PRAM Wear-Leveling Algorithm for Hybrid Main Memory Based on Data Buffering, Swapping, and Shifting

Sung Kyu Park, Hyunchul Seok, Dong-Jae Shin, and Kyu Ho Park
Korea Advanced Institute of Science and Technology (KAIST)
305-701, Guseong-dong, Yuseong-gu, Daejeon, Korea
{skpark, hcseok, djshin, kpark}@core.kaist.ac.kr

ABSTRACT

We propose a novel wear-leveling algorithm for the hybrid main memory architecture which exploits both fast read and write speed of DRAM and low power consumption and high density of PRAM. The wear-leveling algorithm consists of three techniques: DRAM buffering for reducing the write count, multiple data swapping for evening out the write count among all pages, and data shifting evening out the write count among all pages and lines. In order to evaluate performance, we implement a PIN-based wear-leveling simulator. In SPEC CPU2006, our proposed schemes can reduce the write count and maintain the write count equally among all pages and lines with little additional overhead.

1. INTRODUCTION

For several decades, DRAM has been mainly used as the main memory. However, DRAM is not enough to support new applications which require large memory capacity because its idle power has significantly increased in proportion to the size and it has almost reached the limit of increasing the memory capacity. In order to overcome these drawbacks, many researchers have an effort to replace DRAM with Phase Change RAM (PRAM) which is one of the emerging new memory technologies [1], [2], [3]. PRAM has little idle power consumption and PRAM is more scalable than DRAM. However, PRAM has latency and endurance problems to be used for the main memory. The latency problem can be solved by exploiting a hybrid main memory architecture which exploits both fast read and write speed of DRAM and low power consumption and high density of PRAM [3]. However, there still remains the endurance issue.

In this paper, we propose a novel wear-leveling algorithm for the hybrid main memory architecture for overcoming the endurance issue. The wear-leveling algorithm consists of three techniques which are DRAM buffering, data swapping, and data shifting schemes. DRAM buffering scheme stores frequently updated data by line-level granularity, thus reducing the write count to PRAM main memory. Data swapping scheme swaps data in old pages whose write count is large with data in new pages whose write count is small. By swapping multiple pages at once, we efficiently maintain the write count equal among all pages compared with one page swapping. In order to even out the write count among all lines in pages, we apply line-level data shifting scheme with shifting timing consideration.

2. PRAM WEAR-LEVELING METHOD

The basic concept of proposed wear-leveling algorithm in hybrid main memory is illustrated in Figure 1. First, DRAM is used as a buffer of PRAM for reducing the write count to PRAM. There are two issues when designing a DRAM buffering scheme. The first design issue is a management policy that determines which data are evicted when DRAM is full. In our design, we apply Least Recently Used (LRU) scheme which discards the least recently used data first from DRAM for absorbing frequently updated data. The second design issue is a management granularity. In our design, we adapt line-level management granularity. Therefore, it needs no additional bits which identify dirty lines because the request is a line size. In addition, we implement the line-level LRU scheme with a hash table and doubly linked list for reducing the management overhead.

In order to improve the endurance, it is important to make the write count even among all pages of PRAM as well as reduce the write count. Therefore, we design multiple data swapping scheme for evening the write count of pages. Multiple data swapping scheme swaps data in the oldest pages with data in the youngest pages for efficiently handling a wide range of pages. When designing a multiple data swapping scheme, there are several considerations. First, we have to manage the write count of PRAM for identifying which pages are frequently updated. We manage all line write counts with a hash table, then calculating the page write count which is defined as the highest line count among in the page. The page write count is used for selecting oldest and youngest pages. Second, we should consider the swapping period which determines when multiple data swapping operation occurs. It can critically affect the wear-leveling effect and performance overhead like latency and additional write operations. We set the swapping period with memory access number, thus multiple data swapping is invoked if the swapping period is over the threshold value. Finally, we have to determine how many pages are swapped at once. The number of swapping pages also critically affect the wear-leveling effect and performance overhead like latency and additional write operations. With these considerations, we

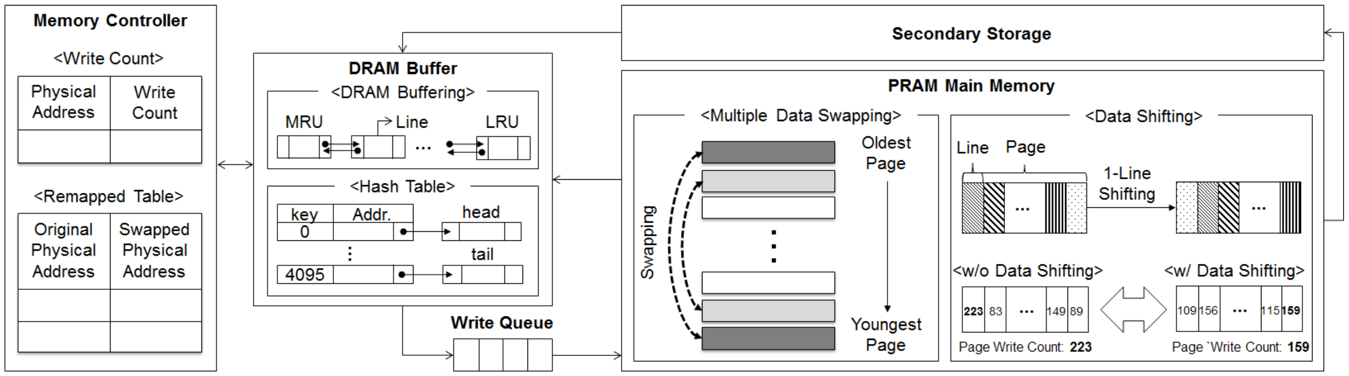

Figure 1: Proposed Wear-Leveling Algorithm in Hybrid Main Memory Architecture

can improve a wear-leveling effect with multiple data swapping scheme. Although we determine these swapping period and the number of swapping pages with various experiments in current design, we'll change them dynamically according to the workload pattern as further work.

Data shifting scheme is necessary to make the line write counts even. Our design considerations of data shifting are a shifting level and shifting period. Similar with data swapping scheme, these considerations have a tradeoff between a wear-leveling effect and overhead. In our design, We first set the shifting level to the line size which is a request size from a last level cache. The shifting period is determined with the write count of each page. When a victim is evicted from DRAM buffer, the write count of the page is checked. If the write count is over the shifting period, the line-leveling shifting operation is invoked. The line-level shifting scheme can make even the write count of all lines as well as reduce the page write count calculated by line write counts. This is because the write operations are distributed in all lines in a page. While it can improve the endurance of PRAM main memory, the shifting operation makes additional page read and write operations. Therefore, it is important to determine an appropriate shifting period for satisfying both wear-leveling effect and low overhead in performance. Although current algorithm sets to the value statically, we'll change it dynamically according to the workload pattern.

3. EXPERIMENT

In order to evaluate performance of wear-leveling algorithm, we implement a PIN-based wear-leveling simulator. We set DRAM size to 19,200 lines (about 1MB), the swapping period to 1,000,000 memory accesses, the number of swapping numbers to 1,000 pages, and the shifting period to 500 page write counts. These configurations are gathered from various experiments. We use mcf and omnetpp from SPEC CPU2006 benchmark. We finally evaluate the average write count, maximum write count, and standard deviation of write count as performance metrics.

Figure 2 shows experimental results with mcf and omnetpp. From these results, we show that the proposed wear-leveling algorithm can considerably reduce the average write count, maximum write count, and standard deviation of write count which means how the write counts are distributed. It means that the proposed wear-leveling algorithm can improve the main memory by reducing the write count and evening out the write counts.

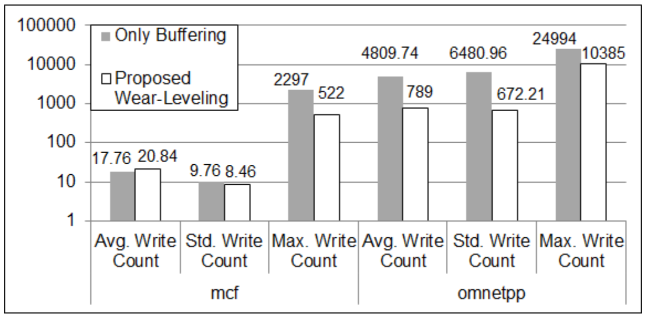

Figure 2: Experimental Result

4. CONCLUSION AND FURTHER WORK

We propose a novel wear-leveling algorithm for the hybrid main memory architecture of DRAM and PRAM. We design three techniques: DRAM buffering, data swapping, and data shifting. In DRAM buffering scheme, we can reduce the write count. In data swapping scheme, we can evening out the page write counts. In data shifting scheme, we can evening out both page and line write counts by line-level shifting. As further work, we'll consider the workload pattern, thus dynamically applying the swapping period and the shifting period. We'll also devise a wear-leveling scheduling for reducing the overhead which is made by additional wear-leveling operations. By scheduling these additional operations, we can reduce the main memory latency.

5. REFERENCES

[1] P. Zhou, B. Zhao, J. Yang, and Y. Zhang, A Durable and Energy Efficient Main Memory Using Phase Change Memory Technology, Proc. *The 36th International Symposium on Computer Architecture (ISCA '09)*, pp. 14-23, 2009.

[2] Y. Park, S.-H. Lim, C. Lee, and K. H. Park, PFFS: A Scalable Flash Memory File System for the Hybrid Architecture of Phase-Change RAM and NAND Flash, Proc. *The 2008 ACM Symposium on Applied Computing (SAC '08)*, pp. 1498-1503, 2008.

[3] M. K. Qureshi, V. Srinivasan, and J. A. Rivers, Scalable High Performance Main Memory System Using Phase-Change Memory Technology, Proc. *The 36th International Symposium on Computer Architecture (ISCA '09)*, pp. 24-33, 2009.