實驗六 STM32 Keypad Scanning

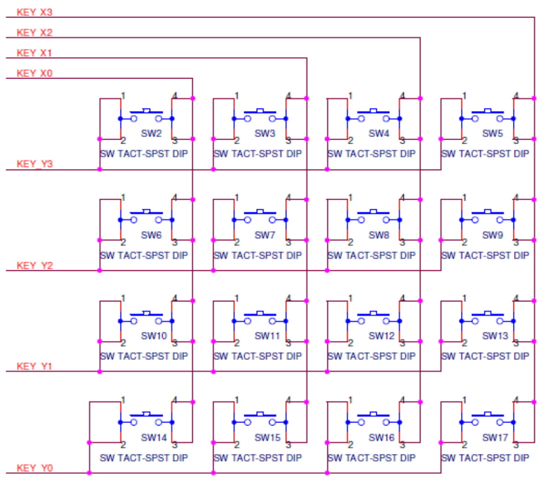
1. 實驗目的

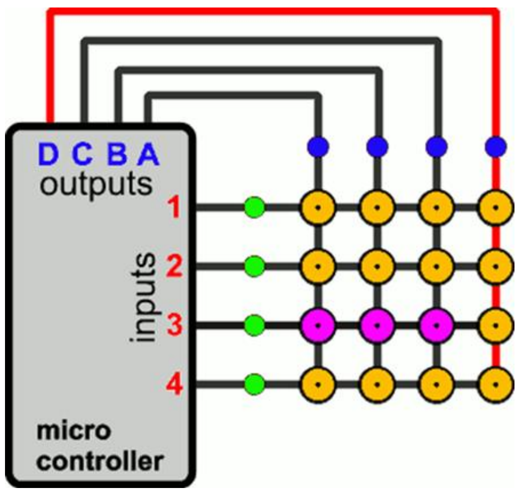
- 了解 STM32 使用原理
- 了解如何使用 C code 控制 STM32
- 設計 7-Seg LED 和 keypad 程式

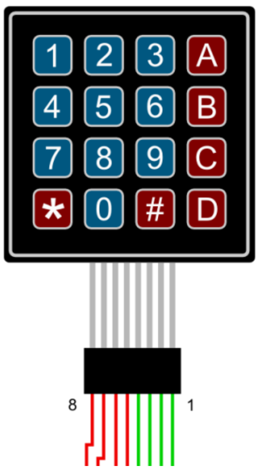
2. 實驗原理

Keypad 電路組成如下，主要是一個 4x4 的鍵盤按鈕所組成會用到 4 個 Input pin 與 4 個 Output pin，其控制原理是利用 Output pin 掃描的方式來決定目前所選擇到的是哪一行按鍵，例如當 KEY X0~3 輸出 1000 而此時若 KEY Y0~3 所讀到的值是 1000 的話則代表 SW14 按鈕被按下。

The circuit diagram of keypad is given below. You're supposed to use 4 input pins and 4 output pins. Use output pins to determine which row you're scanning. For example, when output value of KEY X0~3 is 1000 and input value of KEY Y0~3 is 1000, then we can say that SW14 is pressed.







KEYPAD PINOUT:

PIN 1: COL 4
PIN 2: COL 3
PIN 3: COL 2
PIN 4: COL 1
PIN 5: ROW 4
PIN 6: ROW 3
PIN 7: ROW 2
PIN 8: ROW 1

3. 實驗步驟

3.1. Lab 6.0: Max7219 displayer (10%)

將 Lab5 所完成的 GPIO_init()與 MAX7219_send()改成可以被 C 所呼叫的版本，並新增一個 C file 完成 display function 及利用 max7219_send()將學號顯示於 7 段顯示器上。

Modify your code in lab5.2 to make it callable by C. Add a C file to complete the code given below, display your student ID on 7-Seg LED.

```
//These functions inside the asm file
extern void GPIO_init();
extern void max7219_init();
extern void max7219_send(unsigned char address, unsigned char data);

/**
* TODO: Show data on 7-seg via max7219_send
* Input:
*     data: decimal value
*     num_digs: number of digits will show on 7-seg
* Return:
*     0: success
*     -1: illegal data range(out of 8 digits range)
*/
int display(int data, int num_digs)
{

}
void main()
{
   int student_id = 01234567;
   GPIO_init();
   max7219_init();
   display(student_id, 8);
}
```

3.2. Lab6.1: Keypad Scanning

利用 4 個 input GPIO 與 4 個 output GPIO pin 連接 keypad，當按下 keypad 利用兩顆七段顯示器顯示所對應的數字。

Note: keypad 所使用到的 GPIO 請利用 C 語言的方式初始化，各 GPIO register address 與 structure define 請參考 stm32l476xx.h

Use 4 input GPIO pins and 4 output GPIO pins to connect with keypad. Show the corresponding number of pressed button on 7-Seg LED.

Note: Use C to init GPIO used by keypad. Please refer to stm32l476xx.h for GPIO register address and structure define.

```
#include “stm32l476xx.h”
//TODO: define your gpio pin
#define X0
#define X1
#define X2
#define X3
#define Y0
#define Y1
#define Y2
#define Y3

unsigned int x_pin[4] = {X0, X1, X2, X3};
unsigned int y_pin[4] = {Y0, Y1, Y2, Y3};

/* TODO: initial keypad gpio pin, X as output and Y as input
*/
void keypad_init()
{
}

/* TODO: scan keypad value
* return:
*  >=0: key pressed value
*  -1: no key press
*/
char keypad_scan()
{
}
```

各按鍵對應值為：

	X0	X1	X2	X3
Y0	1	2	3	10
Y1	4	5	6	11
Y2	7	8	9	12
Y3	15	0	14	13

3.3. Lab6.2 處理單或多按鍵 (30%)

利用 keypad 輸入數字並在七段顯示器顯示，各按鍵對應值為：

Show pressed button of keypad on 7-Seg LED. Each value of corresponding button is given below.

	X0	X1	X2	X3
Y0	1	2	3	10
Y1	4	5	6	11
Y2	7	8	9	12
Y3	C	0	C	13

當按多按鍵時，會將按鍵值相加並顯示出來(按 1、5、9 則顯示 15)，若準備顯示的值>99999999，則不更動原本七段顯示器上顯示的數字，直到按下消除鍵(C)。

When multiple buttons are pressed, show the sum of values that buttons pressed representing. If shown value is greater than 99999999, don’t modify the number showing on 7-Seg LeD until button C is pressed.

3.4. BONUS Lab6.3 設計簡易計算機

寫出一個可先乘除後加減的計算機。
輸入數值時，最多三位數字，數值範圍 1~999，若多於三位，則再輸入數字時沒反應（原本 111，再多按一個數字，7-SEG LED 依舊顯示 111 不會改變)；
當按下運算子(+ - * / =)時，會將原先顯示在 7-SEG LED 的數字消除掉，等待數字輸入；
當連續按運算子 (ex:100 - - 9)答案依舊需正確算出；
當輸入完數字和運算子按下等於後，顯示答案（7-SEG LED 答案可顯示超過三位數和負數）；
按下消除鍵後才開始新的運算（消除鍵無論何時按下皆會消除顯示數字，並重新開始運算）
範例影片如下：
https://goo.gl/rn8srq

Design a calculator first doing multiplication and division then do addition and subtraction. Requirements are given below.

Input value should be in the range of 1~999. If input value is already 3 digits, don’t give any responds to button pressed after that.

When operator is pressed, clear the number shown on 7-Seg LED and wait for next number input.

If operator is pressed more than 1 time, answer output should be right though.

After “equal” is pressed, show the answer(negative number and number greater than 999 should be shown).

Example video link is given above.

各按鍵對應值為：

Each value of corresponding button is given below.

	X0	X1	X2	X3
Y0	1	2	3	+
Y1	4	5	6	-
Y2	7	8	9	*
Y3	=	0	C	/