作業系統導論 LAB 1

系所：通訊三 學號 :409430030 姓名:翁佳煌

**<實驗工具>**

1.FreeRTOS即時作業系統。

2.JRE執行Eclipse的必備套件。

3.MinGW，Windows版的gcc。

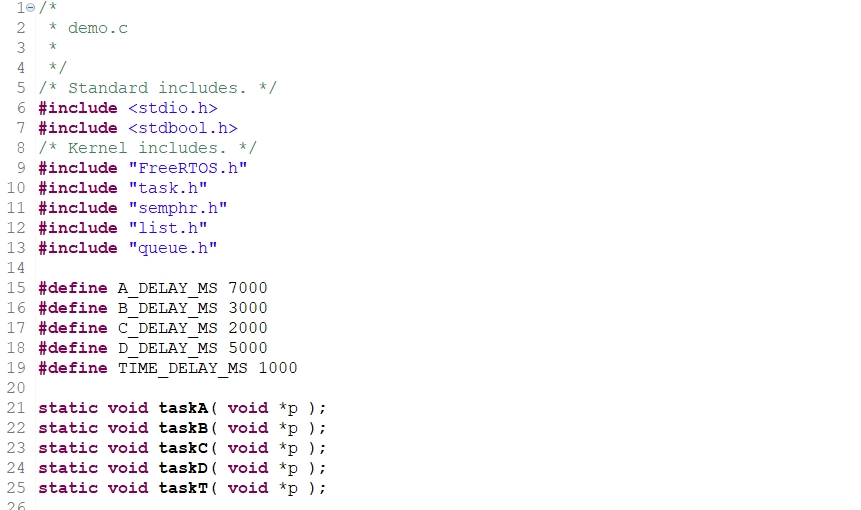
4.Eclipse開發套件。

**<實驗過程與方法>**

第6~13行:include C語言以及FreeRTOS的一些標頭文件。

第15~19行:定義題目所要求的延遲，A七秒喊一次、B三秒喊一次、C兩秒喊一次、D五秒喊一次，在這裡的單位為ms。

第21~25行:是宣告等等用來建立不同優先級的 Task，共建立了五個 Task，分別為 taskD、taskC、taskB、taskA 和 taskT。這些 Task 會在 FreeRTOS 的排程中進行排列，由高優先級的 Task 先執行，低優先級的 Task 則會等待高優先級的 Task 執行完後再執行。



▲圖1.1

接下來第27~67行:在這裡分別定義了五個 Task 的函式實作，分別是 taskA、taskB、taskC、taskD 和 taskT，在這裡舉第27行~35行的taskD程式碼做解釋，其餘taskA 、B、 C、 T皆雷同。第29行呼叫xTaskCreate()， 是 FreeRTOS 中建立 Task 的 API 函數，其中需要傳入下列參數：

1. Task function：Task 的實際執行內容，為一個函數指標。
2. Task name：Task 名稱，可以供 Debug 使用。
3. Stack size：Task 所需要的 Stack 大小，以 byte 為單位。它指定了在創建任務時要分配給任務的堆棧深度，以便該任務能夠執行其內部代碼所需的所有操作，包括所有被調用函數的內部變量和返回地址等。如果分配的stack空間不足，就會導致溢出，這可能會產生不可預測的行為和系統崩潰。在這次的code中，這部分的參數為configMINIMAL\_STACK\_SIZE，它是一個整數，表示以字節為單位的stack大小。

4.Parameter：Task 執行所需要的參數。

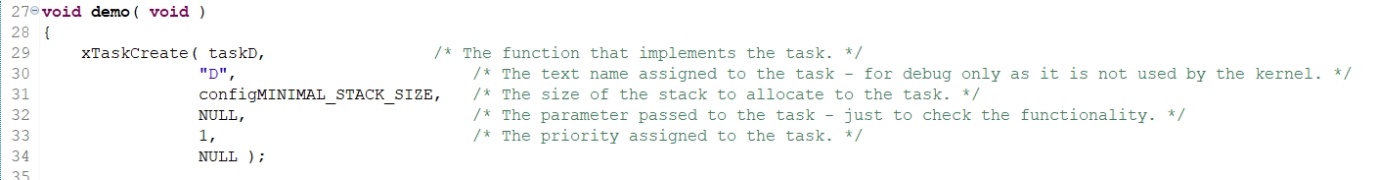
5.Priority：Task 的優先級，數值越高代表優先級越高。

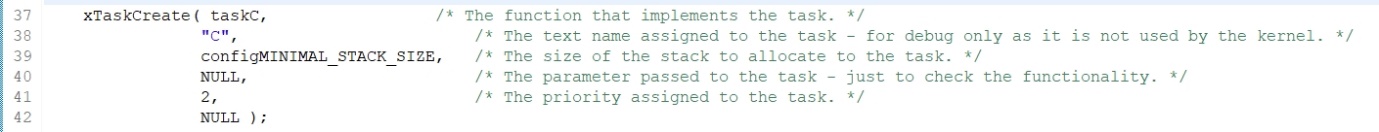
6.Handle：Task 的 handle，用來做記憶體管理和控制 Task 的執行。在本程式碼中並未使用到。

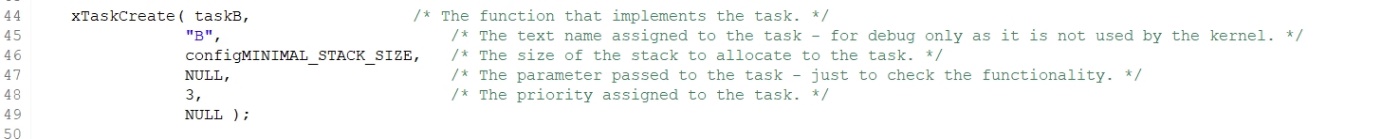
最後，在第65行呼叫vTaskStartScheduler()來啟動排程器決定讓哪個 task 開始執行。

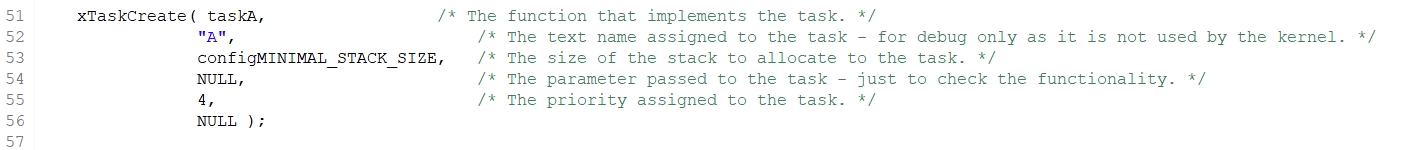
**<補充::參考資料1>**

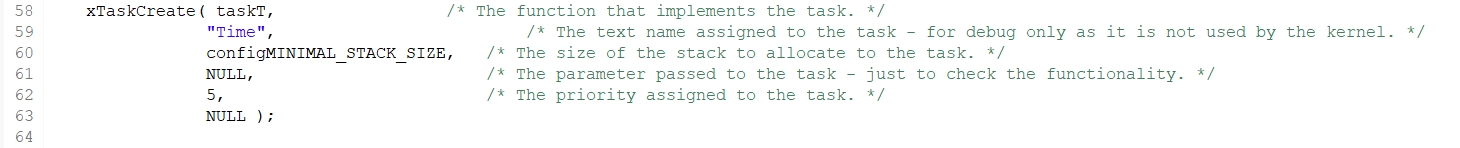
當 vTaskStartScheduler() 被呼叫時，會先建立一個 idle task，這個 task 是為了確保 CPU 在任一時間至少有一個 task 可以執行 (取代直接切換回 kernel task) 而在 vTaskStartScheduler() 被呼叫時自動建立的 user task，idle task 的 priority 為 0 (lowest)，目的是為了確保當有其他 user task 進入 ready list 時可以馬上被執行。







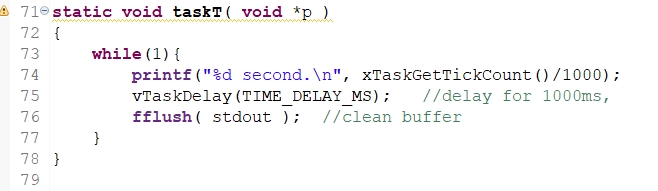


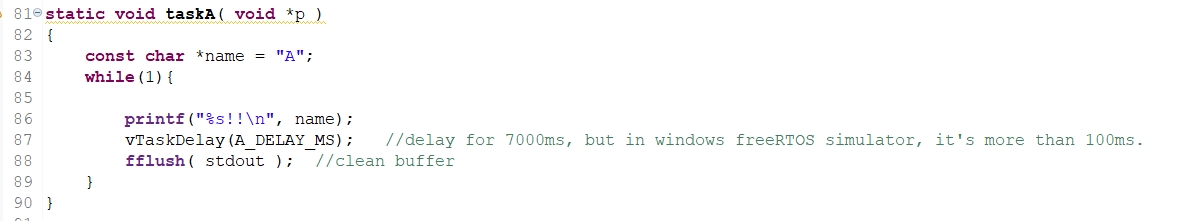


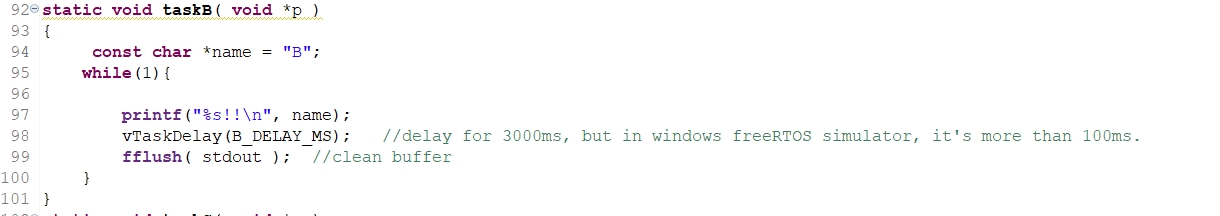


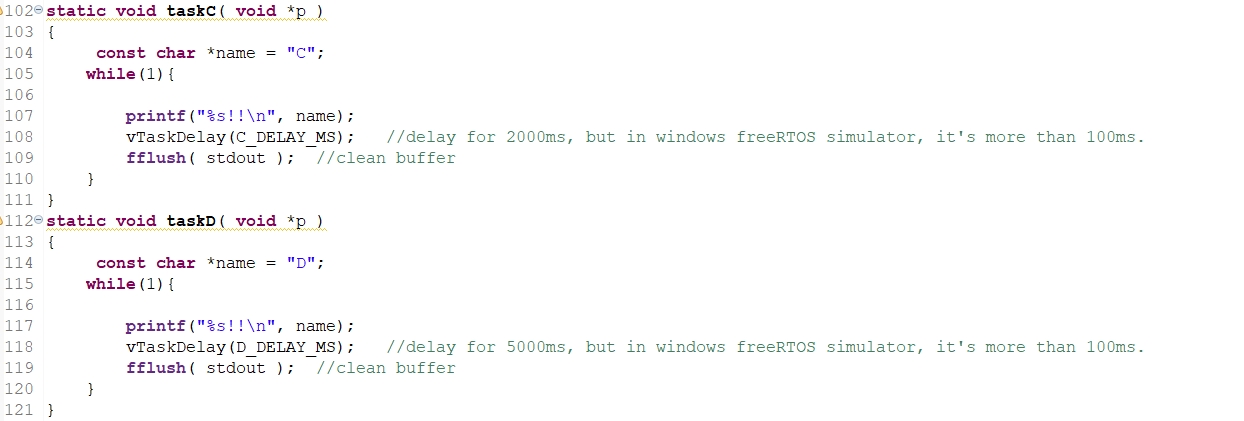
▲圖1.2

再來第71~121行:這大段程式碼也都是大同小異，我們拿第71~79行的taskT來舉例，其餘taskA 、B、 C、 D以此類推，因為taskT是用來印出每一秒的秒數，因此在第74行將他印出，第75行使用vTaskDelay()來做延遲。因此，接下來的taskA 、B、 C、 D的差異也只是印出的文字以及vTaskDelay的時間不同而已。









**<心得與收穫>**

這次是OS這堂課的第一個LAB，雖然老師說明這堂課的LAB不像微處理機是強迫你必須去做的，但我對於OS的知識實在覺得非常有趣，於是還是決定要試著做做看。有事先看了助教放在PPT給我們參考的FreeRTOS的資源，研究完後會發現這次LAB不是那麼困難(又或許是因為被微處理機摧殘過)，雖然實驗順利完成，但我發現網站上還有好多看不懂而且長相詭異的函數，希望藉由這次LAB入門後，自己能更精進OS的架構以及概念。

**<參考資料>**

1. <http://wiki.csie.ncku.edu.tw/embedded/freertos#FreeRTOS%20%E6%9E%B6%E6%A7%8B>
2. <http://www.aosabook.org/en/freertos.html>