EE 209 Lab 5 – Enter the Code

1 Introduction

In this lab you will design a state machine to detect a particular pattern/sequence of bits (0110) considering both Mealy vs. Moore implementation.

2 What you will learn

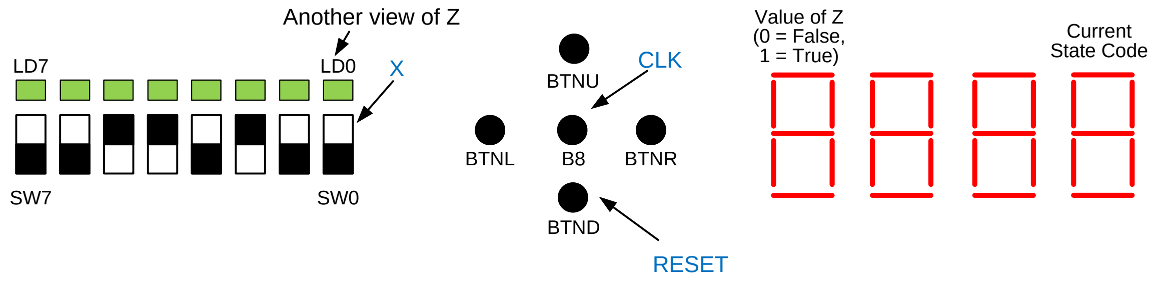
This lab is intended to reinforce your understanding of state machine design.

3 Background Information and Notes

Refer to the lecture slides regarding how to perform state machine design.

For testing purpose we will again use the FPGA. Serving as the X input we will use SWITCH0 (SW0). For the output Z we will display it in 2 ways: as a decimal value on the left-most 7-segment display (digit 0 or 1 for Z's value) and LED0 (LD0) (Z=1 => LED on, Z=0 => LED off). So you can know what state you are in we will display the 2-bit state code {q1,q0} as a decimal digit (values 0-3) on the right-most 7-segment display.

Finally, because state machines transition on every clock edge we need a new value of X for every clock. However, we wanted you to be able to choose the value of X to plug in next and not have to do it at a regular interval. Thus, instead of feeding in a constant frequency clock signal, we will just use B8 (Middle Button) as the clock. Thus, you would set X (SW0) to the desired value and then press BTN0 to generate a clock and see the state update. You can then take as much time as you like to choose the next value of X (move the switch or leave it as is) an then press BTN0 again to generate another clock edge and state transition. In this way you don't have to move the SW0 to quickly or slowly. To reset the FSM you need to press and hold RESET (BTND) and then press the CLK button (B8).



4 Prelab

None.

5 Procedure

You will need to implement a state machine that takes one input bit, X, and produces one output bit, Z. Z should be 1 for a clock cycle whenever the last four bits on X are 0110.

1. Draw a state diagram (on the next page) for this design assuming a MEALY style output.
2. Download the project skeleton from our website and unzip it. It should contain
    a. The module definition of the sequence detector in **seqdet.v**. We bring the state bits: q1 and q0 out as outputs so that we can display the state combination on the FPGAs 7-segment display. Normally, the state is just an internal signal and is not brought out as a true output, but for debugging we did so.
    b. A partially completed testbench that you can complete to simulate your design and ensure it's correctness.
    c. A completed top-level file that connects the switches, buttons, LEDs and 7-segment displays of the FPGA into/out of your actual sequence detector.
3. Once you have the state diagram complete, find the logic implementation on paper/pencil using the methods we described in class. You may use any state assignment (codes) you desire but the initial/reset state must have the code of all 0's. Use a minimal number of flip-flops (no more than needed). You should also use D-Flip flops.
4. Open **seqdet.v** You will need to add the wires, gates, and instances of **dff1** (defined in dff1.v) for your flip-flops to produce the output Z.
5. Simulate your design and ensure its output matches your expectation. In the Simulation view: select the testbench file (**seqdet_tb.v**). Add input values for X waiting for a clock cycle (T ns) after each value. Come up with a sequence that will test each transition in your machine. Then run the simulation and ensure it works correctly.
6. In the Implementation view: synthesize, implement, and generate the programming file for your top-level design. Be sure you click on the **seqdet_top.v** file before you start the synthesis process.
7. Connect an FPGA board, start the Digilent Adept tool, find the seqdet_top.bit file and program the FPGA. Use SW0 as X and BTN0 as the clock to ensure your state machine works.
8. Answer the review questions where you will consider a Moore style output approach.

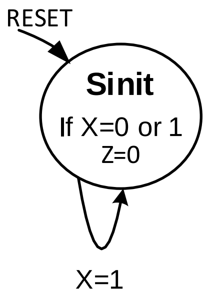
6 Review / Lab Report

Name: ___________________________________ Score: _________

Due: _____________

(Detach and turn this sheet along with any other requested work or printouts)

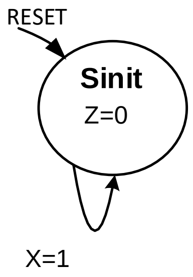
1. Reprint your Mealy-output approach state diagram (Mealy outputs are based on the input and can be shown under some 'if' statement depending on the input)



2. Show your state code assignment for the states above.

3. How many states did your Mealy-style approach require? _________

4. Go back and consider a Moore-style output approach for Z. Draw the required state diagram (you don't actually have to design the logic for this approach, just the state diagram). How many flip-flops would this design require? What size K-map (3-var, 4-var, 5-var, etc. would be required to find the logic for the D-inputs of the flip-flops)?



How many flip-flops will this Moore-Design require? ______

What size K-Maps (3, 4, 5-var, etc.) would your D-inputs require? ________

Turn in the following items:

- Completed seqdet.v and seqdet_tb.v on our website
- The answers to the questions above on this page

7 EE 209 Lab 5 Grading Rubric

Student Name: ____

Item	Outcome	Score	Max.
Mealy-Approach			
• Correct State Diagram	Yes / No		2
• Correct number of flip-flops	Yes / No		0.5
• Correct next state logic	Yes / No		3
• Correct output logic	Yes / No		0.5
• Correct operation (via simulation)	Yes / No		1
Moore-Approach			
• Correct State Diagram	Yes / No		2
• Correct number of flip-flops	Yes / No		0.5
• Correct size of K-Maps	Yes / No		0.5
SubTotal			10
Late Deductions (-1 pts. per day)			
Total			10
Open Ended Comments:			