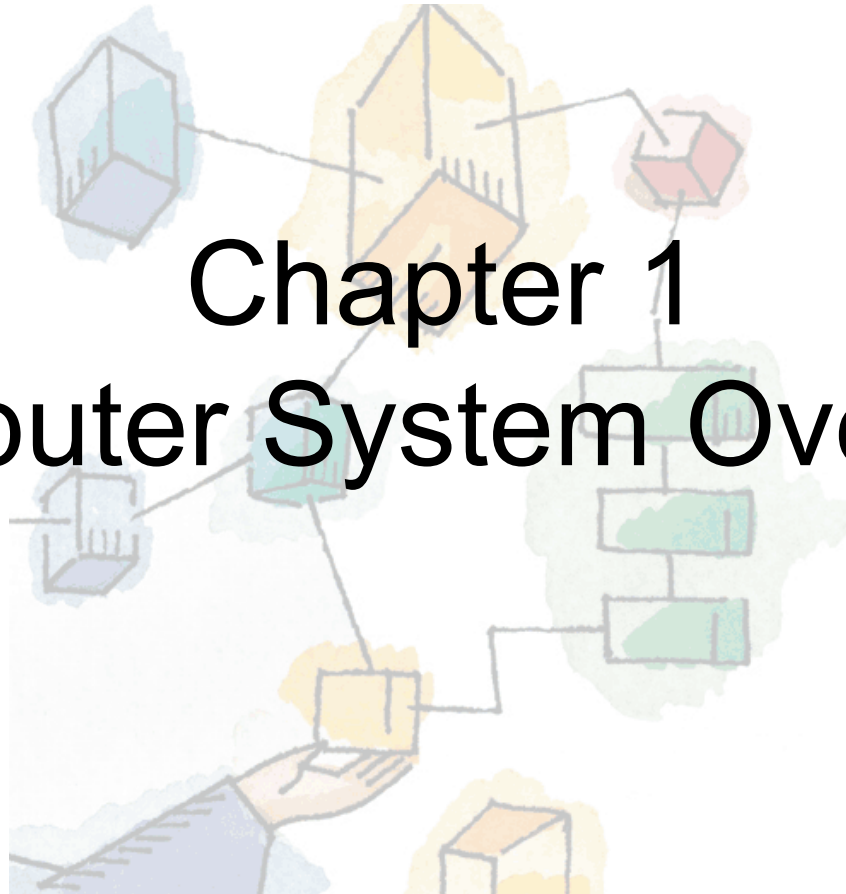


*Operating Systems:  
Internals and Design Principles, 7/E*  
William Stallings

# Chapter 1

## Computer System Overview





# What is an Operating System?

- Operating system goals:
  - Use the computer hardware in an efficient manner.
    - **An OS itself is a software program** that manages the hardware and software resources of a computer.
  - An OS performs basic tasks, such as managing execution of other programs, controlling and allocating memory, controlling input and output devices, managing files and facilitating networking.
  - Make solving user problems easier.
  - Make the computer system convenient to use.





# Operating System Definition

- **resource manager**
  - Manages all resources
  - Decides between conflicting requests for efficient and fair resource use
- **control program**
  - Controls execution of programs to prevent errors and improper use of the computer
- **extended machine**
  - Turns complicated hardware into nice abstractions

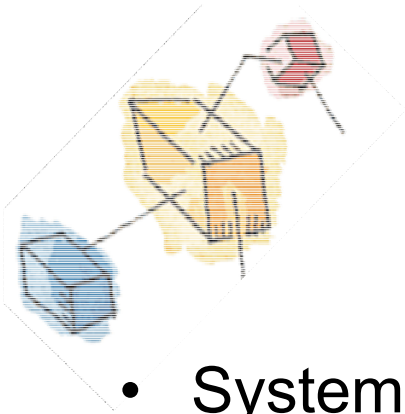




# Operating System Definition (cont.)

- No universally accepted definition
- “Everything a vendor ships when you order an operating system” is good approximation
  - But varies wildly
- “The one program running at all times on the computer” is the **kernel**. Everything else is either a system program (ships with the operating system) or an application program

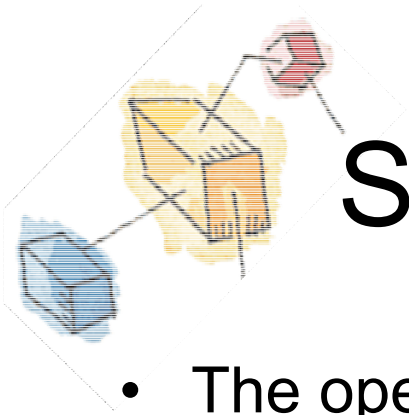




# System Software

- System software is a generic term referring to any computer software whose purpose is to help run the computer system.
- Most of it responsible directly for controlling, integrating, and managing the individual hardware components of a computer system.
- System software is opposed to application software that helps solve user problems directly.





# System Software (cont.)

- The operating system is part of system software.
- However, it is distinguished from other system software:
  - interacts directly with the hardware to provide an interface used by other system/application software
  - domain independent, i.e., can be used to support a broad range of application domains
  - allows different application to share the hardware resources
- Other system software, e.g., compilers, debuggers, system utilities



# Computer Hardware and Software Infrastructure

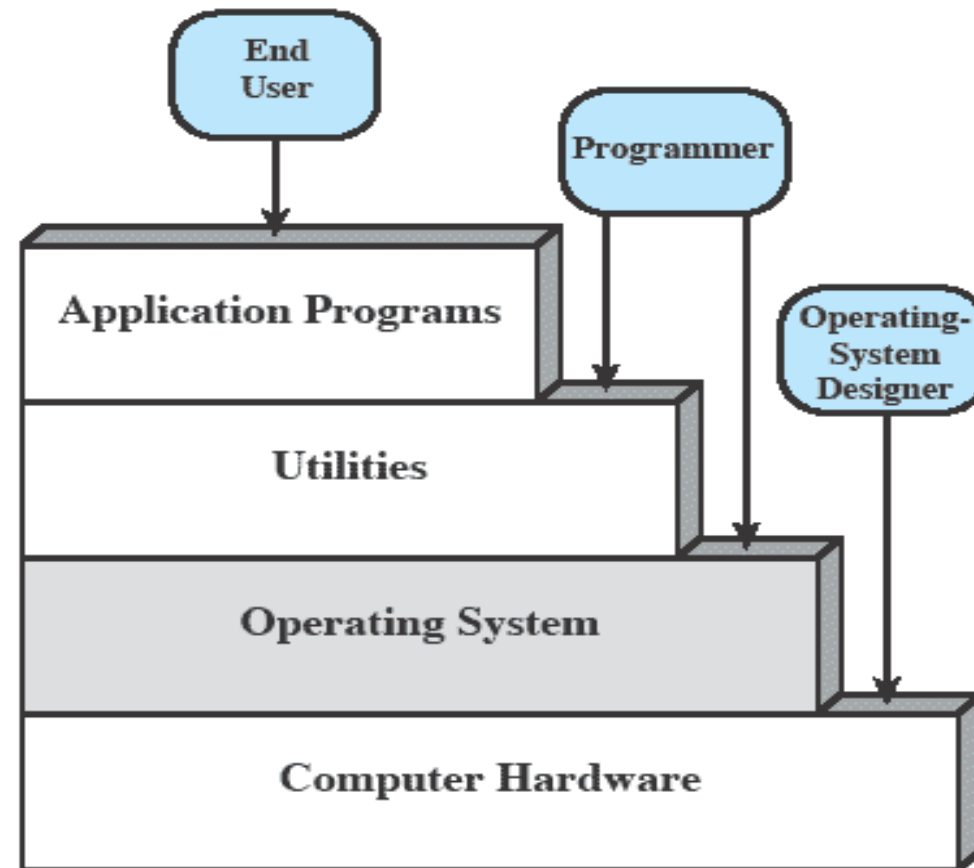


Figure 2.1 Layers and Views of a Computer System



# Why Studying Operating Systems

- Operating systems are an essential part of any computer system – a course on OS is thus an essential part of any computer science education
- Easy to see how to effectively use the computer system
- Enables us to write efficient code
- Learn to design an OS
- ...

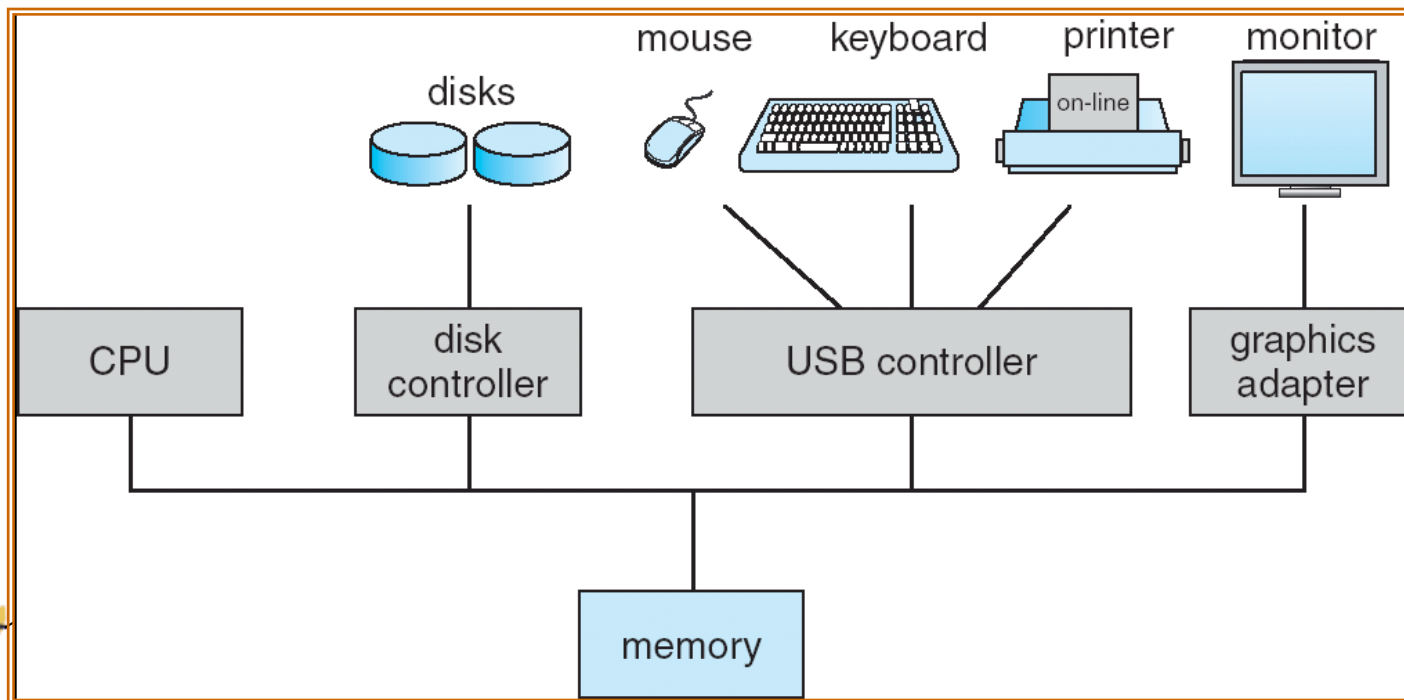




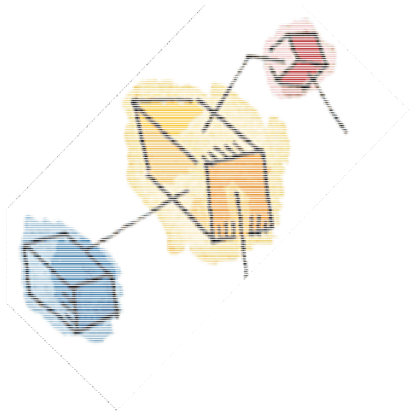


# Computer System Organization

- Computer-system operation
  - One or more CPUs, device controllers connect through common system bus providing access to shared memory
  - Concurrent execution of CPUs and devices competing for memory cycles



# Basic Elements

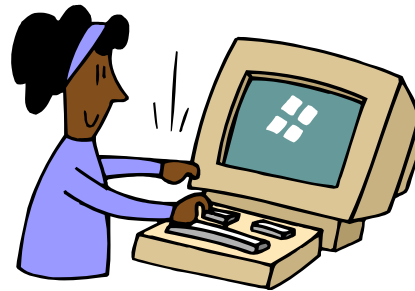


**Processor**

**I/O  
Modules**

**Main  
Memory**

**System  
Bus**





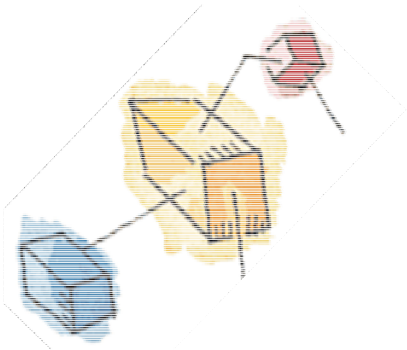
# Processor

Controls the  
operation of the  
computer

Performs the  
data processing  
functions

Referred to as  
the *Central  
Processing Unit*  
(CPU)



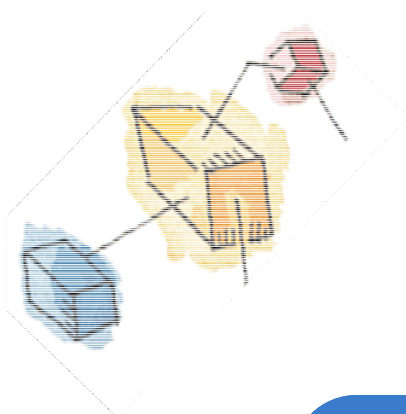


# Main Memory

- Volatile
- Contents of the memory is lost when the computer is shut down
- Referred to as real memory or primary memory



# I/O Modules



Moves data  
between the  
computer and  
external  
environments  
such as:

storage (e.g.  
hard drive)

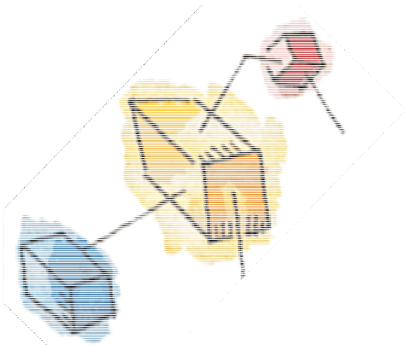
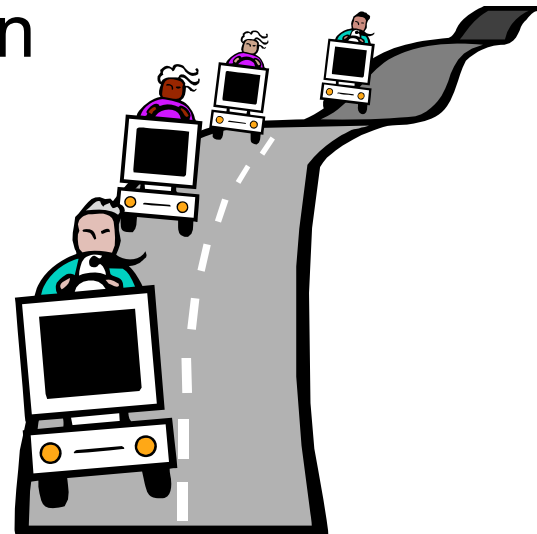
communications  
equipment

terminals



# System Bus

- Provides for communication among processors, main memory, and I/O modules



# Computer Components: Top-Level View

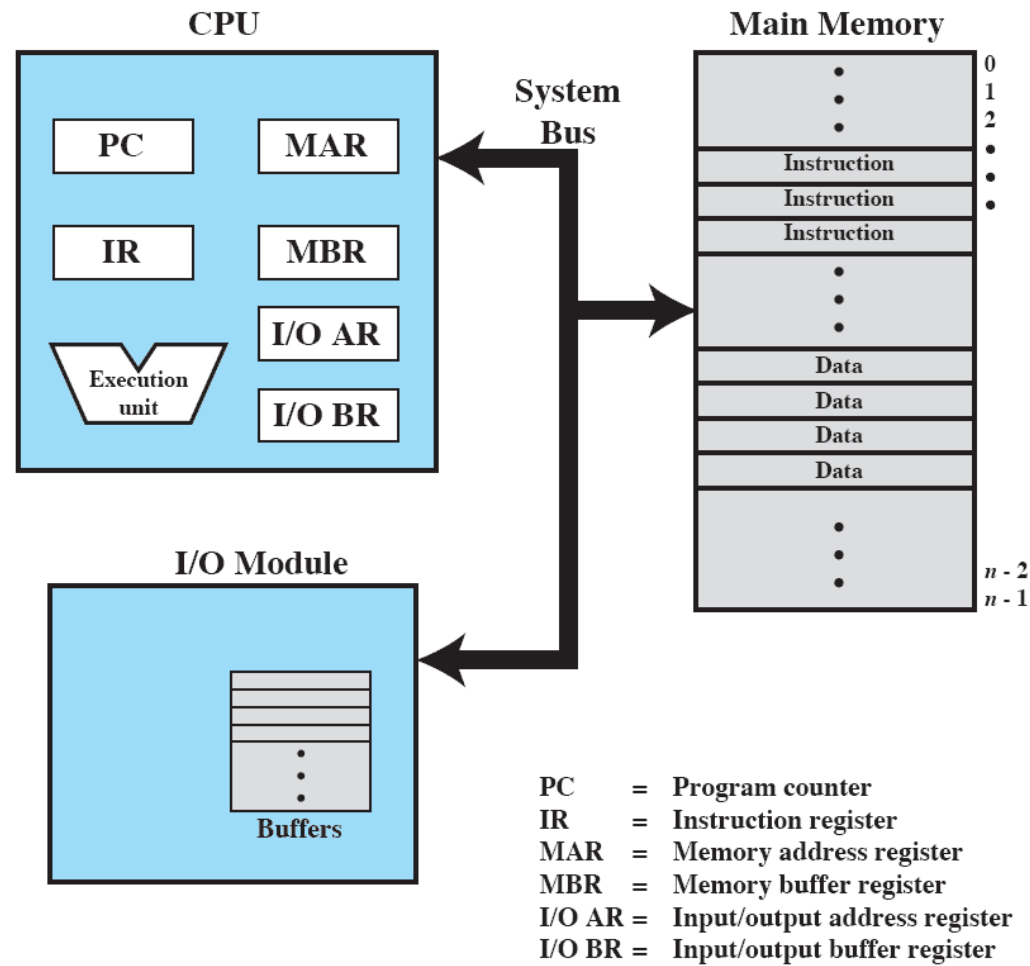
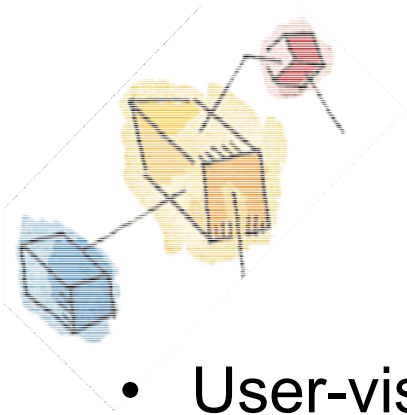


Figure 1.1 Computer Components: Top-Level View

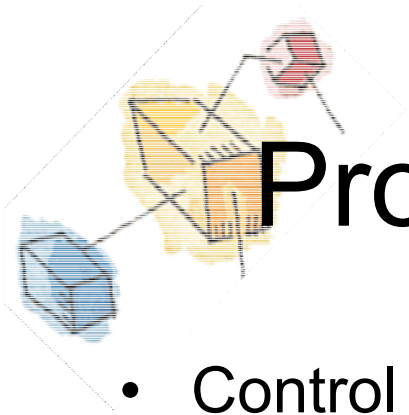


# Processor Registers

- User-visible registers
  - Enable programmer to minimize main memory references by optimizing register use
- May be referenced by machine language
- Available to all programs – application programs and system programs



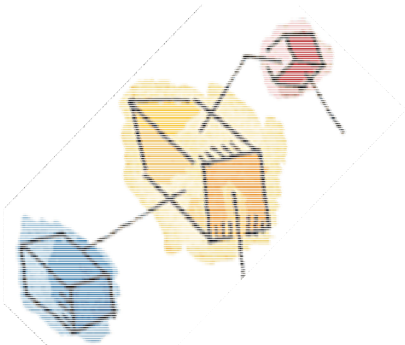




# Processor Registers (cont.)

- Control and status registers
  - Used by processor to control operating of the processor
  - Used by privileged OS routines to control the execution of programs
- Program counter (PC), Instruction register (IR)
- Program status word (PSW)
  - Contains status information
    - Condition codes or flags
      - Bits set by processor hardware as a result of operations, e.g., Positive, negative, zero, or overflow result





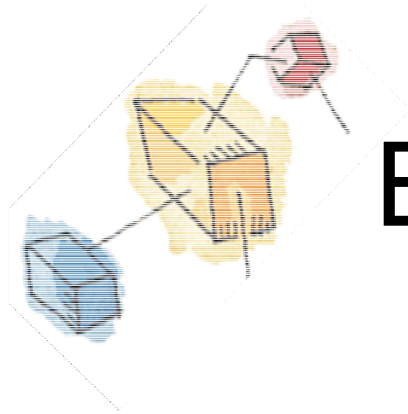
# Instruction Execution

- A program consists of a set of instructions stored in memory

## Two steps:

- processor reads (fetches) instructions from memory
- processor executes each instruction





# Basic Instruction Cycle

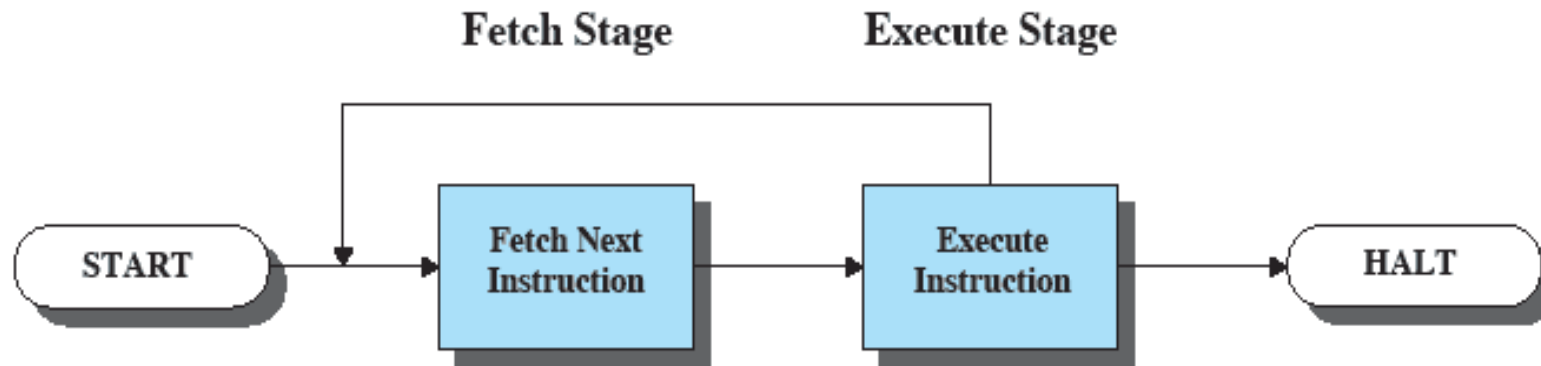


Figure 1.2 Basic Instruction Cycle



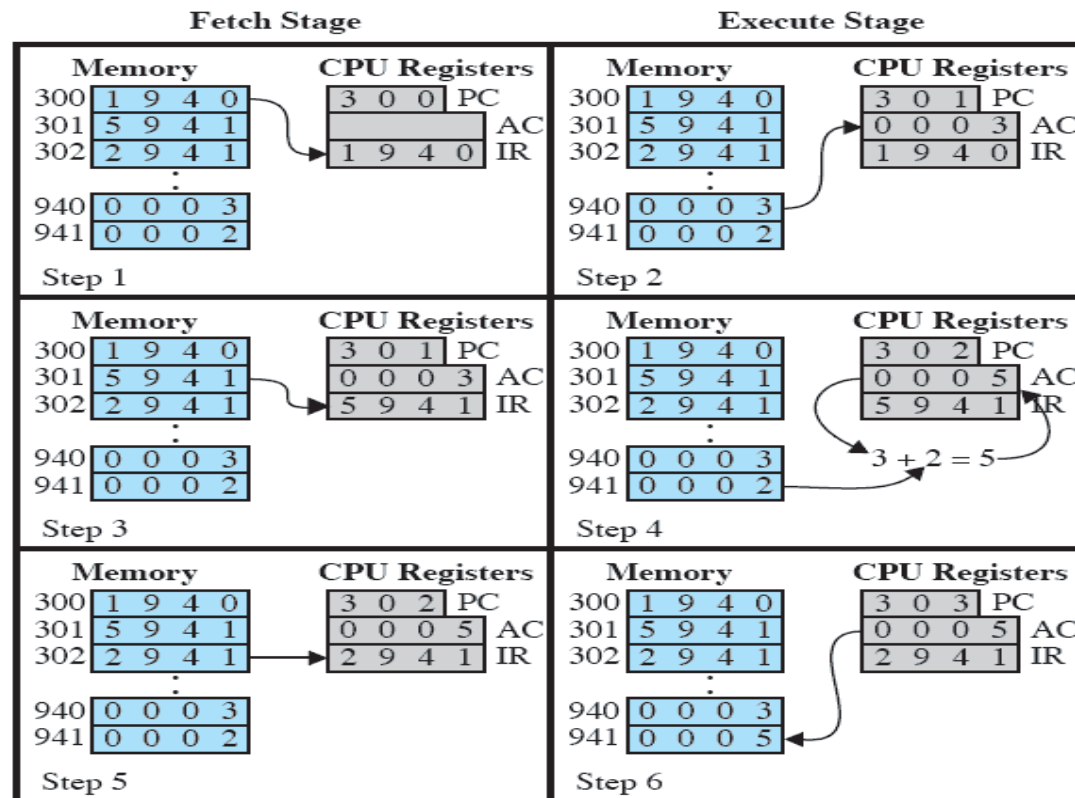


# Instruction Fetch & Execution

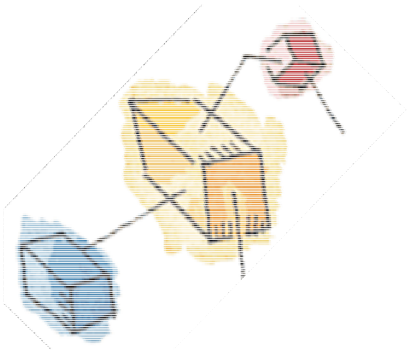
- The processor fetches the instruction from memory
- Program counter (PC) holds address of the instruction to be fetched next
  - PC is incremented after each fetch



# Example of Instruction Execution



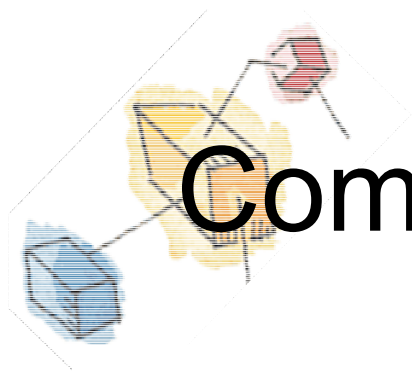
**Figure 1.4 Example of Program Execution**  
(contents of memory and registers in hexadecimal)



# Interrupts

- Interrupt the normal sequencing of the processor
- Provided to improve processor utilization
  - most I/O devices are slower than the processor
  - processor must pause to wait for device
  - wasteful use of the processor





# Common Classes of Interrupts

**Table 1.1**    **Classes of Interrupts**

<b>Program</b>	Generated by some condition that occurs as a result of an instruction execution, such as arithmetic overflow, division by zero, attempt to execute an illegal machine instruction, and reference outside a user's allowed memory space.
<b>Timer</b>	Generated by a timer within the processor. This allows the operating system to perform certain functions on a regular basis.
<b>I/O</b>	Generated by an I/O controller, to signal normal completion of an operation or to signal a variety of error conditions.
<b>Hardware failure</b>	Generated by a failure, such as power failure or memory parity error.



# Instruction Cycle with Interrupts

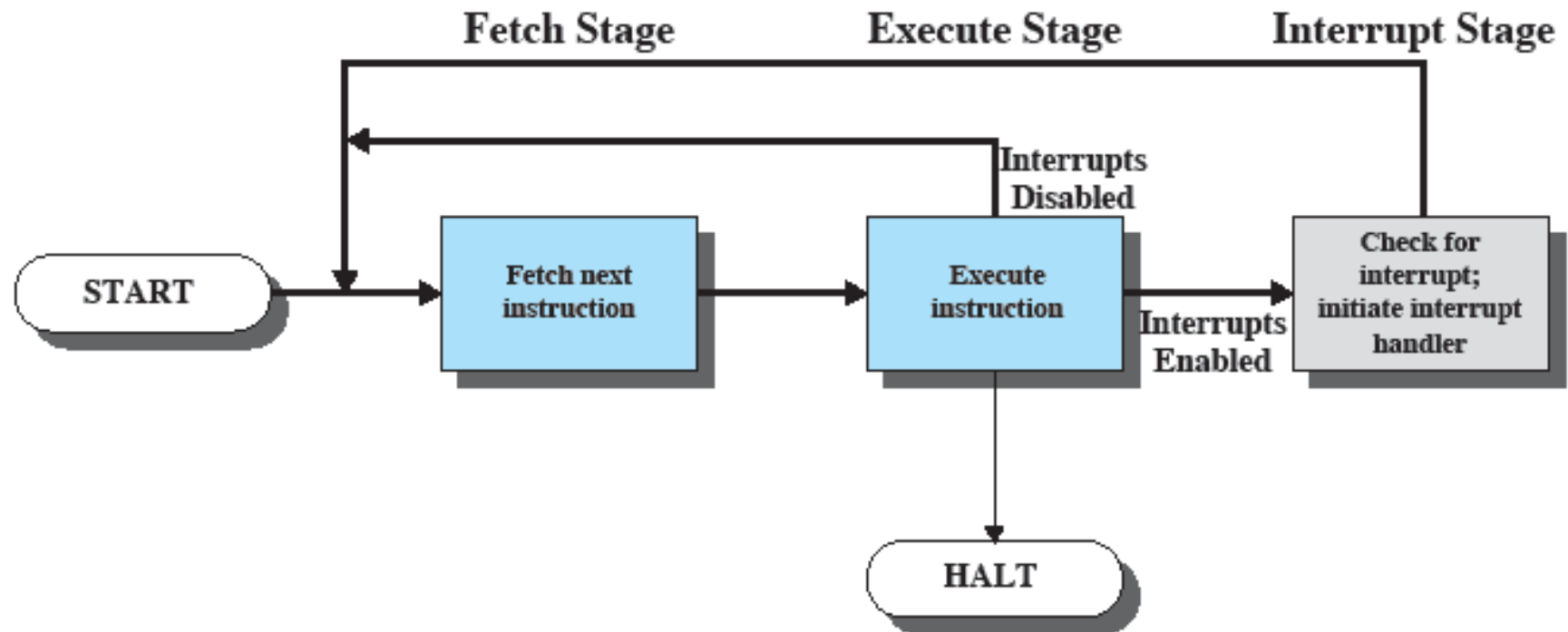
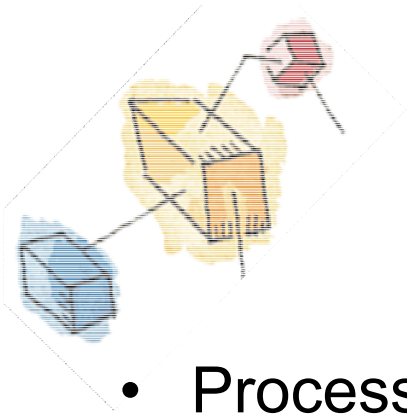


Figure 1.7 Instruction Cycle with Interrupts



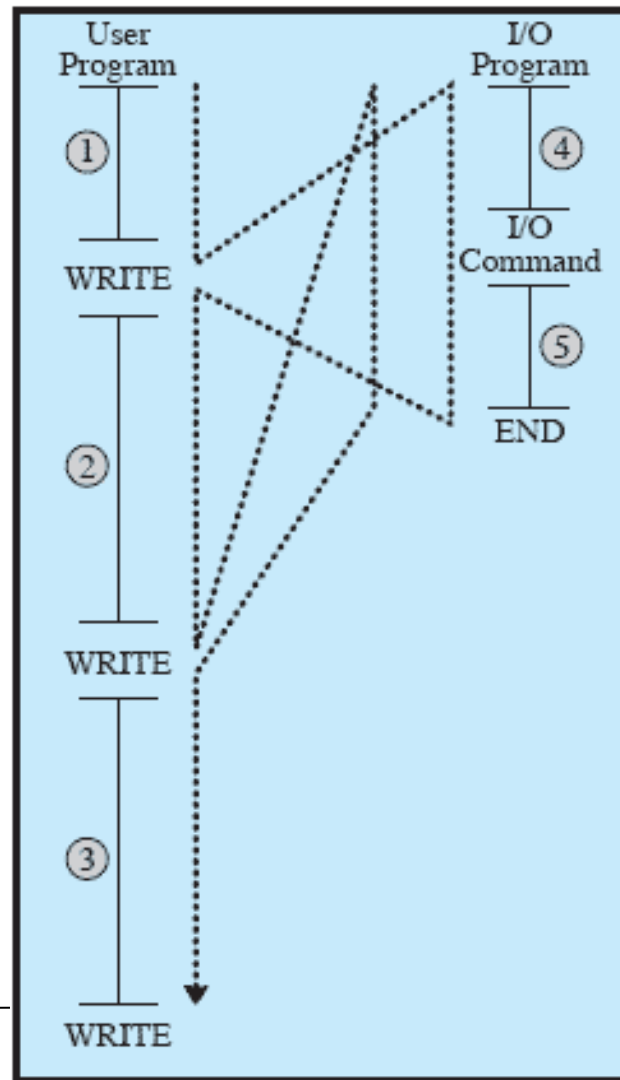


# Interrupt Stage

- Processor checks for interrupts
- If interrupt
  - Suspend execution of program
  - Execute interrupt-handler routine

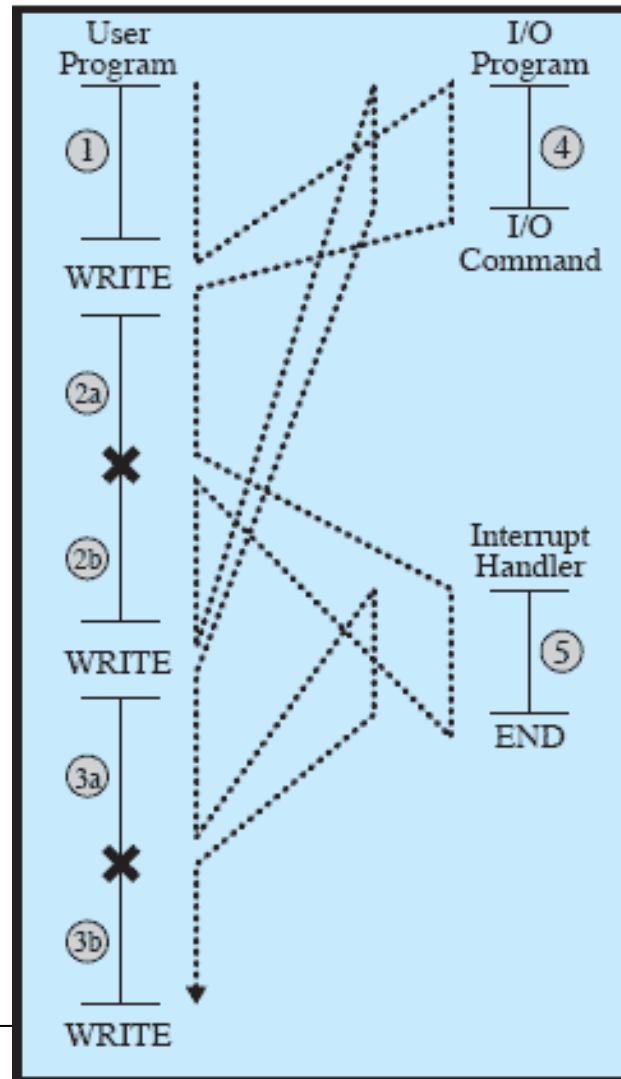


# Flow of Control without Interrupts

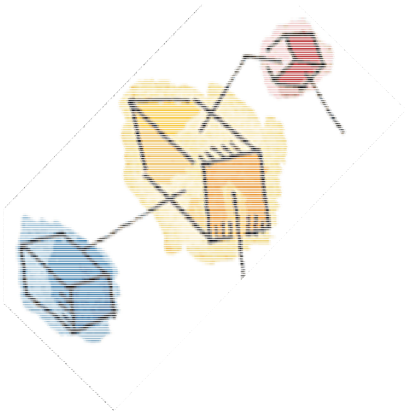


(a) No interrupts

# Interrupts



(b) Interrupts; short I/O wait



# Transfer of Control via Interrupts

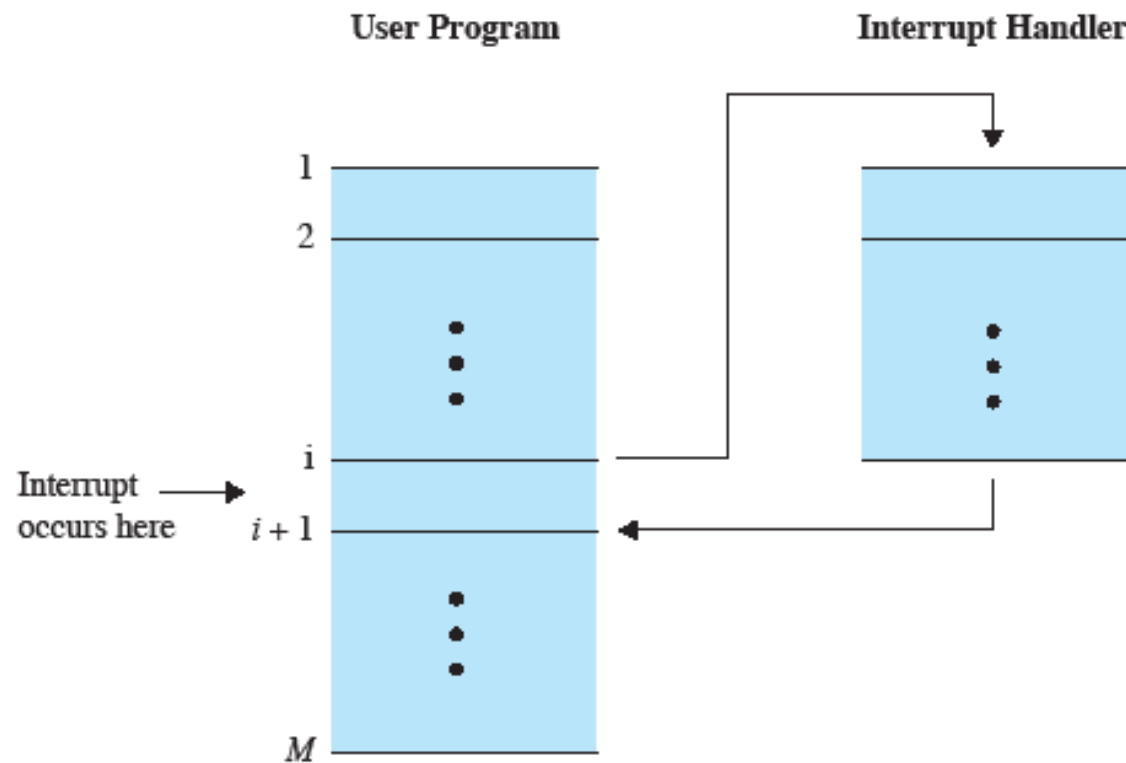


Figure 1.6 Transfer of Control via Interrupts

# Simple Interrupt Processing

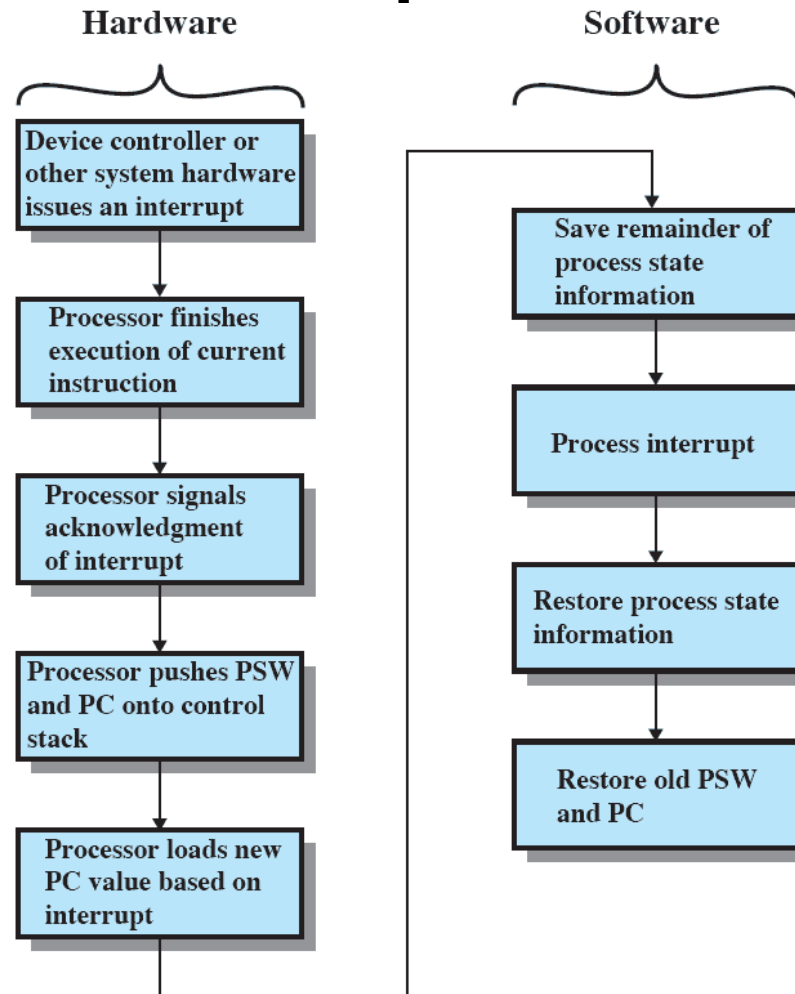
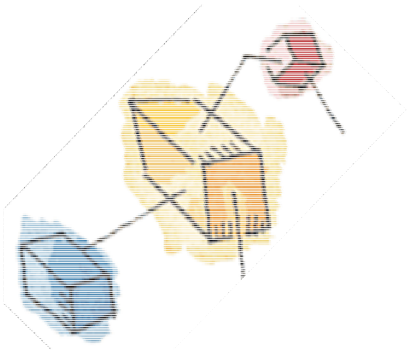


Figure 1.10 Simple Interrupt Processing



# Memory Hierarchy

- Major constraints in memory
  - Amount
  - speed
  - expense
- Memory must be able to keep up with the processor
- Cost of memory must be reasonable in relationship to the other components





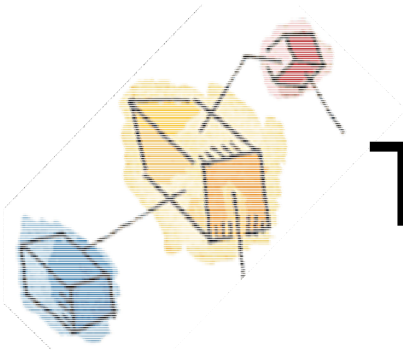
# Memory Relationships

Faster  
access time  
= greater  
cost per bit

Greater capacity =  
smaller cost per  
bit

Greater  
capacity =  
slower access  
speed





# The Memory Hierarchy

□ Going down the hierarchy:

- decreasing cost per bit
- increasing capacity
- increasing access time
- decreasing frequency of access to the memory by the processor

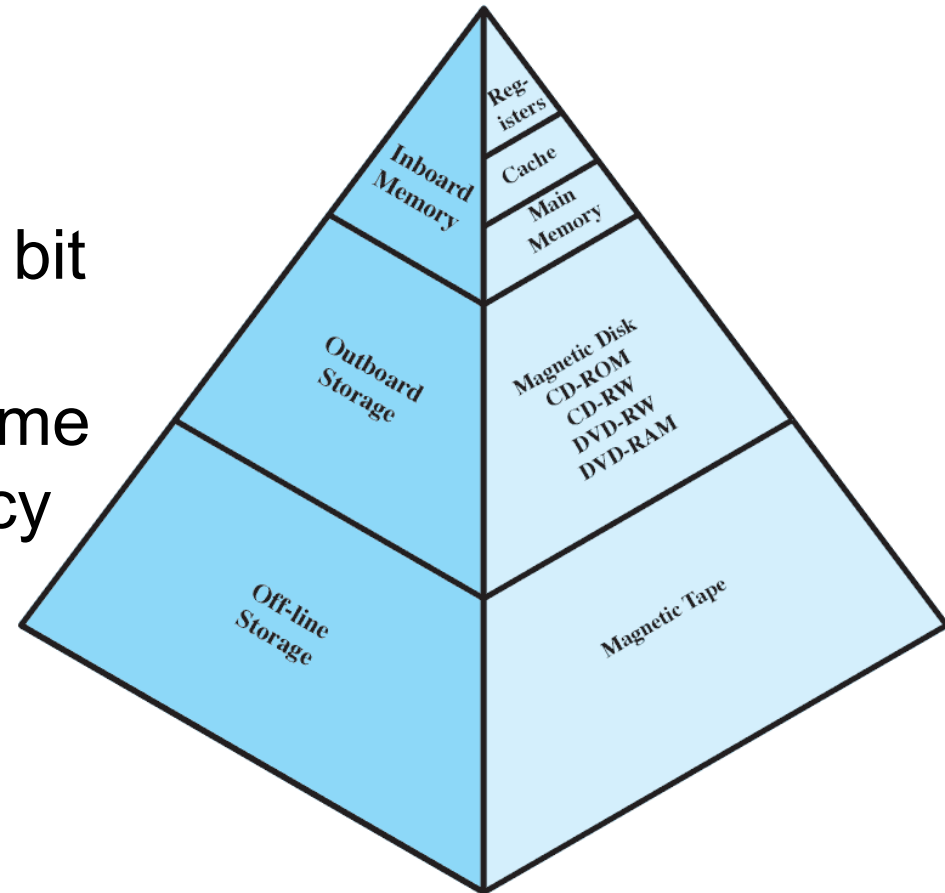
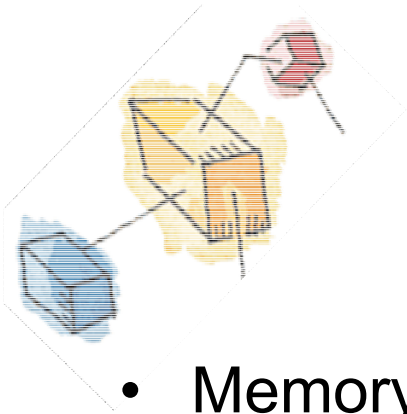


Figure 1.14 The Memory Hierarchy



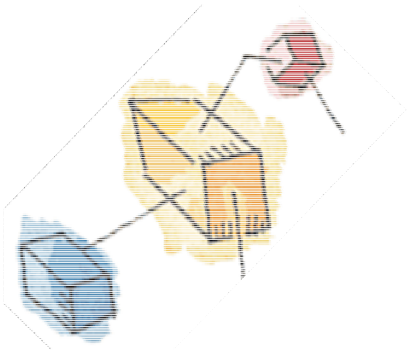




# Principle of locality

- Memory references by the processor tend to cluster
- Data is organized so that the percentage of accesses to each successively lower level is substantially less than that of the level above
- Can be applied across more than two levels of memory

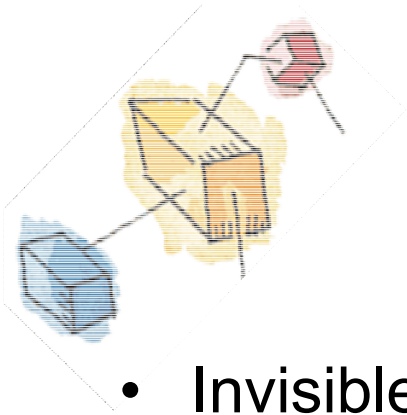




# Secondary Memory

- Also referred to as an auxiliary memory
  - External
  - Nonvolatile
  - Used to store program and data files

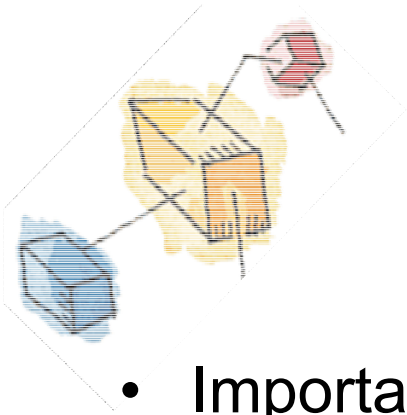




# Cache Memory

- Invisible to the OS
- Interacts with other memory management hardware
- Processor must access memory at least once per instruction cycle
- Processor execution is limited by memory cycle time
- Exploit the principle of locality with a small, fast memory



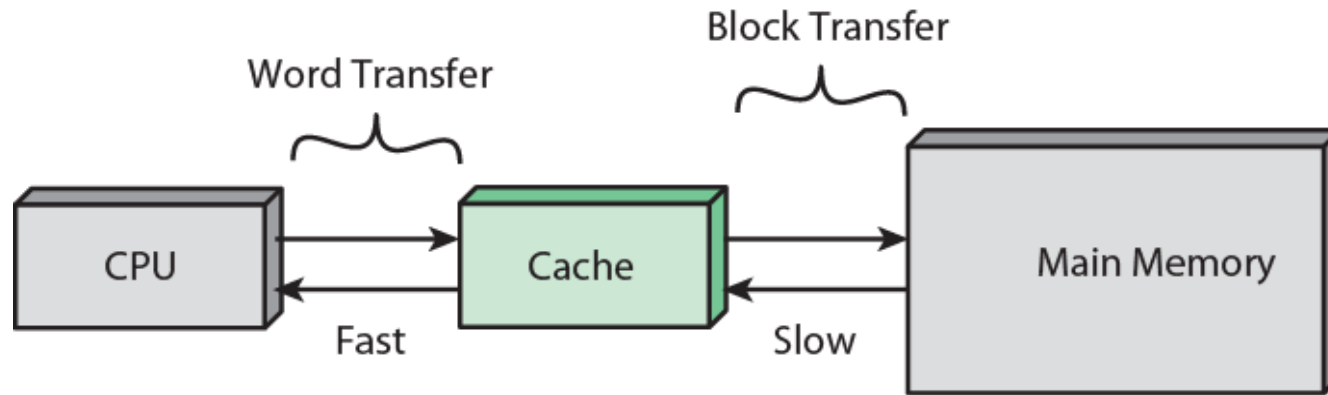


# Caching

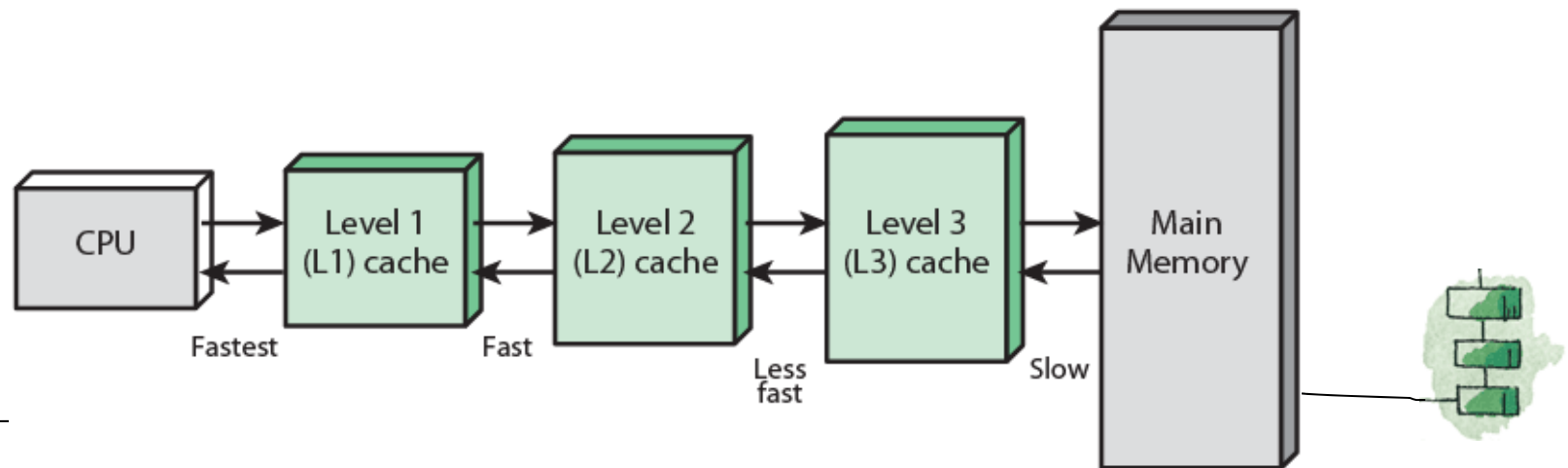
- Important principle, performed at many levels in a computer (in hardware, operating system, software)
- Information in use copied from slower to faster storage temporarily
- Faster storage (cache) checked first to determine if information is there
  - If it is, information used directly from the cache (fast)
  - If not, data copied to cache and used there
- Cache smaller than storage being cached
  - Cache size
  - mapping/replacement/write policies



# Cache and Main Memory



(a) Single cache



(b) Three-level cache organization

Cach

Line  
Number



**Figure 1.17 Cache/Main-Memory Structure**



# I/O Techniques



- When the processor encounters an instruction relating to I/O, it executes that instruction by issuing a command to the appropriate I/O module

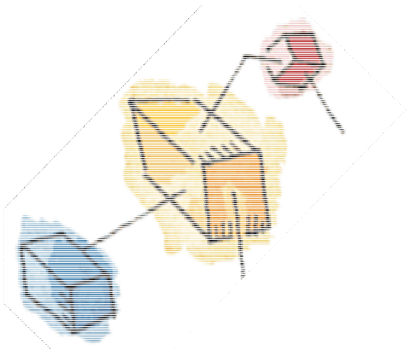
Three techniques are possible for I/O operations:

Programmed I/O

Interrupt-Driven I/O

Direct Memory Access (DMA)





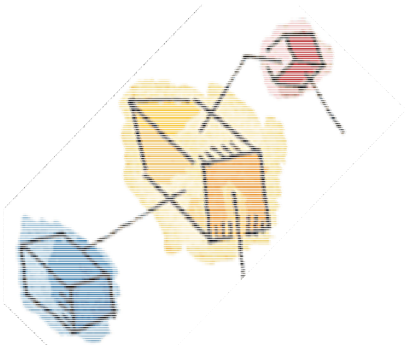
# Programmed I/O

- The I/O module performs the requested action then sets the appropriate bits in the I/O status register
- The processor periodically checks the status of the I/O module until it determines the instruction is complete
- With programmed I/O the performance level of the entire system is severely degraded





# Interrupt-driven I/O



Processor issues an I/O command to a module and then goes on to do some other useful work

The processor executes the data transfer and then resumes its former processing

The I/O module will then interrupt the processor to request service when it is ready to exchange data with the processor

More efficient than Programmed I/O but still requires active intervention of the processor to transfer data between memory and an I/O module





# Interrupt-Driven I/O drawbacks

- Transfer rate is limited by the speed with which the processor can test and service a device
- The processor is tied up in managing an I/O transfer
  - a number of instructions must be executed for each I/O transfer





# Direct Memory Access (DMA)

- Performed by a separate module on the system bus or incorporated into an I/O module

When the processor wishes to read or write data it issues a command to the DMA module containing:

- whether a read or write is requested
- the address of the I/O device involved
- the starting location in memory to read/write
- the number of words to be read/written





# Direct Memory Access (DMA)

- Transfers the entire block of data directly to and from memory without going through the processor
  - processor is involved only at the beginning and end of the transfer
  - processor executes more slowly during a transfer when processor access to the bus is required
- More efficient than interrupt-driven or programmed I/O





# Symmetric Multiprocessors (SMP)

- A stand-alone computer system with the following characteristics:
  - two or more similar processors of comparable capability
  - processors share the same main memory and are interconnected by a bus or other internal connection scheme
  - processors share access to I/O devices
  - all processors can perform the same functions
  - the system is controlled by an integrated operating system that provides interaction between processors and their programs at the job, task, file, and data element levels





# SMP Advantages

## Performance

- a system with multiple processors will yield greater performance if work can be done in parallel

## Scaling

- vendors can offer a range of products with different price and performance characteristics

## Availability

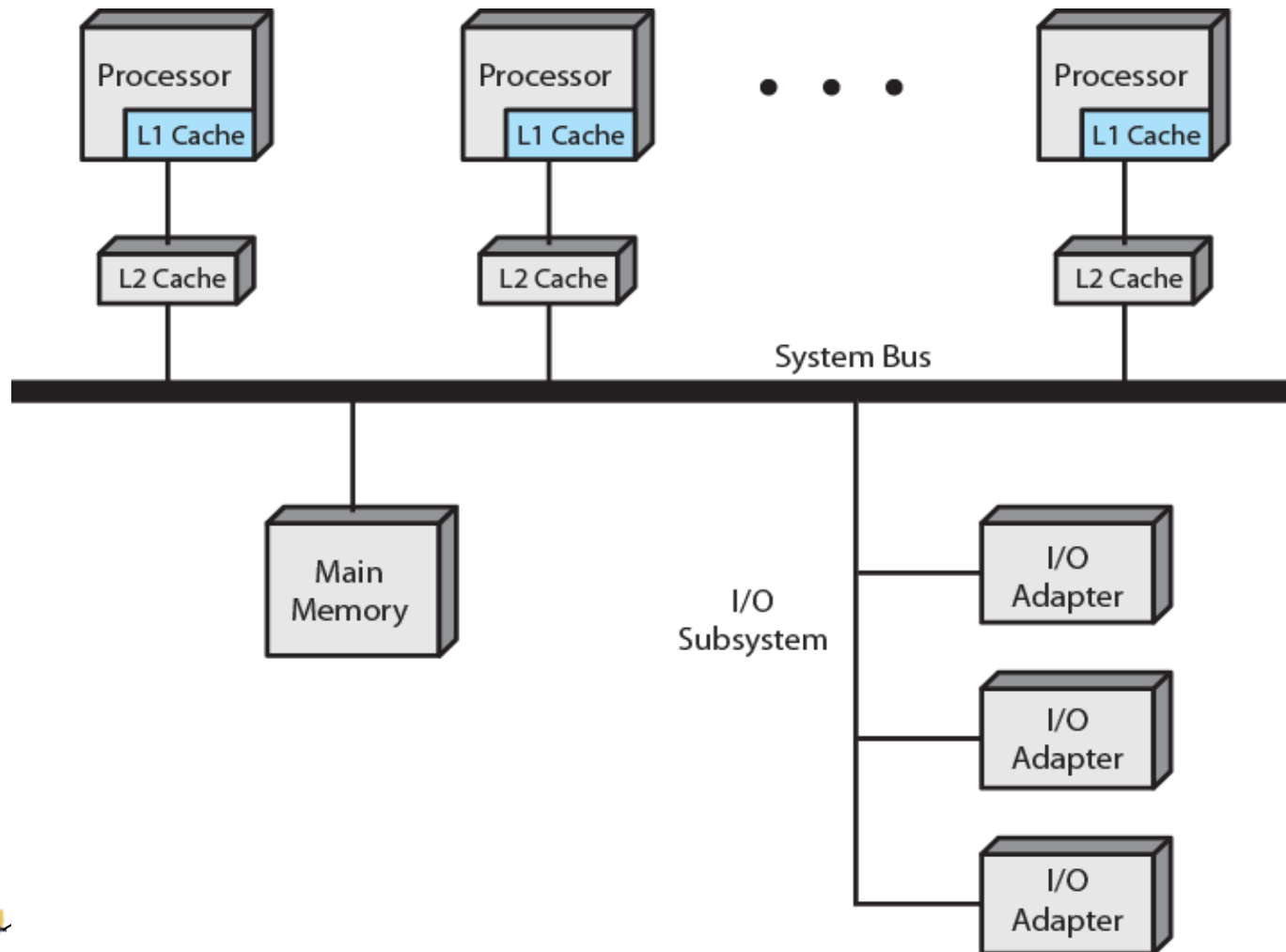
- the failure of a single processor does not halt the machine

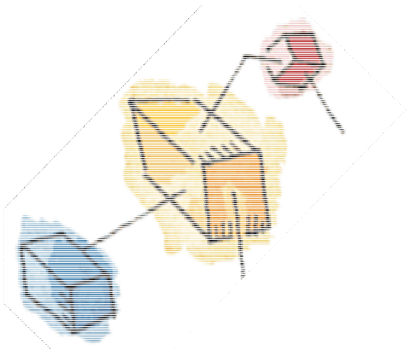
## Incremental Growth

- an additional processor can be added to enhance performance



# SMP Organization





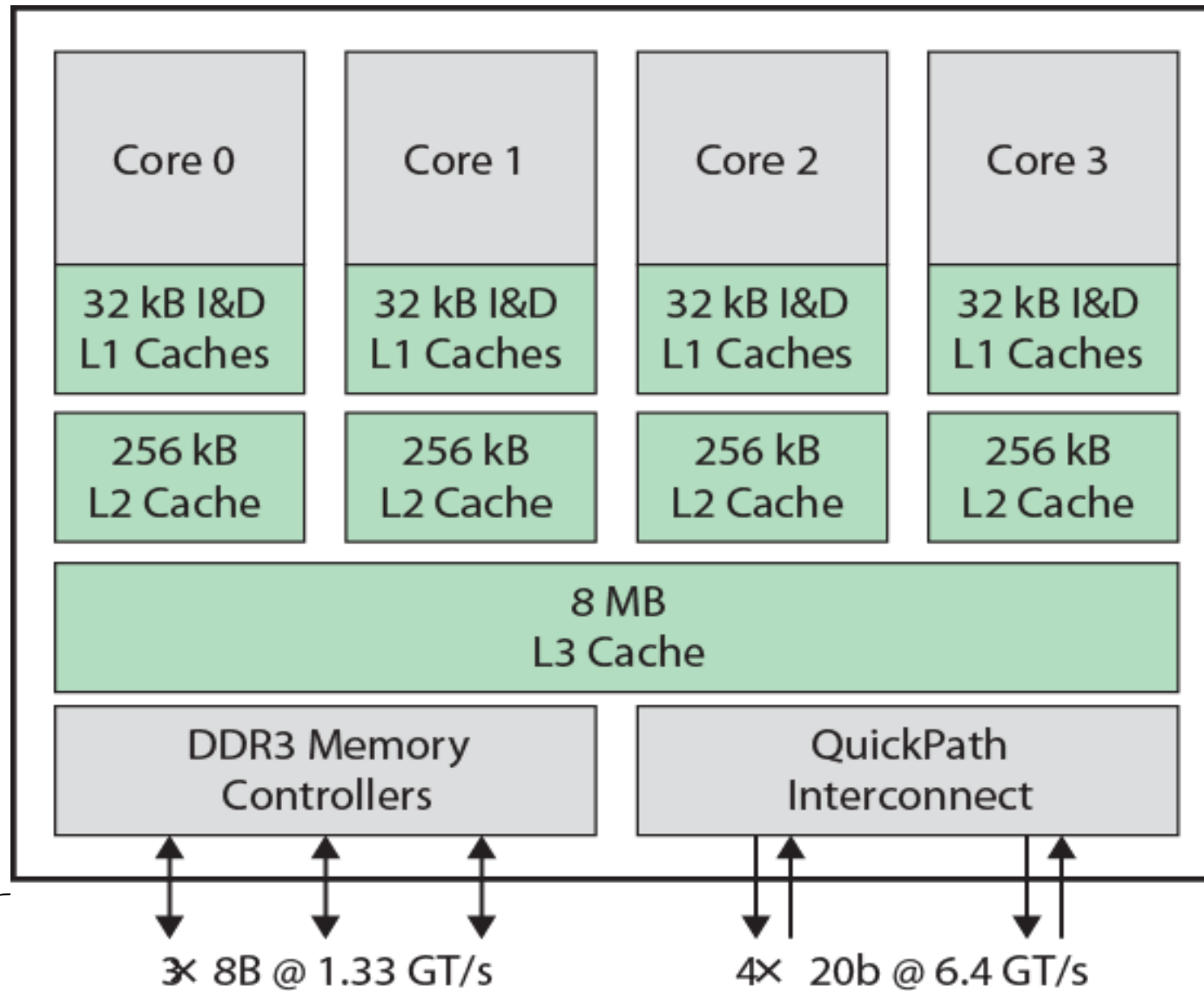
# Multicore Computer

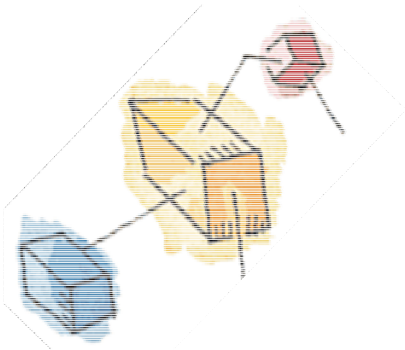
- Also known as a chip multiprocessor
- Combines two or more processors (cores) on a single piece of silicon (die)
  - each core consists of all of the components of an independent processor
- In addition, multicore chips also include L2 cache and in some cases L3 cache





# Intel Core i7





# Summary

- Basic Elements
  - processor, main memory, I/O modules, system bus
  - Instruction execution
    - processor-memory, processor-I/O, data processing, control
  - Interrupt/Interrupt Processing
  - Memory Hierarchy
  - Cache/cache principles
  - Multiprocessor/multicore

