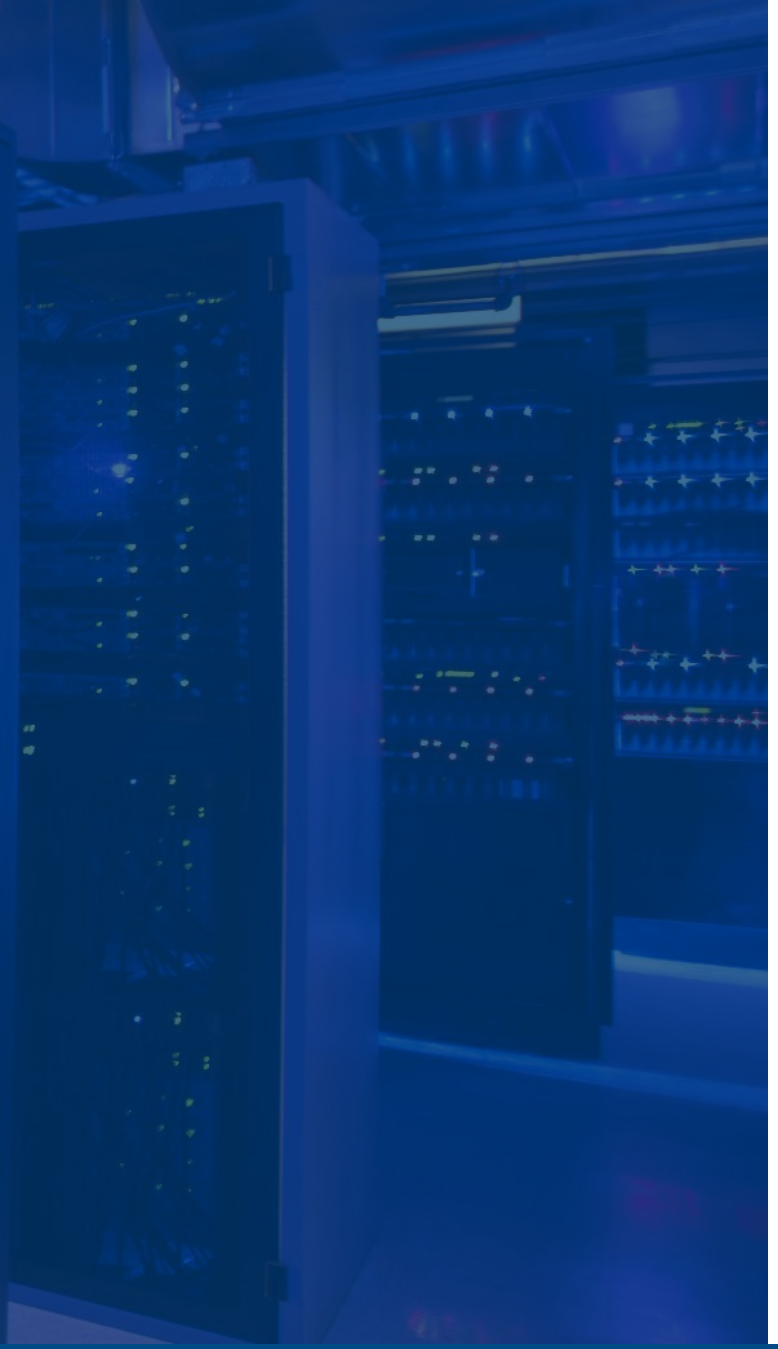
PMDK - State of the Project

Persistent Memory Development Kit

Wu, Dennis

PRC Optane CoE Manager

Intel



Agenda

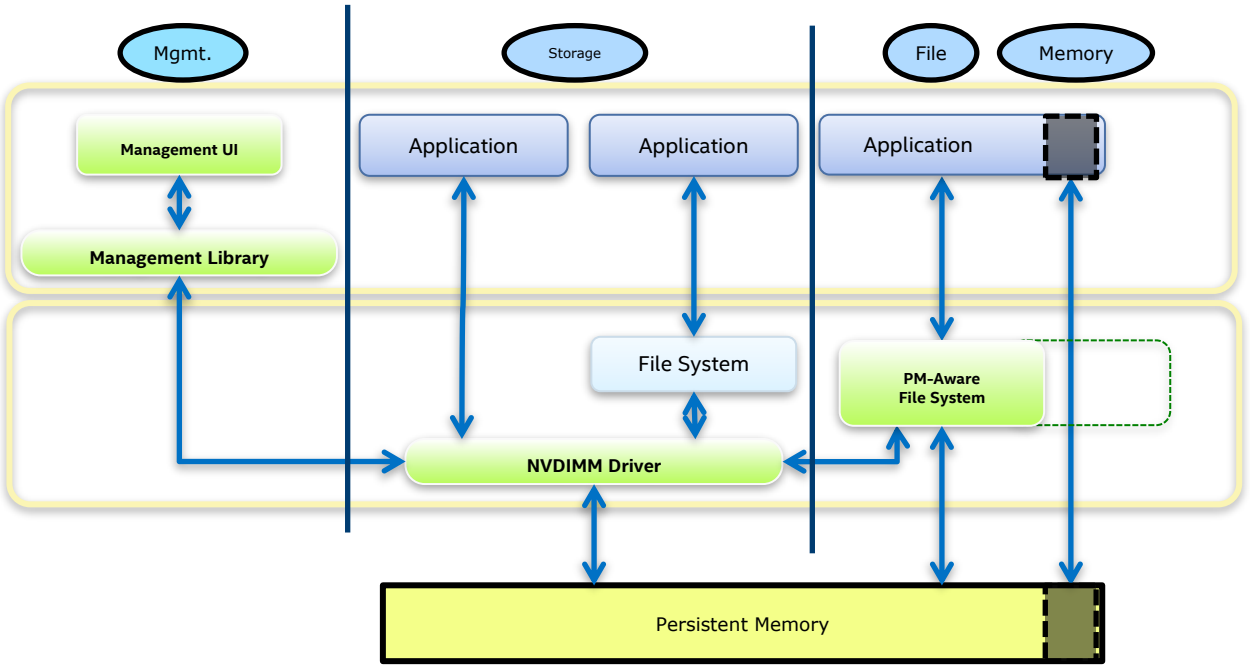
1. 持久内存编程模型的由来
   How we came to be
2. PMDK的方向和目标
   What are we doing and why
3. PMDK现在的状况
   What have we done so far
4. PMDK未来的计划
   What are we doing next

01

持久内存编程模型的由来

持久内存编程模型

- 如何将持久内存设备暴露给应用
  - 如何命名，和连接到持久内存
  - 如何控制执行权限
  - 如何备份、管理数据
  - 简短的回答:
    - 用文件的方式访问持久内存
    - 标准的文件接口，内存映射后直接访问
    - 厂商自己定义，如何备份及管理数据
- 持久内存编程模型由SNIA NVM技术工作组制定及公布



持久内存编程模型简要史

- 2012年6月
  - NVM编程技术工作组成立
  - 关键的一些 OSVs, ISVs, IHVs都有加入
- 2013年1月
  - 组织了第一次持久内存的峰会 (实际名称 “NVM 峰会”)
- 2013年7月
  - 创建了第一个Github的思维实验 （“linux-examples”)
- 2014年1月
  - NVM编程工作组开放了持久内存编程模型1.0版本

基于SNIA 持久内存编程模型编程

在pmem-aware的文件系统上打开一个文件

把这个文件map到用户地址空间

Okay, 现在你就有一个用户地址空间可以操作, have fun!

- 这个方式是必须的，但是并不足够来写出你想要的程序

大量的高优的工作从需求中过来：

- 需要一种方法来追踪持久内存的分配（类似与malloc/free，但是pmem-aware)
- 需要一种方法来实现事务的更新，不能出现脏写的情况
- 需要一些持久内存相关的容器类实现，例如队列，链表等等
- 需要让持久内存编程不容易出错

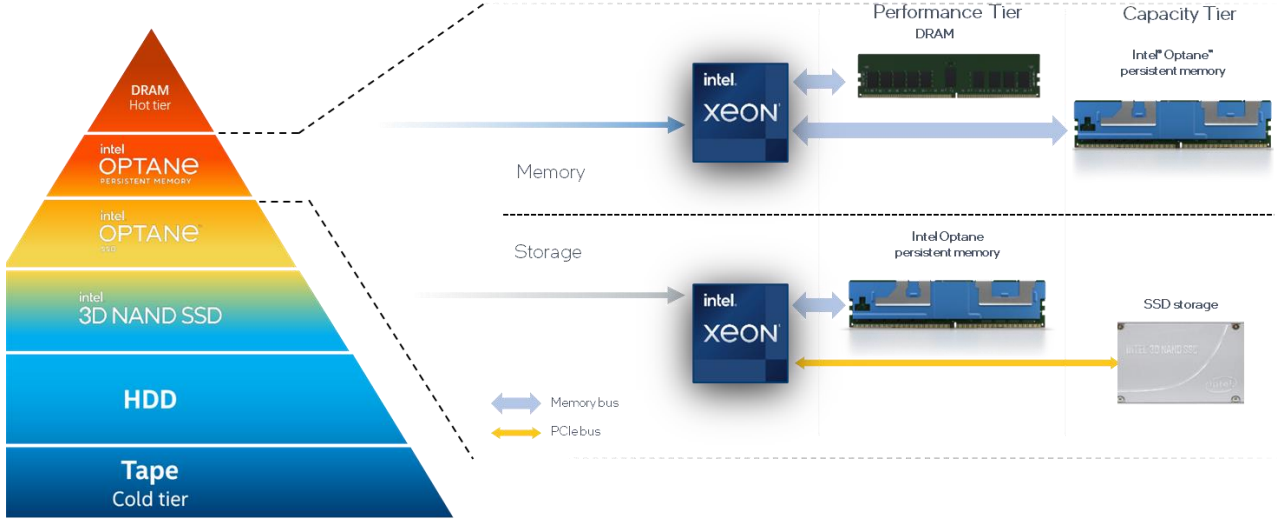
02

PMDK的方向和目标

使用持久内存解决实际问题

持久内存是多维的，它是内存，又是存储.

- 作为内存，比DRAM更便宜，同时容量也更大
  - 可以在几个TB的内存上实现以前不可能实现的场景
- 作为存储，它可以比其它的存储有数量级的性能提升
  - 支持超低延迟的检索和事务，还可以绕过page cache降低对DRAM需求
- 作为两者兼而有之, 它非常独特，需要独特解决方案的新设计



持久内存用作内存

- 持久内存容量很大, 但是性能比DRAM要慢.
- PMem只是异构内存系统中的一种类型
  - NUMA、高带宽内存（HBM）、PMEM等
  - 应用程序通常假定所有内存都是相同的，希望由硬件或操作系统可以管理异构内存（内存模式、内存分层）
    - ……但即使在今天，情况也并非如此

PMDK的目标：帮助应用程序实现智能和可扩展的内存空间管理

持久内存用作存储

- 持久内存容量比传统的存储小, 但是性能远好于传统存储
  - 当年SSD技术，也是经过多年才逐渐被广泛使用
  - 持久内存可以作为存储的缓存和性能层，比如将预写日志和数据分离...
- 由于直接访问，持久内存不使用页面缓存。
  - 这降低了成本，并保证了稳定延迟。

PMDK的目标：帮助开发者修改现有的存储方案并获得性能提升

持久内存作为内存和存储

- 数据库存储引擎的设计本质是如何掩盖存储和内存之间巨大差异的研究
  - 从某种程度上来说，我们现在可以不必要那么做...
- 持久内存是一个新的层，它弥补了内存和存储之间的差距
  - 支持减少访问延迟和写入放大的新技术
  - 容错算法仍然需要记录数据，但现在可以使用cache line粒度的单个加载/存储指令来记录数据

PMDK的目标：帮助开发人员使用合并内存和存储的新技术

PMDK的方向和目标

“Make easy things easy and hard things possible”

- Larry Wall, about Perl programming language.

- PMDK的目标是，而且一直是，让持久内存编程变得简单
- 解决用户经常遇到的复杂且可能具有挑战性的问题
  - 通过多层软件栈实现，每一层都在前层的基础上添加新功能。
  - 应用程序可以选择设当的抽象级别。

03

PMDK目前现状

PMDK-Persistent Memory Development Kit
多种解决方案

易失性的库

libvmemcache

Space-efficient scalable memory-oriented embeddable caching solutions.

libmemkind

Memory allocator with specialized per-kind heap allocation capabilities (PMem, DRAM);
Now with APIs for heterogeneous systems.

remote use cases

librpma

Easy to use library for remote memory access over RDMA; Exposes PMem specific primitives.

持久性的库

pmemkv-java | pmemkv-python | pmemkv-js | pmemkv-…

libpmemkv

Persistent Memory key-value store; Easy to get started with.

libpmemobj-cpp

C++ bindings for libpmemobj and PMem-STL; The easiest and most idiomatic way to write persistent applications.

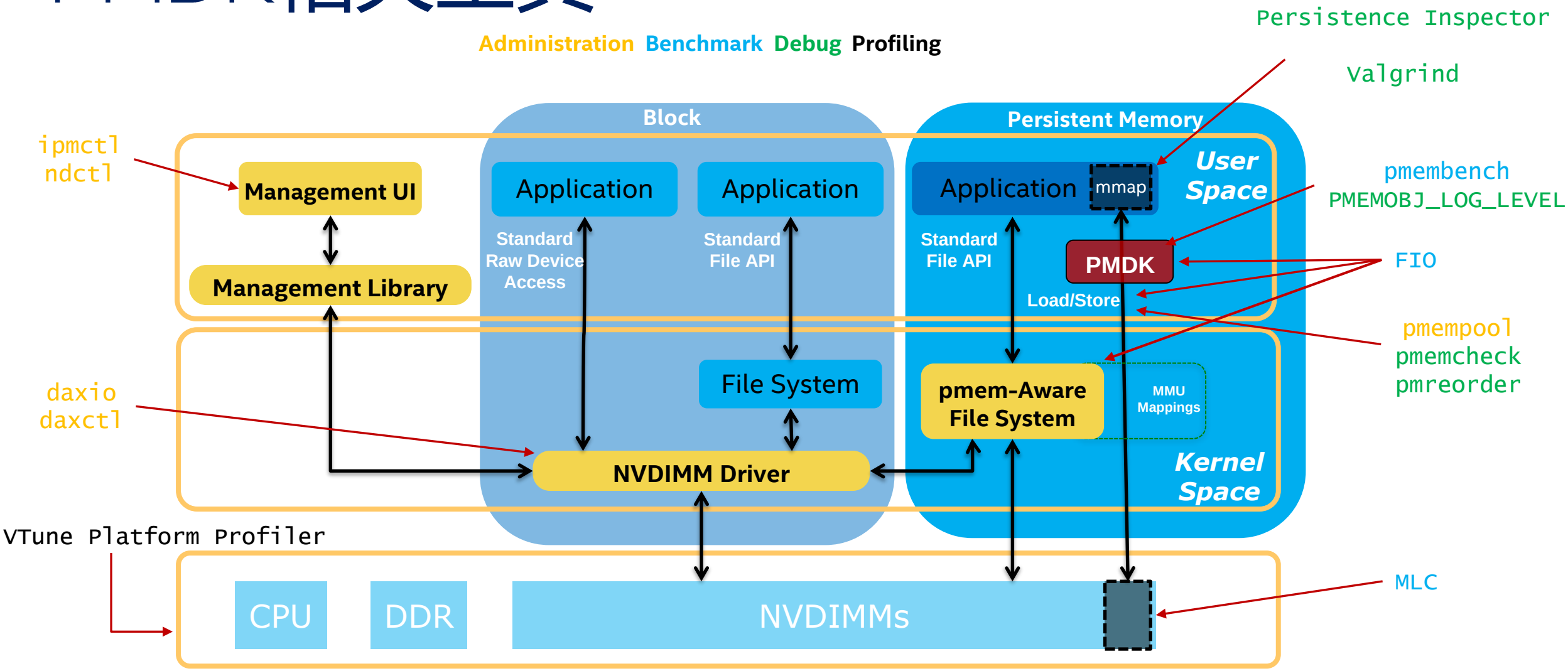
libpmemobj

General purpose transactional object store; Provides memory allocation and transactions needed for complex logic.

libpmem2

Essentials; low-level API that provides an abstraction for all the necessary primitives an application needs to use PMem.

PMDK相关工具


Administration Benchmark Debug Profiling
Persistence Inspector
Valgrind
ipmctl
ndctl
Management UI
Management Library
Block
Application
Application
Standard
Raw Device
Access
Standard
File API
File System
NVDIMM Driver
Persistent Memory
Application
mmap
User
Space
Standard
File API
PMDK
Load/Store
pmem-Aware
File System
MMU
Mappings
Kernel
Space
pmembench
PMEMOBJ_LOG_LEVEL
FIO
pmempool
pmemcheck
pmreorder
daxio
daxctl
VTune Platform Profiler
CPU
DDR
NVDIMMs
MLC

04

PMDK未来计划

异构 & 易用

- 现有的PMDK解决方案主要关注于持久内存，PMem通常是大型多样化系统中的一个组件，必须与内存和存储设备共存。
- 让应用程序开发人员来集成所有这些功能可以实现更大的灵活性，但却难以创建全面的解决方案。
- 我们现在已经开始研究解决方案，让开发人员可以根据他们想要的集成级别进行选择，“让简单的事情变得容易和困难成为可能。”

Pmemkv – 混合引擎

- 现有的pmemkv引擎主要将PMem用于元数据和数据存储。
- 这样做的好处是确保了强大的一致性，但大多数操作的延迟更高。

- 我们现在正在研究混合pmemkv引擎，该引擎将利用DRAM存储其部分结构。
- 这将让用户在更高的性能和更低的主内存消耗之间进行选择。

Memkind - 内存空间管理

- 新版memkind让开发人员能够根据内存的属性（延迟、容量、带宽）而不是内存明确的名称（DRAM、PMEM、HBM）来分配内存。
  - 但这些分配调用仍需要显式管理，这会带来复杂性。

- 我们现在正在开发一个库，用于在内存分配级别进行自动内存分层。该软件将像任何其他内存分配器一样可加载，在用户空间级别透明地管理内存。
  - 最初，在不同类型的内存之间进行选择的启发式方法将很简单，并且基于分配大小。
  - 但我们的目的是研究利用malloc调用堆栈和运行时评测来更好地做出数据放置决策的可能性。

实现新特性

- Intel® Data Streaming Accelerator (Intel® DSA)。 https://software.intel.com/en-us/download/intel-data-streaming-accelerator-preliminary-architecture-specification
  - 无需使用CPU的资源，将内存间的数据的移动（movement ）和转换（transformation）offload给DSA加速器。
  - 可以提供给应用异步数据拷贝的接口。
- ISA support - movdir64b
  - 保证64个字节的原子性


intel®
SPDK, PMDK, Intel® Performance Analyzers
Virtual Forum