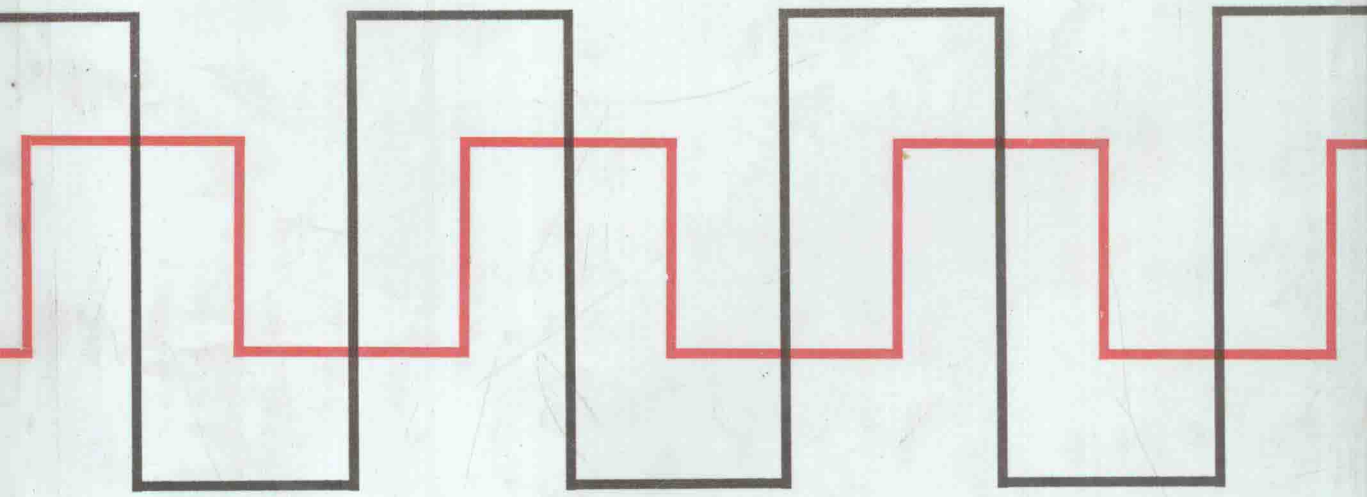
高等学校教材

数字电子技术

赵光义　赵信　编著



延边大学出版社

高等学校教材

数字电子技术

赵光义　赵　信　编著

延边大学出版社

责任编辑：申仁花
封面设计：曹　暄
责任校对：张宏飞

〔书名〕：数字电子技术
〔作者〕：赵光义　赵　信
延 边 大 学 出 版 社 出 版
（吉林省延吉市延边大学院内）
长春市朝阳彩印厂印刷
开本：787×1092 毫米 1/16 印张：19.125
字数：490 千字　印数：1－2,000 本
1998 年 3 月第 1 版　1998 年 3 月第 1 次印刷
ISBN 7－5634－0991－2/TP・3(课)
定价：30.00 元

内容提要

本书内容共分九章。包括数制与编码；逻辑代数基础；集成逻辑门；组合逻辑电路；集成触发器；时序逻辑逻辑电路；脉冲波形的产生与整形；大规模集成电路；A/D 及 D/A 的变换。本书注重理论和实际相结合；具有较多的设计实例，每章都有丰富的习题。

本书可作为高等院校的电子，通信、计算机及自动控制各专业的本科、专科和函授的教材，并可供有关的工程技术人员参考。

前　言

数字电子技术是当前发展最快的学科之一。就器件而言，已经从电子管、晶体管、小规模集成电路（SSI），发展到现在的中规模集成电路（MSI），大规模集成电路（LSI）和超大规模集成电路（VLSI）。近几年又出现了可编程逻辑器件。由于半导体技术的迅速发展，微型计算机广泛应用，所以数字电子技术在现代科学技术领域中占有很重要的地位，在各个领域中得到了广泛的应用。

全书是在《数字电路》教材的基础上编写而成的。在编写教材的过程中，我们既注意加强教材的理论性，又强调实践性，实用性及应用技术的先进性。加强了中规模集成电路的介绍及应用。并适当介绍了大规模集成电路。这样，使学生既能掌握比较系统的基础理论和先进器件应用，又能使学生具有数字逻辑设计的独立工作能力。

全书共分九章。第一章数制与编码。第二章逻辑代数基础。系统讨论逻辑函数的简化方法，为分析和设计数字电路提供数学工具。第三章集成逻辑门电路。讲述了开关电路、TTL 门和 MOS 门电路。第四章组合逻辑电路的分析设计及一些常用组合逻辑电路。第五章触发器。它是学习时序电路的基础。第六章时序电路的分析、设计方法及一些常用时序电路及中规模时序电路。第七章脉冲波形的产生与整形。脉冲波形的产生及变换方法。555 定时器的应用。第八章大规模集成电路主要介绍 ROM 和 PLA 在数字系统的应用。第九章为数/模（D/A）和模数（A/D）转换原理及其电路。在每章末都附有一定数量的习题，帮助学生加强对课程内容的理解。部分习题有一定深度，以便学生在深入掌握课程内容的基础上扩展知识。

本书第一、二、三、四、五、六章由赵光义编写，第七、八、九章由赵信编写。全书由赵光义担任主编。

在编写本书时，虽力求遵从教学规律，利于学生理解掌握本书内容，但由于编者水平有限，书中难免有不妥和错误之处，恳请读者批评、指正。

作者

1997.1

目　录

第一章　数制与码

本章介绍数字系统中常用的数制及其特点，相互转换，负数的表示方法，码的概念及常用的编码。

1.1　数制

表示“数值”大小的各种方法统称为计数制，简称数制。按着不同的进位方法，计数制分为十进制、二进制、十六进制等。每一种进制都是由一组特定的有序符号（称为数字符号或数符）及小数点组成。一种数制中所允许使用的数字符号的总数称为基数。

在任何一种数制中，任何一个数都可以具有由小数点（·）隔开的整数部分和小数部分，且有两种书写形式：位置计数法和多项式表示法。

一、十进制码（Decimal Code）

在生产劳动和日常生活中，最常用的是十进制码。它共有十个数字符号：0、1、2、3、4、5、6、7、8 和 9，把这些数字符号叫数码，十进制的编码很简单，进位规则是“逢十进一”，即每一位计满十就向高一位进一。例如十进制数 $(4348.5)_{10}$，可以把它理解为：

$(4348.5)_{10}=4\times10^3+3\times10^2+4\times10^1+8\times10^0+5\times10^{-1}$而系数 10^3、10^2、10^1、10^0、10^{-1}是根据它们的系数 4、3、4、8 以及 5 与小数点的相对位置得出来的。我们把这种根据数字符号在数中的位置来表示数的大小的方法称为位置计数法，而把数 10^3、10^2、10^1、10^0、10^{-1}叫做权，或叫位权。很显然，十进制数中各位的权是基数 10 的幂。

对于一个具有 n 位整数、m 位小数的十进制 $(N)_{10}$可记为 $(N)_{10}=(a_{n-1}a_{n-2}\cdots a_1a_0\cdot a_{-1}a_{-2}\cdots a_{-m})_{10}$，按多项式表示法可以写为

$$(N)_{10}=(a_{n-1}\times10^{n-1}+a_{n-2}\times10^{n-2}+\cdots+a_0\times10^0+a_{-1}\times10^{-1}+a_{-2}\times10^{-2}+\cdots+a_{-m}\times10^{-m})_{10}=\sum_{i=-m}^{n-1}a_i\times10^i \tag{1.1.1}$$

这里 m、n 均为正整数，a_i 可以是 0、1、2……、9 中的任何一个，10 为基数，10^i 称为第 i 位的权。

二、二进制码（Binary Code）

在数字电路中广泛采用二进制数。在二进制中只有两个数码，即“0”和“1”，它的基数为 2，所以它“逢二进一”。例如，二进制数 $(N)_2=(1101,01)_2$ 表示的值是

$$(N)_2 = 1\times 2^3 + 1\times 2^2 + 0\times 2^1 + 1\times 2^0 + 0\times 2^{-1} + 1\times 2^{-2}$$

对于一个具有 n 位整数、m 位小数的二进制数可表示为

$$(N)_2 = (b_{n-1}b_{n-2}\cdots b_1 b_0 \cdot b_{-1}b_{-2}\cdots b_{-m})_2$$

按多项式表示法可以写为

$$(N)_2 = (b_{n-1}\times 2^{n-1} + b_{n-2}\times 2^{n-2} + \cdots + b_0\times 2^0 + b_{-1}\times 2^{-1} + \cdots + b_{-m}\times 2^{-m})_2$$

$$= \sum_{i=-m}^{n-1} b_i \times 2^i \qquad (1.1.2)$$

式中：m、n 为正整数，2 为基数，b_i 只能是 0、或 1，2^i 为第 i 位的权。

无论是十进制数，还是二进制数，在它的右边加上一个 0，就等于这个数乘上各自的基数，对于十进制来说，增为原来的十倍，对二进制来说，增为原来的两倍。

三、任意进制码

上述的十进制，二进制的表示方法可以推广到 R 进制，它有 R 个数码 0、1、2，…，$R-1$。由此可以得出任意进制数为

$$(N)_R = (k_{n-1}k_{n-2}\cdots k_0 \cdot k_{-1}k_{-2}\cdots k_{-m})_R$$

$$= \sum_{i=-m}^{n-1} k_i \times R^i \qquad (1.1.3)$$

其中，m、n 均为正整数，k_i 为 0，1，2，…，$R-1$ 中任一数码。R 为进制数。

1. 八进制码（Octal Code）

如果 $R=8$，有八个数码：0，1，2，3，4，5，6，7 任何一个八进制数可表示为

$$(\mathrm{N})_8 = \sum_{\mathrm{i}=-\mathrm{m}}^{\mathrm{n}-1} \mathrm{k_i}\cdot 8^{\mathrm{i}} \qquad (1.1.4)$$

例如 $(576)_8 = 5\times 8^2 + 7\times 8^1 + 6\times 8^0$。

2. 十六进制码（Hexadecimal Code）

如果 $R=16$，有十六个码，其中十个数码：0，1，…9，超过十的用英文字母表示，即：A、B、C、D、E、F 代表十、十一、十二、十三、十四、十五。任何一个十六进制数可表示为

$$(N)_{16} = \sum_{i=-m}^{n-1} k_i \cdot (16)^i \qquad (1.1.5)$$

例如：$(8A)_{16} = 8\times 16^1 + A\times 16^0$

以上讨论了十进制、二进制、八进制、十六进制的特点及表示方法，由此不难扩广到 R 进制。表 1.1.1 列出了几个选定的数在不同数制中的对照关系。

表 1.1.1　用不同的数制表示几个选定的数

十　进　制	二　进　制	八　进　制	十六进制
0	0	0	0
1	1	1	1
2	10	2	2
3	11	3	3

十 进 制	二 进 制	八 进 制	十六进制
4	100	4	4
5	101	5	5
6	110	6	6
7	111	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*
16	10000	20	10
17	10001	21	11
18	10010	22	12
19	10011	23	13
20	10100	24	14
32	100000	40	20
100	1100100	144	64

1. 2 数制之间的转换

一个数从一种数制的表示形式转换为等值的另一种数制表示形式，这就是数制转换。不同数制之间相互转换，是根据“如果两个有理数相等，则它们的整数部分和小数部分必定分别相等”的原理进行的。

一、二进制转换为十进制

二进制转换为十进制数用多项式替代法最为适宜。二进制数的多项式表达式如式(1.1.2)所示，它由系数 b_i，基数 2 以及幂 i 组成。所谓多项式代替法就是把这些系数，基数和幂均用十进制数代替，进而按十进制数进行运算，即得与该二进制数等值的十进制数。因此，由(1.1.2)式得

$$(N)_2 = (\sum_{i=-m}^{n-1} b_i \times 2^i)_{10} \qquad (1.2.1)$$

例 1.2.1 $(101011.1) = (\quad)_{10}$

由(1.2.1)式得

$$\begin{aligned} N_2 &= (101011.1)_2 \\ &= (1\times 2^5 + 0\times 2^4 + 1\times 2^3 + 0\times 2^2 + 1\times 2^1 + 1\times 2^0 + 1\times 2^{-1})_{10} \\ &= (43.5)_{10} \end{aligned}$$

实际求解时可以不用上述展开的系数乘权再相加的方法，而可以直接用权相加（相加那些系数为1的各位值）。

上述方法也适合于任何非十进制数到十进制数的转换。对于十六制来说，我们有

$$(N)_{16}=\left(\sum_{i=-m}^{n-1}k_i\times 16^i\right)_{10} \qquad (1.2.2))$$

例 1.2.2　$(12F\cdot B4)_{16}=$（　　）$_{10}$

$$\begin{aligned}(12F\cdot B4)_{16}&=(1\times 16^2+2\times 16^1+F\times 16^0+B\times 16^{-1}+4\times 16^{-2})_{10}\\&=(1\times 16^2+2\times 16^1+15\times 16^0+11\times 16^{-1}+4\times 16^{-2})_{10}\\&=(303.703125)_{10}\end{aligned}$$

二、十进制数转换为二进制数

将十进制数转换为非十进制数采用基数除法较为方便。不过必须将整数部分和小数部分分别进行转换，然后将它们的转换结果合并起来。下面讨论由十进制数到二进制数的转换过程。有一个十进制数 $(N)_{10}$，便有一个二进制数与其对应，即有

$$(N)_{10}=(b_{n-1}b_{n-2}\cdots b_1b_0.b_{-1}b_{-2}\cdots b_{-m})_2$$

其中，整数部分、小数部分分别进行转换。

1. 整数转换——基数除法　十进制整数转换为二进制数，采用逐次除以2，取余数的方法，其步骤如下：

（1）将给定的十进制数除以2，取其余数（1或0）作为二进制数的最低位（*LSB*）；

（2）把前一步所得之商再除以2，取其余数作为二进制数的次低位；

（3）重复步骤（2）将每次所得之商除以2，记下余数，直到最后相除之商为0，这时的余数即为二进制数的最高位（MSB）。

2. 小数转换——乘2取整　十进制小数转换为二进制小数，采用逐次将小数部分乘以2，取乘积的整数部分作为二进制小数的各有关数位。步骤如下：

（1）将给定的十进制小数乘以2，不论乘积整数部分为1还是0，取其整数部分作为二进制小数部分的最高位（MSB）；

（2）将前一步所得乘积中的小数部分再乘以2，取其积的整数部分作为二进制小数部分的次高位；

（3）重复（2），直到小数部分变为0时，转换即告结束；或者小数部分虽未变为0，但二进制小数的位数已达到预定的要求（如位数的要求或者转换的要求）时，运算也可结束。

3. 把一个带有整数和小数的十进制数转换成二进制数时，必须将整数部分和小数部分分别按基数除法和基数乘法进行转换，然后把两者的转换结果合并起来。

例 1.2.3　将十进制数 $(47.39)_{10}$ 转换成二进制数 $(N)_2$，误差 $e\leqslant 2^{-4}$

解：转换过程如下：

整数为47按整数转换方法——基数除法

$$
\begin{array}{r|l}
2 & 47 \quad \text{余数} \\
\hline
2 & 23 \quad \cdots\cdots 1 \quad b_0 \quad LSB \\
\hline
2 & 11 \quad \cdots\cdots 1 \quad b_1 \\
\hline
2 & 5 \quad \cdots\cdots 1 \quad b_2 \\
\hline
2 & 2 \quad \cdots\cdots 1 \quad b_3 \\
\hline
2 & 1 \quad \cdots\cdots 0 \quad b_4 \\
\hline
 & 0 \quad \cdots\cdots 1 \quad b_5 \quad MSB
\end{array}
$$

结果为 $(47)_{10}=(101111)_2$

小数 $(0.39)_{10}$按基数乘法转换

$$
\begin{array}{rl}
 & 0.39 \quad \text{取整} \\
\times & 2 \\
\hline
 & 〔0.〕78 \quad \cdots\cdots 0 \quad b_{-1} \quad MSB \\
\times & 2 \\
\hline
 & 〔1.〕56 \quad \cdots\cdots 1 \quad b_{-2} \\
\times & 2 \\
\hline
 & 〔1.〕12 \quad \cdots\cdots 1 \quad b_{-3} \\
\times & 2 \\
\hline
 & 〔0.〕24 \quad \cdots\cdots 0 \quad b_{-4} \quad LSB
\end{array}
$$

结果为 $(0.39)_{10}=(0.0110)_2$

将整数部分及小数部分的转换结果相加，便得到总的转换结果：

$(47.39)_{10}=(101111.0110)_2$

十进制数转换成其它进制的方法与十进制转换成二进制的方法相同，也是整数部分和小数部分分别进行转换。然后把两次转换结果相加，便得到总的转换结果。读者可根据上述方法自己练习。

三、二进制与八进制、十六进制的相互转换

由于二进制与八进制和十六进制正好满足 2^3 和 2^4 关系，因此它们之间的转换十分方便。

只要将二进制整数部分由右向左每三位或每四位为一组分开，若最高一组不足位，则在有效位左边加 0；小数部分由左向右每三位或每四位的一组分开，若最低位一组不足位，则在有效位右边加 0，则每组独立表示的二进制数，正好就是该组所对应的八进制数或十六进制数。

例 1.2.4 $(1101111\cdot10011)_2=$（　　）

解：$\frac{001}{1}\ \frac{101}{5}\ \frac{111}{7}\cdot\frac{100}{4}\ \frac{110}{6}$

∴ $(1101111\cdot10011)_2=(157.46)_8$

例 1.2.5 $(1110101\cdot10111)_2=(\quad)_{16}$

$\frac{0111}{7}\ \frac{0101}{5}\cdot\frac{1011}{B}\ \frac{1000}{8}$

∴ $(1110101\cdot10111)_2=(75.B8)_{16}$

八进制、十六进制转换为二进制是上述的逆过程，分别将每位八进制数或十六进制数用二进制代码写出来便可。

当要求将八进制和十六进制相互转换时，可通过二进制完成。

十进制数转换成八进制或十六进制数，可采用十进制转换成二进制的方法。也可以采用先把十进制数转换成二进制数，然后再把二进制数转换成八进制或十六进制数。

1.3 二进制数的算术运算

同十进制数一样，二进制数也可以进行加、减、乘、除四则运算，且运算规则也相同，所不同的是进位基数。由于二进制只有两个数字符号（0，1），所以二进制的运算比十进制的运算简单得多，且易于用数字电路来实现。

一、二进制数的加法

二进制数的加法和十进制数的加法一样，都是对应位相加，其加法规则是：

$0+0=0$　　$0+1=1$

$1+0=1$　　$1+1=10$（向高位进位）

即“逢二进一”。

例 1.3.1 求 $(0111)_2$与 $(0011)_2$ 之和

```
  0 1 1 1 (7)
  0 0 1 1 (3)
+1 1 1     进位
  1 0 1 0 (10)
```

其中最低位只是被加数与加数相加，不考虑低位向本位的进位，我们称为“半加”；而其它位还要考虑低位向本位的进位，即被加数＋加数＋低位向本位的进位，我们称为“全加”。

二、二进制数的减法和负数表示法

二进制数相减时，也是对应位相减，不够减时从高位借位，借 1 为 2。其减法规则为：

$1-1=0$　　$1-0=1$

$0-0=0$　　$0-1=1$（向高位借 1 当 2）

例 1.3.2 求 $(1101)_2$ 和 $(0110)_2$ 之差

1 1 0 1（13）

0 1 1 0（6 ）

−1 1　　借位

0 1 1 1（7 ）

需要指出的是，在数字系统或计算机中，为减少设备量，而是先把二进制数表示成它的反码或补码形式，然后用加法代替减法运算。

在这里提出了二进制数的符号问题。在数字设备中“+”、“−”也要数值化，规定将符号位加在数字的最前方。以 0 代表“+”（正），以 1 代表“−”（负）。如表 1.3.1 所示。

表 1.3.1　三种不同的二进码正负数表示法

十　进　制	二　　进　　数											
	原　　码				反　　码				补　　码			
	符号	绝　对　值			符号	绝　对　值			符号	绝　对　值		
−7	1	1	1	1	1	0	0	0	1	0	0	1
−6	1	1	1	0	1	0	0	1	1	0	1	0
−5	1	1	0	1	1	0	1	0	1	0	1	1
−4	1	1	0	0	1	0	1	1	1	1	0	0
−3	1	0	1	1	1	1	0	0	1	1	0	1
−2	1	0	1	0	1	1	0	1	1	1	1	0
−1	1	0	0	1	1	1	1	0	1	1	1	1
−0	0	0	0	0	1	1	1	1	0	0	0	0
+0					0	0	0	0				
1	0	0	0	1	0	0	0	1	0	0	0	1
2	0	0	1	0	0	0	1	0	0	0	1	0
3	0	0	1	1	0	0	1	1	0	0	1	1
4	0	1	0	0	0	1	0	0	0	1	0	0
5	0	1	0	1	0	1	0	1	0	1	0	1
6	0	1	1	0	0	1	1	0	0	1	1	0
7	0	1	1	1	0	1	1	1	0	1	1	1

从表 1.3.1 中可以看出，对正数的原码、反码、补码符号及绝对值的表示都是一样的，但对负数的原码、反码、补码绝对值的表示，则各不相同。下面我们分别加以讨论。

1. 原码表示法

这种表示法同原给定的二进制码的写法完全一致。正符号用数码 0 表示，负符号用数码 1 表示，符号位为最左边的一位。例如

$$+7 = +111 = \underline{0111},\quad -7 = -111 = \underline{1111}$$

2. 反码表示法

二进制的反码是这样得到的。先取一个参考数（参考数的每一位都为 1），从这个参

考数减去原（数）码，即得到1的补码。

$$\begin{array}{ll} 1111 & \text{参考数} \\ -0110 & \text{系　数} \\ \hline 1001 & \text{1补码} \end{array}$$

由于1补码可以通过将原码每一位取反（0→1，1→0）而获得，故又称反码。反码是补码的一个特例，即参考数为1的补码。

3. 补码表示法

二进制的补码是这样得到的。对 N 位二进制数来说，取参考数为比 N 位高一位的（参考数第1位为1，其它位都为0）的二进制数，如果给出的是四位二进制数，参考数就是五位的二进制数。然后从这个参考数减去原（数）码，即得到2补码。例如

$$\begin{array}{ll} 10000 & \text{参考数} \\ -\ 0110 & \text{原　码} \\ \hline 1010 & \text{2补码} \end{array}$$

为了方便，将2补码简称为补码。补码可以通过反码最低位加1而得，看表1.3.1。

上述表示法，不仅适用于整数，也适用于小数。

二进制负数采用上述三种表示法的目的在于对于负数运算，可以用加法代替减法运算。符号也参加运算了。由于采用这种方法，大大减少了运算程序，并节省了设备。下面举例说明。

例1.3.3　已知 $X=+1101$，$Y=+0110$，用原、反码及补码计算 $Z=X-Y$

解：（a）原码运算 $[X]_{原}=01101$ $[Y]_{原}=00110$

$$\begin{array}{l} 01101 \\ -00110 \\ \hline 00111 \end{array}$$

即 $[Z]_{原}=00111$ 其值为 $Z=+0111$。

（b）反码运算

采用反码运算时，需将原数表示成反码，进行反码减法时可按 $[X]_{反}+[-Y]_{反}$ 进行，将减法变为加法运算。其运算结果仍为反码。

即 $[Z]_{反}=[X]_{反}+[-Y]_{反}$ 其算式如下：

$[X]_{反}=01101$　$[-Y]_{反}=11001$

$$\begin{array}{l} 01101 \\ +11001 \\ \hline 100110 \\ +\qquad 1 \\ \hline 00111 \end{array}$$

对反码运算是按下列规则进行的：

$[Z]_{反}=[X+Y]_{反}=[X]_{反}+[Y]_{反}+$符号位进位。

$[Z]_{反}=00111$　其值为 $Z=+0111$。

（C）补码运算

补码运算过程与反码运算相似，按 $〔X〕_{补}+〔-Y〕_{补}$ 进行，将减法运算变为加法运算。其运算结果仍为补码。

$〔X〕_{补}=01101$　　$〔-Y〕_{补}=11010$

$$\begin{array}{r} 01101 \\ +11010 \\ \hline 100111 \end{array}$$

其符号进位自然丢失。即 $〔Z〕_{补}=00111$ 其值为 $Z=+0111$。

以上三种码运算结果相同。

例 1.3.4　已知 $X=+011$，$Y=+101$ 用补码计算 $Z=X-Y$

解：$Z=X+〔-Y〕$　　$〔X〕_{补}=0011$　　$〔-Y〕_{补}=1011$

$$\begin{array}{r} 0011 \\ +1011 \\ \hline 1110 \end{array}$$

即 $〔Z〕_{补}=1110$ 符号位为负，其 $Z=-010$。

由以上例子可以看出

①反码、补码将减法运算变为加法运算。

②反码、补码运算时，符号位也被看成是一位数码，并与数位一样以同样的加法和进位规则来处理，所得结果的符号位也就是正确结果的符号。

③用什么码运算，其运算结果也就是什么码，在转换成原数值时，对反码和补码的运算结果，要特别注意，当符号位为 0 时，数位不变；当符号位为 1 时，应分别将结果求反或求补才是原数值。

④补码运算最简单，故在数字系统和计算机多采用补码运算。

三、二进制数的乘法和除法

1. 二进制数的乘法

为了用二去乘一个二进制数，只需将每个数字向左移动一位即可。例如

$(25)_{10}=(11001)_2$

$(25)_{10}\times 2=(110010)_2=(50)_{10}$

2. 二进制数的除法

将一个二进制数的每一位向右移动一位，等值于将此数除以 2。例如

$(36)_{10}=(0100100)_2$

$(36)_{10}\div 2=(0010010)_2=(18)_{10}$

1.4　十进制数的代码表示

在数字电子计算机中，十进制数的运算处理通常是转换成对应的二进制数进行。除此以外，在计算机的数据输入输出，一些控制应用及商业数据处理等方面，十进制还有一种表示方法，即每位十进制数用其对应的二进制代码来表示，称为二一十进制编码，也