



BY Developers FOR Developers

Storage Developer Conference
September 22-23, 2020

Exploring New Storage Paradigms and Opportunities with Persistent Memory Technology

Daniel Waddington Ph.D.
IBM Research Almaden



Disclaimer

© IBM Corporation 2020

THE INFORMATION CONTAINED IN THIS PRESENTATION IS PROVIDED FOR INFORMATIONAL PURPOSES ONLY. WHILE EFFORTS WERE MADE TO VERIFY THE COMPLETENESS AND ACCURACY OF THE INFORMATION CONTAINED IN THIS PRESENTATION, IT IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. IN ADDITION, THIS INFORMATION IS BASED ON IBM'S CURRENT PRODUCT PLANS AND STRATEGY, WHICH ARE SUBJECT TO CHANGE BY IBM WITHOUT NOTICE.

IBM SHALL NOT BE RESPONSIBLE FOR ANY DAMAGES ARISING OUT OF THE USE OF, OR OTHERWISE RELATED TO, THIS PRESENTATION OR ANY OTHER DOCUMENTATION.

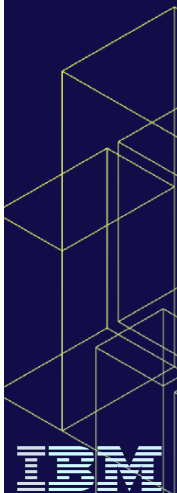
NOTHING CONTAINED IN THIS PRESENTATION IS INTENDED TO, OR SHALL HAVE THE EFFECT OF:

- CREATING ANY WARRANTY OR REPRESENTATION FROM IBM (OR ITS AFFILIATES OR ITS OR THEIR SUPPLIERS AND/OR LICENSORS); OR
- ALTERING THE TERMS AND CONDITIONS OF THE APPLICABLE LICENSE AGREEMENT GOVERNING THE USE OF IBM SOFTWARE.

IBM's statements regarding its plans, directions, and intent are subject to change or withdrawal without notice at IBM's sole discretion. Information regarding potential future products is intended to outline our general product direction and it should not be relied on in making a purchasing decision. The information mentioned regarding potential future products is not a commitment, promise, or legal obligation to deliver any material, code or functionality. Information about potential future products may not be incorporated into any contract. The development, release, and timing of any future features or functionality described for our products remains at our sole discretion.

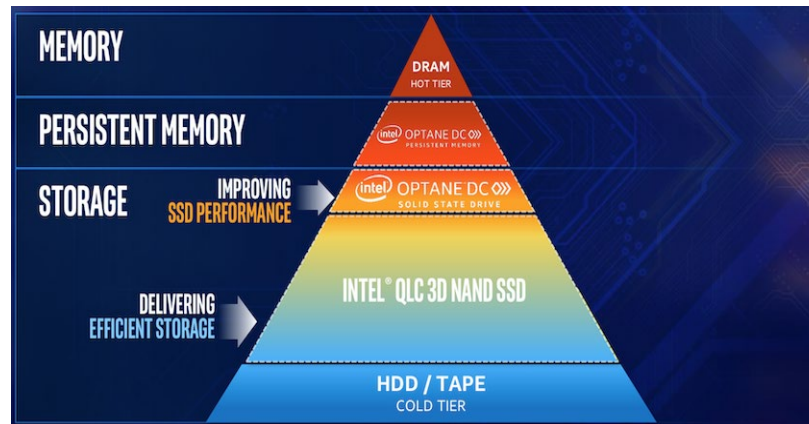
Outline

- What does Persistent Memory really bring to the table?
- Re-thinking Storage
 - Memory Centric Active Storage (MCAS)
- Future Directions



Intel Optane DC PMM (3DXP)

- Intel/Micron 3D-Xpoint is the first-in-breed “Storage Class Memory” designed for the enterprise storage domain
- 3DXP is based on a lattice-arranged Phase-Change-Memory (PCM)
- Intel Optane DC *Persistent Memory* Modules offers DIMM modules that are attached to the memory-bus
- ~3x Slower than DRAM
- 8x capacity than DRAM, at ½ cost per GB
- Current version up to 512GB DIMMs

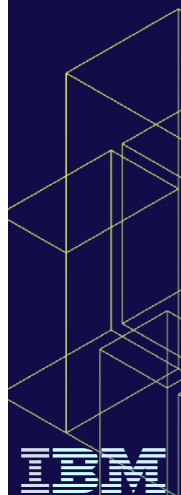


Assessment of Persistent Memory

- Using Intel 3DXP Optane NVDIMMs as a reference point

Key +ve Attributes:

- Load/store accessible e.g. 64B ~ 300ns access latency
→ synchronous access model → s/w design simplification
- Minimal media controller (e.g., no queues, no garbage collection)
→ *both* low latency and high throughput at Queue Depth 1
→ synchronous access model → s/w design simplification
- Small(er) read-write amplification (64B CPU-side, 256B internally)
→ higher-performance for sparse data access
→ simplifies data structures (e.g., avoid need LSM-like structures)
- More predictable → shorter long-tail latency



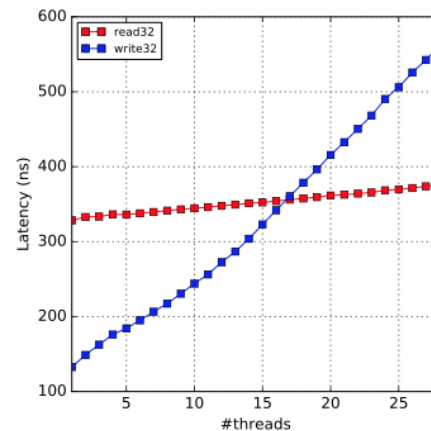
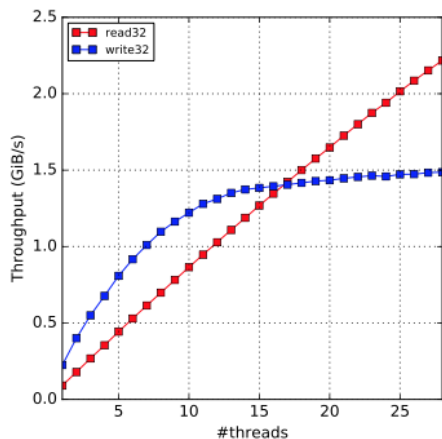
Assessment of Persistent Memory

- Using Intel 3DXP Optane NVDIMMs as a reference point

Key -ve Attributes:

- Asymmetric read/write random/seq. performance
- Writes do not scale with thread count
- Performant use as a block device requires PM-aware file system (e.g. NOVA)

Raw Optane PMM 32-byte random

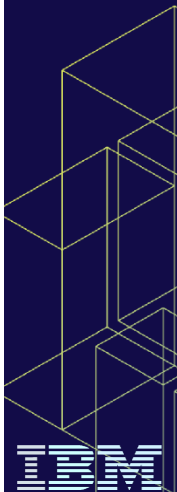


Assessment of Persistent Memory

- Using Intel 3DXP Optane NVDIMMs as a reference point

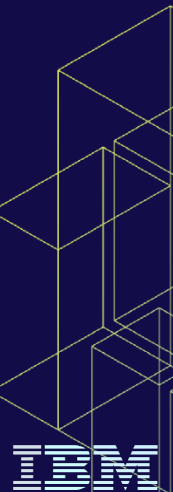
Key -ve Attributes:

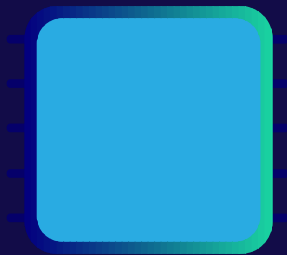
- Fine-grained power-fail consistency 64b aligned
 - onus of software to manage crash-consistency
 - performance loss
- Good performance relies on striping data across multiple DIMMs
- Simple DIMM striping model (e.g., no inter-DIMM erasure coding)
 - limited reliability, availability and serviceability (**RAS**),



Where (3DXP) Persistent Memory Excels

- Small unpredictable access patterns
- Where low latency to read and modify data (make accessible to the CPU) is important
- More reads than writes
- Where rebuild and recovery cost can be reduced by using locally durable and consistent data
- Lower cost, performance and energy capacity 12x128GB = 1.5TB
- Higher cost, performance and energy capacity 12x512GB = 6TB



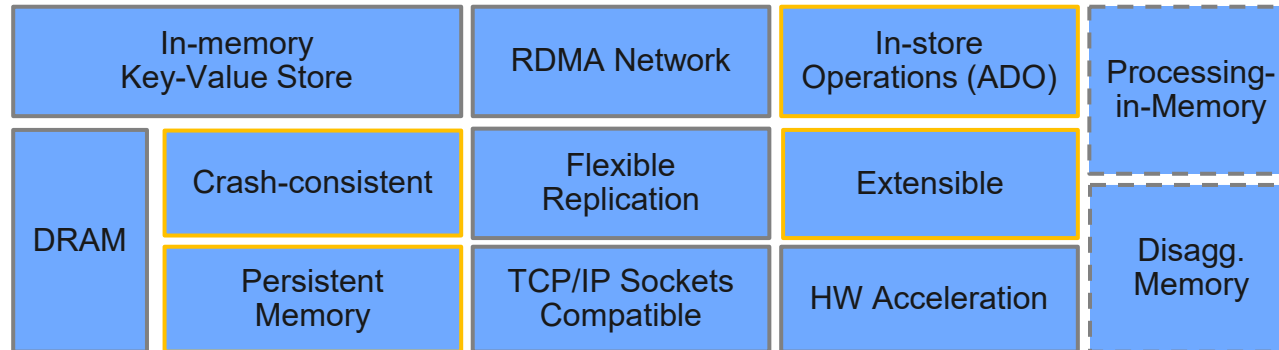


MCAS
REDEFINING STORAGE

IBM's Memory Centric Active Storage

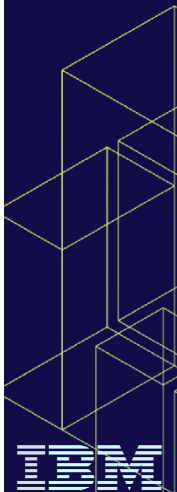
MCAS

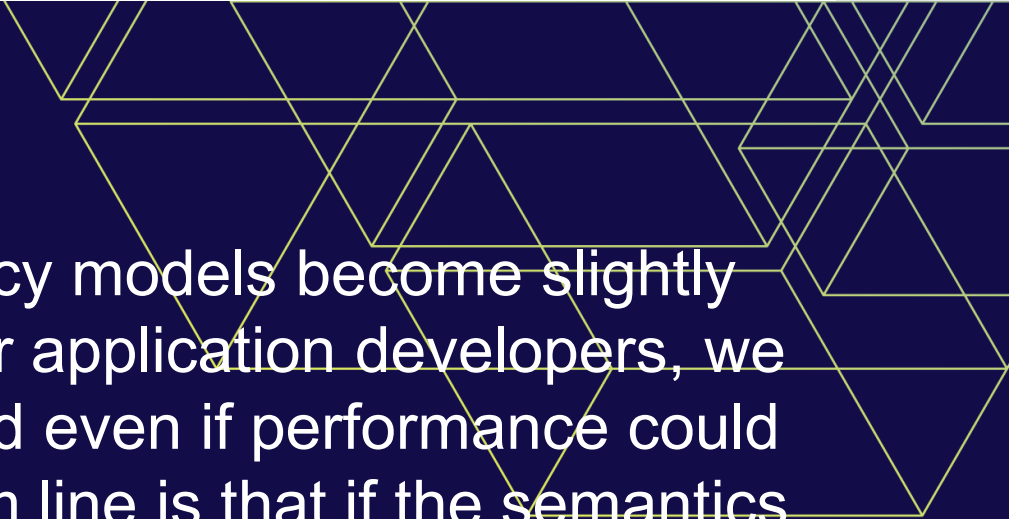
- IBM Research (Almaden) open source project started in 2018
 - <https://github.com/IBM/mcas/>
- Focus on building a new type of storage system with the explicit goal of leveraging new persistent memory technology



MCAS Founding Tenets

- Underlying design is a key-value paradigm with extensions for in-store compute
- Synchronous operations with “immediate” consistency
 - when a put command returns, all data has been made *fully* persistent
- Storage is network-attached for data and resource sharing
- Aimed at fast networks in the data center
 - maintain immediate consistency whenever possible
- Allow arbitrary (sandboxed) in-store operations
- Allow flexibility in implementing crash-consistency for stored data structures
 - by-hand, s/w transactions, h/w transactions, compiler/language
- Avoid memcpy for large transfers and allow direct-DMA





"As soon as consistency models become slightly difficult to understand for application developers, we see that they are ignored even if performance could be improved. The bottom line is that if the semantics of a consistency model are not intuitively clear, application developers will have a hard time building correct applications"

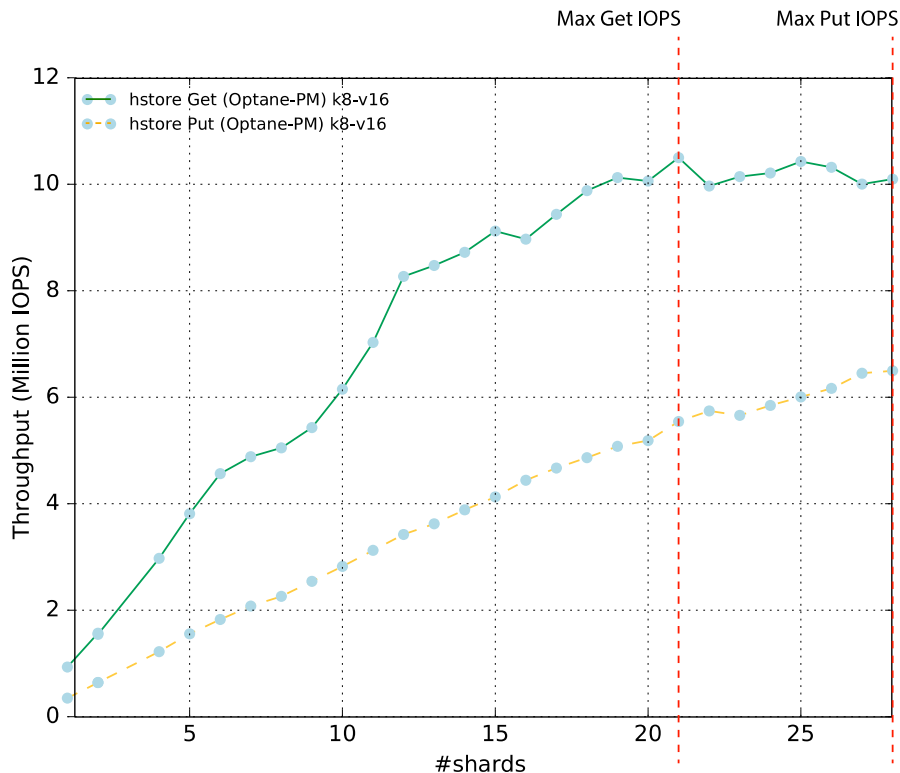
- Andrew Tanenbaum, Maarten Steen

MCAS Base Key-Value Performance

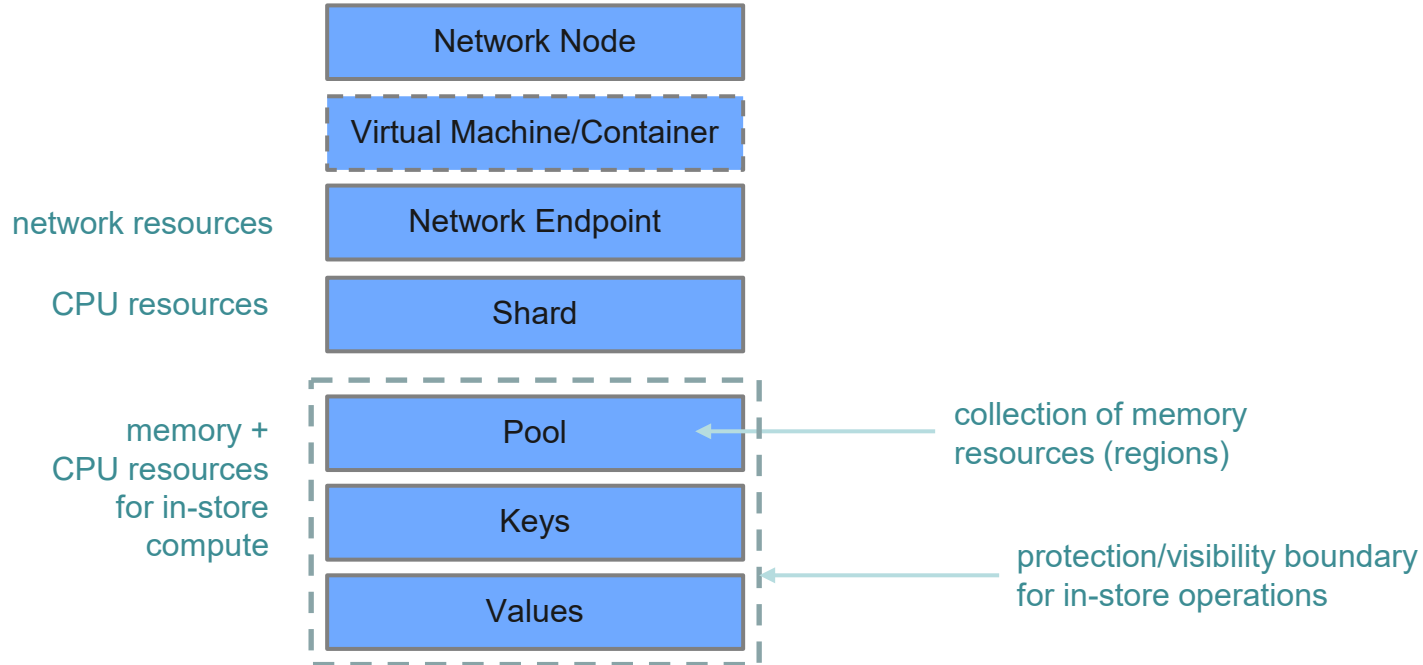
- Without in-store operations
- Full details in ACM/IEEE ISPASS 2020 paper

Single socket, Intel Cascade Lake:

- Random read k8-v16 at ~10M IOPS
- Random write k8-v16 at ~6M IOPS
- ~7us Put RTT

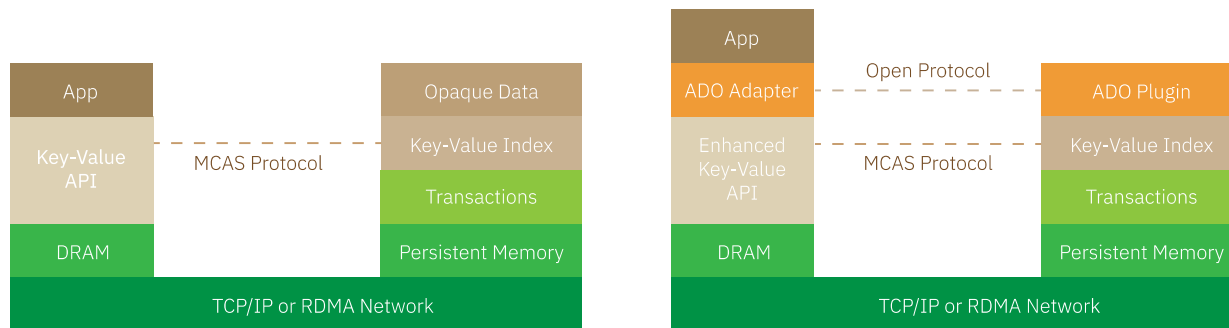


Sharding Architecture



MCAS Open Protocol

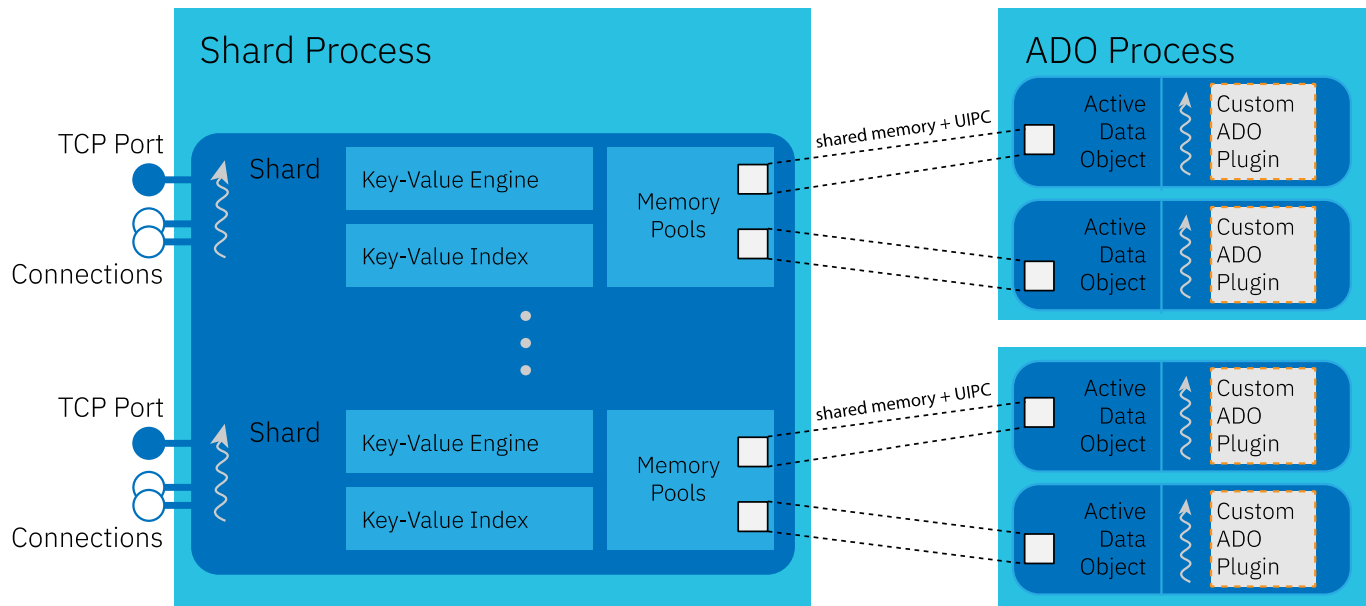
- MCAS provides an open protocol architecture for layering “services” and in-store compute (aka push-down) – Active Data Objects (ADO)
- Approach reduces data movement across the network



- Generic ADO layers: e.g., replication, compression, encryption, erasure coding, snap-shots, tiering...
- Domain-specific ADO layers: e.g., data manipulation, conversion, summarization, data derivation, domain operations (e.g., matrix multiply)

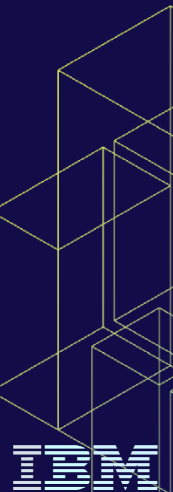
MCAS Architecture

- Flexible deployment on bare metal, VM or container
- C++ primary API, Python secondary

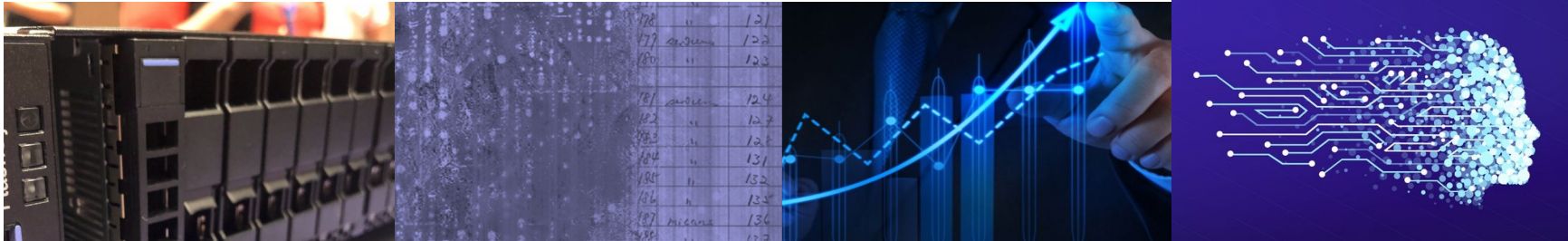


Writing ADO Plugins

- Plugins implement a defined C++ API
- Currently only supports dev-DAX (expect fs-DAX soon)
- Crash-consistency is achieved through either hand-crafting or using s/w transaction instrumented STL containers
- Future work to explore language/compiler based approaches and h/w support for transparent undo logging



Potential MCAS Use-Cases



Storage Services

- versioning/TTL
- encryption/CRC
- logging
- durability
- event notification
- durable metadata

Data Curation

- sorting, filtering
- EDI transform
- real-time compression and decompression
- summary operators

Real-time Data Analytics

- domain specific data structures (e.g., k-d tree)
- core math operations (e.g., mat mul , fft)

Real-time Cognitive

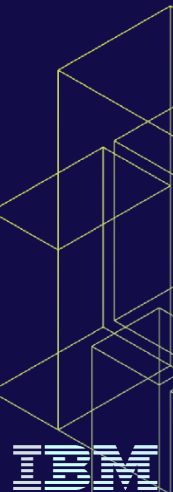
- graph processing
- inferencing
- sparse distributed memory / HTM
- NLP



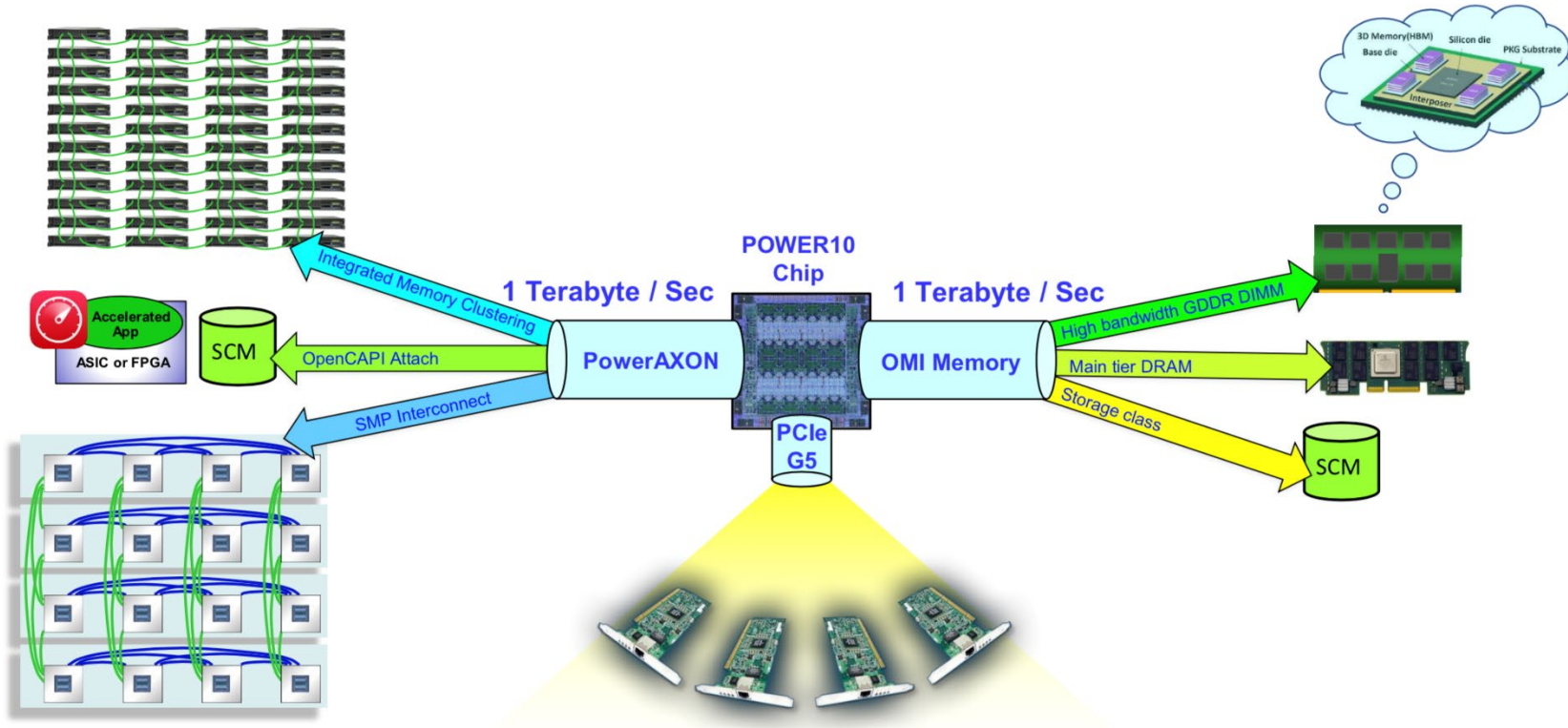
Future Directions for Persistent Memory

Opening the Memory Bus

- Key emerging h/w interconnects
 - IBM OpenCAPI Memory Interface (OMI) / PowerAxon
 - CXL (Compute eXpress Link)
- Cache-coherent attached memory and accelerators
 - drive broader multi-vendor adoption of PM
 - enable CPU-memory disaggregation flexibility
 - allow (custom) “intelligence” to be pushed closer to memory

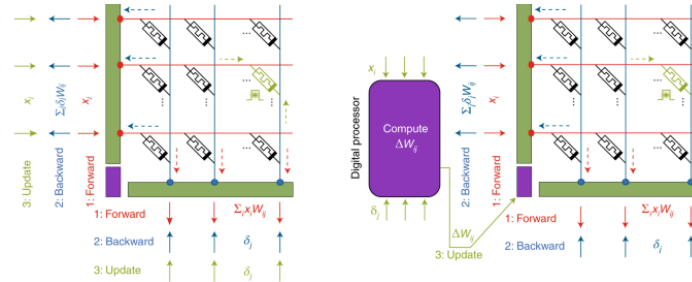


IBM Power 10



Dark Data

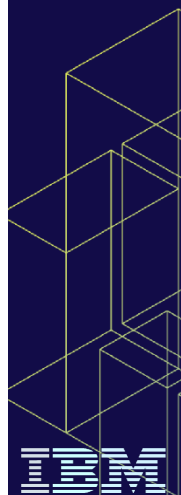
- Capacity scaling out-paces compute and network scaling
 - Need to minimize data movement
 - Look to Processing-In-Memory (PIM) for certain types of data
 - Processing-Near-Memory (e.g., UPMEM)
 - Processing-Using-Memory for resistive memories (e.g., PRIME, GraphPIM)
 - Academic examples or PUM around graph-processing, pointer chasing, genomics, hyper-dimensional computing etc.



https://users.ece.cmu.edu/~saugatag/papers/19ibmjrd_pim.pdf
<https://www.nature.com/articles/s41565-020-0655-z>

Programming Languages

- Making crash-consistent data structures and operations easier for the developer
 - eliminate need for data structure serialization (memory-storage convergence)
- Moving away from frameworks (e.g., PMDK) to compilers and language standardization
 - reduce the need to re-write or refactor legacy s/w
- Leveraging of existing “sandboxing” technologies to maintain integrity of in-place modification
 - e.g., RUST/Splinter type approach, WASM
- In cooperation with additional h/w support (e.g., transparent undo/redo logging in the memory controller)

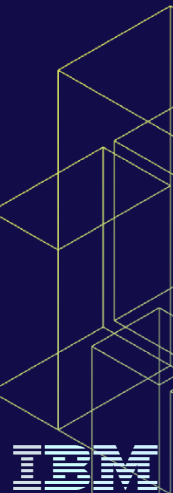


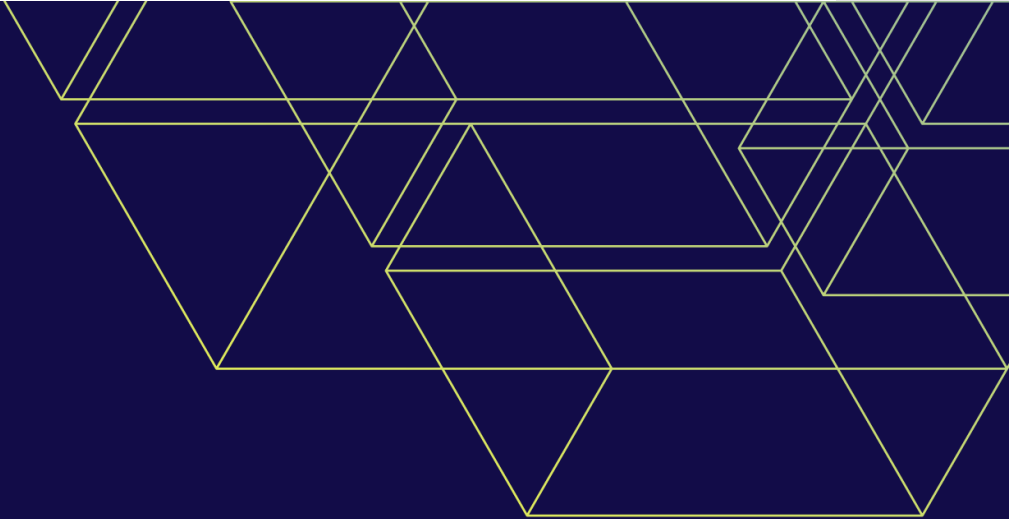
Summary

- Potential of Persistent Memory (PM) technology is starting to be unlocked
- PM isn't (currently) good at everything but is very good at handling certain types of data
- IBM's Memory Centric Active Storage takes a new look at storage design with PM in mind taking traditional key-value paradigm and extending with PM + in-store compute
- Emerging interconnect technologies will likely increase opportunity and volume of PM adoption
- Processing-in-Memory (PIM) concepts could help with compute scaling

MCAS Availability

- Open Source Apache 2.0 licensed
 - <https://github.com/IBM/mcas/>
- Early research prototype
- Welcome new use-cases and contributions





**Please take a moment
to rate this session.**

Your feedback matters to us.