台科大圖書

Digital Logic Design

數位邏輯設計 全

第 5 章 數字系統

5-1 十、二、八、十六進位表示法

一 數字碼分類

在日常生活中，人類早已習慣使用十進位（decimal）中0 至9 等十個數字來表示各種數量，但是十進位表示法並不是唯一可用的數字表示法，其他如二進位、八進位與十六進位等數字表示法，也廣泛地使用在其他領域中。一般而言，利用多個位元可以組成數字碼（codes），而數字碼可以區分為加權碼與非加權碼，如表5-1 所示。

5-1 十、二、八、十六進位表示法

▼ 表 5-1　數字碼的分類

數字碼	名稱	用途
加權碼	十進位	1. 用來表示數值。 2. 可以做算數運算。
	二進位	
	八進位	
	十六進位	
	二進碼十進數（BCD）	
非加權碼	加三碼	1. 用來表示文字與符號等資料。 2. 不可以做算數運算。 3. 作為控制碼用。
	格雷碼	
	美國資訊交換標準代碼（ASCII）	

5-1 十、二、八、十六進位表示法

（一）加權碼

常用的加權碼數字系統有十進位、二進位、八進位、十六進位與二進碼十進數（BCD）等。加權碼的數字系統，一般用來表示數值，並且可以做算術運算（例如加、減、乘、除），無論是哪一種數字系統，其每個數字的加權值（weight）皆不同，所以可以利用5-1 式來表示其數值，即將各數字乘以其加權值，再將各數值加總起來，例如r 進位的數目N 可表示為：

5-1 十、二、八、十六進位表示法

$$N = \underbrace{a_n \cdots a_1\, a_0}_{\text{整數}} \;.\; \underbrace{a_{-1}\, a_{-2} \cdots a_{-m}}_{\text{小數}}{}_{(r)}$$

$$= a_n \times r^n + \cdots + a_1 \times r^1 + a_0 \times r^0 + a_{-1} \times r^{-1} + a_{-2} \times r^{-2} + \cdots + a_{-m} \times r^{-m}$$

5-1 式

5-1 十、二、八、十六進位表示法

其中 r：基底或底數（base）。若$r = 10$，即表示該數值為十進位數。

$a_n \ldots a_{-m}$：即 0 ～ $r-1$之間的任一數字。若十進位時，即 0 ～ 9 之數字。

$r^n \ldots r^{-m}$ ：為$a_n \ldots a_{-m}$ 數字的加權值。

本章將針對十進位、二進位、八進位與十六進位等常用的數字系統做說明，並且介紹彼此之間互換的方法。

5-1 十、二、八、十六進位表示法

（二）非加權碼

常用的非加權碼數字系統有加三碼、格雷碼、美國資訊交換標準代碼（ASCII）等。非加權碼的數字系統，一般用來表示文字與符號等資料，所以不可以做算術運算，一般作為控制碼之用，本章也將介紹各種常用的非加權碼。

5-1 十、二、八、十六進位表示法

二 十進位表示法

十進位（decimal）表示法是人類最常使用的數字表示法，記得小時候每個人學數學時，都會琅琅上口地念「個、拾、百、千、萬、拾萬、百萬等」，這就是我們最熟悉的十進位表示法。

十進位表示法是由0、1、2…9 共十個不同的數字來組成一個數值，也就是以10為基底，逢10 即進位。現在我們舉一個十進位數168. 25，來說明十進位的表示法，如表5-2 所示。

5-1 十、二、八、十六進位表示法

▼ 表 5-2　十進位表示法

項目	說明
十進位表示法	$168.25 = 168.25_{(10)} = 168.25_{(D)}$
加權表示式	$168.25_{(10)} = 1 \times 10^2 + 6 \times 10^1 + 8 \times 10^0 + 2 \times 10^{-1} + 5 \times 10^{-2}$
基底	10
可使用的數字	0 ～ 9
加權值	10^2、10^1、10^0、10^{-1}、10^{-2} 為各個數字的加權值
最高有效位數（Most Significant Digit；簡稱 MSD）	在加權表示式中最左邊的數字 1，其加權值（10^2）最高，所以稱為 MSD。
最低有效位數（Least Significant Digit；簡稱 LSD）	在加權表示式中最右邊的數字 5，其加權值（10^{-2}）最低，所以稱為 LSD。

5-1 十、二、八、十六進位表示法

三 二進位表示法

通常二進位（binary）表示法應用在數位電路、電腦與微處理機等方面，在數位電路中的電壓準位只有高電位與低電位兩種狀態，所以用來表示這兩種狀態的數字定義為位元（bit），其中「1」代表「高電位」，「0」代表「低電位」。

二進位表示法是由0 與1 來組成一個數值，也就是以2 為基底，逢2 即進位。現在我們舉一個二進位數1001.01，來說明二進位的表示法，如表5-3 所示。

5-1 十、二、八、十六進位表示法

▼ 表 5-3　二進位表示法

項目	說明
二進位表示法	$1001.01_{(2)} = 1001.01_{(B)}$
加權表示式	$1001.01_{(2)} = 1 \times 2^2 + 0 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 + 0 \times 2^{-1} + 1 \times 2^{-2}$
基底	2（逢 2 進位）
可使用的數字	0、1
加權值	2^3、2^2、2^1、2^0、2^{-1}、2^{-2} 為各個數字的加權值
最高有效位元（Most Significant Bit；簡稱 MSB）	在加權表示式中最左邊的數字 1，其加權值（2^3）最高，所以稱為 MSB。
最低有效位元（Least Significant Bit；簡稱 LSB）	在加權表示式中最右邊的數字 1，其加權值（2^{-2}）最低，所以稱為 LSB。

5-1 十、二、八、十六進位表示法

四 八進位表示法

目前大部分的數位電路、電腦與微處理機都是使用二進位表示法，但是對於位元數較多的二進位數，較難閱讀與辨識，例如1010010011010110 之二進位數，無法迅速地轉換成人類最常用的十進位數，所以透過八進位與十六進位表示法，可以方便閱讀與轉換成十進位數，因此介紹八進位與十六進位表示法。

八進位（octal）表示法是由0、1、2…7 共八個不同的數字來組成一個數值，也就是以8 為基底，逢8 即進位。現在我們舉一個八進位數153.2，來說明八進位的表示法，如表5-4 所示。

5-1 十、二、八、十六進位表示法

▼ 表 5-4　八進位表示法

項目	說明
八進位表示法	$153.2_{(8)} = 153.2_{(O)}$
加權表示式	$153.2_{(8)} = 1 \times 8^2 + 5 \times 8^1 + 3 \times 8^0 + 2 \times 8^{-1}$
基底	8
可使用的數字	0 ～ 7
加權值	8^2、8^1、8^0、8^{-1} 為各個數字的加權值
最高有效位數（MSD）	在加權表示式中最左邊的數字 1，其加權值（8^2）最高，所以稱為 MSD。
最低有效位數（LSD）	在加權表示式中最右邊的數字 2，其加權值（8^{-1}）最低，所以稱為 LSD。

5-1 十、二、八、十六進位表示法

五 十六進位表示法

十六進位（hexadecimal） 表示法是由0、1、2…9 與A、B、C、D、E、F 共十六個不同的數字與英文字母來組成一個數值，其中A、B、C、D、E、F 分別代表十進制的10、11、12、13、14、15，也就是以16 為基底，逢16 即進位。現在我們舉一個十六進位數5E.8，來說明十六進位的表示法，如表5-5 所示。

5-1 十、二、八、十六進位表示法

表 5-5　十六進位表示法

項目	說明
十六進位表示法	$5E.8_{(16)} = 5E.8_{(H))} = 5E.8H$
加權表示式	$5E.8_{(16)} = 5 \times 16^1 + 14 \times 16^0 + 8 \times 16^{-1}$
基底	16
可使用的數字	0 ～ 9 與 A ～ F
加權值	16^1、16^0、16^{-1} 為各個數字的加權值
最高有效位數（MSD）	在加權表示式中最左邊的數字 5，其加權值（16^1）最高，所以稱為 MSD。
最低有效位數（LSD）	在加權表示式中最右邊的數字 8，其加權值（16^{-1}）最低，所以稱為 LSD。

5－2 數字表示法之互換

目前大部分的數位電路都是使用二進位、八進位或十六進位表示法，而人類習慣使用十進位表示法，所以介紹各種數字表示法之間的互換。

一 二、八、十六進位轉換成十進位

將二、八、十六進位轉換成十進位的步驟如下：

(一) 列出二、八、十六進位數的加權表示式。

(二) 計算出總和，即為該數之十進位值。

5－2 數字表示法之互換

例題 5-1　二進位轉換成十進位

將二進位數$1001.01_{(2)}$轉換成十進位數。

解

$1001.01_{(2)}$

$= 1 \times 2^3 + 0 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 + 0 \times 2^{-1} + 1 \times 2^{-2}$

$= 8 + 1 + 0.25$

$= 9.25_{(10)}$

演練 1

將二進位數$11010.101_{(2)}$轉換成十進位數。

5－2 數字表示法之互換

例題 5-2　八進位轉換成十進位

將八進位數153.2$_{(8)}$ 轉換成十進位數。

解

$153.2_{(8)}$

$= 1 \times 8^2 + 5 \times 8^1 + 3 \times 8^0 + 2 \times 8^{-1}$

$= 64 + 40 + 3 + 0.25$

$= 107.25_{(10)}$

演練 2

將八進位數75.4$_{(8)}$ 轉換成十進位數。

5－2 數字表示法之互換

例題 5-3　　十六進位轉換成十進位

將十六進位數$5E.8_{(16)}$ 轉換成十進位數。

解

$5E.8_{(16)}$

$= 5 \times 16^1 + 14 \times 16^0 + 8 \times 16^{-1}$

$= 80 + 14 + 0.5$

$= 94.5_{(10)}$

演練 3

將十六進位數$12F.8_{(16)}$ 轉換成十進位數。

5-2 數字表示法之互換

二 十進位轉換成二、八、十六進位

將十進位轉換成r 進位（r 為二、八或十六進位）時，整數部分與小數部分的轉換方法不同，其轉換的步驟如表5-6 所示。

表 5-6 十進位轉換成 r 進位（r 為二、八或十六進位）的步驟

項目	整數部分 ⇒ 用**連除法**	小數部分 ⇒ 用**連乘法**
步驟	1. 將整數部分除以 r，商數寫在下方，餘數寫在右方，此餘數即為最低有效位數（LSD）。 2. 再將商數繼續除以 r，直到商數等於 0 為止，此餘數即為最高有效位數（MSD）。 3. 最後**由下而上**寫出所有的餘數，即為 r 進位數的整數部分。	1. 將小數部分乘以 r，此時乘積的整數部分即為最高有效位數（MSD）。 2. 再將乘積的小數部分繼續乘以 r，直到乘積的小數部分等於 0 為止（若小數部分無法為 0，則取足夠的小數位數），此時乘積的整數部分即為最低有效位數（LSD）。 3. 最後**由上而下**寫出所有乘積的整數，即為 r 進位數的小數部分。

5－2 數字表示法之互換

例題 5-4　十進位轉換成二進位

將十進位數$46.6875_{(10)}$ 轉換成二進位數。

解

1. 整數部分⇒ 用連除2 法

2	46	餘數
2	23	0
2	11	1
2	5	1
2	2	1
2	1	0
	0	1

2. 小數部分⇒ 用連乘2 法

整數部分	
	0.6875
×	2
1	.3750
×	2
0	.750
×	2
1	.50
×	2
1	.0

所以 $46.6875_{(10)} = 101110.1011_{(2)}$

5－2 數字表示法之互換

演練 4

將十進位數$58.375_{(10)}$轉換成二進位數。

5－2 數字表示法之互換

例題 5-5　十進位轉換成八進位

將十進位數$46.6875_{(10)}$ 轉換成八進位數。

解

1. 整數部分⇒ 用連除8 法

8 | 46 — 餘數
8 | 5 — 6
0 — 5

2. 小數部分⇒ 用連乘8 法

0 .6875
整數部分 × 8
5 .5000
× 8
4 .0

所以 $46.6875_{(10)} = 56.54_{(8)}$

演練 5

將十進位數$58.375_{(10)}$ 轉換成八進位數。

5-2 數字表示法之互換

例題 5-6　十進位轉換成十六進位

將十進位數$46.6875_{(10)}$ 轉換成十六進位數。

解

1. 整數部分⇒ 用連除16 法

$$
\begin{array}{r|l}
16 & 46 \text{ — 餘數} \\ \hline
16 & 2 \text{ — } 14\ (E) \uparrow \\ \hline
 & 0 \text{ — } 2
\end{array}
$$

2. 小數部分⇒ 用連乘16 法

$$
\begin{array}{lr}
 & 0.6875 \\
\text{整數部分} \times & 16 \\ \hline
\downarrow & (B)11.0000
\end{array}
$$

所以 $46.6875_{(10)} = 2E.B_{(16)}$

演練 6

將十進位數$58.375_{(10)}$ 轉換成十六進位數。

5-2 數字表示法之互換

二、八、十六進位之互換

(一) 二進位與八進位的互換

因為23 = 8，所以3 個位元的二進位可以表示8 種狀態，因此3 個位元的二進位數可對應一個八進位數，所以二進位與八進位互換的步驟如表5-7所示。

表 5-7 二進位與八進位互換的步驟

項目	二進位 ⇒ 八進位	八進位 ⇒ 二進位
步驟	1. 以小數點為分界點，往左及往右以 **3 個位元**為一組，若不足 3 個位元則補 0。 2. 再將每組二進位數，轉換成等值的八進位數字。	1. 將八進位的每一個數字轉換成 **3 個位元**的二進位數。 2. 所得結果整數部分最左邊的 0，以及小數部分最右邊的 0 可以省略。

5-2 數字表示法之互換

例題 5-7　二進位轉換成八進位

將二進位數$11100.10111_{(2)}$轉換成八進位數。

解

$$
\begin{array}{cccccccc}
 & & \text{補} & & & & \text{補} & \\
11100.10111_{(2)} & = & \underline{\color{red}{0}11} & \underline{100} & . & \underline{101} & \underline{11\color{red}{0}} & _{(2)} \\
 & & \downarrow & \downarrow & & \downarrow & \downarrow & \\
 & = & 3 & 4 & . & 5 & 6 & _{(8)}
\end{array}
$$

演練 7

將二進位數$1011.11101_{(2)}$轉換成八進位數。

5－2 數字表示法之互換

例題 5-8　　八進位轉換成二進位

將八進位數$34.56_{(8)}$ 轉換成二進位數。

解

$$\begin{array}{ccccc} 3 & 4 & . & 5 & 6_{(8)} \\ \downarrow & \downarrow & & \downarrow & \downarrow \end{array}$$

$$= \underline{011}\ \underline{100}\,.\,\underline{101}\ \underline{110}_{(2)}$$

$$= 11100.10111_{(2)}$$

演練 8

將八進位數$71.26_{(8)}$ 轉換成二進位數。

5－2 數字表示法之互換

（二）二進位與十六進位的互換

二進位與十六進位互換的方法類似二進位與八進位的互換，因為2^4 =16，所以4 個位元的二進位可以表示16 種狀態，因此4 個位元的二進位數可對應一個十六進位數，所以二進位與十六進位互換的步驟如表5-8 所示。

▼ 表 5-8　二進位與十六進位互換的步驟

項目	二進位 ⇒ 十六進位	十六進位 ⇒ 二進位
步驟	1. 以小數點為分界點，往左及往右以 **4 個位元**為一組，若不足 4 個位元則補 0。 2. 再將每組二進位數，轉換成等值的十六進位數字。	1. 將十六進位的每一個數字轉換成 **4 個位元**的二進位數。 2. 所得結果整數部分最左邊的 0，以及小數部分最右邊的 0 可以省略。

5-2 數字表示法之互換

例題 5-9 二進位轉換成十六進位

將二進位數$11100.10111_{(2)}$轉換成十六進位數。

解

$$11100.10111_{(2)} = \underset{\downarrow}{\underline{\overset{\text{補}}{0001}}}\ \underset{\downarrow}{\underline{1100}}\ .\ \underset{\downarrow}{\underline{1011}}\ \underset{\downarrow}{\underline{\overset{\text{補}}{1000}}}_{(2)}$$

$$= 1\ \ C\ .\ B\ \ 8_{(16)}$$

演練 9

將二進位數$11101.10101_{(2)}$轉換成十六進位數。

5－2 數字表示法之互換

例題 5-10　十六進位轉換成二進位

將十六進位數1 C . B $8_{(16)}$ 轉換成二進位數。

解

$$
\begin{array}{ccccccc}
 & 1 & C & . & B & 8_{(16)} \\
 & \downarrow & \downarrow & & \downarrow & \downarrow \\
= & \underline{0001} & \underline{1100} & . & \underline{1011} & \underline{1000}_{(2)}
\end{array}
$$

$$= 11100.10111_{(2)}$$

演練 10

將十六進位數2F. $A4_{(16)}$ 轉換成二進位數。

5－2 數字表示法之互換

（三）八進位與十六進位的互換

八進位與十六進位互換的方法為透過二進位，如表5-9 所示。

表 5-9　八進位與十六進位互換的步驟

項目	八進位 ⇒ 十六進位	十六進位 ⇒ 八進位
步驟	1. 先將八進位轉換成二進位。 2. 再將二進位轉換成十六進位。	1. 先將十六進位轉換成二進位。 2. 再將二進位轉換成八進位。

5－2 數字表示法之互換

例題 5-11　八進位轉換成十六進位

將八進位數$34.56_{(8)}$轉換成十六進位數。

解

八進位→二進位→十六進位

$$
\begin{array}{ccccccc}
 & 3 & 4 & . & 5 & 6_{(8)} \\
 & \downarrow & \downarrow & & \downarrow & \downarrow \\
= & \underline{011} & \underline{100} & . & \underline{101} & \underline{110}_{(2)} \\
 & 補 & & & & 補 \\
= & \underline{0001} & \underline{1100} & . & \underline{1011} & \underline{1000}_{(2)} \\
 & \downarrow & \downarrow & & \downarrow & \downarrow \\
= & 1 & C & . & B & 8_{(16)}
\end{array}
$$

5－2 數字表示法之互換

演練 11

將八進位數$32.15_{(8)}$ 轉換成十六進位數。

5－2 數字表示法之互換

例題 5-12　十六進位轉換成八進位

將十六進位數$1C.B8_{(16)}$ 轉換成八進位數。

解

十六進位→二進位→八進位

$$
\begin{array}{rcccccc}
 & 1 & C & . & B & 8_{(16)} \\
 & \downarrow & \downarrow & & \downarrow & \downarrow \\
= & \underline{0001} & \underline{1100} & . & \underline{1011} & \underline{1000}_{(2)} \\
= & \underline{011} & \underline{100} & . & \underline{101} & \underline{110}_{(2)} \\
 & \downarrow & \downarrow & & \downarrow & \downarrow \\
= & 3 & 4 & . & 5 & 6_{(8)}
\end{array}
$$

演練 12

將十六進位數$F7.D3_{(16)}$ 轉換成八進位數。

5－3 補數

一 補　數

為了簡化數位硬體電路的結構，通常在計算機中二進位的減法運算是利用加法器來做，因為兩個二進位數相減之差等於被減數減去減數，其結果等於被減數加上負的減數（相當於減數取補數）。

即$A - B = A + (-B) = A + (B\ \text{取補數})$

所以只要先將減數***B*** 取補數，再與被減數***A*** 相加，即可完成減法運算，稱之為補數減法運算。因此本節先針對補數（complement）做詳細的說明，對於基底為r的數字系統而言，有兩種補數的表示方式，一種為$r-1$的補數，另一種為r的補數，例如二進位數字系統有1的補數與2的補數，八進位數字系統有7的補數與8的補數，其他以此類推。

5－3 補數

（一）$r-1$ 的補數

對於基底為r 的數字系統而言，其數字為$0 \sim r-1$ 之間，所以最大的正整數為$r-1$，因此兩個正整數相加之後為$r-1$，則稱此兩數互為$r-1$ 的補數，以$(r-1)$'s 表示。表5-10 所示為r 進位數字系統之$r-1$ 的補數求法。

5－3 補數

表 5-10　$r-1$ 的補數求法

r進位	$r-1$ 的補數	數值
二進位	1 的補數（1's）	1. 將每個數字用 1 去減。 2. 快速解法：將 0 → 1，1 → 0。
八進位	7 的補數（7's）	將每個數字用 7 去減
十進位	9 的補數（9's）	將每個數字用 9 去減
十六進位	15 的補數（15's）	將每個數字用 F ($15_{(10)}$) 去減

5－3 補數

例題 5-13　1 的補數

求二進位數$1010_{(2)}$之1 的補數。

解

1's：將每個數字用1 去減。

$$\begin{array}{r} 1111_{(2)} \\ -\ 1010_{(2)} \\ \hline 0101_{(2)} \end{array}$$

或1's 快速解法：將0 → 1，1 → 0

$$\begin{array}{c} 1010_{(2)} \\ \downarrow \quad \text{取1's} \\ 0101_{(2)} \end{array}$$

所以 $1010_{(2)}$ 之1 的補數為$0101_{(2)}$

演練 13

求二進位數$1100_{(2)}$ 之1 的補數。

5－3 補數

例題 5-14　9 的補數

求十進位數$59.32_{(10)}$之9 的補數。

解

9's：將每個數字用9 去減。

$$
\begin{array}{r}
99.99_{(10)} \\
-\ 59.32_{(10)} \\
\hline
40.67_{(10)}
\end{array}
$$

所以$59.32_{(10)}$之9 的補數為$40.67_{(10)}$

演練 14

求八進位數$27.36_{(8)}$之7 的補數。

5-3 補數

例題 5-15　15 的補數

求十六進位數$B6.2C_{(16)}$ 之15 的補數。

解

15's：將每個數字用F（$15_{(10)}$）去減。

$$\begin{array}{r} FF.FF_{(16)} \\ -\,B6.2C_{(16)} \\ \hline 49.D3_{(16)} \end{array}$$

所以$B6.2C_{(16)}$ 之15 的補數為$49.D3_{(16)}$

演練 15

求十六進位數$D3.1E_{(16)}$ 之15 的補數。

5－3 補數

（二）r 的補數

對於基底為 r 的數字系統而言，其 r 的補數 $=(r-1$ 的補數$)+1$，以 r's 表示。表5-11 所示為 r 進位數字系統之 r 的補數求法。

▼ 表 5-11　r 的補數求法

r 進位	r 的補數	數值
二進位	2 的補數（2's）	1. **2's = 1's +1** 先求 1 的補數（將 0 → 1，1 → 0），再加 1。 2. 快速解法： 從左邊的 MSB 開始，將 0 → 1，1 → 0，直到最右邊的 1 之後的數字不變。
八進位	8 的補數（8's）	**8's =7's +1** 先求 7 的補數（將每個數字用 7 去減），再加 1。
十進位	10 的補數（10's）	**10's = 9's +1** 先求 9 的補數（將每個數字用 9 去減），再加 1。
十六進位	16 的補數（16's）	**16's = 15's +1** 先求 15 的補數（將每個數字用 F $(15_{(10)})$ 去減），再加 1。

5－3 補數

例題 5-16　2 的補數

求二進位數$1010_{(2)}$之2 的補數。

解

2's = 1's +1：先求1 的補數（將0 → 1，1 → 0），再加1。

$$
\begin{array}{r l}
1010_{(2)} & \\
\downarrow & \\
0101_{(2)} & \text{(1's)} \\
+\quad\quad 1 & \\
\hline
0110_{(2)} & \text{(2's)}
\end{array}
$$

或2's 快速解法：從左邊的MSB 開始，將0 → 1，1 → 0，直到最右邊的1 之後的數字不變。

$$
\begin{array}{l l}
1010_{(2)} & \\
\downarrow & \text{取2's} \\
0110_{(2)} &
\end{array}
$$

所以$1010_{(2)}$之2 的補數為$0110_{(2)}$

5－3 補數

演練 16

求二進位數$1100_{(2)}$ 之2 的補數。

5－3 補數

例題 5-17　10 的補數

求十進位數$59.32_{(10)}$之10 的補數。

解

10's = 9's +1：先求9 的補數（將每個數字用9 去減），再加1。

$$
\begin{array}{rl}
99.99_{(10)} & \\
-\ 59.32_{(10)} & \\
\hline
40.67_{(10)} & \color{red}{(9\text{'s})} \\
+\quad\quad 1 & \\
\hline
40.68_{(10)} & \color{red}{(10\text{'s})}
\end{array}
$$

所以$59.32_{(10)}$之10 的補數為$40.68_{(10)}$

演練 17

求八進位數$27.36_{(8)}$之8 的補數。

5－3 補數

例題 5-18　16 的補數

求十六進位數B6.2C$_{(16)}$ 之16 的補數。

解

16's = 15's +1：先求15 的補數（將每個數字用F ($15_{(10)}$) 去減），再加1。

$$
\begin{array}{rl}
FF.FF_{(16)} & \\
-\ B6.2C_{(16)} & \\
\hline
49.D3_{(16)} & (15\text{'s}) \\
+\quad\quad 1 & \\
\hline
49.D4_{(16)} & (16\text{'s})
\end{array}
$$

所以B6.2C$_{(16)}$ 之16 的補數為49.D4$_{(16)}$

演練 18

求十六進位數D3.1E$_{(16)}$ 之16 的補數。

5－3 補數

二 二進位有號數的表示法

若二進位數只能表示數值的大小，並無正負之分，則稱為二進位無號數表示法，例如4 位元的二進位無號數$0000_{(2)} \sim 1111_{(2)}$，可以表示的整數範圍為$0 \sim (2^4 - 1) = 0 \sim 15_{(10)}$。以此類推，對於一個$n$ 位元的二進位無號數而言，可以表示整數的範圍為$0 \sim (2^n - 1)$。

然而在實際的算數（例如加、減、乘、除）運算中，二進位數皆為有號數，有正負之分，而常用的二進位有號數表示法有真值表示法、**1** 的補數表示法與**2** 的補數表示法三種類型，分別說明如下：

5－3 補數

(一) 真值表示法

真值表示法是將最高有效位元(MSB)當成符號位元(sign bit),用來表示正負數,其他位元用來表示數值大小。其中符號位元0 是代表正數,符號位元1 是代表負數,因為符號位元不能相加,故真值表示法較少使用。

我們以 4 位元的二進位有號數為例,說明真值表示法。

1. 正數:符號位元0 + 數值大小。

   例如: $+3_{(10)} = 0011_{(2)}$。

2. 負數:符號位元1 + 數值大小。

   例如: $-3_{(10)} = 1011_{(2)}$。

5－3 補數

所以4位元的二進位有號數，以真值表示法可以表示整數的範圍為$-(2^3-1)\sim+(2^3-1)=-7\sim+7$，如表5-12所示，其中0有兩種表示法，故真值表示法較少使用。以此類推，對於一個***n***位元的二進位有號數而言，真值表示法可以表示整數的範圍為$\boldsymbol{-(2^{n-1}-1)\sim+(2^{n-1}-1)}$。

5－3 補數

例題 5-19　真值表示法

$+6_{(10)}$ 與 $-6_{(10)}$ 的4位元二進位真值表示法為何？

解

1. $+6_{(10)} = 0110_{(2)}$
2. $-6_{(10)} = 1110_{(2)}$

演練 19

$+7_{(10)}$ 與 $-7_{(10)}$ 的4位元二進位真值表示法為何？

5－3 補數

(二) 1 的補數表示法

1 的補數表示法簡稱為「1's 表示法」，它也是將最高有效位元（MSB）當成符號位元，用來表示正負數，其他位元用來表示數值。其中符號位元0是代表正數，符號位元1 是代表負數。

我們以 4 位元的二進位有號數為例，說明1 的補數表示法。

1. 正數：符號位元0 + 數值大小（與真值表示法的正數相同）。

例如： $+3_{(10)} = 0011_{(2)}$。

5－3 補數

2. 負數：符號位元1 + 數值。

即正數取 1 的補數，其方法為將正數的0 → 1，1 → 0。

例如：$-3_{(10)}$ 的4 位元二進位數1 的補數表示法為

$$\because +3_{(10)} = 0011_{(2)}$$

$$\downarrow \quad \text{取 1's}$$

$$\therefore -3_{(10)} = 1100_{(2)}$$

所以4 位元的二進位有號數，若負數以1 的補數表示時，則可以表示整數的範圍為$-(2^3-1) \sim +(2^3-1) = -7 \sim +7$，如表5-12 所示，其中0 也有兩種表示法，故1 的補數表示法較少使用。以此類推，對於一個n 位元的二進位有號數而言，若負數以1 的補數表示時，則可以表示整數的範圍為$\mathbf{-(2^{n-1}-1) \sim +(2^{n-1}-1)}$。

5－3 補數

例題 5-20　1 的補數表示法

－$6_{(10)}$ 的4 位元二進位1 的補數表示法為何？

解

$$\because +6_{(10)} = 0110_{(2)}$$

$$\downarrow \quad \text{取 1's}$$

$$\therefore -6_{(10)} = 1001_{(2)}$$

演練 20

－$7_{(10)}$ 的4 位元二進位1 的補數表示法為何？

5－3 補數

例題 5-21　1 的補數表示法

在4 位元二進位1 的補數表示法中，$1110_{(2)}$ 的十進位值為何？

解　　符號位元 ＝ 1，表示為負數→ $1110_{(2)}$

$$\downarrow \text{取 } 1\text{'s}$$

$$= -0001_{(2)}$$

$$= -1_{(10)}$$

$$\therefore 1110_{(2)} = -1_{(10)}$$

演練 21

在4 位元二進位1 的補數表示法中，$1101_{(2)}$ 的十進位值為何？

5－3 補數

（三）2 的補數表示法

2 的補數表示法簡稱為2’s 表示法，它也是將最高有效位元（MSB）當成符號位元，用來表示正負數，其他位元用來表示數值。其中符號位元0 是代表正數，符號位元1 是代表負數。

我們以 4 位元的二進位有號數為例，說明2 的補數表示法。

1. 正數：符號位元0 ＋ 數值大小（與真值表示法的正數相同）

   例如：$+3_{(10)} = 0011_{(2)}$。

2. 負數：符號位元1 ＋ 數值。

   即正數取 2 的補數 ＝ 正數取1 的補數 ＋ 1

5－3 補數

例如： $-3_{(10)}$ 的4 位元二進位數2 的補數表示法為

$$\because +3_{(10)} = 0011_{(2)}$$

$$\downarrow$$

$$\begin{array}{rl} 1100_{(2)} & (1's) \\ +\quad\;\; 1 & \\ \hline 1101_{(2)} & (2's) \end{array}$$

$$\therefore -3_{(10)} = 1101_{(2)}$$

5－3 補數

所以 4 位元的二進位有號數，若負數以2 的補數表示時，則可以表示整數的範圍為$-(2^3) \sim +(2^3-1) = -8 \sim +7$，如表5-12 所示，其中0 只有一種表示法，即$0_{(10)} = 0000_{(2)}$，且2 的補數表示法可以表示的整數範圍較大，因此在一般的數位電路或計算機中，均使用2 的補數表示法。以此類推，對於一個n 位元的二進位有號數而言，若負數以2 的補數表示時，則可以表示整數的範圍為$\mathbf{-2^{n-1} \sim +(2^{n-1}-1)}$。

5-3 補數

▼ 表 5-12　4 位元的二進位有號數表示整數的範圍

二進位數		對應的十進位數			說明
符號位元	數值大小	真值表示法	負數以 1's 表示法	負數以 2's 表示法	
0	000	+0	+0	+0	正數表示法相同
0	001	+1	+1	+1	
0	010	+2	+2	+2	
0	011	+3	+3	+3	
0	100	+4	+4	+4	
0	101	+5	+5	+5	
0	110	+6	+6	+6	
0	111	+7	+7	+7	

5－3 補數

1 000	–0	–7	–8	負數表示法不同
1 001	–1	–6	–7	
1 010	–2	–5	–6	
1 011	–3	–4	–5	
1 100	–4	–3	–4	
1 101	–5	–2	–3	
1 110	–6	–1	–2	
1 111	–7	–0	–1	
整數範圍	$-(2^3-1) \sim +(2^3-1)$ $= -7 \sim +7$	$-(2^3-1) \sim +(2^3-1)$ $= -7 \sim +7$	$-2^3 \sim +(2^3-1)$ $= -8 \sim +7$	

5－3 補數

例題 5-22　2 的補數表示法

$-6_{(10)}$ 的4 位元二進位2 的補數表示法為何？

解

$$\because\ +6_{(10)} = 0110_{(2)}$$

$$\downarrow$$

$$\begin{array}{rl} 1001_{(2)} & (1\text{'s}) \\ +\qquad 1 & \\ \hline 1010_{(2)} & (2\text{'s}) \end{array}$$

$$\therefore\ -6_{(10)} = 1010_{(2)}$$

演練 22

$-7_{(10)}$ 的4 位元二進位2 的補數表示法為何？

5－3 補數

例題 5-23　2 的補數表示法

在4 位元二進位2 的補數表示法中，$1110_{(2)}$ 的十進位值為何？

解　符號位元 = 1，表示為負數→ $1110_{(2)}$

$$\downarrow \quad \text{取 } 2\text{'s}$$

$$= -0010_{(2)}$$

$$= -2_{(10)}$$

$$\therefore\ 1110_{(2)} = -2_{(10)}$$

演練 23

在4 位元二進位2 的補數表示法中，$1111_{(2)}$ 的十進位值為何？

5-3 補數

三 二進位加法

二進位加法與十進位加法類似，只是兩個二進位數字相加，若和大於或等於2 即進位。我們以兩個1 位元的二進位數相加，以及兩個4 位元的二進位無號數相加為例，說明二進位的加法運算，如表5-13 所示。

▼ 表 5-13　二進位的加法運算

兩個 1 位元的二進位數相加				兩個 4 位元的二進位數相加
0	0	1	1	1 ←進位
				1101
+0	+1	+0	+1	+1100
0	1	1	進位→ 10	進位→ 11001

5－3 補數

例題 5-24　二進位無號數加法

兩個4 位元二進位無號數$1011_{(2)}$ + $1001_{(2)}$，其結果為何？

解

$$
\begin{array}{rr}
 & 11 \leftarrow 進位 \\
 & 1011_{(2)} \\
 & +\ 1001_{(2)} \\
\hline
進位 \rightarrow & 10100_{(2)}
\end{array}
$$

所以 $1011_{(2)} + 1001_{(2)} = 10100_{(2)}$

演練 24

兩個4 位元二進位無號數$1010_{(2)}$ + $1100_{(2)}$ 結果為何？

5－3 補數

四 二進位減法

通常在數位電路或計算機中，二進位的減法運算是利用加法器來做，以達到簡化電路的目的，因為兩個二進位數相減等於被減數減掉減數，其結果等於被減數加上負的減數（相當於減數取補數），所以只要先將減數取補數，再與被減數相加，即可完成減法運算，稱之為補數減法運算，補數減法運算有1 的補數減法運算與2 的補數減法運算，所以我們針對這兩種減法運算做說明。

5－3 補數

（一）1 的補數減法運算

只要將減數取1 的補數，再與被減數相加，即可完成1 的補數減法運算，若有進位產生時，則需將進位與和的最低有效位元（LSB）相加，以獲得正確的答案，稱之為端迴進位（end around carry；簡稱為EAC），在數位電路中，由於1 的補數減法運算電路較複雜，故較少使用。

因為兩數相減之差 ＝被減數－減數

$$\begin{aligned} &= A - B \\ &= A + (-B) \\ &= A + (B \text{ 取1 的補數}) \\ &= A + \overline{B} \end{aligned}$$

所以 1 的補數減法運算之步驟，如表5-14 所示。

5－3 補數

▼ 表 5-14　1 的補數減法運算之步驟

<table>
<tr><th>項目</th><th>$A-B=$ 大－小＝正數時</th><th>$A-B=$ 小－大＝負數時</th></tr>
<tr><td rowspan="2">步驟</td><td colspan="2">1. 先將減數 B 取 1 的補數。
2. 再與被減數 A 相加。</td></tr>
<tr><td>3. 若有進位產生時，則需將進位與和的最低有效位元（LSB）相加，稱之為端迴進位（EAC），且其結果為正數。</td><td>3. 若無進位產生時，其結果即為解答，且為負數。可將結果取 1 的補數之後，再加上負號，即為十進位答案。</td></tr>
</table>

5－3 補數

例題 5-25　1 的補數減法（大－小）

試以4 位元二進位1 的補數執行減法運算$6_{(10)} - 4_{(10)} = 0110_{(2)} - 0100_{(2)}$，其結果為何？

解

1. 先將減數$0100_{(2)}$ 取1 的補數=$1011_{(2)}$
2. 再與被減數$0110_{(2)}$ 相加
3. 若有進位產生時，則需將進位與和的最低有效位元（LSB）相加，且其結果為正數。

5－3 補數

$$
\begin{array}{r}
6_{(10)} \\
-\ 4_{(10)} \\
\hline
2_{(10)}
\end{array}
\qquad
\begin{array}{r}
0110_{(2)} \\
-\ 0100_{(2)} \\
\hline
\end{array}
\quad \Rightarrow \text{取}1\text{'s} \quad
\begin{array}{r}
0110_{(2)} \\
+\ \ 1011_{(2)} \\
\hline
\text{端迴進位}\ \boxed{1}0001_{(2)} \\
+\qquad\quad 1 \\
\hline
0010_{(2)}
\end{array}
\ \leftarrow \text{解答}
$$

符號位元＝0，表示結果為正數

$\therefore 6_{(10)} - 4_{(10)} = 0110_{(2)} - 0100_{(2)} = 0010_{(2)} = 2_{(10)}$

演練 25

試以4 位元二進位1 的補數執行減法運算$7_{(10)} - 3_{(10)} = 0111_{(2)} - 0011_{(2)}$，其結果為何？

5－3 補數

例題 5-26　1 的補數減法（小－大）

試以4 位元二進位1 的補數執行減法運算$4_{(10)} - 6_{(10)} = 0100_{(2)} - 0110_{(2)}$，其結果為何？

解

1. 先將減數$0110_{(2)}$ 取1 的補數=$1001_{(2)}$
2. 再與被減數$0100_{(2)}$ 相加
3. 若無進位產生時，其結果即為解答，且為負數。可將結果取1 的補數之後，再加上負號，即為答案。

5－3 補數

$$\begin{array}{r} 4_{(10)} \\ -\ 6_{(10)} \\ \hline 2_{(10)} \end{array} \qquad \begin{array}{r} 0100_{(2)} \\ -\ 0110_{(2)} \\ \hline \end{array} \Rightarrow \text{取1's} \qquad \begin{array}{r} 0100_{(2)} \\ +\ 1001_{(2)} \\ \hline 1101_{(2)} \end{array}$$

符號位元＝1，表示結果為負數→$1101_{(2)}$ ←解答

↓ 取1's

$= -0010_{(2)} = -2_{(10)}$

$\therefore 4_{(10)} - 6_{(10)} = 0100_{(2)} - 0110_{(2)} = 1101_{(2)} = -2_{(10)}$

演練 26

試以4 位元二進位1 的補數執行減法運算$3_{(10)} - 7_{(10)} = 0011_{(2)} - 0111_{(2)}$，其結果為何？

5－3 補數

（二）2 的補數減法運算

只要將減數取2 的補數，再與被減數相加，即可完成2 的補數減法運算，若有進位產生時，則直接捨進位後其結果即為解答，由於2 的補數減法運算電路較簡單，所以目前在數位電路與計算機中，大多數皆採用2 的補數減法運算。

因為兩數相減之差 ＝ 被減數－ 減數

$$= A - B$$

$$= A + (-B)$$

$$= A + (B\text{ 取2 的補數})$$

$$= A + (B\text{ 取1 的補數} + 1)$$

$$= A + (\overline{B} + 1)$$

1. 2 的補數減法運算之步驟，如表5-15 所示。

5-3 補數

▼ 表 5-15　2 的補數減法運算之步驟

<table>
<tr><th>項目</th><th>$A-B=$ 大 − 小 = 正數時</th><th>$A-B=$ 小 − 大 = 負數時</th></tr>
<tr><td rowspan="3">步驟</td><td colspan="2">1. 先將減數 B 取 2 的補數。
2. 再與被減數 A 相加。</td></tr>
<tr><td>3. 若有進位產生時，則直接捨進位後其結果即為解答，且為正數。</td><td>3. 若無進位產生時，其結果即為解答，且為負數。可將結果取 2 的補數之後，再加上負號，即為十進位答案。</td></tr>
<tr><td colspan="2">4. 運算的結果不可有溢位（overflow），即所得的結果不可以超過其位元數所能表示的範圍，故需判斷是否發生溢位？</td></tr>
</table>

5－3 補數

例題 5-27　2 的補數減法（大－小）

試以4 位元二進位2 的補數執行減法運算$6_{(10)} - 4_{(10)} = 0110_{(2)} - 0100_{(2)}$，其結果為何？

解

1. 先將減數$0100_{(2)}$ 取2 的補數=$1100_{(2)}$
2. 再與被減數$0110_{(2)}$ 相加
3. 若有進位產生時，則直接捨進位後其結果即為解答，且為正數。
4. 判斷是否發生溢位？因為大－小 = 正數－ 正數 = 正數+ 負數，所以不會發生溢位。

5－3 補數

$$\begin{array}{r} 6_{(10)} \\ -\ 4_{(10)} \\ \hline 2_{(10)} \end{array} \qquad \begin{array}{r} 0110_{(2)} \\ -\ 0100_{(2)} \\ \hline \end{array} \quad \Rightarrow \text{取2's} \quad \begin{array}{r} 0110_{(2)} \\ +\ 1100_{(2)} \\ \hline \end{array}$$

捨進位 $\boxed{1}0010_{(2)}$ ←解答

└符號位元＝0，表示結果為正數

$$\therefore 6_{(10)} - 4_{(10)} = 0110_{(2)} - 0100_{(2)} = 0010_{(2)} = 2_{(10)}$$

演練 27

試以4 位元二進位2 的補數執行減法運算$7_{(10)} - 3_{(10)} = 0111_{(2)} - 0011_{(2)}$，其結果為何？

5－3 補數

例題 5-28　2 的補數減法（小－大）

試以4 位元二進位2 的補數執行減法運算$4_{(10)} - 6_{(10)} = 0100_{(2)} - 0110_{(2)}$，其結果為何？

解

1. 先將減數$0110_{(2)}$ 取2 的補數＝$1010_{(2)}$
2. 再與被減數$0100_{(2)}$ 相加
3. 若無進位產生時，其結果即為解答，且為負數。可將結果取2 的補數之後，再加上負號，即為答案。
4. 判斷是否發生溢位？因為小– 大= 正數– 正數 = 正數+ 負數，所以不會發生溢位。

5－3 補數

$$\begin{array}{r} 4_{(10)} \\ -\ 6_{(10)} \\ \hline -\ 2_{(10)} \end{array} \qquad \begin{array}{r} 0100_{(2)} \\ -\ 0110_{(2)} \\ \hline \end{array} \Rightarrow \text{取2's} \qquad \begin{array}{r} 0100_{(2)} \\ +\ 1010_{(2)} \\ \hline 1110_{(2)} \end{array}$$

符號位元=1，表示結果為負數 → $1110_{(2)}$ ←解答

↓ 取2's

$= -0010_{(2)} = -2_{(10)}$

$\therefore 4_{(10)} - 6_{(10)} = 0100_{(2)} - 0110_{(2)} = 1110_{(2)} = -2_{(10)}$

演練 28

試以4 位元二進位2 的補數執行減法運算$3_{(10)} - 7_{(10)} = 0011_{(2)} - 0111_{(2)}$，其結果為何？

5－3 補數

2. 溢　位

當計算機執行算術運算時，若所得的結果超過其位元數所能表示的範圍，則稱為溢位（**overflow**）。例如*n* 位元的二進位有號數之2 的補數表示法，其可以表示的整數範圍為$-2^{n-1} \sim +(2^{n-1}-1)$，若運算結果超過此範圍即為溢位，此時運算的結果發生錯誤，其答案是不正確的。

另外亦可藉由溢位旗標（overflow flag；簡稱OF）來檢查運算結果是否溢位。若有*n* 位元二進位2 的補數減法運算時，則溢位旗標為

5－3 補數

$$OF = C_n \oplus C_{n-1}$$

5-2 式

其中 C_n ：表示最高有效位元（MSB）的進位。

C_{n-1} ：表示次高有效位元（MSB 的右邊位元）的進位。

$\oplus$ ：表示 C_n 與 C_{n-1} 執行互斥或（XOR）運算。

5－3 補數

當C_n與C_{n-1}相同時（同時為0或同時為1），則溢位旗標$OF = 0$，表示運算的結果無發生溢位，其答案是正確的；反之，當C_n與C_{n-1}不同時（一個為0，另一個為1），則溢位旗標$OF = 1$，表示運算的結果發生溢位，其答案是錯誤的。

例如4位元的二進位有號數之2的補數表示法，其可以表示的整數範圍為$-2^{4-1} \sim +(2^{4-1} - 1) = -8 \sim +7$，若正數減負數（即正數加正數），或負數減正數（即負數加負數）時，其結果就有可能超過此範圍而產生溢位，舉例說明如下。

5－3 補數

例題 5-29　　溢位的判斷

試以4 位元二進位2 的補數執行減法運算$-2_{(10)} - 7_{(10)} = 1110_{(2)} - 0111_{(2)}$，其結果為何？

解

1. 先將減數$0111_{(2)}$ 取2 的補數 = $1001_{(2)}$
2. 再與被減數$1110_{(2)}$ 相加
3. 若有進位產生時，則直接捨進位後其結果即為解答。
4. 判斷是否溢位？因為負數– 正數 = 負數+ 負數，所以有可能發生溢位。

 而判斷是否發生溢位的方法有下列兩種：

5 - 3 補數

（1）溢位旗標OF：

$C_{n-1} = 0$

$C_n = 1$

$$\begin{array}{r} -2_{(10)} \\ -7_{(10)} \\ \hline -9_{(10)} \end{array} \qquad \begin{array}{r} 1110_{(2)} \\ -\ 0111_{(2)} \\ \hline \end{array} \Rightarrow \text{取2's} \qquad \begin{array}{r} 1110_{(2)} \\ +\ 1001_{(2)} \\ \hline \text{捨進位}\ \boxed{1}0111_{(2)} \end{array}$$

←錯誤的解答

符號位元 = 0，表示結果為正數

由於$OF = C_n \oplus C_{n-1} = 1 \oplus 0 = 1$，表示運算的結果發生溢位，所以解答是錯誤的。

5－3 補數

(2) **4** 位元二進位**2** 的補數表示的整數範圍為**–8** ～ **+7**。

因為$-2_{(10)}-7_{(10)}=1110_{(2)}-0111_{(2)}=-9_{(10)}$，已超過其整數範圍–8 ～ +7，所以發生溢位，因此解答是錯誤的。

演練 29

試以4 位元二進位2 的補數執行減法運算$3_{(10)}-(-6_{(10)})=0011_{(2)}-1010_{(2)}$，其結果為何？

5-4 二進碼十進數及字元編碼

一般而言，數字碼可以區分為加權碼與非加權碼，而十進位、二進位、八進位與十六進位等常用的加權碼已經在前面幾節介紹過了，還有一個加權碼— BCD 將在本節說明。另外尚有一些常用的非加權碼，可以用來表示文字（例如A、B、C、a、b、c、…）、數字（例如1、2、3、4、…）與符號（例如+、－、<、>、?、$、…）等資料，這些非加權碼無法做算數運算，亦將在本節說明。

5-4 二進碼十進數及字元編碼

一 二進碼十進數（BCD）

人類最常使用的數字系統是十進位，而數位電路與電腦通常只能處理二進位的信號，因此將十進位數轉換為二進位碼，即二進碼十進數（binary-coded decimal；簡稱BCD），BCD 是以4 位元的二進位碼（0000 ～ 1001）來表示十進位數（0 ～ 9）。表5-16 是十進位數、BCD 與二進位碼之間的對照表。

表 5-16　十進位數、BCD 與二進位碼之間的對照表

十進位數	BCD	二進位碼
0	0000	0000
1	0001	0001
2	0010	0010
3	0011	0011
4	0100	0100
5	0101	0101
6	0110	0110
7	0111	0111
8	1000	1000
9	1001	1001
10	0001 0000	1010
11	0001 0001	1011
12	0001 0010	1100
13	0001 0011	1101
14	0001 0100	1110
15	0001 0101	1111

5-4 二進碼十進數及字元編碼

（一）BCD 的特性

1. 由於BCD 是以4 位元的二進位碼來表示十進位數，所以既適合人類閱讀，又適合電腦運算（詳細運算請參考6-3 節）。
2. 由於BCD 是加權碼，其加權值分別為2^3、2^2、2^1、2^0，即8、4、2、1，所以又稱為**8421** 碼。

（二）十進位與BCD 的互換

十進位與BCD 互換的步驟，如表5-17 所示。

5-4 二進碼十進數及字元編碼

▼ 表 5-17 十進位與 BCD 互換的步驟

項目	十進位 ⇒ BCD	BCD ⇒ 十進位
步驟	將十進位的每一個數字轉換成 **4 個位元**的二進位數。	1. 以小數點為分界點，往左及往右以 **4 個位元**為一組，若不足 4 個位元則補 0。 2. 再將每組二進位數，轉換成等值的十進位數字。

5-4 二進碼十進數及字元編碼

例題 5-30　十進位轉換成BCD

將十進位數$19_{(10)}$ 以BCD 表示之。

解

$$
\begin{array}{ccc}
 & 1 & 9_{(10)} \\
 & \downarrow & \downarrow \\
= & \underline{0001} & \underline{1001}_{(BCD)}
\end{array}
$$

所以 $19_{(10)} = 00011001_{(BCD)}$

演練 30

將十進位數$58_{(10)}$ 以BCD 表示之。

5-4 二進碼十進數及字元編碼

例題 5-31　**BCD** 轉換成十進位

將$1100011.010_{(BCD)}$轉換成十進位數。

解

$$
\begin{array}{ccccccc}
 & \text{補} & & & & \text{補} & \\
 & \underline{0110} & \underline{0011} & . & \underline{0100} & _{(BCD)} \\
 & \downarrow & \downarrow & & \downarrow & \\
= & 6 & 3 & . & 4_{(10)} &
\end{array}
$$

所以 $1100011.010_{(BCD)} = 63.4_{(10)}$

演練 31

將$1111001.001_{(BCD)}$轉換成十進位數。

5-4 二進碼十進數及字元編碼

二 加三碼

加三碼（excess－3 code）是將BCD 加上$3_{(10)}$（$=0011_{(2)}$）所形成的編碼方式，表5-18 是十進位數、BCD 與加三碼之間的對照表。

表 5-18　十進位數、BCD 與加三碼之間的對照表

十進位數	BCD	加三碼
0	0000	0011
1	0001	0100
2	0010	0101
3	0011	0110
4	0100	0111
5	0101	1000
6	0110	1001
7	0111	1010
8	1000	1011
9	1001	1100

互補

5-4 二進碼十進數及字元編碼

（一）加三碼的特性

1. 加三碼具有自補特性，即0 的加三碼（$0011_{(Excess-3)}$）取1 的補數後與9 的加三碼（$1100_{(Excess-3)}$）相同，所以0 與9 的加三碼互為補數，1 與8 的加三碼互為補數等，其他以此類推。
2. 由於每一組加三碼至少都包含一個1，所以具有偵錯的能力，但是加三碼無法以加權值的方式表示，所以加三碼並非加權碼。

（二）十進位與加三碼的互換

十進位與加三碼互換的步驟，如表5-19 所示。

5-4 二進碼十進數及字元編碼

▼ 表 5-19　十進位與加三碼互換的步驟

項目	十進位 ⇒ 加三碼	加三碼 ⇒ 十進位
步驟	將十進位的每一個數字**先加上 3**，再轉換成 **4 個位元**的二進位數。	1. 以小數點為分界點，往左及往右以 **4 個位元**為一組，若不足 4 個位元則補 0。 2. 再將每組二進位數，轉換成等值的十進位數字。 3. 再將十進位的每一個數字**減 3**。

5-4 二進碼十進數及字元編碼

例題 5-32　十進位轉換成加三碼

將十進位數$19_{(10)}$ 以加三碼表示之。

解

$$
\begin{array}{rr}
1 & 9_{(10)} \\
+\ 3 & +\ 3 \\
\hline
4 & 12 \\
\downarrow & \downarrow \\
=\ \underline{0100} & \underline{1100}_{\ (Excess-3)}
\end{array}
$$

所以 $19_{(10)} = 01001100_{(Excess-3)}$

演練 32

將十進位數$58_{(10)}$ 以加三碼表示之。

5-4 二進碼十進數及字元編碼

例題 5-33　加三碼轉換成十進位

將加三碼$10100101_{(\text{Excess}-3)}$ 轉換成十進位數。

解

$$
\begin{array}{cccc}
 & \underline{1010} & & \underline{0101}_{(\text{Excess}-3)} \\
 & \downarrow & & \downarrow \\
= & 10 & & 5 \\
 & \underline{-\ \ 3} & & \underline{-\ \ 3} \\
= & 7 & & 2_{(10)}
\end{array}
$$

所以 $10100101_{(\text{Excess}-3)} = 72_{(10)}$

演練 33

將加三碼$10110100_{(\text{Excess}-3)}$ 轉換成十進位數。

5-4 二進碼十進數及字元編碼

三 格雷碼

格雷碼（gray code）是相鄰兩數碼之間僅有一個位元不同的編碼方式，所以是一種最小變化碼，表5-20是二進位碼與格雷碼之間的對照表。

▼ 表 5-20　二進位碼與格雷碼之間的對照表

二進位碼	格雷碼
0000	0 000
0001	0 001
0010	0 011
0011	0 010
0100	0 110
0101	0 111
0110	0 101
0111	0 100
1000	1 100
1001	1 101
1010	1 111
1011	1 110
1100	1 010
1101	1 011
1110	1 001
1111	1 000

如鏡子般反射

5-4 二進碼十進數及字元編碼

（一）格雷碼的特性

1. 由於格雷碼的相鄰兩數碼之間僅有一個位元不同，所以一般可以使用在類比/ 數位轉換器中，以減少資料傳輸與轉換時的錯誤。

2. 格雷碼除了最高有效位元之外，其餘三個位元呈現如鏡子般的反射現象，所以又稱為反射碼（reflected code）。

（二）二進位碼與格雷碼的互換

二進位碼與格雷碼互換的步驟如表5-21 所示。若欲將其他進位（例如八進位、十進位、十六進位等）轉換為格雷碼，則先將其他進位（例如八進位、十進位、十六進位等）轉換為二進位，再將二進位轉換為格雷碼，反之亦然。

5-4 二進碼十進數及字元編碼

表 5-21　二進位碼與格雷碼互換的步驟

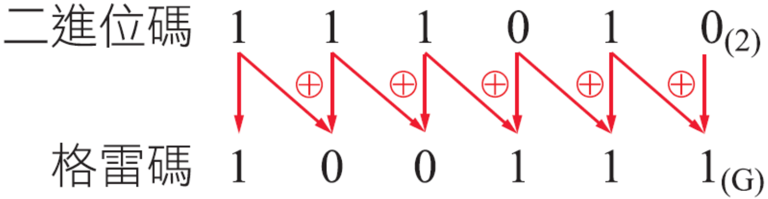
項目	二進位碼 ⇒ 格雷碼	格雷碼 ⇒ 二進位碼
步驟	1. 將二進位碼的最高有效位元作為格雷碼的最高有效位元。 2. 由二進位碼的最高有效位元開始，由左而右將兩兩相鄰的位元做互斥或（XOR）運算，即兩位元相同時，結果為 0，兩位元不同時，結果為 1，並將結果依序向右寫出來，即為格雷碼。	1. 將格雷碼的最高有效位元作為二進位碼的最高有效位元。 2. 由二進位碼的最高有效位元開始，依序與格雷碼的次高有效位元做互斥或（XOR）運算，並將結果依序向右寫出來，即為二進位碼。

5-4 二進碼十進數及字元編碼

例題 5-34　二進位轉換成格雷碼

將二進位碼$111010_{(2)}$ 轉換成格雷碼。

解



所以 $111010_{(2)} = 100111_{(G)}$

演練 34

將二進位碼$0011101_{(2)}$ 轉換成格雷碼。

5-4 二進碼十進數及字元編碼

例題 5-35　格雷碼轉換成二進位

將格雷碼$100111_{(G)}$轉換成二進位碼。

解

格雷碼	1	0	0	1	1	$1_{(G)}$
二進位碼	1	1	1	0	1	$0_{(2)}$

所以 $100111_{(G)} = 111010_{(2)}$

演練 35

將格雷碼$0110001_{(G)}$轉換成二進位碼。

5-4 二進碼十進數及字元編碼

四 美國資訊交換標準代碼（ASCII）

在數位電路與電腦中，除了需要處理二進位的信號之外，也要處理大量的文字與符號等資料，所以必須將這些文字與符號以二進位來編碼，方便不同電腦之間資料傳輸與通訊，因此美國國家標準協會（ANSI）制定了美國資訊交換標準代碼（american standard code for information interchange code；簡稱ASCII），作為不同電腦系統之間，資料傳輸與交換的標準編碼方式。

ASCII 是由7 位元的二進位碼所組成，可以表示2^7 = 128 種不同的文字、數字與符號，任何文字、數字與符號必須轉換成ASCII，才能傳送到電腦內部做處理，故ASCII 又稱為內碼。目前ASCII 廣泛地應用在電腦與輸入、輸出相關週邊設備（例如鍵盤、印表機等）的資料編碼上。

5-4 二進碼十進數及字元編碼

如表5-22 所示為ASCII 表，為了方便查表與閱讀，故將7 位元ASCII 的最左邊3 個位元當成高位元組，最右邊4 個位元當成低位元組，將高位元組做為行（column）座標，低位元組做為列（row）座標，例如英文字母"A" 的位置在第4 行、第1 列，所以其ASCII 為$41_{(16)}=1000001_{(2)}=65_{(10)}$，而英文字母"a" 的位置在第6 行、第1 列，所以其ASCII 為$61_{(16)}=1100001_{(2)}=97_{(10)}$，其他以此類推

5-4 二進碼十進數及字元編碼

▼ 表 5-22　ASCII 表

列	行 十六進位	行 二進位	高位元組（行）							
	十六進位		0	1	2	3	4	5	6	7
		二進位	000	001	010	011	100	101	110	111
低位元組（列）	0	0000	NUL	DLE	SP	0	@	P	`	p
	1	0001	SOH	DC1	!	1	A	Q	a	q
	2	0010	STX	DC2	“	2	B	R	b	r
	3	0011	ETX	DC3	#	3	C	S	c	s
	4	0100	EOT	DC4	$	4	D	T	d	t
	5	0101	ENQ	NAK	%	5	E	U	e	u
	6	0110	ACK	SYN	&	6	F	V	f	v
	7	0111	BEL	ETB	‘	7	G	W	g	w
	8	1000	BS	CAN	(	8	H	X	h	x
	9	1001	HT	EM	)	9	I	Y	i	y
	A	1010	LT	SUB	*	:	J	Z	j	z
	B	1011	VT	ESC	+	;	K	[	k	{
	C	1100	FF	FS	,	<	L	\	l	|
	D	1101	CR	GS	-	=	M	]	m	}
	E	1110	SO	RS	.	>	N	^	n	~
	F	1111	US	US	/	?	O	_	o	DEL

5-4 二進碼十進數及字元編碼

例題 5-36　**ASCII**

請寫出下列文字與數字的ASCII。

1. 英文字母N。2. 數字5。

解

1. 除了利用查表5-21 之外，亦可以推算方式寫出"N" 的ASCII，其方法如下：

因為第一個英文字母 "A" 的ASCII = $41_{(16)}$ = $1000001_{(2)}$ =$65_{(10)}$，而 "N" 為第14 個英文字母，所以其ASCII 為"A" 的ASCII 加上$13_{(10)}$（ =$D_{(16)}$），因此 "N" 的ASCII = $41_{(16)}$ + $D_{(16)}$ = $4E_{(16)}$ = $1001110_{(2)}$或 "N" 的ASCII = $65_{(10)}$ + $13_{(10)}$ = $78_{(10)}$

5-4 二進碼十進數及字元編碼

2. 查表5-21 可知數字"5" 的ASCII = $35_{(16)}$ = $0110101_{(2)}$ = $53_{(10)}$

演練 36

請寫出下列文字與符號的ASCII。

1. 英文字母X
2. 符號&