Chapter 6
An Energy-Efficient 3D Stacked STT-RAM Cache Architecture for CMPs

Guangyu Sun, Xiangyu Dong, Yiran Chen and Yuan Xie

Abstract In this chapter, we introduce how to adopt spin-transfer torque random access memory (STT-RAM) as on-chip L2 caches to achieve better performance and lower energy consumption, compared to traditional L2 cache designs. STT-RAM is a promising memory technology for on-chip cache design because of its fast read access, high density, and non-volatility. Using 3D heterogeneous integrations, it becomes feasible and cost-efficient to stack STT-RAM atop conventional chip multiprocessors (CMPs). However, one disadvantage of STT-RAM is its long write latency and its high write energy. In this chapter, we first stack STT-RAM-based L2 caches directly atop CMPs and compare it against SRAM counterparts in terms of performance and energy. We observe that the direct STT-RAM stacking might harm the chip performance due to the aforementioned long write latency and high write energy. To solve this problem, we then propose two architectural techniques: *read-preemptive write buffer* and *SRAM–STT-RAM hybrid L2 cache*. The simulation result shows that our optimized STT-RAM L2 cache improves performance by 4.91 % and reduces power by 73.5 % compared to the conventional SRAM L2 cache with the similar area.

G. Sun
Peking University, Beijing, China
e-mail: gsun@pku.edu.cn

X. Dong
Qualcomm Research Lab, San Diego, USA
e-mail: rioshering@gmail.com

Y. Chen
University of Pittsburgh, Pittsburgh, USA
e-mail: yic52@pitt.edu

Y. Xie (✉)
Pennsylvania State University, University Park, PA, USA
e-mail: yuanxie@cse.psu.edu

Y. Xie (ed.), *Emerging Memory Technologies*,
DOI: 10.1007/978-1-4419-9551-3_6,

6.1 Introduction

The diminishing return of endeavors to increase clock frequencies and exploit instruction-level parallelism in a single processor has led to the advent of chip multiprocessors (CMPs) [5]. The integration of multiple cores on a single chip is expected to accentuate the already daunting "memory wall" problem [3], and it becomes a major challenge of supplying massive multicore chips with sufficient memories.

The introduction of the three-dimensional (3D) integration technology [6, 35] provides the opportunity of stacking memories atop compute cores and therefore alleviates the memory bandwidth challenge of CMPs. Recently, active research [1, 10, 24] has targeted SRAM caches or DRAM memories stacking.

Spin-transfer torque random access memory (STT-RAM) is a promising memory technology with attractive features such as fast read access, high density, and non-volatility [11, 36]. How to integrate STT-RAM into compute cores on planular chips is the key obstacle since the STT-RAM fabrication involves hybrid magnetic-CMOS processes. Fortunately, 3D integrations enable the cost-efficient integration of heterogeneous technologies, which is ideal for STT-RAM stacking atop compute cores. Some recent work [7, 9] has evaluated the benefits of STT-RAM as a universal memory replacement for L2 caches and main memories in single-core chips.

In this chapter, we further evaluate the benefits of stacking STT-RAM L2 caches atop CMPs. We first develop a cache model for stacking STT-RAM and then compare the STT-RAM-based L2 cache against its SRAM counterpart with the similar area in terms of performance and energy. The comparison shows that (1) for applications that have moderate write intensities to L2 caches, the STT-RAM-based cache can reduce the total cache power significantly because of its zero standby leakage and achieve considerable performance improvement because of its relatively larger cache capacity and (2) for applications that have high write intensities to L2 caches, the STT-RAM-based cache can cause performance and power degradations due to the long latency and the high energy of STT-RAM write operations.

These two observations imply that STT-RAM-based caches might not work efficiently if we directly introduce them into the traditional CMP architecture because of their disadvantages on write latency and write energy. In light of this concern, we propose two architectural techniques, *read-preemptive write buffer* and *SRAM–STT-RAM hybrid L2 cache*, to mitigate the STT-RAM write-associated issues. The simulation result shows that performance improvement and power reduction can be achieved effectively with our proposed techniques even under the write-intensive workloads.

6.2 Related Work

Some previous research focused on the performance improvement by stacking DRAM main memories on top of processors. 3D cache model has been developed to facilitate architectural-level analysis [15, 34]. Performance analysis of 3D stacking

memory was studied by Loi et al. [23]. Li et al. [22] have also reported performance improvement by using stacked SRAM L2 caches for CMPs. Black et al. studied the benefits of stacking a large DRAM or SRAM cache on a Intel Core 2 Duo processor and achieved considerable performance improvement [1]. Loh [24–26] presented an aggressive 3D DRAM integration as on-chip caches and main memories. Kgil et al. [17] have implemented an aggressive CMP method by replacing all the L2 caches with in-order simple processor cores and used 3D stacking DRAM to satisfy the memory capacity and bandwidth requirements. Ghosh et al. [10] proposed a new method to reduce the power consumption in systems where the DRAM is stacked on top of the processor cores. A prototype of the 80-core teraflop processor with an SRAM layer stacked on top of the processor cores, which was designed and fabricated by Intel, also demonstrated the benefits of stacking SRAM memories on CMPs [2].

In order to overcome the memory wall, extensive research has been done to find alternatives of the traditional SRAM and DRAM technologies. Various emerging memory technologies, such as STT-RAM, phase change memory (PCM), floating body DRAM (FBDRAM), have been proposed to replace SRAM/DRAM in different levels of the memory hierarchy [21, 30, 32, 33, 37]. STT-RAM is normally employed as a competitive replacement of SRAM as on-chip memories because of its advantages of fast read speed, low leakage power, and high density [30, 33]. PCM is widely studied as a potential candidate of the main memory because it has higher density and lower standby power compared to DRAM [21, 32, 37]. As an emerging memory compatible with CMOS technology, FBDRAM is also attracting more attention, recently [19, 28]. With these emerging memory technologies, prior research has shown improvements in performance, power consumption, and reliability.

These emerging memories, however, have some common limitations such as long write latency and high write energy. Although the memory capacity is increased after using these high-density memories, the long write latency may offset the benefits and degrade the performance for workloads with intensive write operations. Similarly, although the standby power is reduced by using these emerging memories, the dynamic power can be significantly increased due to high write energy. Some research has been done to hide the long write latency by stalling the write and preempting the blocked read [31]. The replacement policy can also be tailored to reduce the write intensity [38]. Recently, Smullen et al. proposed to scarify the non-volatility to improve the write speed [33].

6.3 Background

This section briefly introduces the background of STT-RAM and 3D integration technologies.

6.3.1 STT-RAM Background

The basic difference between the STT-RAM and the conventional RAM technologies (such as SRAM/DRAM) is that the information carrier of STT-RAM is magnetic tunnel junctions (MTJs) instead of electric charges [36]. As shown in Fig. 6.1, each MTJ contains *a pinned layer* and *a free layer*. The *pinned layer* has fixed magnetic direction, while the *free layer* can change its magnetic direction by spin torque transfers [11]. If the free layer has the same direction as the pinned layer, the MTJ resistance is low and indicates state "0"; otherwise, the MTJ resistance is high and indicates state "1".

The latest STT-RAM technology (spin torque transfer ram, STT-RAM) changes the magnetic direction of the free layer by directly passing spin-polarized currents through MTJs. Compared to the previous generation of MRAM using external magnetic fields to reverse the MTJ status, STT-RAM has the advantage of scalability, which means that the *threshold current* to make the status reversal will decrease as the size of the MTJ becomes smaller.

The most popular structure of STT-RAM cells is composed of one NMOS transistor as the access device and one MTJ as the storage element ("1T1J" structure) [11]. As illustrated in Fig. 6.1, 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 STT-RAM cell are described as follows:

- **Read Operation**: When a *read operation* happens, the NMOS is turned on and a small voltage difference (-0.1 V as demonstrated in [11]) is applied between the bit line (BL) and the source line (SL). This voltage difference causes a current through the MTJ whose value is determined by the status of MTJs. A sense amplifier compares this current to a reference current and then decides whether a "0" or a "1" is stored in the selected STT-RAM cell.
- **Write Operation**: When a *write operation* happens, a large positive voltage difference is established between SLs and BLs for writing "0"s or a large negative one for writing "1"s. The current amplitude required to ensure a successful status

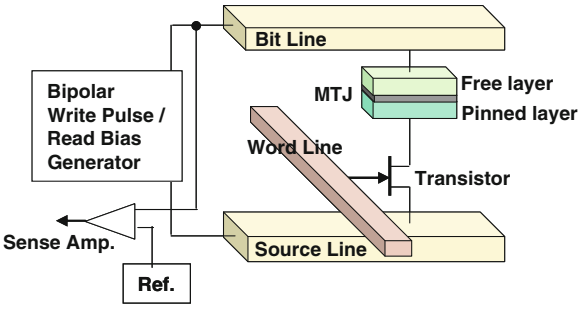

Fig. 6.1 An illustration of one STT-RAM cell

reversal is called threshold current. The current is related to the material of the tunnel barrier layer, the writing pulse duration, and the MTJ geometry [8].

In this work, we use the writing pulse duration of 10 ns [36], below which the writing threshold current will increase exponentially. In addition, we scale the STT-RAM size of previous work [11] down to 65 nm technology node. Assuming the size of MTJs is 65 × 90 nm, the derived threshold current for magnetic reversal is about 195 μA.

6.3.2 3D Integration Overview

The 3D integration technology has recently emerged as a promising means to mitigate interconnect-related problems. By using the vertical through silicon via (TSV), multiple active device layers can be stacked together (through wafer stacking or die stacking) in the third dimension [35].

3D integrations offer a number of advantages over traditional two-dimensional (2D) designs [6]: (1) shorter global interconnects because the vertical distance (or the length of TSVs) between two layers is usually in the range of 10–100 μm [35], depending on manufacturing processes; (2) higher performance because of reducing the average interconnect length; (3) lower interconnect power consumption due to the wire length reduction; (4) denser form factor and smaller footprint; and (5) support for the cost-efficient integration of heterogenous technologies.

In this chapter, we rely on the 3D integration technology to stack a massive amount of L2 caches (2 MB for SRAM caches and 8 MB for STT-RAM caches) on top of CMPs. Furthermore, the heterogenous technology integration enabled by 3D makes it feasible to fabricate STT-RAM caches and CMP logics as two separate dies and then stack them together in a vertical way. Therefore, the magnetic-related fabrication process of STT-RAM will not affect the normal CMOS logic fabrication and keep the integration cost-efficient.

6.4 STT-RAM and Non-uniform Cache Access Models

In this section, we describe an STT-RAM circuit model and a non-uniform cache access (NUCA) model which is implemented with network on chip (NoC).

6.4.1 STT-RAM Modeling

To model STT-RAM, we first estimate the area of STT-RAM cells. As shown in Fig. 6.1, each STT-RAM cell is composed of one NMOS transistor and one MTJ. The size of MTJs is only limited by manufacturing techniques, but the NMOS transistor

Table 6.1 STT-RAM cell specifications

Technology	65 nm
Write pulse duration	10 ns
Threshold current	195 μA
Cell size	$40F^2$
Aspect ratio	2.5

Table 6.2 Comparison of area, access time, and energy consumption (65 nm technology)

Cache size	128 KB SRAM	512 KB STT-RAM
Area	3.62 mm^2	3.30 mm^2
Read latency	2.252 ns	2.318 ns
Write latency	2.264 ns	11.024 ns
Read energy	0.895 nJ	0.858 nJ
Write energy	0.797 nJ	4.997 nJ

has to be sized properly so that it can drive sufficiently large current to change the MTJ status. The current driving ability of NMOS transistor is proportional to its W/L ratio. Using HSPICE simulation, we find that the minimum W/L ratio for the NMOS transistor under 65 nm technology node is around 10 to drive the threshold writing current of 195 μA. We further assume the width of the source or drain regions of an NMOS transistor is $1.5F$, where F is the feature size. Therefore, we estimate the STT-RAM cell size is about $10F \times 4F = 40F^2$. The parameters of our targeted STT-RAM cell are tabulated in Table 6.1.

Despite the difference in storage mechanisms, STT-RAM and SRAM have the similar peripheral interfaces from the circuit designers' point of view. By simulating with a modified version of CACTI [13], our result shows that the area of a 512 KB STT-RAM cache is similar to a 128 KB SRAM cache whose cell is about $146F^2$ (this value is extracted from CACTI). Table 6.2 lists the comparison between a 512 KB *STT-RAM cache bank* and a 128 KB *SRAM cache bank*, which are used later in this chapter, in terms of area, access time, and access energy.

6.4.2 Modeling 3D NUCA Cache

As the caches' capacity and area increase, the wire delay has made the NUCA architecture [18] more attractive than the conventional uniform cache access (UCA) one. In NUCA, the cache is divided into multiple banks with different access latencies according to their locations relative to cores and these banks can be connected through a mesh-based NoC.

Extending the work of CACTI [13], we develop our NoC-based 3D NUCA model. The key concept is to use NoC routers for communications within planular layers, while using a specific through silicon bus (TSB) for communications among different layers. Figure 6.3a illustrates an example of the 3D NUCA structure. There are four cores located in the *core layer* and 32 cache banks in each *cache layer*, and all

layers are connected by TSB, which is implemented with TSVs. This interconnect style has the advantage of short connections provided by 3D integrations. It has been reported that the vertical latency of traversing a 20-layer stack is only 12 ps [27], thus the latency of TSB negligible compared to the latency of 2D NoC routers. Consequently, it is feasible to have single-hop vertical communications by utilizing TSBs. In addition, hybridization of 2D NoC routers with TSBs requires one (instead of two) additional link on each NoC router, because TSB can move data both upward and downward [22].

As shown in Fig. 6.3a, cache layers are on top of core layers and they can either be SRAM or STT-RAM caches. Figure 6.3b shows a detailed 2D structure of cache layers. Every four cache banks are grouped together and routed to other layers via TSBs.

Similar to prior approaches [4, 22], the proposed model supports *data migration*, which moves data closer to their accessing core. For set-associative cache, the cache ways belonging to the set should be distributed into different banks so that data migration can be implemented. In our 3D NUCA model, each cache layer is equally divided into several zones. The number of zones is equal to the number of cores, and each zone has a TSB located at its center. The cache ways of each set are uniformly distributed into these zones. This architecture promises that, within each cache set, there are several ways of cache lines close to the active core. Figure 6.2 gives an illustration of distributing eight ways into four zones. Figure 6.3a shows an example of data migration after which the core in the upper left corner can access the data faster. In this chapter, this kind of data migrations is called *inter-migration* to differentiate another kind of migration policy introduced later.

The advantages of this 3D NUCA cache are as follows: (1) Placing L2 caches in separate layers makes it possible to integrate STT-RAM with traditional CMOS process technology and (2) separating cores from caches simplifies the design of TSBs and routers because TSBs are now connected to cache controllers directly, and there is no direct connection between routers and cache controllers.

We provide one TSB for each core in the model. Considering that the TSV pitch size is reported to be only 4–10 μm [27]; thus, even a 1024-bit bus (much wider than our proposed TSB) would only incur an area overhead of 0.32 mm^2. In our study, the die area of an 8-core CMP is estimated to be 60 mm^2 (discussed later). Therefore, it is feasible to assign one TSB for each core and the TSV area overhead is negligible.

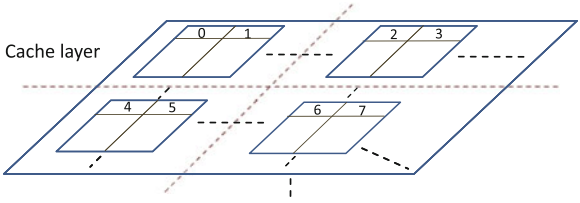

Fig. 6.2 Eight cache ways are distributed in four banks. Assume four cores and accordingly four zones each layer

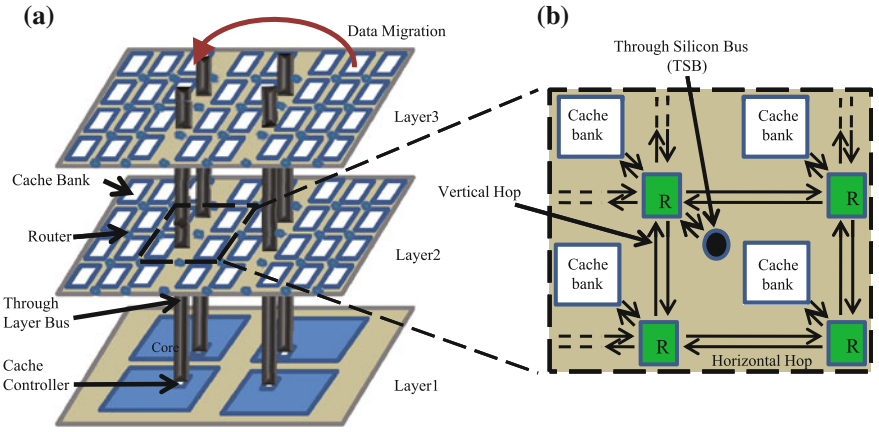

Fig. 6.3 **a** An illustration of the proposed 3D NUCA structure, which includes 1 core layer, 2 cache layers. There are 4 processing cores per core layer, 32 cache banks per cache layer, and 4 through-layer-bus across layers. **b** Connections among routers, caches banks, and through-layer-buses

6.4.3 Configurations and Assumptions

Our baseline configuration is an 8-core in-order processor using the UltraSPARC III ISA. In order to predict the chip area, we investigate some die photos, such as cell processor [16], Sun UltraSPARC T1 [20], etc., and estimate the area of an 8-core CMP without caches to be 60 mm^2. By using our modified version of CACTI [13], we further learn that one cache layer fits to either a 2 MB SRAM or an 8 MB STT-RAM L2 cache, assuming that each cache layer has the similar area to that of core layer (60 mm^2). The configurations are detailed in Table 6.3. Note that the power of processors is estimated based on the data sheet of real designs [16, 20].

We use the Simics toolset [29] for performance simulations. Our 3D NUCA architecture is implemented as an extended module in Simics. We use a few multithreaded benchmarks from *OpenMP2001* [14] and *PARSEC* [12] suites. Since the performance and power of STT-RAM caches are closely related to transaction intensity, we select some simulation workloads as listed in Table 6.4 so that we have a wide range of transaction intensities to L2 caches. The average numbers of total transactions (TPKI)[1] and write transactions (WPKI) of L2 caches are listed in Table 6.4. For each simulation, we fast forward to warm up the caches and then run 3 billion cycles. We use the total IPC of all the cores as the performance metric.

[1] TPKI is the number of total transactions per 1K instructions, and WPKI is the number of write transactions per 1K instructions.

Table 6.3 Baseline configuration parameters

Processors	
# of cores	8
Frequency	3 GHz
Power	6 W/core
Issue width	1 (in order)
Memory parameters	
L1 cache	Private, 16 + 16 KB, 2-way, 64 B line, 2-cycle, write-through, 1 read/write port
SRAM L2	Shared, 2 MB (16 × 128 KB), 32-way, 64 B line, read/write per bank: 7-cycle, write-back, 1 read/write port
STT-RAM L2	Shared, 8 MB (16 × 512 KB), 32-way, 64 B line, read penalty per bank: 7-cycle, write penalty per bank: 33-cycle, write-back, 1 read/write port
Write buffer	4 entry, retire-at-2
Main memory	4 GB, 500-cycle latency
Network parameters	
# of layers	2
# of TSB	8
Hop latency	TSB 1 cycle, V_hop 1 cycle H_hop 1 cycle
Router latency	2-cycle

Table 6.4 L2 transaction intensities

Name	TPKI	WPKI
Galgel	1.01	0.31
Apsi	4.15	1.85
Equake	7.94	3.84
Fma3d	8.43	4.00
Swim	19.29	9.76
Streamcluster	55.12	23.326

6.4.4 SNUCA and DNUCA

Static NUCA (SNUCA) and dynamic NUCA (DNUCA) are two different implementations of the NUCA architecture proposed by Kim et al. [18]. SNUCA statically partitions the address space across cache banks, which are connected via NoC; DNUCA dynamically migrates frequently accessed blocks to the closest banks. These two NUCA implementations result in different access patterns and variable write intensities. In our later simulations, we use both SNUCA-SRAM and DNUCA-SRAM L2 caches as our baselines when evaluating the performance and power benefits of STT-RAM caches.

6.5 Direct Replacing SRAM with STT-RAM as L2 Caches

In this section, we directly replace SRAM L2 caches with STT-RAM ones that have the comparable area and show that without any optimization, a naive STT-RAM replacement will harm both performance and power when the workload write intensity is high.

6.5.1 Same Area Replacement

As shown in Table 6.2, a 128 KB SRAM bank has the similar area as a 512 KB STT-RAM bank does. Thereby, in order to keep the area of cache layers unchanged, it becomes reasonable to replace SRAM L2 caches with STT-RAM ones whose capacity is three times larger. We call this replacement strategy as "same area replacement."

Using this strategy, we integrate as many caches in the cache layers as possible. Considering our baseline SRAM L2 cache has 16 banks and each cache bank has the capacity of 128 KB, we keep the number of banks unchanged but replace each 128 KB SRAM L2 cache bank with a 512 KB STT-RAM cache bank. The read/write access time and read/write energy consumption are tabulated in Table 6.2 for both SRAM and STT-RAM.

6.5.2 Performance Analysis

Because the number of banks remains the same and our modified CACTI shows that 128 KB SRAM bank and 512 KB STT-RAM bank have similar read latencies (2.252 vs. 2.318 ns in Table 6.2), the read latencies of the 2 MB SRAM cache and the 8 MB STT-RAM cache are similar as well. Since the STT-RAM cache capacity is three times larger, the access miss rate to the L2 cache decreases as shown in Fig. 6.4. On average, the miss rates are reduced by 19.0 % and 12.5 % for SNUCA STT-RAM cache and DNUCA STT-RAM cache, respectively.

The IPC comparison is illustrated in Fig. 6.5. Caused by the large STT-RAM cache capacity, the L2 cache miss rate decrease improves the performance of the first two workloads ("galgel" and "apsi"); however, the performance of the rest four workloads is not improved as expected. On average, the performance *degradation* of *SNUCA STT-RAM* and *DNUCA STT-RAM* is 3.09 % and 7.52 %, respectively, compared to their SRAM counterparts.

This performance degradation of direct STT-RAM replacement can be explained by Table 6.4, where we can observe the write operation intensity (presented by WPKI) of "equake," "fma3d," "swim," and "streamcluster" is much higher than that of "galgel" and "apsi." Due to the long latency of STT-RAM write operations, the high write intensity is reflected by the performance loss. When the write intensity is sufficiently high, the resulting performance loss overwhelms the performance

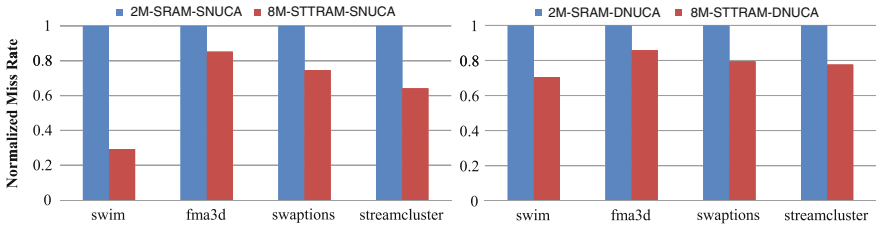

Fig. 6.4 The comparison of L2 caches access miss rates for SRAM L2 cache and STT-RAM L2 cache that have similar area. Larger capacity of STT-RAM cache results in smaller cache miss rates

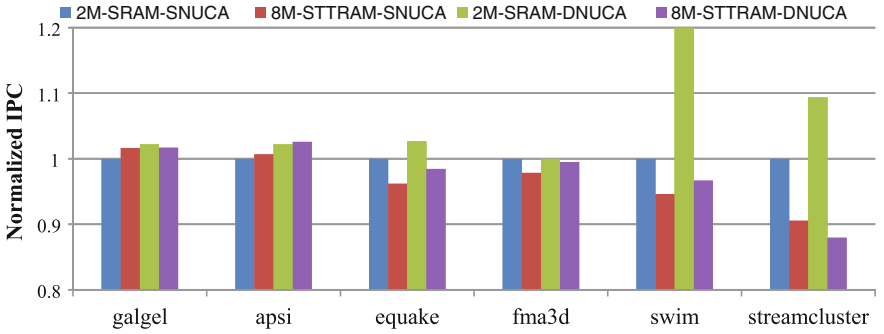

Fig. 6.5 IPC comparison of SRAM and STT-RAM L2 caches (normalized by 2M SNUCA-SRAM cache)

gain achieved by reduced L2 cache miss rate. This observation is further supported by comparison between SNUCA and DNUCA. From Fig. 6.5, one can observe that performance degradation is more significant when we use DNUCA STT-RAM caches because data migrations in DNUCA initiate more write operations than SNUCA does and thus cause high write intensities.

To summarize, we conclude our first observation of using STT-RAM caches as

Observation 1 *Replacing SRAM L2 caches directly with STT-RAM, which has the similar area but with a large capacity, can reduce the access miss rate of the L2 cache. However, the long latency associated with the write operations to the STT-RAM cache has a negative impact on the performance. When the write intensity is high, the benefits caused by miss rate reductions could be offset by the long latency of STT-RAM write operations and eventually result in performance degradation.*

6.5.3 Power Analysis

The major contributors of the total power consumption in caches are leakage power and dynamic power:

- *Leakage Power*: When process technology scales down to sub-90 nm, the leakage power in CMOS technology becomes dominant. Since STT-RAM is a non-volatile memory technology, there is no power supply to each STT-RAM cell, and then, STT-RAM cells do not consume any standby leakage power. Therefore, we only consider peripheral circuit leakage power for STT-RAM caches and the leakage power comparison of 2 MB SRAM and 8 MB STT-RAM is listed in Table 6.5.
- *Dynamic Power*: The dynamic power estimation for the NUCA cache is described as follows. For each transaction, the total dynamic power is composed of the memory cell access power, the router access power, and the power consumed by wire connections. In this chapter, these values are either simulated by HSPICE or obtained from our modified version of CACTI. The access number of routers and the length of wire connections vary from the location of the requesting core and the requested cache lines.

Figure 6.6 shows the power comparison of SRAM and STT-RAM L2 caches. One can observe that

- For SRAM L2 caches, since the leakage power dominates, the total power for SNUCA-SRAM and DNUCA-SRAM is very close. On the contrary, the dynamic power dominates the STT-RAM cache power.
- For all the workloads, STT-RAM caches consume less power than SRAM caches do. The average power savings across all the workloads are about 78 and 68 % for SNUCA and DNUCA, respectively. The power saving for DNUCA STT-RAM is smaller because of the high write intensity caused by data migrations. It is obvious that the "low leakage power" feature makes STT-RAM more attractive to be used

Table 6.5 Leakage power of SRAM and STT-RAM caches at 80 °C

Cache configurations	Leakage power (W)
2 MB 16 × 128 KB SRAM cache	2.089
8 MB 16 × 512 KB STT-RAM cache	0.255

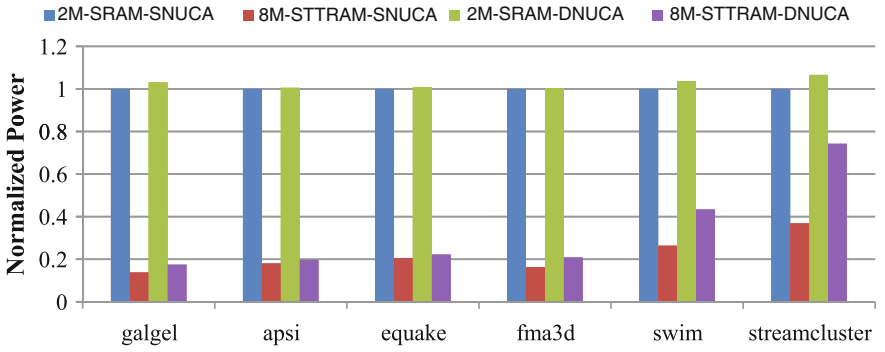

Fig. 6.6 Power comparison of SRAM and STT-RAM L2 caches (normalized by 2 MB SNUCA-SRAM cache)

as large on-chip memory, especially when SRAM leakage power becomes worse with technology scaling.
- The average power savings for the first four workloads are more than 80 %. However, for the workload "streamcluster," the total power saving is only 63 and 30 % for SNUCA and DNUCA, respectively, due to its much higher L2 cache write intensity (see Table 6.4).

To summarize, our second conclusion of direct STT-RAM cache replacement is

Observation 2 *Direct replacing of the SRAM L2 cache with a STT-RAM cache, which has similar area but with larger capacity, can greatly reduce the leakage power. However, when the write intensity is high, the dynamic power increases significantly because of the high energy associated with the STT-RAM write operation and the amount of total power saving could be reduced.*

These two conclusions show that if we directly replace SRAM caches with STT-RAM caches using "same area strategy," the long latency and high energy consumption of STT-RAM write operations can offset the performance and power benefit brought by STT-RAM cache when the cache write intensity is high.

6.6 Novel 3D-Stacked Cache Architecture

In this section, we propose two techniques to mitigate the write operation problem of using STT-RAM caches: *Read-preemptive write buffer* is employed to reduce the stall time caused by the STT-RAM long write latency; *SRAM–STT-RAM hybrid L2 cache* is proposed to reduce the number of STT-RAM write operations and thereby improve both performance and power. Finally, we combine these two techniques together as an optimized STT-RAM cache architecture.

6.6.1 Read-Preemptive Write Buffer

The first observation in Sect. 6.5 shows that the long STT-RAM write latency has a serious impact on the performance. In the scenario where a write operation is followed by several read operations, the ongoing write operation may block the upcoming read operations and cause performance degradations. Although the write buffer design in modern processors works well for SRAM caches, our experiment result in Sect. 6.5.2 shows that this write buffer does not fit for STT-RAM caches due to the large variation between STT-RAM read latency and write latency. In order to make STT-RAM caches work efficiently, we explore the proper write buffer size and propose a "read-preemptive" management policy for it.

6.6.1.1 The Exploration of the Buffer Size

The choice of the buffer size is important. The larger the buffer size is, the more write operations can be hidden. Thereby, the number of stall cycles decreases. However, on the other hand, the larger the buffer size is, the longer time it takes to check whether there is a "hit" in the buffer and then to access it. Furthermore, the design complexity and the area overhead also increase with the buffer size growth. Figure 6.7 shows the relative IPC improvement by using different buffer sizes for workloads "streamcluster" and "swim." Observing the simulation result, we choose the size of 20 entries as the optimal STT-RAM write buffer size. Compared to the SRAM write buffer, which has only 4 entries (as listed in Table 6.3), the STT-RAM write buffer size is much larger and we use 20-entry write buffer for STT-RAM caches in the later simulations.

6.6.1.2 Read-Preemptive Policy

Since the L2 cache can receive requests from the upper level memory (L1 cache) and the write buffer, a priority policy is necessary to solve the conflict that a read request and a write request compete for the execution right. For STT-RAM caches, write operation latencies are much larger than read latencies; thus, our objective is to prevent write operations from blocking read operations. As a result, we have our first rule:

Rule 1 *The read operation always has the higher priority in a competition for the execution right.*

Additionally, consider there is a read request blocked by a write operation that is already in process, the STT-RAM write latency is so large that its retirement may block one or more read request for a long period and further cause performance degradations. In order to mitigate this problem, we propose another read-preemptive rule as follows:

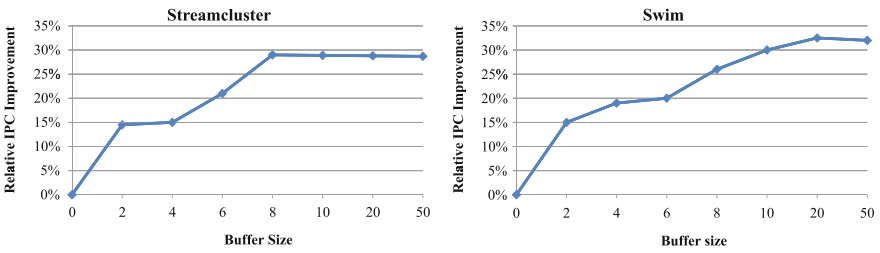

Fig. 6.7 The impact of buffer size. The IPC improvement is normalized by that of 8M STT-RAM cache without write buffer

Rule 2 *When a read request is blocked by a write retirement and the write buffer is not full, the read request can trap and stall the write retirement if the preemption condition (discussed later) is satisfied. Then, the read operation obtains the right of the execution to the cache. The stalled write retirement will retry later.*

Our proposed read-preemptive policy tries to execute STT-RAM read requests as early as possible, but the drawback is that some write retirements need to be re-executed and the possibility of full buffer increases. The pivot is to find a proper *preemption condition*. One extreme method is to stall the write retirement as long as there is a read request, which means that read requests can always be executed immediately. Theoretically, if the write buffer size is large enough, no read request will be blocked. However, since the buffer size is limited, the increased possibility of full buffer could also harm the performance. In some other cases, stalling write retirements for read requests is not always good. For example, if a write retirement almost finishes, no read request should stall the retirement process. Consequently, we propose to use the *retirement accomplishment degree*, denoted as α, as the preemption condition. The *retirement accomplishment degree* is the accomplishment percentage of the ongoing write retirement, below which no preemption will occur.

Figure 6.8 compares the IPC of using different α in our read-preemptive policy. Note that $\alpha = 0\,\%$ represents the non-conditional preemption policy and $\alpha = 100\,\%$ represents the traditional write buffer. We can find that for the workloads with low write intensities, such as "galgel" and "apsi," the performance improves as α increases and the non-conditional preemption policy works the best. However, for the benchmark with high write intensities, like "streamcluster," the performance improves at the beginning but then degrades as α increases. Generally, in this chapter, we set $\alpha = 50\,\%$ to make our read-preemptive policy effective for all the workloads.

A counter is required in order to make the accomplishment degree aware of the cache controller. The counter resets to zero and begins to count the number of cycles when a retirement begins. The cache controller checks the counter and

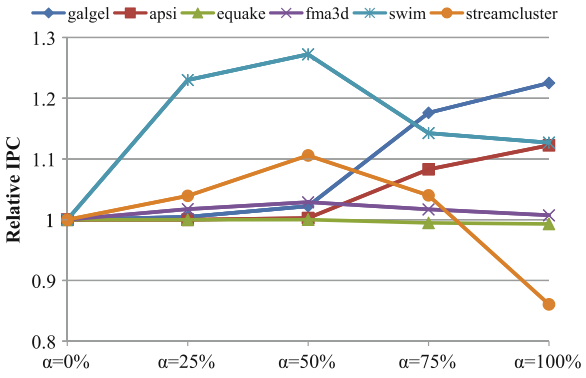

Fig. 6.8 The impact of α on the performance. The IPC values are normalized by that of using the traditional policy

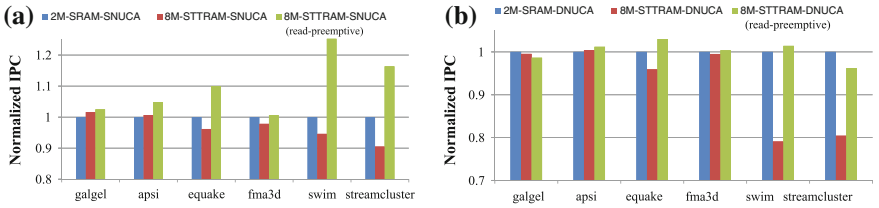

Fig. 6.9 The comparison of IPC among 2M SRAM, 8M STT-RAM with traditional write buffer, and 8M STT-RAM with read-preemptive write buffer (normalized by that of SRAM)

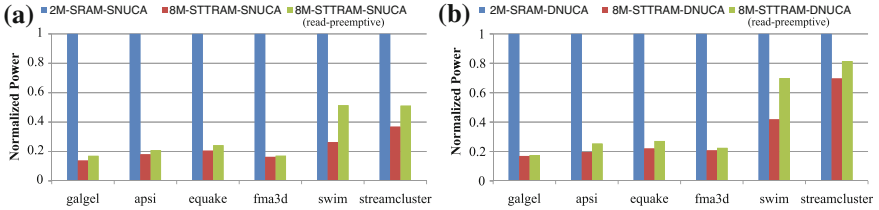

Fig. 6.10 The comparison of total power of 2M SRAM, 8M STT-RAM with traditional write buffer, and 8M STT-RAM with read-preemptive write buffer (normalized by that of SRAM)

decides whether to stall the retirement for the read request. The area of 20 buffer entries can be evaluated as a cache whose size is 20×64 byte (less than 2 KB). We use a 7-bit counter to record the retirement accomplishment degree. Since the area of each 3D-stacked layer is around 60 mm^2, the area overhead of our proposed *read-preemptive write buffer* is less than 1 %. Similarly, the leakage power increase caused by this buffer is also negligible.

Figures 6.9 and 6.10 illustrates the performance and power improvement gained by our proposed *read-preemptive write buffer*. Compared to the IPC of SRAM baseline configurations, the average performance improvements are 9.93 % and 0.41 % for SNUCA and DNUCA, respectively. The average power reductions are 67.26 % and 59.3 % for SNUCA and DNUCA, respectively. Compared to the result of direct STT-RAM replacement shown in Figs. 6.5 and 6.6, the performance degradation is eliminated, but the amount of power savings decrease. It is because using our read-preemptive write buffer causes some re-executions of write operations which consume more power.

6.6.2 SRAM–STT-RAM Hybrid L2 Cache

The aforementioned read-preemptive write buffer hides the STT-RAM long write latency, but the total number of write operations remains the same. In order to reduce the number of write operations to STT-RAM cells, we propose another technique called *SRAM–STT-RAM Hybrid Cache* and show how this technique can further reduce the dynamic power as well as improve the performance.

6.6.2.1 SRAM–STT-RAM Hybrid Cache Implementation

The proposed hybrid cache implementation is that, instead of building a pure STT-RAM cache, we compose the ways in each cache set with a majority of STT-RAM cache lines and a minority of SRAM ones. The main purpose is to keep as many write-intensive data in the SRAM part as possible and hence reduce the number of write operations to the STT-RAM part. In this work, we design an SRAM–STT-RAM hybrid L2 cache with 31 ways of STT-RAM and 1 way of SRAM (*31M1S*).

After having these hybrid cache lines, the second step is to distribute STT-RAM cache lines and SRAM ones into separate cache banks. Considering the SRAM part is the minority in the proposed *31M1S* cache, one partitioning alternative is to distribute these SRAM cache lines into different banks so that there are several SRAM cache lines close to each processing core. However, this method requires each cache bank to be a heterogenous memory array with SRAM and STT-RAM cells and increases the complexity of the cache design. In addition, this distributed partitioning of SRAM cells implies that the SRAM and STT-RAM cells have to be fabricated together. Considering the specialization of the STT-RAM fabrication process, this method also eliminates the cost advantages of stacking STT-RAMs on top of processing cores.

Therefore, we use another alternative that we reduce the number of cache lines in some STT-RAM cache banks compared to the pure STT-RAM cache structure (as shown in Fig. 6.11a that the STT-RAM banks at four corners are smaller than other

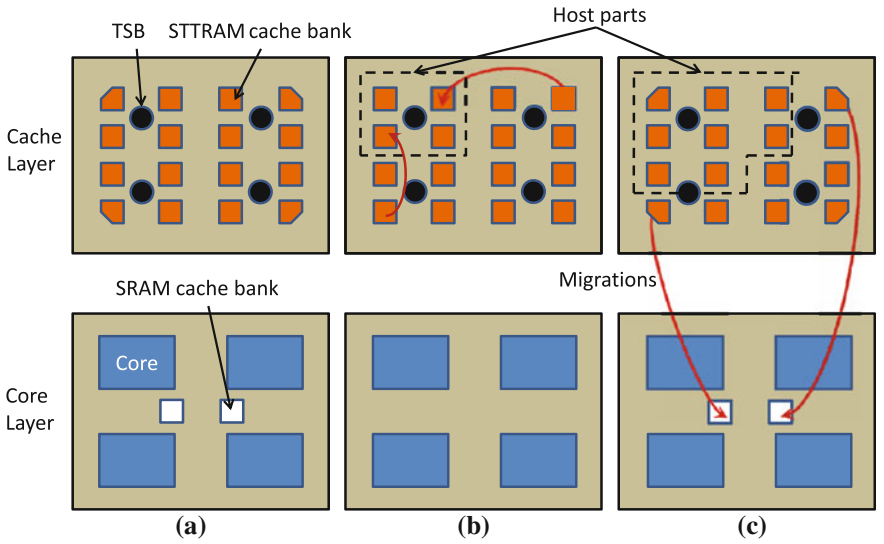

Fig. 6.11 SRAM–STT-RAM hybrid cache implementation **a** one placement method of SRAM and STT-RAM cache banks. **b** data migrations in original STT-RAM caches. **c** data migrations in hybrid SRAM–STT-RAM caches

STT-RAM banks), compensate this cache line loss with SRAM ones, and collect all the SRAM cache lines together to build several entire SRAM banks on the core layer. As shown in Fig. 6.11a, SRAM cache banks are placed in the center of the core layer instead of being distributed. In this method, SRAM and STT-RAM cache banks have no difference from the architectural point of view.

Note that after placing one way of SRAM cache lines in the core layer, the area of the core layer will increase and the area of the cache layer will decrease. In this work, the total size of all the SRAM cache lines is 256 KB, and the derived area overhead is about 12.5 %.

6.6.2.2 Hybrid Cache Management Policy

Another important issue is how to manage the hybrid L2 cache to improve the performance and reduce the power. Because the key point is to reduce the number of write operations to STT-RAM cache cells, we need to move as many write-intensive data in SRAM cache banks as possible. The management policy of the hybrid cache can be described as follows:

- The cache controller is aware of the locations of SRAM cache ways and STT-RAM cache ways. When there is a write miss, the cache controller first tries to place the data in the SRAM cache ways.
- Considering the high probability that a core write data to a specific group of cache lines repeatedly, data in STT-RAM caches should be migrated to SRAM caches if the some cache lines are frequently written to. In this work, data in STT-RAM caches will be migrated to SRAM caches when they are accessed by two successive write operations. This kind of data migration is named *intra-migration* to differentiate *inter-migration* policy introduced in Sect. 6.4. Due to the existence of this intra-migration policy, the number of write accesses from cores to STT-RAM caches can be reduced.
- Note that read operations from cores are also possible to cause data migrations, the number of which could be even larger than that of direct write accesses from cores. Therefore, a new type inter-migration policy is introduced. Figure 6.11b and c compares the banks from which data can be migrated toward the core in upper left corner. Figure 6.11b shows that in original inter-migration policy, the cache layer is divided into 4 uniform groups and there is only one core associative with each part. In this work, banks in each group are named as the *host banks* of their corresponding core. Data can only be migrated from *non-host banks*. For the traditional management policy, the data will be migrated to *host bank*. For the management policy proposed for the hybrid cache, the data can only be migrated to SRAM banks.

Two data migrations are illustrated in Fig. 6.11b for the traditional inter-migration. When using the hybrid SRAM–STT-RAM cache, the *host banks* for a core is redefined as shown in Fig. 6.11c. Two corresponding data migrations are also shown

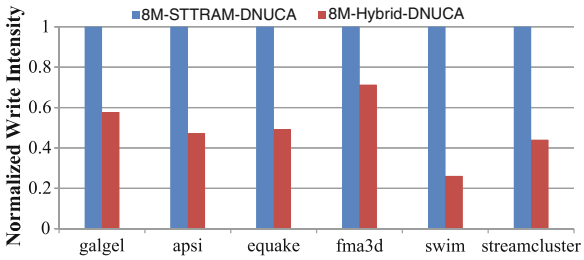

Fig. 6.12 The STT-RAM write intensity to STT-RAM before and after using hybrid SRAM–STT-RAM caches

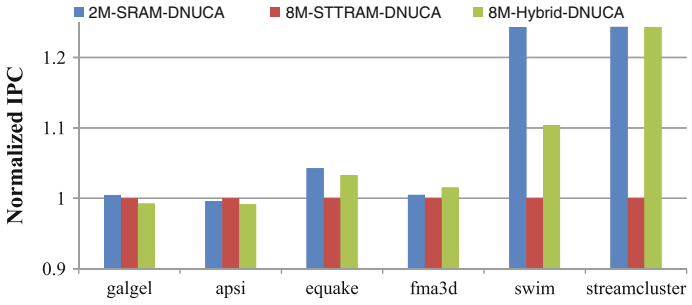

Fig. 6.13 The comparison of IPC among 2M SRAM cache, 8M STT-RAM pure cache, and 8M SRAM–STT-RAM hybrid cache (normalized by the IPC of 8M STT-RAM pure cache)

in Fig. 6.11c. Using this policy, there is no data migration between two STT-RAM cache lines, which reduces the number of write operations greatly. The drawback is that SRAM banks are shared by all cores so that their limited sizes may increase L2 miss rates. Considering we have 8M of total cache size, which is considerably large for most applications, our simulation results show that the increase in L2 miss rates is very small.

Figure 6.12 shows that the number of STT-RAM write operations per 1K instructions is reduced dramatically by using our hybrid SRAM–STT-RAM approach. As a result, the dynamic power associated with write operations to STT-RAM cells is also reduced and the performance is improved. Figure 6.13 shows the performance comparison. On average, the hybrid cache structure improves the performance by 5.65 %, which means that it mitigates the performance loss of STT-RAM caches from 8.48 to 2.61 % compared to their SRAM counterparts.

Figure 6.14 shows the power comparison. We observe that the total power is reduced except for "galgel." It is because both read and write intensities in "galgel" are so small that the dynamic power is very low. Consequently, the introduction of SRAM cache lines in the hybrid cache brings the leakage power back and eliminates the dynamic power reduction achieved by the hybrid structure. However, as the write

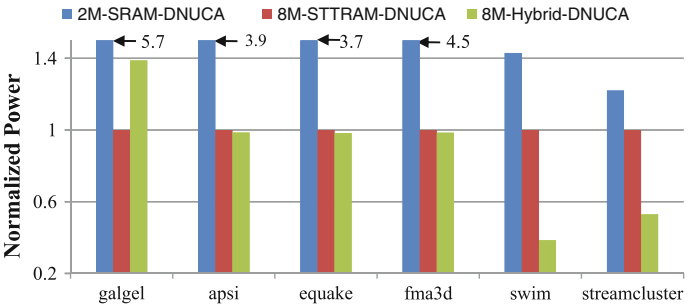

Fig. 6.14 The comparison of total power consumption among 2M SRAM cache, 8M STT-RAM pure cache, and 8M SRAM–STT-RAM hybrid cache (normalized by the total power consumption of 8M STT-RAM pure cache)

intensity increases, the SRAM–STT-RAM hybrid cache starts to save total power consumptions. For example, the total power consumption is cut by more than half for workloads such as "swim" and "streamcluster." On average, after the transition from SRAM caches to STT-RAM ones, our proposed hybrid cache further reduces the total power by 12.45 %.

6.6.3 Combination of Read-Preemptive Buffer and Hybrid Architecture

We combine the two techniques together as an optimized STT-RAM L2 cache architecture. In this architecture, we get more benefits from the advantages of the STT-RAM cache and, at the same time, mitigate the penalties caused by write operations. The performance and power comparisons are shown in Figs. 6.15 and 6.16, respectively. The average IPC is improved by 4.91 % compared to the SRAM SNUCA baseline, while power saving is 73.5 %. Table 6.6 gives an overview of the performance and power improvements.

6.7 Conclusion

STT-RAM is a promising candidate of on-chip memories, and the emerging 3D heterogeneous integration makes it feasible to stack STT-RAM as L2 caches for CMPs. In this work, we present a cache model for STT-RAM L2 cache stacking and evaluate its performance and power benefit. Even though replacing SRAM L2 cache with STT-RAM can result in significant power savings, the drawback comes from STT-RAM's long write latency and high write energy. As a result, for applications with high L2

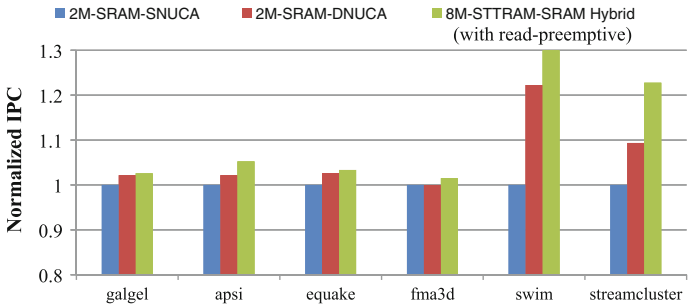

Fig. 6.15 The comparison of IPC among 2 MB SRAM SNUCA cache, 2 MB SRAM DNUCA cache, and 8 MB SRAM–STT-RAM hybrid cache with read-preemptive write buffer (normalized by the IPC of 2 MB SRAM SNUCA cache)

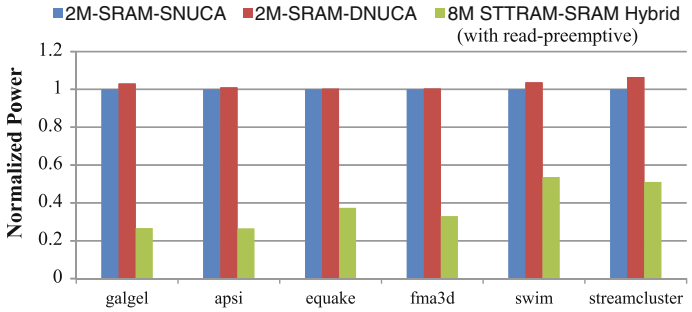

Fig. 6.16 The comparison of total power consumption among 2 MB SRAM cache, 2 MB SRAM cache, and 8 MB SRAM–STT-RAM hybrid cache with read-preemptive write buffer (normalized by the total power consumption of 2 MB SRAM cache)

Table 6.6 The performance and power improvement overview (use 2 MB SRAM L2 SNUCA cache as the baseline)

	Performance (%)	Total power (%)
Read-preemptive buffer	9.93	67.26
Hybrid cache	−2.61	85.45
Combined	4.91	73.5

cache write intensities, the performance can be degraded and the power saving can be reduced. Therefore, we propose two techniques: read-preemptive write buffer to mitigate the performance penalty caused by the long write latency and SRAM–STT-RAM hybrid L2 cache to reduce the number of STT-RAM write operations. Our result shows that these two techniques can make STT-RAM cache work effective for most workloads, regardless of their write intensities.

References

1. Black, B., Annavaram, M., Brekelbaum, N., et al. (2006). Die stacking (3D) microarchitecture. In *MICRO 39: Proceedings of the 39th Annual IEEE/ACM International Symposium on Microarchitecture* (pp. 469–479).
2. Borkar, S. (2008). 3D technology: A system perspective. In *Technical Digest of the International 3D System Integration Conference* (pp. 1–14).
3. Burger, D., Goodman, J. R., & Kagi, A. (1997). Limited bandwidth to affect processor design. *Micro, IEEE*, *17*(6), 55–62.
4. Chishti, Z., Powell, M. D., & Vijaykumar, T. N. (2005). Optimizing replication, communication, and capacity allocation in CMPs. *SIGARCH Computer Architecture News*, *33*(2), 357–368.
5. Davis, J. D., Laudon, J., & Olukotun, K. (2005). Maximizing CMP throughput with mediocre cores. In *PACT '05: Proceedings of the 14th International Conference on Parallel Architectures and Compilation, Techniques* (pp. 51–62).
6. Davis, W. R., Wilson, J., Mick, S., et al. (2005). Demystifying 3D ICs: The pros and cons of going vertical. *IEEE Design and Test of Computers*, *22*(6), 498–510.
7. Desikan, R., Lefurgy, C. R., Keckler, S. W., & Burger, D. (2002). On-chip MRAM as a high-bandwidth low-latency replacement for DRAM physical memories. Technical report.
8. Diao, Z., Li, Z., Wang, S., et al. (2007). Spin-transfer torque switching in magnetic tunnel junctions and spin-transfer torque random access memory. *Journal of Physics: Condensed Matter, 19*(16), 165, 209 (13pp).
9. Dong, X., Wu, X., Sun, G., et al. (2008). Circuit and microarchitecture evaluation of 3D stacking magnetic RAM (MRAM) as a universal memory replacement. In *DAC '08: Proceedings of the 45th annual conference on Design automation* (pp. 554–559).
10. Ghosh, M., & Lee, H. H. S. (2007). Smart refresh: An enhanced memory controller design for reducing energy in conventional and 3D die-stacked DRAMs. In *MICRO '07: Proceedings of the 40th Annual IEEE/ACM International Symposium on Microarchitecture* (pp. 134–145).
11. Hosomi, M., Yamagishi, H., Yamamoto, T., et al. (2005). A novel non-volatile memory with spin torque transfer magnetization switching: Spin-RAM. In *International Electron Devices Meeting* (pp. 459–462).
12. http://parsec.cs.princeton.edu/
13. http://www.hpl.hp.com/research/cacti/
14. http://www.spec.org/
15. Jacob, P., Erdogan, O., Zia, A., et al. (2005). Predicting the performance of a 3D processor-memory chip stack. *IEEE Design and Test of Computers*, *22*(6), 540–547.
16. Kahle, J. A., Day, M. N., Hofstee, H. P., et al. (2005). Introduction to the cell multiprocessor. *IBM Journal of Research and Development*, *49*(4/5), 589–604.
17. Kgil, T., et al., D'Souza, S., Saidi, A., et al. (2006). PicoServer: Using 3D Stacking Technology to Enable a Compact Energy Efficient Chip Multiprocessor. *Proceedings of the 2006 ASPLOS Conference*, *41*(11), 117–128.
18. Kim, C., Burger, D., & Keckler, S. (2002). An adaptive, non-uniform cache structure for wire-delay dominated on-chip caches. In *Proceedings the 10th International Conference on Architectural Support for Programming Languages and Operating Systems*.
19. Kim, J., Chung, S., Jang, T., et al. (2010). Vertical double gate Z-RAM technology with remarkable low voltage operation for DRAM application (pp. 163–164).
20. Kongetira, P., Aingaran, K., & Olukotun, K. (2005). Niagara: A 32-way multithreaded SPARC processor. *IEEE Micro*, *25*(2), 21–29.
21. Lee, B. C., Ipek, E., Mutlu, O., & Burger, D. (2009). Architecting phase change memory as a scalable DRAM alternative. In *Proceedings of ISCA* (pp. 2–13).
22. Li, F., Nicopoulos, C., Richardson, T., et al. (2006). Design and management of 3D chip multiprocessors using network-in-memory. In *ISCA '06: Proceedings of the 33rd, Annual International Symposium on Computer Architecture* (pp. 130–141).
23. Liu, C. C., Ganusov, I., Burtscher, M., & Tiwari, S. (2005). Bridging the processor-memory performance gap with 3D IC technology. *IEEE Design and Test of Computers*, *22*(6), 556–564.

24. Loh, G. H. (2008). 3D-stacked memory architectures for multi-core processors. In *ISCA '08: Proceedings of the 35th International Symposium on Computer, Architecture* (pp. 453–464).
25. Loh, G. H., & Hill, M. D. (2011). Efficiently enabling conventional block sizes for very large die-stacked dram caches. In *MICRO'11* (pp. 454–464).
26. Loh, G. H., & Hill, M. D. (2012). Supporting very large dram caches with compound-access scheduling and missmap. *IEEE Micro* (pp. 70–78).
27. Loi, G. L., Agrawal, B., Srivastava, N., et al. (2006). A thermally-aware performance analysis of vertically integrated (3-D) processor-memory hierarchy. In *DAC '06: Proceedings of the 43rd Annual Conference on Design automation* (pp. 991–996).
28. Lu, Z., Collaert, N., Aoulaiche, M., De Wachter, B., De Keersgieter, A., Schwarzenbach, W., et al. (2010). A novel low-voltage biasing scheme for double gate fbc achieving 5s retention and 10_{16} endurance at 85c. In *IEDM* (pp. 12.3.1–12.3.4). doi:10.1109/IEDM.2010.5703347.
29. Magnusson, P. S., Christensson, M., Eskilson, J., et al. (2002). Simics: A full system simulation platform. *Computer*, *35*(2), 50–58.
30. Nigam, A., Smullen, C., Mohan, V., Chen, E., Gurumurthi, S., & Stan, M. (2011). Delivering on the promise of universal memory for spin-transfer torque ram (stt-ram). In *ISLPED 2011* (pp. 121–126). doi:10.1109/ISLPED.2011.5993623.
31. Qureshi, M., Franceschini, M., & Lastras-Montano, L. (2010). Improving read performance of phase change memories via write cancellation and write pausing. In *HPCA* (pp. 1–11). doi:10.1109/HPCA.2010.5416645.
32. Qureshi, M. K., Srinivasan, V., & Rivers, J. A. (2009). Scalable high performance main memory system using phase-change memory technology. In *Proceedings of ISCA* (pp. 24–33).
33. Smullen, C., Mohan, V., Nigam, A., Gurumurthi, S., & Stan, M. (2011). Relaxing non-volatility for fast and energy-efficient stt-ram caches. In *2011 IEEE 17th International Symposium on High Performance Computer Architecture (HPCA)* (pp. 50–61). doi:10.1109/HPCA.2011.5749716.
34. Tsai, Y. F., Xie, Y., Vijaykrishnan, N., & Irwin, M. J. (2005). Three-dimensional cache design exploration using 3DCacti. In *ICCD '05: Proceedings of the 2005 International Conference on, Computer Design* (pp. 519–524).
35. Xie, Y., Loh, G. H., Black, B., & Bernstein, K. (2006). Design space exploration for 3D architectures. *ACM Journal on Emerging Technologies in Computing Systems*, *2*(2), 65–103.
36. Zhao, W., Belhaire, E., Mistral, Q., et al. (2006). Macro-model of spin-transfer torque based magnetic unnel junction device for hybrid magnetic-CMOS design. In *IEEE International Behavioral Modeling and Simulation, Workshop* (pp. 40–43).
37. Zhou, P., Zhao, B., Yang, J., & Zhang, Y. (2009). A durable and energy efficient main memory using phase change memory technology. In *Proceedings of ISCA* (pp. 14–23).
38. Zhou, P., Zhao, B., Yang, J., & Zhang, Y. (2009). Energy reduction for stt-ram using early write termination. In *ICCAD* (pp. 264–268).