

算數運算（加法二）

上一篇文章談到了 8 bits 的加法運算，現在如果改成是無正負號（Unsigned）的 32 bits 的相加呢？底下是標準的範例程式。

```

DATA_ADD_4BYTE
    MOV    R2,#04H
    CLR    C
$1  MOV    A,@R0
    ADDC   A,@R1
    MOV    @R0,A
    INC    R0
    INC    R1
    DJNZ   R2,$1
    RET
  
```

首先假設 R0=10H、R1=14H，那麼這個範例便是將 13H、12H、11H、10H 這些位址的值與 17H、16H、15H、14H 這些位址的值相加，再假設在 13H 10H 的值分別為 33H、FFH、01H、00H，而在 17H 14H 的值分別為 03H、03H、01H、00H，有了這些預設值之後，我們利用表格來實際推導一下這個範例的運算方法。此時的 R2 是用來做為迴圈數的控制。

次數	C	R1				R0			
		(17H)	(16H)	(15H)	(14H)	(13H)	(12H)	(11H)	(10H)
1. R2=04H R1=14H R0=10H	0	03H	03H	01H	00H	33H	FFH	01H	00H
2. R2=03H R1=15H R0=11H	0	03H	03H	01H	00H	33H	FFH	02H	00H
3. R2=02H R1=16H R0=12H	1	03H	03H	01H	00H	33H	02H	02H	00H
4. R2=01H R1=17H R0=13H	0	03H	03H	01H	00H	37H	02H	02H	00H

表格中標示紅色與藍色的部分，是每次運算時更動到的部分，當系統進行最低 Byte 的加法運算時，先將 (10H) 中的 00H 搬到 Acc 累加器中，再把 (14H) 中的 00H 與 Acc 累加器及 CY 旗標值相加，此時的 CY 值為啟始的預設值 0，因此相加後所得到的值 00H + 00H + 0 = 00H 會再存回 Acc 累加器中，最後把 Acc 累加器的值再搬到 (10H) 的位址裡。此時的加法因為沒有任何的溢位狀況，所以 CY 旗標值還是 0。

第二個 Byte 的相加同第一個 Byte 一樣，由於相加的結果最後是要存回 (11H) 裡，所以 (11H) 的值便成了 02H。

第三個 Byte 的相加有些不同， $FFH + 03H + 0 = 02H$ 產生了溢位，因此 CY 旗標會設定成 1，但這裡的 1 並不會加在這一次的運算裡，因為 ADDC 運算時是利用前一次運算所得到的 CY 值進行運算的。此時的 (12H) 會被填為 02H。

第四個 Byte 的相加因為沒有溢位，所以 CY 值會被再設定成 0，然而在運算加法時會將前一次的 CY 值加進來，所以 (13H) 最後所得到的值是 $33H + 03H + 1 = 37H$ 。

在這個常式範例裡，有幾個必須要特別注意的要點：第一、R0、R1 及 R2 為運算位址的暫存區及迴圈數的設定值，必須避開運算的位址用到這三個暫存器。第二、R0 與 R1 內所設定的四個位址不可有重疊的情形，否則會出現很嚴重的運算錯誤。第三、如果相加的數值不是 32 bits，而是 24 bits 或是 48 bits，只需將 R2 的值改成 3 或 6，並確定 R0 及 R1 所設定的位址範圍沒有重疊即可。

問題來了，如果現在有三個 Unsigned 32 bits 的數值要相加，應該要怎麼做呢？聰明的你，動動腦想看看吧！