SANTA CLARA UNIVERSITY	ELEN 21 Spring 2017	Dr. Sally Wood, Dr. Radhika Grover

Laboratory #7: Slot Machine Inspired Game

For lab sections May 23-26, 2017

I. OBJECTIVES

In this laboratory you will:

- Learn to use the design wizard to create logic components
- Learn to use a comparator and a counter in a circuit
- Use a multiplexor to switch multi-bit input

PROBLEM STATEMENT

A typical slot machine is shown to the left. It is a game of chance is which the player initiates the spinning of three separate displays and tries to stop the display motion at a time when all three display images match.

A slot machine inspired game will use two seven segment displays in place of the three displays in the example. Each seven-segment display will show one of four custom designed symbols.

Status information about the game will be shown in a third seven segment display. When a button is pressed, the two displays showing the symbols begin to change at different rates. The displays continue to change as long as the button remains depressed. When the button is released, the symbols stop changing. The objective of the game is to try to stop when both displays show the same symbol.

- A "win" symbol is shown in the status display if the two displayed symbols match when the button is released.
- A "lose" symbol is shown in the status display if the two displayed symbols do not match when the button is released.
- While the button is depressed and the symbols are changing, a neutral symbol appears in the status display to indicate that the game is in progress.

The game has four speeds: test, slow, medium, and fast. At test speed the symbols change so slowly that it is easy to win and easy to test the function of the circuit. As the speed increases, the game becomes more challenging. The speed is selected using two input switches.

II. PRE-LAB

1. **Define your custom symbols and design your special symbol generator.**
    a. Refer to the seven-segment tutorial in the "Seven_Segment" file that was used in Lab 5.
    b. Copy the text file "bin_7seg_temp.txt " to "spec_7seg_temp.txt " to create a new module "spec_7seg" that will use the same input and output specification, but will generate your specially designed characters instead of 0-9 and A-F. For example, one of your special characters might have a small circle in the lower half using only segments c, d, e, and g.
        i. Use hex inputs 0, 1, 2, and 3 to select your four special symbol shapes for the spinning display used in the game
        ii. Use hex inputs d, e, and f for your status symbol shapes to indicate "win", "lose", and "in progress."
        iii. The other nine input possibilities may be used to add special features to your game.

2. **Design your display control circuit**
    a. The inputs to the two seven-segment displays which show the game symbols should each come from the outputs of a special symbol generator.
    b. The two least significant bits of the 4-bit inputs to the two special symbol generators should each come from a 4:1 Mux. (This means that four 4:1 Muxes are needed for your circuit).
    c. The outputs of the Muxes will be controlled by the "speed," which will be selected by two input switches. Decide which switch settings should correspond to each of the four speeds. (The MUX inputs will be designed in the lab.)
    d. How should the two most significant bits of the special symbol 4-bit input be connected so that the input to the special symbol generator is always 0, 1, 2, or 3?

IV. PROCEDURE

1. **Create new components:**
    a. Use the design wizard to design a 4-bit comparator that will create three outputs: equal, greater than, and less than. Go to libraries→megafunctions→arithmetic and select *lpm_comparitor*. In response to the design wizard prompts, select binary values, unsigned comparisons, and no pipelining.
    b. Use the design wizard to design an 8-bit binary up counter with a clock enable. Go to libraries→megafunctions→arithmetic and select *lpm_counter*. The binary up counter is a circuit that will count in binary from 00000000 to 11111111 and then start over again. The rate at which it counts is controlled by a clock input. The counter can only count in response to a clock input if it is enabled. In response to the design wizard prompts, select a width of 8, a count direction of up, plain binary, clock enable input, and then no other options.
    c. Use the design wizard to design an 8-bit down counter in the same way that the up counter was created. The only difference will be that the direction of the count is down instead of up. The counter will cound down from 11111111 to 00000000 and then start over.
    d. Add the component "clock_counter" to your circuit.

2. **Connect your circuit. Start with the circuit you created for the pre-lab.**

    Two symbol displays

    a. The input to "clock_counter" should be the system clock50.
    b. Connect the up counter and the down counter to your circuit.
        i. Connect one "clock_counter" output to the clock input of the up counter.
        ii. Connect the outher "clock_counter" output to the clock input of the down counter.
        iii. Connect both the clock enable inputs to the push button switch that will enable the display change.
    c. Selected bits from one of the 8-bit counters will provide 2 bits of the 4-bit input to one of the game display symbol generators. The selection will be made with two of the four 4:1 Muxes. The outher counter will provide input to the other game symbol display generator. Each 8-bit counter will have eight outputs named q7 q6 q5 q4 q3 q2 q1 q0 where q0 is the least significant bit.
    d. The up counter will supply inputs to two Muxes that provide the symbol code to one of the displays.
        i. In fast mode, one Mux will output q1 and the other will output q0.
        ii. In medium mode, one Mux will output q2 and the other will output q1.
        iii. In slow mode, one Mux will output q4 and the other will output q3.
        iv. In test mode, one Mux will output q6 and the other will output q5.

v. (You may adjust these later if you would like to change the speeds.)

e. The down counter will supply inputs to the other two Muxes which provide the symbol code for the other symbol display. They will be connected for the four speeds in the same way that was used for the up counter.

Status display

f. Add the comparator to your circuit and connect it to compare the 4-bit inputs of the two game display symbol generators.
g. Add the status display and a new special symbol generator for it.
h. Write the logic expressions for the inputs for the display as a function of the pushbutton and the comparitor equal output to show the "win", "lose", or "in progress" symbols.
i. Add logic to your circuit to control the display

Debugging aids

j. To make the circuit easier to debug, connect the clock inputs and the q0 outputs of both of your counters to LEDs so you can observe internal status if the circuit does not behave as you expect. This will allow you to see if the counters are counting and how fast they are counting even if the display controllers are not working correctly.

3. Test your circuit

a. Verify that the counters are NOT counting when the pushbutton is NOT depressed.
b. Verify that both counters are counting when the pushbutton is pressed.
c. In the very slow test mode, verify that the symbol display sequences are correct. One should display symbols in the order 0, 1, 2, 3, 0, 1, 2, ... and the other should display symbols in the order 0, 3, 2, 1, 0, 3, 2, ...
d. Verify that the status display works correctly.
e. Verify that the speed select switches work correctly.

4. Play the game

V. REPORT

1. Introduction, procedure, results, conclusions and references.
2. Include schematic, test plan, and results for each operation.
3. If you want to use 8 symbols instead of only 4 to make the game harder, list all the things you would need to change.