資訊概論

劉承揚

數字系統與數位邏輯

數字系統與數位邏輯-簡介

- 對於電腦來說真正在運作處理的資料都當作是數字的處理，例如文字、字母、標點符號可以組合出文件資訊，聲音、影像也是轉換成數字來處理，以及在電腦裡面運作執行的指令也是由數字來表示的
- 將先介紹在電腦系統中由數字系統的表示資料方式，並簡介控制０與１積體電路開關的基本電路設計方式

3

電腦系統資料表示法

- 電腦系統以位元（bit，binary digit）做為表達資料的基本單位
- 位元的內含值是０或１兩者之一
- 在這裡我們把０、１視為兩個相異的符號，並沒有數量上的意義

4

「位元」和「位元組」的概念

- 單一個位元只能表示兩種狀態，而把數個位元聚合起來形成的位元組（byte），則可以表示更多狀態
- 想像下面有兩個空格，□□，每個格子中你可以填入０或１；這兩個格子視為一組時，我們會得到００、０１、１０、１１這四種可能的樣式（狀態）
- 一個byte是由8個bit所組成，每一個bit可以表示０或１
- 一個byte可以表示出 28 = 256種組合，亦即一個byte可以表示出256各訊息或符號

5

常見的編碼系統

- EBCDIC (Extended Binary Coded Decimal Interchange Code)
  - IBM
  - 8 bit (2^8=256)
- ASCII (American Standard Code for Information Interchange)
  - ANSI (American National Standards Institute)
  - 7 bit
- BIG5
  - 資策會
  - 16 bit
- UNICODE
  - Unicode Consortium
  - 16 bit

6

數字系統之數量單位

單位	簡寫	準確值	近似值
千位元組 (kilobyte)	KB	2^{10}Bytes	10^{3}Bytes
百萬位元組 (megabyte)	MB	2^{20}Bytes	10^{6}Bytes
十億位元組 (gigabyte)	GB	2^{30}Bytes	10^{9}Bytes
兆位元組 (terabyte)	TB	2^{40}Bytes	10^{12}Bytes
千兆位元組 (petabyte)	PB	2^{50}Bytes	10^{15}Bytes
百京位元組 (exabyte)	EB	2^{60}Bytes	10^{18}Bytes

7

數字系統

- 自古以來, 人類習慣以十進位來計算事物, 同時也會使用到其他的數字系統。
  - 最典型的例子如時間的計算, 時與分採六十進位；
  - 而日與時的換算則為二十四進位；
  - 月與年則使用十二進位來計算；
  - 另外我們也會用一打 (12個) 這樣的單位來計算東西的數量,
  - 而傳統的重量單位－台斤與兩的計算, 則是十六進位, 即 1 台斤等於 16 兩。

8

數字系統

- 電腦和人一樣，亦有屬於自己的數字系統，但由於它只能處理 0 與 1 的資料，所以在電腦的世界中只有二進位系統。
- 不過，如果要人類來閱讀由 0 與 1 組成的一長串資料，實在是相當困難；因此對於電腦內部所存的資料，我們一般會以八進位、十六進位來表示，底下將分別對這些數字系統加以介紹。

9

常用的數字系統

- 十進位數字系統：
  - 十進位是一套以 10 為基數，逢 10 即進位的數字系統，由 0、1、2、3、4、5、6、7、8、9 等十個數元所組成，這套數字系統是目前人類世界中最被廣泛採用的一套系統。
- 二進位數字系統：
  - 二進位是一套以 2 為基數，逢 2 即進位的數字系統，在此系統下，任何數都只能用 0 和 1 兩種數元所組成的符號來表示。這套系統即是電腦所使用的數字系統。

10

常用的數字系統

- 八進位數字系統：
  - 八進位是一套以 8 為基數, 逢 8 即進位的數字系統, 在此系統下,我們只能使用 0、1、2、3、4、5、6、7 等八種數元, 如果運算過程中產生了大於或等於 8 的數, 便要往前進位。

11

常用的數字系統

- 十六進位數字系統：
  - 十六進位是一套以 16 為基數, 逢 16 即進位的數字系統, 此數字系統是由 0、1、2、3、4、5、6、7、8、9、A、B、C、D、E、F 等十六個數元所組成, 相對於十進位來看的話, A=10、B=11、C=12、D=13、E=14 、F=15 。
- 下表將十進位數字 0 到 15 分別以二、八與十六進位來表示：

12

常用的數字系統

十進位數字系統	二進位數字系統	八進位數字系統	十六進位數字系統
0	0000	00	0
1	0001	01	1
2	0010	02	2
3	0011	03	3
4	0100	04	4
5	0101	05	5
6	0110	06	6
7	0111	07	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

TIP

一般二進位數值我們都會加個小括弧並標註 2，例如：$(101101)_2$，而十六進位則標註 16，例如 $(ACD8)_{16}$，依此類推。

圖表 4-1　十、二、八、十六進位數字系統對照表

數字系統間的轉換

- 由於電腦內部是以二進位形式來處理資料，所以當我們輸入資料時，電腦會自動將它轉換成二進位的形式。
- 以下就讓我們進一步來探討各數字系統之間互相轉換的方法。

14

數字系統間的轉換

- 二進位與十進位間的轉換
- 八進位與十進位間的轉換
- 十六進位與十進位間的轉換
  - 與十進位互轉的通則
- 八進位與二進位間的轉換
- 十六進位與二進位間的轉換
- 十六進位與八進位間的轉換
  - 二、八、十六進位轉換的通則

15

二進位與十進位間的轉換

- 二進位轉換成十進位
- 十進位轉換成二進位
  - 整數部分
  - 小數部分

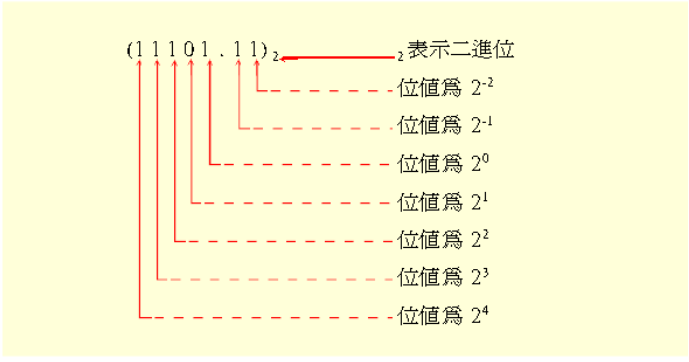
16

二進位轉換成十進位

- 二進位轉換成十進位，其二進位整數部份，在小數點左邊第一位的位值為 2^0、第二位的位值為 2^1、第三位的位值為 2^2 ...；而小數部分，在小數點右邊第一位的位值為 2^{-1}、第二位的位值為 2^{-2} ...等依序類推，只要將每一個二進位數乘以該數的位值，然後相加即可獲得相對應的十進位數值。
- 以下我們以$(11101.11)_2$轉換成十進位來做示範。

17

二進位轉換成十進位



$$(11101.11)_2 = 1 \times 2^4 + 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 + 1 \times 2^{-1} + 1 \times 2^{-2}$$
$$= 16 + 8 + 4 + 0 + 1 + 0.5 + 0.25$$
$$= (29.75)_{10}$$

圖表 4-2　二進位轉換成十進位的示範

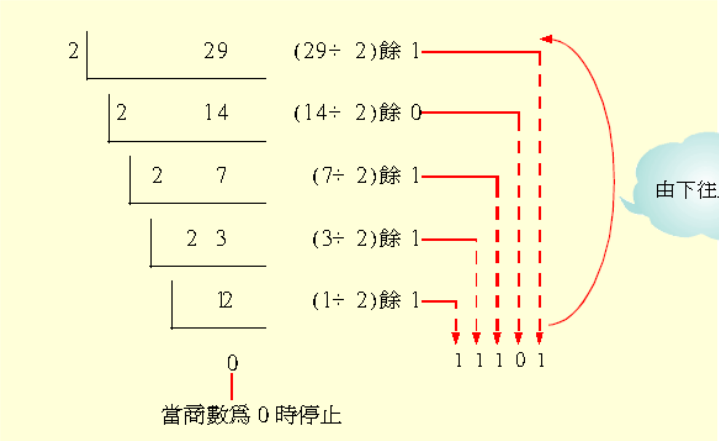
18

十進位轉換成二進位

- 將十進位轉換成二進位，可分為兩個部份來處理；在此我們以 $(29.75)_{10}$ 轉換成二進位來做示範。
- 整數部分
  - 採連續除以 2，並保留「餘數」，直到除法運算後的商數為 0 時停止；然後由最後一次產生的餘數開始，依序由左向右排列，即可完成整數部分的轉換。

19

十進位轉換成二進位-整數部分



圖表 4-3　十進位整數部份轉換成二進位的示範

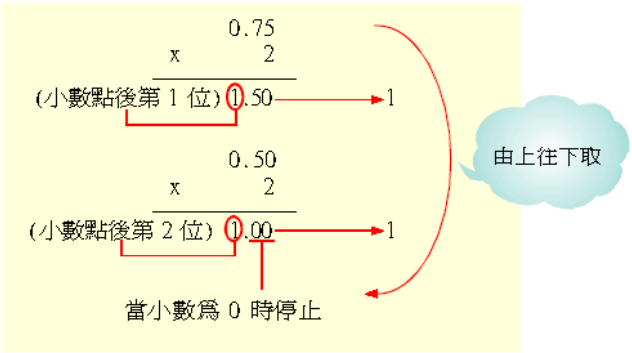
0

十進位轉換成二進位-小數部分

- 將小數部份乘以 2，保留所得乘積的「整數部分」，繼續將乘法運算後所得的小數部分乘以 2，直到所得的小數為 0 時停止；然後由第一次取得的整數開始，依序由左向右排列，即可完成小數部分的轉換。

21

十進位轉換成二進位-小數部分



圖表 4-4　十進位小數部份轉換成二進位的示範

- 最後將整數部份加上小數部份：11101 + 0.11 = 11101.11。所以 $(29.75)_{10} = (11101.11)_2$

22

八進位與十進位間的轉換

- 八進位轉換成十進位
- 十進位轉換成八進位
  - 整數部分
  - 小數部分
- 八進位的轉換原理和二進位相同，其八進位整數部份，在小數點左邊第一位的位值為 8^0、第二位的位值為 8^1 ...。而小數部份，在小數點右邊第一位的位值為 8^{-1}、第二位的位值 8^{-2} ...。因此八進位轉換成十進位，只要將每一個八進位數乘以該數的位值,然後相加即可求得；在此我們以 $(127.3)_8$ 轉換成十進位來做示範。

23

八進位與十進位間的轉換

$$(127.3)_8 = 1x8^2 + 2x8^1 + 7x8^0 + 3x8^{-1}$$
$$= 1x64 + 2x8 + 7x1 + 3x0.125$$
$$= (87.375)_{10}$$

圖表 4-5　八進位轉換成十進位的示範

24

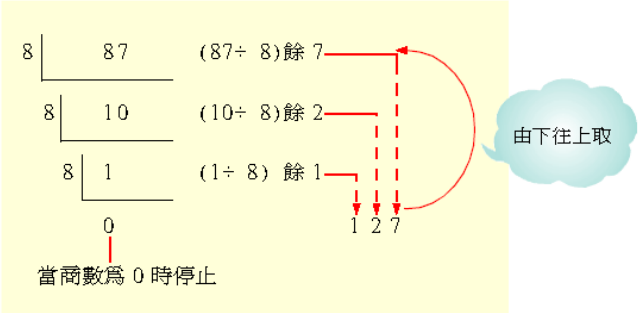
十進位轉換成八進位

- 要將十進位轉成八進位，同樣地可分為整數與小數兩部份來處理；在此我們以 $(87.375)_{10}$ 來示範。

25

十進位轉換成八進位-整數部分

- 採連續除以 8，並保留「餘數」，直到除法運算後的商數為 0 時停止；然後由最後一次產生的餘數開始，依序由左向右排列，即可完成整數部分的轉換。

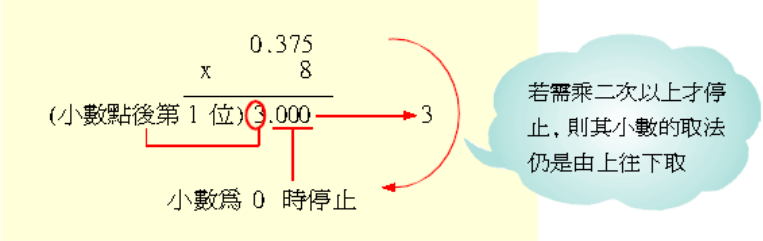


26

圖表 4-6　十進位整數部份轉換成八進位的示範

十進位轉換成八進位-小數部分

- 將小數部份乘以 8，保留所得乘積的「整數部分」，繼續將乘法運算後所得的小數部分乘以 8，直到所得的小數為 0 時停止；然後由第一次取得的整數開始，依序由左向右排列，即可完成小數部分的轉換。



圖表 4-7　十進位小數部份轉換成八進位的示範

- 最後將整數部份加上小數部份：127 + 0.3 = 127.3。所以 $(87.375)_8 = (127.3)_{10}$

27

十六進位與十進位間的轉換

- 十六進位轉換成十進位
- 十進位轉換成十六進位
  - 整數部分
  - 小數部分
- 十六進位的轉換原理和二進位相同，其十六進位整數部份，在小數點左邊第一位的位值為 16^0、第二位的位值為 16^1 ...。而小數部份，在小數點右邊第一位的位值為 16^{-1}、第二位的位值為 16^{-2} ...。因此十六進位轉換成十進位，只要將每一個十六進位數乘以該數的位值，然後相加即可求得；在此我們以 $(BCE.1E)_{16}$ 轉換成十進位來做示範。

28

十六進位轉換成十進位

$$(BCE.1E)_{16} = 11x16^{2} + 12x16^{1} + 14x16^{0} + 1x16^{-1} + 14x16^{-2}$$
$$= 11x256 + 12x16 + 14x1 + 1x0.0625 + 14x0.0039062$$
$$= (3022.67966868)_{10}$$

圖表 4-8　十六進位轉換成十進位的示範

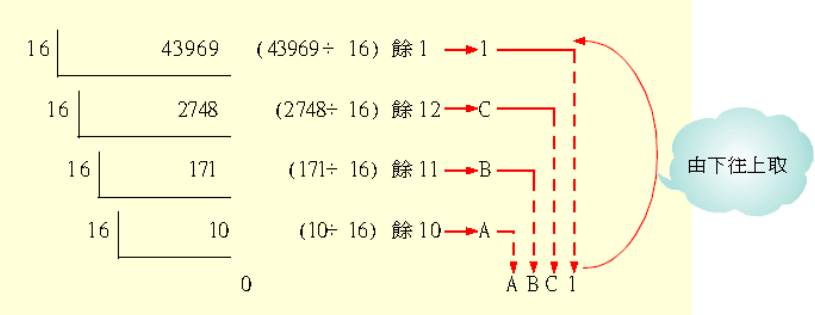
29

十進位轉換成十六進位

- 十進位轉成十六進位的方式，亦分為整數與小數兩部份來處理；在此以 $(43969.6719)_{10}$ 轉成十六進位來做示範。
- 整數部分
  - 採連續除以 16，並保留「餘數」，直到除法運算後的商數為 0 時停止；然後由最後一次產生的餘數開始，依序由左向右排列，即可完成整數部分的轉換。

30

十進位轉換成十六進位-整數部分



圖表 4-9　十進位整數部份轉換成十六進位的示範

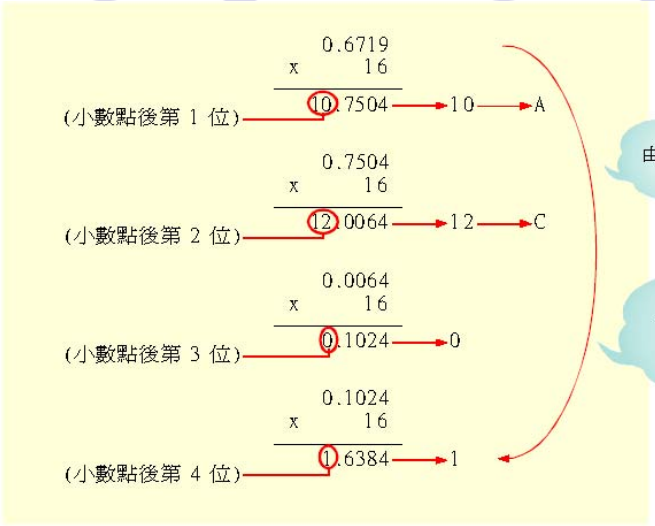
31

十進位轉換成十六進位-小數部分

- 將小數部份乘以 16，保留所得乘積的「整數部分」，繼續將乘法運算後所得的小數部分乘以 16，直到所得的小數為 0 時停止；然後由第一次取得的整數開始，依序由左向右排列，即可完成小數部分的轉換。

32

十進位轉換成十六進位-小數部分



圖表 4-10　十進位小數部份轉換成十六進位的示範

十進位轉換成十六進位

- 最後將整數部份加上小數部份：ABC1+ 0 . AC01 =ABC1.AC01 。
- 所以$(43969.6719)_{10} = (ABC1.AC01)_{16}$

34

與十進位互轉的通則

- 十進位轉成 r 進位的原則：
  - 整數部份除以 r，由下往上取餘數；
  - 小數部份乘以 r，然後由上往下取整數。
- r 進位轉成十進位的原則：
  - 將每個位數乘以對應位值後，全部相加即可。

35

八進位與二進位間的轉換

- 二進位與八進位互相轉換時，請以 3 個 1組為單位來轉換會較為方便。圖表 4-11 為八進位數與等值的二進位數之對照表：

八進位數值	等值之二進位數值
0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111

圖表 4-11
八進位與二進位等值對照表

36

八進位與二進位間的轉換

- 二進位轉換成八進位
- 八進位轉換成二進位

37

二進位轉換成八進位

- 欲將二進位轉換成八進位，只要將二進位的整數部份由右至左，每 3 個分成 1 組，不足 3 個即往前補 0；小數部份則由左至右每 3 個分成 1 組，不足往後補 0，然後再對照上表，將其轉換成對應的八進位數即可。
- 在此我們以 $(11100001110.101)_2$ 轉換成八進位來做示範。

38

二進位轉換成八進位

$$(11100001110.101)_2 = \underset{3}{011}\ \underset{4}{100}\ \underset{1}{001}\ \underset{6}{110}\ .\ \underset{5}{101}$$
$$= (3416.5)_8$$

圖表 4-12　二進位轉換成八進位的示範

39

八進位轉換成二進位

- 同理，當八進位要轉換成二進位時，只要將八進位的數值轉換成每 3 個 1 組的二進位數值即可；在此我們以 $(3416.5)_8$ 轉換成二進位為例來做示範。

$$(3416.5)_8 = (\underset{3}{011}\ \underset{4}{100}\ \underset{1}{001}\ \underset{6}{110}\ .\ \underset{5}{101})_2$$
$$= (11100001110.101)_2$$

圖表 4-13　八進位轉換成二進位的示範

40

十六進位與二進位間的轉換

- 二進位與十六進位轉換時, 採每 4 個 1 組為單位來轉換會較為方便。下表為十六進位與等值的二進位數對照表：

十六進位數值	等值之二進位數值
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001
A	1010
B	1011
C	1100
D	1101
E	1110
F	1111

圖表 4-14　十六進位與二進位等值對照表

十六進位與二進位間的轉換

- 二進位轉換成十六進位
- 十六進位轉換成二進位

42

二進位轉換成十六進位

- 欲將二進位轉換成十六進位，只要將二進位數的整數部份，由右往左每 4 個 1 組進行轉換，不到 4 個時，就在前端補 0；小數部份則是由左往右每 4 個 1 組進行轉換，不到4個時，就在後面補 0，然後再對照上表，將其轉換成對應的十六進位數即可；在此我們以 $(1011111001.0011101)_2$ 轉換成十六進位來做示範。

43

二進位轉換成十六進位

$$(1011111001.0011101)_2 = (\underbrace{0010}_{2}\ \underbrace{1111}_{F}\ \underbrace{1001}_{9}\ .\ \underbrace{0011}_{3}\ \underbrace{1010}_{A})_2$$
$$= (2F9.3A)_{16}$$

圖表 4-15　二進位轉換成十六進位的示範

44

十六進位轉換成二進位

- 同理，當十六進位要轉換成二進位時，只要將十六進位的數值轉換成每 4 個 1 組的二進位數值即可；在此我們以 $(2F9.3A)_{16}$ 轉換成二進位為例來做示範。

$$(2F9.3A)_{16} = (\underset{2}{0010}\ \underset{F}{1111}\ \underset{9}{1001}\ .\ \underset{3}{0011}\ \underset{A}{1010})_2$$
$$= (1011111001.0011101)_2$$

圖表 4-16　十六進位轉換成二進位的示範

45

十六進位與八進位間的轉換

- 八進位轉換成十六進位
- 十六進位轉換成八進位

46

八進位轉換成十六進位

- 欲將八進位轉換成十六進位，可以先轉換成二進位數字，再將二進位轉換成十六進位即可；在此我們以 $(346.7)_8$ 轉換成十六進位為例來做示範：

$$(346.7)_8 = (\underline{011}\ \underline{100}\ \underline{110}\ .\underline{111})_2$$
$$= (0\ \underline{1110}\ \underline{0110}\ .\underline{1110})_2$$
$$\quad\ \ E\quad 6\quad E$$
$$= (E6.E)_{16}$$

圖表 4-17　八進位轉換成十六進位的示範

47

十六進位轉換成八進位

- 欲將十六進位轉換成八進位，必須先轉換成二進位數字之後，再由二進位轉成八進位即可。在此我們以 $(E6.E)_{16}$ 轉換成八進位為例來做示範：

$$(E6.E)_{16} = (\underline{1110}\ \underline{0110}\ .\underline{1110})_2$$
$$= (\underline{011}\ \underline{100}\ \underline{110}\ .\underline{1110})_2$$
$$\quad\ \ 3\quad 4\quad 6\quad .\quad 7$$
$$= (346.7)_8$$

圖表 4-18　十六進位轉換成八進位的示範

48

二、八、十六進位轉換的通則

- 二進位與八進位：
  - 要以 3 個為 1 組來轉換。
- 二進位與十六進位：
  - 要以 4 個為 1 組來轉換。
- 八進位與十六進位：
  - 必須先轉成二進位後再進行轉換。

49

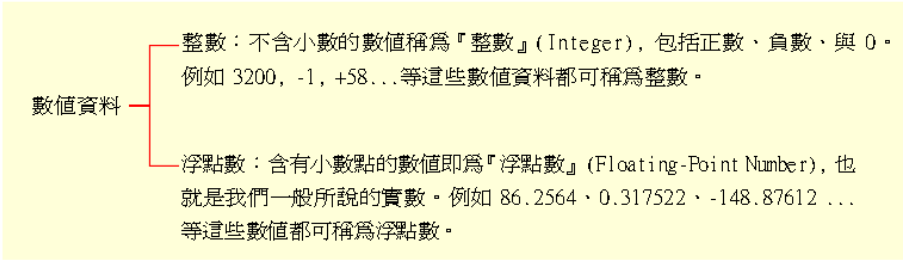
資料表示法

- 數值資料表示法
  - 負數的表示法
  - 浮點資料表示法

50

數值資料表示法

- 數值資料可分為整數與浮點數兩種：



圖表 4-19　數值資料表示法

51

數值資料表示法

- 數值資料中最常見的就是正、負整數的資料，在電腦內部充滿著 0101010101 的訊號，可以用二進位數來表示。但是這樣的二進位數字都只是正整數而已，電腦內部並沒有 "+"、"-" 等符號來表示正、負數，也沒有表示小數點的符號。因此，為了解決這些問題，便有人提出幾種不同的負數與浮點數表示方法。

52

最高位元表示法

- 假設使用n位元來表示正負整數，那麼最左邊的位元 (MSD) 是整數的正負符號，0表示正數，1表示負數，剩下的n - 1位元才是整數的數值大小，正整數的範圍為0 ~ 2^{n-1}-1，負整數的範圍為 -(2^{n-1}-1) ~ 0。

53

1’s 補數

- 假設使用n位元來表示正負整數，那麼最左邊的位元 (MSD) 是整數的正負符號，0表示正數，1表示負數，剩下的n - 1位元才是整數的數值大小，正整數的範圍為0 ~ 2^{n-1}-1，負整數的範圍為 -(2^{n-1}-1) ~ 0。
- 1’s補數的正數表示法和最高位元表示法一樣，但負數表示法就不一樣了，它是將某個正整數的表示法中所有0改為1，所有1改為0，之後得到的二進位字串才是這個正整數對應的負整數。

54

2's 補數

- 假設使用n位元來表示正負整數，那麼最左邊的位元 (MSD) 是整數的正負符號，0表示正數，1表示負數，剩下的n - 1位元才是整數的數值大小，正整數的範圍為0 ~ 2^{n-1}-1，負整數的範圍為 -2^{n-1} ~ 0。
- 2's補數的正數表示法和最高位元表示法、1's補數一樣，但負數表示法就不一樣了，它是將某個正整數的表示法中所有0改為1，所有1改為0，之後得到的二進位字串再加上1，才是這個正整數對應的負整數。

55

數值資料表示法

十進位	最高位元表示法	1's補數	2's補數	十進位	最高位元表示法	1's補數	2's補數
+8	無	無	無	-8	無	無	1000
+7	0111	0111	0111	-7	1111	1000	1001
+6	0110	0110	0110	-6	1110	1001	1010
+5	0101	0101	0101	-5	1101	1010	1011
+4	0100	0100	0100	-4	1100	1011	1100
+3	0011	0011	0011	-3	1011	1100	1101
+2	0010	0010	0010	-2	1010	1101	1110
+1	0001	0001	0001	-1	1001	1110	1111
+0	0000	0000	0000	-0	1000	1111	0000

表2.4不同數值表示法對照表

56

浮點資料表示法

- 浮點數的表示方法相對於正負整數的表示方法，最主要的差別就在於小數點的位置。
- 對於正負整數來說，小數點都固定在最右邊，所以其表示法又稱為定點表示法，定點表示法的規定在所有電腦中都一樣；而浮點數的小數點則是不固定的，而且其顯示方式還因電腦型態不同而有異。

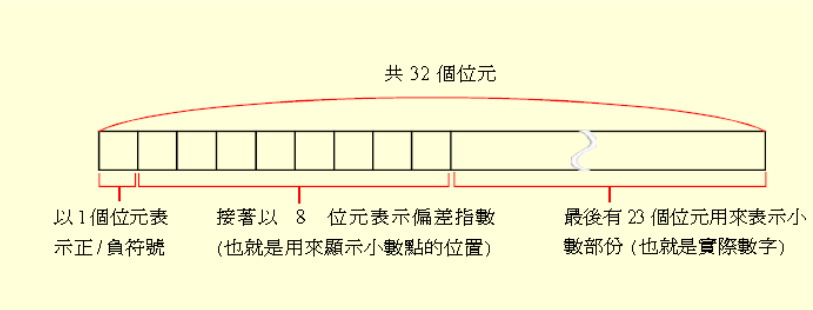
57

浮點資料表示法

- 以 486 電腦的浮點表示法為例，有單精確度 (以 32 位元來顯示浮點數)、雙精確度 (以 64 位元來顯示浮點數)和延伸精確度 (以 80 位元來顯示浮點數) 等三種。以下是單精確度的表示方法：

58

浮點資料表示法



圖表 4-22　單精確度的浮點資料表示法

59

浮點資料表示法

- 正/負符號：0 表示正數, 1 表示負數。
- 偏差指數： 8 個位元可以顯示 0 ~255 個數值, 但必須能顯示正負兩種指數, 故以 127 為指數偏差值, 將指數值+指數偏差值就等於偏差指數。
- 小數部份：這裡的小數部份是指以二進位形式, 且正規化後的浮點數之小數部份。

60

浮點資料表示法

- 接著讓我們以 $(12.25)_{10}$ 為例, 看看如何以單精確度的浮點表示法來表示。首先,它是個正數, 所以第一個位元已經確定是 0 , 接著將之轉換為二進位 $(12.25)_{10} = (1100.01)_2$, 再正規化成 1.10001×2^3 , 此時便可計算出偏差指數為 3+127 = 130 , 再把 130 轉換為以 8 位元顯示的二進位 $(10000010)_2$, 於是可以得到偏差指數的部份就是10000010。

61

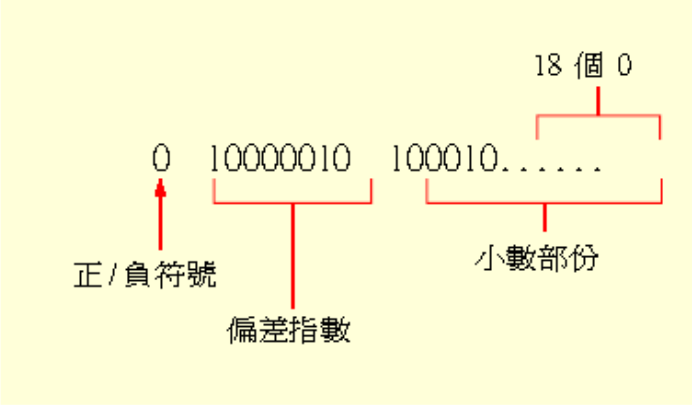
浮點資料表示法

- 最後是小數部份, 小數部份即是正規化後的二進位小數部份, 至於小數點前面那個 1 , 則因為所有正規化的數字都有 1 , 所以電腦會自動記憶, 因此只要顯示後面的小數部份 10001 , 但小數共有 23 個位元, 所以把後面的位元補上 0 即可。

62

浮點資料表示法

- 最後得到的浮點數表示如下：



圖表 4-23　單精確度浮點表示法的示範

63

數位邏輯電路

- 布林代數
- 基本邏輯閘 —AND、OR、NOT
- 其他邏輯閘—NAND、NOR、XOR

64

布林代數

- 布林代數運算所表示的是邏輯上的正確與否，所以布林代數中基本的結果即為此運算之真假；也就是我們一般最廣為人知的0和1，
- 然而布林代數運算種類雖有許多，但任何之邏輯運算仍舊可以基本的四個運算所組成；
  - 這四個運算分別為屬於基本邏輯閘（Logic gate）的AND、OR，和NOT，及屬於萬用邏輯閘的XOR。

65

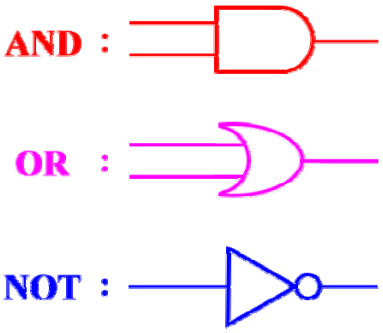
基本邏輯閘

- 以二進位之基本運算為例；基本邏輯閘包括了AND、OR與NOT，每一邏輯閘各有其表示符號，而所做的運算也不同
- 若將其輸入以X和Y表示，則AND運算之結果為X。Y
- 而同樣的輸入下，OR所輸出的便是X＋Y
- 至於NOT便是輸出相反之值了，若輸入為X，則其NOT後之值通常以X’ 表示之。

66

基本邏輯閘

- 基本邏輯閘表示符號



67

基本邏輯閘

- AND

x1	x2	Result
TRUE	TRUE	TRUE
TRUE	FALSE	FALSE
FALSE	TRUE	FALSE
FALSE	FALSE	FALSE

- OR

x1	x2	Result
TRUE	TRUE	TRUE
TRUE	FALSE	TRUE
FALSE	TRUE	TRUE
FALSE	FALSE	FALSE

68

其他邏輯閘：萬用閘

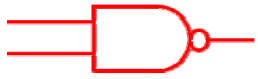
- 萬用邏輯閘便是可將整個電路轉化為只使用單一種邏輯閘而成，而萬用邏輯閘則包括了NAND、NOR與XOR三者
- 若同樣的將其輸入以X和Y示之，則NAND運算之結果為（X’ ．Y’ ）
- 而同樣的輸入下，NOR所輸出的便是（X＋Y）’
- 至於XOR較有不同，其輸出為（XY’ ． X’ Y），然而XOR也是最常被使用到的萬用閘。

69

其他邏輯閘：萬用閘

- 萬用邏輯閘表示符號

NAND ：



NOR ：

XOR ：

70

其他邏輯閘：萬用閘

- XOR

x1	*x2*	*Result*
TRUE	TRUE	FALSE
TRUE	FALSE	TRUE
FALSE	TRUE	TRUE
FALSE	FALSE	FALSE

71