推导：无符号和补码乘法的位级等价性

根据等式(2.6)，我们有 $x'=x+x_{w-1}2^w$ 和 $y'=y+y_{w-1}2^w$。计算这些值的乘积模 2^w 得到以下结果：

$$(x'\cdot y')\bmod 2^w = [(x+x_{w-1}2^w)\cdot(y+y_{w-1}2^w)]\bmod 2^w$$

$$= [x\cdot y+(x_{w-1}y+y_{w-1}x)2^w+x_{w-1}y_{w-1}2^{2w}]\bmod 2^w$$
$$= (x\cdot y)\bmod 2^w \qquad (2.18)$$

由于模运算符，所有带有权重 2^w 和 2^{2w} 的项都丢掉了。根据等式(2.17)，我们有 $x*^t_w y = U2T_w((x\cdot y)\bmod 2^w)$。对等式两边应用操作 $T2U_w$ 有：

$$T2U_w(x*^t_w y) = T2U_w(U2T_w((x\cdot y)\bmod 2^w)) = (x\cdot y)\bmod 2^w$$

将上述结果与式(2.16)和式(2.18)结合起来得到 $T2U_w(x*^t_w y)=(x'\cdot y')\bmod 2^w = x'*^u_w y'$。然后对这个等式的两边应用 $U2B_w$，得到

$$U2B_w(T2U_w(x*^t_w y)) = T2B_w(x*^t_w y) = U2B_w(x'*^u_w y')$$ ■

**2.75 假设我们想要计算 $x\cdot y$ 的完整的 $2w$ 位表示，其中，x 和 y 都是无符号数，并且运行在数据类型 unsigned 是 w 位的机器上。乘积的低 w 位能够用表达式 x*y 计算，所以，我们只需要一个具有下列原型的函数：

```
unsigned unsigned_high_prod(unsigned x, unsigned y);
```

这个函数计算无符号变量 $x\cdot y$ 的高 w 位。

我们使用一个具有下面原型的库函数：

```
int signed_high_prod(int x, int y);
```

它计算在 x 和 y 采用补码形式的情况下，$x\cdot y$ 的高 w 位。编写代码调用这个过程，以实现用无符号数为参数的函数。验证你的解答的正确性。

提示：看看等式(2.18)的推导中，有符号乘积 $x\cdot y$ 和无符号乘积 $x'\cdot y'$ 之间的关系。

由式(2.18)可知补码乘法换成位相同的无符号数乘法结果是

$$x\cdot y+(x_{w-1}y+y_{w-1}x)\cdot 2^w+x_{w-1}y_{w-1}\cdot 2^{2w}$$

第2w位 $x_{w-1}y_{w-1}$ 在无符号数乘法中亦不保留，故只需要 $x_{w-1}y+y_{w-1}x$ 作为第2w-1到第w位中的最低位，故

```
unsigned unsigned_high_prod(unsigned x, unsigned y){
    size_t w = sizeof(unsigned);
    int a = x >> (w-1) & 1;
    int b = y >> (w-1) & 1;
    return signed_high_prod(x,y) + a&y + b&x;
}
```

**2.77 假设我们有一个任务：生成一段代码，将整数变量 x 乘以不同的常数因子 K。为了提高效率，我们想只使用+、-和<<运算。对于下列 K 的值，写出执行乘法运算的 C 表达式，每个表达式中最多使用 3 个运算。

A. K=17　16+1

B. K=-7　1-8

C. K=60　64-4

D. K=-112　16-128

A、 x = (x << 4) + x ;

B、 x = x - (x << 3) ;

C、 x = (x << 6) - (x << 2) ;

D、 x = (x << 4) - (x << 7) ;

**2.79 写出函数 mul3div4 的代码，对于整数参数 x，计算 3*x/4，但是要遵循位级整数编码规则。你的代码计算 3*x 也会产生溢出。

先计算3x，右移两位时，如果是负数，会产生不准确的结果、
本来该向0取整，但会向下取整，除非整除。

-8/4 = -2　-7/4 = -6/4 = -5/4 = -1，

但得到的 1001、1010、1011 右移 2位都是 -2、

故增加一个偏移量、负数加 n-1（除以n时）。

```
int mul3div4(int x) {
    x = (x<<1) + x ;
    int sign_bit = x >> 31 & 1 ;
    sign_bit && (x = x+3) ; // C的early termination特性，
    return x >> 2 ;
}
```

*2.82 我们在一个 int 类型值为 32 位的机器上运行程序。这些值以补码形式表示，而且它们都是算术右移的。unsigned 类型的值也是 32 位的。

我们产生随机数 x 和 y，并且把它们转换成无符号数，显示如下：

```
/* Create some arbitrary values */
int x = random();
int y = random();
/* Convert to unsigned */
unsigned ux = (unsigned) x;
unsigned uy = (unsigned) y;
```

对于下列每个 C 表达式，你要指出表达式是否总是为 1。如果它总是为 1，那么请描述其中的数学原理。否则，列举出一个使它为 0 的参数示例。

A. (x<y)==(-x>-y)

B. ((x+y)<<4)+y-x==17*y+15*x

C. ~x+~y+1==~(x+y)

D. (ux-uy)==-(unsigned)(y-x)

E. ((x>>2)<<2)<=x

A. x = TMin，y = -1时，-x = TMin，仍然小于 -y = 1.

B、左右同时减y加x，是等价的

C、两边同时加1得 -x-y == -(x+y) 总是等价的

D、两边同时取相反数，有 uy-ux == unsigned(y-x) 由于减法的位级运算模式相同，仍总是等价、

E、舍弃掉低2位，由于补码同号时值的加减同无符号数的字面增大减小一样，故总是等价的、