CS2610 Computer Organization Laboratory Lab - 3

Name : Parmar Jignesh Jayantibhai
Roll no : 112101035

Objective:

In this lab you will analyse the efficiency of multiplier topologies in terms of processing delay and power consumption.

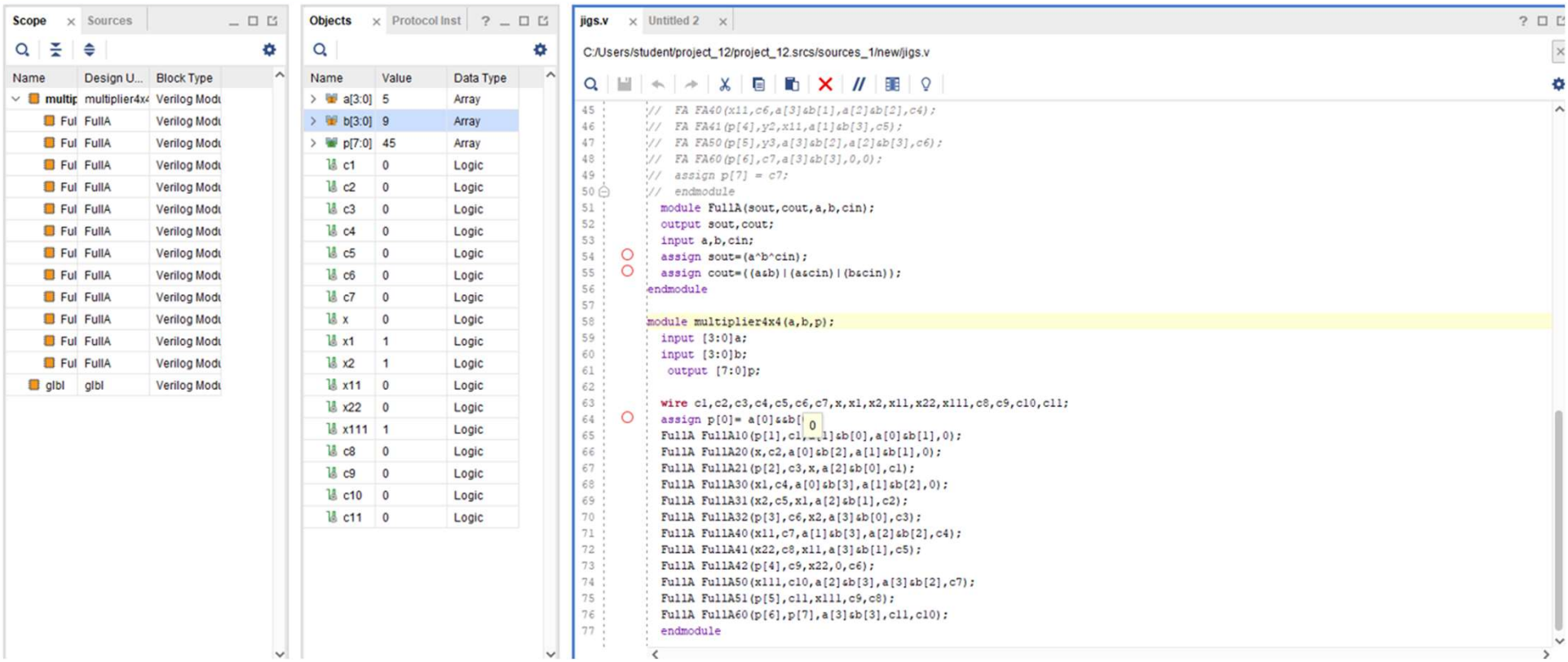
Problem:

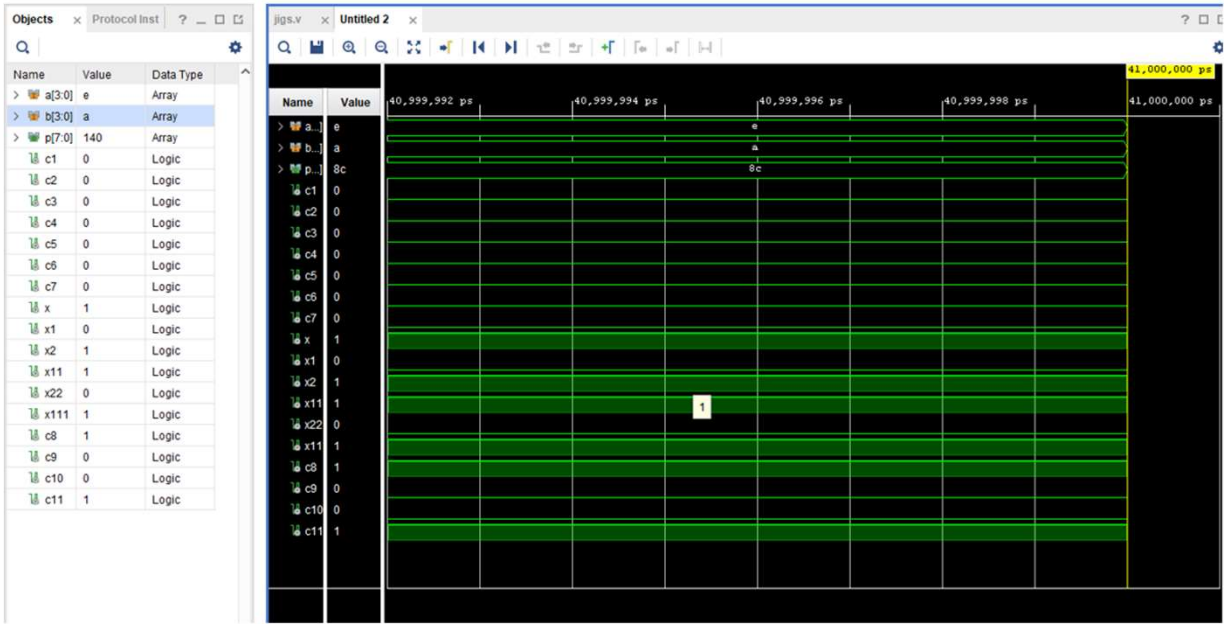
You are given two 4 bit numbers A and B in two 4 bit registers.In Verilog, implement the following:

1) Implement an array multiplier that would multiply two unsigned numbers A and B.

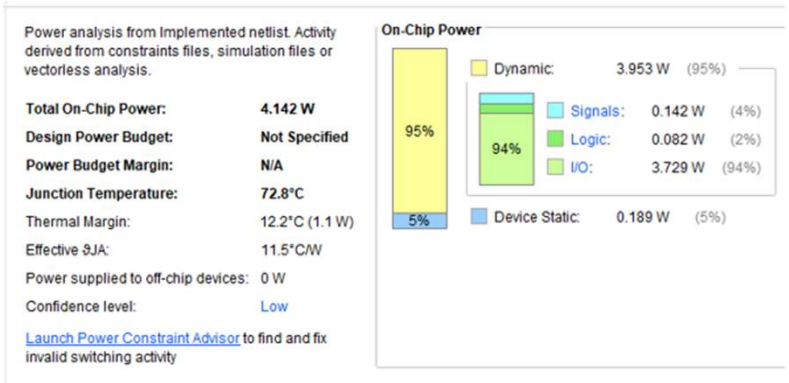
Solution::

Code::

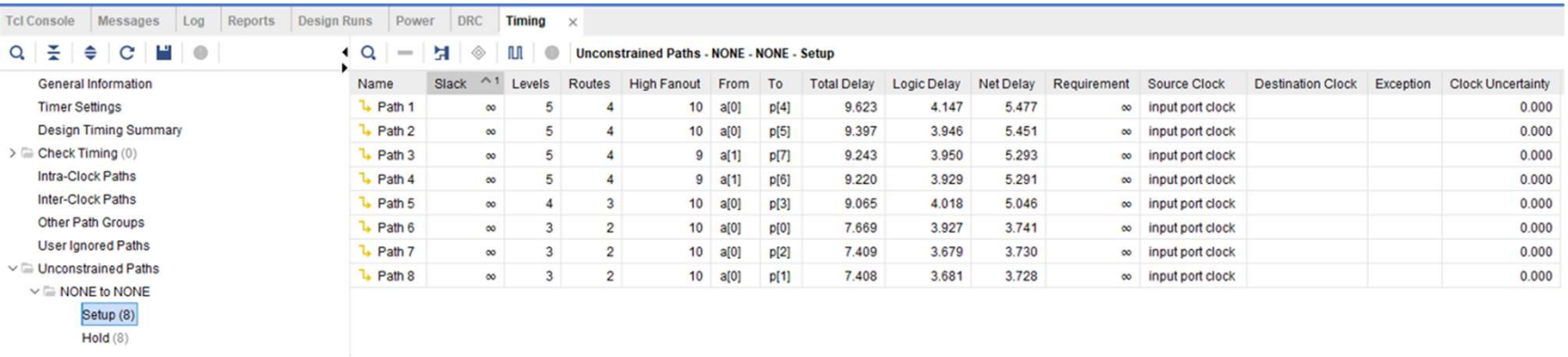

Scope
Sources
Name
Design U...
Block Type
multip
multiplier4x4
Verilog Modu
Ful
FullA
Verilog Modu
glbl
glbl
Verilog Modu
Objects
Protocol Inst
Name
Value
Data Type
a[3:0]
5
Array
b[3:0]
9
Array
p[7:0]
45
Array
c1
c2
c3
c4
c5
c6
c7
x
x1
x2
x11
x22
x111
c8
c9
c10
c11
0
Logic
1
jigs.v
Untitled 2
C:/Users/student/project_12/project_12.srcs/sources_1/new/jigs.v
// FA FA40(x11,c6,a[3]&b[1],a[2]&b[2],c4);
// FA FA41(p[4],y2,x11,a[1]&b[3],c5);
// FA FA50(p[5],y3,a[3]&b[2],a[2]&b[3],c6);
// FA FA60(p[6],c7,a[3]&b[3],0,0);
// assign p[7] = c7;
// endmodule
module FullA(sout,cout,a,b,cin);
output sout,cout;
input a,b,cin;
assign sout=(a^b^cin);
assign cout=((a&b)|(a&cin)|(b&cin));
endmodule
module multiplier4x4(a,b,p);
input [3:0]a;
input [3:0]b;
output [7:0]p;
wire c1,c2,c3,c4,c5,c6,c7,x,x1,x2,x11,x22,x111,c8,c9,c10,c11;
assign p[0]= a[0]&&b[
0
FullA FullA10(p[1],c1,a[1]&b[0],a[0]&b[1],0);
FullA FullA20(x,c2,a[0]&b[2],a[1]&b[1],0);
FullA FullA21(p[2],c3,x,a[2]&b[0],c1);
FullA FullA30(x1,c4,a[0]&b[3],a[1]&b[2],0);
FullA FullA31(x2,c5,x1,a[2]&b[1],c2);
FullA FullA32(p[3],c6,x2,a[3]&b[0],c3);
FullA FullA40(x11,c7,a[1]&b[3],a[2]&b[2],c4);
FullA FullA41(x22,c8,x11,a[3]&b[1],c5);
FullA FullA42(p[4],c9,x22,0,c6);
FullA FullA50(x111,c10,a[2]&b[3],a[3]&b[2],c7);
FullA FullA51(p[5],c11,x111,c9,c8);
FullA FullA60(p[6],p[7],a[3]&b[3],c11,c10);
endmodule



Power consumption::



Timing delay::

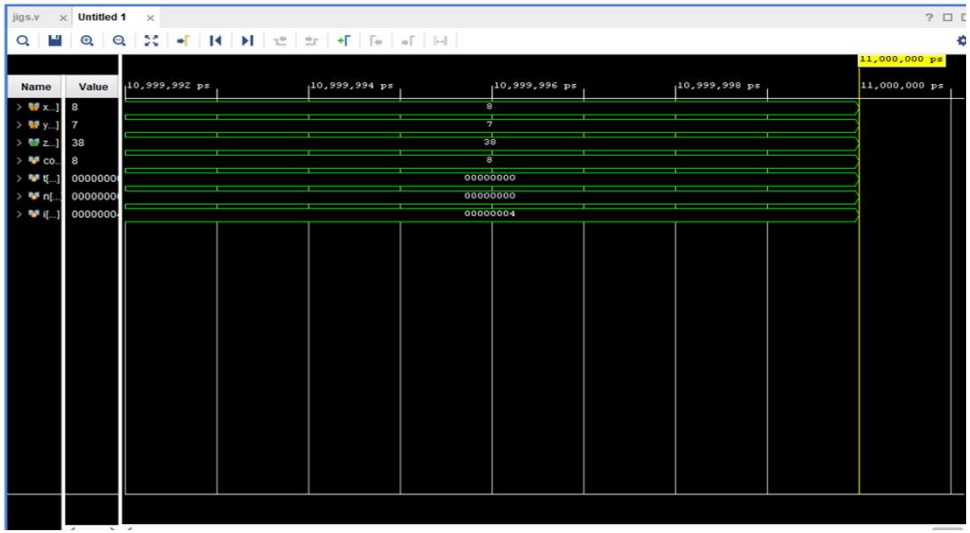


2) Implement Booth's algorithm that would multiply two two's complement numbers A and B.
Solution::
Code::

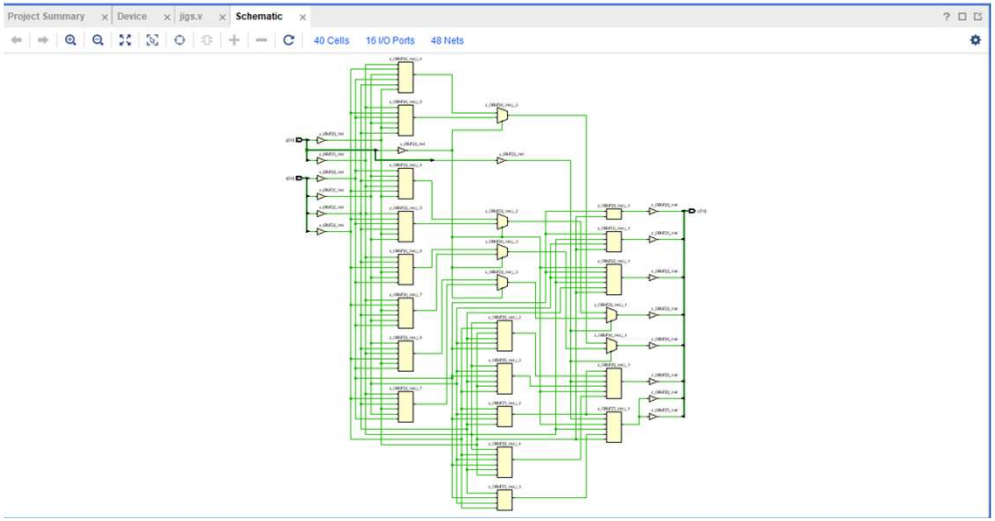
```
module booth_multi(x,y,z);
   input [3:0]x,y;
   output reg [7:0]z;
   reg [3:0] comp;
   integer t,n;
   integer i=0;
   always @(*) begin
   z=8'b0;
   z[3:0]=z[3:0]+y;
   comp=-x;
   for (i=0;i<4;i=i+1) begin
      if(i==0) begin
      if(y[0]==0) begin
            t=z[0];
            n=z[7];
             z=z>>>1;
             z[7]=n;
         end
      else begin
            z[7:4]=z[7:4]+comp;
             t=z[0];
              n=z[7];
               z=z>>>1;
                z[7]=n;
          end
      end
      if(i>0) begin
          if(z[0]-t==0)begin
              t=z[0];
              n=z[7];
              z=z>>>1;
              z[7]=n;
end
else if(z[0]-t==1)begin
          z[7:4]=z[7:4]+comp;

          t=z[0];
          n=z[7];
          z=z>>>1;
          z[7]=n;
          end
else if(z[0]-t==-1)begin
                    z[7:4]=z[7:4]+x;
                    t=z[0];
                    n=z[7];
                    z=z>>>1;
                    z[7]=n;
          end
      end
      end
    end
endmodule
```

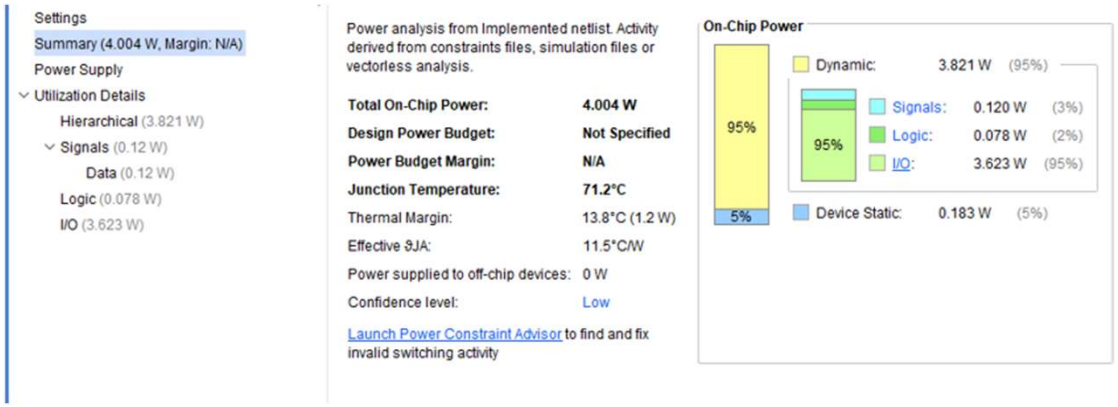
Wave diagram::



Schematic figure::



Power consuptin::



Timing diagram::

Name	Slack ^1	Levels	Routes	High Fanout	From	To	Total Delay	Logic Delay	Net Delay	Requirement	Source Clock	Destination Clock	Exception	Clock Uncertainty
Path 1	∞	4	3	14	x[2]	z[5]	8.658	4.044	4.615	∞	input port clock			0.000
Path 2	∞	4	3	14	x[2]	z[7]	8.254	4.059	4.195	∞	input port clock			0.000
Path 3	∞	4	3	14	x[2]	z[6]	8.185	4.039	4.146	∞	input port clock			0.000
Path 4	∞	5	2	14	x[2]	z[4]	8.018	4.192	3.826	∞	input port clock			0.000
Path 5	∞	5	2	13	x[3]	z[3]	7.958	4.188	3.770	∞	input port clock			0.000
Path 6	∞	3	2	15	y[0]	z[0]	7.543	3.958	3.585	∞	input port clock			0.000
Path 7	∞	3	2	15	y[0]	z[1]	7.348	3.711	3.637	∞	input port clock			0.000
Path 8	∞	3	2	15	y[0]	z[2]	7.236	3.710	3.526	∞	input port clock			0.000

Conclusion::-

The booth's algorithm of multiplication consumed less power as compared to the array multiplication of two 4 bit numbers A and B. Also the booth's algorithm of multiplication has total time delay lesser than the array multiplication of two 4 bit numbers A and B. hence , we can conclude booth's algorithm for multiplication is more efficient compared to the array multiplication.