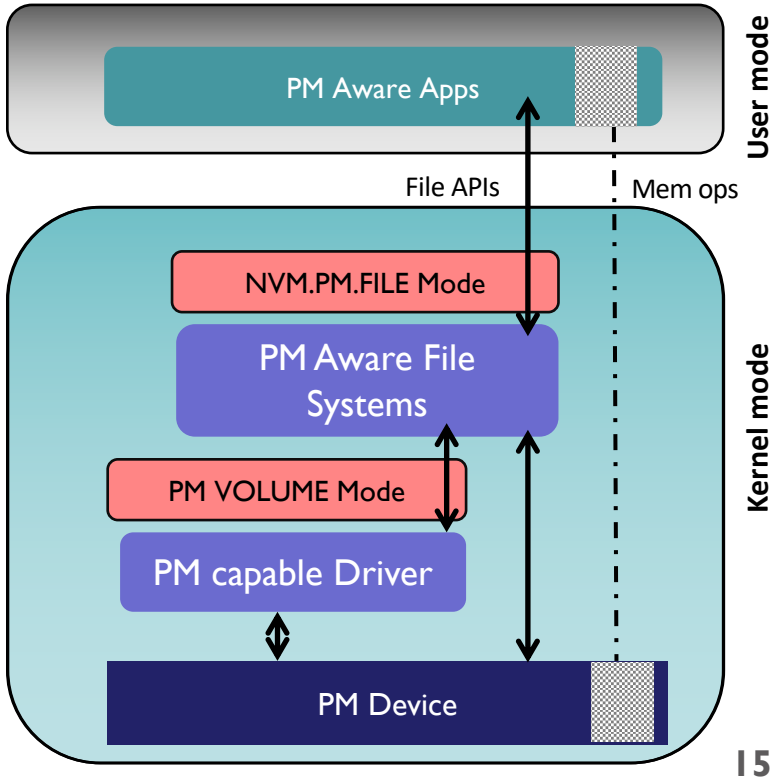
File and Block Mode Extensions

- 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



Map and 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
- Warning! Sync does not guarantee order
  - Parts of CPU cache may be flushed out of order
  - This may occur before the sync action is taken by the application
  - Sync only guarantees that all data in the indicated range has been flushed some time before the sync completes

PM Pointers

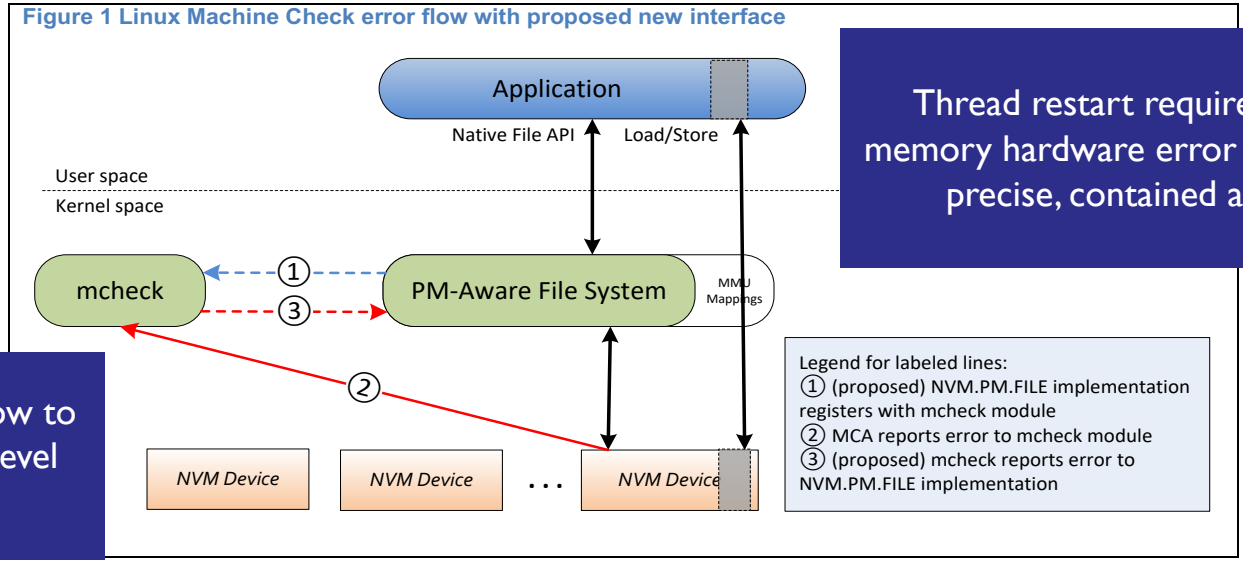
How can one persistent memory mapped data structure refer to another?

- Use its virtual address as a pointer
  - Assumes it will get the same address every time it is memory mapped
  - Requires special virtual address space management
- Use an offset from a relocatable base
  - Base could be the start of the memory mapped file
  - Pointer includes namespace reference

Failure Atomicity

- Current processor + memory systems
  - Guarantee inter-process consistency (SMP)
  - But only provide limited atomicity with respect to failure
    - System reset/restart/crash
    - Power Failure
    - Memory Failure
- Failure atomicity is processor architecture specific
  - Processors provide failure atomicity of aligned fundamental data types
  - Fundamental data types include pointers and integers
  - PM programs use these to create larger atomic updates or transactions
  - Fallback is an additional checksum or CRC

Error Handling – Exceptions Instead of Status



Thread restart required unless memory hardware error detection is precise, contained and live

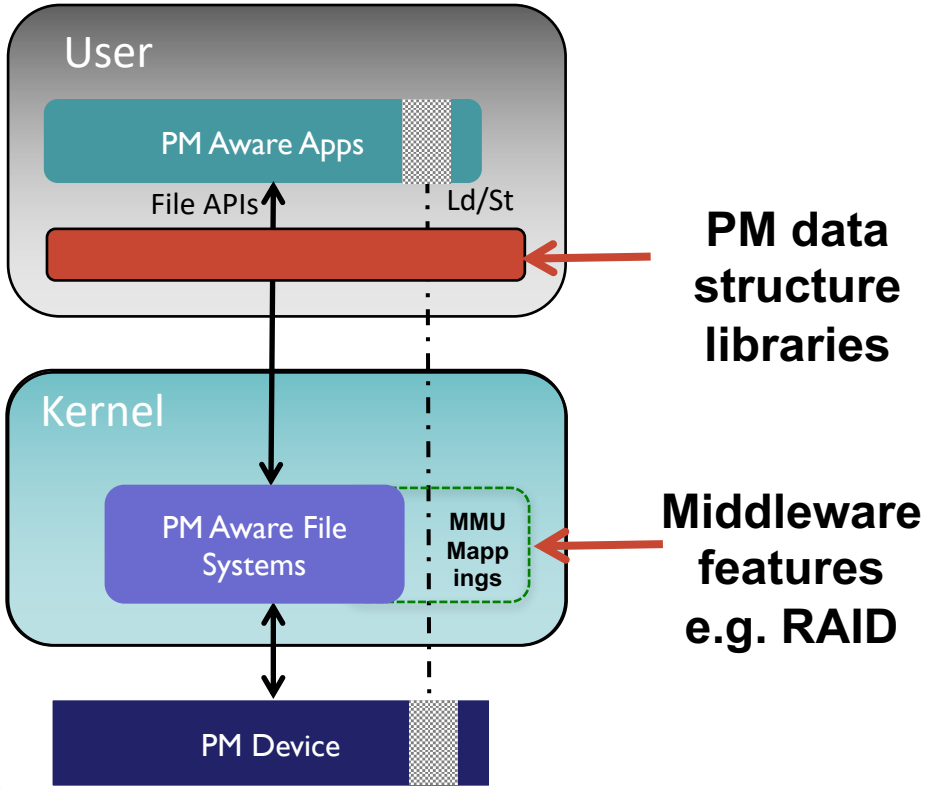
New machine check flow to integrate file and PM level recovery

- Contained: exact memory location(s) are identified
- Precise: instruction execution can be resumed (RTI)
- Live: reported without restart

- Application gets exception if file level recovery fails or backtracking is needed

Recent NVM Programming TWG Work

- Persistent Memory Hardware Threat Model White Paper out for public review
  - Discusses approaches for securing persistent memory, particularly considering unique characteristic of PM
- Remote Access for HA White Paper Published
  - High Availability PM - Remote Optimized Flush



NVM Programming TWG Ongoing Work

- Security
  - PM Hardware Security Threat Model
- Remote persistent memory (via RDMA)
  - Ongoing – optimizations for RDMA worked in multiple forums
  - Remote asynchronous flush
- Higher-level Semantics
  - As we learn more..

Summary

- The NVM Programming Model is aligning the industry (http://pmem.io/)
  - 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

NVDIMM

SNIA Persistent Memory and NVDIMM SIG

Charter

- To accelerate the awareness and adoption of Persistent Memories and NVDIMMs for computing architectures

Activities

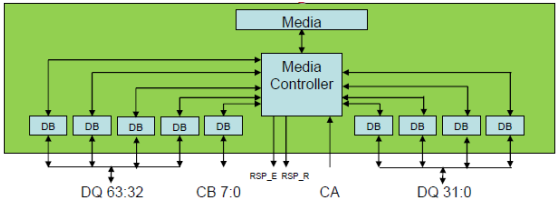
- Educate on the types, benefits, value, and integration of Persistent Memories
- Communicate usage of the NVM Programming Model developed to simplify system integration of current and future PM technologies
- Influence and collaborate with middleware and application vendors to support Persistent Memories
- Develop user perspective case studies, best practices, and vertical industry requirements
- Coordinate with industry standards groups and promote industry standards related to PM and NVDIMM
- Synchronize and communicate a common Persistent Memory taxonomy

Persistent Memory - NVDIMMs

NVDIMM-N

- Host has direct access to DRAM
- CNTLR moves DRAM data to Flash on power fail
- Requires backup power
- CNTLR restores DRAM data from Flash on next boot
- Communication through SMBus
- Byte-addressable DRAM for lowest latency with NAND for persistence backup

NVDIMM-P



- NVDIMM-P interface specification targeting persistent memories and high capacity DRAM memory on DDR4 and DDR5 channels
- Extends the DDR protocol to enable transactional access
- Host is decoupled from the media
- Multiple media types supported
- Supports any latency (ns ~ us)
- JEDEC specification publication in 2018

Technology Comparison

Technology	FeRAM	MRAM	ReRAM	PCM	3D Xpoint	NAND Flash	DRAM NVDIMM
Endurance	10^{12}	10^{12}	10^{6}	10^{8}	10^{6} - 10^{7}	10^{3}	10^{15}
Byte Addressable	yes	yes	yes	yes	yes	no	yes
Latency R/W	70ns-100ns	70ns/70ns	100ns/100µs	20ns/65ns	100ns/500ns	10µs/10µS	40-180ns
Power Consumption	Low	Medium/ Low	Low	Medium	Medium	Low	High
Interface	DRAM	DDR3 DDR4	Flash-like	Proprietary	Proprietary	Toggle ONFI	DDR3 DDR4
Density Path	Low	Gigabit+	Terabit	64Gb+	64Gb+	Gigabit+	Gigabit+

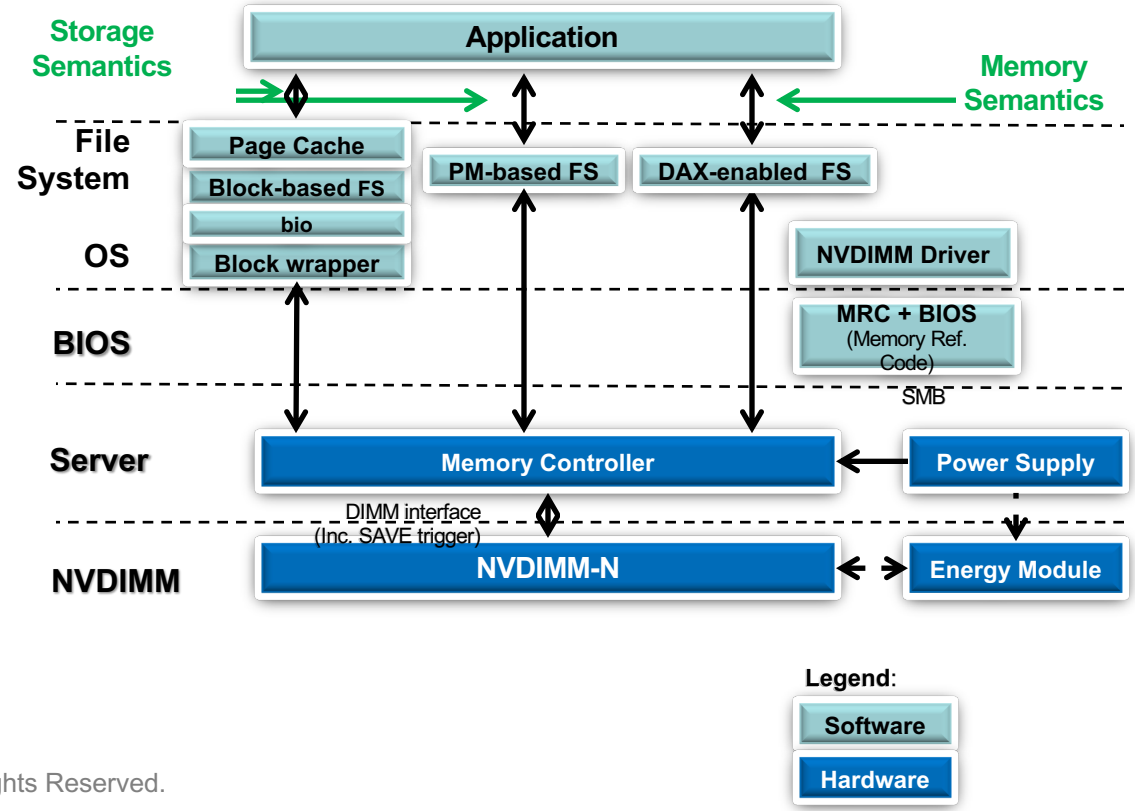
NVDIMM Use Cases

- **In Memory Database**
  - Journaling, reduced recovery time, tables

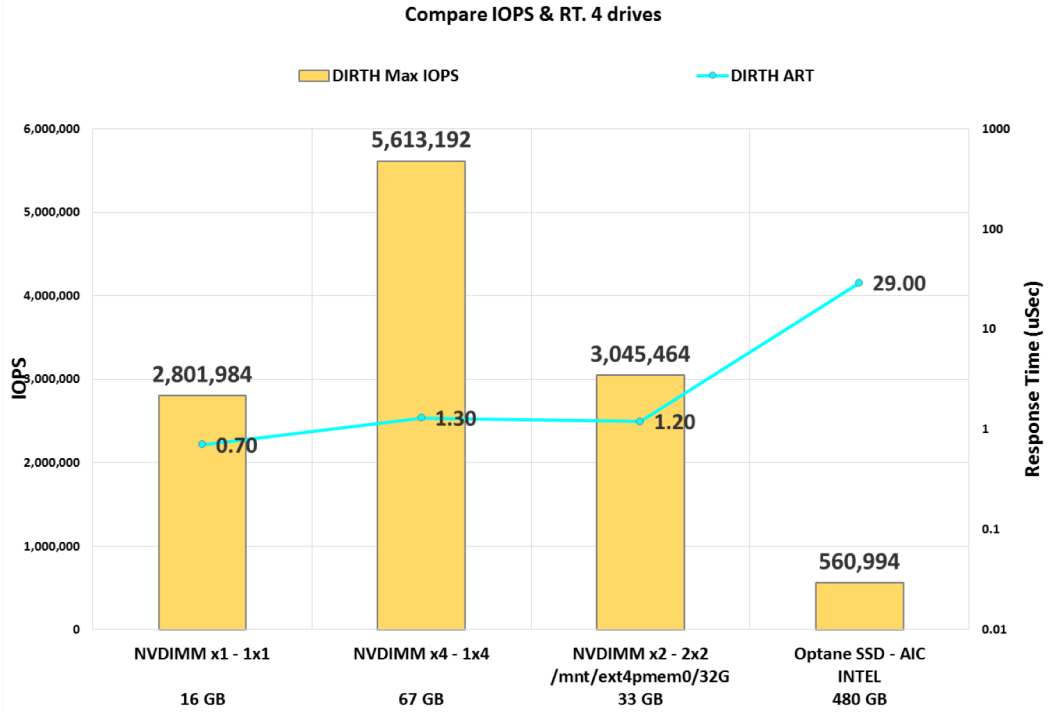
- **Traditional Database**
  - Log acceleration by write combining and caching
- **Enterprise Storage**
  - Tiering, caching, write buffering and meta data storage
- **High-Performance Computing**
  - Check point acceleration and/or elimination

NVDIMM Ecosystem

- Standardized through NFIT and JEDEC
- Linux 4.4+ kernels have the software stack
- Open source library is available for applications



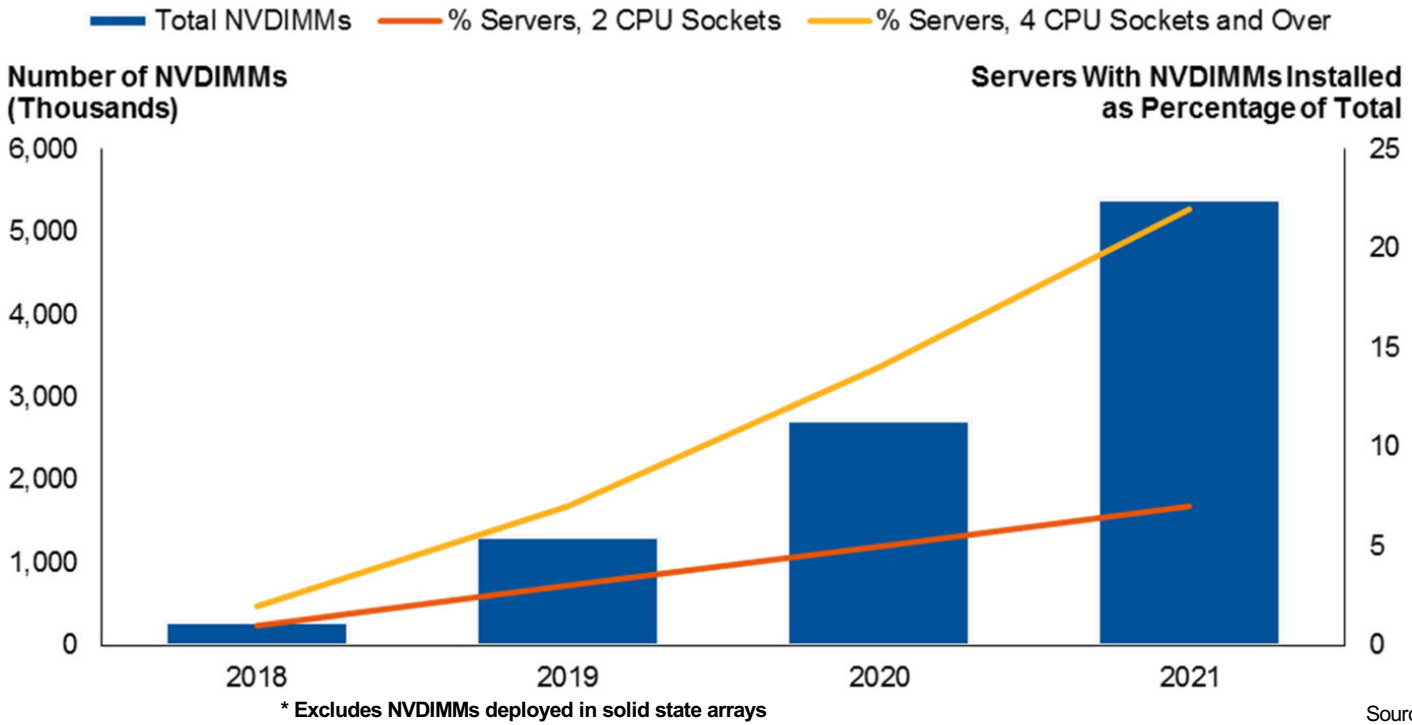
NVDIMM Performance Comparison



- Test Platform: Supermicro X11DRI 16GB DDR4 2400 Mhz RDIMM RAM, Intel XEON 8160 2.1 Ghz 24 core, 16 GB DDR4 JEDEC NVDIMM-N. 480GB Optane SSD
- Software: Ubuntu 16.04.3 LTS Linux 4.10.0-28; DAX File System
- Test Software: Calypso CTS 7.0 fe 1.26.25 be 1.9.317

What is the Outlook?

Nonvolatile Memory Shipments as a Percentage of Server Memory (PB) and NVDIMM Unit Forecast*



* Excludes NVDIMMs deployed in solid state arrays

Source; Gartner, 2017

Infrastructure Changes

- Operating System
- File system changes for memory mapped files
- Memory Management software
- Hypervisors
- Containers
- Allocation of Persistent Memory to Guests
- Coordinating with Guest's use of Persistent Memory
- User space libraries supporting Persistent Memory
- Support for legacy interfaces with Persistent Memory-aware implementations
- Securing application data in a multi-tenant environment

Linux Kernel 4.4+ NVDIMM-N OS Support

SNIA

- Linux 4.2 + subsystems added support of NVDIMMs. Mostly stable from 4.4
- NVDIMM modules presented as device links: /dev/pmem0, /dev/pmem1
- QEMO support (experimental)
- XFS-DAX and EXT4-DAX available

DAX	File system extensions to bypass the page cache and block layer to memory map persistent memory, from a PMEM block device, directly into a process address space.
BTT (Block,Atomic)	Block Translation Table: Persistent memory is byte addressable. Existing software may have an expectation that the power-fail-atomicity of writes is at least one sector, 512 bytes. The BTT is an indirection table with atomic update semantics to front a PMEM/BLK block device driver and present arbitrary atomic sector sizes.
PMEM	A system-physical-address range where writes are persistent. A block device composed of PMEM is capable of DAX. A PMEM address range may span an interleave of several DIMMs.
BLK	A set of one or more programmable memory mapped apertures provided by a DIMM to access its media. This indirection precludes the performance benefit of interleaving, but enables DIMM-bounded failure modes.

Windows NVDIMM-N OS Support

- Windows Server 2016 supports DDR4 NVDIMM-N
- Block Mode
  - No code change, fast I/O device (4K sectors)
  - Still have software overhead of I/O path
- Direct Access
  - Achieve full performance potential of NVDIMM using memory-mapped files on Direct Access volumes (NTFS-DAX)
  - No I/O, no queueing, no async reads/writes
- More info on Windows NVDIMM-N support:
  - https://channel9.msdn.com/events/build/2016/p466
  - https://channel9.msdn.com/events/build/2016/p470

4K Random Write	Thread Count	IOPS	Latency (us)
NVDIMM-N (block)	1	187,302	5.01
NVDIMM-N (DAX)	1	1,667,788	0.52

Persistent Memory Standards

- **JEDEC JESD 245, 245B: Byte Addressable Energy Backed Interface**
  - Defines the host to device interface and features supported for a NVDIMM-N

- **ACPI 6.2**
  - NVDIMM Firmware Interface Table (NFIT)
  - NVM Root and NVDIMM objects in ACPI namespace
  - Address Range Scrub (ARS)
  - Uncorrectable memory error handling
  - Notification mechanism for NVDIMM health events and runtime detected uncorrectable memory error

Source; SNIA Persistent Memory Summit , Jan, 2018

Encryption

- Estimated 10-20% of NVDIMM end-users require encryption
  - Financial – high-speed trading, OLTP
  - Public – DoD
  - Health – medical records
  - Private - corporate IT departments

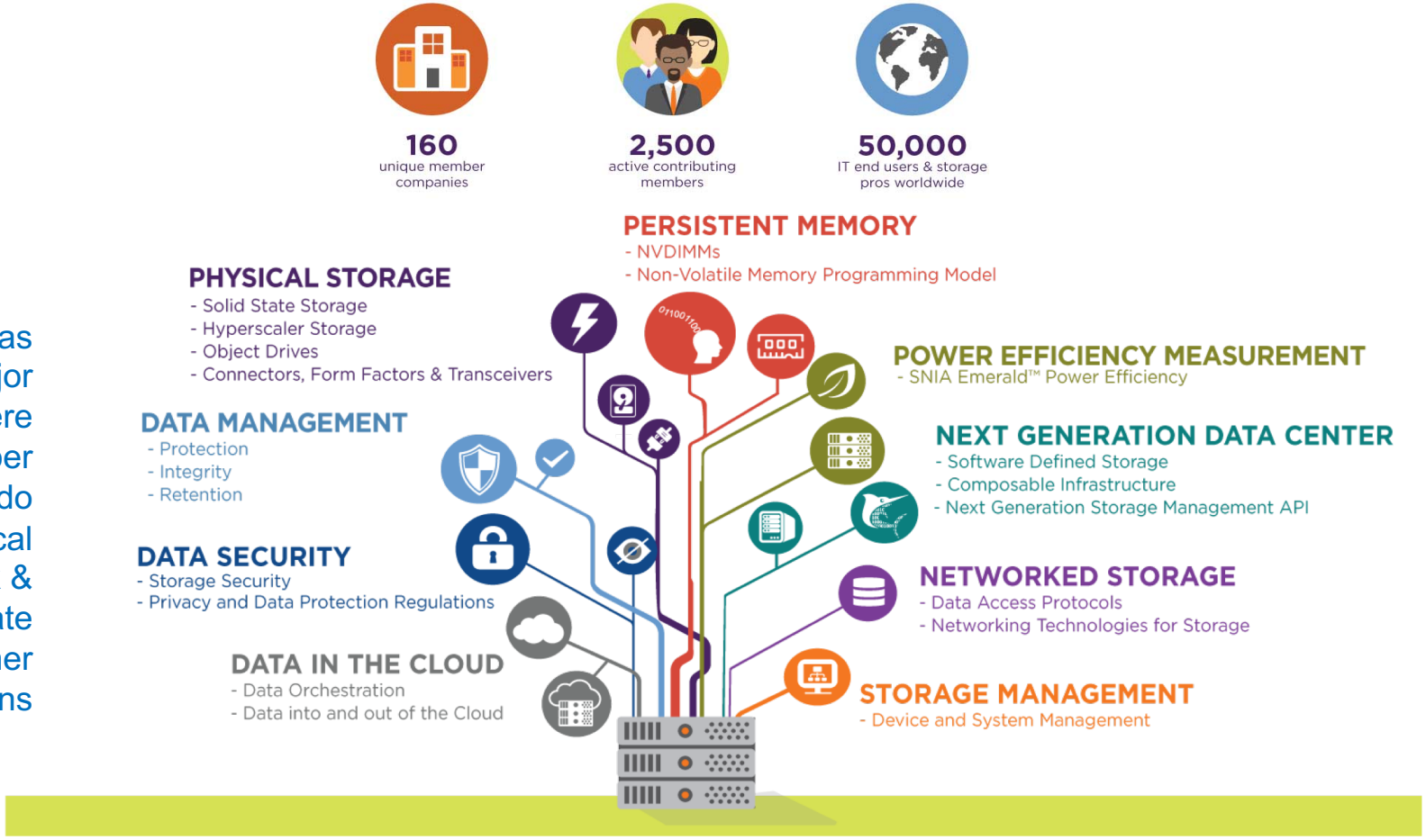
- With block access NVDIMMs the controller chip can manage encryption in the same way as SSDs
- With byte access NVDIMMs the host memory controller needs to provide encryption support
- A key is supplied by the host to support backup with encryption (which could impact performance)
- During a system power loss, in-flight data written from the DRAM to the Flash will be encrypted

Key Takeaways

- Workloads are being re-architected to use large amounts of data placed in local memory
- Data reload times are significant, driving a need to retain data through a power failure
- More Persistent Memory technologies are emerging
- Applications help drive demand for Persistent Memory
- The NVM Programming Model is ideal for NVDIMMs
- Standardization enables wider adoption of Persistent Memory-aware applications

Persistent Memory work is a key area of focus for SNIA's

SNIA has nine major areas where member volunteers do technical work & collaborate with other organizations



You're Invited to Contribute!

- SNIA Persistent Memory & NVDIMM SIG is advancing the awareness & adoption of Persistent Memory and NVDIMMs for computing architectures
- We're looking for companies and individuals to identify work items and activities including
  - Developing user perspective case studies, best practices, and vertical industry requirements
  - Synchronizing and communicating a common Persistent Memory taxonomy
- Join us for a series of open calls beginning May 11 to set our agenda for 2018-2019 – all are invited.
- Visit snia.org/PM for details and to register

Thanks for Attending

Questions?

Visit www.snia.org/pm
for Persistent Memory videos, webcasts, and presentations
(including the January 2018 Persistent Memory Summit)

Backup Slides

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
    (Binary Large Object – set of binary data stored as a single entity)
    - Naming
    - Permissions