**Final Report, Fall 2016**



**College of Engineering and Computing**

**Department of Electrical and Computer Engineering**

**ECE 287: Digital Design Systems**

**Instructor: Dr. Jamieson**

**Project:** Baseball

**Student Name:** Alex Tutkovics and Jacob Brumfield

**Date:** December 16, 2016

**BACKGROUND KNOWLEDGE OF DESIGN**

For the digital design systems project, we implemented a baseball game on the DE2-115 board. With even the slightest knowledge of the sport, it is simple to follow along and play on the board. However to start, our game consists of home runs and outs, also known as home run derby. The game is designed as a multiplayer game. One person acts as the pitcher and the other player as a batter. The game is simple, one player controls the speed of the pitch and the other one attempts to hit the ball when it is located over the plate. There is then a basic “home run” and “out” scoreboard that will reach up to 10. The objective is to be able to score 10 home runs in the fewest amount of attempts.

**HIGH-LEVEL DESCRIPTION**

The game we implemented is all based of in verilog code on quartus II. The code we designed is made up of multiple functions in order to work properly. First, we needed to control a scoreboard that collects input and reads it as an output through the seven segment display. In order to keep the scores of each player, we needed multiple registers that can store up to 4 bits (counting up to 10). Those registers are connected to full adders which control the bit counting. Given the proper input, we then add one bit toward the home run register or the out register.

Another function to our game is the pitcher’s control and bat. For those we had to set up a finite state machine that changes depending on the inputs of the two players. The finite state machine consists 5 bits worth of states. Since we used a finite state machine, our schematic to our project consists of multiple multiplexers and flip flops. The two sequential blocks that we implemented into verilog consisted 2:1, 4:1, and even 8:1 multiplexers.

**DESCRIPTION**

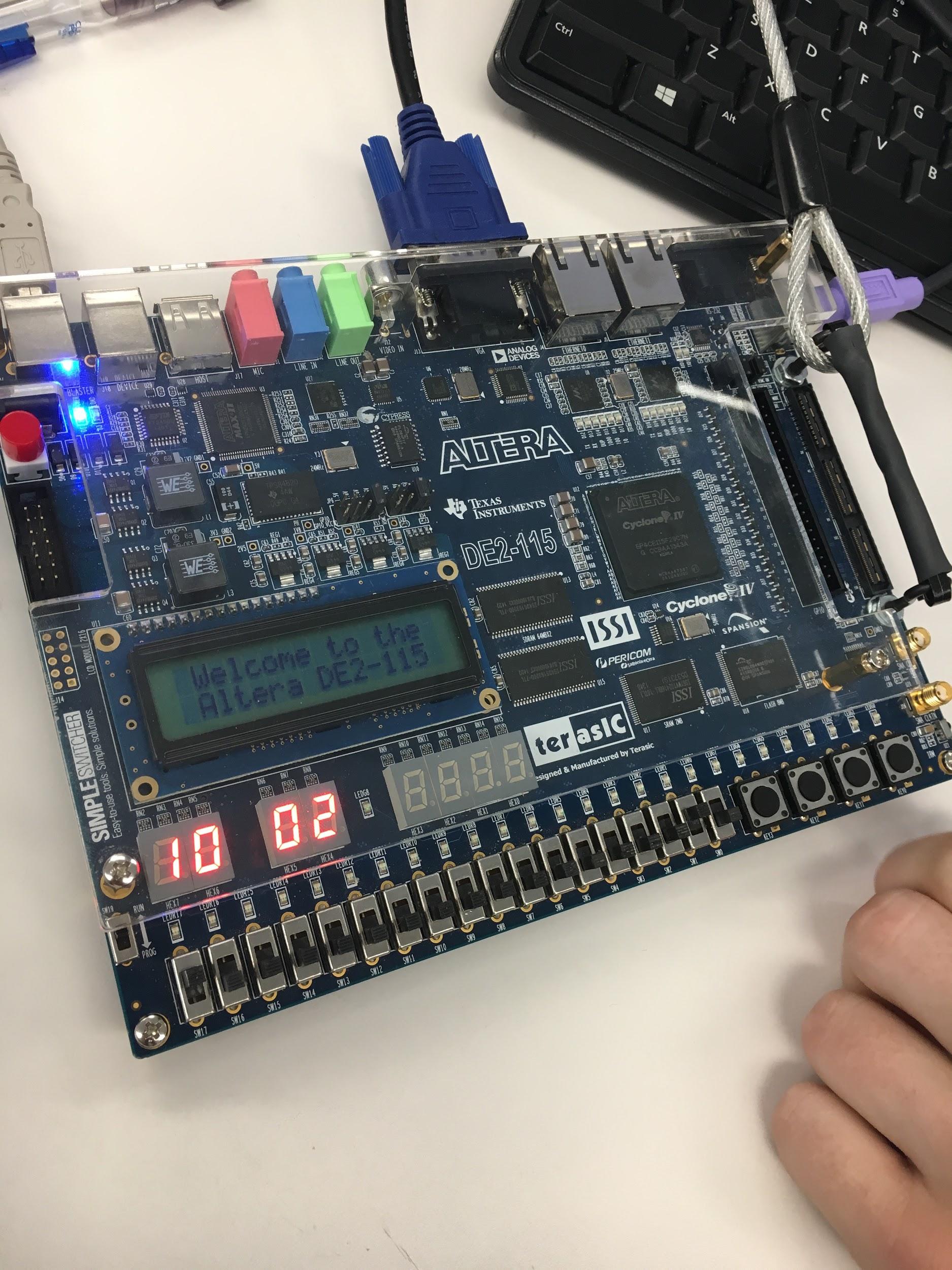
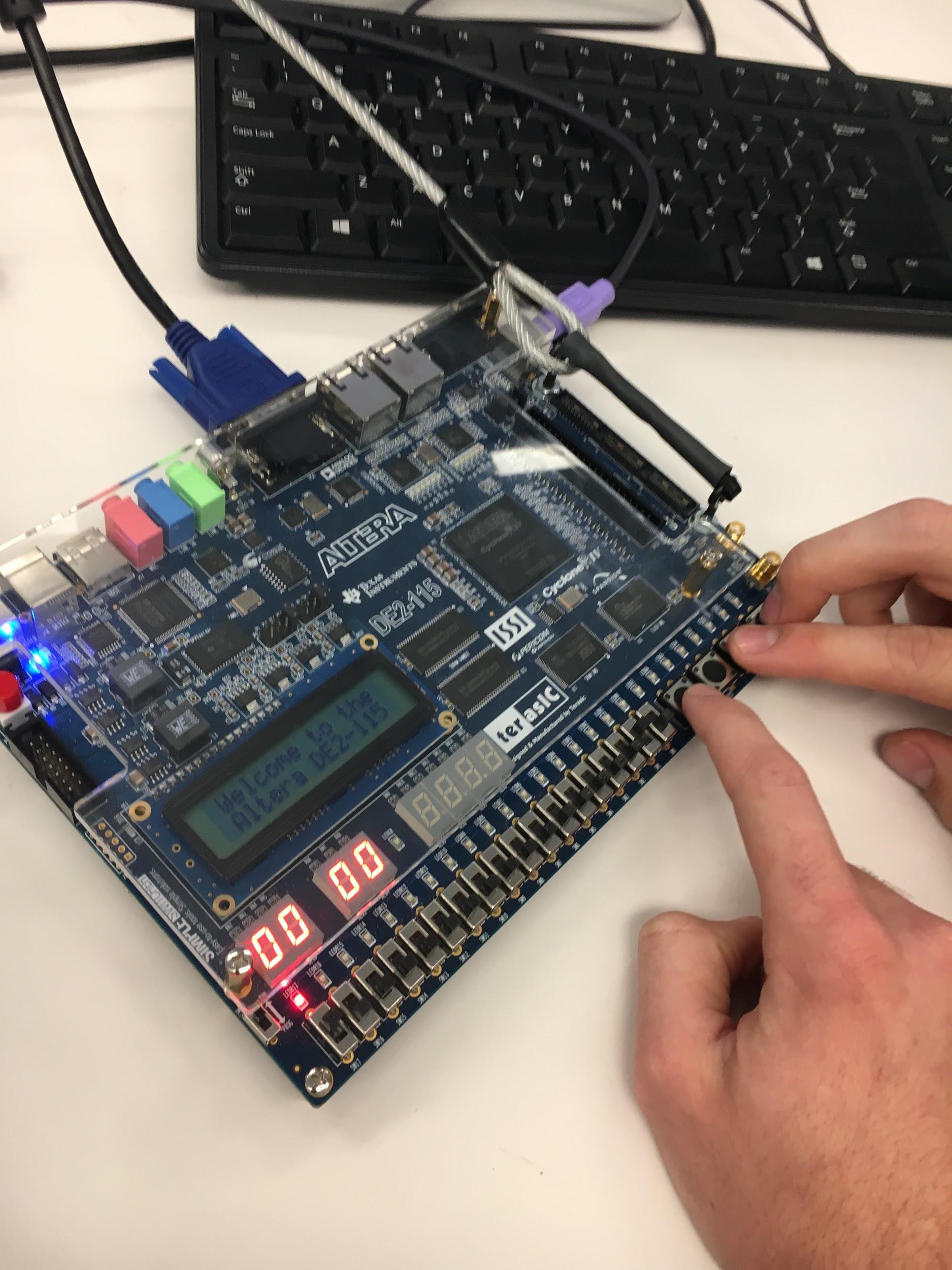
For anyone that would plan on building off on our project, we built this off of multiple components inside one large state machine function. The state machine function we implemented consists of two sequential blocks and one combinational block. There are also twenty case statements which controlled the location of the ball, ability to begin the pitch, scoreboard updates, and checking the batter’s input. To go further into depth, the first sequential block allowed the game to restart all of its components and begin a new game. This resets the score output, pitch, bat, and batter ready input. Then we coded a combinational block where we take the ball’s location step by step with an LED output. However, to even start the game the batter must send a signal to the pitcher saying they are ready. Once that happens, the pitcher will then press a key that allows them to control the speed of the ball. Once the ball reaches the last red LED, the batter must send an input signal. All in which is coded inside the last sequential always block. If the batter and ball line up correctly, the home run scoreboard will increase. However, if the batter is too early or misses it completely, the “out” scoreboard will increase by one. The majority of the states that we used are there to control the ball and the LED that represents the ball.

**RESULTS**

A few important notes to know about our project would be with the key pins located on the DE2-115 board. When the button is pressed down, it signals a 0 and a 1 if the button is released. Therefore, to time up the batter and the ball, the verilog must be corrected for the 0 state. A few things that could be implemented to make out game better would be to interface it with the ps2-keyboard or VGA monitor. The other thing that could make a more realistic approach to our game would be to synchronize a timer and have the LEDs move along a line without a constant need for a pitcher’s input.

**PLAYING GAME**

To play the game, there must be two players. The batter will use the key pin number as “key3” and the switch labeled as “sw17”. The switch the batter uses only indicates when the batter is ready for the pitch. The key button acts as the bat and must be pressed when the ball is over the plate, which is the first red LED (“r0”). The pitcher uses the key pins “key1” and “key0”. The “key0” acts as the ball movement. The “key1” pin acts as a faster pitch. However, to throw the fast pitch, the pitcher must hold down the “key1” pin and press the “key0” constantly till it goes over the plate.



**CONCLUSION**

In conclusion, the baseball game we created was over 1000 lines of verilog with over 20 states. The game also consisted of 46 pin assignments that all play a vital part of the game. The baseball game on the FPGA is a two-person game that need inputs consistently. We used up most of the FPGA in pin assignments, however, the logic and schematic is not very complicated in our design. This game is more advanced than lab 8, but we still lacked the ability to get a challenging chip to work on it.

Youtube link: <https://www.youtube.com/watch?v=Hm-g3lJc2g8>