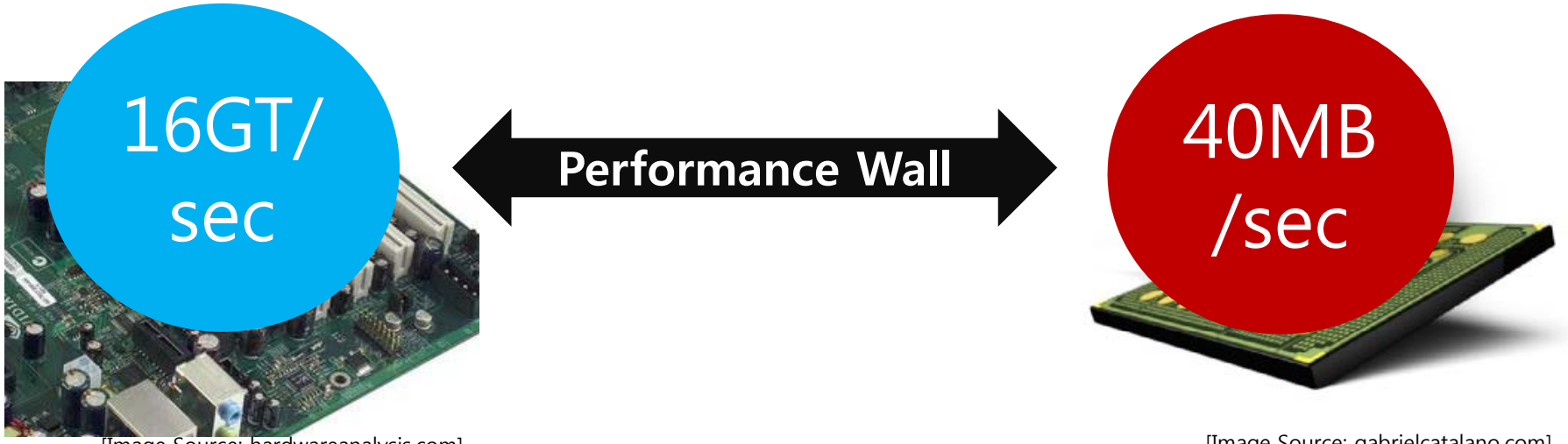
An Evaluation of Different Page Allocation Strategies on High-Speed SSDs

Myoungsoo Jung
Mahmut Kandemir

- Main insight
  - Exploiting internal parallelism is becoming a key design issue in SSDs
- Different level parallelism
  - SSD systems employ multiple NAND flash packages (system-level)
  - In parallel, NAND flash technologies are being developed to extract maximizing parallelism (flash-level)
- Internal parallelism behaviors are still unclear!!
  - simulated 24 flash page allocation strategies geared toward exploiting both system and flash level parallelism
  - present several optimization points based on our findings

Performance Disparity

- High-performance interfaces
  - 16GT/sec in PCI-e Gen3
  - 6Gb/sec in SATA3
- NAND flash memory
  - Maximum bandwidth is limited by 40MB/sec



[Image Source: hardwareanalysis.com]

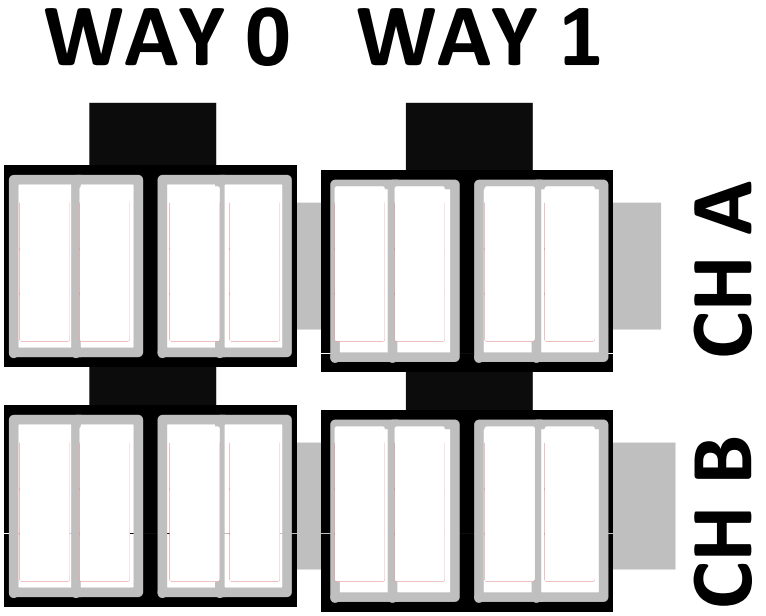
[Image Source: gabrielcatalano.com]

How can we reduce the performance disparity?

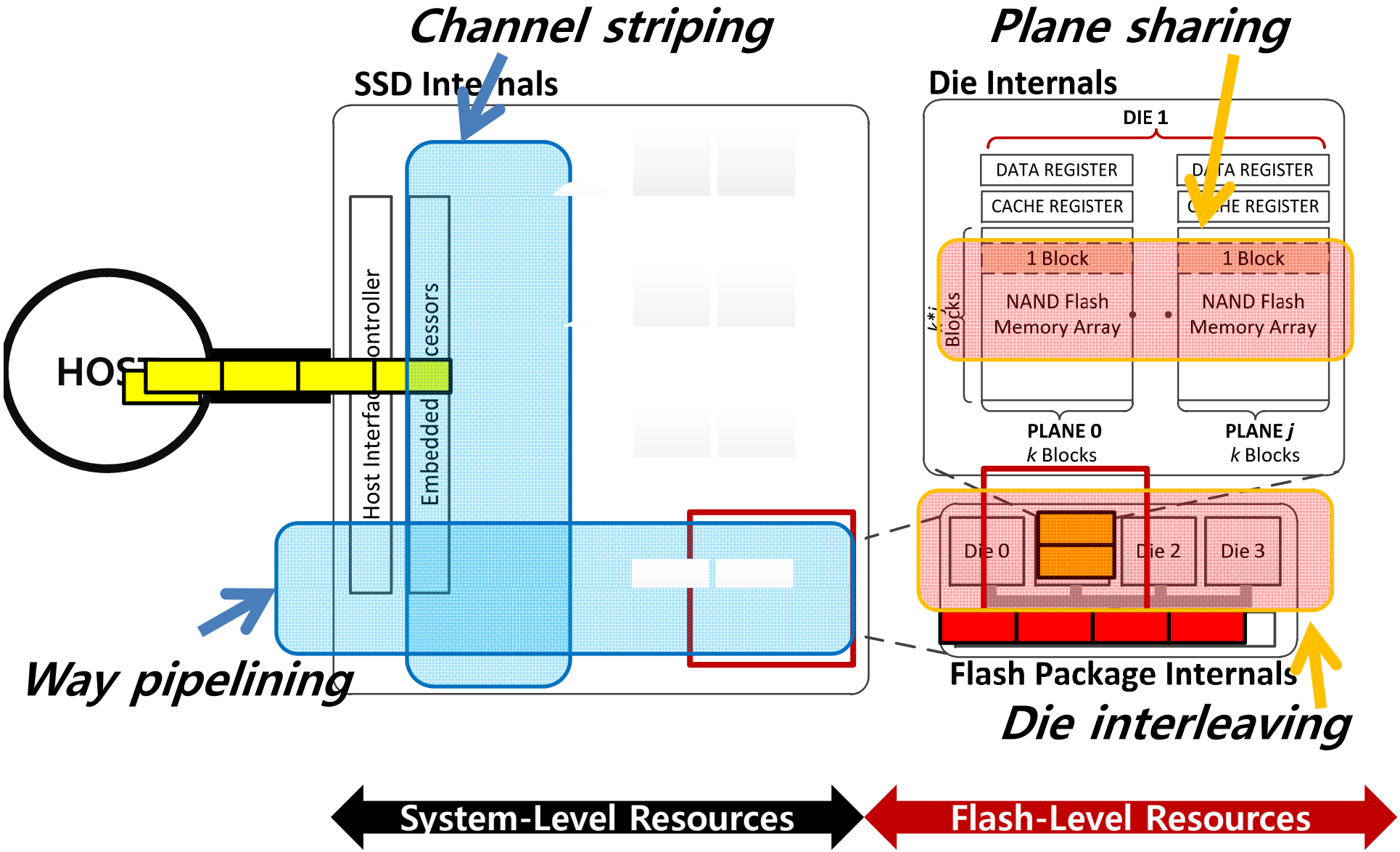
High-speed Interface

Internal parallelism and parallelizing data accesses are key

Multiple Flash Packages



Internals of SSD and Flash



Software Stack & Page Allocation

Host Interface Layer

Page allocation strategies are directly related with physical data layout and access sequences, which have impact on the performance and internal parallelism

Fl

address space and physical addresses

Hardware Abstraction Layer

Committing flash transaction to underlying flash memory chips

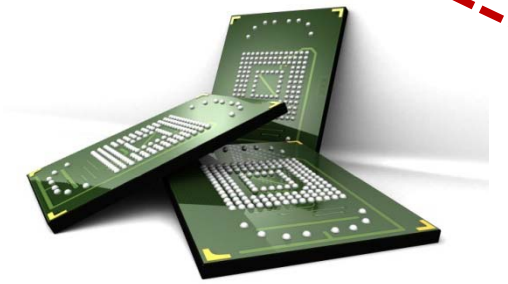
HIL

FTL

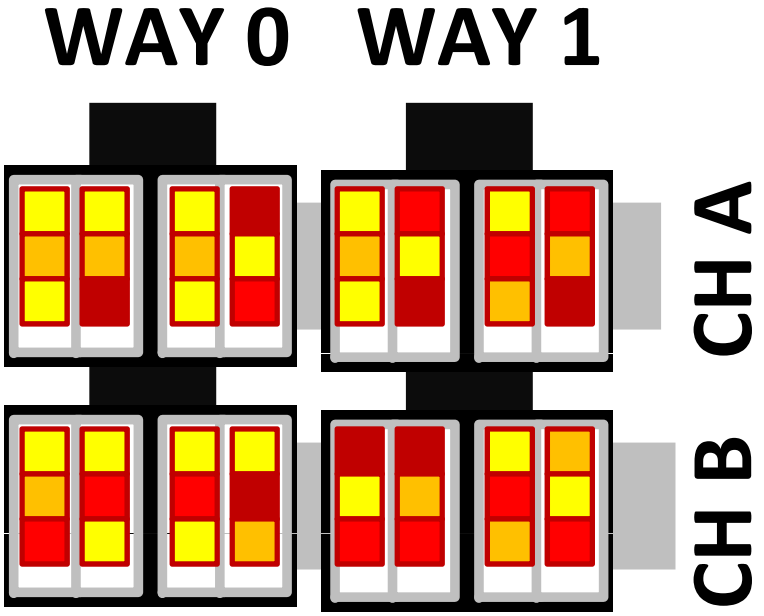
HAL

24

Page Allocation Strategies



[Image:micron.com]



Questions & Findings

- Which page allocation scheme would be globally optimal?

Flash-level resource first page allocation strategies have a better position in a performance angle

- What are the relationship between different level concurrency methods?

Channel first page allocation schemes render high flash-level parallelism difficult

- What are the resource utilization of different page allocation strategies?

With most of the current parallel data access methods, internal resources are significantly underutilized

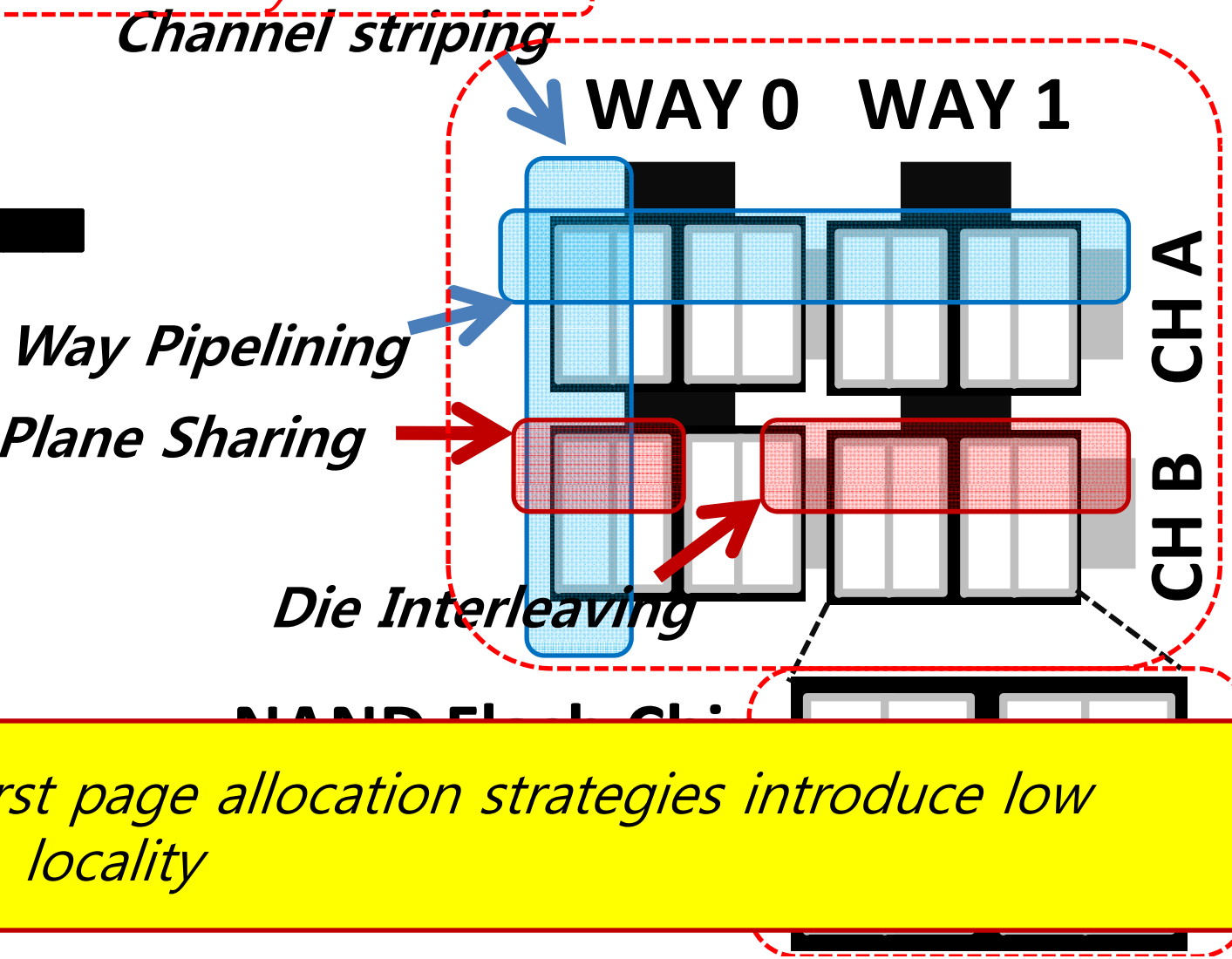
Page Allocation Strategies (Palloc)

- Channel-first pallocs
  - Allocate internal resources in favor of channel striping method
- Way-first pallocs
  - Are oriented forward taking advantage of the way pipelining
- Die-first and plane-first pallocs
  - Allocate die and plane in an attempt to reap the benefit of flash-level parallelism

Channel-first Page Allocation

CWDP -- *Channel-Way-Die-Plane*

CDPW
CDWP
CPDW
[redacted]
CPWD
CWPD



Channel first page allocation strategies introduce low flash-level locality

Way-first Page Allocation

WDCP -- *Way-Die-Channel-Plane*

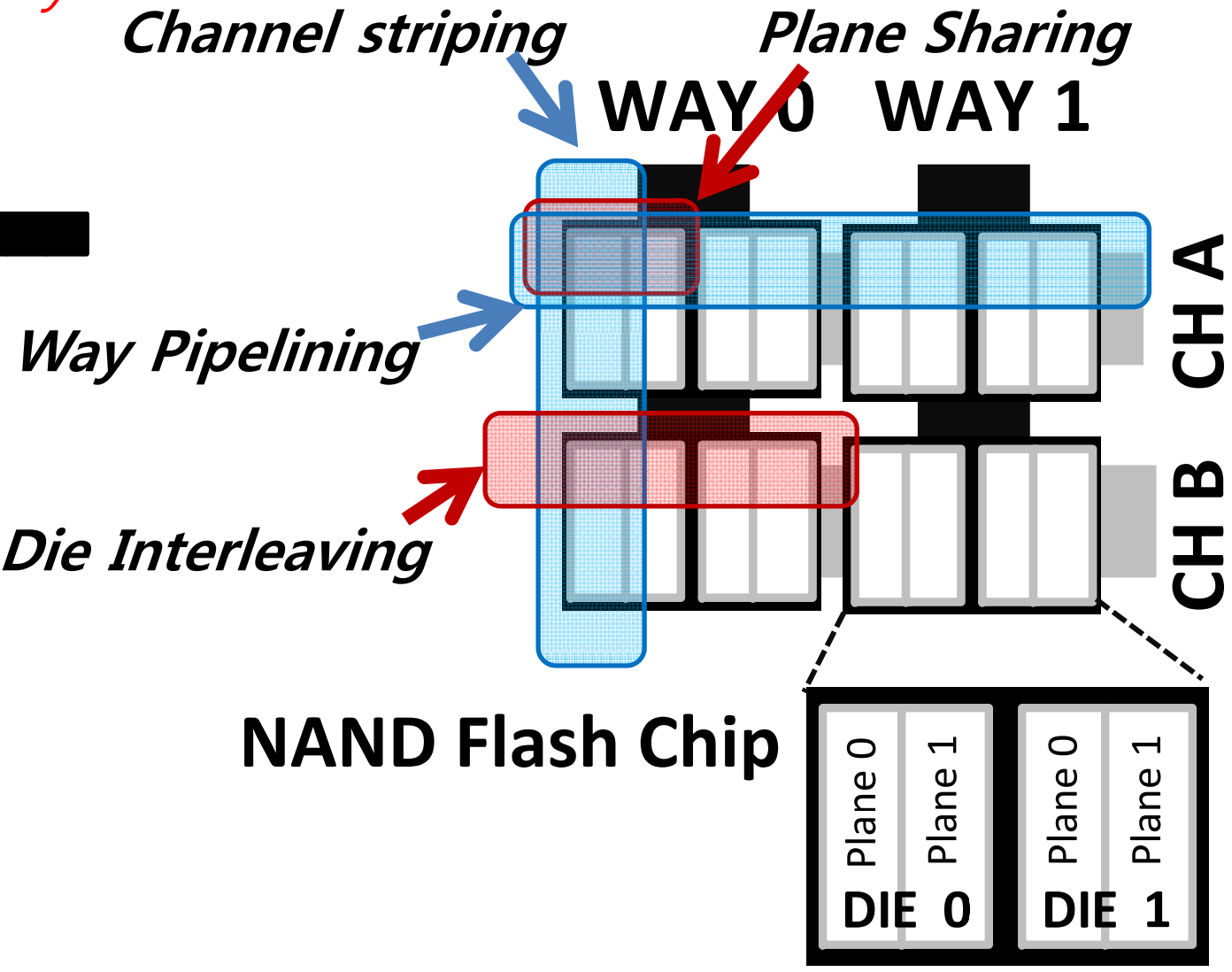
WCPD

WDCP

WDPC

WPCD

WPDC



Die-first Page Allocation

DPWC -- *Die-Plane-Way-Channel*

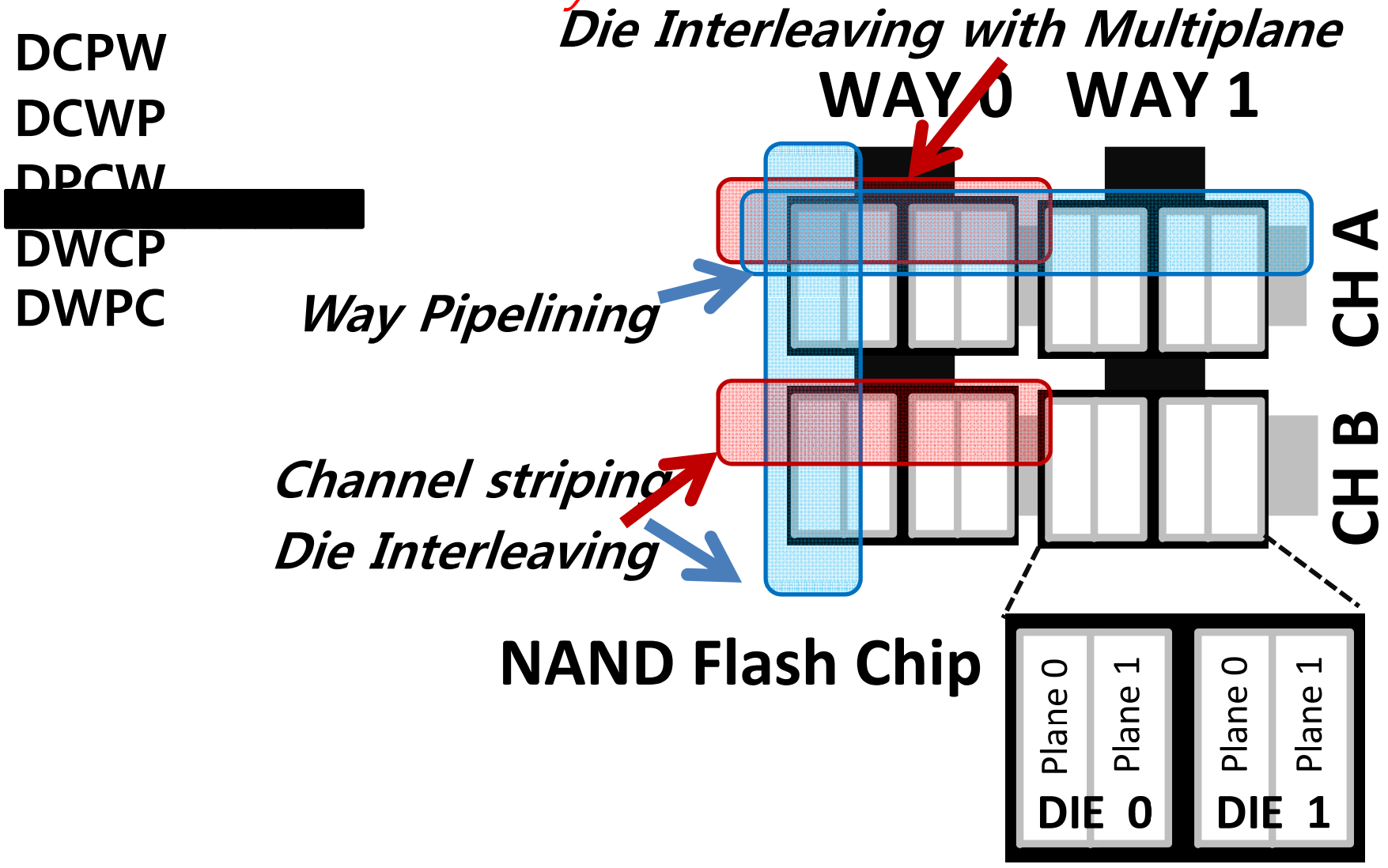
DCPW

DCWP

DPCW

DWCP

DWPC



Plane-first Page Allocation

PWCD -- *Plane-Way-Channel-Die*

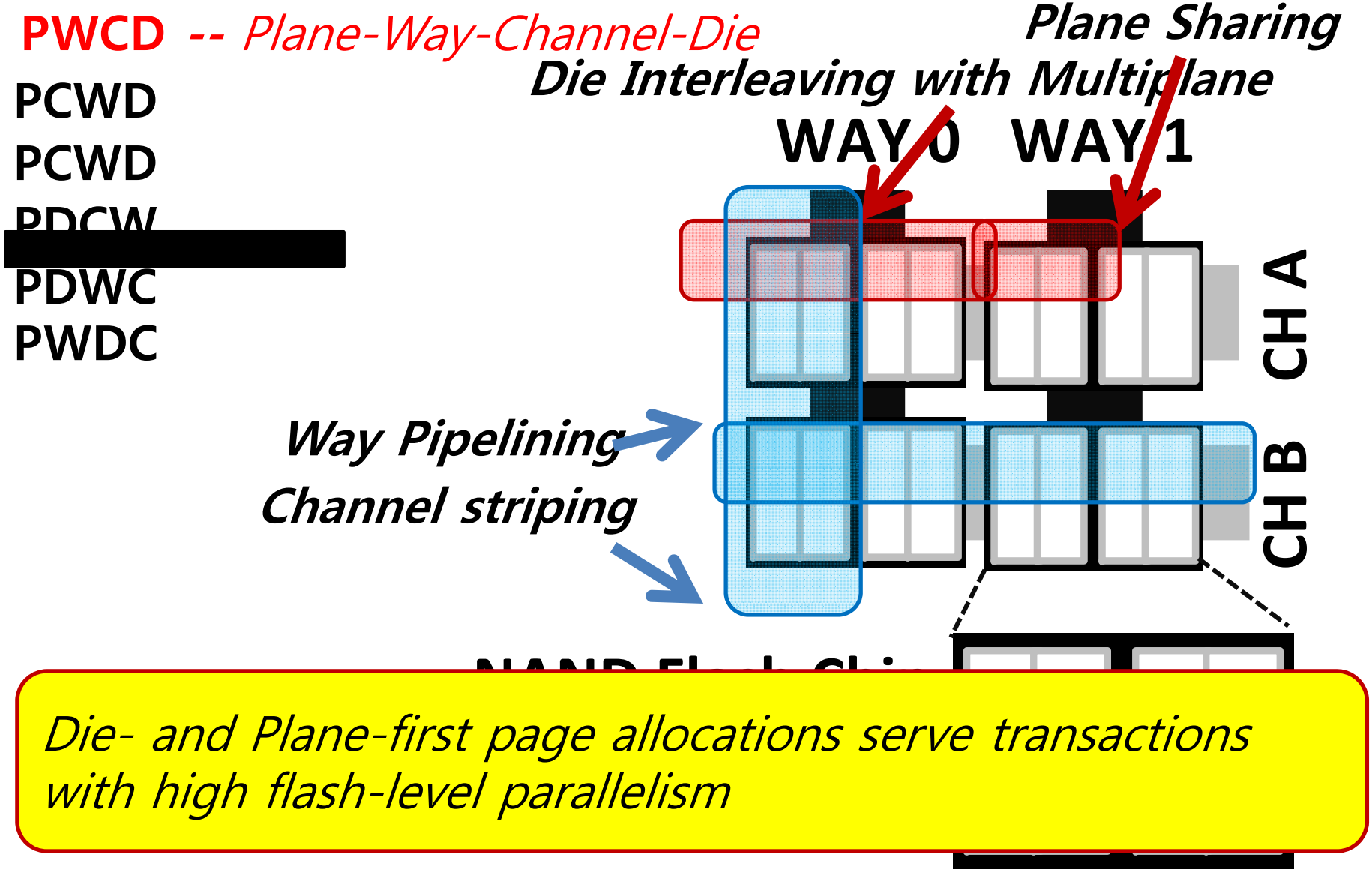
PCWD

PCWD

PDCW

PDWC

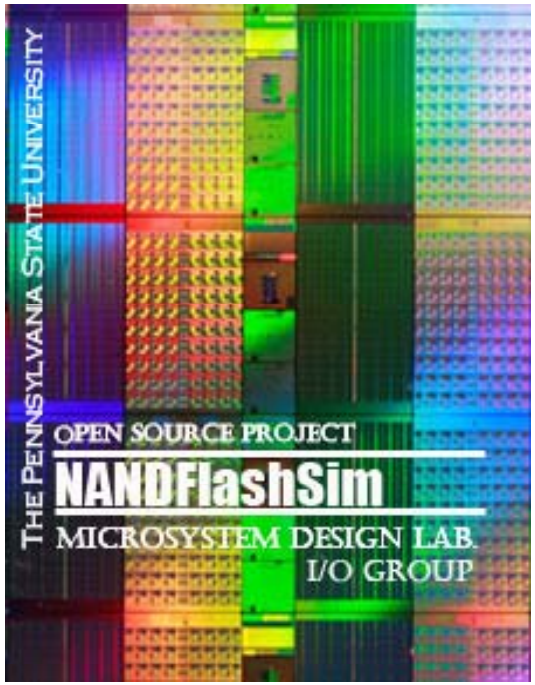
PWDC



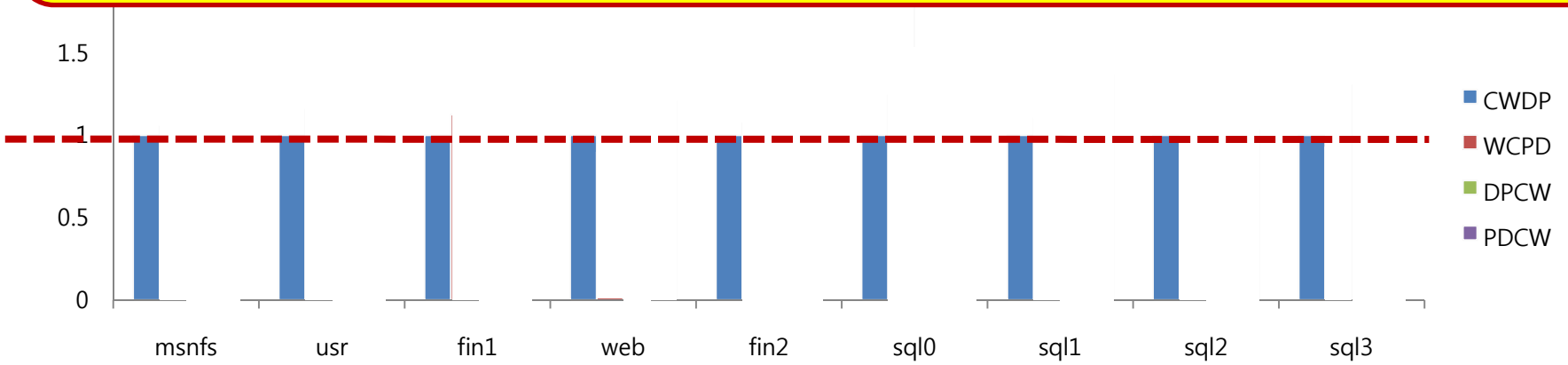
Die- and Plane-first page allocations serve transactions with high flash-level parallelism

SSD Setup

- NAND Flash Chip
  - Fine-grained NAND command
  - Advanced commands
  - Strong address constraints
  - Intrinsic latency variation
- SSD Framework
  - 8 channels, 8 flash per channel (64 total)
  - Dual-die package format, 32 entry queue
  - A page-level mapping and greedy garbage collection algorithm

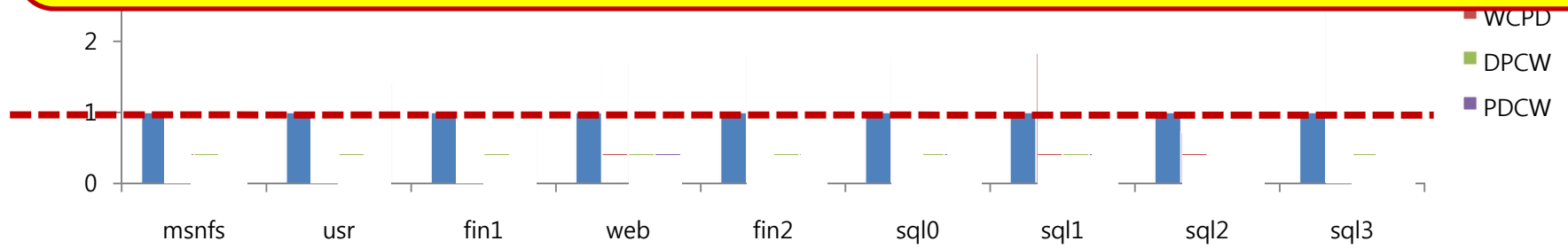


Way and flash-level resource first pallocs have better IOPS performance position than channel-first palloc



[Normalized IOPS]

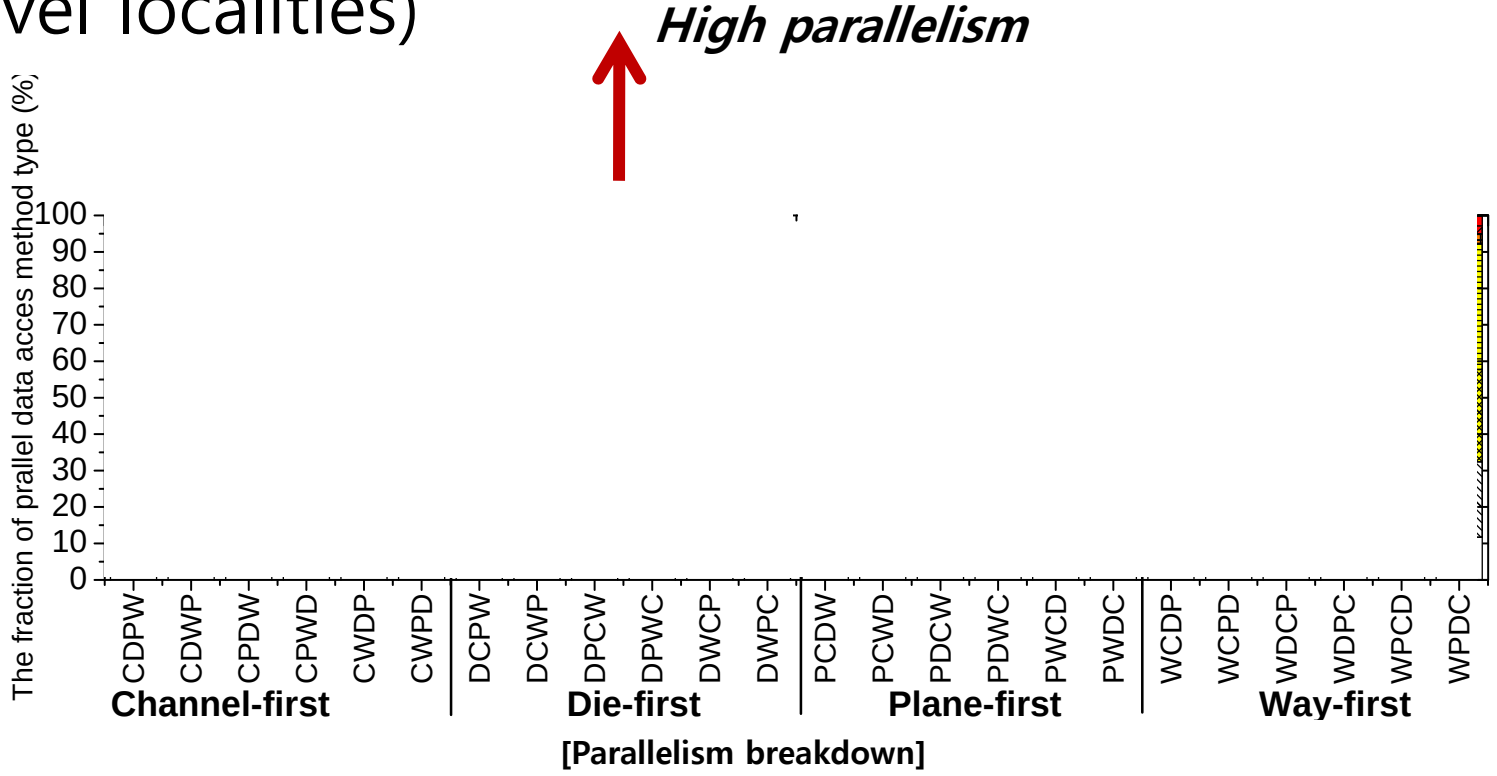
Channel-first palloc provide shorter latencies than flash-level resource first pallocs



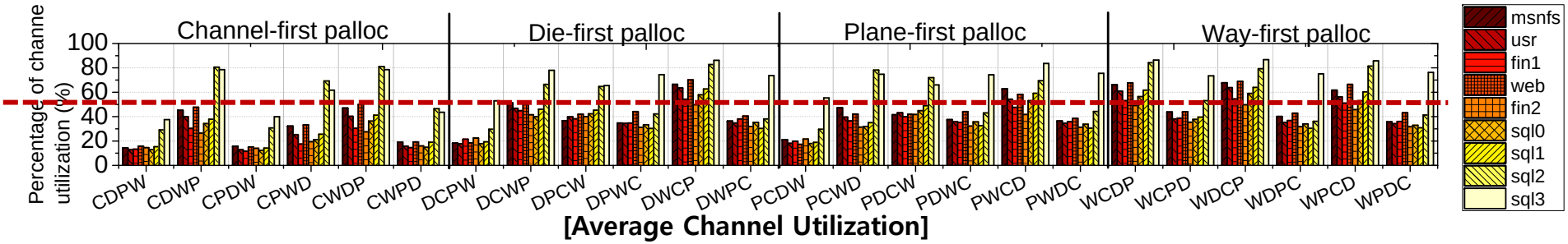
[Normalized Latency]

Parallelism Breakdown

- Low flash-level parallelism is observed under palloc schemes in favor of channel
- They render advanced flash command compositions difficult at runtime (due to low flash-level localities)

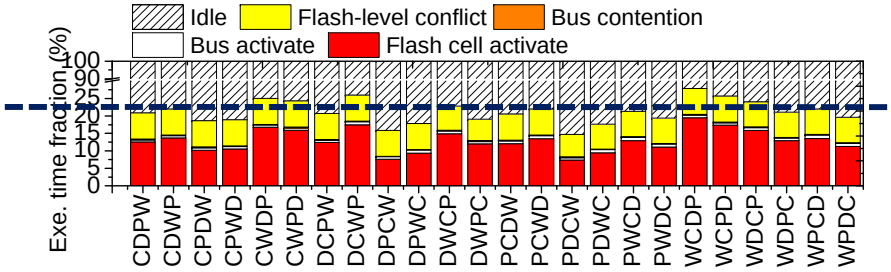


Resource Utilization

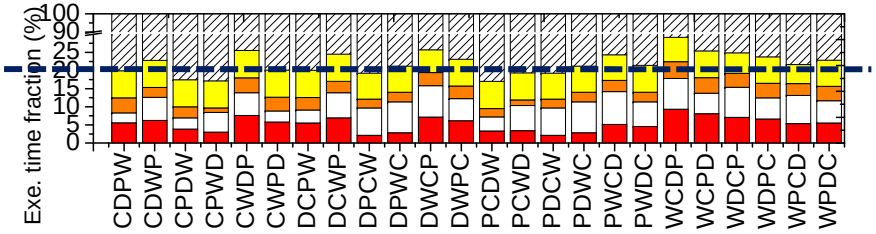


[Average Channel Utilization]

- **channel resources**
  - are utilized about 43.1% on average with most parallel data access methods
- **Idle time**
  - About 80% of the total execution time are spent idle

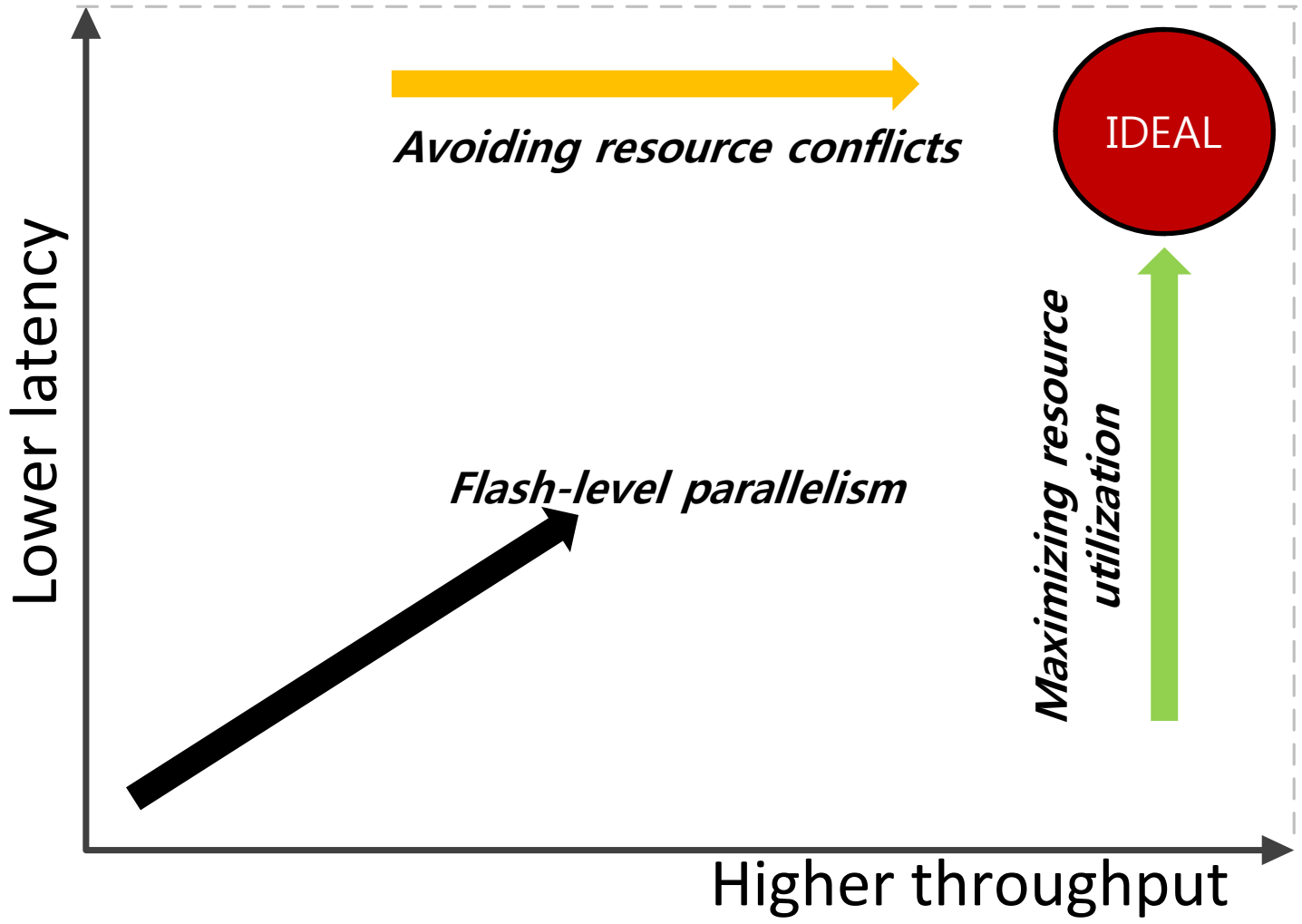


[Write Intensive -- Execution breakdown]



[Read Intensive -- Execution breakdown]

Optimization Point



Future work

- Future work
  - Incorporating a high-speed flash interface (e.g., 400MHz)
  - Further evaluating with varying parameters (e.g., different queue/buffer management, flash firmware, varying internal resource parameters)
  - More diverse workloads including micro-benchmark

Conclusion

1. The channel-and-way striping based palloc scheme is not the best
   - Overall, flash-level resource first palloc schemes have better performance
2. Channel-first page allocation introduces poor flash-level locality and parallelism
3. With most of the current parallel data access methods, internal resources are significantly underutilized

Q&A

- Acknowledgements
  - Umesh Maheshwari (our shepherd)
  - Ellis H. Wilson (Pennsylvania State Univ.)
  - John Shalf (Lawrence Berkley National Lab.)