Miniproject 3

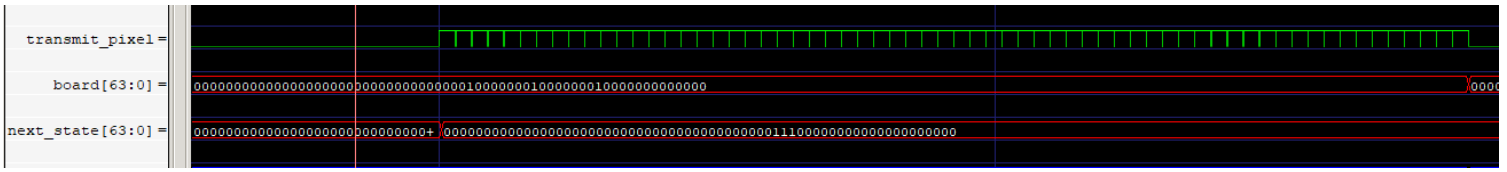
Ramzey Burdette

I did the 3 Games of Life in different colors at once, in different colors.

Starting with how the Game of Life functions in a digital circuit, it is very similar to previous code where you load up results and push it to the shift register LED matrix, but instead of just doing that, you have to take the state that you get from your memory files and update it when you have to transmit the pixels to the board one by one.

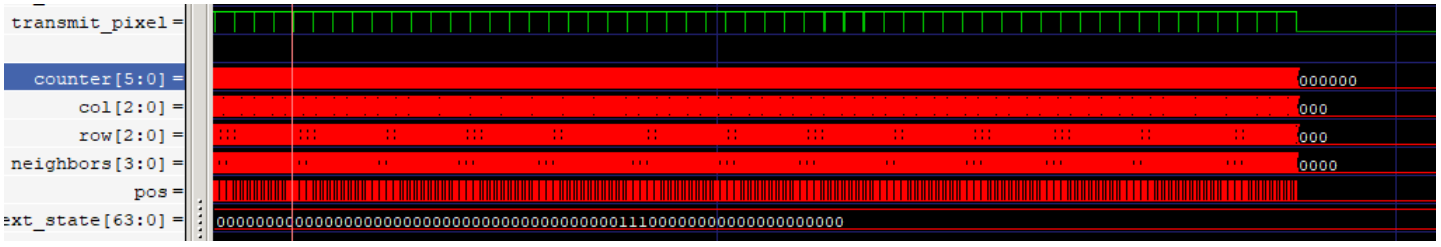
The only main new logic is just the updating of the next state. In my logic, I have the memory which is 64 different 8 bit values, but I also have a board state which is just 1 64-bit integer. One of my goals for this project was to make the entire board work within a 64 bit integer, which was much easier to do in SystemVerilog due to the way you can index into large logic values very easily.

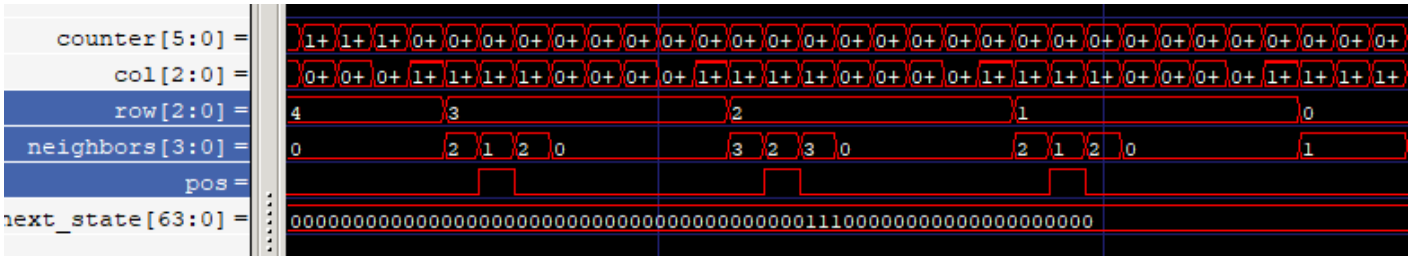
When the controller tells everything to start transmitting pixels to the board, the Game of Life quickly calculates all of the positions for the next state.



You can see that quickly, once transmit_pixel starts, it calculates the next state.
Once the transmit_pixel stage is over is when it pushes the next_state to the board, effectively updating the memory and therefore what is displayed on the board.

To get a better view of the Game of Life running with the red color, you can see that it will rapidly go through all of the cells and their neighbors.





Now doing this combinationally would have been an even better optimization, but to see it increment, doing this sequentially works well.

To run this, I just have 3 of each memory and game module running at the same time, and 1 controller and 1 push to the LED matrix, since they are all changing on the same timeline.