EX.NO: 1 DESIGN AND SIMULATION OF 8 bit Full Adder and 4 bit Multiplier

DATE: 24/01/2022

AIM:

To verify the functionality and timing of your design or portion of your design. To interpret Verilog code into circuit functionality and displays logical results of the described HDL to determine correct circuit operation and to create and verify complex functions in a relatively small amount of time.

TOOLS REQUIRED:

1. Xilinx Vivado Design Suite: WebEdition

PROCEDURE:

1. Start by clicking **Vivado Design Suite: WebEdition**

2. Go to **File → new project → Enter project name**, select the top level source as **HDL** & click **next**.

3. Enter **device properties** as

Product Category : General Purpose
Family : Kintex 7
Device : xc7k70t
Package :fbg484
Speed : -1
Top Level Source Type : HDL
Synthesis Tool : XST (VHDL/Verilog)
Simulator : ISim (VHDL/Verilog)
Preferred Language : Verilog

& **Click next**.

4. **Right click** the device name (XCS3540) in the source window

to create **new source**.

5. Select **Verilog module** and enter **file name** in the new source window & click **next**.

6. Write the **Verilog code** in the **Verilog editor window**.

7. Write the **testbench** by create **new source simulation source.**

8. Run Check syntax through **Process window→ synthesize→** double click **Behavioral Check Syntax→** and removes error if present, with proper syntax & coding.

9. Click on the symbol of FPGA device and then right click→ click on **new source**.

10. Select the desired parameters for simulating the design. In this case **combinational circuit** and **simulation time** click **finish**.

11. Assign **all input signal** (**high or low**) using just **click** on this and save file.

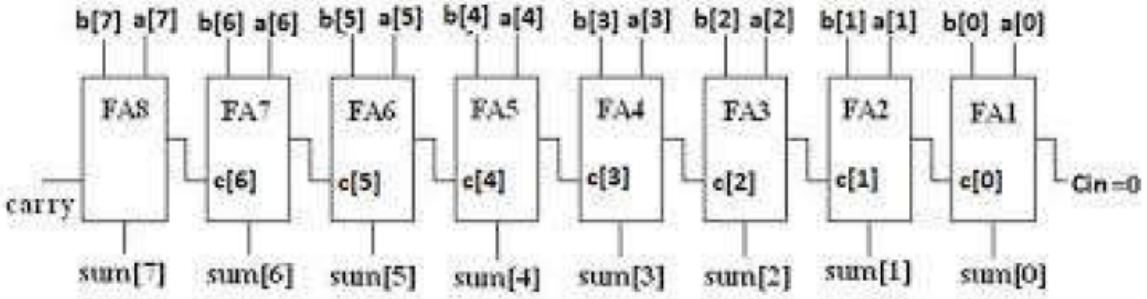
12. From the **source process window**. Click **Behavioral simulation**

from drop-down menu.

13. Double click the **Simulation Behavioral Model**.

14. Verify your design in **wave window** by seeing behavior of output signal with respect to input signal.

8 bit Full Adder:



Truth table for 2 bit Full adder:

Truth Table

Input			Output	
A	B	Cin	Sum	Carry
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Expression

$$\text{Sum} = A \oplus B \oplus Cin$$

$$\text{Cout} = AB + BCin + CinA$$

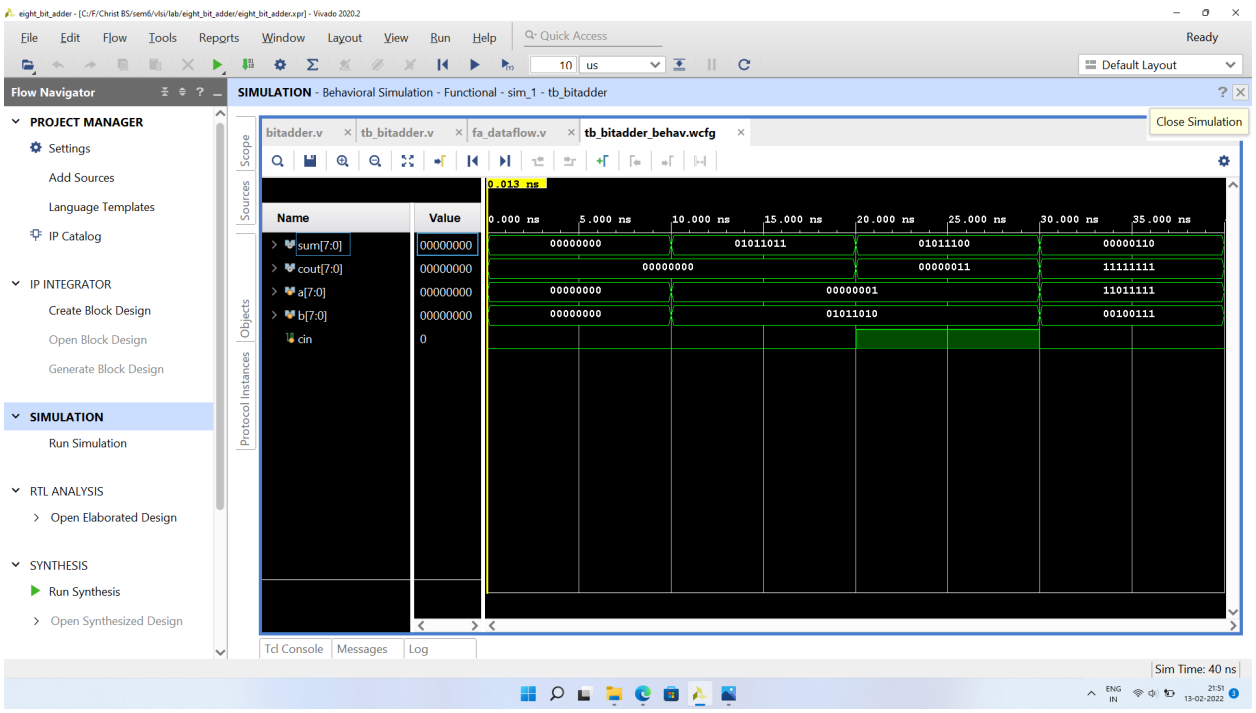
PROGRAM

```
22  //1960628 Prem 8 Bit Adder Dataflow
23  module bitadder(
24      input [7:0]a,
25      input [7:0]b,
26      input cin,
27      output [7:0]sum,
28      output [7:0]cout
29      );
30  fa_dataflow I1(a[0],b[0],cin,sum[0],cout[0]);
31  fa_dataflow I2(a[1],b[1],cout[0],sum[1],cout[1]);
32  fa_dataflow I3(a[2],b[2],cout[1],sum[2],cout[2]);
33  fa_dataflow I4(a[3],b[3],cout[2],sum[3],cout[3]);
34  fa_dataflow I5(a[4],b[4],cout[3],sum[4],cout[4]);
35  fa_dataflow I6(a[5],b[5],cout[4],sum[5],cout[5]);
36  fa_dataflow I7(a[6],b[6],cout[5],sum[6],cout[6]);
37  fa_dataflow I8(a[7],b[7],cout[6],sum[7],cout[7]);
38
39
40
41  endmodule
42
```

TESTBENCH

```
22  //1960628 Prem Testbench 8 Bit Adder
23  module tb_bitadder();
24  wire [7:0]sum;
25  wire [7:0]cout;
26  reg [7:0]a;
27  reg [7:0]b;
28  reg cin;
29  bitadder I1(a,b,cin,sum,cout);
30  initial
31  begin
32  a = 8'b00000000;
33  b = 8'b00000000;
34  cin = 1'b0;
35  #10
36  a = 8'b00000001;
37  b = 8'b01011010;
38  cin = 1'b0;
39  #10
40  a = 8'b00000001;
41  b = 8'b01011010;
42  cin = 1'b1;
43  #10
44  a = 8'b11011111;
45  b = 8'b00100111;
46  cin = 1'b0;
47  #10
48  $finish();
49  end
50  endmodule
51
```

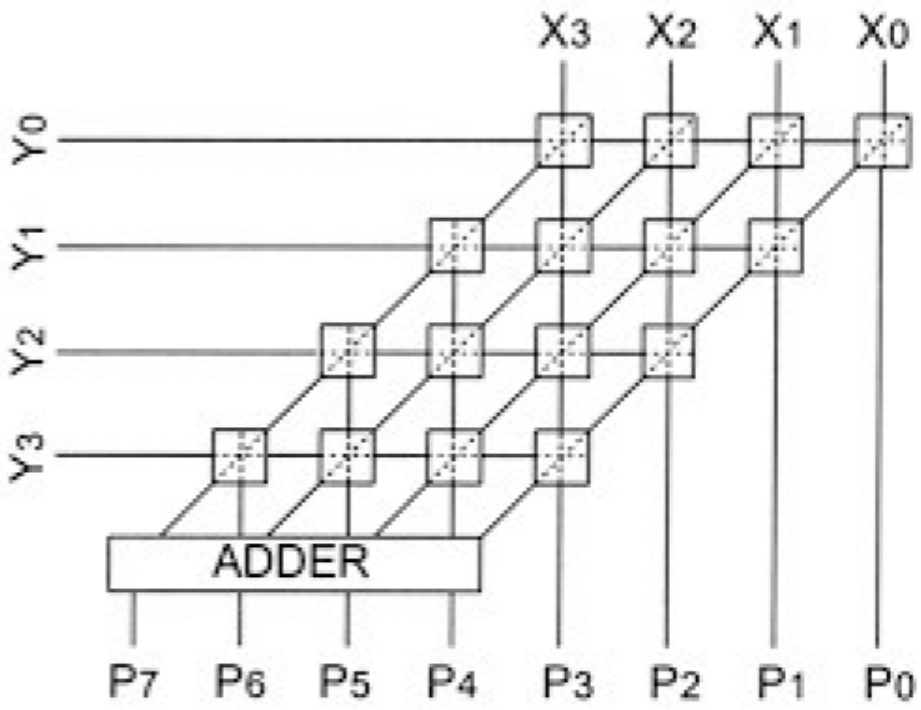
OUTPUT



Result:

Thus the Verilog code was scripted, simulated and waveforms were verified using Xilinx Vivado Webpack.

4 bit Multiplier:

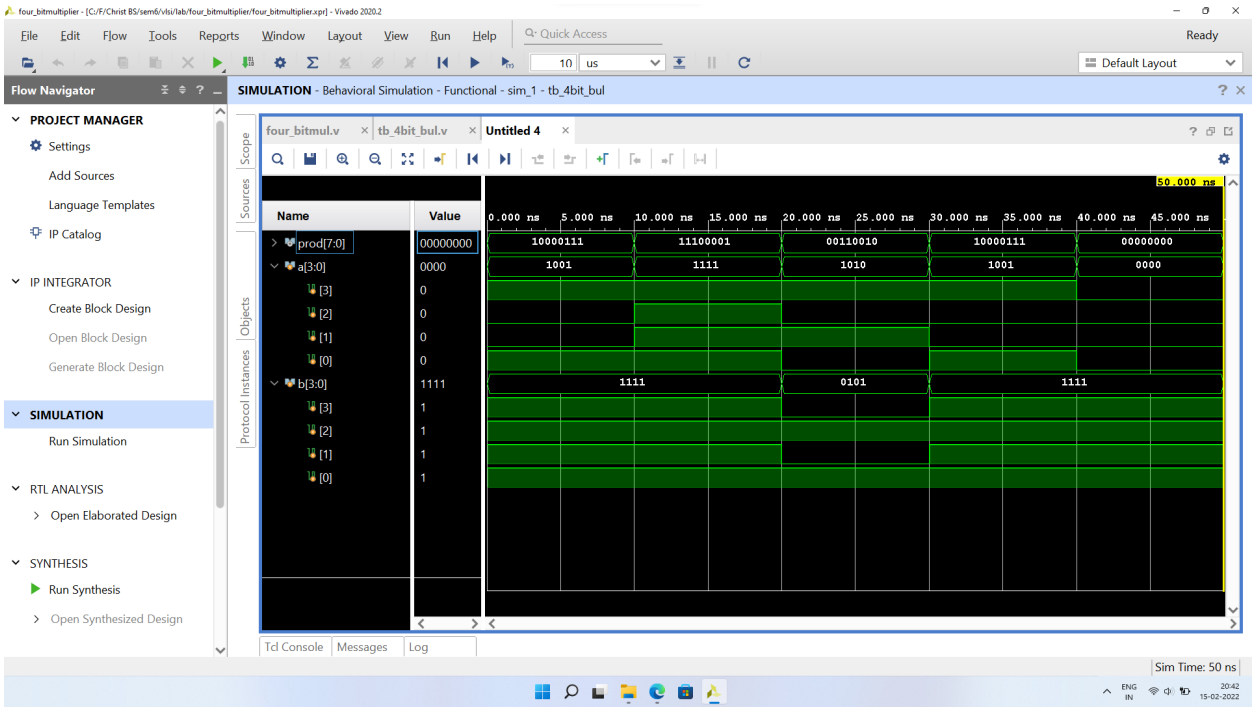


Program:

```
21
22      //1960628 Prem 4 bit multiplier
23      module four_bitmul(
24          output [7:0]prod,
25          input [3:0] a,b
26          );
27      wire [7:0] p0,p1,p2,p3;
28      wire [7:0] sum1,sum2,sum3;
29
30      assign p0 = {4{a[0]}}&b[3:0];
31      assign p1 = {4{a[1]}}&b[3:0];
32      assign p2 = {4{a[2]}}&b[3:0];
33      assign p3 = {4{a[3]}}&b[3:0];
34
35      assign sum1 = p0 + (p1<<1);
36      assign sum2 = sum1 + (p2<<2);
37      assign sum3 = sum2 + (p3<<3);
38      assign prod = sum3;
39      endmodule
40
```

Test Bench:

```
22      //1960628 Prem 4 bit multiplier testbench
23      module tb_4bit_bul();
24      wire [7:0] prod;
25      reg [3:0] a,b;
26
27      four_bitmul I1(prod,a,b);
28      initial
29      begin
30      a = 4'b1001;
31      b = 4'b1111;
32      #10
33      a = 4'b1111;
34      b = 4'b1111;
35      #10
36      a = 4'b1010;
37      b = 4'b0101;
38      #10
39      a = 4'b1001;
40      b = 4'b1111;
41      #10
42      a = 4'b0000;
43      b = 4'b1111;
44      #10
45      $finish();
46      end
47      endmodule
```



Result:

Thus the Verilog code was scripted, simulated and waveforms were verified using Xilinx Vivado Webpack.