4-5 使用者執行緒及核心執行緒

使用者執行緒及核心執行緒

- 學習完本單元，您將可以：
  - 了解使用者執行緒及核心執行緒不同處

使用者執行緒及核心執行緒

- 使用者執行緒（User Thread）
  - 使用者應用程式所撰寫的執行緒。

- 核心執行緒（Kernel Thread)
  - 作業系統核心使用的執行緒。

使用者執行緒及核心執行緒比較

- 環境切換(Context Switch)
- 排程(CPU Scheduling)
- 懸置情況(Block Status)

環境切換

- **使用者執行緒**
  - 執行緒排程器（Thread Scheduler）自動嵌入使用者執行緒的程式內，並自行於處理程序內對執行緒進行排程工作，因此在執行緒進行環境切換過程中，電腦內不必產生中斷去通知作業系統核心，也不必作處理程序環境切換的工作。

- **核心執行緒**
  - 使用者必須透過系統呼叫使用核心執行緒，必須經由中斷執行核心執行緒，必須花費中斷的代價，但共用地址空間的優點仍然存在。

排程

- **使用者執行緒**
  - 中央處理器排程器以處理程序為單位進行排程及環境切換工作，當某一個處理程序輪到中央處理器時，它將所輪到的時間片段分享給依附在處理程序內的執行緒使用。例如某個處理程序分配到時間片段為200ms，若平均分配給五個執行緒，則執行緒排程器分配40ms給每個執行緒執行。

- **核心執行緒**
  - 中央處理器排程器以核心執行緒為單位分配時間片段。例如若某個核心處理程序有三個核心執行緒，則此處理程序分到三個時間片段，也就是每個執行緒各有一個時間片段。

懸置情況

- **使用者執行緒**
  - 若某個處理程序內有一個執行緒進入懸置狀態，則整個處理程序內的所有執行緒均一起懸置。這是因為執行緒排程器自行對執行緒排程，從作業系統角度來看，會認定該處理程序懸置，以致造成該處理程序內所有執行緒均被懸置。

- **核心執行緒**
  - 由於每個核心執行緒是各自排程，所以核心處理程序內若某一個執行緒懸置，它不會造成其他的執行緒被懸置。

主要的使用者執行緒程式庫(User Thread Libraries)

- 使用者應用程式所撰寫的執行緒。

- 主要的使用者執行緒程式庫:
  - POSIX Pthreads
  - Win32 threads
  - Java threads

核心執行緒(Kernel Threads)

- 作業系統核心使用的執行緒。

- 作業系統
  - Windows XP/2000
  - Solaris
  - Linux
  - Tru64 UNIX
  - Mac OS X