

千锋嵌入式学院C语言培训

-位运算

Author: Richard. zhang



位运算概念

- 整数在计算机中用二进制位序列表示,位运算操作符对整数中的某些直接运算。

 - 63: 0000 0000 0000 0000 0000 0000 0011 1111
 - -1: 1111 1111 1111 1111 1111 1111 1111
- ▶ 位运算符的操作数必须为整型数:有符号和无符号的 char, short, int 和 long类型。
- ▶ 二进制的运算法则,相当于十进制的+-*/,但完全不同

0



运算符	含义
&	按位与
	按位或
^	按位异或
~	取反
<<	左移
>>	右移

源自清华 值得信赖



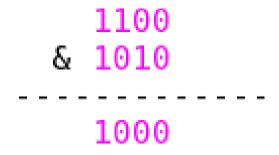
位运算赋值运算符

- ▶ 位运算操作符可以与等号合成赋值操作符
- 例如:
 - ∘ a &= b相当于a = a & b,
 - a <<= 2 相当于a = a << 2
- ▶常见错误
 - ∘ & 和 && 用错
 - |和||用错
 - 。 位操作与逻辑操作



"按位与"运算符(&)

- 两个操作数相应位是1,结果的对应位是1
- ▶ 两个操作数相应位有一个早O 结里对应位是O



- ▶ 与1不变,与0得零,用于将指定位清零
 - 将整数a高16位清零: a&= 0x0000fffff
- ▶ 与各位全是1的数0xffffffff,用于零测试



"按位与"运算符(&)

求 3&5 的结果,二进制表示出来,按位与得 1

```
0000 0011 (3)
& 0000 0101 (5)
0000 0001 (1)
```

▶ (-3)&(-5) 得多少呢?



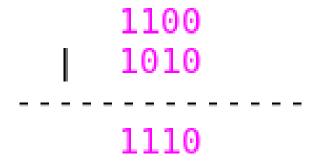
"按位与"运算符(&)

- 两个长度不一样的数据进行"&"操作,结果会如何呢?
- 注意, &、|、^运算符都是要做Usual Arithmetic Conversion的(其中有一步是Integer Promotion), ~ 运算符也要做Integer Promotion, 所以在C语言中其实 并不存在8位整数的位运算,操作数在做位运算之前都至 少被提升为int型。
- unsigned char c = 0xfc; unsigned int i = ~c; // I = 0xffffff03
- 尽量避免不同类型之间的赋值,以免出错。



"按位或"运算符(|)

- 两个操作数相应位是0,结果对应位也得0
- 两个操作数中只要有一个相应位是1,结果对应位得1



- ▶ 或 0 不变,或 1 置 1,用于将指定位置 1
 - 。 将整数a奇数位置1, a |= 0xaaaaaaaa
 - ∘ a |= 0, a不变



"按位或"运算符(|)

计算 3 | 5. 转换成二进制,按位或得7

```
0000 0011 (3)
| 0000 0101 (5)
-----
0000 0111 (7)
```

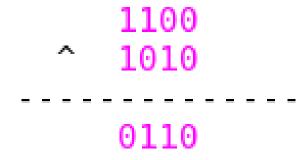
▶ (-3) | (-5) 按位或,计算得 -1

```
1111 1101 (-3)
| 1111 1011 (-5)
------1111 1111 (-1)
```



"按位异或"运算符(^)

- ▶ 二进制里的"半加",不带进位的加法运算
- 两个操作数相应位相同,结果对应位为0
- 两个操作数相应位不同,结果对应位为1



- 与 1 异或等于取反,与0异或不变,加法运算基础
 - ∘ 对整数a取反: a ^= 0xffffffff
- ▶ 与自身异或得 0



"按位异或"运算符(^)

```
计算 3^5, 转换成二进制, 按位或得6。
          0000 0011
                       (3)
          0000 0101
                       (5)
          0000 0110
                       (6)
▶ (-3) ^ (-5) 按位或, 计算得 6
          1111 1101 (-3)
                       (-5)
          1111 1011
          0000 0110
                       (6)
```



"按位异或"运算符(^)

▶ 练习:不使用临时变量,交换变量a和b的值.

```
    a = a ^ b; //记录下ab差异
    b = b ^ a; //从原来的b还原出原来的a
    a = a ^ b; //从原来的a还原出原来的b
```

可以类比下面代码理解

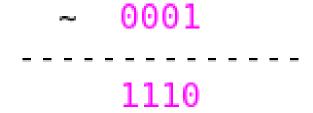
```
    a = a - b; //记录ab之差
    b = b + a; //原来的b加上差得到原来a
    a = b - a; //原来的a减去差得到原来b
```

▶ 异或可用作奇偶较验: 串口, 网络, RAID等



"按位取反"运算符(~)

- ▶ 操作数中为 0 的位,结果对应位置为 1
- ▶ 操作数中为 1 的位,结果对应位置为 0



- ▶ 把操作数和结果看作无符号数的话,相加等于0xffffffff
- ▶ 把操作数和结果看作有符号数的话,相加等于-1
- 取反操作使每一位都翻转,取负数补码时会用到(相反数 取反加1)



"按位取反"运算符(~)

~3 结果是几? 把3的二讲制表示, 按位取反~ 0000 0011 (3)

1111 1100 (-4)

▶ ~(-3) 结果是几?

~ 1111 1101 (-3) -----0000 0010 (2)

▶ 注意3按位取反,结果不是-3。



"左移"运算符(<<)

- 将一个数的各二进制位全部左移若干位,高位移出后舍去,低位补0
- ▶ 3 << 2得多少?把三的二进制表示向左依两位得12.

```
00000011 (3)

00001100 (12)

(-3) << 2, 计算得 -12

11111101 (-3)

11110100 (-12)
```



"右移"运算符(>>)

- 用来将一个数的各二进制位全部右移若干位,低位移出后 舍去; 高位补符号位。
 - 如果被移位的数据(运算符左操作数)是一个无符号数的时候,高位 移入"0"
 - 。如果被移位的数据是一个有符号数,且是一个正数的时候,那么高位移入"O"(即正数的符号位)
 - 如果被移位的数据是一个有符号数,且是一个负数的时候,具体移入的值由不同的编译系统决定。GCC编译系统下,移入的值是符号位(负数的符号位"1"),这时右移被称为"算数右移";



"右移"运算符(>>)

12 >> 2结果是多少? 经移位计算得 3

```
00001100 (12)
00000011 (3)
(-12) >> 2,经计算结果是 -3
11110100 (-12)
```

11111101 (-3)



移位操作与乘除法

- 从上两页的结果我们可以发现
 - a << 1 相当于 a * 2
 - a >> 1 相当于 a / 2
- 移位可以用作快速乘除法操作
 - 。 a*4 可以用 a << 2 代替
 - 。 a/4 可以用 a >> 2 代替
- ▶ 思考:如果左移过多,而改变了符号位呢?



6种位运算的基本概念

- 练习:从键盘输入一个无符号整数,打印出它的二进制表示位序列,每四位之间用空格分开。
- ▶ 练习: 取一个无符号整数a从右边开始的4~7位
- ▶ 练习:对无符号整数a循环右移,即每次移出的低位补到 高位。
- 练习:从键盘输入一个无符号整数,统计其二进制表示中 多少个1。
- 练习:从键盘输入一个无符号整数,统计其二进制表示中最长连续出现的1的序列长度。
- ▶ 编写一个程序,接收用户输入的一个无符号整数,判断这 _ 个整数是否是2的幂。



掩码的概念和应用

- 掩码是人为设定的整数值,用指定某几位。配合基本的位 运算,可以快速的对变量的各个位进行操作。
- 对某些位清0
 - int a, b, mask = 0x000000ff;
 a = 0x12345678;
 b = a & mask; //b = 0x00000078
- 对某些位置1
 - int $a = 0 \times 00000003$, $mask = 0 \times 0000$ ffc00
 - a |= mask; // a = 0x000ffc03



掩码的概念和应用

- 对某些位快速翻转
 - int a = 0x12345678, mask = 0x000ffc00;
 - int b = a^mask; // 把a的10-19位翻转



位运算可能的错误

- ▶ 错误:假设有一个整型变量a,a为16位。现在希望将a的最后一位置为"0",其他位保持不变:a & 0xfffe
- 为了预防这种移植性的问题,应该使用更健壮的代码:
 - a & (~1)



位运算可能的错误

▶ 以下代码正确吗?

▶ 最多移31位

```
    int a=31;
    int x=0xFFFFFFFF;
    printf("%d\n", 0xFFFFFFFF>>31);
    printf("%d\n", x >> a);
    printf("%d\n", 0xFFFFFFFF>>> a);//无符号数
    输出为: 1, -1, 1
```



位运算操作符

- 练习: utf16到utf-8汉字编解码,在公式
 1110xxxx 10xxxxxx 10xxxxxx
 中x的位置依次填入utf16的各位,就得到uft-8码。
- 从键盘输入一个无符号数,把各个字节反序排序,每个字节内部位顺序不变。
- > 实现一个函数,原型如下。c的低10位是数据位,第 int verify(int c)
 - 11位是较验位,对c进行偶较验。
- 从键盘输入4个字符,利用移位操作和或操作,分别把它们的值放入一个整数的4个字节当中。



位字段

```
位字段是建立在结构体声明上的,其中结构体的成员必须声明为整形(字符型),并且指明成员的宽度。
struct page_info {
    unsigned int wordcnt: 10;
    unsigned int bold: 1;
    unsigned int color: 5;
    unsigned int charnum: 8;
};
```



问题 ?

源自清华 值得信赖