**中国科学技术大学计算机学院**

**《数字电路实验》报告**



实验题目：贪吃蛇游戏

学生姓名：Ouedraogo Ezekiel B.

学生学号：PL19215001

完成日期：1/12/2021

计算机实验教学中心制

2020年10月

【实验环境】

Vivado

FPGA板子

【实验过程】

首先设计游戏的背景

**module** cel\_color**(**

**input** VGA\_clk**,**

**input** displayArea**,**

**input** **[**10**:**0**]** xCount**,** yCount**,**

**output** **reg** cel**,** in\_space**,** cel\_border**,** out\_space

**);**

**parameter** CEL\_SIZE **=** 32**;**

**parameter** OUT\_SPACE **=** 1**;**

**parameter** IN\_SPACE **=** 2**;**

**parameter** BORDER **=** 2**;**

**wire** **[**10**:**0**]** q\_x **=** xCount**/(**CEL\_SIZE**);**

**wire** **[**10**:**0**]** q\_y **=** yCount**/(**CEL\_SIZE**);**

**wire** **[**7**:**0**]** cel\_x **=** xCount **-** q\_x**\***CEL\_SIZE**;**

**wire** **[**7**:**0**]** cel\_y **=** yCount **-** q\_y**\***CEL\_SIZE**;**

**integer** i **=** IN\_SPACE **+** BORDER **+** OUT\_SPACE**;**

**integer** j **=** BORDER **+** OUT\_SPACE**;**

**integer** k **=** OUT\_SPACE**;**

**always@(posedge** VGA\_clk**)**

**begin**

**if(**displayArea**)**

**begin**

cel **<=** **(**cel\_x **>=** i**)** **&&** **(**cel\_y **>=** i**)** **&&** **(**cel\_x **<** CEL\_SIZE **-** i**)** **&&** **(**cel\_y **<** CEL\_SIZE **-** i**)** **?** 1 **:** 0**;**

**if(**cel**)**

**begin**

in\_space **<=** 0**;**

cel\_border **<=** 0**;**

out\_space **<=** 0**;**

**end**

**else**

**begin**

in\_space **=** **(**cel\_x **>=** j**)** **&&** **(**cel\_y **>=** j**)** **&&** **(**cel\_x **<** CEL\_SIZE **-** j**)** **&&** **(**cel\_y **<** CEL\_SIZE **-** j**)** **?** 1 **:** 0**;**

**if(**in\_space**)**

**begin**

cel\_border **<=** 0**;**

out\_space **<=** 0**;**

**end**

**else**

**begin**

cel\_border **=** **(**cel\_x **>=** k**)** **&&** **(**cel\_y **>=** k**)** **&&** **(**cel\_x **<** CEL\_SIZE **-** k**)** **&&** **(**cel\_y **<** CEL\_SIZE **-** k**)** **?** 1 **:** 0**;**

**if(**cel\_border**)**

out\_space **<=** 0**;**

**else**

out\_space **=** 1**;**

**end**

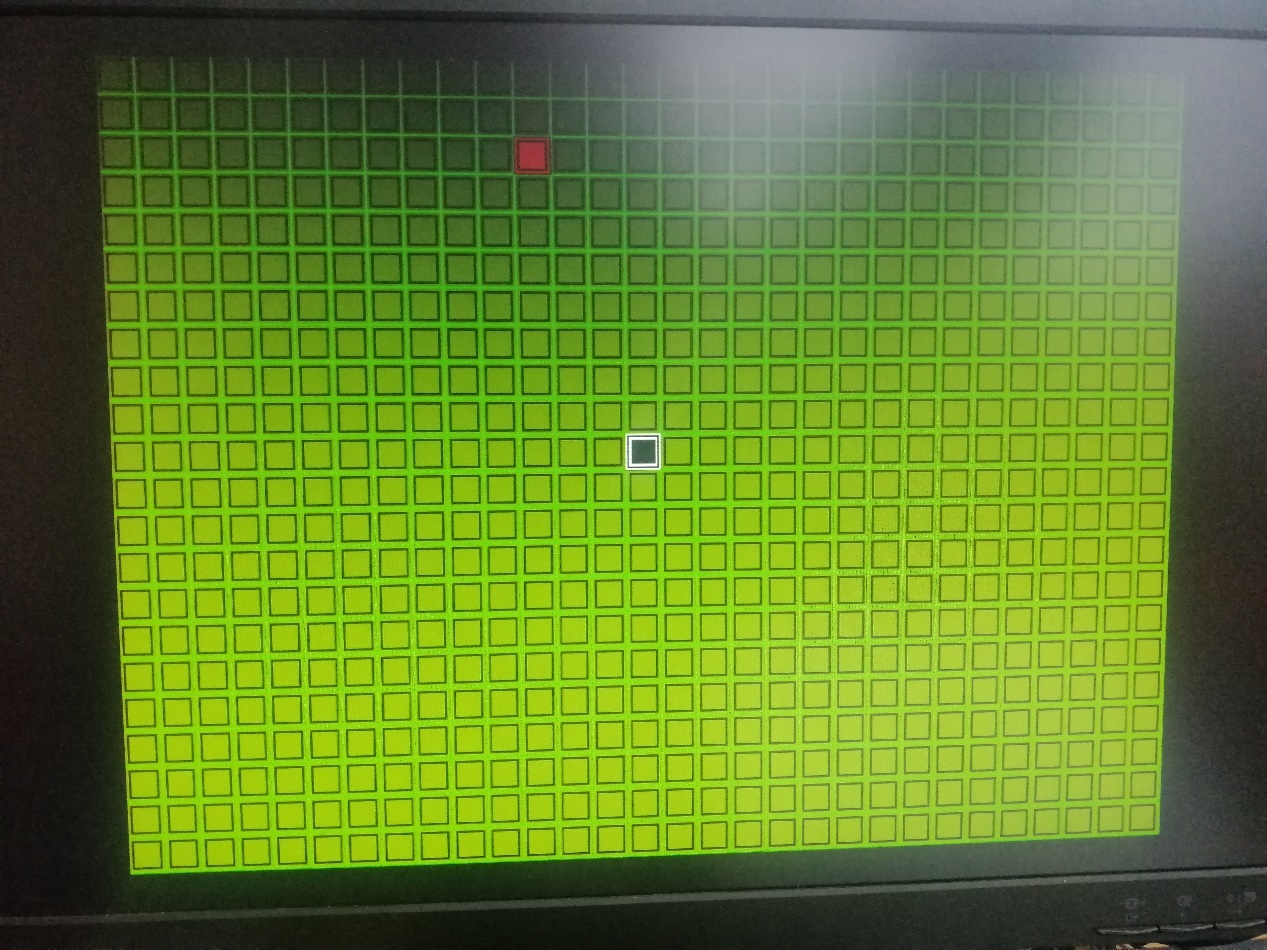
**end**

**end**

**end**

**endmodule**

结果



分析：输入信号为VGA\_clk，displayArea，xCount，yCount。其中displayArea = 1时，表示在显示区域。 xCount和yCount表示位置。

看结果图可以发现背景时很多小正方形组成的。每个正方形含四部分，如下图:

cel

in\_space

out\_space

cel\_border

到每个变量指的部分，其输出变量为1否则为0。四部分都可以设置自己的颜色。所以用模块就可以显示背景，蛇，目标和墙， 只需要到每个背景，蛇，目标或墙时候换颜色。

其次给目标设计一个输出随机位置的模块

**module** randomGrid**(**

**input** VGA\_clk**,**

**input** **[**10**:**0**]** max\_X**,** min\_X**,**

**input** **[**10**:**0**]** max\_Y**,** min\_Y**,**

**output** **reg** **[**10**:**0**]** rand\_X**,**

**output** **reg** **[**10**:**0**]** rand\_Y

**);**

**always** **@(posedge** VGA\_clk**)**

**begin**

rand\_X **<=** **((**rand\_X **+** 3**)** **%** **(**max\_X**-**min\_X**))** **+** min\_X**;**

rand\_Y **<=** **((**rand\_Y **+** 5**)** **%** max\_Y**-**min\_Y**)** **+** min\_Y**;**

**end**

**endmodule**

在设计一个VGA控制模块

**module** VGA\_gen**(**

**input** CLK**,** RESET**,** //clk=65MHzz

**output** displayArea**,**

**output** **reg** **[**10**:**0**]** xCount**,**yCount**,** //x,y pixel

**output** **reg** VGA\_HS**,**VGA\_VS

**);**

**parameter** H\_CNT **=** 11'd1343**;** //136+160+1024+24=1344-1

**parameter** V\_CNT **=** 11'd805**;** //6+29+768+3=806-1

**reg** h\_de**,** v\_de**;** //data enable

**reg** **[**10**:**0**]** h\_cnt**,**v\_cnt**;**

**always@(posedge** CLK**)**

**begin**

**if(**RESET**)**

h\_cnt **<=** 11'd0**;**

**else** **if(**h\_cnt**>=**H\_CNT**)**

h\_cnt **<=** 11'd0**;**

**else**

h\_cnt **<=** h\_cnt **+** 11'd1**;**

**end**

**always@(posedge** CLK**)**

**begin**

**if(**RESET**)**

v\_cnt **<=** 11'd0**;**

**else** **if(**h\_cnt**==**H\_CNT**)**

**begin**

**if(**v\_cnt**>=**V\_CNT**)**

v\_cnt **<=** 11'd0**;**

**else**

v\_cnt **<=** v\_cnt **+** 11'd1**;**

**end**

**end**

**always@(posedge** CLK**)**

**begin**

**if(**RESET**)**

h\_de **<=** 1'b0**;**

**else** **if((**h\_cnt**>=**296**)&&(**h\_cnt**<=**1319**))**

h\_de **<=** 1'b1**;**

**else**

h\_de **<=** 1'b0**;**

**end**

**always@(posedge** CLK**)**

**begin**

**if(**RESET**)**

v\_de **<=** 1'b0**;**

**else** **if((**v\_cnt**>=**35**)&&(**v\_cnt**<=**802**))**

v\_de **<=** 1'b1**;**

**else**

v\_de **<=** 1'b0**;**

**end**

**always@(posedge** CLK**)**

**begin**

**if(**RESET**)**

VGA\_HS **<=** 1'b1**;**

**else** **if(**h\_cnt**<=**11'd135**)**

VGA\_HS **<=** 1'b0**;**

**else**

VGA\_HS **<=** 1'b1**;**

**end**

**always@(posedge** CLK**)**

**begin**

**if(**RESET**)**

VGA\_VS **<=** 1'b1**;**

**else** **if(**v\_cnt**<=**11'd5**)**

VGA\_VS **<=** 1'b0**;**

**else**

VGA\_VS **<=** 1'b1**;**

**end**

**always@(posedge** CLK**)**

**begin**

**if(**h\_de **==** 1'h0**)**

xCount **<=** 11'h0**;**

**else**

xCount **<=** xCount **+** 11'h1**;**

**end**

**always@(negedge** h\_de**)**

**begin**

**if(**v\_de **==** 1'h0**)**

yCount **<=** 11'h0**;**

**else**

yCount **<=** yCount **+** 11'h1**;**

**end**

**assign** displayArea **=** **(**v\_de**==**1 **&&** h\_de**==**1**);**

**endmodule**

最后设计游戏的基本功能

**module** Play**(**

**input** CLK**,** RESET**,**

**input** UP**,** LEFT**,** DOWN**,** RIGHT**,**

**output** **reg** **[**3**:**0**]** VGA\_R**,** VGA\_G**,** VGA\_B**,**

**output** VGA\_HS**,** VGA\_VS

**);**

**parameter** SCREEN\_H **=** 1024**;**

**parameter** SCREEN\_V **=** 768**;**

**parameter** UNIT **=** 32**;**

**parameter** SNAKE\_MAX\_SIZE **=** 20**;**

**wire** VGA\_clk**,** locked**;** //65 MHz

**wire** displayArea**;**

**wire** **[**10**:**0**]** xCount**,** yCount**;** //x ,y pixel

**reg** **[**3**:**0**]** direction**;**

**reg** **[**10**:**0**]** snakeX**[**0**:**SNAKE\_MAX\_SIZE**];**

**reg** **[**10**:**0**]** snakeY**[**0**:**SNAKE\_MAX\_SIZE**];**

**reg** **[**SNAKE\_MAX\_SIZE**:**0**]** snakeBody**;**

**reg** **[**7**:**0**]** snake\_size**;**

**reg** game\_over**;**

**reg** apple**,** border**;**

**reg** **[**10**:**0**]** appleX **=** 64**;**

**reg** **[**10**:**0**]** appleY **=** 64**;**

**wire** **[**10**:**0**]**rand\_X**;**

**wire** **[**10**:**0**]**rand\_Y**;**

**reg** update**;**

**wire** cel**;**

**wire** in\_space**;**

**wire** cel\_border**;**

**wire** out\_space**;**

**reg** **[**27**:**0**]** count**,** max\_count**;**

// generate VGA clock (65 MHz) from input clock (100 MHz)

clk\_wiz\_0 clk\_wiz\_0**(**

**.**clk\_in1 **(**CLK**),**

**.**reset **(**RESET**),**

**.**clk\_out1 **(**VGA\_clk**),**

**.**locked **(**locked**)**

**);**

//VGA controller

VGA\_gen VGA\_gen**(**

**.**CLK**(**VGA\_clk**),** **.**RESET**(~**locked**),**

**.**displayArea**(**displayArea**),**

**.**xCount**(**xCount**),.**yCount**(**yCount**),**

**.**VGA\_HS**(**VGA\_HS**),.**VGA\_VS**(**VGA\_VS**)**

**);**

//get random number to place the apple

randomGrid randomGrid**(**

**.**VGA\_clk**(**VGA\_clk**),**

**.**max\_X**(**SCREEN\_H**-**UNIT**),** **.**min\_X**(**UNIT**),**

**.**max\_Y**(**SCREEN\_V**-**UNIT**),** **.**min\_Y**(**UNIT**),**

**.**rand\_X**(**rand\_X**),**

**.**rand\_Y**(**rand\_Y**)**

**);**

//color cels

cel\_color cel\_color**(**

**.**VGA\_clk**(**VGA\_clk**),**

**.**displayArea**(**displayArea**),**

**.**xCount**(**xCount**),** **.**yCount**(**yCount**),**

**.**cel**(**cel**),** **.**in\_space**(**in\_space**),** **.**cel\_border**(**cel\_border**),** **.**out\_space**(**out\_space**)**

**);**

**integer** i**,**j**;**

**always@(posedge** VGA\_clk **or** **posedge** RESET**)**

**begin**

**if(**RESET**)**

**begin**

// place the snake head at display center

i **=** SCREEN\_H**/**2 **-** UNIT**/**2**;**

j **=** SCREEN\_V**/**2 **-** UNIT**/**2**;**

snakeX**[**0**]** **<=** **(**i **-** i**%**UNIT**);**

snakeY**[**0**]** **<=** **(**j **-** j**%**UNIT**);**

// place apple

i **=** rand\_X**;**

j **=** rand\_Y**;**

appleX **<=** **(**i **-** i**%**UNIT**);**

appleY **<=** **(**j **-** j**%**UNIT**);**

**for(**i **=** 1**;** i **<** SNAKE\_MAX\_SIZE**;** i **=** i **+** 1**)**

**begin**

// place the invisible snake parts outside the scanning area

snakeX**[**i**]** **<=** SCREEN\_H**;**

snakeY**[**i**]** **<=** SCREEN\_V**;**

**end**

max\_count **<=** 28'd65\_000\_000**;**

snake\_size **<=** 1**;**

game\_over **<=** 0**;**

**end**

**else** **if(~**game\_over**)**

**begin**

**if(**update**)**

**begin**

**for(**i **=** 1**;** i **<** snake\_size**;** i **=** i **+** 1**)**

**begin**

snakeX**[**i**]** **<=** snakeX**[**i **-** 1**];**

snakeY**[**i**]** **<=** snakeY**[**i **-** 1**];**

**end**

**case(**direction**)**

4'b1**:** snakeY**[**0**]** **<=** **(**snakeY**[**0**]** **-** UNIT**);**

4'b10**:** snakeX**[**0**]** **<=** **(**snakeX**[**0**]** **-** UNIT**);**

4'b100**:** snakeY**[**0**]** **<=** **(**snakeY**[**0**]** **+** UNIT**);**

4'b1000**:** snakeX**[**0**]** **<=** **(**snakeX**[**0**]** **+** UNIT**);**

**endcase**

**end**

**else**

**begin**

// Detect if snake head hit the apple

**if** **(**apple **&&** snakeBody**[**0**])**

**begin**

i **=** rand\_X**;**

j **=** rand\_Y**;**

appleX **<=** **(**i **-** i**%**UNIT**);**

appleY **<=** **(**j **-** j**%**UNIT**);**

**if(**snake\_size **<** SNAKE\_MAX\_SIZE **)**

snake\_size **<=** snake\_size **+** 1**;**

**end**

// Detect if snake head hit border

**else** **if** **(**border **&&** snakeBody**[**0**])**

game\_over **<=** 1'b1**;**

// Detect if snake head hit the snake body

**else** **if** **(|**snakeBody**[**SNAKE\_MAX\_SIZE **-** 1 **:**1**]** **&&** snakeBody**[**0**])**

game\_over **<=** 1'b1**;**

**end**

**end**

**end**

//check update

**always@(posedge** VGA\_clk **or** **posedge** RESET**)**

**begin**

**if(**RESET**)**

**begin**

count **<=** 28'd0**;**

update **<=** 0**;**

**end**

**else** **if(**count **>=** max\_count**)**

**begin**

count **<=** 28'd0**;**

update **<=** 1**;**

**end**

**else**

**begin**

count **<=** count **+** 28'd1**;**

update **<=** 0**;**

**end**

**end**

//color cel

**always@(posedge** VGA\_clk**)**

**begin**

**if(**displayArea**)**

**begin**

**if(**border**)**

**begin**

**if(**cel**)** **{**VGA\_R**,** VGA\_G**,** VGA\_B**}** **<=** 12'h3**;**

**else** **if(**in\_space**)** **{**VGA\_R**,** VGA\_G**,** VGA\_B**}** **<=** 12'h0**;**

**else** **if(**cel\_border**)** **{**VGA\_R**,** VGA\_G**,** VGA\_B**}** **<=** 12'h3**;**

**else** **if(**out\_space**)** **{**VGA\_R**,** VGA\_G**,** VGA\_B**}** **<=** 12'h0**;**

**else** **{**VGA\_R**,** VGA\_G**,** VGA\_B**}** **<=** 12'h0**;**

**end**

**else** **if(|**snakeBody**)**

**begin**

**if(**cel**)** **{**VGA\_R**,** VGA\_G**,** VGA\_B**}** **<=** 12'h0**;**

**else** **if(**in\_space**)** **{**VGA\_R**,** VGA\_G**,** VGA\_B**}** **<=** 12'hfff**;**

**else** **if(**cel\_border**)** **{**VGA\_R**,** VGA\_G**,** VGA\_B**}** **<=** 12'h0**;**

**else** **if(**out\_space**)** **{**VGA\_R**,** VGA\_G**,** VGA\_B**}** **<=** 12'hfff**;**

**else** **{**VGA\_R**,** VGA\_G**,** VGA\_B**}** **<=** 12'h0**;**

**end**

**else** **if(**apple**)**

**begin**

**if(**cel**)** **{**VGA\_R**,** VGA\_G**,** VGA\_B**}** **<=** 12'h700**;**

**else** **if(**in\_space**)** **{**VGA\_R**,** VGA\_G**,** VGA\_B**}** **<=** 12'h0**;**

**else** **if(**cel\_border**)** **{**VGA\_R**,** VGA\_G**,** VGA\_B**}** **<=** 12'h700**;**

**else** **if(**out\_space**)** **{**VGA\_R**,** VGA\_G**,** VGA\_B**}** **<=** 12'h0**;**

**else** **{**VGA\_R**,** VGA\_G**,** VGA\_B**}** **<=** 12'h0**;**

**end**

**else**

**begin**

**if(**cel**)** **{**VGA\_R**,** VGA\_G**,** VGA\_B**}** **<=** 12'h340**;**

**else** **if(**in\_space**)** **{**VGA\_R**,** VGA\_G**,** VGA\_B**}** **<=** 12'h3**;**

**else** **if(**cel\_border**)** **{**VGA\_R**,** VGA\_G**,** VGA\_B**}** **<=** 12'h340**;**

**else** **if(**out\_space**)** **{**VGA\_R**,** VGA\_G**,** VGA\_B**}** **<=** 12'h70**;**

**else** **{**VGA\_R**,** VGA\_G**,** VGA\_B**}** **<=** 12'h0**;**

**end**

**end**

**end**

//init direction

**always@(posedge** CLK**)**

**begin**

**if(**RESET**)**

direction **<=** 4'b0**;**

**else** **if(**UP**)**

direction **<=** 4'b1**;**

**else** **if(**LEFT**)**

direction **<=** 4'b10**;**

**else** **if(**DOWN**)**

direction **<=** 4'b100**;**

**else** **if(**RIGHT**)**

direction **<=** 4'b1000**;**

**else**

direction **<=** direction**;**

**end**

// Detect if the VGA scanning is hitting the border

**always** **@(posedge** VGA\_clk**)**

**begin**

border **<=** **((**xCount **<=** UNIT**)** **||** **(**xCount **>** SCREEN\_H **-** UNIT**)** **||** **(**yCount **<=** UNIT**)** **||** **(**yCount **>** **(**SCREEN\_V **-** UNIT**)));**

**end**

// Detect if the VGA scanning is hitting the apple

**always** **@(posedge** VGA\_clk**)**

**begin**

apple **<=** **((**xCount **>=** appleX**)** **&** **(**yCount **>=** appleY**)** **&** **(**xCount **<** appleX **+** UNIT**)** **&** **(**yCount **<** appleY **+** UNIT**));**

**end**

// Detect if the VGA scanning is hitting the snake head or snake body

**always@(posedge** VGA\_clk**)**

**begin**

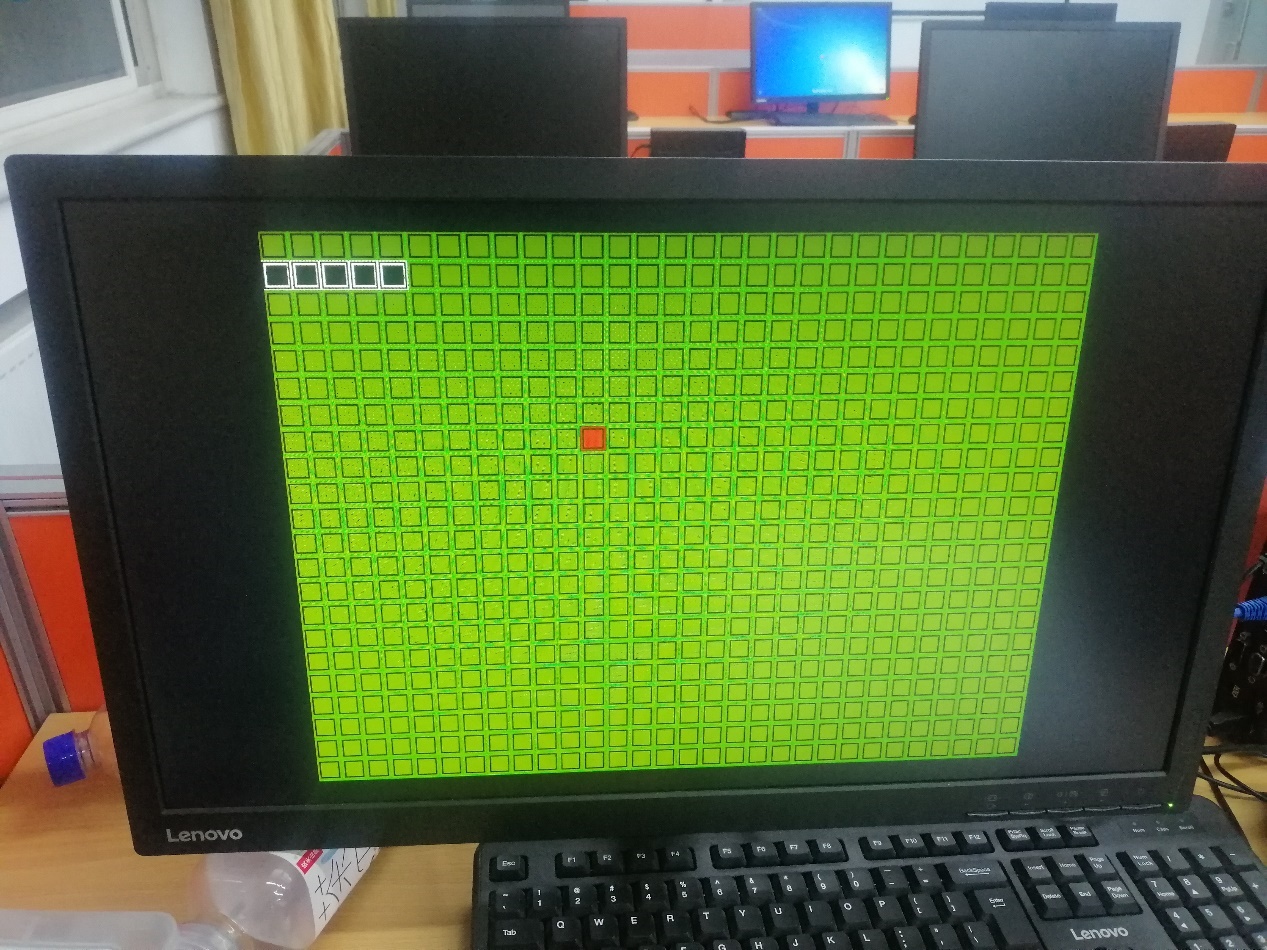
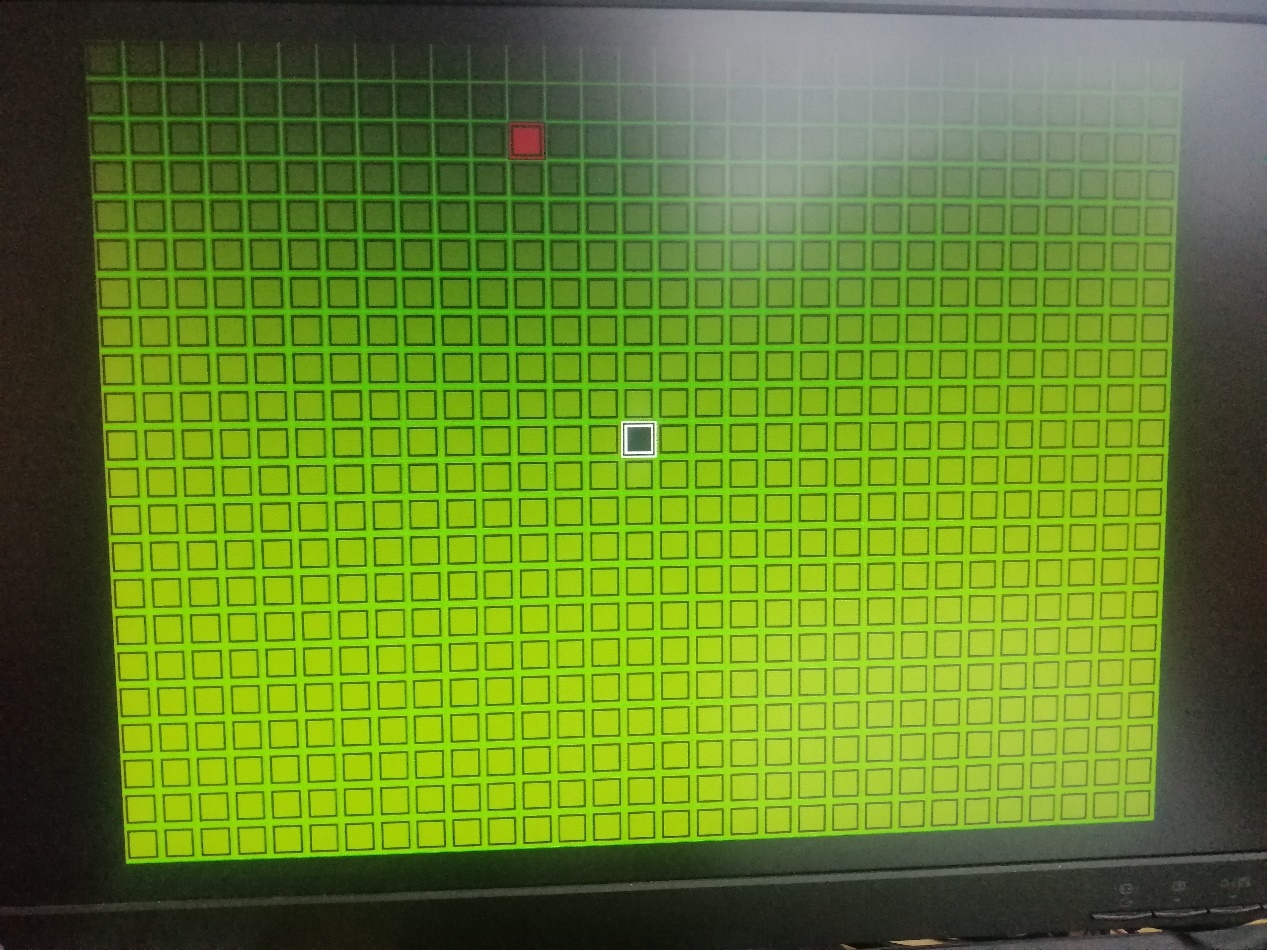
**for(**i **=** 0**;** i **<** SNAKE\_MAX\_SIZE**;** i **=** i **+** 1**)**

snakeBody**[**i**]** **<=** **((**xCount **>=** snakeX**[**i**])** **&** **(**yCount **>=** snakeY**[**i**])** **&** **(**xCount **<** snakeX**[**i**]** **+** UNIT**)** **&** **(**yCount **<** snakeY**[**i**]** **+** UNIT**));**

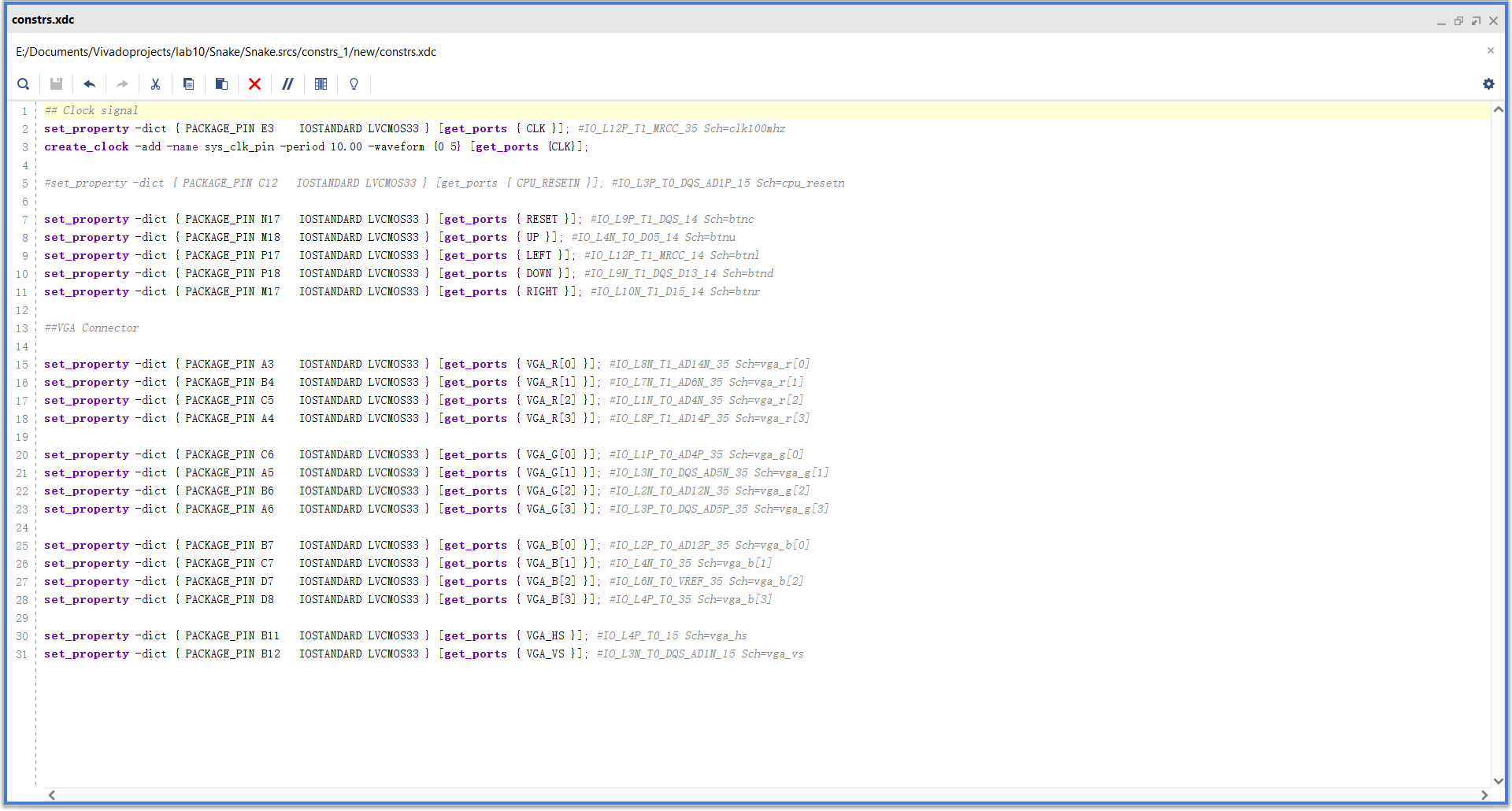
**end**

**endmodule**

结果



XDC文件



FPGA的上下左右按键用来控制蛇的行动。

【总结与思考】

该实验花的大部分时间在了设计完美的图像，所以有些的功能只是做出了最基础的比如蛇吃一个目标后增长，撞到墙游戏暂停，等。