Review Problem 23
Finite State Machine: State Diagram

- Finite number of states
- Not transitions
- Moore machines put outputs on states

Note: We cover Mealy machines here.

<Triggers Input pattern / Outputs>

Labels on Transitions:

- Arrows connecting states
- Transitions (actions) represented by circles
- States represented by circles

Graphical diagram of FSM behavior

State Diagrams
Circuit that is true every 4th cycle.

State Diagram Example

\* \* a26925
```
<table>
<thead>
<tr>
<th>Present State</th>
<th>Input</th>
<th>Output</th>
<th>Next State</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
```

Truth table for sequential circuits.
State Table for 4th cycle circuit
1. Understand the problem

2. Draw the state diagram

3. Use state diagram to produce state table

4. Implement the combinational control logic

FSM Design Process
**Problem 24**

Given the light display shown, build the FSM for a

*move left arrow traffic sign. It should animate*

*Reset all outputs met (15) on*

*Hint: Can any of the bulbs be connected to the same*

*any edge are false*

*Given the light display shown, build the FSM for a*

*signal?

\[
\begin{array}{c}
\text{X} \\
\text{Y} \\
\text{Z} \\
\text{A} \\
\text{B} \\
\text{C} \\
\text{D} \\
\text{E} \\
\text{F} \\
\text{G} \\
\text{H} \\
\text{I} \\
\text{J} \\
\text{K} \\
\text{L} \\
\text{M} \\
\text{N} \\
\text{O} \\
\text{P} \\
\text{Q} \\
\text{R} \\
\text{S} \\
\text{T} \\
\text{U} \\
\text{V} \\
\text{W} \\
\text{X} \\
\text{Y} \\
\text{Z}
\end{array}
\]
Vending Machine Example

State Diagram:
- No change returned
- Single coin slot for dimes, nickels
- Deliver package of gum after >= 10 cents deposited

Vending Machine:
- Mechanism
  - Release
    - Gum
  - Open
- FSM
  - Coin
  - Reset
  - D
  - N
- Sensor
  - C输

For each state, follow last input.
$P_{\bar{S}} = P_{S} + P_{D} \times D$

$P_{\bar{S}} = D + P_{S} \times D$

State Table:

Vending Machine Example (cont.)
Vending Machine Example (cont.)

\[ NS = PS \cdot R + PS \cdot D \]

\[ open = D + PS \cdot R \]
enum A, B, C { PS, ns; // Present state, next state

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

FSMs in Verilog - Declarations
assign out = (ps == C);

or part of next-state logic.

Output logic - could also be "alwayse"

endcase

elsecase

A: if (m) us B;

else 

A: if (m) us B;

elsecase

always_comb begin

endcase

Next State Logic

always_comb begin

endcase

endcase

endcase

endcase

endcase

endcase

endcase

endcase

endcase

endcase

endcase

endcase

endcase

endcase

endcase
endmodule
end
ps := us;
else
ps := A;
if (reset)
always @ (posedge clk) begin

// Sequential Logic (DFF)

RSMs in Verilog – DFFs