Operating Systems

Chapter 1 Computer System Overview

Agenda

1.0 Intro

1.1 Basic Elements

1.2 Processor Registers

1.3 Instruction Execution

1.4 Interrupts

1.5 The Memory Hierarchy

1.6 Cache Memory

1.7 I/O Communication Techniques

Agenda

1.0 Intro

1.1 Basic Elements

1.2 Processor Registers

1.3 Instruction Execution

1.4 Interrupts

1.5 The Memory Hierarchy

1.6 Cache Memory

1.7 I/O Communication Techniques

1.0 Intro

- Operating System 概述
  - 承上：Provides a set of services to system users
  - 启下：Exploits the hardware resources of
    - One or more processors
    - Main memory
    - Manages I/O devices
      - secondary memory
      - Network Card
      - …

Agenda

1.0 Intro

1.1 Basic Elements

1.2 Processor Registers

1.3 Instruction Execution

1.4 Interrupts

1.5 The Memory Hierarchy

1.6 Cache Memory

1.7 I/O Communication Techniques

1.1 Basic Elements (1/2)

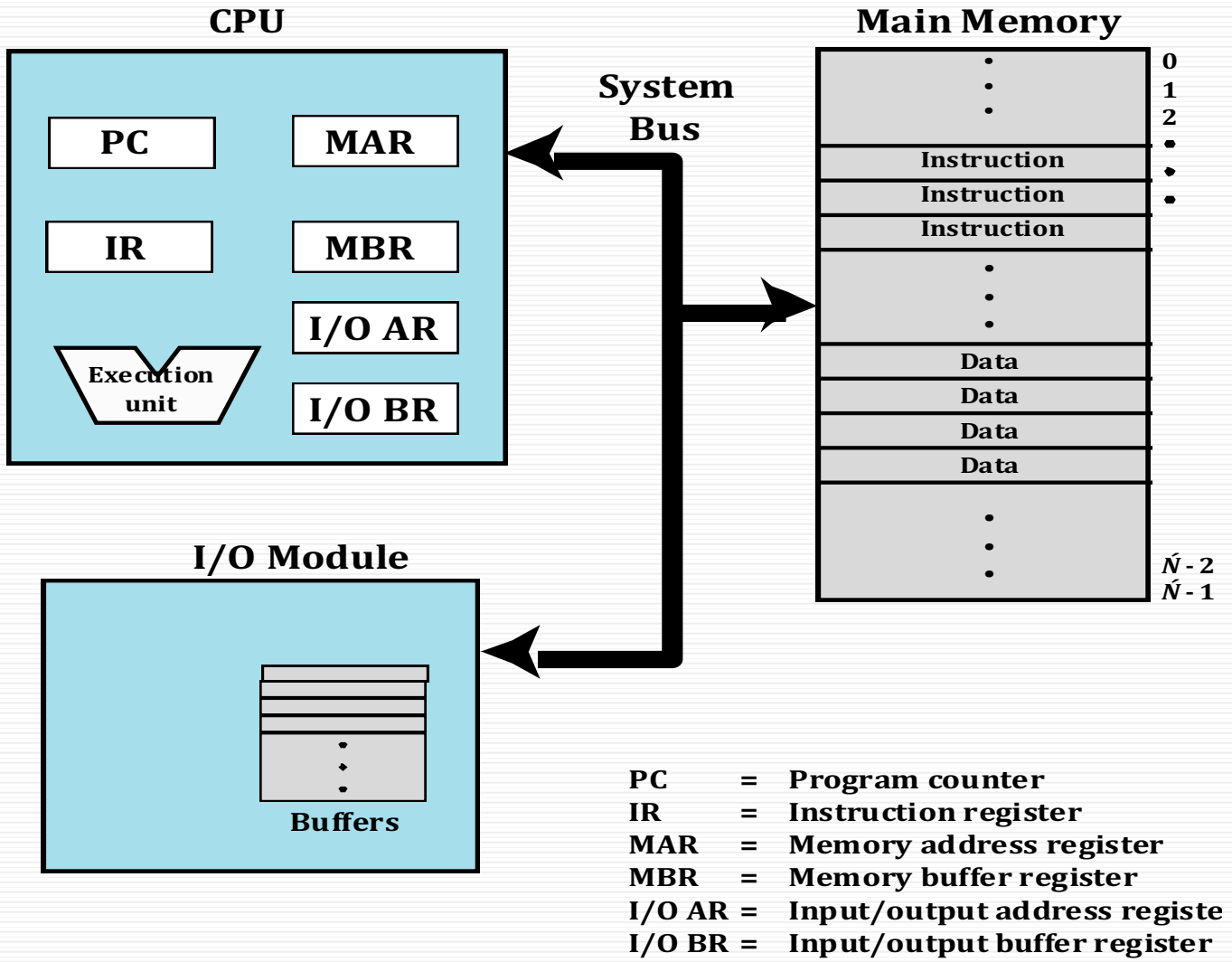


Figure 1.1 Computer Components: Top-Level View

1.1 Basic Elements (2/2)

- Processor(处理器)
- Main Memory(内存)
  - Volatile （易失性）
  - referred to as real memory(实存) or primary memory（主存）
- I/O modules(输入/输出模块)
  - secondary memory devices （disk）
  - communications equipment
  - terminals
- System bus(系统总线)
  - communication among processors, memory, and I/O modules

Agenda

1.0 Intro

1.1 Basic Elements

1.2 Processor Registers

1.3 Instruction Execution

1.4 Interrupts

1.5 The Memory Hierarchy

1.6 Cache Memory

1.7 I/O Communication Techniques

1.2 Processor Registers

- Introduction
  - What is Registers
    - Memory inside CPU
  - Why Registers
    - Enable CPU to minimize main-memory references
  - Can be classified into:
    - User-visible registers(用户可见寄存器)
    - Control and status registers(控制和状态寄存器)

User-Visible Registers

- How to use： May be referenced（访问/存取） by machine/assemble language
- Who will use： Available to all programs
  - application programs
  - system programs

Control and Status Registers (1/2)

- Function： are used to control the operation of the processor
- Most are not visible to the user.
- Some may be accessibly by machine instruction in control or system mode

1.2.2 Control and Status Registers (2/2)

- 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 [more detail next]
  - Other state-related bits, such as:
    - Interrupt enable/disable
    - Supervisor/user mode
    - … …

Agenda

1.0 Intro

1.1 Basic Elements

1.2 Processor Registers

1.3 Instruction Execution

1.4 Interrupts

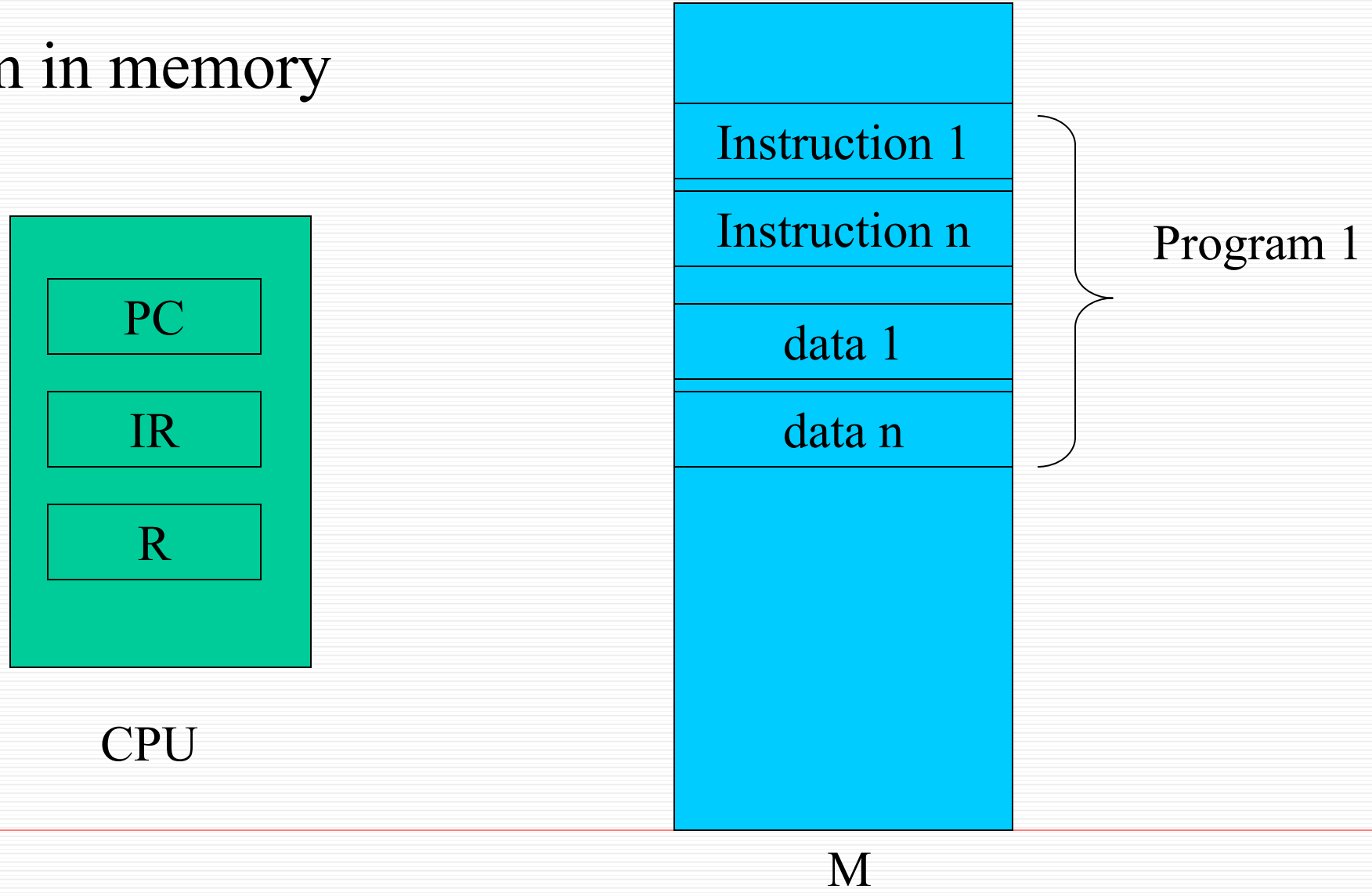
1.5 The Memory Hierarchy

1.6 Cache Memory

1.7 I/O Communication Techniques

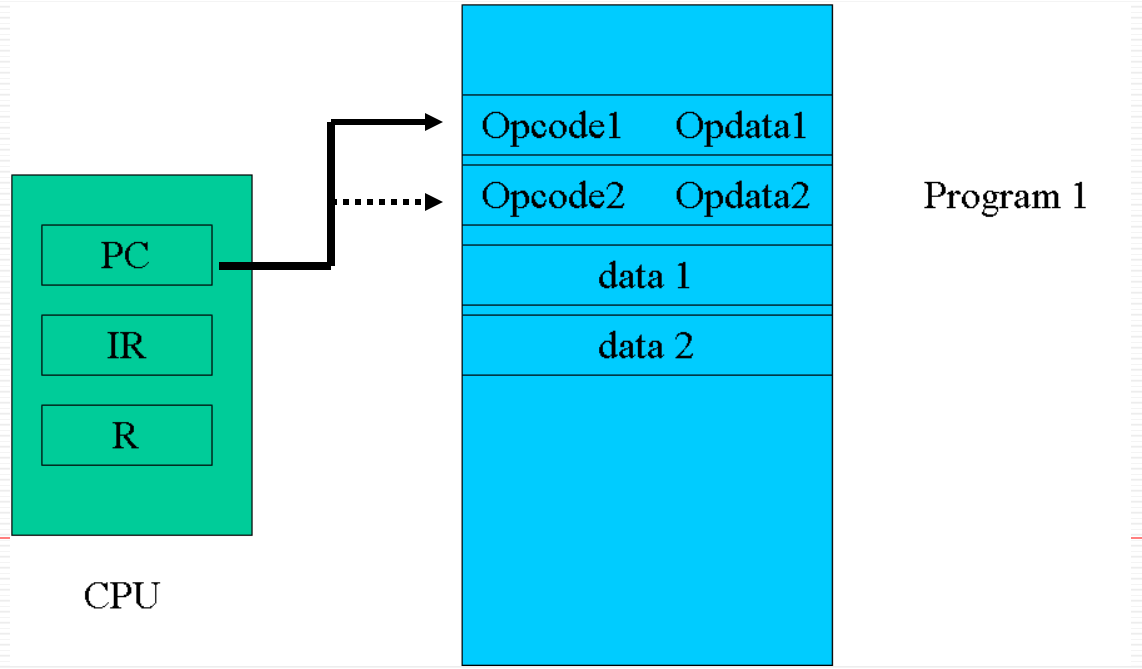
1.3 Instruction Execution (1/7)

- Program in memory



1.3 Instruction Execution (2/7)

- Key point:
  - Program counter (PC) of CPU holds address of the instruction to be fetched next
  - Fetched instruction is placed in the instruction register (IR)
  - Program counter (PC) of CPU is incremented after each fetch



1.3 Instruction Execution (3/7)

- Instruction Cycle(指令周期)
  - The processing required for a single instruction execution

1.3 Instruction Execution (4/7)

- Two stages of each Instruction Execution
  - Processor reads/loads/Fetches instructions from memory
  - Processor Executes each instruction

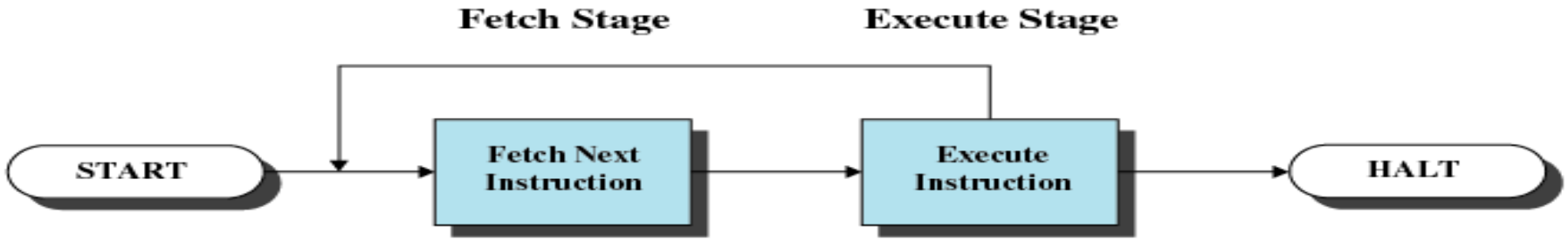
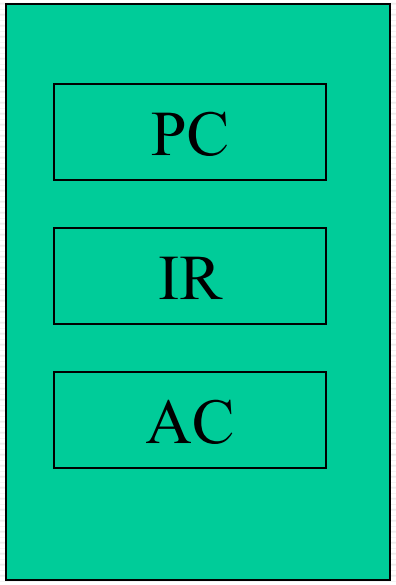


Figure 1.2 Basic Instruction Cycle

1.3 Instruction Execution (5/7)

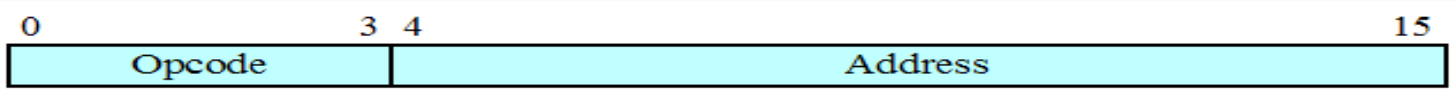
- An Example (cont.): the CPU



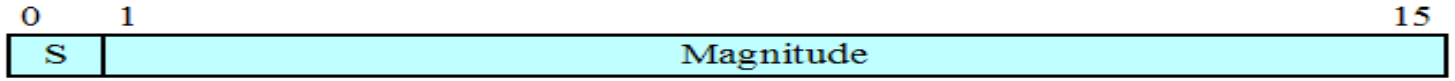
CPU

1.3 Instruction Execution (6/7)

- An Example: the Instruction



(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

一 台 理 想 机 器 的 特 征

1.3 Instruction Execution (11/12)

- An Example (cont.): the process of a program execution

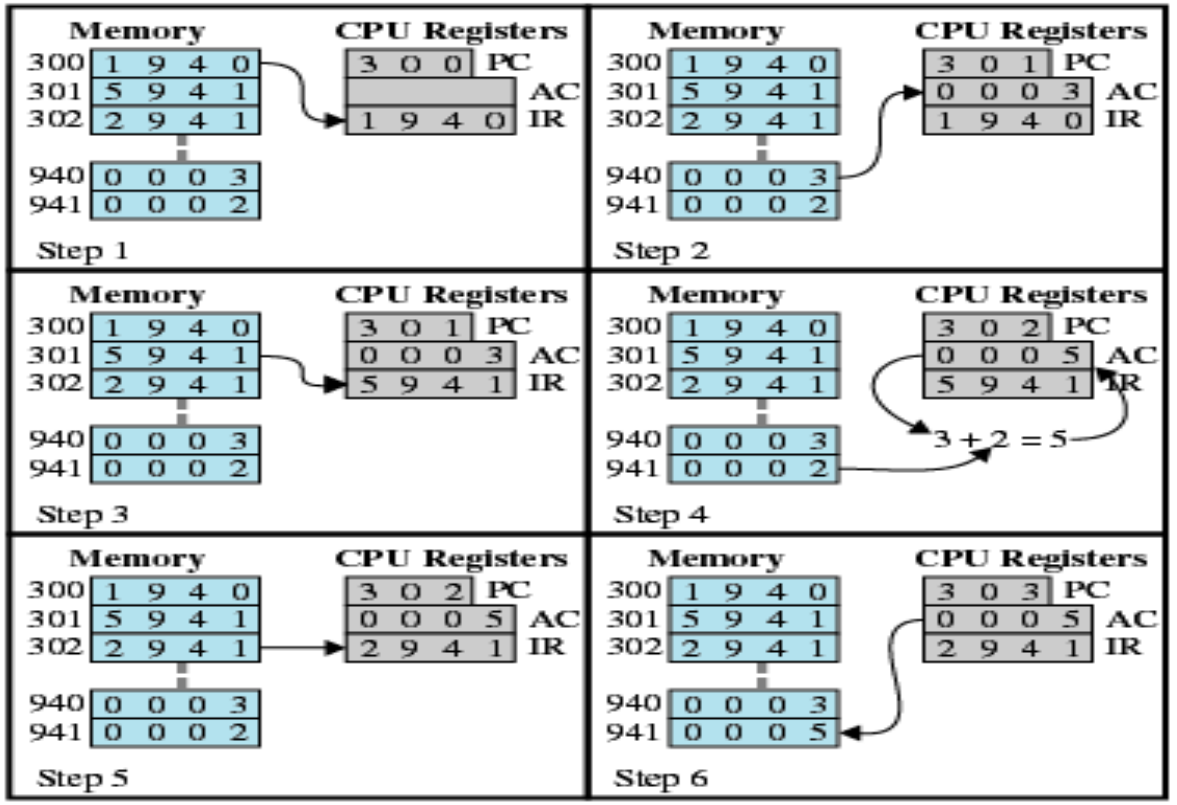


Figure 1.4 Example of Program Execution
(contents of memory and registers in hexadecimal)

指令1

指令2

指令3

Agenda

1.0 Intro

1.1 Basic Elements

1.2 Processor Registers

1.3 Instruction Execution

1.4 Interrupts

1.5 The Memory Hierarchy

1.6 Cache Memory

1.7 I/O Communication Techniques

1.4 Interrupts

1.4.0 Interrupt Introduction (1/2)

- Why Interrupt in computer system?
  - Most I/O devices are slower than the processor， so interrupt can improve CPU's utilization（提高CPU利用率）
    - Processor must pause to wait for device
    - E.g.: printer disk
  - Count （计数时钟中断）
  - Avoid some program to monopolize CPU (避免CPU被独占)

1.4.0 Interrupt Introduction (2/2)

- Introduction Interrupt Definition
  - A mechanism by which other modules (I/O, clock) may interrupt the normal sequencing of the processor.
    - An I/O device can stop what the CPU is operating to provide some necessary service.
  - Interrupt and Restore/Resume
    - Both software and hardware are used to support it

1.4 Interrupts

1.4.0 Interrupt Introduction

1.4.1 Interrupts and the Instruction Cycle

1.4.2 Interrupt Processing

1.4.3 Multiple Interrupts

1.4.4 Multiprogramming

1.4.1 Interrupts and the Instruction Cycle(1/2)

- Instruction Cycle With Interrupts

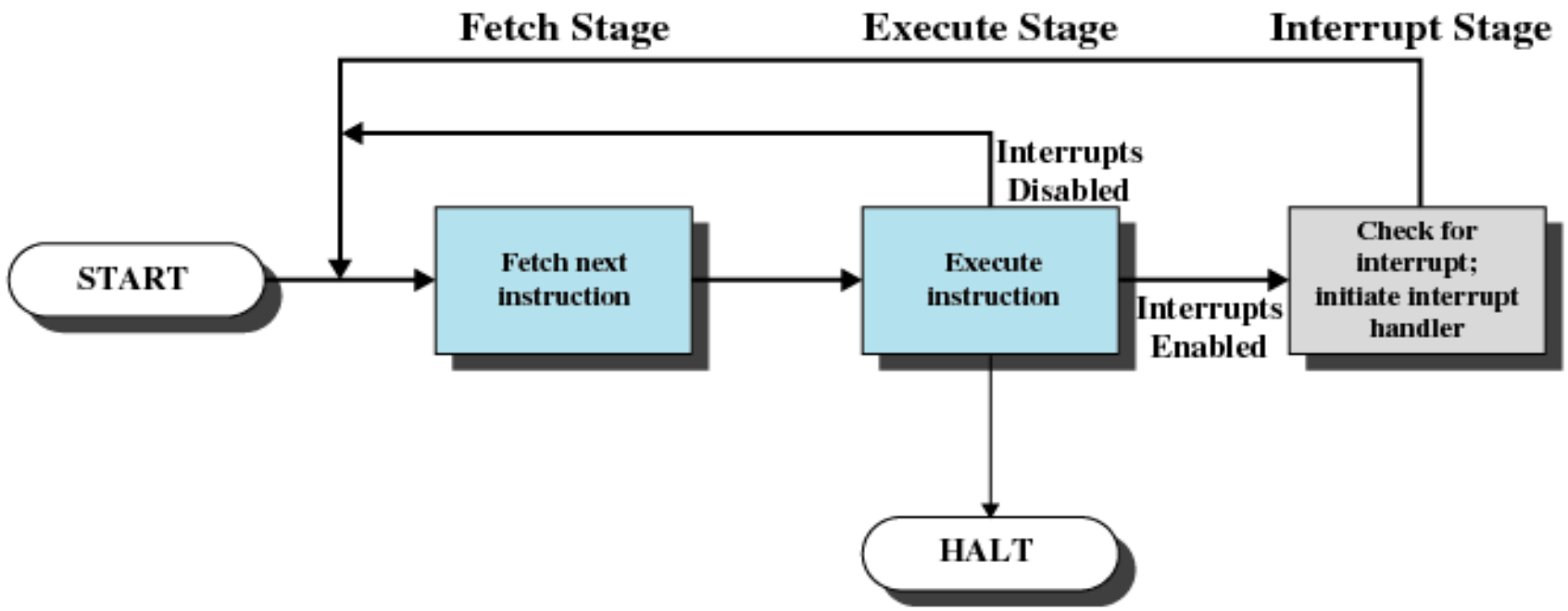
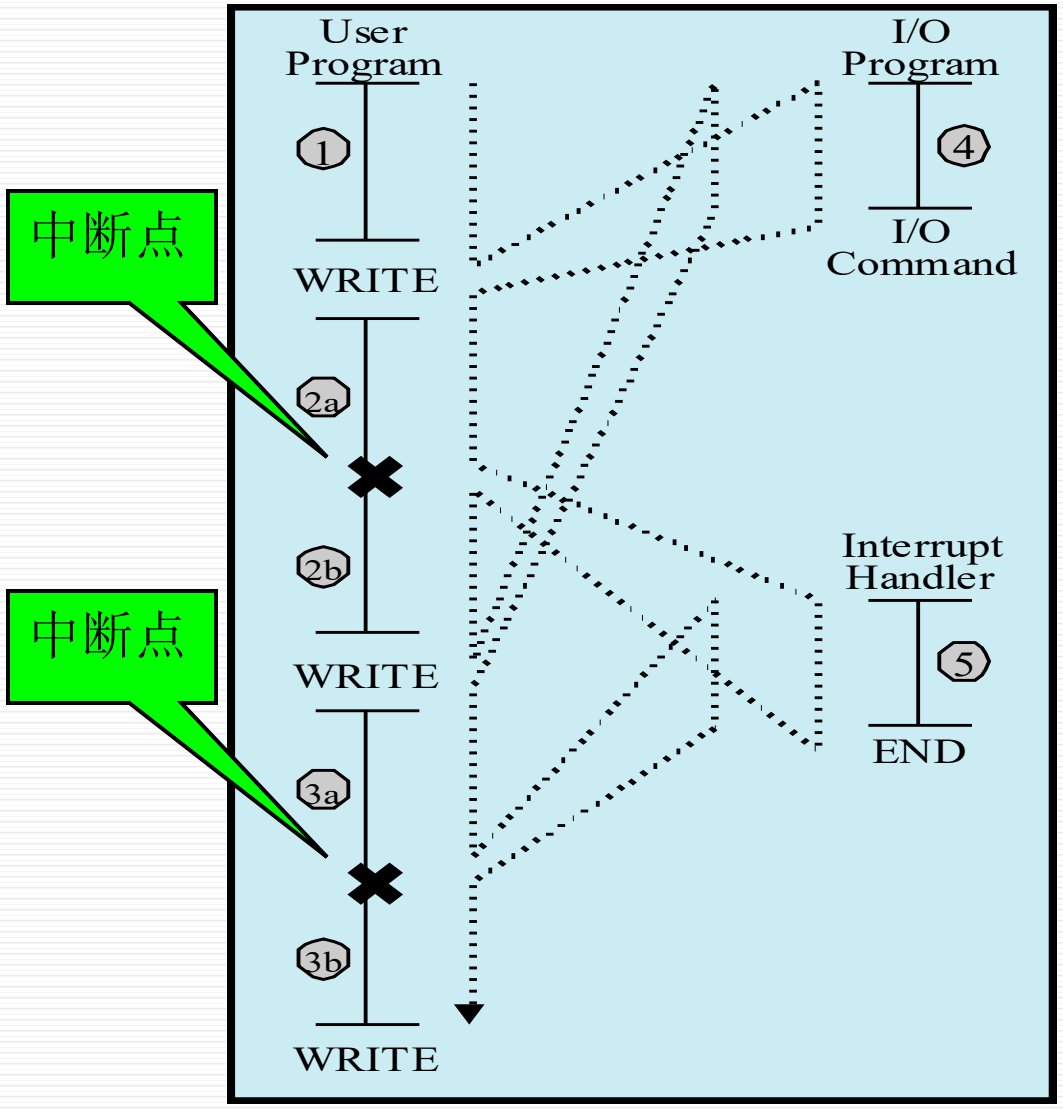
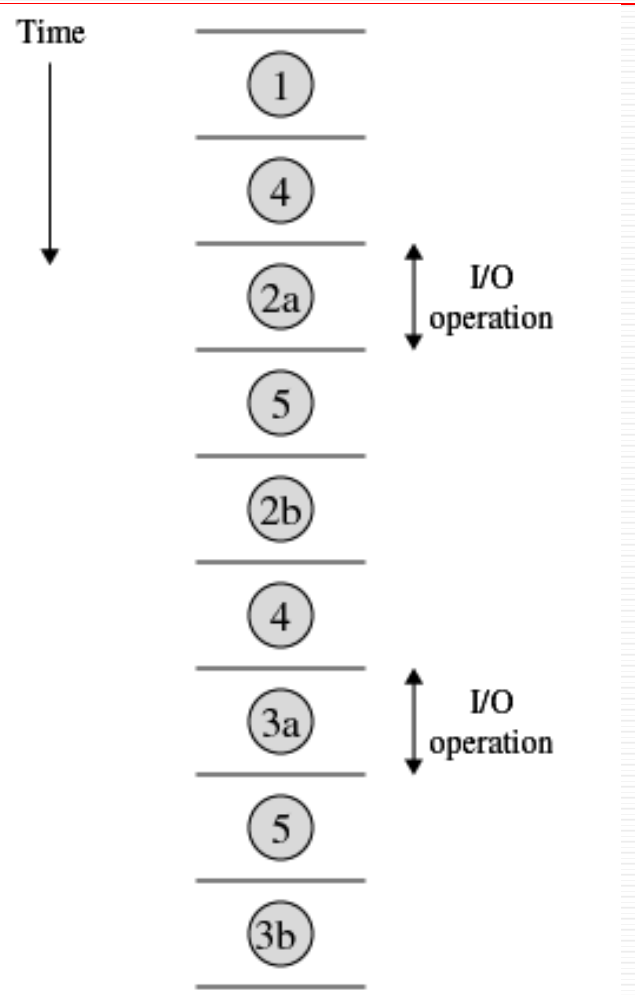


Figure 1.7 Instruction Cycle with Interrupts

1.4.1 Interrupts and the Instruction Cycle(2/2)



(b) Interrupts; short I/O wait



(b) With interrupts
(circled numbers refer
to numbers in Figure 1.5b)

1.4 Interrupts

1.4.0 Interrupt Introduction

1.4.1 Interrupts and the Instruction Cycle

1.4.2 Interrupt Processing

1.4.3 Multiple Interrupts

1.4.4 Multiprogramming

1.4.2 Interrupt Processing (1/5)

- Suspends the normal sequence of execution

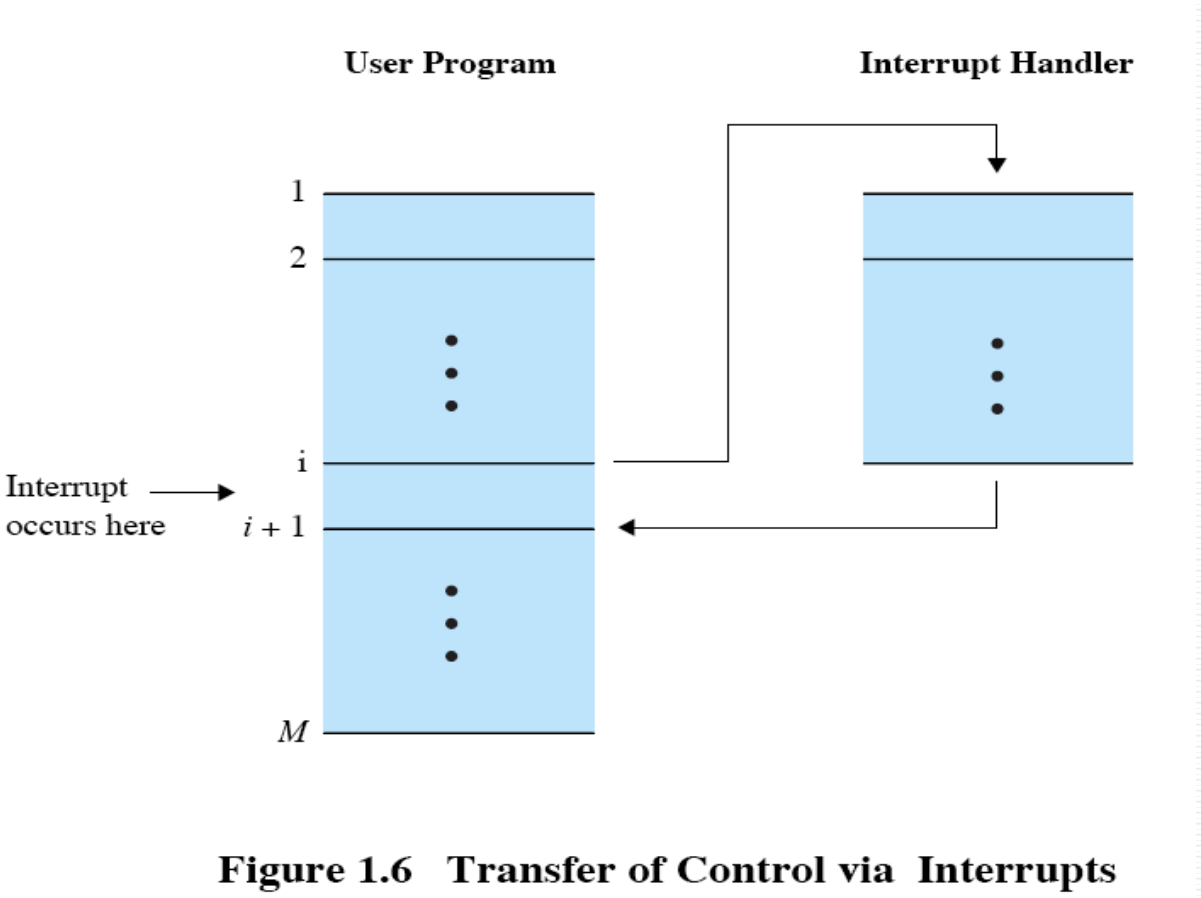


Figure 1.6 Transfer of Control via Interrupts

1.4.2 Interrupt Processing (2/5)

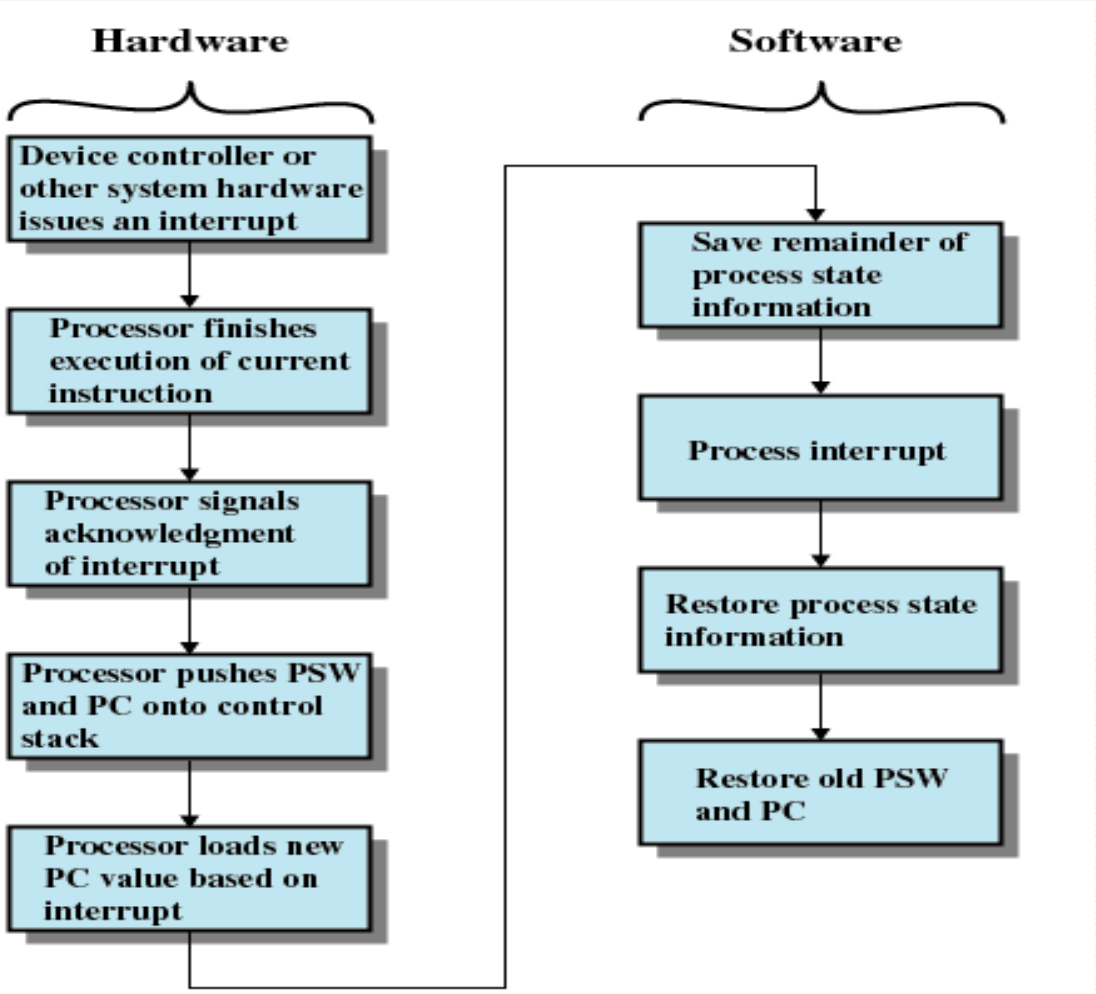


Figure 1.10 Simple Interrupt Processing

1.4.2 Interrupt Processing (3/5)

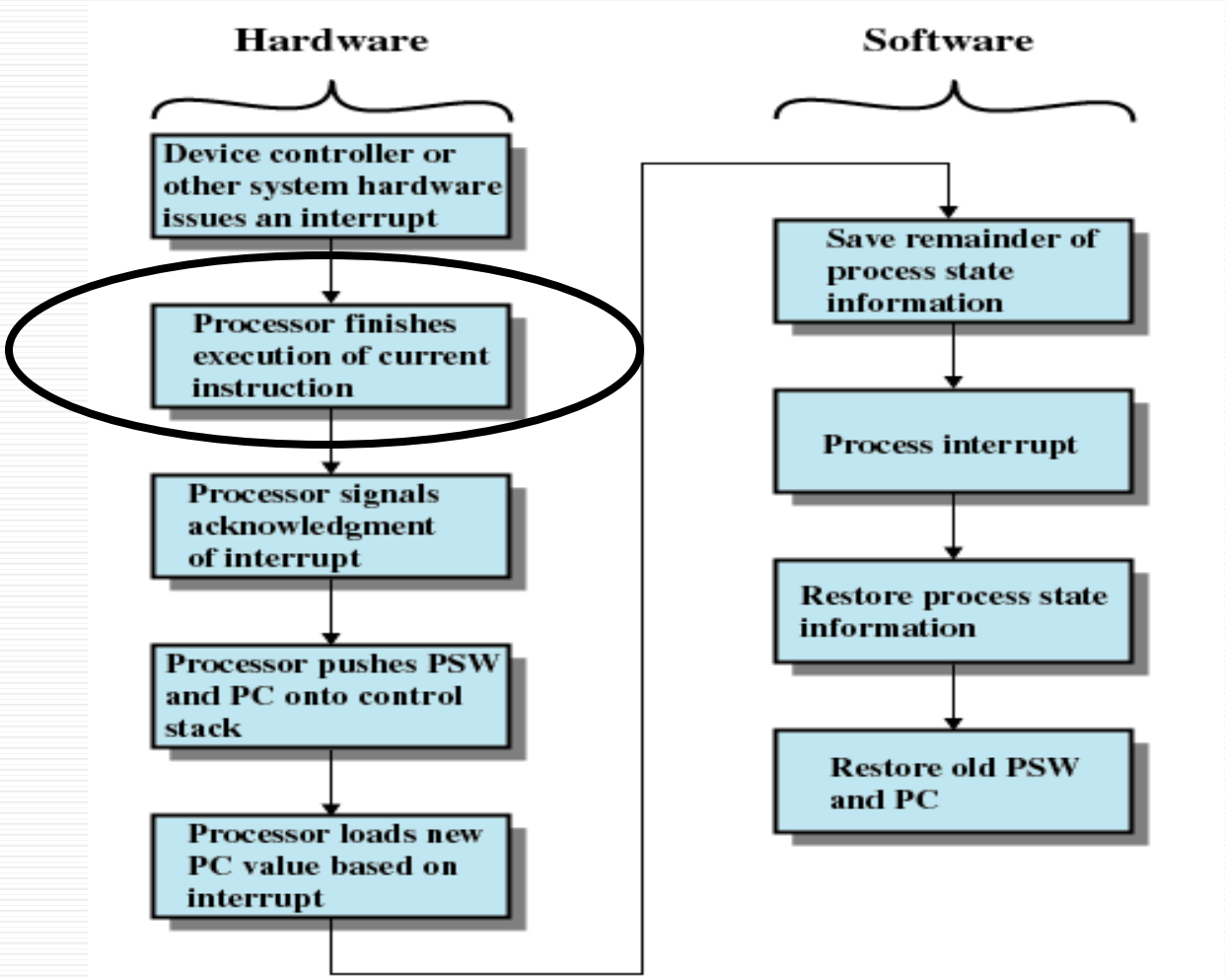


Figure 1.10 Simple Interrupt Processing

1.4.2 Interrupt Processing (4/5)

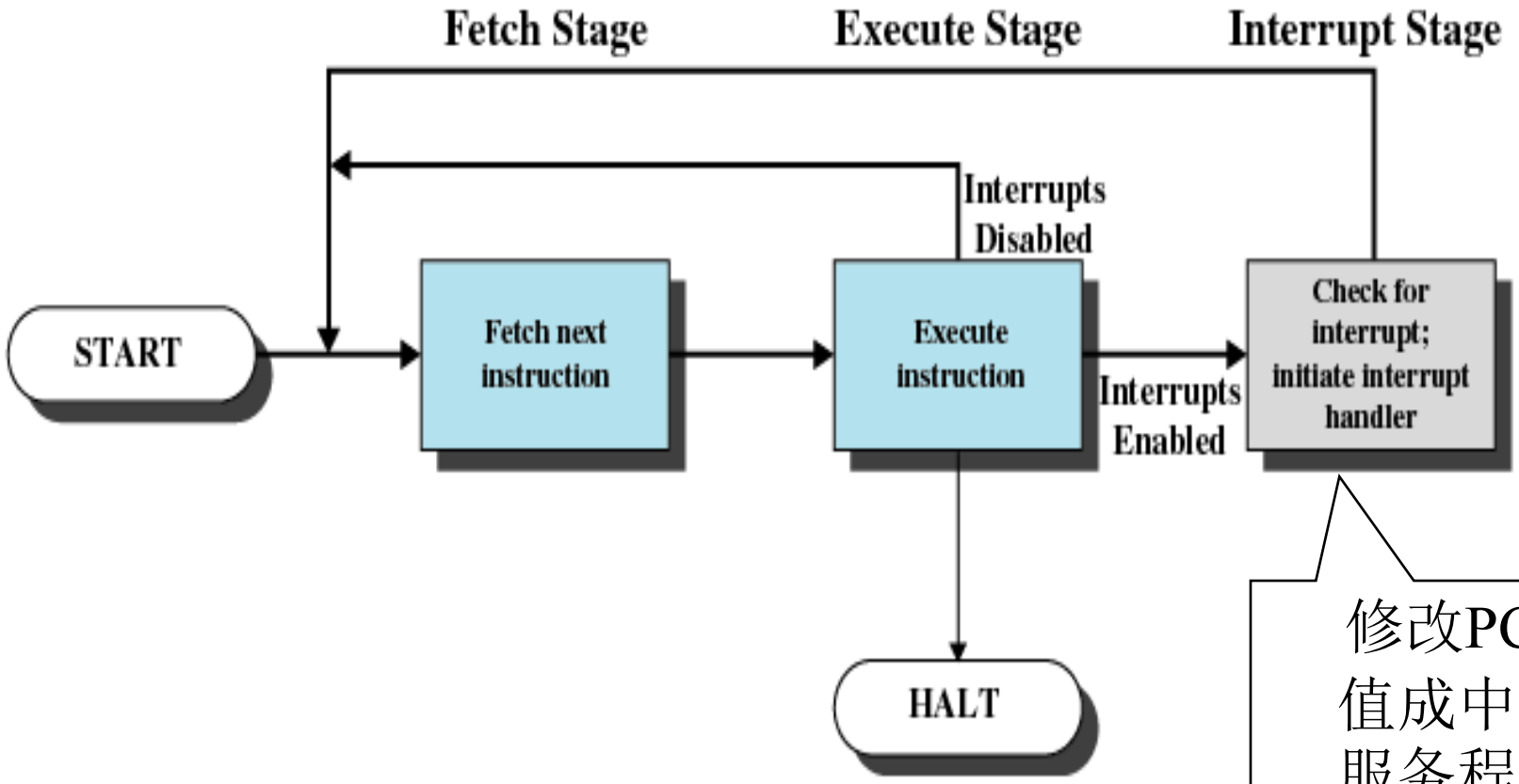


Figure 1.7 Instruction Cycle with Interrupts

1.4.2 Interrupt Processing (5/5)

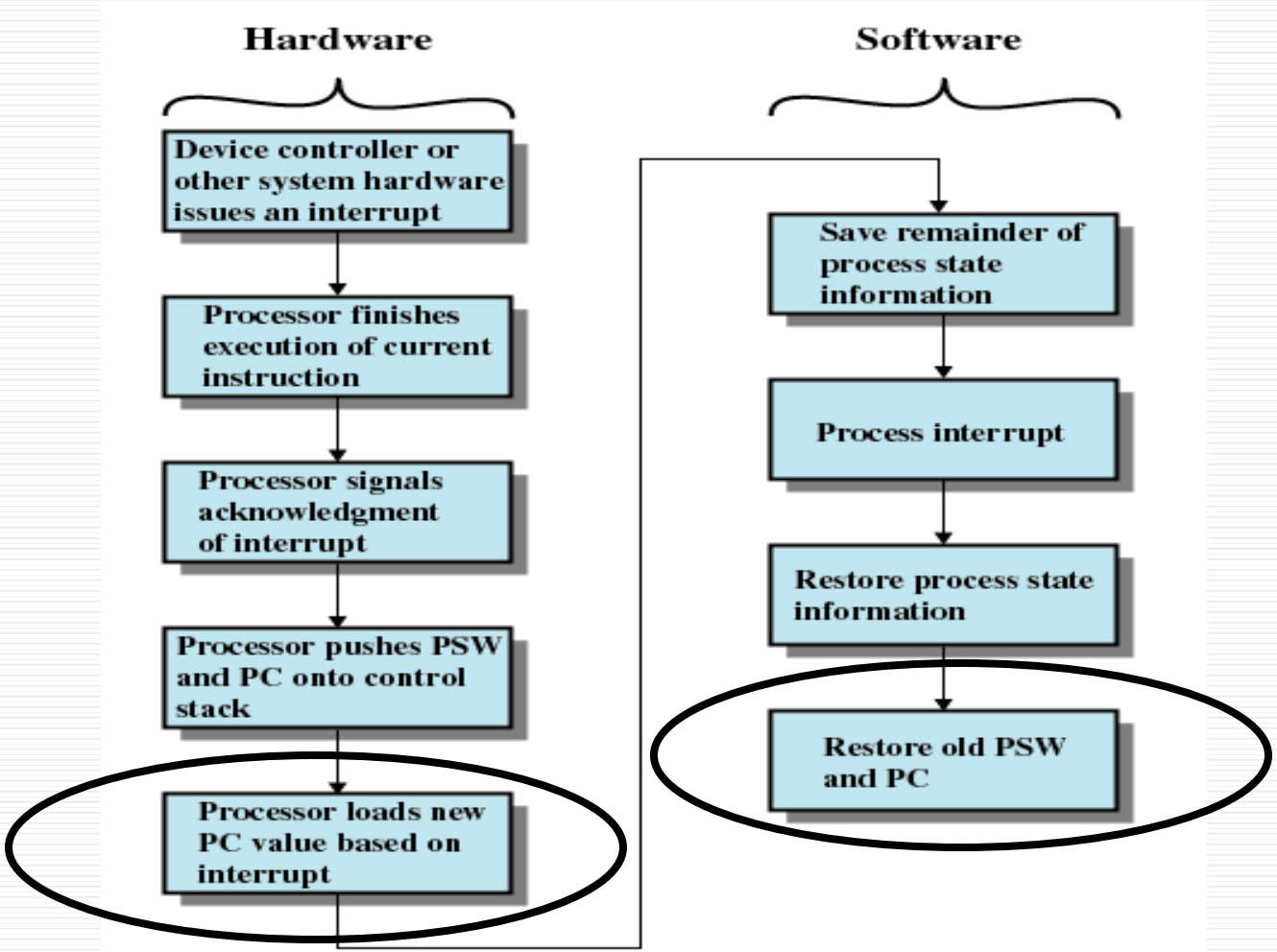


Figure 1.10 Simple Interrupt Processing

1.4 Interrupts

1.4.0 Interrupt Introduction

1.4.1 Interrupts and the Instruction Cycle

1.4.2 Interrupt Processing

1.4.3 Multiple Interrupts

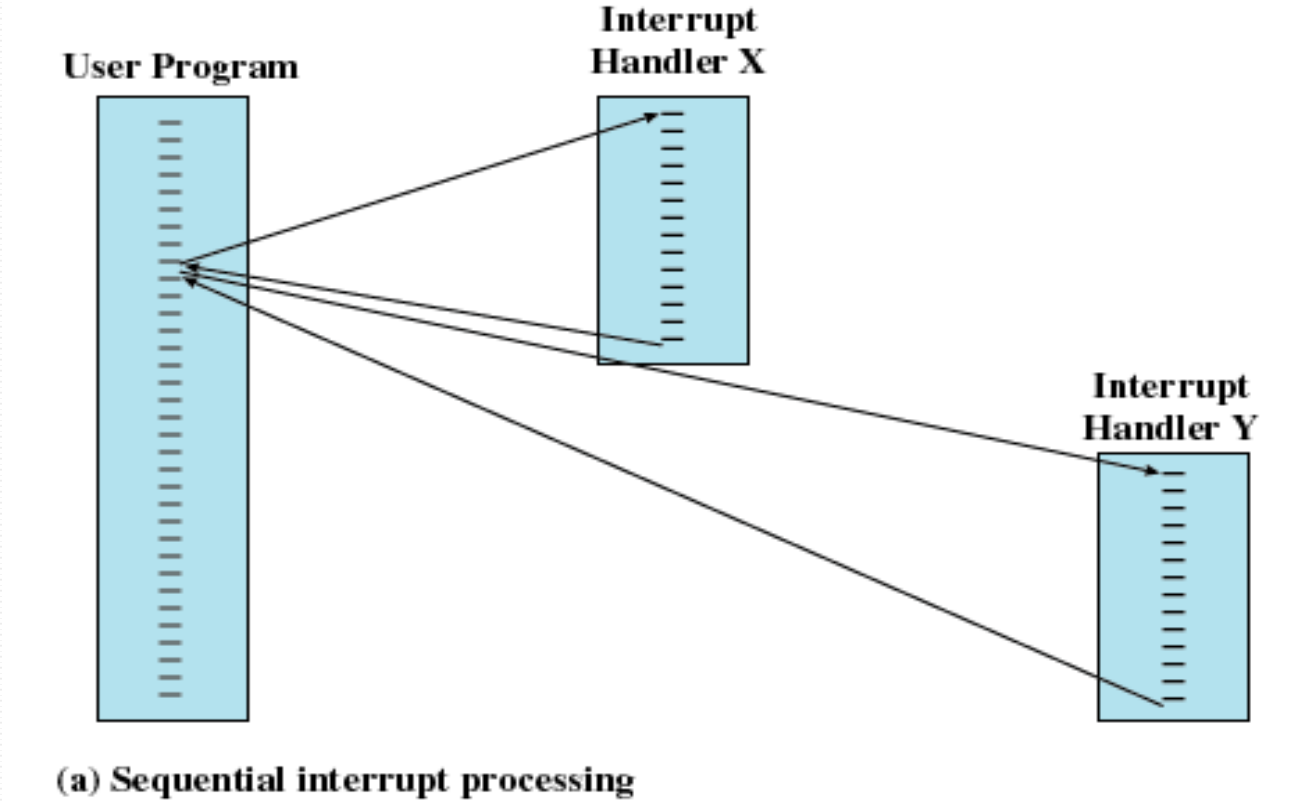
1.4.4 Multiprogramming

1.4.3 Multiple Interrupts (1/4)

- Q: What will happen if an I/O modules rouse a Interrupt, while the CPU is processing instruction of an ISR
  - That is interrupt an ISR

1.4.3 Multiple Interrupts (2/4)

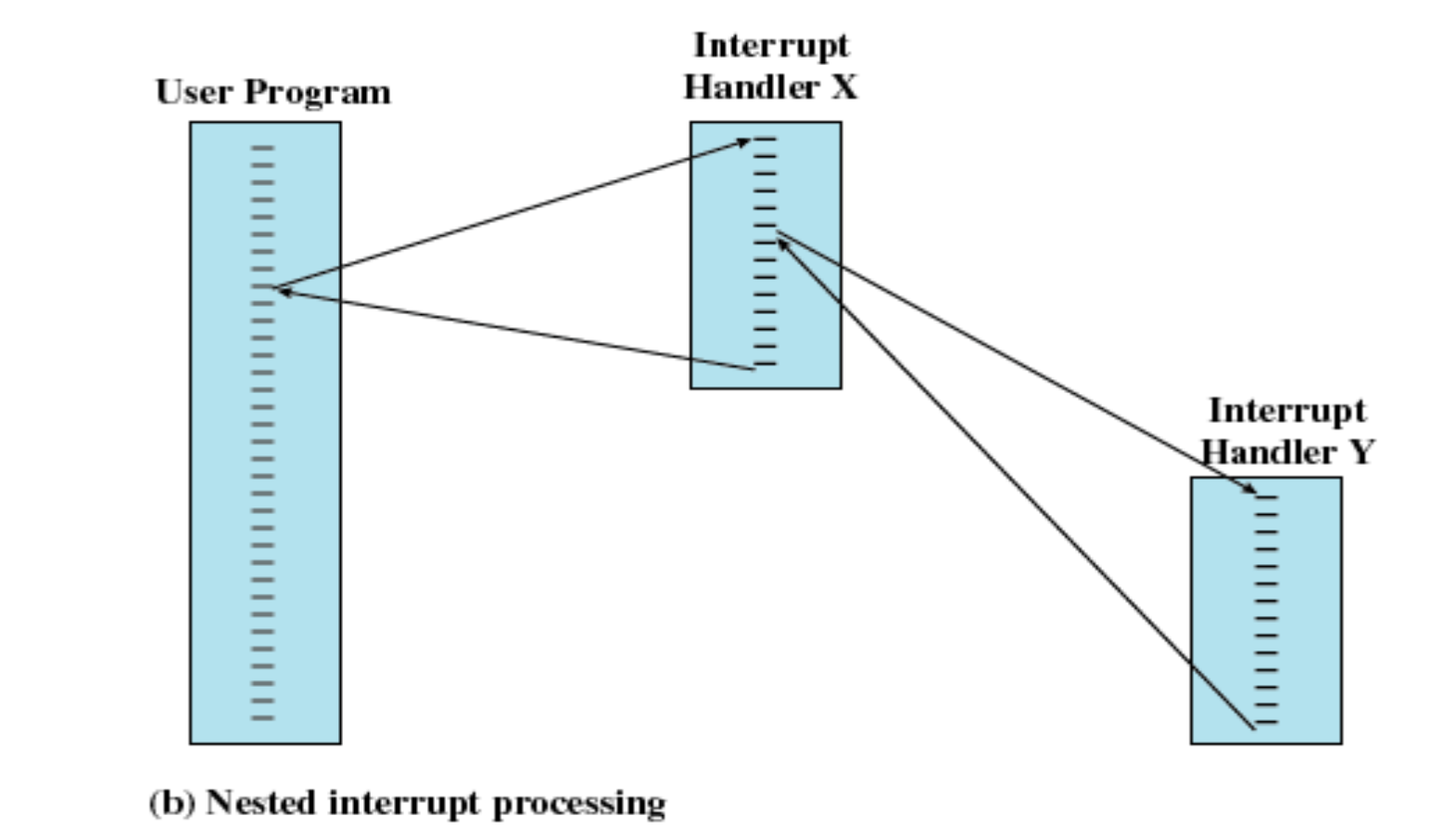
- Method 1： Disable interrupts while an interrupt is being processed



(a) Sequential interrupt processing

1.4.3 Multiple Interrupts (3/4)

- Method 2： Define priorities for interrupts



(b) Nested interrupt processing

1.4.3 Multiple Interrupts (4/4)

- An Example of Method 2

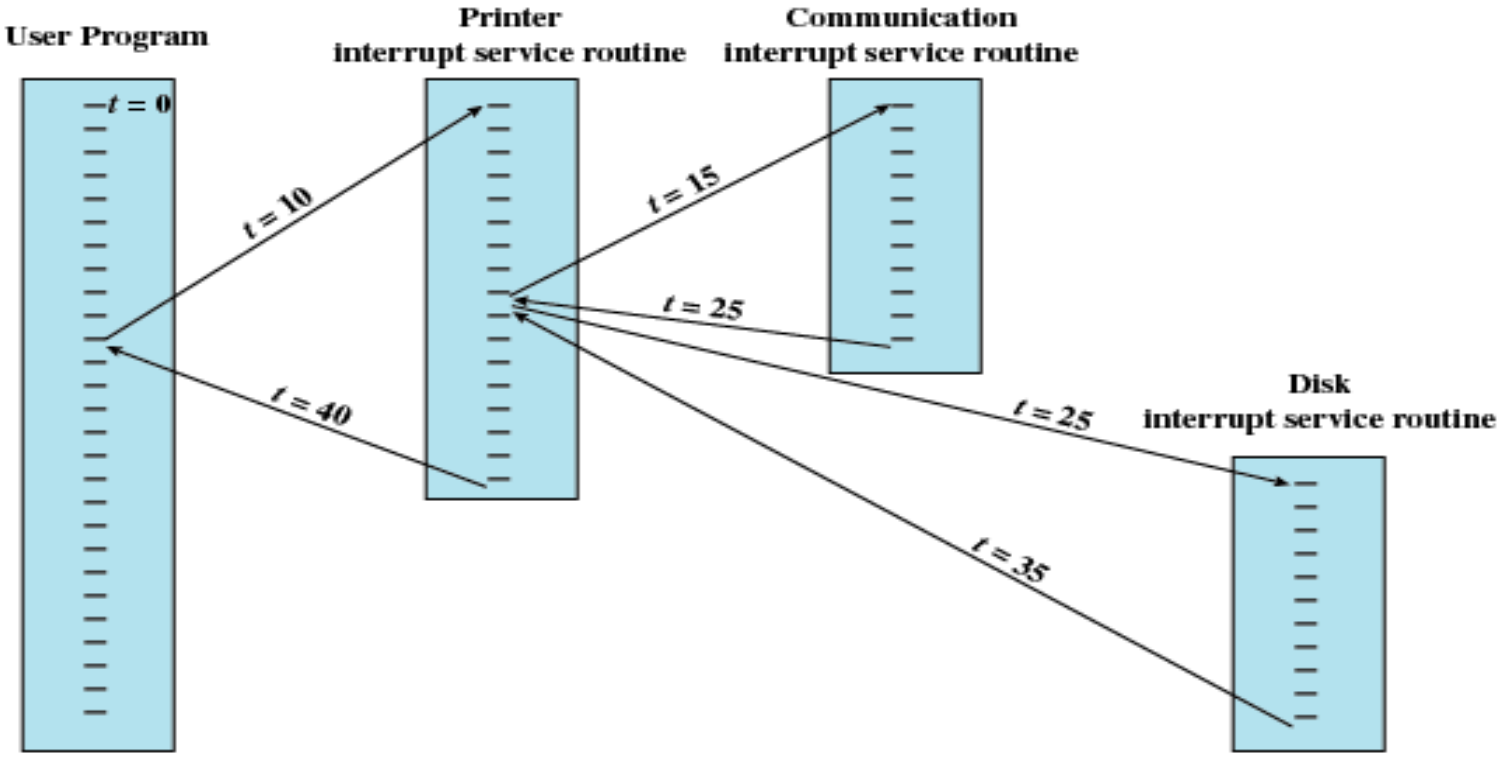


Figure 1.13 Example Time Sequence of Multiple Interrupts

1.4 Interrupts

1.4.0 Interrupt Introduction

1.4.1 Interrupts and the Instruction Cycle

1.4.3 Interrupt Processing

1.4.3 Multiple Interrupts

1.4.4 Multiprogramming

1.4.4 Multiprogramming

- Multiprogramming(多道程序)
  - If the time required to complete an I/O operation is much greater than the user code between I/O calls, then the processor will be idle much of the time
  - A solution to this problem is to allow multiple user programs to be active at the same time

Agenda

1.0 Intro

1.1 Basic Elements

1.2 Processor Registers

1.3 Instruction Execution

1.4 Interrupts

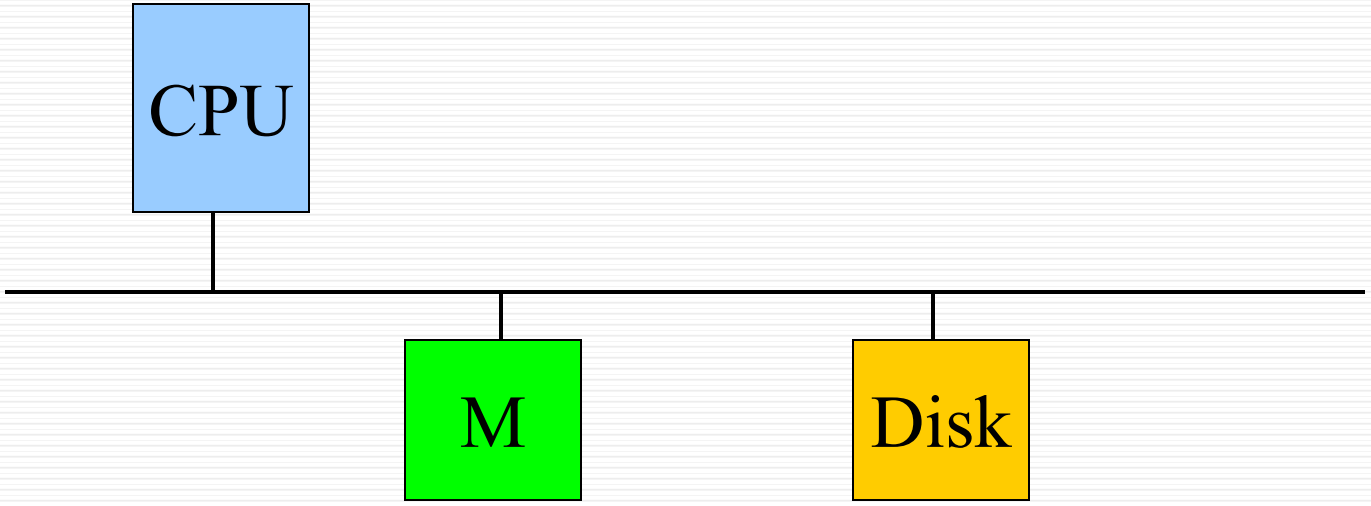
1.5 The Memory Hierarchy

1.6 Cache Memory

1.7 I/O Communication Techniques

1.5 The Memory Hierarchy (1/5)

- The design constraints of a computer's memory
  - How much : Capacity : larger
  - How fast : Speed : faster/ as fast as CPU
  - How expensive : Price : cheaper



1.5 The Memory Hierarchy (2/5)

- (Speed , Price, Capacity)的不可兼得性/矛盾:
  - Faster access time, greater cost per bit
  - Greater capacity, smaller cost per bit
  - Greater capacity, slower access speed

- What can we do?
  - 中庸？
    - To use them all in the right way: Memory Hierarchy

1.5 The Memory Hierarchy (3/5)

- Memory Hierarchy： Three Levels
  - Level 1：板上存储器
  - Level 2：板外存储器
  - Level 3：离线存储器

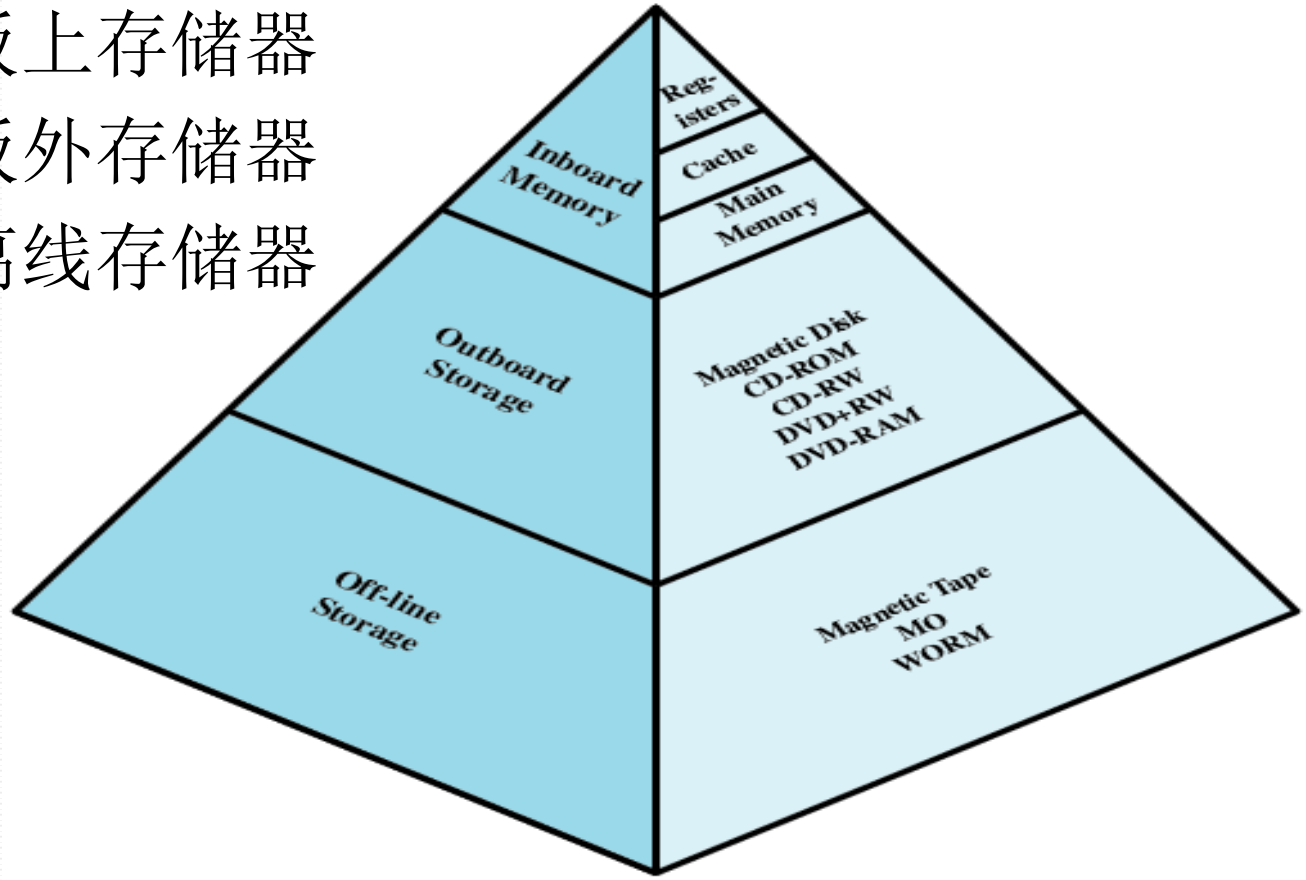


Figure 1.14 The Memory Hierarchy

1.5 The Memory Hierarchy (4/5)

- Going Down the Hierarchy
  - Increasing capacity
  - Increasing access time (that is slow speed)
  - Decreasing cost per bit
  - Decreasing frequency of access of the memory by the processor

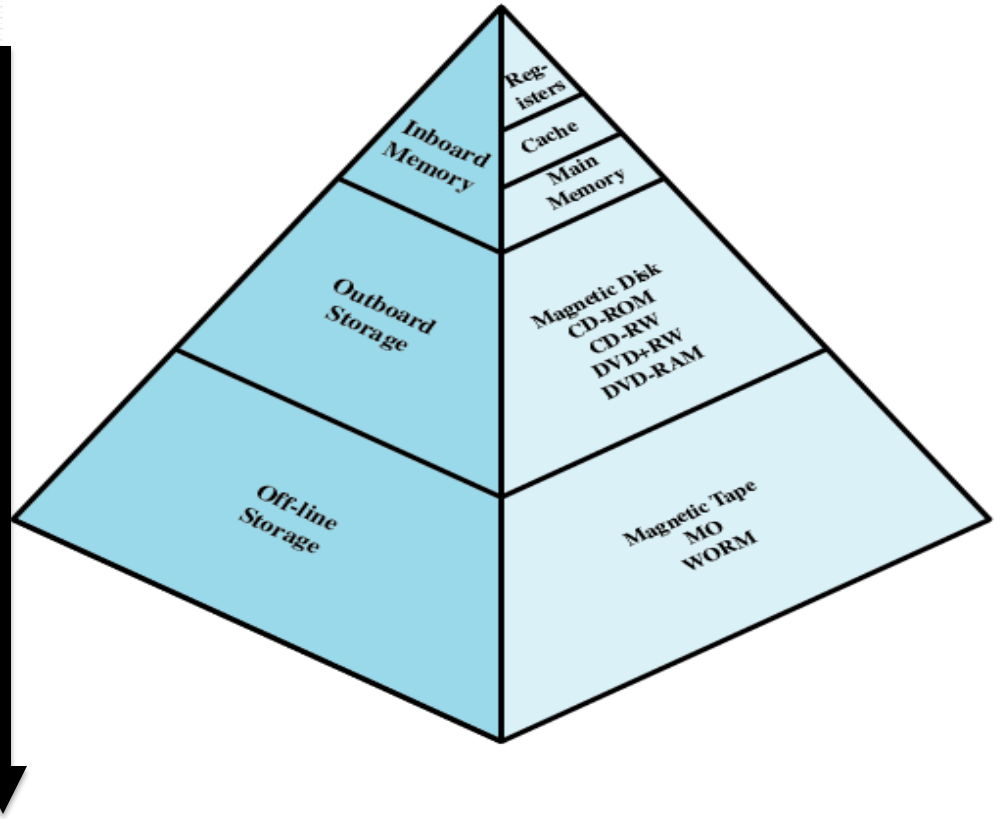


Figure 1.14 The Memory Hierarchy

1.5 The Memory Hierarchy (5/5)

- Obtaining larger and faster memory
  - Get larger capacity
    - Smaller, more expensive, faster memories are supplemented (后备) by larger, cheaper, slower memories
  - Get faster speed
    - larger, cheaper, slower memories are cached（缓存） by Smaller, more expensive, faster memories
- Locality of reference(访问局部性/局部性原理/principle of locality)
  - Spatial locality (空间局部性)
  - Temporal locality (时间局部性)
  - Why it exists

Agenda

1.0 Intro

1.1 Basic Elements

1.2 Processor Registers

1.3 Instruction Execution

1.4 Interrupts

1.5 The Memory Hierarchy

1.6 Cache Memory

1.7 I/O Communication Techniques

1.6 Cache Memory (1/4)

- Exploit the principle of locality
  - Add something cache between fast and slow memory

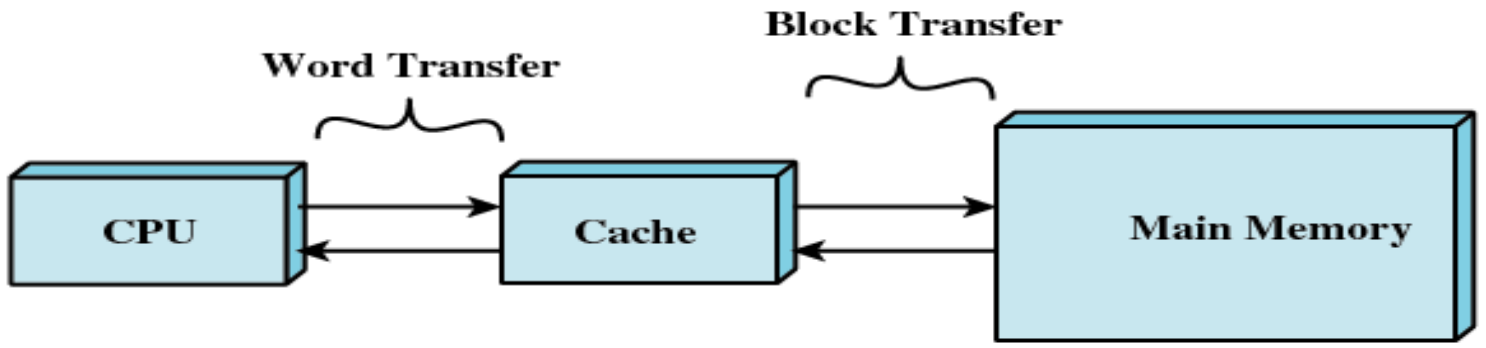
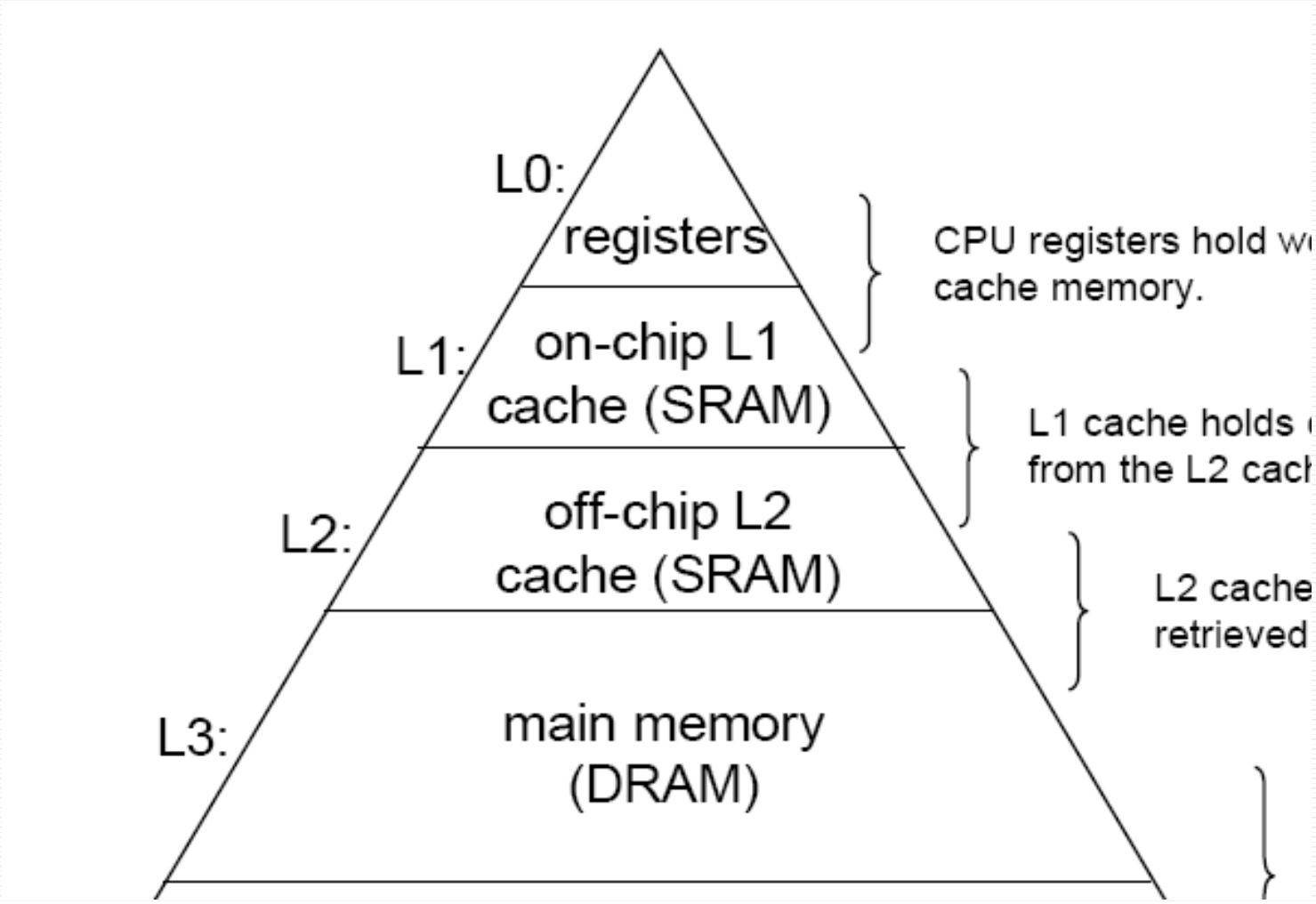


Figure 1.16 Cache and Main Memory

1.6 Cache Memory (2/4)



1.6 Cache Memory (3/4)

- Cache Principles
  - Contains a copy of a portion of main memory
  - Processor first checks cache
    - (Hit) If found, just use it. And do not need access to the memory
    - (Miss) If not found in cache, the block of memory containing the needed information is moved to the cache and delivered to the processor

1.6 Cache Memory (4/4)

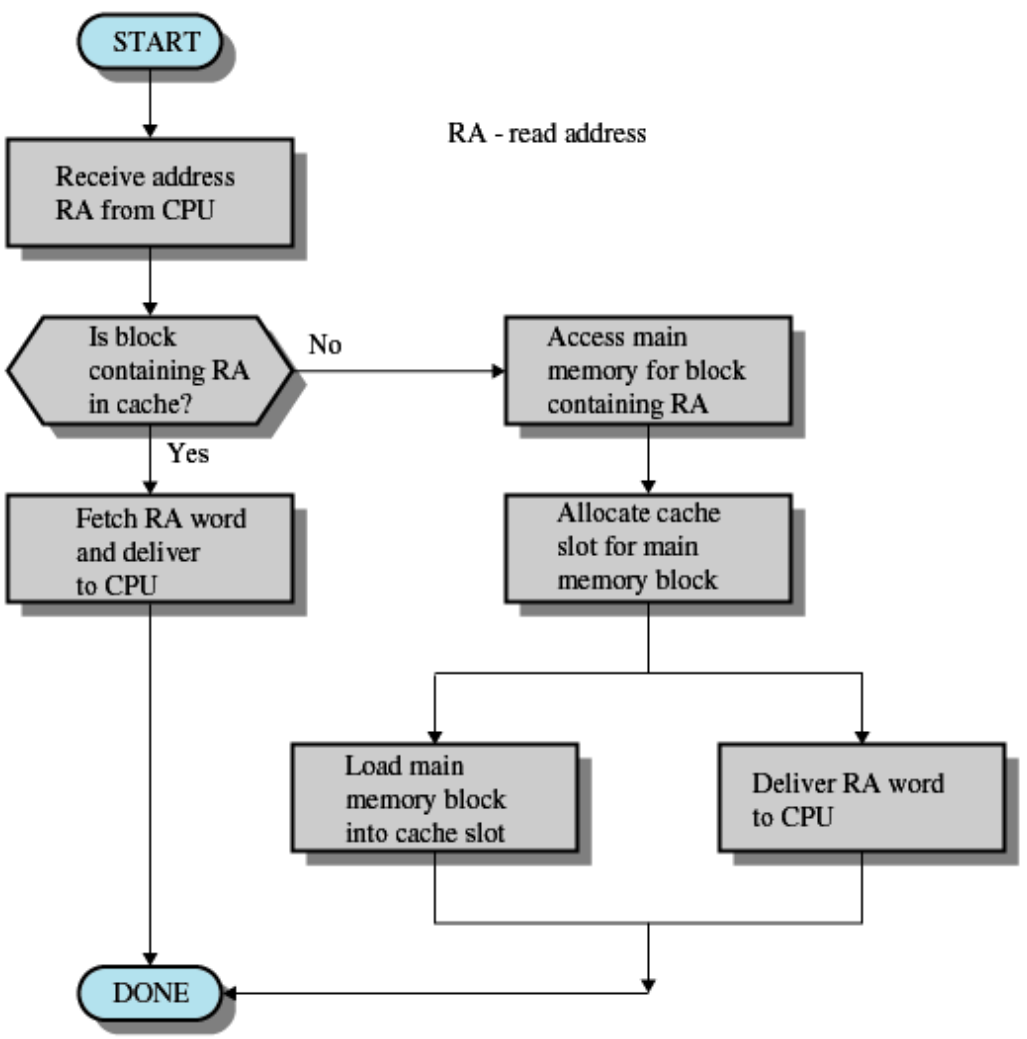


Figure 1.18 Cache Read Operation

Agenda

1.0 Intro

1.1 Basic Elements

1.2 Processor Registers

1.3 Instruction Execution

1.4 Interrupts

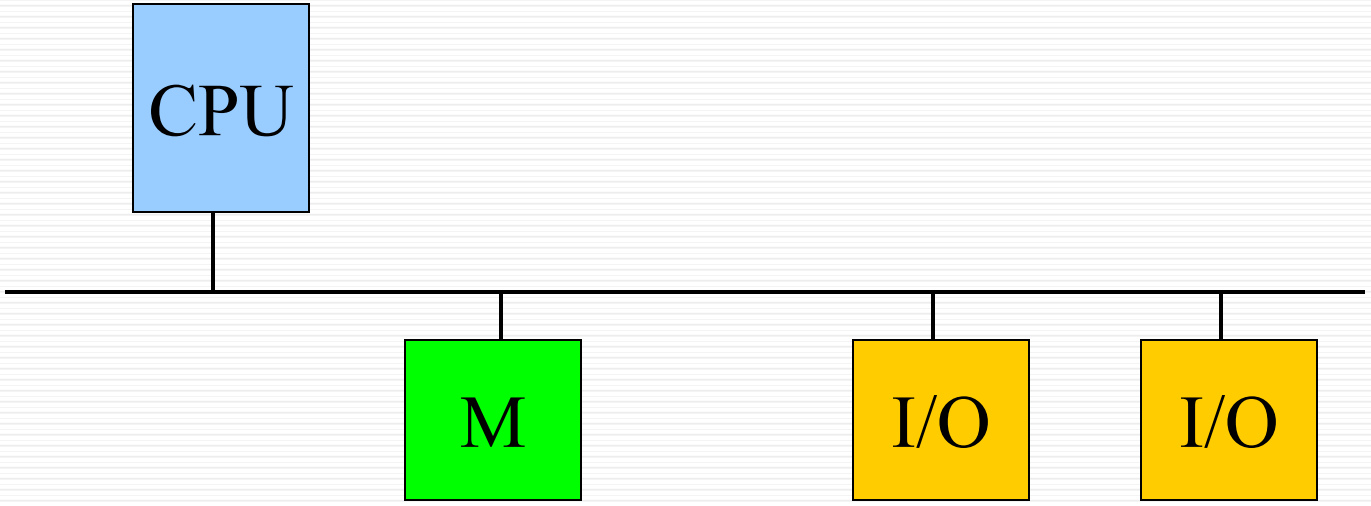
1.5 The Memory Hierarchy

1.6 Cache Memory

1.7 I/O Communication Techniques

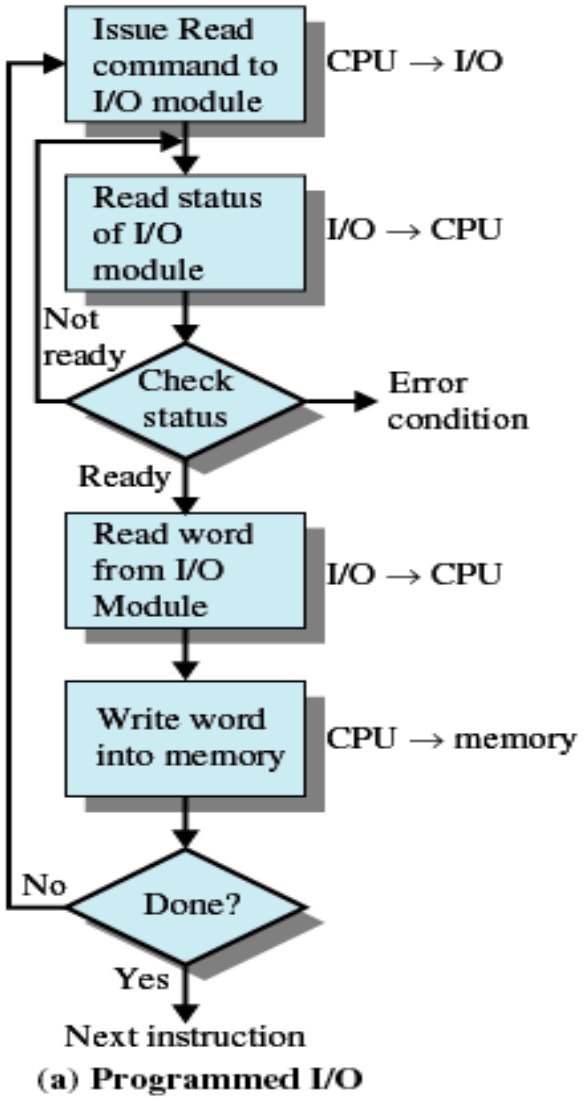
1.7 I/O Communication Techniques (1/4)

- Three methods
  - Programmed I/O
  - Interrupt-Driven I/O
  - Direct Memory Access / DMA



1.7 I/O Communication Techniques (2/4)

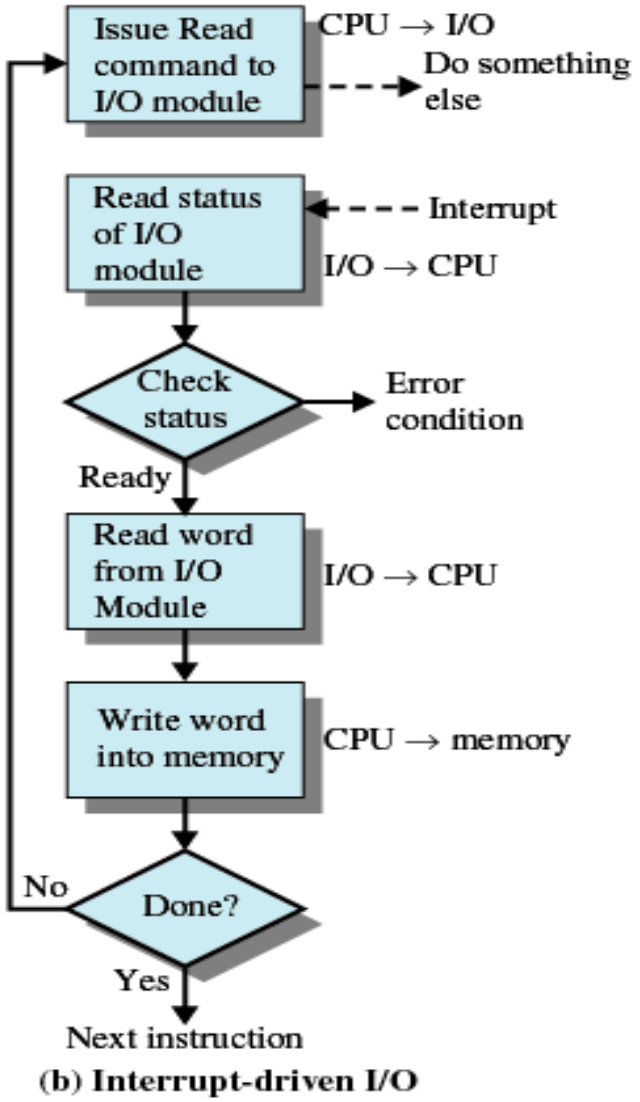
- Method1: Programmed I/O (可编程I/O)
  - I/O module performs the action, not the processor
  - I/O module sets appropriate bits in the I/O status register
  - Processor checks status until operation is complete
  - Disadvantage: It is a time-consuming process that keeps the processor busy needlessly.



(a) Programmed I/O

1.7 I/O Communication Techniques (3/4)

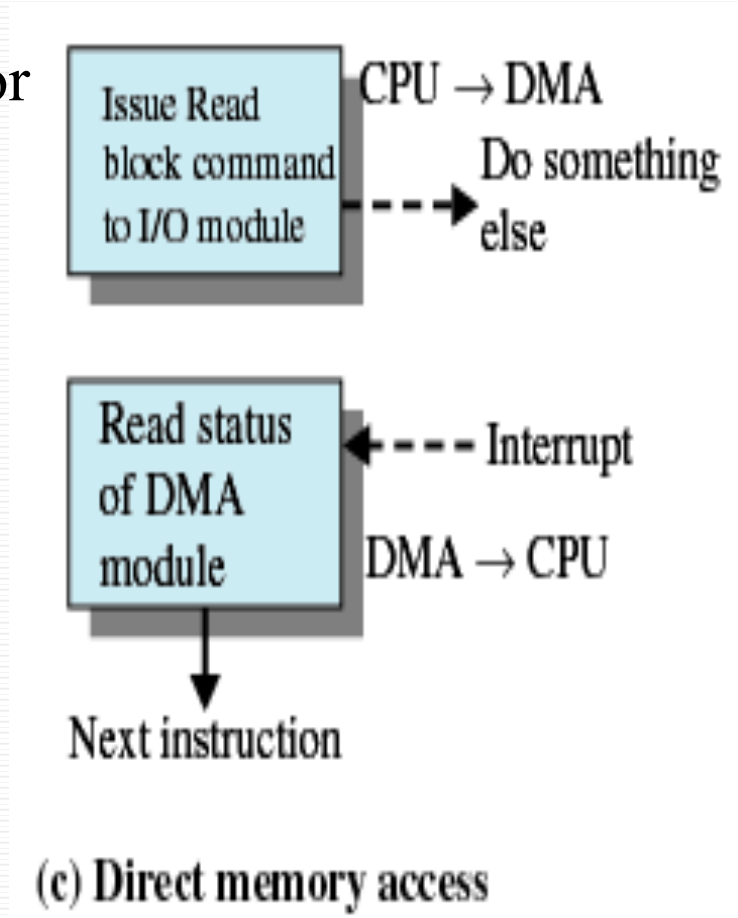
- Method2: Interrupt-Driven I/O
  - Processor is interrupted when I/O module ready to exchange data
  - Processor saves context of program executing and begins executing interrupt-handler
  - Advantage： No needless waiting, so more efficient than programmed I/O
  - Disadvantage ： Still consumes a lot of processor time because every word read or written passes through the processor



(b) Interrupt-driven I/O

1.7 I/O Communication Techniques (4/4)

- Method3: Direct Memory Access
  - Transfers a block of data directly to or from memory，I/O exchanges occur directly with memory
    - Processor grants I/O module authority to read from or write to memory
    - Processor continues with other work
    - An interrupt is sent when the transfer is complete
  - Advantage：Relieves the processor responsibility for the exchange



(c) Direct memory access