`define GREEN 2'b00

`define YELLOW 2'b01

`define RED 2'b10

`define S0 3'b000 // Highway Green, Country Red

`define S1 3'b001 // Highway Yellow, Country Red

`define S2 3'b010 // Highway Red, Country Red

`define S3 3'b011 // Highway Red, Country Green

`define S4 3'b100 // Highway Red, Country Yellow

module sig\_control (

output reg [1:0] hwy,

output reg [1:0] cntry,

input X, // Car on country road

input clock,

input clear

);

reg [2:0] state, next\_state;

// Initial state

initial begin

state = `S0;

next\_state = `S0;

hwy = `GREEN;

cntry = `RED;

end

// State transition on clock

always @(posedge clock) begin

if (clear)

state <= `S0;

else

state <= next\_state;

end

// Output logic based on state

always @(\*) begin

case (state)

`S0: begin hwy = `GREEN; cntry = `RED; end

`S1: begin hwy = `YELLOW; cntry = `RED; end

`S2: begin hwy = `RED; cntry = `RED; end

`S3: begin hwy = `RED; cntry = `GREEN; end

`S4: begin hwy = `RED; cntry = `YELLOW; end

default: begin hwy = `GREEN; cntry = `RED; end

endcase

end

// Next-state logic

always @(\*) begin

case (state)

`S0: next\_state = (X) ? `S1 : `S0;

`S1: next\_state = `S2; // after yellow delay

`S2: next\_state = `S3; // after all-red delay

`S3: next\_state = (X) ? `S3 : `S4;

`S4: next\_state = `S0; // after yellow delay

default: next\_state = `S0;

endcase

end

endmodule

testbench

`define TRUE 1'b1

`define FALSE 1'b0

module stimulus;

wire [1:0] MAIN\_SIG, CNTRY\_SIG;

reg CAR\_ON\_CNTRY\_RD;

reg CLOCK, CLEAR;

// Instantiate signal controller

sig\_control SC(

.hwy(MAIN\_SIG),

.cntry(CNTRY\_SIG),

.X(CAR\_ON\_CNTRY\_RD),

.clock(CLOCK),

.clear(CLEAR)

);

// Monitor outputs

initial begin

$monitor($time, " Main Sig = %b Country Sig = %b Car\_on\_cntry = %b",

MAIN\_SIG, CNTRY\_SIG, CAR\_ON\_CNTRY\_RD);

end

// Clock generation

initial begin

CLOCK = `FALSE;

forever #5 CLOCK = ~CLOCK; // 10 time units per clock cycle

end

// Control clear signal

initial begin

CLEAR = `TRUE;

repeat (5) @(negedge CLOCK);

CLEAR = `FALSE;

end

// Apply stimulus

initial begin

CAR\_ON\_CNTRY\_RD = `FALSE;

#200 CAR\_ON\_CNTRY\_RD = `TRUE;

#100 CAR\_ON\_CNTRY\_RD = `FALSE;

#200 CAR\_ON\_CNTRY\_RD = `TRUE;

#100 CAR\_ON\_CNTRY\_RD = `FALSE;

#200 CAR\_ON\_CNTRY\_RD = `TRUE;

#100 CAR\_ON\_CNTRY\_RD = `FALSE;

#100 $stop;

end

endmodule