CSE 451: Operating Systems
Spring 2010

Module 12.5
Secondary Storage

John Zahorjan
zahorjan@cs.washington.edu
Allen Center 534

Secondary storage

- Secondary storage typically:
  - is anything that is outside of "primary memory"
  - does not permit direct execution of instructions or data retrieval via machine load/store instructions

- Characteristics:
  - it's large: 250-2000GB
  - it's cheap: $0.08/GB
  - it's persistent: data survives power loss
  - it's slow: milliseconds to access
    - why is this slow??
  - it *does* fail, if rarely
    - Or not so rarely these days...

Another trip down memory lane …

IBM 2314
About the size of
  6 refrigerators
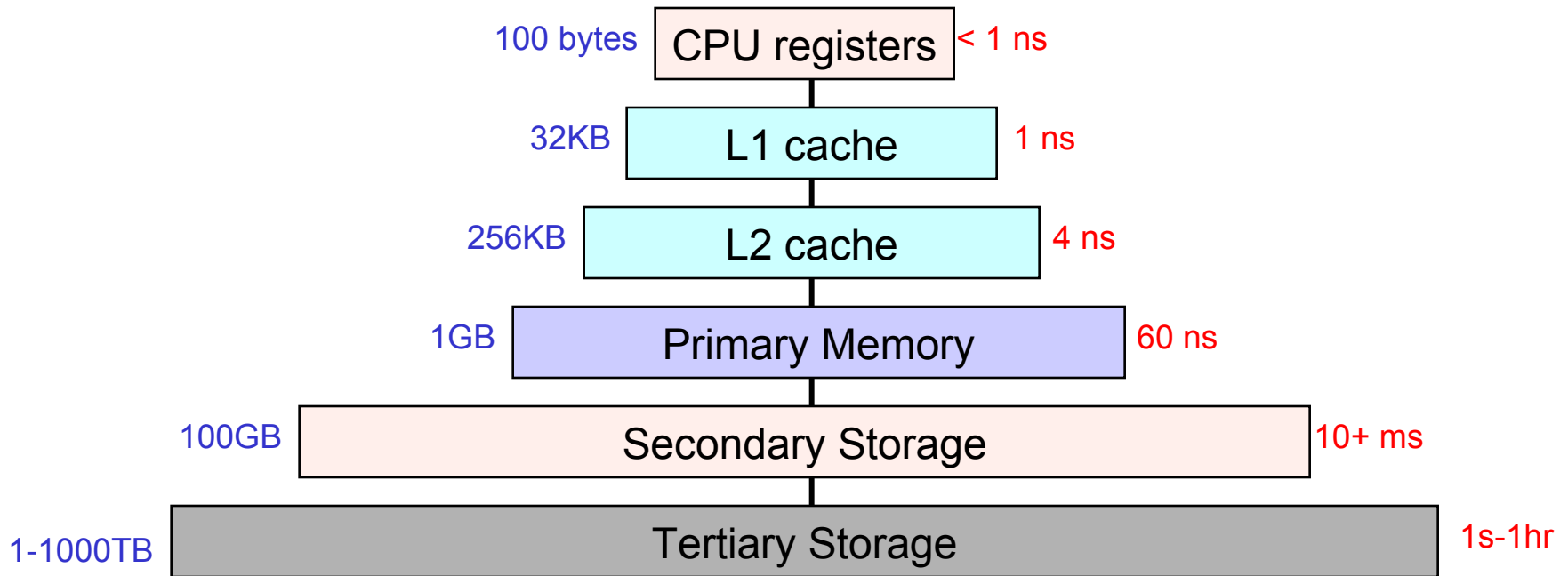8 x 29MB (M!)

Disk trends

- Disk capacity, 1975-1989
  - doubled every 3+ years
  - 25% improvement each year
  - factor of 10 every decade
  - Still exponential, but far less rapid than processor performance

- Disk capacity since 1990
  - doubling every 12 months
  - 100% improvement each year
  - factor of 1000 every decade
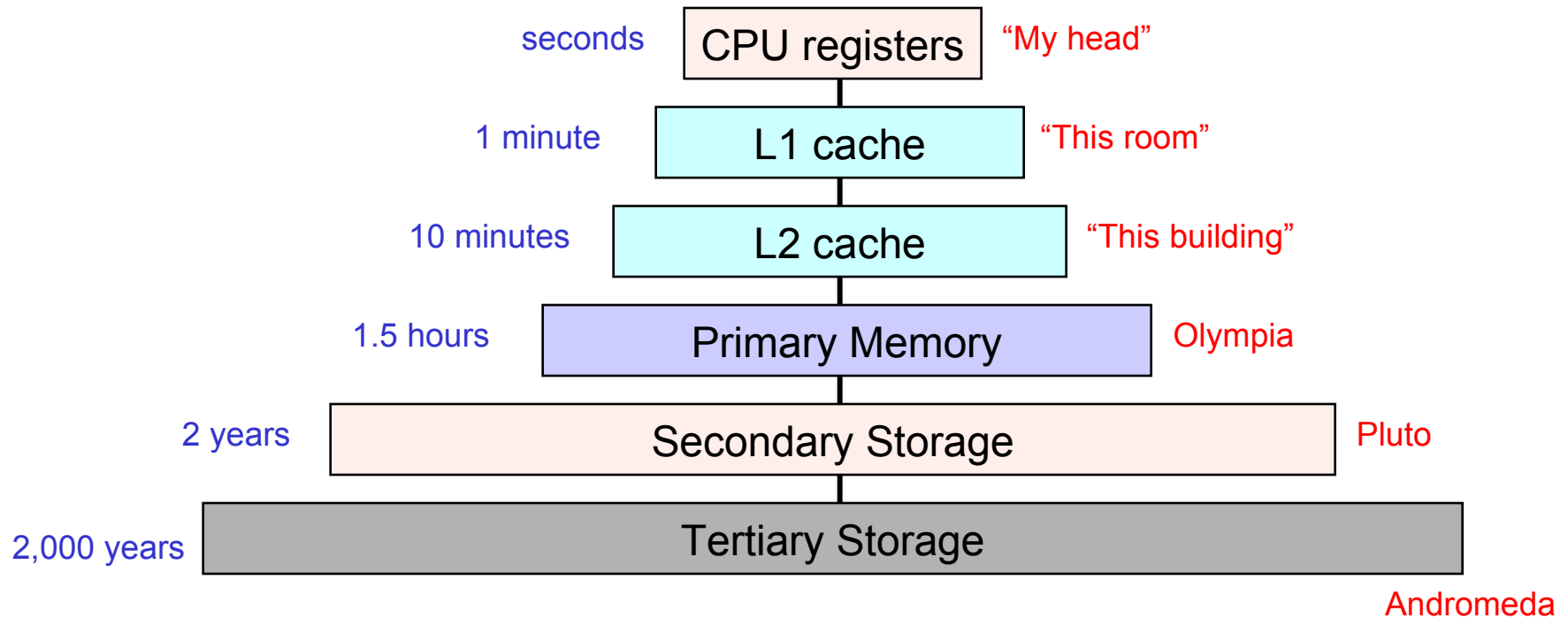 - 10x as fast as processor performance!

- Only a few years ago, we purchased disks by the megabyte (and it hurt!)
- Today, 1 GB (a billion bytes) costs $1 $0.50 from Dell (except you have to buy in increments of 40 80 GB)
  - => 1 TB costs $1K $500, 1 PB costs $1M $500K
- In 3 2 years, 1 GB will cost $.10
  - => 1 TB for $100, 1 PB for $100K

Memory hierarchy

100 bytes	CPU registers	< 1 ns
32KB	L1 cache	1 ns
256KB	L2 cache	4 ns
1GB	Primary Memory	60 ns
100GB	Secondary Storage	10+ ms
1-1000TB	Tertiary Storage	1s-1hr

- Each level acts as a cache of lower levels

Memory hierarchy: distance analogy

|		
seconds	CPU registers	"My head"
1 minute	L1 cache	"This room"
10 minutes	L2 cache	"This building"
1.5 hours	Primary Memory	Olympia
2 years	Secondary Storage	Pluto
2,000 years	Tertiary Storage	

Andromeda

Disks and the OS

- Disks are messy, messy devices
  - errors, bad blocks, missed seeks, etc.
- Job of OS is to hide this mess from higher-level software
  - low-level device drivers (initiate a disk read, etc.)
  - higher-level abstractions (files, databases, etc.)
- OS may provide different levels of disk access to different clients
  - physical disk block (surface, cylinder, sector)
  - disk logical block (disk block #)
  - file logical (filename, block or record or byte #)

Physical disk structure

- Disk components
  - platters
  - surfaces
  - tracks
  - sectors
  - cylinders
  - arm
  - heads

track

sector

surface

cylinder

platter

arm

head

© 2010 Gribble, Lazowska, Levy, Zahorjan
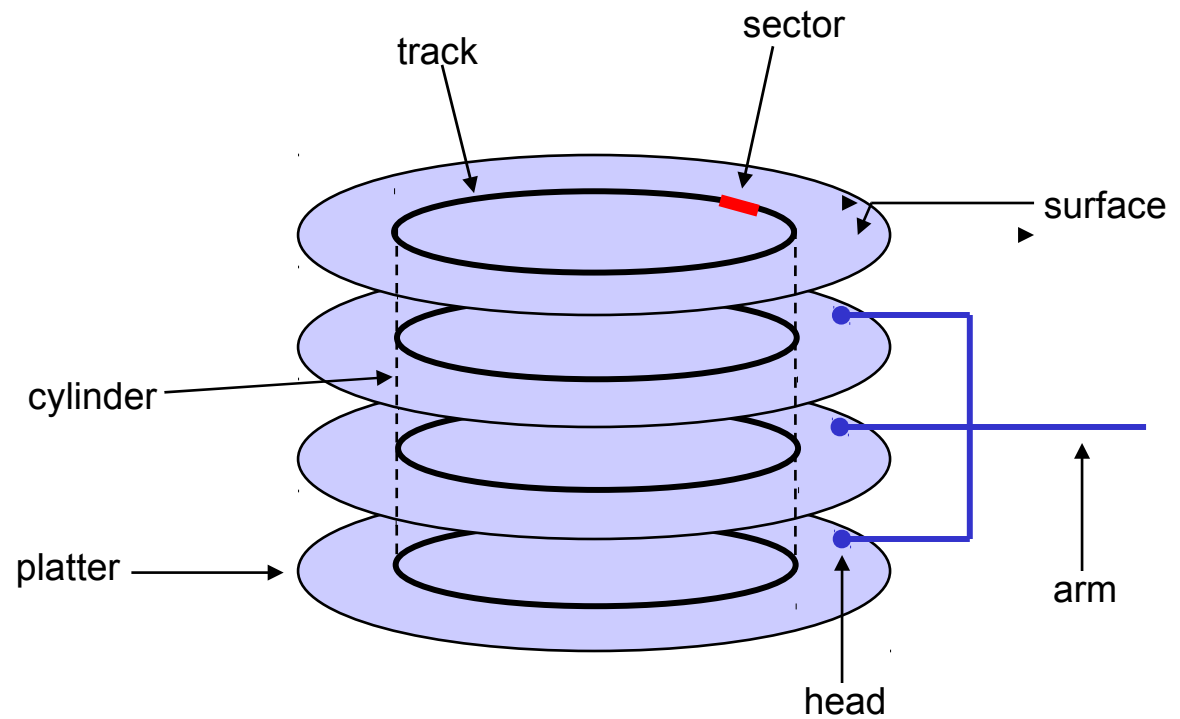
Disk performance

- Performance depends on a number of steps
  - seek: moving the disk arm to the correct cylinder
    - depends on how fast disk arm can move
      - seek times aren't diminishing very quickly (why?)
  - rotation (latency): waiting for the sector to rotate under head
    - depends on rotation rate of disk
      - rates are increasing, but slowly (why?)
  - transfer: transferring data from surface into disk controller, and from there sending it back to host
    - depends on density of bytes on disk
      - increasing, and very quickly
- When the OS uses the disk, it tries to minimize the cost of all of these steps
  - particularly seeks and rotation

Disk scheduling

- Seeks are very expensive, so the OS attempts to schedule disk requests that are queued waiting for the disk
  - FCFS (do nothing)
    - reasonable when load is low
    - long waiting time for long request queues
  - SSTF (shortest seek time first)
    - minimize arm movement (seek time), maximize request rate
    - unfairly favors middle blocks
  - SCAN (elevator algorithm)
    - service requests in one direction until done, then reverse
    - skews wait times non-uniformly (why?)
  - C-SCAN
    - like scan, but only go in one direction (typewriter)
    - uniform wait times

Interacting with disks

- In the old days...
  - OS would have to specify cylinder #, sector #, surface #, transfer size
    - i.e., OS needs to know all of the disk parameters

- Modern disks are even more complicated
  - not all sectors are the same size, sectors are remapped, ...
  - disk provides a higher-level interface, e.g., SCSI
    - exports data as a logical array of blocks [0 ... N]
    - maps logical blocks to cylinder/surface/sector
    - OS only needs to name logical block #, disk maps this to cylinder/surface/sector
    - on-board cache
    - as a result, physical parameters are hidden from OS
      - both good and bad