原码

又称"符号+数值表示", 对于正数, 符号位为0, 对于负数、符号位为1, 其余各位表示数值部分。

将数的真值形式中“+”号用“0”表示，“-”号用“1”表示时，叫做数的原码形式，简称原码。若字长为n位，原码一般可表示为：

$$[X]_{原} = \begin{cases} X & 0 \le X < 2^{n-1} \\ 2^{n-1} - X & -2^{n-1} < X \le 0 \end{cases}$$

当X为正数时[X]原和X一样，即[X]原＝X。

当X为负数时 $[X]_{原} = 2^{n-1} - X$ 。由于X本身为负数，所以，实际上是将｜X｜数值部分绝对值前面的符号位上写成“1”即可。

例： $N_1 = +10011$ $N_2 = -01010$

$[N_1]_{原} = 010011$ $[N_2]_{原} = 101010$

原码表示的特点: 真值0有两种原码表示形式,

即 $[+0]_{原} = 00\ldots0$ $[-0]_{原} = 1\,0\ldots0$

引入反码和补码的原因

原码表示法比较直观，它的数值部分就是该数的绝对值，而且与真值、十进制数的转换十分方便。但是它的加减法运算较复杂。当两数相加时，机器要首先判断两数的符号是否相同，如果相同则两数相加，若符号不同，则两数相减。在做减法前，还要判断两数绝对值的大小，然后用大数减去小数，最后再确定差的符号，换言之，用这样一种直接的形式进行加运算时，负数的符号位不能与其数值部分一道参加运算，而必须利用单独的线路确定和的符号位。要实现这些操作，电路就很复杂，这显然是不经济实用的。为了减少设备，解决机器内负数的符号位参加运算的问题，总是将减法运算变成加法运算，也就引进了反码和补码这两种机器数。

1.3.3　反　码

对于正数，其反码表示与原码表示相同，对于负数，符号位为1，其余各位是将原码数值按位求反。

例：　$N_1 = +10011$　　$N_2 = -01010$

$[N_1]_{反} = 010011$　　$[N_2]_{反} = 1\ 10101$

真值0也有两种反码表示形式，即

$[+0]_{反} = 00\ldots0$　　$[-0]_{反} = 1\ 1\ldots1$

1.3.4 补 码

对于正数，其补码表示与原码表示相同，对于负数，符号位为1，其余各位是在反码数值的末位加"1".

例： $N_1 = +10011$ $N_2 = -01010$

$[N_1]_{补} = 010011$ $[N_2]_{补} = 1\ 10110$

真值0只有一种补码表示形式，即

$[-0]_{补} = [-0]_{反} + 1 = 1\ 1 \ldots 1 + 1$

$= \boxed{1}\ 0\ 0 \ldots 0$

丢弃

机器数的加、减运算

一、原码运算

1、符号位不参与运算,单独处理。

2、设A、B表示绝对值，有下列两类八种情况。

- $(+A)+(+B)=(+A)-(-B)$

  $(-A)+(-B)=(-A)-(+B)$

同号数相加或异号数相减，运算规则为绝对值相加，取被加(减)数的符号。

- $(+A)-(+B)=(+A)+(-B)$

  $(-A)-(-B)=(-A)+(+B)$

同号数相减或异号数相加。运算规则为绝对值相减，取绝大值较大者的符号。

例：N_1 =－0011，N_2 = 1011求[N_1 +N_2]$_{原}$和 [N_1 －N_2]$_{原}$。

解：[N_1]$_{原}$＝10011，[N_2]$_{原}$＝01011

求[N_1 +N_2]$_{原}$，绝对值相减，有

$$\begin{array}{rcccc} & 1 & 0 & 1 & 1 \\ -) & 0 & 0 & 1 & 1 \\ \hline & 1 & 0 & 0 & 0 \end{array}$$

结果取N_2的符号，即：[N_1 +N_2]$_{原}$＝01000

真值为： N_1 +N_2＝1000

求$[N_1-N_2]_{原}$，绝对值相加，有

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

结果取N_1的符号，即：

$$[N_1-N_2]_{原}=11110$$

真值为： $N_1-N_2=-1110$

二、补码运算

可以证明有如下补码加、减运算规则：

$$[N_1+N_2]_{补}=[N_1]_{补}+[N_2]_{补}$$

$$[N_1-N_2]_{补}=[N_1]_{补}+[-N_2]_{补}$$

此规则说明补码的符号位参与加减运算。

例： $N_1=-0011$，$N_2=1011$求$[N_1+N_2]_{补}$
和$[N_1-N_2]_{补}$。

解： $[N_1]_{补}=11101$， $[N_2]_{补}=01011$,
$[-N_2]_{补}=10101$

$[N_1+N_2]_{补}=11101+01011=01000$

$$
\begin{array}{rcccccc}
 & & 1 & 1 & 1 & 0 & 1 \\
+) & & 0 & 1 & 0 & 1 & 1 \\
\hline
丢弃 \longleftarrow & \boxed{1} & 0 & 1 & 0 & 0 & 0
\end{array}
$$

真值为： $N_1+N_2=1000$

$[N_1 - N_2]_{补} = 11101 + 10101$

$$
\begin{array}{rcccccc}
 & & 1 & 1 & 1 & 0 & 1 \\
+) & & 1 & 0 & 1 & 0 & 1 \\
\hline
丢弃 \longleftarrow & \boxed{1} & 1 & 0 & 0 & 1 & 0
\end{array}
$$

真值为：　$N_1 - N_2 = -1110$

三、反码运算

$$[N_1+N_2]_{反}=[N_1]_{反}+[N_2]_{反}$$

$$[N_1-N_2]_{反}=[N_1]_{反}+[-N_2]_{反}$$

当符号位有进位时，应在结果的最低位再加"1".

例： $N_1 = -0011$，$N_2 = 1011$ 求 $[N_1+N_2]_{反}$ 和 $[N_1-N_2]_{反}$。

解： $[N_1]_{反} = 11100$，　　$[N_2]_{反} = 01011$,

$[-N_2]_{反} = 10100$

$[N_1+N_2]_{反} = 11100+01011 = 01000$

$$
\begin{array}{rccccc}
 & 1 & 1 & 1 & 0 & 0 \\
+) & 0 & 1 & 0 & 1 & 1 \\
\hline
\boxed{1} & 0 & 0 & 1 & 1 & 1 \\
+) & & & & \longrightarrow & 1 \\
\hline
 & 0 & 1 & 0 & 0 & 0
\end{array}
$$

真值为： $N_1+N_2=1000$

$$[N_1 - N_2]_{反} = 11100+10100$$

$$
\begin{array}{r}
1\ 1\ 1\ 0\ 0 \\
+)\ \ 1\ 0\ 1\ 0\ 0 \\
\hline
\boxed{\mathbf{1}}\ 1\ 0\ 0\ 0\ 0 \\
+)\ \longrightarrow 1 \\
\hline
1\ 0\ 0\ 0\ 1
\end{array}
$$

真值为： $N_1 - N_2 = -1110$

n 位二进制补码的表数范围： $-2^{n-1} \le N \le 2^{n-1}-1$

十进制	二进制	十六进制	十进制	十六进制
n=8			n=16	
+127	0111 1111	7F	+32767	7FFF
+126	0111 1110	7E	+32766	7FFE
...	...	…	…	...
+2	0000 0010	02	+2	0002
+1	0000 0001	01	+1	0001
0	0000 0000	00	0	0000
-1	1111 1111	FF	-1	FFFF
-2	1111 1110	FE	-2	FFFE
...	...	…	…	...
-126	1000 0010	82	-32766	8002
-127	1000 0001	81	-32767	8001
-128	1000 0000	80	-32768	8000

无符号数

没有了符号位

无符号整数的表数范围：$0 \le N \le 2^{n-1}$

常用于表示地址

作业：

1.把下列数按权展开

$(4517.239)_{10}$ $(10110.0101)_2$ $(785.4AF)_{16}$ $(325.744)_8$

2.二进制计算

10111+101.101=?

10.01*1.01=?

1100-111.011=?

1001.0001/11.101=?

3.二进制转换为十进制、十六进制、八进制

1110101 10111.01 0.110101

4.十进制转换为二进制、八进制、十六进制（精确到小数点后5位）

29 0.207 33.333

作业：

5.写出下列数的原码、反码、补码。

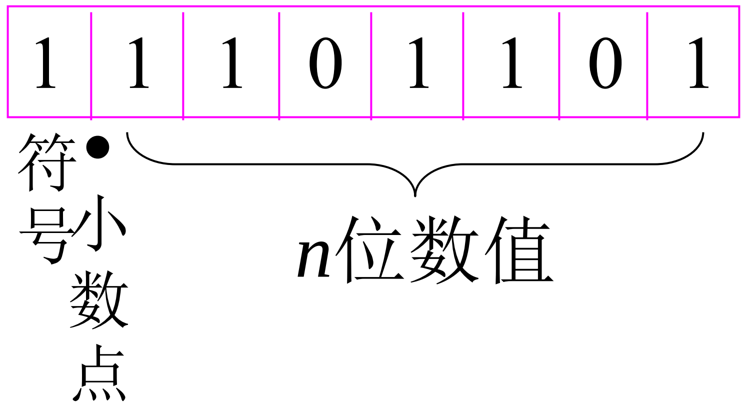
+13　　0　　　-38

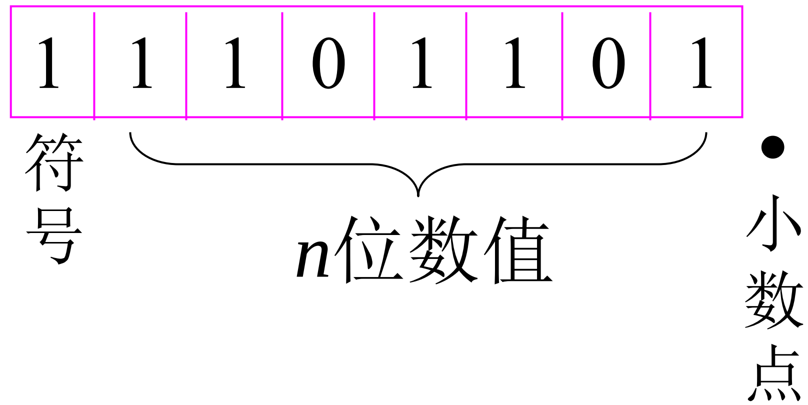
6.设N1=0000101，N2=0011010，
请分别用原码、反码、补码的方式计算[N1+N2]和[N1-N2] 。

数的定点表示与浮点表示

数的定点表示

即小数点的位置固定不变，一般可固定在任何位置，但通常固定在数值部份的最高位之前或最低之后，前者表示纯小数，后者表示纯整数。但机器中并没有小数点，仅仅是一种默认。





$$2^{-n} \leq |N| \leq 1-2^{-n} \qquad\qquad 1 \leq |N| \leq 2^{n}-1$$

如果运算结果小于2^{-n}(或1)，称出现了"下溢"，一般作为0处理，结果大于1- 2^{-n}(或2^{n}-1)，称出现了"上溢"，一般会停机或进入出错处理程序。

数的浮点表示

定点数的数域较小。若既要能表示很小的数，又要能表示很大的数，则采用浮点表示法比较合适。

一般形式为： $N=2^J\times S$

其中2^J称为N的指数部分，表示小数点的位置，S为N的尾数部分，表示数的符号和有效数字。

规格化数：尾数最高数值位非0，即$\frac{1}{2} \le |S| < 1.$

规格化数可以提高运算精度。例如：

$$1011 \rightarrow 2^{100} \times 0.1011 \rightarrow 2^{101} \times 0.01011$$

如果尾数的数值部分只有4位，则后一种表示将产生误差。

数码和字符的代码表示

十进制数（字）的二进制编码

简称为二——十进制码或BCD码，即用若干位二进制数来表示一位十进制数。

8421 BCD码

简称8421码。按4位二进制数的自然顺序，取前十个数依次表示十进制的0～9，后6个数不允许出现，若出现则认为是非法的或错误的。

8421码是一种有权码，每位有固定的权，从高到低依次为8, 4, 2, 1，如：

$$8421码0111=0\times8+1\times4+1\times2+1\times1=7$$

2421码

- 以2，4，2，1为权
- 按位取反得模9的补数

余3码

由8421码加3形成，是一种无权码。如果两个余3码相加没有进位，则和数要减3，否则和数要加3。

例如：0100+0110=0111　　1000+1001=$\boxed{1}$0 1 0 0

$$
\begin{array}{rcccc}
 & 0 & 1 & 0 & 0 \\
+) & 0 & 1 & 1 & 0 \\
\hline
 & 1 & 0 & 1 & 0 \\
-) & 0 & 0 & 1 & 1 \\
\hline
 & 0 & 1 & 1 & 1
\end{array}
\qquad
\begin{array}{rccccc}
 & & 1 & 0 & 0 & 0 \\
+) & & 1 & 0 & 0 & 1 \\
\hline
 & 1 & 0 & 0 & 0 & 1 \\
+) & & 0 & 0 & 1 & 1 \\
\hline
 & \boxed{1} & 0 & 1 & 0 & 0
\end{array}
$$

可靠性编码

能减少错误，发现错误，甚至纠正错误的编码称为可靠性编码。

防止传输过程中的错误（虽然数字信号比模拟信号可靠，更容易还原，但也不排除错误可能，绝大多数单错，多错几率很小）

一、格雷码

又称循环码，有多种形式，共同特点是任意相邻的两个代码之间仅有一位不同。

格雷码常用在计数器中，以防止多计数或少计数。

格雷码的单位距离特性可以降低其产生错误的概率，并且能提高其运行速度。例如，为完成十进制数7加1的运算， 当采用四位自然二进制码时，计数器应由0111变为1000， 由于计数器中各元件特性不可能完全相同，因而各位数码不可能同时发生变化，可能会瞬间出现过程性的错码。变化过程可能为0111→1111→1011→1001→1000。虽然最终结果是正确的，但在运算过程中出现了错码1111，1011，1001，这会造成数字系统的逻辑错误，而且使运算速度降低。若采用格雷码，由7变成8，只有一位发生变化，就不会出现上述错码，而且运算速度会明显提高。

格雷码的编码规则

见黑板

二、奇偶校验码

由信息位和校验位(冗余部分)两部分组成。校验位的取值可使整个校验码中的1的个数按事先的规完成为奇数或偶数。

奇偶校验码可发现奇数位错误，但不能发现偶数位错误。如$\boxed{1}$0011010→$\boxed{1}$0011011出现的错误, 但并不知道是哪一位出了错.虽然$\boxed{1}$0 0 1 1 0 1 0→$\boxed{1}$0 0 1 1 0 0 1出现了错误，但我们无法知道。

海明码

- 可发现错误，还能指出错误的位置

循环永余码CRC

- 理论依据*
- 工作原理
  - 1添加空数据位
  - 2和多项式相除求余数
  - 3和余数模2（类似异或）运算
  - 例题
- CRC可以编程实现，但在有些数字系统（甚至没有处理器，不支持指令系统，也就不可能编程）可由硬件部件实现。

字符代码

字符$A, B, \ldots, Z; a, b, \ldots, z; +, -, 0, 1, 2, \ldots, 9$等用ASCII(美国标准信息交换码)表示.

注：数字0, 1, …, 9与字符0, 1, …, 9是不同的.

ASCII码

		0	1	2	3	4	5	6	7
	$B_7B_6B_5$ 列码	0	0	0	0	1	1	1	1
		0	0	1	1	0	0	1	1
	$B_4B_3B_2B_1$ 行码	0	1	0	1	0	1	0	1
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	LF	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	SI	US	/	?	O	_	o	DEL

读码时，先读列码$B_7B_6B_5$，再读行码$B_4B_3B_2B_1$，则$B_7B_6B_5B_4B_3B_2B_1$即为某字符的七位ASCII码。例如字母K的列码是100，行码是1011，所以K的七位ASCII码是1001011。注意，表中最左边一列的A、B、……、F是十六进制数的六个数码。

作业

分别写出下列各数的8421BCD码、余3码和2421码表示形式。

（1）4368　　（2）39315　　（3）533　　（4）68930