認識 HEX 檔

看過幾篇常用指令的用法後，我們換換口味，介紹一下 intel 原廠所公佈的 HEX 檔標準格式，相信經過本文的介紹，一定可以讓您對 8051 的運作有更進一步的認識。底下是一個程式經編譯器編譯後所得到的 HEX 檔內容：

Example.hex

行號	原始碼
1	:10000000020003787FE4F6D8FD75812B02004A02D6
2	:10001000008FE493A3F8E493A34003F68001F20871
3	:10002000DFF48029E493A3F85407240CC8C333C435
4	:10003000540F4420C8834004F456800146F6DFE4A0
5	:10004000800B01020408102040809000C2E47E0171
6	:100050009360BCA3FF543F30E509541FFEE493A313
7	:1000600060010ECF54C025E060A840B8E493A3FAF7
8	:10007000E493A3F8E493A3C8C582C8CAC583CAF0B1
9	:10008000A3C8C582C8CAC583CADFE9DEE780BEE432
10	:10009000F52AF52BE52B25E02408F8E6F52808E6F7
11	:1000A000F529F590E528F5B07F707E171200E6057A
12	:1000B0002BE52B7002052AC39410E52A940040D446
13	:1000C00080CD2020080001000200040008001007C
14	:1000D000200040008001000200040008001002001
15	:1000E000004000800000D3EF9400EE94004007EF42
16	:0700F0001F70F31E80F022D7
17	:00000001FF

面對這一大串的 16 進位碼，有沒有頭昏眼花的感覺呢？別急別急，經過本文的介紹，你一定會愛上這個會讓你頭昏眼花的機械碼的！

首先我們先介紹 HEX 檔的編碼格式，舉範例程式中第一列說明：

:	10	0000	00	020003787FE4F6D8FD75812B02004A02	D6
1	2	3	4	5	6

為了方便解說，筆者將原始碼以空格區分成六個部分，在實際轉換的原始內容應該沒有空格也沒有行號的。第一個部分是 HEX 檔的啟始格式，檔案一開始應該是一個冒號做為起點；第二部分的兩碼(10H)所代表的是該列總共具備多少個 Bytes 的資料，以本列為例，應該有 16 個 Bytes，因為 10H 換算成十進位應該是 16；第三部分的四碼所代表的是放置資料

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

的啟始位置，換句話說，本列的資料應該是放置在 0000H　000FH 這段位址中，第四部分的兩碼是檢查碼，所代表的是該列有無資料存放，若有則為 00H，若無則為 01H；第五部分則為資料存放區，總共 32 碼 16 個 Bytes，有興趣的可以實際算看看，這裡的機械碼已經是標準的 MCS-51 指令，如果查表的話可以直接知道其功能，稍後的文章會再提到；第六部分為 Checksum 值。

以上每一個資料列的標準格式，再看到最後一列，這是每一個 HEX 檔最後都會加上的一列，他所代表的意思也等於我們寫程式的 END 差不多。當組譯器看到這一段後，就不會再繼續組譯下去。

問題來了，什麼叫做 Checksum 值？它的功用是什麼？很多人可能都有這樣的疑問，所謂的 Checksum 值是一種標準的檢查碼，把它加在每一列機械碼的最後，可以使每一列所有的 16 進位值(兩個為一組)，加總後所得到最後兩位 16 進位碼應為 00H，如果你不太會 16 進位運算的話，沒關係，Windows 中有一個很好用的工具叫『小算盤』，你可以切換到工程計算的 16 進位模式將數值直接輸入，他就會自動幫你算好了。以本列來說，將所有的值加總所得到的值應該是：

10H + 00H + 00H + 00H + 02H + 00H + 03H + 78H + 7FH + E4H + F6H + D8H
+ FDH + 75H + 81H + 2BH + 02H + 00H + 4AH + 02H + D6H = 700H

最後兩個碼果然為 00H ，有興趣的人可以把範例中的每一行都算算看，結果應該都是這樣的。如果不是呢？那你的編譯器一定是壞了，換一套安裝吧！因為如果 Checksum 值不對的話，在連結的時候一定會出現 ERROR，該檔案根本就無法使用，到此為止，對 Checksum 值的重要性有更進一步的瞭解了嗎？

剛剛我們提到可以利用 HEX 檔來看看原始程式是怎麼寫的，如果你手邊沒有紙筆跟 MCS-51 指令集的話，趕快去準備一分吧，接下來才是本文的精華所在唷！

同樣以第一列為例，我們要看的地方是起始位址與資料區的部分，也就是第二與第五部分，一開始先將起始的位置寫上，並寫好第一組數值（如下所示）

0000H　　02

經查表得知 02H 為 LJMP 的指令，佔用三個 Bytes 的空間，因此接下來的兩個 Bytes 應該是其存放的位址資料，所以我們便將這兩組數值填在 02H 的後面（如下所示）

0000H　　02　00　03

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

接下來把所查到的指令詳細地寫在右邊（如下所示），如此你便得到第一行程式了。

```
0000H    02  00  03          LJMP     0003H
```

接下來呢？重覆剛剛的順序，先把起始的位址寫上，這時應該用掉三個 Bytes，所以啟始的位址便移到了 0003H，同樣把 0003H 寫上，再填上第一個數值（如下所示）

```
0000H    02  00  03          LJMP     0003H
0003H    78
```

再經由查表，78H 所代表的是 MOV R0,#data，佔用空間為兩個 Bytes，因此再將下一個 Byte 的數值填上，並將該指令寫在右邊，如此便得到第二行程式（如下所示）

```
0000H    02  00  03          LJMP     0003H
0003H    78  7F              MOV      R0,#7FH
```

開始有趣了，對不對？原來程式還可以這樣子看，如此一來只要能夠拿到 HEX 檔，就等於知道了全部的原始程式了，不是嗎？沒錯！大致上來說的確是如此的，而一般的反組譯程式也是利用這樣的原理所寫出來的，在旗威科技交流網中所提供的 dis51 反組譯程式就是最好的例子。不過這其中還有很多的竅門，必須經過很多的嘗試及多看別人的程式才能體會，本文就先介紹到此，陸陸續續我們會為您解開更多 8051 的神秘面紗。

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