Introduction

Witawas Srisa-an

Chapter 0

Today's Agenda

- Discuss Administrative issues
- Make suggestions on how to survive this class
- Talk a little about the history of OS
- Assign the first homework (prerequisite evaluation), group exercise, lab1

Administrative Stuff

- Take first-day attendance
- About the instructor
- Go over the syllabus
- Help resources
  - TA Introduction
  - Forum
  - Office Hours
  - Online Survival Page
- Course scheduling

Administrative Stuff (cont.)

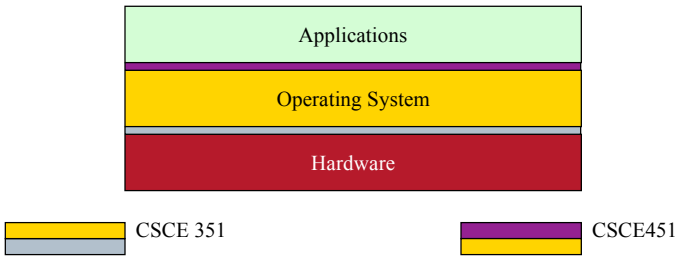
- How to survive this class
  - GIGO does apply
  - Get to know one of your classmates now
  - Participate, participate, participate
    - Answer questions, ask questions
    - Give insightful answers or questions in the forum
    - Show up during office hours
    - You can get up to five points for participation

What Do You Expect?

What You Should Get Out of This Class

- Knowledge of OS internal
- Ability to do system programming
- Ability to modify an OS to fit your needs
- Experience about hardware/software interface
- **Practical OS experience**

CSCE351 Versus CSCE451



Questions

What is an operating system?

How many operating systems have you used? Can you name them?

What are some of the differences among them?

More Questions

What happens when you boot a PC?

What happens when you simultaneously log-in to the department server?

What is the relationship between what you've learned in *Computer Organization (CSCE 230)* and this class?

What is your most favorite programming language?

Credits

- The slide set for this lecture composes of
  - Slides provided by William Stallings accompanying our adopted textbook (but 4th Edition)
  - Slides provided by Andrew Tanenbaum accompanying his textbook "Modern Operating System" (2nd Edition)
  - Some concepts from Silberschatz and Galvin's "Operating Systems" (4th Edition)
  - My own creation

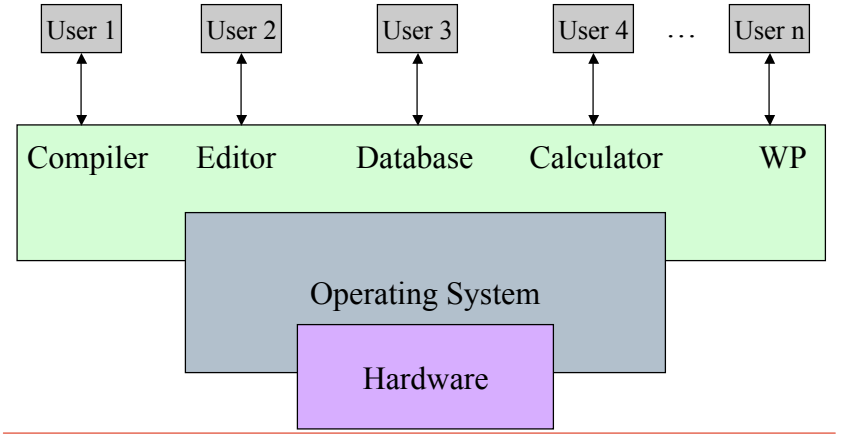
Computer Organization

Witawas Srisa-an

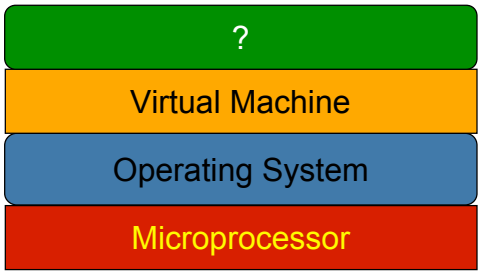
Chapter 1

System Hierarchy

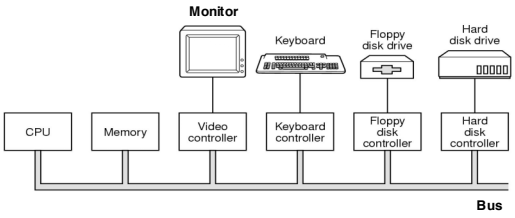
From Operating System Concepts, Silberschatz and Galvin



Abstraction



Basic Elements of Hardware



- Processor(s)
- Main Memory
  - volatile
- I/O modules
  - secondary storage devices
  - communications equipment
  - terminals
- System bus
  - communication among processors, memory, and I/O modules

Top-Level Components

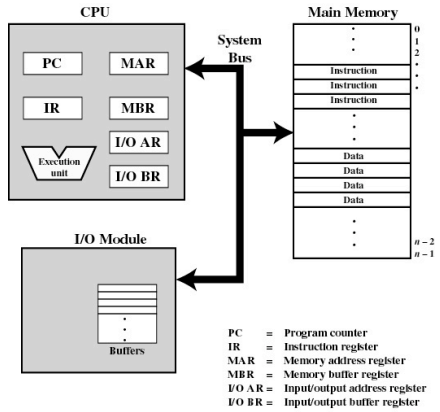


Figure 1.1 Computer Components: Top-Level View

Processor Registers

- User-visible registers
  - Enable programmers to minimize main-memory references by optimizing register use
- Control and status registers
  - Used by the processor to control operation of the processor
  - Used by operating-system routines to control the execution of programs

User-Visible Registers

- May be referenced by machine language
- Available to all programs - application programs and system programs
- Types of registers
  - Data
  - Address
    - Index
    - Segment pointer
    - Stack pointer

User-Visible Registers (2)

- Address Registers
  - Index
    - involves adding an index to a base value to get an address
  - Segment pointer
    - when memory is divided into segments, memory is referenced by a segment and an offset
  - Stack pointer
    - points to top of stack

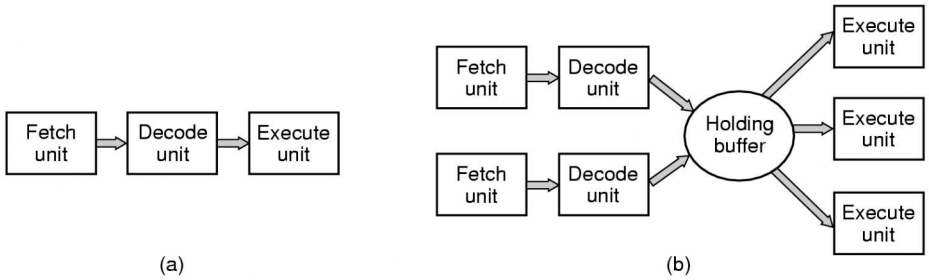
Control and Status Registers

- Program Counter (PC)
  - Contains the address of an instruction to be fetched
- Instruction Register (IR)
  - Contains the instruction most recently fetched
- Program Status Word (PSW)
  - condition codes
  - Interrupt enable/disable
  - Supervisor/user mode

Control and Status Registers (2)

- Condition Codes or Flags
  - Set by the processor hardware to indicate results of operations
  - Can be accessed by a program but not altered
  - Examples
    - positive result
    - negative result
    - zero
    - Overflow

Instruction Cycle



(a) A three-stage pipeline
(b) A superscalar CPU

Instruction Fetch, Decode, and Execute

- The processor fetches the instruction from memory
- Program counter (PC) holds address of the instruction to be fetched next
- Program counter is incremented after each fetch

Instruction Register

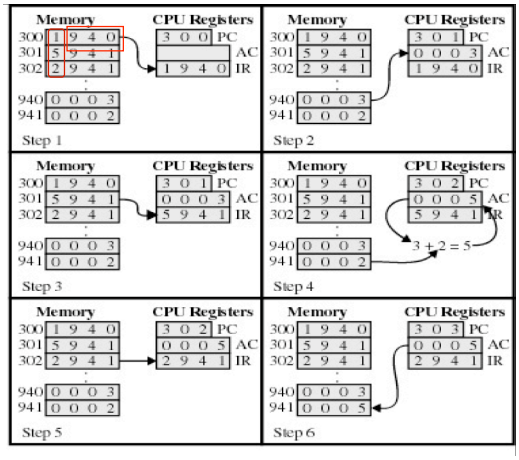
- Fetched instruction is placed in the instruction register
- Instruction is then decoded
- Types of instructions
  - Processor-memory
  - Processor-I/O
  - Data processing
  - Control

Example of a Program Execution

Assume:

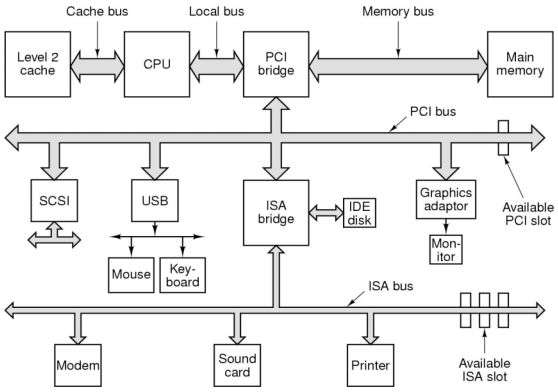
1 = load
2 = store
5 = add

Assume Accumulator Machine



Example of a Real System

Structure of a large Pentium system



How do they communicate?

Example

- Often on-going in my household
  - Week-end to do list
    - Mow the lawn (interactive)
    - Do the laundry (non-interactive)
    - Wash the dishes
    - Clean the house
    - Go out to lunch
    - Cook dinner

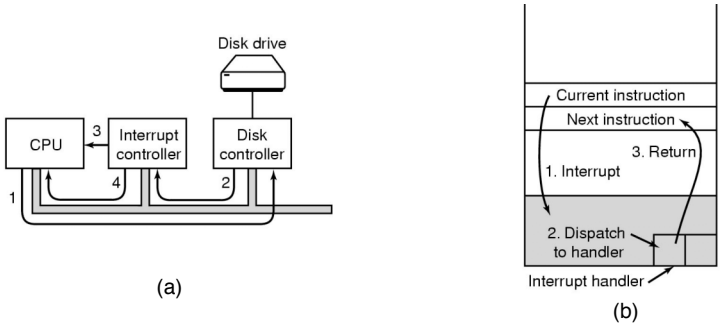
Example (2)

- Key components
  - Need event notification
  - Ability to start a job and let go
    - Dish washing example
      - Hand wash versus machine wash
        - View the dish washer as I/O devices
      - Do your own washing or ask somebody to do it

Interrupts

- An interruption of the normal sequence of execution
- Improves processing efficiency
- Allows the processor to execute other instructions while an I/O operation is in progress
- A suspension of a process caused by an event external to that process and performed in such a way that the process can be resumed

Interrupts (2)



(a) Steps in starting an I/O device and getting interrupt
(b) How the CPU is interrupted

Classes of Interrupts

- Program
  - arithmetic overflow
  - division by zero
  - execute illegal instruction
  - reference outside user's memory space
- Timer
- I/O
- Hardware failure

Interrupt Handler

- A program that determines nature of the interrupt and performs whatever actions are needed
- Control is transferred to this program
- Generally part of the operating system

Interrupt Cycle

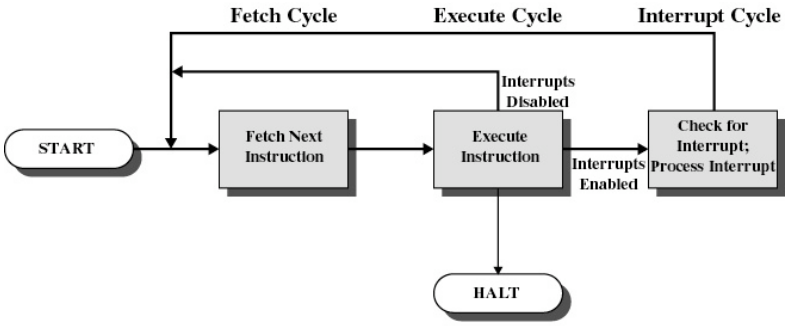
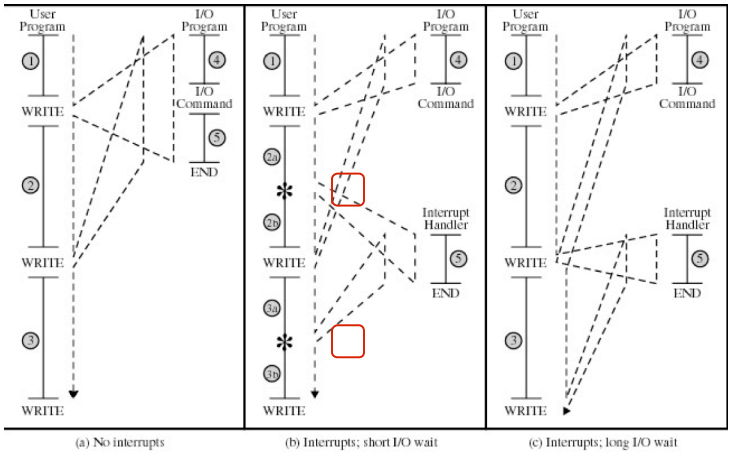


Figure 1.7 Instruction Cycle with Interrupts

Interrupt Cycle (2)

- Processor checks for interrupts
- If no interrupts, processor fetches the next instruction of the current program
- If an interrupt is pending,
  - Suspends the execution of the current program
  - Executes the interrupt handler

Interrupt Cycle (3)



Write 1 is not done; program hangs because Write 2 cannot be processed.

Multiple Interrupts

- Disable interrupts while an interrupt is being processed
  - Processor ignores any new interrupt request signals

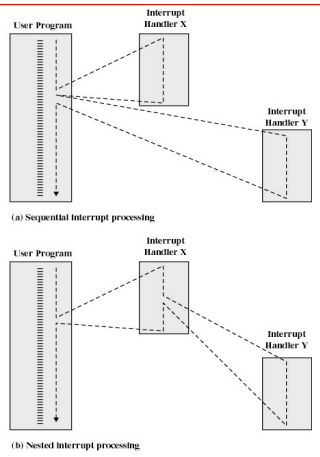


Figure 1.12 Transfer of Control with Multiple Interrupts

Multiple Interrupts Sequential Order

- Disable interrupts so the processor can complete the task
- Interrupts remain pending until the processor enables interrupts
- After the interrupt handler routine completes, the processor checks for additional interrupts

Multiple Interrupts Priorities

- Higher priority interrupts cause lower-priority interrupts to wait
  - Causes a lower-priority interrupt handler to be interrupted
- Example
  - When an input arrives from a communication line, it needs to be absorbed quickly to make room for more input

Multiprogramming

- Processor has more than one program to execute
- The sequence the programs are executed depend on their relative priority and whether they are waiting for I/O
- After an interrupt handler completes, control may not return to the program that was executing at the time of the interrupt

Memory Hierarchy

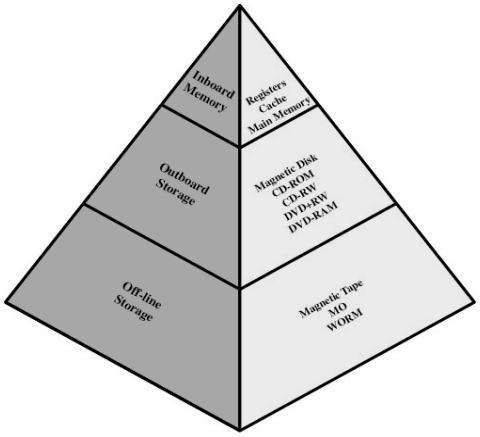
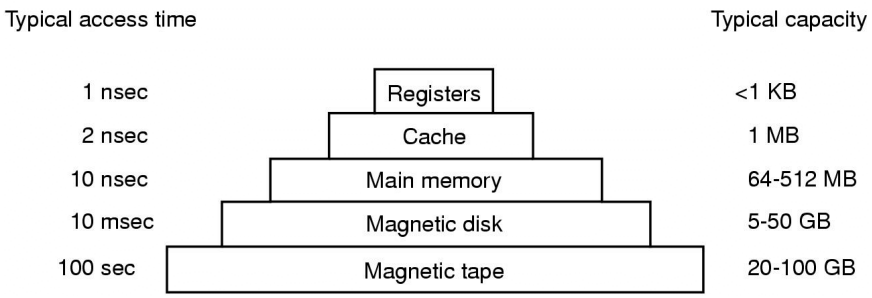


Figure 1.14 The Memory Hierarchy

Memory Hierarchy (2)

Year 2000 numbers



Let's replace with 2005 numbers

Going Down the Hierarchy

- Decreasing cost per bit
- Increasing capacity
- Increasing access time
- Decreasing frequency of access of the memory by the processor
  - locality of reference

Disk Cache

- A portion of main memory used as a buffer to temporarily to hold data for the disk
- Disk writes are clustered
- Some data written out may be referenced again. The data are retrieved rapidly from the software cache instead of slowly from disk

Cache Memory

- Invisible to operating system
- Increase the speed of memory access
- Bridging processor/memory gap
  - Processor speed is faster than memory speed

Cache Memory

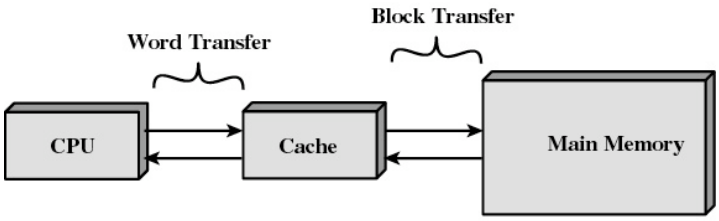


Figure 1.16 Cache and Main Memory

Cache Memory

- Contains a portion of main memory
- Processor first checks cache
- If not found in cache, the block of memory containing the needed information is moved to the cache

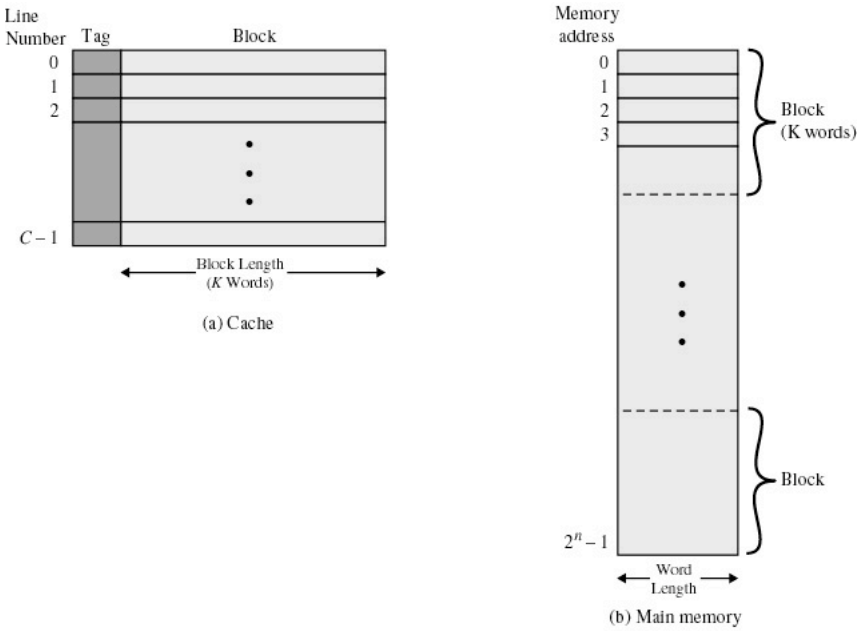


Figure 1.17 Cache/Main-Memory Structure

Lincoln

Cache Design

- Cache size
  - small caches have a significant impact on performance
- Block size
  - the unit of data exchanged between cache and main memory
  - hit means the information was found in the cache
  - larger block size more hits until probability of using newly fetched data becomes less than the probability of reusing data that has been moved out of cache

Cache Design

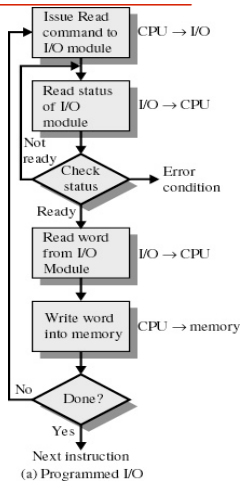
- Mapping function
  - determines which cache location the block will occupy
- Replacement algorithm
  - determines which block to replace
  - Least-Recently-Used (LRU) algorithm

Cache Design

- Write policy
  - When the memory write operation takes place
  - Can occur every time block is updated
  - Can occur only when block is replaced
    - Minimizes memory operations
    - Leaves memory in an obsolete state

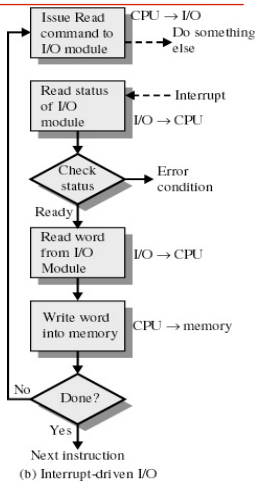
Programmed I/O

- I/O hardware module performs the action, not the processor
- Sets appropriate bits in the I/O status register
- No interrupts occur
- Processor checks status until operation is complete



Interrupt-Driven I/O

- Processor is interrupted when I/O module ready to exchange data
- Processor is free to do other work
  - No needless waiting
- Still consumes a lot of processor time because every word read or written passes through the processor

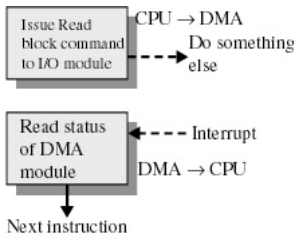


Direct Memory Access (DMA)

- I/O exchanges occur directly with memory
- Processor grants I/O module authority to read from or write to memory
- Relieves the processor responsibility for the exchange
- Processor is free to do other things

Direct Memory Access

- Transfers a block of data directly to or from memory
- An interrupt is sent when the task is complete
- The processor is only involved at the beginning and end of the transfer



(c) Direct memory access

Summary

- Today's computers are complex
  - Multiple CPUs even on a desktop (SMP in a box)
    - dual processors
    - dual core
  - Large memory
    - 16 GB of main memory on the motherboard
  - Complex I/O devices
  - Clusters and server farms
    - resource isolation and binary compatibility
- Need sophisticated software systems to manage the underlying hardware