Connor Aksama
EE 371
January 13, 2023
Lab 1 Report

Procedure

This lab is comprised of the design and testing of a parking lot occupancy simulator. The sequential system was designed in System Verilog, tested in ModelSim, and finally run on an FPGA. The sensors and system reset was simulated using hardware connections to GPIO ports on the given circuit board.

Task 1

I approached this task by following the given development procedure outlined by the lab specification. First, I designed the FSM for the parking lot sensors, which took sensor readings as input, and output signals that indicated whether a car was entering or exiting the lot. Next, I designed a five-bit increment/decrement counter to keep track of the number of cars. Then, these modules were combined in a separate module, which handled GPIO connections, clock setup, and connection between modules. Each of these modules were individually tested by creating and examining waveforms in ModelSim.

Finally, I wired the breadboard circuit in the virtual lab environment with the necessary switches and LEDs, then uploaded the completed System Verilog code to the FPGA to perform final tests.

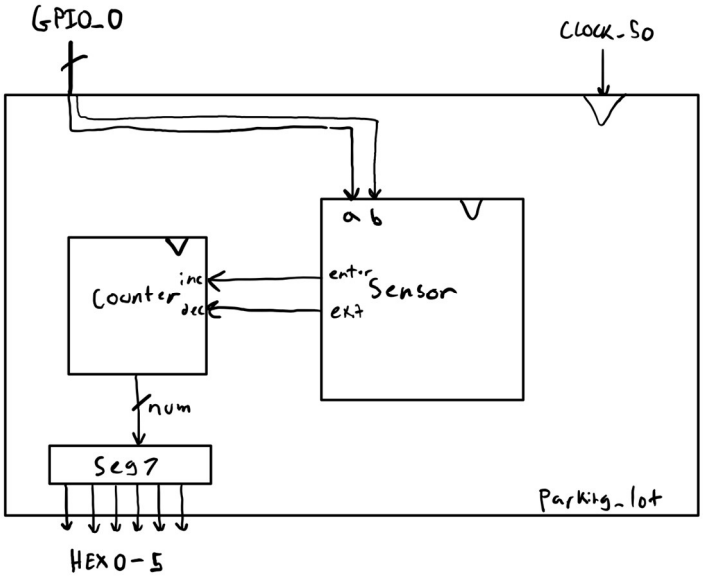

Figure 1. Parking Lot System Block Diagram

This block diagram illustrates the major components of the parking lot system and shows the important connections between modules.

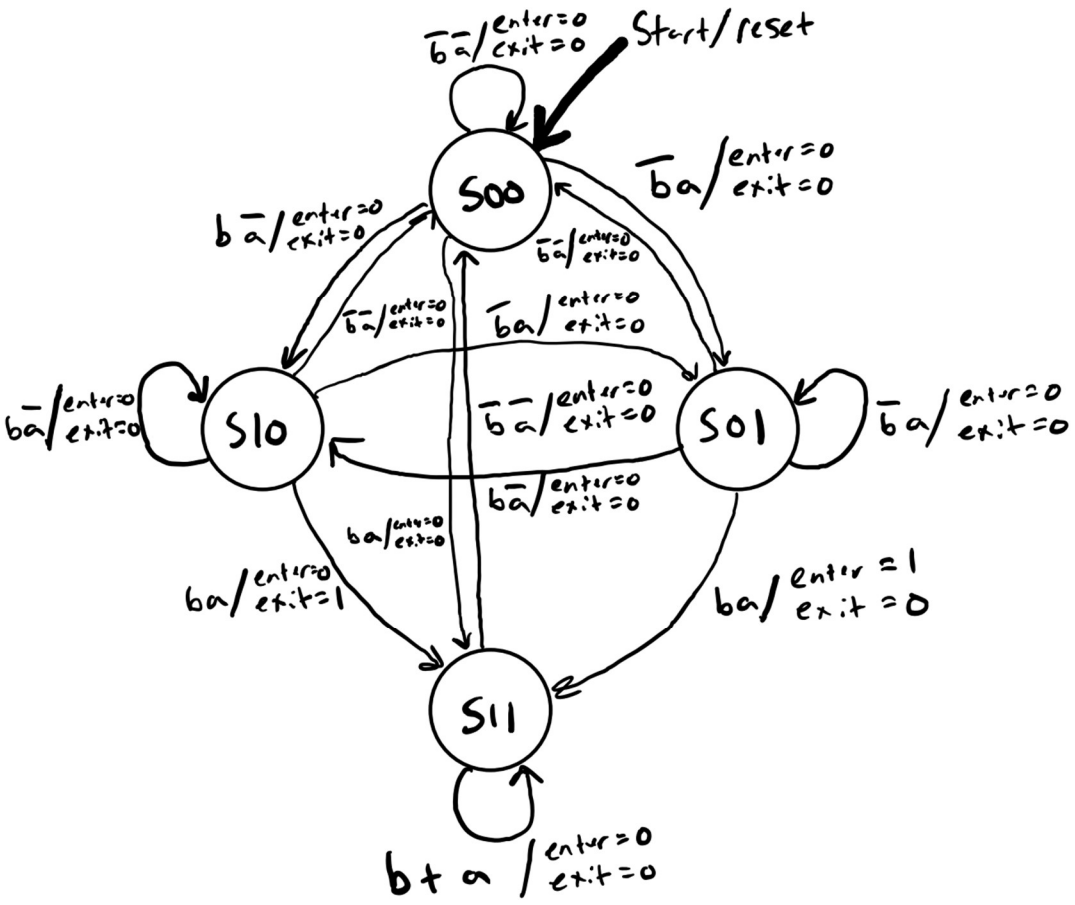

Figure 2. Sensor Module Finite State Machine

This finite state machine defines the behavior for the sensor module of the parking lot system. The signals “b” and “a” correspond to the output of each sensor. An enter or exit signal is only raised if both sensors start low, exactly one of sensors is high for at least one cycle, then both sensors are high. This prevents a singular car from artificially altering the count by moving in and out of one of the sensors.

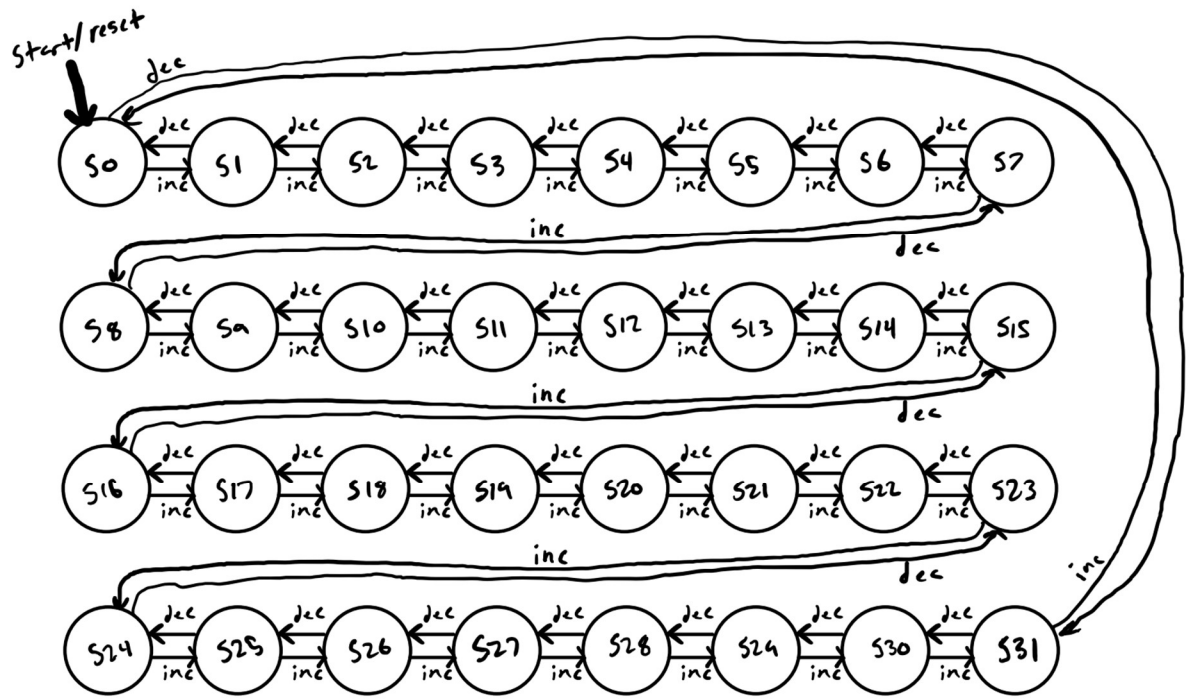

Figure 3. Five-Bit Counter Finite State Machine

The finite state machine for the five-bit counter module. On an “inc” signal, the counter output increases by one, and on a “dec” signal, the count decreases by one. At its maximum value of 31, an “inc” signal causes the output to wrap to 0. Similarly, at its minimum of 0, a “dec” signal causes a wrap to an output of 31. If neither signal is raised, the count remains constant.

Results

Task 1

The following are screenshots from ModelSim simulations for each module used in the parking lot system.

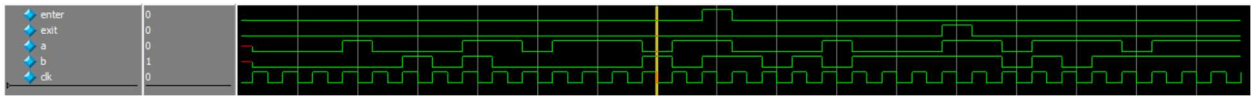

Figure 4. Sensor Module Waveform

This simulation runs through each possible transition between sensor output permutations (e.g. 00->00, 00->01, 00->10, etc.). The simulation demonstrates that the “enter” signal is only raised when transitioning from sensor output 01->11 and the “exit” signal is only raised when transitioning from 10->11.

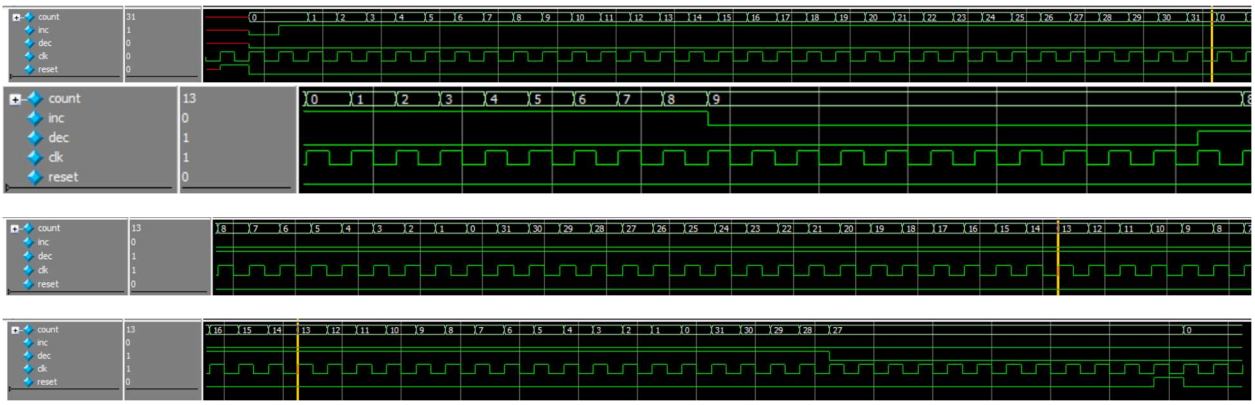

Figure 5. Five-Bit Counter Waveform

This simulation demonstrates that the counter can count from the minimum value 0 to the maximum 31 when “inc” is raised, wrap down to 0 from 31, and that an arbitrary value stays constant when no input is raised. Similarly, the simulation shows that the counter is able to count from the maximum value 31 down to the minimum 0, wrap up to 31 from 0, and stay constant with both inputs low. The simulation also shows the functionality of the reset signal.

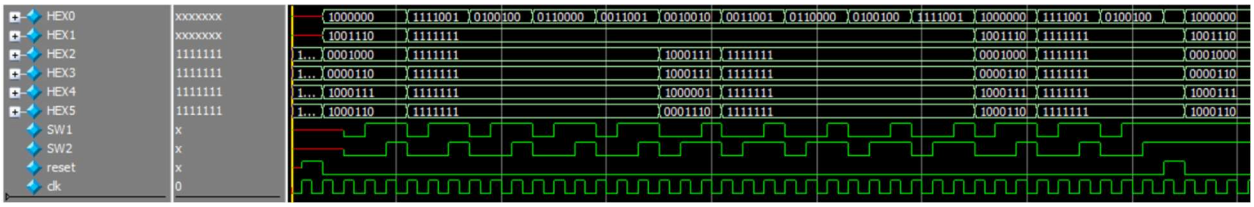

Figure 6. Parking Lot Module Waveform

This simulation shows that on reset, “CLEAR 0” is shown on the HEX displays, then shows only the count when the occupancy increases with appropriate sensor input sequences. When the lot is full, “FULL 5” is displayed. The simulation also shows the correct displays for when the occupancy is decreasing.

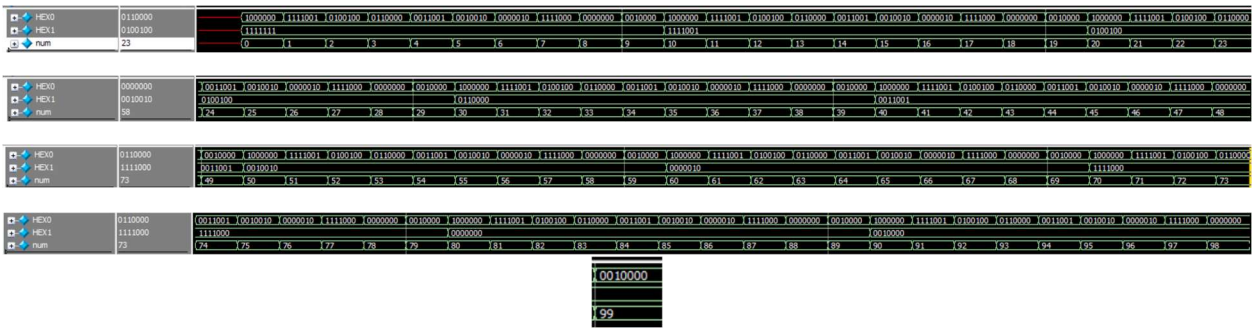

Figure 7. Double Seven Segment Display Waveform

This simulation demonstrates that the correct HEX displays are shown for each valid 2 digit decimal number input (0-99).

Final Product

The final product of this task was a working simulation of a parking lot occupancy system that could determine if a car was entering or exiting a lot and count the number of cars present in the lot – all as required by the lab specification. This count was also shown on the HEX displays along with a message “CLEAR” or “FULL” depending on whether the lot was empty or at capacity. This system was simulated using physical switches connected to GPIO ports to mimic the sensors of the system.

Appendix: SystemVerilog Code

(See next page)