Lecture 14: Caching intro

Thursday, February 22, 2018 9:32 AM

Outline

- Why memory hierarchy
- How to control memory hierarchy
  - Data movement technologies
- Cache basics
  - Direct-mapped
  - Tag & data
  - Miss & hit
  - Handling writes

From last time…

	Typical access time	Cost per gigabyte	Max in machine
SRAM	~1ns	$300	~16B
DRAM	~20ns	$20	~1TB
Flash	~10µs	$0.50	~100TB
Disk	~10ms	$0.03	~100TB
3dxpoint	~100ns-1µs	$1.25	~1TB

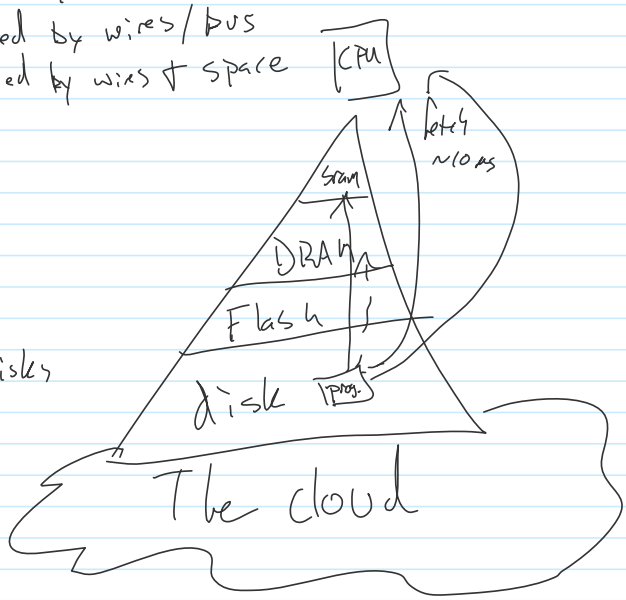
10x 100x 1000x

→ limited by power
→ limited by power & space
→ limited by wires/bus
→ limited by wires & space

not best at anything

Hierarchy

Small amount of SRAM
Some DRAM
lots of storage Flash/disks



how to move data?

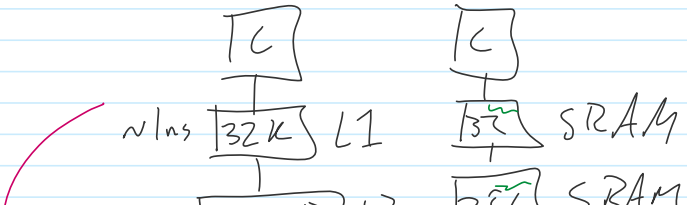
- caching

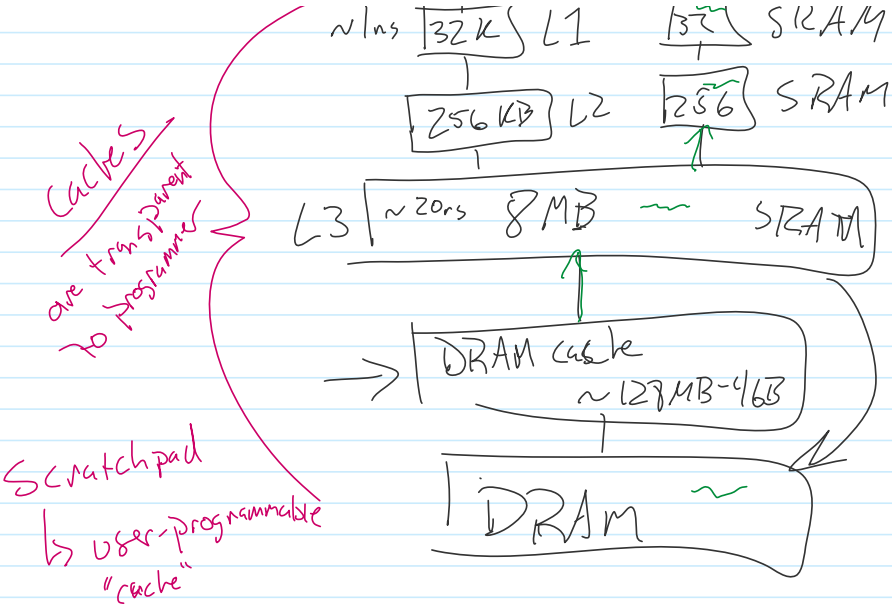
→ copy "some"
↳ small amount
grab stuff near → things nearby are likely to be used
↳ spatial locality
↳ one byte?

temporal locality
↳ reuse same data

move from disk to DRAM
↳ 64kB 1 GB
→ 2 MB

move from DRAM to SRAM
~64 Bytes





Caching

Questions to answer

- Where to look for data?
- How to tell if it's the right data?
- What to do if the data isn't found?
- What to do with writes?
- How to pick a location if multiple options?

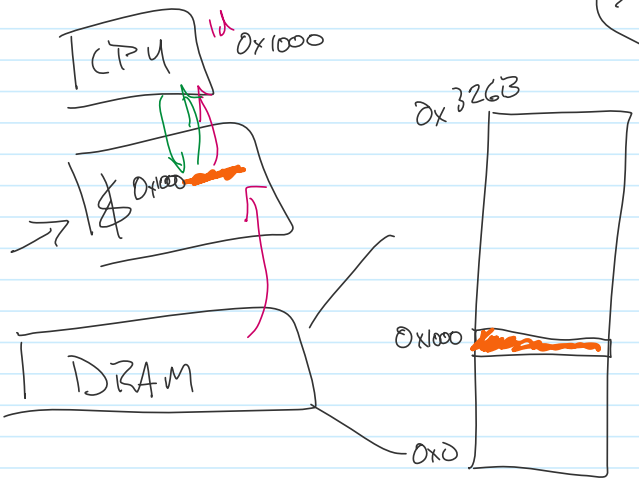
take advantage of locality

Temporal → things are reused

Spatial → likely to access nearby things

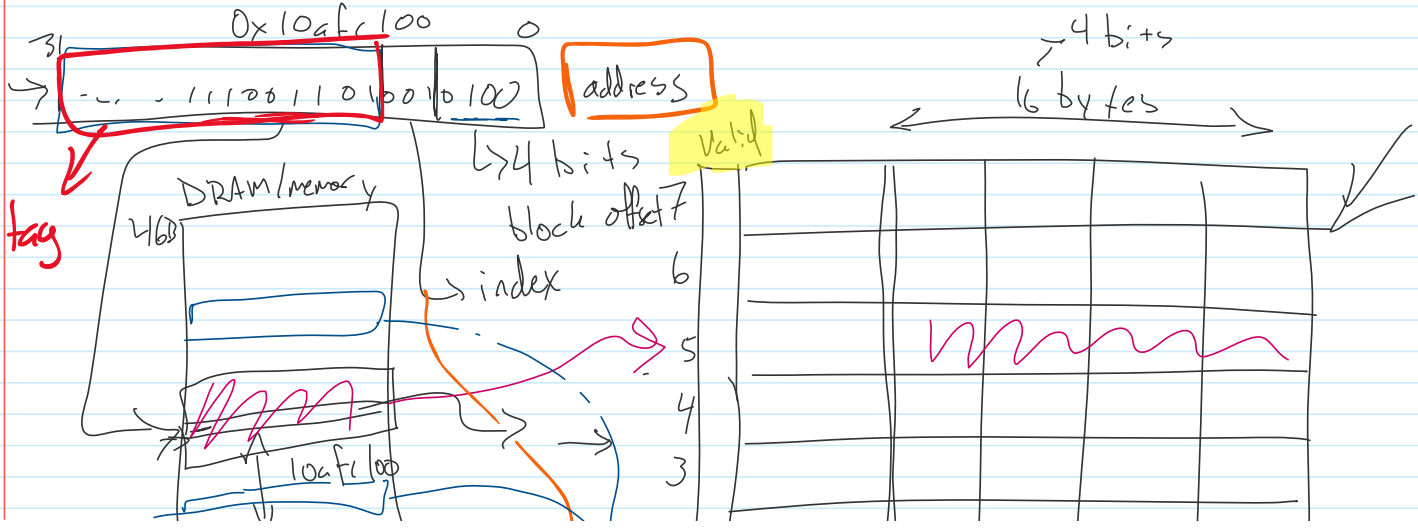
↳ keep things around

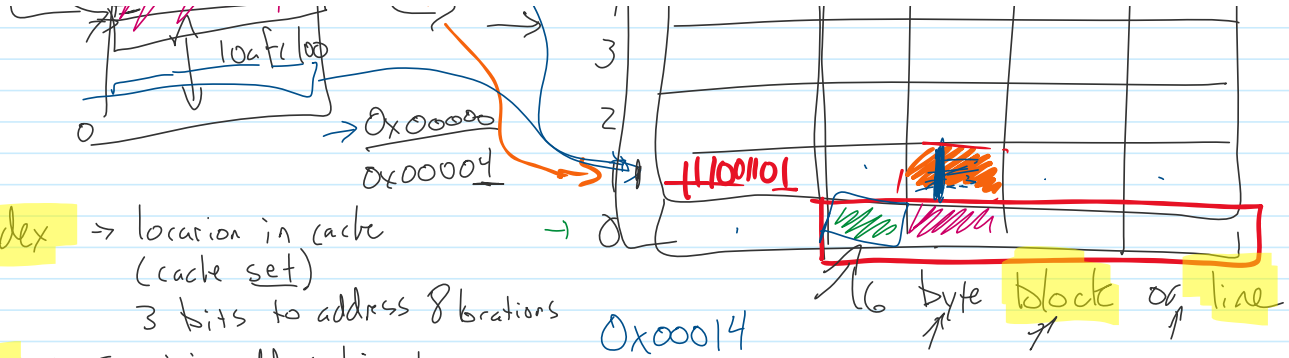
↳ fetch nearby things



Cache → behave like hash map
python dict.

0x1000 →

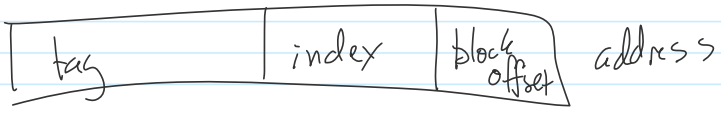


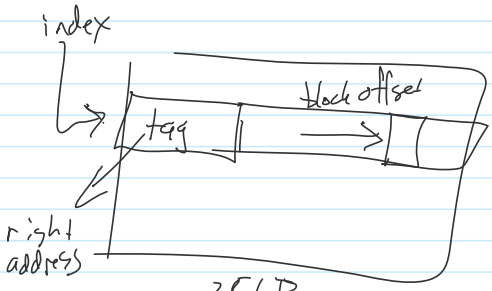


index → location in cache (cache set)
3 bits to address 8 locations

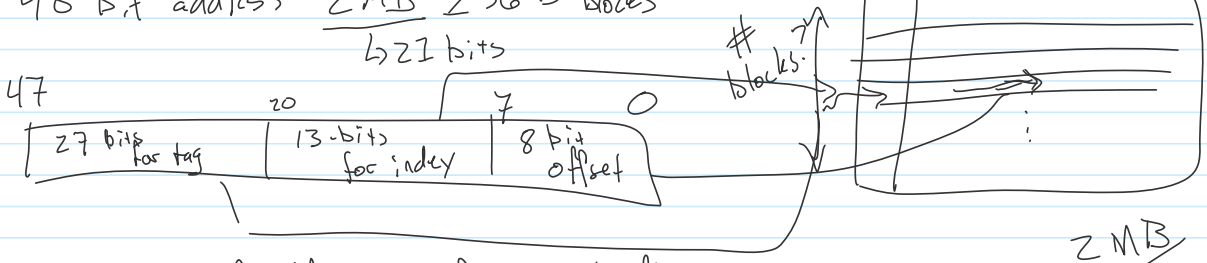
tag → remaining address bits to know which address is cached

block offset → how far into block/line to get data





48 bit address 2MB 256B blocks
↳ 21 bits



which bits of addr. use for block off., index, tag?

$\frac{2MB}{256B} \rightarrow \frac{2^{21}}{2^8} = 2^{13}$