2.7.1 算术运算

加、减、乘、除、求余

add 加

sub 减 subtract

mul 乘 multiply

div 除 divide、求余

shl是一个汇编指令，作用是逻辑左移指令，将目的操作数顺序左移1位或CL寄存器中指定的位数。左移一位时，操作数的最高位移入进位标志位CF，最低位补零。
SHR指令将目的操作数顺序右移1位或CL寄存器指定的位数。逻辑右移1位时，目的操作数的最低位移到进位标志位CF,最高位补零。

rol——循环左移指令：ROL DEST，COUNT
指令功能：把目的地址中的数据循环左移COUNT次，每次从最高位（最左）移出的数据位都补充到最低位（最右），最后从最高位（最左）移出的数据位保存到CF标志位。
标志位影响：CF标志用于保存最后从最高位移出的数据位。如果COUNT=1，OF标志有意义，如果移位前后数据的符号位发生了变化，OF=1；如果符号位没有发生变化，OF=0。如果COUNT>1，OF标志不确定（没有意义）。
ror 是循环右移指令，被移出的位，补回到最左端。
ror al,cl 就是将al的内容，向右循环移位cl指定的位数。如cl=3,就表示移位3次。

2.7.2 逻辑运算

```
 &     |      ^      ~      <<     >>
and   or    xor    not    shl    shr

_rotl()    _rotr()
rol          ror
```

设 AL=1011 0110
rol al, 1; 表示把 AL 循环左移 1 位 注意这里al原先的最高位补到最后一位
al = _rotl(al, 1); 表示把 AL 循环左移 1 位
结果 AL=0110 1101

以下代码用<< | >>实现了循环左移运算：

```
unsigned long rol(unsigned long x, int n)
{
   return x << n | x >> (sizeof(x)*8-n);
}
```

循环左移与左移的区别在于左移造成位的丢失，而循环左移只改变位的位置，而不会消失，且数据可以循环→连续8次循环左移之后会恢复初始状态→密码学中经常用循环左移与循环右移实现加密与解密。

shl: shift left

shr: shift right
xor: exclusive or
rol: rotate left 循环左移
ror: rotate right 循环右移

```
mov ah, 9Dh; AH=1001 1101
rol ah, 1   ; AH=0011 1011
mov ah, 9Dh; AH=1001 1101
mov cl, 2
rol ah, cl  ; AH=0111 0110
```

位运算的作用：

① 与运算可以使某些位变 0；

设 a 是一个 8 位数，要使该数的最低位与最高位都变 0，其它位不变：

```
?011 011?
0111 1110  and)
-----------
0011 0110
```

② 或运算可以使某些位变 1；

```
?011 011?
1000 0001 or)
-----------
1011 0111
```

③ 异或运算可以使某些位反转；

```
0011 0111
1000 0001  xor)
-----------
1011 0110
```

运用 rol 指令把 16 位整数转化成 16 进制格式输出：

http://10.71.45.100/bhh/v2h.asm

运用 rol 指令把 32 位整数转化成 16 进制格式输出：

http://10.71.45.100/bhh/v2h32.asm