Chapter 1: Introduction

Prof. Li-Pin Chang

National Chiao Tung University

WHAT IS AN OPERATING SYSTEM?

What is an Operating System?

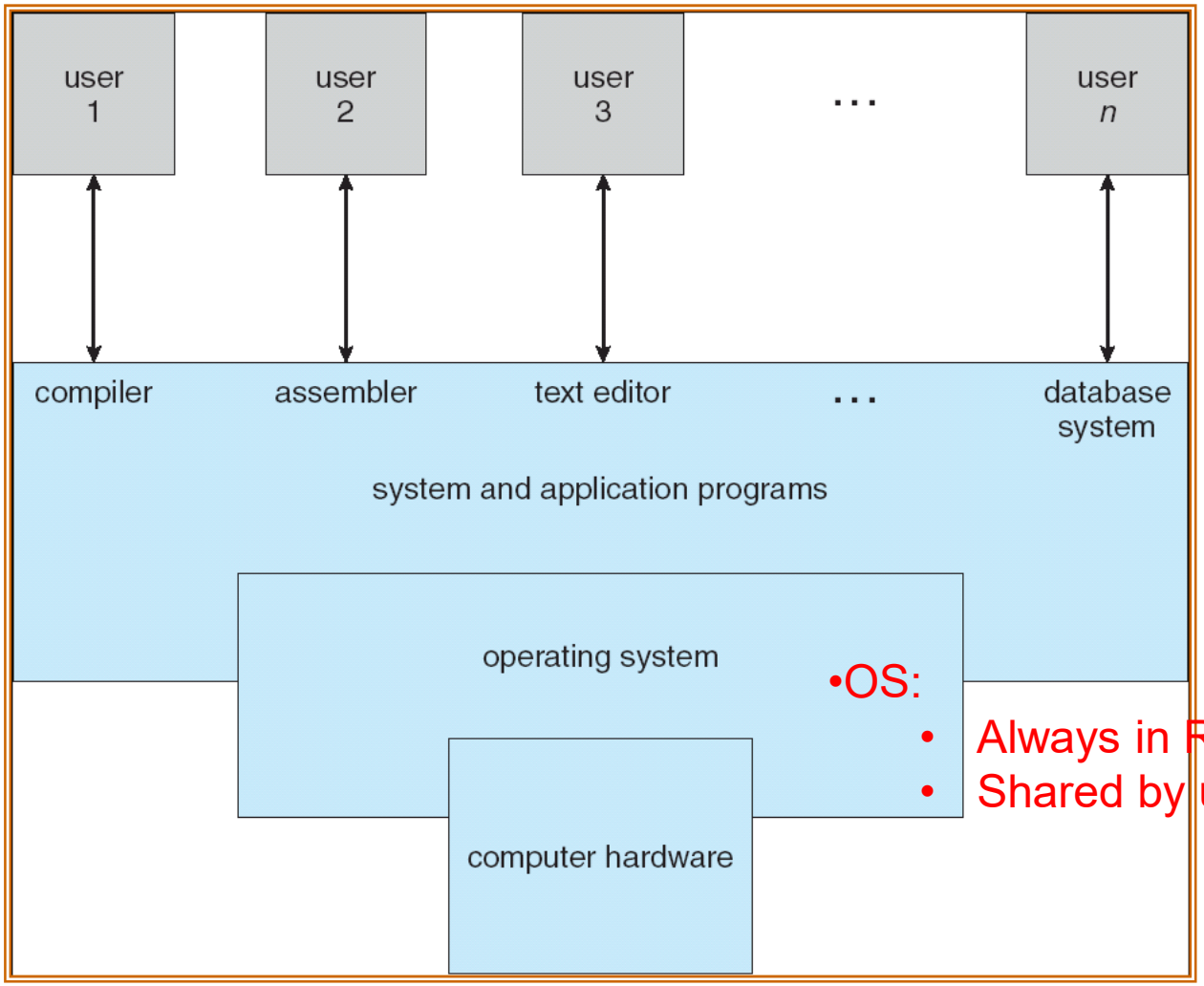
- A program that acts as an intermediary between a user of a computer and the computer hardware
- Operating system goals:
  - Execute user programs and make solving user problems easier
  - Make the computer system convenient to use
  - Use the computer hardware in an efficient manner

Computer System Structure

- Computer system can be divided into four components
  - Hardware – provides basic computing resources
    - CPU, memory, I/O devices
  - Operating system
    - Controls and coordinates use of hardware among various applications and users
  - Application programs – define the ways in which the system resources are used to solve the computing problems of the users
    - Word processors, compilers, web browsers, database systems, video games
  - Users
    - People or other computers
    - Some systems do not interact with human

Four Components of a Computer System

如果需要Access Hardware
需要呼叫OS來執行
→需要中斷、進入kernel mod



Operating System Definition

- OS is a resource allocator
  - Manages all resources
  - Decides between conflicting requests for efficient and fair resource use
- OS is a control program
  - Controls (or monitor) execution of programs to prevent errors and improper use of the computer

最大化每單位時間完成的工作數
→很大的工作可能被無限延後

先來的工作先做
→前面有大工作，後面的工作都會有很大的延遲

Operating System Definition (Cont)

- No universally accepted definition
- "*Everything a vendor ships when you order an operating system*" is good approximation
  - Looks like what Microsoft says…
- "*The one program running at all times on the computer*" is the kernel 做CPU排程、記憶體收集等
  - Everything else is either a system program (ships with the operating system) or an application program
- UNIX OS
  - A kernel
  - A collection of system programs

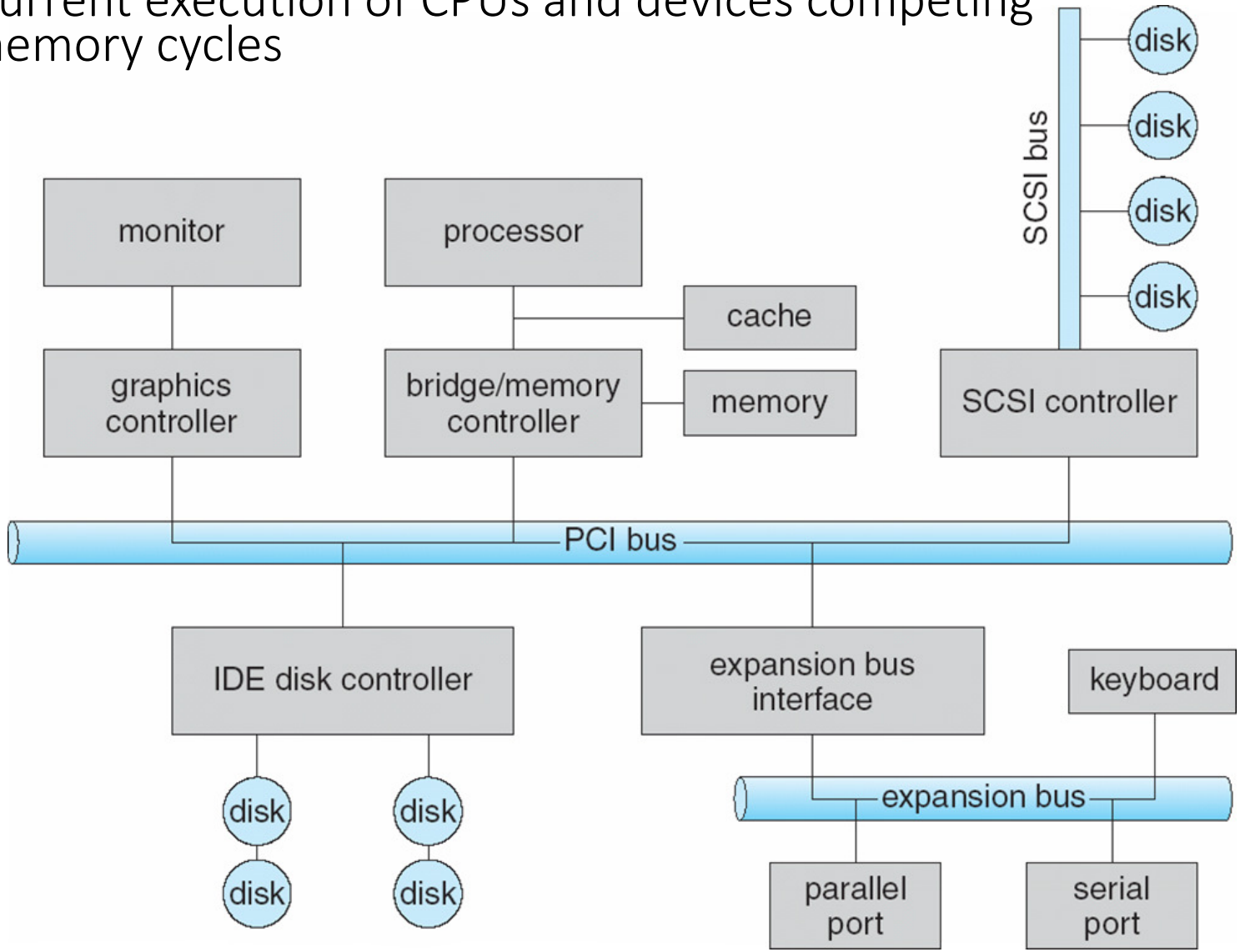
- Which one(s) of the following are part of operating systems?
  a) The CPU scheduler Kernel的一部分
  b) Device drivers 要操控硬體，是
  c) Compilers （有點tricky）
  d) Word processors

I/O STRUCTURE

I/O Hardware

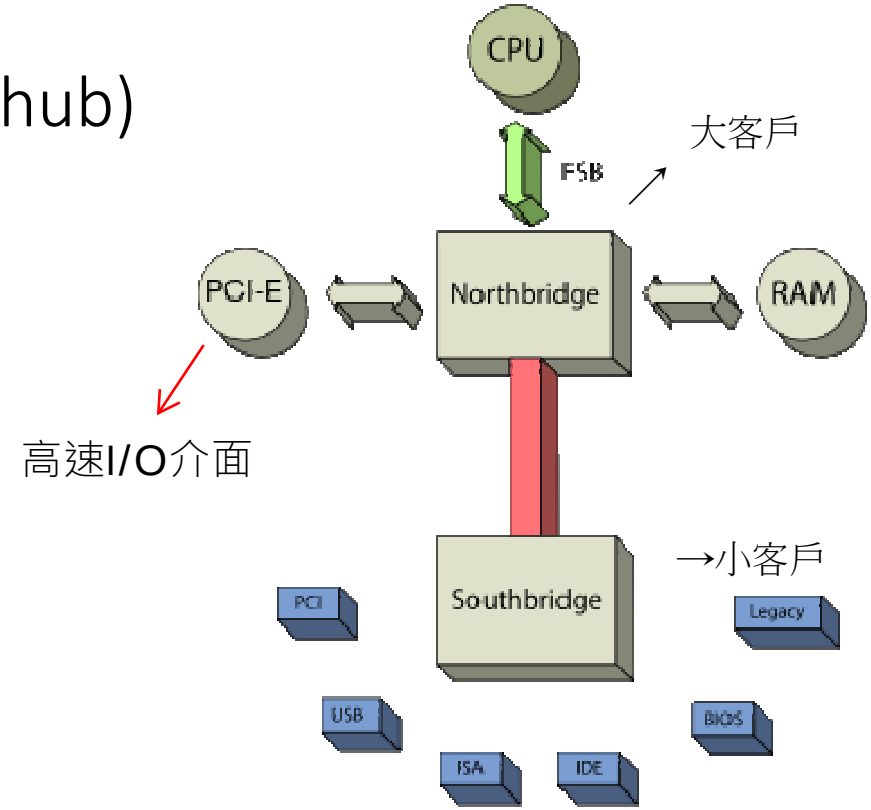
- Incredible variety of I/O devices
- Common concepts
  - Port
  - Bus (daisy chain or shared direct access)
  - Controller (host adapter)
- I/O instructions control devices
- Devices have addresses, used by
  - Direct I/O instructions
  - Memory-mapped I/O

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



PC Organization

- North bridge (memory hub)
  - CPU
  - Memory
  - High-speed devices
    - like display
- South bridge (IO hub)
  - IO
  - Network
  - USB
  - Hard drives
  - ...etc



https://en.wikipedia.org/wiki/Northbridge_(computing)

Device I/O Port Locations on PCs (partial)

I/O address range (hexadecimal)	device
000–00F	DMA controller
020–021	interrupt controller
040–043	timer
200–20F	game controller
2F8–2FF	serial port (secondary)
320–32F	hard-disk controller
378–37F	parallel port
3D0–3DF	graphics controller
3F0–3F7	diskette-drive controller
3F8–3FF	serial port (primary)

I/O Operation

I/O透過port對硬體操作

- I/O devices and CPUs execute concurrently
- Each device controller has a local buffer
- CPU or DMA moves data from/to main memory to/from local buffers
- I/O is from the device to local buffer of controller (and vice versa)
- I/O completion via polling or interrupt

  怎麼知道I/O完成? 等待(笨方法)/中斷(典型)

Polling

- Determines state of device
  - command-ready
  - busy
  - Error
- Busy-wait cycle to wait for I/O from device

Interrupts

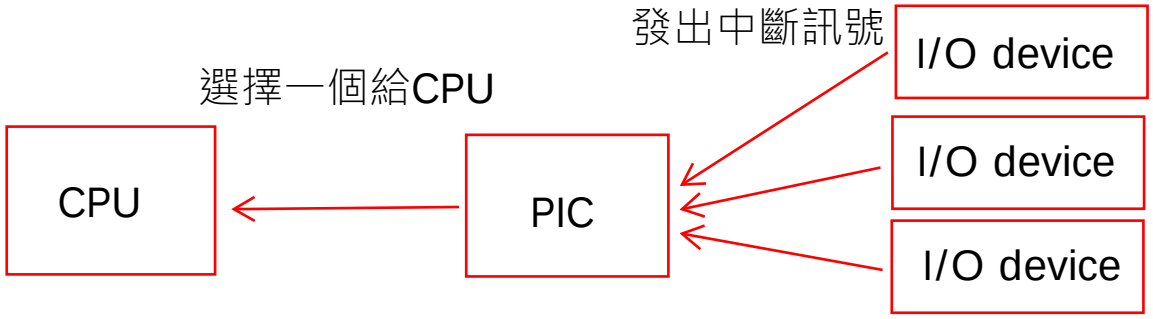
- CPU Interrupt-request line triggered by I/O device
- Interrupt handler receives interrupts
- Maskable to ignore or delay some interrupts
- Interrupt vector to dispatch interrupt to correct handler
  - Based on priority
  - Some nonmaskable
- Interrupt mechanism also used for exceptions

Common Functions of Interrupts

- Interrupt transfers control to the interrupt service routine generally, through the interrupt vector, which contains the addresses of all the service routines
- Interrupt architecture must save the address of the interrupted instruction
- Incoming interrupts are disabled while another interrupt is being processed to prevent a lost interrupt
- A trap is a software-generated interrupt caused either by an error or a user request
- An IRQ is generated by hardware
- An operating system is interrupt driven

- IRQ (interrupt request): hardware interrupts ← I/O completion, timer, etc.
- Trap: divide by zero, access violation, system call (from user to supervisor)

就直接掉到kernel裡處理



硬體產生中斷訊號通知CPU自己完成工作

Legacy PC interrupts

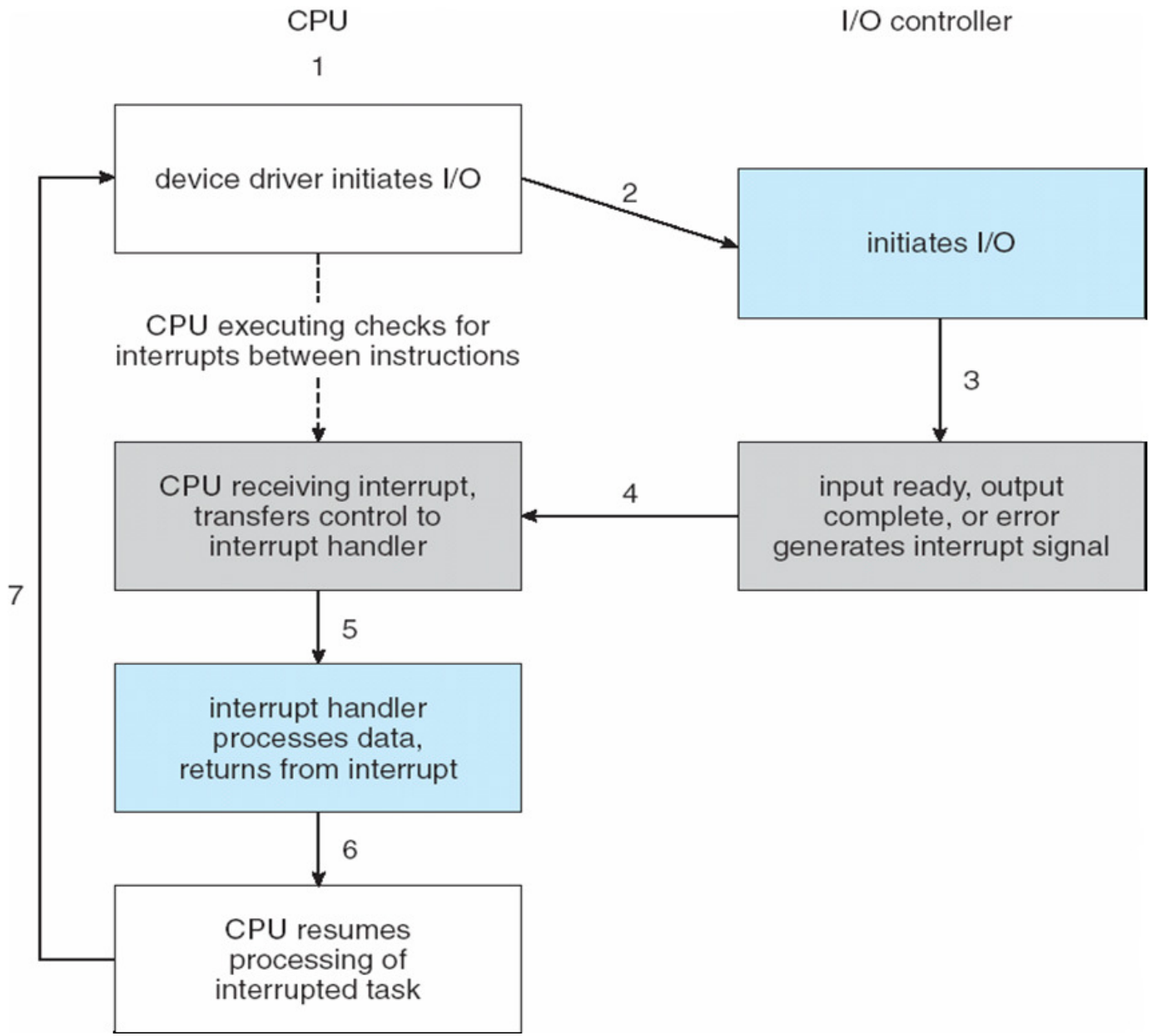
IRQ 0	System Timer
IRQ 1	Keyboard
IRQ 2	Cascaded with IRQ 9
IRQ 3	Default COM2 and COM4
IRQ 4	Default COM1 and COM3
IRQ 5	LPT2
IRQ 6	Floppy Drive Controller
IRQ 7	LPT1
…	

Interrupt	Address	Type	Function
00h	0000:0000h	Processor	Divide By Zero
01h	0000:0004h	Processor	Single Step
02h	0000:0008h	Processor	Nonmaskable Interrupt (NMI)
03h	0000:000Ch	Processor	Breakpoint Instruction
04h	0000:0010h	Processor	Overflow Instruction
05h	0000:0014h	BIOS/Software	Print Screen
05h	0000:0014h	Hardware	Bounds Exception (80286, 80386)
06h	0000:0018h	Hardware	Invalid Op Code (80286, 80386)
07h	0000:001Ch	Hardware	Math Coprocessor Not Present
08h	0000:0020h	Hardware	Double Exception Error (80286, 80386) (AT Only)
08h	0000:0020h	Hardware	System Timer - IRQ 0
09h	0000:0024h	Hardware	Keyboard - IRQ 1
09h	0000:0024h	Hardware	Math Coprocessor Segment Overrun (80286, 80386) (AT Only)
0Ah	0000:0028h	Hardware	IRQ 2 - Cascade from Second programmable Interrupt Controller
0Ah		Hardware	Invalid Task Segment State (80286, 80286) (AT Only)
0Ah		Hardware	IRQ 2 - General Adapter Use (PC Only)
0Bh	0000:002Ch	Hardware	IRQ 3 - Serial Communications (COM 2)
0Bh		Hardware	Segment Not Present (80286, 80386)
0Ch	0000:0030h	Hardware	IRQ 4 - Serial Communications (COM 1)
0Ch		Hardware	Stack Segment Overflow (80286, 80386)
0Dh	0000:0034h	Hardware	Parallel Printer (LPT 2) (AT Only)
0Dh		Hardware	IRQ 5 - Fixed Disk (XT Only)
0Dh		Software	General Protection Fault (80286, 80386)
0Eh	0000:0038h	Software	IRQ 6- Diskette Drive Controller
0Eh		Software	Page Fault (80386 Only)
0Fh	0000:003Ch	Software	IRQ 7 - Parallel printer (LPT 1)

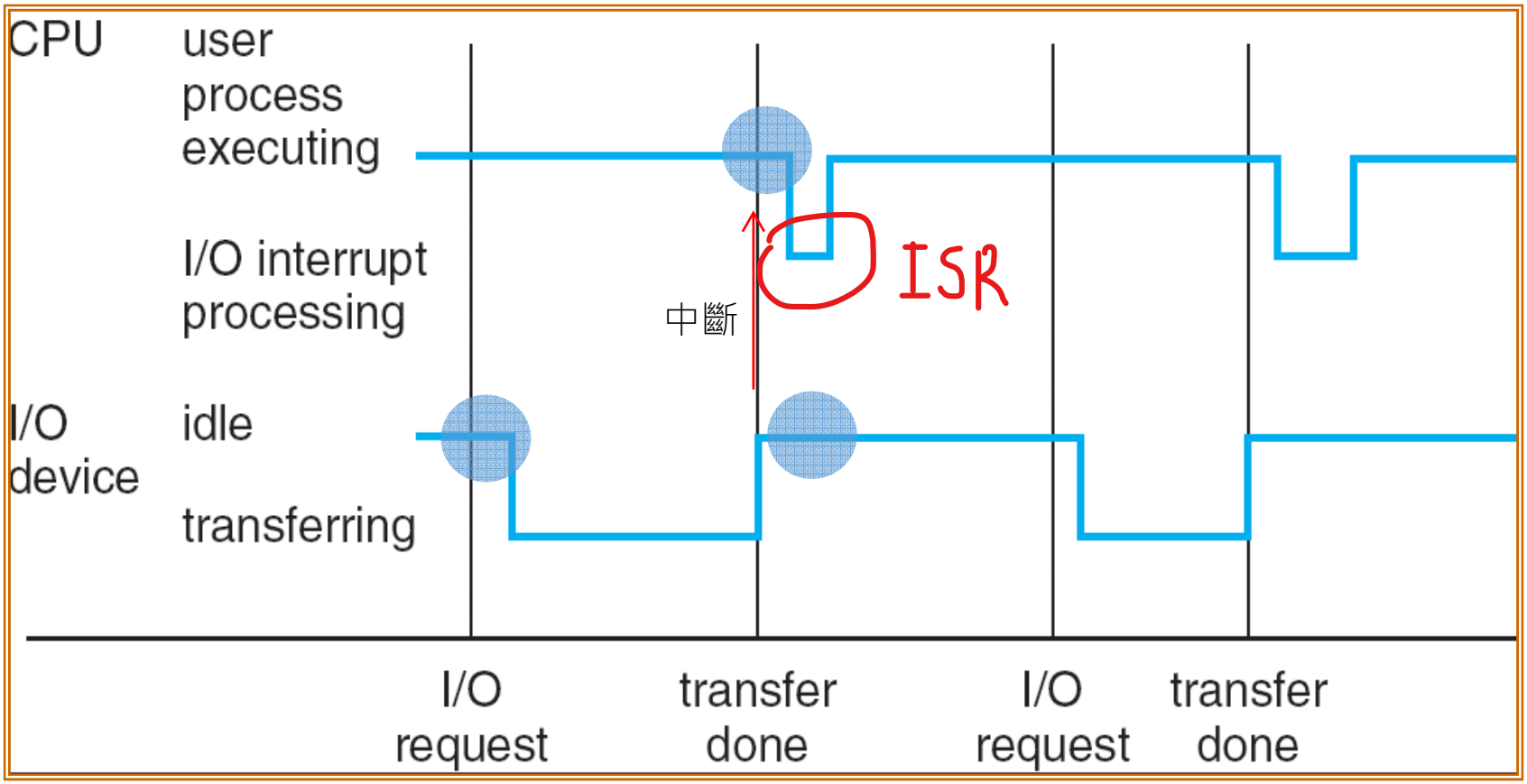
…

Interrupt Handling

- The operating system preserves the state of the CPU by storing registers and the program counter
- Determines which type of interrupt has occurred:
  - Reading I/O registers
  - vectored interrupt system
- Separate segments of code determine what action should be taken for each type of interrupt


CPU
I/O controller
1
device driver initiates I/O
2
initiates I/O
CPU executing checks for interrupts between instructions
3
CPU receiving interrupt, transfers control to interrupt handler
4
input ready, output complete, or error generates interrupt signal
7
5
interrupt handler processes data, returns from interrupt
6
CPU resumes processing of interrupted task

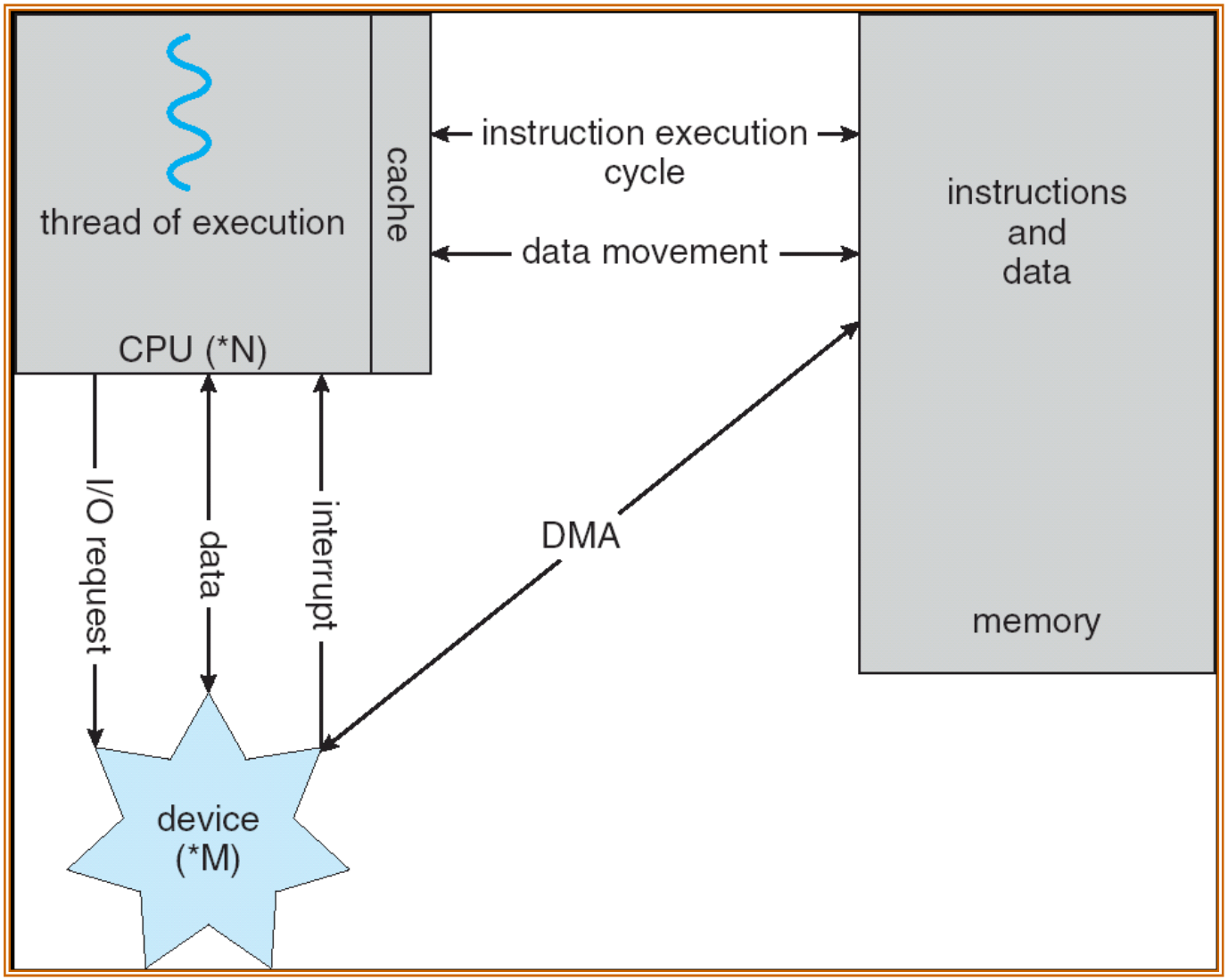
Interrupt Timeline



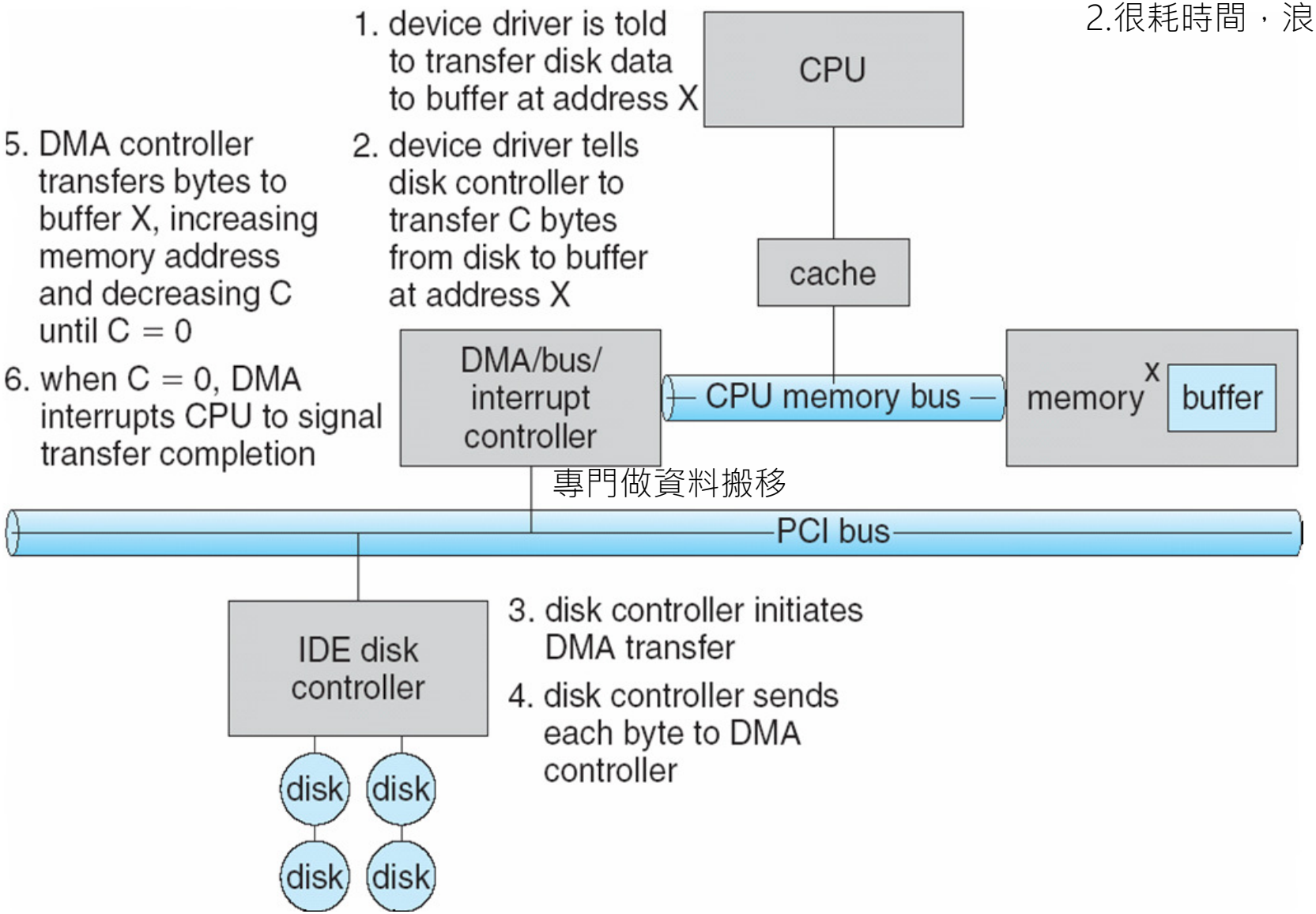
Direct Memory Access(DMA)

- Used for high-speed I/O devices able to transmit information at close to memory speeds
- Device controller transfers blocks of data from buffer storage directly to main memory without CPU intervention
- Only on interrupt is generated per block, rather than the one interrupt per byte

105


thread of execution
CPU (*N)
cache
instruction execution cycle
data movement
instructions and data
memory
I/O request
data
interrupt
DMA
device (*M)

CPU不適合做資料搬移
1.容易汙染cache
2.很耗時間，浪費CPU



From Ch13

真正發生的時間跟程式執行沒關係

Synchronous I/O vs. Asynchronous I/O

資料還沒讀出來時程式不能繼續
但並不代表CPU閒置(可以去做其他的)

- After I/O starts, control returns to the program only upon I/O completion (blocking call; sync I/O)
  - Example: I/O reading
  - read() or fread()

- After I/O starts, control returns to the program without waiting for I/O completion (non-blocking call; async I/O)
  - Example: I/O writing (without O_SYNC)
  - write() or fwrite()

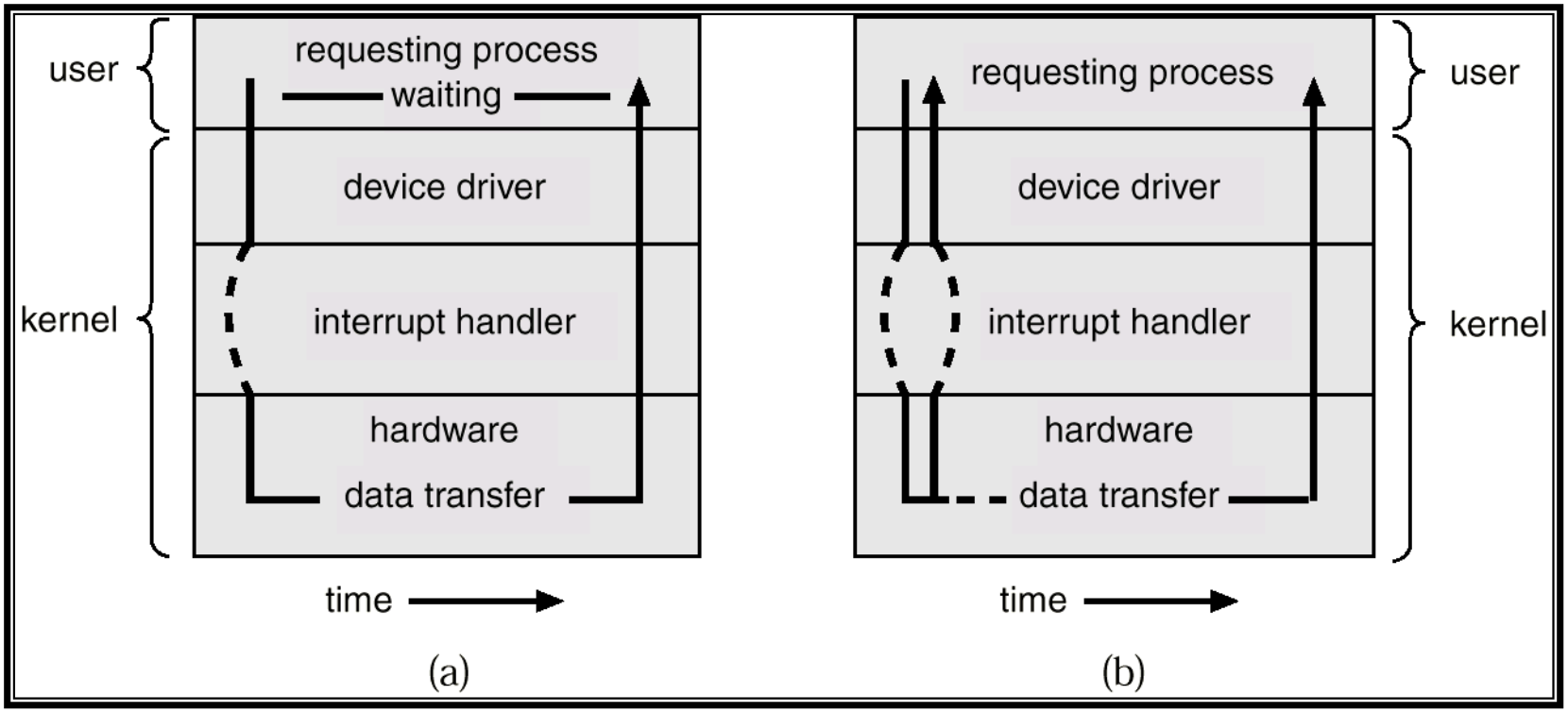
I/O Structure (Async vs Sync)

Synchronous

Blocking I/O

Asynchronous

Non-blocking I/O



- Are the following operations synchronous or asynchronous?
  - fread() s
  - fwrite() as
  - fsync() 確定將修改過的部分寫到磁碟，
  - aio_read() prefetch一些可能要用的部分(投機)
    aio = asynchronous I/O

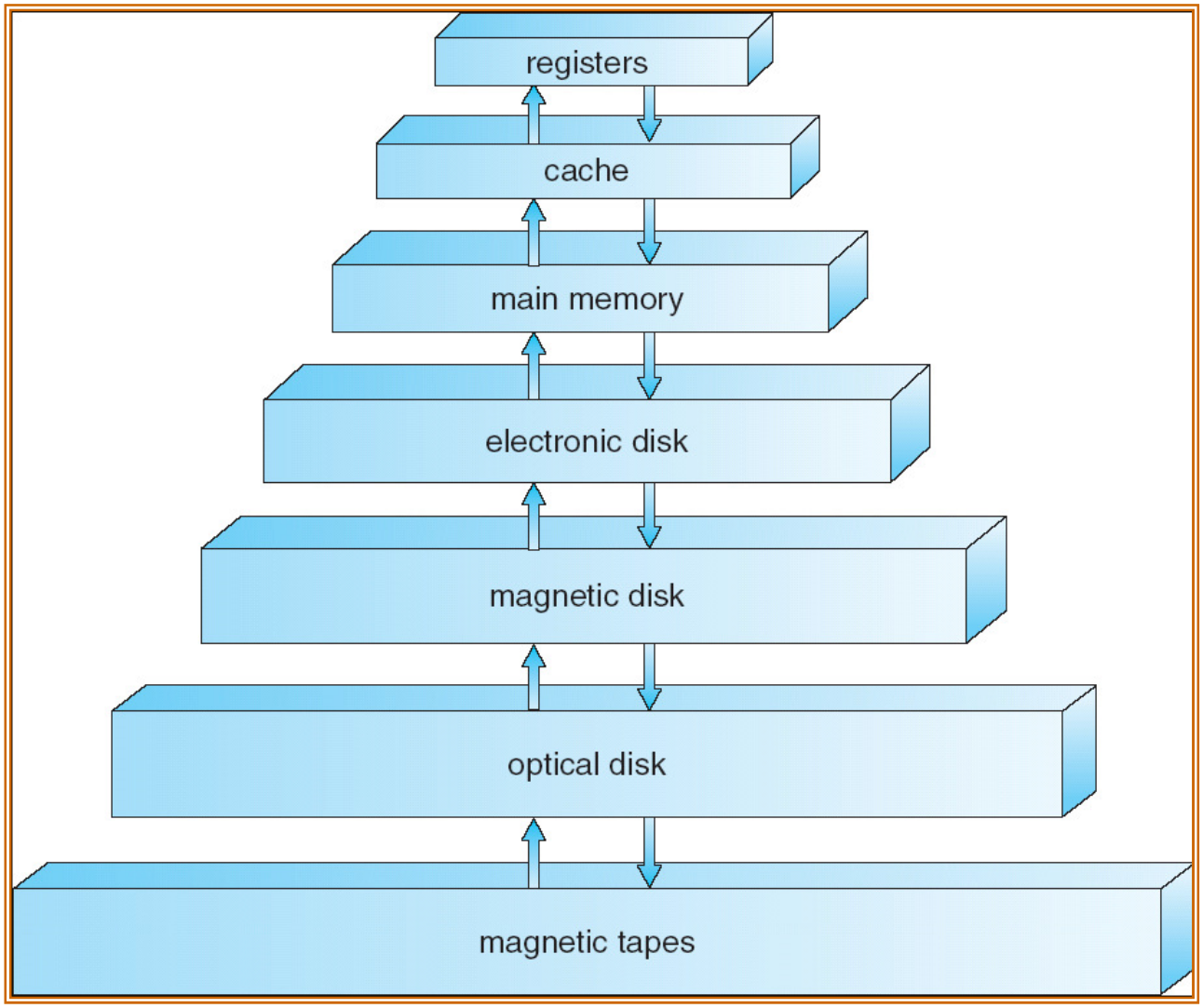
Memory/Storage Structure

Memory Hierarchy

- Main memory – the only large storage media that the CPU can access directly
  - RAM (random access)

- Secondary storage – extension of main memory that provides large nonvolatile storage capacity
  - No random access
  - Magnetic disks – rigid metal or glass platters covered with magnetic recording material
    - Disk surface is logically divided into tracks, which are subdivided into sectors
    - The disk controller determines the logical interaction between the device and the computer

Storage Structure (Memory hierarchy)



Storage Structure (Memory hierarchy)

- Storage systems organized in hierarchy
  - Speed
  - Cost
  - Volatility
- Caching – copying information into faster storage system; main memory can be viewed as a last cache for secondary storage

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 management important design problem
  - Cache size and replacement policy

- What we need?
  - An efficient lookup service
  - A replacement policy that minimizes cache misses

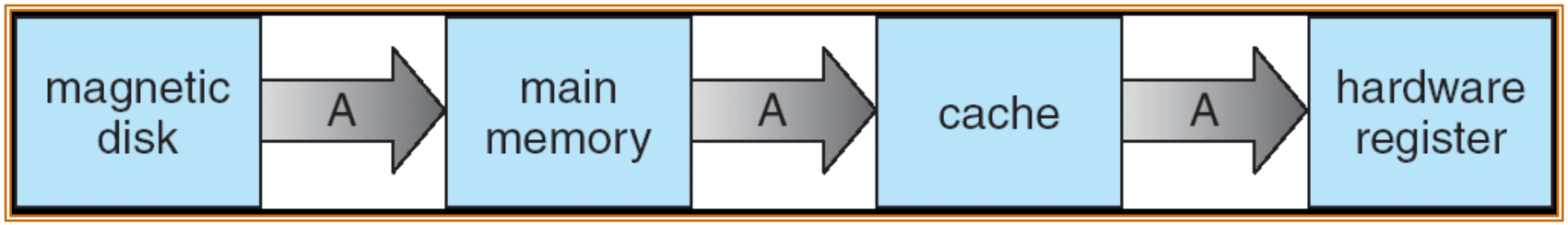
Performance of Various Levels of Storage

- Moving data among storage levels can be explicit or implicit

Level	1	2	3	4
Name	registers	cache	main memory	disk storage
Typical size	< 1 KB	> 16 MB	> 16 GB	> 100 GB
Implementation technology	custom memory with multiple ports, CMOS	on-chip or off-chip CMOS SRAM	CMOS DRAM	magnetic disk
Access time (ns)	0.25 – 0.5	0.5 – 25	80 – 250	5,000.000
Bandwidth (MB/sec)	20,000 – 100,000	5000 – 10,000	1000 – 5000	20 – 150
Managed by	compiler	hardware	operating system	operating system
Backed by	cache	main memory	disk	CD or tape

Migration of Integer A from Disk to Register

- Multitasking environments must be careful to use most recent value, not matter where it is stored in the storage hierarchy



- Multiprocessor environment must provide cache coherency in hardware such that all CPUs have the most recent value in their cache
- Distributed environment situation even more complex
  - Several copies of a datum can exist
  - A cache coherent protocol is needed

- Two important design issues for cache memory are _____

A) speed and volatility
B) lookup method and replacement policy
C) power consumption and reusability
D) size and access privileges

OPERATING-SYSTEM STRUCTURE

Before OS: Computer Startup

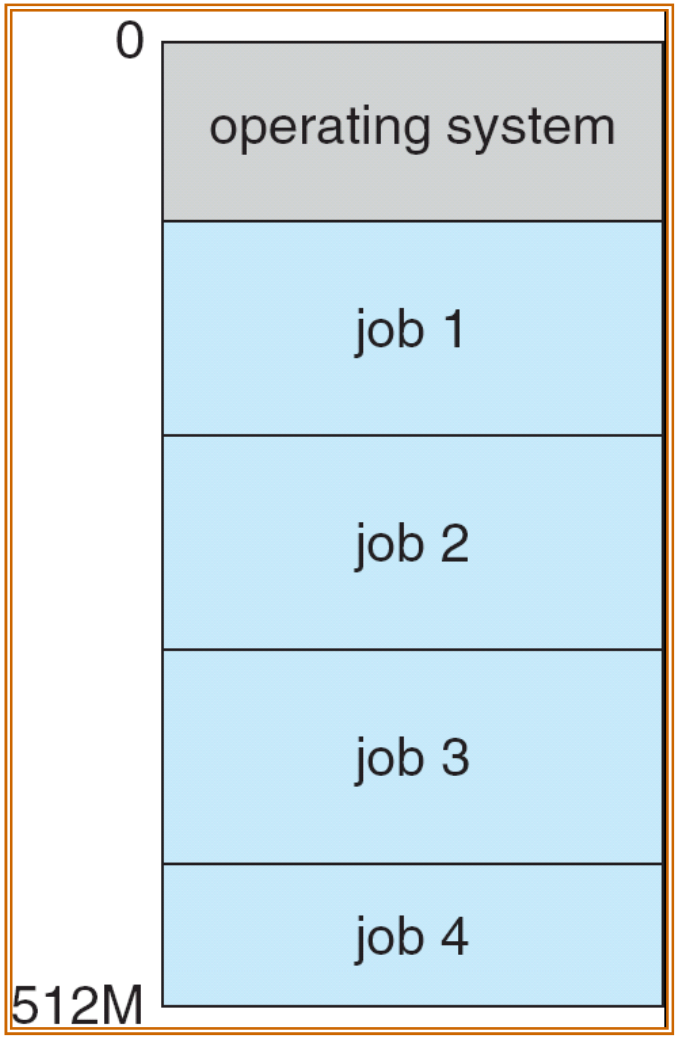
- Bootstrap program is loaded at power-up or reboot
  - Typically stored in ROM or EEPROM, generally known as firmware
  - Initializes all aspects of system
  - Loads operating system kernel and starts execution

Firmware is embedded in hardware devices

Operating System Structure

- Multiprogramming needed for efficiency
  - Single user cannot keep CPU and I/O devices busy at all times
  - Multiprogramming organizes jobs (code and data) so CPU always has one to execute
  - A subset of total jobs in system is kept in memory
  - One job selected and run via job scheduling
  - When it has to wait (for I/O for example), OS switches to another job

Memory Layout for Multiprogrammed System



Operating System Structure

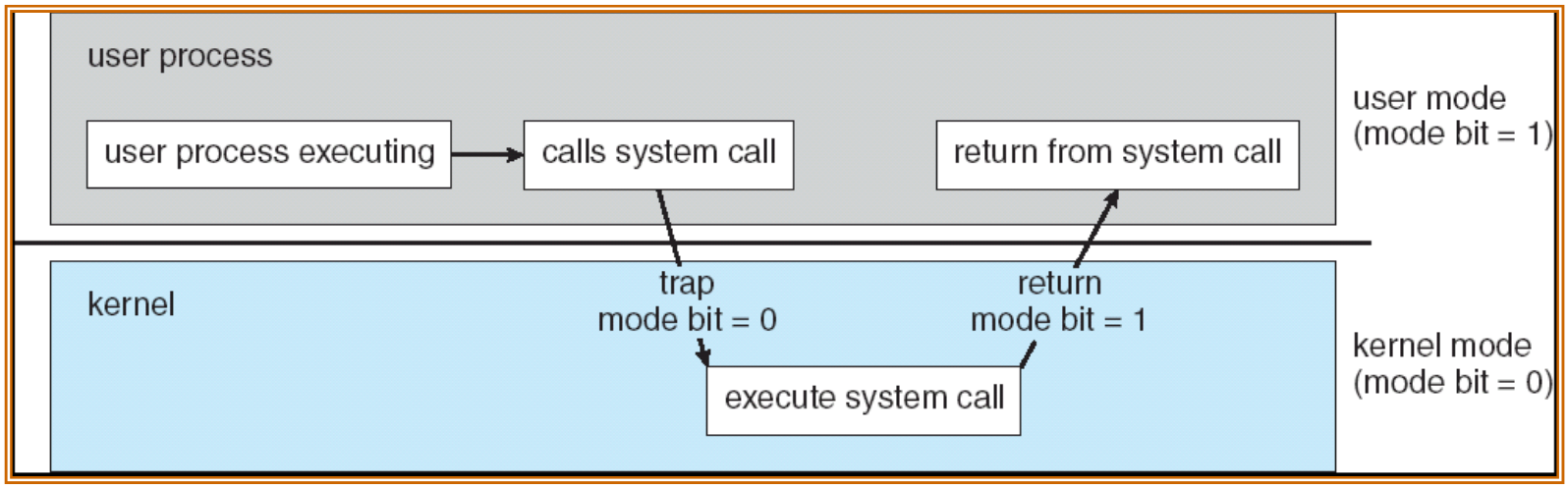
- Timesharing (multitasking) is logical extension in which CPU switches jobs so frequently that users can interact with each job while it is running, creating interactive computing
  - Response time should be < 1 second
  - Each user has at least one program executing in memory ⇨process (ch3)
  - If several jobs ready to run at the same time ⇨ CPU scheduling (ch5)
  - If processes don't fit in memory, swapping moves them in and out to run (ch5)
  - Virtual memory allows execution of processes not completely in memory (ch8)

- Multiprogramming
  - multiple processes are loaded into memory for execution
- Multitasking 程式執行是可交錯的，但程式用完才放CPU
  - Multiprogramming with overlapping task execution
- Timesharing
  - multitasking + periodic switch among processes
    週期性的強迫切換
- Issues
  - Which process should be run next?
  - Which process should be kept in/swapped out from memory?

Dual Mode Operations

- Interrupt driven by hardware
- Software error or request creates exception or trap
  - Division by zero, request for operating system service
  - Memory access violation
  - System calls
- Dual-mode operation allows OS to protect itself and other system components User / Kernel mode, CPU要支援
  - User mode and kernel mode
  - Mode bit provided by hardware
    - Provides ability to distinguish when system is running user code or kernel code
    - Some instructions designated as privileged, only executable in kernel mode
    - System call changes mode to kernel, return from call resets it to user

Transition from User to Kernel Mode



Trap/ exception

THE THREE PILLARS

Pillar 1: Process Management

- A process is a program in execution It is a unit of work within the system Program is a passive entity, process is an active entity
- Process needs resources to accomplish its task
  - CPU time for execution
  - I/O for communication
  - Files for data storage
- A system has many processes running concurrently on one or more CPUs

Process Management Activities

- The operating system is responsible to the following activities in connection with process management:
  - Creating and deleting processes
  - Suspending and resuming processes
  - Providing mechanisms for process communication
  - Providing mechanisms for process synchronization
  - Providing mechanisms for deadlock handling

- A _____ can be used to prevent a user program from never returning control to the operating system

A) portal

B) program counter

C) firewall

D) timer

Pillar 2: Memory Management

- All data in memory before and after processing
- All instructions in memory in order to execute
- Memory management activities
  - Allocating and deallocating memory space as needed
  - Deciding which processes (or parts thereof) and data to move into and out of memory (swapping or paging)
  - Keeping track of which parts of memory are currently being used and by whom (protection)

Pillar 3: Storage Management

- OS provides uniform, logical view of information storage
  - Abstracts physical properties to logical storage unit - file
- File-System management
  - Files usually organized into directories
  - Access control on most systems to determine who can access what
- File system basic activities
  - Creating and deleting files and directories
  - Reading and writing files and directories
  - Mapping files from secondary storage to main memory

Mass-Storage Management

- Entire speed of computer operation hinges on disk subsystem and its algorithms
- OS activities
  - Storage allocation
  - Free-space management
  - Disk scheduling
- Some storage need not be fast
  - Tertiary storage includes optical storage, magnetic tape
  - Varies between WORM (write-once, read-many-times) and RW (read-write)

I/O Subsystem

- One purpose of OS is to hide peculiarities (特點) of hardware devices from the user
- I/O subsystem responsible (主管) to
  - Buffering 銜接不同速度的裝置
    - Hide the latency of the underlying I/O
    - Couple the mismatch of transfer units between I/O and applications
  - Caching
    - Copy data into faster memory for later reference
  - Spooling 讓多個裝置同時I/O不會混雜
    - Overlap device output with the input of data (e.g., printer)

The issue of resource utilization shows up in different forms in different types of operating systems

List the resources that must be carefully managed in one kind of systems but not in the other kind of systems:

a. Mainframe computers
b. Handheld computers

End of Chapter 1