```
module update_frog (clk, new_game, reset, gameover, player_input, frog_x, frog_y);
   input  logic clk, new_game, reset, gameover;
   input  logic [3:0] player_input;
   output logic [3:0] frog_x, frog_y; // 4 bits bc we need range of 0-16 for each coord


   always_ff @(posedge clk) begin
      if (reset || new_game) begin
         frog_y <= 8;
         frog_x <= 0;
      end
      else if (gameover == 0) begin
         if (player_input[3]) // move up
            if (frog_y < 15)
               frog_y <= frog_y + 1;
         if (player_input[2]) // move down
            if (frog_y > 0)
               frog_y <= frog_y - 1;
         if (player_input[1]) // move left
            if (frog_x > 0)
               frog_x <= frog_x - 1;
         if (player_input[0]) // move right
            if (frog_x < 15)
               frog_x <= frog_x + 1;
      end
      if (frog_x == 15)
         frog_x <= 0;


   end

endmodule

module update_frog_testbench ();
   logic clk, new_game, reset, gameover;
   logic [3:0] player_input;
   logic [3:0] frog_x, frog_y;

   update_frog dut(clk, new_game, reset, gameover, player_input, frog_x, frog_y);

   // Set up the clock.
   parameter CLOCK_PERIOD=100;
   initial begin
   clk <= 0;
   forever #(CLOCK_PERIOD/2) clk <= ~clk;
   end

   initial begin
                                    @(posedge clk);
      reset <= 1;                   @(posedge clk);
      reset <= 0;                   @(posedge clk);
      player_input <= 0;            @(posedge clk);
      gameover <= 0;                @(posedge clk);
      // pass through some basic player inputs, during game and
gameover
                                    @(posedge clk);
                                    @(posedge clk);
      player_input[0] <= 1;         @(posedge clk);
      player_input[0] <= 0;         @(posedge clk);
      player_input[0] <= 1;         @(posedge clk);
      player_input[0] <= 0;         @(posedge clk);
                                    @(posedge clk);
                                    @(posedge clk);
      player_input[1] <= 1;         @(posedge clk);
      player_input[1] <= 0;         @(posedge clk);
      player_input[1] <= 1;         @(posedge clk);
      player_input[1] <= 0;         @(posedge clk);
                                    @(posedge clk);
                                    @(posedge clk);
      player_input[2] <= 1;         @(posedge clk);
      player_input[2] <= 0;         @(posedge clk);
      player_input[2] <= 1;         @(posedge clk);
      player_input[2] <= 0;         @(posedge clk);
                                    @(posedge clk);
                                    @(posedge clk);
      player_input[3] <= 1;         @(posedge clk);
```

```
76          player_input[3] <= 0;          @(posedge clk);
77          player_input[3] <= 1;          @(posedge clk);
78          player_input[3] <= 0;          @(posedge clk);
79                                         @(posedge clk);
80                                         @(posedge clk);
81          gameover <= 1;                 @(posedge clk);
82                                         @(posedge clk);
83                                         @(posedge clk);
84          player_input[0] <= 1;          @(posedge clk);
85          player_input[0] <= 0;          @(posedge clk);
86          player_input[0] <= 1;          @(posedge clk);
87          player_input[0] <= 0;          @(posedge clk);
88                                         @(posedge clk);
89                                         @(posedge clk);
90          player_input[1] <= 1;          @(posedge clk);
91          player_input[1] <= 0;          @(posedge clk);
92          player_input[1] <= 1;          @(posedge clk);
93          player_input[1] <= 0;          @(posedge clk);
94                                         @(posedge clk);
95                                         @(posedge clk);
96          player_input[2] <= 1;          @(posedge clk);
97          player_input[2] <= 0;          @(posedge clk);
98          player_input[2] <= 1;          @(posedge clk);
99          player_input[2] <= 0;          @(posedge clk);
100                                        @(posedge clk);
101                                        @(posedge clk);
102         player_input[3] <= 1;          @(posedge clk);
103         player_input[3] <= 0;          @(posedge clk);
104         player_input[3] <= 1;          @(posedge clk);
105         player_input[3] <= 0;          @(posedge clk);
106                                        @(posedge clk);
107                                        @(posedge clk);
108                                        @(posedge clk);
109                                        @(posedge clk);
110
111         $stop; // End the simulation.
112      end
113
114   endmodule
115
```