**“The Pattern Game!”**

Instructions on how to build your very own 16 bit guessing game.

*By Brett Glidden and Collin Kenner*

**Step 1: “Introduction”**

In this tutorial, we will be building a game that generates a pattern that the player attempts to guess before time runs out. The game utilizes several pieces of hardware to create an experience for the player where they must attempt to keep their cool when every mistake they make cuts their remaining time in half. The device also contains a method for pseudo randomly generating the number that the user is going to guess. During gameplay, a clock and speaker enhance the experience by reminding the player just how fast time is flying by. If the player guess the pattern correctly before time runs out, the game shows the user the word “success” letting them know they finished the game before time ran out, but if the user is unable to find the pattern in time. The device reminds them with the word fail that they are hopeless. After this reminder, the user can press any button to return themselves to the welcome screen of the game. The game also features a hard reset button for the shameless losers who can’t face their inevitable defeat. This example is the simplest version of the game, but the game can be augmented in several different ways depending on the builders wishes. The speaker can be set to program sounds for all different phases of the game, or the game can be set to record best time for the winners. It can also be set to work with a 24 bit input if the user wishes for a more challenging experience.

**Step 1: Materials**

1 Basys 3 Board

1 speaker

Xilinx Vivado

**Step 2: Mapping the design**

The main functionality of the device is controlled within the state driver. This contains all 4 stages of the games. The intro state, the difficulty selection state, and the end state. The State Controller is a FSM which outputs the current value of state as a 4 bit binary bus. The functionality of the device can be broken down into several segments.

The button segments are always active, and they allow the user to select game difficulty, enter the difficulty selection mode, and reset the device when necessary. The button segments take in an input and outputs to other modules.

The difficulty Driver is a state machine with three states: easy, medium, and hard. This module has two functions. The first is a driver for the seven segment display to display the words for the actual difficulty. The second function is to output a 16 bit bus to the game module.

The final modules are all related specifically to the seven segment display. The end, init, and game drivers all map to specifically to the seven segment display. The outputs are send through the SSEG select module, and depending on the state being sent from the state machine the module sends the four values to the SSEGDriver.