What We Can Learn from Persistent Memory for CXL

Lawrence Benson[1], Marcel Weisgut[1], Tilmann Rabl[1]

Abstract:

With high-capacity Persistent Memory (PMem) entering the long-established data center memory hierarchy, various assumptions about the performance and granularity of memory access have been disrupted. To adapt existing applications and design new systems, research focused on how to efficiently move data between different types of memory, how to handle varying access latency, and how to trade off price for performance. Even though Optane is now discontinued, we expect that the insights gained from previous PMem research apply to future work on Compute Express Link (CXL) attached memory. In this paper, we discuss how limited hardware availability impacts the performance generalization of new designs, how existing CPU components are not adapted towards different access characteristics, and how multi-tier memory setups offer different price-performance trade-offs. To support future CXL research in each of these areas, we discuss how our insights apply to CXL and which problems researchers may encounter along the way.

1 Introduction

With the arrival of Intel Optane Persistent Memory (PMem) in 2019, research on new data management techniques for byte-addressable persistent memory increased significantly. Among other questions, this research investigates how to handle varying memory access latency, how to place data based on available capacity, and how to design for memory access sizes larger than a single cache line but smaller than a page [BMR21; Lu20; Re18]. However, in 2022, Intel announced that their Optane product line will be discontinued in favor of recent trends toward Compute Express Link (CXL) [GZ22]. We expect that while Optane was abandoned, research based on it still provides valuable insights.

In light of Intel citing CXL as one of the reasons for ending Optane, in this paper, we raise the question: "What can we learn from PMem research for future CXL research?" Based on benchmarks that we conducted in previous work on PerMA-Bench [BPR22], a configurable benchmark framework for PMem access, we look at three insights from PMem that also apply to future research on CXL.

First, we discuss how limited hardware access can lead to solutions that are too specialized for one hardware configuration or not specialized enough. We then discuss how existing CPU components interact with new memory types, based on the prefetching behavior with PMem. Finally, we show that different memory types offer different price-performance trade-offs depending on the use case. Even though CXL-attached memory is not yet generally available, these insights highlight some challenges that future research faces when integrating new memory types into a long-established memory hierarchy.

[1] Hasso Plattner Insitut, Universität Potsdam, Germany. {first.last}@hpi.de

doi:10.18420/BTW2023-48

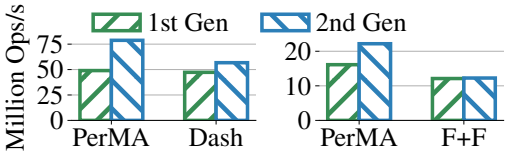

Fig. 1: Performance of PerMA and actual index implementation of Dash and FAST+FAIR.

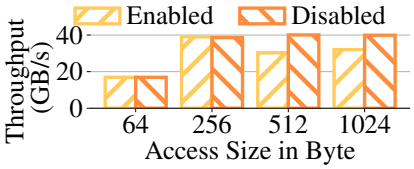

Fig. 2: Impact of prefetcher on PMem random read bandwidth.

2 Transferring Insights from PMem to CXL-Attached Memory

In this section, we discuss how insights derived from PMem transfer to future CXL research.

Persistent Memory.[2] PMem can be used as a volatile DRAM extension (*Memory Mode*) or explicitly as PMem beside DRAM (*App Direct Mode*). When using PMem in App Direct Mode, PMem and DRAM share the application's unified virtual address space, i.e., data stored in both types of memory can be prefetched by the CPU. Data in PMem is accessed via `load/store` instructions issued in the CPU. Through a modified DDR-4 interface (called DDR-T), the CPU communicates with PMem DIMMs in 64 Byte cache lines. However, Optane's internal media access occurs at 256 Byte, causing read and write amplification for access smaller than 256 Byte. PMem's access latency for random reads and writes from and to PMem is ~2-5× higher than DRAM's and read/write bandwidth is ~2.5/5× lower. Thus, even though DRAM and PMem share the same interface, applications designed for PMem have to account for its worse performance. As future CXL-attached memory will have higher latency and, for CXL versions 1.1 and 2.0, lower bandwidth than CPU-attached memory (bound by PCIe 5.0 compared to memory channels), CXL designs face similar issues as PMem designs.

Benchmark Setup. We evaluate two servers with 256 GB PMem DIMMs of the first and second-generation Optane. The first generation server contains a Cascade Lake CPU with 18 cores and six PMem DIMMs at 2933 MT/s. The second generation server contains an Ice Lake CPU with 32 cores and eight PMem DIMMs at 3200 MT/s. Both systems run Ubuntu 20.04 with a 5.4 kernel.

Application Tailoring. To understand how well applications utilize the hardware, we compare the `lookup()` performance of PMem index structures modeled in PerMA-Bench with the actual index implementations. The results obtained with PerMA-Bench show a performance upper-bound, as they include only memory access without computation or branching logic. The results for the hash index Dash [Lu20] and the B-Tree index FAST+FAIR [Hw18] are shown in Figure 1. The memory access for Dash is modeled as a 512 Byte random read, as Dash reads two consecutive 256 Byte hash buckets to find an entry. For FAST+FAIR, we issue 3× random 512 Byte reads that represent B-Tree node lookups. The experiments are run with 16 threads. We observe that while the underlying bandwidth improves across generations, the indexes do not (fully) utilize this. Unlike on

[2] For a more in-depth introduction to PMem, we refer to [BPR22].

the first generation server, Dash spends ~20% of all cycles on non-memory access on the second generation server, which contains more PMem DIMMs, a newer CPU, and better DRAM. Due to the high price of Optane, researchers often do not have access to different setups, which leads to tailoring the application towards only a single setup and, in turn, does not always generalize. On the other hand, we see FAST+FAIR as an index designed for general PMem before Optane became available. The improved memory bandwidth does not translate to the index, as FAST+FAIR spends a lot of time on heavy-weight locking and inefficient PMem access on patterns. As FAST+FAIR was designed pre-Optane, we see that the system is not tailored enough to the underlying hardware, and performance is lost.

Insight 1: Due to limited hardware availability, systems are tailored too much toward a single setup or not tailored enough toward the actual hardware. Through the CXL abstraction, future systems will cover a wider range of memory performance characteristics. Thus, it is important to design and research robust systems that generalize across different hardware and multiple memory tiers.

Prefetching. As a new layer in the long-established memory hierarchy, it is important to understand how well PMem interacts with existing CPU components, which are optimized for caches and DRAM. In Figure 2, we show the impact of the hardware prefetchers for random memory reads on the second-generation Optane server. We en-/disable all hardware prefetchers and run on 16 threads. We see that for small access sizes (< 256 Byte), the prefetcher does not impact performance, i.e., the prefetcher does not prefetch. However, for 512 and 1024 Byte access, the prefetcher speculatively loads unnecessary data not accessed by the user in the background, reducing the available bandwidth for requested reads. We observe this in hardware performance counters, where the underlying bandwidth utilization is identical in both runs, but the effective bandwidth in the application is not. Thus, disabling the prefetcher actually improves performance in this case. This effect is also observable for 2048 Byte access but not for 4096 Byte or more [Da21], as page-size access is a well-understood and optimized pattern in DRAM. As Optane's internal access occurs at 256 Byte granularity, most applications design access in multiples of 256 Byte. As a consequence, a 512 Byte random access to Optane, e.g., a node lookup in FAST+FAIR, spans only two Optane "cache lines", which should not trigger prefetching. However, the prefetcher views these 512 Byte as regular DRAM access, spanning eight consecutive cache lines, and starts prefetching for sequential access.

Insight 2: Prefetchers are highly optimized toward 64 Byte DRAM cache line access, and CXL specifies 64 Byte transfers in the transaction layer [Co22, p. 167]. However, CXL abstracts from the underlying device, i.e., it could support Optane PMem or other memory devices, and memory behind CXL may not be accessible in 64 Byte granularity. As all CPU- and CXL-attached memory is available in the same unified virtual address space, prefetchers operate on both types of memory. Future research should investigate how existing components, like the prefetcher, interact with memory that is not attached directly to the CPU and has different access characteristics. However, unlike Insight 1, this cannot be solved by applications alone and most likely requires hardware changes as well.

	€/GB capacity	seq. read	rnd. read	seq. write	rnd. write
PMem	**12.77**	**0.22**	**0.33**	**0.60**	2.12
DRAM	59.37	0.38	0.46	0.70	**0.91**

Tab. 1: Price-performance of PMem and DRAM. Read/write values in €/GB/s. Calculation based on listing prices from dell.de in February 2022.

Price-Performance. In Table 1, we show the price-performance for basic sequential/random read/write access in PMem and DRAM on the same second-generation Optane server while disregarding persistence. The data access-related prices per GB of throughput are normalized to the device's price per GB to avoid including the higher price for larger capacity. We see that the price per GB capacity is significantly lower for the PMem DIMMs than for the high-end DRAM DIMMs. As PMem is not available in cloud vendors, we base our calculations on the list price on dell.de [De22] in February 2022. Focusing on the relative scale between the listed prices rather than on the exact monetary values, for sequential access and random reads, we observe that PMem has a better price-performance ratio, as the bandwidth is often only 2–3× worse while the price per GB capacity is about 5× better. DRAM outperforms PMem only for 64 Byte random writes, where PMem bandwidth is very low because of high write amplification.

Insight 3: For applications that do not require peak performance or persistence, PMem can be used as a cheaper DRAM alternative with significantly higher capacity. As increasing memory capacity is a selling point of CXL, future research should investigate the price-performance trade-off in multi-tier memory setups for slower and potentially cheaper CXL-attached memory.

3 Conclusion

With PMem, various assumptions about the homogeneity of DRAM access have been disrupted, leading to new challenges and designs. In this paper, we discussed how insights from these designs also apply to future CXL research. New CXL-based approaches need to focus on performance generalizability under initially limited hardware availability. They should consider how interaction with long-established components, such as prefetchers, impacts performance. And finally, in multi-tier memory setups, new designs should consider their economic viability as a key trade-off. While PMem is discontinued for now, we hope that future CXL work builds on these insights to establish a more general understanding of how systems interact with multi-tier memory.

Acknowledgements: This work was partially funded by the German Ministry for Education and Research (01IS18025A/01IS18037A), the German Research Foundation (414984028), and the European Union's Horizon 2020 research and innovation programme (957407).

References

[BMR21] Benson, L.; Makait, H.; Rabl, T.: Viper: An Efficient Hybrid PMem-DRAM Key-Value Store. en, Proceedings of the VLDB Endowment 14/9, pp. 1544–1556, 2021.

[BPR22] Benson, L.; Papke, L.; Rabl, T.: PerMA-bench: benchmarking persistent memory access. en, Proceedings of the VLDB Endowment 15/11, pp. 2463–2476, July 2022, ISSN: 2150-8097, URL: https://dl.acm.org/doi/10.14778/3551793.3551807, visited on: 11/22/2022.

[Co22] Compute Express Link Consortium, I.: Compute Express Link (CXL) Specification, Revision 3.0, Version 1.0, https://www.computeexpresslink.org/download-the-specification, 2022.

[Da21] Daase, B.; Bollmeier, L. J.; Benson, L.; Rabl, T.: Maximizing persistent memory bandwidth utilization for OLAP workloads. In: SIGMOD '21. tex.ids=daaseMaximizingPersistentMemory, ACM, 2021.

[De22] Dell Technologies: Dell Rack Servers, https://www.dell.com/de-de/work/shop/deals/enterprise-deals/poweredge-rack-server-deals, 2022.

[GZ22] Gelsinger, P.; Zinsner, D.: Earnings Call Comments from CEO Pat Gelsinger and CFO Dave Zinsner, https://www.intel.com/content/www/us/en/newsroom/news/intel-reports-first-quarter-2022-financial-results.html, 2022.

[Hw18] Hwang, D.; Kim, W.-H.; Won, Y.; Nam, B.: Endurable Transient Inconsistency in Byte-Addressable Persistent B+-Tree. In. Pp. 187–200, 2018, ISBN: 978-1-931971-42-3, URL: https://www.usenix.org/conference/fast18/presentation/hwang, visited on: 10/22/2021.

[Lu20] Lu, B.; Hao, X.; Wang, T.; Lo, E.: Dash: scalable hashing on persistent memory. Proceedings of the VLDB Endowment 13/8, pp. 1147–1161, Apr. 2020, ISSN: 2150-8097, URL: https://doi.org/10.14778/3389133.3389134, visited on: 07/06/2020.

[Re18] van Renen, A.; Leis, V.; Kemper, A.; Neumann, T.; Hashida, T.; Oe, K.; Doi, Y.; Harada, L.; Sato, M.: Managing Non-Volatile Memory in Database Systems. In: SIGMOD '18. SIGMOD '18, ACM, Houston, TX, USA, pp. 1541–1555, May 2018, ISBN: 978-1-4503-4703-7, URL: https://doi.org/10.1145/3183713.3196897, visited on: 07/06/2020.