Keynote: Rethinking Memory System Design

Onur Mutlu

ETH Zürich and Carnegie Mellon University

Abstract

The memory system is a fundamental performance and energy bottleneck in almost all computing systems. Recent system design, application, and technology trends that require more capacity, bandwidth, efficiency, and predictability out of the memory system make it an even more important system bottleneck [36, 38]. At the same time, DRAM and flash technologies are experiencing difficult technology scaling challenges that make the maintenance and enhancement of their capacity, energy efficiency, and reliability significantly more costly with conventional techniques (see, for example [13, 17–19, 23, 25, 26, 29, 30, 40]). In fact, recent reliability issues with DRAM, such as the RowHammer problem [23], are already threatening system security, predictability and robustness.

In this talk, we first discuss major challenges modern memory systems face in the presence of increasing demand for data and its fast analysis. We then examine some promising research and design directions to overcome these challenges and enable scalable memory systems for the future. We discuss three key solution directions: 1) enabling new memory architectures, functions, interfaces, and better integration of memory and the rest of the system (e.g., [1, 2, 4, 12, 22, 27, 28, 32, 42–44]), 2) designing a memory system that intelligently employs emerging non-volatile memory (NVM) technologies (e.g., [24–26, 31, 33, 41, 49–51]), 3) reducing memory interference and providing predictable performance to applications sharing the memory system (e.g., [3, 14–16, 20–22, 35, 37, 39, 45–48]). If time permits, we will also touch upon our ongoing related work in combating scaling challenges of NAND flash memory (e.g., [5–11, 34]).

References

[1] J. Ahn et al. PIM-Enabled Instructions: A Low-Overhead, Locality-Aware Processing-in-Memory Architecture. In *ISCA*, 2015.

[2] J. Ahn et al. A Scalable Processing-in-Memory Accelerator for Parallel Graph Processing. In *ISCA*, 2015.

[3] R. Ausavarungnirun et al. Staged memory scheduling: Achieving high performance and scalability in heterogeneous systems. In *ISCA*, 2012.

[4] A. Boroumand et al. LazyPIM: An Efficient Cache Coherence Mechanism for Processing-in-Memory. *CAL*, 2016.

[5] Y. Cai et al. Error patterns in MLC NAND flash memory: Measurement, characterization, and analysis. In *DATE*, 2012.

[6] Y. Cai et al. Flash Correct-and-Refresh: Retention-aware error management for increased flash memory lifetime. In *ICCD*, 2012.

[7] Y. Cai et al. Threshold voltage distribution in MLC NAND flash memory: Characterization, analysis and modeling. In *DATE*, 2013.

[8] Y. Cai et al. Program interference in MLC NAND flash memory: Characterization, modeling, and mitigation. In *ICCD*, 2013.

[9] Y. Cai et al. Neighbor-cell assisted error correction for MLC NAND flash memories. In *SIGMETRICS*, 2014.

[10] Y. Cai et al. Read Disturb Errors in MLC NAND Flash Memory: Characterization, Mitigation, and Recovery. In *DSN*, 2015.

[11] Y. Cai et al. Data retention in MLC NAND flash memory: Characterization, optimization and recovery. In *HPCA*, 2015.

[12] K. Chang et al. Low-Cost Inter-Linked Subarrays (LISA): Enabling Fast Inter-Subarray Data Movement in DRAM. In *HPCA*, 2016.

[13] H. David et al. Memory power management via dynamic voltage/frequency scaling. In *ICAC*, 2011.

[14] E. Ebrahimi et al. Fairness via source throttling: a configurable and high-performance fairness substrate for multi-core memory systems. In *ASPLOS*, 2010.

[15] E. Ebrahimi et al. Prefetch-aware shared-resource management for multi-core systems. In *ISCA*, 2011.

[16] E. Ebrahimi et al. Parallel application memory scheduling. In *MICRO*, 2011.

[17] U. Kang et al. Co-architecting controllers and DRAM to enhance DRAM process scaling. In *The Memory Forum*, 2014.

RSP'16, October 06-07 2016, Pittsburgh, PA, USA.
ACM 978-1-4503-4535-4/16/10.
http://dx.doi.org/10.1145/http://dx.doi.org/10.1145/2990299.2990300

[18] S. Khan et al. The efficacy of error mitigation techniques for DRAM retention failures: A comparative experimental study. In *SIGMETRICS*, 2014.

[19] S. Khan et al. PARBOR: An Efficient System-Level Technique to Detect Data-Dependent Failures in DRAM. In *DSN*, 2016.

[20] H. Kim et al. Bounding memory interference delay in COTS-based multi-core systems. In *RTAS*, 2014.

[21] Y. Kim et al. Thread cluster memory scheduling: Exploiting differences in memory access behavior. In *MICRO*, 2010.

[22] Y. Kim et al. A case for subarray-level parallelism (SALP) in DRAM. In *ISCA*, 2012.

[23] Y. Kim et al. Flipping bits in memory without accessing them: An experimental study of DRAM disturbance errors. In *ISCA*, 2014.

[24] E. Kultursay et al. Evaluating STT-RAM as an energy-efficient main memory alternative. In *ISPASS*, 2013.

[25] B. C. Lee et al. Architecting phase change memory as a scalable DRAM alternative. In *ISCA*, 2009.

[26] B. C. Lee et al. Phase change memory architecture and the quest for scalability. *CACM*, 2010.

[27] D. Lee et al. Tiered-latency DRAM: A low latency and low cost DRAM architecture. In *HPCA*, 2013.

[28] D. Lee et al. Adaptive-latency DRAM: Optimizing DRAM timing for the common-case. In *HPCA*, 2015.

[29] J. Liu et al. RAIDR: Retention-aware intelligent DRAM refresh. In *ISCA*, 2012.

[30] J. Liu et al. An experimental study of data retention behavior in modern DRAM devices: Implications for retention time profiling mechanisms. In *ISCA*, 2013.

[31] Y. Lu et al. Loose-ordering consistency for persistent memory. In *ICCD*, 2014.

[32] Y. Luo et al. Characterizing application memory error vulnerability to optimize data center cost via heterogeneous-reliability memory. In *DSN*, 2014.

[33] J. Meza et al. A case for efficient hardware-software cooperative management of storage and memory. In *WEED*, 2013.

[34] J. Meza et al. A Large-Scale Study of Flash Memory Errors in the Field. In *SIGMETRICS*, 2015.

[35] S. Muralidhara et al. Reducing memory interference in multi-core systems via application-aware memory channel partitioning. In *MICRO*, 2011.

[36] O. Mutlu. Memory scaling: A systems architecture perspective. In *IMW*, 2013.

[37] O. Mutlu and T. Moscibroda. Stall-time fair memory access scheduling for chip multiprocessors. In *MICRO*, 2007.

[38] O. Mutlu and L. Subramanian. Research problems and opportunities in memory systems. *SUPERFRI*, 2014.

[39] O. Mutlu et al. Parallelism-aware batch scheduling: Enhancing both performance and fairness of shared DRAM systems. In *ISCA*, 2008.

[40] M. K. Qureshi et al. AVATAR: A Variable-Retention-Time (VRT) Aware Refresh for DRAM Systems. In *DSN*, 2015.

[41] J. Ren et al. ThyNVM: Enabling Software-Transparent Crash Consistency in Persistent Memory Systems. In *MICRO*, 2015.

[42] V. Seshadri et al. RowClone: Fast and efficient In-DRAM copy and initialization of bulk data. In *MICRO*, 2013.

[43] V. Seshadri et al. Gather-Scatter DRAM: In-DRAM Address Translation to Improve the Spatial Locality of Non-unit Strided Accesses. In *MICRO*, 2015.

[44] V. Seshadri et al. Fast Bulk Bitwise AND and OR in DRAM. *CAL*, 2015.

[45] L. Subramanian et al. MISE: Providing performance predictability and improving fairness in shared main memory systems. In *HPCA*, 2013.

[46] L. Subramanian et al. The blacklisting memory scheduler: Achieving high performance and fairness at low cost. In *ICCD*, 2014.

[47] L. Subramanian et al. The application slowdown model: Quantifying and controlling the impact of inter-application interference at shared caches and main memory. In *MICRO*, 2015.

[48] H. Usui et al. DASH: Deadline-Aware High-Performance Memory Scheduler for Heterogeneous Systems with Hardware Accelerators. *TACO*, 2016.

[49] H. Yoon et al. Row buffer locality aware caching policies for hybrid memories. In *ICCD*, 2012.

[50] H. Yoon et al. Efficient data mapping and buffering techniques for multi-level cell phase-change memories. *TACO*, 2014.

[51] J. Zhao et al. FIRM: Fair and high-performance memory control for persistent memory systems. In *MICRO*, 2014.