SDC

STORAGE DEVELOPER CONFERENCE

SNIA SANTA CLARA, 2017

All About Persistent Memory Flushing

Andy Rudoff, Intel

Steve Byan, Oracle

By the End of This Talk, You Will Know…

- Why flushing matters for persistent memory
  - And when you don't have to worry about it
- The difference between:
  - Visibility and persistence
  - msync/fsync, FlushFileBuffers
  - Optimized Flush
  - x86 instructions: CLFLUSHOPT/CLWB
  - Deep Flush
  - Flushing to remote persistence

2017 Storage Developer Conference. © Intel and Oracle. All Rights Reserved.

Why Flushing Matters

2017 Storage Developer Conference. © Intel and Oracle. All Rights Reserved.

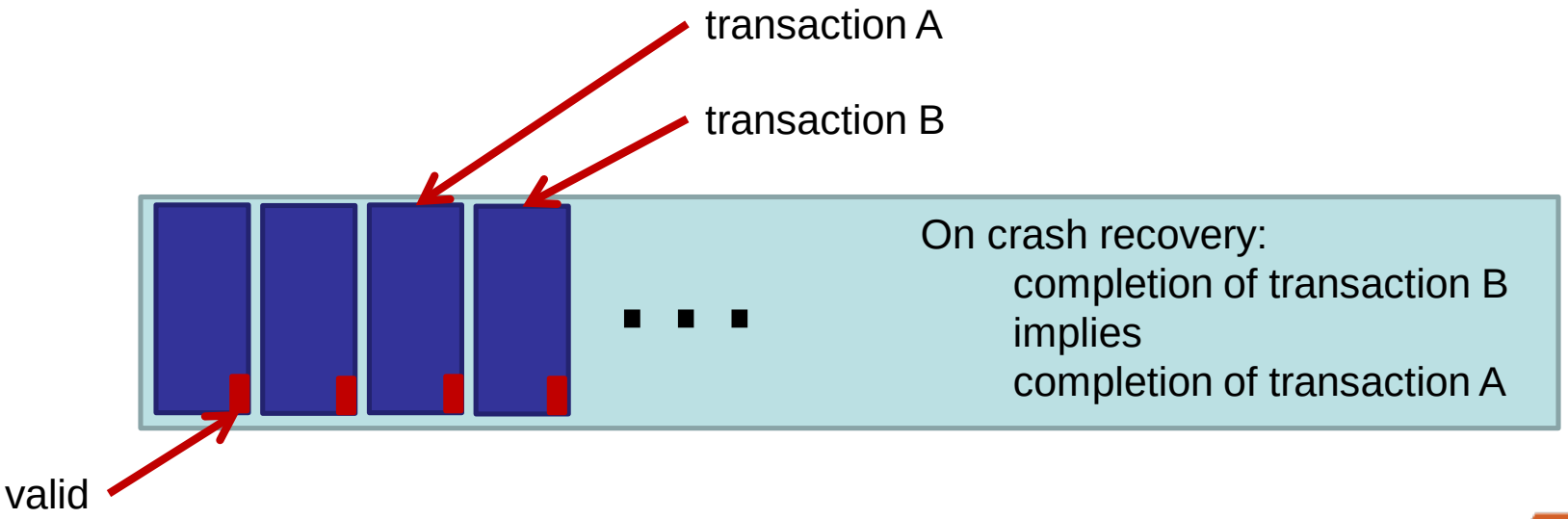
~~Why Flushing Matters~~

Why Store Barriers Matter

2017 Storage Developer Conference. © Intel and Oracle. All Rights Reserved.

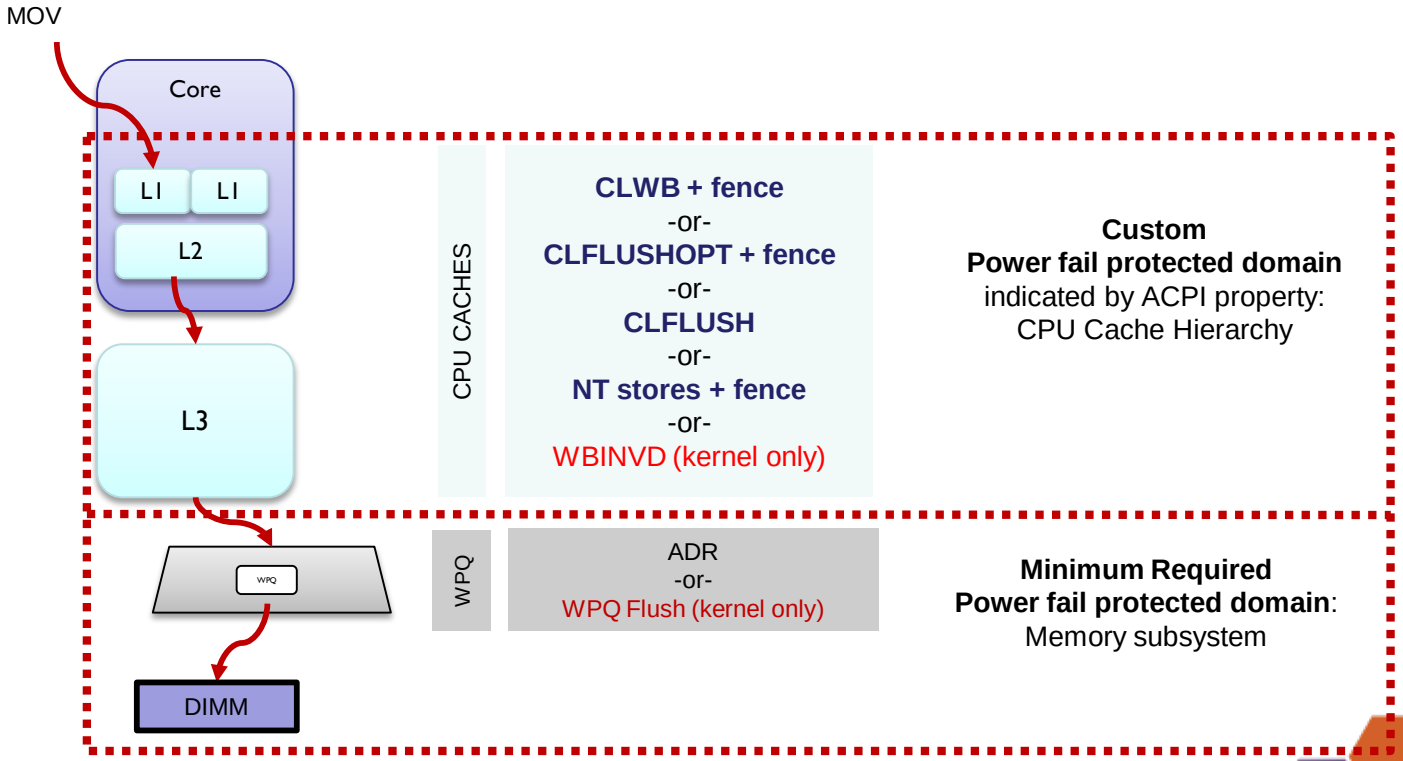
~~Why Flushing Matters~~

Why Store Barriers Matter



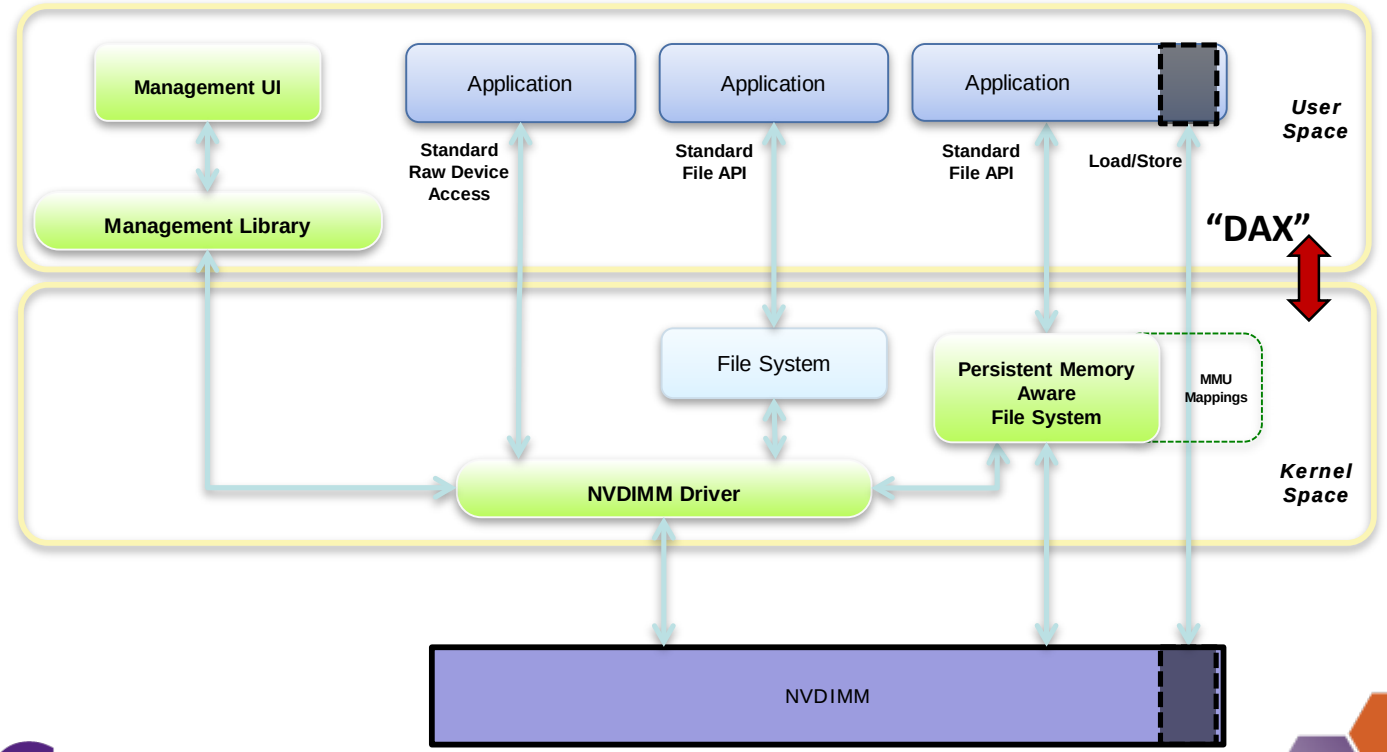
2017 Storage Developer Conference. © Intel and Oracle. All Rights Reserved.

How the Hardware Works



2017 Storage Developer Conference. © Intel and Oracle. All Rights Reserved.

How the Software Stack Works



2017 Storage Developer Conference. © Intel and Oracle. All Rights Reserved.

Flushing for Application Programmers

- Is the flushing requirement new for pmem?
  - Memory-mapped files have always worked this way:
    - Stores are not guaranteed persistent until flush API is called
    - Stores are *visible* before they are persistent
- Do standard flushing APIs work with pmem?
  - Yes, standard APIs work as expected
    - `msync()` on Linux
    - `FlushFileBuffers()` on Windows
    - The kernel will use instructions like CLWB as necessary
- Can Applications just flush with CLWB from user space
  - Only when supported by the kernel/file system
  - Libraries like NVML determine when it is safe

2017 Storage Developer Conference. © Intel and Oracle. All Rights Reserved.


Definitions

- Standard Flush
  - msync/fsync/FlushFileBuffers
  - 30+ year-old interfaces, same semantics
- Optimized Flush
  - Optionally-supported, faster flushing
  - Can avoid locks, kernel calls, rounding up

2017 Storage Developer Conference. © Intel and Oracle. All Rights Reserved.

More Definitions

- Deep Flush
  - Flush to smallest failure domain available to SW
  - Gives up performance for higher RAS
- Remote Flush
  - A store barrier for RDMA to pmem

2017 Storage Developer Conference. © Intel and Oracle. All Rights Reserved.

State of Ecosystem Today

OS Detection of NVDIMMs	**ACPI 6.0+**
OS Exposes pmem to apps	***DAX* provides SNIA Programming Model** Fully supported: • Linux (ext4, XFS) • Windows (NTFS)
OS Supports Optimized Flush	Specified, but evolving (ask when safe) • Linux: **unsafe** except Device DAX • (and new file systems like **NOVA**) • Windows: **safe**
Remote Flush	Proposals under discussion (works today with extra round trip)
Deep Flush	Upcoming Specification
Transactions, Allocators	Built on above via libraries and languages: • http://pmem.io **Much more language support to do**
Virtualization	All VMMs planning to support PM in guest (KVM changes upstream, Xen coming, others too…)

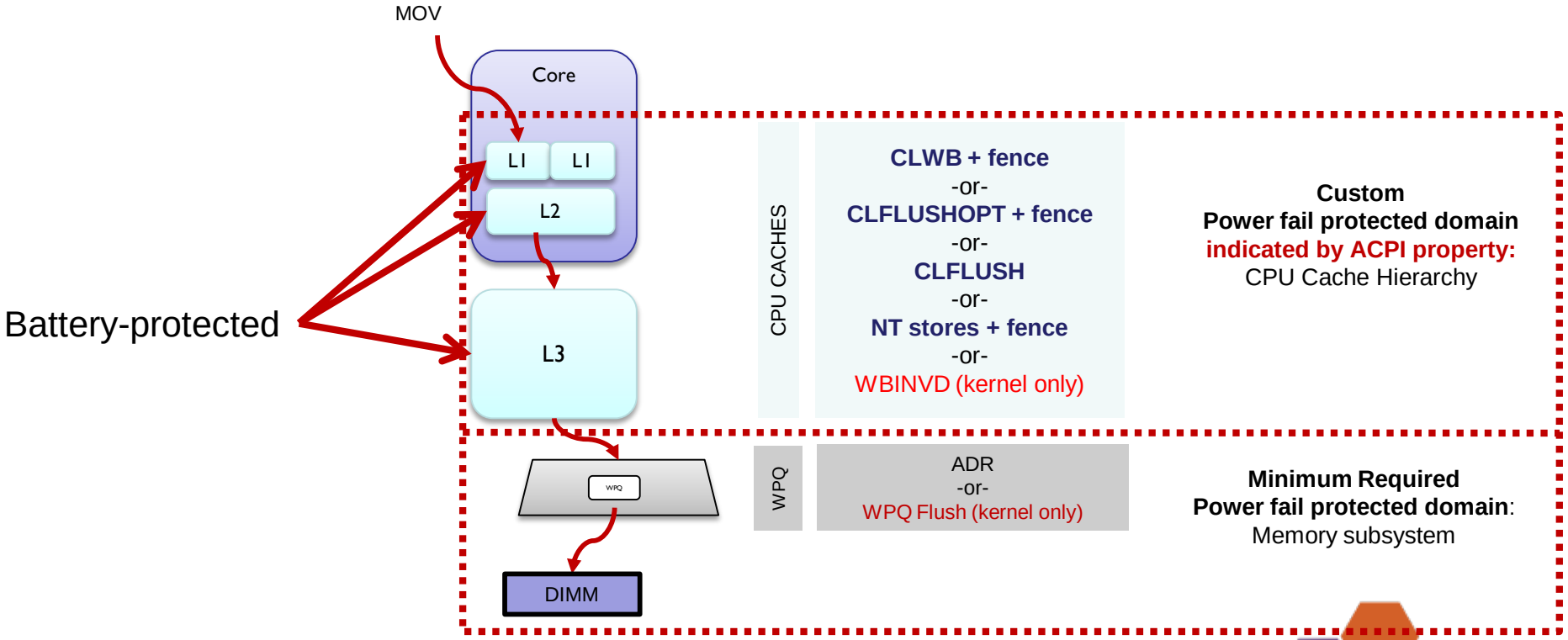
2017 Storage Developer Conference. © Intel and Oracle. All Rights Reserved.

Can Visibility == Persistence?

- Sure, but you need either
  - Write-through caches (performance impact)
  - Flush-on-fail caches (cost)

2017 Storage Developer Conference. © Intel and Oracle. All Rights Reserved.

Flush on Fail Caches



2017 Storage Developer Conference. © Intel and Oracle. All Rights Reserved.

Can Visibility == Persistence?

- Do volatile algorithms now “just work?”
  - CMPXCHG
    - Mostly yes
  - TSX
    - Mostly no
- The real issue:
  - Many algorithms have volatile assumptions!

2017 Storage Developer Conference. © Intel and Oracle. All Rights Reserved.

Can Visibility == Persistence?

- Bottom line:
  - Some custom platforms will have flush-on-fail
  - Maybe some day all platforms
  - Flushing is not going away soon though
  - Apps should honor the ACPI property

2017 Storage Developer Conference. © Intel and Oracle. All Rights Reserved.

When to use Deep Flush

- To limit metadata corruption from an ADR failure
  - Example: file system journal writes
    - ext4 or NTFS journal
- To give up performance for RAS
  - Example: today, some apps call `fsync()`
- Note that sometimes standard flush (`msync()` or `FlushViewOfFile()`) is fast enough, so just use it and get deep flush for free!

2017 Storage Developer Conference. © Intel and Oracle. All Rights Reserved.

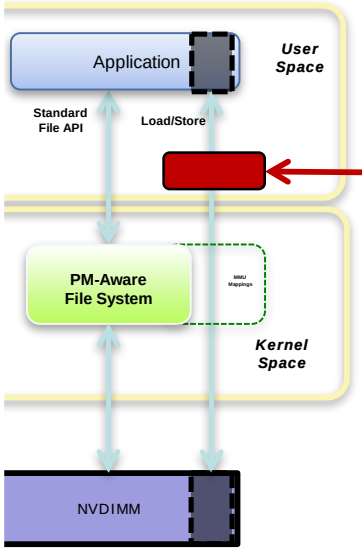
When to Not to use Deep Flush

- Performance critical
  - Remember: deep flush is meant to be used rarely
  - Optimized flush works fine unless HW failure
    - In the HW failure case, deep flush might fail too!

- Advice: writing apps to micromanage flush versus deep flush is a premature optimization

2017 Storage Developer Conference. © Intel and Oracle. All Rights Reserved.

Flushing via Libraries



NVM Libraries

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

Transactional

More libraries being added to the suite over time

2017 Storage Developer Conference. © Intel and Oracle. All Rights Reserved.

If You Feel You Must Flush Yourself

- Use libpmem to determine appropriate flushes
  - Either call it, or steal from it

- But consider using a library that handles flushing for you!

2017 Storage Developer Conference. © Intel and Oracle. All Rights Reserved.

Where Are The Flushes?

```
void push(pool_base &pop, uint64_t value) {
  transaction::exec_tx(pop, [&] {
    auto n = make_persistent<pmem_entry>();

    n->value = value;
    n->next = nullptr;
    if (head == nullptr) {
      head = tail = n;
    } else {
      tail->next = n;
      tail = n;
    }
  });
}
```

Transactional
(including allocations & frees)

2017 Storage Developer Conference. © Intel and Oracle. All Rights Reserved.

Where Are The Flushes?

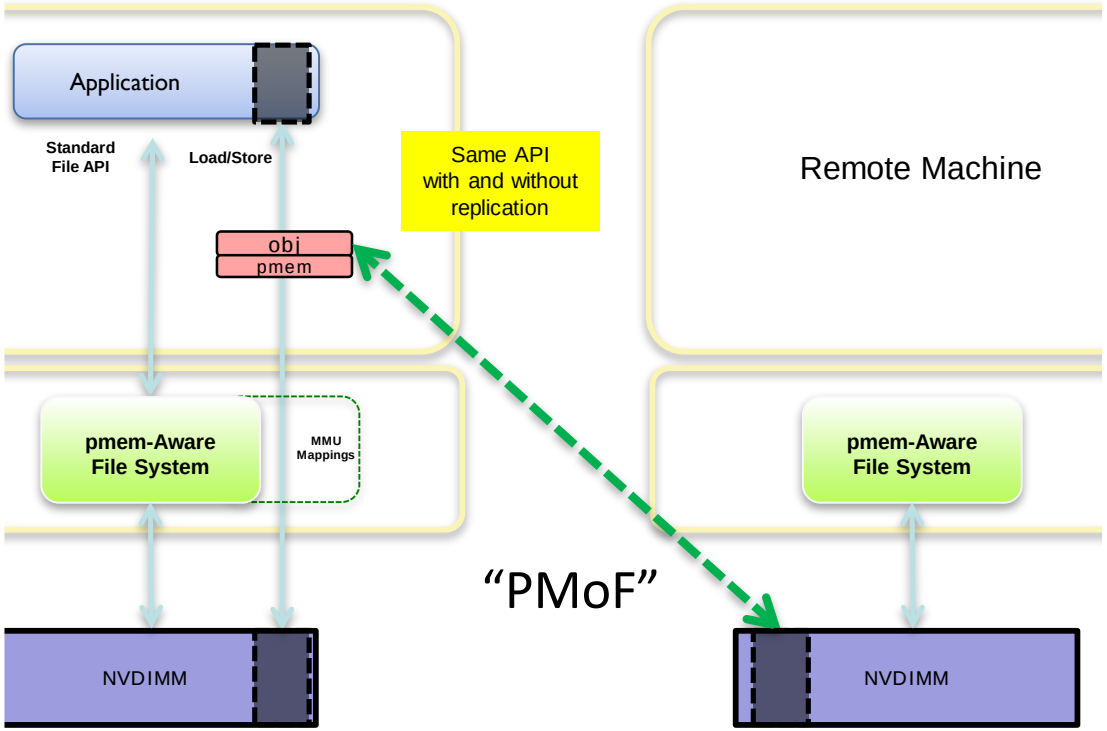
```
PersistentSortedMap employees = new PersistentSortedMap();

…
employees.put(id, data);
```

No flush calls.
Transactional.
Java library handles it all.

See `https://github.com/pmem/pcj`

2017 Storage Developer Conference. © Intel and Oracle. All Rights Reserved.

Libpmemobj Replication: Application Transparent
(except for performance overhead)



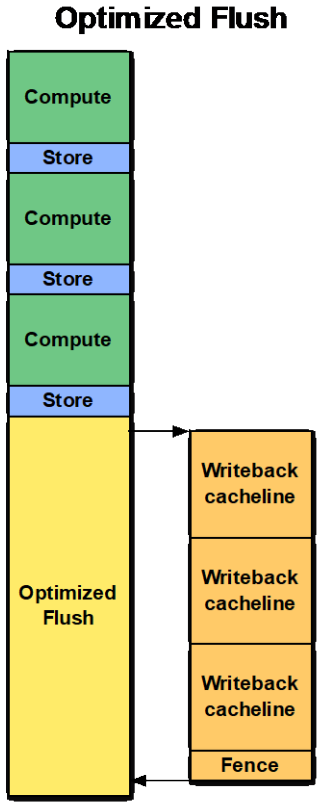
2017 Storage Developer Conference. © Intel and Oracle. All Rights Reserved.

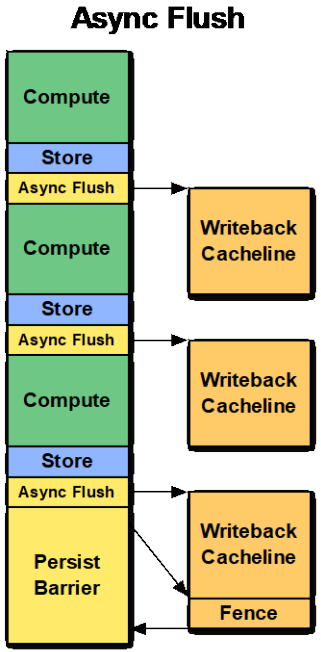
Why is Remote Flushing Any Different?

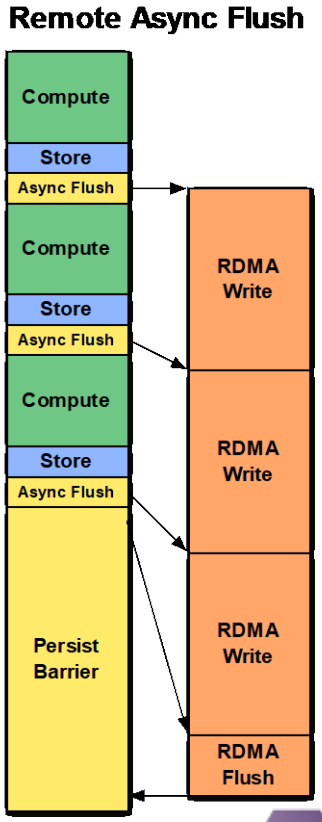
- Locally, the flush is fast
  - No time to context switch away
  - Potentially pipelined by hardware visibility rules
- Remote flushes go across the wire
  - Waiting for them prevents pipelining
  - Adding the store barriers to the data stream brings pipelining back into it

2017 Storage Developer Conference. © Intel and Oracle. All Rights Reserved.

What is Async Flush?







2017 Storage Developer Conference. © Intel and Oracle. All Rights Reserved.

Async Flush SNIA Work in Progress

- Specify the semantics of Async Flush and Persist Barrier
  - Goal: compatible semantics for both local and remote access
- What stores must be flushed?
  - Compiler optimizer instruction reordering
  - Superscalar CPU instruction reordering
  - Multithread store ordering
- What flushes must complete for the Persist Barrier?
  - Again, compiler and CPU instruction reordering
  - Thread-local barrier, not across threads

2017 Storage Developer Conference. © Intel and Oracle. All Rights Reserved.

Summary

- Several type of flushes are available
  - HW provides some, SW provides some
- Flushing won't go away any time soon
- Mostly use standard/optimized flushes
- Mostly apps should depend on libraries
- SNIA TWG still working on some details, but the 99% case is well-defined and implemented on both Windows and Linux

2017 Storage Developer Conference. © Intel and Oracle. All Rights Reserved.