HPCA 2016 Submission **#109**
Confidential Draft: DO NOT DISTRIBUTE

HW assisted migration for hybrid memories

ABSTRACT

Die-stacked DRAM is an emerging technology that has been annoruced to be included with off-package memories resulting in a hybrid memory system. A large body of recent research has investigated the use of die-stacked DRAM as a hardware-manged last-level cache. This approach comes does not expose die-stacked memory for application use which could be beneifical to memory capacity constrained workloads. An alternative approach is to manage both memories as flat address space as part of main memory. Performance of flat addresss space requires efficient page placement and migrations such that most memory access are from the higher performance die-stacked memory. One apporach is for the operating system (OS) to monitor memory access and periodically migrate pages, however OS is limited to coarser granularities migration periods due to overheads of page table updates, and TLB shootdowns. In this paper we describe a clustered hardware migration architecture that transparently migrates pages that can scale to arbutrary number of channels. We also design scalable solutions to storing page access tracking and page remapping tables that can scale to future systems with terabytes of memory. Our results for evaluation of multi-programmed workloads of parsec CPU and rodinia GPU applications show that our solution has **XX%** better performance over recent flat-address space management schemes.

Keywords: Memory architecture, Die-stacked memory

1. INTRODUCTION

In recent years improvement of system performance has been impeded by the memory wall problem [19]. To alleviate the memory wall problem there has been a lot of recent research in the use of 3-D die-stacked memory to provide high performance. Placing 3D memory stacks in the same package as the processor can provide significant improvements in bandwidth and lower power consumption [1]. There has has been significant advancement in the industry including the development of die-stacked memory standards and consortia [6, 9, 15], and various announcements from several processor companies [5, 14, 1].

Current stacking technology may provide on the order of eight 3D DRAM stacks, each with 2GB capacity, for a total of 16GB of fast DRAM [5]. However, many server systems already support *hundreds* of GB of memory and so a few tens will not suffice for the problem sizes and workloads of interest. The resulting system will therefore consist of two types of memory: a first class of fast, in-package, die-stacked memory, and a second class of off-package commodity memory (e.g., double data rate type 3 (DDR3)).

Given such a *Two-Level Memory* (TLM) hybrid memory organization, the challenge then comes from determining how to best organize and manage this system. The goal of any management is to give the performance of the fast in-package memory while still providing the capacity of the larger off-package memory. A large body of recent body of recent research has focused on utilizing the stacked DRAM as a large, high-bandwidth last-level cache (e.g., an "L4" cache), coping with the challenges of managing the large tag storage required and the relatively slower latencies of DRAM (compared to on-chip SRAM) [10, 16, 7, 8, 20, 18, 13, 3, 4]. Such a hardware caching approach has some immediate advantages, especially that of software-transparency and backwards compatibility. As the stacked DRAM is simply another cache that is transparent to the software layers, any existing applications can be run on a system with such a DRAM cache and potentially obtain performance and/or energy benefits [11]. One drawback of the caching approach is that stacked memory is not available for application use. While the capacities of die-stacked memory is likely to be insufficient to serve as the entirety of a system's main memory it is still non-trivial in size and would be beneficial to memory capacity constrained workloads [2].

An alternative configuration is to expose the die-stacked memory as part of the main memory capacity. In this configuration the management and efficient use of the TLM hybrid memory lies on the application and the underlying management option. One recent research explored management by the operating system (OS) [12]. The OS monitors memory usage and periodically performs page migrations to move frequently accessed pages to stacked memory. One of the many challenges that OS or any runtime implementation faces is that it is constrained by the overheads of management. The OS needs to take an interrupt to do anything, additionally the OS then has to traverse the page tables to read the access frequency information, migrate pages and update the page table entries to reflect the change of virtual to physical address mapping and flush the translation lookaside buffers (TLB). As we show in our evaluations, these costs are non-trivial and constraint the OS to set the minimum management intervals or epochs to 0.1 seconds. Such coarse grained intervals leaves a lot of opportunities on the table.

Most recently [17] proposed a transparent hardware schem to manage TLM as a flat address space. The main contributions of that work was the use of storage efficient structures to track page access frequency and remapping tables. They allow migrations to happen between any aribitrary channels and the use of single centralized migrator can become a bottleneck when systems scale to terabytes of memory and 10s of channels. To this end we in this paper we introduce the migration cluster architecture that is specifically designed to scale to systems of future. Our migration clusters group channels of different memory levels into smaller cluster sets and migration is limited to within cluster only. In this clustered architecture we also use page access counting hardware that can store information for millions of pages as well as we use remap tables that can remap millions of pages in the cluster efficiently. We also describe and evaluate novel migration algorithms that are able to optimize for latency and bandwidth sensistive workloads in a CPU-GPU system.

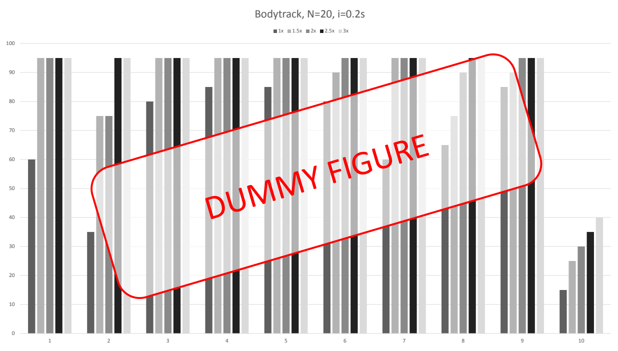

Figure 1: Dummy figure caption 2

Show graph with motivational result.
List the paper's main contributions.
Overview of paper's chapters.

2. BACKGROUND

Description of memory organization
Differences between DDR, HBM, PCM, on-chip/off-chip, benefits and drawbacks of each memory type.
Present the experiment where we identify the optimal NLM organization

3. RELATED WORK

Present the papers in a structured way. Organize them in categories and discuss a category at a time. E.g. HW-assisted, SW-assisted, Hybrid. Separate category for irrelevant papers that worked as motivation.
List all the papers we think are relevant. Dicuss each one briefly.
List the elements that make our work better than the rest.

4. ARCHITECTURE

5. RESULTS

5.1 Experimental Methodology

5.2 Result1

5.3 Result2

5.4 Result3

6. CONCLUSIONS AND FUTURE WORK

7. REFERENCES

[1] Bryan Black, "Die Stacking is Happening," in *Proc. of the Intl. Symp. on Microarchitecture*, Davis, CA, December 2013.

[2] Aamer Jaleel Chiachen Chou and Moinuddin Qureshi, "CAMEO:A Two-Level Memory Organization with Capacity of Main Memory and Flexibility of Hardware-Managed Cache," in *Proc. of the 47th Intl. Symp. on Microarchitecture*, Cambridge, UK, December 2014.

[3] Michel El-Nacouzi, Islam Atta, Myrto Papadopoulou, Jason Zebchuk, Natalie Enright Jerger, and Andreas Moshovos, "A Dual Grain Hit-miss Detector for Large Die-stacked DRAM Caches," in *Proc. of the Conf. on Design, Automation and Test in Europe*, 2013, pp. 89–92.

[4] Fazal Hameed, Lars Bauer, and Jörg Henkel, "Simultaneously Optimizing DRAM Cache Hit Latency and Miss Rate via Novel Set Mapping Policies," in *Proc. of the*, 2013.

[5] Intel, "KnightsLanding," http://www.realworldtech.com/knights-landing-details/.

[6] JEDEC, "Wide I/O Single Data Rate (Wide I/O SDR)," http://www.jedec.org/standards-documents/docs/jesd229.

[7] Djordje Jevdjic, Stavros Volos, and Babak Falsafi, "Die-stacked dram caches for servers," in *Proc. of the Intl. Symp. on Computer Architecture*, 2013.

[8] Xiaowei Jiang, Niti Madan, Li Zhao, M. Upton, R. Iyer, S. Makineni, D. Newell, Y. Solihin, and R. Balasubramonian, "CHOP: Adaptive Filter-Based DRAM Caching for CMP Server Platforms," in *Proc. of the 16th Intl. Symp. on High Performance Computer Architecture*, January 2010, pp. 1–12.

[9] Joint Electron Devices Engineering Council, "JEDEC: 3D-ICs," http://www.jedec.org/category/technology-focus-area/3d-ics-0.

[10] Gabriel H. Loh and Mark D. Hill, "Supporting Very Large Caches with Conventional Block Sizes," in *Proc. of the 44th Intl. Symp. on Microarchitecture*, Porto Alegre, Brazil, December 2011.

[11] Gabriel H. Loh, Nuwan Jayasena, Kevin McGrath, Mike O'Connor, Steven Reinhardt, and Jaewoong Chung, "Challenges in Heterogeneous Die-Stacked and Off-Chip Memory Systems," in *3rd Workshop on SoCs, Heterogeneous Architectures and Workloads (SHAW)*, New Orleans, LA, February 2012.

[12] Mitesh R. Meswani, Sergey Blagodurov, David Roberts, John Slice, Mike Ignatowski, and Gabriel H. Loh, "Heterogeneous memory architectures: A hw/sw approach for mixing die-stacked and off-package memories." in *Proceedings of the International Symposium on High Performance Computer Architecture (HPCA)*, February 2015.

[13] Justin Meza, Jichuan Chang, HanBin Yoon, Onur Mutlu, and Parthasarathy Ranganathan, "Enabling Efficient and Scalable Hybrid Memories Using Fine-Granularity DRAM Cache Management," *Computer Architecture Letters*, vol. 11, no. 2, pp. 61–64, July 2012.

[14] NVIDIA, "NVIDIA Pascal," http://devblogs.nvidia.com/parallelforall/nvlink-pascal-stacked-memory-feeding-appetite-big-data/.

[15] J. Thomas Pawlowski, "Hybrid Memory Cube: Breakthrough DRAM Performance with a Fundamentally Re-Architected DRAM Subsystem," in *Proc. of the 23rd Hot Chips*, Stanford, CA, August 2011.

[16] Moin Qureshi and Gabriel H. Loh, "Fundamental Latency Trade-offs in Architecturing DRAM Caches: Outperforming Impractical SRAM-Tags with a Simple and Practical Design," in *Proc. of the 45th Intl. Symp. on Microarchitecture*, Vancouver, Canada, December 2012.

[17] Jaewoong Sim, Alaa R. Alameldeen, Zeshan Chishti, Chris Wilkerson, and Hyesoon Kim, “Transparent hardware management of stacked dram as part of memory,” in *Proceedings of the 47th Annual IEEE/ACM International Symposium on Microarchitecture*, ser. MICRO-47, 2014.

[18] Jaewoong Sim, Gabriel H. Loh, Hyesoon Kim, Mike O’Connor, and Mithuna Thottethodi, “A Mostly-Clean DRAM Cache for Effective Hit Speculation and Self-Balancing Dispatch,” in *Proc. of the 45th Intl. Symp. on Microarchitecture*, Vancouver, Canada, December 2012.

[19] William A. Wulf and Sally A. McKee, “Hitting the Memory Wall: Implications of the Obvious,” *Computer Architecture News*, vol. 23, no. 1, pp. 20–24, March 1995.

[20] Li Zhao, R. Iyer, R. Illikkal, and D. Newell, “Exploring DRAM cache architectures for CMP server platforms,” in *Proc. of the 25th Intl. Conf. on Computer Design*, October 2007, pp. 55–62.