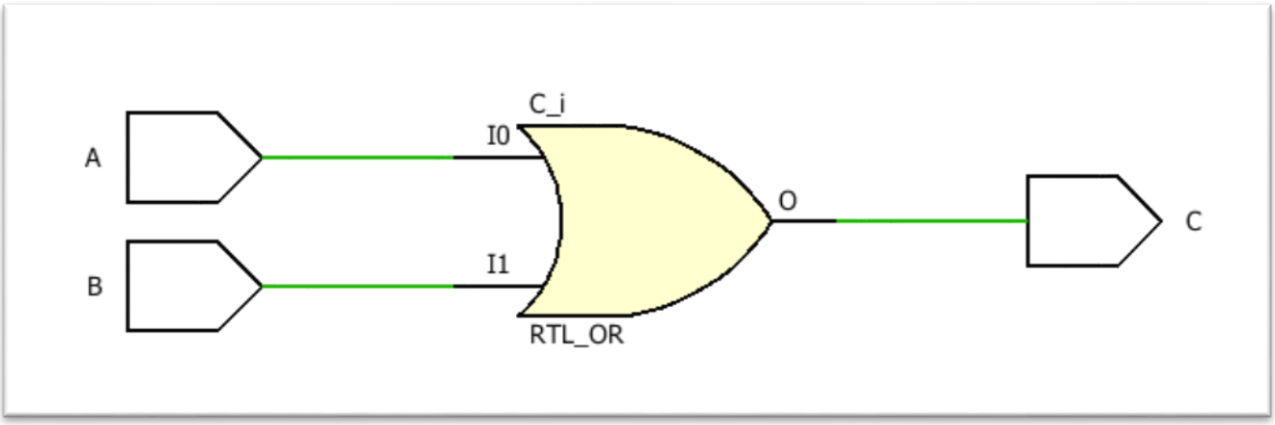
OR GATE DATAFLOW MODEL

VHD code

```
entity OR_GATE is
    Port ( A : in STD_LOGIC;
           B : in STD_LOGIC;
           C : out STD_LOGIC);
end OR_GATE;
architecture DATAFLOW of OR_GATE is
begin
C<= A OR B;
end DATAFLOW;
```

SCHEMEATIC DIAGRAM

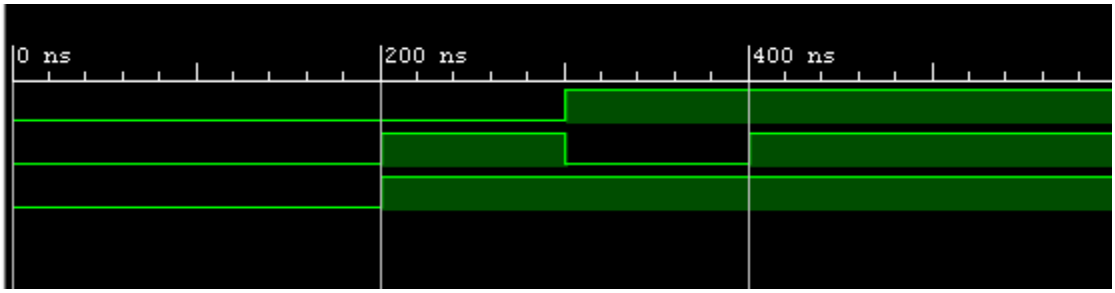


TBW Code

```
entity or_gate_flow is
--  Port ( );
end or_gate_flow;
architecture DATAFLOW of or_gate_flow is
component OR_GATE 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_GATE 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;
```

TBW Waveform

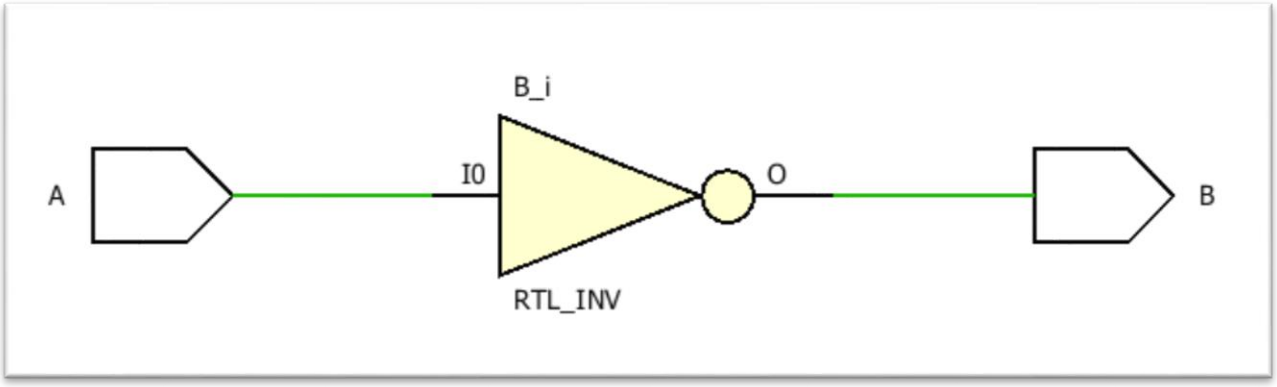


NOT GATE DATAFLOW MODEL

VHD code

```
entity NOT_GATE is
    Port ( A : in STD_LOGIC;
           B : out STD_LOGIC);
end NOT_GATE;
architecture DATAFLOW of NOT_GATE is
begin
B<= NOT A;
end DATAFLOW;
```

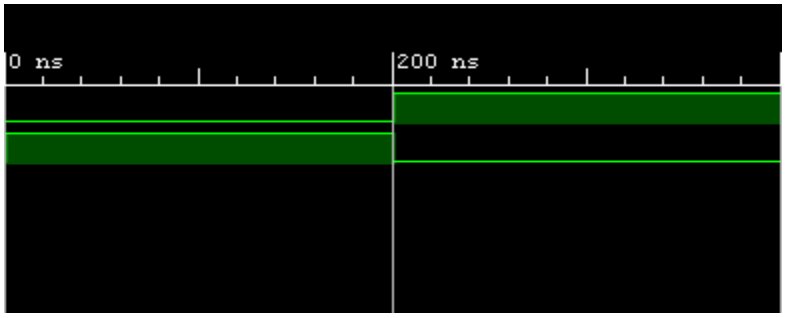
SCHEMEATIC DIAGRAM



TBW Code

```
entity not_gate_flow is

--  Port ( );

end not_gate_flow;

architecture DATAFLOW of  not_gate_flow is

component NOT_GATE is

    Port( A: in STD_LOGIC;

          B: out STD_LOGIC);

     end component;

    Signal A1:STD_LOGIC:='0';

    Signal B1:STD_LOGIC;

begin

UUT:NOT_GATE 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;
```

TBW Waveform

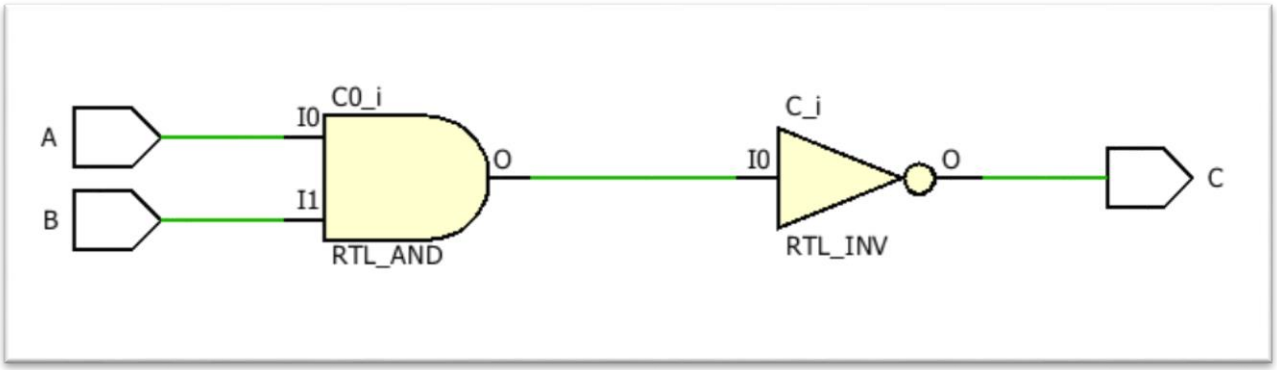


NAND GATE DATAFLOW MODEL

VHD code

```
entity NAND_GATE is
    Port ( A : in STD_LOGIC;
           B : in STD_LOGIC;
           C : out STD_LOGIC);
end NAND_GATE;
architecture DATAFLOW of NAND_GATE is
begin
c<=NOT( A AND B);
end DATAFLOW;
```

SCHEMEATIC DIAGRAM

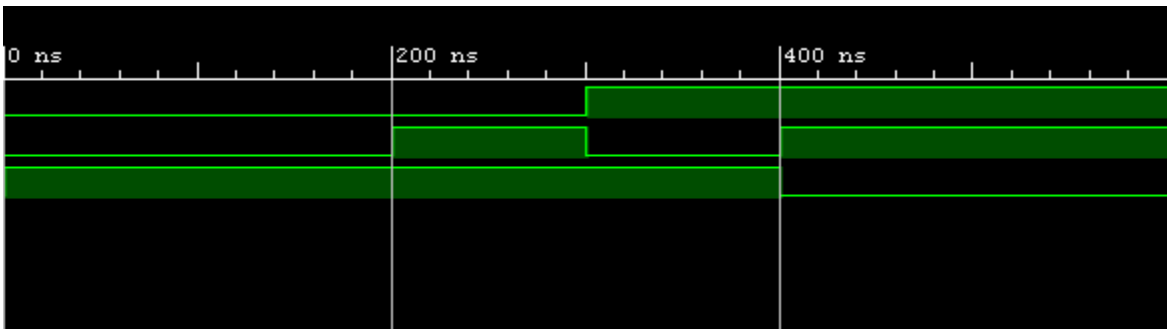


TBW Code

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

architecture DATAFLOW of nand_gate_flow is
component NAND_GATE 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_GATE 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;
```

TBW Waveform



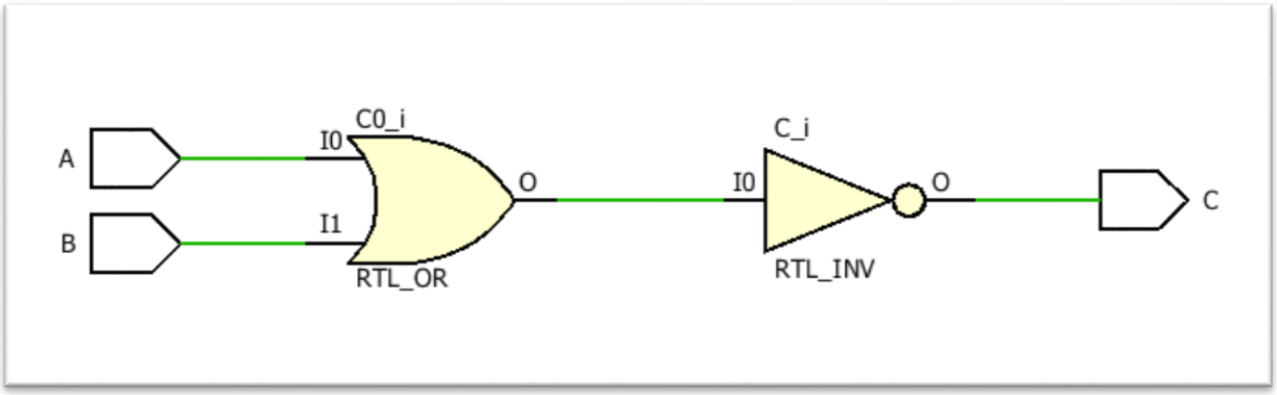
NOR GATE DATAFLOW MODEL

VHD code

```
entity NOR_GATE is
   Port ( A : in STD_LOGIC;
```

```
        B : in STD_LOGIC;
        C : out STD_LOGIC);
end NOR_GATE;
architecture DATAFLOW of NOR_GATE is
begin
C<= NOT(A OR B);
end DATAFLOW;
```

SCHEMEATIC DIAGRAM

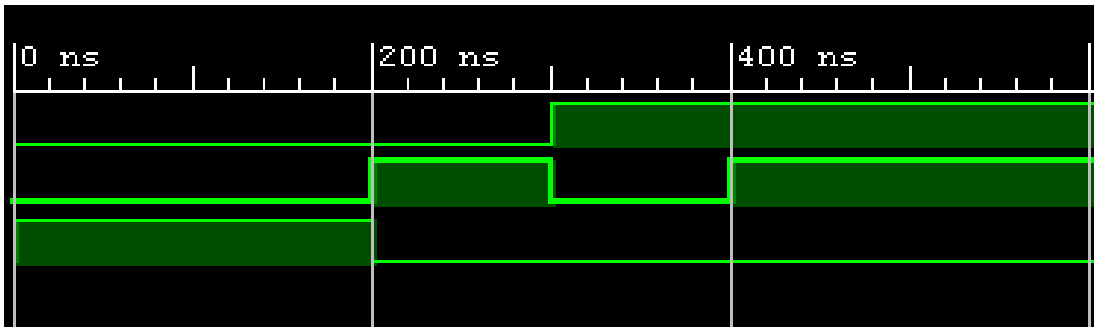


TBW Code

```
entity nor_gate_flow is
--  Port ( );
end nor _gate_flow;
architecture DATAFLOW of  nor_gate_flow is
component NOR_GATE 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_GATE 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;
```

TBW Waveform

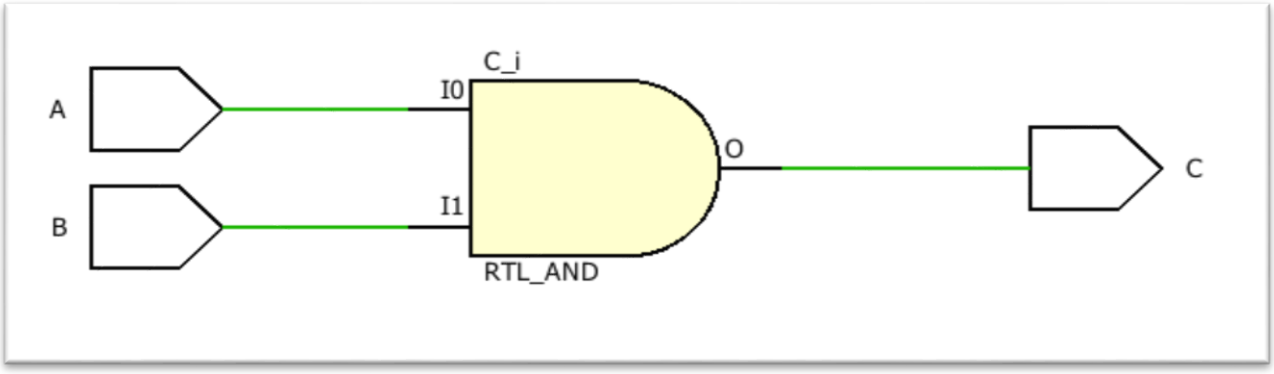


AND GATE DATAFLOW MODEL

VHD code

```
entity AND_GATE is

    Port ( A : in STD_LOGIC;

           B : in STD_LOGIC;

           C : out STD_LOGIC);

end AND_GATE;

architecture DATAFLOW of AND_GATE is

begin

C<= A AND B;

end DATAFLOW;
```

SCHEMEATIC DIAGRAM

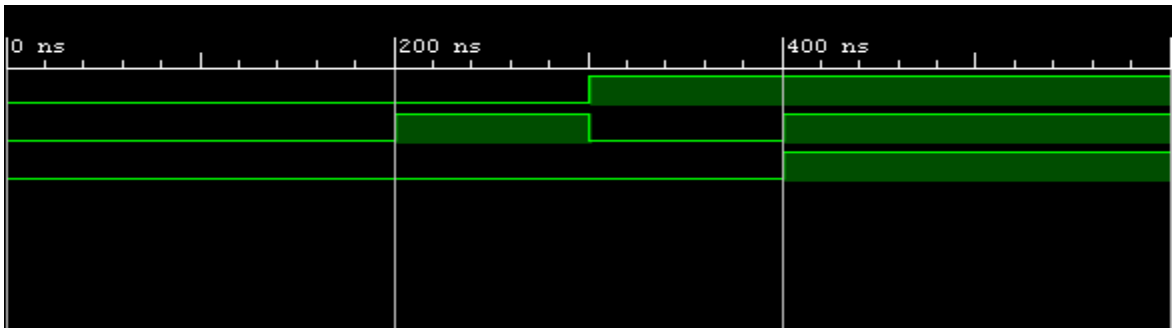


TBW Code

```
entity and_gate_flow is

--  Port ( );

end and_gate_flow;


architecture DATAFLOW of and_gate_flow is

component AND_GATE 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_GATE 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;
```

TBW Waveform

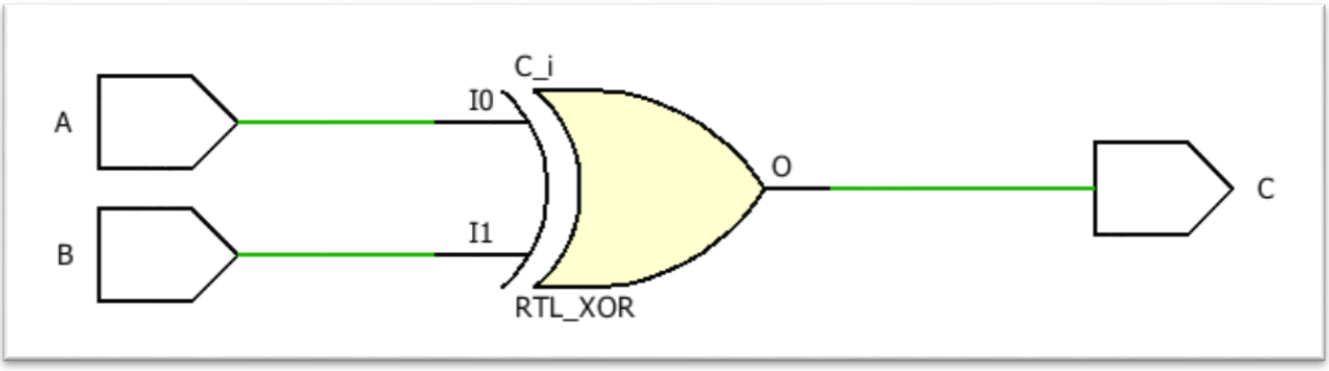


XOR GATE DATAFLOW MODEL

VHD code

```
entity XOR_GATE is

   Port ( A : in STD_LOGIC;

          B : in STD_LOGIC;

          C : out STD_LOGIC);

end XOR_GATE;

architecture DATAFLOW of XOR_GATE is

begin

C<= A XOR B;

end DATAFLOW;
```

SCHEMEATIC DIAGRAM

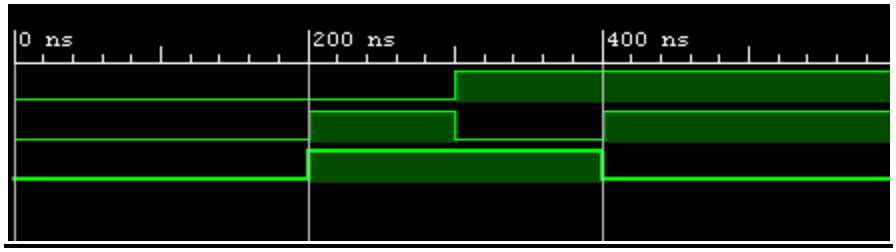


TBW Code

```
entity xor_gate_flow is

--  Port ( );
```

```
end xor_gate_flow;

architecture DATAFLOW of xor_gate_flow is

component XOR_GATE 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_GATE 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;
```

TBW Waveform

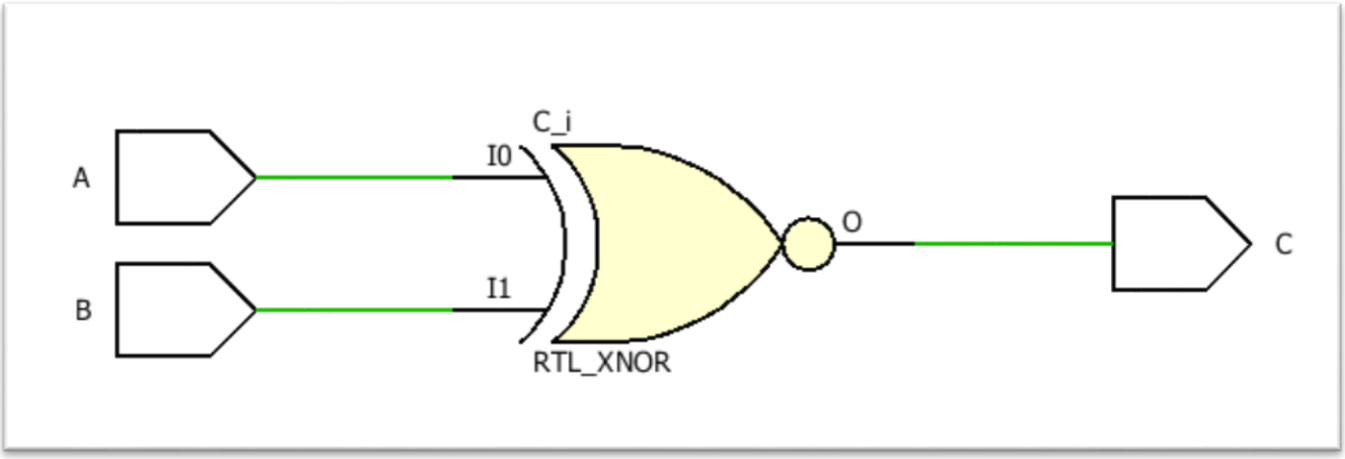


XNOR GATE DATAFLOW MODEL

VHD code

```
entity XNOR_GATE is
    Port ( A : in STD_LOGIC;
           B : in STD_LOGIC;
           C : out STD_LOGIC);
end XNOR_GATE;
architecture DATAFLOW of XNOR_GATE is
begin
C<= A XNOR B;
end DATAFLOW;
```

SCHEMEATIC DIAGRAM

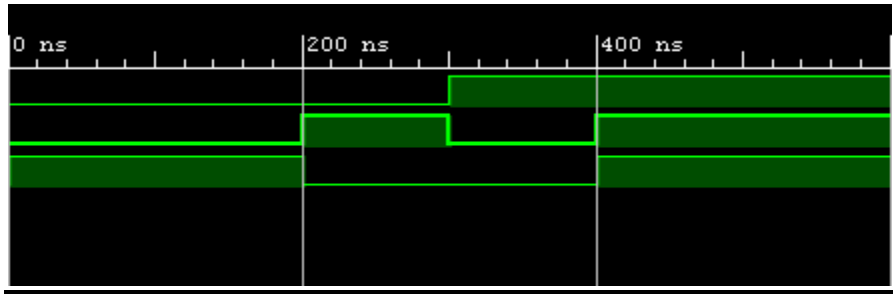


TBW Code

```
entity xnor_gate_flow is
--  Port ( );
end xnor_gate_flow;
architecture DATAFLOW of xnor_gate_flow is
component XNOR_GATE 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_GATE 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;
```

TBW Waveform

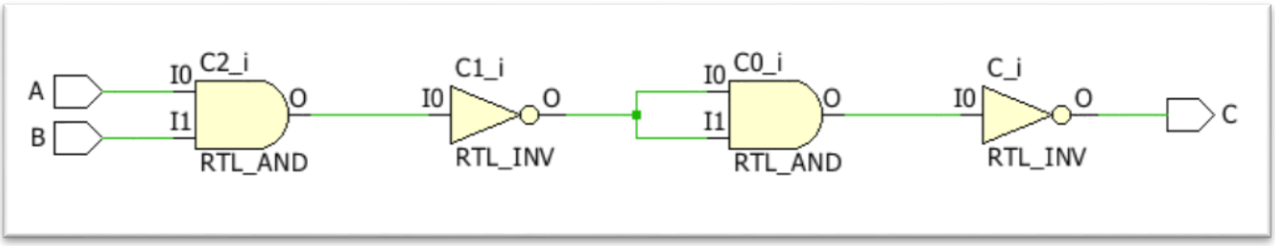


AND GATE FORM NAND GATE DATAFLOW MODEL

VHD code

```
entity NAND_AND is
    Port ( A : in STD_LOGIC;
           B : in STD_LOGIC;
           C : out STD_LOGIC);
end NAND_AND;
architecture Behavioral of NAND_AND is
begin
C<= (A NAND B) NAND (A NAND B);
end Behavioral;
```

SCHEMEATIC DIAGRAM

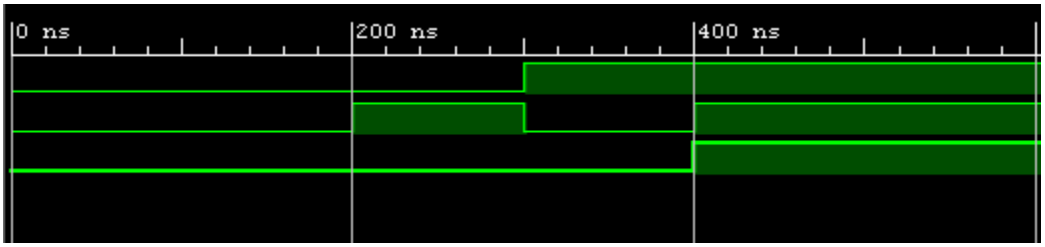


TBW Code

```
entity nand_and_gate_flow is
--  Port ( );
end nand_and_gate_flow;
architecture DATAFLOW of nand_and_gate_flow is
component NAND_AND 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_AND 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;
```

TBW Waveform

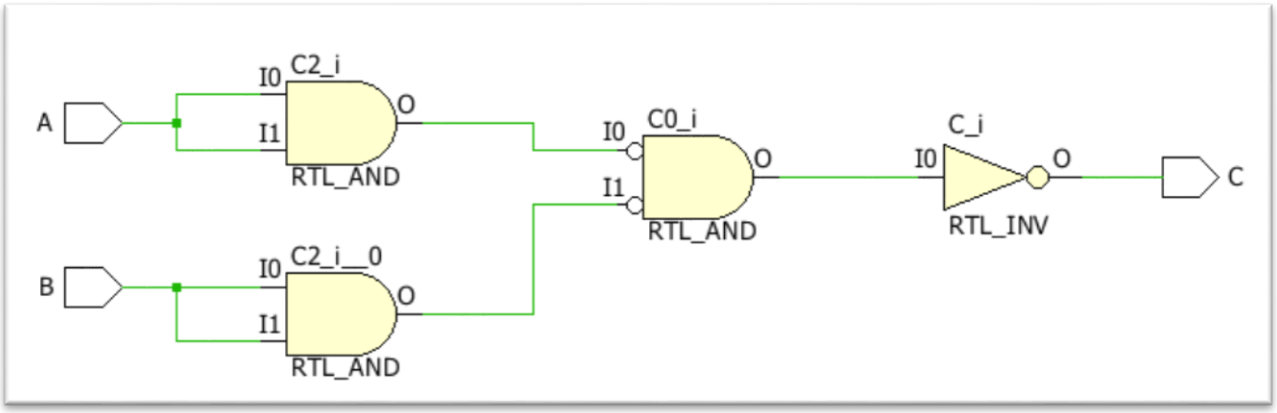


OR GATE FORM NAND GATE DATAFLOW MODEL

VHD code

```
entity NAND_OR is
    Port ( A : in STD_LOGIC;
           B : in STD_LOGIC;
           C : out STD_LOGIC);
end NAND_OR;
architecture Behavioral of NAND_OR is
begin
C<= (A NAND A) NAND (B NAND B);
end Behavioral;
```

SCHEMEATIC DIAGRAM

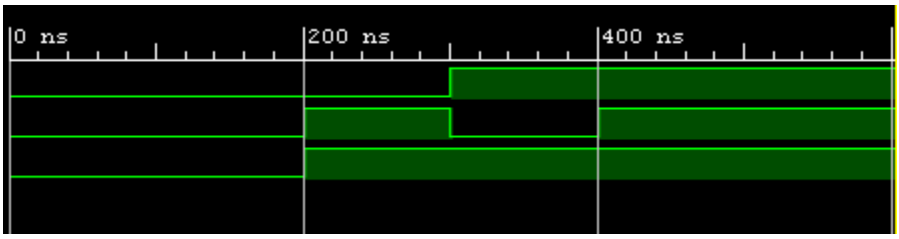


TBW Code

```
entity nand_or_gate_flow is

--  Port ( );

end nand_or_gate_flow;

architecture DATAFLOW of nand_or_gate_flow is

component NAND_OR 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_OR 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;
```

TBW Waveform

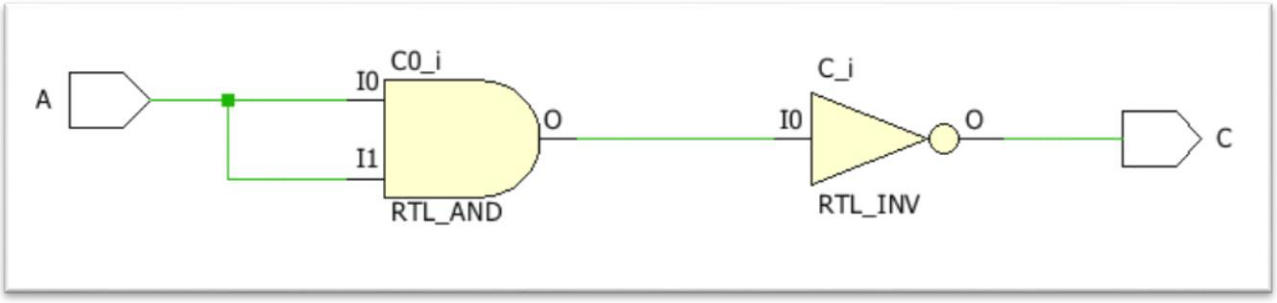


NOT GATE FORM NAND GATE DATAFLOW MODEL

VHD code

```
entity NAND_NOT is

    Port ( A : in STD_LOGIC;

           C : out STD_LOGIC);

end NAND_NOT;

architecture DATAFLOW of NAND_NOT is

begin

C<= A NAND A;

end DATAFLOW;
```

SCHEMEATIC DIAGRAM

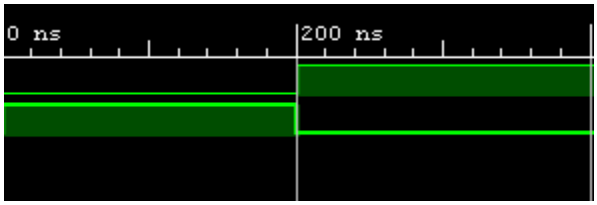


TBW Code

```
entity nand_not_gate_flow is

--  Port ( );

end nand_not_gate_flow;


architecture DATAFLOW of nand_not_gate_flow is

component NAND_NOT is
```

```
    Port( A: in STD_LOGIC;
          C: out STD_LOGIC);
     end component;
    Signal A1:STD_LOGIC:='0';
    Signal C1:STD_LOGIC;
begin
UUT:NAND_NOT port map(A=>A1,C=>C1);
Stim_proc:process
begin
wait for 100ns;
A1<='0';
wait for 100ns;
A1<='1';
wait;
end process;
end DATAFLOW;
```

TBW Waveform



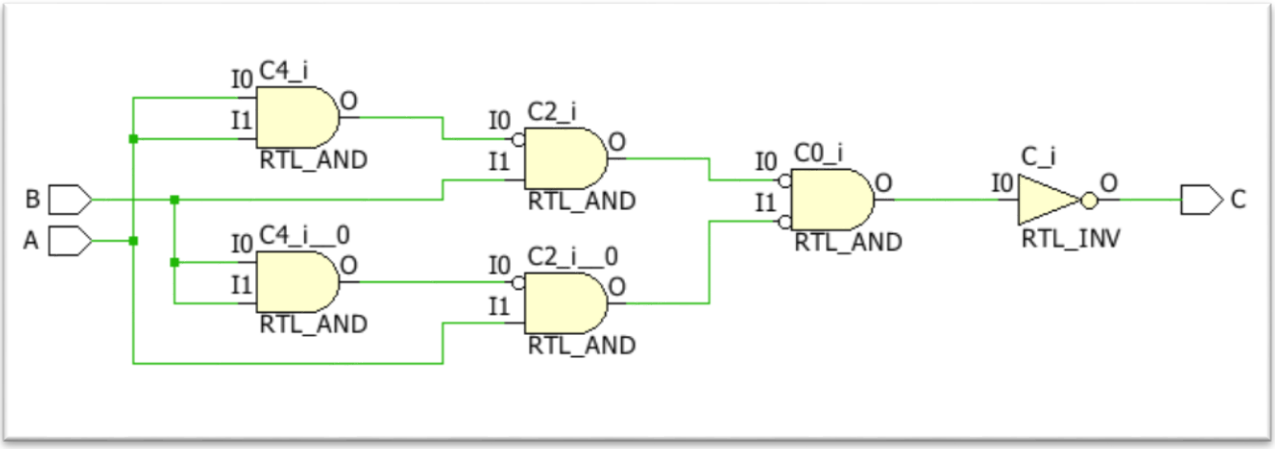
XOR GATE FORM NAND GATE DATAFLOW MODEL

VHD code

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

architecture DATAFLOW of NAND_XOR is
begin
C<= ((A NAND A) NAND B) NAND ((B NAND B) NAND A);
```

end DATAFLOW;

SCHEMEATIC DIAGRAM

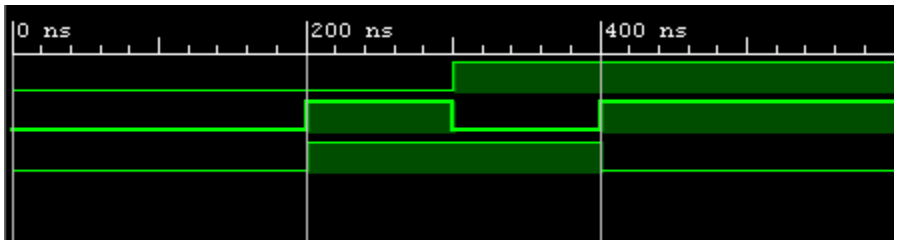


TBW Code

```
entity nand_xor_gate_flow is

--  Port ( );

end nand_xor_gate_flow;

architecture DATAFLOW of nand_xor_gate_flow is

component NAND_XOR 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_XOR 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;
```

TBW Waveform

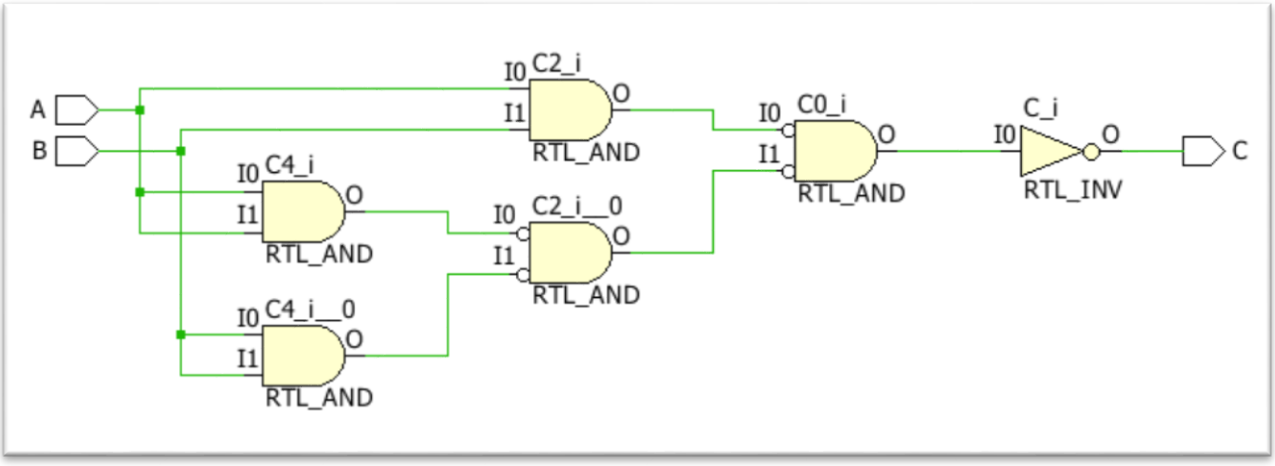


XNOR GATE FORM NAND GATE DATAFLOW MODEL

VHD code

```
entity NAND_XNOR is

    Port ( A : in STD_LOGIC;

           B : in STD_LOGIC;

           C : out STD_LOGIC);

end NAND_XNOR;

architecture DATAFLOW of NAND_XNOR is

begin

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

end DATAFLOW;
```

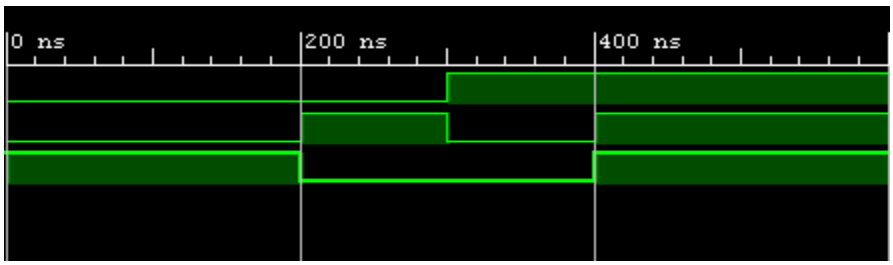
SCHEMEATIC DIAGRAM



TBW Code

```
entity nand_xnor_gate_flow is

--  Port ( );

end nand_xnor_gate_flow;

architecture DATAFLOW of nand_xnor_gate_flow is

component NAND_XNOR 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_XNOR 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;
```

TBW Waveform

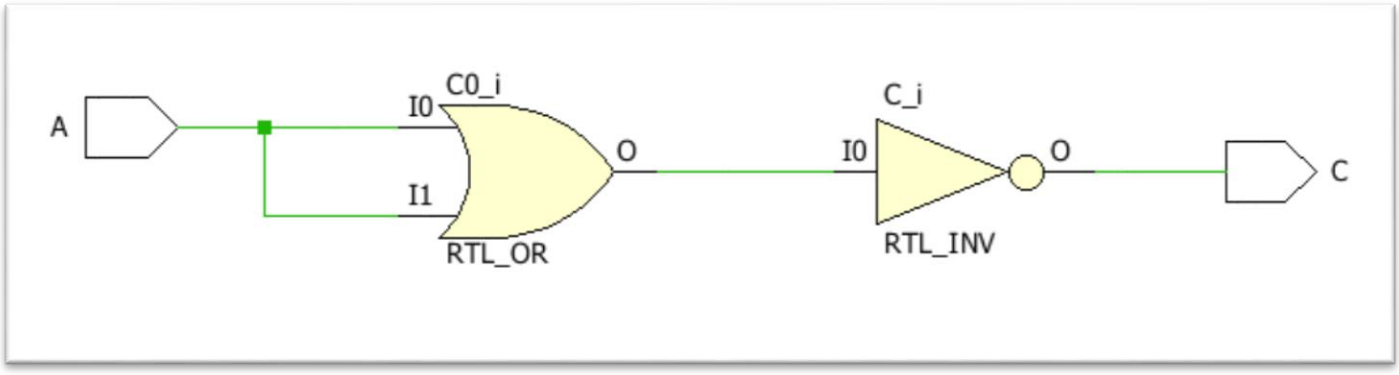


NOT GATE FORM NOR GATE DATAFLOW MODEL

VHD code

```
entity NOR_NOT is
    Port ( A : in STD_LOGIC;
           C : out STD_LOGIC);
end NOR_NOT;
architecture DATAFLOW of NOR_NOT is
begin
C<= A NOR A;
end DATAFLOW;
```

SCHEMEATIC DIAGRAM

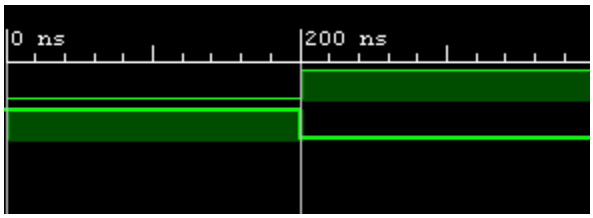


TBW Code

```
entity nor_not_gate_flow is
--  Port ( );
end nor_not_gate_flow;
architecture DATAFLOW of nor_not_gate_flow is
component NOR_NOT is
    Port( A: in STD_LOGIC;
          C: out STD_LOGIC);
     end component;
    Signal A1:STD_LOGIC:='0';
    Signal C1:STD_LOGIC;
begin
UUT:NOR_NOT port map(A=>A1,C=>C1);
Stim_proc:process
```

```
begin
wait for 100ns;
A1<='0';
wait for 100ns;
A1<='1';
wait;
end process;
end DATAFLOW;
```

TBW Waveform

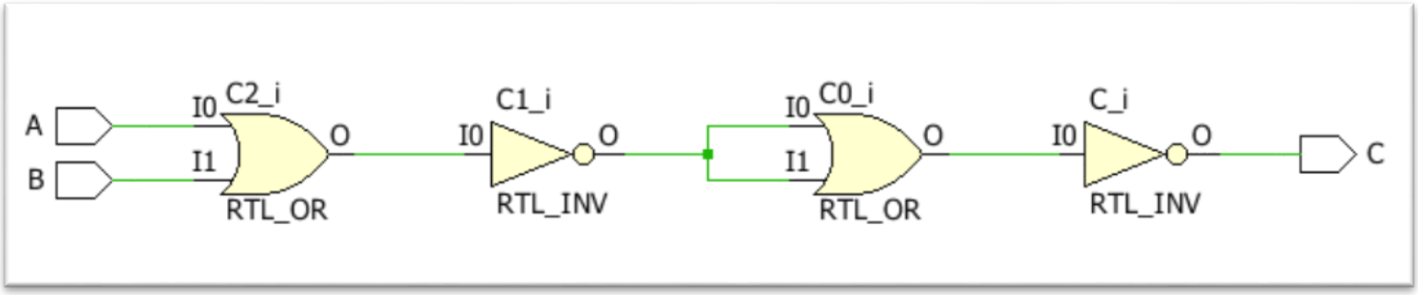


OR GATE FORM NOR GATE DATAFLOW MODEL

VHD code

```
entity NOR_OR is
    Port ( A : in STD_LOGIC;
           B : in STD_LOGIC;
           C : out STD_LOGIC);
end NOR_OR;
architecture DATAFLOW of NOR_OR is
begin
C<= (A NOR B) NOR (A NOR B);
end DATAFLOW;
```

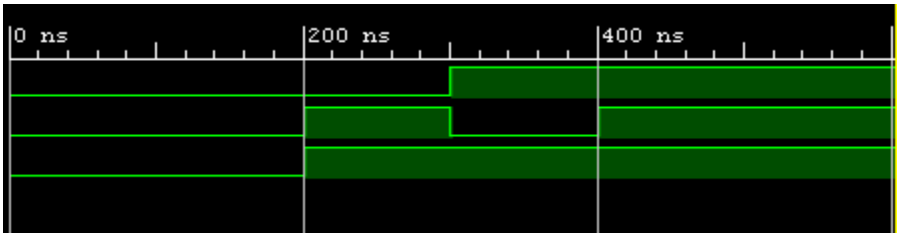
SCHEMEATIC DIAGRAM



TBW Code

```
entity nor_or_gate_flow is

--  Port ( );

end nor_or_gate_flow;

architecture DATAFLOW of nor_or_gate_flow is

component NOR_OR 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_OR 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;
```

TBW Waveform

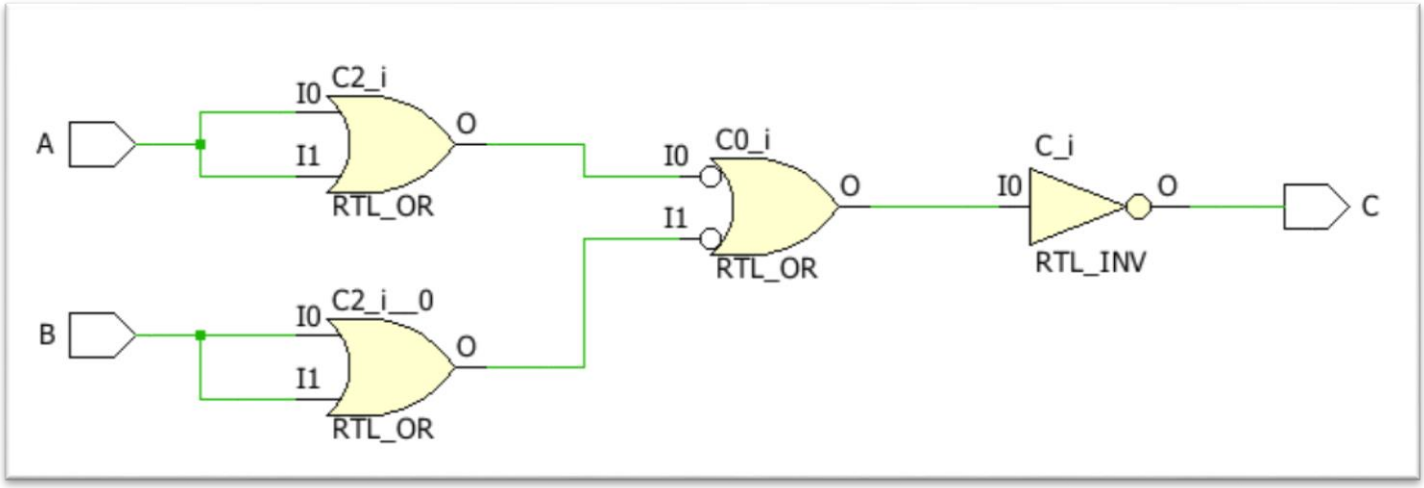


AND GATE FORM NOR GATE DATAFLOW MODEL

VHD code

```
entity NOR_AND is
    Port ( A : in STD_LOGIC;
           B : in STD_LOGIC;
           C : out STD_LOGIC);
end NOR_AND;
architecture DATAFLOW of NOR_AND is
begin
C<= (A NOR A) NOR (B NOR B);
end DATAFLOW;
```

SCHEMEATIC DIAGRAM

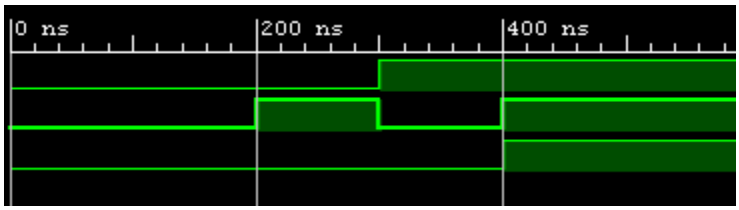


TBW Code

```
entity nor_and_gate_flow is
--  Port ( );
end nor_and_gate_flow;
architecture DATAFLOW of nor_and_gate_flow is
component NOR_AND 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_AND 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;
```

TBW Waveform



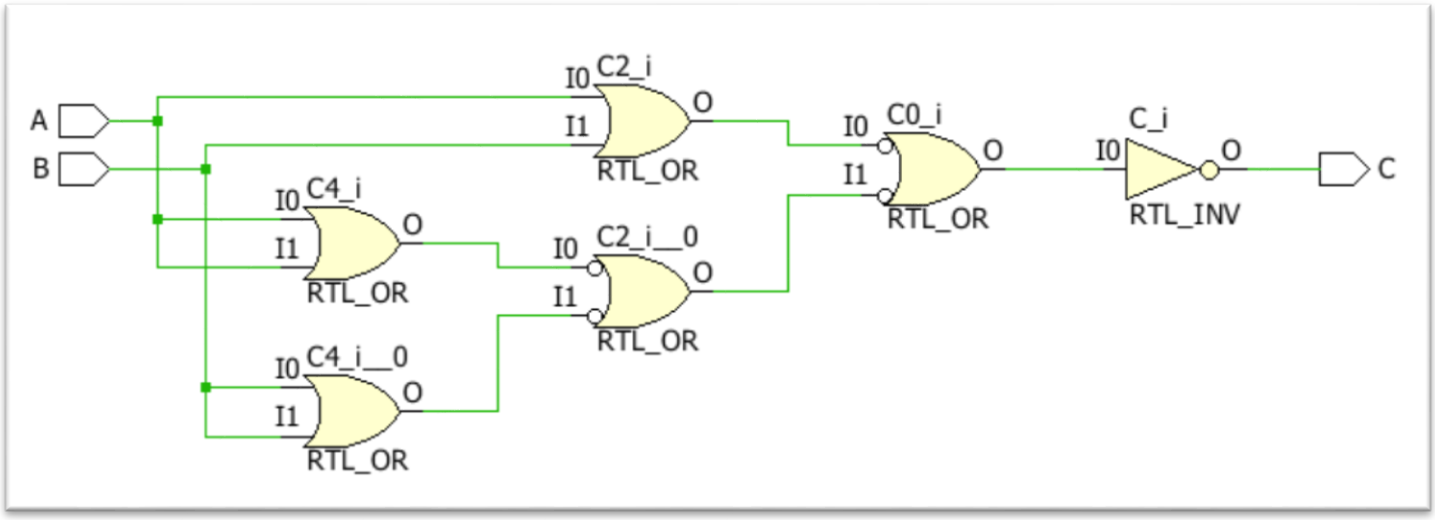
XOR GATE FORM NOR GATE DATAFLOW MODEL

VHD code

```
entity NOR_XOR is
    Port ( A : in STD_LOGIC;
           B : in STD_LOGIC;
           C : out STD_LOGIC);
end NOR_XOR;
architecture DATAFLOW of NOR_XOR is
begin
C<= (A NOR B) NOR ( (A NOR A) NOR (B NOR B) );
```

end DATAFLOW;

SCHEMEATIC DIAGRAM

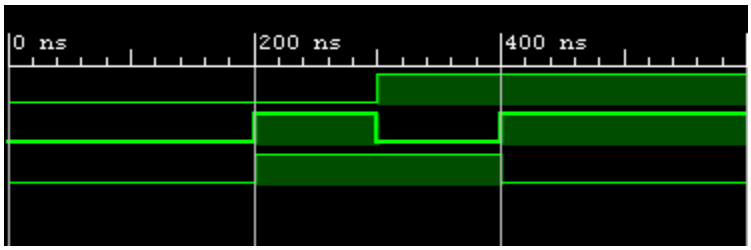


TBW Code

```
entity nor_xor_gate_flow is

--  Port ( );

end nor_xor_gate_flow;

architecture DATAFLOW of nor_xor_gate_flow is

component NOR_XOR 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_XOR 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;
```

TBW Waveform

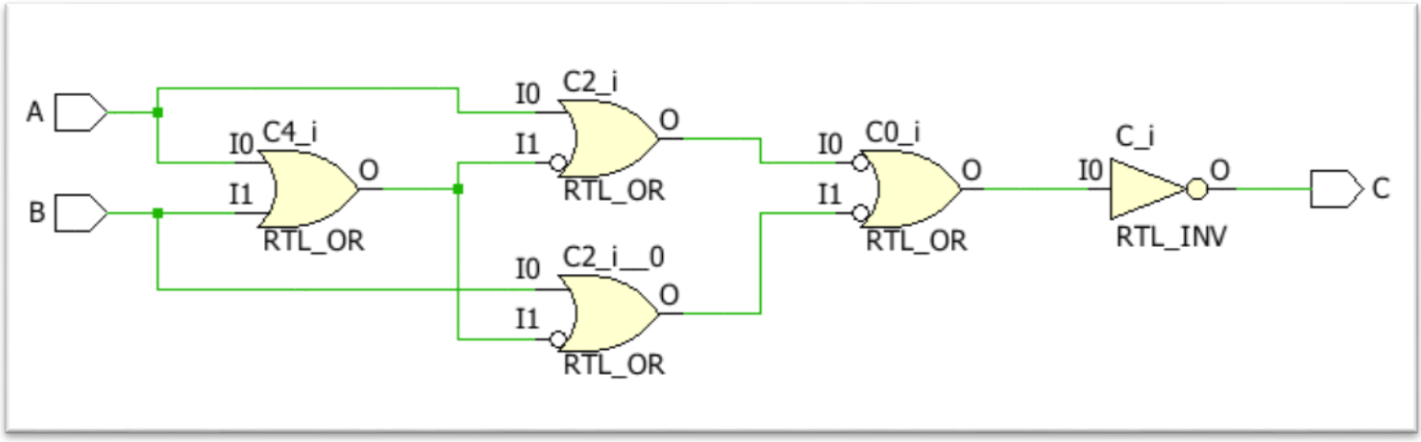


XNOR GATE FORM NOR GATE DATAFLOW MODEL

VHD code

```
entity NOR_XNOR is

    Port ( A : in STD_LOGIC;

           B : in STD_LOGIC;

           C : out STD_LOGIC);

end NOR_XNOR;

architecture DATAFLOW of NOR_XNOR is

begin

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

end DATAFLOW;
```

SCHEMEATIC DIAGRAM

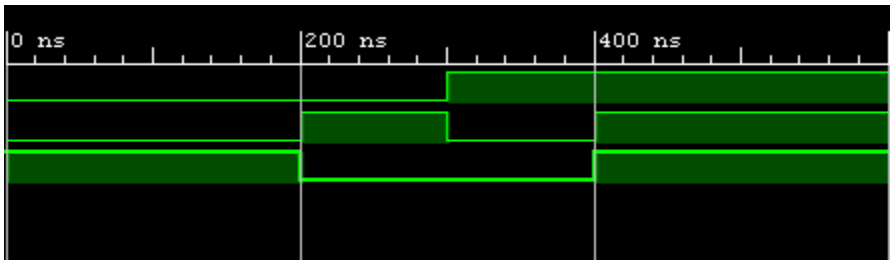


TBW Code

```
entity nor_xnor_gate_flow is

--  Port ( );

end nor_xnor_gate_flow;

architecture DATAFLOW of nor_xnor_gate_flow is

component NOR_XNOR 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_XNOR 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;
```

TBW Waveform

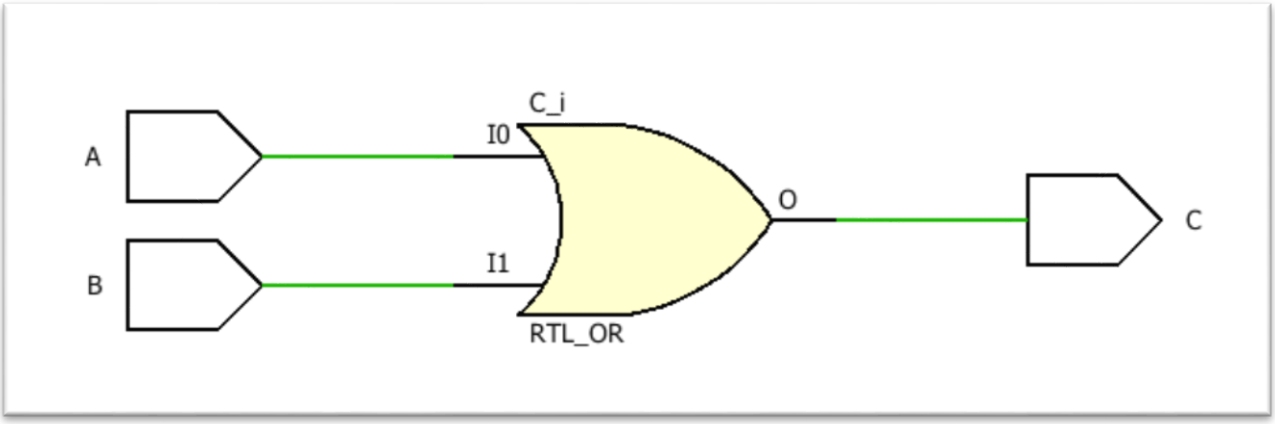


OR GATE BEHAVIORAL MODEL

VHD code

```
entity OR_GATE is
    Port ( A : in STD_LOGIC;
           B : in STD_LOGIC;
           C : out STD_LOGIC);
end OR_GATE;
architecture Behavioral of OR_GATE is
begin
process(A,B)
begin
if(A ='1' or B='1') then
    C<='1';
  else
    C<='0';
 end if;
 end process;
end Behavioral;
```

SCHEMEATIC DIAGRAM

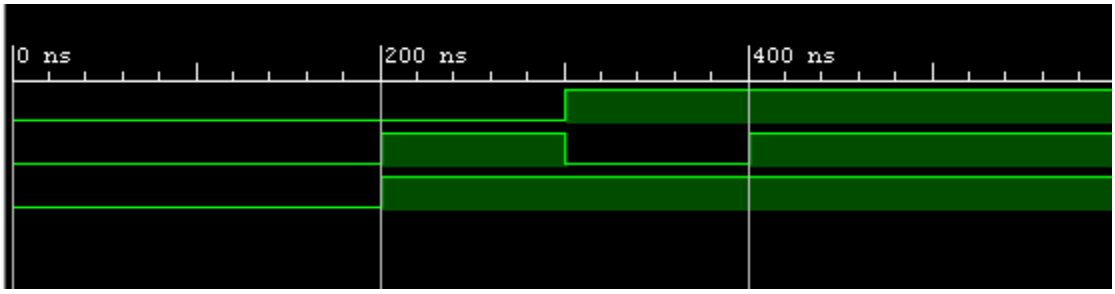


TBW Code

```
entity or_gate_flow is
--  Port ( );
end or_gate_flow;
architecture Behavioral of or_gate_flow is
component OR_GATE 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_GATE 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 Behavioral;
```

TBW Waveform



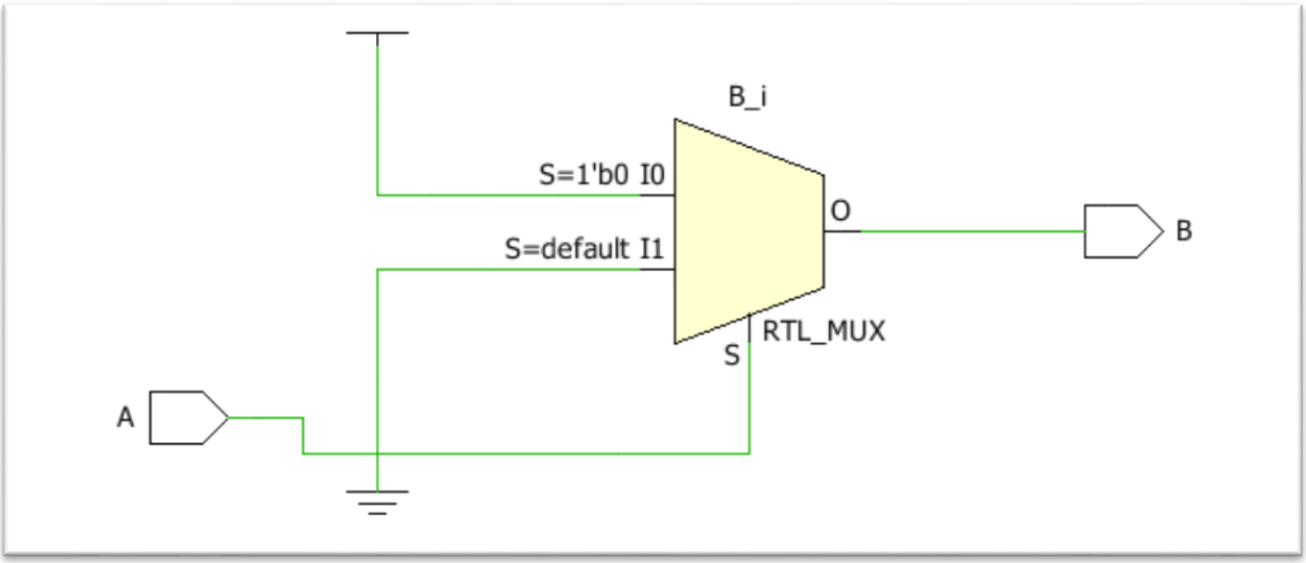
NOT GATE BEHAVIORAL MODEL

VHD code

```
entity NOT_GATE is
```

```
    Port ( A : in STD_LOGIC;
           B : out STD_LOGIC);
end NOT_GATE;
architecture Behavioral of NOT_GATE is
begin
process(A)
begin
if(A ='0') then
    B<='1';
  else
    B<='0';
 end if;
 end process;
end Behavioral;
```

SCHEMEATIC DIAGRAM

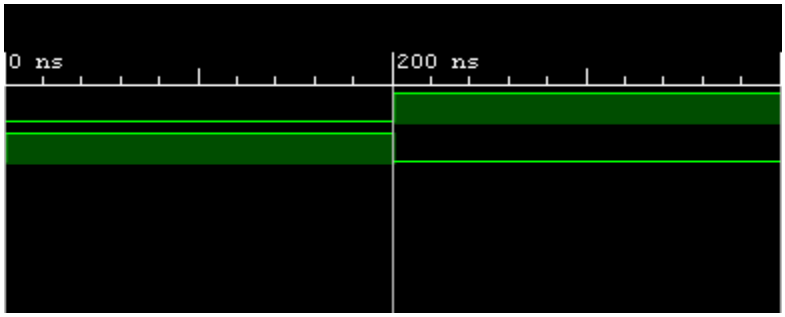


TBW Code

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

architecture Behavioral of not_gate_flow is
component NOT_GATE is
    Port( A: in STD_LOGIC;
```

```
        B: out STD_LOGIC);
    end component;
   Signal A1:STD_LOGIC:='0';
   Signal B1:STD_LOGIC;
begin
UUT:NOT_GATE 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;
```

TBW Waveform



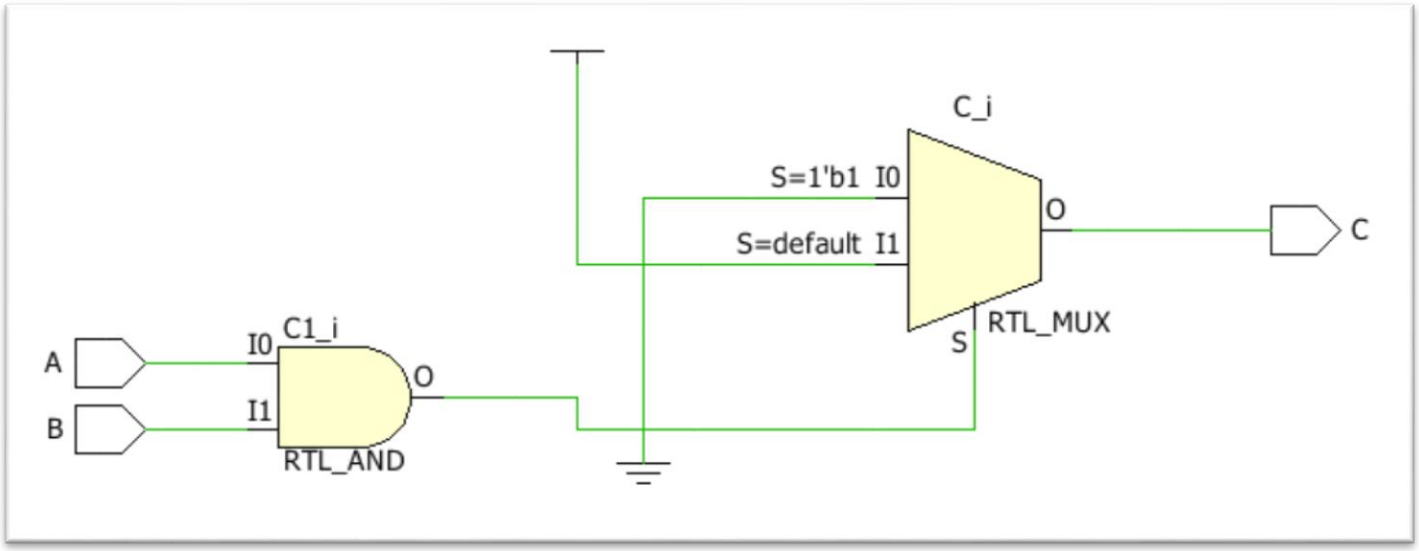
NAND GATE BEHAVIORAL MODEL

VHD code

```
entity NAND_GATE is
   Port ( A : in STD_LOGIC;
          B : in STD_LOGIC;
          C : out STD_LOGIC);
end NAND_GATE;
architecture Behavioral of NAND_GATE is
begin
process(A,B)
begin
```

```
if(A ='1' and B='1') then
    C<='0';
  else
    C<='1';
 end if;
 end process;
end Behavioral;
```

SCHEMEATIC DIAGRAM

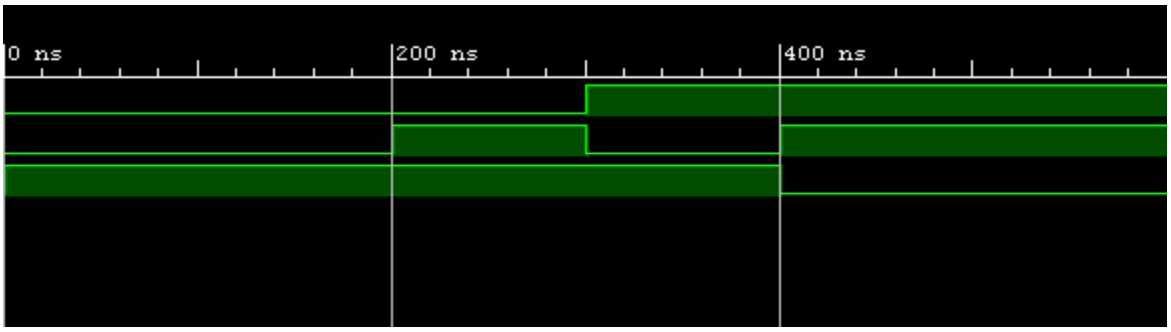


TBW Code

```
entity nand_gate_flow is
--  Port ( );
end nand_gate_flow;
architecture Behavioral of nand_gate_flow is
component NAND_GATE 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_GATE 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 Behavioral;
```

TBW Waveform



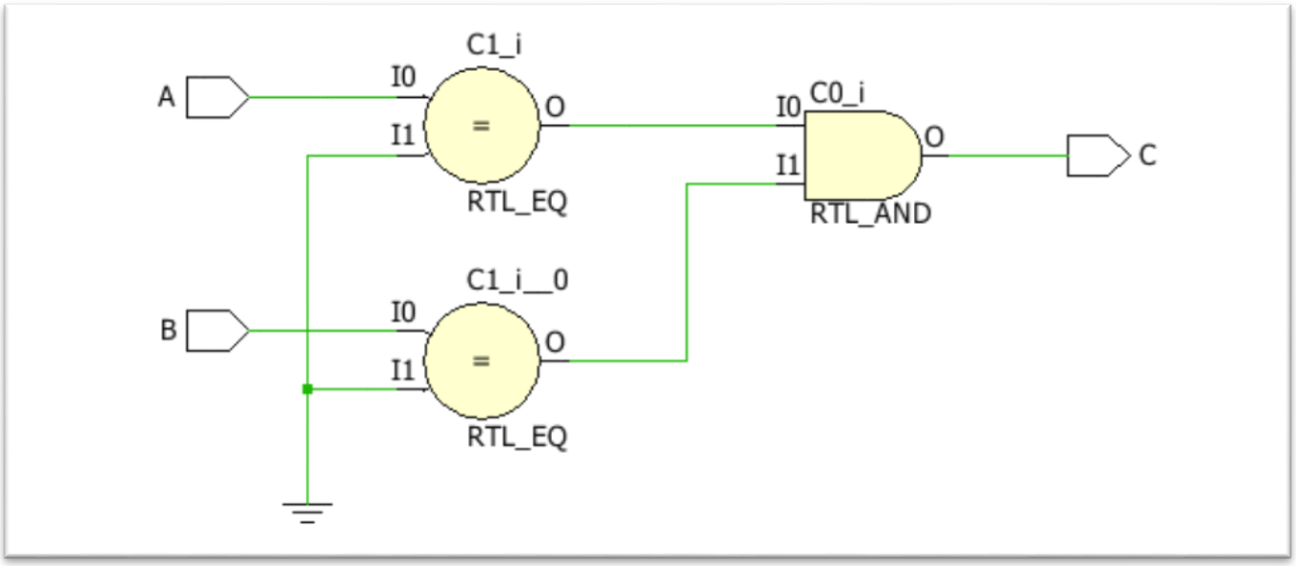
NOR GATE BEHAVIORAL MODEL

VHD code

```
entity NOR_GATE is
    Port ( A : in STD_LOGIC;
           B : in STD_LOGIC;
           C : out STD_LOGIC);
end NOR_GATE;
architecture Behavioral of NOR_GATE is
begin
process(A,B)
begin
if(A ='0' and B='0') then
```

```
    C<='1';
   else
    C<='0';
 end if;
 end process;
end Behavioral;
```

SCHEMEATIC DIAGRAM

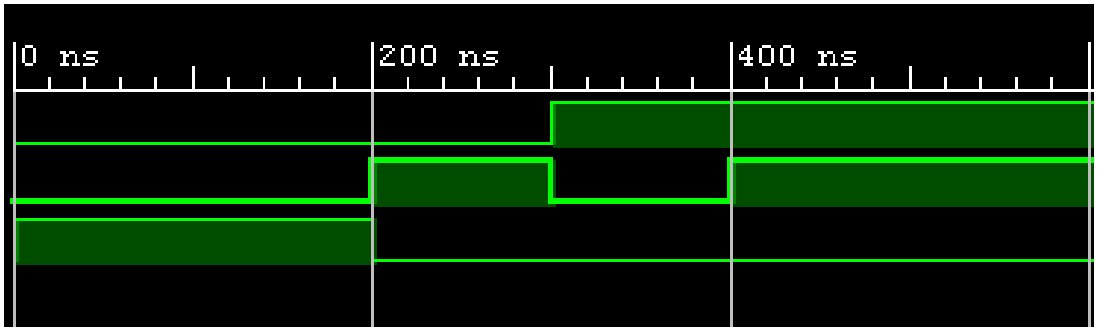


TBW Code

```
entity nor_gate_flow is
--  Port ( );
end nor_gate_flow;
architecture Behavioral of nor_gate_flow is
component NOR_GATE 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_GATE 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 Behavioral;
```

TBW Waveform



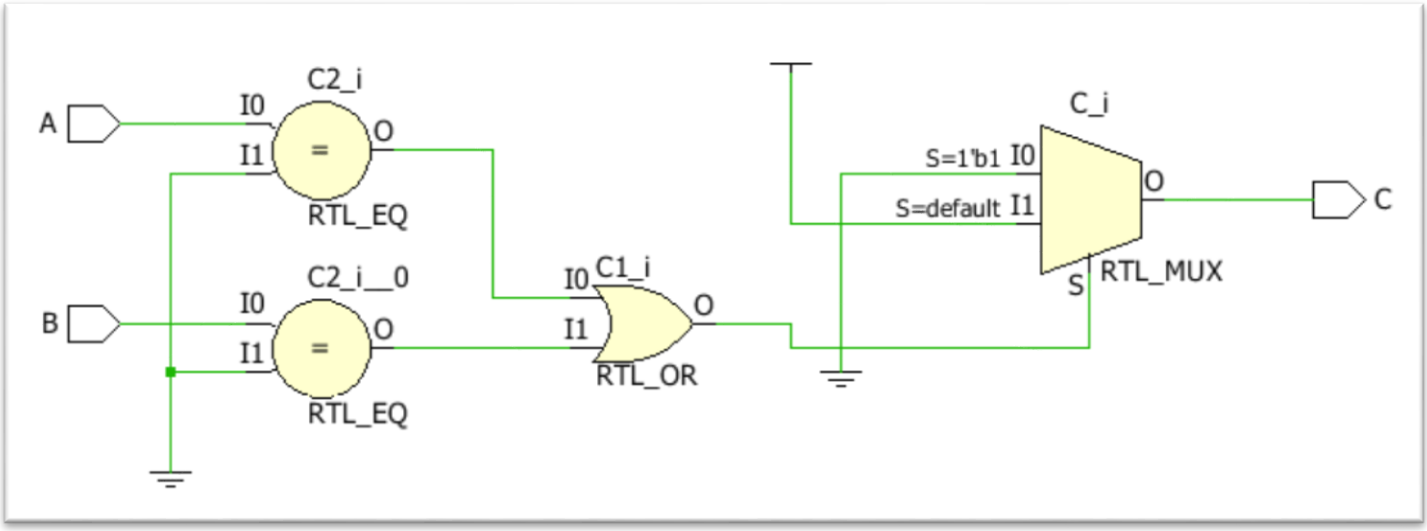
AND GATE BEHAVIORAL MODEL

VHD code

```
entity AND_GATE is
    Port ( A : in STD_LOGIC;
           B : in STD_LOGIC;
           C : out STD_LOGIC);
end AND_GATE;
architecture Behavioral of AND_GATE is
begin
process(A,B)
begin
if(A ='0' or B='0') then
    C<='0';
```

```
  else
    C<='1';
 end if;
 end process;
end Behavioral;
```

SCHEMEATIC DIAGRAM

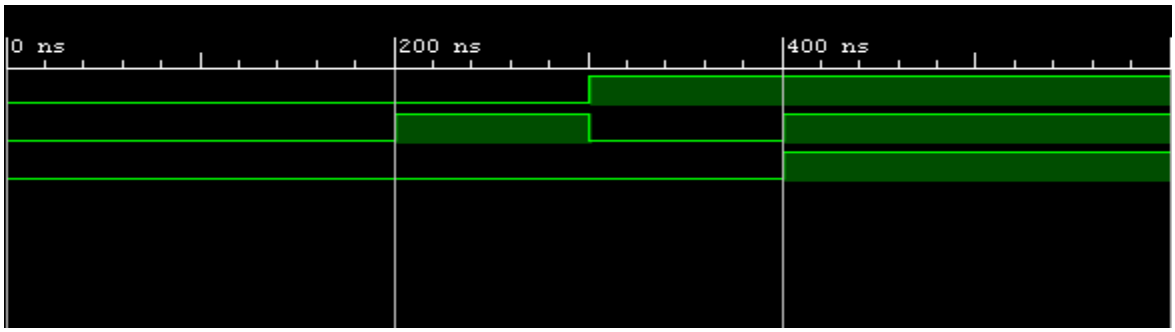


TBW Code

```
entity and_gate_flow is
--  Port ( );
end and_gate_flow;
architecture Behavioral of and_gate_flow is
component AND_GATE 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_GATE 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 Behavioral;
```

TBW Waveform



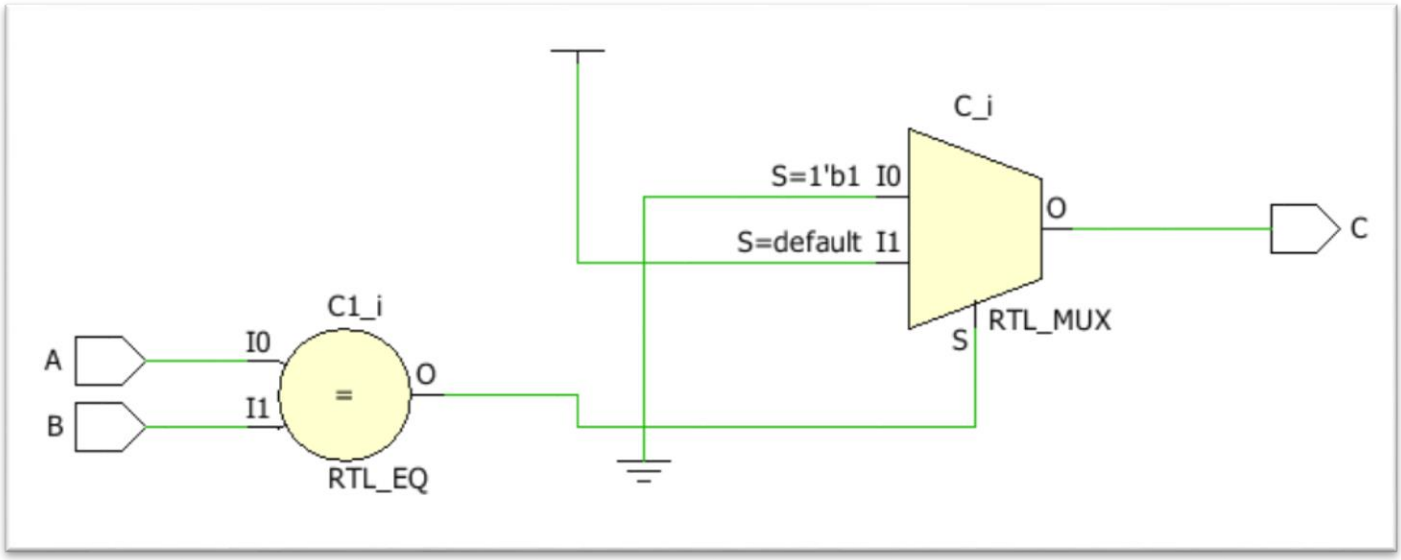
XOR GATE BEHAVIORAL MODEL

VHD code

```
entity XOR_GATE is

    Port ( A : in STD_LOGIC;

           B : in STD_LOGIC;

           C : out STD_LOGIC);

end XOR_GATE;

architecture Behavioral of XOR_GATE is

begin

process(A,B)

begin

if(A = B) then

    C<='0';

   else

    C<='1';

 end if;
```

```
 end process;

end Behavioral;
```

SCHEMEATIC DIAGRAM

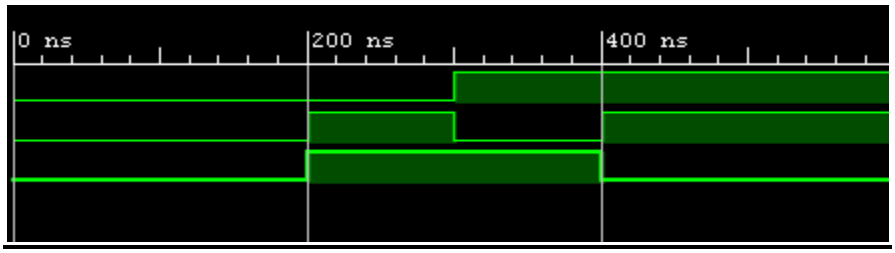


TBW Code

```
entity nor_gate_flow is

--  Port ( );

end nor_gate_flow;


architecture Behavioral of nor_gate_flow is

component XOR_GATE 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_GATE 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 Behavioral;
```

TBW Waveform

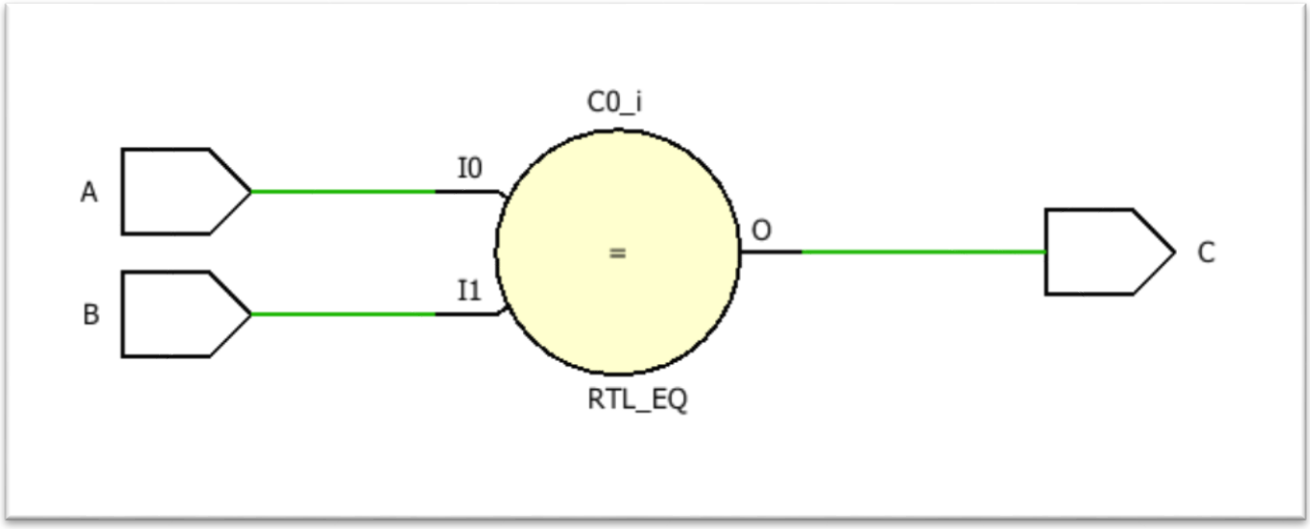


XNOR GATE BEHAVIORAL MODEL

VHD code

```
entity XNOR_GATE is

    Port ( A : in STD_LOGIC;

           B : in STD_LOGIC;

           C : out STD_LOGIC);

end XNOR_GATE;

architecture Behavioralof XNOR_GATE is

begin

process(A,B)

begin

if(A = B) then

    C<='1';

   else

    C<='0';

 end if;

 end process;

end Behavioral;
```

SCHEMEATIC DIAGRAM

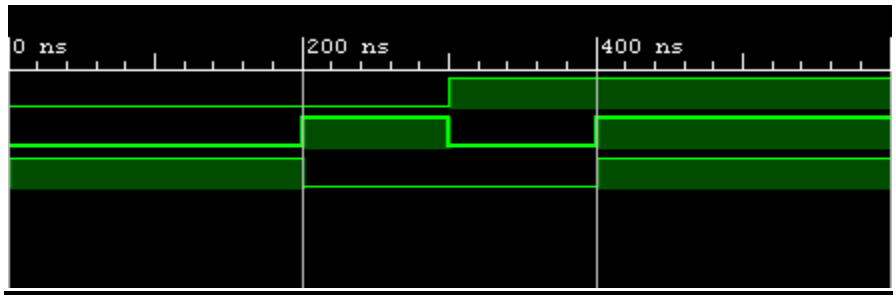


TBW Code

```
entity xnor_gate_flow is

--  Port ( );

end xnor_gate_flow;

architecture Behavioral of xnor_gate_flow is

component XNOR_GATE 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_GATE 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 Behavioral;
```

TBW Waveform

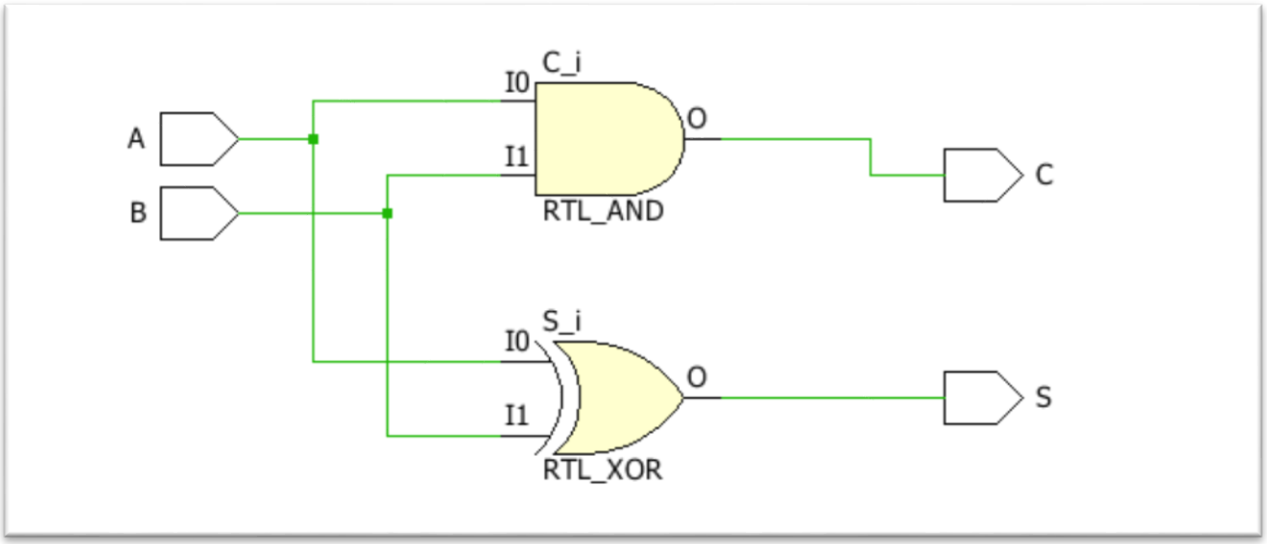


HALF ADDER DATAFLOW MODEL

VHD code

```
entity HALF_ADDER is
    Port ( A : in STD_LOGIC;
           B : in STD_LOGIC;
           S : out STD_LOGIC;
           C : out STD_LOGIC);
end HALF_ADDER;
architecture DATAFLOW of HALF_ADDER is
begin
S<= A XOR B;
C<= A AND B;
end DATAFLOW;
```

SCHEMEATIC DIAGRAM

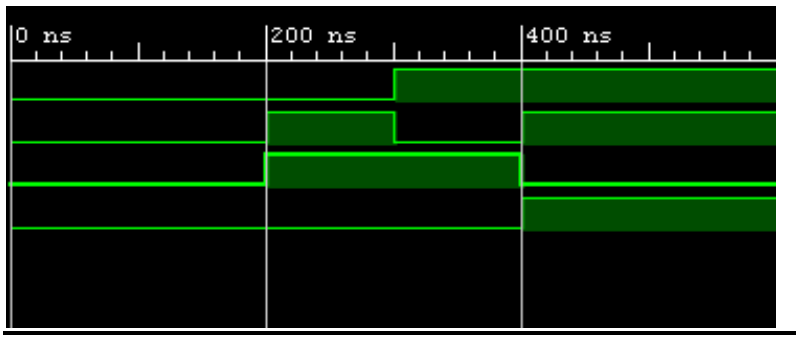


TBW Code

```
entity half_adder_gate_flow is
--  Port ( );
end half_adder_gate_flow;
architecture DATAFLOW of half_adder_gate_flow is
component HALF_ADDER 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 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 DATAFLOW;
```

TBW Waveform



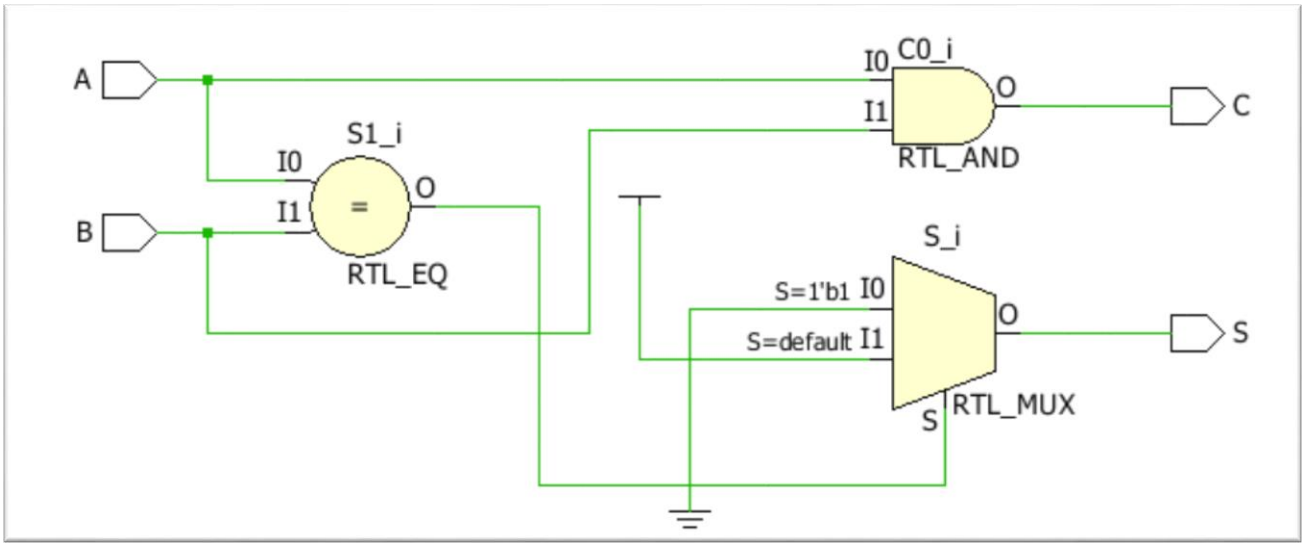
HALF ADDER BEHAVIORAL MODEL

VHD code

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

```
        S : out STD_LOGIC;
        C : out STD_LOGIC);
end HALF_ADDER;
architecture Behavioral of HALF_ADDER 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;
```

SCHEMEATIC DIAGRAM

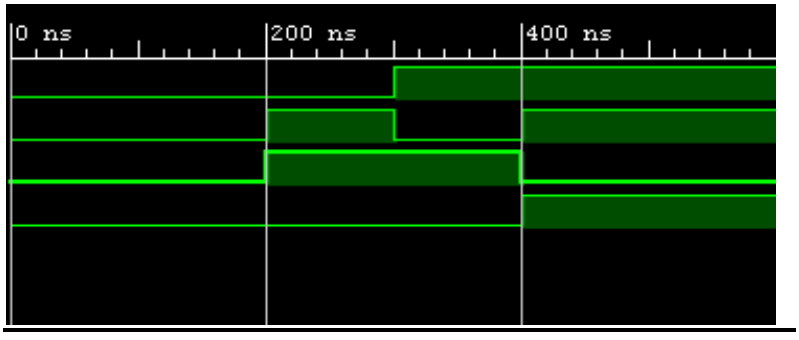


TBW Code

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

```
end half_adder_gate_flow;

architecture Behavioral of half_adder_gate_flow is

component HALF_ADDER 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 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;
```

TBW Waveform

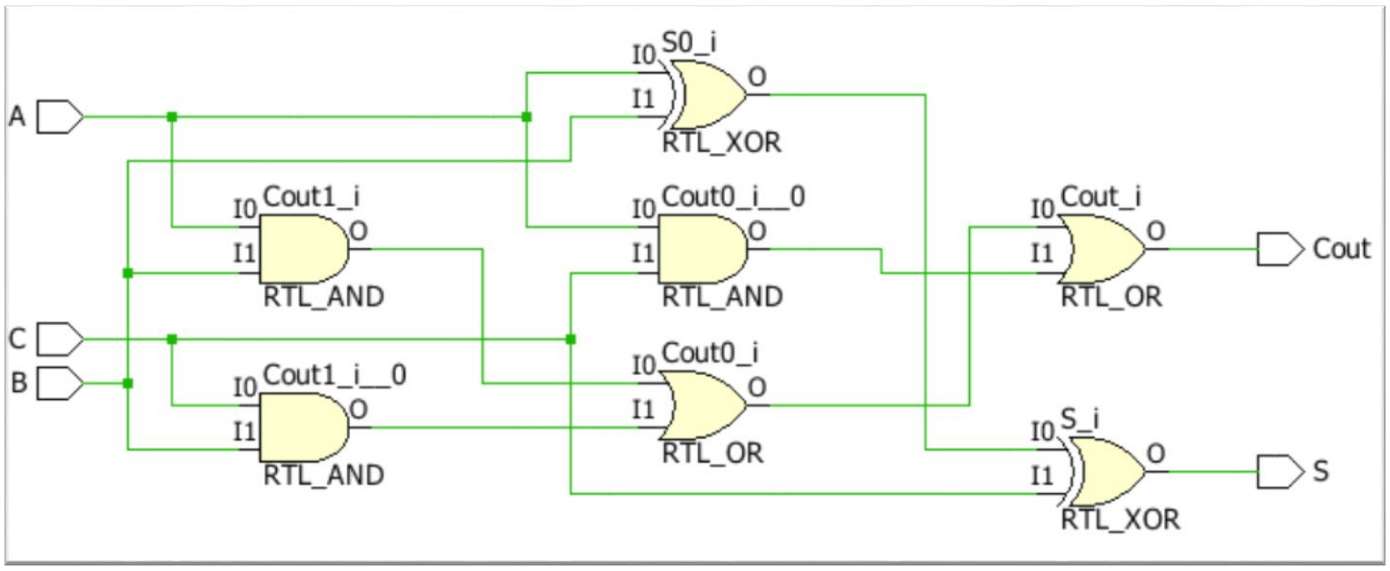


FULL ADDER DATAFLOW MODEL

VHD code

```
entity FULL_ADDER is
    Port ( A : in STD_LOGIC;
           B : in STD_LOGIC;
           C : in STD_LOGIC;
           S : out STD_LOGIC;
           Cout : out STD_LOGIC);
end FULL_ADDER;

architecture DATAFLOW of FULL_ADDER is

begin
S<= (A XOR B) XOR C;
Cout<= (A AND B) OR (C AND B) OR (A AND C);
end DATAFLOW;
```

SCHEMEATIC DIAGRAM



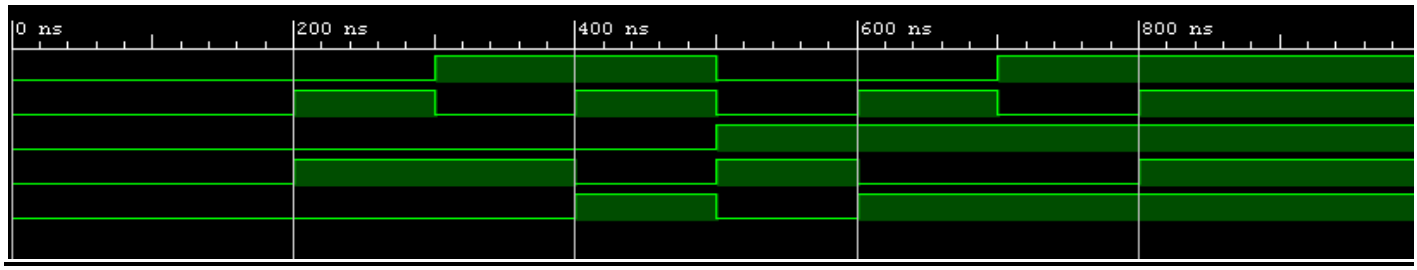
TBW Code

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

```
architecture DATAFLOW of full_adder_gate_flow is

component FULL_ADDER is

    Port( A: in STD_LOGIC;

          B: in STD_LOGIC;

          C: in STD_LOGIC;

          S: out STD_LOGIC;

          Cout: out STD_LOGIC);

     end component;

    Signal A1:STD_LOGIC:='0';

    Signal B1:STD_LOGIC:='0';

    Signal C1:STD_LOGIC:='0';

    Signal S1:STD_LOGIC;

    Signal Cout1:STD_LOGIC;

begin

UUT:FULL_ADDER port map(A=>A1, B=>B1, S=>S1, C=>C1, Cout=> Cout1);

Stim_proc:process

begin

wait for 100ns;

C1<='0';A1<='0';B1<='0';

wait for 100ns;

C1<='0';A1<='0';B1<='1';

wait for 100ns;

C1<='0';A1<='1';B1<='0';

wait for 100ns;

C1<='0';A1<='1';B1<='1';

wait for 100ns;

C1<='1';A1<='0';B1<='0';

wait for 100ns;

C1<='1';A1<='0';B1<='1';

wait for 100ns;

C1<='1';A1<='1';B1<='0';

wait for 100ns;

C1<='1';A1<='1';B1<='1';

wait;
```

```
end process;

end DATAFLOW;
```

TBW Waveform



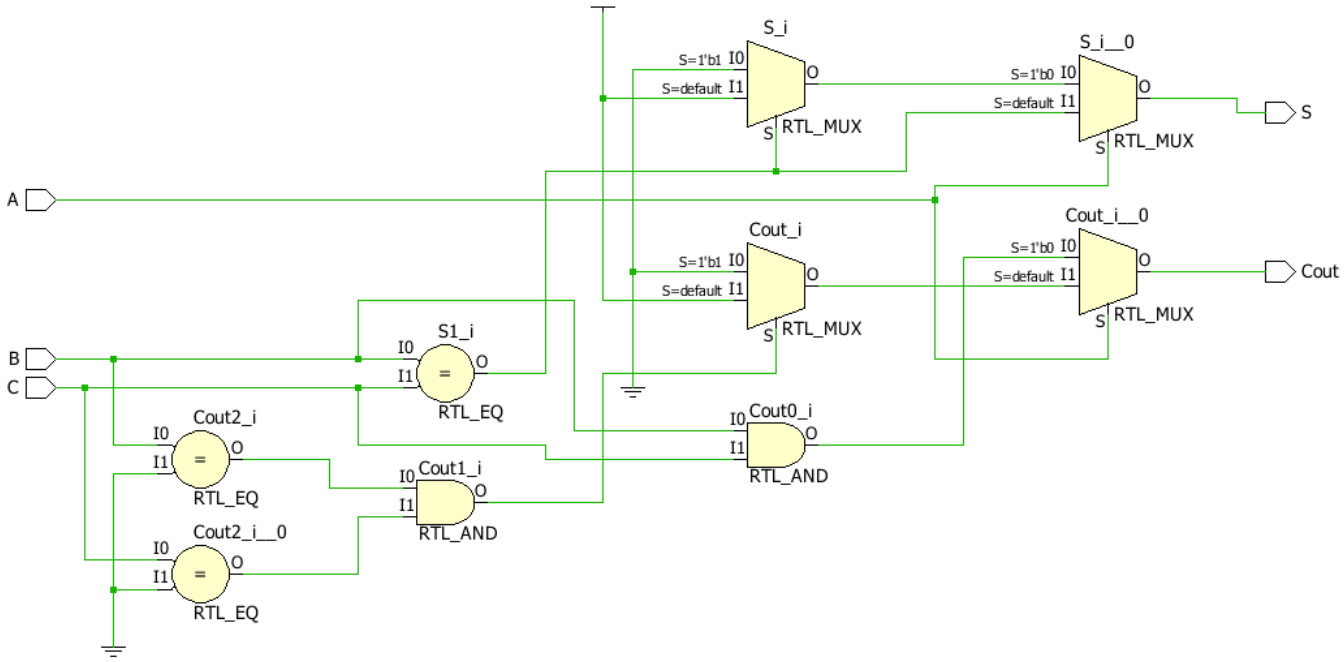
FULL ADDER BEHAVIORAL MODEL

VHD code

```
entity FULL_ADDER is

    Port ( A : in STD_LOGIC;

           B : in STD_LOGIC;

           C : in STD_LOGIC;

           S : out STD_LOGIC;

           Cout : out STD_LOGIC);

end FULL_ADDER;

architecture Behavioral of FULL_ADDER is

begin

process(A,B,C)

begin

if(A='0')then

    if(B=C)then

        S<='0';

    else

        S<='1';

    end if;

else

     if(B=C)then

       S<='1';

     else

       S<='0';

     end if;
```

```
end if;
if(A='0')then
   if(B='1' and C='1')then
      Cout<='1';
   else
      Cout<='0';
   end if;
else
    if(B='0' and C='0')then
      Cout<='0';
  else
      Cout<='1';
  end if;
end if;
end process;
end Behavioral;
```

SCHEMEATIC DIAGRAM



TBW Code

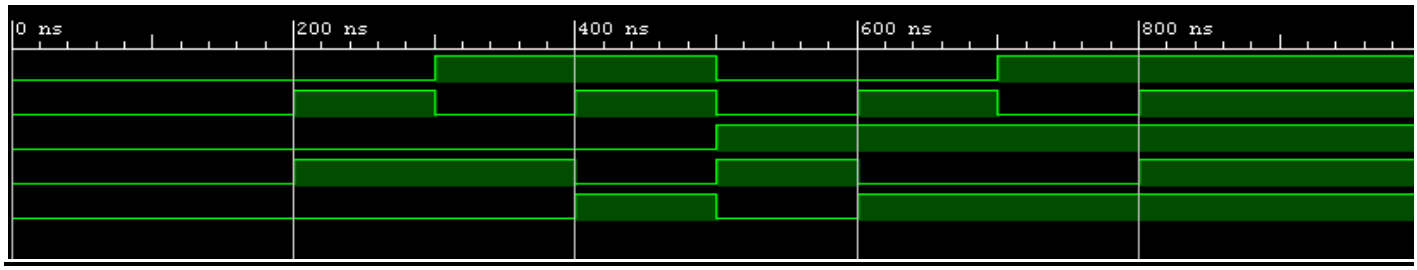
```
entity full_adder_gate_flow is
--  Port ( );
end full_adder_gate_flow;
```

```
architecture Behavioral of full_adder_gate_flow is

component FULL_ADDER is
    Port( A: in STD_LOGIC;
          B: in STD_LOGIC;
          C: in STD_LOGIC;
          S: out STD_LOGIC;
          Cout: out STD_LOGIC);
     end component;
    Signal A1:STD_LOGIC:='0';
    Signal B1:STD_LOGIC:='0';
    Signal C1:STD_LOGIC:='0';
    Signal S1:STD_LOGIC;
    Signal Cout1:STD_LOGIC;
begin
UUT:FULL_ADDER port map(A=>A1, B=>B1, S=>S1, C=>C1, Cout=> Cout1);
Stim_proc:process
begin
wait for 100ns;
C1<='0';A1<='0';B1<='0';
wait for 100ns;
C1<='0';A1<='0';B1<='1';
wait for 100ns;
C1<='0';A1<='1';B1<='0';
wait for 100ns;
C1<='0';A1<='1';B1<='1';
wait for 100ns;
C1<='1';A1<='0';B1<='0';
wait for 100ns;
C1<='1';A1<='0';B1<='1';
wait for 100ns;
C1<='1';A1<='1';B1<='0';
wait for 100ns;
C1<='1';A1<='1';B1<='1';
wait;
```

end process;

end Behavioral;

TBW Waveform

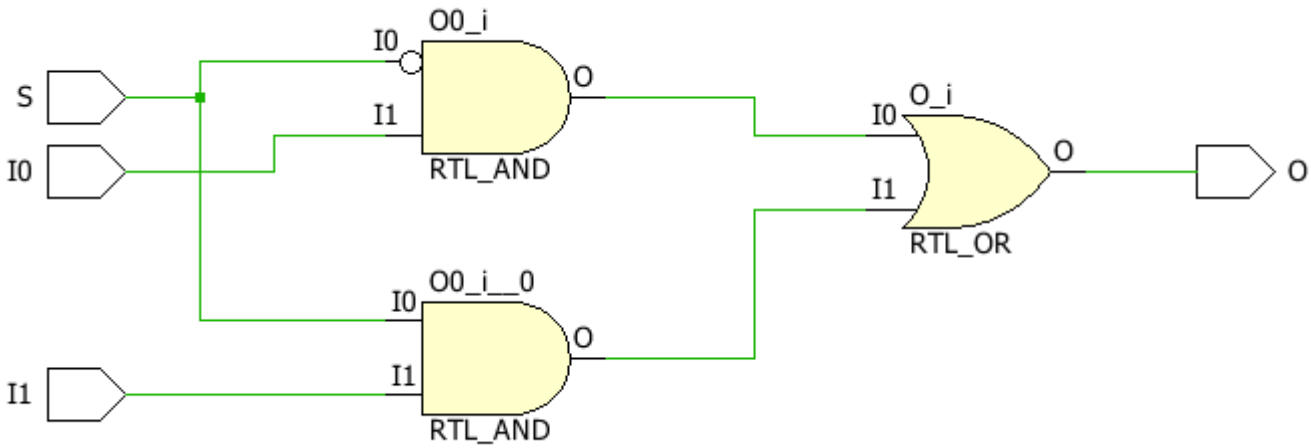


2:1 MUX DATAFLOW MODEL

VHD code

```
entity MUX is
    Port ( I0 : in STD_LOGIC;
           I1 : in STD_LOGIC;
           S : in STD_LOGIC;
           O : out STD_LOGIC);
end MUX;
architecture DATAFLOW of MUX is
begin
O<=((NOT S)AND I0)OR (S AND I1);
end DATAFLOW;
```

SCHEMEATIC DIAGRAM

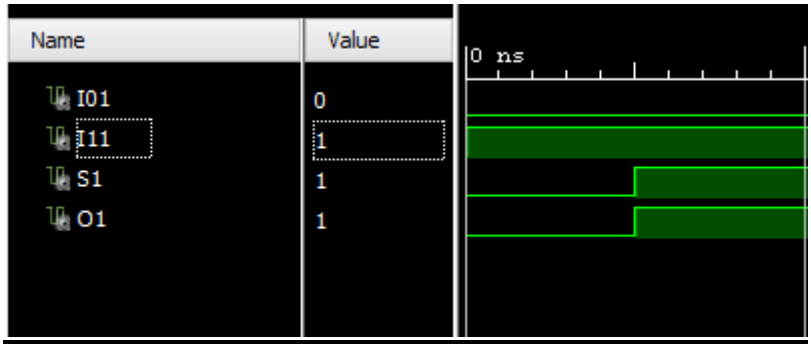


TBW Code

```
entity mux_gate_flow is
--  Port ( );
end mux_gate_flow;
architecture DATAFLOW of mux_gate_flow is
component MUX is
    Port( I0: in STD_LOGIC;
          I1: in STD_LOGIC;
          S: in STD_LOGIC;
          O: out STD_LOGIC);
```

```
    end component;
   Signal I01:STD_LOGIC:='0';
   Signal I11:STD_LOGIC:='1';
   Signal S1:STD_LOGIC:='0';
   Signal O1:STD_LOGIC;
begin
UUT: MUX port map(I0=>I01,I1=>I11,S=>S1,O=>O1);
stim_proc: process
begin
wait for 100ns;
S1<='1';
wait;
end process;
end DATAFLOW;
```

TBW Waveform



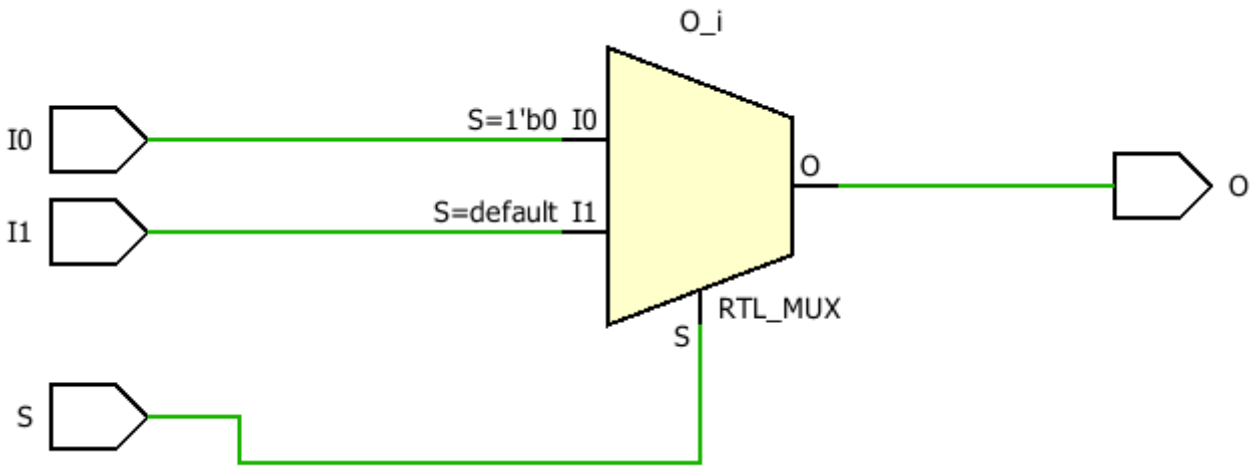
2:1 MUX BEHAVIORAL MODEL

VHD code

```
entity MUX is
   Port ( S : in STD_LOGIC;
          I0 : in STD_LOGIC;
          I1 : in STD_LOGIC;
          O : out STD_LOGIC);
end MUX;
architecture Behavioral of MUX is
begin
process (S,I0,I1)
begin
```

```
if(S='0')then
O<=I0;
else
O<=I1;
end if;
end process;
end Behavioral;
```

SCHEMEATIC DIAGRAM

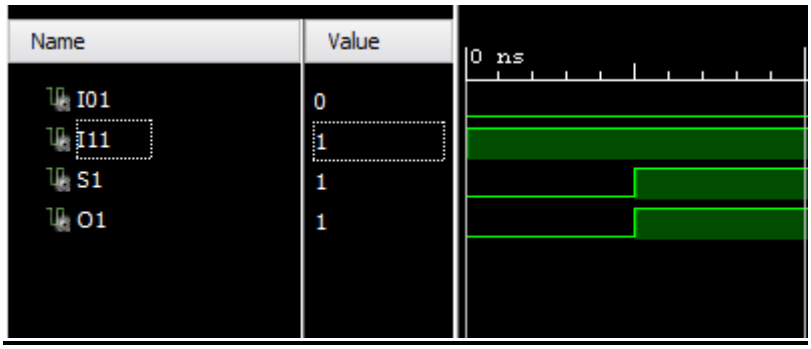


TBW Code

```
entity mux_gate_flow is
--  Port ( );
end mux_gate_flow;
architecture Behavioral of mux_gate_flow is
component MUX is
    Port( I0: in STD_LOGIC;
          I1: in STD_LOGIC;
          S: in STD_LOGIC;
          O: out STD_LOGIC);
     end component;
    Signal I01:STD_LOGIC:='0';
    Signal I11:STD_LOGIC:='1';
    Signal S1:STD_LOGIC:='0';
    Signal O1:STD_LOGIC;
begin
```

```
UUT: MUX port map(I0=>I01,I1=>I11,S=>S1,O=>O1);

stim_proc: process

begin

wait for 100ns;

S1<='1';

wait;

end process;

end Behavioral;
```

TBW Waveform

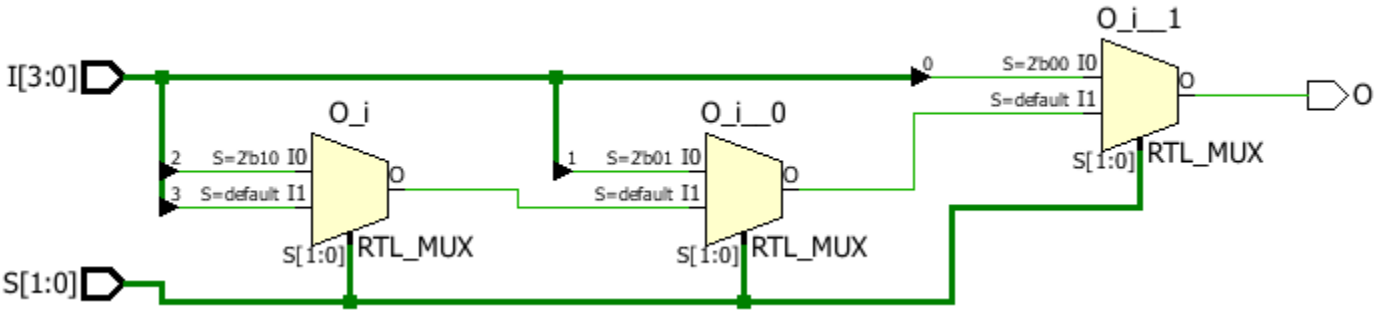


4:1 MUX DATAFLOW MODEL

VHD code

```
entity MUX is

    Port ( I : in STD_LOGIC_VECTOR (3 downto 0);

           S : in STD_LOGIC_VECTOR (1 downto 0);

           O : out STD_LOGIC);

end MUX;

architecture DATAFLOW of MUX is

begin

O<=I(0) when S="00"else

   I(1) when S="01"else

   I(2) when S="10"else

   I(3);

end DATAFLOW;
```

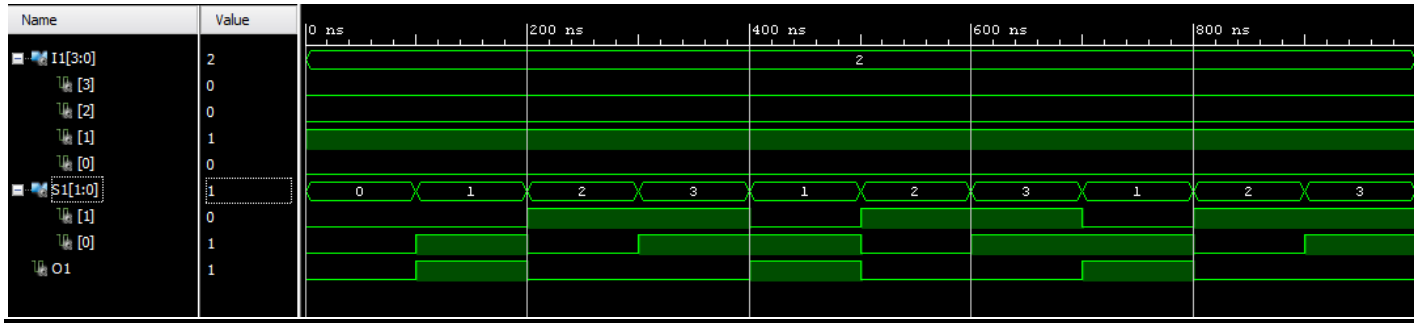
SCHEMEATIC DIAGRAM



TBW Code

```
entity mux_gate_flow is

--  Port ( );

end mux_gate_flow;

architecture DATAFLOW of mux_gate_flow is

component MUX is

    Port ( I : in STD_LOGIC_VECTOR (3 downto 0);

           S : in STD_LOGIC_VECTOR (1 downto 0);

           O : out STD_LOGIC);

end component;

signal I1: STD_LOGIC_VECTOR(3 downto 0):="0010";

signal S1: STD_LOGIC_VECTOR(1 downto 0):="00";

signal O1: STD_LOGIC;

begin

uut: MUX port map(I=>I1,S=>S1,O=>O1);

stim_proc: process

begin

wait for 100ns;

S1<="01";

wait for 100ns;

S1<="10";

wait for 100ns;

S1<="11";

end process;

end DATAFLOW;
```

TBW Waveform

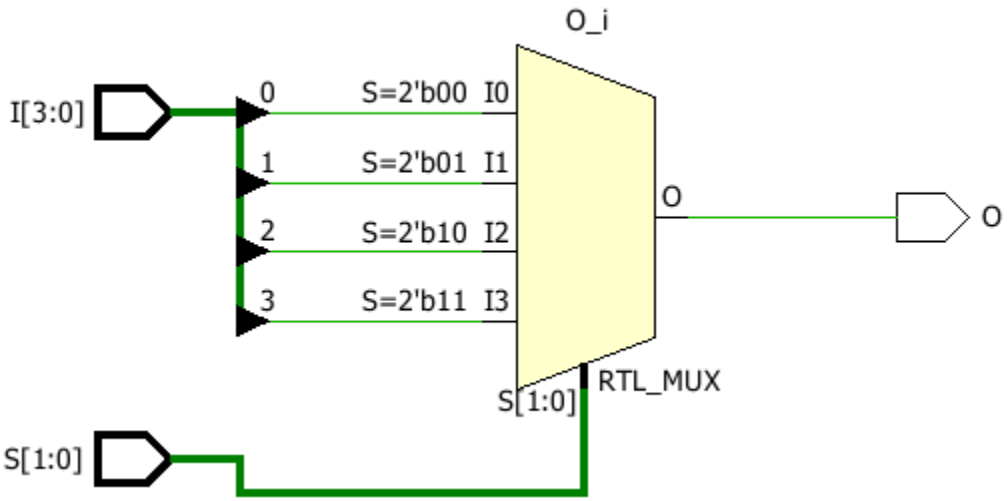


4:1 MUX BEHAVIORAL MODEL

VHD code

```
entity MUX is
   Port ( I : in STD_LOGIC_VECTOR (3 downto 0);
          S : in STD_LOGIC_VECTOR (1 downto 0);
          O : out STD_LOGIC);
end MUX;
architecture Behavioral of MUX is
begin
process(I,S)
begin
case S is
when "00"=>O<=I(0);
when "01"=>O<=I(1);
when "10"=>O<=I(2);
when "11"=>O<=I(3);
when others=>NULL;
end case;
end process;
end Behavioral;
```

SCHEMEATIC DIAGRAM



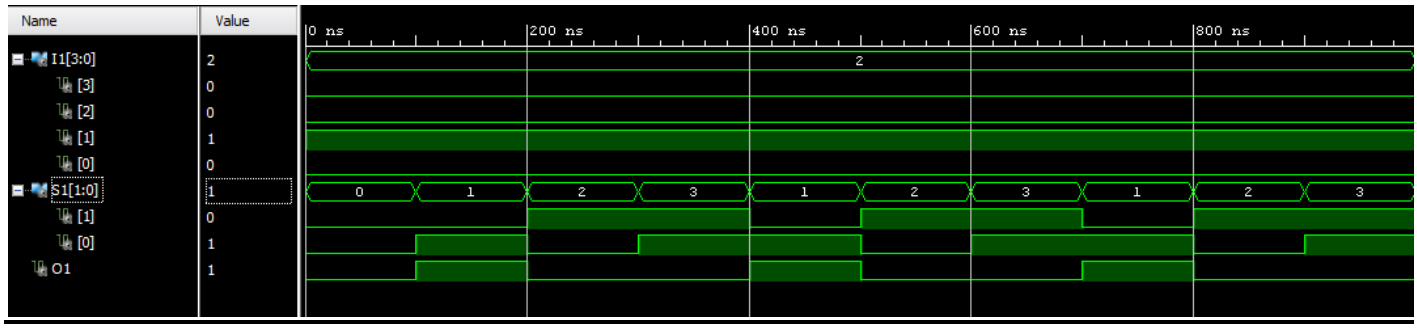
TBW Code

```
entity mux_gate_flow is

--  Port ( );

end mux_gate_flow;


architecture Behavioral of mux_gate_flow is

component MUX is

    Port ( I : in STD_LOGIC_VECTOR (3 downto 0);

           S : in STD_LOGIC_VECTOR (1 downto 0);

           O : out STD_LOGIC);

end component;

signal I1: STD_LOGIC_VECTOR(3 downto 0):="0010";

signal S1: STD_LOGIC_VECTOR(1 downto 0):="00";

signal O1: STD_LOGIC;

begin

uut: MUX port map(I=>I1,S=>S1,O=>O1);

stim_proc: process

begin

wait for 100ns;

S1<="01";

wait for 100ns;

S1<="10";

wait for 100ns;

S1<="11";
```

end process;

end Behavioral;

TBW Waveform



3:8 DECODER BEHAVIORAL MODEL

VHD code

entity DECODE is

Port (I : in STD_LOGIC_VECTOR (2 downto 0);

O : out STD_LOGIC_VECTOR (7 downto 0));

end DECODE;

architecture Behavioral of DECODE is

begin

process(I)

begin

O<="00000000";

case I is

when "000"=> O(0) <='1';

when "001"=> O(1) <='1';

when "010"=> O(2) <='1';

when "011"=> O(3) <='1';

when "100"=> O(4) <='1';

when "101"=>O(5)<='1';

when "110"=>O(6)<='1';

when "111"=>O(7)<='1';

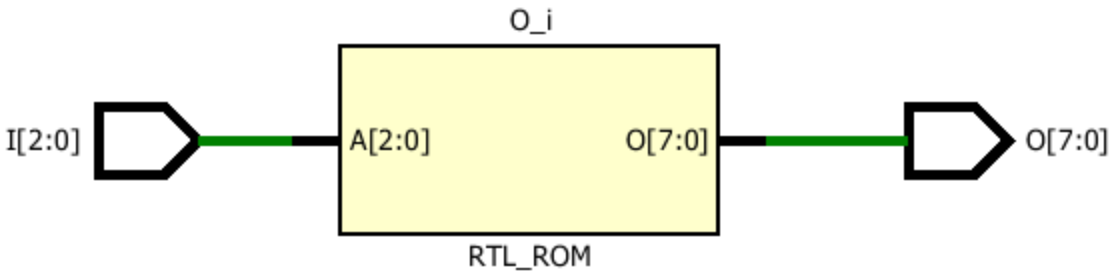
when others=>NULL;

end case;

end process;

end Behavioral;

SCHEMEATIC DIAGRAM



TBW Code

```
entity decode_gate_flow is

--  Port ( );

end decode_gate_flow;

architecture Behavioral of decode_gate_flow is

component DECODE is

    Port ( I : in STD_LOGIC_VECTOR (2 downto 0);

           O : out STD_LOGIC_VECTOR (7 downto 0));

end component;

signal I1: STD_LOGIC_VECTOR(2 downto 0):="000";

signal O1: STD_LOGIC_VECTOR(7 downto 0):="00000000";

begin

uut: DECODE port map(I=>I1,O=>O1);

stim_proc: process

begin

wait for 100ns;

I1<="001";

wait for 100ns;

I1<="010";

wait for 100ns;

I1<="011";

wait for 100ns;

I1<="100";

wait for 100ns;
```

I1<="101";

wait for 100ns;

I1<="110";

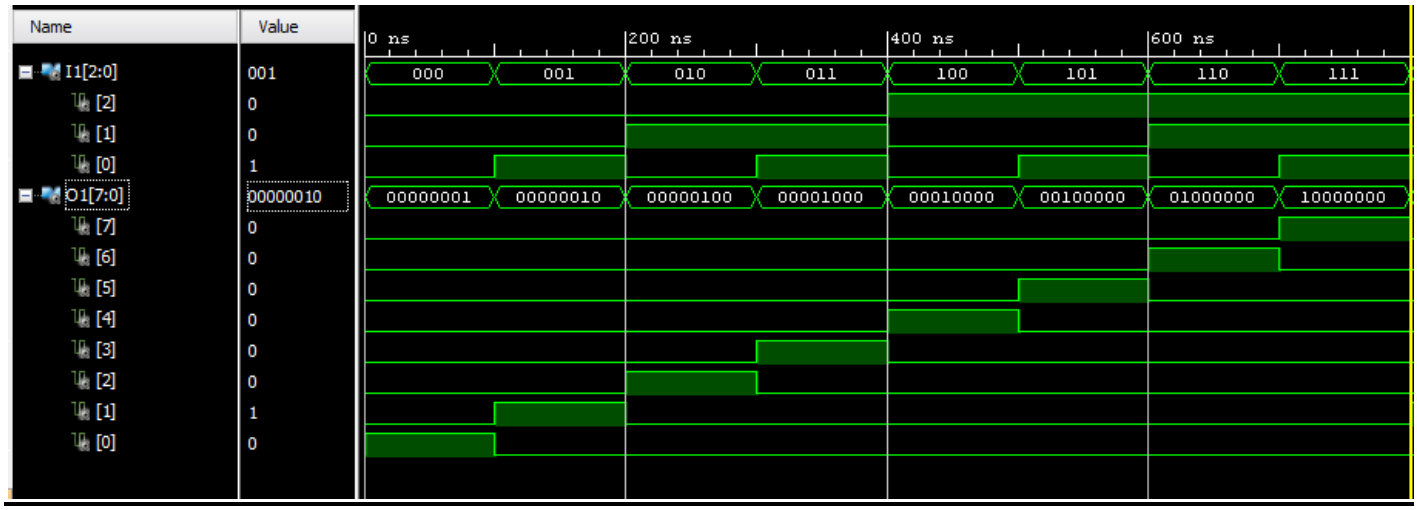
wait for 100ns;

I1<="111";

end process;

end Behavioral;

TBW Waveform



3:8 DECODER DATAFLOW MODEL

VHD code

entity DECODE is

Port (I : in STD_LOGIC_VECTOR (2 downto 0);

O : out STD_LOGIC_VECTOR (7 downto 0));

end DECODE;

architecture DATAFLOW of DECODE is

begin

O(0)<= '1' WHEN I="000" ELSE '0';

O(1)<= '1' WHEN I="001" ELSE '0';

O(2)<= '1' WHEN I="010" ELSE '0';

O(3)<= '1' WHEN I="011" ELSE '0';

O(4)<= '1' WHEN I="100" ELSE '0';

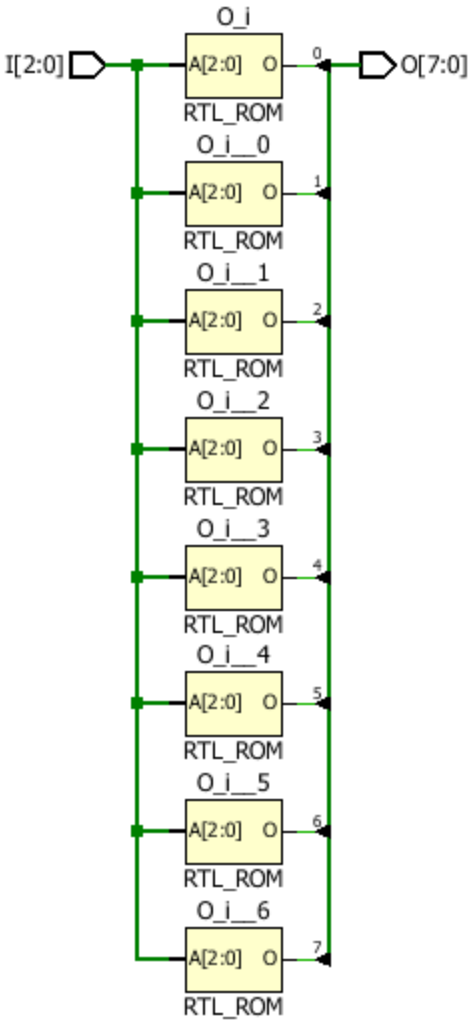
O(5)<= '1' WHEN I="101" ELSE '0';

O(6)<= '1' WHEN I="110" ELSE '0';

O(7)<= '1' WHEN I="111" ELSE '0';

end DATAFLOW;

SCHEMEATIC DIAGRAM

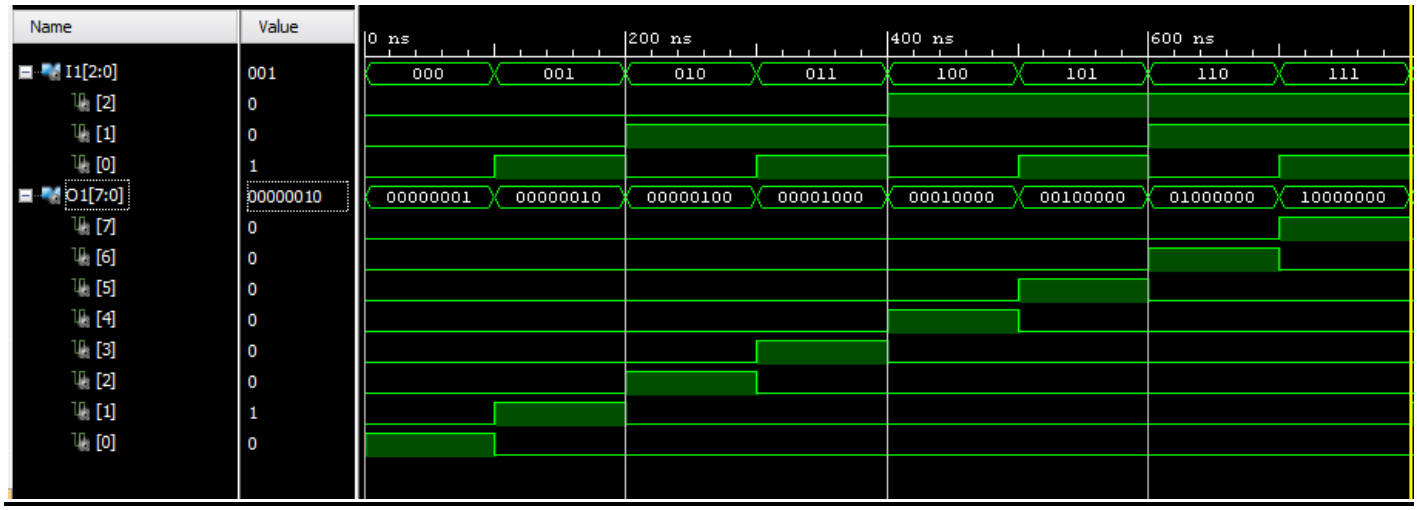


TBW Code

```
entity decode_gate_flow is

--  Port ( );

end decode_gate_flow;

architecture DATAFLOW of decode_gate_flow is

component DECODE is

    Port ( I : in STD_LOGIC_VECTOR (2 downto 0);

           O : out STD_LOGIC_VECTOR (7 downto 0));

end component;

signal I1: STD_LOGIC_VECTOR(2 downto 0):="000";

signal O1: STD_LOGIC_VECTOR(7 downto 0):="00000000";

begin

uut: DECODE port map(I=>I1,O=>O1);
```

```
stim_proc: process

begin

wait for 100ns;

I1<="001";

wait for 100ns;

I1<="010";

wait for 100ns;

I1<="011";

wait for 100ns;

I1<="100";

wait for 100ns;

I1<="101";

wait for 100ns;

I1<="110";

wait for 100ns;

I1<="111";

end process;

end DATAFLOW;
```

TBW Waveform

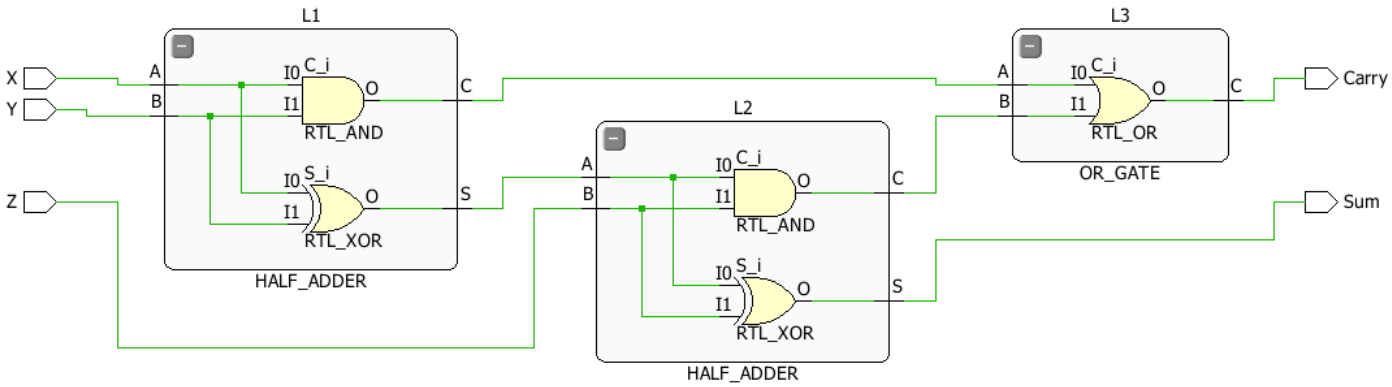


FULL ADDER (STRUCTURAL MODEL)

VHD code

```
entity FULL_ADDER_STR is
    Port ( X : in STD_LOGIC;
           Y : in STD_LOGIC;
           Z : in STD_LOGIC;
           Sum : out STD_LOGIC;
           Carry : out STD_LOGIC);
end FULL_ADDER_STR;
architecture STRUCTURAL of FULL_ADDER_STR is
component HALF_ADDER is
    Port ( A : in STD_LOGIC;
           B : in STD_LOGIC;
           S : out STD_LOGIC;
           C : out STD_LOGIC);
end component;
component OR_GATE is
    Port ( A : in STD_LOGIC;
           B : in STD_LOGIC;
           C : out STD_LOGIC);
end component;
signal S1:STD_LOGIC;
signal C1:STD_LOGIC;
signal C2:STD_LOGIC;
begin
L1: HALF_ADDER port map(X,Y,S1,C1);
L2: HALF_ADDER port map(S1,Z,Sum,C2);
L3: OR_GATE port map(C1,C2,Carry);
end STRUCTURAL;
```

SCHEMEATIC DIAGRAM

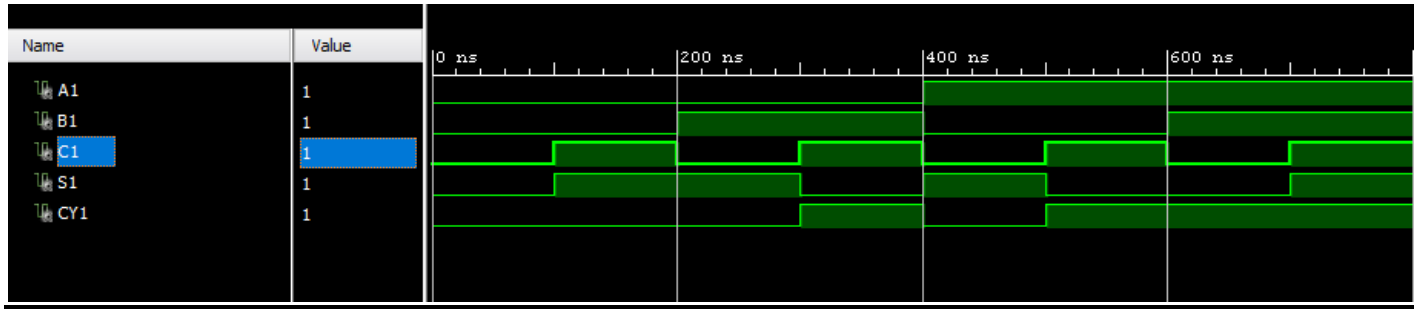


TBW Code

```
entity FULL_ADDER is

--  Port ( );

end FULL_ADDER;

architecture STRUCTURAL of FULL_ADDER is

component FULL_ADDER_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 A1:STD_LOGIC:='0';

signal B1:STD_LOGIC:='0';

signal C1:STD_LOGIC:='0';

signal S1:STD_LOGIC;

signal CY1:STD_LOGIC;

begin

uut: FULL_ADDER_STR Port Map(X=>A1,Y=>B1,Z=>C1,Sum=>S1,Carry=>CY1);

stim_proc:process

begin

A1<='0';

B1<='0';

C1<='0';

wait for 100ns;
```

```
A1<='0';
B1<='0';
C1<='1';
wait for 100ns;
A1<='0';
B1<='1';
C1<='0';
wait for 100ns;
A1<='0';
B1<='1';
C1<='1';
wait for 100ns;
A1<='1';
B1<='0';
C1<='0';
wait for 100ns;
A1<='1';
B1<='0';
C1<='1';
wait for 100ns;
A1<='1';
B1<='1';
C1<='0';
wait for 100ns;
A1<='1';
B1<='1';
C1<='1';
wait;
end process;
end STRUCTURAL;
```

TBW Waveform

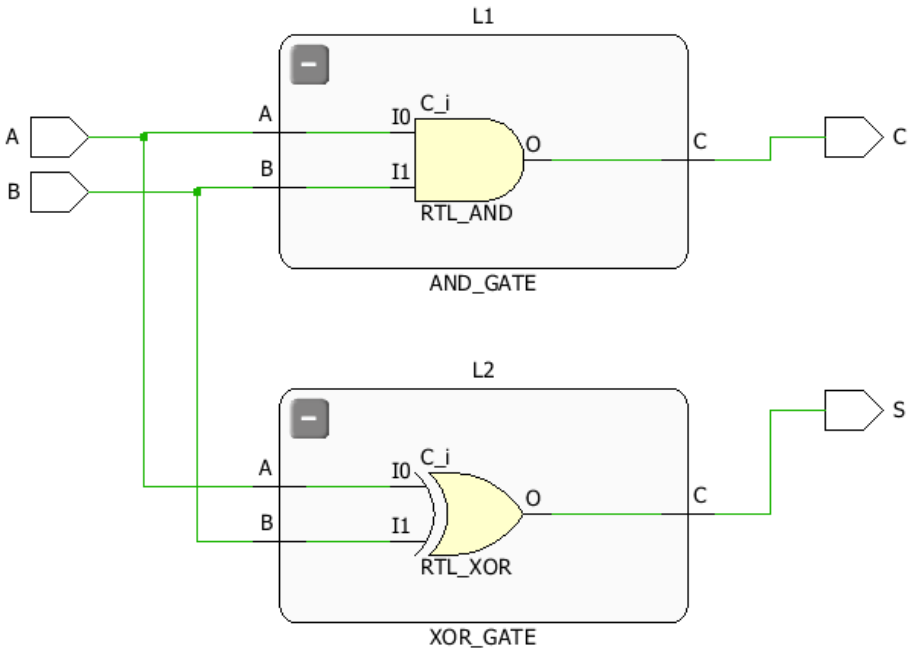


HALF ADDER (STRUCTURAL MODEL)

VHD code

```
entity HALF_ADDER_STR is
    Port ( A : in STD_LOGIC;
           B : in STD_LOGIC;
           S : out STD_LOGIC;
           C : out STD_LOGIC);
end HALF_ADDER_STR;
architecture STRUCTURAL of HALF_ADDER_STR is
component XOR_GATE is
    Port ( A : in STD_LOGIC;
           B : in STD_LOGIC;
           C : out STD_LOGIC);
end component;
component AND_GATE is
    Port ( A : in STD_LOGIC;
           B : in STD_LOGIC;
           C : out STD_LOGIC);
end component;
begin
L1: AND_GATE port map(A,B,C);
L2: XOR_GATE port map(A,B,S);
end STRUCTURAL;
```

SCHEMEATIC DIAGRAM



TBW Code

```
entity half_adder_str is

--  Port ( );

end half_adder_str;

architecture STRUCTURAL of half_adder_str is

component HALF_ADDER_STR 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_STR 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 STRUCTURAL;
```

TBW Waveform