SPA-SSD: Exploit Heterogeneity and Parallelism of 3D SLC-TLC Hybrid SSD to Improve Write Performance

Wenhui Zhang*, Qiang Cao*¶, Hong Jiang†, Jie Yao‡, Yuanyuan Dong§ and Puyuan Yang§
*Wuhan National Laboratory for Optoelectronics, Key Laboratory of Information Storage System,
*Engineering Research Center of Data Storage Systems and Technology, Ministry of Education of China,
*Huazhong University of Science and Technology, Wuhan, China
¶Corresponding Author: caoqiang@hust.edu.cn
†Department of Computer Science and Engineering, University of Texas at Arlington, Arlington, Texas
‡School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan, China
§Alibaba Group., Hangzhou, China

Abstract*—To address the write performance problem suffered by MLC/TLC flash, researchers have proposed hybrid SSD that aims to combine the strengths of SLC flash, used as the write-buffer zone for its superior write performance, and MLC/TLC flash, as the capacity zone for its high storage density. While leveraging SLC as a physical write-buffer zone is proven effective in traditional 2D hybrid SSDs, how to effectively incorporate SLC into a 3D-stacked TLC to form a hybrid SSD has not been studied to the best of our knowledge. Yet this is a timely and important performance issue for 3D-stacked TLC given its one-shot programming scheme that results in much worse write performance than the programming scheme in 2D TLC where pages are associated with different bits of a cell and programmed in sequence separately. We believe that naively adopting the two-physical-zone approach to 3D hybrid SSD will miss a great opportunity for performance optimization because it ignores the inherent four-level parallelism (channel/chip/die/plane) of the flash chip array. To this end, we propose in this paper an *S*LC and *P*arallelism *A*ware hybrid SSD (SPA-SSD) to take full advantages of SLC's superior write performance, the internal multi-level parallelism of SSD, and the high storage density of 3D-stacked TLC flash. Two novel techniques enable SPA-SSD to be highly effective: (1) Type-Parallelism Joint Page Allocation (TPJ-PA), which allocates pages for write transactions according to not only available SLC pages but also parallelism to maximize resource utilization within the hybrid SSD, and (2) Queue-length and Parallelism Constrained Data Migration (QPC-DM), which triggers data migration without degrading user write performance by analyzing the device queue length and available flash resources. To evaluate performance of SPA-SSD, a hybrid SSD simulator, called HybridSim, is developed based on MQSim. Experimental results on HybridSim show that TPJ-PA improves write throughput by 60%, while QPC-DM improves write throughput by up to 10 times. Besides, trace-driven experiments on HybridSSD demonstrate that SPA-SSD improves the write latency to the flash by up to two orders of magnitude over the state-of-the-art designs.

I. Introduction

Due to their very high storage density, 3D-stacked NAND TLC (triple-level cell) flashes have become the mainstream storage medium of solid-state drives (SSD) in mass market[1]. Each TLC cell not only stores three binary data bits but also allows vertical stacking in multiple layers (e.g., 96), providing higher storage density than traditional 2D flash. The three bits of a TLC cell, referred to as LSB, CSB, MSB, are associated with three different pages, where a page is the minimum unit for read and write (program) operations in SSDs. For 2D TLC flash, the three pages sharing the same cells are programmed one-by-one with different latency (e.g., 500μs, 2000μs, 5500μs for LSB, CSB, MSB, respectively[2]), referred to as *page-by-page programming*. Differently, for CT(charge trap)-based 3D TLC flash, the three pages sharing the same cells, known as a page set, are programmed together at once, referred to as *one-shot programming*, a programming method that is the most widely used among the 3D TLC-based products on the market[3][4]. The delay of one-shot programming is determined by not only program latency (usually longer than the MSB program latency in page-by-page programming) but also waiting time to gather full page sets, leading to poorer, often much poorer, write performance than one-by-one programming.

Many approaches have been proposed in designing SLC-MLC/TLC hybrid SSD, where an SLC tie is introduced into MLC/TLC SSD as a write buffer to absorb user writes rapidly, improving the overall write performance[5][6][7][8][9][10]. Unfortunately, to the best of our knowledge, how to effectively address the aforementioned poor write-performance problem in 3D TLC flashes employing one-shot programming has not yet been discussed in the literature. In the existing studies, the SLC tie is often regarded as a centralized zone constructed of multiple SLC chips while the TLC tie forms the storage capacity zone, that is, the whole storage space is divided into two big zones[11] both physically and logically. While the two-zone design simplifies the management of flash resources, it neglects the inherent four-level internal parallelism (channel/chip/die/plane) of the flash chip array, missing a great opportunity to further improve performance of hybrid SSD.

Modern TLC flash chips generally offer an SLC mode to operate a TLC block as an SLC block, improving performance

978-1-5386-6648-7/19/$31.00 ©2019 IEEE
DOI 10.1109/ICCD46524.2019.00088

by sacrificing capacity. This suggests that SLC blocks can be distributed in every channel, every chip, every die, and every plane, which opens up a new opportunity to fully exploit structural heterogeneity and high internal parallelism inherent in hybrid SSDs. Clearly, with a physically distributed SLC tie across all four levels of internal SSD parallelism, the page allocation strategy, which determines which channel/chip/die/plane/block/page to serve a user write, and the data migration policy, which determines when to flush data from the SLC tie (write buffer) to the TLC tie to reclaim SLC pages, must be accordingly rethought and redesigned.

In this study, we first analyze the potential parallelism and data migration/flushing cost under various flash layouts, which are categorized by how the SLC tie is physically distributed in an SLC-TLC hybrid SSD. Based on our analysis and experimental observations, *plane-level distribution*, which distributes SLC blocks and TLC blocks uniformly among all planes, exhibits the highest parallelism and the lowest data migration cost. Then we design a novel 3D SLC-TLC hybrid SSD with *plane-level distribution* flash layout, named ***S**LC and **P**arallelism **A**ware hybrid SSD* (**SPA-SSD**). Specifically, to accommodate the one-shot programming scheme of 3D TLC, and to fully exploit both heterogeneity and parallelism in the 3D SLC-TLC hybrid flash chip array, two key enabling techniques are introduced in SPA-SSD. The first, referred to as ***T**ype-**P**arallelism **J**oint **P**age **A**llocation* (**TPJ-PA**), allocates pages for writes based on both the available SLC pages and parallelism to maximize resource utilization within the hybrid SSD. The second, called ***Q**ueue-length and **P**arallelism **C**onstrained **D**ata **M**igration* (**QPC-DM**), judiciously triggers data migration/flushing to effectively reclaim SLC pages by analyzing the device queue length and available flash resources, without impacting the front-end user writes.

In summary, we make the following contributions:

1) We divide flash layouts in SLC-TLC hybrid SSDs into four categories with respect to SLC flash distribution across TLC flash, and analyze their potential parallelism and data migration cost.
2) We present SPA-SSD, a 3D SLC-TLC hybrid SSD that fully exploits heterogeneity as well as parallelism inherent in such hybrid SSDs. Two enabling techniques, a novel page allocation strategy called TPJ-PA and a novel data migration policy called QPC-DM, make SPA-SSD highly effective. While TPJ-PA allocates planes for user write transactions by considering both available SLC pages and parallelism, QPC-DM performs data migration and serves user requests simultaneously and efficiently.
3) We develop a simulator HybridSim[1], based on MQSim[12], to simulate 3D SLC-TLC hybrid SSD.
4) We evaluate SPA-SSD on HybridSim. Experimental results show that both TPJ-PA and QPC-DM improves write performance significantly. Specifically, compared with the state-of-the-art techniques, TPJ-PA improves write throughput by 60% while QPC-DM improves write throughput by up to 10 times. Simulation results driven by real workloads show that SPA-SSD improves write latency by up to two orders of magnitude over the state-of-the-art hybrid SSD designs.

[1]It is open-sourced at https://github.com/Singularity0817/HybridSim.

The remainder of this paper is organized as follows. Background on 3D TLC and SLC-TLC hybrid SSD designs, and our motivation are presented in Section II. The insight that motivates us to redesign data migration in hybrid SSD is presented in Section III. The design of SPA-SSD is elaborated in Section IV. The evaluation setup and results are presented in Section V and Section VI, respectively. Related works are discussed in Section VII. Conclusions are made in Section VIII.

II. Background and Motivation

Given the focus of this study on improving the write performance of 3D SLC-TLC hybrid SSD, this section presents the necessary background about CT(charge trap)-based 3D TLC flash (Subsection II-A) and insights into SLC-TLC hybrid SSD designs (Subsection II-B) to motivate the SPA-SSD design.

A. 3D TLC and One-Shot Programming

CT-based 3D TLC flashes are emerging to become mainstream media of SSD in recent years because of their high storage density. In contrast to 2D TLC in which pages within one wordline are programmed page-by-page, 3D TLC usually uses the *one-shot programming* scheme, where the three pages within one wordline, known as a *page set*, are programmed together at once[3][4].

The one-shot programming scheme helps to improve the write throughput when writes are extremely intense, because the program latency of a page set alone is shorter than the total program latency of three pages with page-by-page programming scheme. However, the total latency of writing a page set with one-shot programming consists of not only the program latency but also the wait time before a full page set is gathered to trigger the one-shot program operation. Unfortunately, the wait time can be both long and unpredictable, particularly when writes are not intense. To prevent data in volatile DRAM from being lost due to power failure, it is highly desirable to program data into nonvolatile flash as soon as possible. Therefore, user data can stay in DRAM for only a limited period of time, known as *program delay*, after which, data should be programmed immediately.

To accommodate the one-shot programming scheme in CT-based 3D TLC, 3D TLC SSDs usually employ specific dynamic page allocation strategies, which aim to form a page set in one plane as soon as possible[4], rather than the traditional page allocation strategies, which aim to distribute user writes to every channel/chip/die/plane[13][14].

B. SLC-TLC Hybrid SSD

In Fig. 1, we present an architectural overview of SLC-TLC hybrid SSD. Different from a conventional SSD, the flash chip array of an SLC-TLC hybrid SSD is composed of two kinds of flash, SLC and TLC, where SLC flash has low storage density but high write performance, while the exact opposite

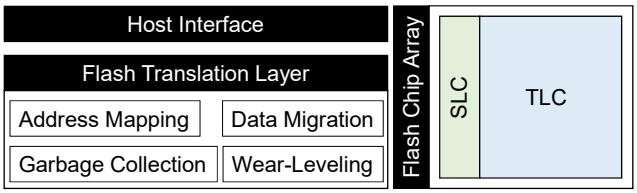


Fig. 1. An architectural overview of SLC-TLC hybrid SSD.

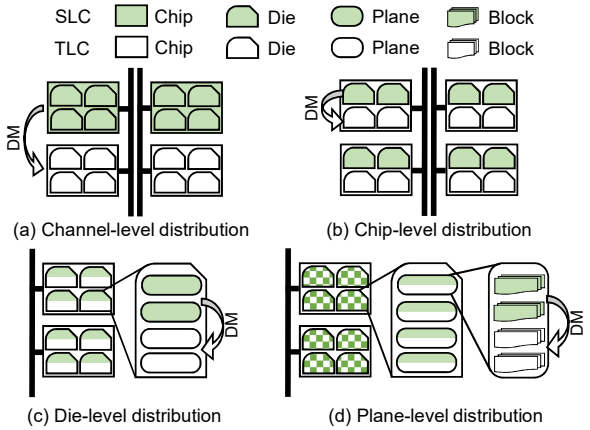


Fig. 2. Four possible flash layouts. The components in green indicate SLC flash while the components in white indicate TLC flash. The arrows indicate the directions of data migration.

is true for TLC flash. By combining SLC and TLC into one SSD, the hope is that both high write performance and high storage density can be achieved simultaneously. The SLC tie in hybrid SSD is usually used as write buffer because of its superior write performance. When the SLC tie is full, data in SLC tie are flushed to the TLC tie and reclaim the SLC pages for future writes. This procedure is known as data migration.

In the following, we first discuss how **flash layout** of flash chip array impacts the design of SLC-TLC hybrid SSD. Then we discuss the state-of-the-art **page allocation strategies** and **data migration policies** in designing hybrid SSD.

1) ***Flash Layout****:* In existing hybrid SSD designs, the storage space is physically and logically divided into the SLC zone and the TLC zone, where each of the two zones is composed of multiple flash chips in the flash chip array[11]. This design is based on the assumption that a flash chip can be functionally either SLC or TLC, which simplifies the design. However, it misses the opportunity to exploit the four-level internal parallelism (channel/chip/die/plane) of the flash chip array. Furthermore, modern TLC flash chips provide an SLC mode to operate TLC flash as SLC flash at the block level, making it possible to distribute SLC blocks across any and/or all of the channel, chip, die and plane levels to gain high access parallelism.

Based on in which of the four SSD levels SLC flash is distributed among TLC flash, we divide possible *flash layouts* in SLC-TLC hybrid SSD into four categories: *channel-level distribution*, *chip-level distribution*, *die-level distribution*, and *plane-level distribution*. Specifically, the *channel-level distribution* layout distributes SLC flash as individual chips across the channels, the *chip-level distribution* layout distributes SLC flash as individual dies on the chips, the *die-level distribution* layout distributes SLC flash as individual planes on the dies, and the *plane-level distribution* layout distributes SLC flash as individual blocks in the planes, as illustrated in Fig. 2.

Due to different distributions of SLC flash, the four flash layouts exhibit different SLC write parallelism and data migration cost. In particular, for *plane-level distribution*, as each plane has SLC pages, SLC writes can fully utilize the channel/chip/die/plane level parallelism to gain extremely high write performance; on the contrary, for *channel-level distribution*, most chips do not provide SLC pages, SLC write can only utilize the channel level parallelism. In addition, data migration in *plane-level distribution* can be done by intra-plane copyback, which occupies only one plane; in contrast, data migration in *channel-level distribution* needs inter-chip data transfer, which occupies two chips and one channel and may degrade front-end user write performance significantly.

In summary, from the perspectives of SLC write parallelism and data migration cost, *plane-level distribution* is the best layout for SLC-TLC hybrid SSD. Unfortunately, existing designs that use SLC chips (hard partitioning or soft partitioning[11]) to construct the SLC zone turn out to be *channel-level distribution* that has the lowest SLC write parallelism and the highest data migration cost.

2) ***Page Allocation Strategy****:* Page allocation is an integral part of address mapping that determines the physical page addresses (including channel/chip/die/plane/block/page IDs) for user write transactions (i.e., page-sized user write requests). For traditional SSDs with a single flash type, static or dynamic page allocation strategy is often used to allocate pages for maximum access parallelism[14]. These page allocation strategies, collectively referred to as *Parallelism-First Page Allocation* (**PF-PA**) in this study, can also be employed in hybrid SSD. Because of the page-type heterogeneity in hybrid SSD, it is the flash type, rather than parallelism, that is the prime consideration when allocating pages in hybrid SSD. The TurboWrite technology, for example, always allocates SLC pages for user write transactions as long as SLC pages are available in the device[15], a strategy we refer to as *TurboWrite Page Allocation*(**TW-PA**) in this study. On the other hand, several studies advocate the hot-cold allocation strategy that differentiates hot data (frequently accessed) from cold data (less frequently accessed) and allocates SLC flash for the former and TLC flash for the latter in hybrid SSD[6][7], one which we call *Hot-Cold Page Allocation* (**HC-PA**) in this study. When all user writes are identified as hot data, HC-PA actually is identical to TW-PA.

When there is an abundance of available SLC pages within each plane, both the PF-PA strategy and TW-PA strategy are able to gain performance benefit from SLC writes and parallelism, while HC-PA may not be able to fully exploit SLC flash because cold data are directly written to TLC flash. On the other hand, when only part of planes provide available SLC

pages, PF-PA, which treats every plane equally, does not fully exploit SLC writes; whereas, TW-PA, which only uses planes with available SLC to serve user write transactions, does not fully exploit parallelism provided by the flash chip array; similarly to TW-PA, HC-PA, which allocates pages according to the characteristic of user requests (e.g., size), is also unlikely to fully exploits SLC writes and parallelism. Therefore, none of existing page allocation strategies, PF-PA, TW-PA, and HC-PA, is able to exploit both the structural heterogeneity and high internal parallelism inherent in hybrid SSDs.

*3) **Data Migration Policy**:* In an SLC-TLC hybrid SSD, SLC flash is used as write buffer for TLC flash by design and, as such, the storage capacity of SLC flash is much smaller than that of TLC flash. Therefore, data stored in SLC flash must be flushed to TLC flash in a timely fashion to reclaim SLC space for serving subsequent user writes, a process known as data migration. The critical data path of data migration usually occupies multiple flash resources, potentially contenting for the same resources required by the front-end write requests and thus significantly degrading user performance during data migration. As a result, data migration is usually performed when the device is idle[5][15] to minimize the negative impact on user writes, a policy referred to as *BackGround Data Migration* (**BG-DM**) in this study. Unfortunately, BG-DM strictly limits the opportunity to perform data migration, leading to poor utilization of SLC flash in hybrid SSD.

III. Insight

The flash chip array in SSD is constructed hierarchically with an abundance of parallelism at each level. We quantify the parallelism of flash chip array as the total number of dies, because each die can perform different flash operation independently[2]. To simplify our analysis without the loss of generality, we limit the number of planes within each die to 1, and the total number of dies equals the total number of planes.

To explore the best opportunity to perform data migration in SLC-TLC hybrid SSD without significantly impacting user write performance, we analyze and observe how different parallelism impacts the write performance in an SSD with 3D TLC flash. This analysis is based on simulation on HybridSim (detailed in Section V), which is a hybrid SSD simulator developed based on MQSim[12].

By constraining the number of channels, the number of chips per channel, and the number of dies per chip, we are able to vary the parallelism provided by the flash chip array in our simulation. For instance, the structural parameters listed in Table I (Section VI) indicate a flash chip array with parallelism of 128 (=8*4*4). When the number of dies per chip is limited to 2, then the parallelism provided is decreased to 64 (=8*4*2). Based on this approach of quantifying and varying parallelism, we compare the write performance of 3D TLC SSD with parallelism ranging from 16 to 128.

The results are illustrated in Fig. 3. In this experiment, we measure the write throughput as a function of parallelism

[2]Though plane is usually regarded as the lowest level of parallelism in flash chip array, the parallelism among planes is severely limiting[13].

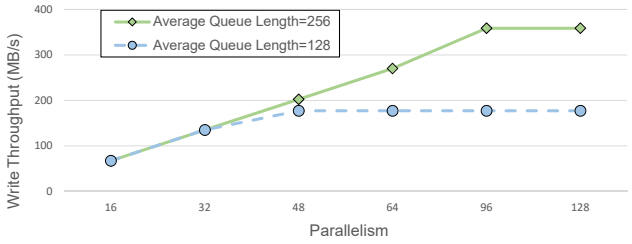


Fig. 3. Write throughput as a function of parallelism and average queue length.

when the average queue length of user requests is set at 128 and 256 respectively, where all user requests are 8KB writes. We first focus on the dotted line (blue) where the average queue length is 128. The line shows an upward trend when parallelism increases from 16 to 48, and then the throughput maintains unchanged when parallelism further increases from 48 to 128. This trend indicates that the maximum throughput can be obtained when parallelism is 48 (close to $\frac{128}{3}$). In other words, for a flash chip array with 128 dies, 48 dies are sufficient to provide maximum throughput when the average queue length is 128. The solid line (green, average queue length=256) in Fig. 3 shows a similar trend, except that the maximum throughput is achieved when parallelism is 96.

This observation reveals a critical insight into the relationship between parallelism and user write throughput of SLC-TLC hybrid SSD. That is, the parallelism provided by flash chip array is usually more than what is required to sustain the maximum user write throughput. In other words, there are plenty of opportunities to perform data migration (or garbage collection) in some dies without affecting user writes. This insight motivates us to improve the data migration policy in SLC-TLC hybrid SSD, detailed in Subsection IV-B.

IV. Design of SPA-SSD

In this section, we detail the design of the proposed SPA-SSD. There are generally three key considerations in the design of a hybrid SSD, namely, SLC flash distribution, page allocation and data migration. To maximize the parallelism of SLC write, SPA-SSD uses the ***plane-level distribution*** flash layout that achieves the best tradeoff between SLC parallelism and user write throughput as analyzed and observed in Subsection II-B1, in which SLC blocks are distributed uniformly across all planes. In the rest of this section we focus on SPA-SSD's page allocation strategy, ***T**ype-**P**arallelism **J**oint **P**age **A**llocation* (**TPJ-PA**) (Subsection IV-A), and data migration strategy, ***Q**ueue-length and **P**arallelism **C**onstrained **D**ata **M**igration* (**QPC-DM**) (Subsection IV-B). While TPJ-PA determines how user write requests are distributed within the flash chip array, QPC-DM determines how flash chip array is managed for data migration in the background.

A. Type-Parallelism Joint Page Allocation Strategy

To accommodate one-shot programming in 3D TLC (detailed in Subsection II-A), and to overcome drawbacks of the

PF-PA, TW-PA, and HC-PA strategies (detailed in Subsection II-B2), we propose the TPJ-PA strategy. In allocating channel/chip/die/plane IDs for user write requests, TPJ-PA takes three important factors into consideration, i.e., "***incomplete TLC page set***", "***available SLC pages***", and "***outstanding write transactions***".

Here is how TPJ-PA accommodates one-shot programming and overcomes the problems of the existing page allocation strategies. When a plane has incomplete TLC page set (i.e., not enough data to form a full page set to trigger one-shot programming immediately), it may trigger one-shot programming after *program delay* if no further TLC writes are allocated to this plane. Since triggering TLC one-shot program operation without gathering a full page set is undesirable because of low resource utilization, subsequent requests should be allocated this plane to gather a full page set as soon as possible. Accordingly, TPJ-PA prioritizes TLC pages from the plane with ***incomplete TLC page set*** for allocation.

When there is no plane with incomplete TLC page set, planes with available SLC pages are chosen to buffer user write requests. However, choosing planes with available SLC pages blindly will lead to the same problem caused by the TW-PA strategy, in which write requests be concentrated in a very small number of planes (where SLC pages are available) while other planes (where only TLC pages are available) are idle, leading to poor resource utilization. Therefore, TPJ-PA searches for not only planes with available SLC pages but also those with no available SLC pages. Specifically, TPJ-PA looks for a plane P_S with not only ***available SLC pages*** but also ***the fewest outstanding write transaction***. Simultaneously, TPJ-PA also looks for a plane P_T ***without available SLC pages*** but with ***the fewest outstanding write transaction***. Once found, the two candidate planes, P_S and P_T, are compared to further determine which plane is actually used to serve the request. The comparison is based on the respective expected write latency, which is roughly calculated according to the number of outstanding write transactions. For instance, the number of outstanding write transactions in P_S is N_{P_S} while the number of outstanding write transactions in P_T is N_{P_T}, then plane P_T is chosen when $N_{P_S} * T_{SLC} > \frac{(3+N_{P_T})}{3} * T_{TLC}$, otherwise, plane P_S is chosen. The T_{SLC} is SLC programming latency while T_{TLC} is TLC one-shot programming latency. The constant '3' in the inequality reflects the fact that forming a TLC page set needs three transactions. This criterion for the comparison is designed to limit the requests served by SLC pages when there are only very few planes with available SLC pages, while also allowing requests to be distributed to other planes (without available SLC pages) to increase the resource utilization. Algorithm 1 shows pseudo-code of TPJ-PA.

In summary, by prioritizing page allocation for minimizing the wait time to fully fill TLC page sets, TPJ-PA accommodates and speeds up one-shot programming of 3D TLC flash; by distributing user writes to not only planes with available SLC pages but also those without available SLC pages, TPJ-PA improves resource utilization over the TW-PA and HC-PA strategies; by allocating more user writes to planes with available SLC pages, TPJ-PA improves utilization of SLC pages over the PF-PA and HC-PA strategies.

Algorithm 1: TPJ-PA Strategy

```
Input: Status of the flash chip array
Output: Candidate_Plane
if There is a plane with incomplete TLC page set then
    Select the plane with incomplete TLC page set;
    Candidate_Plane = the selected plane;
else
    if There is a plane with available SLC pages then
        Select a plane, P_S, with available SLC pages
          and the fewest outstanding transactions;
        Candidate_Plane = P_S;
        if There is a plane with no available SLC
          pages then
            Select a plane, P_T, with no available SLC
              pages and the fewest outstanding
              transactions;
            if N_{P_S} * T_SLC > (3+N_{P_T})/3 * T_TLC then
                /*Using P_T to serve the user write has
                  shorter write latency than using P_S*/
                Candidate_Plane = P_T;
    else
        Select a plane, P_T, with no available SLC
          pages and the fewest outstanding
          transactions;
        Candidate_Plane = P_T;
return Candidate_Plane;
```

B. Queue-length and Parallelism Constrained Data Migration

In 3D SLC-TLC hybrid SSD, when SLC pages are exhausted, subsequent write requests can only be served by TLC pages, which can severely degrade the user write performance because of the long and unpredictable one-shot programming write latency. Data migration that flushes write data buffered in SLC flash to TLC flash can free up unavailable SLC pages for serving subsequent write requests. Unfortunately, performing data migration improperly can also degrade user performance by interfering with user write requests. As a result, prior studies commonly adopt the background migration scheme that triggers data migration only when the device is idle. This scheme, however, can severely limit the utilization and **hit ratio** of SLC pages since their buffered write data may not be flushed to TLC pages for a long period of time, particularly when the front-end user traffic is heavy.

To improve the utilization of SLC flash in SPA-SSD, we present QPC-DM that reclaims SLC flash along with serving user requests while minimizing the negative impact on user requests. The key idea of designing QPC-DM is providing just but not all resources to serve user writes by comparing the density of user workload and parallelism provided by the flash

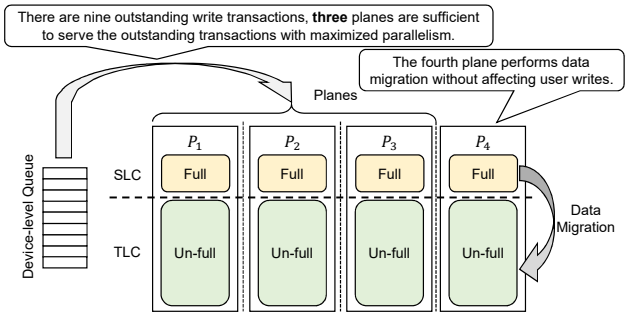


Fig. 4. An example illustrating the QPC-DM strategy.

chip array, in which the density of user workload is quantified by the number of outstanding write transactions while the parallelism is quantified by the number of dies provided by the flash chip array.

*1) **When to trigger data migration?**:* Specifically, considering a situation where there are l outstanding write transactions and the device ran out of SLC flash, meaning that these outstanding writes should be served by TLC pages. As each die can serve at least three write transactions at once with one-shot programming scheme, $l/3$ dies are enough for serving these transactions with maximum parallelism - providing more than $l/3$ dies will not improve the performance, as observed in Section III. Moreover, assume that there are k planes within each die and the advanced command "multi-plane write" is available, then each die is able to serve up to $3k$ write transactions simultaneously, further decreasing the number of required dies to $l/(3k)$ for providing maximum parallelism. To simplify the discussion without the loss of generality, we set $k = 1$, which means that each die contains only one plane. As such, planes perform flash operations independently, and $l/3$ planes are sufficient to provide maximum parallelism for user write requests.

Let the total number of planes be p, then data migration can be triggered without degrading user write performance as long as $l/3$ is less than p, meaning that there is at least one surplus plane (i.e., surplus parallelism) available for data migration. For instance, in Fig. 4, $p = 4$, $l = 9$, then 3 planes are used for serving user writes while performing data migration on the other 1 plane. As the planes performing data migration are independent from those serving user writes, user write performance are unlikely to be degraded significantly - performing data migration occupies processor and mapping table, which may delay user writes slightly. Considering the variance on workload intensity, some planes other than the $l/3$ planes should also be preserved to absorb write bursts. Finally, $P_{dm} = \max(p - l/3 - \sigma, 0)$ planes are allowed to perform data migration during the runtime, in which σ is the number of planes preserved to absorb write bursts.

*2) **When to terminate data migration?**:* Reclaiming all SLC pages of a plane at one time through data migration takes tens of seconds. This time becomes even longer if garbage collection on TLC flash is also triggered. As data migration is usually triggered when the device runs out of free SLC pages, long data migration time means that user requests can only be served by TLC flash for a long period of time, resulting in significantly degraded write performance. Thus, QPC-DM must find the best time to not only trigger data migration but also terminate data migration on planes to enable SLC writes.

The decision to terminate data migration is made when an SLC block within a plane is reclaimed. Specifically, when an SLC block is reclaimed, QPC-DM first checks the number of outstanding write transactions. If the preserved planes are not sufficient to provide the maximum parallelism for the outstanding write transactions, the data migration should be terminated; else, if the number of SLC blocks reclaimed in the plane exceeds a predefined threshold, data migration should also be terminated to free the plane to serve user write transactions; otherwise, the plane should continue data migration to reclaim more SLC blocks.

V. HybridSim, Simulator for Hybrid SSD

To evaluate the performance of SPA-SSD, we developed HybridSim, a simulator for hybrid SSD based on MQSim. In developing HybridSim, we added the following major features:

1) **One-shot programming** scheme for 3D TLC flash.
2) SLC-TLC hybrid flash chip array with the ***plane-level distribution*** flash layout.
3) The **PF-PA**, **TW-PA**, and **HC-PA** are introduced as baseline page allocation strategies.
4) **BG-DM** is introduced as a baseline data migration policy.
5) Implementing **SPA-SSD** by introducing the **TPJ-PA** strategy and **QPC-DM** policy.

To the best of our knowledge, HybridSim is the first 3D SLC-TLC hybrid SSD simulator in the field. It can be used to facilitate new design ideas for 3D SLC-TLC hybrid SSD.

TABLE I
Structural Parameters

Parameter	Value
Flash layout	*plane-level distribution*
Channel num	8
Chip num per Channel	4
Die num per Chip	4
SLC/TLC Block num per Die	92/676
Page num per SLC/TLC block	128/384
Page size	8KB
Total Capacity	265GB
SLC/TLC program latency	500us/5500us
Program delay of one-shot programming	100ms

VI. Experimental Evaluation of SPA-SSD

We evaluate the performance of SPA-SSD based on HybridSim. The structural parameters of the simulated hybrid SSD are listed in Table I. As described in Section IV, SPA-SSD is composed of the two major techniques of TPJ-PA and QPC-DM, a novel page allocation strategy and a novel data migration policy. We first evaluate these two techniques with synthetic workloads (continuous 8KB writes) separately in Subsection VI-A and Subsection VI-B, and then evaluate SPA-SSD as a whole with real workloads in Subsection VI-C.

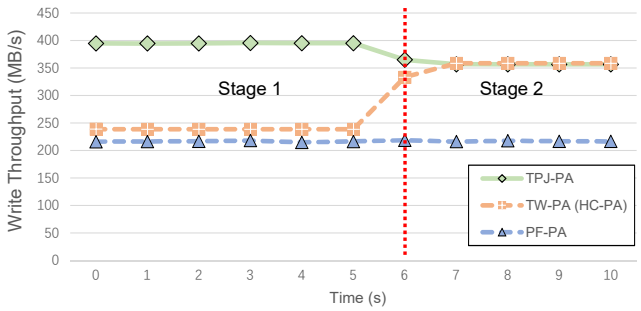


Fig. 5. Evaluation of page allocation strategies.

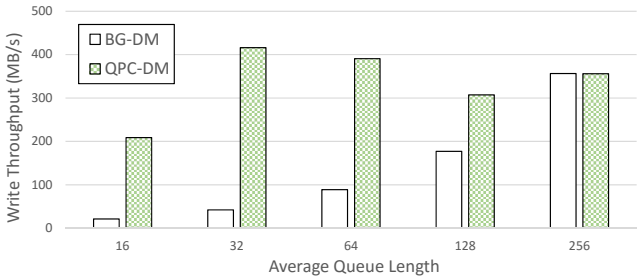


Fig. 6. Evaluation of data migration policies.

A. Benefit of TPJ-PA

We evaluate the write performance of TPJ-PA by comparing it with baseline page allocation strategies PF-PA, TW-PA, and HC-PA. For PF-PA, the CWDP static page allocation scheme is used for distributing transactions to planes equally[13][14]. For HC-PA, the size of a user write request is used for identifying it as hot or cold[7]. To show the difference among these three page allocation strategies clearly, we limit the number of planes providing SLC flash to 16. In other words, only one in eight planes provides SLC flash while other planes provide TLC flash only. BG-DM that triggers data migration only when the device is idle is employed in the hybrid SSD. However, as user requests are sent to the device non-stop in this experiment, the device will not be idle for a long enough period of time, thus, data migration is actually never triggered.

The results are presented in Fig. 5, where the average queue length is set to 256. In the figure, the write throughput along the time axis can be divided to two stages. For the first stage, there are available SLC pages in the flash chip array, while for the second stage, all SLC pages are exhausted. The curves for TW-PA and HC-PA are identical because all user requests (8KB writes) are identified as hot data.

In Fig. 5, we observe that PF-PA exhibits the lowest performance. The reason is twofold. First, PF-PA distributes requests to all planes, thus its utilization of SLC flash is lower than the other two strategies; second, distributing requests uniformly to every plane delays the gathering of TLC page sets. The throughput of TW-PA(HC-PA) in the first stage is significantly lower than that of TPJ-PA. More interestingly, the throughput of TW-PA(HC-PA) in the second stage, where only TLC flash is available for serving user writes, outperforms that in the first stage. The reason behind these results is that TW-PA(HC-PA) only uses the planes providing SLC flash to serve user writes in the first stage, which means that 7/8 planes are not being utilized, greatly wasting the parallelism provided by the flash chip array. On the other hand, in the second stage, though only TLC flash is available, the higher parallelism exploited helps gain high throughput. Different from TW-PA(HC-PA) for which the second stage has higher performance than the first stage, in TPJ-PA, the first stage has higher throughput than its second stage, because TPJ-PA uses not only SLC flash but also TLC flash to serve user requests in the first stage.

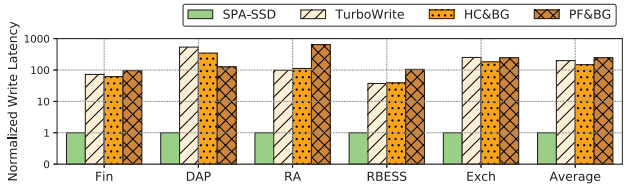
B. Benefit of QPC-DM

In this experiment, we compare hybrid SSD devices with the BG-DM and QPC-DM data migration policies. The simulated devices adopt the TPJ-PA page allocation strategy and we measure the write performance after SLC pages are exhausted so as to gauge the impact of data migration on user performance. In Fig. 6 presents the average write throughput, clearly showing that the throughput with QPC-DM is much higher than that with BG-DM when the average queue length is 128 or shorter. This is because in these cases QPC-DM can perform data migration to reclaim SLC pages without affecting user TLC writes, where the newly freed SLC pages can in turn boost the performance of subsequent write requests. On the contrary, BF-DM does not trigger data migration because the device receives write requests constantly and is never idle during the simulation. When the average queue length is 256, fewer planes can be preserved to perform data migration in QPC-DM, making its throughput close to BG-DM's.

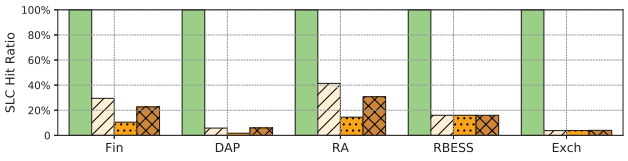
C. Performance on Real Workloads

In this experiment, we evaluate the performance of SPA-SSD on real workloads, compared with devices using TW-PA, HC-PA, or PF-PA as page allocation strategy and BG-DM as data migration policy, denoted as TW&BG, HC&BG, and PF&BG, respectively. It is worth noting that TW&BG is the equivalent of Samsung's TurboWrite technology[15], hence the labeling of TurboWrite for TW&BG in the following.

Fig. 7(a) shows a comparison of write latency (the delay to persistently store user data in flash chips) under real workloads, normalized to that of SPA-SSD. SPA-SSD is shown to have the lowest write latency under all tested workloads. On average, the write latency of SPA-SSD is only 1/200 of TurboWrite, 1/147 of HC&BG, and 1/244 of PF&BG. The short write latency of SPA-SSD comes from its high SLC hit ratio, defined to be the percentage of user writes served by SLC flash, or equivalently the write buffer. As illustrated by Fig. 7(b), over 99% user writes are served by SLC in SPA-SSD, while much smaller percentages of user writes are served by SLC in the other three devices (as low as 3.8%). As the write intensity in the tested workloads is relatively low, the wait time to filling in TLC page sets usually is long (up to 100ms, i.e., the *program delay* of one-shot programming). As a result, TurboWrite, HC&BG, and PF&BG that serve user



(a) Write latency under real workloads, normalized to that of SPA-SSD.



(b) SLC hit ratio.

Fig. 7. Evaluation of page allocation strategies.

writes mainly using TLC flash exhibit extremely long write latency. Lowering the *program delay* can reduce their write latency at the cost of which more TLC program operations are triggered even if page sets are not filled up, which leads to undesirable wastes in both storage space and P/E cycles.

VII. Related Works

Many prior studies have discussed the design of hybrid SSD that contains two types of flash chips for the purpose of improving lifetime and performance. S. Lee et al.[5] proposed FlexFS that divides storage space of SSD to an SLC zone and an MLC zone, where user writes are forwarded to SLC zone, and then data in the SLC zone are migrated to TLC zone. To mitigate the negative impact of data migration in FlexFS, they developed a background migration technique that triggers data migration during idle time, a dynamic allocation technique that forwards user writes to MLC directly, and a locality-aware data management technique that retains hot data in the SLC zone and migrates cold data to the TLC zone. L. Chang[6] also considered the design of SLC-MLC hybrid SSD, and proposed the idea of writing hot data to the SLC zone while writing cold data directly to the TLC zone, which reduces the amount of data migrated from the SLC zone to the TLC zone. Similarly, S. Im and D. Shin[7] proposed ComboFTL that also focus on the management of hot/cold data in SLC-MLC hybrid SSD, where data can not only be moved from the SLC zone to the MLC zone but also the other way around. Specifically, cold data in the SLC zone are migrated to the MLC zone, while hot data are migrated in the opposite direction. Cognizant of the fact that SLC flash chips are worn out faster than MLC/TLC flash chips in hybrid SSD because the former absorb the vast majority of writes, M. Murugan and D. Du proposed Hybrot that forwards even hot data to MLC flash directly to balance cell wear between SLC and MLC[8]. Samsung utilizes TurboWrite in their hybrid SSDs, where user write data are first buffered in the SLC zone, and are then flushed to the TLC zone when the device is idle[15]. In [9], W. Wang et al. discussed the impact of the SLC zone capacity to write performance, and demonstrated that the SLC zone capacity should be adjusted according to the workload to optimize the write performance. C. Chang et al.[16] considered the design of SLC-MLC hybrid SSD for the purpose of meeting service-level objective, and proposed the idea of allocating pages for user writes according to the expected write latency. Specifically, MLC pages are allocated for the user writes as long as the write latency of the user writes will not exceed deadline; otherwise, SLC pages are allocated for the user writes.

Although all these existing studies help advance the research on hybrid SSD, they have not adequately investigated flash layout and page allocation strategies of hybrid SSD, particularly 3D SLC-TLC hybrid SSD, leaving significant room for advancement in hybrid SSD research. Differently, in this study, we analyze the differences among various flash layout strategies and categorize page allocation strategies. Our study is based on 3D SLC-TLC flash SSD that utilizes one-shot programming, which has not been studied in the field to the best of our knowledge but is commonly used in modern SSDs. We also propose a novel page allocation strategy and a novel data migration policy in this study that are shown to be highly effective. We also develop a hybrid SSD simulator, HybridSim, which can be used to facilitate more new design ideas.

VIII. Conclusion

We analyzed the flash layout and page allocation strategies, and data migration policy of 3D SLC-TLC hybrid SSD, and proposed a novel hybrid SSD design, called SPA-SSD. SPA-SSD employs the *plane-level distribution* flash layout scheme to maximize the write parallelism to SLC flash and TLC flash. A novel page allocation strategy, TPJ-PA, is introduced in SPA-SSD, to leverage the advantages of the state-of-the-art page allocation strategies while overcome their drawbacks to fully exploit heterogeneity and parallelism inside 3D SLC-TLC hybrid SSD. An improved data migration policy, QPC-DM, that performs data migration without affecting user write performance is also proposed in SPA-SSD to significantly increase the hit ratio of SLC pages for user write requests. Furthermore, we developed HybridSim, a simulator for 3D SLC-TLC hybrid SSD, to evaluate SPA-SSD. Experiments show that TPJ-PA and QPC-DM are able to improve write throughput by 60% and up to 10 times, respectively. Simulation results based on real workloads further reveal that SPA-SSD improves write latency over the state-of-the-art hybrid SSD designs by a factor of 147 to 244.

Acknowledgment

The authors would like to thank the anonymous referees for their valuable comments and helpful suggestions. This work is supported in part by NSFC No. 61872156, Creative Research Group Project of NSFC No. 61821003, National key research and development program of China (Grant No. 2018YFA0701804), the Fundamental Research Funds for the Central Universities No. 2018KFYXKJC037, Alibaba Group through Alibaba Innovative Research (AIR) Program, and the US NSF under Grant No. CCF-1704504 and CCF-1629625.

REFERENCES

[1] D. James and J. Choe, "TechInsights Memory Technology Update from IEDM18," https://www.techinsights.com/blog/techinsights-memory-technology-update-iedm18, 2019.

[2] L. M. Grupp, J. D. Davis, and S. Swanson, "The Harey Tortoise: Managing Heterogeneous Write Performance in SSDs," in *the 2013 USENIX Annual Technical Conference (ATC' 13)*, San Jose, CA, USA, June 2013, pp. 79–90.

[3] J. Im, W. Jeong, D. Kim, S. Nam, D. Shim, M. Choi, H. Yoon, D. Kim, Y. Kim, H. Park, D. Kwak, S. Park, S. Yoon, W. Hahn, J. Ryu, S. Shim, K. Kang, S. Choi, J. Ihm, Y. Min, I. Kim, D. Lee, J. Cho, O. Kwon, J. Lee, M. Kim, S. Joo, J. Jang, S. Hwang, D. Byeon, H. Yang, K. Park, K. Kyung, and J. Choi, "A 128Gb 3b/cell V-NAND Flash Memory with 1Gb/s I/O Rate," in *IEEE International Solid-State Circuits Conference (ISSCC'15)*, Feb 2015, pp. 1–3.

[4] F. Wu, Z. Lu, Y. Zhou, X. He, Z. Tan, and C. Xie, "OSPADA: One-Shot Programming Aware Data Allocation Policy to Improve 3D NAND Flash Read Performance," in *IEEE 36th International Conference on Computer Design (ICCD'18)*, Oct 2018, pp. 51–58.

[5] S. Lee, K. Ha, K. Zhang, J. Kim, and J. Kim, "FlexFS: A Flexible Flash File System for MLC NAND Flash Memory," in *the 2009 USENIX Annual Technical Conference (ATC' 09)*, San Diego, CA, USA, June 2009, pp. 9–9.

[6] L. Chang, "A Hybrid Approach to NAND-Flash-Based Solid-State Disks," *IEEE Transactions on Computers*, vol. 59, no. 10, pp. 1337–1349, October 2010.

[7] S. Im and D. Shin, "ComboFTL: Improving Performance and Lifespan of MLC Flash Memory Using SLC Flash Buffer," *Journal of Systems Architecture*, vol. 56, no. 12, pp. 641–653, 2010.

[8] M. Murugan and D. H. C. Du, "Hybrot: Towards Improved Performance in Hybrid SLC-MLC Devices," in *the IEEE 20th International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS' 12)*, Washington, DC, USA, August 2012, pp. 481–484.

[9] W. Wang, W. Pan, T. Xie, and D. Zhou, "How Many MLCs Should Impersonate SLCs to Optimize SSD Performance?" in *the 2nd International Symposium on Memory Systems (MEMSYS' 16)*, Alexandria, VA, USA, October 2016, pp. 238–247.

[10] D. Sharma, "System Design for mainstream TLC SSD," in *the Flash Memory Summit*, Santa Clara, CA, USA, August 2014, pp. 1–20.

[11] A. I. Alsalibi, S. Mittal, M. A. Al-Betar, and P. B. Sumari, "A Survey of Techniques for Architecting SLC/MLC/TLC Hybrid Flash Memory–based SSDs," *Concurrency and Computation: Practice and Experience*, vol. 30, no. 13, p. e4420, 2018, e4420 cpe.4420.

[12] A. Tavakkol, J. Gómez-Luna, M. Sadrosadati, S. Ghose, and O. Mutlu, "MQSim: A Framework for Enabling Realistic Studies of Modern Multi-Queue SSD Devices," in *the 16th USENIX Conference on File and Storage Technologies (FAST' 18)*, Oakland, CA, USA, February 2018, pp. 49–66.

[13] Y. Hu, H. Jiang, D. Feng, L. Tian, H. Luo, and S. Zhang, "Performance Impact and Interplay of SSD Parallelism Through Advanced Commands, Allocation Strategy and Data Granularity," in *the 25th International Conference on Supercomputing (ICS' 11)*, Tucson, AZ, USA, June 2011, pp. 96–107.

[14] A. Tavakkol, P. Mehrvarzy, M. Arjomand, and H. Sarbazi-Azad, "Performance Evaluation of Dynamic Page Allocation Strategies in SSDs," *ACM Transactions on Modeling and Performance Evaluation of Computing Systems*, vol. 1, no. 2, pp. 1–33, Jun 2016.

[15] S. E. Co., "Samsung Solid State Drive TurboWrite Technology White Paper," https://images-eu.ssl-images-amazon.com/images/I/914ckzwNMpS.pdf, 2013.

[16] C.-W. Chang, G.-Y. Chen, Y.-J. Chen, C.-W. Yeh, P. Y. Eng, A. Cheung, and C.-L. Yang, "Exploiting Write Heterogeneity of Morphable MLC/SLC SSDs in Datacenters with Service-Level Objectives," *IEEE Transactions on Computers*, vol. 66, no. 8, pp. 1457–1463, August 2017.