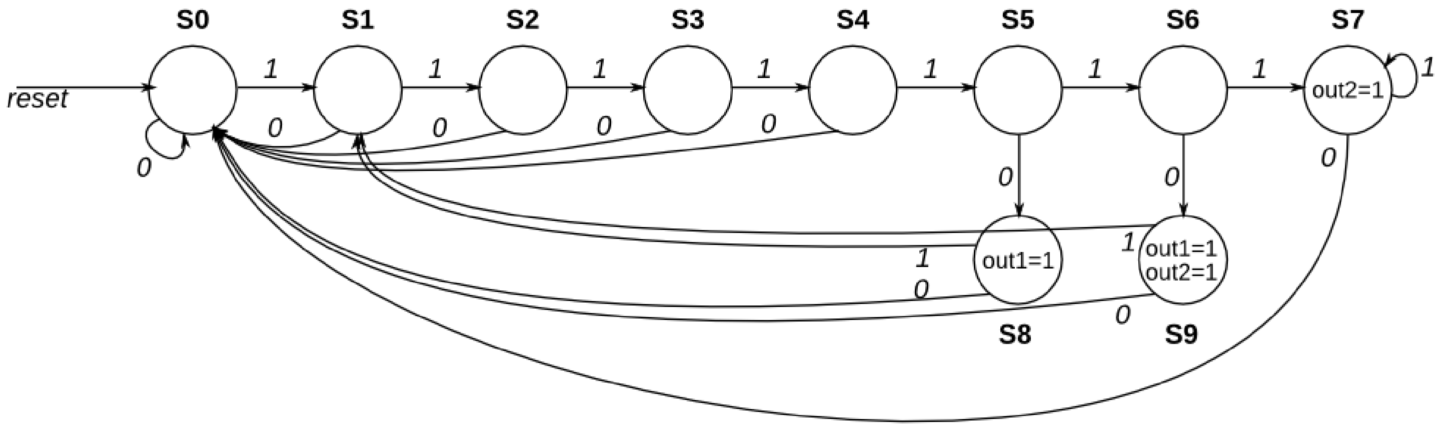
Fsm onehot

Given the following state machine with 1 input and 2 outputs:



Suppose this state machine uses one-hot encoding, where `state[0]` through `state[9]` correspond to the states S0 though S9, respectively. The outputs are zero unless otherwise specified.

Implement the **state transition logic** and **output logic** portions of the state machine (but not the state flip-flops). You are given the current state in `state[9:0]` and must produce `next_state[9:0]` and the two outputs. Derive the logic equations by inspection assuming a one-hot encoding. (The testbench will test with non-one hot inputs to make sure you're not trying to do something more complicated).

```
module top_module(
    input in,
    input [9:0] state,
    output [9:0] next_state,
    output out1,
    output out2);

    always @(*)begin
        next_state[0] = ~in & (|{state[9:7],state[4:0]});
        next_state[1] =  in & state[0] | (|state[9:8] & in);
        next_state[2] =  in & state[1];
        next_state[3] =  in & state[2];
        next_state[4] =  in & state[3];
        next_state[5] =  in & state[4];
        next_state[6] =  in & state[5];
        next_state[7] =  in & (|state[7:6]);
        next_state[8] = ~in & state[5];
        next_state[9] = ~in & state[6];
    end

    assign out1 = state[8] | state[9];
    assign out2 = state[7] | state[9];
endmodule
```