

```
1  /*
2   Ben Davis
3   3/7/24
4   EE 371
5   Lab 6, Task 2
6
7   This module takes in the car prescence inputs and
8   outputs a 7 bit value for the HEX displays to show
9   the available spots.
10  */
11
12  module lot_spots (
13      input logic [1:0] cars, //amount of cars in lots
14      output logic [6:0] hx0, //hex0 value
15      output logic [6:0] hx1, //hex1 value
16      output logic [6:0] hx2, //hex2 value
17      output logic [6:0] hx3 //hex3 value
18  );
19
20      //uses an always comb value to have the hexes
21      //show the available amount of spots left
22      //goes from 3-1, and displays "FULL" if
23      //0 spots are left
24      always_comb begin
25          case(cars)
26              2'b00: begin
27                  hx0 <= 7'b0110000;
28                  hx1 <= 7'b1111111;
29                  hx2 <= 7'b1111111;
30                  hx3 <= 7'b1111111;
31              end
32              2'b01: begin
33                  hx0 <= 7'b0100100;
34                  hx1 <= 7'b1111111;
35                  hx2 <= 7'b1111111;
36                  hx3 <= 7'b1111111;
37              end
38              2'b10: begin
39                  hx0 <= 7'b1111001;
40                  hx1 <= 7'b1111111;
41                  hx2 <= 7'b1111111;
42                  hx3 <= 7'b1111111;
43              end
44              2'b11: begin
45                  hx0 <= 7'b1000111;
46                  hx1 <= 7'b1000111;
47                  hx2 <= 7'b1000001;
48                  hx3 <= 7'b0001110;
49              end
50          endcase
51      end
52  endmodule
53  //testbench
54  module lot_spots_tb();
55      //logic variables
56      logic [1:0] cars;
57      logic [6:0] hx0, hx1, hx2, hx3;
58
59      lot_spots dut(.);
60
61      //tests a case where the lot fills up one car at a time
62      initial begin
63          cars <= 2'b00; #5;
64          cars <= 2'b01; #5;
65          cars <= 2'b10; #5;
66          cars <= 2'b11; #5;
67          $stop;
68      end
69  endmodule
```