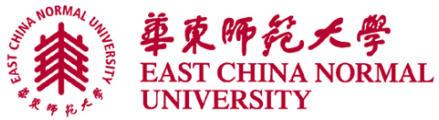



# Parallel all the time

## *PLANE LEVEL PARALLELISM EXPLORATION FOR HIGH PERFORMANCE SOLID STATE DRIVES*

Congming Gao , Liang Shi, Jason Chun Xue, Cheng Ji, Jun Yang, Youtao Zhang
*Chongqing University; East China Normal University; City University of Hong Kong; University of Pittsburgh*

# Outline

***Background***

Problem Statement

SPD: From Plane to Die Parallelism Exploration

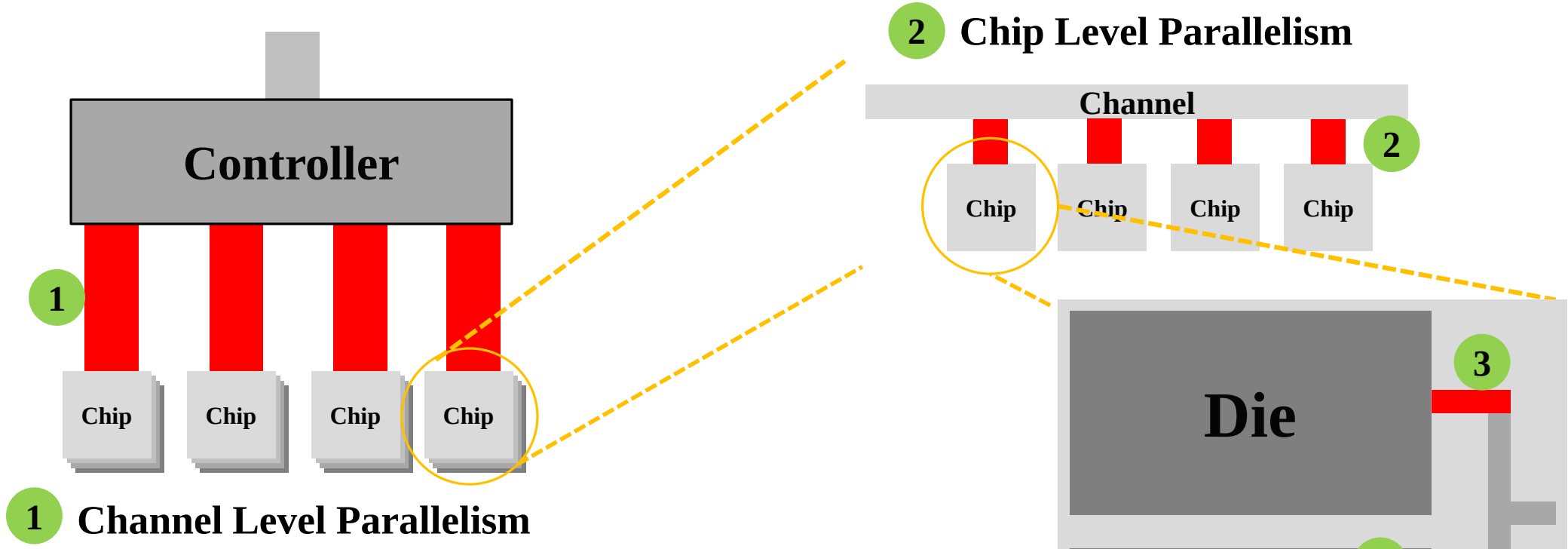
- Overview
- Die Level Write Construction
- Die Level GC

Experiment Setup

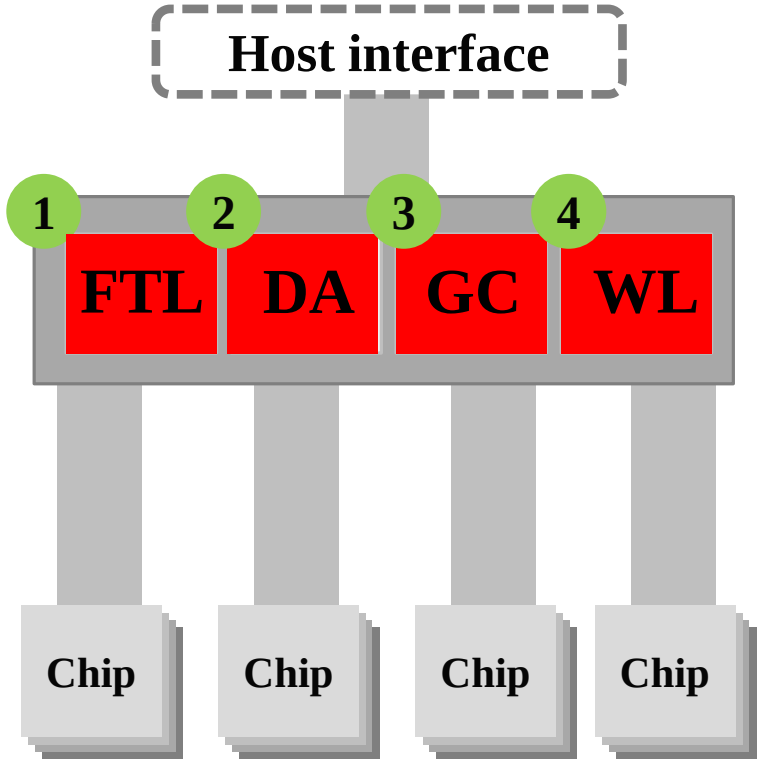
Results

Conclusion

# Parallel Organization



# Controller Design




1 **Logical Address**

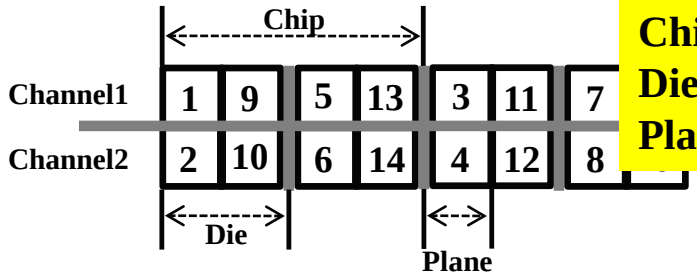
↓

**Flash Translation Layer**

↓

**Physical Address**

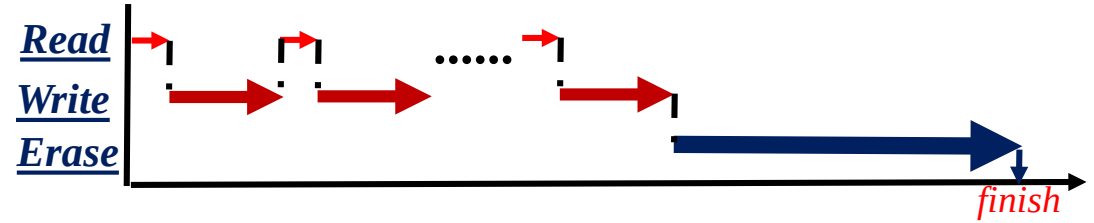
2 **Data Allocation**




**Channel First**
**Chip Second**
**Die Third**
**Plane Last**

[**Jung *et al. USENIX HotStorage'12***]

3

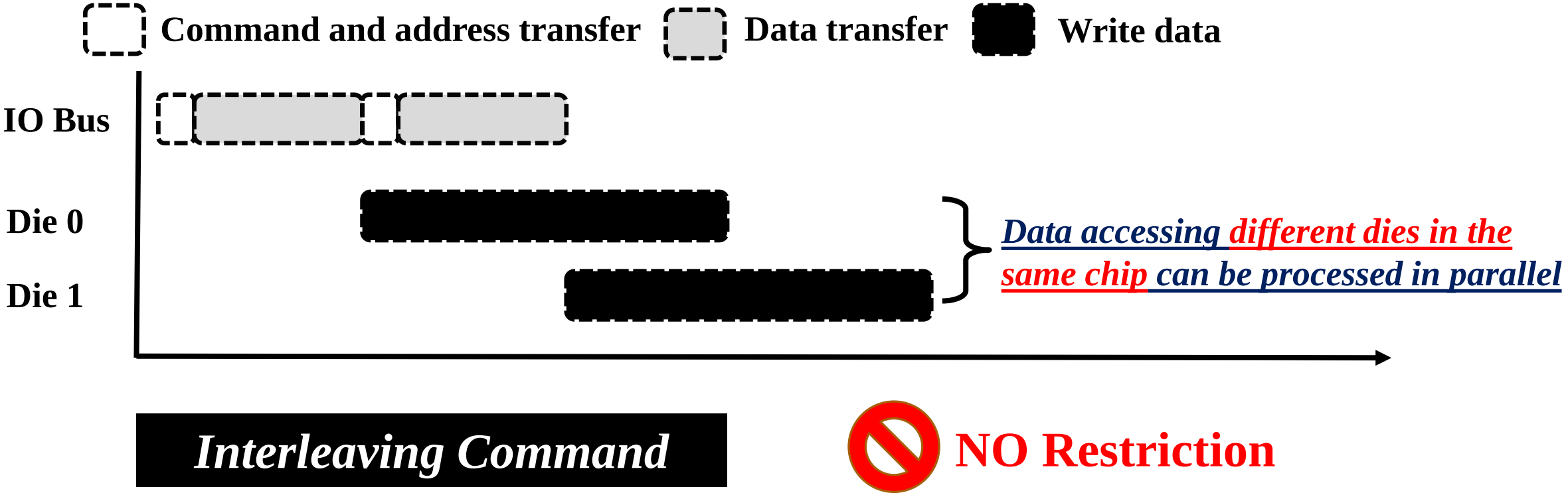



***GC is time consuming***

4 **Wear leveling prolongs the flash lifespan**

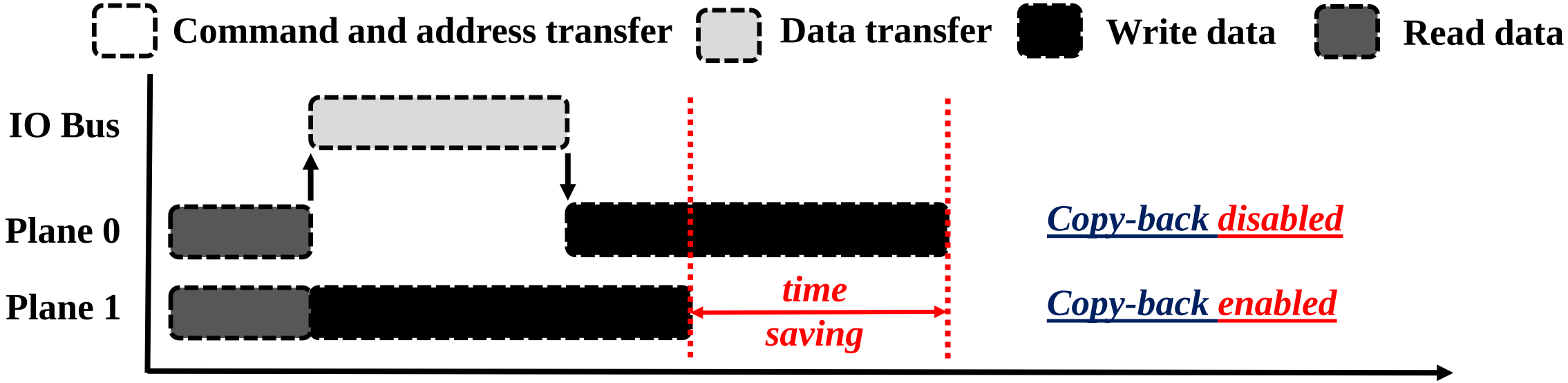
# Advanced Commands

Advanced commands, including **interleaving command, copy-back command and multi-plane command**, are used to exploit internal parallelism of SSDs.



# Advanced Commands

Advanced commands, including **interleaving command, copy-back command and multi-plane command**, are used to exploit internal parallelism of SSDs.

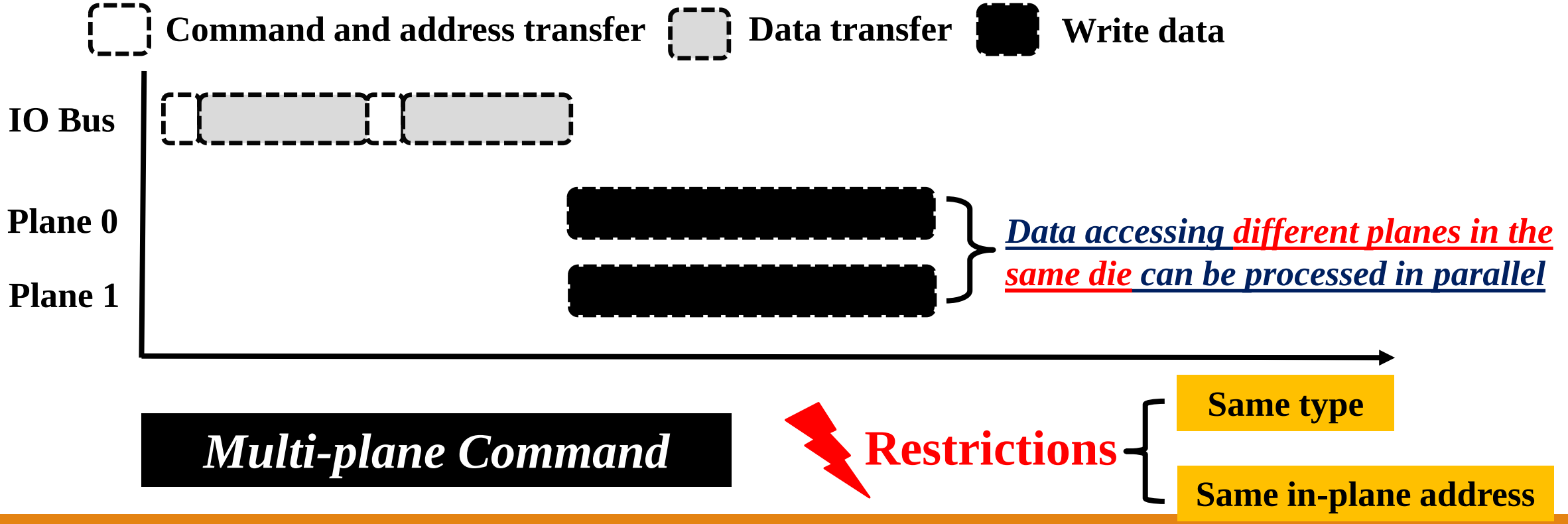



***Copy-back Command***

**NO Restriction**

# Advanced Commands

Advanced commands, including **interleaving command, copy-back command and multi-plane command**, are used to exploit internal parallelism of SSDs.



# Outline

Background

***Problem Statement***

SPD: From Plane to Die Parallelism Exploration

- Overview
- Die Level Write Construction
- Die Level GC

Experiment Setup

Results

Conclusion

# Problem Statement

Due to the restrictions of multi-plane command, plane level parallelism is hard to exploit.

***Based on the restrictions of multi-plane command, operations that access the same die can be categorized into one of the following four cases:***

***Case 1:*** **Operations are issued to *one plane only* (Single Write );**

***Case 2:*** ***It can be degraded to Case 1***

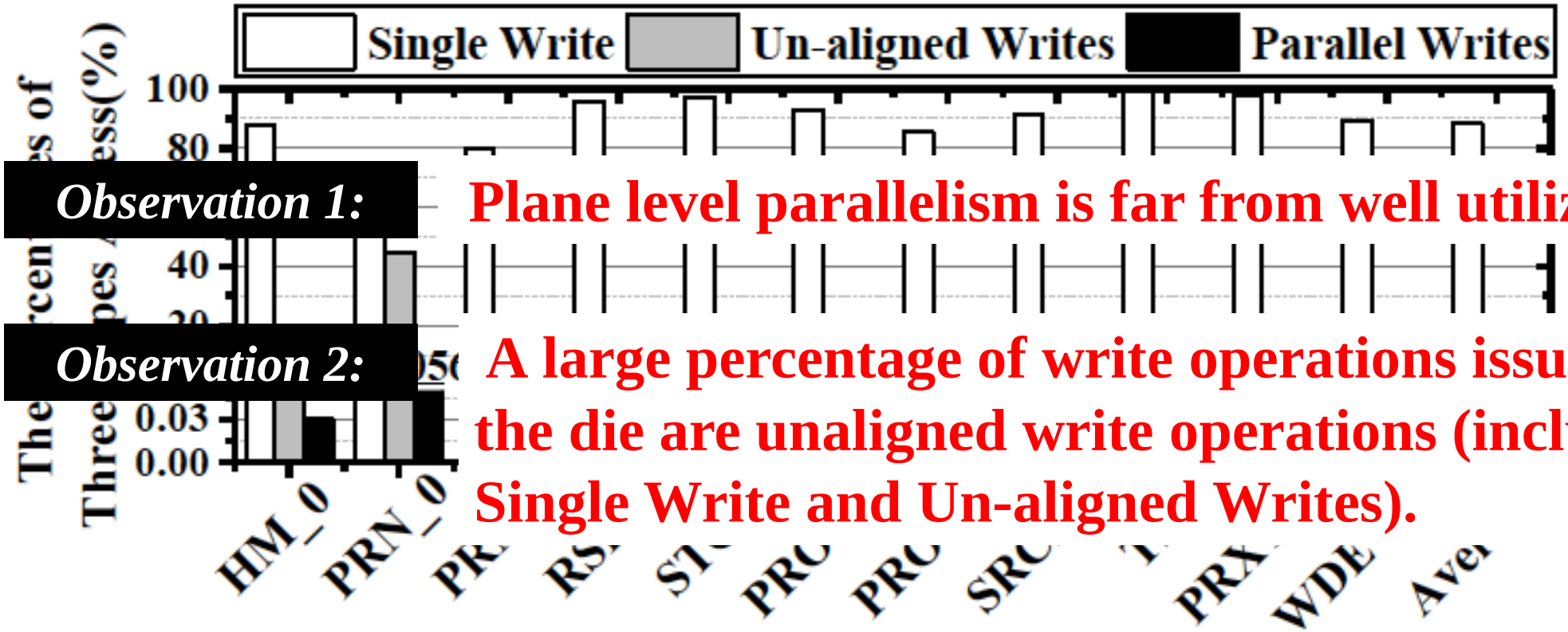
***Case 3:*** **Two same type operations with *unaligned in-plane addresses* are issued to the two planes of the die (Unaligned Writes );**

***Case 4:*** ***Two same type operations* with *aligned in-plane addresses* are issued to the two planes (Parallel Writes ).**

**Case 1, 2 & 3 result in the poor plane level parallelism of SSDs.**

# Problem Statement

The percentages of three cases are collected and presented:

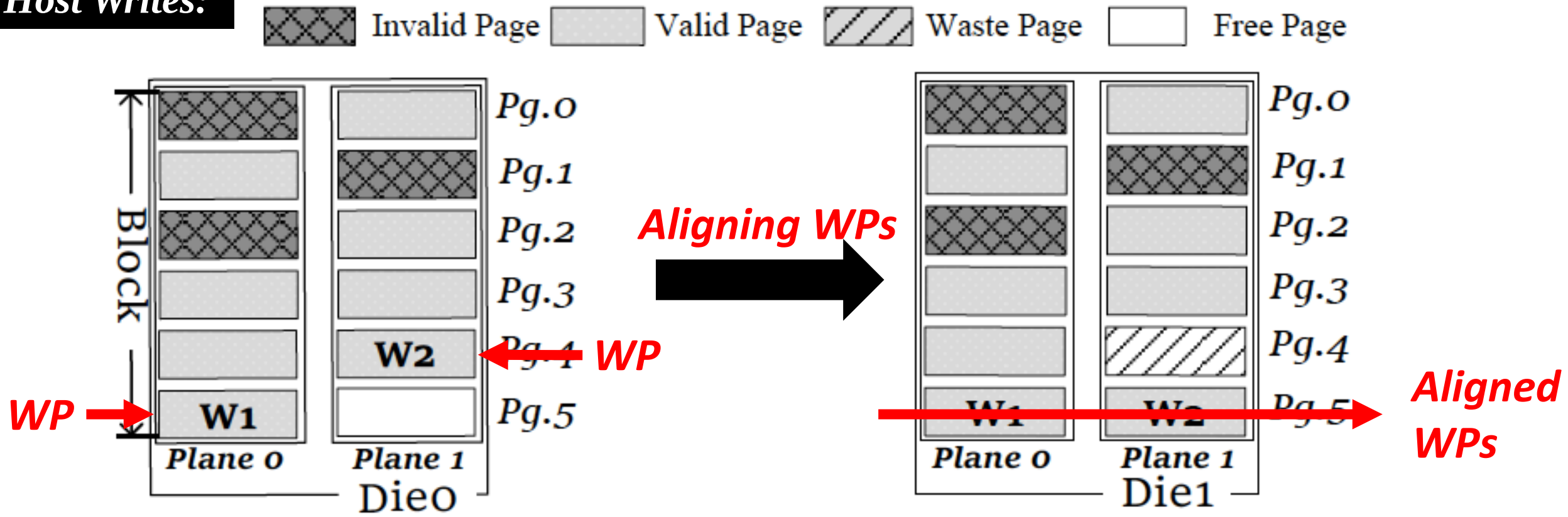



***Observation 1:*** **Plane level parallelism is far from well utilized;**

***Observation 2:*** **A large percentage of write operations issued to the die are unaligned write operations (including Single Write and Un-aligned Writes).**

# Problem Statement

***Host Writes:***




**Un-aligned write points**

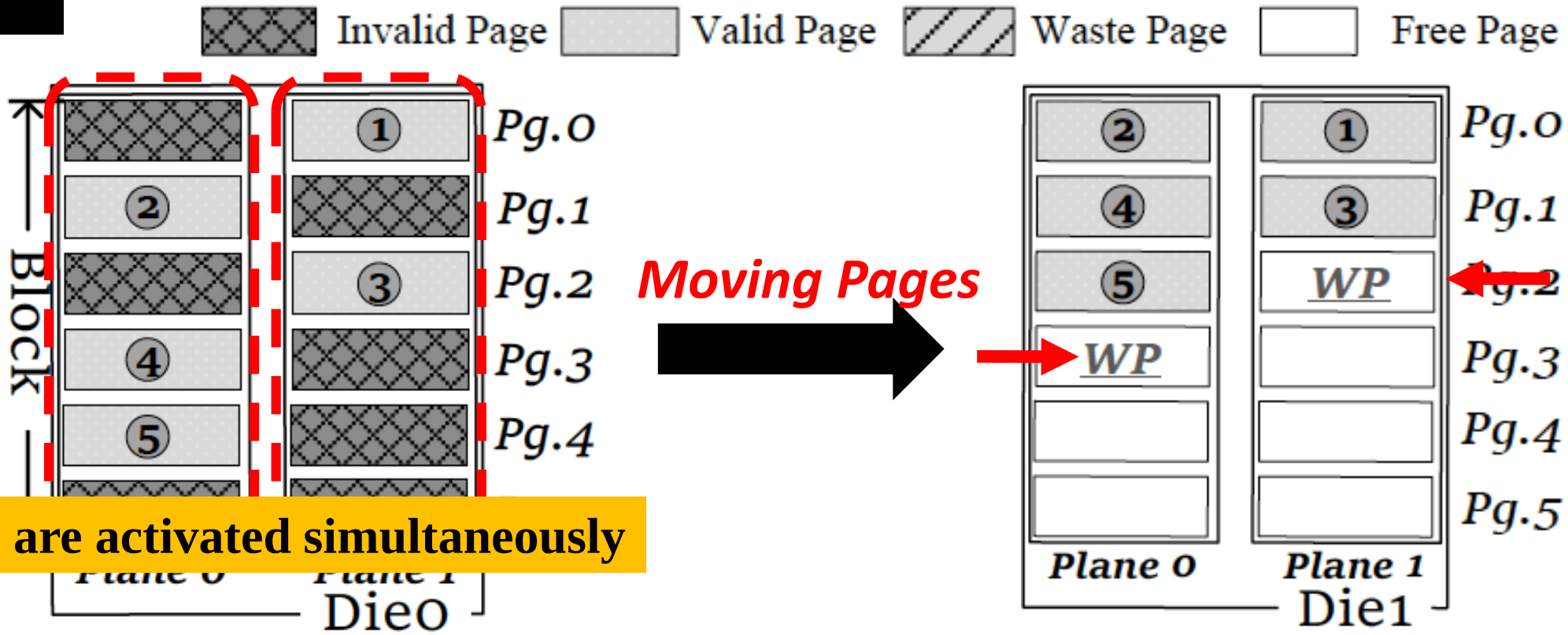
**W1 and W2 are processed in parallel**

**W1 and W2 are processed sequentially**

**But space is wasted.**

# Problem Statement

***GC:***




**GCs are activated simultaneously**

**Valid pages are moved sequentially due to un-aligned in-plane addresses.**

**Write points in new blocks still are un-aligned**

# Problem Statement

**For host writes and GCs,**

***how to align write points in each die***

**so that multi-plane command can be used to exploit plane level parallelism**

# Outline

Background

Problem Statement

***SPD: From Plane to Die Parallelism Exploration***

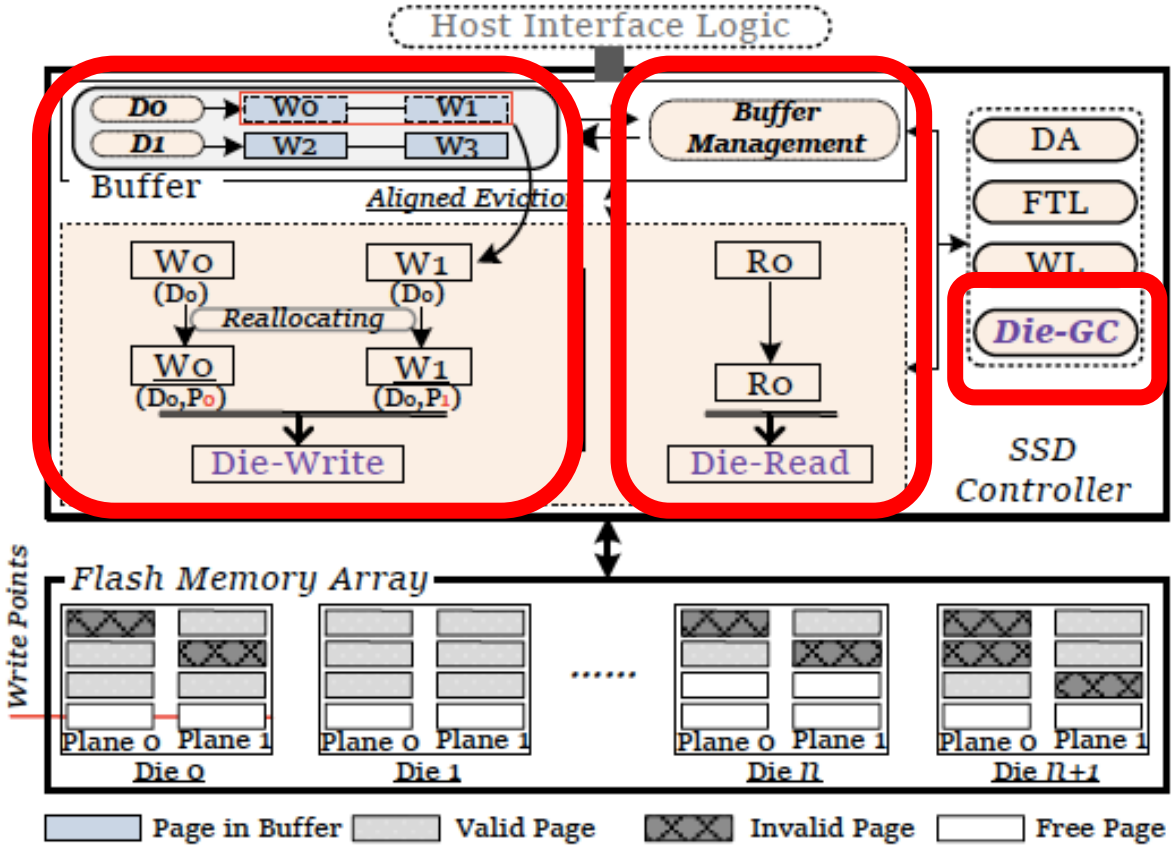
- Overview
- Die Level Write Construction
- Die Level GC

Experiment Setup

Results

Conclusion

# Overview




***SPD, an SSD from plane to die framework***

***ion scheme*** to align the

***Assuming there are 2 planes in a die:***

- ***Die-Write:*** **evicting 2 dirty pages at each time;**
- ***Die-Read:*** **reading 2 pages if possible;**
- ***Die-GC:*** **reclaiming victim blocks in 2 planes simultaneously.**

# Die Level Write Construction

***Two Goals:***

**1. The *amount of data* issued to a die should be a multiple of N pages**

(assuming there are N planes in a die)**;**

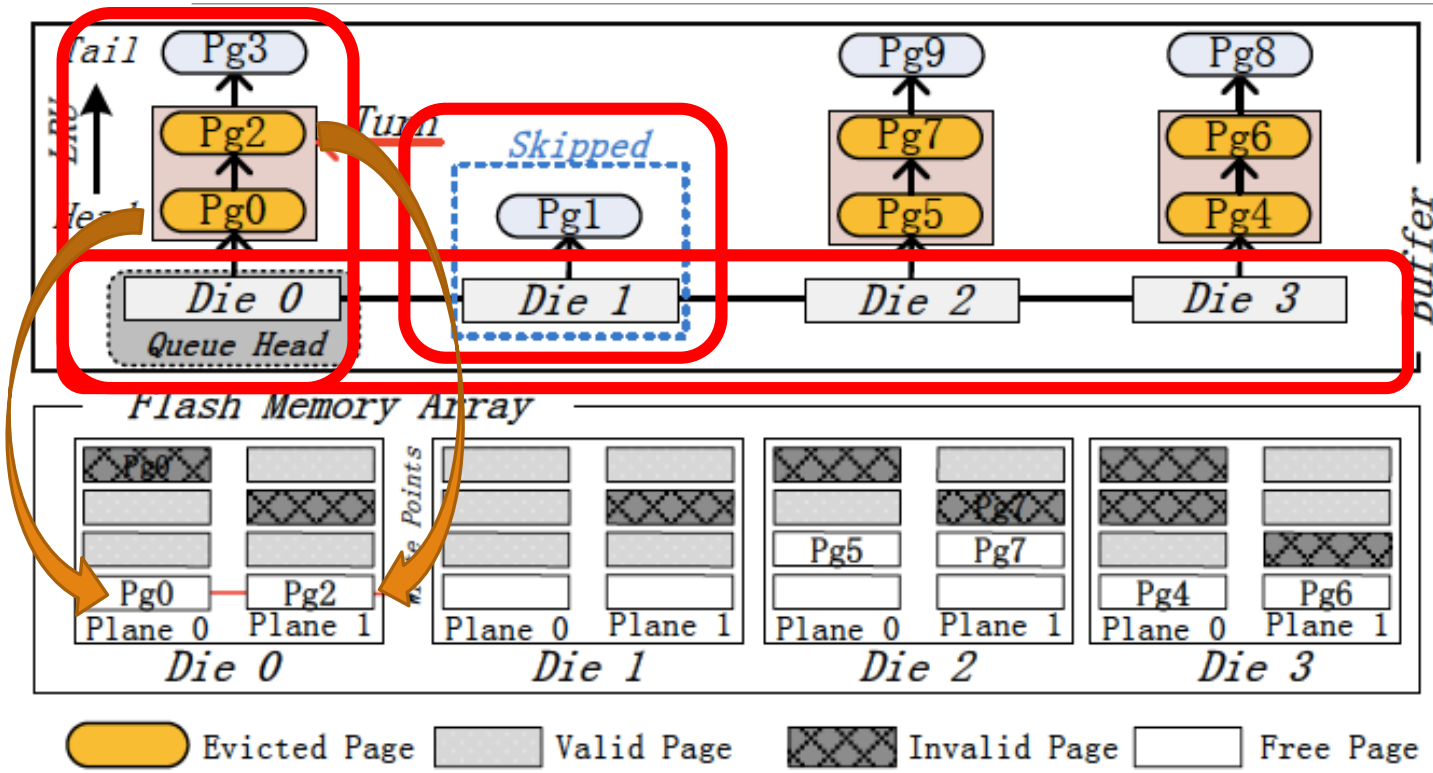
**2. The *starting locations of data* should be aligned for all the planes in the same die.**

**SSD buffer evicts a multiple of *N* dirty pages from one die at a time**

***Buffer Supported Die-Write***

**A plane level dynamic allocation scheme is adopted** [*Tavakkol et al. 2016*]

# Buffer Supported Die-Write




- ***A die queue is maintained;***
- ***Dirty pages are stored based on their die number;***
- ***Only die list containing at least 2 pages are selected.***

**Based on dynamic plane level data allocation,**

***Organization of write buffer and the die level write construction***

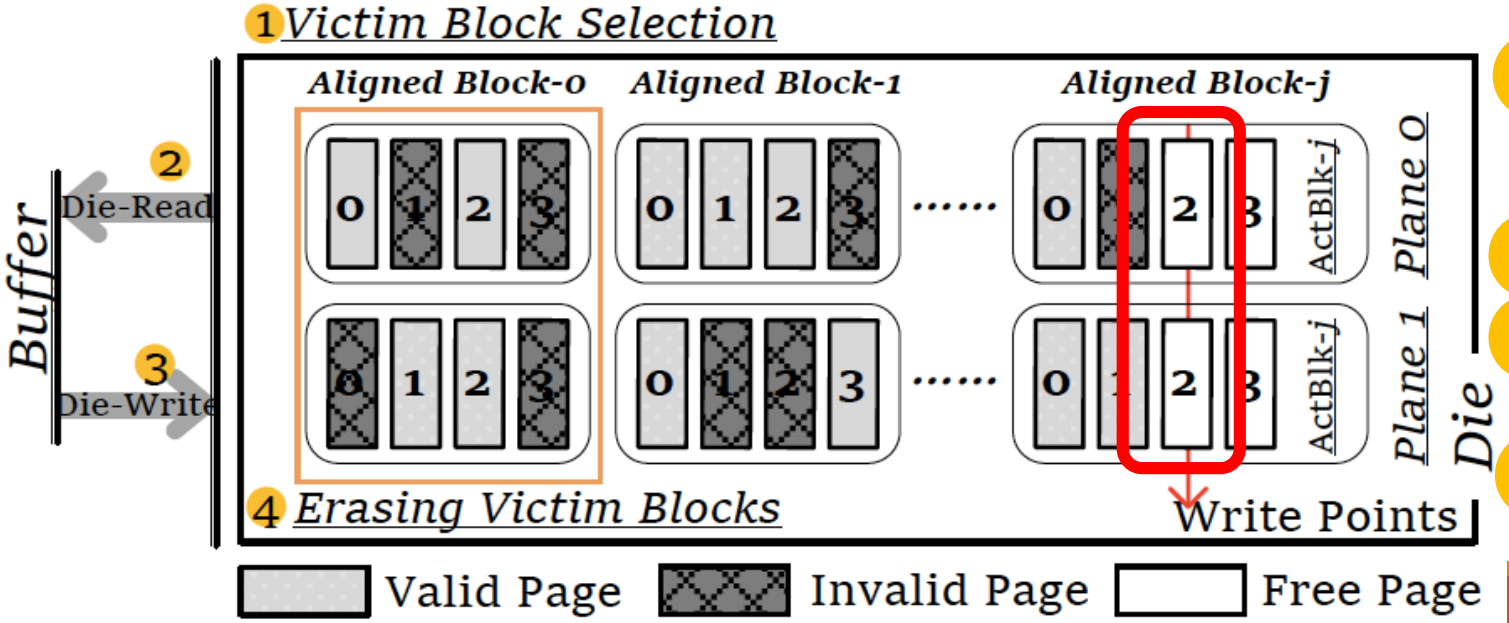
***Die-Write is constructed!!!***

# Die Level GC

**Traditional GC:** ***1. Victim block selection; 2. Valid page movement; 3. Victim block erase***

**Die-GC:** ***Two Goals***

**1. Aligning write points of all planes when GCs are activated;**
**2. Reducing the time cost of valid page movement.**




1. ***The selection process takes the N aligned blocks as a GC unit;***
2. ***Die-Read and Die-Write are used***
3. ***to align write points;***
4. ***Erase operations are executed in parallel without additional cost.***

# Outline

Background

Problem Statement

SPD: From Plane to Die Parallelism Exploration

- Overview
- Die Level Write Construction
- Die Level GC

***Experiment Setup***

Results

Conclusion

# Experiment Setup

- Parameters of Simulated SSD

| | |
|---|---|
| **SSD Configuration** | 512GB;16 Channels; 8 Chips/Channel; 1 Die/Chip; 2 Planes/Die;2048 Blocks/Plane; 256 Pages/Block; 4KB Page; |
| **Timing Parameters** | 0.075 ms for page read; 1.5 ms for page write; 3.8 ms for block erase; 25 ns for byte transfer. |

- Buffer Setting:
  - Size: 1/1000 of the footprint of evaluated workloads;
  - Page organization within a die list: LRU

- Evaluated Workloads

| Workloads | W/R Ratio§ | FP§ | R_V§ | W_V§ | R_S§ | W_S§ |
|---|---|---|---|---|---|---|
| HM_0 | 67.9% | 1.35 | 6.9 | 15.2 | 11.2 | 11.6 |
| PRN_0 | 93.7% | 2.93 | 3.0 | 20.5 | 24.8 | 11.6 |
| PRN_1 | 32.1% | 5.16 | 31.4 | 10.9 | 24.2 | 11.4 |
| RSR_0 | 90.7% | 0.31 | 1.8 | 14.6 | 15.0 | 12.6 |
| STG_0 | 76.9% | 0.28 | 7.4 | 9.3 | 33.6 | 12.6 |
| PROJ_0 | 82.9% | 1.58 | 7.2 | 56.5 | 21.9 | 35.7 |
| PROJ_3 | 4.89% | 1.86 | 21.6 | 2.8 | 11.9 | 29.9 |
| SRC2_0 | 88.6% | 0.52 | 1.9 | 13.6 | 12.2 | 11.0 |
| TS_0 | 82.6% | 0.57 | 4.9 | 15.9 | 17.5 | 11.8 |
| PRXY_0 | 97.06% | 0.17 | 0.27 | 5.8 | 9.6 | 6.2 |
| WDEV_0 | 79.9% | 0.34 | 3.2 | 9.2 | 16.5 | 12.1 |

§ W/R Ratio: Write and Read Requests Ratio;
FP: FootPrint (GB);
R_V/W_V: Read/Write Data Volume (GB);
R_S/W_S: Average Read/Write Request Size (KB).

# Experiment Setup

***Evaluated Schemes:***

- **Baseline-D:** Dirty pages are evicted to different dies for exploiting die level parallelism;
- **Baseline-P:** Based on Baseline-D, dirty pages accessing different planes in the same die are evicted at a time;
- **TwinBlk:** *Aligning write points of planes in the same die through sending data to different planes in a round-robin policy;*
- **ParaGC:** *Aligning write points of active blocks in different planes for reducing the time cost of valid page movement during GC process;*
- **Proposed SPD:**

# Outline

Background

Problem Statement

SPD: From Plane to Die Parallelism Exploration

- Overview
- Die Level Write Construction
- Die Level GC

Experiment Setup

***Results***

Conclusion

# Results

***Results without GC—Latency:***

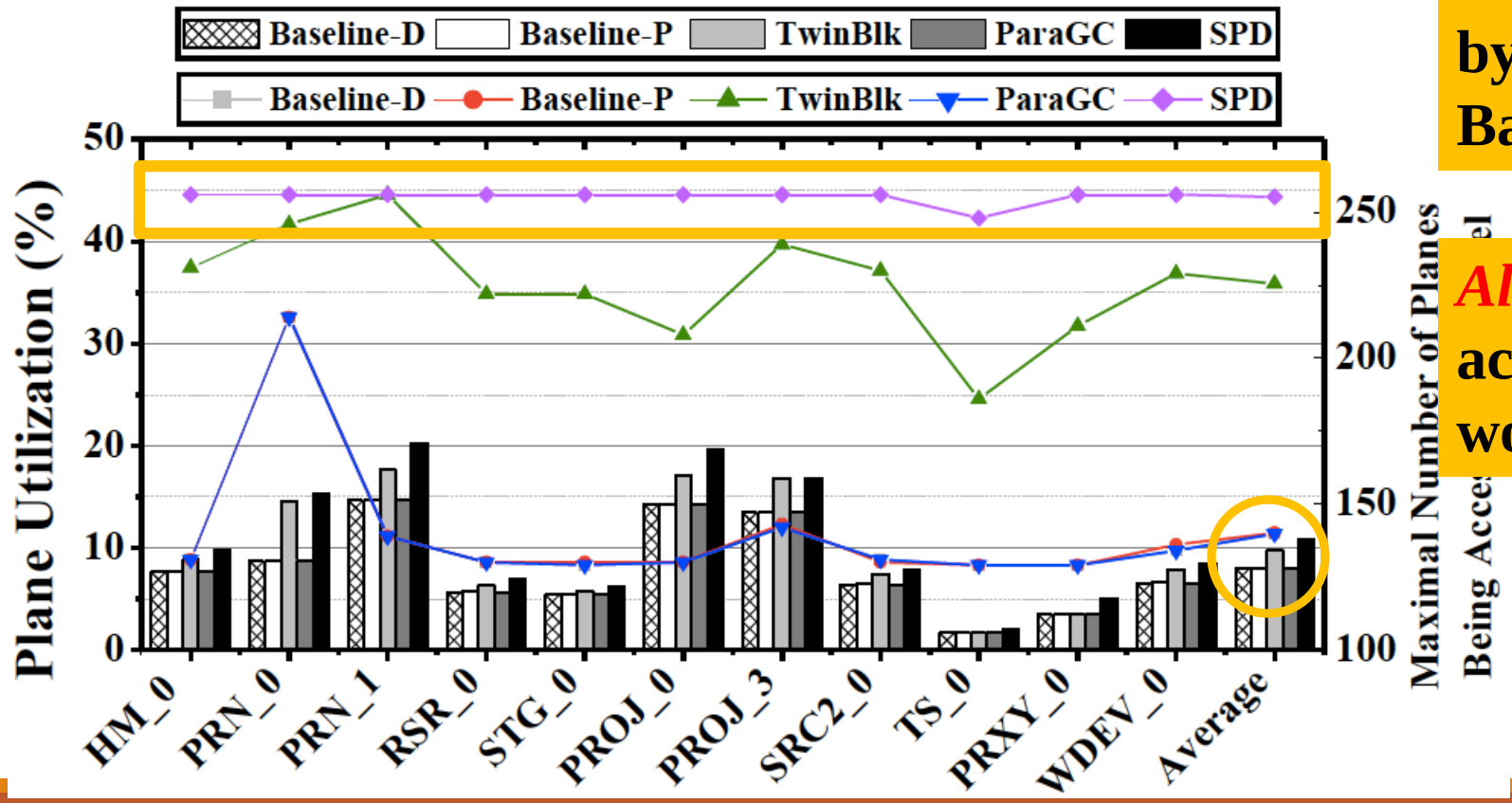
Read Latency Improvement without GC

| | Baseline-D | Baseline-P | TwinBlk | ParaGC | SPD |
|---|---|---|---|---|---|
| *Reduction* | 0 | 0.049% | 0.011 | 0% | 0.096% |

**Read Latencies of five evaluated schemes are similar.**

# Results

***Results without GC—Plane Utilization:***

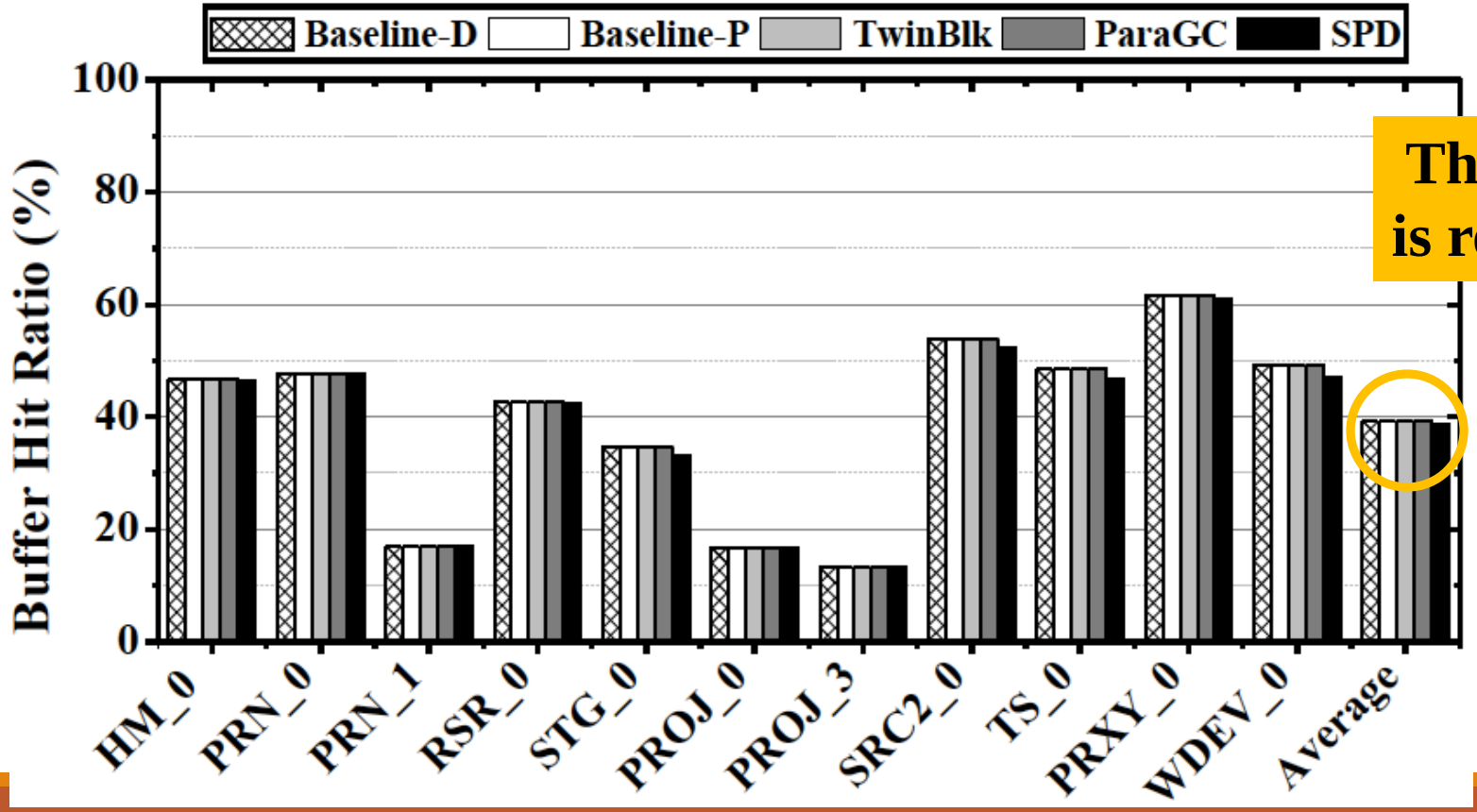



**Plane utilization is increased by *36.5%* compared with Baseline-D**

***All planes*** **of SSD can be accessed in parallel for most workloads.**

# Results

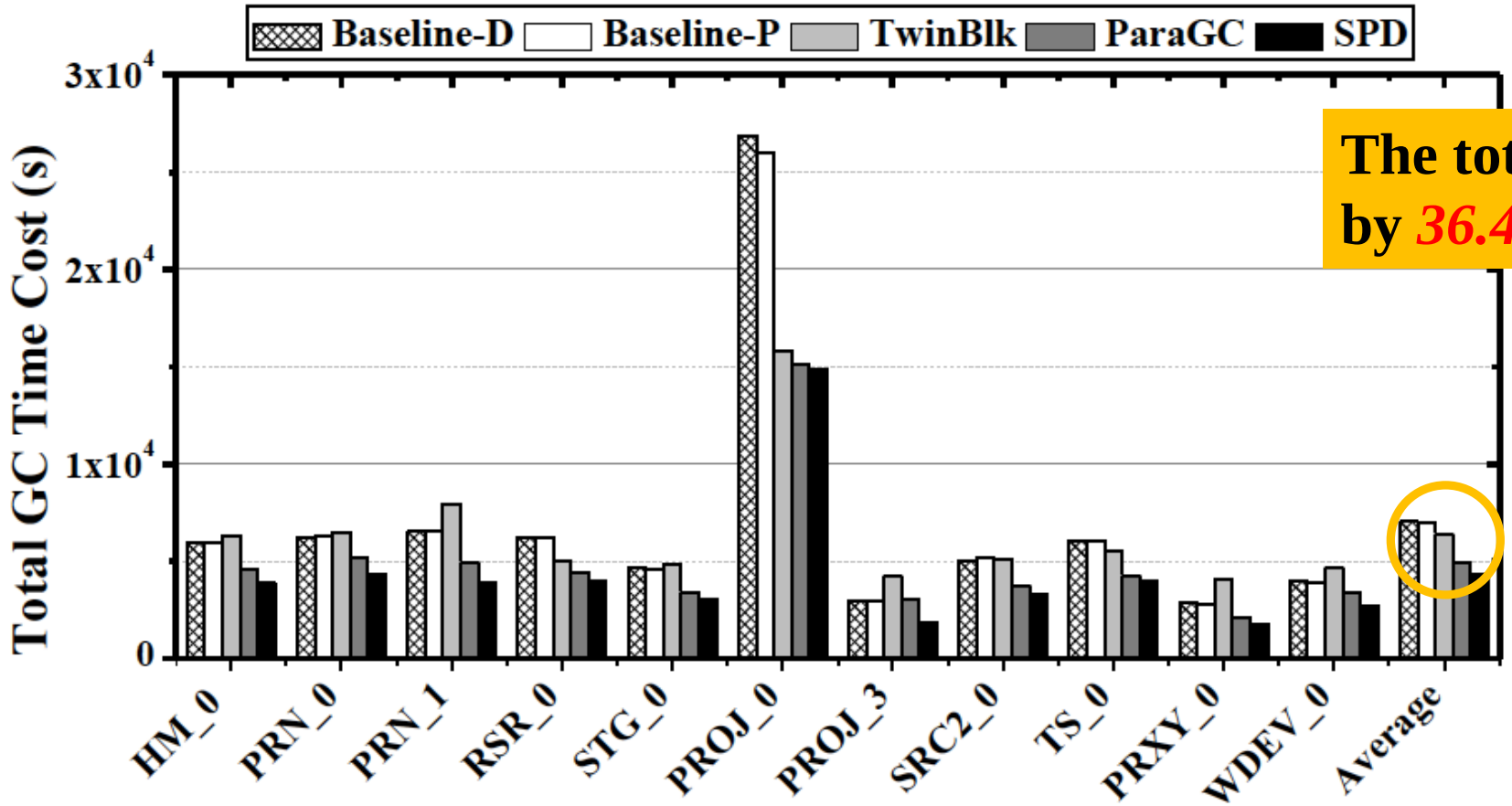
***Results without GC—Buffer Hit Ratio:***




**The average buffer hit ratio is reduced by only *1.92%***

# Results

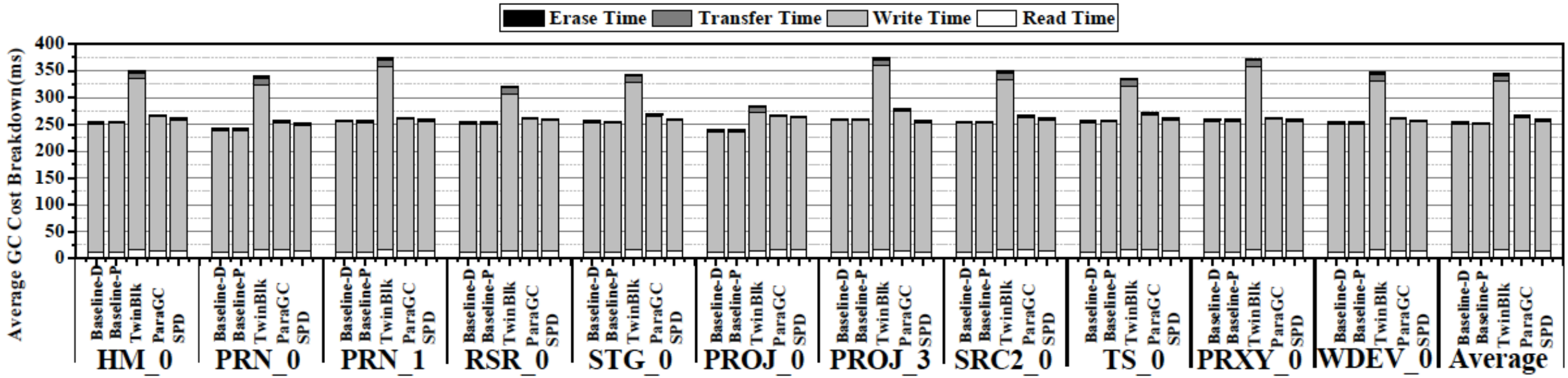
***Results with GC—Total GC Cost:***




**The total GC cost is reduced by *36.4%*, on average.**

# Results

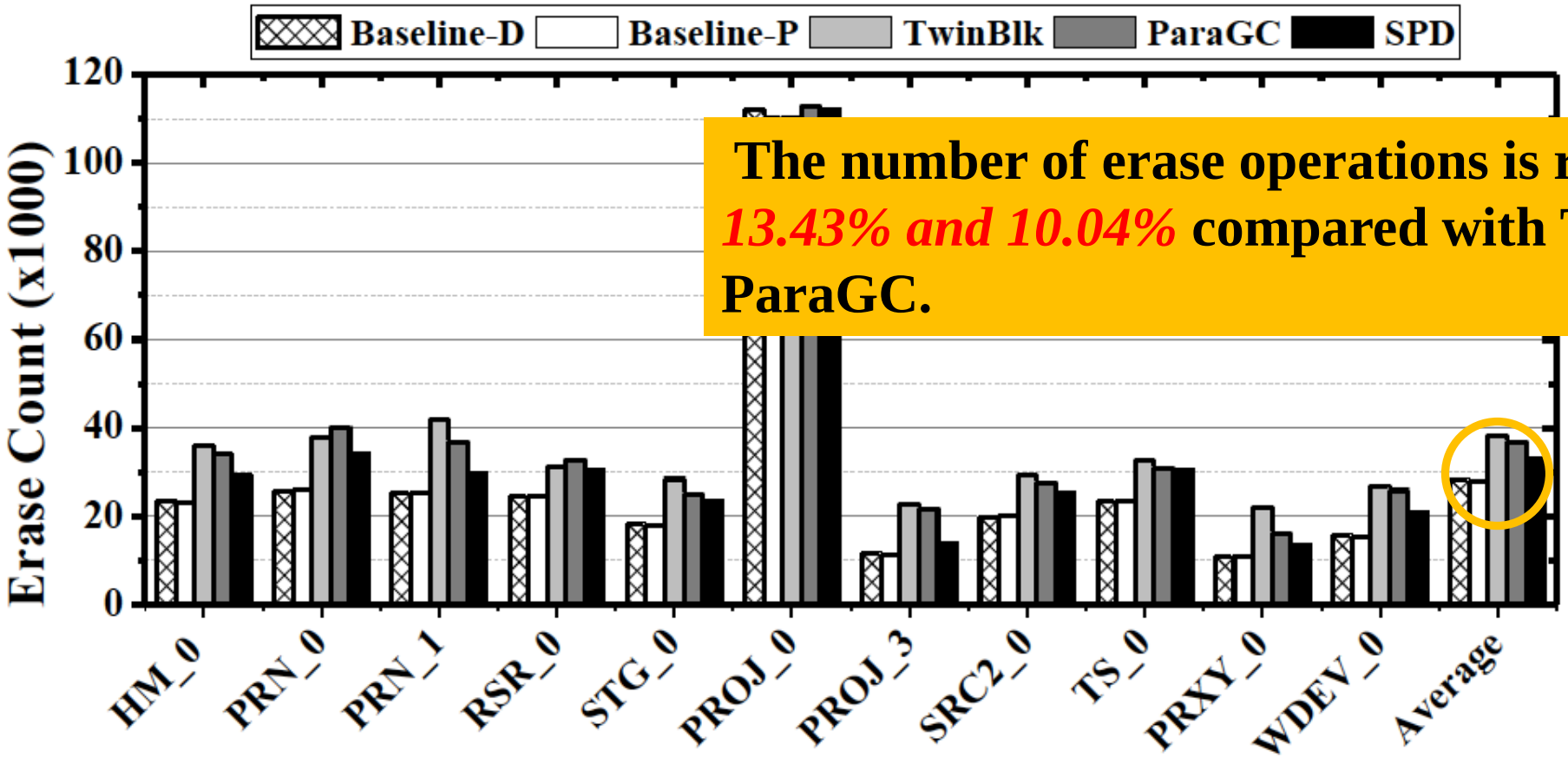
***GC Evaluation—Average GC Cost:***




1. **SPD has the minimal GC cost compared with TwinBlk and ParaGC;**
2. **The GC cost of SPD is similar to that of Baseline-D and Baseline-P.**

# Results

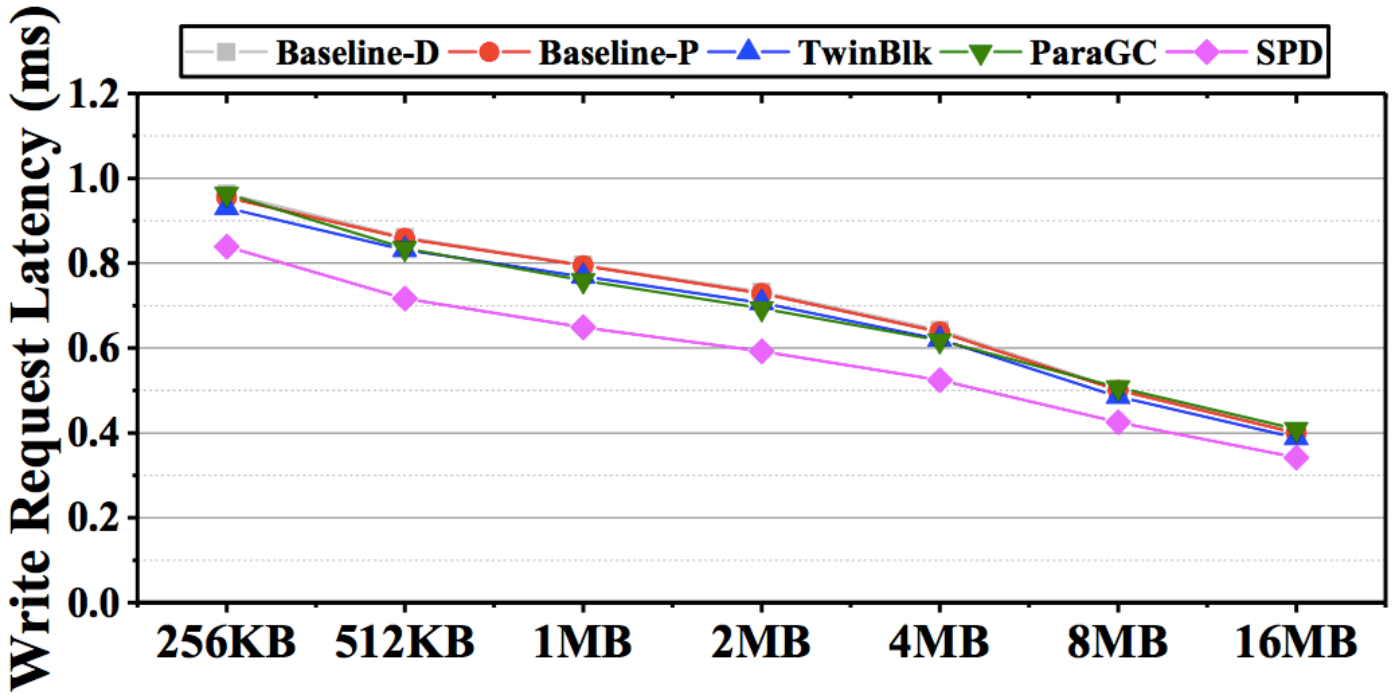
***GC Evaluation—GC Count and GC Induced Erases:***




**The number of erase operations is reduced by *13.43% and 10.04%* compared with TwinBlk and ParaGC.**

# Results

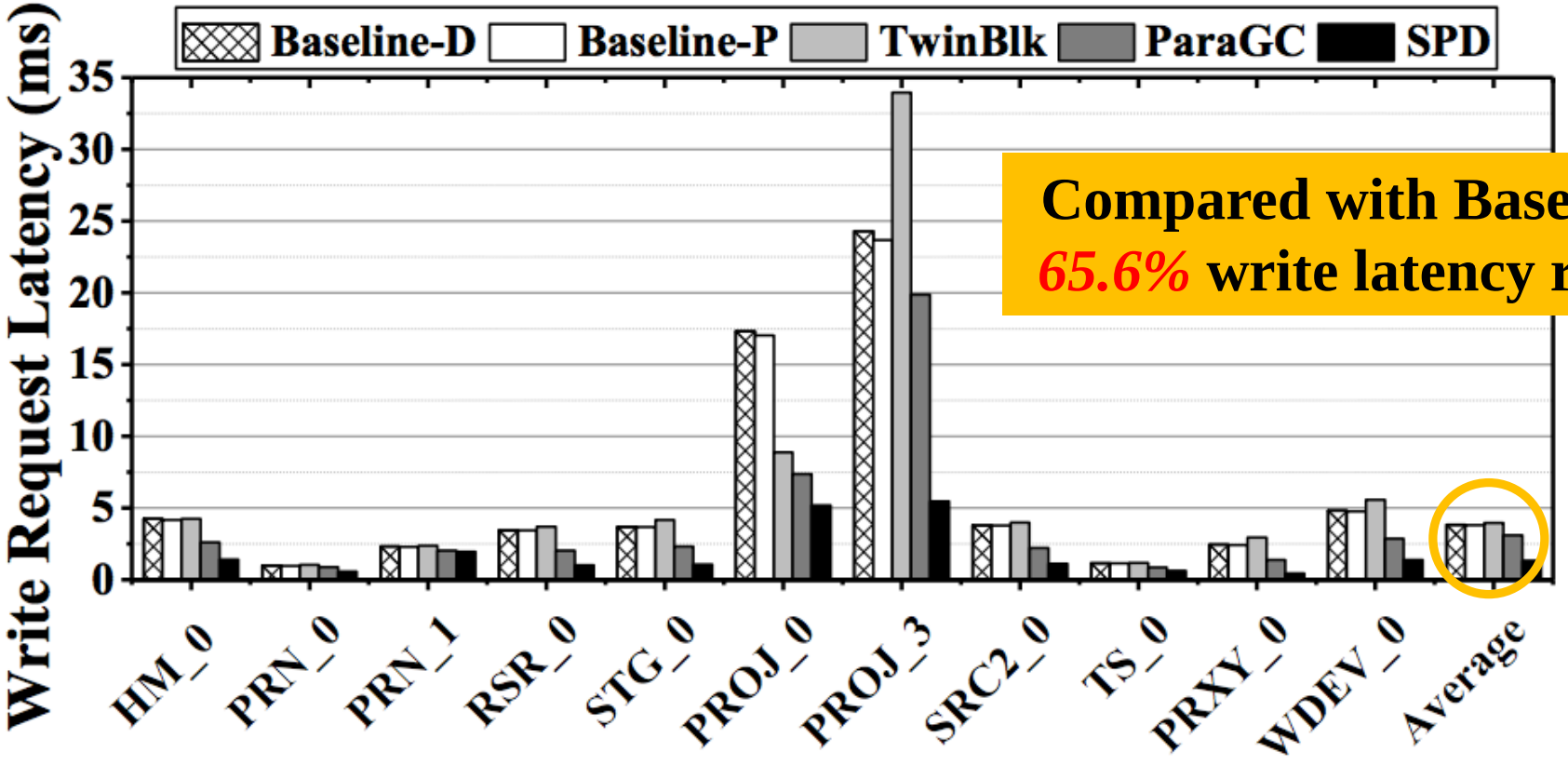
***Sensitive Study—Buffer Size:***




1. **With larger buffer size, the write latencies of all schemes can be further reduced;**
2. **Stable write latency reduction is achieved by SPD with different buffer sizes.**

# Resutls

***Sensitive Study—Four Planes:***




**Compared with Baseline-D, SPD achieves *65.6%* write latency reduction, on average**

# Conclusion

- **Two components are designed in the framework: Die-Write and Die-GC.**
  - **Aligning the write points of all planes in the same die all the time.**

- **The experimental results show that SPD effectively improves write performance of SSDs by *48.61%* on average without impacting read performance .**

# Thanks

*Q & A*