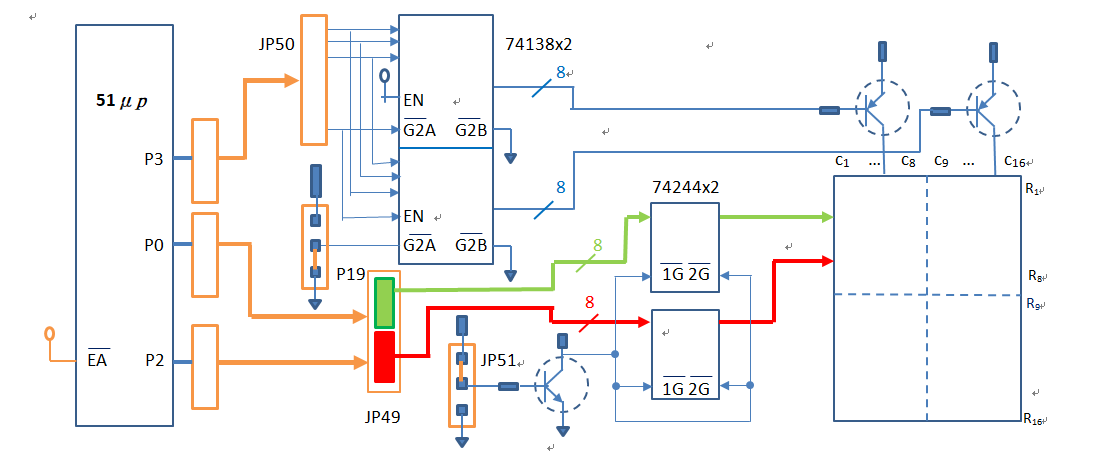
# 微處理機 Lab5 結報

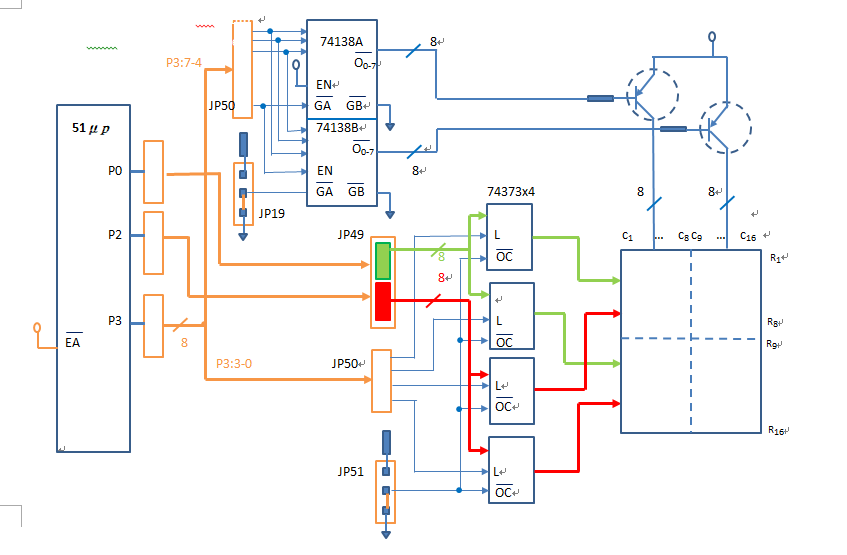
0413335 郭逸琳 0416039 李佳燕

**## 實驗主題與訓練目的**

透過此次實驗實作16\*16 tri-colored dot-matrix LED module

**## 實驗電路**





此兩張圖主要是差在J51屏蔽的位置，如果屏蔽上面兩個pin角的話表示電壓是從low to high，如果屏蔽下面兩個pin角的話就表示電壓是從high to low。經過這次實驗我們發現如果電壓是low to high的話會比較不穩定(容易出現不該有的雜訊)，但如果是屏蔽下面兩個pin角的話表示會限制住顯示的位置(如果是屏蔽左邊的下面兩個pin角的話就不能顯示下面那兩格LED燈的東西，如果是屏蔽右邊的下面兩個pin角的話就不能顯示右邊那兩格LED燈的東西)。

**## 實驗過程**

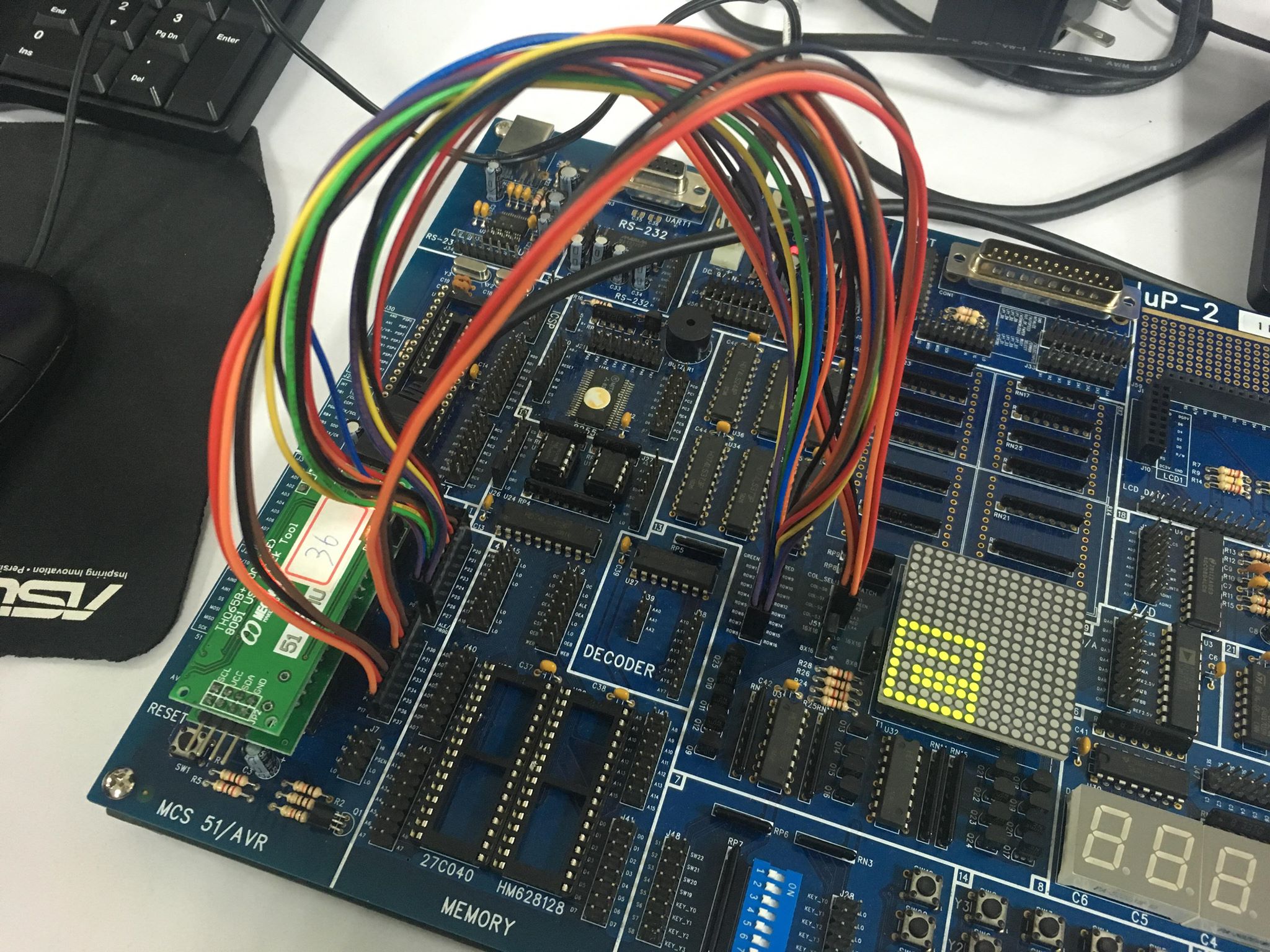
先在uvision裡貼上spec上的code

然後再利用下列方式接線：

P0接到JP49的green，P2接到JP49的red。P3則用兩個4pin的線接，P3的0~3接到J50左邊，P3的4~7接到J50的右邊。

J51則要屏蔽上面的兩個pin角。

實際上接的線路如下：(跑出來的結果會是在左上角的LED區塊亮2的陰文)



**## 解釋程式碼**

P3是決定這16\*16中要設定哪條線，P3的bit0~3是設定綠色跟紅色row(bit0是綠色的row 1~8, bit1是綠色的row 9~16, bit2是紅色的row 1~8, bit3是紅色的row 9~16)，bit4~6是設定要設定哪個column，bit7是決定要設左半邊的column還是右半邊的column，P0則是決定那條線的八個燈泡中哪些要亮綠色(0是亮1是暗)，P2是決定那條線的八個燈泡中哪些要亮紅色。

所以程式中是從左邊開始設起，每設完一個column call一次delay，當全部column都設完後就進入下一個階段。

階段順序為：先亮綠色2的陰文，再亮紅色2的陰文，最後再亮黃色2的陰文(紅色+綠色=黃色)

**## 實驗延伸**

上禮拜五我們這組的要求是原本顯示在左上區塊的2移到左下區塊

因為在green\_2, red\_2, yellow\_2, 我們可以透過設定A的初始值,

決定要讓圖形顯示在左上, 左下, 右上, 右下哪個區塊

(因為A的值會給P3)

P3的0 1 2 3這4個bit是決定"上下(row)"

P3[3] Latch: Red for row9-16

P3[2] Latch: Red for row1-8

P3[1] Latch: Green for row9-16

P3[0] Latch: Green for row1-8

在green\_2

mov A, #1H 改成 mov A, #2H

在red\_2

mov A, #4H 改成 mov A, #8H

在yellow\_2

mov A, #5H 改成 mov A, #aH

這樣就可以亮在左下的區塊了

而P3的高四位4 5 6 7這4個bit是決定"左右(column)"

如果要從原本亮左上的區塊變成亮右上的區塊, 則也是在上面三個label的A的初始值的"高四位"+8

在green\_2

mov A, #1H 改成 mov A, #81H

在red\_2

mov A, #4H 改成 mov A, #84H

在yellow\_2

mov A, #5H 改成 mov A, #85H

這樣就會亮在右上的區塊了

而為什麼要+8, 是因為他column是一行一行set好的,

所以每個圖形都從column8開始設, 之後程式碼會add A, #10H,

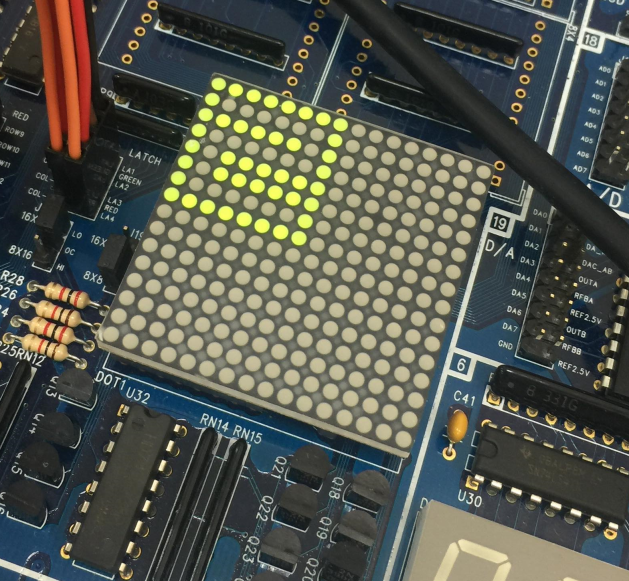
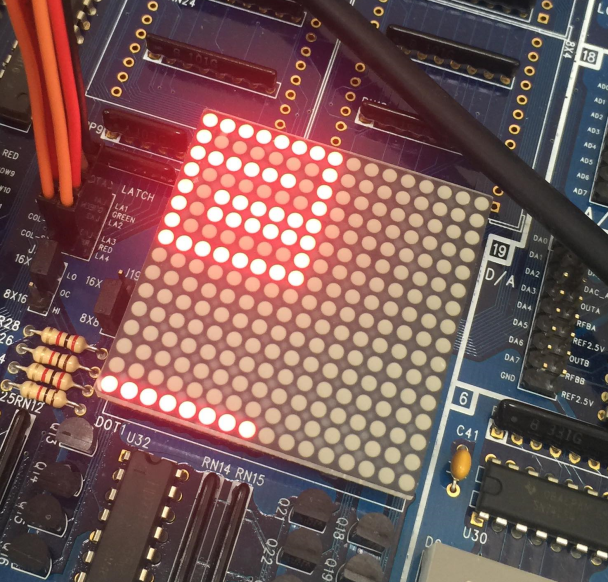
這裡即是設好一個column就往下一個column開始設的意思

**## 實驗心得**

透過這次實驗，我們又多了解了一個模組的使用方式。希望下次還有機會學更多有趣的模組。

**## 實驗問題**

一開始我們很快就做出來了, 我們之後就試著玩電路板, 讓16\*16 tri-colored dot-matrix LED module上陰文的2能顯示在不同的區塊內, 但是當我們要接回來原來實驗的要求的時候, 就發現有雜訊的干擾, 不是code的問題, 經過探討後我們覺得是因為我們是low to high, 而這樣好像比較不穩定, 於是我們就把它改成high to low, 可是這樣就變成原來要顯示的那個區塊是正確的, 而其他電流可通過的區塊卻會全亮. 然後我們試著把全亮的地方壓下來讓她不會亮, 那時我們是想說low to high是0會亮1不會亮, 那如果把1跟0對調會不會就壓下來了?我們試著把code改成上面的這個想法, 結果就變成下面的結果(當是綠色的時候可以全部壓下來, 但當是紅色的時候最下面那排卻壓不下來)

; the sample code is intended for displaying a decimal “2” on the upper-left 8x8 LED component in green, red and yellow color

; alternately, with the 16x16 LED module wired for the 16\*16 operation as shown in (a.2) .

; the sample code guarantees neither syntax error free nor bug free; fix all errors encountered.

; code testing procedure (strongly recommended)

; (1) first run the green\_2 loop alone, making sure that you see a “2” in green appear on the designated 8x8 LED

; (2) then run the red\_2 loop alone to see what should appear.

; (3) finally run the entire code altogether.

; P0: GREEN [why????]

; P2: RED [why????]

; P3[7] 138A-GA P3[3] Latch: Red for row9-16

; P3[6] 138-22 P3[2] Latch: Red for row1-8

; P3[5] 138-21 P3[1] Latch: Green for row9-16

; P3[4] 138-20 P3[0] Latch: Green for row1-8

org 0

mov SP, #50H

mov P3, #0

;call delay

mov P0, #0FFH

mov P2, #0FFH

mov P3, #5H

mov P3, #0

start:

mov R6, #250

green\_2:

mov P3, #0

mov P0, #0H

mov A, #1H

mov P3, A

call delay ; col1 done

anl P3, #0F0H ; ==XXX==

mov P0, #7aH

add A, #10H

mov P3, A ; A:= ???

call delay ; col2 done

mov R7, #4

g2\_loop:

anl P3, #0F0H ; ==XXX==

mov P0, #4AH

add A, #10H

mov P3, A ; A:= ???

call delay ; col3-6 done in sequence

djnz R7, g2\_loop

anl P3, #0F0H ; ==XXX==

mov P0, #4EH

add A, #10H

mov P3, A

call delay ; col7 done

anl P3, #0F0H ; ==XXX==

mov P0, #0H

add A, #10H ; A:= ???

mov P3, A

call delay ; col8 done

djnz R6, green\_2

anl P3, #0F0H ; ==AA==

mov P0, #0FFH ; ==AA==

mov P3, A ; ==AA==

call delay

redd:

mov R6, #250

red\_2:

mov P3, #0

mov P2, #0H

mov A, #4H

mov P3, A

call delay ; col1 done

anl P3, #0F0H ; ==XXX==

mov P2, #7AH

add A, #10H ; A:= ???

mov P3, A

call delay ; col2 done

mov R7, #4

r2\_loop:

anl P3, #0F0H ; ==XXX==

mov P2, #4AH

add A, #10H ; A:= ???

mov P3, A

call delay ; col3-6 done

djnz R7, r2\_loop

anl P3, #0F0H ; ==XXX==

mov P2, #4EH

add A, #10H ; A:= ???

mov P3, A

call delay ; col7 done

anl P3, #0F0H ; ==XXX==

mov P2, #0H

add A, #10H ; A:= ???

mov P3, A

call delay ; col8 done

djnz R6, red\_2

anl P3, #0F0H ; ==BB==

mov P2, #0FFH ; ==BB==

mov P3, A ; ==BB==

call delay

mov R6, #250

yellow\_2:

mov P0, #0H

mov P2, #0H

mov A, #5H

mov P3, A

call delay ; col1 done

anl P3, #0F0H ; ==XXX==

mov P0, #7AH

mov P2, #7AH

add A, #10H ; A:= ???

mov P3, A

call delay ; col2 done

mov R7, #4

y2\_loop:

anl P3, #0F0H ; ==XXX==

mov P0, #4AH

mov P2, #4AH

add A, #10H ; A:= ???

mov P3, A

call delay ; col3-6 done

djnz R7, y2\_loop

anl P3, #0F0H ; ==XXX==

mov P0, #4EH

mov P2, #4EH

add A, #10H ; A:= ???

mov P3, A

call delay ; col7 done

anl P3, #0F0H ; ==XXX==

mov P0, #0H

mov P2, #0H

add A, #10H ; A:= ???

mov P3, A

call delay ; col8 done

djnz R6, yellow\_2

anl P3, #0F0H ; ==CC==

mov P0, #0FFH ; ==CC==

mov P2, #0FFH ; ==CC==

mov P3, A ; ==CC==

call delay

jmp start

delay: push 2

push 3

mov R2, #2

dd1: mov R3, #250

djnz R3, $

djnz R2, dd1

pop 3

pop 2

ret

end