王道考研——组成原理

WWW. CSKAOYAN. COM

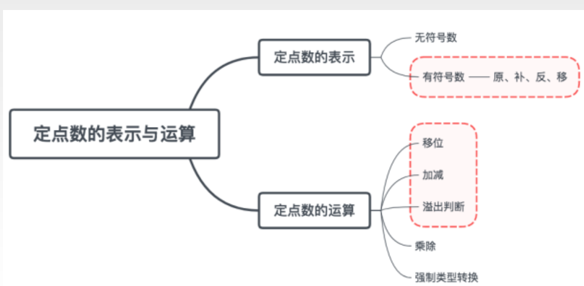
第二章 数据的表示和运算

本节内容

定点数的表示和运算

无符号数
原码

本节总览



无符号数

无符号数：整个机器字长的全部二进制位均为数值位，没有符号位，相当于数的绝对值。

1001 1100B
$=1\times2^7+0\times2^6+0\times2^5+1\times2^4+1\times2^3+1\times2^2+0\times2^1+0\times2^0$
=156D

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
2	4	8	16	32	64	128	256	512	1024	2048	4096	8192	16384	32768	65536

表示范围

8位二进制数：2^8种不同的状态

0000 0000 ～ $\boxed{1111\ 1111}$ = 1 0000 0000 − 1

0 ～ $\boxed{255}$ = 2^8 − 1

n位的无符号数表示范围为：0 ～ 2^n-1

有符号数

+ 156 D = 0 1001 1100B
- 156 D = 1 1001 1100B

真值　　机器数

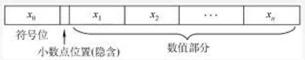
定点表示

+ 156 D = 0 1001 1100B
- 156 D = 1 1001 1100B

真值　　机器数

小数点：隐含存储(定点数：事先约定；浮点数：按规则浮动)

定点小数



+0.75D = 0.11B 存储为011 (未考虑位数扩展)
-0.75D = 1.11B 存储为111

表示范围

$K_{-1}\times 2^{-1} + K_{-2}\times 2^{-2} + \ ... + K_{-m}\times 2^{-m}$

绝对值：0.00 ～ $\boxed{0.11}$ = 1.00 － 0.01

0 ～ $1\text{-}\ 2^{-2}$

有n位尾数的定点小数：$-(1\text{-}\ 2^{-n})$ ～ $1\text{-}\ 2^{-n}$

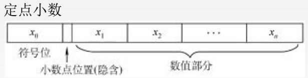
定点表示

+ 156 D = 0 1001 1100B
- 156 D = 1 1001 1100B

真值　　机器数

小数点：隐含存储(定点数：事先约定；浮点数：按规则浮动)

定点小数



+0.75D = 0.11B 存储为011 (未考虑位数扩展)
-0.75D = 1.11B 存储为111

表示范围 $-(1\text{-}\ 2^{-n})$ ～ $1\text{-}\ 2^{-n}$

定点整数

+3D = 011.B 存储为011 (未考虑位数扩展)
-3D = 111.B 存储为111

表示范围

绝对值：0 ～ $2^n - 1$

有n位尾数的定点整数：$-(2^n - 1)$ ～ $2^n - 1$

原码

+ 156 D = 0 1001 1100B
- 156 D = 1 1001 1100B

真值　　机器数

约定：用X表示真值，用$[X]_{原}$表示原码，$[X]_{补}$表示补码，$[X]_{反}$表示反码，$[X]_{移}$表示移码。
假设字长为8位(符号位+数值位)，最高位为符号位

纯小数原码

$x_1 = +0.8125$， $x_2 = -0.8125$　真值(十进制形式)
$x_1 = +0.1101$， $x_2 = -0.1101$　真值(二进制形式)

$[x_1]_{原}=\mathbf{0}.1101$，$[x_2]_{原}=\mathbf{1}.1101$　我们的做法：+换成0，-换成1
$[x_1]_{原}=\mathbf{0}.1101$，$[x_2]_{原}=0.1101 + 1.0000 =\mathbf{1}.1101$　计算机的做法：“加”

$$[x]_{原}=\begin{cases} x & 1>x\geqslant 0 \\ 1-x=1+|x| & 0\geqslant x>-1 \end{cases}$$

$[x_1]_{原}=\mathbf{0}.1101000$，$[x_2]_{原}=\mathbf{1}.1101000$

计算机中：**0**1101000，**1**1101000

若字长为n+1，则原码小数的表示范围为$-(1-2^{-n}) \le x \le 1-2^{-n}$（关于原点对称）

原码

符号、数值分开处理

$[+0]_{原}$=00000…

$[-0]_{原}$=10000…

+ 156 D = 0 1001 1100B

- 156 D = 1 1001 1100B

真值　　机器数

运算： 00001110 + 10001110

根据最高位调整成相应的无符号数运算

→ 00001110 - 00001110

约定：用X表示真值，用$[X]_{原}$表示原码，$[X]_{补}$表示补码，$[X]_{反}$表示反码，$[X]_{移}$表示移码。

假设字长为8位(符号位+数值位)，最高位为符号位

纯整数原码

$x_1 = +14$， $x_2 = -14$　　真值(十进制形式)

$x_1 = +1110$， $x_2 = -1110$　　真值(二进制形式)

$[x_1]_{原}$=**0**,1110， $[x_2]_{原}$=**1**,1110　　我们的做法：+换成0，-换成1

$[x_1]_{原}$=**0**,1110， $[x_2]_{原}$=0,1110 + 1,0000 =**1**,1110　　计算机的做法：“加”

$$[x]_{原}=\begin{cases}0,x & 2^n > x \geqslant 0\\ 2^n - x = 2^n + |x| & 0 \geqslant x > -2^n\end{cases}$$

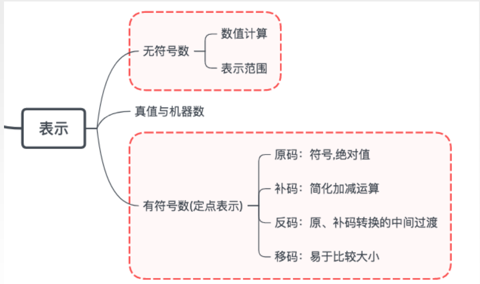
$[x_1]_{原}$=**0**,0001110， $[x_2]_{原}$=**1**, 0001110

计算机中： **0**0001110， **1**0001110

若字长为n+1，则原码整数的表示范围为$-(2^n-1) \leq x \leq 2^n-1$（关于原点对称）

王道考研/CSKAOYAN.COM

知识点回顾



王道考研/CSKAOYAN.COM

本节内容

定点数的表示和运算

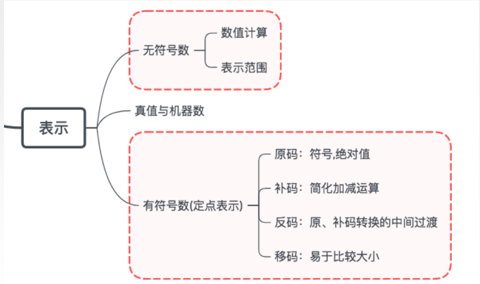
补码

反码

移码

王道考研/CSKAOYAN.COM

本节总览



王道考研/CSKAOYAN.COM

加减运算

有符号数		无符号数
14	00001110	14
-14	+ 10001110	142
0	10011100	156
☹		☺

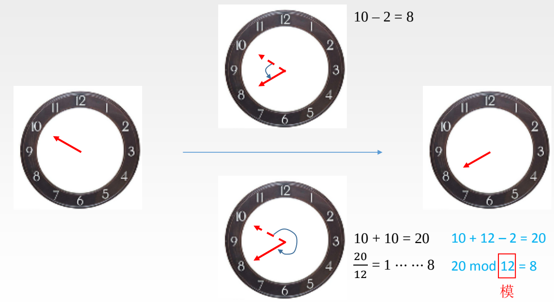
00001110
+ 10001110

00001110
- 00001110

00000000

☺

补码



加减运算

有符号数		无符号数
14	00001110	14
-14	+ 10001110	142
0	10011100	156
☹		☺

00001110
+ 10001110

00001110
- 00001110 → + 1,00000000 - 00001110 → 00001110 + 11110010

00000000 | 100000000

加减运算

有符号数		无符号数
14	00001110	14
-14	+ 10001110	142
0	10011100	156
☹		☺

00001110
+ 10001110

00001110
- 00001110

00000000

补码 正数：与原码相同
负数：符号位与原码相同，数值位由原码取反加1得到

14	00001110	14
-14	+ 11110010	242
0	100000000	256 mod 1,00000000 = 0
☺		☺ 2^8

00001110
+ 1,00000000 - 00001110

0,00000001+ 0,11111111 - 00001110
加1 取反

$[X]_{原}$ → $[X]_{补}$：正数不变；负数符号位不变，数值位取反加1

$[X]_{补}$ → $[-X]_{补}$：连同符号位一起取反加1

补码

对于正数，补码与原码的表示相同，$[x]_补=[x]_原$。
对于负数，原码符号位不变，数值部分按位取反，末位加1（即所谓“取反加1”）
此规则同样适用于由$[x]_补$求$[x]_原$。

$$\begin{array}{r} \mathbf{0}0001110 \\ -\ \ \mathbf{0}0001110 \\ \hline \end{array} \longrightarrow \begin{array}{r} \mathbf{0}0001110 \\ +\ \ 1,00000000 - \mathbf{0}0001110 \\ \hline \end{array}$$

纯整数补码

$$[x]_补 = \begin{cases} 0, x & 2^n > x \geqslant 0 \\ 2^{n+1} + x = 2^{n+1} - |x| & 0 \geqslant x \geqslant -2^n \end{cases} \quad (\bmod\ 2^{n+1})$$

$x_1=+1010$，$x_2=-1010$，字长为8位，则其补码表示为：
$[x_1]_补 = \mathbf{0},0001010$
$[x_2]_补 = 2^8 - 0,0001010 = 10,0000000 - 0,0001010 = \mathbf{1},1110110 = [-x_1]_补$
$[x_2]_原 = \mathbf{1},0001010$

若字长为n+1，则补码的表示范围为$-2^n \le x \le 2^n-1$（比原码多表示-2^n）
$[-2^n]_补 = 10,0000 - 1,0000 = 1,0000$

王道考研/CSKAOYAN.COM

补码

对于正数，补码与原码的表示相同，$[x]_补=[x]_原$。
对于负数，原码符号位不变，数值部分按位取反，末位加1（即所谓“取反加1”）
此规则同样适用于由$[x]_补$求$[x]_原$。

$$\begin{array}{r} \mathbf{0}0001110 \\ -\ \ \mathbf{0}0001110 \\ \hline \end{array} \longrightarrow \begin{array}{r} \mathbf{0}0001110 \\ +\ \ 1,00000000 - \mathbf{0}0001110 \\ \hline \end{array}$$

纯小数补码

$$[x]_补 = \begin{cases} x & 1 > x \geqslant 0 \\ 2 + x = 2 - |x| & 0 > x \geqslant -1 \end{cases} \quad (\bmod\ 2)$$

$x_1=+0.1001$，$x_2=-0.1001$，字长为8位，则其补码表示为：
$[x_1]_补=\mathbf{0}.1001000$
$[x_2]_补 = 2 - 0.1001000 = 10.0000000 - 0.1001000 = \mathbf{1}.0111000 = [-x_1]_补$
$[x_1]_原 = \mathbf{1}.1001000$

若字长为n+1，则补码的表示范围为$-1 \le x \le 1-2^{-n}$（比原码多表示-1）
$[-1]_补=10.0000-1.0000=1.0000$

王道考研/CSKAOYAN.COM

反码

对于正数，反码与原码的表示相同，$[x]_反=[x]_原$。
对于负数，原码符号位不变，数值部分按位取反，
此规则同样适用于由$[x]_反$求$[x]_原$。

表示范围：与原码一样。

纯整数反码

$$[x]_反 = \begin{cases} 0, x & 2^n > x \geqslant 0 \\ (2^{n+1} - 1) + x & 0 \geqslant x > -2^n \end{cases} \quad (\bmod\ 2^{n+1} - 1)$$

$x_1=+1011$，$x_2=-1011$，字长为8位，则其反码表示为：
$[x_1]_反= \mathbf{0},0001011$
$[x_2]_反= 1,1111111 - 0,0001011 = \mathbf{1},1110100$
$[x_1]_原 = \mathbf{1},0001011$

若字长为n+1，则反码的表示范围为$-(2^n-1) \le x \le 2^n-1$（关于原点对称）

王道考研/CSKAOYAN.COM

反码

对于正数，反码与原码的表示相同，$[x]_反=[x]_原$。
对于负数，原码符号位不变，数值部分按位取反，
此规则同样适用于由$[x]_反$求$[x]_原$。

表示范围：与原码一样。

纯小数反码

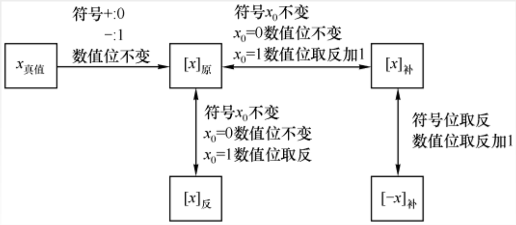
$$[x]_反 = \begin{cases} x & 1 > x \geqslant 0 \\ (2 - 2^{-n}) + x & 0 \geqslant x > -1 \end{cases} \quad (\bmod\ 2 - 2^{-n})$$

$x_1=+0.0110$，$x_2=-0.0110$，字长为8位，则其反码表示为：
$[x_1]_反=\mathbf{0}.0110000$
$[x_2]_反=1.1111111 - 0.0110000 = \mathbf{1}.1001111$
$[x_1]_原 = \mathbf{1}.0110000$

若字长为n+1，则反码的表示范围为$-(1-2^{-n}) \le x \le 1-2^{-n}$（关于原点对称）

王道考研/CSKAOYAN.COM

原补反相互转换



王道考研/CSKAOYAN.COM

原补反相互转换

行数	机器数	真值(十进制)			
		无符号数	原码	反码	补码
1	0000 0000	0	+0	+0	+0,-0
2	0000 0001	1	+1	+1	+1
3	0000 0010	2	+2	+2	+2
…	…	…	…	…	…
126	0111 1101	125	+125	+125	+125
127	0111 1110	126	+126	+126	+126
128	0111 1111	127	+127	+127	+127
129	1000 0000	128	-0	-127	-128
130	1000 0001	129	-1	-126	-127
131	1000 0010	130	-2	-125	-126
…	…	…	…	…	…
253	1111 1100	252	-124	-3	-4
254	1111 1101	253	-125	-2	-3
255	1111 1110	254	-126	-1	-2
256	1111 1111	255	-127	-0	-1

王道考研/CSKAOYAN.COM

移码

行数	机器数	真值(十进制)				
		无符号数	原码	反码	补码	移码
1	0000 0000	0	+0	+0	+0,-0	-128
2	0000 0001	1	+1	+1	+1	-127
3	0000 0010	2	+2	+2	+2	-126
…	…	…	…	…	…	…
126	0111 1101	125	+125	+125	+125	-3
127	0111 1110	126	+126	+126	+126	-2
128	0111 1111	127	+127	+127	+127	-1
129	1000 0000	128	-0	-127	-128	0
130	1000 0001	129	-1	-126	-127	1
131	1000 0010	130	-2	-125	-126	2
…	…	…	…	…	…	…
253	1111 1100	252	-124	-3	-4	124
254	1111 1101	253	-125	-2	-3	125
255	1111 1110	254	-126	-1	-2	126
256	1111 1111	255	-127	-0	-1	127

移码就是在真值X上加上一个常数（偏置值），通常这个常数取2^n 。

$[x]_{移}=2^n+x$

x_1=+10101，x_2=−10101，字长为8位，则其移码表示为：

$[x_1]_{移}=2^7+10101$
=10000000+10101
=1,0010101

$[x_2]_{移}=2^7+(−10101)$
=10000000 +(−10101)
=0,1101011

王道考研/CSKAOYAN.COM

移码

行数	机器数	真值(十进制)				
		无符号数	原码	反码	补码	移码
1	0000 0000	0	+0	+0	+0,-0	-128
2	0000 0001	1	+1	+1	+1	-127
3	0000 0010	2	+2	+2	+2	-126
…	…	…	…	…	…	…
126	0111 1101	125	+125	+125	+125	-3
127	0111 1110	126	+126	+126	+126	-2
128	0111 1111	127	+127	+127	+127	-1
129	1000 0000	128	-0	-127	-128	0
130	1000 0001	129	-1	-126	-127	1
131	1000 0010	130	-2	-125	-126	2
…	…	…	…	…	…	…
253	1111 1100	252	-124	-3	-4	124
254	1111 1101	253	-125	-2	-3	125
255	1111 1110	254	-126	-1	-2	126
256	1111 1111	255	-127	-0	-1	127

移码0111 1110的真值：

1. 转换成无符号数真值：126
2. 减去偏置值1000 0000对应的无符号数真值128得到移码真值：
126 – 128 = -2

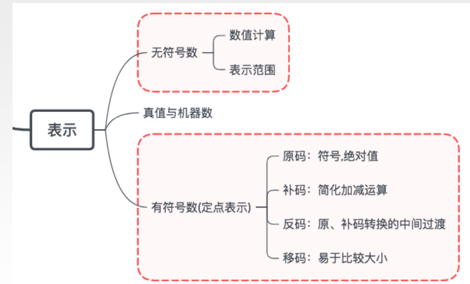
或者：
0111 1110 – 1000 0000
= 1111 1110
对应补码真值-2

王道考研/CSKAOYAN.COM

移码

真值(十进制)	补码	移码
-128	1000 0000	0000 0000
-127	1000 0001	0000 0001
-126	1000 0010	0000 0010
···	···	···
-3	1111 1101	0111 1101
-2	1111 1110	0111 1110
-1	1111 1111	0111 1111
0	0000 0000	1000 0000
1	0000 0001	1000 0001
2	0000 0010	1000 0010
3	0000 0011	1000 0011
···	···	···
124	0111 1100	1111 1100
125	0111 1101	1111 1101
126	0111 1110	1111 1110
127	0111 1111	1111 1111

知识点回顾

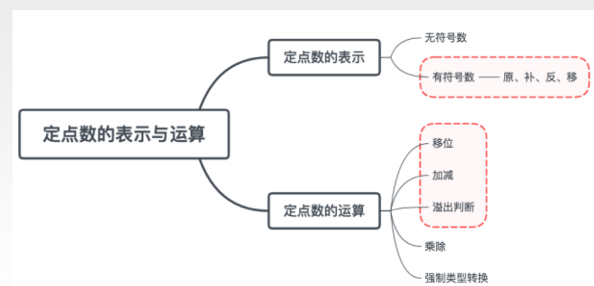


本节内容

定点数的表示和运算

移位运算

本节总览



移位运算

r 进制： $K_n K_{n-1} \dots K_2 K_1 K_0 K_{-1} K_{-2} \dots K_{-m}$

$= K_n \times r^n + K_{n-1} \times r^{n-1} + \dots + K_2 \times r^2 + K_1 \times r^1 + K_0 \times r^0 + K_{-1} \times r^{-1} + K_{-2} \times r^{-2} + \dots + K_{-m} \times r^{-m}$

10 进制：100.0

小数点左移2位：1.000，相当于除以100，即除以10^2

小数点右移1位：1000.，相当于乘以10，即乘以10^1

也可看作固定小数点：

数字右移2位

数字左移1位

右移n位：$\div r^n$　　左移n位：$\times r^n$

1 0 1 1 0 1 0 1

2 进制： $K_7 K_6 K_5 K_4 K_3 K_2 K_1 K_0$

$= K_7 \times 2^7 + K_6 \times 2^6 + K_5 \times 2^5 + K_4 \times 2^4 + K_3 \times 2^3 + K_2 \times 2^2 + K_1 \times 2^1 + K_0 \times 2^0$

机器数采用无符号数：逻辑移位

逻辑左移时，高位移丢，低位添0；逻辑右移时，低位移丢，高位添0

1　0 1 1 0 1 0 1

王道考研/CSKAOYAN.COM

移位运算

r 进制： $K_n K_{n-1} \dots K_2 K_1 K_0 K_{-1} K_{-2} \dots K_{-m}$

$= K_n \times r^n + K_{n-1} \times r^{n-1} + \dots + K_2 \times r^2 + K_1 \times r^1 + K_0 \times r^0 + K_{-1} \times r^{-1} + K_{-2} \times r^{-2} + \dots + K_{-m} \times r^{-m}$

10 进制：100.0

小数点左移2位：1.000，相当于除以100，即除以10^2

小数点右移1位：1000.，相当于乘以10，即乘以10^1

也可看作固定小数点：

数字右移2位

数字左移1位

右移n位：$\div r^n$　　左移n位：$\times r^n$

1 0 1 1 0 1 0 1

2 进制： $K_7 K_6 K_5 K_4 K_3 K_2 K_1 K_0$

$= K_7 \times 2^7 + K_6 \times 2^6 + K_5 \times 2^5 + K_4 \times 2^4 + K_3 \times 2^3 + K_2 \times 2^2 + K_1 \times 2^1 + K_0 \times 2^0$

机器数采用无符号数：逻辑移位

逻辑左移时，高位移丢，低位添0；逻辑右移时，低位移丢，高位添0

1　0 1 1 0 1 0 1 0　　　　1 0 1 1 0 1 0　1

王道考研/CSKAOYAN.COM

移位运算

r 进制： $K_n K_{n-1} \dots K_2 K_1 K_0 K_{-1} K_{-2} \dots K_{-m}$

$= K_n \times r^n + K_{n-1} \times r^{n-1} + \dots + K_2 \times r^2 + K_1 \times r^1 + K_0 \times r^0 + K_{-1} \times r^{-1} + K_{-2} \times r^{-2} + \dots + K_{-m} \times r^{-m}$

10 进制：100.0

小数点左移2位：1.000，相当于除以100，即除以10^2

小数点右移1位：1000.，相当于乘以10，即乘以10^1

也可看作固定小数点：

数字右移2位

数字左移1位

右移n位：$\div r^n$　　左移n位：$\times r^n$

1 0 1 1 0 1 0 1

2 进制： $K_7 K_6 K_5 K_4 K_3 K_2 K_1 K_0$

$= K_7 \times 2^7 + K_6 \times 2^6 + K_5 \times 2^5 + K_4 \times 2^4 + K_3 \times 2^3 + K_2 \times 2^2 + K_1 \times 2^1 + K_0 \times 2^0$

机器数采用无符号数：逻辑移位

逻辑左移时，高位移丢，低位添0；逻辑右移时，低位移丢，高位添0

0 1 1 0 1 0 1 0　　　　0 1 0 1 1 0 1 0　1

王道考研/CSKAOYAN.COM

算术移位

1 0 1 1 0 1 0 1

2 进制： $S\ K_6 K_5 K_4 K_3 K_2 K_1 K_0$

$= (-1)^S \times (K_6 \times 2^6 + K_5 \times 2^5 + K_4 \times 2^4 + K_3 \times 2^3 + K_2 \times 2^2 + K_1 \times 2^1 + K_0 \times 2^0)$

算术移位：机器码采用有符号数

符号位不参与移位

原码：

符号位	绝对值

左移、右移都补0

1,0110101　　真值-53

左移1位(丢0)：1,1101010　　真值-106

右移1位(丢1)：1,0011010　　真值-26　　假设不丢1：1,0011010.1　　真值-26.5

再左移1位(丢1)：1,1010100　　真值-84　　假设不丢1：1, 11010100　　真值-212

再右移1位(丢0)：1,0001101　　真值-13

原码算术移位：左移丢1，运算出错；右移丢1，影响精度。

王道考研/CSKAOYAN.COM

算术移位

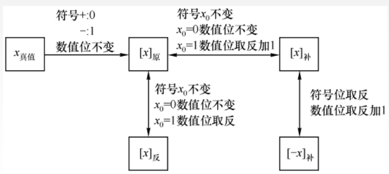
1 0 1 1 0 1 0 1

2 进制： $S\ K_6K_5K_4K_3K_2K_1K_0$

算术移位：机器码采用有符号数

符号位不参与移位

正数：原码、补码、反码一样 → 左移、右移都补0



负数：反码1 <—> 原码0

原码 1,0110101

反码 1,1001010

补码 1,1001011

算术移位

1 0 1 1 0 1 0 1

2 进制： $S\ K_6K_5K_4K_3K_2K_1K_0$

算术移位：机器码采用有符号数

符号位不参与移位

	码　制	添补代码
正数	原码、补码、反码	0
负数	原码	0
	补码	左移添 0
		右移添 1
	反码	1

正数：原码、补码、反码一样

→ 左移、右移都补0

负数：反码1 <—> 原码0

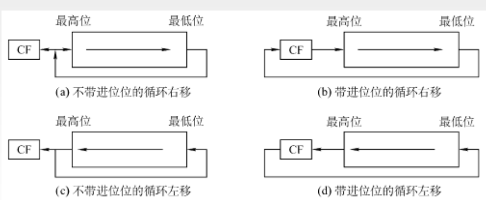
原码 1,0110101

反码 1,1001010

补码 1,1001011

循环移位

1 0 1 1 0 1 0 1



(a) 不带进位位的循环右移

(b) 带进位位的循环右移

(c) 不带进位位的循环左移

(d) 带进位位的循环左移

知识点回顾

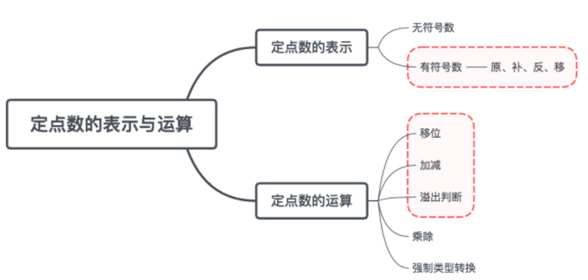
- 移位
  - 逻辑移位
  - 算术移位：左移相当于乘以基数，右移相当于除以基数
  - 空位添补规则
  - 循环移位

本节内容

定点数的表示和运算

加减运算
符号扩展
溢出判断

本节总览



符号扩展

设机器字长为8位（含1位符号位），$A = 15$，$B = -24$，求$[A+B]_补$和$[A-B]_补$

原码

A = +1111 → 0,1111 → 0,0001111
B = -11000 → 1,11000 → 1,0011000

补码

A = +1111 → 0,1111 → 0,0001111
B = -11000 → 1,01000 → 1,1101000

加减运算

设机器字长为8位（含1位符号位），$A = 15$，$B = -24$，求$[A+B]_补$和$[A-B]_补$

补码

A = +1111 → 0,1111 → 0,0001111
B = -11000 → 1,01000 → 1,1101000

$[A+B]_补 = [A]_补 + [B]_补$ = 0,0001111 + 1,1101000 = 1,1110111
原码：1,0001001　真值-9

$[A-B]_补 = [A]_补 + [-B]_补$ = 0,0001111 + 0,0011000 = 0,0100111　真值+39

$[-B]_补$：$[B]_补$连同符号位一起取反加1

$C = 124$，求$[A+C]_补$和$[B-C]_补$

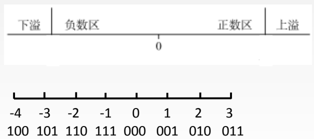
$[A+C]_补$ = 0,0001111 + 0,1111100 = 1,0001011　真值-117
$[B-C]_补$ = 1,1101000 + 1,0000100 =10,1101100　真值+108

溢出判断

设机器字长为8位（含1位符号位），A = 15，B = -24，求$[A+B]_{补}$和$[A-B]_{补}$

C = 124，求$[A+C]_{补}$和$[B-C]_{补}$

$[A+C]_{补} = \boxed{0}0001111 + \boxed{0}1111100 = \boxed{1}0001011$　　真值-117
$[B-C]_{补} = \boxed{1}1101000 + \boxed{1}0000100 = 1\boxed{0}1101100$　　真值+108



方法一：采用一位符号位
设A的符号为A_S，B的符号为B_S，运算结果的符号为S_S，则溢出逻辑表达式为

$$V = A_S B_S \overline{S_S} + \overline{A_S}\,\overline{B_S} S_S$$

若V=0，表示无溢出；
若V=1，表示有溢出。

溢出判断

设机器字长为8位（含1位符号位），A = 15，B = -24，求$[A+B]_{补}$和$[A-B]_{补}$

C = 124，求$[A+C]_{补}$和$[B-C]_{补}$

$[A+C]_{补} = \boxed{0}0001111 + \boxed{0}1111100 = \boxed{1}0001011$　　真值-117
$[B-C]_{补} = \boxed{1}1101000 + \boxed{1}0000100 = 1\boxed{0}1101100$　　真值+108

逻辑表达式
与：如ABC，表示A与B与C
仅当A、B、C均为1时，ABC为1
A、B、C中有一个或多个为0，则ABC为0

或：如A+B+C，表示A或B或C
仅当A、B、C均为0时，A+B+C为0
A、B、C中有一个或多个为1，则A+B+C为1

非：如 $\overline{A}$，表示A非
若A为1，则 $\overline{A}$ 为0
若A为0，则 $\overline{A}$ 为1

方法一：采用一位符号位
设A的符号为A_S，B的符号为B_S，运算结果的符号为S_S，则溢出逻辑表达式为

$$V = \boxed{A_S B_S \overline{S_S}} + \boxed{\overline{A_S}\,\overline{B_S} S_S}$$

若V=0，表示无溢出；
若V=1，表示有溢出。

A_S为1且B_S为1且S_S为0　或　A_S为0且B_S为0且S_S为1

溢出判断

设机器字长为8位（含1位符号位），A = 15，B = -24，求$[A+B]_{补}$和$[A-B]_{补}$

C = 124，求$[A+C]_{补}$和$[B-C]_{补}$

$[A+C]_{补}$ = 0,0001111 + 0,1111100 = 1,0001011　　真值-117
$[B-C]_{补}$ = 1,1101000 + 1,0000100 =10,1101100　　真值+108

方法二：采用一位符号位，根据数据位进位情况判断溢出

	符号位的进位C_S	最高数位的进位C_1
上溢	0	1
下溢	1	0

即：C_S与C_1不同时有溢出
处理“不同”的逻辑符号：异或⊕
溢出逻辑判断表达式为$V=C_S \oplus C_1$
若V=0，表示无溢出；V=1，表示有溢出。

异或逻辑：不同为1，相同为0
$0 \oplus 0 = 0$
$0 \oplus 1 = 1$
$1 \oplus 0 = 1$
$1 \oplus 1 = 0$

溢出判断

设机器字长为8位（含1位符号位），A = 15，B = -24，求$[A+B]_{补}$和$[A-B]_{补}$

C = 124，求$[A+C]_{补}$和$[B-C]_{补}$

$[A+C]_{补}$ = 0,0001111 + 0,1111100 = 1,0001011　　真值-117
$[B-C]_{补}$ = 1,1101000 + 1,0000100 =10,1101100　　真值+108

方法三：采用双符号位
正数符号为00，负数符号为11

$[A+C]_{补}$ = 00,0001111 + 00,1111100 = 01,0001011　　上溢
$[B-C]_{补}$ = 11,1101000 + 11,0000100 = 10,1101100　　下溢

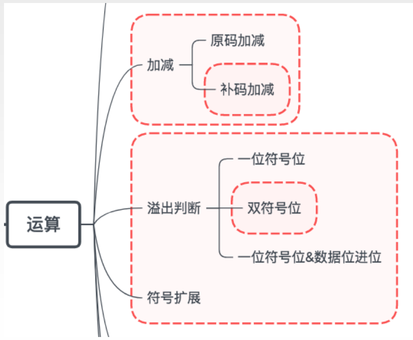
记两个符号位为$S_{S1}S_{S2}$，则$V=S_{S1} \oplus S_{S2}$
若V=0，表示无溢出；若V=1，表示有溢出。

$[A+B]_{补}$ = 00,0001111 + 11,1101000 = 11,1110111
$[A-B]_{补}$ = 00,0001111 + 00,0011000 = 00,0100111

11,1110111　右移1位：11,1111011
00,0100111　左移1位：00,1001110
00,0100111　左移2位：01,0011100　上溢

采用双符号位的移位运算：低位符号位参与移位，高位符号位代表真正的符号

知识点回顾

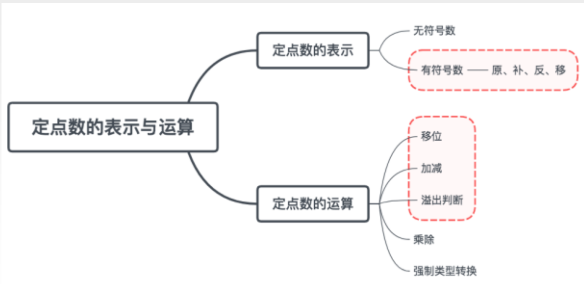


本节内容

定点数的表示和运算

乘法运算

本节总览



原码一位乘法

符号位	绝对值

符号位与数值位分开处理

设机器字长为5位（含1位符号位，$n=4$），$x = -0.1101$，$y = +0.1011$，采用原码一位乘法求$x \cdot y$

符号：一正一负，结果为负，即符号位 $= x_s \oplus y_s$

```
  0.1101
  0.1011
--------
    1101
   1101
  0000
 1101
--------
0.10001111
```

原码一位乘法

$|x|=00.1101$，$|y|=00.1011$

```
        00.0000
+|x|    00.1101
---------------
        00.1101
右移    00.0110
+|x|    00.1101
---------------
        01.0011
右移    00.1001
```

ACC	MQ	
00000	1011	
01101	1011	
00110	1101	1
10011	1101	1
01001	1110	11

原码一位乘法

设机器字长为5位（含1位符号位，$n=4$），$x = -0.1101$，$y = +0.1011$，采用原码一位乘法求$x\,y$

符号：一正一负，结果为负，即符号位 $= x_s \oplus y_s$

```
    0.1101
    0.1011
  --------
      1101
     1101
    0000
   1101
  --------
0.10001111
```

原码一位乘法

$|x|=00.1101$，$|y|=00.1011$

```
        00.0000
+|x|    00.1101
---------------
        00.1101
右移    00.0110
+|x|    00.1101
---------------
        01.0011
右移    00.1001
+0      00.0000
---------------
        00.1001
右移    00.0100
+|x|    00.1101
---------------
        01.0001
右移    00.1000
```

ACC	MQ	
00000	1011	
01101	1011	
00110	1101	1
10011	1101	1
01001	1110	11
01001	1110	11
00100	1111	011
10001	1111	011
01000	1111	1011

符号位$P_s=x_s \oplus y_s=1 \oplus 0=1$

$[x\,y]_原 = 1.10001111$

即$x\,y=-0.10001111$

王道考研/CSKAOYAN.COM

补码一位乘法

设机器字长为5位（含1位符号位，$n=4$），$x = -0.1101$，$y = +0.1011$，采用Booth算法求$x \cdot y$

$[x]_补=11.0011$，$[-x]_补=00.1101$，$[y]_补=0.1011$

```
          00.0000
+[-x]补   00.1101
-----------------
          00.1101
右移      00.0110
+0        00.0000
-----------------
          00.0110
右移      00.0011
+[x]补    11.0011
-----------------
          11.0110
右移      11.1011
+[-x]补   00.1101
-----------------
          00.1000
右移      00.0100
+[x]补    11.0011
-----------------
          11.0111
```

ACC	MQ	
000000	010110	
001101	010110	
000110	101011	0
000110	101011	0
000011	010101	10
110110	010101	10
111011	001010	110
001000	001010	110
000100	000101	0110
110111	000101	0110

y_n（高位）	y_{n+1}（低位）	操　作
0	0	部分积右移一位
0	1	部分积加$[X]_补$ 右移一位
1	0	部分积加$[-X]_补$ 右移一位
1	1	部分积右移一位

根据$y_{n+1} - y_n$判断：

$y_{n+1} - y_n = 0$，加0，右移一位

$y_{n+1} - y_n = 1$，加$[X]_补$，右移一位

$y_{n+1} - y_n = -1$，加$[-X]_补$，右移一位

$[x\,y]_补 = 1.01110001$

即$x\,y = -0.10001111$

王道考研/CSKAOYAN.COM

乘法运算总结回顾

乘法类型	符号位			累加次数	移位		
	参与运算	部分积	乘数		方向	次数	每位次数
原码一位乘法	否	2位	0位	n	右	n	1
补码Booth乘法	是	2位	1位	n+1	右	n	1

王道考研/CSKAOYAN.COM

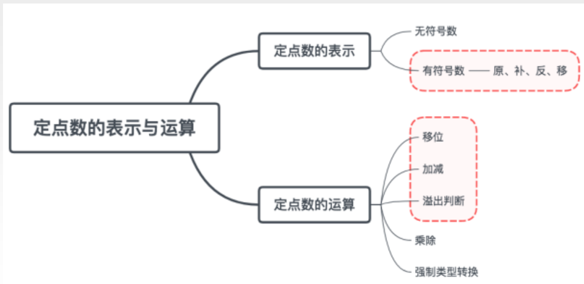
本节内容

定点数的表示和运算

除法运算

王道考研/CSKAOYAN.COM

本节总览



除法

符号位	绝对值

符号位与数值位分开处理

设机器字长为5位（含1位符号位，n=4），x=0.1011，y=0.1101，求x/y

```
            0.1101
0.1101 ) 0.10110
         0.01101
         0.010010
         0.001101
         0.00010100
         0.00001101
         0.000001110
```

x/y结果为0.1101，余数为0.00000111

原码恢复余数法

符号位	绝对值

符号位与数值位分开处理

设机器字长为5位（含1位符号位，n=4），x=0.1011，y=0.1101，采用原码加减交替除法求x/y

$|x|$=00.1011，$|y|$=00.1101，$[|y|]_{补}$=00.1101，$[-|y|]_{补}$=11.0011

	被除数	商
	00.1011	
$+[-\|y\|]_{补}$	11.0011	
	11.1110	0
$+[\|y\|]_{补}$	00.1101	
	00.1011	
左移	01.0110	
$+[-\|y\|]_{补}$	11.0011	
	00.1001	01
左移	01.0010	
$+[-\|y\|]_{补}$	11.0011	
	00.0101	011
	…	…

左移n次，上商n+1次

原码恢复余数法

符号位	绝对值

符号位与数值位分开处理

设机器字长为5位（含1位符号位，n=4），x=0.1011，y=0.1101，采用原码加减交替除法求x/y

$|x|$=00.1011，$|y|$=00.1101，$[|y|]_{补}$=00.1101，$[-|y|]_{补}$=11.0011

	被除数		商
	00.1011		
$+[-\|y\|]_{补}$	11.0011		
	11.1110	a	0
$+[\|y\|]_{补}$	00.1101	b	
	00.1011	a+b	
左移	01.0110	(a+b)×2 = 2a + 2b	
$+[-\|y\|]_{补}$	11.0011	(a+b)×2 - b = 2a + 2b – b = 2a + b	
	00.1001		01
左移	01.0010		
$+[-\|y\|]_{补}$	11.0011		
	00.0101		011
	…		…

左移n次，上商n+1次

本节内容

定点数的表示和运算

原码加减交替法
补码加减交替法

原码恢复余数法

符号位	绝对值

符号位与数值位分开处理

设机器字长为5位（含1位符号位，$n=4$），$x=0.1011$，$y=0.1101$，采用原码加减交替除法求x/y

$|x|=00.1011$，$|y|=00.1101$，$[|y|]_补=00.1101$，$[-|y|]_补=11.0011$

	被除数	
	00.1011	
$+[-\|y\|]_补$	11.0011	
	11.1110	a
$+[\|y\|]_补$	00.1101	b
	00.1011	a+b
左移	01.0110	(a+b)×2 = 2a + 2b
$+[-\|y\|]_补$	11.0011	(a+b)×2 - b = 2a + 2b – b = 2a + b
	00.1001	
左移	01.0010	
$+[-\|y\|]_补$	11.0011	
	00.0101	
	…	

a左移，加b

不恢复余数法：
被除数减去除数，即$|x|+[-|y|]_补$，
若结果为正，商1，左移，再减去除数；
若结果为负，商0，左移，再加上除数。

… 左移n次，上商n+1次

原码不恢复余数法

符号位	绝对值

符号位与数值位分开处理

设机器字长为5位（含1位符号位，$n=4$），$x=0.1011$，$y=0.1101$，采用原码加减交替除法求x/y

$|x|=00.1011$，$|y|=00.1101$，$[|y|]_补=00.1101$，$[-|y|]_补=11.0011$

	被除数
	00.1011
$+[-\|y\|]_补$	11.0011
	11.1110
左移	11.1100
$+[\|y\|]_补$	00.1101
	00.1001
左移	01.0010
$+[-\|y\|]_补$	11.0011
	00.0101
左移	00.1010
$+[-\|y\|]_补$	11.0011

ACC	MQ
01011	00000
11110	00000
11100	00000
01001	00001
10010	00010
00101	00011
01010	00110

原码不恢复余数法

符号位与数值位分开处理

设机器字长为5位（含1位符号位，$n=4$），$x=0.1011$，$y=0.1101$，采用原码加减交替除法求x/y

$|x|=00.1011$，$|y|=00.1101$，$[|y|]_补=00.1101$，$[-|y|]_补=11.0011$

	被除数
	00.1011
$+[-\|y\|]_补$	11.0011
	11.1110
左移	11.1100
$+[\|y\|]_补$	00.1101
	00.1001
左移	01.0010
$+[-\|y\|]_补$	11.0011
	00.0101
左移	00.1010
$+[-\|y\|]_补$	11.0011
	11.1101
左移	11.1010
$+[\|y\|]_补$	00.1101
	00.0111

若余数为负，需$+[|y|]_补$得到正确余数

ACC	MQ
01011	00000
11110	00000
11100	00000
01001	00001
10010	00010
00101	00011
01010	00110
11101	00110
11010	01100
00111	01101

$Q_s=x_s\oplus y_s=0\oplus0=0$
得$x/y=+0.1101$
余0.0111×2^{-4}

补码加减交替法

设机器字长为5位（含1位符号位，n=4），x=+0.1000，y=-0.1011，采用补码加减交替除法求x/y

$[x]_{补}$=00.1000，$[y]_{补}$=$\boxed{11}$.0101，$[-y]_{补}$=00.1011　　$[x/y]_{补}$=1.0101，余0.0111×2^{-4}

	被除数
	00.1000
$+[y]_{补}$	11.0101
	$\boxed{11}$.1101
左移	11.1010
$+[-y]_{补}$	00.1011
	$\boxed{00}$.0101
左移	00.1010
$+[y]_{补}$	11.0101
	$\boxed{11}$.1111
左移	11.1110
$+[-y]_{补}$	00.1011
	$\boxed{00}$.1001
左移	01.0010
$+[y]_{补}$	11.0101
	00.0111

ACC	MQ	
01000	00000	
11101	00001	
11010	00010	
00101	00010	
01010	00100	
11111	00101	
11110	01010	
01001	01010	
10010	10100	
00111	10101	末位恒置1

被除数和除数同号，则被除数减去除数；异号则被除数加上除数。

余数和除数同号，商1，余数左移一位减去除数；
余数和除数异号，商0，余数左移一位加上除数。
重复n次。

王道考研/CSKAOYAN.COM

除法运算总结回顾

除法类型	符号位参与运算	加减次数	移位		说明
			方向	次数	
原码加减交替法	否	N+1或N+2	左	N	若最终余数为负，需恢复余数
补码加减交替法	是	N+1	左	N	商末位恒置1

王道考研/CSKAOYAN.COM

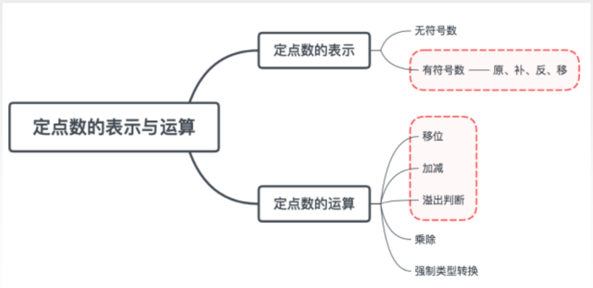
本节内容

定点数的表示和运算

强制类型转换

王道考研/CSKAOYAN.COM

本节总览



王道考研/CSKAOYAN.COM

强制类型转换

```
void main(){
    short x=-4321;                     //short型占用2个字节
    unsigned short y=(unsigned short)x;

    int a=165537, b=-34991;            //int型占用4个字节
    short c=(short)a, d=(short)b;      //short型占用2个字节

    short x=-4321;
    int m=x;
    unsigned short n=(unsigned short)x;
    unsigned int p=n;
}
```

x：1110 1111 0001 1111　　y：1110 1111 0001 1111　真值61215

无符号数与有符号数：
不改变数据内容，
改变解释方式。

长整数变短整数：
高位截断，保留低位。

a：0x000286a1
c：0x86a1　　真值-31071
b：0xffff7751
d：0x7751　　真值30545

短整数变长整数：
符号扩展。

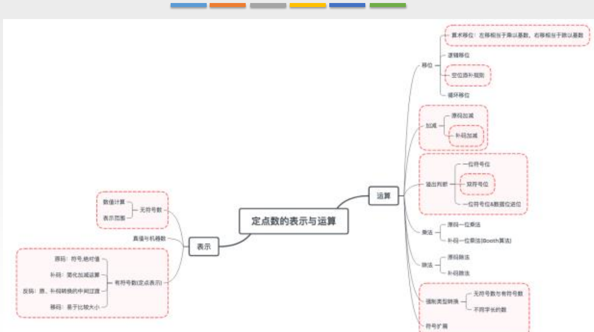
x：1110 1111 0001 1111
　0xef1f
m：1111 1111 1111 1111 1110 1111 0001 1111
　0xffffef1f　　真值-4321
n：1110 1111 0001 1111　　0xef1f　　真值61215
p：0000 0000 0000 0000 1110 1111 0001 1111
　0x0000ef1f　　真值61215

本节内容

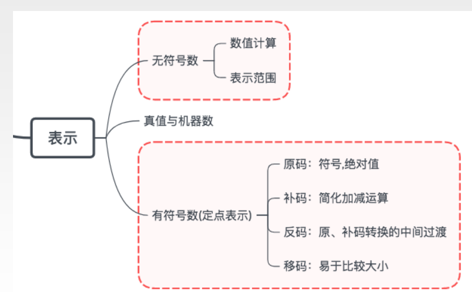
定点数的表示和运算

本节回顾

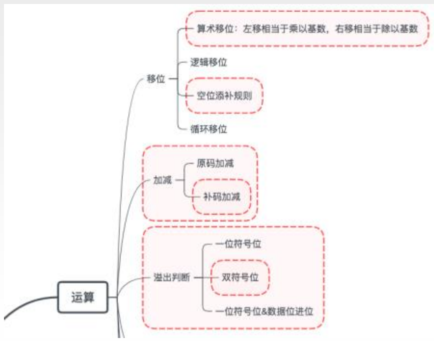
本节回顾



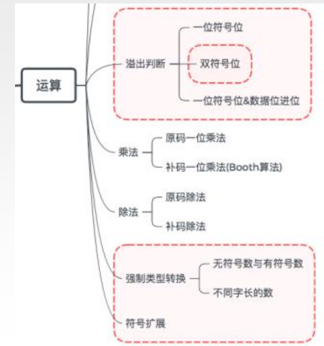
本节回顾



本节回顾


运算
移位
算术移位：左移相当于乘以基数，右移相当于除以基数
逻辑移位
空位添补规则
循环移位
加减
原码加减
补码加减
溢出判断
一位符号位
双符号位
一位符号位&数据位进位

本节回顾


运算
溢出判断
一位符号位
双符号位
一位符号位&数据位进位
乘法
原码一位乘法
补码一位乘法(Booth算法)
除法
原码除法
补码除法
强制类型转换
无符号数与有符号数
不同字长的数
符号扩展