【加减乘除大家庭】

【inc】

加1指令 INC(INCrement by 1)

格式: INC OPRD

----

功能: OPRD<--OPRD+1

这条指令执行结果影响AF、OF、PF、SF、ZF标志位,但不影响CF标志位.

【test】

格式: TEST OPRD1,OPRD2

----

功能: 其中OPRD1、OPRD2的含义同AND指令一样,也是对两个操作数进行按位的'与'运算, ---- 唯一不同之处是不将'与'的结果送目的操作数,即本指令对两个操作数 的内容均不进行修改,仅是在逻辑与操作后,对标志位重新置位.

test A,B=

{

ZF = A|B

}

1、非常普遍的用法是用来测试一方寄存器是否为空:

test ecx, ecx

jz somewhere

如果ecx为零,设置ZF零标志为1,Jz跳转

【cmp】

以CMP DX,CX为例,对标志位的影响如下:

(1) (DX)=(CX)时,则ZF=1;

(2) 两相无符号数比较:

若(DX)>=(CX)时,则CF=0,即无借位.

若(DX)<(CX)时,则CF=1.

(3) 两个带符号数比较

对带符号数判断大小可通过溢出标志OF及符号标志SF共同判断.

当 OF=0,即无溢出时,若SF=0,则(DX)>(CX)

若SF=1,则(DX)<(CX)

当 OF=1,即有溢出时,若SF=1,则(DX)>(CX)

若SF=0,则(DX)<(CX)

IMUL(Integer MULtiply)

格式: IMUL OPRD

----

功能: 完成两个带符号数的相乘

【neg】

取补指令 NEG(NEGate)

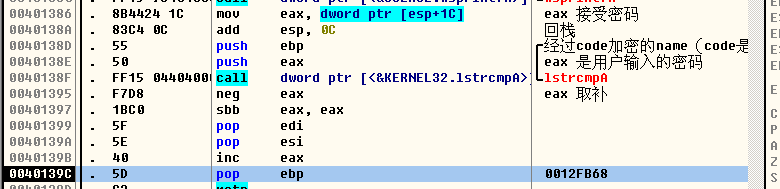
格式: NEG OPRD

----

功能: 对操作数OPRD进行取补操作,然后将结果送回OPRD.取补操作也叫作求补操作,就是

---- 求一个数的相反数的补码.

案例1：neg+sbb的配合



neg:eax = eax取补;错位cf = 0

sbb:eax = eax - eax -cf

整体作用：eax只有是0的时候，neg+sbb 的结果才是0，否则都是FFFFFF

常用以下代码的反汇编

if(function() == 0)

{

return true;

}

else

{

return false;

}

【sbb】

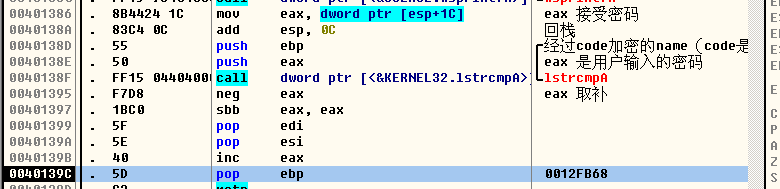
带借位减去指令 SBB(SuBtraction with Borrow)

格式: SBB OPRD1,OPRD2

----

功能: 是进行两个操作数的相减再减去CF进位标志位,即从OPRD1<--OPRD1-OPRD2-CF,其结 ---- 果放在OPDR1中.指令的类型及对标志 位的影响与ADD指令相同,注意立即数不能用 于目的操作数, 两个存储器操作数之间不能直接相减.操作数可为8位或16位的无符 号数或带符号数

案例1：neg+sbb的配合



neg:eax = eax取补;错位cf = 0

sbb:eax = eax - eax -cf

整体作用：eax只有是0的时候，neg+sbb 的结果才是0，否则都是FFFFFF

常用以下代码的反汇编

if(function() == 0)

{

return true;

}

else

{

return false;

}

【跳转家族】

1. 条件跳转：

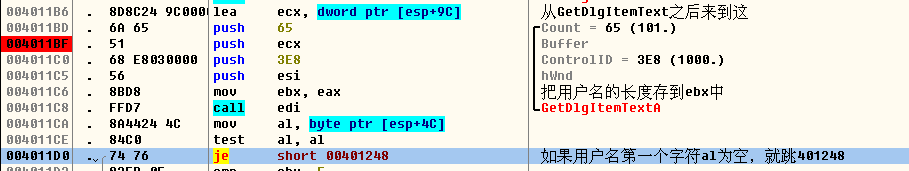
je/jz

je A=

{  
 if ZF==1 就跳到A

}

1. 指令JE与JZ等价,如果ZF为1，就跳转

2. JE,JZ均为一条指令的两种助记符表示方法  


JL/JNGE

格式: JL/JNGE标号

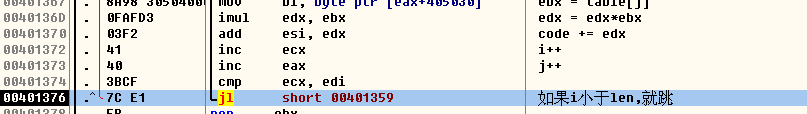
----

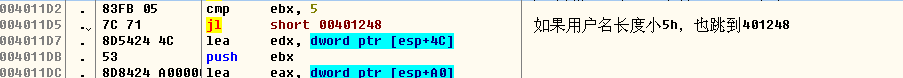
功能: 小于/不大于等于时转移

如

cmp ebx,00000005 如果ebx-5<0,CF = 1,ebx<5,就跳

jl 000430203 那么就跳





JLE/JNG

功能: 小于等于/不大于时转移

