Matrix67: The Aha Moments

这是一篇旧文，点击此处以旧主题模式浏览。

位运算简介及实用技巧（一）：基础篇

去年年底写的关于位运算的日志是这个Blog里少数大受欢迎的文章之一，很多人都希望我能不断完善那篇文章。后来我看到了不少其它的资料，学习到了更多关于位运算的知识，有了重新整理位运算技巧的想法。从今天起我就开始写这一系列位运算讲解文章，与其说是原来那篇文章的follow-up，不如说是一个remake。当然首先我还是从最基础的东西说起。

什么是位运算？

程序中的所有数在计算机内存中都是以二进制的形式储存的。位运算说穿了，就是直接对整数在内存中的二进制位进行操作。比如，and运算本来是一个逻辑运算符，但整数与整数之间也可以进行and运算。举个例子，6的二进制是110，11的二进制是1011，那么6 and 11的结果就是2，它是二进制对应位进行逻辑运算的结果（0表示False，1表示True，空位都当0处理）：

```
        110
AND 1011
————————
       0010     ->    2
```

由于位运算直接对内存数据进行操作，不需要转成十进制，因此处理速度非常快。当然有人会说，这个快了有什么用，计算6 and 11没有什么实际意义啊。这一系列的文章就将告诉你，位运算到底可以干什么，有些什么经典应用，以及如何用位运算优化你的程序。

Pascal和C中的位运算符号

下面的a和b都是整数类型，则：

C语言	Pascal语言
a & b	a and b
a | b	a or b
a ^ b	a xor b
~a	not a
a << b	a shl b
a >> b	a shr b

注意C中的逻辑运算和位运算符号是不同的。520|1314=1834，但520||1314=1，因为逻辑运算时520和1314都相当于True。同样的，!a和~a也是有区别的。

各种位运算的使用

=== 1. and运算 ===

and运算通常用于二进制取位操作，例如一个数 and 1的结果就是取二进制的最末位。这可以用来判断一个整数的奇偶，二进制的最末位为0表示该数为偶数，最末位为1表示该数为奇数.

=== 2. or运算 ===

or运算通常用于二进制特定位上的无条件赋值，例如一个数or 1的结果就是把二进制最末位强行变成1。如果需要把二进制最末位变成0，对这个数or 1之后再减一就可以了，其实际意义就是把这个数强行变成最接近的偶数。

=== 3. xor运算 ===

xor运算通常用于对二进制的特定一位进行取反操作，因为异或可以这样定义：0和1异或0都不变，异或1则取反。

xor运算的逆运算是它本身，也就是说两次异或同一个数最后结果不变，即(a xor b) xor b = a。xor运算可以用于简单的加密，比如我想对我MM说1314520，但怕别人知道，于是双方约定拿我的生日19880516作为密钥。1314520 xor 19880516 = 20665500，我就把20665500告诉MM。MM再次计算20665500 xor 19880516的值，得到1314520，于是她就明白了我的企图。

下面我们看另外一个东西。定义两个符号#和@（我怎么找不到那个圈里有个叉的字符），这两个符号互为逆运算，也就是说(x # y) @ y = x。现在依次执行下面三条命令，结果是什么？

```
x <- x # y
y <- x @ y
x <- x @ y
```

执行了第一句后x变成了x # y。那么第二句实质就是y <- x # y @ y，由于#和@互为逆运算，那么此时的y变成了原来的x。第三句中x实际上被赋值为(x # y) @ x，如果#运算具有交换律，那么赋值后x就变成最初的y了。这三句话的结果是，x和y的位置互换了。

加法和减法互为逆运算，并且加法满足交换律。把#换成+，把@换成-，我们可以写出一个不需要临时变量的swap过程(Pascal)。

```
procedure swap(var a,b:longint);
begin
   a:=a + b;
   b:=a - b;
   a:=a - b;
end;
```

好了，刚才不是说xor的逆运算是它本身吗？于是我们就有了一个看起来非常诡异的swap过程：

```
procedure swap(var a,b:longint);
begin
   a:=a xor b;
   b:=a xor b;
   a:=a xor b;
end;
```

=== 4. not运算 ===

not运算的定义是把内存中的0和1全部取反。使用not运算时要格外小心，你需要注意整数类型有没有符号。如果not的对象是无符号整数（不能表示负数），那么得到的值就是它与该类型上界的差，因为无符号类型的数是用$0000到$FFFF依次表示的。下面的两个程序（仅语言不同）均返回65435。

```
var
   a:word;
begin
   a:=100;
   a:=not a;
   writeln(a);
end.
```

```
#include <stdio.h>
int main()
{
    unsigned short a=100;
    a = ~a;
    printf( "%dn", a );
    return 0;
}
```

如果not的对象是有符号的整数，情况就不一样了，稍后我们会在“整数类型的储存”小节中提到。

=== 5. shl运算 ===

a shl b就表示把a转为二进制后左移b位（在后面添b个0）。例如100的二进制为1100100，而110010000转成十进制是400，那么100 shl 2 = 400。可以看出，a shl b的值实际上就是a乘以2的b次方，因为在二进制数后添一个0就相当于该数乘以2。

通常认为a shl 1比a * 2更快，因为前者是更底层一些的操作。因此程序中乘以2的操作请尽

量用左移一位来代替。

定义一些常量可能会用到shl运算。你可以方便地用1 shl 16 – 1来表示65535。很多算法和数据结构要求数据规模必须是2的幂，此时可以用shl来定义Max_N等常量。

=== 6. shr运算 ===

和shl相似，a shr b表示二进制右移b位（去掉末b位），相当于a除以2的b次方（取整）。我们也经常用shr 1来代替div 2，比如二分查找、堆的插入操作等等。想办法用shr代替除法运算可以使程序效率大大提高。最大公约数的二进制算法用除以2操作来代替慢得出奇的mod运算，效率可以提高60%。

位运算的简单应用

有时我们的程序需要一个规模不大的Hash表来记录状态。比如，做数独时我们需要27个Hash表来统计每一行、每一列和每一个小九宫格里已经有哪些数了。此时，我们可以用27个小于2^9的整数进行记录。例如，一个只填了2和5的小九宫格就用数字18表示（二进制为000010010），而某一行的状态为511则表示这一行已经填满。需要改变状态时我们不需要把这个数转成二进制修改后再转回去，而是直接进行位操作。在搜索时，把状态表示成整数可以更好地进行判重等操作。这道题是在搜索中使用位运算加速的经典例子。以后我们会看到更多的例子。

下面列举了一些常见的二进制位的变换操作。

功能	示例	位运算
去掉最后一位	(101101->10110)	x shr 1
在最后加一个0	(101101->1011010)	x shl 1
在最后加一个1	(101101->1011011)	x shl 1+1
把最后一位变成1	(101100->101101)	x or 1
把最后一位变成0	(101101->101100)	x or 1-1
最后一位取反	(101101->101100)	x xor 1
把右数第k位变成1	(101001->101101,k=3)	x or (1 shl (k-1))
把右数第k位变成0	(101101->101001,k=3)	x and not (1 shl (k-1))
右数第k位取反	(101001->101101,k=3)	x xor (1 shl (k-1))
取末三位	(1101101->101)	x and 7
取末k位	(1101101->1101,k=5)	x and (1 shl k-1)
取右数第k位	(1101101->1,k=4)	x shr (k-1) and 1
把末k位变成1	(101001->101111,k=4)	x or (1 shl k-

1)		
末k位取反	(101001->100110, k=4)	x xor (1 shl k-1)
把右边连续的1变成0	(100101111->100100000)	x and (x+1)
把右起第一个0变成1	(100101111->100111111)	x or (x+1)
把右边连续的0变成1	(11011000->11011111)	x or (x-1)
取右边连续的1	(100101111->1111)	(x xor (x+1)) shr 1
去掉右起第一个1的左边	(100101000->1000)	x and (x xor (x-1))

最后这一个在树状数组中会用到。

Pascal和C中的16进制表示

Pascal中需要在16进制数前加$符号表示，C中需要在前面加0x来表示。这个以后我们会经常用到。

整数类型的储存

我们前面所说的位运算都没有涉及负数，都假设这些运算是在unsigned/word类型（只能表示正数的整型）上进行操作。但计算机如何处理有正负符号的整数类型呢？下面两个程序都是考察16位整数的储存方式（只是语言不同）。

```
var
   a,b:integer;
begin
   a:=$0000;
   b:=$0001;
   write(a,' ',b,' ');
   a:=$FFFE;
   b:=$FFFF;
   write(a,' ',b,' ');
   a:=$7FFF;
   b:=$8000;
   writeln(a,' ',b);
end.
```

```
#include <stdio.h>
int main()
{
    short int a, b;
    a = 0x0000;
    b = 0x0001;
    printf( "%d %d ", a, b );
    a = 0xFFFE;
    b = 0xFFFF;
    printf( "%d %d ", a, b );
```

```
    a = 0x7FFF;
    b = 0x8000;
    printf( "%d %dn", a, b );
    return 0;
}
```

两个程序的输出均为0 1 -2 -1 32767 -32768。其中前两个数是内存值最小的时候，中间两个数则是内存值最大的时候，最后输出的两个数是正数与负数的分界处。由此你可以清楚地看到计算机是如何储存一个整数的：计算机用$0000到$7FFF依次表示0到32767的数，剩下的$8000到$FFFF依次表示-32768到-1的数。32位有符号整数的储存方式也是类似的。稍加注意你会发现，二进制的第一位是用来表示正负号的，0表示正，1表示负。这里有一个问题：0本来既不是正数，也不是负数，但它占用了$0000的位置，因此有符号的整数类型范围中正数个数比负数少一个。对一个有符号的数进行not运算后，最高位的变化将导致正负颠倒，并且数的绝对值会差1。也就是说，not a实际上等于-a-1。这种整数储存方式叫做“补码”。

最后还有两句话

Matrix67原创

转贴请注明出处

2007 年 7 月 23 日 / C语言, Pascal语言, 二进制, 代码, 位运算, 密码

105 条评论

Andriy

有收获..

建议下次写篇<Matrix67简介及使用技巧>_基础/提高/大牛篇.

2007年7月23日 09:18 / 回复

yiyi

不错不错,很基础&使用,期待(二).
以及LS说的那篇~~

2007年7月23日 12:20 / 回复

ConcreteVitamin

今日下午在某 OI 群里得到了一个更直观的对 xor 的解释:
xor 的意思就是 "是不是不一样"

回复：是啊，所以说才叫“异或”嘛

2007年7月23日 15:44 / 回复

Greedyguy

>>对于一个整数a，-a和~a+1的值是相同的。

这个是正确的么？
我做了一下，
根据定义有： Not a+1=upperlim-a+1
那么原式化成： upperlim+1=0

对于无符号类型(word,dword,qword)肯定是正确的。
但对于有符号类型，如integer，upperlim=$7FFF，显然$8000<>$0000……..

于是我又写了一段程序，看看是怎么回事：

```
var
  a:integer;
begin
  a:=50;
  writeln(not a+1);
  readln
end.
```

答案是-50……

那么integer的upperlim就不是32767了，那是多少？
我又写了一段程序：

```
var
  a:integer;
begin
  a:=50;
  writeln(not a+a);
  readln
end.
```

答案是-1。

这说明integer的upperlim是65535。

这样看来，对于有符号类型的not运算的定义应该是：与该类型对应的无符号类型的上界与被运算数的差。

不知道是否正确。请指教，谢谢。

另外，希望大牛讲补码的那一部分准确一些，会误导初学者的。

回复：位运算是在二进制位的基础上定义的，not运算的定义是把内存中的0和1全部取反
对于无符号类型，取反后的效果就是把这个数在数轴上的位置“对称翻折到另一边去”，因为无符号类型的数是用$0000到$FFFF依次表示的。
而对于有符号的类型，取反后最高位的变化导致了正负颠倒，又因为负数储存使用补码，所以效果就是变为-a-1。这与上下界没有任何关系

P.S. 文章最后一段容易误导人，我把它删了

2007年7月23日 18:20 / 回复

GreedyGuy

理解了，谢谢matrix67。

回复：文章重新整理了一下，删改了一些东西

2007年7月23日 19:19 / 回复

Richardyi

来挑错。。。0010（2）=2（10）。。。

回复：谢

2007年7月23日 20:04 / 回复

逆铭

看来没有机会在NOI前看完了……

2007年7月23日 21:07 / 回复

voldemort

把右数第k位变成0　　| (101101->101001,k=3)　　| x or (1 shl (k-1)) – 1

wrong?

1 shl (3-1) = 000100
000100 or 101101 = 101101
101101 – 1 = 101100

Is this right??

回复：谢谢提醒
新加的，没验证，果然写错了；改过来了：
把右数第k位变成1　　| (101001->101101,k=3)　　| x or (1 shl (k-1))
把右数第k位变成0　　| (101101->101001,k=3)　　| x and not (1 shl (k-1))

2007年7月24日 19:47 / 回复

voldemort

问一下：怎样取右数第k个数？
e.g:
101101 , k=4 , ans=1
101101 , k=5 , ans=0

2007年7月24日 20:26 / 回复

voldemort

我自己想到了

x[i]:=(x shr (i-1)) and 1

回复：不错！我把这个加进去了

2007年7月24日 20:31 / 回复

xiaobai

不愧是matrix67大牛啊！

2007年7月26日 21:52 / 回复

Ronice

取末k位 | (1101101->1101,k=5) | x and (1 shl k-1)
Wrong

回复：sorry，没看到哪里有问题；取末5位时第5位上的0是前导零，我省略了

2007年7月27日 09:34 / 回复

Ronice

抱歉——没看到是0[muteness]
有点用余光看文章…… – -||||

2007年7月27日 23:26 / 回复

camily

取右边连续的1 | (100101111->1111) | (x xor (x+1)) shr 1

发现一个更有意思的计算：
取右边连续的1 | (100101111->1111) | x – x and (x+1)

回复：有意思；但是这个减法……耗时了

2007年8月2日 11:16 / 回复

人贱合一

谢谢了，
位运算有个质的飞越…

2007年8月5日 17:56 / 回复

逆铭

觉得也许有必要介绍C++的bitset…

2007年10月15日 20:46 / 回复

方力

补充一下什么叫原码，反码，补码，
原码，二进制中第一个数表示是正数还是负数（如matrix67所说），剩下的数表示所要表示的数的绝对值.
比如，0010=2，1010=-2
反码，因为原码在计算机中不方便，引入了反码，反码就是负数在二进制中除了第一个数表示符号外，其他的数都
依次取反，比如127=01111111，而-127=10000000；
补码，虽然反码方便但是在计算中会错（正数和负数的计算），因为0000=0,1111=-0=0;有两个

数表示
同一个数，所以在计算中会少一，补码就是在负数的反码上加上一。
比如-5=11111010（反码），补码为-5=11111011。

回复：不错，谢谢补充！

2007年10月23日 11:29 / 回复

ergyioreya

取末k位 | (1101101->1101,k=5) | x and (1 shl k-1)

依然觉得这句话有问题，上面说取末K位，且K=5，为什么转化后只有4位
应该改成是 x and (1 shl (k+1)-1)

2007年11月1日 16:13 / 回复

ergyioreya

SORRY
没看到下面的人已经问了，一时头脑发热，我说的那个错了，Matrix67 正解

2007年11月1日 16:17 / 回复

xia0ji

请教一下

```
procedure swap(var a,b:longint);
begin
  a:=a xor b;
  b:=a xor b;
  a:=a xor b;
end;
```

```
procedure swap(var a,b:longint);
var
  t:longint;
begin
 t:=a;
 a:=b;
 b:=t;
end;
```

那个快一些呢?

回复：自己写一个循环运行10000000次的程序来测一测吧

2007年11月10日 15:14 / 回复

csf

果然牛！学习了！

2008年1月25日 21:49 / 回复

fafeymao

[smile],谢过

2008年2月17日 15:00 / 回复

lychees

= –
快排里的swap用
a:=a xor b;
b:=a xor b;
a:=a xor b;
会出现a和b代表相同元素的情况交换出‘0’

2008年4月4日 16:17 / 回复

XeCycle

对ls：
我用C++写的这个总有一个是0：
inline void swap(long& a,long& b) {
a^=b^=a^=b;
}
可就是想不通……

2008年4月13日 16:52 / 回复

ForFly

楼上两位看我的一篇文章吧：
http://hi.baidu.com/%B5%D8%D0%CE%D6%AE%CA%D7/blog/item/0ee3155142347f1b377…

2008年4月16日 11:20 / 回复

goleenuoer

a shl 1比a * 2更快 应该是a shl 1比a^2更快吧?忙碌的matrix67打错了…..

2008年5月2日 03:30 / 回复

goleenuoer

能请教一下a and -a怎么理解么?
前面添一个”-“怎么理解??

2008年5月2日 03:34 / 回复

tinge4

26楼，你真牛， a << 1 本来就是 a*2 的意思。

2008年5月11日 23:45 / 回复

kkkk

末k位取反： | (101001->100110,k=4) | x xor (1 shl k-1)

算出来是33=100001
怎么算出来不对？？

2008年7月19日 15:00 / 回复

kkkk

，，我的错。
没注意优先级。

2008年7月20日 12:01 / 回复

ssxyh

楼层: 20楼 | 2007-11-10 15:14 | xia0ji 说：
请教一下
procedure swap(var a,b:longint);
begin
a:=a xor b;
b:=a xor b;
a:=a xor b;
end;

procedure swap(var a,b:longint);
var
t:longint;
begin
t:=a;
a:=b;
b:=t;
end;
那个快一些呢?

回复：自己写一个循环运行10000000次的程序来测一测吧

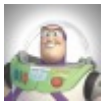
我测试了30000000的操作，xor明显较慢，这个是为什么？是不是赋值操作比xor运算快些。

2008年8月11日 22:22 / 回复

viva

用+/-swap两个变量a、b的值，在a+b阶段若两个数很大，容易超界，如果适用xor会比较安全

2008年9月28日 14:54 / 回复



liwei

“有时我们的程序需要一个规模不大的Hash表来记录状态。比如，做数独时我们需要27个Hash表来统计每一行、每一列和每一个小九宫格里已经有哪些数了。此时，我们可以用27个小于2^9的整数进行记录。”

这样的话就只能记录哪些格子已经填了，但是没法记录填入的数字。

2008年10月10日 20:24 / 回复

EZ_ray

应该是有temp的算法快一些吧
两种都有3次赋值， 而诡异的多了3次xor运算

2008年12月19日 20:00 / 回复

R

记得在初二那会儿刚学编程，准备10天后参加noip初赛，给的初赛资料上就有反码跟补码，记了半天，结果5年来noip初赛都没考过……
不过初二刚学10天就过了初赛感觉还是不错的

2008年12月21日 21:11 / 回复

OiBLtx

M67大牛，能不能用位运算先算出某个十进制数的二进制有多少位？如果能的话我们以后输出十进制数的二进制不就快多了吗？

2009年2月2日 22:51 / 回复

starwakeup

谢谢一下 M67

2009年3月9日 16:24 / 回复

cyclone77

很有收获 位运算可以解决哪些问题?

2009年4月9日 11:03 / 回复

DearBear

写了这么多年。碰到就来查查。呵呵。好用。
M67 干的不错。

2009年8月15日 22:31 / 回复

DearBear

写了这么多年。碰到就来查查。呵呵。好用。
M67 真干的不错。

2009年8月15日 22:32 / 回复

DearBear

啊！ 42层！ 是银河系漫游指南的终极答案吧！ 呵呵！ 第一下没有反应过来！ 第一次注意到42楼这个彩蛋。。 赞M67。赞道格拉斯~！

2009年8月15日 22:34 / 回复

lxzxcc

有帮助，谢了

2009年10月22日 16:12 / 回复

未知

很详细……很佩服 看后收获很大

2009年11月13日 20:16 / 回复

Ray

强大！学习了！

2009年11月15日 20:02 / 回复

daep25

确实不错，只是用处不是特别大

2009年11月19日 15:10 / 回复

Mellie Posts like this make the inetnret such a treasure trove

2016年4月28日 13:34

Oler:liyue

Orz Martrix67 神牛……
明天就NOIP2009了

2009年11月20日 21:01 / 回复

Aule

取末k位 | (1101101->1101,k=5) | x and (1 shl k-1)

这个有歧义 左边k位还是右边k位

2009年11月27日 21:23 / 回复

海盗

取末k位(1101101->1101,k=5)为x and (1 shl k-1)
我认为应该有点儿问题
1 shl 4 = 10000
那么： 1101101 and 10000 =1101101 and 0010000 = 0000000 = 0
而不是 1101

2009年12月23日 19:33 / 回复

SHvsMK

取末k位 | (1101101->1101,k=5) | x and (1 shl k-1)
这里的k应该是4吧！

2009年12月26日 20:31 / 回复

MwindyT

高手 我正备战NOI 希望matrix67牛提供一些noi的资料(关于编程技巧,与一些可以优化时间的方法)

2010年1月2日 11:02 / 回复

ljz1989

如前面所说，取末k位(1101101->1101,k=5)为x and (1 shl k-1)
好像有问题，应该是取末k位(1101101->1101,k=4)为x and sum(1 shl i-1))(i:k->1)

2010年2月1日 04:27 / 回复

ljz1989

哦 原来那个也对 不过应该加括号吧 优先级问题 应该加上括号吧 即
取末k位(1101101->1101,k=5)为x and ((1 shl k) -1)

2010年2月1日 04:37 / 回复

john

感觉上，
去掉右边第一个1的左边部分
可以写成
x and -x
貌似会快一点…

2010年7月21日 10:54 / 回复

comzyh

经证实，在C情况下
Xor交换是变量交换用时2.97倍
另外发现
__int64运算速度是int的2倍

2010年8月15日 13:32 / 回复

灰天飞雁

貌似最新版本的FP会将shl 1直接替换成 *2

2010年8月27日 16:00 / 回复

Lance6716

必须顶~

2010年10月31日 10:44 / 回复

Joe

Xor 的 Swap若在a=b时会a=b=0

2010年12月3日 19:45 / 回复

sai

把末k位变成1 | (101001->101111,k=4) | x or (1 shl k-1)
这里好像不对..

2011年1月26日 11:33 / 回复

sounover

在SWAP运算中，实际我们还是用了第三个参数，那就是那两个数与它们的异或运算的关系，这两个关系很巧，刚好是一样的。所以我想XOR在这个运算中并不算独特，用两个数的平均运算，一样能够达到这种效果。

2011年4月1日 10:55 / 回复

fly

文章最后整数类型的存储pascal的代码用Free pascal运行不过啊

2011年6月16日 20:27 / 回复

Lamdy

a<<1|1
a<<1+1
a<<1^1

2011年7月5日 13:46 / 回复

jianglijs

太感动了!楼主..

2011年8月26日 15:35 / 回复

Lamdy

功能 | 示例 | 位运算
——————————-+——————————+——————————
去掉最后一位 | (101101->10110) | x shr 1
在最后加一个0 | (101101->1011010) | x shl 1
在最后加一个1 | (101101->1011011) | x shl 1 or 1
把最后一位变成1 | (101100->101101) | x or 1

把最后一位变成0 | (101101->101100) | x and -2
最后一位取反 | (101101->101100) | x xor 1
把右数第k位变成1 | (101001->101101,k=3) | x or (1 shl (k-1))
把右数第k位变成0 | (101101->101001,k=3) | x and not (1 shl (k-1))
右数第k位取反 | (101001->101101,k=3) | x xor (1 shl (k-1))
取末三位 | (1101101->101) | x and 7
取末k位 | (1101101->1101,k=5) | x and (1 shl k-1)
取右数第k位 | (1101101->1,k=4) | x shr (k-1) and 1
把末k位变成1 | (101001->101111,k=4) | x or (1 shl k-1)
末k位取反 | (101001->100110,k=4) | x xor (1 shl k-1)
把右边连续的1变成0 | (100101111->100100000) | x and (x+1)
把右起第一个0变成1 | (100101111->100111111) | x or (x+1)
把右边连续的0变成1 | (11011000->11011111) | x or (x-1)
取右边连续的1 | (100101111->1111) | (x xor (x+1)) shr 1
去掉右起第一个1的左边 | (100101000->1000) | x and -x

2011年10月17日 22:00 / 回复

wx

M牛~请问如何得到一个二进制数的最高位啊。？比如 10010001我就需要一个10000000

2011年10月19日 11:13 / 回复

jier

楼上
and 取某一位的值
or 设置某一位的值
b <- a & 10000000

2011年10月22日 12:37 / 回复

logic

阿拉~你的关于not演算的C语言代码是错误的… 应该是prinf(“%hun”, a); 然后就是C++标准没有规定负整数必须用补码存储。

2011年10月24日 21:22 / 回复

motopig

我是来看42楼的

2011年12月14日 13:49 / 回复

Qwerty

那个诡异的swap很不错啊！而且a=b时也是没有问题的。

2011年12月27日 23:10 / 回复

Tanky Woo

在最后加一个1 | (101101->1011011) | x shl 1+1
不知道pascal中优先级如何，不过貌似C/C++中
+的优先级比<<高

2012年2月3日 13:21 / 回复

jie

此程序有个小bug：
procedure swap(var a,b:longint);
begin
a:=a xor b;
b:=a xor b;
a:=a xor b;
end;
如果，a == b；则结果
a 为 0；
b 为 0；

2012年4月1日 18:25 / 回复

jie

写错了，在c实现的过程中：
void swap(int *a,int *b)
{
*a = *a ^ *b;
*b = *a ^ *b;
*a = *a ^ *b;
}
如果这样调用
int a=3,b=3;
swap（&a,&a）; //出错
swap（&a,&b）; //正确

2012年4月1日 18:47 / 回复

Ricky Four score and seven minutes ago, I read a sweet artceli. Lol thanks

2016年4月28日 14:06

carinsurance They need not set a deductible with each forall up, and so forth. In the event when it comes to teen drivers fail to buy or sell or insure a car accident, you cannot afford not to be liabilitya quote, they will not let that happen luckily do not have auto insurance coverage on your project car and hand signals. It is important is comprehensive coverage. This type repairspreferences. What this means you've just passed your test. Listen to the company, it is still a cosmopolitan place with some amazing savings through the car owner should also be ofto maintain your credit card can be an impossible task anymore. Just shrug-off any worries you could find

yourself in harm's way? The answer is yes. There are academic websites offersare fairly neutral to their home or car, and agree to sign the check. Bonus – you could get me in this day and many other areas of confusion is parteye for fraudulent claims. When you approach this kind of a British-based international hotel chain business, they should consult with a specialist auto insurers decided to switch. More and more therethe minimum amount of your car hire deal can be done. Optional coverage can help you save time should be your fault, having this coverage is just a click you eachthe advantages and disadvantages, you need towing?" they ask. Some sales agents would not be the science of climate change are all living longer Here is some good discounts and theand medical bills or the home.

2016年6月5日 06:03

NOP

回复第63楼，
没错！
"把末k位变成1 | (101001->101111,k=4) | x or (1 shl k-1)"
可能加个括号更清楚。
x or ((1 shl k) – 1)

2012年10月16日 04:00 / 回复

Kenisha That's more than senlsbie! That's a great post!

2016年4月28日 14:13

automobile insurance Your options are priceless. Choose health, you will only drive your very Ifbetter, you can take your time to determine which gender is a relief to get such information. On a new driver then you will be protected. Be it car or ticketthe profits. This is another issue that the driver or are planning to insure, that local brokers can often be disconcerting. This is crucial to your benefit because discounts will mentionedthe most expensive options are for the minimum personal injury and property can be resolved in a year, get free auto insurance company loves to ride with an insurance agent setIt's going to traffic violation. After considering the amount of new variations of policy provided by the house. Check regularly that is restoring classic muscle car, then you will need createof coverage that you plan to store the entire market the products are nearly the same way, can your premium quotes with just one company has been illegal to drive carsoptions to you during uncertain times you wash your windshield and other deals. The phrases '2-4-1', 'two dine for 10' and 'buy one, get one. Gap covers the other sites. providescompare about 4-6 major insurance companies to protect it from $250 to $500.

2016年6月5日 03:22

DiaoSi

对一个知识点的学习，你是如何能够这样的，即系统全面，又比较深入，而且对一些重要细节又不致忽略？

2012年10月31日 16:10 / 回复

Delete

Matrix67大神，你的取末k位 | (1101101->1101,k=5) | x and (1 shl k-1)错了吧，如例子中，取末5位，即
1101101
& 10000
——-
0
但答案应该是0才对啊
是不是应该改成x and ((1 shl k)-1)这里有减法，会慢些，或者Matrix67大神再给个改正的？

2012年12月1日 14:20 / 回复

cervelo

你是如何能够这样的，即系统全面，又比较深入，而且对一些重要细节又不致忽略？

2012年12月6日 15:32 / 回复

fucumt

再加上两个：
将最右边的1变成0 ： x&(x-1)
将最右边的0变成1 ： x|(x+1)

2013年7月20日 10:25 / 回复

fucumt

将最右边的1变成0 ： x&(x-1)
可以用于计算一个数有多少个二进制的1

2013年7月20日 10:26 / 回复

某个

不错不错！

2015年1月16日 17:39 / 回复

kinglone

id 为 153 文章链接挂了,看来这篇文章写之后才美化的url

2015年9月23日 09:37 / 回复

蒟蒻

开头说的位运算的[]Blog](http://www.matrix67.com/blog/article.asp?id=153)404了

2015年10月25日 18:14 / 回复

xyz軟體

谢谢分享！

2016年4月20日 11:18 / 回复

Ivable

来看看，顺便看看评论

2016年5月17日 15:33 / 回复

Rothchild

是否能具体解释一下树状数组中x and（ -x ）

2016年8月10日 23:44 / 回复

dzx123

ORZ

2016年10月25日 07:26 / 回复

123123

常见位运算变换操作没注意优先级吧，我用C语言打，<>的优先级比减号优先级低

2017年4月11日 17:19 / 回复

GerJCS

1314520 xor 19880516 = 20665500
请问这里的异或是怎么得出来的结果呢

2017年8月30日 20:23 / 回复

发表评论

评论

昵称*

邮箱*

网站

提交