Examples of State Machines

Exercise 1:

```
module ex83 ( input reset, clk,
              output logic [7:0] y ) ;

   logic [2:0] n ;
   logic [7:0] seq [0:7] = '{ 3, 5, 1, 7, 6, 1, 8, 9 } ;

   always_ff @(posedge clk) n
      <= reset ? '0 : n+1 ;

   assign y = seq[n] ;

endmodule
```

How would you: (a) change the values in the sequence? (b) change the length of the sequence to 6? (c) stop at the last value?

Exercise 2:

```
module ex82 ( output logic unlock,
              input logic [3:0] in,
              input logic clk ) ;

   logic [2:0] n ;

   always_ff @(posedge clk) n
      <= n == 0 && in == 4 ? 1 :
         n == 1 && in == 7 ? 2 :
         n == 2 && in == 2 ? 3 :
         n == 3 && in == 4 ? 4 :
         0 ;

   assign unlock = n == 4 ;

endmodule
```

Rewrite the module to store the sequence in an unpacked array as in the previous example.

Exercise 3: Write the body of a Verilog module named **`edge`** with input **`in`** and output **`out`** that implements the rising-edge detector.

Exercise 4: Write the state transition table for this state machine.

Exercise 5: Write `always_ff` statements that implement these state machines.