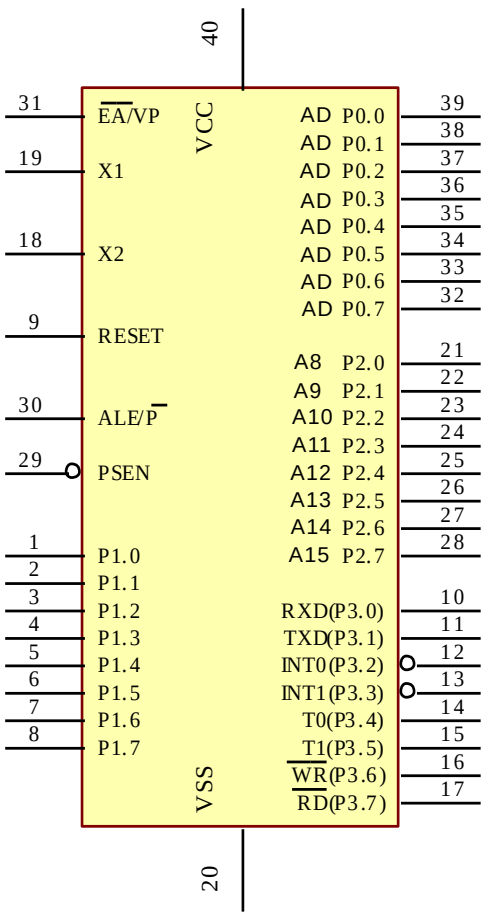


1. P0.0～P0.7 ------ 當成一般 I/O 埠(Port)時，須外接 10KΩ 提升電阻，名為 Port 0。若有外部擴充記憶體時，則其作用為低位元組的 Adress Bus 及 8 位元的 Data Bus。

2. P1.0～P1.7 ------ 一般 I/O 埠(Port)，名為 Port 1。

3. P2.0～P2.7 ------ 當成一般 I/O 埠，名為 Port2。若有外部擴充記憶體時，則其作用為高位元組的 Adress Bus。

4. P3.0～P3.7 ------ 當成一般 I/O 埠，名為 Port3。每支腳兼具第二功能。

5. 第 31 腳 --------- 一般接 5V，若接 0 時，則一律至外部 ROM 提取機械碼執行。在燒錄時則為燒錄電壓供應腳。

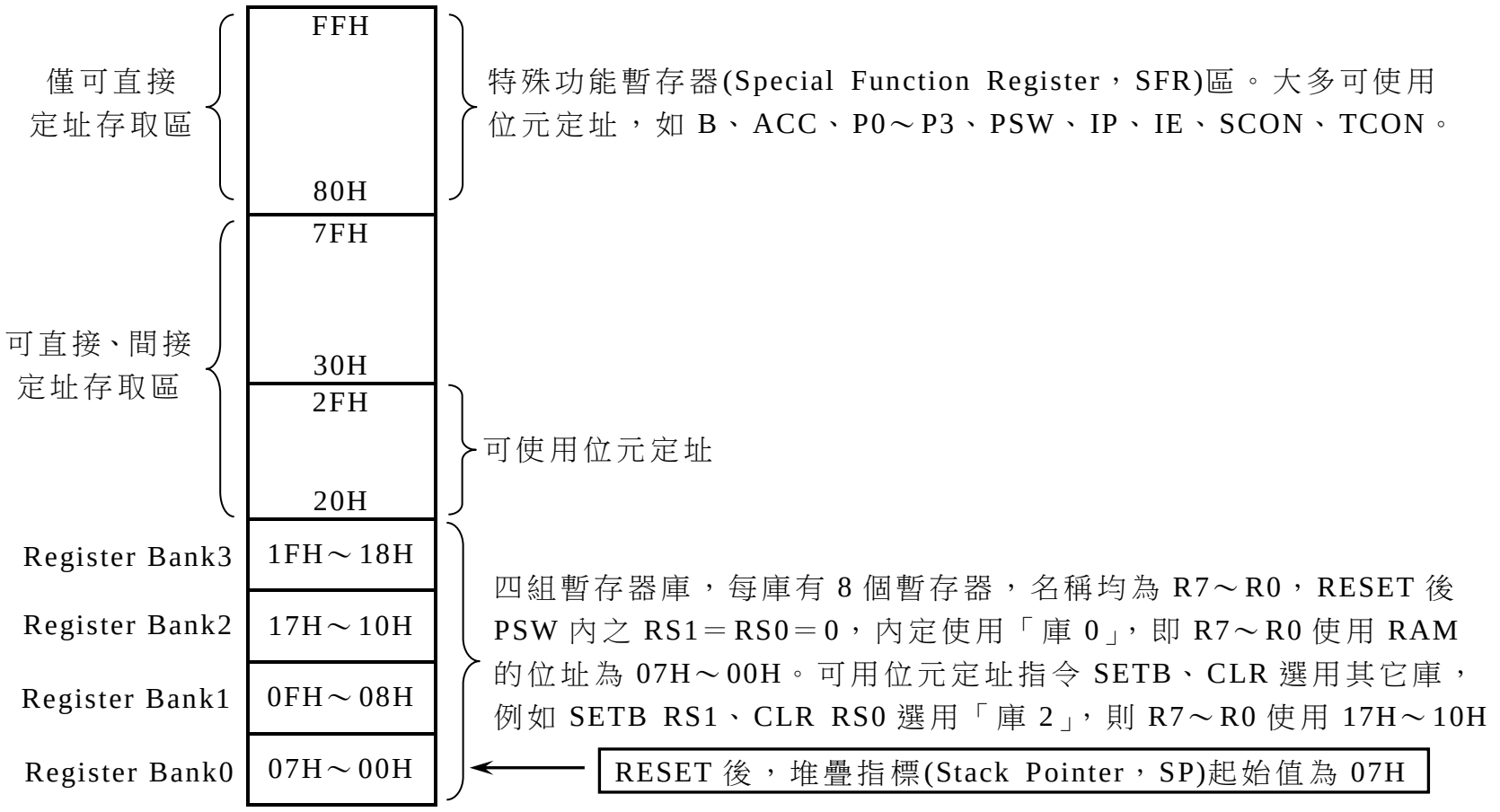
6. 第 18、19 腳 ----工作時脈振盪電路輸入腳。

7. 第 9 腳----------重置(RESET)輸入腳，當加入高電位時，將從位址 000 處開始執行程式，而 Port 0～Port 3 均輸出「High」。

8. 第 30 腳 ---------- 當 8x51 欲至外部記憶體存取指令或資料時，此腳輸出 1 時，表示 Port 0 輸出的是位址信號。

9. 第 29 腳 ---------- 當此腳輸出 1 時，表示 8x51 至外部程式記憶體讀取指令。

一、資料記憶體(RAM)結構圖(共 256Bytes，使用者可用 128Bytes)



1. 直接定址存取方式

```
MOV   A,#66H
MOV   90H,A    ;將資料 66H 存入 90H 處
```

```
Hour   EQU   20H    ;以虛指令將位址 20H 命名為 Hour
MOV    A,Hour       ;將位址 20H 中的資料搬至暫存器 A
        【也可僅以 MOV   A,20H 完成之】
```

2. 間接定址存取方式(必須透過暫存器 R0 或 R1 執行)

```
MOV   R0,#30H   ;範圍為 00H～7FH
MOV   A,@R0     ;將 30H 中的資料搬至 A
```

```
Hour   EQU   30H    ;以虛指令將位址 30H 命名為 Hour
MOV    R0,#Hour     ;將 30H 置入 R0，注意不可省略「#」號，否則會將 30H 中的資料搬至 R0。
MOV    A,@R0
```

3. 位元定址法(指令：SETB、CLR)

```
SETB  20H.3     ;將位址 20H 中的 Bit3 設定為 1
CLR   28H.7     ;將位址 28H 中的 Bit7 清除為 0
```

```
Key    REG   20H.0   ;以虛指令將位元位址 20H.0 命名為 Key
CLR    Key          ;將位址 20H 中的 Bit0 清除為 0
```

二、程式記憶體(Flash Memory)使用說明

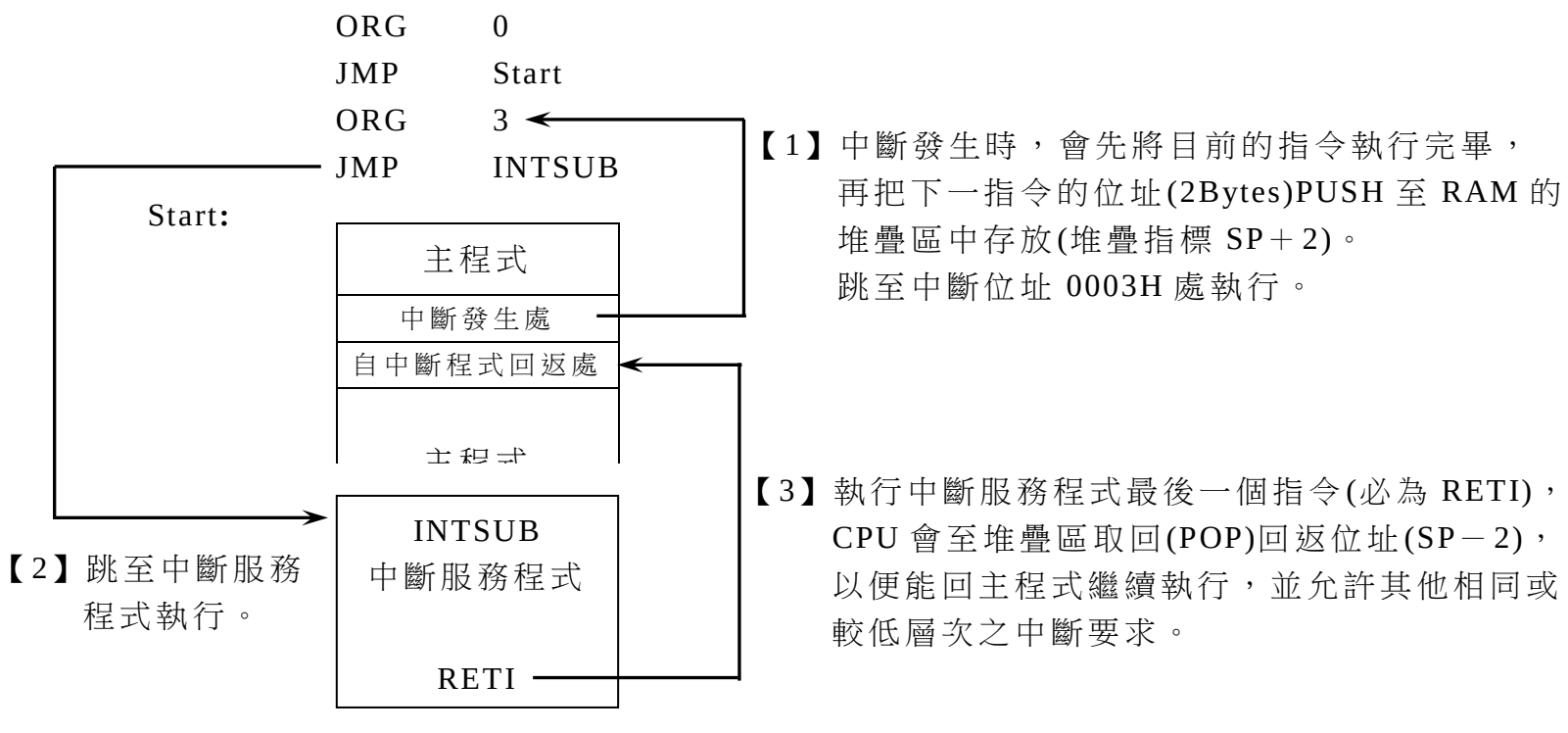
1. 共 4Kbytes 空間(0000H～0FFFH)

2. CPU 被 RESET 後，其程式計數器(Program Counter，PC)之值為 0000H，亦即從此處開始執行程式。

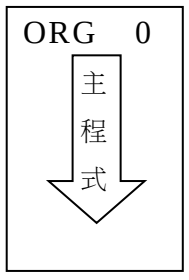
3. 中斷向量：當中斷發生時，程式會跳至相對應的中斷向量位址處執行

中 斷 類 別	向量位址	位址空間
RESET	0000H	3Bytes
外部中斷 0(INT0)	0003H	8Bytes
計時計數 0 中斷(T0)	000BH	8Bytes
外部中斷 1(INT1)	0013H	8Bytes
計時計數 1 中斷(T1)	001BH	8Bytes
串列埠中斷	0023H	

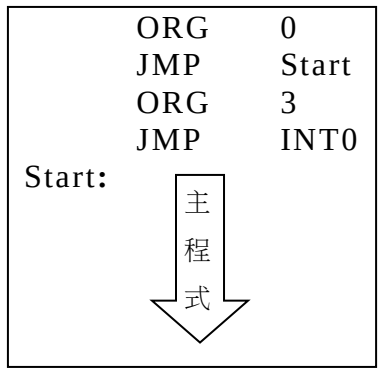
4. 中斷程式執行流程



【範例 1：未使用任何中斷】



【範例 2：使用外部中斷 0】



【範例 3：使用外部中斷 0、1】

```
        ORG     0
        JMP     Start
        ORG     3
        PUSH    A
        PUSH    PSW
        JMP     INT0
        ORG     0BH
        JMP     T0
Start:
```

在範例 3 中，因兩個中斷向量 INT0、T0 之間有 8Bytes 的空間，為避免浪費，可放置一些原本應置於中斷服務程式開頭，用於保存資料的指令，如本範例的 PUSH 指令。

三、程式狀態字語(Program State Word，PSW)，一般 CPU 稱為旗號暫存器(Flag Register)

PSW.7	PSW.6	PSW.5	PSW.4	PSW.3	PSW.2	PSW.1	PSW.0
CY	AC		RS1	RS0	OV		P

1. 進位(Carry)旗號：

當 CPU 執行加法運算時，若 Bit7 有進位則「CY＝1」，否則「CY＝0」。
當 CPU 執行減法運算時，若 Bit7 有借位則「CY＝1」，否則「CY＝0」。
相關指令有：SETB　CY----------設定進位旗號為 1，機械碼長度為 2Bytes。
　　SETB　C----------同上，但機械碼長度為 1Bytes。
　　CLR　C----------清除進位旗號為 0。
　　JB　CY,Next----------檢查 Carry 若為 1 則跳至標記 Next 處執行，機械碼 3Bytes。
　　JC　Next----------同上，但機械碼長度為 2Bytes。
　　JB　PSW.7,Next --------同上，機械碼長度為 3Bytes。
　　JNC　Next----------檢查 Carry 若為 0 則跳至標記 Next 處執行，機械碼 2Bytes。
　　JNB　PSW.7,Next --------同上，機械碼長度為 3Bytes。

2. 輔助進位(Aided Carry)旗號：

又稱「半進位」旗號，當 CPU 執行加法運算時，若 Bit3 有進位則「AC＝1」，否則「AC＝0」。
當 CPU 執行減法運算時，若 Bit3 有借位則「AC＝1」，否則「AC＝0」。
通常 CPU 在執行「DAA」指令時，會依據 CY、AC 旗號進行調整(待敘)。

3. RS(Register Select)旗號：配合 SETB 及 CLR 指令選擇暫存器庫。

4. 溢位(Overflow)旗號：

當 CPU 執行算術運算時，若 8Bits 之結果超出＋127～－128 時，「OV＝1」，否則「OV＝0」。
相關指令有：JB　OV,Next----------檢查 OV 若為 1 則跳至標記 Next 處執行，機械碼 3Bytes。
　　JB　PSW.2,Next --------同上，機械碼長度為 3Bytes。
　　JNB　OV,Next----------檢查 OV 若為 0 則跳至標記 Next 處執行，機械碼 3Bytes。
　　JNB　PSW.2,Next --------同上，機械碼長度為 3Bytes。

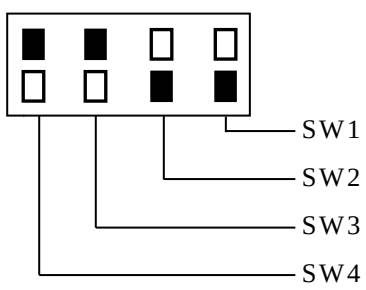
5. 同位元(Parity)旗號：

若累積器 ACC 中有奇數個 1 時，如「00110001」則旗號 P＝1。
若累積器 ACC 中有偶數個 1 時，如「00010001」則旗號 P＝0。

單元 3：CPU 線上模擬器(In-Circuit Emulator，ICE)簡介

一、使用機型：PICE 8051/52，以下簡稱「PICE－52」。

二、PICE－52 側面之指撥開關(DIP SW)設定：



SW1(上)：P3.7 為 RD 控制線
(下)：P3.7 為 IO 埠

SW2(上)： 使用 89C51 僅能設定為上

SW3、SW4： 二者均為上時，使用 PICE－52 內部之石英晶體(11MHz)
二者均為下時，使用目標板上之石英晶體【正常使用】

注意：若 CPU 的 P3.7 作為 RD 控制線時，須另在 PICE 模式中下達【MAP ALL X：UM】指令

三、程式檔案功能

1. LUIT.EXE..... PICE－52 系統驅動程式。
2. HUIT.HLP ..... LUIT 輔助說明檔。
3. P52.BAT ..... 啟動 PICE 之批次檔，內容為【LUIT/P1/B+】，
   P1 表示使用 COM1，
   P2 表示使用 COM2
   B+表示傳輸速率為 19200Baud Rate
4. INF.BAT(自編) ..... PICE－52 啟動後以 Execute 命令執行的批次檔

```
winopen register                (開啟暫存器狀態之視窗)
winlocate register 2 68 21 10   (設定暫存器視窗位置)
winopen program                 (開啟原始程式視窗)
winlocate program 6 35 10 31    (設定程式視窗位置)
winopen data1                   (開啟資料一之視窗)
winlocate data1 18 14           (設定資料一視窗位置)
mdump data1 d:00h               (顯示 89C51 中 RAM 的內容，自位址 00H 開始)
download io.hex                 (下載 User 之檔案)
```

四、數字輸入格式

1. 二進數：00001010B
2. 十進數：10D 或 10
3. 十六進數：10H、0A6H (以英文字母開頭的十六進數，須在前面加「零」

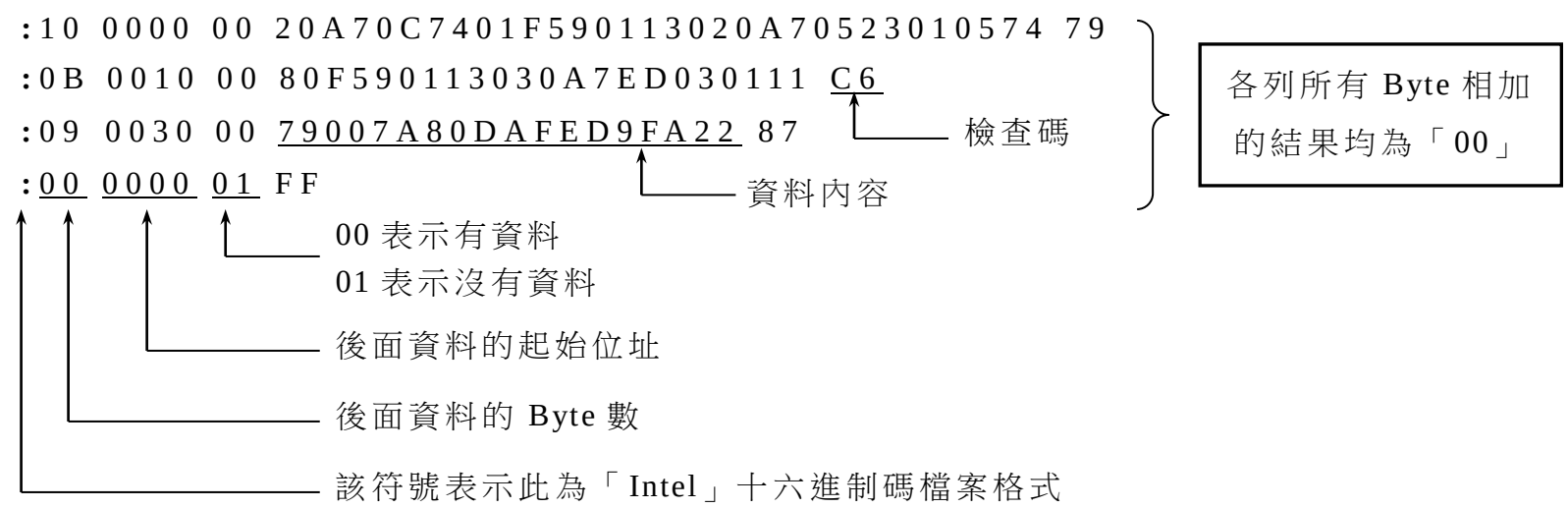
五、常用按鍵功能

1. 【↑】、【↓】.....重複叫出先前執行過的命令。
2. 【Tab】.....選擇已開啟的視窗，被選到的視窗將變成雙線外框。
3. 【Esc】.....取消下達的命令。
4. 【F4】.....游標在主視窗時，可打開或關閉已開啟的視窗。游標在某視窗時，可將之放大或還原。
5. 【F5】.....從目前的 PC 值(一般為 000H)開始執行程式。
6. 【F6】.....停止執行程式。
7. 【F8】.....依目前的 PC 值，單步執行一個指令。
8. 【F9】.....設定中斷點。游標位於「程式視窗」中所在的指令位址被設為中斷點。若該位址已被設為中斷點，則可取消之。
9. 【F10】.....單步執行副程式(例如「CALL」指令)，會進入該副程式全部執行完畢後返回。

六、常用命令

1. CLS.....清除主視窗。
2. DOWNLOAD .....由磁碟下載 HEX 檔案，如「DOWNLOAD TEST.HEX」。
3. RESET.....重置，設定 PC 值為 000H。
4. SETBP.....設立程式中斷點，如「SETBP 400H」，則執行至該處即停止，可視需要設立多處。
5. CLRBP .....清除中斷點，如「CLRBP 400H」、「CLRBP ALL」。
6. INPORT 0～3.....讀取某 Port 的狀態並顯示於螢幕。
7. OUTPRT .....輸出一數值至某 Port，如：OUTPRT 1 56H (輸出 56H 至 Port1)
   OUTPRT 3 0A6H (輸出 A6H 至 Port3)
   OUTPRT 2 BIT 3 0 (輸出 0 至 Port2 的 Bit3)
8. ASSEMBLE .....線上編修組合語言指令，如「ASSEMBLE 100H」，注意新指令碼長度須等於原指令。

七、「HEX」檔的內容



單元 4：控制暫存器與中斷

一、中斷致能暫存器(Interrupt Enable，IE)可位元定址

IE.7	IE.6	IE.5	IE.4	IE.3	IE.2	IE.1	IE.0
EA	×	×	ES	ET1	EX1	ET0	EX0
				T1	INT1	T0	INT0

1. 位元定址法：SETB EA ------ 所有中斷可致能(EA 表示 Enable All)
   CLR EA ------ 所有中斷除能，不論其他中斷位元如何設定，一切中斷均不接受。
   CLR IE.7----- 同上
   SETB ES------- 串列傳輸中斷致能
   CLR ES------- 串列傳輸中斷除能
   SETB ET1 ----- 計時/計數器 1 中斷致能
   CLR EX1----- 外部中斷 INT1 除能
2. 位元組定址法：MOV IE,#10000011B----------計時/計數器 0 及外部中斷 INT0 致能

二、中斷優先權暫存器(Interrupt Priority，IP）可位元定址

IP.7	IP.6	IP.5	IP.4	IP.3	IP.2	IP.1	IP.0
×	×	×	PS	PT1	PX1	PT0	PX0
				T1	INT1	T0	INT0

內定均為「0」，屬相同的低層次優先權，若設定為「1」則為高層次優先權。

雖內定均為相同低層次，但仍有優先輪詢順序：INT0→T0→INT1→T1→串列傳輸

三、外部中斷注意事項

1.低準位觸發時，在中斷服務程式結束前，該中斷要求低準位信號必須消失，否則會一直產生中斷。

2.負緣位觸發時，中斷旗號 IE0、IE1 具栓鎖功能，直到執行中斷服務程式時才會清除為 0，若負緣信號出現二次以上，會被栓鎖二次而執行中斷服務程式二次，產生誤動作，解決方法為：

a.中斷信號加消除彈跳電路。

b.在中斷服務程式結束前，加一段延遲程式(至少 20msec)，再以 CLR IE0(或 IE1)指令將中斷旗號以軟體方式清除之，此種方式必須考慮延遲程式不會影響整體程式的功能。若中斷服務程式執行時間超過 20msec，則不必另加延遲程式。

四、Timer 與 Counter 有何異同？

相同點：使用同一電路，且屬於上數型計數器(Up Counter)。

相異點：觸發時脈來源，Timer 的觸發時脈間隔固定，通常來自系統脈波，Counter 之觸發時脈間隔不定，且來自外部，即 P3.4(T0)及 P3.5(T1)輸入腳。

8051 屬於負緣觸發型計數器，與外部中斷不同的是不具栓鎖功能

五、計數溢位：例如從 40H→41H→•••→FFH→00H 時稱之，並設定中斷旗號 TF0 或 TF0 為 1。

六、計時計數控制暫存器(Timer Control，TCON)可位元定址

TCON.7	TCON.6	TCON.5	TCON.4	TCON.3	TCON.2	TCON.1	TCON.0
TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0

TF1：當 Timer1 計數溢位時，此位元將被設定為「1」，而在執行中斷服務程式時自動清除為 0。

TR1：為「1」時開啟 Timer1 計數動作，「0」時關閉之。此動作由軟體指令設定(SETB TR1)。

TF0 及 TR0 用於 Timer0，作用同 TF1、TR1。

IT0＝0，外部中斷 INT0 為低準位「Low」觸發。

IT0＝1，外部中斷 INT0 為負緣觸發。

IE0：若 INT0 為負緣觸發，產生中斷後，將被 CPU 設定為 1，而在執行中斷服務程式時自動清除為 0。

IT1 及 IE1 用於 INT1，作用同 INT0。

七、計時計數模式暫存器(Timer Mode，TMOD)，不可位元定址

←Timer1→				←Timer0→			
GATE	C/T	M1	M0	GATE	C/T	M1	M0

GATE＝0：在設定 TCON 的 TRx 為「1」後開始計時/計數。

＝1：除設定 TCON 的 TRx 為「1」外，尚需待 INTx 為「1」後才開始。

C/$\overline{T}$＝0：設定為計時模式，計時單位為系統時脈頻率的 1/12。

＝1：設定為計數模式。

M1、M0：工作模式選擇

M1	M0	MODE	功 能 說 明
0	0	0	13Bits 計時計數器
0	1	1	16Bits 計時計數器
1	0	2	8Bits 自動載入型計時計數器
1	1	3	【略】

八、上數型計時計數器之暫存器 THx、TLx 起始值設定法

1. MODE0：由 8 位元的高位元組 THx，5 位元的低位元組 TLx 計數器串接組成 2^{13}＝8192 模計數器。
例如要計數 100 個脈波，其起始值應為 8192－100＝8092，將 8092 除以 32 的商置於 THx，而餘數(=31)置於 TLx。

Macro Assembler 及 Ajon 的指令寫法：
```
MOV   TL0,#(8192－1).MOD.32
MOV   TH0,#(8192－1)/32
```

Keil UV2 的指令寫法：
```
MOV   TL0,#(8192－1) MOD 32
MOV   TH0,#(8192－1)/32
```

2. MODE1：由 8 位元的高位元組 THx，8 位元的低位元組 TLx 計數器組成 2^{16}＝65536 模計數器。
例如要計數 2 個脈波，其起始值應為 65536－2＝65534＝FFFEH，將 FFH 置於 THx，而 FEH 置於 TLx。

Macro Assembler 及 Ajon 的指令寫法：
```
MOV   TL0,#＜(65536－2)
MOV   TH0,#＞(65536－2)
```

Keil UV2 的指令寫法：
```
MOV   TL0,#(65536－50000) MOD 256
MOV   TH0,#(65536－5000)/256
```

3. MODE2：為自動重新載入之 8Bits 計數計時器，計數本體由 TLx 擔任，當其發生計數溢位時，暫存器 THx 之值將重新載入 TLx。

8051 常用指令介紹　(設工作頻率為 12MHz，一個機械週期費時 12 個 clock，即 1us)

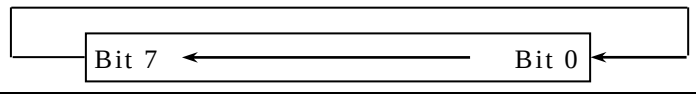
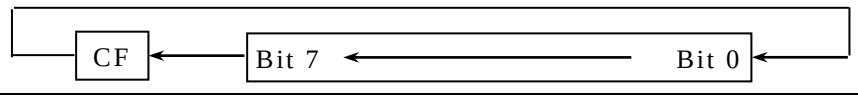
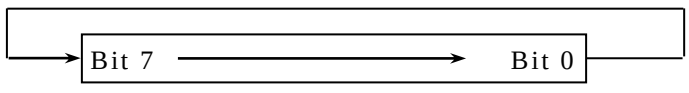
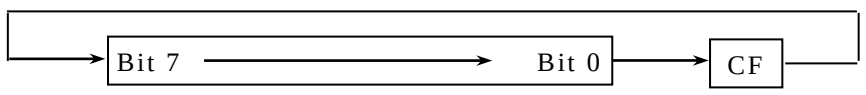
◆定址模式：指令運算元(Operand)取得資料的方式謂之

1. 立即定址法 ---------- 常數，如#00100110B、#26H、#32D(或#32)
2. 直接定址法 ---------- 資料取自記憶體，如 89C51 的 RAM(00H～FFH)，須注意兩點，一為位址 80H～FFH 被 CPU 挪用為特殊暫存器，例如 A、B、P3～P0 等，此外 00H～1FH 是屬於一般暫存器及內定堆疊區範圍，因此我們可任意存取資料的位址為 20H～7FH。
3. 暫存器定址法 --------- 資料取自暫存器，如 R7～R0。
4. 暫存器間接定址法 --- 暫存器(僅能用@R1、@R0)的內容代表記憶體位址，至該處取資料。
5. 位元定址法 ---------- 設定或清除進位旗號，或其他可使用位元定址的 RAM 區域。
6. 隱含定址法 ---------- 指令中沒有運算元，例如「NOP」，CPU 執行時須一個機械週期，但什麼事也沒做，其機械碼為 00H。

◆常用指令：指令中的符號 ------「n」代表 0～7　　「i」代表 0 或 1
「data」代表 00H～FFH　　「direct」代表記憶體位址

搬移(複製)							
指令		動作說明	機械週期	指令		動作說明	機械週期
MOV	A,Rn		1	MOV	direct,@Ri		2
MOV	A,direct		1	MOV	direct,#data		2
MOV	A,@Ri		1	MOV	@Ri,A		1
MOV	A,#data		1	MOV	@Ri,direct		2
MOV	Rn,A		1	MOV	@Ri,#data		1
MOV	Rn,direct		2	PUSH	direct		2
MOV	Rn,#data		1	POP	direct		2
MOV	direct,A		1	XCH	A,Rn	Exchange	1
MOV	direct,Rn		2	XCH	A,direct		1
MOV	direct,direct		2	SWAP	A	A 之高低 4 位元互換	1

算術運算							
指令		動作說明	機械週期	指令		動作說明	機械週期
ADD	A,Rn	A←A+Rn	1	SUBB	A,@Ri		1
ADD	A,direct		1	SUBB	A,#data		1
ADD	A,@Ri		1	INC	A		1
ADD	A,#data		1	INC	Rn		1
ADDC	A,Rn		1	INC	direct	Increment (加 1)	1
ADDC	A,@Ri		1	INC	@Ri		1
ADDC	A,direct	另加進位旗號 CF	1	DEC	A		1
ADDC	A,#data		1	DEC	Rn		1
SUBB	A,Rn		1	DEC	direct	Decrement (減 1)	1
SUBB	A,direct	Subtract with Borrow	1	DEC	@Ri		1

旋轉			
指令		動作說明	機械週期
RL	A	Rotate Left 	1
RLC	A	(with Carry) 	1
RR	A	Rotate Right 	1
RRC	A	(with Carry) 	1

邏輯運算							
指令		動作說明	機械週期	指令		動作說明	機械週期
CLR	C	Clear Carry Flag	1	ORL	A,Rn	OR 運算	1
SET	C	Set Carry Flag	1	ORL	A,@Ri		1
CLR	bit	清除/設定記憶體某位元 如 CLR 20H.2	1	ORL	A,direct		1
SET	bit		1	ORL	A,#data		1
CLR	A	A＝0	1	ORL	direct, A		1
CPL	A	A＝/A (Complement)	1	ORL	direct,#data		1
CPL	bit		1	XRL	A,Rn	Exclusive OR 運算	1
ANL	A,Rn	AND 運算	1	XRL	A,@Ri		1
ANL	A,@Ri		1	XRL	A,direct		1
ANL	A,direct		1	XRL	A,#data		1
ANL	A,#data		1	XRL	direct, A		1
ANL	direct, A		1	XRL	direct,#data		1
ANL	direct,#data		1				

相對位址(距離)跳躍分支						
指令	動作說明	機械週期	指令		動作說明	機械週期
DJNZ Rn,Label	Decrement and Jump to Label if Not Zero	2	JC	Label	進位旗號 CF=1 則跳(MSB 有進位)	2
DJNZ direct,Label		2	JNC	Label	進位旗號 CF=0 則跳	2
CJNE A,direct,Label	Compare and Jump to Label if Not Zero	2	JZ	Label	進位旗號 ZF=1 則跳(運算結果 0)	2
CJNE A,#data,Label		2	JNZ	Label	進位旗號 ZF=0 則跳(結果≠0)	2
CJNE Rn,#data,Label		2	JB	bit,Label	被測試的某位元為 1 則跳， 如 JB P1.0,Label	
CJNE @Ri,#data,Label		2				
JNB bit,Label	被測試的某位元為 0 則跳，如 JB ACC.5,Label	2	JBC	bit,Label	同上，但該位元會被清除為 0	2

一、絕對位址跳躍：

1. 指令 LCALL(或 CALL)：呼叫 0000H～FFFFH 任一位址處之副程式執行，機械碼為 3Bytes。
2. 指令 ACALL：【位置範圍與 AJMP 相同，但 AJMP 無堆疊動作】，機械碼為 2Bytes。
3. 指令 LJMP(或 JMP)：無條件跳至 0000H～FFFFH 任一位址處執行，機械碼為 3Bytes。
4. 指令 AJMP：無條件跳至 0000H～FFFFH 任一位址處執行，機械碼為 2Bytes。
   與 LJMP 不同的是，以每 2K 為一頁(不得跨頁跳躍)，如 0000H～07FFH　0800H～0FFFH
   1000H～17FFH　1800H～1FFFH
   2000H～27FFH　2800H～2FFFH

二、相對位址(距離)跳躍：JC、JNC、JB、JNB、JBC、JZ、JNZ、CJNE、DJNZ、SJMP

跳躍範圍：7FH～80H，即＋127D(往下跳)～－128D(往上跳)

延遲副程式解析(設 CPU 工作頻率為 12MHz，則一個機械週期＝1us)

若外加 12MHz 的頻率，則其時脈週期 T＝$\frac{1}{12M}$ 秒，而 8x51 執行一個機械週期(Machine Cycle)需 12 個 T 週期(T Cycle)，即 1uS，CPU 完成一個指令至少需一個機械週期，稱之為指令週期。

一、Delay500us：

```
        MOV   R7,#248 ---------- 1 Machine Cycle ×1 次   =   1 Machine Cycle
        DJNZ  R7,Delay500us ---- 2 Machine Cycle ×248 次 = 496 Machine Cycle
        NOP   ---------------------- 1 Machine Cycle ×1 次   =   1 Machine Cycle
        RET   ---------------------- 2 Machine Cycle ×1 次   =   2 Machine Cycle
```

總機械週期＝500

二、Delay10ms：

```
        MOV   R6,#34 ------------- 1 ×1 次              =     1 Machine Cycle
DLY：
        MOV   R7,#145 ------------ 1 ×34 次             =    34 Machine Cycle
        DJNZ  R7,$---------------- 2 ×145 次 ×34 次 = 9860 Machine Cycle
        NOP   ---------------------- 1 ×34 次             =    34 Machine Cycle
        DJNZ  R6,#DLY ------------ 2 ×34 次             =    68 Machine Cycle
        NOP   ---------------------- 1 ×1 次              =     1 Machine Cycle
        RET   ---------------------- 2 ×1 次              =     2 Machine Cycle
```

總機械週期＝10000

三、Delay1s：

```
        MOV   R5,#40 ------------- 1 ×1 次                =     1 Machine Cycle
DLY1：
        MOV   R6,#70 ------------- 1 ×40 次               =    40 Machine Cycle
DLY2：
        MOV   R7,#177 ------------ 1 ×70 次 ×40 次        =  2800 Machine Cycle
        DJNZ  R7,$---------------- 2 ×177 次 ×70 次 ×40 次 = 991200 Machine Cycle
        DJNZ  R6,DLY2 ------------ 2 ×70 次 ×40 次        =  5600 Machine Cycle
        DJNZ  R5,DLY1 ------------ 2 ×40 次               =    80 Machine Cycle
        MOV   R5,#138 ------------ 1 ×1 次   =   1 Machine Cycle
        DJNZ  R5,$---------------- 2 ×138 次 = 276 Machine Cycle
        RET   ---------------------- 2 ×1 次   =   2 Machine Cycle
```

至此機械週期為 999721，不足 279 個再加一迴圈。

總機械週期＝1000000