**ISBN** 978-981-14-4787-7
**Proceedings of 2020 the 10th International Workshop on Computer Science and Engineering (WCSE 2020)**
*Shanghai, 19-21 June, 2020, pp. 307-313*
doi: 10.18178/wcse.2020.06.046

# A Dual-Mode on-Demand Page-Level Flash Translation Layer for Large Scale NAND Flash Memory

Jing Chen, Weijie Lin, Shiyue Cao and Zhiyuan Cheng +

College of Information Science & Electronic Engineering, Provincial Key Laboratory of Micro-Nano Electronics and Smart System, Zhejiang University, Hangzhou, China

**Abstract.** With the rapid development of NAND flash technique and NAND-based solid-state devices (flash SSDs), whose capacity has doubled roughly every two years, the mapping table for page-level flash translation layers (FTLs) has become too large to be entirely stored in RAM in the engineering application. In this work, we have proposed a dual-mode on-demand page-level flash translation layer scheme (named DOPFTL hereafter), where extent mapping technology is adopted to compress mappings in a trace-adaptive way. Besides, two improvements are introduced to locate or update mappings for a read/update request with less overhead. One improvement is to add a field of cached translation page address (CTPA) in the global translation directory (GTD), which results in a benefit of time-saving in locating mappings. The other improvement is to append a new entry of LPN-PPN-LEN in compressed mode for updates, which results in a benefit of avoiding split or merge overhead. With these improvements, we have conducted extensive trace-driven evaluations of DOPFTL and compared it with the other start-of-the-art FTL schemes. Experimental results show that DOPFTL can achieve obvious improvement in cache hit rate than the conventional DFTL, which is a classical on-demand page-level FTL. Besides, DOPFTL can achieve an average cache hit rate of up to 0.89, close to that of TPFTL which is a demand-based page-level FTL with a translation page-level caching mechanism. Besides, DOPFTL can achieve obvious reduction in time overhead when compared with DFTL and TPFTL.

**Keywords:** SSD, NAND Flash Memory, Flash Translation Layer (FTL), Address Translation, Mapping Cache.

## 1. Introduction

The storage industry has changed a lot, like cloud storage [1, 2], flash memory and so on. NAND flash memory has undergone plenty of breakthroughs since its invention by Dr Fujio of Toshiba Corporation [3]. Over the past thirty years, NAND flash techniques not only have evolved from 2D (two dimensional) to 3D (three dimensional) [4], but also have increased the number of bits stored in each memory cell from SLC to MLC [5], TLC [6], and finally QLC [7]. Although these technologies enable NAND flash memory to have a huge capacity, NAND flash memory has limited endurance because of erasing. NAND flash memory has a characteristic constraint of erasing before writing, so out-of-place updates [12] have been adopted in flash memory in order to reduce the number of erasing. To support out-of-place updates, flash translation layers (FTLs) have been designed to map physical page number (PPN) with corresponding logical page number (LPN) [8]. Depending on the size of management granularity of NAND flash memory, FTLs can be classified into three types: page-level mapping, block-level mapping, and hybrid mapping. The page-level mapping has better performance than block-level mapping and hybrid mapping [9-11] but it has a distinct disadvantage of using a very large mapping table for large-capacity flash memory. The mapping table is stored in a RAM, which is expensive.

+ Corresponding author.
*E-mail address:* zycheng@zju.edu.cn.

In order to reduce the RAM requirement, Aayush Gupta et al. [12] proposed a classical on-demand page-level FTL (DFTL), which takes advantage of temporal locality. However, it does not consider spatial locality and its cache hit rate of mappings is not high. Recently, Yu Zhou et al. [13] have proposed a new demand-based page-level FTL with a translation page-level caching mechanism (TPFTL), which is considered as an improvement over DFTL. It introduces a prefetching technology to improve cache hit ratio of mappings. However, TPFTL does not consider the split or merge overhead of extent mapping and the disorder way of cached mappings, which can cause distinct time overhead.

In this paper, we have proposed a dual-mode on-demand page-level FTL (DOPFTL). A comparison between the key features of DOPFTL, TPFTL and DFTL are listed in Table 1.

Table. 1: Comparison of FTL schemes for improving cache hit rate and reducing time overhead

| | DOPFTL (this work) | TPFTL [13] | DFTL [12] |
|---|---|---|---|
| Improving cache hit rate | 1. spatial locality by employing dual-mode (original mode and compressed mode) in mapping cache, where extent mapping technology is used in compressed mode;<br>2.trace-adaptive in the space ratio between original mode and compressed mode by fine-granularity cache management | prefetch technology to load mappings to be requested in the near future | temporal locality only |
| Reducing time overhead | 1.Add a new field in the GTD to locate the cached translation page;<br>2. Add an appending entry in the compressed mode to avoid split or merge overhead | using LRU lists for hot mappings | temporal locality only |

# 2. Design of DOPFTL

The overall architecture of the novel FTL scheme proposed in this work is shown in Fig. 1. This DOFTL is an on-demand page-level FTL, and it partitions a whole flash memory into data blocks and translation blocks which are used to store the LPN-PPN mapping table.

## 2.1. Spatial-aware Mapping Scheme by Employing a Dual-Mode in Mapping Cache

In this DOPFTL, we adopt a dual-mode to store frequently requested translation pages in mapping cache: an original mode and a compressed mode. The structure of the dual-mode is shown in Fig. 2. In the compressed mode, we use extent mapping technology of LPN-PPN-LEN to help reduce RAM footprint. In the original mode, PPNs are stored in the ascending order of LPNs. When the PPNs in a translation page can be compressed to no more than the maximum number of LPN-PPN-LEN, 125 in our experiments, it can be stored in compressed mode. Otherwise, it will be stored in original mode, where PPNs are stored in the ascending order of LPNs and the number of PPNs depends on the size of the corresponding translation page.

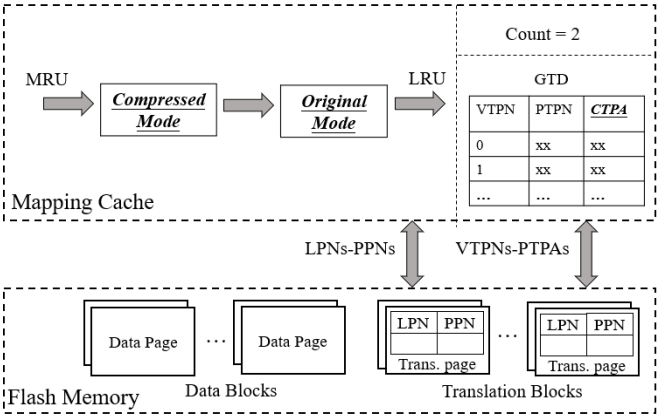


Fig. 1: Overall architecture of DOPFTL scheme proposed in this work, which features (1) a dual-mode operation in mapping cache to improve cache hit rate, (2) addition of a new field of CTPA in GTD for time-saving in locating mappings.

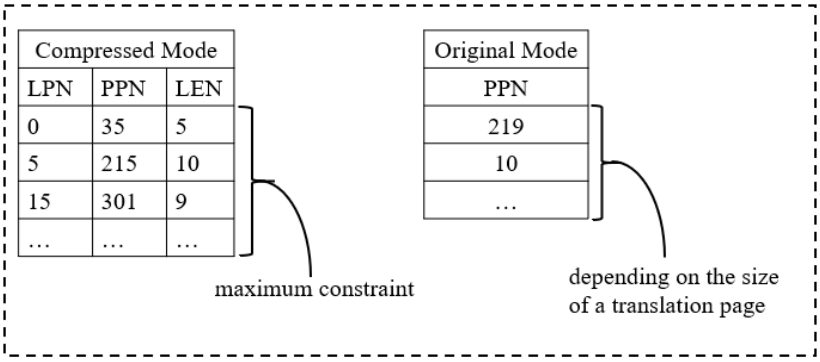


Fig. 2: Dual-mode: compressed mode for storing LPN-PPN-LEN entries where we set an upper limit of 125 entries in this experiment; and original mode storing PPNs in the ascending order of LPNs, whose size depends on the size of the corresponding translation page.

## 2.2. Trace-Adaptive in Space Ratio Between Original Mode and Compressed Mode by Using Fine-Granularity Cache Management

Fine-granularity cache management is adopted in this DOPFTL to make it adaptive to various traces. As shown in Fig. 3, the space for mapping cache is divided into cache blocks whose size is equal to a compressed mode. An original mode consists of four cache blocks, and a compressed mode consists of one cache block. There are two reasons why we set a compression ratio of 4: (1) it is a trade-off between the DRAM overhead and the transition overhead; (2) other compression ratios can cause fragmentation in cache. In this DOPFTL, compressed mode and original mode share the same inventory of cache blocks. Three additional cache blocks are needed for a compressed mode to change into an original mode, while three cache blocks can be released when an original mode change to a compressed mode. In order to avoid frequent transitions, original mode is not allowed to change into compressed mode during the request stage. However, the transition to compressed mode is expected to happen during the evicting stage.

In fact, different traces need different DRAM space ratios between original mode and compressed mode. With the above fine-granularity cache management, the space ratio between original mode and compressed mode can be self-adjustive. In comparison, if the space ratio between original mode and compressed mode is fixed, it will need more cache spaces to meet all the needs for every different trace.

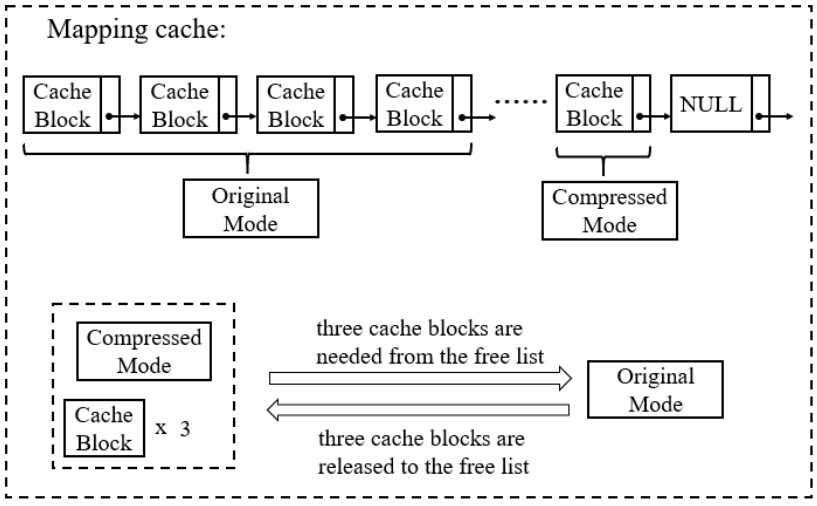


Fig.3: Fine-granularity cache management where the mapping cache is divided into a number of cache blocks with the size of a compressed mode. An original mode can be bi-directionally exchanged with a compressed mode.

## 2.3. Avoiding Split or Merge Overhead in Updating Mappings by Appending a New Entry in Compressed Mode for Updates

The adoption of extent mapping technology in compressed mode in Fig. 2 helps to reduce the RAM requirement. However, it introduces split or merge overhead, as illustrated in Fig. 4(a). To solve this issue, we append a new entry for updates at the end of a compressed mode in our DOPFTL, as shown in Fig. 4(b).

With this appending entry, the latest PPN can be gained by looking up from bottom to top. An example is illustrated in Fig. 4(b), the corresponding PPN for LPN of 26 is 400, not 221 (20 minus 6 and plus 215).

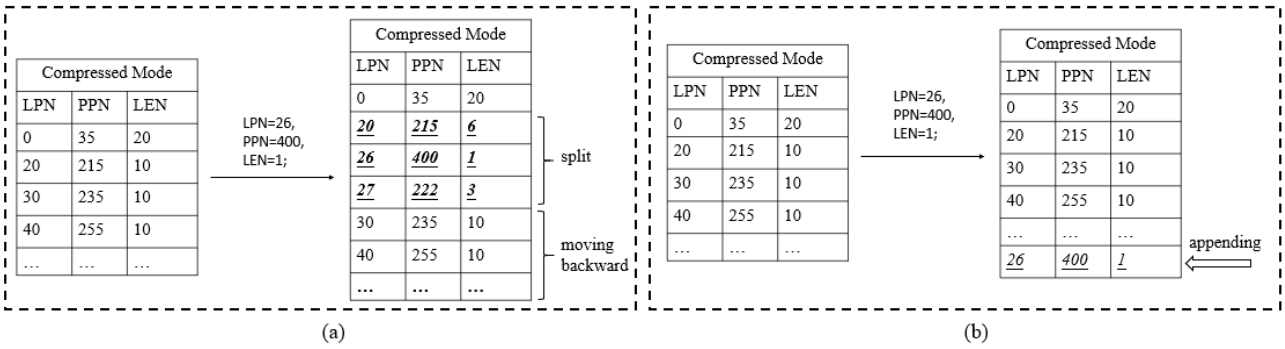


Fig. 4: (a) Example of updating one LPN-PPN mapping in compressed mode by adding a new entry of 26-400-1 in conventional approach, which introduces the split overhead and makes mappings move backward. (b) Approach used in DOPFTL with an appending entry.

## 2.4. Time-saving in Locating Mappings by Adding a Field of Cached Translation Page Address (CTPA) in Global Translation Directory (GTD)

The latest FTLs in the literature, such as TPFTL, do not consider cache index overhead. To reduce the cache index overhead, particularly with fine-granularity cache management, we add a field of cached translation page address (CTPA) into the global translation directory (GTD). An example is illustrated in Fig. 5. Assuming that the size of a NAND flash memory is 2TB, the corresponding mapping cache is 32MB and a cache block is 1KB, the total space taken by CTPA is 1MB. The cached translation page of VTPN of 103 can be located by CTPA of 0x078. Otherwise, before the translation page of VTPN of 103 would be found, there are 500 nodes to be checked whether one of them is the requested one. Without CTPA, the time complexity would be $O(N)$ and N is the number of total nodes in the LRU list. It can be reduced to $O(1)$ with the addition of CTPA.

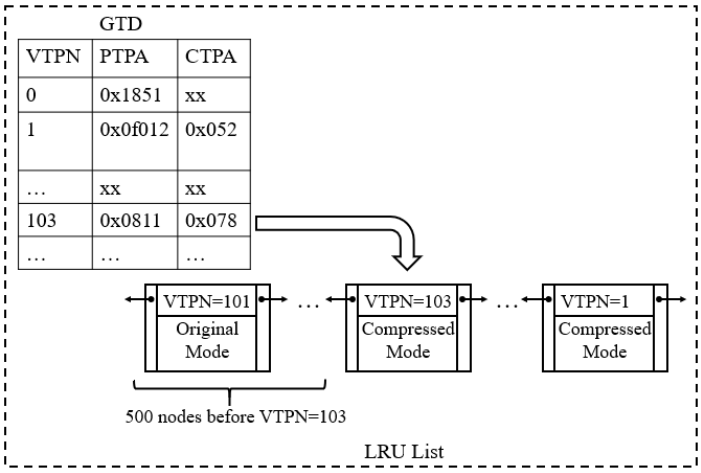


Fig. 5: Example of a new field of CTPA in GTD where the cached translation page of VTPN of 103 can be located by CTPA of 0x78.

# 3. Evaluation

To evaluate this new DOPFTL scheme, we conduct two sets of experiments. Section 3.1 shows the influence of trace-adaptive approach and CTPA on DOPFTL, and section 3.2 shows comparisons between DFTL, TPFTL, "reference-FTL", and DOPFTL. Here, "reference-FTL" refers to a special page-level FTL used in this experiment where sufficient DRAM space is provided to store the entire mapping table and thus has a cache hit rate of 1.

Four traces have been used for evaluation in this work as shown in Table.2, and flash page size is 4KB and flash block size takes 168,600KB in our experiments. Time overhead ratio and cache hit rate are two performance criterion to evaluate DOPFTL.

Table. 2: Traces of PC Test, RocksDB, Dorado, and Cassandra

| | PC Test | Cassandra | RocksDB | Dorado |
|---|---|---|---|---|
| Read Ratio | 10% | 95% | 97% | 36% |
| Total Number | 3810991 | 183742235 | 21225458 | 5355904 |
| Address Space | 1.2TB | 1.2TB | 1.2TB | 1.2TB |

## 3.1. Influence of a Trace-Adaptive Approach and a New Field of CTPA on DOPFTL

(1) Influence of a trace-adaptive approach on DOPFTL

A fixed space ratio between original mode and compressed mode may be not suitable to all traces and can cause the waste of cache space for mappings, but the trace-adaptive approach used in this work can address this problem well. In Fig. 6, we compare the cache hit rates between two sets of experiments: one with a fixed space ratio of 0.4, and one with a trace-adaptive space ratio used in the DOPFTL. The experiment shows that the cache hit rates of DOPFTL are higher than those with a fixed space ratio of 0.4. Based on our observation, the space ratio between compressed mode and original mode was variable and different traces need different space ratio. Therefore, the introduction of a trace-adaptive approach helps to effectively use cache space.

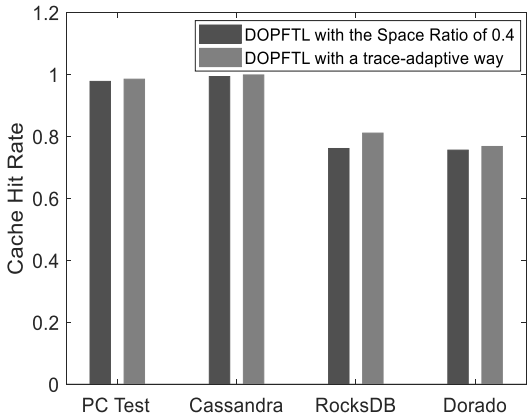


Fig. 6: Cache hit rates of DOPFTL with a trace-adaptive way and DOPFTL with the space ratio fixed to 0.4 for four different traces.

(2) Influence of a new field of CTPA on DOPFTL

As shown in Fig. 7, DOPFTL with CTPA has obviously lower time overhead than DOPFTL without CTPA for all traces. There is the most time overhead reduction by CTPA under the Dorado case since Dorado is the most random trace compared with other three traces. However, time overhead of DOPFTL with CTPA has the least reduction under the RocksDB case since hot mappings are always close to the head of the LRU list, which means that requested PPNs can be soon located. The new field of CTPA can avoid heavy location of requested PPNs in mapping cache and thus DOPFTL have little time overhead.

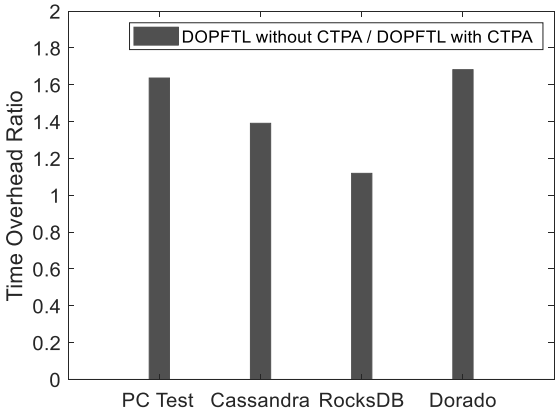


Fig. 7: Time overhead ratios of DOPFTL without CTPA, relative to DOPFTL with CTPA for four different traces.

## 3.2. Comparison Among DFTL, TPFTL, Reference-FTL and DOPFTL

(1) Cache Hit Rate

As shown in Fig. 8, DOPFTL has relatively high cache hit rates under different traces. Compared with DFTL, DOPFTL has constantly higher cache hit rates since DFTL does not consider spatial locality and a way to compress mappings. Besides, DOPFTL has better cache hit rates than TPFTL under the PC test and Dorado cases since DOPFTL takes advantage of spatial locality and always load mappings into cache space in advance. However, DOPFTL does not have a better cache hit rate than TPFTL under some traces like Dorado because DOPFTL can load cold mappings into cache space and do not get rid of them in time.

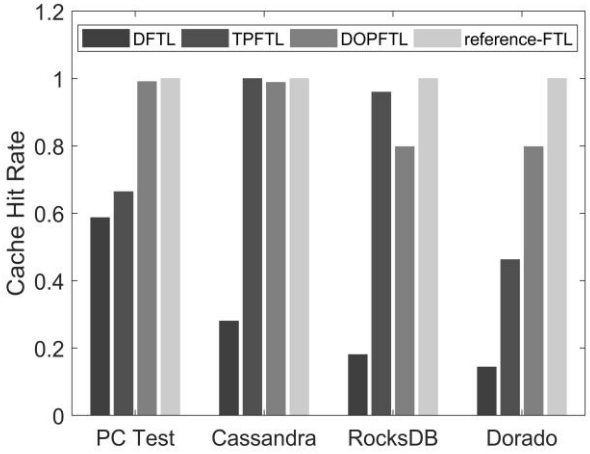


Fig. 8: Cache hit rates of four different FTLs (DFTL, TPFTL, DOPFTL and reference-FTL) for four different traces.

(2) Time Overhead Ratio

The results are plotted in Fig. 9, in which time overhead ratios of DFTL, TPFTL and DOPFTL are relative to those of reference-FTL for four different traces. The figure shows that the time overhead ratios of DOPFTL are the least ones. DOPFTL has obviously lower time overhead than DFTL because its cache hit rates are much higher than DFTL in Fig. 8. Besides, DOPFTL makes a further improvement on time overhead compared with TPFTL because DOPFTL can directly locate requested PPNs by CTPA and avoid split or merge overhead by appending a new entry in a compressed mode. Hence, DOPFTL has less time overhead to locate/update LPN-PPN mappings in mapping cache than DFTL and TPFTL.

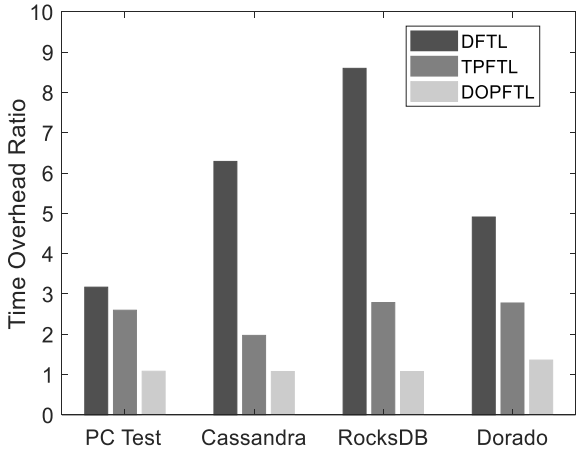


Fig. 9: Time overhead ratios of DFTL, TPFTL, and DOPFTL, relative to those of reference-FTL for four different traces.

# 4. Conclusion

We have designed a dual-mode on-demand page-level mapping FTL named DOPFTL in this paper. DOPFTL has utilized dual modes (original mode and compressed mode) in a trace-adaptive way. Besides, we append a new entry of LPN-PPN-LEN in compressed mode for updates, and add a new field of cached

translation page address (CTPA) into the global translation directory (GTD). Extensive tests have been conducted to verify this DOPFTL scheme. Results have shown that DOPFTL has higher cache hit rates for various traces and less time overhead to locate/update mappings in mapping cache than the conventional FTLs.

## 5. References


[1] C. Long and Z. Qing, "Forensic Analysis to China's Cloud Storage Services," *International Journal of Machine Learning and Computing*, vol. 5, no. 6, p. 467, 2015.

[2] N. Ousirimaneechai and S. Sinthupinyo, "Extraction of trend keywords and stop words from thai facebook pages using character n-Grams," *International Journal of Machine Learning and Computing*, vol. 8, no. 6, pp. 589-594, 2018.

[3] F. Masuoka, M. Asano, H. Iwahashi, T. Komuro, and S. Tanaka, "A 256K flash EEPROM using triple polysilicon technology," in 1985 IEEE International Solid-State Circuits Conference. Digest of Technical Papers, 1985, vol. 28: IEEE, pp. 168-169.

[4] K. Parat and C. Dennison, "A floating gate based 3D NAND technology with CMOS under array," in 2015 IEEE International Electron Devices Meeting (IEDM), 2015: IEEE, pp. 3.3. 1-3.3. 4.

[5] C. S. Bill and S. S. Haddad, "Multiple bits per-cell flash EEPROM capable of concurrently programming and verifying memory cells and reference cells," ed: Google Patents, 1998.

[6] G. Marotta et al., "A 3bit/cell 32Gb NAND flash memory at 34nm with 6MB/s program throughput and with dynamic 2b/cell blocks configuration mode for a program throughput increase up to 13MB/s," in 2010 IEEE International Solid-State Circuits Conference-(ISSCC), 2010: IEEE, pp. 444-445.

[7] S. Liu and X. Zou, "QLC NAND study and enhanced Gray coding methods for sixteen-level-based program algorithms," Microelectronics journal, vol. 66, pp. 58-66, 2017.

[8] J.-Y. Shin et al., "FTL design exploration in reconfigurable high-performance SSD for server applications," in Proceedings of the 23rd international conference on Supercomputing, 2009: ACM, pp. 338-349.

[9] D. Liu, Y. Wang, Z. Qin, Z. Shao, and Y. Guan, "A space reuse strategy for flash translation layers in SLC NAND flash memory storage systems," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 20, no. 6, pp. 1094-1107, 2011.

[10] L.-P. Chang, T.-Y. Chou, and L.-C. Huang, "An adaptive, low-cost wear-leveling algorithm for multichannel solid-state disks," ACM Transactions on Embedded Computing Systems (TECS), vol. 13, no. 3, p. 55, 2013.

[11] Y. Guan, G. Wang, Y. Wang, R. Chen, and Z. Shao, "BLog: block-level log-block management for NAND flash memory storage systems," ACM SIGPLAN Notices, vol. 48, no. 5, pp. 111-120, 2013.

[12] A. Gupta, Y. Kim, and B. Urgaonkar, DFTL: a flash translation layer employing demand-based selective caching of page-level address mappings (no. 3). ACM, 2009.

[13] Y. Zhou, F. Wu, P. Huang, X. He, C. Xie, and J. Zhou, "Understanding and alleviating the impact of the flash address translation on solid state devices," ACM Transactions on Storage (TOS), vol. 13, no. 2, p. 14, 2017.