

南开大学

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

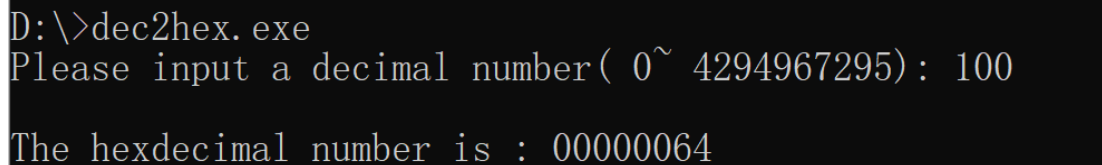
- 1、熟悉汇编语言的数据传送、寻址和算术运算；
- 2、熟悉汇编语言过程的定义和使用；
- 3、熟悉十进制和十六进制的数制转换。

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 ($2^{32}-1$)**。

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



```
D:\>dec2hex.exe
Please input a decimal number( 0~ 4294967295): 100
The hexadecimal number is : 00000064
```

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

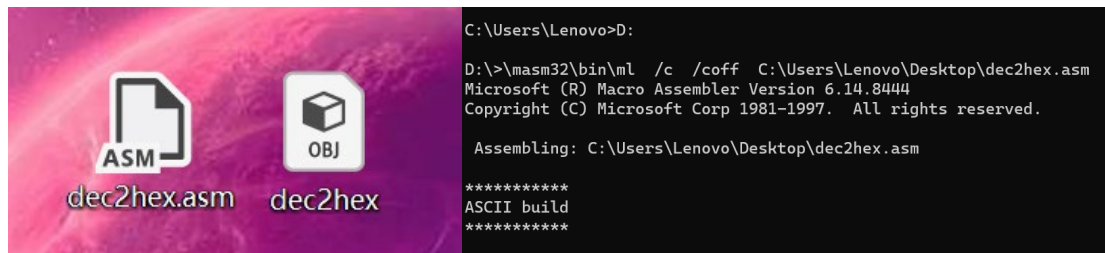
命令行中跳转到.asm 文件所在文件夹后，输入：\masm32\bin\ml/c/coff dec2hex.asm，得到.obj 文件。

3.4 链接

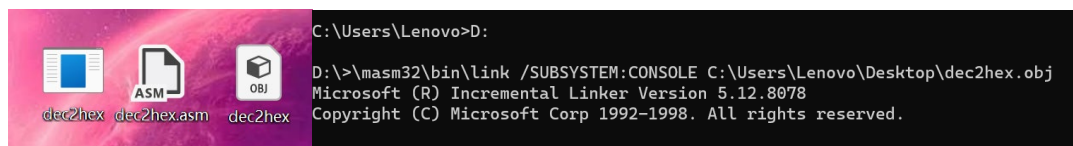
继续输入：\masm32\bin\link /SUBSYSTEM:CONSOLE dec2hex.obj，得到了可执行的.exe 文件。

3.5 实验截图

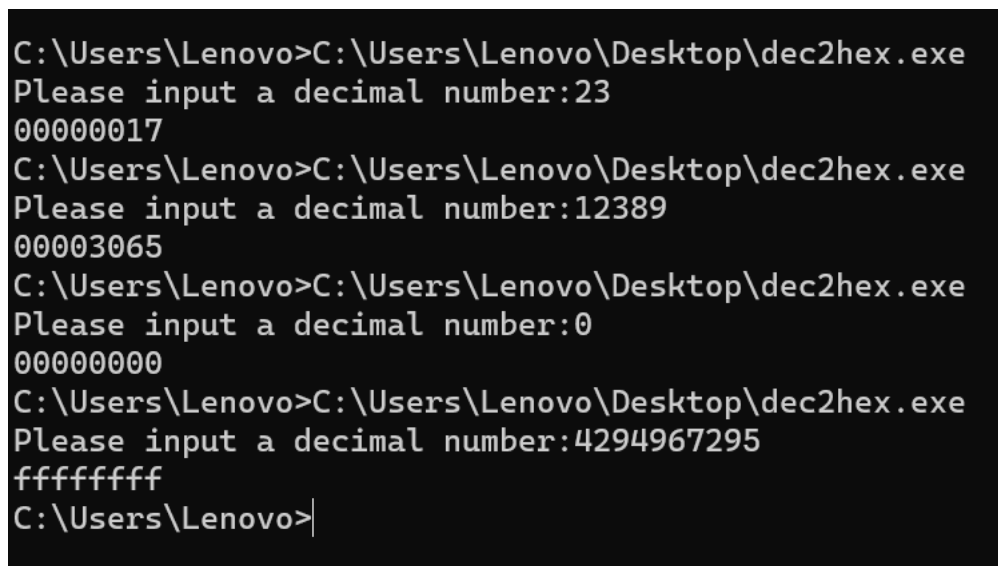
3.5.1 编译部分



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。