

**本科生毕业设计(论文)**

**汇编语言与接口技术上机实验报告**

**Experimental Report of Assembly Language and Interface Technology Course of Beijing Institute of Technology**

|  |  |
| --- | --- |
| 学 院： | 徐特立学院 |
| 专 业： | 计算机科学与技术 |
| 学生姓名： | 刘丹阳，奥登格日乐， |
|  | 林致锋，陈照欣 |
| 指导教师： | 张华平 |

2019 年 6 月 27 日

**原创性声明**

本人郑重声明：所呈交的毕业设计（论文），是本人在指导老师的指导下独立进行研究所取得的成果。除文中已经注明引用的内容外，本文不包含任何其他个人或集体已经发表或撰写过的研究成果。对本文的研究做出重要贡献的个人和集体，均已在文中以明确方式标明。

特此申明。

本人签名： 日 期： 2022 年 6 月 25 日

**关于使用授权的声明**

本人完全了解北京理工大学有关保管、使用毕业设计（论文）的规定，其中包括：①学校有权保管、并向有关部门送交本毕业设计（论文）的原件与复印件；②学校可以采用影印、缩印或其它复制手段复制并保存本毕业设计（论文）；③学校可允许本毕业设计（论文）被查阅或借阅；④学校可以学术交流为目的,复制赠送和交换本毕业设计（论文）；⑤学校可以公布本毕业设计（论文）的全部或部分内容。

本人签名： 日 期： 2022 年 6 月 25 日

指导老师签名： 日 期： 年 月 日

**北京理工大学汇编语言与接口技术课程实验报告**

摘　要

汇编语言是用于电子计算机，微处理器等其他编程器件的低级语言，也被称为是符号语言。在汇编语言中，实际的机器指令被助记符替换，用地址符号或其他标号来代替操作数的地址，在不同的设备中对应着不同的机器语言指令集，通过汇编过程转化成机器指令。

尽管现在汇编语言已不像其他大多数的程序设计语言一样被广泛用于程序设计，在今天的实际应用中，它通常被应用在底层硬件操作和高要求的程序优化的场合。驱动程序、嵌入式操作系统和实时运行程序中都会需要汇编语言。

本报告为2021-2022年第二学期北京理工大学计算机学院汇编语言与接口技术课程的实验报告。本报告完成了原创游戏《抢地盘》，对程序设计的过程进行了具体的说明和展示，对编写的代码进行测试以及对实验运行结果的评估。

**关键词 汇编语言，反汇编，课程心得**

**Experimental Report of Assembly Language and Interface Technology Course of Beijinginstitute of Technology**

Abstract

Assembly language is a low-level language used for programming devices such as computers and microprocessors, and is also called a symbolic language. In assembly language, the actual machine instructions are replaced by mnemonics, and the address symbols or other labels are used to replace the addresses of the operands. Different equipment corresponds to different machine language instruction sets, which are converted into machine instructions through the assembly process.

Although assembly language has not been widely used in programming like most other programming languages, in today's practical applications, it is usually used in low-level hardware operations and high-demand program optimization occasions. Assembly language is needed in drivers, embedded operating systems, and real-time running programs.

This report is an experimental report of the assembly language and interface technology course of the School of Computer Science and Technology, Beijing Institute of Technology in the second semester of 2020-2021. This report completes This report completes the original game "Site Grab", and specifically explains and demonstrates the process of program design, tests the written code, and evaluates the results of the experiment.

**Key Words: The multiplication of large Numbers；Assembly language**

目　录

[摘　要 I](#_Toc107044360)

[Abstract II](#_Toc107044361)

[1.1 项目描述 1](#_Toc107044362)

[1.2 算法实现 1](#_Toc107044363)

[1.2.1 界面绘制 1](#_Toc107044364)

[1.2.2 棋子移动——判断能否移动 2](#_Toc107044365)

[1.2.3 棋子移动——移动后“策反” 4](#_Toc107044366)

[1.2.4 当前棋局胜负判断 5](#_Toc107044367)

[1.2.5单双人模式选择 8](#_Toc107044368)

[2.运行效果截图 11](#_Toc107044369)

[结　论 13](#_Toc107044370)

[参考文献 14](#_Toc107044371)

[附　录 15](#_Toc107044372)

[致　谢 16](#_Toc107044373)

1.1 项目描述

该游戏是一款1V1的益智类游戏。游戏中设置了单人模式和双人模式，单人模式时由电脑与玩家PK，双人模式则可以有两个玩家PK。

游戏规定棋子在固定范围内（以棋子原位置为中心的斜正方形）移动到空位置：假设我方棋子从X移动到Y处，当Y与X在水平线、垂直线和斜对角线上相邻时，X处仍会保留我方棋子；当Y与X间隔一个格子时，X处则不会保留。移动后，Y处上下左右的敌方棋子都会被“策反”为我方棋子。

直到敌方棋子无法移动，则判为失败：

——敌方棋子全部被“策反”；

——敌方棋子无路可走（所有可移动的位置都被我方占领）；

1.2 算法实现

1.2.1 界面绘制

首先我们搭建了一个Win32界面编程的框架，绘制了游戏界面。用一个矩阵map代表地图，1代表A棋子，2代表B棋子，0代表无棋子的可移动位置。

map db 0,0,2,2,2,0

db 2,0,2,2,2,0

db 0,2,0,0,0,0

db 0,0,0,0,0,0

db 0,0,1,1,1,0

db 1,1,1,1,1,0

根据map矩阵绘制出游戏界面。

mov esi,0

.while esi<sz

mov edi,0

.while edi<sz

mov eax,esi

imul eax,40

mov ebx,edi

imul ebx,40

mov ecx,esi

imul ecx,sz

add ecx,edi

mov dl,map[ecx]

.if dl==0

invoke CreateWindowEx,NULL,offset button,offset bu0,WS\_CHILD \\

or WS\_VISIBLE,ebx,eax,40,40,hWnd,ecx,hInstance,NULL

.elseif dl==1

invoke CreateWindowEx,NULL,offset button,offset bu1,WS\_CHILD \\

or WS\_VISIBLE,ebx,eax,40,40,hWnd,ecx,hInstance,NULL

.elseif dl==2

invoke CreateWindowEx,NULL,offset button,offset bu2,WS\_CHILD \\

or WS\_VISIBLE,ebx,eax,40,40,hWnd,ecx,hInstance,NULL

.endif

inc edi

.endw

inc esi

.endw

1.2.2 棋子移动——判断能否移动

根据规则，棋子可以移动的范围是以其为中心的斜正方形，也就是需要判断目标位置是否在此写正方形内，这一步我们通过求得棋子移动前后的位置差的绝对值来进行判断。

map的大小设为sz，按照游戏规则只有移动范围为水平线、垂直线和斜对角线上相邻位置（原位置棋子仍然保留，左图）和水平线、垂直线上间隔一个格子的位置（原位置棋子不保留，右图）。

mov edx,pre;减法操作

.if eax>=edx

sub eax,edx

.else

sub edx,eax

mov eax,edx

.endif

.if eax==1||eax==sz||eax==sz-1||eax==sz+1

mov eax,wParam

mov next,eax

xor eax,eax

mov flag,eax

invoke SetDlgItemText,hWnd,next,addr bu1 ;返回值为bool，存入eax

mov ebx,offset map

mov edx,1

add ebx,next

mov [ebx],dl

.elseif eax==2||eax==sz+sz

ov eax,wParam

mov next,eax

xor eax,eax

mov flag,eax

invoke SetDlgItemText,hWnd,pre,addr bu0

invoke SetDlgItemText,hWnd,next,addr bu1

mov ebx,offset map

mov edx,0

add ebx,pre

mov [ebx],dl ;mov map[pre],edx不可行

mov ebx,offset map

mov edx,1

add ebx,next

mov [ebx],dl

1.2.3 棋子移动——移动后“策反”

当棋子移动后，根据规则，如果移动后位置的上下左右处有敌方棋子，则它们会被“策反”为我方棋子，这一步通过分别对四个方位进行判断，符合上述规则的，更新map矩阵和游戏界面的显示。

;上

mov eax,next

add eax,sz

.if eax<sz\*sz && map[eax]==2

push eax

invoke SetDlgItemText,hWnd,eax,addr bu1

pop eax

mov [ebx+eax],dl

.endif

;下

mov eax,next

.if eax>sz

sub eax,sz

.if eax>=0 && map[eax]==2

push eax

invoke SetDlgItemText,hWnd,eax,addr bu1

pop eax

mov [ebx+eax],dl

mov al,map[eax]

mov temp0, al

.endif

.endif

;左

.if ch>0

mov eax,next

sub eax,1

if map[eax]==2

push eax

invoke SetDlgItemText,hWnd,eax,addr bu1

pop eax

mov [ebx+eax],dl

.endif

endif

;右

mov cl,al

.if ch<sz-1

mov eax,next

add eax,1

.if map[eax]==2

push eax

invoke SetDlgItemText,hWnd,eax,addr bu1

pop eax

mov [ebx+eax],dl

.endif

.endif

1.2.4 当前棋局胜负判断

在每位玩家进行完当前回合后，都需要对棋局的胜负做出判断，为了将战况同步给玩家，在窗口中创建记录玩家得分情况的按钮，实时更新两方占领的地盘数。

当某玩家前进结束时，调用judge函数遍历棋局，统计两方当前棋子数，若另一玩家棋子数为0，则本玩家直接获胜，否则，对另一位玩家的每一个棋子，判断其可跳范围内是否还有空位可占领，若每一个棋子都无路可走，则本玩家也获胜。

.while TRUE

inc esi

.break .if esi>=sz\*sz || esi<0

mov dl,map[esi]

.if dl==buFlag;本方棋子

inc mynum

.elseif dl==rival;敌方

inc rivalnum

xor ebx,ebx

.while TRUE

inc ebx

.break .if ebx>=sz\*sz || ebx<0

.if esi>ebx

mov ecx,esi

sub ecx,ebx

.else

mov ecx,ebx

sub ecx,esi

.endif

mov al,map[ebx]

.if ecx==1

||ecx==sz

||ecx==sz1

||ecx==sz+1||ecx==2||ecx==sz+sz

.if al==0

mov flag\_ctn,1

.endif

.endif

.endw

.endif

.endw

若通过judge函数判断后，发现已经决出胜负，就会弹出Win/Lose消息框，通过按钮覆盖、设置样式和延时函数，实现游戏结束的动画。

.if rivalnum==0||flag\_ctn==0

.if buFlag==1

invoke MessageBox,NULL,addr szWin,addr szResult,MB\_OK

.elseif buFlag==2

invoke MessageBox,NULL,addr szLose,addr szResult,MB\_OK

.endif

mov esi,0

.while esi<sz

mov edi,0

.while edi<sz

mov eax,esi

imul eax,40

mov ebx,edi

imul ebx,40

mov ecx,esi

imul ecx,sz

add ecx,edi

mov dl,map[ecx]

.if buFlag==1

invoke CreateWindowEx,NULL,offset button,\

offset bu1, WS\_CHILD or WS\_VISIBLE or \

WS\_BORDER,ebx,eax,40,40,hW,ecx,hInstance,NULL

.elseif buFlag==2

invoke CreateWindowEx,NULL,offset button,\

offset bu2, WS\_CHILD or WS\_VISIBLE or\ WS\_BORDER,ebx,eax,40,40,hW,ecx,hInstance,NULL

.endif

inc edi

invoke \_sleep;10000000h

.endw

inc esi

.endw

invoke \_sleep

.endifp

1.2.5单双人模式选择

在单人模式中，难点在于电脑端随即移动的实现。由于时间有限，没有对电脑的决策算法进行研发，暂时使用随机数代替。随机数生成主要使用GetTickCount库函数：

random proc value:DWORD

;随机范围为0~value

getR:

invoke GetTickCount

xor edx,edx

mov ecx,value

div ecx

mov eax,edx

.if eax<1

jmp getR

.endif

ret

random endp

A方走完，由机器代替B方进行随即移动，这一步调用的是call b函数。call\_B函数首先生成一个随机数，定位地图上的某个位置，从该位置向下遍历，遇到的第一个B，作为预定前进的棋子。那此棋子是否能够移动，需要forward函数进行判断。如果成功移动了，则B方回合结束，若没有成功，则继续遍历下一个B。

forward函数根据call\_B函数传送过来的位置定位当前的B块，调用随机数生成程序，产生12个随机数，各代表一个规则允许前进的方向，确定其中一个方向之后，在寄存器中存储如果按该方向行进，所到达的位置。

.if cl>7 && cl<12

.if cl==8

sub edi,sz

sub edi,sz

.elseif cl==9

.if column>2

sub edi,2

.endif

.elseif cl==10

.if column<sz-2

add edi,2

.endif

.elseif cl==11

add edi,sz

add edi,sz

.endif

.elseif cl>=0 && cl<=7

.if cl==0

.if column>0

sub edi,sz

sub edi,1

.endif

.elseif cl==1

sub edi,sz

.elseif cl==2

.if column<sz-1

sub edi,sz

add edi,1

.endif

.elseif cl==3

.if column>0

sub edi,1

.endif

.elseif cl==4

.if column<sz-1

add edi,1

.endif

.elseif cl==5

.if column>0

add edi,sz

sub edi,1

.endif

.elseif cl==6

add edi,sz

.elseif cl==7

.if column<sz-1

add edi,sz

add edi,1

.endif

.endif

检查当前位置是否超过地图的范围和当前位置是否为可占领的空位置，如果不符合条件，继续遍历下一个前进的方向；如果符合条件，则将B按照该方向前进，并修改地图信息，将是否前进的标志变量flag置为1，结束遍历。跳出循环之后，如果flag为1，则检查移动后的位置的上下左右四个方向是否有对方的棋子，如果有，则在地图上将对方的棋子“策反”，实现抢地盘的任务，函数结束之前将flag标志传回给call\_B函数。

2.运行效果截图



图2-1 初始界面

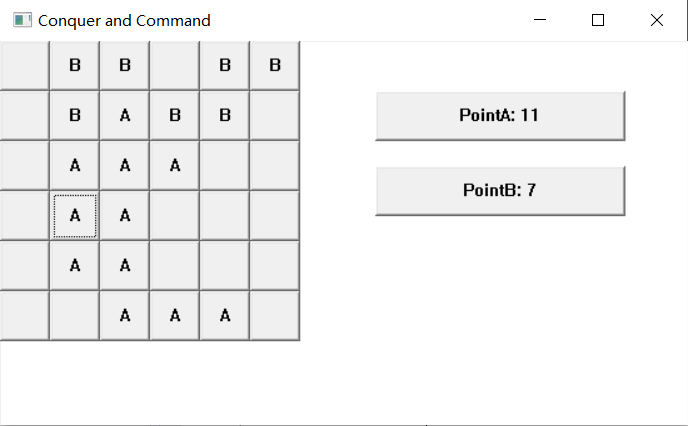


图2-2 运行界面

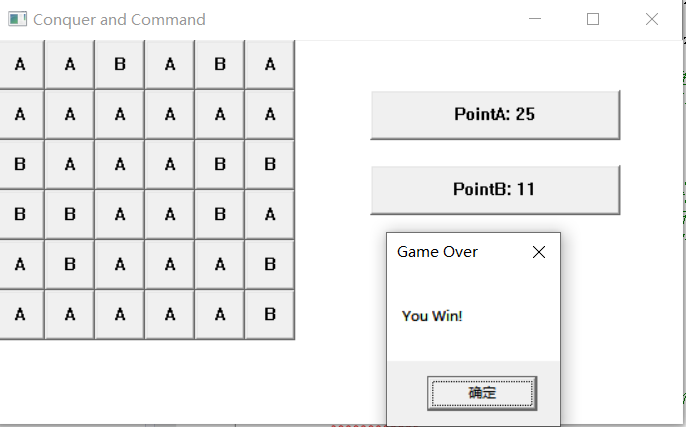


图2-3 结算界面

由上述截图可知，程序能正确执行，且运行正常。

结　论

通过这次实验，对汇编有了更深的了解。总的来说，在windows下进行界面编程的难点主要在以下几个方面。

* 如何进行界面开发？
* 如何处理鼠标事件？
* 如何进行数据计算？

进行界面开发当然是使用系统自带的API了。通过系统API创建窗口并增加通用控件即可实现界面编程。windows鼠标事件主要是通过消息类型来进行判断的，给每一个需要设置鼠标事件的控件设置ID，当鼠标点击了这个控件之后，在WM\_COMMAND事件里面通过ID进行判断。

作为第一个汇编语言大项目，代码难免写得有些冗杂，结构也有些混乱，但是经过了这个实验之后，对汇编语言的编程环境，编程方式等都有了大概的了解。基本上掌握了几个通用寄存器的使用，以及如何构造并遍历二维数组，通过二维数组中的值进行计算等，为下一个实验打下了基础。

参考文献

[1] 李元章,王娟,张全新. 汇编语言与接口技术[M]. 北京：北京理工大学，2018.

[2] 罗云彬. [琢石成器](http://ico.bit.edu.cn/opac/openlink.php?title=%E7%90%A2%E7%9F%B3%E6%88%90%E5%99%A8):Windows环境下32位汇编语言程序设计[M]. 北京：电子工业出版社，2009.

附　录

无。

致　谢

非常感谢张华平老师的批评指正，以及小组成员的努力付出。