微算機實驗報告

Lab # 9-1

姓名：仇健安
系級：電機系
學號：111511239
上課時間：2025/04/22

一、實驗目的：

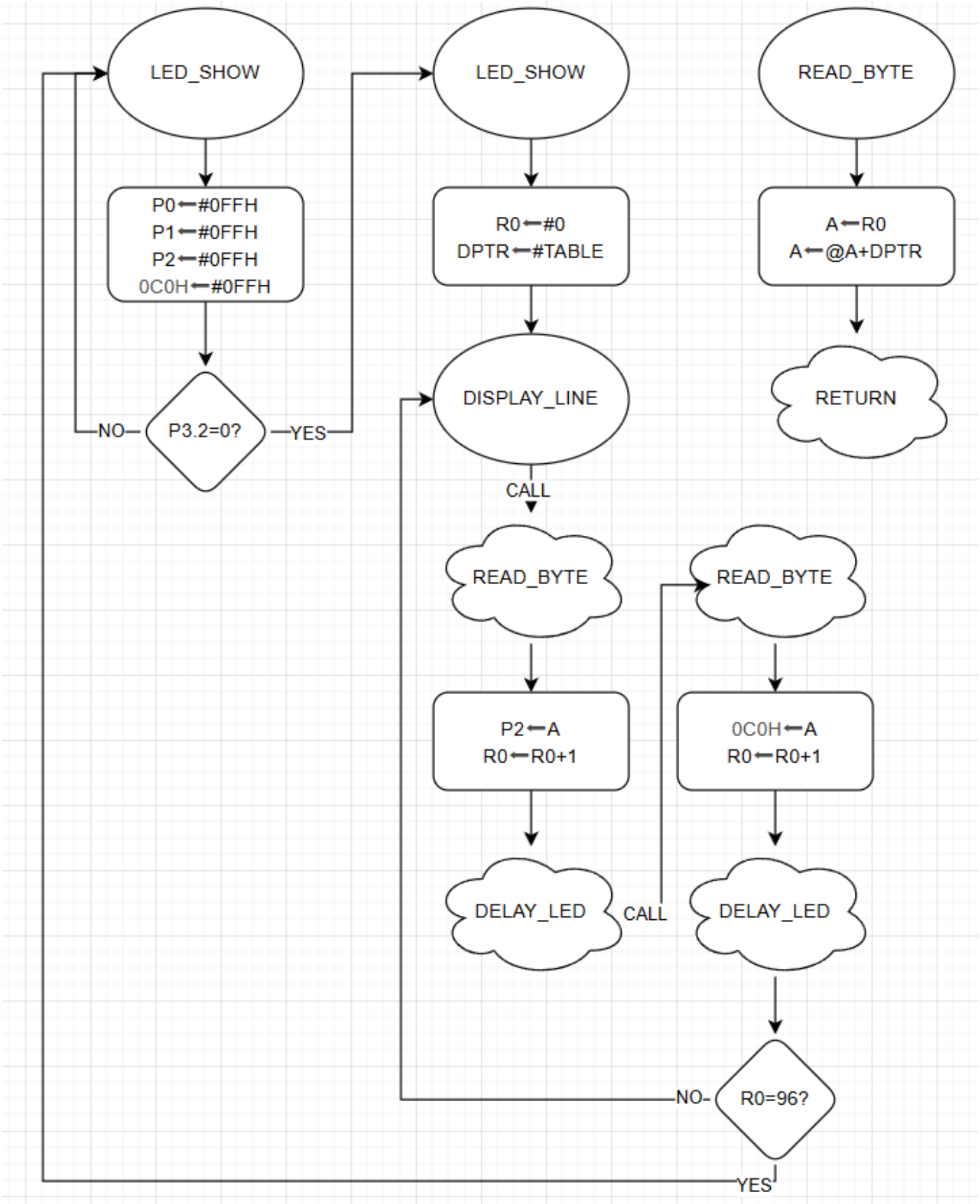
學習如何讀取大型資料表（TABLE）並精準控制 LED 閃爍，利用視覺暫留原理，讓 LED 旋轉模組顯示出清晰的中文字圖案。

二、硬體架構：

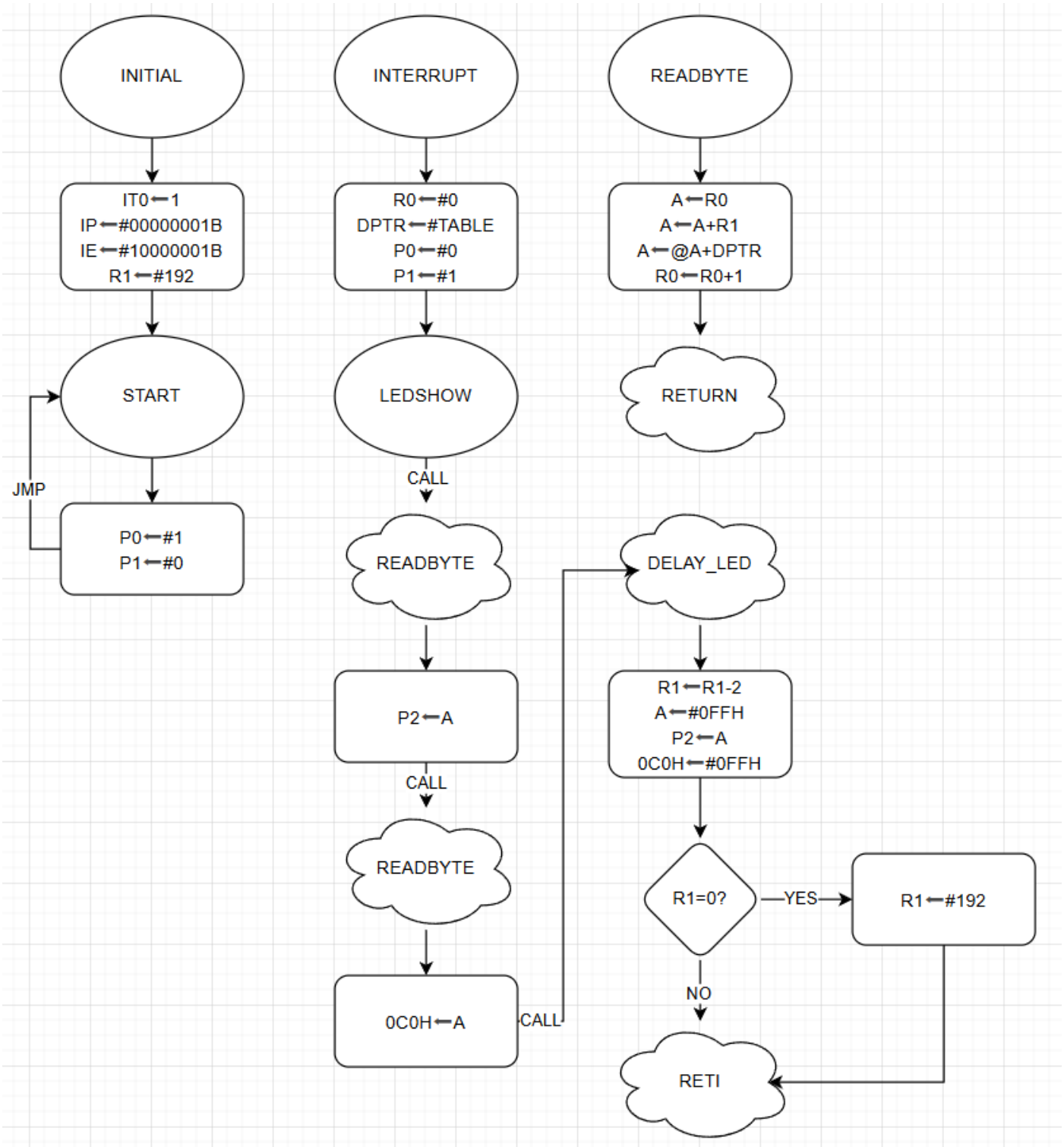
RS232 傳輸線
黑線對 GND
白線對 RXD
綠線對 TCD
紅線對 VCC

三、程式流程圖：

基本題：


LED_SHOW
P0←#0FFH
P1←#0FFH
P2←#0FFH
0C0H←#0FFH
P3.2=0?
NO
YES
LED_SHOW
R0←#0
DPTR←#TABLE
DISPLAY_LINE
CALL
READ_BYTE
P2←A
R0←R0+1
DELAY_LED
READ_BYTE
0C0H←A
R0←R0+1
DELAY_LED
CALL
R0=96?
NO
YES
READ_BYTE
A←R0
A←@A+DPTR
RETURN

進階題：


INITIAL
IT0←1
IP←#00000001B
IE←#10000001B
R1←#192
START
JMP
P0←#1
P1←#0
INTERRUPT
R0←#0
DPTR←#TABLE
P0←#0
P1←#1
LEDSHOW
CALL
READBYTE
P2←A
CALL
READBYTE
0C0H←A
CALL
READBYTE
A←R0
A←A+R1
A←@A+DPTR
R0←R0+1
RETURN
DELAY_LED
R1←R1-2
A←#0FFH
P2←A
0C0H←#0FFH
R1=0?
YES
R1←#192
NO
RETI

四、問題與討論：

(1) 顯示字數超過 8 字以上，TABLE 將會儲存超過 255 bytes，若使用讀 TABLE 的範例程式，一個暫存器將會沒辦法讓 DPTR 使用到 TABLE 內所有的資料，請問該如何克服，請盡可能描述你的想法。

方法一：多段 TABLE + 切換 DPTR（分區處理）

將 TABLE 拆成數段，每段最多 256 bytes（例如 TABLE1、TABLE2 等）。顯示前 8 字時使用 DPTR = #TABLE1，第 9 字以後使用 DPTR = #TABLE2。透過變更 DPTR 所指向的 TABLE 來處理更多資料。

方法二：利用進位旗標 C 判斷跨頁並切換 DPTR

使用兩個暫存器(例如 R0 與 R1)，將它們相加後用來讀取資料。由於 MOVC A, @A+DPTR 的 A 僅能容納 8-bit，因此當 R0 + R1 超過 255 時會觸發進位，8051 會將進位旗標 C 設為 1。此時可檢查 C 是否為 1，若是，則將 DPTR 改為指向 TABLE+256 的下一段資料，否則使用原本的 DPTR。這樣就能依進位狀況切換對應的段落，實現超過 256 bytes 的表格存取。

五、程式碼與註解：

基本題：

```
    ORG 0000H
    JMP LED_SHOW            ; 程式進入點，跳到 LED 顯示主程式

    ORG 0050H               ; 實際程式碼起始位址

;==============================
; 主程式：等待按鍵觸發後顯示LED畫面
;==============================
LED_SHOW:
    ; 將所有 LED（P0, P1, P2, P4）初始化為全暗（全為 1）
    MOV P0, #0FFH
    MOV P1, #0FFH
    MOV P2, #0FFH
    MOV 0C0H, #0FFH         ; P4 用 0C0H 存取（特殊寫法）

    ; 等待按鈕觸發（P3.2 為低電位即為按下）
    JNB P3.2, SCAN_LOOP   ; 若按鍵為 0，進入 SCAN_LOOP
    JMP LED_SHOW            ; 否則持續等待

;==============================
; 顯示 TABLE 資料到 LED
;==============================
SCAN_LOOP:
    MOV R0, #0              ; R0 作為 TABLE 的偏移索引
    MOV DPTR, #TABLE        ; DPTR 指向 TABLE 起始位置
```

```
DISPLAY_LINE:
    CALL READ_BYTE          ; 讀取 TABLE 的第 1 byte
    MOV P2, A               ; 顯示在 P2（平面LED第1資料）
    INC R0
    CALL DELAY_LED          ; 加入延遲讓畫面可見

    CALL READ_BYTE          ; 讀取 TABLE 的第 2 byte
    MOV 0C0H, A             ; 顯示在 P4（側面LED）
    INC R0
    CALL DELAY_LED          ; 再次延遲

    CJNE R0, #96, DISPLAY_LINE ; 若尚未顯示完畢 96 bytes，回去繼續顯示

    SJMP LED_SHOW           ; 顯示完畢，回到主程式等待下一次按鈕觸發

;==============================
; 子程式：讀取 TABLE 中的資料
;==============================
READ_BYTE:
    MOV A, R0               ; A = R0，取得偏移值
    MOVC A, @A+DPTR         ; 從程式記憶體讀取資料 A = *(DPTR + A)
    RET

;==============================
; 子程式：簡單延遲迴圈(調整時長可控制字體寬度)
;==============================
DELAY_LED:
    MOV R7, #10
DELAY1:
    MOV R6, #40
DELAY2:
    DJNZ R6, DELAY2
    DJNZ R7, DELAY1
    RET

;==============================
; TABLE：LED 顯示資料（字：別當我）
; 每字為 16x16 點陣圖，共 32 bytes
;==============================
TABLE:
```

```
; ===== 「別」 =====
DB 0FFH,07FH,0C1H,0BFH,0DDH,0CFH,05DH,0F0H
DB 0DDH,0BDH,0DDH,07DH,0DDH,0BDH,0C1H,0C1H
DB 0FFH,0FFH,0FFH,0FFH,007H,0F0H,0FFH,0BFH
DB 0FFH,07FH,000H,080H,0FFH,0FFH,0FFH,0FFH

; ===== 「當」 =====
DB 0DFH,0FFH,0E7H,0FFH,0F7H,003H,015H,0AAH
DB 0D3H,0AAH,0D7H,0AAH,0D7H,0AAH,0D0H,082H
DB 0D7H,0AAH,0D7H,0AAH,0D3H,0AAH,015H,0AAH
DB 0F7H,003H,0D7H,0FFH,0E7H,0FFH,0FFH,0FFH

; ===== 「我」 =====
DB 0DFH,0FFH,0DBH,0F7H,0DBH,0B7H,0DBH,07BH
DB 001H,080H,0DCH,0FDH,0DDH,0BEH,0DFH,0BFH
DB 0DFH,0DFH,000H,0ECH,0DFH,0F3H,0DDH,0EBH
DB 0D3H,0DDH,05FH,0BEH,0DFH,007H,0FFH,0FFH

END
```

進階題：

```
; ===== 程式進入點與中斷向量設定 =====
ORG 0000H
    JMP INITIAL             ; 程式一開始執行 INITIAL

ORG 0003H
    JMP INTERRUPT           ; 外部中斷0（INT0）觸發後執行 INTERRUPT

ORG 0050H                   ; 程式實際開始位置

; ===== 初始化與主迴圈 =====
INITIAL:
    SETB IT0                ; 設定 INT0 為負緣觸發（falling edge）
    MOV IP, #00000001B      ; INT0 設為高優先權
    MOV IE, #10000001B      ; 啟用全域中斷（EA）與 INT0（EX0）
    MOV R1, #192            ; 顯示位置初始值（用來控制TABLE的起始偏移）

; ===== 主迴圈（背景動作）=====
START:
    MOV P0, #1              ; 可作為背景狀態顯示（此處設定 P0 為高電位）
    MOV P1, #0              ; 可用來確認切換狀態（例如P1點亮）
    JMP START               ; 無限迴圈，等待中斷發生

; ===== 中斷服務程式 =====
INTERRUPT:
    MOV R0, #0              ; 初始化TABLE讀取偏移為0
    MOV DPTR, #TABLE        ; 設定TABLE基址
    MOV P0, #0              ; 中斷中 P0 拉低
    MOV P1, #1              ; P1 拉高作為狀態指示

LEDSHOW:
    CALL READBYTE           ; 讀取一個 byte → A
    MOV P2, A               ; 顯示至 P2（平面LED）
    CALL READBYTE           ; 讀取下一個 byte → A
    MOV 0C0H, A             ; 顯示至 P4（側面LED，P4 無法直接使用需用地址0C0H）
    CALL DELAY_LED          ; 加入延遲形成顯示效果

    CJNE R0, #96, LEDSHOW ; 總共顯示 96 bytes（48組16x16點陣資料）

    ; 顯示完一輪後調整 R1 來形成跑馬燈移動效果
    DEC R1                  ; 向左平移一列
```

```
    DEC R1                  ; 一次兩列（因一字佔兩bytes）
    MOV A, #0FFH
    MOV P2, A               ; 清除LED殘影
    MOV 0C0H, #0FFH

    CJNE R1, #0, RETURN  ; 若尚未顯示完所有位移，則回去顯示
    MOV R1, #192            ; 否則重設初始顯示位置

RETURN:
    RETI                    ; 中斷結束返回主程式

; ===== 讀取TABLE中資料的副程式 =====
READBYTE:
    MOV A, R0               ;A ← 當前偏移位置
    ADD A, R1               ; 加上初始偏移 R1，形成實際顯示位移效果
    MOVC A, @A+DPTR         ; 從程式記憶體中取資料（MOVC 指令只能讀 code memory）
    INC R0                  ; 移動到下一個byte
    RET

; ===== 簡單的延遲副程式（延遲時間可視調整） =====
DELAY_LED:
    MOV R7, #10
DELAY1:
    MOV R6, #40
DELAY2:
    DJNZ R6, DELAY2
    DJNZ R7, DELAY1
    RET

; ===== 字型資料TABLE：顯示「別當我」三字，共 96 bytes（16x3字x2bytes） =====
TABLE:
    ; 前置空白區塊（讓文字出現前有間隔）
    DB 0FFH,0FFH, 0FFH,0FFH, 0FFH,0FFH, 0FFH,0FFH
    DB 0FFH,0FFH, 0FFH,0FFH, 0FFH,0FFH, 0FFH,0FFH
    DB 0FFH,0FFH, 0FFH,0FFH, 0FFH,0FFH, 0FFH,0FFH
    DB 0FFH,0FFH, 0FFH,0FFH, 0FFH,0FFH, 0FFH,0FFH

    ; 更多前置空白區塊（總共3組）
    ; ...（略）...

    ; 「別」 字型資料（16x16 = 32 bytes）
```

```
DB 0FFH,0FFH, 0FFH,07FH, 001H,0BFH, 0BBH,0C7H
DB 03BH,0F8H, 0BBH,0BDH, 0BBH,03DH, 0BBH,0BDH
DB 001H,0C1H, 0FBH,0FDH, 0FFH,0FFH, 007H,0D8H
DB 0F7H,0BFH, 0FFH,03FH, 001H,080H, 0FDH,0FFH

;「當」 字型資料
DB 0FFH,0FFH, 0BFH,0FFH, 0C7H,0FFH, 0EDH,003H
DB 0EBH,0ABH, 027H,0AAH, 0AFH,0AAH, 0AFH,0AAH
DB 0A1H,082H, 0ADH,0AAH, 0AFH,0AAH, 027H,0AAH
DB 0E9H,0ABH, 0ADH,003H, 0C7H,0FFH, 0EFH,0FFH

;「我」 字型資料
DB 0FFH,0FFH, 0BFH,0F7H, 0B7H,0A7H, 0B7H,0B7H
DB 0B7H,037H, 003H,080H, 0BBH,0FBH, 0BBH,0FBH
DB 0BFH,07DH, 0BFH,0BFH, 001H,0D8H, 0BDH,0E7H
DB 0BFH,0DBH, 0BBH,0BCH, 0B7H,07EH, 0A7H,00FH

; 後置空白區塊（讓文字消失時有空白過場）
;...（略）...

END
```

六、心得：

上課內容心得：

這次課程介紹了非 8051 架構下的 I/O 控制方式、中斷處理流程，以及點陣顯示的原理與程式設計。對我來說，這是第一次學習如何用程式控制實體 LED 點陣，老師也補充了 STC15F2K32S2 控制器的特殊 I/O（如 P4 需用 0C0H 存取），雖然一開始不太熟悉，但上課講解與範例說明幫助我更快掌握了操作方式。

實驗內容心得：

在實作旋轉 LED 實驗的過程中，我遇到不少挑戰。首先是拿到一塊壞掉的 LED 模組，花了不少時間確認軟體安裝與 code 本身沒有問題。接著一開始的負緣觸發程式也寫在錯誤的地方，導致文字會左右飄移，最後經過排查錯誤與修正邏輯，終於成功讓「別當我」三個字可以固定位置的出現在旋轉 LED 上，並實現了簡單的跑馬燈效果。整體來說雖然過程中有些挫折，但看到最後成功顯示的畫面非常有成就感，也讓我更熟悉了中斷處理與 LED 點陣資料控制的流程。

Notes:

1. 內容字體大小為 12，間距為單行間距
2. 中文字字體為標楷體
3. 英文字和阿拉伯數字為 Times New Roman
4. 嚴禁抄襲，抄襲者以 0 分計算
5. 請於報告左上角附上照片
6. 每次實驗課繳交上次實驗結報