电子科技大学信息与软件工程学院

实 验 报 告

学　　号 2018091618008

姓　　名 袁昊男

（实验）课程名称 汇编语言程序设计

理论教师 赵　洋

实验教师 赵　洋

电子科技大学教务处制表

电 子 科 技 大 学

实　　验　　报　　告

学生姓名：袁昊男　　学号：2018091618008　　指导教师：赵洋

实验地点：在线实验　　实验时间：2020.05.25

一、实验室名称：信息与软件工程学院实验中心

二、实验名称：编写子程序实现显示控制

三、实验学时：2 学时

四、实验原理：

显示字符串是现实工作中经常要用到的功能，可以编写通用的子程序来实现这个功能。子程序主要解决两个问题，一是如何控制字符串在屏幕上的显示；二是如果输出的内容为二进制的数值，需要先成二进制到十进制的转换，再转换成 ASCII 码输出才能正确地显示结果。

五、实验目的：

编制两个子程序，提供灵活的调用接口，使调用者可以将 word 类型的数值以字符串方式输出，并可以决定显示的位置（行、列）、内容和颜色。

六、实验内容：

编程实现：

(1) 子程序：show_str

功能：在指定的位置，用指定的颜色，显示一个以 0 结束的字符串。

参数：(dh) = 行号（取值范围 0～24），(dl) = 列号（取值范围 0～79），(cl) = 颜色，ds:si 指向字符串的首地址

返回：无

(2) 子程序：dtoc

功能：将 word 型数据转变为表示十进制数的字符串，字符串以 0 为结尾符。

参数：(ax) = word 型数据，ds:si 指向字符串的首地址

返回：无

(3) 编制程序通过调用上述子程序将数值 12345 在屏幕的 8 行 3 列进行显示。

七、实验器材（设备、元器件）：

PC 微机一台

八、实验步骤：

1、 编辑源程序，建立一个以后缀.ASM 的文件。
2、 汇编源程序，检查程序有否错误，有错时回到编辑状态，修改程序中错误行，无错时继续第 3 步。
3、 连接目标程序，产生可执行程序。
4、 用 DEBUG 程序调试可执行程序，记录数据段的内容。

九、实验数据及结果分析

1、 实验思路

(1) 总体思路

在主程序中将待显示的字符串（'12345'）与字符属性（0cah）、显示位置（8 行 3 列）信息作为参数依次传递给 dtoc 和 show_str 子程序，并调用子程序即可实现在屏幕的第 8 行 3 列输出红底高亮闪烁绿色字符串'12345'。

(2) dtoc 子程序

dtoc 子程序将 word 型数据转变成十进制数的字符串，字符串以 0 为结符。主程序在调用 dtoc 子程序时已经将待显示字符串'12345'送入 ax 寄存器中，由于接下来的除法操作会将除法得到的商保存在 ax 中，因此在进行除法前需要将 ax 入栈保护。进行除法操作时，将除数 10 送入 bx 中，使用 div 命令进行除法操作后，余数存储在 dx 中并入栈、商存储在 cx 中。循环执行除法操作，直至得到的商为 0 时结束循环操作。再依次从栈中取出余数的 ASCII 码放入数据段中。由于数据段中初始化数据为全 0，因此在将 ASCII 码放入数据段后，该字符串是以 0 作为结尾符。

(3) show_str 子程序

show_str 子程序在指定的位置显示指定字符属性的字符串。主程序在调用 show_str 子程序时已经将待显示位置的行数送入 dl 寄存器中、将列数送入 dh 寄存器中，将字符属性送入 cx 寄存器中。接下来首先计算待显示位置的目的地址。在 80×25 彩色字符模式的显示缓冲区中，一个字符占 2 字节，因此每行共 160 字节。因此在第 dl 行前共有 dl－1 个完整的行，使用 mul 指令后即可得到行偏移量；与行偏移量同理可得列偏移量。将行与列的偏移量相加即可得到指定位置的总偏移量。此后将 data 段中准备好的字符 ASCII 码与字符属性组装成为 2 字节后，通过循环送入显示缓冲区的对应位置。在程序调用时注意相关寄存器的入栈保护与出栈恢复。

2、 流程图

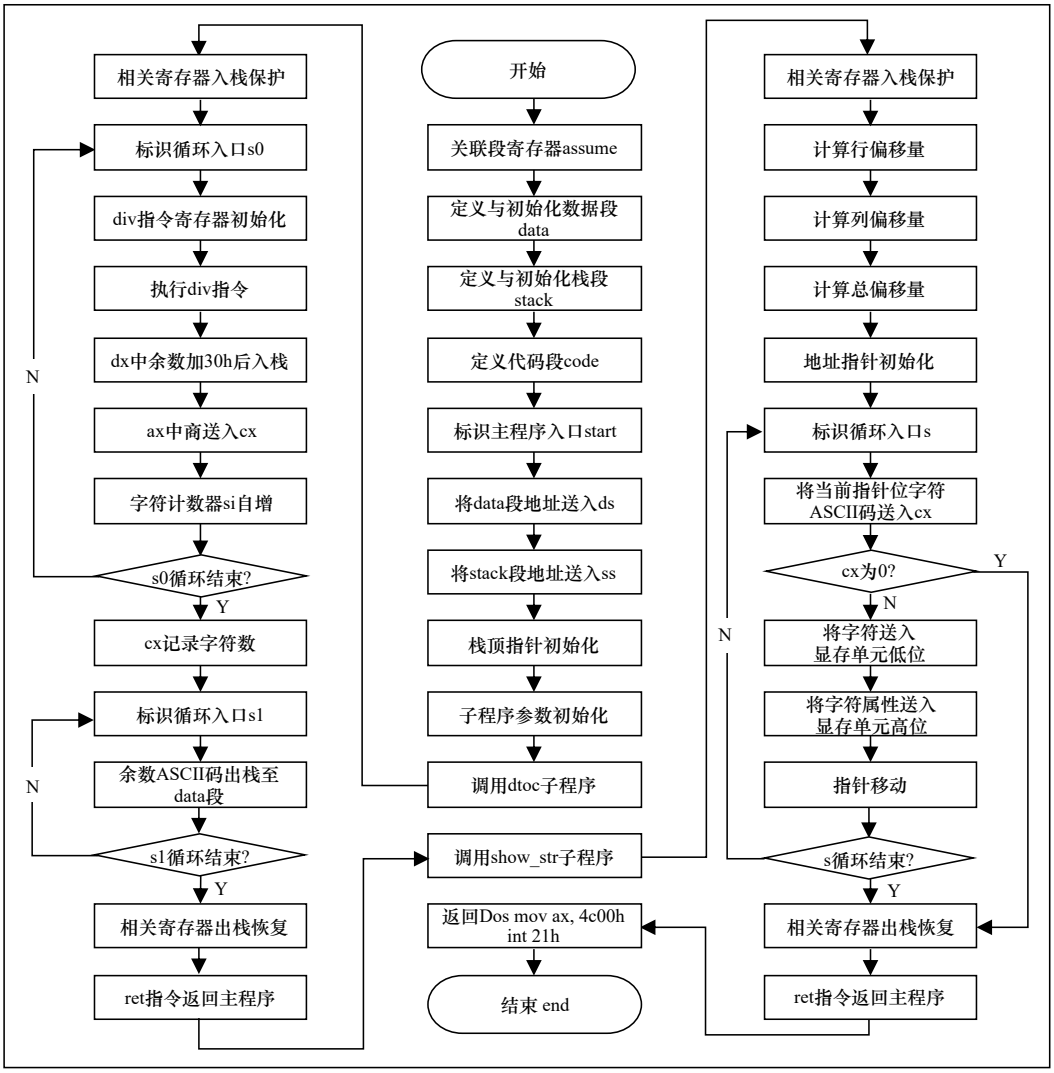

图 1　实验三流程图

3、 实验截图

图 2　运行结果

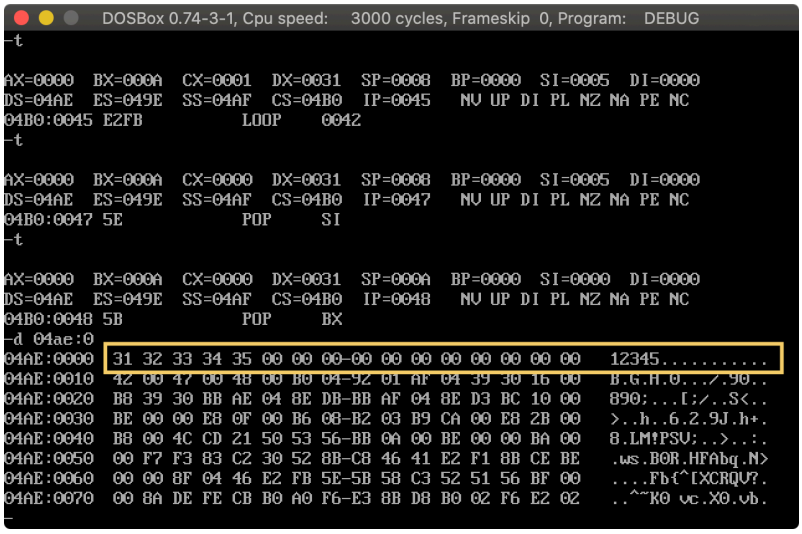

图 3　dtoc 子程序返回后 data 段数据

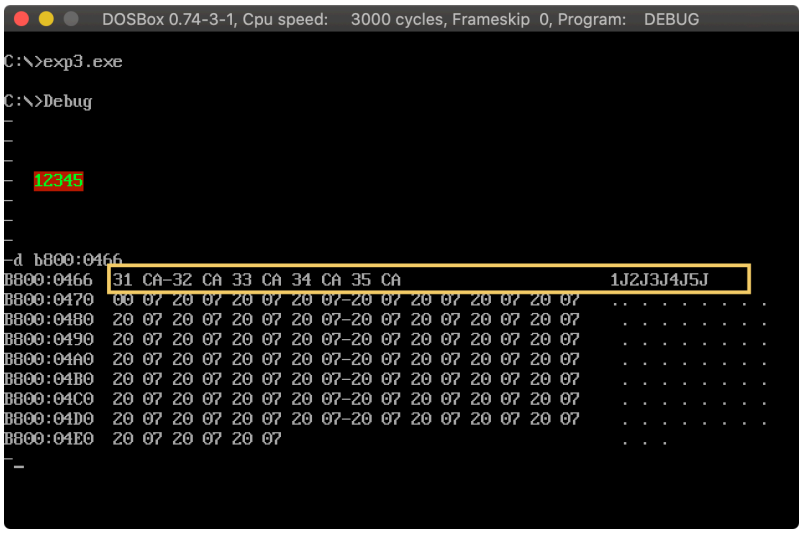

图 4　显示缓冲区中数据

4、 结果分析

程序能正确编译、连接，运行后可以按实验要求在屏幕的第 8 行 3 列输出红底高亮闪烁绿色字符串'12345'。经跟踪、调试，使用 D 命令查看 data 段中数据（如图 3 所示），可以发现待显示字符串 ASCII 码正确存储且字符串以 0 作为结尾符号；使用 D 命令查看显示缓冲区中数据（如图 4 所示），可以发现字符串对应编码按预设格式存储：字符 ASCII 码位于低位，红底高亮闪烁绿色属性（0cah）位于高位。

十、实验结论

代码经 masm 编译、link 连接成功，运行后可以按实验要求在屏幕的第 8 行 3 列输出红底高亮闪烁绿色字符串'12345'。经 Debug 跟踪、调试，data 段及显示缓冲区中数据按预设正确存储。

结合本实验编码过程，可以得出：可以通过向子程序传递参数、调用子程序、获得子程序返回值来实现功能，体现了汇编语言编程中的模块化设计。在实际编程中遇到的现实问题往往比较复杂，对现实问题进行层次分析，进而将其分解为相互联系的子问题，再通过编写子程序进行实现，降低了代码的耦合度，相互联系、功能独立的子程序模块能在一定程度上降低程序出错的可能性，提高了代码的可维护性。

十一、总结及心得体会

1、 在 80×25 彩色字符模式的显示缓冲区中，一个字符占 2 字节（ASCII 码与字符属性编码各占 1 字节），每行字符共占用 160 字节。可以根据待显示位置的行数、列数，结合 mul、add 指令计算出行偏移量、列偏移量与总偏移量，从而将指针定位至显示缓冲区的指定位置。

2、 call 与 ret 指令共同支持汇编语言编程中的模块化设计：call 指令实现程序的转移、ret 指令实现程序的返回。利用 call 与 ret 指令可以用简便的方法，实现多个相互联系、功能独立的子程序来解决一个复杂的问题。在主程序与子程序进行切换时，注意对发生冲突的寄存器入栈保护与出栈恢复。

3、 dtoc 子程序通过对二进制数值循环进行除法来实现十进制字符串的输出。注意当待输出数值过大时，div 除法指令可能产生溢出，可编写 divdw 子程序，以 dword 型被除数、word 型除数、dword 型结果进行 div 运算。

十二、对本实验过程及方法、手段的改进建议

本实验的讲解十分细致，建议增加避免产生除法溢出的子程序 divdw 编写，该子程序内容不算复杂，将三个子程序整合为一个实验使本实验更具综合性。

报告评分：

指导教师签字：

附录：实验程序源码

```
assume cs:code

data segment
    db 10 dup (0)
data ends

stack segment
    dw 8 dup(0)
stack ends

code segment
start:
       mov ax,12345      ; 待显示数值
       mov bx,data
       mov ds,bx         ; data 段地址送入 ds
       mov bx,stack
       mov ss,bx         ; stack 段地址送入 ss
       mov sp,10h        ; 栈顶指针初始化
       mov si,0
       call dtoc         ; 调用 dtoc 子程序

       mov dh,8          ; 行数
       mov dl,3          ; 列数
       mov cx,0cah       ; 字符属性
       call show_str

       mov ax,4c00h
       int 21h

dtoc:
       push ax
       push bx
       push si           ; 保护子程序寄存器中用到的寄存器
       mov bx,10         ; 除数
       mov si,0

  s0:
       mov dx,0          ; 余数位归零
       div bx            ; 执行 ax/bx
       add dx,30h        ; 余数加 30 得到对应十进制 ASCII 码
       push dx           ; 余数入栈
       mov cx,ax         ; 商送入 cx
       inc si            ; 记录循环次数
       inc cx            ; loop 判断
       loop s0           ; 先执行 cx-1,再判断 cx 是否为 0

       mov cx,si         ; cx 为循环次数
       mov si,0          ; si 指向 ds:[0]
  s1:
       pop ds:[si]       ; 将栈中转化好了的数据放到内存中
       inc si
       loop s1

       pop si            ; 出栈恢复
       pop bx            ; 出栈恢复
```

```
       pop ax              ; 出栈恢复
       ret                 ; 返回
show_str:
       push dx
       push cx
       push si             ; 保护子程序寄存器中用到的寄存器


       mov di,0            ; 显示缓存区中的偏移量
       mov bl,dh
       dec bl              ; 第 bl 行之前有 bl-1 个完整的行
       mov al,160
       mul bl              ; 每行 160 字节，用行数乘偏移量得到目标行的偏移量
       mov bx,ax           ; 乘积存储在 ax 中，送入 bx 中
       mov al,2            ; 列的偏移量为 2(第 1 字节为数值，第 2 字节为属性)
       mul dl              ; 与行偏移量同理
       add bl,al           ; 将列偏移量与行偏移量相加，得到指定位置的偏移量。

       mov ax,0b800h
       mov es,ax           ; 指定显示缓存区的内存位置

       mov ax,cx           ; 现将字符属性保存
   s:
       mov ch,0
       mov cl,ds:[si]      ; 首先将当前指向字符串的某个字符存入 cx 中
       jcxz ok             ; 如果 cx 为 0，则转移到 ok 标号执行相应代码
       mov es:[bx+di],cl   ; 将字符传入低地址
       mov es:[bx+di+1],ax ; 将颜色传入高地址
       add di,2            ; 列偏移量为 2
       inc si              ; 字符串的偏移量为 1
       loop s              ; 不为 0，继续复制

ok:    pop dx
       pop cx
       pop si              ; 还原寄存器变量
       ret                 ; 结束子程序调用

code ends
end start
```