Samuel Wynsma and Jarek Thomas

Final Project Design Document

CS 373-1

The goals and project purpose are close to the same as they were in the project proposal.

Our project will be a basic Simon says game with multiple squares that will flash up. The user then presses a switch or buttons which correspond to that particular square. The game will gradually get more complicated until the player messes up or the player goes a certain number of iterations without getting the right pattern. The squares will be varied different colors. If the user makes a mistake, the game will restart.

The purpose of the project is to create and understand how user input and clocks work with flashing squares.

Goals for the project, in order to intended completion/importance:

1. Get the finite state machine set up for Simon says
2. Get the colored squares up on the board.
3. Make sure the colored squares are all different colors.
4. Get the colored squares to flash.
5. Get the flashes to match a particular pattern or sequence.
6. Start by hard-coding a basic pattern
7. Match the buttons/switches on the board to particular squares.
8. Allow the program to check if the squares the user presses match the pattern.
9. Program a “fail” sequence and a “success” sequence when the user presses the wrong/right square.
10. Get the program to reset the pattern in the case of player failure.
11. Get the program to extend in length as we go (with a maximum length)
12. Program a win condition when the player completes up to the maximum length.
13. Turn the basic pattern into a random pattern. (stretch goal)

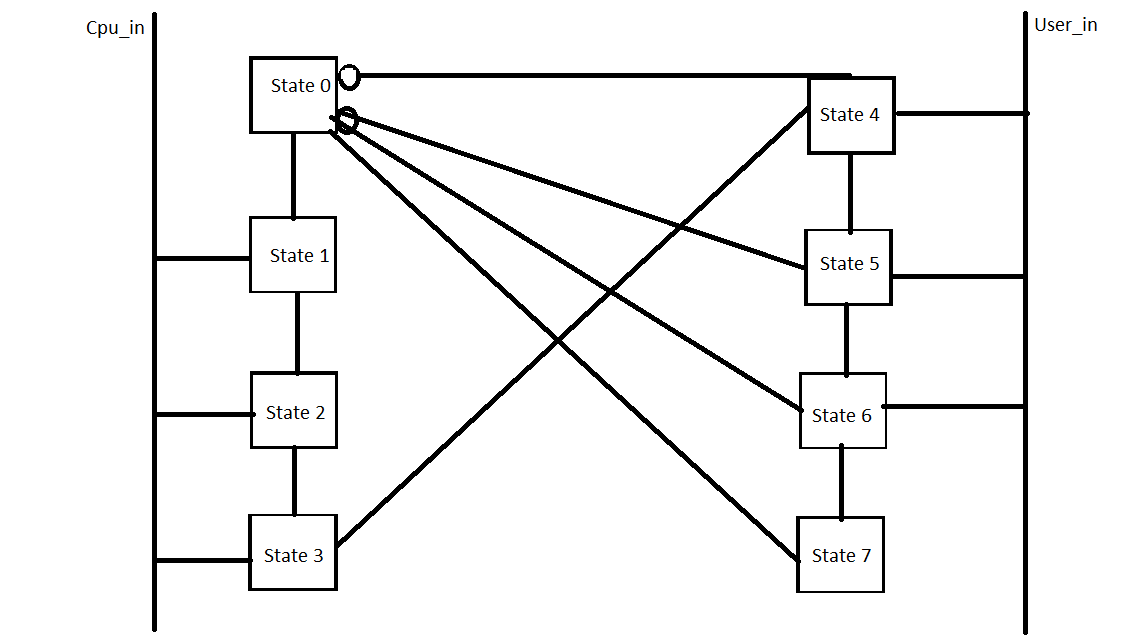
Our hope is to at least make it to Step 5 or step 6 on this list of steps. Each step is dependent on the last one, and so therefore, increasing in difficulty as we progress.

We will have two or three major vhdl files in this project, depending on how far we go.

The first vhdl file is the finite state machine for the Simon Says game. This file contains a 8 state finite state machine representing the game. The first state is the initial state, which is when the game is starting. In this state, no squares have flashed and no user inputs will be recognized. The second, third, and fourth states are where the computer flashes a set pattern of flashing squares. The fifth, sixth, and seventh states are the user input states. The use must match the square pressed by the cpu in the related state. This means that the user in the fifth state presses what was shown in the second state, and so forth. If the user messes up, the game will restart. If the user goes through all these three states successfully, then the program will go to the last state, which is the “Win” state. The cpu will then restart the program.

The second vhdl file will actually display these results to the screen. This file is a vga file designated to display elements on the screen. This file will cause the screen to display the squares that the cpu will flash and the player will replicate. The player will replicate the pattern using switches or buttons if we get that far.

output



This is the state diagram to represent the program, and below is the hardware block diagram.

