第六章 计算机的运算方法

作业：P290　6.2、6.3、6.4、6.5、6.6、6.7、6.8、6.9、6.10、6.11、6.12、6.13、6.14、6.15、6.16、6.17、6.18、6.19、6.20、6.21、6.26、6.32

6.2 已知 X=0.a1a2a3a4a5a6（ai 为 0 或 1），讨论下列几种情况时 ai 各取何值。

（1）$X > \frac{1}{2}$

（2）$X \geq \frac{1}{8}$

（3）$\frac{1}{4} \geq X > \frac{1}{16}$

解：　（1）若要 $X > \frac{1}{2}$，只要 a1=1，a2~a6 不全为 0 即可。

（2）若要 $X \geq \frac{1}{8}$，只要 a1~a3 不全为 0 即可。

（3）若要 $\frac{1}{4} \geq X > \frac{1}{16}$，只要 a1=0，a2 可任取 0 或 1；

当 a2=0 时，若 a3=0，则必须 a4=1，且 a5、a6 不全为 0；

若 a3=1，则 a4~a6 可任取 0 或 1；

当 a2=1 时，　a3~a6 均取 0。

6.3 设 x 为整数，$[x]_补=1,x_1x_2x_3x_4x_5$，若要求 x<-16，试问：x_1～x_5应取何值？

答：

方法 1：

从$[x]_补=1,x_1x_2x_3x_4x_5$可以看出，x 为整数，根据补码的定义，$[x]_补=2^6+x$,

故 $x=[x]_补-2^6=1,x_1x_2x_3x_4x_5-2^6=x_1x_2x_3x_4x_5-2^5<-16$

则 $x_1x_2x_3x_4x_5<16$, $x_1=0$, $x_2x_3x_4x_5$可为任意。

方法 2：

x =-15，$[x]_补=1,10001$

x =-16，$[x]_补=1,10000$

x =-17，$[x]_补=1,01111$

所以，若要求 x<-16，则 $x_1=0$, $x_2x_3x_4x_5$可为任意。

方法 3：

由于$[x]_补=1,x_1x_2x_3x_4x_5$，说明 x 为负数，则有：$x=-(\bar{x}_1\bar{x}_2\bar{x}_3\bar{x}_4\bar{x}_5+1)$

而 x<-16，$-\bar{x}_1\bar{x}_2\bar{x}_3\bar{x}_4\bar{x}_5<-15$，即：$\bar{x}_1\bar{x}_2\bar{x}_3\bar{x}_4\bar{x}_5>15$

则：只有$\bar{x}_1=1$，$\bar{x}_2$、$\bar{x}_3$、$\bar{x}_4$、$\bar{x}_5$为任意值时才成立

或只有 $x_1=0$, $x_2x_3x_4x_5$为任意值时才成立

6.4 设机器数字长 8 位（含 1 位符号为），写出对应下列各真值的原码、反码和补码。

-13/64，29/128，100，-87

答：

-13/64D=-0.0011010，编码取 8 位，则，

其原码为：1.0011010

其反码为：1.1100101

其补码为：1.1100110

29/128D=0.0011101，编码取 8 位，则，

其原码为：0.0011101

其反码为：0.0011101

其补码为：0.0011101

100D=1100100，编码取 8 位，则，

其原码为：01100100

其反码为：01100100

其补码为：01100100

-87D=-1010111，编码取 8 位，则，

其原码为：1，1010111

其反码为：1，0101000

其补码为：1，0101001

6.5 已知$[x]_{补}$，求$[x]_{原}$和 x。

$[x]_{补}$=1.1100；$[x]_{补}$=1.1001；$[x]_{补}$=0.1110；$[x]_{补}$=1.0000；

$[x]_{补}$=10101； $[x]_{补}$=11100； $[x]_{补}$=00111； $[x]_{补}$=10000。

答：

$[x]_{补}$=1.1100，x=-0.0100，$[x]_{原}$=1.0100

$[x]_{补}$=1.1001，x=-0.0111，$[x]_{原}$=1.0111

$[x]_{补}$=0.1110，x=0.1110， $[x]_{原}$=0.1110

$[x]_{补}$=1.0000，x=-1， $[x]_{原}$不存在（5 位编码时）

$[x]_{补}$=10101， x=-1011， $[x]_{原}$=11011

$[x]_{补}$=11100， x=-0100， $[x]_{原}$=10100

$[x]_{补}$=00111， x=0111， $[x]_{原}$=00111

$[x]_{补}$=10000， x=-16， $[x]_{原}$不存在（5 位编码时）

6.6 设机器数字长为 8 位（含 1 位符号位在内），分整数和小数两种情况讨论真值 x 为何值时，$[x]_{补}=[x]_{原}$成立。

解：

当 x 为小数时，若 $x\geq 0$，则 $[x]_{补}=[x]_{原}$成立；

若 $x < 0$，当 $x= -1/2$ 时，$[x]_{补}=[x]_{原}$=1.100 0000，则 $[x]_{补}=[x]_{原}$成立。

当 x 为整数时，若 $x\geq0$，则 $[x]_{补}=[x]_{原}$成立；

若 x< 0，当 x= -64 时，$[x]_{补}=[x]_{原}$=1,100 0000，则 $[x]_{补}=[x]_{原}$成立。

6.7 设 x 为真值，x*为绝对值，说明$[-x^*]_{补}=[-x]_{补}$能否成立。

解：

当 x 为真值，x*为绝对值时，$[-x^*]_{补}=[-x]_{补}$不能成立。原因如下：

（1）当 x<0 时，由于$[-x^*]_{补}$是一个负值，而$[-x]_{补}$是一个正值，因此此时$[-x^*]_{补}=[-x]_{补}$不成立；

（2）当 x≥0 时，由于-x*=-x，因此此时 $[-x^*]_{补}=[-x]_{补}$的结论成立。

6.8 讨论若$[x]_{补}>[y]_{补}$，是否有 x>y？

解：

若$[x]_{补}>[y]_{补}$，不一定有 x>y。[x]补 > [y]补时 x > y 的结论只在 x > 0 且 y > 0，及 x<0 且 y<0 时成立。

因正数补码的符号位为 0，负数补码的符号位为 1，当 x>0、 y<0 时，有 x>y，但则$[x]_{补}<[y]_{补}$；同样，当 x<0、 y >0 时，有 x < y，但$[x]_{补}>[y]_{补}$。

6.9 当十六进制数 9BH、FFH 分别表示原码、反码、补码、移码和无符号数时，所对应的十进制数各为多少（设机器数采用 1 位符号位）？

答：

$[x]_{移}$ = 2n + x（2n＞x ≥ 2n） 其中，x 为真值，n 为整数的位数

	9BH	FFH
十六进制	1001 1011	1111 1111
无符号数	155	255
原码	1，0011011 -27	1，1111111 -127
反码	1，1100100 -100	1，0000000 -0
补码	1，1100101 -101	1，0000001 -1
移码	1，0011011-128 +27	1，1111111-128 +127

6.10 在整数定点机中，设机器数采用 1 位符号位，写出±0 的原码、补码、反码和移码，得出什么结论？

解：

0 的机器数形式如下：（假定机器数共 8 位，含 1 位符号位在内）

真值	原码	补码	反码	移码
+0	0 000 0000	0 000 0000	0 000 0000	1 000 0000
-0	1 000 0000	0 000 0000	1 111 1111	1 000 0000

结论：0 的原码和反码分别有+0 和-0 两种形式，补码和移码只有一种形式，且补码和移码数值位相同，符号位相反。

6.11 已知机器数字长为 4 位（含 1 位符号位），写出整数定点机和小数定点机中原码、补码和反码的全部形式，并注明其对应的十进制真值。

整数定点机				小数定点机			
原码	补码	反码	真值	原码	补码	反码	真值
0,000	0,000	0,000	+0	0.000	0.000	0.000	+0
0,001	0,001	0,001	1	0.001	0.001	0.001	0.125
0,010	0,010	0,010	2	0.010	0.010	0.010	0.250
0,011	0,011	0,011	3	0.011	0.011	0.011	0.375
0,100	0,100	0,100	4	0.100	0.100	0.100	0.500
0,101	0,101	0,101	5	0.101	0.101	0.101	0.625
0,110	0,110	0,110	6	0.110	0.110	0.110	0.750
0,111	0,111	0,111	7	0.111	0.111	0.111	0.875
1,000	0,000	1,111	-0	1.000	0.000	1.111	-0
1,001	1,111	1,110	-1	1.001	1.111	1.110	-0.125
1,010	1,110	1,101	-2	1.010	1.110	1.101	-0.250
1,011	1,101	1,100	-3	1.011	1.101	1.100	-0.375
1,100	1,100	1,011	-4	1.100	1.100	1.011	-0.500
1,101	1,011	1,010	-5	1.101	1.011	1.010	-0.625
1,110	1,010	1,001	-6	1.110	1.010	1.001	-0.750
1,111	1,001	1,000	-7	1.111	1.001	1.000	-0.875
无	1,000	无	-8	无	1.000	无	-1

6.12 设浮点数格式为：阶码 5 位（含 1 位阶符），尾数 11 位（含 1 位数符）。写出 51/128、-27/1024、7.375、-86.5 所对应的机器数。要求如下：

（1）阶码、尾数均为原码。

（2）阶码、尾数均为补码。

（3）阶码为移码、尾数为补码。

答：

根据阶码 5 位（含 1 位阶符），尾数 11 位（含 1 位数符），

$51/128=0.0110011000=0.1100110000*2^{-1}$

其尾数规格化真值：0.1100110000

其阶码真值：-0001

（1）阶码、尾数均为原码：　　1，0001；　0.1100110000

（2）阶码、尾数均为补码：　　1，1111；　0.1100110000

（3）阶码为移码、尾数为补码：0，1111；　0.1100110000

$-27/1024=-0.0000011011=-0.1101100000*2^{-5}$

其尾数规格化真值：-0.1101100000

其阶码真值：-0101

（1）阶码、尾数均为原码：10101, 1.1101100000

（2）阶码、尾数均为补码：11011, 1.0010100000

（3）阶码为移码、尾数为补码：01011, 1.0010100000

7.375=111.011=0.111011*2^3

其尾数规格化真值：0.1110110000

其阶码真值：0011

（1）阶码、尾数均为原码：00011，0.1110110000

（2）阶码、尾数均为补码：00011，0.1110110000

（3）阶码为移码、尾数为补码：10011，0.1110110000

-86.5=-1010110.1=-0.1010110100*2^7

其尾数规格化真值：-0.1010110100

其阶码真值：+0111

（1）阶码、尾数均为原码：　0，0111；　1.1010110100

（2）阶码、尾数均为补码：　0，0111；　1.0101001100

（3）阶码为移码、尾数为补码：1，0111；　1.0101001100

6.13 浮点数格式同上题，当阶码基值分别取 2 和 16 时：

（1）说明 2 和 16 在浮点数中如何表示。

（2）基值不同对浮点数什么有影响？

（3）当阶码和尾数均用补码表示，且尾数采用规格化形式，给出两种情况下所能表示的最大正数和非零最小正数真值。

解：

（1）阶码基值不论取何值，在浮点数中均为隐含表示，即：2 和 16 不出现在浮点格式中，仅为人为的约定。

（2）当基值不同时，对数的表示范围和精度都有影响。即：在浮点格式不变的情况下，基越大，可表示的浮点数范围越大，但浮点数精度越低。

（3）r=2 时，

最大正数的浮点格式为：0，1111；0.111 111 111 1

其真值为：$N_{+max}=2^{15}\times(1-2^{-10})$

非零最小规格化正数浮点格式为：1，0000；0.100 000 000 0

其真值为：$N_{+min}=2^{-16}\times2^{-1}=2^{-17}$

r=16 时，

最大正数的浮点格式为：0，1111；0.1111 1111 11

其真值为：$N_{+max}=16^{15}\times(1-2^{-10})$

非零最小规格化正数浮点格式为：1，0000；0.0001 0000 00

其真值为：$N_{+min}=16^{-16}\times16^{-1}=16^{-17}$

6.14 设浮点数字长为 32 位，欲表示±6 万间的十进制数，在保证数的最大精度条件下，除阶符、数符各取 1 位外，阶码和尾数各取几位？按这样分配，该浮点数溢出的条件是什么？

解：

若要保证数的最大精度，应取阶码的基值=2。

若要表示±6 万间的十进制数，由于 32768（2^{15}）< 6 万 <65536（2^{16}），则：阶码除阶符外还应取 5 位（向上取 2 的幂）。

故：尾数位数=32-1-1-5=25 位

25（32）　　该浮点数格式如下：

阶符（1 位）	阶码（5 位）	数符（1 位）	尾数（25 位）

按此格式，该浮点数上溢的条件为：阶码$\geq 2^5$

6.15 什么是机器零？若要求全 0 表示机器零，浮点数的阶码和尾数应采取什么机器数形式？

解：

机器零指机器数所表示的零的形式，它与真值零的区别是：机器零在数轴上表示为“0”点及其附近的一段区域，即在计算机中小到机器数的精度达不到的数均视为“机器零”，而真零对应数轴上的一点（0 点）。若要求用“全 0”表示浮点机器零，则浮点数的阶码应用移码、尾数用补码表示（此时阶码为最小阶、尾数为零，而移码的最小码值正好为“0”，补码的零的形式也为“0”，拼起来正好为一串 0 的形式）。

6.16 设机器数字长为 16 位，写出下列各种情况下它能表示的数的范围。设机器数采用一位符号位，答案均用十进制表示。

（1）无符号数；

（2）原码表示的定点小数。

（3）补码表示的定点小数。

（4）补码表示的定点整数。

（5）原码表示的定点整数。

（6）浮点数的格式为：阶码 6 位（含 1 位阶符），尾数 10 位（含 1 位数符）。分别写出其正数和负数的表示范围。

（7）浮点数格式同（6），机器数采用补码规格化形式，分别写出其对应的正数和负数的真值范围。

解：（1）无符号整数：$0 \sim 2^{16}-1$，即：0~ 65535；

无符号小数：$0 \sim 1-2^{-16}$ ，即：0 ~ 0.99998；

（2）原码定点小数：$-1+2^{-15} \sim 1-2^{-15}$ ，即：-0.99997~0.99997

（3）补码定点小数：$-1 \sim 1-2^{-15}$ ，即：-1~0.99997

（4）补码定点整数：$-2^{15} \sim 2^{15}-1$ ，即：-32768~32767

（5）原码定点整数：$-2^{15}+1 \sim 2^{15}-1$，即：-32767~32767

（6）据题意画出该浮点数格式，当阶码和尾数均采用原码，非规格化数表示时：

最大负数= 1，11 111；1.000 000 001 ，即 $-2^{-9}\times 2^{-31}$

最小负数= 0，11 111；1.111 111 111，即 $-(1-2^{-9})\times 2^{31}$

则负数表示范围为：$-(1-2^{-9})\times 2^{31}$ —— $-2^{-9}\times 2^{-31}$

最大正数= 0，11 111；0.111 111 111，即 $(1-2^{-9})\times 2^{31}$

最小正数= 1，11 111；0.000 000 001，即 $2^{-9}\times 2^{-31}$

则正数表示范围为：$2^{-9}\times 2^{-31}$ —— $(1-2^{-9})\times 2^{31}$

（7）当机器数采用补码规格化形式时，若不考虑隐藏位，则

最大负数=1，00 000；1.011 111 111，即 $-2^{-1}\times 2^{-32}$

最小负数=0，11 111；1.000 000 000，即 -1×2^{31}

则负数表示范围为：-1×2^{31} —— $-2^{-1}\times 2^{-32}$

最大正数=0，11 111；0.111 111 111，即 $(1-2^{-9})\times 2^{31}$

最小正数=1，00 000；0.100 000 000，即 $2^{-1}\times 2^{-32}$

则正数表示范围为：$2^{-1}\times2^{-32}$ —— （$1-2^{-9}$）$\times2^{31}$

6.17 设机器数字长 8 位（含 1 位符号为），对下列各机器数进行算术左移一位、两位，算术右移一位、两位，讨论结果是否正确。

$[x]_{原}$=0.0011010；　$[x]_{补}$=0.1010100；　$[x]_{反}$=1.0101111；

$[x]_{原}$=1.1101000；　$[x]_{补}$=1.1101000；　$[x]_{反}$=1.1101000；

$[x]_{原}$=1.0011001；　$[x]_{补}$=1.0011001；　$[x]_{反}$=1.0011001；

答：

算术左移一位：

$[x1]_{原}$=0.011 0100；正确

$[x2]_{原}$=1.101 0000；溢出（丢 1）出错

$[x3]_{原}$=1.011 0010；正确

$[y1]_{补}$=0.010 1000；溢出（丢 1）出错

$[y2]_{补}$=1.101 0000；正确

$[y3]_{补}$=1.011 0010；溢出（丢 0）出错

$[z1]_{反}$=1.101 1111；溢出（丢 0）出错

$[z2]_{反}$=1.101 0001；正确

$[z3]_{反}$=1.011 0011；溢出（丢 0）出错

算术左移两位：

$[x1]_{原}$=0.110 1000；正确

$[x2]_{原}$=1.010 0000；溢出（丢 11）出错

$[x3]_{原}$=1.110 0100；正确

$[y1]_{补}$=0.101 0000；溢出（丢 10）出错

$[y2]_{补}$=1.010 0000；正确

$[y3]_{补}$=1.110 0100；溢出（丢 00）出错

$[z1]_{反}$=1.011 1111；溢出（丢 01）出错

$[z2]_{反}$=1.010 0011；正确

$[z3]_{反}$=1.110 0111；溢出（丢 00）出错

算术右移一位：

$[x1]_{原}$=0.000 1101；正确

$[x2]_{原}$=1.011 0100；正确

$[x3]_{原}$=1.000 1100(1)；丢 1，产生误差

$[y1]_{补}$=0.010 1010；正确

$[y2]_{补}$=1.111 0100；正确

$[y3]_{补}$=1.100 1100(1)；丢 1，产生误差

$[z1]_{反}$=1.101 0111；正确

$[z2]_{反}$=1.111 0100(0)；丢 0，产生误差

$[z3]_{反}$=1.100 1100；正确

算术右移两位：

$[x1]_{原}$=0.000 0110（10）；产生误差

$[x2]_{原}$=1.001 1010；正确

$[x3]_{原}$=1.000 0110（01）；产生误差

$[y1]_{补}$=0.001 0101；正确

$[y2]_{补}$=1.111 1010；正确

$[y3]_{补}$=1.110 0110（01）；产生误差

$[z1]_{反}$=1.110 1011；正确

$[z2]_{反}$=1.111 1010（00）；产生误差

$[z3]_{反}$=1.110 0110（01）；产生误差

6.18 试比较逻辑移位和算术移位。

解：逻辑移位和算术移位的区别：

逻辑移位是对逻辑数或无符号数进行的移位，其特点是不论左移还是右移，空出位均补 0，移位时不考虑符号位。

算术移位是对带符号数进行的移位操作，其关键规则是移位时符号位保持不变，空出位的补入值与数的正负、移位方向、采用的码制等有关。补码或反码右移时具有符号延伸特性。左移时可能产生溢出错误，右移时可能丢失精度。

6.19 设机器数字长 8 位（含 1 位符号为），用补码运算规则计算下列各题。

（1）A=9/64，B=-13/32，求 A+B。

（2）A=19/32，B=-17/128，求 A-B。

（3）A=-3/16，B=9/32，求 A+B。

（4）A=-87，B=53，求 A-B。

（5）A=115，B=-24，求 A+B。

答：

（1）A=9/64，B=-13/32，求 A+B。

A=9/64=0.001001，$[A]_{补}$ = 0.0010010

B=-13/32=-0.01101，$[B]_{补}$= 1.1001100

$[A+B]_{补}$=$[A]_{补}$+$[B]_{补}$= 0.0010010+1.1001100=1.1011110， A+B=-0.0100010=-17/64

（2）A=19/32，B=-17/128，求 A-B。

A=19/32=0.1001100，$[A]_{补}$= 0.1001100

B=-17/128=-0.0010001，$[B]_{补}$=1.1101111，$[-B]_{补}$=0.0010001

$[A-B]_{补}$= $[A]_{补}$+ $[-B]_{补}$=0.1001100+0.0010001=0.1011101， A-B=0.1011101=93/128

（3）A=-3/16，B=9/32，求 A+B。

A=-3/16=-0.0011000，$[A]_{补}$=1.1101000

B=9/32=0.0100100，$[B]_{补}$= 0.0100100

$[A+B]_{补}$=$[A]_{补}$+$[B]_{补}$ = 1.1101000+0.0100100=0.0001100， A+B=0.0001100=3/32

（4）A=-87，B=53，求 A-B。

A=-87D=-1010111B，$[A]_{补}$=1，0101001

B=53D=0110101B，$[B]_{补}$=0，0110101，$[-B]_{补}$=1，1001011

$[A-B]_{补}$= $[A]_{补}$+ $[-B]_{补}$=1，0101001+1，1001011=0，1110100，实际结果应为负数，但机器运行结果为正数，溢出。原因是：-87-53=-140，超出 8 位补码最大表示范围（-128）.

（5）A=115，B=-24，求 A+B。

A=115D=1110011B，$[A]_{补}$=0，1110011

B=-24D=-0011000B，$[B]_{补}$=1，1101000

$[A+B]_{补}=[A]_{补}+[B]_{补}$=0，1110011+1，1101000=0，1011011

6.20 用原码一位乘、两位乘和补码一位乘（Booth 法）、两位乘计算 x*y。（只完成补码一位乘）

（1）x=0.110111，y=-0.101110。（2）x=-0.010111，y=-0.010101。

（3）x=19，y=35。（4）x=0.11011，y=-0.11101。

答：（1）x=0.110111，y=-0.101110。

$[x]_{补}$= 0.110111，$[-x]_{补}$ =1.001001

$[y]_{补}$=1.010010，$[y]_{补}$再增加一个附加位得，$[y]_{补}$= 1.0100100

部分积	乘数	附加位	说明
00.000000	*1010010*	*0*	初值$[z_0]_{补}$=0，$y_n y_{n+1}$=00，算术右移 1 位
00.000000	0*101001*	*0*	得$[z_1]_{补}$
+11.001001			$y_n y_{n+1}$=10，部分积加$[-x]_{补}$
11.001001	0*101001*	*0*	
11.100100	10*10100*	*1*	右移 1 位，得$[z_2]_{补}$
+00.110111			$y_n y_{n+1}$=01，部分积加$[x]_{补}$
00.011011	10*10100*	*1*	
00.001101	110*1010*	*0*	右移 1 位，得$[z_3]_{补}$，$y_n y_{n+1}$=00，右移 1 位
00.000110	1110*101*	*0*	得$[z_4]_{补}$
00.000110	1110*101*	*0*	$y_n y_{n+1}$=10，部分积加$[-x]_{补}$
+11.001001			
11.001111			
11.100111	11110*10*	*1*	右移 1 位，得$[z_5]_{补}$
+00.110111			$y_n y_{n+1}$=01，部分积加$[x]_{补}$
00.011110			
00.001111	011110*1*	*0*	右移 1 位，得$[z_6]_{补}$
+11.001001			$y_n y_{n+1}$=10，部分积加$[-x]_{补}$
11.011000	011110		最后一步不移位，得$[x*y]_{补}$

即$[x*y]_{补}$= 11.011000011110　　x*y=-0.100111100010

（2）x=-0.010111，y=-0.010101。

$[x]_{补}$=1.101001，$[-x]_{补}$=0.010111

$[y]_{补}$=1.101011，$[y]_{补}$再增加一个附加位得，$[y]_{补}$=1.1010110

部分积	乘数	附加位	说明
00.000000	1101011	0	初值$[z_0]_{补}$=0，y_ny_{n+1}=10
+00.010111			部分积加$[-x]_{补}$
00.010111			左侧结果右移 1 位
00.001011	1110101	1	得$[z_1]_{补}$，y_ny_{n+1}=11，算术右移 1 位
00.000101	1111010	1	得$[z_2]_{补}$，y_ny_{n+1}=01
+11.101001			部分积加$[x]_{补}$
11.101110			左侧结果右移 1 位
11.110111	0111101	0	得$[z_3]_{补}$，y_ny_{n+1}=10
+00.010111			部分积加$[-x]_{补}$
00.001110			左侧结果右移 1 位
00.000111	0011110	1	得$[z_4]_{补}$，y_ny_{n+1}=01
+11.101001			部分积加$[x]_{补}$
11.110000			左侧结果右移 1 位
11.111000	0001111	0	得$[z_5]_{补}$，y_ny_{n+1}=10
+00.010111			部分积加$[-x]_{补}$
00.001111			左侧结果右移 1 位
00.000111	1000111	1	得$[z_6]_{补}$，y_ny_{n+1}=11
			最后一步不移位，得$[x*y]_{补}$

即$[x*y]_{补}$= 0.000111100011，　x*y=0.000111100011

（3）x=19=010011，y=35=100011。

$[x]_{\text{补}}$= 00,010011，$[-x]_{\text{补}}$=11,101101

$[y]_{\text{补}}$= 00,100011，$[y]_{\text{补}}$再增加一个附加位得，$[y]_{\text{补}}$= 00,1000110

部分积	乘数	附加位	说明
00,000000	*0100011*	*0*	初值$[z_0]_{\text{补}}$=0，y_ny_{n+1}=10
+11,101101			部分积加$[-x]_{\text{补}}$，
11,101101	*0100011*	*0*	
11,110110	1*010001*	*1*	算术右移 1 位得$[z_1]_{\text{补}}$，
11,111011	01*01000*	*1*	y_ny_{n+1}=11，右移 1 位,得$[z_2]_{\text{补}}$
+00,010011			y_ny_{n+1}=01，部分积加$[x]_{\text{补}}$
00,001110	01*01000*	*1*	
00,000111	001*0100*	*0*	右移 1 位，得$[z_3]_{\text{补}}$，y_ny_{n+1}=00
00,000011	1001*010*	*0*	右移 1 位，得$[z_4]_{\text{补}}$，y_ny_{n+1}=00
00.000001	1100*101*	*0*	
+11,101101			y_ny_{n+1}=10，部分积加$[-x]_{\text{补}}$
11,101110	1100*101*	*0*	
11,110111	011001*0*	*1*	右移 1 位，得$[z_5]_{\text{补}}$，
+00,010011			y_ny_{n+1}=01,部分积加$[x]_{\text{补}}$
00,001010	011001*0*	*1*	得$[z_6]_{\text{补}}$,最后一步不移位，得$[x*y]_{\text{补}}$

即$[x*y]_{\text{补}}$= 00,001010011001　　x*y=001010011001

（4）x=0.11011，y=-0.11101

[x]$_补$=00.11011，[-x]$_补$=11.00101,[y]$_补$=11.00011

部分积	乘数 y_n	附加位 y_{n+1}	说明
00.000000 +11.00101	1100011	0	初值$[z_0]_补$=0，y_ny_{n+1}=10 部分积加$[-x]_补$
11.00101 11.10010	1110001	1	左侧结果右移1位 得$[z_1]_补$，y_ny_{n+1}=11，算术右移1位
11.11001 +00.11011	0111000	1	得$[z_2]_补$，y_ny_{n+1}=01 部分积加$[x]_补$
00.10100 00.01010	0011100	0	左侧结果右移1位 得$[z_3]_补$，y_ny_{n+1}=00，右移1位
00.00101	0001110	0	得$[z_4]_补$，y_ny_{n+1}=00，右移1位
00.00010 +11.00101	1000111	0	得$[z_5]_补$，y_ny_{n+1}=10 部分积加$[-x]_补$
11.00111	10001		最后一步不移位，得$[x*y]_补$

即$[x*y]_补$= 11.0011110001　　x*y= -0.1100001111

6.21 用补码加减交替法计算 x÷y .

（1）x=0.100111，y=-0.101011。（2）x=-0.101011，y=0.11011。

（3）x=0.10100，y=-0.10001。 （4）x=13/32，y=-27/32。

答：（1）x= 0.100111，y= 0.101011

$[x]_{补}$ = 0.100111，$[y]_{补}$ =0.101011， $[-y]_{补}$ =1.010101

被除数（余数）	商	说明
0.100111	0.00000	
+1.010101		$[x]_{补}$与$[y]_{补}$同号，+$[-y]_{补}$
1.111100	0	$[R]_{补}$与$[y]_{补}$ 异号，上商“0”
1.111000	0	逻辑左移 1 位
+0.101011		+$[y]_{补}$
0.100011	01	$[R]_{补}$与$[y]_{补}$ 同号，上商“1”
1.000110	01	左移 1 位
+1.010101		+$[-y]_{补}$
0.011011	011	$[R]_{补}$与$[y]_{补}$ 同号，上商“1”
0.110110	011	左移 1 位
+1.010101		+$[-y]_{补}$
0.001011	0111	$[R]_{补}$与$[y]_{补}$ 同号，上商“1”
0.010110	0111	左移 1 位
+1.010101		+$[-y]_{补}$
1.101011	01110	$[R]_{补}$与$[y]_{补}$ 异号，上商“0”
1.010110	01110	左移 1 位
+0.101011		+$[y]_{补}$
0.000001	011101	$[R]_{补}$与$[y]_{补}$ 同号，上商“1”
0.000010	0111011	左移 1 位，末位恒置“1”

即$[x/y]_{补}$ = 0.111011　　，　　$[x/y]$ = +0.111011

（2）x= -0.10101，y= 0.11011

$[x]_{补}$ = 1.01011，$[y]_{补}$ =0.11011，　$[-y]_{补}$ =1.00101

被除数（余数）	商	说明
1.01011	0.00000	
+0.11011		$[x]_{补}$与$[y]_{补}$异号，$+[y]_{补}$
0.00110	1	$[R]_{补}$与$[y]_{补}$ 同号，上商“1”
0.01100	1	逻辑左移 1 位
+1.00101		$+[-y]_{补}$
1.10001	10	$[R]_{补}$与$[y]_{补}$ 异号，上商“0”
1.00010	10	左移 1 位
+0.11011		$+[y]_{补}$
1.11101	100	$[R]_{补}$与$[y]_{补}$ 异号，上商“0”
1.11010	100	左移 1 位
+0.11011		$+[y]_{补}$
0.10101	1001	$[R]_{补}$与$[y]_{补}$ 同号，上商“1”
1.01010	1001	左移 1 位
+1.00101		$+[-y]_{补}$
0.01111	10011	$[R]_{补}$与$[y]_{补}$ 同号，上商“1”
0.11110	100111	左移 1 位，末位恒置“1”

即$[x/y]_{补}$ = 1.00111

$[x/y]$ = -0.11001

（3）x=0.10100，y=-0.10001。本题错误（x 的绝对值大于 y 的绝对值）本题改为 y/x

$[y]_{补}$=1.01111，$[x]_{补}$=0.10100，$[-x]_{补}$=1.01100

被除数（余数）	商	说明
1.01111	0.00000	
+0.10100		$[y]_{补}$与$[x]_{补}$异号，+$[x]_{补}$
0.00011	1	$[R]_{补}$与$[x]_{补}$ 同号，上商“1”
0.00110	1	左移 1 位
+1.01100		+$[-x]_{补}$
1.10010	10	$[R]_{补}$与$[x]_{补}$ 异号，上商“0”
1.00100	10	左移 1 位
+0.10100		+$[x]_{补}$
1.11000	100	$[R]_{补}$与$[x]_{补}$ 异号，上商“0”
1.10000	100	左移 1 位
+0.10100		+$[x]_{补}$
0.00100	1001	$[R]_{补}$与$[x]_{补}$ 同号，上商“1”
0.01000	1001	左移 1 位
+1.01100		+$[-x]_{补}$
1.10100	10010	$[R]_{补}$与$[x]_{补}$ 异号，上商“0”
1.01000	100101	左移 1 位，末位恒置“1”

即$[y/x]_{补}$= 1.00101

y/x=-0.11011

（4）x=13/32=0.01101，y=-27/32=-0.11011

$[x]_补$= 0.01101，$[y]_补$=1.00101， $[-y]_补$=0.11011

被除数（余数）	商	说明
0.01101	0.00000	
+1.00101		$[x]_补$与$[y]_补$异号，+$[y]_补$
1.10010	1	$[R]_补$与$[y]_补$ 同号，上商“1”
1.00100	1	左移 1 位
+0.11011		+$[-y]_补$
1.11111	11	$[R]_补$与$[y]_补$ 同号，上商“1”
1.11110	11	左移 1 位
+0.11011		+$[-y]_补$
0.11001	110	$[R]_补$与$[y]_补$ 异号，上商“0”
1.10010	110	左移 1 位
+1.00101		+$[y]_补$
0.10111	1100	$[R]_补$与$[y]_补$ 异号，上商“0”
1.01110	1100	左移 1 位
+1.00101		+$[y]_补$
0.10011	11000	$[R]_补$与$[y]_补$ 异号，上商“0”
1.00110	110001	左移 1 位，末位恒置“1”

即$[x/y]_补$= 1.10001

[x/y] = -0.01111

6.26 按机器补码浮点运算步骤计算$[x\pm y]_{补}$.

（1）$x=2^{-011}\times$ 0.101 100，$y=2^{-010}\times$（-0.011 100）；

（2）$x=2^{-011}\times$（-0.100 010），$y=2^{-010}\times$（-0.011 111）；

（3）$x=2^{101}\times$（-0.100 101），$y=2^{100}\times$（-0.001 111)。

解：

先将 x、y 转换成机器数形式：

（1）$x=2^{-011}\times$ 0.101 100，$y=2^{-010}\times$（-0.011 100）

[x]补=1，101；0.101 100, [y]补=1，110；1.100 100

[Ex]补=1,101, [y]补=1,110, [Mx]补=0.101 100, [My]补=1.100 100

1）对阶：

[ΔE]补=[Ex]补+[-Ey]补 = 11,101+ 00,010=11,111 < 0,

应 Ex 向 Ey 对齐，则：[Ex]补+1=11，101+00，001=11，110 = [Ey]补

[x]补=1，110；0.010 110

2）尾数运算：

[Mx]补+[My]补= 0.010 110 + 11.100 100=11.111010

[Mx]补+[-My]补=0.010 110 + 00.011100= 00.110 010

3）结果规格化：

[x+y]补=11，110；11.111 010 = 11，011；11.010 000 （尾数左规 3 次，阶码减 3）

[x-y]补=11，110；00.110 010, 已是规格化数。

4）舍入：无

5）溢出：无

则：$x+y=2^{-101}\times$（-0.110 000）

$x-y=2^{-010}\times$0.110 010

（2）$x=2^{-011}\times$（-0.100010）,$y=2^{-010}\times$（-0.011111）

[x]补=1，101；1.011 110, [y]补=1，110；1.100 001

1） 对阶：过程同(1)的 1），则

[x]补=1，110；1.101 111

2）尾数运算：

[Mx]补+[My]补= 11.101111 + 11. 100001 = 11.010000

[Mx]补+[-My]补= 11.101111 + 00.011111 = 00.001110

3）结果规格化：

[x+y]补=11，110；11.010 000,已是规格化数

[x-y]补=11，110；00.001 110 =11，100；00.111000 （尾数左规 2 次，阶码减 2）

4）舍入：无

5）溢出：无

则： $x+y=2^{-010}\times$（-0.110 000）

$x-y = 2^{-100} \times 0.111\ 000$

(3)x=2^{101}*(-0.100101),y=2^{100}*(-0.001111)

答：x 补 = 00,101；11.011011

y 补 = 00,100；11.110001

1.对阶：x 的阶码是 5，y 的阶码是 4，由于小阶向大阶看齐，所以要将 y 的阶码加 1，变成 5，同时 y 的尾数要右移 1 位。所以得到：

x 补 = 00,101；11.011011

y 补’ = 00,101；11.111001 （0 舍 1 入法）

2.尾数求和、求差：

[Sx+Sy’]=11.011011+11.111001=11.010100

[Sx-Sy’]= 11.011011-11.111001=11.011011+00.000111=11.100010

3.规格化：

[x+y]补=00,101；11.010100 是规格化数

[x-y]补=00,101；11.100010 需要左规，即尾数向左移动 1 位，阶码减 1

左规后，[x-y]补=00,100；11.000100

4.舍入：无

5.溢出：无

则：$x+y=2^{101}\times$（-0.101 100）

$x-y =2^{100}\times$（-0.111 100）

32. 设机器字长为 16 位，分别按 4、4、4、4 和 5、5、3、3 分组后，

（1）画出按两种分组方案的单重分组并行进位链框图，并比较哪种方案运算速度快。

（2）画出按两种分组方案的双重分组并行进位链框图，并对这两种方案进行比较。

（3）用 74181 和 74182 画出单重和双重分组的并行进位链框图。

解：（1）4—4—4—4 分组的 16 位单重分组并行进位链框图见教材 286 页图 6.22。

5—5—3—3 分组的 16 位单重分组并行进位链框图如下：

（2）4—4—4—4 分组的 16 位双重分组并行进位链框图见教材 289 页图 6.26。

5—5—3—3 分组的 16 位双重分组并行进位链框图如下：

5—5—3—3 分组的进位时间=2.5ty×3=7.5ty；

4—4—4—4 分组的进位时间=2.5ty×3=7.5ty；

可见，两种分组方案最长加法时间相同。

结论：双重分组并行进位的最长进位时间只与组数和级数有关，与组内位数无关。

（3）单重分组 16 位并行加法器逻辑图如下（正逻辑）：

注意： 1）74181 芯片正、负逻辑的引脚表示方法；

2）为强调可比性，5-5-3-3 分组时不考虑扇入影响；

3）181 芯片只有最高、最低两个进位输入/输出端，组内进位无引脚；

4）181 为 4 位片，无法 5-5-3-3 分组，只能 4-4-4-4 分组；

5）单重分组跳跃进位只用到 181，使用 182 的一定是双重以上分组跳跃进位；

6）单重分组跳跃进位是并行进位和串行进位技术的结合；双重分组跳跃进位是二级并

行进位技术；特别注意在位数较少时，双重分组跳跃进位可以采用全先行进位技术实现；位数较多时，可采用双重分组跳跃进位和串行进位技术结合实现。

求证： $[-x]_{补}=-[x]_{补}$

证明：

（1）当 x 为定点小数时（mod 2）

①若 $[x]_{补}=0.x_1x_2\cdots x_{n-1}x_n$

则：$x=0.x_1x_2\cdots x_{n-1}x_n$， ∴ $-x=-0.x_1x_2\cdots x_{n-1}x_n$

故 $[-x]_{补}=1.\bar{x}_1\bar{x}_2\cdots\bar{x}_{n-1}\bar{x}_n+2^{-n}$

又∵ $[x]_{补}=0.x_1x_2\cdots x_{n-1}x_n$，

则：$-[x]_{补}=-0.x_1x_2\cdots x_{n-1}x_n=2-0.x_1x_2\cdots x_{n-1}x_n=1.\bar{x}_1\bar{x}_2\cdots\bar{x}_{n-1}\bar{x}_n+2^{-n}$（mod 2）

故 $[-x]_{补}=-[x]_{补}$ 成立。

②若 $[x]_{补}=1.x_1x_2\cdots x_{n-1}x_n$

则：$x=-(0.\bar{x}_1\bar{x}_2\cdots\bar{x}_{n-1}\bar{x}_n+2^{-n})$， ∴ $-x=0.\bar{x}_1\bar{x}_2\cdots\bar{x}_{n-1}\bar{x}_n+2^{-n}$

故 $[-x]_{补}=0.\bar{x}_1\bar{x}_2\cdots\bar{x}_{n-1}\bar{x}_n+2^{-n}$

又∵ $[x]_{补}=1.x_1x_2\cdots x_{n-1}x_n$

则：$-[x]_{补}=-1.x_1x_2\cdots x_{n-1}x_n=2-1.x_1x_2\cdots x_{n-1}x_n=0.\bar{x}_1\bar{x}_2\cdots\bar{x}_{n-1}\bar{x}_n+2^{-n}$（mod 2）

故 $[-x]_{补}=-[x]_{补}$ 成立。

得到如下结论：不论定点小数 x 为正数还是负数，$[-x]_{补}=-[x]_{补}$ 成立。

（2）当 x 为定点整数时（mod 2^{n+1}）

①若 $[x]_{补}=0\ x_1x_2\cdots x_{n-1}x_n$

则：$x=x_1x_2\cdots x_{n-1}x_n$， ∴ $-x=-x_1x_2\cdots x_{n-1}x_n$

故 $[-x]_{补}=1\bar{x}_1\bar{x}_2\cdots\bar{x}_{n-1}\bar{x}_n+1$

又∵ $[x]_{补}=0\ x_1x_2\cdots x_{n-1}x_n$，

则：$-[x]_{补}=-0\ x_1x_2\cdots x_{n-1}x_n=2^{n+1}-0\ x_1x_2\cdots x_{n-1}x_n=1\bar{x}_1\bar{x}_2\cdots\bar{x}_{n-1}\bar{x}_n+1$（mod 2^{n+1}）

故 $[-x]_{补}=-[x]_{补}$ 成立。

②若 $[x]_{补}=1\ x_1x_2\cdots x_{n-1}x_n$

则：$x=-(\bar{x}_1\bar{x}_2\cdots\bar{x}_{n-1}\bar{x}_n+1)$， ∴ $-x=\bar{x}_1\bar{x}_2\cdots\bar{x}_{n-1}\bar{x}_n+1$

故 $[-x]_{补}=0\bar{x}_1\bar{x}_2\cdots\bar{x}_{n-1}\bar{x}_n+1$

又∵ $[x]_{补}=1\ x_1x_2\cdots x_{n-1}x_n$

则：$-[x]_{补}=-1\ x_1x_2\cdots x_{n-1}x_n=2^{n+1}-1\ x_1x_2\cdots x_{n-1}x_n=\bar{x}_1\bar{x}_2\cdots\bar{x}_{n-1}\bar{x}_n+1$（mod 2^{n+1}）

故 $[-x]_{补}=-[x]_{补}$ 成立。

得到如下结论：不论定点整数 x 为正数还是负数，$[-x]_{补}=-[x]_{补}$ 成立。

求证：$[x]_补+[y]_补=[x+y]_补$

证明：假定采用定点整数，且$|x|<2^n-1$，$|y|<2^n-1$，$|x+y|<2^n-1$

①当 $x>0$，$y>0$，则 $x+y>0$。

依定义：$[x]_补+[y]_补=x+y=[x+y]_补$

②当 $x>0$，$y<0$，则 $x+y>0$ 或 $x+y<0$。

依定义：$[x]_补=x$，$[y]_补=2^{n+1}+y$，$[x]_补+[y]_补=x+2^{n+1}+y=2^{n+1}+(x+y)=[x+y]_补$

③当 $x<0$，$y>0$，则 $x+y>0$ 或 $x+y<0$。证明同②

④当 $x<0$，$y<0$，则 $x+y<0$。

依定义：$[x]_补=2^{n+1}+x$，$[y]_补=2^{n+1}+y$，$[x]_补+[y]_补=2^{n+1}+x+2^{n+1}+y=2^{n+1}+(2^{n+1}+x+y)=[x+y]_补$

求证：$[x]_补-[y]_补=[x-y]_补$

证明：由于$[x]_补+[y]_补=[x+y]_补$，且$[-x]_补=-[x]_补$

故$[x]_补-[y]_补=[x]_补+[-y]_补=[x+(-y)]_补=[x-y]_补$

补码一位乘法之校正法的推导

设被乘数$[x]_补=x_0.x_1x_2\cdots x_n$，乘数$[y]_补=y_0.y_1y_2\cdots y_n$

①被乘数 x 符号任意，乘数 y 符号为正

则$[x]_补=x_0.x_1x_2\cdots x_n=2+x=2^{n+1}+x \pmod 2$，$[y]_补=y_0.y_1y_2\cdots y_n=0.y_1y_2\cdots y_n=y \pmod 2$

$[x]_补*[y]_补=[x]_补*y=(2^{n+1}+x)*y=2^{n+1}*y+xy$

而 $2^{n+1}*y=2^{n+1}*0.y_1y_2\cdots y_n=2*y_1y_2\cdots y_n$，$y_1y_2\cdots y_n$是一个正整数，则 $2*y_1y_2\cdots y_n=2 \pmod 2$

$\therefore [x]_补*[y]_补=2^{n+1}*y+xy=2+xy=[x*y]_补$

具体在计算机中的运算规则如下：$[z_i]_补$为部分积

$[z_0]_补=0$

$[z_1]_补=2^{-1}(y_n[x]_补+[z_0]_补)$

$[z_2]_补=2^{-1}(y_{n-1}[x]_补+[z_1]_补)$

…

$[z_i]_补=2^{-1}(y_{n-i+1}[x]_补+[z_{i-1}]_补)$

…

$[z_n]_补=[x*y]_补=2^{-1}(y_1[x]_补+[z_{n-1}]_补)$

②被乘数 x 符号任意，乘数 y 符号为负

则$[x]_补=x_0.x_1x_2\cdots x_n=2+x=2^{n+1}+x \pmod 2$，$[y]_补=y_0.y_1y_2\cdots y_n=1.y_1y_2\cdots y_n=2+y \pmod 2$

$y=1.y_1y_2\cdots y_n-2=0.y_1y_2\cdots y_n-1$

$xy=x(0.y_1y_2\cdots y_n-1)=x(0.y_1y_2\cdots y_n)-x$

$\therefore [x*y]_补=[x(0.y_1y_2\cdots y_n)-x]_补=[x(0.y_1y_2\cdots y_n)]_补-[x]_补=[x(0.y_1y_2\cdots y_n)]_补+[-x]_补$

$=[x]_补(0.y_1y_2\cdots y_n)+[-x]_补$

由此可知：乘数 y 符号为负时，相乘后需要加$[-x]_补$进行校正。

具体在计算机中的运算规则如下：

$[z_0]_补=0$

$[z_1]_补=2^{-1}(y_n[x]_补+[z_0]_补)$

$[z_2]_补=2^{-1}(y_{n-1}[x]_补+[z_1]_补)$

…

$[z_i]_补=2^{-1}(y_{n-i+1}[x]_补+[z_{i-1}]_补)$

…

$[z_n]_补=2^{-1}(y_1[x]_补+[z_{n-1}]_补)$

$[x*y]_补=[z_n]_补+[-x]_补$

补码一位乘法之比较法的推导

设被乘数 x 和乘数 y 符号任意，$[x]_补=x_0.x_1x_2\cdots x_n$， $[y]_补=y_0.y_1y_2\cdots y_n$

根据校正法的推导结果可得：

$$
\begin{aligned}
[x*y]_补&=(0.y_1y_2\cdots y_n)*[x]_补-y_0*[x]_补\\
&=(-y_0+2^{-1}y_1+2^{-2}y_2+\cdots+2^{-n}y_n)*[x]_补\\
&=[-y_0+(1-2^{-1})y_1+(2^{-1}-2^{-2})y_2+\cdots+(2^{-(n-1)}-2^{-n})y_n]*[x]_补\\
&=(-y_0+y_1-2^{-1}y_1+2^{-1}y_2-2^{-2}y_2+\cdots+2^{-(n-1)}y_n-2^{-n}y_n)*[x]_补\\
&=[(y_1-y_0)+(y_2-y_1)2^{-1}+\cdots+(y_n-y_{n-1})2^{-(n-1)}+(0-y_n)2^{-n})]*[x]_补\\
&=[(y_1-y_0)+(y_2-y_1)2^{-1}+\cdots+(y_n-y_{n-1})2^{-(n-1)}+(y_{n+1}-y_n)2^{-n})]*[x]_补\ (y_{n+1}=0)
\end{aligned}
$$

具体在计算机中的运算规则如下：

$[z_0]_补=0$

$[z_1]_补=2^{-1}\{(y_{n+1}-y_n)[x]_补+[z_0]_补\}$

$[z_2]_补=2^{-1}\{(y_n-y_{n-1})[x]_补+[z_1]_补\}$

…

$[z_i]_补=2^{-1}\{(y_{n-i+2}-y_{n-i+1})[x]_补+[z_{i-1}]_补\}$

…

$[z_n]_补=2^{-1}\{(y_2-y_1)[x]_补+[z_{n-1}]_补\}$

$[x*y]_补=[z_{n+1}]_补=[z_n]_补+(y_1-y_0)[x]_补$

P231 例 6.3 设浮点数字长 16 位，其中阶码 5 位（含 1 位符号位），尾数 11 位（含 1 位符号位），将十进制数 13/128 写成二进制定点数和浮点数，并分别写出它在定点机和浮点机中的机器数形式。

解：令 x=13/128

则其尾数的二进制形式：x=0.0001101000

其定点数表示： x=0.0001101000

其浮点数规格化形式： $x=0.1101000000*2^{-11}$

在定点机中：$[x]_原=[x]_补=0.0001101000$

在浮点机中：$[x]_原=1,0011;0.0001101000$

$[x]_补=1,1101;0.0001101000$

P232 例 6.4 设浮点数字长 16 位，其中阶码 5 位（含 1 位符号位），尾数 11 位（含 1 位符号位），将十进制数-54 写成二进制定点数和浮点数，并分别写出它在定点机和浮点机中的机器数形式。

解：令 x=-54

　　则其二进制形式：　　　x=-110110

　　其定点数表示：　　　　x=-0000110110

　　其浮点数规格化形式：　 $x=-0.1101100000*2^{110}$

　　在定点机中：$[x]_{原}$=1,0000110110

　　　　　　　　$[x]_{补}$=1,1111001010

　　在浮点机中：$[x]_{原}$=0,0110;1.1101100000

　　　　　　　　$[x]_{补}$=0,0110;1.0010100000

P232 例 6.5 写出对应图 6.2 中四个临界点的原码和补码形式。设图中 n=10,m=4,阶符和数符各取 1 位。

解：原码形式

最大正数真值：$2^{15}*(1-2^{-10})$，　其原码浮点数：0,1111;0.1111111111

最小正数真值：$2^{-15}*2^{-10}$,　　　　其原码浮点数：1,1111;0.0000000001

最大负数真值：$2^{-15}*(-2^{-10})$,　　其原码浮点数：1,1111;1.0000000001

最小负数真值：$2^{15}*(-(1-2^{-10}))$，其原码浮点数：0,1111;1.1111111111

补码形式

最大正数真值：$2^{15}*(1-2^{-10})$，　其补码浮点数：0,1111;0.1111111111

最小正数真值：$2^{-15}*2^{-10}$,　　　其补码浮点数：1,0001;0.0000000001

最大负数真值：$2^{-15}*(-2^{-10})$,　 其补码浮点数：1,0001;1.1111111111

最小负数真值：$2^{15}*(-1)$,　　　 其补码浮点数：0,1111;1.0000000000

P232 例 6.6 设浮点数字长 16 位，其中阶码 5 位（含 1 位符号位），尾数 11 位（含 1 位符号位），写出-53/512 对应浮点规格化数的原码、补码和阶码用移码，尾数用补码的形式。

解：令 x=-53/512

　　则其二进制形式：x=-0.000110101，规格化形式：$x=-0.1101010000*2^{-11}$

　　$[x]_{原}$=1,0011;1.1101010000

　　$[x]_{补}$=1,1101;1.0010110000

　　$[x]_{阶移尾补}$=0,1101;1.0010110000

在 IEEE754 标准中，为什么其短实数、长实数和临时实数的阶码（移码表示）是在以补码表示的基础上加偏移量 3FH、3FFH 和 3FFFH?

解答：

IEEE754 标准中，阶码采用移码表示，尾数采用补码表示，且为规范化的数。

以短实数为例进行说明：（8 位阶码，24 位尾数，阶符和数符各 1 位）

令 x=-53/512

则其二进制形式：x=-0.000110101

规格化形式：　　$x=-0.110\ 1010\ 0000\ 0000\ 0000\ 0000*2^{-11}$

①阶码（补码表示）：1,1111101
尾数（补码表示）：1.001 0110 0000 0000 0000 0000

采用 IEEE754 标准，隐含最高有效位，尾数规范化形式：$x=-0.101\ 0100\ 0000\ 0000\ 0000\ 0000*2^{-100}$
尾数（补码）为：1.0101 1000 0000 0000 0000 0000

②则其阶码（补码）为：1,111 1100，③其阶码（移码）为：0,111 1100

以表达式①也可求其用移码表示的阶码（表达式③），即：
1,111 1101+0,111 1111（3FH）=0,111 1100，成立！

浮点数加减法为什么要对阶？为什么小阶向大阶看齐？
解答：
日常生活中，一个实数的加减需要小数点对齐。
而在计算机中，做浮点数加减法之前，两个浮点数是规格化或规范化的。因此，小数点对齐必须要对阶。
有一个问题我们必须要清楚：计算机中浮点数加减一定是对阶后的尾数加减。因此，尾数必须是一个纯小数，由此我们可以得出这样的结论：小阶向大阶看齐。

如果大阶向小阶看齐，大阶-1，尾数左移 1 位，出现溢出错误，如：
$[x]_补$=0,0110;1.0010100000 和$[y]_补$=0,0111;1.0010100000
$x=-0.1101100000*2^6=-110110=-54$，$y=-0.1101100000*2^7=-1101100=-108$
$[y]_补$阶大，需要向 $[x]_补$看齐，$[y]_补$=0,0110;0.0101000000，$[y]_补$的尾数变成了正值，错误！

如果小阶向大阶看齐，小阶+1，尾数右移 1 位，最多会产生舍去误差，影响精度，如：
$[x]_补$=0,0110;1.0010100000 和$[y]_补$=0,0111;1.0010100000
$[x]_补$阶小，需要向 $[y]_补$看齐，$[x]_补$=0,0111;1.1001010000
运算过程：$[x]_补+[y]_补$=0,0110;1.0010100000+0,0111;1.0010100000
=0,0111;1.1001010000+0,0111;1.0010100000
=0,0111;10.1011110000
=0,1000;1.0101111000
$x+y=-0.1010001000*2^8=-10100010=-162$

就 P273 图 6.14 中的补码表示的规格化数（阶码 2+m 位，尾数 2+n 位，其中阶码和尾数各有 2 位符号位），写出四个临界点 A、B、a 和 b。
解答：
规格化数满足“0.5≤尾数的绝对值＜1”
为了方便计算机判断，实际使用中，规格化数满足“尾数的符号位与最高有效位数值相反”
也即：尾数为正数时：“0.5≤尾数的绝对值＜1”，尾数为负数时：“0.5＜尾数的绝对值≤1”
最大正数 B，其规格化补码浮点数：00,11…11;00.111…11，真值：$2^{2m-1}*(1-2^{-n})$
最小正数 b，其规格化补码浮点数：11,00…00;00.100…00，真值：$2^{-2m}*2^{-1}$
最大负数 a，其规格化补码浮点数：11,00…00;11.011…11，真值：$2^{-2m}*(-2^{-1}-2^{-n})$
最小负数 A，其规格化补码浮点数：00,11…11;11.000…00，真值：$2^{2m-1}*(-1)$

这里需要说明的是：最大负数 a，为什么其尾数 11.011…11 是绝对值最小的负数（也即是最大的负数）：

首先，尾数 11.011…11 是规格化数，正常讲，规格化数应该满足“尾数的绝对值≤0.5＜1”，

$0.5=2^{-1}=0.100\cdots00$，补码表示为：00.100…00，满足规格化条件

而$-0.5=-2^{-1}=-0.100\cdots00$，补码表示为：11.100…00，不满足规格化条件，故只能是与-0.5 相邻的两个数才有可能满足规格化条件，$-0.5+2^{-n}$和$-0.5-2^{-n}$

$-0.5+2^{-n}=-0.100\cdots00+0.0\cdots001=-0.011\cdots1$，变换成补码表示为：11.100…001，不满足规格化条件

$-0.5-2^{-n}=-0.100\cdots00-0.0\cdots001=-0.100\cdots001$，变换成补码表示为：11.000…001，满足规格化条件

浮点乘除法运算

1. 浮点乘法运算：尾数相乘(补码乘法)，阶码相加

①若阶码采用补码表示：$[j_x]_补+[j_y]_补=[j_x+j_y]_补$

②若阶码采用移码表示：根据移码定义：

$[j_x]_移+[j_y]_移=(2^n+j_x)+(2^n+j_y)=(2^n+j_x+j_y)+2^n=2^n+[j_x+j_y]_移$

而根据补码定义：$[j_y]_补=2^{n+1}+j_y$

$[j_x]_移+[j_y]_补=(2^n+j_x)+(2^{n+1}+j_y)=2^{n+1}+(2^n+j_x+j_y)=2^{n+1}+[j_x+j_y]_移=[j_x+j_y]_移$

2. 浮点除法运算：尾数相除(补码除法)，阶码相减

补码除法时一定要注意，被除数的绝对值要小于或等于除数的绝对值

①若阶码采用补码表示：$[j_x]_补-[j_y]_补=[j_x]_补+[-j_y]_补=[j_x+(-j_y)]_补=[j_x-j_y]_补$

②若阶码采用移码表示：

$[j_x]_移+[-j_y]_补=(2^n+j_x)+(2^{n+1}-j_y)=2^{n+1}+(2^n+(j_x-j_y))=2^{n+1}+[j_x-j_y]_移=[j_x-j_y]_移$

移码和补码的关系？

移码定义：$[x]_移=2^n+x\ (-2^n\leqslant x<2^n)$

补码定义：

$$[x]_补=\begin{cases} x & (0\leqslant x<2^n) \\ 2^{n+1}+x=(2^{n+1}-1)+x+1=[x]_反+1 & (-2^n\leqslant x<0) \end{cases}$$

故当 $0\leqslant x<2^n$ 时，$[x]_移=2^n+x=2^n+[x]_补$

当$-2^n\leqslant x<0$ 时，$[x]_移=2^n+x=(2^{n+1}+x)-2^n=[x]_补-2^n$

移码的符号位与补码的符号位正好相反，即移码的符号位为“1”时表示正号，为“0”时表示负号。而移码数值部分的变化与补码的变化规律相同。

阶码采用移码表示时，为什么浮点数乘法其结果的阶码不能用被乘数的阶码加乘数的阶码？如何解决？

解：

若阶码采用移码，则根据定义（$j_x=\pm x_1x_2\cdots x_n$）有：

$[j_x]_移=2^n+j_x \qquad -2^n\leqslant j_x<2^n$

$[j_y]_移=2^n+j_y \qquad -2^n\leqslant j_y<2^n$

$\therefore [j_x]_移+[j_y]_移=2^n+j_x+2^n+j_y=2^n+(2^n+j_x+j_y)=2^n+[j_x+j_y]_移$

即两个阶码相加时，多出了 2^n，故阶码不能用被乘数的阶码加乘数的阶码。

从补码的定义（$j_y = \pm y_1 y_2 \cdots y_{n-1} y_n$）有：$[j_y]_{补} = 2^{n+1} + j_y$（mod 2^{n+1}）

则：$[j_x]_{移} + [j_y]_{补} = 2^n + j_x + 2^{n+1} + j_y$

$$= 2^{n+1} + [2^n + (j_x + j_y)]$$
$$= 2^{n+1} + [j_x + j_y]_{移}$$
$$= [j_x + j_y]_{移} \pmod{2^{n+1}}$$

同理，$[j_x]_{移} + [-j_y]_{补} = [j_x - j_y]_{移}$（mod 2^{n+1}）

而移码和补码表示只有符号位相反，故在移码加减运算时，只需要将加数或减数符号位取反，即变成补码即可实现移码的加减运算。