Review Problem 26

- An ambulance company wants a flashing yellow light that, when a button is held, will instead hold a solid red. Design this machine.
Vending Machine Example (cont.)

- State Table:

<table>
<thead>
<tr>
<th>PS</th>
<th>N</th>
<th>D</th>
<th>out</th>
<th>NS</th>
</tr>
</thead>
<tbody>
<tr>
<td>PS</td>
<td>N</td>
<td>D</td>
<td>Open</td>
<td>NS</td>
</tr>
<tr>
<td>O</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>O</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>X</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>X</td>
<td>1</td>
</tr>
</tbody>
</table>

-they will never happen-

\[ \text{out} = D + PS \cdot N \]

\[ NS = \overline{PS} \cdot N + PS \cdot \overline{D} \]
$\text{open} = D \oplus \overline{\text{psn}}$
$\text{ns} = \overline{\text{psn}} + \text{psn} \overline{D}$

Vending Machine Example (cont.)

- Implementation:
FSMs in Verilog - Declarations

module simple (clk, reset, w, out);
    input logic clk, reset, w;
    output logic out;

    enum { A, B, C} ps, ns; // Present state, next state
FSMs in Verilog – Combinational Logic

// Next State Logic
always_comb begin
  case (ps)
    A: if (w) ns = B;
    else ns = A;
    B: if (w) ns = C;
    else ns = A;
    C: if (w) ns = C;
    else ns = A;
  endcase
end

// Output Logic – could also be "always",
// or part of next-state logic.
assign out = (ps == C);
FSMs in Verilog – DFFs

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

    endmodule