# Project A

Thunderbird Taillight

TCES 330 Digital Systems Design Spring 2019

#### Authors:

Ryan Mateo

Hantha Nyunt

James Stevens

Submission Date: May 27, 2019

#### Table of Contents

[Project A](#_bookmark0) Table of Contents 1

[1: Design:](#_bookmark1) 2

* 1. Part I: Clock Module 2
  2. Part II: FSM Module 3
  3. Part I: Top-Level Module 5

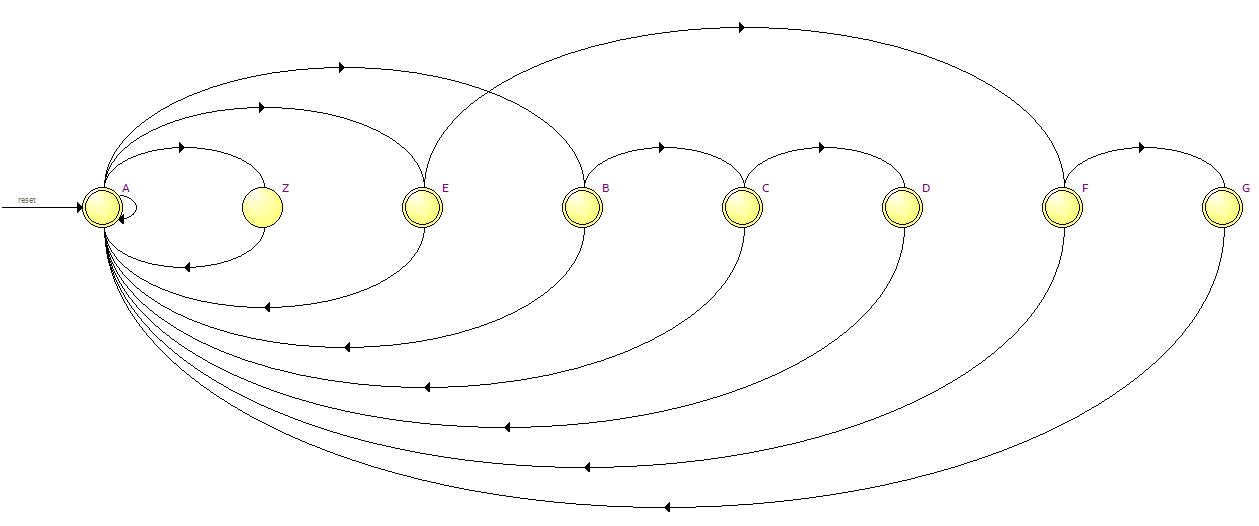
1. [Contributions](#_bookmark6) 6
2. Conclusions 7

**Project A**

The purpose of Project A is to implement the Thunderbird taillight design we made in TCES230 using SystemVerilog and the DE2-115 board. This design was from an older model Ford Thunderbird which had three left and three right taillights, in which the three blinkers when activated, flashed in a sequential order indicating the direction of the turn.

## Design

The first step is to understand the states that need to be executed and in what order. The diagram below is post-code and illustrates the finite state machine of the taillights. It should be noted that each state, in alphabetical order, is an instantiated enumerated typedef where A = 000 through to Z = 111.



#### Figure 1. Moore FSM Quartus

* 1. **Part I**

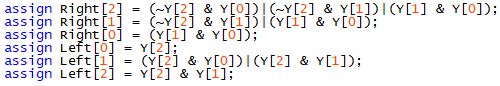
The first step is to create a 1 Hz clock module. This module is our basic 50 MHz clock module we used in previous homework assignments. We will send a parameter during instantiation that limits the clock to 1 Hz for use with the taillights.

* 1. **Part II**

The second objective is to create a Moore Finite State Machine. Using an input of 1 Hz from our clock module, we have three inputs; Left, Right, and Hazards. Those three inputs drive the outputs of the six lights.

This was accomplished using the enumerated states mentioned previously and 8 cases in a combinational logic block. Our default case is A, where A = 000 and that relates to our lights being off by default.

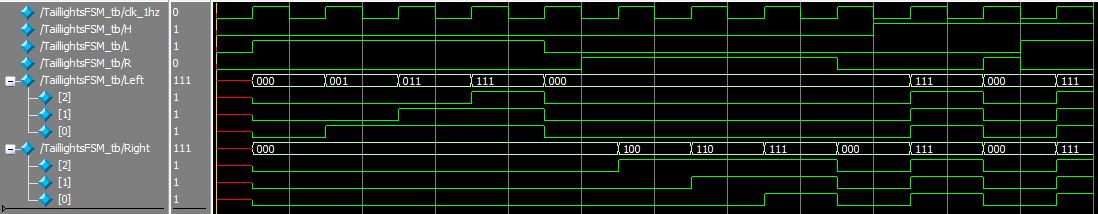
The assignment of the 3-bit output vectors is determined after the cases and is as follows:



#### Figure 2. Combinational Logic

The 3-bit vector Y, is an intermediate signal where Y equals the current state.

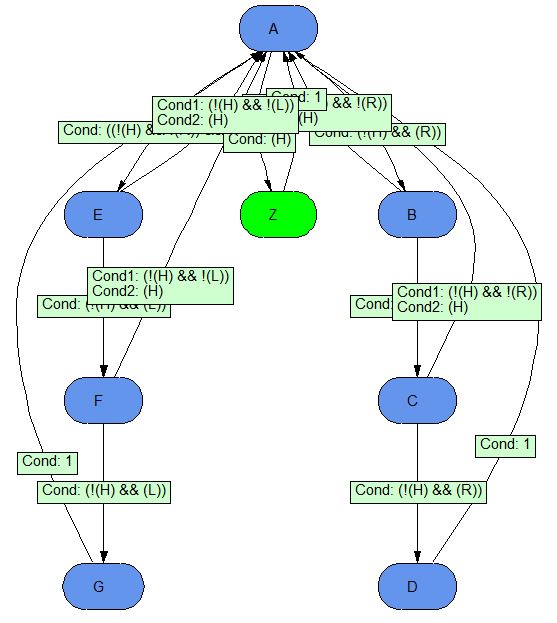
The testbench for the tailights consists of a timing diagram. It also includes a generated state machine diagram we can use for comparision with Figure 1, which is from Quartus.



#### Figure 3. Moore FSM Timing Diagram

This timing diagram shows a proper 1 Hz clock and a test case for all possible states of our three inputs. The outputs for the Left, Right, and Hazards are exactly as expected.

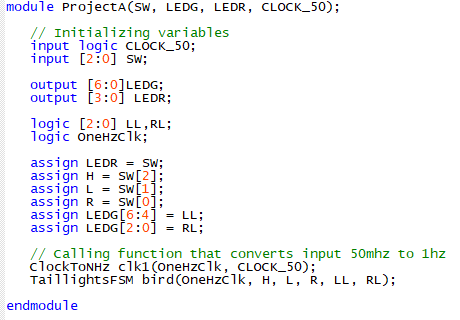
Comparing the Finite State Machine below with the Quartus one we see the same behavior.



#### Figure 4. Moore FSM (ModelSim)

* 1. **Part III**

The final part was to create a top-level module that utilized the clock from the board and with our parameter of 1 Hz, instantiated the Clock module. The clock then controlled the instantiation of the taillights finite state machine.



#### Figure 5. Top-Level Module

## Contributions

Ryan Mateo: Ryan contributed his clock module from previous assignments. He also wrote the FSM module with contributions from the other team members.

Hantha Nyunt: Hantha wrote the Top-Level module and the testbench for the FSM module with minor edits from the other members.

James Stevens: Wrote the report and conducted the ModelSim verification alongside the other team members.

All team members viewed and verified the ModelSim, Quartus, DE2-115 results and this report.

## Conclusion

This project was a well balanced assignment that tested the teams knowledge to apply all lessons learned throughout the quarter as well as coordination and teamwork efforts.