
# PERSISTENT MEMORY

Jim Harris
Intel DCG Storage Group
jimharris@freebsd.org
November 2, 2015

# DIDN'T WE DISCUSS THIS TWO YEARS AGO?

November 2013 – Andy Rudoff from Intel presented on "Enabling Persistent Memory Programming" at FreeBSD Vendor Summit

Since then – lots of progress on software enabling

- SNIA NVM Programming Model v1.1
- NVM Library at pmem.io
- x86 instruction set extensions for persistent memory disclosed
- DAX added to Linux v4.0 kernel for page-cache bypass
- Ext4 and XFS Linux filesystems add DAX support

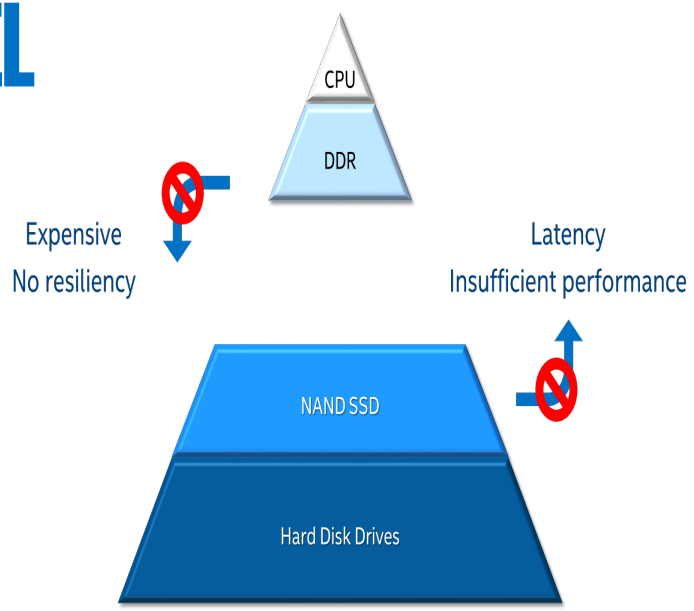
So why talk about this again here?

- More technical details on the underlying SW/HW mechanisms
- Spur discussion on NVM usage models important to FreeBSD vendors
- Start to formulate a plan for enabling persistent memory programming on FreeBSD

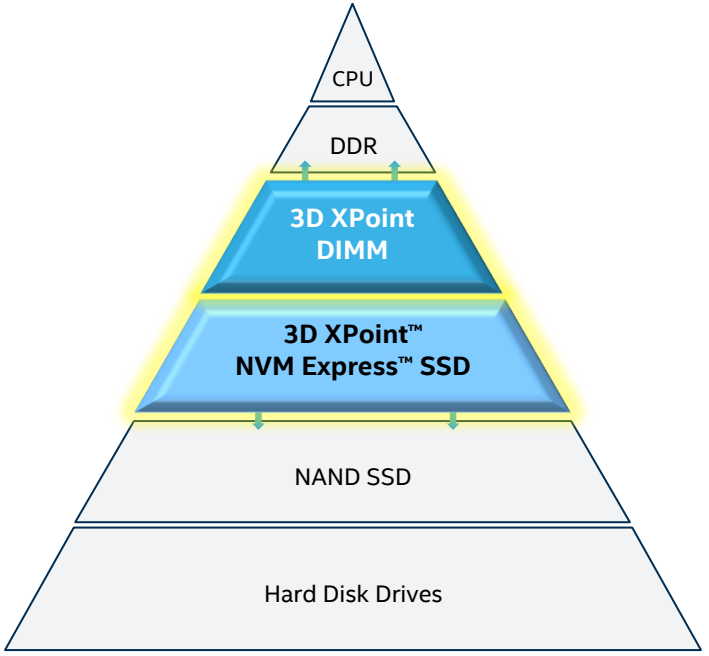
That's all great – but when do we get hardware?

# TRENDS IN NVM PRODUCTS FROM INTEL

- **NAND Based Solutions**
  - Latency is much higher than DRAM latency
  - Much lower endurance compared to DRAM
  - Unlike DRAM no support for load/store semantics



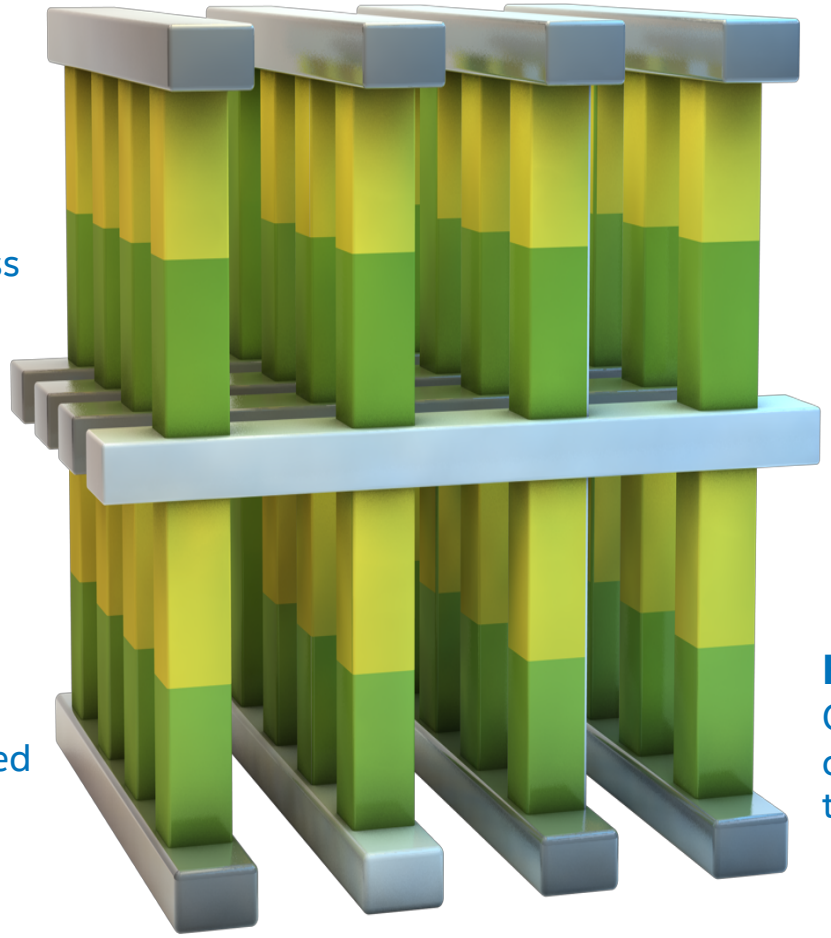

- **Next Generation NVM Solutions**
  - Breakthrough High Density Memory
    - 10 times the density of DRAM
  - Low Latency, closer to DRAM latency
  - Enterprise Class endurance and reliability compared to NAND
    - 1000 times more than NAND
  - Support for Memory Semantics



# WHAT IS 3D XPOINT™?

## Word (Cache Line)

**Crosspoint Structure**
Selectors allow dense packing and individual access to bits



## NVM Breakthrough Material Advances

Compatible switch and memory cell materials

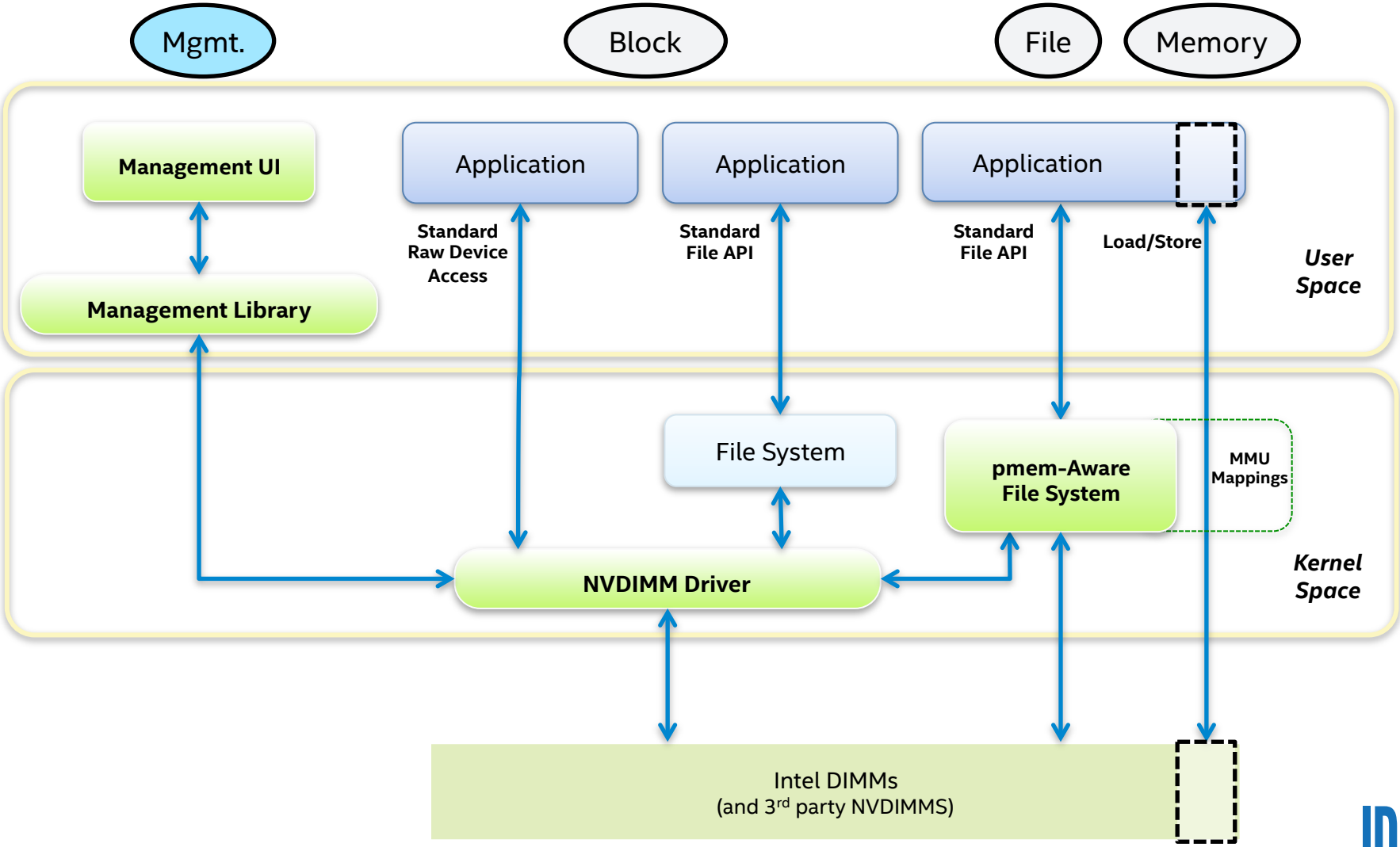
## Large Memory Capacity

**Crosspoint & Scalable**
Memory layers can be stacked in a 3D manner

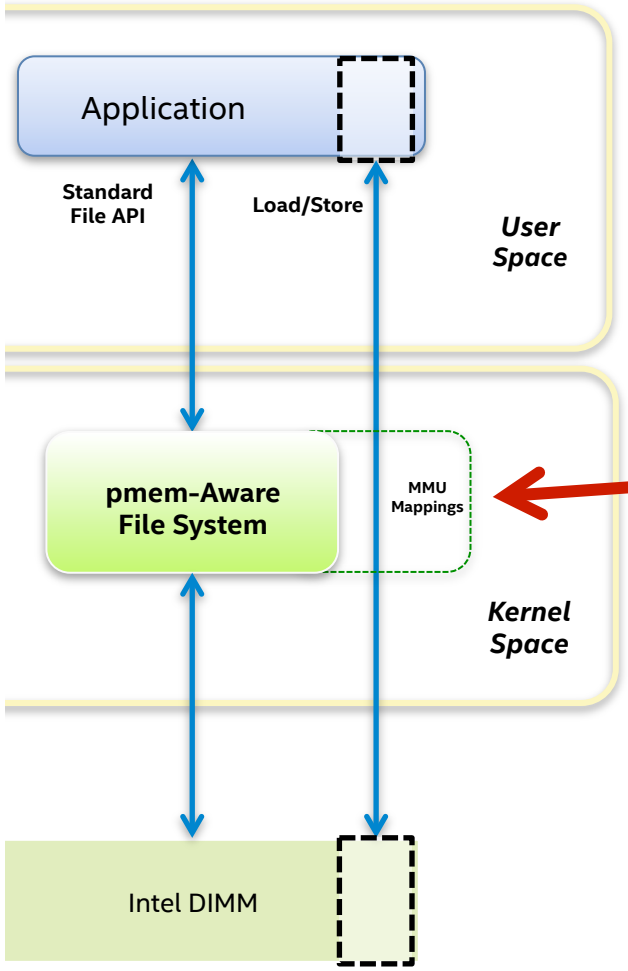
## Immediately Available

**High Performance**
Cell and array architecture that can switch states 1000x faster than NAND

# SOFTWARE ARCHITECTURE



# WHERE WRITES ARE CACHED



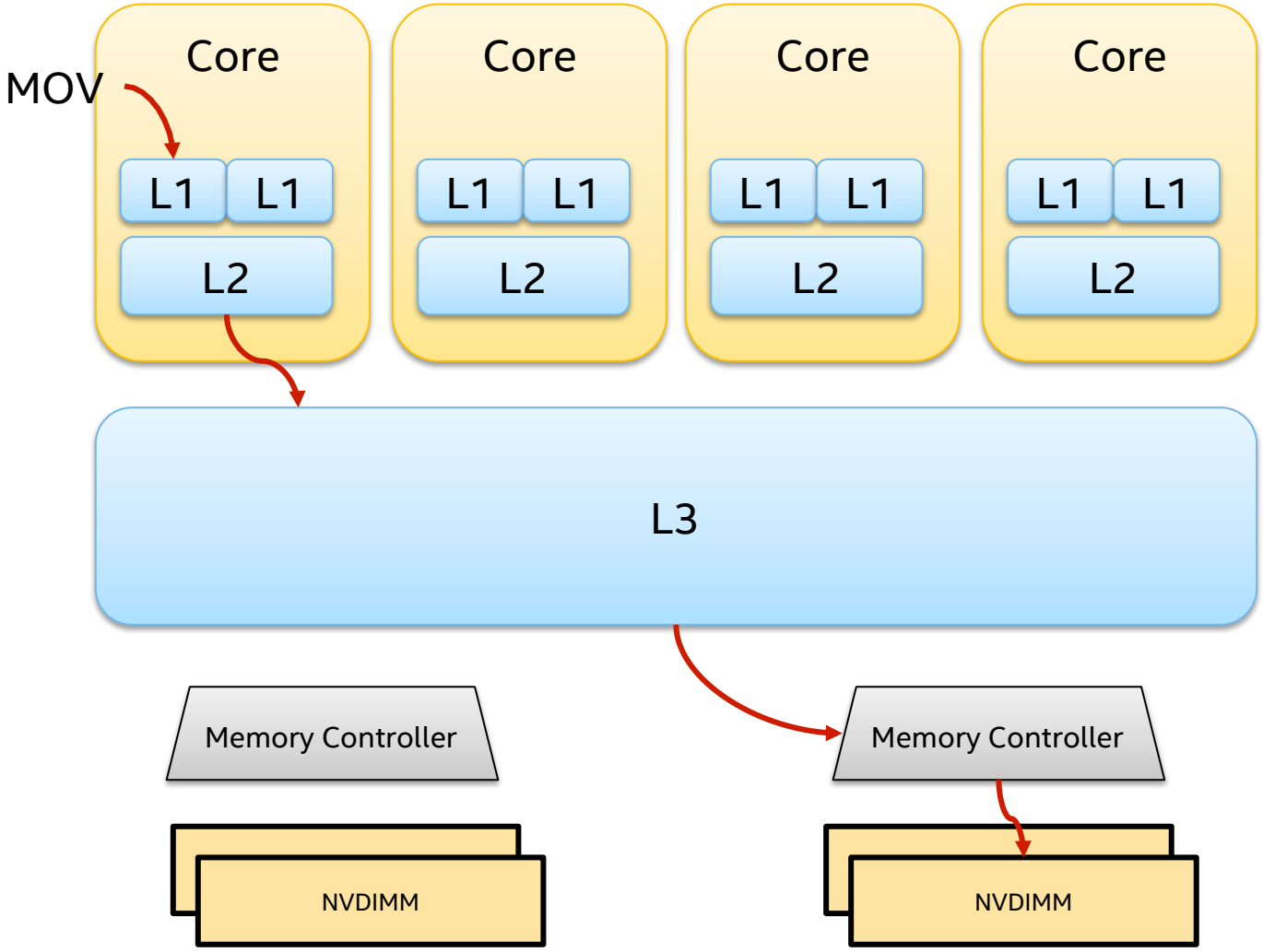

No Page Cache

```
msync()
FlushViewOfFile()
```

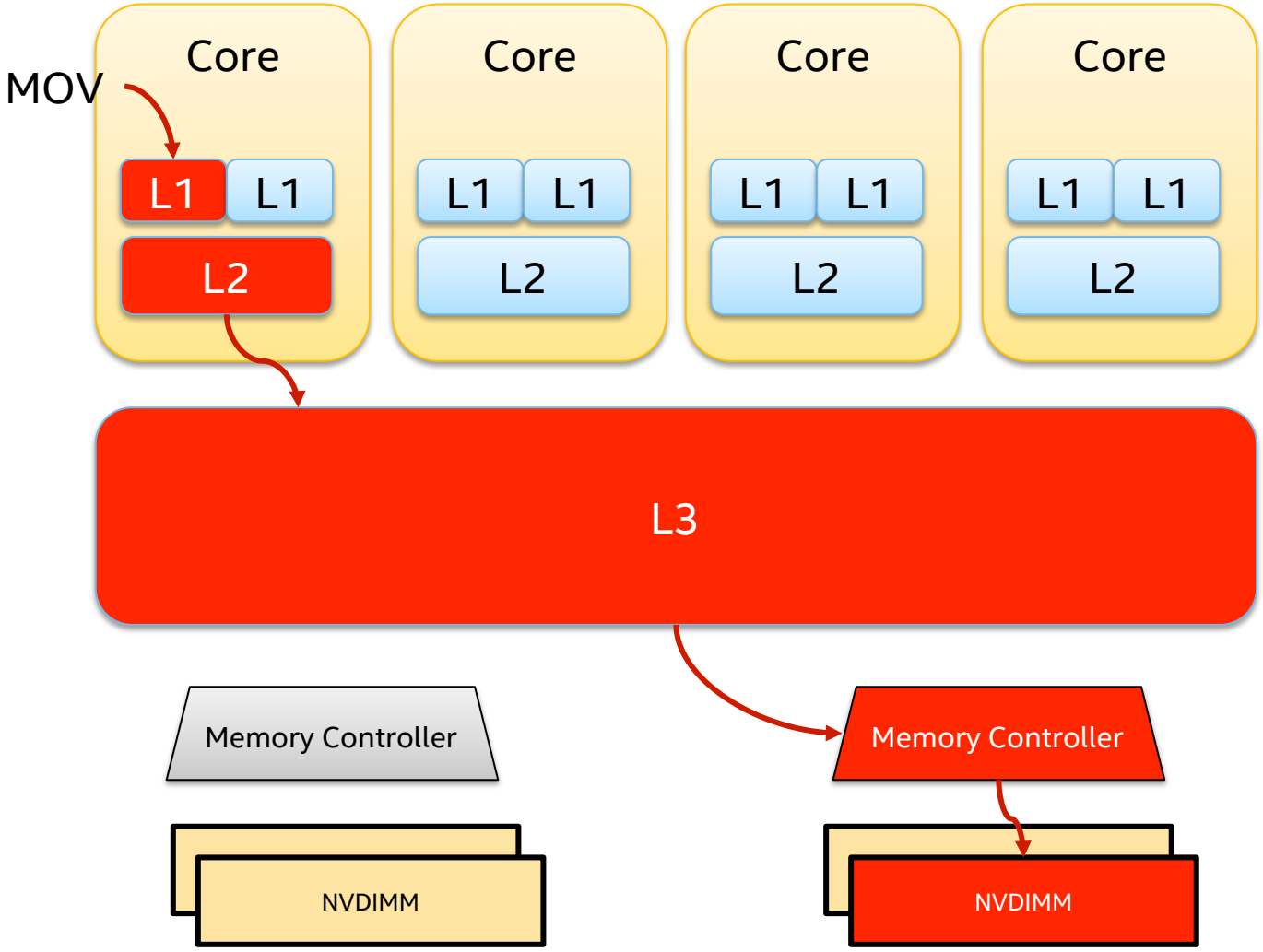
↓

```
pmem_persist()
```

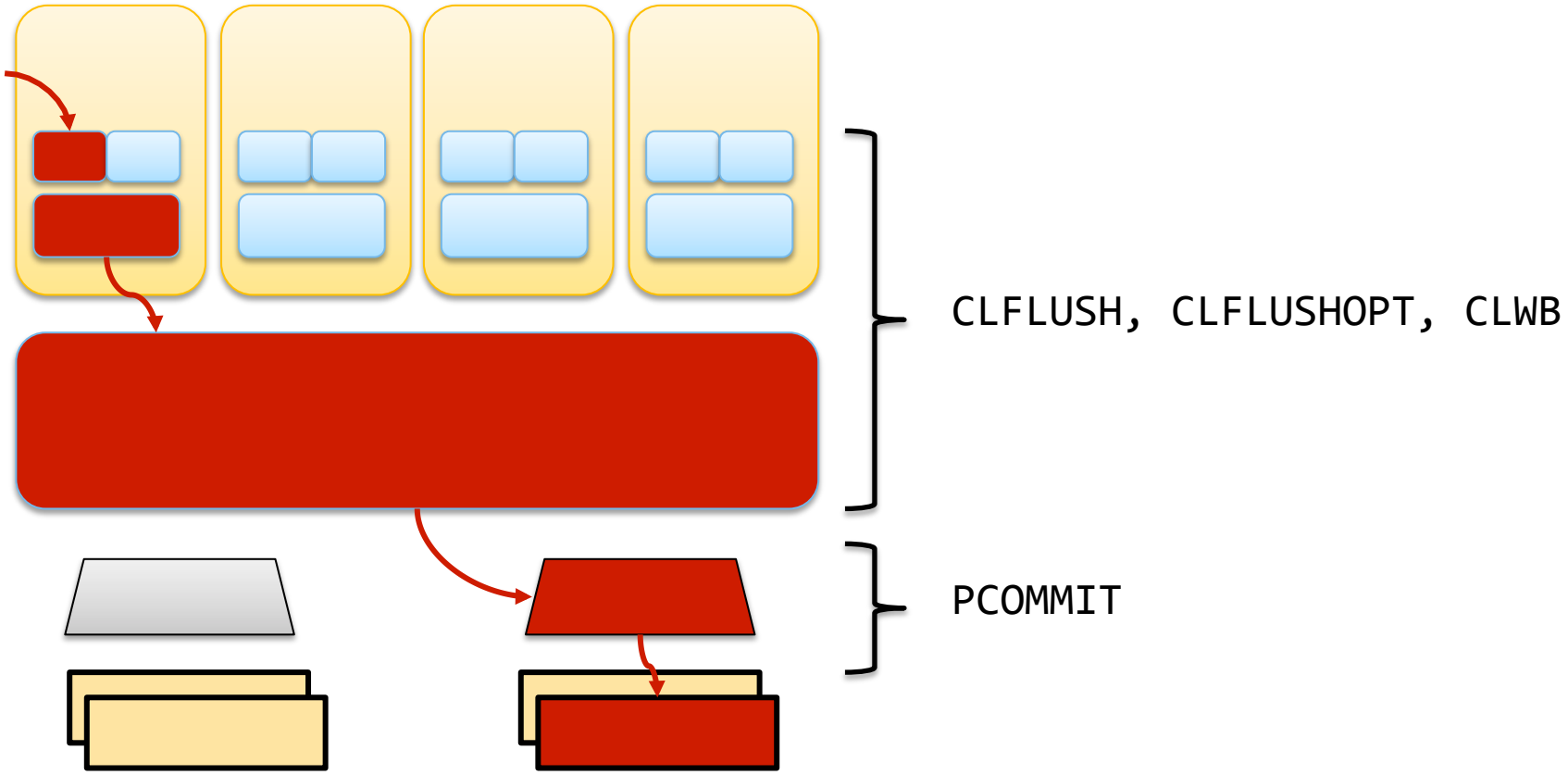
# THE DATA PATH



# HIDING PLACES



# TWO LEVELS OF FLUSHING WRITES



# FLUSHING WRITES FROM CACHES

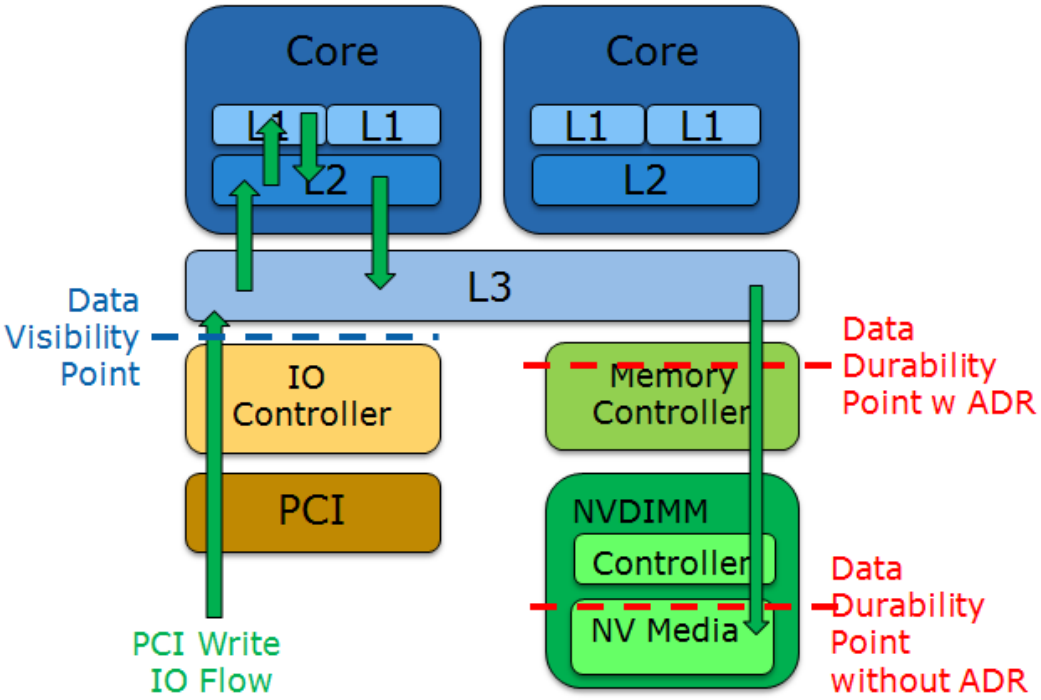
| Instruction | Meaning |
| --- | --- |
| `CLFLUSH addr` | Cache Line Flush: Available for a long time |
| `CLFLUSHOPT addr` | Optimized Cache Line Flush: New to allow concurrency |
| `CLWB addr` | Cache Line Write Back: Leave value in cache for performance of next access |

# FLUSHING WRITES FROM MEMORY CONTROLLER

| Instruction | Meaning |
| --- | --- |
| `PCOMMIT` | Persistent Commit: Flush stores accepted by memory subsystem |
| `Asynchronous DRAM Refresh` | Flush outstanding writes on power failure Platform-Specific Feature |

# RDMA WITH PMEM – VISIBILITY VS DURABILITY

- Write Data Visibility Point – Once data makes it out of the IO controller, HW makes the data visible to all caches and cores, independent of DRAM or NVDIMM devices
- Write Data Durability Point – Typically data is not considered durable until it has been successfully written by the Memory Controller to persistent media



# BLOCK MODE

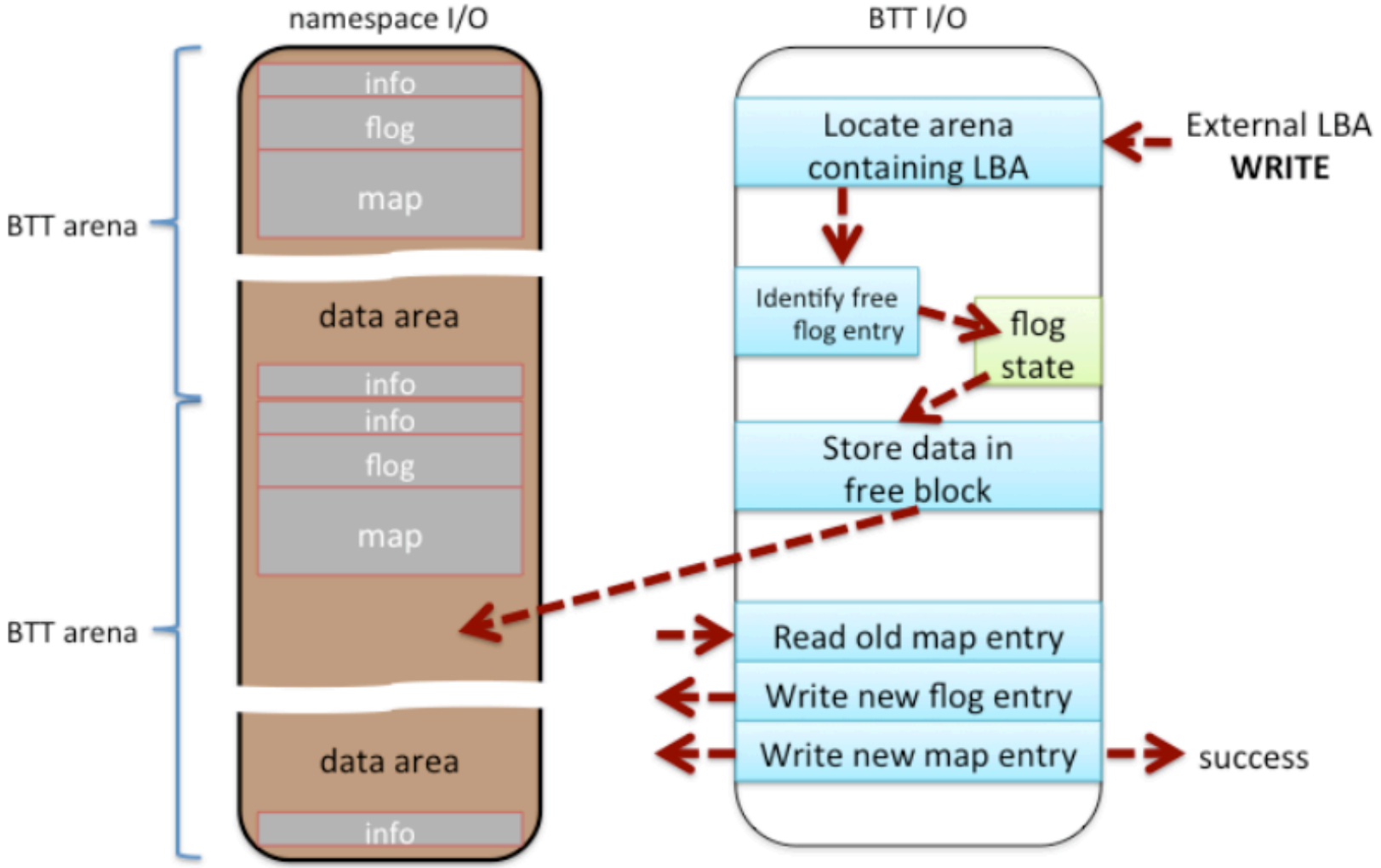
## Persistent memory block mode needs to protect against torn sectors

- Torn sectors = sectors partially written at power failure

## BTT = Block Translation Table

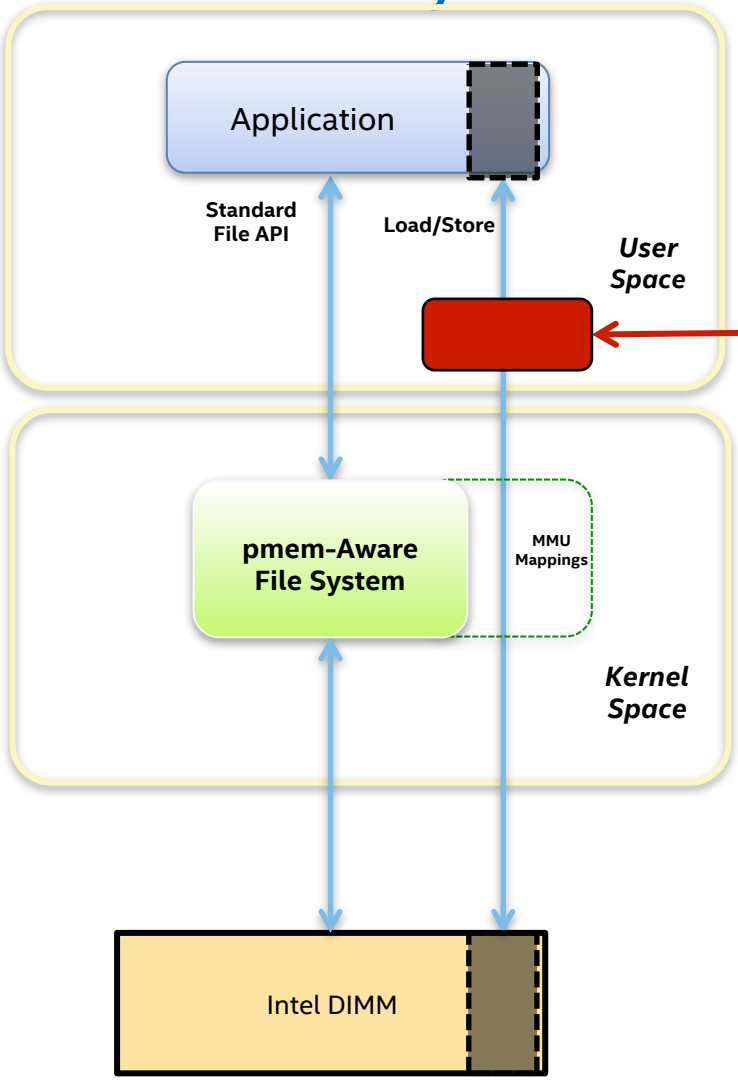
- Block namespace broken up into 512GB BTT “arenas”
- Arena contains:
  - Info block
  - Data area
  - LBA map
  - Flog (free block list + log)

# BLOCK MODE



# NVM LIBRARY: PMEM.IO

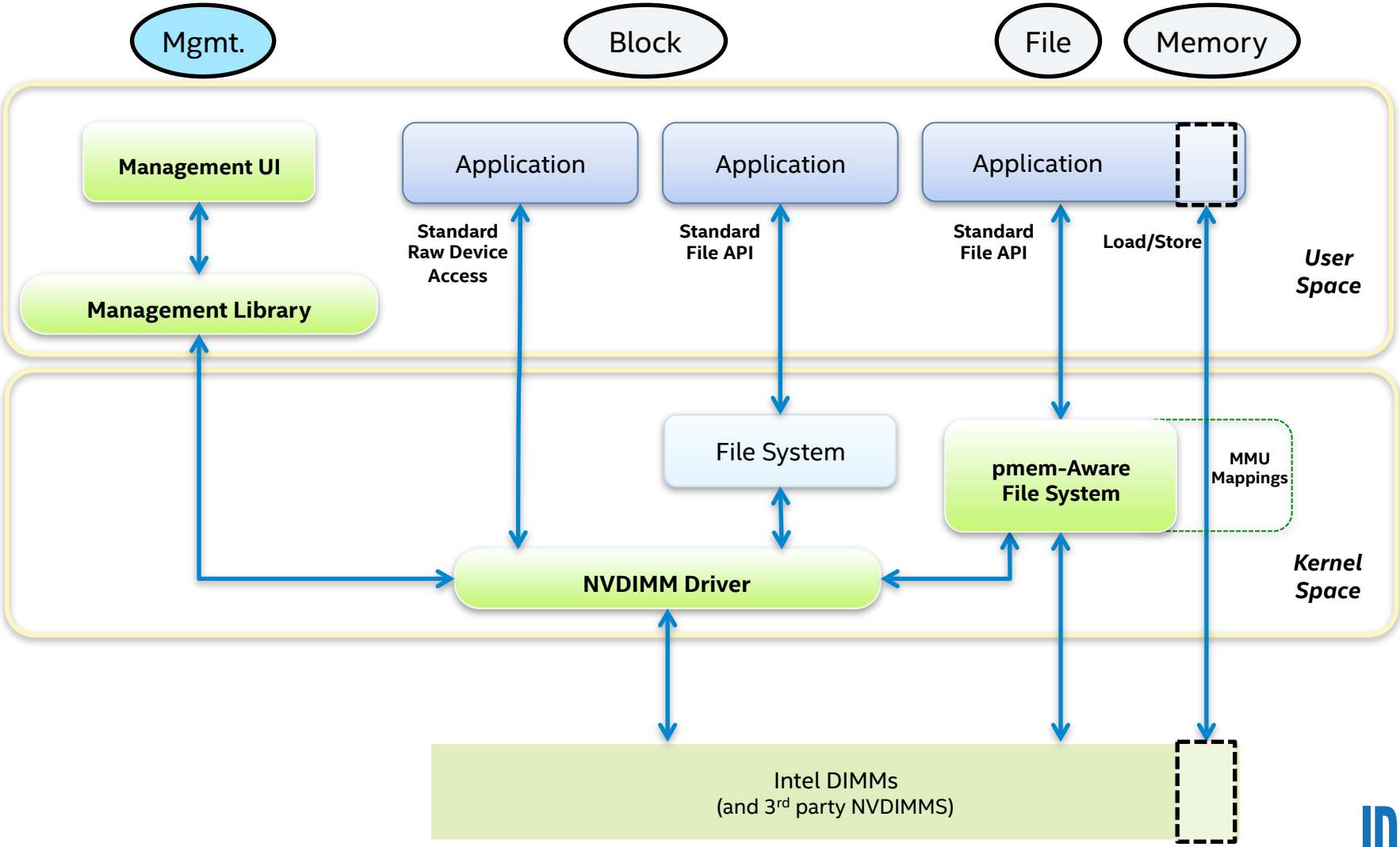
## 64-bit Linux* Initially




- Open Source
  - http://pmem.io
- libpmem
- libpmemobj
- libpmemblk
- libpmemlog
- libvmem
- libvmmalloc

Transactional

# SOFTWARE ARCHITECTURE



# Open Discussion

# FOR MORE INFORMATION...

- SNIA NVM Programming Model
  - http://www.snia.org/forums/sssi/nvmp
- Intel® Architecture Instruction Set Extensions Programming Reference
  - https://software.intel.com/en-us/intel-isa-extensions
- Open Source NVM Library work
  - http://pmem.io
- Linux* kernel support & instructions
  - https://github.com/01org/prd

# EVEN MORE INFORMATION…

- ACPI 6.0 NFIT definition (used by BIOS to expose NVDIMMs to OS)
  - http://www.uefi.org/sites/default/files/resources/ACPI_6.0.pdf
- Open specs providing NVDIMM implementation examples, layout, BIOS calls:
  - http://pmem.io/documents/
- Google* group for pmem programming discussion:
  - http://groups.google.com/group/pmem
- SNIA NVM PM Remote Access for High Availability white paper:
  - http://www.snia.org/sites/default/files/NVM%20PM%20Remote%20Access%20for%20High%20Availability%20V04R1.pdf

# CALL TO ACTION

- Stay abreast of developments in the NVM programming community
- Identify how you can make use of persistent memory
- Attend the FreeBSD Storage Summit