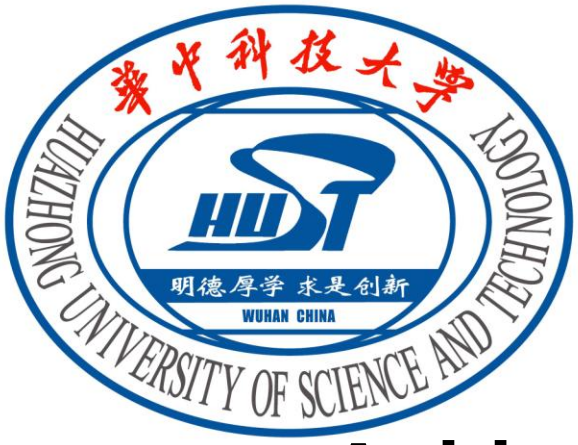


Achieving Page-Mapping FTL Performance at Block-Mapping FTL Cost by Hiding Address Translation (HAT)

Yang Hu
yanghu@foxmail.com
Huazhong University of Sci. & Tech. China

Outline

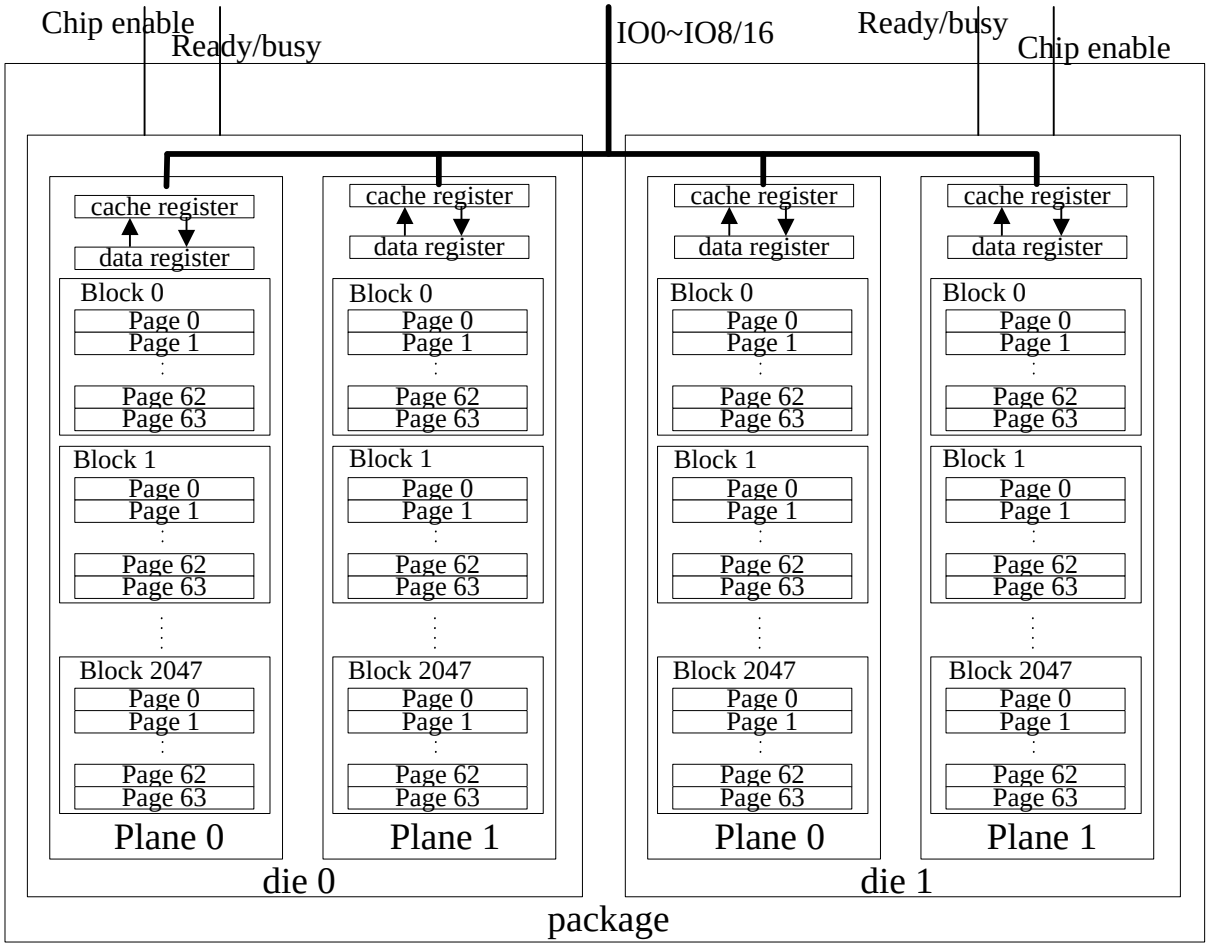
- Background
  - ✓ Nand-flash architecture
  - ✓ Nand-flash limitations
  - ✓ FTLs
- HAT Implementation
- Performance Evaluation
  - ✓ Experiment setup
  - ✓ Average response time
  - ✓ IOPS
  - ✓ Microscopic analysis
  - ✓ Energy consumption

Background

The advantages of Flash

1. Non-volatile
2. Low energy consumption
3. High performance
4. Anti-vibration
5. ……

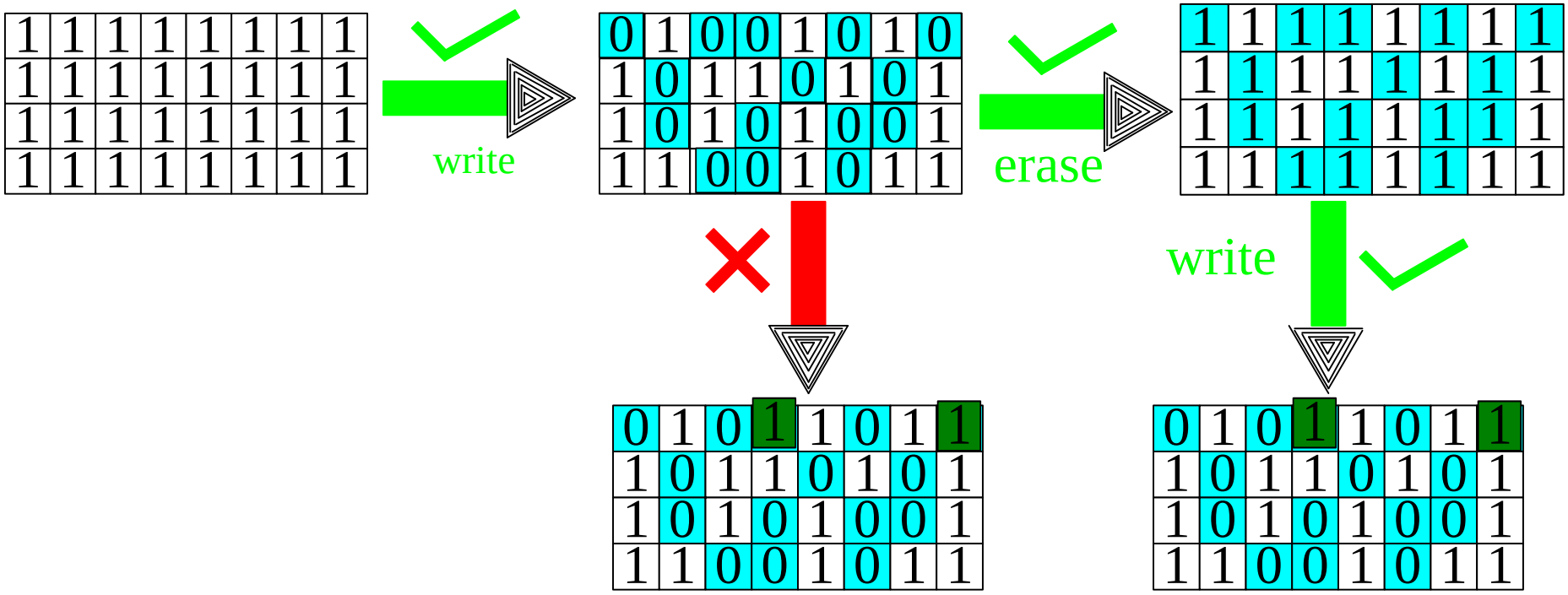
Flash architecture



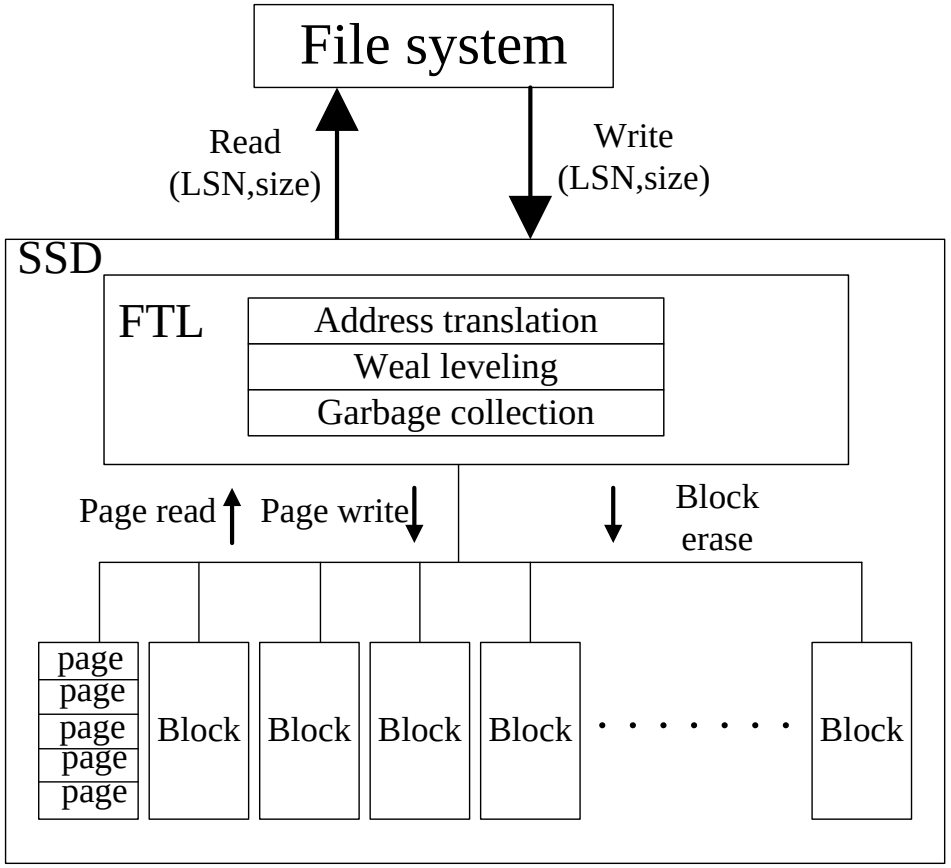
Limitations of Flash Memory

1. Write-after-erase
2. Lifetime (erase cycle)

Write-after-erase



Flash Translation Layer (FTL)



Software middle layer

- Address translation
- Wear-leveling
- Garbage collection

Address translation :

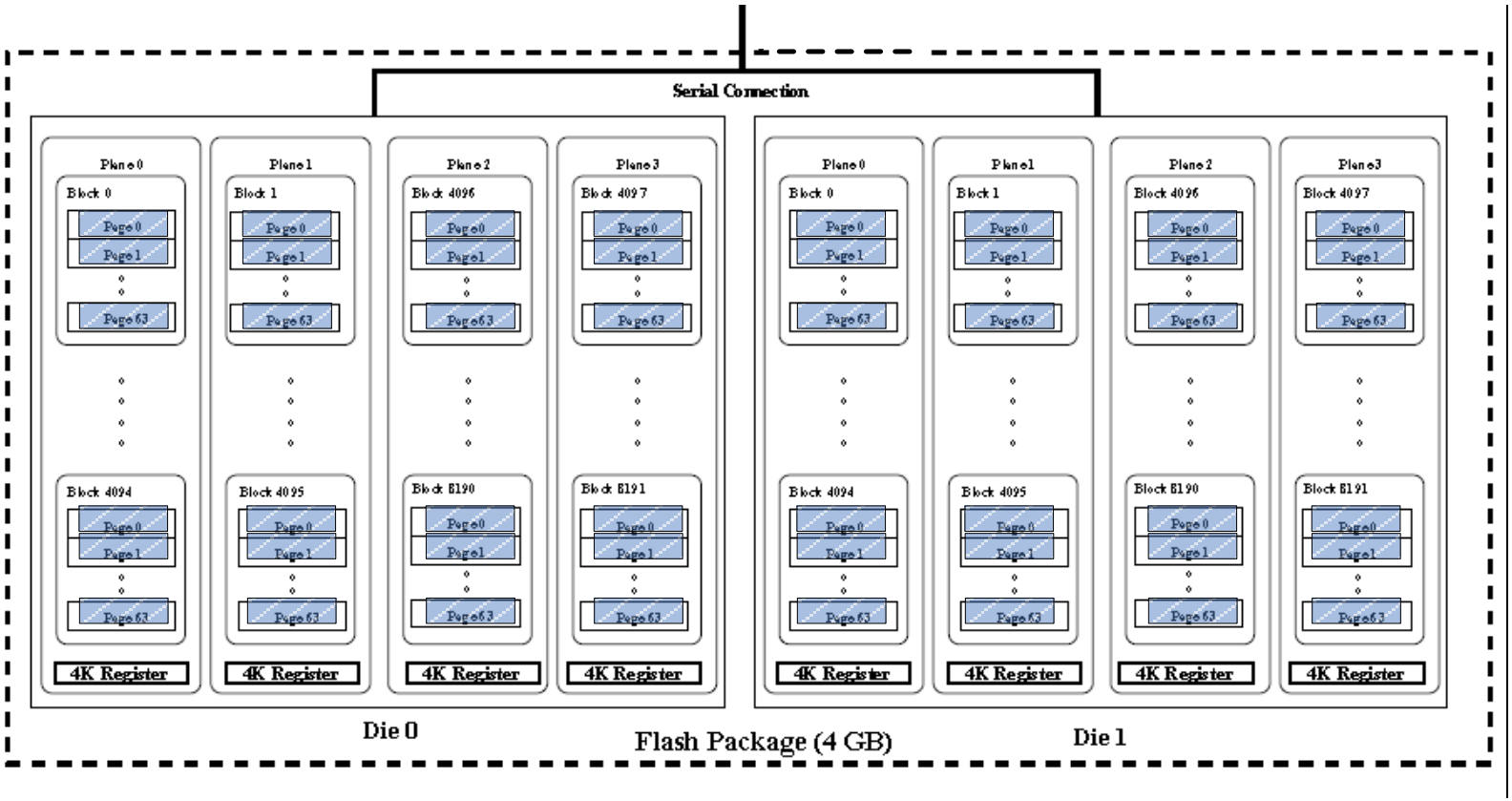
Logical address → Physical address

FTLs

- Page-mapping FTL
- Block-mapping FTL
- Hybrid FTL

Page-mapping FTL

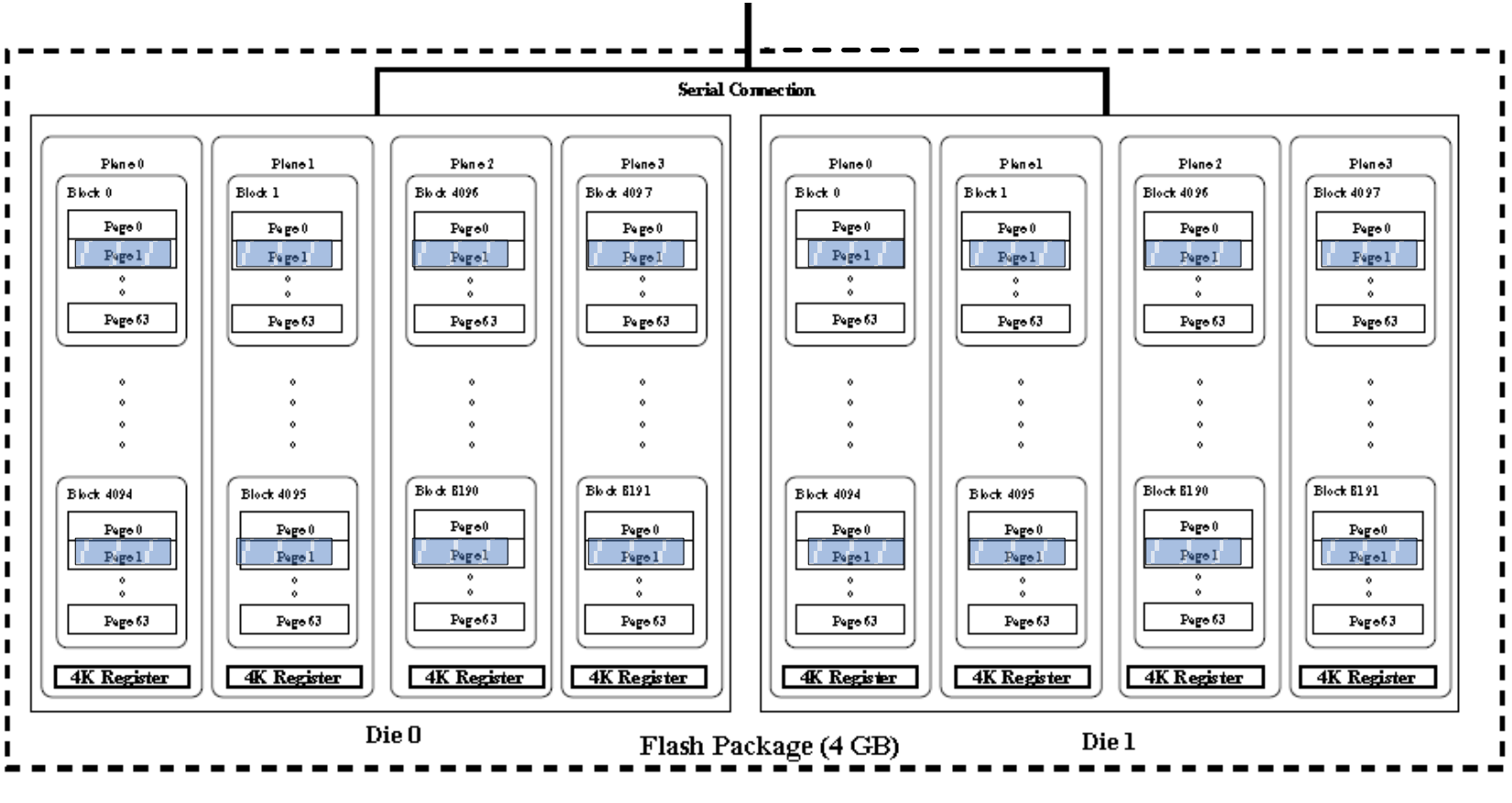
(Write,LPN=65)



Page-mapping FTL can map a logical page to any physical page

Block-mapping FTL

(Write,LPN=65)

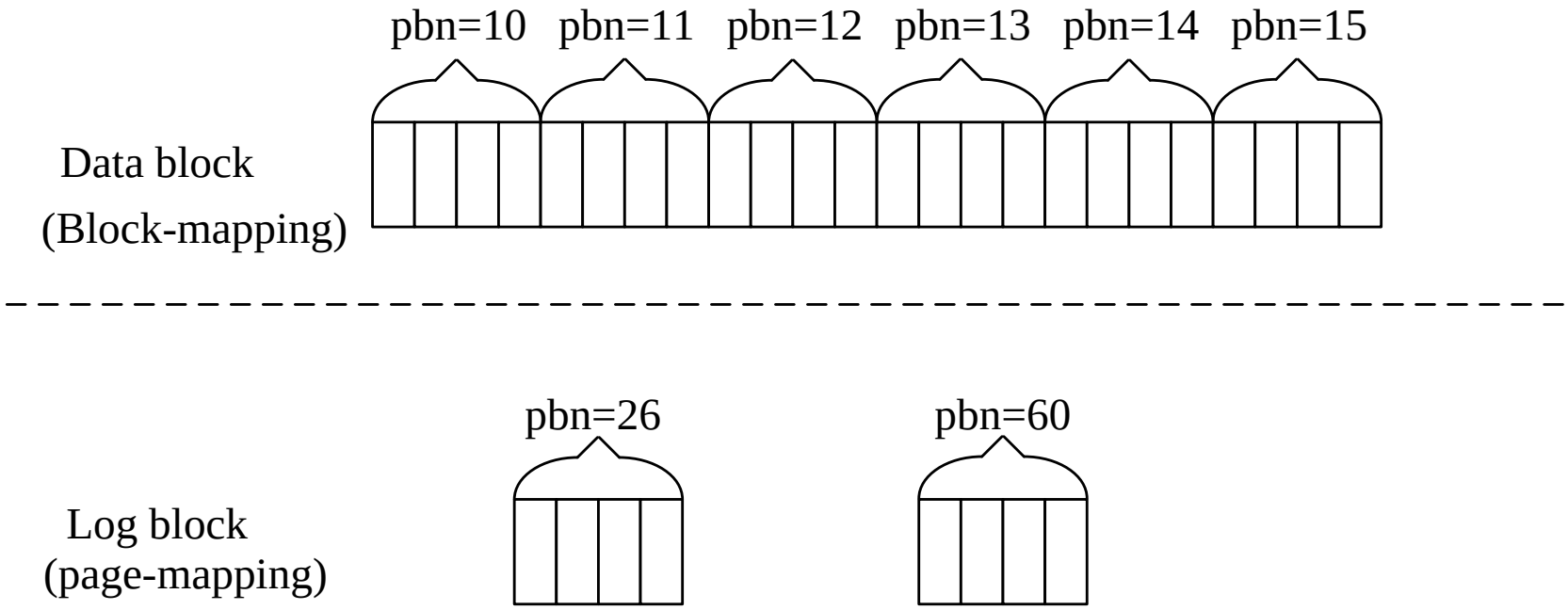


Block-mapping FTL can only map a logical page to a fixed offset of a block

Performance vs. RAM space

	Page mapping	Block mapping
Performance	high	low
RAM space	high	low

Hybrid FTL


pbn=10
pbn=11
pbn=12
pbn=13
pbn=14
pbn=15
Data block
(Block-mapping)
pbn=26
pbn=60
Log block
(page-mapping)

DFTL

Basic idea:

- Based on page mapping FTL
- Utilize the temporal locality of workloads
- Popular mapping information stored in RAM
- Complete mapping information stored in Flash

Missed mapping information in RAM

→

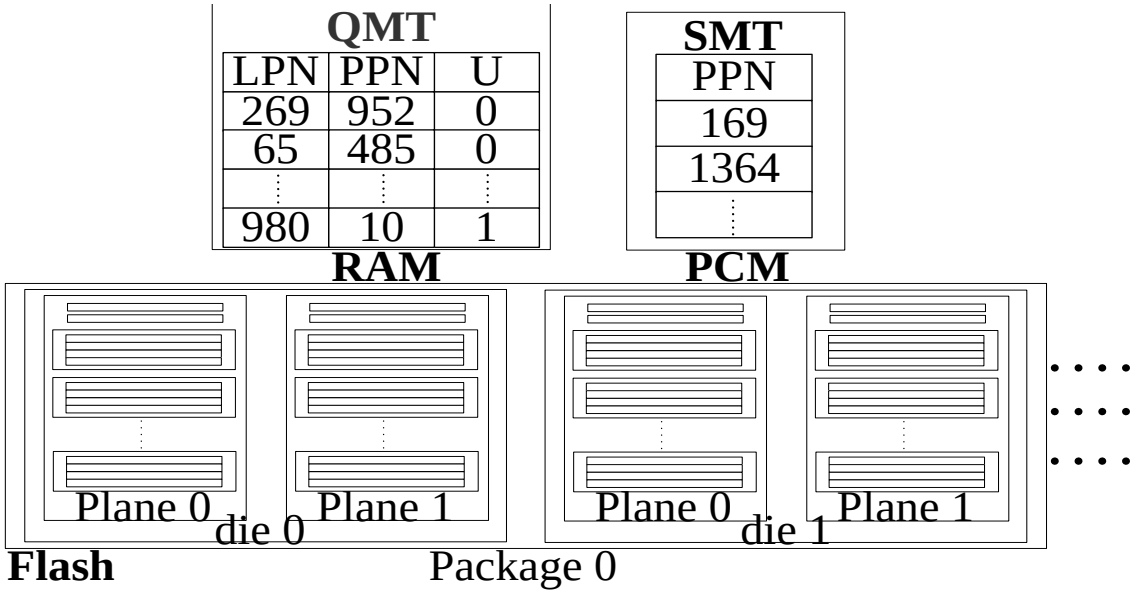
Extra read or write of mapping information from or to flash

→

Performance degradation (extra erasures and garbage collections)

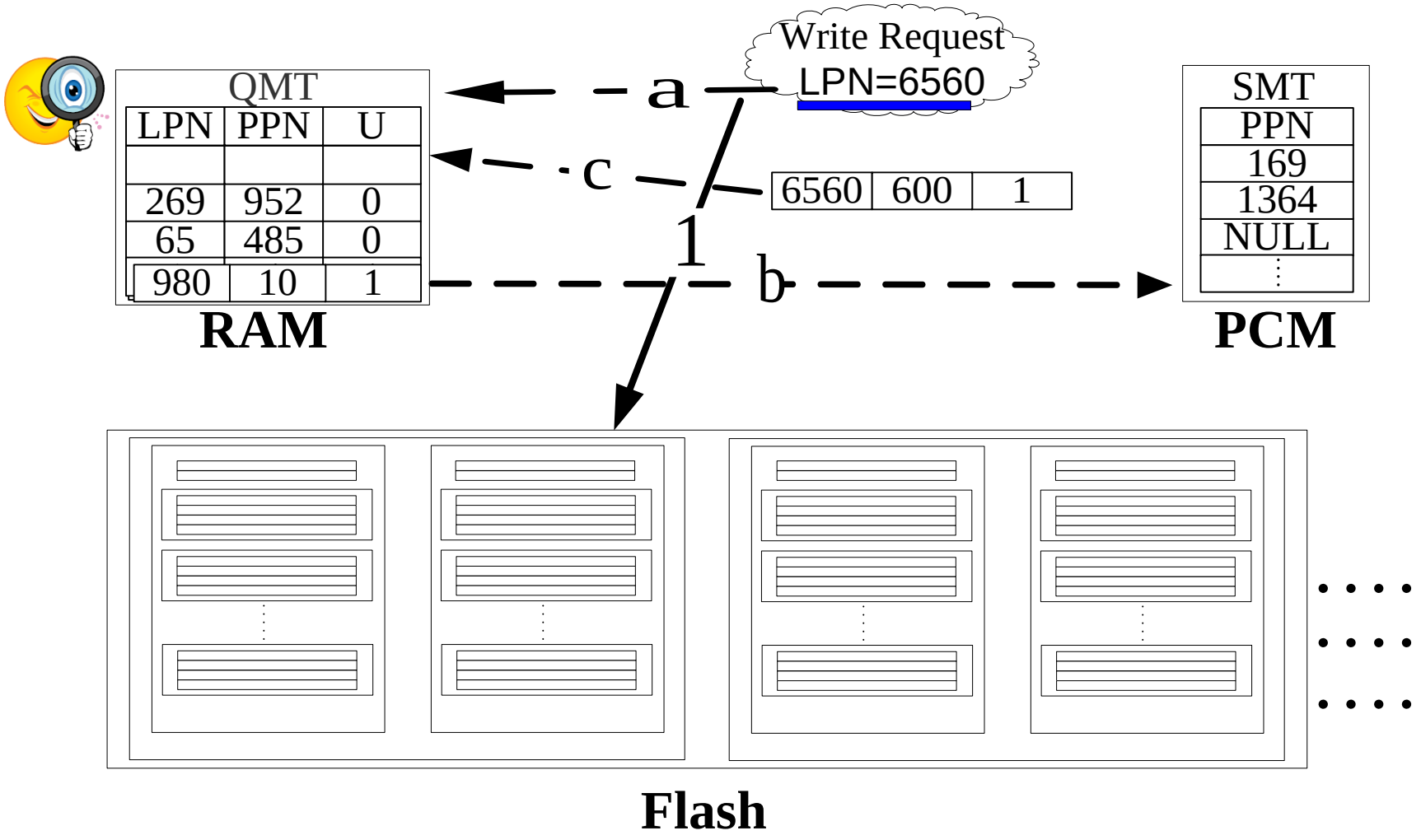
Accesses to data and mapping info in flash compete for a shared critical path!

HAT – A novel FTL that hides the address translation overhead



- Based on pure page mapping FTL
- Entire mapping table is stored in a PCM
- Popular mapping information stored in a small RAM space (CPU cache)
- Two separate access paths for actual data and mapping information

Write request on a QMT miss



Animation

QMT

LPN	PPN	U
269	952	0
65	485	0
980	10	1

RAM

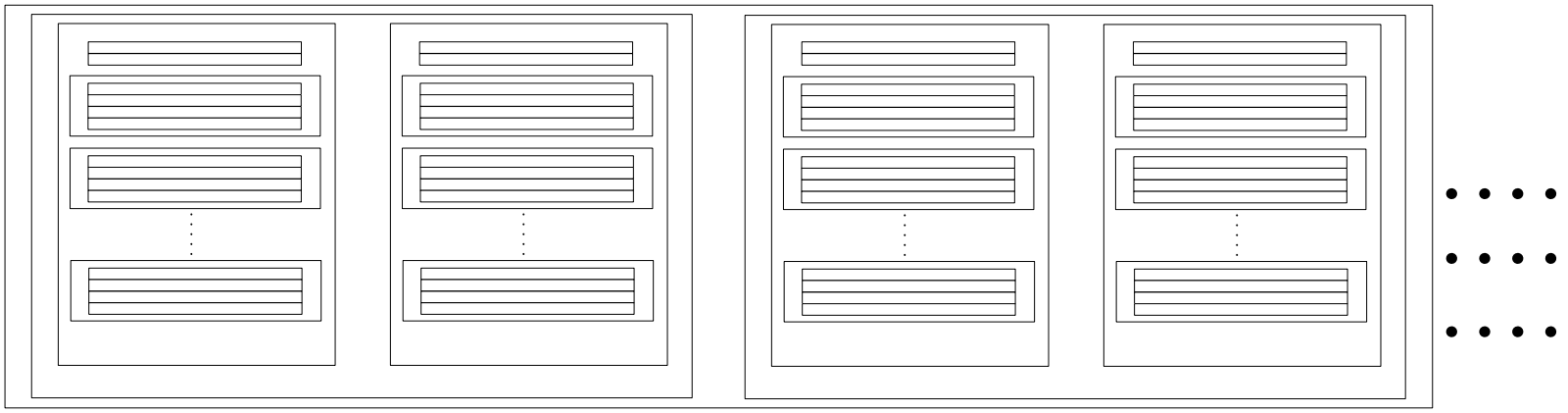
Write Request
LPN=6560

6560	600	1

SMT

PPN
169
1364
NULL
⋮

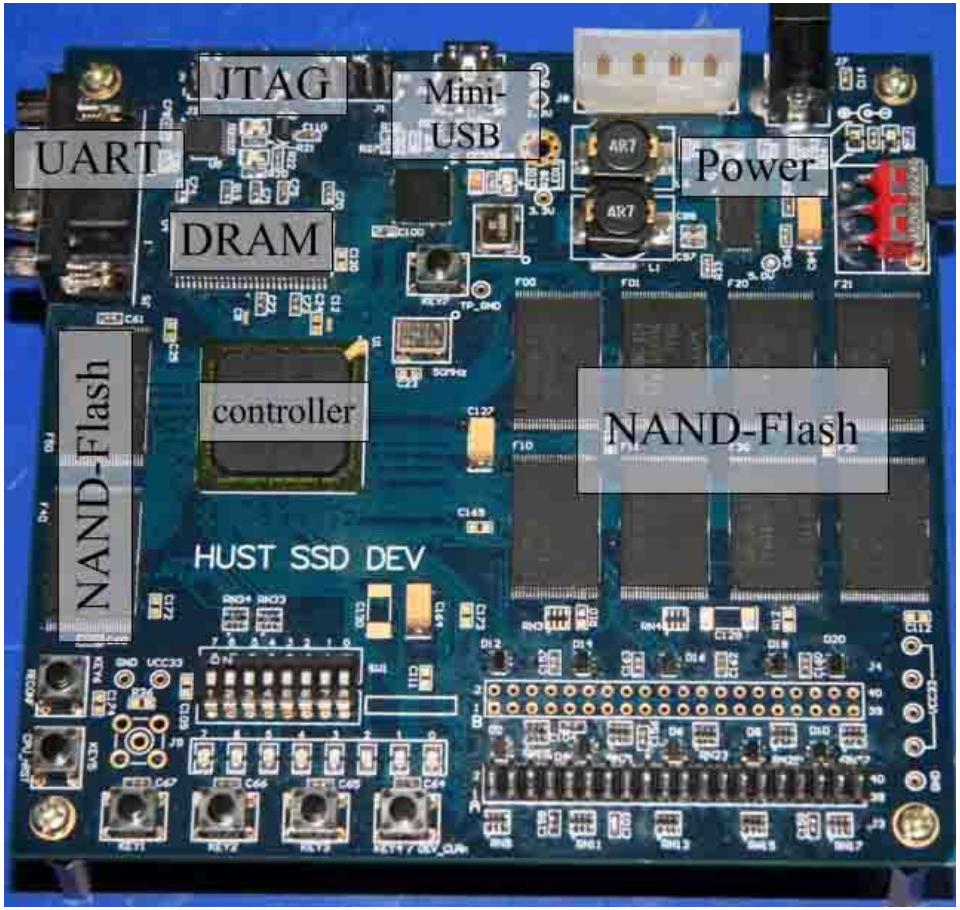
PCM



Flash

Experimental setup

- We implement a simulator for SSD—SSDsim



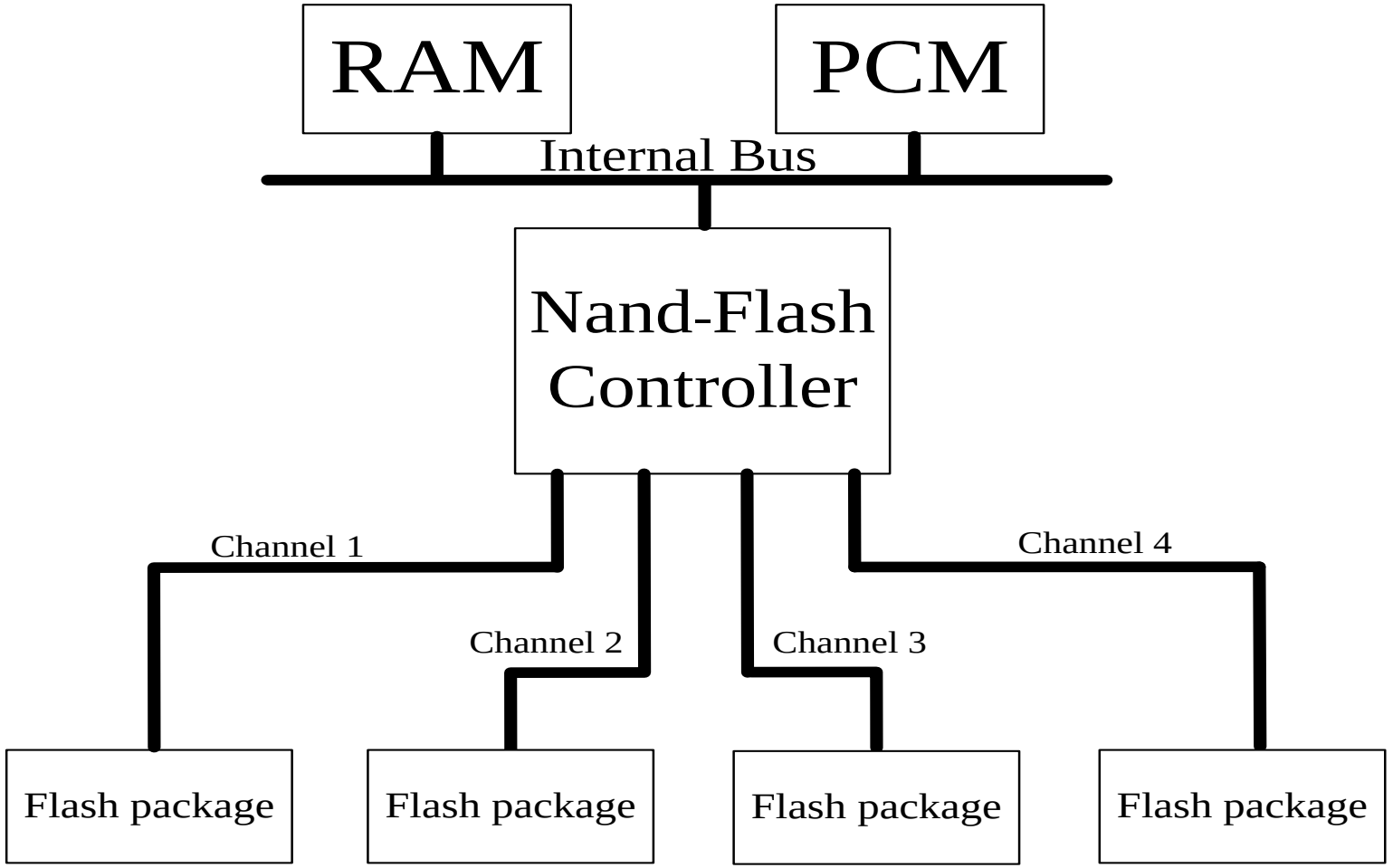
Realistic traces

- Financial1.spc
- Financial2.spc
- Websearch1.spc
- Websearch2.spc
- Websearch3.spc
- Openmail

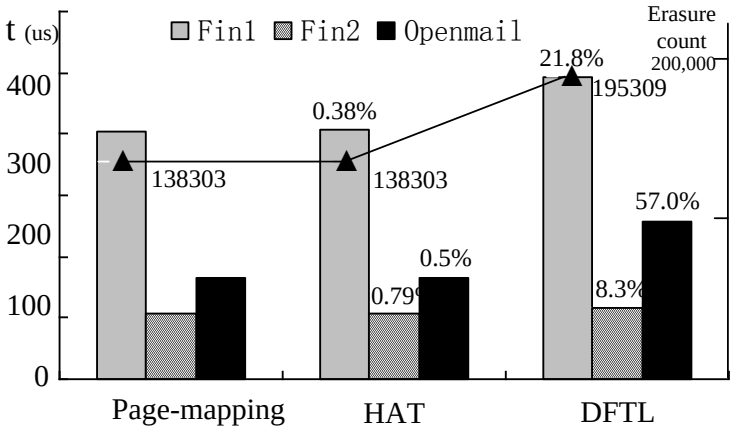
Configuration parameters of SSDsim

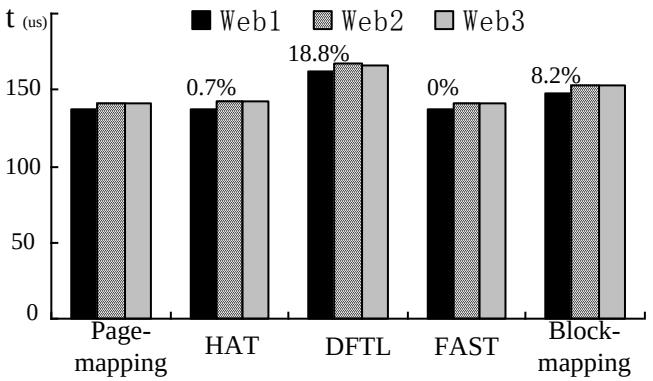
Organization		**Timing characteristics** (us)		**Electrical characteristics** (mA, V)	
Channel	4	t_{BERS}	1500	DRAM r/w	125
Package	1/channel	t_{PROG}	200	refresh	3
Die	4/package	t_R	20	Flash r/w/e	25
Plane	4/die	t_{WC}	0.025	PCM read	8
Block	2048/plane	t_{RC}	0.025	PCM write	35
Page	64/block	PCM r	0.115	Supply Voltage (V)	3.3
Sub-page	4/page	PCM w	90		
Over-provisioned (Redundant space)					0.1
Controller cache (KB)					128
GC threshold					0.2

SSD Organization



Average Response Time of Realistic Traces

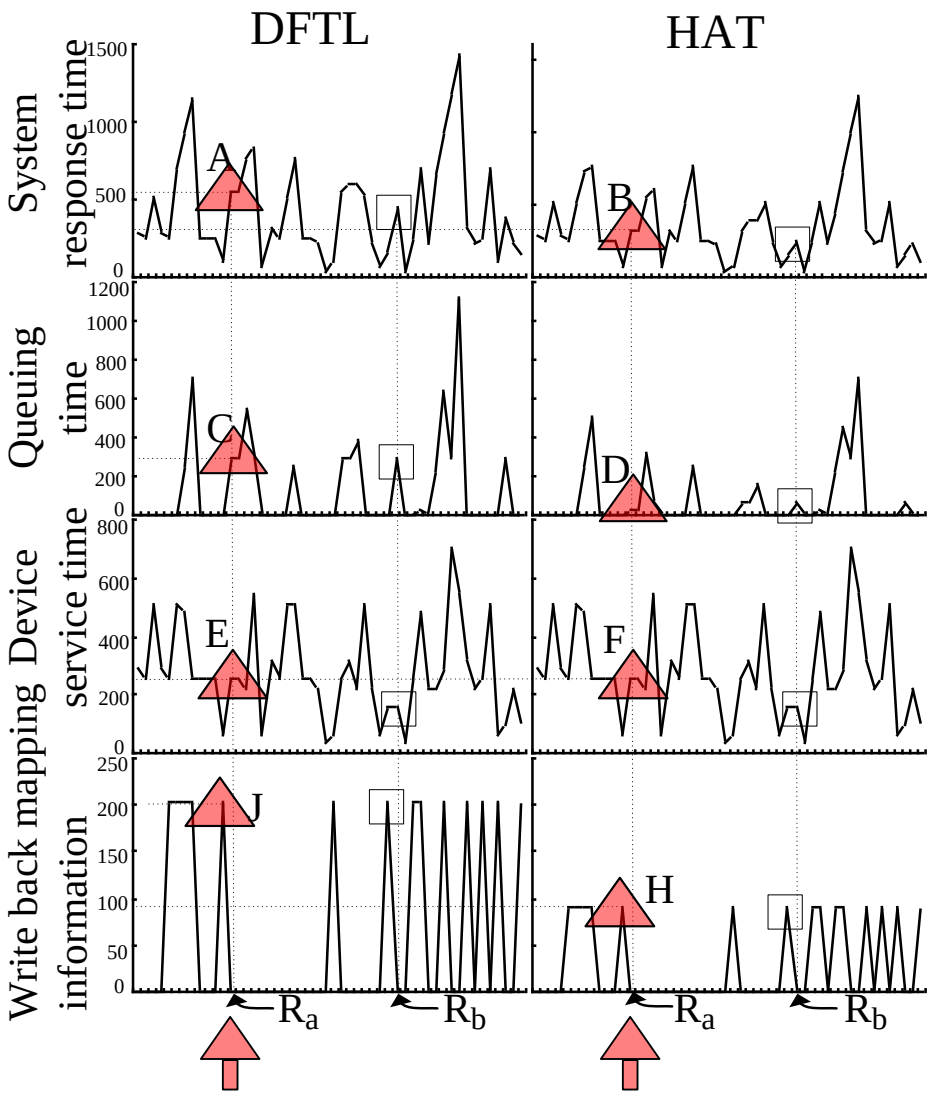




- Financial1:
- Write-dominant
- HAT has almost the same erase number as pure page-mapping FTL
- HAT has the same mapping information hit rate as DFTL (they has the same RAM space)

- Websearch1, 2, 3:
- Read-dominant
- Page-mapping FTL has the same response time as FAST
- Mapping information hit rate of DFTL is 2.5% → large number of extra flash read operations.

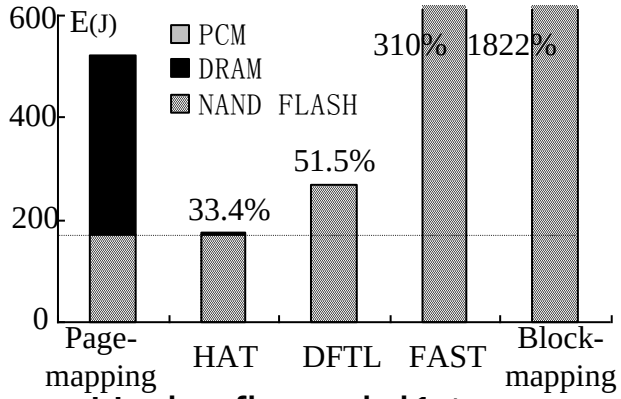
Microscopic Analysis



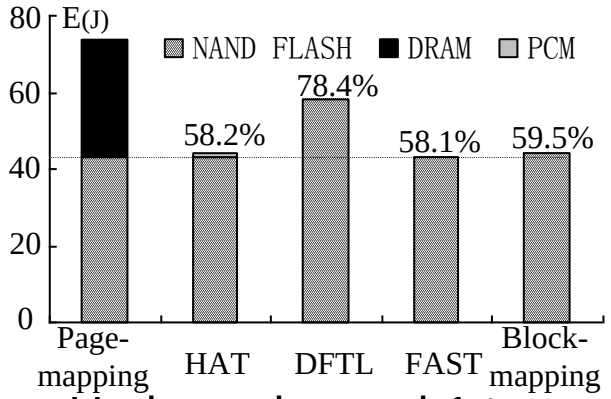
50 consecutive requests

- E, F: DFTL and HAT has the same device service time.
- J, H: DFTL and HAT are triggered a mapping information write back operation by the same earlier request.
- C, D: The same write back operation leads to different queuing time.
  (HAT hides write back mapping information operation)
- A, B: The same request has different response time.
  (DFTL: 520 us, HAT: 300 us)

Energy Consumption



Under financial1 trace



Under websearch1 trace

- Many merge operations consume high energy in FAST and Block-mapping FTL.
- RAM in page-mapping FTL consumes a major part of energy.
- DFTL has some extra mapping information read and write operations, which consume some more energy.
- HAT consumes the least energy.

- HAT, FAST, Block-mapping FTL consume less energy.
- DFTL has many extra read operations, which consume some more energy.
- Page-mapping FTL consumes the most energy due to energy-hungry RAM.

Conclusion

- Page-mapping FTL incurs high RAM cost and energy while block-mapping FTL suffers from inferior performance
- Hybrid FTL, such as FAST, incurs performance degradation under intensive write workloads.
- DFTL strikes a reasonable balance but still suffers due to mapping-and-data contention on critical path
- HAT hides address translation to provide high performance and energy efficiency.
- HAT achieves the performance of the pure page-mapping FTL scheme at a cost of the block-mapping FTL scheme.

Thanks!

Questions?

Any question needs the most detailed answer, please send email to
{ yanghu@foxmail.com }. Yes, you are welcome!