CS 223 Digital Design Laboratory Assignment 4

Smart Parking Management System

Preliminary Report Due: December 7, 2025 23:59

Lab Dates and Times

Section 1: December 8, 2025 Mon. 08:30-12:20 in EA-Z04
Section 2: December 9, 2025 Tue. 08:30-12:20 in EA-Z04
Section 3: December 10, 2025 Wed. 08:30-12:20 in EA-Z04
Section 4: December 8, 2025 Mon. 13:30-17:20 in EA-Z04
Section 5: December 12, 2025 Fri. 08:30-12:20 in EA-Z04
Section 6: December 9, 2025 Tue. 13:30-17:20 in EA-Z04

Location: EA-Z04 (in the EA building, straight ahead past the elevators)
Groups: Each student will do the lab individually. Group size = 1

Preliminary Work [30]

Note: This part must be completed before coming to the lab. Prepare a neatly organized report of your work and submit on Moodle before the start of the lab. Include your code as **plain text, not image!**

Smart Parking Management System

A large parking lot is being digitized to improve traffic flow and efficiency. At the entrance, drivers select their vehicle type using a control panel. The system directs vehicles to one of four zones or keeps them in a waiting queue if no slots are free. Each zone has a distinct average service time (i.e., average duration a car stays parked) after which the car departs and the slot becomes free again:

- Compact: $T_c = 5$ seconds
- SUV: $T_s = 8$ seconds
- Electric: $T_e = 10$ seconds
- VIP: $T_v = 12$ seconds

For simplicity, assume the controller processes vehicle arrivals sequentially, handling at most one vehicle admission per clock cycle. When multiple vehicles arrive simultaneously, they are queued and processed in order of arrival. When a car leaves (timer expires), the next waiting car of the same type is assigned immediately if present; otherwise the slot is marked available.

Design a **finite state machine (FSM)** to manage the parking zones and the waiting queue under the above constraints.

[4] Indicate which type of FSM (Moore or Mealy) you will implement and justify your choice. **Note:** Consider factors such as timing behavior, output stability, and implementation complexity when making your decision.

[10] Provide the state transition diagram, state encoding, state transition and output tables, next-state and output equations, and the FSM schematic.

[4] Determine the minimum number of flip-flops required by your encoding.

[4] Redesign your outputs using decoders to simplify the combinational logic. Use a 3-to-8 decoder to generate one-hot encoded state signals from your state register bits, then express your output functions in terms of these decoded signals. This approach often leads to simpler sum-of-products expressions compared to direct Boolean minimization of state bits.

[8] Write a SystemVerilog module implementing your FSM and a self-checking testbench that verifies functionality.

Smart Parking System on FPGA [70]

Implement your design on the Basys3 FPGA board with the following constraints:

1. Use four of the on-board push buttons to add vehicle arrivals: Compact, SUV, Electric, VIP.
2. Display the *current number of parked vehicles* (0–4) on the 7-segment display in decimal.
3. Use the 16 on-board LEDs to indicate zone status using the pattern `CCC.SSS.EEE.VVV.`:
   Solid ON → zone available, Blinking (>1 Hz) → zone occupied. The dot positions are off.
4. The system must automatically allocate queued vehicles when a zone becomes free according to the timing constants above.

Evaluation:

[10] Display Test: The 7-segment display must correctly show the current number of parked vehicles (0-4) in decimal format. The display should update immediately when vehicles enter or leave zones.

[10] LED Indication Test: Each zone's LEDs must correctly show solid ON when available and blink at approximately 1-2 Hz when occupied. The TA will verify the LED patterns match the specified `CCC.SSS.EEE.VVV.` arrangement.

[10] Functionality Test: Button presses must correctly trigger vehicle arrivals, with proper queueing when zones are occupied. The system should handle simultaneous button presses gracefully and process queued vehicles in order.

[40] State Machine Test: Comprehensive evaluation of FSM behavior including: correct state transitions, proper timing adherence (vehicles leave after specified durations), queue management (FIFO order), idle state behavior when all zones are empty, and correct resumption when new vehicles arrive. The instructor will test edge cases such as rapid button presses and mixed vehicle types.

Clean Up

1. Clean up your lab station and return all parts and the trainer board.
2. CONGRATULATIONS! You are finished with Lab #4 and are one step closer to becoming a computer engineer.

Notes

- Advance work on this lab, and all labs, is strongly suggested.
- Be sure to read and follow the Policies and other related material for CS223 labs, posted in Moodle.

Lab Policies

1. There are three computers in each row in the lab. Don't use middle computers, unless you are allowed by the lab coordinator.
2. You borrow a lab-board containing the development board, connectors, etc. in the beginning. The lab coordinator takes your signature. When you are done, return it to him/her, otherwise you will be responsible and lose points.
3. Each lab-board has a number. You must always use the same board throughout the semester.
4. You must be in the lab, working on the lab, from the time lab starts until you finish and leave (bathroom and snack breaks excepted). Absence at any time is counted as absence from the whole lab that day.
5. No cell phone usage during lab. Tell friends not to call during the lab hours–you are busy learning how digital circuits work!
6. Internet usage is permitted only to lab-related technical sites. No social media, email, or games.
7. If you come to lab later than 30 minutes, you will lose that session completely.

8. When you are done, do not forget to submit your code as a .txt file to the moodle, otherwise your assignment will not be graded.

Recommendations

- Avoid touching FPGA pins directly by hand. Discharge static electricity before handling.
- The white board you set up your circuit on is called a "breadboard". Research how its pins are internally connected.
- Connect power pins (V_{cc} and GND) last, after checking all other connections.
- Use a consistent wire color convention (e.g., red for V_{cc}, black/white for GND).
- If an LED is dim or an IC is hot to touch, check power pin connections for shorts or reversed polarity.