Hybrid Cache Architecture with Disparate Memory Technologies *

Xiaoxia Wu† Jian Li‡ Lixin Zhang‡ Evan Speight‡ Ram Rajamony‡ Yuan Xie†
†Department of Computer Science and Engineering
The Pennsylvania State University, University Park, PA 16802
‡IBM Austin Research Laboratory, Austin, TX 78758
†{xwu,yuanxie}@cse.psu.edu ‡{jianli,zhangl,speight,rajamony}@us.ibm.com

ABSTRACT

Caching techniques have been an efficient mechanism for mitigating the effects of the processor-memory speed gap. Traditional multi-level SRAM-based cache hierarchies, especially in the context of chip multiprocessors (CMPs), present many challenges in area requirements, core–to–cache balance, power consumption, and design complexity. New advancements in technology enable caches to be built from other technologies, such as Embedded DRAM (EDRAM), Magnetic RAM (MRAM), and Phase-change RAM (PRAM), in both 2D chips or 3D stacked chips. Caches fabricated in these technologies offer dramatically different power and performance characteristics when compared with SRAM-based caches, particularly in the areas of access latency, cell density, and overall power consumption. In this paper, we propose to take advantage of the best characteristics that each technology offers, through the use of Hybrid Cache Architecture (HCA) designs.

We discuss and evaluate two types of hybrid cache architectures: inter cache Level HCA (LHCA), in which the levels in a cache hierarchy can be made of disparate memory technologies; and intra cache level or cache Region based HCA (RHCA), where a single level of cache can be partitioned into multiple regions, each of a different memory technology. We have studied a number of different HCA architectures and explored the potential of hardware support for intra-cache data movement and power consumption management within HCA caches. Utilizing a full-system simulator that has been validated against real hardware, we demonstrate that an LHCA design can provide a geometric mean 7% IPC improvement over a baseline 3-level SRAM cache design under the same area constraint across a collection of 25 workloads. A more aggressive RHCA-based design provides 12% IPC improvement over the baseline. Finally, a 2-layer 3D cache stack (3DHCA) of high density memory technology within the same chip footprint gives 18% IPC improvement over the baseline. Furthermore, up to 70% reduction in power consumption over a baseline SRAM-only design is achieved.

Categories and Subject Descriptors

B.3.2 [**Hardware**]: Memory Structures—*cache memories*

General Terms

Design, Experimentation, Performance

Keywords

Hybrid Cache Architecture, Three-dimensional IC

1. INTRODUCTION

The use of caches in computing systems has been a widely-adopted method for addressing long memory access latency, which has been exacerbated by rapid technology scaling. Cache subsystems, particularly on-chip, with multiple layers of large caches have become common in modern processors such as AMD Barcelona®, IBM POWER6® and Intel Nehalem®. However, the advent of Chip Multiprocessors (CMPs) has increased the pressure on achieving good cache performance. Increased amounts of conventional Static Random Access Memory (SRAM) capacity or the introduction of additional cache levels into the hierarchy are constrained by such factors as chip real estate, the balance between cores and caches, and power consumption. Furthermore, more cache levels may also introduce extra performance and design overheads, including the need for efficient, scalable cache coherence protocols; longer cache miss latency; and addressing the presence of multiple copies of a cache line in multiple levels of the cache hierarchy.

One can improve the performance of large caches through Non-Uniform Cache Architecture (NUCA) [17]. NUCA provides a way to manage large on-chip caches where the latency has traditionally been affected by increasing wire delays. In NUCA, a large cache is divided into multiple banks with different access latencies determined by their physical locations to the source of the request. Two main types of NUCA, Static NUCA (SNUCA) and Dynamic NUCA (DNUCA), have been proposed. In SNUCA, a cache line is statically mapped into banks, with the low-order bits of the index determining the bank. In DNUCA, any given line can be mapped to several banks based on the mapping policy. The average latency is reduced by putting frequently

*This work was performed while Xiaoxia Wu was an intern at IBM Austin Research Laboratory.

ISCA'09, June 20–24, 2009, Austin, Texas, USA.

accessed data into banks that are physically closer to the source of the request.

Table 1: Comparison of different memory technologies.

Features	SRAM	eDRAM	MRAM	PRAM
Density	Low	High	High	Very high
Speed	Very Fast	Fast	Fast read Slow write	Slow read Very slow write
Dyn.Power	Low	Medium	Low read; High write	Medium read High write
Leak. Power	High	Medium	Low	Low
Non-volatile	No	No	Yes	Yes
Scalability	Yes	Yes	Yes	Yes

Traditional NUCA only utilizes the varied access latency of cache banks, due to their physical locations, to improve performance. The cache banks are typically of the same size, process, and circuit technology. The overall cache size budget is fixed for the same memory technology (e.g., SRAM). Here, we note that different memory technologies may have significantly different properties: density, read/write latency, dynamic/static power consumption, reliability features, scalability, etc. Table 1 lists important qualitative features of several memory technologies: SRAM, embedded Dynamic RAM (eDRAM), Magnetic RAM (MRAM) [14], and Phase-change RAM (PRAM) [8, 13]. We focus on the power and performance features of these memory technologies in this work. Several observations relevant to this study may be made from Table 1: (1) PRAM has the highest potential density, but it also has the slowest speed. MRAM and eDRAM also have higher density than SRAM, but both are slower than SRAM. Depending on the design, MRAM read speed may be comparable to that of SRAM. (2) MRAM and PRAM have very different read and write features in terms of latency and power consumption, with particularly high write power consumption. (3) SRAM has high static[1] power, while MRAM and PRAM have very low static power due to their non-volatile property. (4) eDRAM has a good overall balance of dynamic and static power.

Consequently, a properly designed cache that is made of differing memory technologies may have the potential to outperform its counterpart of single technology. For example, IBM POWER® processors have off-chip L3 caches made of eDRAM technology. In addition, even though mixed technologies can also be integrated on the same two-dimensional (2D) chip, the emerging three-dimensional (3D) chip integration technologies may provide further design and manufacture cost benefits for on-chip mixed-technology integration [11].

In this paper, we propose and evaluate several Hybrid Cache Architecture (HCA) options to accommodate on-chip cache hierarchies. To fully take advantage of the benefits from varied memory technologies, an HCA allows levels in a cache hierarchy to be constructed from different memory technologies. Alternately, one level of cache can be partitioned into multiple regions of different memory technologies. In addition, we propose techniques such as low-overhead intra-cache data movement and power-aware policies to improve both cache performance and power in an HCA system. Using a hardware calibrated full-system simulator on a suite of 25 workloads, we show that an inter-cache-level LHCA design can provide a geometric mean 7% IPC improvement over a baseline 3-level SRAM cache design under the same area constraint across a collection of 25 workloads. A more aggressive RHCA-based design provides 12% IPC improvement over the baseline. Finally, a 2-layer 3D cache stack (3DHCA) of high density memory technology within the same chip footprint gives 18% IPC improvement over the baseline. All configurations show substantial power reductions.

[1]Static power consumption is mainly due to leakage current. We interchange the notions of static power and leakage power in this work.

This paper makes the following contributions:

- We present a general approach to constructing on-chip cache hierarchies with differing memory technologies. Such a *hybrid cache* may be either inter-cache-level (LHCA) or region-based (RHCA), intra-cache-level. We have studied hybrid caches made of combinations of SRAM, eDRAM, MRAM, and PRAM under the same area constraint.
- In the RHCA design with fast and slow regions, we propose a hierarchical NUCA cache with centralized swap buffer, parallel address search, and LRU replacement across cache regions. A cache region itself can be a conventional NUCA of identical cache tiles that differ only by distance.
- We propose improvements for intra-cache swap operations. We also conduct detailed sensitivity studies for efficient swap operations.
- We propose and evaluate a drowsy hybrid cache technique with significant cache power savings.
- We extend the hybrid cache technique to 3D stacking and evaluate the benefits.

2. BACKGROUND

This section provides background information on two emerging memory technologies (MRAM and PRAM) and introduces 3D integration.

2.1 Magnetic RAM (MRAM)

The basic difference between MRAM and conventional RAM technologies (such as SRAM/DRAM) is that the information carrier of MRAM is a Magnetic Tunnel Junction (MTJ) instead of electric charges [14, 30]. Each MTJ contains *two ferromagnetic layers* and *one tunnel barrier layer*. Figure 1 shows a conceptual illustration of an MTJ. One of the ferromagnetic layers (the reference layer) has a fixed magnetic direction while the other one (the free layer) can change its magnetic direction via an external electromagnetic field or a spin-transfer torque. If the two ferromagnetic layers have different directions, the MTJ resistance is high, indicating a "1" state (the anti-parallel case in Figure 1(a)); if the two layers have the same direction, the MTJ resistance is low, indicating a "0" state (the parallel case in Figure 1(b)).

The MRAM technology to be discussed in this paper is called *Spin-Transfer Torque RAM* (STT-RAM), which is a new generation of MRAM technologies. STT-RAMs change the magnetic direction of the free layer by directly passing a spin-polarized current through the MTJ structure. Compared to the previous generation of MRAMs that used external magnetic fields to reverse the MTJ status, STT-RAMs have the advantage of scalability, as the *threshold current* to make the status reversal will decrease as the size of the MTJ becomes smaller.

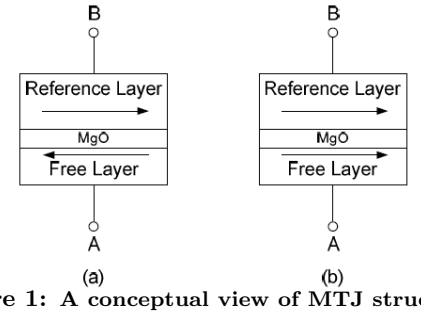

Figure 1: A conceptual view of MTJ structure.

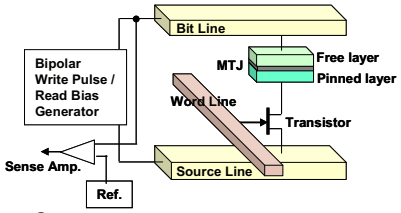

Figure 2: An illustration of an MRAM cell.

In the STT-RAM memory cell design, the most popular structure is composed of one NMOS transistor as the access controller and one MTJ as the storage element ("1T1J" structure) [14]. As illustrated in Figure 2, the storage element, MTJ, is connected in series with the NMOS transistor. The NMOS transistor is controlled by the word-line (WL) signal. The detailed read and write operations for each MRAM cell is described as follows:

Write Operation: When a *write operation* is performed, a positive voltage difference is established between the source-line (SL) and bit-line (BL) for writing for a "0" or a negative voltage difference is established for writing a "1". The current amplitude required to ensure a successful status reversal is called the threshold current. This current is related to the material of the tunnel barrier layer, the writing pulse duration, and the MTJ geometry.

Read Operation: When a *read operation* is desired, the NMOS is turned enabled and a voltage ($V_{BL} - V_{SL}$) is applied between the BL and the SL. This voltage is negative and is usually very small (- 0.1V as demonstrated in [14]). The voltage difference will cause a current to pass through the MTJ, but it is small enough to not invoke a disturbed write operation. The value of the current is determined by the equivalent resistance of MTJs. A sense amplifier compares this current with a reference current and then decides whether a "0" or a "1" is read from the selected MRAM cell.

2.2 Phase-Change RAM (PRAM)

PRAM, a.k.a. phase-change memory (PCM), is a another promising memory technology [8, 13].It has a wide resistance range, which is about three orders of magnitude; therefore, multi-level PRAM allows the storage of multiple bits per cell. The basic structure of a PRAM cell consists of a standard NMOS access transistor and a small volume of phase change material, GST (Ge2Sb2Te5), as shown in Figure 3. The phase change material can be switched from an amorphous phase (reset or "0" state) to a crystalline phase (set or "1" state), or vice versa, with heat. The read and write operations for a PRAM cell is described as follows:

Write Operation: There are two kinds of PRAM write operations, the *SET* operation that switches the GST into

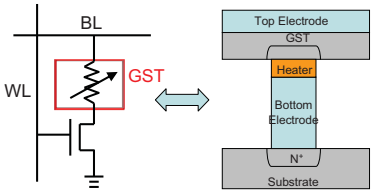

Figure 3: An illustration of a PRAM cell.

crystalline phase and the *RESET* operation that switches the GST into amorphous phase. The *SET* operation crystallizes GST by heating it above its crystallization temperature, and the *RESET* operation melt-quenches GST to make the material amorphous [8]. These two operations are controlled by electrical current: high-power pulses for the *RESET* operation heat the memory cell above the GST melting temperature; moderate power but longer duration pulses for the *SET* operation heat the cell above the GST crystallization temperature but below the melting temperature. The temperature is controlled by passing through a certain amount of electrical current and generating the required Joule heat.

Read Operation: To read the data stored in PRAM cells, a small voltage is applied across the GST. Since the *SET* status and *RESET* status have a large variance on their equivalent resistance, the data is sensed by measuring the pass-through current. The read voltage is set to be sufficiently strong to invoke detectable current but remains low enough to avoid write disturbance. Like other RAM technologies, each PRAM cell needs an access device for control purpose. As shown in Figure 3, every basic PRAM cell contains one GST and one NMOS access transistor. This structure has a name of "1T1R" where "T" stands for the NMOS transistor and "R" stands for GST. The GST in each PRAM cell is connected to the drain-region of the NMOS in series so that the data stored in PRAM cells can be accessed by wordline controlling.

As described, MRAM and PRAM memory technologies are made of different materials than SRAM and eDRAM and have different read/write operations. However, caches constructed from these technologies have similar structure from a logic designer's point of view due to the similarity of the peripheral circuits.

2.3 3D Integration

With continued technology scaling, the on-chip interconnect has emerged as a dominant source of circuit delay and power consumption [16, 17, 29]. 3D ICs have emerged as a promising means to mitigate these interconnect-related problems [9, 16, 29]. Several 3D integration technologies have been explored recently, including wire bonded, microbump, contactless (capacitive or inductive), and through-silicon-via (TSV) vertical interconnects [9]. TSV 3D integration has the potential to offer the greatest vertical interconnect density, and therefore is currently the most promising one among vertical interconnect technologies. In 3D ICs that are based on TSV technology, multiple active device layers are stacked together (through wafer stacking or die stacking) with direct vertical TSV interconnects [29]. 3D ICs offer a number of advantages over traditional two-dimensional (2D) designs [29], such as shorter global interconnects, lower interconnect power consumption, increased on-chip bandwidth and smaller chip footprint. Most relevant to this work, 3D stacking naturally supports the use of mixed-technology integration.

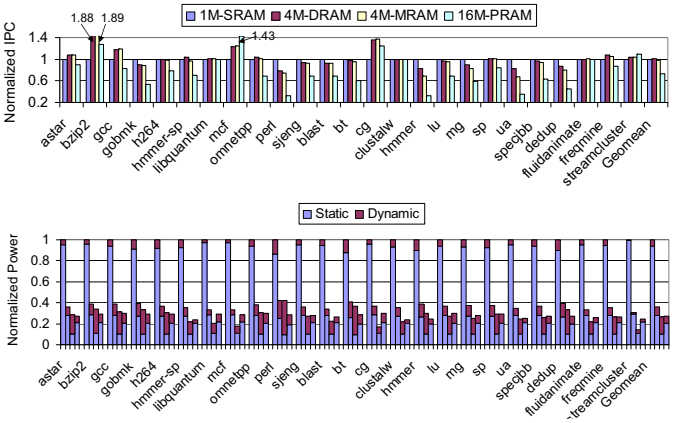

Figure 4: Performance (top, A) and power (bottom, B) comparison of SRAM, eDRAM, MRAM, and PRAM L2 caches under the same area constraint.

3. MOTIVATION

In this section, we present the motivation of our work by comparing the performance and power of caches individually made by pure SRAM, eDRAM, MRAM, or PRAM. We consider a 2-level on-chip cache, where the L2 can be a 1MB SRAM, 4MB eDRAM, 4MB MRAM, or 16MB PRAM under the same area constraint as determined by the appropriate density ratio. We defer the specification of cache parameters (see Table 2) and our simulation methodology to Section 4.

Performance: Figure 4A shows the performance comparison measured by instructions-per-cycle (IPC) of eDRAM, MRAM, and PRAM normalized to SRAM over 25 workloads under the same chip area constraint. Since the memory technologies have significantly different densities and access delays, the cache capacity and latency under the same area budget are also very different. Across the workloads, some applications prefer shorter cache latency over larger capacity, while others do better with the opposite. Some workloads are both cache latency and capacity sensitive, while others are CPU intensive and changes in cache latency and capacity do not affect their performance. As a result, we observe significant IPC variations between SRAM and the other three memory technologies. The rightmost bars are the geometric mean of the workloads. The SRAM-based cache mean IPC is similar to that of a cache constructed from either eDRAM or MRAM. Although the eDRAM and MRAM caches are slower than the SRAM cache, the resulting performance loss is offset by their increased cache capacity. MRAM's relatively long write latency does not greatly hinder the L2 performance. PRAM's average performance is not promising due to its slower read and write access latency, although it does offer the largest capacity. However, we do observe that PRAM is the best choice for a few workloads.

Power: Figure 4B depicts the static and dynamic power comparison for each option for all 25 of the workloads studied. Note that no power-aware design is applied to any of the studied memory technologies in these results. Due to its large static power consumption, SRAM consumes significantly more power than the other three options despite also having the smallest cache capacity. The power consumption variations among eDRAM, MRAM and PRAM can also be seen due to their differing dynamic read, dynamic write, and static power requirements. One may apply customized power-aware techniques to reduce the power consumption of these memory technologies. For example, since MRAM and PRAM are non-volatile, the static power for their memory cells are negligible. Hence, one only needs to reduce the static power of the peripheral circuitry. The static reduction techniques for SRAM and eDRAM tend to be more complicated. On the other hand, the large write energy of MRAM and PRAM, in particular, suggests the need for special power-aware techniques beyond those required for SRAM and eDRAM.

In summary, the divergent latency, density, and power of differing memory technologies may affect the performance and power of a cache depending on what choice is made for the underlying technology. Due to diverse workload characteristics, no single memory technology in consideration has the best power-performance for all workloads under the same chip area constraint. SRAM appears to be much more power hungry than its three counterparts, absent any power-reduction techniques, and consequently gives emerging memory technologies performance leeway in a power constrained design environment. This motivates us to study hybrid caches, which combine the advantages of all these memory technologies in a synergistic fashion for better overall cache power-performance.

4. METHODOLOGY

In this section, we describe our simulation and design methodology. Note that the power and performance comparisons in Figure 4 (Section 3) also use the simulation methodology described here.

4.1 System Configuration

Table 2: Parameters of memory technologies (45nm).

Cache	Norm. Density	Latency (cycles)	Dyn. eng. (nJ)	Static power(W)
SRAM(1MB)	1	8	0.388	1.36
eDRAM(4MB)	4	24	0.72	0.4
MRAM(4MB)	4	read:20 write:60	read:0.4 write:2.3	0.15
PRAM(16MB)	16	read:40 write:200	read:0.8 write:1.5	0.3

We based our parameters on searches of appropriate literature [8, 13, 14, 24, 30] for typical density, latency, and energy numbers for the studied memory technologies, and then scale these to 45nm technology. All cache parameters used in this study were obtained either from CACTI [23] or its modified versions [10]and are shown in Table 2. For all the memory technologies, the cache associativity is 16, the block size is 128B and the bank size is fixed to be 256KB. Since MRAM and PRAM are emerging memory technologies, the projection of their features tends to be more varied than the ones for established technologies such as SRAM and eDRAM, however, we have chosen cache parameters in-line with other researchers' assumptions.

Table 3: System configuration.

Processor	Eight-way issue out-of-order, 8-core, 4GHz
L1	32KB DL1, 32KB IL1, 128B, 4-way, 1 R/W port
L2/L3/L4	See corresponding design cases
Memory	400 cycles lat, memory contr. vs. core speed 1:2

We assume an 8-core CMP system with eight-way issue out-of-order cores. The experiments are conducted using

a full system simulator [5] that has been validated against existing POWER5 hardware [25]. In this paper, we keep the configurations of processor core, L1 caches, on-chip interconnect, and memory system the same, and only study the design of different low-level caches (e.g., L2, L3 or L4) under the same chip area constraint or the same footprint in the case of 3D chip stacking. Table 3 gives our system configuration.

4.2 Workloads

The benchmarks we used in this study are chosen from a wide spectrum of workloads: SpecInt2006 [26], NPB [2], PARSEC [4], BioPerf [1], and SpecJBB [22]. Four PARSEC workloads covering the range of memory footprints of the whole PARSEC suite are selected. Table 4 gives the problem size and other parameters of the benchmarks. For all workloads, we use either sampled reference or native input sets to represent a real-world execution scenario.

Table 4: Workloads.

Benchmarks	Applications and input sizes
SpecInt06	reference input: astar, bzip2, gcc, gobmk, h264 hmmer-sp,libquantum,mcf,omnetpp,perl,sjeng
SPECJBB	IBM JVM version 1.1.8, 16 warehouses
NAS	Class C: cg, lu, mg, sp, ua
BioPerf	reference input: blast, clustalw, hmmer
PARSEC	native: dedup,fluidanimate,freqmine,streamcluster

All results, except those shown in Figure 12, examine workload performance with a single thread of execution. In the case of multithreaded simulations, we run one thread per processor core. In order to reasonably evaluate large cache designs, we construct each simulation in three phases with decreasing simulation speed: (1) we fast forward to a meaningful application phase, which may take 10s - 100s billion of instructions; (2) we warm up the caches by 10s billion of instructions; and (3) we simulate the system cycle-by-cycle for a few billion instructions and collect simulation results. Both performance and power statistics are collected from cycle mode execution. Our cache power model adds the static and dynamic power of the caches used by a workload in the simulation. The static power is obtained from CACTI or its modified versions, as shown in Table 2. The dynamic power factors in the number of read and write accesses and their corresponding per-access energy values are given in Table 2.

4.3 Design Methodology

Throughout the hybrid cache studies presented here, we assume the chip area, or the chip footprint in the 3D integration scenario, is fixed for all the design cases. Figure 5 provides an overview of our design methodology.

In our 2D baseline system, each processor core has three levels of private caches. All three levels of caching are comprised of SRAM. This configuration serves as the *baseline configuration* in this work. One approach to a hybrid cache is to replace SRAM L3 with eDRAM, MRAM or PRAM for larger on-chip cache capacity (Scenario A in Figure 5). This is an inter-cache-Level hybrid cache, or LHCA, which is evaluated in Section 5.

In a 2D chip design scenario, one can merge L2 and L3 to form a hybrid, coarse-grained NUCA cache with L2 fast- and slow-regions made of SRAM and eDRAM/MRAM/PRAM, respectively (Scenario B in Figure 5). The cache regions are mutually exclusive. This is an intra-cache-level or Region-based hybrid cache, RHCA. We discuss this scenario in Section 6. In the same section, we also evaluate a simple power-aware HCA design, called *drowsy hybrid cache*.

In a typical 3D chip stacking design scenario, an L4 cache can be stacked on top of the processor layer. We consider adding a PRAM cache as an L4 due to its high density (Scenario C in Figure 5). Two 3D hybrid caches are shown. One merges all L2, L3 and L4 caches to form a fast, middle, and slow L2–regioned cache with SRAM, eDRAM and PRAM, respectively (Scenario D in Figure 5). The other combines L2 and L3 to form an L2 cache with fast and slow regions comprised of SRAM and eDRAM/MRAM, respectively, with an additional PRAM-based L3 cache (Scenario E in Figure 5). These three design points embody the *3D Hybrid Cache* (3DHCA) and are evaluated in Section 7.

We expect a few benefits of such hybrid cache designs: By applying memory technologies of higher density than SRAM, the effective cache size can increase significantly under the same chip area constraint. Because static power typically dominates in caches, applying non-volatile memory technologies can reduce cache power significantly without extra design overhead. By merging multiple cache levels into one, the multiple cache regions can be checked in parallel, particularly for a lower-level cache, as opposed to the typical approach to sequentially testing each level in the cache hierarchy. When the fast region returns valid data, the search signal to the slow region can be canceled. Additionally, such timing overlapping also offers opportunity for power-aware designs. Due to the mutual exclusivity between the cache regions in a cache level, there are no duplicate cache lines, increasing the effective cache capacity. By reducing the number of cache levels, the coherence traffic between levels is reduced. Within a cache level, the status bit array can be accessed in coordination for better overall allocation of demand (load/store) or prefetch cache lines as well as cache line replacement. Performance can be improved by placing faster cache regions closer to the cache controller and employing mechanisms to maintain hot cache lines in fast regions as appropriate. Power-aware HCA offers extra power-performance benefits to hybrid caches.

5. INTER-LAYER HYBRID CACHE

In the three-level SRAM cache baseline, we use a 256KB L2 cache and a 1MB L3 cache. We assume the cell density ratios between eDRAM, MRAM, and PRAM to SRAM are approximately 4, 4 and 16, respectively, as shown in Table 2. Therefore, a 1MB SRAM, 4MB eDRAM, 4MB MRAM and 16MB PRAM L3 cache all have the same chip area. The type of memory technology in the L3 cache is the only difference between the 3-level SRAM cache baseline system and the LHCA systems with MRAM, eDRAM, or PRAM in our studies.

Performance: Figure 6A shows that the balance between latency and capacity is best stuck by the eDRAM L3 cache configuration. This system delivers the best average performance compared to SRAM, with a 7% improvement due to larger capacity for the same chip area although for some workloads which prefer shorter latency, the benefit of larger capacity is offset by the longer latency. MRAM and PRAM also perform better. Therefore, we choose 256KB L2 + 16-bank 4MB eDRAM L3 as our best 3-level hybrid cache configuration for LHCA. We compare other designs with this configuration in addition to the SRAM-only option in the rest of this paper. Note that even though we

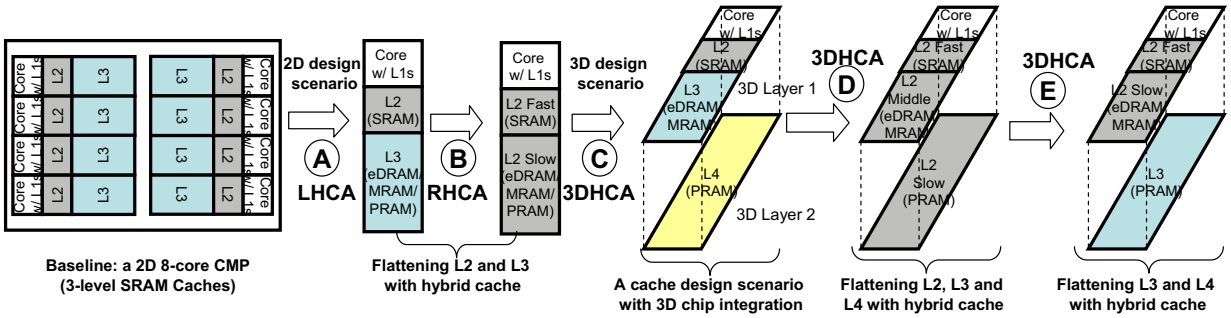

Figure 5: Overview of hybrid cache design methodology.

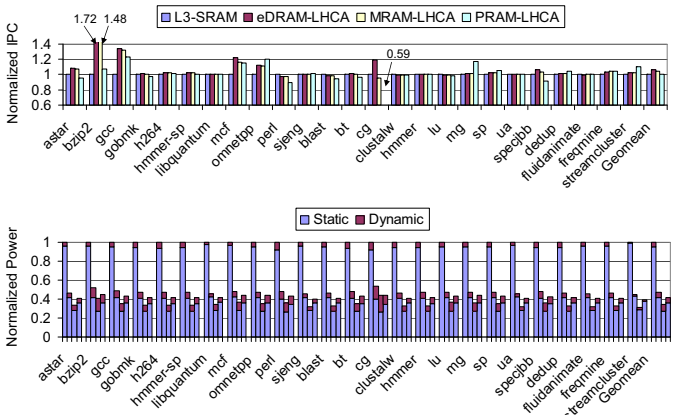

Figure 6: Performance (top, A) and power (bottom, B) comparison of eDRAM/MRAM/PRAM L3 caches and their SRAM baseline.

use larger numbers of banks in larger caches to maintain a constant 256KB per bank as in previous NUCA work [17], our experiments showed negligible performance gain with increasing the number of banks beyond four.

Power: The power comparison is shown in Figure 6B. Similar to Figure 4B in Section 3, we observe significant power savings by replacing SRAM with other memory technologies. Interestingly, the MRAM power results show the SRAM-MRAM hybrid cache to consume the least normalized power, in contrast to the results in Figure 4B. This is because the insertion of a SRAM L2 that is much larger than the L1 removes many accesses that would have otherwise hit in the MRAM L3. Therefore, the MRAM power consumption, particularly write power, is reduced significantly. Recall in Table 2 that MRAM's read and static power is very low. However, due to the addition of a 256KB SRAM L2 to all system configurations, the relative static power consumption of all configurations increases compared to Figure 4B. Note that no power-aware design is applied in these experiments. Power savings come entirely from introducing low-power memory technologies.

Hit rate: Figure 7 depicts the L2 plus L3 hit rate for these four design cases (L2 hit count plus L3 hit count and then divided by the total access count from the processor). With increased cache capacity that is enabled by eDRAM, MRAM and PRAM technologies, the hit rate either remains the same or increases depending on the cache requirements of the workloads. However, the IPC performance is not necessarily improved accordingly due to the longer latency of

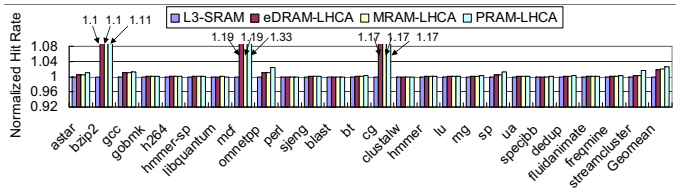

Figure 7: L2 plus L3 hit rate comparison of eDRAM/MRAM/PRAM L3 caches and their SRAM baseline.

these technologies, e.g., in mcf and cg, by comparing Figure 6A and Figure 7. Indeed, hit rate and other cache evaluation metrics typically reveals only part of the performance aspects of a cache design. In the following, we focus on analyzing the IPC performance improvement with our design.

6. REGION-BASED HCA

In this section, we examine the performance and power consumption of the RHCA caches consisting of one fast region made of SRAM and one slow region made of eDRAM, MRAM, or PRAM.

6.1 Hybrid L2 Cache

We propose to flatten the cache hierarchy by merging the eDRAM or MRAM or PRAM L3 into the SRAM L2. The resulting L2 cache thus consists of one small fast (SRAM) region and one large slow (eDRAM or MRAM or PRAM) region, as shown in Scenario B in Figure 5. The large hybrid L2 cache has the potential of providing fast-region access time and large-region capacity simultaneously. Fully exploring this potential requires proper cache line replacement and data migration policies.

6.1.1 Cache Line Migration Policy

Figure 8 depicts the cache line migration policy we use in our RHCA design. This policy uses one sticky bit for each line in the fast region and a two-bit saturating counter for each line in the slow region to control data movement between regions. The saturating counter records the access frequency and the bit indicates if a line cannot be moved, thus *sticky*. We consider a line whose access frequency counter is over the predetermined threshold "hot". A "hot" line is a candidate for migrating into the fast region. Unless noted otherwise, we use $0b'10$ as the threshold value, which is equivalent to the MSB bit of the two-bit counter being 1. On a cache miss, a new line is loaded into the hybrid cache. The new

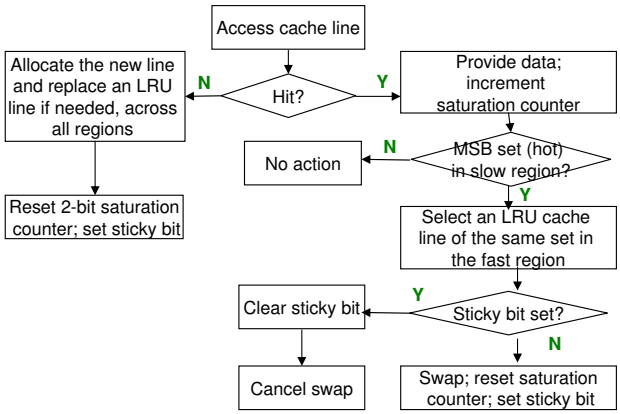

Figure 8: Cache line allocation and migration policy in the RHCA.

line always replaces the LRU line regardless of the region where the LRU line is located. If the new line is inserted into the fast region, its sticky bit is initiated to be 1. If the new line is inserted into the slow region, its saturating counter is initialized to 0.

On a cache hit, if the corresponding cache line resides in the fast region, its sticky bit is always set. If the line resides in the slow region, its saturating counter is incremented by one. If the counter reaches the threshold value, a swap operation to move the line into the fast region will be attempted. The LRU line in the fast region within the same set is selected as the potential destination. If the sticky bit of the selected line in the fast region is 0, we swap the two lines, set the sticky bit for the line moved into the fast region, and reset the saturating counter for the line moved into the slow region. If the sticky bit of the selected line in the fast region is set, we clear the sticky bit and cancel the swap operation. The role of the sticky bit is to protect a line in the fast region once and therefore effectively delay the swap once to avoid unnecessary swapping of lines.

We adopt simple mapping scheme from [17] for our RHCA design. For example, if the cache has 16 ways and there are 16 banks then each cache way can be mapped to any of these banks. In this case, each bank holds one way of the cache data. In order to perform quick search, as we mentioned in Section 4, the multiple cache regions can be checked in parallel. When the fast region returns valid data, the search signal to the slow region can be canceled.

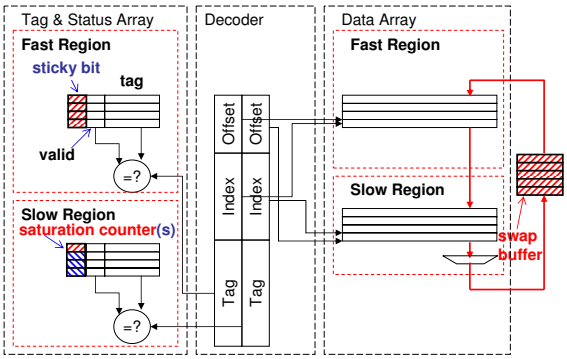

Figure 9: Block diagram of the proposed RHCA. Structures with slash patterns are new components.

6.1.2 *Hardware Support*

The hardware support for the swap operations is shown in Figure 9. The fast region and the slow region each has a tag and status array (left) as well as data array (right) allocated on both sides of the address decoder. The address decoder is replicated to meet timing demands. The trade-offs between the number of replicated decoders and bank partition is beyond the scope of this paper. The main additions are the saturating counters, the sticky bits, and the swap buffer.

A swap operation involves reading out two cache lines from two regions and writing each to the opposite region. Because of the speed difference between two regions and the contention on the cache arrays, a line read out a region may not be able to go to the opposite region immediately and therefore must be temporarily buffered elsewhere. To simplify logic, we propose to utilize a swap buffer and serialize the swap operation as follows. First the data in the slow region is read out and placed into the buffer. Then the data in the fast region is read out and written to the slow region. Finally, the line in the swap buffer is written to fast region. Each of the three steps may take multiple cycles. The swap buffer contains multiple entries and allows multiple outstanding swap operation. Note that the first step is already being done as part of the process of loading the line into the upper level cache. We simply need save the line in the swap buffer before it can be written to the fast region.

An alternative approach is to read both lines in parallel. In this approach, the swap buffer is either double-ported or specially arranged to allow two writes simultaneously. Indeed, the swap support can also be implemented by extending the read and replacement hardware in the existing caches. We have evaluated the sensitivity of swap latency and swap buffer size. The swap buffer is snooped for coherence operations. A snoop hit in the swap buffer will result in a retry response in our simulated system. Our simulation results indicate that such scenario rarely happens and is not a concern for performance degradation. More detailed analysis of swap buffer is shown in Section 6.3.

The base RHCA design uses a saturating counter for every line in the slow region. A saturating counter can be implemented with about 20 transistors, which is a very small overhead relative to the much larger tag and data arrays (hence very small power overhead). To further reduce the overhead of the saturating counters, a saturating counter can be used for a group of cache lines. In such a case, the saturating counter will record the access frequency of the corresponding group of cache lines. A more important goal for such placement is to see if there is a constructive aliasing behavior between the accesses of the cache lines in the same group. In other words, if the cache lines in the same group have similar access patterns, the saturating counter will be able to trigger cache line swap to the faster region earlier and help improving the performance. Furthermore, a placement of one saturating counter per group of cache lines also facilitates *multi-line swap*, which moves a group of cache lines at one time. The multi-line swap has the "effects" of intra-cache prefetching because it pushes some lines into the fast region based on access patterns of other lines.

6.1.3 *Drowsy RHCA*

The coordination support among the regions of a hybrid cache and the parallel address search among cache regions also opens new directions for power-aware designs. One sim-

ple yet effective approach is to keep the slow region of a hybrid cache in drowsy mode [12]. For a slow region made of eDRAM, the refresh operations can be operated at a slower rate with lower voltage. In the case of a slow region made of a non-volatile memory, drowsy mode can be power-gating the non-volatile memory cells and/or corresponding peripheral CMOS logic. A cache line in a slow region is woken up only when it needs to provide data or execute swap operations.

6.2 Results

In this section, we present the experimental results for RHCA and drowsy RHCA.

6.2.1 Single-threaded performance

Table 5: Fast-slow region hybrid cache L2 parameters.

RHCA (fast + slow)	Fast region	L2 size (latency)
SRAM + eDRAM	256KB (6 cycles)	4MB (24 cycles)
SRAM + MRAM	256KB (6 cycles)	4MB (r: 20; w: 60)
SRAM + PRAM	256KB (6 cycles)	16MB (r: 40; w: 200)

RHCA can be SRAM-eDRAM, SRAM-MRAM or SRAM-PRAM based, similar to the LHCA design points. The RHCA cache design parameters for the proposed hybrid L2 cache are listed in Table 5. For all design cases, the block size is 128B, the bank size is 256KB and the cache has one read/write port. The cache associativity is 16 and 64 for the first two cases and the third case, respectively. We compare the RHCA designs with the 3-level SRAM-only baseline as shown in Figure 5 and the SRAM-eDRAM based LHCA (the best LHCA in Section 5). Note that in Table 5, each RHCA configuration is 256KB less in total size compared to its LHCA counterpart. This is to avoid complicated indexing schemes often associated with odd-sized caches. Nonetheless, even with the slightly smaller capacity, the RHCA still outperforms LHCA. We also compare our counter-based data migration design with the generational promotion approach first proposed for DNUCA by Kim et al.[17]. Generational promotion moves a line to a closer bank on each hit. Not that in the DNUCA configuration, the mapping, search, and replacement polices are the same with our RHCA.

Figure 10A illustrates the performance of SRAM-eDRAM based RHCA. We observe that different benchmarks have different results because they have various cache requirement: some benchmarks prefer large size and some prefer small latency. Basically, there are three categories: 1) benchmarks that require large cache size, have relatively high miss rate, and can obtain benefit from cache locality (e.g. *bzip*2, *mcf*, *omnetpp*, *cg* and *mg*). For these benchmarks, RHCA offers high performance improvement; 2) benchmarks that require modest cache size (e.g., *hmmer* and *sp*), the benefit of RHCA may not outperform data movement overhead so that the performance of RHCA is slightly degraded compared to 3-level SRAM baseline and LHCA; 3) benchmarks that are not memory intensive (e.g., *clustalw*), the performance is almost the same for the four comparison cases. Although different benchmarks show various performance trend, we still observe that the RHCA design has a geometric mean performance improvement of almost 9% over the SRAM-only design. It is 2% faster than LHCA and 4% faster than DNUCA. RHCA outperforms DNUCA because of the difference in the speed of data movement. RHCA moves frequently-accessed cache lines directly

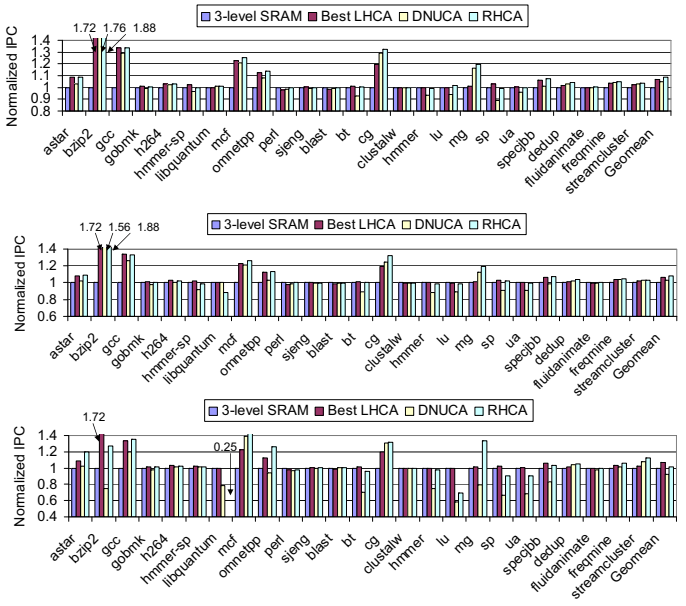

Figure 10: Performance of SRAM-eDRAM (top, A), SRAM-MRAM (middle, B) and SRAM-PRAM (bottom, C) RHCA.

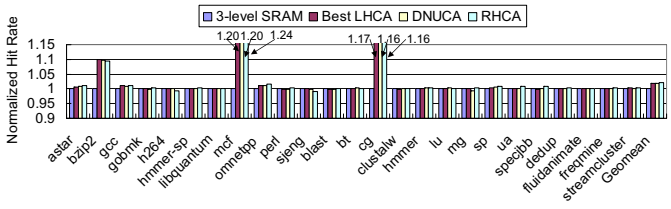

Figure 11: L2/L2 plus L3 hit rate comparison.

to the fast region from any bank in the slow region. DNUCA moves cache lines one bank at a time. Because of the large number of banks in the slow region, it often requires many more hits before a line moves into the fast region. In addition, since the swap frequency is high in DNUCA it results in higher swap overhead compared to RHCA.

Figure 11 illustrates the hit rate comparisons for 3-level SRAM baseline, best LHCA, DNUCA, and SRAM-eDRAM RHCA (L2 plus L3 hit rate for the first two cases, L2 hit rate for the remaining two cases). It is interesting to observe that although the hit rate for the latter three cases are close, their IPCs (Figure 10A) vary more due to the difference of hit counts in different cache regions/levels, which have differing latencies. The more pronounced latency differences in LHCA and RHCA help them outperform 3-level SRAM baseline and DNUCA.

Figure 10B depicts the performance for the SRAM-MRAM RHCA design. It shows that SRAM-MRAM RHCA has similar, albeit slightly reduced, performance than SRAM-eDRAM RHCA. This is interesting, because it means that MRAM's relatively slower write latency does not hurt its overall performance due to its faster read operations. Figure 10C shows that SRAM-PRAM RHCA has a 6% performance degradation relative to SRAM-eDRAM LHCA due to its long write latency. Nonetheless, it has slightly better performance than the 3-level SRAM baseline configuration.

In summary, SRAM-eDRAM RHCA can achieve a larger performance improvement over the SRAM-only cache design

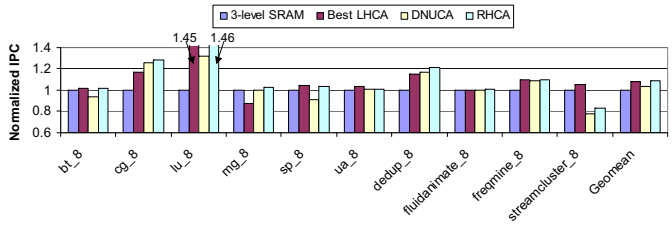

Figure 12: Performance comparison for multi-core design

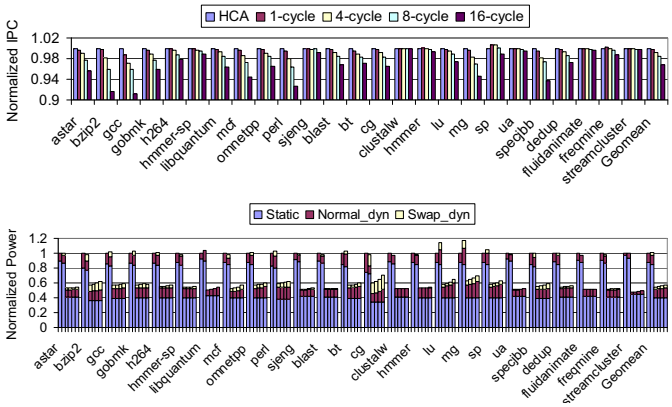

Figure 13: Performance (top, A) and power (bottom, B) comparison for SRAM-eDRAM RHCA with different wake-up latencies.

and also outperforms SRAM-eDRAM LHCA. The SRAM-PRAM RHCA design is not very promising for an L2 cache due to the long latency of PRAM. However, we will show that it is a promising technology for lower level caches due to its high density in Section 7. SRAM-eDRAM RHCA and SRAM-MRAM RHCA have similar performance. We focus on the analysis of SRAM-eDRAM RHCA in the following sections.

6.2.2 Multi-threaded performance

Previous experiments focused on the RHCA evaluation for single-threaded runs of the workloads. In this section, we examine the RHCA designs on an eight-core CMP. We adopt the same line migration policy in the single-core design. Figure 12 shows the performance comparison between the SRAM-eDRAM RHCA design, the two baselines, and the DNUCA design for multi-thread benchmarks. As a matter of fact, RHCA has modestly better speedups for multi-threaded runs than for single-threaded runs compared to LHCA and DNUCA because RHCA in the CMP moves hot lines and their tags closer to the cores more efficiently, effectively reducing the overhead of coherence traffic. We expect a multi-core oriented RHCA design will improve multi-threaded workloads further. Note that, *streamcluster* shows performance degradation in DNUCA and RHCA, indicating inefficient data swap for the dominant writes in this workload. As a result, region-based cache techniques that facilitate read- and write-often data structures will help [28].

6.2.3 Power savings with drowsy hybrid cache

To generalize the evaluation of the hybrid drowsy cache option yet maintain a reasonable power comparison, we assume fixed static power of zero in the drowsy mode and only vary the wake-up and sleep transition time. In addition, we assume linearly-scaled transition voltage of wake-up and sleep transitions. The wake up time from sleep mode to fully functional mode is set to be 1/4/8/16 cycles in order to study the performance loss and power saving under different wake-up times. The performance and power comparison among the baseline and different wake-up latencies are shown in Figure 13.A and Figure 13.B (the six bars in Figure 13.B are the power for LHCA, RHCA, and power aware design with four different wake-up cycles). The results indicate that up to 8 cycles wake-up latency incur less than 1% loss in IPC. With 16 cycles wake-up latency, the performance degradation compared to HCA is less than 3%. In exchange, average power saving is around 50%. Note that the power saving is beyond what the LHCA design have achieved (53% in Section 5). In Figure 13.B, the power consumption is partitioned into three segments: static power, dynamic power for normal data access and dynamic power for data migration (swap operations). The remaining static power of the drowsy hybrid cache mainly comes from the fast region, which is not drowsy. The configuration of the plots here uses 2-hit saturating counter threshold. If using 1-hit threshold, one would expect more swap power. Due to the swap power overhead, the power consumption of RHCA is slightly higher than that of LHCA.

6.3 Sensitivity Study

In this section, we briefly describe several sensitivity studies for SRAM-eDRAM based RHCA design.

6.3.1 Threshold sensitivity of saturating counters

The default threshold of 2 means that a line swap is triggered if it has been hit twice. A larger threshold makes more conservative choices and may miss more opportunities. A lower threshold makes more aggressive choices and may thrash the fast region.

To better understand the effects of the threshold, we vary the threshold from one hit to three hits in our experiments and show the performance and power results in Figure 14. From the performance perspective, the results show that most of benchmarks are insensitive to the threshold. A few benchmarks, *bzip2*, *cg*, *lu*, and *specjbb*, prefer three hits (reduce cache pollution), while *mg* prefers one hit (one possible reason is that lots of data are used twice or three times). The performance difference between different thresholds is less than 1%. These results suggest that a one-hit threshold is sufficient for most applications, which effectively can be substituted by the existing LRU bit in the status array. However, we also find that higher threshold values tend to help prefetched data more, and to reduce cache pollution to the fast region.

From the power perspective, many benchmarks consume more swap power when the counter threshold is one because there are more swap operations. Some benchmarks have different trend since the performance is also considered when calculating the power consumption. More power may be consumed with even lower energy consumption due to the delay difference. For some benchmarks, the swap power for 1-hit is very small. The reason is that in our simulation, before the swap occurs, the cache line is checked to guarantee that it does not reside in other queues (waiting for some requests). Therefore, the swap operations may be much less than the hit counts if many cache lines are in queues before swapping. Another observation is that more swap operations offer higher performance improvement in RHCA (e.g.,

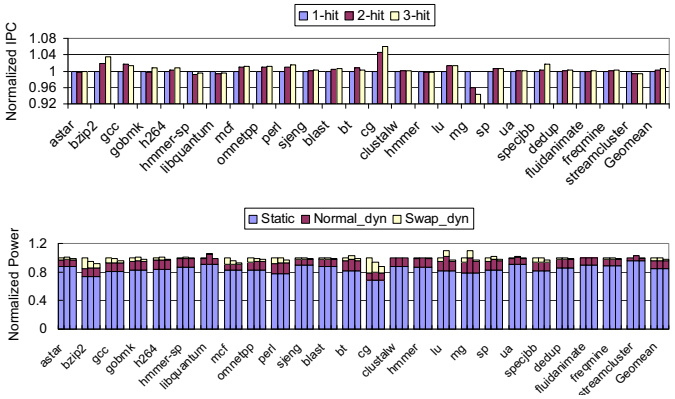

Figure 14: Performance (top, A) and Power (bottom, B) comparison for different threshold value of saturating counters for SRAM-eDRAM RHCA.

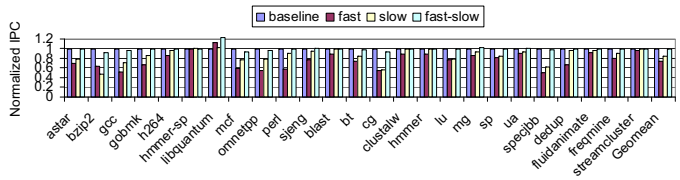

Figure 15: Different replacement policies comparison for SRAM-eDRAM RHCA.

*bzip*2, *mcf* and *cg*), indicating that swap helps to reduce the latency for frequently accessed data for these benchmarks.

6.3.2 Sensitivity of swap latency and buffer entries

As discussed in Section 6.1.2, a swap may be done in three steps with a two swap buffers or two steps with one swap buffer. For a design with two swap buffers, we conservatively assume that one swap operation takes 38 cycles. It includes one read/write latency for the fast region (6 cycles), one read/write latency for the slow region (24 cycles), and two bus transfers. For a bus width of 32 bytes, transferring 128 bytes cache line data from the cache/buffer to the buffer/cache takes 4 cycles. On the other hand, for a design with one swap buffer, we assume that one swap operation requires 76 cycles (6*2 + 24*2 + 4*4 cycles). We find that many workloads perform better with the two-step swap process and 16 entries are sufficient for all the benchmarks.

6.3.3 Replacement and insertion policy

In the previous study, data replacement is based on LRU bits to choose the LRU line in both regions. We have also evaluated three other replacement policies: (1) Replace the line in the fastest bank and evict this line to lower level; (2) Replace the line in the slowest bank and evict this line to lower level; (3) Put the line in the fastest bank, move the existing line in the fastest bank to a randomly selected bank in slow region, and evict the line in the selected slow bank. These results are shown in Figure 15. Again, performance is application dependent. However, overall we observe that LRU-based policy performs equally well with option 3 for most workloads, both of which are better than the other two options. This observation indicates that HCA cache policy that builds upon conventional LRU support is simple and effective.

6.3.4 Adaptive and Multi-line Swap

From the comparison between LHCA and RHCA in Section 6.2.1, we observe that swaps improves overall performance of the 25 workloads. However, they can also pollute the cache for some of them, particularly when swap frequency is low. Based on this observation, we implement a simple adaptive swap scheme in the RHCA design. When the ratio between the number of swaps and the number of accesses to the slow region is less than a threshold (in our evaluation, the threshold is set to be 15%), we disable swap operations and make the slow region become a victim buffer of the fast region.

For multi-line swap, we can use a counter to keep track of accesses to a corresponding group of cache lines instead of each individual line. There are two ways to update counters in this case. The first is to update the counter based on hits to a given line in the group, but the movement is group based. The second is to update the counter on hits to any line in the group. We use the first counting mechanism. When a swap is triggered, multiple cache lines, two in our experiments, can be moved in tandem. In addition, three hits threshold is applied to help prefetch.

Overall, with the inclusion of adaptive swapping and multi-line movements, RHCA outperforms LHCA by around 5% in mean IPC improvement as shown between the "Best LHCA" and "RHCA" bars in Figure 16 (Section 7).

7. 3D HYBRID CACHE STACKING

3D cache stacking enables the addition of more cache levels without sacrificing the number of cores. These extra cache levels should be at least a few times larger than the cache level above it in the cache hierarchy to effectively reduce miss rate. We assume the 3D cache layer has the same footprint as its corresponding 2D processor core and the original caches attached to the core. If a memory technology of the same density is used, then multi-layer 3D cache stacking is anticipated. However, multi-layer 3D stacking may incur mounting problems in power delivery, cooling, and TSV efficiency. Therefore, we expect a denser memory technology to be an alternative approach to multi-layer 3D cache stacking. In this paper, we consider PRAM (Scenario C in Figure 5). Besides its high density, PRAM also has very low static power, which further helps address the cooling issues with 3D. We scale the latency and power parameters of PRAM as shown in Table 2 for the 3D cases. We assume the processor and memory domain clock frequencies of 3D are the same as its 2D counterpart.

Similar to the approach in Section 6, one can combine some or all of the L2, L3 and L4 caches to obtain a region-based hybrid cache (RHCA). The design issues of this extension are also similar to what we have discussed in Section 6. We also study two more design scenarios: combining all L2, L3 and L4 to obtain a three-region hybrid cache of fast, middle and slow regions (Scenario D in Figure 5); and combining L2 and L3 as we did in Section 6.1 and logically upgrade L4 to L3 (Scenario E in Figure 5). As a result, we have three 3DHCA designs for comparison: (1) Four-level caches with 256KB SRAM L2, 4MB eDRAM L3 and 32MB PRAM L4 (3HCA-C, Scenario C in Figure 5); (2) A three-region RHCA with 256KB SRAM fast region, 3.75MB eDRAM middle region and 28MB PRAM slow region (3DHCA-D, 32MB in total, Scenario D in Figure 5); (3) A 4MB SRAM-eDRAM

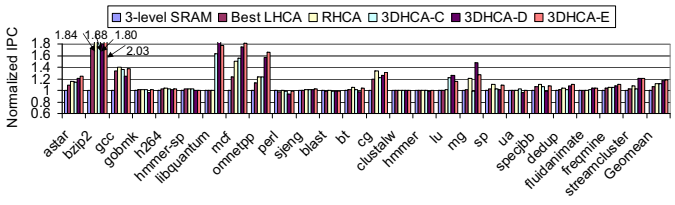

Figure 16: 3DHCA performance comparison.

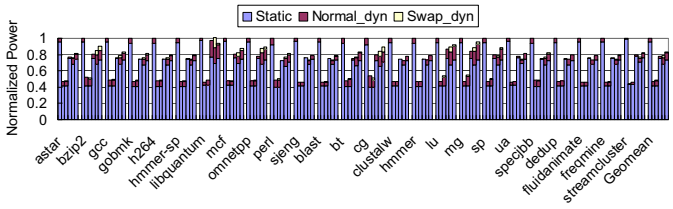

Figure 17: 3DHCA power comparison.

RHCA L2 (Section 6) and 32MB L3 cache (3DHCA-E, Scenario E in Figure 5). In 3DHCA-D, the frequently used cache line in the PRAM-based slow region can be mitigated to fast region as well as middle region (LRU-based replacement policy) in order to prevent thrashing the fast region. The frequently used cache line in the middle region is mitigated to the fast region, which is the same as two-region RHCA.

7.1 Performance and power evaluation

Figure 16 illustrates the performance comparison of these three design cases with the 3-level SRAM baseline, SRAM-eDRAM LHCA (Scenario A in Figure 5) and SRAM-eDRAM RHCA (Scenario B in Figure 5) with adaptive swapping and multiline movements. The results show that all three design cases exhibit large improvements over 3-level SRAM baseline and SRAM-eDRAM LHCA. In addition, they achieve on average 0.5%, 4% and 6% IPC improvement over SRAM-eDRAM RHCA. Among them, Scenario E has better performance than both Scenario C and D and it achieves 18% IPC improvement than the pure SRAM baseline. We observe that although the total cache capacity of 3DHCA-D is smaller than that of 3DHCA-C, it has average better performance, indicating that the multi-region hybrid cache is efficiently used to take advantage of the latency and capacity tradeoffs. Since 3DHCA-E has best performance in 3D design cases, it indicates that RHCA performs better in middle level caches, such as the L2, where both latency and capacity issues affect sensitive applications.

Figure 17 illustrates the power comparison of these three 3D design cases with 3-level SRAM baseline, SRAM-eDRAM LHCA, and SRAM-eDRAM RHCA (the six bars in Figure 17 are the power for 3-level SRAM, LHCA, RHCA, 3DHCA-C, 3DHCA-D, and 3DHCA-E). We observe that the leakage power of 3D design cases are much larger than that of LHCA and RHCA because of the peripheral circuits for the extra PRAM layer. Among the three 3D design cases, 3DHCA-E assumes more power than 3DHCA-C and 3DHCA-D. The power of 3DHCA-D is slightly larger than that of 3DHCA-C although the capacity of 3DHCA-D is smaller. The reason is that the swap operations in 3DHCA-D occur power overhead. The results indicate that all three design cases have higher power consumption than LHCA and RHCA in 2D case but still have lower power consumption than 3-level SRAM baseline.

8. RELATED WORK

There are several NUCA studies for single core and chip multi-processors (CMP) in the literature [3, 7, 15, 17]. Kim et al propose the novel NUCA concept for large caches and compare several SNUCA and DNUCA designs [17] in which data movement is based on generational promotion. Subsequently, distance associativity based NUCA, called NuRapid, is proposed in single core and multi-core designs [7]. NuRapid decouples data placement from tag placement by separating it from set associativity. In [3], transmission line based NUCA is presented for multi-core design and a prefetch scheme is evaluated for performance improvement. Recently, the sharing degree of shared L2 NUCA caches in CMP design was examined [15]. However, in these NUCA designs, the access latency differences are mainly from interconnect delays. In our RHCA design, the latency as well as power differences are from disparate memory technologies. Additionally, our RHCA is a hierarchical design. At a high level, RHCA is made of cache regions of different sizes with differing memory technologies. At a base level, a cache region itself can be a conventional NUCA.

Recently, 3D has enabled mixed-technology integration and offers advantages of lower global wire delay and smaller area. Previous work focuses on the performance improvement and power reduction by stacking cache or main memory on top of processors [6, 18, 19, 20]. Typically, these stacked caches are SRAM-based, as opposed to our LHCA design. In addition to LHCA, we also evaluate RHCA and 3DHCA, which may consist of a level of RHCA cache, to fully explore the HCA design space.

In the context of stacking DRAM memory in 3D, Loh [20] proposes an interesting 3D design of on-chip main memory to boost chip performance. Recently, a reconfigurable cache in 3D stacking is proposed, in which the baseline cache is made of SRAM and a reconfigurable eDRAM based cache can be turned on/off based on the cache size requirements [21]. Another work studied SRAM-MRAM based hybrid cache to improve performance of pure MRAM based cache [27]. Read-write aware hybrid cache design combining SRAM with non-volatile memories MRAM/PRAM is also evaluated [28]. In this work, we study eDRAM, MRAM and PRAM in our HCA cache design, which can be applied in both a 2D and 3D context.

9. CONCLUSIONS AND FUTURE WORK

In this paper, we have presented a hybrid cache architecture to construct on-chip cache hierarchies with differing memory technologies. We have proposed both inter- and intra-cache level hybrid cache designs. We have studied hybrid caches made of combinations of SRAM, eDRAM, MRAM and PRAM under the same area constraint. In addition, we have proposed and evaluated low-overhead intra-cache data movement power-aware policies and their hardware support to both improve cache performance and reduce power. For a collection of 25 workloads, the geometric mean of simulation results based on a hardware calibrated full-system simulator show that an inter-cache-layer HCA design can provide 7% IPC improvement over a baseline 3-level SRAM cache design under the same area constraint. A more aggressive RHCA-based design provides 12% IPC improvement over the baseline. Finally, a 2-layer 3D cache

stack (3DHCA) of high density memory technology within the same chip footprint gives 18% IPC improvement over the baseline. We also observe a power reduction of up to 70% across all configurations.

Overall, we have shown the potential of applying hybrid caches to re-balance the cache subsystem design, and we have discussed a design direction to further improve hybrid cache power-performance. As an initial study, we have mainly presented hybrid cache performance in the context of single-threaded execution. Multithreaded workloads on chip multiprocessors opens further avenues for exploration beyond the initial results presented here. Furthermore, although we focus on power-performance perspective in our HCA design, thermal, reliability, and endurance (e.g. current PRAM has low endurance) are also important issues that need further investigation.

10. ACKNOWLEDGMENTS

We owe a debt gratitude to Elmootazbellah N. Elnozahy for his guidance and support. Balaram Sinharoy, Hung Le, William J. Starke and Chung-Lung Shum have offered helpful insights to refine the ideas in this work. We also appreciate the insightful comments and constructive suggestions from the anonymous reviewers. Xiaoxia Wu and Yuan Xie were supported in part by NSF grants 0702617 and 0643902.

11. REFERENCES

[1] D. A. Bader, Y. Li, T. Li, and V. Sachdeva. BioPerf: A Benchmark Suite to Evaluate High-performance Computer Architecture on Bioinformatics Applications. In *Proceedings of the 2005 IEEE International Symposium on Workload Characterization*, pages 163–173, 2005.

[2] D. Bailey, J. Barton, T. Lasinski, and H. Simon. The NAS parallel benchmarks. In *Technical report RNR-91-002 revision2*, pages 453–464, 1991.

[3] B. M. Beckmann and D. A. Wood. Managing Wire Delay in Large Chip-Multiprocessor Caches. In *International Symposium on Microarchitecture*, pages 319–330, 2004.

[4] C. Bienia, S. Kumar, J. P. Singh, and K. Li. The PARSEC Benchmark Suite: Characterization and Architectural Implications. In *Proceedings of the 17th International Conference on Parallel Architectures and Compilation Techniques*, October 2008.

[5] P. Bohrer, J. Peterson, M. Elnozahy, R. Rajamony, A. Gheith, R. Rockhold, C. Lefurgy, H. Shafi, T. Nakra, R. Simpson, E. Speight, K. Sudeep, E. V. Hensbergen, and L. Zhang. Mambo: a full system simulator for the powerpc architecture. *SIGMETRICS Perform. Eval. Rev.*, 31(4):8–12, 2004.

[6] B. Bryan, A. Murali, B. Ned, D. John, J. Lei, H. L. Gabriel, M. Don, M. Pat, W. N. Donald, P. Daniel, R. Paul, R. Jeff, S. Sadasivan, S. John, and W. Clair. Die Stacking (3D) Microarchitecture. In *International Symposium on Microarchitecture*, pages 469–479, 2006.

[7] Z. Chishti, M. D. Powell, and T. N. Vijaykumar. Optimizing Replication, Communication, and Capacity Allocation in CMPs. *SIGARCH Comput. Archit. News*, 33(2):357–368, 2005.

[8] L. Chung. Cell Design Considerations for Phase Change Memory as a Universal Memory. In *International Symposium on VLSI Technology, Systems and Applications*, pages 132–133, 2008.

[9] W. R. Davis, J. Wilson, S. Mick, J. Xu, H. Hua, C. Mineo, A. M. Sule, M. Steer, and P. D. Franzon. Demystifying 3D ICs: the Pros and Cons of Going Vertical. *IEEE Design and Test of Computers*, 22(6):498– 510, 2005.

[10] X. Dong, X. Wu, G. Sun, Y. Xie, H. Li, and Y. Chen. Circuit and microarchitecture evaluation of 3D stacking magnetic RAM (MRAM) as a universal memory replacement. In *Design Automation Conference*, pages 554–559, 2008.

[11] X. Dong and Y. Xie. System-level Cost Analysis and Design Exploration for Three-Dimensional Integrated Circuits (3D ICs). In *Asia and South Pacific Design Automation Conference*, 2009.

[12] K. Flautner, N. S. Kim, S. Martin, D. Blaauw, and T. Mudge. Drowsy caches: simple techniques for reducing leakage power. *SIGARCH Comput. Archit. News*, 30(2):148–157, 2002.

[13] S. Hanzawa, N. Kitai, K. Osada, A. Kotabe, Y. Matsui, N. Matsuzaki, N. Takaura, M. Moniwa, and T. Kawahara. A 512KB Embedded Phase Change Memory with 416kB/s Write Throughput at 100uA Cell Write Current. In *IEEE International Solid-State Circuits Conference*, pages 474–616, 2007.

[14] M. Hosomi, H. Yamagishi, T. Yamamoto, and et al. A Novel Nonvolatile Memory with Spin Torque Transfer Magnetization Switching: Spin-RAM. In *International Electron Devices Meeting*, pages 459–462, 2005.

[15] J. Huh, C. Kim, H. Shafi, L. Zhang, D. Burger, and S. W. Keckler. A NUCA substrate for flexible CMP cache sharing. In *International Conference on Supercomputing*, pages 31–40, 2005.

[16] J. W. Joyner and J. D. Meindl. Opportunities for Reduced Power Dissipation Using Three-dimensional Integration. In *Interconnect Technology Conference*, pages 148–150, 2002.

[17] C. Kim, D. Burger, and S. W. Keckler. An adaptive, non-uniform Cache Structure for Wire-delay Dominated On-chip Caches. In *International Conference on Architectural Support for Programming Languages and Operating Systems*, pages 211–222, 2002.

[18] F. Li, C. Nicopoulos, T. Richardson, Y. Xie, V. Narayanan, and M. Kandemir. Design and Management of 3D Chip Multiprocessors Using Network-in-Memory. *International Symposium on Computer Architecture*, 34(2):130–141, 2006.

[19] C. Liu, I. Ganusov, M. Burtscher, and S. Tiwari. Bridging the Processor-memory Performance Gap with 3D IC Technology. *IEEE Design and Test of Computers*, 22(6):556– 564, 2005.

[20] G. H. Loh. 3D-Stacked Memory Architectures for Multi-core Processors. In *International Symposium on Computer Architecture*, pages 453–464, 2008.

[21] N. Madan, L. Zhao, N. Muralimanohar, A. Udipi, R. Balasubramonian, R. Iyer, S. Makineni, and D. Newell. Optimizing communication and capacity in a 3D stacked reconfigurable cache hierarchy. In *High Performance Computer Architecture*, pages 262–274, Feb. 2009.

[22] R. Morin, A. Kumar, and E. Ilyina. A multi-level comparative performance characterization of specjbb2005 versus specjbb2000. In *Proceedings of the IEEE International Workload Characterization*, pages 67–75, Oct. 2005.

[23] N. Muralimanohar, R. Balasubramonian, and N. Jouppi. Optimizing NUCA Organizations and Wiring Alternatives for Large Caches with CACTI 6.0. In *Proceedings of the 40th Annual IEEE/ACM International Symposium on Microarchitecture*, pages 3–14, Washington, DC, USA, 2007. IEEE Computer Society.

[24] F. Pellizzer, A. Pirovano, F. Ottogalli, M. Magistretti, M. Scaravaggi, P. Zuliani, M. Tosi, A. Benvenuti, P. Besana, S. Cadeo, T. Marangon, R. Morandi, R. Piva, A. Spandre, R. Zonca, A. Modelli, E. Varesi, T. Lowrey, A. Lacaita, G. Casagrande, P. Cappelletti, and R. Bez. Novel utrench Phase-change Memory Cell for Embedded and Stand-alone Non-volatile Memory Applications. In *Symposium on VLSI Technology*, pages 18–19, 2004.

[25] B. Sinharoy, R. N. Kalla, J. M. Tendler, R. J. Eickemeyer, and J. B. Joyner. Power5 system microarchitecture. *IBM J. Res. Dev.*, 49(4/5):505–521, 2005.

[26] SPEC. Standard Performance Evaluation Corporation. http://www.spec.org/cpu2006/. 2006.

[27] G. Sun, X. Dong, Y. Xie, J. Li, and Y. Chen. A novel architecture of the 3D stacked MRAM L2 cache for CMPs. In *High Performance Computer Architecture*, pages 239–249, Feb. 2009.

[28] X. Wu, J. Li, L. Zhang, E. Speight, and Y. Xie. Power and Performance of Read-Write Aware Hybrid Caches with Non-volatile Memories. In *Design, Automation and Test in Europe*, 2009.

[29] Y. Xie, G. H. Loh, B. Black, and K. Bernstein. Design Space Exploration for 3D architectures. *J. Emerg. Technol. Comput. Syst.*, 2(2):65–103, 2006.

[30] W. Zhao, E. Belhaire, Q. Mistral, C. Chappert, V. Javerliac, B. Dieny, and E. Nicolle. Macro-model of Spin-Transfer Torque based Magnetic Tunnel Junction device for hybrid Magnetic-CMOS design. In *IEEE International Behavioral Modeling and Simulation Workshop*, pages 40–43, 2006.