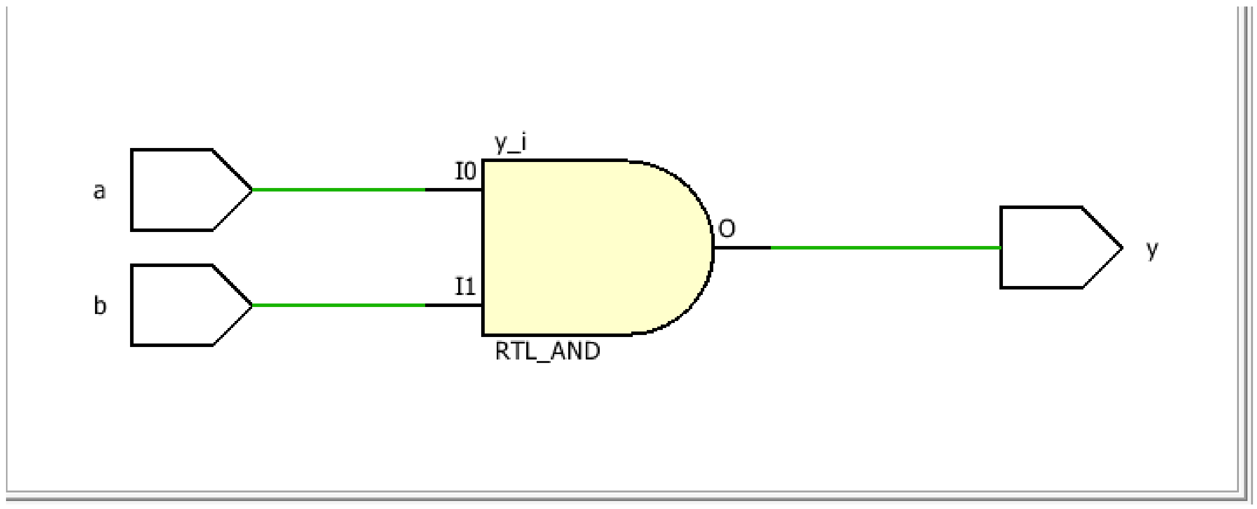
DAY – 1

IMPLEMENTATION OF AND GATE IN DATAFLOW MODEL.

VHDL CODE :

```
entity and_df is
   Port ( a : in STD_LOGIC;
          b : in STD_LOGIC;
          y : out STD_LOGIC);
end and_df;

architecture Dataflow of and_df is

begin
y <= a and b;

end Dataflow;
```

SCHEMATIC:

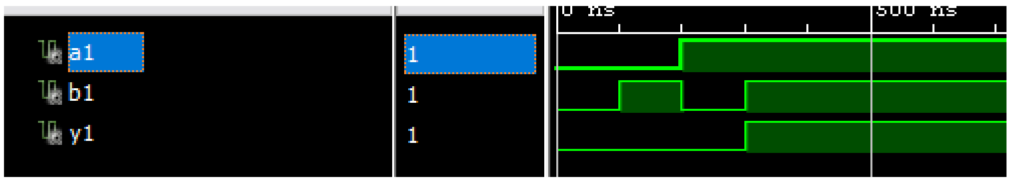


TBW CODE:

```
entity and_df_tb is

--  Port ( );

end and_df_tb;


architecture Behavioral of and_df_tb is

component and_df is

   Port ( a : in STD_LOGIC;

          b : in STD_LOGIC;

          y : out STD_LOGIC);

end component;

signal a1:std_logic:='0';

signal b1:std_logic:='0';

signal y1:std_logic;


begin

uut:and_df port map(a=>a1,b=>b1,y=>y1);

stim_proc:process

begin

wait for 100 ns;

a1<='0';

b1<='1';

wait for 100 ns;

a1<='1';

b1<='0';
```

```
wait for 100 ns;

a1<='1';

b1<='1';

wait;

end process;


end Behavioral;
```

WAVEFORM:



IMPLEMENTATION OF OR GATE IN DATAFLOW MODEL.

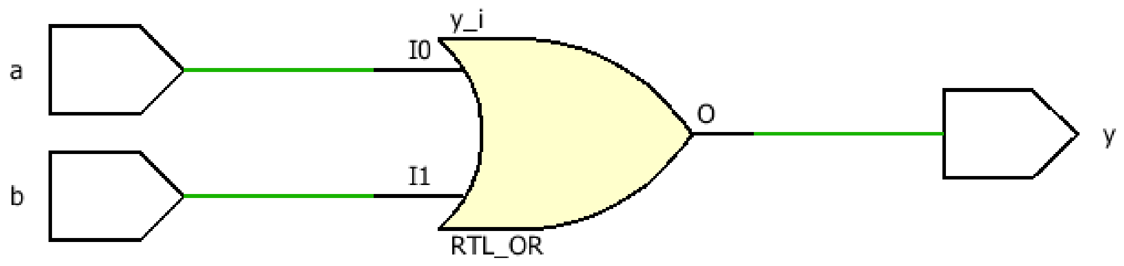
VHDL CODE:

```
entity or_df is

   Port ( a : in STD_LOGIC;

          b : in STD_LOGIC;

          y : out STD_LOGIC);

end or_df;


architecture Dataflow of or_df is


begin
```

y<= a or b;

end Dataflow;

SCHEMATIC:



TBW CODE :

entity or_df_tb is

-- Port ( );

end or_df_tb;

architecture Behavioral of or_df_tb is

component or_df is

   Port (a : in STD_LOGIC;

         b : in STD_LOGIC;

         y : out STD_LOGIC);

end component;

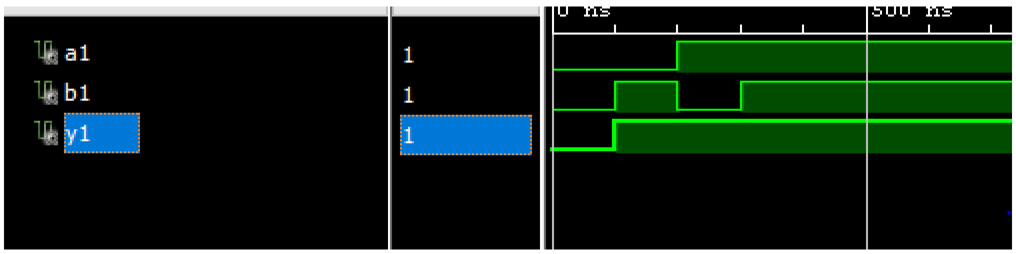
signal a1: std_logic := '0';

signal b1: std_logic := '0';

signal y1: std_logic;

```
begin
uut:or_df port map (a=>a1,b=>b1,y=>y1);
stim_proc:process
begin
wait for 100 ns;
a1<='0';
b1<='1';
wait for 100 ns;
a1<='1';
b1<='0';
wait for 100 ns;
a1<='1';
b1<='1';
wait;
end process;
end Behavioral;
```

WAVEFORM:

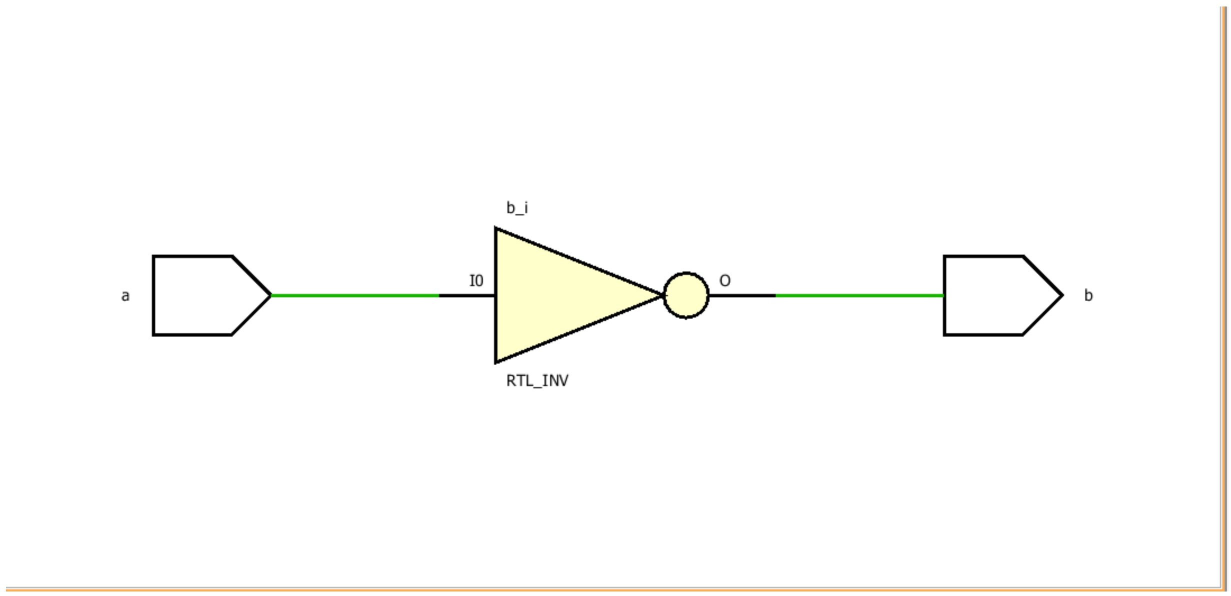


IMPLEMENTATION OF NOT GATE IN DATAFLOW MODEL

VHDL CODE:

```
entity NOTGATE_df is
   Port ( a : in STD_LOGIC;
          b: out STD_LOGIC);
end NOTGATE_df;


architecture Dataflow of NOTGATE_df is
begin
b<= not a;
end Dataflow;
```

SCHEMATIC DIAGRAM :

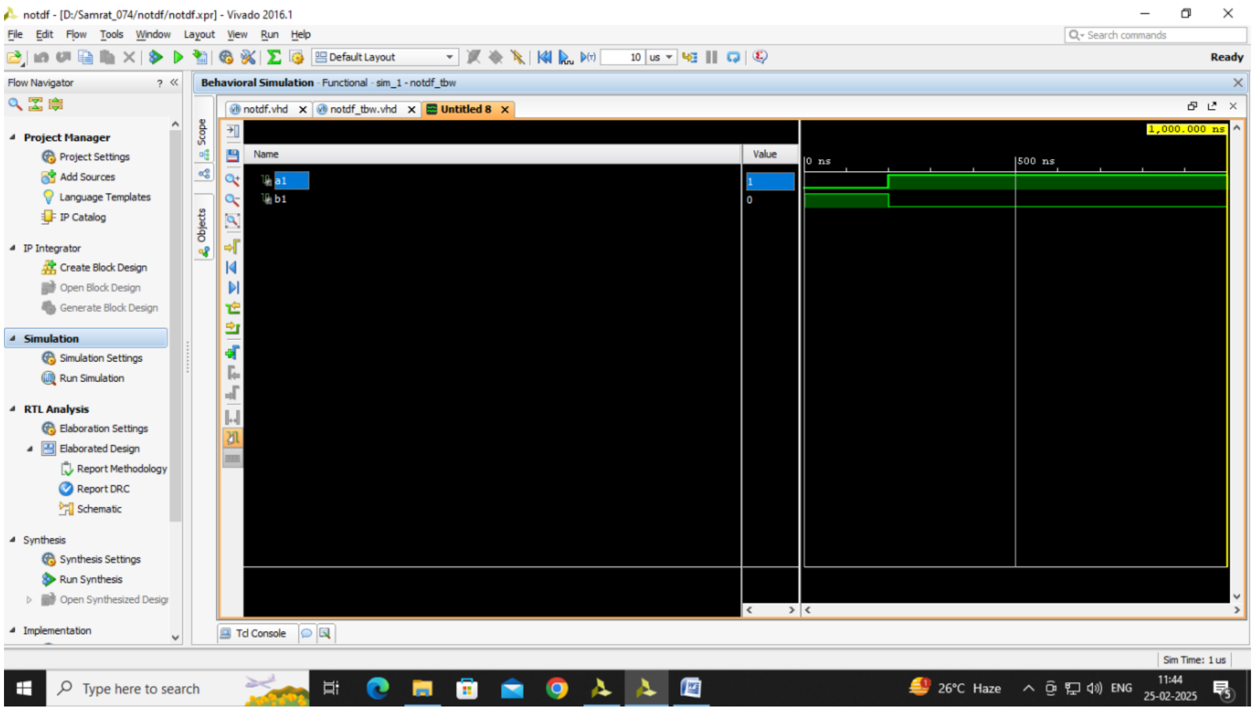


TBW CODE :

```
entity NOTGATE_tbw is
--  Port ( );
end NOTGATE_tbw;
```

```
architecture Dataflow of NOTGATE_df is


component NOTGATE_df is
   Port ( a : in STD_LOGIC;
          b : out STD_LOGIC);
end component;
signal a1:STD_LOGIC='0';
signal b1:STD_LOGIC='0';
begin
uut:and_df port map(a=>a1,b=>b1);
stim_proc:process
begin
wait for 100 ns;
a1<='1';
wait ;
end process;
end Behavioral;
```

SIMULATION:


notdf - [D:/Samrat_074/notdf/notdf.xpr] - Vivado 2016.1
Behavioral Simulation - Functional - sim_1 - notdf_tbw
Name
Value
a1
1
b1
0
0 ns
500 ns
1,000.000 ns
Sim Time: 1 us

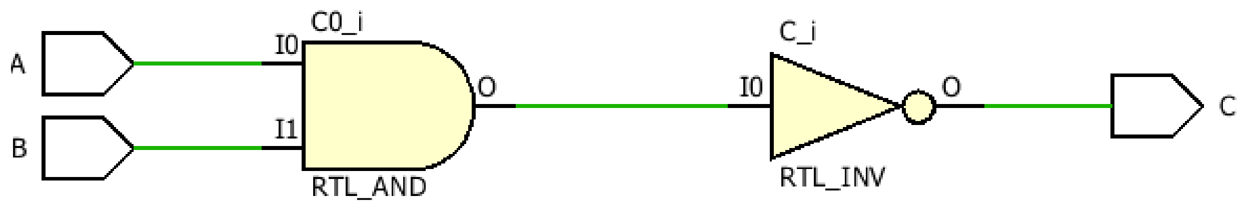
Nand GATE

VHD CODE

```
entity NAND_df is
   Port ( A : in STD_LOGIC;
          B : in STD_LOGIC;
          C : out STD_LOGIC);
end NAND_df;


architecture Dataflow of NAND_df is


begin
C<=A nand B;
```

end Dataflow;

Schematic Diagram



TEST BENCH CODE

entity NAND_df_TBW is

-- Port ( );

end NAND_df_TBW;

architecture Dataflow of NAND_df_TBW is

component NAND_df is

   Port (A : in STD_LOGIC;

          B : in STD_LOGIC;

          C : out STD_LOGIC);

end component;

signal a1:STD_LOGIC := '0';

signal B1:STD_LOGIC := '0';

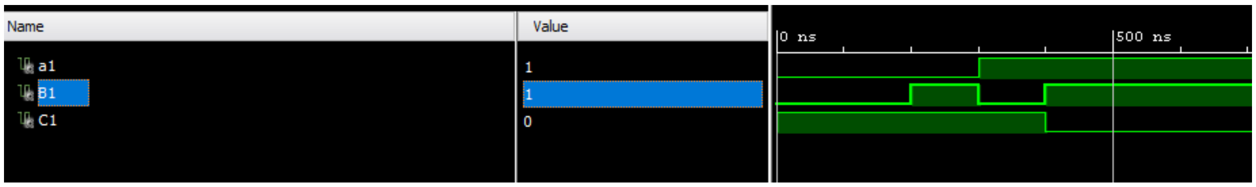
signal C1:STD_LOGIC;

begin

uut:NAND_df Port Map(A=>a1,B=>b1,C=>c1);

```
stim_proc:process

begin

wait for 100ns;

a1<='0';

b1<='0';

wait for 100ns;

a1<='0';

b1<='1';

wait for 100ns;

a1<='1';

b1<='0';

wait for 100ns;

a1<='1';

b1<='1';

wait;

end process;

end Dataflow;
```

TEST BENCH WAVEFORM

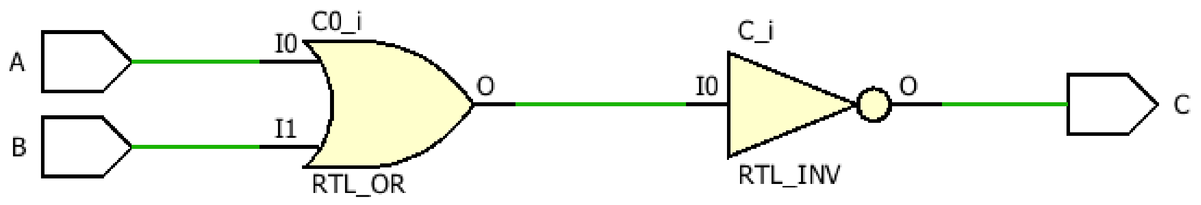


NOR GATE

```
entity NOR_df is
```

```
    Port ( A : in STD_LOGIC;
           B : in STD_LOGIC;
           C : out STD_LOGIC);
end NOR_df;

architecture Dataflow of NOR_df is

begin
C<=A nor B;

end Dataflow;
```

Schematic Diagram



TEST BENCH CODE

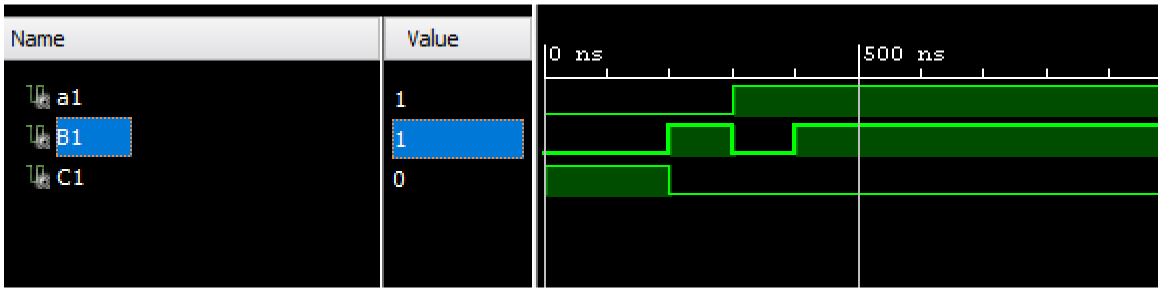
```
entity NOR_df_TBW is
--  Port ( );
end NOR_df_TBW;

architecture Dataflow of NOR_df_TBW is
component NOR_df is
```

```
    Port ( A : in STD_LOGIC;

           B : in STD_LOGIC;

           C : out STD_LOGIC);

end component;

signal a1:STD_LOGIC :='0';

signal B1:STD_LOGIC :='0';

signal C1:STD_LOGIC;

begin

uut:NOR_df Port Map(A=>a1,B=>b1,C=>c1);

stim_process:process

begin

wait for 100 ns;

a1<='0';

b1<='0';


wait for 100 ns;

a1<='0';

b1<='1';

wait for 100 ns;

a1<='1';

b1<='0';

wait for 100 ns;

a1<='1';

b1<='1';

wait;
```

end process;

end Dataflow;

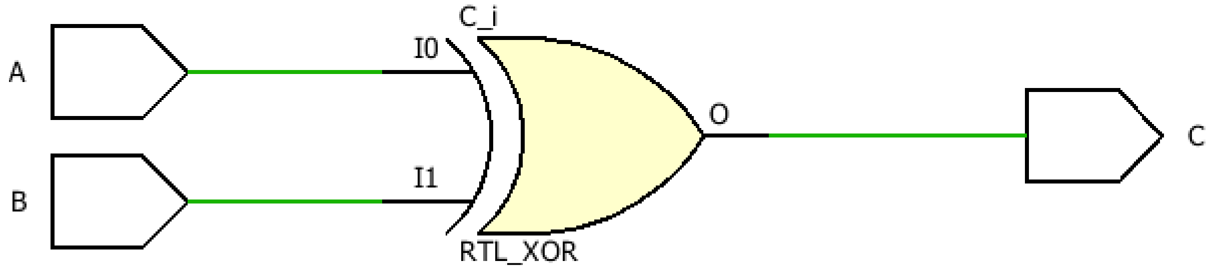
TEST BENCH WAVEFORM



XOR GATE

```
entity XOR_df is
    Port ( A : in STD_LOGIC;
           B : in STD_LOGIC;
           C : out STD_LOGIC);
end XOR_df;

architecture Dataflow of XOR_df is

begin
C<=A XOR B;

end Dataflow;
```

Schematic Diagram

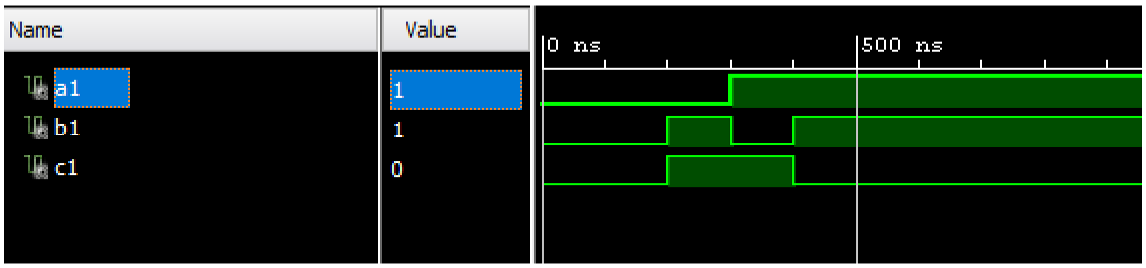


TEST BENCH CODE

```
entity XOR_df_TBW is

--  Port ( );

end XOR_df_TBW;


architecture Dataflow of XOR_df_TBW is

component XOR_df is

   Port ( A : in STD_LOGIC;

          B : in STD_LOGIC;

          C : out STD_LOGIC);

end component;

signal a1:STD_LOGIC :='0';

signal b1:STD_LOGIC :='0';

signal c1:STD_LOGIC;

begin

uut:XOR_df Port Map(A=>a1,B=>b1,C=>c1);

stim_proc:process

begin
```

```
wait for 100ns;

a1<='0';

b1<='0';

wait for 100ns;

a1<='0';

b1<='1';

wait for 100ns;

a1<='1';

b1<='0';

wait for 100ns;

a1<='1';

b1<='1';

wait;

end process;

end Dataflow;
```

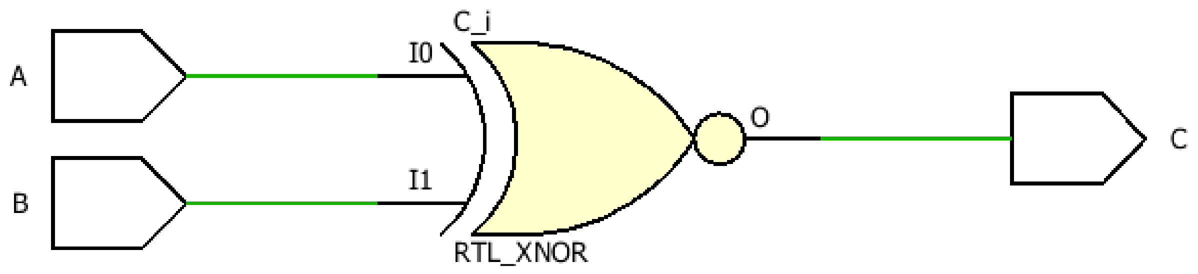
TEST BENCH WAVEFORM



XNOR GATE

```
entity XNOR_df is

   Port ( A : in STD_LOGIC;
        B : in STD_LOGIC;

        C : out STD_LOGIC);

end XNOR_df;


architecture Dataflow of XNOR_df is


begin

C<=A XNOR B;


end Dataflow;
```

Schematic Diagram



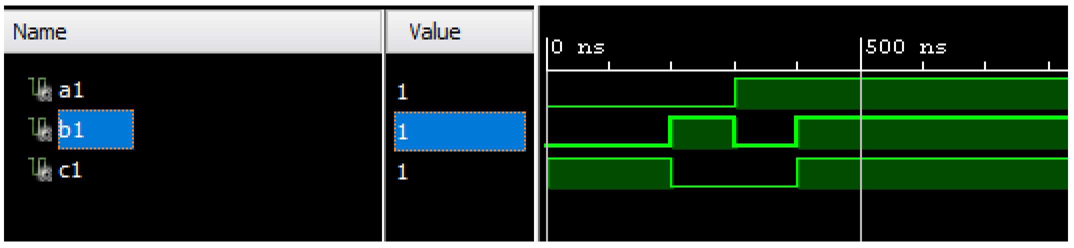
TEST BENCH CODE

```
entity XNOR_df_TBW is

--  Port ( );

end XNOR_df_TBW;


architecture Dataflow of XNOR_df_TBW is
```

```
component XNOR_df is

   Port ( A : in STD_LOGIC;

          B : in STD_LOGIC;

          C : out STD_LOGIC);

end component;

signal a1:STD_LOGIC :='0';

signal b1:STD_LOGIC :='0';

signal c1:STD_LOGIC;

begin

uut:XNOR_df Port Map(A=>a1,B=>b1,C=>c1);

stim_proc:process

begin

wait for 100ns;

a1<='0';

b1<='0';

wait for 100ns;

a1<='0';

b1<='1';

wait for 100ns;

a1<='1';

b1<='0';

wait for 100ns;

a1<='1';

b1<='1';

wait;
```

```
end process;

end Dataflow;
```

TEST BENCH WAVEFORM

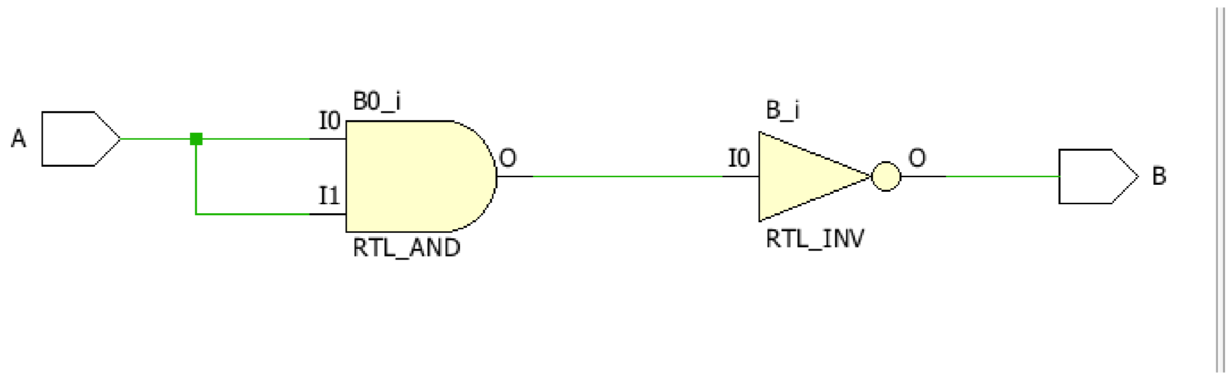


Day-2

Not using NAND

```
entity  NOT_NAND_df is

   Port ( A : in STD_LOGIC;

          B : out STD_LOGIC);

end NOT_NAND_df;


architecture Dataflow of NOT_NAND_df is


begin

B<=A nand A;


end Dataflow;
```

Schematic Diagram

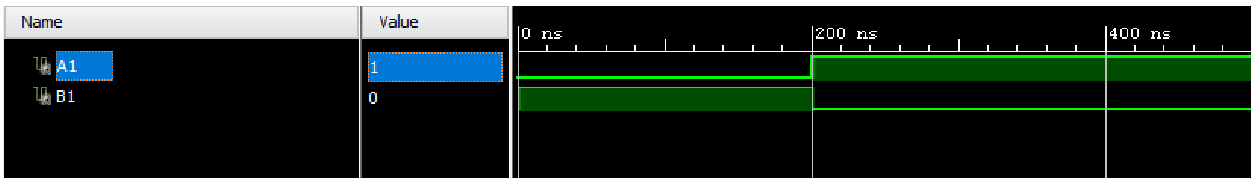


TEST BENCH CODE

```
entity NOT_NAND_df_TBW is

--  Port ( );

end NOT_NAND_df_TBW;


architecture Dataflow of NOT_NAND_df_TBW is

component NOT_NAND_df is

   Port ( A : in STD_LOGIC;

          B : out STD_LOGIC);

end component;

signal A1:STD_LOGIC := '0';

signal B1:STD_LOGIC;

begin

uut:NOT_NAND_df Port Map(A=>a1,B=>b1);

stim_proc:process

begin
```

```
wait for 100ns;

a1<='0';

wait for 100ns;

a1<='1';

wait;

end process;


end Dataflow;
```

TEST BENCH WAVEFORM

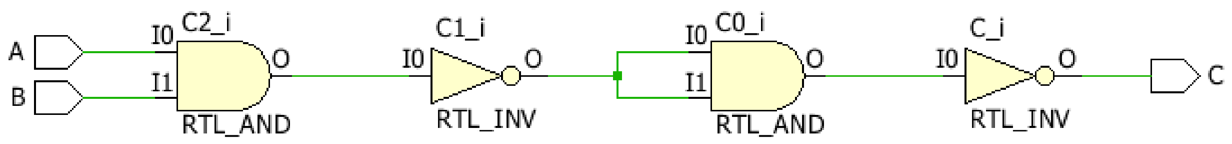


AND using NAND

```
entity AND_NAND_df is

   Port ( A : in STD_LOGIC;

          B : in STD_LOGIC;

          C : out STD_LOGIC);

end AND_NAND_df;


architecture Dataflow of AND_NAND_df is


begin

C<=(A NAND B) NAND (A NAND B);
```

end Dataflow;

Schematic Diagram

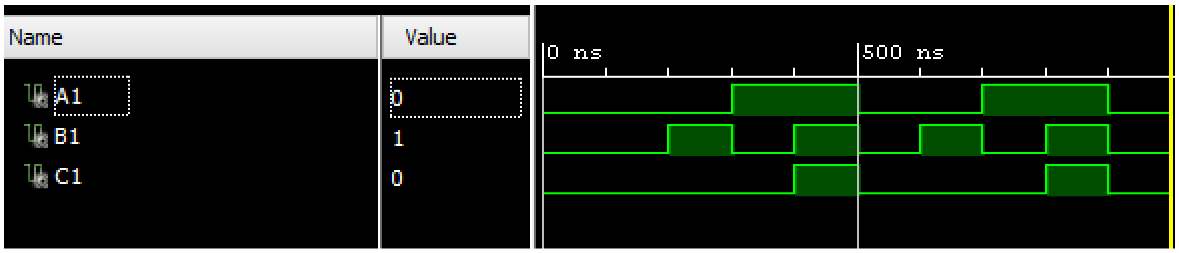


TEST BENCH CODE

```
entity AND_NAND_df_TBW is
--  Port ( );
end AND_NAND_df_TBW;


architecture Dataflow of AND_NAND_df_TBW is
component AND_NAND_df is
   Port ( A : in STD_LOGIC;
          B : in STD_LOGIC;
          C : out STD_LOGIC);
end component;
signal A1:STD_LOGIC := '0';
signal B1:STD_LOGIC := '0';
signal C1:STD_LOGIC;
begin
uut:AND_NAND_df Port Map(A=>a1,B=>b1,C=>c1);
```

```
stim_proc:process

begin

wait for 100ns;

a1<='0';

b1<='0';

wait for 100ns;

a1<='0';

b1<='1';

wait for 100ns;

a1<='1';

b1<='0';

wait for 100ns;

a1<='1';

b1<='1';

end process;



end Dataflow;
```

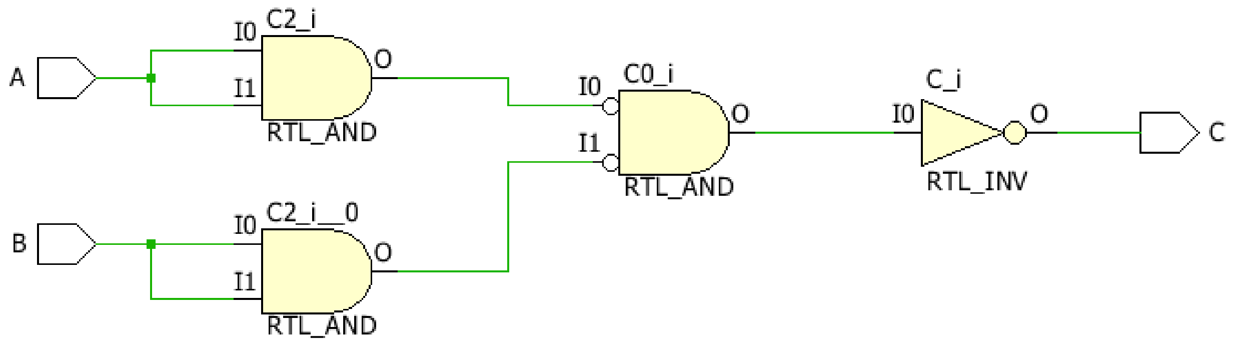
TEST BENCH WAVEFORM



OR using NAND

```
entity OR_NAND_df is

    Port ( A : in STD_LOGIC;

           B : in STD_LOGIC;

           C : out STD_LOGIC);

end OR_NAND_df;


architecture Dataflow of OR_NAND_df is


begin

C<=(A NAND A) NAND (B NAND B);


end Dataflow;
```

Schematic Diagram



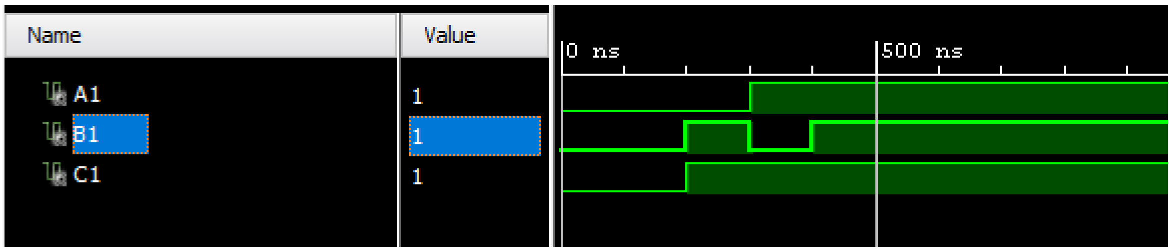
TEST BENCH CODE

```
entity OR_NAND_df_TBW is

--  Port ( );
```

```
end OR_NAND_df_TBW;


architecture Dataflow of OR_NAND_df_TBW is

component OR_NAND_df is

   Port ( A : in STD_LOGIC;

          B : in STD_LOGIC;

          C : out STD_LOGIC);

end component;

signal A1:STD_LOGIC := '0';

signal B1:STD_LOGIC := '0';

signal C1:STD_LOGIC;

begin

uut:OR_NAND_df Port Map(A=>a1,B=>b1,C=>c1);

stim_proc:process

begin

wait for 100ns;

a1<='0';

b1<='0';

wait for 100ns;

a1<='0';

b1<='1';

wait for 100ns;

a1<='1';

b1<='0';

wait for 100ns;
```

```
a1<='1';

b1<='1';

wait;

end process;

end Dataflow;
```

TEST BENCH WAVEFORM



Name	Value
A1	1
B1	1
C1	1

XOR using NAND

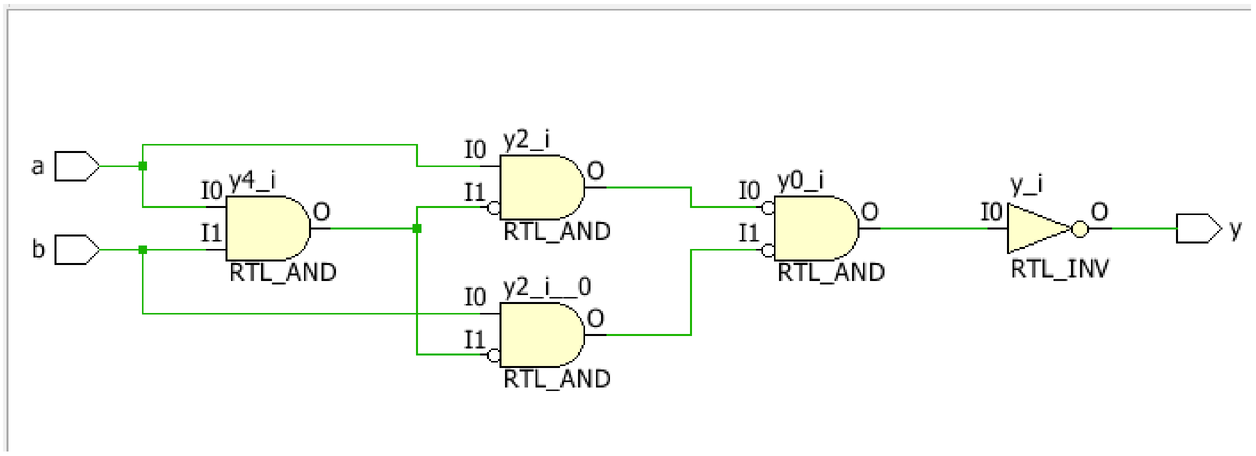
Vhdl code:

```
entity xor_nand is

   Port ( a : in STD_LOGIC;

          b : in STD_LOGIC;

          y : out STD_LOGIC);

end xor_nand;


architecture Behavioral of xor_nand is


begin
```

y<= (a nand (a nand b)) nand (b nand(a nand b));

end Behavioral;

Schematic Diagram:



Tbw code:

```
entity xor_nand_tbw is
--  Port ( );
end xor_nand_tbw;


architecture Behavioral of xor_nand_tbw is
component xor_nand is
   Port ( a : in STD_LOGIC;
          b : in STD_LOGIC;
          y : out STD_LOGIC);
end component;
```

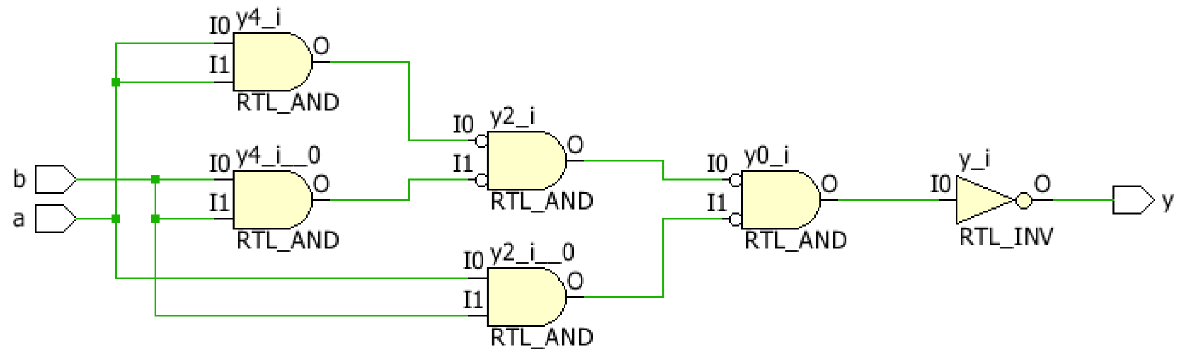
```
signal a1:std_logic:='0';

signal b1:std_logic:='0';

signal y1:std_logic;


begin

uut:xor_nand port map(a=>a1,b=>b1,y=>y1);

stim_proc:process

begin

wait for 100 ns;

a1<='0';

b1<='1';

wait for 100 ns;

a1<='1';

b1<='0';

wait for 100 ns;

a1<='1';

b1<='1';

wait;

end process;


end Behavioral;
```

XNOR using NAND

Vhdl code:

```
entity xnor_nand is
   Port ( a : in STD_LOGIC;
          b : in STD_LOGIC;
          y : out STD_LOGIC);
end xnor_nand;

architecture Behavioral of xnor_nand is

begin
y<= ((a nand a) nand (b nand b)) nand (a nand b);

end Behavioral;
```

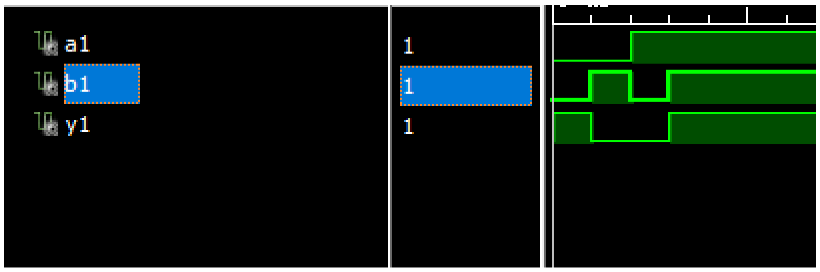
schematic Diagram:



Tbw code:

```
entity xnor_nand_tbw is

--  Port ( );

end xnor_nand_tbw;


architecture Behavioral of xnor_nand_tbw is




component xnor_nand is

   Port ( a : in STD_LOGIC;

          b : in STD_LOGIC;

          y : out STD_LOGIC);

end component;


signal a1:std_logic:='0';
signal b1:std_logic:='0';

signal y1:std_logic;


begin

uut:xnor_nand port map(a=>a1,b=>b1,y=>y1);

stim_proc:process

begin

wait for 100 ns;

a1<='0';

b1<='1';

wait for 100 ns;

a1<='1';

b1<='0';

wait for 100 ns;

a1<='1';

b1<='1';

wait;

end process;


end Behavioral;
```

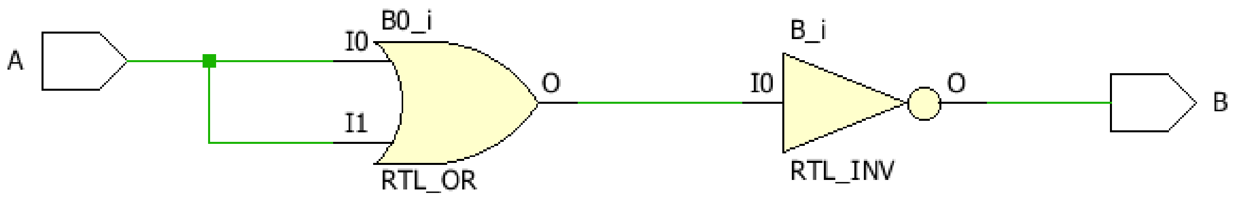
waveform:



NOT using NOR

```
entity NOT_NOR_df is
    Port ( A : in STD_LOGIC;
           B : out STD_LOGIC);
end NOT_NOR_df;


architecture Dataflow of NOT_NOR_df is


begin
B<=A NOR A;


end Dataflow;
```

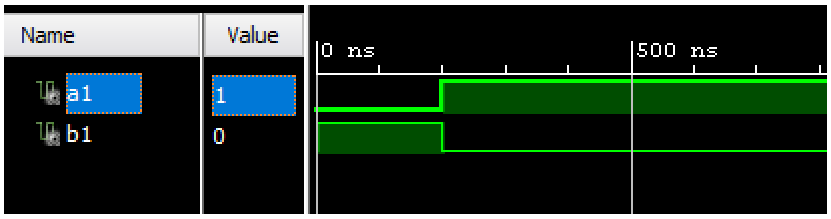
Schematic Diagram



TEST BENCH CODE

```
entity NOT_NOR_df_TBW is
--  Port ( );
end NOT_NOR_df_TBW;

architecture Dataflow of NOT_NOR_df_TBW is
component NOT_NOR_df is
   Port ( A : in STD_LOGIC;
          B : out STD_LOGIC);
end component;
signal a1:STD_LOGIC := '0';
signal b1:STD_LOGIC;
begin
uut:NOT_NOR_df Port Map(A=>a1,B=>b1);
stim_proc:process
begin
wait for 100ns;
a1<='0';
wait for 100ns;
a1<='1';
wait;
end process;


end Dataflow;
```

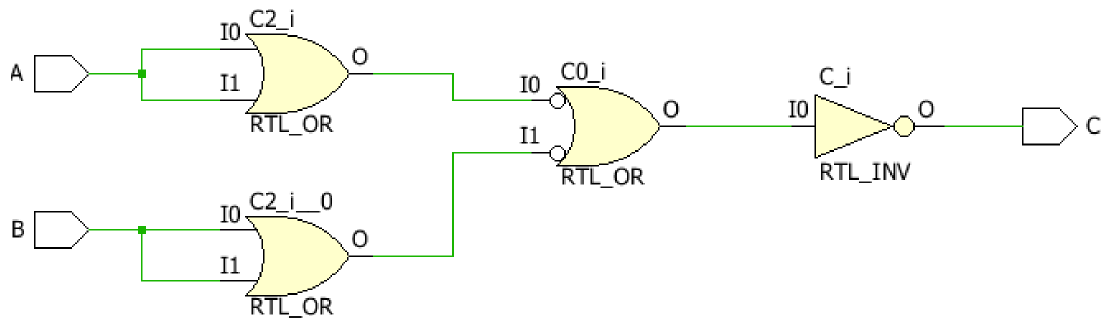
TEST BENCH WAVEFORM



AND using NOR

```
entity AND_NOR_df is

   Port ( A : in STD_LOGIC;

          B : in STD_LOGIC;

          C : out STD_LOGIC);

end AND_NOR_df;


architecture Dataflow of AND_NOR_df is


begin

C<=(A NOR A) NOR (B NOR B);


end Dataflow;
```

Schematic Diagram

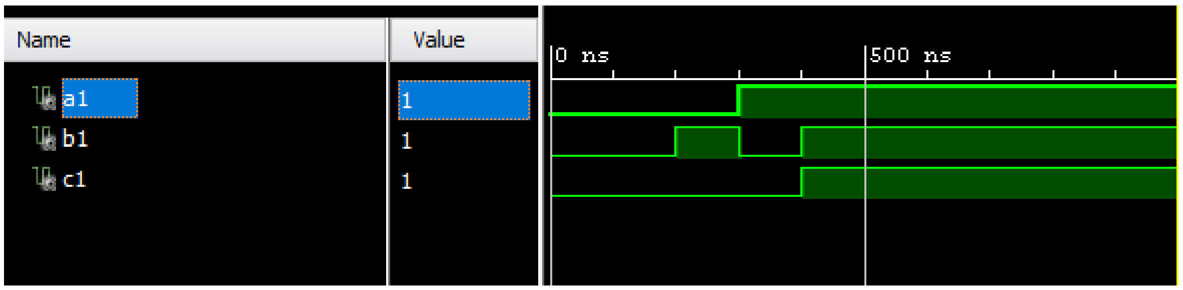


TEST BENCH CODE

```
entity AND_NOR_df_TBW is
--  Port ( );
end AND_NOR_df_TBW;


architecture Dataflow of AND_NOR_df_TBW is
component AND_NOR_df is
    Port ( A : in STD_LOGIC;
           B : in STD_LOGIC;
           C : out STD_LOGIC);
end component;
signal a1:STD_LOGIC :='0';
signal b1:STD_LOGIC :='0';
signal c1:STD_LOGIC;
begin
uut:AND_NOR_df Port Map(A=>a1,B=>b1,C=>c1);
stim_proc:process
begin
wait for 100ns;
a1<='0';
```

```
b1<='0';
wait for 100ns;
a1<='0';
b1<='1';
wait for 100ns;
a1<='1';
b1<='0';
wait for 100ns;
a1<='1';
b1<='1';
wait;
end process;


end Dataflow;
```

TEST BENCH WAVEFORM

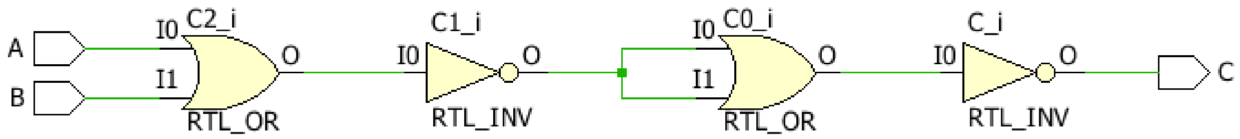


Name	Value
a1	1
b1	1
c1	1

OR using NOR

```
entity OR_NOR_df is
   Port ( A : in STD_LOGIC;
          B : in STD_LOGIC;

          C : out STD_LOGIC);

end OR_NOR_df;


architecture Dataflow of OR_NOR_df is


begin

C<=(A NOR B) NOR (A NOR B);


end Dataflow;
```

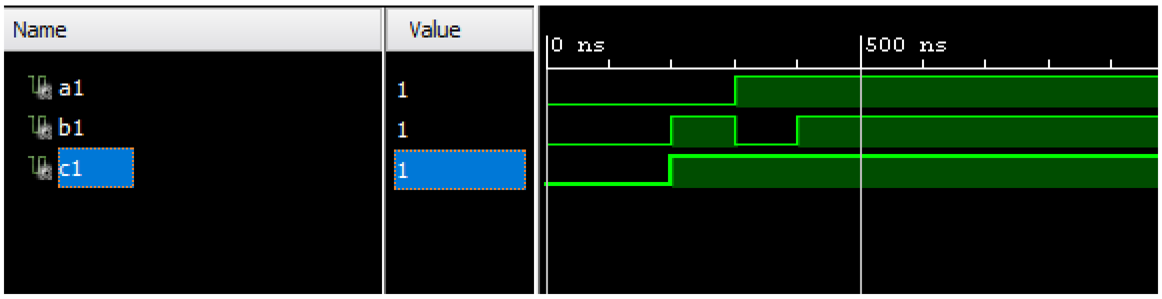
Schematic Diagram



TEST BENCH CODE

```
entity OR_NOR_df_TBW is

--  Port ( );

end OR_NOR_df_TBW;


architecture Dataflow of OR_NOR_df_TBW is

component OR_NOR_df is

   Port ( A : in STD_LOGIC;
        B : in STD_LOGIC;

        C : out STD_LOGIC);

end component;

signal a1:STD_LOGIC :='0';

signal b1:STD_LOGIC :='0';

signal c1:STD_LOGIC;

begin

uut:OR_NOR_df Port Map(A=>a1,B=>B1,C=>c1);

stim_proc:process

begin

wait for 100ns;

a1<='0';

b1<='0';

wait for 100ns;

a1<='0';

b1<='1';

wait for 100ns;

a1<='1';

b1<='0';wait for 100ns;

a1<='1';

b1<='1';

wait;

end process;



end Dataflow;
```

TEST BENCH WAVEFORM



OR using NOR

entity AND_NOR_df is

    Port (A : in STD_LOGIC;

           B : in STD_LOGIC;

           C : out STD_LOGIC);

end AND_NOR_df;

architecture Dataflow of AND_NOR_df is

begin

C<=(A NOR A) NOR (B NOR B);

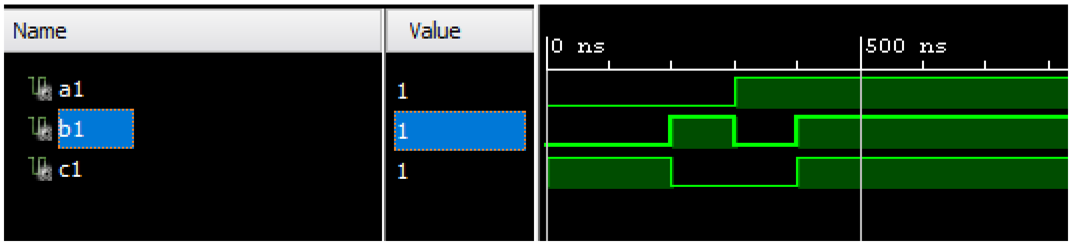
end Dataflow;

tbw code:

```
architecture Dataflow of AND_NOR_df_TBW is

component AND_NOR_df is

   Port ( A : in STD_LOGIC;

          B : in STD_LOGIC;

          C : out STD_LOGIC);

end component;

signal a1:STD_LOGIC :='0';

signal b1:STD_LOGIC :='0';

signal c1:STD_LOGIC;

begin

uut:AND_NOR_df Port Map(A=>a1,B=>b1,C=>c1);

stim_proc:process

begin

wait for 100ns;

a1<='0';

b1<='0';

wait for 100ns;

a1<='0';

b1<='1';

wait for 100ns;

a1<='1';

b1<='0';

wait for 100ns;

a1<='1';

b1<='1';
```

```
wait;

end process;


end Dataflow;
```

TEST BENCH WAVEFORM

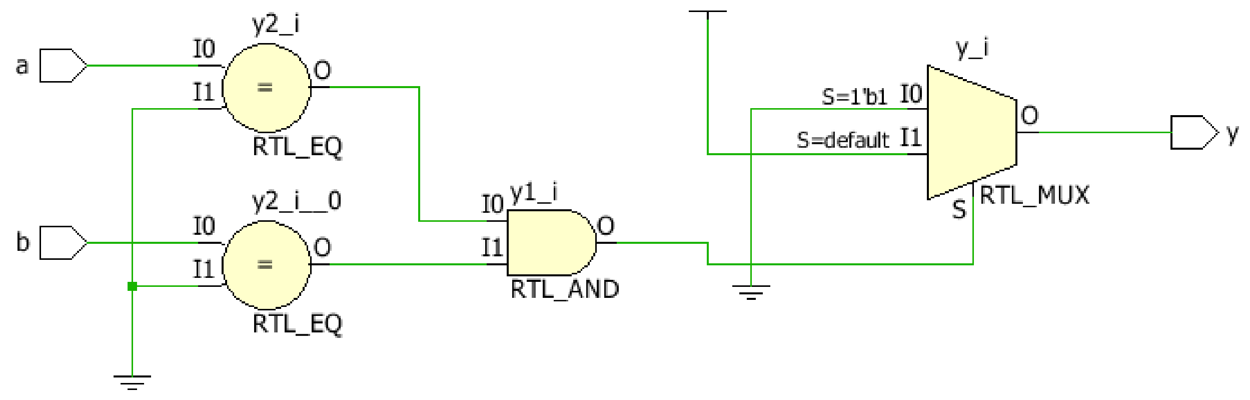


IMPLEMENTATION OF OR GATE IN BEHAVIORIAL MODE

VHDL code:

```
entity or_bv is
   Port ( a : in STD_LOGIC;
          b : in STD_LOGIC;
          y : out STD_LOGIC);
end or_bv;

architecture Behavioral of or_bv is

begin
process(a,b)
begin
if(a='0' and b= '0') then
y<='0';
else
y<='1';
end if;
end process;


end Behavioral;
```

SCHEMATIC:

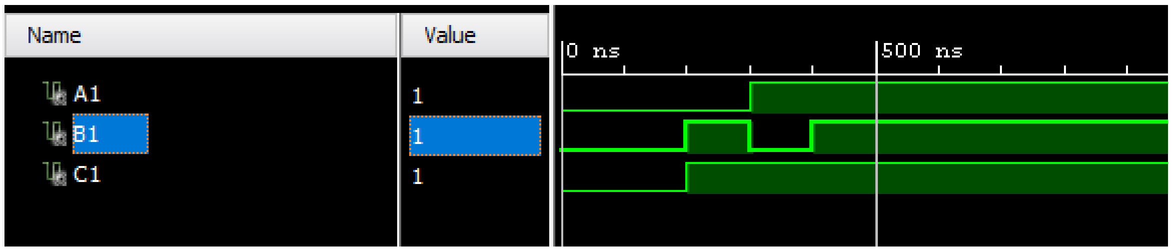


TBW CODE:

```
entity or_tbw is

--  Port ( );

end or_tbw;


architecture Behavioral of or_tbw is

component or_bv is

   Port ( a : in STD_LOGIC;

          b : in STD_LOGIC;

          y : out STD_LOGIC);

end component;

signal a1:STD_LOGIC :='0';

signal b1:STD_LOGIC :='0';

signal c1:STD_LOGIC;


begin

uut:or_bv Port Map(a=>a1,b=>b1,y=>c1);
```

```
stim_proc:process
begin
wait for 100ns;
a1<='0';
b1<='0';
wait for 100ns;
a1<='0';
b1<='1';
wait for 100ns;
a1<='1';
b1<='0';wait for 100ns;
a1<='1';
b1<='1';
wait;
end process;
end Behavioral;
```

WAVEFORM:

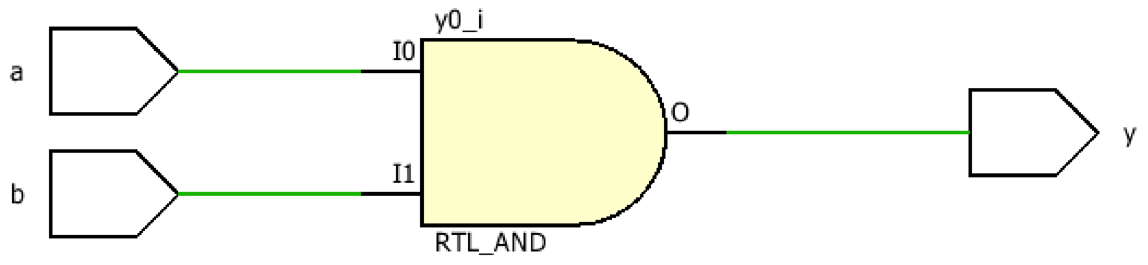


IMPLEMENTATION OF AND GATE IN BEHAVIORIAL MODE

VHDL CODE:

```
entity AND_DF is
   Port ( a : in STD_LOGIC;
          b : in STD_LOGIC;
          y : out STD_LOGIC);
end AND_DF;

architecture Behavioral of AND_DF is

begin
process(a,b)
begin if(a='1' and b='1')then
y<='1';
else
y<='0';
end if;
end process;

end Behavioral;
```

schematic:

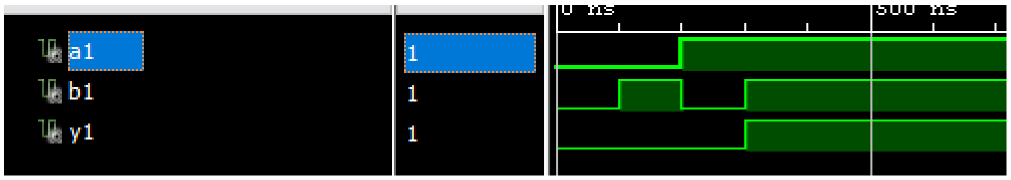


Tbw code:

```
entity and_bv_tbw is

--  Port ( );

end and_bv_tbw;


architecture Behavioral of and_bv_tbw is

component AND_DF is

   Port ( a : in STD_LOGIC;

          b : in STD_LOGIC;

          y : out STD_LOGIC);

end component;

signal a1:STD_LOGIC :='0';

signal b1:STD_LOGIC :='0';

signal c1:STD_LOGIC;


begin

uut:AND_DF Port Map(a=>a1,b=>b1,y=>c1);
```

```
stim_proc:process

begin

wait for 100ns;

a1<='0';

b1<='0';

wait for 100ns;

a1<='0';

b1<='1';

wait for 100ns;

a1<='1';

b1<='0';wait for 100ns;

a1<='1';

b1<='1';

wait;

end process;

end Behavioral;
```

wave form:

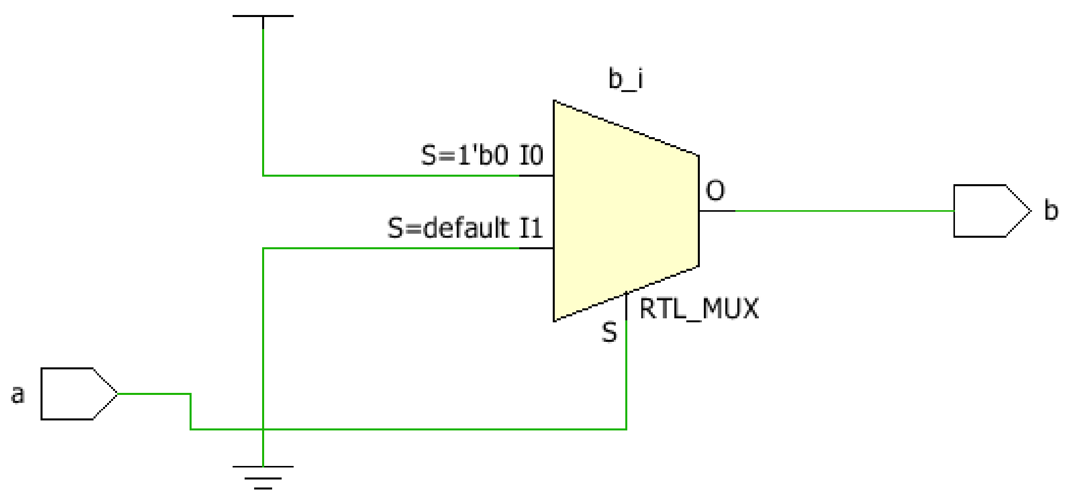


IMPLEMENTATION OF NOT GATE IN BEHAVIORIAL MODE

VHDL CODE:

```
entity NOT_BV is
   Port ( a : in STD_LOGIC;
          b : out STD_LOGIC);
end NOT_BV;

architecture Behavioral of NOT_BV is
begin
process(a)
begin
if(a='0')then
b<='1';
else
b<='0';
end if;
end process;



end Behavioral;
```

SCHEMATIC:

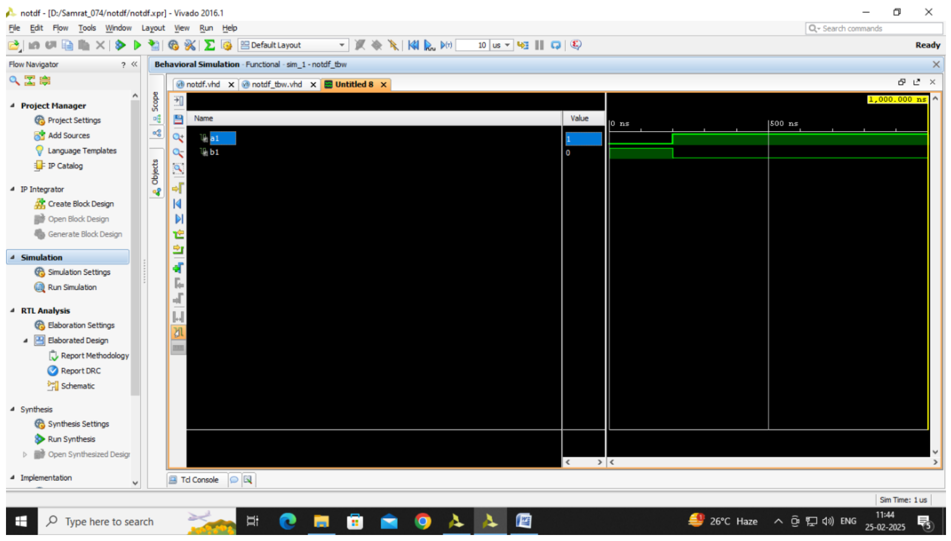


TBW CODE:

```
entity NOT_DF_TBW is

--  Port ( );

end NOT_DF_TBW;


architecture Behavioral of NOT_DF_TBW is

component NOT_BV is

   Port ( a : in STD_LOGIC;

          b : out STD_LOGIC);

end component;

signal a1:STD_LOGIC :='0';

signal b1:STD_LOGIC;


begin

uut:NOT_BV Port Map(a=>a1,b=>b1);
```

```
stim_proc:process

begin

wait for 100ns;

a1<='0';


wait for 100ns;

a1<='1';


wait;

end process;

end Behavioral;
```

WAVEFORM:



IMPLEMENTATION OF NAND Gate IN BEHAVIORIAL MODE

Truth Table:

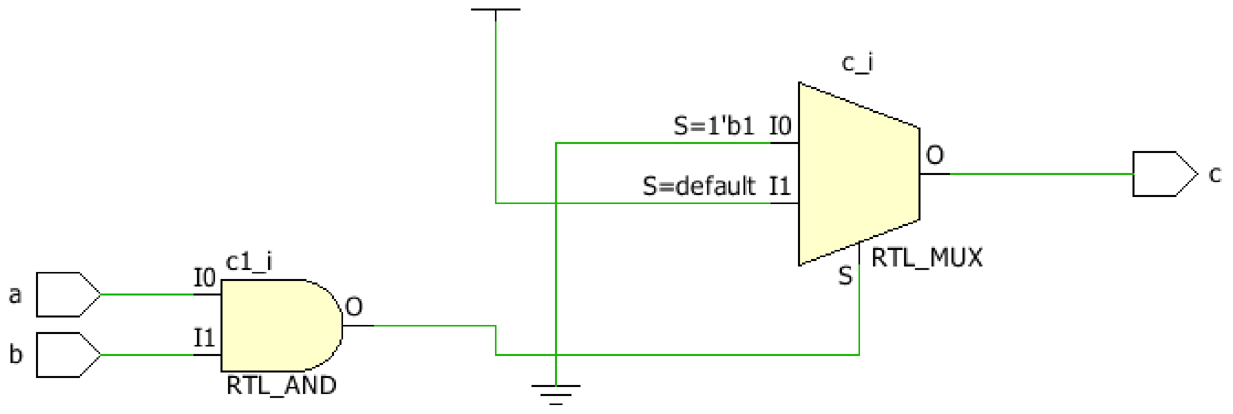
A	B	C
0	0	0
0	1	0
1	0	0
1	1	1

VHDL Code

```
entity nand_bv is

   Port ( a : in STD_LOGIC;

          b : in STD_LOGIC;

          c : out STD_LOGIC);

end nand_bv;


architecture Behavioral of nand_bv is


begin


process(a,b)

begin

if(a='1' AND b='1') then

   c<='0';

else
```

```
    c<='1';
end if;
end process;


end Behavioral;
```

SCHEMATIC

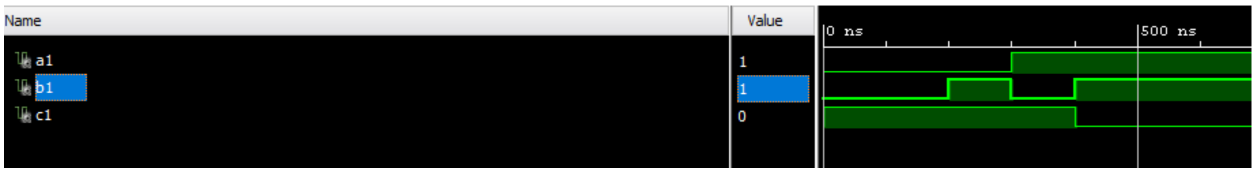


TBW Code

```
architecture Behavioral of nand_bv_tbw is
component nand_bv is
    Port ( a : in STD_LOGIC;
           b : in STD_LOGIC;
           c : out STD_LOGIC);
end component;
Signal a1:STD_LOGIC:='0';
Signal b1:STD_LOGIC:='0';
Signal c1:STD_LOGIC;
```

```
begin

uut: nand_bv port map(a=>a1,b=>b1,c=>c1);

stim_proc: process

begin

wait for 100 ns;

a1<='0';

b1<='0';


wait for 100 ns;

a1<='0';

b1<='1';


wait for 100 ns;

a1<='1';

b1<='0';


wait for 100 ns;

a1<='1';

b1<='1';

wait;


end process;

end Behavioral;
```

Waveform



IMPLEMENTATION OF NOR Gate IN BEHAVIORIAL MODE

Truth Table:

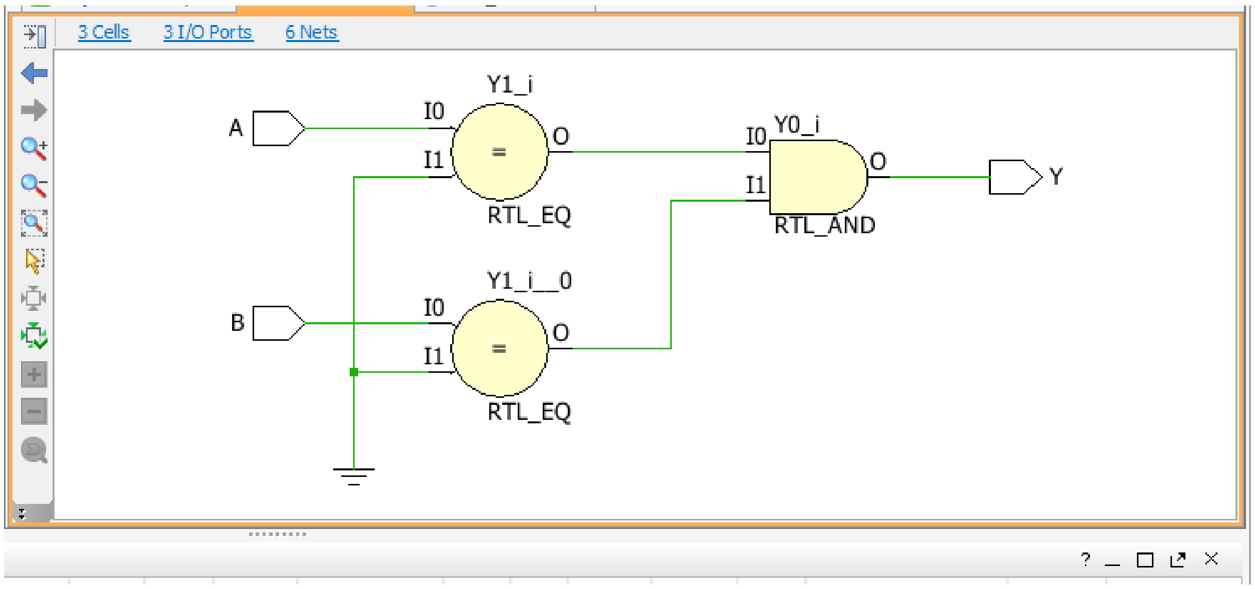
A	B	C
0	0	0
0	1	1
1	0	1
1	1	0

VHDL Code

```
architecture Behavioral of nor_bv is


begin

process(a,b)

begin

if(a='0' AND b='0') then

   c<='1';

else

   c<='0';

end if;

end process;
```

end Behavioral;

SCHEMATIC



TBW Code

architecture Behavioral of nor_bv_tbw is

component nor_bv is

```
    Port ( a : in STD_LOGIC;
           b : in STD_LOGIC;
           c : out STD_LOGIC);
```

end component;

Signal a1:STD_LOGIC:='0';

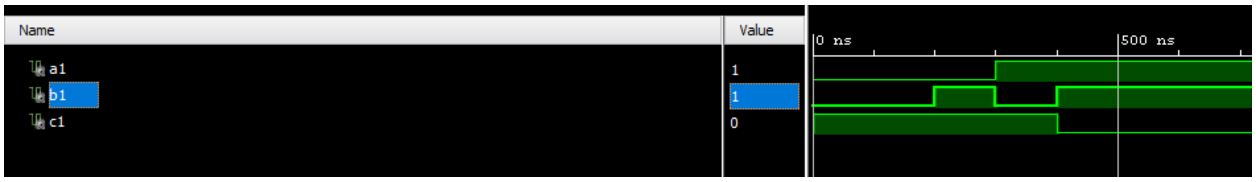
Signal b1:STD_LOGIC:='0';

Signal c1:STD_LOGIC;

begin

```
uut: nor_bv port map(a=>a1,b=>b1,c=>c1);

stim_proc: process

begin

wait for 100 ns;

a1<='0';

b1<='0';


wait for 100 ns;

a1<='0';

b1<='1';


wait for 100 ns;

a1<='1';

b1<='0';


wait for 100 ns;

a1<='1';

b1<='1';

wait;


end process;

end Behavioral;
```

Waveform



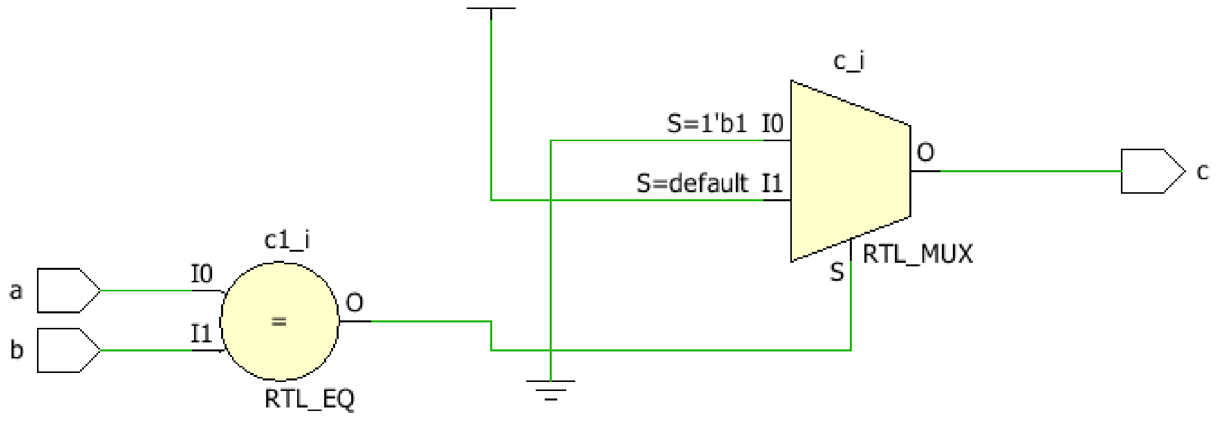
IMPLEMENTATION OF XOR Gate IN BEHAVIORIAL MODE

VHDL Code

```
entity xor_bv is
   Port ( a : in STD_LOGIC;
          b : in STD_LOGIC;
          c : out STD_LOGIC);
end xor_bv;

architecture Behavioral of xor_bv is

begin
process(a,b)
begin
if(a=b) then
   c<='0';
else
   c<='1';
end if;
end process;
```

end Behavioral;

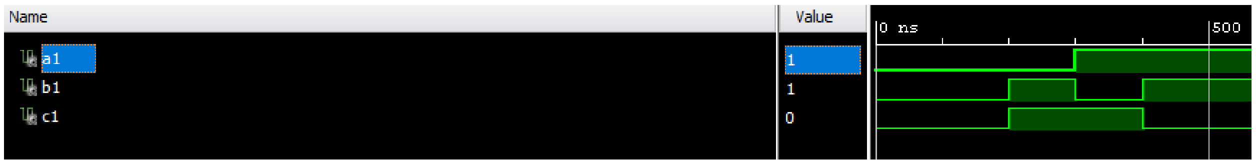
SCHEMATIC



TBW Code

```
entity xor_bv_tbw is

--  Port ( );

end xor_bv_tbw;


architecture Behavioral of xor_bv_tbw is

component xor_bv is

   Port ( a : in STD_LOGIC;

          b : in STD_LOGIC;

          c : out STD_LOGIC);

end component;

Signal a1:STD_LOGIC:='0';

Signal b1:STD_LOGIC:='0';
Signal c1:STD_LOGIC;

begin

uut: xor_bv port map(a=>a1,b=>b1,c=>c1);

stim_proc: process

begin

wait for 100 ns;

a1<='0';

b1<='0';


wait for 100 ns;

a1<='0';

b1<='1';


wait for 100 ns;

a1<='1';

b1<='0';


wait for 100 ns;

a1<='1';

b1<='1';

wait;


end process;

end Behavioral;
```

Waveform



IMPLEMENTATION OF XNOR Gate IN BEHAVIORIAL MODE

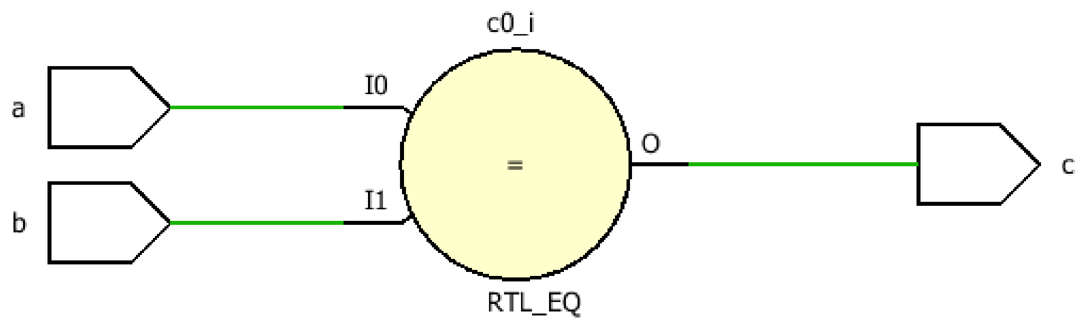
Truth Table:

A	B	C
0	0	0
0	1	0
1	0	0
1	1	1

VHDL Code

```
entity xnor_bv is
  Port ( a : in STD_LOGIC;
       b : in STD_LOGIC;
       c : out STD_LOGIC);
end xnor_bv;


architecture Behavioral of xnor_bv is


begin
process(a,b)
```

```
begin
if(a=b) then
   c<='1';
else
   c<='0';
end if;
end process;


end Behavioral;
```



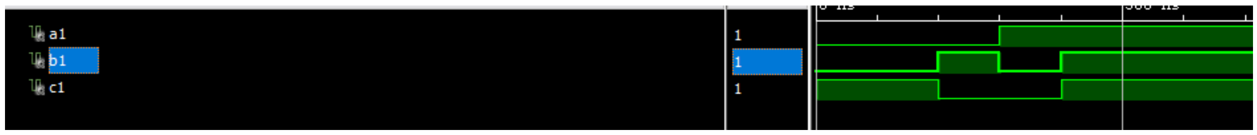
TBW Code

```
architecture Behavioral of xnor_bv_tbw is
component xnor_bv is
   Port ( a : in STD_LOGIC;
          b : in STD_LOGIC;
          c : out STD_LOGIC);
```

```
end component;

Signal a1:STD_LOGIC:='0';

Signal b1:STD_LOGIC:='0';

Signal c1:STD_LOGIC;

begin

uut: xnor_bv port map(a=>a1,b=>b1,c=>c1);

stim_proc: process

begin

wait for 100 ns;

a1<='0';

b1<='0';


wait for 100 ns;

a1<='0';

b1<='1';


wait for 100 ns;

a1<='1';

b1<='0';


wait for 100 ns;

a1<='1';

b1<='1';

wait;
```

```
end process;

end Behavioral;
```

Waveform



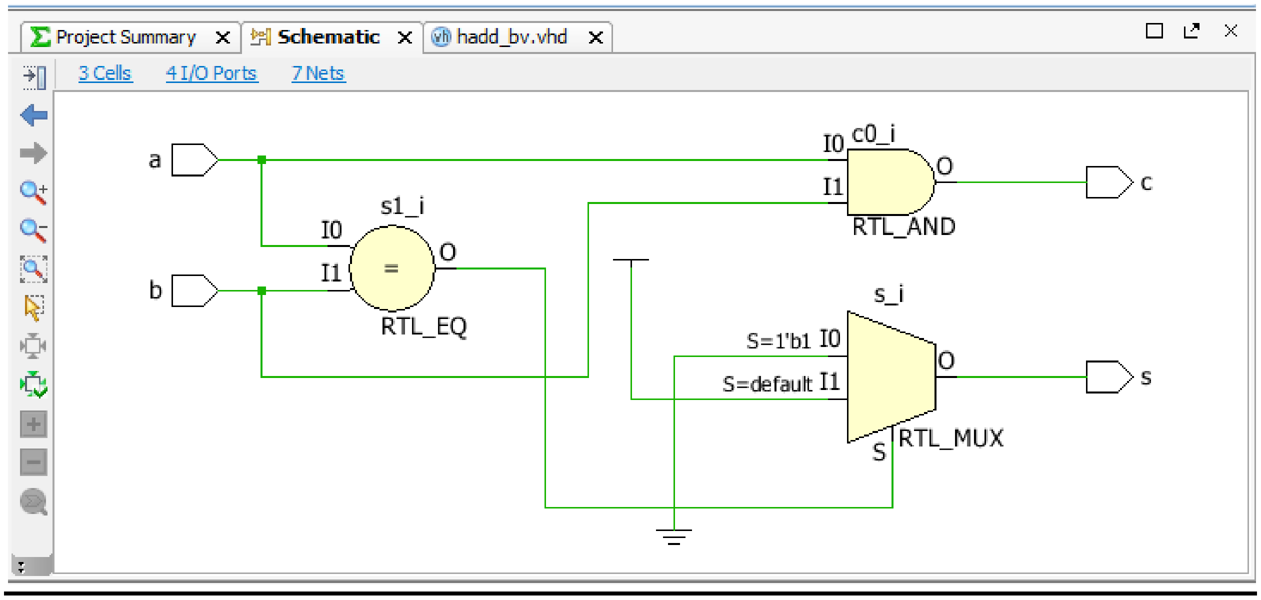
Half Adder Behavioral

VHD Code

```
entity half_adder_df is
   Port ( a : in STD_LOGIC;
          b : in STD_LOGIC;
          s : out STD_LOGIC;
          c : out STD_LOGIC);
end half_adder_df;


architecture Behavioral of half_adder_df is


begin
```

```
process(a, b)

begin

if(a = b) then

   s <= '0';

else

   s <= '1';

end if;

if(a = '1' and b='1') then

   c <= '1';

else

   c <= '0';

end if;

end process;



end Behavioral;
```

Schematic Diagram



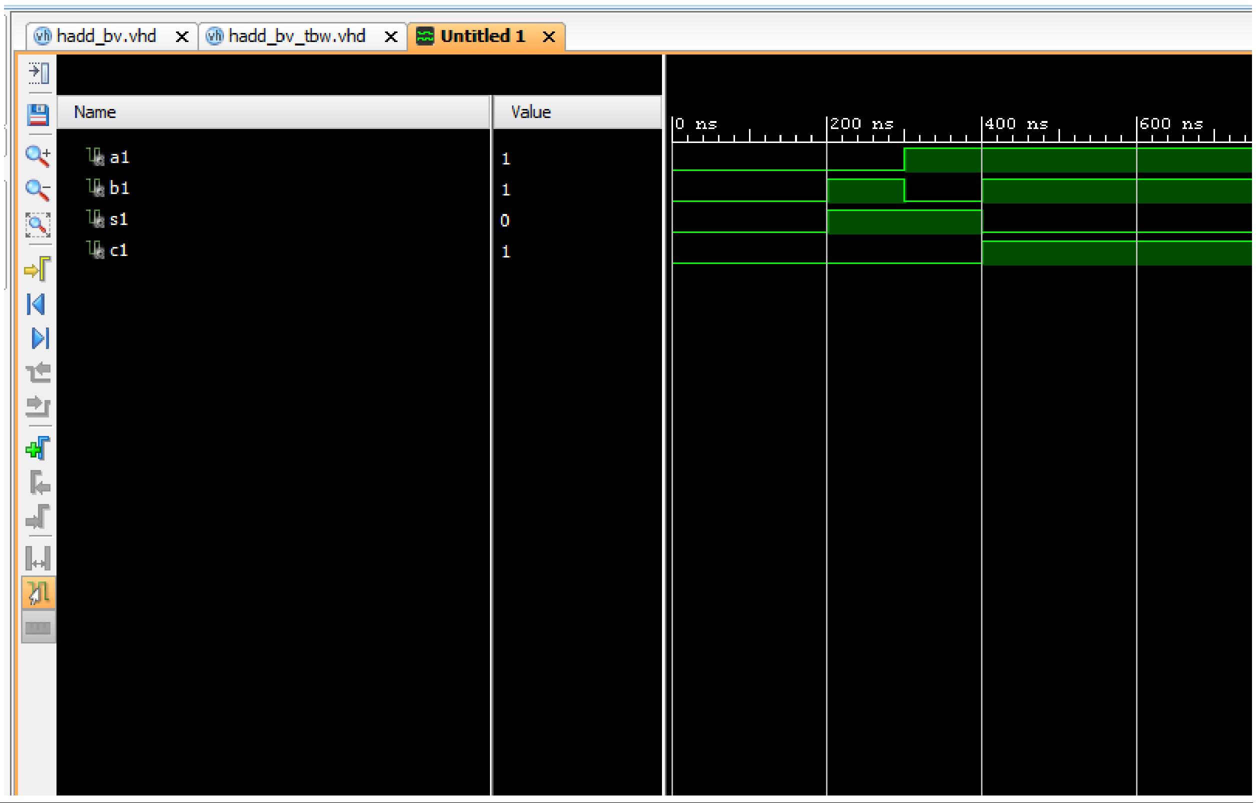
Test Bench Code

```
entity half_adder_df _tbw is

--  Port ( );

end half_adder_df _tbw;


architecture Behavioral of half_adder_df _tbw is


component half_adder_df is

Port ( a : in STD_LOGIC;

          b : in STD_LOGIC;

          s : out STD_LOGIC;
```

```
        c : out STD_LOGIC);
end component;


Signal a1: STD_LOGIC := '0';
Signal b1: STD_LOGIC := '0';
Signal s1: STD_LOGIC;
Signal c1: STD_LOGIC;
begin
uut: half_adder_df port map (a=>a1, b=>b1, s=>s1, c=>c1);
stim_proc: process
begin
wait for 100ns;
a1 <= '0';
b1 <= '0';
wait for 100ns;
a1 <= '0';
b1 <= '1';
wait for 100ns;
```

```
a1 <= '1';

b1 <= '0';

wait for 100ns;

a1 <= '1';

b1 <= '1';

wait;

end process;

end Behavioral;
```

Test Bench Waveform



Name	Value
a1	1
b1	1
s1	0
c1	1

Truth Table

A	B	S	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Full Adder Behavioral

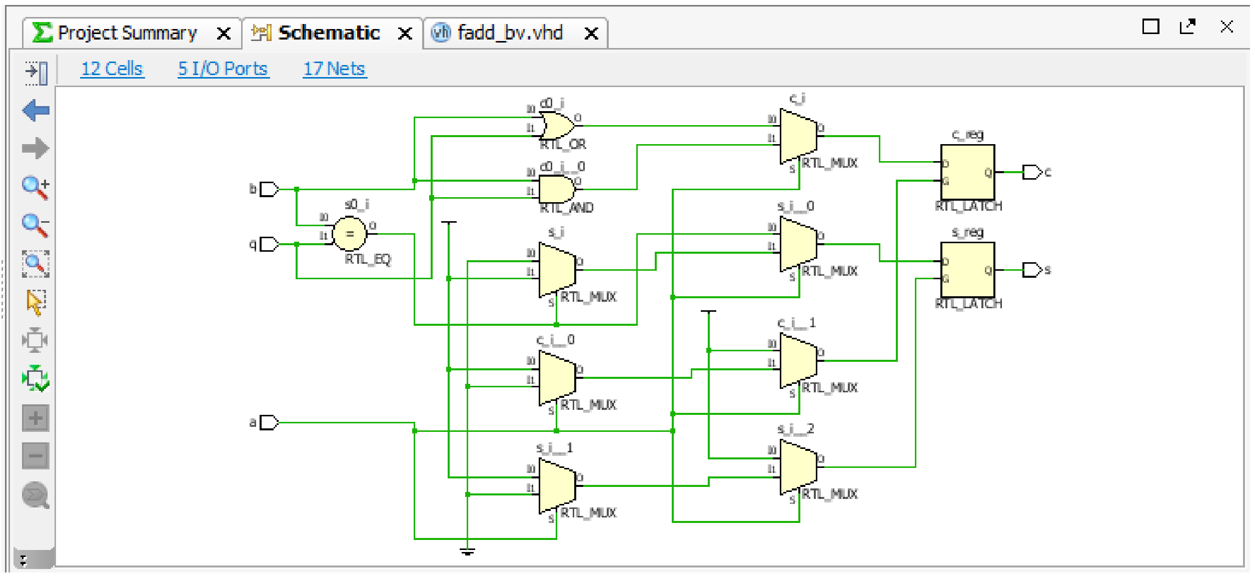
VHD Code

```
entity fadd_bv is
   Port ( a : in STD_LOGIC;
          b : in STD_LOGIC;
          q : in STD_LOGIC;
          s : out STD_LOGIC;
          c : out STD_LOGIC);
end fadd_bv;

architecture Behavioral of fadd_bv is

begin
process(a,b,q)
begin
```

```
if(a = '0')then

if(b = q)then

s <= '0';

else

s <= '1';

end if;

if(b = '1' AND q = '1')then

c <= '1';

else

c <= '0';

end if;

end if;

if(a = '1')then

if(b = q)then

s <= '1';

else

s <= '0';

end if;
```

```
if(b = '1' OR q = '1')then

c <= '1';

else

c <= '0';

end if;

end if;

end process;




end Behavioral;
```

Schematic Diagram



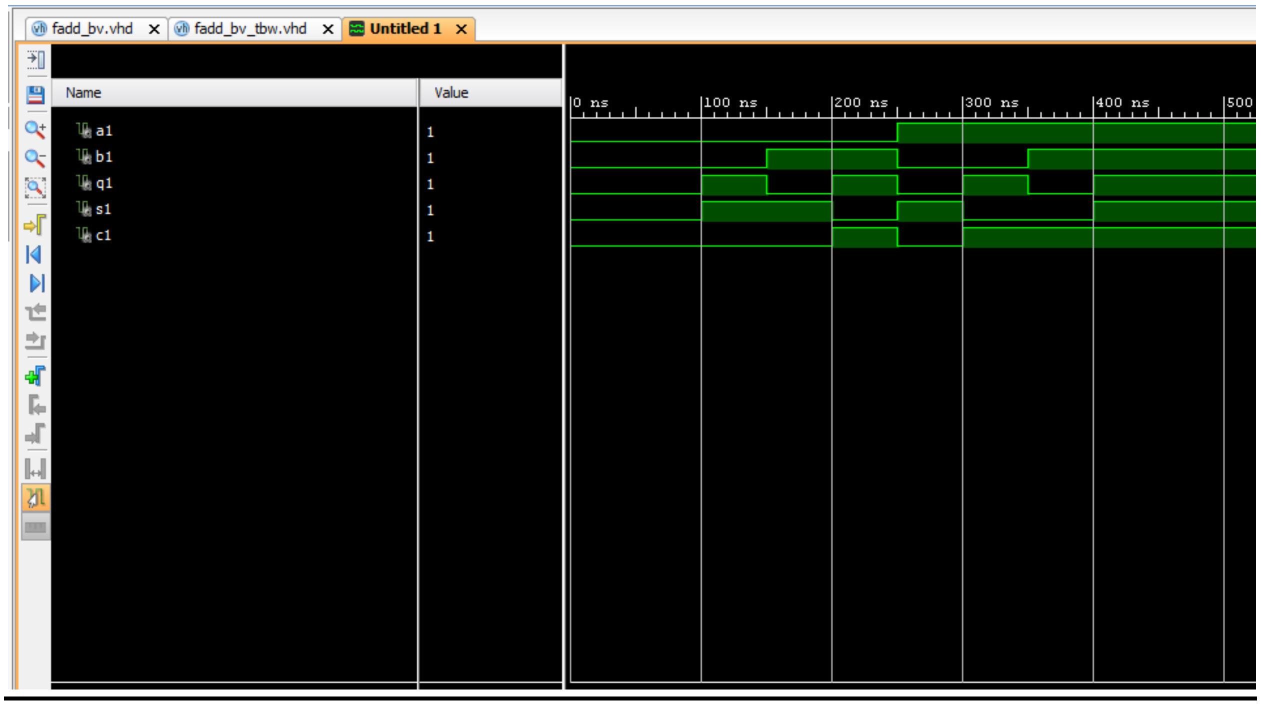
Test Bench Code

```
entity fadd_bv_tbw is

--  Port ( );

end fadd_bv_tbw;


architecture Behavioral of fadd_bv_tbw is


component fadd_bv is

Port ( a : in STD_LOGIC;

       b : in STD_LOGIC;

       q : in STD_LOGIC;

       s : out STD_LOGIC;

       c : out STD_LOGIC);

end component;


Signal a1: STD_LOGIC := '0';

Signal b1: STD_LOGIC := '0';

Signal q1: STD_LOGIC := '0';
Signal s1: STD_LOGIC;

Signal c1: STD_LOGIC;

begin

uut:fadd_bv Port Map(a=>a1,b=>b1,q=>q1,s=>s1,c=>c1);

stim_proc:process

begin

        wait for 50 ns;

        a1 <= '0';

        b1 <= '0';

        q1 <= '0';

        wait for 50ns;

        a1 <= '0';

        b1 <= '0';

        q1  <= '1';

        wait for 50ns;

        a1 <= '0';

        b1 <= '1';

        q1  <= '0';
```

```
 wait for 50ns;

 a1 <= '0';

 b1 <= '1';

 q1  <= '1';

 wait for 50ns;

 a1 <= '1';

 b1 <= '0';

 q1  <= '0';

 wait for 50ns;

 a1 <= '1';

 b1 <= '0';

 q1  <= '1';

 wait for 50ns;

 a1 <= '1';

 b1 <= '1';

 q1  <= '0';

wait for 50ns;
```

```
      a1 <= '1';

      b1 <= '1';

      q1  <= '1';



       wait;

   end process;



end Behavioral;
```

Test Bench Waveform



Name	Value
a1	1
b1	1
q1	1
s1	1
c1	1

Truth Table

A	B	Q	S	C
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

Full Adder [Structural]

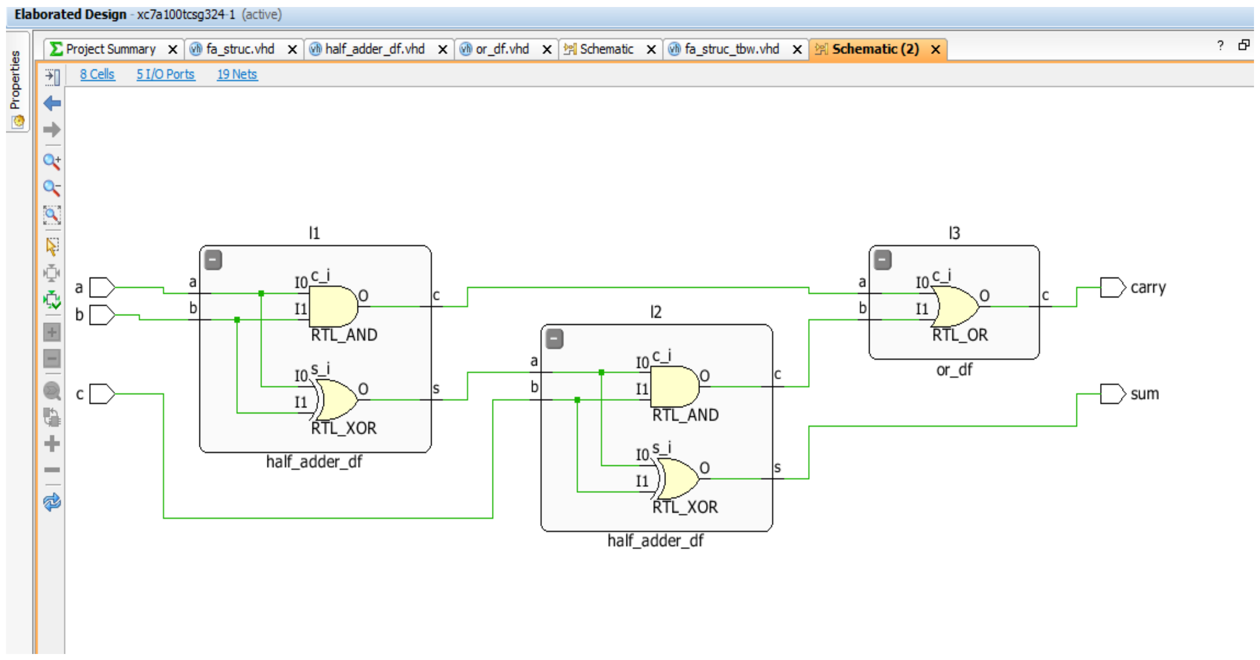
VHD Code

```
entity fa_struc is
   Port ( a : in STD_LOGIC;
          b : in STD_LOGIC;
          c : in STD_LOGIC;
          sum : out STD_LOGIC;
          carry : out STD_LOGIC);
end fa_struc;


architecture structural of fa_struc is


component half_adder_df is
   Port ( a : in STD_LOGIC;
          b : in STD_LOGIC;
          s : out STD_LOGIC;
          c : out STD_LOGIC);
```

```
end component;

component or_df is
    Port ( a : in STD_LOGIC;
           b : in STD_LOGIC;
           c : out STD_LOGIC);
end component;

Signal c1, c2, s1 : std_logic;

begin

l1:half_adder_df Port Map(a, b, s1, c1);
l2:half_adder_df Port Map(s1, c, sum, c2);
l3:or_df Port Map(c1, c2, carry);
```

```
end structural;
```

Schematic Diagram



Test Bench Code

```
entity fa_struc_tbw is
--  Port ( );
end fa_struc_tbw;


architecture Behavioral of fa_struc_tbw is
```

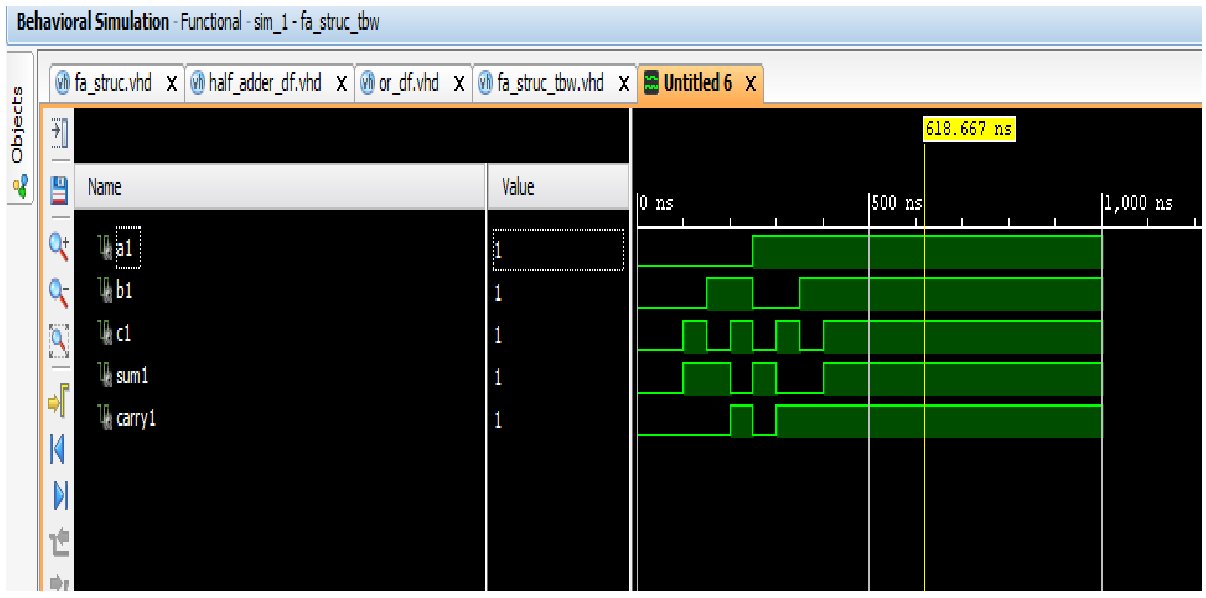
```
component fa_struc is
Port ( a : in STD_LOGIC;
       b : in STD_LOGIC;
       c : in STD_LOGIC;
       sum : out STD_LOGIC;
       carry : out STD_LOGIC);
end component;

Signal a1: STD_LOGIC := '0';
Signal b1: STD_LOGIC := '0';
Signal c1: STD_LOGIC := '0';
Signal sum1: STD_LOGIC;
Signal carry1: STD_LOGIC;
begin
uut:fa_struc Port
Map(a=>a1,b=>b1,c=>c1,sum=>sum1,carry=>carry1);
stim_proc:process
begin
```

```
wait for 50 ns;

a1 <= '0';

b1 <= '0';

c1 <= '0';

wait for 50ns;

a1 <= '0';

b1 <= '0';

c1  <= '1';

wait for 50ns;

a1 <= '0';

b1 <= '1';

c1  <= '0';


wait for 50ns;

a1 <= '0';

b1 <= '1';

c1  <= '1';

wait for 50ns;
```

```
a1 <= '1';

b1 <= '0';

c1  <= '0';

wait for 50ns;

a1 <= '1';

b1 <= '0';

c1  <= '1';

wait for 50ns;

a1 <= '1';

b1 <= '1';

c1  <= '0';

wait for 50ns;

a1 <= '1';

b1 <= '1';

c1  <= '1';


wait;

end process;
```

end Behavioral;

Test Bench Waveform



Name	Value
a1	1
b1	1
c1	1
sum1	1
carry1	1

ALU Behavioral (03/04/2025)

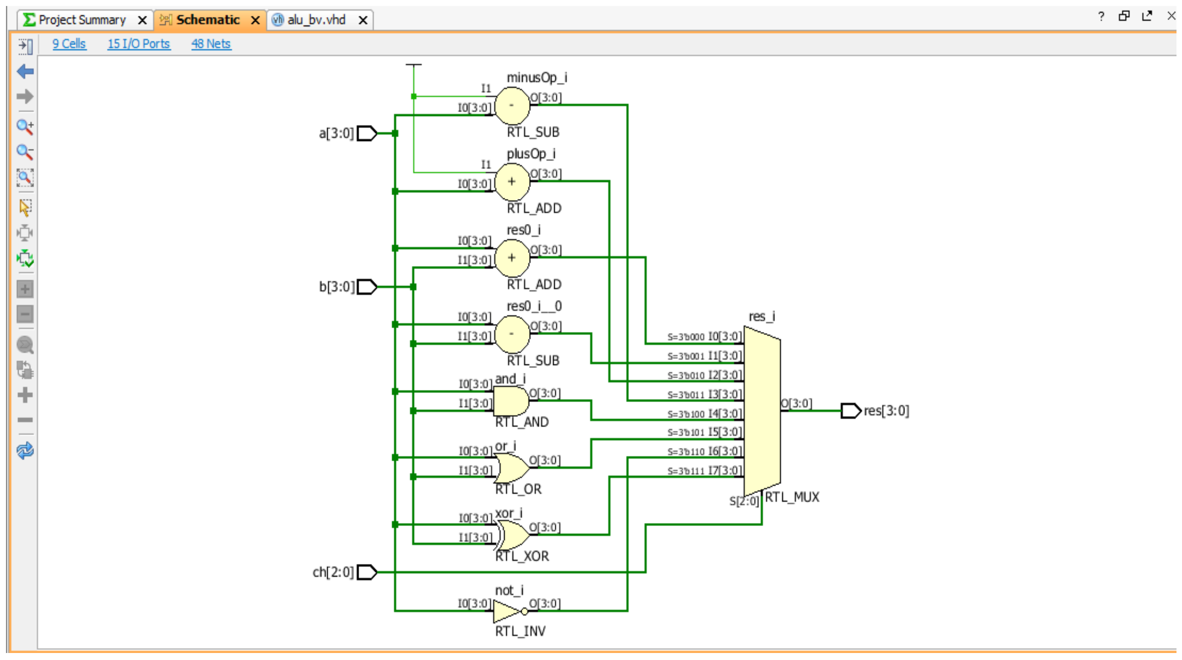
VHD Code

```
entity alu_bv is
   Port ( a : in unsigned (3 downto 0);
          b : in unsigned (3 downto 0);
          ch : in STD_LOGIC_VECTOR (2 downto 0);
          res : out unsigned (3 downto 0));
end alu_bv;
```

```
architecture Behavioral of alu_bv is

begin

process(a, b, ch)
begin
case ch is
when "000" => res <= a+b;
when "001" => res <= a-b;
when "010" => res <= a+1;
when "011" => res <= a-1;
when "100" => res <= a and b;
when "101" => res <= a or b;
when "110" => res <= not a;
when "111" => res <= a xor b;

when others => NULL;
```

```
end case;

end process;


end Behavioral;
```

Schematic Diagram



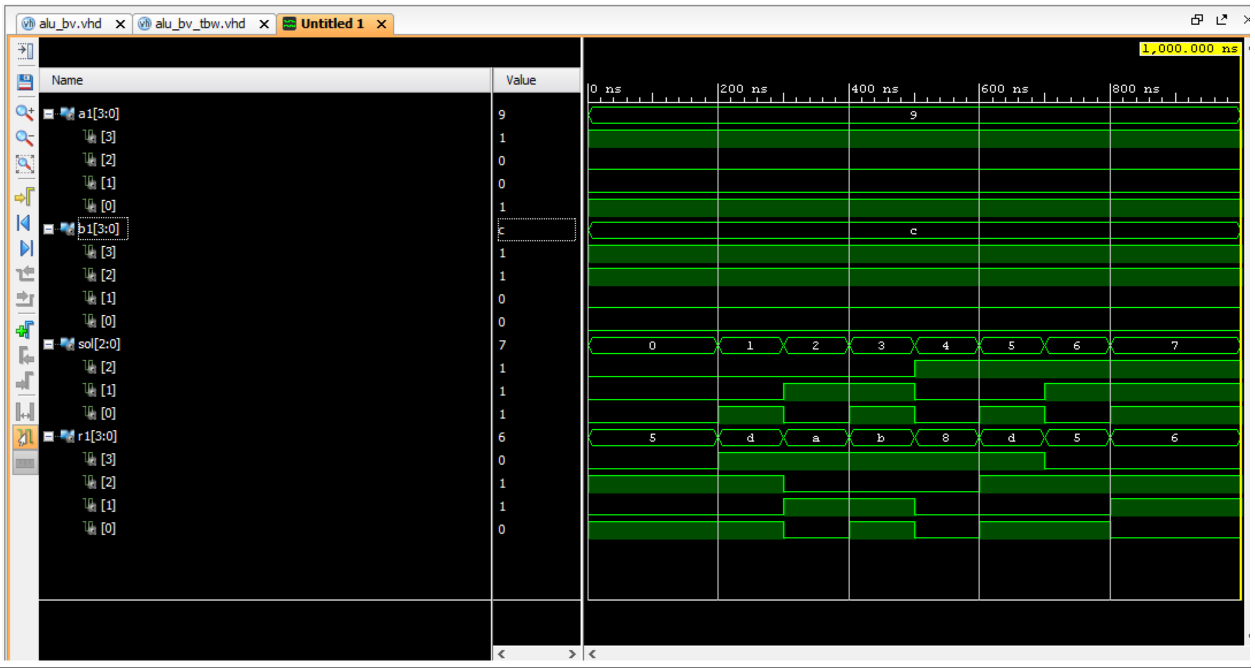
Test Bench Code

```
entity alu_bv_tbw is

--  Port ( );

end alu_bv_tbw;

architecture Behavioral of alu_bv_tbw is
```

```
component alu_bv is
   Port ( a:in unsigned (3 downto 0);
          b:in unsigned (3 downto 0);
          s:in STD_LOGIC_VECTOR (2 downto 0);
          r:out unsigned (3 downto 0));
end component;
Signal a1:unsigned(3 downto 0):="1001";
Signal b1:unsigned(3 downto 0):="1100";
Signal sol:STD_LOGIC_VECTOR(2 downto 0):="000";
Signal r1:unsigned(3 downto 0);
begin
uut:alu_bv port map(a=>a1, b=>b1, s=>sol, r=>r1);
stim_proc: process
begin
wait for 100ns;
sol<="000";
wait for 100ns;
```

```
sol<="001";
wait for 100ns;
sol<="010";
wait for 100ns;
sol<="011";
wait for 100ns;
sol<="100";
wait for 100ns;
sol<="101";
wait for 100ns;
sol<="110";
wait for 100ns;
sol<="111";
wait;
end process;

end Behavioral;
```

Test Bench Waveform



Name	Value
a1[3:0]	9
[3]	1
[2]	0
[1]	0
[0]	1
b1[3:0]	c
[3]	1
[2]	1
[1]	0
[0]	0
sol[2:0]	7
[2]	1
[1]	1
[0]	1
r1[3:0]	6
[3]	0
[2]	1
[1]	1
[0]	0