

**本科生课程设计**

**汇编语言与接口技术组队实验报告——PACMAN**

**Team experiment report on assembly language and interface technology——PACMAN**

|  |  |
| --- | --- |
| 学 院： | 计算机学院 |
| 专 业： | 计算机科学与技术 |
| 学生姓名： | 张驰、魏慧聪、曹哲瑀、王浚哲 |
| 学 号： | 1120191600、1120191866、1120191407、1120191197 |
| 指导教师： | 李元章 |

2022 年 6 月 22 日

**汇编语言与接口技术组队实验报告——PACMAN**

摘　要

本次实验过程中使用汇编语言，以visual studio 2019作为开发的IDE，进行了组队编程。本队伍熟练使用汇编语言，实现了游戏PACMAN，并在初代PACMAN游戏的追逐模式的基础上将分散模式融入道具，并加入了一些道具功能。

在本次组队实验的过程中，我们队伍中的每个人积极参与，从中掌握了汇编语言开发技术，进一步加深了对汇编语言的理解。

**关键词：汇编语言与接口技术；游戏编程；PACMAN**

**Team experiment report on assembly language and interface technology**

Abstract

In this experiment, assembly language was used and visual studio 2019 was used as the IDE for team programming. The team skillfully used assembly language to realize the game Pacman, and integrated the decentralized mode into props on the basis of the chase mode of the early generation Pacman games, and added some props functions.

In the process of this team experiment, everyone in our team actively participated, mastered the assembly language development technology, and further deepened the understanding of assembly language.

**Key Words: Assembly language and interface technology; Game programming; PACMAN game**

目　录

[摘　要 I](#_Toc106921367)

[Abstract II](#_Toc106921368)

[第1章 实验目的 1](#_Toc106921369)

[第2章 实验环境 1](#_Toc106921370)

[第3章 PACMAN游戏与程序简介 1](#_Toc106921371)

[3.1 PACMAN游戏简介 1](#_Toc106921372)

[3.2 游戏规则 1](#_Toc106921373)

[3.3 程序运行方式 3](#_Toc106921374)

[3.4 组队实验分工介绍： 3](#_Toc106921375)

[第4章 游戏的分模块介绍 4](#_Toc106921376)

[4.1 模块1——界面设计与显示方式 4](#_Toc106921377)

[4.2 模块2——控制逻辑与碰撞检测 4](#_Toc106921378)

[4.3 幽灵追逐逻辑模块 6](#_Toc106921379)

[4.4 道具设计与道具效果实现模块 7](#_Toc106921380)

[第5章 实验心得体会 8](#_Toc106921381)

# 第1章 实验目的

本次实验通过使用ASM汇编语言编写PACMAN游戏，并在基础PACMAN游戏中加入一些创新点，从而掌握Windows系统汇编程序的基本结构，掌握基本的汇编指令。通过本次实验，使队伍更加熟练进行汇编程序的编写和调试，学会调用C库函数，提高汇编语言编程能力，加强对汇编指令的理解。

# 第2章 实验环境

本次实验在Windows10系统下进行，使用MASM,以Visual Studio 2019作为开发的IDE。

# 第3章 PACMAN游戏与程序简介

3.1 PACMAN游戏简介

吃豆人是电子游戏历史上的经典街机游戏，由Namco公司的岩谷彻设计并由Midway Games在1980年发行。PacMan被认为是80年代最经典的街机游戏之一。Pacman创造了第一个活生生的游戏角色，也是第一个引入了AI的游戏。2020年是Pacman诞生的40周年，这一年有诸多关于Pacman的有趣故事。到现在，依旧可以在各种电影中看到Pacman的身影，Pacman的形象已经成为了一种大众文化符号。游戏中有四个幽灵，分别是Blinky、Pinky、Inky和Clyde。最初的吃豆人游戏中有三个模式，分别是追逐模式，分散模式和惊吓模式。

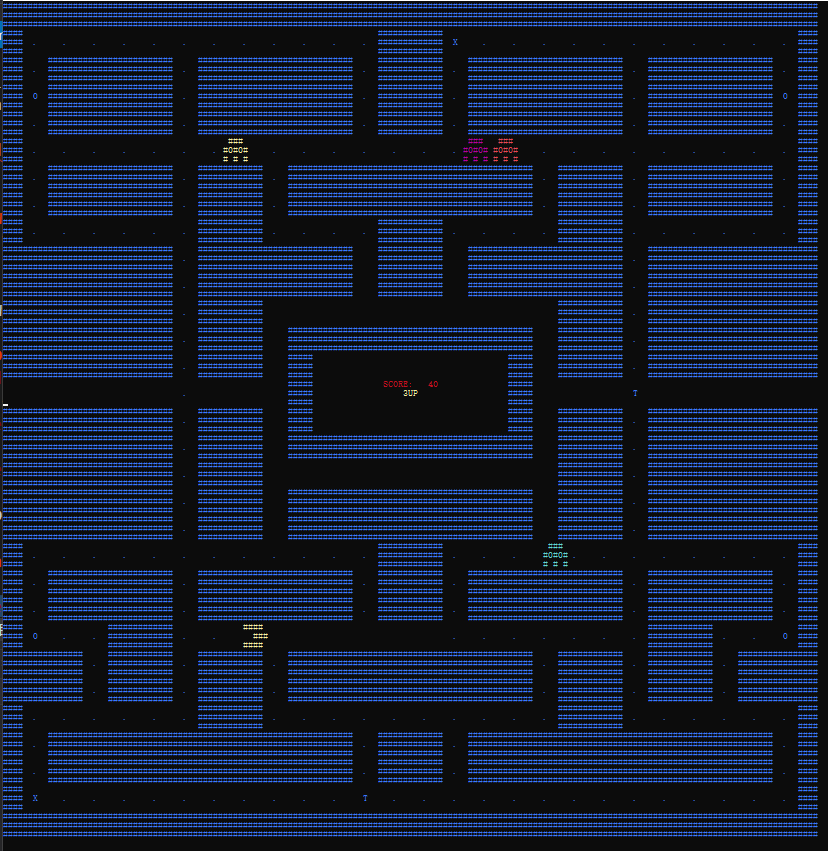
本次组队实验编写的游戏采用了初代吃豆人的经典地图，在实现了追逐模式的基础上将分散模式融入道具，并加入了其他功能道具。

3.2 游戏规则

游戏界面如下图所示：



图 1：初始进入页面



图表 2：游戏页面

使用W、A、S、D控制吃豆人吃掉游戏迷宫中所有的豆子，且在路径中尽可能避免被幽灵吃掉。

在所有生命消耗完毕前，吃掉场上的所有豆子，即获得胜利。生命值消耗殆尽，游戏结束。

游戏中包含以下道具：

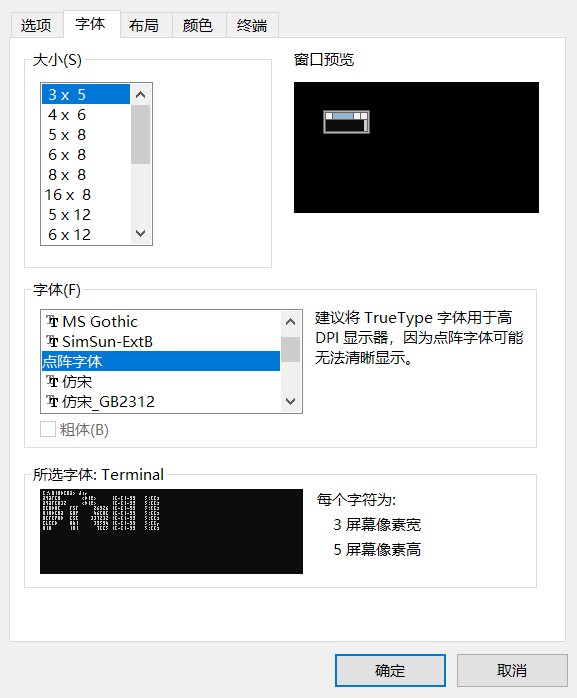
道具O：该道具干扰鬼的逻辑，在一定时间内幽灵会尽可能避免追逐PACMAN

道具X：该道具可以为PACMAN增加一点生命

道具T：该道具可以让场上的所有鬼暂停移动

3.3 程序运行方式

该游戏运行在cmd窗口中，为了游戏大小窗口合适，需要设置窗口属性的字体为3\*5的点阵字体。



3.4 组队实验分工介绍：

本次实验过程中包含界面设计、控制逻辑设计、碰撞检测、四只追逐PACMAN的幽灵追逐方式的设计、道具设计等多个模块。

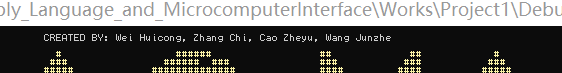
在本次组队设计中，魏慧聪同学完成了游戏中幽灵追逐PACMAN的方案构思与实现、道具设计、地图和人物设计模块。曹哲瑀同学完成了游戏中幽灵与PACMAN的距离计算、幽灵移动方向计算、幽灵移动实现模块。张驰同学完成了游戏中控制逻辑以及碰撞检测、游戏过程的地图改动、道具设计模块。王浚哲同学完成了游戏中界面设计显示、帧数刷新、分数计算模块。

# 第4章 游戏的分模块介绍

4.1 模块1——界面设计与显示方式

该部分负责初始显示界面（图1）和游戏页面背景的显示（图2蓝色部分）。

图一主要由两部分组成，第一部分显示作者信息：



第二部分显示主要图像内容。

图1输出函数如下：

1. splashScreen PROC
2. Call Crlf
3. mov edx, OFFSET createdBy  ;输出作者行
4. Call WriteString
5. Call Crlf
6. ;Call Crlf
7. Call pacManWord  ;主要输出函数
8. Call Crlf
9. ret
10. splashScreen ENDP

进入main函数先调用splashScreen函数，splashScreen函数3、4行输出作者行，作者信息已先行写入define.inc中，以字节的形式保存：

1. createdBy BYTE "                                CREATED BY: Wei Huicong, Zhang Chi, Cao Zheyu, Wang Junzhe", 0ah, 0

第二部分的输出由pacManWord函数完成

1. pacManWord PROC       ;生成初始界面
2. Call yellowText
3. mov ecx, 13
4. mov edx, OFFSET pac-138
5. printPacLoop:
6. add edx, 139
7. Call writeStrWithSP ;从pac第一行开始打印，每行打印完接换行
8. loop printPacLoop
9. mov ecx, 17
10. print14to30: ;打印第一部分
11. mov esi, -1
12. add edx, 139
13. begin1:
14. inc esi  ;esi + 1
15. .if esi == 139
16. dec ecx ;ecx - 1
17. jmp loop1
18. .endif
19. .if esi < 52   ;从这里开始打印
20. mov eax, 14
21. call SetTextColor
22. mov al, byte ptr[edx+esi] ;相对基址变址寻址
23. call WriteChar
24. .elseif byte ptr[edx+esi] == '0' ;edx+esi字节中的值为 0， 0打印为白色
25. mov eax, 15
26. call SetTextColor
27. mov al, byte ptr[edx+esi]
28. call WriteChar
29. .elseif byte ptr[edx+esi] == '1'
30. mov eax, 0 ;1打印为黑色
31. call SetTextColor
32. mov al, byte ptr[edx+esi]
33. call WriteChar
34. .elseif esi >= 52
35. .if esi < 100
36. mov eax, 4
37. call SetTextColor
38. mov al, byte ptr[edx + esi]
39. call WriteChar
40. .else
41. mov eax, 3
42. call SetTextColor
43. mov al, byte ptr[edx + esi]
44. call WriteChar
45. .endif
46. .endif
47. jmp begin1
48. loop1:
49. ……
50. begin2:
51. ……
52. jmp begin2
53. loop2:
54. ……
55. loop3:
56. add edx, 139
57. Call writeStrWithSP
58. loop loop3
59. call Crlf
60. pacManWord ENDP

其中，SetTextColor函数读取当前eax寄存器中数值，每种数值对应一种颜色。

后面几个循环操作针对当前行不同的构成进行了调整，但读取字符再区分颜色输出字符的本质相同。

图1的显示是由字符打印实现的，原始字符已经存入define.inc中的pac中：

panManWord函数分行对pac进行打印，区分每个字符以不同的颜色，从而最终实现图1的结果。

图 3：pac展示

按图1要求输入任意字符后，利用Clrscr函数清空屏幕

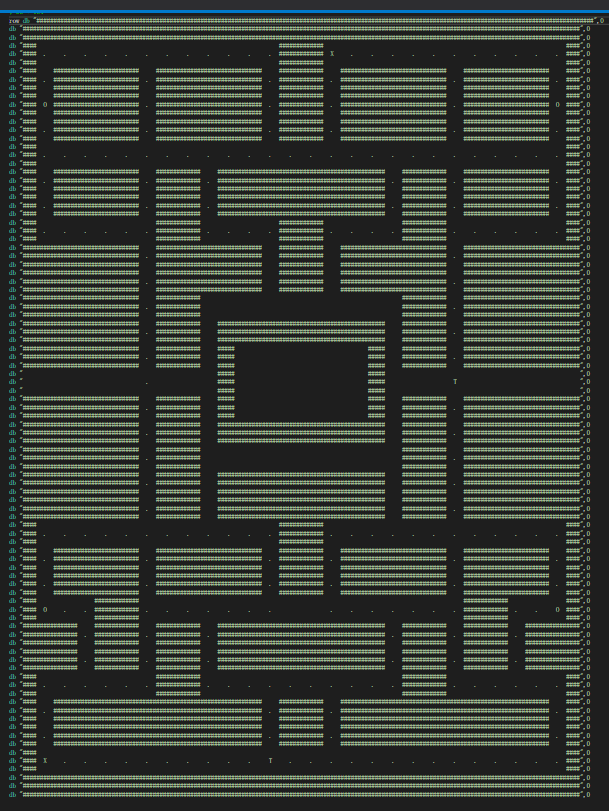
图2输出的内容一直保持直达达成失败或胜利条件，此间，每次每次帧率刷新更新一次吃豆人和四个幽灵的位置，但是地图保持不变，所用函数为drawStringBoard:

1. drawStringBoard proc
2. Call valuesToDrawBoard
3. mov ecx, 93
4. printMap:
5. add edx, 164
6. Call writeStrWithSP
7. loop printMap
8. doneWithMap:
9. ret
10. drawStringBoard endp

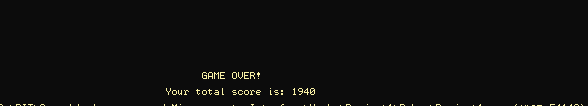
valueToDrawBoard函数将输出位置改至左上角第一个字符，并将字体颜色设置为蓝色，最后把地图row的初地址移至edx中准备开始打印row中所存储地图：

1. valuesToDrawBoard PROC
2. mov dx, 0
3. Call Gotoxy
4. mov eax, 9
5. Call SetTextColor
6. mov edx, offset row-164
7. ret
8. valuesToDrawBoard ENDP

row存储地图的方式和上文所提的pac相同：



writeStrWithSP为简单的WriteString函数和一次换行。

当满足胜利或失败条件后，延迟200ms后，清空屏幕，输出战果：

youWon函数和gameover函数比较简单且结构完全相同，区别仅有输出内容不同：

1. gameOver proc
2. Call Clrscr
3. mov dh, 12
4. mov dl, 34
5. Call Gotoxy
6. mov edx, offset thatsAllSheWrote
7. Call WriteString
8. mov dh, 14
9. mov dl, 28
10. Call Gotoxy
11. mov edx, offset result
12. Call WriteString
13. mov eax, totalScore
14. Call WriteDec
15. Call Crlf
16. mov eax, 1000
17. call Delay
18. ret
19. gameOver endp

4.2 模块2——控制逻辑与碰撞检测

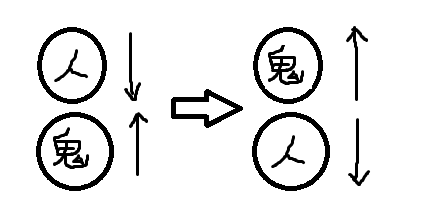
控制逻辑的实现比较简单，即按下WASD按钮，控制PACMAN向上下左右四个方向移动。每次移动后，将PACMAN的x位置和y位置存放进入到BH和BL寄存器中。根据WASD的按键，判断是否遇到了墙体，如果遇到的不是墙体，计算下一次的PACMAN位置，进行第一轮的碰撞检测。比较特殊的一点在于，在移动的过程中，要注意地图上特殊位置的检测（传送点位置的传送检测），要对位置进行调整。同时，在控制逻辑模块中，也记录了当PACMAN遭遇到相应的道具时，设置相应的增益时间。

相应的代码如下所示：(其中省区了一些重复的代码)

1. pacManControls PROC
2. Call ReadKey
3. mov BH, pacx
4. mov BL, pacy
5. ;向下走，记方向为 3
6. .IF AL == 's' || AL == 'S'
7. inc BL
8. Call charAtXY
9. mov pacdir, 3
10. mov pacChar, offset pacmanEatShape+36
11. .IF AL != '#'
13. inc pacy
14. call ghostsVsPacman
15. .if al == 'O'
16. add bufftime, 50
17. .endif
18. .if al == 'T'
19. add pausetime, 60
20. .endif
21. .if al == 'X'
22. inc totalLives
23. .endif
24. .ENDIF
25. ;向上走，记方向为 1
26. .ELSEIF AL == 'w' || AL == 'W'
27. ...
28. ;向右走，计方向为 2
29. .ELSEIF AL == 'd' || AL == 'D'
30. ...
31. ;向左走，记方向为 4
32. .ELSEIF AL == 'a' || AL == 'A'
33. ...
34. .ENDIF
35. cmp pacx, 0
36. je teleportPacMan1
37. cmp pacx, 162
38. je teleportPacMan2
39. jmp nothingHere

碰撞检测模块ghostsVsPacman需要调用两次，分别是第一次碰撞检测和第二次碰撞检测。其中第一次碰撞检测的调用位于控制逻辑模块之中。

碰撞检测的过程需要一次比较pacman的中心点位置与四只Ghosts的x位置与y位置是否相等，如果相等，则判定为碰撞成功，死亡，减少一次生命并重置地图。需要进行两次碰撞的原因在于需要避免人与鬼对冲而过导致未检测到的情况。第一次碰撞检测的目的是检测对冲而过导致人与鬼进行碰撞的情况，第二轮碰撞检测则是检测正常的相遇情况。第一次碰撞检测时，只有人朝目标方向进行了移动，是为了防止人与鬼对冲的情况；第二次碰撞检测时，人与鬼均朝目标方向进行了移动，是正常的检测逻辑。



碰撞检测的部分代码如下：（删去了重复部分）

1. ghostsVsPacman proc uses eax
2. mov ah, ghos1x
3. cmp pacx, ah
4. je pacmanY1
5. pacmanX2:
6. ...
7. pacmanX3:
8. ...
9. pacmanX4:
10. ...
11. pacmanY1:
12. mov ah, ghos1y
13. cmp pacy, ah
14. je DeadMan
15. jmp pacmanX2
16. pacmanY2:
17. ...
18. pacmanY3:
19. ...
20. pacmanY4:
21. ...
22. DeadMan:
23. Call caughtInTheAct
24. noDice:
25. ret
26. ghostsVsPacman endp

4.3 幽灵追逐逻辑模块

模块

4.4 道具设计与道具效果实现模块

如上游戏简介中所示，本游戏中设置了三种道具：

道具O：该道具干扰鬼的逻辑，在一定时间内幽灵会尽可能避免追逐PACMAN

道具X：该道具可以为PACMAN增加一点生命

道具T：该道具可以让场上的所有鬼暂停移动

道具的设计体现在地图之中，地图中的符号’·’表示吃豆人要吃到的豆子，而地图上的符号’O’ ,’X’ ,’T’分别表示不同的符号。

其中，O道具和T道具均为增益道具，具有相应的时长，在吃到道具后时长设置为固定值，之后随着帧的变化时长逐渐减少。当减少到0时，道具的效果结束。X道具直接使当前的生命值增加一点即可。

其中，判定吃到道具的代码已经在控制逻辑模块中进行了展示。下面展示三种道具的效果体现：

1. ;鬼的移动函数
2. moveGhosts proc
3. .if pausetime > 0
4. jmp movedone
5. .endif
6. ...
7. movedone:
8. ret
9. moveGhosts endp
10. .if bufftime > 0
11. dec bufftime
12. .endif
13. .if pausetime > 0
14. dec pausetime
15. .endif
16. .if bufftime > 0
17. ;决定下一次向哪个方向移动
18. Call directGhostsReverse
19. .else
20. call directGhosts
21. .endif

其中，增加生命道具的效果体现已经在控制逻辑模块进行了展示。

# 第5章 实验心得体会

在本次实验中，