

**汇编语言与逆向技术课程实验报告**

**实验二：dec2hex**

****

学 院 网络空间安全学院

专 业 信息安全

学 号 2211044

姓 名 陆皓喆

班 级 信息安全

**一、实验目的**

**1、**熟悉汇编语言的数据传送、寻址和算术运算；

**2、**熟悉汇编语言过程的定义和使用；

**3、**熟悉十进制和十六进制的数制转换。

**二、实验内容**

编写汇编程序dec2hex.asm，编译成dec2hex.exe。dec2hex.exe的功能是将Windows命令行输入的十进制无符号整数，转换成对应的十六进制整数，输出在Windows命令行中，如图1所示。

输入的十进制无符号整数的范围是**0到4294967295（232-1）。**

输出对应的十六进制整数，对应的范围是**00000000h到FFFFFFFFh。**

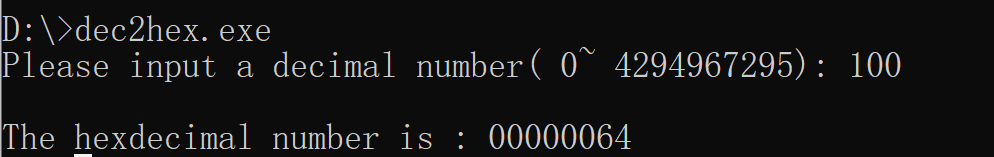


图1. dec2hex.exe将十进制100转换成十六进制00000064

**2.1** 使用StdIn函数获得用户输入的十进制整数。StdIn函数的定义在\masm32\include\masm32.inc，库文件是\masm32\lib\masm32.lib。StdIn函数的定义“StdIn PROTO :DWORD,:DWORD”，有两个参数，第一个是内存存储空间的起始地址，第二个是内存存储空间的大小。函数的例子：

.data

buf BYTE 20 DUP(0)

.code

invoke StdIn, addr buf, 20

invoke StdOut, addr buf

**2.2** 用户输入的十进制数对应的ASCII编码字符串存储在内存中，编写过程dec2dw，将ASCII字符串转换成DWORD数据。例如，将字符串“100”转换成DWORD数据00000064h。

**2.3** 编写过程dw2hex，将DWORD数据转换成十六进制数的ASCII字符串。例如，将DWORD数据00000064h转换成ASCII字符串“00000064”。

**2.4** 使用StdOut函数在Windows命令函中输出十六进制整数的ASCII字符串。StdOut函数的定义在\masm32\include\masm32.inc，库文件是\masm32\lib\masm32.lib。StdOut函数的定义“StdOut PROTO :DWORD”，只有一个参数，是内存存储空间的起始地址。函数使用的例子同StdIn函数的例子。

**2.5** 使用ml将dec2hex.asm文件汇编到dec2hex.obj目标文件，编译命令：“\masm32\bin\ml/c/coff dec2hex.asm”。

**2.6** 使用link将目标文件dec2hex.obj链接成dec2hex.exe可执行文件，链接命令：“\masm32\bin\link /SUBSYSTEM: CONSOLE dec2hex.obj”。

**三、实验过程**

**3.1代码实现部分**

.386

.model flat, stdcall

option casemap :none

include \masm32\include\windows.inc

include \masm32\include\kernel32.inc

include \masm32\include\masm32.inc

includelib \masm32\lib\kernel32.lib

includelib \masm32\lib\masm32.lib

.data;定义数据段

decstr BYTE 20 DUP(0),0

str\_f BYTE "Please input a decimal number:"

decnum DWORD 0

const10 DWORD 10

hexstr BYTE 8 DUP(30h);规定8个位置的起始位都是48

.code;定义代码段

dec2dw PROC

mov esi,0;用于计数

mov edx,0

mov eax,0

mov ebx,0;这四个寄存器全部赋值为0

L1:

mov dl,[decstr+esi];从第一位开始一次取一个字节的字符串

sub dl,30h;转换成数值，存在dl里

mov ebx,eax;将eax的值赋值给ebx

shl eax,1;将eax逻辑左移一位，即乘以2

shl ebx,3;将ebx逻辑左移三位，即乘以8

add eax,ebx;把ebx加到eax中，就能得到原始值的10倍

add eax,edx;加上edx中的那一位

inc esi;esi表示计数器，计算出目前正在进行操作的位数

mov bl,[decstr+esi];取下一位的值传给bl

cmp bl,0h;将bl的值与0进行比较

jnz L1;如果bl的值不为0的话就继续执行L1语句，如果为0就跳出

mov decnum,eax;将eax的值传给decnum即可

ret

dec2dw ENDP

dw2hex PROC

mov edx,7h;通过[hexstr+edx]来把十六进制字符串依次加入

mov ecx,0h;记录decnum中要移动的是第几位，从最低位开始索引

mov ebx,0h;记录要把每一位16进制数移动几位

L2:

mov eax,decnum;将decnum的值传给eax寄存器

mov ebx,ecx;将ecx的值传给ebx

shl ebx,2;相当于乘4，因为每个16进制数是4位；相当于是右移了两位

L3:

cmp ebx,0;将ebx与0进行比较，若等于0则直接跳转L4

je L4

shr eax,1;将eax左移一位

dec ebx;将ebx做减一处理，相当于循环，直到ebx为0为止

jmp L3

L4:

and eax,0fh;按位与，0f中只有后四位是1，才能使eax中存的16进制数最后四位留下来，这一位正好是我们要取得数字

cmp eax,9h;将eax的内容与9做比较，判断是数字还是字母，是数字的话就加30h，是字母的话就加57h

jle L5

add eax,57h

jmp L6

L5:

add eax,30h

L6:

mov [hexstr+edx],al;将al传给对应的数字

inc ecx;将ecx做增加处理

cmp edx,0h;将edx与0作比较，若等于0则结束，若不等于0则edx减一，重新进行L2的运算

je L7

dec edx

jmp L2

L7:

invoke StdOut,addr hexstr

ret

dw2hex ENDP

main PROC

invoke StdOut,addr str\_f;调用StdOut，进行输出

invoke StdIn,addr decstr,20;调用StdIn，进行输入

CALL dec2dw;调用dec2dw

CALL dw2hex;调用dw2hex

invoke ExitProcess,0

main ENDP

END main

**3.2基本思路**

① dec2dw这个过程的思路就是遍历字符串的每一位（由高到低），转成数值的方法是减去48，对每位数值进行：数值+eax\*10的操作。\*10操作通过位运算实现（左移3位+左移1位，相当于\*8+\*2=\*10）。最后将eax寄存器里的值赋给decnum，得到十六进制数即可。

② dw2hex这个过程的思路是挨个读取decnum的每一位数字，用到的方法是先把要进行转换的这位数字移到最后四位（每位16进制数是4个二进制位），通过位运算实现（移动4i次，i是从低到高第几位），然后和0fh进行与运算，这样就得到了最后四个二进制位，再判断是否是字母后，赋值给hexstr的对应位。

**3.3编译**

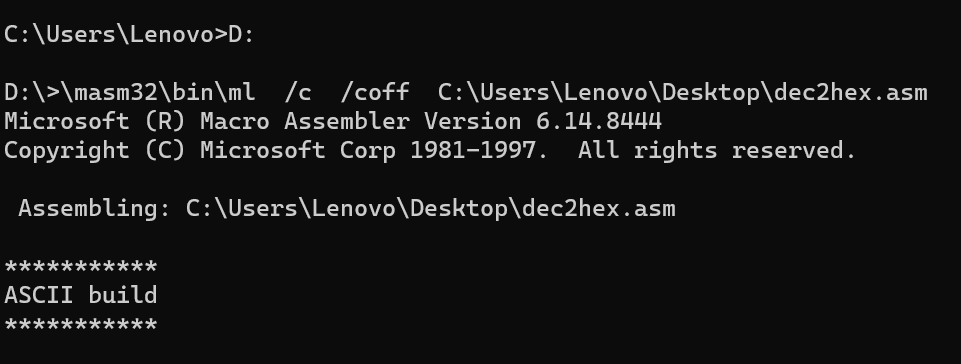
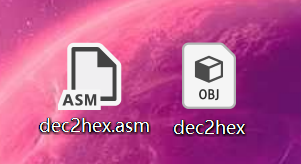
命令行中跳转到.asm文件所在文件夹后，输入：\masm32\bin\ml/c/coff dec2hex.asm，得到.obj文件。

**3.4链接**

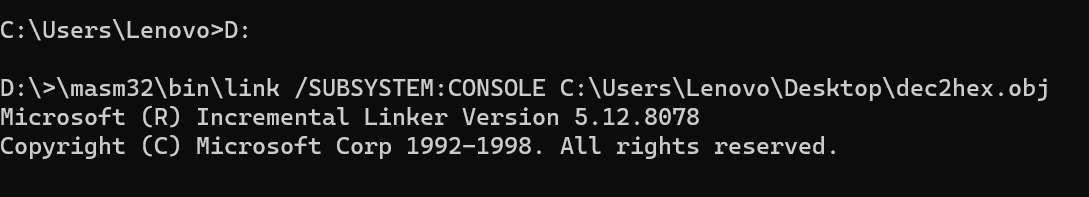
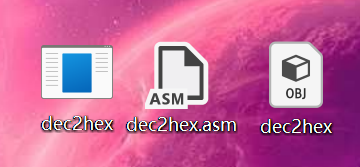
继续输入：\masm32\bin\link /SUBSYSTEM:CONSOLE dec2hex.obj，得到了可执行的.exe文件。

**3.5实验截图**

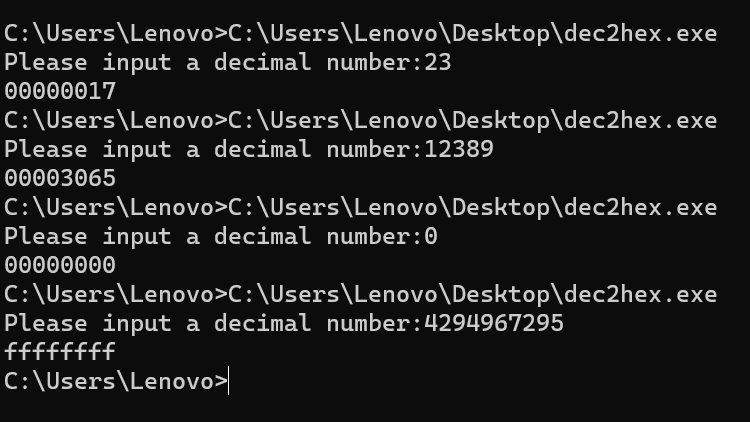
3.5.1编译部分



3.5.2链接部分



3.5.3结果部分



**四、实验结论及心得体会**

通过本次实验，我学会了一些基本的汇编语句操作，如赋值，加法，减法，判断、循环、有条件跳转和无条件跳转，以及如何使用位运算去简化乘法运算。本次实验让我在一定程度上有了从C++那种高级语言向汇编语言过渡的适应，希望今后我能够更加理解、明白、运用汇编语言。