Health-Binning: Maximizing the Performance and the Endurance of Consumer-Level NAND Flash

Roman A. Pletka Saša Tomić

IBM Research – IBM Zurich Research Laboratory
CH-8803 Rüschlikon, Switzerland
{rap,sat}@zurich.ibm.com

Abstract

In recent years, the adoption of NAND flash in enterprise storage systems has been progressing rapidly. Todays all-flash storage arrays exhibit excellent I/O throughput, latency, storage density, and energy efficiency. However, the advancements in NAND technology are driven mostly by the consumer market, which makes NAND flash manufacturers focus primarily on reducing cost ($/GiB) and increasing the storage density by technology node scaling, by increasing the number of bits stored per cell, and by stacking cells vertically (3D-NAND). This comes at the cost of reduced endurance of the raw NAND flash, larger variations across blocks, and longer latencies, especially with extremely high error rates (due to the use of read-retry operations).

In this paper, we present Health Binning, a technique that facilitates bringing low-cost consumer-level flash to the quality required for enterprise-level storage systems. Health Binning determines the wear characteristics of each block in the background and uses this information in the data-placement process to map hotter data to healthier blocks and colder data to less healthy blocks.

Health Binning significantly improves the endurance and performance of the storage system: It actively narrows the block wear distribution and moves endurance from being dictated by the worst blocks towards to a value corresponding to the average endurance of all blocks, resulting in up to 80 % enhanced endurance compared with other wear-leveling schemes. At the same time, the probability of reads with high raw bit error rates (RBER) is reduced, thereby decreasing the number of read-retry operations throughout the device lifetime.

SYSTOR '16, June 6–8, 2016, Haifa, Israel.

ACM 978-1-4503-4381-7/16/06…$15.00.
http://dx.doi.org/10.1145/2928275.2928279

Categories and Subject Descriptors D.4.2 [*Storage management*]: Secondary storage; C.3 [*Special-purpose and application-based systems*]: Real-time and embedded systems; D.4.8 [*Performance*]: Simulations

Keywords Flash memory, endurance, wear leveling, data placement, solid-state drives, solid-state storage systems

1. Introduction

During the past decade, radical changes in NAND flash technology were mainly driven by the consumer market. Scaling down the technology node resulted in increased storage density and in lower cost per GiB, which in turn increased the adoption of flash. However, apart from the desirable increase in storage density and lower cost per GiB, shrinking the technology node and increasing the number of bits stored per cell also entail (1) a decrease in the specified program-erase cycles (PEC) and hence reduced device lifetime, (2) an increase in access time, especially if errors are corrected using read-retry commands, and (3) weaker data retention.

For example, when comparing a 25 nm SLC with a 19 nm MLC NAND flash, the specified endurance dropped from 100,000 to 3000 PECs, even with 2.5× bits required for the error-correction code (ECC). The page read latency also increased by 3.5× to 120 µs. Data retention for enterprise applications dropped from 12 to 3 months, at 40 °C.

These undesirable trends of worsening endurance, reliability, and performance are likely to continue, in the medium to long term. Although the emerging 3D-NAND has temporarily concealed these issues by reverting to the larger technology node on the order of 40 nm, the already announced continued scaling of the technology node and the number of layers will unquestionably relaunch the earlier trends of the planar (2D) technology [19].

The specifications of a flash technology define the endurance of a block in terms of a number of PECs before it exceeds an error rate that cannot be corrected by ECC. Traditionally, the Flash Translation Layer (FTL) attempts to equalize the PEC across blocks. This makes perfect sense if we strictly adhere to the specification and expect that each

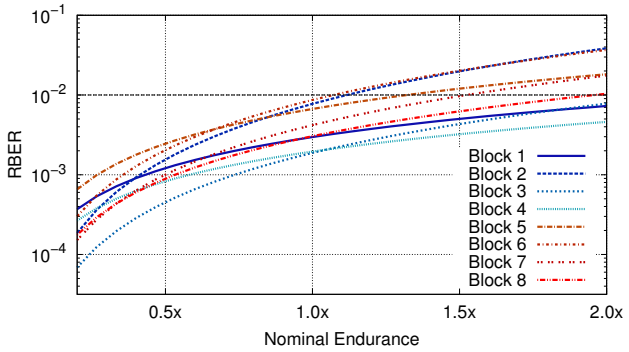


Figure 1. RBER of different consumer-level flash blocks as a function of the P/E cycles. Some blocks have several times higher endurance than others, but conventional wear-leveling techniques cannot take advantage of this without risking early device failure and poor performance.

block will reach *exactly* its target endurance. However, there is an unavoidable variability across blocks, and therefore manufacturers conservatively provide the specifications based on (nearly) the worst blocks expected.

Figure 1 illustrates the measured Raw Bit Error Rates (RBER) of the worst page obtained from the characterization of eight blocks from a real consumer-level 16 nm MLC flash chip as a function of the nominal endurance, which corresponds to the PEC normalized to the manufacturer-specified block endurance. This small set of blocks had been carefully selected from a large number of characterized blocks to illustrate certain common properties. There is a huge variability of the maximum endurance between the blocks. Some blocks can sustain several times more PECs than the others before reaching the same ECC limit. A similar variability across blocks can be observed even in enterprise-level flash, although to a smaller extent, likely because of a careful selection of silicon dies from the production process.

This result indicates that balancing PECs across blocks does not necessarily improve device endurance. Instead, balancing the RBER across blocks would improve endurance, as it guarantees that the ECC will be able to correct errors for a longer time. As many modern ECCs achieve an acceptable unrecoverable bit error rate by correcting up to 1% of errors (10^{-2}) [25], we assume support for a similarly strong ECC. However, the techniques presented in this paper are applicable with any ECC strength.

Non-optimal RBER balancing (e.g., by placing write-hot data to the worst blocks) will result in uncorrectable ECC errors and early retirement of some blocks. Early in device life, these blocks will cause *poor performance* due to frequent read-retry and other data-recovery operations. And after retiring them, the capacity of these blocks will be reduced from the FTL over-provisioning, putting more stress on the Garbage Collection (GC) process, thereby increasing write amplification and ultimately accelerating reaching the *premature end* of device life. Already at approx. 50% of the theoretically achievable endurance, the entire device will have to be declared unusable because the over-provisioning has become exhausted.

The ideal approach to achieving good performance and endurance from flash with high variability across blocks would be to wear out the healthiest blocks more and the unhealthiest blocks less. In this way, all blocks in the device would reach their endurance limit at approximately the same time, resulting in maximum device endurance and performance until the very end. The open questions in such a technique are (1) how to identify and track the healthiest blocks, and (2) how to efficiently wear them out more. In this paper we provide answers to these two questions.

The contributions of this paper are:

- We introduce the dynamically-tracked RBER of flash blocks as a metric of their age, wear, and health status.
- We present Health Binning, a technique that uses the dynamically-tracked RBER to significantly improve the performance and endurance of NAND-flash storage.
- As a part of Health Binning, we show how the use of Block Grading, a technique that periodically associates a health grade to each block, is a remarkably efficient way to enable wear balancing at computationally low overhead compared with full sorting of blocks. Block Grading is therefore suitable to be implemented in computationally-limited environments such as an SSD.

The remainder of the paper is organized as follows: Section 2 gives an overview of the block-management architecture. Health Binning and Block Grading are described in Section 3, followed by an evaluation in Section 4. Related work is discussed in Section 5, and conclusions and future work are described in Section 6.

2. Block Management Architecture

In this section, we describe a generic architecture of a flash-based storage device. A high-level overview of the block management is shown in Figure 2. The modifications to the architecture that lead to an optimal implementation of Health Binning with Block Grading will be presented later, in Section 3.

Flash storage devices typically have an FTL that performs a number of critical flash management operations. First, an FTL includes a Log Structured Array (LSA) on top of a physical address space, providing the user with a consistent view of the storage device and, at the same time allowing the device to perform maintenance operations without impacting the user. There are many ways to implement an LSA, but a common, simple, and best-performing implementation is through some form of a Logical to Physical Table (LPT). An LPT has an entry for each user page - typically a 4KiB Logical Block Address (LBA), which is translated into a physical location called Physical Block Addresses (PBA), where the page data will be stored. The LPT is accessed on

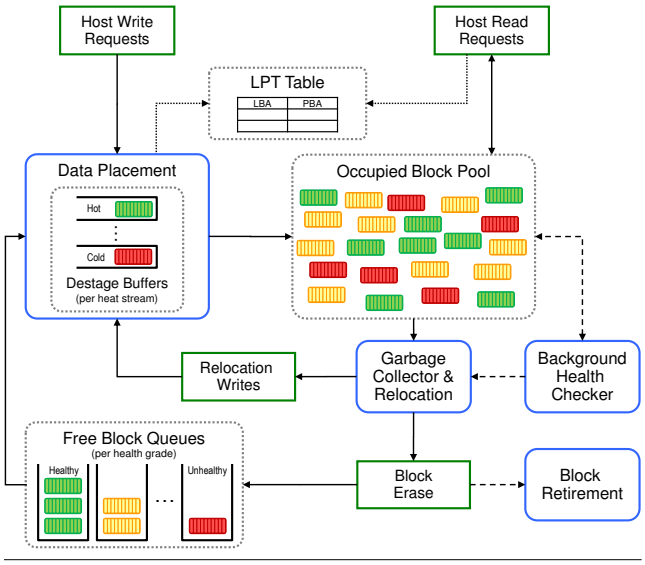


Figure 2. Block-management overview.

every read operation to determine the physical location of the requested data and is updated on every write operation by the data-placement unit.

The use of the LSA transforms the write workload into a sequential write stream of large chunks, preferably striped over all flash channels (and dies) and with added parity protection data for recovery purposes. It also allows the device to have different page sizes for the user and the physical storage. For example, it is possible to compress user data as well as store many logical pages in a physical page. The LSA also enables exposing 4 KiB (or even smaller) logical pages even though the physical flash pages are 8 KiB or 16 KiB.

The data-placement unit decides in real time which blocks should be used for what type of data, such that logical data pages with similar update frequency are collocated into the same heat stream. Doing so reduces write amplification as a large amount of data in the same block is approximately invalidated at the same time, and effectively allows the average update frequency of each block to be adjusted according to its wear, thereby enhancing the endurance and performance of the device. Similarly as in [11], the data-placement unit also updates the heat of an LBA on each write request, such that host writes increase an LBA's heat and relocation writes decrease it.

Today, there are typically 256 or 512 physical pages organized in a flash block. Before writing data, the target location must be erased, an operation that can only be performed at block granularity. Therefore, the data-placement unit draws freshly erased blocks from one of the Free Block Queues (FBQ) into the destage buffers, and separate destage buffers are maintained for each data stream. After all pages in a block have been written, the block is moved from the destage buffer into the occupied block pool.

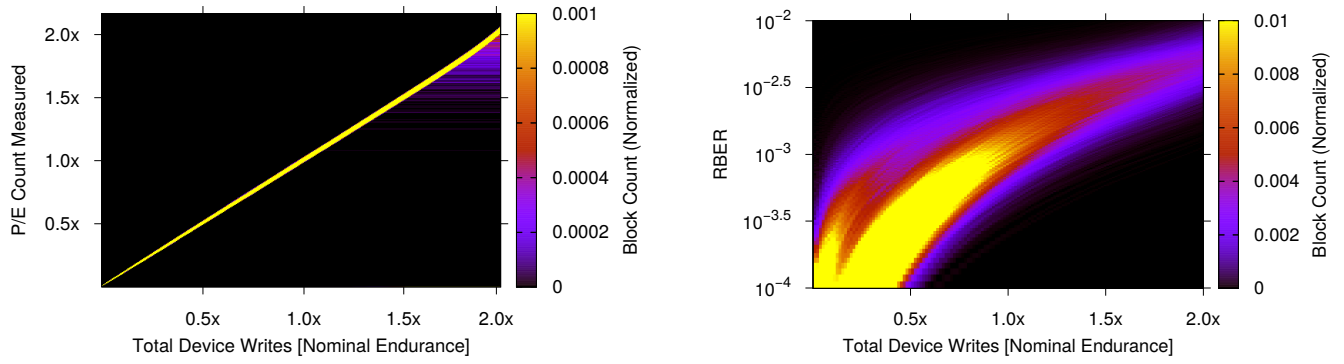
In an LSA, when a user overwrites or trims (deletes or unmaps) data of a logical page, the old data copy is not immediately erased, but only marked as invalid. At some point, nearly all blocks in the device are written with data, and to allow further writes, the GC process chooses a block with a high percentage of invalidated data. The GC issues relocation writes for all still valid data pages, which are then written to some other location by the data-placement unit. The LPT is updated accordingly, and after relocation has completed, the block is erased and placed back into a FBQ (or retired if its worst page exceeds the error-correction capability of the ECC).

Upon initialization, all blocks are erased and placed into the FBQs. A conventional FBQ would be a simple FIFO or LIFO queue. In contrast, in Health Binning with Block Grading, we maintain separate queues for blocks with similar health (i.e., the blocks have the same health grade). At any point in time, the FBQs should have enough blocks to absorb potential bursts of user writes, and GC is triggered when the queue size falls below a critical limit. Once the physical space has almost been filled up for the first time, the number of blocks in the FBQs must not be too large because these blocks do not hold valid data and therefore artificially reduce over-provisioning and increase write amplification [10]. However, the FBQs are necessary if a device is to provide stable and reliable performance.

3. Health Binning with Block Grading

NAND-flash storage devices typically implement static and dynamic Wear Leveling (WL). *Dynamic WL* targets improved endurance by balancing the PECs across blocks. Upon overwrites and relocations, data is placed into free blocks with the least PEC. Although dynamic WL narrows the distribution of PECs across blocks, there still is a huge gap to the actual available endurance, as the same PEC of two blocks does not necessarily indicate that they have the same remaining endurance.

Static WL identifies the least worn blocks holding static data, and relocates still valid data from them. After this, these least worn (healthiest) blocks are available for writing data to them. These least used blocks have a significantly higher remaining endurance than the average, but because of the low number of invalidated pages (there might even be no invalidated page) they have not been identified and selected by GC. Static WL has been found to increase endurance on top of dynamic WL [3, 16]. Static WL is also used to efficiently address data-retention limitations (i.e., the time that data will be reliably readable under the given environmental conditions), a property becoming more and more important because of the restrictive specifications of newer flash generations. It is highly desirable to limit the static-WL activities, (1) to ensure that static data is not moved frequently, and (2) to confine additional write amplification if the selected blocks hold only (or nearly only)



(a) P/E cycle distribution

(b) Block wear distribution.

Figure 3. Conventional wear leveling under a highly-skewed workload (Zipfian 95/20) equalizes the P/E cycle counts (left), but the number of read errors varies significantly across blocks (right). This limits overall endurance and results in increased latencies due to read retries.

valid data [7, 9, 23]. If performed too aggressively, static WL may significantly increase write amplification, reducing the number of writes available to the user (i.e., reduced logical device endurance).

In Figure 3(a), we show a heat map of the PEC across blocks during the lifetime of a device when a conventional WL algorithm that attempts to balance the PEC across all blocks is used. The evaluation was performed by means of simulations using a device populated with 19 nm c-MLC flash chips, and a highly-skewed write workload following a Zipfian distribution where 95 % of the writes are concentrated in 20 % of the device address space. Despite the high skew, the PEC distribution stayed extremely narrow with increasing device writes. The horizontal lines that start to appear after $1.1\times$ device writes and depart from the average curve represent blocks that had to be retired because they had reached the error correction capability of the ECC. The slope of the curve slightly increases towards the end of life as more and more blocks are being retired. This increase further accelerates wear out. However, as illustrated in Figure 3(b), towards device wear-out, there are many blocks that still have significantly better health (lower wear) than others. Once a certain number of blocks has been retired, the device can no longer accept writes, even though many blocks are still healthy. Assuming that a device can only tolerate a few percent of all blocks being retired, device endurance is effectively reduced to less than 60 % of the maximum achievable endurance when all cells would have been used to their outermost limits.

As we have seen earlier in Figures 1 and 3, blocks typically reach their ECC limits at different PECs, thus balancing the PEC across blocks will lead only to limited improvements in terms of endurance. PEC balancing can only yield good endurance when blocks have very similar characteristics, which is not the case for modern NAND flash.

Health Binning targets improving the endurance beyond the capabilities of traditional WL. To do so, Health Binning (1) uses the RBER as a metric for block health, (2) tracks the block health of all blocks periodically and assigns a health grade to each block, and (3) improves endurance using careful data placement rather than by increasing the number of writes in the device that conventional WL techniques do.

In Health Binning, the hottest data is immediately *placed* on the healthiest blocks from the FBQ, and cold data is placed on less healthy blocks. As Health Binning is purely a data-placement algorithm, it does not cause data relocation and does not increase write amplification. Therefore, when the workload exhibits skew, Health Binning can *only improve* endurance.

As there is no change in write amplification, the effects of Health Binning can be measured in either the physical or the logical domain (equivalently). In this paper, we opted for quantifying improvements in physical endurance as it excludes any variations due to GC as well as the size of over-provisioning. Furthermore, in our studies, we analyzed various GC algorithms and configurations, and confirmed that reasonable changes in the GC have only a marginal effect on the relative physical endurance when comparing WL schemes and Health Binning. Therefore, all our results use the same GC and over-provisioning configurations. Below we will present our approach for estimating the block health.

3.1 Estimating the Block Health

As some research groups have recently suggested, it is reasonable to consider the RBER when estimating the block health, rather than the PEC [14, 17, 21], because the acceptable unrecoverable bit-error rate of an ECC algorithms, typically in the range of 10^{-13} to 10^{-16}, is bounded by the RBER (rather than by the block's PEC). In this section we describe the most important challenges associated with health estimation based on RBER, and our approach to solving them.

In Figure 1 we have seen that some blocks (e.g., Block 5) look unhealthy in the beginning (have a higher RBER than others), but become healthier than others (e.g., Block 1) towards the end of life. Therefore, the RBER in early life cannot

be used as an estimator of the overall block endurance. In the Health Binning scheme, we have implemented continuous background scrubbing and monitoring of block health, called Background Health Checker (BGHC), as shown in Figure 2.

As BGHC scrubs through the pages of a block, it determines the RBER of each page in the block. In our real hardware environment we utilize information from the ECC decoder of the flash controller to determine the RBER. Based on the RBER of the worst page in the block, the block is graded (classified) relative to other blocks. Doing so significantly reduces the amount of meta-data that needs to be tracked in the device, as we only have to track a single value for the entire block.

It is important to note that only by measuring the RBER of the worst page for each block individually, outliers in the RBER distributions are taken into account to achieve an acceptable unrecoverable bit-error rate. This is in contrast to recent observations of uncorrectable errors not being correlated with the single RBER reported by SSDs [22].

Overall, blocks should be distributed approximately equally across the different health grades. For example, assuming four health grades, the best 25% of blocks would be in the healthiest grade, the next 25% in a less healthy grade, etc. From our evaluations, a non-balanced distribution into health grades has only minor impact on the results. The number of health grades should be proportional to the number of heat levels being tracked. We also found that having more health grades than heat levels improves endurance only marginally.

3.2 Efficient Block Wear-out

Health grades are used only during data placement (i.e., not in the WL or the GC process). Upon erasing, blocks are placed into the grade's FBQ, so that future data placement can take place in a more efficient way. When data from a particular heat stream needs to be written to a new block, a block is taken from the most appropriate FBQ. As the heat-to-health mapping places the write-hottest data to the healthiest block and less write-hot data to less healthy blocks, the healthiest blocks will naturally see more P/E cycles than the less healthy blocks – simply because the write-hottest data is the most likely to be overwritten by the user.

With Health Binning alone, blocks holding a large amount of, or only static data will not be moved, and static WL is still required. However, we use static WL for the sole purpose of ensuring that blocks do not hold data for longer than the retention time manufacturers recommend, typically a few months. As a consequence, in the worst case, a block will only see at most a few PECs per year due to static WL. This is negligible compared with the wear induced by host workloads.

Next, we show how Health Binning with Block Grading, in the most natural way, achieves an increased wear-out of the healthiest blocks and a reduced wear-out of the less-healthy blocks, and quantify the endurance gains.

4. Evaluation

Here, we first describe the evaluation environment, and then present results comparing Health Binning with other approaches.

4.1 Description of the evaluation environment

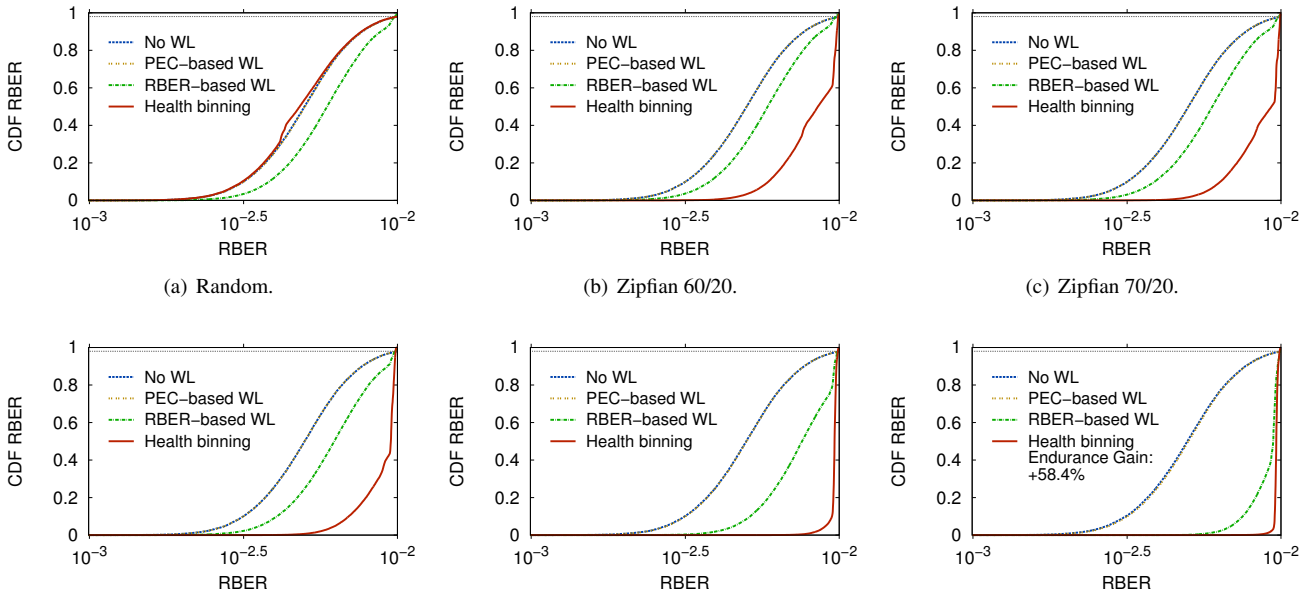
Our evaluations are based on a simulator and a real hardware environment using an FPGA-based flash controller. The hardware environment can, in addition, be operated with real flash chips attached or with emulated flash chips. Both the simulator and the emulated hardware use a flash model described later.

Clearly, improving endurance directly with real flash chips (and boards) takes too much time and budget, so this was not an option. For example, decent SSDs reach 100 k writes per second of 4 KiB data blocks to flash. Assuming an endurance of 3000 PECs, a single test of a device with 1 TB capacity would take approximately 3 months to wear out all blocks. Further, our target boards (on which we can measure the RBER) have significantly higher endurance than conventional consumer SSDs. Therefore, in the scope of this paper we are only able to present results from the simulation and the emulation environments.

However, we have confirmed an extremely similar behavior of the real and the emulated hardware, and also the simulator (all written by different people). This allowed us to perform back-to-back testing across the implementations, thereby reducing the possibility of incorrect conclusions originating from implementation errors. This approach gives us high confidence in our approach and all our implementations.

All results presented in this paper, unless otherwise noted, are based on large scale characterization data from real flash chips, namely 19 nm and 16 nm MLC flash technologies from various manufacturers. For each technology we utilized characterization data from a large number of blocks to build a Gaussian mixture model which allowed us generating block parameters for a much larger amount of blocks [20]. Doing so, we were able to model the RBER of thousands of flash blocks in our simulations. In fact, we evaluated different configurations ranging from 10s to 100s of thousands of flash blocks generated and did not observe significantly different behavior for the various configurations. In other words, Health Binning could be beneficial to most, if not all, NAND-flash controller designs. As in our simulators the properties of the flash blocks are modeled, no real data is actually being written, but all the flash-management functions outlined in Figure 2 are implemented.

Once a few percent of the blocks have been retired as they had reached the error correction capability of the ECC, write amplification jumps abruptly, and the performance of the device drops suddenly. We therefore decided to perform our evaluations only up to the moment when 2 % of the total number of flash blocks get retired. As flash controllers typically use RAID-like parity schemes, this limit leaves just



(a) Random. (b) Zipfian 60/20. (c) Zipfian 70/20.

(d) Zipfian 80/20. (e) Zipfian 95/20. (f) Zipfian 98/20.

Figure 4. CDF of block wear at the end of life for different write workloads on a 19nm cMLC device. The horizontal dotted line at 98 % block wear marks the block retirement limit.

enough over-provisioning to handle the failure of an entire flash chip.

We used both uniform and skewed workloads in the evaluation. Uniform-random write workloads are typically the worst workload for flash-based storage devices because blocks that are garbage-collected will on average, see the lowest number of invalidated pages. In contrast, it is a known fact that real-life workloads are not purely uniform and that some LBAs are more likely to be accessed than others [8, 13]. Workloads following a Zipfian or Pareto distribution [1] follow this principle and are therefore a much more realistic class of workloads with the advantage of having an exact mathematical definition compared with real-world traces. We also evaluated sequential writes workloads, but do not show the results here, because for those workloads the maximum benefit is obtained even with a simple 1-grade Health Binning scheme and by always taking the healthiest of the free blocks.

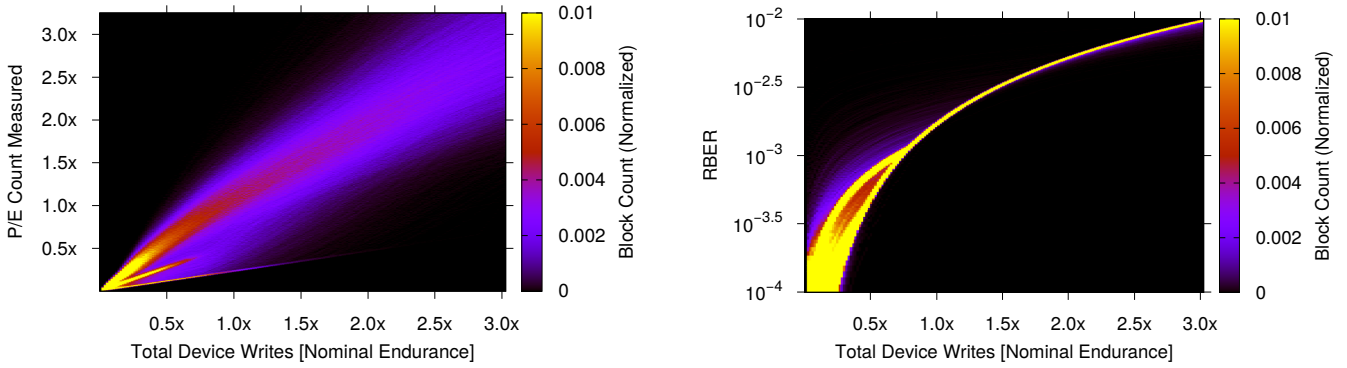
4.2 Impact of Workload Skew on Endurance

Figure 4 compares the Cumulative Distribution Function (CDF) of the measured RBER at end of the simulation runs for a set of different workloads and a 19 nm cMLC NAND flash model using no WL, traditional WL based on PEC and RBER balancing, and Health Binning. Because the type of workloads used here accesses every LBA sooner or later, static WL rarely gets an opportunity to relocate data. Therefore, only dynamic WL is actively used.

For a uniform random workload, the best strategy is to always pick the free block with the lowest RBER for placing new writes as is done by the RBER-based WL algorithm, which results in roughly 10 % additional endurance compared with the other schemes. In Figure 4(a) the endurance gain corresponds to the area between RBER-based WL curve and the curves of other WL schemes. Health Binning is not able to narrow the RBER distribution because the update frequencies of all LBAs have the same average, therefore segregating writes is ineffective: In fact, the segregation of writes results in less good blocks being selected for placing data which supposedly seems to be colder (i.e., because it had been relocated by GC) but in reality has the same expected update frequency than other writes due to the workload characteristics. This inaccurate placement decision ultimately

Algorithm	Endurance gain over no WL					
Technology	Random	Zipfian				
		60/20	70/20	80/20	95/20	98/20
WL PEC:						
A: 19nm cMLC	0.24 %	−0.05 %	0.07 %	0.00 %	0.10 %	0.48 %
B: 19nm eMLC	0.01 %	0.16 %	0.22 %	0.35 %	0.86 %	1.21 %
C: 16nm cMLC	0.05 %	−0.76 %	−0.76 %	−0.79 %	−0.55 %	−0.54 %
WL RBER:						
A: 19nm cMLC	9.91 %	9.22 %	10.61 %	13.22 %	26.19 %	45.42 %
B: 19nm eMLC	1.38 %	2.38 %	2.52 %	4.90 %	27.36 %	44.65 %
C: 16nm cMLC	11.08 %	18.14 %	19.76 %	23.12 %	47.95 %	70.87 %
Health Binning:						
A: 19nm cMLC	−0.53 %	33.18 %	37.67 %	42.68 %	56.44 %	58.79 %
B: 19nm eMLC	−0.12 %	19.17 %	23.91 %	21.03 %	50.94 %	51.70 %
C: 16nm cMLC	−0.39 %	46.34 %	53.46 %	60.33 %	78.55 %	78.80 %

Table 1. Endurance improvement over no WL, for uniform random and skewed write workloads.



(a) P/E cycle distribution.

(b) Wear distribution.

Figure 5. P/E cycle and wear distributions using Health Binning under a Zipfian 95/20 workload.

results in slightly lower endurance compared to no WL. Although having the lowest endurance, uniform random-write workloads over the entire address space of the device are however not likely to be encountered in real-world usage over prolonged periods of time.

As soon as the workload exhibits skew, segregating based on the update frequency quickly pays off: Already with a lightly skewed Zipfian 60/20 workload, where 60 % of the writes target 20 % of the address space, Health Binning significantly outperforms other WL schemes, reaching 33.18 % additional endurance. With increasing skew, heat segregation separates hot from cold data better, further facilitating the operation of Health Binning. Beside Health Binning, RBER-based WL also achieves some endurance gains with increasing skew because write-hot data becomes more common than colder data, and thus an increased wear-out of the best blocks (by placing more and more hot data to them) becomes more likely. The absolute endurance gains are, however, significantly lower than with Health Binning.

In contrast, there is no visible difference between no WL and PEC-based WL. Figure 3(a), shows that, despite the high skew, the PEC distribution is very narrow for PEC-based WL, as would be expected from such a scheme. This means that the scheme operates properly, but nevertheless sub-optimally for the real flash chips. This conforms with observations made by other research groups [3, 16].

For the extremely-skewed Zipfian 98/20 workload, the RBER distribution is also extremely narrow. This means that Health Binning was able to utilize the healthiest blocks much more than the least healthy blocks, and in the end all blocks reached their ECC correction capability almost at the same time. Therefore, the device exhibited extremely stable performance and the highest endurance. Moreover, the point in time at which the device will have to be replaced owing to wear-out becomes also more predictable.

An overview of different flash devices and their measured endurance gains is given in Table 1. We used large-scale flash characterization data from three different flash technologies to model the block behavior in the evaluation. The table illustrates that Health Binning outperforms all other wear-leveling schemes for any skewed workload, but also that the endurance gains can vary significantly between different technologies. The results further indicate that, because of the increasing variability in the block characteristics of smaller technology nodes, also the endurance improvements from Health Binning and RBER-based WL generally increase. Overall, the qualitative statements and observations made above with respect to the results in Figure 4 are also valid for the other flash technologies presented in the table.

4.3 Analysis of Health Binning

Let us now focus on the heavily-skewed Zipfian 95/20 write workload using Health Binning. Figure 5 shows the PEC and the RBER distributions for the entire evaluation instead of only reporting them at the end of life. We use the same device as introduced in Section 3, which is populated with 19 nm c-MLC flash chips.

Clearly, Health Binning narrows the RBER distribution effectively already very early in the device lifetime and keeps it narrow (Figure 5(b)). The distribution widens only marginally towards the end of the evaluation, demonstrating the effectiveness of Health Binning, which results in the high endurance gains over other WL schemes presented above. At the same time, the PEC distribution (shown in Figure 5(a)) widens soon after the start, owing to the different wear properties of individual blocks. This is in contrast to the PEC-based WL results presented in Figure 3. With Health Binning, the best blocks endure roughly 2.5× more PECs than the worst ones at the end of life.

We then deliberately reduced the number of distinct block parameters to only 100 different types of blocks without reducing the total number of blocks, as we wanted to study the ability of Health Binning to balance changes in block wear characteristics rapidly. This is important because blocks in real flash devices may behave significantly differently at distinct points of their lifetime as we showed in Figure 1

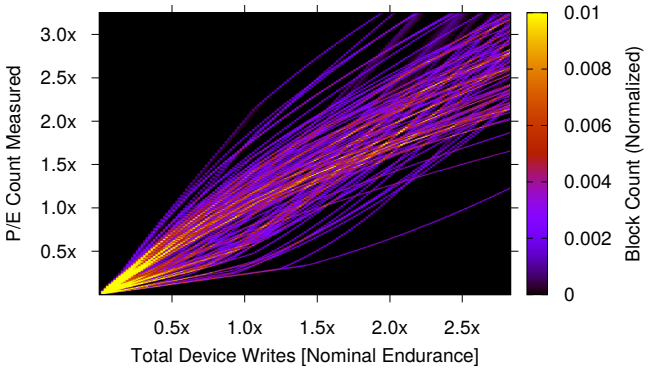


Figure 6. P/E cycle distribution using Health Binning with 100 different block parameters and a Zipfian 95/20 workload.

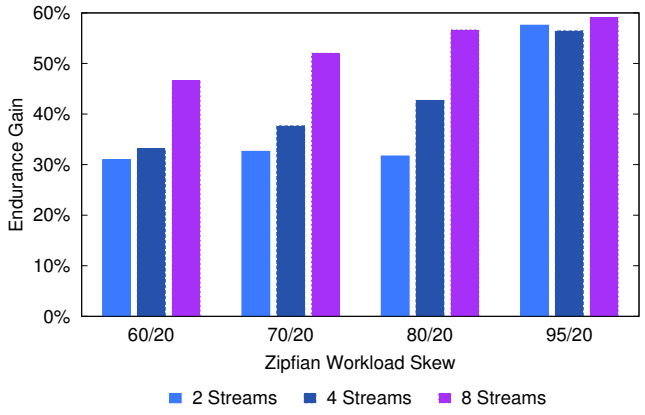


Figure 7. Endurance gain as a function of the number of streams for different workload skews.

where some blocks appear to age quickly at the beginning, but last significantly longer than other blocks that were initially very healthy but later on quickly saw accelerated wear-out compared to the average block behavior. All other parameters are kept unchanged.

The evaluation results are presented in Figure 6. Here, the PEC evolution of individual block types can be clearly identified by the different curves: Blocks that exhibit a decrease in the slope with increasing number of total device writes correspond to blocks that initially manifested a lower RBER than the average block, but later on turned out to be less good blocks. In contrast, blocks with a slowly ascending slope at the beginning, but a steadily increasing slope with increasing number of total device writes, can be identified as initially underestimated blocks with high endurance. The change in the slope of the curves demonstrates the virtue of quickly adapting to intrinsic block properties.

4.4 Health Binning Segregation Granularity

The endurance gain obtained from Health Binning also depends on the granularity at which the data-placement unit segregates write operations based on their update frequency. This granularity corresponds to the number of streams that are maintained. In our simulations we maintain separate streams for relocation and host write requests to capture the workload-dependent differences in the temporal locality of these types of write requests better.

An overview of the endurance gains measured is illustrated in Figure 7 for different number of streams ranging from 2 to 8 streams. The presented results are based on the 19 nm cMLC flash devices. The benefit from more streams is highest with low to moderately skewed workloads (Zipfian 70/20 and Zipfian 80/20). For highly skewed workloads (i.e., Zipfian 95/20), Health Binning already achieves more than 95 % of the possible endurance, hence the additional benefits of having more streams is less pronounced.

Intuitively, one can say that, overall, endurance should increase with more streams. However, as the use of more streams also increases the number of flash blocks being assigned for data placement at any point in time, the over-provisioning is artificially reduced by those blocks, which consequently leads to increased write amplification and less logical endurance on the higher level. Also, more streams typically require more DRAM for destaging buffers which may not be available in flash controllers.

5. Related Work

The physical reasons behind the RBER, such as program disturb, quantum-level noise effects, erratic tunneling, data retention, read disturb, etc., have been analyzed by Mielke et al. [14]. The threshold voltage distribution has a strong effect on the observed RBER, and as such has been studied extensively [4, 5], primarily as a way to predict future flash behavior and to design more effective error-tolerance mechanisms. Papandreou et al. [18] go further and propose dynamic adjustment of the read voltage thresholds, to minimize the RBER dynamically. Although we do not discuss these advanced techniques for improving endurance in the paper, we have verified in simulations that our result do not change when such techniques are used. Recently, Schroeder et al. [22] performed a large scale analysis on reliability of SSDs in the field and found that the RBER and uncorrectable errors reported by the SSDs are not correlated. Although it is unknown how the drives internally determine the RBER this could be a product of PEC-based WL and outliers in the RBER distribution.

Based on a competitive analysis, Ben-Aroya et al. [2] conclude that GC and block erasure policies are best separated from WL, and that heuristics for predicting future request sequences might be able to improve endurance, if the algorithm considers all three policies.

A simple way to address limited endurance is to increase over-provisioning or add a large write cache [24]. Increasing the actual physical flash capacity without making this additional space available to the user can greatly reduce write amplification, but significantly increases the cost per GiB of the device [10].

Chang et al. [7] introduce a static WL mechanism which balances the PECs across blocks. Their main objective is to force the relocation of cold data after a given period of time has passed. They observe the problem that placing hot and cold data together results in more relocations, and assume (but do not describe) the existence of dynamic WL in the FTL. They also do not describe how heat is determined and whether heat segregation is performed.

Different studies exist that leverage segregation of data according to its update frequency upon data placement. Hu et al. [11] propose a scheme called Container Marking which combines data placement and segregation with PEC-based WL to reduce write amplification. Min et al. [15] introduce a flash-aware file system organized in a log-structured fashion that separates data into four streams.

Several approaches can be found in the literature that try to leverage the RBER to improve flash reliability. But all of them lead to increased write amplification. Cai et al. [6] propose a data refresh algorithm for retention errors due to charge loss in flash cells. The algorithm periodically reads entire flash blocks to determine their RBER, and, if needed, corrects and reprograms them (in-place). Pan et al. [17] describe how under the same program-erase cycling, blocks can have very different RBER. They evaluate uniformly random workloads and propose a modified GC scheme that selects blocks based primarily on the RBER. They show that such a scheme can noticeably improve the efficiency of WL, but at the cost of increased write amplification. Later, Peleato et al. [21] went one step further and proposed an RBER-based WL scheme. Besides measuring the RBER, they also use the page program time and the PEC to build a linear model to predict the error rate of blocks. The WL algorithm consists of maintaining a list of the coldest blocks in the occupied block pool and exchanging the healthiest of the coldest blocks with the most worn blocks in the free block pool. Clearly, this implies additional writes that increase write amplification. Nevertheless, using this approach, they report improvements over PEC-based WL. In contrast to these approaches, *Health Binning does not change write amplification*, because it changes only the assignment of already erased blocks to data streams.

6. Conclusions

Endurance and performance are of critical importance, and sustaining those with next-generation flash is imperative. We have observed significant differences in the endurance of flash blocks of the same device types with recent flash technologies. Unfortunately, as we showed, traditional WL approaches cannot mitigate these deficiencies.

In this paper, we presented a new flash management technique called Health Binning that uses the RBER of blocks as a metric of their health (wear) to make informed decisions with respect to data placement and wear leveling, instead of relying on just the PEC of blocks. As the technique neither interferes with GC nor causes any additional writes, it does not affect write amplification – a property we have not seen elsewhere in the previous research on the health management for NAND flash. Based on our results, we expect that in real-world workloads, which typically exhibit high skew in I/O access patterns, Health Binning can increase endurance by up to 80 %, depending on the flash technology used. Health Binning therefore helps closely achieve the average endurance of all blocks. A non-negligible side effect is that the reduction achieved in block-health variance additionally reduces the number of read retries and RAID reconstructions (inside an SSD or on the higher level) as the device ages. However, this has not been quantified in this paper because it heavily depends on the utilized error-correction techniques and hence is out of the scope of WL techniques.

Device endurance characterization is a time-consuming process even when done by means of simulations. Therefore we did not include results from any 3D-NAND flash chips for which characterization data is currently emerging. Based on the recent research on 3D-NAND flash [12], we are very confident that our methods can seamlessly operate also with such devices. We believe that Health Binning will become an indispensable part of flash management regarding the latest technology trends in NAND flash, and may be applicable to other emerging non-volatile memory technologies in the future. Last but not least, the results presented in this paper are in line with a real implementation of Health Binning we integrated into one of our commercially available product line.

Acknowledgments

We gratefully acknowledge Thomas Parnell and Nikolaos Papandreou, from IBM Research – Zurich, for providing insights into flash characterization and modeling, and the teams at IBM Systems for their support. Further, we would like to thank Nikolas Ioannou, from IBM Research – Zurich, for his contributions to one of the simulation environments. Special thanks go to Evangelos Eleftheriou, from IBM Research – Zurich, for his support.

References

[1] B. C. Arnold. *Pareto distribution*. Wiley Online Library, 1985.

[2] A. Ben-Aroya and S. Toledo. Competitive analysis of flash-memory algorithms. In *Proceedings of 14th Annual European Symposium on Algorithms*, ESA '06, pages 100–111, Sept. 2006.

[3] S. Boboila and P. Desnoyers. Write endurance in flash drives: Measurements and analysis. In *Proceedings of the 8th USENIX Conference on File and Storage Technologies*, FAST '10, pages 115–128, 2010.

[4] Y. Cai, E. F. Haratsch, O. Mutlu, and K. Mai. Threshold voltage distribution in MLC NAND flash memory: Characterization, analysis, and modeling. In *Proceedings of the Conference*

on Design, Automation and Test in Europe, DATE '13, pages 1285–1290, Mar. 2013.

[5] Y. Cai, O. Mutlu, E. F. Haratsch, and K. Mai. Program interference in MLC NAND flash memory: Characterization, modeling, and mitigation. In *Proceedings of the IEEE Conference on Computer Design*, ICCD '13, pages 123–130, Oct. 2013.

[6] Y. Cai, G. Yalcin, O. Mutlu, E. F. Haratsch, A. Cristal, O. Unsal, and K. Mai. Flash correct-and-refresh: Retention-aware error management for increased flash memory lifetime. In *Proceedings of the IEEE Conference on Computer Design*, ICCD '12, pages 94–101, Sept./Oct. 2012.

[7] L.-P. Chang. On efficient wear leveling of large-scale flash-memory storage systems. In *Proceedings of the 2007 ACM Symposium on Applied Computing*, SAC '07, pages 1126–1130, Mar. 2007.

[8] L. Cherkasova and M. Gupta. Analysis of enterprise media server workloads: Access patterns, locality, content evolution, and rates of change. *IEEE/ACM Trans. Netw.*, 12(5):781–794, Oct. 2004.

[9] E. Gal and S. Toledo. Algorithms and data structures for flash memories. *ACM Computing Surveys*, 37(2):138–163, June 2005.

[10] X.-Y. Hu, E. Eleftheriou, R. Haas, I. Iliadis, and R. Pletka. Write amplification analysis in flash-based solid state drives. In *Proceedings of the Israeli Experimental Systems Conference*, SYSTOR '09, pages 10:1–10:9, May 2009.

[11] X.-Y. Hu, R. Haas, and E. Eleftheriou. Container marking: Combining data placement, garbage collection and wear levelling for flash. In *Proceedings of the 19th IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems*, MASCOTS '11, pages 237–247, July 2011.

[12] J.-H. Lee, S.-M. Joe, and H.-J. Kang. Characterization of reliability in 3-D NAND flash memory. In *Proceedings of IEEE International Conference on Solid-State and Integrated Circuit Technology*, ICSICT '14, pages 1–4, Oct. 2014.

[13] S.-W. Lee and B. Moon. Design of flash-based DBMS: An in-page logging approach. In *Proceedings of the 2007 ACM SIGMOD International Conference on Management of Data*, SIGMOD '07, pages 55–66, June 2007.

[14] N. Mielke, T. Marquart, N. Wu, J. Kessenich, H. P. Belgal, E. Schares, F. Trivedi, E. Goodness, and L. R. Nevill. Bit error rate in NAND flash memories. In *Proc. 46th Annual International Reliability Physics Symposium*, IRPS '08, pages 9–19, Apr./May 2008.

[15] C. Min, K. Kim, H. Cho, S.-W. Lee, and Y. I. Eom. SFS: Random write considered harmful in solid state drives. In *Proceedings of the 10th USENIX Conference on File and Storage Technologies*, FAST '12, pages 12–12, Feb. 2012.

[16] M. Murugan and D. H. C. Du. Rejuvenator: A static wear leveling algorithm for NAND flash memory with minimized overhead. In *Proceedings of the 27th IEEE Symposium on Mass Storage Systems and Technologies*, MSST '11, pages 1–12, May 2011.

[17] Y. Pan, G. Dong, and T. Zhang. Error rate-based wear-leveling for NAND flash memory at highly scaled technology nodes. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 21(7):1350–1354, July 2013.

[18] N. Papandreou, T. Parnell, H. Pozidis, T. Mittelholzer, E. S. Eleftheriou, C. J. Camp, T. J. Griffin, G. A. Tressler, and A. A. Walls. Using adaptive read voltage thresholds to enhance the reliability of MLC NAND flash memory systems. In *Proceedings of the 24th ACM Great Lakes Symp. on VLSI*, GLSVLSI '14, pages 151–156, May 2014.

[19] K.-T. Park et al. Three-dimensional 128Gb MLC vertical NAND flash memory with 24-WL stacked layers and 50MB/s high-speed programming. *IEEE Journal of Solid-state Circuits*, 50(1):204–213, Jan. 2015.

[20] T. Parnell, N. Papandreou, T. Mittelholzer, and H. Pozidis. Modelling of the threshold voltage distributions of sub-20nm NAND flash memory. In *IEEE Global Communications Conference*, GLOBECOM '14, pages 2351–2356, Dec. 2014.

[21] B. Peleato, H. Tabrizi, R. Agarwal, and J. Ferreira. BER-based wear leveling and bad block management for NAND flash. In *IEEE International Conference on Communications*, ICC '15, pages 295–300, June 2015.

[22] B. Schroeder, R. Lagisetty, and A. Merchant. Flash reliability in production: The expected and the unexpected. In *Proceedings of the 14th USENIX Conference on File and Storage Technologies*, FAST '16, pages 67–80, Feb. 2016.

[23] S. E. Wells. Method for wear leveling in a flash EEPROM memory. Patent, US 5,341,339, 1994.

[24] J. Yang, N. Plasson, G. Gillis, and N. Talagala. HEC: Improving endurance of high performance flash-based cache devices. In *Proceedings of the 6th International Systems and Storage Conference*, SYSTOR '13, pages 10:1–10:11, June/July 2013.

[25] K. Zhao, W. Zhao, H. Sun, T. Zhang, X. Zhang, and N. Zheng. LDPC-in-SSD: Making advanced error correction codes work effectively in solid state drives. In *Proceedings of the 11th USENIX Conference on File and Storage Technologies*, FAST '13, pages 243–256, Feb. 2013.