**EC311 Final Project Report**

Whack-a-Mole: FPGA



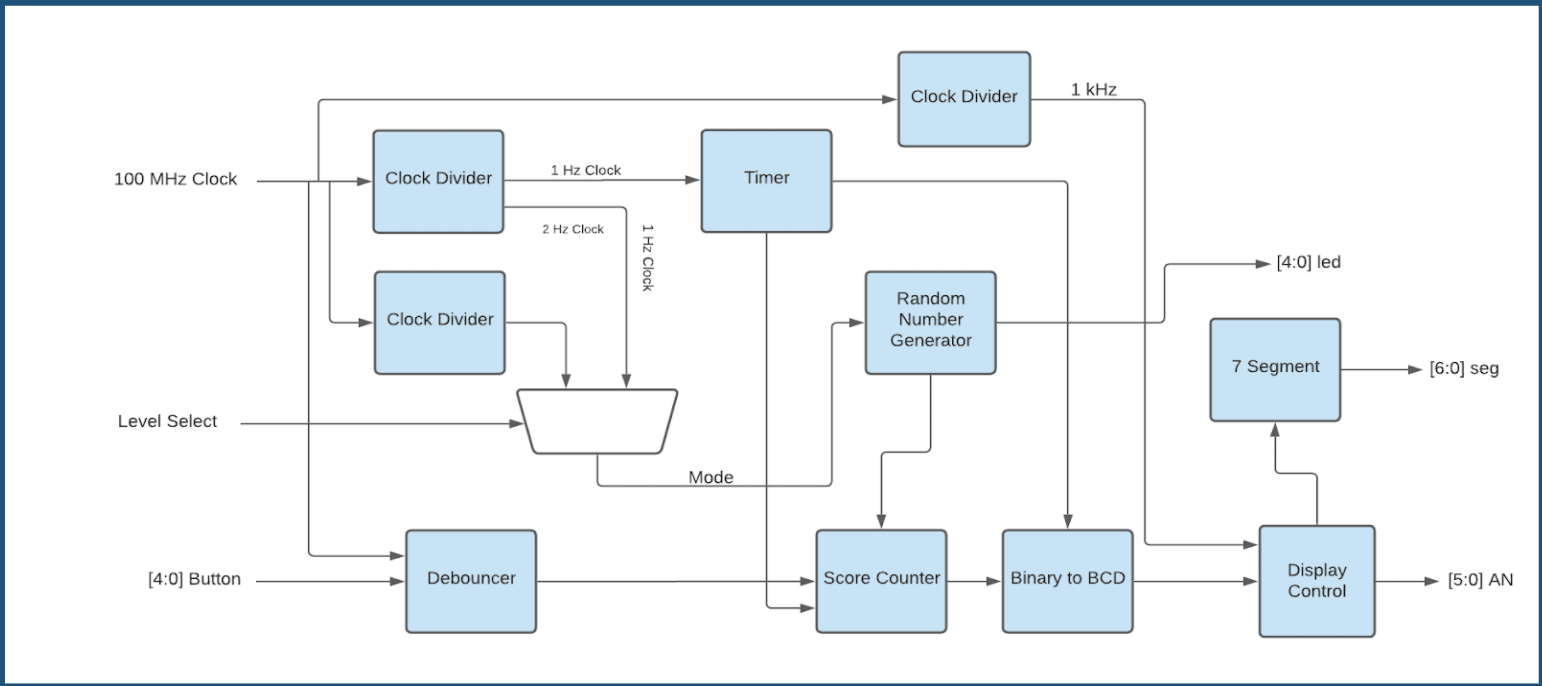
By Stacia Kolodziejski, Jack Crowley, Radhey Patel, Rajiv Ramroop

**Objective**

The objective of the project was to design the whack-a-mole game on the Nexys A7 FPGA board. The moles in this game were all represented by a series of 5 randomly blinking LEDs. If an LED was on, then that was the “mole” to whack. The whacking was done by the corresponding push button to the lit LED. We used the 7-segment display to display the score of the player and the timer that kept track of the game time. We also used a switch for the user to choose which level they wanted to play before the game started. There were two levels of our game; the easier level was for the LEDs to light up every one second and the harder level of the game was the LEDs lighting up every half second.

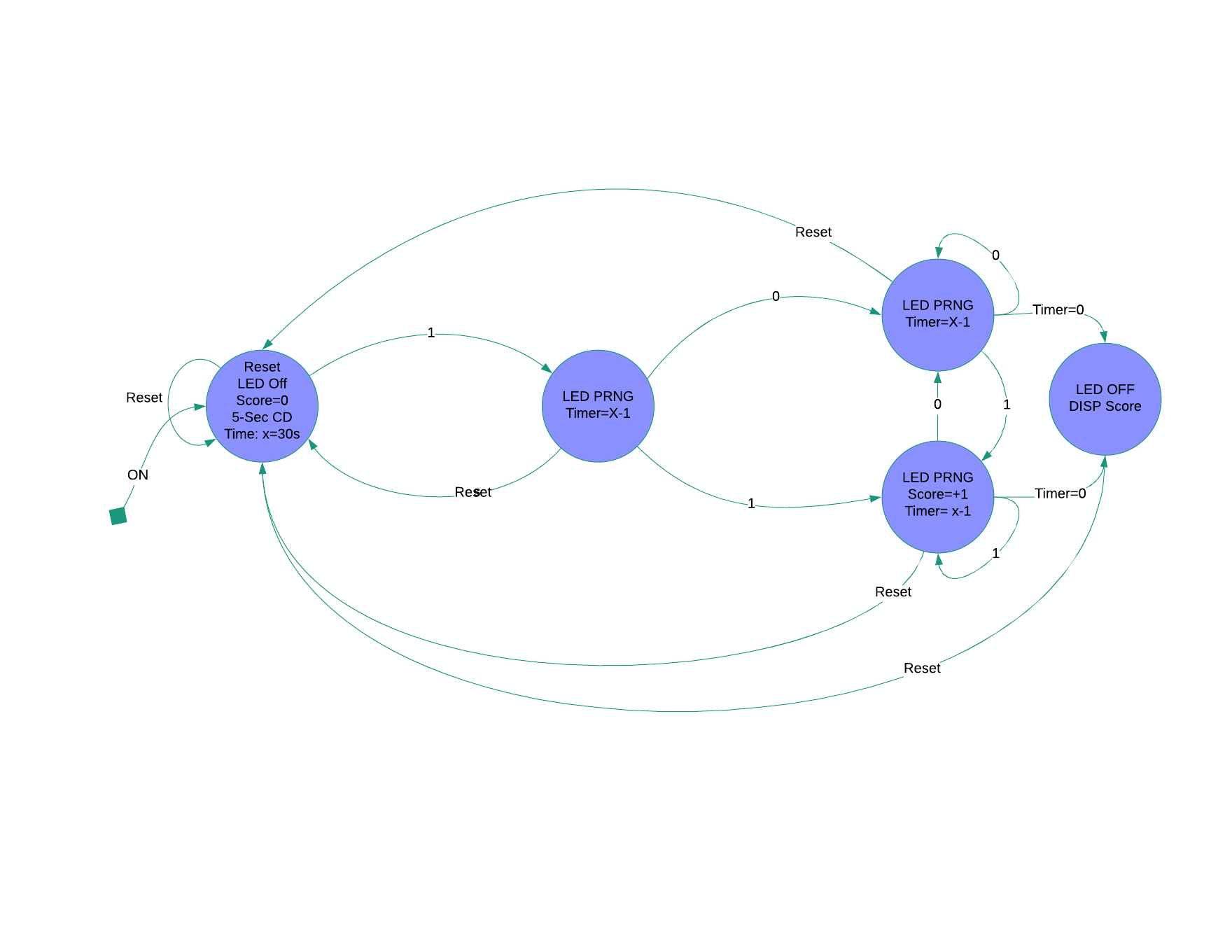
**Design Choices/System Functionality**

We wanted to create a fun, yet difficult game where the player can whack a mole with the press of a button. To achieve this, we chose very specific modules that work together. The purpose of each module is described below. A block diagram is provided to illustrate the specific modules that were used in the design of this project.



***Figure 1 - Block Diagram***

*This block diagram includes all of the modules used in our design. It is an overview of the top module which instantiates all the other modules. A global reset was included in each module, but for the purpose of not complicating the block diagram, it was not included.*

****

***Figure 2 - State Diagram***

*This State Diagram includes all of the states used in the design.*

Top Module

We used a top module for instantiating all of our other modules. The game features a 5-second timer that counts down to 0. During this time, the user may flip a switch to select a different game mode that is harder. Once the timer reaches 0, another timer is turned on that counts down from 30. This timer is the total game time. While the countdown timer is not equal to 0, the user may increment their score by selecting the correct push button which corresponds to a mole. The game ends once this timer is equal to 0, and the user's score will remain on the board until the reset button is clicked, which will start this sequence over again.

Multiplexer

We used a 2:1 multiplexer to determine what mode the game was in. Upon the flip of a switch, the game mode would go from easy to hard, and vice versa. A one second clock is used for the easy level of the game where the LEDs will light up every second whether the correct button is hit or not. The hard level of the game is used with a two second clock to make the LEDs light up faster.

Timer

We used this module to count down from both 5 and then 30. The starting sequence of counting down from 5 occurs first, and then the 30 second countdown begins. Once the 30 second countdown reaches 0, the game is over and the final score is displayed. The game does not start over unless the reset button is pressed. After 30 seconds, the LEDs stop blinking randomly.

Clock Dividers

For each level of the game and the timer, we needed to create clock dividers that would use the global 100 MHz clock as the input and then output a 1Hz, a 2Hz clock, and a 1kHz clock. The 1 Hz clock was used for the 5 and 30 second countdowns of the game and the easier level of the game. The 2 Hz clock we used for the harder level of the game. Here, the LEDs would light up every half second instead of every second and the user would have to push the correct button in the given time. The 1kHz clock was used in the display control to time the switching of the display. We used three different modules for these clock speeds that we instantiated in the top module.

Display Control

The display control module allows all 8 of the displays to be lit up simultaneously. It also controls where the timer is displayed, and where the score is displayed. To do this, the module uses an internal counter and a case statement for the anodes and which number (tens, or ones) to be used.

7 Segment Display

The 7-segment display was used to display the first 5 second countdown, the 30 second countdown, and the score of the game. The score was displayed on the right side of the display, while the countdowns were displayed on the left side of the display.

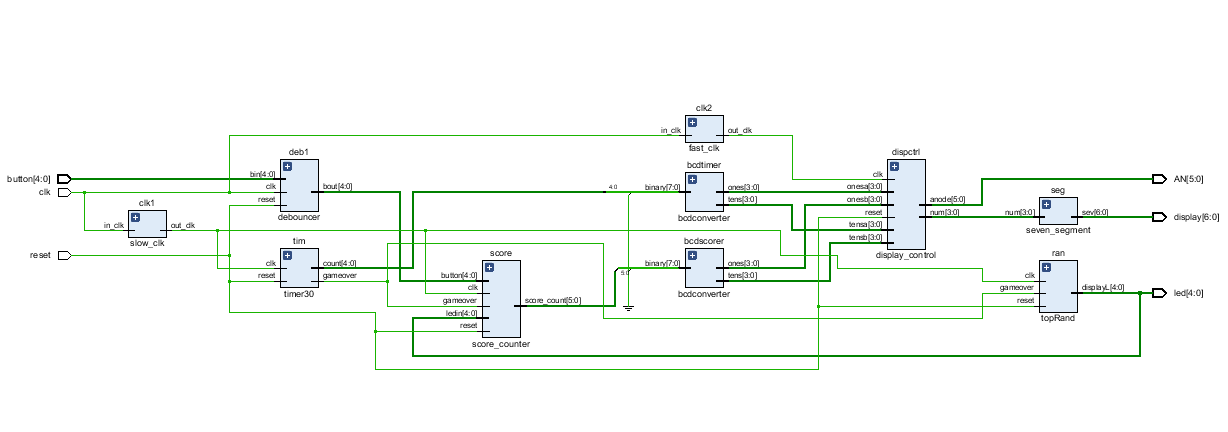
**Game Logic**

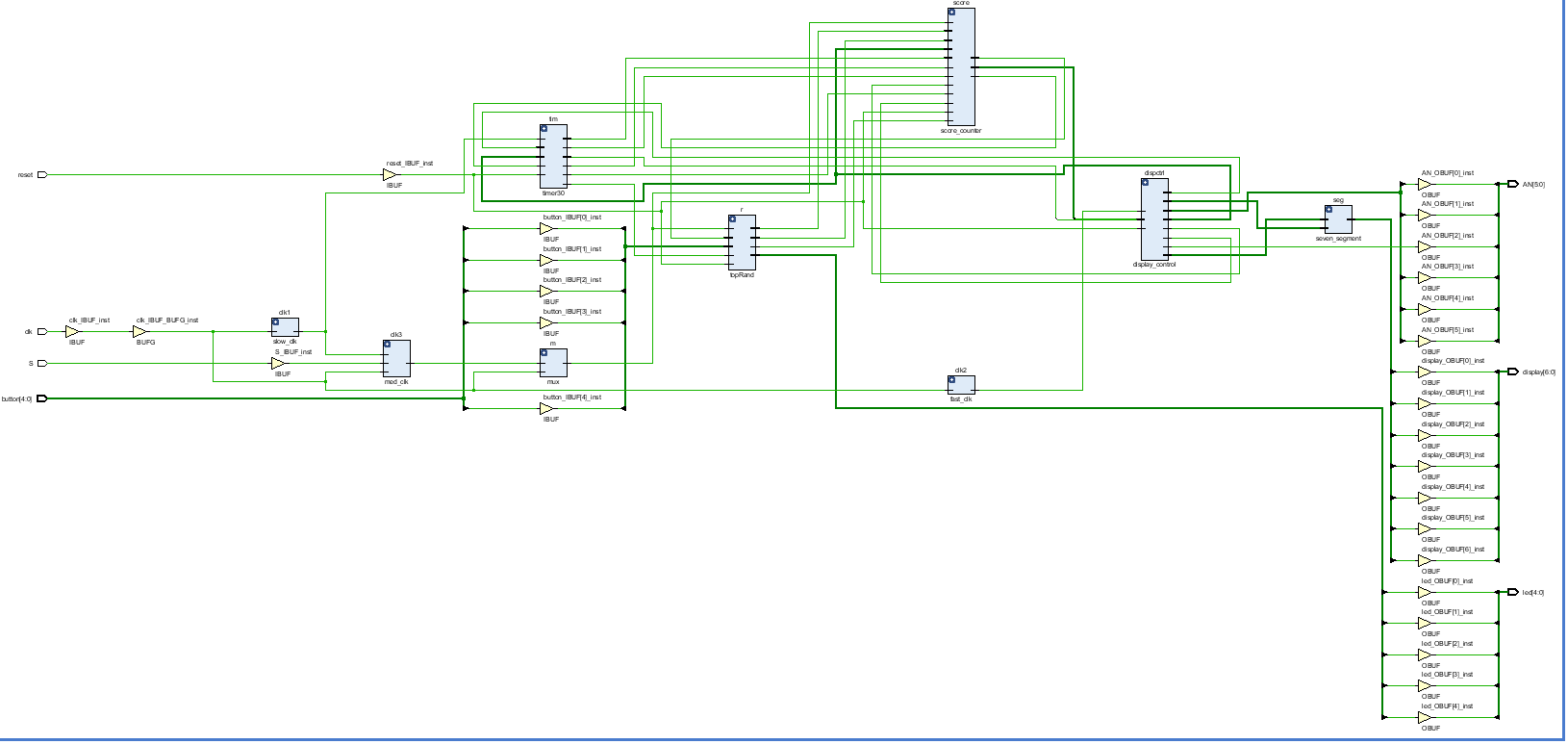
Random Number Generator

The LEDs that light up corresponding to a designated push-button needed to be randomized for us to recreate the real game of whack-a-mole. For this, we needed to use a random number generator module that would output a random number and then be assigned to an LED. This would ensure that the chosen LED each second (or half a second) would be randomized. The random number generator was not a random number generator, because we used a Linear Feedback Shift Register (LFSR). An LFSR is a way to generate pseudo random data inside of an FPGA. It is pseudo random, because given a state of the LFSR pattern, you can predict the next state. In other words, the sequence of the LED is not random. As such, a major limitation of our design is that anyone who plays this game repeatedly will be able to perfectly predict what buttons to press.

Score Counter

The score counter module was used to keep track of the score. Each time the correct button was pressed corresponding to the wired LED and within the given time, the score would increment. The score was displayed on the 7 segment display through the duration of the game, incrementing as the user hits the correct button, and once the game ends after 30 seconds, the score goes back to zero.





***Figure 3 and 4 - Schematics***

*The schematics generated show exactly which inputs get fed into other modules, and which output gets generated by each module. The inputs to the entire system include a 5-bit button to represent the 5 push buttons used. Clk is the global 100MHz clock that gets divided into several clock dividers. Reset represents the global reset. The outputs are AN[5:0] which represents the anodes that control what displays are lit up. Display [6:0] represents the 7 segment display and led[4:0] represents the 5 LEDs that are pseudo-randomly lit up.*

**Conclusion**

During this project process, we learned that testing each module as we made it was the best way to approach this project. In addition, we learned that having as many modules as possible was better than having only a few with more code. Having specific modules made it easier to debug our code once we put the top module together and implemented it on the FPGA board. We also learned the importance of having testbenches for the modules. When we first tried our code on the FPGA, we had many issues that we could not completely figure out and we had not made testbenches for most of the modules we created. However, once we made testbenches for each module it was very easy for us to see which modules worked well and which modules were not working. In doing so, we were able to successfully debug our code and implement our game correctly on the FPGA.