Journal of Engineering Science and Technology Review 12 (2) (2019) 46 - 52

Research Article

JOURNAL OF
Engineering Science and Technology Review

www.jestr.org

A Multi-Channel Redundant Check Storage Method for Solid-State Disks based on Cold and Hot Data Stripes

Xin Ye*, Zhengjun Zhai and Xiaochang Li

School of Computer Science and Engineering, Northwestern Polytechnical University Xi'an, 710072, PR China

Received 2 February 2019; Accepted 29 April 2019

Abstract

Solid-state disks (SSDs) with NAND Flash media are widely used in various aspects of data storage because of their fast read/write speed, low power consumption, and non-volatile and superior vibration resistance. Due to the inherent physical limitations of NAND Flash, the reliability of SSDs decreases greatly over time. A multi-channel redundant check storage technology based on cold and hot data stripes is proposed in this study to improve the reliability of the SSD without degrading read/write performance. First, this design proposed a storage array similar to Raid-4 built in the SSD whose inner storage array was partitioned into independent data and parity storage areas accommodating user data and parity information, respectively. Second, the flash translation layer (FTL) with different area management policies was established for two storage areas above. The data area and the parity area were used to store and manage user data and parity through their respective FTLs, including different address mapping, wear-leveling, and garbage collection methods. Finally, in the FTL of the parity area, the stripes were classified into cold stripes and hot stripes according to the updating times of each stripe. Therefore the cold stripe parity could be distinguished and directly written into the NAND Flash. The parity to the hot stripe was stored in the data buffer first and subsequently written into the NAND Flash when the hot stripe turned into a cold stripe. Results show that our proposed method improves the reliability and read/write performance of the SSD by establishing a Raid-4-like memory array inside the SSD. Meanwhile, applying dual FTLs and hot and cold stripe technology can reduce the page number of data erasure and write, thus improving the service life of the NAND Flash. Compared with the traditional Raid-5 array, the number of write parity in our design can be reduced by average 40%, and read/write performance improves over 30% at the same time. The study provides a new design for improving the reliability of SSDs, especially those with frequent write requests in practical applications.

Keywords: Solid-state disk, Cold and hot stripe, Multichannel, Redundant check

1. Introduction

Hard-disk-drives (HDDs), as traditional information storage equipment, are extensively applied because of their large storage capacities. However, since the HDD is limited by mechanical structure constraints, its read/write speed is not fast enough to suit the needs of rapid data access.

In contrast to traditional disks, solid-state disks (SSDs) do not depend on mechanical movements to read and write and instead adopt NAND Flash chips, which store data by changing the charge on the storage medium. Compared with HDDs, SSDs have the advantages of high performance, low power consumption, and strong seismic performance [1]. Therefore, SSDs have been widely used for large data centers, high-performance servers, data backup, and other storage fields that require high reliability.

The storage media of SSDs are the NAND Flash chips. The physical characteristics of NAND Flash chips involve the following characteristics: 1. Erase before writing. The basic unit of the NAND Flash chip is page which does not match erase granularity, block. 2. Limited number of program/erase commands. The maximum number of erasures of the NAND Flash in the SLC structure is approximately 100,000, while the maximum erasures number in the MLC structure is only approximately 10,000 [2]. 3. Bit error. The value of a bit in the NAND Flash may be inverted when reading or writing due to the inherent characteristics of its hardware. That is, the stored content will change from "1" to "0" or from "0" to "1," which can lead to data errors, and the probability of such errors will increase when the NAND Flash usage time increases [3]. The abovementioned characteristics will eventually lead to the failure of SSD.

The error correcting code (ECC) technology is widely used in SSD to improve its reliability and reduce its chance of failure. Conventional ECCs include Hamming code, BCH code, and so on. ECCs are used to check if data are stored in idle areas of NAND Flash pages. Additional space is needed to obtain a relatively strong error correction capability in the case of fixed-storage capacity, but such capability of the ECC is limited because of the limited idle area in the NAND Flash chip [4]. ECCs cannot correct errors when the number of data bits reversed in the flash memory exceeds the ECC error correction capability or when the entire chip fails [5].

Due to the shortcomings of ECC technology, the redundant arrays of independent drive (RAID) technology has been used to improve SSD performance, especially in fields that require high reliability of SSD. Similar to disk arrays, SSDs are essentially storage arrays. Thus, RAID technology can be employed to ensure data integrity and improve SSD reliability [6]. At present, the RAID

*E-mail address: yexinishere@163.com

ISSN: 1791-2377 © 2019 Eastern Macedonia and Thrace Institute of Technology. All rights reserved.
doi:10.25103/jestr.122.07

technology used in SSDs is generally in the RAID-5 mode. That is, N channels in the SSD are regarded as N stripes. Data and parity are stored in each channel during normal SSD operation.

Benefiting from the use of parity, when problems arise one or more NAND Flash chips in a channel, the data in the failed channel can be recovered by performing an exclusive "XOR" operation on the data in the remaining channels and the parity [7]. Although RAID-5 technology can improve reliability, three problems may happen when it is directly used in SSDs. First, additional parity needs to be written into the SSD, which increases the erasures number of SSD, thus reducing the service life of the disk. Second, when the data in the SSD need to be updated, even if the required updating is minimal, the parity needs to be regenerated, thereby reducing the read–write performance of the system. Third, given that data and parity will be written into each channel of the SSD, a channel with writing parity will occupy the write time of the user data of that channel, which will lead to the degraded reading and writing performance of the whole SSD.

The present study aims to solve three problems of the SSD after the RAID-5 technology is introduced. A multi-channel redundant check method (HCS-Raid) for SSD based on hot and cold data stripes was proposed. In HCS-Raid, one channel of SSD was employed to store parity, whereas the other channels were used to independently store user data. This method adopted dual FTLs to manage user data and parity. In the aspect of user data management, HCS-Raid can use existing technologies to address mapping, wear-leveling, garbage collection, and bad block management operations. In the aspect of parity management, HCS-Raid can map the parity and the wear-leveling by judging the hot and cold stripes and by establishing a new FTL.

2. State of the Art

The principle of multi-channel parallel operation of SSD indicates that its corresponding technology was similar to that of RAID-0. However, RAID-0 only improved the read–write performance of SSD and does not improve the reliability of SSD. The read–write performance and reliability of SSD both need to be improved, and RAID-4 or RAID-5 technology was generally used in SSDs to improve reliability through parity.

Although the introduction of RAID-5 technology into SSD can effectively improve reliability, the problems mentioned in the first section of the study will still arise if the technology is directly adopted. Relevant research has been carried out to solve these problems. Wang Yu et al. [8] proposed a scheme with CR5M based on the RAID-5 architecture. In particular, CR5M was implemented in the RAID-5 architecture between flash memory chips, and mirror flash memory chip was added on each channel as a write buffer for all flash memory chips on that channel, thereby improving SSD lifetime and write throughput. The disadvantage of this approach was that common SSDs typically had only one or two flash chips per channel, and adding image flash chips to all channels will be costly [9]. Chang et al. [10] proposed a self-balancing stripe data verification scheme in which each channel was divided into data area and parity area. The data blocks corresponding to the data area in the adjacent channels were exchanged in pairs, and the obtained parity were stored in the parity areas of the other channels. In this scheme, three paths could be chosen for the read operation of any data block on the SSD, which allowed for the full use of parallel channels to minimize read delay [11]. Lee [12-13] and others divided storage spaces into data areas and parity areas by using storage class memory to cache parity. Eight data channels were considered in the scheme. Each four consecutive logical pages (distributed into four channels) form a stripe, and the corresponding parities were stored in the parity areas of the other channels. Yu et al. [14-15] adopted RAID-4 technology by using the NAND Flash chip based on SLC to store parity and another NAND Flash chip based on MLC to store user data, thus further improving reliability while ensuring storage efficiency. Lee and Yu both utilized highly reliable memory chips to store parity. Although the scheme can solve the problem in SSD reliability affected by the frequent writing of parity, the problem of degraded SSD performance had not been solved when parity was written. At the same time, adopting different types of chips in the SSD could lead to degraded system structure and reduce overall reliability [16-18]. Im [19-20] and others adopted the delay update technology based on partial parity to improve read/write performance and SSD reliability by reducing the parity capacity and the number of writes. The writing of the parity was reduced by adopting a delay scheme. Although the method could solve the effect of the frequent writing of parity on the reliability of SSD, it couldn't properly distinguish the parity. When that capacity of the SSD and the written data were both increased, the parity could also increase. Due to the limited internal cache of the SSD, the parity was frequently written into the NAND Flash, which ultimately led to the reduced reliability of the SSD [21-22]. Meng et al. [23] proposed a method to prolong the service life of SSD arrays based on compression. Given the similarity between old and new data, when writing dirty data, the dirty data and the old data were XOR-operated first and then compressed, and the data were sequentially written into a special buffer without updating the parity disk. The total amount of written data in the method was significantly reduced, thus achieving the purpose of prolonging service life. However, when the data types being written differ, the efficiency of the method could be significantly reduced. Mao et al. [24] proposed the hybridity-based disk array consisting of a set of SSD, which served as data disks, and two mechanical hard disks. Given that the total capacity of the HDDs was much larger than that of SSDs, a portion of one of the HDDs was used as a parity disk, whereas multiple SSDs formed a Raid-4 structure. That scheme was good for the frequent write and update operations of the calibration data processed by the mechanical hard disk. Moreover, this method could effectively avoid the write loss of the SSD. However, the system was complex and its process was difficult to implement. Du et al. [25] proposed wele-RAID for the wear-leveling of member SSDs in the array based on RAID-5 by improving the parity data redistribution mechanism.

The above research results to improve SSD reliability and poor versatility were mainly achieved by changing the existing SSD architecture. However, the study on improving the reliability of SSD through redundant check technology was rare. In the present study, a multi-channel redundant check technique for SSDs based on hot and cold stripes was applied. First, dual FTLs were established for the SSD. User data and parity were stored and managed separately. The FTLs were mainly used to independently manage the parity and improve the reliability of the SSD. Second, the hot and cold stripe concepts were introduced in the study. The wear on the NAND Flash can be reduced by reducing the number of writes of the parity generated in the hot stripe, and the reliability of the SSD can be further improved.

The remainder of this study is organized as follows. The third section describes the specific algorithm of the HSC-Raid. The fourth section presents the HSC-Raid, RAID0-SSD, and RAID-SSD algorithms, which are evaluated by three different types of test cases. The last section draws the conclusions.

3. Methodology

This chapter first introduces the architecture of the SSD storage verification method based on hot and cold data stripes and then describes the working principle of the method.

The method designed in this study has two characteristics. First, an independent NAND Flash channel is used to store the parity, whereas the other NAND Flash channels are employed to store the user data. The parity and the user data have their own independent mapping relationships. Second, the cold and hot data stripe concept is introduced into the timing of writing the parity. By judging whether the parity belongs to hot stripes or cold stripes, we can decide whether this parity can be written directly into the NAND Flash chip or temporarily stored in the buffer of the SSD

3.1 General framework

The traditional SSD has the characteristic of multi-channel parallel access. The NAND Flash chips on these channels all store user data. In the case of SSDs that use RAID-5 verification techniques, the user data and the parity are evenly distributed into each channel. Figure 1a shows the SSD consisting of four channels, which are divided into two stripes (Stripe0 and Stripe1). Data D0–D2 and parity P0 are stored in Stripe0, whereas data D3–D5 and parity P1 are stored in Stripe1.

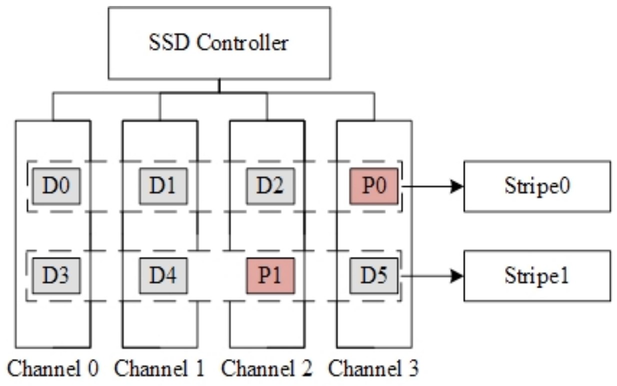

Fig. 1. Solid-state disk framework in RAID-5 mode

Fig 1 shows that user data and parity are distributed evenly in each channel of the SSD. However, three problems may arise when adopting the above storage structure. First, the channel cannot write user data at the same time when writing parity. For example, when channel 3 is writing parity P0, data D5 cannot be written, which directly affects the writing efficiency of the user data. Second, when the user data are updated, the parity is updated at the same time, which increases the number of times of erasing and writing the data block that stores the parity, thereby reducing the lifetime of the SSD. Third, the storage locations of the user data and the parity are fixed, which is disadvantageous to the use the original wear-leveling technology of the SSD, particularly in terms of prolonging the service life of the SSD.

Aimed at the problem of using RAID-5 technology in SSD, this study designs a new storage structure that divides the internal channels of the SSD into user data channel (D-channel) and parity channel (P-channel). Fig 2 shows the proposed configuration in which channels 0–3 are D-channels, whereas channel 4 is a P-channel. The D-channel and the P-channel are completely independent in terms of access control, and the SSD controller handles the access to the user data and the parity, respectively. The details of how this scheme works are described in Sections 3.2 and 3.3.

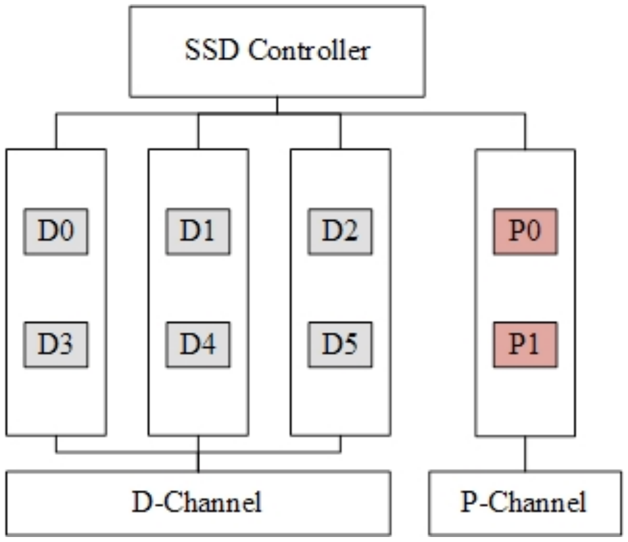

Fig. 2. Solid-state disk framework in RAID-5 mode

The use of the above storage structure can effectively avoid the problems encountered by RAID-5 technology. First, the user data and the parity are stored separately, and the writing of user data is not affected when the parity is written. Second, when the user data are updated, the original parity does not need to be erased. Only the parity of the updated data needs to be regenerated (the specific method is described in Section 3.2). Finally, the D-Chanel and the P-Chanel have separate FTLs, which allow the user data and the parity to be wear-leveled independently of one another based on the NAND Flash wear-and-tear and garbage collection operations.

3.2 Principle

In this section, we will introduce the principle of HCS-Raid in five aspects. First, we will introduce the structures of the two independent FTLs in the HCS-Raid system. Second, we will describe how HCS-Raid writes the user data and the parity when writing data into the SSD. Third, we will describe how HCS-Raid works when the data need to be updated. Fourth, we will discuss how HCS-Raid collects garbage. Finally, we will describe how to use and control hot and cold stripes when parity is written.

3.2.1 Dual FTLs

As described in Section 3.1, SSDs are divided into data area and parity area. Each area has an independent FTL for read/write control. The structure of the address mapping table of the FTL of the data area is shown in Table 1.

Table. 1. Data Area Address Mapping Table Structure

LBA	PBA	SN	STATE

The data area address mapping table adopts the page-level mapping configuration. The address mapping table contains physical address (PBA), logical address (LBA), stripe number (SN), and data state (State). SN represents the stripe in which data are located, while State denotes the condition of the physical page, i.e., idle state (free), valid state, and invalid state. When the data block has no data, this data block is in a free state. The data block is in a valid state when user data are written. When the user data are updated, the block is in an invalid state, and the physical page in the invalid state is eventually returned to the idle state by the garbage collector of the FLT.

The structure of the parity address mapping table is shown in Table 2. The address mapping table includes information such as Stripe–Number–SN, Parity–PBA, Parity–State, Erase–Count, and Update–Count.

Table 2 Data Checkout Area Address Mapping Table Structure

Parity-PBA	SN	Parity-State	Erase-Count	Update-Count

The physical address of the parity represents the specific physical location where the parity is stored. The stripe number denotes the stripe that corresponds to the parity. The state information of the parity block indicates whether the parity is valid or not. Unlike the state information of the data area, the state information of the parity block includes two states: valid and invalid. A valid state indicates that the parity stored in the Parity–PBA address is valid, whereas an invalid state indicates that the parity stored and then updated in the Parity–PBA address is invalid and can be recovered. Erase–Count records the number of erasures of the physical page that corresponds to the Parity–PBA address, and its initial value is 0. Update–Count is the number of updates to the data on the stripe that corresponds to the SN. Update–Count is automatically incremented by 1 when one or more data on the data stripe is updated.

3.2.2 Data Write

When the host side needs to write data into the SSD, three writing situations may arise. First, the amount of data written at any one time is the same as the length of a stripe. Second, the amount of data written at any one time is less than the length of a stripe. Third, the amount of data written at any one time is greater than the length of a stripe.

When the SSD receives the user data to be written, the verification module performs an XOR operation to generate parity P according to the written data bit by bit. For example, when the code of the written data is "01001," the parity P obtained by performing the XOR operation bit by bit is P=0 XOR 1 XOR 0 XOR 0 XOR 1 = 0, in which 0 is the parity of this write operation. After the parity is generated, the data and the parity are written into the SSD.

When the amount of data written into the SSD at any one time happens to be the same as the stripe length in the HSC-Raid, as shown in Stripe0 in Fig 3, data D0 to D2 are written into D-Chane0 to D-Chanel2, respectively. At this time, Stripe0 becomes full. While writing the data, parity P0 is written into the P-channel by using the principle that the D-channel and the P-channel work independently. At the same time, the address mapping tables of the data area and the parity area are updated.

When the amount of data written at any one time is less than the length of a stripe, only D-channel0 and D-channel1 need to be written as data D3 and D4, as shown in Stripe1 in Fig 3. When calculating parity P1, the data of all three channels are also subjected to XOR operation. When D-channel2 is not written into the data, it is in a free state at this time, and all data in D-channel2 are 0xFF. Therefore, P1=D-channel0 XOR D-channel1 XOR 0. When P1 is calculated, it is written into the address mapping table of the P-channel, and the same data area and check area are updated.

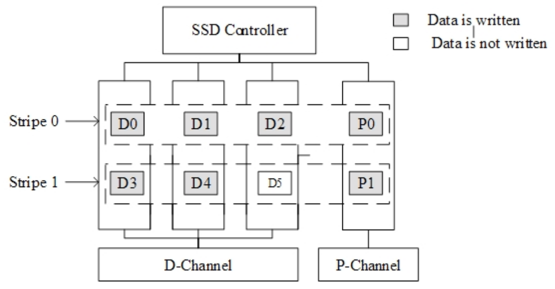

Fig. 3. Data Write

When the amount of data written into the SSD at any one time is greater than the length of a stripe, the data need to be divided into two parts. The first part is set to be equal to the length of the stripe, whereas the rest is smaller than the length of the stripe. After the data partition is completed, the data and the parity can be written in accordance with the above two situations.

The process described above is a case of writing data only once. The above process indicates that when the write data size and the stripe size do not match, one or more D-channels appear in a stripe, which are not writing data, and in the free state. However, these channels, which are in a free state, are not always in this state. Moreover, when the data have the appropriate size, they are written directly into this channel, and the parity is updated at the same time.

3.2.3 Data Update

Given the physical characteristics of the NAND Flash, the SSD adopts the offsite update mode when certain data need to be updated. For example, when D0 data need to be updated, the new data are first written into D6 in D-channel2, and the D0 status is marked as invalid. If the original parity P0 remains unchanged, then the data block is only marked as an invalid state, the data are not deleted, and a new parity cannot be produced. This scheme not only improves the reading and writing efficiency of the system but also reduces the writing time of the parity, thus prolonging the service life of the NAND Flash. When the updated data are written into D5, both D5 and P1 perform an XOR operation to generate the new parity P1', and simultaneously write P1' to the P2 position, in which the original P1 is marked as invalid (Fig 4).

3.2.4 Garbage collection

The garbage collection will work with FTL when much data are in the invalid state. This scheme will erase the invalid data blocks and mark the erased data blocks as free. The parity also needs to be regenerated.

As shown in Fig 5, the blocks in D0, D3 and D4 are in the invalid state. D1 is in a free state, whereas D2 is in the active state. By comparison, two data blocks can be found in Stripe1 in the invalid state, which has the most bands in the invalid state. Therefore, Stripe1 can be garbage-collected.

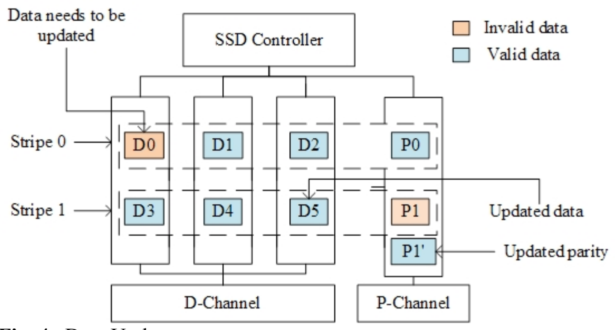

Fig. 4. Data Update

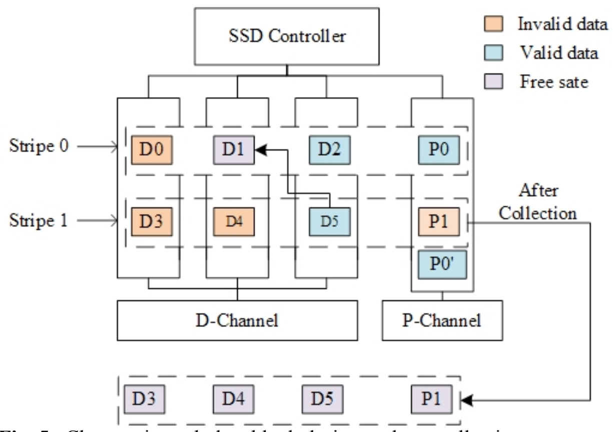

Fig. 5. Changes in each data block during garbage collection

Through garbage collection, the system automatically erases all data blocks in Stripe0 and flags the data block status as free, whereas data D5 are written into Stripe0, and a new parity P0' is generated.

Similarly, garbage collection is required when the P-channel has numerous invalid verification data. The garbage collection mechanism of the P-channel is the same as that of the D-channel, except for the timing of the garbage collection. The P-channel will independently judge the status of its stored data and decide when the garbage collection operation should be carried out.

3.2.5 Principle of Cold and Hot Stripes

The parity is also stored on the NAND Flash chip. When a data block on a stripe is written into new data, a new parity will be generated by the XOR operation. If each generated parity is immediately written into the P-channel, the storage system will be heavily loaded, and the reading and writing performance of the SSD will be affected. At the same time, the frequent writing of parity will accelerate the wear of the NAND Flash chip and reduce its service life.

We design a technique of writing calibration information based on cold and hot stripes to improve the work efficiency of the system and reduce wear on the NAND Flash chips. Given that each parity corresponds to a fixed stripe, the system will automatically record the number of the new data write mode for each stripe. If a stripe is frequently written into new data, then the parity of the stripe will not be directly written into the P-channel but instead will be temporarily stored in the system cache. When the cached parity is to be written back to the P-channel, it is not written directly to the original physical location but is exchanged with the least number of updates in the validation block.

4 Result Analysis

The SSDsim simulation platform is used to verify our design, particularly to evaluate the performance of the HSC-Raid. By modifying the SSDsim, the SSD with five channels and cache of 128 MB can be virtualized. Each channel of the SSD consists of two NAND Flash chips based on the SLC architecture. Each NAND Flash chip has 8192 blocks.

4.1 Performance Test

Three kinds of common traces are considered to accurately simulate the performance of the algorithm. Table 3 provides a detailed description of the three types of traces. RAID0-SSD and RAID5-SSD are tested in this study

Table 3 Details of Category 3 traces

	Number of read requests	Number of write requests	Total requests	Write request proportion
Trace1	4941220	60015	5001235	1.2%
Trace2	2265738	929944	3195682	29.1%
Trace3	1253623	4058340	5311963	76.4%

Fig 6 depicts the test of the overall elapsed time in which all requests in our time trace are continuously entered into the simulation. On the basis of the test results, RAID0-SSD has the highest overall execution efficiency because it has not generated parity. The HSC-Raid algorithm has higher execution efficiency than RAID5-SSD because it has adopted the dual FTL method and the hot and cold stripe classification processing method.

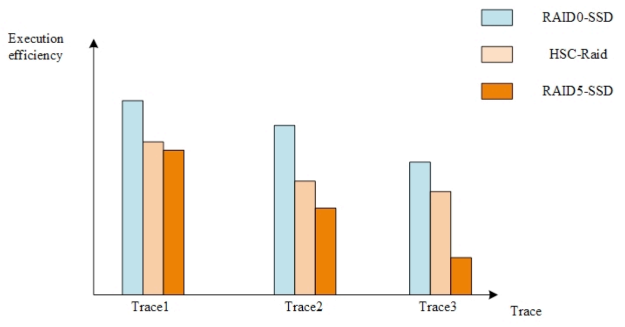

Fig. 6. Test results for overall elapsed time

Given the relatively few write requests in Trace1, and because most instructions are read requests, RAID0-SSD, HSC-Raid, and RAID5-SSD are nearly identical in terms of efficiency.

In Trace2, the number of write requests is increased, and RAID0-SSD has the highest execution efficiency because the generation and writing of parity have not been conducted. HSC-Raid has higher execution efficiency than RAID5-SSD because it can store and manage the user data and the parity separately, and the writing of parity has not affected the writing of user data.

Compared with RAID0-SSD, both HSC-Raid and RAID5-SSD have attained substantial parity in Trace3 because of the large number of write requests. Meanwhile, HSC-Raid used the dual FTL method and the cold and hot stripe method. Thus, the generation and writing of parity have both led to a much lower impact on SSD performance than RAID0-SSD.

4.2 Wear Rate Test

The test cases in Table 3 are also used to evaluate the wear rate of the NAND Flash chips when the SSD adopts HSC-Raid, the RAID0-SSD and the RAID5-SSD algorithms. Fig 7 shows that results of the NAND Flash chip wear rate test by using different algorithms.

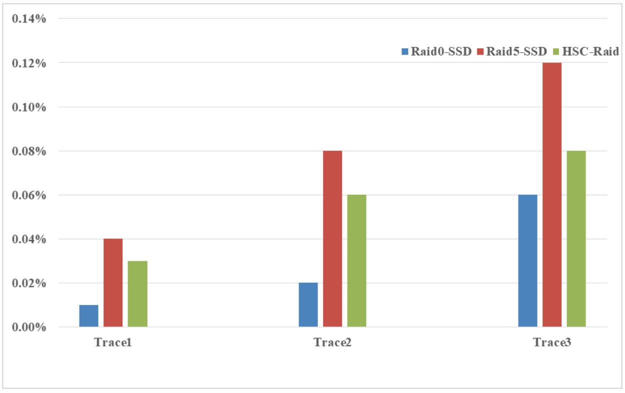

Fig. 7. NAND Flash Wear Rate Test Results

When tested with Trace1, the NAND Flash wear rates are low for all three algorithms because of the relatively few write requests. RAID0-SSD has the lowest wear rate on the NAND Flash because it did not generate and write parity.

When tested with Trace2, the wear rates on the NAND Flash of the three algorithms increased due to the rise in write requests. Similarly, the RAID0-SSD algorithm has the lowest wear rate on the NAND Flash because of the absence of parity. HSC-Raid has lower wear rate on the NAND Flash than RAID5-SSD because it used the hot and cold stripe method to reduce the writing of the calibration information.

When Trace3 is used in the test, the wear of the NAND Flash chips is the greatest because Trace3 has the largest proportion of write requests among the three test cases. The wear rate of the NAND Flash chips with HSC-Raid is much lower than that of RAID5-SSD because of the use of the cold and hot stripe method and the dual FTL technology.

5. Conclusions

A multi-channel redundant check technique for SSDs based on cold and hot data stripes was designed in this study to minimize the impact on the read/write performance of the SSD and improve its reliability. This technique attained a reliability similar to that of RAID5-SSD. At the same time, the method could improve read/write performance by storing the user data separately from the parity. The parity of the hot stripe was temporarily stored in the buffer by introducing the concept of the cold and hot stripe as a manner of reducing the wear on the NAND Flash, thus further improving the reliability of the SSD. Moreover, given that the user data were separately stored and managed from the parity, the read/write performance of the SSD will generally not be affected when the parity was written into the SSD. The following conclusion can be drawn:

(1) The wear rate of the NAND Flash on HSC-Raid disks is lower than that on RAID5-SSD, whereas the wear rate of the NAND Flash on HSC-Raid disks is similar to that on RAID0-SSD with the increase of write requests.

(2) The SSD with HSC-Raid is superior to the SSD with RAID5-SSD in terms of reading and writing performance, especially when write requests are increased, and the improvement of HSC-Raid performance is more apparent.

This study can serve as guide to improve the reliability of SSDs on the premise of having to guarantee the read/write performance of these SSDs. In the present study, two FTLs are employed to manage user data and parity, which will lead to the increase of address mapping scale and the decrease of system response speed. Therefore, further optimizing these two FTLs according to the characteristics of user data and parity is necessary to enhance the performance of SSDs.

Acknowledgements

This work was supported by the Shaanxi Science and Technology Co-ordination and Innovation Project (No. 2016TZC-G-8-3).

References

1. Wang Shiyuan., "*Research on Error Characteristics Modeling of Nand Flash and Applications*". Master thesis of Harbin Institute of Technology, China, 2016, pp.18-21.
2. Li Hongyan, Gui Chao, Xiao Kun., "An ECC Design Method Based on The Heterogeneous Perception of MLC SSD". *Journal of Chinese Studies*. 2018, pp.136-141.
3. S. Lee, B. Lee, K. Koh, and H. Bahn., "A Lifespan-aware Reliability Scheme for RAID-based Flash Storage". In: *Acm Symposium on Applied Computing*, TaiChung, Taiwan: ACM, 2011, pp.374–379.
4. Li Xujin., "*Research on Key Technologies of Nand Flash Solid State Storage Reliability*". Master thesis of Harbin Institute of Technology, China, 2018, pp. 22-25.
5. Kevin M., Greenan, Darrell D.E., Long, Ethan L., Miller., "Building Flexible Fault-Tolerant Flash-based Storage Systems". In: *Proceedings of the Fifth Workshop on Hot Topics in System Dependability*, Lisbon, Portugal: IEEE, 2009, pp.123-129.
6. Yongkun Li, Patrick P.C., Lee., "Analysis of Reliability Dynamics of SSD RAID". *IEEE Transactions on Computers*, 65(4), 2016, pp.1131-1144.
7. Gaoxiang Xu, Zhipeng Tan, Dan Feng., "Cap: Exploiting Data Correlations to Improve the Performance and Endurance of SSD RAID". In: *36th International Conference on Computer Design*, Orlando, USA: IEEE, 2018, pp.59-66.
8. Wang Y., Wang W., Xie T., "CR5M: A Mirroring-powered Channel-RAID5 Architecture for an SSD". In: *Mass Storage Systems and Technologies (MSST)*, Santa Clara, USA: IEEE, 2014: pp.1-10.
9. Ping Huang, Pradeep Subedi, Xubin He., "FlexECC: Partially Relaxing ECC of MLC SSD for Better Cache Performance". In: *USENIX ATC'14 Proceedings of the 2014 USENIX conference on USENIX Annual Technical Conference*, Philadelphia, PA, AMOLD: 2014, pp. 489-500.
10. Chang Y B, Chang L P., "A Self-balancing Stripeing Scheme for NAND-flash Storage Ssystems". In: *Proceedings of the 2008 ACM symposium on Applied computing*, Fortaleza, Brazil: ACM, 2008, pp.1715-1719.
11. Qin Yi., *"Research on Algorithms to Enhance Availability for SSDs*". Master thesis of Huazhong University of Science & Technology, China, 2014, pp.19-22.

12. Y. Lee, S. Jung, and Y. H. Song., "FRA: A Flash-aware Redundancy Array of fFlash Storage Devices". In: *Proceedings of the 7th International Conference on Hardware/Software Codesign and System Synthesis*, Grenoble, France: ISSS, 2009, pp. 163–172.
13. Zhan Ling, Wu Wei, Wang Fang., "Research on a RAID5/6 Writing Optimization Technique Based on SSD Caching". *Journal of Chinese Mini-Micro Computer Systems*, 39(10), 2018, pp.2226-2232.
14. Yu Ruirong, "*RAID4-structured Heterogeneous-Chip-Based SSD*". Master thesis of Huazhong University of Science and Technology, China, 2015, pp.17-19.
15. Wen Pan, Feng Liu, Tao Xie., "SPD-RAID4: Splitting Parity Disk for RAID4 Structured Parallel SSD Arrays". In: *2013 IEEE International Conference on High Performance Computing and Communications & 2013 IEEE International Conference on Embedded and Ubiquitous Computing*, Zhangjiajie, China: IEEE, 2013, pp.9-16.
16. X. Jimenez, D. Novo, and P. Ienne., "Wear Unleveling: Improving Nand Flash Lifetime by Balancing Page Endurance". In: *12th USENIX Conference on File and Storage Technologies*, Santa Clara, CA: USENIX, 2014, pp. 47–59.
17. Pan Yubiao., "*Study on Performance Optimization for SSD-based RAID Arrays*". Master thesis of University of Science and Technology of China, China, 2015, pp.23-25.
18. J. Ouyang, S. Lin, S. Jiang, Z. Hou, Y. Wang, Y. Wang., "SDF: Software-defined Flash for Web-scale Internet Storage Systems". In: *Proceedings of the 19th international conference on Architectural support for programming languages and operating systems*, Salt Lake City, USA: ACM, 2014, pp. 471–484.
19. Suzhen Wu, Weidong Zhu, and Guixin Liu., "GC-Aware Request Steering with Improved Performance and Reliability for SSD-Based RAIDs". In: *IEEE International Parallel and Distributed Processing Symposium*, Rio de Janeiro, Brazil: IEEE, 2018, pp.296-305.
20. S. Im and D. Shin., "Flash-aware RAID techniques for dependable and high-performance flash memory SSD". *IEEE Transactions on Computers*, 60(1), 2011, pp.80–92.
21. Yangsup Lee, Sanghyuk Jung, Yong Ho Song., "FRA: A Flash-aware Redundancy Array of Flash Storage Devices". In: *Proceedings of the 7th IEEE/ACM international conference on Hardware/software codesign and system synthesis*, Grenoble, France: ISSS, 2009, pp.163-172.
22. Yu-Bin Chang, Li-Pin Chang., "A Self-Balancing Striping Scheme for NAND-Flash Storage Systems". In: *Proceedings of the 2008 ACM symposium on Applied computing*, Fortaleza, Brazil: ACM, 2008, pp.1715-1719.
23. Meng Wentao, "*Extending Lifetime of Solid State Disk Arrays Based on Data Compression*". Master thesis of Huazhong University of Science and Technology, China, 2017, pp.29-32.
24. B.Mao, H.Jiang, D. Feng., "A hybridparity-based disk array for enhanced performance and reliability". In: *2010 IEEE International Symposium on Parallel & Distributed Processing*, Atlanta, USA: IEEE, 2010, pp.135-142.
25. Seol, Jinho Shim, Hyotaek, Kim, Jaegeuk., "A Buffer Replacement Algorthm Exploiting Muti-chip Parallelism in Solid State Disks". In: *Proceedings of the 2009 international conference on Compilers, architecture, and synthesis for embedded system*s Grenoble, France: CASES, 2009, pp.137-146.