INTERNSHIP TASK -4

DELIVERABLE: VERILOG CODE,SIMULATION RESULTS, AND PERFORMANCE ANALYSIS

1. Simple Verilog Code (4-tap FIR Filter)

```
module FIR_Filter (
    input clk,
    input reset,
    input [3:0] x,       // 4-bit unsigned input
    output reg [7:0] y   // 8-bit output
);
    reg [3:0] x_reg0, x_reg1, x_reg2, x_reg3; // shift registers
    parameter h0 = 1, h1 = 2, h2 = 3, h3 = 4; // coefficients

    always @(posedge clk or posedge reset) begin
        if (reset) begin
            x_reg0 <= 0; x_reg1 <= 0;
            x_reg2 <= 0; x_reg3 <= 0;
            y <= 0;
        end else begin
            // shift inputs
            x_reg3 <= x_reg2;
            x_reg2 <= x_reg1;
            x_reg1 <= x_reg0;
            x_reg0 <= x;
```

```
            // FIR equation: y[n] = h0*x[n] + h1*x[n-1] + h2*x[n-2] + h3*x[n-3]
            y <= h0*x_reg0 + h1*x_reg1 + h2*x_reg2 + h3*x_reg3;
        end
    end
endmodule
```

2. Simple Testbench

```
module tb_FIR_Filter;
    reg clk;
    reg reset;
    reg [3:0] x;
    wire [7:0] y;

    FIR_Filter uut (
        .clk(clk),
        .reset(reset),
        .x(x),
        .y(y)
    );

    // Clock generation
    initial clk = 0;
    always #5 clk = ~clk;

    // Stimulus
    initial begin
        $display("Time\tX\tY");
        $monitor("%g\t%d\t%d", $time, x, y);
        reset = 1; x = 0;
        #10 reset = 0;
```

```
        x = 1; #10;
        x = 2; #10;
        x = 3; #10;
        x = 4; #10;
        x = 5; #10;
        x = 6; #10;
        $finish;
    end
endmodule
```

3. Expected Simulation Output (Text Only)

Assuming input pattern [1, 2, 3, 4, 5, 6], here's what the output would look like:

Time	X	Y
0	0	0
10	1	1
20	2	1*1 + 2*2 = 5
30	3	1*2 + 2*1 + 3*3 = 15
40	4	1*3 + 2*2 + 3*1 + 4*4 = 30
50	5	1*4 + 2*3 + 3*2 + 4*1 = 30
60	6	1*5 + 2*4 + 3*3 + 4*2 = 40

4. Performance (Simple Version)

- **Latency:** 4 clock cycles (due to 4-tap shift register)
- **Throughput:** 1 output per clock cycle after the first 4 inputs
- **Hardware Use:**
    - 4 shift registers
    - 4 constant multipliers
    - 1 adder tree

Summary

Component	Value
Type	4-tap FIR filter
Bit Width	4-bit input, 8-bit output
Simulated?	Yes, with testbench
Easy to Implement?	Yes

------Thank you-------