Homework_3 : Thunderbird Turn Signal

Introduction

In this lab, you will design a finite state machine in SystemVerilog to control the taillights of a 1965 Ford Thunderbird[1]. There are three lights on each side that operate in sequence to indicate the direction of a turn. Figure 1 shows the tail lights and Figure 2 shows the flashing sequence for (a) left turns and (b) right turns.

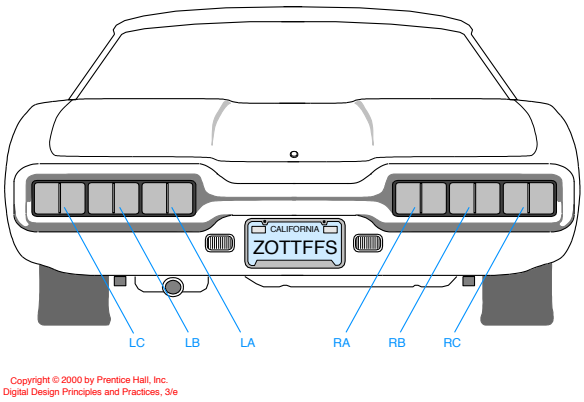

Figure 1. Thunderbird Tail Lights

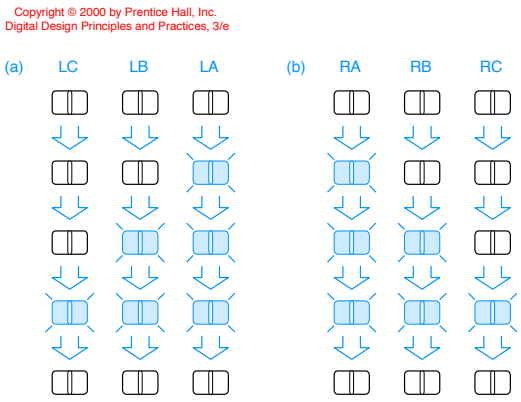

Figure 2. Flashing Sequence (shaded lights are illuminated)

This lab is divided into four parts: design, SystemVerilog entry, simulation, and implementation. If you follow the steps of FSM design carefully and ask questions at the beginning if a part is confusing, you will save yourself a great deal of time. As always, don't forget to refer to the "What to Turn In" section at the end of this lab before you begin.

[1] This homework is derived from a lab proposed by S. Harris and D.M. Harris in Digital Design and Computer Architecture ARM Edition.

1) Design

Your FSM should have the following module declaration:

```
module Homework_3(input logic clk,
                  input logic reset,
                  input logic left, right,
                  output logic la, lb, lc, ra, rb, rc);
```

You may assume that `clk` runs at the desired speed (e.g. about 1 Hz).

On reset, the FSM should enter a state with all lights off. When you press **left**, you should see LA, then LA and LB, then LA, LB, and LC, then finally all lights off again. This pattern should occur even if you release **left** during the sequence. If **left** is still down when you return to the lights off state, the pattern should repeat. **right** is similar. It is up to you to decide what to do if the user makes **left** and **right** simultaneously true; make a choice to keep your design *easy*.

Sketch a state transition diagram. Define your state encodings. **Hint: with a careful choice of encoding, your output and next state logic can be quite simple.**

Choose a set of state encodings. At this point, you could write state transition and output tables and then a set of next state and output equations.

2) SystemVerilog Entry

Create a new project named Homework_3. Choose the usual EP4CE22F17C6 FPGA.

Create a new SystemVerilog HDL file and save it as Homework_3.sv. Enter Verilog code for your FSM.

3) Simulation

Create a Homework_3_tb.sv file that convincingly demonstrates that the FSM performs all functions correctly. Simulate your FSM in ModelSim with your testbench.

You'll probably have errors in your SystemVerilog file or testbench at first. Get used to interpreting the messages from ModelSim and correct any mistakes. In fact, it's good if you have bugs in this lab because it's easier to learn debugging now than later when you are working with a larger system!

4) Hardware Implementation

Download your design onto the DE0-Nano board and test it in hardware.

Use DE0_Nano_SystemBuilder.exe to create a new project named DE0_Nano_3.

Add the Homework_3.sv file to the project.

For the implementation on the DE0-Nano board, you will have to derive a low frequency clock from the 50 MHz clock in order to be able to see something!

You'll need to connect the inputs and outputs of the Homework_3 module to push buttons (for the "left" and "right" signals) and LEDs (for the "la, lb…" signals).

Create a Signal Tap file to test your design.

Compile the design. Fix any errors that might be found.

Look at the compilation report. Under Analysis & Synthesis, look at the resource utilization summary. Check that the number of register and I/O pins matches your expectations.

Download the design to the DE0-Nano board. Test your design with Signal Tap and watch the LEDs.

What to Turn In

Validate your design on INGInious (the evaluation will mainly be done through INGInious and will be based on the number of attempts)

Upload on Moodle 2 files:

- A PDF file untitled "Homework3-Report-StudentLastName-StudentFirstName.pdf" that includes (follow the order and clearly mark each item):
    - Your FSM design, including a completed state transition diagram for your FSM. Scanned images are acceptable so long as they are easily readable.
    - Your Homework_3.sv code
    - Your Homework_3_tb.sv code
    - Image of your ModelSim waveforms demonstrating that your FSM performs all tasks correctly. Please display your signals in the following order: `clk`, `reset`, `left`, `right`, `lc`, `lb`, `la`, `ra`, `rb`, `rc`.
    - Your DE0_Nano_3.sv code
    - How many registers and I/O pins does your design use? Does it match your expectations?
    - Image of your Signal Tap waveforms demonstrating that your FSM performs all tasks correctly. Please display your signals in the following order: `clk`, `reset`, `left`, `right`, `lc`, `lb`, `la`, `ra`, `rb`, `rc`.
    - Briefly describe how you tested the system on the DE0-Nao board and whether it worked according to the specifications
- A ZIP file untitled "Homework2-Simu-StudentLastName-StudentFirstName.zip" that includes all the simulation files.

Deadline: Sunday, 16 October, 23:55.