Readme

Ignore all the files not in Final Project, that is my older files from earlier in the semester. Will remove for final version.

Final Module is the top module for the project.

**FinalProject**

**Input Ouput**

There are 4 inputs and 3 outputs. Switches are the 16 switches on the basys, check is the button to restart the FSM, reset is the button that resets the random number generator and restart the process. Clk is the clock that everything follows. Ssegs is the 7 seg display bits, an is the an for the seven seg. LEDS are the 15 led’s above the switches.

**Logic Variables**

Power 0 – Power 15 choose which FSM to turn on, all of these should be 0 except one.

Cc0 – cc15 These are the 15 outputs from the FSM.

**Modules**

Counter was disables in my latest round of testing as I was just testing FSM\_Case\_0. This should be constantly adding one to the previous value each time the clock rests, it is a 4 bit number so it should reset back to 0 after it counts to 15. I tested this in the simulation and it seems to work as intended.

CLK\_div2 is the clock divider, this might not be needed.

randNumTable has the inputs randNum and reset and it outputs randomNumberDecimal. This takes the 4 bit random generated number and outputs the corresponding 16 bit number when reset is pushed. This is only for passing to the hex seven segment display.

Big\_FSM, is a Mux that that the randNum and powers on a single FSM depending on what the random number generators.

FSM\_Case\_0 – FSM\_Case\_15 are the 15 different FSM cases, when it gets the power siginal the FSM will start and check to see if the switches match the hex number displayed. If the switches match then cc is 1 and the 7seg displays COOL, if it is wrong then it outputs 0 and Crap is displayed. If the switch is in the correct position then the LED above it is also lit up. I have only tested for FSM\_Case\_0, case1 – case15 are just copy and pasted from case 0. **TODO,** Change the bits in case 1 -case15 to match whatever 4digit hex numbers that you came up with. Please add the numbers to the table in the other word doc in this directory.

CC\_MUX, takes all of the powers as inputs, and all of the cc’s as inputs. It output’s whichever CC matches the power that is on.

BC\_DEC, is the module that was taken from one of the labs, it just outputs Crap/ cool. There are two versions in the code. The one labeled FOR DEBUG ONLY bypass the two mux’s that are below and directly outputs to the 7 seg display.

hexSevSeg is a 7 seg display module that is supposed to take a 16-bit number and output it as a 4 digit hex number. I took this from a YouTube tutorial, and it does not seem to work. I have yet to troubleshoot it.

The two mux’s at the bottom choose whether to output the hexSevSeg or BC\_DEC depending if displayhex is pushed down then Hex sev seg is displayed, if DisplayCC is pushed down the DC\_DEC is displayed. **The mux’s are not working as intended.** Bypass the mux to see what should happen.