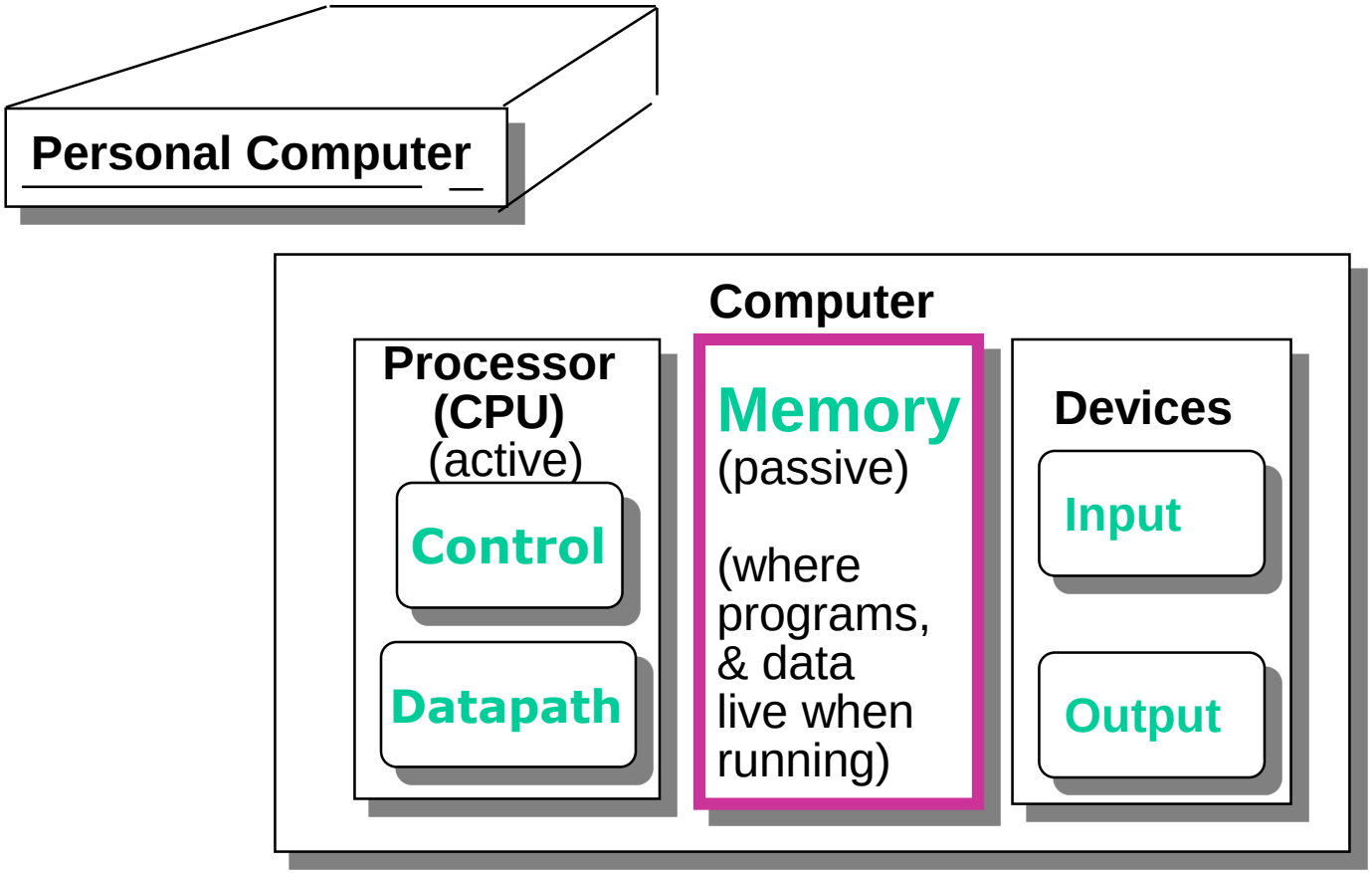
ECE232: Hardware Organization and Design

Lecture 21: Memory Hierarchy

Overview

- Ideally, computer memory would be large and fast
  - **Unfortunately, memory implementation involves tradeoffs**
- **Memory Hierarchy**
  - **Includes caches, main memory, and disks**
- **Caches**
  - **Small and fast**
  - **Contain subset of data from main memory**
  - **Generally close to the processor**
- **Terminology**
  - **Cache blocks, hit rate, miss rate**
- **More mathematical than material from earlier in the course**

Recap: Machine Organization



Memory Basics

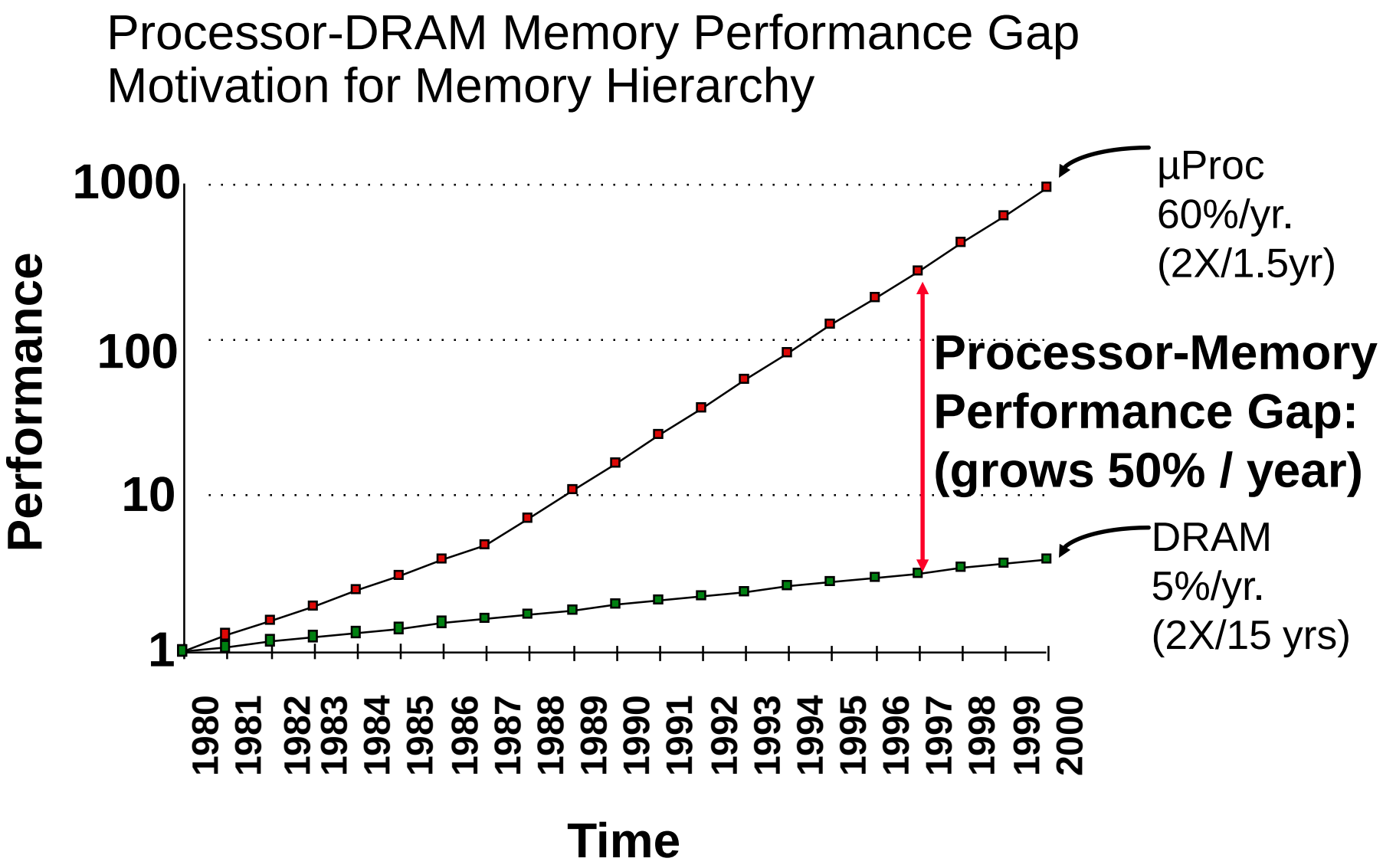
- Users want large and fast memories
- Fact
  - Large memories are slow
  - Fast memories are small
- Large memories use **DRAM** technology: **D**ynamic **R**andom **A**ccess **M**emory
  - High density, low power, cheap, slow
  - Dynamic: needs to be "refreshed" regularly
- DRAM access times are 50-70ns at cost of $10 to $20 per GB
  - FPM (*Fast Page Mode*)

- Fast memories use **SRAM**: **S**tatic **R**andom **A**ccess **M**emory
  - Low density, high power, expensive, fast
  - Static: content lasts "forever" (until lose power)
- SRAM access times are .5 – 5ns at cost of $400 to $1,000 per GB

Memory Technology

- SRAM and DRAM are: Random Access storage
  - Access time is the same for all locations (Hardware decoder used)
- For even larger and cheaper storage (than DRAM) use hard drive (Disk): **Sequential Access**
  - Very slow, Data accessed sequentially, access time is location dependent, considered as I/O
  - Disk access times are 5 to 20 million ns (i.e., msec) at cost of $.20 to $2.00 per GB

Processor-Memory Speed gap Problem



Need for speed

- Assume CPU runs at 3GHz
- Every instruction requires 4B of instruction and at least one memory access (4B of data)
  - 3 * 8 = 24GB/sec
- Peak performance of sequential burst of transfer (*Performance for random access is much much slower due to latency*)
- **Memory bandwidth and access time is a performance bottleneck**

Interface	Width	Frequency	Bytes/Sec
4-way interleaved PC1600 (DDR200) SDRAM	4 x64bits	100 MHz DDR	6.4 GB/s
Opteron Hyper-Transport memory bus	128bits	200 MHz DDR	6.4 GB/s
Pentium 4 "800 MHz" FSB	64bits	200 MHz QDR	6.4 GB/s
PC2 6400 (DDR-II 800) SDRAM	64bits	400 MHz DDR	6.4 GB/s
PC2 5300 (DDR-II 667) SDRAM	64bits	333 MHz DDR	5.3 GB/s
Pentium 4 "533 MHz" FSB	64bits	133 MHz QDR	4.3 GB/s

FSB – Front-Side Bus; DDR – Double Data Rate; SDRAM – Synchronous DRAM

Need for Large Memory

- Small memories are fast
- So just write small programs

"640 K of memory should be enough for anybody"
-- Bill Gates, 1981

- Today's programs require large memories
  - Data base applications may require Gigabytes of memory

The Goal: Illusion of large, fast, cheap memory

- How do we create a memory that is large, cheap and fast (most of the time)?

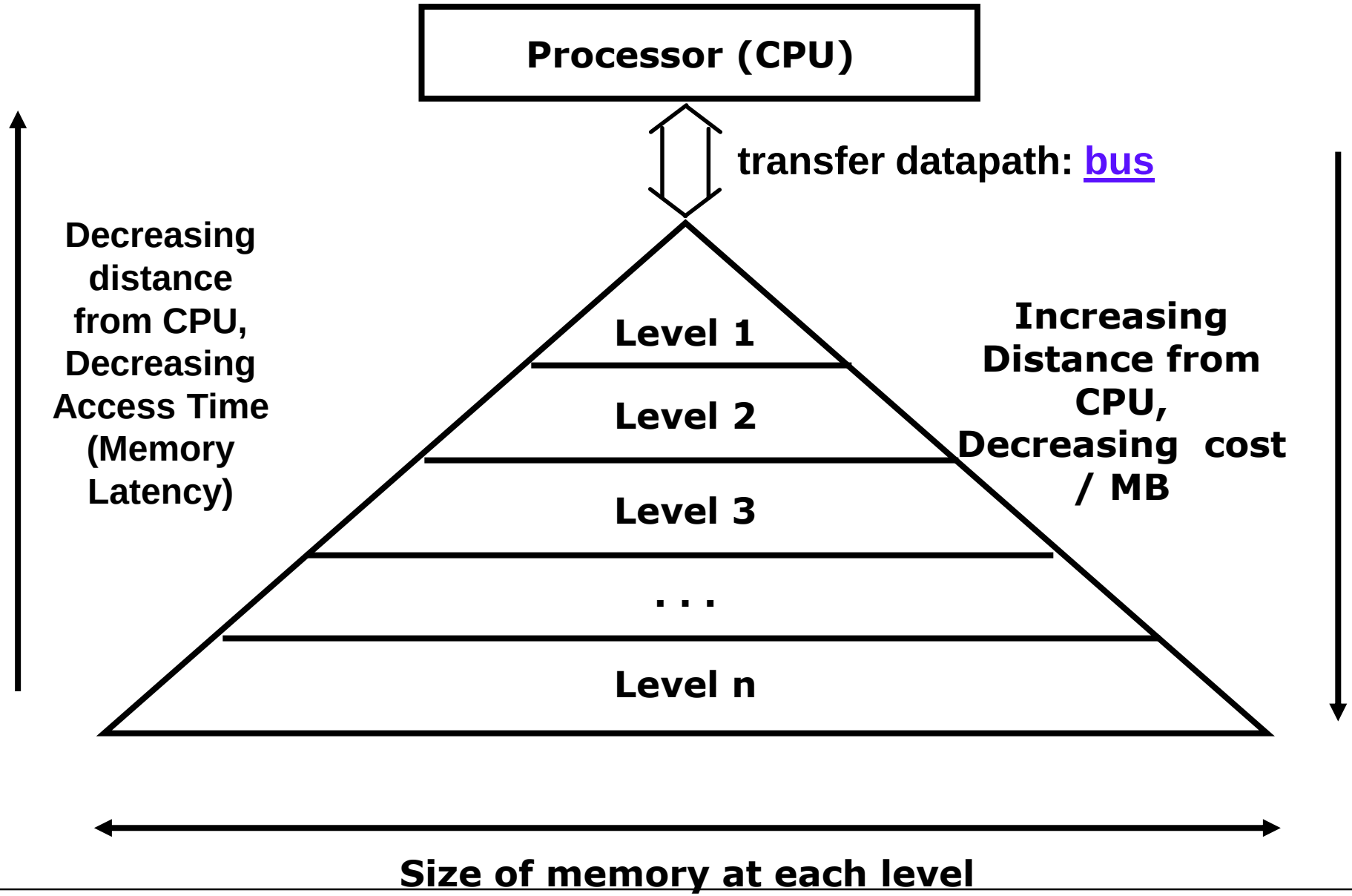
- Strategy: Provide a Small, Fast Memory which holds a subset of the main memory – called cache
  - Keep frequently-accessed locations in fast cache
  - Cache retrieves more than one word at a time
  - Sequential accesses are faster after first access

Memory Hierarchy

- Hierarchy of Levels
  - Uses smaller and faster memory technologies close to the processor
  - Fast access time in highest level of hierarchy
  - Cheap, slow memory furthest from processor

- The aim of memory hierarchy design is to have access time close to the highest level and size equal to the lowest level

Memory Hierarchy Pyramid

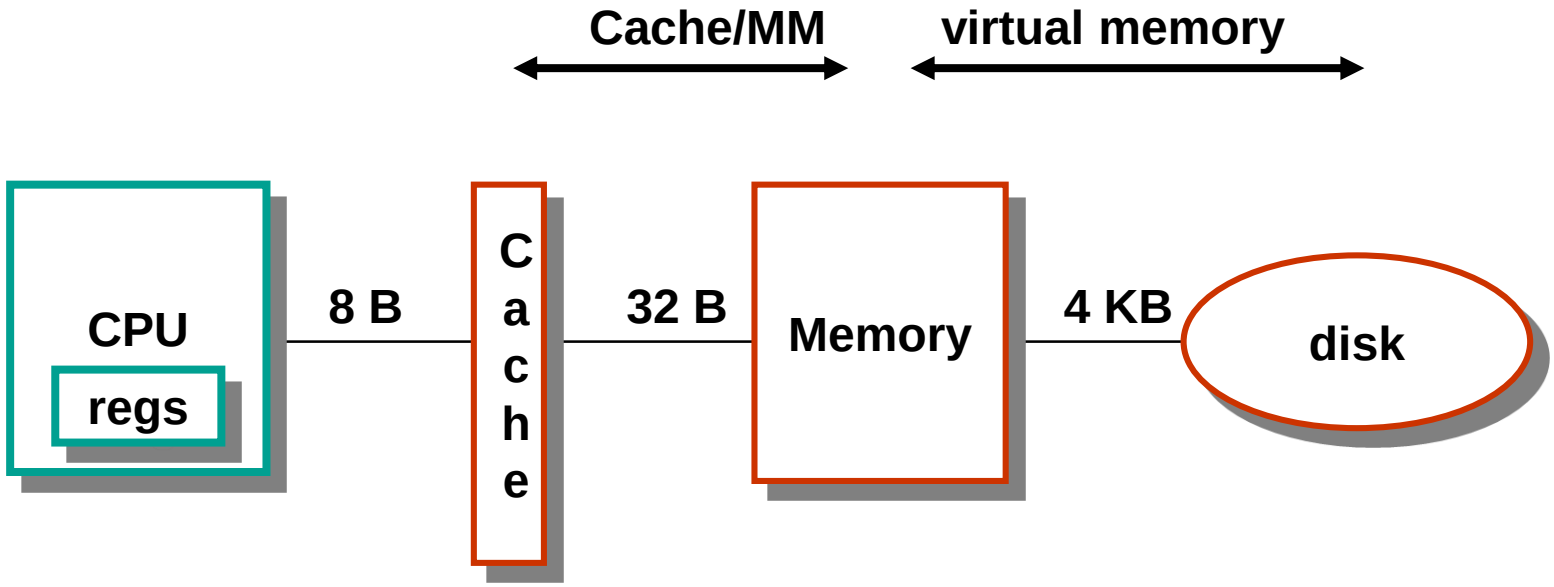


Basic Philosophy

- Move data into ‘smaller, faster’ memory
- Operate on it
- Move it back to ‘larger, cheaper’ memory
  - How do we keep track if changed
- What if we run out of space in ‘smaller, faster’ memory?

- Important Concepts: Latency, Bandwidth

Typical Hierarchy



- Notice that the data width is changing
  - Why?
- Bandwidth: Transfer rate between various levels
  - CPU-Cache: 24 GBps
  - Cache-Main: 0.5-6.4GBps
  - Main-Disk: 187MBps (serial ATA/1500)

Why large blocks?

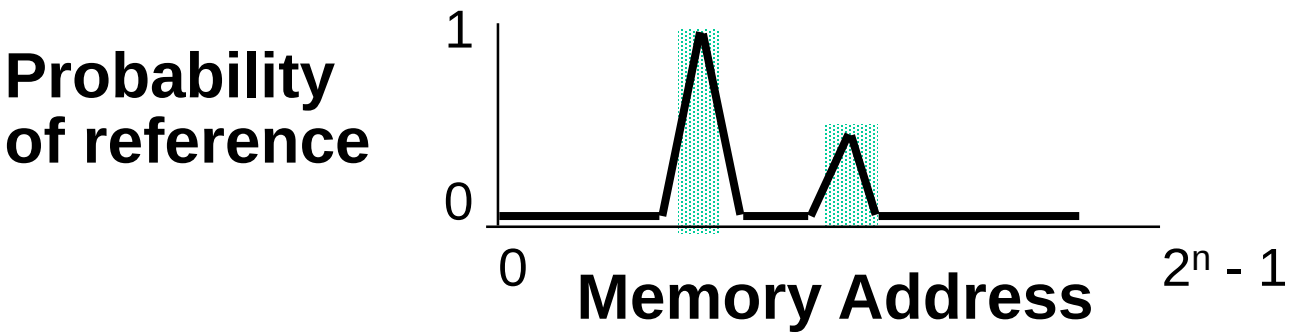
- Fetch large blocks at a time
  - Take advantage of spatial locality

```
for (i=0; i < length; i++)
    sum += array[i];
```

  - `array` has spatial locality
  - `sum` has temporal locality

Why Hierarchy works: Natural Locality

- The **Principle of Locality**
  - Programs access a relatively small portion of the address space at any second



- Temporal Locality (Locality in Time):⇒ Recently accessed data tend to be referenced again soon
- Spatial Locality (Locality in Space):⇒ nearby items will tend to be referenced soon

Taking Advantage of Locality

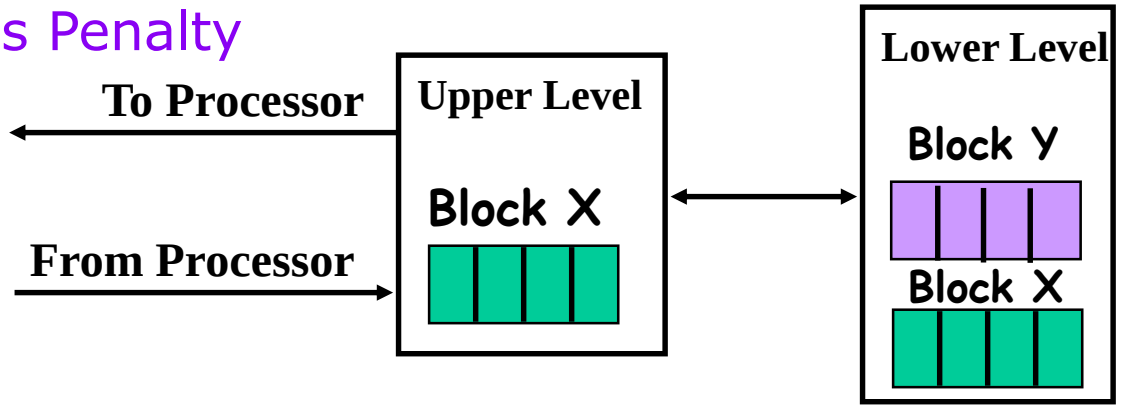
- Memory hierarchy
- Store everything on disk
- Copy recently accessed (and nearby) items from disk to smaller DRAM memory
  - Main memory
- Copy more recently accessed (and nearby) items from DRAM to smaller SRAM memory
  - Cache memory attached to CPU

Principle of Locality

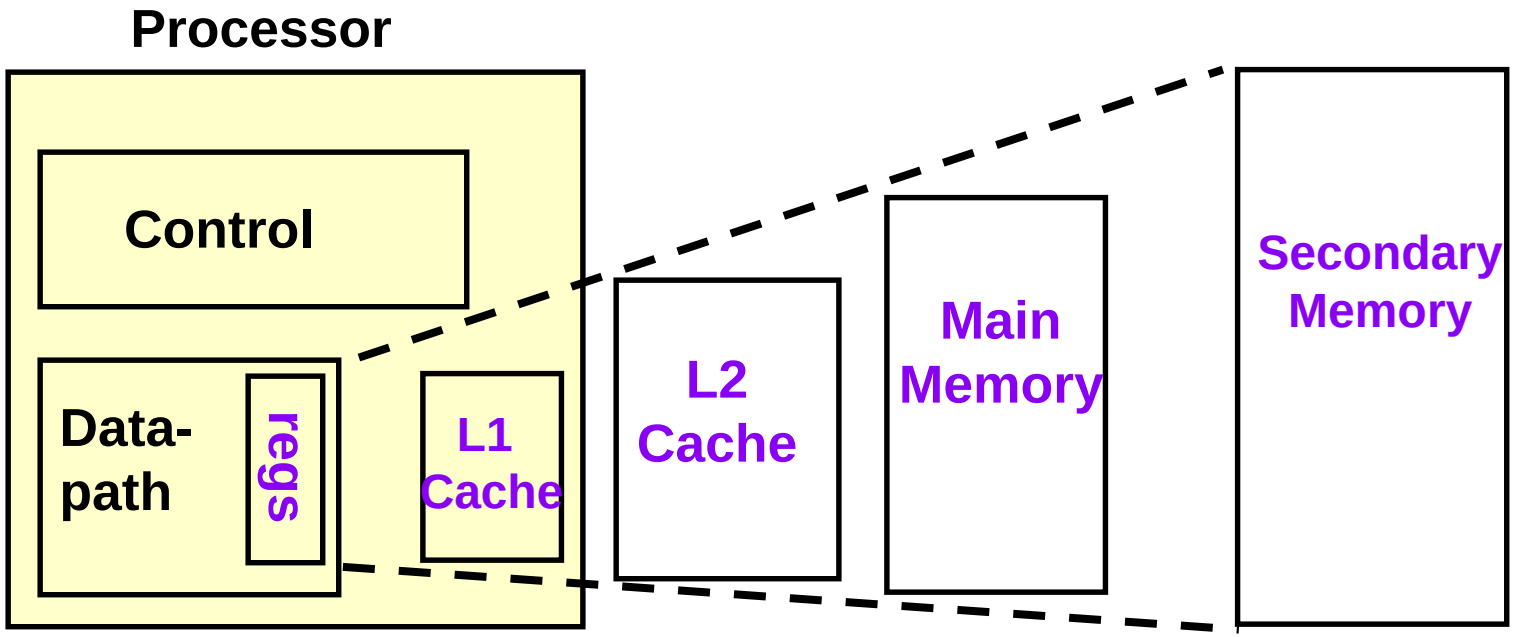
- Programs access a small proportion of their address space at any time
- Temporal locality
  - Items accessed recently are likely to be accessed again soon
  - e.g., instructions in a loop, induction variables
- Spatial locality
  - Items near those accessed recently are likely to be accessed soon
  - E.g., sequential instruction access, array data

Memory Hierarchy: Terminology

- Hit: data appears in upper level in block X
- Hit Rate: the fraction of memory accesses found in the upper level
- Miss: data needs to be retrieved from a block in the lower level (Block Y)
- Miss Rate = 1 - (Hit Rate)
- Hit Time: Time to access the upper level which consists of Time to determine hit/miss + upper level access time
- Miss Penalty: Time to replace a block in the upper level + Time to deliver the block to the processor
- Note: Hit Time << Miss Penalty



Current Memory Hierarchy

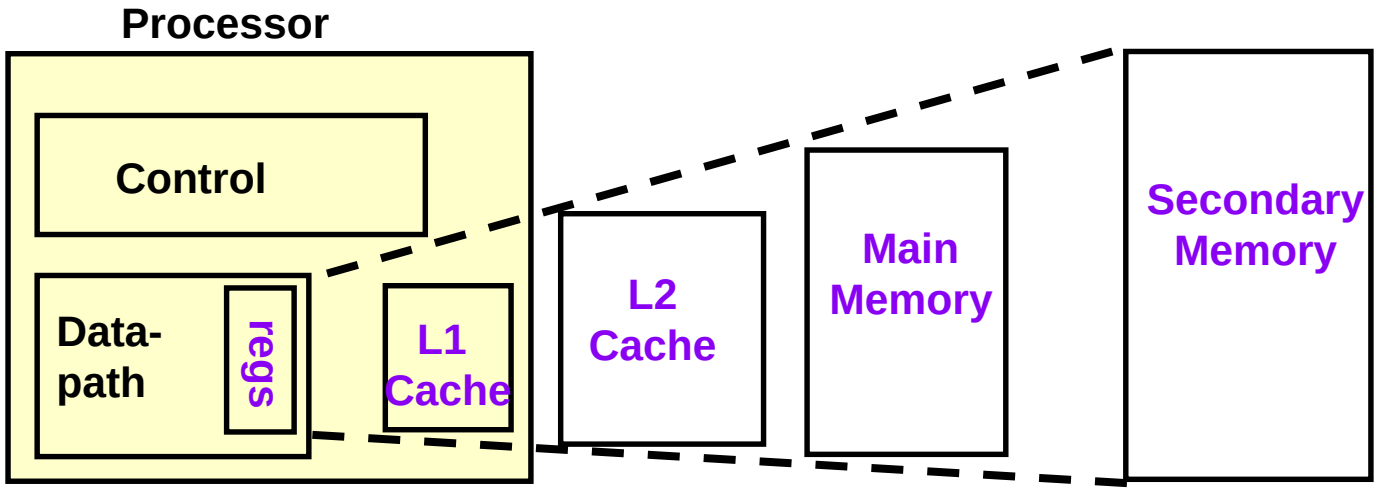


Speed(ns):	1ns	2ns	6ns	100ns	10,000,000ns
Size (MB):	0.0005	0.1	1-4	1000-6000	500,000
Cost ($/MB):	--	$10	$3	$0.01	$0.002
Technology:	Regs	SRAM	SRAM	DRAM	Disk

- Cache - Main memory: Speed
- Main memory – Disk (virtual memory): Capacity

How is the hierarchy managed?

- Registers « Memory
  - By the compiler (or assembly language Programmer)
- Cache « Main Memory
  - By hardware
- Main Memory « Disks
  - By combination of hardware and the operating system
  - virtual memory



Summary

- Computer performance is generally determined by the memory hierarchy
- An effective hierarchy contains multiple types of memories of increasing size
- Caches (our first target) are a subset of main memory
  - Caches have their own special architecture
  - Generally made from SRAM
  - Located close to the processor
- Main memory and disks
  - Bulkier storage
  - Main memory: Volatile (loses data when power removed)
  - Disk: Non-volatile