Lab 07

Handout#07: FIR Filter using Verilog HDL

Designing FIR filter ‘

Design Code

```
21  // Malak Majeedullah khan
22  module FIR_Filter(clk, reset, data_in, data_out);
23  parameter N = 16;
24  input clk, reset;
25  input [N-1:0] data_in;
26  output reg [N-1:0] data_out;
27
28  // coefficients defination
29  // Moving Average Filter, 3rd order
30  // four coefficients; 1/(order+1) = 1/8 = 0.125
31  // 0.125 x 128(scaling factor) = 16 = 6'b010000
32  wire [5:0] b0 =  6'b010000;
33  wire [5:0] b1 =  6'b010000;
34  wire [5:0] b2 =  6'b010000;
35  wire [5:0] b3 =  6'b010000;
36  wire [5:0] b4 =  6'b010000;
37  wire [5:0] b5 =  6'b010000;
38  wire [5:0] b6 =  6'b010000;
39  wire [5:0] b7 =  6'b010000;
40  wire [N-1:0] x1, x2, x3,x4,x5,x6,x7;
41
```

```
42      // Create delays i.e x[n-1], x[n-2], .. x[n-N]
43      // Instantiate D Flip Flops
44      DFF DFF0(clk, 0, data_in, x1); // x[n-1]
45      DFF DFF1(clk, 0, x1, x2);      // x[x[n-2]]
46      DFF DFF2(clk, 0, x2, x3);      // x[n-3]
47      DFF DFF3(clk, 0, x3, x4);      // x[x[n-2]]
48      DFF DFF4(clk, 0, x4, x5);
49      DFF DFF5(clk, 0, x5, x6);      // x[x[n-2]]
50      DFF DFF6(clk, 0, x6, x7);
51
52
53      //  Multiplication
54      wire [N-1:0] Mul0, Mul1, Mul2, Mul3,Mul4,Mul5,Mul6,Mul7;
55      assign Mul0 = data_in * b0;
56      assign Mul1 = x1 * b1;
57      assign Mul2 = x2 * b2;
58      assign Mul3 = x3 * b3;
59      assign Mul4 = x4 * b4;
60      assign Mul5 = x5 * b5;
61      assign Mul6 = x6 * b6;
62      assign Mul7 = x7 * b7;
```

```
64
65      // Addition operation
66      wire [N-1:0] Add_final;
67      assign Add_final = Mul0 + Mul1 + Mul2 + Mul3 + Mul4 + Mul5 + Mul6 + Mul7;
68
69      // Final calculation to output
70      always@(posedge clk)
71      data_out <= Add_final;
72      endmodule
73      module DFF(clk, reset, data_in, data_delayed);
74      parameter N = 16;
75      input clk, reset;
76      input [N-1:0] data_in;
77      output reg [N-1:0] data_delayed;
78
79      always@(posedge clk, posedge reset)
80      begin
81          if (reset)
82          data_delayed <= 0;
83          else
84          data_delayed <= data_in;
85      end
86      endmodule
87
88
```

Test bench

F:/8th semster/DSD/lab 01/lab_07/lab_07.srcs/sources_1/new/FIR_Filter.v

```
 89      module FIR_TB;
 90      parameter N = 16;
 91      reg clk, reset;
 92      reg [N-1:0] data_in;
 93      wire [N-1:0] data_out;
 94      FIR_Filter inst0(clk, reset, data_in, data_out);
 95      // input sine wave data
 96      initial
 97      $readmemb("F:\8th semster\DSD\lab 01\DSDLab-Sheets/signal.data", RAMM);
 98      // Create the RAM
 99      reg [N-1:0] RAMM [31:0];
100      // create a clock
101      initial
102      clk = 0;
103      always
104      #10 clk = ~ clk;
105
106      // Read RAMM data and give to design
107      always@(posedge clk)
108          data_in <= RAMM[Address];
109
110      // Address counter
111      reg [4:0] Address;
112      initial
113      Address = 1;
114      always@(posedge clk)
115      begin
116          if (Address == 31)
117              Address = 0;
118          else
119              Address = Address + 1;
120      end
121
122      endmodule
```