CS 152A

# Introductory Digital Design Lab

LAB #3

Floating Point Conversion

February 23, 2015

Grade:

-----------

Farhad Shahmohammadi

Name 1: Anthony Pimentel

SID 1: 004019166

Name2: Jason Tieu

SID2: 304047667

## Introduction and requirement

The goal of this lab was to design a stopwatch circuit that would count minutes and seconds and display the current time on the seven-segment display on the Nexys3 board. Both the minutes and the seconds can reach a maximum value of 59 before returning to 0. Left alone, the timer counts normally. However, there are 4 inputs to the timer: a button that **pauses** the timer, a button that **resets** the counter to 00:00, a switch that puts the timer into **adjust** mode (increases the value of the selected section of the timer twice per second), and a switch that **selects** which section of the timer to adjust (minutes or seconds).

We had to split a 100 MHz Master Clock into 4 clocks to use elsewhere in the timer: a 1 Hz clock that counts the seconds normally, a 2 Hz clock that increments either the minutes or the seconds during adjust mode, a faster clock (50 – 700 Hz) to show the correct value of the timer on the seven-segment display, and a clock to blink the display while in the adjust mode (> 1 Hz).

Lastly, since we were using buttons and switches on the FPGA board, we had to deal with noise in our input. For this reason, we had to implement a debouncer, which would clean up our input so that we wouldn't respond to all the glitches.

## Design Description

We designed the architecture of the Stopwatch using 5 main modules.

*ClockSplitter:*

This module takes the 100 MHz Master Clock as input and outputs the 4 other clocks we used (described above). To do this, we calculated the number of cycles of the

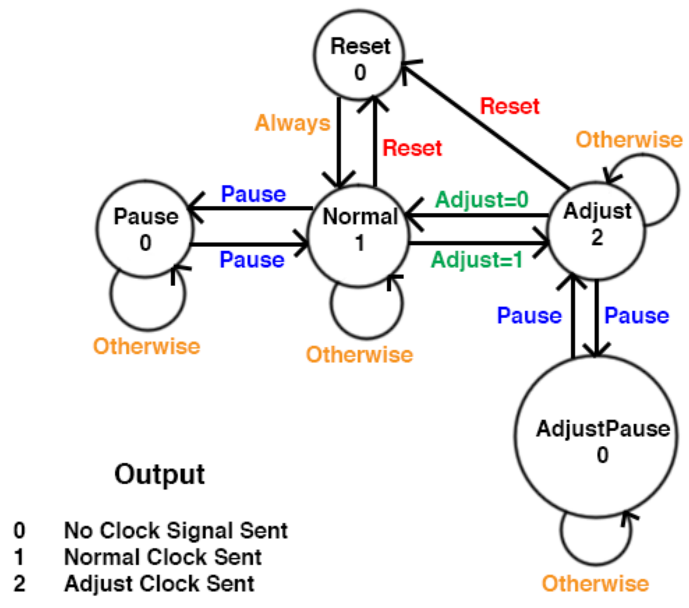
Master Clock that corresponds to a flip in each of these clocks and flipped them after that many cycles.

*Debouncer:*

This module simply samples the values of the input buttons/switches at a certain frequency so that we don't get noise when we are reading them and using their values. This avoids situations where we think we press buttons multiple times even though we only press it once.

*StateMachineModule:*

This module keeps track of the state of the stopwatch. Pause transitions occur after the release of the Pause button, while Adjust transitions occur as long as Adjust has the appropriate value, and Reset transitions occur as long as the Reset button is pressed. This module sends two signals: one when the counter should increment the time, and another when we are currently adjusting the time (used later in the Seven-Segment Display module. Below is a diagram of the state machine that governs this stopwatch.



*CounterModule:*

This module just keeps track of the time. Every time it receives a signal from the State Machine Module, it increments the current time (accounting for overflow). When the stopwatch is in adjust mode, it uses the value of the SEL switch to increment the proper digits of the clock.

*SevenSegmentDisplay:*

The last module displays the value of the stopwatch on the FPGA board. It does this by quickly cycling through the digits of the display and lighting up the appropriate segments to represent the number of the stopwatch. It also handles blinking the appropriate digits of the time when the stopwatch is adjusting either the seconds or the minutes.

## Simulation Documentation

For our test bench, we mainly focused on trying to make sure the each digit for the counter was being incremented correctly (the one's place went from 0-9 and ten's place went from 0-5 for seconds and minutes). We left the normal clock on to see each cycle was being handled correctly. Most of our testing was actually on the Nexys 3 board. We used the LED lights to monitor the activity for the pause, reset, adjust, and select signals. It was much easier to 'debug' using the Nexys 3 board since we were able to get immediate feedback from our test cases. Using the simulation would be much more difficult and tedious since we would have to decide when exactly to mimic a button press or flip a switch. Our test cases were mainly on-the-fly decisions to press a button such as pause or reset. This was the best way to test since we could not assume that the buttons and switches would be changed in any particular order. By randomly changing them, we could see if the counter behavior was accurate.

## Conclusion

The biggest difficulty we encountered was having the seven-segment display to display correctly. When we first tested using the Nexys 3 board, it was very difficult to debug the sporadic change in the seven-segment display. Even when asking for help, figuring out the issue was incredibly difficult. Anthony was able to fix our issues by rewriting our modules and altering some code to better fit our design. After that, everything was working smoothly and we easily fixed the rest of the minor bugs.

## Individual Contribution

During the lab sessions, Jason helped with the design of each module as Anthony was writing the code. For this lab report, Anthony took care of the first half and Jason finished the second half (50% and 50%). Since Anthony was able to take care of our biggest issues and bugs, he contributed much more in this lab. In terms of getting the project to complete, Anthony contributed 75% of the work and Jason contributed 25%.