**微机原理与技术接口**

**课程设计报告**

**学生姓名： 葛英杰**

**学 号： 201805120611**

**专业班级： 计算机科学与技术183班**

**任课教师： 舒红波**

目录

[1. 程序总体介绍 3](#_Toc62207414)

[2. 大体功能模块 3](#_Toc62207415)

[3. 运用到的所学的知识 3](#_Toc62207416)

[4.操作实现流程图 3](#_Toc62207417)

[5.心得体会 6](#_Toc62207418)

[6.源代码 6](#_Toc62207419)

# 程序总体介绍

本程序主要实现了多任务。扬声器调用，图形绘制，按键输入等功能，以及在移动过程中变色，暂停的功能

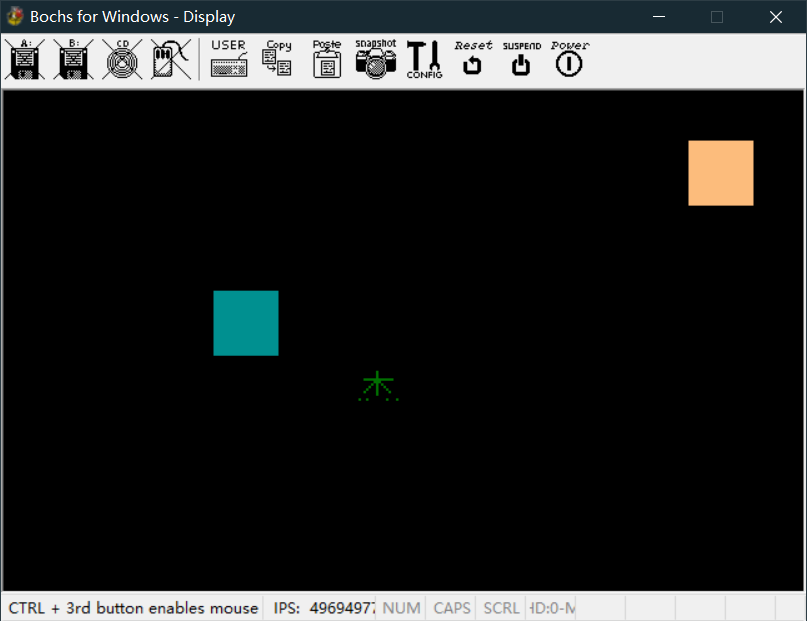
# 大体功能模块

1. 普通中断
2. 按键中断
3. 扬声器
4. 图形绘制
5. 中文点阵绘制
6. 定时器

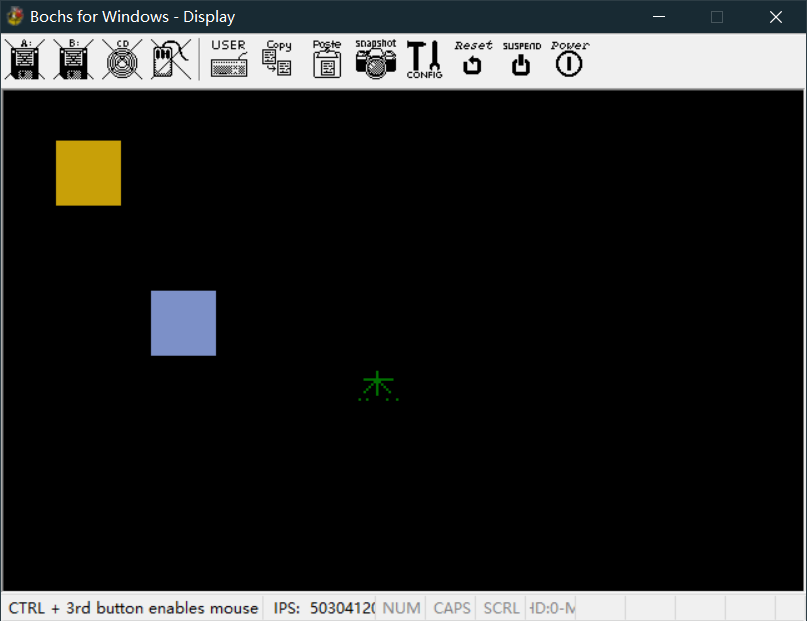
# 运用到的所学的知识

1. 中断
2. 扬声器
3. 键盘中断
4. 像素点打印
5. 定时器

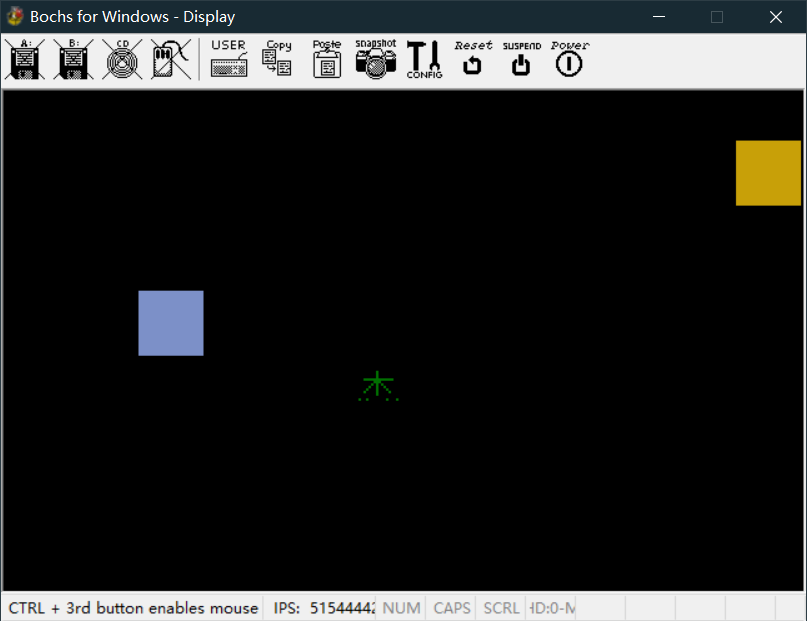
# 4.操作实现流程图



按1，2，3，4变换颜色

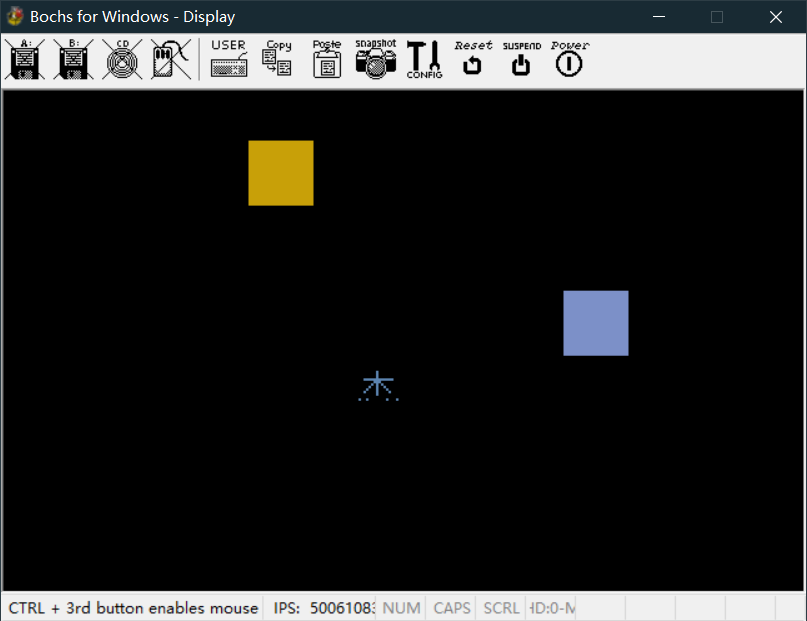


E停止



Ctrl+a

名字变色



# 5.心得体会

通过这次开发，我初步掌握了汇编语言的用法，实现了定时器，像素点绘制，键盘中断等功能，虽然实现的任务比较简单，但实际上代码量也很长，因为所有的任务逻辑都需要自己来实现，而没有现成的函数，而且寄存器数量有限，需要用Push和Pop压入堆栈中使用，才能不导致程序段之间冲突。

# 6.源代码

org 0x8400

jmp main

;变量定义区

sx dw 77 ;临时变量，用来存储当前任务中矩形的位置

sy dw 88 ;临时变量，用来存储当前任务中矩形的位置

ex dw 154 ;临时变量，用来存储当前任务中矩形的位置

ey dw 88 ;临时变量，用来存储当前任务中矩形的位置

fg dw 1

width dw 25 ;宽度

hight dw 25 ;高度

startx1 dw 80 ;1号矩形的位置

starty1 dw 80

startx2 dw 20 ;二号矩形的位置

starty2 dw 20

colnum db 78 ;颜色相关

colr db 55

colg db 44

colb db 44

flag db 0 ;是否暂停

combineflag db 0 ;组合键标志

left\_1 db 1 ;1号矩形是否左移标志

left\_2 db 0 ;2号矩形是否左移标志

i dw 1

j dw 1

f dw 1000 ;频率

timerlow dw 0x30 ;定时器相关

timerhigh dw 0x60

name3 dw 0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0

dw 0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0

dw 0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0

dw 0,0,0,1,1,1,1,1,1,1,1,1,1,1,1,0,0,0,0,0

dw 0,0,0,0,0,0,0,1,1,1,0,0,0,0,0,0,0,0,0,0

dw 0,0,0,0,0,0,1,0,1,0,1,0,0,0,0,0,0,0,0,0

dw 0,0,0,0,0,1,0,0,1,0,0,1,0,0,0,0,0,0,0,0

dw 0,0,0,0,1,0,0,0,1,0,0,0,1,0,0,0,0,0,0,0

dw 0,0,0,1,0,0,0,0,1,0,0,0,0,1,0,0,0,0,0,0

dw 0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0

dw 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0

dw 0,1,0,0,1,0,0,0,0,0,0,0,1,0,0,0,1,0,0,0,

;=========================================================

; 主函数

;=========================================================

main:

sti ;中断标志

mov ah,00h

mov al,13h

int 10h

mov bx,0xa000 ;绘图

mov es,bx

mov word [ds:0x24], kb

mov word [ds:0x26], 0

beg:

cmp byte [flag], 1 ;如果flag为1，直接进入中断结束，死循环，直至重新再按下e

je stop

mov word [ds:0x20], task1

mov word [ds:0x22], 0

; call slp

mov word [ds:0x20], task2

mov word [ds:0x22], 0

; call slp

mov word[ds:0x20],task\_printname

mov word[ds:0x22],0

; call slp

jmp beg

stop:

mov word [ds:0x20], task3

mov word [ds:0x22], 0

jmp beg

jmp $

;========================================

; 任务1，打印第一个矩形

;========================================

task1:

push ax

push dx

;cmp byte[flag],1

; je task1\_exit

mov word[timerhigh],0x60

mov word[timerlow],0x20

call timeset

mov ax, word [startx1]

mov word [sx], ax

mov ax, word [starty1]

mov word [sy], ax

mov byte [colnum], 0

call printsoildrec

cmp word [startx1], 294

je switchleft1

cmp word [startx1], 0

je switchleft0

jmp task1print

;一号矩形方向，打印相关

switchleft0:

mov byte [left\_1], 0

call output

jmp task1print

switchleft1:

mov byte [left\_1], 1

call output

jmp task1print

task1print:

mov byte [colnum], 77

cmp byte [left\_1], 1

je printleft

printright: ;右移打印

add word [sx], 1

add word [startx1], 1

call printsoildrec

jmp task1\_exit

printleft: ;左移打印

sub word [sx], 1

sub word [startx1], 1

call printsoildrec

jmp task1\_exit

task1\_exit:;结束这一次1号矩形的中断

pop ax

pop dx

mov al, 0x20

mov dx, 0x20

out dx, al

iret

;====================================

; 任务2，打印第二个矩形

;====================================

task2:

push ax

push dx

; cmp byte[flag],1

; je task2\_exit

mov byte[timerlow],0x00

mov byte[timerhigh],0x40

call timeset

mov ax, word [startx2]

mov word [sx], ax

mov ax, word [starty2]

mov word [sy], ax

mov byte [colnum], 0

call printsoildrec

; call slp

cmp word [startx2], 294 ;起始点碰壁

je switchleft1\_2

cmp word [startx2], 0 ;碰撞检测

je switchleft0\_2

jmp task2print

switchleft0\_2:

mov byte [left\_2], 0

call output

jmp task2print

switchleft1\_2:

mov byte [left\_2], 1

call output

jmp task2print

task2print:

mov byte [colnum], 66

cmp byte [left\_2], 1

je printleft\_2

printright\_2:

add word [startx2], 1

add word [sx], 1

call printsoildrec

jmp task2\_exit

printleft\_2:

sub word [startx2], 1

sub word [sx], 1

call printsoildrec

jmp task2\_exit

task2\_exit:

pop ax

pop dx

mov al, 0x20

mov dx, 0x20

out dx, al

iret

;==============

;任务3，死循环

;==============

task3:

push ax

push dx

task3\_exit:

pop ax

pop dx

mov al, 0x20

mov dx, 0x20

out dx, al

iret

;====================

; 打印名字任务

;====================

task\_printname:

push di

push bp

push bx

push ax

mov di,name3

printname:

cmp word[j],12

ja end1

inc word[j]

mov word[i],1

printword:

cmp word[i],20

ja printname

mov bp,[di]

cmp bp,1

jnz next

print:

mov bx,32140

mov ax,word[j]

mov dx,320

mul dx

add bx,ax

add bx,word[i]

add bx,3200

mov byte[es:bx],120

add di,2

inc word[i]

jmp printword

next:

add di,2

inc word[i]

jmp printword

end1:

mov al,0x20

mov dx,0x20

out dx,al

pop ax

pop dx

pop di

pop bp

iret

;============================

; 键盘输入任务

;============================

kb: ;键盘输入

push dx

push ax

mov dx, 0x60

in al, dx

cmp al,0x1d

je combinebeg

cmp al,0x1e

je combinekey

cmp al, 0x12 ;判断输入是否为小写的e

je kb1

cmp al,0x02

je kb2

cmp al,0x03

je kb3

cmp al,0x04

je kb4

cmp al,0x05

je kb5

jmp k\_exit

kb1:

mov byte [combineflag],0

cmp byte [flag], 1

je switch0

cmp byte [flag], 0

je switch1

jmp k\_exit

kb2:

mov byte [combineflag],0

mov word [colnum],77

mov word [colr],0

mov word [colg],100

mov word [colb],100

call setcol

jmp k\_exit

kb3:

mov byte [combineflag],0

mov word [colnum],77

mov word [colr],95

mov word [colg],100

mov word [colb],50

call setcol

jmp k\_exit

kb4:

mov byte [combineflag],0

mov word [colnum],66

mov word [colr],50

mov word [colg],40

mov word [colb],66

call setcol

jmp k\_exit

kb5:

mov byte [combineflag],0

mov word [colnum],66

mov word [colr],10

mov word [colg],8

mov word [colb],9

call setcol

jmp k\_exit

combinebeg: ;组合键判定

cmp byte[combineflag],0

jnz setflag

mov byte[combineflag],1

jmp k\_exit

setflag:

mov byte[combineflag],0

jmp k\_exit

combinekey:

cmp byte[combineflag],0

je unmatchedkey

mov byte[combineflag],0

mov word[colnum],120

mov word[colr],23

mov word[colg],33

mov word[colb],44

call setcol

jmp k\_exit

switch0:

mov byte [flag], 0

jmp k\_exit

switch1:

mov byte [flag], 1

jmp k\_exit

unmatchedkey:

mov byte [combineflag],0

k\_exit: ;键盘中断结束

mov dx, 0x20

mov al, 0x61

out dx, al

pop ax

pop dx

iret

;=========================

; 打印矩形

;=========================

printsoildrec: ;绘制正方形

push cx ;总体思路：每次循环都会打印一行像素点，然后返回到psrbeg，使ey,sy+1，打印下一行，如果cx大于hight表示已经打印完毕，把ey和sy还原到初始值结束

push ax ;cx表示已经打印的行数,终点的位置放于ax

mov cx,0

mov ax,[sx]

add ax,[width]

mov [ex],ax

mov ax,[sy]

mov [ey],ax

mov word [fg],1

printrecbeg: ;纵坐标是否+1的判断

cmp cx,[hight]

ja psrexit

call printline

add word [ey],1

add word [sy],1

add cx,1

jmp printrecbeg

psrexit:

mov cx, [hight]

add cx, 1

sub word [ey], cx

sub word [sy], cx

pop ax

pop cx

ret

printline:

push bx

push ax

push bp

mov bx,[sy] ;计算目标打印起始点的在屏幕中的位置，存于bx中

mov ax,320

mul bx

add ax,[sx]

mov bx,ax

mov bp,[ey] ;计算目标打印终点在屏幕中的位置，存于bp中

mov ax,320

mul bp

add ax,[ex]

mov bp,ax

mov al,[colnum]

printlineswitch:

cmp bx,bp ;如果起点坐标大于终点坐标直接结束

ja plexit

mov byte [es:bx],al ;打印像素点

add bx,[fg]

jmp printlineswitch

plexit:

pop bp

pop ax

pop bx

ret

setcol: ;颜色设置

push dx

push ax

mov dx,0x3c8

mov al,[colnum]

out dx,al

mov dx,0x3c9

mov al,[colr]

out dx,al

mov al,[colg]

out dx,al

mov al,[colb]

out dx,al

pop dx

pop ax

ret

slp:

push di

mov di, 0

begslp:

inc di

cmp di,0x10

ja begslp

pop di

ret

;============================

; 扬声器

;===========================

;dive:

; mov dx,12h

; mov ax,34deh

; div di

; ret

output:

push ax

push dx

push di

mov di,[f]

;call dive

mov dx,12h

mov ax,34deh

div di

mov al,10110110b

out 43h,al

out 42h,al

mov al,ah

out 42h,al

;call open

in al,61h

or al,00000011b

out 61h,al

;call delay

mov dx,5h

mov ax,0

s: sub ax,1

sbb dx,0

cmp ax,0

jne s

cmp dx,0

jne s

;call close

in al,61h

and al,11111101b

out 61h,al

pop ax

pop dx

pop di

ret

;=========================

; 定时器

;=========================

timeset:

push ax

push dx

mov dx,0x43

mov al,0x34

out dx,al

mov dx,0x40

mov al,[timerlow]

out dx,al

mov dx,0x40

mov al,[timerhigh]

out dx,al

pop ax

pop dx

ret