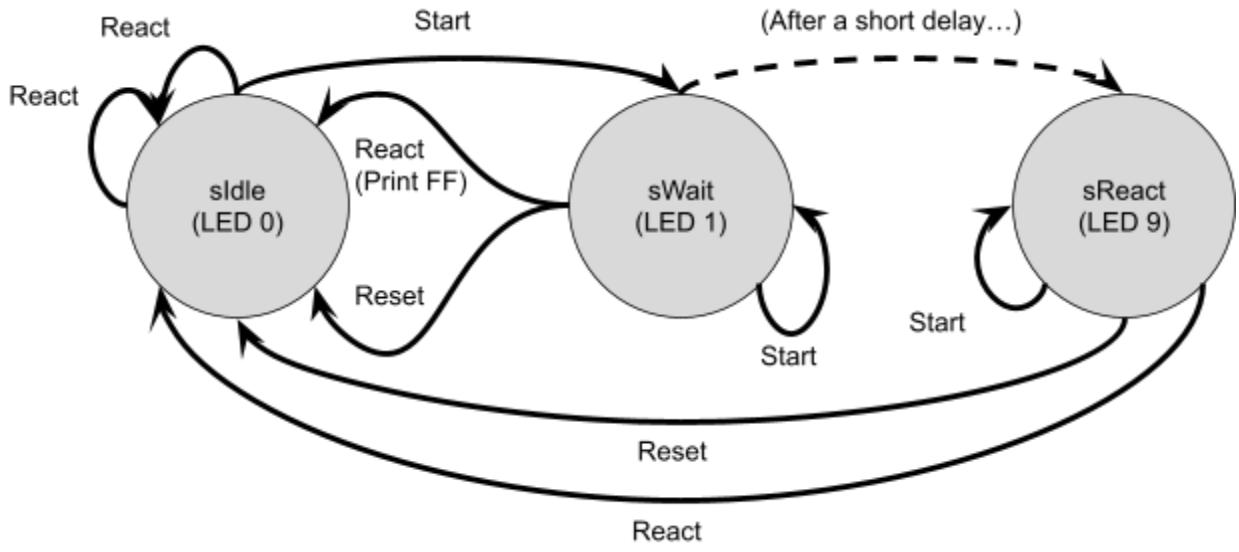
CS232 Project 3: Reaction Timer

September 30th, 2024

Task: Reaction Timer

This project involves designing a reaction time testing circuit using a state machine written in VHDL. The circuit will display the user's reaction time in milliseconds on a two-digit 7-segment display. The process begins when the user presses the reset button, which sets the system to an idle state with "00" displayed and the rightmost green LED (LED 0) lit. Upon pressing the start button, the system enters a waiting phase, lighting up Green LED 1. After a random delay, the rightmost LED (LED 9) will light up, signaling the user to press the reaction button as quickly as possible. The circuit then starts counting in milliseconds, and the user's reaction time is shown on the display.

Circuit:



Simulation:

https://drive.google.com/file/d/1jLTH2Qmb7pNWzyT1mq8v4D9Bb14y4yC3/view?usp=sharing

Explanation:

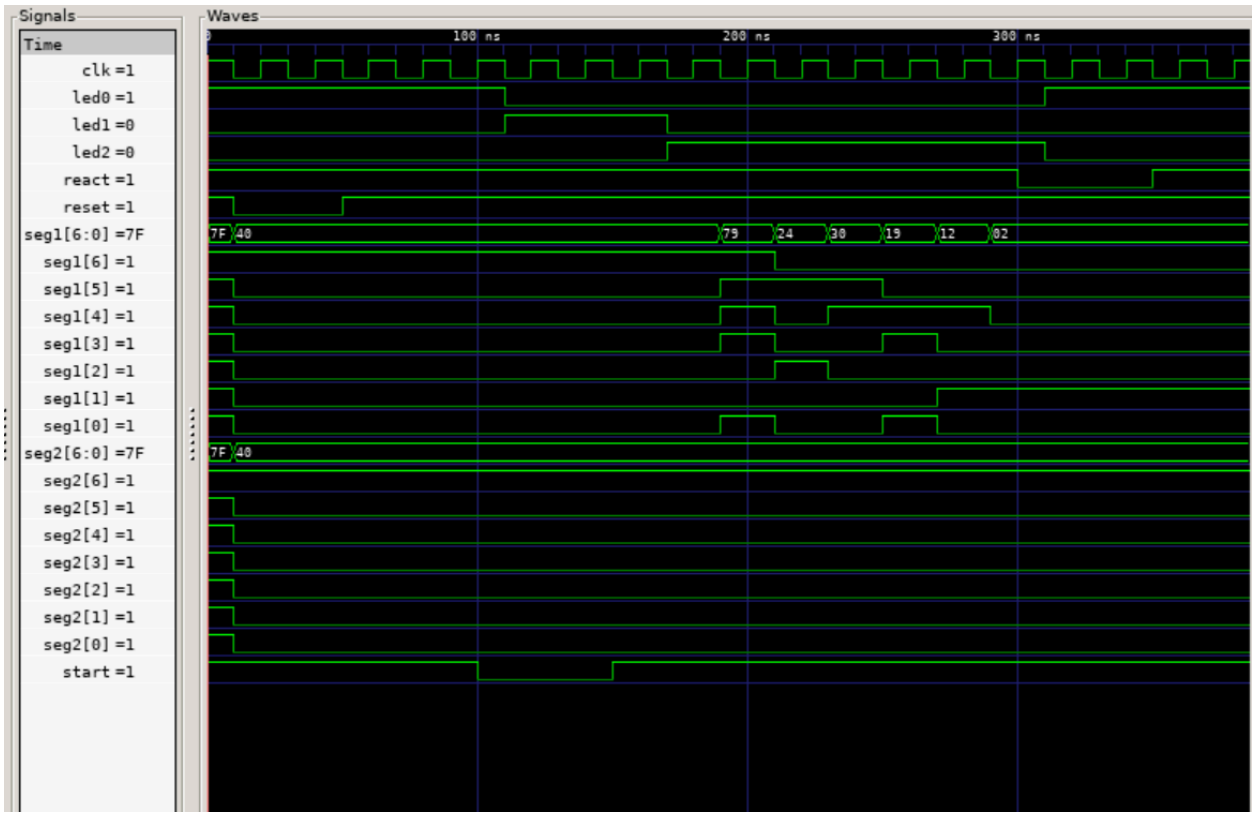
There was already a number printed on the screen, so I started by pressing the reset button to turn it back to 0. Once that was done, I could start the reaction test by hitting the middle button. I know it switched because LED 0 (Which represented the idle state) turned off and LED 1 (Which represented the wait state) turned on. After a short delay, the LED turned on and the timer started going off. I had to press the third button to stop it. Once that was done, I could then immediately start again by pressing the start button. Afterwards, I tested pressing the react button early, to get an FF display, which worked!

Extension 1: Making the top level file a VHDL file

https://drive.google.com/file/d/1-8SuTyEh-rr7_A6nkYVAjcO4wXLpP4Wj/view?usp=sharing

In this extension (project name: reactionvhdl), puts the base BDF file into a full VHDL file, and runs the same as before via the board.

Extension 2/3: Simulating a test bench for the said VHDL file and making an animation for it (2 in 1!)



To take the previous extension a step further (project name: reactiontest), I made a new file called reactiontest.vhdl to simulate the buttons being pressed in GTKwave, which can be seen above. The only difference is because the clocks on the board and GTKwave run differently, I set the mstime vector to take the last 8 digits of count, instead of the first 8. On the diagram, All the segments are on since it represents no segments being on. After start is pressed (by the dip because that's the signal buttons usually send when pressed) it switches to sWait (led1 is on instead of 0), then starts counting up, since seg1 starts shuffling through all the digits. When react's signal dips, it pauses, just as it would in the real project!

https://drive.google.com/file/d/1s8ho9fkpxr1DquUZP6U4IF3R_dHfwaW-/view?usp=sharing

Using this predetermined sequence, I also visualized it by mapping all of the signals, leds and buttons to a virtual python machine, which can be seen through the link above.

Extension 4: 2 Player Reaction Fight

https://drive.google.com/file/d/1cV8bZhvND4C-orn4znpBJISBe6CBCKKw/view?usp=sharing

This extension (project name: twoplayer) allows 2 people to play, although in the demonstration I'm just playing by myself because I don't have any friends right now. :(In any case, since there are only 3 buttons, to avoid overlap the reset button has been turned into a switch, which when flipped resets to the idle state as shown. I've also allocated another 7 digit display to show which player (Player 1 or Player 2, buttons 2 and 1 respectively) pressed first, and showing that reaction speed. It also shows FF and the opposite player's number in case they pressed too early. Generally, it works about the same.

Acknowledgements

TA Dean had to sit through everything since I kept messing up the base project, a lot of troubleshooting had to be done. I feel bad for him. Give him my condolences. Other TAs and Professor Li helped troubleshoot some extensions!