

Writing Code for Synthesis. Electronic Dice Game

PURPOSE - In this lab you will learn about how to write synthesizable Verilog code. As an application you will design an electronic dice game.

1. Designing an electronic dice game

In this laboratory you will design an electronic dice game.

The idea of the dice game is that two counters are used to simulate the roll of the dice. Each counter counts in the sequence 1,2,3,4,5,6,1,2,... After the "roll" of the dice, the sum of the values in the two counters will be in the range 2 through 12. The rules are as follows:

1> After the first roll of the dice, the player wins if the sum is 7 or 11. The player loses if the sum is 2,3 or 12. Otherwise, the player obtained on the first roll is referred to as a point, and he or she must roll the dice again.

2> On the second or subsequent roll of the dice, the player wins if the sum equals the point, and he or she loses if the sum is 7. Otherwise, the player must roll again until he or she finally wins or loses.

The inputs to the dice game come from two push buttons, Rb (Roll button) and Reset. Reset is used to initiate a new game. When the roll button is pushed, the dice counters count at a high speed, so the values cannot be read on the display. When the roll button is released, the values in the two counters are displayed, and the game can proceed. If the Win light is ON or the Win light is not on, the player must push the button again.

You will type (or cut and paste but correcting eventual/intentional ☺ misprints...) the Verilog files (posted on the lab webpage), which will implement the dice game and synthesize and implement it. Then, you will download the bitstream file to the lab boards and verify the game.

You will have to flow-chart of the Electronic dice game showing all the internal interconnect between components. **You will have to attach this diagram to the lab note-book you will turn in at the end of the semester!**

The output of the dice game will have to drive the one 7-segment LEDs of the BASYS board. The output W or L will indicate "U" in case that the player **wins**, "L" in case that the player **loses**, or "A" in case that the player has to play/roll again. When the game is reset the 7-segment LEDs display "0". Figure (**using the manual for the BASYS board & pg. 3 & 4 of Pegasus Board manual**) out what pin assignments you have to use to properly activate the 7 segments of each 7-segment LED. Also, assign the proper pins such that to use the RESET push button as the Reset of the game entity, the SPARE push button as Rb of the game entity. For the CLK_IN input in your game entity assign pin number 36, i.e. use NET CLK LOC =

P36 in the User Constraint File (ucf) of your project, which will have to be used for implementation!

Optional

It would be nice if you have displayed the two final numbers of the two dices any time when they are rolled (and why not also the numbers of the upper faces of the two dices while they roll in a flickering manner). For doing that you will have to modify the above Verilog files accordingly in order to drive the two 7-segment LED properly which are used to display "UI" when the player wins and "LO" when the player loses.

SUMMARY -- During this lab you learned general rules about how to write synthesizable code in Verilog and you implemented and verified an electronic dice game with Foundation Series.
