```
1   // Eugene Ngo
2   // 11/18/2022
3   // EE 271
4   // Lab 4
5
6   // victory takes in clk, reset, LED9, LED1, L, and R as 1-Bit inputs and returns 7-Bit
    winner.
7   // This module determines who the winner of the Tug of War is by checking to see if the LED
    is at the
8   // far right or far left, if it is left then Player 1 (P1) wins and such is displayed and
    if it is right
9   // then Player 2 (P2) wins and that is displayed.
10
11   module victory (winner, clk, reset, LED9, LED1, L, R);
12     input logic clk, reset;
13     input logic LED9, LED1, L, R;
14     output logic [6:0] winner;
15
16     enum {off, P1, P2} ps, ns;
17
18     always_comb begin
19        case(ps)
20           off:   if(LED9 & L & ~R) ns = P1;
21
22                  else if(LED1 & R & ~L) ns = P2;
23
24                  else ns = off;
25
26           P1: ns = P1;
27
28           P2: ns = P2;
29
30        endcase
31
32        if(ns == P1) winner = 7'b1111001;
33        else if (ns == P2) winner = 7'b0100100;
34        else winner = 7'b1111111;
35
36     end
37
38     always_ff @(posedge clk) begin
39        if(reset)
40           ps <= off;
41        else
42           ps <= ns;
43     end
44
45  endmodule
46
47  // victory_testbench tests all expected, unexpected, and edgecase behaviors
48  module victory_testbench();
49     logic clk, reset;
50     logic LED9, LED1, L, R;
51     logic [6:0] winner;
52
53     victory dut (.winner, .clk, .reset, .LED9, .LED1, .L, .R);
54
55     parameter CLOCK_PERIOD = 100;
56     initial begin
57        clk <= 0;
58        forever #(CLOCK_PERIOD / 2)
59        clk <= ~clk;
60     end
61
62     initial begin
63        reset <= 1;                                  @(posedge clk);
64                                                     @(posedge clk);
65        reset <= 0;                                  @(posedge clk);
66                                                     @(posedge clk);
67        LED9 <= 1; LED1 <= 0; L <= 1; R <= 0;        @(posedge clk);
68                                                     @(posedge clk);
69        LED9 <= 0; LED1 <= 1;                        @(posedge clk);
70                                                     @(posedge clk);
```

```
71          LED9 <= 1; LED1 <= 1;                   @(posedge clk);
72                                                  @(posedge clk);
73                     LED1 <= 0;          R <= 1;  @(posedge clk);
74                                                  @(posedge clk);
75          reset <= 1;                             @(posedge clk);
76                                                  @(posedge clk);
77          reset <= 0;                             @(posedge clk);
78                                                  @(posedge clk);
79          LED9 <= 0; LED1 <= 1; L <= 0; R <= 1;   @(posedge clk);
80                                                  @(posedge clk);
81          LED9 <= 1;                              @(posedge clk);
82                                                  @(posedge clk);
83          LED9 <= 0;             L <= 1;          @(posedge clk);
84                                                  @(posedge clk);
85          $stop;
86       end
87    endmodule
```