An Efficient Cache Management Scheme for Capacitor Equipped Solid State Drives*

Congming Gao*, Liang Shi*, Yejia Di*, Qiao Li‡, Chun Jason Xue‡ and Edwin H.-M. Sha†

*Chongqing University, China, ‡City University of Hong Kong, China, †East China Normal University, China

ABSTRACT

Within SSDs, random access memory (RAM) has been adopted as cache inside controller for achieving better performance. However, due to the volatility characteristic of RAM, data loss may happen when sudden power interrupts. To solve this issue, capacitor has been equipped inside emerging SSDs as interim supplier. However, the aging issue of capacitor will result in capacitance decreases over time. Once the remaining capacitance is not able to write all dirty pages in the cache back to flash memory, data loss may happen. In order to solve the above issue, an efficient cache management scheme for capacitor equipped SSDs is proposed in this work. The basic idea of the scheme is to bound the number of dirty pages in cache within the capability of the capacitor. Simulation results show that the proposed scheme achieves encourage improvement on lifetime and performance while power interruption induced data loss is avoided.

CCS CONCEPTS

• **Information systems** → **Flash memory**; • **Computer systems organization** → *Embedded systems*; Reliability;

KEYWORDS

Cache Management, Capacitor, Flash Memory

1 INTRODUCTION

Solid State Drives (SSDs) have been widely used in embedded systems, data centers, and emerging Internet of Things (IoT) devices due to its well identified advantages. Within SSDs, random access memory (RAM) has been widely adopted for achieving better performance. However, due to the volatility characteristic of RAM, there is a risk of data loss when sudden power supply interruption happens, leading to the degradation of robustness of SSDs. In order to solve this issue, capacitor is suggested to be equipped within emerging SSDs as interim power supplier to avoid the data loss [1][2]. However, one of the key issues for the capacitor is the aging problem, which introduces decreased capacitance, especially due to the impact of temperature [3]. With capacitor aging, once the remaining capacitance of capacitor is not able to write all dirty pages back to flash memory, data loss happens. In this work, approaches are proposed to indirectly detect the capacitance of capacitor and then solve the capacitor aging issue with achieving performance and lifetime improvement.

From our investigation, there are few works focusing on capacitor aging induced data loss issue in SSDs. One of the reasons is that traditional SSDs do not equip capacitor within SSDs. The other one is that most of the recent products provide enough capacitance to make sure the dirty pages can be successfully written back to flash memory in normal case [4]. However, for some extreme environment, SSDs are going to suffer from high or low temperature so that the interim capacitor will be highly aged, leading the decrease of the integrity of resided data. In order to solve this issue, Guo et al. [5] proposed to use emerging non-volatile memory as cache to store dirty pages. However, these memories are always expensive and these non-volatile memories generally have lifetime issue. In addition, Huang et al. [4] proposed a smartbackup strategy, which aims at writing dirty pages back to SSDs with faster program speed [6]. But once resided capacitor is highly aged and the number of dirty pages exceeds the capability of remaining capacitance, data loss still occurs.

In this paper, a cache management scheme is proposed to solve the above issue. The basic idea of the scheme is to bound the number of dirty pages in the cache. The proposed scheme is realized in two steps: First, a periodical dirty page budget detection scheme is proposed. Second, with the dirty page budget, a smart dirty page synchronizing scheme is proposed. This scheme can reduce the impact from synchronizing dirty pages through carefully selecting the synchronization timing. To our best acknowledge, this is the first work in designing a cache management scheme without adding other components to combat the capacitor aging issue inside SSDs. With applying this approach, the data loss issue can be avoided and the performance impact also can be minimized. The major contributions of this work are as follows: 1) Proposed a dirty page budget detection, which acquires the number of dirty pages within the capability of capacitor; 2) Proposed a smart dirty page synchronizing scheme, which can achieve the performance and lifetime improvement of SSDs; 3) Implemented the proposed work in an SSD simulator for validating the effectiveness of the proposed cache management scheme.

The rest of the paper is organized as follows. Section 2 presents the background. Section 3 presents the motivation of this work. In Section 4, an efficient cache management is presented. Experiments are presented in Section 5. Finally, Section 6 summarizes the paper.

*Corresponding author: Liang Shi. Email: shi.liang.hk@gmail.com.

GLSVLSI'18, Chicago, IL, USA

 978-1-4503-5724-1/18/05...$15.00
DOI: 10.1145/3194554.3194639

2 BACKGROUND

2.1 Capacitor Equipped SSDs

Currently, two types of capacitors, super-capacitor [5] and Tantalum capacitor [2], have been widely adopted in advanced SSDs. For super-capacitor, its lifetime is quite sensitivity to temperature. Based on Arrhenius law lifetime model [3], the lifetime of super-capacitor is reduced by half when the temperature is increased by 10°C. For example, previous work [5] presented that the capacitance degradation of super-capacitors under 60°C can be as high as 30% within five years. For Tantalum capacitor, it can operate with a larger temperature range (e.g., -40°C to 105°C) [1]. However, if the temperature falls out of the range, capacitance will be significantly decreased as well. For example, previous work [7] presented that with a low temperature (e.g., -55°C), the capacitance of Tantalum capacitor will drop by at least 20%. Currently, advanced SSDs are widely equipped with capacitors to avoid the data loss. However, from the above discussion on the aging issue of capacitors, it is still a key problem when the capacitors are aged with a significant capacitance decrease.

2.2 Reducing Capacitor Aging Induced Impact

Capacitor aging is able to introduce data loss while the number of dirty pages exceeds the remaining capability of capacitor. In order to avoid this issue, emerging non-volatile memory is adopted as a part of cache inside SSD controller, which can be used to store dirty pages for preventing data loss issue [5][8]. In previous works, Guo et al. [5] proposed that when sudden power supply interruption occurs, dirty pages can be moved to phase change random access memory (PRAM) instead of NAND flash memory with less energy demanding. Similarly, Kim et al. [8] proposed to use non-volatile memory for storing these dirty pages exceeding the capability of capacitor. In addition, similar issue also exists in main memory. Kateja et al. [9] proposed to decouple the capacitor requirement and size of main memory through bounding the volume of dirty data in the main memory.

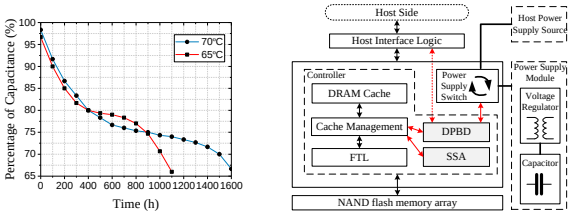
However, the above works still are un-known to the remaining capacitance of capacitor so that how many dirty pages can be resided in cache is hard to determine.

3 MOTIVATION

3.1 Capacitor aging may induce data loss

In work [10], authors show that the maximum energy supplied by capacitor is proportional to the current capacitance of capacitor. For the energy supplied by capacitor, it is in charge of writing dirty pages back to flash memory when power interruption occurs, which is proportional to the energy supplied by capacitor as well [5]. Based on these two works, we can find that the maximum number of dirty pages within the capability of capacitor is proportional to the current capacitance of capacitor.

However, the capacitance will be decreased with capacitor aging. Figure 1(a) presents the capacitance degradation of super-capacitor over time [3]. As shown in this figure, the capacitance is constantly decreasing during its whole lifetime. In addition, we also find that temperature has a significant influence on the lifetime. Since both super-capacitor and Tantalum capacitor belong to electrolytic capacitors, the relationship between lifetime and temperature can be



(a) Relative capacitance change with 65°C and 70°C [3].

(b) The framework of proposed approach.

Figure 1: The characteristic of capacitor aging and the architecture of proposed scheme.

Table 1: Normalized access latency and write amplification.

	Baseline	Method 1	Method 2
Read Lat.	1	0.8531	0.7607
Write Lat.	1	1.9927	0.1386
Write Amp.	1	1.6136	15.093

described by the law of Arrhenius [3], which shows that the lifetime degradation of capacitor matches exponential decrease with increased relative temperature. That is, at extreme environment (high temperature for super-capacitor or low temperature for tantalum capacitor), the capacitance of capacitor will decreases sharply.

Based on above descriptions, once the capacitance of the equipped capacitor inside SSDs is not able to synchronize all dirty pages back to SSDs, data loss happens.

3.2 Dirty page synchronization induced impact

The straightforward method to avoid the capacitor aging induced data loss is to bound the number of dirty pages in the cache. There exist two related methods can be used. The first method (Method 1) is to synchronize dirty pages back to SSD once the number of dirty pages reaches the capability of the capacitor [9]. The second method (Method 2) is to synchronize dirty pages during idle time of SSDs to avoid the performance impact from above method, which has been widely adopted in SSDs for achieving performance improvement [11], especially for garbage collections [12].

In preliminary experiment, STG_0 trace is selected as an example and three schemes are implemented in SSDsim[11]. In the experiment, the dirty page capability is set to 70%, which means that current capacitance of capacitor only can support to write 70% cache capacity of dirty pages. More details are presented at Section 5. Baseline represents the case that there are no capacitor aging problem inside SSDs.

Table 1 presents the normalized access latency and write amplification. From the results, we have following observations: First, Method 1 has smaller write amplification, but introduces a high write access latency. This is because that frequently synchronization operations will block latter coming host requests. Second, Method 2 has a much better access performance, but induces a high write amplification due to synchronizing more dirty pages back to SSD.

From the observations, we can find that although synchronization can avoid data loss issue, the timing of synchronizing dirty pages should be carefully selected to reduce the impact on both performance and lifetime.

4 AN EFFICIENT CACHE MANAGEMENT SCHEME

4.1 Overview

Figure 1(b) presents the overview of the architecture, where two modules are added: Dirty Page Budget Detection (DPBD) and Smart Synchronizing Activation (SSA).

4.2 Dirty Page Budget Detection Module

The basic idea of DPBD is to periodically synchronize the pages in the cache back to SSDs via interim power supplier. The process of the detection module is as follows: during the process of periodically synchronizing pages back to SSDs, dirty pages are synchronized first, and then clean pages are synchronized repeatedly until the capacitor is exhausted. For obtaining dirty page budget, a page counter is used to record the number of synchronized pages, which determines dirty page budget.

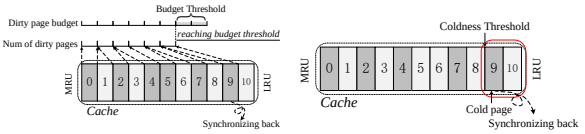
In addition, another important issue is the detection period, which indicates the period between two continuous DPBD processes. In this work, we propose to use a fixed threshold, 20 days, as the detection period, which can be set based on the user's demand. After detection period reaches 19 days, idle time should be found for activating detection process in the last day. Otherwise, detection process is activated with highest priority.

In order to minimize the impact of capacitance degradation during two continual DPBD process, a redundant space from the acquired number of pages to the dirty page budget is added. In this work, five discrete budgets are used to maintain the redundancy, 100%, 90%, 80%, 70% and 0%, respectively. Since the lifetime of capacitor reaches the end of life when capacitance drops to 70% of rated capacitance, the last budget is set to 0% after the fourth one (70%). Note that, if the dirty page budget is 0%, write through policy should be adopted in cache management.

4.3 Smart Synchronizing Activation Module

In this section, a smart synchronizing scheme is proposed to improve both the performance and lifetime of SSDs. The basic idea of the scheme is to synchronize dirty pages based on the dirty page budget. If the number of dirty pages in the cache is approaching the budget, they should be synchronized as soon as possible. Otherwise, they are delayed to avoid immature synchronization. In the following, the above scheme is presented in two cases: idle time synchronization and busy time synchronization.

Idle time synchronization: In order to avoid immature synchronization induced severe write amplification in Method 2, two new synchronization timings are selected. The first one is that a new budget threshold is used to determine the activation of the synchronization process. If the number of dirty pages approaches the budget and the distance between them is smaller than the defined threshold, dirty pages are synchronized. Otherwise, no synchronization is activated. The second one is that if there are cold dirty pages in the cache, the synchronization process can be activated even the distance between number of dirty pages in the cache and the budget is smaller than the defined threshold. In this case, these cold dirty pages are synchronized back to SSDs in advance during idle time for minimizing the impact on performance. In this work, we assume that the cache is organized in LRU list. In this setting, another threshold is added to indicate the coldness. With above approaches, the immature synchronization can be avoided and the number of dirty pages in the cache can be minimized. Figure 2 shows an example for the above two thresholds. Figure 2(a) shows the first case. Assuming dirty page budget is 8, the number of dirty pages is 6, and budget threshold is 6. That is, the number of current dirty pages reaches budget threshold so that page 9 should be synchronized back. Figure 2(b) shows the second case. In this case, assuming coldness threshold is 9, which indicates that pages after the 9th page are identified as cold pages. In this figure, page 9 belonging to cold pages is synchronized back. Note that, in this work, budget and coldness thresholds are set as 5% of dirty page budget and capacity of cache, respectively.



(a) Budget threshold. (b) Coldness threshold.

Figure 2: Two new synchronization timings.

Busy time synchronization: When there are no idle time, we propose to evict pages based on original LRU policy if the number of dirty pages does not approach the budget. Otherwise, dirty pages are synchronized in a higher priority.

Based on the proposed scheme, not only the number of dirty pages can be bounded below the dirty page budget, but also the synchronization induced performance and lifetime degradation can be minimized. During synchronizing dirty pages back to SSDs, these dirty pages are synchronized with write protection for avoiding losing data consistency [9].

5 EXPERIMENT

5.1 Experiment Setup

In this paper, we use a trace driven simulator, SSDsim [11], to evaluate the proposed framework. For the experiment, an 128GB SSD, which is configured with 4 channels with each channel equipped with 4 chips, is modeled. Each flash page size is set to 4KB. The cache capacity is set to 128MB.

The workloads include a set of MSR Cambridge traces [13], two Financial traces [14], and one personal computer traces collected via DiskMon [15]. In the experiment part, four schemes, Baseline, Method 1, Method 2, and the proposed scheme, are implemented. Each scheme excepting Baseline contains three cases, 90%, 80% and 70%, which present the dirty page budgets. If dirty page budget is smaller than 70%, the budget is set as 0%. When dirty page budget is set as 0%, all schemes synchronize dirty pages back to SSDs immediately so that all three schemes will achieve same results.

5.2 Results and Analysis

In the following, the access latency and write amplification are evaluated. In the following figures, results of Method 1, Method 2 and proposed scheme are normalized to the result of Baseline.

Write Latency: Figure 3 presents the write latency evaluations among four schemes, where dirty page budgets are set as 90%, 80% and 70%, respectively. In this results, we can find that Method 1 commonly introduces a high write latency, Method 2 achieves the minimal write latency. First,

Method 1 only synchronizes dirty pages back to SSDs when the number of dirty pages approaches the budget. Hence, there exist more conflicts between synchronized dirty pages and host I/O requests. That is, the poor write performance of Method 1 directly comes from the synchronization induced conflict on host requests. Second, the reason why Method 2 achieves the best write performance is that dirty pages are usually synchronized back in idle time so that host I/O requests can be cached.

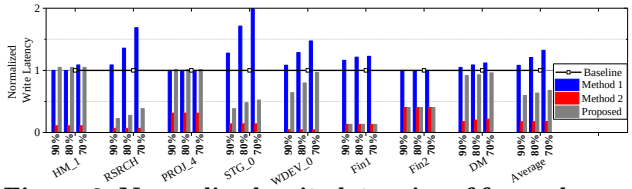


Figure 3: Normalized write latencies of four schemes.

In addition, in most cases, the proposed scheme achieves better write latency compared with Method 1 and Baseline with little degradation compared with Method 2. The reason comes from that the proposed scheme only synchronizes dirty pages when the number of dirty pages approaches the dirty page budget. In this case, compared with Method 1, the proposed scheme is able to cache more host write requests due to synchronizing dirty pages in advance.

Read Latency: Figure 4 presents the normalized read latency. In this figure, we find that read latency is significantly reduced compared with Baseline. The reason comes from that if dirty pages are synchronized, there will be fewer dirty pages supposed to be evicted when read miss occurs. Two observations can be found from the results: First, in most cases, Method 2 achieves the best read performance among these four schemes. This is because Method 2 achieves better write latency, which reduces the impact of read/write interference [16].

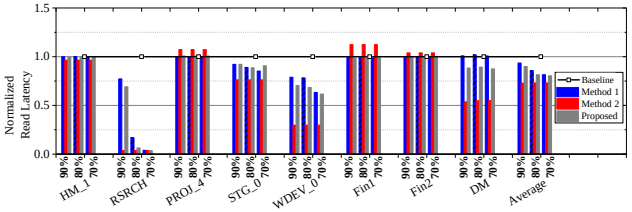


Figure 4: Normalized read latencies of four schemes.

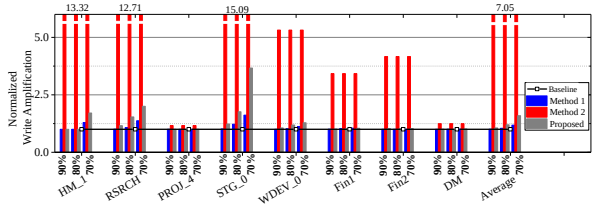


Figure 5: Normalized write amplifications of four schemes.

Second, when the budget is smaller, read latency is smaller in some cases. This is because more pages in cache are clean. However, there also exist some exceptions, such as PROJ_4 of Method 2. This is because frequently synchronization delays incoming read requests [16].

Write Amplification: Figure 5 presents the normalized write amplification. In this figure, we find that Method 1 achieves similar write amplification to Baseline. But Method 2 has the worst write amplification. Different from these two schemes, the proposed scheme achieves similar write amplification to Method 1. This is because that the number of additional write requests is significantly reduced through smartly activating synchronization. In addition, some cases, such as DM2 of Method 2, achieve close results to the Baseline as well. The reason comes from that there exist burst of accessing requests so that less idle time can be used to synchronize dirty pages.

Based on the above results, we can find that the proposed scheme is efficient in improving both performance and lifetime for capacitor equipped SSDs.

6 CONCLUSION

In this paper, first, a dirty page budget detection scheme is proposed. Then, a smart synchronizing scheme is proposed to reduce the synchronizing dirty pages induced passive impacts on SSDs. Experiments show that the proposed scheme improves both performance and lifetime while the capacitor aging induced data loss issue is being avoided.

7 ACKNOWLEDGEMENT

This work is supported by the NSFC 61772092 and 61572411.

REFERENCES

[1] Power Loss Imminent (PLI) Technology. In *Intel Corporation*, 2014.
[2] Woon Hak Kang, Sang Won Lee, Bongki Moon, Yang Suk Kee, and Moonwook Oh. Durable write cache in flash memory ssd for relational and nosql databases. In *SIGMOD*. ACM, 2014.
[3] G. Alcicek, H. Gualous, P. Venet, R. Gallay, and A. Miraoui. Experimental study of temperature effect on ultracapacitor ageing. In *EPE*. IEEE, 2007.
[4] Min Huang, Yi Wang, Liyan Qiao, Duo Liu, and Zili Shao. Smartbackup: An efficient and reliable backup strategy for solid state drives with backup capacitors. In *HPCC*. IEEE, 2015.
[5] Jie Guo, Jun Yang, Youtao Zhang, and Yiran Chen. Low cost power failure protection for mlc nand flash storage systems with pram/dram hybrid buffer. In *DATE*. ACM, 2013.
[6] Yuming Chang, Yuanhao Chang, Teiwei Kuo, Yungchun Li, and Hsiangpang Li. Achieving slc performance with mlc flash memory. In *DAC*. ACM, 2015.
[7] Chris Reynolds. Tantalum capacitor technology : Options for high-temperature and harsh-environment applications. *Power Electronics Magazine*, 2017.
[8] Dongwook Kim and Sooyong Kang. Dual region write buffering: making large-scale nonvolatile buffer using small capacitor in ssd. In *SAC*. ACM, 2015.
[9] Rajat Kateja, Anirudh Badam, and Sriram Govindan. Viyojit: Decoupling battery and dram capacities for battery-backed dram. In *ISCA*. ACM, 2017.
[10] Woojin. Choi, Prasad. Enjeti, and J. W. Howze. Fuel cell powered ups systems: design considerations. In *PESC*. IEEE, 2003.
[11] Yang Hu, Hong Jiang, Dan Feng, Lei Tian, Hao Luo, and Shuping Zhang. Performance impact and interplay of ssd parallelism through advanced commands, allocation strategy and data granularity. In *ICS*. ACM, 2011.
[12] Mingchang Yang, Yuming Chang, Chewei Tsao, Pochun Huang, Yuanhao Chang, and Teiwei Kuo. Garbage collection and wear leveling for flash memory: Past and future. In *SMARTCOMP*. IEEE, 2014.
[13] Dushyanth Narayanan, Eno Thereska, Austin Donnelly, Sameh Elnikety, and Antony Rowstron. Migrating server storage to ssds: analysis of tradeoffs. In *EuroSys*. ACM, 2009.
[14] UMassTraceRepository. OLTP Application I/O Traces. http://www.memblaze.com/en/index.php?c=ajax&a=test1.
[15] Microsoft. DiskMon. https://docs.microsoft.com/zh-cn/sysinternals/downloads/diskmon.
[16] Congming Gao, Liang Shi, Mengying Zhao, Chun Jason Xue, Kaijie Wu, and Edwin H. M. Sha. Exploiting parallelism in i/o scheduling for access conflict minimization in flash-based solid state drives. In *MSST*. IEEE, 2014.