Lecture 26
July 11

Cache

The problem: large difference in speed b/w different types of storage

- Cache
  - Fast data storage that is used to store frequently accessed data
  - After used to compensate for slow data access
  - If we can keep the info that we need to access in the cache, we can access it quickly
  - E.g. web cache: store recently accessed web pages. If one is accessed again, get it form the cache

Memory cache

- A small amount of SRAM placed between the CPU & memory
- Typically same speed as the processor (or 2X slower)
- Approx 256kb – 1MB in modern desktop PC
- Mirrors some of the information from RAM
- When the CPU needs to access memory:
  - First, check the cache
  - If it's there, return it (1-2 cycles)
  - If not, fetch it from memory. Give it to the process **and** store it in the cache (5-10 cycles)
- If it's in the cache: **cache hit**
- If not: **cache miss**
- Most architectures are set up so several adjacent words are read & cached for each miss
  - i.e. get a group of 4 or 8 words into the cache w/each miss
- two problems:
  - what if the cache is full?
  - Will storing this data in cache actually help?

Locality of reference

- Keeping the right data in the cache is the hard part
  - We have ato guess what will be accessed next
- Most memory (& disk) accesses are not random
  - Often access the same data
    - … or nearby data
    - e.g. instruction in a lop (same data)
    - e.g. next instruction, next array element (nearby data), netxt data in a file
- so if we keep recent & nearby data in the cache, we have a good chance of a cache hit

Access time

- How long will it take (on average) to do a memory access?
  - Assume:
  - 95% cache hit, 1 cycle access
  - 5% cache miss, 6 cycle access
- then average access time:
  - 0.95(1) + 0.05(6) = 1.25 cycles

Cache Memory

- how do we decide what to throw away & what to keep in the cache?
  - And how do we keep track of what's in the cache?
- Easiest method: direct cache
  - Every memory address is assigned one spot in the cahce where it could be sotred
  - Use the last bits of the word's address & use that as the cache address (index)
- E.g. 8 bit memory address and an 8 word (3 bit) cache memory address

tag									index		

- The index indicates a memory address' (potential) address in the cache