



哈尔滨工业大学（深圳）

实践教学报告

学 院： 计算机科学与技术学院

题 目： 实验四：IO 实验

姓 名： 李秋阳

学 号： 180110527

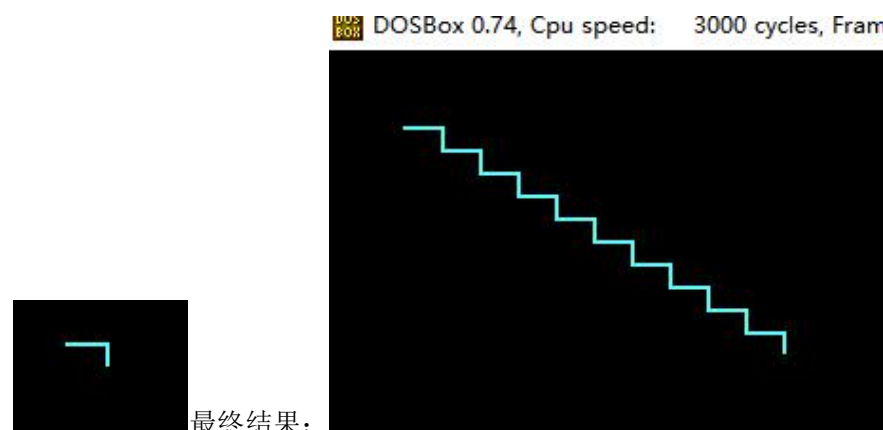
专 业： 计算机科学与技术

日 期： 2019 年 10 月 31 日

一、问题描述

必做题：

题目：通过 BIOS 和 DOS 功能调用实现“下楼梯功能”：(提示：用写像素中断画线)从键盘输入一个字符，控制下一个阶梯，一个阶梯由一条水平线和一条垂直线构成，如图



最终结果：

通过 video 中断、像素中断、dos 功能等实现动态显示“下楼梯功能”

选做题 1：

题目：创建一个文件，从键盘输入任意字母和数字（长度小于等于 10），然后删除字符串中的数字，将字母存到文件中。结果如图：

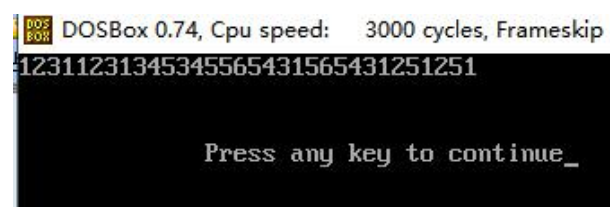


黑色屏幕中输入字符，白色文本中保存了字母

通过汇编语言实现的磁盘文件存取技术对筛选过的信息实现存储。

选做题 2：

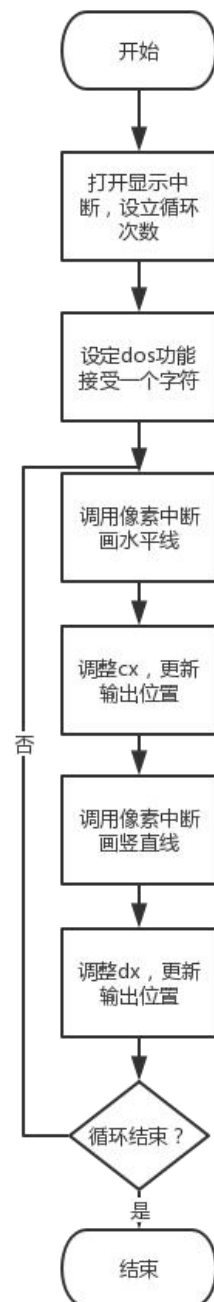
设计一个乐曲程序，通过计算机发出音响并奏出《两只老虎》的乐曲。程序中已经设置了乐谱。结果如图：屏幕显示音符的数字表示，同时扬声器发出对应音符声音



利用计算机控制发声的原理，实现演奏乐曲的程序。

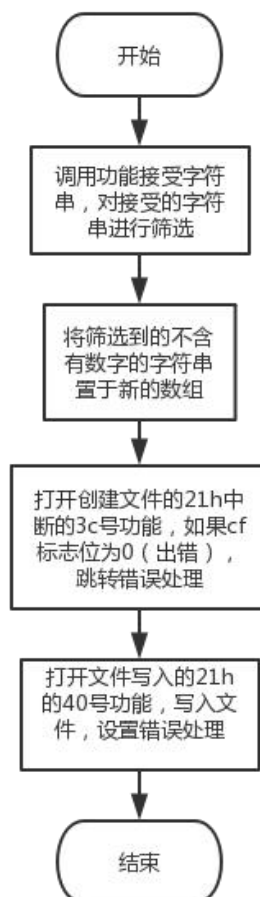
二、解决方案

必做题：本题目需要使用的中断是像素中断（在屏幕上画线）和显示中断，通过 int 16 的 0 号功能等待接受字符串作为输出的信号，在需要输出的位置用像素中断分别输出一道阶梯的水平线与竖直线。调用循环分别对 x 轴坐标和 y 轴坐标进行递增，更新需要输出的位置，实现功能。



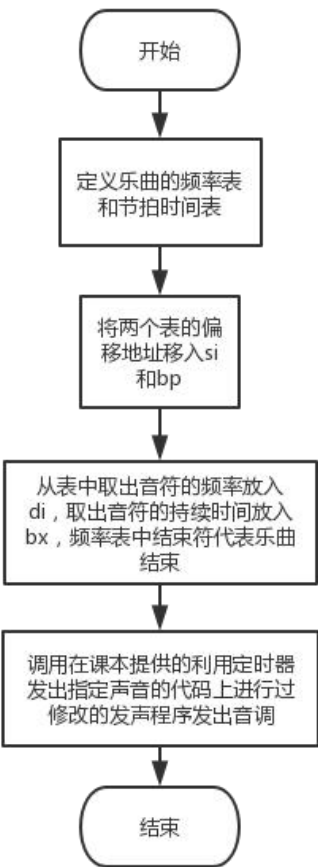
选做题 1: 文件的创建: 通过调用 21h 中断的 3ch 号功能实现。在数据段的 filename 单元中存储文件的 asciz 串。ah 置入功能号 3ch, ds:dx 指向 asciz 串, 即 filename, cx 置入文件属性 00 (一般文件)。调用 21h 中断后, 会改变 cf 标志位, 若创建成功, cf=1, 此时 ax 内存储文件代号, 将其放入 fh 的单元内存储。若失败, 则 cf=0, 可用 jc 指令跳转至错误处理。

文件的写入: 调用 21h 中断的写文件功能 (40h), 入口参数: (ah)=40h, (bx)=文件代号, 文件代号已存储在 fh 单元中, (cx)=要写入的字节数, 字节数即为字符串长度, 已记录在 strlen 单元中; ds:dx 指向 str 的首地址。若成功, 则 cf=0, (ax)=写入字节数; 若失败, 则 cf=1, 可通过 jc 指令跳转至错误处理。文件的关闭: 写入完成后, 必须关闭文件, 以确保操作系统将文件记录在磁盘上。其他参数不变, 将 3eh 置入 ah, 调用 21h 中断关闭文件。



选做题 2: 为演奏的乐曲定义一个频率表和一个节拍时间表。分别将两个表的偏

移地址移入 **si** 和 **bp**，从表中取出音符的频率放入 **di**，取出音符的持续时间放入 **bx**，频率表中结束符代表乐曲结束。调用在课本提供的利用定时器发出指定声音的代码上进行过修改的发声程序发出音调。



三、具体实现

必做题

```
datas segment
    w dw 10
    h dw 6
    turns dw 30
    start_x dw 20
    start_y dw 20
datas ends
```

```
stack segment stack
    dw 32 dup(?)
stack ends

code segment
    assume
cs:code,ds:datas,ss:stack

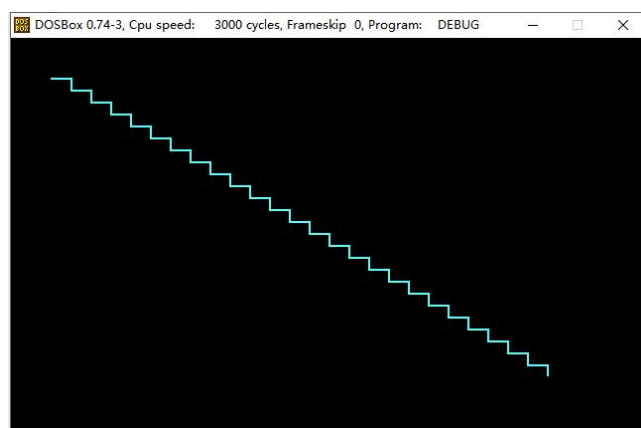
main proc near
```

```
start:
    mov ax,datas
    mov ds,ax
    mov ax,stack
    mov ss,ax
    mov sp,40h
    mov ah,0
    mov al,4
```

| | | |
|-------------------|----------------|----------------|
| int 10h | push cx | mov cx,start_x |
| mov cx,turns | push dx | mov dx,start_y |
| loops: | mov cx,w | mov al,1 |
| call hor_ver | li: | mov ah,0ch |
| mov ah,0 | push cx | int 10h |
| int 16h | mov cx,start_x | inc start_y |
| loop loops | mov dx,start_y | pop cx |
| mov ah,0 | mov al,1 | loop ld |
| mov al,3 | mov ah,0ch | pop dx |
| int 10h | int 10h | pop cx |
| mov ah,4ch | inc start_x | pop bx |
| int 21h | pop cx | pop ax |
| main endp | loop li | ret |
| hor_ver proc near | mov cx,h | hor_ver endp |
| push ax | ld: | code ends |
| push bx | push cx | end start |

结果

| | |
|--------------------|---------------------|
| -u | |
| 076F:0026 B44C | MOV AH,4C |
| 076F:0028 CD21 | INT 21 |
| 076F:002A 50 | PUSH AX |
| 076F:002B 53 | PUSH BX |
| 076F:002C 51 | PUSH CX |
| 076F:002D 52 | PUSH DX |
| 076F:002E 8B0E0000 | MOV CX,[0000] |
| 076F:0032 51 | PUSH CX |
| 076F:0033 8B0E0600 | MOV CX,[0006] |
| 076F:0037 8B160800 | MOV DX,[0008] |
| 076F:003B B001 | MOV AL,01 |
| 076F:003D B40C | MOV AH,0C |
| 076F:003F CD10 | INT 10 |
| 076F:0041 FF060600 | INC WORD PTR [0006] |
| 076F:0045 59 | POP CX |



选做题一

datas segment

```

buff db 100,0,100 dup(0) ;数据缓存区
outrec db 10 dup(0) ;存放筛选后结果
turns dw 0 ;记录字母个数
file_name db 'output1.txt',00h ;文件名
create_err db 0dh,0ah,'error:fail to create file!','$' ;创建文件错误
open_err db 0dh,0ah,'error:fail to openfile!','$' ;打开文件错误
write_err db 0dh,0ah,'error:fail to write file!','$' ;写文件错误
close_err db 0dh,0ah,'error:fail to close file!','$' ;关闭文件错误
whandle dw ? ;保存文件句柄

```

datas ends

stack segment

dw 32 dup(?)

stack ends

codes segment

assume

cs:codes,ds:datas,ss:stack

main proc near

start:

mov ax,datas

mov ds,ax

mov ax,stack

mov ss,ax

mov sp,40h

lea dx,buff

mov ah,0ah

int 21h

mov al,buff[1]

mov bx,0

mov bp,0

comp:

cmp bl,al

jz save

cmp buff[bx+2], 'a'

jb fai


| | | |
|---|---|--|
| cmp buff[bx+2],'z' jna suc cmp buff[bx+2],'a' jb fai cmp buff[bx+2],'z' jna suc jmp fai suc: inc turns mov ah,buff[bx+2] mov outrec[bp],ah inc bp fai: inc bx jmp comp save: mov ah,3ch mov cx,0 lea dx,file_name int 21h jc error1 mov ah,3dh | mov al,1 lea dx,file_name int 21h jc error2 mov whandle,ax mov ah,40h mov bx,whandle mov cx,turns lea dx,outrec int 21h jc error3 mov ah,3eh mov bx,whandle int 21h jc error4 jmp exit error1: lea dx , create_err mov ah , 9 int 21h jmp exit error2: | lea dx , open_err mov ah , 9 int 21h jmp exit error3: lea dx , write_err mov ah , 9 int 21h jmp exit error4: lea dx , close_err mov ah , 9 int 21h jmp exit exit: mov ah,4ch int 21h main endp codes ends end start |
|---|---|--|

结果

```
D:\>debug file.exe
-u
077D:0000 B86A07      MOV     AX,076A
077D:0003 8ED8          MOV     DS,AX
077D:0005 B87907      MOV     AX,0779
077D:0008 8ED0          MOV     SS,AX
077D:000A BC4000      MOV     SP,0040
077D:000D 8D160000     LEA     DX,[0000]
077D:0011 B40A          MOV     AH,0A
077D:0013 CD21          INT     21
077D:0015 A00100      MOV     AL,[0001]
077D:0018 BB0000      MOV     BX,0000
077D:001B BD0000      MOV     BP,0000
077D:001E 3ADB          CMP     BL,AL

-u
077D:00A2 90          NOP
077D:00A3 8D16B600     LEA     DX,[00B6]
077D:00A7 B409          MOV     AH,09
077D:00A9 CD21          INT     21
077D:00AB EB0C          JMP     00B9
077D:00AD 90          NOP
077D:00AE 8D16D200     LEA     DX,[00D2]
077D:00B2 B409          MOV     AH,09
077D:00B4 CD21          INT     21
077D:00B6 EB01          JMP     00B9
077D:00B8 90          NOP
077D:00B9 B44C          MOV     AH,4C
077D:00BB CD21          INT     21
077D:00BD 81EC8801     SUB     SP,0188
077D:00C1 56          PUSH    SI

-gb9
qw23ert6g
AX=3E06 BX=0006 CX=0006 DX=0066 SP=0040 BP=0006 SI=0000 DI=0000
DS=076A ES=075A SS=0779 CS=077D IP=00B9  NV UP EI PL ZR NA PE NC
077D:00B9 B44C          MOV     AH,4C
```



选做题 2

datas segment

mus_freq dw 262,294,330,262,262,294,330,262 ;mus_freq 数组存放的是乐谱,每一个数表示一个音符

dw 330,349,392,330,349,392,392,440

dw 392,349,330,262,392,440,392,349

dw 330,262,294,196,262,294,196,262

mus_time dw 2500,2500,2500,2500,2500,2500,2500,2500 ;mus_freq 中对应音符持续时间

```

dw 5000,2500,2500,5000,1200,1200,1200,1200,2500,2500

dw 1200,1200,1200,1200,2500,2500,2500,2500,5000,2500,5000

mes      db 31h,32h,33h,31h,31h,32h,33h,31h,33h,34h,35h,33h,34h,35h,35h,36h      ;
乐谱的数字表示
db 35h,34h,33h,31h,35h,36h,35h,34h,33h,31h,32h,35h,31h,32h,35h,31h,'$'

datas ends

stack segment
dw 32 dup(?)
stack ends
codes segment
assume
cs:codes,ds:datas,ss:stack
start:
mov ax,datas
mov ds,ax
mov ax,stack
mov ss,ax
mov sp,40h
mov bp,0
mov si,0
in al,61h
and al,0fch
loops:
cmp mes[bp],'$'
jz exit
mov dl,mes[bp]
mov ah,2
int 21h
mov al,0b6h
out 43h,al
mov
di,mus_freq[bp+si]
mov dx,12h
mov ax,348ch
div di
out 42h,al
mov al,ah
out 42h,al
in al,61h
mov ah,al
or al,3
out 61h,al
call del
inc bp
inc si
dec dx
mov al,ah
out 61h,al
jne loops
exit:
mov ah,4ch
int 21h
del proc near
push ax
push bx
push cx
push dx
mov bx,500
wait1:
mov
cx,mus_time[bp+si]
delay:
loop delay
dec bx
jnz wait1
pop dx
pop cx
pop bx
pop ax
ret
del endp
codes ends
end start

```

结果:

```
D:\>debug music.exe
-u
0779:0000 B86A07      MOV     AX,076A
0779:0003 8ED8        MOV     DS,AX
0779:0005 B87507      MOV     AX,0775
0779:0008 8ED0        MOV     SS,AX
0779:000A BC4000      MOV     SP,0040
0779:000D BD0000      MOV     BP,0000
0779:0010 BE0000      MOV     SI,0000
0779:0013 E461      IN      AL,61
0779:0015 24FC      AND     AL,FC
0779:0017 3E        DS:
0779:0018 80BE800024  CMP     BYTE PTR [BP+0080],24
0779:001D 7434      JZ      0053
0779:001F 3E        DS:
0779:0020 8A968000  MOV     DL,[BP+0080]

-u
0779:0045 E661      OUT     61,AL
0779:0047 E80D00      CALL    0057
0779:004A 45        INC     BP
0779:004B 46        INC     SI
0779:004C 4A        DEC     DX
0779:004D 8AC4      MOV     AL,AH
0779:004F E661      OUT     61,AL
0779:0051 75C4      JNZ     0017
0779:0053 B44C      MOV     AH,4C
0779:0055 CD21      INT     21
0779:0057 50        PUSH    AX
0779:0058 53        PUSH    BX
0779:0059 51        PUSH    CX
0779:005A 52        PUSH    DX
0779:005B BBF401  MOV     BX,01F4
0779:005E 3E        DS:
0779:005F 8B8A4000  MOV     CX,[BP+SI+0040]
0779:0063 E2FE      LOOP    0063

-g53
12311231345345565431565431251251
AX=3030 BX=0000 CX=015D DX=00D5 SP=0040 BP=0020 SI=0020 DI=0106
DS=076A ES=075A SS=0775 CS=0779 IP=0053  NU UP EI PL ZR NA PE NC
0779:0053 B44C      MOV     AH,4C
```

四、实验总结

经过四次汇编实验,我开始重新思考汇编语言这一门学科的意义。如何理解一台计算机的运作呢?我们常用的一些编程语言是怎样通过编译器实现的呢?汇编语言实验让我结合理论知识去了解更加低层次的计算机。汇编语言是计算机技术的基础,之所以说汇编重要,其一个重要的原因就是,汇编语言让我更好的理解高级语言。汇编语言对于内存的操作都是基于内存地址的,而高级语言中的指针概念,说白了就是内存的地址。指针的学习和应用中最头疼的就是在指针这个抽象的概念和实际的内存单元之间建立

思维映射，而这些恰恰是我们在汇编语言学习中频繁做的一件平常事。另外，对于高级语言中的数据类型、形参实参、函数调用、全局变量、局部变量等概念及操作，我们都可以用汇编语言中的一些操作相关联，把这些抽象的概念和过程，通过汇编语言形成一个具体的映像，深度剖析，这样我们才能真正学会、学好高级语言。