Lecture 18: Memory Systems

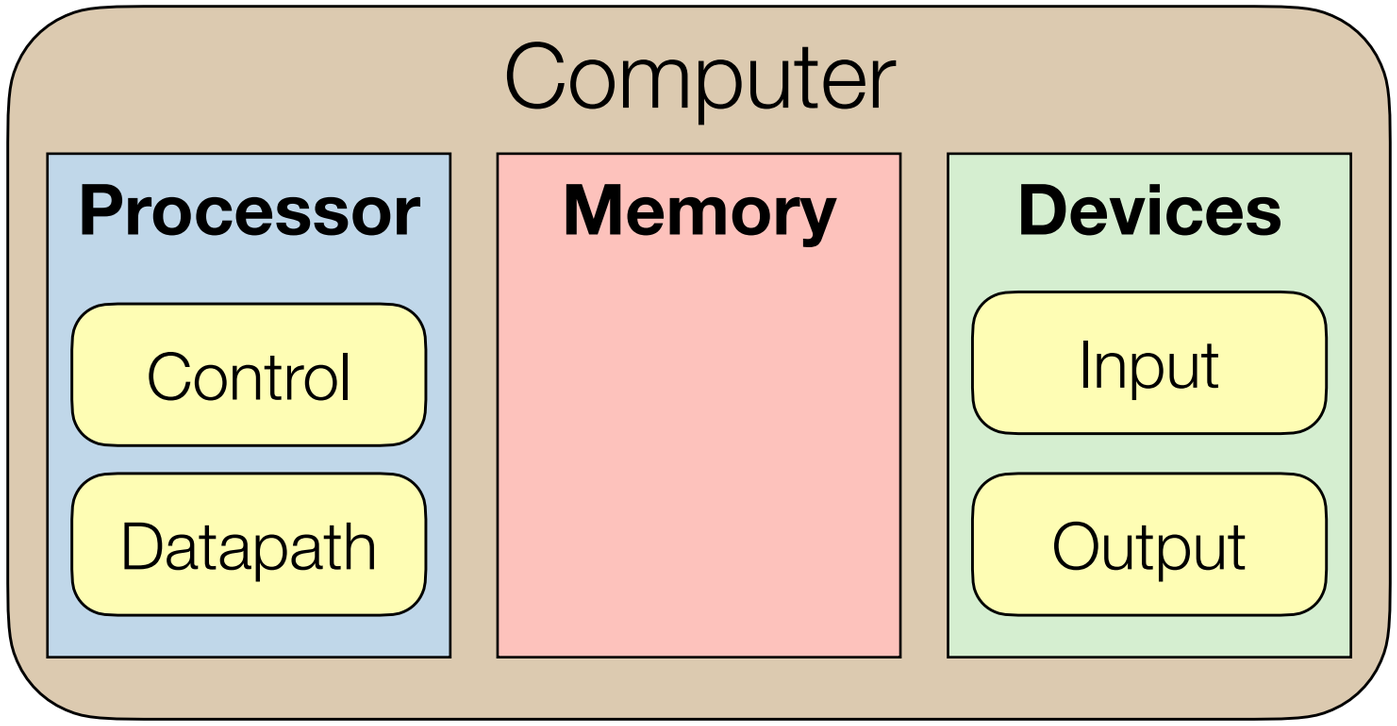
Fall 2023

Jason Tang

Topics

- Memory hierarchy
- Memory operations
- Cache basics

Computer Organization



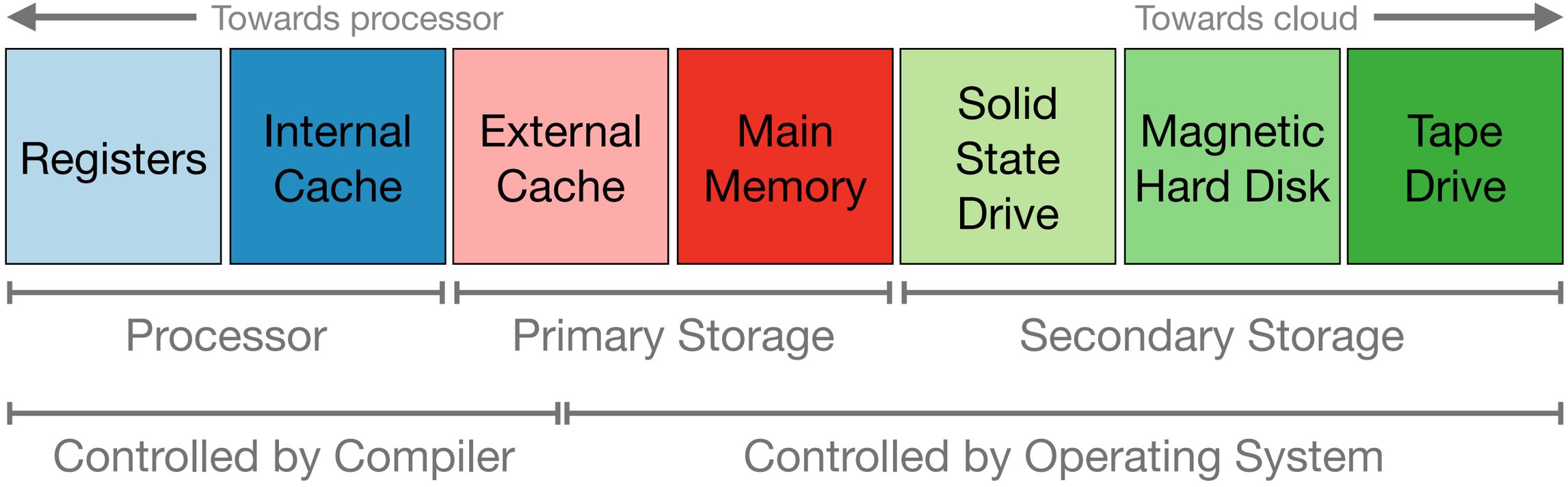
- So far, discussion has focused on processor design
- Memory system is not just a bunch of flip-flops
  - Its design greatly affects system performance

Memory Hierarchy

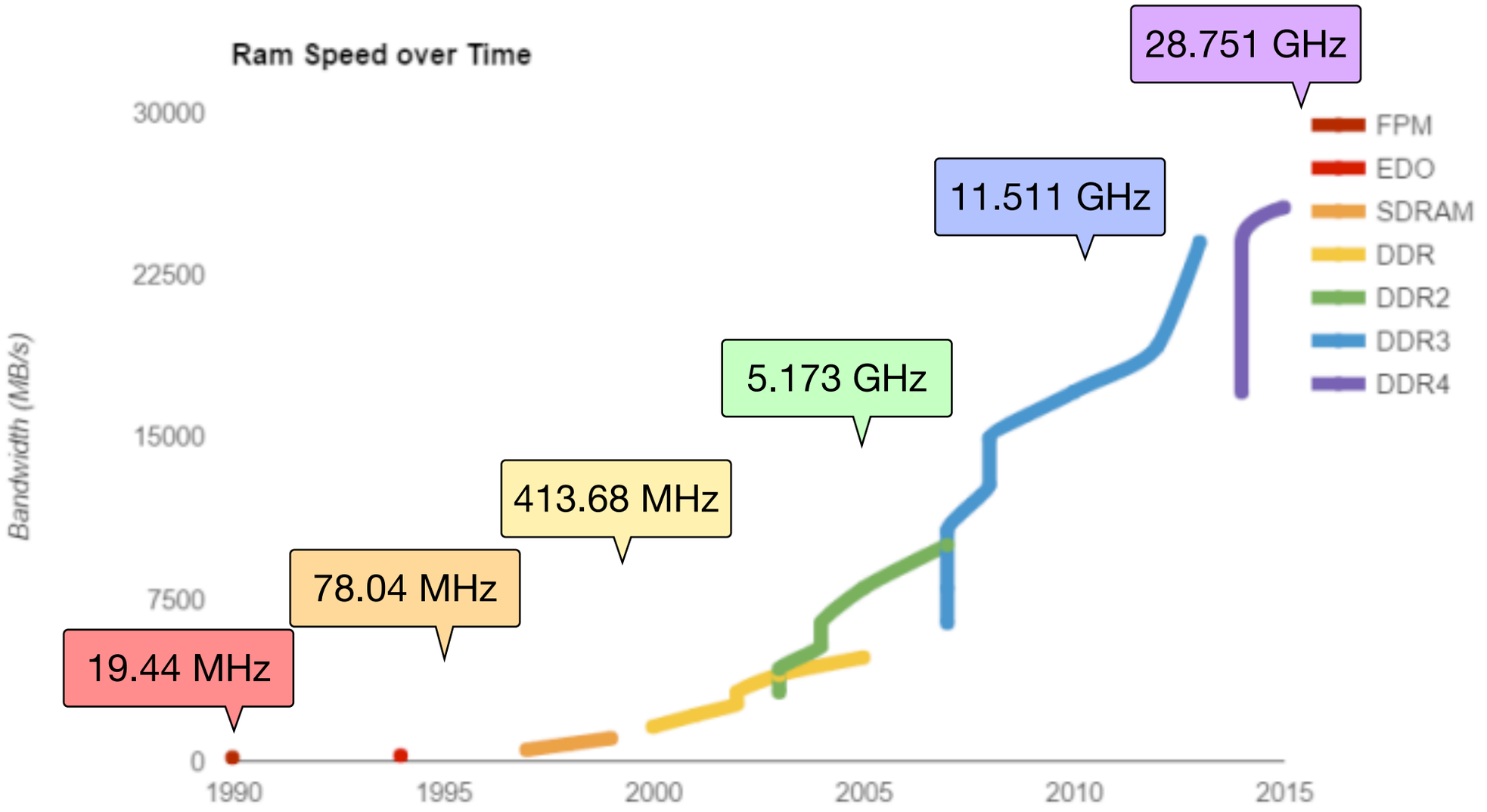
- Multiple memory systems on a modern computer

Faster
Smaller
More Expensive

Slower
Larger
Less Expensive



Processor to Memory Performance



- Processor-Memory performance gap has grown over years

- Memory is bottleneck for many operations

Graph from https://blog.logicalincrements.com/2016/03/ultimate-guide-computer-ram/; CPU frequencies from https://www.singularity.com/charts/page61.html

Memory Terminology

- Temporal Locality: Data recently accessed are likely to be accessed again
- Spatial Locality: If some data are accessed, nearby data are likely to be accessed next
- Line: Size of a data block at a particular memory level
- Hit: Data appear in a some line for a particular memory level
  - Hit Rate: Ratio of accesses resolved by that memory level
  - Hit Time: Time to access memory, equal to Memory Access Time + Time to determine hit/miss

Memory Technology Types

- Random Access: access time is [nearly] same for any location, regardless of previous access
    - SRAM: Static Random Access Memory
    - DRAM: Dynamic Random Access Memory
    - NVRAM: Non-Volatile Random Access Memory
- Linear Access: access time varies from location to location
    - “Almost-Linear”: can jump to a location, and then linear access
    - Sequential Access: must access memory purely linearly

Examples of Memory Technologies

Memory Technology	Examples
ROM	PROM, EPROM, Bootstrap
SRAM	CPU Registers, Cache
DRAM	DDR, GPU DDR
NVRAM	SSD, NVMe, SD, eMMC, Firmware
Magnetic	Floppy Drive, Hard Disk
Optical	CD, DVD, Blu-ray
Sequential Access	Tape drive, Paper tape

SRAM

- Basic flip-flop, using six to eight transistors per bit
- Very fast, very small
  - Access time is measured in picoseconds
  - Instead of accessing a single bit at a time, SRAM bytes are accessed a row at a time
- Low power, but volatile
  - Consumes little power to maintain memory while system is idle

http://www.radio-electronics.com/info/data/semicond/memory/sram-memory-basics-tutorial.php

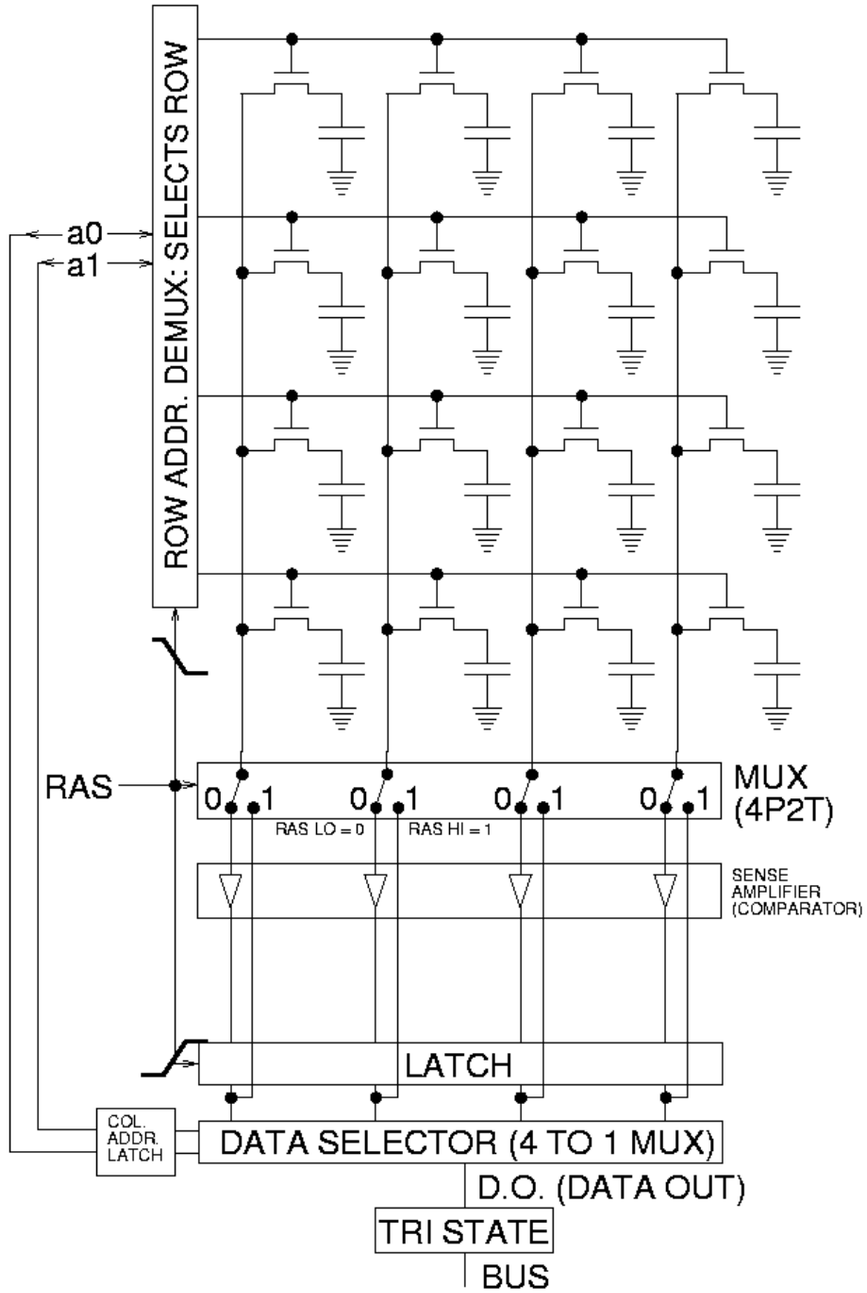
DRAM

- Memory bit stored in a capacitor
    - Because capacitor discharges over time, it must be periodically refreshed (hence “dynamic” RAM)
- Simpler than SRAM, but slower (measured in nanoseconds) and uses more power
    - Every time a bit is read, its capacitor is discharged and thus the read value must be rewritten back to that capacitor
    - Separate DRAM clock responsible for refreshing all of DRAM every (often) 64 milliseconds, either all at once, or one row at a time staggered
        - During a refresh, access to that DRAM is *stalled*

http://www.radio-electronics.com/info/data/semicond/memory/dram-technology-basics-tutorial.php

DRAM Organization

- During a read, a row address strobe (RAS) discharges the capacitors selected by the row address line; the resulting values are stored into a row buffer
  - That row buffer is rewritten back to DRAM
- Column address then selects which of those bits in the row buffer to then return to the processor
- Modern DRAM consist of multiple parallel banks to handle simultaneous requests



https://en.wikipedia.org/wiki/Dynamic_random-access_memory

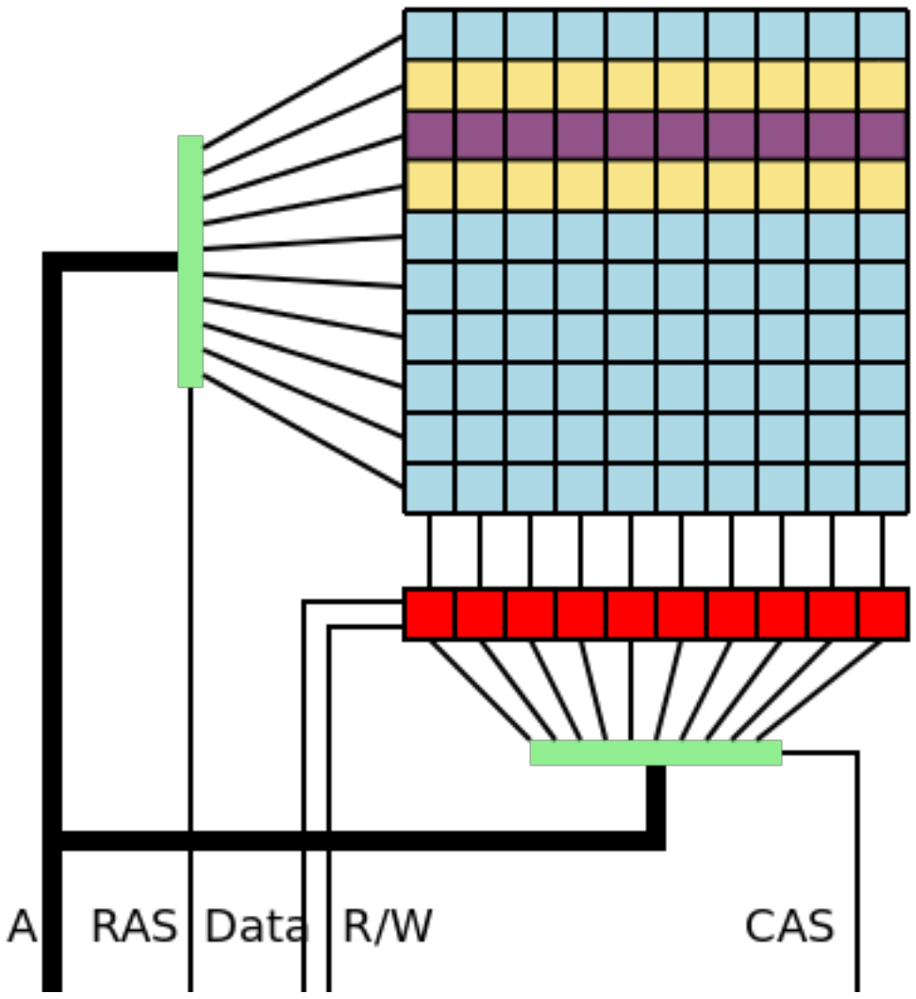
Rowhammer Attack

- DRAM capacitors can leak charges to nearby rows, especially given modern densely packaged DRAM

- When two rows are constantly refreshed, the row in between could have its cells' voltages change

  - If that victim row is not refreshed soon enough, then a disturbance error occurs

  - This results in an arbitrary write to memory, bypassing software security schemes



https://en.wikipedia.org/wiki/Row_hammer

NVRAM

- Heavily used in modern systems:
  - For system execution, in firmware and firmware settings
  - As secondary storage, in solid-state drives (SSD), USB flash drives, and SD cards
- Two variants: NAND flash (more common) and NOR flash
- Block erasure: must erase entire block at a time
- Memory wear: may only erase the same block finite times

NAND versus NOR Flash

- NOR flash has random byte-level access, is faster to read, but slower to erase
  - Good for mostly read-only operations, like storing code
- NAND flash is cheaper and smaller than NOR flash, but must read and program an entire row at a time
  - Good for general data storage
- Flash memory is also susceptible to Rowhammer-type attacks

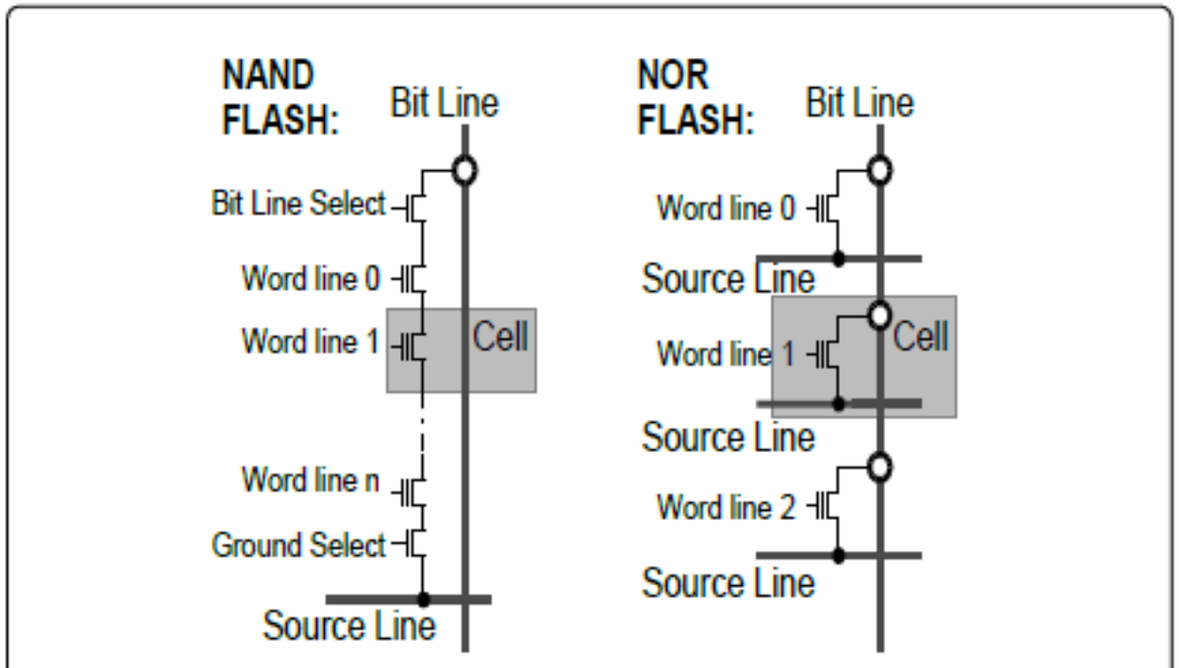


Figure 2: NAND Flash and NOR Flash memory cells arrangement.

https://www.omicsonline.org/open-access/nand-flash-memory-organization-and-operations-2165-7866-1000139.php?aid=46500

Magnetic Hard Disk

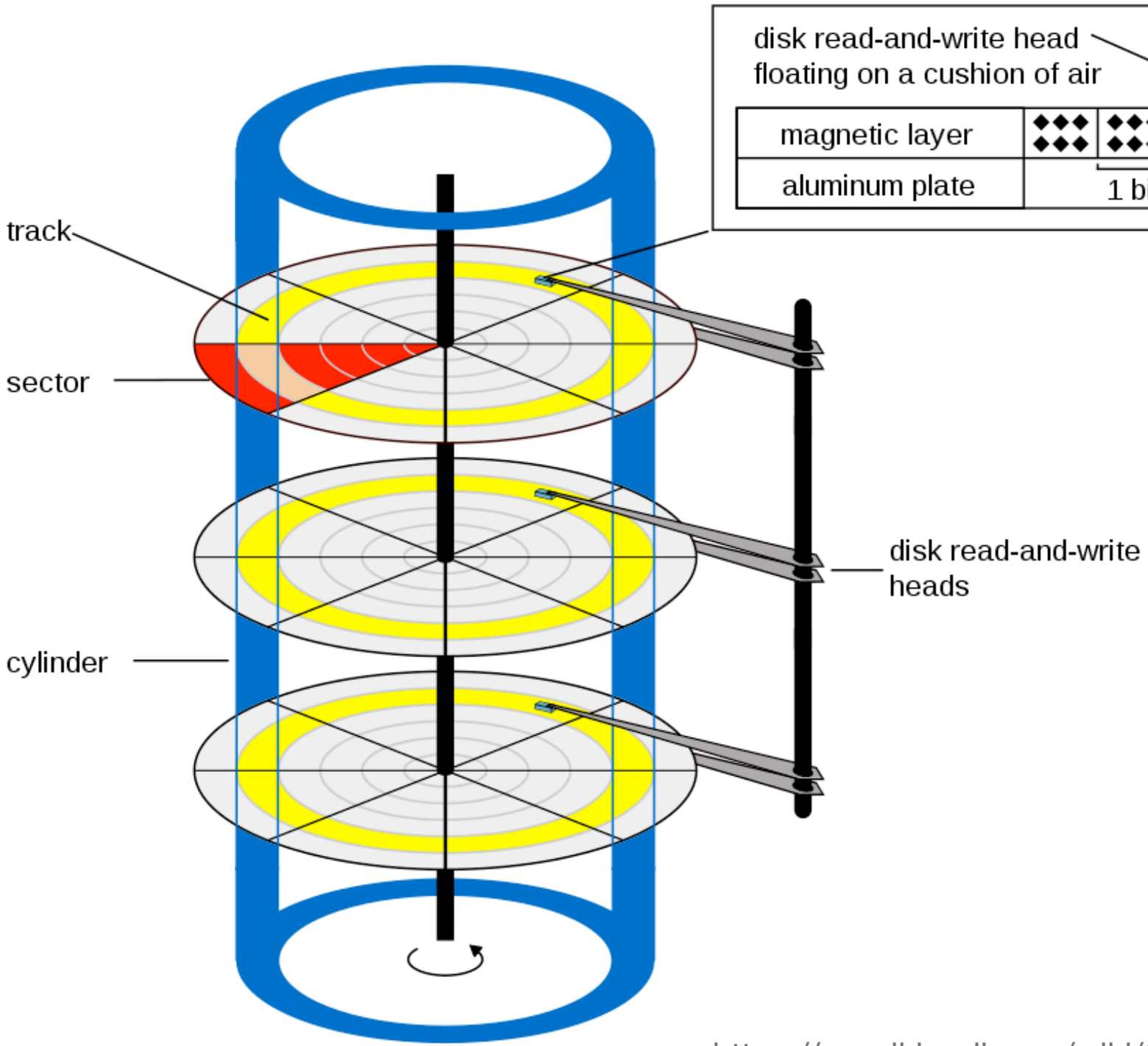
- Spindle motor spins a stack of platters coated with magnetic material
  - Spins from 5400 to over 10000 RPMs
- Actuator motor moves a disk head over the platters, to sense polarity of the track underneath

https://www.technobuffalo.com/2012/11/24/western-digital-expands-high-performance-wd-black-hard-drive-line-to-4tb/

Magnetic Hard Disk

- Cylinder: All of the tracks under the disk read heads for all platters
- Sector: A fixed-size subdivision of a track, often 512 bytes
- Positioning time (random-access time): time to move disk arm to desired cylinder (seek time) plus time for desired sector to rotate under disk head (rotational latency)
  - Measured in milliseconds
- Transfer rate: rate which data flow between drive and computer
  - Modern SATA III transfer rate is 6 Gb/sec

Hard Drive Geometry



https://en.wikipedia.org/wiki/Cylinder-head-sector

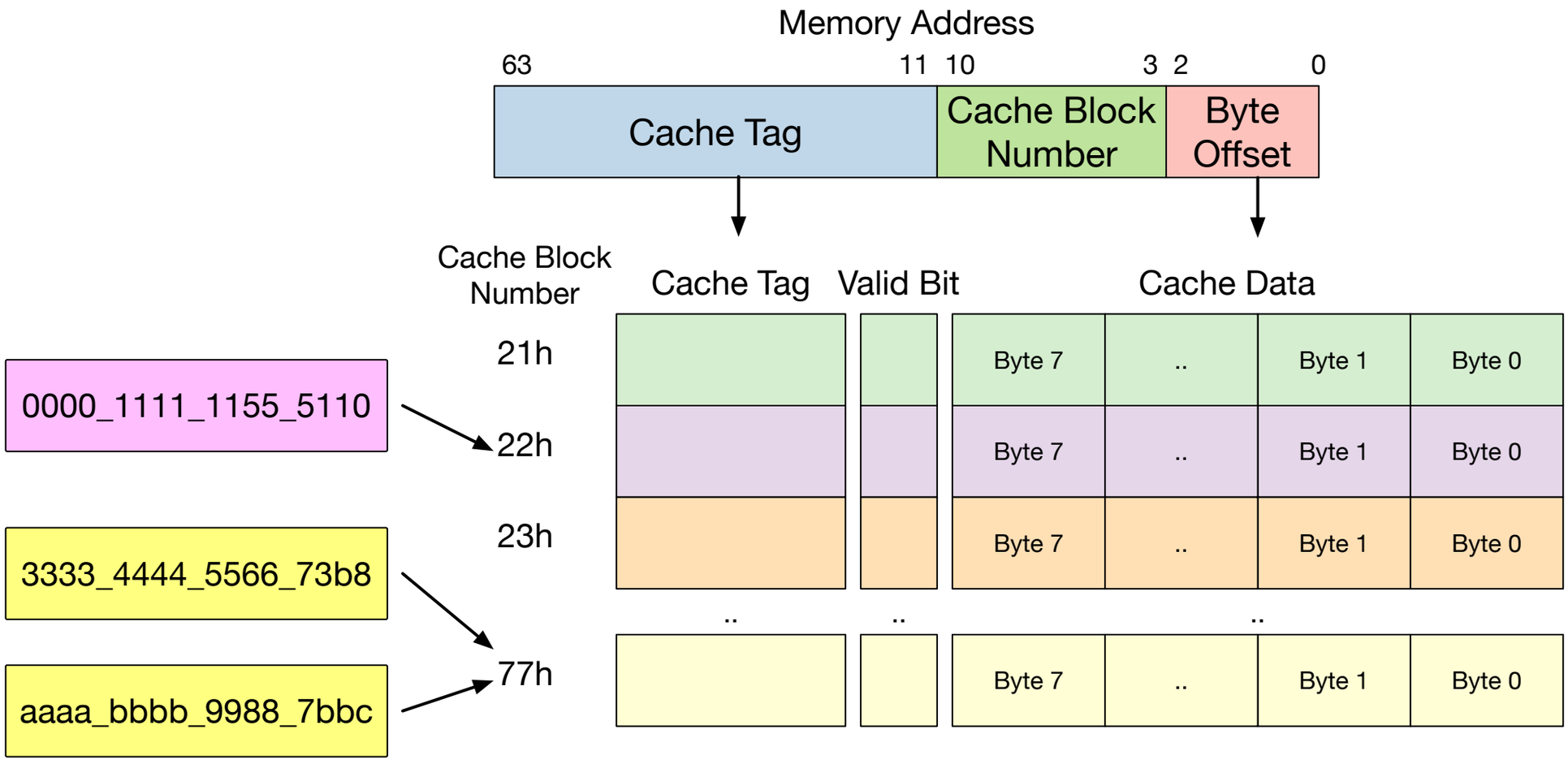
Caching

- Technically, a cache is any memory device that stores a **subset** of a slower memory system in a faster memory system
    - Example: frequently accessed data files are stored in the SSD of a hybrid disk system
    - Example: frequently read files are stored in a ramdisk
    - Example: frequently accessed memory bytes are stored in processor cache
- “Cache” for this class refers to the processor cache
    - Constructed as a series of blocks

Cache Basics

- Upon a request for some bit of memory, cache is first checked
- If requested address is in cache *and that address is* valid, then data are returned (a hit)
- Otherwise, target data are brought from main memory into cache (miss)

Direct Mapped Cache



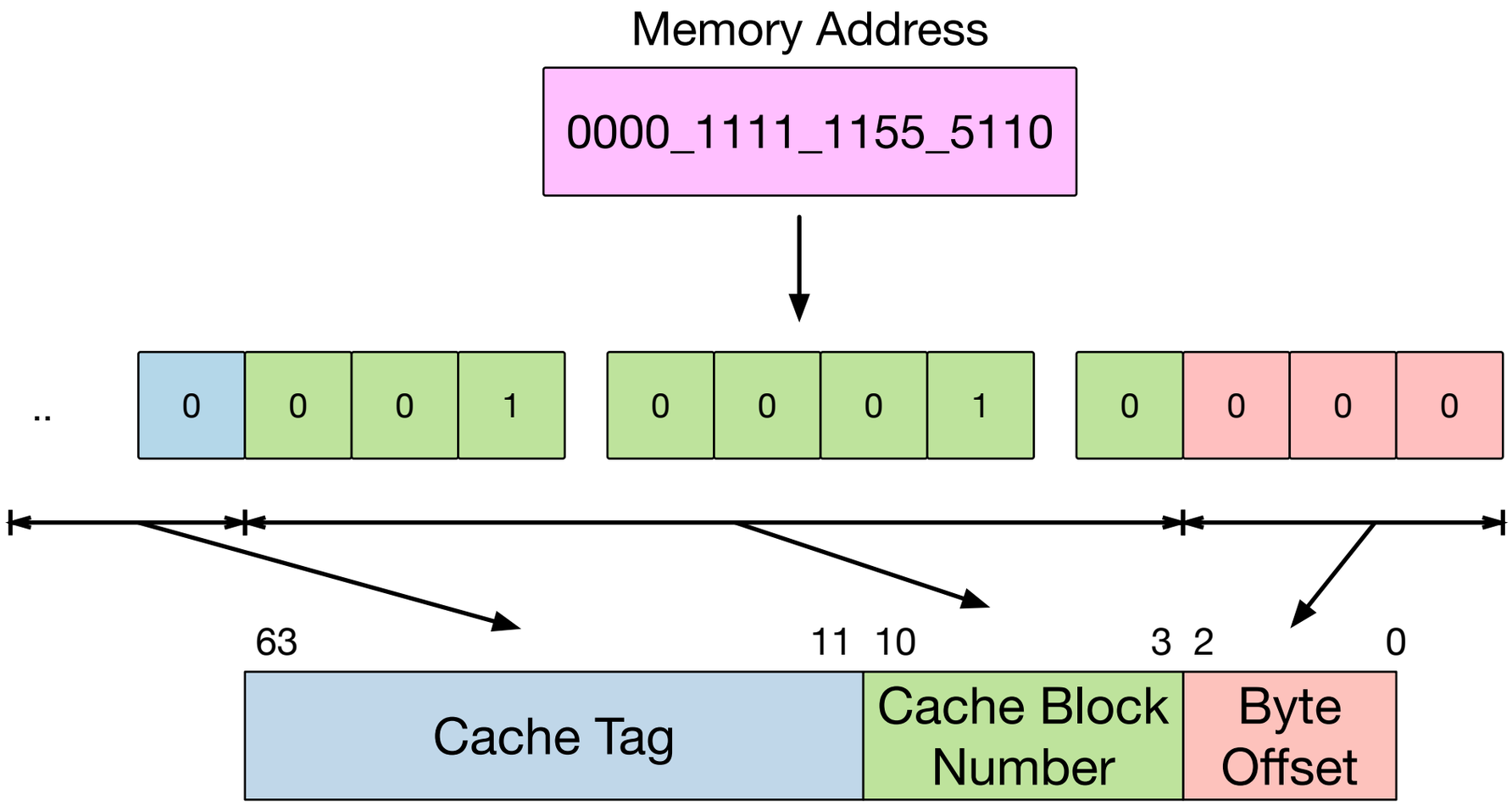
64-bit addressing, 256 cache blocks, 64-bit cache lines

- Simplest cache is direct mapped
  - Each cache location holds a single value and corresponds to a memory address: (block address) modulus (number of blocks in cache)

Cache Metadata

- Index: resulting cache block number given an address
  - For direct-mapped cache, the address's lower **n** bits give cache block number
  - Mathematically, $block\ number = floor(\frac{address}{block\ size})$
- When multiple addresses map to same index, need some way to distinguish which particular memory address is being cached
  - Tag: extra bits record which particular address is held
  - Valid bit: bit indicating if tag holds a valid address

Calculating Cache Block Number



Building a Direct Cache

- Cache size depends upon number of cache blocks, address length, data word size

- Example: For this cache, each cache block has:

  - 53 tag bits, 1 valid bit, 64 data bits

  - 256 cache blocks

  - Total cache size is 30208 bits, of which 16384 bits (2048 bytes) are cached