LECTURE 10. 多執行緒

講師: 李根逸 (Ken-Yi Lee), E-mail: feis.tw@gmail.com

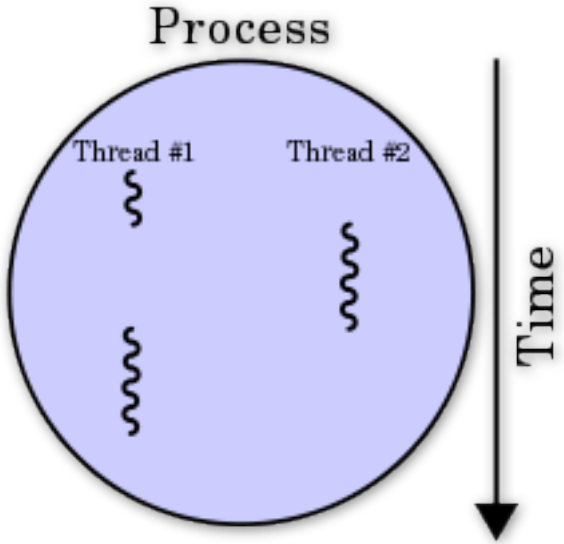
★ ★ ★ ★

★ ★ ★ ★

程式、程序與執行緒

- **程式 (program)**、行程 **(process)** 與執行緒 **(thread)** 的關係：
  - 程式指的是執行檔本身
  - 程序指的是一個載入記憶體後可執行的程式
  - 執行緒指的是一個程式中一段獨立的執行序列

不同執行緒間會共享某些同程序的資源，例如全域變數



多執行緒與平行處理

- 執行緒是處理器的可獨立運算單位，因此我們在多核心的處理器上可以做到同時使用不同的核心去執行不同的執行緒。
- 在不同的作業系統或編譯器裡，可能需要使用不同的函式庫來產生與執行多個執行緒
  - Windows 裡的 Win32 API
  - Unix-like 或 Mac 系統裡的 pthread
  - C++11 裡的 std::thread

Windows 的 CRT 函式庫

- **CRT (C Runtime Library) 是 Windows 作業系統提供的執行期函式庫**
- **<process.h> 內的 _beginthread**
  - http://goo.gl/DLDRR

```
uintptr_t _beginthread(
    void( __cdecl *start_address )( void * ),
    unsigned stack_size,
    void *arglist
);
```

[範例] helloworld.cpp

[範例] data.cpp

[範例] order.cpp

在沒有泛型的 C 裡我們使用 void 指標傳遞參數

程序結束強制終止執行緒

【練習】小時鐘

- 試著寫一個小時鐘，可以使用鍵盤的 '+' 與 '-' 來增加或減少時間

[練習] timer.cpp

【範例】將陣列的值增加

- 試寫一程式利用多平行緒同時增加陣列的值

[範例] inc.cpp

執行緒間的同步

- 要讓執行緒之間互相同步比較常見的方式是利用事件 **(event)** 的傳遞：
  - CreateEvent(): 產生事件
    - http://goo.gl/0ptGv
  - SetEvent(): 啟動事件
  - WaitForMultipleObjects(): 等待多個事件被啟動

[範例] sync.cpp

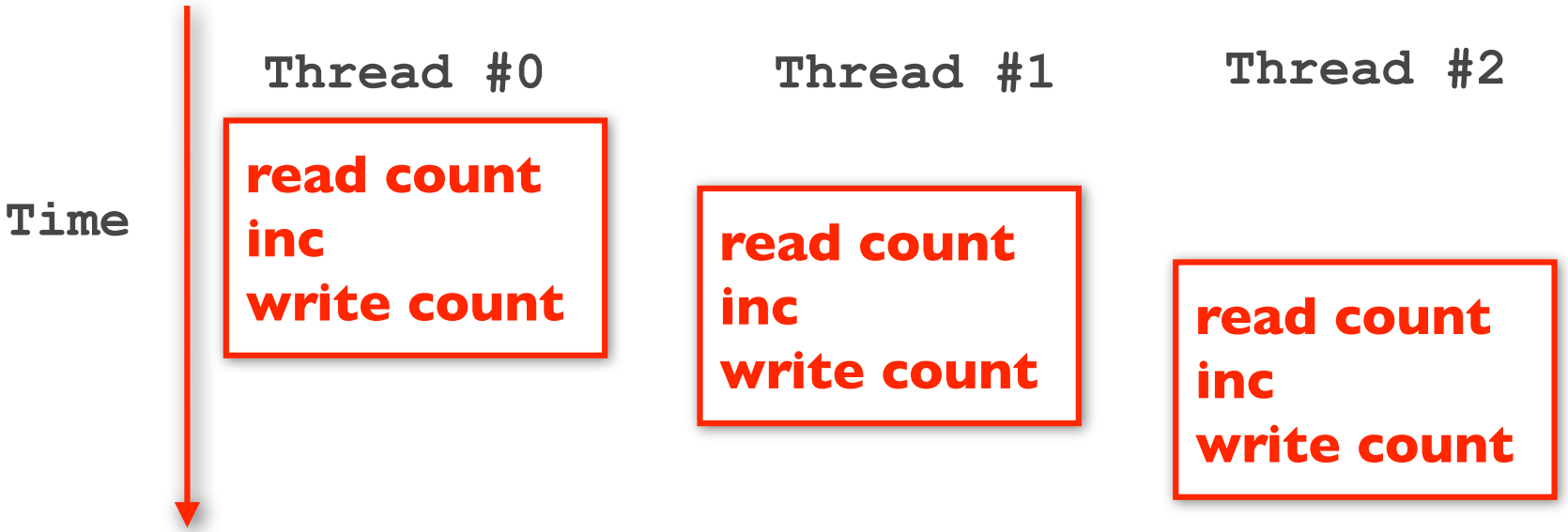
競爭危害的避免

- 競爭危害 **(race condition)** 通常發生在多個不同執行緒修改相同資料的狀態可能會因為執行順序的不同造成資料最後狀態未知的情況：

```
count++ => count = count + 1;
           (write) (read) (inc)
```

count

(int) 0



設定臨界區段

- 一個臨界區段同時只能有一個執行緒進入 **(enter)** 。其他執行緒會在要進入的地方等待直到已經進入臨界區段的執行緒離開 **(leave)**
  - InitializeCriticalSection(): 初始化臨界區段
  - EnterCriticalSection(): 申請進入臨界區段
  - LeaveCriticalSection(): 離開臨界區段

[範例] critical_section.cpp

【練習】計數器

- 讓計數器的結果正確

[練習] count.cpp

- 最多可以有多少個執行緒？