Design of Digital Circuits and Systems, Lab 1

Parking Lot Occupancy Counter

Lab Objectives

This lab is a refresher on finite state machines (FSMs) that you learned about and designed in EE271 or CSE369.

Read the *whole* lab *before* starting on any work. If you have any questions at any point, first double-check the lab document. If you still cannot find the answer, you should ask a TA.

Task #1 – Parking Lot Occupancy Counter

Consider a parking lot with a single gate for both entry and exit. To keep track of the occupancy of the lot, we decide to use **two photosensors** to track when cars enter or exit, as shown in Figure 1.

When an object is between the photo transmitter and the photo receiver, the light is blocked and the corresponding output is asserted to `1`. By monitoring the events of both sensors, we can determine whether a car is entering or exiting, or even if a pedestrian is passing through! You may assume that two cars won't be entering/exiting at the same time.

For example, the following sequence indicates that a car entered:

1) Initially, both sensors are unblocked (*i.e.*, `{outer, inner} == 2'b00`)
2) Sensor `outer` becomes blocked (*i.e.*, `2'b10`)
3) Both sensors are blocked (*i.e.*, `2'b11`)
4) Sensor `outer` becomes unblocked (*i.e.*, `2'b01`)
5) Both sensors are unblocked (*i.e.*, `2'b00`)

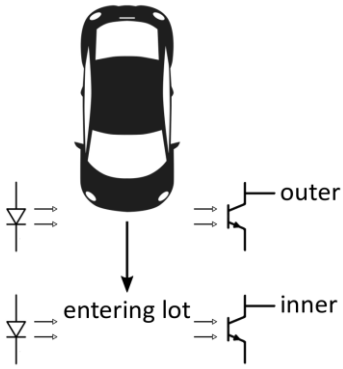

Figure 1: *The parking lot photosensor setup.*

Your task is to design a parking lot occupancy counter as follows:

1) Design and implement an FSM for the **car detection** with two input signals, `outer` and `inner`, and two output signals, `enter` and `exit`. The `enter` and `exit` signals assert true for one clock cycle when a car enters or exits the lot, respectively.
   a) You may assume that cars will not change direction while entering or exiting the parking lot.
   b) Make sure that your FSM does not detect pedestrians (how would the input pattern differ?).
2) Design and implement a **car counter** with two control signals, `incr` and `decr`, which increment and decrement the counter, respectively, when asserted.
   a) Assume that the maximum capacity of the parking lot is 16 spots.
3) Design and implement a module for the **parking lot occupancy**, which combines the car detection and the counter. Your system should have the following properties:

a) Use off-board switches (*i.e.*, not the `SW`s) to mimic the two photosensor outputs.

b) Display the current car count on the seven-segment displays `HEX1` and `HEX0`, with the following exceptions:
   i. If the counter reaches `16`, display "`FULL`" on `HEX5` – `HEX`**`2`**.
   ii. When the lot is empty, display "`CLEAR`" on `HEX5` – `HEX`**`1`** and the number "`0`" on `HEX0`.

c) Use 2 *off-board* LEDs (*i.e.*, not your LEDRs) to indicate the values of the `outer` and `inner` signals.
   i. A logical `1` should turn the corresponding LED on and a logical `0` should turn it off.

> Connecting off-board components requires the use of the GPIO pins of the DE1-SoC board. Please refer to the document "*`GPIO_Guide.pdf`*" for information on their usage in LabsLand.
>
> a) Wire the anodes of 2 LEDs to FPGA output ports.
>
> b) Wire the middle pins of the 3 switches to FPGA input ports of your choosing as your three inputs (`reset`, `outer`, and `inner`).

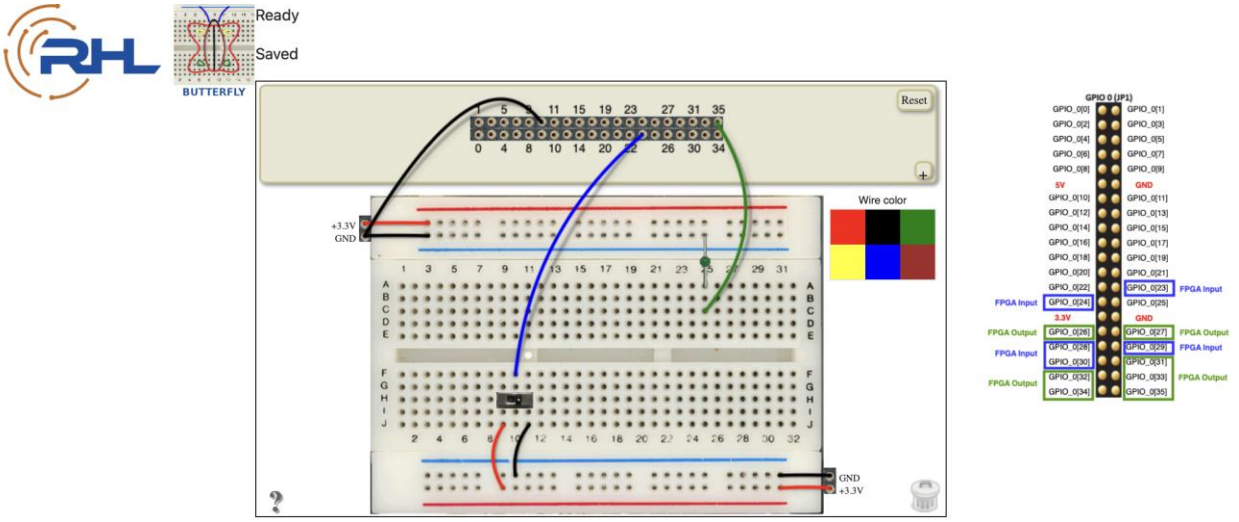

Figure 2: *LabsLand GPIO headers (from GPIO_Guide.pdf) for your reference.*

Lab Demonstration/Turn-In Requirements

⚠ Make sure that you read the document "*371_Assignments.pdf*" for details about what to expect during the lab demo and what we expect to be in your submitted lab report and SystemVerilog code. You can also read "*Lab_Report_Example.pdf*" for more detailed examples.

In-Person Demo

- Demonstrate your working parking lot occupancy counter on the DE1-SoC.
- Demonstrate your reset functionality.
- Be prepared to answer 1-2 questions about your design to the TA.

Lab Report (submit as PDF on Gradescope)

- Include the required **Design Procedure**, **Results**, and **Experience Report** sections.
- Don't forget to also submit your commented SystemVerilog files (`.sv`), including testbenches!

Lab 1 Rubric

Grading Criteria	Points
Name, student ID, lab number	2 pts
Design Procedure ▪ System block diagram and diagrams for car detection and counting	12 pts
Results ▪ Simulations for top-level and car detection (none needed for car counting) ▪ *Present state* included in all FSM simulations	10 pts
Experience Report	6 pts
SystemVerilog code uploaded	5 pts
Code Style	5 pts
LAB DEMO	20 pts
	60 pts