|  |  |
| --- | --- |
| 001 | jsjlogo |

**合肥工业大学**

**计算机与信息学院**

**计算机系2018级**

**汇编语言实验报告**

**姓 名： 孙淼**

**专业年级： 计算机科学与技术18-2班**

**学 号： 2018211958**

一）实验目的

1、熟悉DEBUG程序中的命令，学会在DEBUG下调试运行汇编语言源程序。

2、掌握8086/8088的寻址方式及多字节数据的处理方法。

（二）实验设备

硬件环境：IBM-PC及其兼容机

软件环境：操作系统DOS3.0版本以上

调试程序DEBUG.COM

（三）实验内容和要求

实验内容：

1、利用DEBUG程序中的“E”命令，将两个多字节数“12345678H”和“FEDCBA98H”分别送入起始地址为DS:0200H和DS:0204H两个单元中。

2、分别用直接寻址方式和寄存器间接寻址方式编写程序段，实现将DS:0200H单元和DS:0204H单元中的数据相加，并将运算结果存放在DS:0208H单元中。

实验要求：

本次实验的内容均在DEBUG下完成，实现数据的装入、修改、显示；汇编语言程序段的编辑、汇编和反汇编；程序的运行和结果检查。

（四）预习要求

1、复习DOS启动方式和DOS有关命令的使用；

2、阅读本书后的附录，熟悉掌握DEBUG调试程序中的A、D、E、G、Q、R、T、U命令的书写格式及功能；

3、复习寻址方式和相关的数据传送指令及算术运算指令。

（五）实验步骤

1、启动DOS操作系统

2、运行DEBUG.COM程序（若当前盘为C）

C:＞DEBUG↙

– ;(“–”为DEBUG提示符,仅当屏幕出现该提示符后，才可输入DEBUG命令)

3、用“A”命令编辑和汇编源程序

–A ↙

186E：0100 MOV AX，[0200]↙

186E：0103 MOV BX，[0202]↙

186E：0107 ADD AX，[0204]↙

186E：010B ADC BX，[0206]↙

186E：010F MOV [0208]，AX↙

186E：0112 MOV [020A]，BX↙

186E：0116 ↙

4、用“U”命令反汇编验证源程序

–U CS:0100↙

186E：0100 A10002 MOV AX，[0200]

186E：0103 8B1E0202 MOV BX，[0202]

186E：0107 03060402 ADD AX，[0204]

186E：010B 131E0602 ADC BX，[0206]

186E：010F A30802 MOV [0208]，AX

186E：0112 891E0A02 MOV [020A]，BX

186E：0116

–

**注意：**

* 检查源程序是否有错误，若有则返回到第3步用“A 地址”命令进行修改,直到程序无错误。
* DEBUG约定在其命令或源程序中所涉及到的数据均被看作十六进制数，其后不用“H”说明。
* 在DEBUG下，源程序中不能使用“标号”、“变量”和一些伪指令。
* 在DEBUG下，大小写不敏感。

5、程序未执行前，用“R”命令观察，相关寄存器的内容。

–R↙

如：（AX）= ，（BX）= ，（SI）= ，

（DI）= ，（CS）= ，（IP）= 。

6、在程序未执行前，用“D”命令观察DS:0200H，DS:0204H、DS:0208H单元的内容

–D 200 20B ↙

7、用“G”命令运行源程序

–G =100 0116 ↙

8、再用“D”命令观察DS:0200H～DS:020BH字节单元中的内容

–D 200 20B↙

9、用“T”单步操作命令对源程序单步执行，观察AX、BX、CS、IP寄存器内容的变化，并与预计的结果相比较。

**注意：**

* D命令显示内存数据，注意观察多字节数据在内存中的存放方式。
* 指令执行的时候，IP寄存器随着变化，注意观察。

（六）实验结果分析

1、程序运行结果：（AX）= ，（BX）=

DS:0208H四字节单元的内容：

2、试用寄存器间接寻址方式编写程序段，完成上述程序段的功能。（参考下列程序流程图）

|  |
| --- |
| SI←200H |

|  |
| --- |
| DI←204H |

|  |
| --- |
| BX←208H |

|  |
| --- |
| AX← [SI]  DX←[SI+2] |

|  |
| --- |
| AX←AX+[DI]  DX←DX+[DI+2 ] |

|  |
| --- |
| [BX]←(AX)  [BX+2]←(DX) |

开始

结束

（七）、附加题目：

1. 从DS:0000H开始的5个16位带符号数分别记为x,y,z,v,w，用E命令初始化为：540,1,-1,1080,0, 试求w=(v-(x\*y+z-540))/x
2. 完成BX中1的个数的统计。
3. 用串操作指令将DS：0000开始20个字节初始化为：0CCH。并将这20个字节的内容复制到其后的20个字节里。

**实验一 Debug程序的使用**

1. **实验目的**

1、熟悉DEBUG程序中的命令，学会在DEBUG下调试运行汇编语言源程序。

2、掌握8086/8088的寻址方式及多字节数据的处理方法。

1. 实验内容

1、利用DEBUG程序中的“E”命令，将两个多字节数“12345678H”和“FEDCBA98H”分别送入起始地址为DS:0200H和DS:0204H两个单元中。

2、分别用直接寻址方式和寄存器间接寻址方式编写程序段，实现将DS:0200H单元和DS:0204H单元中的数据相加，并将运算结果存放在DS:0208H单元中。

要求：

本次实验的内容均在DEBUG下完成，实现数据的装入、修改、显示；汇编语言程序段的编辑、汇编和反汇编；程序的运行和结果检查。

1. **实验过程和程序**

1、利用DEBUG程序中的“E”命令，将两个多字节数“12345678H”和“FEDCBA98H”分别送入起始地址为DS:0200H和DS:0204H两个单元中。

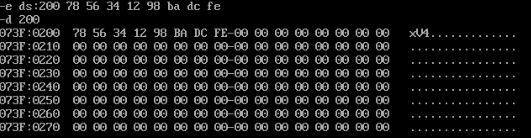
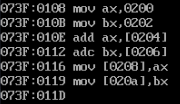
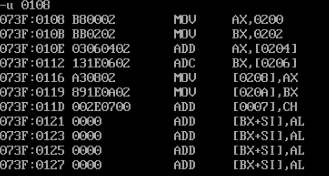
2、分别用直接寻址方式和寄存器间接寻址方式编写程序段，实现将DS:0200H单元和DS:0204H单元中的数据相加，并将运算结果存放在DS:0208H单元中。

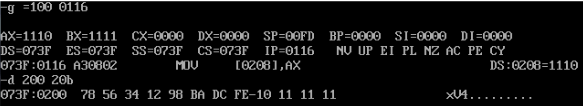
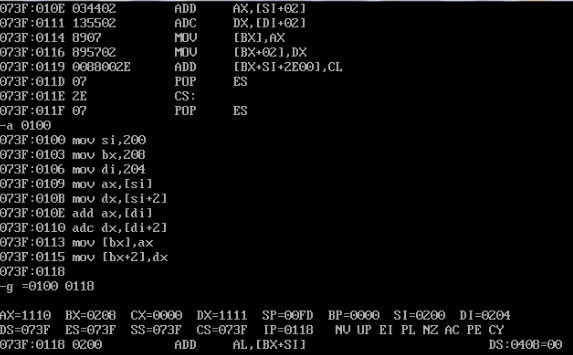
3、从DS:0000H开始的5个16位带符号数分别记为x,y,z,v,w，用E命令初始化为：540,1,-1,1080,0, 试求w=(v-(x\*y+z-540))/x

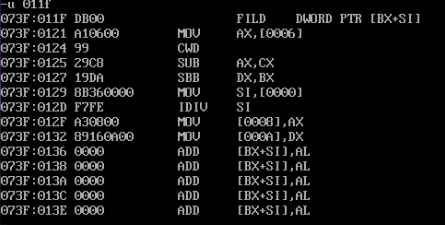
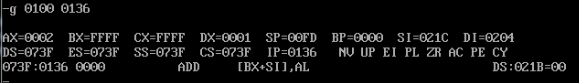
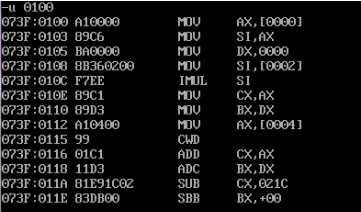
4、完成BX中1的个数的统计。

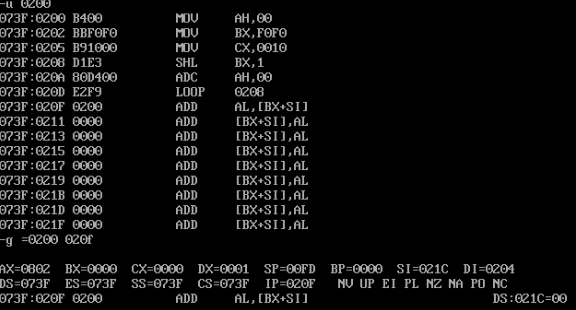
5、用串操作指令将DS：0000开始20个字节初始化为：0CCH。并将这20个字节的内容复制到其后的20个字节里。

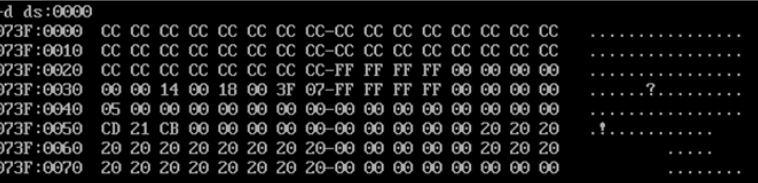
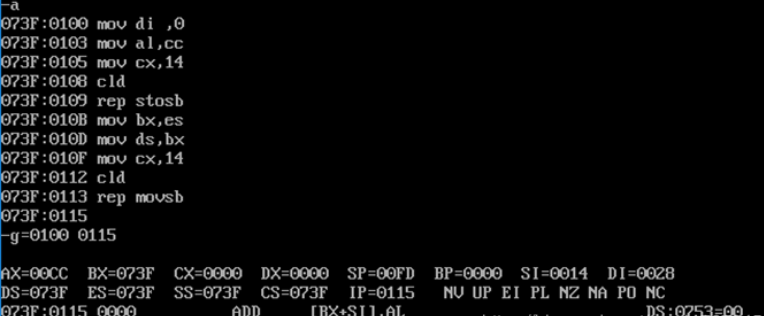
1. **实验结果（包括必要的截图）**

****用直接寻址方式编写的程序段

****用寄存器间接寻址方式编写的程序段

****从DS:0000H开始的5个16位带符号数分别记为x,y,z,v,w，用E命令初始化为：540,1,-1,1080,0, 试求w=(v-(x\*y+z-540))/

****完成BX中1的个数的统计。

用串操作指令将DS：0000开始20个字节初始化为：0CCH。并将这20个字节的内容复制到其后的20个字节里。

1. **实验体会**

本次实验是第一次真正操作debug来编程，不能复制粘贴已经给的源码真的很伤啊。。。注意到存储器寻址方式中，源操作数给出的为偏移地址。而物理地址=16\*（DS）加上偏移地址。

**实验二 汇编语言程序设计(顺序、多分支、循环)**

1. **实验目的**

1、掌握顺序和循环程序结构和设计方法；

2、熟悉在PC机上建立、汇编、连接、调试和运行8086/8088汇编语言程序的过程。

1. **实验内容**

1、X、Y、Z、V均为字变量，在X、Y、Z、V字单元中存放是16位带符号数。试编写汇编语言程序完成以下功能：

①计算表达式值（V–（X\*Y+Z-720））/X；

②将上述表达式运算结果整数放在SUM1单元，余数放在SUM2单元。

2、使用地址表实现如下功能：根据输入的数字1－7，分别显示相应的英文星期名，如果输入其他字符，则重新输入。

3、求一个班50名学生成绩的平均值、最大值和最小值,并将结果显示出来。

4、用串操作指令将DS：0000开始20个字节初始化为：0CCH。并将这20个字节的内容复制到其后的20个字节里。

1. **实验过程和程序**

计算表达式值（V–（X\*Y+Z-720））/X；将上述表达式运算结果整数放在SUM1单元，余数放在SUM2单元。

dseg segment

 x dw 2

 y dw 360

 z dw 0

 v dw 5

 sum1 dw ?

 sum2 dw ?

 dseg ends

 cseg segment

      assume cs:cseg,ds:dseg

 start:

       mov ax,dseg

       mov ds,ax

       mov ax,x

       mov bx,y

       mul bx

       mov bx,z

       add ax,bx

       sub ax,2d0h

       mov bx,v

       sub bx,ax

mov ax,bx

       cbw

       mov bx,x

       div bx

mov sum1,ax

       mov sum2,dx

       mov ah,4ch

       int 21h

 cseg ends

      end    start

使用地址表实现如下功能：根据输入的数字1－7，分别显示相应的英文星期名，如果输入其他字符，则重新输入。

data segment

a dw l1,l2,l3,l4,l5,l6,l7

s1 db 'monday $'

s2 db 'tuesday $'

s3 db 'wednesday $'

s4 db 'thursday$'

s5 db 'friday$'

s6 db 'startday$'

s7 db 'sunday$'

msg db ' num$'

msg1 db 0dh,0ah,' over$'

data ends

code segment

start: mov ax,data

mov ds,ax

start1: mov dx,offset msg

mov ah,9

int 21h

mov ah,1

int 21h

cmp al,'1'

jb start1

cmp al,'8'

ja start1

and ax,000fh

dec ax

shl ax,1

mov bx,ax

jmp a[bx]

l1:lea dx,s1

jmp s

l2:lea dx,s2

jmp s

l3:lea dx,s3

jmp s

l4:lea dx,s4

jmp s

l5:lea dx,s5

jmp s

l6:lea dx,s6

jmp s

l7:lea dx,s7

s:mov ah,9

int 21h

jmp d

start2: mov ah,9

int 21h

mov ah,1

int 21h

jmp far ptr start1

d:mov dx,offset msg1

jmp start2

mov ax,4c00h

int 21h

code ends

end start

求一个班50名学生成绩的平均值、最大值和最小值,并将结果显示出来。

dseg segment

score db 10 dup(80,70,60,90,90)

avg db ?

max db 0dh,0ah,?

min db 0dh,0ah,?

dseg ends

cseg segment

      assume cs:cseg,ds:dseg

start:

       mov ax,dseg

       mov ds,ax

       mov cx,50

       mov ax,0

       mov bh,0

       mov bl,100

       mov si,0

l:    add al,score[si]

       adc ah,0

       inc si

loop l

mov dx,si

       div dl

       mov avg,al

       mov cx,50

       mov si,0

l1:   cmp score[si],bh

       jbe s1

       mov bh,score[si]

s1:   inc si

loop l1

       mov cx,50

       mov si,0

l2:   cmp score[si],bl

       ja s2

       mov bl,score[si]

s2:   inc si

       loop l2

       mov al,bl

       mov ah,0

       mov cl,10

       div cl

       cmp al,0

       jz gw1

       cmp al,10

       jz mf1

       mov ch,ah

       add al,30h

       mov dl,al

       mov ah,2

       int 21h

       add ch,30h

       mov dl,ch

       mov ah,2

       int 21h

       jmp done1

gw1:   mov dl,ah

       add dl,30h

       mov ah,2

       int 21h

       jmp done1

mf1:   mov dl,31h

       mov ah,2

       int 21h

       mov dl,30h

       mov ah,2

       int 21h

       mov dl,30h

       mov ah,2

       int 21h

done1:

      mov dl,0dh

      mov ah,2

      int 21h

      mov dl,0ah

      mov ah,2

      int 21h

      mov al,bh

      mov ah,0

      mov cl,10

      div cl

      cmp al,0

       jz gw

       cmp al,10

       jz mf

       mov ch,ah

       add al,30h

       mov dl,al

       mov ah,2

       int 21h

       add ch,30h

       mov dl,ch

       mov ah,2

       int 21h

       jmp done

gw:   mov dl,ah

      add dl,30h

       mov ah,2

       int 21h

       jmp done

mf:   mov dl,31h

       mov ah,2

       int 21h

       mov dl,30h

       mov ah,2

       int 21h

       mov dl,30h

       mov ah,2

       int 21h

done: mov ah,4ch

       int 21h

cseg ends

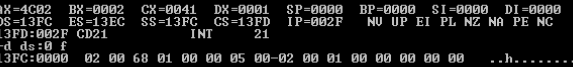
end    start

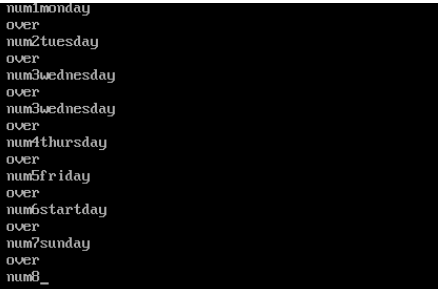
用串操作指令将DS：0000开始20个字节初始化为：0CCH。并将这20个字节的内容复制到其后的20个字节里。

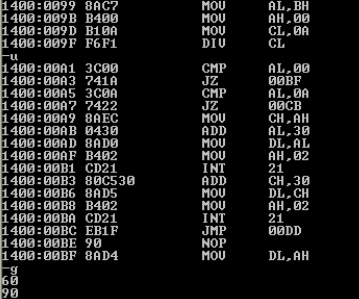
见实验一附加题部分。。。

1. **实验结果（包括必要的截图）**

①计算表达式值（V–（X\*Y+Z-720））/X；

****②将上述表达式运算结果整数放在SUM1单元，余数放在SUM2单元。

使用地址表实现如下功能：根据输入的数字1－7，分别显示相应的英文星期名，如果输入其他字符，则重新输入

求一个班50名学生成绩的平均值、最大值和最小值,并将结果显示出来。

用串操作指令将DS：0000开始20个字节初始化为：0CCH。并将这20个字节的内容复制到其后的20个字节里。

见实验一附加题部分。。。

1. **实验体会**

通过这次试验，掌握了顺序和循环程序结构和设计方法；熟悉在PC机上建立、汇编、连接、调试和运行8086/8088汇编语言程序的过程。并在编程过程中熟悉了算术运算、位操作和串操作等指令。算术运算中，对于不同类型数进行运算时，要进行扩展，还有不要忽视进位；对于循环和控制转移指令，如果有多个循环或转移指令，则前一个循环或转移结束后，一定到无条件转移到结尾处。

**实验三 循环和子程序设计**

1. **实验目的**

1、掌握循环程序和子程序的设计方法；

2、熟悉在PC机上建立、汇编、连接、调试和运行8086/8088汇编语言程序的过程。

1. **实验内容**

1. 从键盘读入一个字符串，以Enter结束，字符串不超过50个字符，并打印该字符串(附加题：查找中间是否有‘asm’子串。如果有，输出‘Yes’；否则，输出‘No’)。

2. BL中的只有一位为1。编写程序测试，并输出提示信息“The X Bit is 1”，要求：地址表。

3. 从键盘读入一个字符串，以Enter结束，字符串不超过50个字符，并打印该字符串(附加题：查找中间是否有‘asm’子串。如果有，输出‘Yes’；否则，输出‘No’)。

4. 编写一个子程序计算z=f(x,y)=x\*y+x-y（x,y,z有符号数内存数）。要求通过堆栈(寄存器、内存)传送所有参数。(要求输入输出实现，且有提示)

5. 实践C🡪Asm的转换过程。

1. **实验过程和程序**

从键盘读入一个字符串，以Enter结束，字符串不超过50个字符，并打印该字符串(附加题：查找中间是否有‘asm’子串。如果有，输出‘Yes’；否则，输出‘No’)。  
data segment

s1 db 51,?,51 dup(&apos;$&apos;)

s3 db 0ah,0dh,&apos;Yes$&apos;

s4 db 0ah,0dh,&apos;No$&apos;

data ends

dseg segment

s2 db &apos;asm&apos;

dseg ends

cseg segment

      assume cs:cseg,ds:data,es:dseg

start:

       mov ax,data

       mov ds,ax

       mov ax,dseg

       mov es,ax

       lea dx,s1

       mov ah,0ah

       int 21h

       mov dl,0dh

       mov ah,2

       int 21h

       mov dl,0ah

       mov ah,2

       int 21h

       lea dx,s1+2

       mov ah,9

       int 21h

       mov cl,s1+1

       cmp cl,3

       jb notfound

       lea si,s1

       lea di,s2

       mov dx,si

        mov bx,di

       mov ah,48

       cld

again:

       mov cx,3

       repz cmpsb

       jz found

       add dx,1

       mov si,dx

       mov di,bx

       dec ah

       jnz again

       jmp notfound

       jmp over

found:

       lea dx,s3

       mov ah,9

       int 21h

       jmp over

notfound:

       lea dx,s4

       mov ah,9

       int 21h

over:

       mov ah,4ch

       int 21h

cseg ends

      end    start

BL中的只有一位为1。编写程序测试，并输出提示信息“The X Bit is 1”，要求：地址表。

dseg segment

a db 2

msg0 db 0dh, 0ah,'The 0 Bit is 1','$'

msg1 db 0dh, 0ah,'The 1 Bit is 1','$'

msg2 db 0dh, 0ah,'The 2 Bit is 1','$'

msg3 db 0dh, 0ah,'The 3 Bit is 1','$'

msg4 db 0dh, 0ah,'The 4 Bit is 1','$'

msg5 db 0dh, 0ah,'The 5 Bit is 1','$'

msg6 db 0dh, 0ah,'The 6 Bit is 1','$'

msg7 db 0dh, 0ah,'The 7 Bit is 1','$'

addrtb1 dw  func0,func1,func2,func3,func4,func5,func6,func7

dseg ends

cseg segment

      assume cs:cseg,ds:dseg

start:

       mov ax,dseg

       mov ds,ax

       mov cx,8

       mov si,0

       mov bl,a

       cmp bl,0

       jz exit

l:     shr bl,1

       jc l1

       inc si

       loop l

jmp exit

l1:    mov ax,si

       and ax,0fh

       shl ax,1

       mov bx,ax

       jmp addrtb1[bx]

func0:lea dx,msg0                    jmp output

func1:lea dx,msg1                    jmp output

func2:lea dx,msg2                    jmp output

func3:lea dx,msg3                    jmp output

func4:lea dx,msg4                    jmp output

func5:lea dx,msg5                    jmp output

func6:lea dx,msg6                    jmp output

func7:lea dx,msg7                    jmp output

output:mov ah,9

        int 21h

exit: mov ah,4ch

       int 21h

cseg ends

      end    start

dseg segment

a db 8

msg0 db 0dh, 0ah,'The 0 Bit is 1','$'

msg1 db 0dh, 0ah,'The 1 Bit is 1','$'

msg2 db 0dh, 0ah,'The 2 Bit is 1','$'

msg3 db 0dh, 0ah,'The 3 Bit is 1','$'

msg4 db 0dh, 0ah,'The 4 Bit is 1','$'

msg5 db 0dh, 0ah,'The 5 Bit is 1','$'

msg6 db 0dh, 0ah,'The 6 Bit is 1','$'

msg7 db 0dh, 0ah,'The 7 Bit is 1','$'

dseg ends

cseg segment

      assume cs:cseg,ds:dseg

start:

       mov ax,dseg

       mov ds,ax

       mov cx,8

       mov si,0

       mov bl,a

       cmp bl,0

       jz exit

 l:     shr bl,1

       jc l1

       inc si

       loop l

       jmp exit

l1:    mov ax,si

       and ax,0fh

       mov bx,ax

       shl ax,1

       add bx,ax

       add bx,offset jmptb1

       jmp bx

jmptb1:

        jmp   near ptr func0

        jmp   near ptr func1

        jmp   near ptr func2

        jmp   near ptr func3

jmp   near ptr func4

        jmp   near ptr func5

        jmp   near ptr func6

        jmp   near ptr func7

func0:lea dx,msg0

                    jmp output

func1:lea dx,msg1

                    jmp output

func2:lea dx,msg2

                    jmp output

func3:lea dx,msg3

                    jmp output

func4:lea dx,msg4

                    jmp output

func5:lea dx,msg5

                    jmp output

func6:lea dx,msg6

                    jmp output

func7:lea dx,msg7

                    jmp output

output:mov ah,9

        int 21h

exit: mov ah,4ch

       int 21h

cseg ends

      end    start

编写一个子程序计算z=f(x,y)=x\*y+x-y（x,y,z有符号数内存数）。要求通过堆栈(寄存器、内存)传送所有参数。(要求输入输出实现，且有提示)

dseg segment

x db 2

y db 4

z db ?

dseg ends

cseg segment

      assume cs:cseg,ds:dseg

start:

       mov ax,dseg

       mov ds,ax

       call result

       mov ah,4ch

       int 21h

result proc

        push ax

        push bx

        mov al,x

        imul y

        mov bl,x

        add al,bl

        sub al,y

        mov z,al

        pop bx

        pop ax

        ret

result endp

cseg ends

      end    start

data segment

    x dw 123

    y dw 56

    z dw ?

data ends

code segment

    assume cs:code,ds:data

Start:

mov ax,data

mov ds,ax

sub sp,2

   push x

push y

   call c

   pop z

   mov ah,4ch

   int 21h

c proc

   push bp

mov bp,sp

   push ax

   push bx

  push dx

   mov ax,[bp+6]

   mov bx,[bp+4]

   mul bx

   mov dx,[bp+6]

   sub dx,bx

   add ax,dx

   mov [bp+8],ax

    pop dx

  pop bx

  pop ax

  pop bp

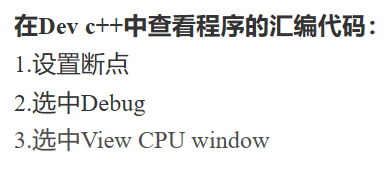
  ret 4

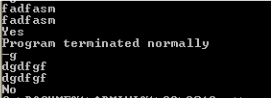
  mov ah,4ch

int 21h

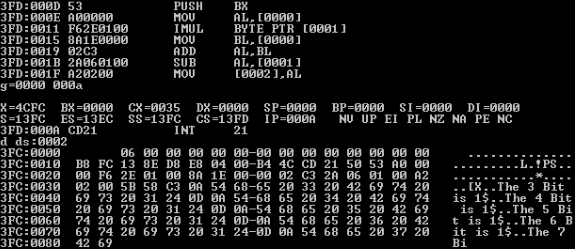
code ends

    end start

实践C🡪Asm的转换过程。  
**实验结果（包括必要的截图）**

从键盘读入一个字符串，以Enter结束，字符串不超过50个字符，并打印该字符串(附加题：查找中间是否有‘asm’子串。如果有，输出‘Yes’；否则，输出‘No’)。

BL中的只有一位为1。编写程序测试，并输出提示信息“The X Bit is 1”，要求：地址表。

编写一个子程序计算z=f(x,y)=x\*y+x-y（x,y,z有符号数内存数）。要求通过堆栈(寄存器、内存)传送所有参数。(要求输入输出实现，且有提示)

实践C🡪Asm的转换过程。



1. **实验体会**

通过本次试验,我发现且程序中，以及在输出时，输入的字符串末尾要注意加结束标志’$’,还有在输出字符串前要先输出个回车换行，否则回车后看不到输出结果。还了解了如何使用地址表和转移表，以及怎样通过堆栈(寄存器、内存)传送所有参数调用子程序。最后实践了C到Asm的转换过程，使我对C和汇编有了更多的认识。

**实验四 宏与中断程序设计**

1. **实验目的**

1、汇编语言宏程序设计；

2、 掌握中断服务子程序的编写。

1. **实验内容**

1、编写一个宏，求三个数的最小数，原型为：MIN3 x,y,z,min

2、挂接1CH中断，正计时60秒后退出。要求屏幕显示0-59的秒数。

3、VC++中编写内联汇编，将字节数组的每个元素高低四位互换。

char dbyte[] = {0x34,0x45,0x56,0x67,0xaf};

1. **实验过程和程序**

编写一个宏，求三个数的最小数，原型为：

MIN3 x,y,z,min

min macro x,y,z

  mov ax,x

  cmp ax,y

  jl next

  mov ax,y

next: cmp ax,z

  jl final

  mov ax,z

  final: nop

  endm

  code segment

  assume cs:code

  start: min 4 -6 3

  mov ax,4c00h

  int 21h

  code ends

  end start

挂接1CH中断，正计时60秒后退出。要求屏幕显示0-59的秒数。

intno equ 1ch

data segment

 pian dw ?

 duan dw ?

 string db 48

 data ends

code segment

 assume cs:code,ds:data

 start:  mov ax,data

 mov ds,ax

begin:

  mov ax,351ch

    int 21h

    mov pian,bx

    mov duan,es

    push ds

    mov dx,offset myproc

    mov ax,seg myproc

    mov ds,ax

    mov ax,251ch

    int 21h

    pop ds

    mov cx,0

    mov bx,0

next: cmp cx,1080;   60\*18

     jnz next

     mov dx,pian

     mov ax,duan

     mov ds,ax

     mov ax,251ch

     int 21h

mov ax,4c00h

   int 21h

myproc proc

     sti

  inc cx

   inc bx

    cmp bx,18

jz print

   jmp over

print:

     mov bl,string

     cmp bl,58

jc p0

mov al,bl

sub al,48

 mov ah,00h

mov dl,10

div dl

mov bl,ah

mov dl,al

add dl,48

mov ah,02h

int 21h

mov dl,bl

add dl,48

mov ah,02h

int 21h

jmp p1

p0:

    mov dl,string

    mov ah,02h

    int 21h

p1:

inc string

     mov ah,02h

    mov dl,0dh

    int 21h

   mov bx,0  ;

mov dl,0dh;

int 21h

over: iret ;

msg:     ;

string db 'one second $'

    myproc endp

code ends

end start

data segment

 pian dw ?

 duan dw ?

 string db 48

 data ends

 code segment

 assume cs:code,ds:data

start:

mov ax,data

mov ds,ax

mov ax,351ch

int 21h

mov pian,bx

mov duan,es

push ds

mov dx,offset myproc

mov ax,seg myproc

mov ds,ax

mov ax,251ch

int 21h

pop ds

begin:

mov cx,0

mov bx,0

next:

cmp cx,1080;   60\*18

     jnz next

     mov dx,pian

     mov ax,duan

     mov ds,ax

     mov ax,251ch

     int 21h

   mov ax,4c00h

   int 21h

  myproc proc

sti

push ax

   push bx

   push cx

   push si

   mov ah,0bh

   int 21h

   sub al,00h

jz ok

   mov ah,07h

   int 21h

   cmp al,61h

   jz crazy

   jmp ok crazy:

   pop si

   pop cx

   pop bx

   pop ax

   mov ah,0ch;

  int 21h

   mov ah,02h

   mov dl,0ah

   int 21h ;

jmp begin

   mov cx,0

mov bx,0

mov string,48;string

  jmp over ok:

   pop si

   pop cx

   pop bx

   pop ax

   inc cx

   inc bx

   cmp bx,18

   jz print

    jmp over print:

    mov bl,string

    cmp bl,58

   jc p0

    mov al,bl

    sub al,48

    mov ah,00h

    mov dl,10

    div dl

    mov bl,ah

    mov dl,al

    add dl,48

    mov ah,02h

    int 21h

mov dl,0dh

   int 21h

 ; jmp begin

  mov cx,0

  mov bx,0

  mov string,48;string

jmp over      ok:

  pop si

  pop cx

  pop bx

  pop ax

  inc cx

  inc bx

  cmp bx,18

  jz print

  jmp over print:

  mov bl,string

  cmp bl,58

  jc p0

  mov al,bl

  sub al,48

  mov ah,00h

   mov dl,10

   div dl

   mov bl,ah

   mov dl,al

   add dl,48

   mov ah,02h

   int 21h

mov dl,bl

   add dl,48

   mov ah,02h

   int 21h

   jmp p1

p0:

   mov dl,string

   mov ah,02h

   int 21h

p1:

  inc string

   mov ah,02h

   mov dl,0dh

   int 21h

   mov bx,0  ;

mov dl,0dh

;int 21h

   over: iret

;msg:

;string db 'one second $'

   myproc endp

code ends

end start

VC++中编写内联汇编，将字节数组的每个元素高低四位互换。

char dbyte[] = {0x34,0x45,0x56,0x67,0xaf};

#include<iostream>

using namespace std;

  char shu[]={0x34, 0x45, 0x56, 0x67, 0xaf };

int main(){

    for (int i = 0; i < 5; i++)

   cout<<(int)shu[i]<<"  ";

cout << endl;

  \_asm{mov esi,0

   mov ecx,5

   again:

   mov dl, shu[esi]

   mov eax,0

   next1:

   ror dl,1

   inc eax

   cmp eax,4

   jne next1

   mov shu[esi],dl

   add esi,1

   loop again

     }

  for ( i = 0; i < 5; i++)cout << (int)shu[i] << " ";

  cout << endl;

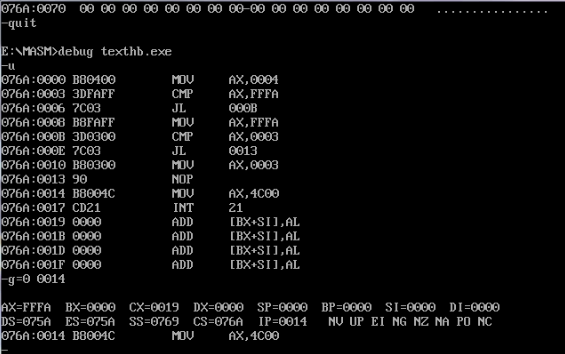
  return 0;

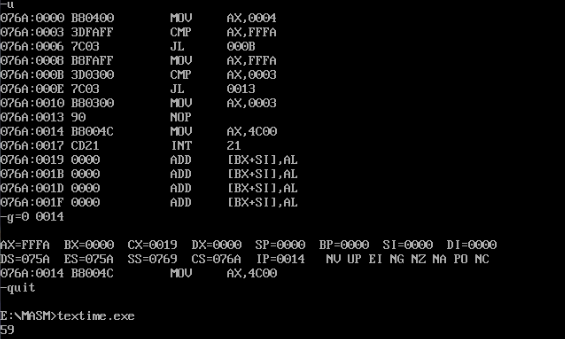
 }

1. **实验结果（包括必要的截图）**

编写一个宏，求三个数的最小数，原型为：

MIN3 x,y,z,min



挂接1CH中断，正计时60秒后退出。要求屏幕显示0-59的秒数。

VC++中编写内联汇编，将字节数组的每个元素高低四位互换。

char dbyte[] = {0x34,0x45,0x56,0x67,0xaf};

1. **实验体会**

通过本次试验，掌握了汇编语言宏程序设计，熟悉了中断服务子程序的基本知识了解了中断服务子程序的编写和汇编语言的编写，也了解到了中断服务子程序与一般子程序在编写和调用返回上的不同之处。