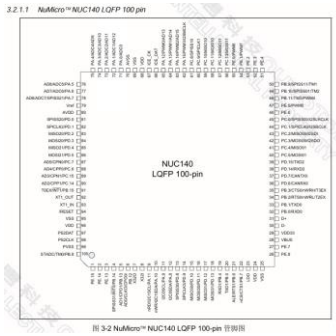
微處理機系統與介面技術 LAB 1

系所：電機 學號 :612415013 姓名：蕭宥羽

<實驗器材>

NUC 140 V2.0 開發板



<實驗過程與方法>

要求四個七段顯示器分別顯示 5013，並在 9 宮格按鍵按下後在最右邊顯示按下的數字，並將另外三個七段顯示器關閉

有一支主程式(main.c)跟兩支副程式(Scankey.c and Seven_Segment.c)，Scankey.c 負責處理按鍵的偵測 Seven_Segment.c 則是負責處理四顆七段顯示器的顯示

Block	Pin	Function
7-Seg LED	GPE0~7	Row
	GPC4~7	Column
Key Matrix	GPA0~5	GPIO

這邊是 GPIO 對應開發版上七段顯示器及按鍵的 pin

七段顯示器 ：PE0-7 決定顯示器要亮哪一根 PC4-7 決定要亮哪一顆顯示器，七段顯示器不能同時顯示，所以會用肉眼無法辨別的掃描方式去顯示。

按鍵 ：3*3 的按鍵，用 3+3 個 GPIO 去控制(PA0-5)，用掃描的方式去判斷哪個按鍵被按下。

31	30	29	28	27	26	25	24
PMD15		PMD14		PMD13		PMD12	
23	22	21	20	19	18	17	16
PMD11		PMD10		PMD9		PMD8	
15	14	13	12	11	10	9	8
PMD7		PMD6		PMD5		PMD4	
7	6	5	4	3	2	1	0
PMD3		PMD2		PMD1		PMD0	

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
DOUT[15:8]							
7	6	5	4	3	2	1	0
DOUT[7:0]							

Bits	描述	
[2n+1:2n]	PMDn	GPIOx I/O Pin[n] 模式控制 决定 GPIOx 管脚的 I/O 类型 00 = GPIO port [n] 管脚为输入模式 01 = GPIO port [n] 管脚为输出模式 10 = GPIO port [n] 管脚为开漏模式 11 = GPIO port [n] 管脚为准双向模式

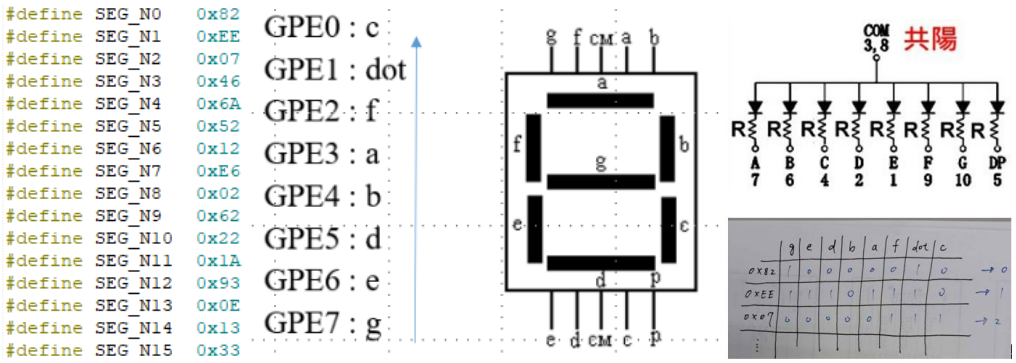
31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
PIN[15:8]							
7	6	5	4	3	2	1	0
PIN[7:0]							

PMD 暫存器可以設定 pin 的模式(4 種模式 所以會用到 32bits)

DOUT and PIN 輸出及輸入的值(只會用到前 16bits)

<Mian function code>

1. Seven_Segment.c



因為共陽極七段顯示器的 CM 端已接在高電位，必須將對應的顯示器接腳設為低電位。這樣才能讓電流從高電位的 CM 端流向低電位的 LED 接腳，從而點亮指定的段位。

```
void OpenSevenSegment(void)
{
    //GPIO_SetMode(PC, (BIT4 | BIT5 | BIT6 | BIT7), GPIO_PMD_OUTPUT);
    PC -> PMD = (PC -> PMD & 0xFFFF00FF)|0x5500;    // 設置 port c 4-7為output模式
    PC->DOUT &= 0xFF0F;                             // 將 PC4-PC7 的DOUT狀態設為 low其他保留原來狀態


    GPIO_SetMode(PE, (BIT0 | BIT1 | BIT2 | BIT3 | BIT4 | BIT5 | BIT6 | BIT7), GPIO_PMD_QUASI);  // 設置 port E 0-7 為 QUASI 模式
    PE->DOUT &= ~(BIT0 | BIT1 | BIT2 | BIT3 | BIT4 | BIT5 | BIT6 | BIT7);                          //將 PE0-PE7 的 DOUT狀態設為low
}
```

這邊設置 GPIO 的模式及初始狀態

設置模式用了兩種方式，用 GPIO_SetMode 跟直接填暫存器 PC->PMD，後者的這個方法需要先用 and 保留沒有要用的位元狀態，避免影響其他 GPIO，在用 or 去設置要使用的腳位狀態

```
void ShowSevenSegment(uint8_t no, uint8_t number)
{
    uint8_t seg7_scan[4] = {0x10, 0x20, 0x40, 0x80};        //對應7段顯示器
    PE->DOUT &= 0xFF00;                                     //保留PE高位元 清除PE0-7
    PE->DOUT |= SEG_BUF[number];                            //根據number參數 從SEG_BUF中讀取對應數字 設置道PE0~PE7中

    PC->DOUT &= 0xFF0F;                                     //保留PC其他位元 清除PC4-7
    PC->DOUT |= seg7_scan[no];                              //根據no參選擇哪一位顯示器顯示數字

}

//關閉所有7段顯示器 將pc4-7設為0 避免顯示干擾
void CloseSevenSegment(void)
{
    PC->DOUT &= 0xFF0F;
}
```

ShowSevenSegment : no 控制要亮哪一顆七段顯示器 number 控制要顯示什麼數字

CloseSevenSegment : 將 PC4-7 清除，避免顯示器的干擾

2. Scankey.c

```
void OpenKeyPad(void)
{
    //GPIO_SetMode(PA, (BIT0 | BIT1 | BIT2 | BIT3 | BIT4), GPIO_PMD_QUASI);
    PA -> PMD = (PA -> PMD & 0xFFFFF000)|0x0FD5;
}
```

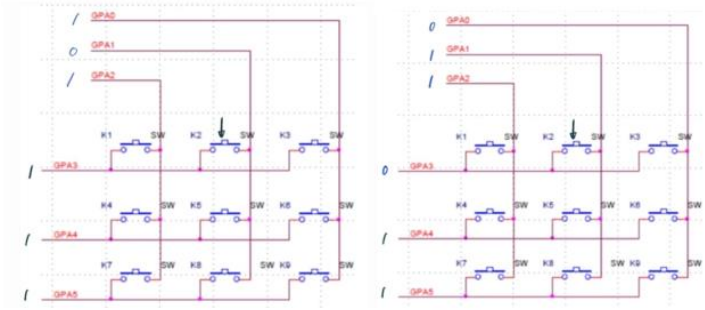
這邊設置 GPIO 的模式 PA0-2 output_mode(01) PA3-5 quasi_mode(11)

```
uint8_t ScanKey(void)
{

  PA0=1; PA1=1; PA2=0; PA3=1; PA4=1; PA5=1;
  CLK_SysTickDelay(10);
  if (PA3==0) return 1;
  if (PA4==0) return 4;
  if (PA5==0) return 7;

  PA0=1; PA1=0; PA2=1; PA3=1; PA4=1; PA5=1;
  CLK_SysTickDelay(10);
  if (PA3==0) return 2;
  if (PA4==0) return 5;
  if (PA5==0) return 8;

  PA0=0; PA1=1; PA2=1; PA3=1; PA4=1; PA5=1;
  CLK_SysTickDelay(10);
  if (PA3==0) return 3;
  if (PA4==0) return 6;
  if (PA5==0) return 9;
  return 0;
}
```



PA0-2 會一直循環其中一個設為 low，PA3-5 會設為 high，當鍵按下去看 PA3-5 哪個會示 low 就可以知道是哪個按鍵被按下去了(按下後 return 對應數字 沒按 return 0)，加上 delay 防止按鍵抖動

3. main.c

```
void Display_7seg(uint16_t value)
{
  uint8_t i;
  for(i=0;i<4;i++){
    CloseSevenSegment();
    if(value!=0 & i==0){
      id[0] = value;
      ShowSevenSegment(0,id[0]);
    }
    if(value==0){
      id[0] = 3;
      ShowSevenSegment(i,id[i]);
    }
    CLK_SysTickDelay(5000);
  }
}
```

Id[4] 放學號 5013，這邊會傳入 value，如果 value 不是 0 則 id[0]會是 vlaue 的值，並只讓第 0 顆七段顯示器亮 如果是 0 的話就顯示原本的 5013。

```
int main(void)
{
    uint16_t j;
    SYS_Init();
    OpenSevenSegment();
    OpenKeyPad();

    while(1) {
    j=ScanKey();
    Display_7seg(j);
    }
}
```

SYS_Init(); 對系統進行初始化 `#define MCU_CLOCK_FREQUENCY  50000000  //Hz`

j 接收按鍵的值，作為 value 傳入上面的函數，並一直循環。

<心得與收穫>

雖然這次的實驗看似簡單，但在實作過程中發現，若要成功完成，必須對晶片的架構和 GPIO 控制有深入的了解。每個步驟都需要細心設置，例如對七段顯示器和按鍵的 GPIO 引腳配置，以及系統初始化等，都要求具備對硬體資源的清晰認識。