CHAPTER 02 數位資料表示法

到底數位是什麼呢？

- 數位在電學上是指不連續變化的數量表示法。
- 何謂不連續變化？
  - 實數是連續變化的數量表示法，因為任兩數之間還可以找到第三個數介於它們之間，而且到最後是沒有空隙的。
  - 整數是不連續變化的數量表示法，例如整數1和整數2之間，我們再也找不到任何整數是介於它們之間的。

到底數位是什麼呢？

- 針對不連續變化的數量，可以用位元(binary digit；bit)的組合來計數。
- 位元是數位資訊的基本粒子，也是電腦儲存或傳遞資料的最小單位，常用0或1來表示。
- 電腦會採用位元表示資料，主要是因為電子元件的穩定狀態有兩種，單一的0或1稱為位元(bit)。
  - 「開」(通常用來表示“1”)
  - 「關」(通常用來表示“0”)

到底數位是什麼呢？

- 早期電腦以8個位元為存取單位，因此8個位元稱為位元組(byte)。
- 兩個位元可以有 2的2次方 共4種組合(00, 01, 10, 11)。
- 每增加一個位元，組合數就加倍。
- n個位元可以有 2^n 種不同的組合，就可用來表示 2^n 種不同物件。

1位元	2位元	3位元	4位元	5位元
0	00	000	0000	00000
1	01	001	0001	00001
	10	010	0010	00010
	11	011	0011	00011
		100	0100	00100
		101	0101	00101
		110	0110	00110
		111	0111	00111
			1000	01000
			1001	01001
			1010	01010
			1011	01011
			1100	01100
			1101	01101
			1110	01110
			1111	01111
				10000
				10001
				10010
				10011
				10100
				10101
				10110
				10111
				11000
				11001
				11010
				11011
				11100
				11101
				11110
				11111

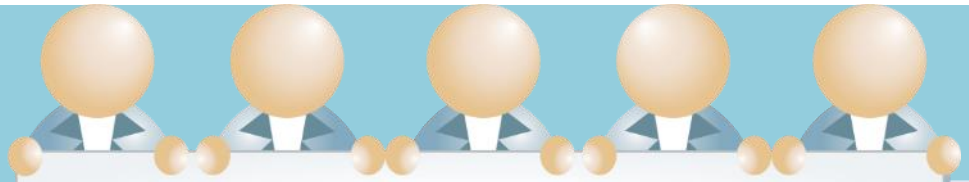
一到五個位元的各種組合

到底數位是什麼呢？

- 8個位元可以有 2的8次方 共256種組合，足以表示每一個英文字母(大小寫共52個)、數字(0到9共10個)和標點符號。ASCII就是這類型組合的公定標準。
- 16個位元可以有 2的16次方 共65,536種組合，遠超過常用的中文字數目，因此16個位元可表示中文字。

到底數位是什麼呢？

- 為避免各國文字的位元表示方式有所衝突，萬國碼(Unicode)依實現方式不同，而以不同位元個數的組合來公定各國文字。
- 由於電腦的存取機制以位元組為基本單位，所以表示資料所需的位元數，通常是8、16及32等。



IT 魔 法 小 百 科

關於資料容量的單位，常見的有KB、MB、GB及TB四種。

「B」代表的是Byte(位元組)，不是Bit(位元)。

「K」代表了2^{10}，為1,024，大約是一千左右。

「M」是$2^{20} = 2^{10} \times 2^{10} = 1{,}048{,}576$，大約是百萬左右。

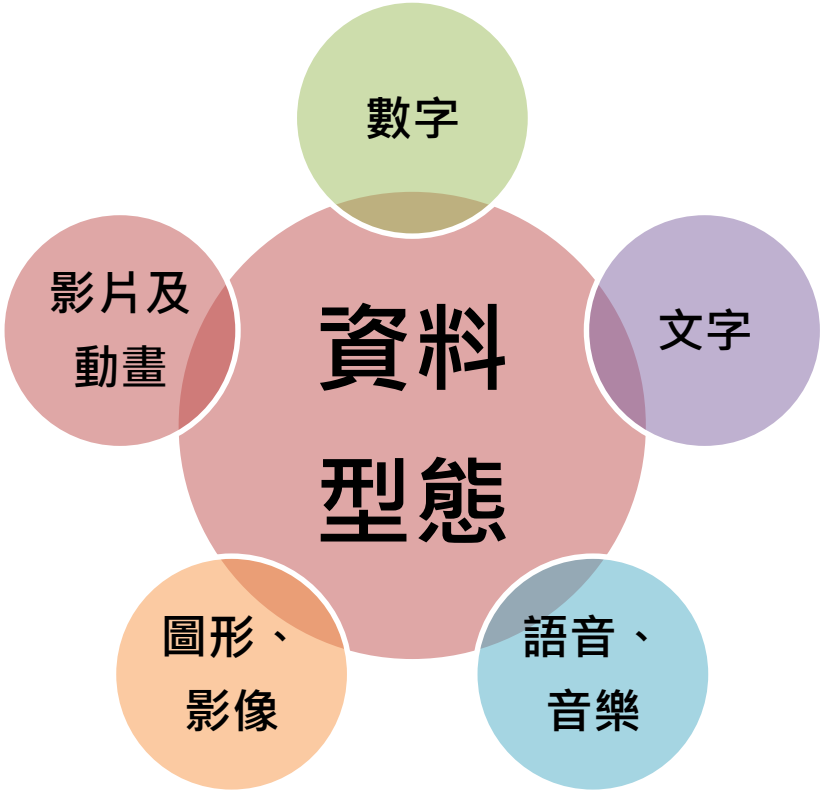
對於2^x的估算，我們常以2^{10}為簡化的捷徑，因為它和10^3(也就是1000)非常接近。

IT 魔 法 小 百 科

縮寫單位	全名	精確位元組個數	大約位元組數	範例
KB	Kilo Byte	$2^{10} = 1,024$	一千(10^3)	這個檔案的大小約238 KB。
MB	Mega Byte	$2^{20} = 1,048,576$	一百萬(10^6)	此大姆碟的容量為512 MB。
GB	Giga Byte	$2^{30} = 1,073,741,824$	十億(10^9)	本片D V D的容量為4.7 GB。
TB	Tera Byte	$2^{40} = 1,099,511,627,776$	一兆(10^{12})	這部高容量磁碟可儲存20 TB的資料。

2-1 資料型態

- 電腦需要處理的**資料型態**(data type)有：數字、文字、語音、音樂、圖形、影像、影片及動畫等，會編碼成位元字串儲存在電腦裡，等到顯示或列印時，再解碼成原來的資料格式。



2-1 資料型態

- 影像數位化
  - 以黑白照片為例，照片的一小部分記錄每個方格的灰度(0~255)，每個方格可用八位元來表示(八個0與1可以有256種組合)。可依同樣道理將彩色圖片數位化。
- 聲音數位化
  - CD唱片上的取樣是每秒約四萬四千次，每一次取樣的聲波，都可轉化成相對應的位元。

2-1 資料型態

- 數位化的資訊方便編輯、處理、儲存、傳輸及播放，以便更有效精確地表達意念。
- 可用電腦編輯及整合不同的數位化資訊，精確安排各種複雜媒體出現的順序、時間及播放設備。
- 可利用電腦強大的處理及搜尋功能，提供多媒體的互動方式，加強虛擬實境的真實感。
- 透過網際網路無遠弗屆的牽引，使數位化資訊即時傳送到世界每一個角落。

2-2 二進位表示法

- 古巴比倫人所用的數字系統是六十進位法，逢「六十」進一，現在除了每分鐘六十秒及每小時六十分外，此法已不多見。
- 現今公制是以十為基數，採用十進位法，滿「十」進一。

2-2 二進位表示法

- 一個數字在不同的位置上所表示的數值也就不同。

523		
5 在百位上則表示5個百	2 在十位上就表示2個十	3 在個位上表示3個一

- $523 = 5 \times 10^2 + 2 \times 10^1 + 3$。

2-2 二進位表示法

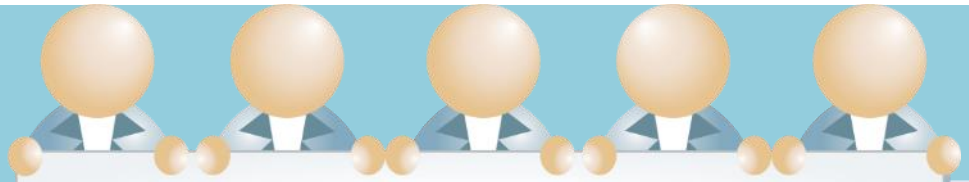
- 若以B為基數，則 $d_n d_{n-1} \ldots d_2 d_1 . r_1 r_2 \ldots r_{m-1} r_m$ 所表示的數為：

$$d_n \times B^{n-1} + d_{n-1} \times B^{n-2} + \ldots + d_2 \times B^1 + d_1 \times B^0 + r_1 \times B^{-1} + r_2 \times B^{-2} + \ldots + r_{m-1} \times B^{-(m-1)} + r_m \times B^{-m}$$

- 在此 $B^0 = 1$

2-2 二進位表示法

- 電腦電子元件最穩定簡單的狀態為「開(1)」與「關(0)」，故目前通行電腦用二進位符號來儲存資料。
- 因為一個位元組有八個位元，可切成兩個十六進位數，因此電腦系統也常使用十六進位數來顯示資料。
- 十六進位系統的數字0到15，分別以阿拉伯數字的0~9及A~F表示。
- 二位元字串 11010011 可表示成 $D3_{16}$ 或 0xD3 (x起頭，代表該數為十六進位數)。



IT 魔 法 小 百 科

台北101大樓在2004年落成，號稱是當時世界第一大樓。本書作者趙老從上個世紀起，就住在1011樓了，什麼？1011？那不是超高樓層嗎？啊哈！其實是二進位的1011，也就是$1\times2^3 + 0\times2^2 + 1\times2^1 + 1 = 11$，是十進位的11樓啦！

為了要避免混淆，如果不是十進位表示的數字，我們通常會在數字的右下方註明它的基數，例如：1011_2就是指二進位的1011。

2-3 各種進位表示法的轉換

十六進位的數字符號及其所對應的十進位及二進位

十進位	二進位	十六進位	十進位	二進位	十六進位
0	0	0	8	1000	8
1	1	1	9	1001	9
2	10	2	10	1010	A
3	11	3	11	1011	B
4	100	4	12	1100	C
5	101	5	13	1101	D
6	110	6	14	1110	E
7	111	7	15	1111	F

十進位數與二進位數的互換

- 二進位數 $d_n d_{n-1} \ldots d_2 d_1 . r_1 r_2 \ldots r_{m-1} r_m$ 所表示的數為：

$$d_n \times 2^{n-1} + d_{n-1} \times 2^{n-2} + \ldots + d_2 \times 2^1 + d_1 + r_1 \times 2^{-1} + r_2 \times 2^{-2} + \ldots + r_{m-1} \times 2^{-(m-1)} + r_m \times 2^{-m}$$

- 只要將每個二進位數字和它所對應的2的次方項(以十進位表示)相乘即可。

範例 1

10110101.1101_2所對應的十進位數為181.8125

	1	0	1	1	0	1	0	1	.	1	1	0	1
	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0		2^{-1}	2^{-2}	2^{-3}	2^{-4}
→	2^7	+	2^5	+	2^4	+	2^2	+	2^0	+	2^{-1}	+	2^{-2}	+	2^{-4}
=	128	+	32	+	16	+	4	+	1	+	0.5	+	0.25	+	0.0625
=	181.8125														

範例 2

十進位181所對應的二進位數為10110101_2

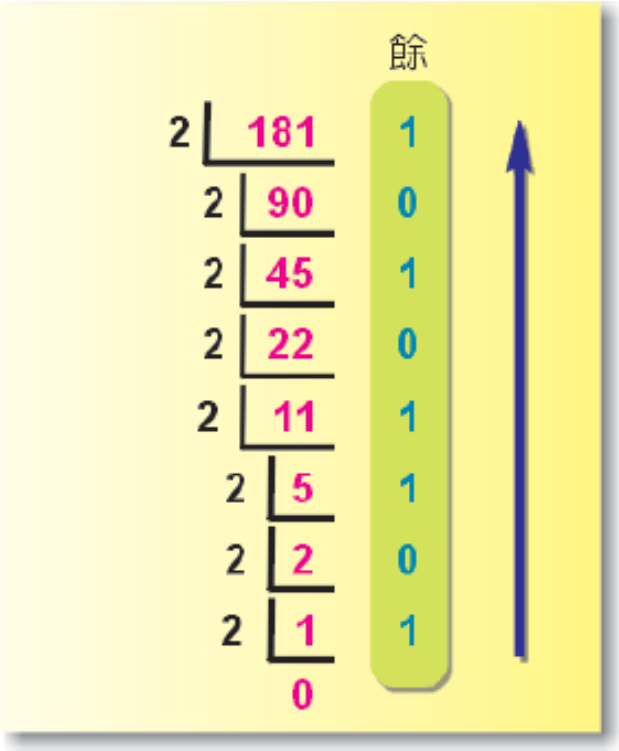
181÷2得商數90，餘數1

→d_1為 1

90÷2得商數45，餘數0

→d_2為 0

...以此類推。



範例 3

十進位0.8125所對應的二進位數為0.1101_2

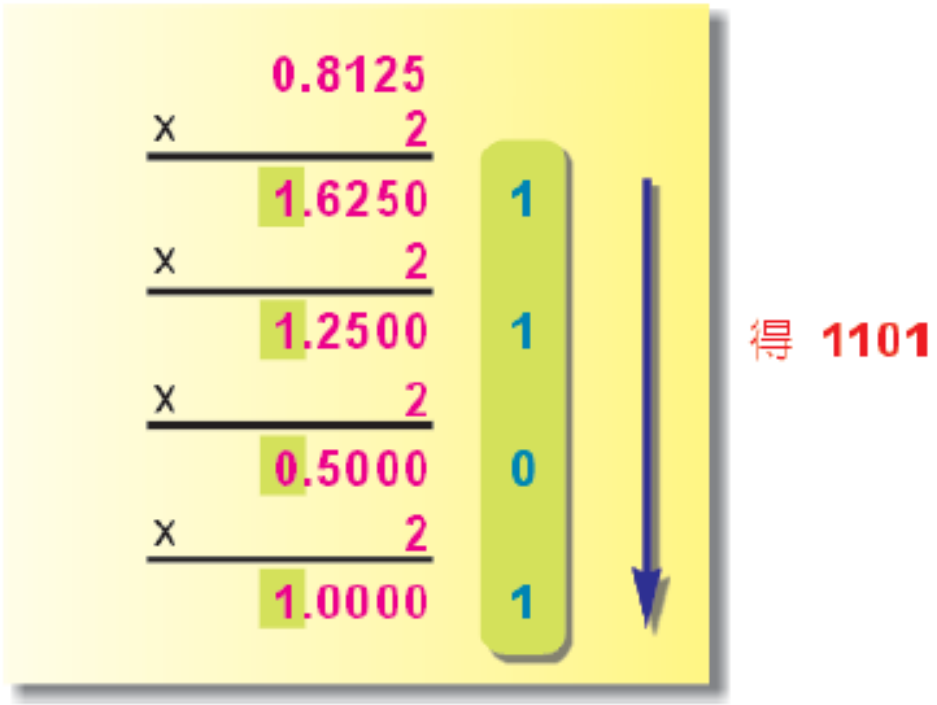
$0.8125 \times 2 = 1.625$

$\rightarrow r_1$為1

剩下小數$0.625 \times 2 = 1.25$

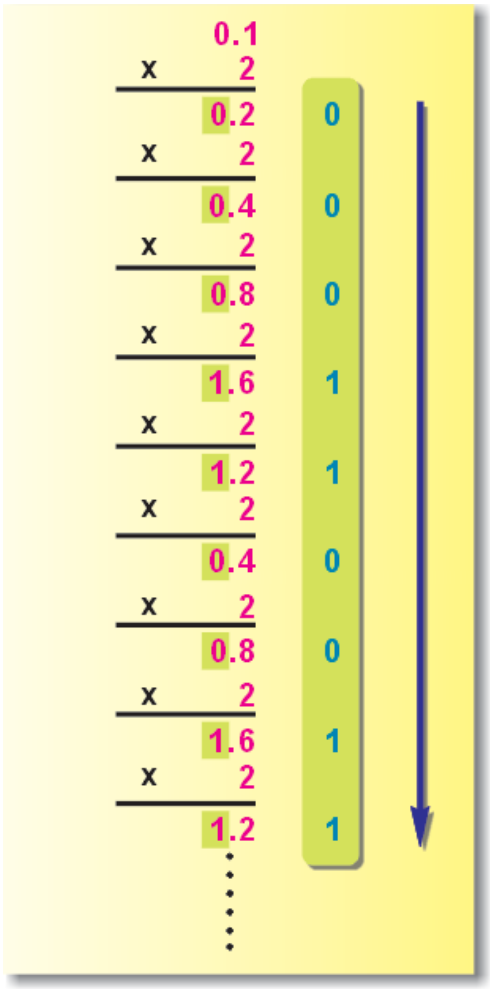
$\rightarrow r_2$為1

...以此類推。



範例 4

十進位 0.1 所對應的二進位數為無窮位數 $0.000110011\ldots_2$

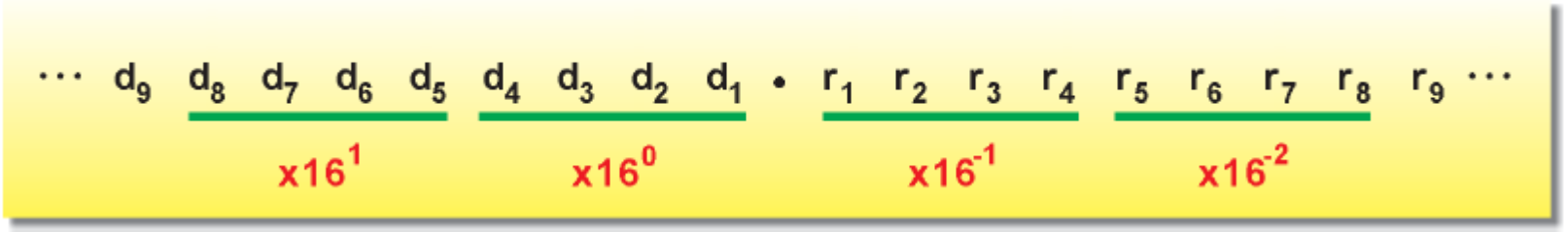


得 0.000110011...

(為循環小數 $0.\overline{00011}$)

二進位數與十六進位數的互換

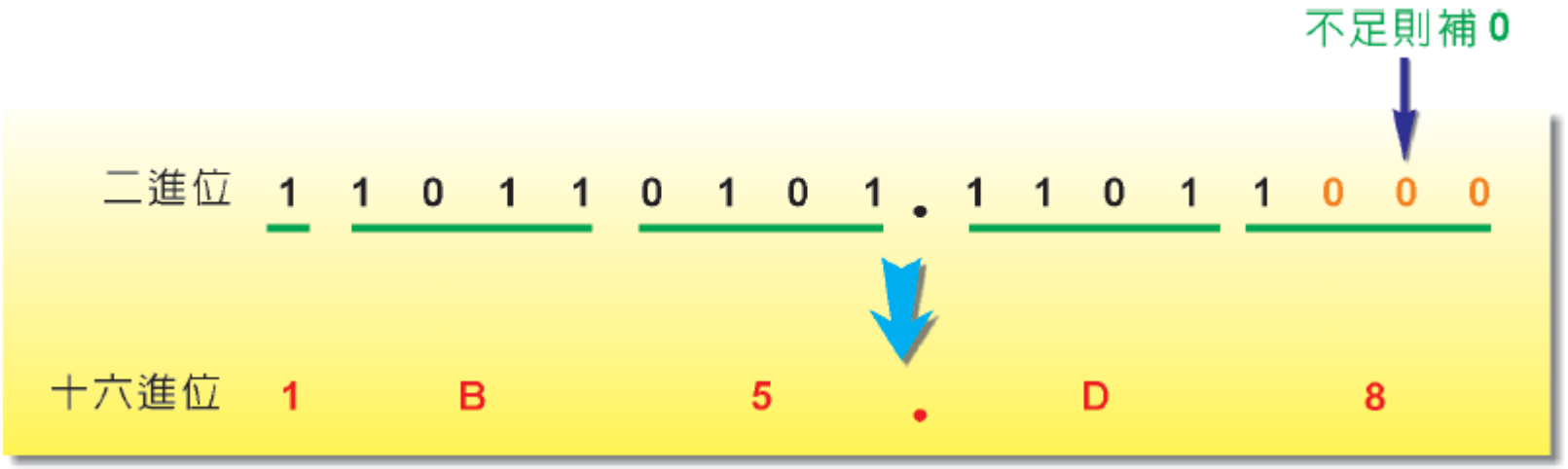
- 因為16為2的整數次方，所以二進位數和十六進位數可說是系出同門。



二進位數換成十六進位數時，每四個位數合成一項

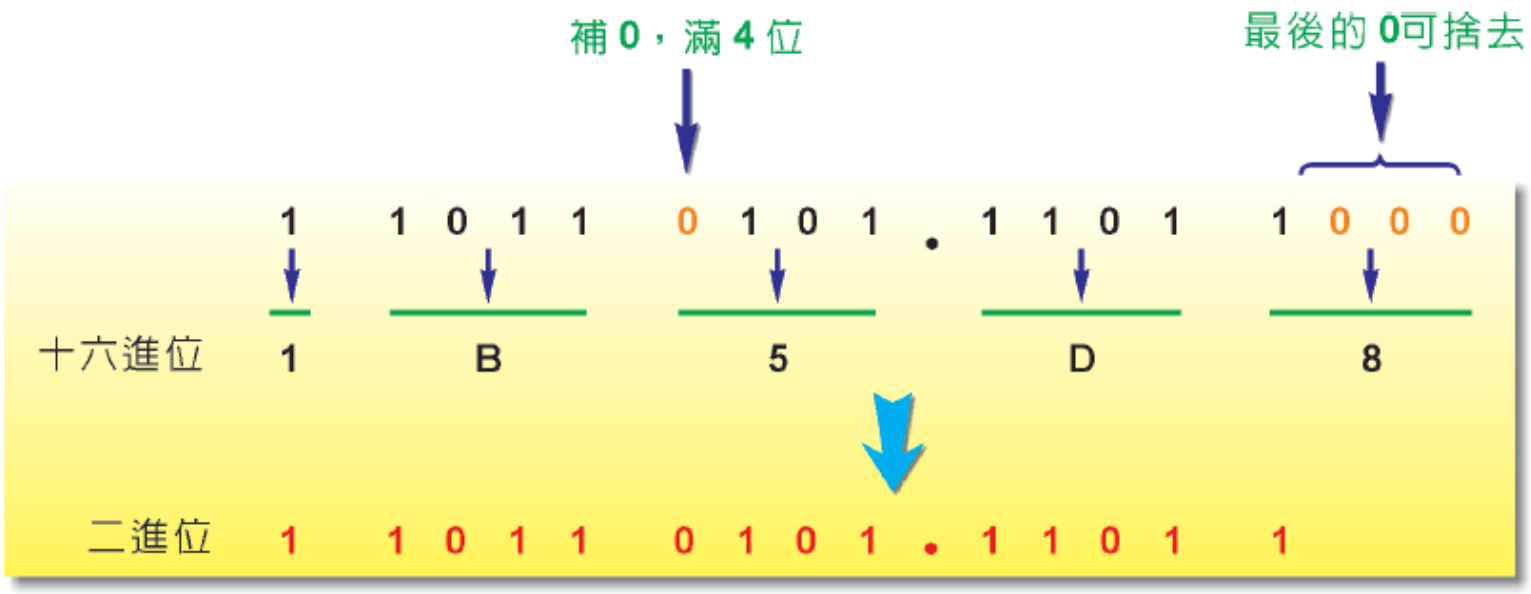
範例 5

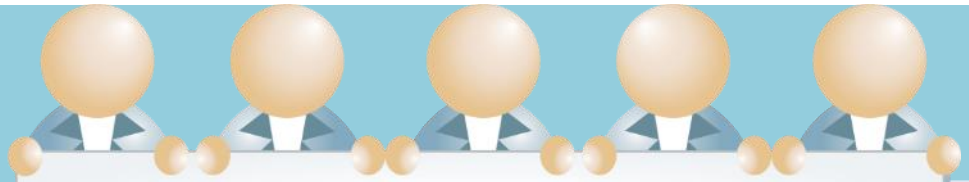
110110101.11011_2 的十六進位表示法為 $1B5.D8_{16}$



範例 6

$1B5.D8_{16}$ 的二進位表示法為 110110101.11011_2





IT 魔 法 小 百 科

「二八年華」常被用來形容含苞待放的青春歲月，指的是「二乘以八」等於十六歲左右的年輕朋友們，本書作者趙老撰寫初版時已是百戰沙場的歐吉桑，居然號稱剛度過二八年華，這到底是怎麼一回事呢？原來是十六進位的二十八，也就是 $x28 = 2\times16^1 + 8 = 40$。有道是：「二八年華應猶在，只是進位改。」

2-4 整數表示法

- 只表示非負的整數，只要將最小的位元字串(亦即全為0的字串)給0，依序表示到最大的數即可。
- *n* 個位元就可表示 2^n 個數，所表示的整數範圍為 $0 \sim 2^n-1$。
- 例如：使用8個位元，可表示$0 \sim 2^8-1$間的所有整數，也就是從0~255的所有整數。

無正負符號的整數

- 位元字串與十進位數的對應表

以8位元所表示的「無正負符號的整數」

位元字串	十進位數
00000000	0
00000001	1
00000010	2
⋮	⋮
11111110	254
11111111	255

帶正負符號大小表示法

- 若要同時表示正數和負數，最直接的作法是採用「帶正負符號大小表示法」。
- 位元字串的最左邊位元當作符號位元(0為正數；1為負數)，剩下的n-1個位元用來表示數的大小。
  - 以位元0開頭的整數範圍為 $0 \sim 2^{n-1}-1$
  - 以位元1開頭的整數範圍為 $0 \sim -(2^{n-1}-1)$

帶正負符號大小表示法

- 若使用8個位元，則可表示 $-(2^7-1) \sim 2^7-1$ 間的所有整數(-127 ~ 127)。
- 此法的潛在問題：
  - 有兩個0，+0(000...00)和-0(100...00)。
  - 正數和負數的運算(例如加和減)並不直接。
- 目前電腦並不採用這種方法表示整數。

以8位元所表示的「帶正負符號大小表示法」

位元字串	十進位數
00000000	0
00000001	1
⋮	⋮
01111111	127
10000000	-0
10000001	-1
⋮	⋮
11111111	-127

補數表示法

- 補數的概念是指要補多少才滿。
- 假設到超級市場買東西，共買793元，若付千元大鈔：
  - 將千元大鈔放一旁，嘴巴唸793，在另一旁拿出1元，唸794；再拿出1元，唸795；再拿出5元，唸800；再拿出100元，唸900；再拿出100元，唸1000。共拿出1+1+5+100+100=207元，正好是要找的錢。
  - 793元還差207元就可「補」成1000元。

一補數表示法

- 「一補數表示法」與「二補數表示法」仍以位元字串最左邊的位元當作符號位元(0為正數；1為負數)；其餘的n-1個位元則用來表示正負符號外的數值大小。
- 其正數表示方式與「帶正負符號大小表示法」相同，負數表示法則有所不同。

一補數表示法

- 十進位數值轉換成一補數表示法，步驟如下：

步驟 1	步驟 2	步驟 3
• 先忽略其符號，將數字的部分轉成二進位數值。	• 若二進位數值超過n-1個位元，則為溢位(overflow)，無法進行轉換；否則在它的左邊補0，直到共有n個位元為止。	• 若所要轉換的數為正數或零，則步驟2所得數值即為所求；若為負數，則將每個位元做補數轉換(原0轉1；原1轉0)。

範例 7

41的一補數表示法為何？

第一步

將41轉成二進位數值101001。

第二步

在二進位數值左邊補上0，使00101001共有8個位元，因為要表示的數為正數，所以00101001即為所求。

範例 8

-41的一補數表示法為何？

第一步

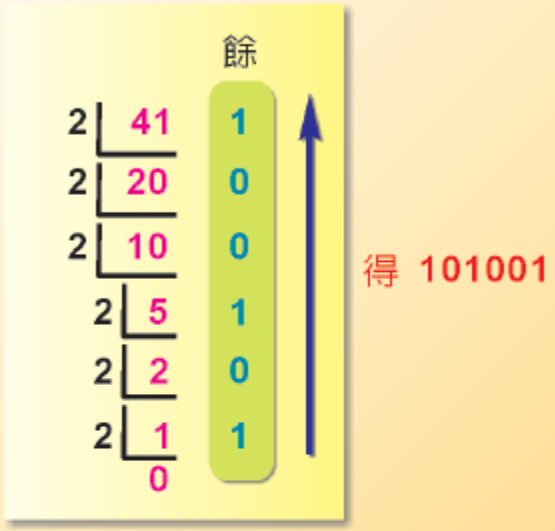
將41轉成二進位數值101001。

第二步

在二進位數值左邊補上0，使00101001共有8個位元；因要表示負數，所以將原為0的轉成1；原為1的轉成0，得11010110。

-41的八位元一補數表示法為11010110

第一步：將41轉成101001



第二步：在左邊補滿0，得00101001

第三步：做補數動作，得11010110

00101001

0變1&1變0

11010110

範例 9

一補數「11010110」所表示的值為多少？

第一步

因為最左邊位元是1，所以將補數原0轉1；原1轉0，得00101001。

第二步

再將二進位的00101001轉成十進位的41，然後加上一個負號，得-41。

一補數表示法

- 一補數法也碰到「兩個0」的問題。
  以八位元為例，00000000和11111111都是0，
  會造成計算上的困擾。
- 其加減法也不是那麼直接。
- 所以一補數法並非目前電腦表示整數所用的方式。

二補數表示法

- 「二補數表示法」是目前電腦表示整數所用的方法。
- 補數方法：
  - 以位元字串最左邊的位元當作符號位元，以它來表示數的正負(0為正數；1為負數)。
  - 其餘n-1個位元則用來表示正負符號外的數值大小。

二補數表示法

- 十進位數值轉換成二補數表示法，步驟如下：

步驟 1	步驟 2	步驟 3
• 先忽略其符號，將數字的部分轉成二進位數值。	• 若二進位數值超過n-1個位元，則為溢位(overflow)，無法進行轉換；否則在它的左邊補0，直到共有n個位元為止。	• 若要轉換數為正數或零，則步驟2所得數值即為所求；若為負數，則最右邊的那些0及最右邊的第一個1保持不變，將其餘位元做補數轉換(原0轉1；原1轉0)。

範例１０

40的二補數表示法為何？

第一步

先將40轉成二進位數值101000。

第二步

在二進位數值左邊補0，使00101000共有8個位元，因要表示數為正數，故00101000即為所求。

範例１１

-40的二補數表示法為何？

第一步

先將40轉成二進位數值101000。

第二步

在二進位數值左邊補0，使00101000共有8個位元。因要表示數為負數，所以最右邊的三個0及第一個1維持不變，其餘將原0轉1；原1轉0，得11011000。

範例１２

二補數「11011000」所表示的值為多少？

第一步

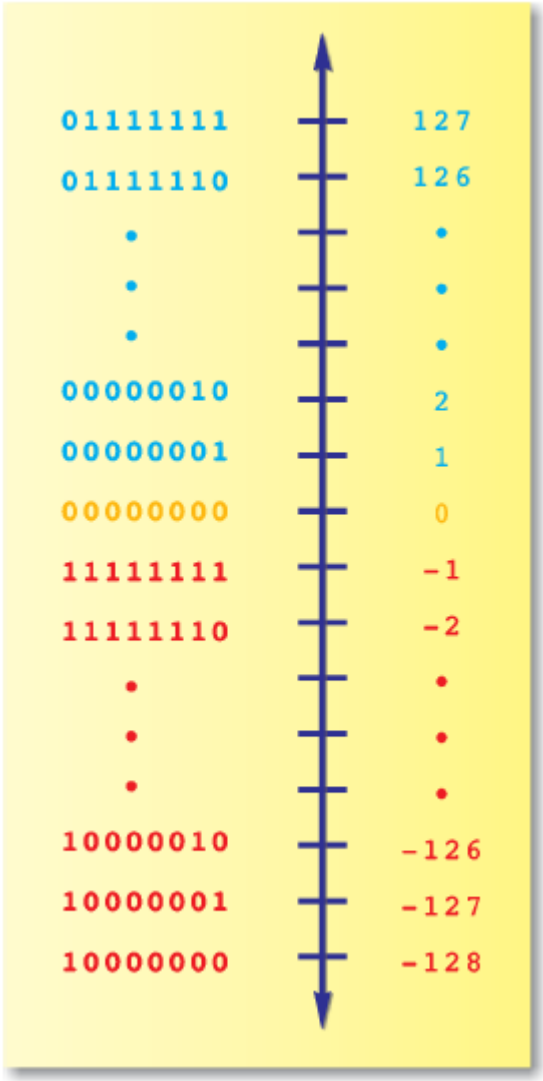
因為最左邊位元是1，所以先保留最右邊的三個0及最右邊的第一個1，再將其他位元原0轉1；原1轉0，得00101000。

第二步

再將二進位00101000轉成十進位的40，然後加上一個負號，得-40。

二補數表示法

- 二補數的0只有一個，以八位元為例，就是00000000。



八位元二補數表示法的位元字串與數值之對應

二補數的加法

- 先將所加的兩個數之二補數位元對齊，從最右邊的位元開始加起，若相對位置的位元相加為二或以上，則有進位。
- 若有進位，則往左邊傳遞；若最左邊位元相加有進位，則忽略這個進位。
- 兩正數相加後，若最左邊符號位元為1，則有溢位(overflow)；兩負數相加後，若最左邊符號位元為0，則有溢位(overflow)。

範例１３

二補數表示法的兩正數相加

			1					← 進位	
	16	0	0	0	1	0	0	0	0
+)	24	0	0	0	1	1	0	0	0
	40	0	0	1	0	1	0	0	0

範例１４

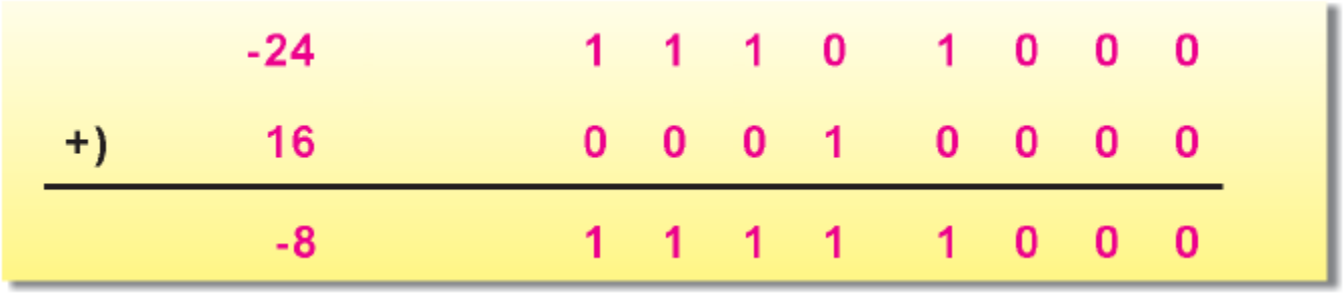
二補數表示法的一正一負相加，且結果為正

最左邊的位元相加有進位，忽略不管。

範例１５

二補數表示法的一正一負相加，且結果為負

最左邊的位元相加並沒有進位。



	-24	1 1 1 0 1 0 0 0
+)	16	0 0 0 1 0 0 0 0
	-8	1 1 1 1 1 0 0 0

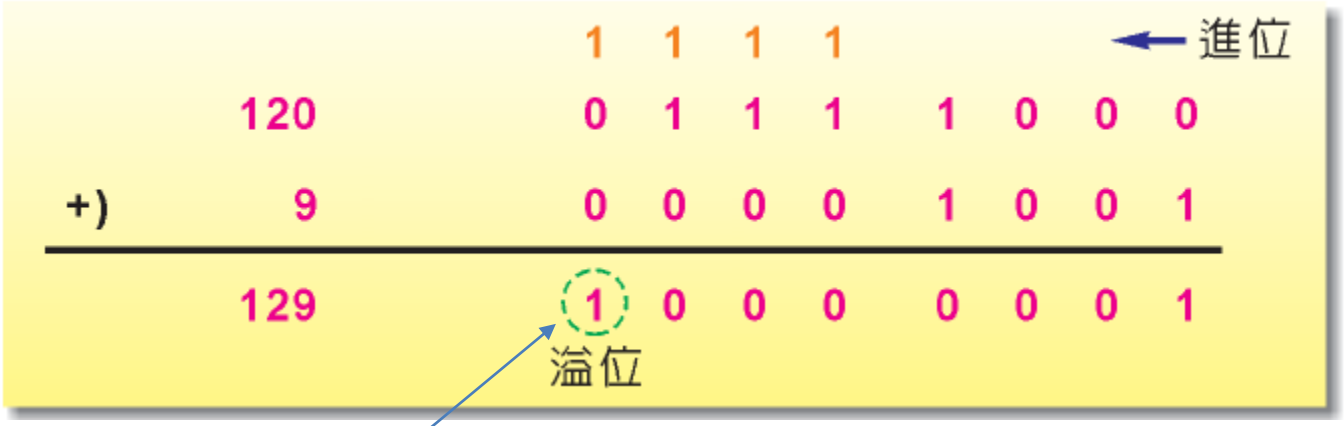
範例１６

二補數表示法的兩負數相加

最左邊的位元相加有進位，忽略不管。

範例１７

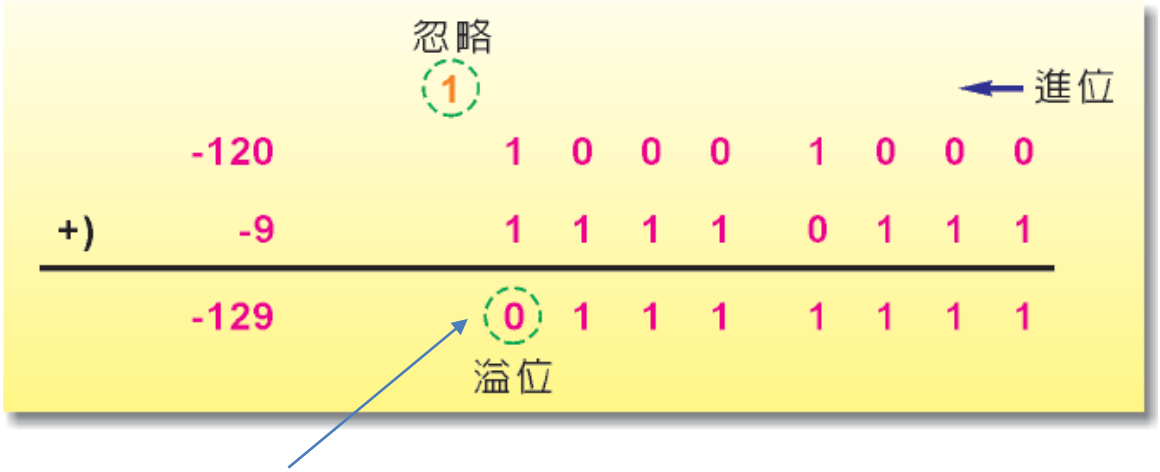
二補數表示法的兩正數相加結果超過正數儲存範圍，稱為溢位(overflow)。



符號位元相加結果為1，也就是兩正數相加反為負數。是因為8位元的二補數最大正數為$2^{n-1}-1$ (=127)，而在此結果為129，已超過正數儲存範圍。

範例１８

二補數表示法的兩負數相加結果小於負數儲存範圍，稱為溢位(overflow)。



符號位元相加結果為0，也就是兩負數相加反為正數。是因為8位元的二補數最小負數為 -2^{n-1} (=-128)，而在此結果為-129，已小於負數儲存範圍。

二補數的加法

- 牽涉到負數的二補數加法，情況比較複雜：
  - 40的二位元字串為00101000，-40的二補數字串是11011000，將位元符號視為數值的一部分，將二進位字串換成十進位，得216，正好是256-40，也就是2^8-40。
  - 24的二位元字串為00011000，-24的二補數字串是11101000，將位元符號視為數值的一部分，將二進位字串換成十進位，得232，正好是256-24，也就是2^8-24 。

二補數的加法

- 2^8的二進位字串 = 1 0000 0000。

	2^8	1	0	0	0	0	0	0	0	0
–)	40		0	0	1	0	1	0	0	0
	-40二補數表示法		1	1	0	1	1	0	0	0
			0變1；1變0				維持不變			

- 結論：一個二補數負數 –x 所表示成的二位元字串數值為 2^n-x。

二補數的加法

- 令 x 和 y 為兩正數， x +(-y) 就是一正一負相加的情況，-y 的二補數表示法之數值為 2^n-y。

狀況一

x > y

- 相加結果 x-y 應為正數。
- 二補數相加，得到x+(2^n-y)= 2^n+(x-y)，在此的 2^n 會造成最左邊忽略掉的進位，故得 x-y。

二補數的加法

狀況二
x = y

- 相加結果 x-y 應為0。
- 二補數相加，得到 $x+(2^n-y)=2^n+(x-y)$，在此的 2^n 會造成最左邊忽略掉的進位，故得 x-y=0。

狀況三
x<y

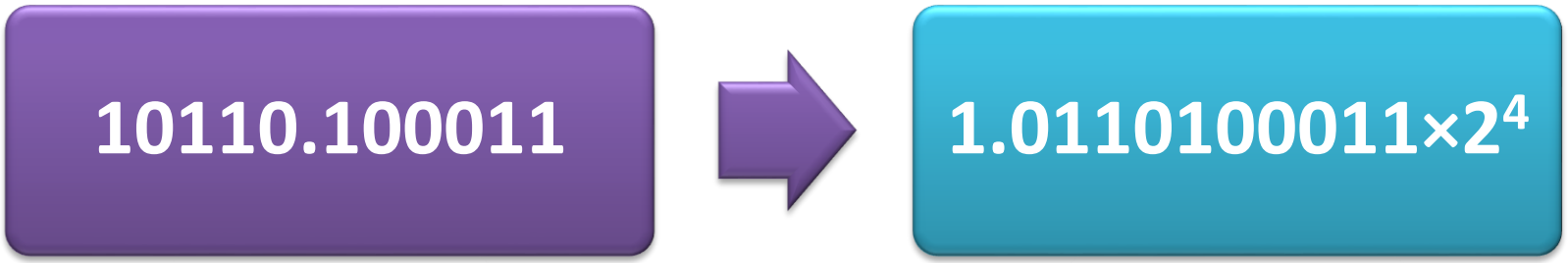
- 相加結果 x-y 應為負數，其值為 -(y-x)。
- 二補數相加，得到 $x+(2^n-y)=2^n+(y-x)$，在此的 2^n 會造成最左邊忽略掉的進位，-(y-x)的二補數數值正好就是 $2^n-(y-x)$。

2-5 浮點數表示法

- 浮點數表示法是電腦表示實數最常用的方式。
- 「536.87」表示成科學記號為「5.3687×10^2」，浮點數表示法的運作原理亦同，會移動小數點，使其「浮動」到標準的位置。
- 在有限位元數的情況下，浮動小數點所能表示的數值範圍比固定小數點位置的方式大許多。

2-5 浮點數表示法

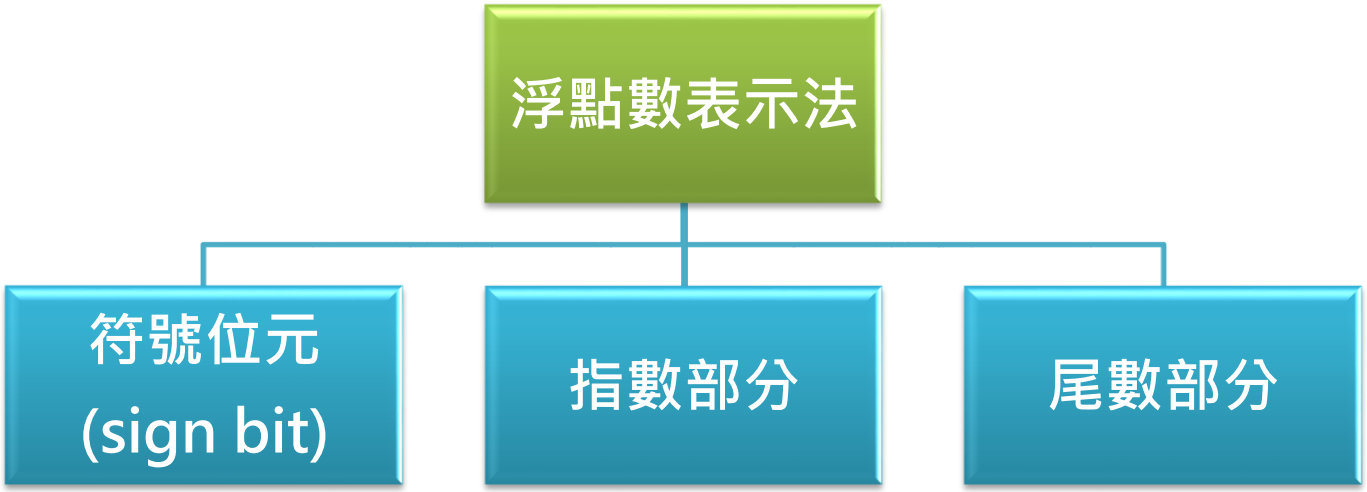
- 科學記號標準化動作：



  - 小數點左邊的數值一定是1。
  - 小數點右邊的0110100011稱為**尾數**(mantissa)，而**指數**(exponent)為4。

2-5 浮點數表示法

- 目前所採用的浮點數表示法以IEEE 754標準為主，主要有三部分：



2-5 浮點數表示法

- **單倍精準數**：以1個位元表示符號；8個位元表示指數；23個位元表示尾數部分。
- **雙倍精準數**：以1個位元表示符號；11個位元表示指數；52個位元表示尾數部分。

單倍精準數

- 符號位元：1個位元，以0表示正數；以1表示負數。
- 指數部分：8個位元，以過剩127(Excess 127：將位元數值減去127所得的值，才是真正所儲存的值)方式表示。8個位元所存的數值可從0～255，共有2^8種變化。
- 尾數部分：23個位元，從標準化的小數點後開始存起，不夠的位元部分補0。

範例１９

給定一實數 10110.100011，轉換成二進位為？

第一步

轉換成 1.0110100011×2^4 是正數，故符號位元為0，尾數部分為0110100011，指數部分為4，以過剩127方式儲存，須加上127，得131。

第二步

將131轉換成二進位，得10000011 。因此10110.100011若按IEEE 754標準儲存，為01000001101101000110000000000000。

範例２０

給定一實數 -0.0010011，轉換成二進位為？

第一步

轉換成 -1.0011×2^{-3} 是負數，故符號位元為1，尾數部分為0011，指數部分為-3，以過剩127方式儲存，須加上127，得124。

第二步

將124轉換成二進位，得01111100。因此-0.0010011若按IEEE 754標準儲存，為10111110001100000000000000000000。

範例２１

01000010100101000110000000000000 所儲存的數值為多少？

第一步

位元符號為0，所以是正數，指數部分是10000101＝十進位133，再減去127，得6。

第二步

01000010100101000110000000000000所儲存的數值為1.0010100011×2^{6}，也就是1001010.0011。

範例２２

10000010100101000110000000000000 所儲存的數值為多少？

第一步

位元符號為1，所以是負數，指數部分是00000101＝十進位5，再減去127，得-122。

第二步

10000010100101000110000000000000所儲存的數值為$-1.0010100011\times2^{-122}$。

單倍精準數所能表示的數字範圍

- 最小正數為00000000100000000000000000000000，其數值為 $+2^{-126}$
- 最大正數為01111111011111111111111111111111，其數值為 $(2-2^{-23})\times 2^{127}$。
- 最大負數為10000000100000000000000000000000，其數值為 -2^{-126}
- 最小負數為11111111011111111111111111111111，其數值為 $-(2-2^{-23})\times 2^{127}$。

2-6 ASCII及Unicode

- 美國國家標準局在1963年時發表的ASCII(唸成Asskey；美國國家資訊交換標準碼)是當今最普及的公定標準。
- 標準ASCII以7個位元儲存一字符，共有$2^7=128$種組合。電腦的儲存常用的位元組為8個位元，多出來的位元用來儲存錯誤檢驗位元(parity bit)。
- 擴充型ASCII用8個位元儲存一字符，有$2^8=256$種組合，可儲存非英文符號、圖形符號及數學符號等。

請點  ASCII符號對照表

Unicode

- 美國萬國碼制訂委員會於1988-1991年間訂定的Unicode (萬國碼) 字符編碼標準，已成為ISO認證之標準(ISO10646)。
- Unicode發展出下列多種編碼方式：
  - UTF-8 在全球資訊網最通行。
  - UTF-16 為JAVA及Windows所採用。
  - UTF-32 則為一些UNIX系統使用。

Unicode

- Unicode前面128個符號為ASCII字符，其餘則為英、中、日、韓文以及其他非英語系國家之常用文字。
- Unicode中最大宗的分類是CJK，主要是中文、日文及韓文之漢字集。

在http://www.unicode.org/charts/ 網址裡，提供了各種不同類別字符的對照表。

Unicode符號對照表

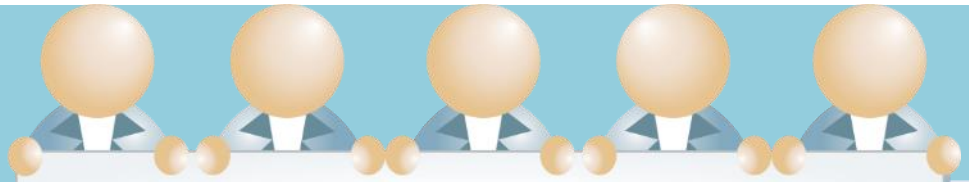
範　圍	代表的字符群
0000-007F	基本拉丁字符（與ASCII相同）
0080-024F	擴充的拉丁字符
0370-03FF	希臘字符
0E00-0E7F	泰文
0E80-0EFF	寮文
2200-22FF	數學符號
2500-25FF	方塊圖形及幾何圖形
3040-30FF	平假名及片假名
4000-9FFF	CJK；中文、日文及韓文之漢字

EBCDIC

- 除了ASCII和Unicode外，IBM的EBCDIC也是某些機型上常用的編碼方式。
- 國際標準局(ISO)用四個位元組(也就是32位元)制定一種編碼方式，可以有2^{32}種組合，可表示多達4,294,967,296種字符。

Big5 / GB

- 以正體字而言，大五碼(Big5；約一萬六千字)是廣受歡迎的一種編碼方式，盛行於台灣及香港。
- 以簡體字而言，國標(GB；約八千字)是廣受歡迎的編碼方式，盛行於大陸地區。
- 這些字體已逐步被包含於Unicode的CJK字集中，未來的整合一致化指日可待。



IT 魔 法 小 百 科

在實際應用上，Unicode 並非皆以16位元儲存字元。以UTF-8為例，傳統的ASCII字符仍以一個位元組儲存(位元組首位為0，後面的7位元為原ASCII的編碼)，其餘非ASCII字符，再依類別而有不同長度的編碼方式。

例如：

「A」的UTF-16為「0041」，UTF-8則為「41」；「趙」的UTF-16為「8D99」，UTF-8則為「E8B699」。