2013 Fall CSE140L

*Digital Systems Laboratory*

Lecture #8

by

Dr. Choon Kim
CSE Department, UCSD
LAB#3: Vending Machine Controller Design

• It has many states – Init, zero cents, 5 cents, 10 cents, ....
• How to move from one state to other state?
• How to detect & handle dispensing?
• How to detect & handle One-dollar bill, credit-card and coin input?
• How to handle special cases?
• ....
Hint: LAB3 works to be implemented

- VM (Vending Machine) operation:
  - two always blocks (one for state transition, the other for output & next state determination)
  - special cases handling

- Transition between initial state and VM-enabled modes:
  - initial -> VM-enabled
  - VM-enabled -> initial

- Reading inputs (sw0,1,2,5,9, etc.):
  - always(*).... or, assign....
Suggested Approach to LAB#3

Let's do quick review of topic from last class first, ....
Mealy vs. Moore FSM

Mealy Machine: \( y(t) = f(x(t), s(t)) \)
Moore Machine: \( y(t) = f(s(t)) \)

\[ s(t+1) = g(x(t), s(t)) \]

Mealy Machine:

Moore Machine:
FSM design example – Moore vs. Mealy

A circuit which removes one 1 (i.e., the first 1) from every string of 1s on the input stream:

<table>
<thead>
<tr>
<th>input</th>
<th>output</th>
</tr>
</thead>
<tbody>
<tr>
<td>..... 00000...... =&gt; ..... 00000......</td>
<td></td>
</tr>
<tr>
<td>..... 01010...... =&gt; ..... 00000......</td>
<td></td>
</tr>
<tr>
<td>..... 00110...... =&gt; ..... 00100......</td>
<td></td>
</tr>
<tr>
<td>..... 01110...... =&gt; ..... 01100......</td>
<td></td>
</tr>
<tr>
<td>..... 11011...... =&gt; ..... 10010......</td>
<td></td>
</tr>
</tbody>
</table>
FSM design example – Moore vs. Mealy

- Remove one 1 from every string of 1s on the input
Moore FSM Verilog model: (Not necessary working solution)

```verilog
module reduce (input       clk,   in,
                output     reg   out);

parameter      zero = 2'b00,   one1 = 2'b01,   two1s = 2'b10;
reg[2:1]    state,  next_state;

always @(posedge clk) begin
    state = next_state;
end

always  @(in,   state)  begin
    case (state)
        zero:    begin
            out = 0;
            if (in)     next_state= one1;
            else       next_state= zero;
        end
        one1:    begin
            out  = 0;
            if (in)     next_state= two1s;
            else       next_state= zero;
        end
        two1s:   begin
            out = 1;
            if  (in)    next_state= two1s;
            else        next_state= zero;
        end
        default:  begin
            out = 1;
            next_state= two1s;
        end
    endcase
end
end module
```

add default: statement here...
**Mealy FSM Verilog model:** *(Not necessary working solution)*

```verilog
module reduce (input  clk,  in,
                output  reg   out);

parameter  zero = 1'b0,   one = 1'b1;
reg   state,  next_state;

always @(posedge  clk)  begin
  state = next_state;
end

always  @(in,   state)  begin
  case (state)
    zero:    begin
              out = 0;
              if (in)     next_state= one;
              else       next_state= zero;
            end
    one:    begin
              out  = 0;
              if (in)    begin
                          out = 1;
                          next_state= one;
              end
              else      begin
                          out = 0;
                          next_state= zero;
              end
            end
  endcase
end
end module
```

![ FSM Diagram ]

*add default: statement here...*