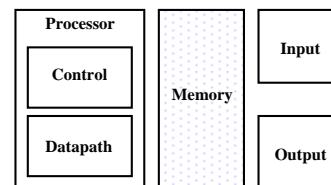


Memory

CIT 595
Spring 2007

Big Picture: Where are we now?

- We will focus on memory organization
- Understand how the organization affects the system performance



Von Neumann Model

CIT 595

11 - 2

Kinds of Memory: Type I

RAM - Random Access Memory

- Access time is the same for all locations hence *Random Access*
- Memory can be read and written
- Is memory specified when you buy a computer e.g. 128 MB RAM
- The instructions and data are stored when executing your programs
- Is *volatile* i.e. once the power is off, the information is lost

CIT 595

11 - 3

Kinds of Memory: Type II

ROM – Read Only Memory

- Memory that can be *only* read
- Use to store critical information necessary to operate the system
 - E.g. Program necessary to boot the computer (BIOS – Basic Input/Output System)
 - E.g. In a laser printer, used to store FONTS
- Is *non-volatile* i.e. information is still retained once the power is turned off

CIT 595

11 - 4

Kinds of RAM: Type I

SRAM – Static RAM

- Similar to memory we studied in chapter 3
 - SR Flip-Flop, D Flip-Flop
- 1-bit information (*memory cell*) needs cross-coupled gates
 - Consists of 8 transistors per cell (1 NAND/NOR gate requires 4 transistors)
 - Can be optimized to use 6 transistors per cell

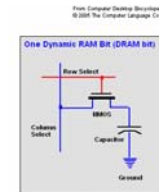
CIT 595

11 - 5

Kinds of RAM: Type II

DRAM - Dynamic RAM

- 1-memory cell consists of one *capacitor* and *transistor*
 - Capacitor is used to store charge
 - Transistor acts as a switch which allows data to be read or written
- DRAM access is slow



1. Capacitors slowly leak their charge over time and hence must be refreshed every few milliseconds to prevent data loss
2. During read, a level detector (sense amplifier) is attached to capacitor to determine the value of stored (amount of charge). The detector device requires a small amount of time to determine if we have a '0' or a '1'.

CIT 595

11 - 6

DRAM vs. SRAM

- Since DRAM requires only 1 transistor and capacitor compared to 6 transistors in SRAM
 - DRAM chip is more *denser* i.e. stores more bits per chip
- Since DRAM needs to be periodically refreshed, and bit read also takes time, they are slow compared to SRAM
 - Hence where access time of data is considered, SRAM is *faster*

CIT 595

11 - 7

Performance/Cost/Capacity

In general

- Slow memory is cheap and has more storage capacity
- Fast memory is expensive and has less storage capacity

As user you want

- Faster memory, will ensure faster data fetching, less wait for the processor, in turn shortens response time
- But also want to be able to have more storage at an affordable price

So how do we get best of everything?

- Use Memory *Hierarchy*

CIT 595

11 - 8

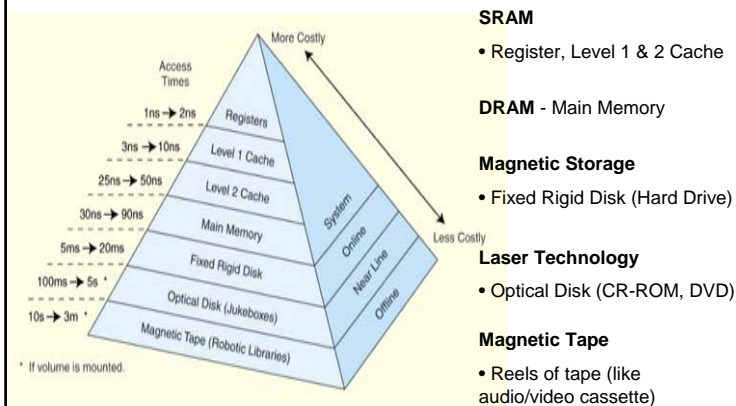
Memory Hierarchy

- To provide the best performance at the lowest cost, memory is organized in a hierarchical fashion
- Small, fast storage elements are *kept in* the CPU (i.e. on-chip)
- Larger, slower main memory is *accessed through* the data bus
- Larger, (almost) permanent storage in the form of disk and tape drives is still further from the CPU

CIT 595

11 - 9

Memory Hierarchy: Pyramid Structure



CIT 595

11 - 10

Processor<->Cache<->Other Memory

- Each level of memory keeps a subset of the data contained in the lower memory-level (i.e. from larger memory)
- To access a particular piece of data, the CPU first sends a request to its nearest memory, i.e. *cache*
- If the data is not in cache, then *main memory* is queried. If the data is not in main memory, then the request goes to *disk*
- Once the data is located at a level, then the data, and a number of its nearby data elements are fetched into cache memory
 - E.g. if data from address x is requested, then data from address X + 1, X + 2, etc. is also sent
 - A *block* (data from multiple blocks) of data is transferred

CIT 595

11 - 11

Why is a block of data transferred?

- Data between levels is transferred using a bus
 - Bus itself takes sometime to transfer data
 - so it would more effective to use this opportunity to get some other data you might require in the future during one bus transaction
- So why get data that is nearby? This because of program structure:
 - **Temporal Locality:** referenced memory is likely to be referenced again soon (e.g. code within a loop)
 - **Spatial Locality:** memory close to referenced memory is likely to be referenced soon (e.g., data in a sequentially access array)
 - **Sequential locality:** Instructions tend to be accessed sequentially
 - The above three are known as *Principles of Locality*

CIT 595

11 - 12

Memory Interleaving

- Main memory usually consists of more than one RAM chip
 - Hence if you buy memory to upgrade you buy a **Memory Module**
- Access is more efficient when memory is organized into banks of chips with the addresses interleaved across the chips
- With **low-order** interleaving, the low order bits of the address specify which memory bank contains the address of interest
- Accordingly, in **high-order** interleaving, the high order address bits specify the memory bank/module

CIT 595

11 - 13

Kinds of Memory Interleaving

Module 0	Module 1	Module 2	Module 3	Module 4	Module 5	Module 6	Module 7
0	4	8	12	16	20	24	28
1	5	9	13	17	21	25	29
2	6	10	14	18	22	26	30
3	7	11	15	19	23	27	31

High-Order Interleaving

Module 0	Module 1	Module 2	Module 3	Module 4	Module 5	Module 6	Module 7
0	1	2	3	4	5	6	7
8	9	10	11	12	13	14	15
16	17	18	19	20	21	22	23
24	25	26	27	28	29	30	31

Low-Order Interleaving

CIT 595

11 - 14

Next in Discussion

- What is the need for a cache?
- How is it organized?
 - Cache structure is different than main memory
- Study the interaction between cache level and main memory level

CIT 595

11 - 15