**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;

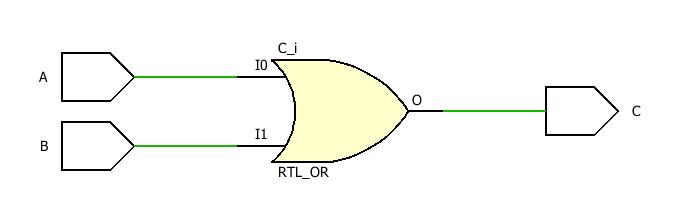
architecture DATAFLOW of OR\_GATE is

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;

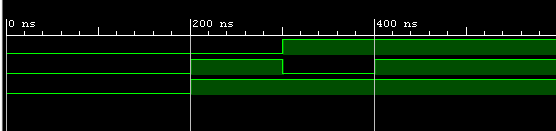
A1<='1';B1<='1';

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;

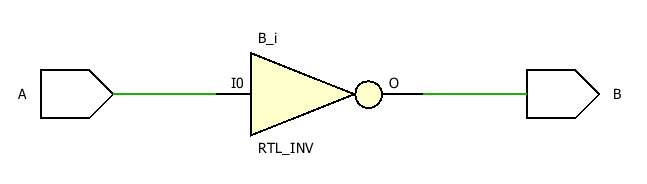
architecture DATAFLOW of NOT\_GATE is

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;

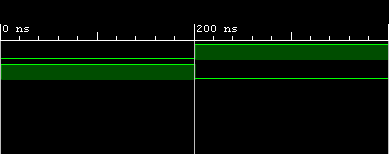
A1<='1';

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;

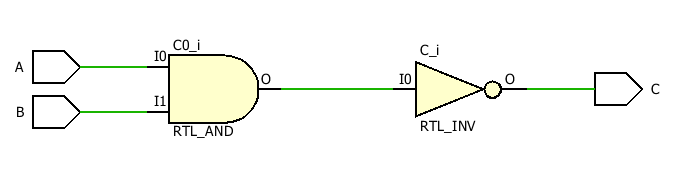
architecture DATAFLOW of NAND\_GATE is

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;

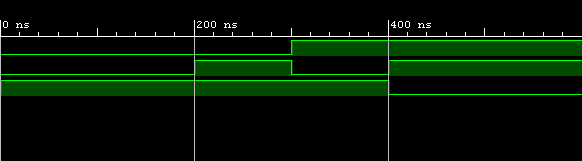
A1<='1';B1<='1';

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;

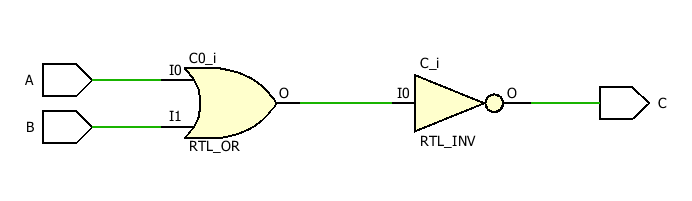
architecture DATAFLOW of NOR\_GATE is

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;

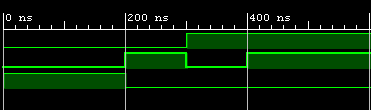
A1<='1';B1<='1';

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;

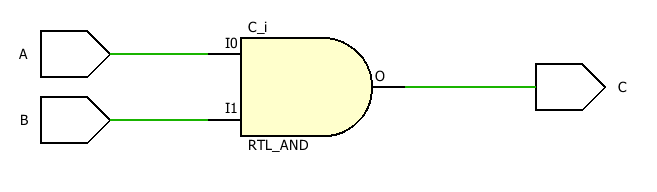
architecture DATAFLOW of AND\_GATE is

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;

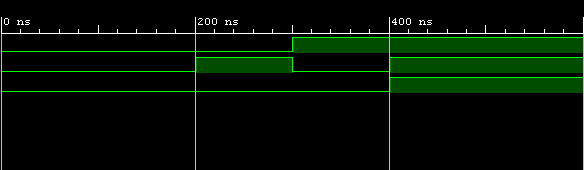
A1<='1';B1<='1';

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;

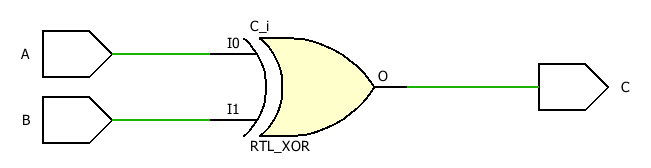
architecture DATAFLOW of XOR\_GATE is

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;

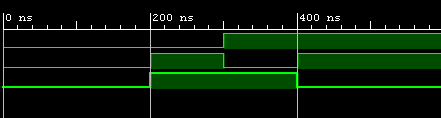
A1<='1';B1<='1';

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;

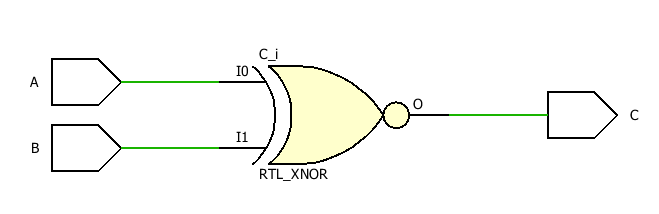
architecture DATAFLOW of XNOR\_GATE is

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;

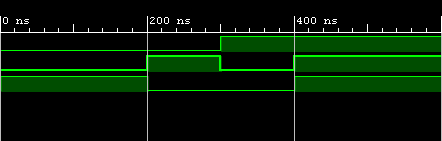
A1<='1';B1<='1';

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;

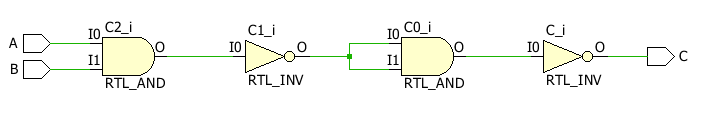
architecture Behavioral of NAND\_AND is

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;

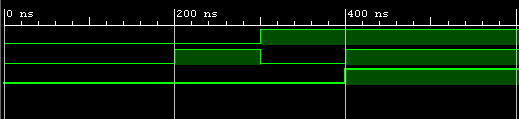
A1<='1';B1<='1';

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;

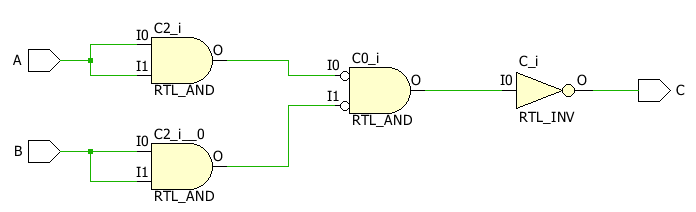
architecture Behavioral of NAND\_OR is

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;

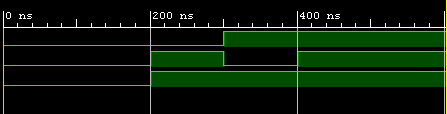
A1<='1';B1<='1';

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;

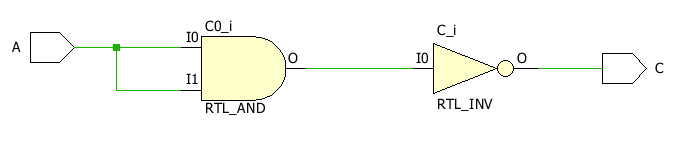
architecture DATAFLOW of NAND\_NOT is

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;

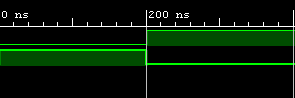
A1<='1';

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;

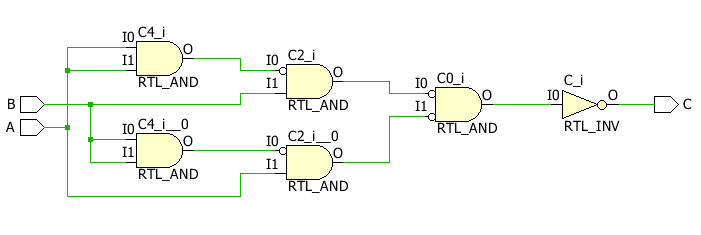
architecture DATAFLOW of NAND\_XOR is

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;

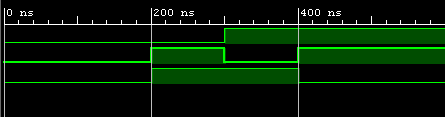
A1<='1';B1<='1';

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;

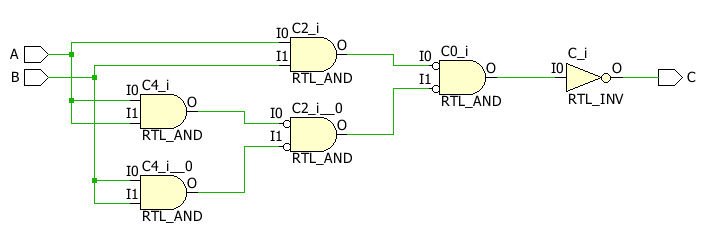
architecture DATAFLOW of NAND\_XNOR is

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;

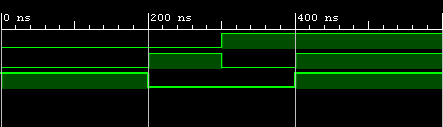
A1<='1';B1<='1';

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;

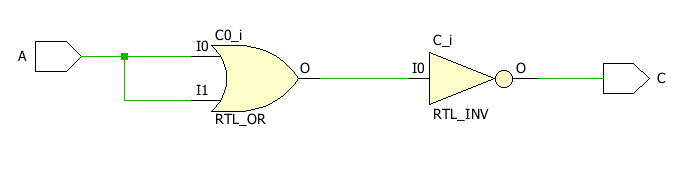
architecture DATAFLOW of NOR\_NOT is

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;

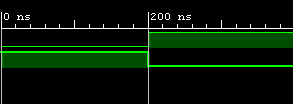
A1<='1';

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;

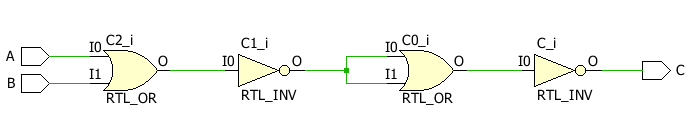
architecture DATAFLOW of NOR\_OR is

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;

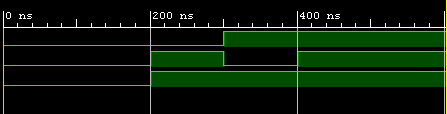
A1<='1';B1<='1';

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;

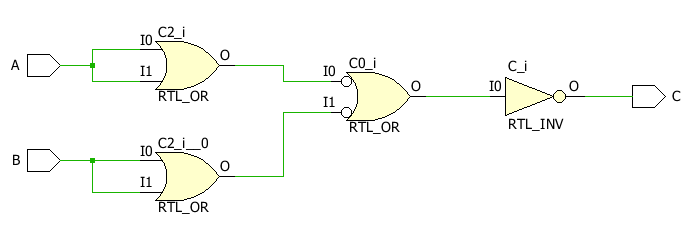
architecture DATAFLOW of NOR\_AND is

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;

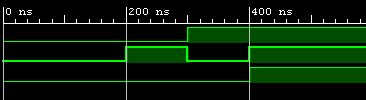
A1<='1';B1<='1';

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;

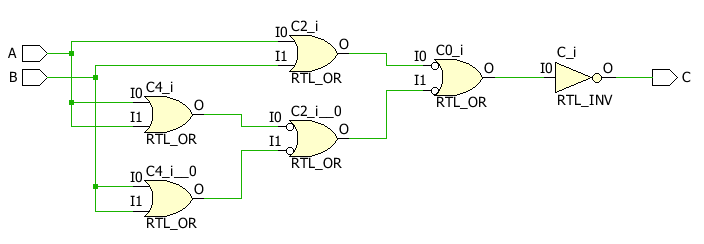
architecture DATAFLOW of NOR\_XOR is

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;

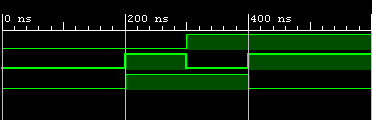
A1<='1';B1<='1';

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;

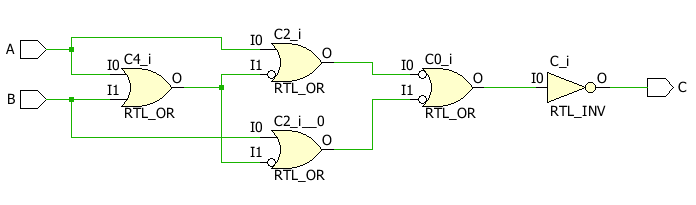
architecture DATAFLOW of NOR\_XNOR is

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;

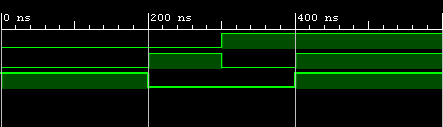
A1<='1';B1<='1';

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

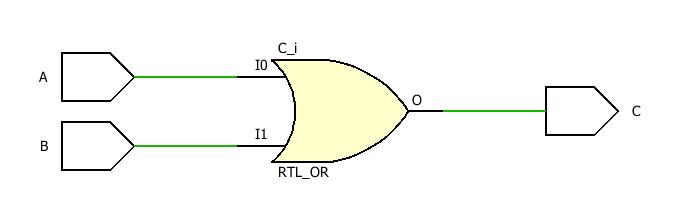
C<='0';

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;

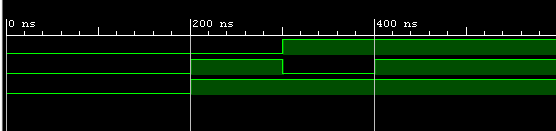
A1<='1';B1<='1';

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

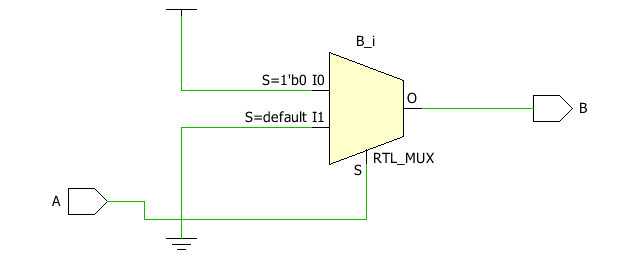
B<='0';

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;

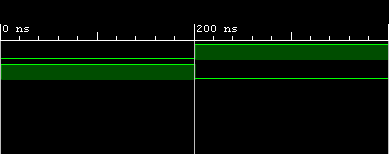
A1<='1';

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

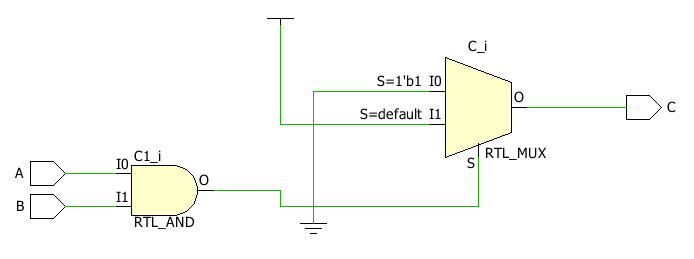
C<='1';

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;

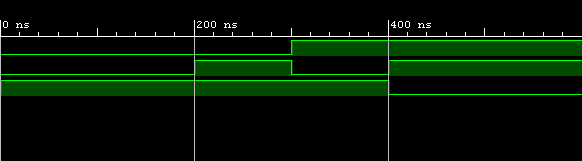
A1<='1';B1<='1';

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

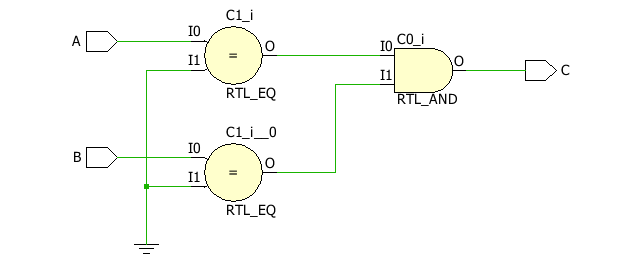
C<='0';

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;

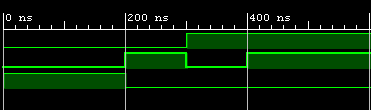
A1<='1';B1<='1';

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

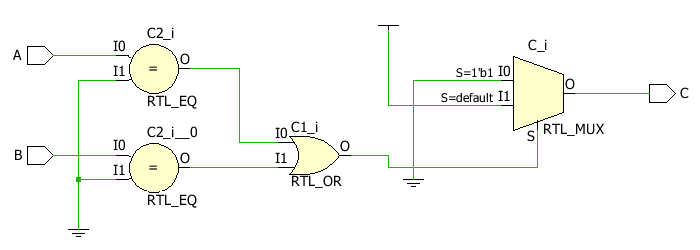
C<='1';

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;

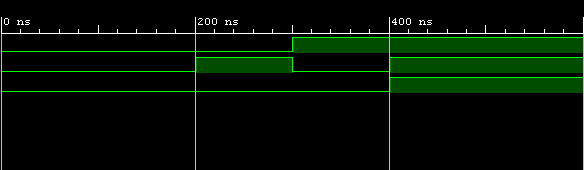
A1<='1';B1<='1';

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

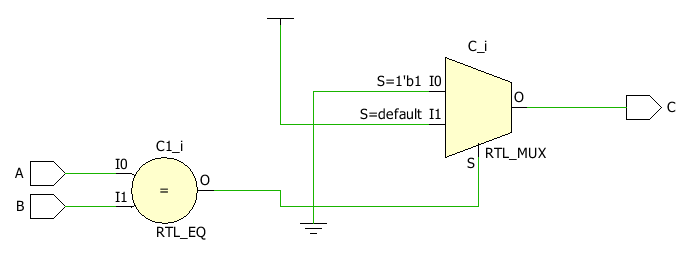
C<='1';

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;

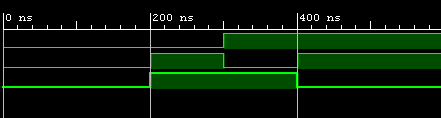
A1<='1';B1<='1';

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

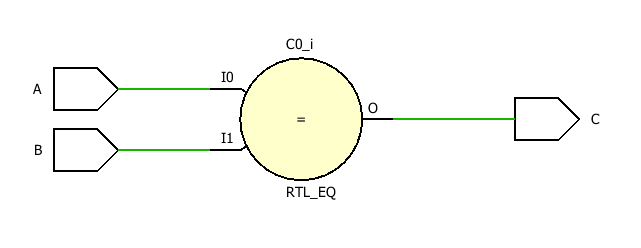
C<='0';

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;

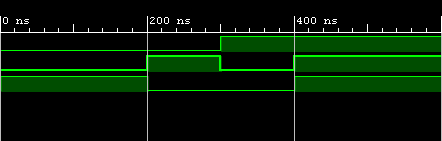
A1<='1';B1<='1';

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

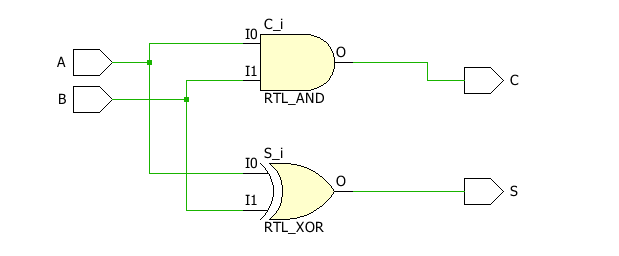
begin

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;

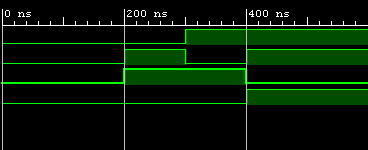
A1<='1';B1<='1';

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

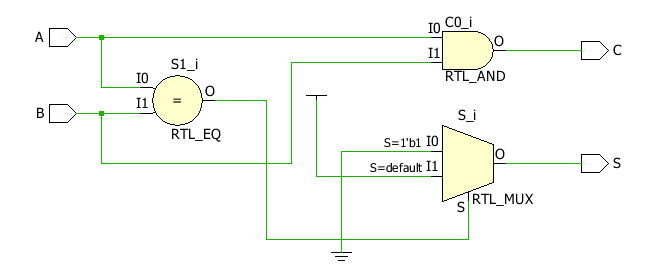
C<='0';

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;

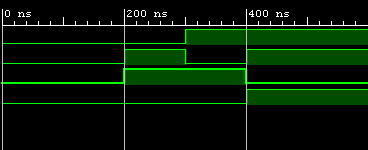
A1<='1';B1<='1';

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

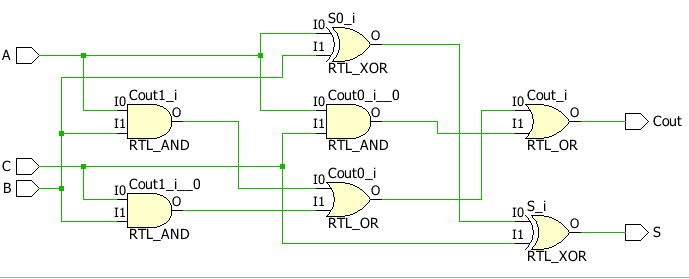
begin

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;

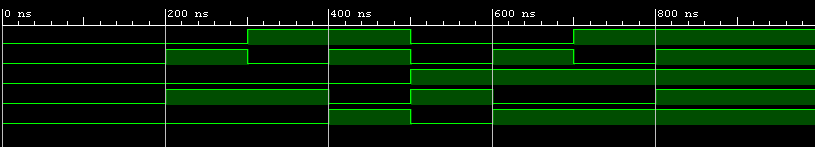
C1<='1';A1<='1';B1<='1';

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';

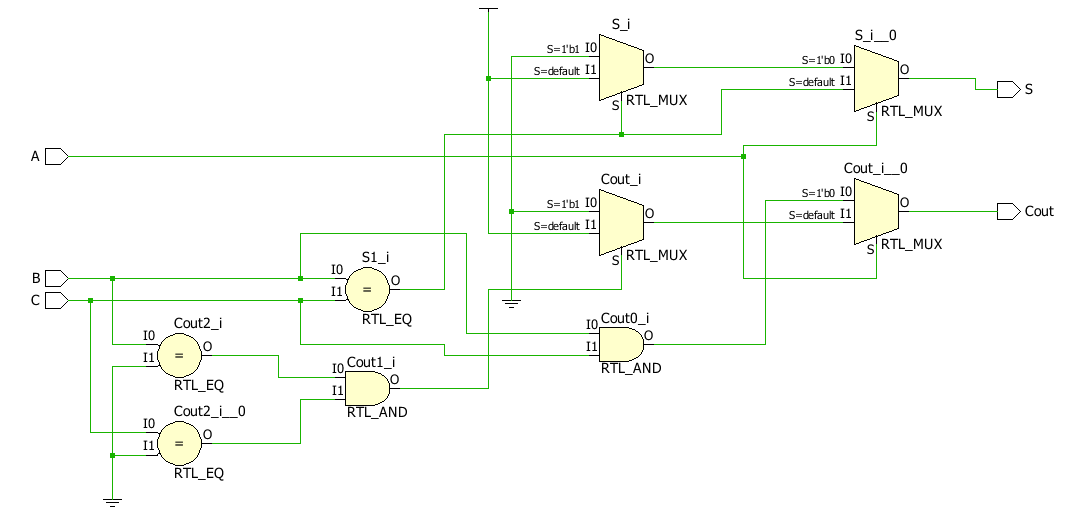
end if;

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;

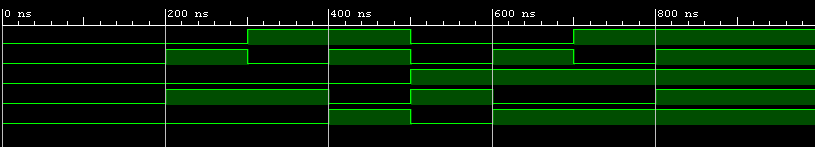
C1<='1';A1<='1';B1<='1';

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;

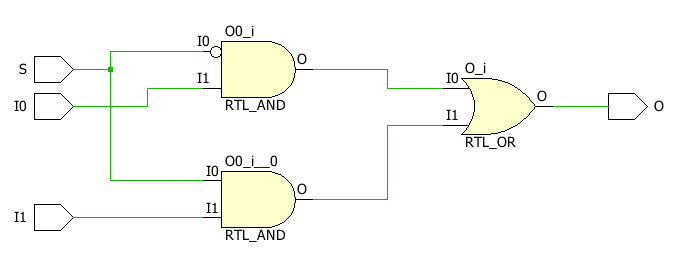
architecture DATAFLOW of MUX is

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;

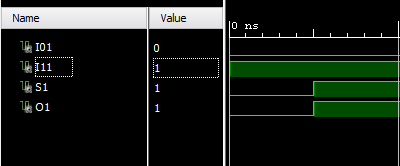
S1<='1';

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

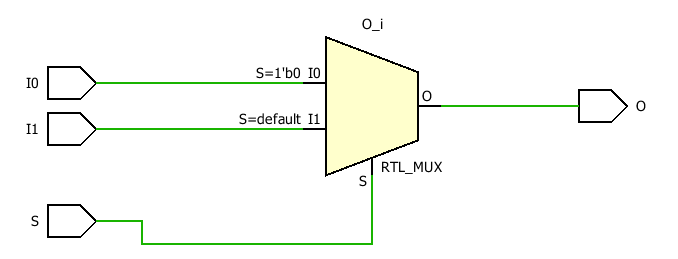
O<=I1;

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;

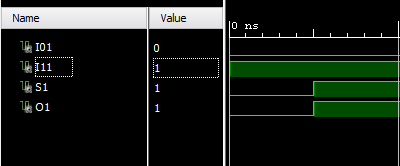
S1<='1';

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

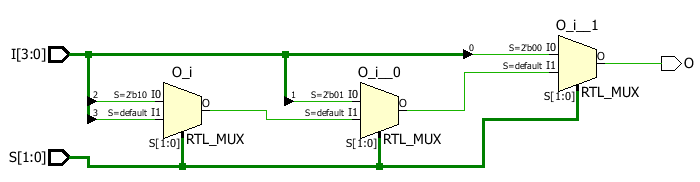
I(1) when S="01"else

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";

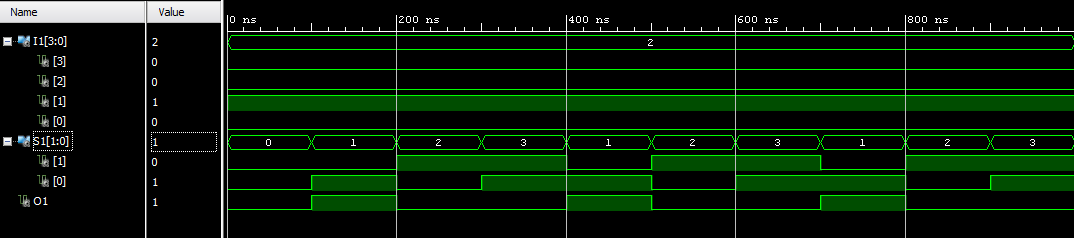
wait for 100ns;

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);

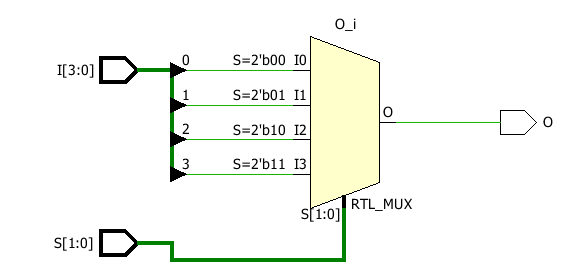
when others=>NULL;

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";

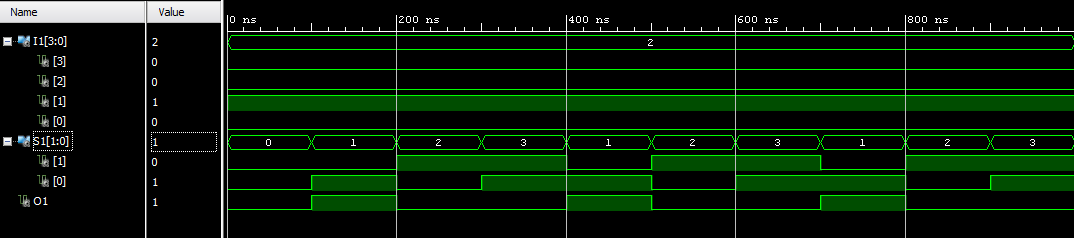
wait for 100ns;

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';

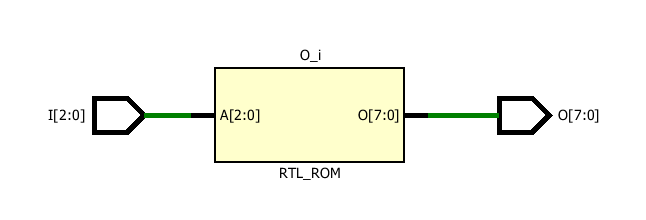
when others=>NULL;

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";

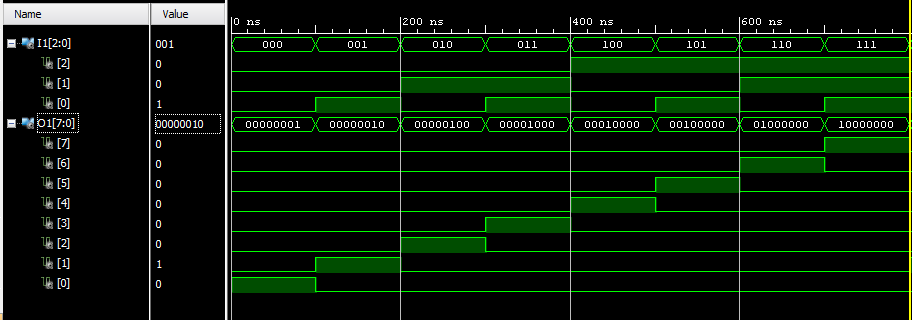
wait for 100ns;

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';

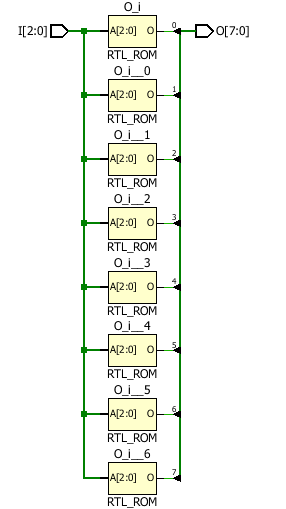
O(5)<= '1' WHEN I="101" ELSE '0';

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";

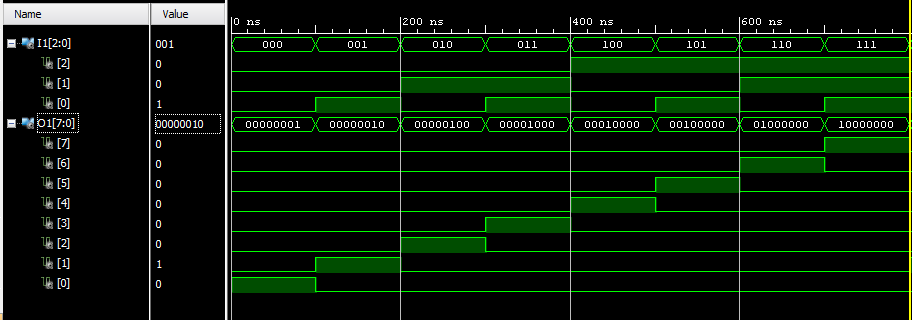
wait for 100ns;

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

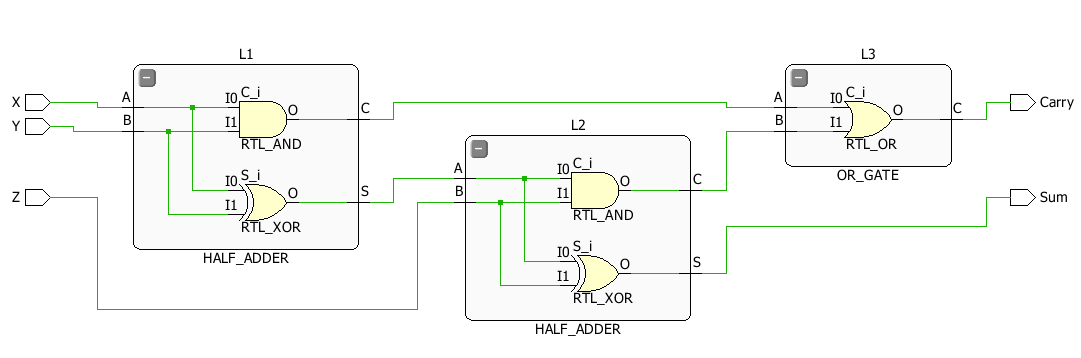
L1: HALF\_ADDER port map(X,Y,S1,C1);

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';

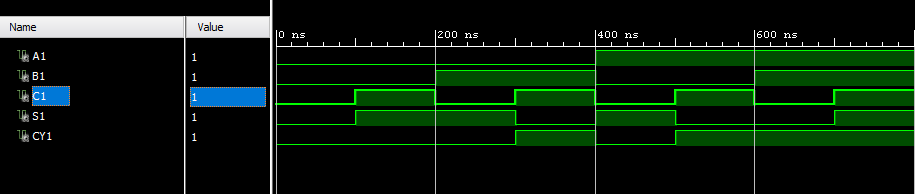
C1<='1';

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;

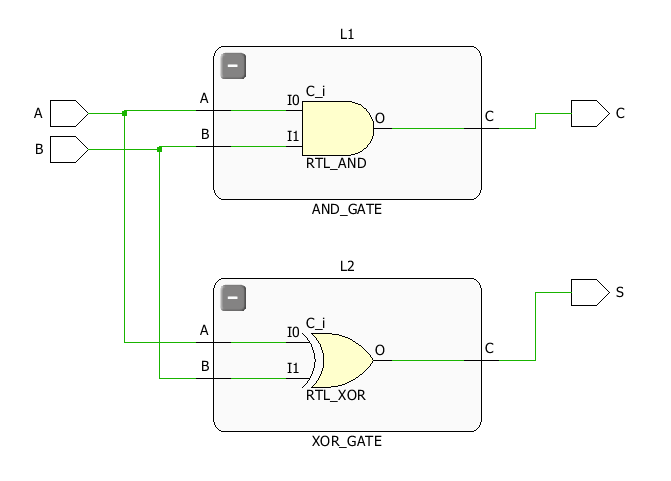
begin

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;

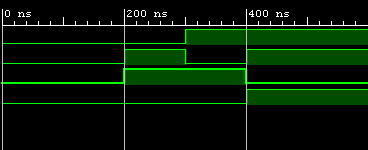
A1<='1';B1<='1';

wait;

end process;

end STRUCTURAL;

**TBW Waveform**

****