國立交通大學
National Chiao Tung University

A Flash Scheduling Strategy for Current Capping in Multi-Power-Mode SSDs

Li-Pin Chang, Chia-Hsiang Cheng, and Kai-Hsiang Lin
Department of Computer Science
National Chiao-Tung University, Taiwan

Presented at ASPDAC 2017, Japan

Flash Storage Devices

- High performance, low power consumption, low heat dissipation
  - Ideal storage solution for various computer systems
- Enterprise-scale servers
  - Non-volatile Memory Express (NVMe)
- Personal computers
  - Solid State Disks (SSDs)
- Smart devices
  - Embedded Multimedia Cards (eMMCs)
  - Universal Flash Storage (UFS)

Interface Power Modes

- Embedded storage interfaces may change the peak current limit for power saving and budgeting
- eMMC 5.1
  - 200 mA~900 mA
- USB 3.0
  - High-power mode: 900 mA
  - Low-power mode: 150 mA
- External storage device can be powered by the interface or an AC adaptor

Interface Bandwidth vs. Flash Parallelism

- High storage interface bandwidth
  - A PCIe lane supports up to 1GB/s
    - An NVMe SSD employ multiple PCIe lanes
  - SATA-3 supports up to 600 MB/s
  - eMMC 5.1 supports up to 400 MB/s

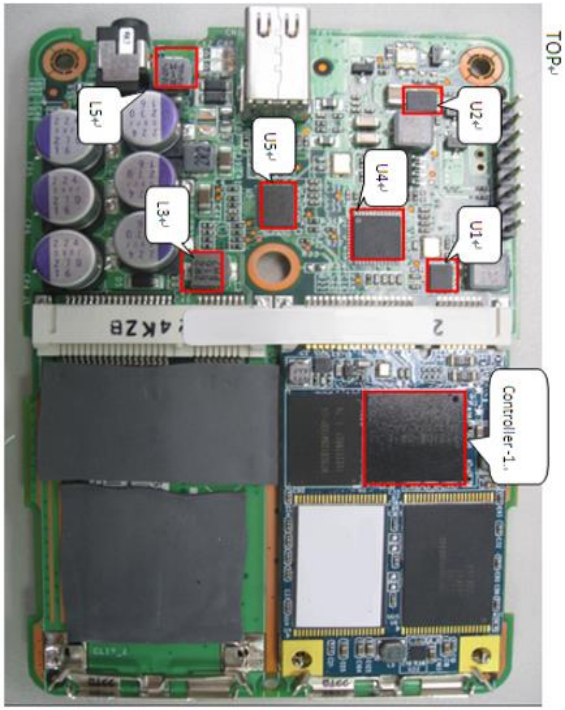
- The read throughput of a typical flash memory bank is only 40 MB/s [2]
  - Exploiting flash memory parallelism to fully utilize the bus/interface bandwidth
  - An NVMe SSD has more than 16*4 flash memory banks
  - A recently announced eMMC has 2*4 banks [3]

Peak Current vs. Flash Parallelism

- A page read consumes up to 50 mA on a typical flash memory bank
- In the eMMC, 2*4=8 concurrent page reads incur a peak current 50*8=400 mA
- Much higher than the default power class 200 mA
- Current overloading will cause unexpected voltage dropping

Current-Capping in SSDs

- The max. current supply of a Thunderbolt port is 550 mA
- Switching an SSD from the AC-powered mode to port-powered mode
  - Peak current 760 mA to 544 mA
  - Read thru. 590 MB/s to 315 MB/s
  - Peak curr. 30% ↓ but thru. 47%↓
- If current capping is disabled
  - The host loses connection to the SSD!
  - Data loss!



Problem Definition

- *Current capping*
  - control the peak current under a limit (cap) at all times
- Goals
  - Maximize SSD internal parallelism
  - Avoid peak current overloading

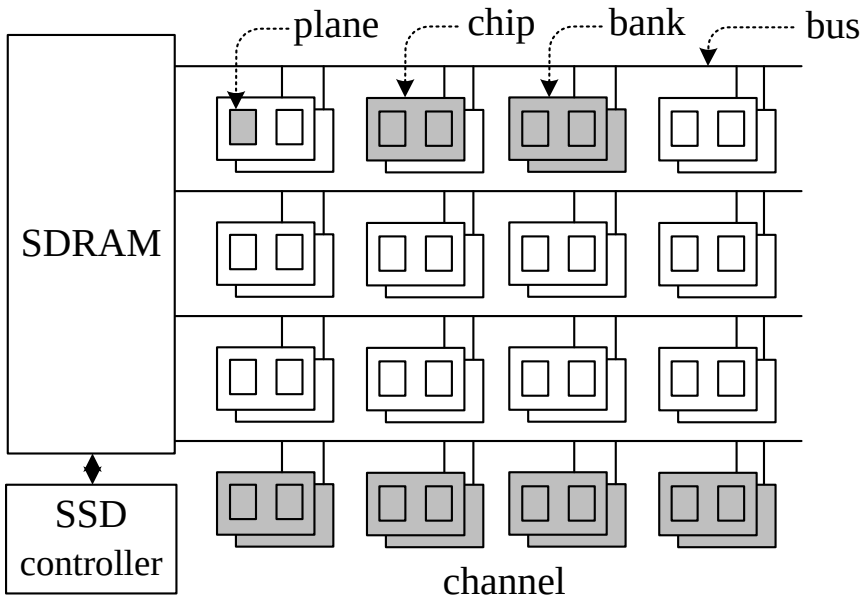
- In this study, a firmware approach is proposed

Ideas and Challenges

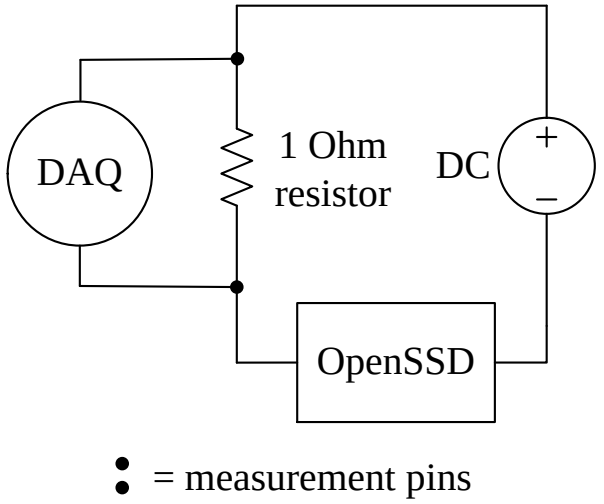
- Observations
  - The current varies during a flash operation
  - The current of different types of operations differs a lot
- Basic idea
  - To adaptively schedule flash operations to suppress the peak current under a cap at all times
- Challenges
  - Need to build accurate models of flash current usage
  - Too slow to check the total current at every time point
  - Flash current usage is subject to the aging process

Current Measurement

- Jasmine OpenSSD platform
- Measure flash current in terms of bank



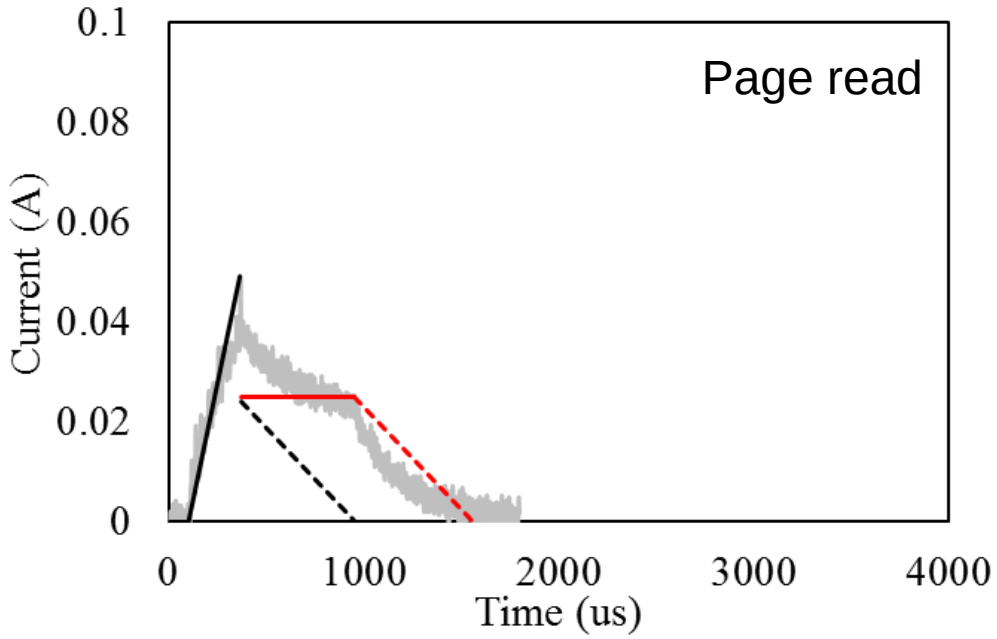
(a)



(b)

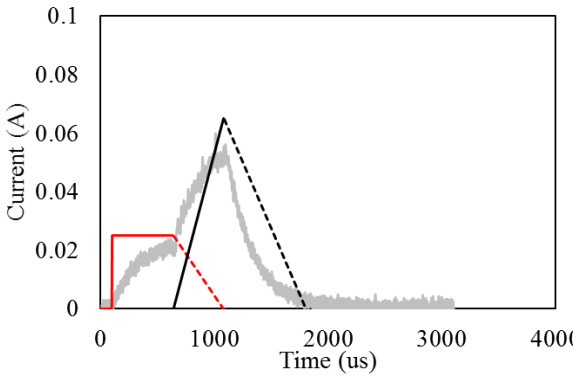
Page Read Current Usage

- flash busy →Bus transfer
- Both bus and flash incur tail currents
- Waveform of LSB and MSB pages are similar

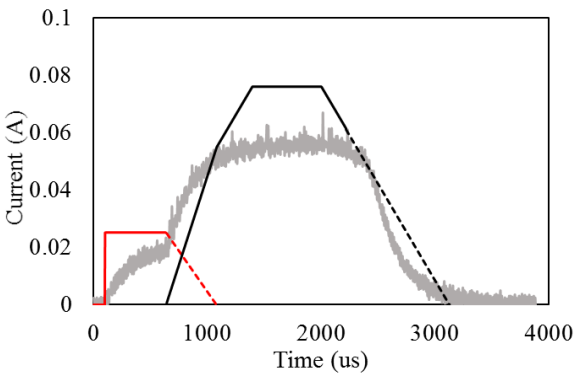


Page Write and Block Erase

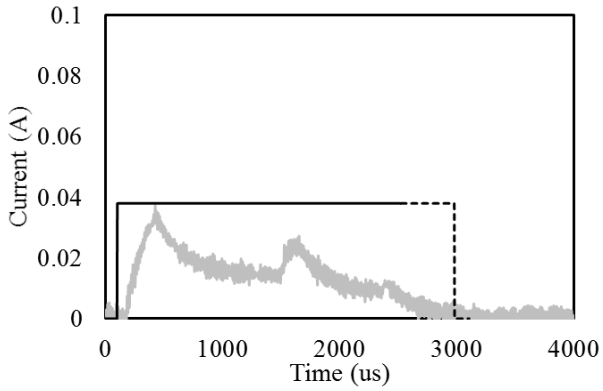
- Writing MSB pages consumes more energy than writing LSB pages
- Erasing a block does not involve bus transfer



LSB page write



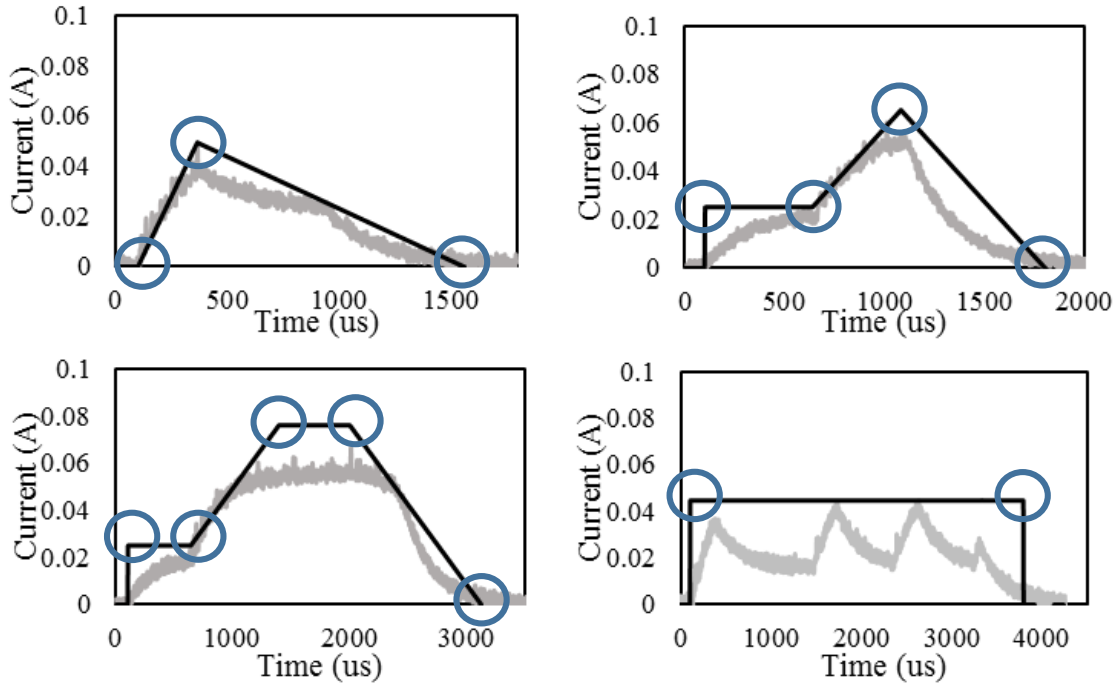
MSB page write



Block erase

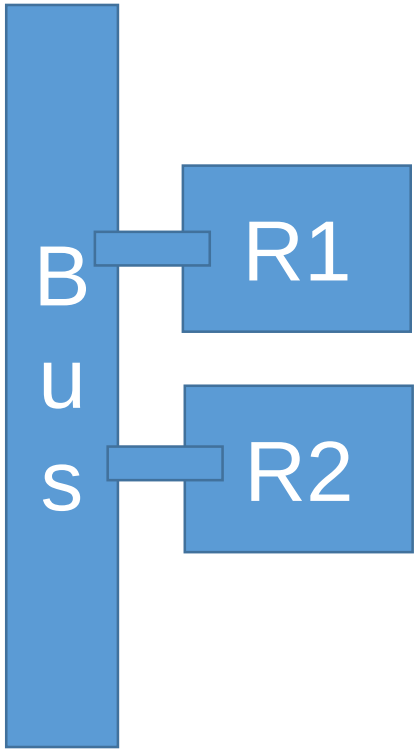
Corner-Based Flash Current Models

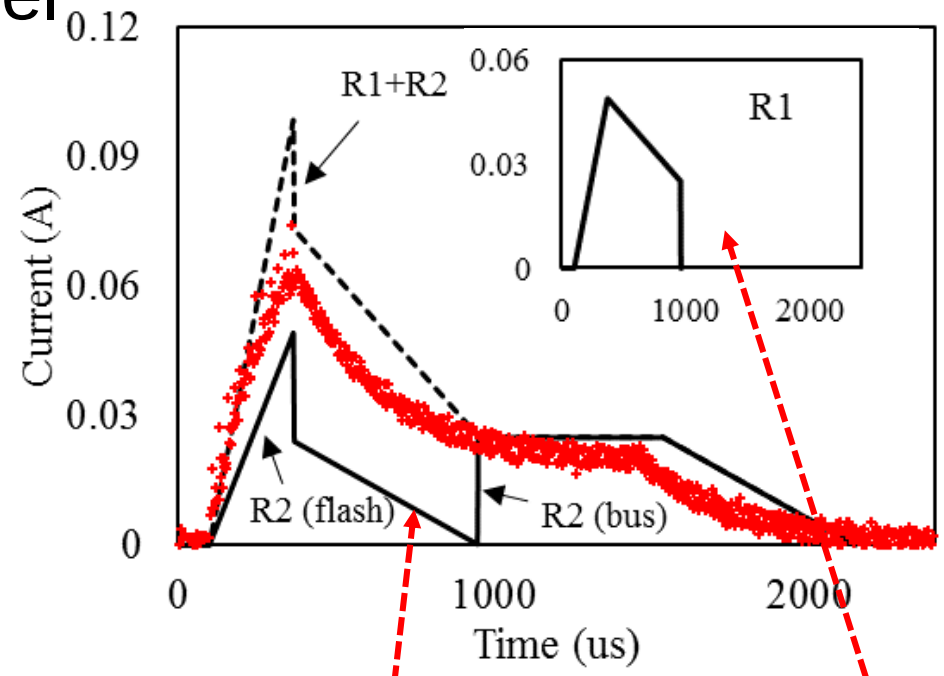
- The current usage of each type of flash operation can be modeled using a series of linear functions
  - A corner is the junction of two linear functions (us, mA)
  - Current model of R = {(0,0), (265,50), (1450,0)}
  - A corner represents a local maximum or minimal of current usage



Model-Based Current Estimation

- Two simultaneous page reads R1 and R2 in the same channel



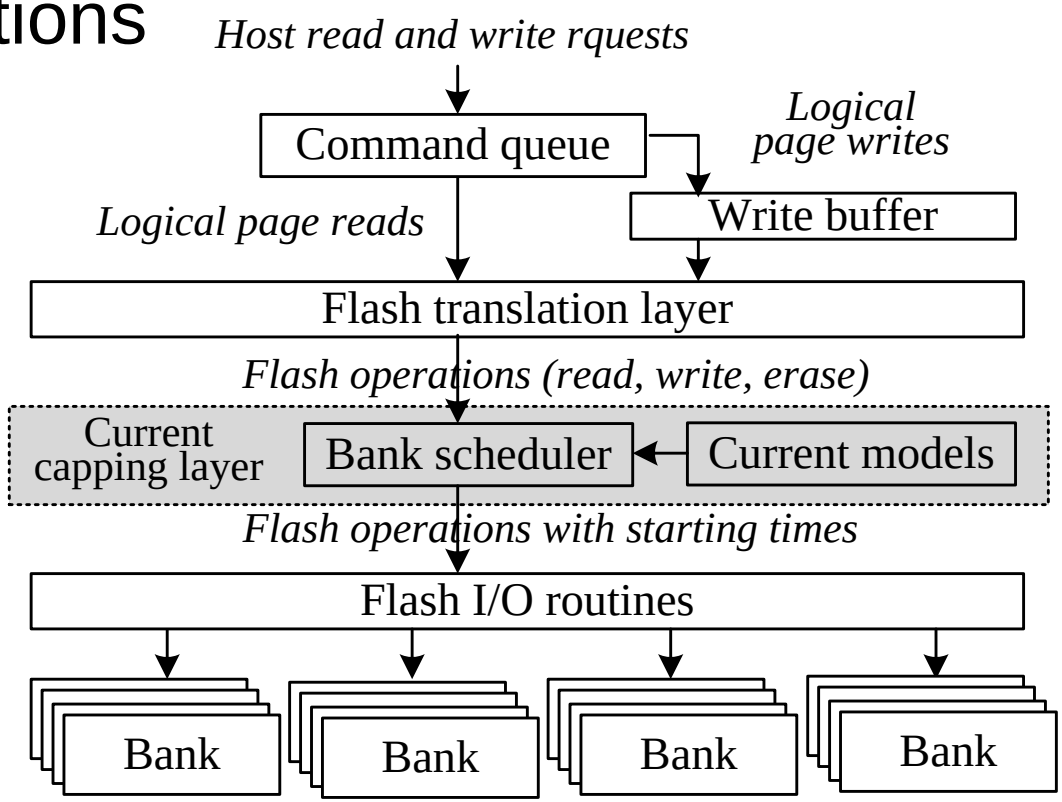


Bus contention splits the current usage of R2

R1 has no bus tail current

Current-Capping Flash Scheduling

- Between FTL and low-level flash routines
- Decide the actual starting time of bank operations

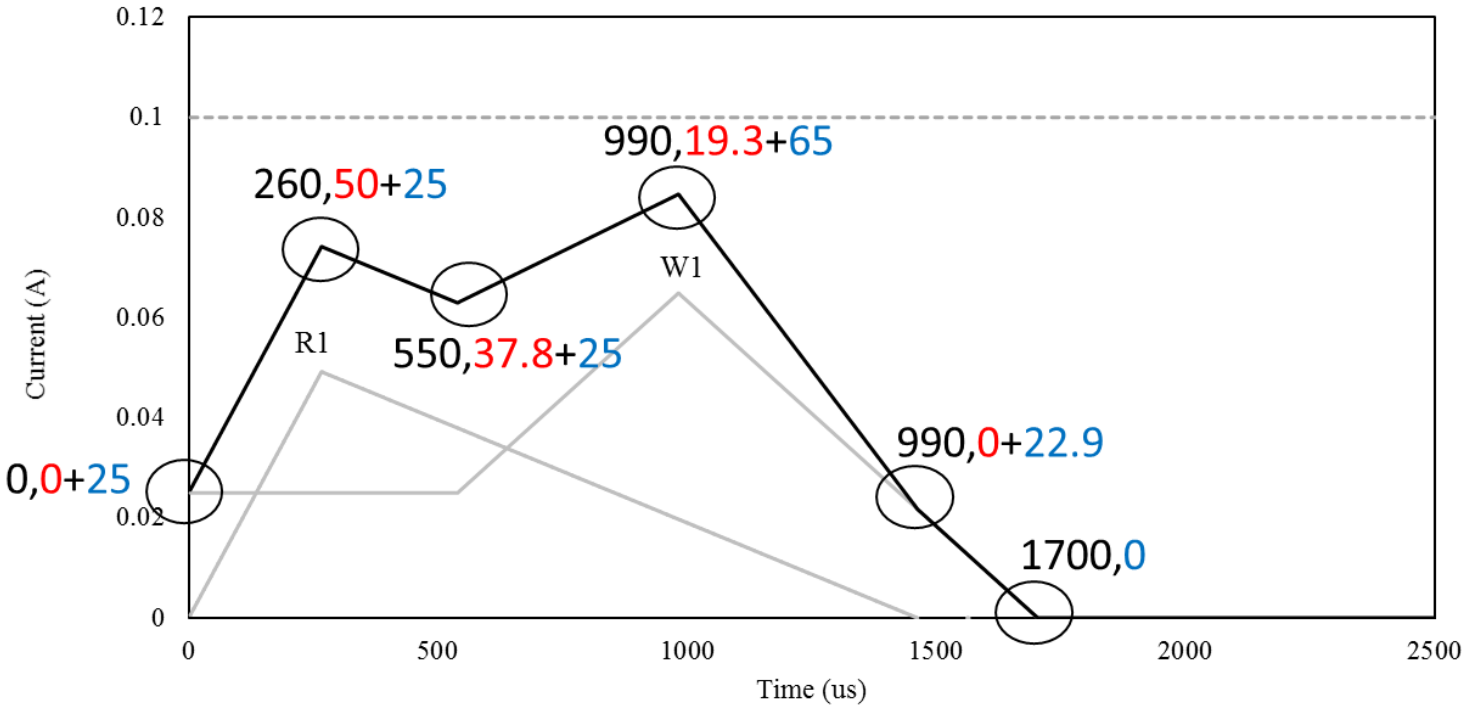


Flash Scheduling

- Bank status
  - Ready → Selected → Scheduled → Busy → Ready
- Bank selection
  - Channel with fewest busy banks to reduce bus contention
  - Favoring read over other operations for better read response
- Bank scheduling
  - A corner list L={} initially
  - Scheduling an operation = adding new corners to L
  - No corners should have current value > current cap

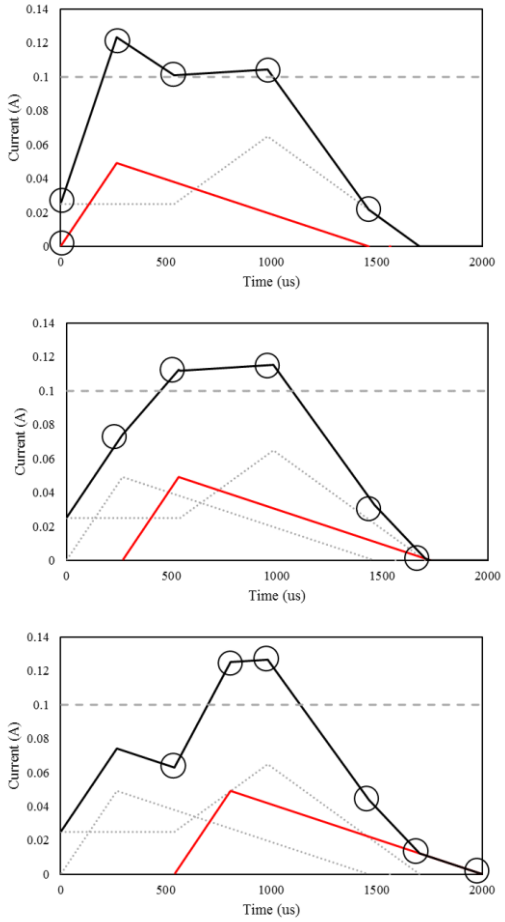
Flash Scheduling in Action

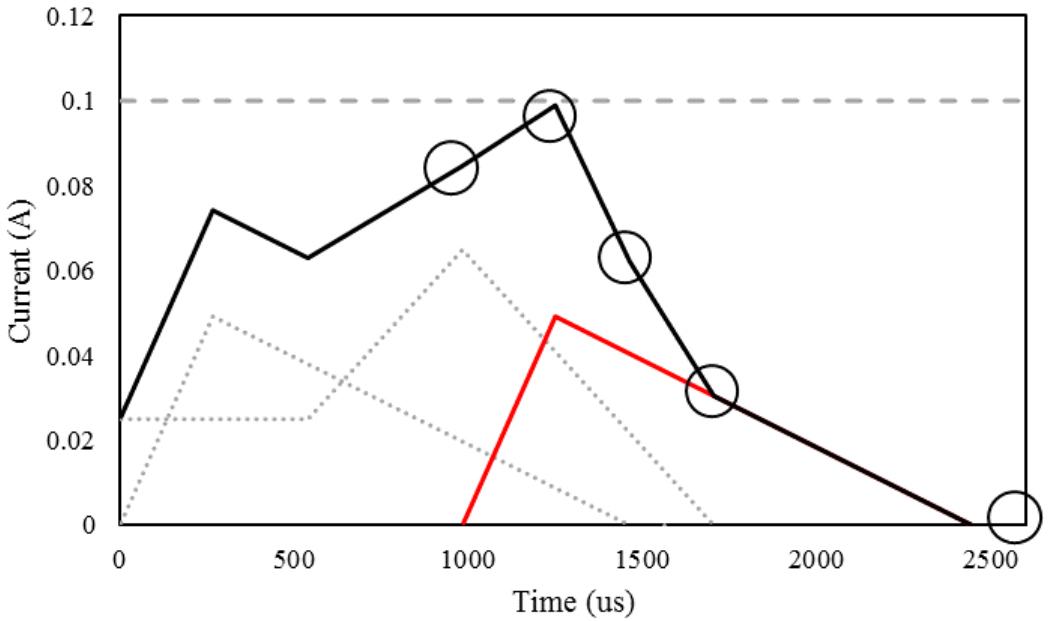
- Current cap = 100 mA
- R1, LSB W1 can safely start at time 0



Flash Scheduling in Action

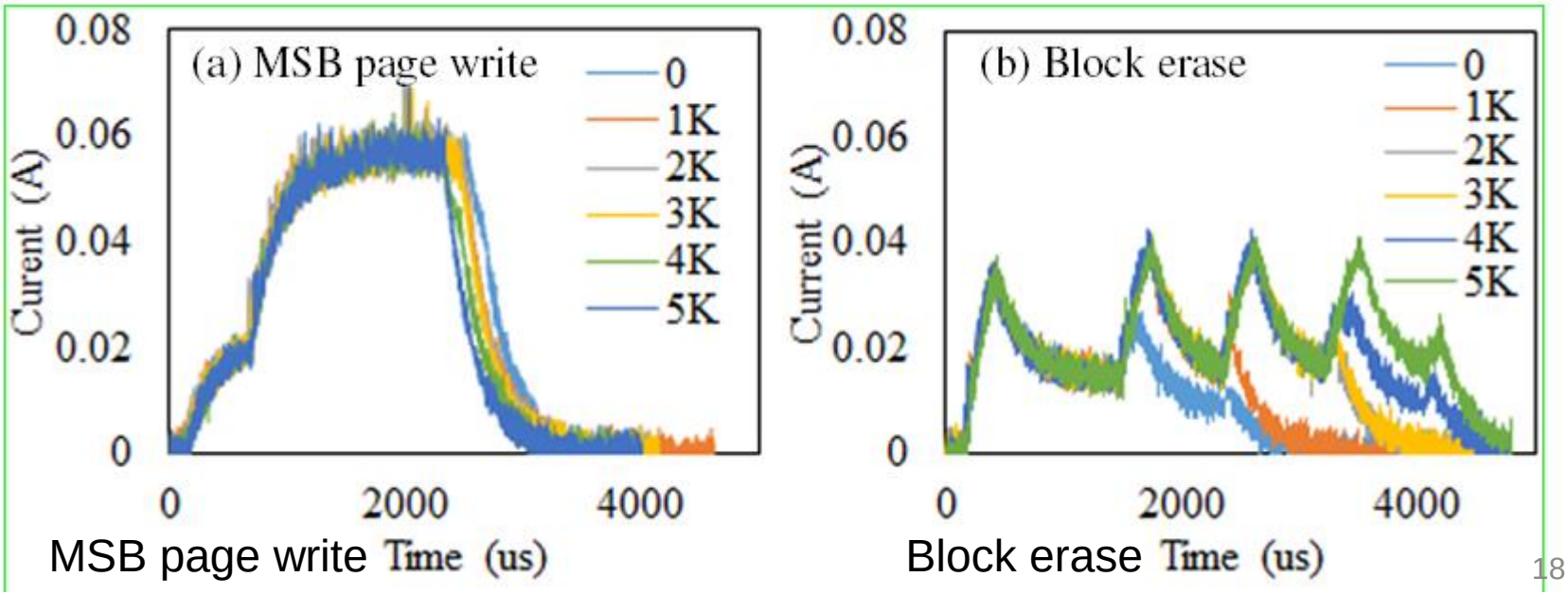
- R2 is scheduled for time 990 to avoid cap





Flash Aging

- Aging barely affect peak current
- Aging marginally speeds up MSB page write, but significantly slows down block erase
  - Erase current model evolves as flash ages



Experimental Setup

- Workloads

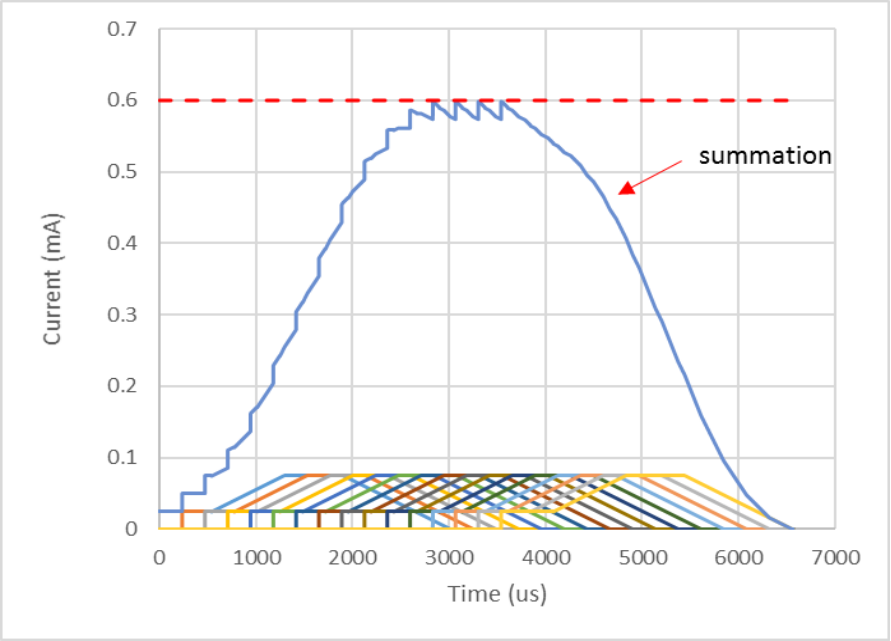
Workload	Volume size	Read (%)	Avg. req. size	Seq (%)	Type
PC	40 GB	31.20%	11.6 KB	22.80%	Mixed
TBL	30 GB	4.10%	7.95 KB	17.50%	Mixed
NB	20 GB	54.80%	18.8 KB	33.90%	Mixed
SVR1	400 GB	48.70%	11.1 KB	7.10%	Random
SVR2	400 GB	11.50%	19.2 KB	75.50%	Sequential
TPCC	180 GB	66.70%	8.1 KB	0.01%	Random

- Flash organizations

Parameter	Value	Parameter	Value
Bank organization	4 ch*4 bank	Overprovision	10%
Page, block size	32 KB, 4 MB	Native cmd. queue	32 requests
Write buffer size	4 MB	Current caps	200 ~ 1200 mA

Experimental Setup

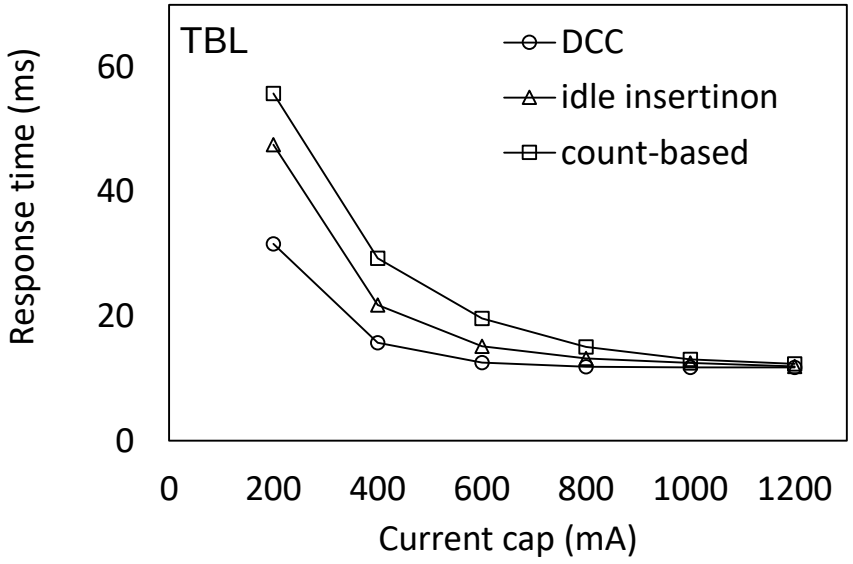
- Dynamic current capping (DCC)
  - Corner-based model
- Idle insertion [4]
  - Insert an idle between each request
- Count-base [1]
  - Square waveforms

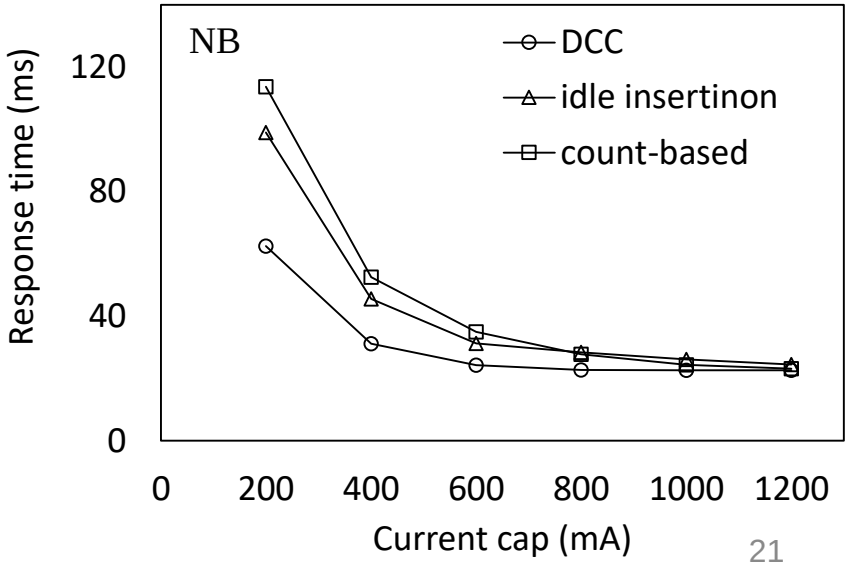


Cap (mA)	Idle (μs)
1200	40
1000	110
800	169
600	236
400	357
200	727

Results

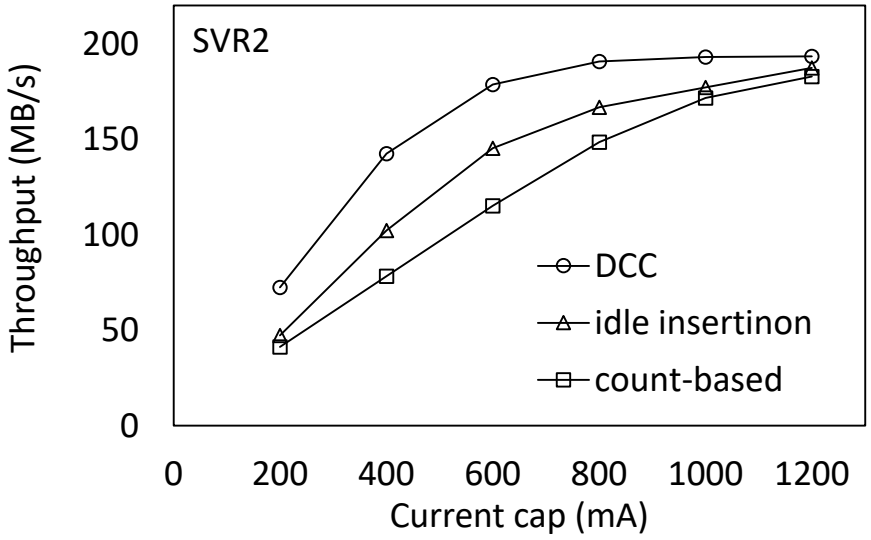
- Response
  - The advantage become significant when capping lower 600 mA
  - The response time of TBL is shorter than NB due to the concentration of write requests

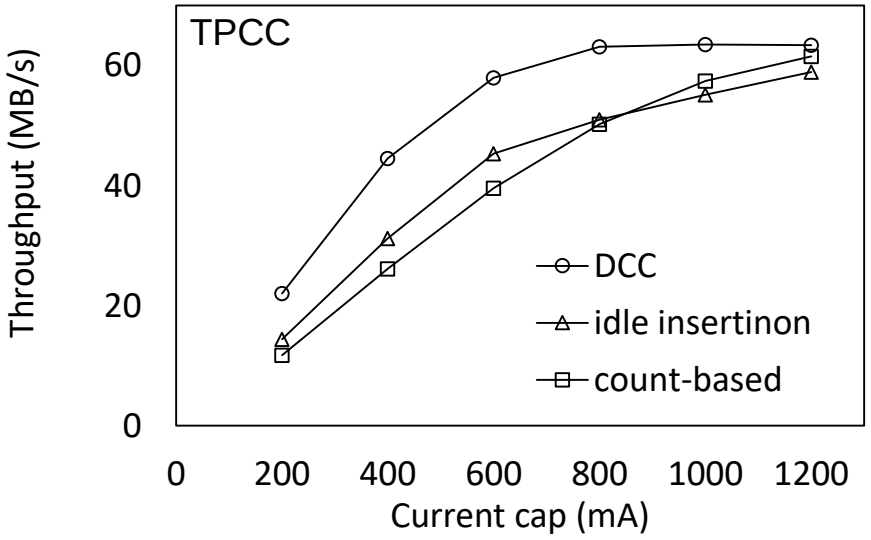




Results

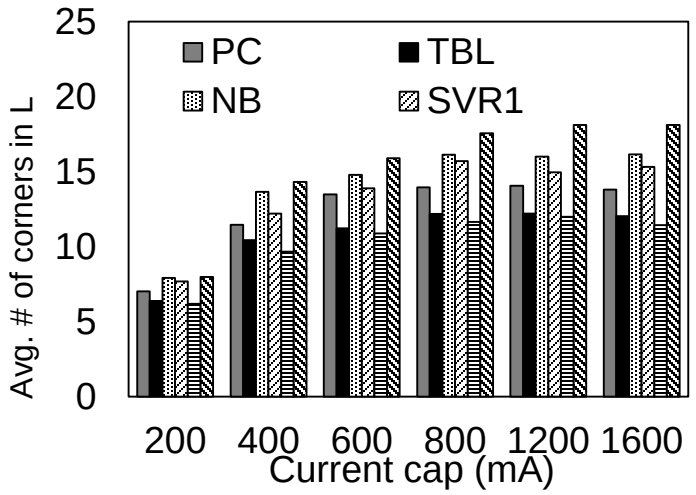
- Throughput
  - DCC reached the highest throughput from capping higher than 800 mA
  - Throughput of DCC higher than other algorithm under any cap current

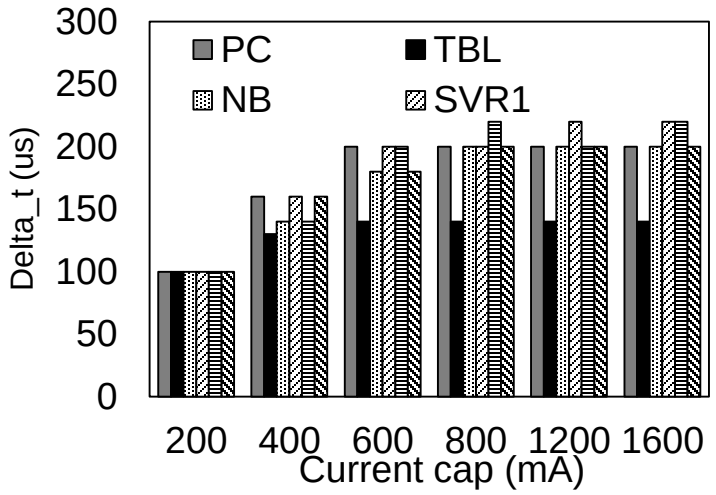




Results

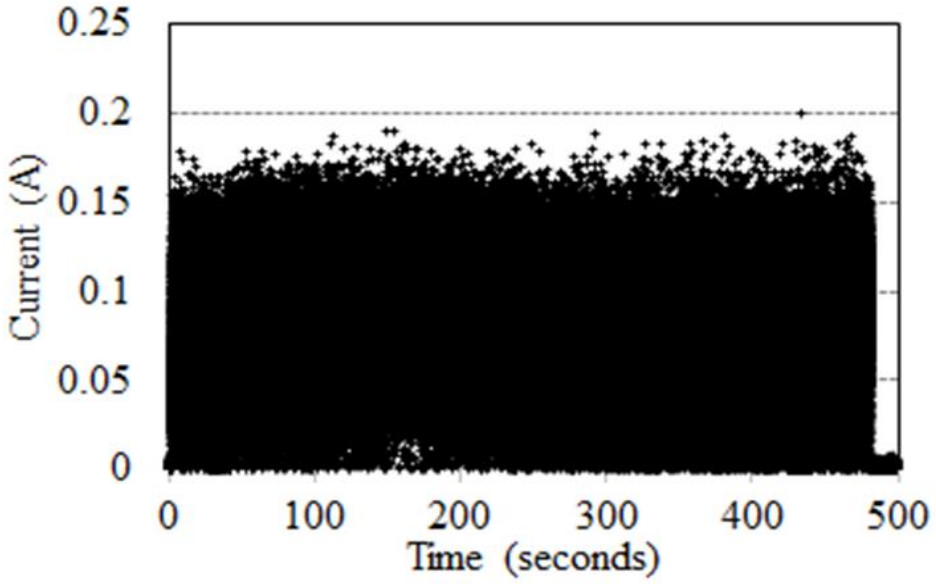
- Scheduling overhead
  - TB & SVR with frequent high-current write request restrict the flash parallelism
  - Preserving less than 250 us to schedule a single request

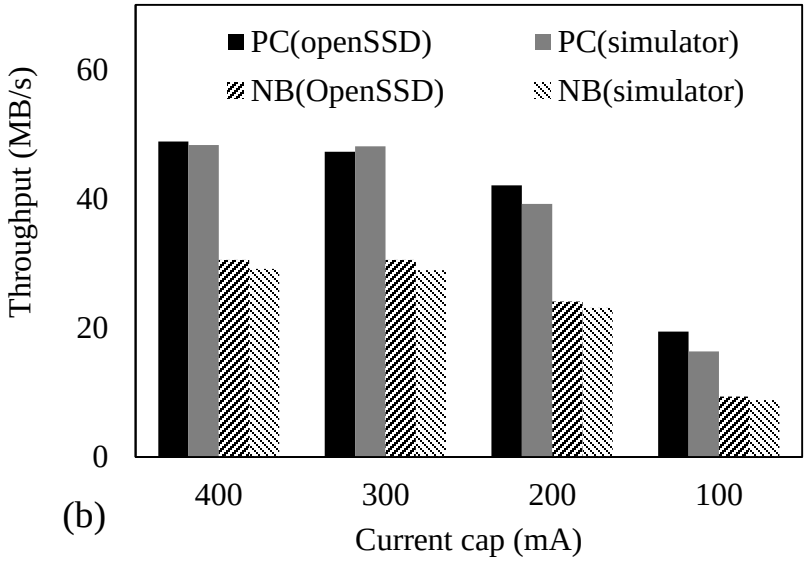




Results

- Implement on OpenSSD Jasmine platform and successfully limit the current under 200 mA
- Throughput on OpenSSD is close to simulator





Conclusion

- Rich flash parallelism may incur high instantaneous peak current
  - Performance vs. Power budgeting/throttling
- This study introduces a firmware-level scheduling strategy for current capping
  - Realistic current models of flash operations
  - Fast corner-based flash scheduling
  - A proof-of-concept based on OpenSSD
- Future work
  - Evolving current models with consideration of flash aging and process variation

Thanks for listening
Q&A

Reference

[1] T. Hatanaka and K. Takeuhi. Nand controller system with channel number detection and feedback for power-efficient high-speed 3D-SSD. Solid-State Circuits, IEEE Journal of, 47(6):1460-1468, June 2012.

[2] Samsung Electronics Company. K9GBG08U0A 32Gb Adie MLC NAND Flash Data Sheet, 2009.

[3] Sandisk. iNAND Extreme embedded ash drive, 2014.

[4] K. Takeuchi. Novel co-design of nand flash memory and nand flash controller circuits for sub-30 nm low-power high-speed solid-state drives (ssd). IEEE Journal of Solid-State Circuits, 44(4):12271234, April 2009.