CS M152A Project 4

Design a Parking Meter

In this lab, you will design and implement a finite state machine (FSM) for a parking meter.

Introduction:

In this lab, you are required to design an FSM to model a parking meter that simulates coins being added and displays the appropriate time remaining.

System Specifications:

The specifications for the “parking_meter” module have been described below. The input buttons represent different coin denominations and the seven-segment LED display will display the time remaining before the meter expires in seconds.

The inputs to the system have been listed in the table below

Inputs	Function
add1	add 60 seconds
add2	add 120 seconds
add3	add 180 seconds
add4	add 300 seconds
rst1	reset time to 16 seconds
rst2	reset time to 150 seconds
clk	frequency of 100 Hz
rst	resets to the initial state

As soon as a button is pushed, the time should be added immediately. The output is modeled as 4 seven segment displays which display the time remaining. Each add and reset button is high for at most one clock cycle.

- In the **initial state**, the seven-segment displays should be flashing 0000 with period 1 sec and duty cycle 50% (on for 0.5 sec and off for 0.5 sec). - When any add button

(add0, add1, add2, or add3) is pressed, the display adds to the corresponding time and starts counting down.

- When less than 180 seconds are remaining, the display should flash with a period of 2 seconds and a 50% duty cycle. You should have alternate counts on the display like 180, blank,178, blank, 176, …). Make sure you blink such that even values show up and odd values are blanked out.
- When the time has expired, the display should flash 0000 with period 1 sec and duty cycle 50% (on for 0.5 sec and off for 0.5 sec).
- For example, if add4 is then pushed, the display should read 300 seconds and begin counting down (at 1 Hz). When the timer counts down to 180 seconds and add2 is pushed, the display should then read 300 seconds (120 + 180) and continue counting down. If rst1 goes high, then the display gets reset to 16 seconds and starts flashing accordingly while counting down.
- The max value of time will be 9999 and any attempt to increment beyond 9999, should result in the counter latching to 9999 and counting down from there.
- Use input clock(**clk**) frequency of 100 Hz
- Include a global input reset (**rst**) which takes the FSM to the **initial state**.
- Do not account for multiple inputs being pressed at the same time.
- Though you do not have access to the FPGA, you will be required to design the output module which displays the time on the 4 seven segment displays for full credit.

Output Module:

The output module consists of a seven-segment vector **led_seg(7-bit vector)** which displays the actual value fed to the 4 segments corresponding to the digits being displayed. The order of the mapping is from CA to CG with CA being the most significant bit. (refer to the link below)

The anodes driving each of these segments are (one bit signals) **a1,a2,a3,a4**. Please refer to the Xilinx Nexys 3 reference manual to design the seven-segment display module. You will need to multiplex the seven segment displays with the anodes as would be required in actual hardware.

https://reference.digilentinc.com/reference/programmable-logic/nexys-3/reference-manual

Include 4 other output ports (**val1, val2, val3, val4**) each a 4-bit vector, which displays the actual digit in BCD (binary coded decimal) corresponding to each of the segments. You will be given partial credit for these ports even if your seven-segment implementation is not accurate.

Module Input/Output Interface

```
module parking_meter(
    input             rst,        // Global Reset
    input             clk,        // 100 Hz Clock
    input             rst1,       // Input to reset count to 16  seconds
    input             rst2,       // Input to reset count to 150 seconds
    input             add1,       // Input to increment count by 60  seconds
    input             add2,       // Input to increment count by 120 seconds
    input             add3,       // Input to increment count by 180 seconds
    input             add4,       // Input to increment count by 300 seconds
    output reg        a1,         // Anode AN0 (N16)
    output reg        a2,         // Anode AN1 (N15)
    output reg        a3,         // Anode AN2 (P18)
    output reg        a4,         // Anode AN3 (P17)
    output reg [6:0]  led_seg,    // Cathodes [Order: CA, CB, CC, CD, CE, CF, CG]
    output reg [3:0]  val1,       // Ones digit of timer count
    output reg [3:0]  val2,       // Tens digit of timer count
    output reg [3:0]  val3,       // Hundred digit of timer count
    output reg [3:0]  val4        // Thousand digit of timer count
);
```

Deliverables

When you finish, the following should be submitted for this lab:

1. **Verilog source code** for the "parking_meter" module. The file should be named exactly as "parking_meter.v". If you are using sub-modules, define them within parking_meter.v. Please DO NOT submit them as separate files.
2. **Verilog testbench** you used to evaluate your design. Note that your testbench is graded based on the correctness of the waveforms generated in your report. Please name the file "testbench_UID.v" where UID is your UCLA ID. Make sure you test **all the** special cases.
3. **Lab Report**: Please refer to the syllabus for the basic components of your lab report.
    1. Include the FSM diagram and explain the states and transitions of your FSM.
    2. Explain how you test your design and include simulation waveforms
    3. Schematics can be generated from ISE but please explain how your Verilog code results in the RTL generated. Include the 'Design Summary' section of the synthesis report and the summary of your implementation (map) report and write 1-2 lines on the conclusions you draw from these reports. requirement
    4. Please name your report "report_UID.pdf" where UID is your UCLA ID.
4. **Video**: Please refer to 'Syllabus' for details. Please name your video "video_UID.xxx" where UID is your UCLA ID.

Submission Checklist:

- There is no late submission for this project.
- It is recommended that you have your submission ready an hour or two before the deadline so that you do not face problems due to long upload times etc.