微算機實驗報告

Lab #03

姓名：洪巧芸

系級：資工 114

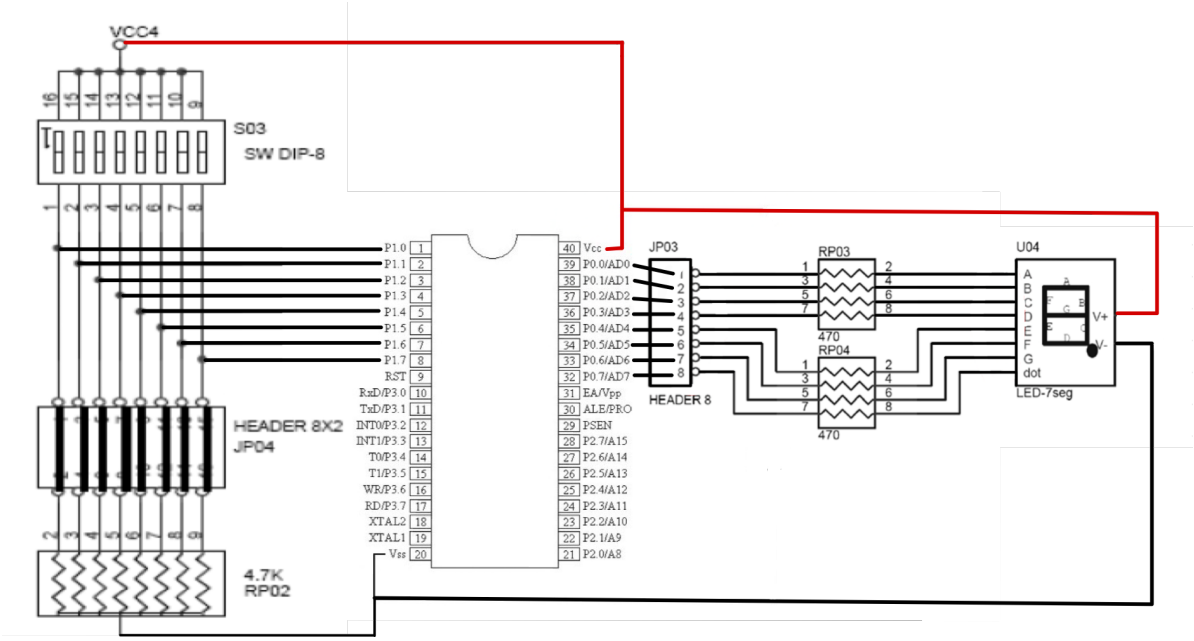
學號：110550143

上課時間：2023/10/3

一、實驗目的：

了解七段顯示器的電路結構及相關的控制方法，用 8bits 的信號輸入去控制七段顯示器的亮燈位置，並進一步學習資料表 (TABLE)的建立與使用，將所有需要在後續程式中使用到的資料先燒錄到板子上，再在需要使用時抓取目標數所在的儲存格內容轉換成輸出訊號。

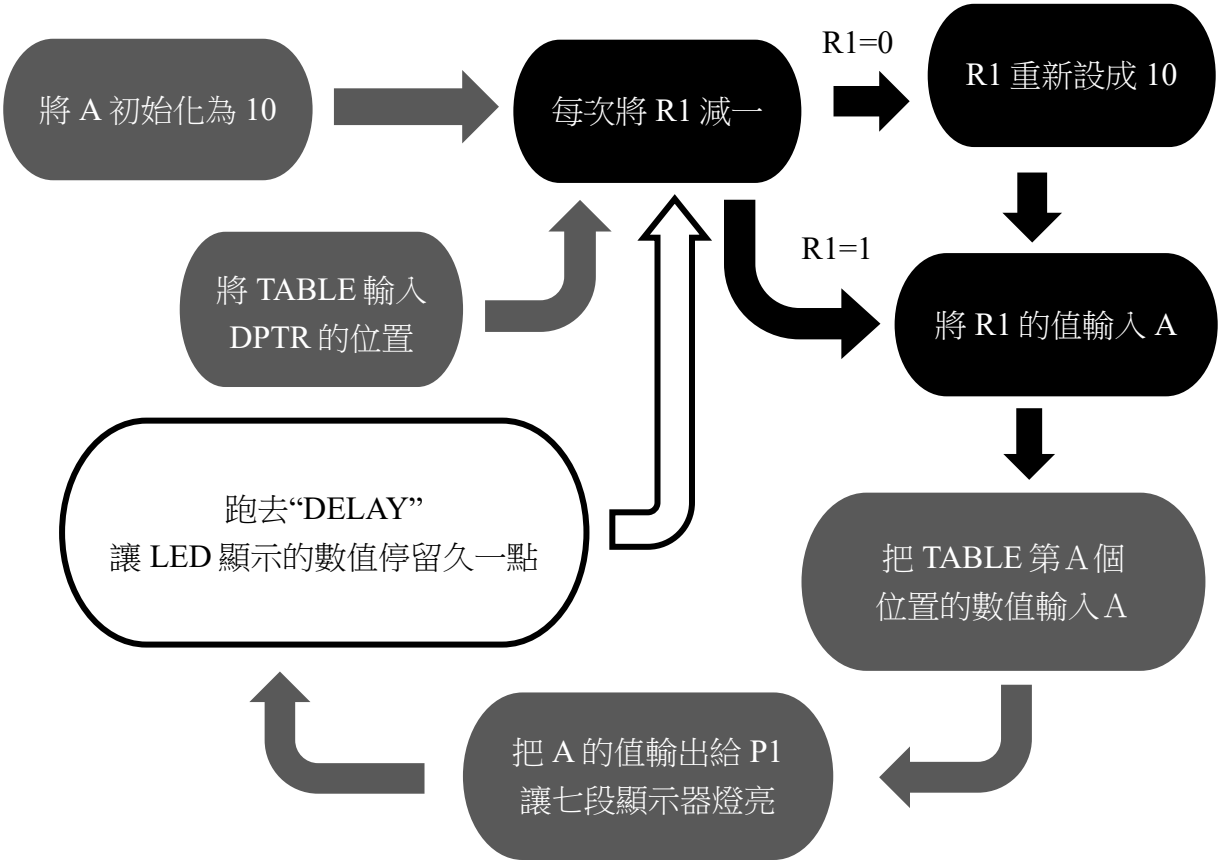
二、硬體架構：



三、程式流程圖：

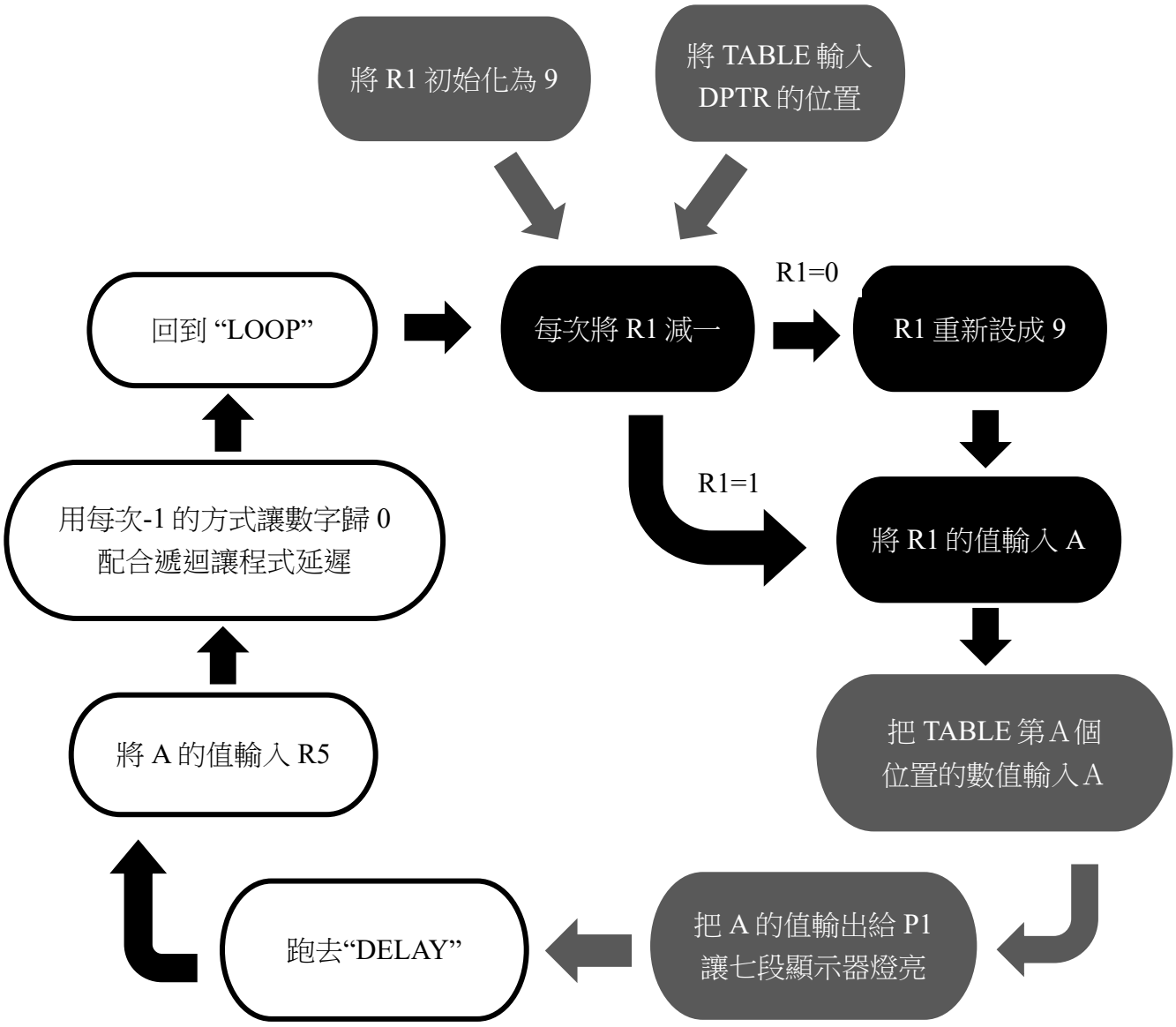
兩個題目都是希望藉由迴圈讓七段顯示器上的數字重複跳動，指撥開關的變動並不會影響顯示的數值。因此大致的邏輯均為設一個變數(R1)，每次將其 -1，並將對應到的 Table 值輸入七段顯示器中使燈亮，並在變數歸零時重新射程與初始化時相同的值。(初始化時數值要定義為要重複顯示的數字量 +1，因為配合 DJNZ 第一行不會被讀取到)

A. 基本題



B. 進階題

相較基礎題增加了一個利用指撥開關改變七段顯示器燈亮的速度，因此需要在將 A 輸入給七段顯示器後輸入到 DELAY 迴圈中，去改變燈亮的延遲時間，以實現燈亮速度改變的要求。



四、問題與討論：

(1) 若要在七段顯示七上顯示字母『A』、『b』、『C』、『d』、『E』與『F』，則於 JP03 中要輸入的訊號為何?

目標顯示字母	JP03 輸入訊號
A	P0 = 10001000B
b	P0 = 10000011B
C	P0 = 11000110B
d	P0 = 10100001B
E	P0 = 10000110B
F	P0 = 10001110B

(2) 當我擁有 500 筆 4bits 的資料時，我可以使用什麼方法佔用最少的空間儲存全部的資料？(請以 table 的使用為出發，列舉資料的儲存定義格式與資料讀取之方法)

每兩筆資料合併儲存，並設立兩個暫存器決定應該抓取資料前段或後段。

資料	儲存位置 A(4bits)+B(4bits)	R1	A	程式碼
1~256 組	A 區段	0	提取 A 區段資料	ANL A,#11110000
257~512 組	B 區段	1	提取 B 區段資料	ANL A,#00001111

(3) 假設我需要在下面的 TABLE 中取出 0x55 的資料，請寫出我須下達的指令。

TABLE	16 進制	抓取所需指令
DB 0FFH	0xFF	MOV A, #00H;
DB 10101010B	0xAA	MOV A, #01H;
DB 01010101B	0x55	MOV A, #02H;
DB 123DEC	0x7B	MOV A, #03H;

因為 0x55 是 16 進位的表示法，所以先將 TABLE 的數值全部轉換為 16 進制表示，配合數字所在的位置不同，改變最一開始給 A 的初始值。對應上面的表

格，我們可以知道運用 MOV 與 MOVC 去抓取目標值所需要的代碼為：MOV A, #02H; MOVC A, @DPTR+A;

五、程式碼與註解：

A、基本題

```
1   //初始設置
2   ORG 0000H;
3   AJMP MAIN;
4   ORG 0050H;
5
6   MAIN :
7   |   MOV     R1, #0AH; //因為要輸出0-9共10個數字, 因此將A的初始值設為10
8   |   MOV     DPTR, #NUM; //將NUM的位置存入DPTR
9
10  LOOP:
11  |   DJNZ    R1, RESET; //每次將R1減一, 若R1≠0, 則跳到"RESET"執行
12  |   |   |   |   |   |   若R1=0, 則跳到下一行執行 (12)
13  |   MOV     R1, #0AH; //把R1的數值改為9
14
15  RESET:
16  |   MOV     A, R1; //把R1的值輸入A
17  |   MOVC    A, @A+DPTR; //將Table(DPTR)的第A筆數值輸給A
18  |   MOV     P0, A; //把A的信號輸入LED燈裡面
19  |   ACALL   DELAY; //跳出迴圈, 執行"DELAY"副程式
20  |   JMP     LOOP; //跳回"LOOP", 也就是重新執行這個迴圈
21
22  /*"DELAY": 同Lab1, 為避免LED的數值跳動太快,
23  | 導致我們肉眼無法判讀, 因此需要延遲LED每次的顯示時長*/
24  DELAY:
25  |   MOV R5, #0FFH;
26  DELAY1:
27  |   MOV R6, #0FFH;
28  DELAY2:
29  |   MOV R7, #05H;
30  DELAY3:
31  |   DJNZ R7, DELAY3;
32  |   DJNZ R6, DELAY2;
33  |   DJNZ R5, DELAY1;
34  |   RET;
35
```

```
36  /*"NUM": 印出的學號table, 因為R1是從9開始往下扣
37  | 所以9要在最下面, 並依次序往上寫*/
38  NUM:
39  |   DB  00; //因為是用DJNZ, 所以第0條不會被讀到
40  |   DB  11000000B; //0
41  |   DB  11111001B; //1
42  |   DB  10100100B; //2
43  |   DB  10110000B; //3
44  |   DB  10011001B; //4
45  |   DB  10010010B; //5
46  |   DB  10000010B; //6
47  |   DB  11111000B; //7
48  |   DB  10000000B; //8
49  |   DB  10011000B; //9
50
51  END;
```

B、進階題

```
1   //初始設置
2   ORG 0000H;
3   AJMP MAIN;
4   ORG 0050H;
5
6   MAIN :
7   |   MOV     R1, #09H; //因為學號只有9位, 所以將R1初始化為9
8   |   MOV     DPTR, #ME; //將ME的位置存入DPTR
9
10  LOOP:
11  |   DJNZ    R1, RESET; //每次將R1減一, 若R1≠0, 則跳到"RESET"執行
12  |   |   |   |   |   |   若R1=0, 則跳到下一行執行 (13)
13  |   MOV     R1, #09H; //把R1的數值改為9
14
15  RESET:
16  |   MOV     A,  R1; //把R1的值輸入A
17  |   MOVC    A,  @A+DPTR; //將Table(DPTR)的第A筆數值輸給A
18  |   MOV     P0, A; //把A的信號輸入LED燈裡面
19  |   MOV     A,  P1; //把指撥開關的數值輸入A
20  |   ACALL   DELAY; //跳出迴圈, 執行"DELAY"副程式
21  |   JMP     LOOP; //跳回"LOOP", 也就是重新執行這個迴圈
22
23  /*"DELAY": 同Lab1, 為避免LED的數值跳動太快,
24  |  導致我們肉眼無法判讀, 因此需要延遲LED每次的顯示時長*/
25  DELAY:
26  |   MOV R5, A; //把A的數值輸入R%,
27  |   |   |   |   此時A代表指撥開關的數值, A越大迴圈跑越多次, LED跳動越慢
28
29  DELAY1:
30  |   MOV R6, #0FFH; //令R6初始值為255
31  DELAY2:
32  |   MOV R7, #09H; //令R7初始值為9
33  DELAY3:
34  |   DJNZ R7, DELAY3;
35  |   DJNZ R6, DELAY2;
36  |   DJNZ R5, DELAY1;
37  |   RET;
38
```

```
39  /*"ME": 印出的學號table, 因為R1是從9開始往下扣
40  |  所以學號的第一位要在最下面, 並依次序往上寫*/
41  ME:
42  |   DB  00; //因為是用DJNZ, 所以第0條不會被讀到
43  |   DB  10110000B; //3
44  |   DB  10011001B; //4
45  |   DB  11111001B; //1
46  |   DB  11000000B; //0
47  |   DB  10010010B; //5
48  |   DB  10010010B; //5
49  |   DB  11000000B; //0
50  |   DB  11111001B; //1
51  |   DB  11111001B; //1
52
53  END;
```

六、心得：

1. 對於上課內容的心得感想：

這次的課程加入了”Table”的概念，讓程式可以運行的複雜程度多了一個幅度的提升，且可以直接依照 Table 的傳入值去尋找對應的數值，最特別的是因為 DJNZ 與 Table 配合，因此在 Table 中的第一行是不會被讀取到的，但卻有其存在之必要性，與我以往所學到的程式精簡有很大的不同，從來沒有想過會有「會被略過，但需要這行」的情況。

2. 對於實驗內容的心得感想：

這次實驗時是直接照著老師上課講解的 PPT 去實作 LED 的數字燈亮，但一開始燈亮一直是亂碼，沒有辦法呈現出數字，只有數字 4 會正確的顯示出來，其他都只會有部分區段燈亮，重新檢查電路後才發現是接線問題，如果街相反了在輸入 Table 中的 DB 時就應該將值整個對稱後書寫，程式碼需要隨著電路接線改變而改變，在之後的實驗中要格外注意。

	印出數字 1	**印出數字 2**	**印出數字 4**
接線與 PPT 相同	11111001B	10100100	10011001
接線與 PPT 相反	10011111B	00100101	10011001