**试验5**

**1.编写a.c的程序并解析**

void showchar(char a,int b);

main()

{

showchar(‘a’,2);

}

void showchar(char a,int b)

{

\*(char far\*)(0xb8000000+160\*10+80)=a;

\*(char far\*)(0xb8000000+160\*10+81)=b;

}

对应的汇编代码如下:

MAIN:

PUSH BP ;保存栈顶

MOV BP,SP ;建立新的的栈顶

MOV AX,0002 ;AX=0002

PUSH AX ;AX进栈

MOV AL,61 ;AL=61

PUSH AX ;AX进栈

CALL 020B ;进入子程序

POP CX ;将 0061 出栈到CX

POP CX ;将 0002 出栈到CX

POP BP ;恢复栈顶

RET

SHOWCHAR:

PUSH BP ;保存栈顶

MOV BP,SP ;建立新的的栈顶

MOV AL,[BP+04] ;因为 push ip ,push bp 占用4个单元,所以加4找到61H存放的单元

MOV BX,0B800H ;

MOV ES,BX ;ES=0B800H

MOV BX,0690H ;BX=0690H

MOV ES:[BX],AL ;把61H放到显存低8位显示字符

MOV AL,[BP+06] ;找到02H存放的单元

MOV BX,0B800H ;

MOV ES,BX ;

MOV BX,0691H ;

MOV ES:[BX],AL ;把61H放到显存高8位显示颜色

POP BP ;恢复栈顶

RET ;pop ip

结论：main函数通过（在子程序调用前）将要传递的参数压栈（从右向左一个一个压栈）传给showchar子函数。子函数返回后出栈将参数清除。

1. **编写b.c的程序并解析**

void showchar(int ,int ,...);

main()

{

showchar(8,2, ’a’, ’b’ , ’c’ , ’d’ , ’e’ , ’f’ , ’g’ , ’h’ );

}

void showchar(int n,int color,...)

{

int a; //因为多次使用变量a用si代表h整型变量a。

For(a=0;a!=n;a++) //开始赋值为零 xor si,si

//cmp si,[BP+04] [BP+04]存放的是8

//不等执行大括号程序，否则退出 用JNZ指令

{

\*(char far\*)(0xb8000000+160\*10+80+a+a)=\*(int \*)(\_BP+8+a+a) ;

// MOV BX,BP 将子函数建立的栈顶地址给BX

// ADD BX,SI

//ADD BX,SI 加了两次a

//ADD BX,08H 因为bp指向的地址下有 bp, ip, n, color 8个字节单元保存在栈中

//MOV AL,[BX] 找到存放字符’a’放到AL中

//PUSH AX 压栈保存字符’a’

//MOV AX,SI 将变量赋给AX

//CWD 扩展为dword型

//PUSH DX 压栈保存

//PUSH AX

//MOV AX,SI 将变量赋给AX

//CWD 扩展为dword型

//POP BX 将保存的dword型弹出

//POP CX

//ADD BX,AX 将两个dword型相加（对应a+a）高位放到CX中，低位放到BX中

//ADD CX,DX

//ADD BX,0690H

//ADC CX,0B800H

//MOV ES,CX

//POP AX 将保存的字符‘a' 弹出

//MOV [BX],AL 放到显存中

\*(char far\*)(0xb8000000+160\*10+81+a+a)=color;

//与上基本一样

//最后 inc si

}

结论：...可以代表一系列的参数，cwd 将word转换为dword型,同时验证上面的结论

}

以上我们只分析了showchar子函数。

1. **编写简单的printf函数，只支持%c,%d**

int wen=0;

void shao\_printf(char \*buffer,...);

main ()

{

shao\_printf(“the char %c value is:%d%n\0”,’a’,’a’);

shao\_printf(“i am happy !\0”);

}

void shao\_printf(char \*buffer,...)

{

int i,j,shao,ge,shi,bai;

i=0;//查找字符串

j=0;//查找参数

k=0;

while(buffer[i]!='\0')

{

if(buffer[i]==’%’)

{

if(buffer[i+1]==’d’)

{

shao=\*(int \*)(\_BP+6+j+j);

bai=shao/100+48;

shi=shao%100/10+48;

ge=shao%10+48;

\*(char far\*)(0xb8000000+160\*10+80+k+k+wen+wen)=bai ;

\*(char far\*)(0xb8000000+160\*10+80+k+k+2+wen+wen)=shi;

\*(char far\*)(0xb8000000+160\*10+80+k+k+4+wen+wen)=ge ;

k=k+3;

j++;

i=i+2;

}

else if(buffer[i+1]==’c’)

{

\*(char far\*)(0xb8000000+160\*10+80+k+k+wen+wen)=\*(int \*)(\_BP+6+j+j) ;

k++;

j++;

i=i+2;

}

else if(buffer[i+1]==’n’)

{

wen=wen+80;i=i+2;

}

}

else

{

\*(char far\*)(0xb8000000+160\*10+80+k+k+wen+wen)=buffer[i];

i++;

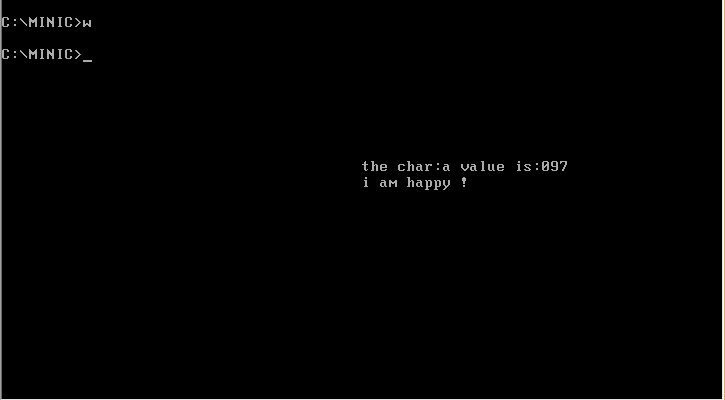
k++;

}

}

}

修改结果如下：



因为突然发现 多次输出会覆盖上一次的 所以我定义wen为全局变量且多加一个换行功能。