```
/* Name: Eugene Ngo
   Date: 1/13/2023
   Class: EE 371
   Lab 1: Parking Lot Occupancy Counter*/

// carSensor takes inputs from two sensors, a and b, and output "1" to either enter or exit for 1 clock cycle
// whenever an entering or exiting vehicle is detected.
module carSensor (a, b, enter, exit, clk, reset);
   input logic a, b, clk, reset;
   output logic enter; // car entering
   output logic exit; //  car exiting

   enum {none, entering01, exiting01, entering11, exiting11, entering10, exiting10, idle} ps, ns;

   // Logic for next state
   always_comb begin
      case(ps)
         none: if (~a & ~b) ns = none;
               else if (~a & b) ns = entering01;
               else if (a & ~b) ns = exiting10;
               else ns = idle;
         entering01: if (~a & ~b) ns = none;
               else if (~a & b) ns = entering01;
               else if (a & ~b) ns = idle;
               else ns = entering11;
         exiting01: if (~a & ~b) ns = none;
               else if (~a & b) ns = exiting01;
               else if (a & ~b) ns = idle;
               else ns = exiting11;
         entering11: if (~a & ~b) ns = none;
               else if (~a & b) ns = entering01;
               else if (a & ~b) ns = entering10;
               else ns = entering11;
         exiting11: if (~a & ~b) ns = none;
               else if (~a & b) ns = exiting01;
               else if (a & ~b) ns = exiting10;
               else ns = exiting11;
         entering10: if (~a & ~b) ns = none;
               else if (~a & b) ns = idle;
               else if (a & ~b) ns = entering10;
               else ns = entering11;
         exiting10: if (~a & ~b) ns = none;
               else if (~a & b) ns = idle;
               else if (a & ~b) ns = exiting10;
               else ns = exiting11;
         idle: if (~a & ~b) ns = none;
               else ns = idle;
      endcase
   end // always_comb

   //output logic for exiting: outputs 1 to exit when an exiting vehicle is detected.
   always_comb begin
      case(ps)
         exiting01: if (~a & ~b) exit = 1'b1;
                     else exit = 1'b0;
         default: exit = 1'b0;
      endcase
   end // always_comb

   //DFFs
   always_ff @(posedge clk) begin
      if (reset)
         ps <= none;
      else
         ps <= ns;
   end // always_ff
endmodule // carSensor

// carSensor_testbench tests all expected, unexpected, and edgecase behaviors
module carSensor_testbench();
   logic a, b, clk, reset, enter, exit;
   logic CLOCK_50;

   carSensor dut (.a(b), .b(a), .clk(CLOCK_50), .reset, .enter, .exit);

   // Setting up a clock.
   parameter CLOCK_PERIOD = 100;
   initial begin
      CLOCK_50 <= 0;
      forever #(CLOCK_PERIOD/2) CLOCK_50 <= ~CLOCK_50; // toggle the clock forever
   end // initial

   initial begin
      // reset
      reset <= 1;                      repeat(3) @(posedge CLOCK_50);

      //enters
      reset <= 0;      a <= 0; b <= 0; repeat(2) @(posedge CLOCK_50);
                       a <= 1; b <= 0; repeat(2) @(posedge CLOCK_50);
                       a <= 1; b <= 1; repeat(2) @(posedge CLOCK_50);
                       a <= 0; b <= 1; repeat(2) @(posedge CLOCK_50);
                       a <= 0; b <= 0; repeat(2) @(posedge CLOCK_50);
      //exits
                       a <= 0; b <= 0; repeat(2) @(posedge CLOCK_50);
                       a <= 0; b <= 1; repeat(2) @(posedge CLOCK_50);
                       a <= 1; b <= 1; repeat(2) @(posedge CLOCK_50);
                       a <= 1; b <= 0; repeat(2) @(posedge CLOCK_50);
                       a <= 0; b <= 0; repeat(2) @(posedge CLOCK_50);

      // direction changes while entering
                       a <= 0; b <= 0; repeat(2) @(posedge CLOCK_50);
                       a <= 1; b <= 0; repeat(2) @(posedge CLOCK_50);
                       a <= 1; b <= 1; repeat(2) @(posedge CLOCK_50);
                       a <= 1; b <= 0; repeat(2) @(posedge CLOCK_50);
                       a <= 1; b <= 1; repeat(2) @(posedge CLOCK_50);
                       a <= 0; b <= 1; repeat(2) @(posedge CLOCK_50);
                       a <= 1; b <= 1; repeat(2) @(posedge CLOCK_50);
                       a <= 0; b <= 1; repeat(2) @(posedge CLOCK_50);
                       a <= 0; b <= 0; repeat(2) @(posedge CLOCK_50);

      // direction changes while exiting
                       a <= 0; b <= 0; repeat(2) @(posedge CLOCK_50);
                       a <= 0; b <= 1; repeat(2) @(posedge CLOCK_50);
                       a <= 1; b <= 1; repeat(2) @(posedge CLOCK_50);
                       a <= 0; b <= 1; repeat(2) @(posedge CLOCK_50);
                       a <= 1; b <= 1; repeat(2) @(posedge CLOCK_50);
                       a <= 1; b <= 0; repeat(2) @(posedge CLOCK_50);
                       a <= 1; b <= 1; repeat(2) @(posedge CLOCK_50);
                       a <= 1; b <= 0; repeat(2) @(posedge CLOCK_50);
                       a <= 0; b <= 0; repeat(2) @(posedge CLOCK_50);
      $stop;
   end // intial

endmodule // carSensor_testbench

```