LWPMFS: Light-Weight Persistent Memory Filesystem

Ravi Pokala (aka rpokala@freebsd.org)

FreeBSD Developer Summit, 2018-10-18

Who am I? Why am I here?

Who am I? Why am I here?

panasas
Beyond Storage

- Panasas – High-Performance Storage Appliances
  - Founded in 1999; grew out of research from Carnegie Mellon University's Parallel Data Lab
  - Clustered storage – data spread across lots individual servers
    - Custom hardware platform: blades, power-supplies, integrated network switches
  - Native DirectFlow Protocol (kernel modules for Linux and macOS)
    - Direct parallel access to storage nodes
  - Legacy Protocols (NFSv3, CIFSv2)
    - Access goes through protocol gateway, which does parallel access behind the scenes

Who am I? Why am I here?

panasas
Beyond Storage

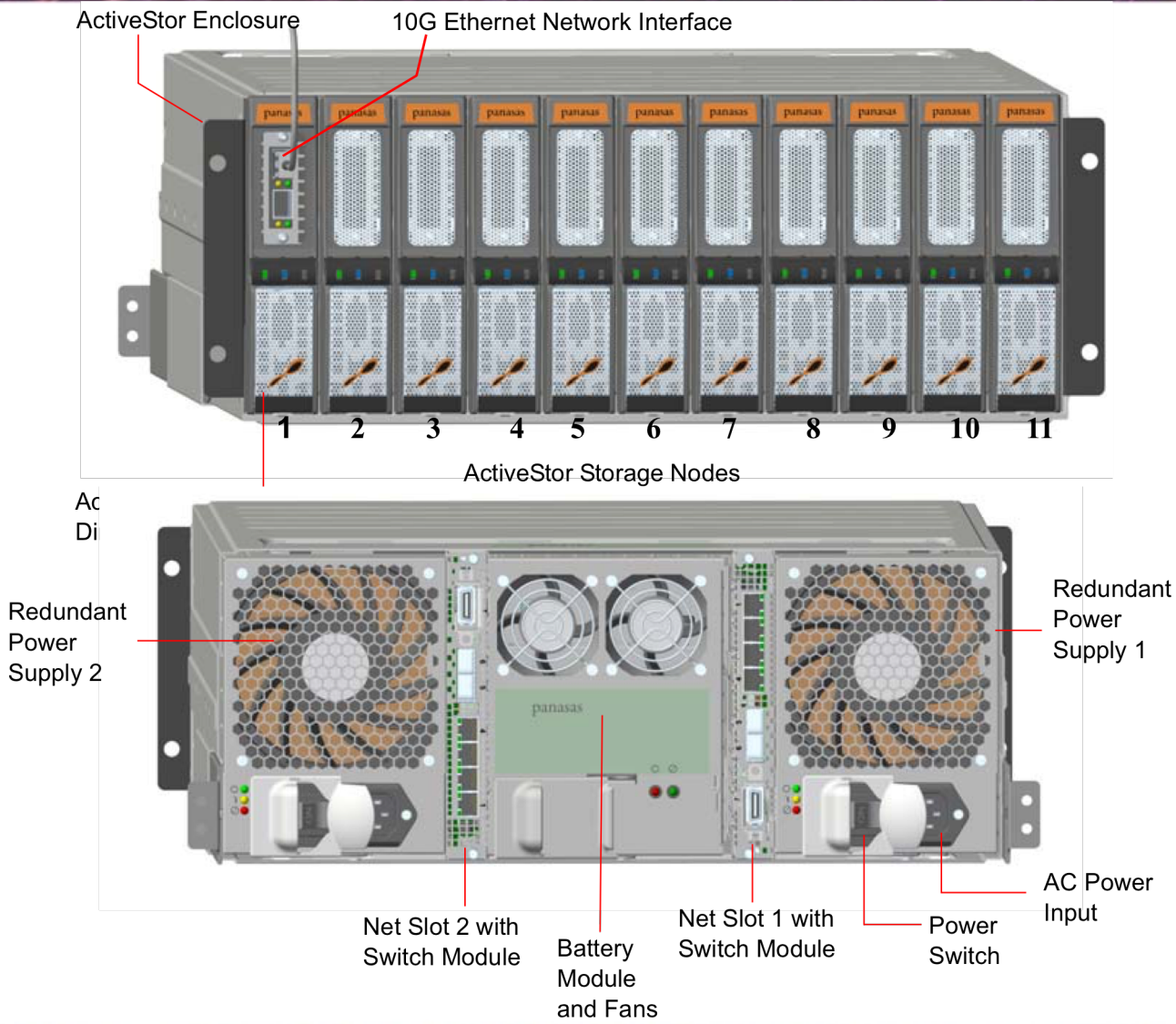
- Ravi Pokala – rpokala@freebsd.org
  - Joined Panasas in late 2002; various hardware-facing hacking ever since
  - FreeBSD src-committer since 2015-11
- Caveat: I am not the original author of what is being presented
  - I inherited the bottom levels of the stack
  - Colleague now maintains the higher levels
    - Scheduling conflict – could not attend to present this himself
  - Take open questions to freebsd-hackers@freebsd.org mailing list

Power-failure and Panasas

Power-failure and Panasas: Problem

- Problem: High-Performance Storage requires caching
  - HDD caches and motherboard DRAM are both volatile
    - Loss of power => loss of data
  - First-generation Panasas hardware was designed about 17 years ago
    - Few good non-volatile memory options
      - PCI card w/ battery-backed DRAM
      - ???
    - PCI is slow compared to main system DRAM
    - PCI cards were too big for our custom blade form-factor
    - When the battery is drained, data is lost
    - Battery can't be hot-swapped
    - Does not help with drive cache contents

Power-failure and Panasas: Custom Hardware



Power-failure and Panasas: Solution

- Solution: Shared enclosure-wide battery module
  - When both power-supplies lose AC input, switch to DC input from the battery
    - De-assert AC_GOOD signal
  - Blades see un-interrupted DC input
  - Entire blade – CPU, system DRAM, HDDs, drive caches – stays online
  - Software detects that both AC_GOODs are de-asserted => running from battery
  - Expedited power-fail shutdown path
    - Stream out of DRAM to reserved space; defer proper placement until reboot after AC input is restored
  - One battery module per enclosure
    - Hot-swappable
    - Relatively cheap sealed lead-acid cells (like a motorcycle battery)

PoPower-failure and Panasas: Imperfect Solution

panasas
Beyond Storage

- Not without problems
  - Custom battery module requires custom monitoring and management
    - System has to perform safe, automated testing
  - Lead-acid cells are big and heavy
    - Expensive to ship
    - Hard to replace at top-of-rack
  - More recently: shipping restrictions
    - Lead is toxic
    - Batteries can be dangerous => more padding, more paperwork

Power-failure and Panasas: 2012 – COTS?

panasas
Beyond Storage

- Fast-forward: 2012
- Move protocol gateways and metadata management to off-the-shelf hardware
  - Faster, cheaper adoption of new CPUs and networking
- Still need a power-fail solution
  - Memory vendor pitched new idea called "NV-DIMM"
    - Connected to normal DDR3 memory bus
    - Looks like DRAM, because it's made from DRAM ...
      - Normal DDR3 speeds, normal DRAM latencies
      - Byte addressable
    - ... until power fails ...

Power-failure and Panasas: 2012 – NVDIMM?

panasas
Beyond Storage

- Theory of operation
  - Power-supply raises signal when AC is lost
  - NV-DIMM detaches from DDR3 bus, self-refreshes using supercap
  - Automatically copies DRAM contents into NAND
- Pros
  - Everything we like about main memory
- Cons
  - Only supported by specific motherboards and power-supplies
  - Required non-standard motherboard firmware
  - Supercaps are large, and their lifespan is unproven

Power-failure and Panasas: 2012 – No NVDIMM

panasas
Beyond Storage

- Too much science-experiment, not enough product
  - Plenty of other work to do while the technology solidified
- COTS experiment was very educational, but never shipped

Power-failure and Panasas: 2015 – COTS Product

panasas
Beyond Storage

- 2015: Building on previous experiments, create a shippable COTS product
- DDR4 DRAM+NAND NVDIMMs are real products
  - Supercap packs are smaller
  - Supercap lifetimes better characterized
  - Standardized by JEDEC and ACPI
    - ... in mid-2015
  - Widely supported by motherboards, power-supplies, and firmware
    - ... by 2017

Power-failure and Panasas: 2015 – COTS Product

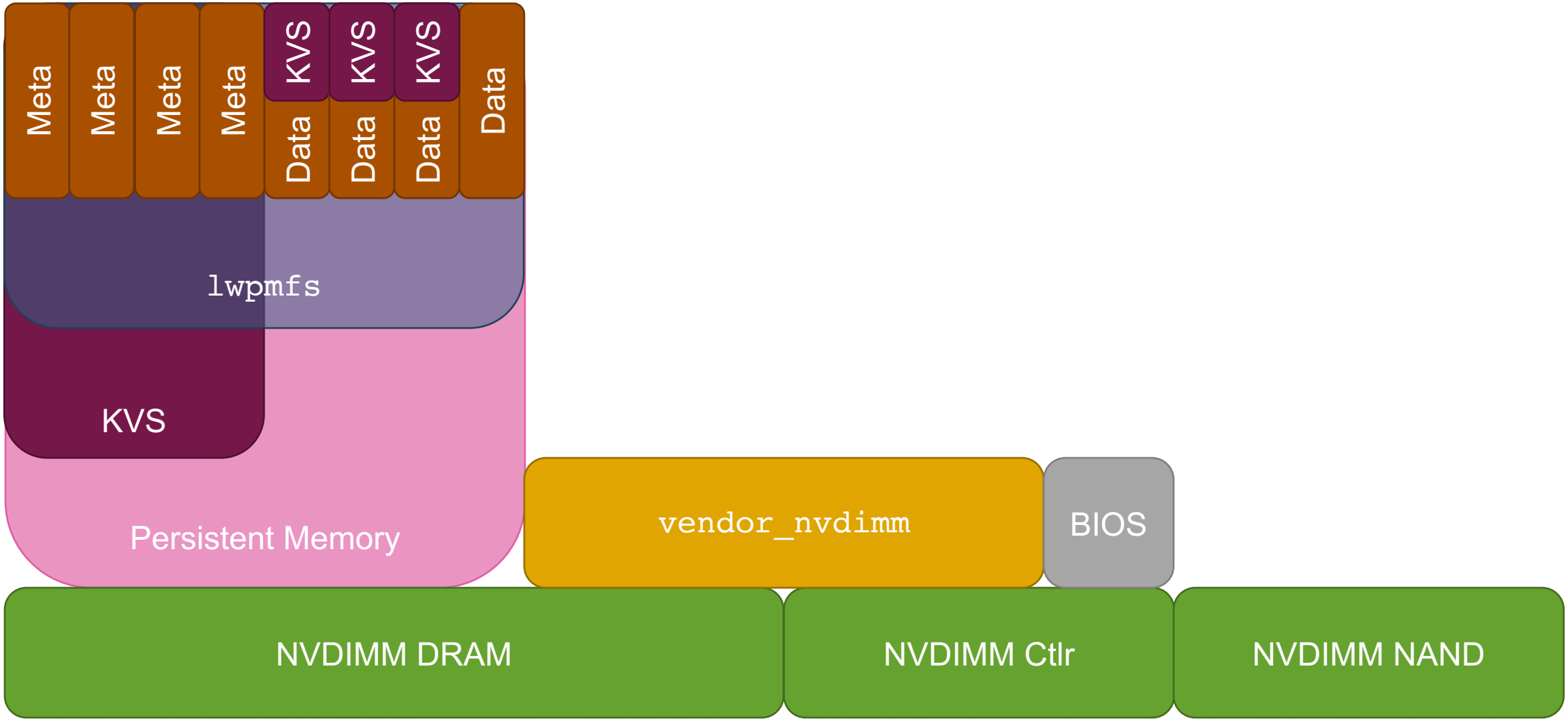
panasas
Beyond Storage

- In early 2015, NVDIMM products were real, but not-quite-standard
  - Board firmware supported proprietary APIs from two NVDIMM vendors
  - Several board and NVDIMM firmware upgrades required for stable functionality
- Long lead times for logistics, development, testing, etc
  - Couldn't just wait for JEDEC / ACPI to be adopted

NVDIMM Support

panasas
Beyond Storage

NVDIMM Support – Full Stack



NVDIMM Support – Low-level (Proprietary)

panasas
Beyond Storage

- Motherboard communicates with NVDIMM over SMBus
  - Just like SPD EEPROM or JEDEC temperature sensor
- Driver for SMBus controller (part of memory controller (part of CPU))
  - Eventually committed as `imcsmb(4)` ({Sandy,Ivy}bridge-Xeon and {Broad,Has}well-Xeon)

NVDIMM Support – Low-level (Proprietary)

panasas
Beyond Storage

- Driver for NVDIMM vendor proprietary interface
  - Ported from Linux driver
  - Sources had a compatible license, but were only accessible under NDA
  - Only useful for this particular pre-standard NVDIMM, so no point in upstreaming anyway
  - Parts were extracted and committed as `jedec_dimm(4)`
- NVDIMM DRAM exposed as byte-addressable `/dev/nvdimm` device node
  - `read(2)` and `write(2)` do `uiomove(9)`
  - `write(2)` includes instructions to flush the appropriate cachelines
    - `CLFLUSHOPT` isn't available on {Broad,Has}well, so simulated with `MOVQ` / `MOVNTI` / `SFENCE`
- `ioctl(4)`s to arm and disarm auto-save, check supercap health, etc.

NVDIMM Support -- Persistent Memory

panasas
Beyond Storage

- Abstract interface
  - Byte-addressable
  - Memory-mapped
  - Contiguous virtual address space
  - Explicit write-back caching semantics
  - Explicit fencing around transactions
  - Persistence only guaranteed after synchronous commit
  - Writes between commits may be re-ordered

NVDIMM Support -- Persistent Memory

panasas
Beyond Storage

- Three implementations
  - Direct – PM backed by `/dev/nvdimm` device – base class
    - Commit / fence: `sfence` (if needed)
    - Writeback: Same `MOVQ` / `MOVNTI` / `SFENCE` tricks as NVDIMM driver
  - `mmap` – PM semantics on mmap-ed user-space file
    - PM backed by file in `lwpmfs`
    - Commit: `fsync(2)`
    - Writeback: `msync(2)`
  - `kmmap` – PM semantics on vnode object
    - PM backed by file in `lwpmfs`, from inside kernel
    - Commit: `VOP_FSYNC(9)`
    - Writeback: `vm_map_sync(9)`

Buffer-cache bypass mode:
Explicit writeback not required

NVDIMM Support -- Persistent Memory

panasas
Beyond Storage

- Buffer-cache bypass
  - VOP_GETPAGES() adjusts mappings, does not do memcpy()s
  - VOP_PUTPAGES() not called

NVDIMM Support – KVS

panasas
Beyond Storage

- Key-Value Store
  - 64-bit keys, Arbitrary-sized values
  - Multiple K/V pairs can be updated atomically (transactional)
- Values are copy-on-write
  - Refcounted sparse tree of block pointers and checksums
  - Never modifies a referenced block in-place
- Create-time parameters
  - Maximum number of K/V pairs
  - Maximum number of modifications in a single transaction
  - Number and size (at least 16 bytes, power-of-two) of blocks for value data

NVDIMM Support – KVS

panasas
Beyond Storage

- K/V pair metadata (record)
  - Key, value
  - Transaction ID
  - Transaction size
  - Flags
  - Checksum
- KVS reserves spaces for N (max pairs) + M (max in-flight txns) records

NVDIMM Support – KVS

panasas
Beyond Storage

- Update single record
  - Find free record
  - Increment transaction ID
  - Update record data
  - Update record checksum
  - Power-fail during modification?
    - Record being modified has invalid checksum
    - Previous version of record is still present; use it

NVDIMM Support – KVS

panasas
Beyond Storage

- Update multiple records in a transaction
  - Just like updating a single record, but all records being updated will share a transaction ID
  - Power-fail during modification?
    - Transaction has fewer valid records than specified
    - Throw out all records modified by the transaction
- Reads & concurrent updates
  - Increment refcount on value's radix tree
  - If value is modified w/ non-zero refcount, leaf blocks (and indirect blocks) are copied
  - Copies are modified and marked dirty
  - When modification is committed, only dirty blocks need to be committed

NVDIMM Support – LWPMFS

panasas
Beyond Storage

- LWPMFS is built on top of both KVS and PM
  - `newfs_lwpmfs` splits underlying device into a KVS for metadata, and blocks for data
  - Elaborate code to estimate proper KVS sizing – some assumptions based on our use-case
  - Remainder of device used for LWPMFS data blocks
- LWPMFS metadata stored in KVS
  - Key=inode
  - Value
    - Generic FS metadata (size, times, parent, permissions, etc)
    - List of FS data blocks
- LWPMFS data stored outside of KVS

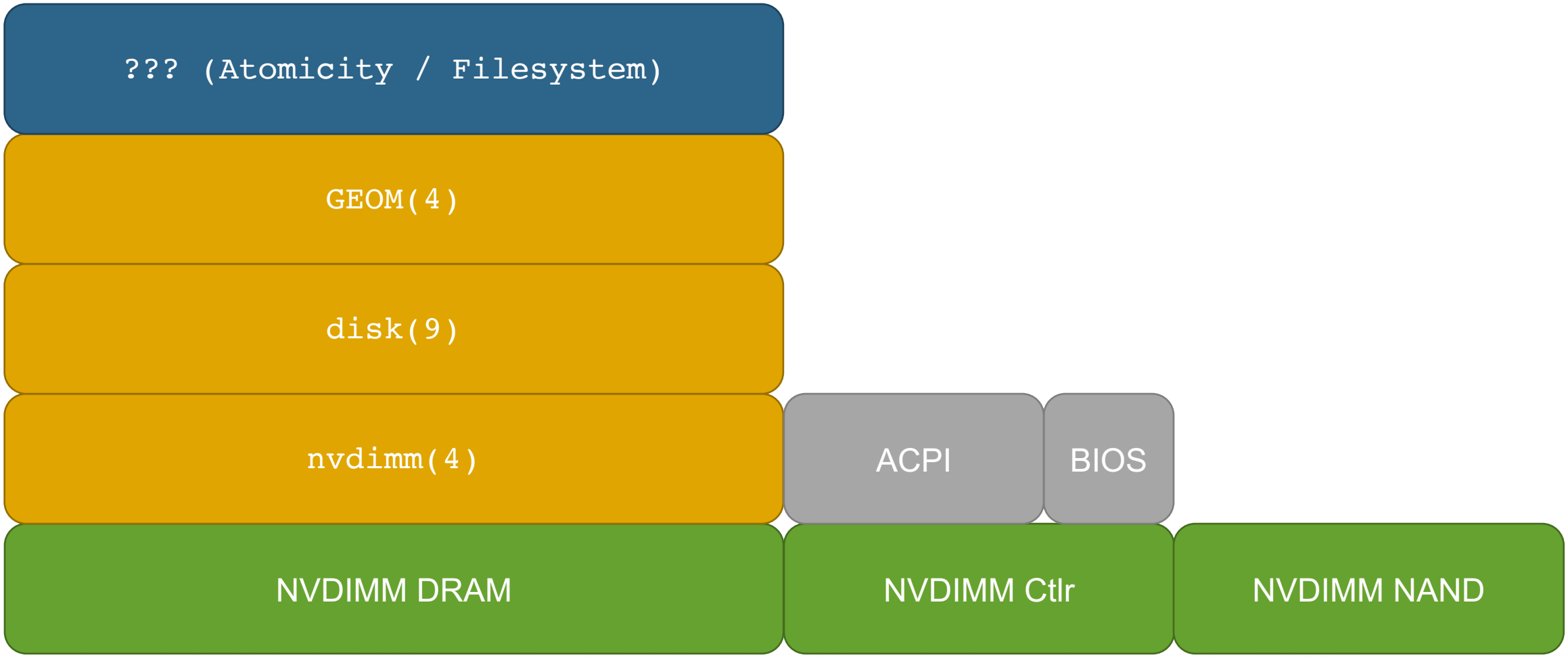
Standard NVDIMM Support

Standard NVDIMM Support – In-tree

panasas
Beyond Storage

- Konstantin Belousov (Kostik, aka kib@freebsd.org) has been working on this
- Two related commits earlier this week
  - r339386 – pmap_large_map() KPI
    - map very large contiguous physical regions
      - With the size of NVDIMM that we used, and our use-case, this was not an issue that we needed to address
    - Optimized cache flushing
      - `CLFLUSHOPT` when possible, falling back to `CLFLUSH`
        - Our `MOVQ` / `MOVNTI` / `SFENCE` trick might be better than `CLFLUSH`? Need to discuss
  - r339391 – JEDEC / ACPI NVDIMM device driver
    - Uses ACPI “NFIT” table to discover and enumerate NVDIMMs
    - Hooks into `disk(9)`, `geom(4)`

Standard NVDIMM Support – Full Stack



Standard NVDIMM Support – Still needed

panasas
Beyond Storage

- Something to enforce atomicity and prevent partial updates
- DAX (Direct Access) filesystem (buffer-cache bypass)
- Some of these things are available through PMDK
  - Intel Persistent Memory Development Kit
  - Why didn't Panasas use that in the first place?
    - At the time, it didn't cover all our needs, and might not have supported FreeBSD at all?
    - Now PMDK does support FreeBSD (but not in ports yet...)
- Panasas PM / KVS / LWPMFS code is probably not directly useful upstream
  - But we're happy to share it, if it will help fill the gaps!


panasas
Beyond Storage
Q & A

Links

- https://people.freebsd.org/~rpokala/2018-10-DeveloperSummitLWPMFS.pdf