〔习题2.19〕假设DS=2000H，BX=1256H，SI=528FH，位移量TABLE=20A1H，[232F7H]=3280H，[264E5H]

=2450H，试问执行下列段内间接寻址的转移指令后，转移的有效地址是什么？

（1） JMP BX

（2） JMP TABLE[BX]

（3） JMP [BX][SI]

〔解答〕

（1）转移的有效地址EA= BX=1256H

（2）转移的有效地址

EA= [DS:20A1H+1256H]=[232F7]=3280H

（3）转移的有效地址

EA= [DS:1256H+528FH]=264E5H=2450H

（6）已知字符串string包含有32KB内容，将其中的’$’符号替换成空格。

（

6）解答1：不使用串操作指令（更好）

           mov si,offset string

           mov cx,8000h              ; 32k=2^15=8000h

again:     cmp byte ptr [si], ‘$’   ;‘$’ ＝24h

           jnz next                  ; 不要采用 jz 进行分支

           mov byte ptr [si], ‘  ’  ;‘ ’＝20h

next:      inc si

           loop again                ; dec cx

                                    ; jnz again

（6）解答2：使用串操作指令

           mov di,offset string

           mov al,’$’

           mov cx,8000h

           cld

again:     scasb

           jnz next

           mov byte ptr es : [di-1], ‘  ’

next:      loop again

〔习题2.25〕对下面要求，分别给出3种方法，每种方法只用一条指令。   
（1）使CF=0    （2）使AX=0    （3）同时使AX=0和CF=0

〔解答〕

（1）解答：

         clc

         and ax,ax

         or ax,ax

（2）解答：

         xor ax,ax

         and ax,0

         mov ax,0

（3）解答：

         and ax,0

         xor ax,ax

         sub ax,ax

〔习题3.7〕假设myword是一个字变量，mybyte1和mybyte2是两个字节变量，指出下列语句中的错误原因。   
（1） mov byte ptr [bx],1000

（2） mov bx,offset myword[si]

（3） cmp mybyte1,mybyte2

（4） mov al,mybyte1+mybyte2

（5） sub al,myword

（6） jnz myword

〔解答〕

（1）1000超出了一个字节范围

（2）寄存器的值只有程序执行时才能确定，而offset是汇编过程计算的偏移地址，故无法确定，改为lea bx,myword[si]

（3）两个都是存储单元，指令不允许

（4）变量值只有执行时才确定，汇编过程不能计算

（5）字节量AL与字量myword，类型不匹配

（6）Jcc指令只有相对寻址方式，不支持间接寻址方式

〔习题3.11〕请设置一个数据段mydataseg，按照如下要求定义变量：

（1） my1b为字符串变量：Personal Computer

（2） my2b为用十进制数表示的字节变量：20

（3） my3b为用十六进制数表示的字节变量：20

（4） my4b为用二进制数表示的字节变量：20

（5） my5w为20个未赋值的字变量

（6） my6c为100的常量

（7） my7c表示字符串：Personal Computer

mydataseg segment

my1b      db ‘Personal Computer’

my2b      db 20

my3b      db 14h           ;20h

my4b      db 00010100b

my5w      dw 20 dup(?)

my6c      equ 100          ;my6c = 100

my7c      equ <Personal Computer>

mydataseg ends

〔习题4.22〕过程定义的一般格式是怎样的？子程序入口为什么常有PUSH指令、出口为什么有POP指令？下

面的程序段有什么不妥吗？若有，请改正：

crazy      PROC                  ；crazy PROC

           push ax               ；

           xor ax,ax             ；  xor ax,ax

           xor dx,dx             ；  xor dx,dx

again:     add ax,[bx]           ；again:    add ax,[bx]

           adc dx,0              ；  adc dx,0

           inc bx                ；  inc bx

           inc bx                ；  inc bx

           loop again            ；  loop again

           ret                   ；  ret

           ENDP crazy            ； crazy    ENDP

将bx中的无符号数按照10进制显示在屏幕上

stack\_area segment

db 60h dup(?)

top label word

stack\_area ends

proname segment

main proc far

assume cs:proname,ss:stack\_area

start:

mov ax,stack\_area

mov ss,ax

lea sp,top

mov bx,1234h

mov si,10d

xor cx,cx

mov ax,bx

next:

mov dx,0

div si

push dx

inc cx

cmp ax,0

jnz next

print:

pop dx

add dl,30h

mov ah,2

int 21h

loop print

mov ah,4ch

int 21h

main endp

proname ends

end start

实验内容

1、数据段中X、Y、Z、V均为字变量，存放的是16位带符号数。编写汇编语言程序计算表达式值

（V–（X\*Y+Z-480））/X。要求：

（1）Z的初值为你学号的最低2位数；

（2）使用MASM和LINK生成可执行程序后，使用Debug装入内存；

（3）使用U、D命令查看代码段和数据段的内容，特别注意数据段名及变量在代码段中的形式；使用G命令执行代码，并查看结果；

（4）用E命令修改各个变量的内容，再次用G命令执行并查看。

编译过程参考第三章PPT、各个命令的使用步骤参考DEBUG使用PPT。

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：

（1）使用地址表实现分支功能，在分支中输出提示信息，如第一个分支输出“The 1 Bit is 1”；

（2）编译后直接执行，该题后（包括该题）不得再使用debug！

3、求一个班50名学生成绩的最小分数,并将结果显示出来，各分数的初值自定义。

实验过程和程序

1：

设x,y,v为10，z为39

stack segment stack

db 512 dup(?)

stack ends

data segment

x dw 0010

y dw 0010

z dw 0039

v dw 0010

result dw 0

data ends

code segment 'code'

assume cs:code , ss:stack

assume ds:data

start:

mov ax,data

mov ds,ax

mov ax,x

mov bx,y

imul bx

mov cx,ax

mov bx,dx

mov ax,z

cwd

add cx,ax

adc bx,dx ;x\*y+z

mov ax,0480

cwd

sub cx,ax

sbb bx,0 ;x\*y+z-480

mov ax,v

cwd

sub ax,cx

sbb dx,bx ;v-(x\*y+z-480)

mov bx,x

idiv bx

mov result,ax

mov ah,004ch

int 21h

code ends

end start

2：

datas segment

str1 db 'the 1 bit is 1',0dh,0ah,'$'

str2 db 'the 2 bit is 1',0dh,0ah,'$'

str3 db 'the 3 bit is 1',0dh,0ah,'$'

str4 db 'the 4 bit is 1',0dh,0ah,'$'

str5 db 'the 5 bit is 1',0dh,0ah,'$'

str6 db 'the 6 bit is 1',0dh,0ah,'$'

str7 db 'the 7 bit is 1',0dh,0ah,'$'

str8 db 'the 8 bit is 1',0dh,0ah,'$'

datas ends

codes segment

assume cs:codes,ds:datas

start:

mov ax,datas

mov ds,ax

mov bl,00010000b

mov cx,8

mov si,0

again:

shl bl,1

jc next

add si,3

loop again

next:

add si,offset jmptable

jmp si

jmptable:

jmp near ptr l1

jmp near ptr l2

jmp near ptr l3

jmp near ptr l4

jmp near ptr l5

jmp near ptr l6

jmp near ptr l7

jmp near ptr l8

l1:

lea dx,str1

jmp output

l2:

lea dx,str2

jmp output

l3:

lea dx,str3

jmp output

l4:

lea dx,str4

jmp output

l5:

lea dx,str5

jmp output

l6:

lea dx,str6

jmp output

l7:

lea dx,str7

jmp output

l8:

lea dx,str8

output:

mov ah,9

int 21h

mov ah,4ch

int 21h

codes ends

end start

3：

stack segment stack

stack ends

data segment

score db 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49

data ends

code segment

assume cs:code,ds:data,ss:stack

start:

mov ax,data

mov ds,ax

mov al,score[40];任意取一个初值

cbw

mov cl,50

again:

cmp cl,0

jb print

mov bl,cl

add bl,bl

cbw

cmp score[bx],al

jb change

loop again

change:

mov al,score[bx]

print:

mov dx,ax

cbw

add dx,30h

mov ah,2

int 21h

jmp done

done:

mov ah,004ch

int 21h

code ends

end start

**1、编写一个子程序计算z=f(x,y)=x\*y+y-x+2020（x,y,z有符号数字数），要求：**

**（1）假设x\*y的结果还是16位；**

**（2）输入参数通过堆栈传送，输出参数通过AX传送；**

**（3）主程序调用后，结果需要输出到屏幕上。**

stack segment stack

dw 1024 dup(?)

stack ends

data segment

x dw 5

y dw 10

z dw 0

data ends

code segment 'code'

assume cs:code,ds:data,ss:stack

start: mov ax,data

mov ds,ax

sub sp,2 ;预留空间存放z

push y

push x

call cal ;调用子程序

pop z

mov ax,z

test ax,ax

;此处也可以用cmp ax,0等

jnz done1

mov dl,'0' ;为0，直接输出

mov ah,02h

int 21h

jmp over

done1: xor dx,dx

mov bx,10

push bx

add ax,0

jns done2

mov cx,ax

mov dl,'-'

mov ah,02h

int 21h

mov ax,cx

neg ax

done2: xor dx,dx

div bx

add dx,30h

push dx

cmp ax,0

jz done3

jmp done2

done3: pop dx

cmp dx,10

jz over

mov ah,02h

int 21h

jmp done3

over: mov ah,4ch

int 21h

cal proc

push bp

mov bp,sp

push ax

push bx

push dx

mov ax,[bp+4]

mov bx,[bp+6]

imul bx

add ax,[bp+4]

sub ax,bx

add ax,2020

mov [bp+8],ax

pop dx

pop bx

pop ax

pop bp

ret 4

cal endp

code ends

end start

1. 编写一个子程序计算z=f(x,y)=x/y+x-y（x,y,z有符号数内存字数）。要求通过堆栈传送输入参数，输出参数通过AX传递。(要求输入输出实现，且有提示)。

2. 编写一个宏，求三个数的最大数，原型为：MAX3 x,y,z,min，最大值要求输出。

3. 挂接1CH中断，正计时90秒后退出。要求屏幕显示0-89的秒数。

1. data segment

str1 db 'Please Input x',0dh,0ah,'$'

str2 db 'Please Input y',0dh,0ah,'$'

str3 db 'The Result is z',0dh,0ah,'$'

x db 0

y db 0

z db 0

data ends

stack segment

dw 128 dup(0)

stack ends

code segment

assume cs:code,ds:data,ss:stack

start: mov ax,data

mov ds,ax

mov ax,stack

mov ss,ax

mov sp,128

mov ah,09h

mov dx,offset str1

int 21h

mov ah,01h

int 21h

sub al,30h

mov x,al

mov ah,02h

mov dl,0ah

int 21h

mov ah,09h

mov dx,offset str2

int 21h

mov ah,01h

int 21h

sub al,30h

mov y,al

mov ah,02h

mov dl,0ah

int 21h

mov al,x

push ax

mov al,y

push ax

call far ptr Function

mov ax,4c00h

int 21h

Function proc far

push bp

mov bp,sp

push bx

push cx

push dx

mov cl,[bp+6]

mov bl,[bp+8]

mov al,bl

mov ah,0

idiv cl

sub bl,cl

add al,bl

call print

pop dx

pop cx

pop bx

pop bp

ret

Function endp

print proc

or al,30h

mov z,al

mov ah,09h

mov dx,offset str3

int 21h

mov ah,02h

mov dl,z

int 21h

ret

print endp

code ends

end start

2. ---------------------------------------------------------------------

data segment

X db 4

Y db 5

Z db 6

data ends

stack segment

dw 128 dup(0)

stack ends

MAX3 macro x,y,z

push bx

push cx

if x gt y

mov al,x

if z gt x

mov al,z

endif

else mov al,y

if z gt y

mov al,z

endif

endif

pop cx

pop bx

endm

code segment

assume cs:code,ds:data,ss:stack

start: mov ax,data

mov ds,ax

mov ax,stack

mov ss,ax

MAX3 X,Y,Z

mov ah,02h

or al,30h

mov dl,al

int 21h

mov ax,4c00h

int 21h

code ends

end start

3.-------------------------------------------------------------------------------------------------------

intno equ 1ch

data segment

oldisr dw ?,?

timer db 100

counter dw 90

isdone db 0

data ends

code segment

assume cs:code,ds:data

start:

mov ax,data

mov ds,ax

mov ax,0

mov es,ax

cli

mov ax,es:[intno\*4]

mov oldisr[0],ax

mov ax,es:[intno\*4+2]

mov oldisr[2],ax

sti

cli

mov word ptr es:[intno\*4],offset isr

mov word ptr es:[intno\*4+2],seg isr

sti

waithere:

cmp isdone,1

jnz waithere

exit:

cli

mov ax,oldisr[0]

mov es:[intno\*4],ax

mov ax,oldisr[2]

mov es:[intno\*4+2],ax

sti

mov ax,4c00h

int 21h

isr proc far

push dx

push ax

mov ax,data

mov ds,ax

sti

inc timer

again:

cmp timer,1000/55

jb done

mov timer,0

mov ah,2

mov dl,13

int 21h

mov ax,counter

mov dl,10

div dl

mov dh,ah

mov dl,al

mov ah,2

add dl,30h

int 21h

mov dl,dh

add dl,30h

int 21h

dec counter

jge done

mov isdone,1

done:

pushf

call dword ptr oldisr

cli

pop ax

pop dx

iret

isr endp

code ends

end star