California State University, Northridge
ECE 524L - Advanced FPGA Design
Fall 2022

Lab 7: Pulse Width Modulation

November 11th, 2022
Instructor: Saba Janamian

Introduction:

The purpose of this lab was to introduce the topic of pulse width modulation (PWM) and how to design modules that can implement this concept for an FPGA. We will also learn how to implement PWM with changing the intensity levels of RGB leds and implement rotations on a servo. This lab has important concepts like how we should setup our frequency for our system so that we can limit vibrations, led flickering and setting oscillations so that we can lower wasted power.

Procedure:

To begin the lab we first will implement a basic PWM controller. While this won't be the module we use for the entire lab for all four parts of the experiment, it is a way for us to first learn what a PWM is and how it works as well as how to design in Vivado. Figure 1 below is a basic PWM design that we create first just to get an understanding of it. It has a binary counter and a comparator. When the duty cycle, input and duty is set the comparator the comparator's output will be asserted. It is asserted when the counter's value is below a threshold value.

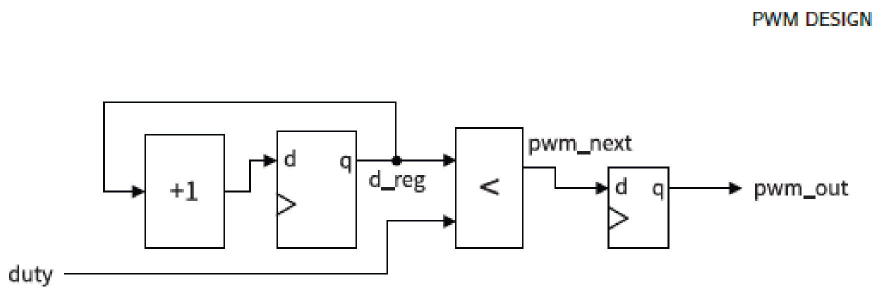

Fig 1. Basic PWM Design

Part 1:

In part 1 we will design a module that gradually dims the LEDs. It only needs two modules and two modules will be a recurring step for the other parts of the lab as well. Another recurring step in the other sections is the use of a PWM enhanced module. This module will first be created in this part and reused later on. The PWM enhanced is taking the foundation from the first PWM design and refining it to better suit the conditions that will be set for other parts of the lab. The Enhanced PWM is used to accommodate for 100% of the duty cycle used in part 1. We also need more registers and states. Initially we had a register for the data but now we need more.

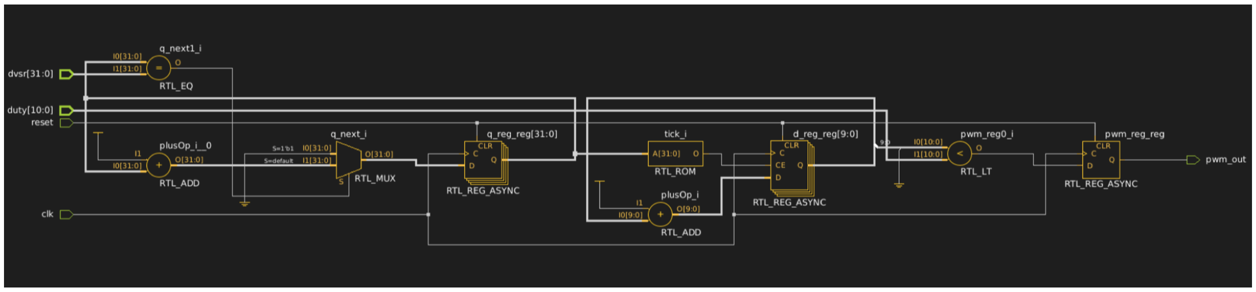

Fig 2. Enhanced PWM schematic

After the Enhanced PWM module is created, inside the top module of the design we have to make sure it meets the following requirements. The resolution, system clock frequency, pwm frequency and the gradient change rate. We also need to set up conditions for switches. Using a process block we will create a counter that will adjust the frequency of the system clock and another process block to change the duty cycle.

Part 2:

In this part of the lab we will reuse the enhanced PWM module and alter the top module to implement a sine function that will change the duty cycle. In the top module we will create a function that will implement a loop that will calculate the angles needed for the sine wave function. We will still use the same two process blocks from part 1 but will need to add another process block that will specifically handle the sine function. When the duty cycle gets to 2^Resolution the duty linear will increase otherwise it will be 0.

Part 3:

Part 3 of this lab will use the enhanced PWM module and reuse parts of the top module from part 2. In this part we are creating a rainbow effect for the RGB values and to get this effect we can change their intensity level. In order to implement the rainbow effect we need to set up a counter for it and make sure that it uses all 256 values for the max resolution. Based on the clock frequency at the rising edge we want to capture 3 bits of the rainbow counter. This is because the color registers are only a single bit.

Part 4:

For this part we will design a module to implement rotation for a servo. We will have a position counter that will keep track of what location the servo rotation is at. We have a function that will track the number of cycles per second and it will return an integer value based on a rounded value.

Conclusion:

This was a very interesting and useful lab because it showed how we can setup a PWM and as well as implement a servo. There was a bit of confusion and bugs with parts 1 to 3. The bugs came from the fact that session 1 and 2 of the labs did things slightly differently for each module. The clock frequency also played a role in causing bugs for the demo. The RGB leds would illuminate but the full features were not working. In part 2 the RGB would get dim or brighter but very slowly.

Appendix:

Youtube: https://www.youtube.com/watch?v=d68v69CAIf0&feature=youtu.be

Github: https://github.com/csun-ece/fa22-e524-lab7-troykerim