Chapter 2 Computer System Structures

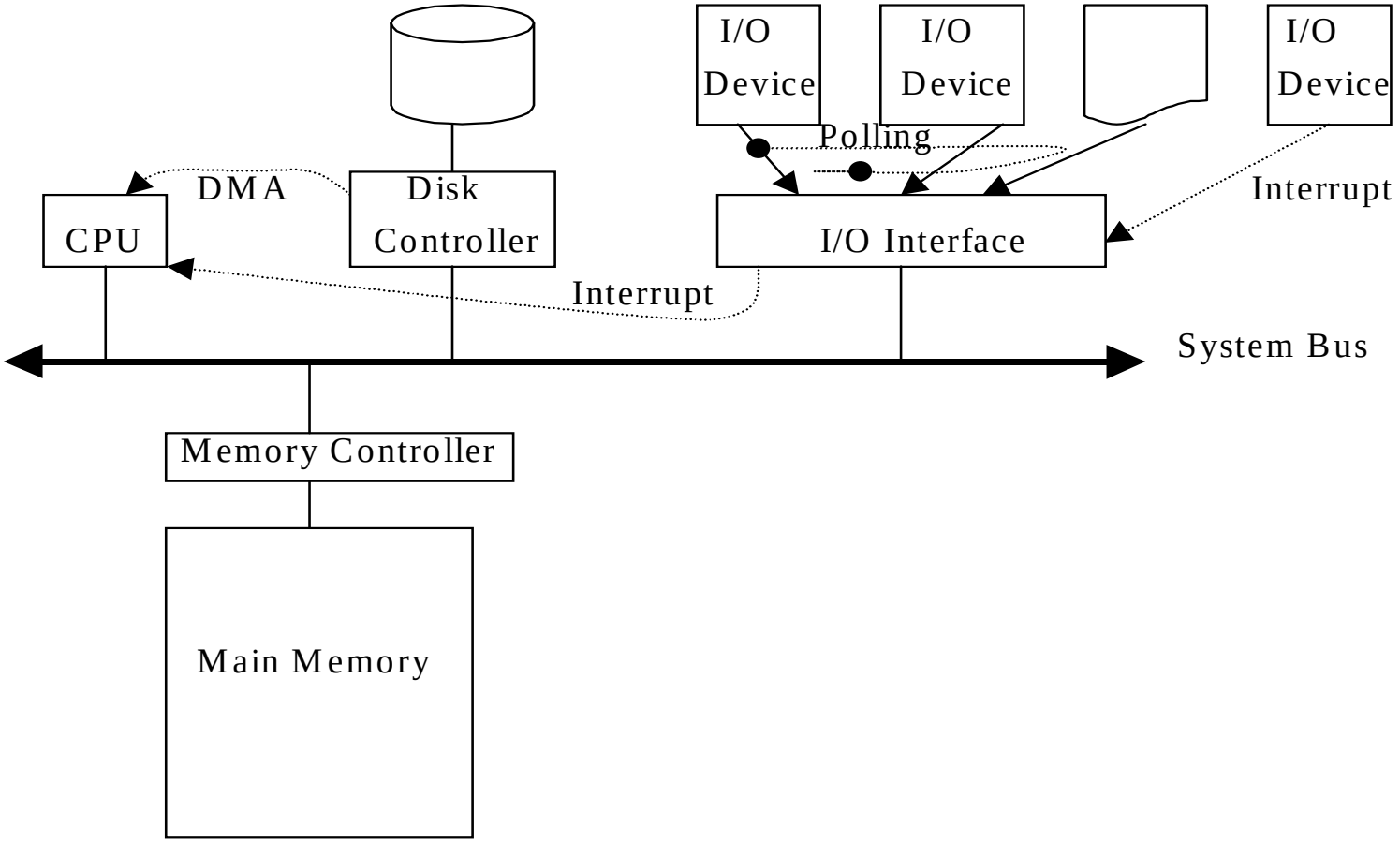


圖 1.15　現代電腦的硬體結構

Modern Computer System(1)

2.1 Computer System Operation

- 任何程式欲被執行，都必須由硬碟中經由匯流排搬至主記憶體，然後再透過中央處理器執行。

- 控制硬碟工作的介面是硬碟控制器（Disk Controller），控制輸出 / 輸入設備的介面是輸出 / 輸入設備介面卡（Interface Card）。

Modern Computer System(2)

- 當中央處理器正在工作的同時，硬碟控制器可能也正在進行讀 / 寫的工作，而輸出 / 輸入設備介面卡也可能同時在作輸出 / 輸入的工作，它們是並行（Concurrent）的在執行工作。

- 記憶體控制器（Memory Controller ）出面安排它們使用主記憶體的次序，這是因爲主記憶體在同一時間只能有一個設備使用。

- The CPU and the device controllers can execute concurrently, competing for memory cycles. A memory controller is provided to synchronize access to the memory.

Modern Computer System(3)

- 中央處理器及設備在使用主記憶體時，記憶體控制器提供同步（Synchronous）的功能。
- 同步是指依某種次序安排中央處理器及設備使用主記憶體，而不允許它們同時使用主記憶體。
- 程式由中央處理器執行的過程中，有可能某個設備已完成工作，它必須通知中央處理器，這時設備就必須發出中斷（Interrupt）訊號；中央處理器亦可能需要隨時監看某一些設備的狀態，則必須使用輪詢（Polling）功能 。

Modern Computer System(4)

- 中斷與輪詢不適用於大量的資料傳輸，此時直接記憶體存取（Direct Memory Access）簡稱DMA，它就派上用場。

輪詢(Polling)

- 一個主單元輪流去詢問各個從屬單元的狀態。

- One master unit circularcheck the status of one or several slave unit.

中斷(Interrupt)

- 在電腦中，某個設備或程式欲通知中央處理器，並請中央處理器處理事情，則它們可以發出中斷，引起中央處理器的注意，並使得中央處理器停止正在進行的工作，而先去處理中斷事務。

- Interrupt is an event that alters the sequence in which a processor executes instruction.

- Interrupt increase CPU throughput.

中斷及輪詢的比較

- 中斷相對有很高的效率，它增加中央處理器之工作量；但是在電腦系統中，有時輪詢還是無法避免的。例如設備介面卡接了8個設備，當有設備以中斷通知中央處理器作處理時，中央處理器必須輪詢8個設備，以確定那個設備發生中斷，然後再進行處理。

Interrupt Service(1)

2.2 Interrupt

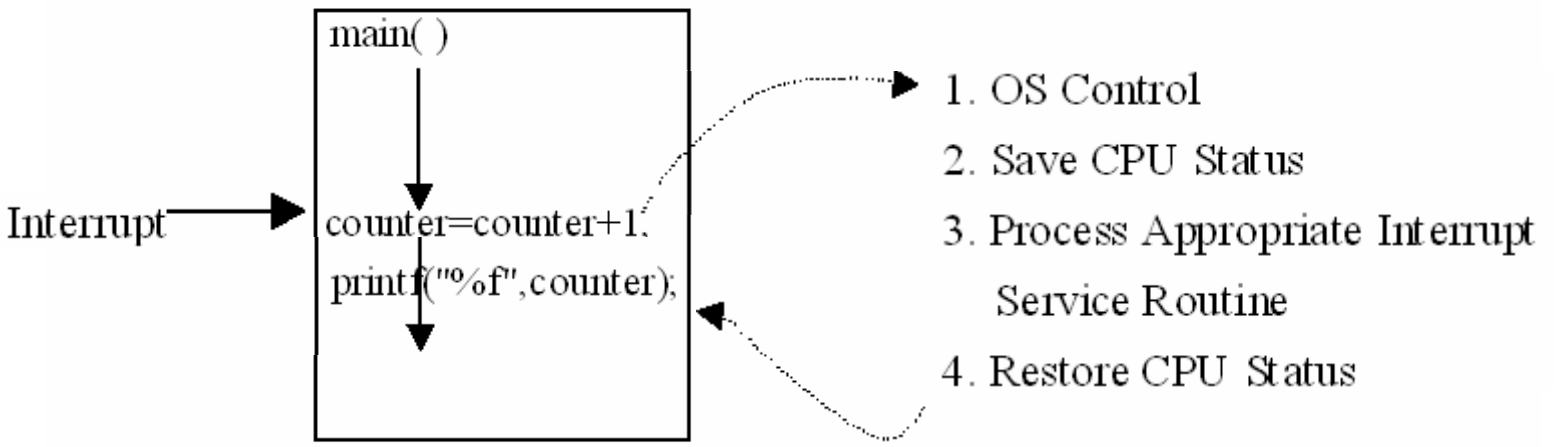


圖 1.16　作業系統處理中斷的步驟

Interrupt Service(2)

- 中斷發生後，作業系統必須處理中斷的步驟。
  - 因中斷關係，即刻將程式控制權交回作業系統。
  - 作業系統把main( )這個程式執行到一半的狀態儲存起來（包含暫存器，程式計數器...等）。
  - 作業系統判斷是發生那一種中斷，且呼叫適當的中斷服務常式（Interrupt Service Routine）執行。
  - 當中斷服務常式執行完畢後，作業系統將已儲存的狀態回復至暫存器，程式計數器...內。
  - 跳回main( )內，執行中斷時的下一個敘述。

中斷的種類(Interrupt Type)

- 外部中斷（External Interrupt）
  - 由硬體所產生的中斷。
- 內部中斷（Internal Interrupt）
  - 因爲錯誤或不合法所造成的中斷，例如：堆疊溢位（Stack Overflow）、除以零（Divided by Zero）、違反保護（Protect Violation）。
- 軟體中斷（Software Interrupt ）
  - 系統呼叫是屬於軟體中斷。系統呼叫有時又稱爲監督者呼叫（Supervisor Call），簡稱SVC。

中斷的另一種分類

- 監督者呼叫（Supervisor Call）
  - System call
- 輸出入中斷(I/O Interrupt)
- 外部中斷（External Interrupt）
  - Clock, Console
- 開機中斷（Restart Interrupt）
- 程式測試中斷（Software Check Interrupt ）
- 機器測試中斷(Machine Check Interrupt)

陷阱（Trap）

- 系統呼叫這個軟體中斷是事先安排且可預期的，因此我們說系統呼叫是一種陷阱（Trap）。

- 所謂陷阱，是指軟體所產生的中斷；其他如除以零，不合法使用記憶體...等軟體所產生之錯誤，均稱爲陷阱。
- A trap is a software generated interrupt caused either an error or by a specific request from user program. e.g. divided by zero, invalid memory access, SVC.

系統呼叫處理步驟(1)

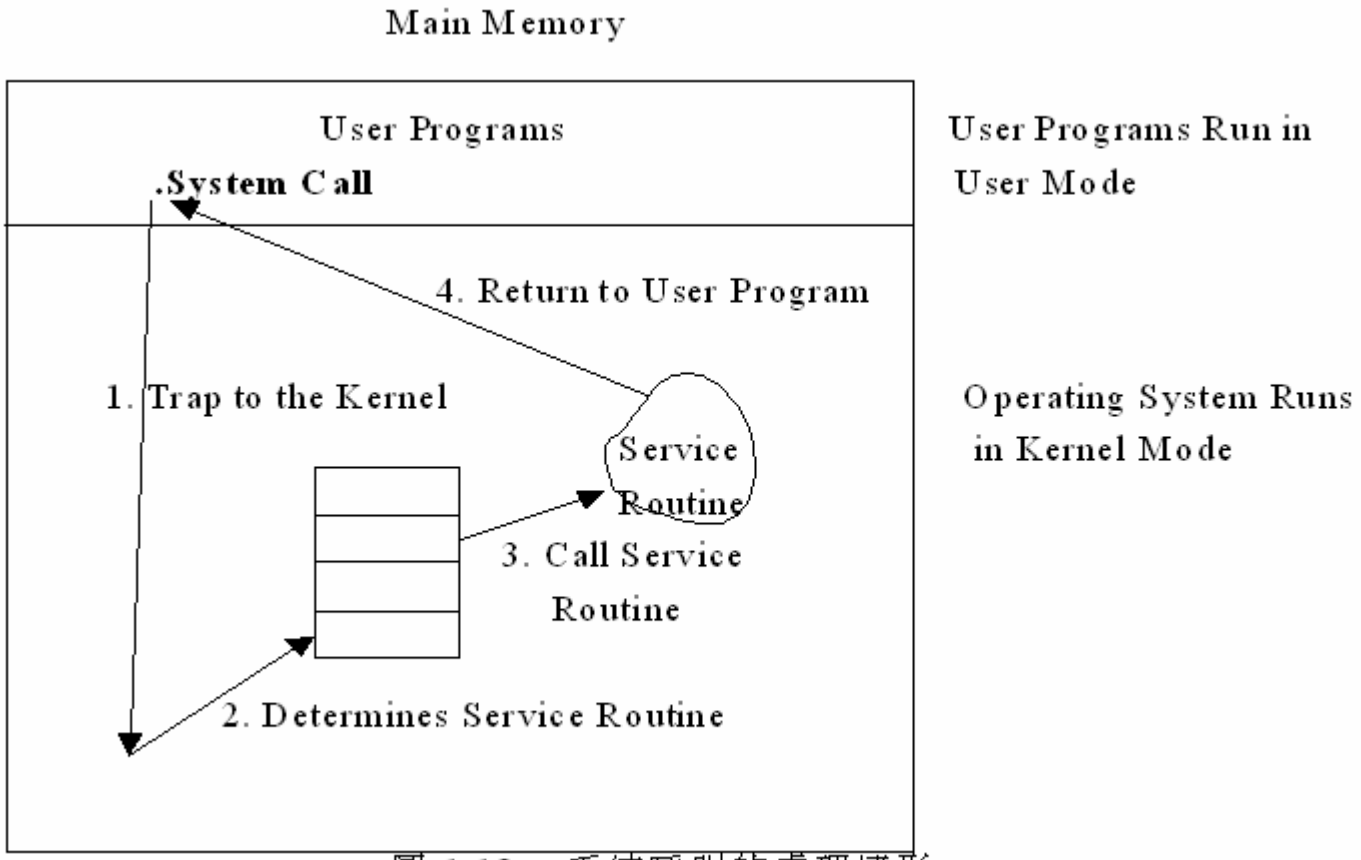


圖 1.18　系統呼叫的處理情形

系統呼叫處理步驟(2)

- 當使用者程式產生陷阱之後，執行次序便跳至監督者模式。
- 作業系統決定作那一種服務。
- 執行中斷服務。
- 執行完畢回到原來陷阱之後，繼續執行。

中斷的優先等級(Interrupt Priority)(1)

- 當某個中斷發生後，正式處理此中斷時，若還有較高優先等級的中斷發生，則正在處理的中斷服務常式會被中斷，而由較高優先等級的中斷服務常式來執行。
- A higher priority interrupt will be taken event if a low priority interrupt is active.

- 編號小的優先等級高。

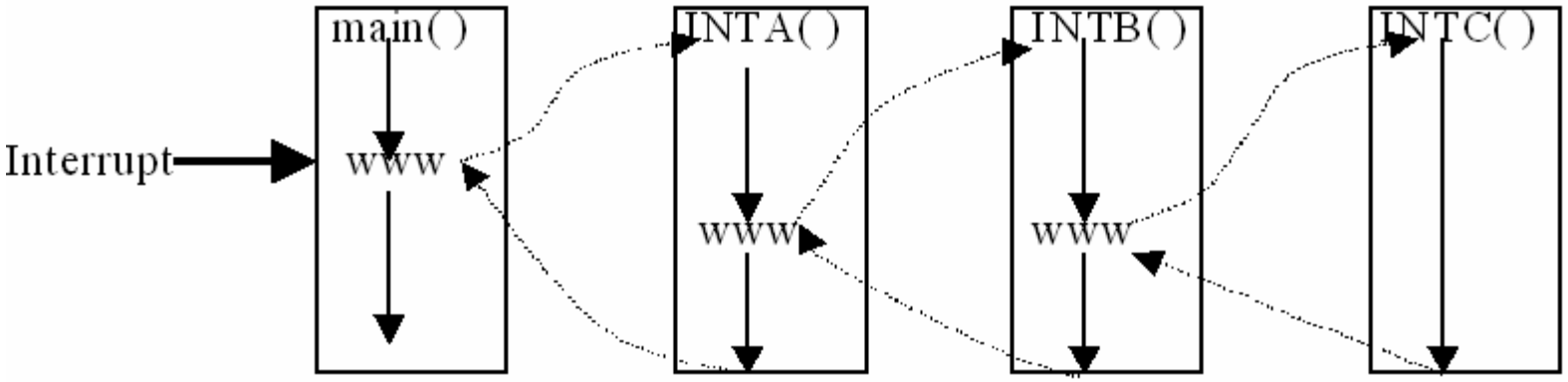


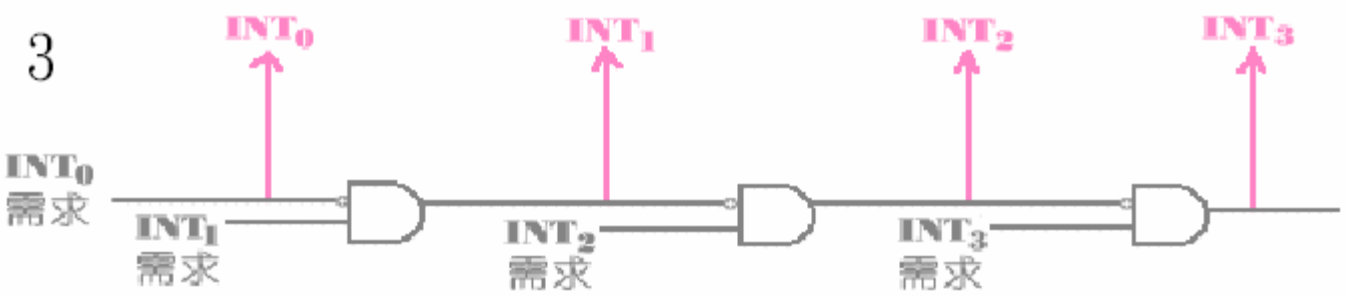
圖 1.17　較高優先等級中斷取代較低優先等級中斷的處理情形

中斷的優先等級(Interrupt Priority)(2)

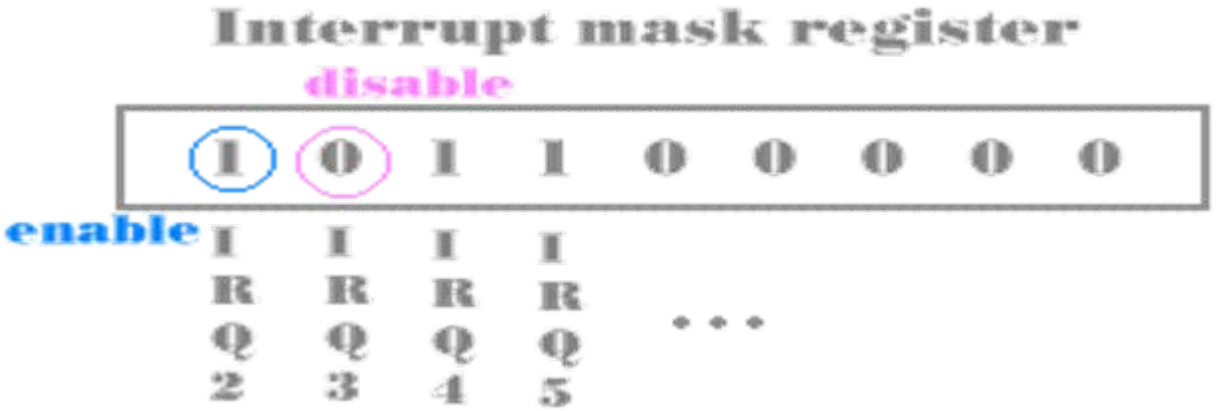
- Polling
- Daisy Chain
  - 分辨low, high priority。
- Parallel Priority Interrupt (Independent Request)

Daisy Chain and Parallel Priority Interrupt

- Daisy Chain



- Parallel Priority Interrupt



中斷控制反應(Interrupt Control Classes)

- 中斷發生後可以有三種控制反應
  - 立即回應（Occur）:例如系統呼叫，系統保護...等中斷均屬於高優先等級，必須立刻回應。
  - 暫緩處理（Pending）:例如輸出 / 輸入中斷，若有其他更重要的中斷發生，輸出 / 輸入中斷可以暫緩處理，但最後還是要處理。
  - 忽略（Ignore）:例如溢位中斷，平時規定要處理，但有時我們可以選擇忽略處理，也就是不去處理它。

I/O Interrupts

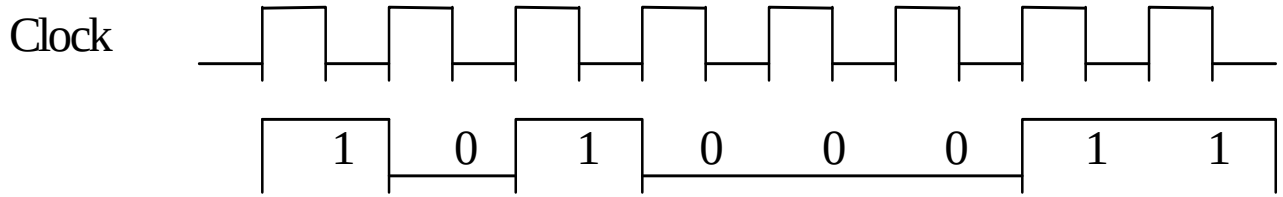
2.3 I/O Structure

- 同步輸入/輸出(Synchronous I/O)
  - Wait I/O
  - I/O start, then I/O completion, control is return to the user process. e.g. scanf()

- 非同步輸入/輸出(Asynchronous I/O)
  - No wait I/O 各作各的
  - Return control to the user program without waiting for the I/O to complete. e.g. kbhit().
  - The main advantage of asynchronous I/O is the increased system efficiency. While I/O is taking place, the CPU can be used for processing or even scheduling other I/O.

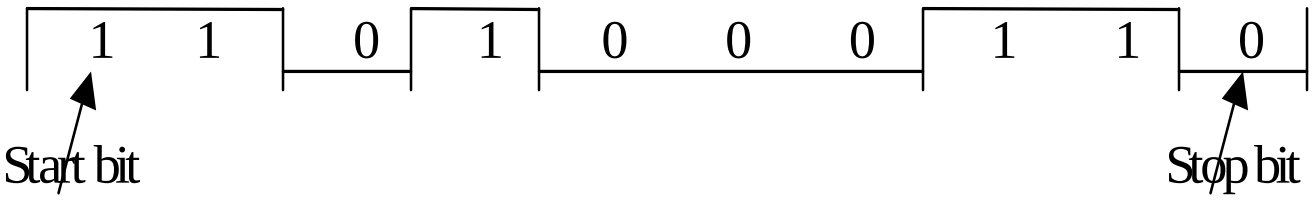
同步輸入/輸出(Synchronous I/O)

- 傳送與接收的雙方共用一個時鐘脈衝(Clock)。



非同步輸入/輸出(Asynchronous I/O)

- 在傳送的資料內加上同步的信號，當接收端偵測到同步信號時，便知道接下來的信號是資料。



直接記憶體存取(Direct Memory Access)(1)

- 每次中斷僅處理少量的資料，這對速度慢的設備沒問題。

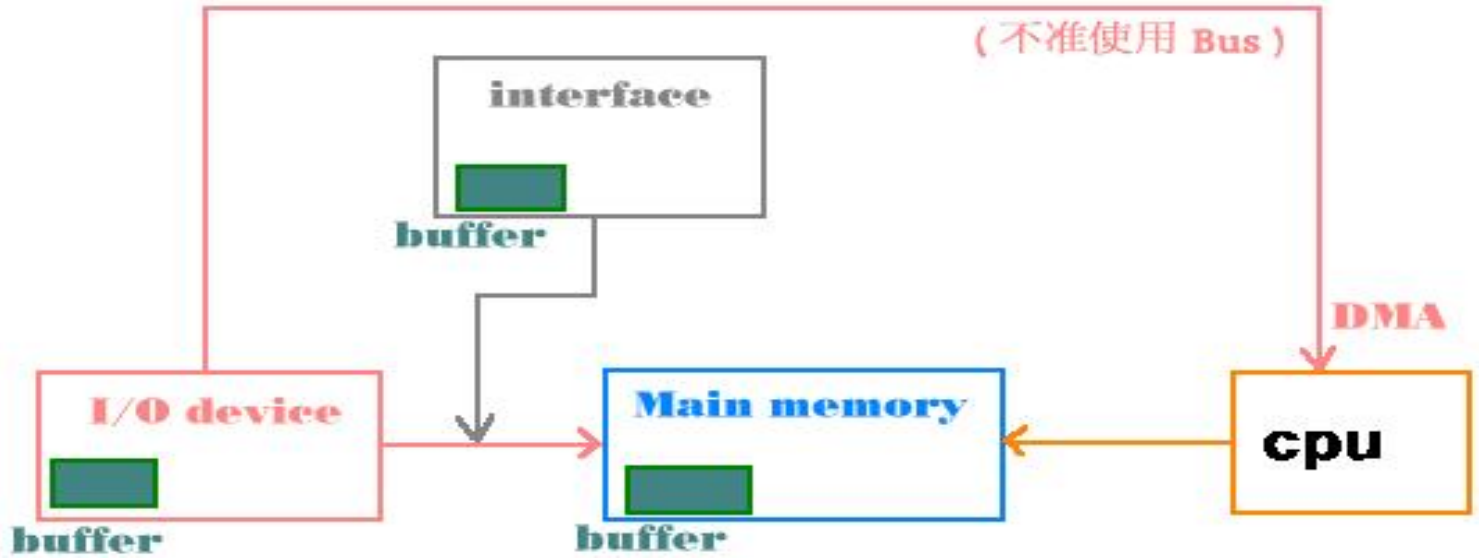
- 對高速設備卻要經過中斷處理步驟，並請中斷服務常式處理，往往耗費龐大的時間，則徒增浪費，此時我們就必須用到直接記憶體存取功能。

直接記憶體存取(Direct Memory Access)(2)

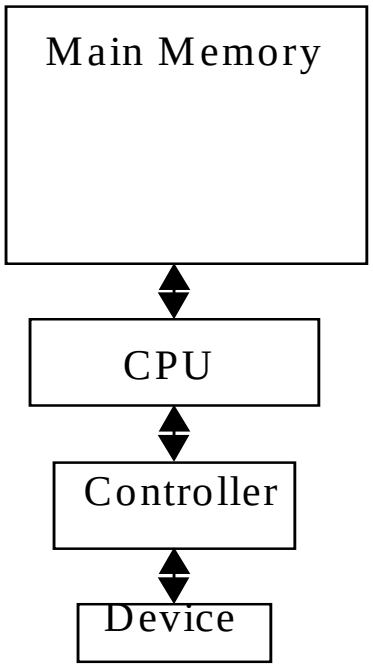
- 直接記憶體存取，就是當中央處理器被中斷之後，它用不著儲存中央處理器狀態（一般中斷要儲存暫存器，程式計數器...等狀態），而是讓中央處理器閒置到直接記憶體存取完畢；當中央處理器閒置時，週邊設備直接控管匯流排，並經由匯流排傳送大量資料至主記憶體內，或從主記憶體內取回大量資料。

- When interrupt CPU, CPU status not saved, it only delay(idle) CPU processing until DMA complete. The device controller transfer an entire block of data to/from its own buffer from/to memory directly.

直接記憶體存取(Direct Memory Access)(3)



直接記憶體存取(Direct Memory Access)(4)



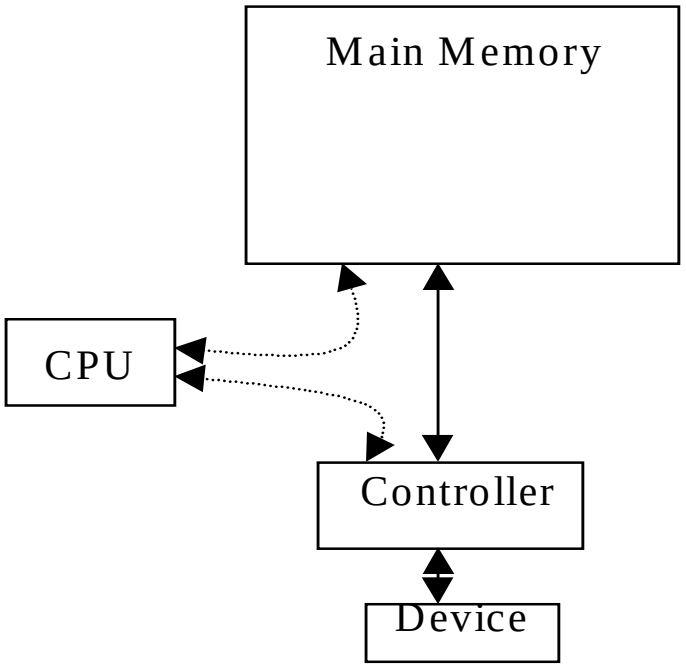


圖 1.19a　利用中斷處理輸出 / 輸入　　圖 1.19b　直接記憶體存取

圖 1.19　直接記憶體存取與傳統中斷處理輸出 / 輸入的方式

直接記憶體存取(Direct Memory Access)(4)

- 直接記憶體存取功能有以下特性：
  - 一次傳送大量的資料。
  - 減少中斷的次數。
  - 適用於高速輸出 / 輸入設備。
  - 增加輸出 / 輸入設備之產量。

週期盜取（Cycle Stealing）

- 當週邊設備要求進行主記憶體存取時，它中斷中央處理器，它用不著儲存中央處理器狀態，並使中央處理器延遲一個記憶體週期（Memory Cycle），週邊設備利用這極短的時間，至主記憶體內存取一或二個位元組。
- When I/O request memory access, it may cause a CPU request to be delayed one memory cycle period.

- 另一種說法是趁中央處理器在作計算而不用Bus時，進行memory access.

緩衝區(Buffer)(1)

- 爲了達到高速設備與低速設備之緩衝，此兩設備之間必須有緩衝區（Buffer）。
- A buffer is an area of memory for holding data during I/O transfer.
- Buffering is a method of overlapping the I/O of a job with its own computation.

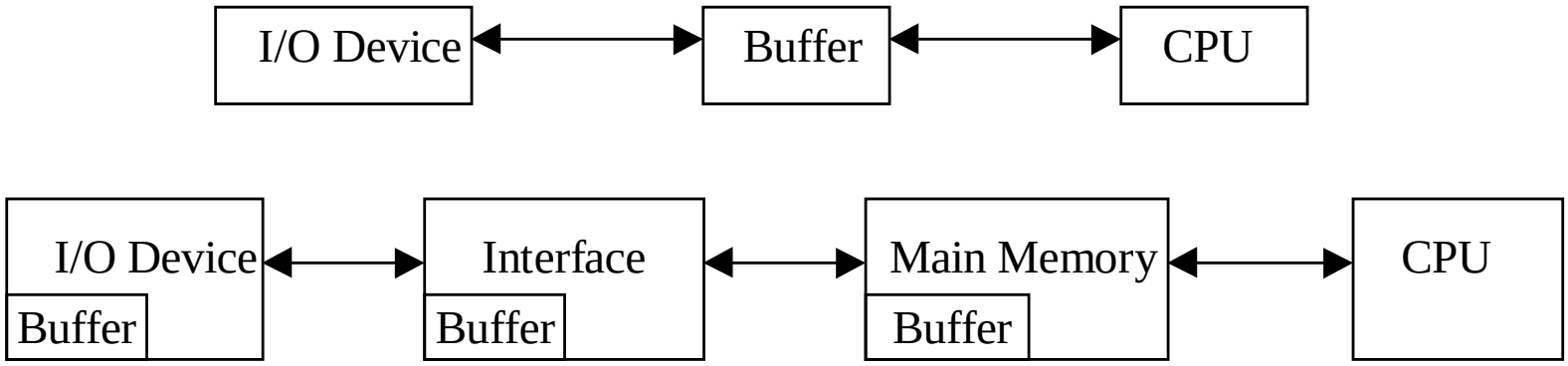
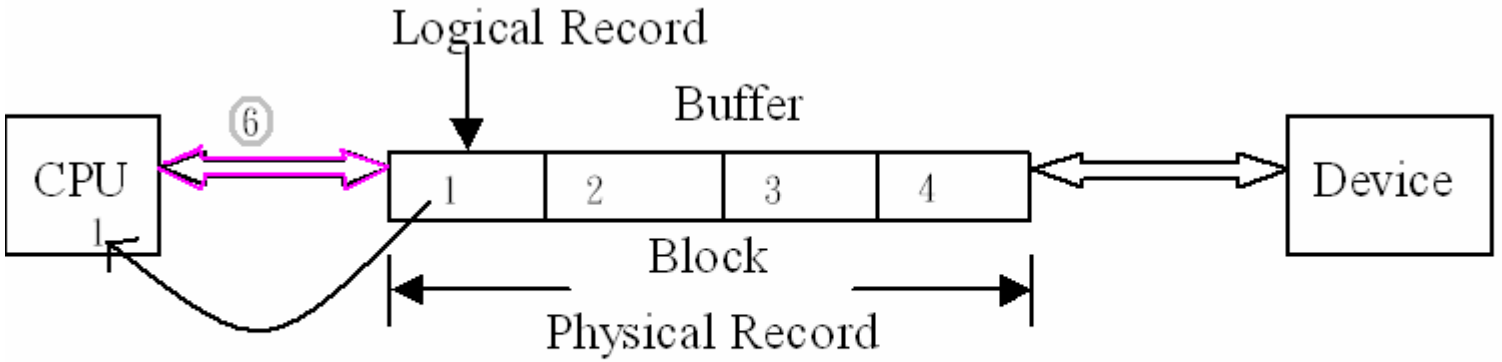


圖 1.21　輸出 / 輸入設備、中央處理器、緩衝區的關係圖

緩衝區(Buffer)(2)

- 輸出 / 輸入設備一次存取的單位爲實體記錄（Physical Record）。
- 從使用者觀點，將所收集的一筆資料存入檔案內，就是指邏輯記錄。
- Blocking 將數個邏輯記錄組成實體記錄。

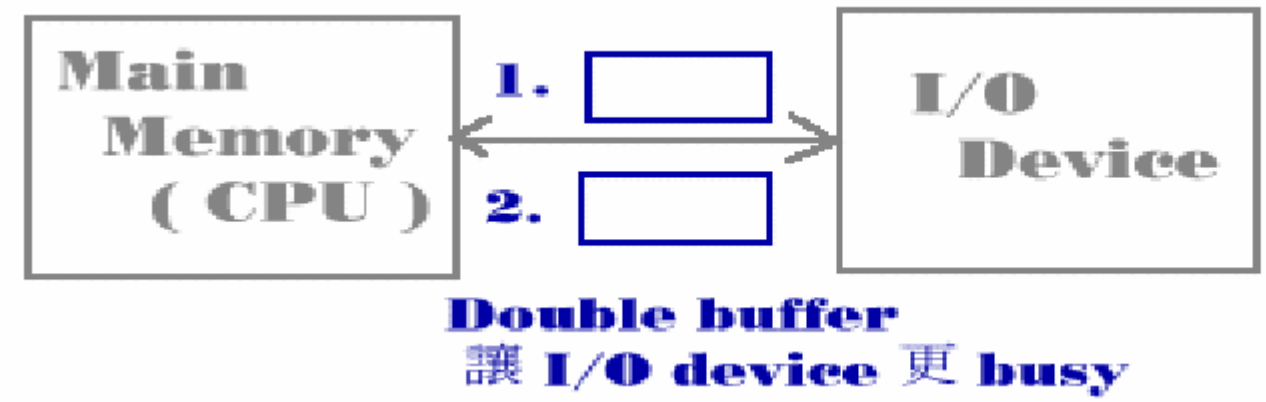


1 Physical Record = 1 Block = Buffer = 1 Sector = N Logical Records

圖 9.5　邏輯記錄與實體記錄

Double Buffer

- I/O device將資料擺入buffer1後通知CPU去拿。
- CPU在拿時I/O device將資料擺入buffer2



Storage Structure

2.4 Storage Structure

- Von Neumann Architecture
  - Stored Machine

- Main Memory
  - 揮發性（Volatility）。主記憶體、快取記憶體及暫存器爲揮發性，只要不供電，其內部的資料就消失了。

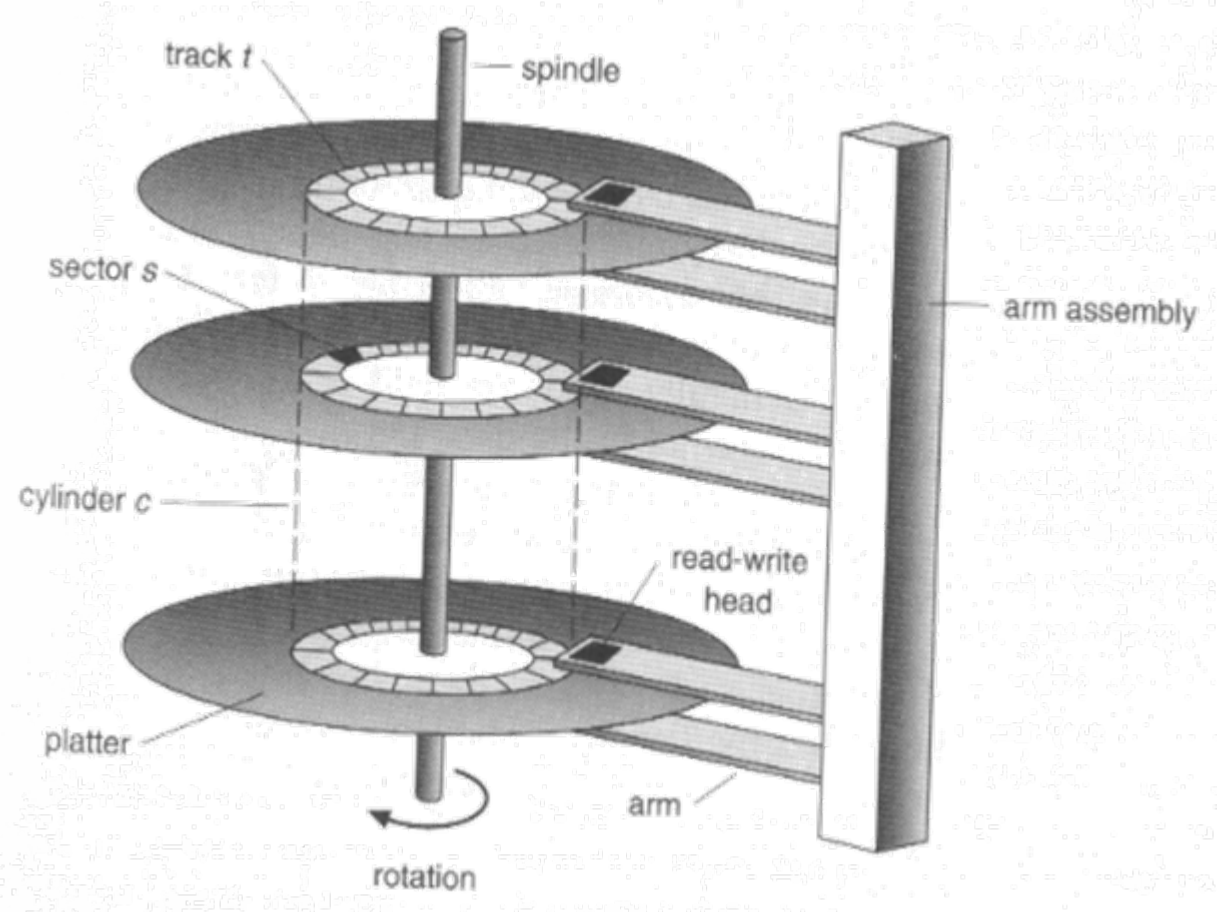
- Secondary Memory
  - 非揮發性（Non - Volatility）。亦即電源關掉，內部資料不會消失。

Main Memory

- Memory mapped I/O
  - Memory address are mapped to the device registers(RS232, display)
- Device mapped I/O
  - Using special I/O instructions to control I/O devices.
- IBM PC 兩種方式均提供。

Secondary Memory

- Disk --> Random access
- Tape --> Sequential access



階層式記憶體(Hierarchical Storage)(1)

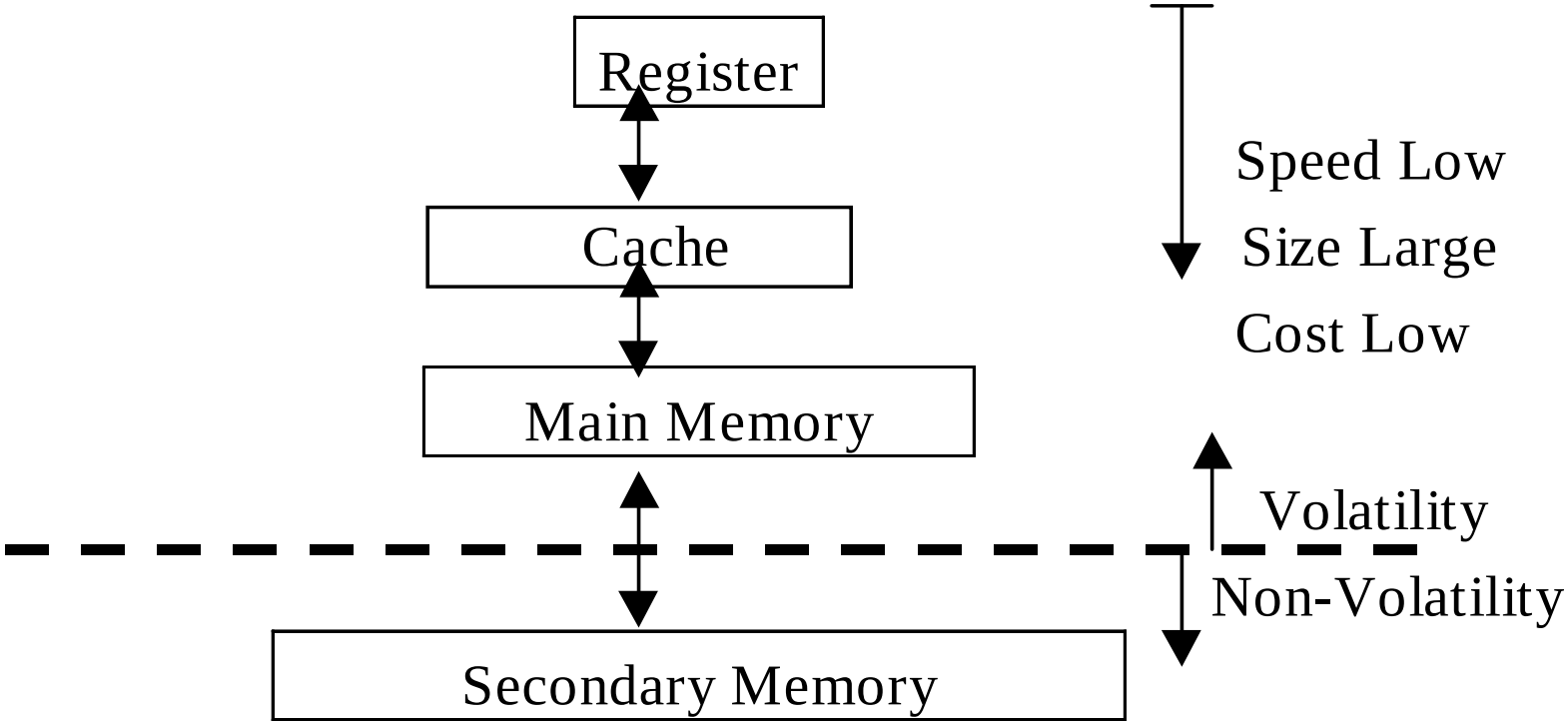


圖 1.20　階層式記憶體結構

階層式記憶體(Hierarchical Storage)(2)

- 任何必須執行的程式與資料，都必須由輔助記憶體搬至主記憶體內，而且最後要送至中央處理器內的暫存器執行。

- 輔助記憶體、主記憶體、快取記憶體、暫存器的組合架構，是一種階層式記憶體（Hierarchical Storage）。

階層式記憶體的特性

- 存取速度爲輔助記憶體速度最慢，暫存器速度最快。
- 每單位的價格爲輔助記憶體最低，暫存器最高。
- 電腦內的儲存量爲輔助記憶體最大，暫存器最少。
- 若作業系統能夠巧妙的將欲存取的資料安排在快取記憶體內，則相對有效率多了！
- 若選取適度容量的快取記憶體，並配合作業系統置換政策（Replacement Policy），大約可以達到80%~99%的命中率（Hit Ratio）。

地址空間(Address Space)

- 一個程式所能合法使用之記憶體範圍。
- The set of all address available to program.

系統保護（System Protection）(1)

2.5 Hardware Protection

- 系統保護之目的在防止非經授權者使用系統資源，同時確保不正確的程式不會因此造成其他程式執行不正確。
- To ensure that incorrect program can't cause other programs to execute incorrectly.

- 雙模保護（Dual Mode Protection）
  - 使用者模式（User Mode）。
  - 監督者模式（Monitor Mode、Supervisor Mode、System Mode 、 Privileged Mode 、 Protect Mode ）。

系統保護（System Protection）(2)

- 作業系統通常在監督者模式下執行，而在監督者模式內所使用之命令爲特權指令（Privileged Instruction），例如
  - 暫停系統（Halt System）
  - 打開 / 關閉中斷（Turn Interrupt On /Off）
  - 轉換模式（Change Mode）
  - 所有的輸出/輸入指令...等均爲特權指令

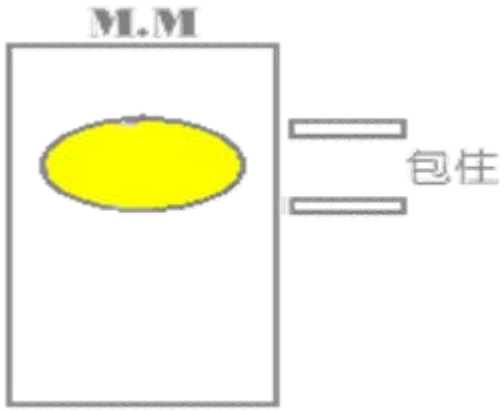
輸出 / 輸入保護（I / O Protection ）

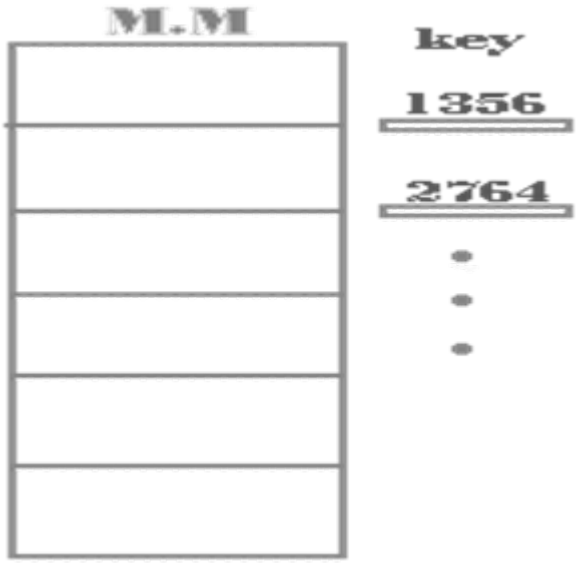
- 通常我們透過系統呼叫來進行輸出 / 輸入工作，它用來保護系統資源及週邊設備 。

記憶體保護（Memory Protection）

- 當載入器將執行模組載入主記憶體定位後，程式的地址空間便被確定；若程式於執行過程中，要存取非地址空間的位置，除非透過系統呼叫建立連繫，否則被視爲不合法的存取記憶體，這就是記憶體保護。

- Bounded register
- Memory protection keys





中央處理器保護（CPU Protection）(1)

- 若一個程式持續執行迴圈而永不停止，將造成浪費中央處理器資源的問題；不幸的是，在學理上我們無法確定任何一個程式是否永不停止的執行，這就是有名的停止問題（Halting Problem）。

- 爲了保護中央處理器，我們可以在作業系統中爲每個程式設定計時器（Timer），當時間逾時（Time Out）則會產生中斷（Interrupt），以停止程式繼續執行。

中央處理器保護（CPU Protection）(2)

- 在一個分時系統（Time　Sharing）中，每個使用者均給予固定時間片段（Time Slice）佔有中央處理器，當時間片段用罄，則將中央處理器讓給下一位使用者。

- Context Switch

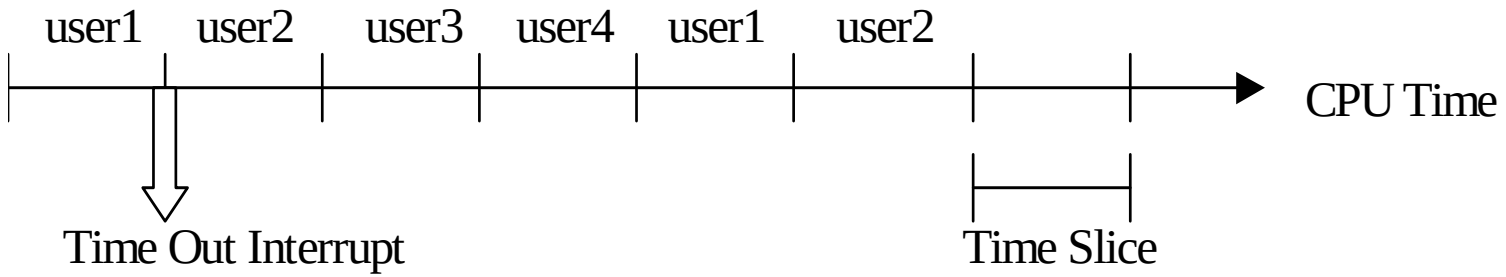


圖 1.14　分時系統

檔案保護（File Protection）

- 使用者可以合法使用自己目錄（Directory）內的檔案，但其他電腦使用者必須經過授權後方可使用別人的檔案，這便是對檔案存取能力（Access Capability）之保護。