Exp 13: 3-Bit FSM-Based Counter

Objectives:

- To gain experience designing Finite State Machines configured as counters
- To gain experience using the Verilog behavioral method for implementing FSMs

Background: Once you gain a bit of experience working with FSM, you'll find that the only complicated step in implementing FSM is in the design of the state diagram. Generating the state diagram is always the first step in any FSM design that you actually plan on working properly. But you're in luck because just about the simplest FSM to design is a basic counter. The counter in this activity is a basic counter but it does have a special count sequence in a feeble attempt to make this FSM more exciting. Maybe it's not as simple as advertised.

Assignment: Design a 3-bit counter that counts in the following sequence: 0,2,4,5,7,0,2… Your FSM should contain a count enable input (CEN) that allows the counter to progress through the sequence: connect the count enable input to the right-most button on the development board. Use Verilog behavioral modeling to implement your design. Connect the counter outputs to the 7-segment display input as somewhat indicated in Figure 21. For this circuit, you should start by first generating a state diagram. You must include the CEN input in your state diagram as opposed to controlling the CLK with external logic as you did with the shift register circuit. Keep in mind that your FSM is going to use the S_CLK for its synchronous elements.

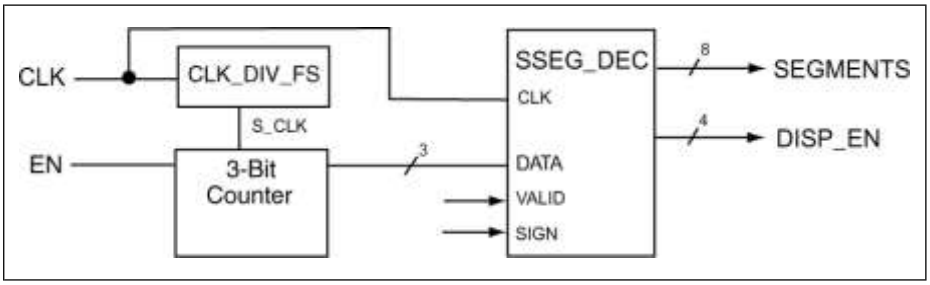

Figure 1: The high-level diagram of the circuit you'll implement in this lab activity.

Questions:

1. How many flip-flops did this FSM design use? Recall that even if you did not explicitly create a flip flop, the synthesizer would still create flip flops for you based on your behavioral model.

2. How hard would it to add four more numbers into the counting sequence using the Verilog behavioral approach to FSM design?

3. What would happen if this FSM were to suddenly find itself displaying the countof '3'?

Exp 14: Sequence Detector FSMs

Objectives:

- To design and implement a FSM that can be used for something relativelyuseful.
- To gain more experience using the Verilog behavioral method for implementing FSMs

Background: The design of sequence detectors using FSMs is one of the more basic uses for a FSM. The designs are not overly complicated yet they provided immediate gratification to the FSM designers. In order to convince you that FSMs really work and they're actually on the cool side, you'll be implementing a FSM in this lab activity that acts as a sequence detector.

Assignment: Design a sequence detector that will detect the provided sequence. Make your FSM a Moore-type FSM the resets when the one of the desired sequences are found. Drop your FSM designs into the circuit shown in Figure 22. The two special sequences are listed below.

- If BT0 is pressed: 0 1 1 1 0 1
- If BT0 is not pressed: 0 1 1 0 0 1

For this design, the SEQ_DVR modules use the values on the switches to generated inputs to the FSM. The BC_DEC module simply displays an annoying message when both the correct and incorrect sequences are found.

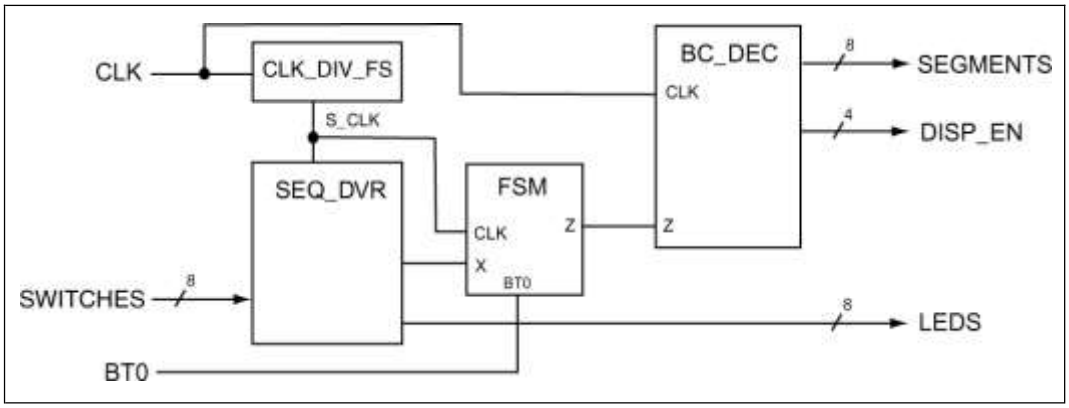

Figure 2: The top-level circuit for testing your sequence detector.

Questions:

1. Describe at least two applications where sequence detectors could potentially be useful.

2. Provide Mealy-type state diagrams for the resetting version of the FSM you designed in this lab activity.

3. Briefly explain the function of the SEQ_DVR box in Figure 22. You'll need to take a look at the Verilog model for this question.