# Post Lab //Mod 10 counter //RA2111004010177

module mod\_10\_counter\_177(

input clk, // Clock input input reset, // Reset input output reg [3:0] count // 4-bit counter output

);

always @(posedge clk or posedge reset) begin if (reset) begin count <= 4'b0000; // Reset to 0 end else begin if (count == 4'b1001) begin

count <= 4'b0000; // Reset to 0 when count reaches 9 end else begin

count <= count + 1; // Increment count end end end endmodule

//Test bench for mod 10 counter //RA2111004010177 module mod\_10\_counter\_using\_tff\_177(

input clk, // Clock input input reset, // Reset input output reg [3:0] count // 4-bit counter output

);

reg [3:0] tff\_in; // Input for T flip-flops // Instantiate 4 T flip-flops for the counter tff\_177 tff\_0(.clk(clk), .t(tff\_in[0]), .q(count[0])); tff\_177 tff\_1(.clk(clk), .t(tff\_in[1]), .q(count[1])); tff\_177 tff\_2(.clk(clk), .t(tff\_in[2]), .q(count[2])); tff\_177 tff\_3(.clk(clk), .t(tff\_in[3]), .q(count[3]));

always @(posedge clk or posedge reset) begin if (reset) begin tff\_in <= 4'b0000; // Reset to 0 end else begin if (count == 4'b1001) begin

tff\_in <= 4'b0000; // Reset to 0 when count reaches 9 end else begin

tff\_in <= tff\_in + 4'b0001; // Increment count end end end endmodule

# //Test Bench for mod 10 //RA2111004010177

module mod\_10\_counter\_using\_tff\_tb\_177;

// Parameters

parameter CLK\_PERIOD = 10; // Clock period in ns

// Signals

reg clk = 0; // Clock signal reg reset = 0; // Reset signal wire [3:0] count; // Counter output // Instantiate the mod 10 counter mod\_10\_counter\_using\_tff mod\_10\_counter\_inst (

.clk(clk),

.reset(reset),

.count(count)

);

// Clock generation always #((CLK\_PERIOD/2)) clk = ~clk; // Initial stimulus initial begin // Apply reset for a few cycles reset = 1; #20; reset = 0;

// Test counter for 100 clock cycles repeat (100) begin #CLK\_PERIOD; end // End simulation end endmodule //output

