Write Amplification due to ECC on Flash Memory or Leave those Bit Errors Alone

Sangwhan Moon
Texas A&M University
sangwhan@tamu.edu

A. L. Narasimha Reddy
Texas A&M University
reddy@ece.tamu.edu

***Abstract*—While flash memory is receiving significant attention because of many attractive properties, concerns about write endurance delay the wider deployment of the flash memory.**

This paper analyzes the effectiveness of protection schemes designed for flash memory, such as ECC and scrubbing. The bit error rate of flash memory is a function of the number of program-erase cycles the cell has gone through, making the reliability dependent on time and workload. Moreover, some of the protection schemes require additional write operations, which degrade flash memory's reliability. These issues make it more complex to reveal the relationship between the protection schemes and flash memory's lifetime.

In this paper, a Markov model based analysis of the protection schemes is presented. Our model considers the time varying reliability of flash memory as well as write amplification of various protection schemes such as ECC. Our study shows that write amplification from these various sources can significantly affect the benefits of these schemes in improving the lifetime. Based on the results from our analysis, we propose that bit errors within a page be left uncorrected until a threshold of errors are accumulated. We show that such an approach can significantly improve lifetimes by up to 40%.

I. Introduction

Flash memory is receiving wide attention and their deployment is steadily increasing, since it can have higher performance while consuming lower power than Hard Disk Drives (HDDs). However, limited lifetime of flash memory is one of the main concerns. Since the write and erase operations on flash memory wear it out gradually, after a certain number of operations, data could potentially be lost. This write endurance problem is related to the physical features of flash memory.

Flash memory can employ single-level cells (SLC) or multi-level cells (MLC). While MLC allows significant improvements of capacity over SLC, the lifetime of MLC is comparably lower, at 10,000 Program/Erase (P/E) cycles compared to 100,000 P/E cycles for SLC.

Many studies have investigated the bit error failure behavior of SLC and MLC. Two notable studies in this direction include [1], [2]. These studies point out that the bit error rate of the flash memory increases with increased number of P/E cycles. In fact, these studies model the bit error rate as an exponential function of the number of P/E cycles the cell has gone through. This variable bit error rate requires further study to understand the implication on the flash memory.

This paper focuses its attention on reliability of MLC flash memory. First, MLC has lower write endurance than SLC and hence requires more attention in understanding and improving the reliability. Second, higher capacity flash memory is demanded and MLC flash memory is promising to be on the market in the future. In this paper, flash means MLC, unless otherwise explicitly stated.

Many approaches have been suggested to overcome the write endurance limitation. An Error Correcting Code (ECC) [3] encodes data and stores the encoded data in order to detect and correct errors at a page level, of size, say 4KB. The ECC is checked and used to correct any detected errors whenever the page is read. Scrubbing [4] actively scans flash memory and detects/corrects errors using ECC. Flash memory employs a Flash Translation Layer (FTL) in order to provide wear-leveling of blocks. Flash cells have to be erased before they can be programmed, requiring a copy-on-write mechanism. This copy-on-write mechanism is exploited to spread the writes across all the memory such that frequent writes to one address do not result in one place being worn out while other places are not written.

Some of the protection methods mentioned above, however, require additional writes and these writes in turn can cause wear out of flash memory. For example, log-like writing of FTL can cause fragmentation which requires garbage collection process, which results in writes and erases when it moves fragmented data to a different place in the memory. Pages are corrected and rewritten to the memory when ECC detects bit errors in the pages. The recovery process issues an additional write to write the corrected page. While ECC is beneficial to detect and correct the bit errors, the extra writes lead to higher bit error rates and potentially can lead to lower lifetimes. Scrubbing increases the chances to detect and correct errors in a page, before the page accumulates too many errors to become uncorrectable by ECC. However, frequent recovery from scrubbing could lead to frequent extra writes and in turn could lower the flash memory's lifetime.

The number of excess writes that result due to a single write request is termed as *write amplification*. While these excess writes may be necessary, the writes cause wear out of flash cells and hence can potentially reduce the flash memory's lifetime.

In this paper, we explore the relationship between the write amplification of the data protection schemes and the reliability of the flash memory. The paper makes the following significant contributions:

- We provide a model for analyzing flash memory, taking variable bit error rate and the write amplification of the data protection schemes into account.
- We show that write amplification of ECC and garbage collection can contribute up to a loss of 50% of data lifetime in flash memory.
- We propose a novel technique to reduce the write amplification of ECC that improves the lifetime significantly by up to 40%.

Sec. II introduces various protection schemes for flash memory. Sec. III builds reliability model of these protection schemes. Sec. IV shows evaluation. Sec. V proposes our novel protection scheme. Sec. VI introduces related work. Sec. VII discusses open issues and future work. Sec. VIII concludes this paper and discusses future work.

II. Flash Memory Protection Schemes

A number of techniques have been developed to protect flash memory. In this section, widely used methods are discussed.

A. Error Correcting Codes

ECC encodes data into check bits, and the encoded data is exploited to detect and correct errors in the data. The number of detectable and correctable errors is highly dependent on the complexity of the employed ECC. Since MLC is prone to more errors than SLC, it requires stronger multi-bit correcting ECC like BCH code or Reed-Solomon code instead of SECDED Hamming code which is widely used in SLC. The storage and calculation overhead of ECC depends on the level of protection that is desired.

B. Scrubbing

ECC can be used to correct/detect errors only when data is accessed. In order to protect data that may not be frequently accessed in normal workloads, data on the flash memory are actively scanned and errors found are *scrubbed*. Data scrubbing [4] can be done during the idle periods of the flash memory. Scrubbing rate can be either constant or exponentially distributed. In general, a large portion of data on the flash memory are cold data and this makes scrubbing essential for data protection though it consumes energy for scanning the memory.

C. Wear-leveling and Garbage Collection

The flash blocks have to be erased before they can be rewritten. If blocks are overwritten with new data, hot blocks will wear out some locations faster than the rest of the memory. The FTL tries to spread the writes over the entire memory such that all the blocks wear out uniformly, in order to increase the lifetime of the memory. Erase operations of flash memory operate in units of a block, for example, 512KB, while write operations are done in units of a page, 4KB. Consequently, a number of valid pages alive in a block to be erased have to be moved to another block before an erase. This recycling process is called *garbage collection*.

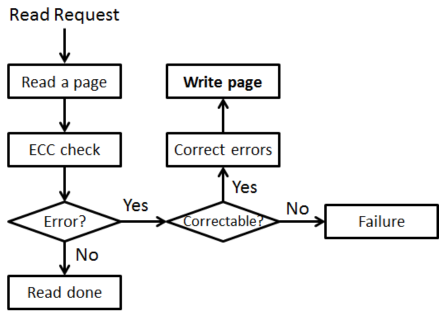

Fig. 1. Write amplification from recovery process

D. Write Amplification

The protection schemes discussed in the previous section generate undesirable additional writes. This write amplification has been a just minor overhead to HDDs. On the other hand, for flash memory, the write amplification severely degrades reliability of memory, since the reliability is highly dependent on the number of writes done to the flash memory. Main sources of the write amplification are discussed in this section.

1) Garbage Collection: During the garbage collection process, moving live pages from one place to another place results in increasing the number of writes issued to the memory. Different algorithms are employed for garbage collection. These algorithms try to minimize the write amplification, for example, through selection of appropriate blocks with the least number of live pages and postponing garbage collection as long as possible. Write amplification due to garbage collection is strongly dependent on the space utilization of the flash memory [5]. When the flash memory is nearly full, garbage collection initiates quicker and results in being less efficient since a larger fraction of the blocks are still live.

2) Recovery Process: Data recovery can be initiated due to ECC level error detection. In most of the recovery processes, at least one write is required to write corrected page back to the memory. Fig. 1 describes the process of write amplification from recovery process. This write amplification is inevitable and the number of amplified writes is highly dependent on page inspection rate which is statistically an average access rate to the page. The term write amplification is known to be caused only by writes; however, we are figuring out reads also lead to write amplification. This has a significant effect on the flash memory's lifetime in modern computing environment where reads are dominant compared to writes. Fig. 2 compares the traditional point of view to write amplification on left side, where only writes amplifies writes, to our point of view to write amplification on right side, where both reads and writes contribute to write amplification. w_{ecc} represents the write amplification from recovery process, and f_{ecc} is a fraction factor.

ECC can correct a number of errors in a page simultaneously and it results in only one write. For instance, fixing a bit error in a page takes one redundant write, while fixing ten bit

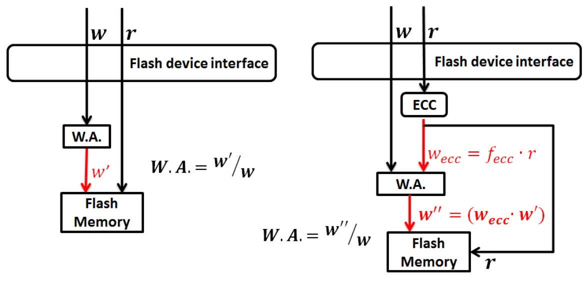

Fig. 2. Write amplification from the traditional point of view (left) and our point of view (right).

Error Type	A	B
Read disturb	3.114e-7	2.169e-4
Data retention	3.297e-6	1.827e-4

TABLE I
RAW BIT ERROR RATE $\lambda(x) = A \cdot \exp(B \cdot x)$

errors in a page also needs one additional write. Therefore, intensive recovery makes flash memory robust as well as it amplifies writes and hurts the memory, making the design of a recovery process challenging.

Read-after-write mechanism is a popular technique to protect flash memory from write errors. Whenever write operation is done to a page, correctness of write is confirmed by read the page immediately after write. It rewrites the page if write errors are detected in the page which leads to write amplification.

III. Reliability Model

Our first goal is to build a reliability model and analyze the various factors affecting the lifetime of flash memory. While there have been many studies on the reliability analysis of flash memory, to the best of our knowledge our work here provides the first model for MLC flash memory considering write amplification. We assume that workload is random and uniformly distributed over entire memory such that the page access rate is constant.

A. Raw Bit Error Rate

A number of studies including [1], [2] have investigated the bit error behavior of flash memory. According to them, there are different sources of bit error of flash memory: read disturb, electron discharge, and write failure. These studies model the bit error rate as an exponential function of the number of P/E cycles the cell has gone through. We start with the assumption that bit errors are independent and their probabilities are exponentially distributed, and then we employ the data from the measurement study of [2] to model the rate of change of bit error rate. Table I shows the employed model. $\lambda(x)$ is the raw bit error rate at x P/E cycles.

Write error is immediately recovered by read-after-write mechanism of flash memory; however, the recovery process requires extra write operation to write corrected data back which wears out flash memory faster. In this paper we assume that read-after-write scheme is not working if it is not specified.

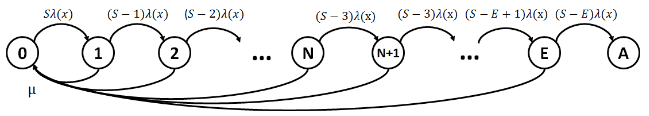

Fig. 3. A canonical Markov model of uncorrectable page error probability.

B. Uncorrectable Page Error Probability

ECC can correct errors up to a certain number of bits in a page. When higher number of bit errors occur within a page, the page fails or other techniques such as RAID have to provide data protection.

As shown in Fig. 3, the canonical Markov model is typically used to build a statistical model of reliability of ECC for a page. E is the number of correctable errors, S is the number of bits per page, $\lambda(x)$ is the bit error rate at x P/E cycles from the model of Sec. III-A, and μ is page recovery rate. The recovery rate is in fact the average page access rate, since the access interval time for the same page is much higher than ECC correction time. State A is the absorbing state where the number of errors exceed what ECC can correct and hence results in a page failure at the memory level. From an analysis of the Markov model the probability of reaching the absorption state A can be obtained.

The bit error rate varies by P/E cycles and it can be considered by modeling a series of Markov models with different bit error rates. If we assume that access interval time for the same page, $1/\mu$, is sufficiently large and time varying nature of $\lambda(x)$ is relatively small enough, we can treat $\lambda(x)$ as constant at each P/E cycles x in the series. We can estimate the uncorrectable page error probability at each slot using steady state analysis of Markov chain. Then, the series of these probability $g(x)$ is integrated over P/E cycles to get an expected P/E cycles to data loss.

C. Mean Time to Data Loss

Since perfect wear-leveling and uniformly distributed random workload are assumed, writes and resulting erases are uniformly spread out over the entire flash memory; statistically, for a flash device of N pages, N writes to the flash device is equivalent to one write to each page on average. Therefore, the MTTDL of the flash device is $\mathrm{MTTDL_d} = (\mathrm{MTTDL_p} \cdot \mathrm{N})/\mathrm{N}$ where $\mathrm{MTTDL_p}$ is the MTTDL of a page shown in Eq. (1).

$$\mathrm{MTTDL_p} = \lim_{k \to \infty} \sum_{j=1}^{k} \left(jg(j) \prod_{i=1}^{j-1} (1 - g(i)) \right) \tag{1}$$

D. Write Amplification

Write amplifications are caused by garbage collection, recovery process and read-after-write which are denoted as α_{gc}, α_{rcv} and α_{raw}. These write amplification increase the bit error rate and hence increase the uncorrectable page error probability and impact the data lifetime. α_{gc} is dependent on space utilization and the range of it is estimated to range from 1.0 to 4.9 in [5]. α_{rcv} is estimated as $\alpha_{rcv} = \left(\mu \sum_{i=1}^{N} P_i\right)/w$, where μ is the page recovery rate, w is the average write workload to a page and P_i is the probability of staying at

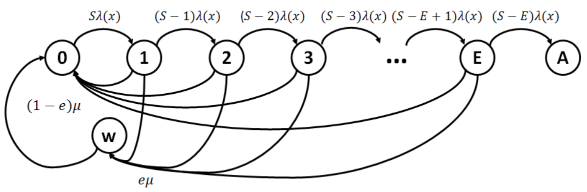

Fig. 4. A Markov model of uncorrectable page error probability considering write errors and read-after-write.

state i of Markov model in Fig. 3 of Sec. III-B. When write errors are considered and read-after-write mechanism is turned on, recovery process of read-after-write produces extra writes. Fig. 4 shows the model of uncorrectable page error probability considering write errors and read-after-write mechanism, where e is the probability of write error occurring. S is the number of bits per page, $\lambda(x)$ is the bit error rate at x P/E cycles from the model of Sec. III-A, μ is the page recovery rate, E is the number of correctable bits by ECC and A represents an absorbing state when errors cannot be recovered by ECC. Write amplification from read-after-write is $\alpha_{raw} = \left(e\ \mu \sum_{i=1}^{N} P_i\right)/w$ where w is the write workload and e is the write error probability.

We assume that the sources of write amplification work independently and overall write amplification is $\alpha = \alpha_{gc} \cdot \alpha_{rcv} \cdot \alpha_{raw}$. In practice, they do not work independently, for example, while moving live pages for garbage collection errors in the pages can be found and corrected when it moves to the new place. We retain study on their dependency as future work.

The write amplifications are emulated by changing sampling of error probability function. Let $g(x)$ be an uncorrectable page error probability, and $\alpha = 1.3$, then new error probability $g'(x) = g(1.3x)$.

IV. Evaluation

Various aspects of the expected lifetime of flash memory is explored in this section. This includes lifetime changes under the consideration of ECC, scrubbing, memory usage patterns such as space utilization and hot/cold dichotomy. As mentioned above, we exploited bit error rate mainly from [2], specifically 3x nm memory with the employment of 61-bit correctable (out of 4KB page) binary BCH code.

A. Practical Issues

We built a reliability model for flash memory in the previous section. When we evaluate this model, we are confronted with many practical issues.

Relative MTTDL We evaluate the lifetime of memory and normalize by the lifetime of reference memory. For example, if the reference memory's MTTDL is 1.5 years, and the evaluated memory's MTTDL is 0.5 years, the normalized lifetime of the memory is given as 0.33. We expect this to focus our attention on the relative strengths of various protection schemes, rather than the absolute lifetimes which are dependent on memory vendor. The reference memory is set to be protected by 61-bit correctable ECC without any degradation from write amplification.

r:w	5000	10000	15000	20000	25000	30000
1:1	1.0302	1.0839	1.2125	1.4430	1.7011	1.8738
3:1	1.0308	1.0889	1.2475	1.6287	2.3165	3.0930
5:1	1.0309	1.0899	1.2560	1.6862	2.5968	3.9032
7:1	1.0310	1.0904	1.2598	1.7142	2.7571	4.4806
9:1	1.0310	1.0906	1.2619	1.7308	2.8609	4.9130

TABLE II
Write amplification from ECC recovery at different P/E cycles

Environment As we show later, the results are dependent on the space and throughput utilization of memory. Unless otherwise mentioned, the results assume a space utilization of 0.5 and throughput utilization of 0.5. Specifically the space capacity is 80GB, maximum throughput is about 120MB/s and r:w ratio is 3:1. The P/E cycle is about 80 minutes per page on average. Note that these are arbitrary choices, and we vary some of these to study the sensitivity of results to these values. Operating system in fact has to exploit TRIM command to know which data is invalidated by deletion. We assume that the TRIM command is being issued by file system whenever delete is done with zero overhead.

Sources of Failure Sources of bit errors evaluated in this paper are read disturb and retention failure; as we will show, write failure is also evaluated, however, the result is not included if not specified since read-after-write mechanism can recover the write failure immediately and we find that its effect on lifetime of flash memory is negligibly small. Other sources such as whole device failure and software errors are not yet modeled. We retain them as future work and focus on sources of failure sensitive to write behavior of flash though they need to be taken into account to get a full picture of flash memory's reliability.

B. Sources of Degradation

We look at the relative contribution to the loss of lifetime resulting from the various sources of write amplification.

ECC Table. II reveals the relationship between workload and write amplification from ECC recovery. The amount of write workload is fixed and read workload varies by r:w ratio. The higher r:w ratio implies intensive recovery rate and, hence, frequent extra writes for recovery.

To analyze how much write amplification is harmful to lifetime, we evaluated the life time of flash memory. Fig. 5 shows the impact of various factors on lifetime. When the write amplification of garbage collection is considered, the lifetime of a single memory decreases by 20%. The write amplification from ECC recovery contributes to a loss of additional 30% of the lifetime. This shows the importance of devising data protection techniques that are less write intensive.

Scrubbing Scrubbing is well-known to be useful for seldom referred blocks. For HDDs, scrubbing is beneficial if it works in the background and does not interfere with foreground job; however, for flash memory, intensive scrubbing may be harmful due to write amplification. The impact of scrubbing on lifetime of flash memory is shown in Fig. 5. We have investigated what exactly causes this degradation, and found

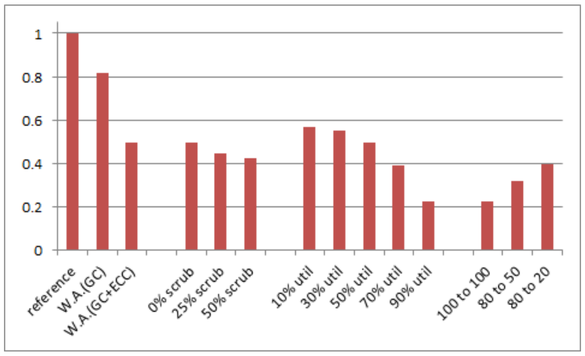

Fig. 5. Impact of various factors on lifetime (Relative MTTDL).

that page error rate is increased by write amplification from frequent ECC recovery than it is reduced by boosted recovery rate. Though our evaluation cannot find benefit from scrubbing MLC flash memory, we cannot argue that scrubbing is always harmful.

Garbage Collection Write amplification from garbage collection is highly dependent on space utilization and hot/cold distribution of data. Their impact on the lifetime of flash memory is discussed here.

Space Utilization As mentioned before, the amount of write amplification is strongly related to space utilization. Earlier work has studied the relationship between write amplification from garbage collection and space utilization [5]. We exploit this relationship to see the impact of space utilization on lifetime of flash memory.

Fig. 5 shows the change of lifetime as a function of space utilization. As we expected, lifetime is less at higher utilizations due to garbage collection. This implies increasing utilization 50% to 70% or 90% is possibly more expensive than increasing a number of memories keeping space utilization at 50%. This also brings out an important question: if we need to sacrifice some of useful capacity for improving lifetime of an MLC, is it better to sacrifice that capacity for an SLC at the same cost, since SLC can provide better write endurance than MLC? We leave this question for future research.

Read-after-write Write error, in addition to read disturb error and data retention error, is one of three major sources of bit errors in flash memory; however, write errors can be simply detected and corrected by read-after-write mechanism as we describe in section III-D. A Markov model of Fig. 4 is evaluated in Fig. 6. It shows write amplification due to read-after-write reduces lifetime less than 3% when write error probability is about 0.1% which is impractically high. In this paper, write error is not considered and read-after-write mechanism is turned off.

Hot/Cold Dichotomy In real systems, data exhibit hot and cold behavior, where a few blocks receive significant fraction of requests while other blocks receive a smaller fraction of requests. We assumed that wear-leveling works well enough so that access rates for hot and cold blocks are fair in a long term observation; however, in a short term observation, hot blocks are more likely to be overwritten, which results in more page invalidations before they are reclaimed. Due to this behavior

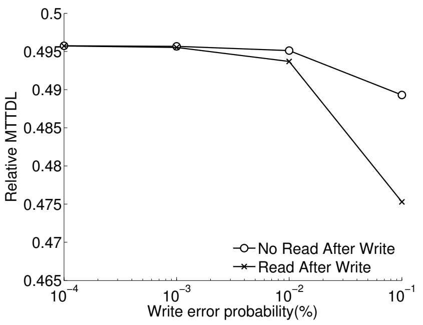

Fig. 6. Read-after-write mechanism

n	5000	10000	15000	20000	25000
$= 1$	0.0286	0.0756	0.1657	0.2463	0.2105
≤ 3	0.0295	0.0823	0.2077	0.4022	0.4604
≤ 5	0.0295	0.0824	0.2096	0.4323	0.5824
> 5	6.57e-10	3.12e-7	8.50e-5	0.0072	0.1163

TABLE III
PROBABILITY DISTRIBUTION OF THE NUMBER OF ACCUMULATED BIT ERRORS (n) RECOVERED BY ECC

of hot and cold blocks, garbage collection can gather invalid pages more efficiently by reclaiming hot blocks more often and this results in less write amplification. We exploited data of write amplification from recent study of [8] which reveals the relation of write amplification and hotness of data. The evaluation is shown in Fig. 5. In the figure, x to y means x% of throughput is concentrated on y% of space. Space utilization is set to 90% in this evaluation since [8] provides results of 85% and 90% of utilization. The result shows that hot blocks increase lifetime of flash memory because of more efficient garbage collection.

V. Threshold-based Error Correcting Code

Our analysis showed that, given sufficient workload intensity, most errors within a page are corrected rapidly so that only a few bit errors accumulate before an ECC action corrects and writes that page to the memory. As pointed out earlier the writes of ECC can decrease data lifetimes. With sufficiently intensive recovery rate, could we postpone these writes from ECC? To avoid the disadvantage of extra writes of ECC recovery, we propose threshold-based ECC correction. In this scheme, data is corrected by ECC and the correct data is returned to the requesting process, but data is not corrected on the memory until the number of bit errors within the page reach a threshold.

In other words, new scheme leaves damaged page until the number of errors exceed some threshold. For instance, with 200MB/s workload in 80GB SSD, 10^{-7} bit error rate means about one bit error is detected/corrected in 4KB page on average. Disk scrubbing scans SSD in idle time and corrects and rewrites data to correct just one or two bits in each page. Threshold-based ECC correction waits until 10, 20, or a certain number of errors are accumulated, and then corrects data to reduce the number of extra writes, which is critical to flash memory's reliability. As Table. III shows, most of the bit errors are corrected before a page accumulates a significant

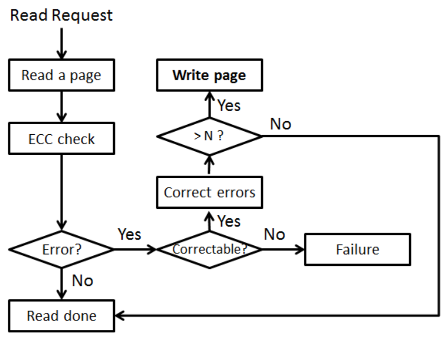

Fig. 7. Write Amplification and Threshold-based ECC

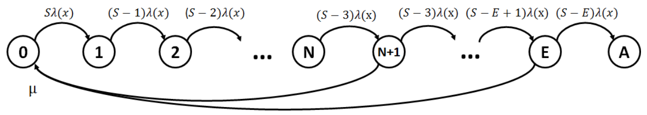

Fig. 8. A Markov model of uncorrectable page error probability with threshold-based ECC correction

number of errors. Our new scheme significantly reduces write amplification from ECC recovery, since considerable portion of write amplification comes from repairing one or few errors rather than many errors at once.

Fig. 7 describes how threshold-based ECC works. For read requests for a corrupt page, threshold-based ECC delays writing the corrected data until it accumulates bit errors to a threshold, N. Fig. 8 shows a Markov model for threshold-based ECC, where S is the number of bits per page, $\lambda(x)$ is the bit error rate at x P/E cycles from the model of Sec. III-A, μ is the page recovery rate, N is the threshold, E is the number of correctable bits by ECC and A represents an absorbing state when errors cannot be recovered by ECC.

Table. IV shows evaluation of threshold-based ECC. x% threshold means that ECC related writes are skipped until the accumulated bit errors in the page reach x% of correctable errors within the page. Since leaving too many errors produces drastic growth of page error rate, there exists an optimal number for the threshold. Table. IV shows that leaving about 70% of correctable errors, 42 bit errors in a 4KB page, with 61-bit ECC is optimal. In general, the optimal threshold will be a function of the workload intensity and we leave the question of determining optimal threshold to future work.

VI. Related Work

Studies on write amplification of flash memory [5]–[8] analyze the impact of garbage collection on write amplification. We study other main sources of write amplification such as ECC recovery, and we move our focus on the write amplification from ECC recovery since it has a huge impact on lifetime of flash memory according to our analysis.

The work [9] looks at reliability of MLC type flash memory, and estimates uncorrectable bit error rate using a binomial distribution.

VII. Future Work

We made a number of assumptions to make the analysis tractable. We plan to relax these assumptions in the future.

Threshold(%)	0	10	30	50	70	90
R.MTTDL	0.496	0.614	0.671	0.694	0.702	0.696

TABLE IV
Relative MTTDL of Threshold-based ECC

We considered static thresholds for threshold-based ECC and showed that it could lead to improvements in lifetime. In general, the thresholds will depend on the workload intensity and characteristics along with the memory level bit error rates. As future work, we plan to analyze and design optimal thresholds.

We also plan to explore more cross-layer coordination opportunities in the future for improving flash memory's lifetime further. While our work here showed that a higher level of threshold for our threshold-based ECC may not be beneficial, could this be different if we have a data protection at the system level, for example, replicas or parity protection. Such approaches will require coordination of memory level and system level policies. We plan to not only study such policies, but potential for implementing them.

VIII. Conclusion

This paper studied the implications of write amplification from various aspects. Studies on write amplification of flash memory before [5]–[8] analyze the impact of garbage collection. We study other main sources of write amplification such as ECC recovery. In this paper, a Markov model was exploited to show that write amplification can have a significant impact on the lifetime of flash memory, since the bit error rate of flash memory increases exponentially. Our analysis reveals that lifetime loss due to the write amplification is about 50% and a considerable amount of loss comes from frequent ECC recovery. We proposed threshold-based ECC which leaves errors on flash until it accumulates bit errors to a certain threshold. Our new scheme is shown to increase lifetime by up to 40%. We made a number of assumptions to make the analysis tractable. We plan to relax these assumptions in the future.

References

[1] L. Grupp et. al., "Characterizing flash memory: Anomalies, observations, and applications." In MICRO'09.

[2] H. Sun et. al., "Quantifying reliability of solid-state storage from multiple aspects," in SNAPI'11.

[3] F. Sun et. al., "On the use of strong BCH codes for improving multilevel NAND flash memory storage capacity," in SiPS'06.

[4] T. Schwarz et. al., "Disk scrubbing in large archival storage systems," in MASCOTS'04.

[5] X. Hu et. al., "Write amplification analysis in flash-based solid state drives," in SYSTOR'09.

[6] A. Jagmohan, M. Franceschini, and L. Lastras, "Write amplification reduction in NAND flash through multi-write coding," in Mass Storage Systems and Technologies (MSST'10), May 2010.

[7] S. Boboila and P. Desnoyers, "Write Endurance in Flash Drives: Measurements and Analysis." in FAST '10, Feb. 2010.

[8] P. Desnoyers, "Mathematical Models of Write Amplification in FTLs," Presentation in NVRAMOS'11, 2011.

[9] N. Mielke, T. Marquart, N. Wu, J. Kessenich, H. Belgal, E. Schares, F. Trivedi, E. Goodness, and L. Nevill, "Bit error rate in NAND flash memories," in IEEE International Reliability Physics Symposium, 2008.