2 Pager - Whack A Mole

Major components and explanation of the logic behind them and what needs to happen with them.

**Random number generator** - We have chosen a linear-feedback shift register to randomize which moles will be visible at any one time. Linear feedback shift registers are simple to construct and do not require an ALU to generate a pseudorandom bit stream. We were also considering a linear congruential generator to generate a pseudorandom sequence, but the amount of logic required for it is much much more. A linear congruential generator implements the following formula: x(n+1) = [a\*x(b) + b] mod m. This kind of generator would need an ALU and a separate division logic unit to implement the modulo. This would use up a lot more space on our chip than the linear feedback shift register. The linear feedback shift register uses a sequence of registers that are fed into each other, except at least register outputs go into a number of xor gates, which is then fed back into the first register. This pseudorandom number generator is a deterministic and periodic finite state machine; the number of registers determines the number of states. We are going to use a 7-bit LFSR as we will be looking at 3 bits at a time from this LFSR. We are using 3 bits so we do not have to include any illegal state logic for our game board, which has 8 possible places the mole can be found. A 7 bit LFSR has a period for any seed of 127, which is more than the recommended amount of 64 for a board with 8 possibilities. This also means that this LFSR has 127 states for a given seed. A 7 bit LFSR needs 3 2-input XOR gates and 7 registers to cycle through.

**Decision logic -**

Button presses: Go to score counting routine and reset 555 timer.

Timer: Always active. Set for low duty cycle and period of 3 seconds.

Score counting: if <9, go up 1, else 0. Output to decoder (always). If person lets 3 seconds pass, then goes to 0. Will require multiple registers.

Mole placement: on 555 pulse, set board to state determined by our RNG.

Startup: reset everything and seed RNG.

**Inputs and outputs** - The LED’s will utilize 8 output pins and the button portion of the LED’s will comprise 8 input pins. The 555 timer will require an additional input pin, an output pin to trigger, and a further 4 output pins to connect to a decoder driving a 7 segment display.

**External components** - In order to fully utilize the chip, many of the IO’s will be used. We plan to utilize 8 tactile LED’s, corresponding to 16 of the IO ports. We plan to output to a decoder that controls a 7 segment display, in order to display the number of correct “whacks” in a row. We will as well need a 555 timer to drive the automatic changes after 3 seconds. Lastly, we will utilize a switch to turn the game on and off.

**Testing strategy:**

We will create a verilog testbench that simulates the functionality of the chip. It will need to be comprehensive in order to simulate all combinations of button presses vs lit LED’s vs score vs what is stored in the linear feedback shifter. Based on our calculations, this would correspond to 81920 test scenarios.

