State Machines Tutorial

CSE 141 L
Finite State Machines (FSMs)

- Useful for designing many different types of circuits
- 3 basic components:
  - Combinational logic (next state)
  - Sequential logic (store state)
  - Output logic
- Different encodings for state:
  - Binary (min FF’s), Gray, One hot (good for FPGA), One cold, etc
module simple_fsm(input clk, start, output restart);
reg [1:0] state, next_state;
parameter S0 = 2’d00, S1 = 2’d01, S2 = 2’d10; // binary encode
always @ (*)
begin : next_state_logic
    case (state)
        S0: begin
            if (start) next_state = S1;
            else next_state = S0;
        end
        S1: begin next_state = S2; end
        S2: begin
            if (restart) next_state = S0;
            else next_state = S2;
        end
        default: next_state = S0;
    endcase
end // continued to the right

// continued from left
always @(posedge clk)
begin : state_assignment
    state <= next_state;
end
endmodule
Tips on FSMs

● Don’t forget to handle the default case
● Use two different always blocks for next state and state assignment
  ● Can do it in one big block but not as clear
● Outputs can be a mix of combin. and seq.
  ● Moore Machine: Output only depends on state
  ● Mealy Machine: Output depends on state and inputs