“Nexys Says”

A simple game based off Simon Says

List of Authors (Adam Jesse, Abdul Wasay, Anthony Williamson, Graeden Ball)
Electrical and Computer Engineering Department
School of Engineering and Computer Science
Oakland University, Rochester, MI
e-mails: ajesse@oakland.edu, wasay@oakland.edu, awilliamson@oakland.edu, gball@oakland.edu

I. Introduction

Our project idea is to create a simple game using the RGB light and buttons on the FPGA board. The premise of the game is to be shown a series of colors and then input that color value using the buttons. For example, the light might flash Red, Green, and Blue. After a short time, the light will turn off and the user will have to press the button that corresponds to that color. We will be combing many different topics covered in class and in lab. In our design we will use registers that we use for many of our laboratory assignments and homework We will be implementing a finite state machine (FSM) to move the game along and another FSM to check the input to make sure that it is correct. We will also use counters for points and generating a color combination. Many of the components taught in class will be implemented in this design.

II. Methodology

The classic game we attempted to emulate on our board is commonly known as “Simon Says.”

A. Overview

The game “Nexys says”, is a simple sequence memory game that is implemented using digital logic design and the Nexys A7 100T FPGA board. The goal of this game is to commit to memory the order of a sequence of colors, and then successfully reproduce the sequence to move on to the next sequence. The length of each sequence is increased after each successful sequence entry, allowing the user to face a more complex sequence as they advance in the game. To begin the game, the user must push the middle button of the 5-button setup installed on the board. Following this, the red-green-blue (RGB) led will flash a sequence of colors that the user will have one chance to memorize. After the user has committed the sequence to memory, they will then have to repeat the sequence correctly by selecting which register they want to write to via ‘address’ switches on the board and entering corresponding colors using the remaining four buttons which represent a color. If the input string is correct, a separate 7-segment display will show a “P”, allowing the user to pass to the next level, or an “F” will be displayed, which represents failing and the game will reset if the input string is incorrect.
Additionally,

B. Structure

The structure of the circuit is as follows. The main FSM has input variables: start, RO, zC, and zP.Start will be input from the user to start the game, essentially activating the FSM. RO is the register output, zC is a value that stands for the completion of the color counter cycle, and zP is the results of the pattern finder. The output of the FSM is the variables: ER, EC, EP, PC, and DC. ER will enable the register, EC enables the color display, EP enables the pattern finder FSM, PC is the enable for the point counter, and DC is the input of what color is to be displayed by the LED. The main FSM controls the game by enabling and disabling the other components that each have separate tasks.

The “Pattern Finder” FSM is responsible for detecting the correct color pattern and then reporting whether the input is correct or not. This FSM takes input from the decoder, enable EP, and input from the register. Once the input is tested, the FSM outputs enable EIC for the color display, input color IC, and the results of the pattern finding operation.

The rest of the circuit is comprised of the register, two counters, a seven-segment display controller, an input decoder, and a RGB LED controller. The register is used to store the “random” color output from the eight-bit counter. The register is used to enable or disable the color input from that counter. The other counter is used to count the score of the user. The input decoder is used to convert the input into a signal that is easy for the patter FSM to interpret. Finally, there is a controller for the seven-segment display and a controller for the RGB LED.

The controllers will take input and convert the signal into a color or a number.

III. Experimental Setup

To verify that our game is functioning correctly, our group will use two main methods to test and debug our simple game. Timing Diagrams and physically testing the game to insure it works correctly. We believe that through the use of these two methods, our group will be able to efficiently debug our game and ensure that it works as it is intended in all situations.

The use of timing diagrams will allow us to see if our VHDL program is working as intended. We plan to do this by laying out all the internal and external signals onto a timing diagram and inputting various values into the program via the Testbench to ensure that the program is functioning as intended and is outputting the correct value in different situations. If an error is found, we will trace the signal with the error back through the various components of our game until the point where the signal is initially wrong is found, then that component will be rechecked for errors and fixed.

Once it is insured that the program is working correctly, then we will move onto physically testing the game. This process is mainly to test the user interface and to ensure that the game runs smoothly. We will test this by playing the game many times to see if it is consistently providing the proper outputs as shown in the timing diagram. In addition, we will test the game to find unnecessary delays and bugs that would cause a bad user experience. Once these are identified, they will be traced back to their start via the timing diagram and VHDL code and changes will be made.

With the help of these testing processes, our group is confident that we will be able to efficiently solve any problems in our simple game and ensure that it works efficiently and well for the users to enjoy.