技术文件

完成时间：2017 年 12 月 8 日

智能系统设计
课程报告

2017 年

项目名称： 基于 Diligent Nexys3 Board 的棋社梦想策略类游戏

设计小组编号：44

设计小组名单：王頔（组长），刘可

上海交通大学 电子信息与电气工程学院

姓名	班级	学号	具体负责的工作	联系方式
王頔	F1403002	5140309028	硬件平台搭建，界面设计，棋社梦想游戏逻辑，报告撰写	18217539651
刘可	F1403002	5140309025	硬件平台搭建，五子棋人工智能，报告撰写	

上海交通大学 电子信息与电气工程学院

地　址：东川路 800 号

邮　编：200240

摘　要：本文是基于 **Diligent Nexys3 Board** 的棋社梦想策略类游戏的开发报告，主要包括了项目的内容介绍、实现过程、技术难点与创新点和性能测试结果。**Diligent Nexys3 Board** 是一种常用的 **FPGA** 板卡，本项目利用该板实现了棋社梦想这一策略类游戏，同时以 **VGA** 屏幕显示，用键盘进行交互。游戏的主要内容是，玩家进入系统后，可以通过赢棋或养猪来赚钱，当金钱积攒到一定程度时便可以取胜。玩家可以与系统下五子棋，五子棋用 α - β 剪枝算法实现人工智能。项目选用 **XPS** 和其 **SDK** 作为开发软件，并添加其 **MicroBlaze** 软核用于 **VGA** 屏幕和键盘驱动，以 **VHDL** 和 **C++**作为开发工具。

关键词：策略类游戏，人工智能，五子棋，**FPGA**，**XPS**

上海交通大学 电子信息与电气工程学院

地　址：东川路 800 号

邮　编：200240

目　　录

1. 概述

1.1 编写说明

本文是基于 Diligent Nexys3 Board 的棋社梦想策略类游戏的开发报告，主要包括了项目的内容介绍、实现过程、技术难点与创新点和性能测试结果。编写此文档是为了讲解游戏系统的全貌和使用方法，同时详细记录系统设计过程与难点解决，希望为后人在使用类似产品、开发类似项目方面提供参考。此文适合有一定 FPGA 开发基础或有志于开发嵌入式游戏的人员阅读。由于笔者能力有限，其中难免有错误，欢迎批评指正。

1.2 名词定义

系统：棋社梦想游戏系统的简称

1.3 硬件开发环境

Nexys3 实验板

VGA 显示屏

标准 USB 接口的键盘

PC 一台

1.4 软件开发环境

Xilinx Platform Studio 14.2

Xilinx Sofrware Development Kit

1.5 缩略语

PC：personal computer，表示与 Nexys3 开发板连接的电脑

AI：artificial intelligence，表示五子棋算法中的人工智能

XPS：Xilinx Platform Studio，表示所用软件的 VHDL 和硬件架构搭建部分

SDK：Xilinx Sofrware Development Kit，表示所用软件的 C 语言编写部分

PVP：person vs person，表示五子棋游戏的人人对战模式

PVC：person vs computer，表示五子棋游戏的人机对战模式（用户为黑棋）

CVP：computer vs person，表示五子棋游戏的机人对战模式（电脑为黑棋）

2. 系统总述

2.1 系统组成

棋社梦想游戏系统的整体框图如图 2.1 所示，系统可以大致分为四个部分：硬件部分（右下角，图中未标出）、硬件设置部分、VHDL 部分和 C 语言部分。

其中硬件部分由 Nexys3 实验板、VGA 显示器和 USB 标准连接的键盘组成。其中显示器和键盘可以实现用户与系统的交互。

硬件设置部分在 PC 端完成，这一部分主要完成底层的工作，如 VGA 屏幕显示、键盘的驱动和中断设置，系统的代码部分将在之后利用这些设置完成相关功能。

C 语言部分和 VHDL 部分共同组成了系统的代码，其中 VHDL 部分主要完成基础功能，比如 VGA 的逐行扫描，而游戏中最复杂的部分均在 C 语言中实现，包括屏幕设计、游戏 AI 和按键处理等。

关于数据的交互，如图所示，C 语言部分的数据通过 32 位总线传递到 VHDL 中处理；硬件之间通过 USB 数据线和 VGA 数据线进行连接。

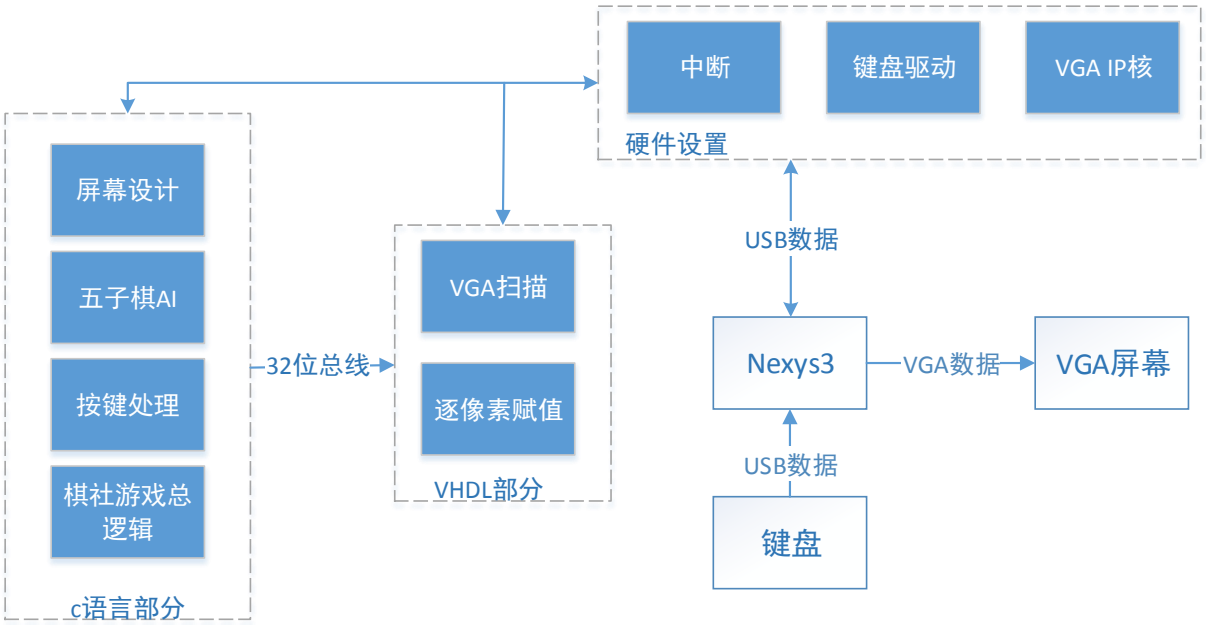

图 2.1 系统整体框图

2.2 系统的主要功能

棋社梦想游戏主要是一个通过下棋获取金钱，并通过一些手段赚钱，最终积攒到一定财富便可以获胜的养成类游戏。

系统的主要功能如图 2.2 所示，玩家在进入系统后将会有吃饭、卖猪、买猪、下棋和结束游戏五个选项，选择不同的选项则体力值、钱数会有相应变化。下棋（五子棋）后如果赢棋则会钱数增加，否则减少；当玩家有一定的钱数之后，便可以选择养猪来赚钱，猪饲料随着每局结束后会减少；经过一段时间，猪长成熟后便可卖出以获得金钱。

需要注意的是，如果猪无法按时获取饲料，则会死亡，此时将会赔本；如果玩家在游戏中任何时刻体力值或钱数小于零，则游戏结束，玩家输；当玩家选择结束游戏时，如果钱数大于 30 则获胜，否则即判为输。

在游戏中，玩家在 VGA 屏幕上观看游戏界面，通过键盘的方向键控制光标，用 enter 键确认。

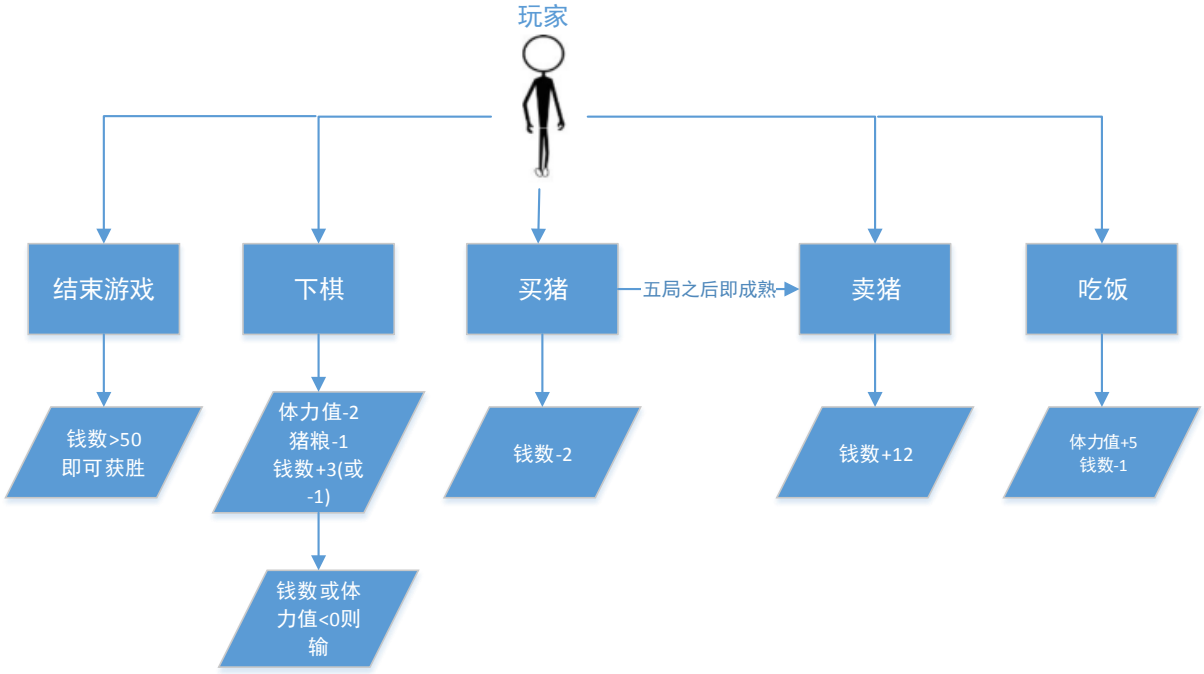

图 2.2 系统主要功能框图

3. 棋社梦想游戏系统实现过程

整个棋社梦想游戏系统的软硬件开发工作流程如下：从 ISE 基本设计输入开始，先生成 XPS 硬件系统架构，再到 SDK 中进行软件开发，接着在 ISE 完成综合、约束、映射等编译步骤，最终生成可下载文件进行板级的调试验证。

其中，我们使用 XPS 添加了一个带 MicroBlaze 软核的嵌入式系统，这样我们便可以将开发平台转移到 SDK 上，利用 C 语言完成较复杂的程序设计；与此同时，此 IP 核自动生成了键盘驱动和 VGA 显示驱动。

所以，棋社梦想游戏系统实现过程主要分为四部分讲解：硬件配置、棋社梦想游戏总体逻辑、五子棋人工智能和屏幕显示。

3.1 硬件配置

硬件配置主要就是添加 MicroBlaze 软核的过程，完成了 IP 核的配置、键盘驱动和中断配置的功能。这一过程在 ftp 上有详细的资料讲解[1]，在此不再赘述，主要分为以下三个基本步骤：

1. 在 Xilinx Platform Studio 中新建一个基于 Microblaze 的硬件系统
2. 修改 vga_ip 的 MPD、vhd 文件
3. 修改源文件 user_logic.v

3.2 棋社梦想游戏总体逻辑

棋社梦想的总体逻辑如图 3.1 所示，可以看出，程序由一个 while 主循环和四个 if 判断组成。

在刚开机时，进入 while 循环之前程序会初始化界面，此时显示的是主菜单。

接着进入循环，一般情况下，循环中需要等待用户落子才能继续，当用户落子后，由几个 if 判断来进行相应处理，如图所示，用户请求包括转换界面、落子、主菜单界面的用户请求（如结束游戏、吃饭等）。而当程序处在电脑落子的状态下时，不需要等待用户按键就直接开始下一循环。所以，为了处理按键时的逻辑正确，程序将电脑落子作为一个单独部分提出。

当游戏结束时，会先显示结果，再按任意键重新开始新的一局。

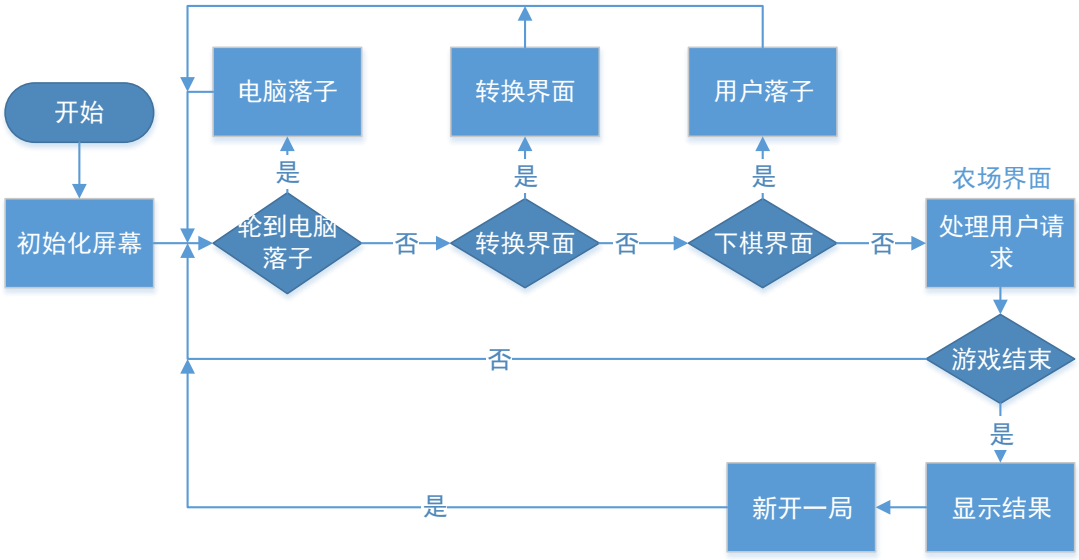

图 3.1 棋社梦想总体逻辑

3.3 五子棋人工智能

五子棋人工智能是游戏系统的核心模块，它包含了输赢判断、禁手判断、搜索函数（用于遍历范围内可以落子的点）、估值函数（用于判断某一点是否适合落子）四个部分；最后，文中还讲解了人机对战、人人对战的单局流程。

3.3.1 输赢判断

输赢判断是最基础的人工智能函数，它获取的输入为下棋的横竖坐标，返回 0 或 1 表示有无赢棋出现。

此函数流程图如图 3.2 所示[4]，具体的逻辑如下：

1. 以落子点为中心，依次遍历水平、竖直、左斜、右斜四个方向；
2. 任意方向有五子相连，则判为赢，返回 1；否则返回 0。

其中扫描任意方向时采用了枚举法，现以白方落子、水平方向扫描为例，说明具体的逻辑：先向右扫描并计数，至第一个黑棋处返回；再向左扫描并继续计数，至第一个黑棋处返回；如果扫描到了五个白子，则记录白方胜利，否则继续扫描其他方向。

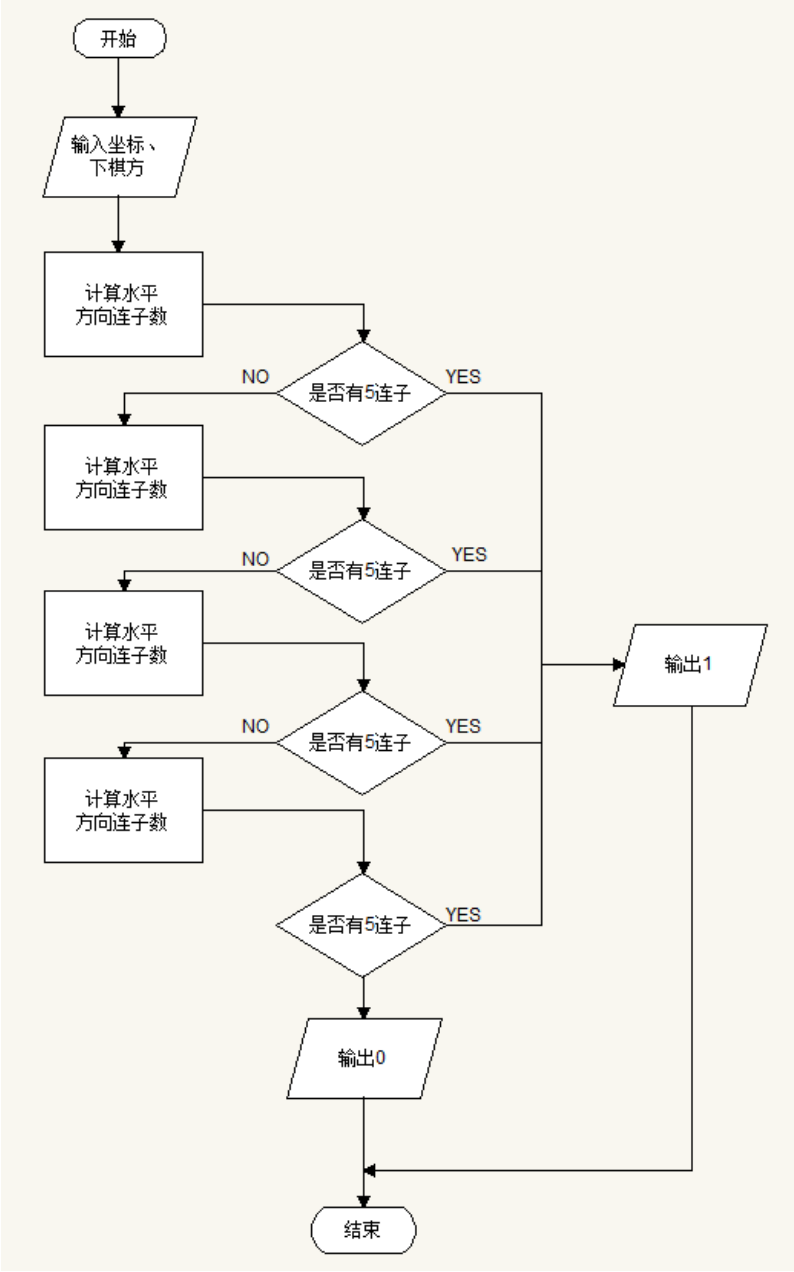

图 3.2 输赢判断流程图

3. 3. 2 禁手判断

禁手规则之所以会出现，是因为五子棋中先行的一方有明显的优势。禁手一般需要禁止双三、双四、长连三种情况。具体如图 3.3[2]所示其中从上到下各行依次表示双三、双四和长连。

禁手判断的程序实现也采用了枚举法，具体方法与输赢判断类似。其中需要注意的是要避免对方棋子的干扰，比如对方如果已经挡住了相连三子的一边，此三子则不能判定为活三。

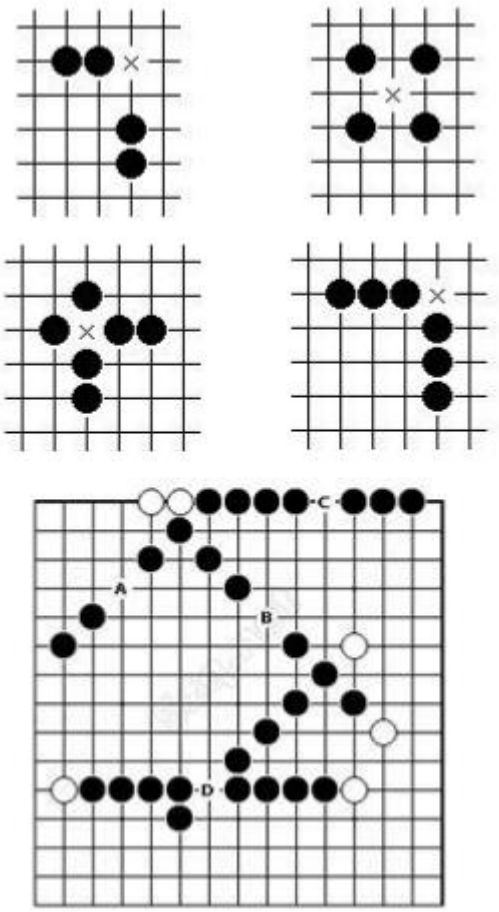

图 3.3 禁手的三种情况

3.3.3 估值函数

估值函数是 AI 用于判断某一点是否适合落子的，该函数获取的输入是位置坐标，输出是这一点的评分。

我们采取的具体评分方法是对于五子、四子、三子、二子相连、独立一子与未堵、一头堵、两头堵组合而成的 15 种情况分别赋予不同的分数，各分数值如表 3.1 所示。在函数中，先分别计算某一点水平、竖直、左斜、右斜四个方向的得分，然后将四个方向的分数值相加，得到这一点的总的分数。估值函数整体流程如图 3.4 所示。

表 3.1 估值函数评分方法

	未堵	一头堵	两头堵
连成五子	1000000	1000000	1000000
连成四子	300000	2500	100
连成三子	3000	500	100
连成两子	700	200	0
独立一子	200	50	0

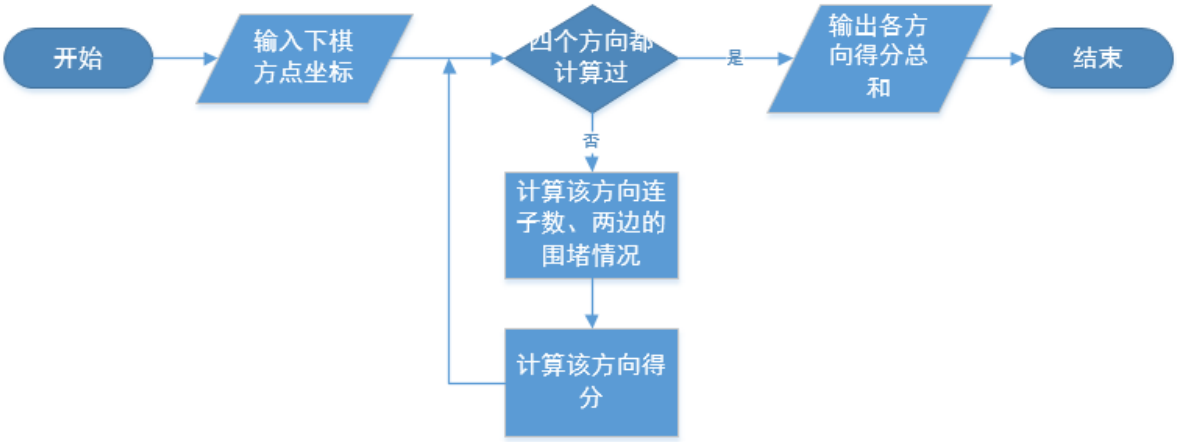

图 3.4 估值函数流程图

3.3.4 搜索函数

搜索函数的作用是便于 AI 判断范围内最适合落子的点，我们在系统设计过程中先尝试了遍历搜索，计算量太大，于是采用了课程推荐的 α - β 剪枝算法，在此具体讲解剪枝算法的实现。

α - β 剪枝技术是在极小极大分析法的基础上提出的一种更高效的分析方法。极小极大分析法实际是先生成一棵博弈树，然后再计算其倒推值，但这样做效率较低。（极大极小分析法具体讲解可参考资料 3。）

α - β 剪枝技术的基本思想是边生成博弈树边计算评估各节点的倒推值，并且根据评估出的倒推值范围，及时停止扩展那些已无必要再扩展的子结点，即相当于剪去了博弈树上的一些分支，从而节约了机器开销，提高了搜索效率。

剪枝算法在五子棋 AI 中的应用如下[4]：

1. 将走棋方定为 max 方，对其子结点的评估值取极大值，即选择对自己最为有利的走法。将对阵方定为 min 方，对其子结点的评估值取极小值，即选择对走棋方最不利的走法。
2. α 剪枝：
   a) 从左路分支的结点倒推得到某一层 max 节点的值，可表示到此为止得以“落实”的走法最佳值，记为 α 。此值可作为 max 方走法指标的下界。
   b) 在搜索此 max 结点的其它子结点时，如果发现一个回合（两步棋）之后评估值变差，即孙结点评估值低于下界 α 值，就可以剪掉此枝。
3. β 剪枝：
   a) 由左路分枝的叶节点倒推得到某一层 min 节点的值，可表示到此为止对方走法的钳制值，记为 β 。此值可作为 max 方无法实现走法指标的上界。
   b) 在搜索该 min 节点的其它子节点时，如果发现一个回合之后钳制局面减弱，即孙节点评估值高于上界 β 值，则便可以剪掉此枝，即不再考虑此“软着”的延伸。

3.3.5 人人对战与人机对战单局流程

两种模式的单局流程图如图 3.5，3.6 所示。

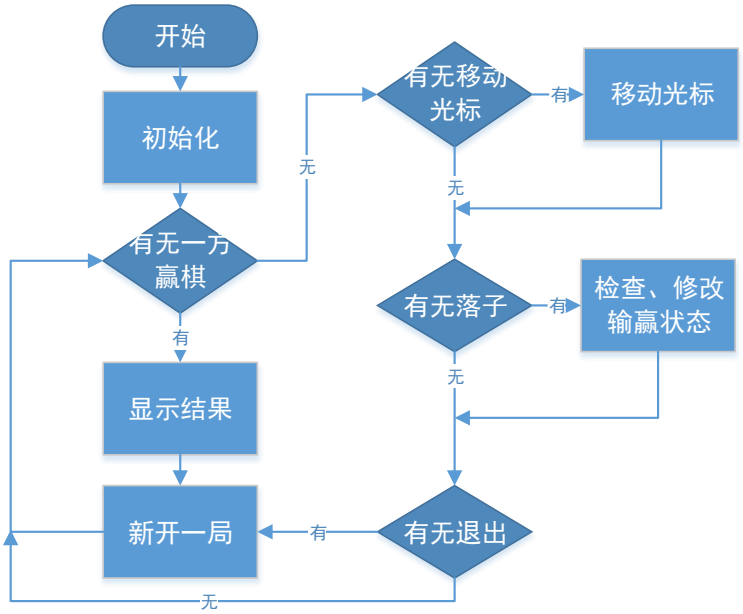

图 3.5 人人对战单局流程图

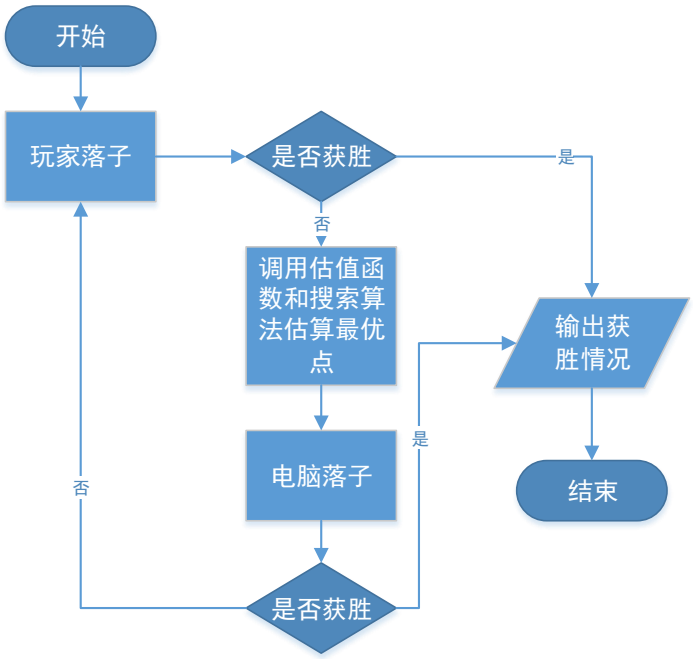

图 3.6 人机对战单局流程

3.4 屏幕显示

3.4.1 程序基础

屏幕显示功能是整个系统中非常重要的模块，由 C 语言和 VHDL 语言共同完成。

其中，VHDL 完成的功能有三点：

1. 利用中断实现 VGA 扫描、像素值输出；
2. 创建并维护多个屏幕数组：接收来自 C 语言部分的数据赋值给数组，并将数组中的值输出给 VGA；
3. 画棋盘线、棋子（圆形）这样简单且无变化的图案；

因为 VHDL 部分较为基础，资料完善，且以前课程有不少类似内容，在此不再讲解实现过程，具体可参考附录 A 中 VHDL 部分的代码。

C 语言完成的功能为整体布局和各种图案设计，将在 3.4.2 和 3.4.3 中详细讲解。

正如前文提到，C 语言与 VHDL 语言两部分数据的交互是通过 MicroBlaze 的 32 位的总线实现的，这 32 位总线的利用方式如下：

1. [0 : 7]位表示要修改的像素的横坐标，[8 : 15]位表示纵坐标；
2. [16 : 23]位表示屏幕状态：农场界面或下棋界面；
3. [24 : 31]位为该像素的颜色，其取值为 0~3，对应四种颜色。颜色对应代码如表 3.1 所示。

表 3.2 像素颜色对应代码

编号	二进制码	颜色	下棋界面功能	农场界面功能
0	8’b11111111	白色	白棋	背景
1	8’b00000000	黑色	黑棋、棋盘线	文字
2	8’b11011011	蓝色	背景	光标
3	8’b11100000	红色	光标	文字

3.4.2 整体布局

游戏系统共有两个页面：主菜单页面和下棋页面。

主菜单页面如图 3.7 所示。屏幕左侧是各种参数的值，从上到下依次是：饲养的猪的数量、猪食余量、体力值、钱数和成熟的猪的数量。屏幕右侧则代表了五个选项，其中蓝色的是光标所在位置。屏幕底端则显示“welcome to the farm”，欢迎玩家参与游戏。

需要说明的是，系统中所使用的 VGA 屏幕有 480*640 个像素点，为了程序中控制方便，我们将 4*4 个像素合并成为一个逻辑像素，所以在程序中，主菜单页面体现为一个 120*160 的数组。

当玩家按下键盘 c 键时，游戏会转换到下棋页面，如图 3.8 所示。

下棋页面左侧为 15*15 的棋盘，在棋盘线的交点上显示黑白双方的棋子；右侧为菜单，显示当前落子倒计时，如果超时将自动换为对方落子。

在程序中，棋盘体现为一个 15*15 的数组用于显示落子情况，菜单体现为一个 10*30 的数组，合并像素赋值以便于控制。

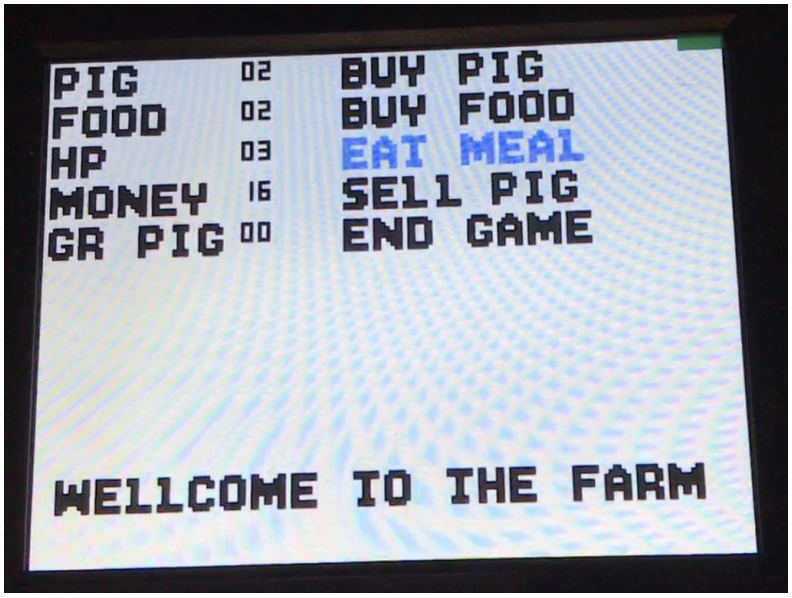

图 3.7 主菜单页面

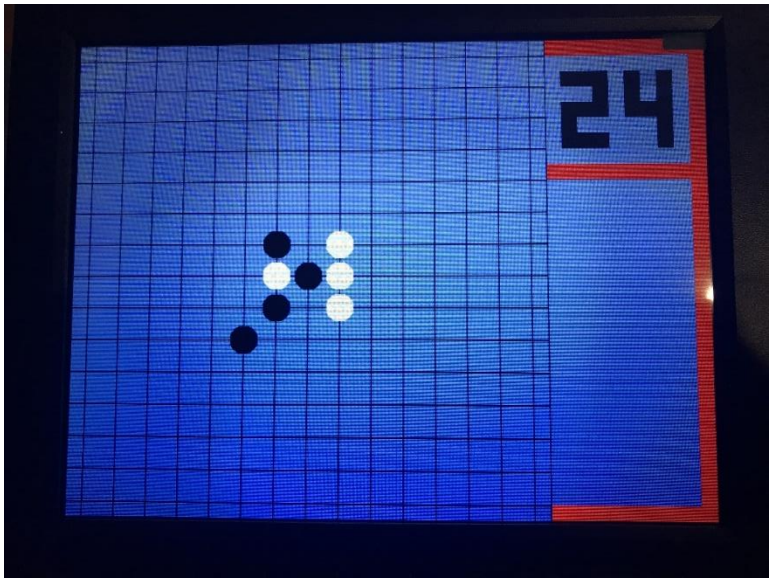

图 3.8 下棋界面

3.4.3 图案设计

（1） 数字图案

数字图案较为复杂且经常变化，所以在 C 语言中编写，如图 3.9 所示，每个数字在 3*6 的区域内涂色。具体而言，在 C 语言中定义函数 drawNumber，并在 main 函数中调用。

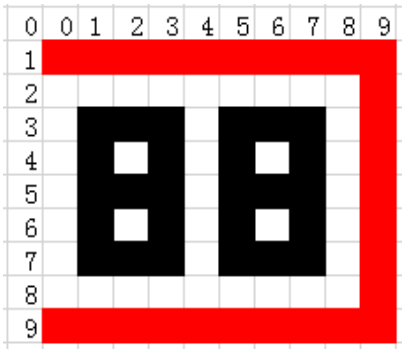

图 3.9 数字图案设计

（2） 获胜图案

获胜图案如图 3.10 所示，以赢方的棋子摆出“WIN”的图案即可。同样在 C 语言中设计，定义 drawWin 函数并调用。

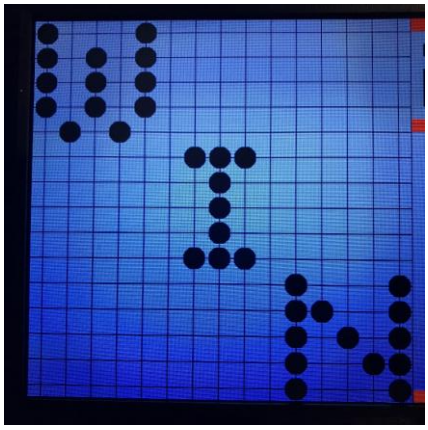

图 3.10 获胜图案设计

（3） 字母图案

字母图案最为复杂，因为程序比较底层，没有字母相关的库可以调用，所以我们从网上找了一个图案设计并手动输入，如图 3.12 所示。

每个字母基本上在 6*8 的区域内，像素点较多，以字母‘F’和‘A’为例，如图 3.11。且共有 26 个字母，所以工作量较大。

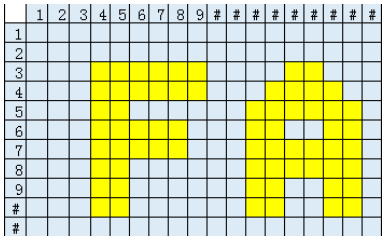

图 3.11 字母图案设计范例

图 3.12 字母图案设计

4. 技术难点和创新点

4.1 创新点

本游戏在课程传统项目五子棋的基础上改进，结合近年来比较流行的养成类游戏，开发出了一款嵌入式的养成类游戏。游戏最终的效果类似童年时的游戏机：黑白字母显示当前体力值和财富，按方向键控制光标，通过赢棋积攒财富，到农场买猪并投饲料……屏幕最下方的“WELCOME TO THE FARM”使复古的情节变得更加明显。

4.2 技术难点

4.2.1 难点 1：代码简化

在实现过程中，因为字母图案比较复杂，而且数量很多，所以有上千行代码；加之程序其他部分，最终导致程序行数过多，超过内存上限，所以需要简化代码。

我们逐步的简化思路如下：

1. 定义 drawPixel 函数，将每个像素点的赋值定义为一个函数，使两行合并为一行
2. 定义 drawLine 函数，将字母图案中按列简化，某一列满格则合并为一个语句，这样可以讲八句合一，如图 3.5 中的‘F’第一、二列
3. 定义 drawLine2 函数，通过二进制数表示字母图案，这样每一个字母只需要六行即可，如图 3.5 中‘F’第三列的二进制数为 11011000

4.2.2 难点 2：按键处理

系统中有两个页面，所以按键处理比较容易出错，例如，我们曾遇到过如下问题：

1. 在一个 while 循环中放两个等待按键的判断，导致需要按键两次才能实现一个功能
2. 电脑落子的部分放在了按键判别里，导致按键以后屏幕一直闪烁，无法实现功能
3. 更多时候，按键没有任何反应，难以推出真正原因，只能多次尝试

4.2.3 难点 3：软件使用

软件部分没有相关资料，网上能查到的内容也非常有限，所以我们曾遇到很大的阻碍。于是，我们总结出了几点容易出错和困惑的地方，这也是我们遇到最困扰的部分，希望可以为之后使用 XPS 与 SDK 的同学做一些参考。

1. 运行时的问题
   a) 运行程序时，要先在 XPS 中综合下载，再到 SDK 中点击 debug 和运行，否则 c 语言中的内容不会显示
   b) 重新下载程序到板子上后，要重启一次 SDK，否则会显示连接失败，无法正常运行

c) 在 SDK 的程序修改后，常需要重启，否则 debug 过程会卡住

d) 综合、下载、运行过程都较慢，XPS 中下载大约需要 3-5 分钟，SDK 中需要 3 分钟左右

e) 运行过程中，程序有时会出现突然卡住的现象(尤其刚开机时)，此时等一会儿或重新下载程序即可

2. 报错时的问题

a) 软件报错时，信息在隐藏的小窗中，进而软件会直接下载之前的程序，所以体现出来的是程序始终没有任何改变

b) 在 SDK 中点击运行之后，会有多个报错信息出现，其中有的可以直接忽视，有的则会有影响

i. 典型有影响的错误信息：c 语言中的语法错误；程序行数过多，板子内存越界

ii. 典型可忽视的错误信息：各种配置问题，如找不到某文件

c) 报错信息可能不够直接，比如板子内存越界会显示为 byte64 被覆盖

5. 系统功能及测试

刚进入系统时的开机界面如图 5.1 所示，刚开机时体力值为 5，其余为 0。右边蓝色光标通过 wasd 方向键控制，enter 键输入。

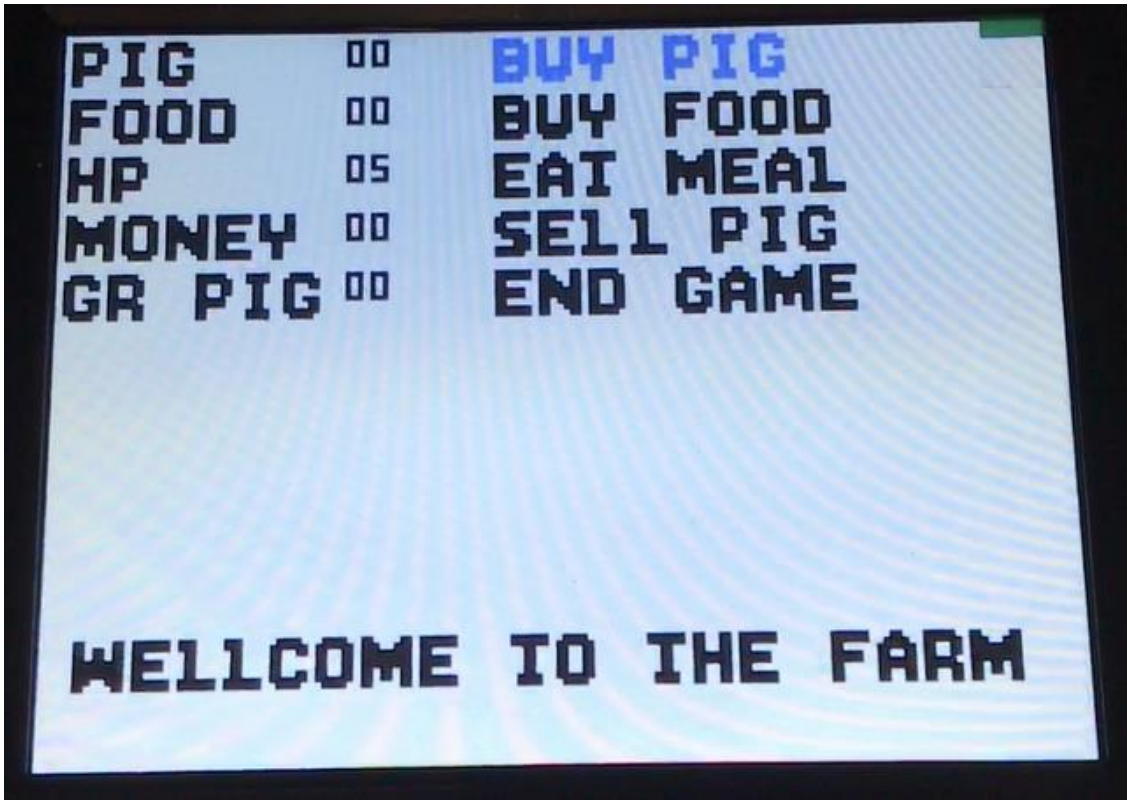

图 5.1 系统开机界面

按下 c 键切换到下棋界面，如图 5.2 所示。下棋时通过 wasd 四个方向键控制光标，enter 键下棋。下棋界面有人人对战和人机对战两种模式，通过右下角的标志“PVP”、“PVC”或“CVP”来区分，通过空格键切换模式。右上角倒计时表示每方落子可以思考的时长为 30 秒，超时则自动换为对方下棋。

此局玩家获胜，结果如图 5.3 所示，用黑色棋子摆出“WIN”的形状。

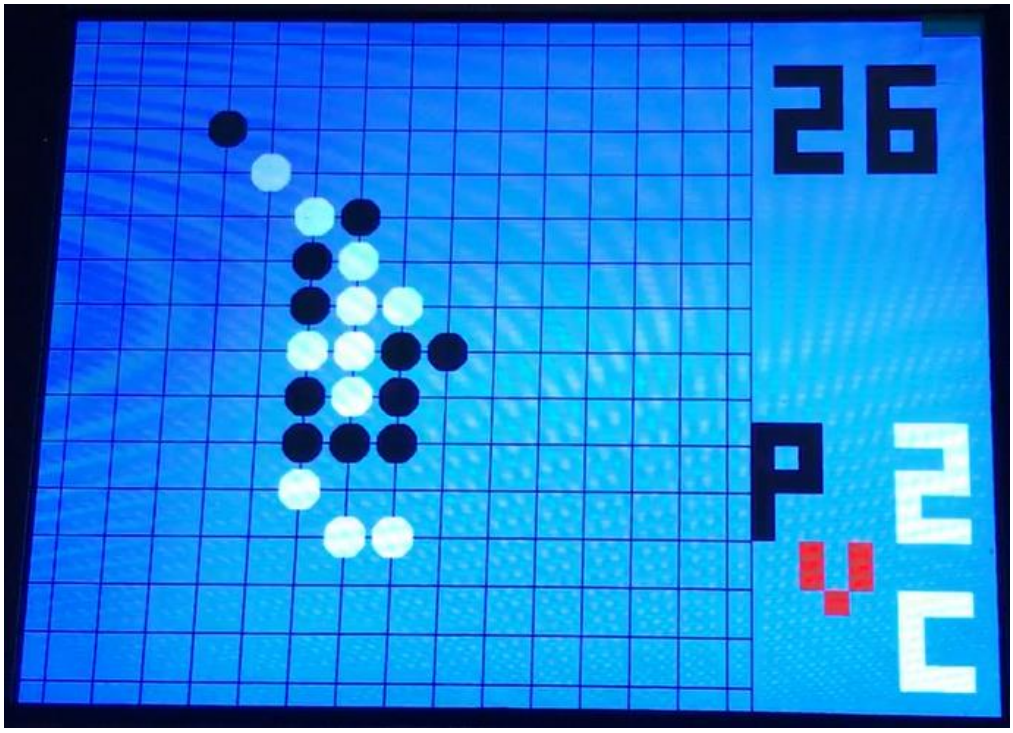

图 5.2 下棋界面

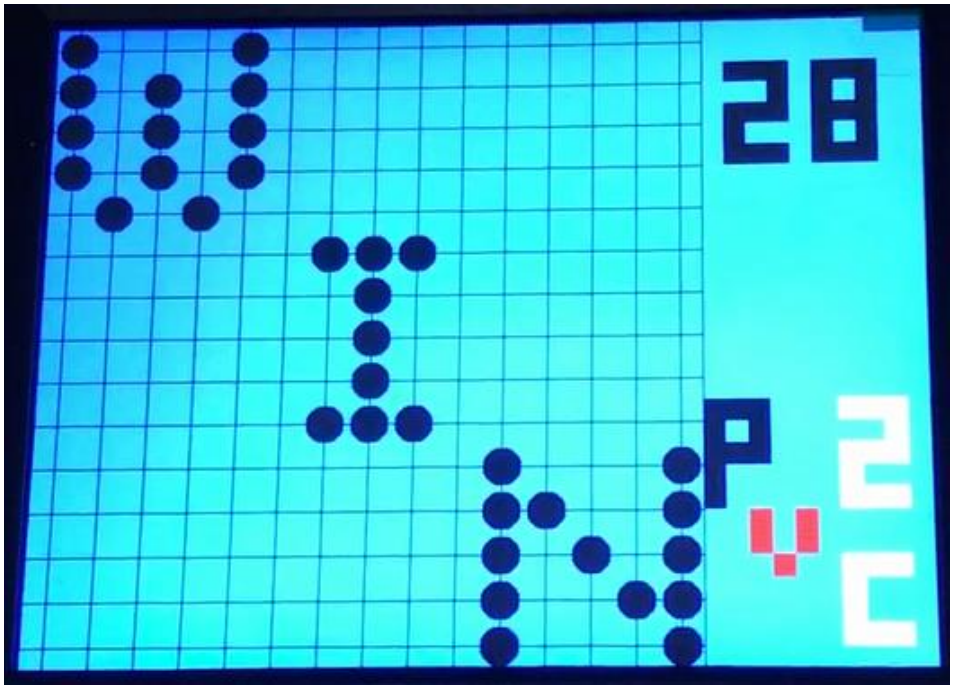

图 5.3 玩家获胜界面

接着返回主菜单观察此时数据，如图 5.4 所示。可以看到下一局棋体力值减 2，钱数在赢棋后加 5。

图 5.4 玩家获胜后参数变化

再下一局棋使得电脑胜利，观察参数变化如图 5.5 所示，可以看到体力值减少 2，输棋后玩家钱数减 1。

图 5.5 玩家输棋参数变化

此时如果选择结束游戏，因为钱数不到 30，则会用红色显示“YOU DIE”表示玩家战败，如图 5.6 所示。此时按任意键即可重新开始一局。

图 5.6 玩家战败页面

再新开一局，并使得游戏运行一段时间后，目前参数如图 5.7 所示。此时已经有两只猪快要长成熟（再有一局即可成熟），猪食和体力值较少，剩余钱数较多。

图 5.7 运行一段时间后参数

所以用一枚金币吃一顿饭后，体力值加五；用一枚金币买一分猪食后，饲料数加五；再买两只猪，花去 4 枚金币。完成这些操作后系统参数如图 5.8 所示。

图 5.8 完成操作后的参数

接着下一局棋后，有两只猪长成，如图 5.9 所示。同时，猪食减 4（每一只猪一局需要一份猪食）。

此时卖掉这两只猪，可以发现猪数减二，钱数增加 24（每只猪可以卖 12 枚金币），如图 5.10。

图 5.9 下一局棋后的参数

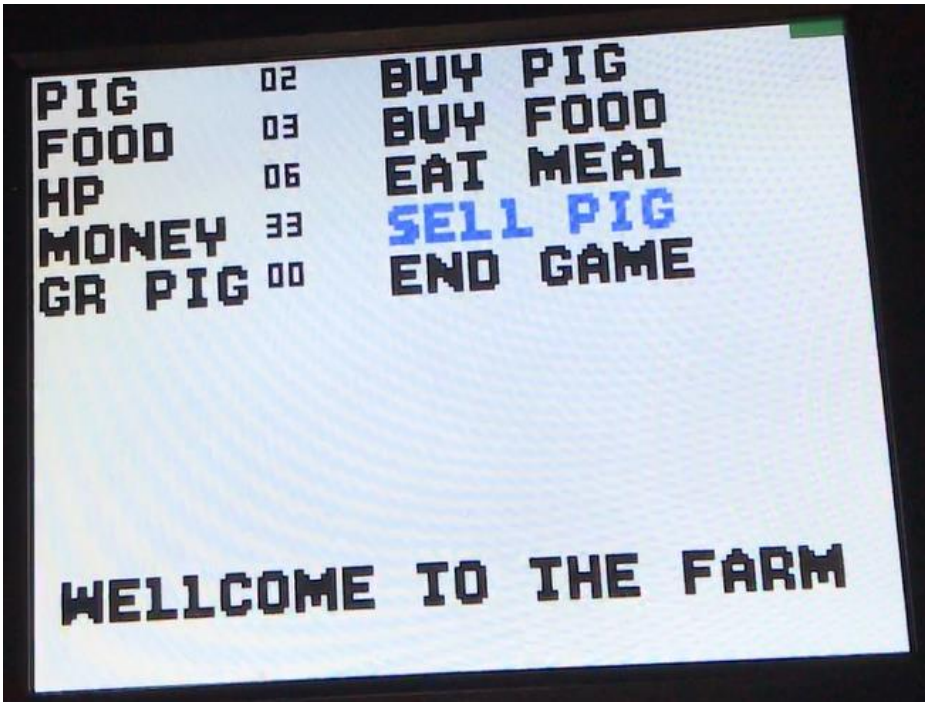

图 5.10 完成操作后的参数

因为此时钱数大于 30，选择结束游戏，则会显示“YOU WIN”，表示玩家获胜，如图 5.11 所示。

图 5.11 玩家获胜页面

最后，系统整体实物连接图如图 5.12 所示。图中左边从上到下依次是：VGA 显示屏幕、键盘、Nexys3 实验板，右边为 PC 端；各实物之间主要依靠 USB 数据线和 VGA 数据线相连，具体连接方式参考图 2.1 系统整体框图。

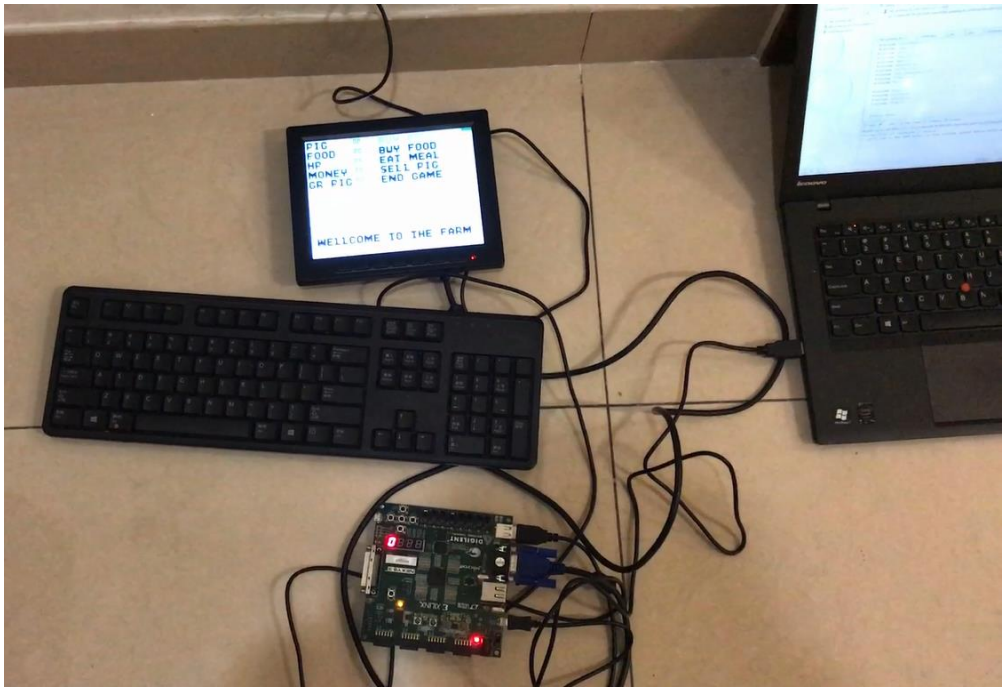

图 5.12 系统整体实物连接图

6. 致谢

在本次实验中，我们得到了来自老师、助教和其他各组的帮助。在他们的帮助下，我们能顺利地完成本次实验，解决实验产生的问题，在此感谢所有人对我们的帮助。

首先要感谢杨老师的帮助，不论我们遇到什么难以解决的问题，只要发邮件或当面求助，老师总会及时回复，仔细解答，这令我们十分感激！FTP 上详细的资料也使我们上手软件、设计系统变得更加容易。

其次要感谢助教老师，非常热心地帮我们解答软件使用方面的各种难题，自己不熟悉时甚至会帮我们上网搜索，为我们减轻了很多负担！

还要感谢我的组员，大四上我们俩的时间都非常紧，但是我们还是愿意一起研究，互相理解，这是多么难得。这是我们大学期间合作的最后一个实验了。三年以来，我们一起合作了所有的实验课，到如今终于画上了句号。互相理解和鼓励使得大学期间繁忙的日子不再那么艰难，谢谢你。

最后要感谢其他所有同学给我们的帮助、支持和鼓励。答辩完后我们其实并不太满意，是你们为我们竖起大拇指，谢谢你们。

7. 参考文献

[1] 上海交大电子工程系. 基于 XilinxEDK 的 VGA 设计参考[EB/OL]. ftp://202.120.39.248.

[2] 上海交大电子工程系. 五子棋规则说明-五子棋系统使用说明书[EB/OL]. ftp://202.120.39.248.

[3] 上海交大电子工程系. 2016-智能系统设计讲座 2-五子棋设计&实验平台[EB/OL]. ftp://202.120.39.248.

[4] 上海交大电子工程系. 智能系统设计-第 028 组设计报告，2016

附录 A　部分软件程序

由于页数限制，我们选取了部分软件程序附在报告中，以便查看。具体选取了：

1. VHDL 部分屏幕数组赋值和输出的部分代码
2. C 语言主函数部分的 while 主循环
3. C 语言屏幕设计部分中的 drawPixel、drawLine 和 drawLine2 三个函数
4. C 语言 AI 部分的输赢判断和估值函数

1.　VHDL 程序

给屏幕数组赋值的部分代码：

```
always @( posedge Bus2IP_Clk )
  begin
    if ( Bus2IP_Reset == 1 )
        begin
                    for (i = 0; i < 225; i=i+1)
                      board[i] = 2'b10;
                    for (i = 0; i < 300; i=i+1)
                      menu[i] = 2'b10;
                            //welcome[i] = 2'b10;
//                  for (i = 0; i < 96; i=i+1)
//                  for (j = 0; j < 128;j=j+1)
//                    welcome[i*128+j] = 2'b00;
                  end
       else
         begin
       if ( Bus2IP_WrCE == 1'b1)
                    begin
                      if (Bus2IP_Data[16 : 23]== 0)//棋盘部分修改
                                begin
                                        page = 0;//棋盘
                                   board[Bus2IP_Data[0 : 7] + Bus2IP_Data[8 : 15] * 15] = Bus2IP_Data[24 : 31];
                       end
                                else if (Bus2IP_Data[16 : 23]== 1)//菜单部分修改
                                begin
                                        page = 0;
                                   menu[Bus2IP_Data[0 : 7] + Bus2IP_Data[8 : 15] * 10] = Bus2IP_Data[24 : 31];
                                end
                                else if (Bus2IP_Data[16 : 23] == 2) //主菜单
                                begin
                                        page = 2;//welcome 页面 96*128 个像素点
                                        welcome[ Bus2IP_Data[0 : 7] + Bus2IP_Data[8 : 15] * 160 ] = Bus2IP_Data[24 : 31];
                                end
                    end
       end
  end
```

输出 VGA 值的部分代码：

```
always@(*)
  begin
    if (vga_ena == 1)
        begin
                    if (page == 0)//棋盘部分&菜单部分
                              begin
                                        if (x_cor < 481)//棋盘
                                                  begin
                                                            if((x_cor%32-16)*(x_cor%32-16)+(y_cor%32-16)*(y_cor%32-16)<196
                                                                      && board[x_cor / 32 + y_cor / 32 * 15]!= 2)
                                                                vga_data <= colormap[board[x_cor / 32 + y_cor / 32 * 15]]; //
画棋子
                                                            else if(x_cor[5 : 9]==5'b10000 || y_cor[5 : 9]==5'b10000 || x_cor==480)
                                                                      vga_data <= linecolor;//画棋盘线
                                                            else vga_data <= colormap[2];
```

```
                                        end
                                    else//菜单部分
                                     begin
                                      if (menu[(x_cor - 480) / 16 + y_cor / 16 * 10] != 2)
                                                vga_data <= colormap[menu[(x_cor - 480) / 16 + y_cor / 16 * 10]];
                                            else
                                                vga_data <= colormap[2];
                                     end
                            end
                    else if (page == 2)//welcome 开始界面
                                    begin
                                            if (welcome[x_cor / 4 + y_cor / 4 * 160] != 0)
                                                vga_data <= colormap[ welcome[x_cor / 4 + y_cor / 4 * 160] ];
                                            else
                                                vga_data <= colormap[0];
                                    end
                end
        else
                vga_data <= 0;
 end
```

2. C 语言之主程序

```
while (1)
  {
     if (page == 1 && turn==COMPUTER_PLAYER && (status==PVC ||status==CVP )&& win_flag==0) //电脑下棋
       {
          if (step_flag==0) {
             if (x_cur-1<0)
                x_pos=x_cur+1;
             else
                x_pos=x_cur-1;
                y_pos=y_cur;
                step_flag=1;
          }
          else {
             if(level==2||level==3||level == 1)
             {
                EvaluateComputerMove(board_state, 0, MIN_INFINITY, MAX_INFINITY, 0, 0);
                x_pos=maxMoveX;
                y_pos=maxMoveY;
                xil_printf("\r\n computer \r\n");
             }
//              else
//              {
//                 everyScore(Computer);
//                 current_pos=best(Computer);
//                 x_pos=current_pos.y;
//                 y_pos=current_pos.x;
//                 xil_printf("\r\n computer \r\n");
//              }

          }
          xil_printf("\r\n%x, %x\r\n", x_pos, y_pos);

          if(status==CVP)
          DrawChess(x_pos, y_pos, 1-turn);
          else
          DrawChess(x_pos, y_pos, turn);


          board_state[x_pos][y_pos]=COMPUTER_PLAYER;
          board_record[BackTimes].x=x_pos;
          board_record[BackTimes].y=y_pos;
          BackTimes++;

          count=0;
          time0=0;
          if (CheckWin(x_pos, y_pos, turn)) {
             xil_printf("\r\nComputer Player wins!\r\n");
             win_flag=1;
```

```
            DrawWinning(0,1, EMPTY);
            if(status==CVP)
               DrawWinning(0,1, 1-turn);
            else
               DrawWinning(0,1, turn);

            //money--;
            afterGame(0);
            turn=HUMAN_PLAYER;
         }
         else
         {
            turn=HUMAN_PLAYER;
            xil_printf("\r\nHuman Player's turn!\r\n");
         }
      }//电脑下棋

    else {
     do {//等待键盘输入
        if (turn==COMPUTER_PLAYER && status==PVC && page== 1)
                  break;
           dip_check=XGpio_DiscreteRead(&dip, 1);
           StatusReg = XPs2_GetStatus(&Ps2Inst);
        }while((StatusReg & XPS2_STATUS_RX_FULL) == 0);
     BytesReceived = XPs2_Recv(&Ps2Inst, &RxBuffer, 1);
     key_count = (key_count + 1) % 3;
     if (key_count==0){

     if (win0 == 1) win0++;
     if (die == 1) die++;

     if (die == 0 && win0 == 0){//按下去的时候

        if ( RxBuffer==0x21 )//c 键
        {
              page = (page + 1) % 2; //从 1 变成 0 或 从 0 变成 1
              if (page == 1) //棋盘
              {
                 InitializeGame(x_cur, y_cur);
                 status=PVP;
              }
              else //开始菜单
              {
                 drawBuy(cursor0);
                 drawFoodNum(pig,food,hp,money,grownPig);
                 //drawFood(1);
                 //drawWin(0,0,2,2);//蓝色
              }
              //break;
        }

    if (page == 0) //主菜单
    {
       if (RxBuffer==0x1D)//光标上
       {
          cursor0--;//光标上移
          if (cursor0 < 0) cursor0 = 4;
          drawBuy(cursor0);

       }
       if (RxBuffer==0x1B)//光标下
       {
          cursor0 = (cursor0 + 1) % 5;//光标下移
          drawBuy(cursor0);

       }

       if  (RxBuffer==0x29)//空格 买东西
       {
          switch (cursor0)
```

```
            {
              case BUY_PIG: buyPig(); break;
              case SELL_PIG: sellPig(); break;
              case EAT_MEAL:
                        money--;
                        hp+=5;
                        drawFoodNum(pig,food,hp,money,grownPig);
                        break;
              case BUY_FOOD:
                        money--;
                        food+=5;
                        drawFoodNum(pig,food,hp,money,grownPig);
                        break;
             case 4://end game
                        if (money >= 10) win0 = 2;
                        else die = 2;
            }
          }

        }



        if(page == 1 && die == 0 && win0 == 0) //玩游戏的页面
        {
          if (turn==HUMAN_PLAYER || (turn==COMPUTER_PLAYER && status==PVP)) {
//          do {//等待键盘输入
//            if (turn==COMPUTER_PLAYER && status==PVC)
//              break;
//            dip_check=XGpio_DiscreteRead(&dip, 1);
//            StatusReg = XPs2_GetStatus(&Ps2Inst);
//          }while((StatusReg & XPS2_STATUS_RX_FULL) == 0);
//          BytesReceived = XPs2_Recv(&Ps2Inst, &RxBuffer, 1);
//          key_count=(key_count+1)%3;
           //if (key_count==0) {
//            if (RxBuffer==0x21)
//            {
//              page = 0;
//              drawLetter('A',10,10,1,2);//初始化画开始界面
//              break;
//            }
//            if (RxBuffer==0x21 && win_flag==0) {//改变难度？
//              DrawNumber(level,6,18,EMPTY);
//
//              if(level==1)
//                level=2;
//              else if(level==2)
//                level=3;
//              else
//                level=1;
//
//              if(status==PVC )
//              DrawNumber(level,6,18,0);
//              else if(status==CVP)
//              DrawNumber(level,6,18,1);
//              else
//              DrawNumber(level,6,18,EMPTY);
//
//            }
            if (RxBuffer==0x1D && win_flag==0) {//上下左右移动 wasd 上
              EraseCursor(x_cur, y_cur);
              if (y_cur<=0)
                y_cur=14;
              else
                y_cur--;
              DrawChess(x_cur, y_cur, CURSOR);
            }
            if (RxBuffer==0x1B && win_flag==0) {//下
              EraseCursor(x_cur, y_cur);
              if (y_cur>=14)
                y_cur=0;
```

```
    else
        y_cur++;
    DrawChess(x_cur, y_cur, CURSOR);
}
if (RxBuffer==0x1C && win_flag==0) {//左
    EraseCursor(x_cur, y_cur);
    if (x_cur<=0)
        x_cur=14;
    else
        x_cur--;
    DrawChess(x_cur, y_cur, CURSOR);
}
if (RxBuffer==0x23 && win_flag==0) {//右
    EraseCursor(x_cur, y_cur);
    if (x_cur>=14)
        x_cur=0;
    else
        x_cur++;
    DrawChess(x_cur, y_cur, CURSOR);
}

if (RxBuffer==0x5A && win_flag==0) {//下棋
    DrawBack(3,9,EMPTY);
    if (board_state[x_cur][y_cur]==EMPTY) {

        if(status==CVP)
        DrawChess(x_cur, y_cur, 1-turn);
        else
        DrawChess(x_cur, y_cur, turn);

        board_state[x_cur][y_cur]=turn;
        board_record[BackTimes].x=x_cur;
        board_record[BackTimes].y=y_cur;
        BackTimes++;
        count=0;
        time0=0;
        if (turn==COMPUTER_PLAYER)
            step_flag=1;
        if (CheckWin(x_cur,y_cur,turn)==1) {
            xil_printf("\r\nHuman Player wins!\r\n");
            win_flag=1;
            DrawWinning(0, 1, EMPTY);
            if(status==CVP)
                DrawWinning(0, 1, 1-turn);
            else
                DrawWinning(0, 1, turn);

            if (status==CVP || status==PVC)
            {
                //money+=3;
                afterGame(1);
            }
        }
        else if (CheckBan(x_cur,y_cur,turn)==1){
            xil_printf("\r\nComputer Player wins!\r\n");
            win_flag=1;
            DrawWinning(0, 1, EMPTY);
            if(status==CVP)
                DrawWinning(0, 1, turn);
            else
                DrawWinning(0, 1, 1-turn);
            if (status==CVP || status==PVC)
            {
                //money--;
                afterGame(0);
            }
        }
        else {
            if (turn==HUMAN_PLAYER)
                turn=COMPUTER_PLAYER;
            else
```

```
            turn=HUMAN_PLAYER;
          xil_printf("\r\nComputer Player's turn!\r\n");
      }
    }
  }
  if (RxBuffer==0x29 && turn==HUMAN_PLAYER && win_flag==0) {//change mode 空格
    count=0;time0=0;

    if (status==PVP) {
      x_cur=7;
      y_cur=7;
      for (i=0; i<256; i++) {
          board_record[i].x=0;
          board_record[i].y=0;
              }
      InitializeGame(x_cur, y_cur);status=PVP;
      DrawStatus(0, 18, EMPTY,status);
      status=PVC;
      DrawNumber(level,6,18,0,1);
      DrawStatus(0, 18, COMPUTER_PLAYER,status);
    }
    else if(status==PVC) {
      x_cur=7;
      y_cur=7;
      for (i=0; i<256; i++) {
          board_record[i].x=0;
          board_record[i].y=0;
              }
      InitializeGame(x_cur, y_cur);status=PVP;
      DrawStatus(0, 18, EMPTY,status);
      status=CVP;
      DrawNumber(level,6,18,1,1);
      DrawStatus(0, 18, COMPUTER_PLAYER,status);
      turn=COMPUTER_PLAYER;
    }
    else if(status==CVP) {
      x_cur=7;
      y_cur=7;
      for (i=0; i<256; i++) {
          board_record[i].x=0;
          board_record[i].y=0;
              }
      InitializeGame(x_cur, y_cur);status=PVP;
      DrawStatus(0, 18, EMPTY,status);
      status=PVP;
      DrawStatus(0, 18, COMPUTER_PLAYER,status);
    }
  }
  if (RxBuffer==0x76) {//退出键
    x_cur=7;
    y_cur=7;
    for (i=0; i<256; i++) {
        board_record[i].x=0;
        board_record[i].y=0;
            }
    InitializeGame(x_cur, y_cur);status=PVP;
  }
  if (RxBuffer==0x2D) {//悔棋
    if(BackTimes>0){
     BackTimes--;
     x_cur=board_record[BackTimes].x;
     y_cur=board_record[BackTimes].y;
     board_state[x_cur][y_cur]=EMPTY;
     DrawChess(x_cur,y_cur,EMPTY);
     turn=1-turn;
     if(status==PVC)
     {
     BackTimes--;
     x_cur=board_record[BackTimes].x;
     y_cur=board_record[BackTimes].y;
     board_state[x_cur][y_cur]=EMPTY;
```

```
                DrawChess(x_cur,y_cur,EMPTY);
                turn=HUMAN_PLAYER;
                }
                DrawBack(3,9,turn);
                        }
                        }
        //}
      }


    }//page == 1

}//die win 均为 0

    if (die == 3)
      {
        die = 0;
        drawDie(0,0,0,2);//清空原先的颜色
        page = 0;//重新开始游戏
        pig = 0;
        grownPig = 0;
        food = 0;
        hp = 5;
        money = 0;
        cursor0 = 0;
        head = 0;
        tail = 0;
        pigs[head] = 0;

        drawBuy(cursor0);
        drawFoodNum(pig,food,hp,money,grownPig);
        //drawFood(1);//最左边 黑色

      }
    if (die == 2)
      {
        drawDie(0,0,3,2);
        drawFoodNum(pig,food,hp,money,grownPig);
        die++;
      }

    if (win0 == 3)
      {
        win0 = 0;
        drawWin(0,0,0,2);//清空
        page = 0;//重新开始游戏
        pig = 0;
        grownPig=0;
        food = 0;
        hp = 5;
        money = 0;
        cursor0 = 0;
        head = 0;
        tail = 0;
        pigs[head] = 0;
        drawBuy(cursor0);
        drawFoodNum(pig,food,hp,money,grownPig);
        //drawFood(1);//最左边 黑色
      }
    if (win0 == 2)
      {
        drawWin(0,0,2,2);//蓝色
        drawFoodNum(pig,food,hp,money,grownPig);
        win0++;
      }


}//keycount == 0
}  //else 非电脑下棋
}//while 1
```

3. C 语言之屏幕设计部分

```
void drawPixel(int x_pos, int y_pos, int turn,int page)
{
   int vga_input;
   vga_input=((x_pos)<<24)+((y_pos)<<16)+(page<<8)+turn;
   VGA_IP_mWriteReg(XPAR_VGA_IP_0_BASEADDR, 0, vga_input);
}

void drawLine(int x_pos, int y_pos, int turn,int page)//竖着 8 个像素
{
   int vga_input;
      vga_input=((x_pos)<<24)+((y_pos)<<16)+(page<<8)+turn;
      VGA_IP_mWriteReg(XPAR_VGA_IP_0_BASEADDR, 0, vga_input);
      vga_input=((x_pos)<<24)+((y_pos+1)<<16)+(page<<8)+turn;
      VGA_IP_mWriteReg(XPAR_VGA_IP_0_BASEADDR, 0, vga_input);
      vga_input=((x_pos)<<24)+((y_pos+2)<<16)+(page<<8)+turn;
      VGA_IP_mWriteReg(XPAR_VGA_IP_0_BASEADDR, 0, vga_input);
      vga_input=((x_pos)<<24)+((y_pos+3)<<16)+(page<<8)+turn;
      VGA_IP_mWriteReg(XPAR_VGA_IP_0_BASEADDR, 0, vga_input);
      vga_input=((x_pos)<<24)+((y_pos+4)<<16)+(page<<8)+turn;
      VGA_IP_mWriteReg(XPAR_VGA_IP_0_BASEADDR, 0, vga_input);
      vga_input=((x_pos)<<24)+((y_pos+5)<<16)+(page<<8)+turn;
      VGA_IP_mWriteReg(XPAR_VGA_IP_0_BASEADDR, 0, vga_input);
      vga_input=((x_pos)<<24)+((y_pos+6)<<16)+(page<<8)+turn;
      VGA_IP_mWriteReg(XPAR_VGA_IP_0_BASEADDR, 0, vga_input);
      vga_input=((x_pos)<<24)+((y_pos+7)<<16)+(page<<8)+turn;
      VGA_IP_mWriteReg(XPAR_VGA_IP_0_BASEADDR, 0, vga_input);
}

void drawLine2(int x_pos, int y_pos, int turn, int page,int num0,int num1,
         int num2,int num3, int num4, int num5, int num6,int num7)
{
   if (num0) drawPixel(x_pos,y_pos,turn,page);
   if (num1) drawPixel(x_pos,y_pos+1,turn,page);
   if (num2) drawPixel(x_pos,y_pos+2,turn,page);
   if (num3) drawPixel(x_pos,y_pos+3,turn,page);
   if (num4) drawPixel(x_pos,y_pos+4,turn,page);
   if (num5) drawPixel(x_pos,y_pos+5,turn,page);
   if (num6) drawPixel(x_pos,y_pos+6,turn,page);
   if (num7) drawPixel(x_pos,y_pos+7,turn,page);

}
```

4. C 语言之 AI 部分

```
int CheckWin(int x_pos, int y_pos, int turn)
{
      int i,j,count;
      //水平
      count=1;
      for (i=x_pos+1, j=y_pos; i<15; i++) {
                  if (board_state[i][j]==turn)
                              count++;
                  else
                              break;
      }
      for (i=x_pos-1, j=y_pos; i>=0; i--) {
                  if (board_state[i][j]==turn)
                              count++;
                  else
                              break;
   }
      if (count==5)
                  return 1;

      //垂直
      count=1;
      for (i=x_pos, j=y_pos+1; j<15; j++) {
                  if (board_state[i][j]==turn)
                              count++;
```

```
                else
                        break;
        }
        for (i=x_pos, j=y_pos-1; j>=0; j--) {
                if (board_state[i][j]==turn)
                        count++;
                else
                        break;
        }
        if (count==5)
                return 1;

        //左上至右下
        count=1;
        for (i=x_pos+1,j=y_pos+1; i<15 && j<15; i++,j++) {
                if (board_state[i][j]==turn)
                        count++;
                else
                        break;
        }
        for (i=x_pos-1,j=y_pos-1; i>=0 && j>=0; i--,j--) {
                if (board_state[i][j]==turn)
                        count++;
                else
                        break;
        }
        if (count==5)
                return 1;

        //左下至右上
        count=1;
      for (i=x_pos+1,j=y_pos-1; i<15 && j>=0; i++,j--) {
                if (board_state[i][j]==turn)
                        count++;
                else
                        break;
        }
        for (i=x_pos-1,j=y_pos+1; i>=0 && j<15; i--,j++) {
                if (board_state[i][j]==turn)
                        count++;
                else
                        break;
        }
        if (count==5)
                return 1;

        //no win
        return 0;
}

int CalculateValue(int board[][15], int turn, int x_pos, int y_pos)
{
        int i, j, count;
        int flag1, flag2;
        int value[4];

        flag1=DEAD;
        flag2=DEAD;
        count=1;
        for (i=x_pos+1, j=y_pos; i<15; i++) {
                if (board[i][j]==turn)
                        count++;
                else if (board[i][j]==EMPTY) {
                        flag1=LIVE;
                        break;
                }
                else {
                        flag1=DEAD;
                        break;
                }
        }
        for (i=x_pos-1, j=y_pos; i>=0; i--) {
```

```
		if (board[i][j]==turn)
			count++;
		else if (board[i][j]==EMPTY) {
			flag2=LIVE;
			break;
		}
		else {
			flag2=DEAD;
			break;
		}
}
value[0]=ValueTable(count, flag1, flag2);

flag1=DEAD;
flag2=DEAD;
count=1;
for (i=x_pos, j=y_pos+1; j<15; j++) {
		if (board[i][j]==turn)
			count++;
		else if (board[i][j]==EMPTY) {
			flag1=LIVE;
			break;
		}
		else {
			flag1=DEAD;
			break;
		}
}
for (i=x_pos, j=y_pos-1; j>=0; j--) {
		if (board[i][j]==turn)
			count++;
		else if (board[i][j]==EMPTY) {
			flag2=LIVE;
			break;
		}
		else {
			flag2=DEAD;
			break;
		}
}
value[1]=ValueTable(count, flag1, flag2);

flag1=DEAD;
flag2=DEAD;
count=1;
for (i=x_pos+1, j=y_pos+1; i<15 && j<15; i++, j++) {
		if (board[i][j]==turn)
			count++;
		else if (board[i][j]==EMPTY) {
			flag1=LIVE;
			break;
		}
		else {
			flag1=DEAD;
			break;
		}
}
for (i=x_pos-1, j=y_pos-1; i>=0 && j>=0; i--, j--) {
		if (board[i][j]==turn)
			count++;
		else if (board[i][j]==EMPTY) {
			flag2=LIVE;
			break;
		}
		else {
			flag2=DEAD;
			break;
		}
}
value[2]=ValueTable(count, flag1, flag2);

flag1=DEAD;
```

```
	flag2=DEAD;
	count=1;
	for (i=x_pos+1, j=y_pos-1; i<15 && j>=0; i++, j--) {
		if (board[i][j]==turn)
			count++;
		else if (board[i][j]==EMPTY) {
			flag1=LIVE;
			break;
		}
		else {
			flag1=DEAD;
			break;
		}
	}
	for (i=x_pos-1, j=y_pos+1; i>=0 && j<15; i--, j++) {
		if (board[i][j]==turn)
			count++;
		else if (board[i][j]==EMPTY) {
			flag2=LIVE;
			break;
		}
		else {
			flag2=DEAD;
			break;
		}
	}
	value[3]=ValueTable(count, flag1, flag2);

	if (turn==COMPUTER_PLAYER)
		return value[0]+value[1]+value[2]+value[3];
	else
		return -(value[0]+value[1]+value[2]+value[3]);
}
```