Electronics and
Computer Science

UNIVERSITY OF
Southampton

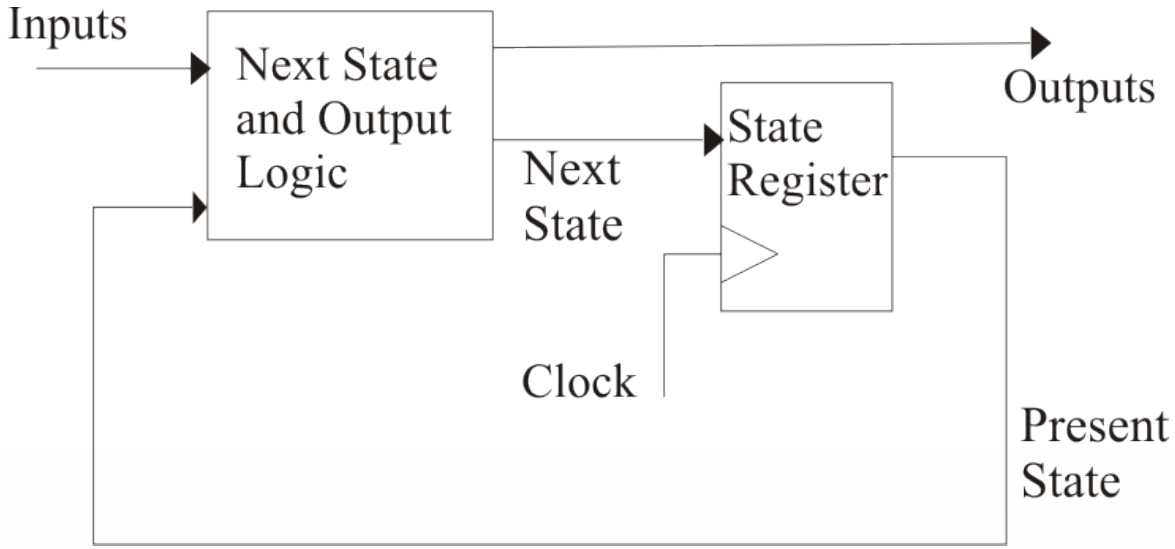
6. SystemVerilog for Sequential Design (2)

Electronics and
Computer Science

UNIVERSITY OF
Southampton

6.1 State Machines

SystemVerilog Model of State Machine



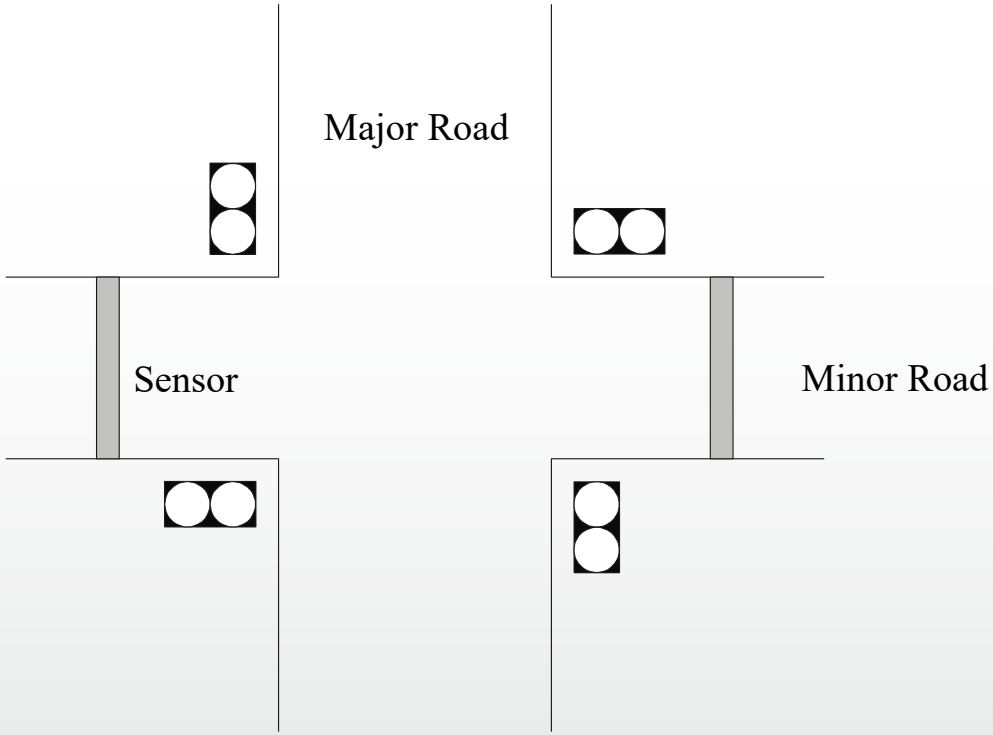
- SystemVerilog describes concurrent hardware
  - Each always block describes a piece of hardware
  - Therefore we can have more than one always block in a module
  - This model has one combinational block (Next State and Output) and one sequential (State Register)

SystemVerilog Model of State Machine

- Use an enumerated type for the states
  - Don't need to do a state assignment
- Combinational part is modelled with always_comb
  - Assign to next state and to outputs
  - Case statement – one branch for each state
  - If statements in each branch for Mealy (conditional) outputs and for selecting next state
  - Default assignments to each output and next state at start, to avoid accidental memory
- Sequential part is modelled with always_ff
  - Just like a flip-flop

State Machine Example

- Traffic Light Controller



Lights can be:

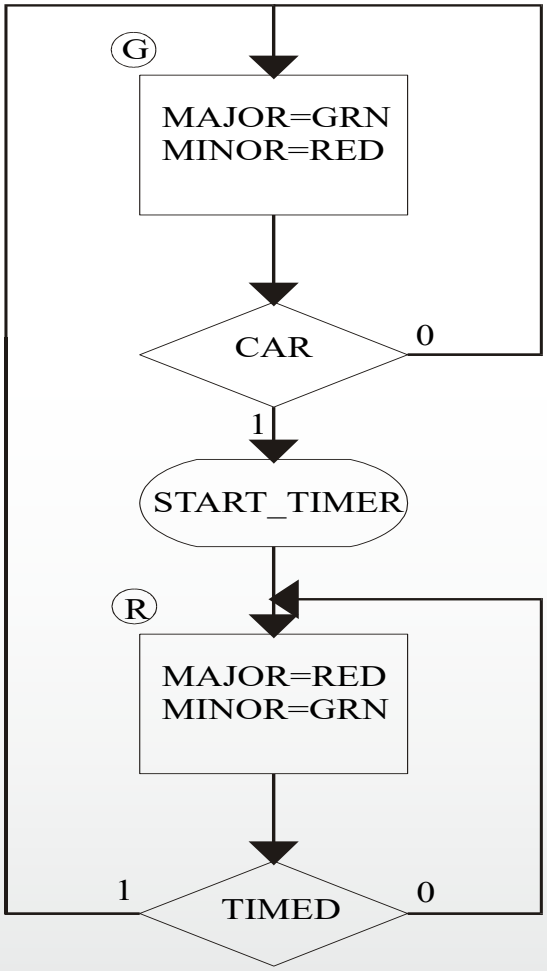
Major Road Red
Minor Road Green

or:

Major Road Green
Minor Road Red

2 states

ASM Chart



Two states (G and R)

Two inputs (CAR and TIMED)

Three outputs

(Minor=Green/Major=Red;
Major=Green/Minor=Red;
Start_timer)

```
module traffic (output logic start_timer,
    major_green, minor_green,
    input logic clock, n_reset, timed, car);

enum {G, R} present_state, next_state;

always_ff @(posedge clock, negedge n_reset)
  begin: SEQ //label
    if (!n_reset)
      present_state <= G;
    else
      present_state <= next_state;
  end
```

```
always_comb
  begin: COM
  start_timer = '0;
  minor_green = '0;
  major_green = '0;
  next_state = present_state;
  unique case (present_state)
    G: begin
       major_green = '1;
       if (car)
         begin
         start_timer = '1;
         next_state = R;
         end
       end
    R: begin
       minor_green = '1;
       if (timed)
         next_state = G;
       end
    endcase
  end
endmodule
```

Comments

- present_state and next_state must be variables of enumerated type
- Care is needed with combinational process - can easily create latches - *BAD!*
  - Default values for outputs and next_state
- always blocks are labelled – need begin and end.
- unique case checks that all states are included

- There are other ways to do this, but this is recommended.

For you to do

- How would you build the timer? Can you find a suitable design from an earlier lecture?