

```
1  module UserInput(clk, reset, buttons, out);
2      input logic clk, reset;
3      input logic [1:0] buttons;
4      output logic [1:0] out;
5
6      logic [1:0] ps, ns;
7
8      always_comb begin
9          ns = buttons;
10     end
11
12     assign out = (buttons & ~ps);
13
14     always_ff @(posedge clk) begin
15         if (reset)
16             ps <= 2'b00;
17         else
18             ps <= ns;
19     end
20
21 endmodule
22
23 module UserInput_testbench();
24     logic clk, reset;
25     logic [1:0] buttons, out;
26
27     UserInput dut (clk, reset, buttons, out);
28
29     parameter CLOCK_PERIOD=100;
30     initial begin
31         clk <= 0;
32         forever #(CLOCK_PERIOD/2) clk <= ~clk;
33     end
34
35     initial begin
36         reset <= 1; buttons <= 2'b0; @ (posedge clk);
37         reset <= 0; @ (posedge clk);
38         buttons <= 2'b01; @ (posedge clk);
39         buttons <= 2'b00; @ (posedge clk);
40         buttons <= 2'b10; @ (posedge clk);
41         buttons <= 2'b00; @ (posedge clk);
42         buttons <= 2'b11; @ (posedge clk);
43         buttons <= 2'b00; @ (posedge clk);
44         buttons <= 2'b01; @ (posedge clk);
45         buttons <= 2'b10; @ (posedge clk);
46         buttons <= 2'b11; @ (posedge clk);
47         buttons <= 2'b00; @ (posedge clk);
48         buttons <= 2'b01; @ (posedge clk);
49         buttons <= 2'b10; @ (posedge clk);
50         buttons <= 2'b11; @ (posedge clk);
51         buttons <= 2'b00; @ (posedge clk);
52         buttons <= 2'b01; @ (posedge clk);
53         buttons <= 2'b10; @ (posedge clk);
54         buttons <= 2'b11; @ (posedge clk);
55         buttons <= 2'b00; @ (posedge clk);
56         buttons <= 2'b01; @ (posedge clk);
57         buttons <= 2'b10; @ (posedge clk);
58         buttons <= 2'b11; @ (posedge clk);
59         buttons <= 2'b00; @ (posedge clk);
60         $stop;
61     end
62 endmodule
```