SSD Architecture and System-level Controllers

These slides were made by

Prof. Myoungsoo Jung of Yonsei University.

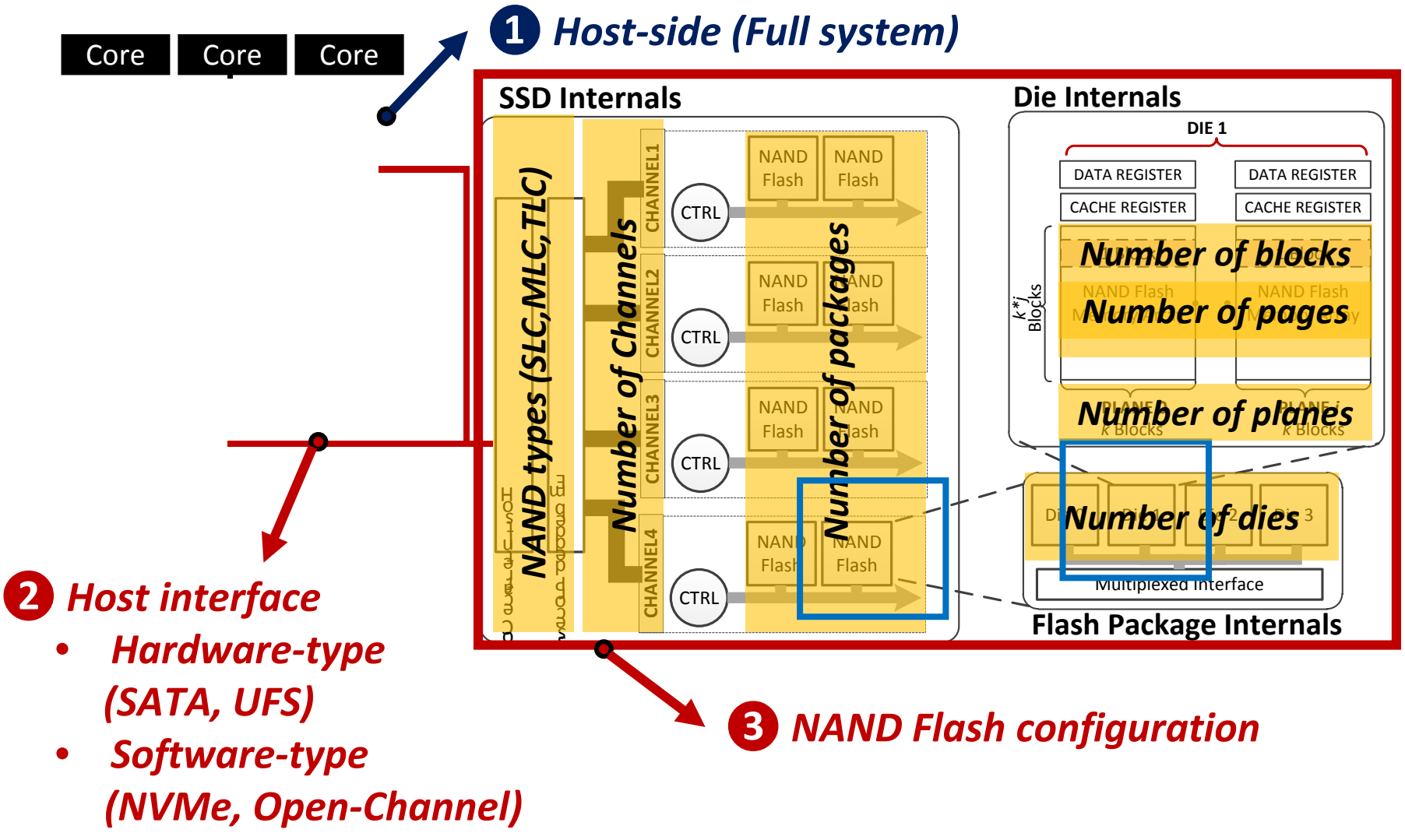
Outline

- Holistic Viewpoint (Overview)
- SSD Architecture
- Parallelism Overview
- Page Allocation Strategies
- Evaluation Studies for Parallelism
- Host Interface Overview
- Case Studies
  - Parallelism-Aware Host Interface I/O Scheduler
  - GC-Aware Host Interface I/O Scheduler
- Simulation Infrastructure (for your future research)

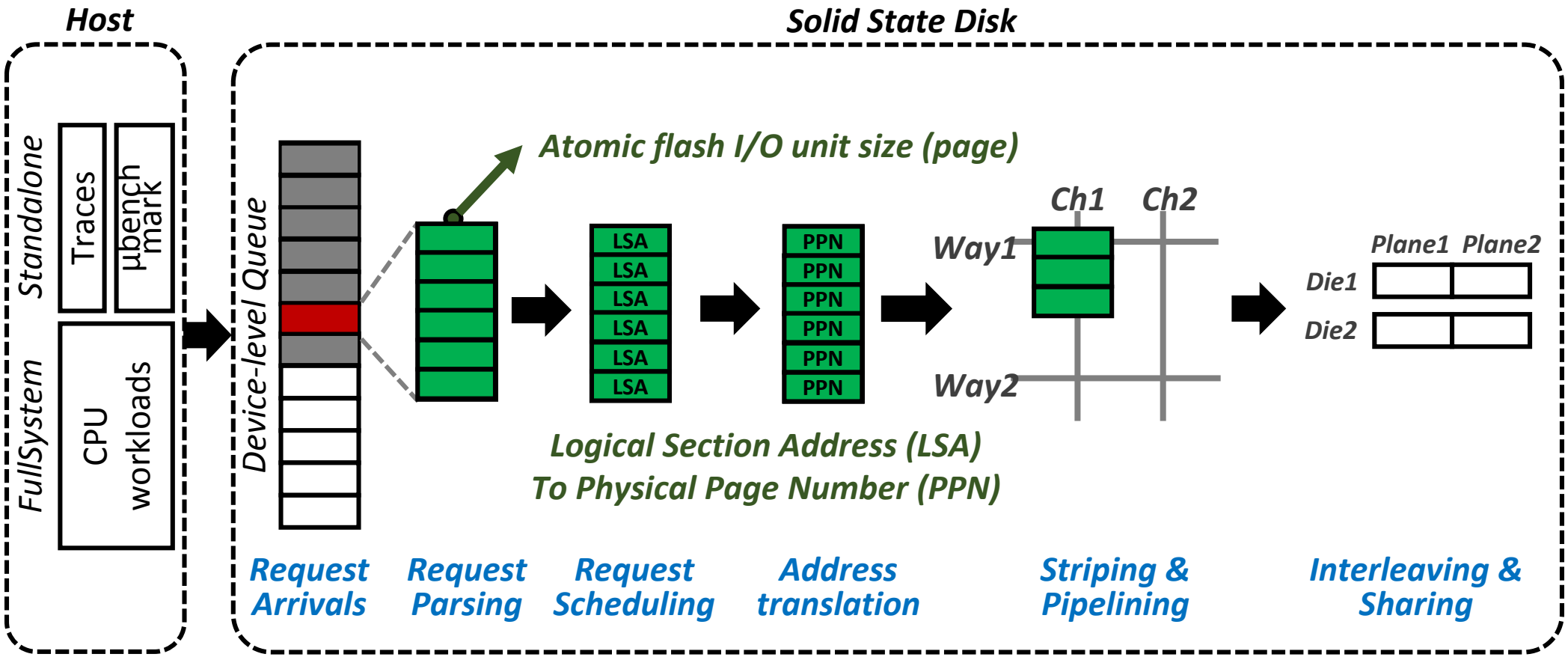
Outline

- Holistic Viewpoint (Overview)
- SSD Architecture
- Parallelism Overview
- Page Allocation Strategies
- Evaluation Studies for Parallelism
- Host Interface Overview
- Case Studies
  - Parallelism-Aware Host Interface I/O Scheduler
  - GC-Aware Host Interface I/O Scheduler
- Simulation Infrastructure (for your future research)

Holistic Viewpoint (Hardware)



Holistic Viewpoint (Software)

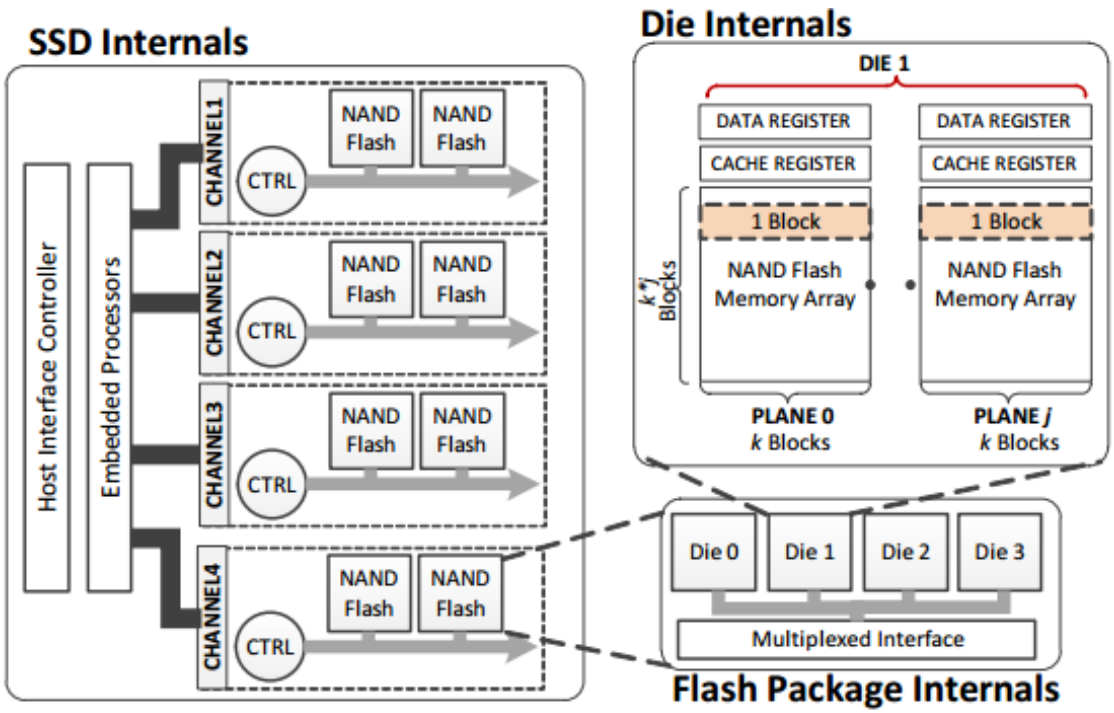


Outline

- Holistic Viewpoint (Overview)
- SSD Architecture
- Parallelism Overview
- Page Allocation Strategies
- Evaluation Studies for Parallelism
- Host Interface Overview
- Case Studies
  - Parallelism-Aware Host Interface I/O Scheduler
  - GC-Aware Host Interface I/O Scheduler
- Simulation Infrastructure (for your future research)

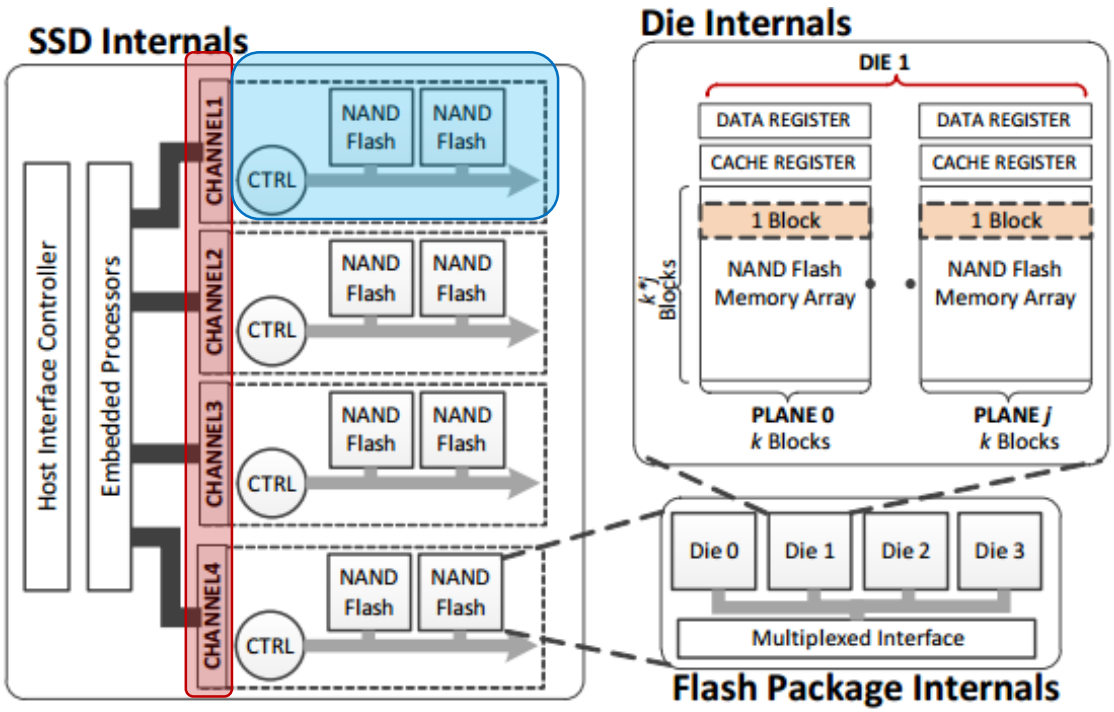
SSD Architecture

- At the top of SSD internals, there is host interface controller that parses the incoming requests
- Embedded CPU(s) is employed for flash firmware such as
  FTL, buffer $,
  I/O scheduler,
  parallelism
  management



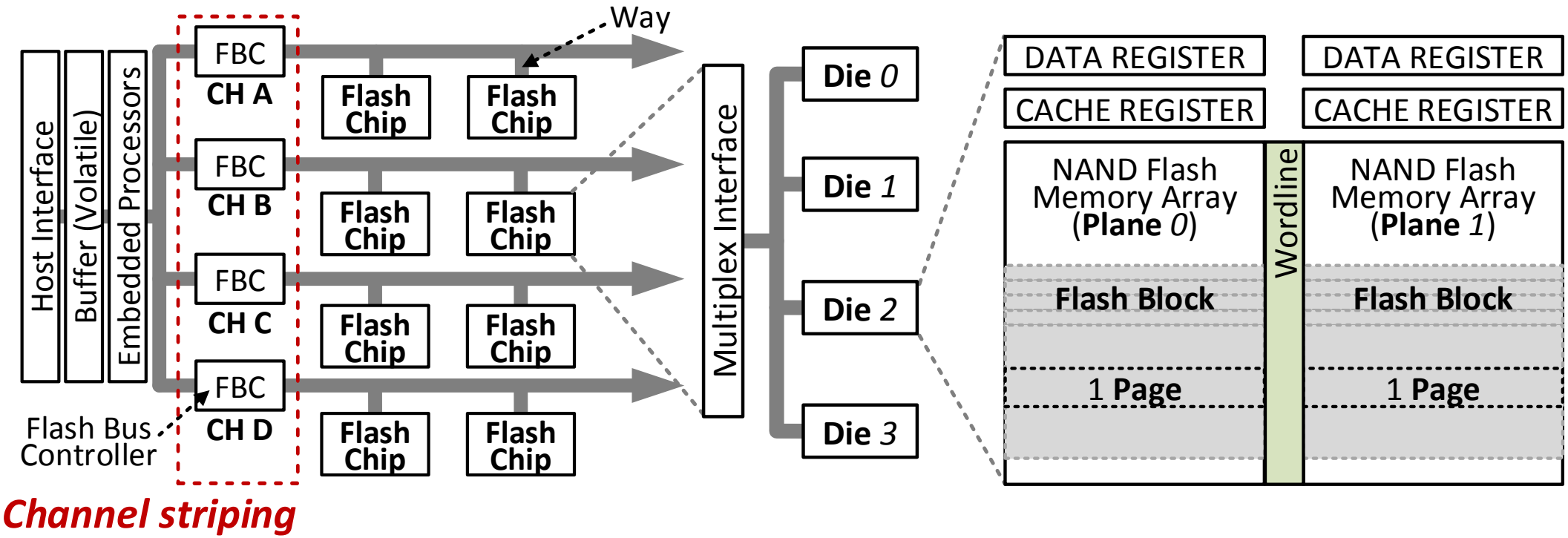
SSD Architecture

- Underneath the embedded processor, multiple flash controllers exist, each connecting a memory bus, referred to as channel
- Within a bus, there are multiple flash packages, each having flash interface, called way



SSD Internal Parallelism

1) System-level parallel data access ***2) Flash-level parallel data access***

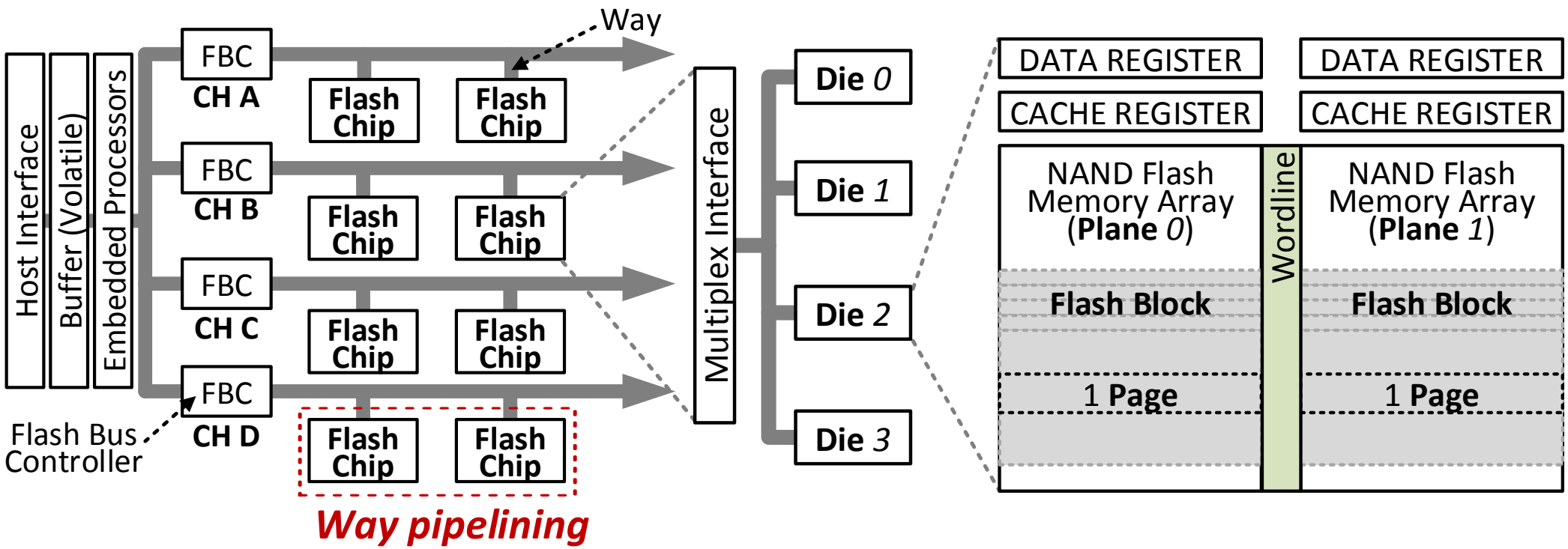


Channel striping

- An I/O request is striped over multiple channels

SSD Internal Parallelism

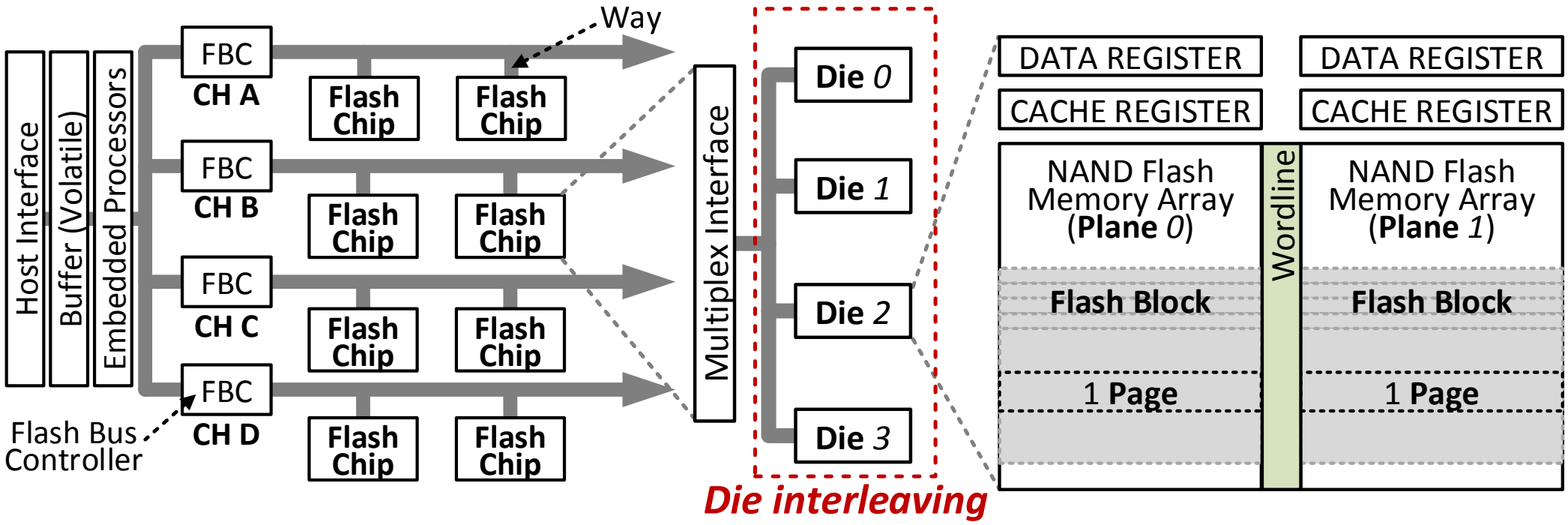
1) System-level parallel data access 2) Flash-level parallel data access



- As it shares the channel, an I/O request cannot be perfectly striped in parallel
- However, NAND flash transactions consists of multiple phases, and individual NAND flash can still work simultaneously

SSD Internal Parallelism

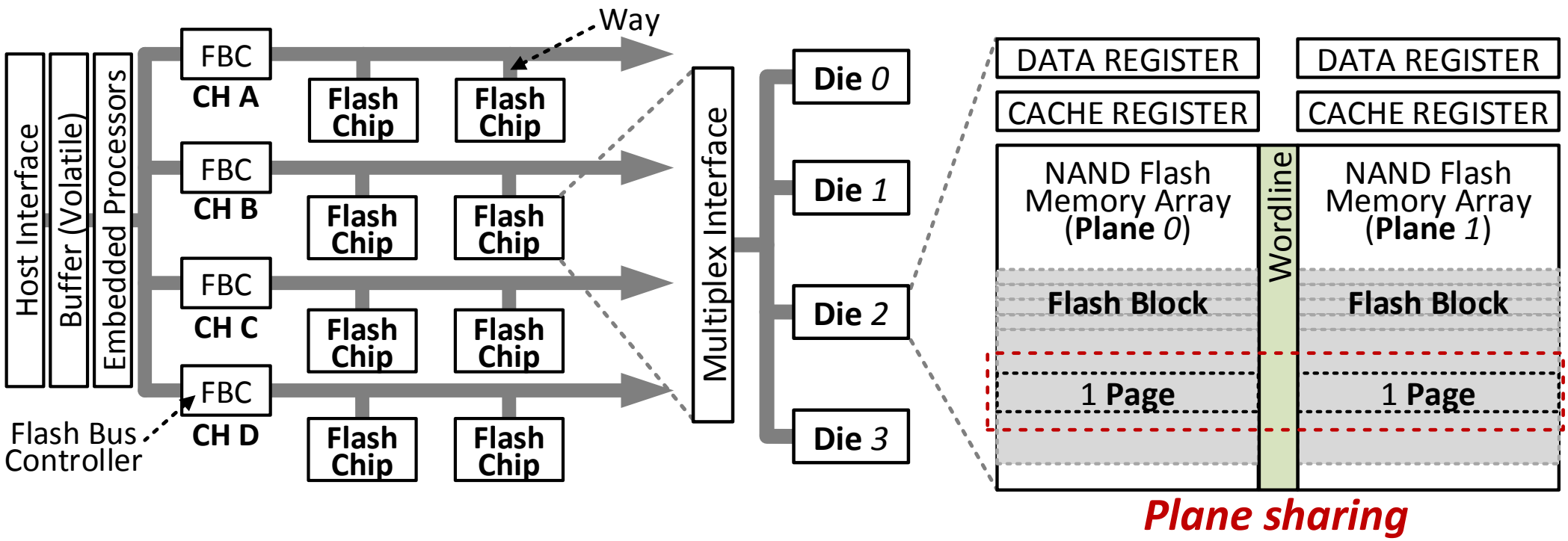
1) System-level parallel data access ***2) Flash-level parallel data access***



- A striped/pipelined request can be further interleaved within a chip

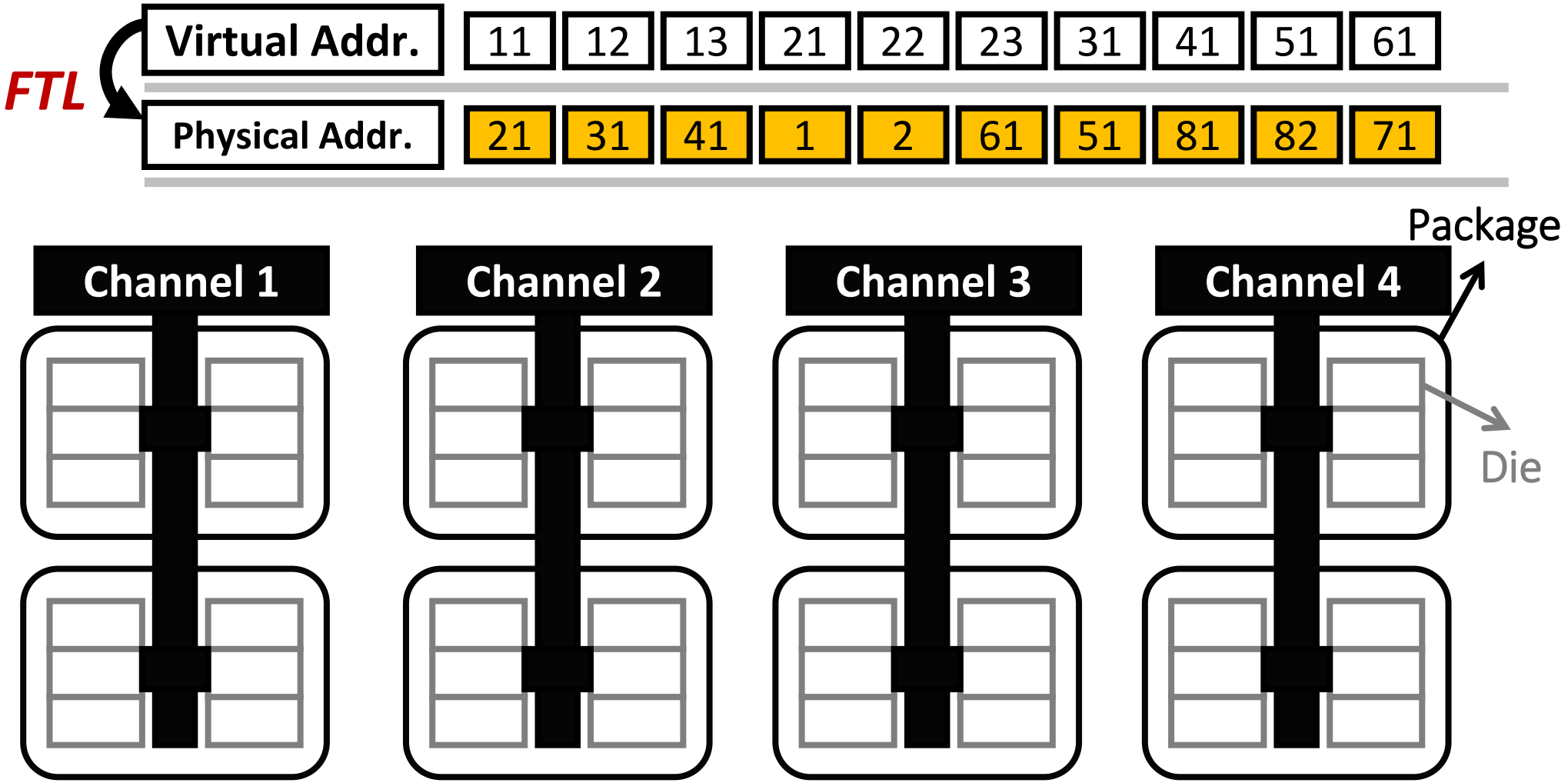
SSD Internal Parallelism

1) System-level parallel data access ***2) Flash-level parallel data access***



- Multiple planes simultaneously works using shared wordline(s)

SSD Internal Parallelism



Parallelism Overview

- Note that different vendors use different naming rules (interleaving, stripping, way, etc.)
- You need to understand four different level of parallelism based on the contexts
  - Plane sharing (=multiple-mode operation, two-plane operation, etc)
  - Die interleaving (=interleaved die operation, bank interleaving, etc)
  - Way pipelining (=package interleaving)

Outline

- Holistic Viewpoint (Overview)
- SSD Architecture
- Parallelism Overview
- Page Allocation Strategies
- Evaluation Studies for Parallelism
- Host Interface Overview
- Case Studies
  - Parallelism-Aware Host Interface I/O Scheduler
  - GC-Aware Host Interface I/O Scheduler
- Simulation Infrastructure (for your future research)

Software Stack (review) & 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

Hardware Abstraction Layer

Committing flash transaction to underlying flash memory chips

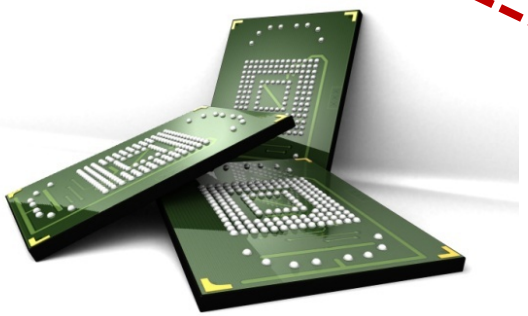
HIL

FTL

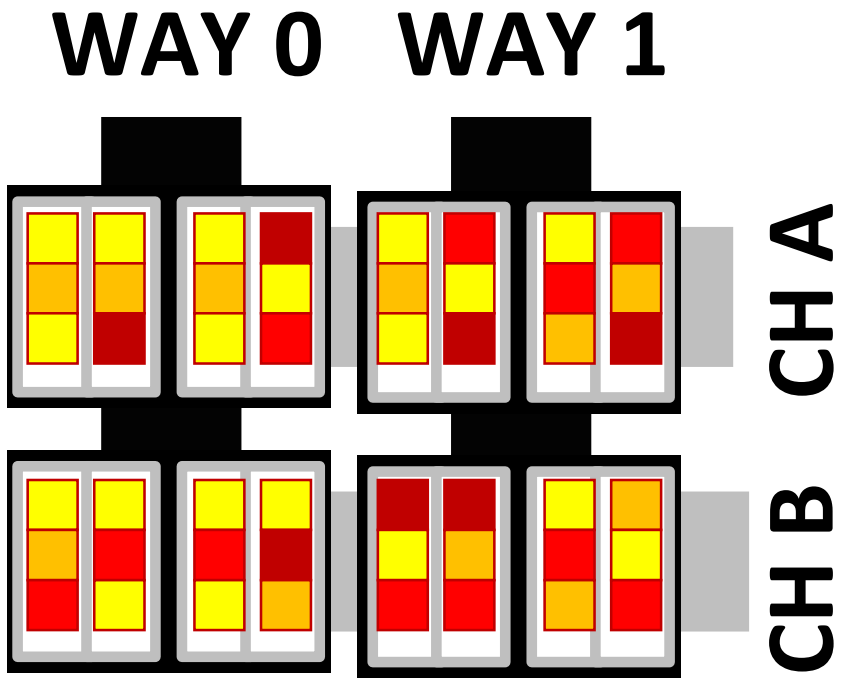
HAL

24

Page Allocation Strategies



[Image:micron.com]



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

CPWD

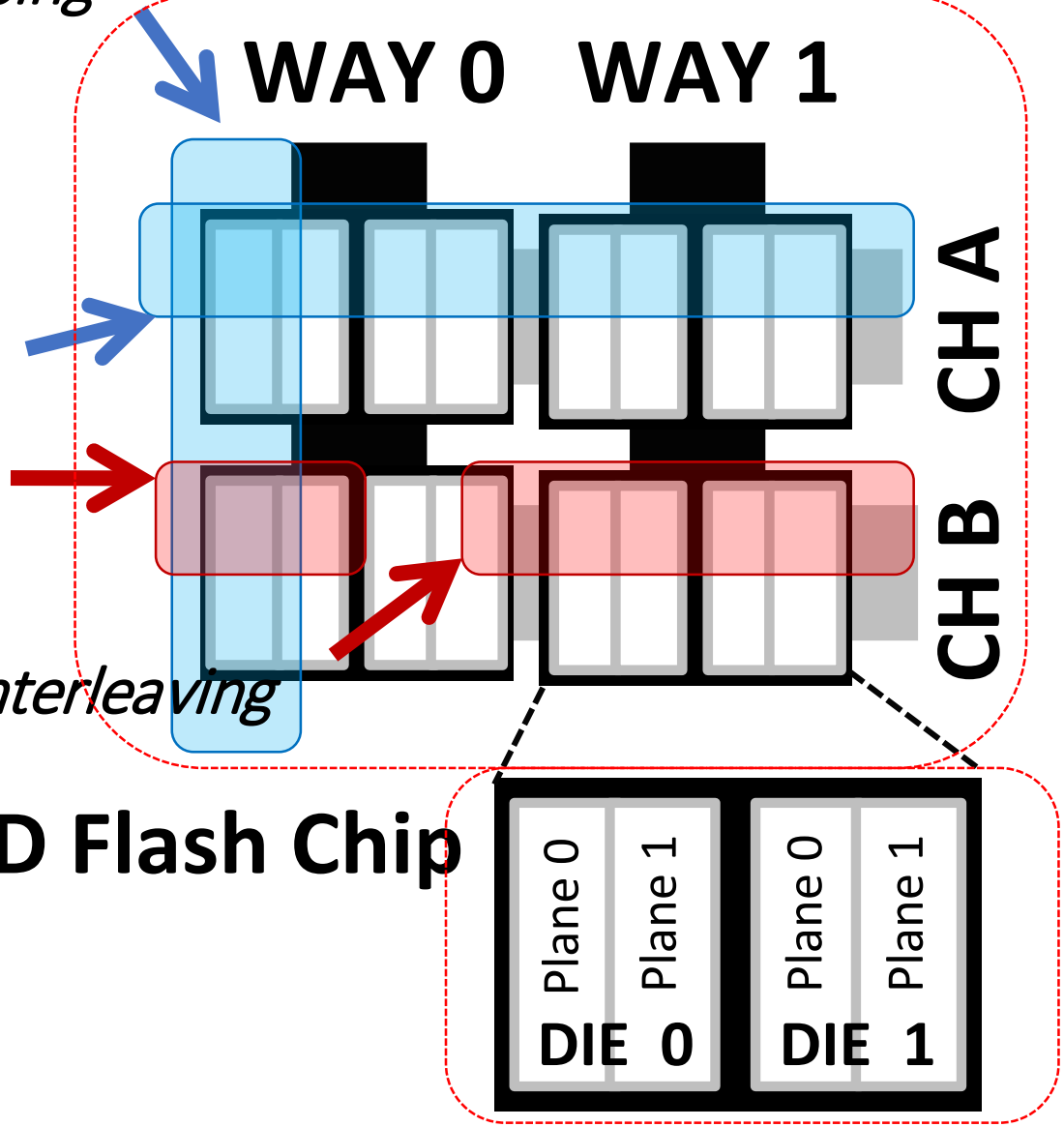
CWPD

Channel striping

Way Pipelining

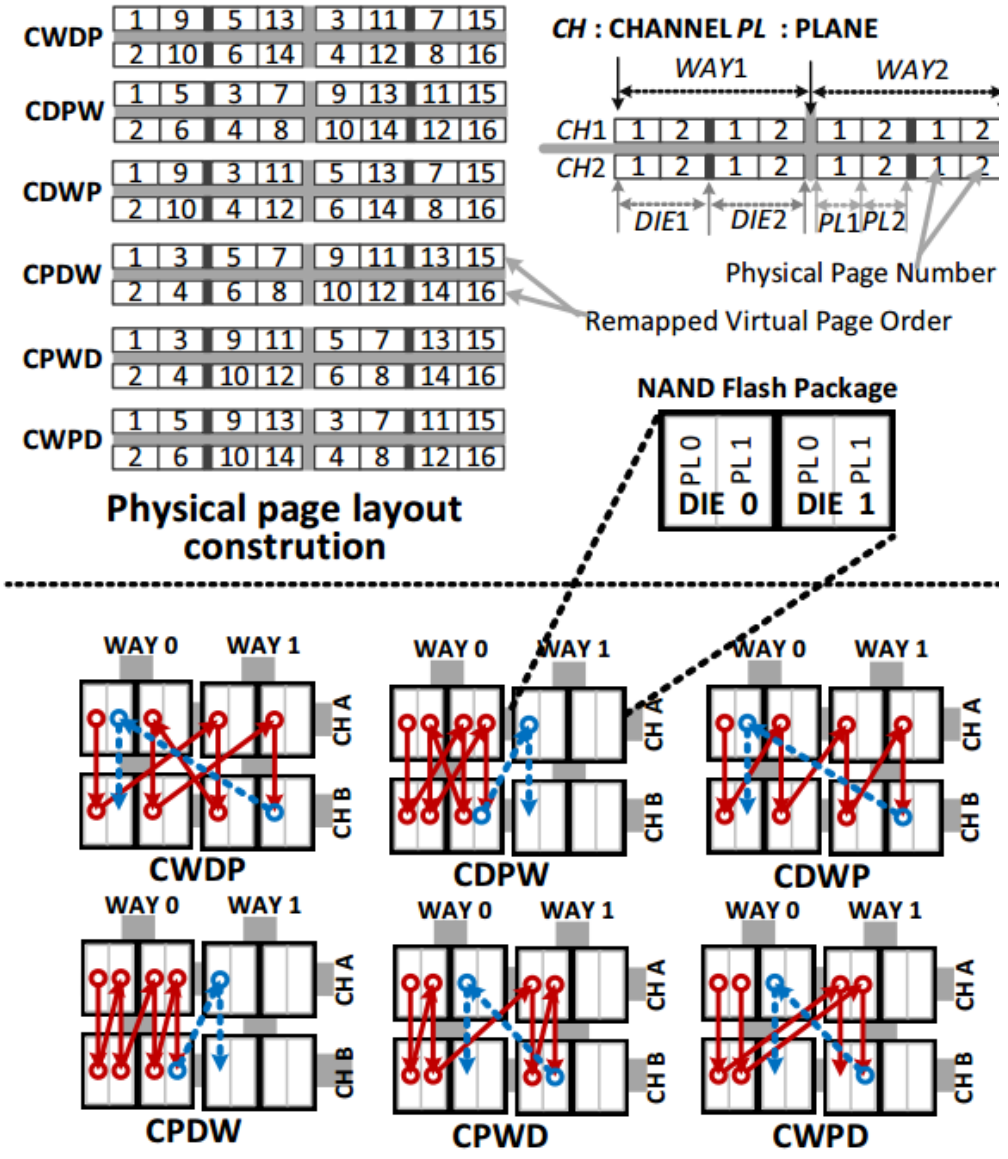
Plane Sharing

Die Interleaving



Channel-first Page Allocation

- This page allocation strategies give priority to the order of channel, way, die and plane
- Some channel first page allocation strategies introduce low flash-level locality



Way-first Page Allocation

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

WCPD

WDCP

WDPC

WPCD

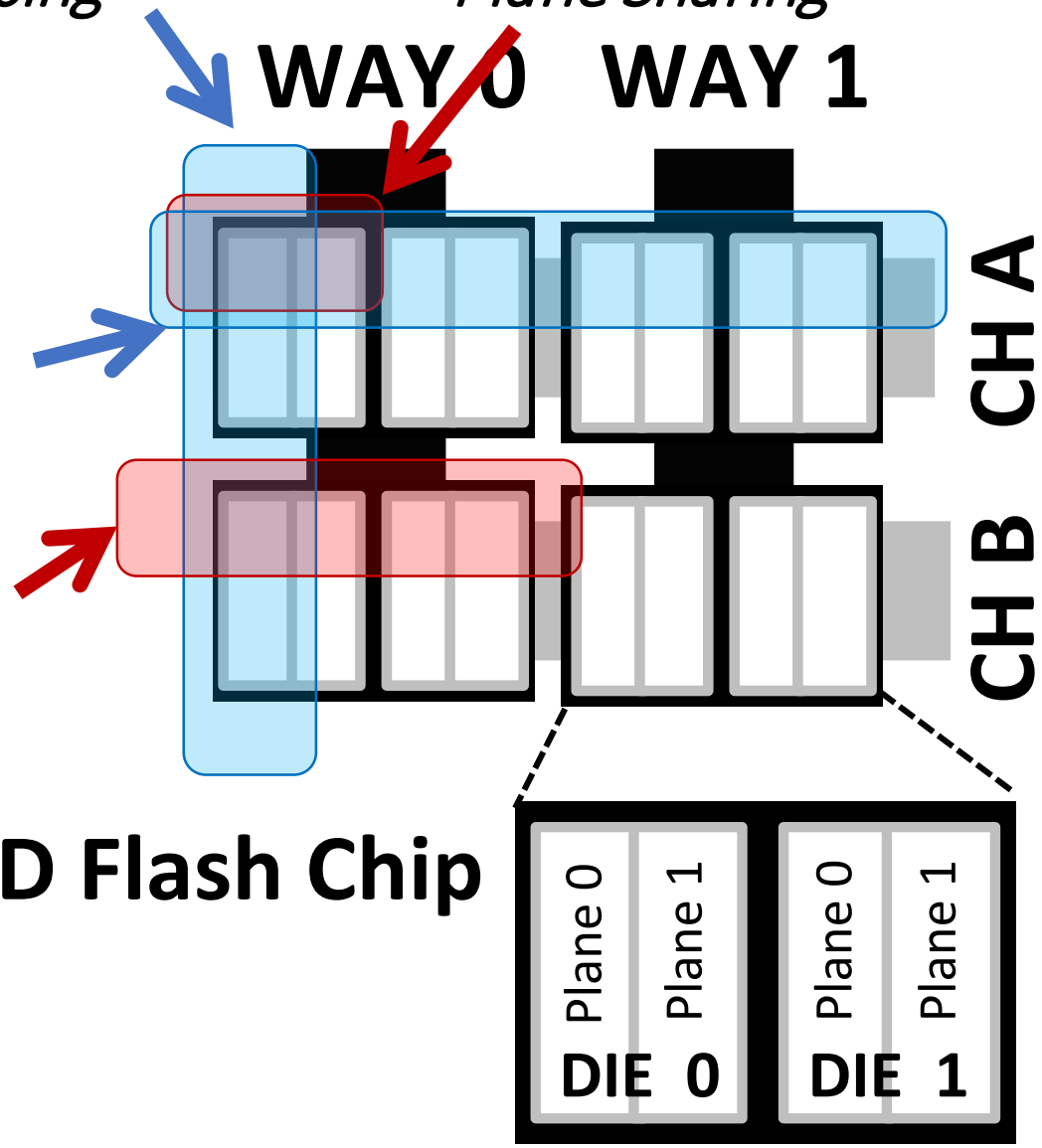
WPDC

Channel striping

Plane Sharing

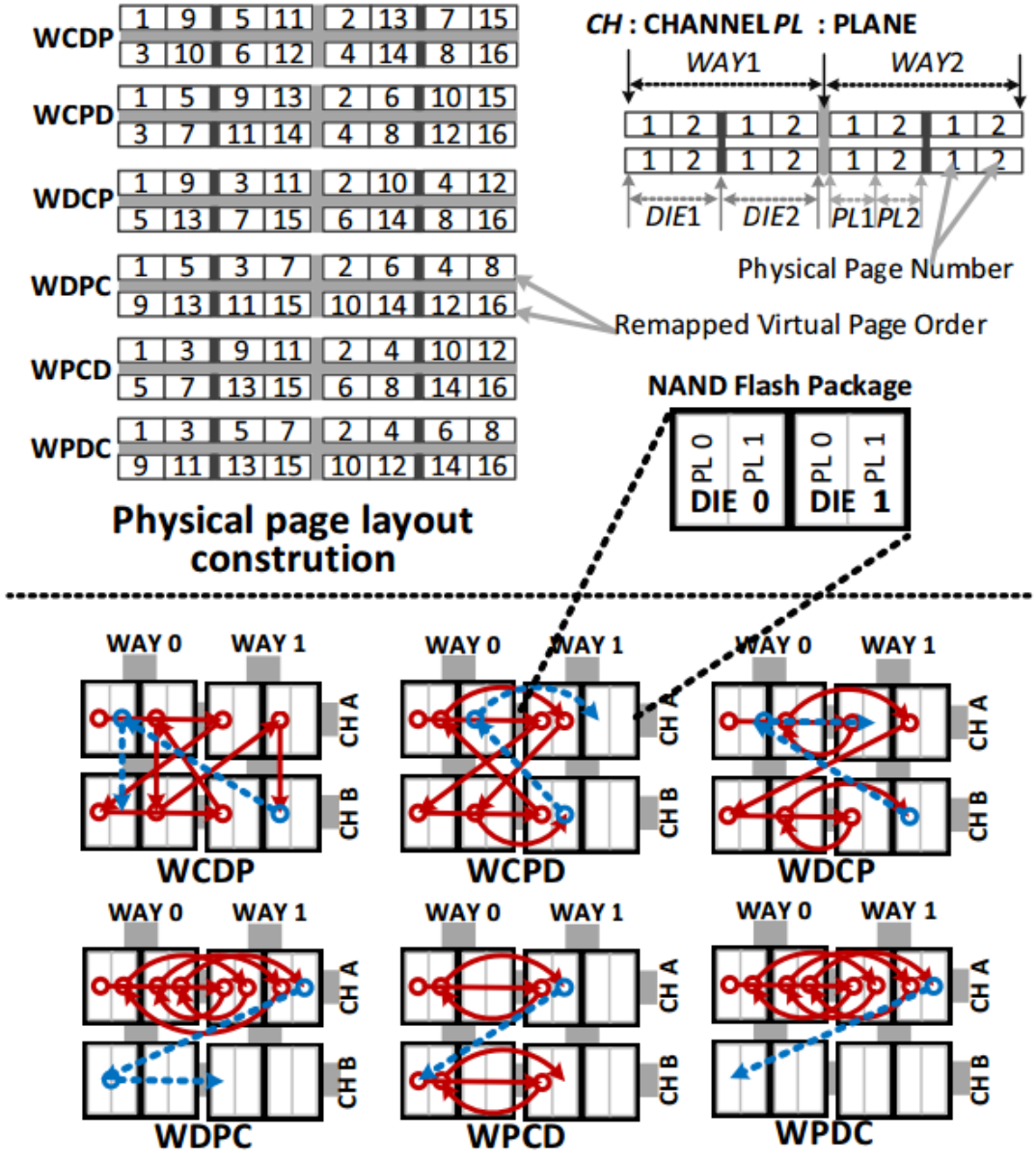
Way Pipelining

Die Interleaving



Way-first Page Allocation

- This allocation assigns
  1. the way resources in a channel
  2. stripe all the requests over multiple ways
  3. interleave the flash-level resources.
- Although it allocates the system-level resource first, some favor flash-level resources



Die-first Page Allocation

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

DCPW

DCWP

DPCW

DWCP

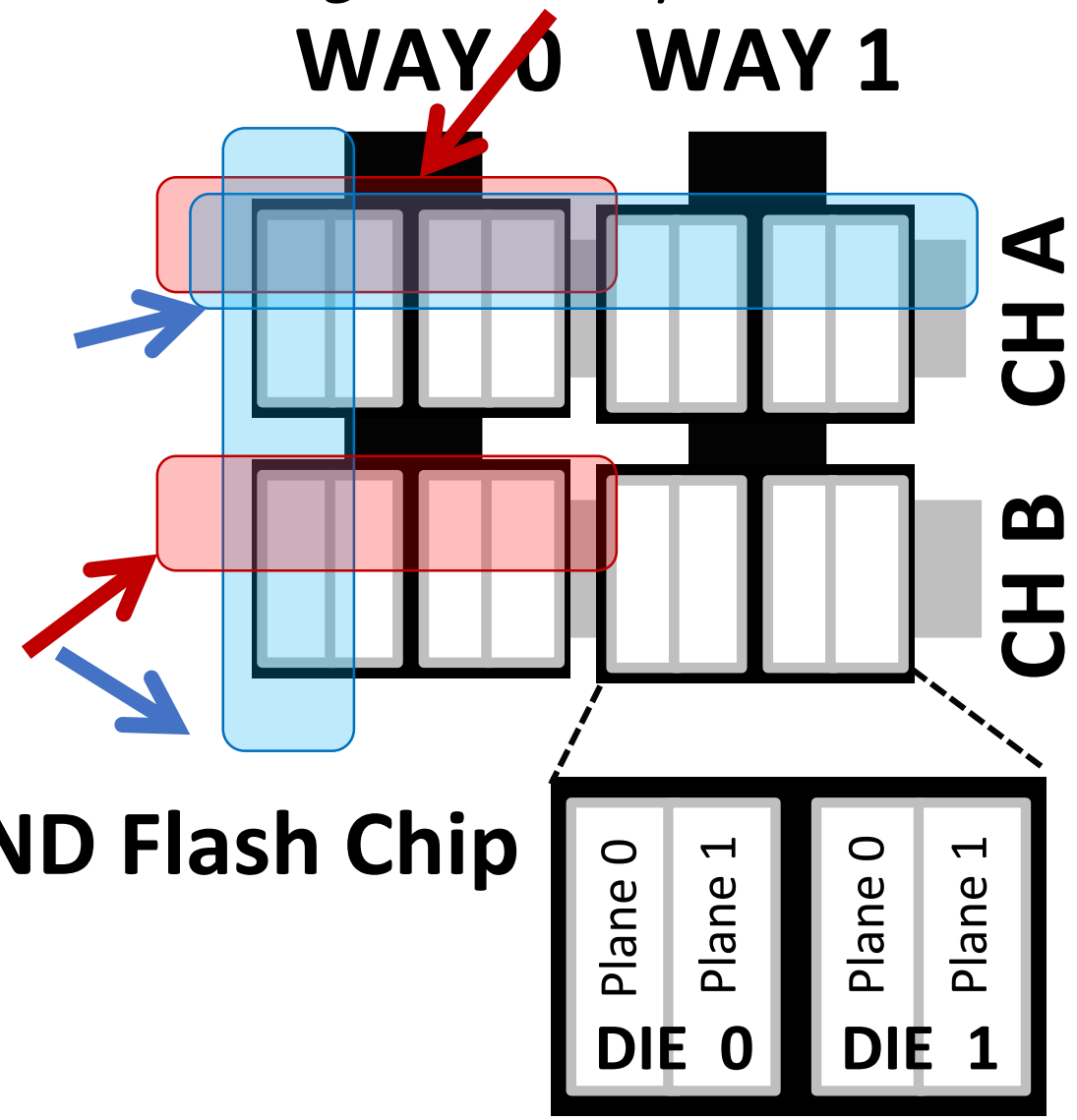
DWPC

Die Interleaving with Multiplane

Way Pipelining

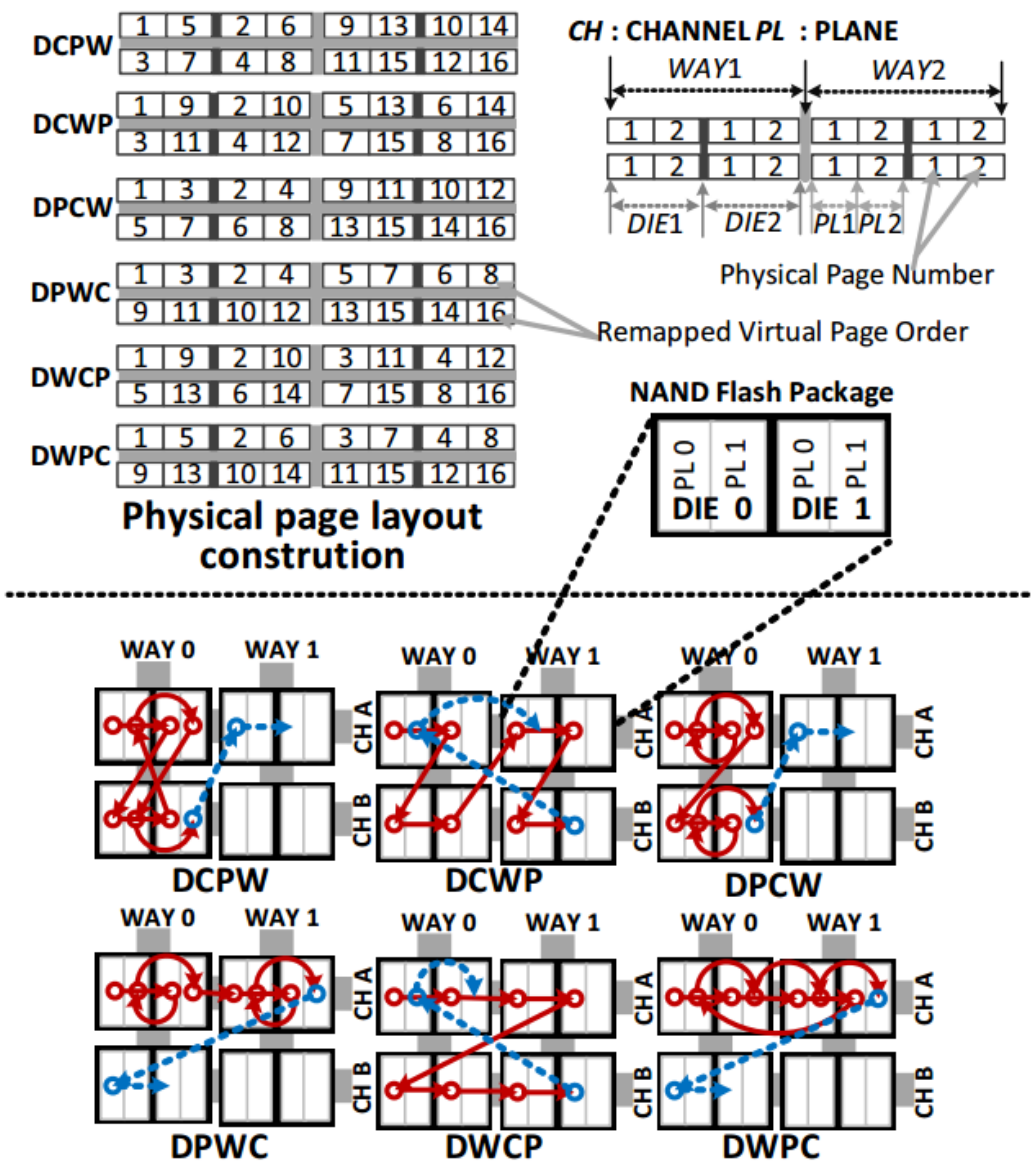
Channel striping

Die Interleaving



Die-first Page Allocation

- The die-first page allocation schemes favor the exploitation of the die-interleaving method
- It could also accommodate system-level resources instead of the flash-level resources based on the access patterns (DCWP/DWCP).



Plane-first Page Allocation

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

PCWD

PCWD

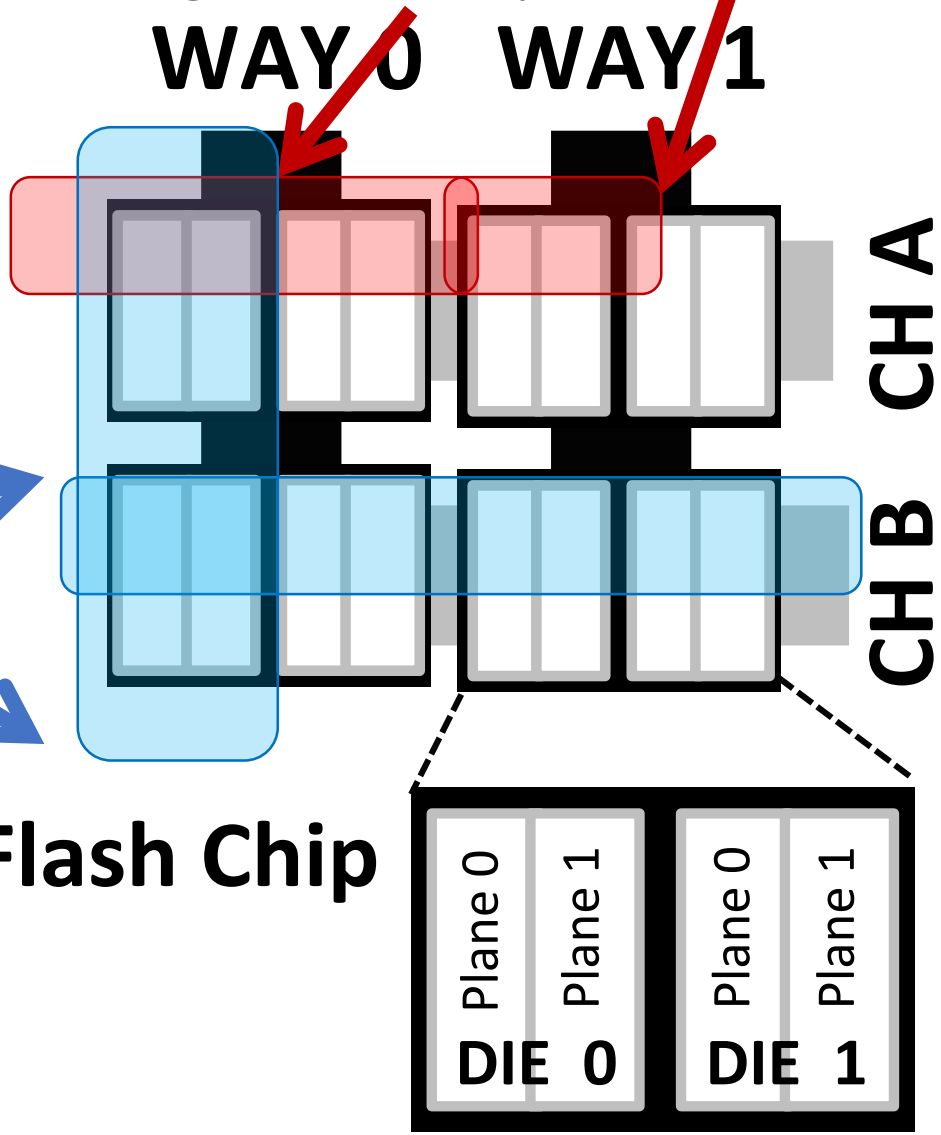
PDCW

PDWC

PWDC

Plane Sharing

Die Interleaving with Multiplane



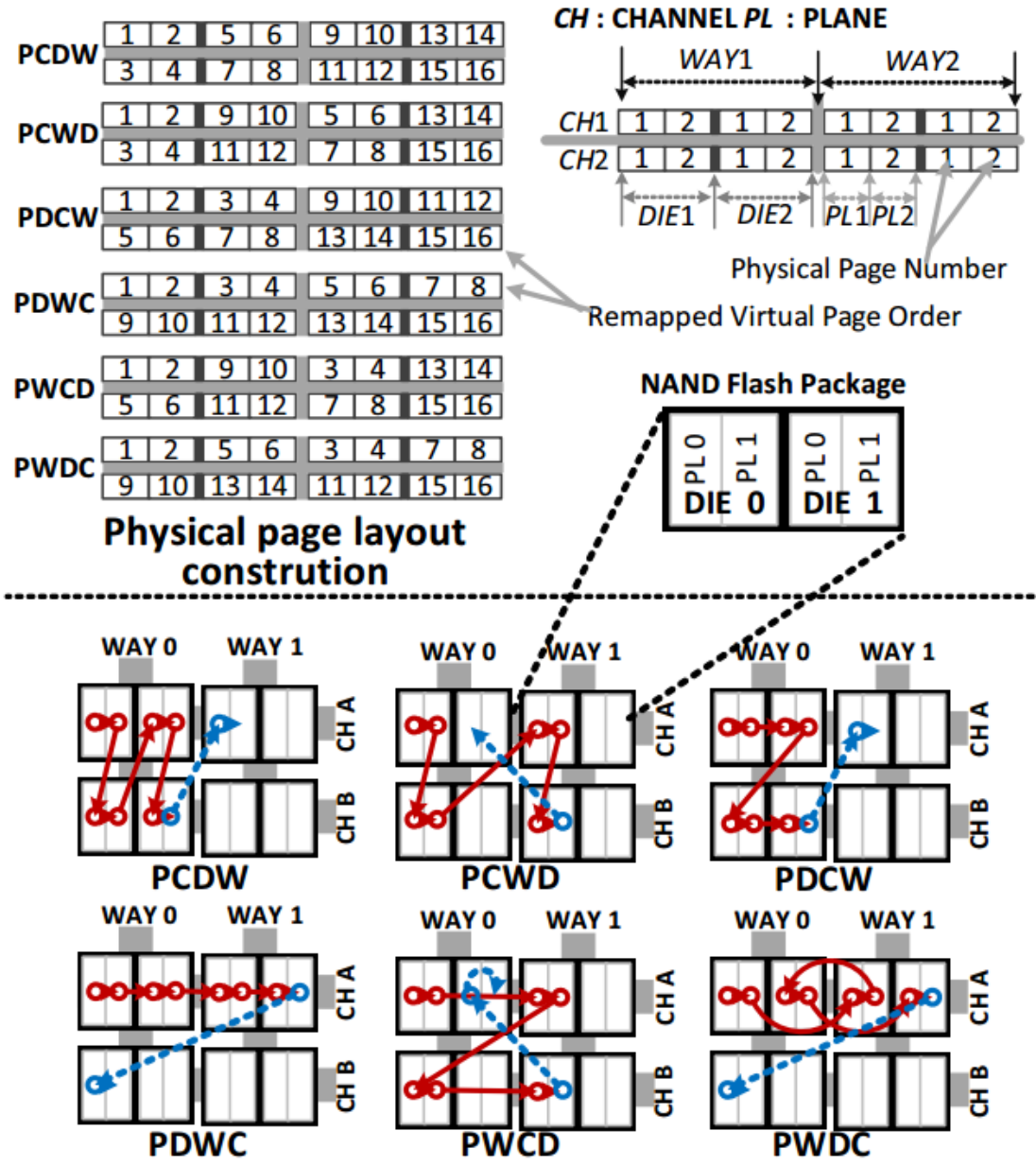
Way Pipelining

Channel striping

NAND Flash Chip

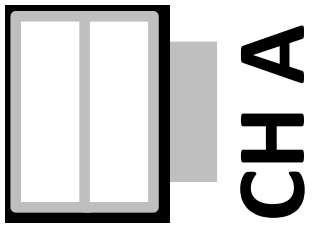
Plane-first Page Allocation

- It parallelizes data accesses with plane sharing, which can in turn improve the storage throughput
- An excellent option for realizing the benefits of both inter- and intra-request parallelisms



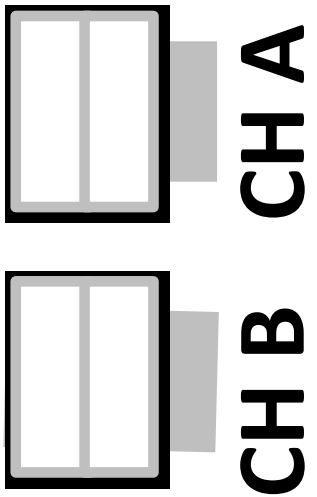
Channel-first vs plane-first

Channel-first page allocation



Total latency : 200 us

Plane-first page allocation

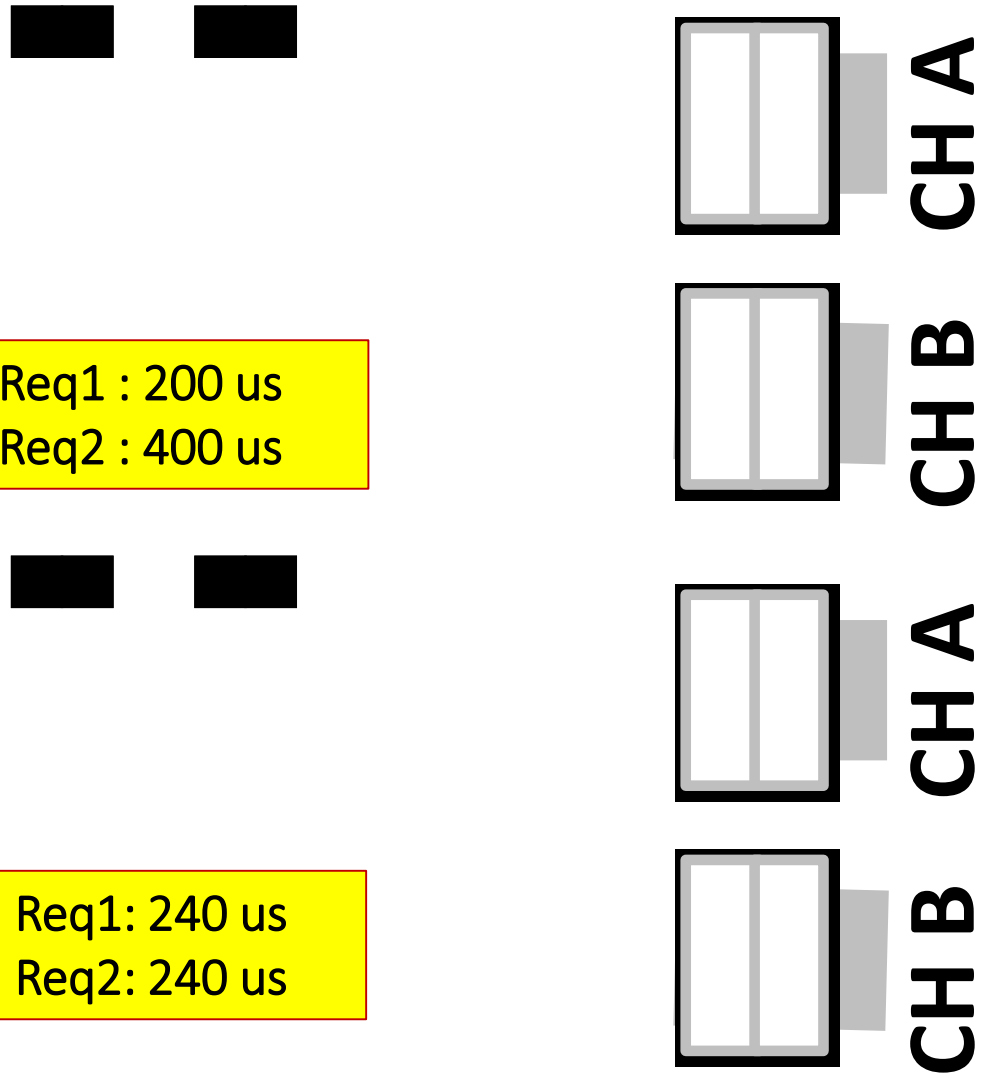


Total latency : 240 us

Assumption in this example
Legacy : 200 us (1 page)
Plane sharing : 240 us (2 pages)

Channel-first vs plane-first

Channel-first page allocation



Req1 : 200 us
Req2 : 400 us

Plane-first page allocation in favor of way pipelining

Req1: 240 us
Req2: 240 us

Assumption in this example
Legacy : 200 us (1 page)
Plane sharing : 240 us (2 pages)

Outline

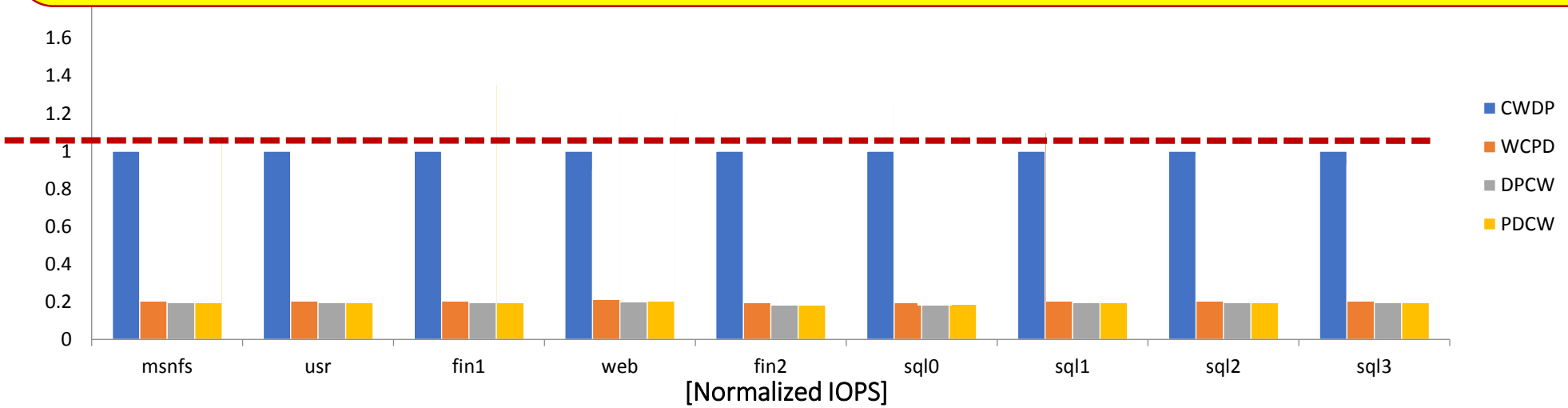
- Holistic Viewpoint (Overview)
- SSD Architecture
- Parallelism Overview
- Page Allocation Strategies
- Evaluation Studies for Parallelism
- Host Interface Overview
- Case Studies
  - Parallelism-Aware Host Interface I/O Scheduler
  - GC-Aware Host Interface I/O Scheduler
- Simulation Infrastructure (for your future research)

SSD Setup

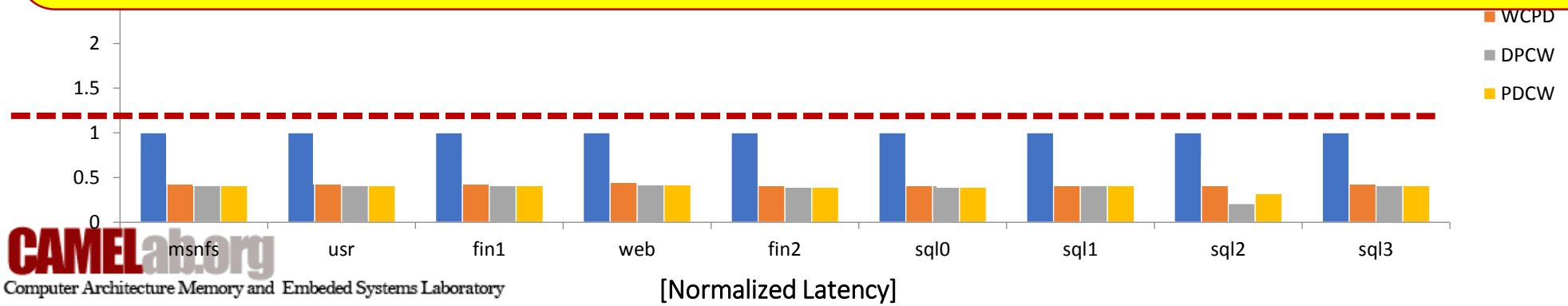
- NAND Flash Chip
  - Fine-grained NAND command
  - Advanced commands
  - Strong address constraints
  - Intrinsic latency variation
- SSD Framework
  - 8 channels, 8 flash per channel (64 total)
  - Dual-die package format, 32 entry queue
  - A page-level mapping and greedy garbage collection algorithm

Performance Comparison

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

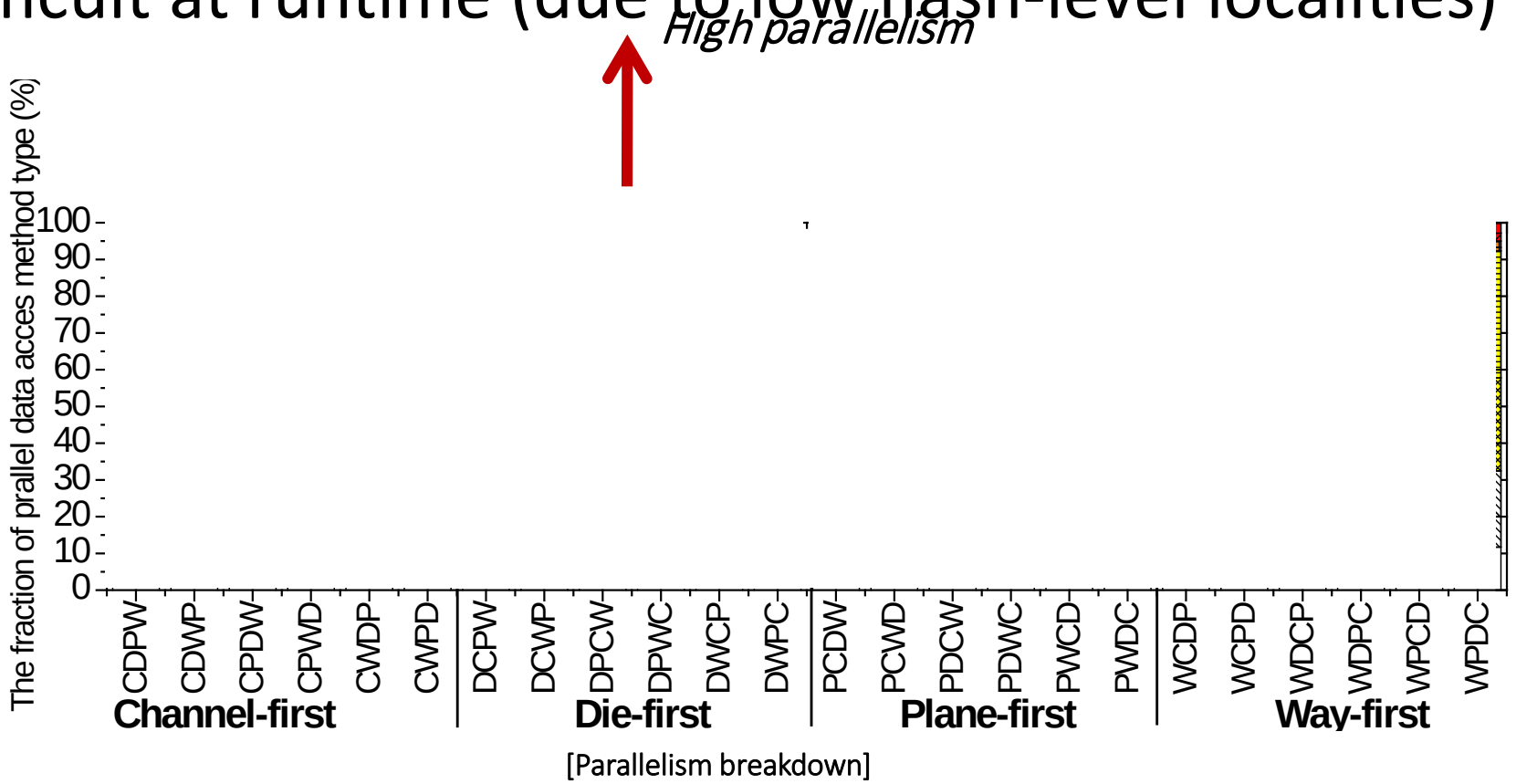


Channel-first palloc provide shorter latencies than flash-level resource first pallocs

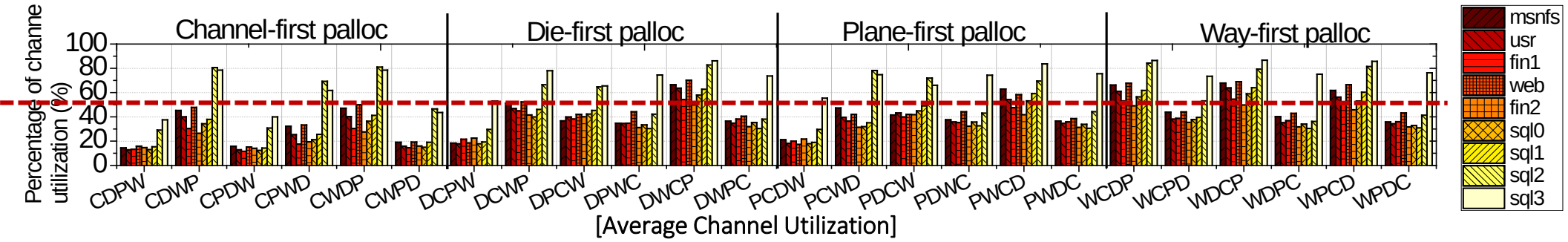


Parallelism Breakdown

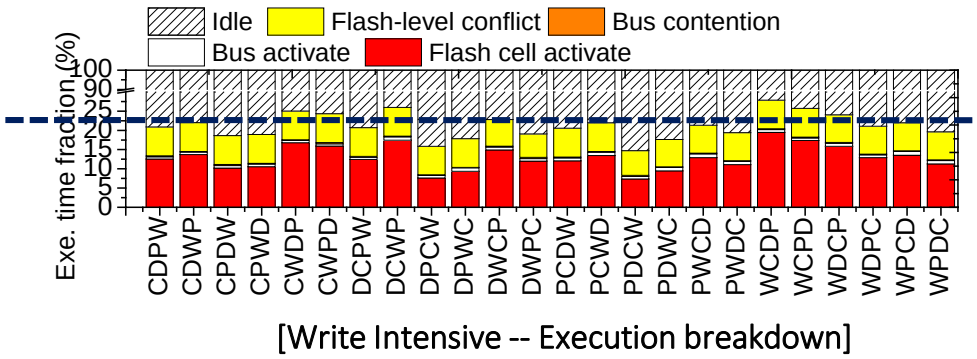
- Low flash-level parallelism is observed under palloc schemes in favor of channel
- They render advanced flash command compositions difficult at runtime (due to low flash-level localities)

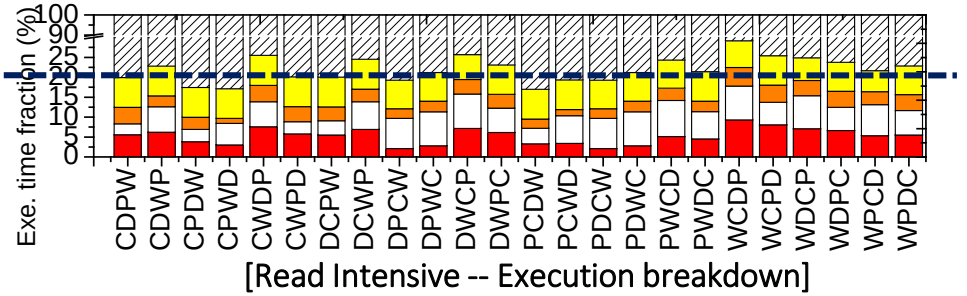


Resource Utilization



- **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





Optimization Point