Tyler Swenson

EE457

Lab 8 Report

4/5/2025

EE457 PWM

# Introduction

In this lab, we will be implementing a string of PWM pulse width modulation to be able to light up with various duty cycles to make it look like a red light is bouncing back and forth. The design will continue to bounce an led back and forth with three dim LEDs around each side to give more of an effect to the movement.

# Theory of Operation

## Requirements

1. The design will reset when the reset signal is asserted low.
   1. Resetting will set the right-most LED as the 100% duty cycle and restart.
2. The 100% duty cycle LED will oscillate back and forth through the 10 LEDs
3. On either side of the 100% LED, there will be a 60%, 20%, and 10% duty cycle LED in decreasing duty cycle from the center LED
4. SW(0) will change the speed.
   1. There will be a slow speed and a fast speed for the LED bouncing
5. All inputs and resets will be synchronized.

## Description of the Design

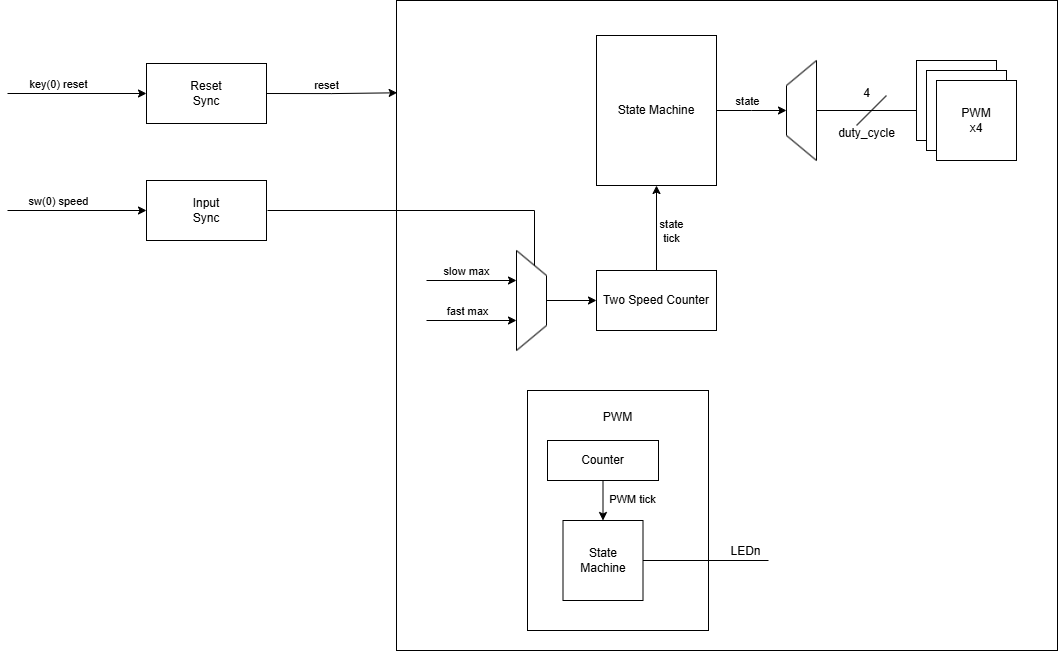


Figure 1: Design block diagram

In this design, only one PWM will be used for each duty cycle, and the output will be piped to the LED that is currently utilizing it. This will reduce the number of PWM signals I need to manage and allow reusability. The design will continue to oscillate the 100% LED back and forth with the 60%, 20%, and 10% surrounding it respectively. SW(0) will then be used to switch between two different speeds determined by the two-speed counter to use for the speed of the state machine. KEY(0) is then used for a reset, and when pressed, will set the 100% LED back to the right-most LED.

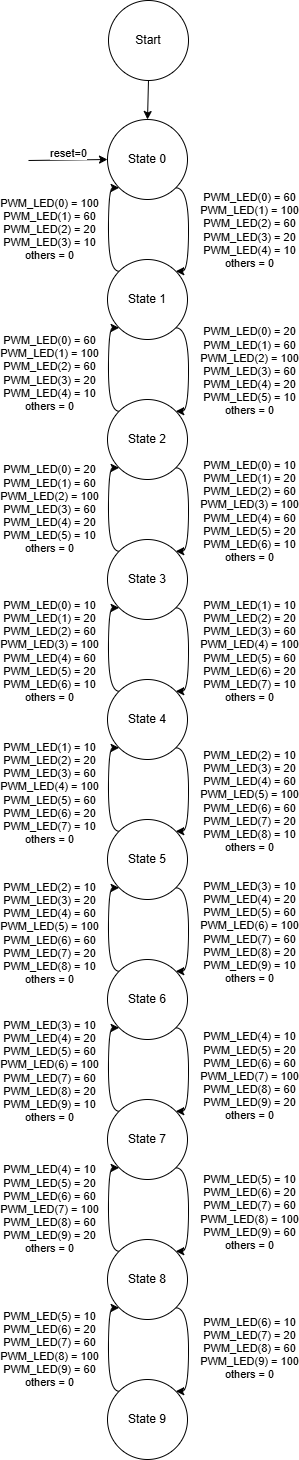


Figure 2: Design state machine

# Verification

## Test Plan

To test the design, a testbench will test the LED going back and forth to see the 100% bounce around with the 60%, 20%, and 10% LEDs surrounding it. To test this, the testbench will simply wait for the state machine to cycle through. The switch will then be used to change the speed of the oscillating LED, and again, the testbench will just wait for the state machine to cycle through. There will be a little bit of extra time to allow for the state machine to reach an LED that isn’t on the end when a change is made. Then finally, the reset will be tested to ensure the 100% goes to the right-most position when pressed.

A screen shot of a computer program

AI-generated content may be incorrect.

Figure 3: Code snippet for testbench

## Test Bench



Figure 4: PWM duty cycle changing and LED assignment

A black and green screen with numbers

AI-generated content may be incorrect.

Figure 5: Looping state machine and position used for 100% LED



Figure 6: Speed change on switch change

A screen shot of a number

AI-generated content may be incorrect.

Figure 7: Reset signal to check the state returns to 0

# Conclusion

In this lab, I learned how to design a fairly simple PWM state machine to bounce the LED back and forth. To challenge myself, I wanted to only use the number of PWMs as I had duty cycles, i.e. 4 for 100%, 60%, 20%, 10%. As recommended in the lab I used the for generate loop which allowed me to initialize how many I needed. This was a little more challenging, as I had to use one std\_logic\_vector to set the bits for each of the PWM that it would alter in the PWM blocks. I ran into an issue with understanding the MSB and LSB in the std\_logic\_vector that I provided to the for generate block. This caused the LEDs to be in reverse brightness on the initial run. After that, the design worked as expected.