Operating Systems: Internals and Design Principles

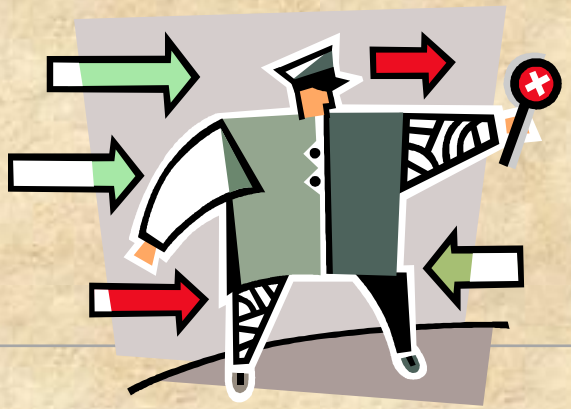
Chapter 1 Computer System Overview

Seventh Edition

By William Stallings

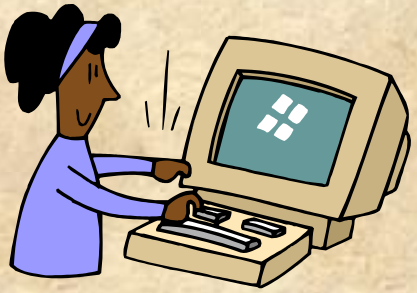
Operating System

- Exploits the hardware resources of one or more processors to provide a set of services to system users
- Manages the processor, secondary memory and I/O devices



Basic Elements

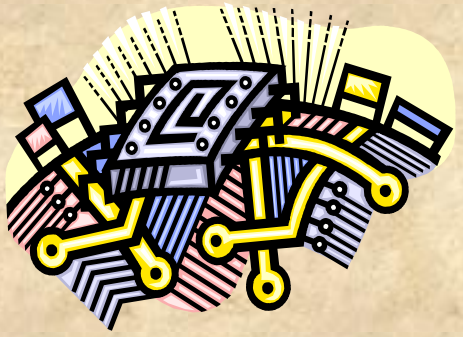
- Processor
- I/O Modules
- Main Memory
- System Bus



Processor

Controls the operation of the computer

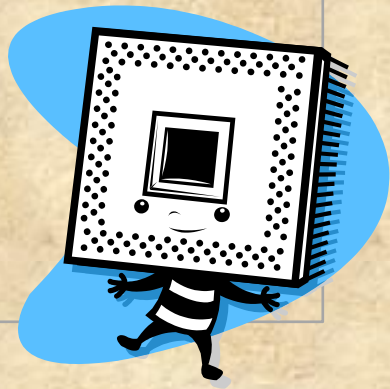
Performs the data processing functions



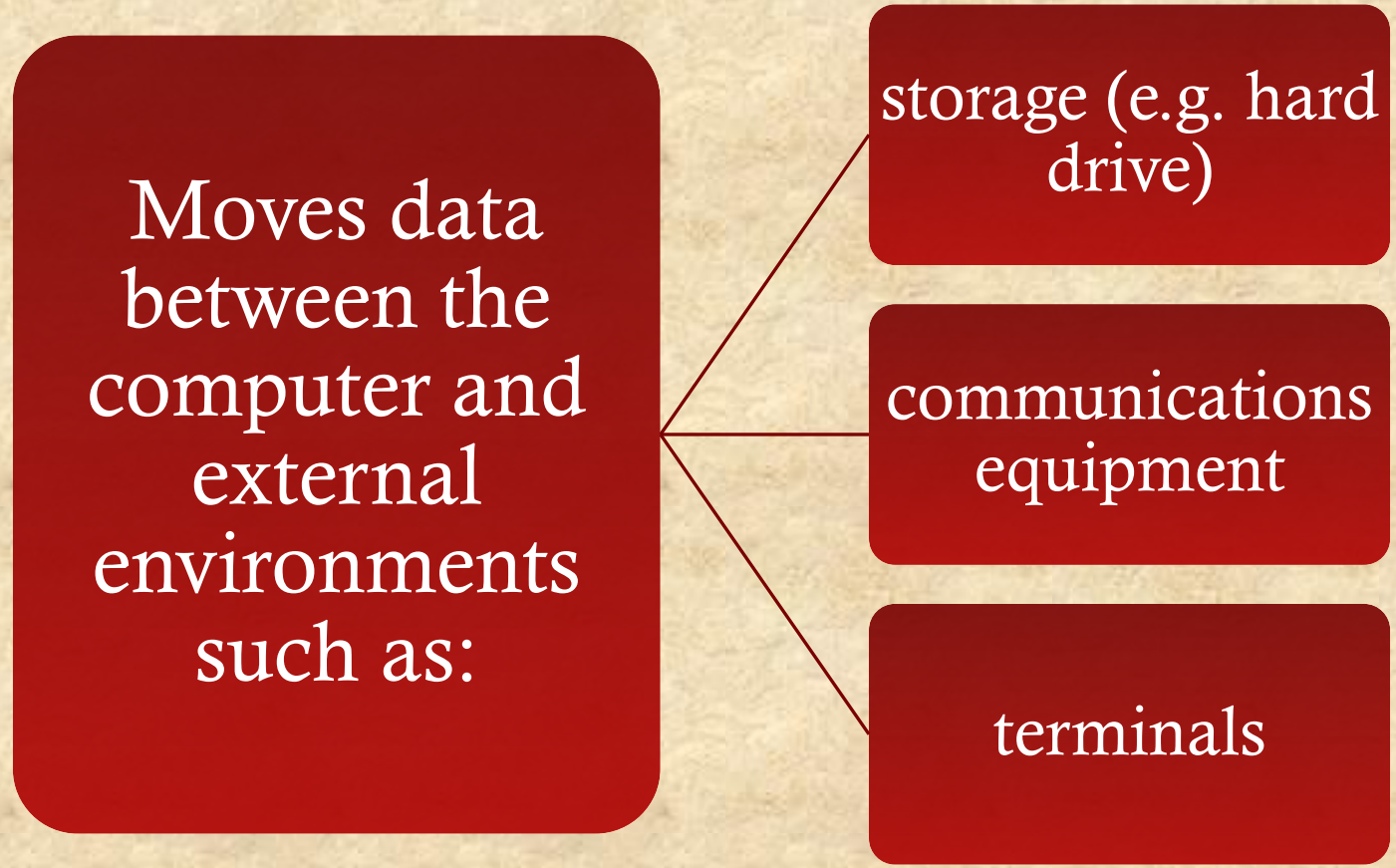
Referred to as the *Central Processing Unit* (CPU)

Main Memory

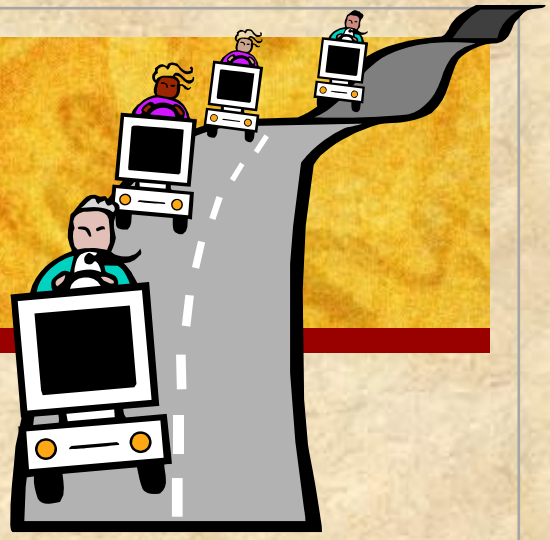
- Volatile: contents of the memory are lost when the computer is shut down
- Also referred to as real memory or primary memory



I/O Modules


Moves data between the computer and external environments such as:
storage (e.g. hard drive)
communications equipment
terminals

System Bus



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

Top-Level View

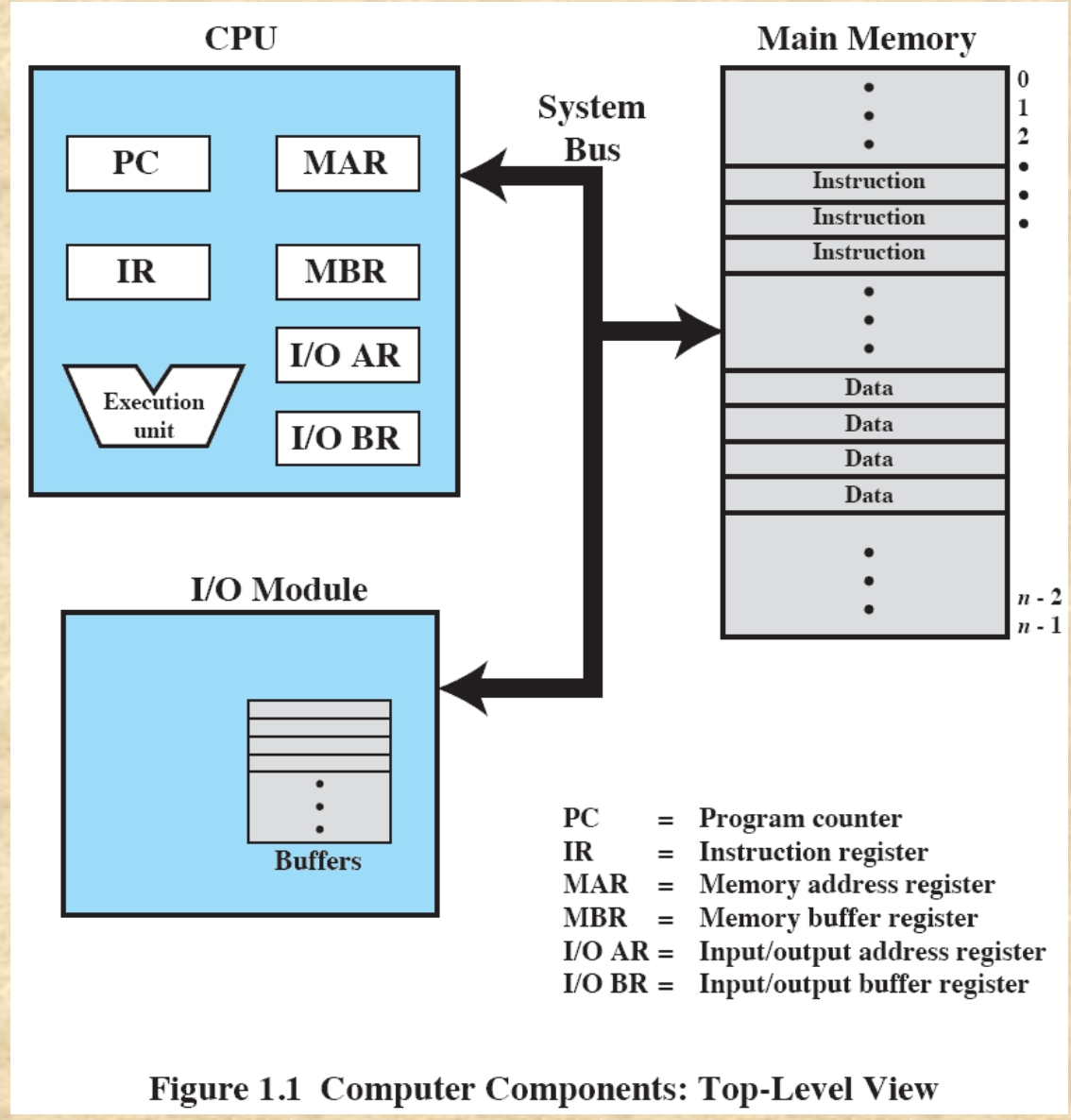
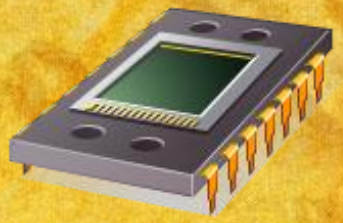


Figure 1.1 Computer Components: Top-Level View



Microprocessor

- Invention that brought about desktop and handheld computing
- Processor on a single chip
- Fastest general purpose processor
- Multiprocessor capability
  - Each chip contains multiple processors (cores); each core may execute multiple threads

Graphical Processing Units (GPUs)

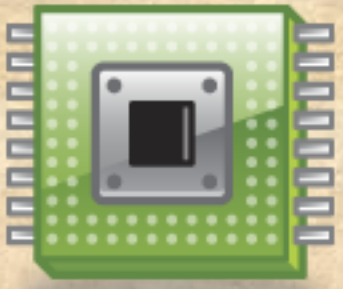
- Provide efficient computation on arrays of data using Single-Instruction Multiple Data (SIMD) techniques
- Used for general numerical processing
- Physics simulations for games
- Computations on large spreadsheets

Digital Signal Processors (DSPs)

- Deal with streaming signals such as audio or video
- Used to be embedded in devices like modems
- Encoding/decoding speech and video (codecs)
- Support for encryption and security

System on a Chip (SoC)

- To satisfy the requirements of handheld devices & embedded systems, the microprocessor is giving way to the SoC
- Components such as DSPs, GPUs, codecs and main memory, in addition to the CPUs and caches, are on the same chip



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

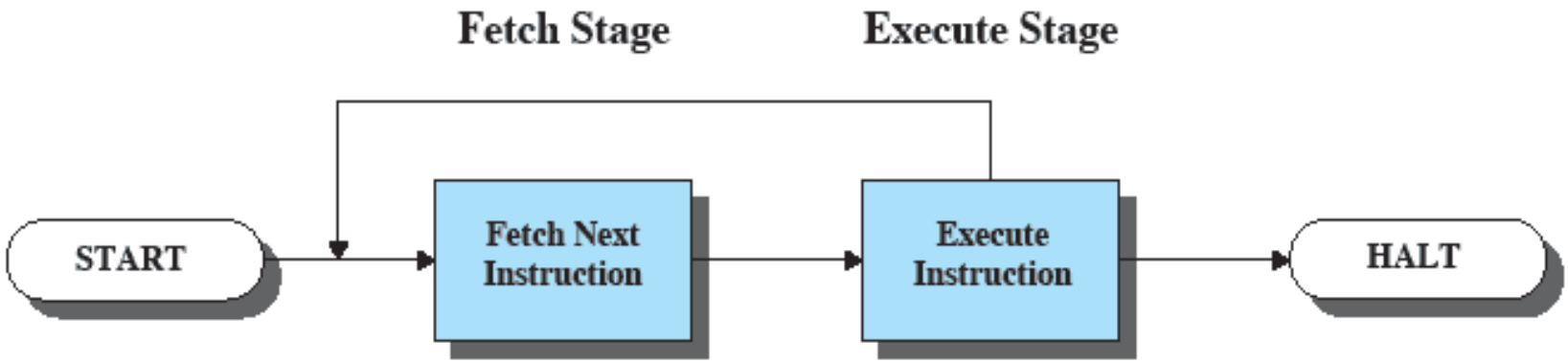
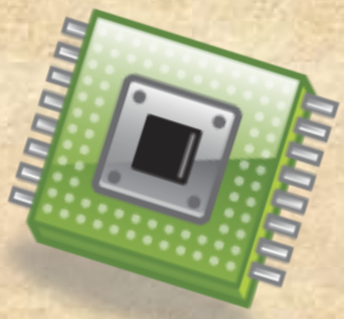


Figure 1.2 Basic Instruction Cycle

Instruction Fetch and Execute

- 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

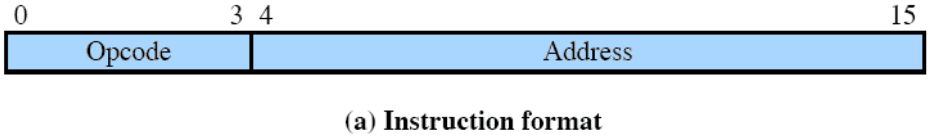


Instruction Register (IR)

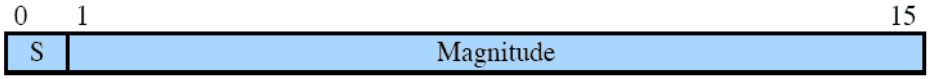
Fetched instruction is loaded into Instruction Register (IR)

- Processor interprets the instruction and performs required action:
  - Processor-memory
  - Processor-I/O
  - Data processing
  - Control

Characteristics of a Hypothetical Machine



(a) Instruction format



(b) Integer format

Program counter (PC) = Address of instruction
Instruction register (IR) = Instruction being executed
Accumulator (AC) = Temporary storage

(c) Internal CPU registers

0001 = Load AC from memory
0010 = Store AC to memory
0101 = Add to AC from memory

(d) Partial list of opcodes

Figure 1.3 Characteristics of a Hypothetical Machine

Example of Program Execution

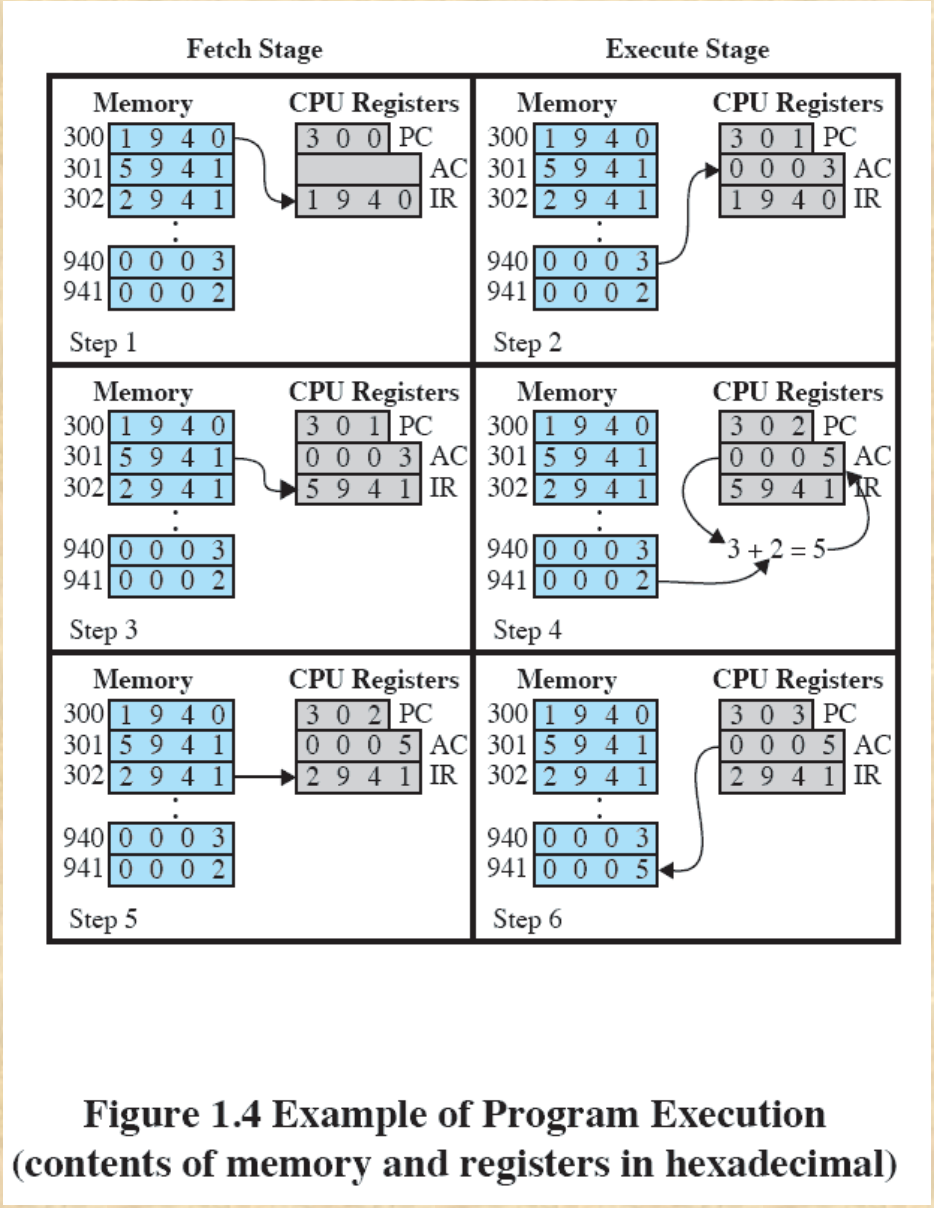


Figure 1.4 Example of Program Execution (contents of memory and registers in hexadecimal)

Interrupts

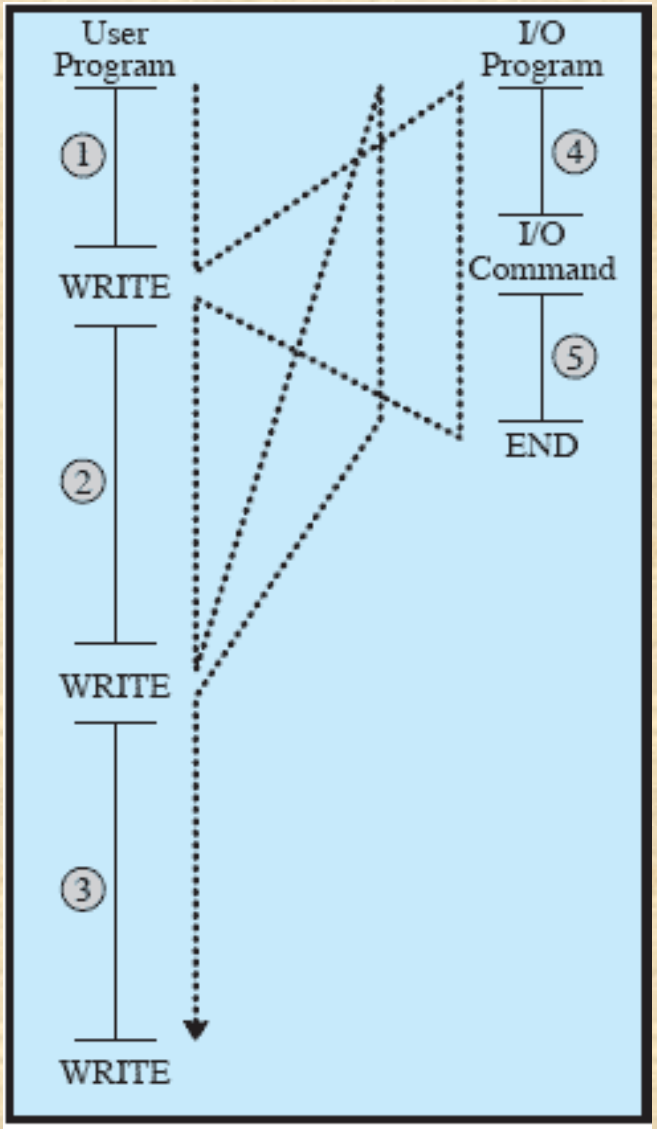
- Interrupt the normal sequencing of the processor
- Provided to improve processor utilization
  - most I/O devices are slower than the processor
  - processor must pause to wait for device
  - wasteful use of the processor

Common Classes of Interrupts

Table 1.1 Classes of Interrupts

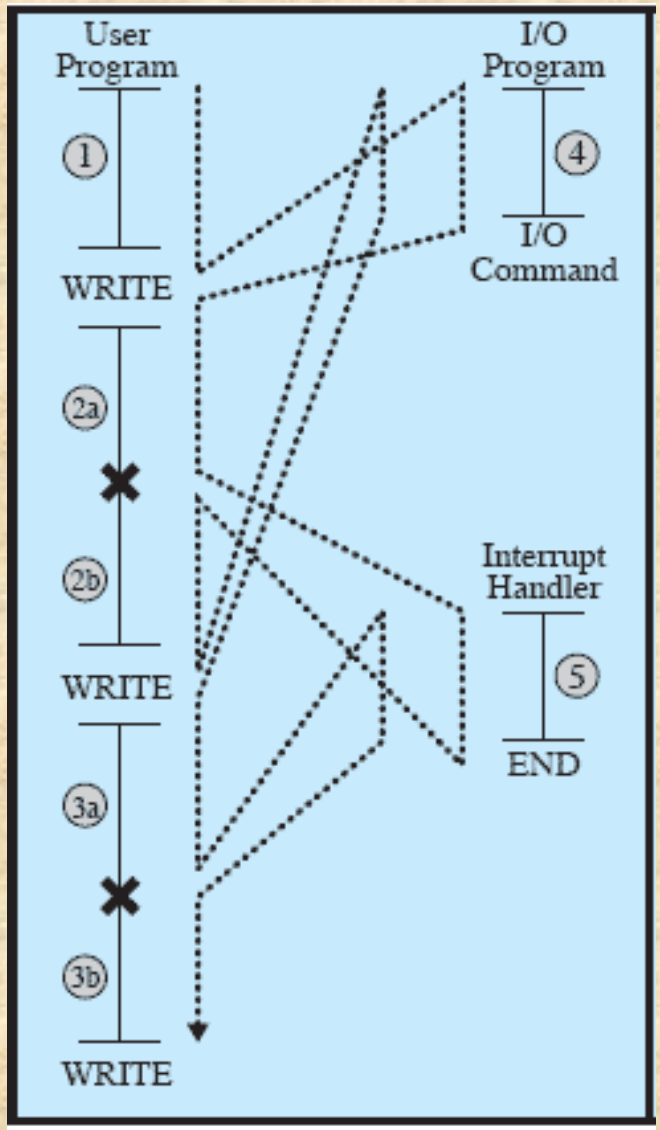
Program	Generated by some condition that occurs as a result of an instruction execution, such as arithmetic overflow, division by zero, attempt to execute an illegal machine instruction, and reference outside a user's allowed memory space.
Timer	Generated by a timer within the processor. This allows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

Flow of Control Without Interrupts



(a) No interrupts

Interrupts: Short I/O Wait



(b) Interrupts; short I/O wait

Transfer of Control via Interrupts

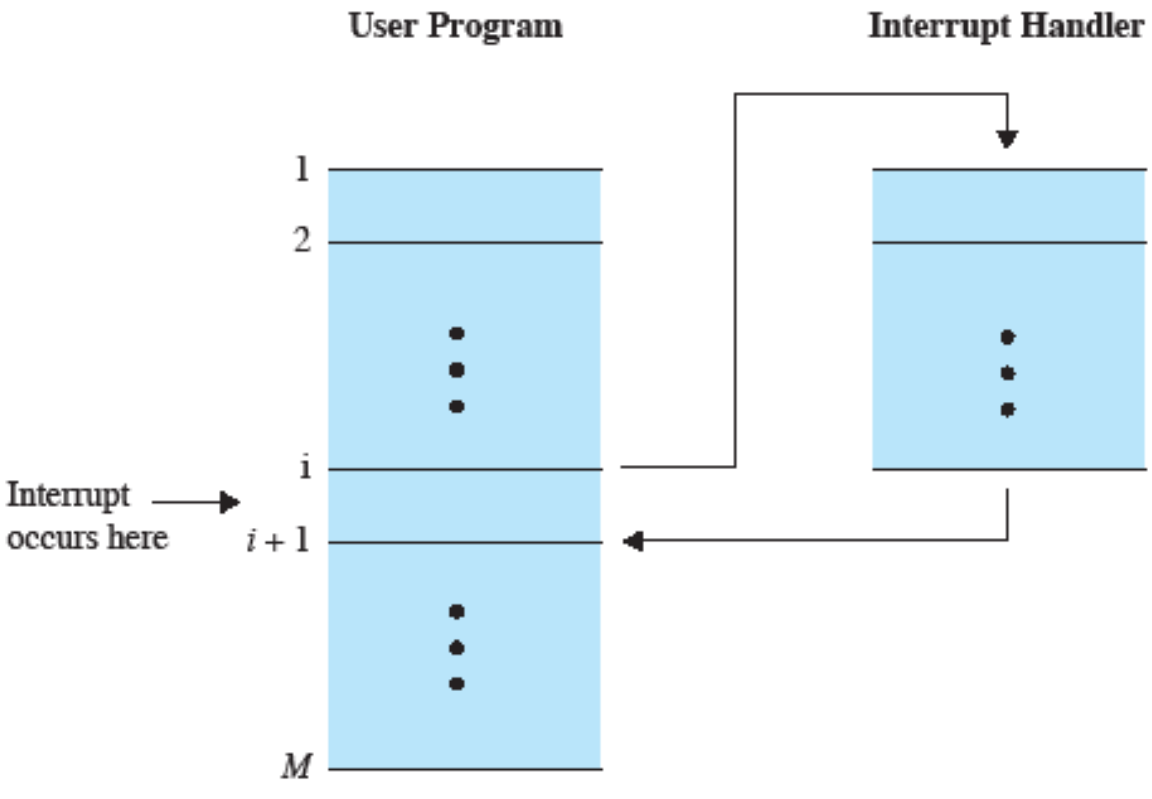


Figure 1.6 Transfer of Control via Interrupts

Instruction Cycle With Interrupts

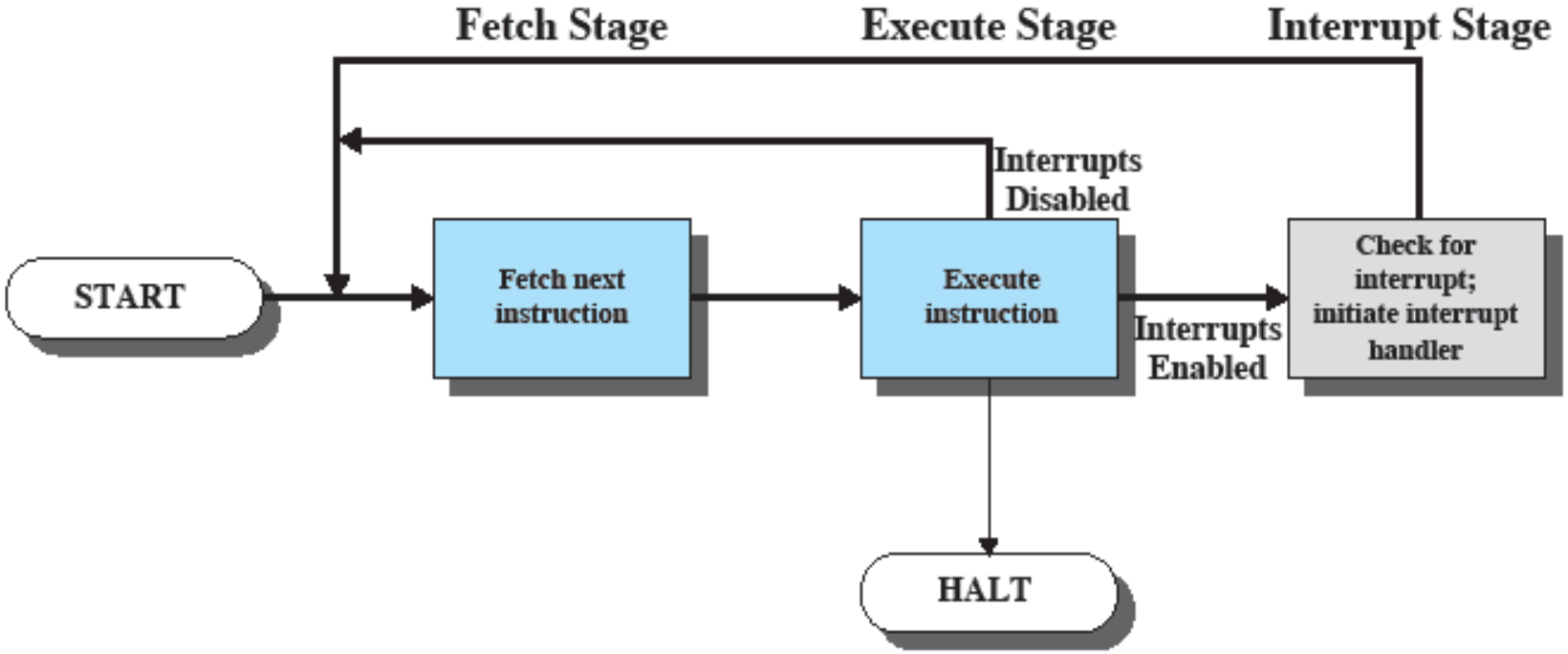
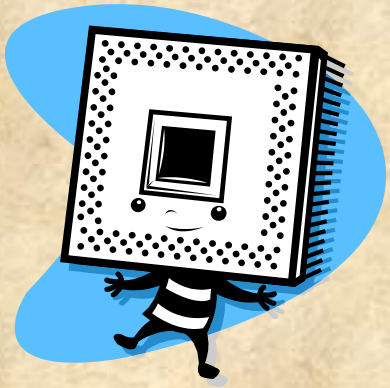


Figure 1.7 Instruction Cycle with Interrupts

Simple Interrupt Processing



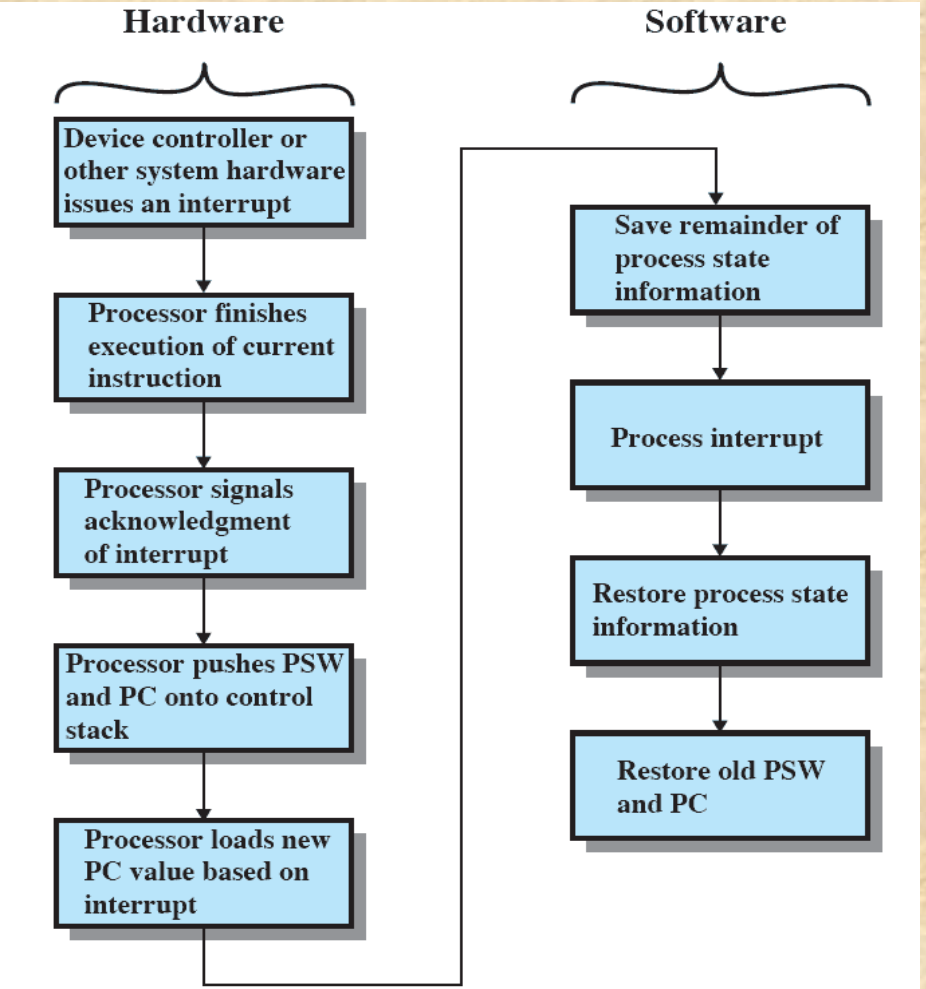


Figure 1.10 Simple Interrupt Processing

Multiple Interrupts

An interrupt occurs while another interrupt is being processed

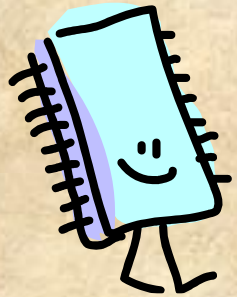
- e.g. receiving data from a communications line and printing results at the same time

Two approaches:

- disable interrupts while an interrupt is being processed
- use a priority scheme

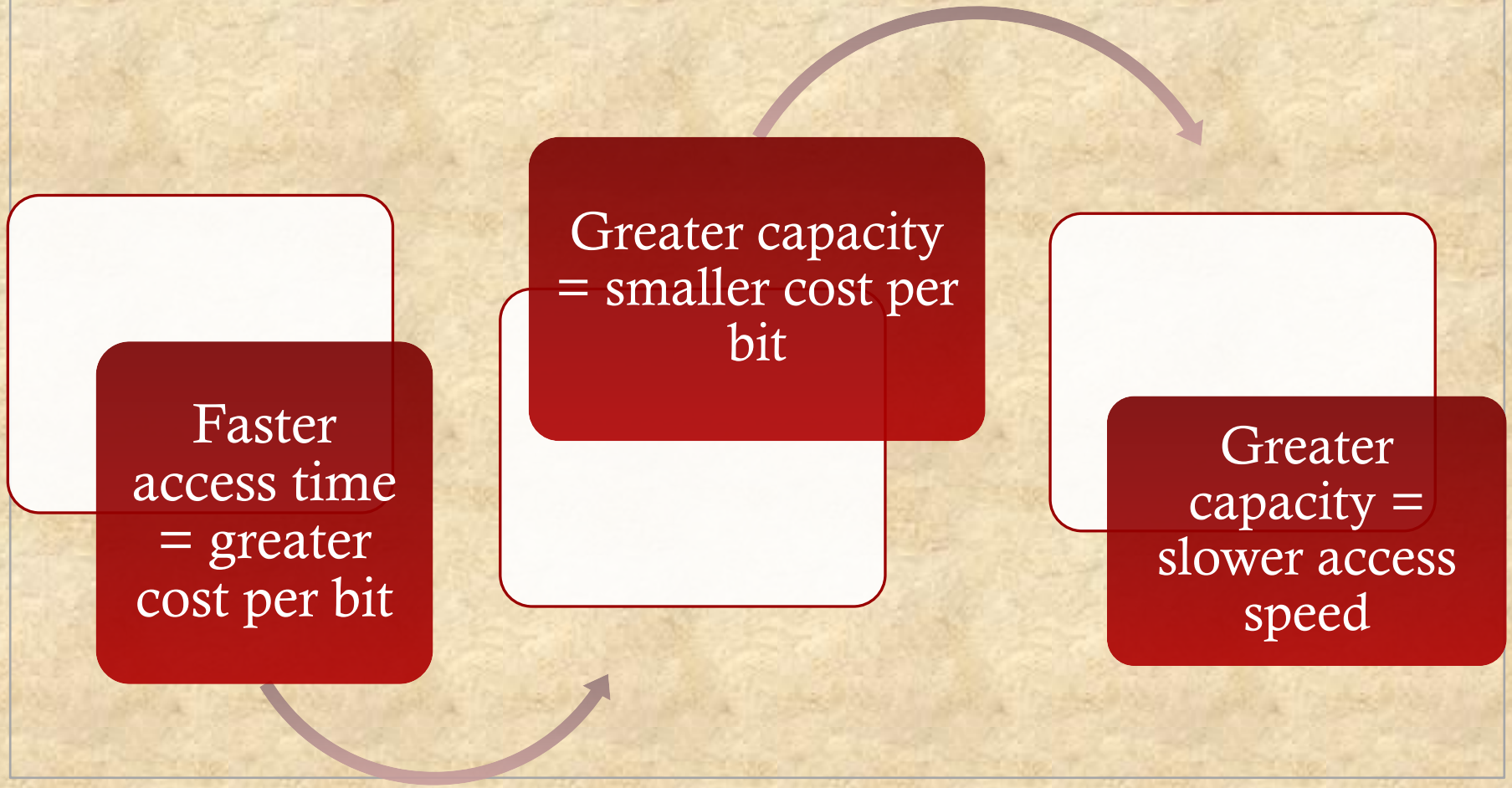
Memory Hierarchy

- 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



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

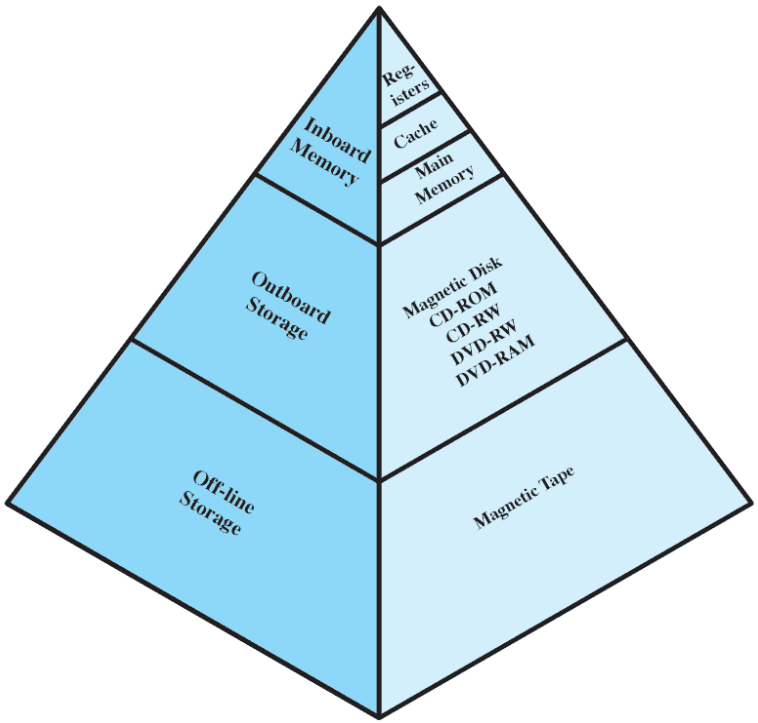
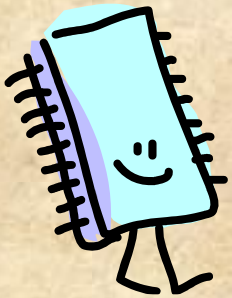


Figure 1.14 The Memory Hierarchy

Performance of a Simple Two-Level Memory



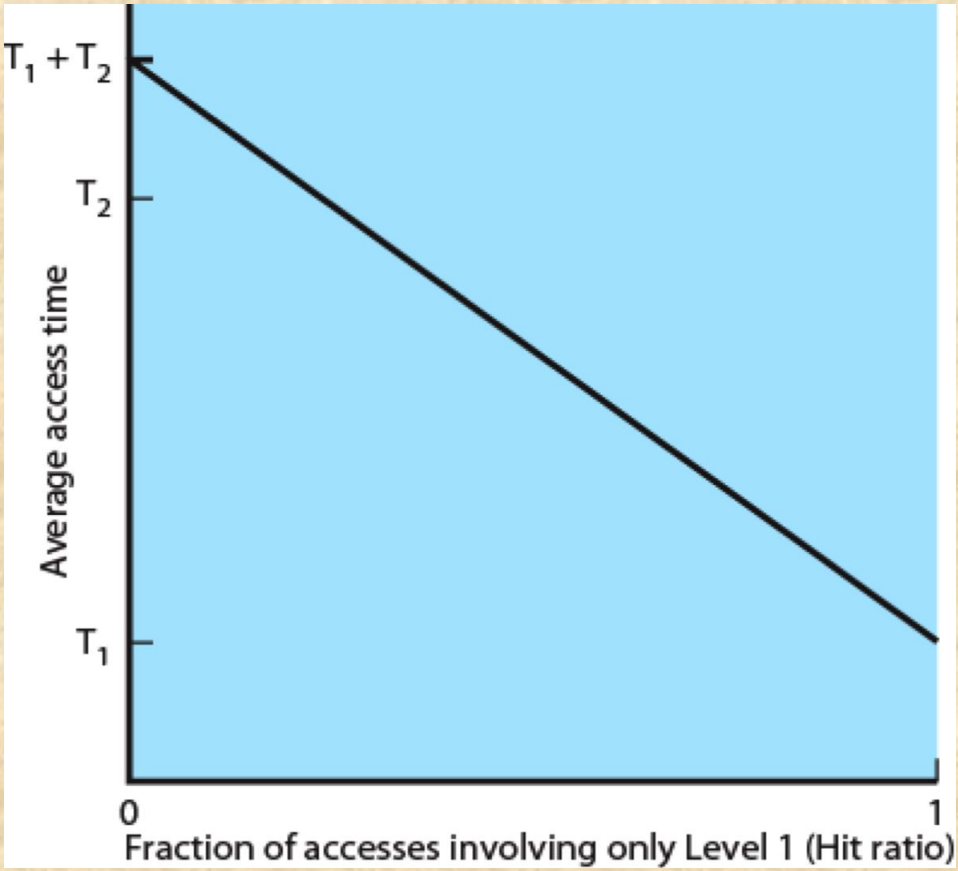
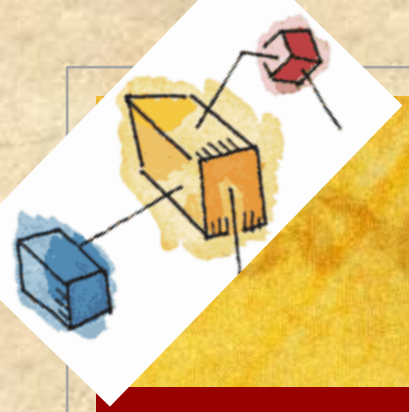
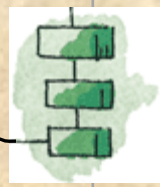


Figure 1.15 Performance of a Simple Two-Level Memory



Example

- Speed of fast memory (T1): 0.1
- Speed of slow memory (T2): 1.0
- Hit ratio for fast memory: .95
- Average access time = .15
  (.95 * .1) +(.05 * (1.0 + 0.1))



Principle of Locality

- Memory references by the processor tend to cluster
  - Spatial locality: a reference to one memory location usually means nearby locations will be referenced too
  - Temporal locality: if a location is referenced once, it will probably be accessed again soon.

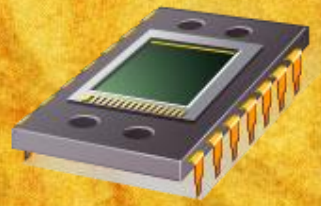
Principle of Locality

- In a hierarchical memory, data can be organized so that the percentage of accesses to each successively lower level is substantially less than that of the level above
  - i.e., locations in current locality should be in the faster levels of memory.
- Can be applied across more than two levels of memory

Memory Hierarchy

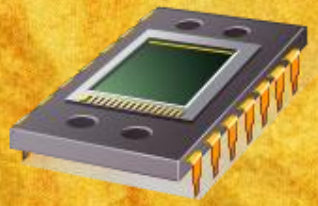
- Cache Memory: fastest; volatile; contains a subset of main memory
  - Most processors have more than one level
- Main Memory: slower; also volatile
- Disk: slowest, non-volatile, used to store programs and data permanently

Cache Memory



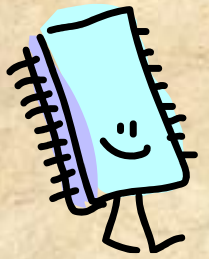
- Invisible to the OS
- Processor must access memory at least once per instruction cycle
- Processor execution time is limited by memory cycle time
- Exploit the principle of locality with a small, fast memory

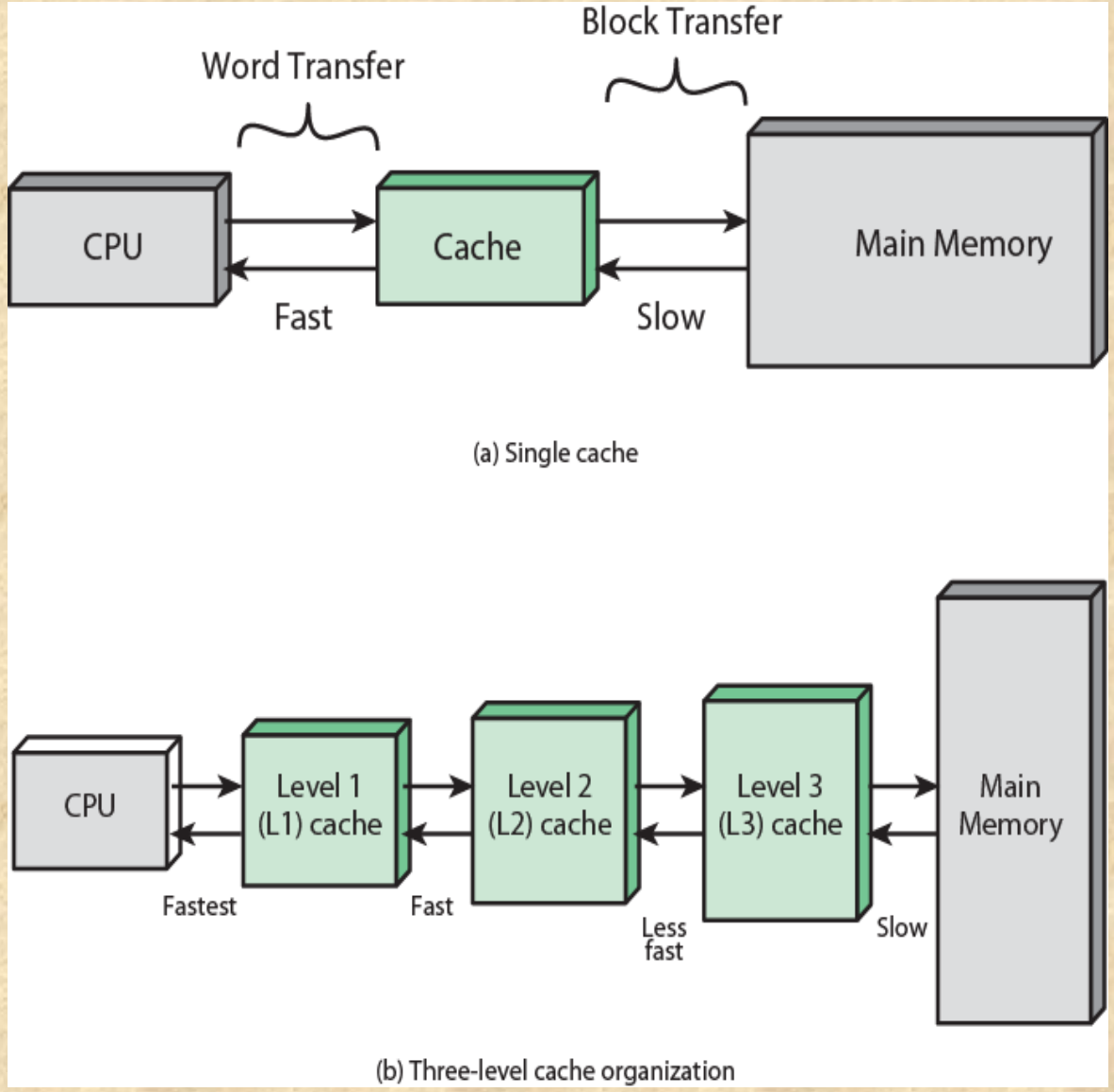
Cache Principles



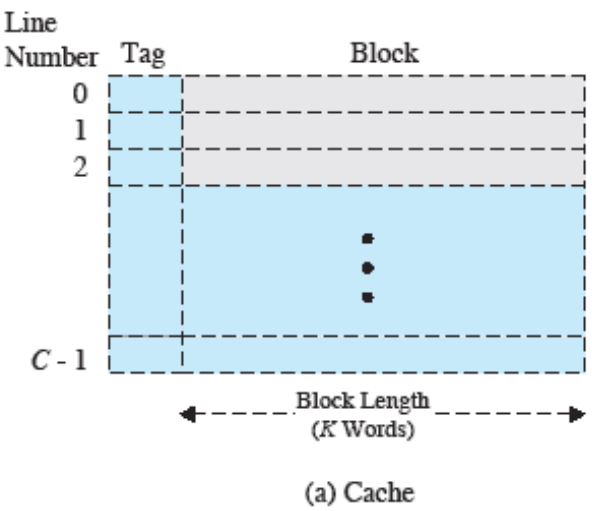
- On a memory reference, the processor first checks cache
- If not found, a block of memory is read into cache
- Locality makes it likely that many future memory references will be to other bytes in the block

Cache and Main Memory

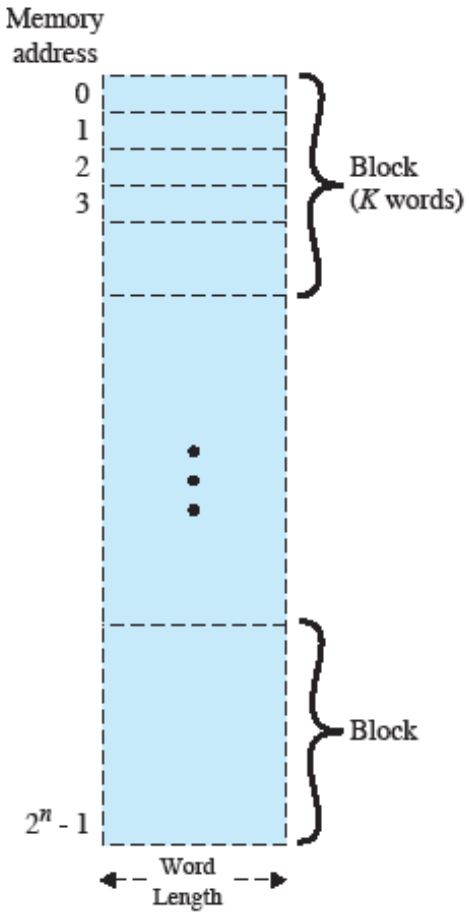




Cache/Main-Memory Structure



(a) Cache



(b) Main memory

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

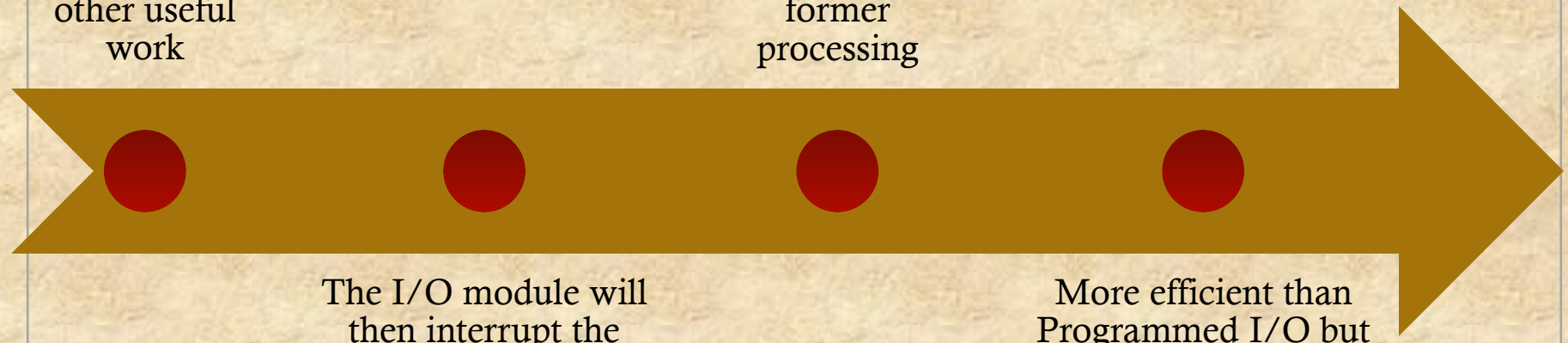
Interrupt-Driven I/O

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

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

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

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



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

- 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

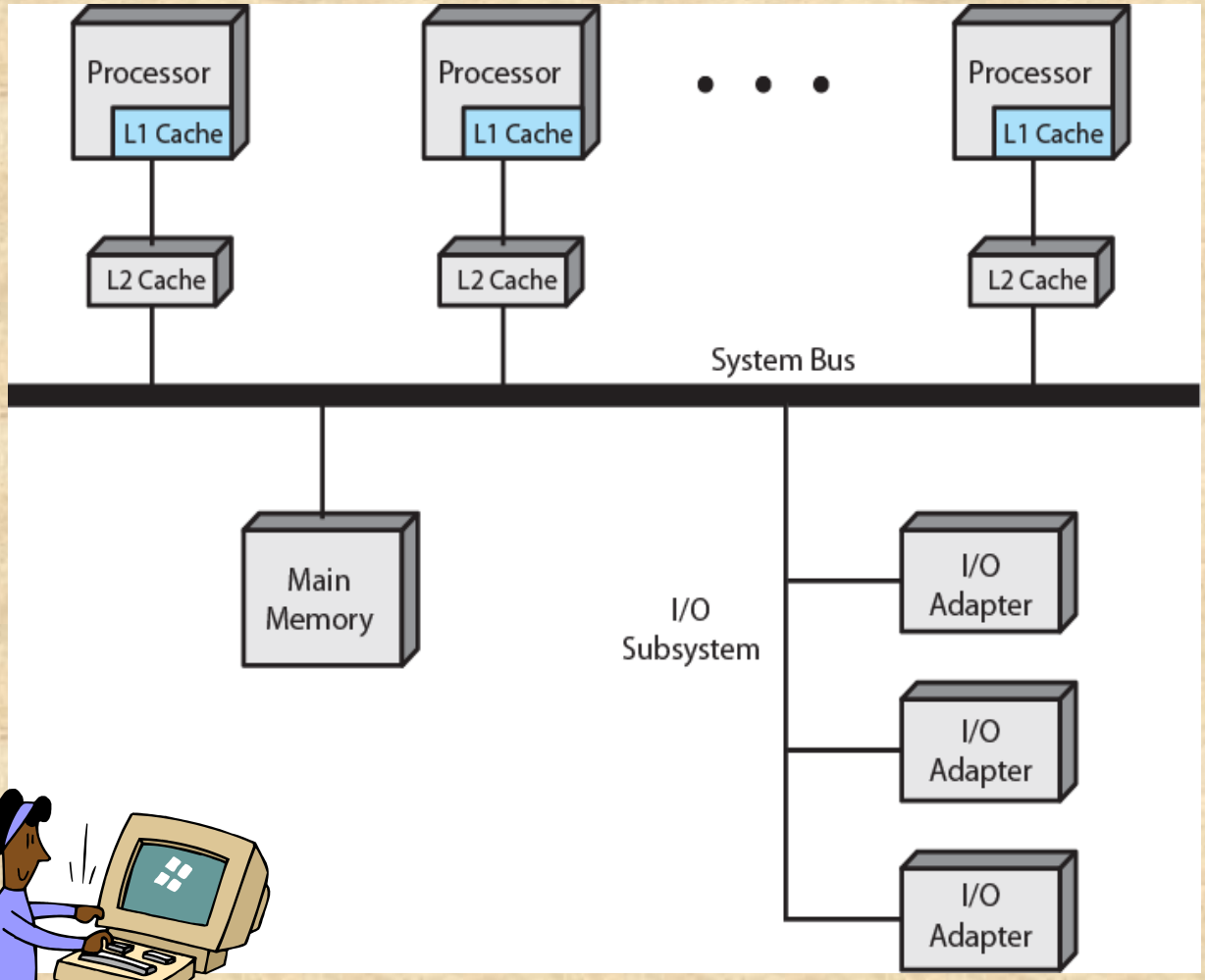
Availability

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

Incremental Growth

- an additional processor can be added to enhance performance

SMP Organization



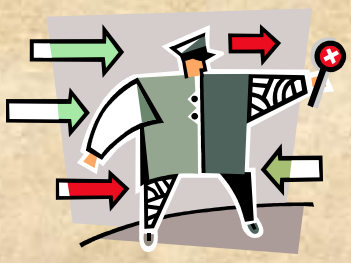
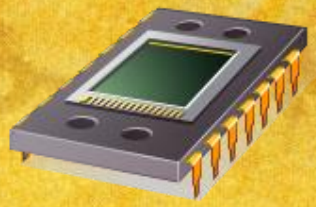
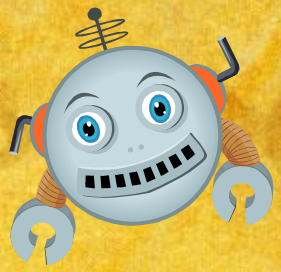


Figure 1.19 Symmetric Multiprocessor Organization



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

Supports two forms of external communications to other chips:

DDR3 Memory Controller

- brings the memory controller for the DDR (double data rate) main memory onto the chip
- with the memory controller on the chip the Front Side Bus is eliminated

QuickPath Interconnect (QPI)

- enables high-speed communications among connected processor chips

Intel Core i7

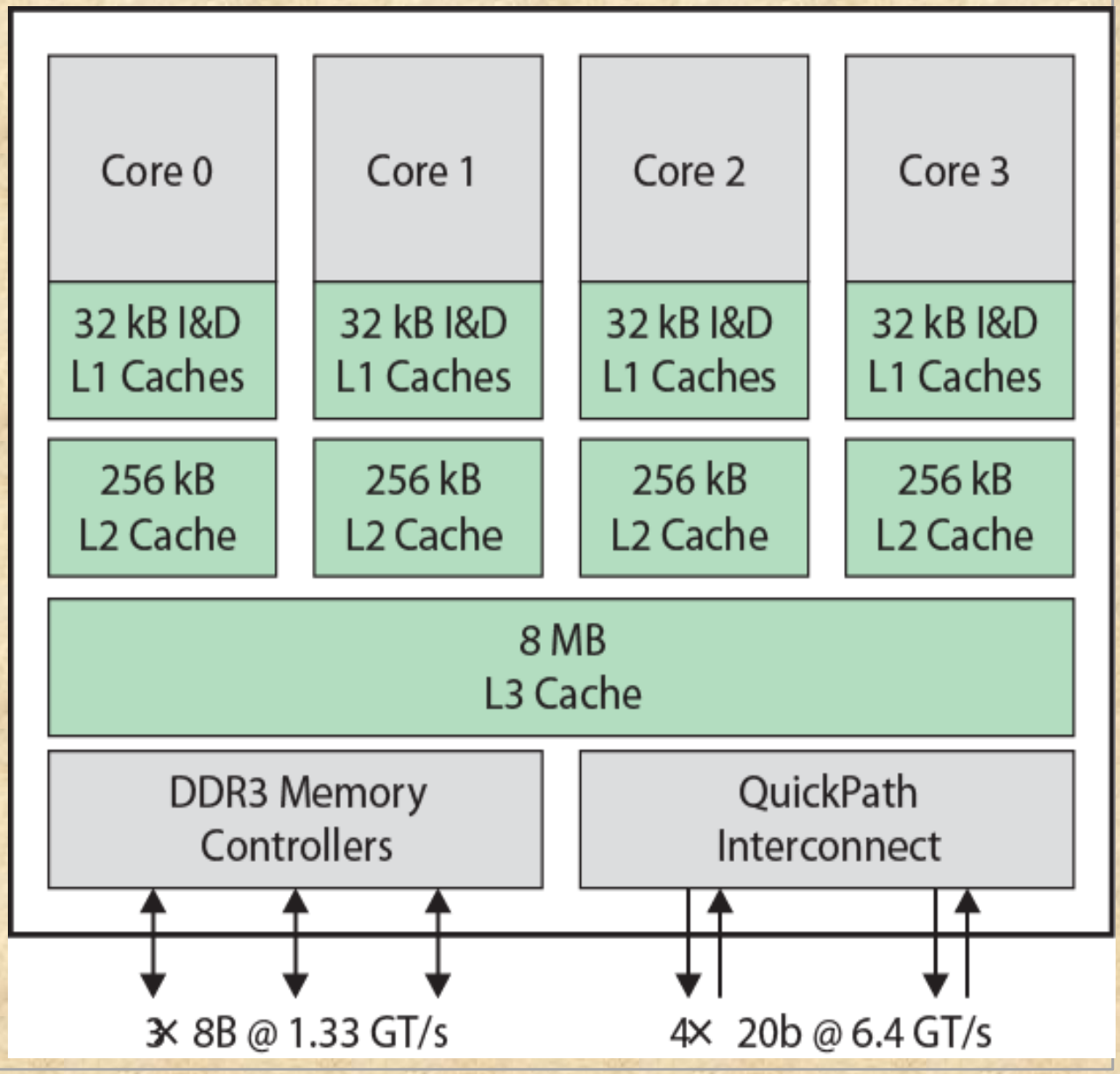


Figure 1.20 Intel Corei7 Block Diagram

Summary

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