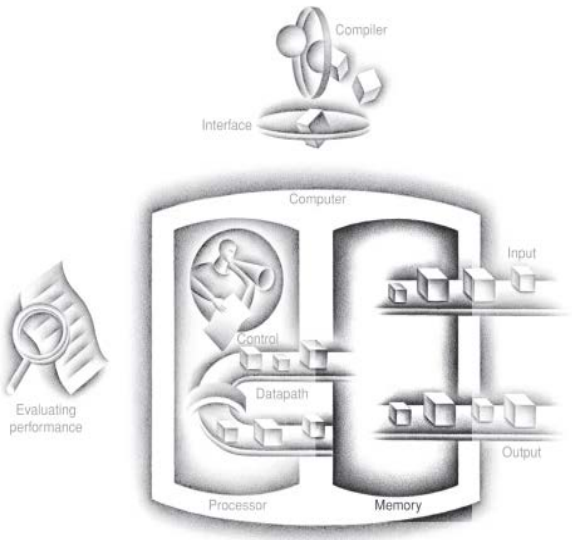
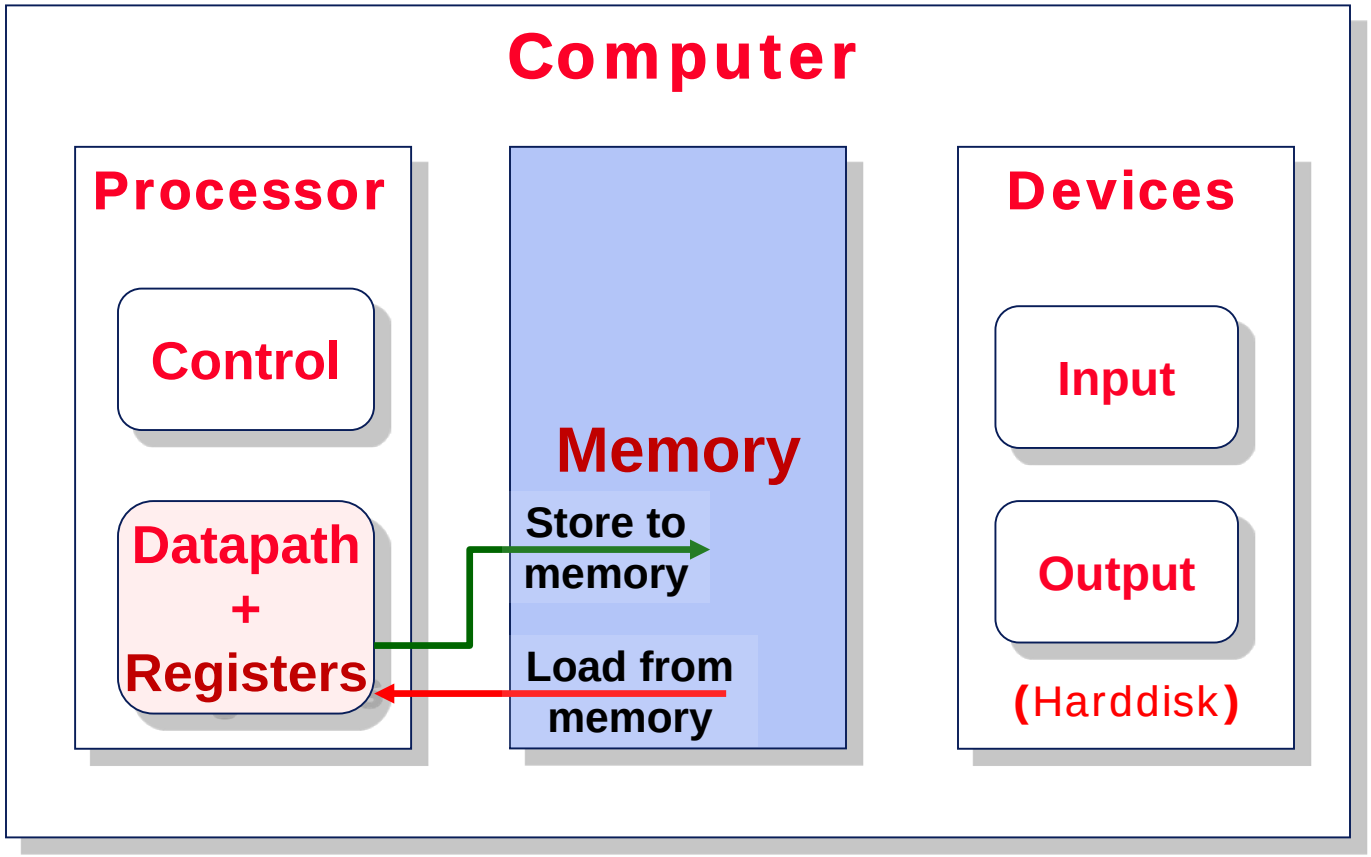
Chapter 5
Large and Fast : Exploiting the Memory Hierarchy

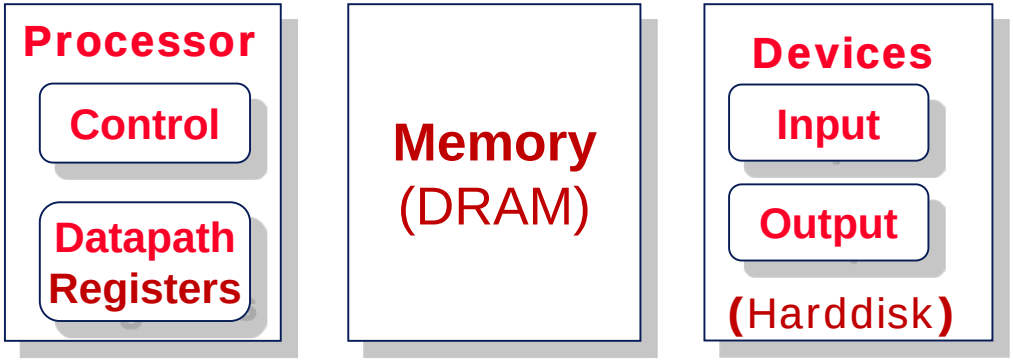


Data Transfer: The Big Picture



Registers are in the datapath of the processor. If operands are in memory we have to **load** them to processor (registers), operate on them, and **store** them back to memory.

Quality vs. Quantity



	Capacity	Latency	Cost/GB
Register	100s Bytes	20 ps	$$$$
SRAM	100s KB	0.5-5 ns	$$$
DRAM	100s MB	50-70 ns	$
Hard Disk	100s GB	5-20 ms	Cents
Ideal	**1 GB**	**1 ns**	**Cheap**

The “Memory Wall”

- Processor vs. DRAM speed disparity continues to grow

- What we want:
  - A **BIG** and **FAST** memory
  - Memory system should perform like 1GB of SRAM (1ns access time) but cost like 1GB of slow memory

Key concept:

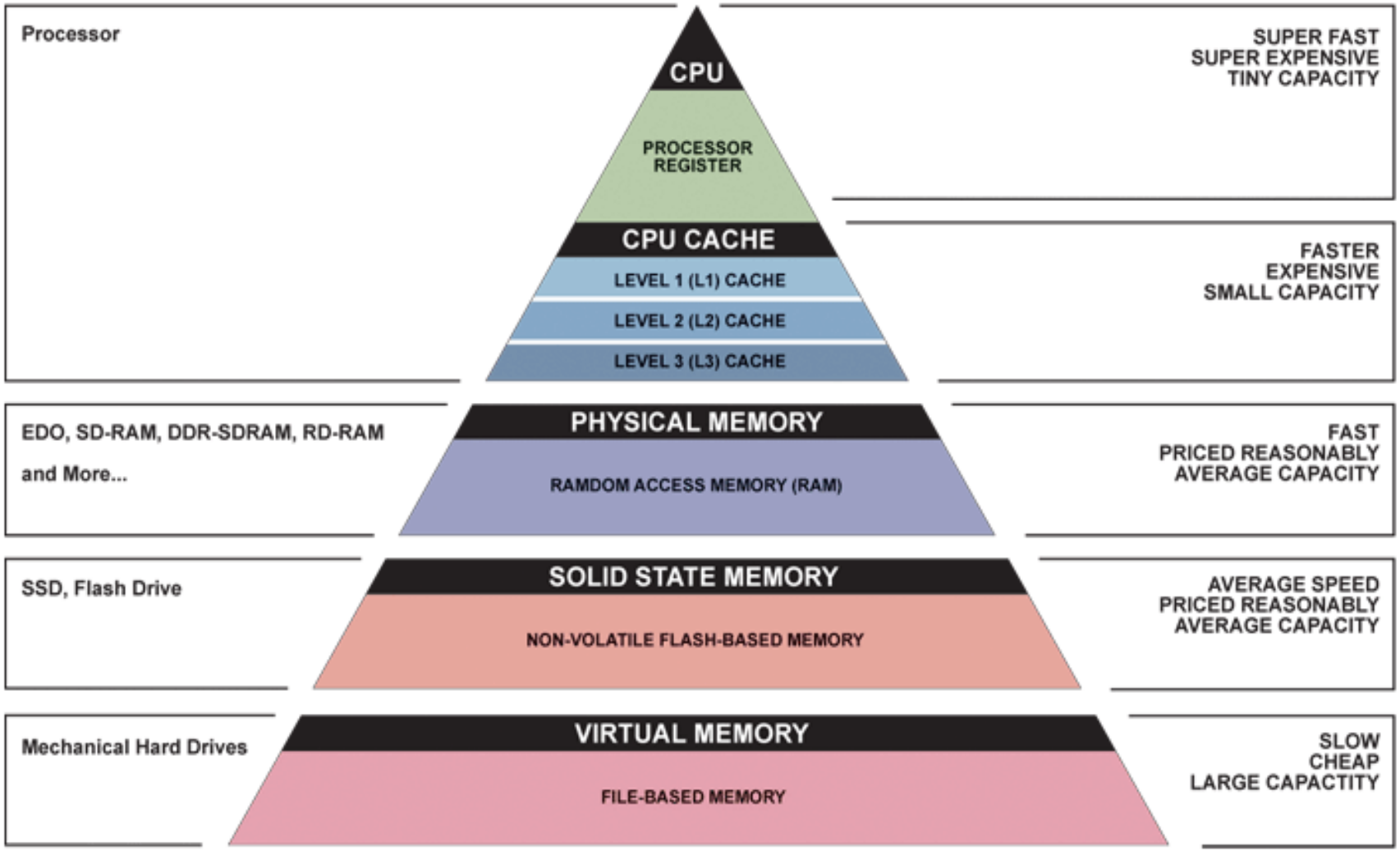
Use a hierarchy of memory technologies:

- Small but fast memory near CPU
- Large but slow memory farther away from CPU

- Good memory hierarchy (cache) design is increasingly important to overall performance
  - 1980: no cache in µproc,
  - 1995: 2-level cache on chip

1989: first Intel CPU with a cache on chip
2020 ?

Great Idea : Principle of Locality / Memory Hierarchy



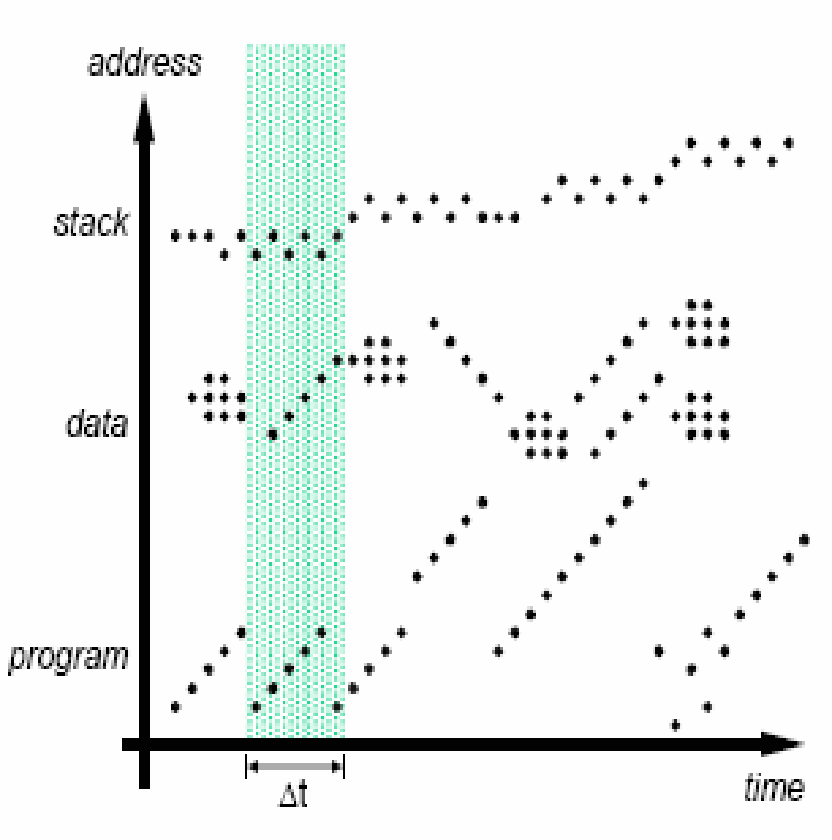
Principle of Locality (pp. 374)

```
int sum = 0;
int x[1000];

for(int c = 0; c < 1000; c++) {
  sum += c;

  x[c] = 0;
}
```

- "Programs will access **only a small proportion** of the address space **at any time**"
  - Not any place accessed randomly with equal probably!
- Temporal Locality (locality in time)
  - Items accessed are likely to be accessed again soon
  - e.g., instructions and data in a loop
- Spatial Locality (locality in space)
  - Items near the accessed item are likely to be accessed soon
  - e.g., sequential instruction access, an array of data

How to Take Advantage of Locality

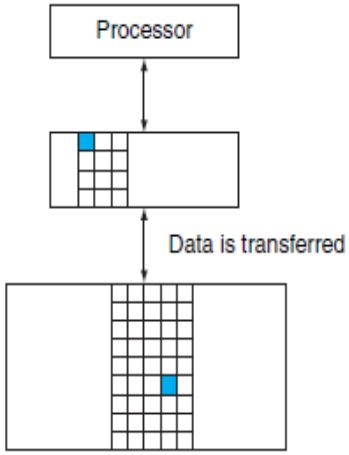
- *Keep most recently accessed data items closer to the processor*
  - Caches Main memory Hard Disk



Capture the working set and keep it in the memory closest to CPU

The Memory Hierarchy Levels : Terminology

- Block (or cache line): unit of copying
  - It has multiple words



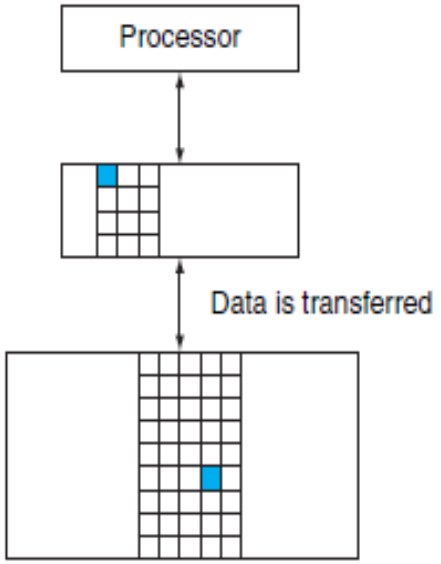
- If accessed data is present in upper level
  - Hit : access satisfied by upper level
  - Hit ratio = #hits / #accesses

- If accessed data is absent
  - Miss : block copied from lower level
  - Miss ratio = #misses / #accesses = 1 - Hit ratio
  - Time taken : Miss Penalty (Time to replace a block in that level with the corresponding block from a lower level)

- Hit Time << Miss Penalty

Common Framework for Memory Hierarchy

- Q1 block placement
- Q2 block identification
- Q3 block replacement
- Q4 write policy

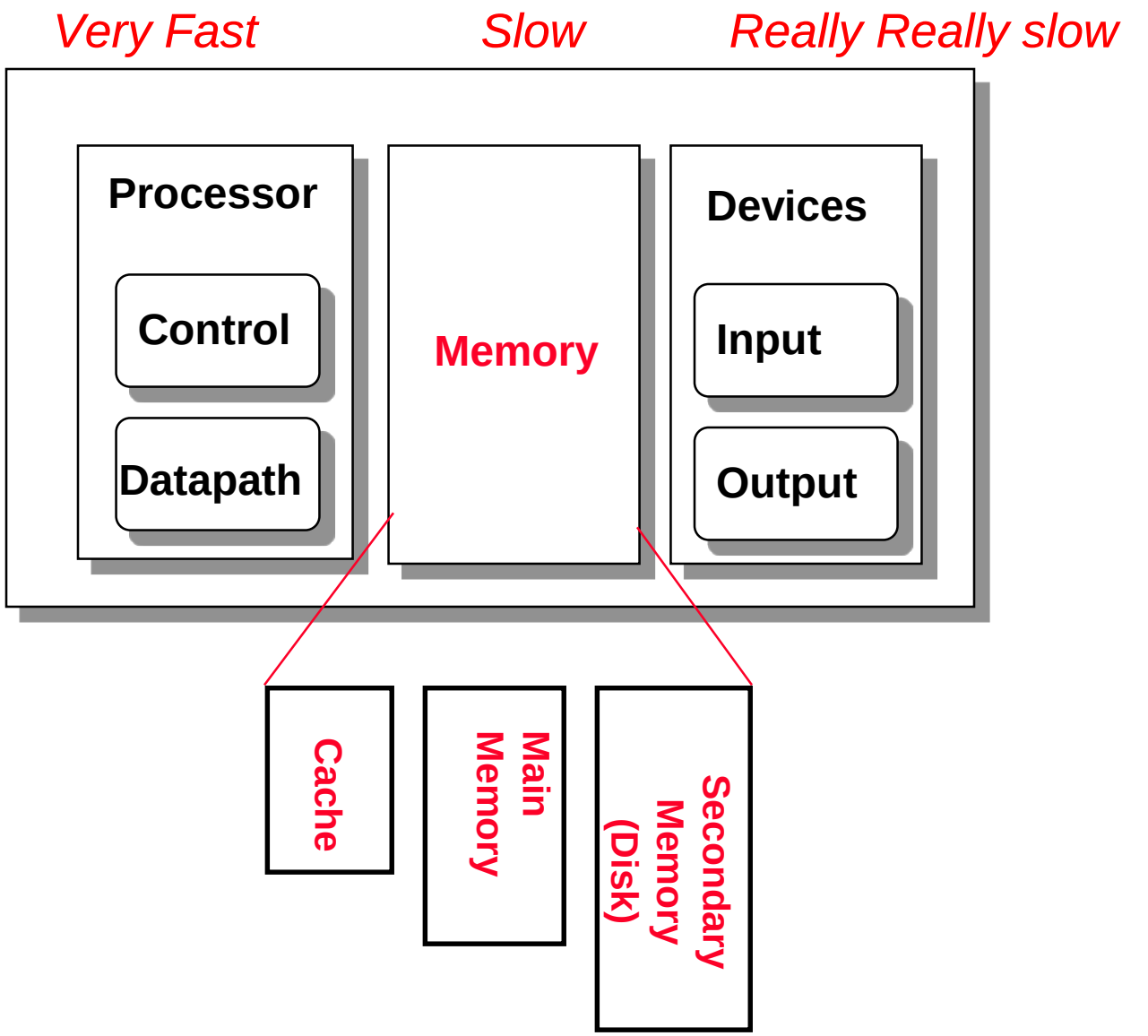


- **Modern systems?** **Figure 5.44**

Different Memory Technologies

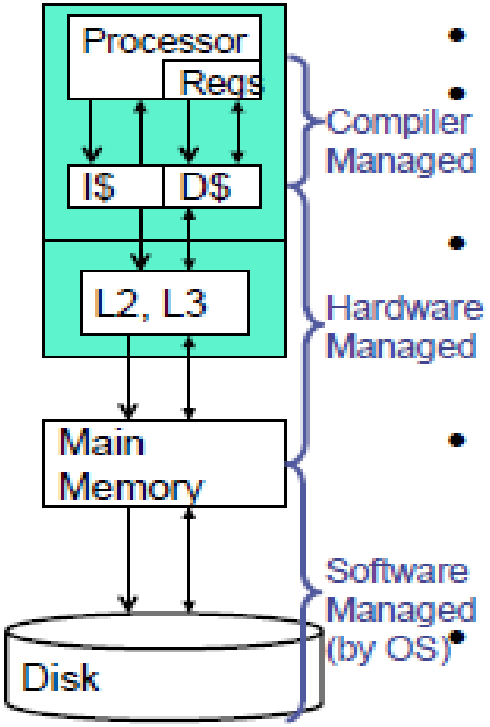
SRAM, DRAM, NVM, and disk

Review: Major Components of a Computer



How to Take Advantage of Locality

- Copy more recently accessed (plus nearby) items from DRAM to smaller SRAM
  - Caches
- Copy recently accessed (plus nearby) items from disk to smaller DRAM
  - Main memory
- Initially, store everything on disk
  - e.g., your program, data



** Keep most recently accessed data items closer to the processor*

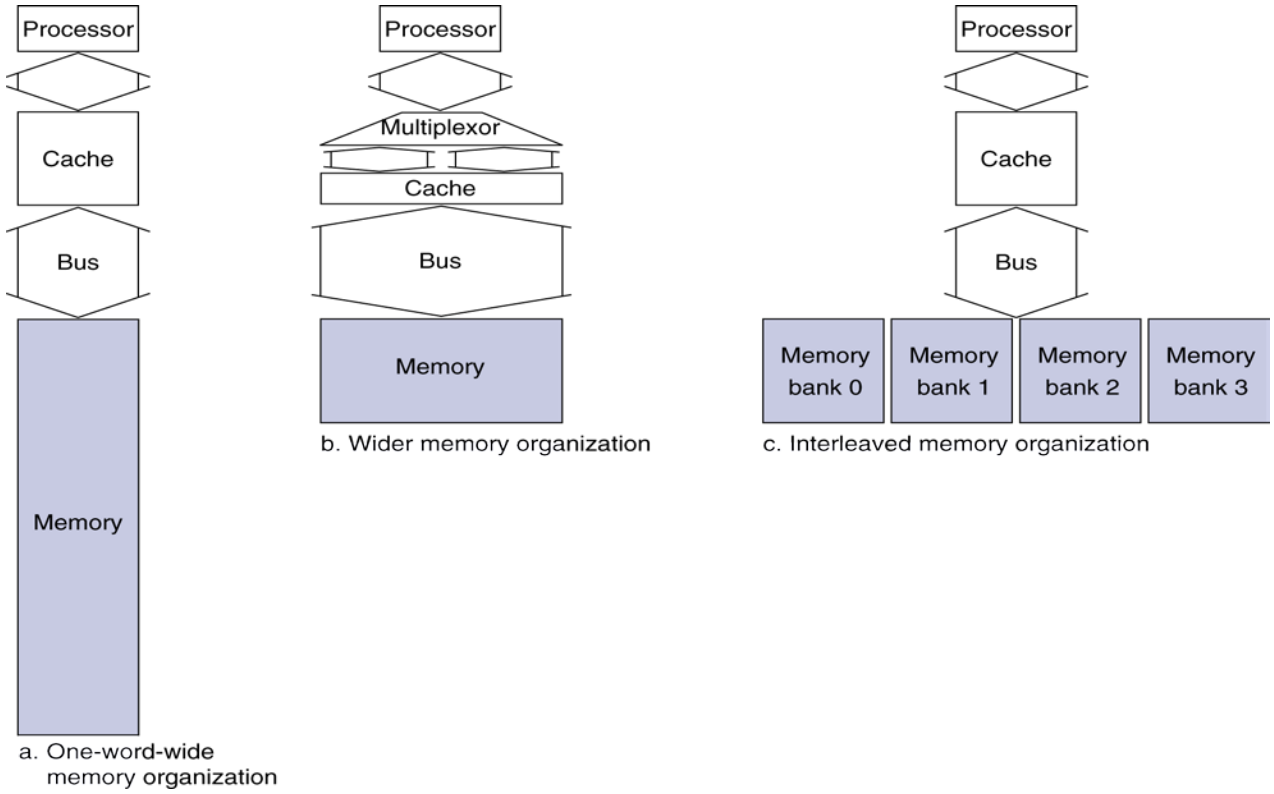
** Move blocks consisting of contiguous words closer to the processor*

Memory Hierarchy Technologies

- Caches use *SRAM* for speed and technology compatibility
  - Fast (typical access times of 0.5 to 2.5 nsec)
  - Low density (6-8 transistor cells), higher power, expensive
  - Static: content will last “forever” (as long as power is left on)

- Main memory uses *DRAM* for size (density)
  - Slower (typical access times of 50 to 70 nsec)
  - High density (1 transistor cells), lower power, cheaper
  - Dynamic: needs to be “refreshed” regularly (~ every 8 ms)
    - consumes1% to 2% of the active cycles of the DRAM
  - Addresses divided into 2 halves (row and column)
    - *RAS* or *Row Access Strobe* triggering the row decoder
    - *CAS* or *Column Access Strobe* triggering the column selector

- DRAM Technology
  - Figure 5.4
  - Figure 5.5
- Increasing Memory Bandwidth



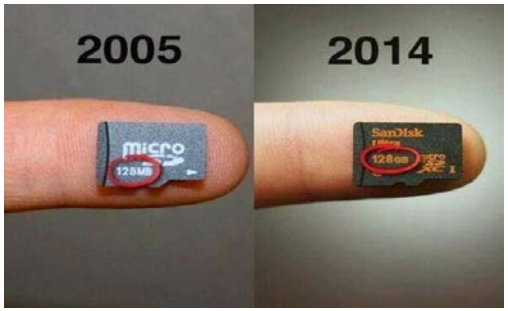
a. One-word-wide memory organization

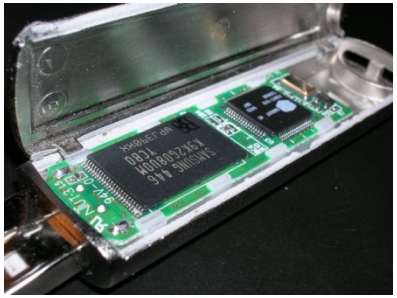
b. Wider memory organization

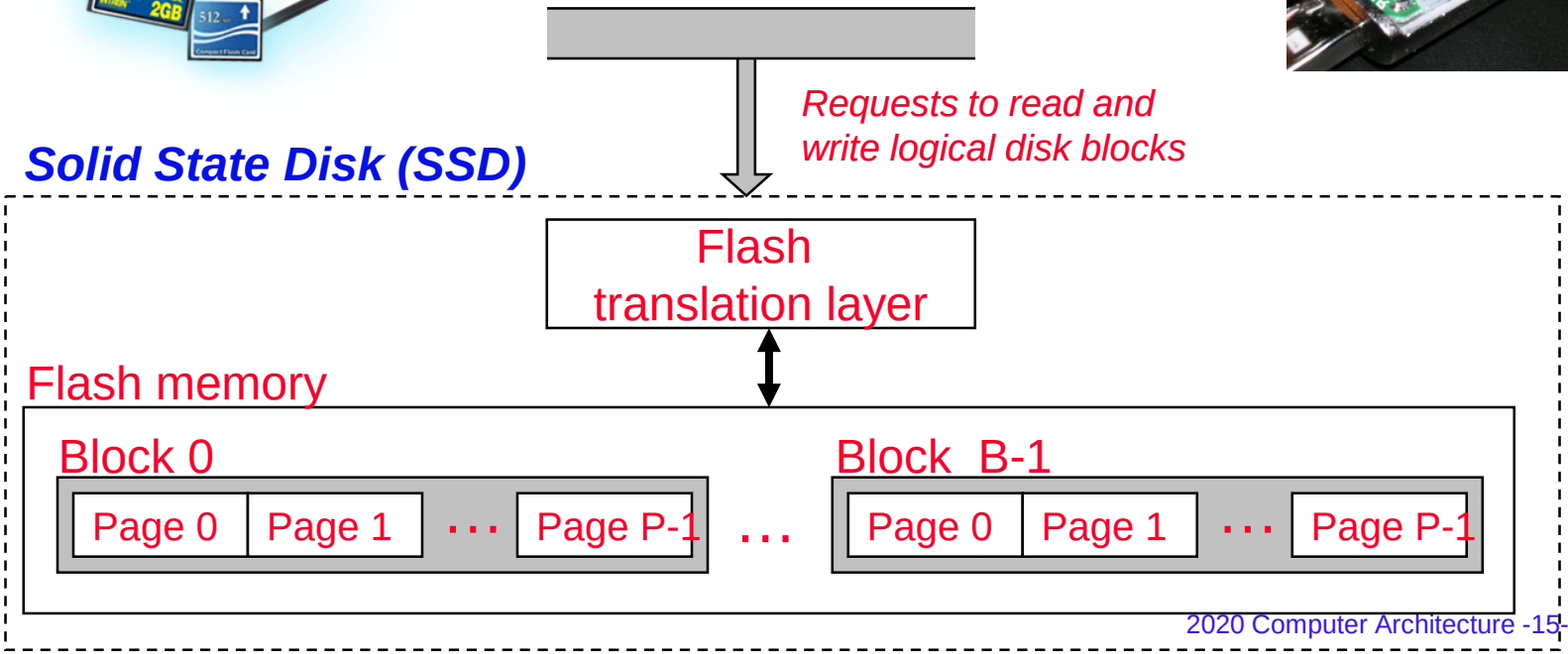
c. Interleaved memory organization

Flash Storage (NVM)

- Non volatile semiconductor storage
  - 1000× faster than disk
  - Smaller, lower power, more robust
  - But more $/GB (between disk and DRAM)





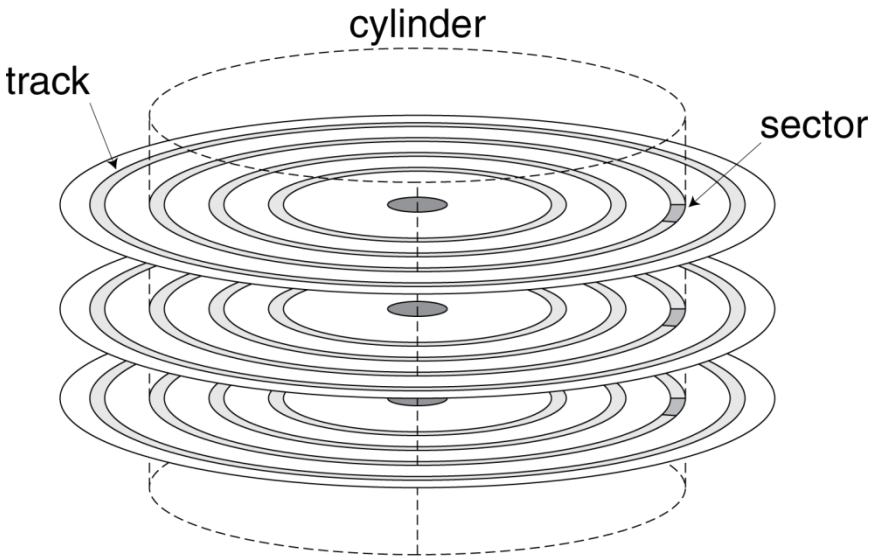


Flash Types

- NOR flash: bit cell like a NOR gate
  - Allow random access to any memory location
  - Used for instruction memory in embedded systems(e.g., ROM)
- NAND flash: bit cell like a NAND gate
  - Denser (bits/area), but block-at-a-time access
  - Cheaper per GB
  - Used for USB keys, media storage, …
- Flash bits wears out after 5000's of accesses
  - Not suitable for direct RAM or disk replacement
  - Wear leveling: remap data to less used blocks

Disk Memory(or Storage)

- Nonvolatile, rotating magnetic storage



Access to a sector involves

. Queuing delay if other accesses are pending

. Seek: move the heads

. Rotational latency

. Data transfer

. Controller overhead

Disk Sectors and Access

- Example
  - 512B sector, 15,000rpm, 4ms average seek time, 100MB/s transfer rate, 0.2ms controller overhead, idle disk
  - Average disk read time : 4ms seek time
    + ½ / (15,000/60) // 2ms rotational latency
    + 512 / 100MB/s // 0.005ms transfer time
    + 0.2ms controller delay
    = 6.205ms
  - *If actual average seek time is 1ms, Average read time = 3.2ms*

- SCSI, ATA, SATA
- Disk drives include caches

Disk Performance Issues

- Manufacturers use average seek time
- Principle of Locality leads to smaller average seek time
- Disk controller allocates physical sectors on disk
  - Present *logical sector interface* to host
- Disk drives include caches
  - Can prefetch sectors in anticipation of access
  - Hence, can avoid seek time and rotational delay

Summary