PMEM-201-1
Software and Applications

Chairs: Arthur Sainio, SMART Modular
Jeff Chang, AgigA Tech
Co-Chairs, SNIA Persistent Memory and NVDIMM SIG
Speakers – Part 1

- Doug Voigt, HPE
  - *The SNIA NVM Programming Model*

- David Tseng, Bigtera
  - *NVDIMM: The Savior of SSD Endurance in CEPH*

- Sreekanth Garigala, Western Digital
  - *Performance Benefits of NVDIMMs in Enterprise Data Storage Platforms*

- Myoungsoo Jung, Yonsei University
  - *Design of PRAM-based Persistent NVDIMM Controllers to Prepare the Data Age*
Panelists – Part 2

- Brian Bulkowski, Aerospike
- Scott Miller, Dreamworks Animation
- Jia Shi, Oracle
- Rakesh Radhakrishnan, VMware
Upcoming Events Featuring Persistent Memory Talks

Flash Memory Summit
2018
Santa Clara, CA

Storage Developer Conference 2018
9/24-9/27 Santa Clara, CA

Persistent Memory Summit
JANUARY 24, 2019 | SANTA CLARA, CA

SDC discount registration cards in FMS bags & at SNIA booth 820

Complimentary registration now open at snia.org/pm-summit
Flash Memory Summit 2018 – Exhibit Floor

Come and talk to us and get involved!
JEDEC booth #803

FMS Persistent Memory Track Presented by:

JEDEC
OpenFabrics Alliance
SNIA
NVM Programming Model
Overview and Status

System behaviors
for broad application support

By Doug Voigt
NVM Programming Model TWG - Mission

- Accelerate the availability of software that enables Persistent Memory (PM) 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
  - Describes opportunities in networks and processors
SNIA NVM Programming Model

- **Version 1.2** approved by SNIA in June 2017
- 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

<table>
<thead>
<tr>
<th></th>
<th>Block Mode Innovation</th>
<th>Emerging PM Technologies</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>User View</strong></td>
<td>IO</td>
<td>Persistent Memory</td>
</tr>
<tr>
<td>NVM.FILE</td>
<td></td>
<td>NVM.PM.FILE</td>
</tr>
<tr>
<td><strong>Kernel Protected</strong></td>
<td>NVM.BLOCK</td>
<td>NVM.PM.VOLUME</td>
</tr>
<tr>
<td>Media Type</td>
<td>Disk Drive</td>
<td>Persistent Memory</td>
</tr>
<tr>
<td>NVDIMM</td>
<td>Disk-Like</td>
<td>Memory-Like</td>
</tr>
</tbody>
</table>

The current version (1.2) of the specification is available at

[https://www.snia.org/sites/default/files/technical_work-final/NVMProgrammingModel_v1.2.pdf](https://www.snia.org/sites/default/files/technical_work-final/NVMProgrammingModel_v1.2.pdf)
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/Move
  - Data is loaded into or stored from processor registers
  - Processor waits for data during instruction
  - No status returned – errors generate exceptions
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
Past Accomplishments

- NVM Programming Model 1 published December 2013
- Best business application award, FMS 2014
- NVM Programming Model 1.1 published March 2015
- Remote Access for High Availability white paper published 2016
- PM Atomics Transactions white paper published 2016
- NVM Programming Model 1.2 published June 2017
  - Major new installment on error handling
  - Optimized Flush Allowed
  - Deep Flush
Ongoing NVMPM Work Items

- **NVM Programming Model Specification 1.3**
  - Update specification to reflect learning from implementations
  - Incorporate learning from remote access white paper
    - Asynchronous Flush
    - Remote persistence ordering, error handling

- **Remote Access Collaboration with OFA OFIWG**
  - PM Remote Access for HA V1.1
  - Expand remote access use case enumeration

- **PM Security threat model**
  - Identify potential security gaps created by PM

- **PM Management**
  - Integration into Redfish and Swordfish
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