

课 程 设 计 报 告

课程名称： 微机原理

题 目： 简易坦克大战

班 级：计算机科学与技术183

学 号： 201805120624

姓 名： 吴昊

**2021年1月**

## 程序总体介绍

本程序使用vscode作为编辑器，nasm作为编译器，使用bochs进行运行，实现的功能有

1. 实现了开始界面，游戏界面、胜利界面和失败界面

2. 在游戏界面带有背景音乐，背景音乐是一段曲调的无限循环

3. 使用多线程使得弹框、背景音乐、发射子弹可以同时进行

4. 使用WSAD控制坦克，按j发射子弹，一次只能放一个子弹，按e从开始界面切换到游戏界面，按p暂停

5. 子弹碰到边界会消失，子弹通过运算只会出现在当前坦克的炮口前端向前发射

6. 坦克碰到边界就被阻挡，无法再次前进

7. 使用坦克攻击上方的障碍、障碍会掉血并且变色，累计攻击五次取得游戏胜利

8. 坦克碰到上方的障碍会死亡，游戏失败

9. 暂停按钮，有着浮起和按下两种状态显示，按下时暂停主页面的一切定时器活动

10. 右下角底部有吴字进行左右移动，碰到边界会自动改变方向

## 大体功能模块：

1. 主函数模块，主要控制页面切换以及使用定时器进行多任务划分和页面布置
2. 键盘驱动模块：响应wsadjpe这几个键并且完成相应的功能
3. 坦克控制模块：根据当前不同的值选择不同的坦克以及绘制
4. 子弹控制模块：判断当前有无子弹，控制子弹飞行，以及进行子弹碰撞检测
5. 音乐播放模块：播放一段无限循环的背景音乐
6. 底部汉字移动模块：控制底部汉字的左右移动
7. 空任务模块：用于暂停和非游戏页面的定时器切换

## 运用到的所学的知识

1. 进入320\*200的图形界面模式， 并且可以进行绘图
2. 使用端口修改相应色号的颜色
3. 键盘驱动，通过按下不同的返回值进行不同的响应
4. 使用定时器实现多任务
5. 修改定时器频率
6. 设置赫兹和打开关闭蜂鸣器

## 操作实现流程图

1. 开始界面

*gameStart:*

    sti

    mov byte [*colNum*], 3

    mov word [*startX*], 100

    mov word [*startY*], 80

    mov word [*startAddr*], *KAI*

    call *print*

    mov word [*startX*], 120

    mov word [*startY*], 80

    mov word [*startAddr*], *SHI*

    call *print*

    mov word [*startX*], 140

    mov word [*startY*], 80

    mov word [*startAddr*], *YOU*

    call *print*

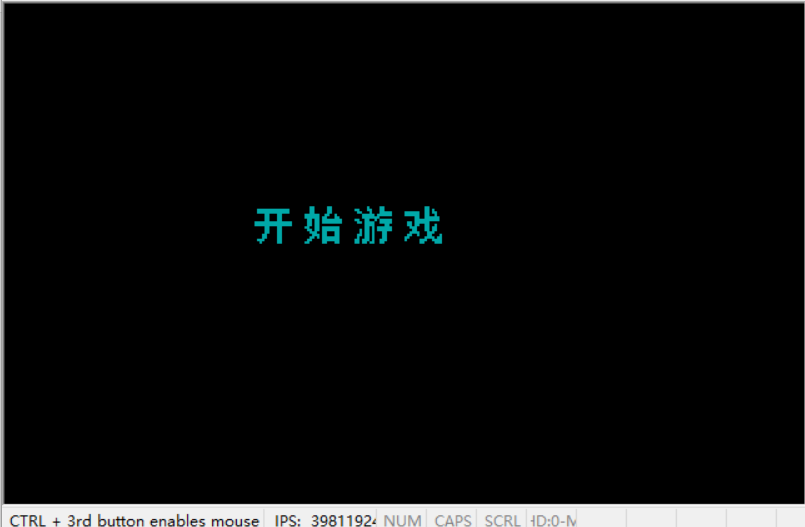
    mov word [*startX*], 160

    mov word [*startY*], 80

    mov word [*startAddr*], *XI*

    call *print*

    jmp *mainControl*



1. 坦克根据WSAD进行移动和进行方向变更，碰到边界无法继续移动

  cmp byte [*dir*], 1 ;方向变更代码

    je *printTank1*

    cmp byte [*dir*], 2

    je *printTank2*

    cmp byte [*dir*], 3

    je *printTank3*

    cmp byte [*dir*], 4

    je *printTank4*

*printTank1:*

    mov word [*startAddr*], *TANK1*

    jmp *printTank*

*printTank2:*

    mov word [*startAddr*], *TANK2*

    jmp *printTank*

*printTank3:*

    mov word [*startAddr*], *TANK3*

    jmp *printTank*

*printTank4:*

    mov word [*startAddr*], *TANK4*

    jmp *printTank*

*printTank:*

    mov byte [*colNum*], 55

    call *print*

坦克移动和边界判断

*kbW:*

    cmp byte [*flag*], 1

    jne *kbExit*

    cmp byte [*pauseFlag*], 1

    je *kbExit*

    mov byte [*dir*], 1

    mov word [*startAddr*], *CLEAR*

    call *print*

    cmp word [*startY*], 30

    ja *decY*

    mov byte [*flag*], 3

    mov word [*hp*], 0

    mov word [*sx*], 0

    mov word [*sy*], 0

    mov word [*colNum*], 0

    call *printSoildRec*

    call *closeBeep*

    jmp *kbExit*

*decY:*

    dec word [*startY*]

    jmp *kbExit*

*kbS:*

    cmp byte [*flag*], 1

    jne *kbExit*

    cmp byte [*pauseFlag*], 1

    je *kbExit*

    mov byte [*dir*], 2

    mov word [*startAddr*], *CLEAR*

    call *print*

    cmp word [*startY*], 154

    jb *incY*

    jmp *kbExit*

*incY:*

    inc word [*startY*]

    jmp *kbExit*

*kbA:*

    cmp byte [*flag*], 1

    jne *kbExit*

    cmp byte [*pauseFlag*], 1

    je *kbExit*

    mov byte [*dir*], 3

    mov word [*startAddr*], *CLEAR*

    call *print*

    cmp word [*startX*], 0

    ja *decX*

    jmp *kbExit*

*decX:*

    dec word [*startX*]

    jmp *kbExit*

*kbD:*

    cmp byte [*flag*], 1

    jne *kbExit*

    cmp byte [*pauseFlag*], 1

    je *kbExit*

    mov byte [*dir*], 4

    mov word [*startAddr*], *CLEAR*

    call *print*

    cmp word [*startX*], 304

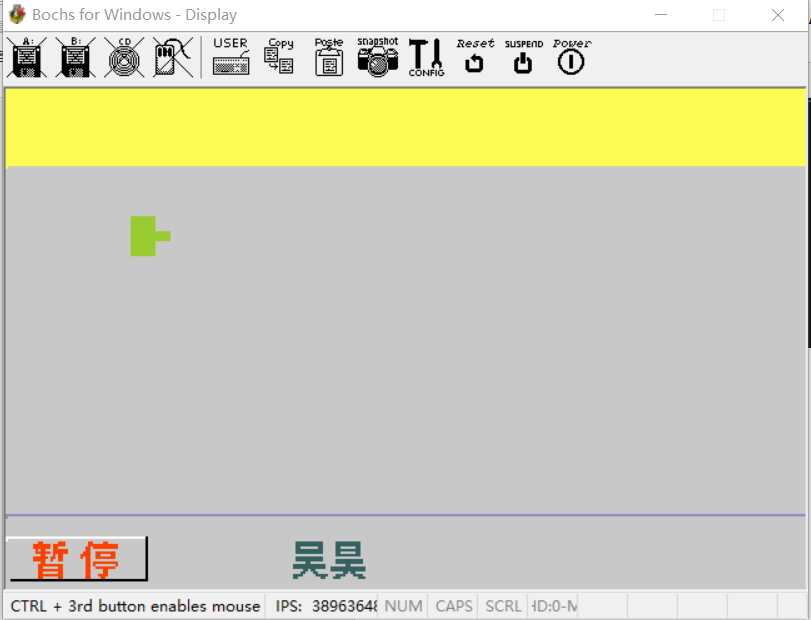
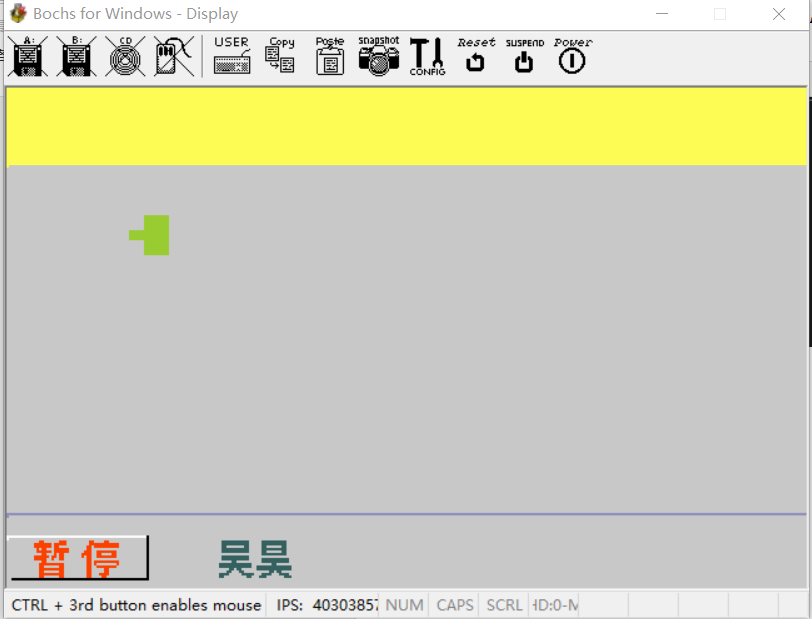
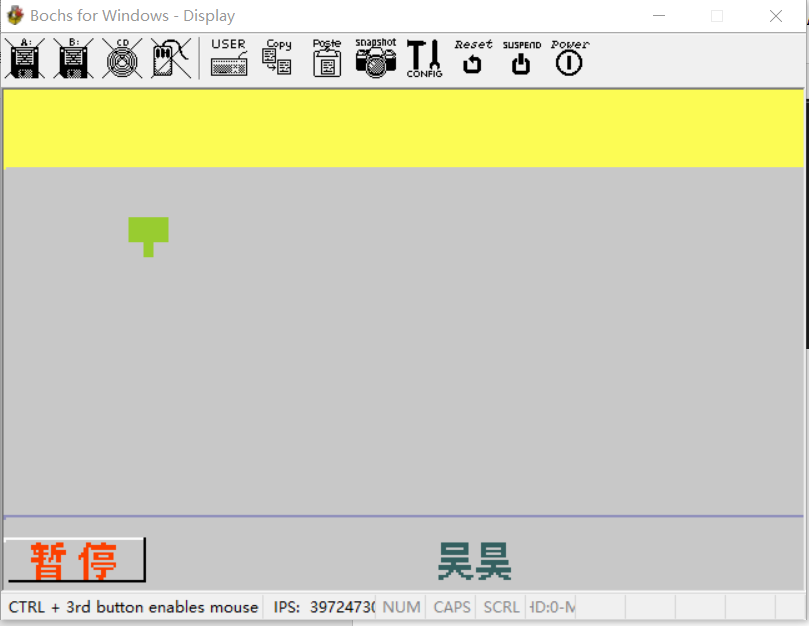
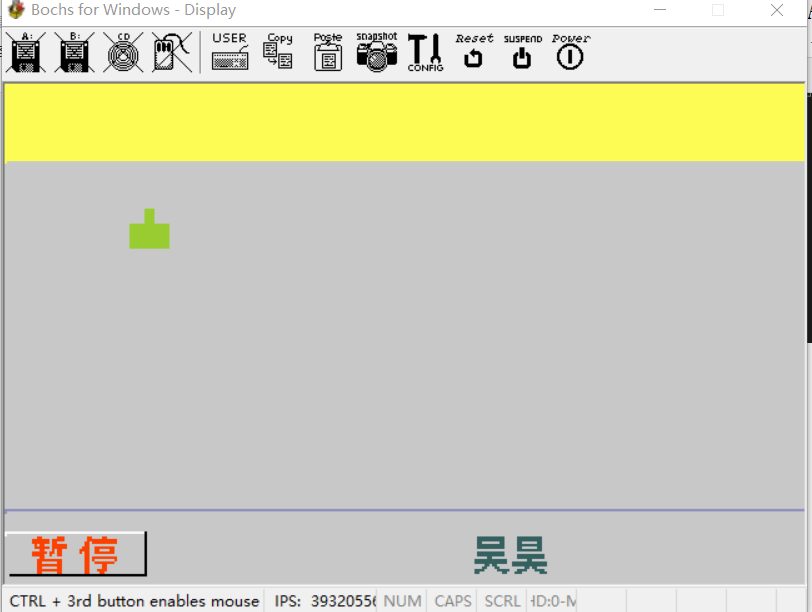
    jb *incX*

    jmp *kbExit*

*incX:*

    inc word [*startX*]

    jmp *kbExit*



1. 坦克子弹发射，坦克会在当前炮口的前端位置发射一个向前的炮弹

子弹发射位置的计算

   cmp byte [*bflag*], 0

    je *bulletControlExit*

    mov word [*bstartAddr*], *BULLETCLEAR*

    call *bprint*

    cmp byte [*bdir*], 1

    je *bdirw*

    cmp byte [*bdir*], 2

    je *bdirs*

    cmp byte [*bdir*], 3

    je *bdira*

    cmp byte [*bdir*], 4

    je *bdird*

    jmp *bulletControlExit*

*bdirw:*

    dec word [*bstartY*]

    cmp word [*bstartY*], 1

    jb *bclear*

    jmp *bulletprint*

*bdirs:*

    inc word [*bstartY*]

    cmp word [*bstartY*], 165

    ja *bclear*

    jmp *bulletprint*

*bdira:*

    dec word [*bstartX*]

    cmp word [*bstartX*], 1

    jb *bclear*

    jmp *bulletprint*

*bdird:*

    inc word [*bstartX*]

    cmp word [*bstartX*], 315

    ja *bclear*

    jmp *bulletprint*

*bulletprint:*

    mov word [*bstartAddr*], *BULLET*

    call *bprint*

    mov word [*sy*], 0

    mov bx, [*sy*]

    add bx, [*hight*]

    add bx, 2

    cmp bx, [*bstartY*]

    ja *bulletHit*

    jmp *bulletControlExit*

子弹发射的判断和方向判断

*kbullet:*

    cmp byte [*flag*], 1

    jne *kbExit*

    cmp byte [*pauseFlag*], 1

    je *kbExit*

    cmp byte [*bflag*], 1

    je *kbExit*

    mov byte [*bflag*], 1

    cmp byte [*dir*], 1

    je *kbulletW*

    cmp byte [*dir*], 2

    je *kbulletS*

    cmp byte [*dir*], 3

    je *kbulletA*

    cmp byte [*dir*], 4

    je *kbulletD*

    jmp *kbExit*

*kbulletW:*

    mov bx, [*startX*]

    add bx, 6

    mov [*bstartX*], bx

    mov bx, [*startY*]

    sub bx, 4

    mov [*bstartY*], bx

    mov bl, [*dir*]

    mov [*bdir*], bl

    mov word [*bstartAddr*], *BULLET*

    call *bprint*

    jmp *kbExit*

*kbulletS:*

    mov bx, [*startX*]

    add bx, 6

    mov [*bstartX*], bx

    mov bx, [*startY*]

    add bx, 16

    mov [*bstartY*], bx

    mov bl, [*dir*]

    mov [*bdir*], bl

    mov word [*bstartAddr*], *BULLET*

    call *bprint*

    jmp *kbExit*

*kbulletA:*

    mov bx, [*startX*]

    sub bx, 4

    mov [*bstartX*], bx

    mov bx, [*startY*]

    add bx, 6

    mov [*bstartY*], bx

    mov bl, [*dir*]

    mov [*bdir*], bl

    mov word [*bstartAddr*], *BULLET*

    call *bprint*

    jmp *kbExit*

*kbulletD:*

    mov bx, [*startX*]

    add bx, 16

    mov [*bstartX*], bx

    mov bx, [*startY*]

    add bx, 6

    mov [*bstartY*], bx

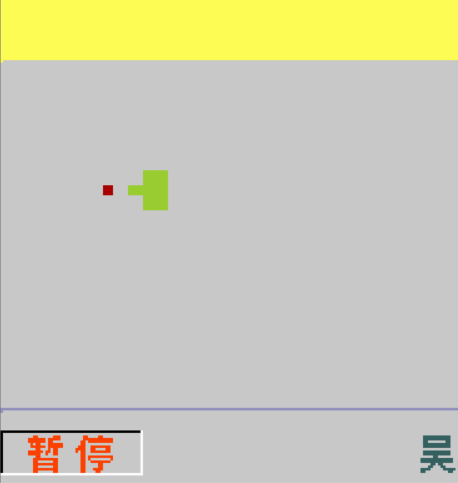
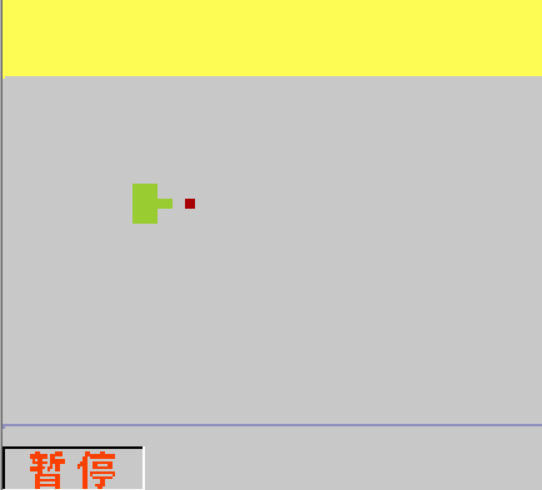
    mov bl, [*dir*]

    mov [*bdir*], bl

    mov word [*bstartAddr*], *BULLET*

    call *bprint*

    jmp *kbExit*



1. 顶部障碍物的绘制，子弹击中顶部障碍物后，顶部障碍物会血量减1，总共有5滴血，如果血量归零则游戏胜利

顶部障碍物的绘制

*printSoildRec:*

    push cx

    push ax

    mov cx,0

    mov ax,[*sx*]

    add ax,[*width*]

    mov [*ex*],ax

    mov ax,[*sy*]

    mov [*ey*],ax

    mov word [*fg*], 1

*psrBeg:*

    cmp cx,[*hight*]

    ja *psrExit*

    call *printLine*

    add word [*ey*],1

    add word [*sy*],1

    add cx,1

    jmp *psrBeg*

*psrExit:*

    mov cx, [*hight*]

    add cx, 1

    sub word [*ey*], cx

    sub word [*sy*], cx

    pop ax

    pop cx

    ret

子弹击打顶部障碍的判断

*bulletHit:*

    dec word [*hp*]

    mov bx, [*hp*]

    mov [*colNum*], bl

    mov word [*sx*], 0

    mov word [*sy*], 0

    call *printSoildRec*

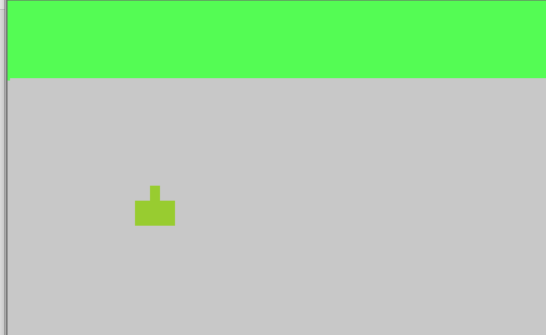
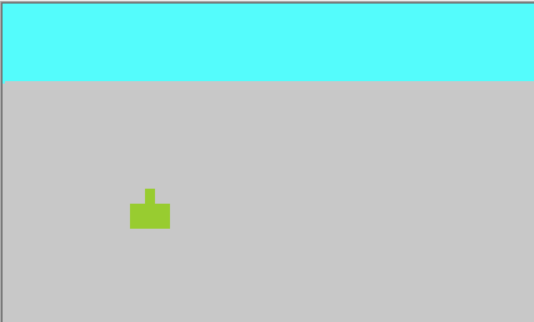
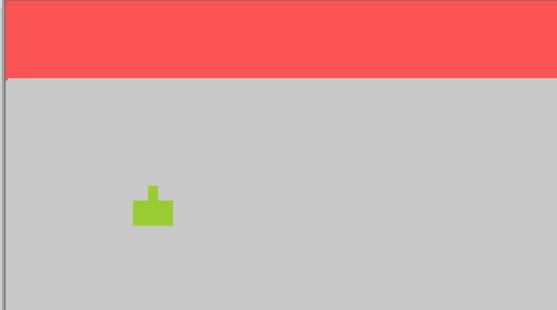
    mov word [*bstartAddr*], *BULLETCLEAR*

    call *bprint*

    cmp word [*hp*], 10

    jb *bulletWin*

    jmp *bclear*



1. 暂停功能，按下会有按下的效果，再次按会有弹起效果

暂停的实现，将时间片全部分给空任务

*mainPause:*

    mov word [ds:0x20], *emptyTask*

    mov word [ds:0x22], 0

    ;call sleep

    jmp *timerControl*

按下和松开效果的绘制

*kpause:*

    cmp byte [*flag*], 1

    jne *kbExit*

    cmp byte[*pauseFlag*], 0

    je *pauseTo1*

    mov byte [*pauseFlag*], 0

    mov byte [*colNum*], 17

    mov word [*sx*], 0

    mov word [*sy*], 179

    mov word [*ex*], 56

    mov word [*ey*], 179

    mov word [*fg*], 1

    call *printLine*

    mov word [*sx*], 0

    mov word [*sy*], 179

    mov word [*ex*], 0

    mov word [*ey*], 196

    mov word [*fg*], 320

    call *printLine*

    mov byte [*colNum*], 18

    mov word [*sx*], 0

    mov word [*sy*], 196

    mov word [*ex*], 56

    mov word [*ey*], 196

    mov word [*fg*], 1

    call *printLine*

    mov word [*sx*], 56

    mov word [*sy*], 179

    mov word [*ex*], 56

    mov word [*ey*], 196

    mov word [*fg*], 320

    call *printLine*

    jmp *kbExit*

*pauseTo1:*

    mov byte [*pauseFlag*], 1

    mov byte [*colNum*], 18

    mov word [*sx*], 0

    mov word [*sy*], 179

    mov word [*ex*], 56

    mov word [*ey*], 179

    mov word [*fg*], 1

    call *printLine*

    mov word [*sx*], 0

    mov word [*sy*], 179

    mov word [*ex*], 0

    mov word [*ey*], 196

    mov word [*fg*], 320

    call *printLine*

    mov byte [*colNum*], 17

    mov word [*sx*], 0

    mov word [*sy*], 196

    mov word [*ex*], 56

    mov word [*ey*], 196

    mov word [*fg*], 1

    call *printLine*

    mov word [*sx*], 56

    mov word [*sy*], 179

    mov word [*ex*], 56

    mov word [*ey*], 196

    mov word [*fg*], 320

    call *printLine*

    jmp *kbExit*

1. 底部吴的移动

*botControl:*

    push dx

    push ax

    mov word [*botStartAddr*], *BOTCLEAR*

    call *botPrint*

    cmp word [*botStartX*], 290

    jae *botChangeFlag1*

    cmp word [*botStartX*], 60

    jbe *botChangeFlag0*

    cmp byte [*botFlag*], 0

    je *botDirChange0*

    jmp *botDirChange1*

*botChangeFlag1:*

    mov byte [*botFlag*], 1

    jmp *botDirChange1*

*botChangeFlag0:*

    mov byte [*botFlag*], 0

    jmp *botDirChange0*

*botDirChange0:*

    inc word [*botStartX*]

    mov word [*botStartAddr*], *WU*

    call *botPrint*

    jmp *botExit*

*botDirChange1:*

    dec word [*botStartX*]

    mov word [*botStartAddr*],*WU*

    call *botPrint*

    jmp *botExit*

*botExit:*

    mov al, 0x20

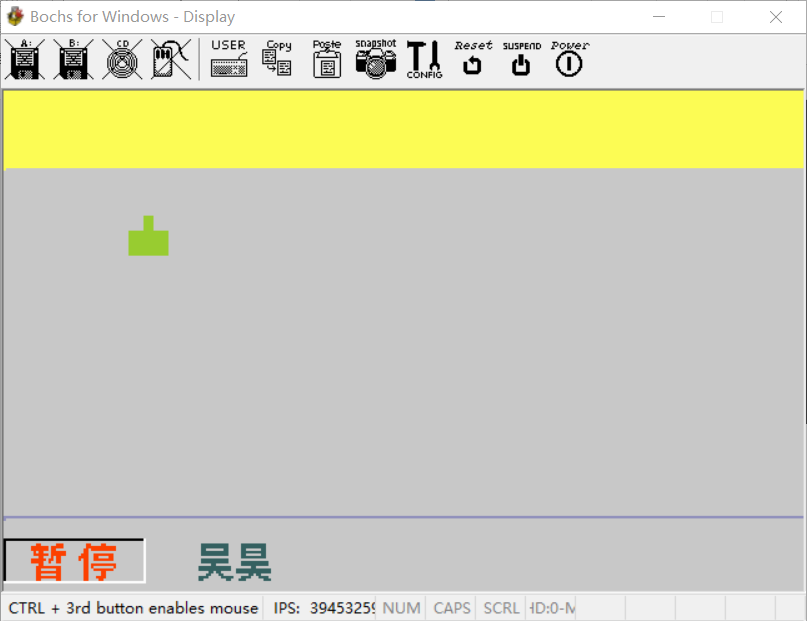
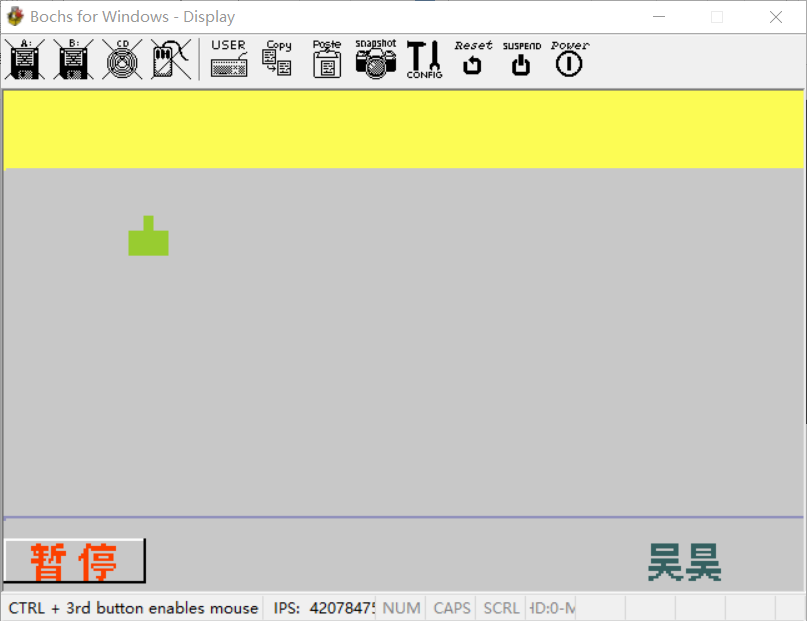
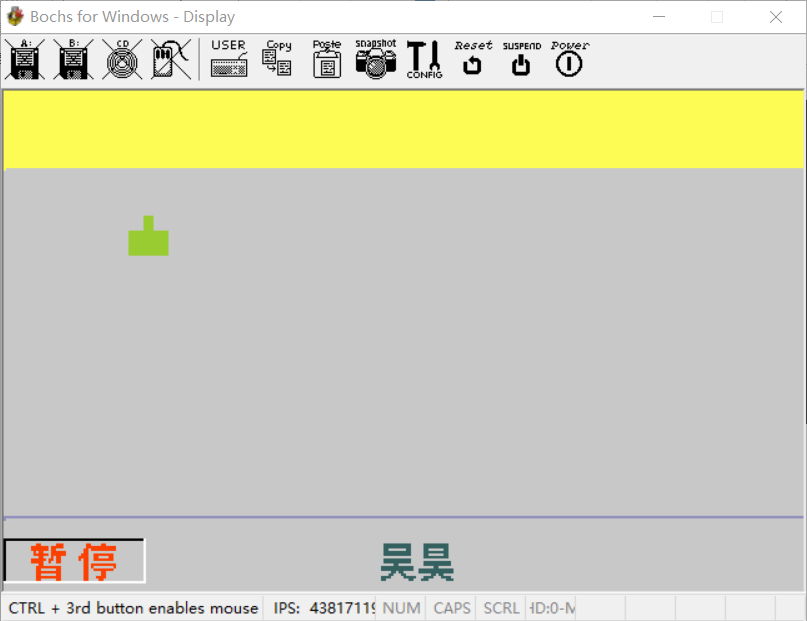
    mov dx, 0x20

    out dx, al

    pop ax

    pop dx

    iret



1. 背景音乐

*playSound:*

    push ax

    push dx

    push bx

    inc word [*soundCount*]

*soundBeg:*

    cmp word [*soundCount*], 11

    ja *soundAdd*

    jmp *soundExit*

*soundAdd:*

    call *closeBeep*

    mov word[*soundCount*], 0

    add word [*soundNow*], 2

    mov bx, [*soundNum*]

    cmp [*soundNow*], bx

    jae *soundClearNow*

    jmp *soundAddNow*

*soundClearNow:*

    mov word [*soundNow*], 0

*soundAddNow:*

    mov bx, [*soundNow*]

    mov bx, [*soundList* + bx]

    mov [*hz*], bx

    call *setHZ*

    call *openBeep*

*soundExit:*

    mov al, 0x20

    mov dx, 0x20

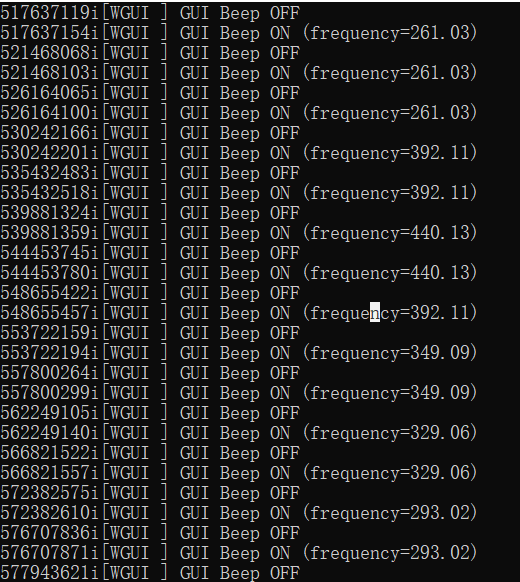
    out dx, al

    pop bx

    pop ax

    pop dx

    iret



1. 失败页面的绘制

*gameLose:*

    cli

    mov byte [*colNum*], 0

    mov byte [*colR*], 0

    mov byte [*colG*], 0

    mov byte [*colB*], 0

    call *setColor*

    mov byte [*colNum*], 10

    mov byte [*colR*], 63

    mov byte [*colG*], 0

    mov byte [*colB*], 0

    call *setColor*

    mov word [*startX*], 100

    mov word [*startY*], 80

    mov word [*startAddr*], *YOU*

    call *print*

    mov word [*startX*], 120

    mov word [*startY*], 80

    mov word [*startAddr*], *XI*

    call *print*

    mov word [*startX*], 140

    mov word [*startY*], 80

    mov word [*startAddr*], *SHI2*

    call *print*

    mov word [*startX*], 160

    mov word [*startY*], 80

    mov word [*startAddr*], *BAI*

    call *print*

    mov word [*startAddr*], *CLEAR*

    mov word [*startX*], 10

    mov word [*startY*], 180

    call *print*

    mov word [*startX*], 30

    mov word [*startY*], 180

    call *print*

    mov word [*botStartAddr*], *BOTCLEAR*

    call *botPrint*

    mov byte [*colNum*], 18

    mov word [*sx*], 0

    mov word [*sy*], 170

    mov word [*ex*], 320

    mov word [*ey*], 170

    mov word [*fg*], 1

    call *printLine*

    mov word [*sx*], 0

    mov word [*sy*], 179

    mov word [*ex*], 56

    mov word [*ey*], 179

    mov word [*fg*], 1

    call *printLine*

    mov word [*sx*], 0

    mov word [*sy*], 179

    mov word [*ex*], 0

    mov word [*ey*], 196

    mov word [*fg*], 320

    call *printLine*

    mov word [*sx*], 0

    mov word [*sy*], 196

    mov word [*ex*], 56

    mov word [*ey*], 196

    mov word [*fg*], 1

    call *printLine*

    mov word [*sx*], 56

    mov word [*sy*], 179

    mov word [*ex*], 56

    mov word [*ey*], 196

    mov word [*fg*], 320

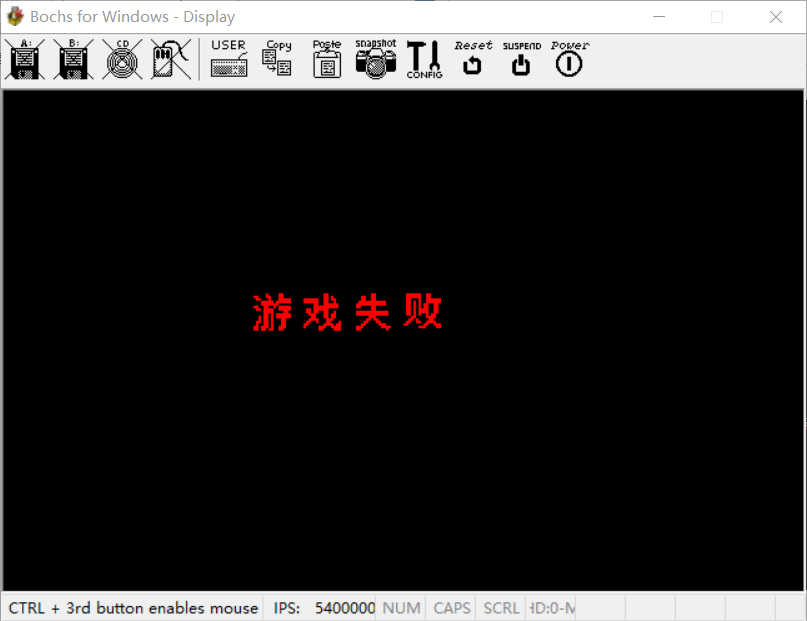
    call *printLine*

    mov word [ds:0x20], *emptyTask*

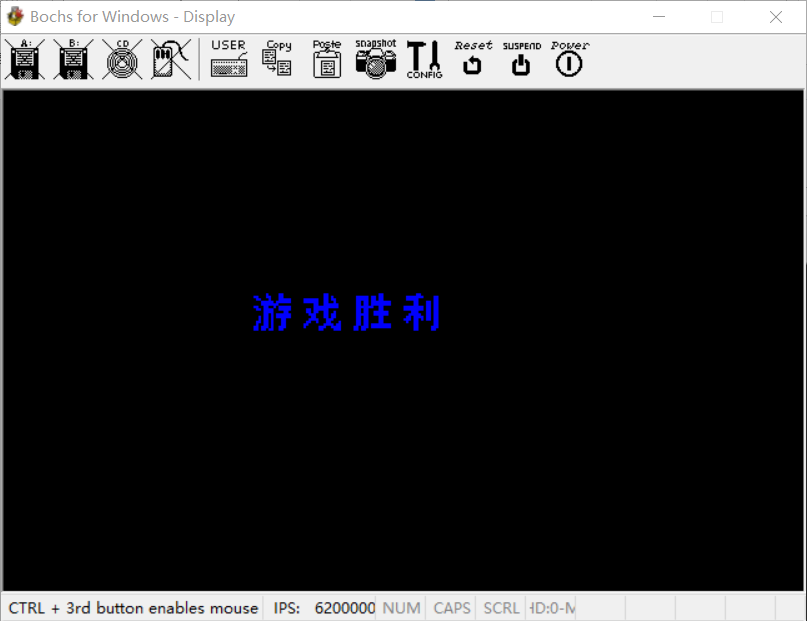
    mov word [ds:0x22], 0

    call *sleep*

    jmp *mainControl*



1. 游戏胜利的绘制



## 心得体会

本次课设完成的还是非常匆忙的，不过想要实现的功能都基本实现了。这次的多任务控制足足有五个任务需要轮番控制，我采用的是划分时间片的方式，就是说将定时器任务的地址给定时器之后，使用延时函数进行一段时间的延时，这段延时的时间就是这个任务的运行时间，时间到了就修改定时器的地址，将其改成下个任务的地址，但是在实际的使用之中每个任务的时间划分也不是均分的，所以事后需要对时间要求比较高的背景音乐进行仔细的调整。

本次任务中比较麻烦的部分是坦克有四个方向，我们发射子弹需要从不同的位置发射出去，每一个位置都需要经过计算得到，虽然这个部分比较细微，但是想要得到比较好的效果还是要花一些功夫的。

还有一些比较简单但是比较亮眼巧妙的地方，这个暂停按钮的按下和抬起的效果实际上是通过上左和下右的线条黑白切换来实现的，这个比较容易，但是效果很好。

汉字的绘制，本程序中有很多汉字需要使用，这就关系到汉字点阵，这部分也是我唯一需要联网的部分，我的方法是先上网搜索汉字点阵，然后得到一个由方框和空格组成的点阵，这个点阵我再使用vscode自带的批量修改功能改成01矩阵。

这次课设可以说是将本学期学会的汇编知识都用了上去，也只用到了课内学到的知识，可以说是一个很好的对自己的检测了。