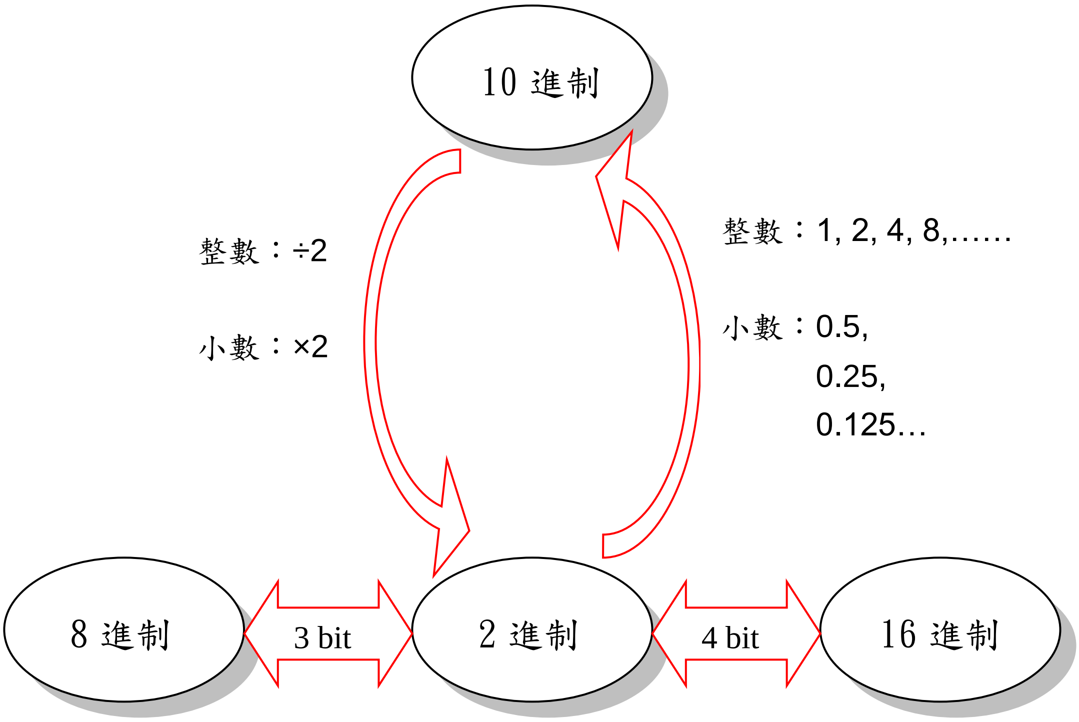
進制轉換



二進制(Binary)補數(complement)：

1’sc = 0 和 1 顛倒過來即是

2’sc = 1’sc + 1

例如：

11001000_2 的 1’sc = 00110111_2 (0 和 1 顛倒過來)

11001111_2 的 2’sc = 00110001_2 (先轉成 1 的補數再加 1)

光碟速度：

VCD 1x = 150KB/sec

DVD 1x = VCD 9x = 1350KB/sec

進制對照表

10 進制	2 進制	8 進制	16 進制
0	0000	0	0
1	0001	1	1
2	0010	2	2
3	0011	3	3
4	0100	4	4
5	0101	5	5
6	0110	6	6
7	0111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F

用於數量

$Y = 10^{24} \fallingdotseq 2^{80}$

$Z = 10^{21} \fallingdotseq 2^{70}$

$E = 10^{18} \fallingdotseq 2^{60}$

$P = 10^{15} \fallingdotseq 2^{50}$

$T = 10^{12} \fallingdotseq 2^{40}$

$G = 10^{9} \fallingdotseq 2^{30}$

$M = 10^{6} \fallingdotseq 2^{20}$

$K = 10^{3} \fallingdotseq 2^{10}$

用於時間

$m = 10^{-3}$

$\mu = 10^{-6}$

$n = 10^{-9}$

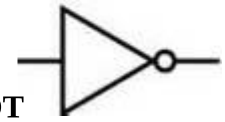
$p = 10^{-12}$

$N^0 = 1$， $(N \neq 0)$

$2^0 = 1$
$2^1 = 2$
$2^2 = 4$
$2^3 = 8$
$2^4 = 16$
$2^5 = 32$
$2^6 = 64$
$2^7 = 128$
$2^8 = 256$
$2^9 = 512$
$2^{10} = 1024$
$2^{11} = 2048$
$2^{12} = 4096$
$2^{13} = 8192$
$2^{14} = 16384$
$2^{15} = 32768$
$2^{16} = 65536$

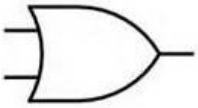
$2^{-1} = 0.5$
$2^{-2} = 0.25$
$2^{-3} = 0.125$
$2^{-4} = 0.0625$
$2^{-5} = 0.03125$
$2^{-6} = 0.015625$

邏輯運算及真值表



NOT

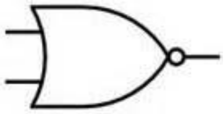
A	B
0	**1**
1	**0**



OR

A	B	Y
0	0	**0**
0	1	**1**
1	0	**1**
1	1	**1**

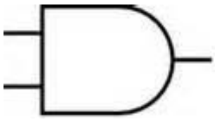
有 1 則 1



NOR

A	B	Y
0	0	**1**
0	1	**0**
1	0	**0**
1	1	**0**

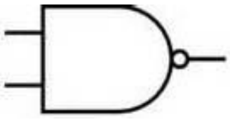
有 1 則 0



AND

A	B	Y
0	0	**0**
0	1	**0**
1	0	**0**
1	1	**1**

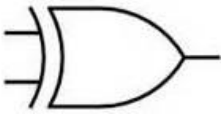
有 0 則 0



NAND

A	B	Y
0	0	**1**
0	1	**1**
1	0	**1**
1	1	**0**

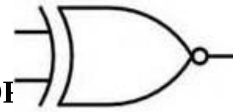
有 0 則 1



XOR

A	B	Y
0	0	**0**
0	1	**1**
1	0	**1**
1	1	**0**

同性相斥，異性相吸



EQV / XNOR

A	B	Y
0	0	**1**
0	1	**0**
1	0	**0**
1	1	**1**

相同則 1

二進制運算：

```
   0          0          1          1
 + 0        + 1        + 0        +  1
 ----       ----       ----       -----
 = 0        = 1        = 1        = 10
```

例：

```
    1010         1111
 +  1001      +     1
 -------      -------
 = 10011      = 10000
```

※　8 進制及 16 進制是用來方便「看」的，不是因為計算的需求。

邏輯運算：

一個一個 bit 運算，沒有進位或借位的問題

例：

```
       1010           1010
 OR    1001     AND   1001
 ----------     ----------
  =    1011       =   1000
```

※ AND 常拿來當「遮罩」用，強迫某些 bit 成為 0

```
       10011011
 AND   11110000
 --------------
  =    10010000
```

（後面 4 個 bit 被遮掉了）

電腦處理資訊的單位(由小到大)：

Bit(位元): 一個 0 或一個 1

Byte(位元組): 8 個 Bit 為一個 Byte

Word(字組): CPU 一次能處理的長度，通常是資料匯流排的寬度

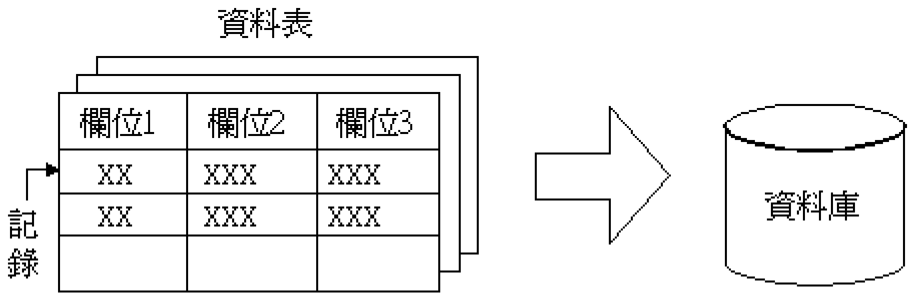
Character(字元): 一個中文或英文字碼

Field(欄位): 由很多字元組成

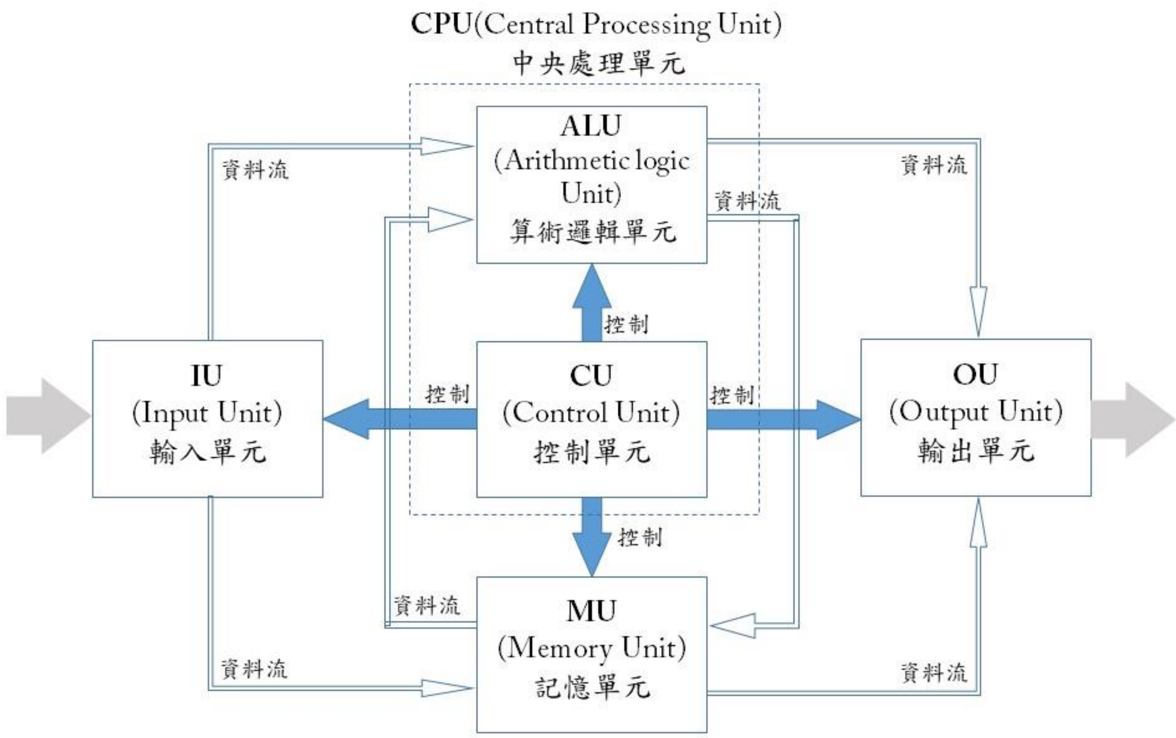
Record(紀錄): 由很多欄位組成

Table(資料表): 由很多記錄組成

Database(資料庫): 由很多資料表組成



五大方塊：



ASCII 編碼表(重要字碼起始位置)

ASCII	Symbol	Description
32		Space
47	/	Slash or divide
48	0	Zero
64	@	At symbol
65	A	Uppercase A
97	a	Lowercase a

例：A 編碼為 65_{10}，則 B 為 66_{10}，依次類推

同位檢查位元（Parity Check Bit）：

奇同位：

在傳輸的資料後面加上一個 bit，將”1”的數目湊成**奇**數個。

資　料								同位元
1	0	1	1	0	0	1	0	1
0	1	1	1	0	1	0	1	0

偶同位：

在傳輸的資料後面加上一個 bit，將”1”的數目湊成偶數個。

資　料								同位元
1	0	1	1	0	0	1	0	0
0	1	1	1	0	1	0	1	1

二進制表示負數方式(2 補數法)

1. 第一個位元用來表示正負號，其餘表大小。

2. 負數要先轉成 2 補數才是它所代表的值。

3. 表示範圍： -2^{n-1} ~ $+2^{n-1}-1$ (n: bit 數)

假設有 3 個位元：

	符號	數 量	
0 開頭，為正數	0	00	+ 0
	0	01	+ 1
	0	10	+ 2
	0	11	+ 3
1 開頭，為負數	1	00 → 100	- 4
	1	01 → 11	- 3
	1	10 → 10	- 2
	1	11 → 01	- 1

4. 因為負數是用 2 補數表示，所以要轉回十進制時須再做一次 2 補數

例 1： 負數 101_2 轉回十進制

1(負號)，大小為 01，其 2 補數為 11，故為 -3

例 2： 負數 11111001_2 轉回十進制

1(負號)，大小為 1111001，其 2 補數為 0000111，故為 -7

另一種方法：將所有位元值加總，減去符號位元值

例 1：

4 1

1 0 1 = (1 + 0) -4(符號位元) = -3

例 2：

128 64 32 16 8 1

1 1 1 1 1 0 0 1 = (1+0+0+8+16+32+64) – 128 = -7

基底補數的減法運算(b'sc)

規則；(1)將減數取 b'sc 後與被減數相加

(2)相加結果：

有進位：將進位丟掉，結果為正值

無進位：將結果再取一次 b'sc，結果為負值

以 2'sc 為例：

例 1：

$0010\ 0011_2\ (35_{10}) - 0000\ 1100_2\ (12_{10}) = 0001\ 0111_2\ (23_{10})$

```
                            0010 0011
0000 1100 的 2'sc         + 1111 0100
                          -----------
                          1 0001 0111
```

有進位，捨去，結果為正值

例 2：

$0010\ 0000_2\ (32_{10}) - 0011\ 0011_2\ (51_{10}) = -0001\ 0011_2(-19_{10})$

```
                            0010 0000
0011 0011 的 2'sc         + 1100 1101
                          -----------
                          0 1110 1101
                2'sc  →     0001 0011
```

無進位，結果再取一次 2'sc，結果為負值