

数字系统设计课程设计报告

--基于FPGA的游戏设计

学 校: 中山大学

院 系: 数据科学与计算机学院

专 业: 软件工程（移动信息工程）

年 级: 15级

指导老师: 李国桢，保研翔

|  |  |  |
| --- | --- | --- |
| **学号** | **姓名** | **分工（人均分配25%）** |
| 15352383 | 杨鑫 | 负责U6 U7(较繁琐)模块 |
| 15352218 | 林燕娜 | 负责U4 U5(较繁琐)模块 |
| 15352458 | 庄博伟 | 负责U3 U8模块，报告主编辑 |
| 15352457 | 朱裕章 | 负责U1 U2模块，报告副编辑 |

二零一六年十二月二十四日

摘要

本次课程设计采用了 Xilinx 公司的 Basys3 系列的 XC7A35T-ICPG236C FPGA 来计算运行贪吃蛇游戏并驱动VGA显示游戏过程，实现方式为采用 verilog HDL　语言编写，主要涉及组合逻辑电路、时序逻辑电路（状态机）、数字模块、VGA显示等相关知识。本次课程设计的系统电路设计主要采用模块化的思想实现。

关键字：FPGA 贪吃蛇 Verilog HDL VGA 状态机

Abstract

This course is designed to calculate and run the game of the Snake and to drive the VGA to display the game, which is based on XC7A35T-ICPG236C FPGA from Xilinx, Basys3. This course is designed using Verilog HDL language It mainly involves combinational logic circuit, sequential logic circuit Machine, digital module, VGA display and other related knowledge. The curriculum design of the system circuit design use a modular idea to achieve.

Keywords: FPGA Snake Game Verilog HDL VGA sequential logic circuit Machine

目录

[1、前言与设计背景 4](#_Toc470509303)

[2、系统分析与方案设计 5](#_Toc470509304)

[（1）系统电路设计 6](#_Toc470509305)

[（2）时钟分频模块设计 6](#_Toc470509306)

[（3）按键消抖模块设计 8](#_Toc470509307)

[（4）游戏流程控制模块设计 9](#_Toc470509308)

[（5）苹果生成模块设计 11](#_Toc470509309)

[（6）蛇身控制模块设计 12](#_Toc470509310)

[（7）显示综合模块设计 14](#_Toc470509311)

[（8）VGA控制模块设计 16](#_Toc470509312)

[（9）顶层模块 19](#_Toc470509313)

[4．软件调试与结果展示 23](#_Toc470509314)

[5.课程设计总结与心得 25](#_Toc470509315)

# 1、前言与设计背景

《数字系统设计》是一门技术性、应用性很强的学科，这门课程将理论与实验相互结合进行，让我们逐步掌握了基本的数字逻辑设计概念。通过此门课程我们学习了从组合电路，逻辑电路到触发器，最后接触了VHDL和Verilog两种主要硬件描述语言，并结合实验课完成了一些基本例子的实现。本课程设计的目的其实就是让我们在理论学习的基础上，通过完成一个涉及时序逻辑、组合逻辑、相关输出的，具有实用性、趣味性的小系统设计，使我们不但能够将课堂上学到的理论知识与实际应用结合起来，而且能够对分析、解决实际的数字电路问题进一步加深认识，为今后能够独立进行某些数字应用系统的开发设计工作打下一定的基础。

而本次课程采用的就是Verilog HDL语言。Verilog HDL语言是一种广泛应用的硬件描述语言，可以用在硬件设计流程的建模、综合和模拟等多个阶段。随着硬件设计规模的不断扩大，应用硬件描述语言进行描述的FPGA结构，成为设计专用集成电路和其他集成电路的主流。

在布置课程设计的时候，李国桢老师说明了课程设计的评分标准为：课程设计涵括本学期内容越广为优秀，并且小组人数（上限为4人）越多课程设计也应越有难度。在综合考虑课程内容、课程难度、知识拓展和兴趣挑战，本小组在课程设计选题讨论会上一致决定课程设计内容为编写一个小游戏并且用VGA显示作为知识拓展。考虑到实现难度以及 Verilog HDL 实现可能性，本小组最终决定利用 Verilog HDL 语言编写实现经典小游戏贪吃蛇。

# 2、系统分析与方案设计

贪吃蛇是经典小游戏，本设计采用VGA显示的方式将游戏展现出来。游戏中玩家通过四个按键控制蛇的身体上下左右移动来吃屏幕中出现的苹果，苹果是随机出现的。当蛇吃的一个苹果时，蛇身体变长一个单位同时会有另一个苹果出现。如果蛇身达到一定的长度（玩家获得的分数）时，游戏胜利；如果蛇头撞墙或者撞到自己身体，则游戏失败。

所要设计的贪吃蛇游戏基于Xilinx 公司的 Basys3 系列的 XC7A35T-ICPG236C FPGA芯片为硬件，采用Verilog HDL语言编写程序。在本设计中，有个5按键，分别是上、下、左、右、复位/游戏开始按键。它们都是输入信号，输出是VGA，在电脑屏幕实现VGA显示汉字，游戏画面。其系统设计结构框图如图所示：

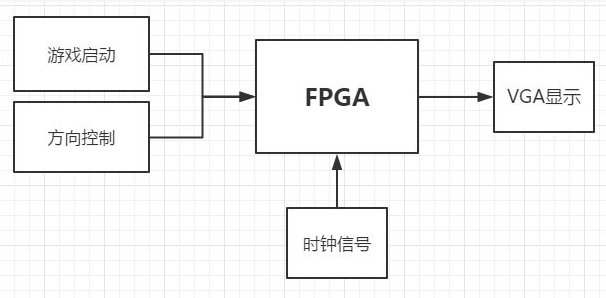


图2.1 贪吃蛇游戏系统设计结构框图

3．硬件电路设计

# （1）系统电路设计

硬件电路设计包含以下几个模块：

U1：Clock\_divide 时钟分频模块

U2：Button\_Scaning 按键消抖模块

U3: Game\_Control 游戏流程控制模块

U4：Eating\_Apple 苹果生成模块

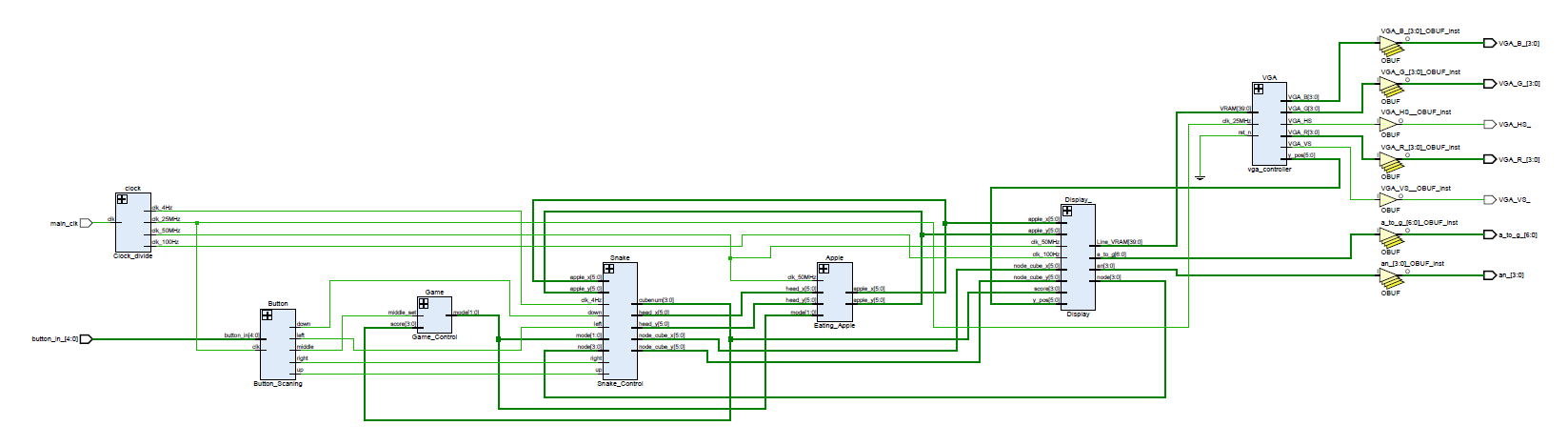
U5：Snake\_Control 蛇身控制模块

U6：Display 显示综合模块

U7：vga\_controller VGA控制模块

U8: Top 顶层模块

完成后整个电路的RTL view如下图所示：



# （2）时钟分频模块设计

利用了触发器、寄存器，以时钟的一个上升沿作为触发条件，用寄存器记录时钟上升沿的次数。当时钟上升沿次数达到100M次，拉高记录秒的信号电平，输出一个一秒时间的高电平，否则拉低该信号电平。该模块中将100MHz时钟频率划分了50MHz、25MHz、119Hz、4Hz，分别主要用作系统主时钟频率、显示带宽（800\*521\*60）、数码管扫描、控制贪吃蛇的运动速度。

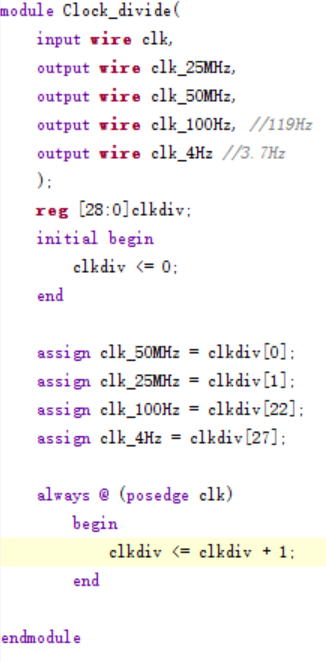


图3.2.1 时钟分频模块

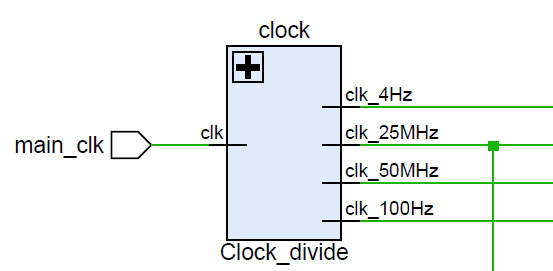


图3.2.2 时钟分频模块硬件设计图

# （3）按键消抖模块设计

原理：在单片机时代，对于按键我们可以选择硬件消抖和软件消抖，软件消抖是采用if语句和延时语句多次检测按键处电平的变化，浪费资源而且还有一定概率出错。硬件消抖的原理就是在按键旁边焊上一个电容。这样当按键变化时，传输到CPU的信号不会马上发生电平变化，而是伴随一个充电的过程。也就是说当按键电平真正变化时，按键已经经过了抖动阶段，当电容容值选取合理时，就能达到比较完美的消抖效果。在使用 FPGA 时，我们使用的是硬件描述语言，完成的是数字电路的设计。只要用数字电路来取代这个简单的电容，也就能实现硬件防抖的效果。

实现：利用寄存器模拟电容充电和放电的过程，当按键寄存器还没达到设定数值并且按钮处于高电平状态，“电容”充电，否则“电容”放电；如果寄存器的值达到一定的数值，就输出一个代表按钮被按下的高电平信号，否则输出低电平信号。

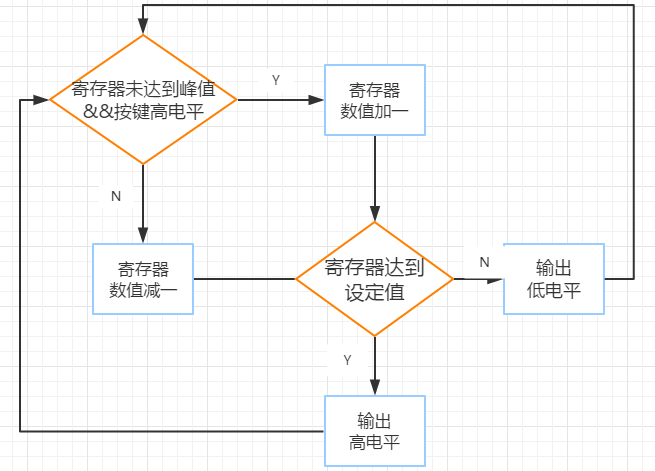


图3.3.1按键消抖模块实现

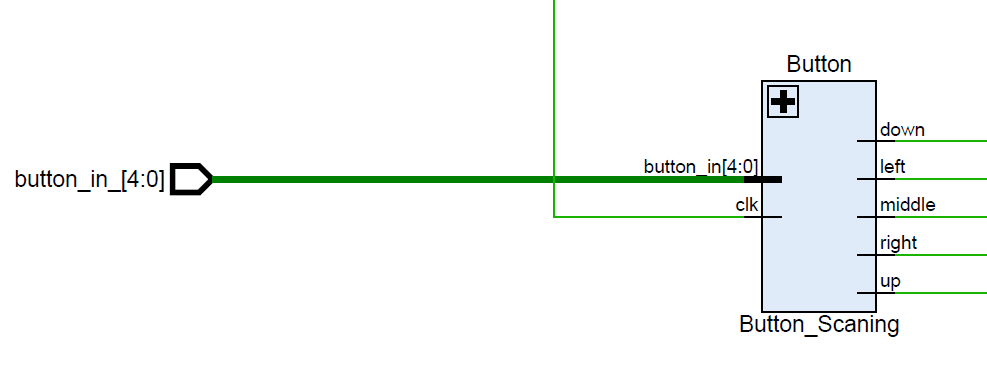


图3.3.2按键消抖模块硬件设计图

# （4）游戏流程控制模块设计

游戏控制模块（Game\_Control）的设计基于有限状态机设计技术。状态机就是状态转移图，状态在不同的条件下跳转到自己或不同状态的图。就理论而言，任何时序模型都可以归结为一个状态机。在许多方面，有限状态机都有其难以超越的优越性，这主要表现在一下几个方面：

（1）高效的顺序控制模型。状态机克服了纯硬件数字系统顺序方式控制不灵活的缺点。状态机的工作方式是根据控制信号按照预先设定的状态进行顺序运行的。

（2）容易利用现成的EDA工具进行优化设计。由于状态机构件简单，设计方案相对固定，特别是可以作一些独具特色的规范固定的表述，使得这一切为HDL综合器尽可能自动的发挥其强大的优化功能提供便利条件。

（3）系统性能稳定。状态机容易构成性能良好的同步时序逻辑模块，这对于对付大规模逻辑电路设计中令人深感棘手的竞争冒险现象无疑是一个上佳的选择。

（4）设计实现效率高。与HDL的其他描述方式相比，状态机的表述形式相对固定却又灵活多样，且程序层次分明，结构清晰，易读易懂，排错修改便捷。

（5）高速性能。在高速通信和高速控制方面，状态机更有其巨大的优势。

游戏控制模块从其他模块中获得信号，从而决定游戏状态，同时向其他信号发出控制信号。由程序执行流程图可知，共4种状态：

状态一：游戏开始前，按下开始游戏键进入状态二。

状态二：游戏正在进行，达成某一条件进入状态三或状态四。

状态三：游戏胜利，按下重置键返回状态二。

状态四：游戏失败，按下重置键返回状态二。

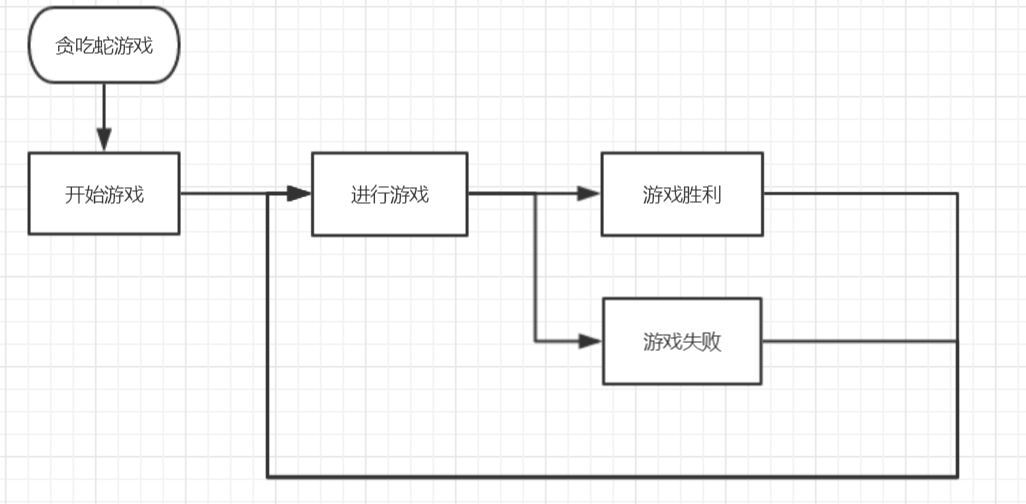


图3.4.1 贪吃蛇程序执行流程图

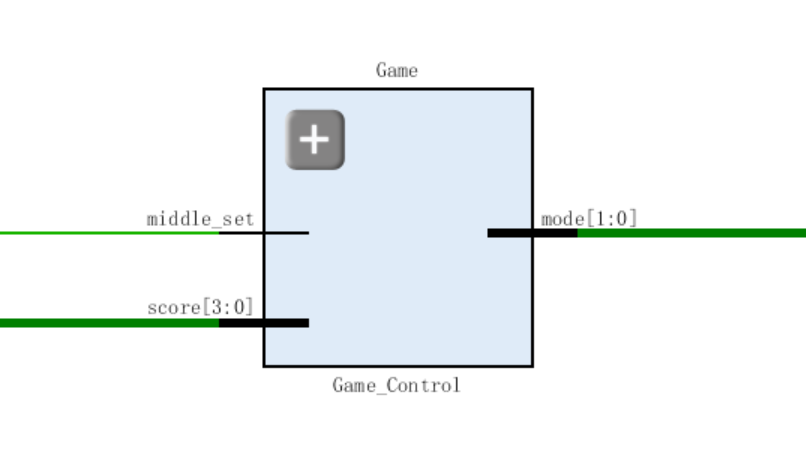


图3.4.2 游戏流程控制模块硬件设计图

# （5）苹果生成模块设计

苹果生成模块，在游戏一开始以及贪吃蛇吃到苹果后随机生成苹果。每个时钟周期 random\_num 都在变， 而我们吃下苹果的时刻却因走法、 按键的时间等有所不同，所以不同时刻吃下苹果后下一个苹果出现的地方近似随机。

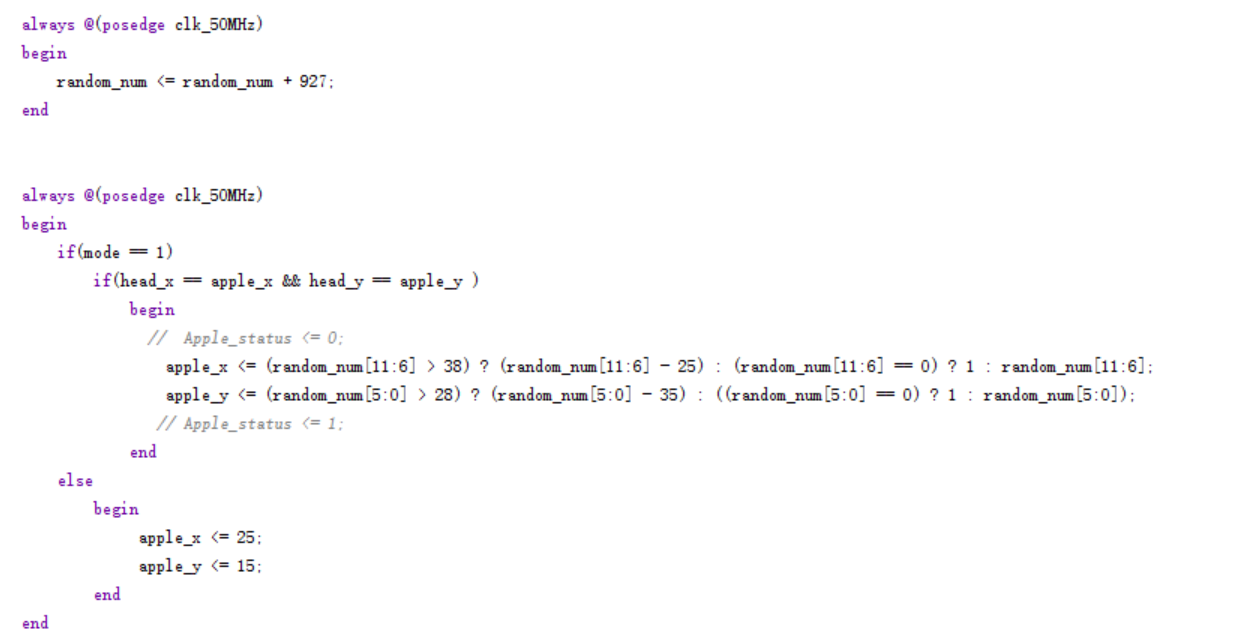


图3.5.1 苹果生成模块关键代码

注\*：考虑到屏幕分辨率是640\*480，像素为16\*16，所以格子为40\*30；

墙为第一行、最后一行、第一列、最后一列；所以生成苹果的时候必须在墙

的内部，所以apple\_x∈[1,38],apple\_y∈[1,28]

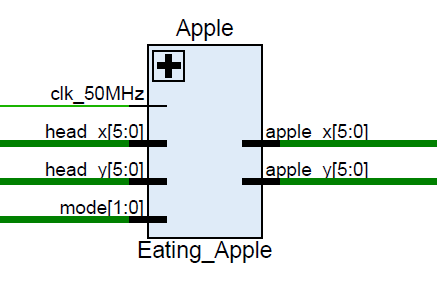


图3.5.2 苹果生成模块硬件设计图

# （6）蛇身控制模块设计

在贪吃蛇游戏进行过程中，每0.5s蛇移动一次，并且，通过读控制键key的值，实现上下左右四个方向的移动，并且判断是否死亡。

A.蛇最长的长度为16，利用一个16位的数组，存储蛇的每一节节点所在的坐标。声明：

reg [5:0] cube\_x[15:0];

reg [5:0] cube\_y[15:0];

reg[5:0]一共六位，存储每一个点的坐标

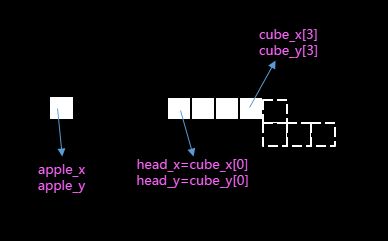


图3.6.1 蛇的构想模型

B.保持蛇的移动：

保持蛇的移动，通过数组cube\_x,cube\_y的变化来实现，在本次代码中，实现的是数组的每一位向后移动，而蛇的头节点坐标，就是cube\_x[0],cube\_y[0]根据下一步所要进行的方向进行变化；

数组移动的部分代码：

Begin

cube\_x[1]<=cube\_x[0];

cube\_y[1]<=cube\_y[0];

cube\_x[2]<=cube\_x[1];

cube\_y[2]<=cube\_y[1];

cube\_x[3]<=cube\_x[2];

cube\_y[3]<=cube\_y[2];

.....

cube\_x[15]<=cube\_x[14];

cube\_y[15]<=cube\_y[14];

end

在这个数组移动的过程中，很容易发现，蛇的头节点是没有赋值的，因为蛇的头节点需要根据当前状态进行处理。

头节点处理的部分代码：

case(direct) //移动蛇头

UP:

cube\_y[0] <= cube\_y[0] - 1;

DOWN:

cube\_y[0] <= cube\_y[0] + 1;

LEFT:

cube\_x[0] <= cube\_x[0] - 1;

RIGHT:

cube\_x[0] <= cube\_x[0] + 1;

endcase

C.判断是否为死亡状态

蛇死亡有以下情况：

①撞墙；

②咬伤自己

这部分较为简单，就不贴代码了。

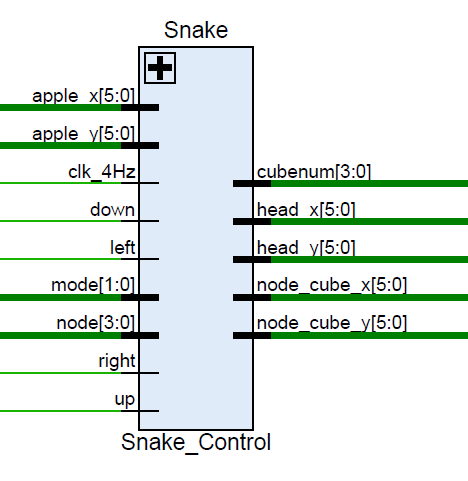


图3.6.2 蛇身控制模块硬件设计图

# （7）显示综合模块设计

在我们本次课程设计中，显示综合模块的主要功能如下：

* 控制七段管输出当前游戏分数
* 从蛇身控制/苹果生成模块中读取画
* 面信息，生成行显存，输出到VGA驱动模块

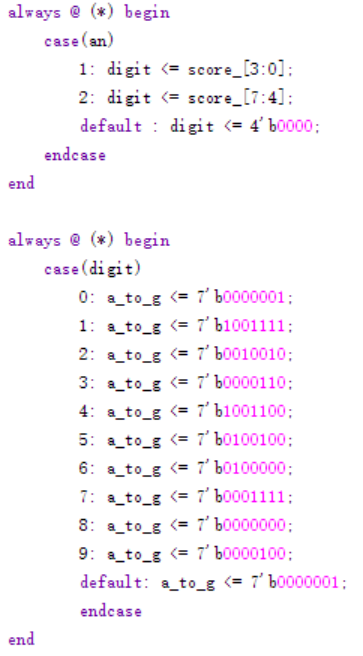
 

图3.7.1 显示综合模块I/O / 七段管的部分配置代码

在此前的实验课程设计中，我们已经较为熟练地掌握了Basys\_3开发板内置的七段LED数码管的控制，类似地，我们取用其中两位作为游戏的得分面板。

在这一模块中，更为重要的工作是生成“行显存”供VGA驱动模块读取，思路如下：通过从VGA驱动模块中读入逻辑纵坐标y\_pos，读入蛇身节点信息与苹果坐标信息，经由逻辑比较将行显存Line\_VRAM相应的某一位设置为0/1即可。



图3.7.2 生成行显存的代码块

此处较为精髓的部分为蛇身的遍历方式：通过一个变化频率高于远高于逻辑行刷新频率的寄存器j进行蛇身坐标的遍历。随着寄存器j数值改变，蛇身控制模块会逐一输出蛇身每一个节点的逻辑坐标，经过比较后更改行显存的值即可。

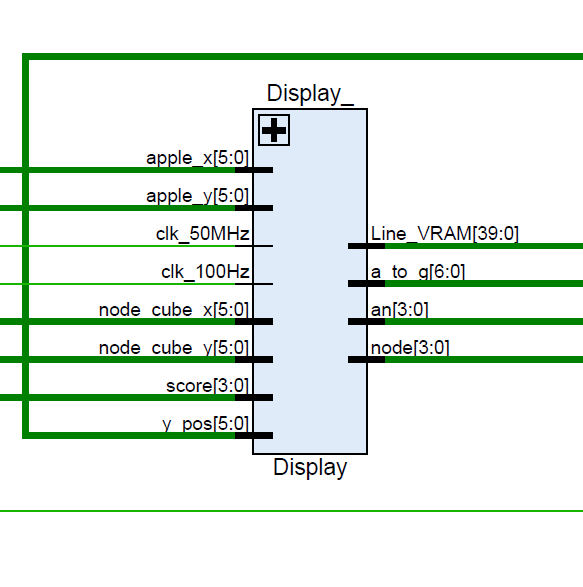


图3.7.3显示综合模块硬件设计图

# （8）VGA控制模块设计

本次课程设计中，我们使用了Basys\_3开发板的VGA端口作为游戏的画面输出，因此需要设计一个VGA驱动模块用于输出图像。我们的VGA驱动模块的主要功能有以下两点：

* 经由VGA端口向显示器输出行（Horizontal\_Sync）与场同步信号（Vertical\_Sync）；
* 读取由显示综合模块生成的显存，转化为相应的色彩信号VGA\_R、VGA\_G、VGA\_B输出到显示器。

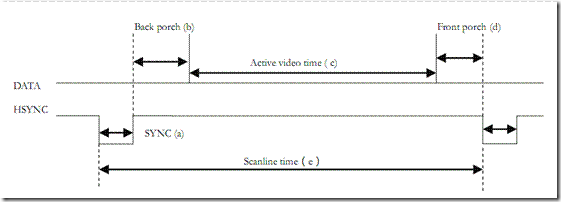


图3.8.1行同步时序

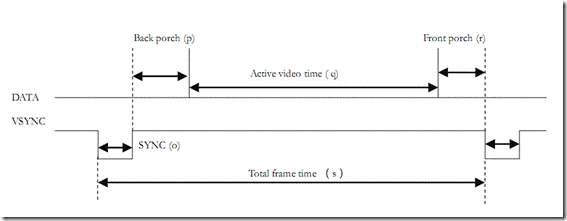


图3.8.2场同步时序

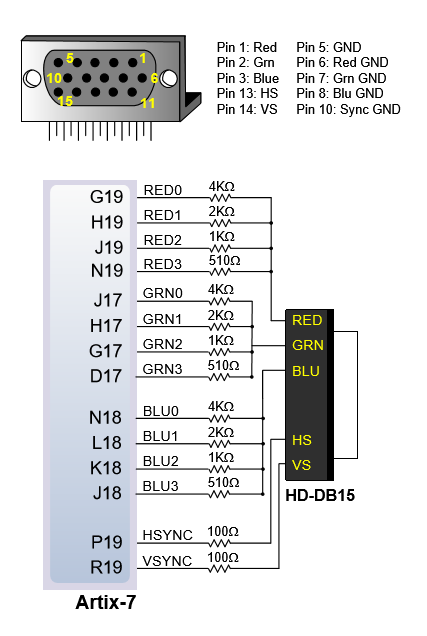
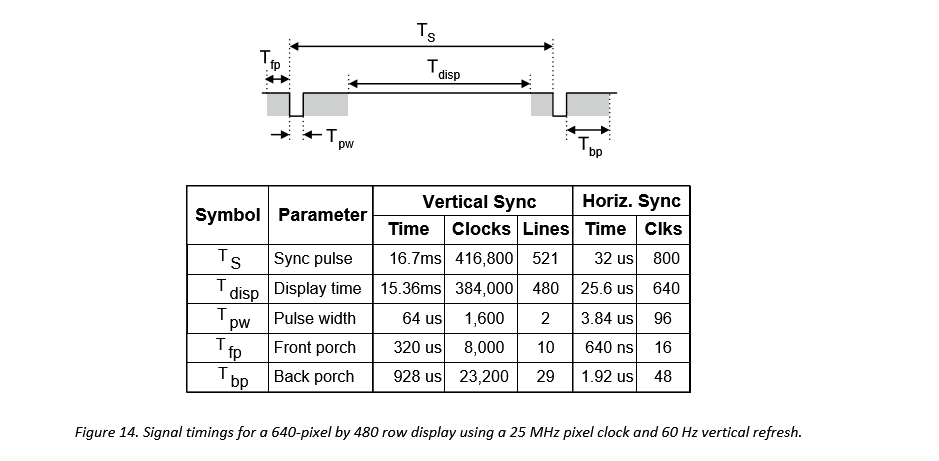


图3.8.3 640\*480分辨率，60Hz输出时序表 / Basys\_3开发板VGA端口图

在查阅资料，根据游戏设计目标斟酌后，我们选择了640\*480分辨率作为最终输出的分辨率，经由分频模块获取25MHz的时钟作为VGA驱动模块的主时钟。根据行场同步时序表与VGA显示原理，通过计数生成行场同步脉冲信号VGA\_HS和VGA\_VS。

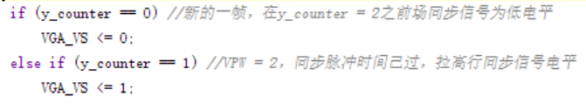
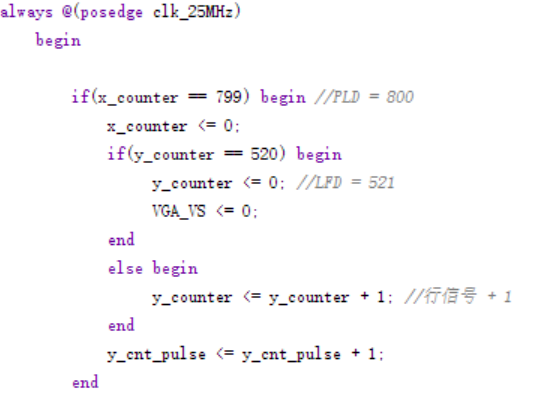


图3.8.4 生成行场同步信号的部分描述代码

出于减小游戏资源消耗的目的，我们将16\*16的实际像素块视为单位逻辑像素，因此VGA驱动模块也将同时负责实际像素与逻辑像素之间的数值转化，在游戏逻辑上，我们将画面划分成了一个40\*30的区域。

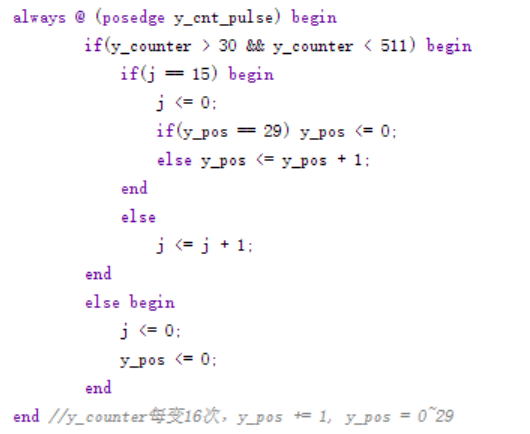
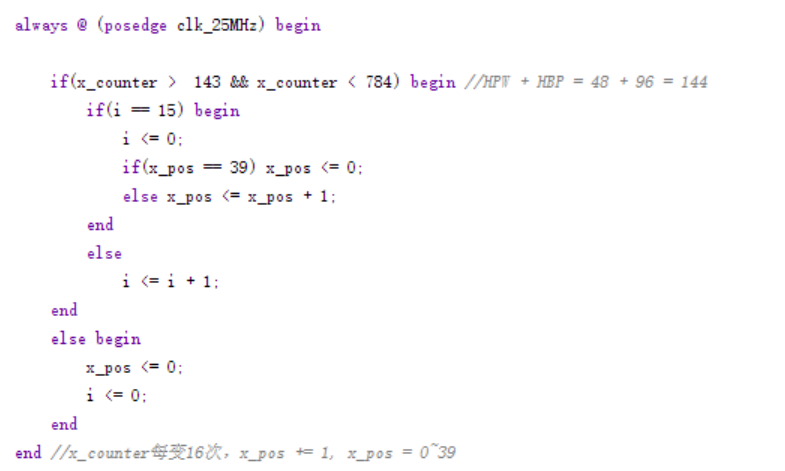


图3.8.5 通过计数实现实际像素与逻辑像素的转化

在Verilog模块中，我们将实际像素坐标记为x\_counter与y\_counter，转化后的逻辑像素坐标记为x\_pos与y\_pos。

在测试多色彩输出时出现了预料之外的错误，更换多台显示器后发现显示效果不可控。出于稳定性考虑，本次课程设计中VGA驱动模块仅输出黑色（RGB\_0，0，0）与白色（RGB\_255，255，255）构成画面。

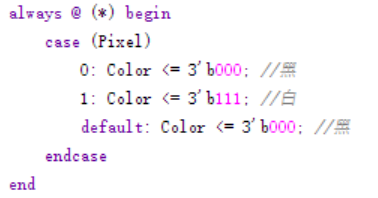
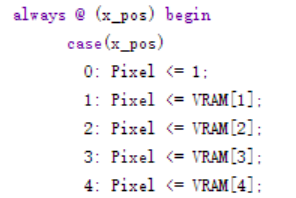


图3.8.6 通过读取行显存VRAM，将预存的图像输出到屏幕上

因为只有两种色彩，即只有0/1两种状态，每一行的显存VRAM只需要30位就可以记录下每一行的色彩信息。通过读取对应像素处VRAM的数值，VGA驱动模块就能将预存在VRAM中的信息转化为实际图像输出了。

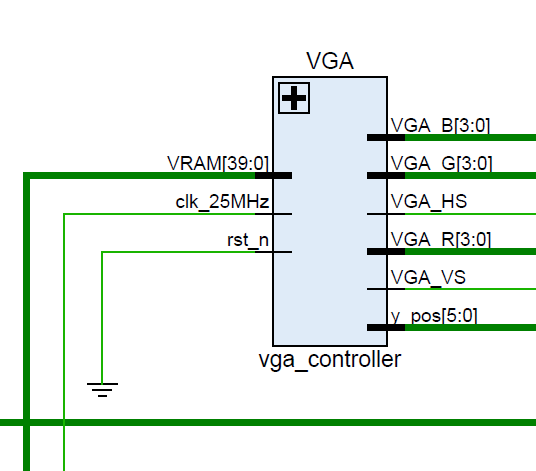


图3.8.7 VGA控制模块硬件设计图

# （9）顶层模块

将上述的底层模块设计添加到顶层模块设计，实现顶层文件调用其他模块。

module Top(

input wire main\_clk, //主时钟 100MHz

input wire [4:0] button\_in\_, //五颗按钮：中、上、下、左、右

output wire [6:0] a\_to\_g\_, //七段管

output wire [3:0] an\_, //七段管

output wire VGA\_HS\_,//行同步信号

output wire VGA\_VS\_,//场同步信号

output wire [3:0] VGA\_R\_,

output wire [3:0] VGA\_G\_,

output wire [3:0] VGA\_B\_

);

wire clk\_25MHz\_;

wire clk\_50MHz\_;

wire clk\_4Hz\_;

wire clk\_100Hz\_;

wire [1:0] mode\_;

wire [3:0] score\_;

wire middle\_;

wire up\_;

wire down\_;

wire left\_;

wire right\_;

wire [79:0] VRAM\_; //行显存

wire [5:0] y\_pos\_; //行指针

wire [5:0] apple\_x\_;

wire [5:0] apple\_y\_;

wire [5:0] head\_x\_;

wire [5:0] head\_y\_;

wire [5:0] node\_cube\_x\_;

wire [5:0] node\_cube\_y\_;

wire [3:0] node\_;

Clock\_divide clock(

.clk( main\_clk ), //100MHz

.clk\_25MHz( clk\_25MHz\_ ),

.clk\_50MHz( clk\_50MHz\_ ),

.clk\_100Hz( clk\_100Hz\_ ),

.clk\_4Hz( clk\_4Hz\_ )

);

Button\_Scaning Button(

.clk( clk\_25MHz\_ ), //25MHz

.button\_in( button\_in\_ ),//

.middle( middle\_ ), //0,消抖后的按键

.up( up\_ ), //1

.down( down\_ ),//2

.left( left\_ ),//3

.right( right\_ )//4

);

Game\_Control Game(

.mode( mode\_ ), //输出游戏状态

.score( score\_ ), //传入游戏分数

.middle\_set( middle\_ ) //开始游戏

);

Display Display\_(

.clk\_100Hz( clk\_100Hz\_ ), //用于七段管扫描

.clk\_50MHz( clk\_50MHz\_ ),

.score( score\_ ), //传入游戏分数

.y\_pos( y\_pos\_ ), //传入行指针

.node\_cube\_x( node\_cube\_x\_ ), //传入蛇身节点x坐标

.node\_cube\_y( node\_cube\_y\_ ), //传入蛇身节点y坐标

.node( node\_ ), //传出蛇身节点位置

.apple\_x( apple\_x\_ ),//传入苹果x坐标

.apple\_y( apple\_y\_ ),//传入苹果y坐标

.Line\_VRAM( VRAM\_ ), //传出行显存

.a\_to\_g ( a\_to\_g\_ ),

.an ( an\_ )

);

vga\_controller VGA(

.clk\_25MHz( clk\_25MHz\_ ), //25MHz

.rst\_n( ), //复位信号，rst\_n == 1时画面全黑

.VRAM( VRAM\_ ), //从另一个模块传过来的图像信息, 每2bits记录一个坐标的色彩信息

.y\_pos( y\_pos\_ ), //行信号传给Display模块

.VGA\_HS( VGA\_HS\_ ),//行同步信号

.VGA\_VS( VGA\_VS\_ ),//场同步信号

.VGA\_R( VGA\_R\_ ),

.VGA\_G( VGA\_G\_ ),

.VGA\_B( VGA\_B\_ )

);

Snake\_Control Snake(

.clk\_4Hz( clk\_4Hz\_ ),

.mode( mode\_ ), // 传入游戏状态

.up( up\_ ),

.down( down\_ ),

.right( right\_ ),

.left( left\_ ), // 四个按钮状态

.apple\_x( apple\_x\_ ), //传入苹果x坐标

.apple\_y( apple\_y\_ ), //传入苹果y坐标

.node( node\_ ), //传入蛇身节点位置

.cubenum( score\_ ),//得分，最大15

.head\_x( head\_x\_ ), //传出蛇头x坐标

.head\_y( head\_y\_ ), //传出蛇头y坐标

.node\_cube\_x( node\_cube\_x\_ ), //传出蛇身节点x坐标

.node\_cube\_y( node\_cube\_y\_ ) //传出蛇身节点y坐标

);

Eating\_Apple Apple(

.clk\_50MHz( clk\_50MHz\_ ),

.mode( mode\_ ), //传入游戏状态

.head\_x( head\_x\_ ), //传入蛇头x坐标

.head\_y( head\_y\_ ), //传入蛇头y坐标

.apple\_x( apple\_x\_ ), //传出苹果x坐标

.apple\_y( apple\_y\_ ) //传出苹果y坐标

);

endmodule

至此，硬件语言设计已基本完成。

# 4．软件调试与结果展示

在初始的计划制定中，我们预想达到的游戏界面效果如下：

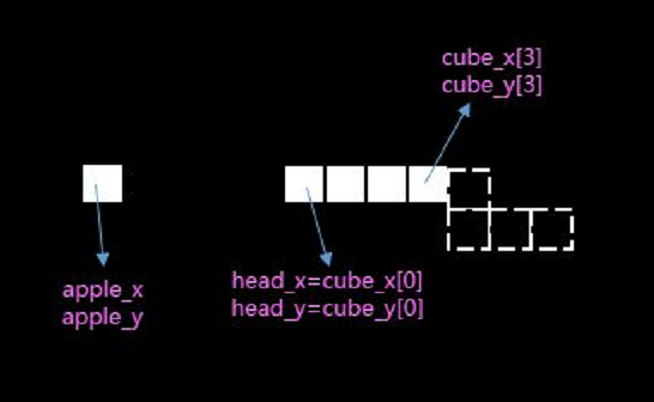


图4.1 预期效果

而经过设置端口接入，连接VGA端口，接入显示屏，将代码综合，生成编程文件后，烧入板子运行程序。进行了一系列的软件调试后，得到最终效果。

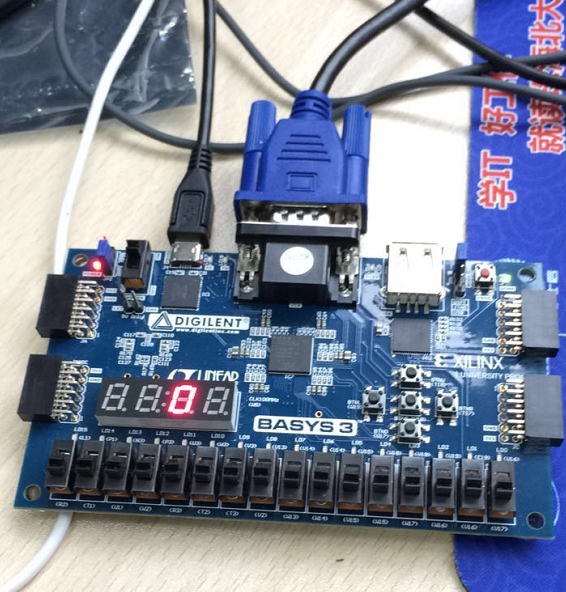


图4.2 FPGA连接情况



图4.2 游戏进行时画面

数码管记录得分，四面为墙，上面的一块为苹果，下面的为蛇的部分，总体来说达到预期结果。

# 5.课程设计总结与心得

# 1．设计中遇到的问题及解决办法

（1）万事开头难，起初对贪吃蛇游戏的设计以及FPGA转VGA毫无头绪，就算学习了Verilog语言的相关知识，也不知代码从何打起。即无法在大局上把握一个相关设计应有的过程。

解决办法：多与同学老师交流设计方法与构思，必要时可以搜索相关资料，大学中的学习不仅仅包括独立完成，借鉴模仿和交换经验也是学习的一种必要方式。

（2） 编程时，最常见也是最头疼的就是在各个模块调试的时候显示句子无定义，导致程序无法编译，但整个模块的语法没有错误。

解决办法：将各个模块之间调试有先后顺序，且将所有模块调试放在一个文件夹存放，以致可以相互调用。

（3）经常导致语法错误，如：“；”没有写上，变量类型没有预先标明，前后变量名字由于缺少一个或多一个字母而导致出错。

解决办法：对照错误，认真检查程序，看哪个地方的标点，变量没有写上或标明。

（4）进行编译或输出结果时，经常得到的不是预想中的结果。

解决办法：弄明白该模块实现什么功能，调整好输入变量进行调试，不能完全采用随机变量。在方面观察的情况下，可以采用高电平或低电平代替。将输出的结果与预期结果对比，找出不同之处，针对该处的模块进行细节修改调试，直到实现想要的结果。

# 2．完成设计后总结的经验

（1）进行一项较复杂的数字系统设计进要行充分的方案论证，不可盲目就动手去做，要充分了解我们将要进行的内容，制定一项大概的计划会帮助我们更快解决问题完成设计。

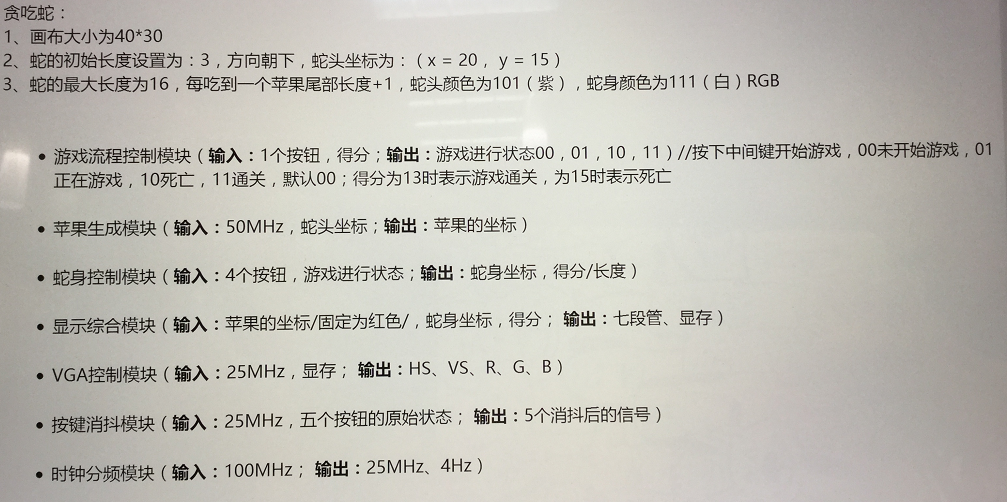


图5.3.1初步讨论规划

（2）实验中对每一个细节部分都要全面思考，要对特殊情况进行处理，因为对于这种完整系统的设计，会有许多的分支情况，而我们就需要理清头绪，认真列出每一种可能出现的情况，再对每一种情况进行相应的处理即可。

（3）在编写 Verilog 语言过程中，很关键的是考虑硬件电路的时序问题，尤其是always 块语句的并行运行和各个块之间的逻辑关系。在设计按钮消抖模块的时候，尝试了多种实现方式，体会到 Verilog 逻辑之间的强关联性。在处理判断按钮是否被按下（作为 always 语句的敏感条件）时，多敏感条件往往起不到理想中的效果，而在设计过程中发现，将敏感条件进行或位运算之后复制给一个 wire 型变量，并且用这个变量的上升沿作为敏感条件，会收到更为理想的效果。

（4）遇到问题，要顺藤摸瓜，分析清楚，不可胡乱改动，每做一次改变都要有充分的理由。我们在debug时，每次只改变一种变量，再将结果与之前的结果进行比对，就能看出此变量在模块中的作用。切忌一次性改变过多变量，这样只会让自己混淆。

（5）模块化设计方法的优点在于其简洁性，但是在实验设计中也发现，在实验最终结果确定之前，要尽量减少模块重叠嵌套。

（6） 遇到问题花了很长时间没有解决掉，我们也可以搜索相关资料解决问题，请教师兄与老师，不需拘泥与自己埋头苦干。

# 3．心得体会

杨鑫：

从一开始，我们就打算挑战难度稍微高一些的VGA显示实现。在考虑诸多因素之后，我们才最终决定了做贪吃蛇这一经典游戏，也算作对早期电子游戏的致敬。在本次课程设计中，从讨论实现方案到实际编程，我们都充分运用了在设计时钟的实验课程中汲取到的经验，查阅了不少资料，在尝试与失败中一步步摸索，才终于完成了这次课程设计。在实现游戏的过程中，我曾有气馁，有困倦，但幸运的是，我有几位好队友和我一起努力发现并解决问题。  
 最终我们的小蛇在显示屏上灵活地动起来的那一刻，我觉得一切都值了。

林燕娜：

没想到自己真的能做出一个小游戏，满满的成就感。

很感谢整个小组的通力合作。小组在尝试了vga多次黑屏，多次颜色模块不如自己所设想的艰辛之后，通过不断调试以及舍弃一些颜色后，最后终于有了结果。  
 在实现这项project之前，查了很多相关微博、论文、资料等，努力去理解一个游戏是怎么实现的，并努力学习其中的硬件思想、逻辑。其实，这次project真正的难点，在于vga的实现。一般的笔记本是没有vga的相关输出接口，所以只能跑去实验室，晚上也只能跑去研究生师兄的实验室。脱离了实验室，就不能看到输出的效果，就不能调试代码。这给我们带来了莫大的不便，也耽误了project的进程。  
 另外，vivado每次运行都要10几分钟，运行时间非常长，这使效率变得非常低。很多时候，只是改变一个语句，就得等10几分钟后才看到效果。等待的时间远远大于编程时间。  
 但是，一切一切的困难最终过去了，经过一个多月的努力，由于那一直消除不掉的bug，真的很想直接放弃。但是，不到最后一刻，坚决不放弃的我们，获得了相应的奖励，终于debug掉，实现了自己的设想。  
 最好莫过于有付出有回报。

朱裕章：

初期进行电路设计的时候，我们并没有重视电路原理图的设计，只是根据设想方案和预期目标进行电路描述，这使得我们在调试硬件程序的过程中，我们常常遇到“牵一发而动全身”的情况，就是一处看似无关紧要的代码改动，则其他模块的运行也跟着改变了。最后经过尝试，发现是在编写代码的过程中没有很好地考虑时序问题，而在电路硬件描述设计中，时序问题和逻辑问题是贯穿整个电路的关键，而它又常常被我们忽视，这也许是软件编程上养成的惯性思维。而这次课程设计，恰恰正好和软件编程思维有很大的冲突，这也正好给我们脑子里的“思维习惯”进行纠正和“分区”。

在课程设计完成的瞬间，突然觉得在成就感之中夹杂这心酸和压力的释放。在回顾进行课程设计过程小组整体遇到的问题困难的时候，发现我们更多的程序问题在于对 Verilog HDL编程语言的不熟悉，包括语法要求和语句执行结果，比如说是case语句的执行，always 语句的敏感条件的要求和执行过程等等。由于语言的不熟悉，在调试和debug过程中带来了很多的困惑和不解。所以，在进行一个项目设计之前，应该对其要求的背景知识和设计要求知识有充分的了解，不要在“战争中学习”，毕竟如果没有充足的知识背景，在后期修正项目设计中会浪费更多的时间精力。

庄博伟：

通过这次课程设计，我们感觉在多方面都有所提高。通过这次课程设计，综合运用本专业所学课程的理论知识，从而培养和提高了我们独立解决问题，实现各个功能的能力，巩固与扩充了Verilog等课程所学的内容，掌握设计的方法和步骤，让我们独立思考的能力也有不少的提高。同时，此次课程设计中自顶向下的设计理念也深刻烙印在我们的脑中，让我对所学的理论知识有了更深刻的理解。

磕磕绊绊这么久，最后成功的一刻，令人沉醉。

这是我们第一次通过所学知识独立完成一次完整的课程设计，让我们受益匪浅，也让我们真正明白理论与实践相结合的重要性。通过具体实践才能让自己清楚哪些知识已经掌握，哪些知识仍需巩固加强。但正所谓坚持就是胜利，要想取得成功，必须要有努力付出，这样所取得的结果才更有意义。

在此课程设计中，不仅实际动手能力有所提高，对设计过程的了解逐渐加深，更进一步激发了我们对专业知识的兴趣，并能结合实际存在的问题在专业领域内进行更深入的学习。

附录：

Gluttonous Snake V1.0