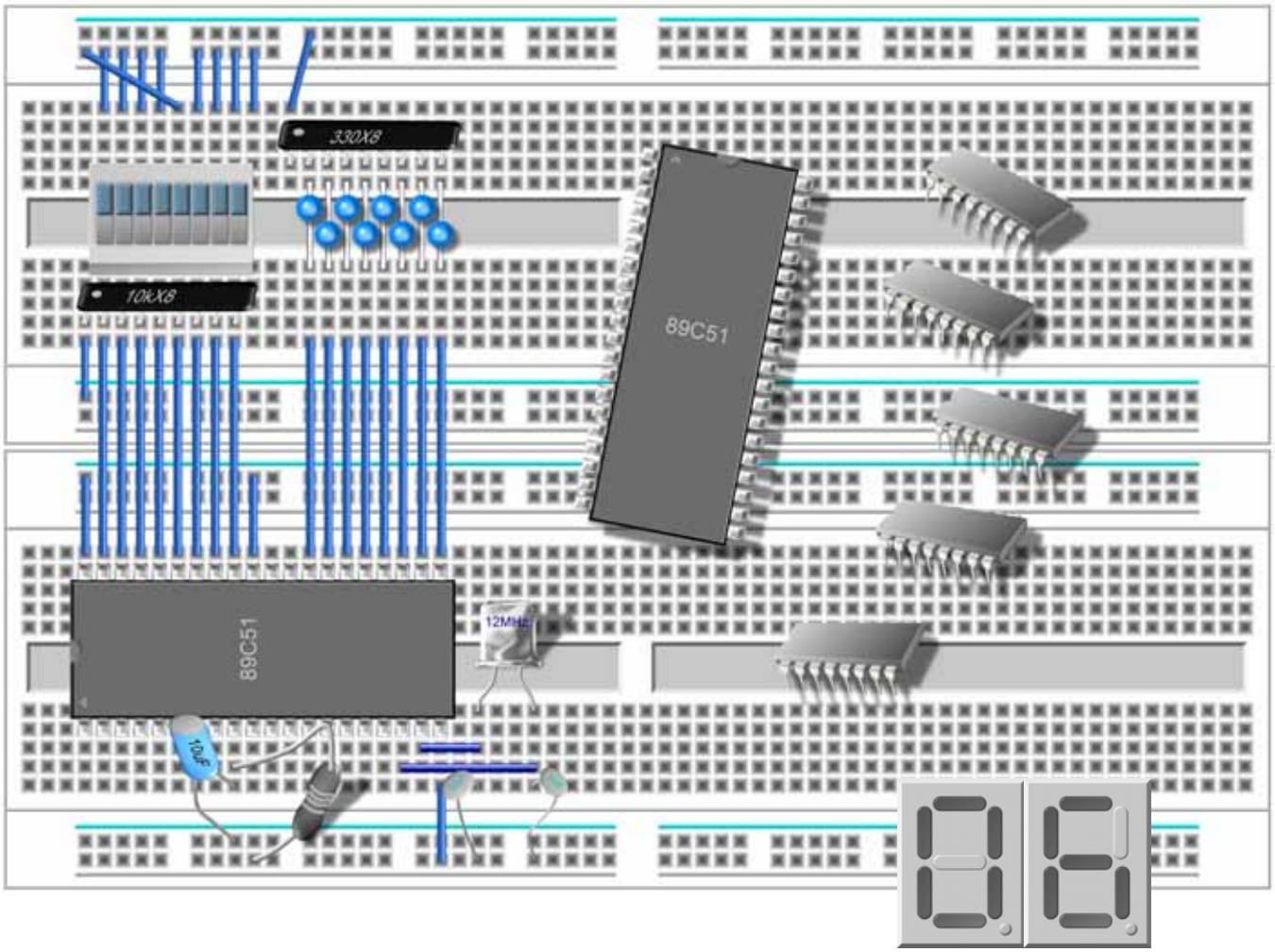


計時計數器之應用

本章內容豐富，主要包括三部分：

- 硬體部分：

  認識 8051 計時計數器的架構，以及其四種模式。

- 指令部分：

  詳細說明布林運算指令。

- 程式與實作部分：

  計時器之應用程式、碼表程式、頻率產生器程式、計頻器應用程式等。

6-1 8051 之計時計數器

計時計數器(timer/counter)也是一種中斷裝置，MCS-51 提供內部計時及外部計數功能，當計時或計數達到終點，即提出中斷，而 CPU 將暫時放下目前所執行的程式，先去執行特定的程式，待完成特定的程式後，再返回剛才放下的程式。譬如說，老師正在講課時，突然下課鐘響，老師立即暫停課程進度，先下課，待下次上課，再繼續剛才暫停的課程。這樣的動作就是「計時計數器中斷」。

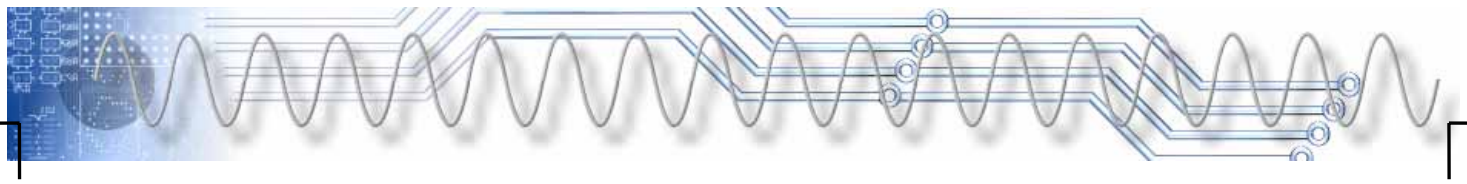
6-1-1 MCS-51 之計時計數器

8051 提供兩個 16 位元的計時計數器，分別是Timer 0 及Timer 1(簡稱T0 及T1)，而 8052 則提供三個 16 位元的計時計數器，除了 8051 的T0 與T1 外，還多一個Timer 2。這三個計時計數器可做爲內部計時器或外部計數器，若當成內部計時器時，則是計數內部的脈波，以 12MHz的計數時鐘脈波系統爲例，將此計數時鐘脈波除 12 後，送入計時器，因此，計時器所計數的脈波週期爲 1 微秒。若採 16 位元的計時模式，則最多可計數 2^{16}個脈波(**65,536**)，約 0.0655 秒。

若當成外部計數器時，則是計數由T0 或T1 接腳送入的脈波。同樣地，若採 16 位元的計時模式，則最多可計數 2^{16}個脈波，也就是 65,536 個計數量，相當可觀！

MCS-51 的計時計數器可規劃成四種工作模式，分別是Mode 0、Mode 1、Mode 2 及Mode 3，Mode 0 爲兩個 13 位元的計時計數器，其最大計數量爲 2^{13}(即 **8192**)；Mode 1 爲兩個 16 位元的計時計數器，其最大計數量爲 2^{16}，爲較常使用的工作模式；Mode 2 爲兩個 8 位元但可自動載入的計時計數器，其最大計數量爲 2^{8}(即 **256**)，至於「自動載入」功能稍後再說明；Mode 3 爲一個 8 位元的計時計數器、一個 8 位元的計時器，屬於少用的工作模式。

當完成計時或計數時，將產生中斷，其中 Timer 0 的中斷向量爲 **0BH**、Timer 1 的中斷向量爲 **1BH**、Timer 2 的中斷向量爲 **2BH**。



6-1-2 計時計數器模式暫存器 TMOD

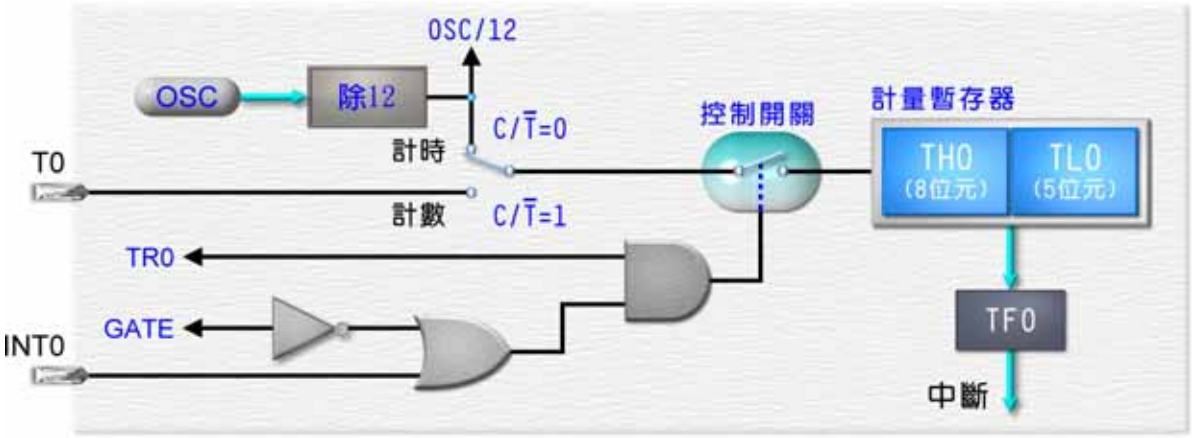
計時計數器模式暫存器 TMOD 的功能是設定計時計數器的工作模式、計數信號來源及啓動計時計數器方式等，如下圖所示：

（圖1） TMOD 暫存器

TMOD 模式暫存器的高四位元(TMOD.7 至 TMOD.4)用以設定 Timer 1 的工作模式，而低四位元(TMOD.3 至 TMOD.0)用以設定 Timer 0 的工作模式，這兩部分的結構類似，唯控制對象不同而已。以低四位元爲例，GATE 位元爲 Timer 的閘控開關，用以決定其啓動方式。若 GATE=0，則只要 TR0 位元=1，即可啓動 Timer 0，稱爲內部啓動或軟體啓動。若 GATE=1，則須先將 TR0 位元設定爲 1，再等待 INT0 接腳爲高態，方可啓動 Timer 0，稱爲外部啓動或硬體啓動。 C/$\overline{T}$ 位元爲計時計數切換開關，若 C/$\overline{T}$ 位元=0，則 Timer 0 爲內部計時器，用以計數由 OSC/12 的脈波。若 C/$\overline{T}$ 位元=1，則 Timer 0 即爲外部計數器，用以計數由 T0 接腳輸入的脈波。

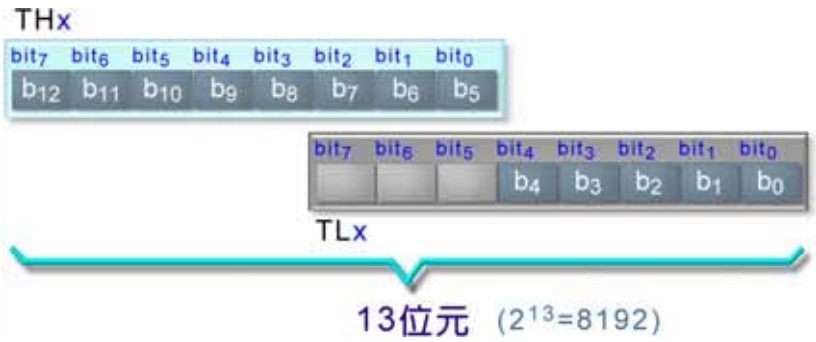
M1 及 M0 兩個位元可規劃工作模式，這四種工作模式，如下說明：

MODE 0



（圖2） MODE 0 工作模式(Timer 0 為例)

如上圖所示，Mode 0 工作模式提供兩個 13 位元的計時計數器(Timer 0 及 Timer 1)，其計數量分別放置在 THx 與 TLx 兩個 8 位元的計量暫存器裡，其中 THx 放置 8 位元、TLx 放置 5 位元，如下圖所示：



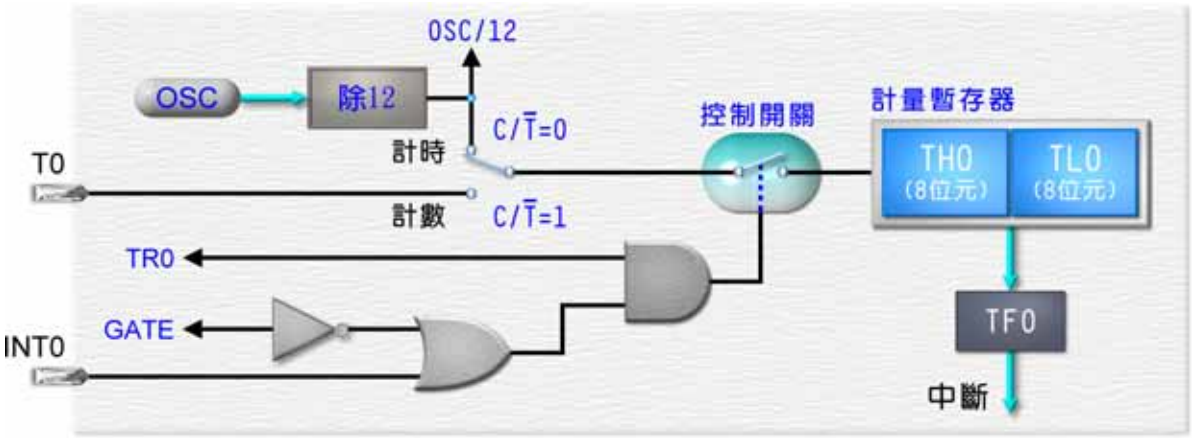
（圖3） Mode 0 工作模式之計量

若要執行計時功能，則將 C/T 位元設定爲 0，CPU 將計數被除 12 的系統時脈，每個時脈爲 1 微秒。若要執行計數功能，則將 C/T 位元設定爲 1，CPU 將計數從 Tx 接腳輸入的脈波。

計時計數器受「控制開關」所控制，開啓這個開關的方法有兩個，第一種是外部啓動，也就是將 GATE 位元設定爲 1，再將 TRx 位元設定爲 1，然後等待 INTx 接腳的信號，當 INTx 接腳爲高態時，即可啓動這個計時計數器。第二種是內部啓動，也就是將 GATE 位元

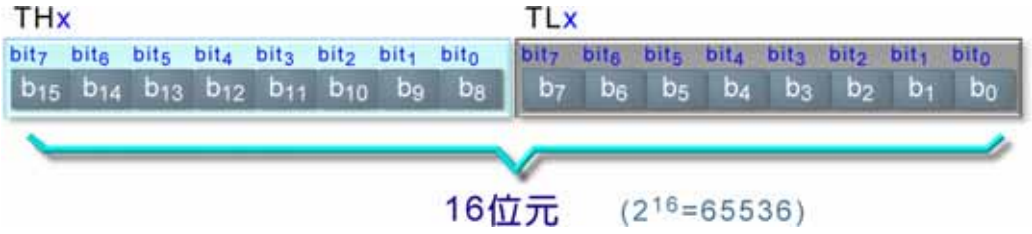
設定爲 0，接下來，只要將 TRx 位元設定爲 1，即可啓動這個計時計數器。

MODE 1



（圖4） MODE 1 工作模式(Timer 0 為例)

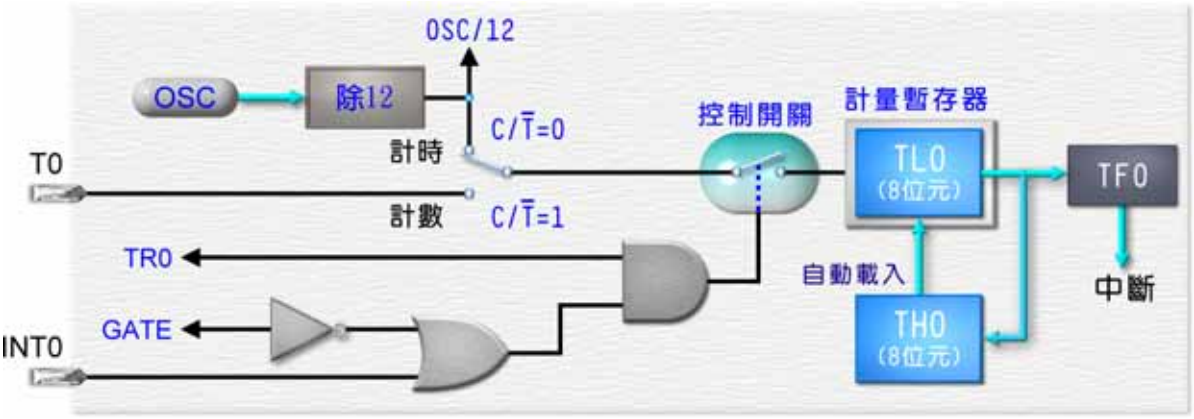
如上圖所示，Mode 1 工作模式提供兩個 16 位元的計時計數器(Timer 0 及 Timer 1)，其計數量分別放置在 THx 與 TLx 兩個 8 位元的計量暫存器裡，其中 THx 放置 8 位元、TLx 放置 8 位元，如下圖所示：



（圖5） Mode 1 工作模式之計量

此工作模式的計時/計數功能切換方式，與 Mode 0 完全一樣；而啓動計時計數器的方式，與 Mode 0 完全一樣。計數量 Mode 1 又比 Mode 0 還大！換言之，Mode 1 可以完全取代 Mode 0，所以，很少人使用 Mode 0 了(難用又沒必要性)。

MODE 2

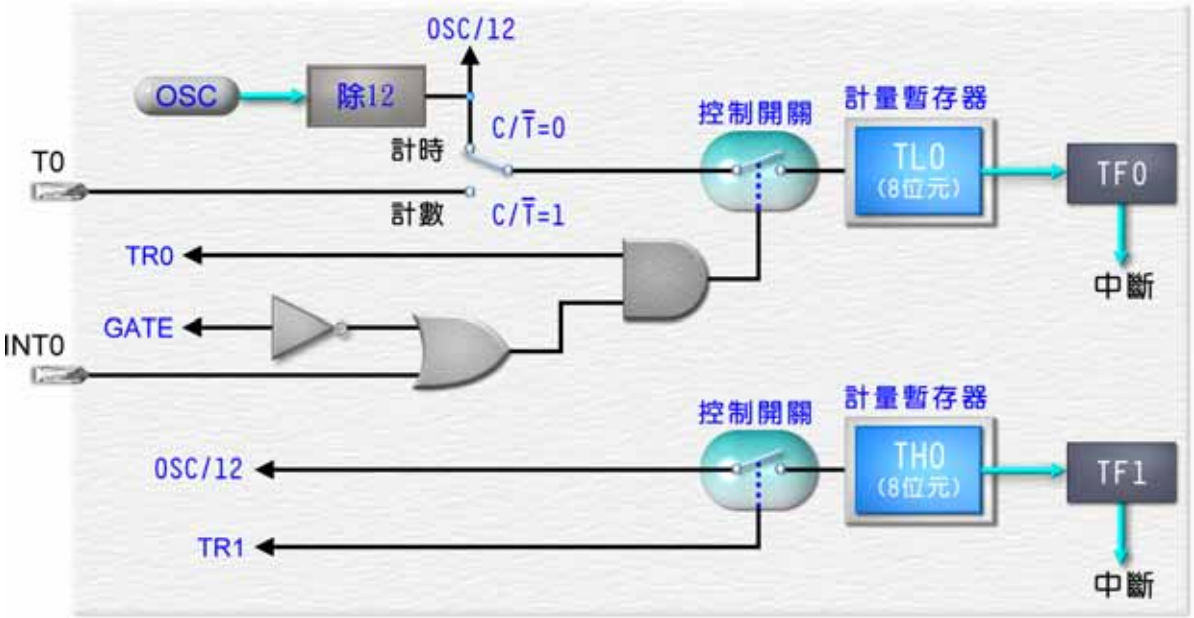


（圖6） MODE 2 工作模式(Timer 0 為例)

如上圖所示，Mode 2 工作模式提供兩個 8 位元可自動載入的計時計數器(Timer 0 及 Timer 1)，其計數量放置在 TLx 計量暫存器裡，當該計時計數器中斷時，將會自動將 THx 計量暫存器裡的計數量，載入到 TLx 裡。由於只有 8 位元，因此，其計數範圍僅 256。

此工作模式的計時/計數功能切換方式，與 Mode 0 完全一樣；而啓動計時計數器的方式，與 Mode 0 完全一樣。

MODE 3



（圖7） MODE 3 工作模式

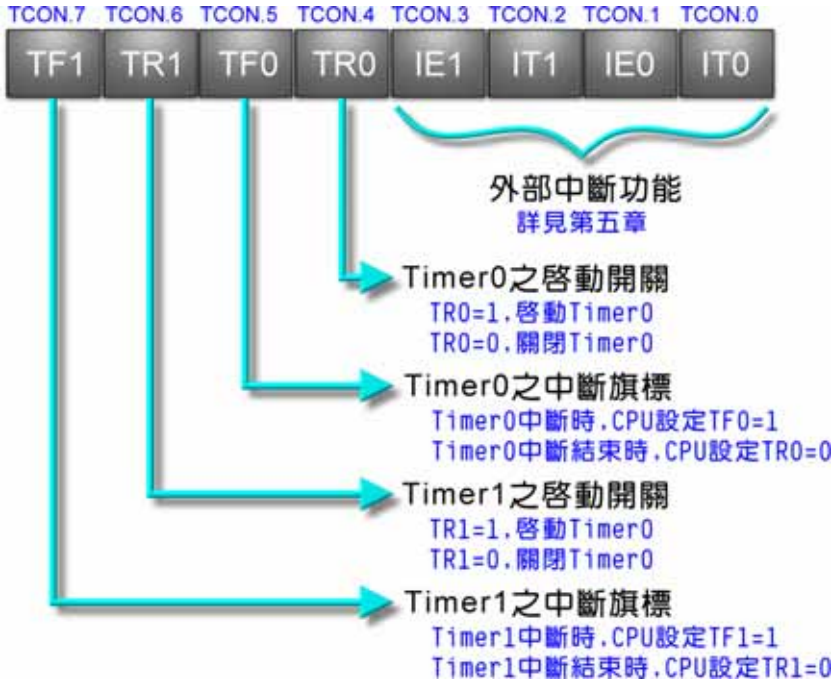
如上圖所示，Mode 3 工作模式是一種特殊的模式，提供一個 8 位元的計時計數器 Timer 0 及一個 8 位元的計時器 Timer 1，而其奇特的架構，已不太像真正的 Timer 0 或 Timer 1 了。

其中的 Timer 0 計時計數器是由 T0、INT0 接腳，TR0、GATE 位元，以及 TL0 計量暫存器所構成，除了不具有自動載入功能外，與 Mode 2 的 Timer 0 幾乎完全一樣。

其中的 Timer 1 計時器是由 TR1 位元及 TH0 計量暫存器所構成，除了不具有計數及自動載入功能外，幾乎可以 Mode 2 的 Timer 1 所取代。筆者看不出有任何 Mode 3 工作模式存在的理由，但這個模式的存在是事實。

6-1-3 計時計數器控制暫存器 TCON

計時計數器控制暫存器 TCON 的高四位元提供計時計數器的啓動開關，以及中斷時的旗標，如下圖所示：



（圖8） TCON 暫存器

6-1-4 計量暫存器

MCS-51 的計量暫存器是 THx 及 TLx 兩個八位元的暫存器，除了 Mode 3 外，TH0、TL0 是 Timer 0 所使用的計量暫存器，TH1、TL1 是 Timer 1 所使用的計量暫存器，若是 8052，則還有 Timer 2 所使用的 TH2、TL2 計量暫存器。

MCS-51 的計時計數器是一種正數的計數器，當計數到滿(溢位)時，即產生中斷。計量暫存器就像是一條跑道，而其終點位置是固定的，若要計數多少，就從終點往前推多少，以做爲起點。例如要在 400 公尺的跑道上，舉行 100 公尺的跑步比賽，則從終點(400 公尺)處往前推 100 公尺，也就是 300 公尺處，做爲起跑點。

不同模式的計時計數，其最大計量值各不同，如下：

Mode 0：8192
Mode 1：65536
Mode 2 及 Mode 3：256

而設定計數量的方式也有點差異，如下說明：

MODE 0

由於在Mode 0 工作模式下，TLx計量暫存器只使用 5 位元，而 2^5=32，我們要把計數起點的值，除以 32，其餘數放入TLx計量暫存器；而其商數放入THx計量暫存器。例如要使用Timer 0 計數 6000，則填入計量暫存器的指令如下：

```
        MOV     TL0, #(8192-6000).MOD.32    ;取 5 位元的餘數
        MOV     TH0, #(8192-6000)/32        ;取 5 位元的商數
```

MODE 1

由於在Mode 1 工作模式下，TLx、THx計量暫存器各使用 8 位元，而 2^8=256，我們要把計數起點的值，除以 256，其餘數放入TLx計量暫存器；而其商數放入、THx計量暫存器。例如要使用Timer 0 計數 50000，則填入計量暫存器的指令如下：

```
        MOV     TL0, #(65536-50000).MOD.32  ;取 8 位元的餘數
        MOV     TH0, #(65536-50000)/32      ;取 8 位元的商數
```

此外，我們還可以利用組譯程式所提供的符號『<、>』，以決定取用高八位元(>)或低八位元(<)，如下：

```
        MOV     TL0, #<(65536-50000)        ;取用低 8 位元
        MOV     TH0, #>(65536-50000)        ;取用高 8 位元
```

如果還嫌麻煩，直接使用負的計數量即可，如下：

```
        MOV     TL0, #<-50000               ;取用低 8 位元
        MOV     TH0, #>-50000               ;取用高 8 位元
```

嘿嘿，前述兩種方式，可不適用於 Mode 0 喔！

MODE 2

由於在Mode 2 工作模式下，只使用TLx計量暫存器，但THx計量暫存器做爲自動載入的值，而其中都使用 8 位元(2^8=256)，所以只要把 256 減去計數起點的值，再分別放入TLx及THx計量暫存器即可。例如要使用Timer 0 計數 100，則填入計量暫存器的指令如下：

```
        MOV     TL0, #(256-100)             ;填入計數量
        MOV     TH0, #(256-100)             ;填入自動載入值
```

當然，還是可以直接使用負的計數量，如下：

```
        MOV     TL0, #-100                  ;填入計數量
        MOV     TH0, #-100                  ;填入自動載入值
```

嘿嘿，前述兩種方式，可不適用於 Mode 0 喔！

MODE 3

由於在 Mode 3 工作模式下，使用 TL0 計量暫存器做爲第一個計時計

數器的計數量，而 TH0 計量暫存器做爲第二個計時器的計數量。有用到的計時計數器或計時器才須要填入，例如只要使用第一個計時計數器，則只須填入 TL0 計量暫存器；若只要使用第二個計時器，則只須填入 TH0 計量暫存器；若兩個都要使用，則分別將個別的值填入 TL0 及 TH0 計量暫存器。而填入 TL0 或 TH0 計量暫存器的方法，與 Mode 2 一樣。

6-1-5 計時計數器的應用

計時計數器的應用包括三項措施，即中斷向量的設置、計時計數器中斷的設定、計數量的設定，啓動計時計數器，以及中斷副程式的撰寫，如下說明：

設置中斷向量

計時計數器中斷向量的設定與 INT 外部中斷向量類似，而其中斷向量分別是 0BH、1BH 及 2BH，通常會在這些位址上，以「JMP　XXX」指令，跳至特定的中斷副程式，然後在該中斷副程式之中，才提供真正的服務。例如希望 Timer 0 中斷後，執行 XXX 中斷副程式，則其中斷向量設置如下：

```
        ORG     0
        JMP     START           ;取用高 8 位元
        ORG     0BH             ;Timer 0 中斷向量
        JMP     XXX             ;執行 XXX 中斷副程式
        :
```

中斷設定

中斷的設定包括開啓中斷開關(即 IE 暫存器的設定)、中斷優先等級的設定(即 IE 暫存器的設定)、中斷信號的設定(即 TCON 暫存器的設定)、設定新的堆疊位置等。基本上，IE 暫存器及 IP 暫存器的設定，可以利用 MOV 指令、SETB 指令或 CLR 指令，例如要開啓「總開關」、「T0 開關」，則可以下列命令：

```
MOV     IE, #10000010B
```

也可以使用下列命令：

```
SETB    IE.7
SETB    IE.1
```

若位置不清楚，則可直接指定「開」的名稱，即：

```
SETB    EA
SETB    ET0
```

同理，IP 暫存器、TCON 暫存器的設定，也可以使用 MOV 指令或 SETB 指令，不過，使用 SETB 指令比較簡單，且具可讀性，例如要把 T1 中斷的優先等級提高，則：

```
SETB    PT1
```

至於堆疊的位置問題，程式預置堆疊指標指向 07H 位置，也就是從 08H 開始存放堆疊資料，而 08H 正是暫存器庫 1(即 RB1)的位置，爲避免衝突而破壞資料，因此，將堆疊指標改到其它地方，例如要把堆疊移到 30H，則：

```
MOV     SP, #30H
```

計數量設定

在啓動計時計數器之前，必須先設定計數量，而設定計數量的方法，依工作模式的不同，而有所不同，在剛才的單元裡(6-1-4 節)已說明，在此不贅述。

啓動計時計數器

啓動計時計數器的方法，只要在程式中以「SETB　TRx」指令即可，例如要啓動 Timer 0，則使用「SETB　TR0」指令。

中斷副程式

「中斷副程式」就是一種副程式，而這種副程式與一般副程式之最大差異是中斷副程式的最後一道指令是「RETI」，一般副程式的最後一道指令是「RET」。

6-2 8052 之 Timer 2

本單元爲選擇性教材，若教學時間不夠，或不使用 8052，則可跳過本單元。Timer 2 爲 8052 才有的計時計數器，其架構與用法，和 Timer 0、Timer 1 有些不同，Timer 2 有三種工作模式，如下說明：

6-2-1 T2CON 暫存器

（圖9） T2CON 暫存器

T2CON 暫存器是 Timer 2 的控制暫存器，其中各位元的功能，如下說明：

- **TF2**：本位元爲 Timer 2 溢位旗標，當 Timer 2 中斷時，CPU 會將 TF2 位元設定爲 1；不過，結束 Timer 2 中斷時，CPU 並不會將 TF2 恢復，必須在程式中，以「CLR　TF2」指令將它恢復爲 0。
- **EXF2**：本位元爲 Timer 2 的外部旗標，當 T2EX 接腳(即 P1.0)輸入負緣信號時，且 EXEN2 位元爲 1，即進入「捕獲模式」或「自動載入模式」，此時 EXF2 位元將被設定爲 1，並產生 Timer 2 中斷；不過，結束 Timer 2 中斷時，CPU 並不會將 EXF2 恢復，必須在程式中，以「CLR　EXF2」指令將它恢復爲 0。
- **RCLK**：本位元爲串列埠接收時脈選擇位元。當 RCLK 位元爲 1 時，串列埠將以 Timer 2 溢位脈波做爲在 Mode 1 或 Mode 3 模式時，接收的時脈信號。若 RCLK 位元爲 0，則串列埠將以 Timer 1 溢位脈波做爲接收的時脈信號。
- **TCLK**：本位元爲串列埠傳輸時脈選擇位元。當 TCLK 位元爲 1 時，串列埠將以 Timer 2 溢位脈波做爲在 Mode 1 或 Mode 3 模式時，傳輸的時脈信號。若 TCLK 位元爲 0，則串列埠將以 Timer 1 溢位脈波做爲傳輸的時脈信號。
- **EXEN2**：本位元爲 Timer 2 的外部致能控制位元，當本位元爲 1

時，若 Timer 2 未被做爲串列埠的時脈產生器時，且 T2EX 接腳輸入一個負緣觸發信號，即可使 Timer 2 進入捕獲模式或自動載入模式。若本位元爲 0 時，則 Timer 2 將不理 T2EX 接腳的信號變化。

- **TR2**：本位元爲 Timer 2 的啓動位元，當本位元爲 1 時，即可啓動 Timer 2。若本位元爲 0 時，則停用 Timer 2。

- **C/$\overline{\text{T2}}$** ：本位元爲 Timer 2 計時計數功能切換開關，當本位元爲 1 時，Timer 2 將執行外部計數功能，以計數 T2 接腳所輸入的脈波信號。若本位元爲 0 時，則 Timer 2 將執行內部計時功能，以計數系統的時鐘脈波。

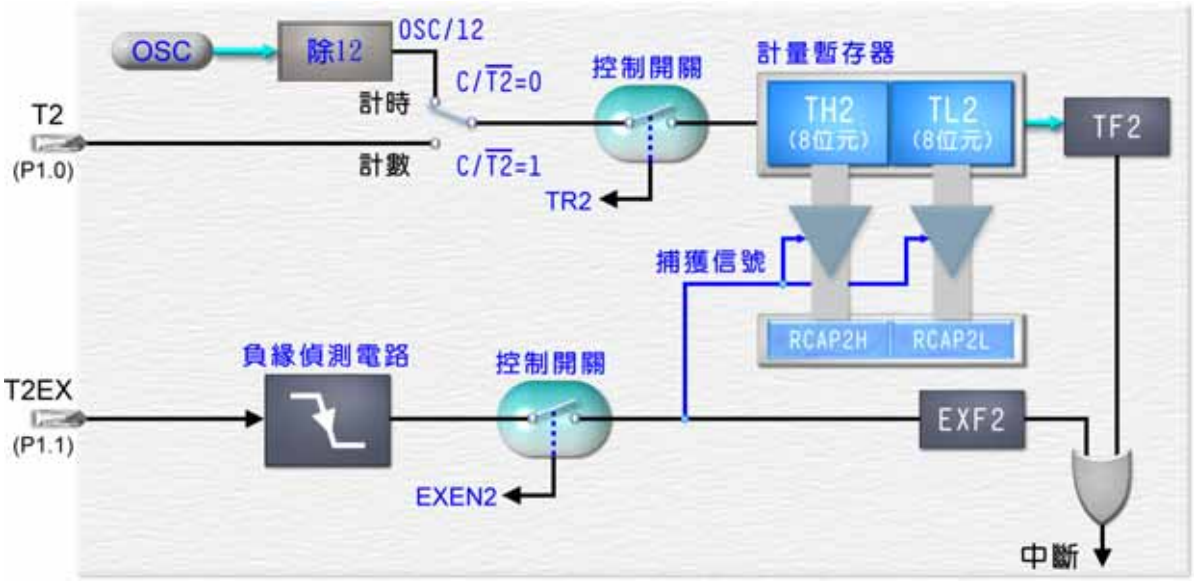
- **CP/$\overline{\text{RL2}}$** ：本位元爲 Timer 2 的工作模式切換位元，當本位元爲 1 時，若 EXEN2=1，且 T2EX 接腳輸入一個負緣觸發信號，Timer 2 將產生捕獲的動作，將 TH2 與 TL2 的資料存入 RCAP2H 與 RCAP2L。當本位元爲 0 時，若有溢位發生，或 EXEN2=1，且 T2EX 接腳輸入一個負緣觸發信號，Timer 2 將產生自動載入的動作，將 RCAP2H 與 RCAP2L 的資料載入 TH2 與 TL2。

Timer 2 提供三種工作模式，其設定方式，可歸納如下表所示：

RCLK+TCLK	CP/$\overline{\text{RL2}}$	TR2	Mode
0	0	1	16 位元自動載入模式
0	1	1	16 位元捕獲模式
1	X	1	鮑率產生器模式
X	X	0	不使用

6-2-2 捕獲模式

捕獲模式(Capture Mode)是將 TH2 與 TL2 暫存器的資料(16 位元)，抓進 RCAP2H 及 RCAP2L 暫存器，其架構如下圖所示：



（圖10）**捕獲模式**

若要使用捕獲模式，必須將 T2CON 暫存器裡的CP/$\overline{RL2}$ 位元設定爲 1。如同 Timer 0、Timer 1 一樣，捕獲模式可計數內部時鐘脈波(OSC/12)，也可以計數由 T2 接腳輸入的外部脈波，只要將 T2CON 暫存器裡的C/$\overline{T2}$ 位元設定爲 0，則爲內部計時器；將 T2CON 暫存器裡的C/$\overline{T2}$ 位元設定爲 1，則爲外部計數器。另外，T2CON 暫存器裡的 EXEN2 位元也要設定爲 1，才能進行捕獲模式。而 Timer 2 的啓動開關爲 TR2，若將 TR2 設定爲 1，即可啓動 Timer 2；TR2=0，即可停用 Timer 2。

啓動 Timer 2 後，Timer 2 即進行計數工作，若偵測到 T2EX 接腳輸入信號中含有負緣信號，即啓動捕獲信號，將當時 TH2 暫存器的內容，將被複製到 RCAP2H 暫存器、TL2 暫存器的內容，將被複製到 RCAP2L 暫存器，同時 EXF2 位元設定爲 1，並產生 Timer 2 中斷。不過，Timer 2 的中斷並不影響計數的動作，待 Timer 2 計數溢位時，則 TF2 位元設定爲 1，並產生 Timer 2 中斷。

歸納上述，若要採捕獲模式工作，必須：

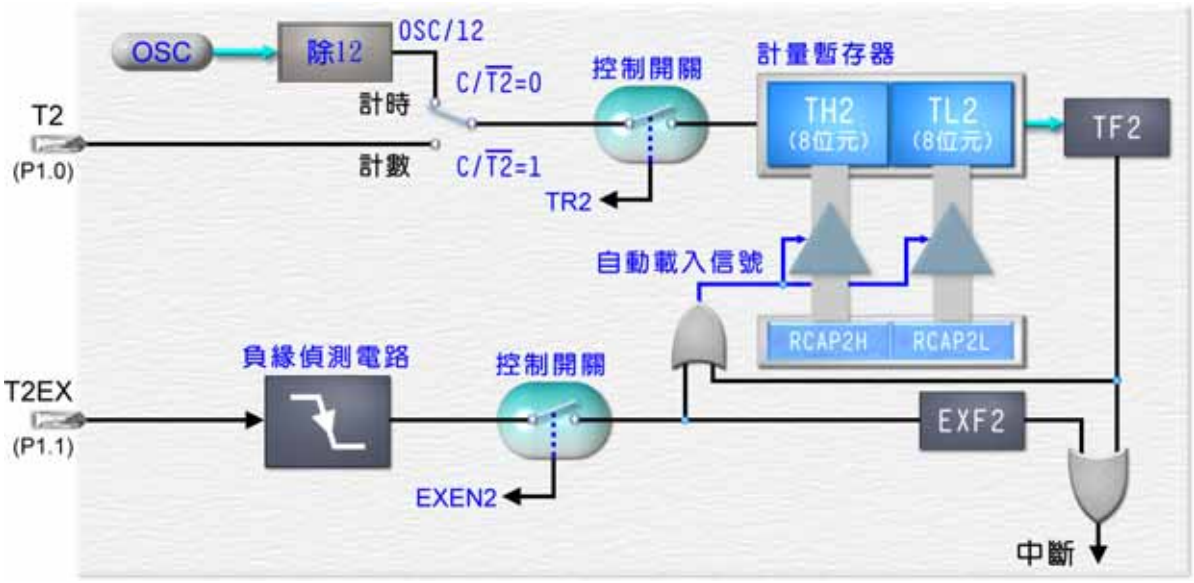
1. CP/$\overline{RL2}$ =1
2. EXEN2=1

再使 TR2=1，即可進入捕獲模式，Timer 2 即可計數。若 T2EX 接腳輸入信號中含有負緣，即啓動捕獲信號，同時產生 Timer 2 中斷。當 Timer

2 計數溢位，又產生 Timer 2 中斷。

6-2-3 自動載入模式

自動載入模式(Auto-Reload Mode)是自動將 RCAP2H 及 RCAP2L 暫存器的資料(16 位元)，載入 TH2 與 TL2 暫存器，其架構如下圖所示：



（圖11）**自動載入模式**

若要使用自動載入模式，必須將 T2CON 暫存器裡的CP/$\overline{RL2}$ 位元設定爲 0。Timer 2 的自動載入模式與 Timer 0、Timer 1 的 Mode2 類似，唯 Timer 0、Timer 1 的 Mode2 是 8 位元的自動載入功能，Timer 2 的自動載入模式則是 16 位元。同樣地，自動載入模式可計數內部時鐘脈波(OSC/12)，也可以計數由 T2 接腳輸入的外部脈波，只要將 T2CON 暫存器裡的C/$\overline{T2}$ 位元設定爲 0，則爲內部計時器；將 T2CON 暫存器裡的C/$\overline{T2}$ 位元設定爲 1，則爲外部計數器。另外，T2CON 暫存器裡的 EXEN2 位元也要設定爲 1，才能進行自動載入模式。而 Timer 2 的啓動開關爲 TR2，若將 TR2 設定爲 1，即可啓動 Timer 2；TR2=0，即可停用 Timer 2。

啓動 Timer 2 後，Timer 2 即進行計數工作，若偵測到 T2EX 接腳輸入信號中含有負緣，即啓動自動載入信號，將當時 RCAP2H 暫存器的內容，將被複製到 TH2 暫存器、RCAP2L 暫存器的內容，將被複製到 TL2 暫存器，同時 EXF2 位元設定爲 1，並產生 Timer 2 中斷。不過，Timer 2

的中斷並不影響計數的動作，待 Timer 2 計數溢位時，則 TF2 位元設定爲 1，並產生 Timer 2 中斷。

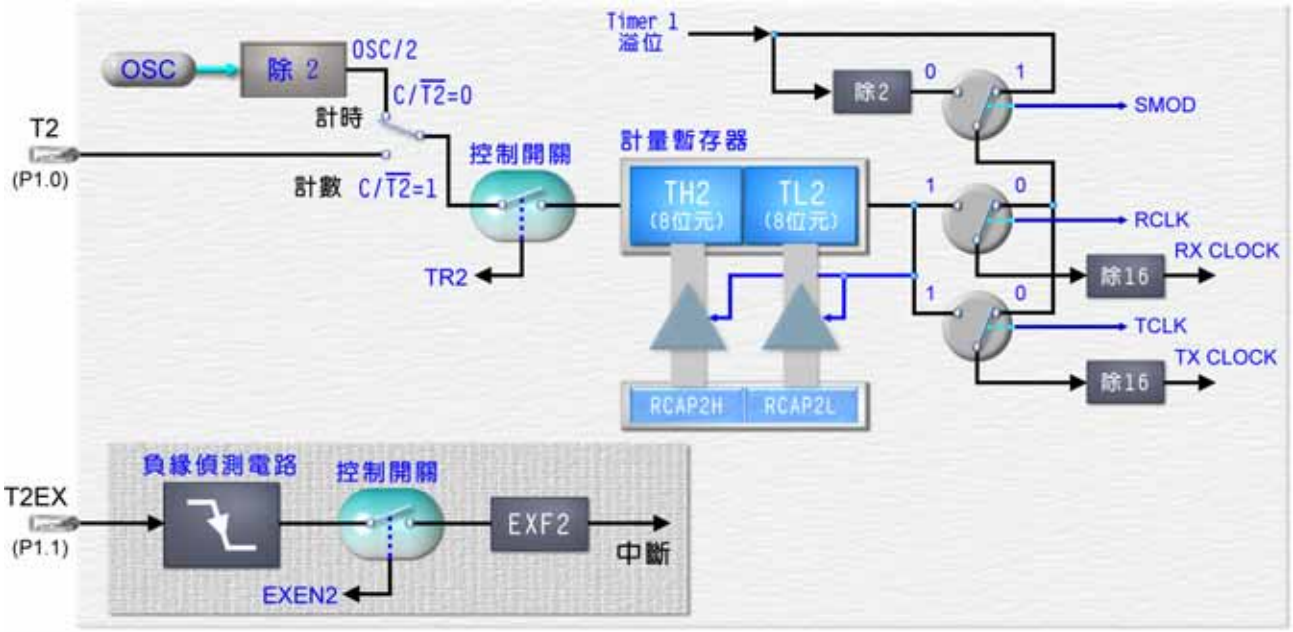
歸納上述，若要採自動載入模式工作，必須：

1. CP/$\overline{RL2}$ =0
2. EXEN2=1

再使 TR2=1，即可進入自動載入模式，Timer 2 即可計數。若 T2EX 接腳輸入信號中含有負緣，即啓動自動載入信號，同時產生 Timer 2 中斷。當 Timer 2 計數溢位，又產生 Timer 2 中斷。

6-2-4 鮑率產生器模式

MCS-51 的串列埠傳輸率(鮑率)可由 Timer 1 或 Timer 2 所產生的溢位脈波來控制，Timer 2 的鮑率產生器模式(Baud Rate Generator Mode)就是提供串列埠傳送與接收的時鐘脈波，其架構如下圖所示：



（圖12）**鮑率產生器模式**

如上圖所示，在鮑率產生器模式下，Timer 2 可分爲兩個獨立的部分，在下面的區塊裡，若 EXEN2 位元設定爲 1，則只要偵測到 T2EX 接腳上有負緣信號，即可使 EXF2 旗標設定爲 1，同時產生 Timer 2 中斷。

在其它部分則爲鮑率產生器，若要使用 Timer 2 鮑率產生器所產生的時鐘脈波，則須 T2CON 暫存器裡的 RCLK 位元或 TCLK 位元設定爲 1。當 RCLK=1，則 Timer 2 鮑率產生器將提供串列埠接收所須之時鐘脈波；TCLK=1，則 Timer 2 鮑率產生器將提供串列埠傳送所須之時鐘脈波。此外，T2CON 暫存器裡的 $C/\overline{T2}$ 位元設定爲 1，Timer 2 將計數由 T2 接腳所輸入之外部脈波信號，以產生溢位脈波；若 $C/\overline{T2}$ 位元設定爲 0，Timer 2 將計數由 OSC/2 之內部時鐘脈波信號，以產生溢位脈波，因此，其所產生的鮑率爲：

$$\frac{OSC/2}{16\times[65536-(RCAP2H,RCAP2L)]}$$

當然，Timer 2 的啓動開關還是爲 TR2，若將 TR2 設定爲 1，即可啓動 Timer 2；TR2=0，即可停用 Timer 2。

若 RCLK 位元或 TCLK 位元都不爲 1，則串列埠將採用 Timer 1 所產生的時鐘脈波。

6-3 布林運算指令

布林運算指令的功能是進行單一位元邏輯運算，其中包括 12 個指令，在此將它們分爲 6 大類來介紹：

清除指令

清除指令的功能是進位位元 CY 或可位元定址的某個位元，使之爲 0。

- CLR　C
  - **說明**：清除進位位元 CY，不管進位位元爲 1 或 0，經過本指令後，將變成 0，即 CY→CY=0。
  - **組譯後大小**：1 byte　　**執行時間**：12 個時鐘脈波
  - **範例**：

```
指令：CLR　C
執行後：CY=0
```

- CLR *bit*
  - **說明**：清除指定的位元 *bit*，不管 *bit* 爲 1 或 0，經過本指令後，將變成 0，即 *bit* → *bit* =0。
  - **組譯後大小**：2 bytes　　**執行時間**：12 個時鐘脈波
  - **範例**：

```
指令：CLR   PSW.0
執行後：PSW.0=0
```

設定指令

設定指令的功能是設定運算元，使之爲 1，其中運算元可爲進位位元 CY 或可位元定址的某個位元。

- SETB C
  - **說明**：設定進位位元 CY，不管 CY 爲 1 或 0，經過本指令後，將變成 1，即 CY→CY=1。
  - **組譯後大小**：1 byte　　**執行時間**：12 個時鐘脈波
  - **範例**：

```
指令：SETB   C
執行後：CY=1
```

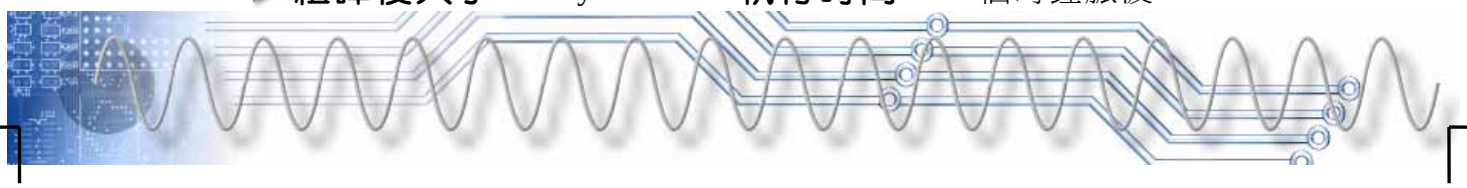
- SETB *bit*
  - **說明**：設定指定的位元 *bit*，不管 *bit* 爲 1 或 0，經過本指令後，將變成 1，即 *bit* → *bit* =1。
  - **組譯後大小**：2 bytes　　**執行時間**：12 個時鐘脈波
  - **範例**：

```
指令：SETB   RS1
執行後：RS1=0
```

補數指令

補數指令的功能是對進位位元 CY 或可位元定址的某個位元取補數。

- CPL C
  - **說明**：對進位位元 CY 取補數，也就是反相，即 CY→CY=$\overline{CY}$ 。
  - **組譯後大小**：1 byte　　**執行時間**：12 個時鐘脈波



▶**範例**：

```
指令：CPL    C
若執行前：CY=1
執行後：CY=0
```

- CPL　*bit*
  - ▶**說明**：對指定的位元 *bit* 取補數，也就是反相，即 *bit* →*bit* = $\overline{bit}$ 。
  - ▶**組譯後大小**：2 bytes　　**執行時間**：12 個時鐘脈波
  - ▶**範例**：

```
指令：CPL    P1.2
若執行前：P1.2=0
執行後：P1.2=1
```

AND 指令

AND 指令的功能是進位位元 CY 與指定的位元進行 AND 運算，而其結果將存回 CY。

- ANL　C, *bit*
  - ▶**說明**：將進位位元 CY 與 *bit* 進行 AND 運算，其結果存入 CY，即 CY • *bit* →CY。
  - ▶**組譯後大小**：2 bytes　　**執行時間**：24 個時鐘脈波
  - ▶**範例**：

```
指令：ANL    C, P2.1
若執行前：CY=1、P2.1=0
執行後：CY=0、P2.1=0
```

- ANL　C, */bit*
  - ▶**說明**：將進位位元 CY 與 *bit* 的補數進行 AND 運算，其結果存入 CY，即 CY • $\overline{bit}$ →CY。
  - ▶**組譯後大小**：2 bytes　　**執行時間**：24 個時鐘脈波
  - ▶**範例**：

指令：ANL C, /P2.1
若執行前：CY=1、P2.1=0
執行後：CY=1、P2.1=0

OR 指令

OR 指令的功能是進位位元 CY 與指定的位元進行 OR 運算，而其結果將存回 CY。

- ORL C, *bit*
  - **說明**：將進位位元 CY 與 *bit* 進行 ORD 運算，其結果存入 CY，即 CY + *bit* →CY。
  - **組譯後大小**：2 bytes **執行時間**：24 個時鐘脈波
  - **範例**：

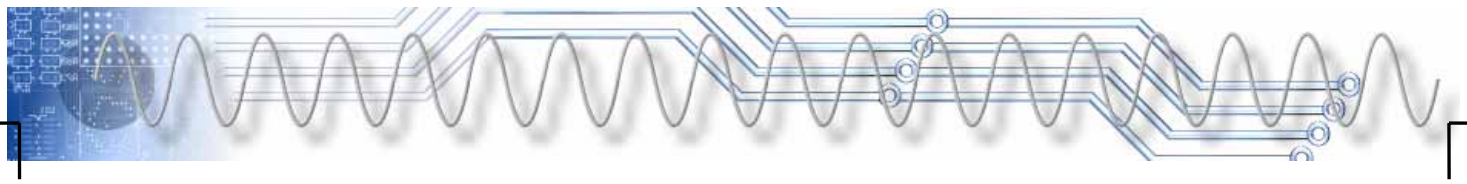
指令：ORL C, P2.1
若執行前：CY=1、P2.1=0
執行後：CY=1、P2.1=0

- ORL C, */bit*
  - **說明**：將進位位元 CY 與 *bit* 的補數進行 OR 運算，其結果存入 CY，即 CY + $\overline{bit}$ →CY。
  - **組譯後大小**：2 bytes **執行時間**：24 個時鐘脈波
  - **範例**：

指令：ORL C, /P2.1
若執行前：CY=1、P2.1=0
執行後：CY=1、P2.1=0

位元複製指令

位元複製指令的功能是將指定位元複製到進位位元 CY，或將進位位元 CY 複製到指定位元。



- MOV　C, *bit*
  - **說明**：將 *bit* 位元複製到進位位元 CY，即 *bit* → CY。
  - **組譯後大小**：2 bytes　　**執行時間**：12 個時鐘脈波
  - **範例**：

```
指令：MOV　C, P1.0
若執行前：CY=0、P1.0=1
執行後：CY=1、P1.0=1
```

- MOV　*bit*, C
  - **說明**：將進位位元 CY 複製到 *bit* 位元，即 CY → *bit*。
  - **組譯後大小**：2 bytes　　**執行時間**：24 個時鐘脈波
  - **範例**：

```
指令：MOV　P1.0, C
若執行前：CY=0、P1.0=1
執行後：CY=0、P1.0=0
```

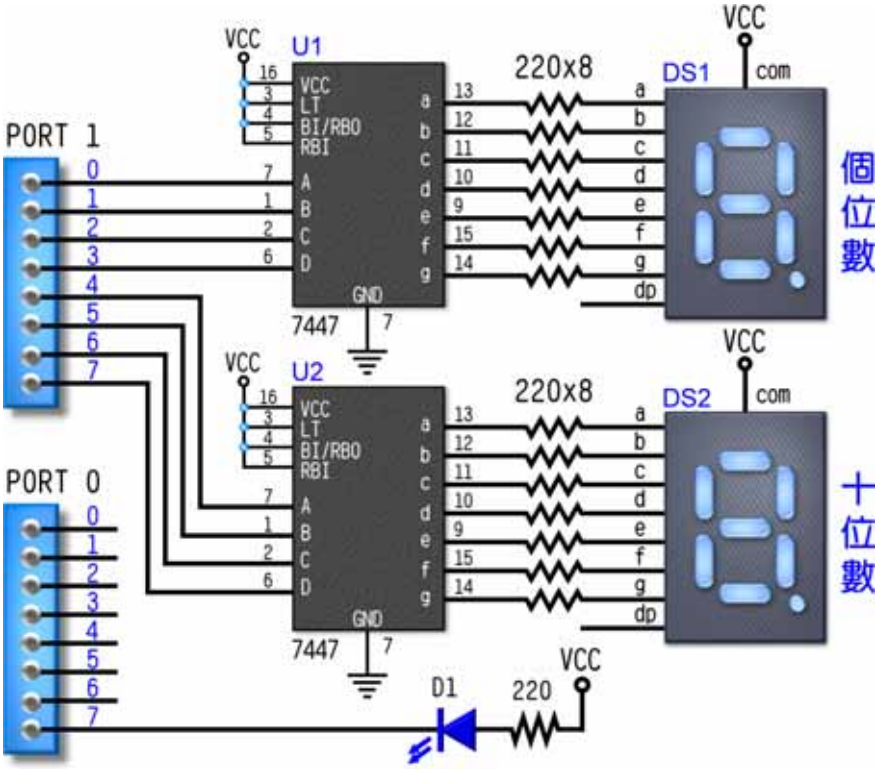
6-4 實例演練

在本單元裡提供四個範例，如下所示：

6-4-1 60 秒計時器實例演練之垂詢方式

- **功能說明**

如圖 13 所示，P1 的低四位元連接個位數之 7447 與七節顯示器，P1 的高四位元連接十位數之 7447 與七節顯示器。在此將利用 Timer 0 做爲計時裝置，兩個七節顯示器從「00」開始顯示，每 1 秒增加 1，到達「59」後，再從「00」開始，也就是 60 秒的計時器。每 60 秒，D1 切換一次(原本亮的，變成滅；原本不亮的，變成亮)。



（圖13）60 秒計時電路圖

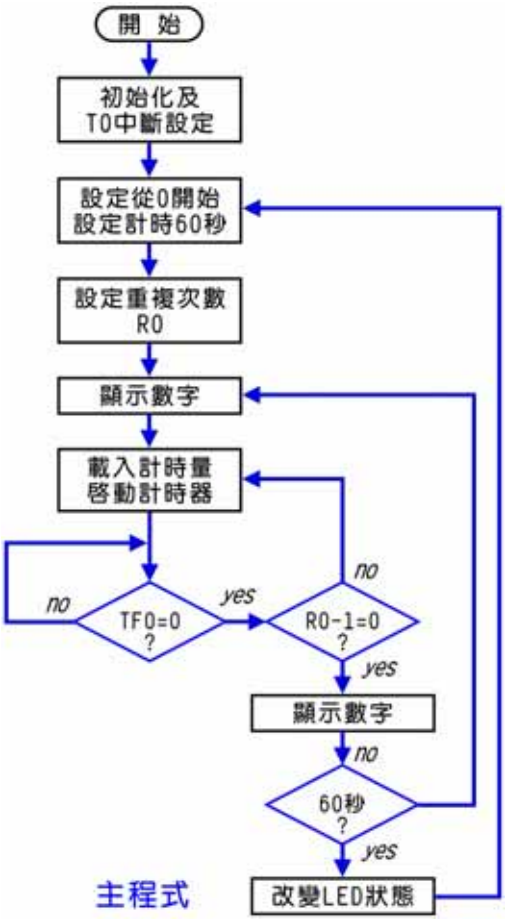
參考程式

若要使用 Timer 0 來計時，可使用 Mode 0、Mode 1、Mode 2 或 Mode 3，以 12MHz 的系統而言，如下說明：

1. 當在 Mode 0 模式工作時，每次最多可計數 8192，約 8 毫秒，若只**計數 5000**，則爲 5 毫秒，必須**重複 200 次**才延遲 1 秒鐘(5m×200)。
2. 當在 Mode 1 模式工作時，每次最多可計數 65536，約 65 毫秒，若只**計數 50000**，則爲 50 毫秒，必須**重複 20 次**才延遲 1 秒鐘(50m×20)。
3. 當在 Mode 2 或 Mode 3 模式工作時，每次最多可計數 256，約 0.25 毫秒，若只**計數 250**，則爲 0.25 毫秒，必須**重複 4000 次**才延遲 1 秒鐘(0.25m×4000)。

在此以 Mode 0 爲例，每次計數量爲 5000，而重複 200 次後，即將顯示數字增加 1。若顯示數字由 59 變爲 00 時，則以 CPL 指令改變

P0.7 所輸出的狀態。在此，將使用 EQU 及 REG 虛擬指令，EQU 指令是將程式裡含有其左邊字串的變數，以其右邊的數值代替，以「MODE　EQU　00H」爲例，如果在程式中，遇到 MODE，將以 00H 代替之。通常在左邊的字串都是比較具可讀性的文字，而右邊爲一些參數、常數等，表面上看不出所以然的數值，如此，一來可增程式的可讀性，二來若需要修改參數，只要在前面一個地方修改即可，而不必到程式中一個個替換。基本上，REG 指令與 EQU 指令類似，而其右邊爲一個記憶體位址、輸出入埠或暫存器。



```
MODE            EQU     00H             ;計時計數器模式(Mode 0)
COUNT           EQU     5000            ;計數量(5ms)
TIMES           EQU     200             ;重複次數
DISP            REG     P1              ;七節顯示器
LED             REG     P0.7            ;LED
;=============使用垂詢方式=================================
                ORG     0               ;程式從 0 位址開始
START:          MOV     DISP, #FFH      ;關閉七節顯示器
                CLR     LED             ;設定 LED 初始狀態
                MOV     TMOD, #MODE     ;設定計時計數器模式
LOOP:           MOV     R1, #0          ;設定七節顯示器初始數字
                MOV     R3, #60         ;設定計時 60 秒
NEXT:           MOV     R0, #TIMES      ;設定重複次數(200 次)
                MOV     A, R1           ;取回顯示數字
                DA      A               ;BCD 調整
                MOV     R1, A           ;存回數字
                MOV     DISP, A         ;顯示數字
AGAIN:          MOV     TH0, #(8192-COUNT)/32      ;設定計數量
                MOV     TL0, #(8192-COUNT).MOD.32 ;設定計數量
                SETB    TR0             ;啓動 Timer 0
;=========================================================
WAIT:           JBC     TF0, TIMEOUT    ;垂詢是否中斷
                JMP     WAIT
TIMEOUT:        CLR     TR0             ;關閉計時器
;-------------中斷-----------------------------------------
                DJNZ    R0, AGAIN       ;重複 200 次
;-------------1 秒鐘---------------------------------------
                INC     R1              ;數字加 1
                DJNZ    R3, NEXT        ;進行下一秒
                CPL     LED             ;改變 LED 狀態
                JMP     LOOP            ;跳至 LOOP 形成一個迴圈
                END
```

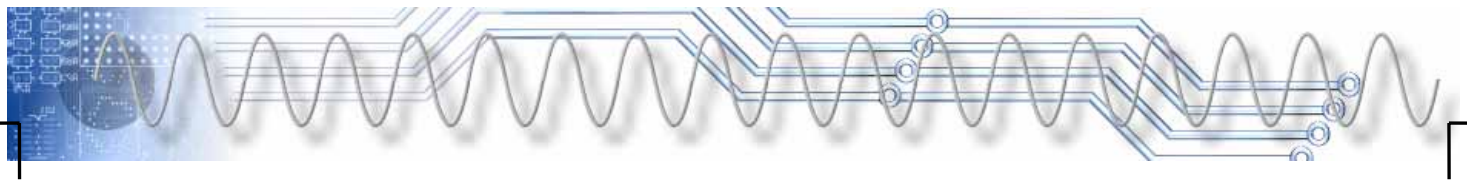
計時器實驗(ch6-1.asm)

若要使用 Mode 1 爲例，且每次計數量爲 50000，而重複 20 次後，即將顯示數字增加 1。則只要將上述程式的前三列改如下：

```
MODE            EQU     01H             ;計時計數器模式(Mode 1)
COUNT           EQU     -50000          ;計數量
TIMES           EQU     20              ;重複次數
```

再把 AGAIN 標籤那一列及其下一列改如下：

```
AGAIN:          MOV     TH0, #>COUNT    ;設定計數量
                MOV     TL0, #<COUNT    ;設定計數量
```



操作

1. 依功能需求與電路結構撰寫程式，然後將該程式組譯與連結，以產生*.HEX 檔。
2. 請按圖 13 連接線路，再使用實體模擬器，載入該程式(*.HEX)，以模擬該電路的動作。若有非預期的狀況，則檢視線路的連接狀況，看看哪裡出問題？並將它記錄在實驗報告裡。
3. 若實體模擬功能正常，請將程式燒錄到 89C51，再把該 89C51 放入實體電路，以取代剛才的實體模擬器，然後直接送電，看看是否正常？
4. 撰寫實驗報告。

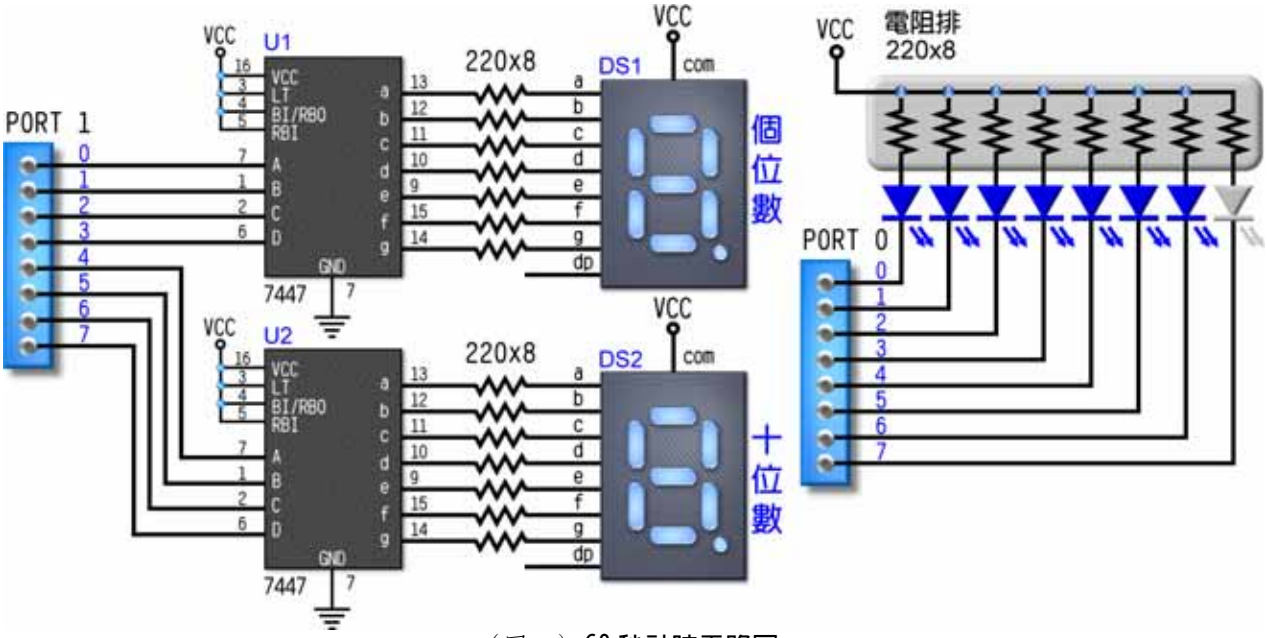
思考一下

1. 在本實驗裡，所採用的是 Timer 0，若要採用 Timer 1，應如何修改？
2. 若要使用 Mode 2 或 Mode 3 來完成本實驗的功能，程式應如何修改？

6-4-2 60 秒計時器實例演練之中斷方式

功能說明

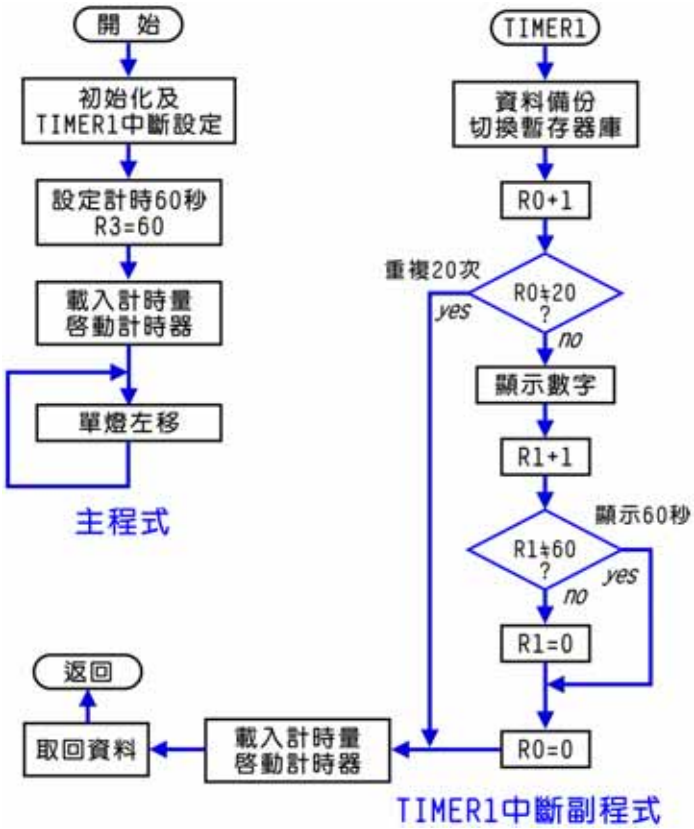
在 6-4-1 節中的程式是以垂詢(polling)的方式，主程式就一直執行「WAIT:　JBC　TF0, TIMEOUT」，而沒有所謂的中斷向量與中斷副程式，看似簡單，但，整個程式就陷入「垂詢」狀態，而無法執行其它的動作。就像是老師在課堂上，從第一分鐘就開始問「有沒有問題」，直到下課，亂沒有效率的！如圖 14 所示，基本上這個電路圖與圖 13 類似，只是 P0 接了 8 個 LED，我們希望主程式在 P0 上執行單燈左移的功能，而 Timer 0 中斷才去服務 60 秒的計時工作。



（圖14）60 秒計時電路圖

參考程式

依功能需求與電路結構得知，主程式執行單燈左移功能。Timer 1 中斷副程式則先判斷是否已重複中斷 20 次，若未達 20 次，則繼續啓動 Timer 1 功能。若已達 20 次，表示已經過 1 秒，將七節顯示器的數字增加 1。然後判斷是否已達 60 秒，若未達 60 秒則只須將重複次數歸零，再繼續啓動 Timer 1 功能。若已達 60 秒，則將顯示數字與重複次數都歸零，再繼續啓動 Timer 1 功能。



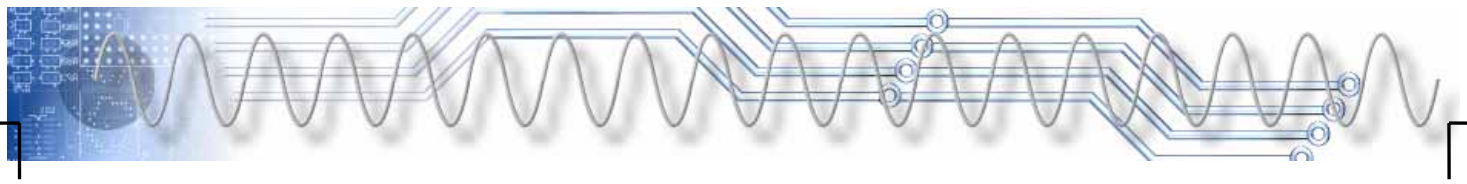

```
MODE            EQU     10H             ;計時計數器模式(Mode 1)
COUNT           EQU     -50000          ;計數量
TIMES           EQU     20              ;重複次數
DISP            REG     P1              ;七節顯示器
LED             REG     P0              ;LED
;=============使用中斷方式=================================
                ORG     0               ;程式從 0 位址開始
                JMP     START           ;跳至 START
                ORG     1BH             ;設定中斷向量
                JMP     TIMER1          ;執行中斷副程式
START:          MOV     DISP, #FFH      ;關閉七節顯示器
                SETB    EA              ;打開中斷總開關
                SETB    ET1             ;打開 Timer 1 中斷開關
                MOV     TMOD, #MODE     ;設定計時計數器模式
                MOV     SP, #70H        ;移開堆疊位置
                SETB    RS0             ;切換到 RB1 暫存器庫
                MOV     R1, #0          ;設定七節顯示器初始數字
                MOV     R0, #0          ;設定已重複次數為 0
                MOV     R3, #60         ;設定計時 60 秒
                MOV     DISP, #FFH      ;關閉數字
                MOV     TH1, #>COUNT    ;設定計數量
                MOV     TL1, #<COUNT    ;設定計數量
                SETB    TR1             ;啓動 Timer 1
```

```
;============單燈左移(主程式功能)=======================
                CLR     RS0             ;切換回 RB0 暫存器庫
                MOV     A, #FEH         ;單燈左移初始值
L_LOOP:         MOV     LED, A          ;輸出到 LED
                CALL    DELAY           ;呼叫延遲副程式
                RL      A               ;左移
                JMP     L_LOOP          ;跳至 L_LOOP 形成一個迴圈
;============主程式結束==================================
;============60 秒計時(中斷副程式開始)====================
TIMER1:         PUSH    PSW             ;儲存程式狀態字組暫存器
                PUSH    A               ;儲存 ACC
                SETB    RS0             ;切換到 RB1
                CLR     RS1             ;切換到 RB1
                INC     R0              ;R0+1(重複次數增加 1 次)
                CJNE    R0, #TIMES, AGAIN    ;是否已重複 20 次
;------------1 秒鐘-----------------------------------
                MOV     A, R1           ;取回顯示數字
                DA      A               ;BCD 調整
                MOV     R1, A           ;存回數字
                MOV     DISP, A         ;顯示數字
                INC     R1              ;數字加 1
                DJNZ    R3, NEXT        ;是否已顯示 60 秒
                MOV     R3, #60         ;從頭開始
                MOV     R1, #0          ;從 00 秒開始
NEXT:           MOV     R0, #0          ;重新計數下一秒鐘
AGAIN:          MOV     TH1, #>COUNT    ;設定計數量
                MOV     TL1, #<COUNT    ;設定計數量
                SETB    TR1             ;啓動 Timer 1
                POP     A               ;取回 ACC
                POP     PSW             ;取回程式狀態字組暫存器
                RETI
;============60 秒計時(中斷副程式結束)====================
;============DELAY 副程式開始============================
DELAY:          MOV     R7, #200
D1:             MOV     R6, #250
                DJNZ    R6, $
                DJNZ    R7, D1
                RET
;============DELAY 副程式結束============================
                END
```

計時器實驗(ch6-2.asm)

操作

1. 依功能需求與電路結構撰寫程式，然後將該程式組譯與連結，以產生*.HEX 檔。

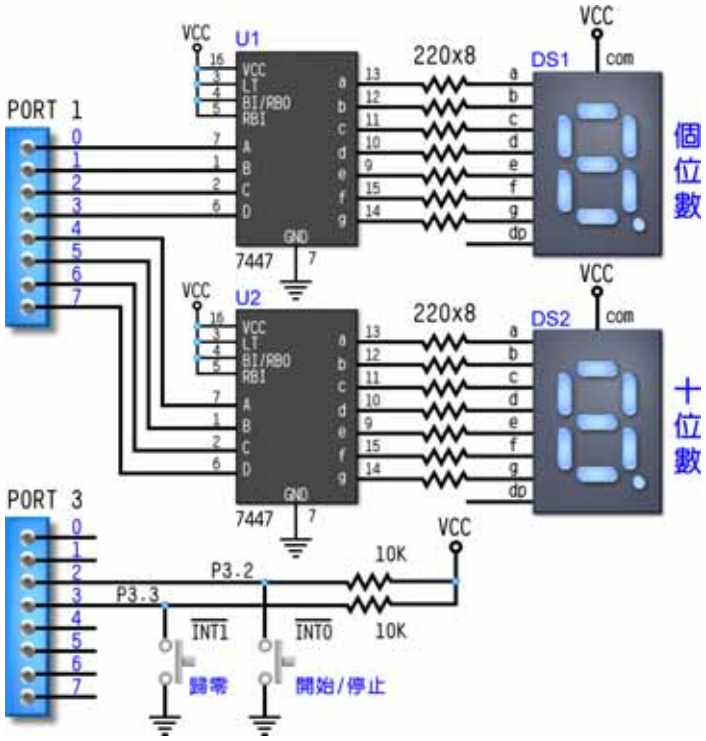


2. 請按圖 14 連接線路，再使用實體模擬器，載入該程式(*.HEX)，以模擬該電路的動作。若有非預期的狀況，則檢視線路的連接狀況，看看哪裡出問題？並將它記錄在實驗報告裡。
3. 若實體模擬功能正常，請將程式燒錄到 89C51，再把該 89C51 放入實體電路，以取代剛才的實體模擬器，然後直接送電，看看是否正常？
4. 撰寫實驗報告。

思考一下

1. 請將本實驗裡，主程式的功能改成霹靂燈功能？
2. 請修改本實驗的程式，以採用 Mode 2 自動載入的功能？

6-4-3 碼表實例演練



（圖15）**簡易碼表電路圖**

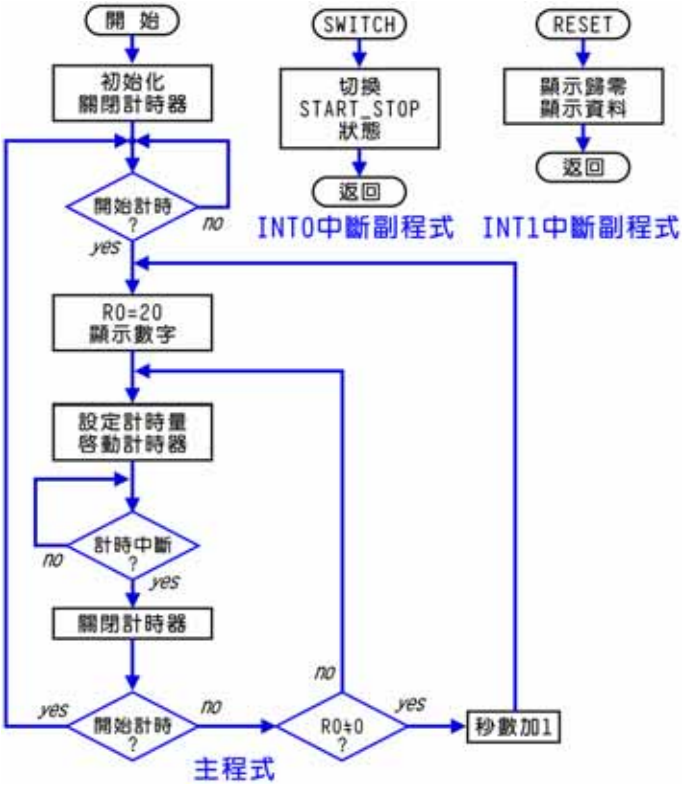
功能說明

如圖 15 所示，INT0 所接的按鈕開關具有啓動碼表及停止碼表的功能，按一下 INT0 按鈕開關，即可開始計時，同樣地，七節顯示器

上每秒增加 1；再按一下 INT0 按鈕開關，即可停止計時。INT1 所接的按鈕開關的功能是將碼表歸零，按一下 INT0 按鈕開關，則不管有沒有計時，七節顯示器將從 00 開始。

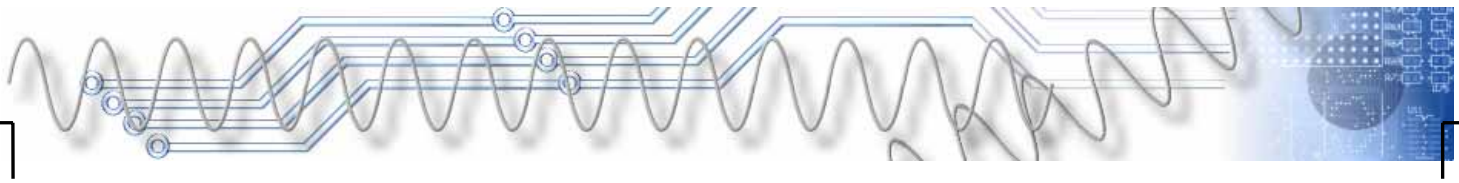
參考程式

基本上，在此所採用的 1 秒鐘計時與顯示，與 6-4-1 節類似。而在本單元最主要的是設計一個具有切換功能(Toggle)的中斷，也就是第一次 INT0 中斷時，開始計時；第二次 INT0 中斷時，停止計時；第三次 INT0 中斷時，又開始計時...，就像一個切換開關一樣。爲達此目的，在此利用一個位元(即 21H.0)，每次 INT0 中斷時，就改變其狀態。而 1 秒鐘計時與顯示裡，每次 Timer 0 中斷時，就檢查一次 21H.0，若 21H.0=1，則繼續計時與顯示；若 21H.0=0，則停止計時與顯示。至於 INT1 中斷所要做的事，只是將七節顯示器所對應的記憶體位置(在此設定爲 20H)清除爲 0 而已。




```
MODE    EQU   01H          ;計時計數器模式(Mode 1)
COUNT   EQU   -50000       ;計數量
TIMES   EQU   20           ;重複次數
DATAS   REG   20H          ;顯示資料所放置的位置
```

```
START_STOP    REG     21H.0           ;啓閉狀態所放置的位置
DISP          REG     P1              ;七節顯示器
;=============使用垂詢方式=================================
              ORG     0               ;程式從 0 位址開始
              JMP     START           ;跳至 START
              ORG     03H             ;設定中斷向量
              JMP     SWITCH          ;執行 SWITCH 中斷副程式
              ORG     13H             ;設定中斷向量
              JMP     RESET           ;執行 RESET 中斷副程式
START:        MOV     DISP, #FFH      ;關閉七節顯示器
              SETB    EA              ;打開中斷總開關
              SETB    EX0             ;打開 INT0 中斷開關
              SETB    EX1             ;打開 INT1 中斷開關
              SETB    PX1             ;提高 INT1 中斷之優先等級
              SETB    IT0             ;設定 INT0 負緣觸發
              SETB    IT1             ;設定 INT1 負緣觸發
              MOV     TMOD, #MODE     ;設定為 Timer 0、Mode 1
              SETB    START_STOP      ;關閉計時器
              MOV     DATAS, #0       ;從 0 開始
              CLR     TR0
STOP:         JNB     START_STOP, $   ;檢查是否開始計時
              CLR     C               ;清除 CY 旗標
;-----------------------------------------------------------
;=============100 秒計時(中斷副程式開始)====================
;-----------------------------------------------------------
NEXT:         MOV     R0, #TIMES      ;設定重複次數(20 次)
              MOV     A, DATAS        ;取回顯示資料
              DA      A               ;BCD 調整
              MOV     DATAS, A        ;取回顯示資料
              MOV     DISP, A         ;顯示秒數
AGAIN:        MOV     TH0, #>COUNT    ;設定計數量
              MOV     TL0, #<COUNT    ;設定計數量
              SETB    TR0             ;啓動 Timer 0
;===========================================================
WAIT:         JBC     TF0, TIMEOUT    ;垂詢是否中斷
              JMP     WAIT
TIMEOUT:      CLR     TR0             ;關閉計時器
;-------------中斷-----------------------------------------
              JB      START_STOP, STOP     ;檢查是否繼續計時
              DJNZ    R0, AGAIN       ;重複 20 次
;-------------1 秒鐘---------------------------------------
              INC     DATAS           ;數字資料增加 1
              JMP     NEXT            ;跳至 NEXT 形成一個迴圈
;===========================================================
;
;=============INT 0 中斷副程式開始(啓閉)====================
SWITCH:       CPL     START_STOP      ;切換啓閉開關
```



```
                RETI
;=============INT 0 中斷副程式結束(啓閉)====================
;
;=============INT 1 中斷副程式開始(歸零)====================
RESET:          MOV     DATAS, #0       ;清除顯示資料
                MOV     DISP, DATAS     ;顯示資料
                RETI
;=============INT 1 中斷副程式結束(歸零)====================
                END
```

碼表實驗(ch6-3.asm)

操作

1. 依功能需求與電路結構撰寫程式，然後將該程式組譯與連結，以產生*.HEX 檔。
2. 請按圖 15 連接線路，再使用實體模擬器，載入該程式(*.HEX)，以模擬該電路的動作。若有非預期的狀況，則檢視線路的連接狀況，看看哪裡出問題？並將它記錄在實驗報告裡。
3. 若實體模擬功能正常，請將程式燒錄到 89C51，再把該 89C51 放入實體電路，以取代剛才的實體模擬器，然後直接送電，看看是否正常？
4. 撰寫實驗報告。

思考一下

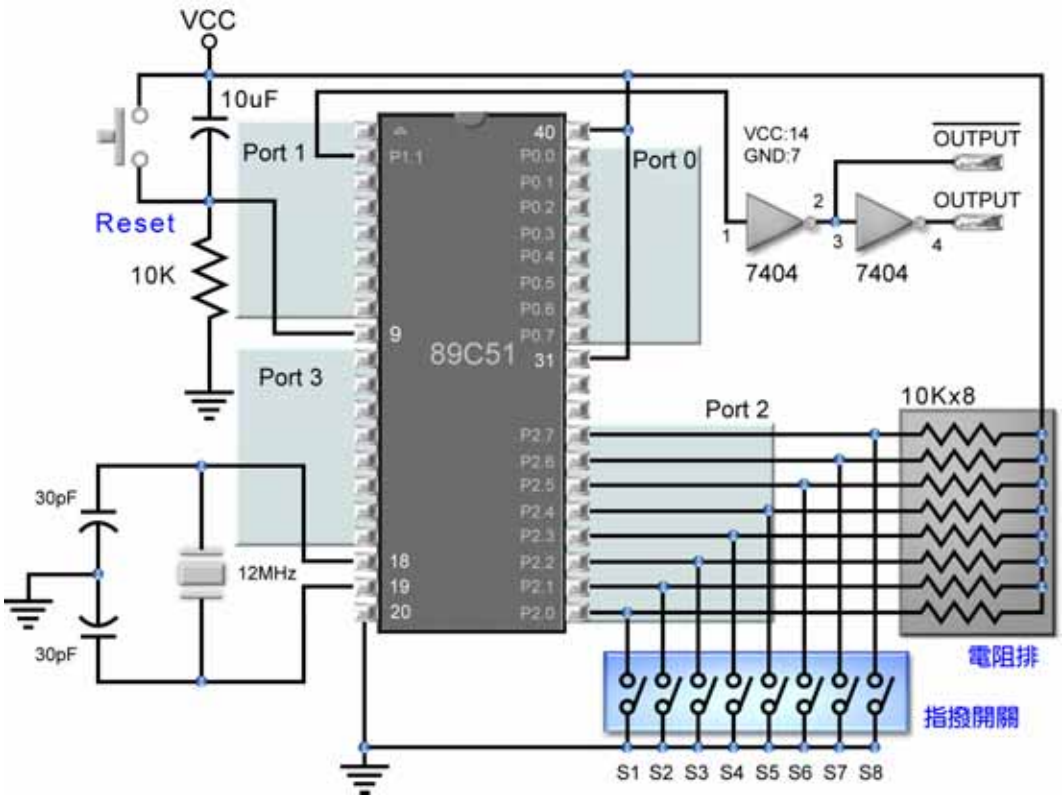
1. 在本實驗裡所使用 Timer 0，是以垂詢方式，請修改成以中斷方式執行？
2. 在此我們將 INT1 中斷的優先等級提高，若沒有做這項設定，將會有什麼結果？

6-4-4 頻率產生器實例演練

功能說明

如圖 16 所示，指撥開關的狀態由 P2 輸入，在此將利用其中的 S1、S2、S3 及 S4 設定輸出的頻率。若 S1 開關 on 的話，不管其它開關爲何，OUTPUT 端輸出 100KHz；若 S1 開關 off、S2 開關 on 的話，不管其它開關爲何，OUTPUT 端輸出 10KHz；若 S1 開關 off、S2

開關 off、S3 開關 on 的話，不管其它開關爲何，OUTPUT 端輸出 1KHz；若 S1 開關 off、S2 開關 off、S3 開關 off、S1 開關 on 的話，不管其它開關爲何，OUTPUT 端輸出 100Hz，如下表所示：



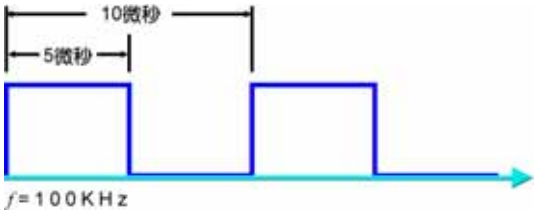
（圖16）**簡易頻率產生器電路圖**

S1	S2	S3	S4	OUTPUT
0	×	×	×	100KHz
1	0	×	×	10KHz
1	1	0	×	1KHz
1	1	1	0	100Hz

開關 on 爲 0、off 爲 1

參考程式

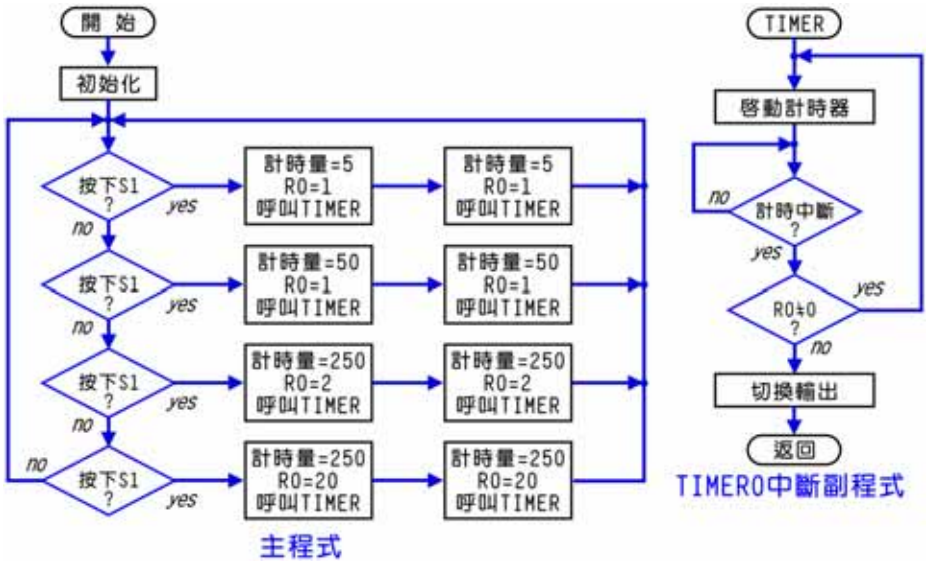
若 *f*=100KHz，則 T=10 微秒，輸出脈波每 5 微秒就要變化一次，如下所示：



同理，10KHz、1KHz、100Hz 的信號，整理如下表所示：

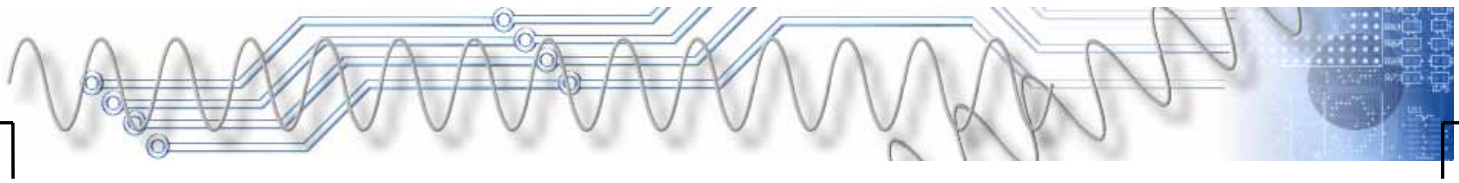
頻率	週期	變化時間	R0	COUNT
100KHz	10μs	5μs	1	COUNT 1=5
10KHz	100μs	50μs	1	COUNT 2=50
1KHz	1000μs	500μs	TIMES1=2	COUNT 3=250
100Hz	10000μs	5000μs	TIMES2=20	COUNT 3=250

因此，在此撰寫一個計時副程式 TIMER，其計時的時間是 R0×TH0 微秒，而 TH0 的計數量分別是 5、50、250 及 250，R0 的數量分別是 1、1、2 及 20，如此就可的到 5μs、50μs、500μs 及 5000μs 的計時。每完成一個計時，就將輸出反相一次。




```
MODE        EQU     02H             ;計時計數器模式(Mode 2)
COUNT1      EQU     -5              ;計數量 1
COUNT2      EQU     -50             ;計數量 2
COUNT3      EQU     -250            ;計數量 3
TIMES1      EQU     2               ;重複次數 1
TIMES2      EQU     20              ;重複次數 2
OUTPUT      REG     P1.1            ;輸出脈波信號
```

```
;============使用垂詢方式=================================
                ORG     0               ;程式從 0 位址開始
                JMP     START           ;跳至 START
START:          CLR     OUTPUT          ;設定輸出初始狀態
                MOV     TMOD, #MODE     ;設定計時計數器模式
                MOV     SP, #70H        ;移開堆疊位置
LOOP:           JNB     P2.0, S1        ;偵測 S1 開關
                JNB     P2.1, S2        ;偵測 S2 開關
                JNB     P2.2, S3        ;偵測 S3 開關
                JNB     P2.3, S4        ;偵測 S4 開關
                JMP     LOOP            ;跳至 LOOP 形成一個迴路
;============(輸出 100KHz)================================
S1:             MOV     TH0, #COUNT1    ;設定計數量(5)
                MOV     TL0, #COUNT1    ;設定計數量(5)
                MOV     R0, #1          ;執行一次計時動作
                CALL    TIMER           ;執行 TIMER 計時副程式
                MOV     R0, #1          ;執行一次計時動作
                CALL    TIMER           ;執行 TIMER 計時副程式
                JMP     LOOP            ;跳回 LOOP 形成一個迴路
;============(輸出 10KHz)=================================
S2:             MOV     TH0, #COUNT2    ;設定計數量(50)
                MOV     TL0, #COUNT2    ;設定計數量(50)
                MOV     R0, #1          ;執行一次計時動作
                CALL    TIMER           ;執行 TIMER 計時副程式
                MOV     R0, #1          ;執行一次計時動作
                CALL    TIMER           ;執行 TIMER 計時副程式
                JMP     LOOP            ;跳回 LOOP 形成一個迴路
;============(輸出 1KHz)==================================
S3:             MOV     TH0, #COUNT3    ;設定計數量(250)
                MOV     TL0, #COUNT3    ;設定計數量(250)
                MOV     R0, #TIMES1     ;設定重複次數(2)
                CALL    TIMER           ;執行 TIMER 計時副程式
                MOV     R0, #TIMES1     ;設定重複次數(2)
                CALL    TIMER           ;  執行 TIMER 計時副程式
                JMP     LOOP            ;跳回 LOOP 形成一個迴路
;============(輸出 100Hz)=================================
S4:             MOV     TH0, #COUNT3    ;設定計數量(250)
                MOV     TL0, #COUNT3    ;設定計數量(250)
                MOV     R0, #TIMES2     ;設定重複次數(20)
                CALL    TIMER           ;執行 TIMER 計時副程式
                MOV     R0, #TIMES2     ;設定重複次數(20)
                CALL    TIMER           ;執行 TIMER 計時副程式
                JMP     LOOP            ;跳回 LOOP 形成一個迴路
;============(計時副程式)=================================
TIMER:          SETB    TR0             ;啓動 Timer 0
WAIT:           JBC     TF0, TIMEOUT    ;垂詢是否中斷
                JMP     WAIT
```



```
TIMEOUT:    CLR    TR0             ;關閉計時器
            DJNZ   R0, TIMER       ;重複執行 R0 次
            CPL    OUTPUT          ;切換輸出狀態
            RET
;=========================================================
            END
```

頻率產生器實驗(ch6-4.asm)

操作

1. 依功能需求與電路結構撰寫程式，然後將該程式組譯與連結，以產生*.HEX 檔。
2. 請按圖 16 連接線路，再使用實體模擬器，載入該程式(*.HEX)，以模擬該電路的動作。若有非預期的狀況，則檢視線路的連接狀況，看看哪裡出問題？並將它記錄在實驗報告裡。
3. 若實體模擬功能正常，請將程式燒錄到 89C51，再把該 89C51 放入實體電路，以取代剛才的實體模擬器，然後直接送電，看看是否正常？
4. 撰寫實驗報告。

思考一下

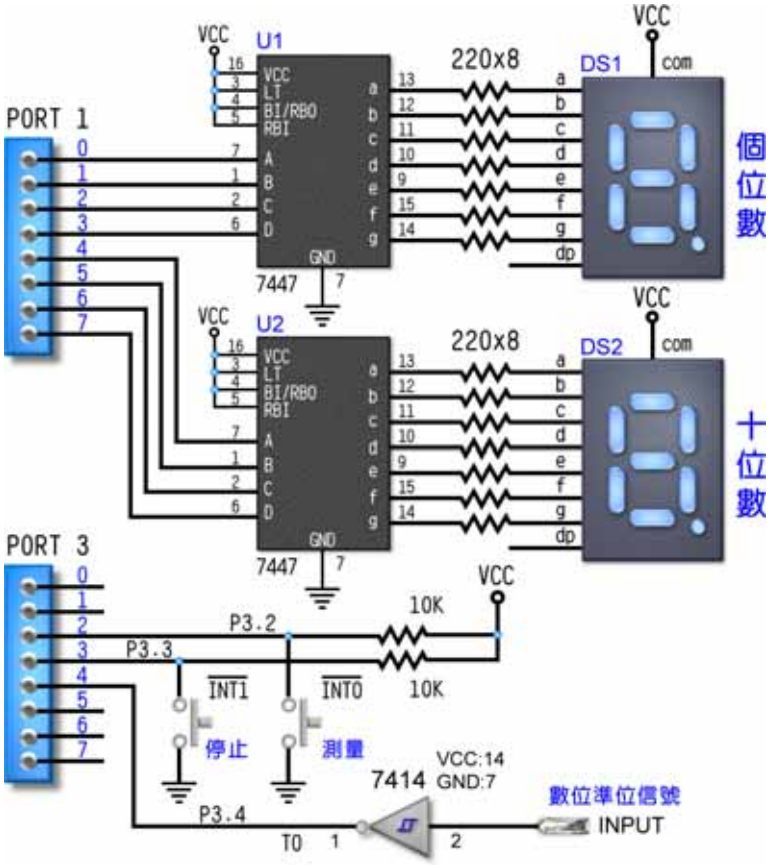
1. 在本實驗裡所使用的指撥開關，在此將它們的優先等級設成 S1>S2>S3>S4，也就是 S1 開關 on 的話，其它開關的狀態就不考慮；S1 開關 off、S2 開關 on、的話，其它開關的狀態就不考慮...，以此類推，是不是以使用波段開關來代替，會比較適切？如何修改電路？
2. 試使用兩位數的 BCD 指撥開關，做爲頻率的設定，以設計一個能夠提供 1K 到 99K Hz 的頻率產生器？

6-4-5 計頻器實例演練

功能說明

如圖 17 所示，所要測試的信號由 INPUT 輸入數位信號，經過 7414 反閘，接到 P3.4(即 T0 接腳)。另外，P3.2 連接 INT0 按鈕開關、P3.3

連接 INT1 按鈕開關。按一下 INT0 按鈕開關，則進行頻率測試、按一下 INT1 按鈕開關，則停止頻率測試。兩個七節顯示器經 7447 連接到 PORT 1，其中高四位元爲十位數、低四位元爲個位數。在此希望能正確量測出 1KHz 到 99KHz 之頻率，並將它顯示出來，若輸入 1K 到 1.999K 之頻率，則顯示 01、2K 到 2.999K 之頻率，則顯示 02…，以此類推。若低於 1KHz，則不顯示。若高 100KHz，則只顯示 100KHz 以下部分。

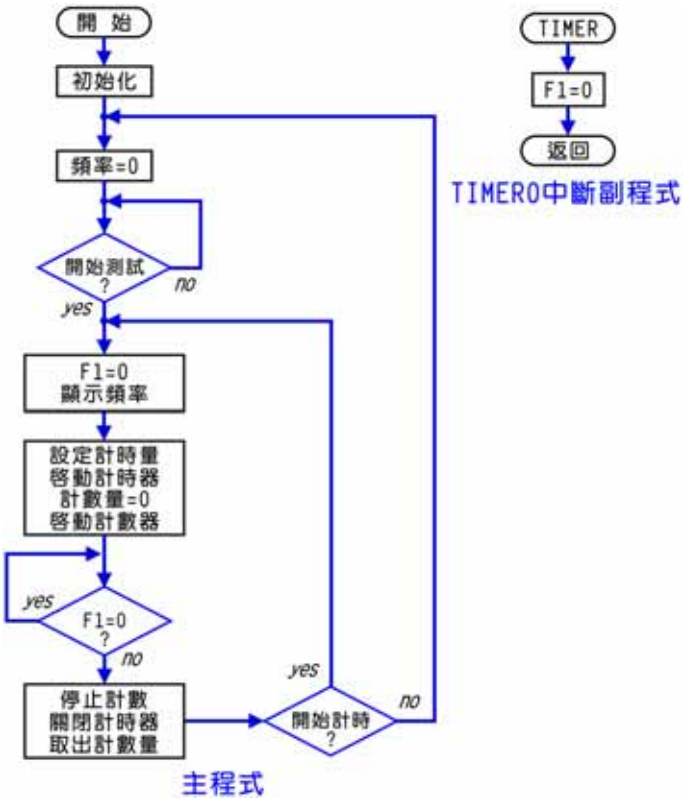


（圖17）**簡易計頻器電路圖**

參考程式

在此是利用一個計時器與一個計數器，計時器每 1 毫秒鐘中斷一次，開始計時時，同時開始計數 INPUT 端的脈波；計時器中斷時，即停止計數器，再把計數的結果，送到七節顯示器。由於計數的週期 T=1ms，所以計數的結果就是千赫(f=1/T KHz)。Timer 1 將設定爲 mode 1 的計數器、Timer 0 將設定爲 mode 1 的計時器，如下所示：

Timer 0 計時器採中斷方式，Timer 1 計數器採垂詢方式。




```
MODE            EQU     01010001B       ;Timer1：mode 1、計數器
                                        ;Timer0：mode 1、計時器
COUNT           EQU     -1000           ;計時量(1ms)
DATA            EQU     20H             ;頻率放置位址
DISP            REG     P1              ;七節顯示器
;=============================================================
                ORG     0               ;程式從 0 位址開始
                JMP     START           ;跳至 START
                ORG     0BH             ;設定中斷向量
                JMP     TIMER           ;執行 TIMER 中斷副程式
START:          MOV     DISP, #FFH      ;關閉七節顯示器
                MOV     IE, #10001010B  ;打開中斷開關
                MOV     TMOD, #MODE     ;設定計時計數器模式
                MOV     SP, #70H        ;移開堆疊位置
```

```
;=========================================================
S1:             MOV     DATA, #0        ;設定七節顯示器初始數字
                JB      P3.2, $         ;開始測試嗎？
LOOP:           CLR     F1              ;清除 F1 旗標
                MOV     A, DATA         ;取回數字資料
                DA      A               ;BCD 調整
                MOV     DISP, A         ;顯示頻率
                MOV     TH0, #>COUNT    ;設定計時量(1ms)
                MOV     TL0, #<COUNT    ;設定計時量(1ms)
                SETB    TR0             ;啓動 Timer 0 計時器
                MOV     TH1, #0         ;歸零計數量
                MOV     TL1, #0         ;歸零計數量
                SETB    TR1             ;啓動 Timer 1 計數器
                JNB     F1, $           ;等待 1ms
                CLR     TR1             ;停止計數器
                CLR     TR0             ;停止計時器
                MOV     DATA, TL1       ;取出計數量
                JB      P3.3, LOOP      ;停止測試嗎？
                JMP     S1              ;跳至 S1 形成一個迴圈
;=============TIMER 中斷副程式==============================
TIMER:          SETB    F1
                RETI
;=========================================================
                END
```

計頻器實驗(ch6-5.asm)

操作

1. 依功能需求與電路結構撰寫程式，然後將該程式組譯與連結，以產生*.HEX 檔。
2. 請按圖 17 連接線路，再使用實體模擬器，載入該程式(*.HEX)，以模擬該電路的動作。若有非預期的狀況，則檢視線路的連接狀況，看看哪裡出問題？並將它記錄在實驗報告裡。
3. 若實體模擬功能正常，請將程式燒錄到 89C51，再把該 89C51 放入實體電路，以取代剛才的實體模擬器，然後直接送電，看看是否正常？
4. 撰寫實驗報告。

思考一下

1. 分析本實驗的程式，然後畫出其程式流程圖。
2. 在本實驗裡，爲何只取用 TL1 暫存器裡的資料，做爲所要顯示的

頻率？

3. 若要使計頻器更精細，應如何處理？

6-5 即時練習

在本章裡探討 8051 的計時計數器，包括計時計數器的結構與設定、計時計數器的四種模式、與計時計數器相關的暫存器、計時計數程式的撰寫等，以及布林運算指令，內容非常豐富！在此請試著回答下列問題，以確認對於此部分的認識程度。

1. 試述在 12MHz 的 8051 系統裡，計時計數器的四種工作模式，每種模式最多可計時多少時間？
2. 在 8051 的指令裡，若要使用計時計數器，做爲外不計數之用，除了工作模式的選擇外，最關鍵性的設定爲何？
3. 試說明何謂「自動載入」功能？在 8051 的計時計數器裡，哪一種工作模式可提供此項功能？
4. 在 6-4 節的實例演練裡，所應用 8051 計時計數器的方式，包括「垂詢方式」及「中斷方式」，試說明此兩種方式的異同？
5. 若要使用 mode 0，試說明設定其計時計數量的方式？
6. 若要使用 mode 1，試說明設定其計時計數量的方式？
7. 若要使用 mode 2，試說明設定其計時計數量的方式？
8. 試以圖形說明 mode 3 的工作模式與架構？
9. 在 12MHz 的 8051 系統裡，若要使用 mode 0 產生 0.5 秒的延遲，程式應如何撰寫？
10. 試說明「CPL」與「CLR」指令的功能？