ECE 3300 Final Project

Hannah Smith
Electrical and Computer Engineering
California Polytechnic University
Pomona

Justin Phuc Le Nguyen
Electrical and Computer Engineering
California Polytechnic University
Pomona

Nicholas Gomez
Electrical and Computer Engineering
California Polytechnic University
Pomona

***Abstract*—This document covers the final project completed for the fall semester of ECE 3300. Our group used a NEXYS A7 board to implement a Verilog code of a simple dice game. The document outlines the rules of how the dice game is played and how players can win each round or overall win the game. It then delves into each of the Verilog modules and their distinct purposes. The paper describes how users interact with the NEXYS A7 board to partake in the simple dice game. Overall this document will thoroughly explain how we as a group transformed a simple dice game into an interactive code with the use of a NEXYS A7 board.**

***Index Terms*—dice, verilog, NEXYS A7**

I. Introduction

This semester we have studied digital circuit design and how to implement it using Verilog. This document will cover the final project assigned for ECE 3300 Digital Circuit Design Verilog, in which we were instructed to create a project that uses a NEXYS A7 board and implements different Verilog modules. For this project, our group decided to create a simple dice game. This document is organized to describe that dice game and how it is played. After that, each of the Verilog modules written to design that dice game is then described. From there, how the NEXYS A7 board is used and what the desired inputs and outputs are described within this document. The objective of this project is to present the knowledge we have gained throughout this semester as we learned about Verilog and began creating new designs.

II. Dice Game Overview

The Dice game involves two players, each rolling two dice to generate their own unique values to sum up. The sums are compared, and the player with the higher value is the winner. The player score is being displayed alongside the dice and updates after each roll. The winner's score increases, while the loser's remains unchanged.

III. Verilog Modules

In order to implement this simple dice game, several Verilog modules were written. This following sections outline each module and their functionality. The modules are as folows: driver, clock, dice game, seven-segment display, and vga.

A. Driver

The Driver module serves as the top-level module for the Dice game. This module includes four inputs for the clock pulses, clock reset, general reset, and a roll trigger in addition to seven outputs which include the segment outputs, the anode select, the hsync, the vsync, and red, green, and blue outputs. It consists of multiple connections between sub modules, starting with 4-bit wires (dice1, dice2, dice3, and dice 4) representing the values of four dice. Additionally, there are two 4-bit wires (player1wins and player2wins) indicating the total number of wins for player 1 and player2. A 1-bit wire (clk25) represents a clock signal with a frequency of 25 MHz, and another 1-bit wire (lock) indicates whether the clock is locked or not. The Dirver module uses the wires described abover to connect four different modules, with the first being the Clock Wizard (clk_wiz_0) which generates a 25MHz clock. The second module (DiceGame) connected manages the logic of the dice game, including rolling and determining the winner. The seven-segment displays module (SSD) is then connected to the dice and player score wires to display their values using the output signals segment outputs and anode select. Lastly, the module to control the VGA output (Dice VGA) is connected.

B. Clock

Clock management has a crucial role in this project. The "clk_wiz_0" IP core was used to provide flexibility in configuring the clock parameters. In this project, the IP was configured to operate as a Phase-Locked Loop (PLL) to meet the timing constraints. The output "clk_out1" was modified from the default 100MHz to 25.1 MHz to strategically align the clock frequency with our specific needs. This lower frequency ensures optimal performance and has advantages such as power efficiency and heat dissipation. This is an ideal frequency to control the VGA module so the output of this module is connected to DiceVGA in addition being connected to DiceGame. This module was written using the clocking wizard feature in Vivado.

C. Dice Game

The DiceGame module is the heart of this project. It plays the biggest role in this project because it generates random values for the dice. This module takes inputs such as the general clock, reset signal, and roll trigger to output the values of the four dice and the scores of two players (player1score and player2score). This module has a sub module called the Linear Feedback Shift Register (LFSR) to generate random values and is stored in wires die1_temp, die2_temp, die3_temp, and die4_temp. See below for more information. The DiceGame module then uses an always block managed by the clock and reset signal to sum the two pairs of dice and determine the winner of the round. If a roll trigger is activated during a

clock cycle, the total values of two pairs of dice are compared with a winner determined. This will set a value for another wire called the increment flag to ensure that the score only gets incremented once per role. If there was no roll trigger is activated, then the increment flag is reset. After the always block, the player scores and the dice values are assigned to their outputs using the assign statement. It is important to note that player scores increment on the next roll for the previous roll.

1) Linear Feedback Shift Register: As mentioned earlier the DiceGame module implements another module called the LFSR. This stands for linear feedback shift register and is used to generate random numbers. This module uses the same clock, reset, and roll trigger input signals as DiceGame. When reset the values of the dice will set to prewritten values between numbers 1 and 6. Within an always loop when roll trigger is active each number will then change to another value between the numbers 1 and 6. Each othese numbers is then assigned to the four dice and the remainder of the DiceGame module continues.

D. Seven-Segment Display

The purpose of the Seven-Segment Display module (SSD) is to display the numeric digits on the NEXYS A7 board using the 8 seven-segment display LEDs available. This module has multiple inputs to represent the clock pulse (clock), general reset (reset), and the different digits (in0 to in7). It outputs the appropriate segment values (segment_outputs) and selects which anode is needed (anode_select). The module has a state-machine approach to it and it is managed by three always blocks all controlled by the clock pulse and reset. The first block increments a value called digit select in order to determine which digit to display. The clock frequency is fast enough that to our eyes all digits will display however this always block is necessary since each digit needs to be output on its own since they each have their own input. The second always block manages the anode select which determines which spot on the seven-segment display the digit is displayed on. The third always block uses the digit select from the first always block in order to display the value of the input digit. This always block incorporates case statements to determine the correct seven-segment outputs for the possible digits. This module receives its inputs from the Driver module so that anode 0 is the first players score with anode 1 and 2 as their dice in addition to anode 7 as the second players score with anode 5 and 6 as their dice.

E. VGA

The last module (DiceVGA) controls the VGA output. This module has 6 inputs signals representing the clock pulse (clock), reset (reset), and the value of each dice (number1, number2, number3, and number4) in addition to 5 outputs representing color outputs (red, blue, and green) and vertical and horizontal sync pulses (hsync and vsync). The always block within this module starts with the logic to determine the position of four squares each with a number inside. This is the visual representation of a dice on screen. These will become the outputs for the colors red, green, and blue. From there the always block uses if statements to determine the timing of the for the vertical and horizontal sync pulses. This logic was used from a previous lab since it is specific to the NEXYS A7 board and output monitor being used.

IV. NEXYS A7 Board

The NEXYS A7 board is an FPGA developement board designed as a part of the Xilinx Artix-7 FPGA family. While it has many features, the ones used for this project are 2 of the 16 available switches, 1 of the 6 available buttons, and the 8 seven-segment displays. The switches all the way to the right are set to be the two resets (clock reset and reset) with the clock reset on the left. The center button is set to be the roll trigger. The seven segment display is then set to show from left to right the score of player 2 and their two dice then two digits turned off with the two dice of player 1 next with their score being in the last position. The game is played by just simply pressing the roll trigger button.

V. References

"Nexys A7: FPGA Trainer Board Recommended for ECE Curriculum," Digilent. https://digilent.com/shop/nexys-a7-fpga-trainer-board-recommended-for-ece-curriculum/