第 3 章　运算方法和运算部件

1. 在讨论计算机中用到的数值时，经常遇到哪些记数制？

答：经常遇到的记数制有二进制、八进制、十六进制和十进制。1 位八进制数用 3 位二进制数来表示，1 位十六进制数用 4 位二进制数来表示，因此在计算机内部，这 3 种记数制并没有差别，仅是在书写时，同一数据当用八进制或十六进制表示时，比二进制短得多，也容易识别。

1 位十进制数在计算机中也是用 4 位二进制数来表示的，称为 BCD 码。同一代码表示多位数据时，在不同的记数制中，其值是不同的。例如代码 00010001，当其为 BCD 码（8421 码）时，等于十进制数 11；当其为十六进制（或二进制）时，等于十进制数 17。

2. 今有两数 00100110 和 01000111，求两数之和。

（1）两数都是二进制码，结果用十进制表示。

（2）两数都是 BCD 码，结果用十进制表示。

答：（1）二进制码相加。

$$\begin{array}{r} 0\,0\,1\,0\quad 0\,1\,1\,0 \\ +\ 0\,1\,0\,0\quad 0\,1\,1\,1 \\ \hline 0\,1\,1\,0\quad 1\,1\,0\,1 \end{array}$$

运算结果用十进制表示为 64＋32＋8＋4＋1＝109。两数相加为 38＋71＝109。

（2）BCD 码相加。

$$\begin{array}{rl} & 0\,0\,1\,0\quad 0\,1\,1\,0 \\ + & 0\,1\,0\,0\quad 0\,1\,1\,1 \\ \hline & 0\,1\,1\,0\quad 1\,1\,0\,1 \\ + & \qquad 1\quad 0\,1\,1\,0 \\ \hline & 0\,1\,1\,1\quad 0\,0\,1\,1 \end{array}$$

低位加 6，进位为 1

用十进制表示的结果为 73。两数相加为 26＋47＝73。

本题的目的是了解二进制数和二-十进制数（BCD 码）的运算方法的差别，相同代码表示的运算结果是不同的。

3. 人们习惯使用的是十进制数据，而计算机内部运算的是二进制数据，通常是怎样处理的？

答：人们输入的十进制数据一般是用 ASCII 码表示的，先按位转换成 BCD 码，然后将 BCD 码转换成二进制码，在计算机中运算，运算结果再转换成 BCD 码，按位用十进制码输出。

某些计算机设置有十进制运算指令，可以用硬件直接对两个 BCD 码数据进行运算。

4. 将$(73)_{10}$转换成二进制码。

答：（1）用十进制数转换。

2⌊73	余数	结果
2⌊36	1	低位
2⌊18	0	↓
2⌊9	0	↓
2⌊4	1	↓
2⌊2	0	↓
2⌊1	0	↓
0	1	高位

$(73)_{10}=(1001001)_2$

(2) 用 BCD 码转换。

2⌊ 0111 0011	余数	结果
2⌊ 0011 0110*	1	低位
2⌊ 0001 1000**	0	↓
2⌊ 0000 1001	0	↓
2⌊ 0000 0100	1	↓
2⌊ 0000 0010	0	↓
2⌊ 0000 0001	0	↓
0000 0000	1	高位

$(01110011)_{BCD}=(1001001)_2$

注：* 高位除 2 得高位商 0011，高位余数 0001，将该余数加到低位，所以低位为 (0011+1010)/2=1101/2，得低位商 0110 和余数 1。

** 低位(0110+1010)/2=10000/2，得低位商 1000 和余数 0。下同。

如果高位无余数，低位直接除 2。

(3) 采用乘加方法转换。

$(73)_{10}=(7\times10+3)_{10}=(0111\times1010+0011)_2$

$$\begin{array}{r} 0\ 1\ 1\ 1 \\ \times\ 1\ 0\ 1\ 0 \\ \hline 0\ 1\ 1\ 1 \\ 0\ 1\ 1\ 1 \\ \hline 1\ 0\ 0\ 0\ 1\ 1\ 0 \end{array} \qquad \begin{array}{r} 1\ 0\ 0\ 0\ 1\ 1\ 0 \\ +\quad\ 0\ 0\ 1\ 1 \\ \hline 1\ 0\ 0\ 1\ 0\ 0\ 1 \end{array}$$

$(73)_{10}=(1001001)_2$

5. 采用乘加方法，将$(254)_{10}$转换成二进制码。

答： $(254)_{10}=2\times10^2+5\times10+4=((2\times10)+5)\times10+4$

$=((0010\times1010)+0101)\times1010+0100$

$=(10100+0101)\times1010+0100$

$=11001\times1010+0100$

$=11111010+0100$

$=(11111110)_2$

注：在计算机中一般采用乘加方法，所以将公式变换一下，以利于编程和节省运算时间。

6. 将$(0.75)_{10}$转换成二进制码。

答：(1) 用十进制转换。

结果	0.75×2
高位 1	.50×2
低位 1	.00

$(0.75)_{10}=(0.11)_2$

(2) 用 BCD 码转换。

$(0.75)_{10}=(0.01110101)_{BCD}$

		0111 0101	×2
		1110 1010	
结果		0111 0110	高位 +6+1(低位来的进位)，低位 +6
高位	1	0101 0000	×2
↓		1010 0000	
		0110	高位 +6
低位	1	0000 0000	

$(0.75)_{10}=(0.11)_2$

7. 将$(0.73)_{10}$转换成二进制码(小数点后取 4 位有效值)。

答：(1) 用十进制转换。

结果	0.73×2
高位 1	.46×2
↓ 0	.92×2
↓ 1	.84×2
低位 1	.68

$(0.73)_{10}=(0.1011)_2$

(2) 用 BCD 码转换。

$(0.73)_{10}=(0.01110011)_{BCD}$

		0111 0011	×2
		1110 0110	
结果		0110	高位＋6
高位	1	0100 0110	×2
↓		1000 1100	
		0001 0110	高位＋1(低位来的进位)，低位＋6
	0	1001 0010	×2
	1	0010 0100	
		0110	高位＋6
低位		1000 0100	×2
	1	0000 1000	
		0110	高位＋6
		0110 1000	

$(0.73)_{10}=(0.1011)_2$

8. 将$(73.75)_{10}$转换成二进制码。

答：由于整数部分和小数部分的转换方法不同，所以要分别计算。由第 4 题得$(73)_{10}=$

$(1001001)_2$，由第 6 题得$(0.75)_{10}=(0.11)_2$，所以$(73.75)_{10}=(1001001.11)_2$。

9. 将下列二进制数转换成十进制数。

(1) 1001001　　(2) 11000111

答：将二进制数除以$(10)_{10}$，余数即为结果的最低位，如果商$\geqslant(10)_{10}$，将商再除以$(10)_{10}$，其余数为次低位……当商$<(10)_{(10)}$时，运算结束。$(10)_{10}=(1010)_2$。

(1)

```
                  1 1 1   ---- 十位   商<(1010)2
1 0 1 0 ) 1 0 0 1 0 0 1
          1 0 1 0
          ---------
            1 0 0 0 0 1
              1 0 1 0
            -----------
              0 1 1 0 1
                1 0 1 0
              -----------
                0 0 1 1   ---- 个位
```

$(1001001)_2=(01110011)_{BCD}=(73)_{10}$

(2)

```
                  1 0 0 1 1   ---- 商>(1010)2
1 0 1 0 ) 1 1 0 0 0 1 1 1
          1 0 1 0
          ---------
            0 1 0 0 1 1 1
              1 0 1 0
            -----------
              1 0 0 1 1
                1 0 1 0
              -----------
                1 0 0 1   ---- 个位
```

```
                1   ---- 百位
1 0 1 0 ) 1 0 0 1 1
          1 0 1 0
          ---------
            1 0 0 1   ---- 十位
```

$(11000111)_2=(000110011001)_{BCD}=(199)_{10}$

10. 将二进制码 0.1110 转换成十进制码。

答：

结果	$1\,1\,1\,0\times1\,0\,1\,0=1\,0\,0\,0\,1\,1\,0\,0$
高位 $1\,0\,0\,0$	$1\,1\,0\,0\times1\,0\,1\,0=0\,1\,1\,1\,1\,0\,0\,0$
↓ $0\,1\,1\,1$	$1\,0\,0\,0\times1\,0\,1\,0=0\,1\,0\,1\,0\,0\,0\,0$
低位 $0\,1\,0\,1$	$0\,0\,0\,0$

$(0.1110)_2=(0.1000\ 0111\ 0101)_{BCD}=(0.875)_{10}$

11. 将二进制码 11001001.0101 转换成八进制数和十六进制数。

答：$(11001001.0101)_2=(\underline{011}\ \underline{001}\ \underline{001}.\ \underline{010}\ \underline{100})_2=(311.24)_8$

$(11001001.0101)_2=(\underline{1100}\ \ \underline{1001}.\underline{0101})_2=(C9.5)_{16}$

12. 将八进制数$(57.43)_8$转换成二进制数和十六进制数。

答：$(57.43)_8=(\underline{101}\ \underline{111}.\underline{100}\ \underline{011})_2=(101111.100011)_2$。八进制转换成十六进制过程：先转换成二进制，再转换成十六进制。在计算机内部，八进制、十六进制和二进制的表现形式是相同的。

$(57.43)_8=(101111.100011)_2=(\underline{0010}\ \underline{1111}.\ \underline{1000}\ \underline{1100})_2=(2F.8C)_{16}$

13. 将十进制数 7 $\frac{3}{4}$和$\frac{3}{64}$转换成二进制数。

答：$\frac{1}{4}=2^{-2}$，$\frac{3}{4}=11_2\times2^{-2}=(0.11)_2$，所以 $7\frac{3}{4}=(111.11)_2$，$\frac{1}{64}=2^{-6}$，$\frac{3}{64}=11_2\times2^{-6}=(0.000011)_2$。

14. 完成下列二进制运算(手工运算)。

(1) 101.111+11.011　　(2) 1001.10−110.01

(3) 1001.10×11.01　　(4) 101110111÷1101

答：(1) 101.111+11.011=1001.010

```
  1 0 1.1 1 1
+ 0 1 1.0 1 1
-------------
  1 0 0 1.0 1 0
```

(2) 1001.10−110.01=11.01

```
  1 0 0 1.1 0
-   1 1 0.0 1
-------------
      1 1.0 1
```

(3) 1001.10×11.01=11110.1110

```
      1 0 0 1.1 0
    ×     1 1.0 1
    -------------
      1 0 0 1 1 0
    1 0 0 1 1 0
  1 0 0 1 1 0
-------------------
  1 1 1 1 0.1 1 1 0
```

(4) 101110111÷1101=11100

```
                  1 1 1 0 0
          ---------------------
1 1 0 1 ) 1 0 1 1 1 0 1 1 1
            1 1 0 1
          -----------
            1 0 1 0 0
              1 1 0 1
            -----------
                1 1 1 1
                1 1 0 1
              -----------
                    1 0 1 1
```

15. 请写出 X 为小数和整数时的补码公式。

答：当 X 为小数时，定义如下：

$$[X]_{补}=\begin{cases}X, & 0\leqslant X<1\\ 2+X=2-|X|, & -1\leqslant X<0\end{cases}$$

当 X 为整数时，定义如下：

$$[X]_{补}=\begin{cases}X, & 0\leqslant X<2^n\\ 2^{n+1}+X=2^{n+1}-|X|, & -2^n\leqslant X<0\end{cases}$$

其中 n 为数值部分的长度(不包括符号位)。

16. 当 X 为负值时，$[X]_{补}$ 与 $[X]_{反}$ 有什么关系？

答：当 X 为负值时，$[X]_{补}=2+X$，$[X]_{反}=2-2^{-n}+X$，所以 $[X]_{补}=[X]_{反}+2^{-n}$。

本题假设 X 为小数，n 为小数点后的有效位数。

在计算机中，从负数的原码转换成补码时，通常采用取数的反码，并在最低位+1 的方法。

17. 为什么在计算机中进行加减法运算时都采用补码运算的方法？

答：如果采用原码运算，由于加减法运算的逻辑表达式是不同的，所以需要采用两套电路来分别执行加法运算和减法运算。而采用补码运算时，由于下列两公式的存在：

$$[X+Y]_{补}=[X]_{补}+[Y]_{补}$$

$$[X-Y]_{补}=[X]_{补}+[-Y]_{补}$$

所以加减法运算都可以用加法来实现，节省了一套电路。而从 $[Y]_{补}$ 转变到 $[-Y]_{补}$ 是很方便的，因为在运算部件中，数据都是存放在寄存器中的，寄存器的每一位都有两个电位不同的输出端，如果 $[Y]_{补}$ 从一个输出端输出，则 $[-Y]_{补}$ 从另一个输出端输出，并在最低位+1 即可。+1 信号送到加法器最低位的进位输入端。

18. 当 X 为负数时，为什么 $[X]_{补}$ 和 $[X]_{反}$ 能表示的数值范围不同(假设 X 为小数，举例

说明)?

答：设 $X=-0.1010$，$Y=-0.0110$，$X+Y=-0.1010-0.0110=-1$。

$[X]_{补}+[Y]_{补}=1.0110+1.1010=1.0000$，即为$-1$。

$[X+Y]_{反}=0.1111$(反码运算规则见下题)，两负数相加，结果的符号位为0(表示正数)，说明该结果不正确，即溢出。由此说明$[X]_{补}$可以为-1，而$[X]_{反}$不能为-1，其最小的负数为$[X]_{反}=1.0000$，即 $X=-(1-2^{-4})$。

19. 试举例说明反码运算的规则。

答：$[X+Y]_{反}=[X]_{反}+[Y]_{反}+$最高位产生的进位信号(加到最低位)。

设 X、Y 为整数，数值部分有4位，符号位1位(为最高位)。举例如下：

(1) $X=-1$，$Y=+2$

$[X]_{反}=11110$，$[Y]_{反}=00010$

$$\begin{array}{r} 1\ 1\ 1\ 1\ 0 \\ +\quad 0\ 0\ 0\ 1\ 0 \\ \hline 1\ 0\ 0\ 0\ 0\ 0 \\ +\qquad\qquad 1 \\ \hline 0\ 0\ 0\ 0\ 1 \end{array}$$

最高位进位加到最低位

$X+Y=1$

(2) $X=-7$，$Y=2$

$[X]_{反}=11000$，$[Y]_{反}=00010$

$$\begin{array}{r} 1\ 1\ 0\ 0\ 0 \\ +\ 0\ 0\ 0\ 1\ 0 \\ \hline 1\ 1\ 0\ 1\ 0 \end{array}$$

最高位无进位，最低位不需要加1

$[X+Y]_{反}=11010$，$X+Y=-0101$(即-5)

(3) 参考第18题，$[X]_{反}=1.0101$，$[Y]_{反}=1.1001$

$[X+Y]_{反}$：

$$\begin{array}{r} 1.0\ 1\ 0\ 1 \\ +\ 1.1\ 0\ 0\ 1 \\ \hline 1\ 0.1\ 1\ 1\ 0 \\ +\qquad\quad 1 \\ \hline 0.1\ 1\ 1\ 1 \end{array}$$

最高位有进位，加到最低位

溢出

反码运算需要将最高位产生的进位信号加到最低位，且零值有两种编码。在计算机中一般不采用反码运算。因此本题仅供参考，暂时可以不必考虑。

20. 求下列各数的原码：

(1) $[X]_{补}=0.1010$　　(2) $[X]_{反}=0.1010$

(3) $[X]_{补}=1.1010$　　(4) $[X]_{反}=1.1010$

答：(1) $[X]_{原}=0.1010$　　(2) $[X]_{原}=0.1010$

(3) $[X]_{原}=1.0110$　　(4) $[X]_{原}=1.0101$

已知补码求原码的方法和已知原码求补码的方法类似。反码的情况相同。

21. 设 $X=(-69)_{10}$，求$[X]_{原}$、$[X]_{补}$和$[X]_{移}$。

答：$[X]_{原}=11000101$　　最高位为符号位。

$[X]_{补}=10111011$　　同上。

$[X]_{移}=00111011$　　同上。

22. 设$[X]_{补}=0.1011$，$[Y]_{补}=1.1110$，求$[X+Y]_{补}$和$[X-Y]_{补}$之值。

答：$[-Y]_{补}=0.0010$

$[X+Y]_{补}=[X]_{补}+[Y]_{补}=0.1011+1.1110=0.1001$

$[X-Y]_{补}=[X]_{补}+[-Y]_{补}=0.1011+0.0010=0.1101$

23. 设$[X]_{补}=1.0101$，$[Y]_{补}=0.0010$，求$[X+Y]_{补}$和$[X-Y]_{补}$之值。

答：$[-Y]_{补}=1.1110$

$[X+Y]_{补}=[X]_{补}+[Y]_{补}=1.0101+0.0010=1.0111$

$[X-Y]_{补}=[X]_{补}+[-Y]_{补}=1.0101+1.1110=1.0011$

24. 设 $X=-0.1101$，$Y=-0.1001$，求$[X+Y]_{补}$之值。

答：$[X]_{补}=1.0011$，$[Y]_{补}=1.0111$

$[X+Y]_{补}=[X]_{补}+[Y]_{补}=1.0011+1.0111=10.1010$　溢出

25. 已知$[X]_{移}=10101$，$[Y]_{移}=10011$，求$[X+Y]_{移}$和$[X-Y]_{移}$之值。

答：$[Y]_{补}=00011$，$[-Y]_{补}=11101$

$[X+Y]_{移}=[X]_{移}+[Y]_{补}=10101+00011=11000$

$[X-Y]_{移}=[X]_{移}+[-Y]_{补}=10101+11101=10010$

26. 写出下列各二进制数的原码、补码和反码：

0.1010，0，−0，−0.1010，0.1111，−0.0100

答：

X	$[X]_{原}$	$[X]_{补}$	$[X]_{反}$
0.1010	0.1010	0.1010	0.1010
0	0.0000	0.0000	0.0000
−0	1.0000	0.0000	1.1111
−0.1010	1.1010	1.0110	1.0101
0.1111	0.1111	0.1111	0.1111
−0.0100	1.0100	1.1100	1.1011

27. 已知$[X]_{补}$为下列各值：(1)0.1110，(2)1.1100，(3)0.0001，(4)1.1111，(5)1.0000。求 X(真值)。

答：X 的真值如下：

(1)0.1110　(2)−0.0100　(3)0.0001　(4)−0.0001　(5)−1.0000

28. 已知 $X=0.1011$，$Y=-0.0101$。求$[X]_{补}$、$[X/2]_{补}$、$[X/4]_{补}$、$[2X]_{补}$、$[Y]_{补}$、$[Y/2]_{补}$、$[Y/4]_{补}$和$[2Y]_{补}$。

答：$[X]_{补}=0.1011$

$[X/2]_{补}=0.0101$　舍去最低位

$[X/4]_{补}=0.0010$　再舍去最低位

$[2X]_{补}$为溢出

$[Y]_{补}=1.1011$

$[Y/2]_{补}=1.1101$　舍去最低位

$[Y/4]_{补}=1.1110$　再舍去最低位

$[2Y]_{补}=1.0110$

二进制数除以 2，可将数右移 1 位实现之，但要保留符号位不变；乘以 2，将数左移 1 位，最低位补充 0，要判别数据是否溢出。

29. 用压缩十进制数串表示下列十进制数：+66，−78，+254，−396，+1980，−1992。

答：用一个字节存放两个十进制数位，其值用 BCD 码或 ASCII 码的低 4 位表示，符号位也占半个字节并放在最低数字位之后，其值可从 4 位二进制码中的 6 种冗余状态中选用，例如用 B(1011)表示正号，D(1101)表示负号，并规定数字和符号位个数之和必须为偶数，否则在最高位之前补一个 0。用压缩十进制串表示的数据如下：

数字	+66	−78	+254	−396	+1980	−1992
压缩十进制串	066B	078D	254B	396D	01980B	01992D

例如，066B 在计算机内表示则为 0000 0110 0110 1011。

30. 9 位补码(内含 1 个符号位)定点整数和定点小数所能表示的绝对值最大的负数是多少？

答：绝对值最大的负数即是最小的负数。定点整数为−256，定点小数为−1。

31. 说出 $n+1$ 位(含符号位)定点整数(原码、补码、反码和移码)的范围。说出 $n+1$ 位(含符号位)定点小数(原码、补码、反码和移码)的范围。

答：(1) 定点整数：

原码　$-(2^n-1)\sim(2^n-1)$

补码　$-2^n\sim(2^n-1)$

反码　$-(2^n-1)\sim(2^n-1)$

移码　$-2^n\sim(2^n-1)$

(2) 定点小数：

原码　$-(1-2^{-n})\sim(1-2^{-n})$

补码　$-1\sim(1-2^{-n})$

反码　$-(1-2^{-n})\sim(1-2^{-n})$

移码　在计算机中移码仅可能用于阶码，所以无小数。

32. 数据 1.0000，当其为原码、补码和反码时表示的真值是多少？

答：原码表示真值−0，补码表示真值−1，反码表示真值$-(1-2^{-4})$。

33. 补码(含符号位)010101、101010、0.10101 和 1.01010 扩充成 8 位后等于什么？

答：扩充后应保持数值不变，所以整数扩充的位数应安排在左面，小数扩充的位数应安排在右面。整数按符号扩充，小数补以 0，所以扩充后的值为 00010101、11101010、0.1010100和 1.0101000。

34. 设有 5 位二进制数据，当其为无符号整数时，写出数据范围；若为有符号数，最高位为符号，写出数据范围。上述各种情况各能表示多少个数据？

答：无符号整数范围为$(0\sim31)_{10}$，相邻数据的间隔为 1，所以可表示 32 个数据。

有符号原码表示的数据范围为$(-15\sim+15)_{10}$，由于有两种状态 00000 和 10000 均表示数据0_{10}(+0 和−0)，所以能表示的数据为 31 个。

反码表示的数据范围也为$(-15\sim+15)_{10}$，由于有两种状态 00000 和 11111 均表示数据0_{10}(+0 和−0)，所以能表示的数据为 31 个。

补码表示的数据范围为$(-16\sim+15)_{10}$，数据 0 只有一种表示形式 00000。如果$[X]_{补}=00000$，则$[-X]_{补}=11111+1$(反码+1)$=00000$，即正 0($[X]_{补}$)和负 0($[-X]_{补}$)均为 00000。$[X]_{补}$ 10000 表示的数据为$(-16)_{10}$，假如将$[-X]_{补}$转换成原码则为 11111+1=

100000，应为+16，但此时的原码已溢出。5 位补码能表示 32 个数据。

35. 设 $X=0.1001$，$Y=-0.1011$，试用原码一位乘法、补码一位乘法（布斯算法）和原码两位乘法计算 $X\cdot Y$ 的乘积，并说明求部分积的次数（即加减运算次数）和移位次数。

答：$[X]_{补}=0.1001$，$[-X]_{补}=1.0111$，$2[-X]_{补}=110.1110$，$[Y]_{补}=1.0101$

（1）原码一位乘法。

部分积	乘数	说明
0 0.0 0 0 0	1 0 1 1	初始值
0 0.1 0 0 1		$+X$
0 0.1 0 0 1		1 →（右移 1 位）
0 0.0 1 0 0	1 1 0 1	
0 0.1 0 0 1		$+X$
0 0.1 1 0 1		1 →
0 0.0 1 1 0	1 1 1 0	
0 0.0 0 0 0		$+0$
0 0.0 1 1 0		1 →
0 0.0 0 1 1	0 1 1 1	
0 0.1 0 0 1		$+X$
0 0.1 1 0 0		1 →
0 0.0 1 1 0	0 0 1 1	

$|X\cdot Y|=0.1001\times 0.1011=0.01100011$

$X\cdot Y=-0.01100011$

$[X\cdot Y]_{原}=1.01100011$

（2）补码一位乘法。

部分积	乘数	说明
0 0.0 0 0 0	1 0 1 0 1 0	初始值，乘数最后位补一个 0
1 1.0 1 1 1		$+[-X]_{补}$
1 1.0 1 1 1		1 →（右移 1位）
1 1.1 0 1 1	1 1 0 1 0 1	
0 0.1 0 0 1		$+[X]_{补}$
0 0.0 1 0 0		1 →
0 0.0 0 1 0	0 1 1 0 1 0	
1 1.0 1 1 1		$+[-X]_{补}$
1 1.1 0 0 1		1 →
1 1.1 1 0 0	1 0 1 1 0 1	
0 0.1 0 0 1		$+[X]_{补}$
0 0.0 1 0 1		1 →
0 0.0 0 1 0	1 1 0 1 1 0	
1 1.0 1 1 1		$+[-X]_{补}$
1 1.1 0 0 1	1 1 0 1	

$[X \cdot Y]_{补}=11.10011101 \quad X \cdot Y=-0.01100011$

(3) 原码两位乘法。

部分积	乘数	C	说明
0 0 0.0 0 0 0	1 0 1 1	0	初始值
1 1 1.0 1 1 1			$+[-X]_{补}$
1 1 1.0 1 1 1		1	2 →
1 1 1.1 1 0 1	1 1 1 0		
1 1 1.0 1 1 1			$+[-X]_{补}$
1 1 1.0 1 0 0		1	2 →
1 1 1.1 1 0 1	0 0 1 1		
0 0 0.1 0 0 1			$+[X]_{补}$
0 0 0.0 1 1 0	0 0 1 1		

$|X \cdot Y|=0.01100011 \quad X \cdot Y=-0.01100011$

$[X \cdot Y]_{原}=1.01100011$

(4) 加减法运算和移位次数。

设数值部分有 n 位，令 $n=4$，则

原码一位乘法：加法 4 次(n 次)，移位 4 次(n 次)。

补码一位乘法：加减法 5 次($n+1$ 次)，移位 4 次(n 次)。

原码两位乘法：加减法 3 次($n/2+1$ 次)，移位 2 次($n/2$ 次)。

36. 请说明乘法运算部分积的符号位取多少位？

答：一位乘法，部分积符号位取 2 位，因为加减时，数值部分有可能会产生进位到符号位，此时两符号位中的高位是真正的符号位。同理，两位乘法要再增加 1 位符号位，即成为 3 符号位，因为加减时，数值部分有可能产生进位到第 2 符号位，此时最高位是真正的符号位。乘法运算过程中的移位为算术移位，移位时要保持符号位不变。举例如下：设 $X=0.1111$，$Y=0.1010$。执行原码两位乘法。

部分积	乘数	说明	
0 0 0.0 0 0 0	1 0 1 0	初始值	
0 0 1.1 1 1 0		$+2X$	
0 0 1.1 1 1 0		2 →	
0 0 0.0 1 1 1	1 0 1 0		
0 0 1.1 1 1 0		$+2X$	部分积进位到第 2 符号位
0 1 0.0 1 0 1		2 →	
0 0 0.1 0 0 1	0 1 1 0		

$X \cdot Y=0.1111 \times 0.1010=0.10010110$

37. 试画出阵列乘法器的逻辑框图，并说明之。

答：为了进一步提高乘法运算速度，可采用类似于人工计算的方法，用图 3.1 所示的一个阵列乘法器完成 $X \cdot Y$ 乘法运算($X=X_1X_2X_3X_4$，$Y=Y_1Y_2Y_3Y_4$)。

人工计算方法如下：

$$
\begin{array}{cccccc}
 & & X_1 & X_2 & X_3 & X_4 \\
 & \times) & Y_1 & Y_2 & Y_3 & Y_4 \\
\hline
 & & X_1Y_4 & X_2Y_4 & X_3Y_4 & X_4Y_4 \text{---该行+初始值得 } P_1 \\
 & X_1Y_3 & X_2Y_3 & X_3Y_3 & X_4Y_3 & \text{--------该行+}P_1\text{ 得 }P_2 \\
X_1Y_2 & X_2Y_2 & X_3Y_2 & X_4Y_2 & & \text{----------该行+}P_2\text{ 得 }P_3 \\
+) \quad X_1Y_1 & X_2Y_1 & X_3Y_1 & X_4Y_1 & & \text{----------该行+}P_3\text{ 得 }P_4\text{(结果)} \\
\hline
P_{4-1} & P_{4-2} & P_{4-3} & P_{4-4} & P_{4-5} & P_{4-6} \quad P_{4-7} \quad P_{4-8}
\end{array}
$$

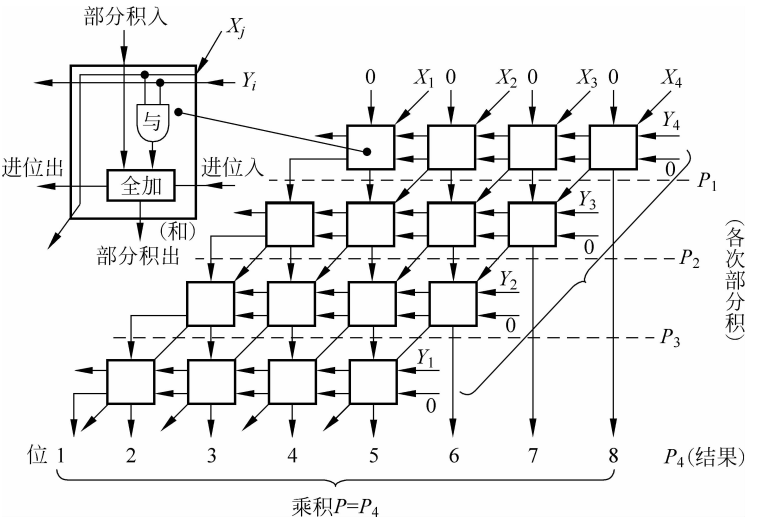

图 3.1　阵列乘法器

图中的每一个方框包含一个与门和一位全加器。阵列的每一行送入乘数 Y 的同一数位，例如第一行的各方框都送入乘数的 Y_4，第二行各方框都送入乘数的 Y_3……而各行错开形成的每一斜列则送入被乘数的每一数位。由于部分积的初始值为 0，所以第一行的部分积输入均为 0；由于最右边的斜列无进位输入，故进位输入也为 0。乘积(P_4)得 8 位，最高位为 P_{4-1}，依次为 P_{4-2}，P_{4-3}，…，P_{4-8}。

38. 试证明采用加减交替法的原码除法运算的正确性。

答：加减交替法是对恢复余数法的一种修正，当某一次求得的余数 R_i 为负时，不是恢复它，而是继续求下一位商，但用加上除数(+Y)的办法来取代(−Y)操作，其他操作依然不变，证明如下。

在恢复余数除法中，若第 $i-1$ 次求商的余数为 R_{i-1}，下一次求商的余数为 R_i，则

$R_i=2R_{i-1}-Y$　(R_{i-1}左移 1 位，再减去除数 Y)

如果 $R_i<0$，商的第 i 位上 0，并执行以下操作：恢复余数(+Y)，将余数左移 1 位，再减 Y，得 R_{i+1}。其过程可用公式表示如下：

$R_{i+1}=2(R_i+Y)-Y=2R_i+Y$。由此得到证明。

39. 定点除法运算为什么先要判断一下被除数是否小于除数？

答：如果被除数不小于除数，商将大于或等于 1，溢出，不能得到正确结果。

40. 叙述定点原码一位除法(不恢复余数法)的运算规则。假设 $X=-0.10110$，$Y=0.11111$，用不恢复余数法求 X/Y 之商。

答：两个原码数相除，其商的符号为两数符号的异或值，数值为两数绝对值相除的商，通常采用加减交替法，即为不恢复余数法。

除法运算规则如下（被除数和除数为绝对值）：

（1）被除数减除数，余数为正，则溢出；余数为负，上商 0，余数左移 1 位后加除数。

（2）以后的操作遵循以下规则：

- 当余数为正时，上商 1，求下一位商的办法是余数左移 1 位，再减去除数。
- 当余数为负时，上商 0，求下一位商的办法是余数左移 1 位，再加上除数。
- 重复此步操作，当达到商的有效位长度时，运算结束。

（3）如果最后一次商 0，而又要求得出正确余数，则再加上除数，即得正确余数；如果最后一次商 1，即为正确余数。

设 $X=-0.10110$，$Y=0.11111$　则

$|X|=0.10110$　$|Y|=0.11111$　$[-Y]_{补}=1.00001$

被除数或余数	说明
0 0.1 0 1 1 0	被除数
1 1.0 0 0 0 1	$+[-Y]_{补}$
1 1.1 0 1 1 1	商 0，$\xleftarrow{1}$（左移 1 位）
1 1.0 1 1 1 0	
0 0.1 1 1 1 1	$+[Y]_{补}$
0 0.0 1 1 0 1	商 1，$\xleftarrow{1}$
0 0.1 1 0 1 0	
1 1.0 0 0 0 1	$+[-Y]_{补}$
1 1.1 1 0 1 1	商 0，$\xleftarrow{1}$
1 1.1 0 1 1 0	
0 0.1 1 1 1 1	$+[Y]_{补}$
0 0.1 0 1 0 1	商 1，$\xleftarrow{1}$
0 1.0 1 0 1 0	
1 1.0 0 0 0 1	$+[-Y]_{补}$
0 0.0 1 0 1 1	商 1，$\xleftarrow{1}$
0 0.1 0 1 1 0	
1 1.0 0 0 0 1	$+[-Y]_{补}$
1 1.1 0 1 1 1	商 0
0 0.1 1 1 1 1	$+[Y]_{补}$（恢复余数）
0 0.1 0 1 1 0	商的符号＝1＋0＝1

$|商|=0.10110$，商＝1.10110（原码）

$|余数|=0.10110\times 2^{-5}$，余数＝$1.10110\times 2^{-5}$（原码）

41. 叙述定点补码一位除法的运算规则。设 $X=0.10110$，$Y=-0.11111$，用补码一位除法求 X/Y 之商。

答：在被除数的绝对值小于除数的绝对值（即商不溢出）的情况下，补码一位除法的运算规则如下：

(1) 如果被除数与除数同号，用被除数减去除数；如果两数异号，用被除数加上除数。如果所得余数与除数同号，上商 1；如果余数与除数异号，上商 0。该商即为结果的符号位。

(2) 求商的数值部分。

如果上次上商 1，将余数左移一位后减去除数；如果上次上商 0，将余数左移一位后加上除数。然后判断本次操作后的余数，如果余数与除数同号，上商 1；如果余数与除数异号，上商 0。设数值部分有 n 位，重复执行 $n-1$ 次。

(3) 商的最后一位一般采用恒置 1 的办法，此时最大误差为 $\pm 2^{-n}$。如果对商的精度要求较高，则可按(2)的规则再进行一次操作，以求得商的第 n 位。当除不尽时，若商为负，要在商的最低位加 1，使商从反码值转变为补码值；若商为正，最低位不需要加 1。

设 $X=0.10110$，$Y=-0.11111$　用补码一位除法求 X/Y 之商的方法如下：

$$[Y]_{补}=1.00001 \quad [-Y]_{补}=0.11111$$

被除数或余数	说明
0 0.1 0 1 1 0	被除数
1 1.0 0 0 0 1	$+[Y]_{补}$
1 1.1 0 1 1 1	商 1，$\leftarrow$ 1
1 1.0 1 1 1 0	
0 0.1 1 1 1 1	$+[-Y]_{补}$
0 0.0 1 1 0 1	商 0，$\leftarrow$ 1
0 0.1 1 0 1 0	
1 1.0 0 0 0 1	$+[Y]_{补}$
1 1.1 1 0 1 1	商 1，$\leftarrow$ 1
1 1.1 0 1 1 0	
0 0.1 1 1 1 1	$+[-Y]_{补}$
0 0.1 0 1 0 1	商 0，$\leftarrow$ 1
0 1.0 1 0 1 0	
1 1.0 0 0 0 1	$+[Y]_{补}$
0 0.0 1 0 1 1	商 0，$\leftarrow$ 1
0 0.1 0 1 1 0	
1 1.0 0 0 0 1	$+[Y]_{补}$
1 1.1 0 1 1 1	商 1

商(反码)=1.01001　商(补码)=1.01010　商=−0.10110

另有一种方法，最后一次 $+[Y]_{补}$ 不进行，而采取商的最低位恒置“1”的方法，此时商(补码)=1.01001。

42. 定点数的加减法运算如何判断溢出？

答：定点数加减法运算一律转换成补码加法运算处理：

$$[X-Y]_{补}=[X]_{补}+[-Y]_{补}$$

$$[X+Y]_{补}=[X]_{补}+[Y]_{补}$$

补码加法运算判断溢出的条件如下：

(1) 符号相同两数相加，结果符号与加数(或被加数)的符号不同，为溢出。

(2) 任意符号两数相加，如果数值部分最高位的进位与符号位的进位不同，为溢出。

(3) 采用双符号位 $f_{s2}f_{s1}$，即正数的符号位为 00，负数的符号位为 11，符号位参与运算，相加结果的两个符号位不同 $f_{s2}\neq f_{s1}$，为溢出，f_{s2} 为第 2 符号位，f_{s1} 为第 1 符号位。

例：$X=0.1011$，$Y=0.0110$，$X+Y$ 的 3 种判断溢出的方法如下：

(1)
$$\begin{array}{r} 0.1011 \\ +0.0110 \\ \hline 1.0001 \end{array}$$
结果的符号与加数不同

(2)
$$\begin{array}{r} 0.1011 \\ +0.0110 \\ \hline 01.0001 \end{array}$$
←← 无有进进位位

(3)
$$\begin{array}{r} 00.1011 \\ +00.0110 \\ \hline 01.0001 \end{array}$$
双符号位不同 ($f_{s2}=0$，$f_{s1}=1$)

上例说明：满足 3 个条件中的任意 1 条，均可判断为溢出。由双符号位表示的补码又称变形补码。

43. 如何判断定点数乘/除法运算的结果是否溢出？

答：定点原码乘法运算的结果不会溢出，因为两个绝对值＜1 的定点小数相乘不会≥1，两个 n 位整数相乘其结果不会超过 $2n$ 位(结果取双倍字长)。补码运算仅当 $X=-1$ 且 $Y=-1$ 时(小数)或 $X=-2^n$ 且 $Y=-2^n$ 时(整数)才会产生溢出。

除法运算先要比较两数的大小，要求被除数的绝对值小于除数，否则为溢出。

补码乘法运算举例如下。

例 1：$[X]_{补}=1.0000$，$[Y]_{补}=1.0000$，求 $X\cdot Y$。

答：$[-X]_{补}=01.0000$

部分积	乘数	说明
00.0000	0000	
00.0000	0000	+0，→1
00.0000	0000	+0，→1
00.0000	0000	+0，→1
00.0000	0000	+0，→1
01.0000		$+[-X]_{补}$
01.0000	0000	

$X\cdot Y=01.0000\ 0000=(+1)_{10}$　溢出

例 2：$[X]_{补}=1.0000$，$[Y]_{补}=1.0001$，求 $X\cdot Y$。

答：$[X]_{补}=11.0000$　$[-X]_{补}=01.0000$

部分积	乘数	说明
00.0000	0001	
11.0000		$+[X]_{补}$
11.0000		→1
11.1000	0000	
11.1100	0000	+0，→1
11.1110	0000	+0，→1
11.1111	0000	+0，→1
01.0000		$+[-X]_{补}$
00.1111	0000	

$X\cdot Y=0.11110000$　不溢出

44. 将数据$(123)_{10}$转换成规格化浮点数。已知阶码 5 位(含符号位),基数为 2 和 8 两种情况,采用移码;尾数 9 位(含符号位),采用补码。

答:$(123)_{10}=(1111011)_2$

当基数为 2 时,有

$(123)_{10}=0.1111011\times 2^7$

阶码$=(7)_{10}=(10111)_2$(移码),尾数$=0.11110110$

当基数为 8 时,有

$(123)_{10}=(0.001111011)_2\times 2^9=(0.001111011)_2\times 8^3$

阶码$=(3)_{10}=(10011)_2$(移码),尾数只能保留 9 位,要将最低位舍去,今假设采用 0 舍 1 入法,则得尾数$(0.00111110)_2$。

45. 浮点数阶码 4 位(含符号位),尾数 9 位(含符号位)均采用补码,求数值范围。基数为 2 和 8 两种情况。哪种情况能表示的数值范围大,精度高?

答:基数为 2 时,最大阶码为 0111,最大正数为$2^7(1-2^{-8})$,最小负数为$2^7(-1)$,数值范围为$-2^7\sim 2^7(1-2^{-8})$。

基数为 8 时,最大阶码为 0111,最大正数为$8^7(1-2^{-8})$,最小负数为$8^7(-1)$,数值范围为$-8^7\sim 8^7(1-2^{-8})$。

基数为 8 时,表示的数值范围大。精度取决于尾数的长度,两者是相同的。

46. 今有以下两浮点数(阶码和尾数都用补码表示)

X:阶码 0000　尾数 0.10000111

Y:阶码 1111　尾数 0.11100100

求 $X-Y$ 之值。

答:(1) 对阶:

$[-Y_{阶}]_{补}=0001$　$[X_{阶}-Y_{阶}]_{补}=0000+0001=0001$(阶差)

取大阶 0000,$Y_{尾}$ 右移 1 位,得 0.01110010。

(2) 尾数相减,有

$[-Y_{尾}]_{补}=1.10001110$

$[X_{尾}-Y_{尾}]_{补}=0.10000111+1.10001110=0.00010101$

(3) 向左规格化:尾数左移 3 位,得 0.10101000,阶码-3,$[阶码]_{补}=0000+1101=1101$(-3 的二进制补码为 1101)。

(4) 舍入:不需要。

结果:阶码为 1101,尾数为 0.10101000,均为补码。

47. 今有以下两浮点数(阶码用移码表示,尾数用补码表示):

X:阶码 1111　尾数 0.10000111

Y:阶码 1111　尾数 0.11100100

求 $X+Y$ 之值。

答:$[-Y_{阶}]_{补}=1001$

(1) 求阶差:

$[X_{阶}-Y_{阶}]_{移}=[X_{阶}]_{移}+[-Y_{阶}]_{补}=1111+1001=1000$

阶差为 0,取阶值 1111。

(2) 尾数相加,有

$0.10000111+0.11100100=1.01101011$

(3) 向右规格化：尾数右移 1 位，得 0.101101011，阶码＋1，$[阶码]_{移}=1111+0001$，阶码上溢，运算中止，发溢出中断。

48. 今有以下两浮点数(阶码用移码表示，尾数用补码表示)：

X：阶码 0010　尾数 0.10000111

Y：阶码 0001　尾数 0.11100100

求 $X-Y$ 之值。

答：$[-Y_{阶}]_{补}=0111$

(1) 求阶差：

$[X_{阶}]_{移}+[-Y_{阶}]_{补}=0010+0111=1001$

取大阶 0010，$Y_{尾}$ 右移 1 位得 0.01110010。

(2) 尾数相减：

$[-Y_{尾}]_{补}=1.10001110$

$[X_{尾}-Y_{尾}]_{补}=0.10000111+1.10001110=0.00010101$

(3) 规格化：尾数左移 3 位，得 0.10101000，阶码－3，$[阶码]_{移}=0010+1101=1111$，溢出(下溢)。

(4) 将阶码和尾数置全 0(机器 0)。

49. 一个 IEEE 754 标准单精度浮点数如下：

0000　1111　1110　1111　1100　0000　0000　1111

请说明阶码和尾数各为多少？

答：单精度浮点数字长 32 位，阶码用增码(即移码)表示，尾数用原码表示。设最高位为第 31 位，最低位为第 0 位，则：

第 31 位为尾数符号位 0。

第 30～23 位为阶码 00011111。

第 22～0 位为尾数，在尾数前还有 1 位隐藏位，所以实际的尾数在最高位前面补 1，为 1110 1111 1100 0000 0000 1111，共 24 位。

50. 设有两浮点数 X 和 Y，阶和尾数均以补码表示，已知：X 的阶码为 0010，尾数为 0.1001；Y 的阶码为 1101，尾数为 1.0111。求 $X\cdot Y$ 和 X/Y 之值。

答：(1) 求 $X\cdot Y$ 之值，步骤如下。

① 阶码相加：0010＋1101＝1111。

② 尾数相乘：$X_{尾}\cdot Y_{尾}=-0.01010001$，或$[X_{尾}\cdot Y_{尾}]_{补}=1.10101111$(乘法过程略)。

③ 向左规格化：左移 1 位，阶码－1。

乘积的阶码＝阶码－1＝1111＋1111＝1110(补码)。

乘积的尾数＝1.01011110(补码)。

④ 舍入(取 4 位结果)：1.0101＋0.0001＝1.0110(补码)。

结果：阶码 1110，尾数 1.0110，均以补码表示。

(2) 求 X/Y 之值，步骤如下。

① 阶码相减：0010＋0011＝0101($[-Y_{阶}]_{补}=0011$)。

② 尾数相除：采用原码 1 位除法(加减交替法)，$X_{尾}=0.1001$，$|Y_{尾}|=0.1001$，$[-|Y_{尾}|]_{补}=1.0111$，被除数尾数取两符号位。

被除数或余数	说明
0 0.1 0 0 1	初始值
1 1.0 1 1 1	$-Y$(即$+[-\|Y\|_{尾}]_{补}$)
0 0.0 0 0 0	商 1
0 0.0 0 0 0	$\leftarrow\!\!\underline{1}$(左移 1 位)
1 1.0 1 1 1	$-Y$
1 1.0 1 1 1	商 0
1 0.1 1 1 0	$\leftarrow\!\!\underline{1}$
0 0.1 0 0 1	$+Y$
1 1.0 1 1 1	商 0
1 0.1 1 1 0	$\leftarrow\!\!\underline{1}$
0 0.1 0 0 1	$+Y$
1 1.0 1 1 1	商 0
1 0.1 1 1 0	$\leftarrow\!\!\underline{1}$
0 0.1 0 0 1	$+Y$
1 1.0 1 1 1	商 0
	$\|商_{尾}\|=1.0000$

尾数符号$=0+1=1$

$[尾数]_{补}=[-|商_{尾}|]_{补}=1.0000$

不需要进行规格化和舍入操作。

结果：阶码$=0101$，$[尾数]_{补}=1.0000$。

51. 浮点数的阶码为什么经常采用移码？

答：因为移码便于比较(正数符号位为 1，负数的符号位为 0，编码大的数其值也大)，并便于实现加减运算。移码的最小值正好是机器 0。

52. 规格化浮点数阶码有 P 位，尾数有 M 位，各自包含 1 符号位，且都用补码表示，该数的基数 r 为 2，请说出用这种格式能表示的全部不同的规格化数的总个数。当 $r=8$ 时，又能表示多少个规格化数(不考虑机器 0)。

答：对于同一阶码，尾数能表示的个数为 2^M 个，当 $r=2$ 时，能表示的规格化数为 $2^M\times\frac{1}{2}=2^{M-1}$。阶码的个数为 2^P，所以能表示的全部规格化浮点数为 $2^P\cdot 2^{M-1}=2^{P+M-1}$ 个。

当 $r=8$ 时，能表示的浮点数为 $2^P\cdot 2^M\cdot\frac{7}{8}=\frac{7}{8}\times 2^{P+M}$ 个。

53. 在计算机内部，浮点数相加是否满足加法结合律，即 $x+(y+z)=(x+y)+z$？请说明理由。

答：在计算机内部，由于浮点运算存在精度问题(由规格化及舍入引起的)，所以不满足加法的结合律，运算结果为非精确值，存在误差。今举例如下：

设 $x=2^{-4}\times 0.1000$，$y=2^{-4}\times 0.1001$，$z=2^0\times 0.1000$，尾数保留 4 位。

(1) 采用截断舍入法。

$$x+(y+z)=2^{-4}\times 0.1000+(2^{-4}\times 0.1001+2^0\times 0.1000)$$

$$=2^{-4}\times 0.1000+2^{0}\times 0.1000$$
$$=2^{0}\times 0.1000$$
$$(x+y)+z=(2^{-4}\times 0.1000+2^{-4}\times 0.1001)+2^{0}\times 0.1000$$
$$=2^{-3}\times 0.1000+2^{0}\times 0.1000$$
$$=2^{0}\times 0.1001$$

两者不相等，相差 2^{-4}。

(2) 采用 0 舍 1 入法。

$$x+(y+z)=2^{-4}\times 0.1000+(2^{-4}\times 0.1001+2^{0}\times 0.1000)$$
$$=2^{-4}\times 0.1000+2^{0}\times 0.1001$$
$$=2^{0}\times 0.1010$$
$$(x+y)+z=(2^{-4}\times 0.1000+2^{-4}\times 0.1001)+2^{0}\times 0.1000$$
$$=2^{-3}\times 0.1001+2^{0}\times 0.1000$$
$$=2^{0}\times 0.1001$$

两者不相等，相差 2^{-4}。

54. 在计算机中，经常采用的数据校验码是什么？一般使用在什么场合？

答：经常采用的数据校验码是奇偶校验码、海明校验码和循环冗余校验码(CRC)。一般在传送数据时为每一字节数据配以一个奇(或偶)校验位。主存储器为每一字配以海明校验码。CRC 一般用于磁盘、磁带等存储器中。

55. 今有 4 个数：00001111、11110000、00000000 和 11111111，问：

(1) 其码距为多少？能纠正和(或)发现多少位错，如果出现数据 00011111 应纠正成什么数？如何纠正？

(2) 如果再加上两个数 00010000、11001111(共 6 个数)，其码距为多少？能纠正和(或)发现多少位错？

(3) 如果为上述各数加上奇校验位，求奇校验位之值。

答：(1) 4 个数之间有 4 位不同，故码距为 4，能纠正 1 位错和发现两位错。如果出现数据 00011111，应纠正成 00001111，纠正的办法是取出错位的反码。

(2) 码距是根据数据中任意两个合法码之间最少有几位二进制位不同而确定的。由于 00000000 和 00010000 仅有 1 位不同，故码距为 1，假如数据 00010000 出错而成为 00000000 时，由于后者仍为合法码，所以不能发现和纠正该错误。

(3) 除了 00010000 的奇校验位为 0 外，其余数据的奇校验位均为 1。

56. 设待校验的数据 $D_8\sim D_1=10101011$，其最高位为 D_8，若采用能纠正 1 位错的海明校验码，求出其校验码。如何纠错？

答：根据公式 $2^r\geqslant k+r+1$，今数据位 $k=8$，所以校验位 $r=4$，设 4 位校验位为 $P_1\sim P_4$ 并采用偶校验。8 位数据位为：$\begin{matrix}D_8&D_7&D_6&D_5&D_4&D_3&D_2&D_1\\1&0&1&0&1&0&1&1\end{matrix}$，则：

$$P_1=D_1\oplus D_2\oplus D_4\oplus D_5\oplus D_7=1+1+1+0+0=1$$
$$P_2=D_1\oplus D_3\oplus D_4\oplus D_6\oplus D_7=1+0+1+1+0=1$$
$$P_3=D_2\oplus D_3\oplus D_4\oplus D_8=1+0+1+1=1$$
$$P_4=D_5\oplus D_6\oplus D_7\oplus D_8=0+1+0+1=0$$

纠错时，将 8 位数据码和 4 位校验码组成的汉明码按下列顺序排列：

$$H_{12}\,H_{11}\,H_{10}\,H_9\,H_8\,H_7\,H_6\,H_5\,H_4\,H_3\,H_2\,H_1$$

$$D_8\;\;D_7\;\;D_6\;\;D_5\;P_4\;D_4\;D_3\;D_2\;P_3\;D_1\;P_2\;P_1$$

P_1、P_2、P_3 和 P_4 各处于 H_1、H_2、H_4 和 H_8 位置(2 的幂次)。检查 $P_1\sim P_4$ 之值，如果上述公式均成立，说明无错。如果仅 P_1 公式有误，说明检验位 P_1 出错；同样，如果仅是 P_2、P_3 或 P_4 公式之一有误，说明是检验位 P_2、P_3 或 P_4 之一有误。假如有一个以上公式有误，说明数据位有误，出现错误的位置由 H 的下标相加而得到。例如，当 P_1 和 P_3 公式有误时，其对应位置的 H 下标为 1 和 4，所以出错的位置在 $H_5(1+4=5)$，即 D_2。检查 P 的公式，在 P_1 和 P_3 的公式中有 D_2，而在 P_2 和 P_4 公式中无 D_2，因此当 D_2 出错时，反映出公式 P_1 和 P_3 有误；同理，如果 P_1、P_2 和 P_4 有错，其对应位置的 H 下标为 1、2 和 8，所以出错的位置在 $H_{11}(1+2+8=11)$，即 D_7。

查出错误码后，取其反码，即得正确码。

57. 计算机准备传送的数据是$(100110)_2$，采用 CRC 校验码，生成多项式选用 $X^3+X^1+X^0$。

(1) 要求计算校验位。

(2) 如何判断数据是否出错？

答：(1) 计算校验位的步骤如下：

① 将数据表示成多项式 $M(x)$。

$$M(x)=x^5+x^2+x$$

② 将 $M(x)$乘以 x^3，目的是空出 3 位，以便拼装 3 位校验码。

$$M(x)\cdot x^3=x^8+x^5+x^4$$

③ 用 $G(x)$对 $M(x)\cdot x^3$ 进行模 2 除。$G(x)$为生成多项式。

$$\frac{M(x)x^3}{G(x)}=\frac{x^8+x^5+x^4}{x^3+x^1+x^0}=x^5+x^3+1+\frac{x+1}{x^3+x+1}$$

余数多项式 $R(x)=x+1$。

④ 将 $M(x)x^3$ 与 $R(x)$相加得 CRC 多项式 $x^8+x^5+x^4+x+1$。

CRC 码为 100110011，其中信息码为 100110，校验码为 011。

(2) 判断数据是否出错。

将 CRC 多项式除以 $G(x)$(模 2 除)，如能除尽，表示无错误；如余数不为 0，说明有错，而且错误位不同，余数也不相同，所以不是任意一个多项式都能用做生成多项式。

$$\frac{\text{CRC 多项式}}{G(x)\text{多项式}}=\frac{x^8+x^5+x^4+x+1}{x^3+x+1}=\frac{100110011}{1011}=101001\text{，余数为 }0。$$

运算过程如下：

```
                1 0 1 0 0 1
          ┌──────────────────
1 0 1 1 ) 1 0 0 1 1 0 0 1 1
          1 0 1 1
          ─────────
            0 1 0 1 0
              1 0 1 1
            ─────────
              0 0 0 1 0 1 1
                    1 0 1 1
                  ─────────
                    0 0 0 0
```

除尽，无错误。

58. 在运算器中，完成 n 位加减法运算一般需要哪些逻辑电路？

答：一般遵循补码运算规则进行加减法运算。需要下列逻辑电路。

(1) 两个寄存器。用于存放两个源操作数，其中一个还用于保存运算结果。寄存器用 n 位触发器组成。

(2) n 位加法器。如果要判别带符号的数的运算是否发生溢出情况，则需要增加判别溢出的电路或增加 1 位加法器；如果数的长度超过 n 位(例如 $2n$ 位)，则需进行 2 次运算，其中低位运算可能会向高位产生进位或借位信号。为此在运算器中一般设置有溢出触发器 V 和进位/借位触发器 C，称为条件码或状态码。

(3) 传送数据的门电路。包括寄存器向加法器传送原码或反码的门电路以及运算结果从加法器传送到寄存器的门电路。如有必要(例如将反码变为补码，或双字长运算低 n 位产生进位信号)则形成+1 信号送往 n 位加法器的最低位。

(4) 其他。一般在运算器中还设置 N 和 Z 两个条件码，当运算结果为负数时 $N=1$，否则为 0；当运算结果为零时，$Z=1$，否则 $Z=0$。

另外，在运算器中一般设置有通用寄存器，用于暂存参与运算的数据和运算结果。

59. 接上题，如果运算器增加乘除法运算功能，主要增加哪些逻辑电路？

答：增加一个寄存器，因为运算过程中需要保留部分乘积或余数；增加实现移位功能的电路；设置一个计数器控制运算过程中相加(减)和移位的次数。其他还有一些完成某些功能的电路，例如用于求乘积或商的符号(正数/负数)、上商以及与加/减运算不同的判溢出逻辑电路等。

在设计运算器时，对所有的实现细节都要进行详细考虑。

60. 在以下的空格中，填上有关浮点数运算的内容。

完成浮点数运算的部件分成阶码运算和 A 两部分。加法运算一般按以下顺序执行：①对阶，② B ，③ C ，④舍入。在上述各阶段中，凡是涉及阶码运算内容时，都要判断中间结果或最终结果是否溢出，溢出分为 D 和下溢两种情况，当发生下溢时，将运算结果置为 E ；而发生 D 情况时，则按出错处理。

阶码一般用补码或 F 表示，尾数要维持规格化表示形式，以达到尽量保持数据的 G 。

答：A——尾数加、减、乘、除运算；B——尾数运算；C——规格化；D——上溢；E——机器零；F——移码；G—精度。