World Academy of Science, Engineering and Technology
International Journal of Electronics and Communication Engineering
Vol:9, No:12, 2015

Exploring SSD Suitable Allocation Schemes Incompliance with Workload Patterns

Jae Young Park, Hwansu Jung, Jong Tae Kim

***Abstract*—**In the Solid-State-Drive (SSD) performance, whether the data has been well parallelized is an important factor. SSD parallelization is affected by allocation scheme and it is directly connected to SSD performance. There are dynamic allocation and static allocation in representative allocation schemes. Dynamic allocation is more adaptive in exploiting write operation parallelism, while static allocation is better in read operation parallelism. Therefore, it is hard to select the appropriate allocation scheme when the workload is mixed read and write operations. We simulated conditions on a few mixed data patterns and analyzed the results to help the right choice for better performance. As the results, if data arrival interval is long enough prior operations to be finished and continuous read intensive data environment static allocation is more suitable. Dynamic allocation performs the best on write performance and random data patterns.

***Keywords*—**Dynamic allocation, NAND Flash based SSD, SSD parallelism, static allocation.

I. Introduction

THE performance of SSD depends on both hardware like architecture, interface speed, and NAND Flash performance and software called Flash Translation Layer (FTL). Because FTL affects greatly the SSD's performance, a lot of researches have been studied FTL as an important research theme [1]. In initial FTL studies, they focused on reducing garbage collection overhead by considering block mapping and hybrid mapping. In recent researches, they make efforts to increase the efficiency of multi-level parallelism, because almost SSDs have multi-channel architecture as page mapping method. [2]-[5].

In order to increase the efficiency of multi-level parallelism, parallel processing should be applied to as many commands as possible by allocating addresses. The allocation scheme determines the method of address allocation. There are two allocation schemes. The first one is the dynamic allocation that write the address the anywhere of the total SSD. Another one is the static allocation that can be written at specified addresses. [6]. Read operation limits available addresses, because it has specified addresses. On the other hand, write operation has no specified addresses, so it can be written the anywhere of the total SSD. Therefore, dynamic allocation method is more efficient with no delay due to its allocation flexibility of addresses. However, the write-oriented allocation scheme can cause reading performance decrease, because there will be delay until previous operation execution when read operation is allocated at the addresses that is not used in parallel processing. Although incompatible characteristics between dynamic allocation and static allocation, dynamic allocation is more popular than another one, because write operation is much slower than read operation, so increasing write performance helps improving total SSD's performance. On the other hand, static allocation can be more efficient for read operation-oriented workload that writes once reads many times. However, the researches on SSD's performance improvement using static allocation have not been studied enough.

The efficiency of allocation scheme is affected significantly workload pattern based on SSD's features. Therefore, allocation scheme is used suitably for your situations, because there is no absolute priority between two allocation schemes.

In this paper, we simulate various workload patterns, and then analyzed simulated results. As a result of the analyses, we figure out suitable allocation scheme for each workload pattern. In Section II, we describe the parallelism of the SSD and allocation schemes. In Section III, we analyze the simulation results. Finally, the result analyses and considerations are summarized in Section IV.

II. Address Allocation

The SSD has several channels, and NAND Flash package in each channel shares channel bus, as shown in Fig. 1, NAND Flash package has multi-level parallel architecture with one die for capacity and performance. A die is a minimum unit of parallel processing. To process operations like read, write, and clear in parallel, the operations are located at each different die (except for advanced command). If all of three read commands are located at same die as shown Fig. 1, there will be no parallelization. In other case that the commands are in each different channel, channel level processing develop. In another case that the commands are in same channel but in different dies, it runs similarly to internal parallelism. Therefore, parallelization priority is in channel-die-package order.

To enhance the read performance, it should be parallel like channel level parallelism or internal parallelism in Fig. 1 but it may be not the optimized performance in view of overall SSD performance. Fig. 2 shows the difference of time depending on the dynamic allocation and static allocation. Dynamic allocation assigns any executable die without restraint when the dies occupied by other operations like Fig. 2. Static allocation waits to end other operations writing a predetermined channel, chip, die, and plane. In the case of reading, dynamic allocation

Jae Young Park, Hwansu Jung and Jong Tae Kim are with the Department of Electrical and Computer Engineering, Sungkyunkwan University, Suwon, South Korea (e-mail: jyp8389@gmail.com, chs2756@skku.edu, jtkim@skku.edu).

parallelized only two read operations because two write operations are assigned to one die. On the other hand, static allocation parallelized all read operations. Thus, in the write operation, dynamic allocation benefits the amount of time 'A' while, in the read operation, loss the amount of time 'B' compared to static allocation. Considering only the workload of example, dynamic allocation outperforms static allocation because the amount of time 'A' is greater than 'B'. However, since read operations can be repeated, time benefit 'B' can be occurred also repeatedly until the data is modified or erased.

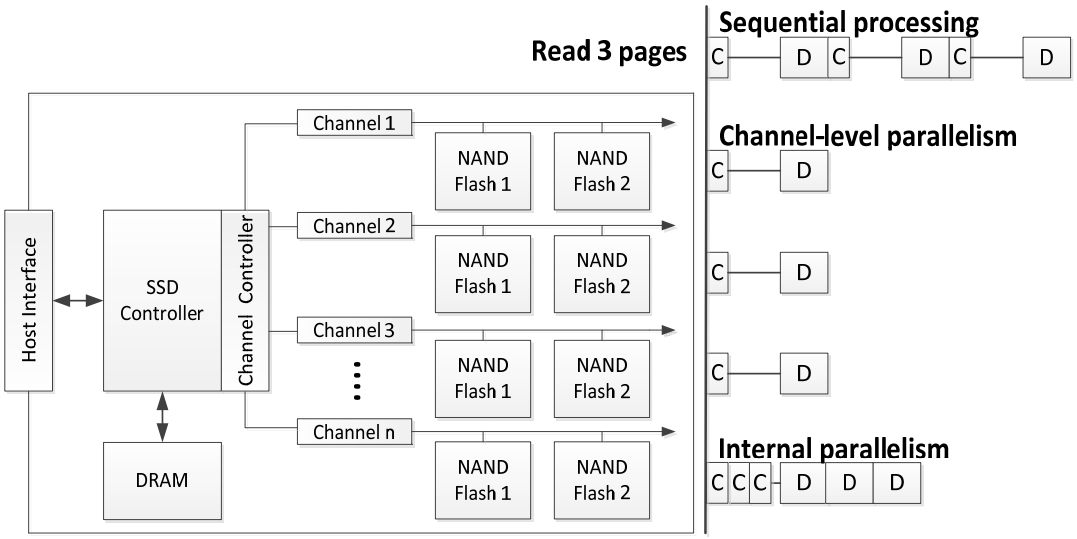

Fig. 1 General SSD architecture and parallel processing examples

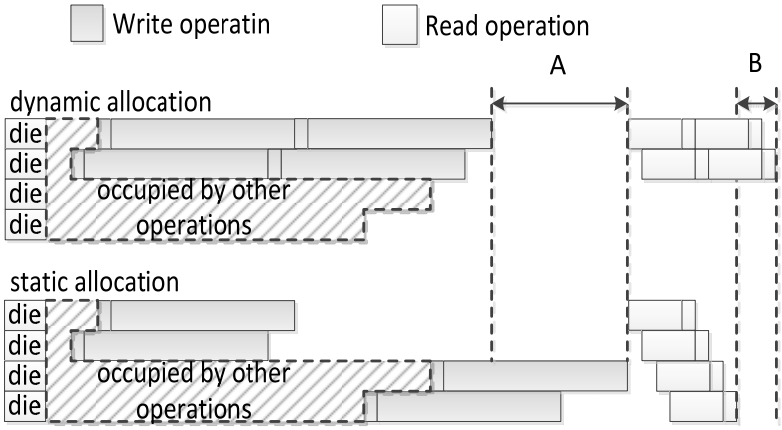

Fig. 2 difference of time by allocation schemes

III. Analysis to Find Suitable Allocation Scheme

The advantage which is obtained by the dynamic allocation is a reliable current benefits and the advantage obtained by static allocation is uncertain future benefits. Therefore, static allocation is used with caution. In this paper, we were assumed to situations where there is a difficulty of finding an appropriate allocation scheme and generated the workloads. The simulation is used the generated workload and we analyzed the simulation result. Simulation is used FlashSim and SSD architecture is 8-2-2(channel-package-die) and NAND Flash performance of operations are one page read 100us and one page write 1000us [7].

A. Not Occupied by Other Operations

When there is no interference due to other operations because data arrival interval is long enough prior operations to be finished, there are no disadvantage both dynamic allocation and static allocation caused by writing. In the case of reading also, all of operations will perform optimally through optimal parallelization. Therefore, dynamic allocation and static allocation exhibit the same results as Fig. 3 Syn A if dies are not occupied by other operations.

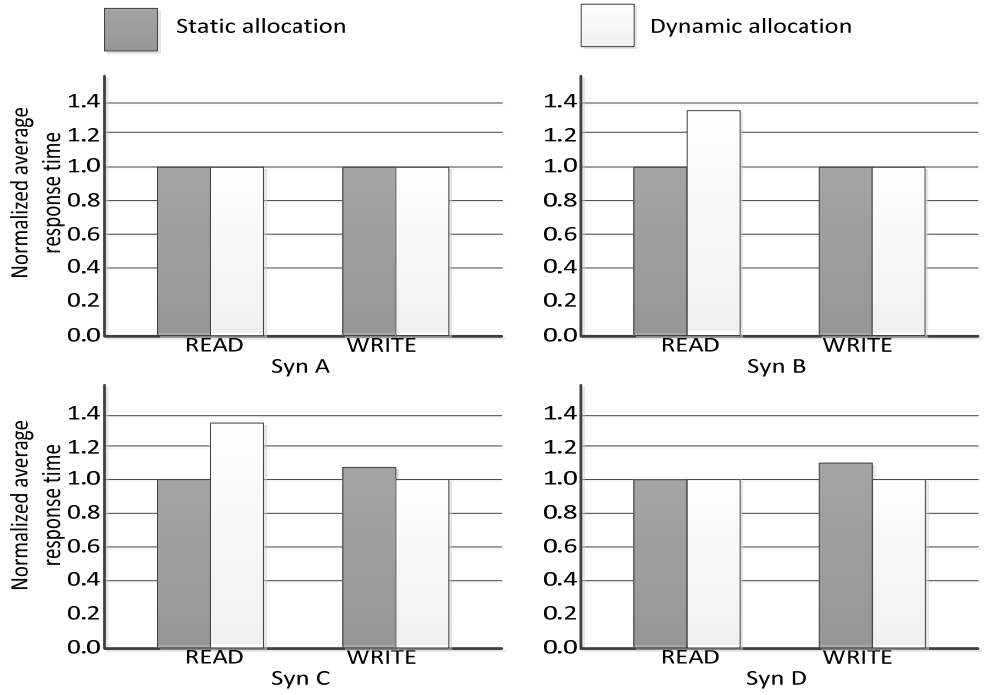

Fig. 3 Normalized average response time corresponding to the data pattern and different allocation schemes

B. Modified Few Pages with Continuous Data

If the dies which will be assigned the present operations are not occupied, the performance of dynamic allocation and static allocation are the same. If a few pages are modified, the performance is changed due to allocate schemes. Static allocation assigns the new physical address to keep parallelism continuously even though the data is modified. Dynamic allocation, on the other hand, assigns the new physical address at random depending on present SSD state and it makes no longer parallel. The disadvantage caused by not been parallelized reduces the read performance. Fig. 3 Syn B is the example and it shows the disadvantage is almost 1.5 times. The specific disadvantage rate depends on how much not to be parallelized. Static allocation is suitable for this condition except wear-leveling problem because read advantage exists and write disadvantage does not. Whereas, read advantage does not exists and write disadvantage does in dynamic allocation.

C. Continuous and Large Amount of Data

The situation is the same as the example of Fig. 2. Current operations are delayed caused by dies that are occupied by other operations. The simulation results in Syn C of Fig. 3 show that read response time of static allocation is shorter and write response time is longer comparing with dynamic allocation. Dynamic allocation can assign a logical page to any physical page whereas static allocation is restricted as per static allocation rule. However, the flexibility of allocation causes to break read parallelization. Thus, write response time of dynamic allocation is shorter than static allocation and read response time is longer. In this condition, there are both advantage and disadvantage, for that reason, suitable allocation scheme will depend on the ration of the read and write. If the read operation is dominant static allocation will be suitable. Otherwise, dynamic allocation will be suitable.

D. Random and Large Amount of Data

As the result Syn D of Fig. 3, write response time of static allocation is 1.15 times slower and read response time is the same. This is the same as the example of Syn C, but it is only different that reads data are formed at random. Random data is not called continuous address despite waiting to write specific area. The data is not parallelized and there is no read response time advantage with static allocation. Therefore, dynamic allocation is suitable in this condition.

IV. Conclusion

Choosing suitable allocation scheme is an import factor in determining the total SSD's performance. However, there have not been studies on guideline for choosing appropriate allocation scheme between dynamic allocation and static allocation. In this paper, we set the situations that were difficult to determine the allocation method. We simulated the situations, and then analyzed the simulation results. In case of a few data correction, there is both gain by static allocation and no loss by dynamic allocation. In case of continuous data patterns, there is loss by dynamic allocation and gain by static allocation. Therefore, static allocation is more efficient in environment with high proportion of read. Dynamic allocation is efficient when data is random or high proportion of write.

Acknowledgment

This research is result of a study on the "Leaders Industry-University Cooperation" Project, supported by the

Ministry of Education, Science & Technology (MEST).

REFERENCES

[1] E. Gal and S. Toledo, "Algorithms and Data Structures for Flash Memories," ACM Computing Surveys, vol. 37, no. 2, pp. 138-163, June 2005.

[2] JK Kang, J.S Kim, C. Park, H. Park and J. Lee "A multi-channel architecture for high-performance NAND flash-based storage system," Journal of Systems Architecture, Volume 53, pp. 644-658, 2007.

[3] Y. Hu, H. Jiang, D. Feng, L. Tian, H. Luo, and S. Zhang, "Performance Impact and Interplay of SSD Parallelism through Advanced Commands, Allocation Strategy and Data Granularity," Proc. Int'l Conf. Supercomputing (ICS '11), May/June 2011.

[4] F. Chen, R. Lee, and X. Zhang, "Essential Roles of Exploiting Internal Parallelism of Flash Memory Based Solid State Drives in High-Speed Data Processing," Proc. IEEE Seventh Int'l Conf. High Performance Computer Architecture (HPCA '11), Feb. 2011.

[5] Y. Hu, H. Jiang, D. Feng, L. Tian, H. Luo and C. Ren "Exploring and Exploiting the Multilevel Parallelism Inside SSDs for Improved Performance and Endurance," IEEE Transactions on computers, Vol. 62, No. 6, pp. 1141-1155, 2013.

[6] J. Shin, Z. Xia, N. Xu, R. Gao, X. Cai, S. Maeng, and E. Hsu, "FTL Design Exploration in Reconfigurable High-Performance SSD for Server Applications," Proc. 23rd Int'l Conf. Supercomputing (ICS '09), June 2009.

[7] Y. Kim, B. Tauras, A. Gupta, and B. Urgaonkar, "Flashsim: A simulator for nand flash-based solid-state drives," in Proc. of the First International Conference on Advances in System Simulation, Washington, DC, USA, pp. 125–131, 2009.