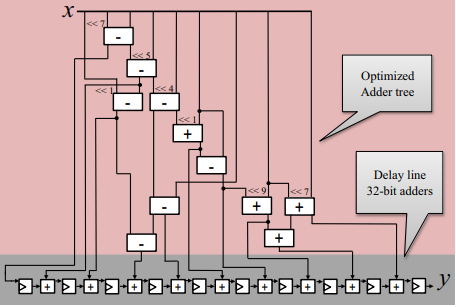
**Topic: Design of area and energy-efficient digital CMOS FIR filters with approximate adder circuits**

**T Raja Aadhithan**

**602162021**

**Expected RTL:**

****

**Code DUT:**

module fir(input [31:0]x, output [31:0]y, input clk); // 32 bit input,output

    reg [31:0] s1,s2,s3,s4,s5,s6,a1,a2,a3,a4,a5; // for inter connect transfers

    reg [31:0] d[9:0]; //used for delay [1/z]

    reg [31:0] sum[8:0]; // add previous time slot output to current slot

    // combinational logics: happens at x(n)

    always@(\*)begin

        s1 = (x << 7) - x;  //subracting x from x^128

        s2 = s1 - (x << 5); //subtracting x^32 from s1

        s3 = (x << 1) - s2; //subtracting s2 from x^2

        s4 = s2 - (x << 4); //subtracting x^16 from s2

        s5 = s4 - x;        //subtracting x from s4

        a1 = s4 + (x << 1); //adding x^128 and x

        s6 = a1 - x;        //subtracting x from a1

        a2 = s6 + (x << 9); //adding x^512 and s6

        a3 = (x << 7) + x;  //adding x^128 and x

        a4 = a2 + a3;       //adding a2 and a3

        a5 = s3 + s5;       //adding s3 and s5

    end

    // adds x(n-1) output to x(n)

    always@(\*)begin

        sum[0] = d[0] + s2;

        sum[1] = d[1] + s3;

        sum[2] = d[2] + a5;

        sum[3] = d[3] + s5;

        sum[4] = d[4] + a1;

        sum[5] = d[5] + s6;

        sum[6] = d[6] + a2;

        sum[7] = d[7] + a4;

        sum[8] = d[8] + a3;

    end

    //synchronizes the output with respect to the clock as well as add delay

    always@(posedge clk)begin

        d[0] <= s1;

        d[1] <= sum[0];

        d[2] <= sum[1];

        d[3] <= sum[2];

        d[4] <= sum[3];

        d[5] <= sum[4];

        d[6] <= sum[5];

        d[7] <= sum[6];

        d[8] <= sum[7];

        d[9] <= sum[8];

    end

    assign y = d[9]; // final output is transferred to the block output

endmodule

**Code TB:**

module fir\_tb;

    reg [31:0] x;   // 32 bit input to FIR

    reg clk = 0;    // Clock Signal

    wire [31:0] y;  // 32 bit output fron FIR

    fir ut(x,y,clk); // Istantiating the DUT

    initial forever #5 clk = !clk; // setting clock period as 10ns

    initial begin

        x = 1; #100; // x[n-9] to x[9] will have the same input 1

        $display("for x = %0d y is %0d",x,y);

        x = 2; #100; // x[n-9] to x[9] will have the same input 2

        $display("for x = %0d y is %0d",x,y);

        x = 3; #50;//x[n-9] to x[n-5] will have the input 2, x[n-4] to x[n] will have input 3

        $display("for x = %0d y is %0d",x,y);

        x = 4; #50;//x[n-9] to x[n-5] will have the input 3, x[n-4] to x[n] will have input 4

        $display("for x = %0d y is %0d",x,y);

        x = 124; #50; // for 1st 5 cycles

        x = 1116988; #50; // for 2nd 5 cycles

        $display("for x = %0d y is %0d",x,y);

        x = 42; #20; // for 2 cycles

        x = 124; #30; // for 3 cycles

        x = 34; #20; // for 2 cycles

        x = 67; #20; // for 2 cycles

        x = 1116988; #10; // for 1 cycle

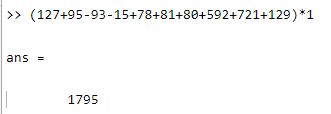
        $display("for x = %0d y is %0d",x,y);

    end

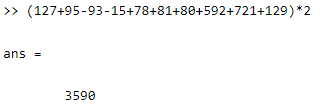
endmodule

**Expected Output from MATLAB:**

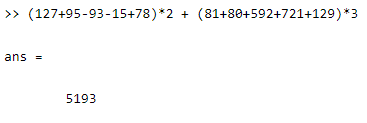
1. When x = 1:



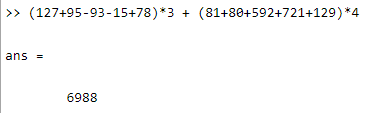
1. When x = 2:



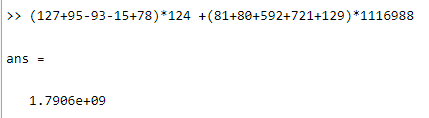
1. When x = 2 for 5 units and 3 for 5 units:



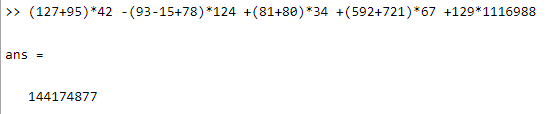
1. When x = 3 for 5 units and 4 for 5 units:



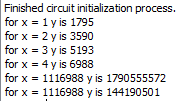
1. When x = 124 for 5 units and 1116988 for 5 units:



1. When x is changing frequently:



**Output from Xilinx ISE:**

****

All the 6 cases have given the same output.