Nang Thiri Wutyi
20113
11/26/2024

Quiz 2

1. a. **The critical path**

Clock-to-Q Delay: 40 ps (min),60 ps (max)
Propagation Delay per Gate: 80 ps (min),100 ps (max)
Setup Time: 50 ps
Hold Time: 45 ps

Direct paths

Path 1: From R1 → G2 → G4 → R2

Delay=Clock-to-Q (R1)+G2 Delay+G4 Delay+Setup Time (R2)

Max Delay=60 ps+100 ps+100 ps+50 ps=310 ps

Path 2: From R1 → G3 → G4 → R2

Delay=Clock-to-Q (R1)+G3 Delay+G4 Delay+Setup Time (R2)

Max Delay=60 ps+100 ps+100 ps+50 ps=310 ps

Feedback path

R1→G3→G2→G4→R2

Delay:

Clock-to-Q of R1: 60 ps (max)

G3G3G3: 100 ps (max)

G2G2G2: 100 ps (max)100

G4G4G4: 100 ps (max)100

Setup time of R2: 50 ps

Total Delay: 60+100+100+100+50=410 ps

Critical Path is **R1→G3→G2→G4→R2**

b. Maximum clock rate

$f = \frac{1}{Tcritical} = \frac{1}{410p} = 2.44$ GHz

c. Hold-time violation?

thold(R2)≤tclock-to-q(R1)+tcombinational_path_min.

Path 1: R1→G2→G4→R2R1 → G2 → G4 → R2R1→G2→G4→R2

Minimum propagation delays:

G2: 80 ps

G4: 80 ps

Total tcombinational_path_min=80+80=160 ps

Path 2: R1→G3→G4→R2R1 → G3 → G4 → R2

tcombinational_path_min=160 ps

The **minimum delay** for any data path is:

tclock-to-q(R1)+tcombinational_path_min=40+160=200 ps.

This 200 ps delay is larger than thold(R2)=45ps, ensuring the hold time is satisfied.

The register hold-time cannot be violated.

2. **Design Module**

```
module fsm_module (

    input wire clk,

    input wire reset,

    input wire out1,

    input wire out2,

    input wire out3,

    output reg r1,

    output reg r2,
    output reg r3
);
    // State encoding
    typedef enum reg [1:0] {Idle=2'b00, G1=2'b01, G2=2'b10, G3=2'b11} state_t;
    state_t state, next_state;

    // Sequential logic for state transition
    always @(posedge clk or posedge reset) begin
        if (reset)
            state <= Idle;
        else
            state <= next_state;
    end

    // Combinational logic to decide next state and outputs
    always @(*) begin
        // Default values for r1, r2, r3
        r1 = 0;
        r2 = 0;
        r3 = 0;
        next_state = state;

        case(state)
            Idle: begin
    if (out1 == 1) begin

        r1 = 1;

        next_state = G1;

    end

    else if (out2 == 1) begin

        r2 = 1;

        next_state = G2;

    end

    else if (out3 == 1) begin

        r3 = 1;

        next_state = G3;

    end

end

G1: begin

    r1 = 1;  // Stay in G1, output r1=1

    next_state = Idle;  // Return to Idle after G1

end

G2: begin

    r2 = 1;  // Stay in G2, output r2=1

    next_state = Idle;  // Return to Idle after G2

end

G3: begin

    r3 = 1;  // Stay in G3, output r3=1

    next_state = Idle;  // Return to Idle after G3
```

```
            end
            default: next_state = Idle;  // Default state
        endcase
    end
endmodule
```

Testbench Module

```
module tb_fsm_module;
    reg clk;
    reg reset;
    reg out1, out2, out3;
    wire r1, r2, r3;

    // Instantiate the module
    fsm_module uut (
        .clk(clk),
        .reset(reset),
        .out1(out1),
        .out2(out2),
        .out3(out3),
        .r1(r1),
        .r2(r2),
        .r3(r3)
    );
```

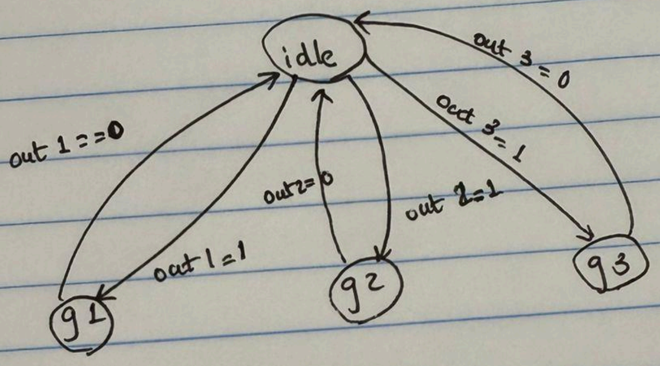
```
// Clock generation

always #5 clk = ~clk;


// Test vectors

initial begin

    // Initializing values

    clk = 0;

    reset = 1;

    out1 = 0;

    out2 = 0;

    out3 = 0;


    // Apply reset

    #10 reset = 0;


    // Test case 1: out1 = 1, transition to G1

    #10 out1 = 1;

    #10 out1 = 0;


    // Test case 2: out2 = 1, transition to G2

    #10 out2 = 1;

    #10 out2 = 0;
```

```
        // Test case 3: out3 = 1, transition to G3

        #10 out3 = 1;

        #10 out3 = 0;


        // Finish the simulation

        #10 $finish;

    end


    // Monitor signals

    initial begin

        $monitor("Time=%0t, reset=%b, out1=%b, out2=%b, out3=%b, r1=%b, r2=%b, r3=%b",

                 $time, reset, out1, out2, out3, r1, r2, r3);

    end

endmodule
```

Result

```
[2024-11-26 19:55:57 UTC] iverilog '-Wall' '-g2012' design.sv testbench.sv  &&
unbuffer vvp a.out

Time=0, reset=1, out1=0, out2=0, out3=0, r1=0, r2=0, r3=0

Time=10, reset=0, out1=0, out2=0, out3=0, r1=0, r2=0, r3=0

Time=20, reset=0, out1=1, out2=0, out3=0, r1=1, r2=0, r3=0

Time=25, reset=0, out1=1, out2=0, out3=0, r1=1, r2=0, r3=0

Time=30, reset=0, out1=0, out2=0, out3=0, r1=1, r2=0, r3=0

Time=35, reset=0, out1=0, out2=0, out3=0, r1=0, r2=0, r3=0

Time=40, reset=0, out1=0, out2=1, out3=0, r1=0, r2=1, r3=0

Time=45, reset=0, out1=0, out2=1, out3=0, r1=0, r2=1, r3=0

Time=50, reset=0, out1=0, out2=0, out3=0, r1=0, r2=1, r3=0
```

```
Time=55, reset=0, out1=0, out2=0, out3=0, r1=0, r2=0, r3=0

Time=60, reset=0, out1=0, out2=0, out3=1, r1=0, r2=0, r3=1

Time=65, reset=0, out1=0, out2=0, out3=1, r1=0, r2=0, r3=1

Time=70, reset=0, out1=0, out2=0, out3=0, r1=0, r2=0, r3=1

Time=75, reset=0, out1=0, out2=0, out3=0, r1=0, r2=0, r3=0

testbench.sv:47: $finish called at 80 (1s)

Done
```


idle
out 3 = 0
out 3 = 1
out 1 ==0
out2=0
out 2=1
out 1=1
g1
g2
g3