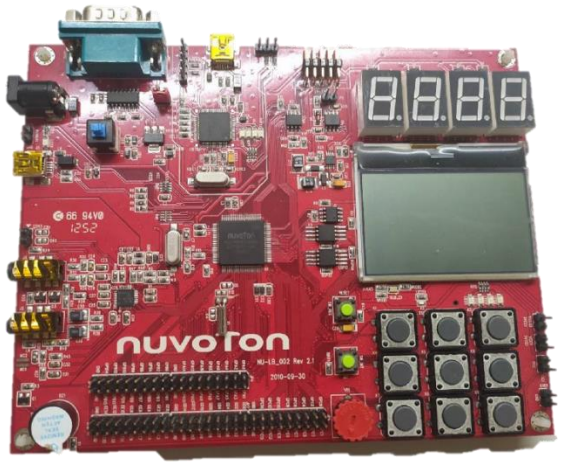
微處理機系統與介面技術 LAB 1

系所：通訊三　　學號 :409430030　　姓名:翁佳煌

<實驗器材>

NUC 140 開發板



<實驗過程與方法>

我一開始不知道可以直接拿 Seven_Segment.c 和 Scankey.c 裡面的 Function 來做使用，於是我最初是一個一個把 PC4~7 依序打開(如下圖 1.1)，並依照 PE0~7 的規定，顯示自己的學號後四碼 0030，且因為七段顯示器不能夠同時顯示，所以要以很快的速度去掃描讓它顯示，因此我設定 CLK_SysTickDelay(50)。

```
//1=on,0=off
    PC4=1; //最右邊
    PC5=0;
    PC6=0;
    PC7=0;

    //0=on,1=off
    PE0=0;
    PE1=1;
    PE2=0;
    PE3=0;
    PE4=0;
    PE5=0;
    PE6=0;
    PE7=1;
    CLK_SysTickDelay(50);
    //print(0)-----------------------//
```

```
    PC4=0;
    PC5=1;
    PC6=0;
    PC7=0;

    PE0=0;
    PE1=1;
    PE2=1;
    PE3=0;
    PE4=0;
    PE5=0;
    PE6=1;
    PE7=0;
    CLK_SysTickDelay(50);
    //print(3)-----------------------//
```

```
    PC4=0;
    PC5=0;
    PC6=1;
    PC7=0;

    PE0=0;
    PE1=1;
    PE2=0;
    PE3=0;
    PE4=0;
    PE5=0;
    PE6=0;
    PE7=1;
    CLK_SysTickDelay(50);
    //print(0)-----------------------//

    PC4=0;
    PC5=0;
    PC6=0;
    PC7=1;

    PE0=0;
    PE1=1;
    PE2=0;
    PE3=0;
    PE4=0;
    PE5=0;
    PE6=0;
    PE7=1;
    CLK_SysTickDelay(50);
    //print(0)-----------------------//
```

▲圖 1.1

之後和組員討論後發現能夠直接使用 Seven_Segment.c 和 Scankey.c 裡面的 Function，於是 main 函示直接變得乾淨且簡短(如圖 1.2)

```
int main(void)
{
  OpenSevenSegment();
  while(1){

      ShowSevenSegment(0,0);
      CLK_SysTickDelay(50);
      CloseSevenSegment();

      ShowSevenSegment(1,3);
      CLK_SysTickDelay(50);
      CloseSevenSegment();

      ShowSevenSegment(2,0);
      CLK_SysTickDelay(50);
      CloseSevenSegment();

      ShowSevenSegment(3,0);
      CLK_SysTickDelay(50);
      CloseSevenSegment();

      Bonus();


    }
}
```

▲圖 1.2

另外，在 Seven_Segment.c 裡頭的 OpenSevenSegment 函示中的功能就是把 PC 和 PE 設定成可以輸入或是輸出，因為可能在不同功能上要有不同的輸入輸出需求。

有一連串的 define 是為了讓共陽極的七段顯示器顯示的方法，在 ShowSevenSegment 函示中，變數 no 指定哪一個 PC 要打開，且用 temp 去接 SEG_BUF[number]裡存的數字去做位元運算，並指定要亮的部分。(如圖 1.3)

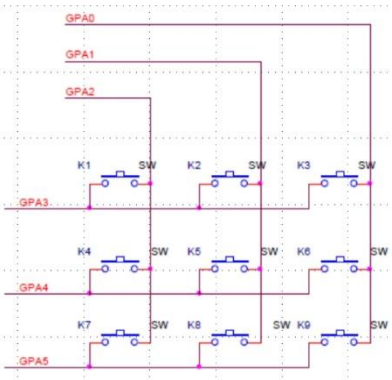
```
#define SEG_N0    0x82
#define SEG_N1    0xEE
#define SEG_N2    0x07
#define SEG_N3    0x46
#define SEG_N4    0x6A
#define SEG_N5    0x52
#define SEG_N6    0x12
#define SEG_N7    0xE6
#define SEG_N8    0x02
#define SEG_N9    0x62
#define SEG_N10   0x22
#define SEG_N11   0x1A
#define SEG_N12   0x93
#define SEG_N13   0x0E
#define SEG_N14   0x13
#define SEG_N15   0x33
```

```
void ShowSevenSegment(uint8_t no, uint8_t number)
{
    uint8_t temp,i;
  temp=SEG_BUF[number];

  for(i=0;i<8;i++)
     {
    if((temp&0x01)==0x01)
       switch(i) {
         case 0: PE0=1; break;
         case 1: PE1=1; break;
         case 2: PE2=1; break;
         case 3: PE3=1; break;
         case 4: PE4=1; break;
         case 5: PE5=1; break;
         case 6: PE6=1; break;
         case 7: PE7=1; break;
         }
       else
         switch(i) {
         case 0: PE0=0; break;
         case 1: PE1=0; break;
         case 2: PE2=0; break;
         case 3: PE3=0; break;
         case 4: PE4=0; break;
         case 5: PE5=0; break;
         case 6: PE6=0; break;
         case 7: PE7=0; break;
         }
       temp=temp>>1;
    }
      switch(no) {
        case 0: PC4=1; break;
        case 1: PC5=1; break;
        case 2: PC6=1; break;
        case 3: PC7=1; break;
      }
}
```

▲圖 1.3

再來 Bonus 的部分，當初看不太懂電路圖(如圖 1.4)為什麼要這麼做，後來經助教說明才知道這麼做可以減少電線成本，且只需六條就可以實現，雖然程式執行會比較耗時，但多出的時間極小，基本上不會影響。




```
uint8_t ScanKey(void)
{
  PA0=1; PA1=1; PA2=0; PA3=1; PA4=1; PA5=1;
  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;
  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;
  if (PA3==0) return 3;
  if (PA4==0) return 6;
  if (PA5==0) return 9;
  return 0;
}
```

▲圖 1.4

<心得與收穫>

雖然 GPIO 是 LAB1 的實驗應該會是最簡單的，但一開始我就遇到了許多問題，不知道該從哪裡開始下手，經過了好幾次嘗試才慢慢研究出來，在更深入理解之後，才發現有很多東西我雖然可以在開發板上顯示出來，但其背後的很多運作原理是我必須花時間釐清的。有點期待又有點害怕之後的實驗會困難到我想不出方法解決，但我會不斷提醒自己克服恐懼，這樣才能更加突破且學到更多東西!