由終端裝置變成主控制器 (二)

主控制器的規劃

上一篇文章，我們簡單介紹了 Controller 與 Device 的不同，現在我們要把 DIO-I 控制板從 Device 變成 Controller，在規劃上會有哪些不同呢？在正式探討之前，我們先要建立一些基本的概念：

程式一開始的部分是變數宣告區，接下來是主程式及中斷啟始位址規劃區，然後是系統啟動區，最後才是主迴圈規劃區，在這四個區域以外的部分，通稱為副程式區，副程式區包含了定時中斷、串列中斷、及各項獨立的動作。

如果您確實地看過 DIO-I 控制板的核心程式的話，您可以發現 DIO-I 控制板本身所有的動作都是 READY 的狀態，等待 RS485 的資料一進來，動作就直接啟動。然而本身並沒有任何一項動作是主動觸發的，除了定時檢查以外…

重點來了，如果我們要將 DIO-I 控制板改成 Controller，那麼定時中斷就變成一個很重要的觸發源，在沒有任何外部觸發的條件下，可以自行觸發。當然，在主迴圈中也可以加入觸發源，不過這樣的規劃並不好，因為在主迴圈中很難掌握觸發的時序，容易造成外部動作紊亂或是不動作的問題。有了這些概念後，我們來看看 DIO-I 控制板的核心程式應該如何改寫。

我們的規劃，是先以一個 Controller 對一個 Device 進行控制的方式改寫（1 對 1），選用的 Device 是 AT2051 控制板，如果您是 RS485 講座的忠實讀者，那麼您會發現本程式的動作方式和 RS485 講座中所提到的溫控系統很類似，最大的差別在於 RS485 講座是利用 PC 做為 Controller，而本文是直接以 DIO-I 控制板當作 Controller，因此在撰寫的過程中，則必須考慮 DIO-I 控制板所能處理的資料量多寡與速度上限制，因為 AT89C2051 沒有 Intel P4 的強悍多功運算能力，也沒有 DDR RAM 的高速儲存、交換資料的能力，更沒有充裕的資料空間放進一大堆雜七雜八的堆疊與呼叫，妥善規劃每一個動作的環節就變得很重要。

```
;
;---DIO PROGRAM BY ASSEMBLER---
;
MS10    EQU     9217        ;10mS Timer
B_RATE  EQU     FDH         ;Baud_Rate
;
TMH     EQU     (65536-MS10) / 256          ;Timer high byte setting
TML     EQU     (65536-MS10) .MOD. 256      ;Timer low byte setting
;
CR      EQU     0DH
LF      EQU     0AH
;
RLY0    REG     P1.7        ;Relay setting
RLY1    REG     P1.6
```


旗威科技有限公司　地址：高雄市三民區昌裕街 18-1 號　網址：http://www.chipware.com.tw
技術專線：07-395-5152　技術支援傳真：07-395-5155　E-mail：chipware@chipware.com.tw

```
RLY2    REG     P1.5
RLY3    REG     P1.4
DRIVE   REG     P3.7        ;Serial drive setting
;
CHK_KEY REG     29H.0       ;KEY=1 <- OK, KEY=0 <- ERROR
MOTION  REG     29H.1       ;MOTION=1 <- RELAY WORK, MOTION=0 <- DATA SEND
;
ID      EQU     2AH         ;Buffer setting
INTR_CNT EQU    2BH
DG_IN   EQU     2CH
DG_OUT  EQU     2DH
I_CNT   EQU     2EH
O_CNT   EQU     2FH
I_BUF   EQU     40H         ;(40H-4FH)16 BYTES
O_BUF   EQU     30H         ;(30H-3EH)15 BYTES
TP_BUF  EQU     3FH         ;TEMPERATURE DATA BUFFER
STACK   EQU     50H
```

以上這一段與原先的核心程式比較，僅多了 TP_BUF 的規劃，這是為了儲存讀取的溫度值所規畫的暫存區。

```
;
;---FOR FUNCTION SETTING---
;
X1      REG     DG_IN.0
X2      REG     DG_IN.1
X3      REG     DG_IN.2
X4      REG     DG_IN.3
Y1      REG     DG_OUT.0
Y2      REG     DG_OUT.1
Y3      REG     DG_OUT.2
Y4      REG     DG_OUT.3
T1      EQU     28H
T2      EQU     27H
RESULT  REG     29H.2
OLD_R   REG     29H.3
;
H_LIMIT EQU     31
L_LIMIT EQU     30
```

在這一段程式中，我們僅加入了 H_LIMIT、L_LIMIT 兩個臨界溫度的判斷，嚴格來說，這兩個值應該擺在前一段程式中，但是為了區分這是量測溫度專用的數值，因此才將這兩個數值放在這裡。在 8051 初學者講座中，我們有提到記憶體規畫的一些概念，其中我們也有概略地提到變數的宣告應該區分為數值變數與位址變數兩大類，並分開來寫，所謂的數值變數，就是這裡所用到的 H_LIMIT、L_LIMIT 兩個臨界溫度值，而位址變數則是前一段程式中所提到的 TP_BUF 溫度值暫存區。

```
;
;---PROGRAM STARTER---
;
        ORG     0000H       ;MAIN PROGRAM
        JMP     START
        ORG     0003H
        RETI
        ORG     000BH       ;TIMER0 INTERRUPT PROGRAM
        JMP     T0_ISR
```

```
;
        ORG     0013H
        RETI
        ORG     001BH
        RETI
        ORG     0023H     ;SERIAL INTERRUPT PROGRAM
        JMP     SI0_ISR
```

這一段程式與核心程式完全一樣，也不需要進行修改。

```
;
;---MAIN PROGRAM---
;
        ORG     0030H
START   MOV     R2,#00H
$       DJNZ    R2,$
        MOV     SP,#STACK
        MOV     P1,#FFH
        MOV     P3,#FFH
        MOV     INTR_CNT,#00H
        MOV     I_CNT,#00H
        MOV     O_CNT,#00H
        MOV     DG_IN,#00H
        MOV     DG_OUT,#00H
;FOR FUNCTION
        MOV     T1,#00H
        MOV     T2,#00H
        CLR     OLD_R
        CLR     RESULT
;CLR IO BUFFER
        MOV     R1,#I_BUF
        MOV     R7,#10H
$1      MOV     @R1,#00H
        INC     R1
        DJNZ    R7,$1
        MOV     R0,#O_BUF
        MOV     R7,#10H
$2      MOV     @R0,#00H
        INC     R0
        DJNZ    R7,$2
;
        CALL    READ_ID
        CALL    SET_ISR
;
$LOOP   MOV     A,ID
        CJNE    A,#0,$FUNC
        JMP     $LOOP
$FUNC   CALL    FUNCTION
        JMP     $FUNC
```

系統啟始區與主迴圈都是包含在主程式的部份，由$LOOP 開始到主程式結束的這一小段便是主迴圈區，這裡的程式動作會不斷地循環執行。

```
;
;---NORMAL CONTROL---
;
FUNCTION
;INPUT
        MOV     A,TP_BUF
;
```

```
;CHECK STATE
            CJNE    A,#H_LIMIT,$1
$1          JC      $2
            SETB    RESULT
            JMP     $OUTPUT
$2          CJNE    A,#L_LIMIT,$3
$3          JNC     $OUTPUT
            CLR     RESULT
;
;OUTPUT
$OUTPUT
            JNB     RESULT,$4
            SETB    Y1
            SETB    Y2
            JMP     $END
$4          CLR     Y1
            CLR     Y2
$END        RET
```

這一段是程式的主要動作，如果高於溫度的臨界上限，則啟動 Relay，如果低於下限則關閉。

```
;
;---ID SETTING; SWITCH STATE---
;
READ_ID
      MOV       A,P1
      CPL       A
      ANL       A,#0FH
      CJNE      A,#CR,$1
      MOV       A,#17
      JMP       $2
$1    CJNE      A,#LF,$2
      MOV       A,#16
$2    MOV       ID,A
      RET
```

讀取 ID 的作用並不是用來設定 DIO-I 控制板的 ID，而是用來決定外部 Device 的 ID 值，如果 AT2051 控制板的 ID 是 3，那麼 DIO-I 控制板就必須撥成 3 才能與其溝通。

```
;
;---SERIAL INTERRUPT PROGRAM---
;
SI0_ISR
            PUSH    A
            PUSH    PSW
;
            JB      RI,$RXD
            JMP     $1
;READ DATA
$RXD
            CLR     RI
;
            CALL    STR_OVER
            JNB     CHK_KEY,$1
$INPUT      MOV     A,SBUF
            CALL    SAVE_INPUT_DATA
            CJNE    A,#LF,$1            ;A=LF -> INPUT DATA END
```

```
            CALL    TEMPERATURE     ;SAVE TEMPERATURE DATA TO BUFFER
;CLEAR INPUT BUFFER
$C_BUF      MOV     R4,#0DH
$LOOP       MOV     R1,#I_BUF
            MOV     @R1,#00H
            INC     R1
            DJNZ    R4,$LOOP
            MOV     I_CNT,#00H
;
$1          JB      TI,$TXD
            JMP     $END
;TRAMSMIT DATA
$TXD
            CLR     TI
            SETB    DRIVE
;
$END        POP     PSW
            POP     A
            RETI
```

以上這一段程式修正了很多，不過最主要的修正是在輸入與輸出資料的比重有所不同所進行的修正。此時您可能會有一個疑問，這時的 DIO-I 控制板不是 Controller 嗎？那麼輸出資料的部分該怎麼觸發呢？又該怎麼規劃輸出的資料呢？是不是也要把值放在 OUT_BUF 中呢？關於這些問題，其實您只要繼續往下看，就可以知道了。

```
;
;---CHECK INPUT DATA LENGTH---
;
STR_OVER
            MOV     A,I_CNT
            CJNE    A,#12,$WORDS
$WORDS      JC      $OK
            MOV     I_CNT,#00H
            MOV     R1,#I_BUF
            CLR     CHK_KEY     ;CHK_KEY=0 -> OVER LENGTH -> I_CNT=0, R1=I_BUF+0
            JMP     $END
$OK         SETB    CHK_KEY     ;CHK_KEY=1 -> OK
$END        RET
```

在此必須改變一下字串長度的判斷設定，因為原先的設定是針對由 PC 端傳送的指令所規劃的，現在卻是來自 AT2051 控制板的回應，資料長度不同，如果不改會造成收到的資料出現很嚴重的錯誤。

```
;
;---SAVE DATA TO BUFFER---
;
SAVE_INPUT_DATA
            CJNE    A,#'T',$END
            MOV     R1,#I_BUF
            MOV     I_CNT,#0
$END        CALL    UPPER
            MOV     @R1,A
            INC     R1
            INC     I_CNT
            RET
```

這一小段程式也是跟著 AT2051 控制板所回應的指令所做的調整。

```
;
;---SAVE TEMPERATURE DATA TO BUFFER---
;
TEMPERATURE
            MOV     R1,#I_BUF+4
            MOV     A,@R1
            SUBB    A,#'0'
            MOV     B,#10
            MUL     AB
            MOV     TP_BUF,A
;
            MOV     R1,#I_BUF+5
            MOV     A,@R1
            SUBB    A,#'0'
            ADD     A,HS_BUF
            MOV     TP_BUF,A
            RET
```

這一段程式是最重要的部分，也是變化性最大的部分，因為讀取方式的設計不同，相對就會影響整個系統的運作效能，一定要小心。如果您的系統很小，所用的 Device 也都回應很單純的資料，那麼您可以用很『浪費』的寫法，將所有的讀取值一個一個判斷，如果您必須爭取判斷的時間，加大系統靈活運用的空間，那麼我們所提供的這個寫法會是您比較好的選擇。一來可以節省暫存空間的浪費，二來可以擴充規畫成歷史資料的寫法，這要端看您的應用而定了。

```
;
;---TO UPPER---
;
UPPER
            CJNE    A,#'a',$1
$1          JC      $END
            CJNE    A,#'z'+1,$2
$2          JNC     $END
            CLR     C
            SUBB    A,#20H
$END        RET
```

小寫換成大寫，與之前的核心程式相同。

```
;
;---SAVE OUTPUT DATA TO O_BUF & SEND 1ST BYTE---
;
ENQUARY
      CLR       DRIVE     ;ENABLE DRIVE
      MOV       SBUF,ID
      RET
```

這裡的詢問動作，是由系統所觸發的唷！這也是串列中斷啟動的重要動作。

旗威科技有限公司　地址：高雄市三民區昌裕街 18-1 號　網址：http://www.chipware.com.tw
技術專線：07-395-5152　技術支援傳真：07-395-5155　E-mail：chipware@chipware.com.tw

```
;
;---TIMER INTERRUPT PROGRAM---
;
T0_ISR
            MOV     TH0,#TMH
            MOV     TL0,#TML
            PUSH    A
            PUSH    PSW
            CALL    READ_ID
;SAVE INPUT STATE TO DG_IN
$input_state
            MOV     A,P3
            RR      A
            RR      A
            CPL     A
            ANL     A,#0FH
            MOV     DG_IN,A
;ID=0 -> DG_OUT=DG_IN
$id0_motion
            MOV     A,ID
            INC     A
            DJNZ    A,$out_state
            MOV     DG_OUT,DG_IN
;RELAY WORK BY DG_OUT DATA
$out_state
            MOV     A,DG_OUT
            JB      ACC.0,$ON_0
            SETB    RLY0
            JMP     $1
$ON_0       CLR     RLY0
$1          JB      ACC.1,$ON_1
            SETB    RLY1
            JMP     $2
$ON_1       CLR     RLY1
$2          JB      ACC.2,$ON_2
            SETB    RLY2
            JMP     $3
$ON_2       CLR     RLY2
$3          JB      ACC.3,$ON_3
            SETB    RLY3
            JMP     $counter
$ON_3       CLR     RLY3
;1 sec COUNTER
$counter
            INC     INTR_CNT
            MOV     A,INTR_CNT
            CJNE    A,#100,$T0_END
            MOV     INTR_CNT,#00H
;
            CALL    ENQUARY
;1S COUNTER
            MOV     A,T1
            JZ      $T2
            DEC     T1
$T2   MOV A,T2
            JZ      $T0_END
            DEC     T2
;
$T0_END
            POP     PSW
            POP     A
            RETI
```


旗威科技有限公司　地址：高雄市三民區昌裕街 18-1 號　網址：http://www.chipware.com.tw
技術專線：07-395-5152　技術支援傳真：07-395-5155　E-mail：chipware@chipware.com.tw

在定時中斷中，我們加入了定時詢問的動作，時間間隔為 1 秒。當然這個速度並不算快，還可以將速度調整得更快，但是由於我們在規畫之初僅有一秒的 Timer 可用，如果要加快速度，則必須將 Timer 做適度的調整。

```
;
;---INTERRUPT SETTING---
;
SET_ISR
        MOV TMOD,#21H
        MOV TCON,#50H
        MOV TH0,#TMH
        MOV TL0,#TML
        MOV SCON,#50H
        MOV TH1,#B_RATE
        SETB    ET0
        SETB    ES
        SETB    EA
        RET
;
;---BINARY TO ASCII---
;
BIN2ASC
        CJNE    A,#9,$1
        JMP  $NUM
$1      JC   $NUM
        SUBB    A,#10
        ADD A,#41H
        JMP  $OUT
$NUM    ADD A,#30H
$OUT
        RET
```

以上整個程式裡最大的改變，是加入了 TEMPERATURE 的副程式，這個副程式的功能是用來讀取 AT2051 控制板所量測到的溫度值用的，而原先接受串列指令所會執行的動作，我們也做了必要的修改。其餘的設定則先予以保留，因為程式還需要做適度的擴充。另外在主迴圈的 FUNCTION 副程式中，我們改寫成超過溫度上限則啟動 RLY，低於溫度下限則關閉 RLY，這個動作與到此為止，我們完成了 DIO-I 控制板的核心程式了！先消化一下，我們再來看看實際上運作的情形。

旗威科技有限公司　地址：高雄市三民區昌裕街 18-1 號　網址：http://www.chipware.com.tw
技術專線：07-395-5152　技術支援傳真：07-395-5155　E-mail：chipware@chipware.com.tw