Advanced Science and Technology Letters
Vol.51 (CES-CUBE 2014), pp.24-27
http://dx.doi.org/10.14257/astl.2014.51.06

Efficient IO and Layout Management Issue of Filesystem for Flash device

Seung-Ho Lim

Department of Digital Information Engineering
Hankuk University of Foreign Studies
slim@hufs.ac.kr

Abstract. Inside the Flash storage device, Flash Translation Layer (FTL) remaps logical address to physical address to hide physical limitation of Flash memory cells. Due to the address translation, intentional logical separation of file system's layout does not directly applied to physical separation. As a result, the separate-intended file systems' requests are mixed in physical location, which degrades write throughput, as well as flash's lifecycle. In this paper, we propose virtually separable Block management scheme for Flash storage system by introducing new common command interface and separable Block management in FTL. The experimental results show that the proposed scheme increases IO performance, as well as reduces flash-internal overhead.

Keywords: Flash Memory, File system, FTL, Virtually separable Block management.

1 Introduction

There are two main physical limits for NAND flash memory. First, cell-erase operation should be preceded by cell-write operation, which means that the data cannot be written in same place without erasing it. Second, the base unit of erase operation is 'Block', whereas base unit of write operation is 'Page', where block is much larger than page. Actually, Block is composed of several Pages. The physical limitation of NAND flash memory is covered by special firmware called Flash Translation Layer (FTL) [1][2]. File systems can treat Flash device just like usual storage media with the logical block address supported by FTL, and FTL hides internal mapping information and management schemes from host file systems.

Since file system uses Flash device as a view of logical address, there is a mismatch for view point between logical block and physical block. As a result, traditional file system architecture and layout design is little effective for Flash-based storage. File system's data are mixed in physical location for Flash memory-based storage system even though they are logically separated by file system on purpose, as described in Figure 1. The unintentional physical mix for logically separated data from file system degrades IO bandwidth as well as latency, significantly.

In this paper, we propose virtually separable Block management scheme for Flash storage system. For the separable Block management system, we propose common

ISSN: 2287-1233 ASTL

command interface between file systems and underline flash device to support the separation. The virtual separation can make file system's read/write requests sequentially, which leads bandwidth increase. In addition, the GC efficiency of FTL gets better, which decrease flash-internal overhead and increase lifetime of Flash device. The experimental results show that the proposed scheme increases IO performance, as well as reduces flash-internal overhead.

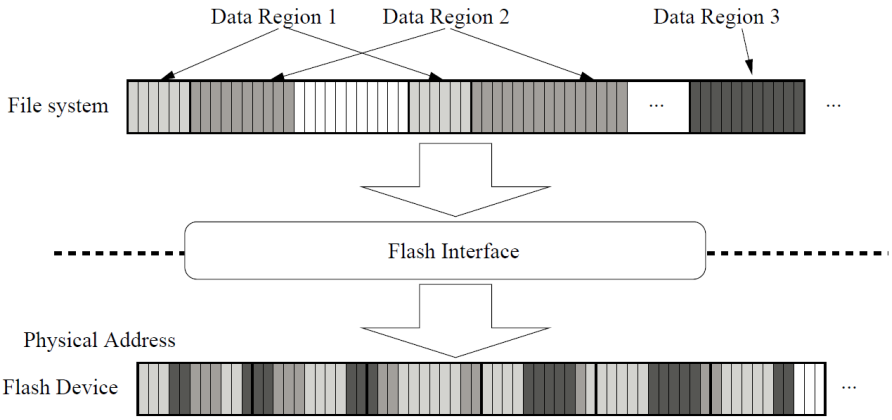

Fig. 1. File System's IOs are mixed within Flash Blocks

2 Related Work

There is previous work that considers request characteristics and patterns for file system and apples it to flash storage system and FTL development. Chang et al separate metadata requests and user data requests based on the knowledge of the layout of file systems and metadata filter to reduce file systems performance degradation [4], where different FTL management algorithms are applied to each metadata region and user data region, explicitly. It is most related work to the work of this paper, in that they tried to separate metadata and user data in Flash storage device. However, our approach is different from [4] in that there are only one FTL in our system, whereas there are two FTLs in one system. In addition, our system gives much more common design methodology for separating file system's overall data. According to our design, the several file system's region can be separated virtually in Flash device without any addition filtering algorithm, and all the separated regions are managed by only one FTL, which is efficient for manufacturing Flash storage device. From next Section, the detailed system design issues are described.

3 Virtually Separable Flash

The unintentional physical mix for logically separated data from file system degrades IO bandwidth as well as latency, significantly. In this paper, we propose virtually separable Block management scheme for Flash storage system. For the separable Block management system, we propose common command interface between file systems and underline flash device to support the separation. In the proposed system, file system has identifier, i.e., ID, to identify data type. For instance, file system having metadata, user data, and journal data has three types of data to be stored in separated region, the IDs are allocated for each type of data, and the file system makes read/write commands with the logical address, size, and its ID. In the flash devices, the flash Blocks are virtually separated according to the data types. For instance, data and metadata, and journal data are assembled under the separated flash Blocks virtually, where the 'virtually' means that the Blocks are not necessarily consecutive in physical position. When write command arrives, FTL, which makes overall management of flash devices, writes the arriving data to the relevant Block according to its ID. The ID-based Block management Flash storage system is described in Figure 2.

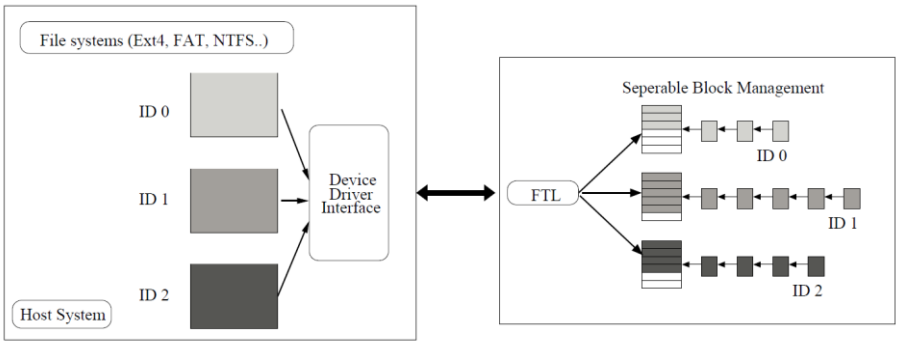

Fig. 2. The File System's IOs are managed and stored in separable Flash Blocks according to their types

As a result, Blocks having same ID are virtually connected each other, while Blocks having different ID are virtually separated. This virtual separation can make file system's read/write requests sequentially, which leads bandwidth increase. In addition, the GC efficiency of FTL gets better, which decrease flash-internal overhead and increase lifetime of Flash device. The experimental results show that the proposed scheme increases IO performance, as well as reduces flash-internal overhead.

4 Implementation and Analysis

We have evaluated the proposed scheme with well-known file system benchmark, called IOzone [5]. The IOzone file system benchmark generates and measures bandwidth for file system operations. The bandwidth of write and random write operation is estimated as request size varies from 4KB to 64KB. The experimental results are shown in Table 1 for sequential write and random write requests, respectively. For the sequential write requests, the proposed scheme gives slightly better bandwidth than legacy system. However, the proposed scheme outperforms legacy system for random write requests greatly.

Table 1. Experimental Results for IOzone Benchmark for sequential write and random write, respectively, as request size increases from 4KB to 64KB.

Benchmarks		4KB	8KB	16KB	32KB
Sequential Write	Separable	1843	1983	2073	2152
	non-Separable	1617	1924	2027	2038
Random Write	Separable	1834	1977	2071	2152
	non-Separable	1137	1331	1324	1531

5 Conclusion

This paper proposes efficient separable management scheme for Flash storage system by introducing new common command interface between file systems and underline flash device. In the proposed system, file system has identifier, i.e., ID, to identify data type, where the IDs are assigned for each type of data, and the file system makes read/write commands with the corresponding ID. In the flash devices, the flash Blocks having same ID are virtually connected, where the $virtually$ means that the Blocks are not necessarily consecutive in physical position. The experimental results show that the proposed scheme increases write throughput, as well as reduces flash-internal overhead by making separable Block management within Flash device.

References

1. Ban, A.: Flash file system, United States Patent. no.5, 404, 485, 1995.
2. Intel Corporation, “Understanding the flash translation layer (FTL) specification”, http://developer.intel.com/.
3. Micron, “Garage Collection in Single-Level Cell NAND Flash Memory”, Technical Note, TN-2960.
4. Chang, Y. H., Wu, P. L., Kuo, T. W. and Hung, S. H.: An adaptive file-system-oriented FTL mechanism for flash-memory storage systems. ACM Transactions on Embedded Computing SYstems, Vol. 11, Issue 1, 2012.
5. Norcott, W.: IOzone for Filesystem performance Benchmarking. http://www.iozone.org/, 2008.

