From DRAM to NVM : Technologies & Challenges

Manolis Katsaragakis

mkatsaragakis@microlab.ntua.gr

National Technical University of Athens

&

KU Leuven

Overview

- Introduction
- NVM Challenges
- Intel Optane DC Persistent Memory
- Conclusion

Overview

- **Introduction**
- NVM Challenges
- Intel Optane DC Persistent Memory
- Conclusion

Volatile and Non Volatile Memories

- **Volatile memory**, is computer memory that requires power to maintain the stored information; it retains its contents while powered on but when the power is interrupted, the stored data is quickly lost.

- Traditional types of Non-Volatile Memories and Storage:

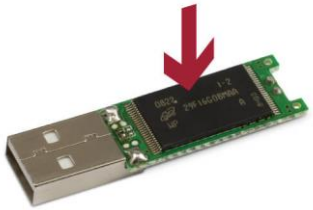
SRAM/DRAM

Volatile and Non Volatile Memories

- **Non-volatile memory** (**NVM**) or **non-volatile storage** is a type of computer memory that can retrieve stored information even after having been power cycled.

- Traditional types of Non-Volatile Memories and Storage:

Read-only Memory (ROM)



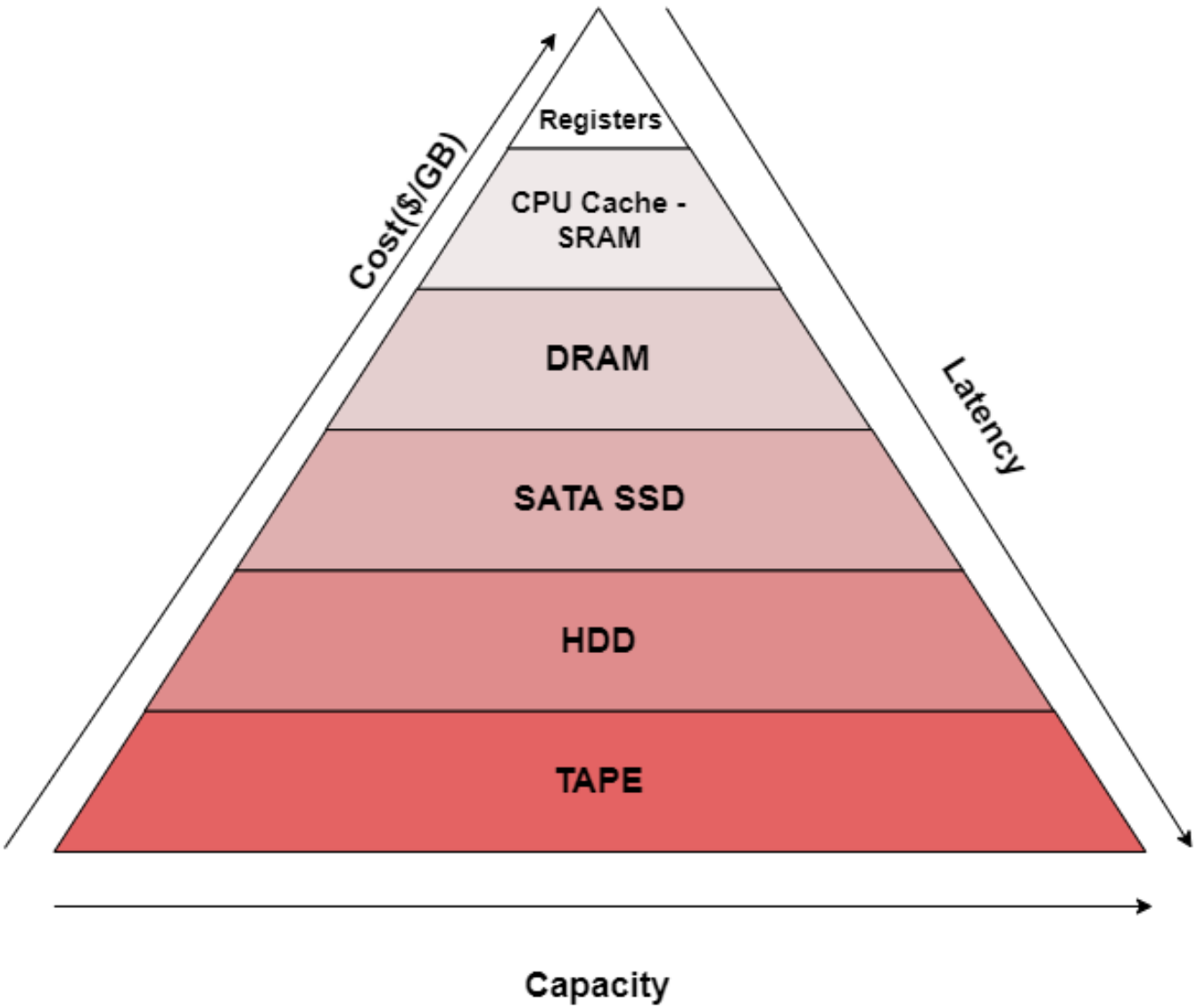
NAND/NOR Flash

Memory Cards

Solid-State Drive (SSD)

Hard Disk Drive (HDD)

Traditional Memory Hierarchy



DRAM vs SSD/HDD

SRAM/DRAM TECHNOLOGIES

SSD/HDD TECHNOLOGIES

DRAM vs SSD/HDD

SRAM/DRAM TECHNOLOGIES

- Volatile

SSD/HDD TECHNOLOGIES

- Non Volatile

DRAM vs SSD/HDD

SRAM/DRAM TECHNOLOGIES

- Volatile
- Low Access Latency

SSD/HDD TECHNOLOGIES

- Non Volatile
- High Access Latency

DRAM vs SSD/HDD

SRAM/DRAM TECHNOLOGIES

- Volatile
- Low Access Latency
- High Cost per GB

SSD/HDD TECHNOLOGIES

- Non Volatile
- High Access Latency
- Low Cost per GB

DRAM vs SSD/HDD

SRAM/DRAM TECHNOLOGIES

- Volatile
- Low Access Latency
- High Cost per GB
- Byte Addressable

SSD/HDD TECHNOLOGIES

- Non Volatile
- High Access Latency
- Low Cost per GB
- Block-Word Addressable

DRAM vs SSD/HDD

SRAM/DRAM TECHNOLOGIES

- Volatile
- Low Access Latency
- High Cost per GB
- Byte Addressable
- Fast load/store Commands

SSD/HDD TECHNOLOGIES

- Non Volatile
- High Access Latency
- Low Cost per GB
- Block-Word Addressable
- Expensive I/O Commands

DRAM vs SSD/HDD

SRAM/DRAM TECHNOLOGIES

- Volatile
- Low Access Latency
- High Cost per GB
- Byte Addressable
- Fast load/store Commands
- High Leakage Power
- Limited Density Scaling

SSD/HDD TECHNOLOGIES

- Non Volatile
- High Access Latency
- Low Cost per GB
- Block-Word Addressable
- Expensive I/O Commands
- Mechanical Components

DRAM vs SSD/HDD

SRAM/DRAM TECHNOLOGIES

- Volatile
- Low Access Latency
- High Cost per GB
- Byte Addressable
- Fast load/store Commands
- High Leakage Power
- Limited Density Scaling

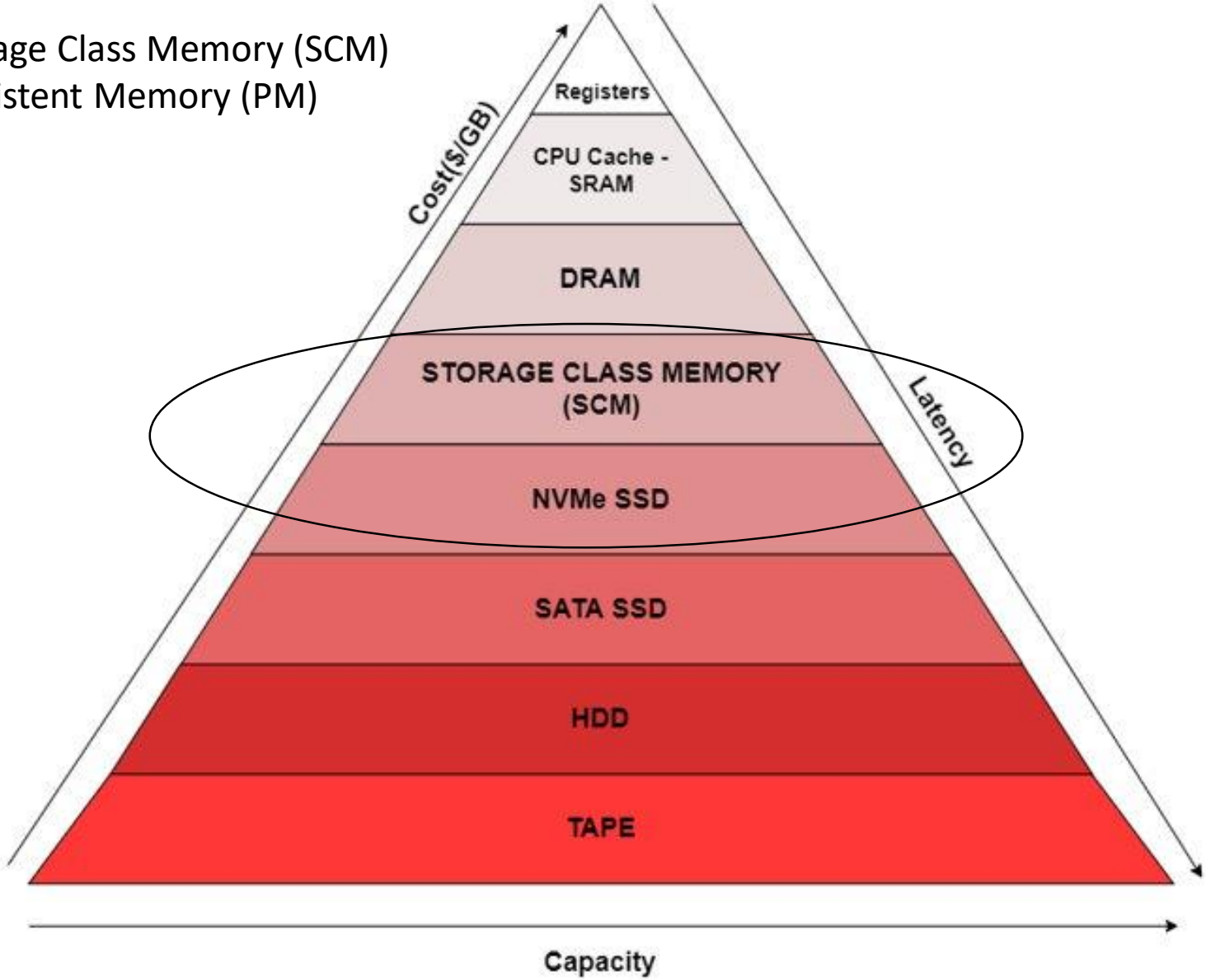
SSD/HDD TECHNOLOGIES

- Non Volatile
- High Access Latency
- Low Cost per GB
- Block-Word Addressable
- Expensive I/O Commands
- Mechanical Components

Can we combine them, while avoiding the pitfalls of each technology?

Emerging Non Volatile Memories – Storage Class Memories

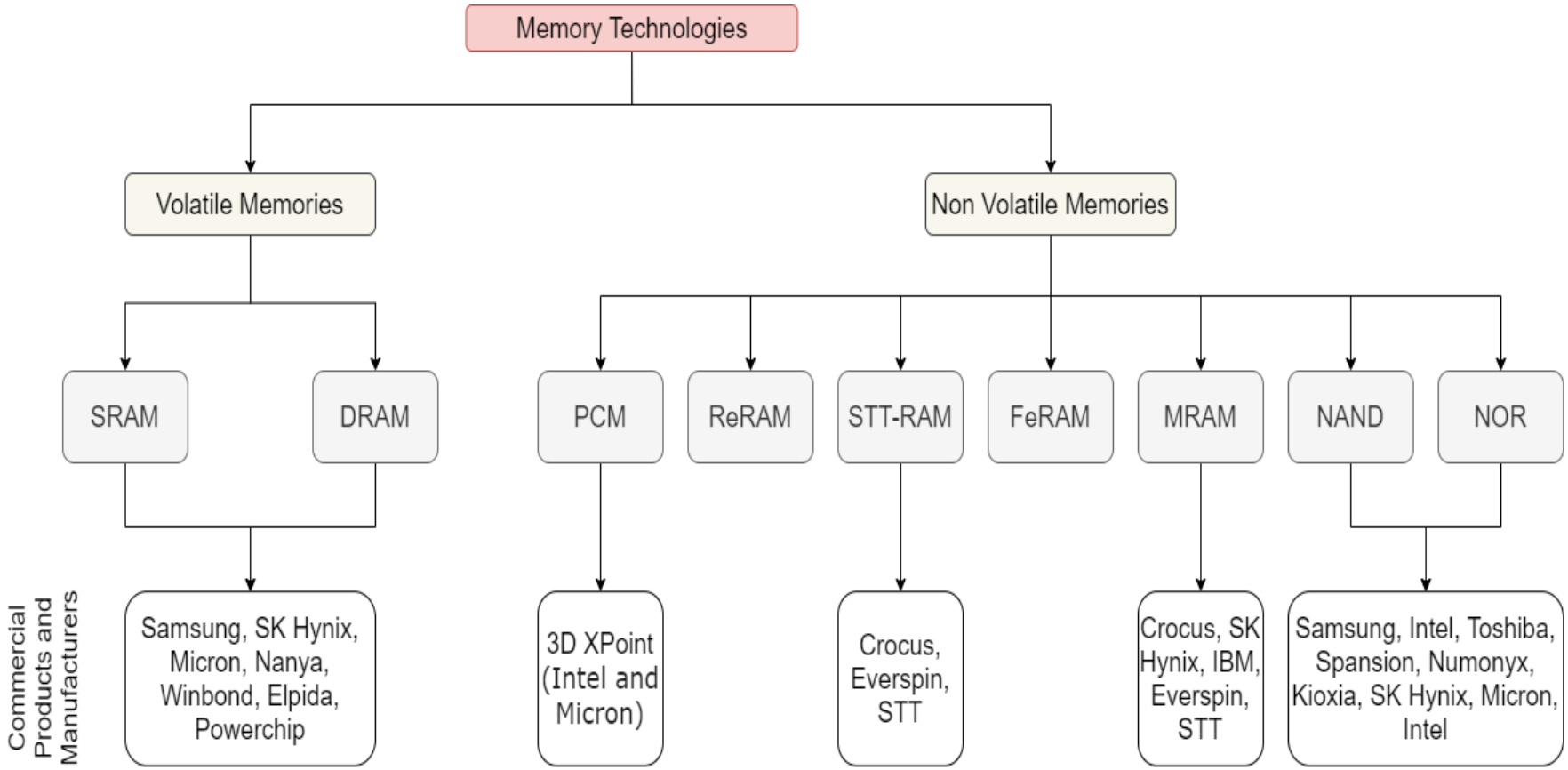
- Storage Class Memory (SCM)
- Persistent Memory (PM)



SCM Characteristics

- Non volatile
- No periodically data refresh required
- Near-zero standby power
- Near DRAM access latency
- Byte-addressability
- High density and scalability
- Assymetric properties of reads and writes
- Limited write endurance

Overview of Volatile/Non Volatile Memory Technologies



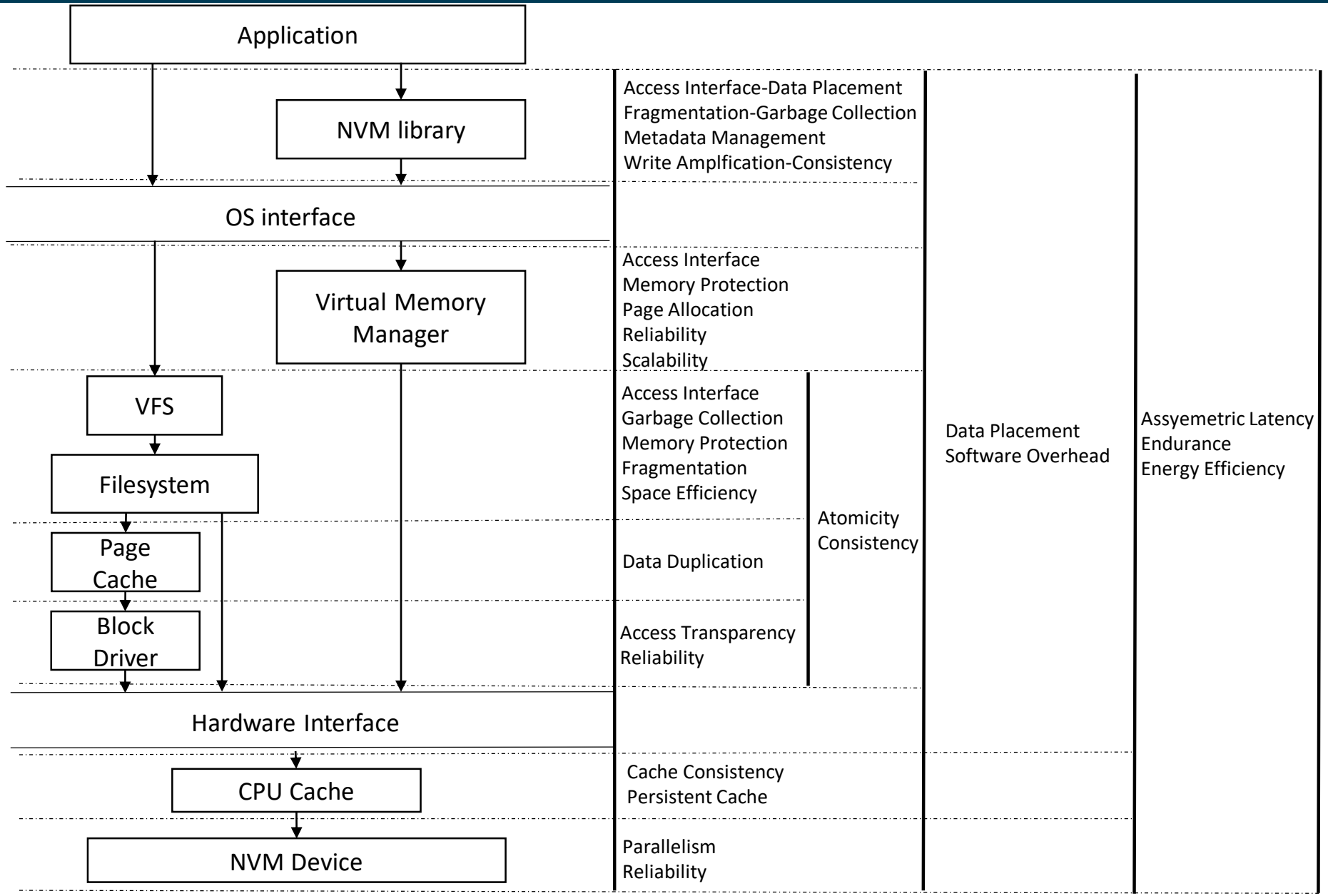
Memory Technologies Comparison

	SRAM	DRAM	PCM	ReRAM	STT-RAM	FeRAM	MRAM	NAND Flash	HDD
Read Latency	0.2-2 ns	10 ns	20-85 ns	10-20 ns	5-15 ns	25 ns	5-10 ns	15-35 μs	8.5-9.5 ms
Write Latency	0.2-2 ns	10 ns	150-1000 ns	100 ns	10-30 ns	75 ns	12 ns	200-500 μs	8.5-9.5 ms
Access Granularity	64B	64B	64B	64B	64B	64B	64B	512B	512B
Cell Size (F^2)	100-120	60-100	4-12	4-10	6-50	2-5	32	4-6	2/3
Write Endurance (writes/cell)	10^{16}	10^{18}	10^{8}	10^{10}	10^{12}-10^{15}	10^{15}	10^{15}	10^{4}-10^{5}	$> 10^{16}$
Program Energy/bit	High	2 pJ	100 pJ	2 pJ	0.02 pJ	2 pJ	120 pJ	10 nJ	NA
Leakage Power	High	Medium	Low	Low	Low	Low	Low	Low	Low
Volatile	Yes	Yes	No	No	No	No	No	No	No

Overview

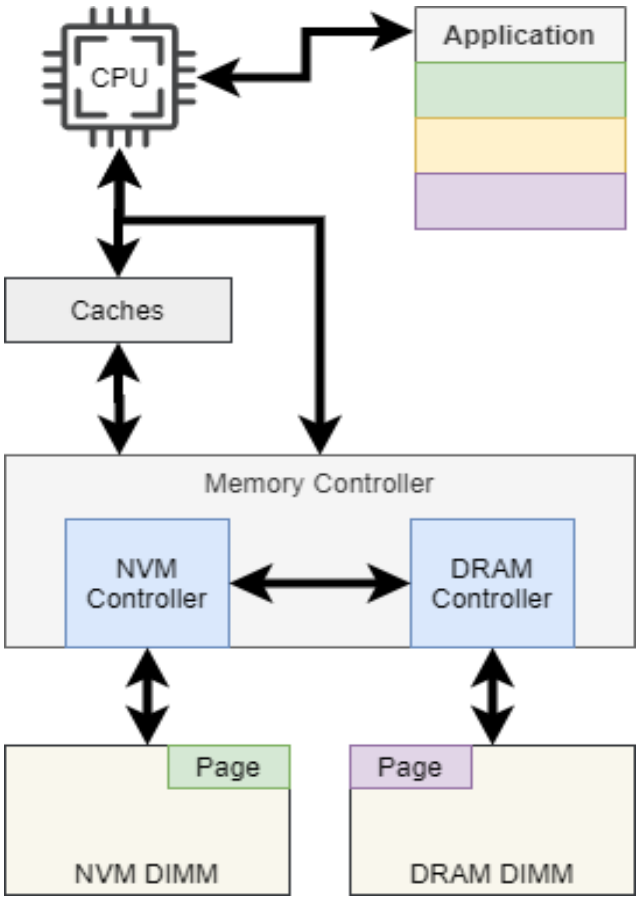
- Introduction
- **NVM Challenges**
- Intel Optane DC Persistent Memory
- Conclusion

Overview of NVM Challenges

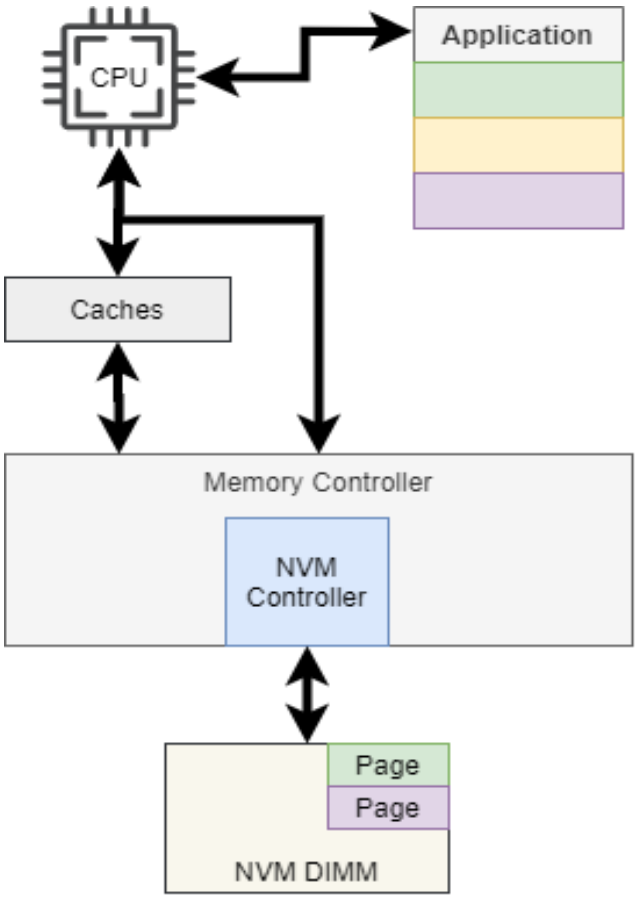


From Computer Architecture Perspective

How can we embed NVM technologies to existing architectures?

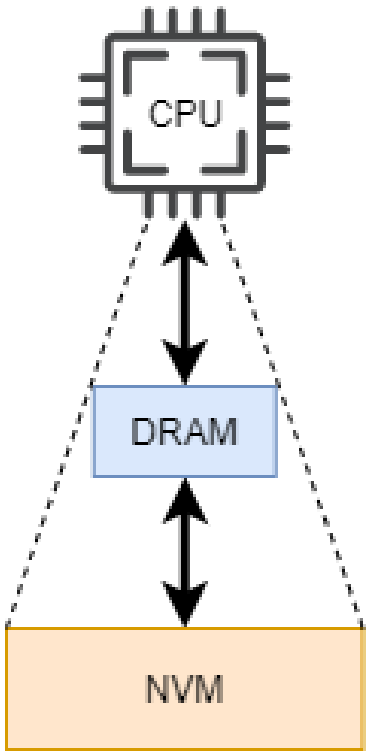


(a) Hybrid DRAM-NVM Architecture

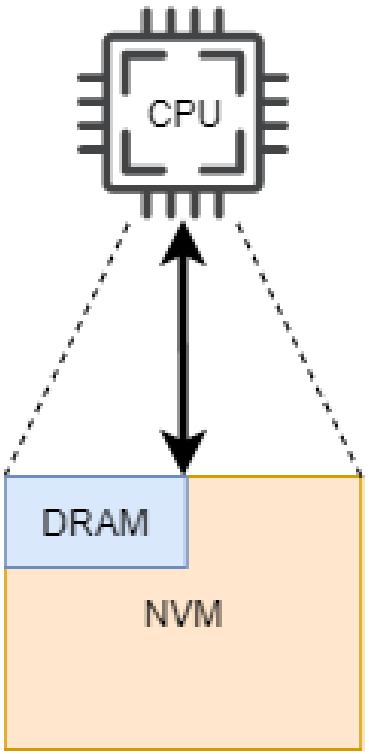


(b) Single NVM Architecture

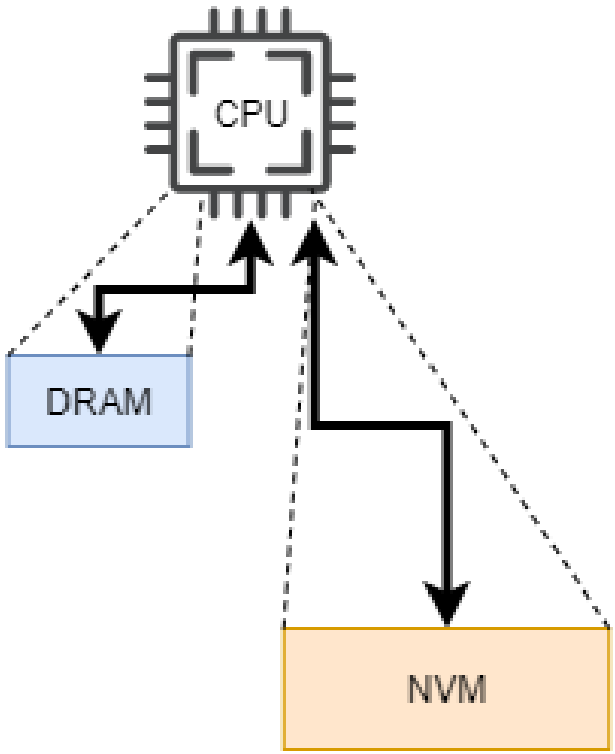
Logical Architecture in Hybrid Schemes



(a)DRAM as cache, NVM as main memory



(b) NVM as extension of DRAM



(c) DRAM as temporary buffer,
NVM as main memory

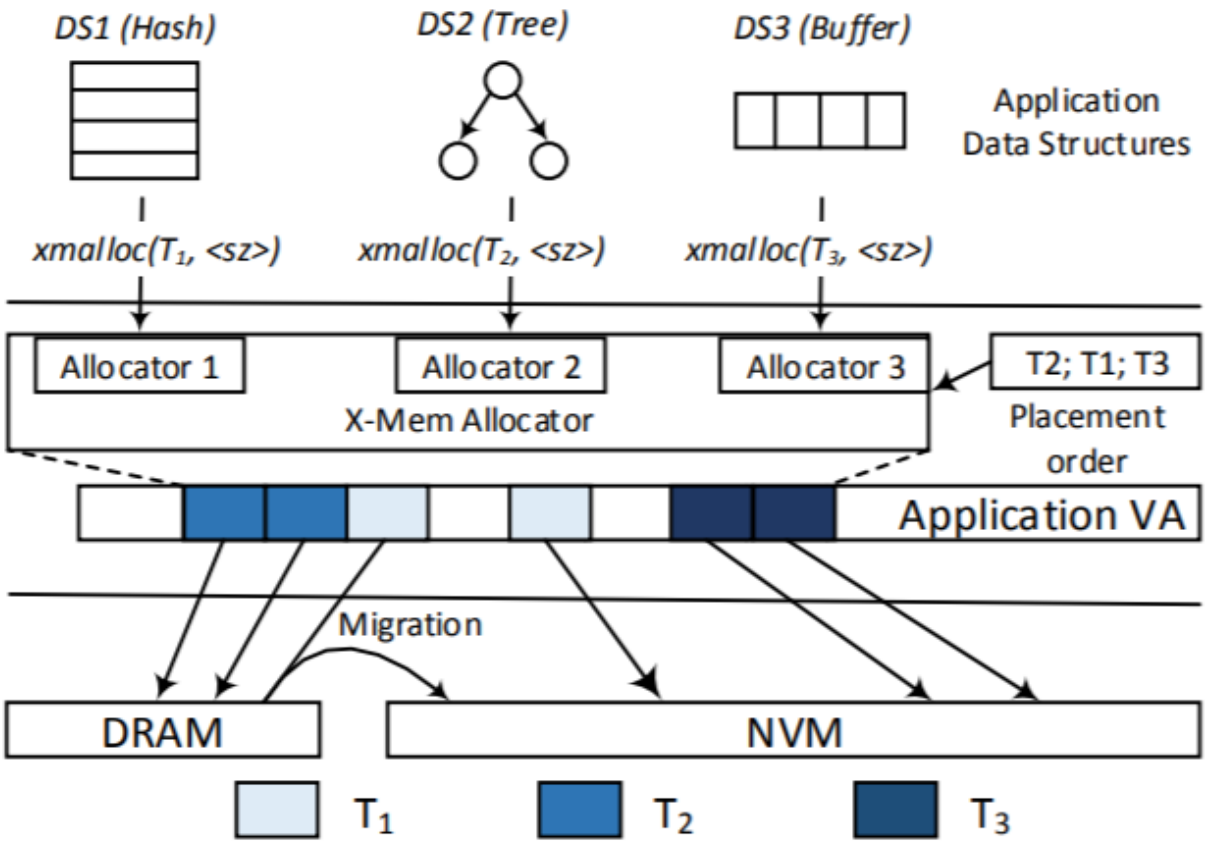
From Software Perspective

- How applications can benefit from SCM?
- How should OS adapt on SCM-oriented architectures?
- Which data should be stored in SCM and which in DRAM in hybrid architectures?
- How to keep data consistent?
- How should memory management change?
- How can we handle fault tolerance in case of system crashes?
- Can traditional file system be stored in SCM?

A Simple Example of a Software Design Challenge : Data Structure Design

- What criteria should a data structure allocation scheme satisfy?
  1. Respect r/w asymmetry
  2. Respect the reduced write endurance
  3. Maximize throughput
  4. Consistency & Persistence
  5. Scalability
  6. Logging and Recovery
  7. Efficient Space Utilization
  8. Other specific goals
- What data structures should someone select?
- Where a structure should be stored?

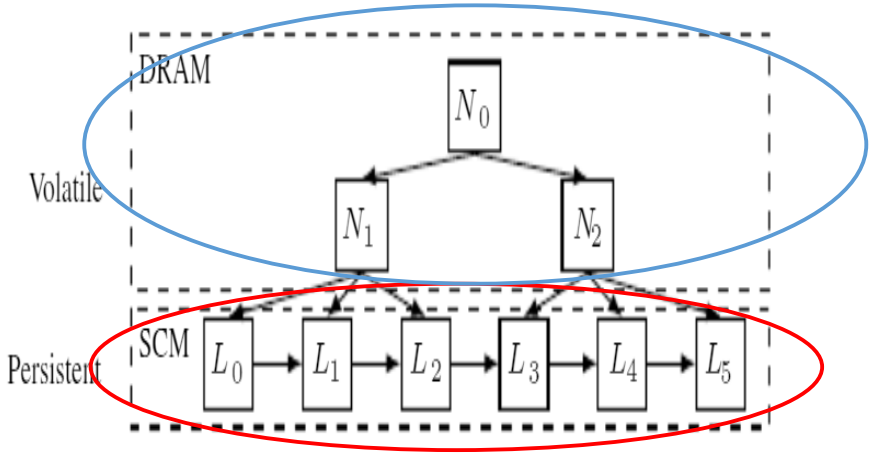
A Simple Example of a Software Design Challenge : Data Structure Design



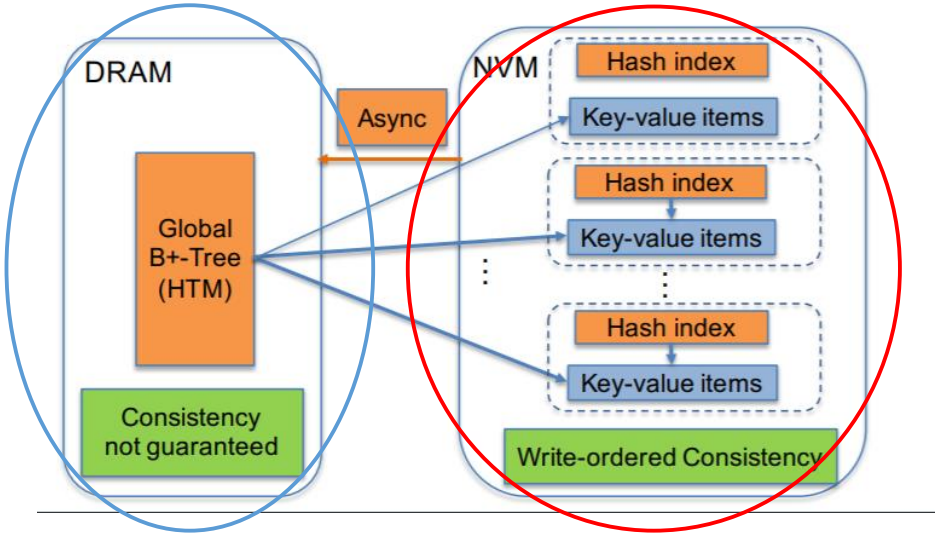
Source: https://dl.acm.org/doi/pdf/10.1145/2882903.2915251

A Simple Example of a Software Design Challenge : Data Structure Design

- Which data should be stored in SCM and which in DRAM in a hybrid architecture?
- A simple example:



Source: https://dl.acm.org/doi/pdf/10.1145/2882903.2915251



Source: https://www.usenix.org/sites/default/files/conference/protected-files/atc17_slides_xia.pdf

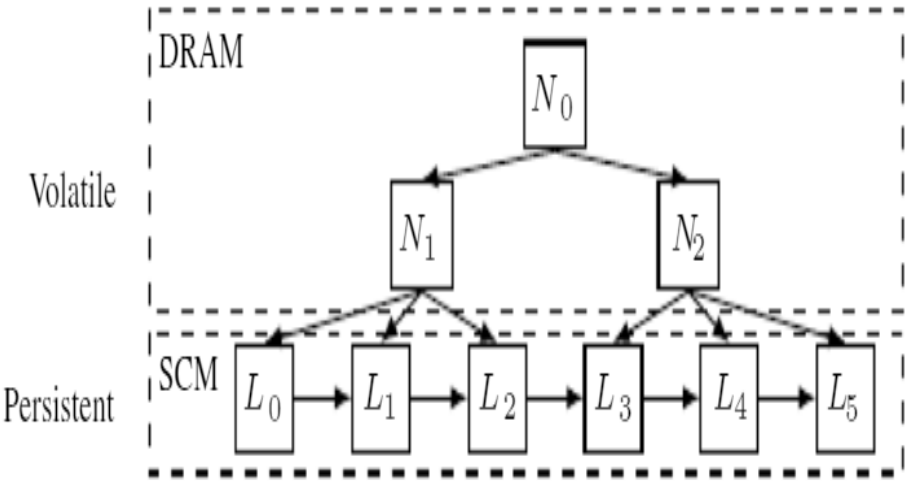
Non Persistent Data!

Persistent Data!

In case of a system crash what happens?

A Simple Example of a Software Design Challenge : Data Structure Design

FPTree

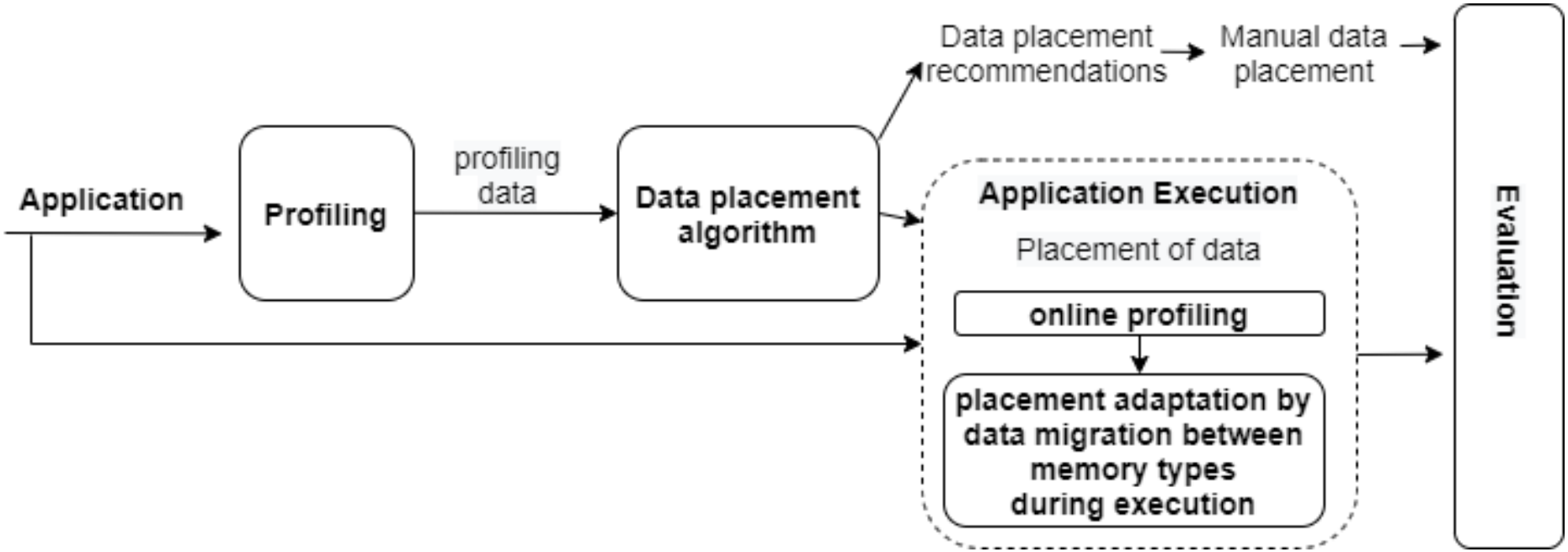


Source: https://dl.acm.org/doi/pdf/10.1145/2882903.2915251

- leaf nodes are placed in SCM using a persistent linked-list
- inner nodes are placed in DRAM and can be rebuilt as long as the leaves are in a consistent state
- Respect write endurance in SCM, i.e. minimize the number of writes in leafs

Selective persistence can be described as keeping in SCM the minimal set of primary data on which all the implementation effort for consistency will be focused, and rebuilding all non-primary data that is placed in DRAM upon recovery

A Simple Example of a Software Design Challenge : Typical Flow of Data Placement



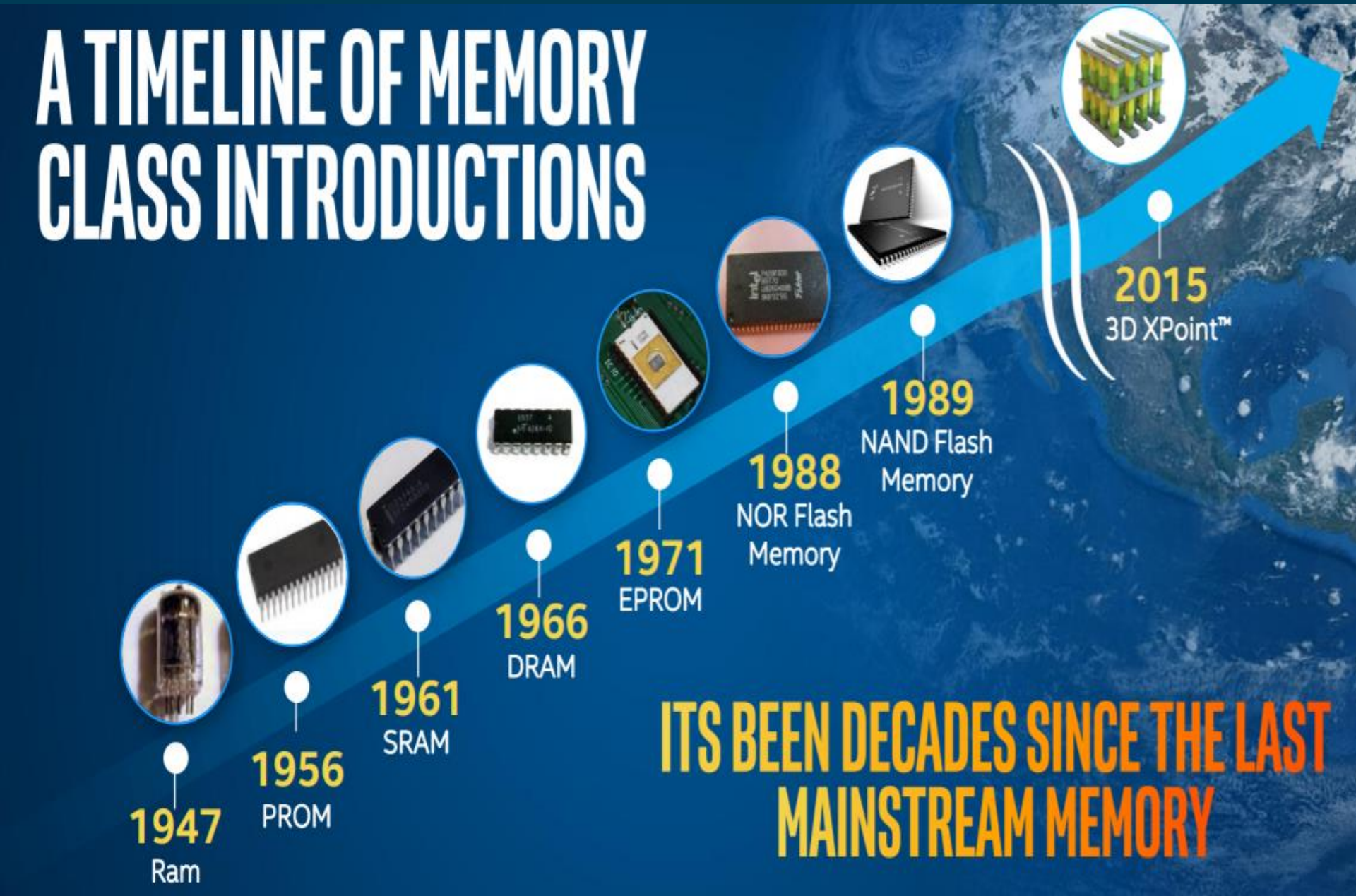
How can we avoid bottlenecks on NVM?

- **Performance Insensitivity** : Is read bandwidth < α GB/s and write bandwidth < β GB/s?
- **Write Throttling** : Does write bandwidth exceed γ GB/s?
- **Write Amplification** : Do we notice high percentage of small size writes(< δ bytes)?

Overview

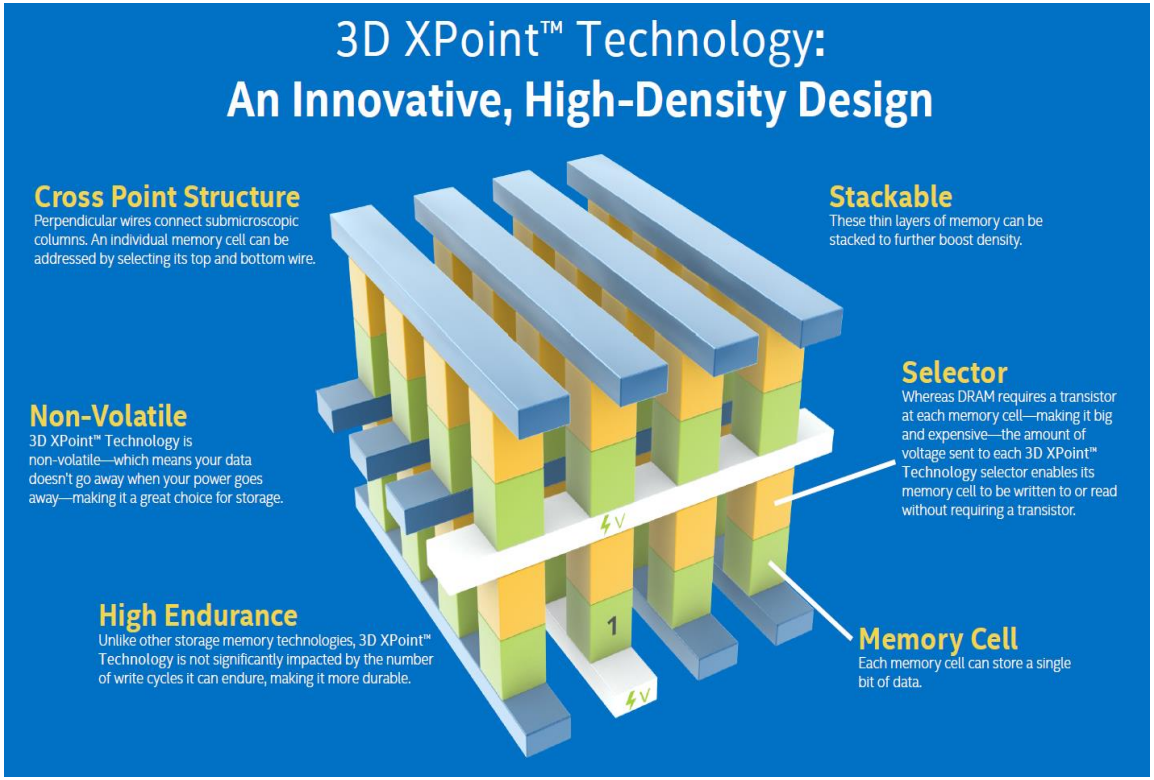
- Introduction
- NVM Challenges
- **Intel Optane DC Persistent Memory**
- Conclusion

Intel-Micron 3D Xpoint Architecture



Intel-Micron 3D Xpoint Architecture

- Benefits of both DRAM and NAND flash memory
- Better endurance than NAND memory cells
- 3D Xpoint faster than NAND and has higher write endurance



*Intel and Micron claimed that 3D XPoint was **1000x faster** than NAND, with **1000x** the **endurance**, and **10x** the **density** potential of DRAM*

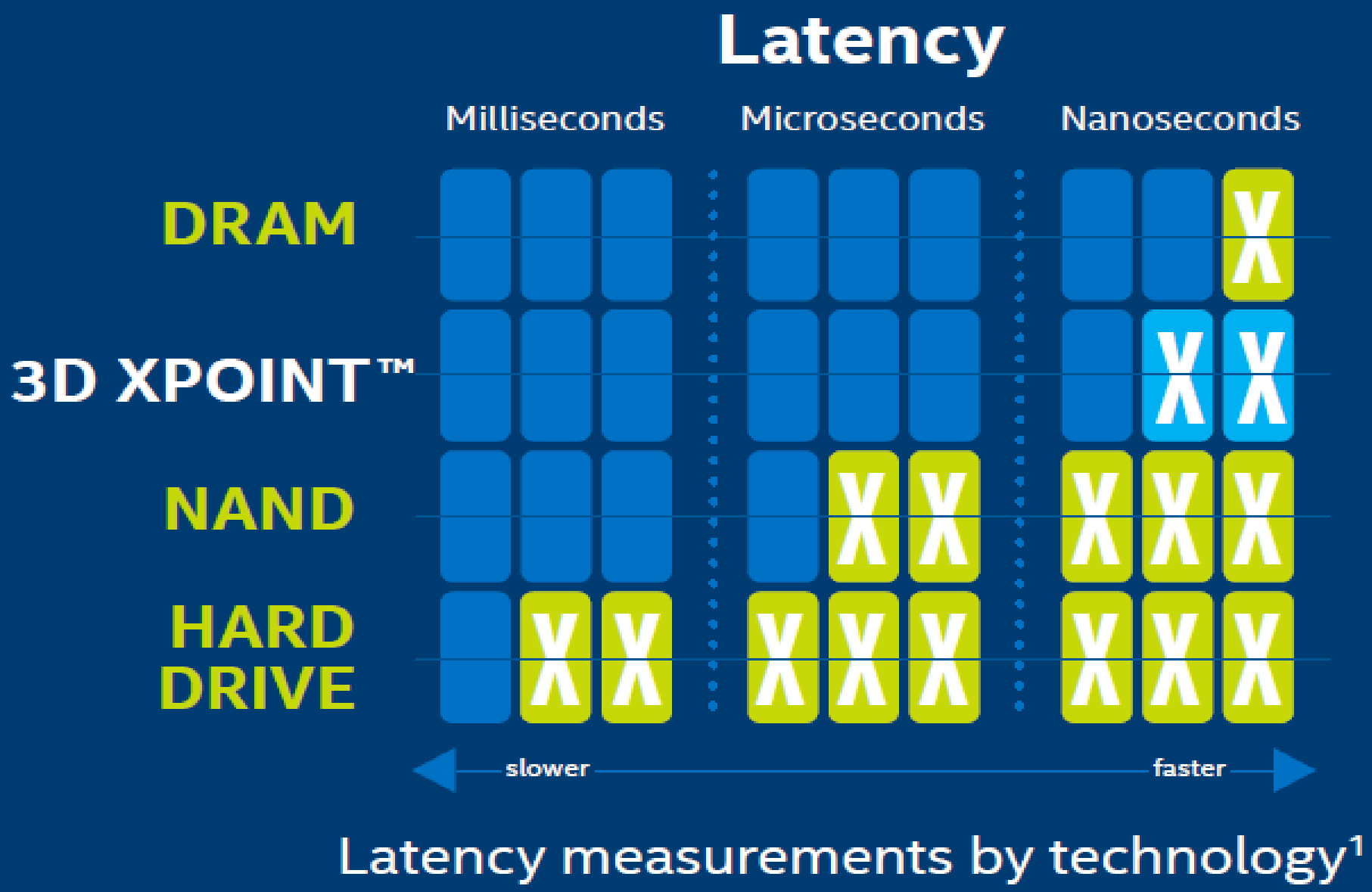
Inte Optane DC Persistent Memory

	Intel® Optane™ DC Persistent Memory 	Intel® Optane™ DC SSD with Intel® Memory Drive Technology 	Intel® Optane™ DC SSD 
Interface	Memory Channel	PCIe* Bus	PCIe* Bus
Capacity	Up to 512 GB per DIMM	Up to 1.5 TB per SSD	Up to 1.5 TB per SSD,
Intel Platform	2nd Generation Intel® Xeon® Scalable	Any	Any
Function	App Direct Mode: Persistent Memory Memory Mode: Volatile Memory Storage Over App Direct Mode: Persistent Storage	Volatile Memory	Persistent Storage
Form Factor	DIMM	U.2, M.2, AIC	U.2, M.2, AIC
Operating System	Windows*, Linux*, VMware ESXi*	Linux	Any

- Intel® Optane™ DC persistent memory is an innovative memory technology that delivers a unique combination of affordable large capacity and support for data persistence
- Based on 3D Xpoint Technology (Phase Change Memory-PCM)

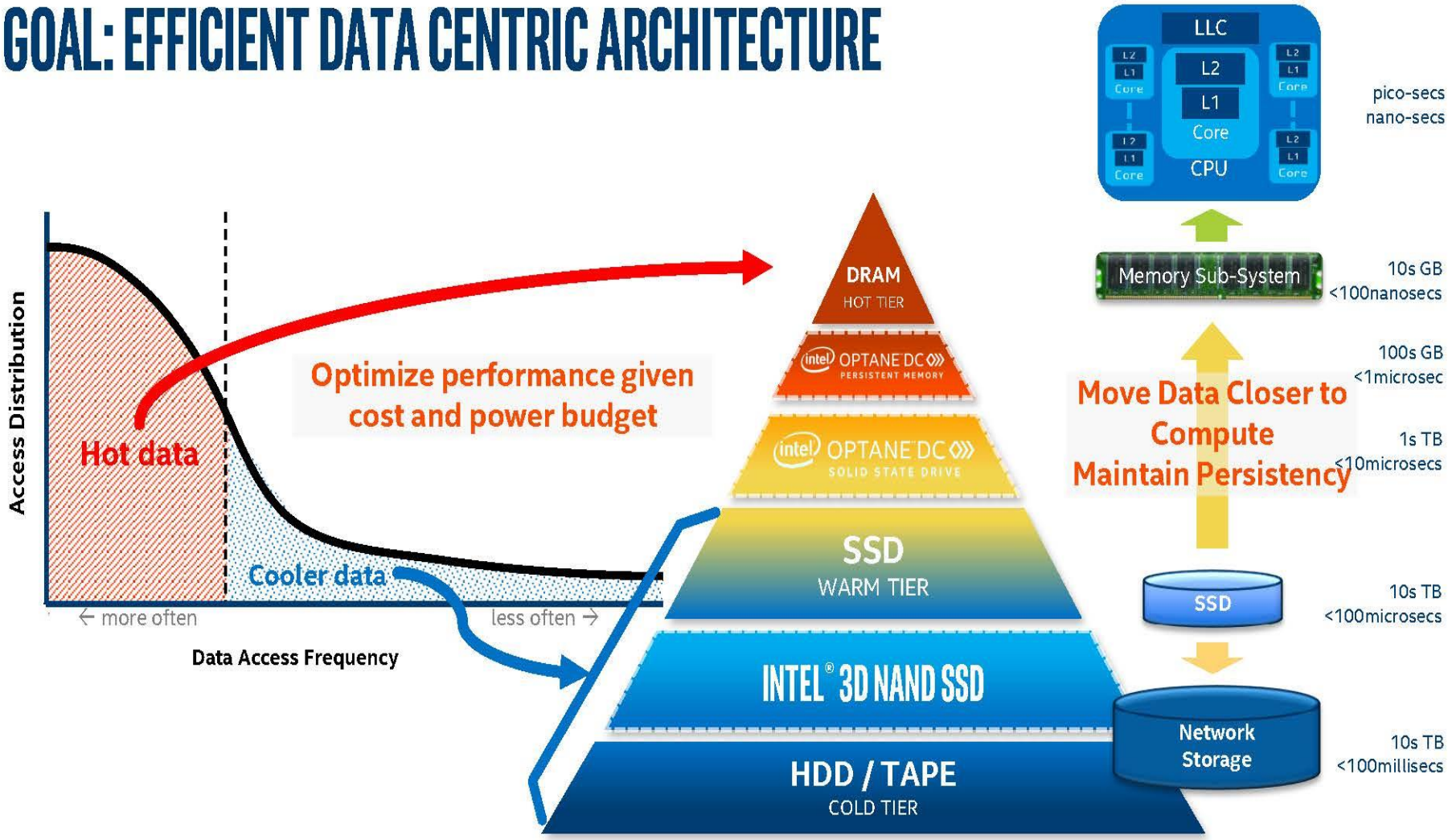
128, 256, 512 GB
~ 6.5$/GB
DDR4 Pin Compatible

Intel-Micron 3d XPoint



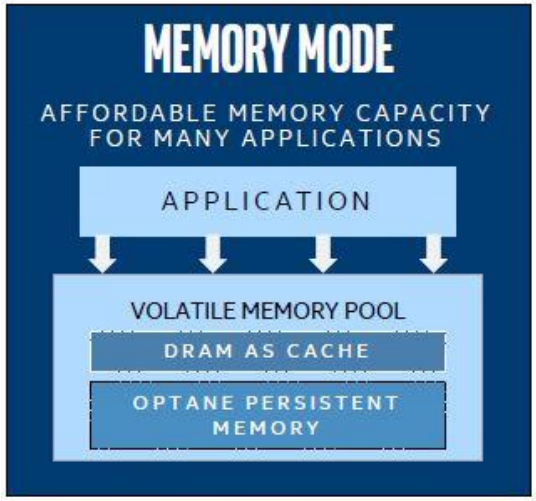
Source: https://www.anandtech.com/Show/Index/9470?cPage=5&all=False&sort=0&page=1&slug=intel-and-micron-announce-3d-xpoint-nonvolatile-memory-technology-1000x-higher-performance-endurance-than-nand

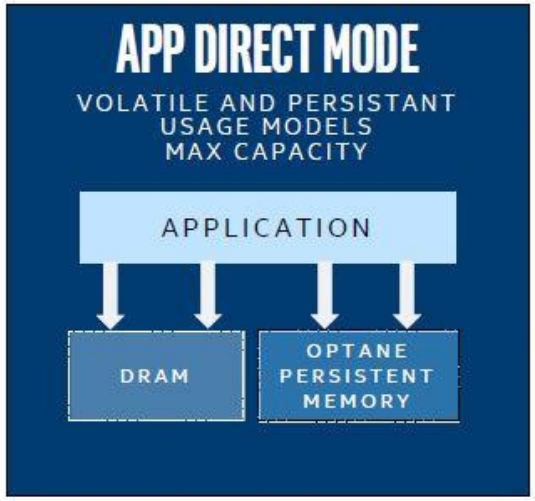
GOAL: EFFICIENT DATA CENTRIC ARCHITECTURE



Source: https://www.anandtech.com/Show/Index/9470?cPage=5&all=False&sort=0&page=1&slug=intel-and-micron-announce-3d-xpoint-nonvolatile-memory-technology-1000x-higher-performance-endurance-than-nand

Intel® Optane™ DC Persistent Memory





Platform Compatible w/ DDR4 Interface

Persistent and Volatile Usage Models

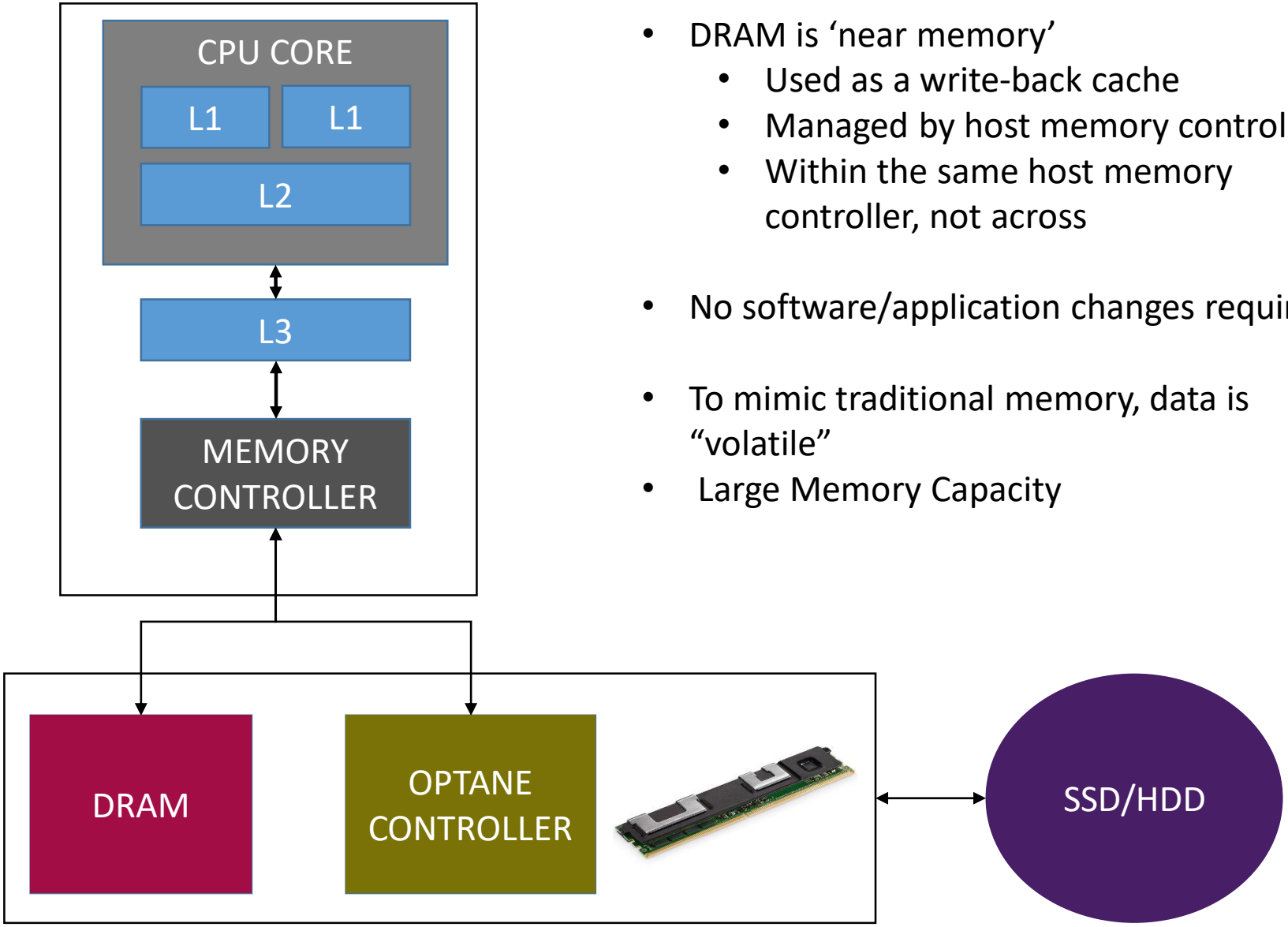
128GB to 512GB capacities per module

Legacy Workloads

Optimized Workloads

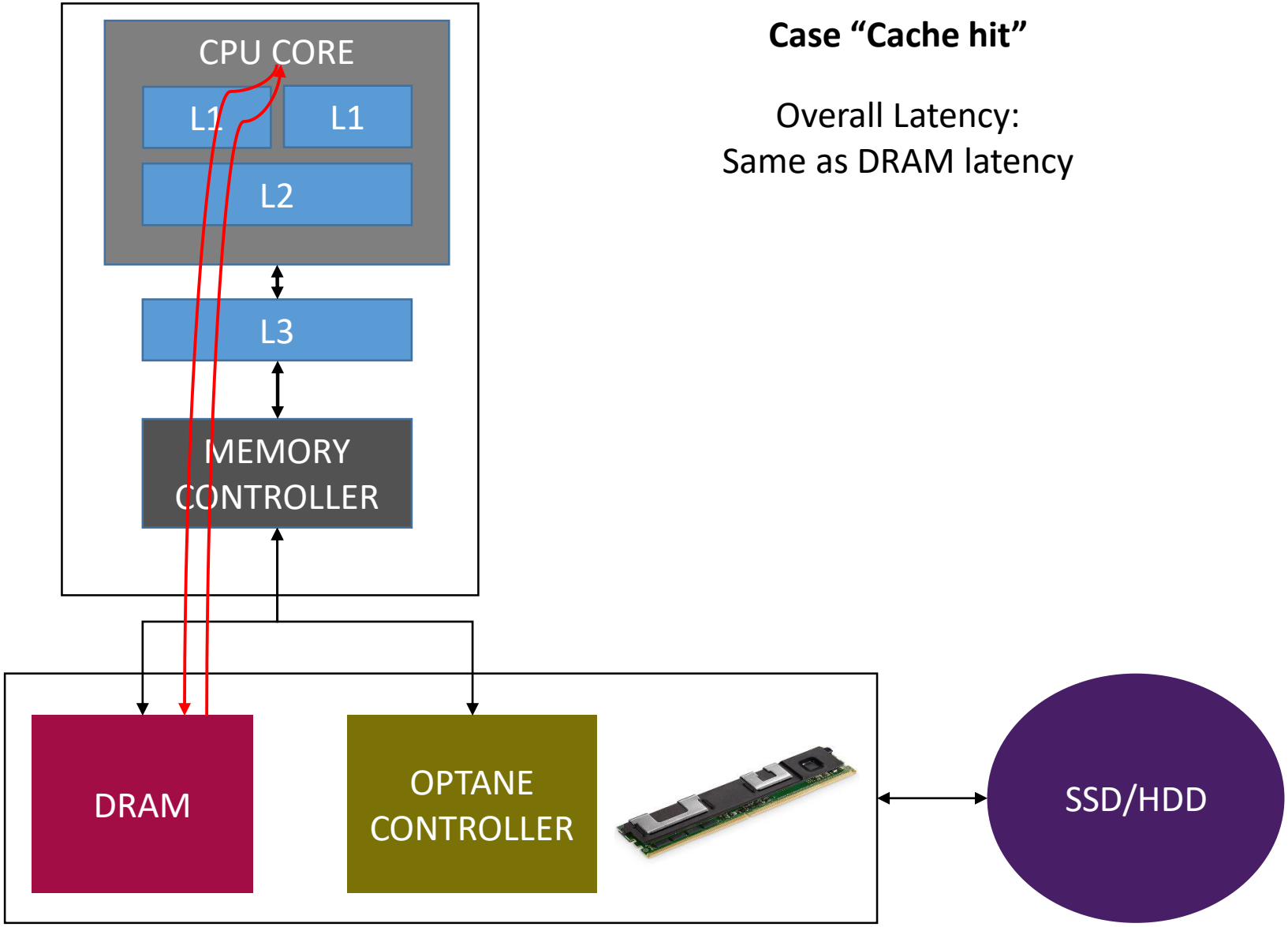
Module and Processor Co-Architecture from Day 1

Intel Optane DC: Memory Mode



- DRAM is 'near memory'
  - Used as a write-back cache
  - Managed by host memory controller
  - Within the same host memory controller, not across
- No software/application changes required
- To mimic traditional memory, data is "volatile"
- Large Memory Capacity

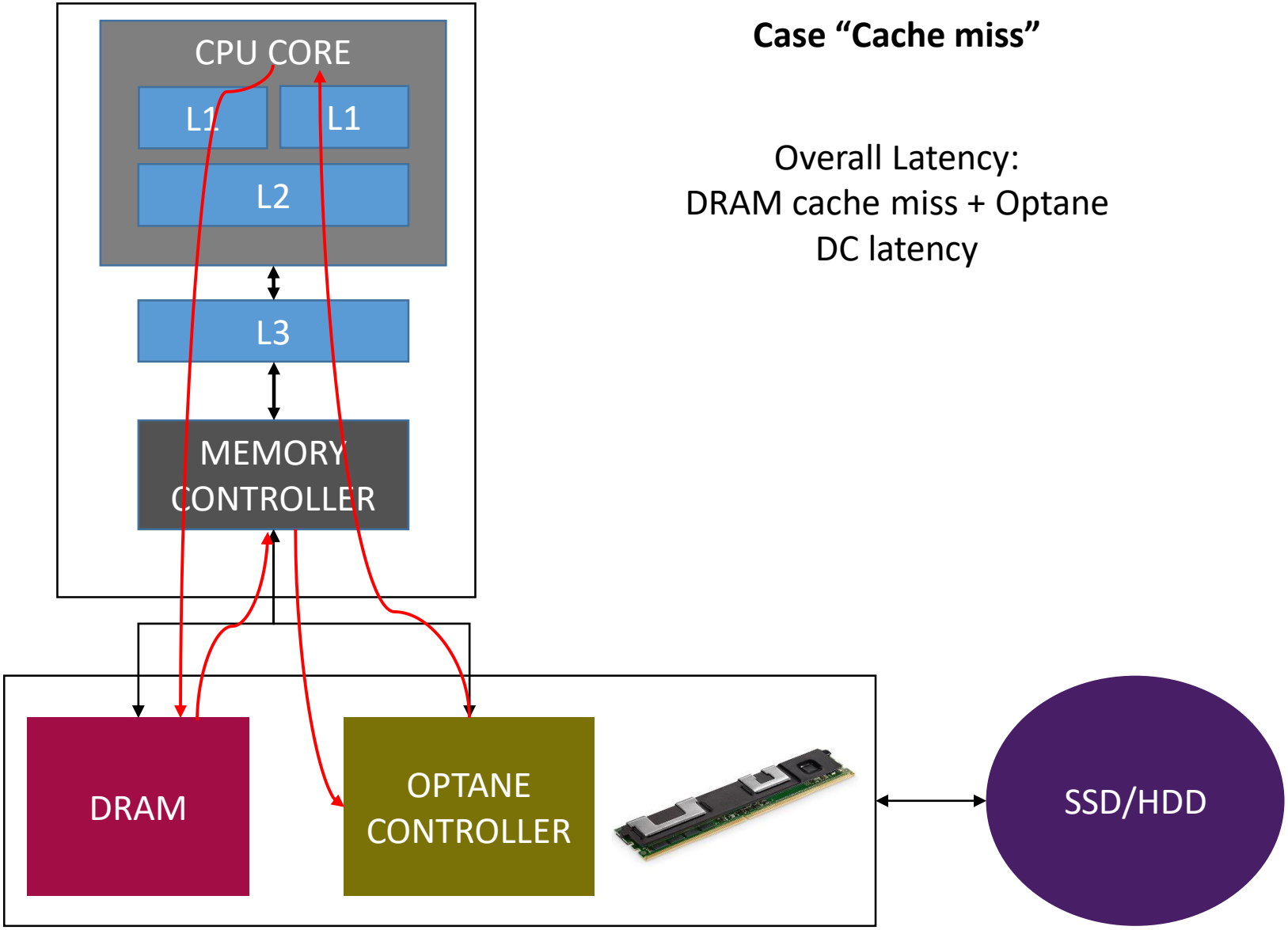
Memory Mode Transaction Flow



Case "Cache hit"

Overall Latency:
Same as DRAM latency

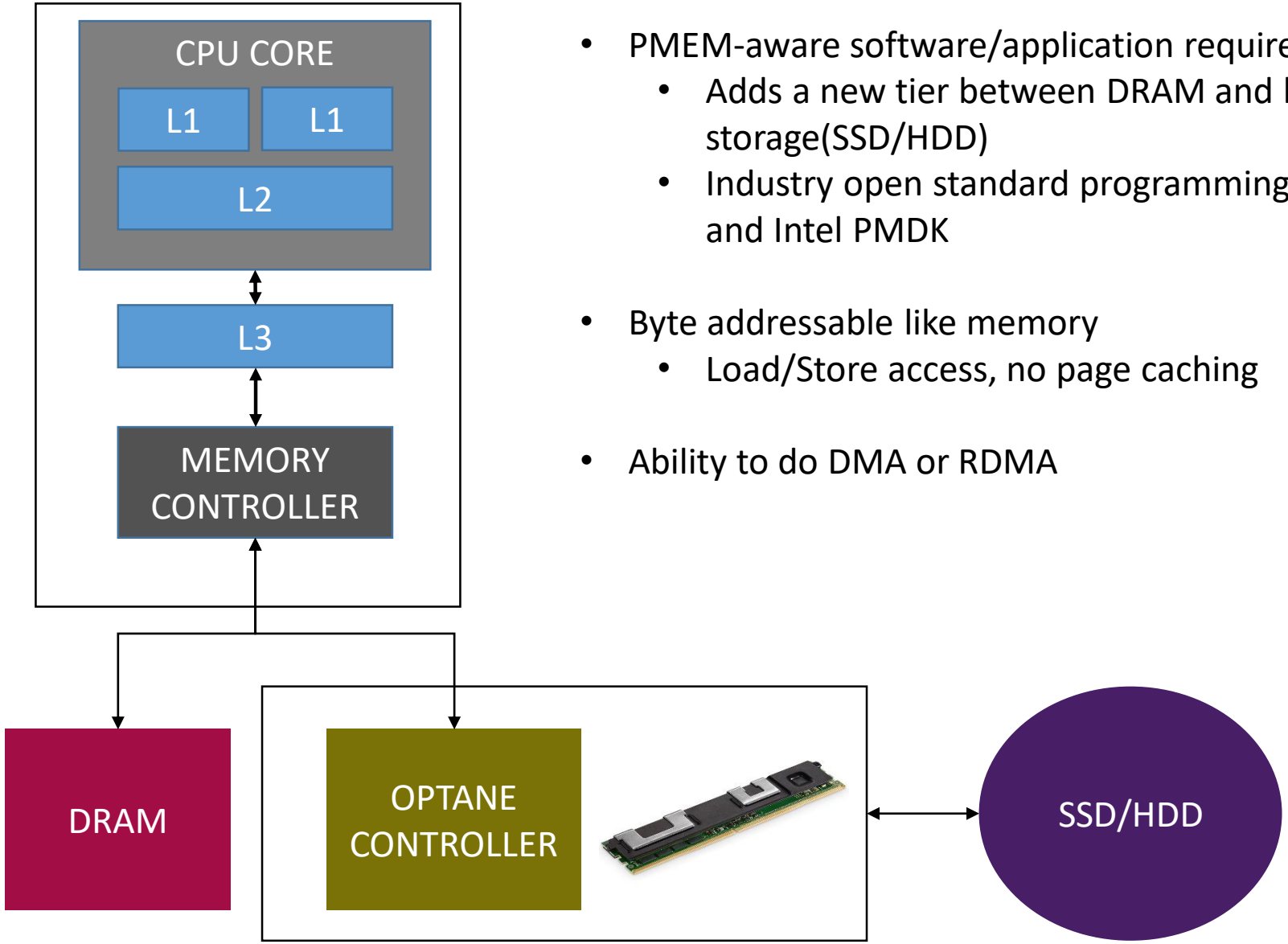
Memory Mode Transaction Flow



Case "Cache miss"

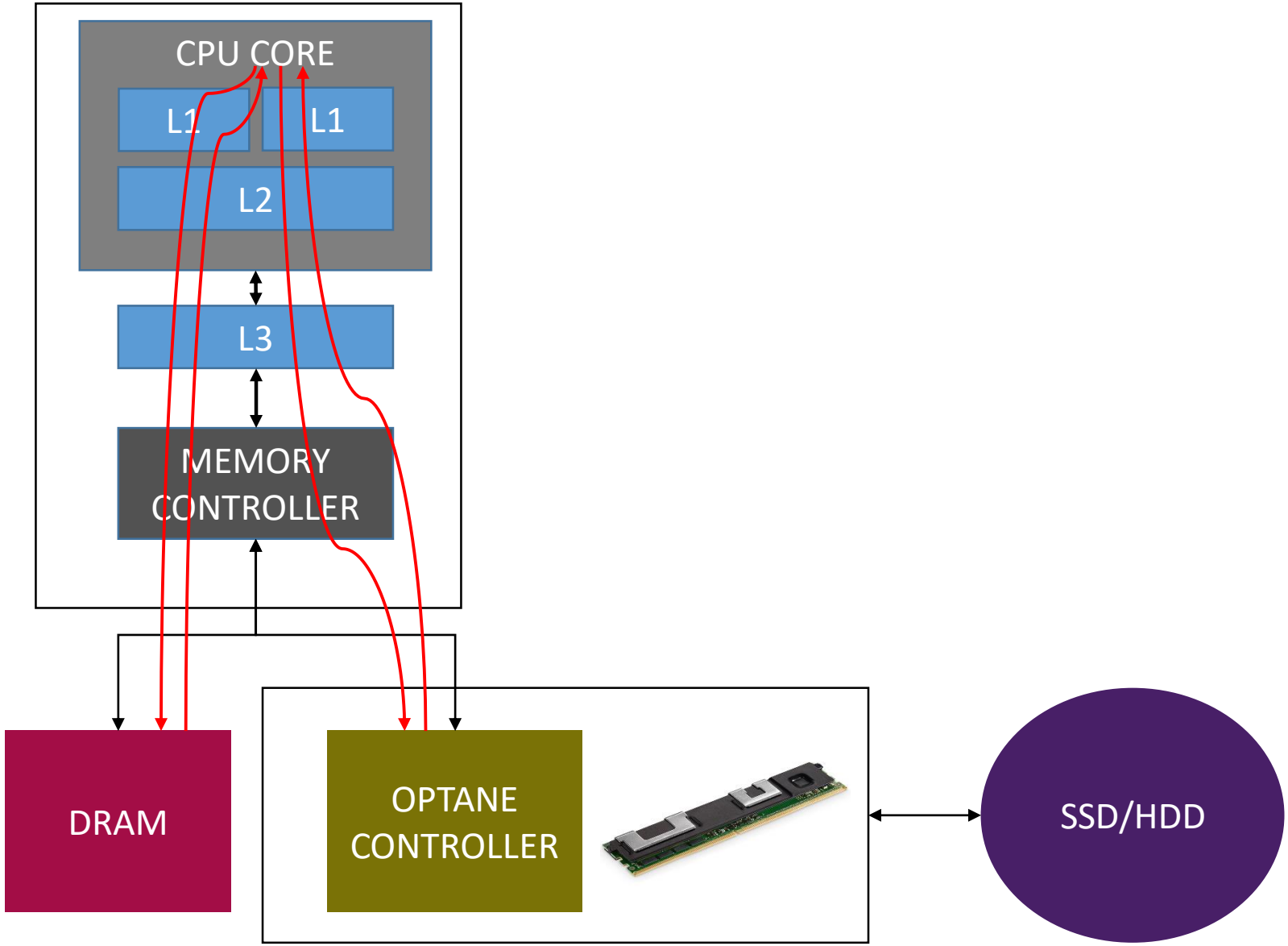
Overall Latency:
DRAM cache miss + Optane DC latency

Intel Optane DC: App Direct Mode



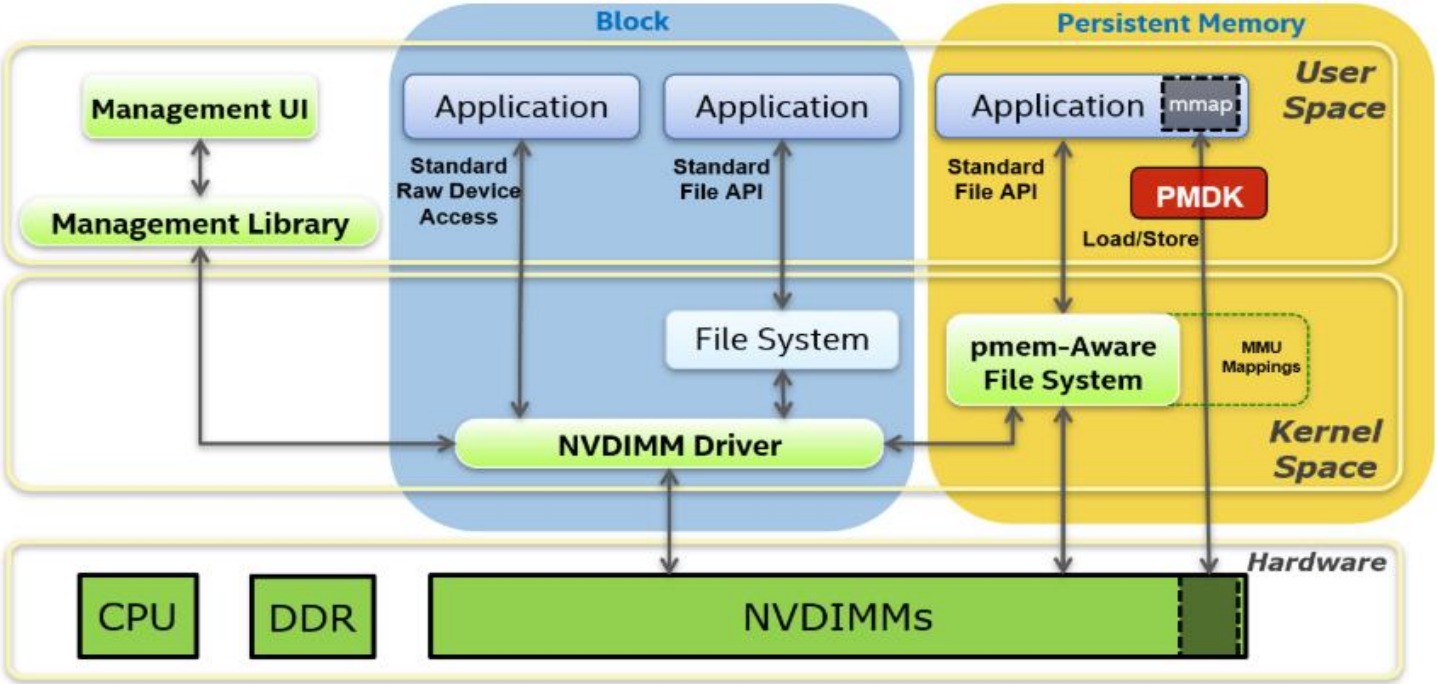
- PMEM-aware software/application required
  - Adds a new tier between DRAM and block storage(SSD/HDD)
  - Industry open standard programming model and Intel PMDK
- Byte addressable like memory
  - Load/Store access, no page caching
- Ability to do DMA or RDMA

App Direct Mode Transaction Flow



Intel Optane DC programming API : PMDK

How applications can access persistent memory devices?



Source: https://docs.pmem.io/persistent-memory/getting-started-guide/what-is-pmdk

- traditional POSIX standard APIs such as read, write, pread, and pwrite, or load/store operations such as memcpy
- application I/O bypasses existing filesystem page caches and goes directly to/from the persistent memory media

Intel Optane DC programming API : PMDK

- The **Persistent Memory Development Kit (PMDK),** formerly known as NVML, is a growing collection of libraries and tools
- Source code of PMDK: https://github.com/pmem/pmdk/
- PMDK offers application developers many libraries and features:
  - **libpmem:** provides low-level persistent memory support
  - **libpmemobj:** provides a transactional object store, providing memory allocation, transactions, and general facilities for persistent memory programming
  - **libpmemblk:** supports arrays of pmem-resident blocks, all the same size, that are atomically updated
  - **libpmemlog:** provides a pmem-resident log file
  - **libvmem:** turns a pool of persistent memory into a volatile memory pool, similar to the system heap but kept separate and with its own malloc-style API
  - **libvmmalloc:** library transparently converts all the dynamic memory allocations into persistent memory allocations
  - **libpmemtool:** provides support for off-line pool management and diagnostics
  - **librmem:** provides low-level support for remote access to persistent memory utilizing RDMA-capable RNICs.
  - **libvmemcache:** is an embeddable and lightweight in-memory caching solution. It's designed to fully take advantage of large capacity memory, such as Persistent Memory with DAX, through memory mapping in an efficient and scalable way

USAGE EXAMPLE: SAP HANA

- **SAP HANA** is an:
  - in-memory
  - column-oriented
  - relational database management system.

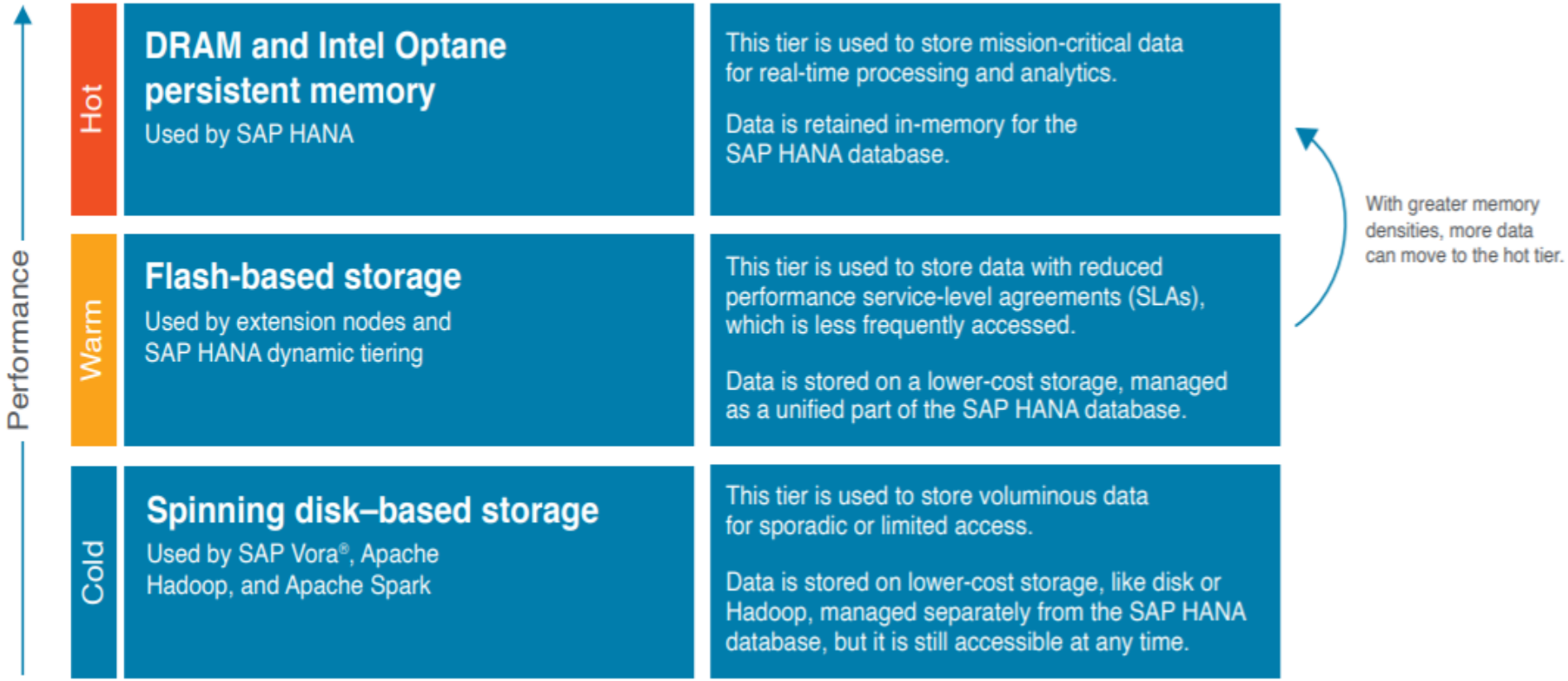
- Its primary function as a database server is to store and retrieve data as requested by the applications.

- Performs advanced analytics (predictive analytics, spatial data processing, text analytics, text search, streaming analytics, graph data processing) and includes extract, transform, load (ETL) capabilities as well as an application server.

Source: https://blogs.sap.com/2018/12/03/sap-hana-persistent-memory/

USAGE EXAMPLE: SAP HANA

- SAP HANA is the first major database platform that is specifically optimized for **Intel® Optane™ DC persistent memory**
- Uses it in App Direct Mode



Source: https://blogs.sap.com/2018/12/03/sap-hana-persistent-memory/

USAGE EXAMPLE: SAP HANA

	Before (DRAM + NAND SSD)	After (DRAM + Intel Optane Persistent Memory)
System Architecture	CPU, DRAM (Hot), SSD (Warm)	CPU, DRAM (Hot), Intel Optane PMem (Hot)
Data Access	Block Driver	Application Managed (App Direct)
Data Size	4 Kilobyte Block	64 Byte Cache Line
Read Latency (99.99%)	NAND SSD Range: 25,000–650,000 nanoseconds Typical: 70,000 nanoseconds	Intel Optane PMem Range: 100s–2,000 nanoseconds Typical: 100s nanoseconds

Faster start times for less downtime

6TB dataset in SAP HANA

Traditional System (with SSD storage)	50 min	
Persistent Memory	4 min	**12.5x** improvement

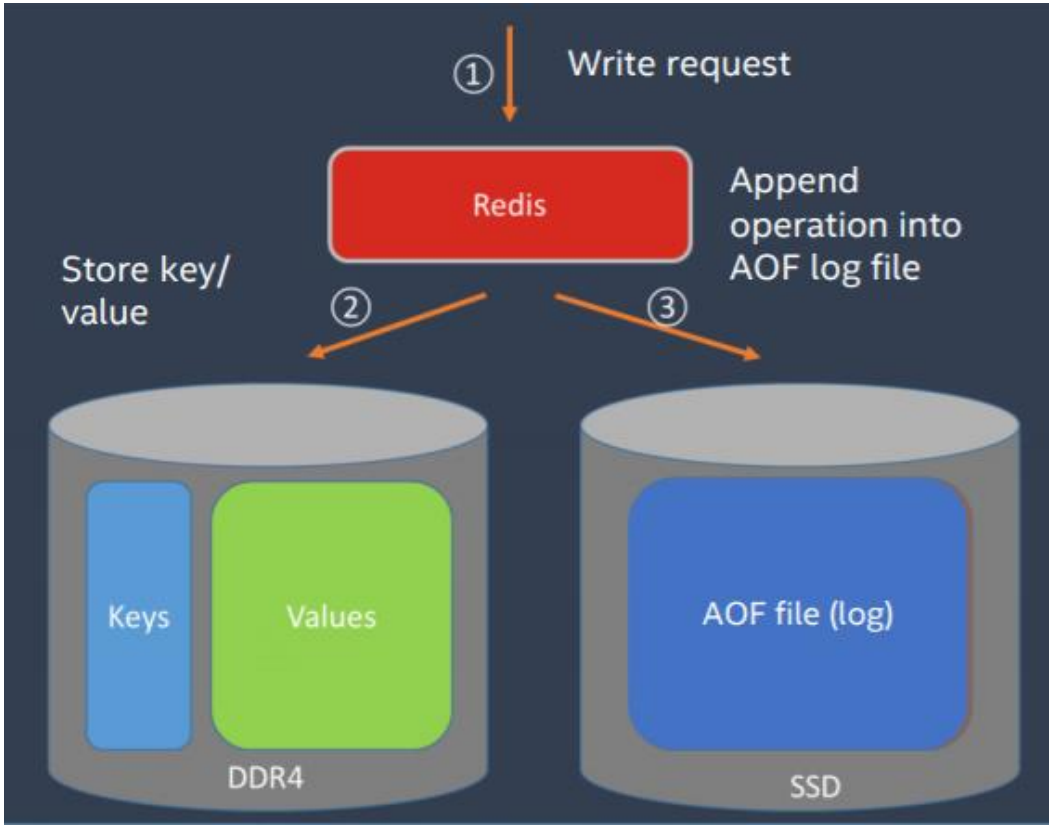
Increased memory capacity while reducing TCO

Memory capacity per CPU	**> 3 TB**

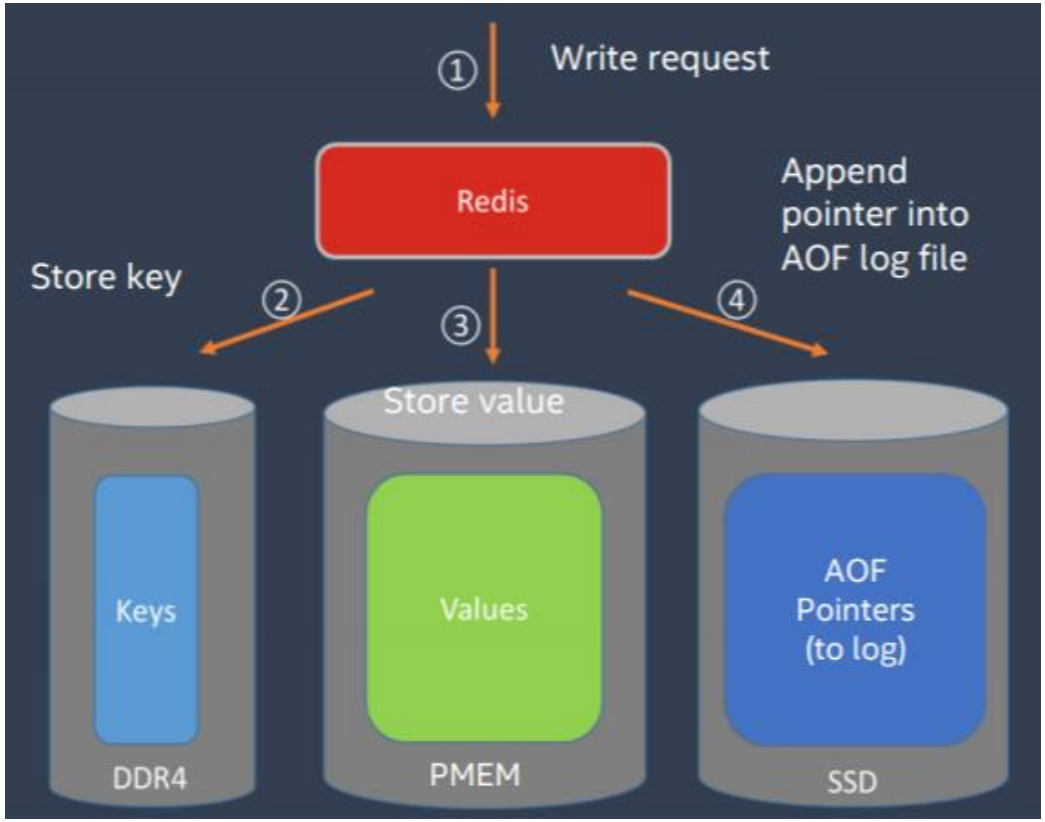
Source: https://blogs.sap.com/2018/12/03/sap-hana-persistent-memory/

USAGE EXAMPLE: REDIS

Redis is an open source (BSD licensed), in-memory data structure store, used as a database, cache and message broker



USAGE EXAMPLE: REDIS



- Reduce TCO by moving large portion of data from DRAM to Intel® Optane™ DC persistent memory.
- Optimize performance by using the values stored in persistent memory instead of creating a separate copy of the log in SSD (only pointer written to log):
  - Direct access vs. disk protocol
- Moving Value to App Direct reduces DRAM and optimizes logging by 2.27x

Overview

- Introduction - General about Non Volatile Memories (NVM)
- NVM Challenges
- Intel Optane DC Persistent Memory
- **Conclusion**

Conclusion

- Non Volatile Memories are an uprising trend in storage and memory
- Challenges in terms of computer architecture
- Challenges in terms of software design
- Intel Optane DC

Thank you!