Last Class!

Today

⇒ Persistent Storage
  ⇒ HDDs
  ⇒ Flash-based SSDs

⇒ Advice
  → students (ugrads, grads)

⇒ wrap-up
  → what did we learn?

Flash-based Solid-state Storage Devices (SSDs)

⇒ "new" base technology in 1980's (Toshiba)

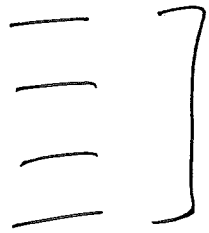
basics: flash **cell** (storage for one/few bits)

⇒ traps charge in cell

SLC high ⇒ 0, low ⇒ 1
[single-level cell]

multiple bits/cell

MLC



Flash chip

⇒ interface / internals

⇒ store many bytes of data

Device:

many chips

any storage:

larger blocks

(sector: 512 bytes, larger blocks: 4KB)

Interface:

read (page #)

⇒ data

write:

weird (SSD)

⇒ erase (block #)

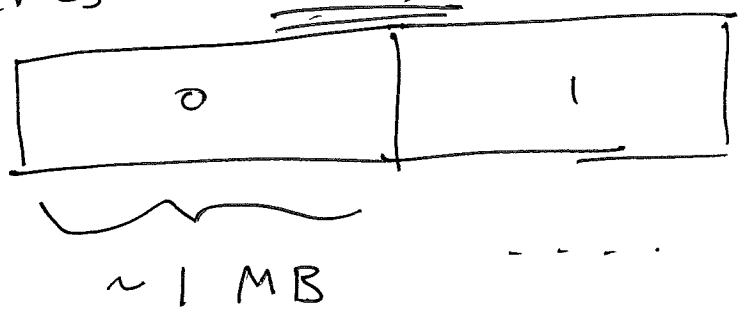
⇒ enables us to program pages in block (1 time each)

⇒ program (page #)

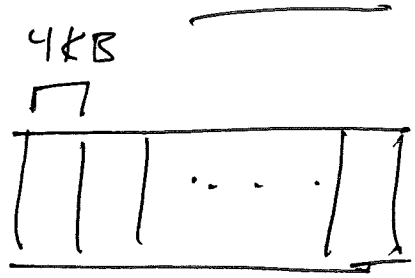
sets bits of that page to desired content

Internals:

series of blocks:



each block divided into pages (SSD)



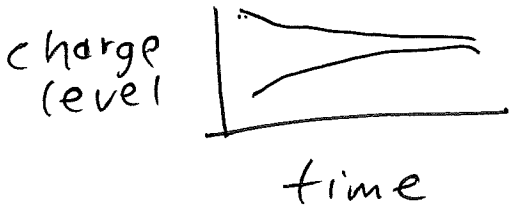
Performance

		latency (much faster than HDD)
read	⇒	~ 10 μs (microseconds)
erase (block)	⇒	~ 1 or 2 ms
program	⇒	~ 100s μs

Reliability

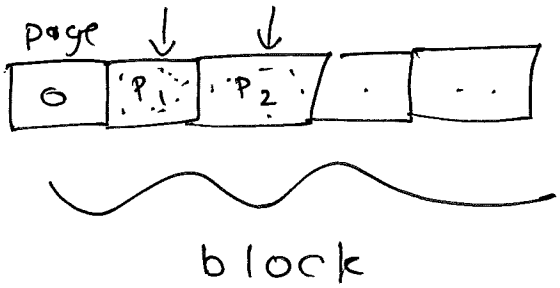
⇒ Primary problem: wear out

"wear leveling"
goal: spread out load writes



~ 10 K → 100 K erase/program cycles

⇒ Secondary problem:
read/program disturbance



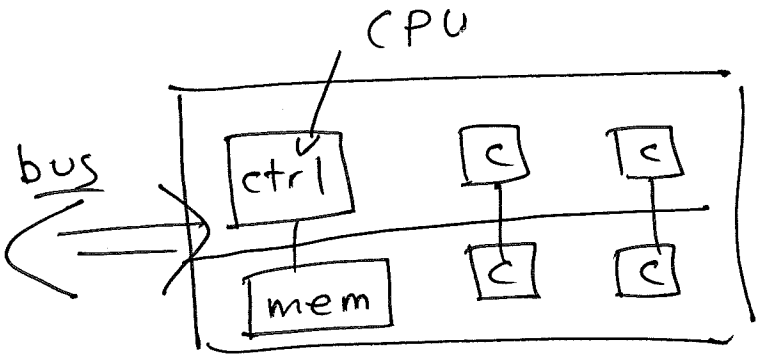
Goal: Build HDD replacement

w/ same API

HDD API: read, write (blocks)

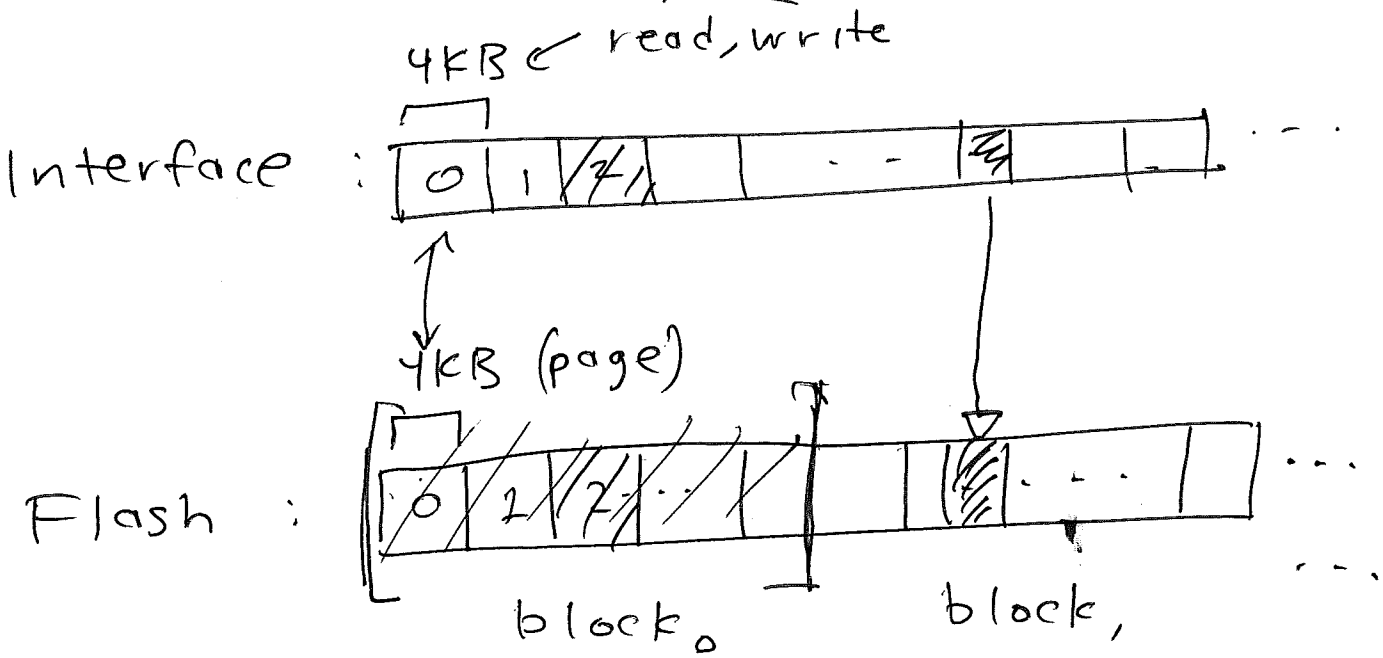
Map reads/writes

⇒ Flash reads, erase, program



Flash-based SSD

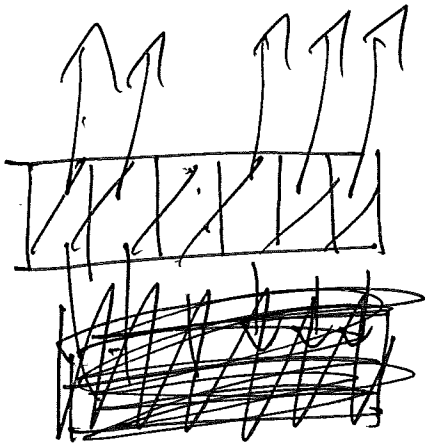
Naive : Direct map



interface read (block = 100) ⇒ flash read (page = 100)

write (block = 2) ⇒ read block (page by page) ⇒ erase block

Flash SSD memory



program each page (mostly w/ old, new too page)

write ( ) ⇒ read (block)
erase (block)
program (block)

Problem #1: [way too slow]

Problem #2: block wear out
(in control of programmer)

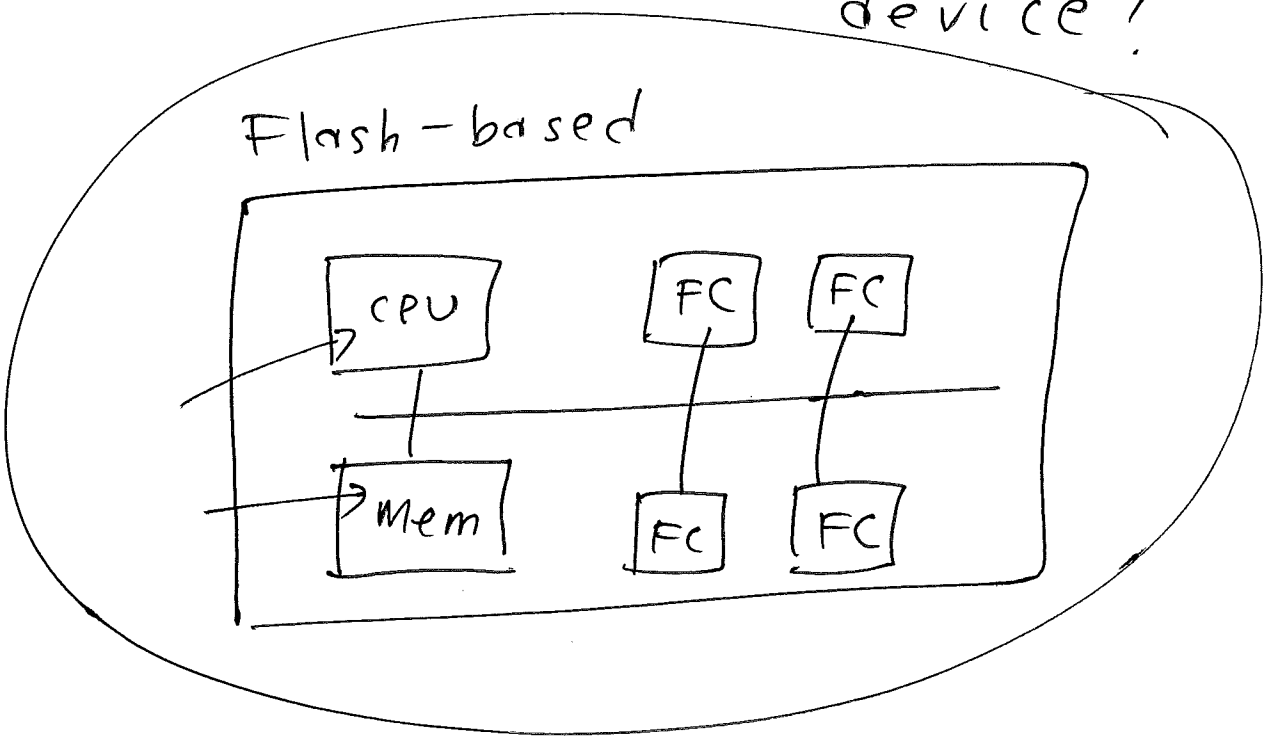
Question:

How to design Flash-based SSD

so as to

1) get <u>good</u> write perf?

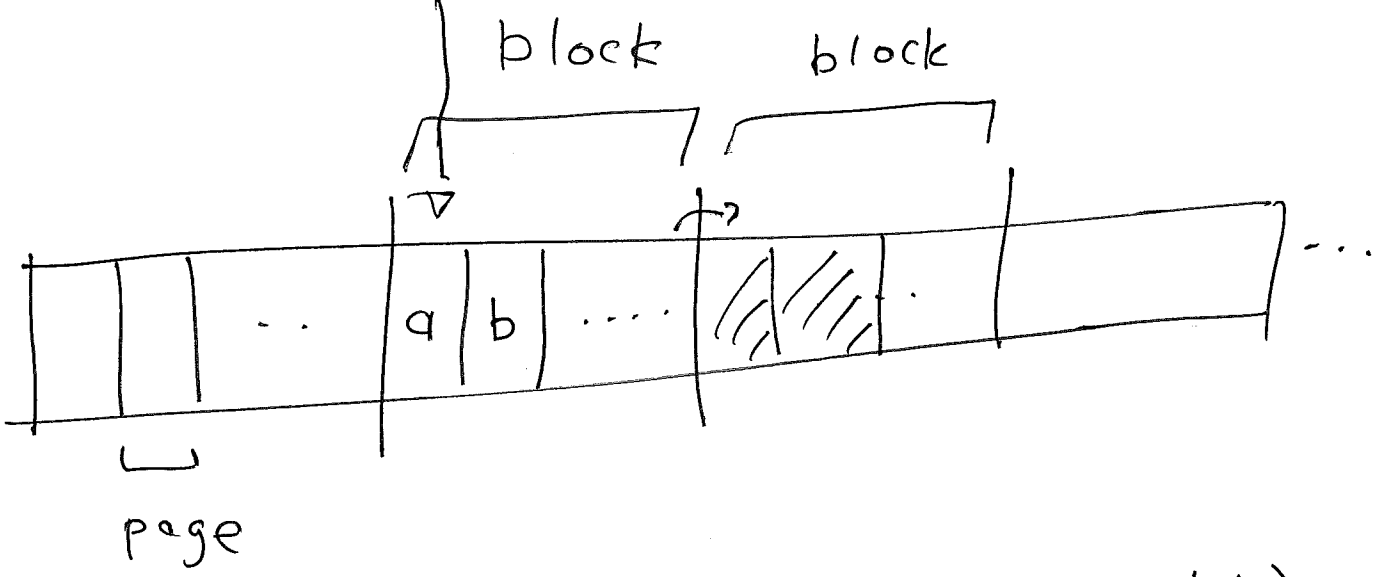
2) not allow user to quickly wear out device?



interface:

4KB

write (address = 100, data = 'a')

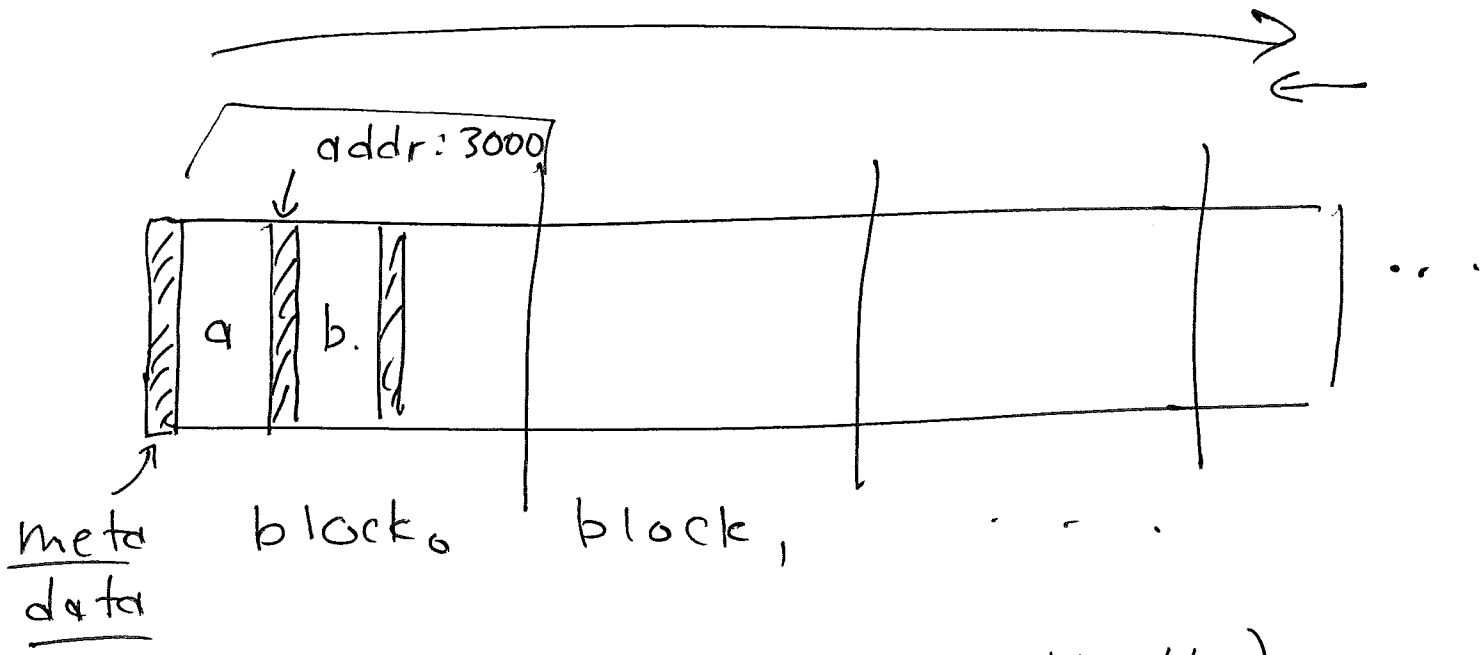


write (address = 2000, data = 'b')

⇒ "log structuring" (append only)
"logging"

Problems: 1) how to find info?
2) run out of space?

1) how to find info?



write (address = 100, "a")

address : 100

write (address = 3000, "b")

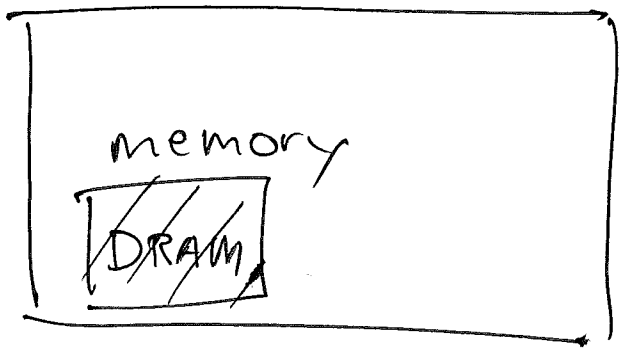
read (addr = 3000) ⇒

need more info:

[mapping table]

Mapping Table:

Flash Translation Layer (FTL)



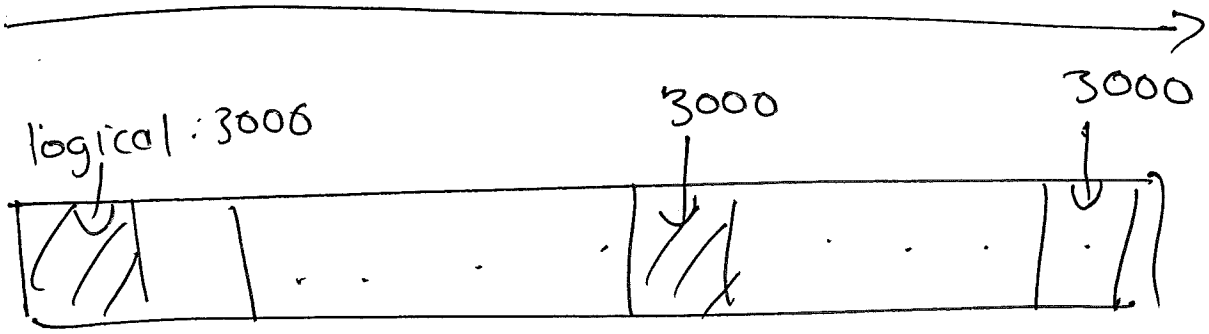
for each page: logical => physical

$$\begin{pmatrix} 3000 & \rightarrow & 2 \\ 100 & \rightarrow & 3 \\ 4500 & \rightarrow & 4 \end{pmatrix}$$

Problem: what if device loses power?

→ (many solutions)

What if device fills up?



⇒ need: free "old" copies of data

garbage

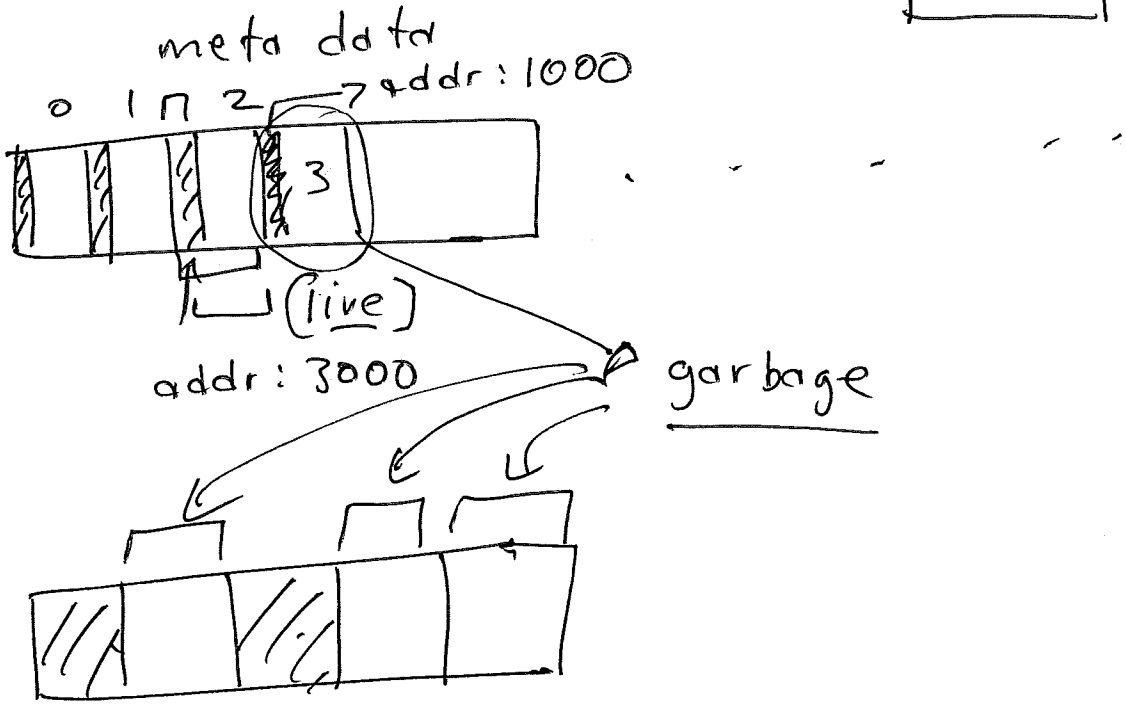
⇒ reclaiming space: (garbage collection)

⇒ Garbage collection:

⇒ how to tell if page is garbage?

mem (map table)

3000 → 2
1000 → 100



⇒ how reclaim free space?

⇒ move live data new block

⇒ erase entire old block

+ reuse it

CS: 354

"The End"

=> Admin:

hw5 graded

Competition: Soon

hw6: [due tomorrow]

"optional" final:

[=> late fri, early sat: produce grade]

final: monday @ night

=> [I prepared for this exam.

=> _____]

Exam:

⇒ prepare

⇒ 1 page cheat sheet

(both sides)