```systemverilog
module UserIn (clk, reset, key, out);
   input logic clk, reset, key;
   output logic out;

   typedef enum logic {A, B} state;
   state ps, ns;

   always_comb begin
      case (ps)
         A: if (key)    ns = B;
            else        ns = A;
         B: if (key)    ns = B;
            else        ns = A;
      endcase
   end

   assign out = ((ps == A) & key);

   always_ff @(posedge clk) begin
      if (reset)
         ps <= A;
      else
         ps <= ns;
   end

endmodule

module UserIn_testbench();

   logic clk, reset, key;
   logic out;

   UserIn dut(clk, reset, key, out);

   parameter CLOCK_PERIOD=100;
   initial begin
      clk <= 0;
      forever #(CLOCK_PERIOD/2) clk <= ~clk;
   end

   initial begin              @(posedge clk);
                              @(posedge clk);
      reset <= 1;             @(posedge clk);
      reset <= 0; key <= 0;   @(posedge clk);
                              @(posedge clk);
                              @(posedge clk);
                              @(posedge clk);
      key <= 1;               @(posedge clk);
      key <= 0;               @(posedge clk);
      key <= 1;               @(posedge clk);
                              @(posedge clk);
                              @(posedge clk);
                              @(posedge clk);
                              @(posedge clk);
                              @(posedge clk);
      key <=0;                @(posedge clk);
                              @(posedge clk);

      $stop;
   end
endmodule
```