Boston University

College of Engineering

Department of Electrical and Computer Engineering

Mario Plus

EC 551 Final Project Report

Team Name:

Five Heroes on the Lang Ya Mountain

Team Members:

Tao Liu, Jiaze Li, Yong Jia

Dongyang Zhang, Yizhou Zhong

Abstract

This project is an arcade game which is compiled by Verilog on FPGA board. It combines three parts which are joy stick, FPGA and VGA. The program involves three different states, including character selection, Mario world and star war. What's more, four heroes can be picked up in character selection. In Mario world, players first select the character, and then have the ability to control the hero to fight with monster by moving, jumping and attacking or go to the space by taking mushroom. In star war, players can handle the spacecraft to shoot down enemy plan and death star. No matter whether win or lost, the players must go back to Mario world. If they dead in Mario world, it would turn to the beginning, character selection.

Short Description

TEST.v: This is the top module that connects each module together to produce a complete game. I chose this name early at the beginning of the project and it is too late to change it when the game grows larger and larger…

graph.v: This module takes the input signals from the buttons and clock then outputs the information of the RGB value for display. This module contains the whole game logic and all the display of the game, so it is extremely long that it has over 1000 lines of codes. The first half of the code defines the game logic and the other half defines the display.

xxx.xco: There are multiple XCO files inside the graph module. Each file represents a certain sprite used for the texture inside the game. They are generated from the Xilinx core generator as Block ROM.

vga_sync.v: This module is provided by the professor. It takes in clock signal and gives out horizontal and vertical sync signal for the VGA display and a pair of coordinate on the screen as well. The coordinate is used in graph module.

clk_pixel: This is a clock divider that produces a clock signal approximate to 25GHz for the VGA module according to the industry standard.

debounce.v: This module takes two single bit inputs, clk and button. The output of this module is click. It is used to provide smoother controlling for players.

timer_1ms.v: This module take a single bit input clk. The output of this module is clk_1ms which is computed by dividing the inputs.

User Guide

As the program is finished with Xilinx ISE 14.7, some problems may occur with lower version. For example, in ISE 14.2, the .xco file used will come into error of re-declaration of module during synthesizing. You can either comment the .v file of it or delete the .v file in the project folder.

The version you have is the final version with all the functions of the game. However, it is a “Button Version” that can only be played with the 5 buttons on FPGA. If you want to go further to interface it with some controlling device such as the joystick, please feel free to modify the UCF file. Don’t forget to modify the signal used inside the Verilog program considering whether the control signal is positive or negative. You can easily find out how the buttons control the game inside graph.v and make your own.

Instruction for Modification

If you are ambitions and want to improve the game, it is not easy to go over the whole code to understand the logic. I can give you some advices.

1. **Replacing Sprites**

You can replace the sprite used in the original game with your own .xco files. Be sure to change the codes for displaying in the latter part of the program. You can find them easily if you catch the key word “rgb_now”. Alto notice that sprites have their own sizes. Make sure the address you give the .xco module is correct.

2. **Adding Scenes**

As you see, there are 3 scenes in the game so if you want to add more, you can just define more if phases and make transport inside different scenes.

3. **Developing AI**

Actually, I make comments of each part of the game so you can easily figure what a certain part of code is for. However, there are some “tricky codes” that may not be easy to understand. Please feel free to discuss with me via funnytao@bu.edu. Thank you.