ECE/CS 552: Cache Concepts

Lecture notes based in part on slides created by Mark Hill, David Wood, Guri Sohi, John Shen and Jim Smith

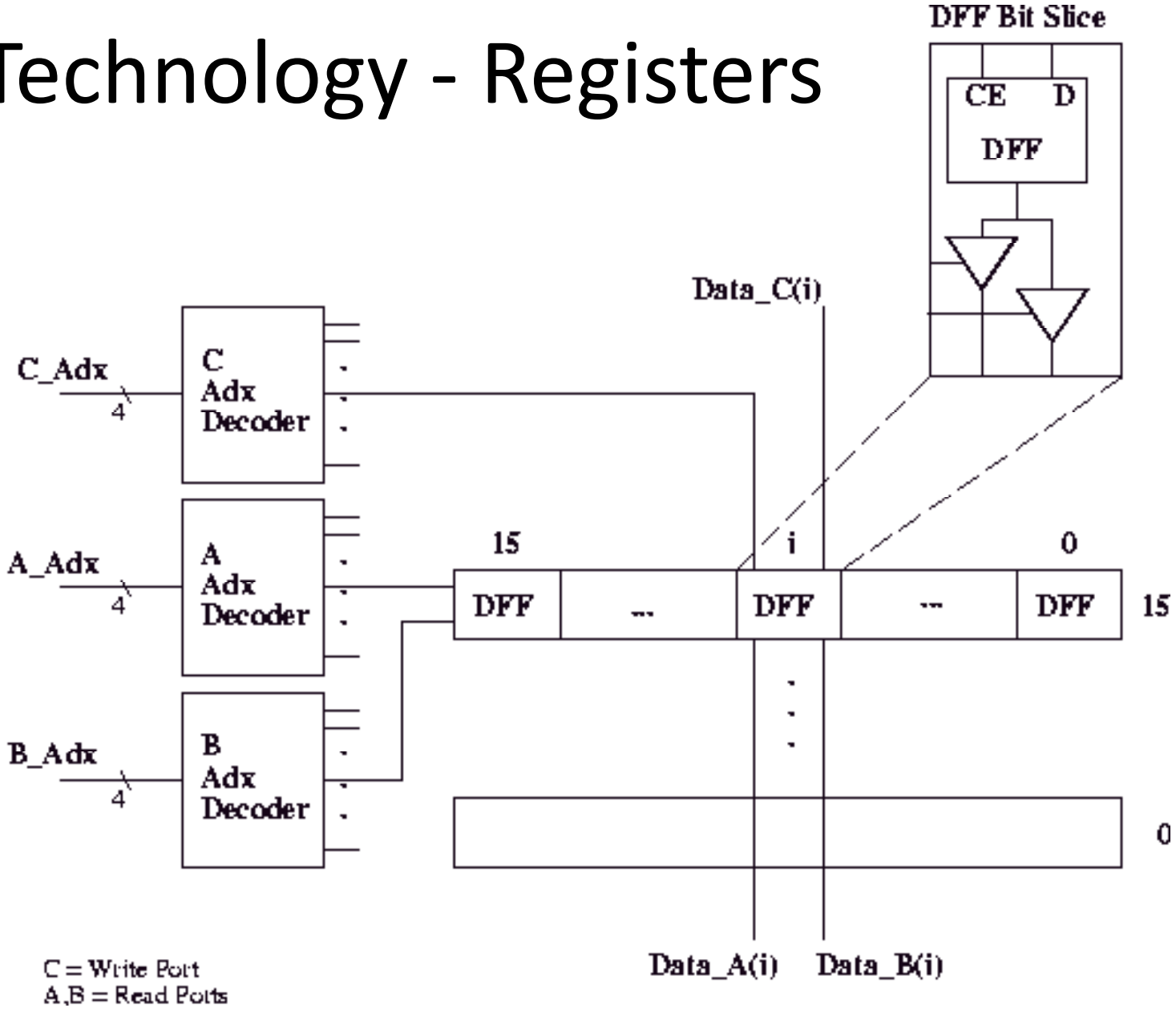
Big Picture

- Memory
  - Just an “ocean of bits”
  - Many technologies are available
- Key issues
  - Technology (how bits are stored)
  - Placement (where bits are stored)
  - Identification (finding the right bits)
  - Replacement (finding space for new bits)
  - Write policy (propagating changes to bits)
- Must answer these regardless of memory type

Types of Memory

Type	Size	Speed	Cost/bit
Register	< 1KB	< 1ns	$$$$
On-chip SRAM	8KB-6MB	< 10ns	$$$
Off-chip SRAM	1Mb – 16Mb	< 20ns	$$
DRAM	64MB – 1TB	< 100ns	$
Disk	40GB – 1PB	< 20ms	~0

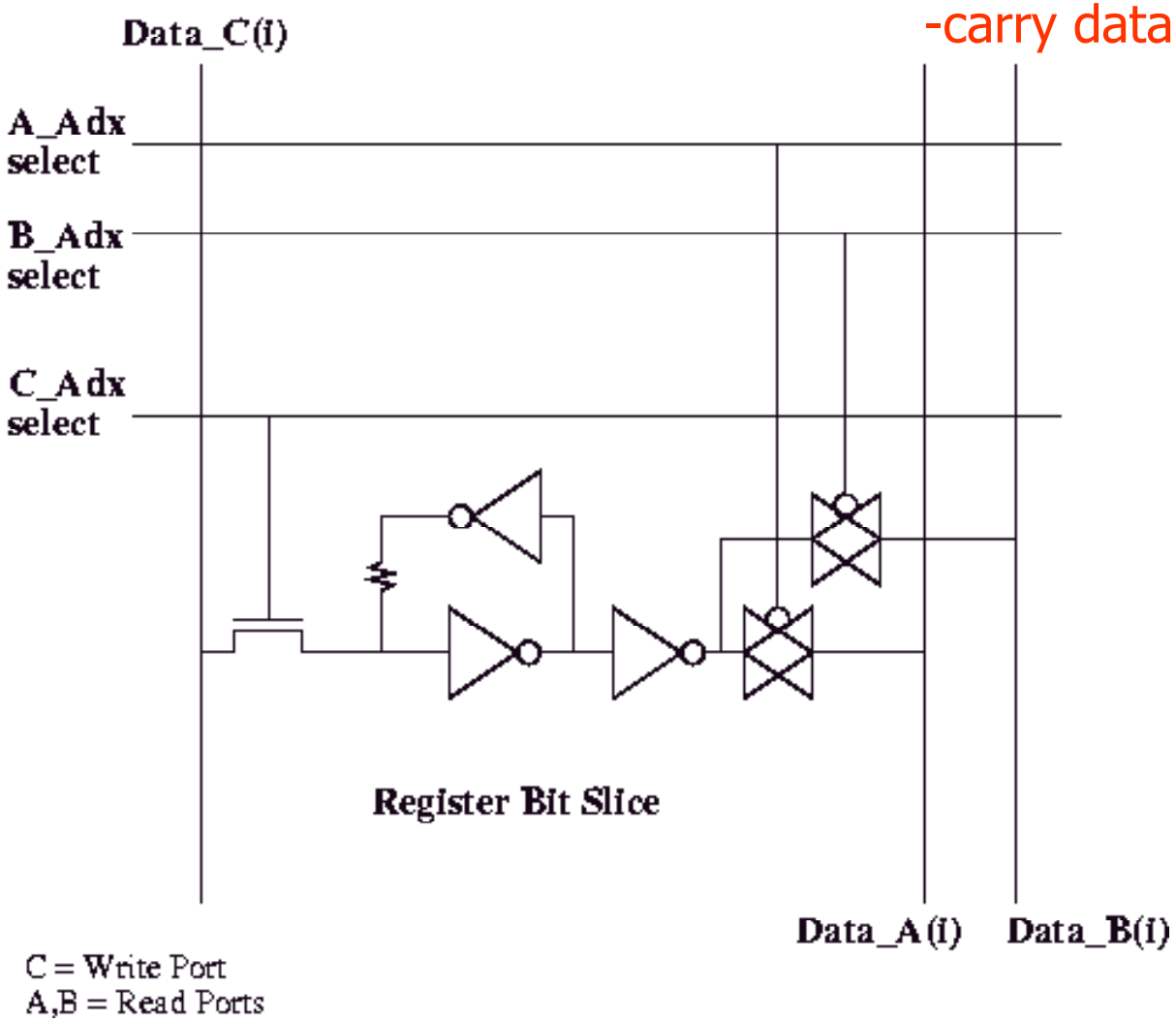
Technology - Registers



Technology – SRAM

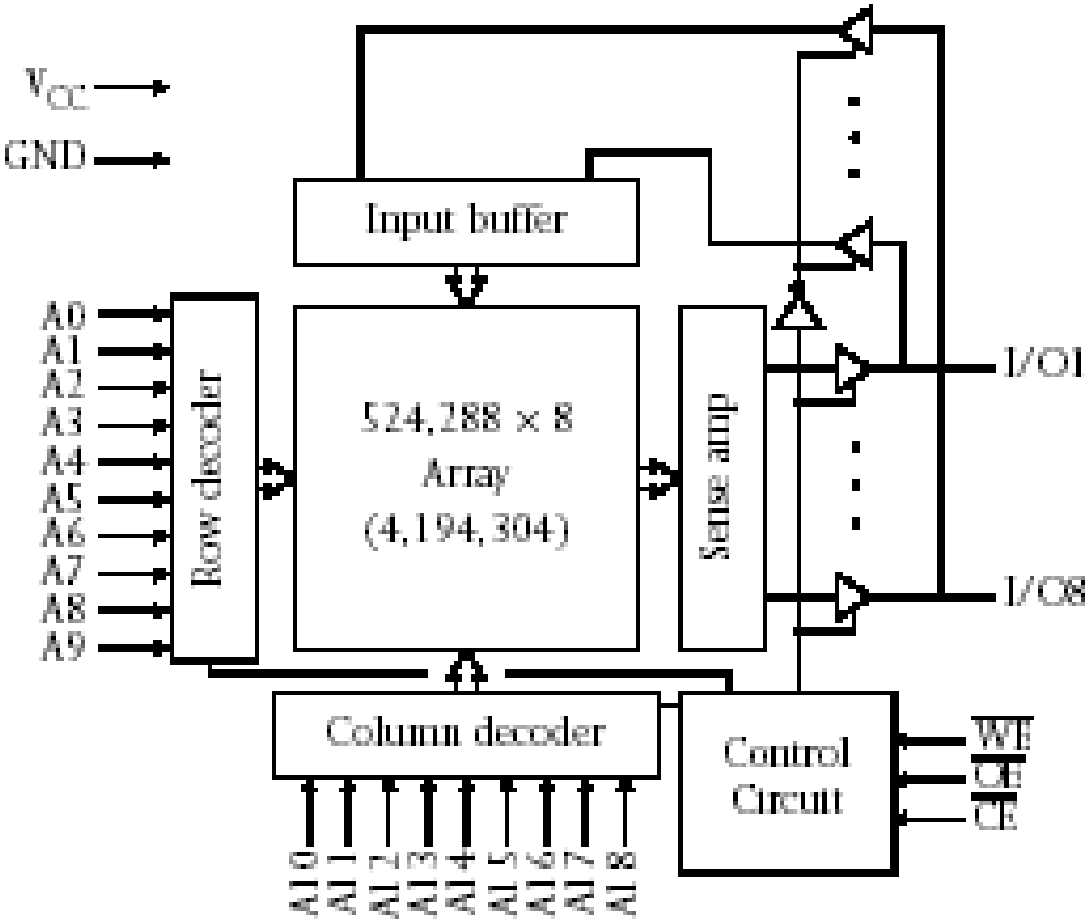
"Bit" Lines
-carry data in/out

"Word" Lines
-select a row



Technology – Off-chip SRAM

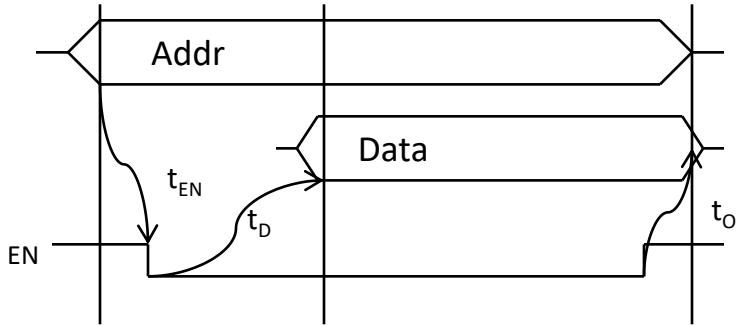
- Commodity
  - 1Mb – 16Mb
- 554 Lab SRAM
  - AS7C4096
  - 512K x 8b
    - 512KB
    - Or 4Mb
  - 10-20ns
- Note: sense amp
  - Analog circuit
  - High-gain amplifier

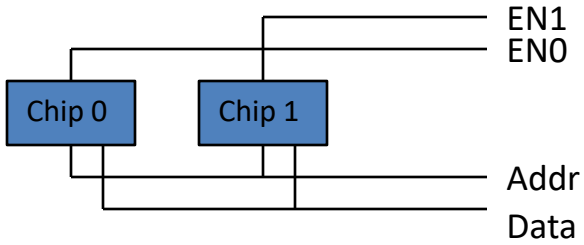


Technology – DRAM

- Logically similar to SRAM
- Commodity DRAM chips
  - E.g. 1Gb per chip
  - Standardized address/data/control interfaces
- Very dense
  - 1T per cell (bit)
  - Data stored in capacitor – decays over time
    - Must rewrite on read, refresh
- Density improving vastly over time
- Latency barely improving

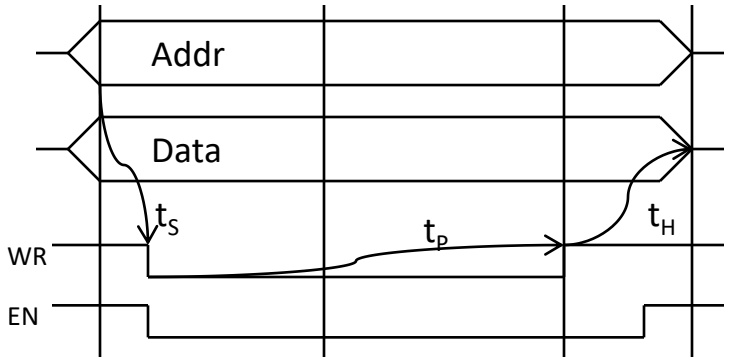
Memory Timing – Read

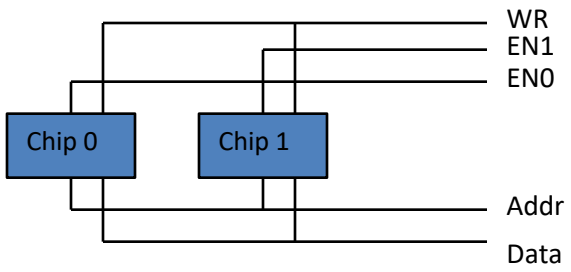




- Latch-based SRAM or asynchronous DRAM (FPM/EDO)
  - Multiple chips/banks share address bus and tristate data bus
  - Enables are decoded from address to select bank
    - E.g. bbbbbbb0 is bank 0, bbbbbbb1 is bank 1
- Timing constraints: straightforward
  - t_{EN} setup time from Addr stable to EN active (often zero)
  - t_D delay from EN to valid data (10ns typical for SRAM)
  - t_O delay from EN disable to data tristate off (nonzero)

Memory Timing - Write





- WR & EN triggers write of Data to ADDR
- Timing constraints: not so easy
  - t_S setup time from Data & Addr stable to WR pulse
  - t_P minimum write pulse duration
  - t_H hold time for data/addr beyond write pulse end
- Challenge: WR pulse must start late, end early
  - $>t_S$ after Addr/Data, $>t_H$ before end of cycle
  - Requires multicycle control and/or glitch-free clock divider

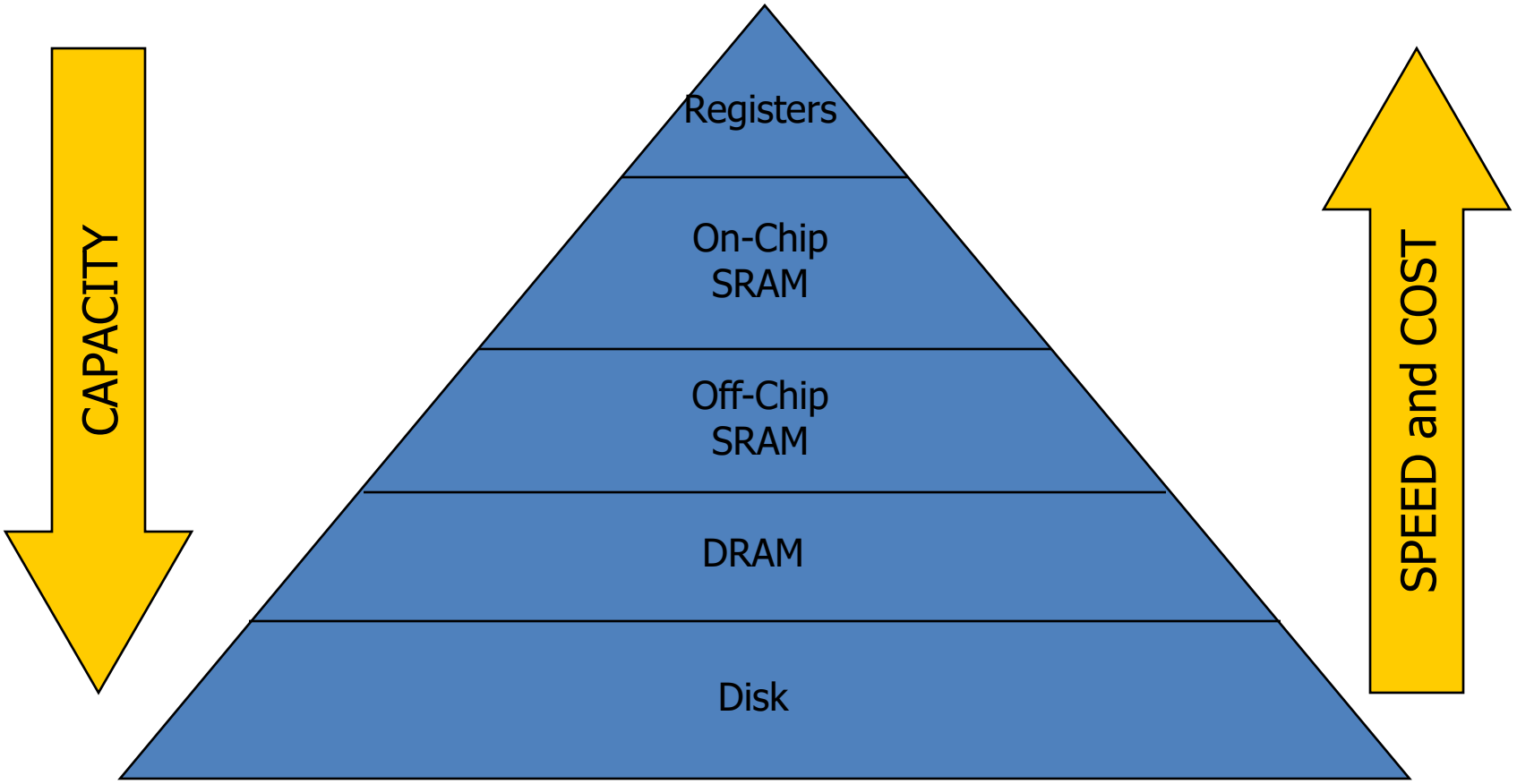
Technology – Flash

- Similar 2D array as SRAM/DRAM, but nonvolatile
  - Moving to 3D with stacked bit cells
- Bits stored as charge in floating gate
  - Coincident selection detects presence of charge
  - Multilevel cells (MLC) now common (not binary)
- Writes
  - Requires high voltage for writes to store charge
  - Writes are slow, done in bulk
  - Endurance issues – write leveling required in firmware

Technology – Disk

- Covered in more detail later (input/output)
- Bits stored as magnetic charge
- Still mechanical!
  - Disk rotates (3600-15000 RPM)
  - Head seeks to track, waits for sector to rotate to it
  - Solid-state replacements in the works
    - MRAM, etc.
- Glacially slow compared to DRAM (10-20ms)
- Density improvements astounding (100%/year)

Memory Hierarchy



Why Memory Hierarchy?

- Need lots of bandwidth

$$BW = \frac{1.0 inst}{cycle} \times \left[\frac{1 Ifetch}{inst} \times \frac{4B}{Ifetch} + \frac{0.4 Dref}{inst} \times \frac{4B}{Dref} \right] \times \frac{1 Gcycles}{\text{sec}}$$
$$= \frac{5.6 GB}{\text{sec}}$$

- Need lots of storage
  - 64MB (minimum) to multiple TB
- Must be cheap per bit
  - (TB x anything) is a lot of money!
- These requirements seem incompatible

Why Memory Hierarchy?

- Fast and small memories
  - Enable quick access (fast cycle time)
  - Enable lots of bandwidth (1+ L/S/I-fetch/cycle)
- Slower larger memories
  - Capture larger share of memory
  - Still relatively fast
- Slow huge memories
  - Hold rarely-needed state
  - Needed for correctness
- All together: provide appearance of large, fast memory with cost of cheap, slow memory

Why Does a Hierarchy Work?

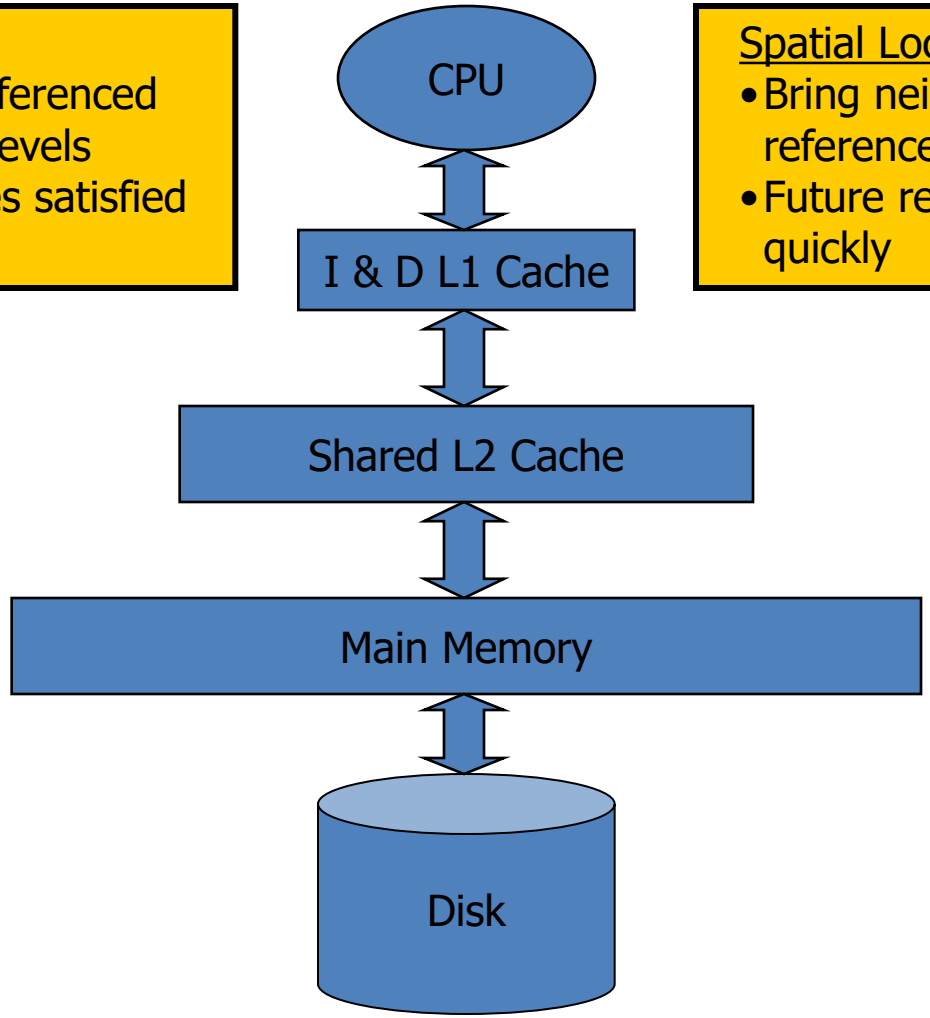
- Locality of reference
  - Temporal locality
    - Reference same memory location repeatedly
  - Spatial locality
    - Reference near neighbors around the same time
- Empirically observed
  - Significant!
  - Even small local storage (8KB) often satisfies >90% of references to multi-MB data set

Why Locality?

- Analogy:
  - Library (Disk)
  - Bookshelf (Main memory)
  - Stack of books on desk (off-chip cache)
  - Opened book on desk (on-chip cache)
- Likelihood of:
  - Referring to same book or chapter again?
    - Probability decays over time
    - Book moves to bottom of stack, then bookshelf, then library
  - Referring to chapter n+1 if looking at chapter n?

Memory Hierarchy

<u>Temporal Locality</u>
- Keep recently referenced items at higher levels
- Future references satisfied quickly



<u>Spatial Locality</u>
- Bring neighbors of recently referenced to higher levels
- Future references satisfied quickly

Summary

- Memory technology
- Memory hierarchy
- Temporal and spatial locality

- Cache design issues in following lecture