// Combined heating\_automation module

module heating\_automation(

input wire clk, // System clock

input wire reset, // System reset

input wire [7:0] temp\_sensor, // Temperature sensor input (8-bit temperature value)

input wire manual\_override, // Manual override switch

input wire heating\_enable, // Heating system enable

output reg heater // Output control for the heater

);

// State definitions

typedef enum logic [1:0] {

OFF = 2'b00,

ON = 2'b01

} state\_t;

state\_t current\_state, next\_state;

// Parameters

parameter TEMP\_THRESHOLD = 8'd20; // Temperature threshold (e.g., 20 degrees)

// Sequential block for state transitions

always @(posedge clk or posedge reset) begin

if (reset) begin

current\_state <= OFF;

end else begin

current\_state <= next\_state;

end

end

// Combinational logic for next state

always @(\*) begin

case (current\_state)

OFF: begin

if (manual\_override || (heating\_enable && (temp\_sensor < TEMP\_THRESHOLD))) begin

next\_state = ON;

end else begin

next\_state = OFF;

end

end

ON: begin

if (!manual\_override && (temp\_sensor >= TEMP\_THRESHOLD)) begin

next\_state = OFF;

end else begin

next\_state = ON;

end

end

default: next\_state = OFF;

endcase

end

// Output logic

always @(posedge clk or posedge reset) begin

if (reset) begin

heater <= 1'b0;

end else begin

case (next\_state)

OFF: heater <= 1'b0;

ON: heater <= 1'b1;

default: heater <= 1'b0;

endcase

end

end

endmodule

// Testbench for heating\_automation

module tb\_heating\_automation;

reg clk;

reg reset;

reg [7:0] temp\_sensor;

reg manual\_override;

reg heating\_enable;

wire heater;

// Instantiate the module

heating\_automation uut (

.clk(clk),

.reset(reset),

.temp\_sensor(temp\_sensor),

.manual\_override(manual\_override),

.heating\_enable(heating\_enable),

.heater(heater)

);

// Clock generation

initial begin

clk = 0;

forever #5 clk = ~clk; // 10ns clock period

end

// Test sequence

initial begin

// Initialize inputs

reset = 1;

temp\_sensor = 8'd25;

manual\_override = 0;

heating\_enable = 0;

#10 reset = 0;

// Test: Heating enable with low temperature

#10 heating\_enable = 1; temp\_sensor = 8'd18; // Heater should turn ON

#20 temp\_sensor = 8'd22; // Heater should turn OFF

// Test: Manual override

#10 manual\_override = 1; // Heater should turn ON regardless of temp

#20 manual\_override = 0; temp\_sensor = 8'd25; // Heater should turn OFF

// Test: Reset condition

#10 reset = 1; // Heater should turn OFF

#10 reset = 0; // Resume operation

#50 $stop; // End simulation

end

endmodule