1. Construct a 30-second down counter (timer) with pause function. When the counter goes to 0, all the LEDs will be lighted up. You can use one push button for reset and one other for pause/start function.

1.1 Write the spec (inputs, outputs, and function table) of the design.

1.2 Draw the related block/logic diagram.

1.3 Use an FSM to implement the function of pause/start function. Use one LED to represent current state.

1.4 Use Verilog to implement 1.3 and verify the design with simulation results.

Design specification:

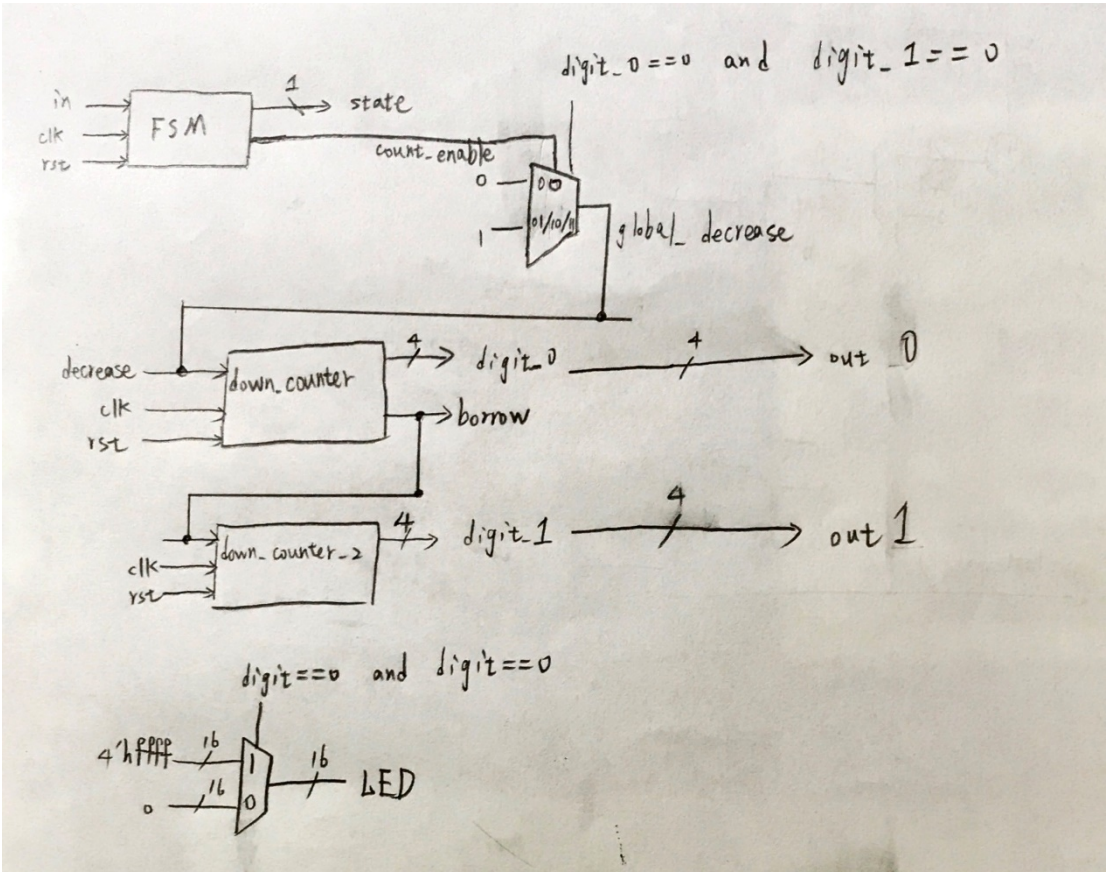
Input: in, clk, rst

Output: [3:0]out1, out0

Output: state, count_enable

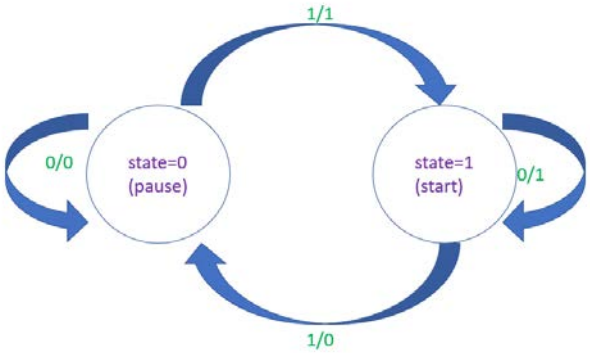
Output: [15:0]LED

Block diagram:



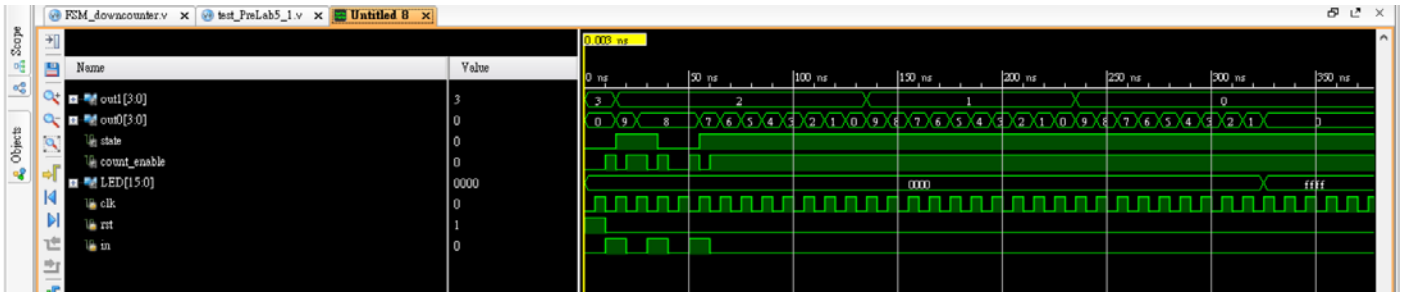
Design Implementation:

state diagram:



According to the current input to decide the next state as shown in the above state diagram. For example, if state = 1 (start) now, and in = 1, state would change to 0 in the next moment. If in = 0, it will keep its state, which is state = 1 (start). With this FSM implementation, I can decide if the timer is counting or in the state of stop. And output state would be connected to one LED to show the current state. If LED is on, it means the timer is counting. If it's off, it means the timer stops counting.

Discussion:



As waveform shows, at first, its initial state is 0 (pause). Then I set in = 1, it starts to count. And I set in = 0, it stops counting. And I set in = 1 again, it counts to the end which is 0. In the meantime, LEDs are all on. Therefore, according to the simulation result, the design for FSM is correct so that I can implement it on the FPGA board.

Conclusion:

It's my first time to write FSM in Verilog, which is difficult at first. However, when I read the handout given by professor and Verilog code, I understand how to express FSM in Verilog. Then, it becomes easier for me to write Verilog myself. It also helps me have the basic understanding of Lab5, which is a great preparation.