HERITAGE INSTITUTE OF TECHNOLOGY

Computer Organization & Architecture Lab

CSE 2252

Lab Assignments

Stream: CSE   Section: B   Group :2

2nd year 2nd semester

Session: 2024 – 2025

NAME	CLASS ROLL NUMBER	AUTONOMY ROLL NUMBER
Satyam Kumar Singh	2351097	12623001136
Sunil Kumar Shao	2351098	12623001174
Smit Kumar	2351099	12623001158
Devayanee Gupta	2351100	12623001067
Shreya Denre	2351101	12623001148
Arijit Acharya	2351102	12623001037

INDEX

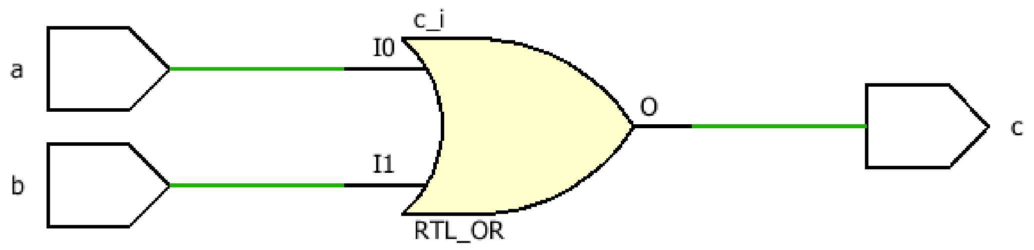
DAY 1

OR Gate (Dataflow Model)

1. VHDL Code

```
entity ord is
Port ( a : in STD_LOGIC;
b : in STD_LOGIC;
c : out STD_LOGIC);
end ord;
architecture Dataflow of ord is
begin
c<=a OR b;
end Dataflow;
```

2. RTL Diagram

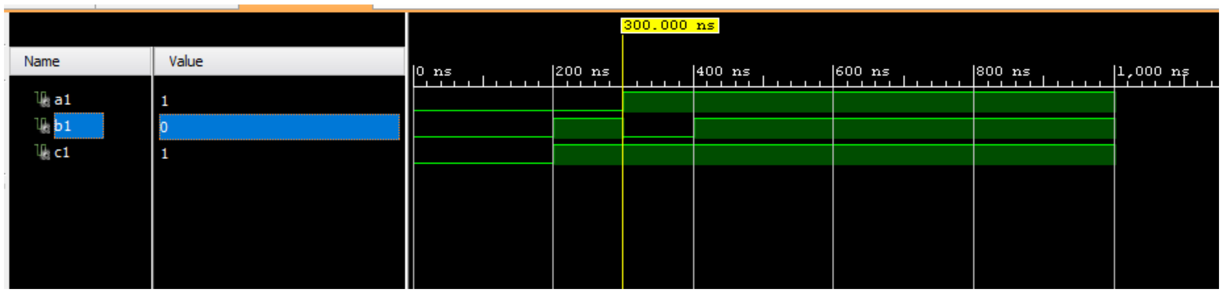


3. TBW Code

```
entity ord_tbw is
-- Port ( );
end ord_tbw;
architecture Behavioral of ord_tbw is
component ord 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: ord 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;
```

4.Waveform

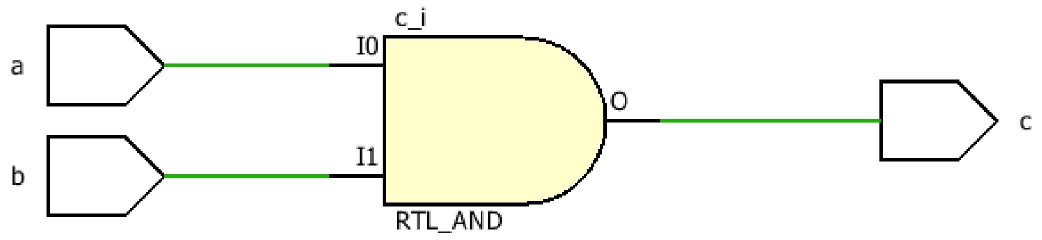


AND Gate (Dataflow Model)

1. **VHDL Code**

```
entity andd is
   Port ( a : in STD_LOGIC;
         b : in STD_LOGIC;
         c : out STD_LOGIC);
end andd;
architecture Dataflow of andd is
begin
c<= a and b;
end Dataflow;
```

2.RTL Diagram

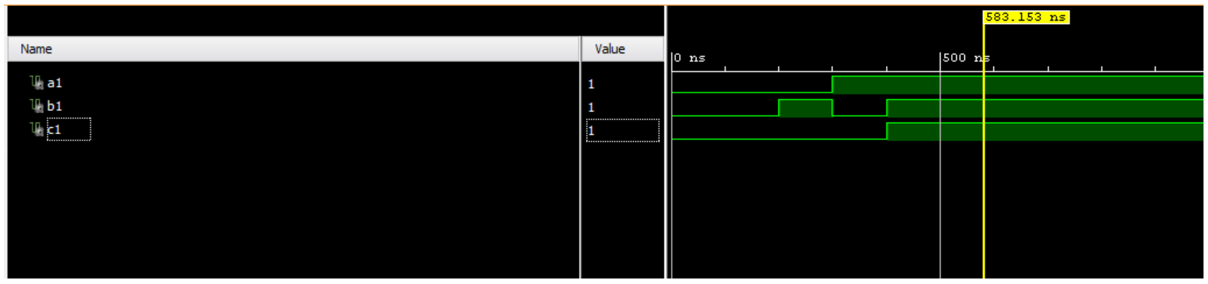


3.TBW Code

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

architecture Behavioral of andd_tbw is
component andd 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: andd 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;
```

4.Waveform

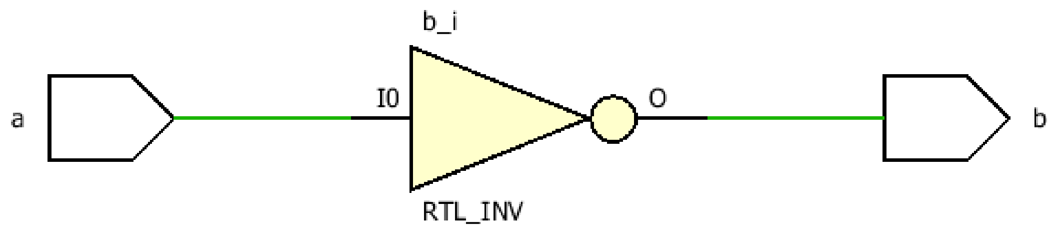


NOT Gate (Dataflow Model)

1. **VHDL Code**

```
entity notd is
   Port ( a : in STD_LOGIC;
          b : out STD_LOGIC);
end notd;
architecture Dataflow of notd is
begin
b<= not a;
end Dataflow;
```

2. **RTL Diagram**

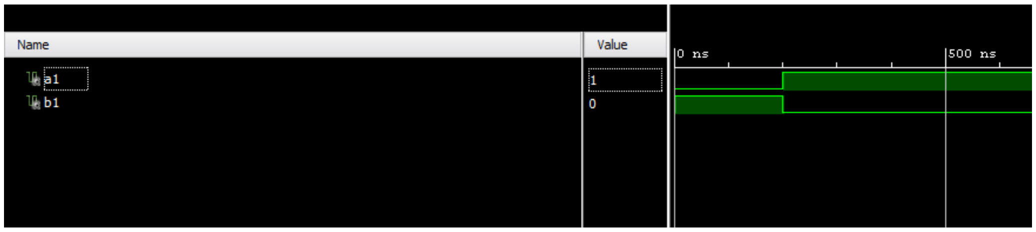


3. **TBW Code**

```
entity notd_tbw is
--  Port ( );
end notd_tbw;
architecture Behavioral of notd_tbw is
component notd is
   Port ( a : in STD_LOGIC;
          b : out STD_LOGIC);
end component;
signal a1: STD_LOGIC:='0';
signal b1: STD_LOGIC;
begin
uut: notd port map(a=>a1,b=>b1);
stim_proc: process
begin
wait for 100 ns;
a1<='0';
```

wait for 100 ns;
a1<='1';
wait;
end process;
end Behavioral;

4. Waveform

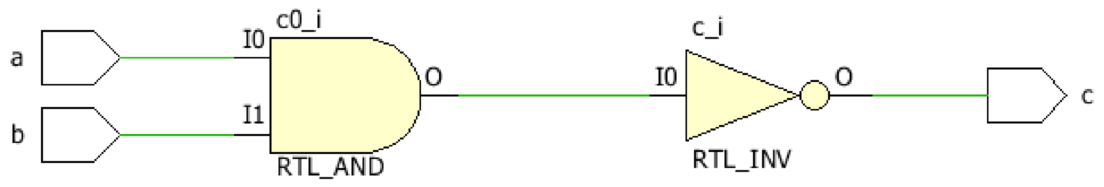


NAND Gate (Dataflow Model)

1.VHDL Code

```
entity nandd is
   Port ( a : in STD_LOGIC;
         b : in STD_LOGIC;
         c : out STD_LOGIC);
end nandd;
architecture Dataflow of nandd is
begin
c<=a nand b;
end Dataflow;
```

2.RTL Diagram

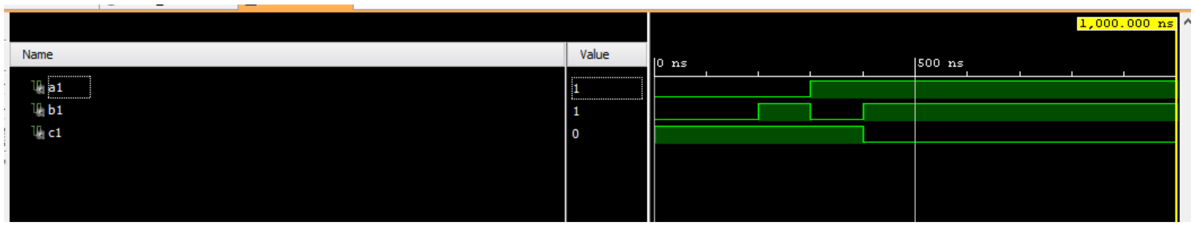


3.TBW Code

```
entity nandd_tbw is
--  Port ( );
end nandd_tbw;
architecture Behavioral of nandd_tbw is
component nandd 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: nandd 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;
```

4.Waveform

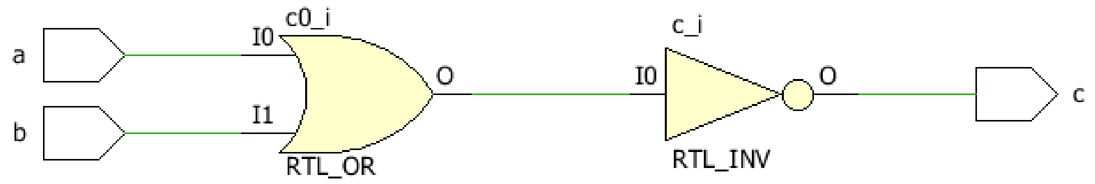


NOR Gate (Dataflow Model)

1.VHDL Code

```
entity nord is
   Port ( a : in STD_LOGIC;
         b : in STD_LOGIC;
         c : out STD_LOGIC);
end nord;
architecture Dataflow of nord is
begin
c<=a nor b;
end Dataflow;
```

2.RTL Design

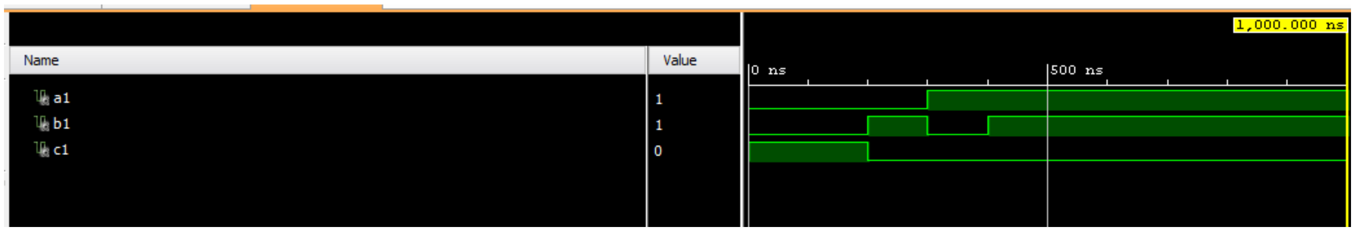


3.TBW Code

```
entity nord_tbw is
--  Port ( );
end nord_tbw;
architecture Behavioral of nord_tbw is
component nord 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: nord 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;
```

4.Waveform

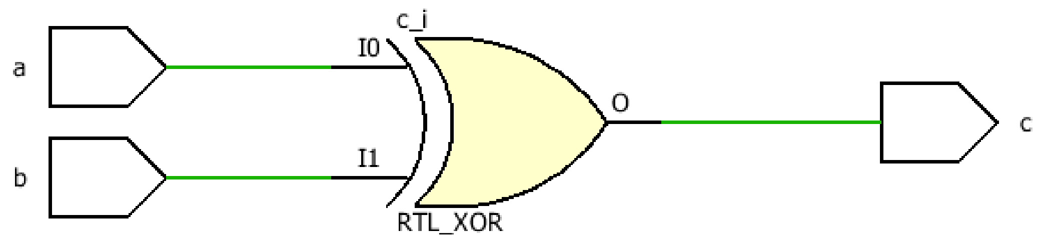


XOR Gate (Dataflow Model)

1.VHDL Code

```
entity xord is
   Port ( a : in STD_LOGIC;
         b : in STD_LOGIC;
         c : out STD_LOGIC);
end xord;
architecture Dataflow of xord is
begin
c<=a xor b;
end Dataflow;
```

2.RTL Diagram

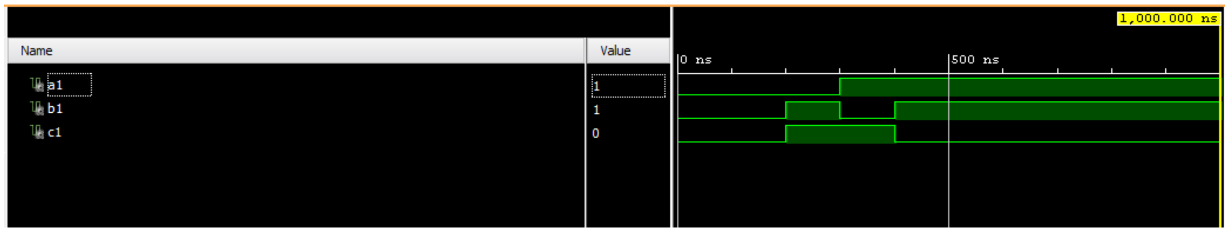


3.TBW Code

```
entity xord_tbw is
--  Port ( );
end xord_tbw;
architecture Behavioral of xord_tbw is
component xord 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: xord 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;
```

4.Waveform

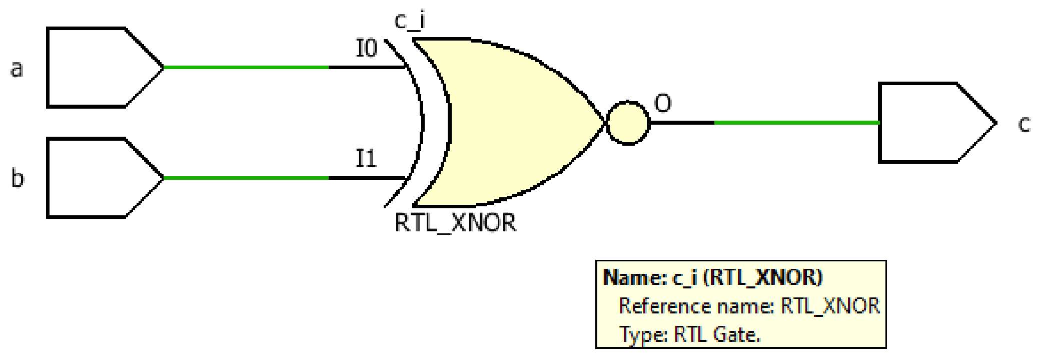


XNOR Gate (Dataflow Model)

1.VHDL Code

```
entity xnord is
   Port ( a : in STD_LOGIC;
          b : in STD_LOGIC;
          c : out STD_LOGIC);
end xnord;
architecture Dataflow of xnord is
begin
c<=a xnor b;
end Dataflow;
```

2.RTL Design

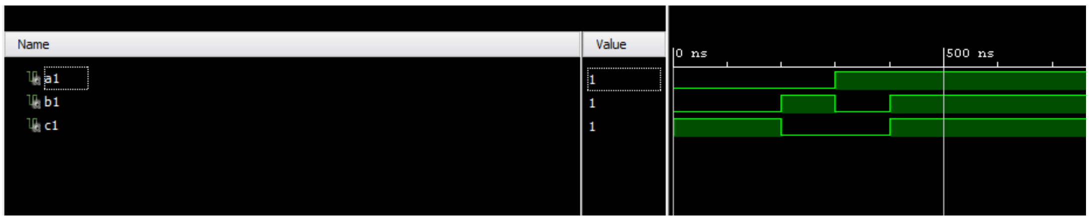


3.TBW Code

```
entity xnord_tbw is
-- Port ( );
end xnord_tbw;
architecture Behavioral of xnord_tbw is
component xnord 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: xnord 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;
```

4.Waveform



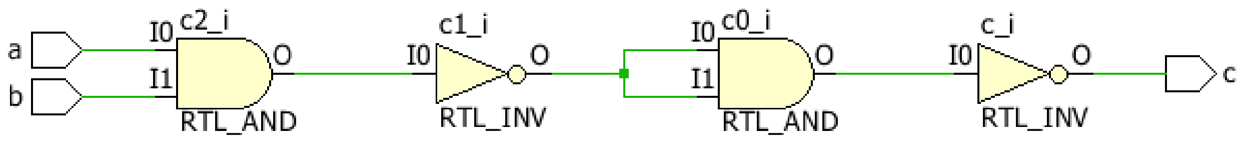
DAY 2

AND Gate using NAND Gates (Dataflow Model)

1. VHDL Code

```
entity and_nandd is
   Port ( a : in STD_LOGIC;
          b : in STD_LOGIC;
          c : out STD_LOGIC);
end and_nandd;
architecture Dataflow of and_nandd is
begin
c<=(a nand b) nand (a nand b);
end Dataflow;
```

2. RTL Design

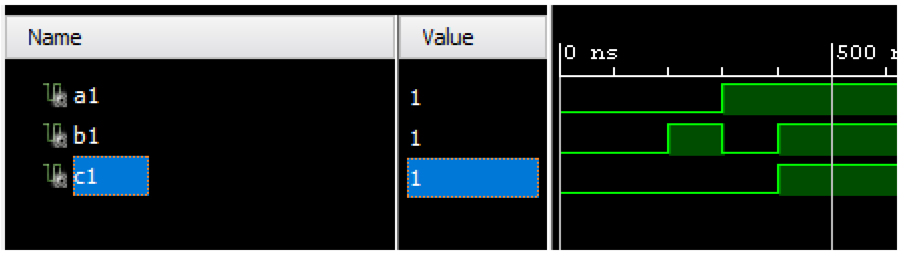


3.TBW Code

```
entity and_nandtbw is
-- Port ( );
end and_nandtbw;
architecture Behavioral of and_nandtbw is
component and_nandd 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_nandd 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;
```

4. Waveform

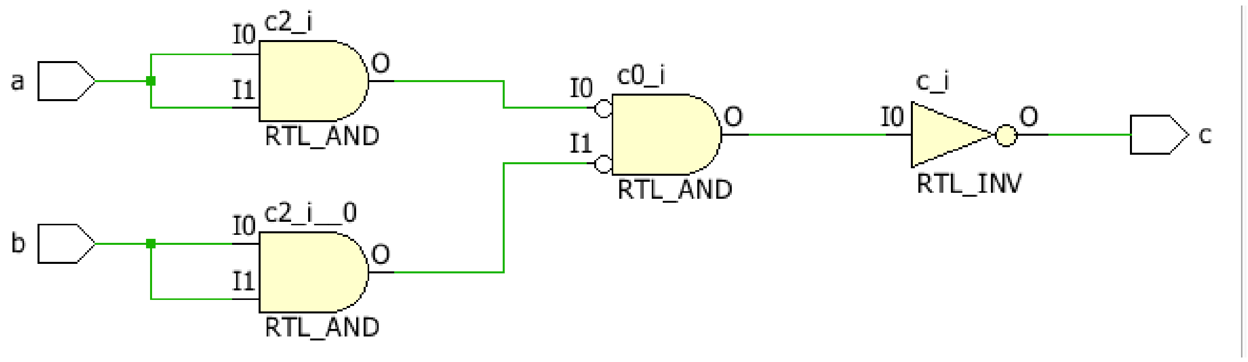


OR Gate using NAND Gates (Dataflow Model)

1. **VHDL Code**

```
entity or_nandd is
   Port ( a : in STD_LOGIC;
          b : in STD_LOGIC;
          c : out STD_LOGIC);
end or_nandd;
architecture Dataflow of or_nandd is
begin
c<=(a nand a) nand (b nand b);
end Dataflow;
```

2. **RTL Design**

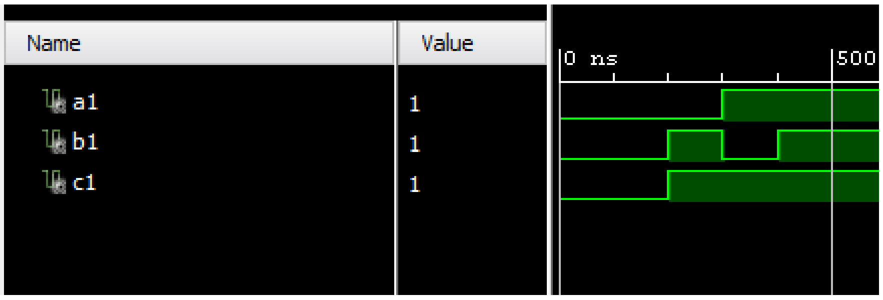


3. **TBW Code**

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

architecture Behavioral of or_nandtbw is

component or_nandd 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_nandd 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;
```

4. **Waveform**



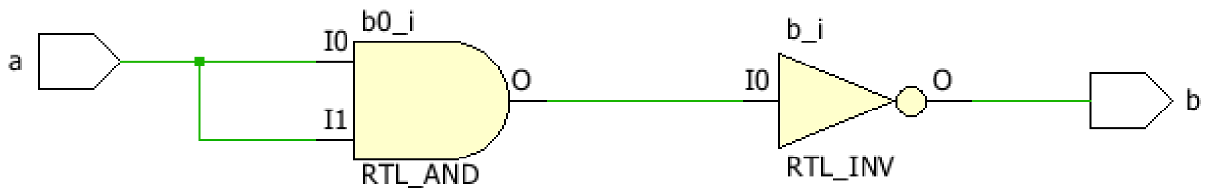
Name	Value
a1	1
b1	1
c1	1

NOT Gate using NAND Gates (Dataflow Model)

1. **VHDL Code**

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

architecture Dataflow of not_nandd is
begin
b<=a nand a;
end Dataflow;
```

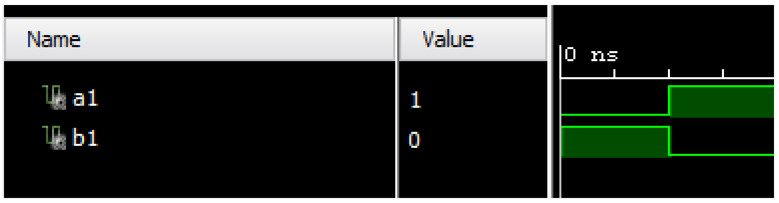
2. **RTL Design**



3. **TBW Code**

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

4. Waveform

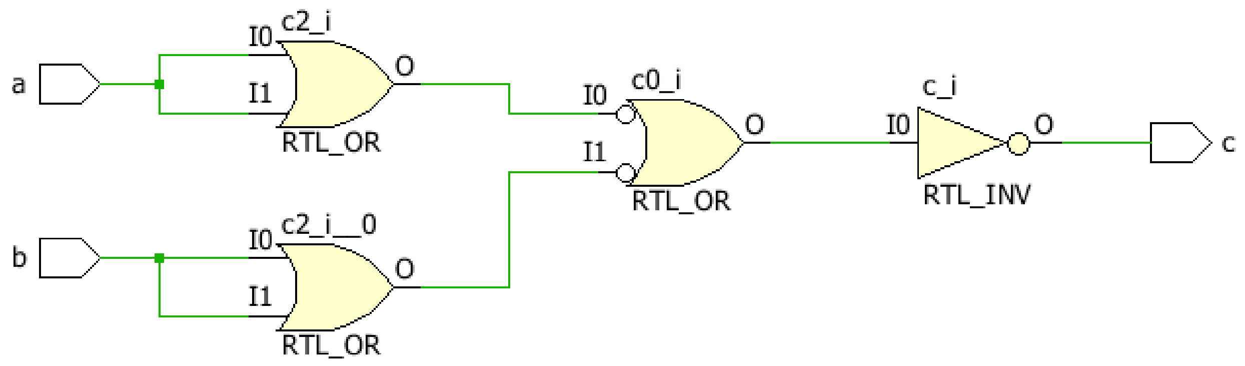


AND Gate using NOR Gates (Dataflow Model)

1. **VHDL Code**

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

architecture Dataflow of and_nord is
begin
c<=(a nor a) nor (b nor b);
end Dataflow;
```

2. **RTL Design**

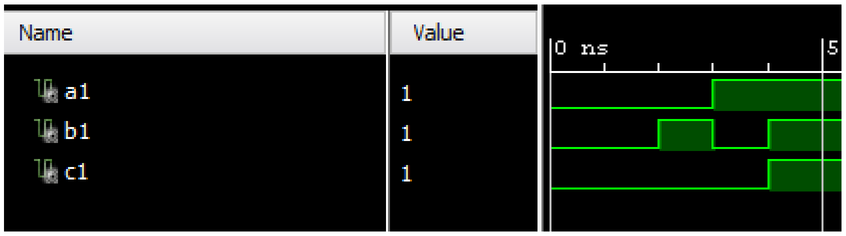


3. **TBW Code**

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

architecture Behavioral of and_nortbw is
component and_nord 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_nord 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;
```

4. Waveform

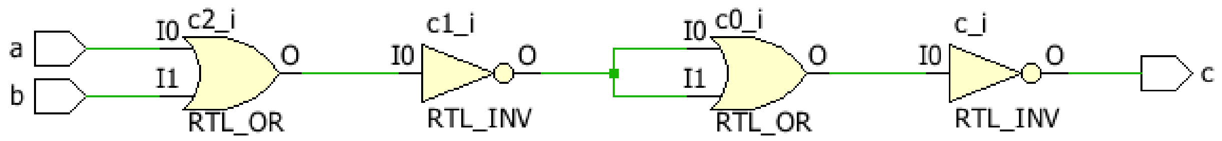


OR Gate using NOR Gates (Dataflow Model)

1. **VHDL Code**

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

architecture Dataflow of or_nord is
begin
c<=(a nor b) nor ( a nor b);
end Dataflow;
```

2. **RTL Design**

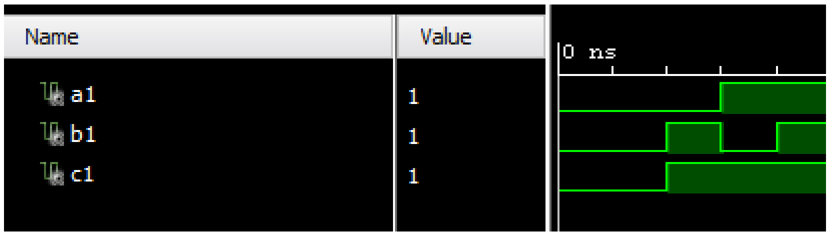


3. **TBW Code**

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

architecture Behavioral of or_nortbw is
component or_nord 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_nord 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;
```

4. Waveform



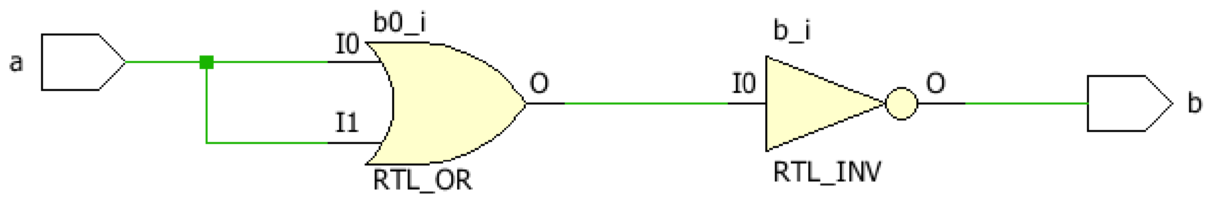
Name	Value
a1	1
b1	1
c1	1

NOT Gate using NOR Gates (Dataflow Model)

1. **VHDL Code**

```
entity not_nord is
   Port ( a : in STD_LOGIC;
          b : out STD_LOGIC);
end not_nord;
architecture Dataflow of not_nord is
begin
b<=a nor a;
end Dataflow;
```

2. **RTL Design**

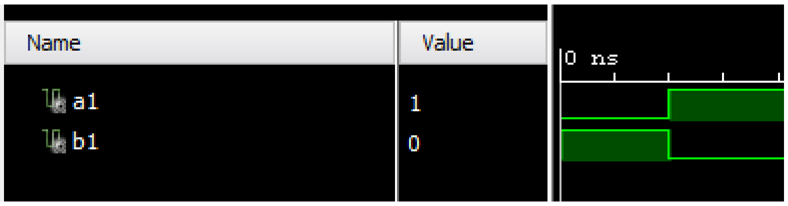


3. **TBW Code**

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

architecture Behavioral of not_nortbw is
component not_nord is
   Port ( a : in STD_LOGIC;
          b : out STD_LOGIC);
end component;
signal a1:STD_LOGIC:='0';
signal b1:STD_LOGIC;
begin
uut: not_nord port map (a=>a1,b=>b1);
stim_proc: process
begin
```

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

4. Waveform



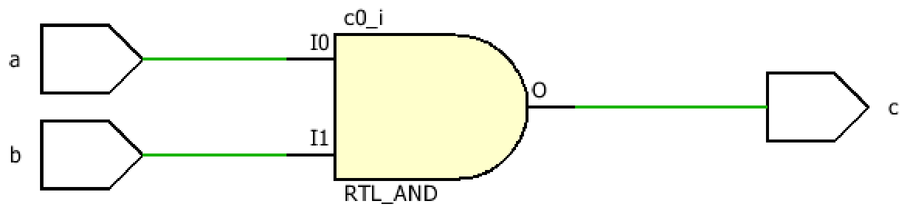
DAY 3

AND Gate (Behavioral Model)

1.VHDL Code

```
entity and_bv is
   Port ( a : in STD_LOGIC;
          b : in STD_LOGIC;
          c : out STD_LOGIC);
end and_bv;
architecture Behavioral of and_bv is
begin
process(a,b)
begin
if(a='1'and b='1')then
c<='1';
else
c<='0';
end if;
end process;
end Behavioral;
```

2.RTL Design

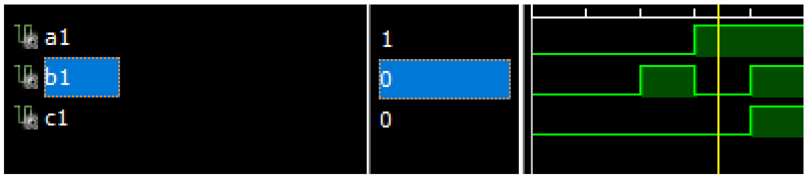


3.TBW Code

```
entity and_bvtbw is
--  Port ( );
end and_bvtbw;
architecture Behavioral of and_bvtbw is
component and_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: and_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;
```

4.Waveform

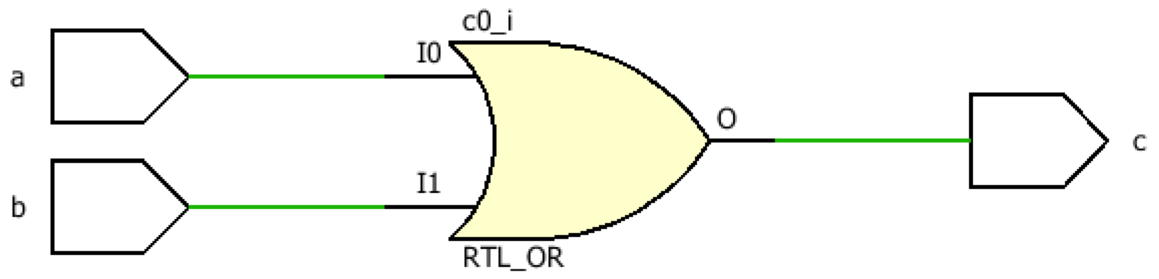


OR Gate (Behavioral Model)

1.VHDL Code

```
entity or_bv is
   Port ( a : in STD_LOGIC;
         b : in STD_LOGIC;
         c : out STD_LOGIC);
end or_bv;
architecture Behavioral of or_bv is
begin
process(a,b)
begin
if(a='1'or b='1')then
c<='1';
else
c<='0';
end if;
end process;
end Behavioral;
```

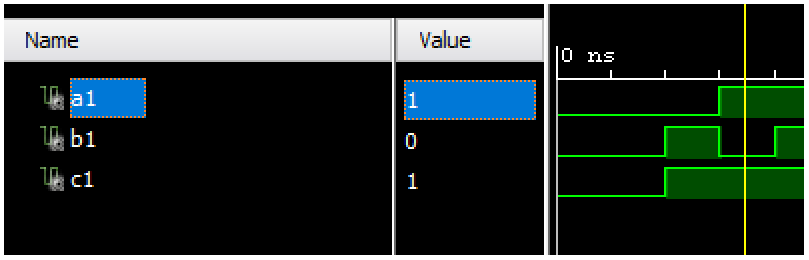
2.RTL Design



3.TBW Code

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

architecture Behavioral of or_bvtbw is
component or_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: or_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;
```

4.Waveform

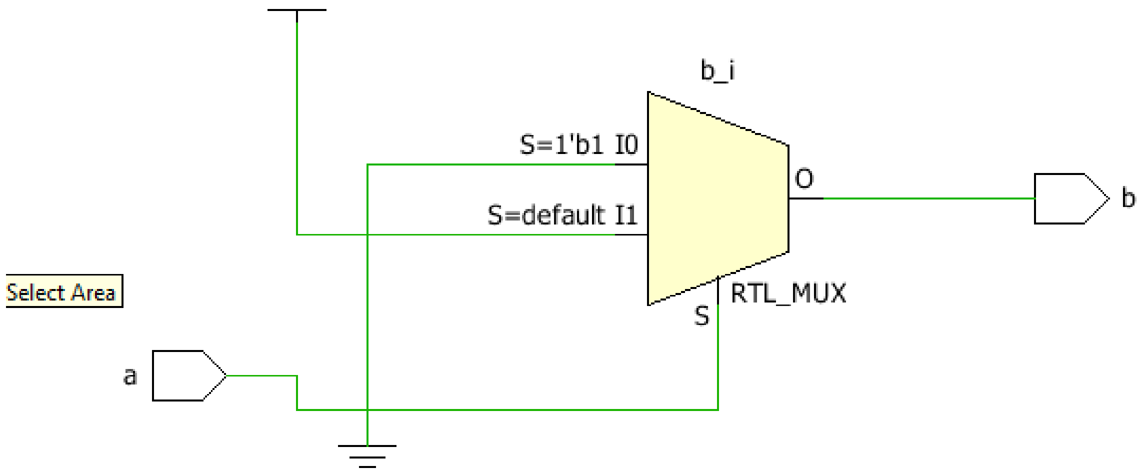


NOT Gate (Behavioral Model)

1.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='1')then
b<='0';
else
b<='1';
end if;
end process;
```

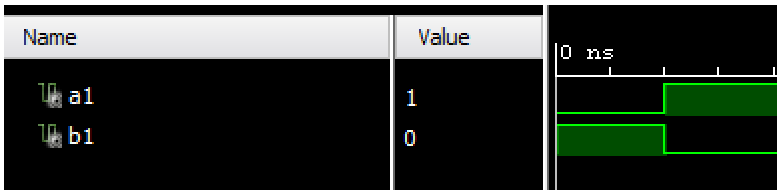
2.RTL Design



3.TBW Code

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

architecture Behavioral of not_bvtbw 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 100 ns;
a1<='0';

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

4.Waveform



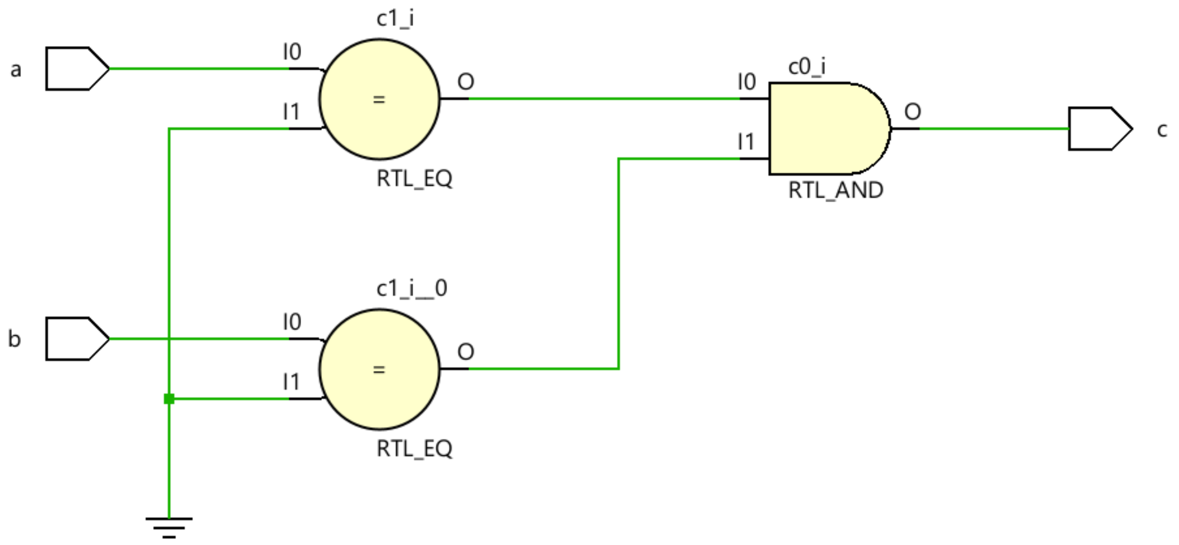
Name	Value
a1	1
b1	0

NOR Gate (Behavioral Model)

1.VHDL Code

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

2.RTL Design

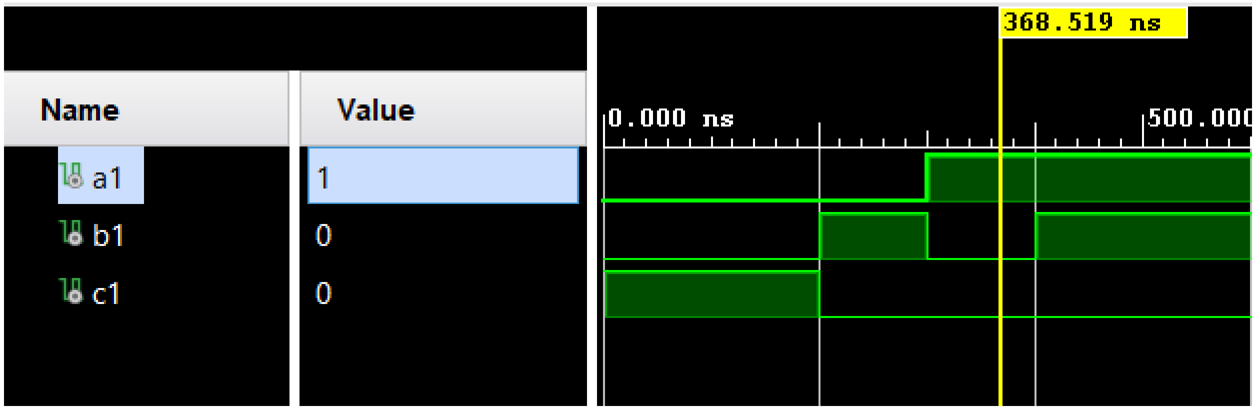


3.TBW Code

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

```
architecture Behavioral of nor_bvtbw 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;
```

4.Waveform

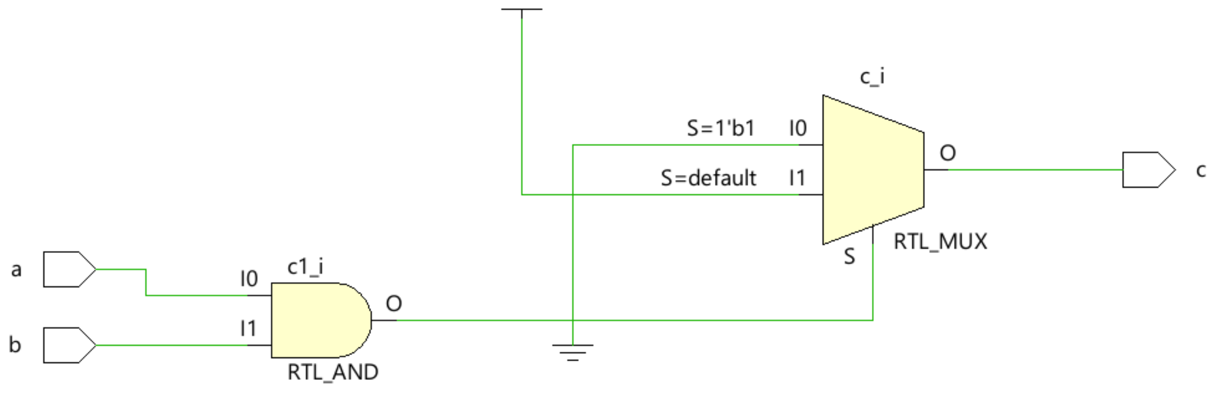


NAND Gate (Behavioral Model)

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;
```

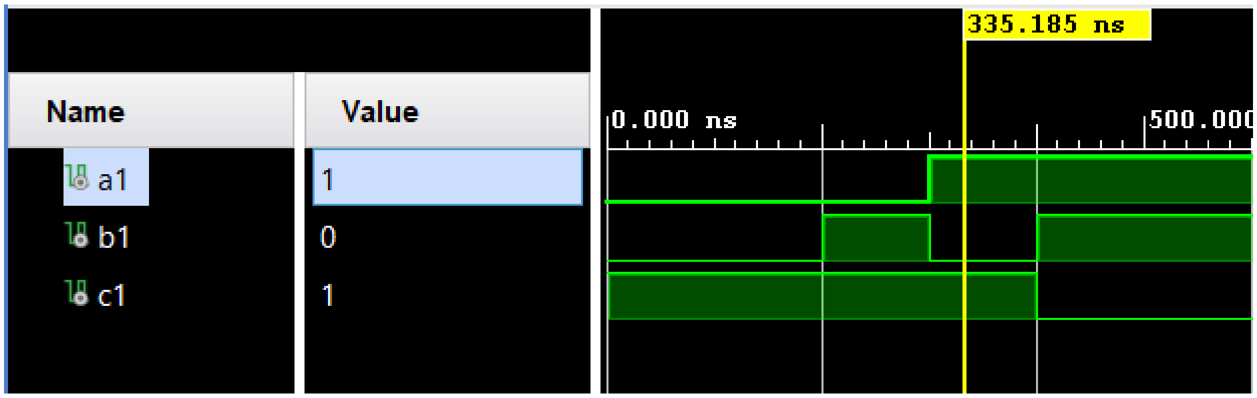
2.RTL Design



3.TBW Code

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

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;
```

4.Waveform

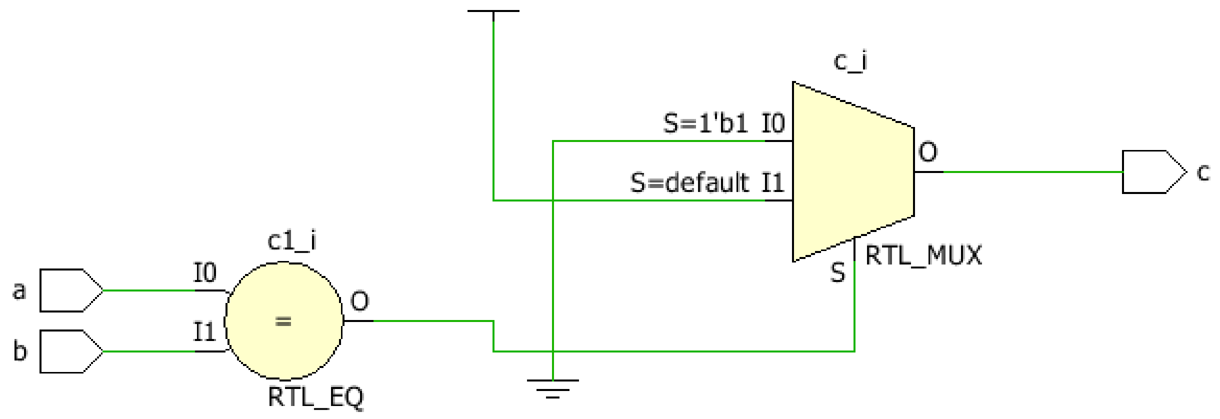


XOR Gate (Behavioral Model)

1.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;
```

2.RTL Design

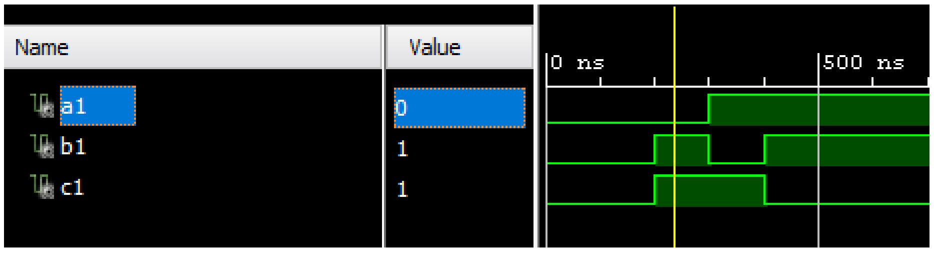


3.TBW Code

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

```
architecture Behavioral of xor_bvtbw 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;
```

4.Waveform

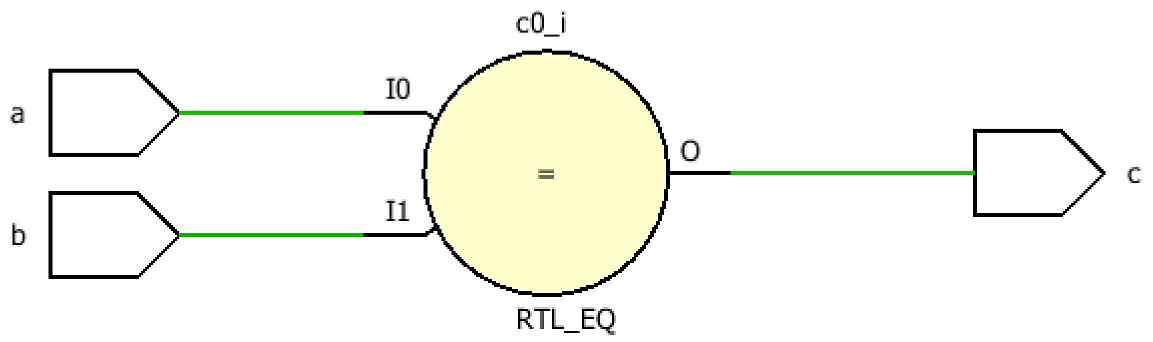


XNOR Gate (Behavioral Model)

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;
```

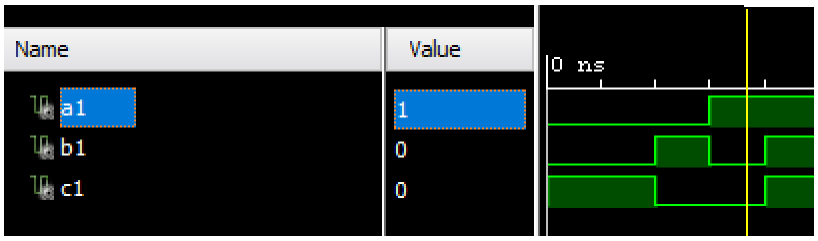
2.RTL Design



3.TBW Code

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

architecture Behavioral of xnor_bvtbw 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;
```

4.Waveform



Name	Value
a1	1
b1	0
c1	0

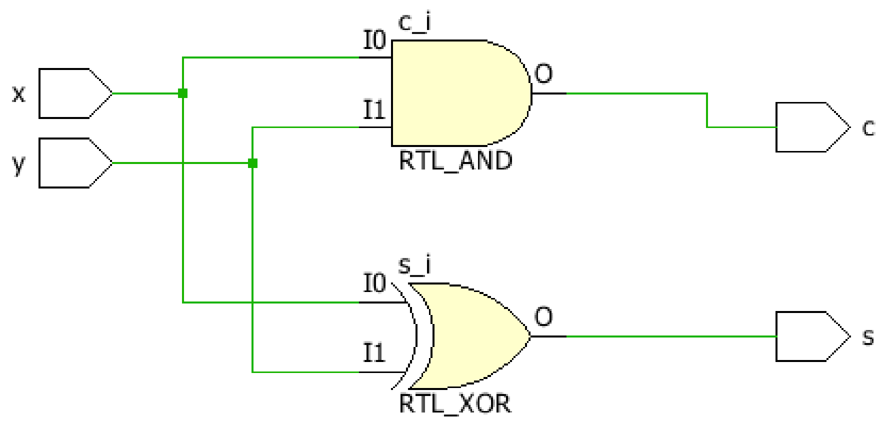
DAY 4

Half-Adder (Dataflow Model)

1.VHDL Code

```
entity half_adder_df is
   Port ( x : in STD_LOGIC;
        y : in STD_LOGIC;
        s : out STD_LOGIC;
        c : out STD_LOGIC);
end half_adder_df;
architecture Dataflow of half_adder_df is
begin
s<= x xor y;
c<= x and y;
end Dataflow;
```

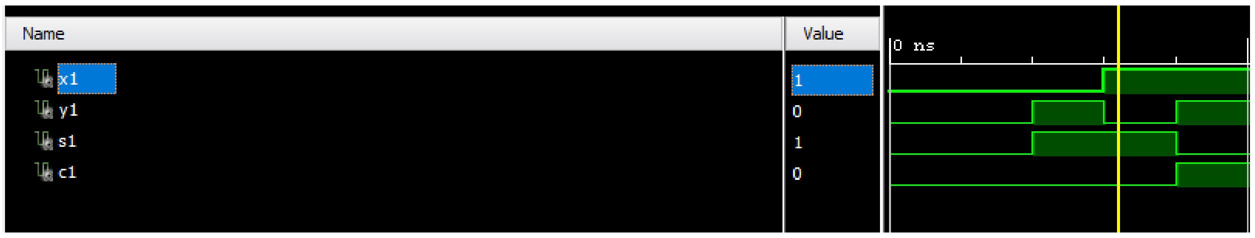
2.RTL Design



3.TBW Code

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

architecture Behavioral of half_adderdf_tbw is
component half_adderdf is
   Port ( x : in STD_LOGIC;
        y : in STD_LOGIC;
        s : out STD_LOGIC;
        c : out STD_LOGIC);
end component;
signal x1: STD_LOGIC:='0';
signal y1: STD_LOGIC:='0';
signal s1: STD_LOGIC;
signal c1: STD_LOGIC;
begin
uut: half_adderdf port map(x=>x1,y=>y1,s=>s1,c=>c1);
stim_proc: process
begin
wait for 100 ns;
x1<='0';
y1<='0';
wait for 100 ns;
x1<='0';
y1<='1';
wait for 100 ns;
x1<='1';
y1<='0';
wait for 100 ns;
x1<='1';
y1<='1';
wait;
end process;
end Behavioral;
```

4.Waveform

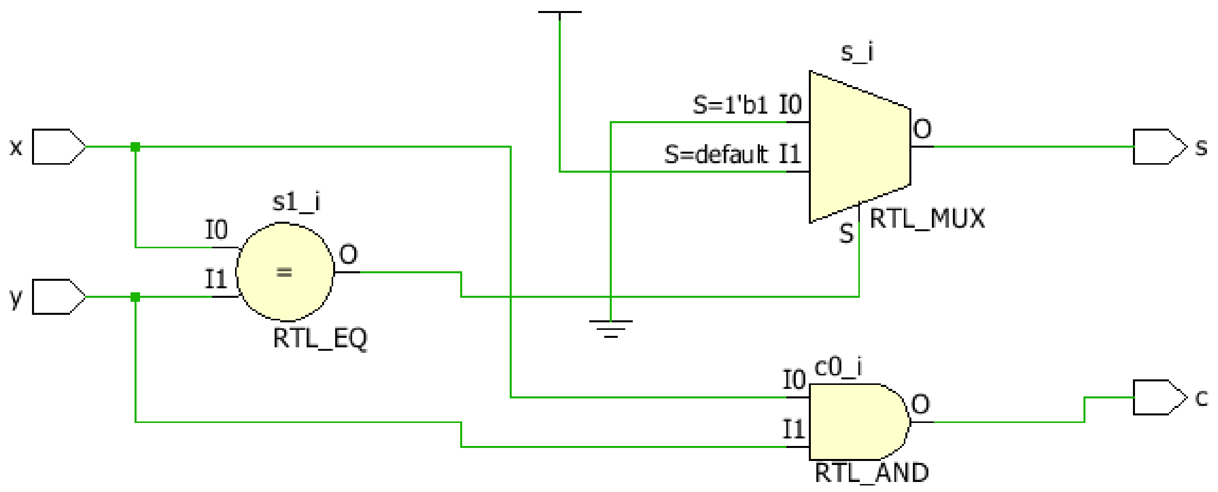


Half-Adder (Behavioral Model)

1.VHDL Code

```
entity half_addf is
   Port ( x : in STD_LOGIC;
        y : in STD_LOGIC;
        s : out STD_LOGIC;
        c : out STD_LOGIC);
end half_addf;

architecture Behavioral of half_addf is
begin
process(x,y)
begin
if(x=y)then
s<='0';
else
s<='1';
end if;
if(x='1' and y='1')then
c<='1';
else
c<='0';
end if;
end process;
end Behavioral;
```

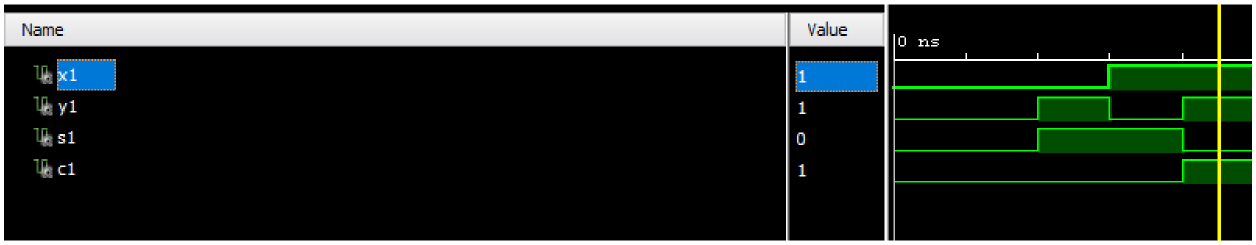
2.RTL Design



3.TBW Code

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

architecture Behavioral of half_addf_tbw is
component half_addf is
   Port ( x : in STD_LOGIC;
        y : in STD_LOGIC;
        s : out STD_LOGIC;
        c : out STD_LOGIC);
end component;
signal x1: STD_LOGIC:='0';
signal y1: STD_LOGIC:='0';
signal s1: STD_LOGIC;
signal c1: STD_LOGIC;
begin
uut: half_addf port map(x=>x1,y=>y1,s=>s1,c=>c1);
stim_proc: process
begin
wait for 100 ns;
x1<='0';
y1<='0';
wait for 100 ns;
x1<='0';
y1<='1';
wait for 100 ns;
x1<='1';
y1<='0';
wait for 100 ns;
x1<='1';
y1<='1';
wait;
end process;
end Behavioral;
```

4.Waveform

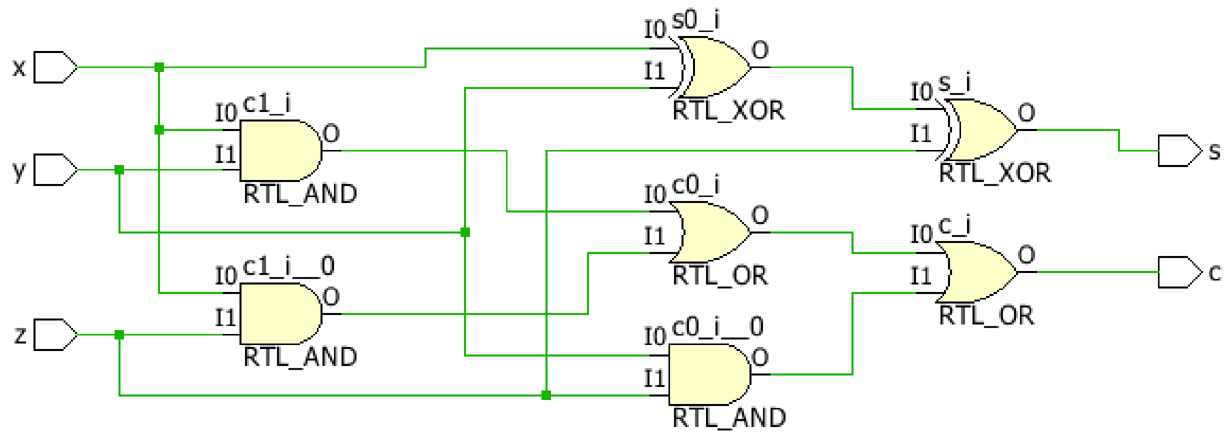


Full-Adder (Dataflow Model)

1.VHDL Code

```
entity full_adder_df is
   Port ( x : in STD_LOGIC;
        y : in STD_LOGIC;
        z : in STD_LOGIC;
        s : out STD_LOGIC;
        c : out STD_LOGIC);
end full_adder_df;

architecture Dataflow of full_adder_df is

begin
s<=(x xor y) xor z;
c<=( x and y) or ( x and z) or ( y and z);

end Dataflow;
```

2.RTL Design

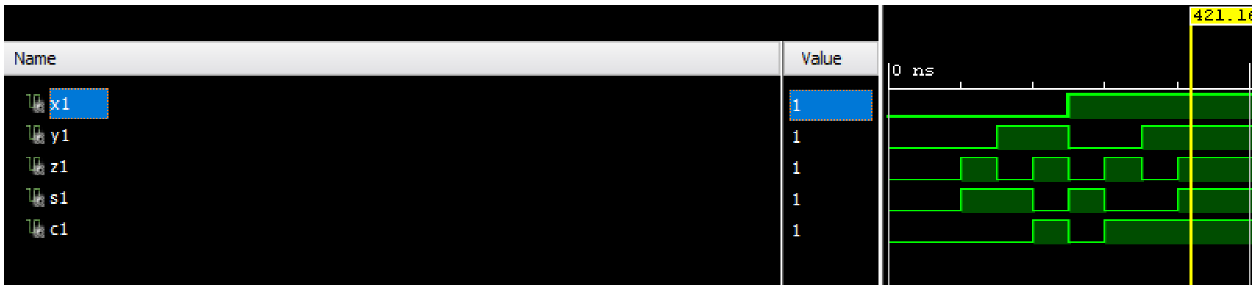


3.TBW Code

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

architecture Behavioral of full_adder_dftbw is
component full_adder_df is
   Port ( x : in STD_LOGIC;
        y : in STD_LOGIC;
        z : in STD_LOGIC;
        s : out STD_LOGIC;
        c : out STD_LOGIC);
end component;
signal x1: STD_LOGIC:='0';
signal y1: STD_LOGIC:='0';
signal z1: STD_LOGIC:='0';
signal s1: STD_LOGIC;
signal c1: STD_LOGIC;
begin
uut: full_adder_df port map(x=>x1,y=>y1,z=>z1,s=>s1,c=>c1);
stim_proc: process
begin
wait for 50 ns;
x1<='0';
y1<='0';
z1<='0';
wait for 50 ns;
x1<='0';
y1<='0';
z1<='1';
wait for 50 ns;
x1<='0';
y1<='1';
z1<='0';
wait for 50 ns;
x1<='0';
y1<='1';
z1<='1';
wait for 50 ns;
x1<='1';
y1<='0';
z1<='0';
wait for 50 ns;
x1<='1';
y1<='0';
z1<='1';
wait for 50 ns;
x1<='1';
y1<='1';
z1<='0';
wait for 50 ns;
x1<='1';
```

y1<='1';
z1<='1';
wait;
end process;
end Behavioral;

4.Waveform



Name	Value
x1	1
y1	1
z1	1
s1	1
c1	1

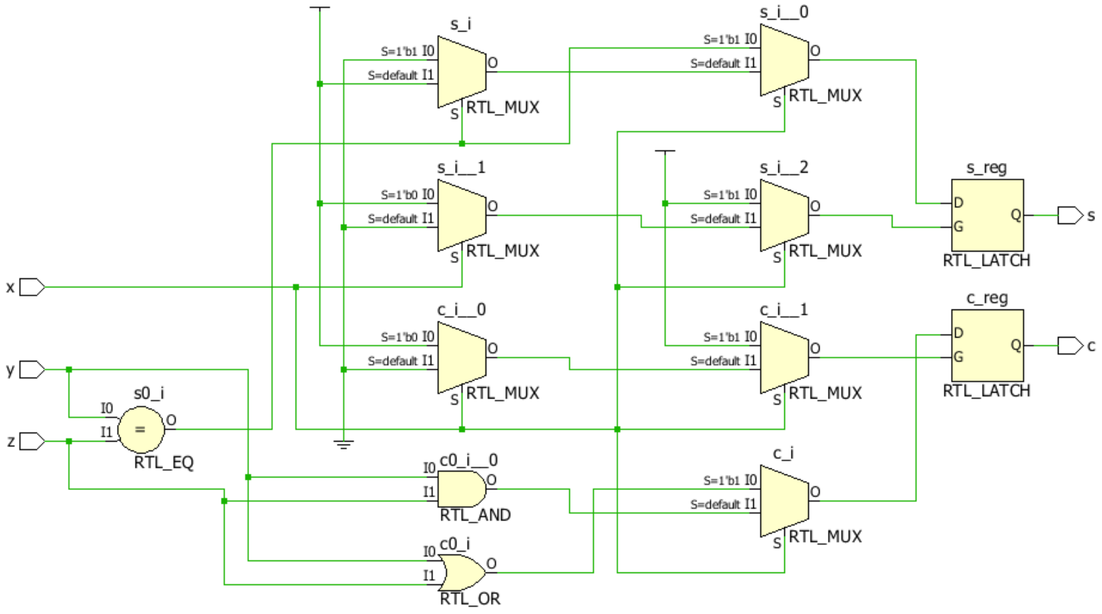
Full-Adder (Behavioral Model)

1.VHDL Code

```
entity full_adder_bv is
   Port ( x : in STD_LOGIC;
        y : in STD_LOGIC;
        z : in STD_LOGIC;
        s : out STD_LOGIC;
        c : out STD_LOGIC);
end full_adder_bv;

architecture Behavioral of full_adder_bv is

begin
process(x,y,z)
begin
if(x='0')then
if(y=z)then
s<='0';
else
s<='1';
end if;
if(y='1' and z='1')then
c<='1';
else
c<='0';
end if;
end if;

--second half
if(x='1')then
if(y=z)then
s<='1';
else
s<='0';
end if;
if(y='1' or z='1')then
c<='1';
else
c<='0';
```

end if;
end if;
end process;
end Behavioral;

2.RTL Design



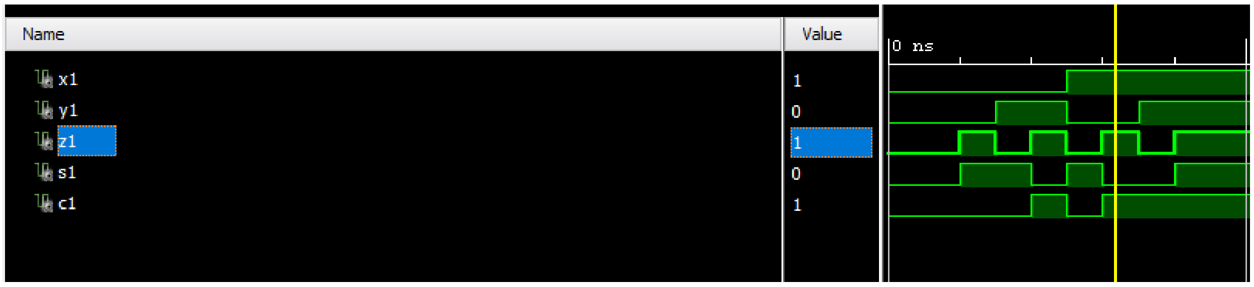
3.TBW Code

entity full_adder_bvtbw is
-- Port ( );
end full_adder_bvtbw;

architecture Behavioral of full_adder_bvtbw is
component full_adder_bv is
  Port (x : in STD_LOGIC;
       y : in STD_LOGIC;
       z : in STD_LOGIC;
       s : out STD_LOGIC;
       c : out STD_LOGIC);
end component;
signal x1: STD_LOGIC:='0';
signal y1: STD_LOGIC:='0';
signal z1: STD_LOGIC:='0';
signal s1: STD_LOGIC;
signal c1: STD_LOGIC;
begin
uut: full_adder_bv port map(x=>x1,y=>y1,z=>z1,s=>s1,c=>c1);
stim_proc: process
begin
wait for 50 ns;

```
x1<='0';
y1<='0';
z1<='0';
wait for 50 ns;
x1<='0';
y1<='0';
z1<='1';
wait for 50 ns;
x1<='0';
y1<='1';
z1<='0';
wait for 50 ns;
x1<='0';
y1<='1';
z1<='1';
wait for 50 ns;
x1<='1';
y1<='0';
z1<='0';
wait for 50 ns;
x1<='1';
y1<='0';
z1<='1';
wait for 50 ns;
x1<='1';
y1<='1';
z1<='0';
wait for 50 ns;
x1<='1';
y1<='1';
z1<='1';
wait;
end process;
end Behavioral;
```

4.Waveform



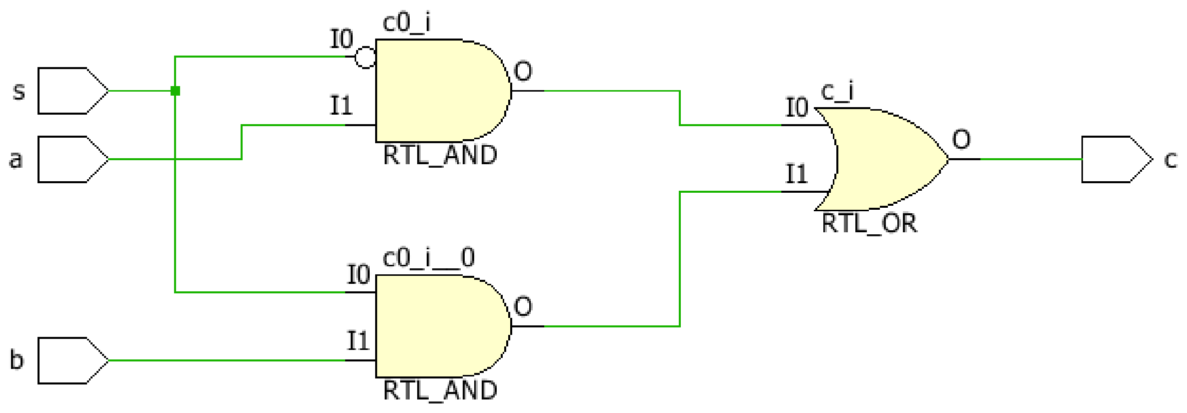
```
Name    Value
x1      1
y1      0
z1      1
s1      0
c1      1
```

2:1 MUX (Dataflow Model)

1.VHDL Code

```
entity mux2_1 is
   Port ( a : in STD_LOGIC;
         b : in STD_LOGIC;
         c : out STD_LOGIC;
         s : in STD_LOGIC);
end mux2_1;

architecture Dataflow of mux2_1 is

begin
c<=((not s) and a) or ( s and b);

end Dataflow;
```

2.RTL Design

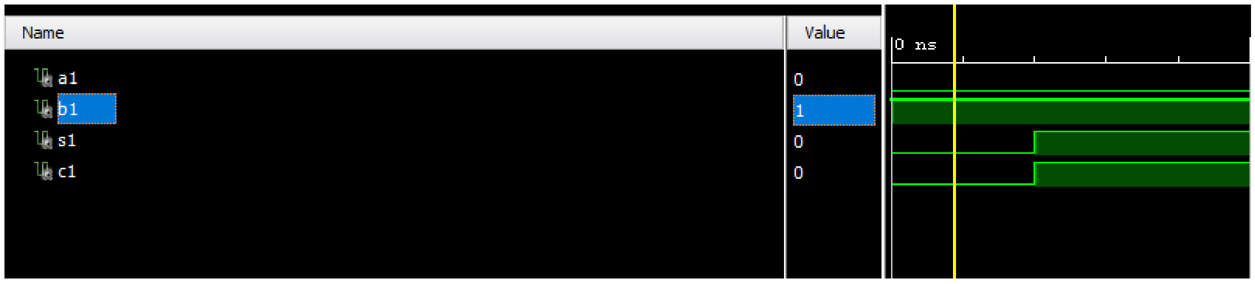


3.TBW Code

```
entity mux2_1_tbw is
-- Port ( );
end mux2_1_tbw;

architecture Behavioral of mux2_1_tbw is
component mux2_1 is
   Port ( a : in STD_LOGIC;
         b : in STD_LOGIC;
         c : out STD_LOGIC;
         s : in STD_LOGIC);
```

```
end component;
signal a1: STD_LOGIC:='0';
signal b1: STD_LOGIC:='1';
signal s1: STD_LOGIC:='0';
signal c1: STD_LOGIC;
begin
uut: mux2_1 port map(a=>a1,b=>b1,c=>c1,s=>s1);
stim_proc: process
begin
wait for 100ns;
s1<='0';

wait for 100ns;
s1<='1';

wait;
end process;
end Behavioral;
```

4.Waveform

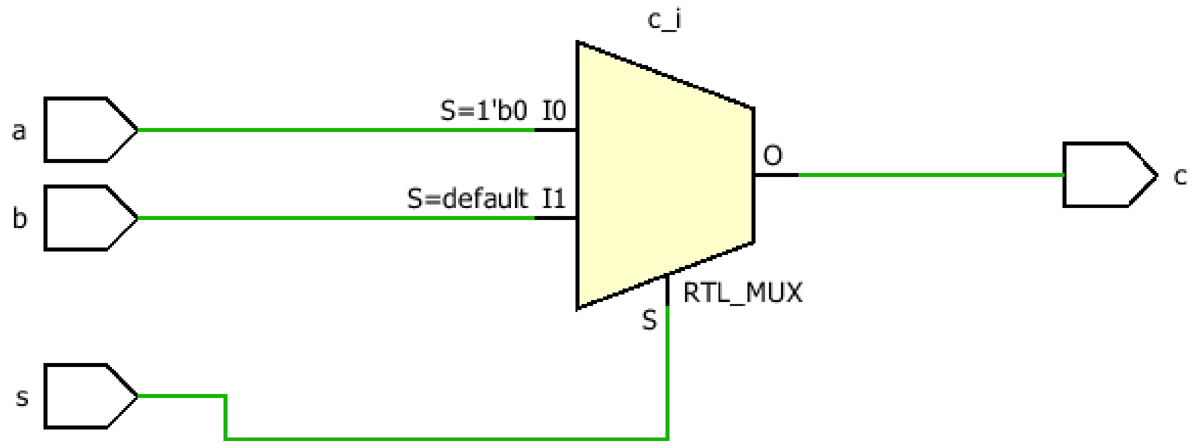


2:1 MUX (Behavioral Model)

1.VHDL Code

```
entity mux2_1_bv is
   Port ( a : in STD_LOGIC;
         b : in STD_LOGIC;
         c : out STD_LOGIC;
         s : in STD_LOGIC);
end mux2_1_bv;

architecture Behavioral of mux2_1_bv is

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

2.RTL Design

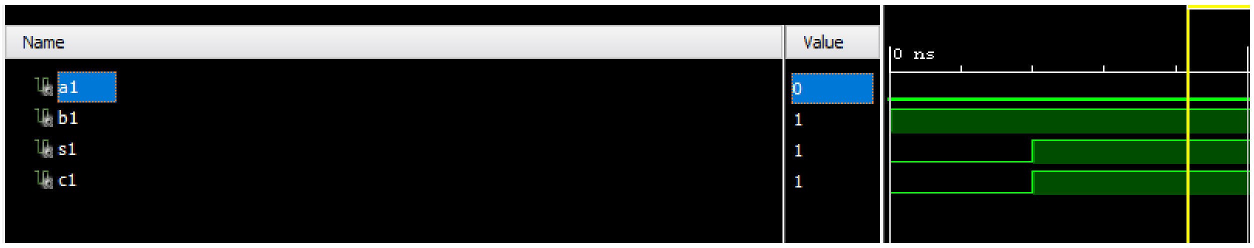


3.TBW Code

```
entity mux2_1_bvtbw is
-- Port ( );
end mux2_1_bvtbw;
```

```
architecture Behavioral of mux2_1_bvtbw is
component mux2_1_bv is
  Port ( a : in STD_LOGIC;
       b : in STD_LOGIC;
       c : out STD_LOGIC;
       s : in STD_LOGIC);
end component;
signal a1: STD_LOGIC:='0';
signal b1: STD_LOGIC:='1';
signal s1: STD_LOGIC:='0';
signal c1: STD_LOGIC;
begin
uut: mux2_1_bv port map(a=>a1,b=>b1,c=>c1,s=>s1);
stim_proc: process
begin
wait for 100ns;
s1<='0';

wait for 100ns;
s1<='1';

wait;
end process;
end Behavioral;
```

4.Waveform



```
Name    Value
a1      0
b1      1
s1      1
c1      1
```

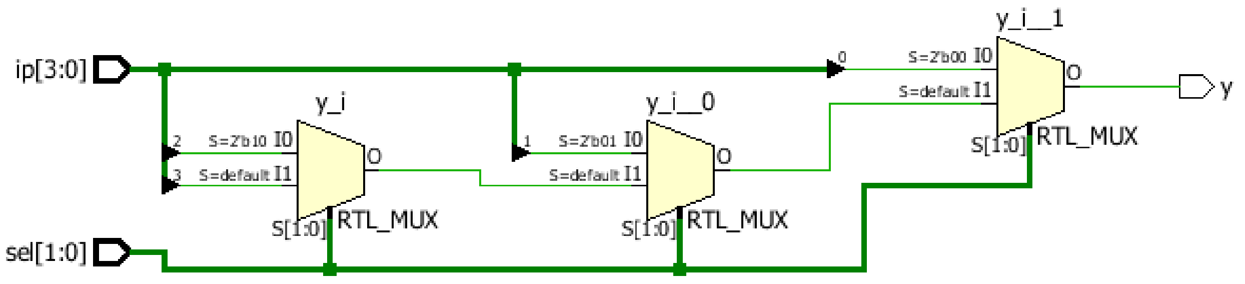
DAY 5

4:1 MUX (Dataflow Model)

1.VHDL Code

```
entity mux4_1_df is
   Port ( ip : in STD_LOGIC_VECTOR (3 downto 0);
        sel : in STD_LOGIC_VECTOR (1 downto 0);
        y : out STD_LOGIC);
end mux4_1_df;
architecture Dataflow of mux4_1_df is
begin
y<=ip(0) when sel="00" else
ip(1) when sel="01" else
ip(2) when sel="10" else
ip(3);
end Dataflow;
```

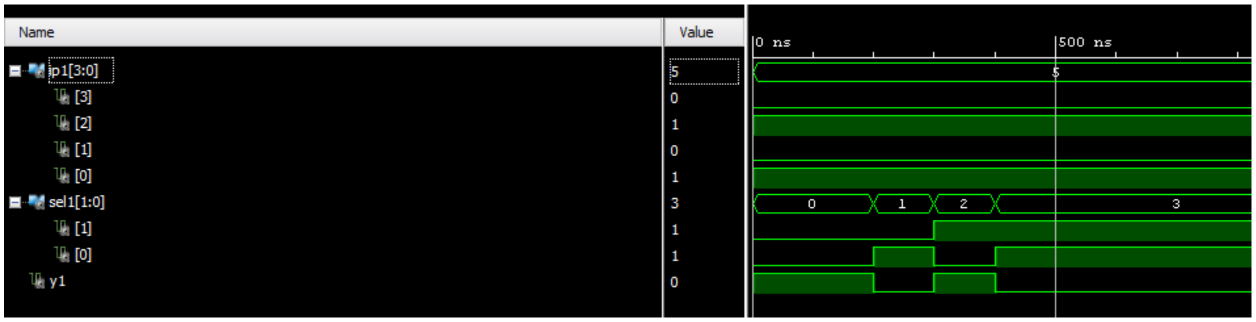
2.RTL Design



3.TBW Code

```
entity mux4_1_dftbw is
-- Port ( );
end mux4_1_dftbw;

architecture Behavioral of mux4_1_dftbw is
component mux4_1_df is
   Port ( ip : in STD_LOGIC_VECTOR (3 downto 0);
        sel : in STD_LOGIC_VECTOR (1 downto 0);
        y : out STD_LOGIC);
end component;
Signal ip1: STD_LOGIC_VECTOR(3 downto 0):="0101";
Signal sel1: STD_LOGIC_VECTOR(1 downto 0):="00";
Signal y1:STD_LOGIC;
begin
uut: mux4_1_df port map(ip=>ip1,sel=>sel1,y=>y1);
stim_proc: process
begin
wait for 100ns;
sel1<="00";
wait for 100ns;
sel1<="01";
wait for 100ns;
sel1<="10";
wait for 100ns;
sel1<="11";
wait;
end process;
end Behavioral;
```

4.Waveform

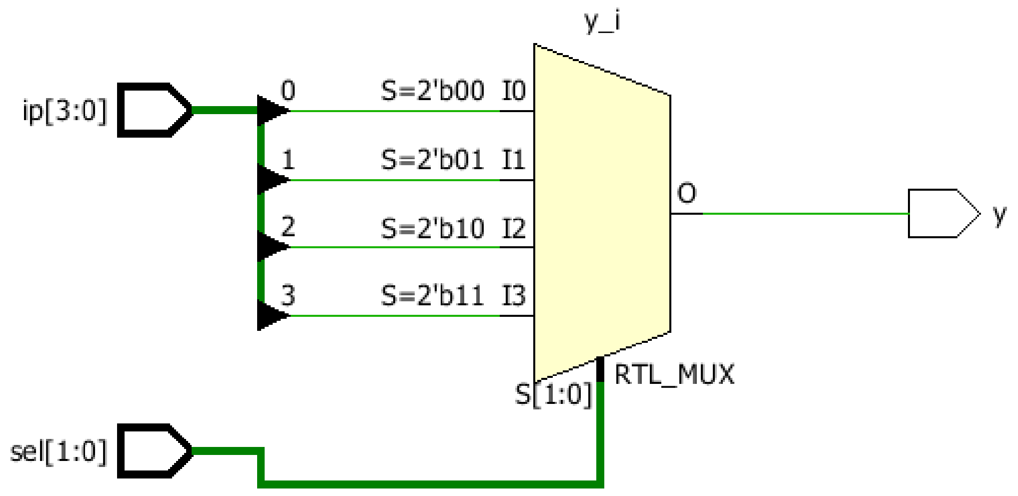


4:1 MUX (Behavioral Model)

1.VHDL Code

```
entity mux4_1_bv is
   Port ( ip : in STD_LOGIC_VECTOR (3 downto 0);
        sel : in STD_LOGIC_VECTOR (1 downto 0);
        y : out STD_LOGIC);
end mux4_1_bv;

architecture Behavioral of mux4_1_bv is

begin
process(ip,sel)
begin
case sel is
when "00" => y <=ip(0);
when "01" => y <=ip(1);
when "10" => y <=ip(2);
when "11" => y <=ip(3);
when others=>NULL;
end case;
end process;
end Behavioral;
```

2.RTL Design

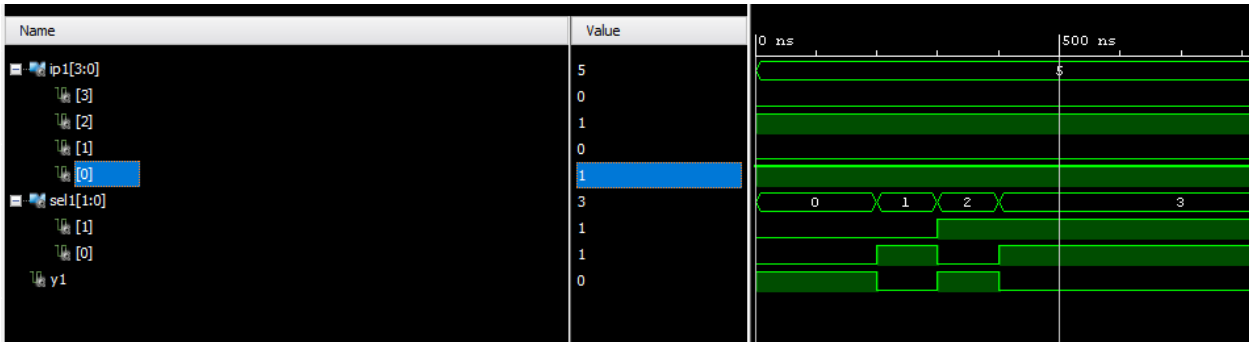


3.TBW Code

```
entity mux4_1_bvtbw is
```

```
--  Port ( );
end mux4_1_bvtbw;

architecture Behavioral of mux4_1_bvtbw is
component mux4_1_bv is
   Port ( ip : in STD_LOGIC_VECTOR (3 downto 0);
        sel : in STD_LOGIC_VECTOR (1 downto 0);
        y : out STD_LOGIC);
end component;
Signal ip1: STD_LOGIC_VECTOR(3 downto 0):="0101";
Signal sel1: STD_LOGIC_VECTOR(1 downto 0):="00";
Signal y1:STD_LOGIC;
begin
uut: mux4_1_bv port map(ip=>ip1,sel=>sel1,y=>y1);
stim_proc: process
begin
wait for 100ns;
sel1<="00";
wait for 100ns;
sel1<="01";
wait for 100ns;
sel1<="10";
wait for 100ns;
sel1<="11";
wait;
end process;
end Behavioral;
```

4.Waveform

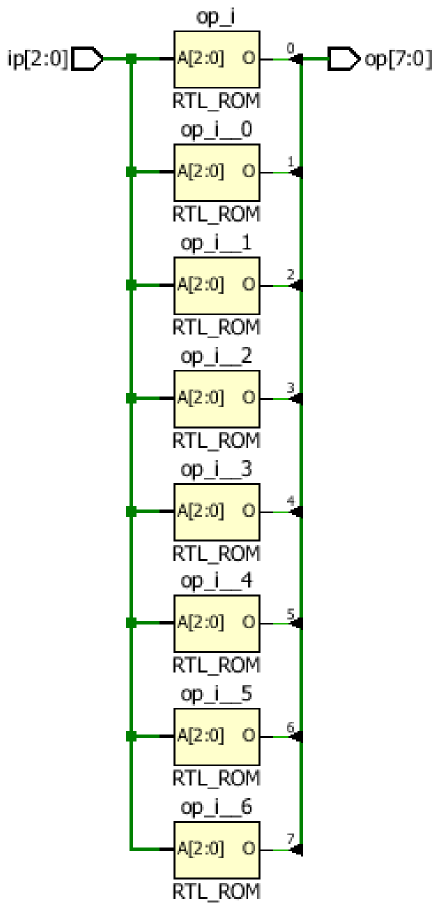


3:8 decoder (Dataflow Model)

1.VHDL Code

```
entity decoder3_8_df is
   Port ( ip : in STD_LOGIC_VECTOR (2 downto 0);
          op : out STD_LOGIC_VECTOR (7 downto 0));
end decoder3_8_df;
architecture Dataflow of decoder3_8_df is
begin
op(0)<='1' when ip="000" else '0';
op(1)<='1' when ip="001" else '0';
op(2)<='1' when ip="010" else '0';
op(3)<='1' when ip="011" else '0';
op(4)<='1' when ip="100" else '0';
op(5)<='1' when ip="101" else '0';
op(6)<='1' when ip="110" else '0';
op(7)<='1'when ip="111" else '0';
end Dataflow;
```

2.RTL Design

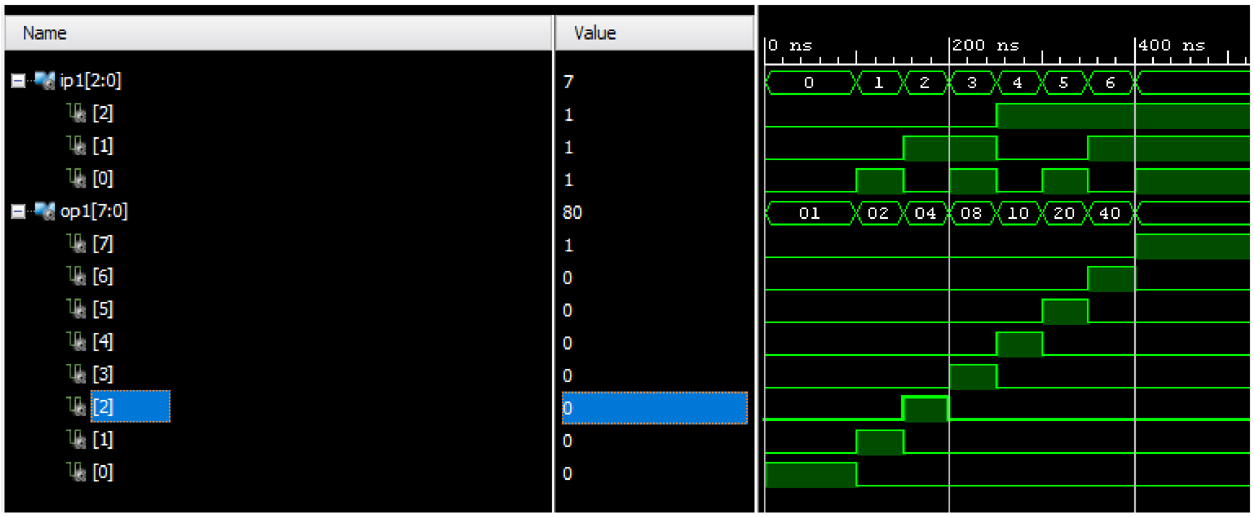


3.TBW Code

```
entity decoder3_8_df_tbw is
--  Port ( );
end decoder3_8_df_tbw;

architecture Behavioral of decoder3_8_df_tbw is
component decoder3_8_df is
   Port ( ip : in STD_LOGIC_VECTOR (2 downto 0);
        op : out STD_LOGIC_VECTOR (7 downto 0));
end component;

signal ip1:STD_LOGIC_VECTOR(2 downto 0):="000";
signal op1:STD_LOGIC_VECTOR(7 downto 0);
begin
uut: decoder3_8_df port map(ip=>ip1,op=>op1);
stim_proc: process
begin
wait for 50ns;
ip1<="000";

wait for 50ns;
```

```
ip1<="001";

wait for 50ns;
ip1<="010";

wait for 50ns;
ip1<="011";
wait for 50ns;
ip1<="100";
wait for 50ns;
ip1<="101";
wait for 50ns;
ip1<="110";
wait for 50ns;
ip1<="111";
wait;
end process;

end Behavioral;
```

4.Waveform

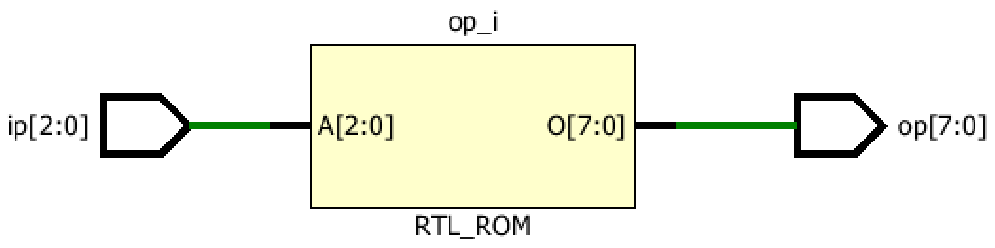


3:8 decoder (Behavioral Model)

1.VHDL Code

```
entity decoder3_8_bv is
   Port ( ip : in STD_LOGIC_VECTOR (2 downto 0);
         op : out STD_LOGIC_VECTOR (7 downto 0));
end decoder3_8_bv;

architecture Behavioral of decoder3_8_bv is

begin
process(ip)
begin
case ip is
when "000" =>op<="00000001";
when "001" =>op<="00000010";
when "010" =>op<="00000100";
when "011" =>op<="00001000";
when "100" =>op<="00010000";
when "101" =>op<="00100000";
when "110" =>op<="01000000";
when "111" =>op<="10000000";
when others => NULL;
end case;
end process;
end Behavioral;
```

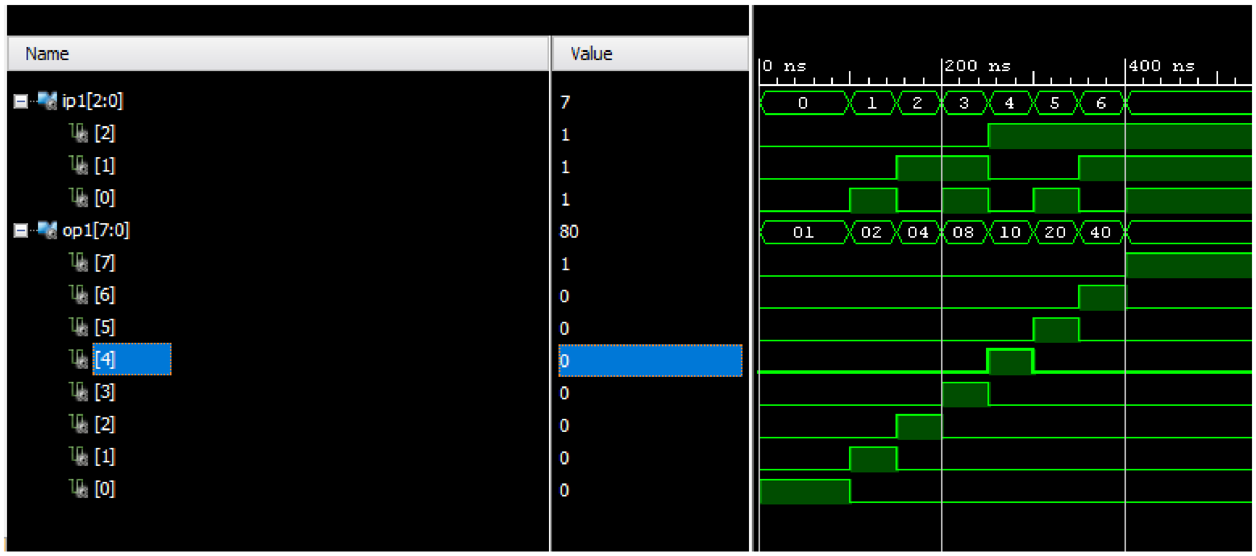
2.RTL Design



3.TBW Code

```
entity decoder3_8_bv_tbw is
--  Port ( );
end decoder3_8_bv_tbw;

architecture Behavioral of decoder3_8_bv_tbw is
component decoder3_8_bv is
   Port ( ip : in STD_LOGIC_VECTOR (2 downto 0);
          op : out STD_LOGIC_VECTOR (7 downto 0));
end component;

signal ip1:STD_LOGIC_VECTOR(2 downto 0):="000";
signal op1:STD_LOGIC_VECTOR(7 downto 0);
begin
uut: decoder3_8_bv port map(ip=>ip1,op=>op1);
stim_proc: process
begin
wait for 50ns;
ip1<="000";

wait for 50ns;
ip1<="001";

wait for 50ns;
ip1<="010";

wait for 50ns;
ip1<="011";

wait for 50ns;
ip1<="100";

wait for 50ns;
ip1<="101";

wait for 50ns;
ip1<="110";

wait for 50ns;
ip1<="111";
wait;
end process;

end Behavioral;
```

4.Waveform



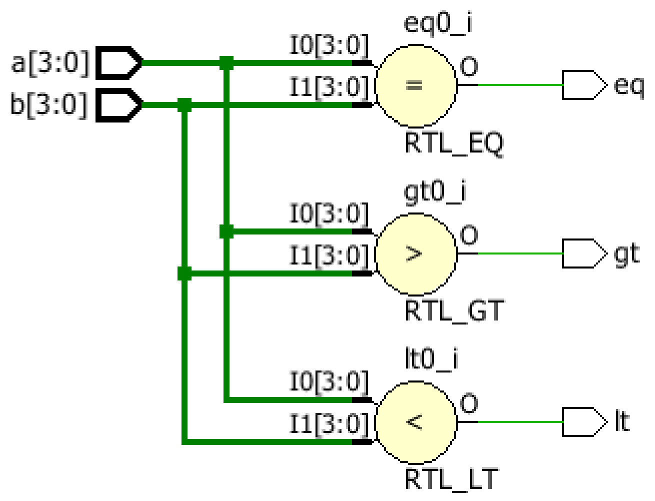
Name
Value
ip1[2:0]
7
[2]
1
[1]
1
[0]
1
op1[7:0]
80
[7]
1
[6]
0
[5]
0
[4]
0
[3]
0
[2]
0
[1]
0
[0]
0
0 ns
200 ns
400 ns
0
1
2
3
4
5
6
01
02
04
08
10
20
40

4-bit Comparator (Dataflow Model)

1.VHDL Code

```
entity comparator_4_df is
   Port ( a : in STD_LOGIC_VECTOR (3 downto 0);
         b : in STD_LOGIC_VECTOR (3 downto 0);
         eq : out STD_LOGIC;
         lt : out STD_LOGIC;
         gt : out STD_LOGIC);
end comparator_4_df;

architecture Dataflow of comparator_4_df is

begin
eq<='1' when a=b else '0';
lt<='1' when a<b else '0';
gt<='1' when a>b else '0';

end Dataflow;
```

2.RTL Design

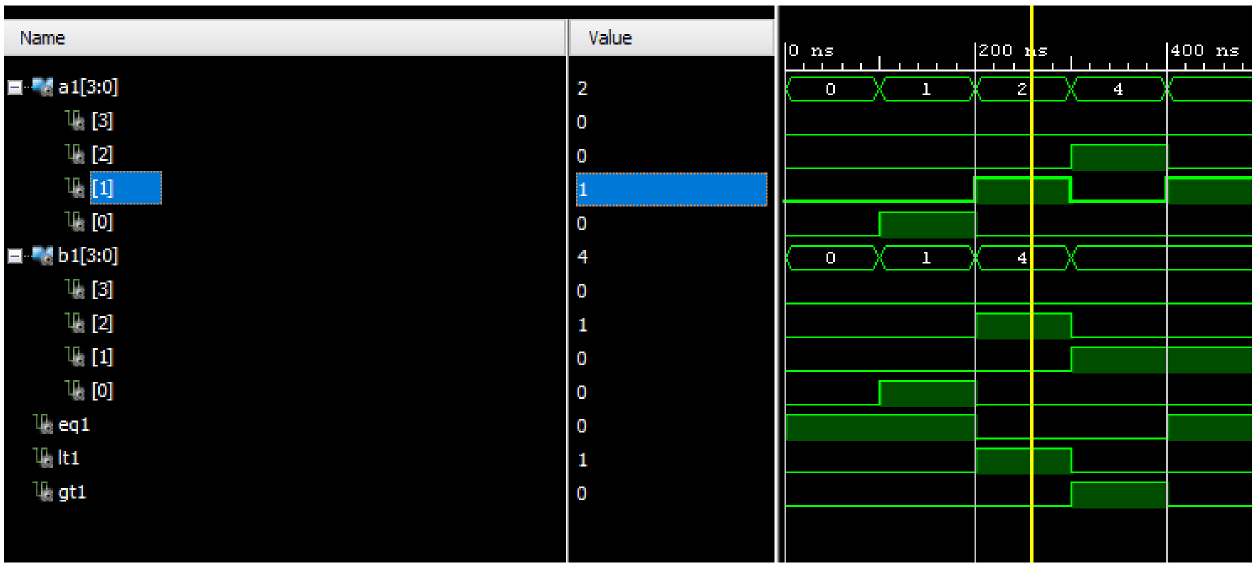


3.TBW Code

```
entity comparator_4_df_tbw is
-- Port ( );
end comparator_4_df_tbw;

architecture Behavioral of comparator_4_df_tbw is
component comparator_4_df is
```

```
   Port ( a : in STD_LOGIC_VECTOR (3 downto 0);
         b : in STD_LOGIC_VECTOR (3 downto 0);
         eq : out STD_LOGIC;
         lt : out STD_LOGIC;
         gt : out STD_LOGIC);
end component;
signal a1:STD_LOGIC_VECTOR(3 downto 0):="0000";
signal b1:STD_LOGIC_VECTOR(3 downto 0):="0000";
signal eq1:STD_LOGIC;
signal lt1:STD_LOGIC;
signal gt1:STD_LOGIC;
begin
uut: comparator_4_df port map(a=>a1,b=>b1,eq=>eq1,lt=>lt1,gt=>gt1);
stim_proc: process
begin
wait for 100ns;
a1<="0001";
b1<="0001";

wait for 100ns;
a1<="0010";
b1<="0100";

wait for 100ns;
a1<="0100";
b1<="0010";

wait for 100ns;
a1<="0010";
b1<="0010";

wait;
end process;

end Behavioral;
```

4.Waveform



Name	Value
a1[3:0]	2
[3]	0
[2]	0
[1]	1
[0]	0
b1[3:0]	4
[3]	0
[2]	1
[1]	0
[0]	0
eq1	0
lt1	1
gt1	0

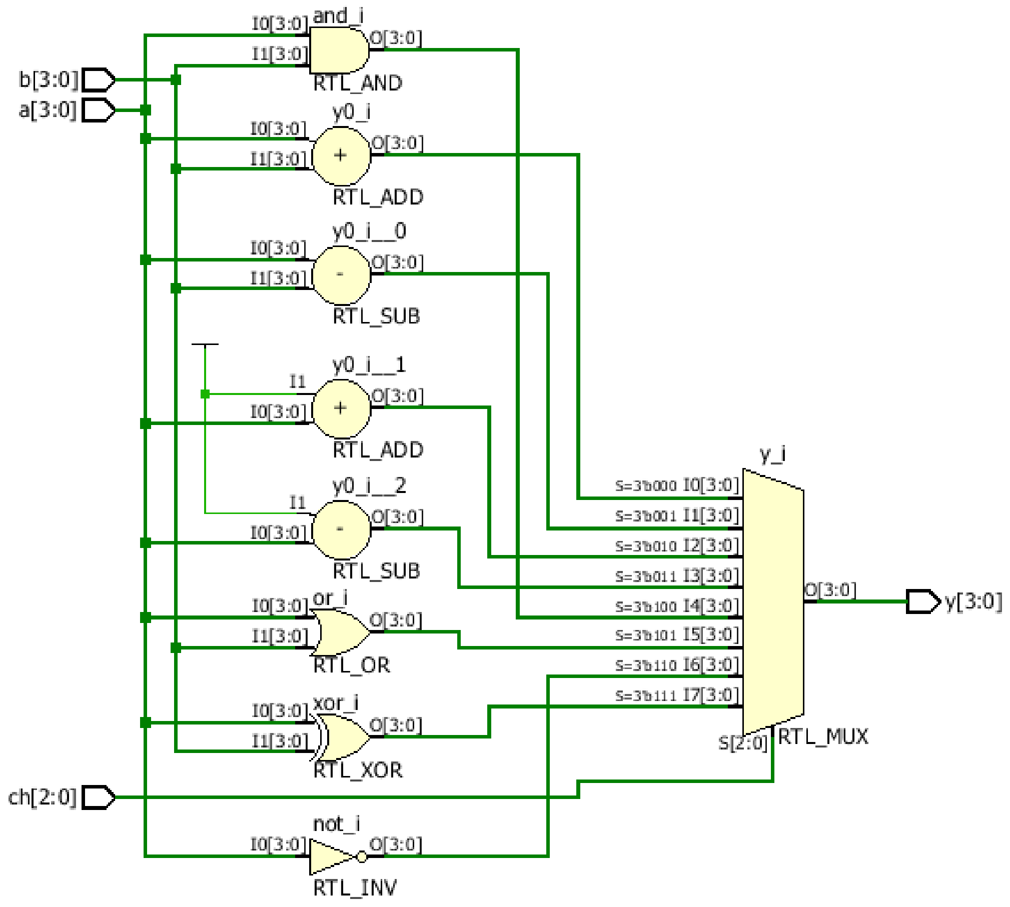
DAY-6

Arithmetic Logic Unit (Behavioral Model)

1.VHDL Design

```
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);
        y : 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" => y<=a+b;
when "001" => y<=a-b;
when "010" => y<=a+"0001";
when "011" => y<=a-"0001";
when "100" => y<=a AND b;
when "101" => y<=a OR b;
when "110" => y<= NOT a;
when "111" => y<=a XOR b;
when others=> null;
end case;
end process;
end Behavioral;
```

2.RTL Design

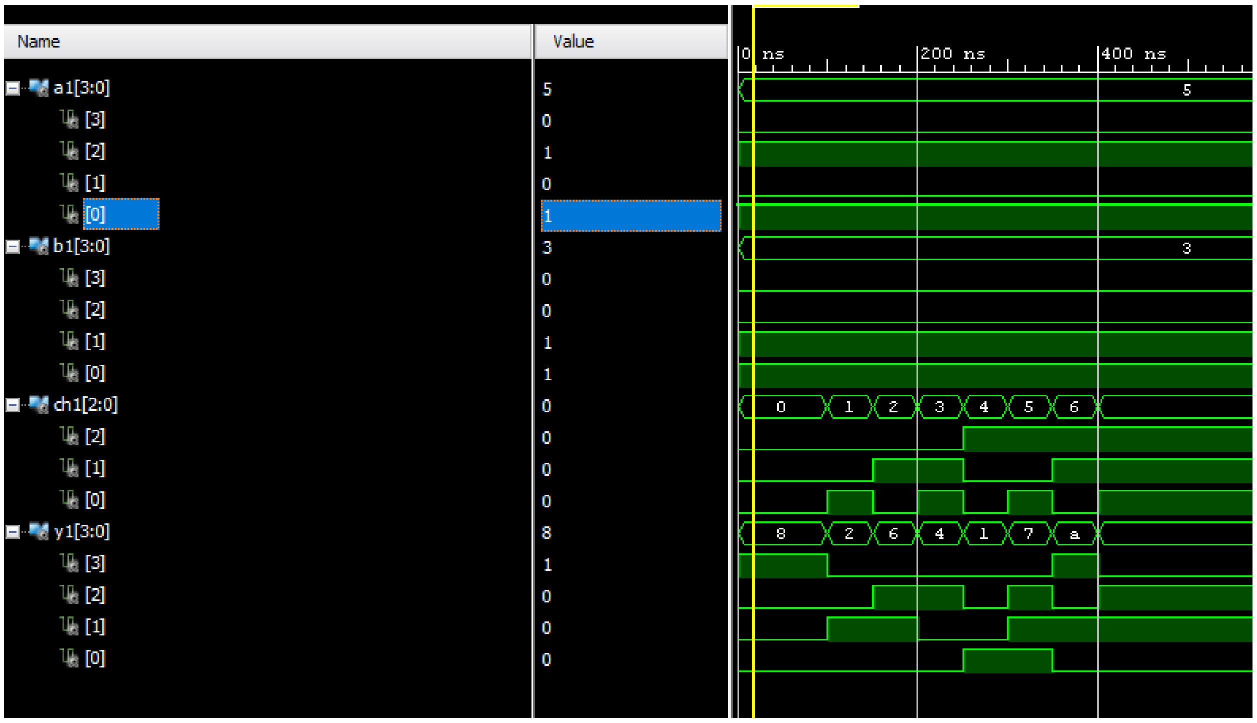


3.TBW Code

```
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);
       ch : in STD_LOGIC_VECTOR (2 downto 0);
       y : out UNSIGNED (3 downto 0));
end component;
signal a1:UNSIGNED (3 downto 0):="0101";
signal b1:UNSIGNED (3 downto 0):="0011";
signal ch1:STD_LOGIC_VECTOR (2 downto 0):="000";
signal y1:UNSIGNED (3 downto 0);
begin
uut: alu_bv port map(a=>a1,b=>b1,ch=>ch1,y=>y1);
stim_proc: process
begin
```

```
 wait for 50ns;
 ch1<="000";
 wait for 50ns;
 ch1<="001";
 wait for 50ns;
 ch1<="010";
 wait for 50ns;
 ch1<="011";
 wait for 50ns;
 ch1<="100";
 wait for 50ns;
 ch1<="101";
 wait for 50ns;
 ch1<="110";
 wait for 50ns;
 ch1<="111";
 wait;
 end process;
end Behavioral;
```

4.Waveform



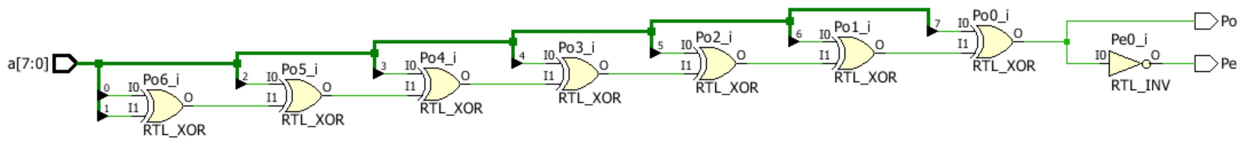
Name	Value
a1[3:0]	5
[3]	0
[2]	1
[1]	0
[0]	1
b1[3:0]	3
[3]	0
[2]	0
[1]	1
[0]	1
ch1[2:0]	0
[2]	0
[1]	0
[0]	0
y1[3:0]	8
[3]	1
[2]	0
[1]	0
[0]	0

8-bit Parity Checker (Behavioral Model)

1.VHDL Design

```
entity parity_8 is
   Port ( a : in STD_LOGIC_VECTOR (7 downto 0);
         Po : out STD_LOGIC;
         Pe : out STD_LOGIC);
end parity_8;

architecture Behavioral of parity_8 is

begin
process (a)
variable temp:STD_LOGIC;
begin
temp:=a(0) XOR a(1);
for i in 2 to 7 loop
temp:= a(i) XOR temp;
end loop;
Po<=temp;
Pe<= NOT temp;
end process;

end Behavioral;
```

2.RTL Design

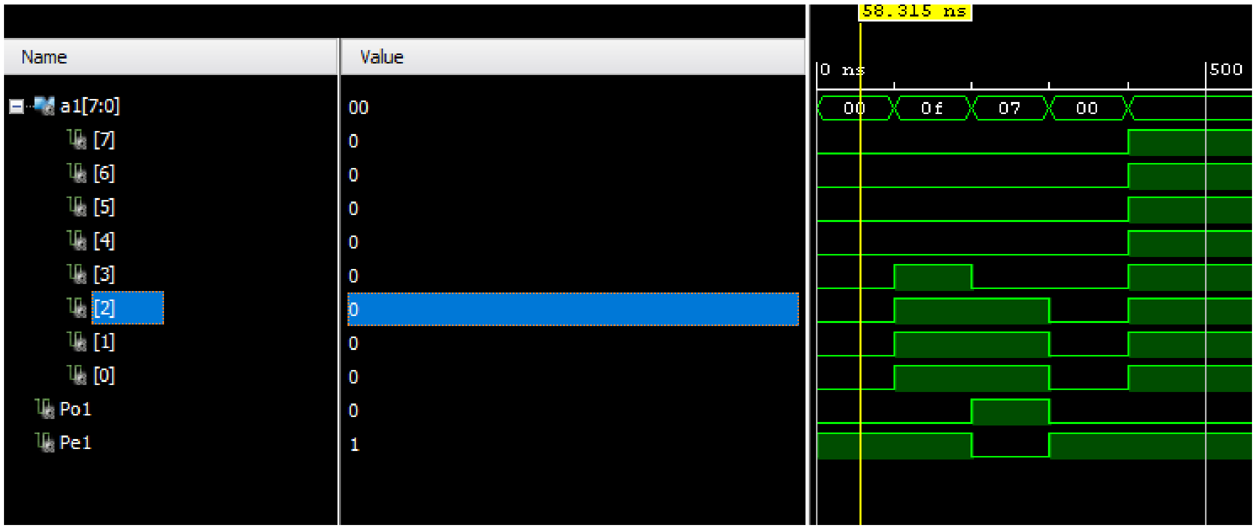


3.TBW Code

```
entity parity_8_tbw is
--  Port ( );
end parity_8_tbw;

architecture Behavioral of parity_8_tbw is
component parity_8 is
   Port ( a : in STD_LOGIC_VECTOR (7 downto 0);
         Po : out STD_LOGIC;
         Pe : out STD_LOGIC);
end component;
```

```
signal a1:STD_LOGIC_VECTOR (7 downto 0):="00000000";
signal Po1: STD_LOGIC;
signal Pe1: STD_LOGIC;

begin
uut: parity_8 port map(a=>a1,Po=>Po1,Pe=>Pe1);
stim_proc: process
begin
--even
wait for 100 ns;
a1<="00001111";

--odd
wait for 100 ns;
a1<="00000111";

--odd
wait for 100 ns;
a1<="00000000";

--even
wait for 100 ns;
a1<="11111111";

wait;
end process;
end Behavioral;
```

4.Waveform



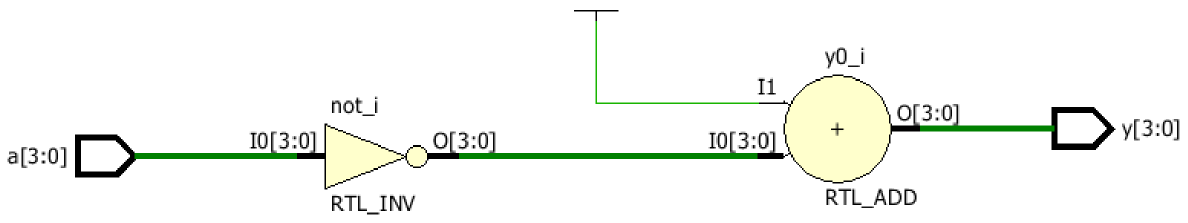
Name	Value
a1[7:0]	00
[7]	0
[6]	0
[5]	0
[4]	0
[3]	0
[2]	0
[1]	0
[0]	0
Po1	0
Pe1	1

2’s Complement (Behavioral Model)

1.VHDL Design

```
use IEEE.NUMERIC_STD.ALL;
entity complement_2 is
   Port ( a : in UNSIGNED (3 downto 0);
        y : out UNSIGNED (3 downto 0));
end complement_2;

architecture Behavioral of complement_2 is

begin
process(a)
begin
y<=(NOT a) + "0001";
end process;

end Behavioral;
```

2.RTL Design

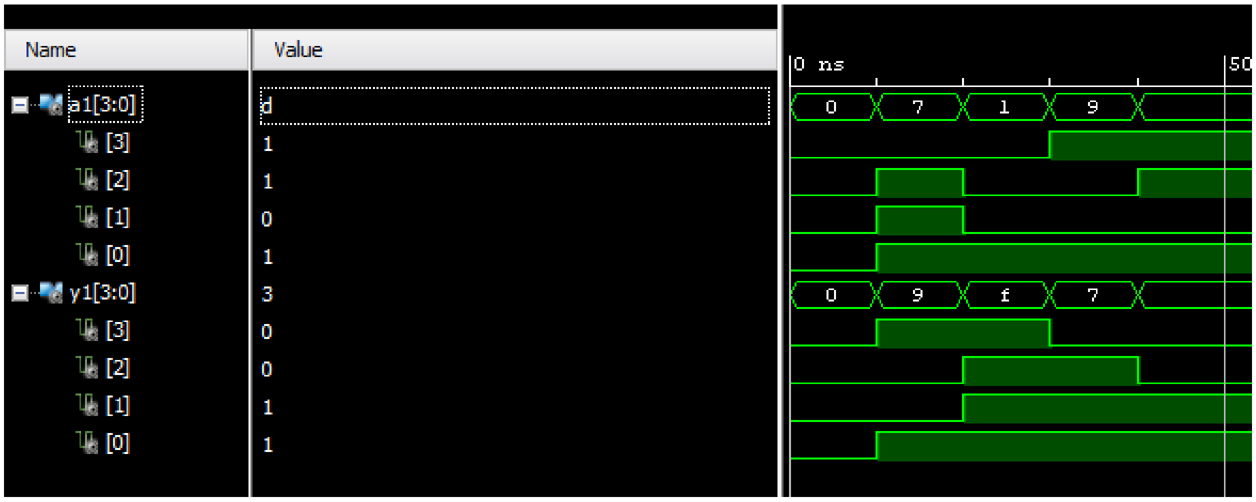


3.TBW Code

```
use IEEE.NUMERIC_STD.ALL;
entity complement_2_tbw is
--  Port ( );
end complement_2_tbw;

architecture Behavioral of complement_2_tbw is
component complement_2 is
   Port ( a : in UNSIGNED (3 downto 0);
        y : out UNSIGNED (3 downto 0));
end component;
signal a1:UNSIGNED (3 downto 0):="0000";
signal y1:UNSIGNED (3 downto 0);
```

```
begin
uut: complement_2 port map(a=>a1,y=>y1);
stim_proc: process
begin
wait for 100 ns;
a1<="0111";
wait for 100 ns;
a1<="0001";
wait for 100 ns;
a1<="1001";
wait for 100 ns;
a1<="1101";
wait;
end process;
end Behavioral;
```

4.Waveform



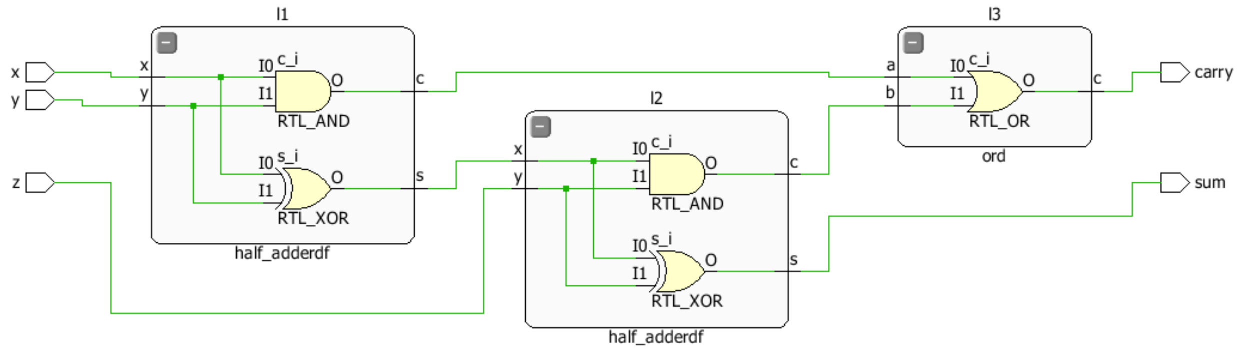
DAY 7

Full Adder (Structural Model)

1.VHDL Design

```
entity fulladder_str is
   Port ( x : in STD_LOGIC;
        y : in STD_LOGIC;
        z : in STD_LOGIC;
        sum : out STD_LOGIC;
        carry : out STD_LOGIC);
end fulladder_str;

architecture Structural of fulladder_str is
component ord is
   Port ( a : in STD_LOGIC;
        b : in STD_LOGIC;
        c : out STD_LOGIC);
end component;
component half_adderdf is
   Port ( x : in STD_LOGIC;
        y : in STD_LOGIC;
        s : out STD_LOGIC;
        c : out STD_LOGIC);
end component;
signal s1,c1,c2:STD_LOGIC;
begin
l1:half_adderdf port map (x,y,s1,c1);
l2:half_adderdf port map(s1,z,sum,c2);
l3:ord port map(c1,c2,carry);
end Structural;
```

2.RTL Design

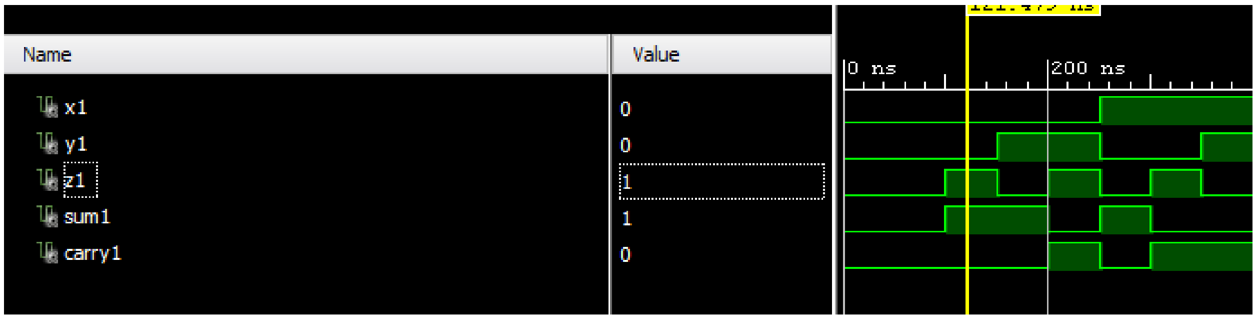


3.TBW Code

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

architecture Behavioral of fulladder_str_tbw is
component fulladder_str is
  Port ( x : in STD_LOGIC;
       y : in STD_LOGIC;
       z : in STD_LOGIC;
       sum : out STD_LOGIC;
       carry : out STD_LOGIC);
end component;
signal x1: STD_LOGIC:='0';
signal y1: STD_LOGIC:='0';
signal z1: STD_LOGIC:='0';
signal sum1: STD_LOGIC;
signal carry1: STD_LOGIC;
begin
uut: fulladder_str port map(x=>x1,y=>y1,z=>z1,sum=>sum1,carry=>carry1);
stim_proc: process
begin
wait for 50 ns;
x1<='0';
y1<='0';
z1<='0';
wait for 50 ns;
x1<='0';
y1<='0';
z1<='1';
wait for 50 ns;
```

```
x1<='0';
y1<='1';
z1<='0';
wait for 50 ns;
x1<='0';
y1<='1';
z1<='1';
wait for 50 ns;
x1<='1';
y1<='0';
z1<='0';
wait for 50 ns;
x1<='1';
y1<='0';
z1<='1';
wait for 50 ns;
x1<='1';
y1<='1';
z1<='0';
wait for 50 ns;
x1<='1';
y1<='1';
z1<='1';
wait;
end process;
end Behavioral;
```

4.Waveform



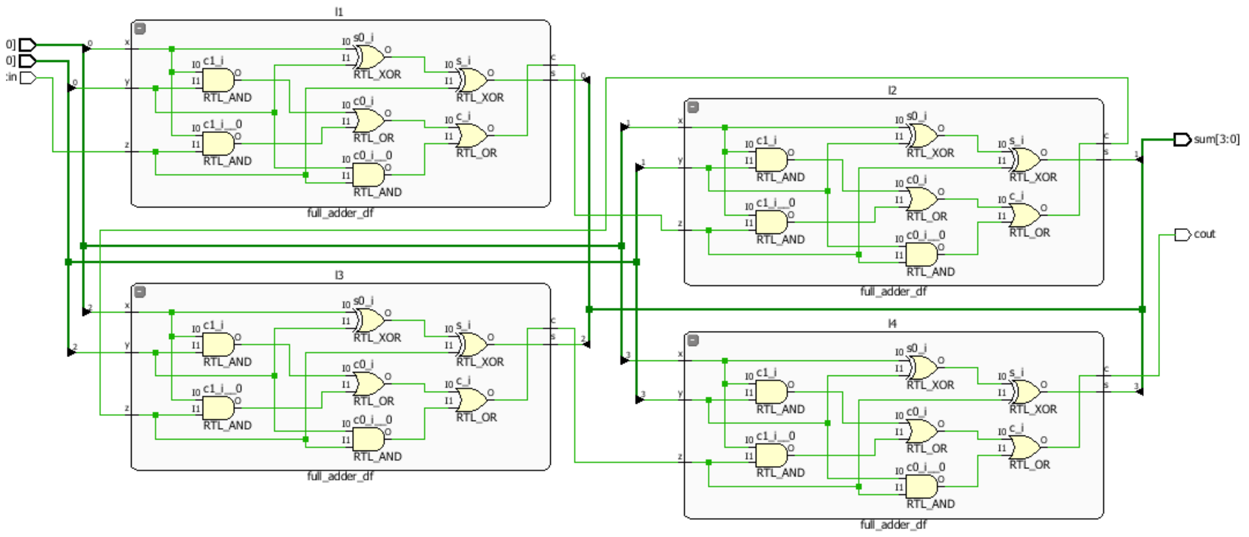
Name	Value
x1	0
y1	0
z1	1
sum1	1
carry1	0

4-bit Ripple Carry Adder (Structural Model)

1.VHDL Design

```
entity ripple_carry is
   Port ( a : in STD_LOGIC_VECTOR (3 downto 0);
        b : in STD_LOGIC_VECTOR (3 downto 0);
        cin : in STD_LOGIC;
        sum : out STD_LOGIC_VECTOR (3 downto 0);
        cout : out STD_LOGIC);
end ripple_carry;

architecture Structural of ripple_carry is
component full_adder_df is
   Port ( x : in STD_LOGIC;
        y : in STD_LOGIC;
        z : in STD_LOGIC;
        s : out STD_LOGIC;
        c : out STD_LOGIC);
end component;

signal c0,c1,c2:STD_LOGIC;
begin
l1:full_adder_df port map (a(0),b(0),cin,sum(0),c0);
l2:full_adder_df port map (a(1),b(1),c0,sum(1),c1);
l3:full_adder_df port map (a(2),b(2),c1,sum(2),c2);
l4:full_adder_df port map (a(3),b(3),c2,sum(3),cout);

end Structural;
```

2.RTL Design

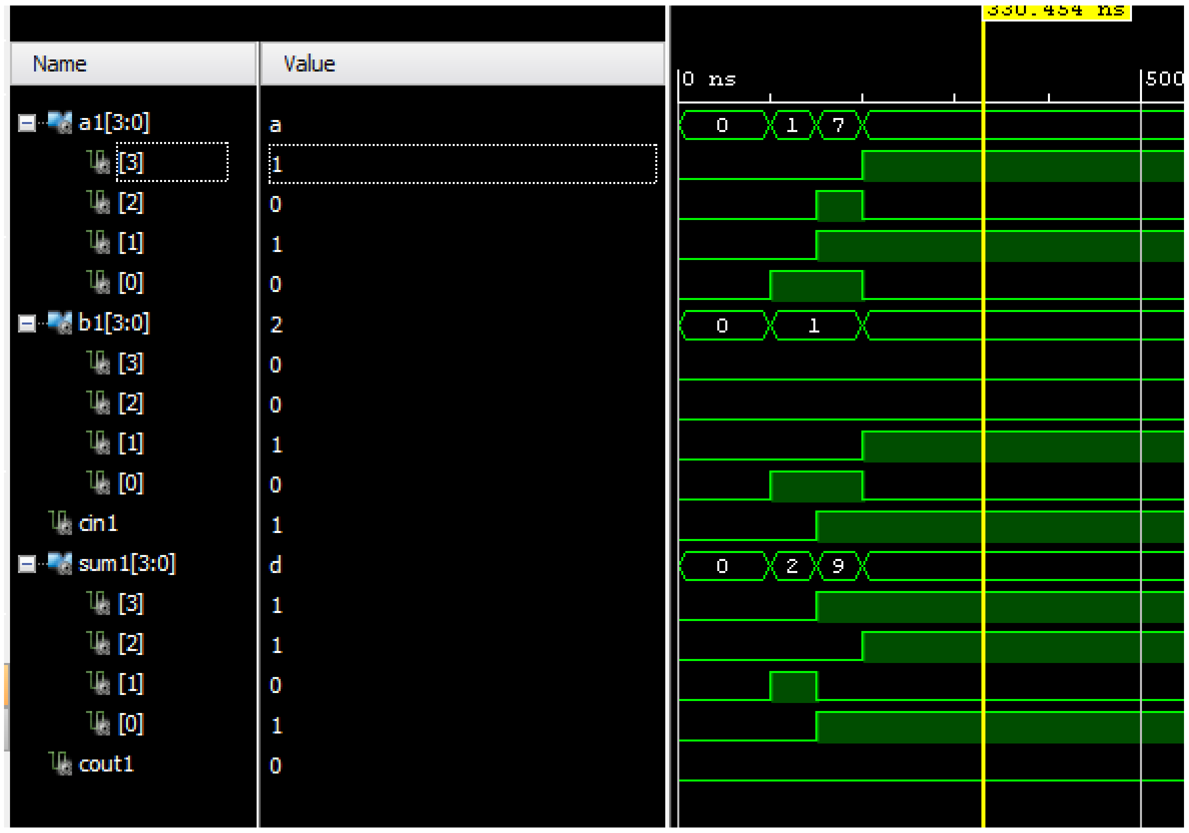


3.TBW Code

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

architecture Behavioral of ripple_carry_tbw is
component ripple_carry is
   Port ( a : in STD_LOGIC_VECTOR (3 downto 0);
        b : in STD_LOGIC_VECTOR (3 downto 0);
        cin : in STD_LOGIC;
        sum : out STD_LOGIC_VECTOR (3 downto 0);
        cout : out STD_LOGIC);
end component;

signal a1: STD_LOGIC_VECTOR (3 downto 0):="0000";
signal b1: STD_LOGIC_VECTOR (3 downto 0):="0000";
signal cin1: STD_LOGIC:='0';
signal sum1: STD_LOGIC_VECTOR (3 downto 0);
signal cout1: STD_LOGIC;

begin
uut: ripple_carry port map(a=>a1,b=>b1,cin=>cin1,sum=>sum1,cout=>cout1);
stim_proc: process
begin
wait for 50 ns;
a1<="0000";
b1<="0000";
cin1<='0';
```

```
wait for 50 ns;
a1<="0001";
b1<="0001";
cin1<='0';

wait for 50 ns;
a1<="0111";
b1<="0001";
cin1<='1';

wait for 50 ns;
a1<="1010";
b1<="0010";
cin1<='1';

wait;
end process;
end Behavioral;
```

4.Waveform

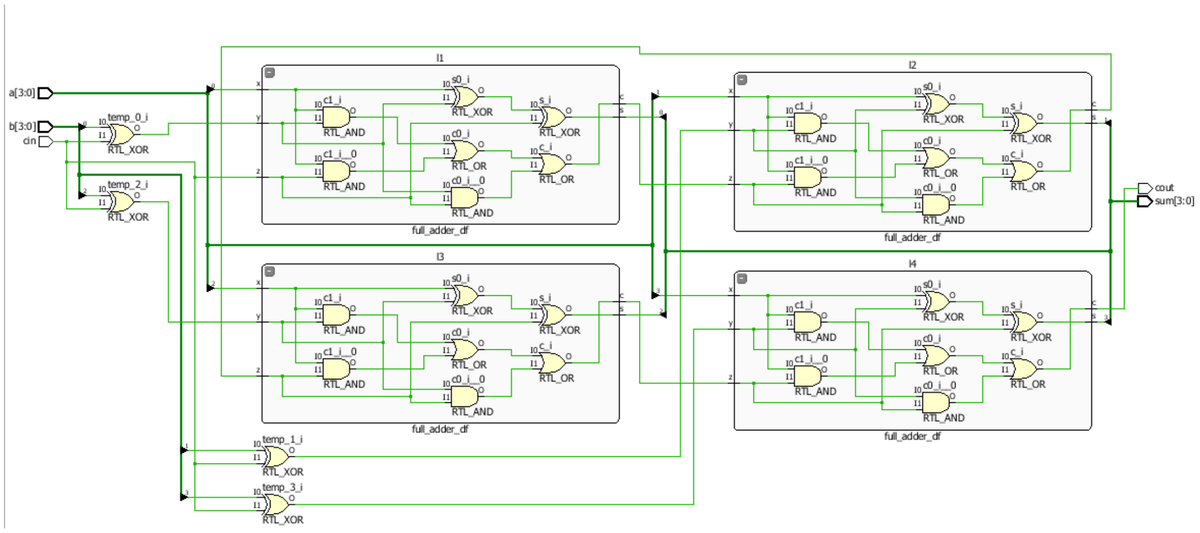


Adder-Subtractor Composite Unit (Structural Model)

1.VHDL Design

```
entity add_sub_comp is
   Port ( a : in STD_LOGIC_VECTOR (3 downto 0);
        b : in STD_LOGIC_VECTOR (3 downto 0);
        cin : in STD_LOGIC;
        sum : out STD_LOGIC_VECTOR (3 downto 0);
        cout : out STD_LOGIC);
end add_sub_comp;

architecture Structural of add_sub_comp is

component full_adder_df is
   Port ( x : in STD_LOGIC;
        y : in STD_LOGIC;
        z : in STD_LOGIC;
        s : out STD_LOGIC;
        c : out STD_LOGIC);
end component;

signal c0,c1,c2:STD_LOGIC;
signal temp:STD_LOGIC_VECTOR(3 downto 0);
begin
gk:for i in 0 to 3 generate
temp(i)<=b(i) XOR cin;
end generate;

l1:full_adder_df port map (a(0),temp(0),cin,sum(0),c0);
l2:full_adder_df port map (a(1),temp(1),c0,sum(1),c1);
l3:full_adder_df port map (a(2),temp(2),c1,sum(2),c2);
l4:full_adder_df port map (a(3),temp(3),c2,sum(3),cout);

end Structural;
```

2.RTL Design

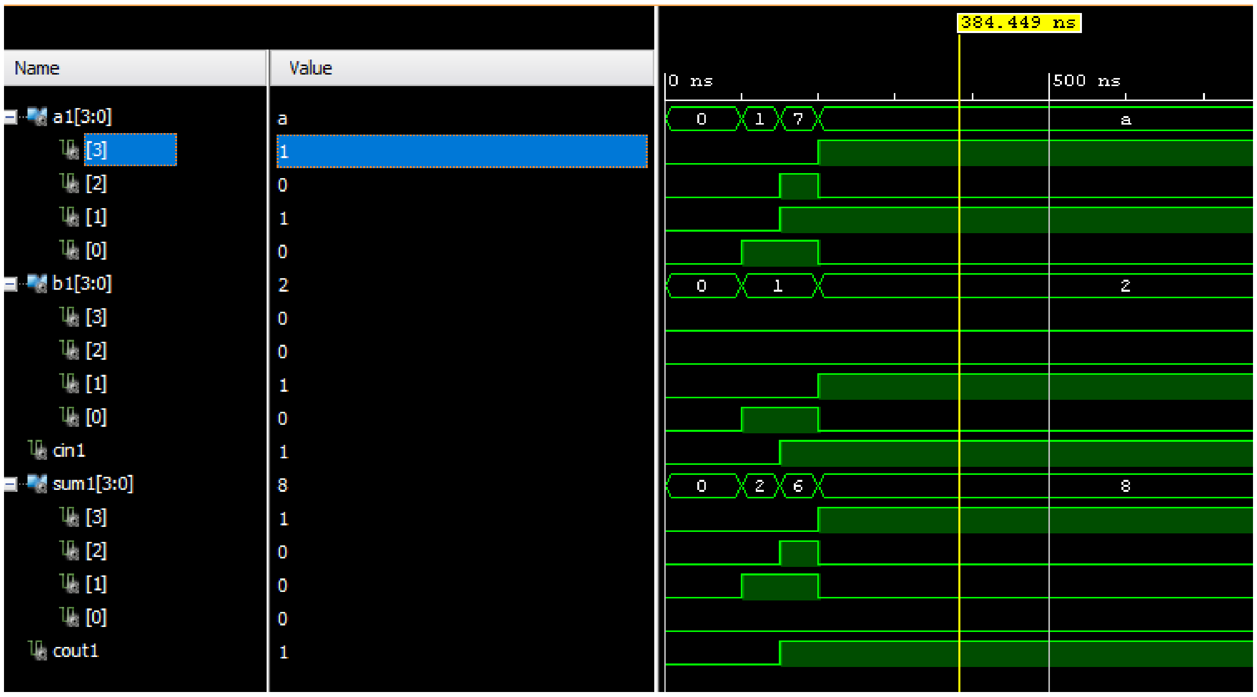


3.TBW Code

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

architecture Behavioral of add_sub_comp_tbw is
component add_sub_comp is
   Port ( a : in STD_LOGIC_VECTOR (3 downto 0);
          b : in STD_LOGIC_VECTOR (3 downto 0);
          cin : in STD_LOGIC;
          sum : out STD_LOGIC_VECTOR (3 downto 0);
          cout : out STD_LOGIC);
end component;

signal a1: STD_LOGIC_VECTOR (3 downto 0):="0000";
signal b1: STD_LOGIC_VECTOR (3 downto 0):="0000";
signal cin1: STD_LOGIC:='0';
signal sum1: STD_LOGIC_VECTOR (3 downto 0);
signal cout1: STD_LOGIC;

begin
uut: add_sub_comp port map(a=>a1,b=>b1,cin=>cin1,sum=>sum1,cout=>cout1);
stim_proc: process
begin
wait for 50 ns;
a1<="0000";
b1<="0000";
cin1<='0';

wait for 50 ns;
```

```
a1<="0001";
b1<="0001";
cin1<='0';

wait for 50 ns;
a1<="0111";
b1<="0001";
cin1<='1';

wait for 50 ns;
a1<="1010";
b1<="0010";
cin1<='1';

wait;
end process;
end Behavioral;
```

4.Waveform

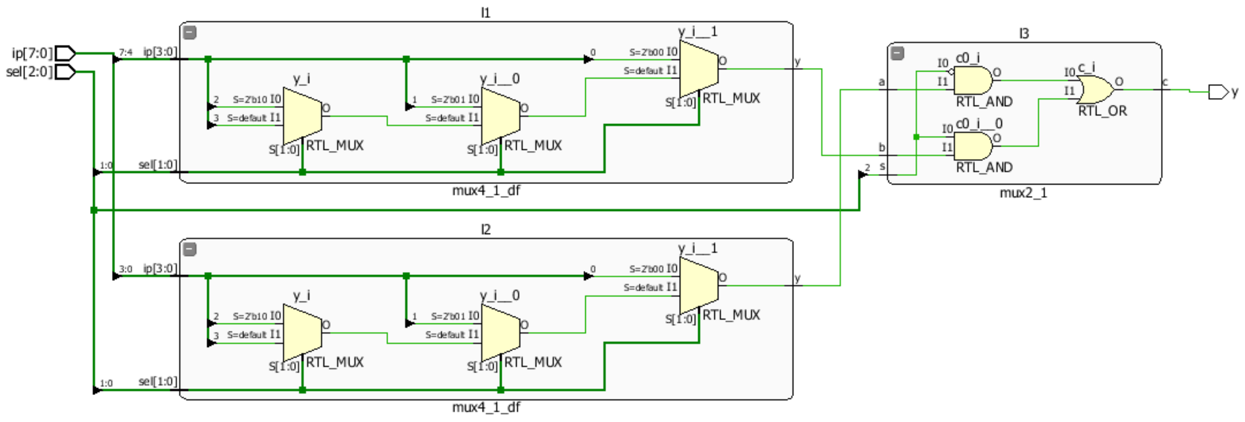


8:1 MUX (Structural Model)

1.VHDL Design

```
entity mux8_1 is
   Port ( ip : in STD_LOGIC_VECTOR (7 downto 0);
        sel : in STD_LOGIC_VECTOR (2 downto 0);
        y : out STD_LOGIC);
end mux8_1;

architecture Structural of mux8_1 is
component mux4_1_df is
   Port ( ip : in STD_LOGIC_VECTOR (3 downto 0);
        sel : in STD_LOGIC_VECTOR (1 downto 0);
        y : out STD_LOGIC);
end component;

component mux2_1 is
   Port ( a : in STD_LOGIC;
        b : in STD_LOGIC;
        c : out STD_LOGIC;
        s : in STD_LOGIC);
end component;
signal x1,x2:STD_LOGIC;
begin
l1:mux4_1_df port map(ip(7 downto 4),sel(1 downto 0),x1);
l2:mux4_1_df port map(ip(3 downto 0),sel(1 downto 0),x2);
l3:mux2_1 port map(x2,x1,y,sel(2));

end Structural;
```

2.RTL Design

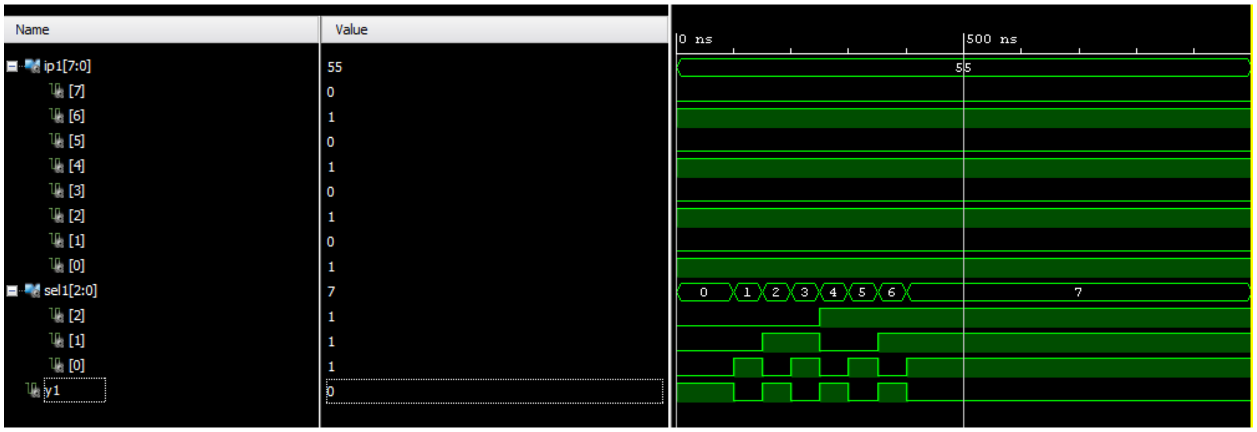


3.TBW Code

```
entity mux8_1_tbw is
--  Port ( );
end mux8_1_tbw;
architecture Behavioral of mux8_1_tbw is
component mux8_1 is
   Port ( ip : in STD_LOGIC_VECTOR (7 downto 0);
        sel : in STD_LOGIC_VECTOR (2 downto 0);
        y : out STD_LOGIC);
end component;

signal ip1:STD_LOGIC_VECTOR(7 downto 0):="01010101";
signal sel1:STD_LOGIC_VECTOR(2 downto 0):="000";
signal y1:STD_LOGIC;
begin
uut:mux8_1 port map(ip=>ip1,sel=>sel1,y=>y1);
stim_proc:process
begin
wait for 50 ns;
sel1<="000";

wait for 50 ns;
sel1<="001";

wait for 50 ns;
sel1<="010";

wait for 50 ns;
sel1<="011";
```

```
wait for 50 ns;
sel1<="100";

wait for 50 ns;
sel1<="101";

wait for 50 ns;
sel1<="110";

wait for 50 ns;
sel1<="111";

wait;
end process;
end Behavioral;
```

4.Waveform



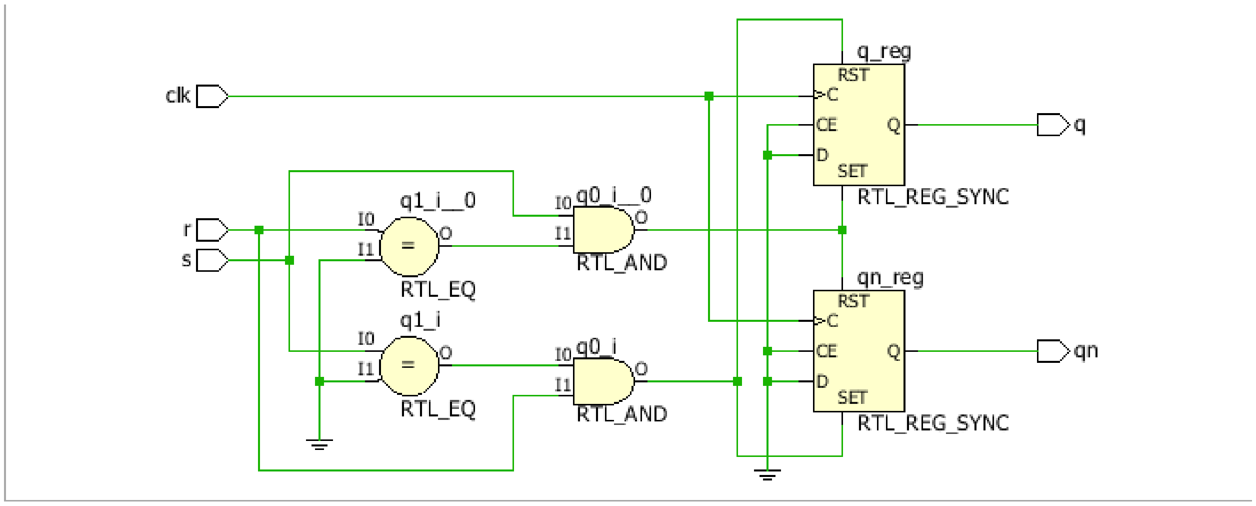
DAY 8

SR Flip-Flop (Behavioral Model)

1.VHDL Design

```
entity sr_ff is
   Port ( s : in STD_LOGIC;
         r : in STD_LOGIC;
         clk : in STD_LOGIC;
         q : out STD_LOGIC;
         qn : out STD_LOGIC);
end sr_ff;

architecture Behavioral of sr_ff is

begin
process(s,r,clk)
begin
if (clk'event and clk='1') then
if(s='0' and r='1') then
q<='0';
qn<='1';

elsif(s='1' and r='0') then
q<='1';
qn<='0';

elsif(s='1' and r='1') then
q<='X';
qn<='X';
end if;
end if;
end process;
end Behavioral;
```

2.RTL Design

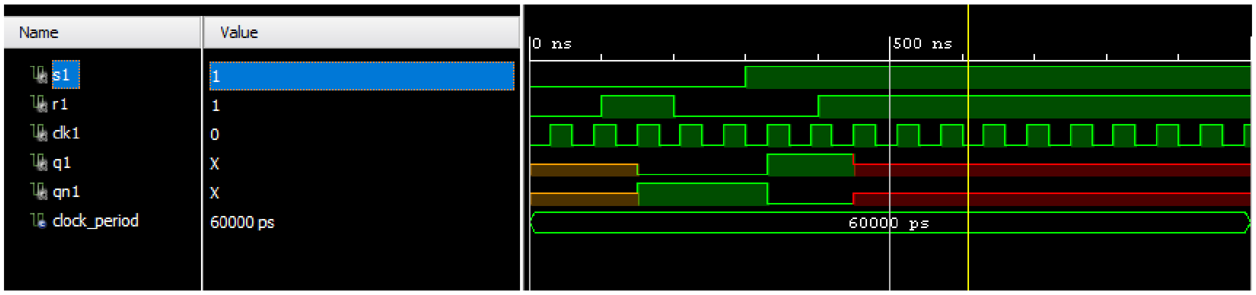


3.TBW Code

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

architecture Behavioral of ss_ff_tbw is
component sr_ff is
   Port ( s : in STD_LOGIC;
        r : in STD_LOGIC;
        clk : in STD_LOGIC;
        q : out STD_LOGIC;
        qn : out STD_LOGIC);
end component;
constant clock_period:time:=60 ns;
signal s1:STD_LOGIC:='0';
signal r1:STD_LOGIC:='0';
signal clk1:STD_LOGIC:='0';
signal q1:STD_LOGIC;
signal qn1:STD_LOGIC;
begin
uut: sr_ff port map(s=>s1,r=>r1,clk=>clk1,q=>q1,qn=>qn1);
clk1<=not clk1 after clock_period/2;
stim_proc:process
begin
wait for 100 ns;
s1<='0';
r1<='1';

wait for 100 ns;
s1<='0';
r1<='0';
```

```
wait for 100 ns;
s1<='1';
r1<='0';
wait for 100 ns;
s1<='1';
r1<='1';

wait;
end process;

end Behavioral;
```

4.Waveform

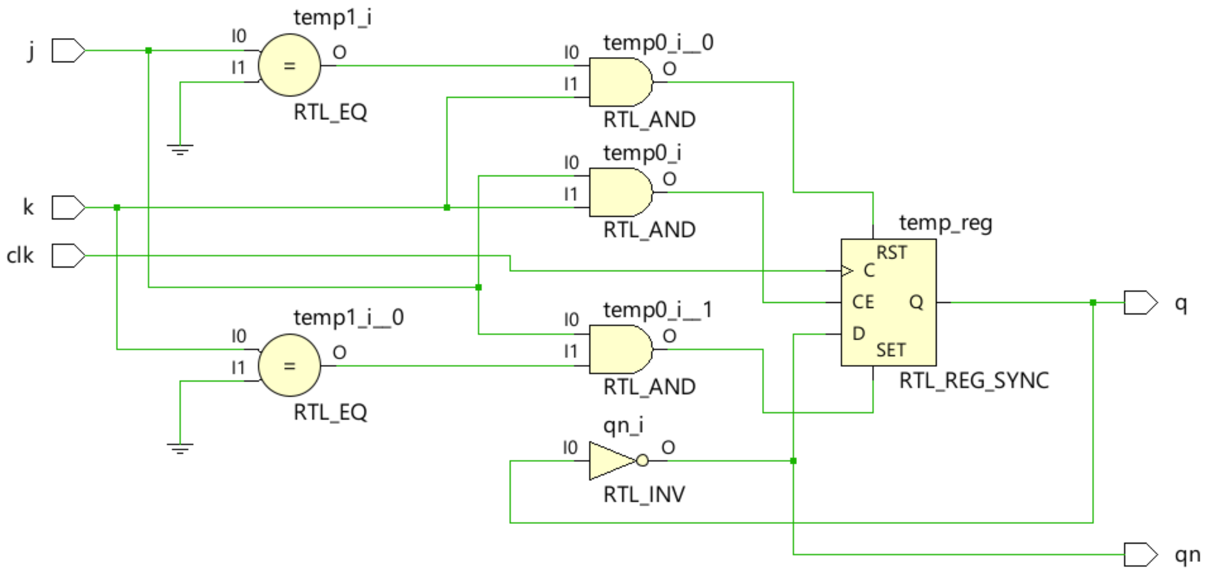


JK Flip-Flop (Behavioral Model)

1.VHDL Design

```
entity jk_ff is
   Port ( j : in STD_LOGIC;
        k : in STD_LOGIC;
        clk : in STD_LOGIC;
        q : out STD_LOGIC;
        qn : out STD_LOGIC);
end jk_ff;
architecture Behavioral of jk_ff is
signal temp : STD_LOGIC := '0';
begin
process(clk)
begin
   if (clk'event and clk = '1') then
      if (j = '0' and k = '1') then
         temp <= '0';  -- Reset
      elsif (j = '1' and k = '0') then
         temp <= '1';  -- Set
      elsif (j = '1' and k = '1') then
         temp <= not temp;  -- Toggle
      end if; -- (j=0 and k=0): Hold -> do nothing
   end if;
end process;
q <= temp;
qn <= not temp;
end Behavioral;
```

2.RTL Design

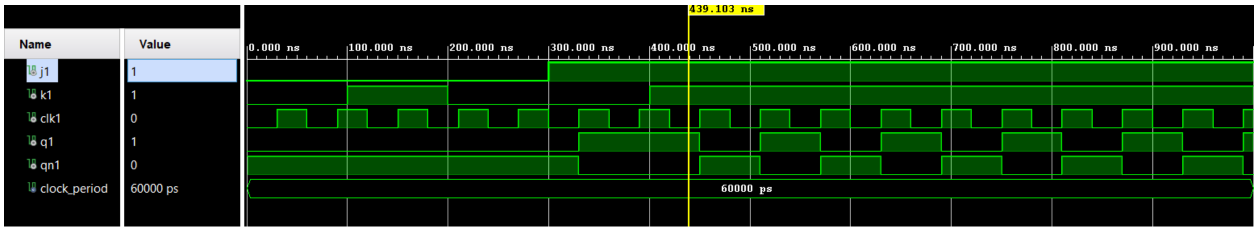


3.TBW Code

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

architecture Behavioral of jk_ff_tbw is
component jk_ff is
   Port ( j : in STD_LOGIC;
        k : in STD_LOGIC;
        clk : in STD_LOGIC;
        q : inout STD_LOGIC;
        qn : inout STD_LOGIC);
end component;

constant clock_period:time:=60 ns;
signal j1:STD_LOGIC:='0';
signal k1:STD_LOGIC:='0';
signal clk1:STD_LOGIC:='0';
signal q1:STD_LOGIC;
signal qn1:STD_LOGIC;
begin
uut: jk_ff port map(j=>j1,k=>k1,clk=>clk1,q=>q1,qn=>qn1);
clk1<=not clk1 after clock_period/2;
stim_proc:process
begin
wait for 100 ns;
```

```
j1<='0';
k1<='1';
wait for 100 ns;
j1<='0';
k1<='0';
wait for 100 ns;
j1<='1';
k1<='0';
wait for 100 ns;
j1<='1';
k1<='1';
wait;
end process;
end Behavioral;
```

4.Waveform

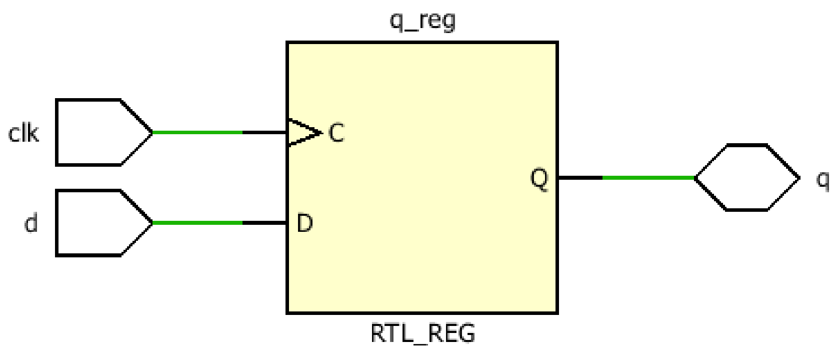


D Flip-Flop (Behavioral Model)

1.VHDL Design

```
entity d_ff is
   Port ( d : in STD_LOGIC;
        clk : in STD_LOGIC;
        q : inout STD_LOGIC);
end d_ff;
architecture Behavioral of d_ff is
begin
process(d,clk)
begin
if(clk'event and clk='1') then
q<=d;
end if;
end process;
end Behavioral;
```

2.RTL Design

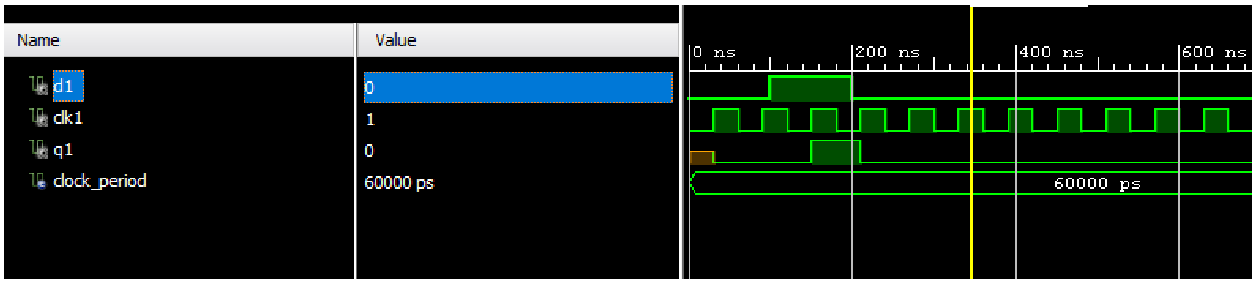


3.TBW Code

```
entity d_ff_tbw is
--  Port ( );
end d_ff_tbw;
architecture Behavioral of d_ff_tbw is
component d_ff is
   Port ( d : in STD_LOGIC;
        clk : in STD_LOGIC;
        q : inout STD_LOGIC);
end component;
```

```
constant clock_period:time:=60 ns;
signal d1:STD_LOGIC:='0';
signal clk1:STD_LOGIC:='0';
signal q1:STD_LOGIC;
begin
uut: d_ff port map(d=>d1,clk=>clk1,q=>q1);
clk1<=not clk1 after clock_period/2;
stim_proc:process
begin
wait for 100 ns;
d1<='1';
wait for 100 ns;
d1<='0';
wait;
end process;
end Behavioral;
```

4.Waveform

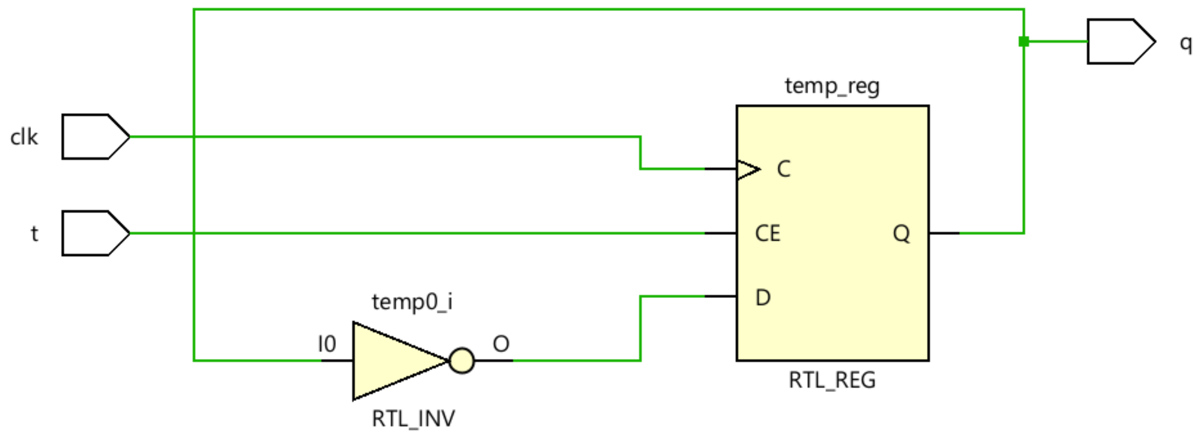


T Flip-Flop (Behavioral Model)

1.VHDL Design

```
entity t_ff is
   Port (
      t   : in  STD_LOGIC;
      clk : in  STD_LOGIC;
      q   : out STD_LOGIC
   );
end t_ff;
architecture Behavioral of t_ff is
   signal temp : STD_LOGIC := '0';
begin
   process(clk)
   begin
      if(clk'event and clk='1') then
         if t = '1' then
            temp <= not temp;
         end if;
      end if;
   end process;
   q <= temp;
end Behavioral;
```

2.RTL Design

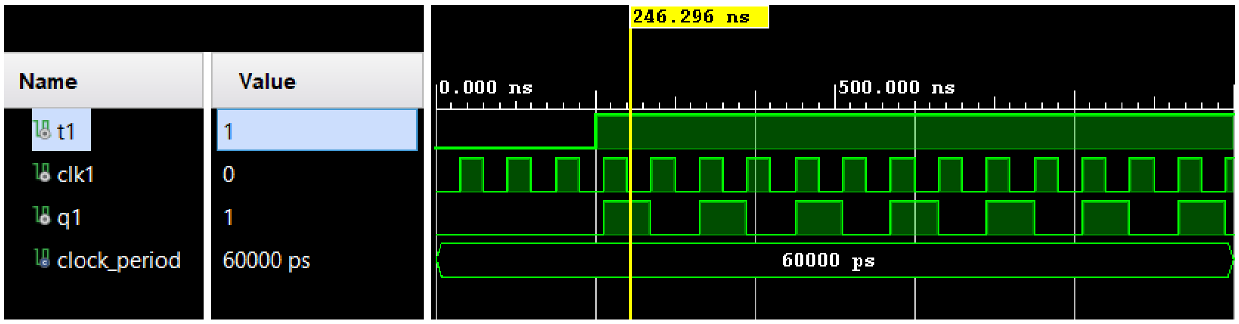


3.TBW Code

```
entity t_ff_tbw is
-- Port ( );
```

```
end t_ff_tbw;

architecture Behavioral of t_ff_tbw is
component t_ff is
   Port ( t : in STD_LOGIC;
        clk : in STD_LOGIC;
        q : out STD_LOGIC);
end component;
constant clock_period:time:=60 ns;
signal t1:STD_LOGIC:='0';
signal clk1:STD_LOGIC:='0';
signal q1:STD_LOGIC;
begin
uut: t_ff port map(t=>t1,clk=>clk1,q=>q1);
clk1<=not clk1 after clock_period/2;
stim_proc:process
begin
wait for 100 ns;
t1<='0';
wait for 100 ns;
t1<='1';
wait;
end process;
end Behavioral;
```

4.Waveform

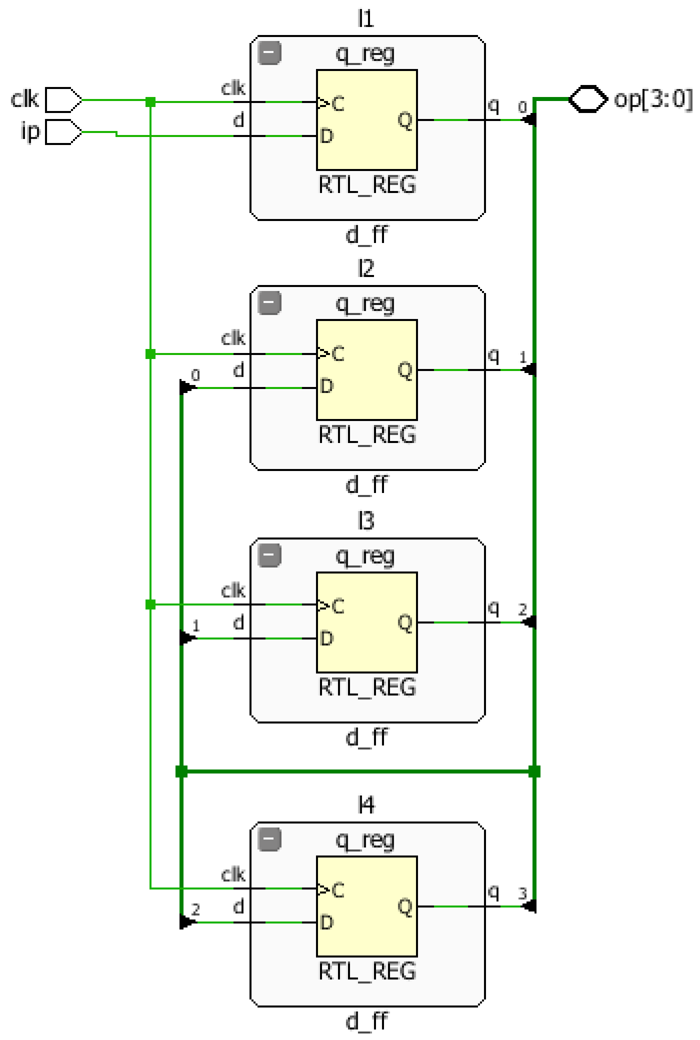


4-bit Shift Register (Structural Model)

1.VHDL Design

```
entity shift_register is
   Port ( ip : in STD_LOGIC;
        clk : in STD_LOGIC;
        op : inout STD_LOGIC_VECTOR (3 downto 0));
end shift_register;

architecture Structural of shift_register is
component d_ff is
   Port ( d : in STD_LOGIC;
        clk : in STD_LOGIC;
        q : inout STD_LOGIC);
end component;

begin
l1: d_ff port map(ip,clk,op(0));
l2: d_ff port map(op(0),clk,op(1));
l3: d_ff port map(op(1),clk,op(2));
l4: d_ff port map(op(2),clk,op(3));

end Structural;
```

2.RTL Design

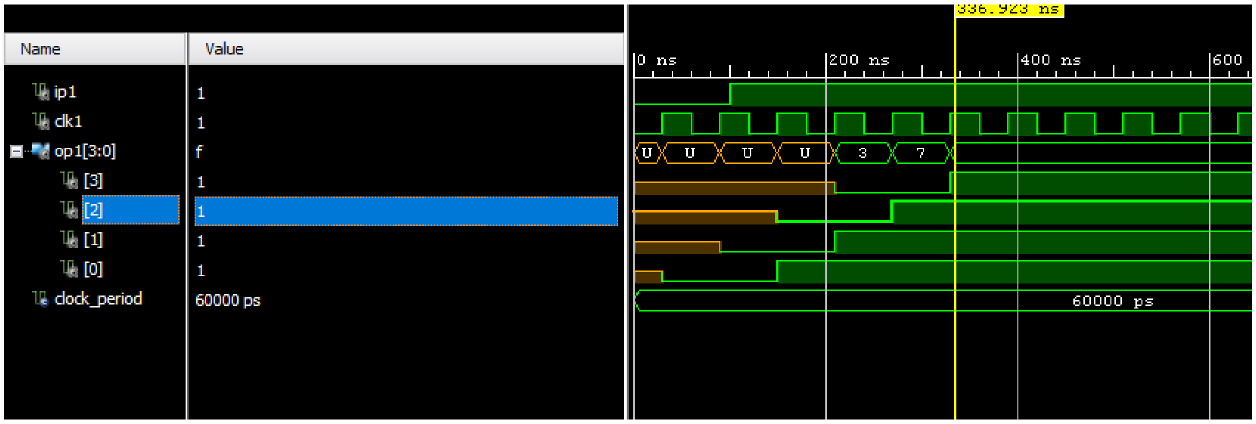


3.TBW Code

```
entity shift_register_tbw is
--  Port ( );
end shift_register_tbw;
architecture Behavioral of shift_register_tbw is
component shift_register is
   Port ( ip : in STD_LOGIC;
         clk : in STD_LOGIC;
         op : inout STD_LOGIC_VECTOR (3 downto 0));
end component;
constant clock_period:time:=60 ns;
```

```
signal ip1:STD_LOGIC:='0';
signal clk1:STD_LOGIC:='0';
signal op1:STD_LOGIC_VECTOR(3 downto 0);
begin
uut: shift_register port map(ip=>ip1,clk=>clk1,op=>op1);
clk1<=not clk1 after clock_period/2;
stim_proc:process
begin
wait for 100 ns;
ip1<='1';
wait;
end process;
end Behavioral;
```

4.Waveform



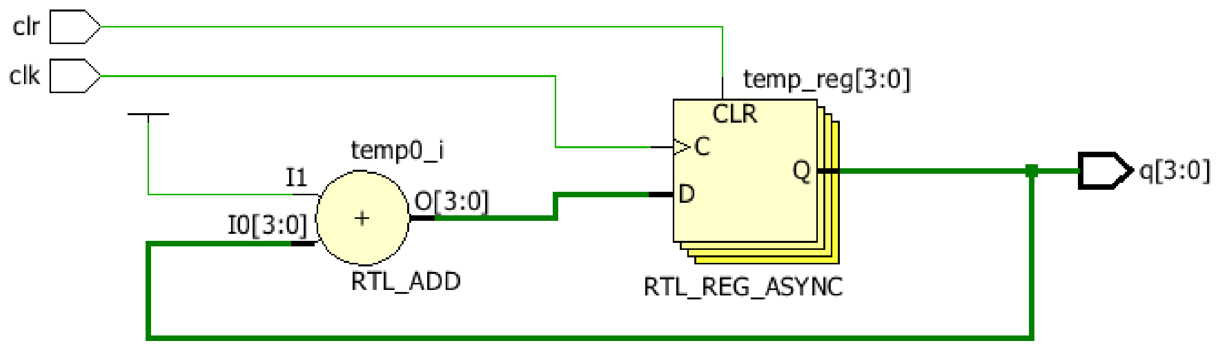
DAY 9

4-bit Up Counter (Behavioral Model)

1.VHDL Design

```
use IEEE.NUMERIC_STD.ALL;
entity bitup_4 is
   Port ( clr : in STD_LOGIC;
        clk : in STD_LOGIC;
        q : out unsigned(3 downto 0));
end bitup_4;
architecture Behavioral of bitup_4 is
signal temp:unsigned(3 downto 0):="0000";
begin
process(clr,clk)
begin
if(clr='1') then
temp<="0000";
else
if (clk'event and clk='1') then
temp<=temp+"0001";
end if;
end if;
q<=temp;
end process;
end Behavioral;
```

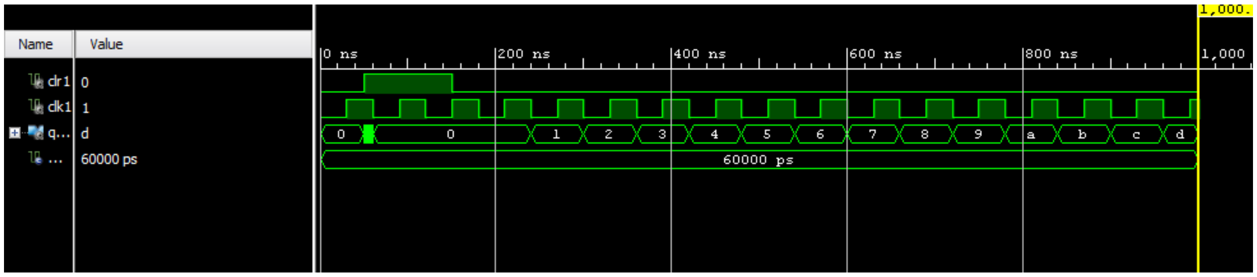
2.RTL Design



3.TBW Code

```
entity bit4_up_tbw is
-- Port ( );
end bit4_up_tbw;
architecture Behavioral of bit4_up_tbw is
component bitup_4 is
  Port ( clr : in STD_LOGIC;
       clk : in STD_LOGIC;
       q : out unsigned(3 downto 0));
end component;
constant clock_period:time:=60 ns;
signal clr1:STD_LOGIC:='0';
signal clk1:STD_LOGIC:='0';
signal q1:UNSIGNED(3 downto 0);
begin
uut: bitup_4 port map(clr=>clr1,clk=>clk1,q=>q1);
clk1<=not clk1 after clock_period/2;
stim_proc:process
begin
wait for 50 ns;
clr1<='1';
wait for 100 ns;
clr1<='0';
wait;
end process;
end Behavioral;
```

4.Waveform

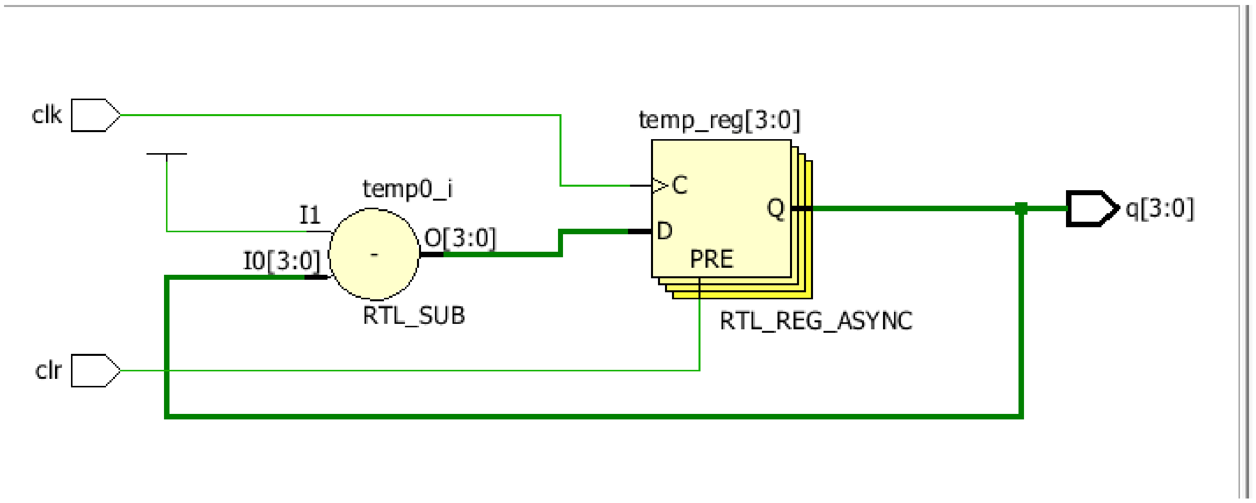


4-bit Down Counter (Behavioral Model)

1.VHDL Design

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.NUMERIC_STD.ALL;
entity bit4_down is
  Port ( clr : in STD_LOGIC;
       clk : in STD_LOGIC;
       q : out unsigned (3 downto 0));
end bit4_down;

architecture Behavioral of bit4_down is
signal temp:unsigned (3 downto 0):="0000";
begin
process(clr,clk)
begin
if(clr='1') then
temp<="1111";
else
if (clk'event and clk='1') then
temp<=temp-"0001";
end if;
end if;
q<=temp;
end process;
end Behavioral;
```

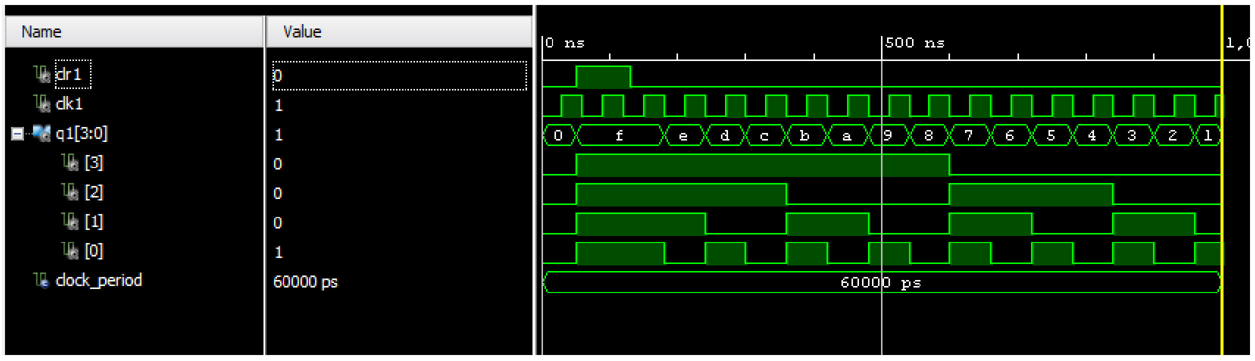
2.RTL Design



3.TBW Code

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.NUMERIC_STD.ALL;
entity bit4_down_tbw is
--  Port ( );
end bit4_down_tbw;

architecture Behavioral of bit4_down_tbw is
component bit4_down is
   Port ( clr : in STD_LOGIC;
          clk : in STD_LOGIC;
          q : out unsigned (3 downto 0));
end component;
constant clock_period:time:=60 ns;
signal clr1:STD_LOGIC:='0';
signal clk1:STD_LOGIC:='0';
signal q1:UNSIGNED(3 downto 0);
begin
uut: bit4_down port map(clr=>clr1,clk=>clk1,q=>q1);
clk1<=not clk1 after clock_period/2;
stim_proc:process
begin
wait for 50 ns;
clr1<='1';
wait for 80 ns;
clr1<='0';
wait;
end process;
end Behavioral;
```

4.Waveform



Name
Value
clr1
clk1
q1[3:0]
[3]
[2]
[1]
[0]
clock_period
0
1
1
0
0
0
1
60000 ps
0 ns
500 ns
0 f e d c b a 9 8 7 6 5 4 3 2 1
60000 ps