

```

1  module collision (clk, reset, RedPixels, frog_x, frog_y, gameover);
2
3      input logic [15:0][15:0] RedPixels;
4      input logic [3:0] frog_x;
5      input logic [3:0] frog_y;
6      input logic clk;
7      input logic reset;
8      output logic gameover;
9
10
11     always_ff @(posedge clk) begin
12         integer i;
13         integer j;
14
15         if (reset) begin
16             gameover <= 0;
17
18         end
19
20         else if ( RedPixels[frog_x][frog_y] == 1) begin
21             gameover <= 1;
22         end
23     end
24 endmodule
25
26 module collision_testbench ();
27     logic [15:0][15:0] RedPixels;
28     logic frog_x;
29     logic frog_y;
30     logic clk;
31     logic reset;
32     logic gameover;
33
34     collision dut(clk, reset, RedPixels, frog_x, frog_y, gameover);
35
36     // Set up the clock.
37     parameter CLOCK_PERIOD=100;
38     initial begin
39         clk <= 0;
40         forever #(CLOCK_PERIOD/2) clk <= ~clk;
41     end
42
43     initial begin
44
45         @(posedge clk);
46         reset <= 1;
47         reset <= 0;
48         @(posedge clk);
49         // have scenario of no collision -> collision
50         frog_x <= 0; frog_y <= 0;
51         RedPixels[1] <= 16'b1110000110011111;
52         frog_x <= 1; frog_y <= 1;
53
54         @(posedge clk);
55         @(posedge clk);
56         @(posedge clk);
57         @(posedge clk);
58         @(posedge clk);
59         @(posedge clk);
60         @(posedge clk);
61         @(posedge clk);
62         @(posedge clk);
63         @(posedge clk);
64         @(posedge clk);
65         @(posedge clk);
66         @(posedge clk);
67         @(posedge clk);
68         @(posedge clk);
69         @(posedge clk);
70         @(posedge clk);
71         @(posedge clk);
72
73         $stop; // End the simulation.
74     end
75 endmodule

```