微處理機第六次實驗報告

組員: 0416020 巫垣佑、0416074 徐福臨

1.實驗名稱：STM32 Keypad Scanning

2.實驗目的：

- 了解如何使用 C code 控制 STM32
- 設計 7-seg LED 和 keypad 程式

3.實驗步驟

3.1：將上個 LAB 完成的 GPIO_init()、MAX7219_send()改成可以被 C 呼叫的版本，於 C file 完成 display function，並利用 MAX7219_send()顯示學號於 7-seg 上。

在 .s 檔中，把 GPIO_init()、MAX7219_send()等會在 C file 用到的 function 加上 .global，C file 這邊 extern 進來後，注意參數傳遞的方式，就可以從 c file 使用了。

3.2 Keypad scanning：分別四個 input、output GPIO pin 連接 keypad，按下 keypad 上的按鍵後，顯示於 7-seg 上。

output：PA8、PA9、PA10、PA12。

input：PB5、PB6、PB7、PB9。

c file 端 include stm321476xx.h，標頭檔須放在 src 裡面。

做法：

debouncing：

每次去抓 GPIOB->IDR，有值表示可能有按按鍵，連續抓 45000 次後，GPIOB->IDR 若還有輸入，表示真的有 input。

button scanning：

每次將一個 column 對應 pin 的值 set high，掃過每個 row，GPIOB->IDR 與對應位置的值做 AND 運算，若不為 0，表示第 i 個 column 第 j row 的按鈕被按下了。

找到按下的按鈕後，用 MAX7219_send()將結果顯示在 7-seg 上。

3.3 多按鍵問題：

紀錄每個位置的狀態，有按下或沒按下。

只有當某個按鍵是從沒按下到按下，result 加上該位置對應的數字(若大於 999999999 則不動作)，並顯示。

若沒有按下則更新狀態為沒按下。

3.4 Bonus 簡易計算機

先乘除後加減，遇到乘除都先算完，之後遇到加或減，則將前面的結果先運算完。

用到的變數有：before、calculating、tmp。calulating 預想是正在做成除的運算；before 是前面算完的結果；tmp 是目前讀到的數字。

用到的運算子有：bef_op、cal_op、now_op。bef_op 是 before 後接著的運算子，過程中只會是加或減；cal_op 是 calculating 與 tmp 的運算子，運算過程中是乘或除；now_op 是目前讀到的運算子。

由先前提要可以將目前的運算式子寫成：

before bef_op (calculating cal_op tmp)

處理連續按運算子：

如果是讀到運算子，now_op 會更新，在讀到數字後，才能確定剛剛的運算子是甚麼，同時進行算式的更新。

算式的更新只有在運算子後遇到第一個數字才做。now_op 為加或減，可以想像目前運算式長這樣：before (+ | -) calculating (*|/) tmp (+|-)，這時就可以把這段算完後，結果放進 before，bef_op 更新；若 now_op 為乘或除，則是長 before (+|-) calculating (*|/) tmp (*|/)，calculating 和 tmp 可以先算，更新 calculating、cal_op。

等號出現後，一樣照運算式算完即可。

4.實驗結果：

demo 時即可展現。

5.心得：

這次的 LAB 轉到 C 上實作了，寫起來也方便許多，還有前人寫好 STM32 的標頭檔可以使用，不用自己定義，令人見識到開源的好處。