CMPE 415 - Homework #3

Rescue POW

Start assignment as soon as possible. Work individually, but you can ask your classmates for help when you get stuck, with consideration to the course collaboration policy (please read it in the course website).

Before getting started, you should go through the Verilog notes located under Course Readings on the course home page.

Notes:

- Clearly state whether your design is fully functional, and state the failing sections if any exist.
- Make sure your design and code are easily readable and understandable (clear and well commented).

You are going to create a simple video game using the LEDs and push buttons to simulate saving prisoners of war (POWs).

The games will go as follows:

- To start the game, press RESET. One or two random LEDs should be lit indicating the positions of the POWs. However, there is also an XOR overlay of a blinking LED at position 0 representing your position.

- As you press the left button/the right button, your position, represented by the blinking cursor, moves.

- Move towards one of the POWs. Once you have reached them, press the center button. This rescues the POW and the corresponding the LED no longer stays lit after the cursor is moved.

- When all the POWs are saved (1 or 2 depending on the initial randomization), the game is over.

You must create some observable action to indicate the end of game, such as making all the LEDs blink on and off.

Your Part

- To store the POW position(s), create an 8-bit flip flop module.
  - The declaration should be as follows: pow_position (pos,clk,rst ,init,d);
    - pos is 8 bit output
    - init and d are 8 bit inputs
  - One every clock, load _init if rst is pressed and load d otherwise
init should be provided from random2
- YOU MUST USE THE ALWAYS BLOCK WITH NO ASSIGN STATEMENTS OR INSTANTIATIONS

- To store your position, create an 8-bit shift register module.
  - The declaration should be as follows: position_shifter(pos, clk, rst, sr, sl);
  - On every clock, it should initialize to POS[7:0]=8'b00000001 when rst is high and otherwise it should right shift when shift_right is high and left when shift_left is high
  - YOU MUST USE THE ALWAYS BLOCK WITH NO ASSIGN STATEMENTS OR INSTANTIATIONS

- To create the LED display:
  - AND each of the 8 bits from the position register with the output of the pulser to create a blinking indicator. Then XOR this result with the output of the POW position flipflop.
  - The pulser will have a duty cycle less than 50% so that the cursor is shown as short blinks of light when not over a POW and the inverse when over a POW.
  - YOU CAN USE WHATEVER YOU LIKE TO ACHIEVE THIS

- You need to figure out what to provide to the d input of the pow position flip flop
- You need to figure out to create the effect of clearing some bits if the cursor position is the same as a POW position
- Use a NOR to figure out when the game is over
- Remember to create some indication that the game is over

Turn in

- Full design compile and verification in Artix board hardware
- Create and submit one or multiple Verilog testbench modules that tests your submodule designs.
- Create a report that explains your design and your testing in software and hardware
- Include the output of your Verilog test bench(s).
- Include screen shot of your working design in hardware and possibly a video link for your design (optional) to help the grader.