微算機實驗報告

Lab #01

姓名：洪巧芸

系級：資工 114

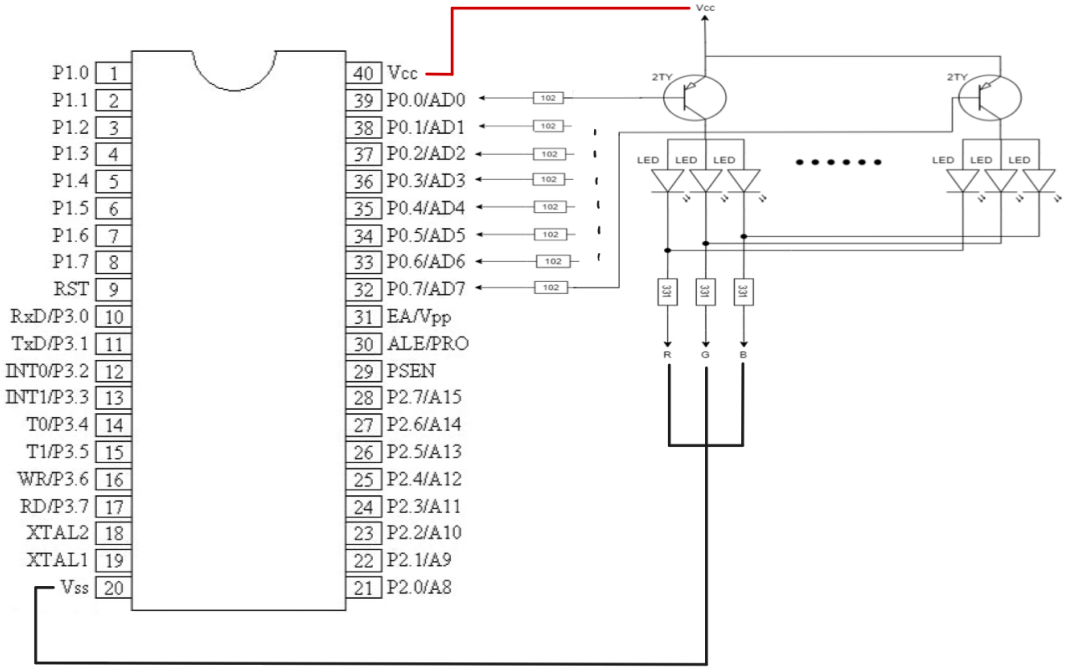
學號：110550143

上課時間：2023/9/18

一、實驗目的：

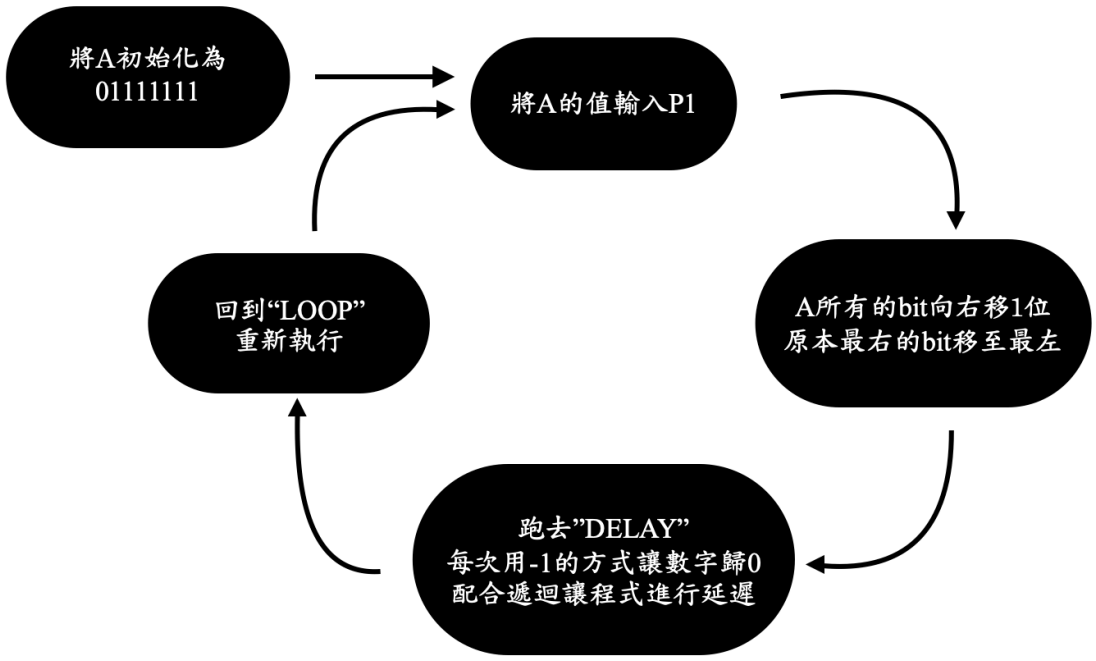
讓我們熟悉 Keil uVision 的環境以及 8051 組合語言，並運用基礎的 MOV, ACALL 等指令配合 MAIN, LOOP, DELAY 不同的 function 去實現 LED 的燈亮排序與顏色轉變。

二、硬體架構：

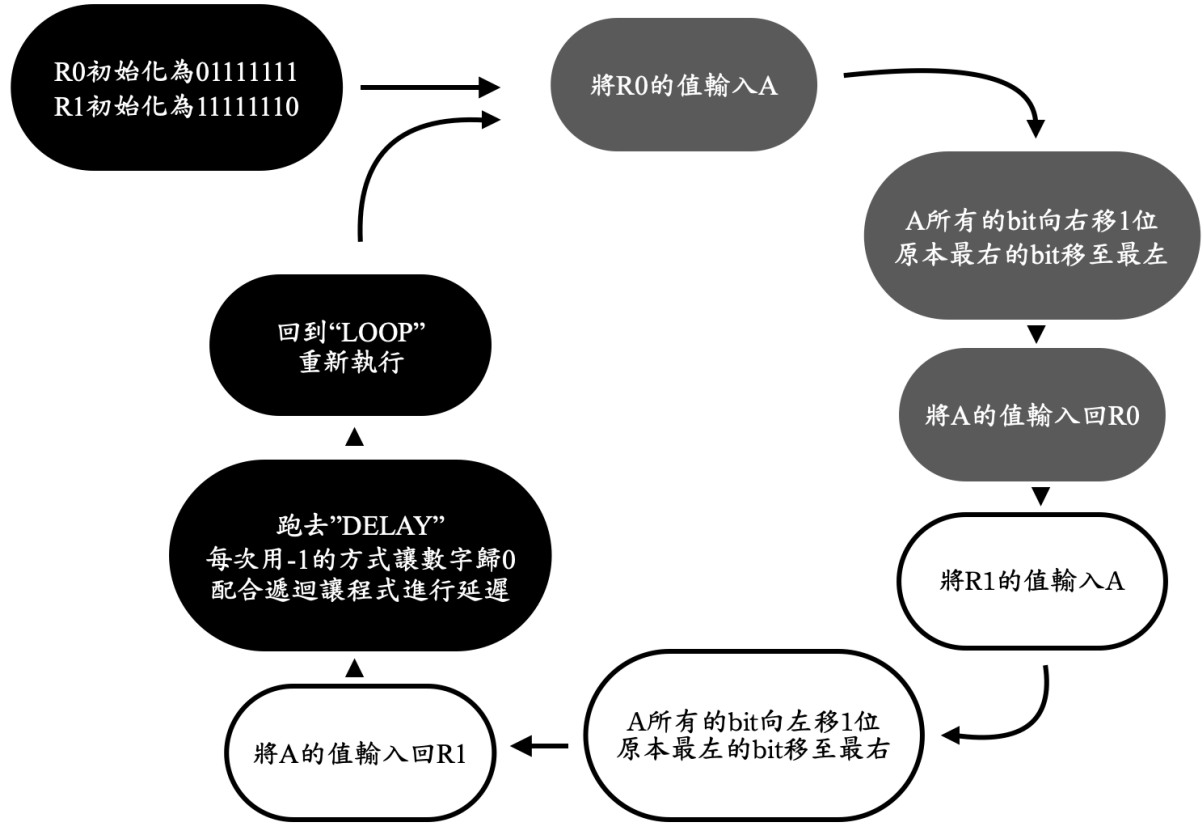


三、程式流程圖：

A. 基本題



B. 進階題



四、問題與討論：

(1) 若時間隔設定為 0.7 sec，則時間延遲的副程式為何?(請寫出精確解)

	指令時間	Delay Time	(R5,R6,R7)
原本	DJNZ = 24 cycles = 2 μs	Delay time(μs) {2+(2+(2+(4*R7)+4)*R6+4)*R5+4}/12	(255,255,5)
改為 0.7s	MOV = 2cycles = 1 μs	1+(1+(1+2*R7+2)*R6+2)*R5+2 = 700000	(43,52,155)

```
1   target = 699997
2   solutions = []
3
4   for x in range(1, target):
5       for y in range(1, target):
6           for z in range(1, target):
7               if 3*x + 3*x*y + 2*x*y*z == target:
8                   solutions.append((x, y, z))
9
10  for solution in solutions:
11      print(f'x = {solution[0]}, y = {solution[1]}, z = {solution[2]}')
```

(2) 請說明 SJMP、AJMP 與 LJMP 各自功能及三者的差異。

三者均為無條件轉移指令，以改變程序計數器 PC 中的內容爲宗旨，讓程式跳轉到指定位子。

	可跳轉範圍	佔用儲存空間
SJMP rel	8 bit = 256 bytes	2 bytes
AJMP addr11	11 bit = 2K bytes	2 bytes
LJMP addr16	16 bit = 64K bytes	3 bytes

(3) 請計算基礎題的範例程式執行從第四行到至第十行所需的時間。(Delay function 為講義前面的工作時間計算範例)

```
4    MAIN :                ;
5         MOV    A, #7FH   ;   -> 12 cycles
6    LOOP :                ;
7         MOV    P1, A     ;   -> 12 cycles
8         RR     A         ;   -> 12 cycles
9         ACALL  DELAY     ;   -> 24 cycles + 846093*12 cycles
10        JMP    LOOP      ;   -> 24 cycles
```

Total: (7+846093)*12 cycles = 846100 μs

五、程式碼與註解：

A、基本題

```
1   //初始設定
2   ORG 0000H;
3   AJMP MAIN;
4   ORG 0050H;
5
6   MAIN :
7   |   MOV A, #7FH; //將A初始化為01111111 -> 最左邊的燈亮
8
9   LOOP:
10  |   MOV P0, A ; //把A的值輸入P0, 也就把01111111的信號輸入LED燈裡面
11  |   RR A; //讓A所有的bit都向右移一位, 原本在最右的bit則移至最左
12  |   ACALL DELAY; //跳出迴圈, 跑到 "DELAY" 這個function
13  |   JMP LOOP; //跳到 "LOOP", 也就是在重新執行這個迴圈
14
15  DELAY:
16  |   MOV R5, #0FFH; //令R5的初始值為255
17
18  DELAY1:
19  |   MOV R6, #0FFH; //令R6的初始值為255
20
21  DELAY2:
22  |   MOV R7, #05H; //令R7的初始值為5
23
24  DELAY3:
25  |   DJNZ R7, DELAY3 ; //每一次都將R7的數值-1, 如果減完之後R7≠0, 就跳到 "DELAY3" 繼續執行
26  |   |   |   |   |   |   如果減完之後R7=0, 就直接執行下一行
27  |   DJNZ R6, DELAY2 ; //每一次都將R6的數值-1, 如果減完之後R6≠0, 就跳到 "DELAY2" 繼續執行
28  |   |   |   |   |   |   如果減完之後R6=0, 就直接執行下一行
29  |   DJNZ R5, DELAY1 ; //每一次都將R5的數值-1, 如果減完之後R5≠0, 就跳到 "DELAY1" 繼續執行
30  |   |   |   |   |   |   如果減完之後R5=0, 就直接執行下一行
31  |   RET; //回到ACALL的下一行, 也就是JMP LOOP
32  END;
```

B、進階題

```
ORG 0000H;
AJMP MAIN;
ORG 0050H;

MAIN :
    MOV A, #00H; //宣告A 為00000000
    MOV R3, #00H; //宣告R3 為00000000
    MOV R0, #0FEH; //將R0初始化為01111111 -> 最左邊的燈亮
    MOV R1, #07FH; //將R1初始化為10000000 -> 最右邊的燈亮

LOOP:
    //因為只有A可以進行RR或RL, 因此會先把數值輸入A進行轉換之後再存回去R裏面
    MOV A, R0; //把R0的值輸入A
    RR A; //讓A所有的bit都向右移一位, 原本在最右的bit則移至最左
    MOV R0, A ; //把右移過的A輸回去R0

    MOV A, R1; //把R1的值輸入A
    RL A; //讓A所有的bit都向左移一位, 原本在最左的bit則移至最右
    MOV R1, A ; //把右移過的A輸回去R0

    ANL A, R0; //把R1和R0 and 起來, 得到現在應該要亮的燈的位子, 因為現在R1與A兩者的數值一樣
               之後又會需要用到R1, 但A可以視為一個temp, 因此直接用A和R0去and,
               並把最後的結果數值輸入A -> 現在A就會代表應該要亮的LED燈位置

    MOV P0, A; //把A的值輸入P0, 也就把A當前的信號輸入LED燈裡面
    ACALL DELAY; //跳出迴圈, 跑到 "DELAY" 這個function
    JMP LOOP; //跳到 "LOOP", 也就是在重新執行這個迴圈

//和A部分一樣
DELAY:
    MOV R5, #0FFH; //令R5的初始值為255
DELAY1:
    MOV R6, #0FFH ; //令R6的初始值為255
DELAY2:
    MOV R7, #05H; //令R7的初始值為5
DELAY3:
    DJNZ R7, DELAY3; //每一次都將R7的數值-1, 如果減完之後R7≠0, 就跳到 "DELAY3" 繼續執行
                       如果減完之後R7=0, 就直接執行下一行
    DJNZ R6, DELAY2; //每一次都將R6的數值-1, 如果減完之後R6≠0, 就跳到 "DELAY2" 繼續執行
                       如果減完之後R6=0, 就直接執行下一行
    DJNZ R5, DELAY1; //每一次都將R5的數值-1, 如果減完之後R5≠0, 就跳到 "DELAY1" 繼續執行
                       如果減完之後R5=0, 就直接執行下一行
    RET ; //回到ACALL的下一行, 也就是JMP LOOP
END:
```

六、心得：

1.對於上課內容的心得感想。

以往的實驗課程老師通常不會在實驗前對於實驗內容有太多的講述，但這堂課老師會以晚上的實作範例先進行講解，讓我對於晚上的實作有初步的了解，

在實作時能夠更輕易地上手，也能夠比較快速進入狀況，了解這個沒有接觸過且相較於其他程式較不直觀的組合語言。

2.對於實驗內容的心得感想。

平常接觸的程式語言通常都是在電腦上面運算，可以看到的就是電腦螢幕的資料跳動，或是數值轉變，並沒有將程式碼輸出到其他硬體設備上，因此這堂課的實驗對我而言格外新奇，可以將自己的程式輸出到另外的 LED 燈上，並改變他的閃爍模式與顏色。此外這堂課也讓我意識到了討論的重要性，現場打程式可以讓我在有問題時馬上發問，向助教或同學請教，相較於其他課程的課後作業，我覺得這個方法更能夠讓我快速了解程式的基本運算原理。