Lab 01

Vivado Design Flow, Verilog Gate Level & Hierarchical Modelling

1. Design, Implement and investigate a 4-bit parallel adder by cascading four Full Adders.

Part (a). Implement the Full Adder using gate-level modelling and test the design to make sure it is working.

Submit the Verilog design and test bench code screenshots with name signatures. Also submit the tcl console and timing diagram outputs.

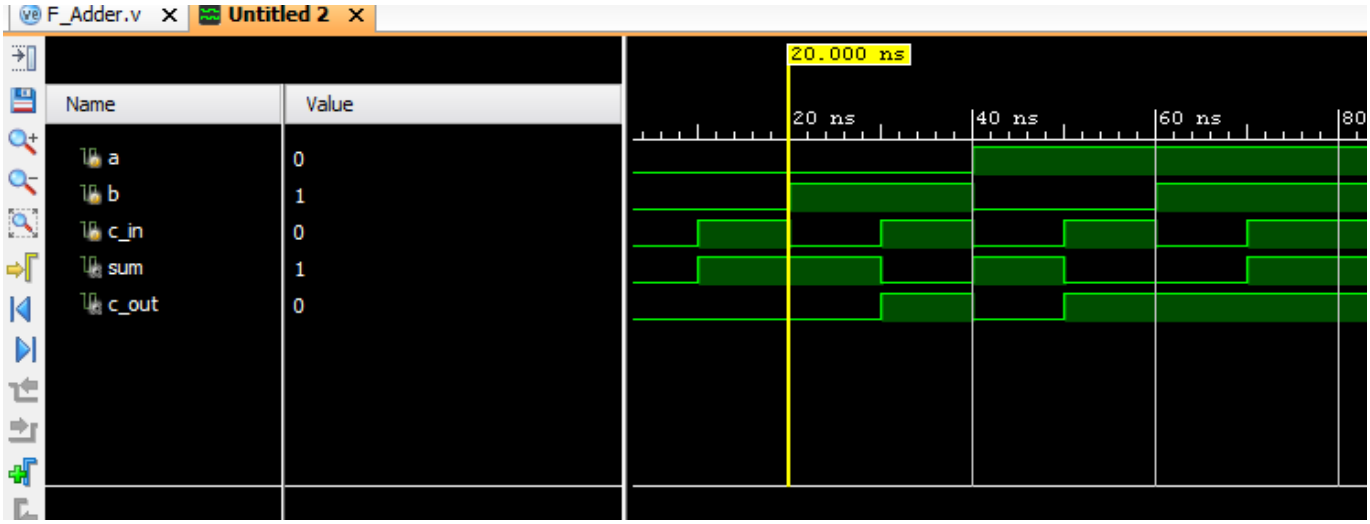
Verilog code gate level modelling (Full Adder)

```
22
23      //Malak Majeed Ullah khan  0344-8952407
24      module Full_Adder(sum, c_out, a, b, c_in);
25      input wire a, b, c_in;
26      output wire sum, c_out;
27
28      // wires
29      wire s1, c1, s2;
30
31      // logic description
32      xor xor1(s1, a, b);
33      xor xor2(sum, s1, c_in);
34
35      and and1(c1, a, b);
36      and and2(s2, s1, c_in);
37      xor xor3(c_out, c1, s2);
38
39      endmodule
40
41
```

Test Bench

```
42  module F_AdderTB();
43  reg a, b, c_in;
44  wire sum, c_out;
45
46  initial
47  begin
48    #00  a = 1'b0; b = 1'b0; c_in = 1'b0;
49    #10  a = 1'b0; b = 1'b0; c_in = 1'b1;
50    #10  a = 1'b0; b = 1'b1; c_in = 1'b0;
51    #10  a = 1'b0; b = 1'b1; c_in = 1'b1;
52    #10  a = 1'b1; b = 1'b0; c_in = 1'b0;
53    #10  a = 1'b1; b = 1'b0; c_in = 1'b1;
54    #10  a = 1'b1; b = 1'b1; c_in = 1'b0;
55    #10  a = 1'b1; b = 1'b1; c_in = 1'b1;
56  end
57
58  initial
59  $monitor ($time, "ns, inputs = %b  outputs = %d" , {a, b, c_in}, {c_out, sum});
60
61
62  Full_Adder FA0(sum, c_out, a, b, c_in);
63
64  endmodule
65
```

Timing diagram of Full adder



Part (b). Implement the 4-bit adder by connecting pre-designed full adder modules. Submit the design codes screenshots with name signatures

Designing code

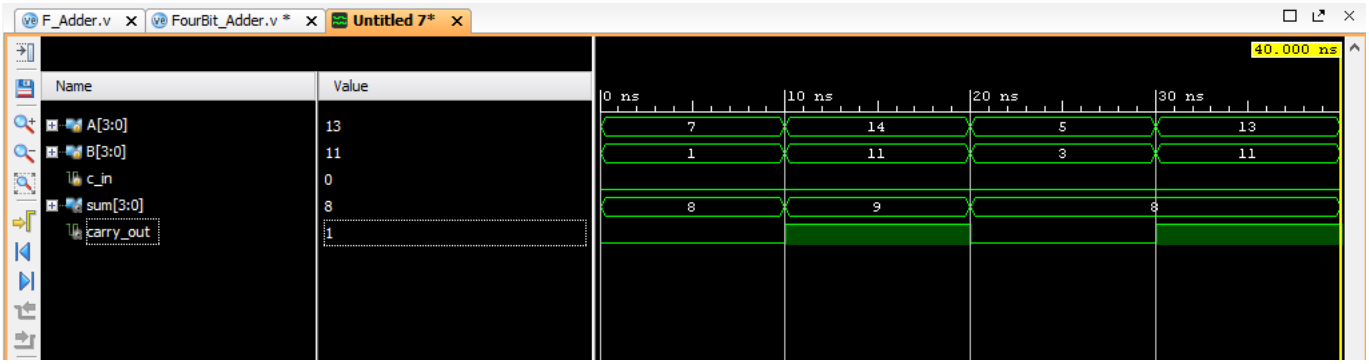
```
22  // Malak Majeedullah khan
23  // 033-17-0024
24
25  module FourBit_Adder(carry_out, sum, A, B, c_in);
26
27  input [3:0] A,B;
28  input c_in;
29  output [3:0] sum;
30  output carry_out;
31
32  //wires
33  wire c1,c2,c3;
34
35
36  //instantiating the full adder
37  Full_Adder FA0(c1, sum[0], A[0], B[0], c_in);
38  Full_Adder FA1(c2, sum[1], A[1], B[1], c1);
39  Full_Adder FA2(c3, sum[2], A[2], B[2], c2);
40  Full_Adder FA3(carry_out, sum[3], A[3], B[3], c3);
41
42  endmodule
43
```

c. Test the design using Verilog test bench. Submit the test bench codes with name signatures along with the tcl console outputs and timing diagrams.

Test bench

F:/8th semster/DSD/lab 01/Lab_01/Lab_01.srcs/sources_1/new/FourBit_Adder.v

```
46  module Four_BitAdder_TB();
47  reg [3:0] A, B;
48  reg c_in;
49  wire [3:0] sum;
50  wire carry_out;
51
52  //instantiate
53  FourBit_Adder AB0(carry_out, sum, A, B, c_in);
54
55  initial
56  begin
57      #00 A = 4'b0111; B = 4'b0001; c_in = 1'b0;
58      #10 A = 4'b1110; B = 4'b1011; c_in = 1'b0;
59      #10 A = 4'b0101; B = 4'b0011; c_in = 1'b0;
60      #10 A = 4'b1101; B = 4'b1011; c_in = 1'b0;
61      #10 $finish;
62  end
63  initial
64  begin
65  $display("Author: Malak Majeed Ullah khan\n");
66  $monitor($time, "ns, A = %d, B = %d, sum = %d", A, B, {carry_out, sum});
67  end
68
```



TCL console output

```
# run 1000ns
Author: Malak Majeed Ullah khan

                  0ns, A =  7, B =  1, sum =  8
                 10ns, A = 14, B = 11, sum = 25
                 20ns, A =  5, B =  3, sum =  8
                 30ns, A = 13, B = 11, sum = 24
$finish called at time : 40 ns : File "F:/8th semste
INFO: [USF-XSim-96] XSim completed. Design snapshot
```

2. Design, implement and investigate a 16-to-1 multiplexer by interconnecting five 4-to-1 multiplexers using hierarchical modelling approach.

a. Provide the design with schematic diagram.

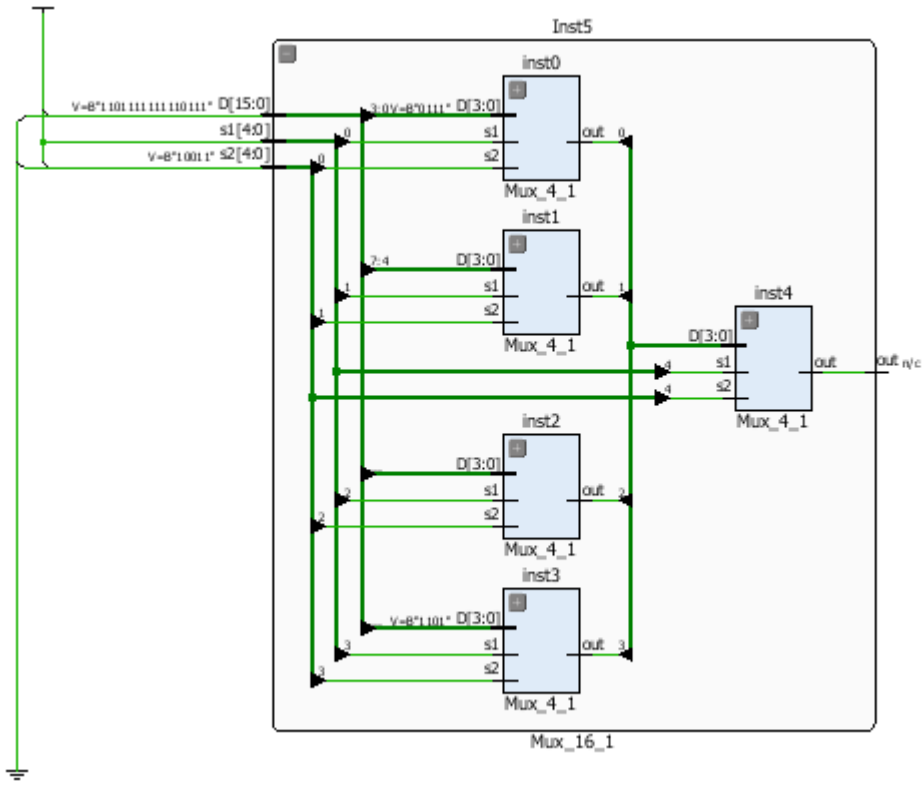
F:/8th semster/DSD/lab 01/Lab_01/Lab_01.srcs/sources_1/new/Mux_4_to_1.v

```
22
23  // Malak Majeed ullah khan
24  module Mux_4_1(out, s1, s2, D);
25
26  //defining ports
27  input [3:0] D; //4 input data
28  input s1,s2; // two selector switches
29  output out;   // output
30
31  //defining wires
32  wire w1,w2,w3,w4,w5,w6;
33
34  //gate level logic for mux
35  not not1(w1, s1);  // no gate
36  not not2(w2, s2);
37  and and1(w3, D[0], w1, w2);  // and gate logic
38  and and2(w4, D[1], w1, s2);
39  and and3(w5, D[2], s1, w2);
40  and and4(w6, D[3], s1, s2);
41
42  or or1(out, w3, w4, w5, w6); // output from the wires is given to out
43
44  endmodule // end of the module
45
```

16 Bit Multiplexer designing code

```
4
5   // Malak Majeedullah khan
6   module Mux_16_1(out, s1, s2, D);
7   //dimensions and sizes
8   input [15:0] D;   // input data 16 switches
9   input [4:0] s1, s2;   // two selector switches
10  output out;   // output data
11
12  // wires
13  wire [3:0]c;  //creating internal  wires
14
15  //instantiation of the mux
16  Mux_4_1 inst0(c[0], s1[0], s2[0], D[3:0] );   // mux 1
17  Mux_4_1 inst1(c[1], s1[1], s2[1], D[7:4] );   // mux 2
18  Mux_4_1 inst2(c[2], s1[2], s2[2], D[11:8]);   // mux 3
19  Mux_4_1 inst3(c[3], s1[3], s2[3], D[15:12]);  // mux 4
20
21  //Final 5th mux to combine all output
22  Mux_4_1 inst4(out, s1[4], s2[4], c);
23
24  endmodule   // end of design module
```

Schematic design

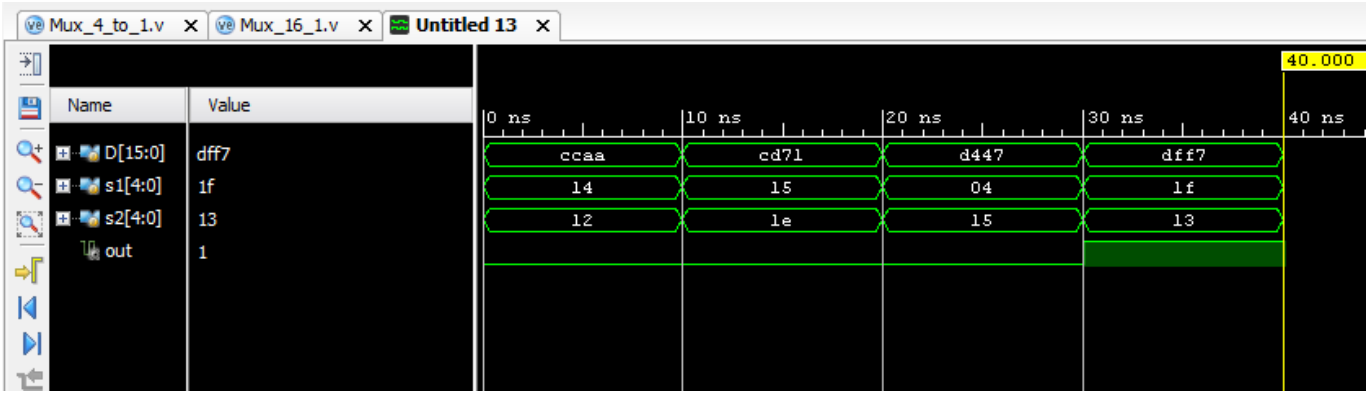


Test bench for 16 bit multiplexer

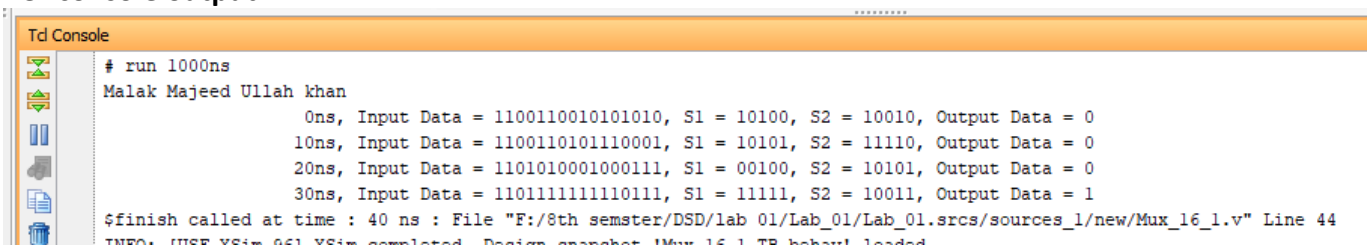
F:/8th semster/DSD/lab 01/Lab_01/Lab_01.srcs/sources_1/new/Mux_16_1.v

```
29 module Mux_16_1_TB();     // creating test bench
30 //defining ports
31 reg [15:0]D;  // data
32 reg [4:0] s1,s2;  // selector
33 wire out;      // output
34
35 //instantiation of top level model
36 Mux_16_1 Inst5 (out, s1, s2,D);
37 initial
38 begin   // assigning values to the data and switches
39     #00 D = 16'b1100110010101010; s1 = 5'b10100; s2 = 5'b10010;
40     #10 D = 16'b1100110101110001; s1 = 5'b10101; s2 = 5'b11110;
41     #10 D = 16'b1101010001000111; s1 = 5'b00100; s2 = 5'b10101;
42     #10 D = 16'b1101111111110111; s1 = 5'b11111; s2 = 5'b10011;
43     #10 $finish;
44 end
45
46 initial
47 begin
48 $display("Malak Majeed Ullah khan");    // display command to display text in console
49 // monitoring the output in console
50 $monitor($time,"ns, Input Data = %b, S1 = %b, S2 = %b, Output Data = %b ", D, s1, s2, out);
51 end
52 endmodule  // end of test bench module
```

Part (c). Test the design using both test bench simulation. Submit the test bench codes with tcl console and
timing diagrams.
Timing diagram



TCL console output



Output on FPGA board