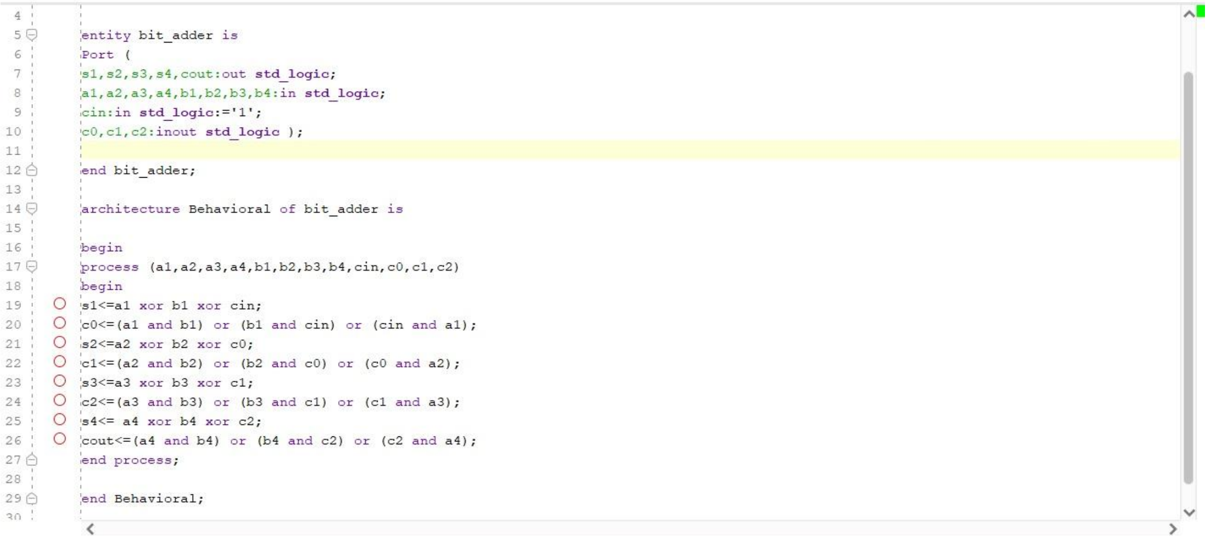
Experiment 8

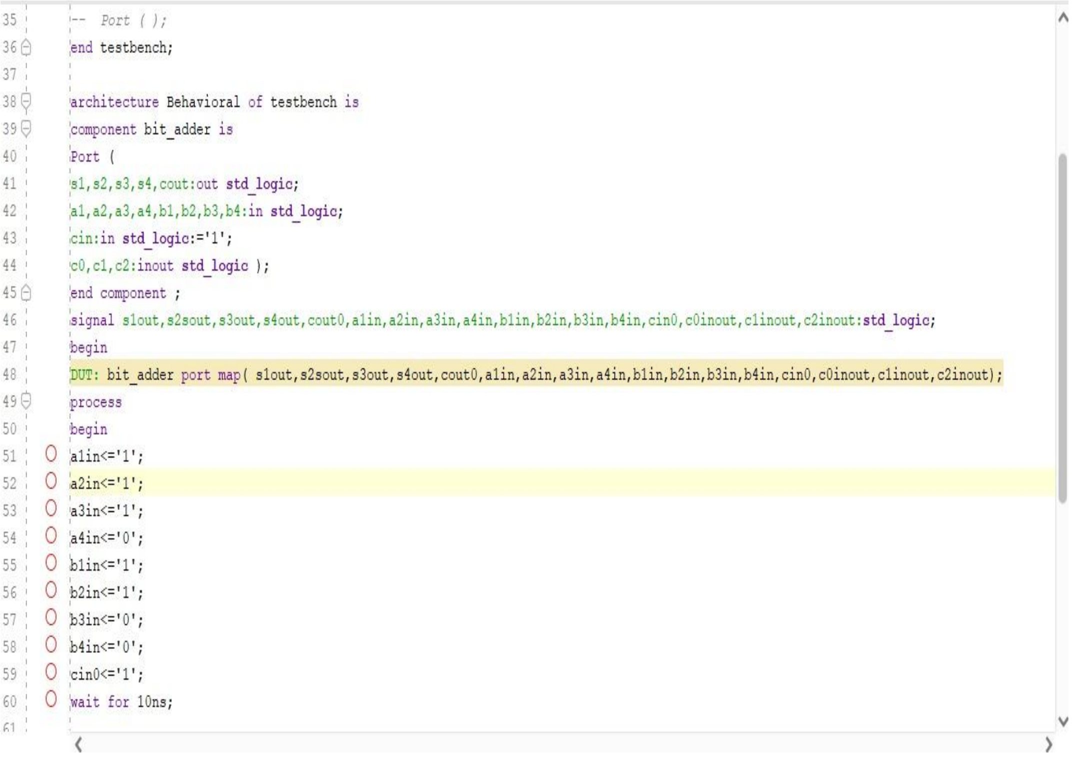
Implementing 4-bit adder

Design code: Structural



```
entity bit_adder is
Port (
s1,s2,s3,s4,cout:out std_logic;
a1,a2,a3,a4,b1,b2,b3,b4:in std_logic;
cin:in std_logic:='1';
c0,c1,c2:inout std_logic );

end bit_adder;

architecture Behavioral of bit_adder is

begin
process (a1,a2,a3,a4,b1,b2,b3,b4,cin,c0,c1,c2)
begin
s1<=a1 xor b1 xor cin;
c0<=(a1 and b1) or (b1 and cin) or (cin and a1);
s2<=a2 xor b2 xor c0;
c1<=(a2 and b2) or (b2 and c0) or (c0 and a2);
s3<=a3 xor b3 xor c1;
c2<=(a3 and b3) or (b3 and c1) or (c1 and a3);
s4<= a4 xor b4 xor c2;
cout<=(a4 and b4) or (b4 and c2) or (c2 and a4);
end process;

end Behavioral;
```

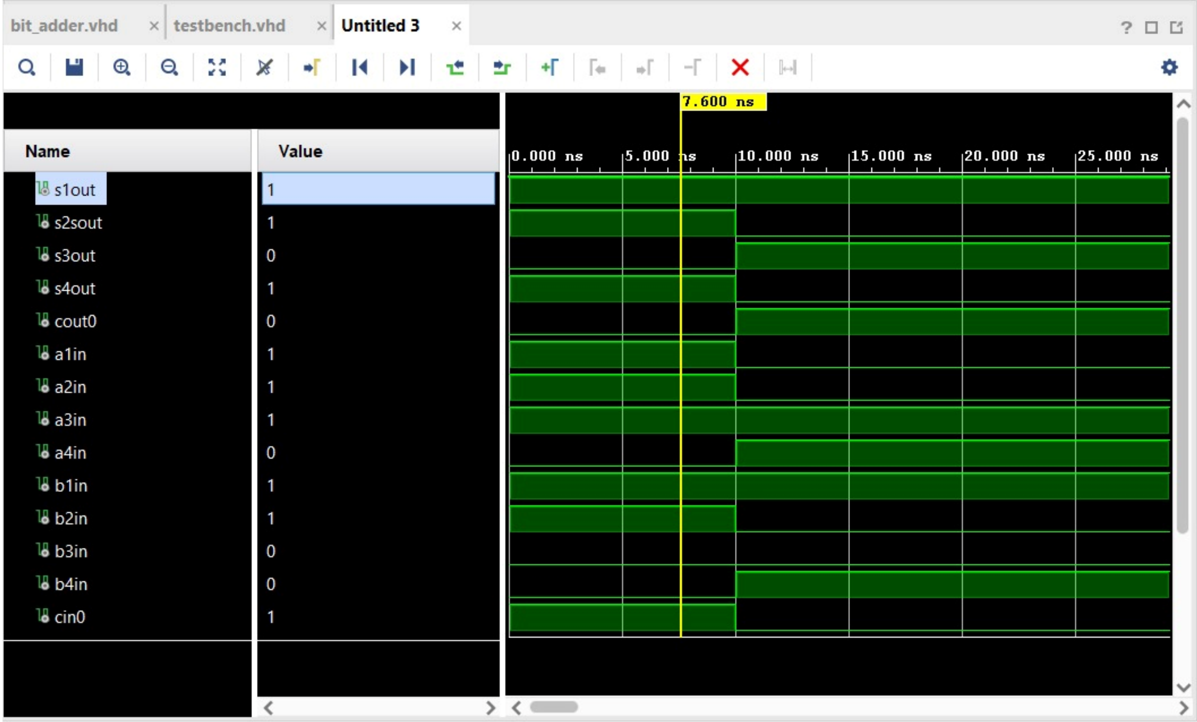
Testbench:



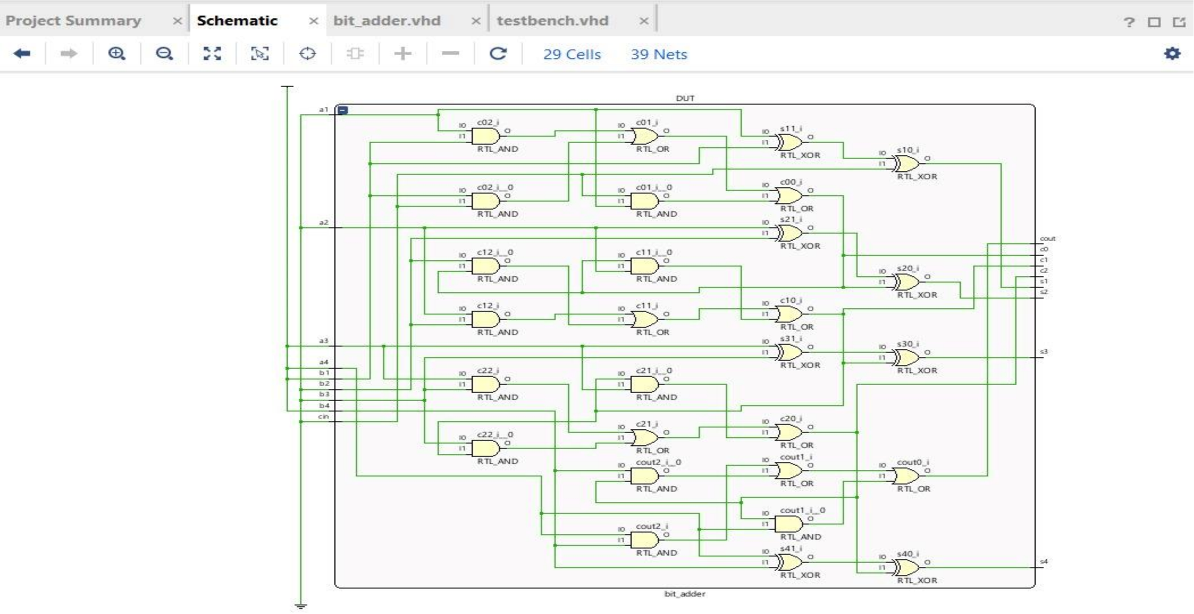
```
-- Port ( );
end testbench;

architecture Behavioral of testbench is
component bit_adder is
Port (
s1,s2,s3,s4,cout:out std_logic;
a1,a2,a3,a4,b1,b2,b3,b4:in std_logic;
cin:in std_logic:='1';
c0,c1,c2:inout std_logic );
end component ;
signal s1out,s2sout,s3out,s4out,cout0,a1in,a2in,a3in,a4in,b1in,b2in,b3in,b4in,cin0,c0inout,c1inout,c2inout:std_logic;
begin
DUT: bit_adder port map( s1out,s2sout,s3out,s4out,cout0,a1in,a2in,a3in,a4in,b1in,b2in,b3in,b4in,cin0,c0inout,c1inout,c2inout);
process
begin
a1in<='1';
a2in<='1';
a3in<='1';
a4in<='0';
b1in<='1';
b2in<='1';
b3in<='0';
b4in<='0';
cin0<='1';
wait for 10ns;
```

Debarghya Debnath E22ECEU0002
Experiment 8

Simulation:



Name	Value
s1out	1
s2sout	1
s3out	0
s4out	1
cout0	0
a1in	1
a2in	1
a3in	1
a4in	0
b1in	1
b2in	1
b3in	0
b4in	0
cin0	1



Result:- 4-bit adder has been implemented successfully, and their test results are shown above.