Exploring Latency-Power Tradeoffs in Deep Nonvolatile Memory Hierarchies

Doe Hyun Yoon
doe-hyun.yoon@hp.com

Tobin Gonzalez
tobin.gonzalez@hp.com

Parthasarathy Ranganathan
partha.ranganathan@hp.com

Robert S. Schreiber
rob.schreiber@hp.com

Intelligent Infrastructure Lab
Hewlett-Packard Labs
Palo Alto, CA, 94304

ABSTRACT

To handle the demand for very large main memory, we are likely to use nonvolatile memory (NVM) as main memory. NVM main memory will have higher latency than DRAM. To cope with this, we advocate a less-deep cache hierarchy based on a large last-level, NVM cache. We develop a model that estimates average memory access time and power of a cache hierarchy. The model is based on captured application behavior, an analytical power and performance model, and circuit-level memory models such as CACTI and NVSim. We use the model to explore the cache hierarchy design space and present latency-power tradeoffs for memory intensive SPEC benchmarks and scientific applications. The results indicate that a flattened hierarchy lowers power and improves average memory access time.

Categories and Subject Descriptors

B.3.3 [**Memory Structures**]: Performance Analysis and Design Aids; B.3.1 [**Memory Structures**]: Semiconductor Memories

General Terms

Design

Keywords

Nonvolatile memory, Memory hierarchy, Latency-power tradeoff

1 Introduction

A modern microprocessor has a private SRAM L1 cache (16-32KB); a private SRAM L2 cache (128-512KB); and a shared last-level cache (LLC) using SRAM or embedded DRAM, as large as 30MB in high-end processors. New technologies such as 3D stacking and byte-addressable nonvolatile memory (NVM) are expected to bring even higher capacity caches. As a result, the cache hierarchy is becoming deeper, with L4 and L5 caches (3D-stacked, on-package, or off-chip DRAM caches, or all of them together).

Power and energy efficiency have become the number one design criterion; hence, designing a memory hierarchy should be an optimization procedure considering multiple objectives including performance and power. The key performance characteristic of cache is average memory access time (AMAT). Two factors dominate AMAT: hit rate, which is mostly determined by the size of LLC, and average latency for a hit. The traditional design strategy for reducing AMAT is a deep cache hierarchy, but this may lead to poor power efficiency. Increasing total capacity improves AMAT only slightly but at the cost of significant increase in power, and a deep hierarchy increases AMAT when the working set is bigger than the intermediate level caches.

NVM main memory changes the cache architecture problem; NVM main memory provides larger capacity but at the cost of higher latency than that of DRAM main memory. For NVM main memory systems, we advocate a 3-level cache hierarchy with an NVM LLC. A flat hierarchy burns less energy than a deeper hierarchy. An NVM LLC has near-zero standby power, and this allows a larger on-chip (or 3D-stacked) cache than a conventional SRAM or DRAM cache, increasing hit rate.

We first develop a latency-power tradeoff model in Section 2. With the model we undertake a co-design study of the optimal depth of a hierarchy in Section 3. Then, we expand the model to support new byte-addressable NVM and show latency-power tradeoffs of various designs. We discuss the limitations of the proposed approach and future work in Section 6 and conclude this paper in Section 7.

2 Latency-Power Tradeoff Model

In this section, we develop a latency-power tradeoff model. The main objective of this study is not to pinpoint a specific optimal configuration but to explore diverse design directions with potential future memory technologies and to identify which direction is most power efficient. The model is fast, so that we can practically explore a large design space.

Figure 1 delineates a high-level organization of the proposed model, which has a performance and power model

CF'12, May 15–17, 2012, Cagliari, Italy.

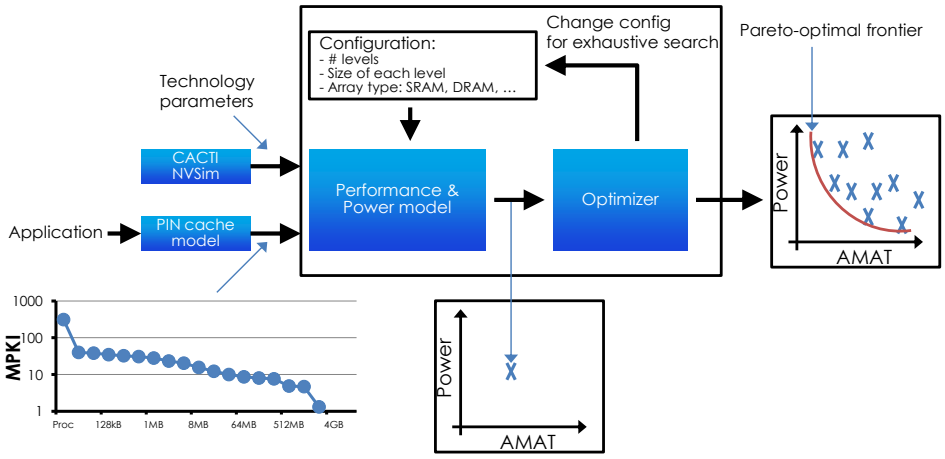


Figure 1: High-level overview of the proposed memory hierarchy analysis framework.

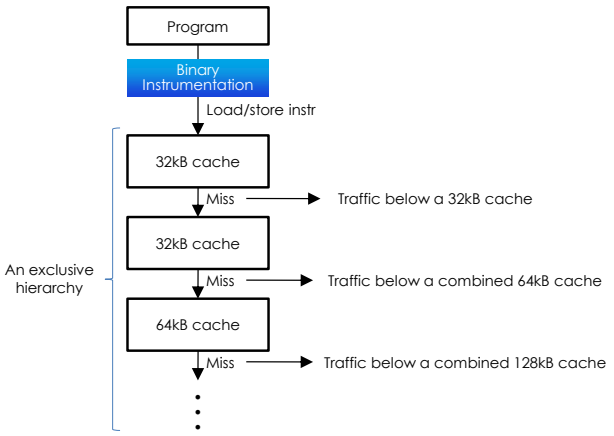


Figure 2: A cache model for PIN to profile MPKI vs. cache sizes.

and an optimizer. The performance and power model takes application characteristics and technology parameters and then estimates AMAT and power (including static and dynamic power) of the given memory hierarchy (the configuration). Each configuration can be represented as a point on a power-AMAT plane. The optimizer exhaustively explores a design space and identifies the Pareto-optimal frontier, presenting the latency-power tradeoff.

In the remainder of this section, we describe the details of the model: Section 2.1 and Section 2.2 discuss the inputs to the model; Section 2.3 describes the performance and power model; and Section 2.4 discusses the optimizer.

2.1 Application Characteristics

We capture application characteristic using a binary instrumentation tool, Pin [12]. We implement a hierarchical cache model as a pintool to profile *MPKI* (misses per thousand instructions) for various cache sizes (from 32KB to $8GB$). Figure 2 illustrates how we profile traffic (MPKI) of each cache size. Each cache is 8-way set associative, and the number of sets is scaled accordingly to increase cache size: 64 sets in a 32kB cache, 128 sets in a 64kB cache, etc. This methodology is very similar to those of Lin *et al.* [10] and Murphy *et al.* [14].

The performance and power model in Section 2.3 uses the MPKI vs. cache size information to estimate AMAT and power. In this study, we use a subset of SPEC CPU 2006 benchmark suite [25] (we use mostly memory-intensive applications but include non-memory-intensive applications also) as well as scientific applications including Mantevo MiniApps [22] and Graph 500 [1]. Table 1 summarizes the applications in this study, and Figure 3 shows the profiled MPKI vs. cache size curves for SPEC CPU 2006 applications, Mantevo MiniApps, and Graph 500.

2.2 Technology Parameters

We leverage CACTI 6 [13] and NVSim [28] to draw technology parameters such as latency, energy per access, and static power. For each cache level, we allow 3 different technologies: SRAM, DRAM, and PCRAM (phase-change memory). We use PCRAM as an example NVM. SRAM and DRAM are modeled in CACTI, and PCRAM in NVSim.

2.3 Performance and Power Model

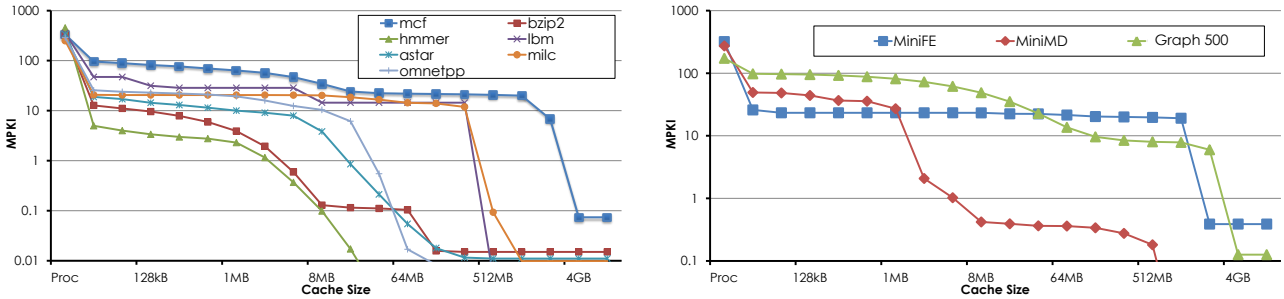
The performance and power model takes the application characteristics and technology parameters and estimates AMAT and power of a given configuration.

Configuration: The configuration (in Figure 1) specifies a cache hierarchy to be evaluated: n is the number of cache levels; $C(i)$ is cache capacity at level i ($1 \leq i \leq n$); and a cache memory at level i can be one of SRAM, DRAM or PCRAM. For a given configuration, we define MPKI, access latency, static power, energy per read, and energy per write at cache level i as $M(i)$, $L(i)$, $P_s(i)$, $E_r(i)$, and $E_w(i)$, respectively, where these values are from the profiled application characteristics (in Section 2.1) and technology parameters (in Section 2.2). In addition, we define $M(0)$, $L(n+1)$, and $E_r(n+1)$ for easier formulation of equations: $M(0)$ is the number of load instructions per thousand instructions, and $L(n+1)$ and $E_r(n+1)$ are latency and energy per read of main memory (e.g., DRAM), respectively.

Performance model: We assume an in-order core and measure performance using AMAT as shown in Equation 1. While an aggressive out-of-order core or simultaneous multithreading can hide memory latency and leverage memory-level parallelism, lower AMAT, in general, indicates better performance as shown in [18].

Table 1: Workloads.

Benchmark Suite	Application	Description	Input
SPEC CPU 2006	mcf	combinatorial optimization	SPEC reference input
	bzip2	compression and decompression	
	hmmer	search a gene sequence database	
	omnetpp	discrete event simulation	
	astar	2D path finding	
	milc	physical and quantum chromodynamics - MIMD lattice computation	
	lbm	computational fluid dynamics, lattice boltzmann model	
Mantevo	MiniFE	Unstructured implicit finite element codes	$160 \times 160 \times 160$
	MiniMD	Force computation in molecular dynamics	$80 \times 80 \times 80$
Graph 500	Graph 500	Breadth first search on a large graph	scale = 22



(a) SPEC CPU 2006 (b) MiniFE, MiniMD, and Graph 500

Figure 3: Traffic (MPKI) vs. cache sizes.

$$AMAT = L(1) + \sum_{i=1}^{n} M(i) \times L(i+1) \quad (1)$$

Power model: We consider power only in caches and ignore power in wires across cache levels and other components in order to keep the model simple. Total power in a cache hierarchy, P_{total}, is a sum of static and dynamic power. Estimating static power consumption is straightforward as shown in Equation 2.

$$P_{static} = \sum_{i=1}^{n} P_s(i) \quad (2)$$

Calculating dynamic power is a little bit tricky since we need to translate dynamic energy consumption into dynamic power. We first calculate total dynamic energy consumption for 1000 instructions as shown in Equation 3.

$$E_{dynamic} = M(0) \times E(1) + \sum_{i=1}^{n} M(i) \times (E_r(i+1) + E_w(i)) \quad (3)$$

Then, we get total dynamic power in Equation 4. The denominator in Equation 4 is an estimated time to execute 1000 instructions, and we assume an in-order core with cycle time of T_{cyc} (1ns in our study).

$$P_{dynamic} = \frac{E_{dynamic}}{(1000 - M(0)) \times T_{cyc} + AMAT \times M(0)} \quad (4)$$

2.4 Exploring the Design Space and Pareto-Optimal Frontier

As shown in Figure 1, the result of the performance and power model can be represented as a point on a power-AMAT plane. The optimizer runs the performance and power model iteratively, varying the configuration, and plots the operating points of all the possible configurations on a power-AMAT plane. Then it identifies the Pareto-optimal frontier, showing latency-power tradeoff.

We change the number of levels between 2 and 6. We evaluate SRAM caches up to 2GB in Section 3 to show negative impacts of a large SRAM cache. In Section 4, SRAM cache size is limited to 32MB, DRAM cache size is between 4MB and 64MB, and PCRAM cache size is between 16MB and 1GB. We apply these constraints to avoid unreasonably large SRAM caches or tiny PCRAM / DRAM caches.

Figure 4 shows an example running with the Graph 500 application. Red dots are Pareto-optimal, and blue dots are not. Among those optimal operating points, we annotate minimum latency, minimum power, and power-efficient configurations. All those configurations (red dots) are Pareto-optimal, the minimum latency configuration uses too much power, the minimum power configuration has poor performance, and the power-efficient configurations are balanced in both latency and power. From this, we argue for designing a power-efficient cache hierarchy rather than minimizing latency; designing a cache hierarchy to further minimize latency beyond the power-efficient point causes skyrocketing cache power.

3 Depth of a Cache Hierarchy

In this section, we use the latency-power model to study optimal depth of a cache hierarchy. We compare Pareto-

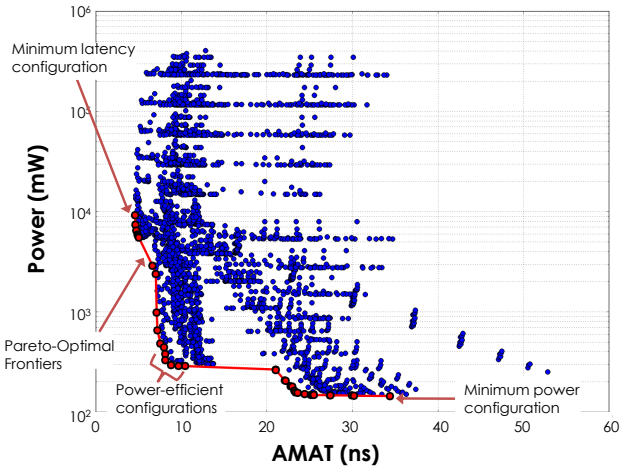


Figure 4: An example of exhaustive search and Pareto-optimal frontier.

optimal frontiers of cache hierarchies with different number of levels, from simple 2-level hierarchies to 6-level hierarchies.

Figure 5 shows latency-power tradeoffs of 2- to 6-level cache hierarchies for the Graph 500 application; As shown in Figure 5(a), the difference among the 3- to 6-level hierarchies is not critical. We evaluated more than 6 levels, but they were even worse, and we do not show them in Figure 5.

The rectangle in Figure 5(a) is magnified in Figure 5(b). The 2-level hierarchies achieve lower AMAT at low power, the 3-level hierarchies are better when higher power is allowed, and a hierarchy with 4 or more levels is worse than the a 3-level hierarchy.

Figure 6 shows the same latency-power tradeoffs for the MiniFE application. MiniFE has a flat traffic curve as shown in Figure 3(b) (streaming access pattern). This is due to sparse matrix vector multiplication (SpMV) in each iteration of the conjugate gradient solver in MiniFE. Hence, AMAT stays at around 7.5ns unless the largest cache is 2GB (top left operating points), which can hold the entire sparse matrix (around 1.5GB). Even a very large cache (e.g,. 1GB) cannot reduce AMAT effectively (top right operating points) but incurs orders of magnitude higher power.

We analyze other applications, and they show the same trend. Increasing the depth of a cache hierarchy beyond 3 levels does not reduce latency and often use more power. In general, a 3-level hierarchy balances latency and power. Large SRAM caches are not power efficient due to an order of magnitude higher power.

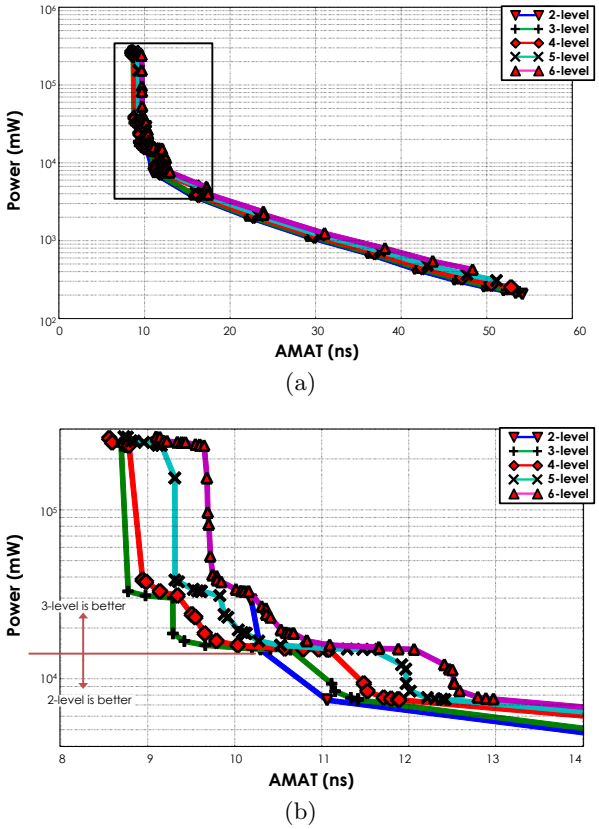


Figure 5: Graph 500 – power-performance tradeoffs of 2- to 6-level hierarchies. The rectangle in (a) is shown in (b).

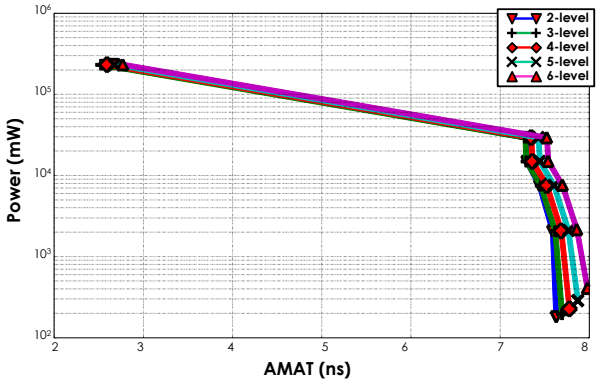


Figure 6: MiniFE – power-performance tradeoffs of 2- to 6-level hierarchies.

4 Cache Hierarchy for NVM Main Memory

Researchers have recently proposed new byte-addressable NVM such as PCRAM as a scalable substitute for DRAM as main memory [8, 21] The main advantage with NVM main memory is higher density and lower standby power than DRAM.

Previously, researchers also suggested a DRAM cache to compensate for high access time of NVM main memory [21, 11]. We analyze the latency-power tradeoffs of cache hierarchies with heterogeneous technologies: SRAM, DRAM and PCRAM. We show that a large NVM LLC is power efficient. A large NVM LLC reduces costly off-chip traffic and does not increase power by much. An advanced cache management scheme can further improve the efficiency of NVM LLC for streaming applications.

4.1 Cache Hierarchy with Heterogeneous Technologies

Figure 7 shows latency-power tradeoffs of cache hierarchies with heterogeneous technologies for the Graph 500 application. *SRAM cache + DRAM main memory* denotes the

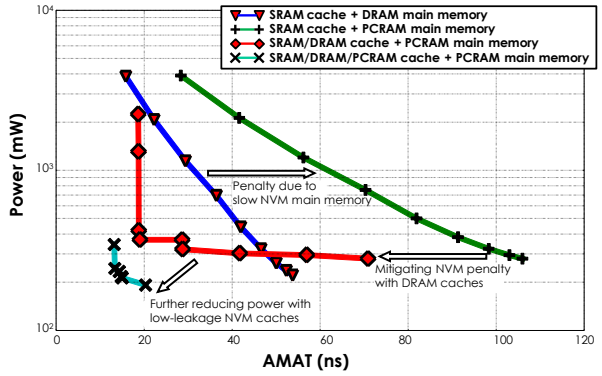


Figure 7: Effects of disparate technologies in main memory and a cache hierarchy (the Graph 500 application).

traditional SRAM-only cache hierarchies with DRAM main memory. When we replace DRAM main memory with PCRAM, the Pareto-optimal frontier moves away due to high access time of PCRAM (*SRAM cache + PCRAM main memory*). One of the advantages with PCRAM main memory is higher memory capacity than DRAM, reducing page fault rates, but our analysis does not show this benefit with PCRAM.

As prior work on a DRAM cache [21, 11], we use a DRAM cache to compensate for high PCRAM access time. We assume a 3D-stacked DRAM cache that leverages high TSV (through silicon via) bandwidth [27]. We present a latency-power tradeoff with mixed SRAM and DRAM caches (*SRAM/DRAM cache + PCRAM main memory*). We let the optimizer choose memory technology of each cache level, and the Pareto-optimal designs use SRAM for L1 and DRAM for L2 and L3. A DRAM cache uses lower power than an SRAM cache, so the SRAM/DRAM heterogeneous hierarchies allow a larger DRAM cache than SRAM, which reduces slow off-chip accesses.

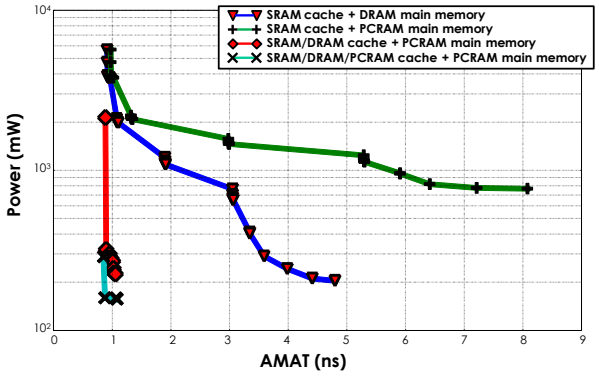
We finally show a latency-power tradeoff with mixed SRAM, DRAM, and PCRAM caches (*SRAM/DRAM/PCRAM cache + PCRAM main memory*). PCRAM has very low leakage power; hence, it allows even larger 3D-stacked caches (e.g., 1GB) still at low power unlike SRAM and DRAM caches. Note that the best AMAT achieved with the mixed SRAM/DRAM/PCRAM cache hierarchies is only comparable to that of SRAM-only hierarchies. In fact, SRAM-only hierarchies can achieve even better performance if we allow larger than 32MB SRAM caches, but that incurs orders of magnitude higher power.

4.2 Cache Friendly Applications

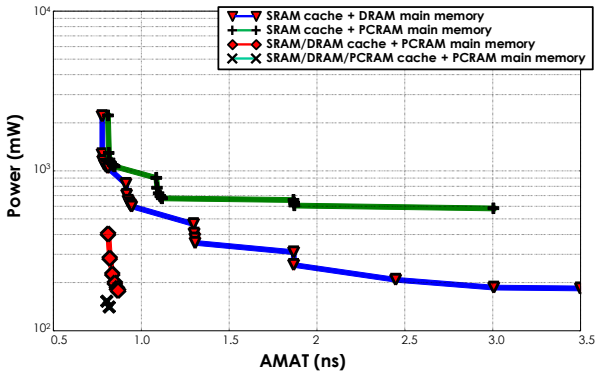
Figure 8 shows the latency-power tradeoffs for cache friendly applications: astar and bzip2. Other cache friendly applications such as hmmer and MiniMD show similar behavior, and we do not present them here. Similar to Graph 500, a large DRAM cache makes up for the penalty with PCRAM main memory, and a PCRAM cache achieves superb power-efficiency.

4.3 Memory Intensive Applications

Figure 9 depicts the same analysis for memory intensive applications: mcf, milc, and lbm. These applications have MPKI bigger than 10 with a 1MB SRAM cache. Unlike the cache friendly applications, the SRAM/DRAM cache hierarchies cannot fully make up for the increased LLC miss



(a) astar



(b) bzip2

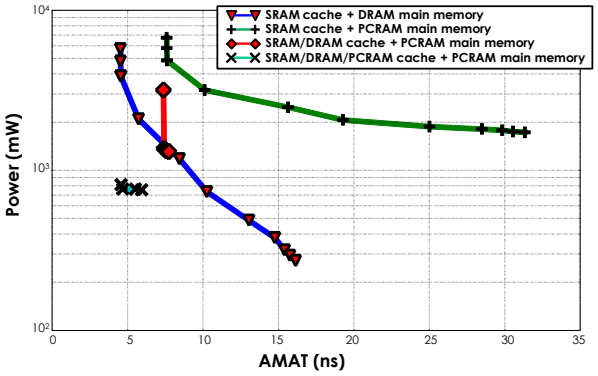
Figure 8: Power-performance tradeoffs with disparate technologies for astar and bzip2

penalty with PCRAM main memory, leading to less power efficiency than SRAM-only hierarchies with DRAM main memory. A PCRAM cache is effective in theses applications also. In lbm and milc, a 1GB PCRAM cache completely holds the whole working set, and in mcf, a 1GB PCRAM cache cuts off-chip traffic by half compared to the largest SRAM cache (32MB).

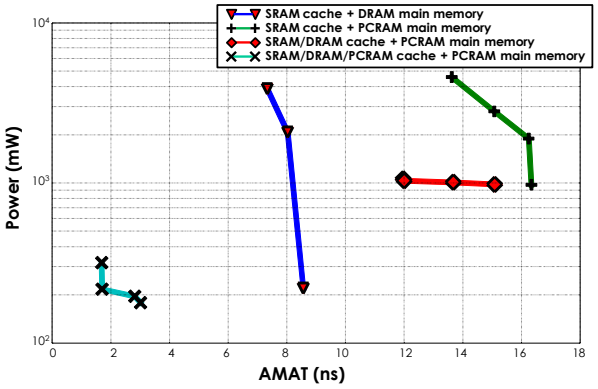
4.4 Streaming Access Patterns

A cache is not effective for streaming access patterns; a program scans a data array, which is much bigger than the largest cache. As discussed, SpMV in MiniFE has such an access pattern. The sparse matrix size is 1.5GB, and even the largest PCRAM cache, only 1GB in our study, cannot hold the whole working set. This results in the latency-power tradeoff in Figure 10(a); DRAM and PCRAM caches are worse than SRAM-only caches.

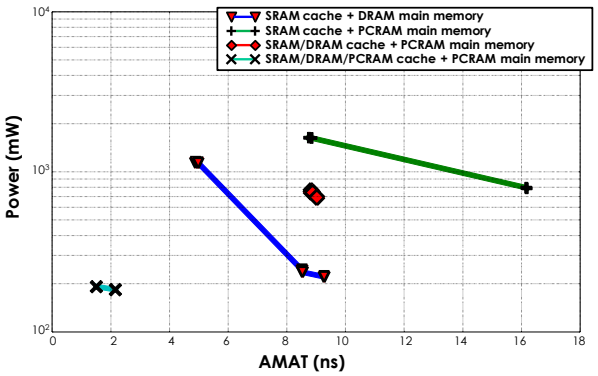
Recent research on cache management policies such as dynamic insertion policy (DIP) [19] and re-reference interval prediction (RRIP) [4] can preserve a fraction of working set in the cache for streaming access patterns. To incorporate such advanced cache designs in our model, we use an alternative replacement policy (instead of LRU) in the hierarchical cache model (Section 2.1). The alternative policy inserts a cache line only when there are empty slots. Once all the 8 ways of a set are filled with valid cache lines, those 8 lines are never replaced. This, relatively simple, policy does not work for most cases but lets a fraction of the sparse matrix stay in the cache hierarchy even if the whole sparse matrix



(a) mcf

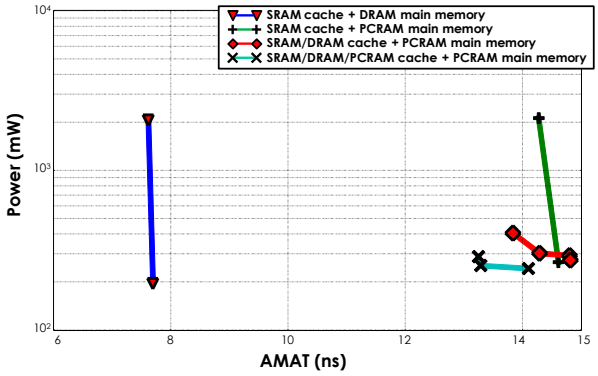


(b) milc

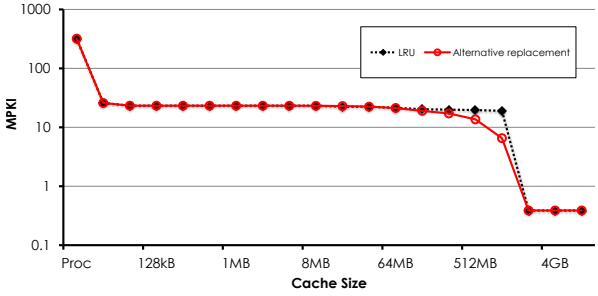


(c) lbm

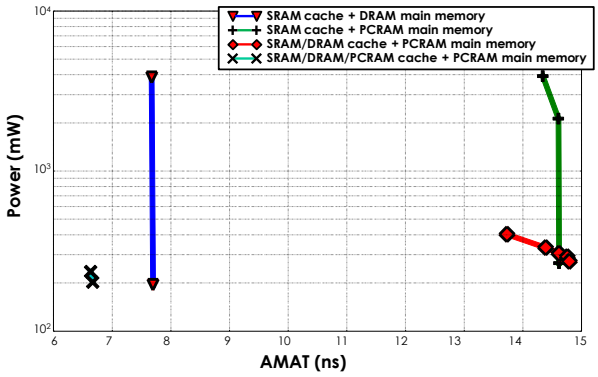
Figure 9: Power-performance tradeoffs with disparate technologies for mcf, milc, and MiniFE.



(a) MiniFE



(b) MiniFE traffic with different replacement policies



(c) MiniFE with an alternative replacement policy

Figure 10: MiniFE with an alternative replacement policy.

is bigger than caches, mimicking DIP and RRIP. We apply the alternative policy only to caches larger than 32MB.

Figure 10(b) compares MPKI vs. cache sizes with a normal LRU replacement policy and the alternative policy. With LRU, traffic does not reduce unless cache size is bigger than the working set (1.5GB), whereas the alternative policy reduces traffic with caches ranging 128MB to 1GB.

We apply this traffic curve to the latency-power model (Figure 10(c)). A 1GB PCRAM cache effectively suppresses off-chip traffic; hence, the SRAM/DRAM/PCRAM cache hierarchies with PCRAM main memory achieves the most power efficient operating points and outperforms the SRAM-only hierarchies with DRAM main memory.

5 Related Work

Our work builds on an extensive amount of prior work. This includes research on analytical memory models and recent research on NVM-based systems.

Memory models: Multilevel cache hierarchy models [15, 6] are focused on average memory access time but do not consider power efficiency. They argue for two-level hierarchies as opposed to single-level cache. Our latency-power model is based on these models but is extended to support arbitrary number of cache levels and power estimation.

Jacob *et al.* developed a closed form solution for optimal size of each cache level and suggested that the cheapest memory level be increased in the first place [3]. This obser-

vation is in line with our analysis that incorporating a large PCRAM (3D-stacked) cache is effective.

Moguls [26] is another memory model that considers bandwidth and power (only dynamic power) in designing a memory hierarchy. Moguls uses the application-oblivious $\sqrt{2}$ model and do not analyze application-dependent behavior in detail.

New byte-addressable NVM: Nonvolatile memories such as PCRAM have been suggested as a scalable substitute for DRAM in many papers [8, 21, 17] since DRAM scaling is approaching its limit. Qureshi *et al.* proposed a DRAM cache to compensate for slow PCRAM main memory [21] and combining SLC and MLC PCRAM [17]. Studies on NVM main memory are focused on performance and reliability. Our work optimizes cache hierarchies (as opposed to optimizing a cache level), considering both performance and power.

A PCRAM-based cache is suggested by Joo *et al.* [5], focusing on PCRAM's finite write endurance. Dong *et al.* proposed 3D-stacked magnetic memory (MRAM) caches, claiming better power efficiency [2]. The approach is not thorough design space exploration including SRAM, DRAM, and NVRAM as in our work.

6 Limitations and Future Work

We developed a latency-power model that explores the design space exhaustively, but the model has limitations. We assumed an in-order core for performance and power estimation, but modern processors have latency-hiding techniques: out-of-order processing, non-blocking caches, simultaneous multi-threading, prefetching, etc. For a system that can hide memory latency, AMAT is not an adequate metric for performance. We leave developing a better performance model to future work.

We propose a PCRAM cache to increase cache capacity without increasing static power. PCRAM (and other NVM technologies) suffer from write endurance; a memory cell has limited lifetime. Most authors assume endurance of 10^8 write cycles, but recently developed fully-confined PCRAM cells have much longer lifetime (more than 10^{11} write cycles) [7]. We can also combine wear-leveling [20], failure tolerance [23, 29, 24, 16] and write buffers (SRAM or DRAM) to cope with write endurance.

Our estimate of poor power efficiency in SRAM-only hierarchies is based on CACTI 6. A recent study on circuit-level cache modeling revealed that SRAM leakage power can be reduced with power control mechanisms [9]. We believe that the big picture we present in this paper will not change significantly even with aggressive power control techniques. We leave incorporating various power reduction/control techniques to future work.

7 Conclusions

We develop a latency-power model of memory hierarchies; the proposed model reports both AMAT and power and is yet simple enough to enable exhaustive search for identifying latency-power tradeoffs.

We first use the model to compare cache hierarchies with different depths. Our analysis shows that deep hierarchies are less power efficient than flat hierarchies (2 or 3 levels) and that large SRAM caches demand a large amount of static power.

Then, we embrace new byte-addressable NVM technologies in our model. We corroborate prior work; NVM main memory degrades performance due to high NVM latency, and DRAM caches can make up for this penalty. But, large DRAM caches (similar to large SRAM caches) draw a large amount of static power, leading to poor power efficiency. We suggest a 3D-stacked NVM cache; an NVM cache can be even larger than SRAM or DRAM caches, while leakage power is much less. We also discuss streaming access patterns; combining NVM caches and advanced cache management schemes can potentially reduce costly off-chip traffic for streaming access patterns.

While there are several opportunities for refinement in the proposed approach, we believe the co-design by model-driven design space exploration is important and will lead to a more optimized system design.

8 Acknowledgment

This material is based upon work supported by the Department of Energy under Award Number DE - SC0005026.

9 Disclaimer

10 References

[1] Graph 500. `http://www.graph500.org/`.

[2] X. Dong, X. Wu, Y. Xie, Y. Chen, and H. Li. Stacking MRAM atop microprocessors: An architecture-level evaluation. *IET Computers & Digital Techniques*, 5(3), 2011.

[3] B. L. Jacob, P. M. Chen, S. R. Silverman, and T. N. Mudge. An analytical model for designing memory hierarchies. *IEEE Transactions on Computers*, 45:1180–1194, Oct. 1996.

[4] A. Jaleel, K. Theobald, S. C. Steely Jr., and J. Emer. High performance cache replacement using re-reference interval prediction (RRIP). In *Proc. the 37th Ann. Int'l Symp. Computer Architecture (ISCA)*, Jun. 2010.

[5] Y. Joo, D. Niu, X. Dong, G. Sun, N. Chang, and Y. Xie. Energy- and endurance-aware design of phase change memory caches. In *Proc. the Conf. Design Automation and Test in Europe (DATE)*, Mar. 2010.

[6] N. P. Jouppi and S. J. E. Wilton. Tradeoffs in two-level on-chip caching. In *Proc. the 21st Ann. Int'l Symp. Computer Architecture (ISCA)*, Apr. 1994.

[7] I. S. Kim, S. L. Cho, D. H. Im, E. H. Cho, D. H. Kim, G. H. Oh, D. H. Ahn, S. O. Park, S. W. Nam, J. T. Moon, and C. H. Chung. High performance PRAM cell scalable to sub-20nm technology with below $4F^2$ cell size, extendable to DRAM applications. In *Proc. the Symp. VLSI Technology (VLSIT)*, Jun.

[8] B. C. Lee, E. Ipek, O. Mutlu, and D. Burger. Architecting phase change memory as a scalable DRAM alternative. In *Proc. the 36th Ann. Int'l Symp. Computer Architecture (ISCA)*, Jun. 2009.

[9] S. Li, K. Chen, J. H. Ahn, J. B. Brockman, and N. P. Jouppi. CACTI-P: Architecture-level modeling for SRAM-based structures with advanced leakage reduction techniques. In *Proc.*

the IEEE/ACM Int'l Conf. Computer-Aided Design (ICCAD), Nov. 2011.

[10] J. M. Lin, Y. Chen, W. Li, Z. Tang, and A. Jaleel. Memory characterization of SPEC CPU 2006 benchmark suite. In *Proc. the 11th Workshop for Computer Architecture Evaluation of Commercial Workloads (CAECW)*, Feb. 2008.

[11] G. H. Loh and M. D. Hill. Efficiently enabling conventional block sizes for very large die-stacked DRAM caches. In *Proc. the 44th Ann. IEEE/ACM Int'l Symp. Microarchitecture (MICRO)*, Dec. 2011.

[12] C.-K. Luk, R. Cohn, R. Muth, H. Patil, A. Klauser, G. Lowney, S. Wallace, V. J. Reddi, and K. Hazelwood. PIN: Building customized program analysis tools with dynamic instrumentation. In *Proc. the ACM Conf. Programming Language Design and Implementation (PLDI)*, Jun. 2005.

[13] N. Muralimanohar, R. Balasubramonian, and N. P. Jouppi. CACTI 6.0. Technical report, HP Labs., Apr. 2009.

[14] R. C. Murphy, A. Rodrigues, P. Kogge, and K. Underwood. The implications of working set analysis on supercomputing memory hierarchy design. In *Proc. the International Conference on Supercomputing (ICS)*, Jun. 2006.

[15] S. Przybylski, M. Horowitz, and J. Hennessy. Characteristics of performance-optimal multi-level cache hierarchies. In *Proc. the 16th Ann. Int'l Symp. Computer Architecture (ISCA)*, Jun. 1989.

[16] M. K. Qureshi. Pay-As-You-Go: Low overhead hard-error correction for phase change memories. In *Proc. the 44th IEEE/ACM Int'l Symp. Microarchitecture (MICRO)*, Dec. 2011.

[17] M. K. Qureshi, M. Fraceschini, L. Lastras, and J. Karidis. Morphable memory system: A robust architecture for exploiting multi-level phase change memories. In *Proc. the 37th Ann. Int'l Symp. Computer Architecture (ISCA)*, Jun. 2010.

[18] M. K. Qureshi, M. Franceschini, and L. Lastras. Improving read performance of phase change memories via write cancellation and write pausing. In *Proc. the 16th Int'l Symp. High-Performance Computer Architecture (HPCA)*, Jan. 2010.

[19] M. K. Qureshi, A. Jaleel, Y. N. Patt, S. C. Steely Jr., and J. Emer. Adaptive insertion policies for high-performance caching. In *Proc. the 34th Ann. Int'l Symp. Computer Architecture (ISCA)*, Jun. 2007.

[20] M. K. Qureshi, J. Karidis, M. Franceschini, V. Srinivasan, L. Lastras, and B. Abail. Enhancing lifetime and security of PCM-based main memory with start-gap wear leveling. In *Proc. the 42nd IEEE/ACM Int'l Symp. Microarchitecture (MICRO)*, Dec. 2009.

[21] M. K. Qureshi, V. Srinivasan, and J. A. Rivers. Scalable high-performance main memory system using phase-change memory technology. In *Proc. the 36th Ann. Int'l Symp. Computer Architecture (ISCA)*, Jun. 2009.

[22] Sandia National Laboratories. Mantevo project. `https://software.sandia.gov/mantevo/index.html`.

[23] S. Schechter, G. H. Loh, K. Strauss, and D. Burger. Use ECP, not ECC, for hard failures in resistive memories. In *Proc. the 37th Ann. Int'l Symp. Computer Architecture (ISCA)*, Jun. 2010.

[24] N. H. Seong, D. H. Woo, V. Srinivasan, J. A. Rivers, and H.-H. S. Lee. SAFER: Stuck-at-fault error recovery for memories. In *Proc. the 43rd IEEE/ACM Int'l Symp. Microarchitecture (MICRO)*, Dec. 2010.

[25] Standard Performance Evaluation Corporation. SPEC CPU 2006. `http://www.spec.org/cpu2006/`, 2006.

[26] G. Sun, C. J. Hughes, C. Kim, J. Zhao, C. Xu, Y. Xie, and Y.-K. Chen. Moguls: A model to explore the memory hierarchy for bandwidth improvements. In *Proc. the 38th Ann. Int'l Symp. Computer Architecture (ISCA)*, Jun. 2011.

[27] D. H. Woo, N. H. Seong, D. L. Lewis, and H.-H. S. Lee. An optimized 3D-stacked memory architecture by exploiting excessive, high-density TSV bandwidth. In *Proc. the 16th Int'l Symp. High-Performance Computer Architecture (HPCA)*, Jan. 2010.

[28] C. Xu, X. Dong, N. P. Jouppi, and Y. Xie. Design implications of memristor-based RRAM cross-point structures. In *Proc. the Design, Automation and Test in Europe (DATE)*, Mar. 2011.

[29] D. H. Yoon, N. Muralimanohar, J. Chang, P. Ranganathan, N. P. Jouppi, and M. Erez. FREE-p: Protecting non-volatile memory against both hard and soft errors. In *Proc. the 17th Int'l Symp. High-Performance Computer Architecture (HPCA)*, Feb. 2011.