Front. Comput. Sci.
DOI 10.1007/s11704-014-4185-4

RESEARCH ARTICLE

Exploiting write power asymmetry to improve phase change memory system performance

Qi WANG (✉)[1,2], Donghui WANG[1], Chaohuan HOU[1]

1 Institute of Acoustics, Chinese Academy of Sciences, Beijing 100190, China
2 School of Physics, University of Chinese Academy of Sciences, Beijing 100049, China

Abstract Phase change memory (PCM) is a promising candidate to replace DRAM as main memory, thanks to its better scalability and lower static power than DRAM. However, PCM also presents a few drawbacks, such as long write latency and high write power. Moreover, the write commands parallelism of PCM is restricted by instantaneous power constraints, which degrades write bandwidth and overall performance. The write power of PCM is asymmetric: writing a zero consumes more power than writing a one. In this paper, we propose a new scheduling policy, write power asymmetry scheduling (WPAS), that exploits the asymmetry of write power. WPAS improves write commands parallelism of PCM memory without violating power constraint. The evaluation results show that WPAS can improve performance by up to 35.5%, and 18.5% on average. The effective read latency can be reduced by up to 33.0%, and 17.1% on average.

Keywords phase change memory, write power asymmetry, command scheduling

1 Introduction

As the number of processor cores increases, the number of concurrently running applications (or threads) increases. This in turn increases memory capacity requirement. However, current DRAM-based main memory is hitting the walls of scalability and power. It is difficult for DRAM technology to scale down to 20 nm[1)]. Furthermore, DRAM memory system consumes 20% to 40% energy of the whole server system [1–3]. Compared to DRAM, non-volatile memories (NVMs) have significant advantages, such as better scalability and lower static power. Therefore, NVMs are promising technologies for next-generation memory system, such as phase change memory (PCM), resistive random access memory (RRAM), and magnetoresistive random access memory (MRAM). Among these NVMs, PCM is a promising candidate [4–7].

The read latency and read power of PCM are in the same range with those of DRAM. However, compared to DRAM, PCM has longer write latency and higher write power, which raises significant challenges in enabling real adoption of PCM. Although write requests are not on the critical path of memory performance, serving a write request blocks subsequent read requests that access the same bank. Thus, write requests can increase the effective latency of read requests, which has significant impact on memory performance. Previous work shows that removing all writes can improve performance by 39.0% on average [8]. Therefore, improving write performance is critical for PCM memory system.

PCM write is a power-intensive operation. Moreover, PCM write power is asymmetric: writing a zero (RESET) consumes more power than writing a one (SET). For PCM memory, the number of write commands that can be served concurrently (called write commands parallelism) is limited due to the maximum power constraint. Data-comparison-write (DCW) [9] only writes the modified bits by reading the old

Received April 20, 2014; accepted September 29, 2014

E-mail: qi0104@126.com

[1)] Process integration, devices and structures (PIDS). ITRS 2012. http://www.itrs.net/links/2012itrs/home2012.htm.

data and comparing it with the new data. By doing so, the power consumption of a write command is reduced. Based on DCW, Hay et al. [5] propose Power-token to improve write commands parallelism without violating the power constraint. However, Power-token disregards the power asymmetry of writing a zero and writing a one, and thus limits write commands parallelism.

In this paper, we propose a new scheduling policy, write power asymmetry scheduling (WPAS), to improve write commands parallelism. Instead of assuming all of the modified bits are zeros as Power-token does, WPAS calculates the power consumption of every write command considering the write power asymmetry. Since writing a one consumes less power than writing a zero and large number of modified bits are ones, WPAS can issue more writes than Power-token under the same power constraint. WPAS improves write commands parallelism, and thus improves PCM memory performance.

We evaluate the effectiveness of WPAS compared with Power-token. The evaluation results show that WPAS improves PCM memory system performance by up to 35.5%, and 18.5% on average. Moreover, WPAS can reduce effective read latency by up to 33.0%, and 17.1% on average. We also evaluate the effectiveness of WPAS under varied system configurations: power ratio[2], LLC size, queue depth, and address mapping scheme. In addition, we apply WPAS to the sub-rank memory system, which is called WPAS-S. The evaluation results show that WPAS-S can further improve performance by 8.4% on average, compared with WPAS. We also evaluate the latency, area, and power overheads of WPAS. The results show that the implementation overheads are low and acceptable in practice.

The remainder of this paper is organized as follows. Section 2 describes background and motivation. Section 3 presents our proposed WPAS technique. Section 4 describes the evaluation methodology. Section 5 shows the evaluation results. Section 6 presents related work, and Section 7 concludes this paper.

2 Background and motivation

2.1 PCM

PCM is a type of non-volatile memory that exploits the property of phase change material to store bit information. A PCM cell is comprised of a transistor and phase change material. The phase change material, such as Ge2Sb2Te5 [10], has two states: crystalline state and amorphous state. The crystalline state has low resistance, which represents “1”. The amorphous state has high resistance, which represents “0”. Applying different amount of electrical current to a PCM cell can switch the phase change material between crystalline state and amorphous state.

Figure 1 shows the write mechanism of a PCM cell. A high and short current is applied to the cell to RESET it to amorphous state, which represents writing a zero. A low and long current is applied to the cell to SET it to crystalline state, which represents writing a one. We can see that writing a zero consumes more power than writing a one. The power ratio is 2.0x in [11], 3.0x in [12], and 5.0x in [4].

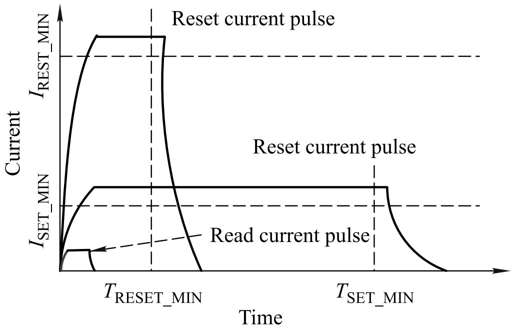

Fig. 1 Write mechanism of a PCM cell

2.2 Motivation

Writing a PCM cell consumes high power, and the maximum power that can be provided to write PCM is limited. Thus, the number of write commands that can write PCM simultaneously is limited. Prior research work Power-token proposes to reduce the power consumption of a write command using DCW, and thus schedules more write commands concurrently under the power constraint. Power-token disregards of the content to be written, and assumes the power consumption to write a modified PCM bit is as large as that of writing a zero. However, writing a zero consumes more power than writing a one for PCM. Power-token overestimates the power consumption of a write command as the data to write is a mix of zeros and ones. Consequently, Power-token limits write commands parallelism of PCM memory system, which leads to inferior performance.

Figure 2 shows the instructions per cycle (IPC) of Power-token, normalized to Unlimited. Unlimited ignores the maximum power constraint, and thus can achieve the largest write commands parallelism. We can see that the IPC of Power-token is less than Unlimited by 53.4% on average. Actually,

[2] Power ratio refers to the power of writing a zero divided by that of writing a one in this paper.

Power-token does not fully exploit the possible write commands parallelism of PCM. Thus, there is a large space of performance improvement. The experiment result motivates us to propose WPAS to further improve the performance of PCM memory system.

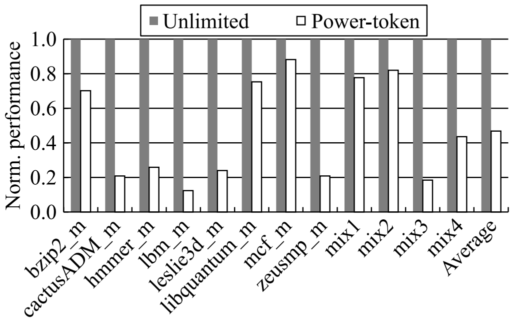

Fig. 2 IPC of Power-token normalized to Unlimited

3 WPAS

In this section, we describe the design of WPAS. Figure 3 presents an overview of WPAS architecture. The gray frames show our modifications. WPAS identifies the modified bits in the LLC, and propagates the modification information to memory controller. When the memory controller issues a write command, it must ensure that all the corresponding PCM chips have enough power supply to support this write.

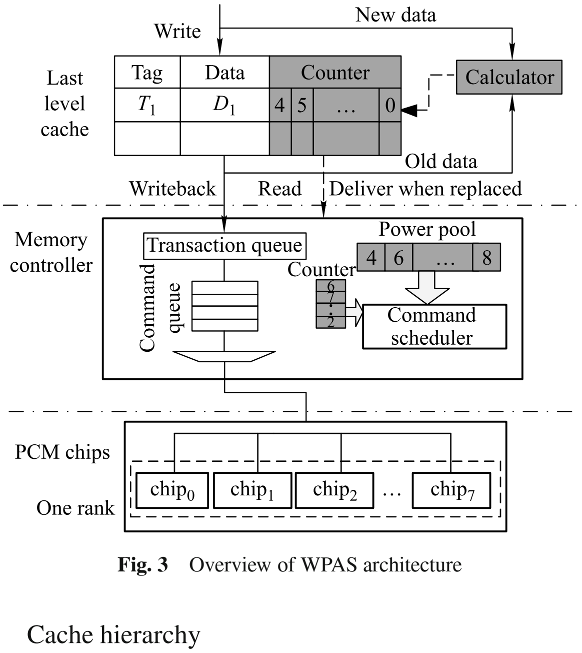

Fig. 3 Overview of WPAS architecture

3.1 Cache hierarchy

We first describe the modifications of cache hierarchy to support WPAS. To calculate the power consumption of every write, we track the information of modified zeros and ones in the LLC. Since a cache line data is mapped to multiple PCM chips, we embed multiple counters in each cache line. Each counter records the number of modified bits of the cache line data that are mapped to the same chip. When the data of upper cache line is written back to LLC, the old data is read from LLC. The calculator compares the old data with the new one and calculates the total number of modified bits at chip granularity. Since a cache line can be written multiple times before written back to memory, the data in the LLC is not alway the same with the memory data. Thus, we calculate the number of modified bits on a conservative way used in Power-token. As long as a bit is modified once, we add the count.

Considering the write power asymmetry, the number of modified zeros is transformed to the number of modified ones. For instance, the numbers of modified zeros and ones are 5 and 10, respectively. Assume the power ratio is 2.0x. Then, the total number of modified bits after transforming is 20, which is recorded in the counter. When the LLC evicts a cache line data to PCM memory, it also sends the modification information in counters to the memory controller.

To calculate the number of modified zeros, an extra read operation is needed before writing. The read does not incur extra latency overhead as it can be overlapped with tag matching of the write [13].

Assuming a 64-bytes cache line is interleaved across eight chips, then each chip need an 8-bits counter to record the number of modified bits. However, our experiments show that only 17.4% bits of a cache line are modified on average, when it is evicted to PCM memory. Considering the tradeoff between overhead and performance, we use a 4-bits counter per chip. Therefore, the storage overhead is 6.3%. Compared with Power-token that uses 3-bits counter, WPAS only incurs extra storage overhead by 1.6%.

To evaluate the area and power overheads, we implement the comparator and counters in Verilog HDL and synthesize the design using synopsys design compiler with 90 nm technology library of TSMC. Then, we use CACTI 6.0 [14] to evaluate the total area and power of whole LLC at 90 nm technology process. Evaluation results show that cache modification incurs area and power overheads by 3.2% and 3.9%, respectively.

3.2 Memory controller

In this section, we present the modifications in memory controller and the scheduling policy of WPAS. The memory controller maintains a copy of the counters that record the number of modified bits of each write command. The memory controller also maintains a power pool to record the number of writing bits that each chip can support currently. Before issuing a write command, the memory controller compares the counters with the power pool at chip granularity. If all the values in the counters are smaller than the corresponding values in the power pool, then the write command can be issued. After issuing a write command, the memory controller decreases the values in power pool. Otherwise, the memory controller schedules other commands that satisfy the scheduling condition. When a write command completes, the memory controller increases the corresponding values in power pool. Since WPAS distinguishes the power consumptions of writing a zero and writing a one, the initial values in power pool with WPAS are different from those with Power-token. Assuming the power ratio is 2.0x, then the initial values of power pool in WPAS is twice of the corresponding values in Power-token.

Figure 4 presents an example of the memory controller scheduling in Power-token and WPAS. For simplicity, we assume each chip has two 4-bit wide banks and a write command spans four chips. We also assume each chip can support to write four zeros concurrently at most, and the power ratio is 2.0x. Since Power-token disregards the content to write, all the initial values in power pool are 4. After the memory controller issuing Write X, the values in power pool are reduced to be 2, 3, 3, and 1. Write Y modifies 3 bits of chip$_0$, while chip$_0$ can only support 2 bits writing. As a result, Write Y cannot be issued, as shown in Fig. 4(a). In contrast, WPAS takes the power asymmetry of writing a zero and writing a one into consideration. Therefore, all the initial values in power pool are 8 in WPAS. After scheduling Write X, there is still enough power supply to schedule Write Y, as shown in Fig. 4(b). Thus, WPAS improves write commands parallelism under the same power constraint.

Figure 5 shows the timing diagram of Power-token and WPAS[3]. In DDRx protocol, the minimum time interval between the beginning of issuing a write command and the end of data are written to memory array is ($t_{\mathrm{BURST}} + t_{\mathrm{AL}} + t_{\mathrm{CWD}} + t_{\mathrm{WR}} + t_{\mathrm{RP}}$) [15]. Since Write Y cannot write PCM array until Write X completes, it can only be issued until time t_{11} in Power-token. However, Write Y and Write X can write PCM array concurrently in WPAS. Therefore, Write Y can be issued immediately at time t_5 when Write X completes data burst (t_{BURST}) and modification information transferring (t_{MOD}).

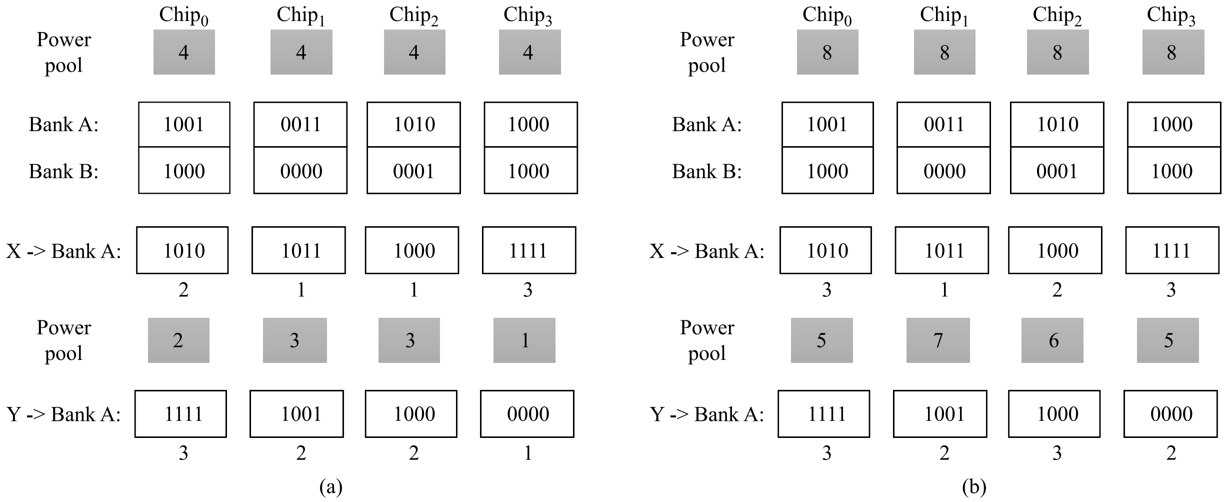

Fig. 4 The memory controller scheduling in Power-token and WPAS. (a) Power-token; (b) WPAS

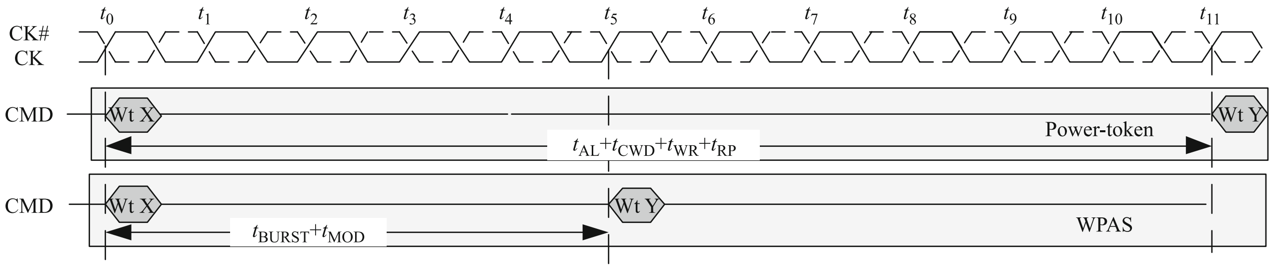

Fig. 5 The timing diagram of Power-token and WPAS. tAL=0, tCWD=1, tWR=6, tRP=4. These parameters derive from Micron data sheet [?]

[3] The parameters derive from Micron data sheet. http://www.micron.com/products/dram/ddr3-sdram#

When a cache line data in LLC is evicted to memory, the modification information in counters is also transferred to memory, which incurs one cycle latency overhead (t_{MOD}). The memory controller compares the counts with values in power pool when issuing a write command. This comparison operation can be overlapped with other operations, such as checking whether a bank is activated. Thus, it does not incur extra latency overhead. The latency overhead (t_{MOD}) of WPAS is much less than the write latency of PCM.

The memory controller maintains a power tool and adds additional comparator, which incurs extra area and power overheads. We also implement these components in Verilog HDL and synthesize them using synopsys design compiler. Evaluation results show that our modifications in memory controller incur 0.043 mm^2 area overhead and 0.28 mW power overhead with 90 nm process.

4 Evaluation methodology

4.1 System configurations

We evaluate the effectiveness of WPAS by using a multiprocessor full-system simulator Gem5 [16] and a cycle accurate memory system simulator DRAMsim2 [17] that is modified to simulate PCM. Gem5 is used to gather memory traces, which are input to the enhanced DRAMsim2. Table 1 shows the baseline system configurations. We use 32 MB LLC to filter frequently accessed data. The power ratio is 2.0x [11].

Table 1 Baseline system configurations

Parameter	Configuration
Processor	Alpha, 8-cores, out-of-order, 2 GHz
	32 KB I-caches, 64 KB D-caches
L1 caches	2-way associative, 64 B cache line, 1-cycle latency
L2 caches	2 MB, 8-way associative, 64 B cache line, 10-cycle latency
LLC	32 MB, 16-way associative, 64 B cache line, 20-cycle latency
	32-entry transaction queue
Memory	32-entry command queue per rank
controller	chan:row:col:bank:rank address mapping
	close-page row buffer policy
Main	2 ranks per channel, 8 banks per rank
memory	32 768 rows/bank, 1 024 columns/row
	t_{RCD}: 55 ns, t_{RP}: 150 ns, P_{w0} = 2 P_{w1}

4.2 Workloads

We use eight workloads and four mixed workloads from SPEC CPU2006 benchmark [18] for experimental evaluation. Table 2 lists all workloads and summarizes the LLC cache misses per-kilo instructions (MPKI), read-to-write ratio (R/W ratio) and percentage of ones among modified bits (One_ratio) for each workload. We select these workloads as their MPKIs are larger than 1. We run each workload at the multiprogramming mode with the reference input set. For each workload, we execute 1 billion instructions and select a representative segment of 10 million memory accesses.

Table 2 Workload characteristics

Workloads	Description	MPKI	R/W ratio	One_ratio
bzip2_m	8 copies of bzip2	1.79	2.41	0.60
cactusADM_m	8 copies of cactusADM	17.34	1.29	0.56
hmmer_m	8 copies of hmmer	1.98	1.03	0.49
lbm_m	8 copies of lbm	22.01	1.34	0.96
leslie3d_m	8 copies of leslie3d	8.01	1.68	0.98
libquantum_m	8 copies of libquantum	15.45	2.64	0.98
mcf_m	8 copies of mcf	17.15	4.03	0.55
zeusmp_m	8 copies of zeusmp	9.17	1.89	0.89
mix_1	2bzip-2mcf -2hmmer-2libquantum	3.81	1.66	0.54
mix_2	2leslie3d-2lbm -2cactusADM-2zeusmp	12.35	1.35	0.66
mix_3	2hmmer-2libquantum -2lbm-2zeusmp	8.85	1.22	0.73
mix_4	2bzip2-2cactusADM -2leslie3d-2mcf	12.28	1.42	0.97

5 Evaluation results

In this section, we evaluate the effectiveness of WPAS. We use the Power-token policy as the baseline. We first evaluate the performance and effective read latency of WPAS. Since memory system performance varies among different configurations, we then conduct a sensitivity analysis of WPAS to show its effectiveness, including power ratio, LLC size, queue depth, and address mapping scheme. We finally evaluate the effectiveness of WPAS when it is applied to the subrank memory system.

5.1 Performance improvement

We first evaluate the performance improvement of WPAS. Figure 6 shows the execution time speedup normalized to the baseline. WPAS can improve the performance by 18.5% on average. This is because WPAS improves write commands parallelism. Figure 7 shows the distribution of how many writes are served concurrently. We only present two workloads as other workloads have similar trend. The maximum write commands parallelism is 8, because there are 8 banks per rank in our configuration. We can see that WPAS improves the percentage of large write commands parallelism compared with baseline. For instance, the percentage of processing 4 writes concurrently improves from 0.6% to 26.3%

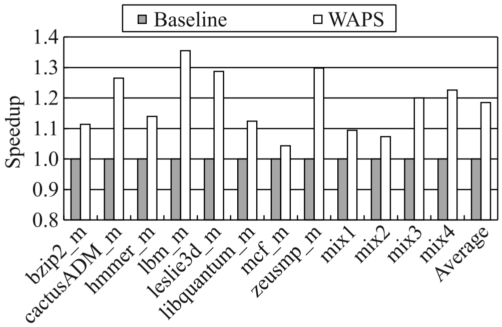

Fig. 6 Execution time speedup

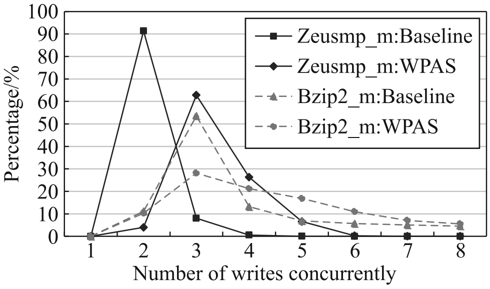

Fig. 7 Distribution of concurrent writes

for zeusmp_m, and from 13.2% to 21.2% for bzip2_m due to WPAS. As a result, WPAS improves PCM memory system performance.

Figure 6 shows that WPAS can improve performance by up to 35.5% for workload lbm_m. In contrast, WPAS only achieves 4.3% speedup for workload mcf_m. The difference of performance improvements is due to three major factors: MPKI, R/W ratio, and One_ratio. The MPKI of cactusADM_m is larger than that of bzip2_m, which allows cactusADM_m benefits more from improved write commands parallelism. Therefore, cactusADM_m achieves higher speedup than bzip2_m due to WPAS. Workloads cactusADM_m and mcf_m have similar MPKI, while the performance speedup of cactusADM_m is higher than that of mcf_m. This is because cacutsADM_m has lower R/W ratio, which means larger benefits by accelerating writes. The One_ratio also has significant impact on the performance improvement of WPAS. For example, mix4 has similar MPKI and R/W ratio with mix2. However, the One_ratio of mix4 is 0.97, which is larger than 0.66 of mix2. High One_ratio indicates that WPAS has large opportunity to improve write commands parallelism through accurately managing the power consumption of writing. As a result, mix4 achieves 22.6% speedup, while mix2 only achieves 7.3% speedup.

5.2 Effective read latency reduction

WPAS can not only speedup the execution time of workload, but also reduce the effective read latency. In this section, we evaluate the impact of WPAS on the effective read latency. Figure 8 shows the effective read latency reduction caused by WPAS. All the results are normalized to the baseline. WPAS can reduce effective read latency by 17.1% on average and up to 33.0% for lbm_m. WPAS improves write commands parallelism, and thus accelerates the processing of write commands. This in turn reduces the queuing time of read requests. Therefore, WPAS can reduce the effective read latency.

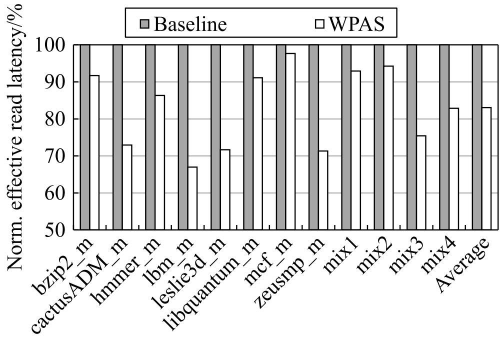

Fig. 8 Effective read latency

5.3 Sensitivity analysis

The behavior of memory system usually varies under different configurations. In order to evaluate how system configurations affect the effectiveness of WPAS, we conduct a sensitivity analysis to WPAS in this section. We evaluate the following performance-critical factors: power ratio, LLC size, queue depth, and address mapping scheme.

5.3.1 Power ratio

PCM technology is still under development. The power ratios of writing a zero to writing a one are diverse among different PCM prototypes and products. For instance, the power ratio is 2x in [11], while it is 5.0x in [4]. Therefore, we first evaluate the impact of power ratio on the effectiveness of WPAS. We evaluate three power ratios: 2.0x [11] and 3.0x [12], and 5.0x [4]. Figure 9 shows the performance speedup in varied power ratios.

As the power ratio increases, the performance speedup increases for all workloads. Since the baseline assumes all the modified bits are zeros, larger power ratio results in larger waste of write commands parallelism. Therefore, WPAS can achieve higher performance improvement under larger power ratio.

5.3.2 LLC size

LLC size affects the behavior of memory accesses to PCM

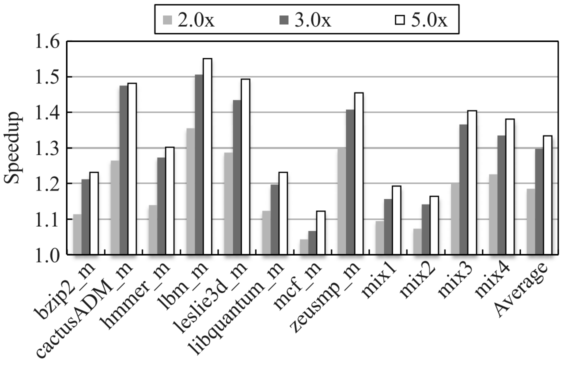

Fig. 9 Impact of power ratio

memory. Therefore, we evaluate the impact of LLC size on the effectiveness of WPAS. We vary LLC size from 8 MB to 64 MB. Figure 10 shows the speedup under varied LLC sizes. The speedup decreases as the LLC size increases for some workloads, such as bzip2_m, mix1. This is because larger cache results in less writes to memory. Consequently, the possibility to find writes that can be scheduled concurrently reduces. However, the trend is different for some other workloads, such as lbm_m, zeusmp_m. This is because writes may be stalled in the queue due to insufficient power supply, which increases the queuing times of reads and writes. Thus, larger cache size unexpectedly results in higher speedup.

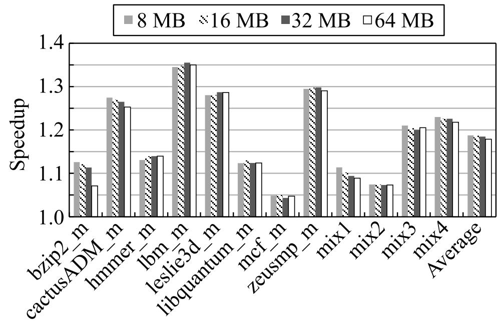

Fig. 10 Impact of LLC size

5.3.3 Queue depth

Figure 11 shows the performance speedup when we vary the queue depth from 8 to 64. As the command queue depth increases, so does the speedup. This is because the possibility of finding writes that satisfy the power limitation also increases as the queue depth increases. Larger write commands parallelism results in higher performance improvement.

5.3.4 Address mapping scheme

Memory system uses an address mapping to translate a physical address into an actual address of memory array. Table 1 shows four kinds of typical address mapping schemes, which are different in priorities on channel, rank, bank, row, and column. The address mapping scheme determines the distribution of memory accesses, and thus affects write commands parallelism. Figure 12 shows the execution time speedup of WPAS under four schemes. We can see that WPAS improves performance more apparently under Scheme$_2$ and Scheme$_3$ than Scheme$_1$ and Scheme$_4$. This is because Scheme$_2$ and Scheme$_3$ can better distributes reads and writes to different ranks and banks. Therefore, WPAS can schedule writes of different banks concurrently.

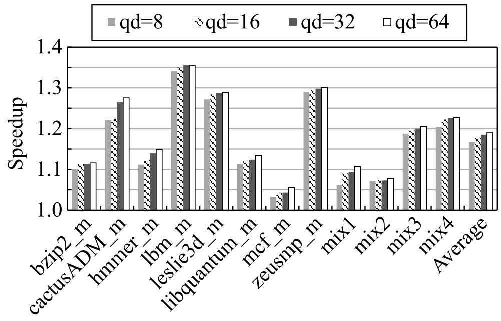

Fig. 11 Impact of queue depth (qd)

Table 1 Address mapping scheme

Scheme	Address mapping priority
1	chan:rank:row:col:bank
2	chan:row:col:bank:rank
3	chan:row:col:rank:bank
4	chan:row:rank:bank:col

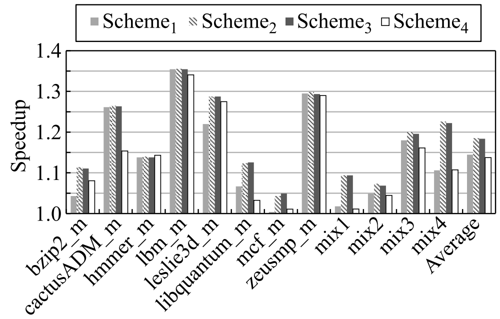

Fig. 12 Impact of address mapping scheme

5.4 Sub-rank memory

In the traditional memory architecture, a write spans multiple memory chips, such as eight. Therefore, a frequently written chip may block write scheduling even though other chips still have enough power supply to support more writes. Sub-rank memory architecture divides a large rank into multiple small ranks. Thus, a write spans less memory chips in sub-rank memory. Sub-rank memory is also common, such as mini-rank [19], HP's MC-DIMM [20]. In this section, we apply

WPAS to sub-rank memory, namely WPAS-S, and evaluate its effectiveness. Each write command only accesses one chip in our evaluation.

Figure 13 shows the performance speedup of WPAS and WPAS-S, normalized to the baseline. Compared to WPAS, WPAS-S can further improve performance by 8.4% on average, and up to 16.0% for mix3. This is because in WPAS, the memory controller must ensure that all the 8 chips have enough power supply when issuing a coarse-grained write command. If any chip does not have enough power supply, then this write will be stalled. However, the memory controller only needs to check if the associated chip has enough power supply when issuing a fine-grained write command in WPAS-S. Therefore, WPAS-S can schedule more write commands than WPAS, and thus further improve performance.

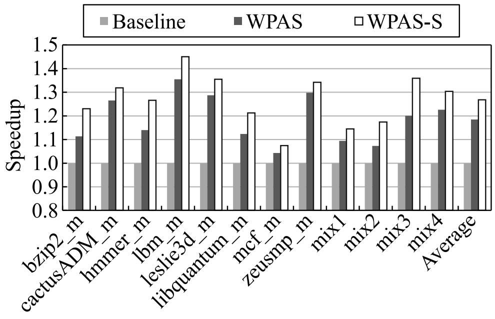

Fig. 13 Speedup of sub-rank memory

6 Related work

6.1 Reducing writes

PCM has write issues, such as long write latency and high write power. Therefore, reducing writes to PCM memory can not only improve performance, but also reduce energy consumption.

Yang et al. [9] propose data-comparison-write (DCW) to reduce unmodified bits writes to PCM memory. Before writing new data to PCM, DCW reads the old data and compares it with the new data. WPAS also utilizes DCW to reduce bits writes to PCM, and thus save power of one write command. Cho and Lee [21] propose Flip-N-Write to further reduce writes to PCM by writing either flipped or unflipped data. If the number of modified bits is less than half of the old data, Flip-N-Write writes the unflipped new data. Otherwise, Flip-N-Write writes the flipped new data. Flip-N-Write can be complementary to WPAS to further reduce power of one write command. Lee et al. [4] propose to add a buffer to each PCM bank. Multiple writes to the same location can be coalesced in the buffer, and thus writes to PCM are reduced. Hybrid DRAM/PCM memory can exploit the advantages of DRAM and PCM while avoids their drawbacks [22–25]. By caching or placing frequently accessed data in DRAM, writes to PCM can be reduced.

6.2 Improving parallelism

Many techniques are proposed to improve write commands parallelism, and thus improve performance of PCM memory. In addition to Power-token, we present other related works that improve write commands parallelism in this section.

Jiang et al. [26] propose two fine-grained power budgeting techniques (FPB) to improve write commands parallelism of MLC PCM, which is an expansion of Power-token. FPB-IPM reduces the power consumption of a write command by splitting the first RESET operation into multiple SET operations when programming a cell. FPB-GCP adds a global charge pump to provide extra power for heavily written PCM chips as these chips would block write commands. Improving the parallelism of PCM chips can also improve write commands parallelism. Conventional electrical bus cannot support a large number of memory chips due to its limitations of insufficient load capacity and signal traversing speed. To overcome this issue, Li et al. [27] propose to utilize the photonics to link the memory controller and PCM chips. Ham et al. [28] propose to add memory buffers in DIMM to keep integrity of signal. These techniques are orthogonal to WPAS.

6.3 Exploiting the asymmetry of SET and RESET

A few research works that utilize the latency and power asymmetries of RESET and SET are related to WPAS. SET takes longer time than RESET, while RESET consumes higher power (and energy) than SET.

Qureshi et al. [8] propose PreSET to improve PCM write performance by utilizing the latency asymmetry. Before the dirty cache line data is evicted to PCM memory, PreSET proactively SETs all the bits in the corresponding memory line. Chen et al. [29] propose out-of-position-writes to reduce write energy that exploits the write energy asymmetry. When writing data, out-of-position-writes chooses an out-of-position block that consumes the least energy to write. WPAS differs from out-of-position-writes because WPAS exploits the power asymmetry to improve write commands parallelism. Yue and Zhu [30] propose two-stage-write that divides a write into two stages: write-0 stage and write-1 stage. All zeros are written at an accelerated speed in write-0 stage, and all ones are written with large parallelism without violating power constraint in write-1 stage. Two-stage-write improves the performance of one write command. However, WPAS allows to issue more write commands concurrently under same power constraint.

7 Conclusion

PCM is a promising technique to replace DRAM as main memory. However, the long write latency and high write power of PCM raise challenges in its adoption. Although Power-token has been proposed to improve PCM performance, there is still a large improvement space. This paper makes the following contributions:

1) We propose a new scheduling policy WPAS to improve PCM memory system performance. WPAS exploits the power asymmetry of writing a zero and writing a one to improve write commands parallelism without violating the power constraint.
2) We quantitatively evaluate the latency, area, and power overheads to implement WPAS. The evaluation results show that the implementation overhead of WPAS is low.
3) We conduct an extensive evaluation to show the effectiveness of WPAS compared with Power-token. Experiment results show that WPAS can improve PCM memory performance by up to 35.5% and 18.5% on average. Furthermore, we show the effectiveness of WPAS under variant memory system configurations.

Acknowledgements This work was supported by National Science and Technology Major Projects of China (2009ZX01 034-001-002-005) and Knowledge Innovation Project of Institute of Acoustics, Chinese Academy of Sciences.

References

1. Lefurgy C, Rajamani K, Rawson F, Felter W, Kistler M, Keller T W. Energy management for commercial servers. IEEE Computer, 2003, 36(12): 39–48
2. Lim K, Ranganathan P, Chang J, Patel C, Mudge T, Reinhardt S. Understanding and designing new server architectures for emerging warehouse-computing environments. In: Proceedings of 35th International Symposium on Computer Architecture. 2008, 315–326
3. Udipi A N, Muralimanohar N C, Niladrish B, Rajeev D, Al J, Norman P. Rethinking DRAM design and organization for energy-constrained multi-cores. SIGARCH Computer Architecture News, 2010, 38(3): 175–186
4. Lee B C, Ipek E, Mutlu O, Burger D. Architecting phase change memory as a scalable dram alternative. SIGARCH Computer Architecture News, 2009, 37(3): 2–13
5. Hay A, Strauss K, Sherwood T, Loh G H, Burger D. Preventing PCM banks from seizing too much power. In: Proceedings of 44th Annual International Symposium on Microarchitecture. 2011, 186–195
6. Yue J, Zhu Y. Exploiting subarrays inside a bank to improve phase change memory performance. In: Proceedings of Design, Automation Test in Europe Conference Exhibition. 2013, 386–391
7. Ni J, Hu W, Li G, Tan K, Sun D. Bp-tree: a predictive B+-tree for reducing writes on phase change memory. IEEE Transaction on Knowledge and Data Engineering, 2014, 26(10): 2368–2381
8. Qureshi M K, Franceschini M M, Jagmohan A, Lastras L A. PreSET: improving performance of phase change memories by exploiting asymmetry in write times. In: Proceedings of 39th Annual International Symposium on Computer Architecture. 2012, 380–391
9. Yang B, Lee D, Kim J, Cho J, Lee J, Yu S Y, Gon B. A low power phase-change random access memory using a data-comparison write scheme. In: Proceedings of International Symposium on Circuits and Systems. 2007, 3014–3017
10. Yamada N O, Eiji N, Kenichi A. Nobuo T, Masatoshi. Rapid phase transitions of GeTeSb2Te3 pseudobinary amorphous thin films for an optical disk memory. Journal of Applied Physics, 1991, 69(5): 2849–2856
11. Kang S, Cho W Y, Cho B H, Lee K J, Lee C S. Oh H R, Choi B G, Wang Q, Kim H J, Park M H, Ro Y H, Kim S, Kim D E, Cho K S, Ha C D, Kim Y R, Kim K S, Hwang C R, Kwak C K, Byun H G, Shin Y S. A 0.1/spl mu/m 1.8V 256Mb 66MHz synchronous burst PRAM. In: Proceedings of International Conference on Solid-State Circuits - Digest of Technical Papers. 2006, 487–496
12. On H, Cho B H, Cho W Y. Enhanced write performance of a 64 Mb phase-change random access memory. In: Proceedings of International Conference on Solid-State Circuits-Digest of Technical Papers. 2005, 581–584
13. Lee Y, Kim S, Hong S, Lee J. Skinflint DRAM system: minimizing DRAM chip writes for low power. In: Proceedings of 17th International Symposium on High Performance Computer Architecture. 2013, 25–34
14. Muralimanohar N, Balasubramonian R, Jouppi N P. CACTI 6.0: a tool to model large caches. HP Laboratories, 2009, 22–31
15. Bruce J, Spencer W, Wang D. Memory systems-cache, DRAM, disk. Morgan Kaufmann. 2008, 428–429
16. Binkert N, Beckmann B, Black G, Reinhardt S K, Saidi A, Basu A, Hestness J, Hower D R, Krishna T, Sardashtis, Sen R, Sewell K, Shoaib M, Vaish N, Hill M D, Wood D A. The gem5 simulator. SIGARCH Computer Architecture News, 2011, 39(2): 1–7
17. Rosenfeld P, Cooper B E, Jacob B. DRAMSim2: a cycle accurate memory system simulator. Computer Architecture Letters, 2011, 10(1): 16–19
18. Standard Performance Evaluation Corporation. SPEC CPU 2006.
19. Zheng H, Lin J, Zhang Z, Gorbatov E, David H, Zhu Z. Mini-rank: adaptive DRAM architecture for improving memory power efficiency. In: Proceedings of 41st International Symposium on Microarchitecture. 2008, 210–221
20. Ahn J H, Leverich J, Schreiber R S, Jouppi N P. Multicore DIMM: an energy efficient memory module with independently controlled DRAMs. Computer Architecture Letters, 2009, 8(1): 5–8
21. Cho S, Lee H. Flip-N-Write: a simple deterministic technique to im-

prove PRAM write performance, energy and endurance. In: Proceedings of 42nd Annual International Symposium on Microarchitecture. 2009, 347–357

22. Qureshi M K, Srinivasan V, Rivers J A. Scalable high performance main memory system using phase-change memory technology. In: Proceedings of 36th Annual International Symposium on Computer Architecture. 2009, 24–33
23. Ramos L E, Gorbatov E, Bianchini R. Page placement in hybrid memory systems. In: Proceedings of International Conference on Supercomputing. 2011, 85–95
24. Lee S, Bahn H, Noh S H. Characterizing memory write references for efficient management of hybrid PCM and DRAM memory. In: Proceedings of 19th International Symposium on Modeling, Analysis Simulation of Computer and Telecommunication Systems. 2011, 168–175
25. Lee H G, Baek S, Nicopoulos C, Kim J. An energy- and performance-aware DRAM cache architecture for hybrid DRAM/PCM main memory systems. In: Proceedings of 29th International Conference on Computer Design. 2011, 381–387
26. Jiang L, Zhang Y, Childers B R, Yang J. FPB: fine-grained power budgeting to improve write throughput of multi-level cell phase change memory. In: Proceedings of the 45th Annual IEEE/ACM International Symposium. on Microarchitecture. 2012, 1–12
27. Li Z, Zhou R, Li T. Exploring high-performance and energy proportional interface for phase change memory systems. In: Proceedings of 17th International Symposium on High Performance Computer Architecture. 2013, 210–221
28. Ham B K, Chelepalli T J, Lee N, Xue B C. Disintegrated control for energy-efficient and heterogeneous memory systems. In: Proceedings of 19th IEEE International Symposium on High Performance Computer Architecture. 2013, 424–435
29. Chen J, Chiang R C, Huang H H, Venkataramani G. Energy-aware writes to non-volatile main memory. ACM SIGOPS Operating Systems Review, 2012, 5(3): 48–52
30. Yue J, Zhu Y. Accelerating write by exploiting PCM asymmetries. In: Proceedings of 17th International Symposium on High Performance Computer Architecture. 2013, 282–293

Qi Wang is a PhD candidate in Institute of Acoustics, Chinese Academy of Sciences, China. Her research interests include VLSI design, computer architecture, and emerging memory technologies.

Donghui Wang received his BS from Tsinghua University, China in 1997. He received the PhD from Institute of Semiconductors, Chinese Academy of Sciences in 2002. Now, he is a professor of Institute of Acoustics, Chinese Academy of Sciences. His research interests include digital signal processor design, VLSI design and signal processing.

Chaohuan Hou received his BS from Peking University, China in 1958. He is a professor of Institute of Acoustics, Chinese Academy of Sciences, China. He was elected as a member of the Academic Division of Science and Technology, the Chinese Academy of Sciences in 1995. His research interests include VLSI signal processing, DSP design, and CPU design.