組別：6，組員：王柏堯、張彧豪

微處理機 Lab5 Report

一、 實驗名稱：7-Seg LED

二、 實驗目的：

- 了解 MAX7219 使用原理。
- 設計 7-Seg LED 程式。

三、 實驗步驟：

1. Without code B decode mode。
2. Use code B decode mode。
3. 使用 7-seg LED 顯示 Fibonacci 數。

四、 實驗結果與分析：

1. Without code B decode mode：

第一題比第二題稍微難一點，主要在於這題沒有預設的 encode，必須手動把 0~A 對應的編碼輸到 arr 這個變數裡，因為這塊板子是 little-endian 的關係，所以由低位址往高位址一次讀一個 byte 所讀到的數值就是 0~A。詳細做法是每次讀一個 byte，接著把計數器加 1，將讀到的 byte 送入 MAX7219Send 這個程式片段做輸出在七段顯示器上的動作，再來 delay 一秒，接著迭代做一樣的事情，直到計數器大於 15，再將計數器歸零後重新開始動作。

2. Use code B decode mode：

這題跟第一題做法差不多，只是這題可以使用 MAX7219 內建的 encode 編碼方式，所以只要將數字輸進 student_id 這個變數就可以了，因為 MAX7219 對於數字的

編碼就是數字本身的二進位表示法，這題也是採用版子就是 little-endian 的關係，一次讀一個 byte，並把讀到的 byte 送入 MAX7219Send 依照它所在的 digit 輸到七段顯示器上，這樣的做法可以讓程式變成一個簡單的迴圈，簡化程式的複雜度。

3. 使用 7-seg LED 顯示 Fibonacci 數：

這題是這次實驗中最難的一題，這題我們可以分成幾個部分來說明：

- 計算 fib 的值：

  這部分算是最簡單的，要這題裡面需要記住當下的 fib 值與前一次的 fib 值，要計算新值的時候就將當下與前一次的值相加即可，並更新前一次的 fib 值以供下一次運算。

- Debounce 相關問題處理：

  這題的 Debounce 跟第四次實驗的最後一題差不多，都是讓 CPU 一直 debounce，直到發生某些事件才去做對應的運算。不一樣的地方在於有限狀態機的狀態數量不同，這題會需要再多兩個狀態，一共五個狀態，多的兩個狀態是用來區分到底是按一下後馬上放開還是按了超過一秒才放開，所以在 debounce 內部也要加入計時的功能，讓有限狀態機可以知道何時已經超過一秒了並做出相對應的狀態變化。

- 輸出處理：

  前兩題的輸出都跟這題不太一樣，這題因為要輸出的東西不是一個一個平均分配在記憶體中，而是一個不只 1 位的整數的每一位數值，所以不能再用前兩題的方法直接去得到要輸在七段顯示器上某位的值，這也是我們一開始卡住的地方，本來想找有沒有模數運算，最後發現此版本的組合語言並不

支援，只好改用除法以及 MLS(乘法和減法)來完成模數運算。

在輸出前必須檢查輸出的值是不是超過一億，如超過一億就必須強制輸出成-1，這個檢查必須寫在進入輸出迴圈之前就要做，這樣才能避免輸出錯誤。

確認輸出資料正確後，把輸出這整個動作當成迴圈，迴圈內做的事情就是把資料輸出在七段顯示器上某一位，要拿到輸出資料的方法就是利用除法取餘數，餘數就是我們要的輸出資料，迭代做下去直到商數和餘數都為 0，表示所有的輸出資料都輸出了，跳出迴圈，重新進入 debounce 抓取按鈕訊號。

五、 心得討論與應用聯想：

這次 LAB 很難，我們一直打不開 MAX7219，也很難測出錯誤在哪，所幸在一行行檢查下找出錯誤，實際上若輸出已經做完了，剩下的顯示並沒有大問題，畢竟就是一個字一個字丟入，但因為沒有模除，所以還要自己手刻，非常的累，至於第三題的 FIB，我們結合了之前的方式，並讓大於指定範圍的數值固定，才不會因為不斷加而 overflow，但令我們意外的是，按了八下的-1 後，居然會使顯示器錯誤，在仔細檢查下，才發現是 stack 的 PUSH 沒有對應的 POP，找到了這錯誤後，終於結束了這次 LAB。