微算機實驗報告

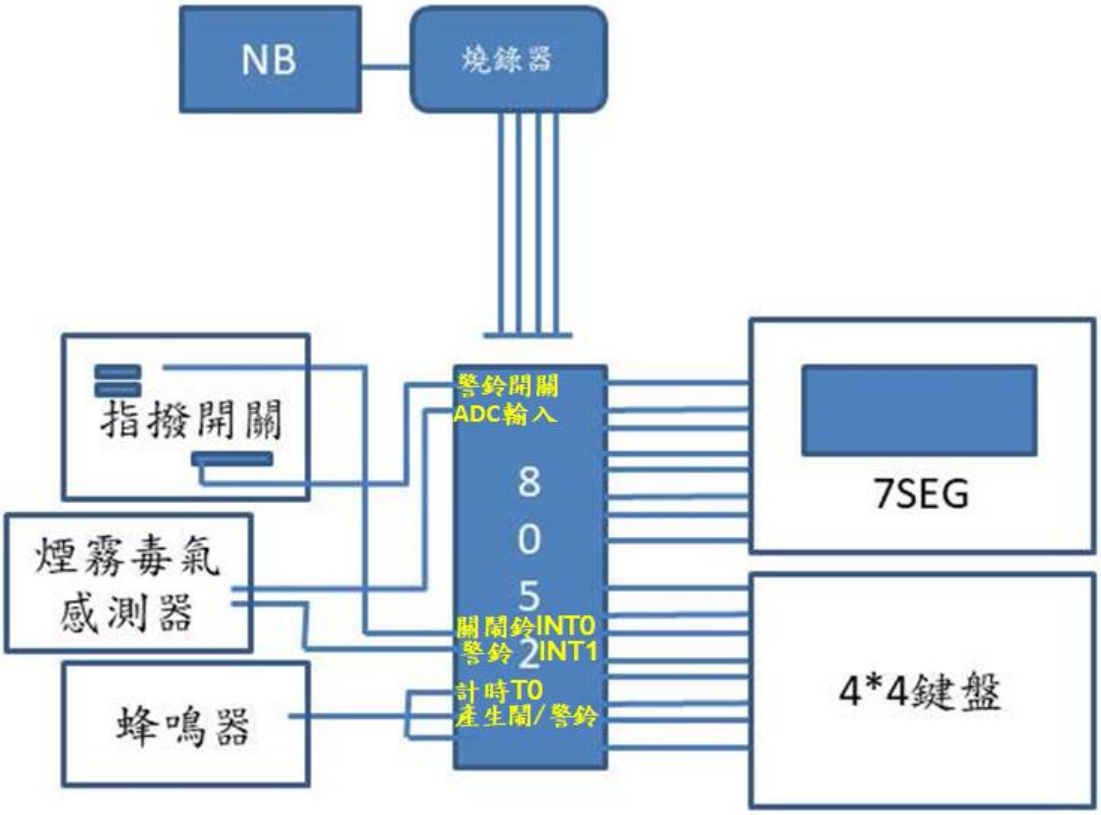
期末專題-毒氣檢測電子鬧鐘

姓名：王思敏
系級：醫工四
學號：C043001
上課時間：2 EF,JK

一、專題名稱：

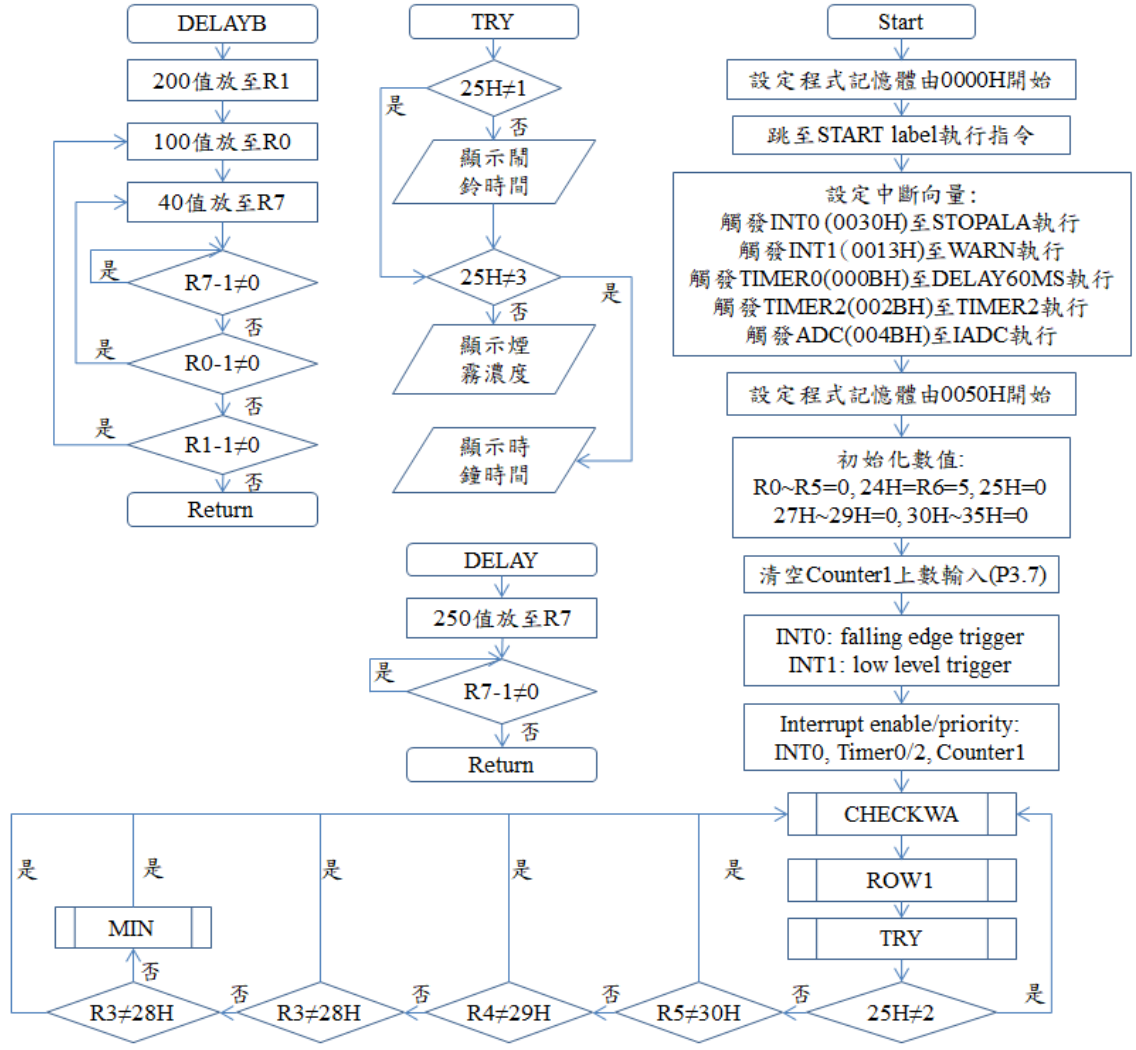
毒氣檢測電子鬧鐘

二、硬體接線：

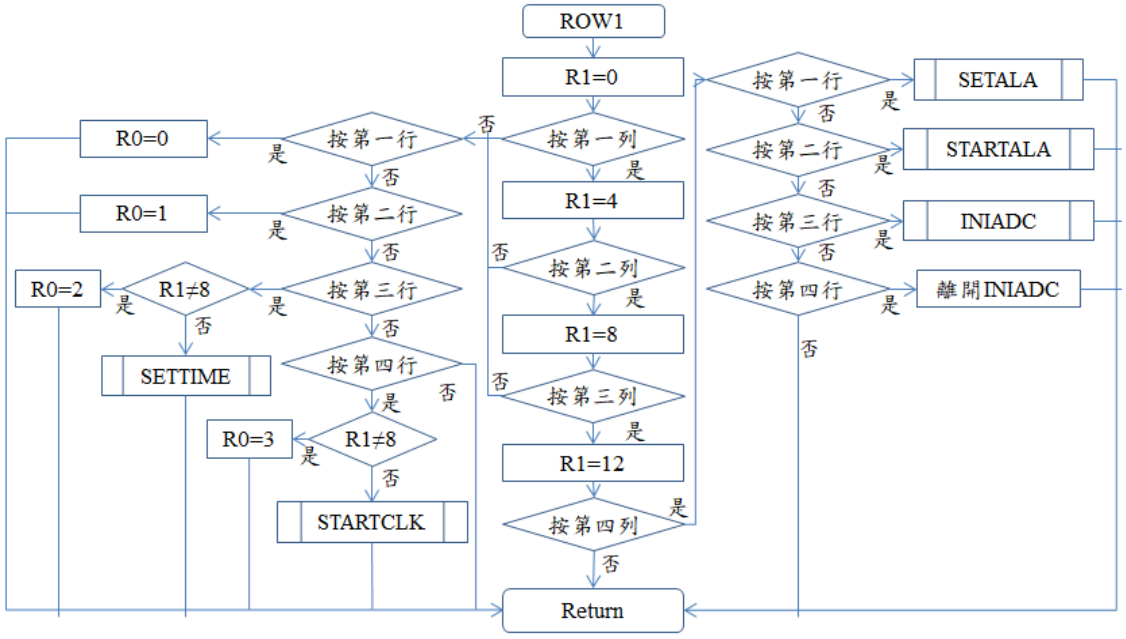


三、程式流程圖：

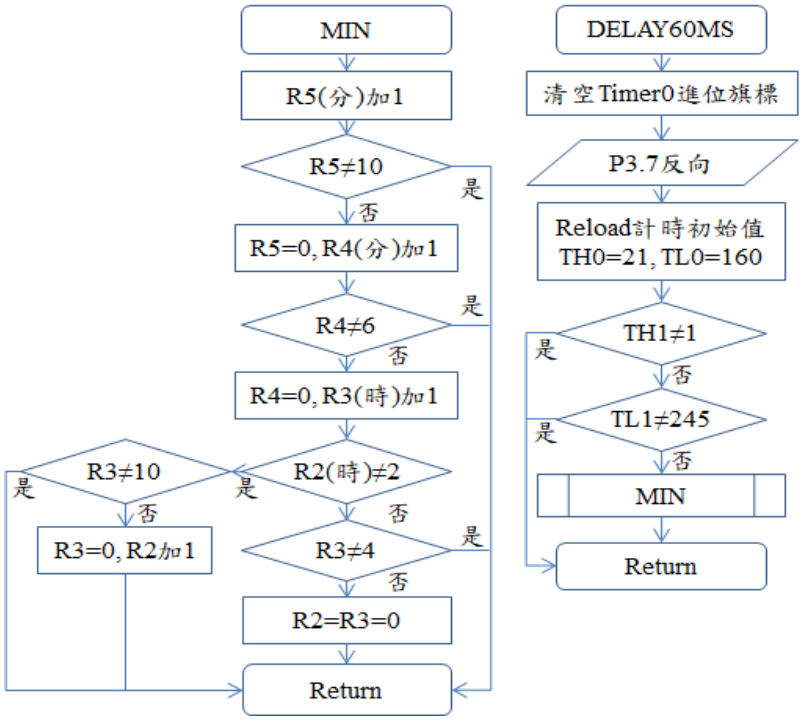
主程式、延遲副程式(長、短)、顯示副程式:



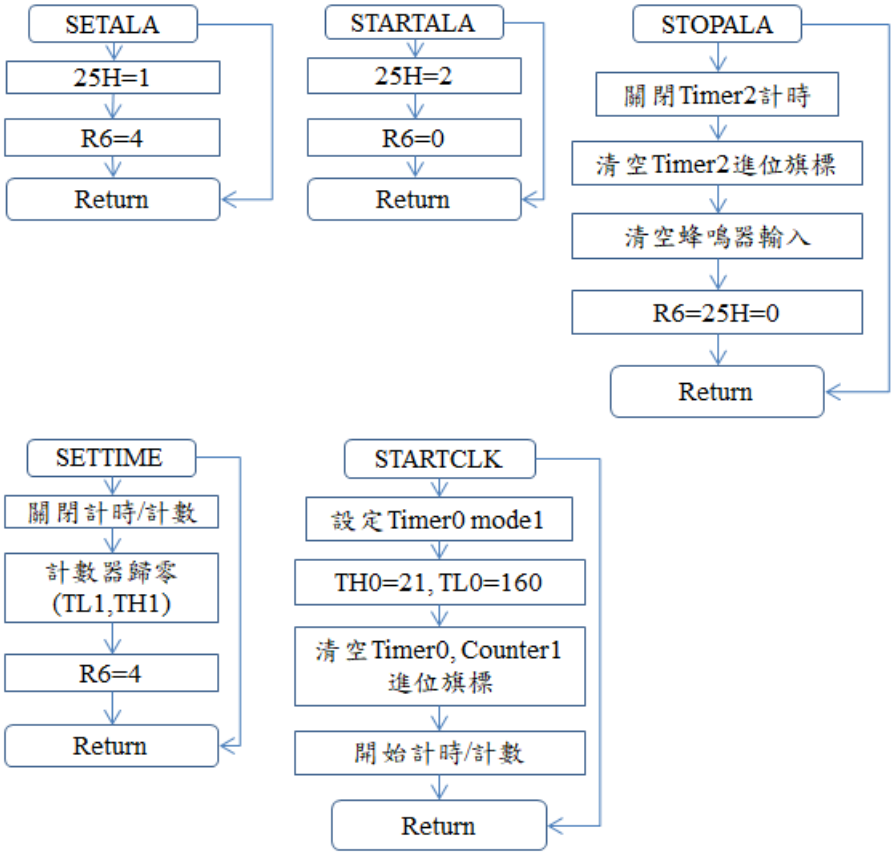
4*4 鍵盤副程式:



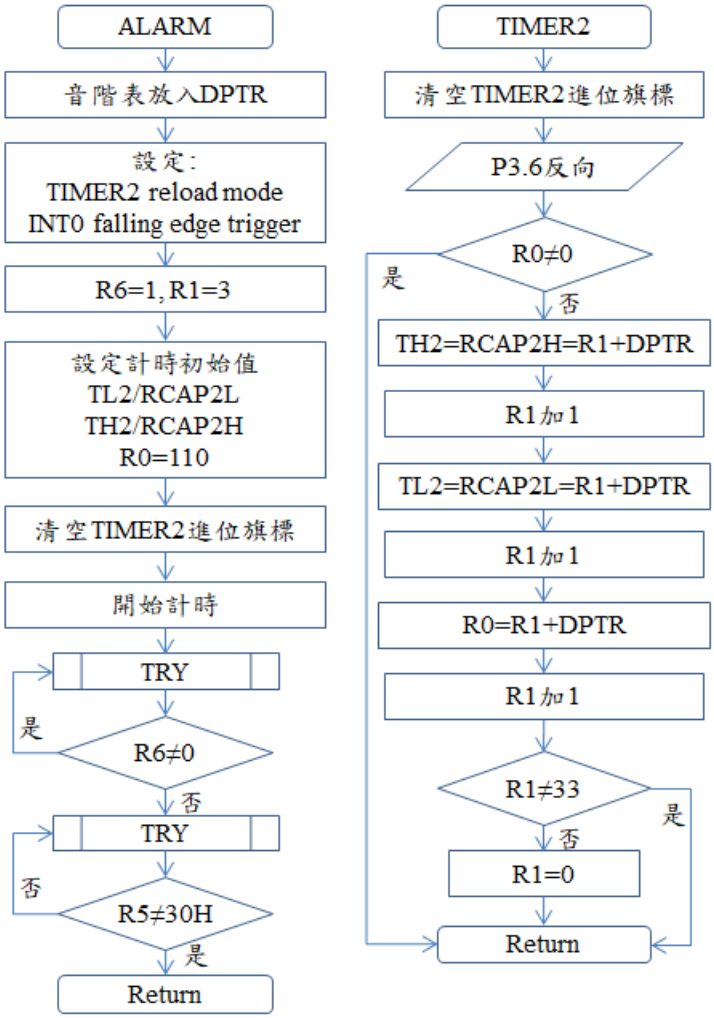
時鐘計時&計算副程式:



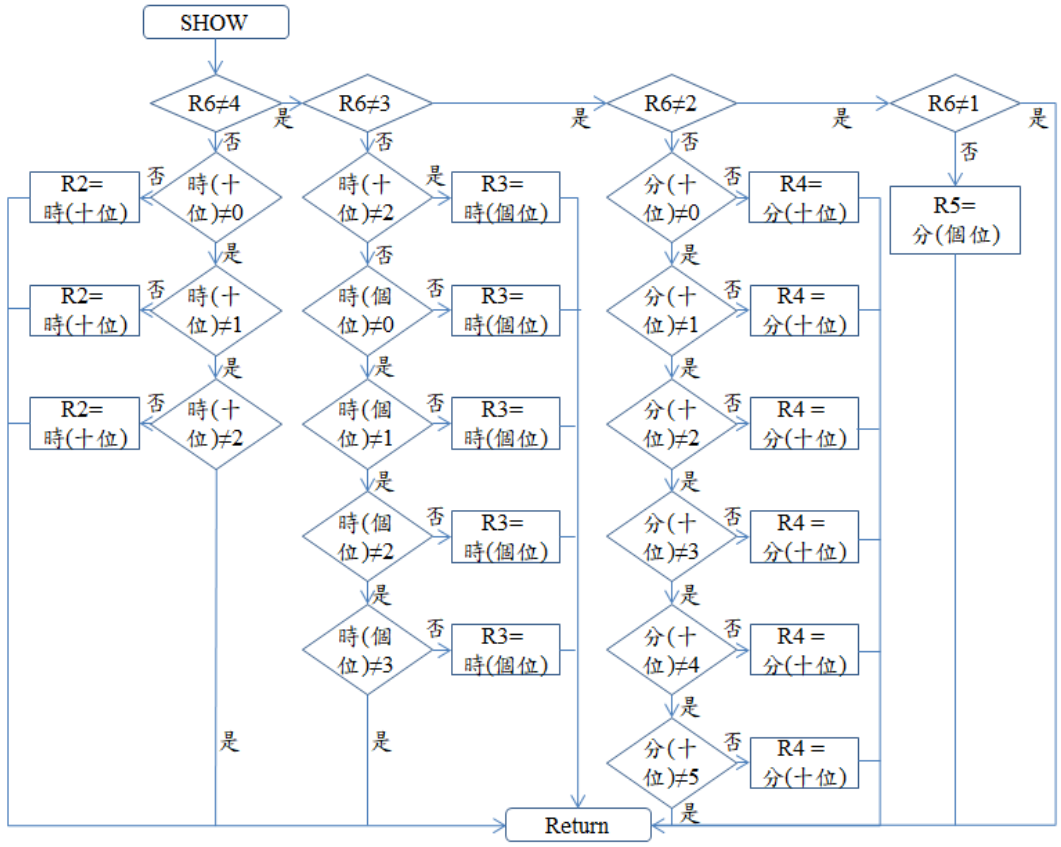
開啟時間/鬧鈴設定&開始計時&關閉鬧鈴副程式:



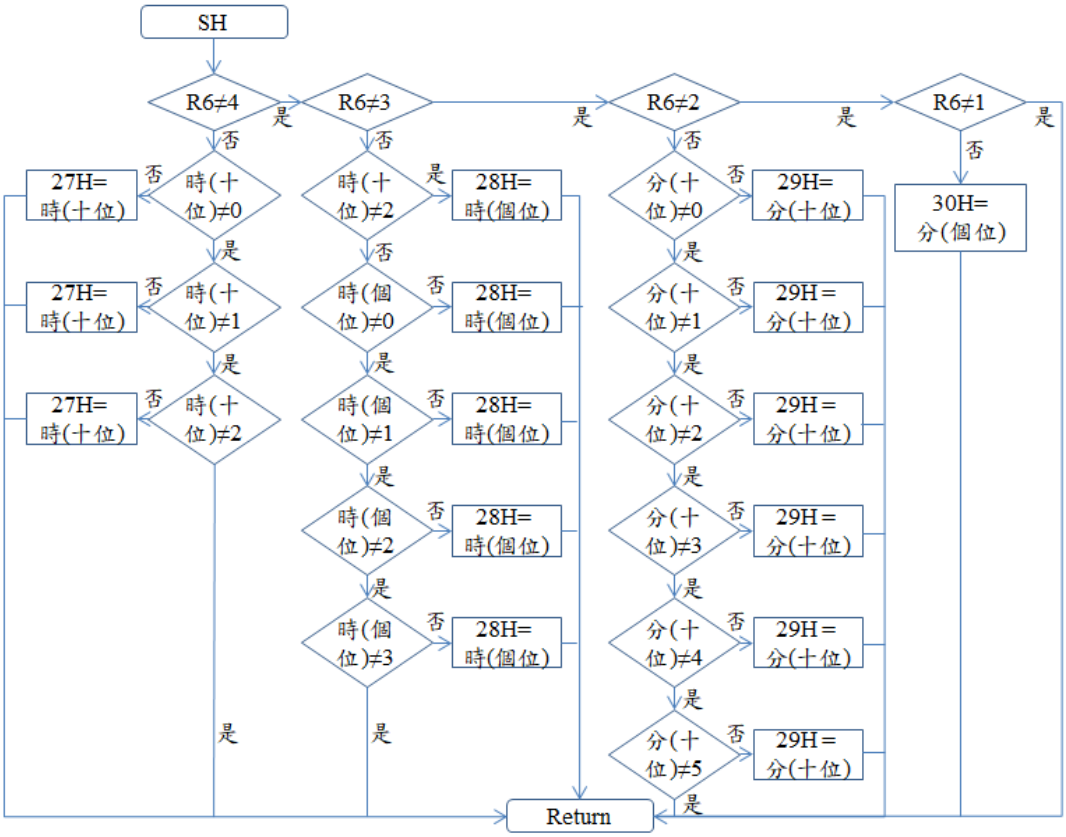
鬧鈴副程式:


ALARM
音階表放入DPTR
設定:
TIMER2 reload mode
INT0 falling edge trigger
R6=1, R1=3
設定計時初始值
TL2/RCAP2L
TH2/RCAP2H
R0=110
清空TIMER2進位旗標
開始計時
TRY
是
R6≠0
否
TRY
否
R5≠30H
是
Return
TIMER2
清空TIMER2進位旗標
P3.6反向
R0≠0
是
否
TH2=RCAP2H=R1+DPTR
R1加1
TL2=RCAP2L=R1+DPTR
R1加1
R0=R1+DPTR
R1加1
R1≠33
是
否
R1=0
Return

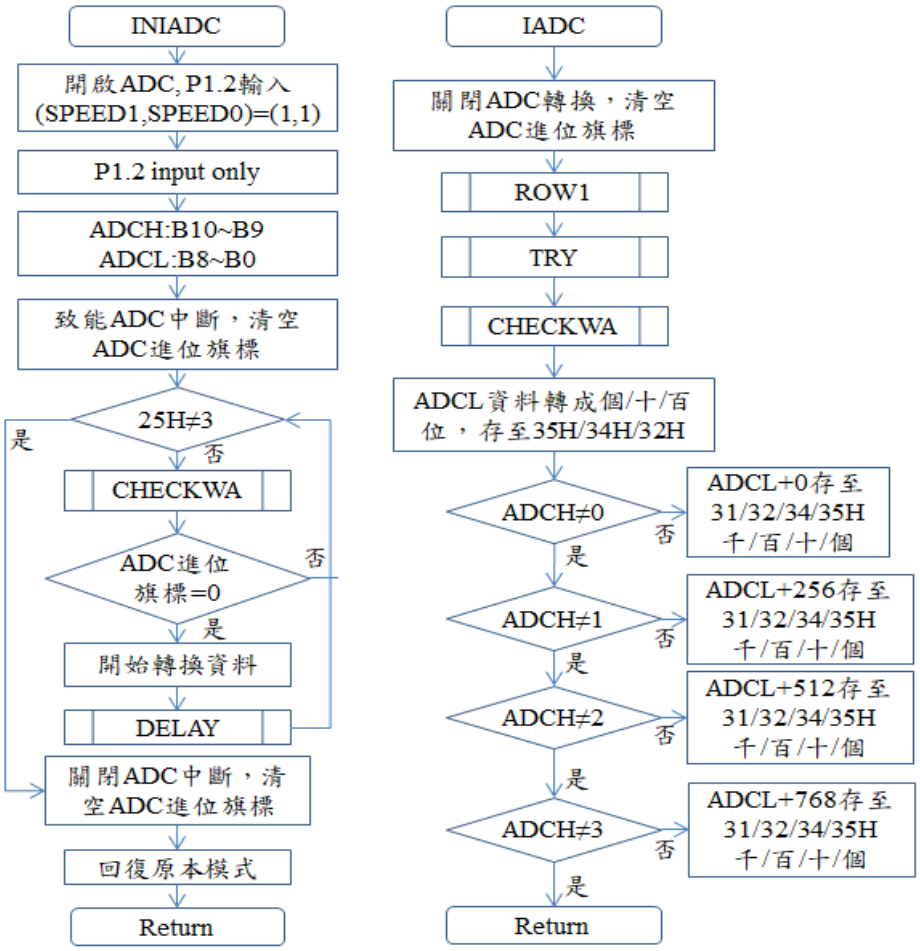
判斷時鐘時間輸入副程式:



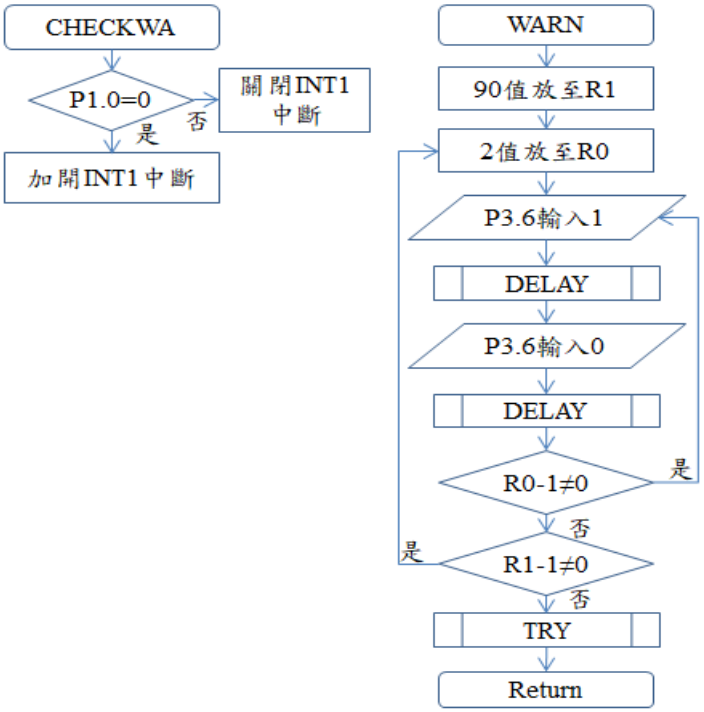
判斷鬧鈴時間輸入副程式:



類比輸入轉數位副程式:



煙霧警示鈴開啟判斷&煙霧警示鈴副程式:

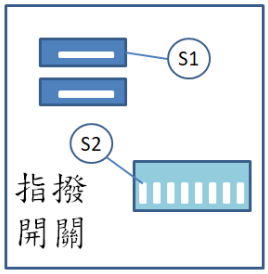


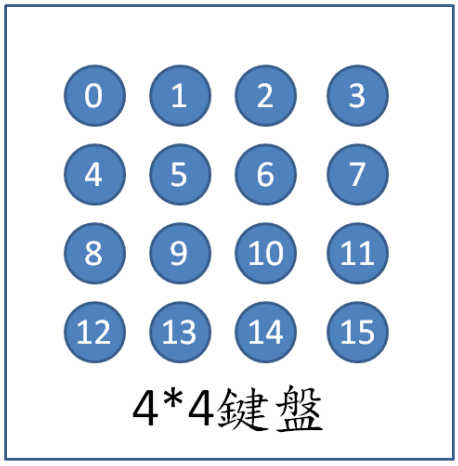
四、功能介紹：

期末專題的內容:毒氣檢測電子鬧鐘

共有三種模式:

1. 純粹作為時鐘使用
2. 作為鬧鐘使用
3. 煙霧或毒氣濃度檢測，前 2 種模式下皆能使用此功能(除鬧鈴正在鈴響時)
   (可偵測氨氣、氮氧化合物、醇類、芳族化合物、硫化物、煙霧等)





使用說明:

1. 4*4 鍵盤:
   a. 按鍵 0~9: 時鐘/鬧鈴時間設定輸入，可輸入時間為 00:00~23:59
      (其餘輸入法 ex.25:00、22:70 設定為無法輸入)
   b. 按鍵 10: 時鐘時間設定(同時停止時鐘功能)
   c. 按鍵 11: 開始時鐘功能
   d. 按鍵 12: 鬧鈴時間設定
   e. 按鍵 13: 開啟鬧鐘功能
   f. 按鍵 14: 開啟煙霧濃度檢測功能
      (此功能下不可使用 0~9 按鍵)
   g. 按鍵 15: 回到原本模式(時鐘 or 鬧鐘)

濃度轉換:
電壓輸入: 每提高 0.1V
增加 20ppm

$$ADC\ result = 1024 * \frac{Vin}{VDD}$$

$$20.48 = 1024 * \frac{0.1V}{5V}$$

2. 指撥開關:
   a. 彈跳開關 S1: 關掉鬧鈴使用
      *若於一分鐘內關閉鬧鈴，至下一分鐘為止無法使用鍵盤
      *只要壓下開關便可關閉鬧鈴，若不壓開關即便超過一分鐘鬧鈴也會繼續響
   b. 指撥開關 S2: 開啟煙霧警示鈴響功能
      *關閉時當煙霧/毒氣超標會發出警鈴
      *開啟時當煙霧/毒氣超標不會發出警鈴

五、程式碼與註解：

```
$INCLUDE   (REG_MPC82G516.inc)   //引入變數名稱位址定義檔案
      ORG      00H
      JMP START                          //程式由 START 開始執行
                                         //中斷設定
      ORG      03H                       //INT0 中斷向量
      JMP STOPALA                        //INT0 觸發至 STOPALA 執行
      ORG      13H                        //INT1 中斷向量
      JMP WARN                           //INT1 觸發至 WARN 執行
      ORG      2BH                       //TIMER2 中斷向量
      JMP TIMER2                         //TIMER2 進位旗標變 1 至 TIMER2 執行
      ORG      0BH                       // TIMER1 中斷向量
      JMP DELAY60MS                  //TIMER1 進位旗標變 1 至 DELAY60MS 執行
      ORG      4BH                       //類比輸入轉數位中斷向量
      JMP IADC                           //轉換完至 IDAC 執行
      ORG      50H                        //程式記憶體由 50H 開始
START:                                   //初始設定
      MOV      R0,#0                      //R0~R1 暫存器存放 0(計數)
      MOV      R1,#0
      MOV      R2,#0                      //R2~R5 暫存器存放 0(存時鐘時間)
      MOV      R3,#0
      MOV      R4,#0
      MOV      R5,#0
      MOV      R6,#5                      //R6 暫存器存放 5(計數)
      MOV      25H,#0                     //25H 存放 0(控制鬧鐘使用模式)
      MOV      27H,#0                     //27H~30H 存放 0(存放鬧鈴時間)
      MOV      28H,#0
      MOV      29H,#0
      MOV      30H,#0
      MOV      31H,#0                 //31H,32H,34H,35H 存放 0(存放空氣檢測值)
      MOV      32H,#0
      MOV      33H,#0                     //33H 存放 0(暫存數值)
      MOV      34H,#0
      MOV      35H,#0
      MOV      24H,#5                     //24H 存放 0(暫存數值)
      CLRP3.7                            //清空 Counter1 上數輸出(接至 IT1)
      SETB     IT0                       //INT0 設定 Falling edge trigger
```

```
    CLRIT1                          //INT1 設定 Low level trigger
    MOV     IE,#10101011B           //致能 EA 與 Timer0, 1, 2, INT0
    MOV     IP,#00101011B           //設定優先序
BEGIN:                              //主程式迴圈
    CALL    CHECKWA                 //呼叫 CHECKWA(開啟煙霧警鈴副程式)
    CALL    ROW1                    //呼叫 ROW1(鍵盤副程式)
    CALL    TRY                     //呼叫 TRY(七段顯示器副程式)
    MOV     A,25H
    CJNE    A,#2,BEGIN              //若 25H 為 2 開啟鬧鈴時間判斷
    MOV     A,R5                    //鬧鈴分(個位)
    CJNE    A,30H,BEGIN
    MOV     A,R4                    //鬧鈴分(十位)
    CJNE    A,29H,BEGIN
    MOV     A,R3                    //鬧鈴時(個位)
    CJNE    A,28H,BEGIN
    MOV     A,R2                    //鬧鈴時(十位)
    CJNE    A,27H,BEGIN
    CALL    ALARM                   //以上皆相等呼叫 ALARM(鬧鈴副程式)
    JMP BEGIN                       //回 BEGIN label
DELAY60MS:                          //Timer0 時鐘計時中斷副程式
    CLRTF0                          //清空 Timer0 進位旗標
    CPL P3.7                        //反向 Counter1 輸入
    MOV     TH0,#21                 //重新存入計時初始值(單次 60ms)
    MOV     TL0,#160
    MOV     33H,R7
    MOV     R7,TH1                  //判斷計時時間是否達 1 分鐘(上數 500 次)
    CJNE    R7,#1,BACKDELAY
    MOV     R7,TL1
    CJNE    R7,#245,BACKDELAY
    MOV     TL1,#0                  //上數達 500 次，計數初始值設回 0
    MOV     TH1,#0
    MOV     R7,33H
    CALL    MIN                     //呼叫 MIN(時間計算副程式)
BACKDELAY:
    RETI                            //回主程式
MIN:                                //時間上加一分鐘副程式
    INC R5                          //分(個位)加 1
    CJNE    R5,#10,GOBACK
```

```
    MOV     R5,#0                   //分(個位)達 10 則歸零，且分(十位)加 1
    INC R4
    CJNE    R4,#6,GOBACK
    MOV     R4,#0                   //分(十位)達 6 則歸零，且時(個位)加 1
    INC R3
    CJNE    R2,#2,HOUR              //若時(十位)≠2 則至 HOUR 執行
    CJNE    R3,#4,GOBACK   //當時(十位)=2 且時(個位)≠4 至 GOBACK 執行
    MOV     R3,#0                  //當時(十位)=2 且時(個位)=4，則時全部歸 0
    MOV     R2,#0
    JMP GOBACK
HOUR:
    CJNE    R3,#10,GOBACK   //時(十位)≠2 且時(個位) ≠10 至 GOBACK 執行
    MOV     R3,#0    //時(十位)≠2 且時(個位) =10，時(個位)歸零，時(十位)加 1
    INC R2
GOBACK:
    RET                             //回主程式
ROW1:                               //4*4 按鍵輸入副程式
    MOV     P2,#01111111B           //若有按第一列按鍵則至 COL1 執行
    CALL    DELAY
    MOV     A,P2
    ANL     A,#0FH
    MOV     R1,#0                    //R1 存放 0
    CJNE    A,#0FH,COL1
ROW2:
    MOV     P2,#10111111B            //若有按第二列按鍵則至 COL1 執行
    CALL    DELAY
    MOV     A,P2
    ANL     A,#0FH
    MOV     R1,#4                    //R1 存放 4
    CJNE    A,#0FH,COL1
ROW3:
    MOV     P2,#11011111B            //若有按第三列按鍵則至 COL1 執行
    CALL    DELAY
    MOV     A,P2
    ANL     A,#0FH
    MOV     R1,#8                    //R1 存放 8
    CJNE    A,#0FH,COL1
ROW4:
```

```
    MOV     P2,#11101111B               //若有按第四列按鍵則至 CO1 執行
    CALL    DELAY
    MOV     A,P2
    ANL     A,#0FH
    MOV     R1,#12                      //R1 存放 12
    CJNE    A,#0FH,CO1
    JMP BACKSHOW
COL1:
    CJNE    A,#0EH,COL2                 //若未按第一行按鍵則至 COL2 執行
    MOV     R0,#0                       //有按第一行按鍵，R0 存放 0
    JMP GOSHOW
COL2:
    CJNE    A,#0DH,COL3                 //若未按第二行按鍵則至 COL3 執行
    MOV     R0,#1                       //有按第二行按鍵，R0 存放 1
    JMP GOSHOW
COL3:
    CJNE    A,#0BH,COL4                 //若未按第三行按鍵則至 COL4 執行
    CJNE    R1,#8,CO3                   //若非第三列按鍵則至 CO3 執行
    CALL    SETTIME          //按第三列按鍵，呼叫 SETTIME(設定時鐘時間)
    JMP BACKSHOW
COL4:
    CJNE    A,#07H,BACKSHOW    //未按第四行按鍵則至 BACKSHOW 執行
    CJNE    R1,#8,CO4                   //若非第三列按鍵則至 CO4 執行
    CALL    STARTCLK         //按第三列按鍵，呼叫 STARKCLK(時鐘開始)
    JMP BACKSHOW
CO1:
    CJNE    A,#0EH,CO1_1           //按第四列，若非第一行則至 CO1_1 執行
    CALL    SETALA           //按第四列第一行，呼叫 SETALA(設定鬧鐘時間)
    JMP BACKSHOW
CO1_1:
    CJNE    A,#0DH,CO1_2           //按第四列，若非第二行則至 CO1_2 執行
    CALL    STARTALA    //按第四列第一行，呼叫 STARTALA(開啟鬧鈴判斷)
    JMP BACKSHOW
CO1_2:
    CJNE    A,#0BH,CO1_3           //按第四列，若非第三行則至 CO1_3 執行
    MOV     A,24H
    CJNE    A,25H,CO1_2_1      //若當前不為空氣檢測模式則至 CO1_2_1 執行
    JMP BACKSHOW
```

```
CO1_2_1:
    MOV     24H,25H
    MOV     25H,#3                  //25H 存放 3(空氣檢測)
    CALL    DELAYB                  //呼叫 DELAYB(延遲副程式)
    CALL    INIADC                   //呼叫 INIADC(空氣檢測副程式)
    JMP BACKSHOW
CO1_3:
    CJNE    A,#07H,BACKSHOW   //按第四列，非第四行至 BACKSHOW 執行
    MOV     25H,24H                  //從空氣檢測模式回原本模式
    CALL    DELAYB                  //呼叫 DELAYB(延遲副程式)
    JMP BACKSHOW
CO3:
    MOV     R0,#2                    //非第三列按鍵，R0 存放 2
    JMP GOSHOW
CO4:
    MOV     R0,#3                    //非第三列按鍵，R0 存放 3
    JMP GOSHOW
GOSHOW:
    CJNE    R6,#0,GOSHOW1          //計數器 R6≠0 則至 GOSHOW1 執行
    JMP BACKSHOW
GOSHOW1:
    MOV     R7,25H
    CJNE    R7,#1,GOSHOW2
    CALL    SH      //若 25H(控制鬧鐘模式)=1 則呼叫 SH(鍵入鬧鈴時間副程式)
    JMP BACKSHOW
GOSHOW2:
    CJNE    R7,#3,GOSHOW3       //25H=3(顯示空氣煙霧濃度中，跳回主程式)
    JMP BACKSHOW
GOSHOW3:
    CALL    SHOW              //25H≠1,3 則呼叫 SHOW(鍵入時鐘時間副程式)
BACKSHOW:
    RET                             //回主程式
SHOW:                               //輸入時鐘時間副程式
    CJNE    R6,#4,SHOW1
    MOV     A,R1
    ADD     A,R0
    CJNE    A,#0,SHOW_1        //R6=4，若時(十位)=0 輸入時(十位)
    MOV     R2,A
```

```
    DEC     R6                  //計數值減 1
    CALL    DELAYB              //延時防止按鍵彈跳
    JMP BACK
SHOW_1:
    CJNE    A,#1,SHOW_2         //R6=4，若時(十位)=1 輸入時(十位)
    MOV     R2,A
    DEC     R6                  //計數值減 1
    CALL    DELAYB              //延時防止按鍵彈跳
    JMP BACK
SHOW_2:
    CJNE    A,#2,BACKBACK       //R6=4，若時(十位)=2 輸入時(十位)
    MOV     R2,A
    DEC     R6                  //計數值減 1
    CALL    DELAYB              //延時防止按鍵彈跳
    JMP BACK
SHOW1:
    CJNE    R6,#3,SHOW2
    MOV     A,R1
    ADD     A,R0
    CJNE    R2,#2,SHOW1_1_1
    CJNE    A,#0,SHOW1_1        //R6=3，若時(十位)=2 且時(個位)=0 則輸入
    MOV     R3,A
    DEC     R6                  //計數值減 1
    CALL    DELAYB              //延時防止按鍵彈跳
    JMP BACK
SHOW1_1:
    CJNE    A,#1,SHOW1_2        //R6=3，若時(十位)=2 且時(個位)=1 則輸入
    MOV     R3,A
    DEC     R6                  //計數值減 1
    CALL    DELAYB              //延時防止按鍵彈跳
    JMP BACK
SHOW1_2:
    CJNE    A,#2,SHOW1_3        //R6=3，若時(十位)=2 且時(個位)=2 則輸入
    MOV     R3,A
    DEC     R6                  //計數值減 1
    CALL    DELAYB              //延時防止按鍵彈跳
    JMP BACK
SHOW1_3:
```

```
    CJNE    A,#3,BACKBACK       //R6=3，若時(十位)=2 且時(個位)=3 則輸入
    MOV     R3,A
    DEC     R6                  //計數值減 1
    CALL    DELAYB              //延時防止按鍵彈跳
    JMP BACK
BACKBACK:
    JMP BACK
SHOW1_1_1:
    MOV     R3,A                //R6=3，若時(十位)≠2 則輸入時(個位)
    DEC     R6                  //計數值減 1
    CALL    DELAYB              //延時防止按鍵彈跳
    JMP BACK
SHOW2:
    CJNE    R6,#2,SHOW3
    MOV     A,R1
    ADD     A,R0
    CJNE    A,#0,SHOW2_1        //R6=2，若分(十位)=0 則輸入分(十位)
    MOV     R4,A
    DEC     R6                  //計數值減 1
    CALL    DELAYB              //延時防止按鍵彈跳
    JMP BACK
SHOW2_1:
    CJNE    A,#1,SHOW2_2        //R6=2，若分(十位)=1 則輸入分(十位)
    MOV     R4,A
    DEC     R6                  //計數值減 1
    CALL    DELAYB              //延時防止按鍵彈跳
    JMP BACK
SHOW2_2:
    CJNE    A,#2,SHOW2_3        //R6=2，若分(十位)=2 則輸入分(十位)
    MOV     R4,A
    DEC     R6                  //計數值減 1
    CALL    DELAYB              //延時防止按鍵彈跳
    JMP BACK
SHOW2_3:
    CJNE    A,#3,SHOW2_4        //R6=2，若分(十位)=3 則輸入分(十位)
    MOV     R4,A
    DEC     R6                  //計數值減 1
    CALL    DELAYB              //延時防止按鍵彈跳
```

```
        JMP BACK
SHOW2_4:
        CJNE    A,#4,SHOW2_5        //R6=2，若分(十位)=4 則輸入分(十位)
        MOV     R4,A
        DEC     R6                  //計數值減 1
        CALL    DELAYB              //延時防止按鍵彈跳
        JMP BACK
SHOW2_5:
        CJNE    A,#5,BACK           //R6=2，若分(十位)=5 則輸入分(十位)
        MOV     R4,A
        DEC     R6                  //計數值減 1
        CALL    DELAYB              //延時防止按鍵彈跳
        JMP BACK
SHOW3:
        CJNE    R6,#1,BACK
        MOV     A,R1
        ADD     A,R0
        MOV     R5,A
        DEC     R6                  //計數值減 1
        CALL    DELAYB              //延時防止按鍵彈跳
        JMP BACK
BACK:
        RET                         //回主程式
SH:                                 //輸入鬧鈴時間副程式
        CJNE    R6,#4,SH1
        MOV     A,R1
        ADD     A,R0
        CJNE    A,#0,SH_1           //R6=4，若時(十位)=0 輸入時(十位)
        MOV     27H,A
        DEC     R6                  //計數值減 1
        CALL    DELAYB              //延時防止按鍵彈跳
        JMP BA
SH_1:
        CJNE    A,#1,SH_2           //R6=4，若時(十位)=1 輸入時(十位)
        MOV     27H,A
        DEC     R6                  //計數值減 1
        CALL    DELAYB              //延時防止按鍵彈跳
        JMP BA
```

```
SH_2:
    CJNE    A,#2,BABA           //R6=4，若時(十位)=2 輸入時(十位)
    MOV     27H,A
    DEC     R6                  //計數值減 1
    CALL    DELAYB              //延時防止按鍵彈跳
    JMP BA
SH1:
    CJNE    R6,#3,SH2
    MOV     A,R1
    ADD     A,R0
    MOV     R7,27H
    CJNE    R7,#2,SH1_1_1
    CJNE    A,#0,SH1_1          //R6=3，若時(十位)=2 且時(個位)=0 則輸入
    MOV     28H,A
    DEC     R6                  //計數值減 1
    CALL    DELAYB              //延時防止按鍵彈跳
    JMP BA
SH1_1:
    CJNE    A,#1,SH1_2          //R6=3，若時(十位)=2 且時(個位)=1 則輸入
    MOV     28H,A
    DEC     R6                  //計數值減 1
    CALL    DELAYB              //延時防止按鍵彈跳
    JMP BA
SH1_2:
    CJNE    A,#2,SH1_3          //R6=3，若時(十位)=2 且時(個位)=2 則輸入
    MOV     28H,A
    DEC     R6                  //計數值減 1
    CALL    DELAYB              //延時防止按鍵彈跳
    JMP BA
SH1_3:
    CJNE    A,#3,BABA           //R6=3，若時(十位)=2 且時(個位)=3 則輸入
    MOV     28H,A
    DEC     R6                  //計數值減 1
    CALL    DELAYB              //延時防止按鍵彈跳
    JMP BA
BABA:
    JMP BA
SH1_1_1:
```

```
    MOV     28H,A           //R6=3，若時(十位)≠2 則輸入時(個位)
    DEC     R6                  //計數值減 1
    CALL    DELAYB              //延時防止按鍵彈跳
    JMP BA
SH2:
    CJNE    R6,#2,SH3
    MOV     A,R1
    ADD     A,R0
    CJNE    A,#0,SH2_1          //R6=2，若分(十位)=0 則輸入分(十位)
    MOV     29H,A
    DEC     R6                  //計數值減 1
    CALL    DELAYB              //延時防止按鍵彈跳
    JMP BA
SH2_1:
    CJNE    A,#1,SH2_2          //R6=2，若分(十位)=1 則輸入分(十位)
    MOV     29H,A
    DEC     R6                  //計數值減 1
    CALL    DELAYB              //延時防止按鍵彈跳
    JMP BA
SH2_2:
    CJNE    A,#2,SH2_3          //R6=2，若分(十位)=2 則輸入分(十位)
    MOV     29H,A
    DEC     R6                  //計數值減 1
    CALL    DELAYB              //延時防止按鍵彈跳
    JMP BA
SH2_3:
    CJNE    A,#3,SH2_4          //R6=2，若分(十位)=3 則輸入分(十位)
    MOV     29H,A
    DEC     R6                  //計數值減 1
    CALL    DELAYB              //延時防止按鍵彈跳
    JMP BA
SH2_4:
    CJNE    A,#4,SH2_5          //R6=2，若分(十位)=4 則輸入分(十位)
    MOV     29H,A
    DEC     R6                  //計數值減 1
    CALL    DELAYB              //延時防止按鍵彈跳
    JMP BA
SH2_5:
```

```
    CJNE    A,#5,BA          //R6=2，若分(十位)=5 則輸入分(十位)
    MOV     29H,A
    DEC     R6               //計數值減 1
    CALL    DELAYB           //延時防止按鍵彈跳
    JMP BA
SH3:
    CJNE    R6,#1,BA         //R6=1，則輸入分(個位)
    MOV     A,R1
    ADD     A,R0
    MOV     30H,A
    DEC     R6               //計數值減 1
    CALL    DELAYB           //延時防止按鍵彈跳
    JMP BA
BA:
    RET                      //回主程式
TRY:                         //七段顯示器顯示副程式
    MOV     36H,A
    MOV     R7,25H
    CJNE    R7,#1,TRY1       //25H=1 則顯示鬧鈴時間
    MOV     A,30H            //顯示分(個位)
    ADD     A,#01110000B
    MOV     P0,A
    CALL    DELAY
    MOV     A,29H            //顯示分(十位)
    ADD     A,#10110000B
    MOV     P0,A
    CALL    DELAY
    MOV     A,28H            //顯示時(個位)
    ADD     A,#11010000B
    MOV     P0,A
    CALL    DELAY
    MOV     A,27H            //顯示時(十位)
    ADD     A,#11100000B
    MOV     P0,A
    CALL    DELAY
    JMP BACKTRY
TRY1:
    CJNE    R7,#3,TRY2       //25H=3 則顯示空氣煙霧濃度
```

```
    MOV     R6,#255
TRY1_1:
    MOV     A,35H               //顯示濃度(個位)
    ADD     A,#01110000B
    MOV     P0,A
    CALL    DELAY
    MOV     A,34H               //顯示濃度(十位)
    ADD     A,#10110000B
    MOV     P0,A
    CALL    DELAY
    MOV     A,32H               //顯示濃度(百位)
    ADD     A,#11010000B
    MOV     P0,A
    CALL    DELAY
    MOV     A,31H               //顯示濃度(千位)
    ADD     A,#11100000B
    MOV     P0,A
    CALL    DELAY
    DJNZ    R6,TRY1_1
    MOV     R6,#0
    JMP BACKTRY
TRY2:
    MOV     A,#01110000B          //25H≠1,3 則顯示時鐘時間
    ADD     A,R5                  //顯示分(個位)
    MOV     P0,A
    CALL    DELAY
    MOV     A,#10110000B
    ADD     A,R4                  //顯示分(十位)
    MOV     P0,A
    CALL    DELAY
    MOV     A,#11010000B
    ADD     A,R3                  //顯示時(個位)
    MOV     P0,A
    CALL    DELAY
    MOV     A,#11100000B
    ADD     A,R2                  //顯示時(十位)
    MOV     P0,A
    CALL    DELAY
```

```
BACKTRY:
    MOV     A,36H
    RET
SETTIME:                        //開啟設定時鐘時間(同時停止時鐘功能)副程式
    CLRTR0                      //停止計時
    CLRTF0
    CLRP3.7         //TO P3.4
    MOV     TL1,#0              //計數器歸零
    MOV     TH1,#0
    MOV     R6,#4               //鍵入時間/鬧鈴計數器=4
    RET
STARTCLK:                       //開啟時鐘功能副程式
    MOV     TMOD,#01010001B     //設定 Timer0 mode 1, Counter1 mode 1
    MOV     TH0,#21             //設定計時初始值
    MOV     TL0,#160
    MOV     R6,#0               //鍵入時間/鬧鈴計數器=0
    CLRTF1                      //清空 Counter1 進位旗標
    CLRTF0                      //清空 Timer0 進位旗標
    SETB    TR0                 //開始計時
    SETB    TR1                 //開始計數
    RET                         //回主程式
DELAY:                          //延遲副程式(短)
    MOV     R7,#250
DELAY2:
    DJNZ    R7,DELAY2
    RET
DELAYB:                         //延遲副程式(長)
    MOV     R1,#200
DELAYB1:
    MOV     R0,#100
DELAYB2:
    MOV     R7,#40
DELAYB3:
    DJNZ    R7,DELAYB3
    DJNZ    R0,DELAYB2
    DJNZ    R1,DELAYB1
    RET
CHECKWA:                        //警示鈴中斷設定副程式
```

```
    JB   90H,CHECKWA1
    MOV     IE,#10101111B          //P1.0=0 則開啟警示鈴中斷(INT1)
    MOV     IP,#00101111B
    JMP BACKCHE
CHECKWA1:
    MOV     IE,#10101011B          //P1.0=0 則關閉警示鈴中斷(INT1)
    MOV     IP,#00101011B
BACKCHE:
    RET                            //回主程式
SETALA:                            //開啟設定鬧鈴時間副程式
    MOV     25H,#1                 //25H 存放 1(設定鬧鈴時間模式)
    MOV     R6,#4
    RET
STARTALA:                          //開啟鬧鈴功能副程式
    MOV     25H,#2                 //25H 存放 2(開啟鬧鈴判斷模式)
    MOV     R6,#0
    RET
ALARM:                             //鬧鈴副程式
    MOV     DPTR,#SCALE            //鬧鈴 SCALE table 放進 DPTR
    MOV     T2CON,#00000000B       //設定 TIMER2 reload mode
    MOV     R6,#1
    MOV     R1,#3
    MOV     R0,#110
    SETB    IT0                    //設定 INT0 falling edge trigger
    MOV     TH2,#11110111B         //設定計時初始值
    MOV     TL2,#00011111B
    MOV     RCAP2H,#11110111B
    MOV     RCAP2L,#00011111B
    CLRTF2                         //清空 Timer2 進位旗標
    SETB    TR2                    //開始計時
LOOPALA:
    CALL    TRY                    //呼叫 TRY(顯示副程式)
    CJNE    R6,#0,LOOPALA          //R6≠0 則繼續 LOOPALA
LOOPALA1:
    CALL    TRY                    //呼叫 TRY(顯示副程式)
    MOV     A,R5
    CJNE    A,30H,BAALA        //若 R5(時鐘，分)≠30H(鬧鈴，分)則回主程式
    JMP LOOPALA1
```

```
BAALA:
      RET                                  //回主程式
TIMER2:                                    //鬧鈴中斷副程式
      CLRTF2                               //清空 Timer2 進位旗標
      CPL P3.6                             //蜂鳴器輸入反向
      DJNZ    R0,BACKALA                   //若 R0=0 則更新計時初始值與週期數
      MOV     A,R1
      MOVC    A,@A+DPTR
      MOV     TH2,A
      MOV     RCAP2H,A
      INC R1
      MOV     A,R1
      MOVC    A,@A+DPTR
      MOV     TL2,A
      MOV     RCAP2L,A
      INC R1
      MOV     A,R1
      MOVC    A,@A+DPTR
      MOV     R0,A
      INC R1
      CJNE    R1,#33,BACKALA
      MOV     R1,#0
BACKALA:
      RETI                                 //回主程式
STOPALA:                                   //停止鬧鈴中斷副程式
      CLRTR2                               //關閉 Timer2 計時
      CLRTF2                               //清空 Timer2 進位旗標
      CLRP3.6                              //清空蜂鳴器輸入
      MOV     R6,#0
      MOV     25H,#0          //25H 歸零(故要重新按開啟鬧鈴才會有鬧鈴功能)
      RETI                                 //回主程式
INIADC:                                    //開啟類比轉數位副程式
      MOV     ADCTL,#0E2H   //開啟 ADC, (SPEED1,SPEED0)=(1,1), P1.2 輸入
      ORL     P1M0,#00000100B          //P1M0,bit2=1，使 P1.2 只允許輸入
      ANL     P1M1,#11111011B          //P1M1,bit2=0
      ORL     AUXR,#01000000B        //ADRJ=1(ADCH:B10~B9, ADCL:B8~B0)
      ORL     AUXIE,#00000010B         //致能 ADC
      ANL     ADCTL,#11101111B         //清空 ADC 進位旗標
```

```
        ORL     ADCTL,#00001000B        //開始轉換資料
        CALL    DELAY                   //延遲副程式
LOOPADC:
        MOV     R7,25H
        CJNE    R7,#3,BAADC             //若 25H=3 則繼續 ADC 迴圈
        CALL CHECKWA                    //呼叫 CHECKWA(警示鈴中斷副程式)
        MOV     22H,A
        MOV     A,ADCTL
        ANL     A,#00010000B
        CJNE    A,#00000000B,LOOPADC //確定已進過 ADC 中斷
        ORL     ADCTL,#00001000B        //開始轉換資料
        CALL    DELAY                   //延遲副程式
        JMP LOOPADC
BAADC:
        ANL     ADCTL,#11100111B        //停止轉換
        MOV     24H,#5                  //24H(存 25H 原本的模式)存回 5
        MOV     A,22H
        RET
IADC:                                   //類比轉數位中斷副程式
        ANL     ADCTL,#11100111B    //關閉 ADC 轉換資料，清空 ADC 進位旗標
        CALL    ROW1                    //呼叫 ROW1(鍵入副程式)
        CALL    TRY                     //呼叫 TRY(顯示副程式)
        CALL CHECKWA                    //呼叫 CHECKWA(警示鈴功能開關)
        MOV     A,ADCL                  //10-bit 類比資料轉換為個、十、百、千位
        MOV     B,#100
        DIV AB
        MOV     32H,A                   //32H 存百位
        MOV     A,B
        MOV     B,#10
        DIV AB
        MOV     34H,A                   //34H 存十位
        MOV     A,B
        MOV     35H,A                   //35H 存個位
        MOV     A,ADCH
ADC0:
        CJNE    A,#0,ADC1               //ADCH=0，則 ADCL+0
        MOV     31H,#0
        JMP BACKADC
```

```
ADC1:
    CJNE    A,#1,ADC2              //ADCH=1，則 ADCL+256
    MOV     A,35H
    CJNE    A,#0,ADC1_1            //個位(35H)加 6
    ADD     A,#6
    MOV     35H,A
    JMP ADC1_1_0
ADC1_1:
    CJNE    A,#1,ADC1_2            //個位(35H)加 6
    ADD     A,#6
    MOV     35H,A
    JMP ADC1_1_0
ADC1_2:
    CJNE    A,#2,ADC1_3            //個位(35H)加 6
    ADD     A,#6
    MOV     35H,A
    JMP ADC1_1_0
ADC1_3:
    CJNE    A,#3,ADC1_4            //個位(35H)加 6
    ADD     A,#6
    MOV     35H,A
    JMP ADC1_1_0
ADC1_4:
    ADD     A,#6                   //個位(35H)加 6 減 10
    SUBB    A,#10
    MOV     35H,A
    MOV     A,34H                  //十位(34H)進位
    ADD     A,#1
    MOV     34H,A
ADC1_1_0:
    MOV     A,34H                  //十位(34H)加 5
    CJNE    A,#0,ADC1_1_1
    ADD     A,#5
    MOV     34H,A
    JMP ADC1_B
ADC1_1_1:
    CJNE    A,#1,ADC1_1_2          //十位(34H)加 5
    ADD     A,#5
```

```
    MOV     34H,A
    JMP ADC1_B
ADC1_1_2:
    CJNE    A,#2,ADC1_1_3           //十位(34H)加 5
    ADD     A,#5
    MOV     34H,A
    JMP ADC1_B
ADC1_1_3:
    CJNE    A,#3,ADC1_1_4           //十位(34H)加 5
    ADD     A,#5
    MOV     34H,A
    JMP ADC1_B
ADC1_1_4:
    CJNE    A,#4,ADC1_1_5           //十位(34H)加 5
    ADD     A,#5
    MOV     34H,A
    JMP ADC1_B
ADC1_1_5:
    ADD     A,#5                    //十位(34H)加 5 減 10
    SUBB    A,#10
    MOV     34H,A
    MOV     A,32H                    //百位(32H)進位
    ADD     A,#1
    MOV     32H,A
ADC1_B:
    MOV     A,32H                    //百位(32H)加 2
    ADD     A,#2
    MOV     32H,A
    MOV     31H,#0                   //千位(31H)放 0
    JMP BACKADC
ADC2:
    CJNE    A,#2,ADC3               //ADCH=2，則 ADCL+512
    MOV     A,35H
    CJNE    A,#8,ADC2_1             //個位(35H)加 2 減 10
    ADD     A,#2
    SUBB    A,#10
    MOV     35H,A
    MOV     A,34H                   //十位(34H)進位
```

```
    ADD     A,#1
    MOV     34H,A
    JMP ADC2_1_0
ADC2_1:
    CJNE    A,#9,ADC2_2             //個位(35H)加 2 減 10
    ADD     A,#2
    SUBB    A,#10
    MOV     35H,A
    MOV     A,34H                   //十位(34H)進位
    ADD     A,#1
    MOV     34H,A
    JMP ADC2_1_0
ADC2_2:
    ADD     A,#2                    //個位(35H)加 2
    MOV     35H,A
ADC2_1_0:
    MOV     A,34H
    CJNE    A,#9,ADC2_1_1           //十位(34H)加 1 減 10
    ADD     A,#1
    SUBB    A,#10
    MOV     34H,A
    MOV     A,32H                   //百位(32H)進位
    ADD     A,#1
    MOV     32H,A
    JMP ADC2_B
ADC2_1_1:
    CJNE    A,#10,ADC2_1_2          //十位(34H)加 1 減 10
    ADD     A,#1
    SUBB    A,#10
    MOV     34H,A
    MOV     A,32H                   //百位(32H)進位
    ADD     A,#1
    MOV     32H,A
    JMP ADC2_B
ADC2_1_2:
    ADD     A,#1                    //十位(34H)加 1
    MOV     34H,A
ADC2_B:
```

```
    MOV     A,32H                   //百位(32H)加 5
    ADD     A,#5
    MOV     32H,A
    MOV     31H,#0                  //千位(31H)放 0
    JMP BACKADC
ADC3:
    CJNE    A,#3,GOBAADC            //ADCH=3，則 ADCL+768
    MOV     A,35H
    CJNE    A,#0,ADC3_1             //個位(35H)加 8
    ADD     A,#8
    MOV     35H,A
    JMP ADC3_1_0
ADC3_1:
    CJNE    A,#1,ADC3_2             //個位(35H)加 8
    ADD     A,#8
    MOV     35H,A
    JMP ADC3_1_0
ADC3_2:
    ADD     A,#8                    //個位(35H)加 8 減 10
    SUBB    A,#10
    MOV     35H,A
    MOV     A,34H                   //十位(34H)進位
    ADD     A,#1
    MOV     34H,A
ADC3_1_0:
    MOV     A,34H
    CJNE    A,#0,ADC3_1_1           //十位(34H)加 6
    ADD     A,#6
    MOV     34H,A
    JMP ADC3_2_0
ADC3_1_1:
    CJNE    A,#1,ADC3_1_2           //十位(34H)加 6
    ADD     A,#6
    MOV     34H,A
    JMP ADC3_2_0
ADC3_1_2:
    CJNE    A,#2,ADC3_1_3           //十位(34H)加 6
    ADD     A,#6
```

```
    MOV     34H,A
    JMP ADC3_2_0
ADC3_1_3:
    CJNE    A,#3,ADC3_1_4                   //十位(34H)加 6
    ADD     A,#6
    MOV     34H,A
    JMP ADC3_2_0
GOBAADC:
    JMP BACKADC
ADC3_1_4:
    ADD     A,#6                            //十位(34H)加 6 減 10
    SUBB    A,#10
    MOV     34H,A
    MOV     A,32H                           //百位(32H)進位
    ADD     A,#1
    MOV     32H,A
ADC3_2_0:
    MOV     A,32H
    CJNE    A,#0,ADC3_2_1                   //百位(32H)加 7
    ADD     A,#7
    MOV     32H,A
    MOV     31H,#0
    JMP BACKADC
ADC3_2_1:
    CJNE    A,#1,ADC3_2_2                   //百位(32H)加 7
    ADD     A,#7
    MOV     32H,A
    MOV     31H,#0
    JMP BACKADC
ADC3_2_2:
    CJNE    A,#2,ADC3_2_3                   //百位(32H)加 7
    ADD     A,#7
    MOV     32H,A
    MOV     31H,#0
    JMP BACKADC
ADC3_2_3:
    ADD     A,#7                            //百位(32H)加 7 減 10
    SUBB    A,#10
```

```
    MOV     32H,A
    MOV     31H,#1                          //千位放 1
BACKADC:
    CALL CHECKWA
    CALL    TRY
    RETI
WARN:                                       //煙霧(毒氣)濃度超標警示鈴中斷副程式
    MOV     R1,#90
WARN1:
    MOV     R0,#2
WARN2:
    SETB    P3.6                    //蜂鳴器輸入 1
    CALL    DELAY
    CLRP3.6                         //蜂鳴器輸入 0
    CALL    DELAY
    DJNZ    R0,WARN2                //重複產生音調
    DJNZ    R1,WARN1
    CALL    TRY                     //呼叫 TRY(顯示副程式)
    RETI
SCALE:                                      //鬧鈴表
    DB  11110111B,00011111B,110     //La
    DB  11110100B,11001110B,87      //Fa
    DB  11110001B,00010111B,65      //Do
    DB  11110100B,11001110B,87      //Fa
    DB  11110110B,00001000B,98      //So
    DB  11111000B,10001000B,250     //●Do
    DB  11110110B,00001000B,98      //So
    DB  11110111B,00011111B,110     //La
    DB  11110110B,00001000B,98      //So
    DB  11110001B,00010111B,65      //Do
    DB  11110100B,11001110B,175     //Fa
END
```