A page lifetime-aware scrubbing scheme for improving reliability of Flash-based SSD

Nan Li, Qiyou Xie, Yinan Wang, Xiangyu Liu, Husheng Liu and Wei Yi

National University of Defense Technology, No. 109, Deya Road, Changsha, Hunan 410073, China

Abstract: Solid-state drive (SSD) has gain prevalence in consumer and enterprise storage markets. However, its reliability is declining with the wearing of the Flash memory. The error correcting codes (ECCs) are generally applied in SSD to cope with bit errors caused by abrasion, however, the lifetime of pages in Flash is still limited to their fixed correctability. Due to different process and usage, the pages of Flash deteriorate in different speeds, which limited the service time of SSD. In order to prolong the lifetime of SSD, we proposed a Page Lifetime-aware Scrubbing (LaScrub) scheme to find dangerous pages which exist more bit errors. The simulation results show the proposed scheme is able to enhance the reliability of SSD. It gains 86% storage space compared with normal SSD at the same reliability requirement.

Keywords: Solid-state disk, Flash, Lifetime, Scrubbing scheme, Reliability, LaScrub

Classification: Circuits and modules for storage


1 Introduction

In order to reduce the cost per bit, storage manufacturers are aiming to raise the storage density by storing more bits in one single float gate. Therefore the tolerable Program/Erase (P/E) cycles are decreasing and the minimum ECC (error correcting code) requirement is increasing. As a result, the reliability of SSD becomes a key concern for users. In order to prolong the lifetime of SSD, wear-leveling (WL) algorithms are adopted universally. Generally, the distribution of P/E cycles is the measurement for evaluating their performance. The researchers do their best to make all blocks in SSD erased evenly. They believed that the pages with same P/E cycles will hold equivalent raw bit error rate (RBER). However, many factors, such as retention time, program/read interference, process difference and page significance (LSB or MSB), may influence the RBER value and cause a great discrepancy among different pages. The lifetime of SSD is limited by the weakest page.

In traditional hard disk drive (HDD) storage systems, the scrubbing scheme is commonly adopted to find the defect sectors. However, it’s seldom used in SSD storage systems. As the mapping relation between logical sector address and physical page number is changing over time, the uncorrectable pages are hardly located on system level. In this paper, a page lifetime-aware scrubbing (LaScrub)
scheme is introduced to SSD controller based on the bit error characteristics of NAND Flash memory. Also a page residual lifetime prediction (PRLP) scheme is proposed to assess the risk of occurrence of uncorrectable sector errors (USE). In order to find the risky pages, active and passive searching methods are designed for pages with cold and hot data respectively. To reduce the scrubbing overhead, a bit error logging scheme is designed for blocks with cold data and a shortening ECC scheme for blocks with hot data. By these methods, the reliability of SSD can be enhanced, and users earned time to replace and backup the high risky drives.

The paper is organized as follows. Section 2 reviews relevant researches on the mechanism and reliability problem of NAND Flash and reinforcing methods. Section 3 analyzes the characteristics of bit error in detail. Section 4 proposes a page residual lifetime predicting model and countermeasures for high risky pages. Section 5 provides detailed simulations to validate the proposed schemes. Conclusions are given at last.

2 Recent researches on SSD's reliability optimization

The reliability of SSD is mainly determined by the storage medium: Flash memory. The Flash cells store the information by means of holding different amount of electrons. However, many interference factors may change the memory cell state. Some positive factors like program disturb, read disturb may make the Float Gate (FG) over-programed. Some negative factors like retention time may drain the electrons [1].

In order to prolong the service life of SSD, Wear-Leveling (WL) algorithms are universally adopted to keep all Flash blocks are erased evenly. The essential elements considered in WL algorithm are erase count number (ECN) [2], using density [3] and garbage collection (GC) efficiency [4]. However, the effect of WL algorithms highly depends on the application environment [5]. Owing to the scaling of Flash technology, the variance across blocks or among the Flash pages within a block becomes increasingly significant. Even if all blocks are aged evenly, some Flash blocks or pages may suffer more bit errors. Pan et al. in [6] proposed a dynamic BER-based greedy (DBG) method to maximize the wear leveling efficiency. Also, the pages in a block deteriorate in different speeds. Jimenez et al. in [7] suggested a wear unleveling (WU) method to alleviate the pressure in weak pages. However, the RBERs of Flash pages are determined by many factors and changing over time. For example, a young block with long retention time may has the same average BER value with an aged block with short retention time. So in fact, some wrong judgments may put the SSD into danger.

Scrubbing scheme is commonly used in HDD-based Redundant Array of Independent Disks (RAID) system to locate latent sector errors (LSEs) in advance [8] [9]. The defect sectors are recovered by the parity drives. If we solely apply the scrubbing scheme to SSD, there is no parity for recovery even though USEs are detected. Recently, the intra-disk redundancy (IDR) schemes are employed for high reliable SSD [10] [11]. However, the overhead in performance and space is significant after SSD adopts IDR and scrubbing scheme simultaneously. If we scrub the total SSD and pick out the risk pages in time before they become
unrecoverable, the reliability of SSD can be ensured. Ilias et al. in [9] make a thorough comparison between the IDR and scrubbing schemes in HDD-based system. And the IDR schemes are more excellent than the most efficient scrubbing scheme. However, the precondition is that all sectors in HDD have the same probability of LSEs’ occurrence. Supposing that we can precisely locate the weak pages according to the characteristics of Flash memory and reduce the performance overhead for scrubbing, the scrubbing scheme is able to surpass the IDR schemes.

**3 Bit Error Characteristics of NAND Flash memory**

In NAND Flash memory, many factors contribute to RBER, such as P/E cycles, retention time, write and read disturb. In order to analyze the bit error characteristics of NAND Flash, experiments were performed on five commercial MLC Flash devices with same technology. It can be found that the suffered P/E cycles determine the electron loss speed and the retention time decides the amount of loss. Although many other factors may influence the RBER of Flash memory, those interferences are ignorable compared with retention error.

3.1 Program/Erase Cycles

![Fig. 1. RBER of four successive pages while P/E cycles increasing.](image-url)

In order to study the aging process of NAND Flash, we firstly programmed the pages in blocks sequentially with random data and then read out in turn to locate the bit errors. The data retention time of all tested pages is very short and equal. As depicted in Fig 1, the RBER in four successive pages is growing exponentially with P/E cycles on the whole. The bold solid curve in the middle is obtained by an exponential fitting method. It can be concluded that the page which suffers more P/E cycles will be more risky to lose data. Also, the practical RBER values fluctuate around the fitting curves. It is mainly because the pages’ stored content is changing. The pages which constitute a block deteriorate at significantly different speeds.
3.2 Retention Time
The retention errors of five pages were tested after 0 to 10K P/E cycles at room temperature. Every page has same page number and is programmed with same data. As shown in Fig. 2, the RBER of all pages are increasing over time with slow speed. The minimum ECC requirement of the tested Flash memory is 60-bit ECC per 1141 bytes of data. That means the RBER threshold is $6.6 \times 10^{-3}$. The oldest page with 10K P/E cycles can only store data for about 2700 hours under the threshold. The younger pages are stronger to combat the retention errors. It proves that the retention time of one page is closely related to its aging degree.

Fig. 2. Retention error in pages with different P/E cycles.

3.3 Write Disturb
Compare to the retention error, the write operations charge more electrons to surrounding cells, causing the nearby cells jump to other logical states. In the test, all disturbances affect the RBER concurrently. It’s hard to distinguish the program interference from other factors. However, it is more difficult for the programming voltage to disturb a cell that already holds many electrons, but easier for it to disturb a cell that holds few electrons [1]. It means that the LSB page have higher RBER than MSB page soon after being programmed. The experiment data validates this explanation. We collect the bit error number of four adjacent pages after 10K P/E cycles. As shown in Fig 3, the number of bit errors in LSB page is higher than that of MSB page at first. However, the RBER of MSB pages exceeds that of LSB page after 120 hours. Owing to program interference, the RBER of LSB page increase slowly or decrease after being programmed. However, the influence of program disturb is submerged after about 120 hours retention time.

Because higher voltage applied to nearby cells during programming [12], the disturb severity of a single program operation is far greater than a read operation. The influence of read disturb is distinguishable after millions of read operations [13]. Thus the read disturb test did not perform independently.
As the deterioration are different in Flash pages, the pages with high RBER are more risky and determine the lifetime of SSD. In order to improve the reliability of SSD, we propose a scheme to search the dangerous pages and predict their residual life. If the lifetime of the observed pages is not long enough, remedy schemes are carried out according to the hotness of stored data.

We define that the time duration from firstly programmed until data unrecoverable is life time of Flash’s page. We also define two RBER thresholds named as hard threshold ($RBER_{th}$) and soft threshold ($RBER_{sth}$) for the proposed scrubbing scheme. The $RBER_{th}$ equals the ECC boundary, e.g. $6.6 \times 10^{-3}$ for the under test chips. The $RBER_{sth}$ is the maximal allowed RBER in required uncorrectable bit error rate (UBER). E.g. the UBER requirement for consumer’s SSD is $10^{-15}$ [14]. Therefore, the $RBER_{sth}$ of the tested chips is $2.6 \times 10^{-3}$ for consumer’s SSD.

4.1 Page Residual Lifetime Prediction

The RBER of programmed Flash pages is increasing over time and the growing speed is slowing. In order to get pages’ residual life, the controller should sample enough RBER values in fitting methods. However, the overhead for sampling and estimating is significant. In order to cut down the prediction cost, a conservative method are proposed to roughly estimate the residual lifetime. In this method, the last RBER values and access time of the observed pages are firstly recorded, then the average RBER growth speed is calculated by computing the quotient of the increased RBER and elapsed time when those pages are revisited. The average RBER are used to estimate the page residual lifetime. It’s given as follows,

$$t_R = \frac{RBER_{th} - RBER_{cur}}{RBER_{cur} - RBER_{last}} \times (t_{cur} - t_{last})$$  \hspace{1cm} (1)
where $RBER_{cur}$ and $RBER_{last}$ represent the current and last RBER value. And $t_R$, $t_{cur}$ and $t_{last}$ denote the residual retention time, the current and last sampling moments respectively. As shown in Fig 2, the average speed is obviously faster than the future speed. Thus the average speed can be used to replace the practical speed to obtain a conservative residual retention time.

As shown in Fig 4, the calculation process is equivalent to draw a straight line through two adjacent sample points. And the intersection of the straight line and the threshold line is the conservative estimation of retention time. The $t_{R1}$ and $t_{R2}$ represent two successive predictions of residual retention time. In practical application, the estimated value is regarded as the next checking time and used to calculate new residual lifetime. Thus, the actual residual lifetime can be estimated after several predictions.

![Graph showing the estimation process](image)

**Fig. 4.** Schematic of residual lifetime estimating process.

4.2 Find Dangerous Pages

In practical SSD controller, the overhead will be significant when the SSD controller records all pages’ RBER information. The key point is to pick out the weak pages and blocks for observation. Although there is wear leveling (WL) algorithms in Flash Translation Layer (FTL), the blocks in SSD are unlikely to age evenly. The pages in aged blocks which suffer more abrasion are risky to become uncorrectable. Besides, the pages in blocks which store data for a long time are subject to more retention errors. In most personal and enterprise computers, much of used space is occupied by cold data which is seldom accessed by users. The Flash blocks that store cold data are rarely recycled by WL algorithm because the copy back overhead is heavy, especially when their capacity becomes large. In order to find the dangerous pages, the following measures are adopted.
At first, the programmed blocks into hot blocks and cold blocks are classified according to the read frequency. In order to correct bit errors, the ECC module in SSD works at every read operation. The distribution of bit errors in hot blocks can be monitored nearly in real-time. But the bit error condition in cold blocks is unknown. The controller need to periodically access the pages to ensure their security. If the RBER of Flash pages in a block exceeds its threshold, it will be marked as a risky block and be picked out for observation. Secondly, two tables are established to store their RBER information for the risky hot and cold blocks respectively. In order to save storage space, only one representative page per block are selected. As shown in Fig 5, the RBER table item contains the block number, page number, the last RBER, the last access time and the predicted residual lifetime. In order to quickly access the weakest page, the table items are organized by a balance binary tree (BST). At last, the proposed PRLP scheme are used to estimate the page residual lifetime. The information in RBER table is updated after every prediction.

### Data entry

<table>
<thead>
<tr>
<th>Block</th>
<th>Page</th>
<th>RBERlast</th>
<th>tlast</th>
<th>tR</th>
</tr>
</thead>
</table>

### Keyword: block

- **Block 10**
- **Block 6**
- **Block 15**

### keyword: residual lifetime

- **tR: 100 hours**
- **tR: 80 hours**
- **tR: 200 hours**

![Fig. 5. Organization of table for the proposed PRLP scheme.](image)

#### 4.3 Remedy Scheme

When the RBER of the representative page in a block exceeds, remedy methods need to be carried out to avoid uncorrectable sector errors. As the data in hot blocks is accessed frequently, all valid pages have to be copied to a free block to ensure the reliability and performance. Whereafter the aged blocks are recycled. However, these recycled blocks are subject to short retention time and high risk if they are reused. Hence we propose a shortening ECC scheme to improve the correctability of ECC module. In the scheme, the data bits length shrinks but the number of parity bits remains unchanged. Specifically, one code takes charge of less data bits and the correctability increases on the whole.

The Step 1 in Fig 6 illustrates the organization of ECC in Flash pages. The parity bits follow on the heels of the data bits. In the shortening ECC scheme, the length of data bits shrink a half at every step. At the same time, the average correctability is doubled. Since the number of parity bits per page is increased, the number of data bits is reduced accordingly. For example, a page which can store 16 KB data is able to store 14.5 KB in step 2 and 12 KB in step 3. It means about
90% to 75% of storage space are recycled. According to the theory of BCH code, the shortened ECC fill “0” at the data bit end to recover the original codes in coding and decoding process. The codec in SSD controller only requires very small changes to implement the proposed scheme.

For cold blocks, the SSD controller needs to carry out more copy operations because they have stored more valid pages. In order to reduce the copying overhead, a bit error scrubbing scheme are proposed to prolong the retention time. In the scheme, the bit error location and flipping direction acquired from ECC decoder are recorded if the page residual lifetime is not long enough. When the page is accessed next time, the SSD controller will correct the bit errors in scrubbing log at first and the ECC module focus only on the fresh bit errors. However, this scheme makes an impact on the read performance owing to requiring extra read operations. Actually, the cold block is seldom accessed and will not engross much reading resource.

A bit error table (BET) is used to log the error bit location in this scheme. Usually, the length of data in an ECC is short than the page length, such as 1024 bytes data per code. The data cells in an ECC are called as a sector in this paper. Controller allocates the same amount of space for each sector for logging bit error. The node in the BET contains 15 bits error location and one bit error value. One sector requires 120 bytes of table for ECC with 60 bit correctability. When one cold block is chosen for scrubbing, the error bits of all pages in it are recorded. And it will be moved from the weak page list. If these pages are accessed next time and become unrecoverable, the BET will be enabled.

However, the space overhead for storing BET is not neglectable. For example, supposing that the required ECC for a MLC chip is 60 bits per 1K bytes data, a
16K bytes page needs almost 2K bytes for the BET. Obviously, SSD do not have so much RAM space for these tables. Thus the controller need to refresh these BETs to Flash memory periodically. Compared with the traditional scrubbing schemes, only 12.5% space is used for a page and will not produce abrasion to Flash memory. In order to further decrease the space overhead, controller can first scrub the MSB page and observe the LSB page. Because one scrubbing operation is equal to reset the bit error number in a page to zero, the growing speed of bit errors is slowing, which means the page lifetime is doubled at least.

5 Evaluation

5.1 Page RBER distribution

The reliability of SSD is closely related to the page RBER distribution. To get the RBER distribution of SSD, we use a two peaks Gaussian distribution to describe the RBER distribution of LSB and MSB pages in one block respectively, in which only the influence of P/E cycles and retention time is considered. The RBER distribution of four schemes (NONE, DBG, WU and LaScrub) are calculated. NONE represents the scheme with none BER-aware method. In order to test the performance of DBG scheme, we add some amount of RBER difference which follows the Bounded Gaussian Distribution in [6]. We suppose that the process difference is eliminated after employing the DBG scheme. In WU scheme, 25% of the pages in Flash block are fully relieved. LaScrub indicates the proposed scheme. Since the correctability of ECC in the LaScrub scheme is doubled after employing the remedy methods, the RBERs of the pages in observed blocks are halved in this simulation. The capacity of the emulated SSD is 256 GB and the lifetime is assumed ten years. The retention time of its storage data obeys a uniform distribution. To avoid the influence of WL algorithms, we suppose all blocks in the SSD wear evenly and the P/E cycles equal to 3000, which is the maximum tolerable erase times of the tested devices.

![RBER distribution of the emulated SSD](image.png)
As shown in Fig 8, the RBER distribution of the SSD after 10 years’ usage are simulated. The distributions of the four schemes have two peaks. It’s mainly caused by the different capability of LSB and MSB page. The DBG and WU scheme can decrease the number of pages with high RBER value compared with NONE scheme. However, these schemes cannot control the page RBER which grows with retention time and P/E cycles. LaScrub is able to locate the dangerous pages in time and estimate their residual lifetime. The reinforcements are carried out according the situation of Flash blocks. The dash line in Fig 8 is the RBER threshold of the tested device. The pages at the right of the dash line are uncorrectable. The unrecoverable page ratios are 24.14%, 23.40%, 15.89%, 0.01% for NONE, DBG, WU and PRLP, respectively. As the tradeoff, the proposed scheme consume 10% to 25% storage space for ECC and logging bit errors.

5.2 Capacity Gain

We assume that the life of SSD is ended when encountering an unrecoverable sector error. So the SSD lifetime is decided by the weakest page. As shown in Fig 9, the four successive pages own different aging speeds. As the NONE and WU scheme do not change the ECCs’ correctability, they have the same RBER threshold. For NONE scheme, the odd LSB page determine the tolerable P/E cycles. The WU scheme alleviates the bad pages by skipping them when programming. So it can release the strength of strong pages which is able to tolerate more program cycles. Because the proposed scheme elevates the capability of ECC, the Flash pages can endure more P/E operations. Table 1 shows the suffered P/E cycles of the four pages in NONE PRLP, WU with 25% and 50% relief. At last, the total available storage space are calculated, which shows that the proposed scheme(PRLP) can provide more usage space for SSD.
Table I. Available space of one Flash block

<table>
<thead>
<tr>
<th>Scheme</th>
<th>Even LSB</th>
<th>Even MSB</th>
<th>Odd LSB</th>
<th>Odd MSB</th>
<th>Available Space</th>
</tr>
</thead>
<tbody>
<tr>
<td>NONE</td>
<td>22.8k</td>
<td>22.8k</td>
<td>22.8k</td>
<td>22.8k</td>
<td>89.1 MB</td>
</tr>
<tr>
<td>WU (25%)</td>
<td>22.8k</td>
<td>30.7k</td>
<td>30.7k</td>
<td>30.7k</td>
<td>112.2 MB</td>
</tr>
<tr>
<td>WU (50%)</td>
<td>22.8k</td>
<td>30.7k</td>
<td>33.4k</td>
<td>33.4k</td>
<td>117.5 MB</td>
</tr>
<tr>
<td>PRPL</td>
<td>33.8k</td>
<td>33.8k</td>
<td>33.8k</td>
<td>33.8k</td>
<td>126.6 MB</td>
</tr>
</tbody>
</table>

5.3 Tolerable P/E cycles

If the retention time is taken into account, the tolerable P/E cycles will be decreased greatly. The simulation parameters remain unchanged except the P/E cycles. Suppose that all blocks wear evenly and the suffered P/E cycles are increasing gradually. When an uncorrectable error occurs, the suffered P/E cycles indicate the SSD’s endurance and are used to compute the available space. The DBG+WU scheme in Table 2 represents the combination of the block-level DBG scheme and the page-level WU scheme. As shown in Table 2, we compared the tolerable P/E cycles and the available space gain of the five schemes. Because NONE only considers the P/E cycles, it performs worse than other schemes. WU (25%) obtains 11.5% growth in P/E cycles, but the available space gain is only 7%. It’s mainly because the alleviated pages provide less storage space. The performance of DBG is closely related to the process difference in block level. The DBG scheme does not conflict with the WU scheme. So DBG+WU perform better than every standalone method. The proposed scheme apperceives the page RBER which comes from all sides. The weak pages and blocks are picked out and reinforced by remedy methods. It performs better than all other schemes and achieves 86% available space gain than the NONE scheme.

Table II. Total available storage space of a 256 GB SSD

<table>
<thead>
<tr>
<th>Scheme</th>
<th>P/E cycles</th>
<th>Available space</th>
<th>Gain</th>
</tr>
</thead>
<tbody>
<tr>
<td>NONE</td>
<td>1291</td>
<td>323 PB</td>
<td>1</td>
</tr>
<tr>
<td>WU (25%)</td>
<td>1439</td>
<td>351 PB</td>
<td>1.07</td>
</tr>
<tr>
<td>DBG</td>
<td>1787</td>
<td>447 PB</td>
<td>1.38</td>
</tr>
<tr>
<td>DBG+WU (25%)</td>
<td>2046</td>
<td>464 PB</td>
<td>1.44</td>
</tr>
<tr>
<td>PRPL</td>
<td>2401</td>
<td>600 PB</td>
<td>1.86</td>
</tr>
</tbody>
</table>

6 Conclusion

In SSD, the difference in RBER comes from many aspects, such as manufacturing, memory architecture, FTL algorithm and application environment. The WL algorithm designers hope all blocks wear evenly. However, the SSD controller has to make a tradeoff between the reliability and performance. In this paper, our proposed scheme focuses on picking out the weak pages. To get the appropriate time for taking measures, the residual lifetime of them are estimated. According to the temperature of the storage data, we carry out shortening ECC or bit error logging scheme to prolong their service life with the least overhead. At last, the simulation results show the advantages of our scheme.

Acknowledgement

This work is partly supported by the National Natural Science Foundation of China (No.61704191 and No.61701509).