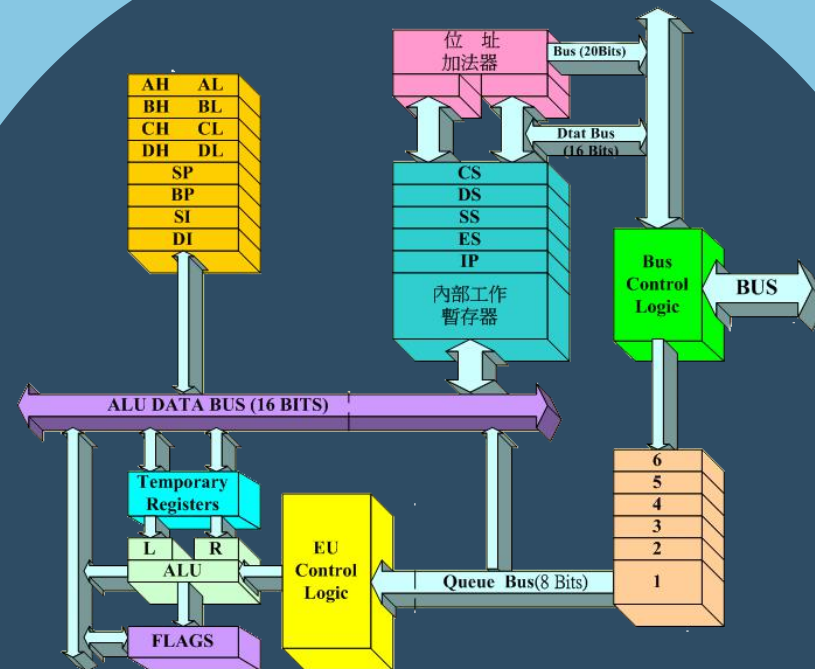



应用：字符串的输入

贺利坚 主讲




汇编语言程序设计
Assembly Language

要解决的问题

问题：设计一个最基本的字符串输入程序，需要具备下面的功能

- (1) 在输入的同时需要显示这个字符串；
- (2) 一般在输入回车符后，字符串输入结束；
- (3) 能够删除已经输入的字符——用退格键。

讨论

- (1) 在输入的同时需要显示这个字符串
需要用栈的方式来管理字符串的存储空间

- (2) 在输入回车符后，字符串输入结束。

输入回车符后，在字符串中加入0，表示字符串结束。

- (3) 在输入的同时需要显示这个字符串。

每次有新的字符输入和删除一个字符的时候，都应该重新显示字符串，即从字符栈的栈底到栈顶，显示所有的字符。

操 作	结 果
输入 "a"	a
输入 "b"	ab
输入 "c"	abc
输入 "d"	abcd
输入退格键	abc
输入退格键	ab
输入回车键	ab

程序的处理过程

- (1) 调用int 16h读取键盘输入；
- (2) 如果不是字符：①如果是退格键，从字符栈中弹出一个字符，显示字符栈中的所有字符；继续执行(1)；②如果是Enter 键，向字符栈中压入0，返回。
- (3) 如果是字符键：字符入栈；显示字符栈中的所有字符；继续执行(1)；

```
assume cs:code, ds:data
code segment
```

```
start:
```

```
    mov ax, data
    mov ds, ax
    mov si, 0
    mov dh, 12
    mov dl, 20
    call getstr
```

```
return: mov ax, 4c00h
        int 21h
```

```
getstr:... ;完整的接收字符串输入的子程序
code ends
end start
```

```
data segment
```

```
    db 32 dup (?)
```

```
data ends "栈"空间
```

```
getstr: push ax
getstrs: mov ah, 0
        int 16h
```

```
        cmp al, 20h
```

```
        jb nochar ; 小于20h为非字符
```

```
        ;字符入栈
```

```
        ;显示栈中的字符
```

```
        jmp getstrs
```

```
nochar: ;处理非字符
```

```
        cmp ah, 0eh ;退格键的扫描码
```

```
        je backspace
```

```
        cmp ah, 1ch ;回车键的扫描码
```

```
        je enter
```

```
        jmp getstrs
```

```
;对退格键、回车键的处理
```

```
        pop ax
```

```
        ret ; getstr结束
```

```
;字符(al)入栈
mov ah, 0;
call charstack
```

```
;字符出栈
mov ah, 1
call charstack
```

```
;显示栈中的字符
mov ah, 2
call charstack
```

```
backspace: ;退格
```

```
        ;字符出栈
```

```
        ;显示栈中的字符
```

```
        jmp getstrs
```

```
enter: ;回车
```

```
        mov al, 0
```

```
        ;0字符入栈
```

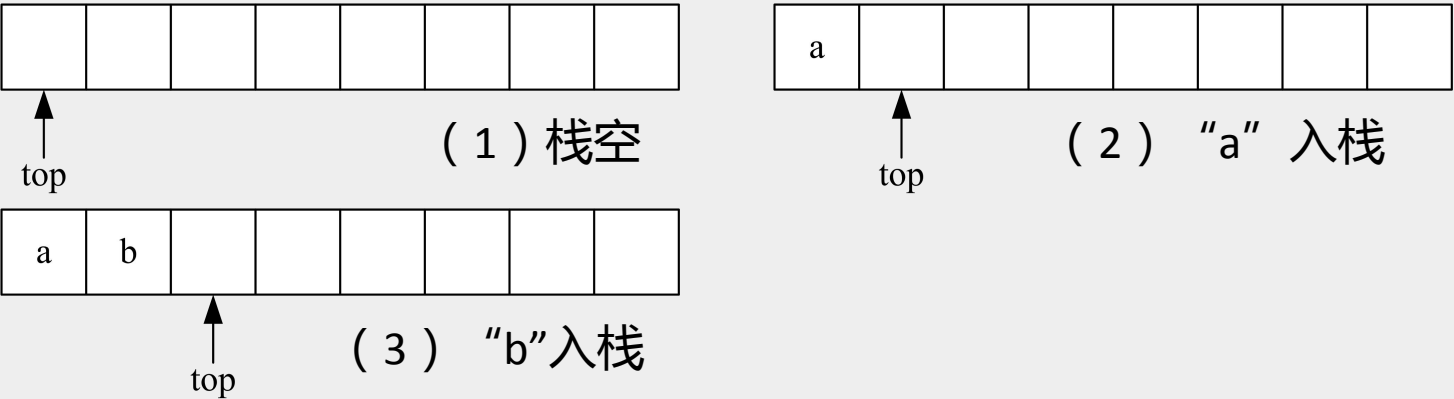
```
        ;显示栈中的字符
```

子程序：字符栈的入栈、出栈和显示

参数说明：

- 📁 (ah)=功能号，0表示入栈，1表示出栈，2表示显示；
- 📁 对于0号功能：(al)=入栈字符；
- 📁 对于1号功能：(al)=返回的字符；
- 📁 对于2号功能
 - 📄 (dh)、(dl) =字符串在屏幕上显示的行、列位置；
 - 📄 ds:si 指向字符串的存储空间，字符串以0为结尾符。

字符栈的访问规则



```
;字符(al)入栈  
mov ah,0;  
call charstack
```

```
;字符出栈  
mov ah,1  
call charstack
```

```
;显示栈中的字符  
mov ah,2  
call charstack
```

```
data segment  
    db 32 dup (?)  
data ends  
ds:si
```

“栈”空间

实现字符栈的入栈、出栈和显示

;功能子程序实现

charstack:

```
    jmp short charstart
    table dw
        charpush,charpop,charshow
    top dw 0 ;栈顶
```

charstart:

```
    push bx
    push dx
    push di
    push es
```

;实现各功能

```
sret: pop es
      pop di
      pop dx
      pop bx
      ret
```

```
code ends
end start
```

```
    cmp ah,2
    ja sret
    mov bl,ah
    mov bh,0
    add bx,bx
    jmp word ptr table[bx]
```

```
charpush:mov bx,top
          mov [si][bx],al
          inc top
          jmp sret
```

```
charpop:cmp top,0
         je sret
         dec top
         mov bx,top
         mov al,[si][bx]
         jmp sret
```

charshow:

在(dh)行(di)列显示

```
    mov bx,0b800h
    mov es,bx
    mov al,160
    mov ah,0
    mul dh
    mov di,ax
    add dl,dl
    mov dh,0
    add di,dx
```

```
    mov bx,0
```

```
charshows:cmp bx,top
           jne noempty
           mov byte ptr es:[di],' '
           jmp sret
```

```
noempty:mov al,[si][bx]
         mov es:[di],al
         mov byte ptr es:[di+2],' '
         inc bx
         add di,2
         jmp charshows
```

```
;字符入栈
mov ah,0;
call charstack
```

```
;字符出栈
mov ah,1
call charstack
```

```
;显示栈中的字符
mov ah,2
call charstack
```

```
data segment
    db 32 dup (?)
data ends
```

“栈”空间

ds:si