FULL ADDER

AIM:

To design, implement and analyze all the three models for full adder.

Design:

First, VHDL code for half adder was written and block was generated. Half adder block as component and basic gates, code for full adder is written. The truth tables are as follows:

HALF ADDER:

A	B	SUM	CARRY
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

FULL ADDER:

A	B	CIN	SUM	COUT
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

SUM	00	01	11	10
0		1		1
1	1		1	

CARRY	00	01	11	10
0			1	
1		1	1	1

SUM = A XOR B XOR C; CARRY = AB + AC + BC;

--Structural model for Half Adder

```
library IEEE;
use IEEE.STD_LOGIC_1164.all;

entity HA is
        port(A,B:in STD_LOGIC; Sum, Carry:out STD_LOGIC);
end HA;

architecture struct of HA is
        component myXOR
                port(in1,in2:in STD_LOGIC; out1:out STD_LOGIC);
        end component;
        begin
                X1: myXOR port map(A,B,Sum);
                Carry<=A and B;
end struct;
```

--Structural model for Full Adder

```
library IEEE;
use IEEE.STD_LOGIC_1164.all;

entity FA is
        port(x,y,cin:in std_logic; s, cout:out std_logic);
end FA;

architecture struct of FA is
        signal s1,c1,c2 :std_logic;
        component HA
                port(A,B:in STD_LOGIC; sum, Carry:out STD_LOGIC);
        end component;
        begin
                HA1: HA port map(x,y, s1,c1);
                HA2: HA port map(s1,cin, s,c2);
                cout<=c1 or c2;
end struct;
```

RTL VIEW (Structural):

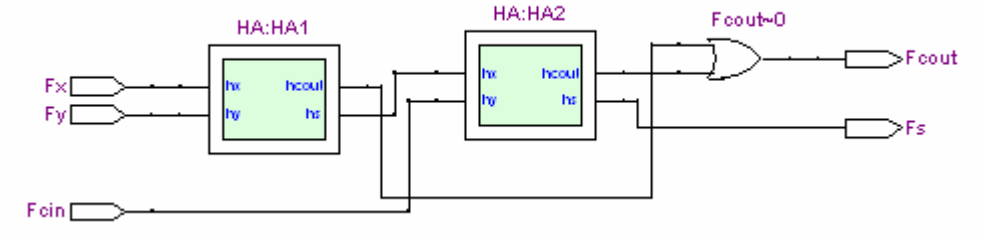


Fig. Full Adder

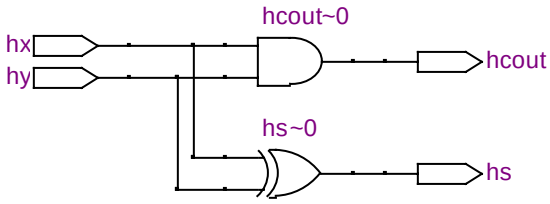
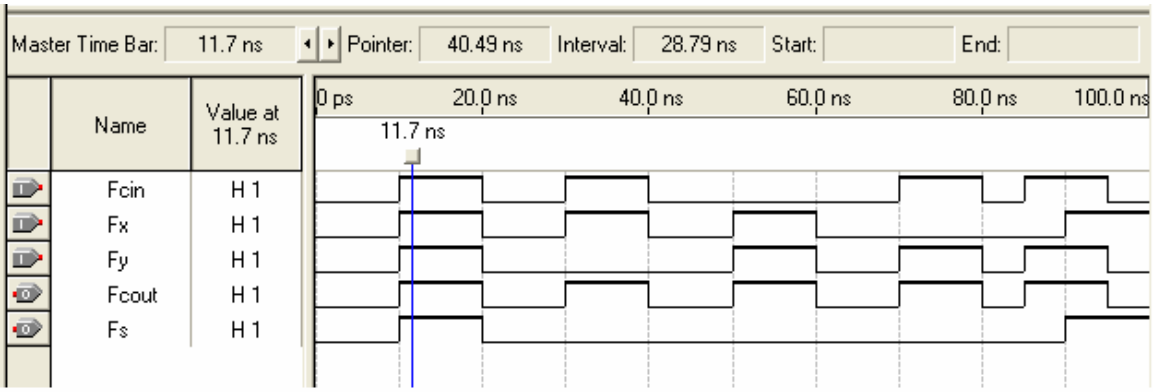


Fig. Half Adder

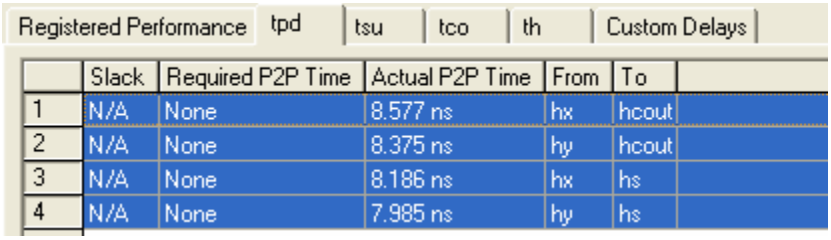
SIMULAION WAVEFORM:



The output shown above is directly taken from the SCF editor of MAX PLUSII BASE LINE.

ANALYSIS:

Timing Analyzer result (Structural):



Registered Performance | tpd | tsu | tco | th | Custom Delays

	Slack	Required P2P Time	Actual P2P Time	From	To
1	N/A	None	8.577 ns	hx	hcout
2	N/A	None	8.375 ns	hy	hcout
3	N/A	None	8.186 ns	hx	hs
4	N/A	None	7.985 ns	hy	hs

FLOW SUMMARY (Structural):

```
Fitter Status : Successful - Thu Oct 19 08:44:16 2006
Quartus II Version : 6.0 Build 202 06/20/2006 SP 1 SJ Web Edition
Revision Name : Adder
Top-level Entity Name : FA
Family : Stratix
Device : EP1S10F484C5
Timing Models : Final
Total logic elements : 2 / 10,570 ( < 1 % )
Total pins : 5 / 336 ( 1 % )
Total virtual pins : 0
Total memory bits : 0 / 920,448 ( 0 % )
DSP block 9-bit elements : 0 / 48 ( 0 % )
Total PLLs : 0 / 6 ( 0 % )
Total DLLs : 0 / 2 ( 0 % )
```

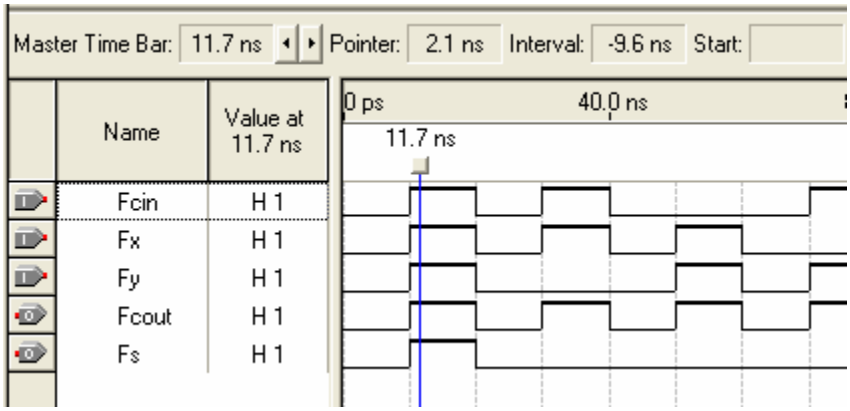
--VHDL code for DATA FLOW model of Full Adder:

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity FA_DF is
        port(Fx, Fy, Fcin : in BIT; Fs, Fcout : out BIT);
end FA_DF;

architecture FA_dataflow of FA_DF is
        begin
                Fs <= Fx XOR Fy XOR Fcin;
                Fcout <= (Fx AND Fy) OR (Fx AND Fcin) OR (Fy AND Fcin);
end FA_dataflow;
```

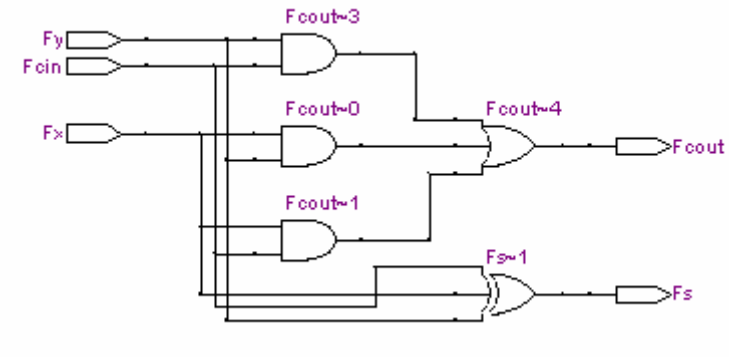
FLOW SUMMARY (Data Flow):

```
Fitter Status : Successful - Thu Oct 19 08:44:16 2006
Quartus II Version : 6.0 Build 202 06/20/2006 SP 1 SJ Web Edition
Revision Name : Adder
Top-level Entity Name : FA
Family : Stratix
Device : EP1S10F484C5
Timing Models : Final
Total logic elements : 2 / 10,570 ( < 1 % )
Total pins : 5 / 336 ( 1 % )
Total virtual pins : 0
Total memory bits : 0 / 920,448 ( 0 % )
DSP block 9-bit elements : 0 / 48 ( 0 % )
Total PLLs : 0 / 6 ( 0 % )
Total DLLs : 0 / 2 ( 0 % )
```

SIMULATION WAVEFORM (DATA FLOW):



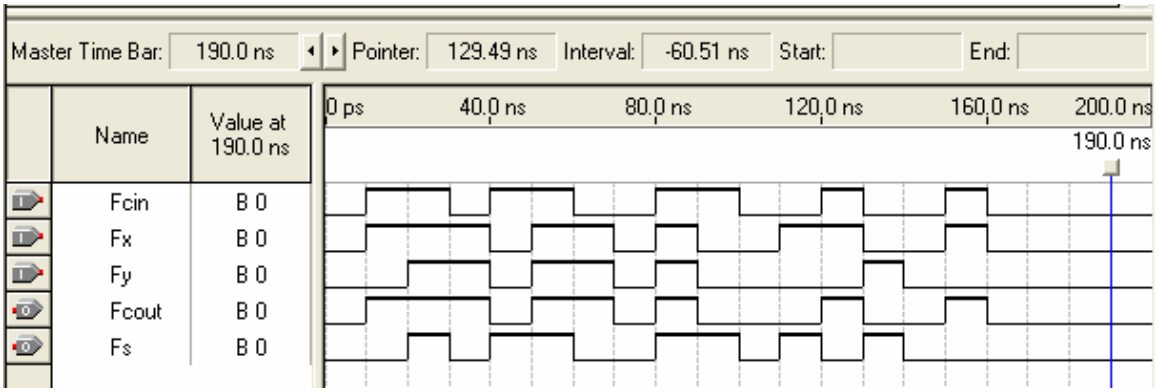
RTL VIEW (Data Flow):



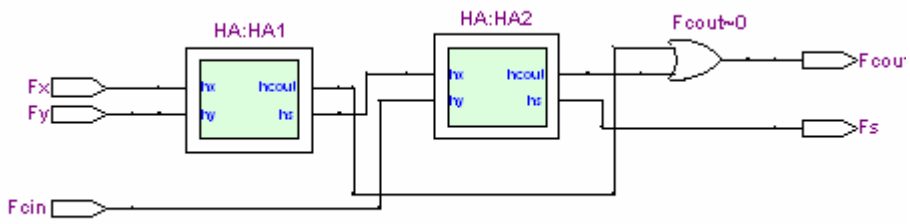
--VHDL code for BEHAVIORAL model of Full Adder

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity FA_Bhr is
        port(Fx, Fy, Fcin : in BIT; Fs, Fcout : out BIT);
end FA_Bhr;

architecture FA_struct of FA_Bhr is
        component HA
                port (hx, hy :in BIT; hs, hcout: out BIT);
        end component;
        signal s1, c1, c2 : BIT;
        begin
                HA1: HA port map (Fx, Fy, s1, c1);
                HA2: HA port map (s1, Fcin, Fs, c2);
                Fcout <= c1 OR c2;
end FA_struct;
```

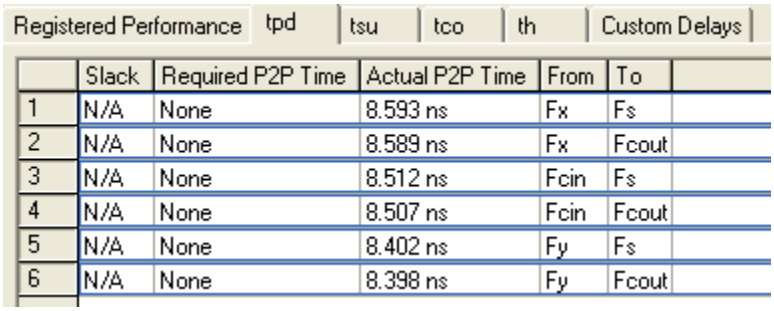
SIMULATION WAVEFORM (Behavioral):



RTL VIEW (Behavioral):



TIMING ANALYSIS RESULT (Behavioral):



Registered Performance | tpd | tsu | tco | th | Custom Delays

	Slack	Required P2P Time	Actual P2P Time	From	To
1	N/A	None	8.593 ns	Fx	Fs
2	N/A	None	8.589 ns	Fx	Fcout
3	N/A	None	8.512 ns	Fcin	Fs
4	N/A	None	8.507 ns	Fcin	Fcout
5	N/A	None	8.402 ns	Fy	Fs
6	N/A	None	8.398 ns	Fy	Fcout

Flow Status Successful - Thu Oct 19 13:30:13 2006
Quartus II Version 6.0 Build 202 06/20/2006 SP 1 SJ Web Edition
Revision Name Adder
Top-level Entity Name FA_Bhr
FamilyStratix
Met timing requirements Yes
Total logic elements 2 / 10,570 (< 1 %)
Total pins 5 / 336 (1 %)
Total virtual pins 0
Total memory bits 0 / 920,448 (0 %)
DSP block 9-bit elements 0 / 48 (0 %)
Total PLLs 0 / 6 (0 %)
Total DLLs 0 / 2 (0 %)
Device EP1S10F484C5
Timing Models Final

CONCLUSION:

1. Truth table for all the models of full adder are verified from output waveform.
2. RTL viewer and timing analysis of different models are obtained.

FULL SUBTRACTOR

AIM:

To develop a VHDL code for a full subtractor.

TRUTH TABLE

X	Y	Bin	D	Bout
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

VHDL CODE(dataflow):

```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;//standard library packages
ENTITY fullsubd IS
PORT(A,B,BORIN: IN BIT;
        DIFF,BOR:OUT BIT);//input and output declaration
END fullsubd;
ARCHITECTURE dataflow OF fullsubd IS
BEGIN
DIFF<=A XOR B XOR BORIN;
BOR<=((NOT A) AND B) OR (BORIN AND (NOT (A XOR B)));//expressions for
outputs
END dataflow;
```

VHDL Code(behavioral):

```
LIBRARY IEEE;
```

```
USE IEEE.STD_LOGIC_1164.ALL;
USE IEEE.STD_LOGIC_ARITH.ALL;//standard library packages
ENTITY fullsubb IS
PORT(A,B,BORIN: IN BIT;
        DIFF,BOR: OUT BIT); )//input and output declaration
END fullsubb;
ARCHITECTURE BEHAVE OF fullsubb IS
BEGIN
       PROCESS(A,B,BORIN)//sensitivity list
              BEGIN
                     IF A='0' AND B='0' THEN BOR<=BORIN;DIFF<=BORIN;

                     ELSIF(A='0' AND B='1' AND BORIN='0')OR (B='1' AND
A='1' AND BORIN='1')
                     THEN DIFF<='1';BOR<='1';
                     ELSIF(A='1' AND B='1' AND BORIN='0')OR (A='1' AND
B='0' AND BORIN='1')
                     THEN DIFF<='0';BOR<='0';
                     ELSIF(A='1' AND B='0' AND BORIN='0' ) THEN
BOR<='0';DIFF<='1';
                     ELSE BOR<='1';DIFF<='0';
                     END IF;
              END PROCESS;//marks end of the process
       END BEHAVE;
```

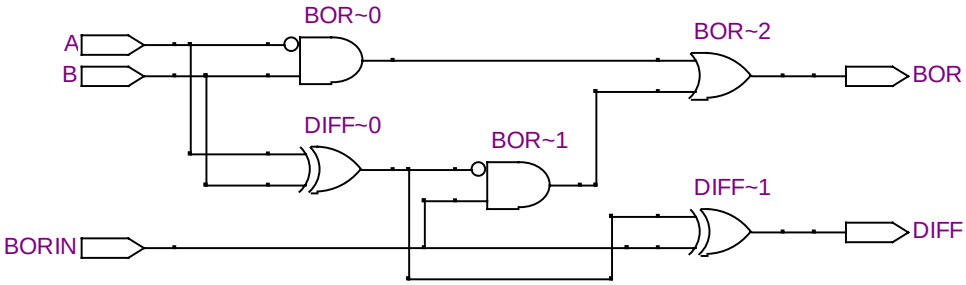
VHDL Code(Structural):

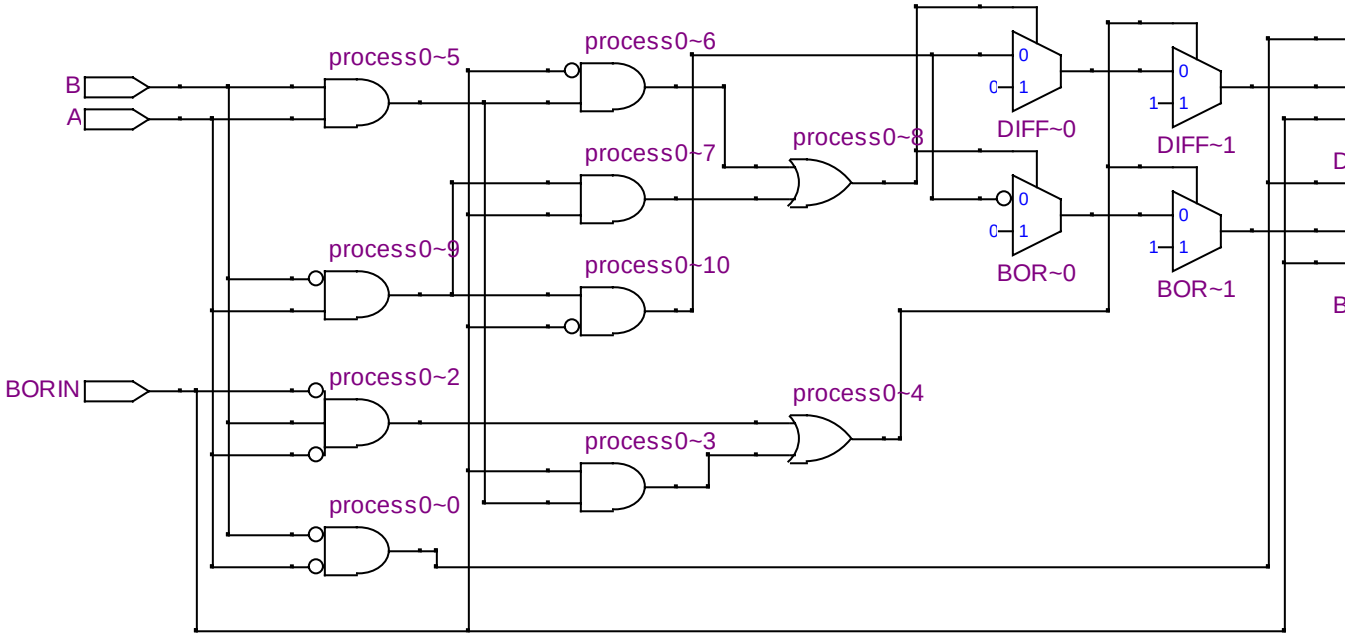
```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;//standard library packages
ENTITY fullsubs IS
PORT(A,B,BORIN: IN BIT;
       DIFF,BOR:OUT BIT); );//input and output declaration
END fullsubs;
ARCHITECTURE struc OF fullsubs IS
COMPONENT halfsubd  //basic component declarations
PORT(A,B:IN BIT;
        DIFF,BOR:OUT BIT);
END COMPONENT;
COMPONENT or2bit
PORT(A,B:IN BIT;C:OUT BIT);
END COMPONENT;
SIGNAL BOR1,BOR2,DIFF1:BIT;
BEGIN//mapping
HS1:halfsubd PORT MAP(A,B,DIFF1,BOR1);
HS2:halfsubd PORT MAP(DIFF1,BORIN,DIFF,BOR2);
```

O1:or2bit PORT MAP(BOR1,BOR2,BOR);
END struc;

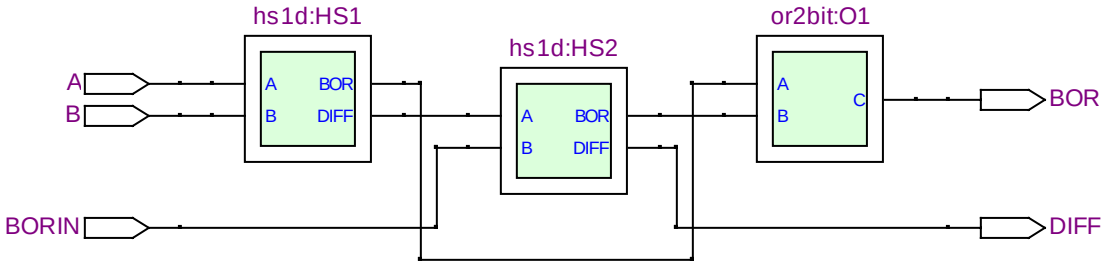
RTL Viewer (Dataflow):

RTL Viewer(Behavioral):

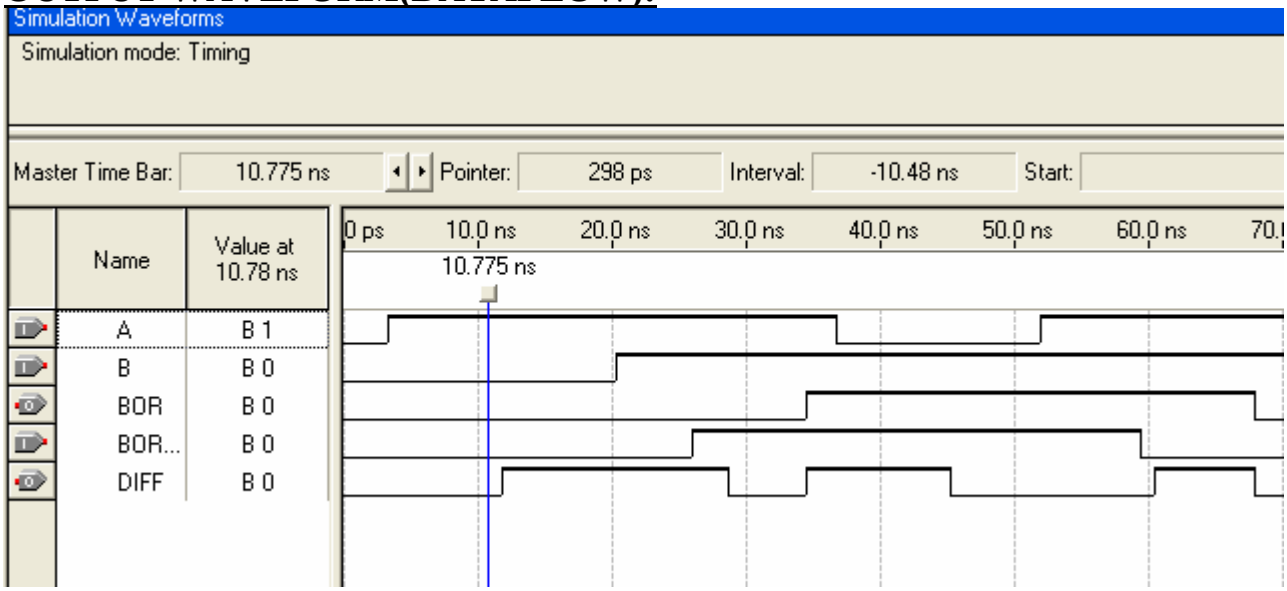




RTL Viewer(Structural):



OUTPUT WAVEFORM(DATAFLOW):

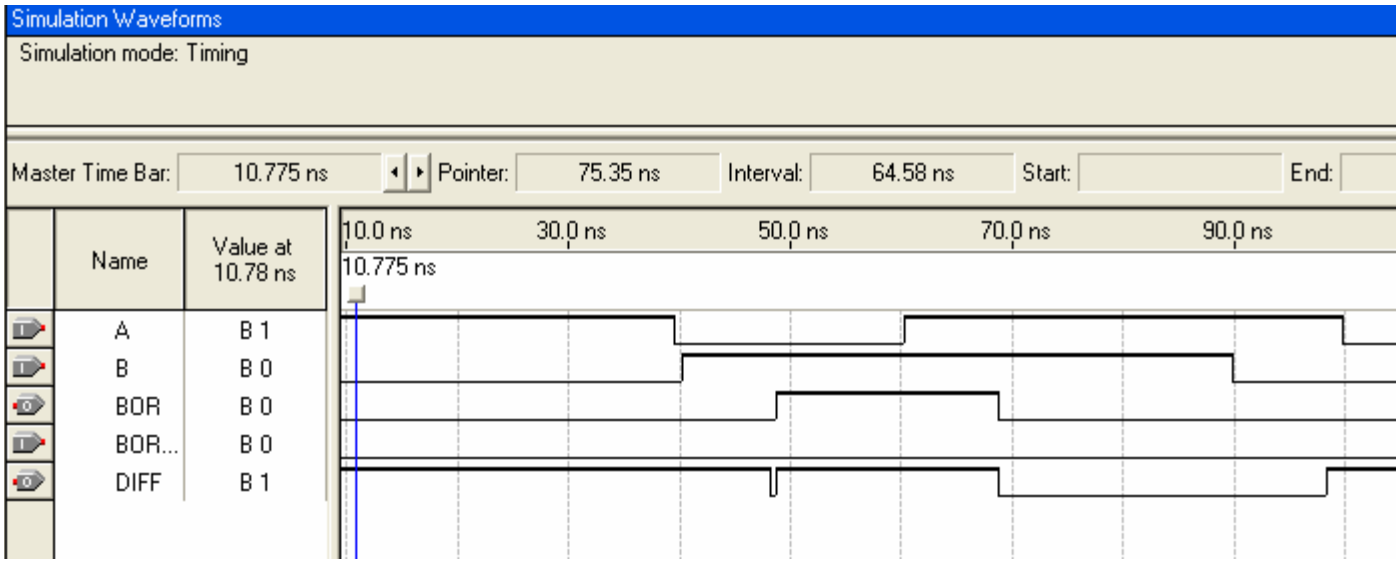


TIMING ANALYSIS(Dataflow):

Registered Performance | tpd | tsu | tco | th | Custom Delays

	Slack	Required P2P Time	Actual P2P Time	From	To
1	N/A	None	8.593 ns	BORIN	DIFF
2	N/A	None	8.589 ns	BORIN	BOR
3	N/A	None	8.512 ns	A	DIFF
4	N/A	None	8.507 ns	A	BOR
5	N/A	None	8.402 ns	B	DIFF
6	N/A	None	8.398 ns	B	BOR

OUTPUT WAVEFORM(BEHAVIORAL):

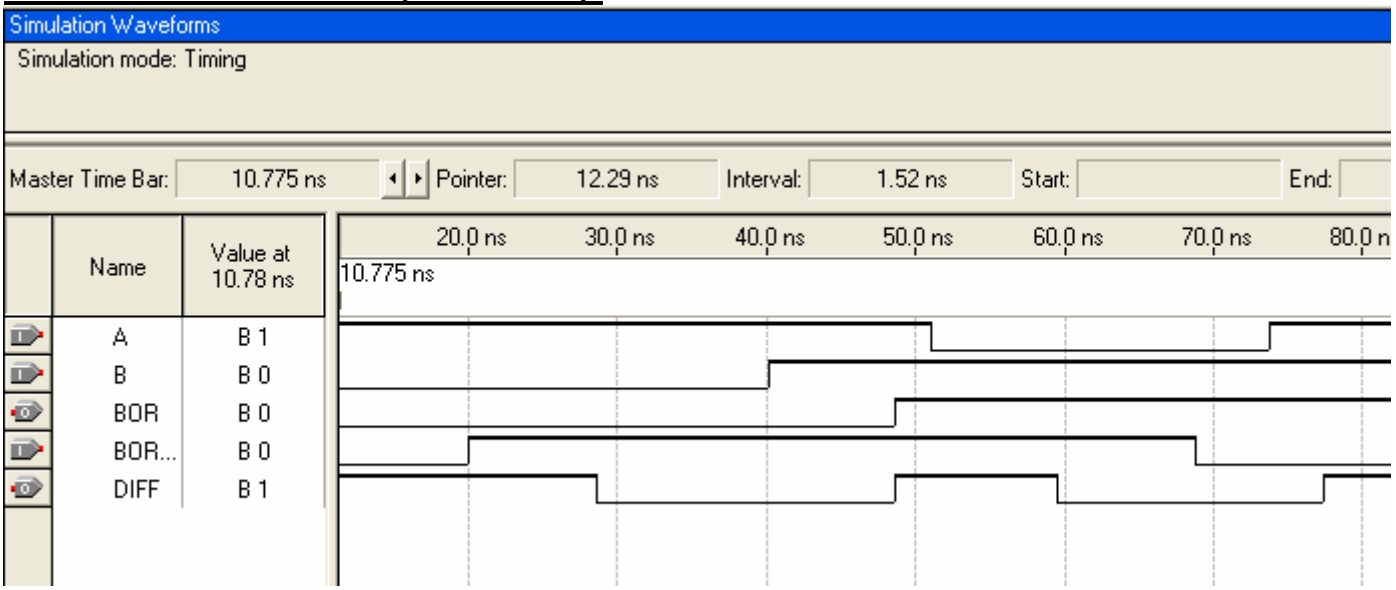


TIMING ANALYSIS(BEHAVIORAL):

Registered Performance | tpd | tsu | tco | th | Custom Delays

	Slack	Required P2P Time	Actual P2P Time	From	To
1	N/A	None	8.593 ns	B	DIFF
2	N/A	None	8.589 ns	B	BOR
3	N/A	None	8.512 ns	A	DIFF
4	N/A	None	8.507 ns	A	BOR
5	N/A	None	8.402 ns	BORIN	DIFF
6	N/A	None	8.398 ns	BORIN	BOR

OUTPUT WAVEFORM(Structural):



TIMING ANALYSIS(Structural):

Registered Performance | tpd | tsu | tco | th | Custom Delays

	Slack	Required P2P Time	Actual P2P Time	From	To
1	N/A	None	8.593 ns	BORIN	DIFF
2	N/A	None	8.589 ns	BORIN	BOR
3	N/A	None	8.512 ns	A	DIFF
4	N/A	None	8.507 ns	A	BOR
5	N/A	None	8.402 ns	B	DIFF
6	N/A	None	8.398 ns	B	BOR

CONCLUSIONS:

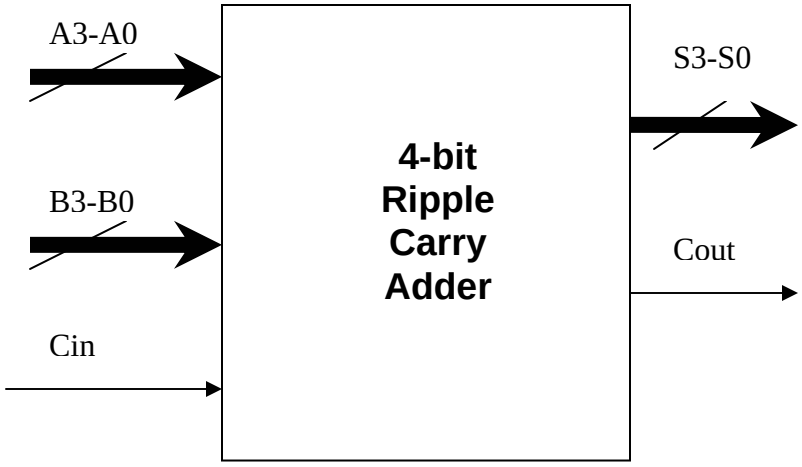
We find that the full subtractor circuit gives us the difference and borrow of two numbers and the result is same irrespective of the type of modeling.
We find from timing analysis the gate delays are also same in each type of modeling.

4-bit RIPPLE CARRY ADDER

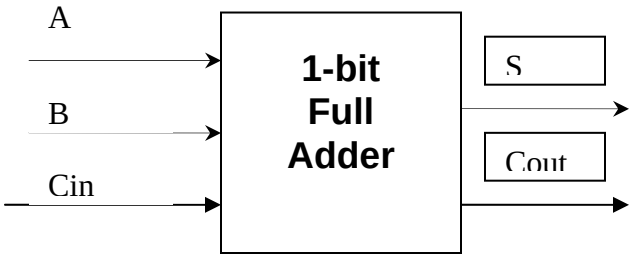
AIM:

To develop a VHDL code for a four bit ripple carry adder.

ENTITY:



BASIC COMPONENTS: 1-Bit Full Adder



Truth table for full adder

A	B	Cin	S	Cout
0	0	0	0	0
0	0	1	1	1
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

EQUATIONS FOR SUM AND CARRY :

1-bit Full Adder

S=A xor B xor Cin
Cout=AB+Cin(A xor B)

4-BIT RIPPLE CARRY ADDER

INPUTS:

A=A3A2A1A0 B=B3B2B1B0 Cin=C0

OUTPUTS:

S=S3S2S1S0 Cout=C4

EQUATIONS:

S0=A0 xor B0 xor C0
C1=A0B0+C0(A0 xor B0)

S1=A1 xor B1 xor C1
C2=A1B1+C1(A1 xor B1)

S2=A2 xor B2 xor C2
C3=A2B2+C2(A2 xor B2)

S3=A3 xor B3 xor C3
C4=A3B3+C3(A3 xor B3)

VHDL Program: Structural Model

```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;//Packages Available
ENTITY bit4srca IS
PORT(A,B:IN BIT_VECTOR(3 DOWNTO 0);
        SEL:IN BIT;//Declaration of input ports
        COUT:OUT BIT;
        X:OUT BIT_VECTOR(3 DOWNTO 0));//Declaration of output ports
END bit4srca;
ARCHITECTURE struc OF bit4srca IS
COMPONENT fa1d//basic component:1-bit full adder
PORT(X,Y,CIN:IN BIT;
         COUT:OUT BIT);
END COMPONENT;
SIGNAL C:BIT_VECTOR(3 DOWNTO 1);//Signal declaration

BEGIN//beginning of architecture
FA0:fa1d PORT MAP(A(0),B(0),Cin,S(0)C(1));//mapping for 1-bit full adder
FA1:fa1d PORT MAP(A(1),B(1),C(1),S(1),C(2));
FA2:fa1d PORT MAP(A(2),B(2),C(2),S(2),C(3));
FA3:fa1d PORT MAP(A(3),B(3),C(3),S(3),Cout);//4-full adders
END struc;//
```

Behavioral Model

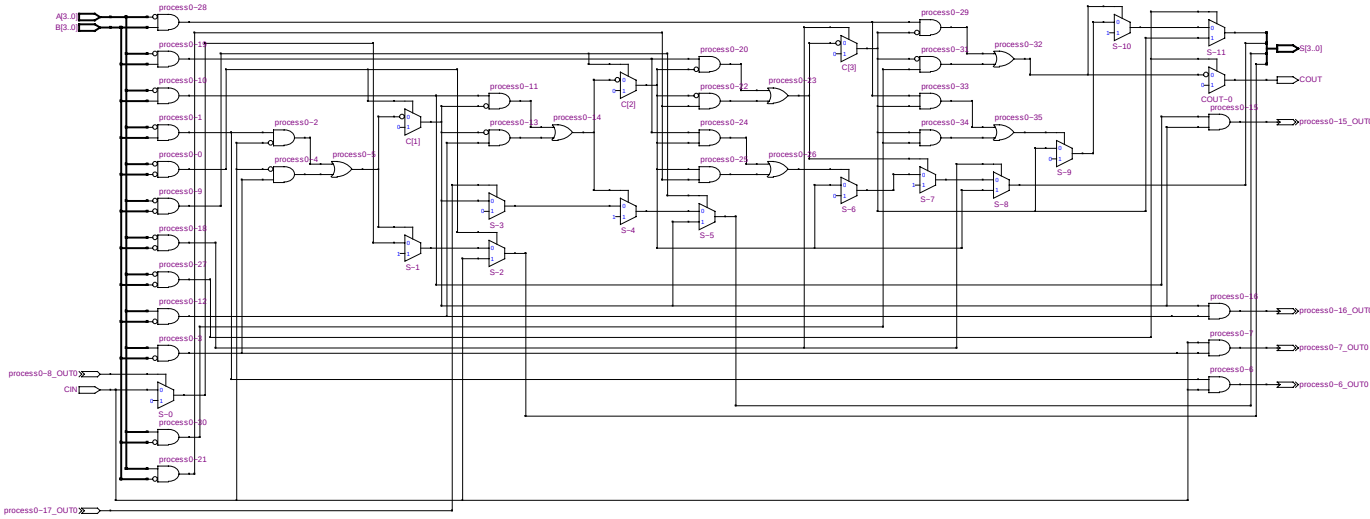
```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
USE IEEE.STD_LOGIC_ARITH.ALL;
USE IEEE.STD_LOGIC_UNSIGNED.ALL;

//entity declaration
ENTITY rca4b IS
PORT(A,B:IN BIT_VECTOR(3 DOWNTO 0);
        CIN:IN BIT ;COUT:OUT BIT;
        S:OUT BIT_VECTOR(3 DOWNTO 0));
END rca4b;

ARCHITECTURE behav OF rca4b IS
SIGNAL C:BIT_VECTOR(3 DOWNTO 1);//signal declaration.
BEGIN
      PROCESS(A,B,CIN)
            BEGIN
                  IF A(0)='0' AND B(0)='0' THEN C(1)<='1';
                        IF CIN='1' THEN S(0)<='1';
                        ELSE S(0)<='0';
                        END IF;
                  ELSIF ((A(0)='0' AND B(0)='1' AND CIN='0')OR (B(0)='0'
AND A(0)='1' AND CIN='0'))
                   THEN S(0)<='1'; C(1)<='0';
                  ELSIF(A(0)='0' AND B(0)='1' AND CIN='1')OR (A(0)='1'
AND B(0)='0' AND CIN='1')
                        THEN S(0)<='0';C(1)<='1';
                  ELSIF CIN='0' THEN C(1)<='1'; S(0)<='0';
                        ELSE C(1)<='1';S(0)<='1';
                  END IF;
```

Data Flow Model

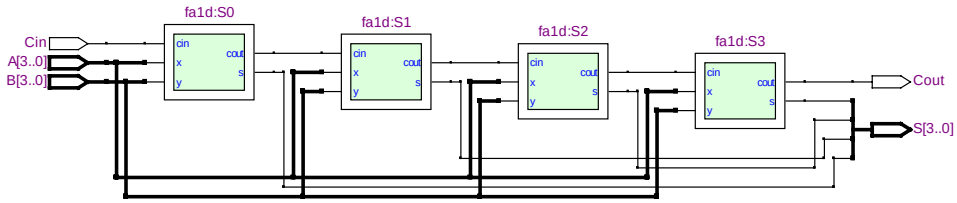
```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
USE IEEE.STD_LOGIC_ARITH.ALL;
USE IEEE.STD_LOGIC_UNSIGNED.ALL;

ENTITY rca4d IS
PORT(A,B:IN BIT_VECTOR(3 DOWNTO 0);
   Cin:IN BIT;
   Cout:OUT BIT;
   S:OUT BIT_VECTOR(3 DOWNTO 0));
END rca4d;

ARCHITECTURE dataflow OF rca4d IS
SIGNAL C:BIT_VECTOR(3 DOWNTO 1);
```

```
BEGIN
        S(0)<=A(0) XOR B(0) XOR Cin;
        C(1)<=(A(0) AND B(0)) OR (A(0)AND Cin) OR (B(0) AND Cin);
        S(1)<=A(1) XOR B(1) XOR C(1);
        C(2)<=(A(1) AND B(1)) OR (A(1)AND C(1)) OR (B(1) AND C(1));
        S(2)<=A(2) XOR B(2) XOR C(2);
        C(3)<=(A(2) AND B(2)) OR (A(2)AND C(2)) OR (B(2) AND C(2));
        S(3)<=A(3) XOR B(3) XOR C(3);
        Cout<=(A(3) AND B(3)) OR (A(3)AND C(3)) OR (B(3) AND C(3));
END dataflow;
```

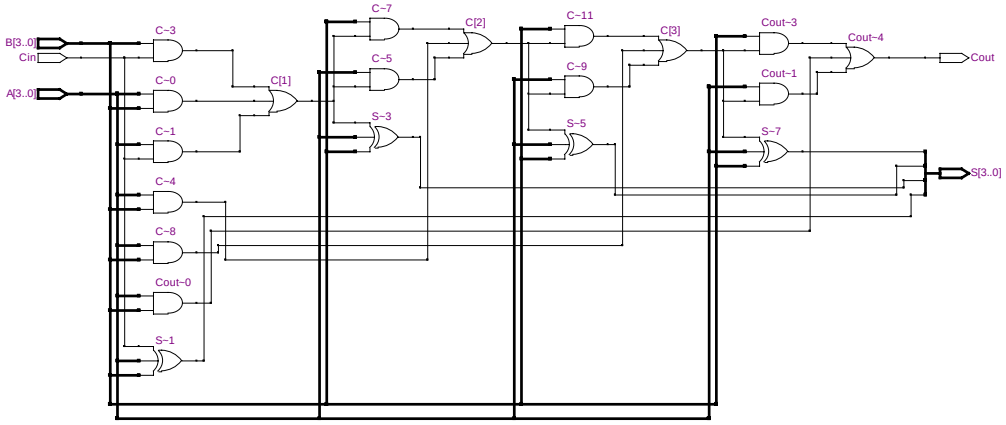
RTL Viewer (Behavioral)



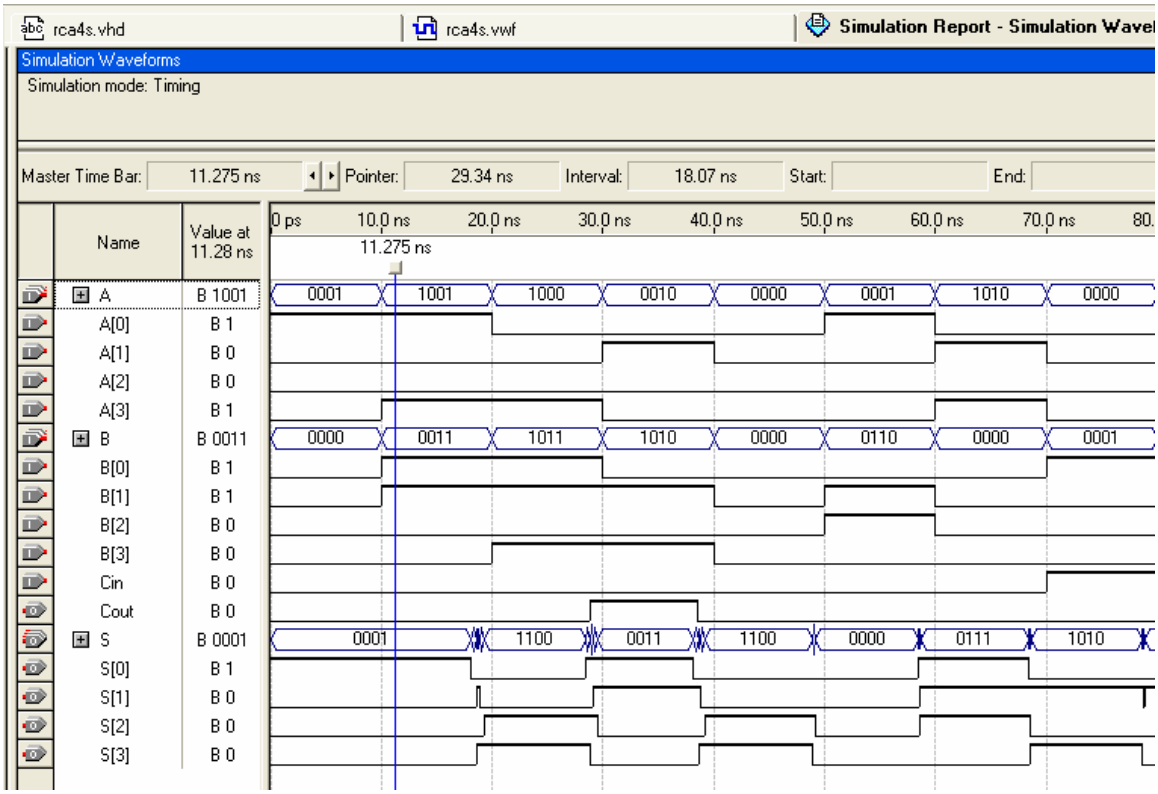
RTL Viewer(Structural)



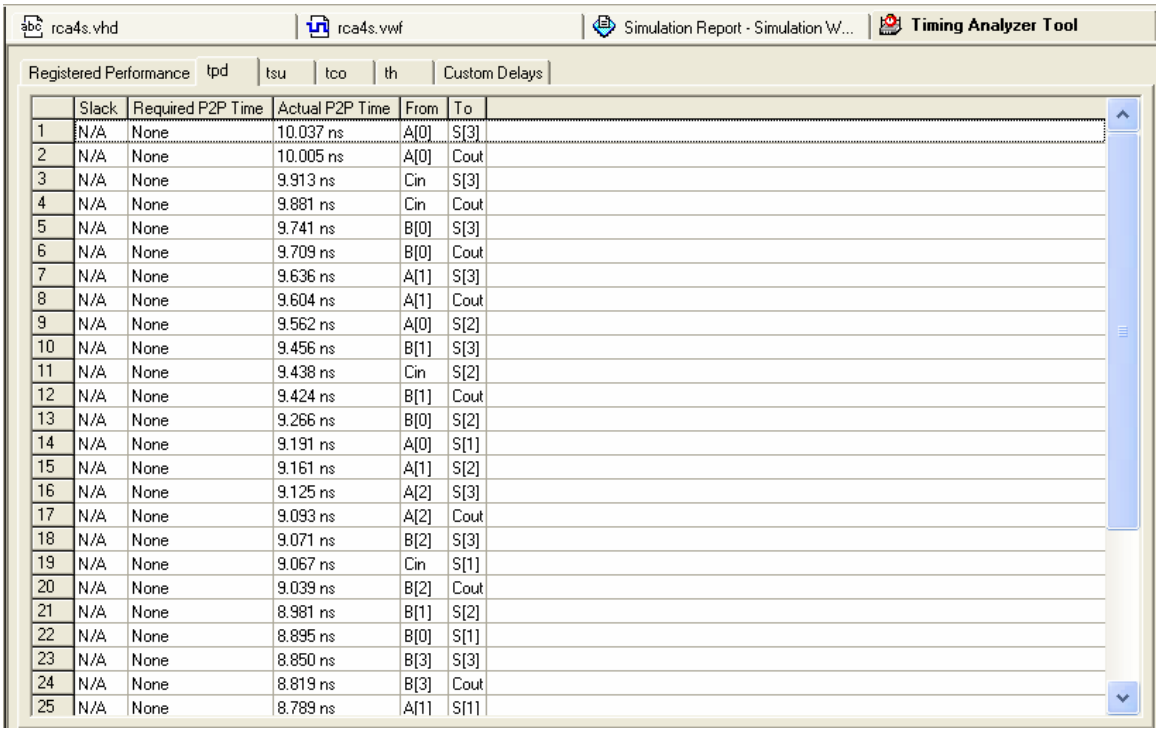
RTL Viewer(Dataflow)



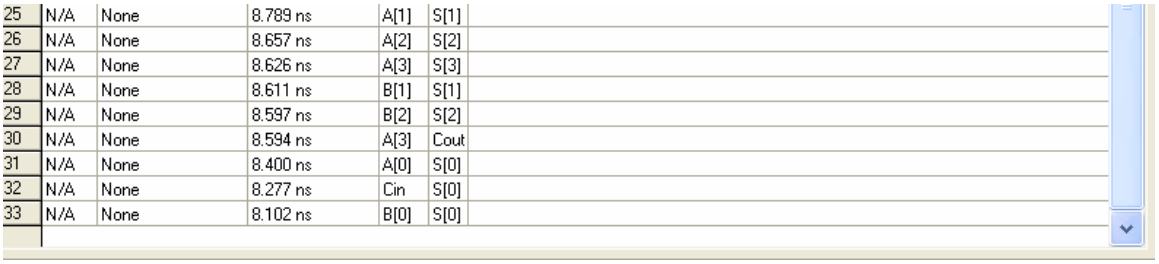
Simulation result:



Timing analysis:

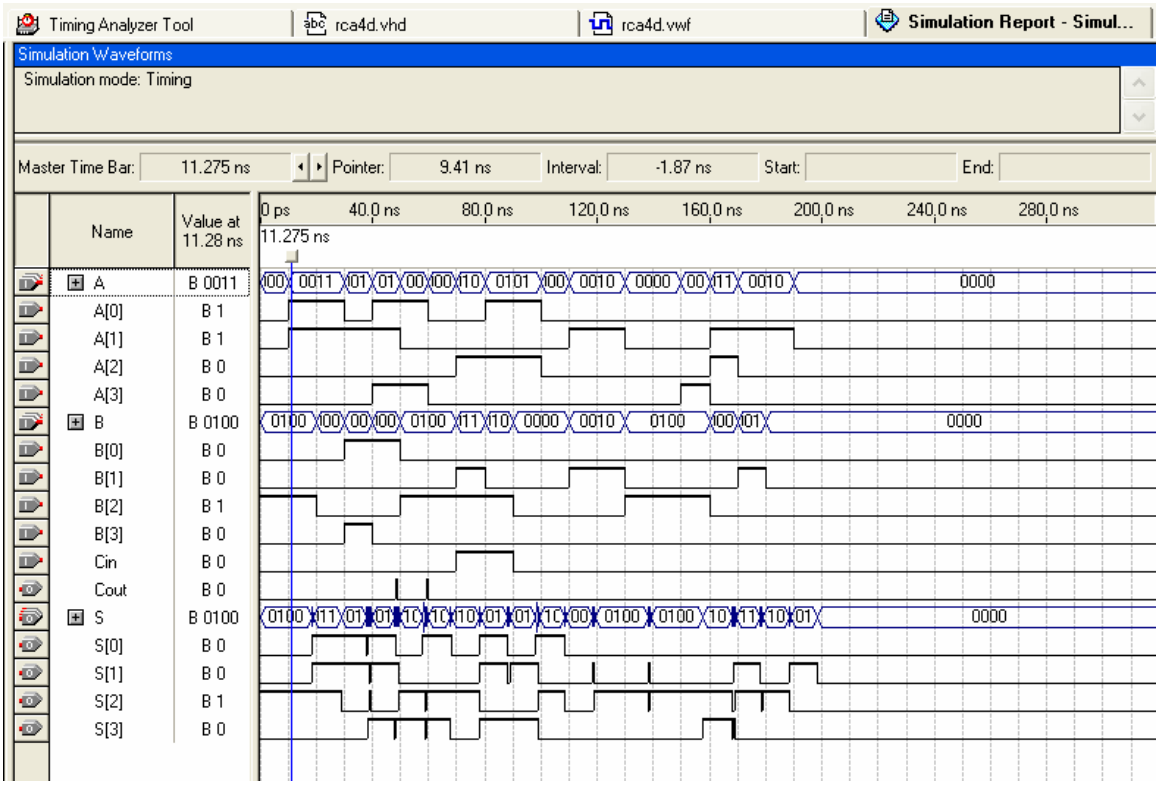


	Slack	Required P2P Time	Actual P2P Time	From	To
1	N/A	None	10.037 ns	A[0]	S[3]
2	N/A	None	10.005 ns	A[0]	Cout
3	N/A	None	9.913 ns	Cin	S[3]
4	N/A	None	9.881 ns	Cin	Cout
5	N/A	None	9.741 ns	B[0]	S[3]
6	N/A	None	9.709 ns	B[0]	Cout
7	N/A	None	9.636 ns	A[1]	S[3]
8	N/A	None	9.604 ns	A[1]	Cout
9	N/A	None	9.562 ns	A[0]	S[2]
10	N/A	None	9.456 ns	B[1]	S[3]
11	N/A	None	9.438 ns	Cin	S[2]
12	N/A	None	9.424 ns	B[1]	Cout
13	N/A	None	9.266 ns	B[0]	S[2]
14	N/A	None	9.191 ns	A[0]	S[1]
15	N/A	None	9.161 ns	A[1]	S[2]
16	N/A	None	9.125 ns	A[2]	S[3]
17	N/A	None	9.093 ns	A[2]	Cout
18	N/A	None	9.071 ns	B[2]	S[3]
19	N/A	None	9.067 ns	Cin	S[1]
20	N/A	None	9.039 ns	B[2]	Cout
21	N/A	None	8.981 ns	B[1]	S[2]
22	N/A	None	8.895 ns	B[0]	S[1]
23	N/A	None	8.850 ns	B[3]	S[3]
24	N/A	None	8.819 ns	B[3]	Cout
25	N/A	None	8.789 ns	A[1]	S[1]

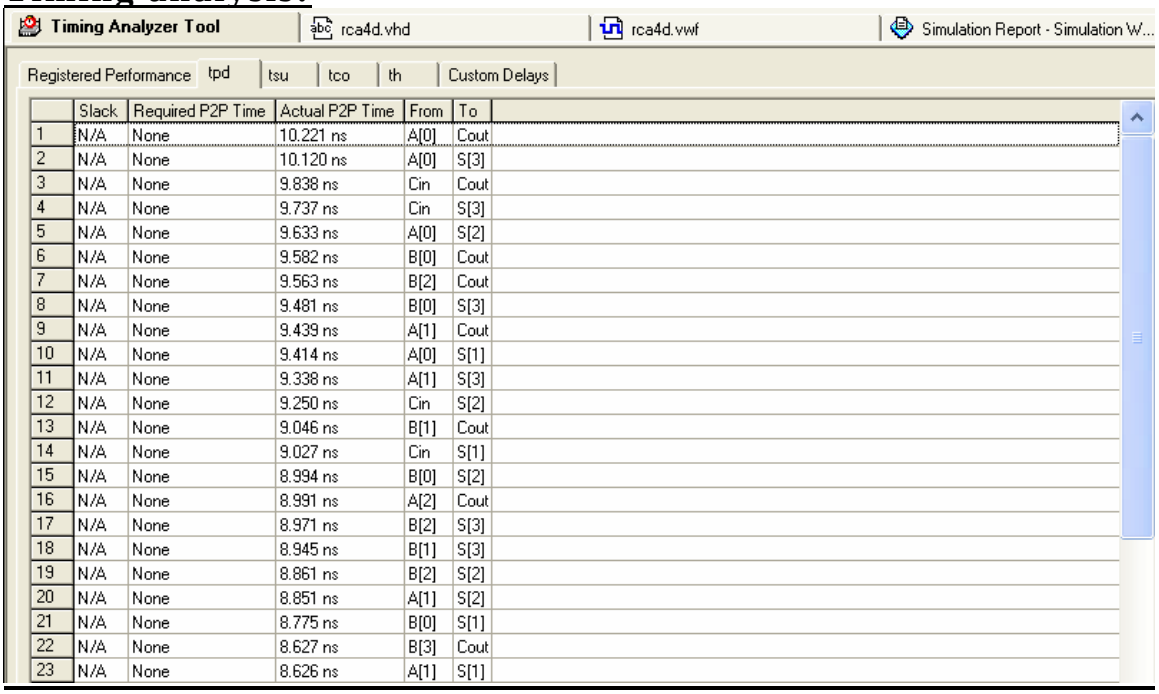


25	N/A	None	8.789 ns	A[1]	S[1]
26	N/A	None	8.657 ns	A[2]	S[2]
27	N/A	None	8.626 ns	A[3]	S[3]
28	N/A	None	8.611 ns	B[1]	S[1]
29	N/A	None	8.597 ns	B[2]	S[2]
30	N/A	None	8.594 ns	A[3]	Cout
31	N/A	None	8.400 ns	A[0]	S[0]
32	N/A	None	8.277 ns	Cin	S[0]
33	N/A	None	8.102 ns	B[0]	S[0]

Simulation result:



Timing analysis:



	Slack	Required P2P Time	Actual P2P Time	From	To
1	N/A	None	10.221 ns	A[0]	Cout
2	N/A	None	10.120 ns	A[0]	S[3]
3	N/A	None	9.838 ns	Cin	Cout
4	N/A	None	9.737 ns	Cin	S[3]
5	N/A	None	9.633 ns	A[0]	S[2]
6	N/A	None	9.582 ns	B[0]	Cout
7	N/A	None	9.563 ns	B[2]	Cout
8	N/A	None	9.481 ns	B[0]	S[3]
9	N/A	None	9.439 ns	A[1]	Cout
10	N/A	None	9.414 ns	A[0]	S[1]
11	N/A	None	9.338 ns	A[1]	S[3]
12	N/A	None	9.250 ns	Cin	S[2]
13	N/A	None	9.046 ns	B[1]	Cout
14	N/A	None	9.027 ns	Cin	S[1]
15	N/A	None	8.994 ns	B[0]	S[2]
16	N/A	None	8.991 ns	A[2]	Cout
17	N/A	None	8.971 ns	B[2]	S[3]
18	N/A	None	8.945 ns	B[1]	S[3]
19	N/A	None	8.861 ns	B[2]	S[2]
20	N/A	None	8.851 ns	A[1]	S[2]
21	N/A	None	8.775 ns	B[0]	S[1]
22	N/A	None	8.627 ns	B[3]	Cout
23	N/A	None	8.626 ns	A[1]	S[1]

24	N/A	None	8.458 ns	B[1]	S[2]
25	N/A	None	8.400 ns	A[2]	S[3]
26	N/A	None	8.396 ns	B[3]	S[3]
27	N/A	None	8.316 ns	A[0]	S[0]
28	N/A	None	8.290 ns	A[2]	S[2]
29	N/A	None	8.241 ns	A[3]	Cout
30	N/A	None	8.237 ns	B[1]	S[1]
31	N/A	None	8.236 ns	Cin	S[0]
32	N/A	None	7.828 ns	A[3]	S[3]
33	N/A	None	7.680 ns	B[0]	S[0]

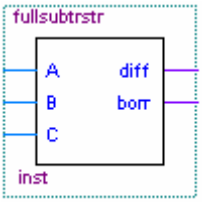
CONCLUSIONS:

We find that the when two 4-bit no.s are given as input the output is a 4-bit sum vector followed by a carry out. We also find that the delay in the first sum and carry out is 9 gates delay.

AIM:

To implement and design a full subtractor in structural model.

BLOCK DIAGRAM:



a:1st input to the full subtractor
b:2nd input to the full sutrator
c:borrow in
diff.:difference output
borr:borrow out

TRUTH TABLE:

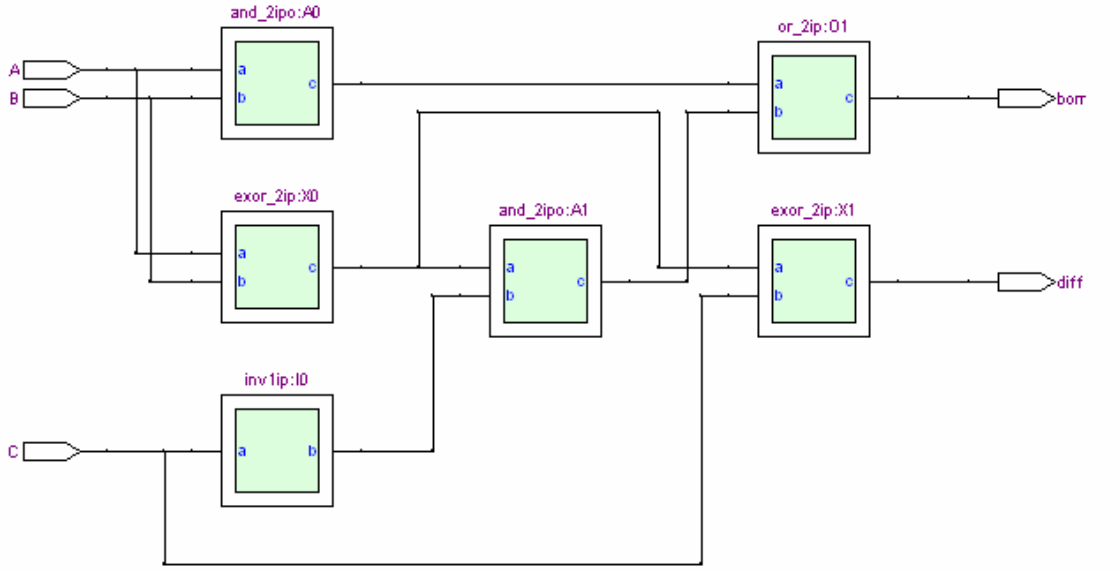
a	b	BIN	D	BOUT
0	0	0	0	0
0	0	1	1	1
1	0	0	1	0
1	0	1	0	0
0	1	0	1	1
0	1	1	0	1
1	1	0	0	0
1	1	1	1	1

VHDL CODE:

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
use ieee.std_logic_unsigned.all;
entity fullsubtrstr is
port (A,B,C: in bit; diff,borr: out bit);
end fullsubtrstr;
architecture structural of fullsubtrstr is
component exor_2ip
port (a,b :in bit;c: out bit);
```

```
end component;
component and_2ipo
port (a,b :in bit;c: out bit);
end component;
component or_2ip
port (a,b : in bit; c :out bit);
end component;
component inv1ip
port (a:in bit;b:out bit);
end component;
signal indif,inbor,bor,x:bit;
begin
X0:exor_2ip port map(A,B,indif);
X1:exor_2ip port map(indif,c,diff);
A0:and_2ipo port map(A,B,inbor);
I0:inv1ip port map(C,x);
A1:and_2ipo port map(indif,x,bor);
O1:or_2ip port map (inbor,bor,borr);
end structural;
```

RTL VIEW:



ANALYZER:

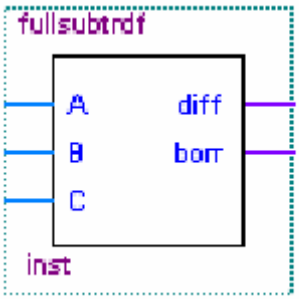
Flow Status	Successful - Mon Oct 16 22:14:34 2006
Quartus II Version	5.0 Build 148 04/26/2005 SJ Web Edition
Revision Name	fullsubtrstr
Top-level Entity Name	fullsubtrstr
Family	Cyclone
Device	EP1C6Q240C8
Timing Models	Final
Met timing requirements	Yes
Total logic elements	2 / 5,980 (< 1 %)
Total pins	5 / 185 (2 %)
Total virtual pins	0
Total memory bits	0 / 92,160 (0 %)
Total PLLs	0 / 2 (0 %)

TIMING ANALYZER:

	Type	Slack	Required Time	Actual Time	From	To	From Clock	To Clock	Failed Paths
1	Worst-case tpd	N/A	None	10.009 ns	C	diff			0
2	Total number of failed paths								0

SIMULATION WAVEFORMS:

AIM:
To design and implement a full subtractor in data flow model.

BLOCK DIAGRAM:


ENTITY:
a:1st input to the full subtractor
b:2nd input to the full sutrator
c:borrow in
diff.:difference output
borr:borrow out

TRUTH TABLE:

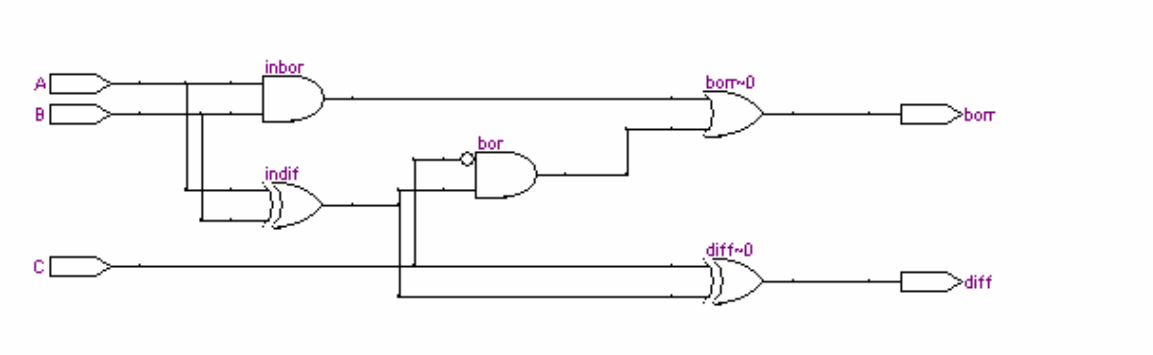
a	b	BIN	D	BOUT
0	0	0	0	0
0	0	1	1	1
1	0	0	1	0
1	0	1	0	0
0	1	0	1	1
0	1	1	0	1
1	1	0	0	0
1	1	1	1	1

VHDL CODE:

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
use ieee.std_logic_unsigned.all;
entity fullsubtrdf is
port (A,B,C:in std_logic;diff,borr: out std_logic);
end fullsubtrdf;
```

```
architecture  dataflow of fullsubtrdf is
signal inbor,bor,indif:std_logic;
begin
indif<=A xor B;
inbor<=A and B;
diff<=indif xor C;
bor<=indif and (not C);
borr<=bor or inbor;
end dataflow;
```

RTL VIEW:



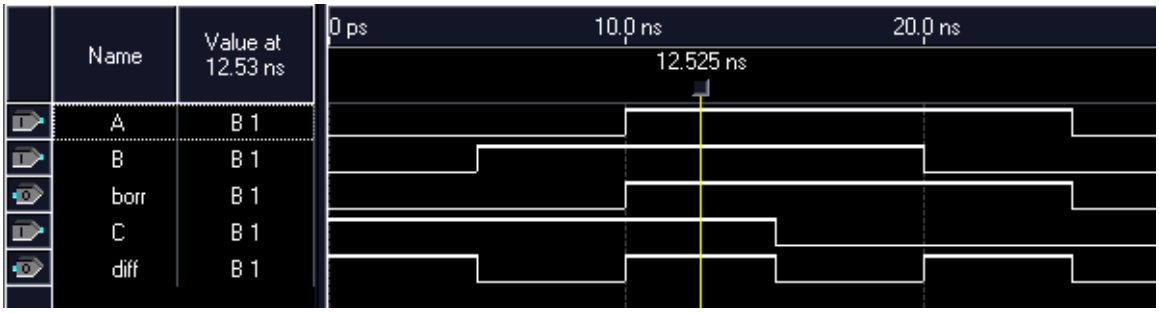
ANALYSIS:

Flow Status	Successful - Mon Oct 16 21:53:14 2006
Quartus II Version	5.0 Build 148 04/26/2005 SJ Web Edition
Revision Name	fullsubtrdf
Top-level Entity Name	fullsubtrdf
Family	Cyclone
Device	EP1C6Q240C8
Timing Models	Final
Met timing requirements	Yes
Total logic elements	2 / 5,980 (< 1 %)
Total pins	5 / 185 (2 %)
Total virtual pins	0
Total memory bits	0 / 92,160 (0 %)
Total PLLs	0 / 2 (0 %)

TIMING ANALYSIS:

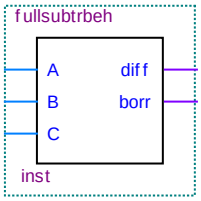
	Type	Slack	Required Time	Actual Time	From	To	From Clock	To Clock	Failed Paths
1	Worst-case tpd	N/A	None	10.009 ns	B	diff			0
2	Total number of failed paths								0

SIMULATION WAVEFORMS:



AIM: To design a full subtractor and implement its behavioural model using VHDL.

BLOCK DIAGRAM:

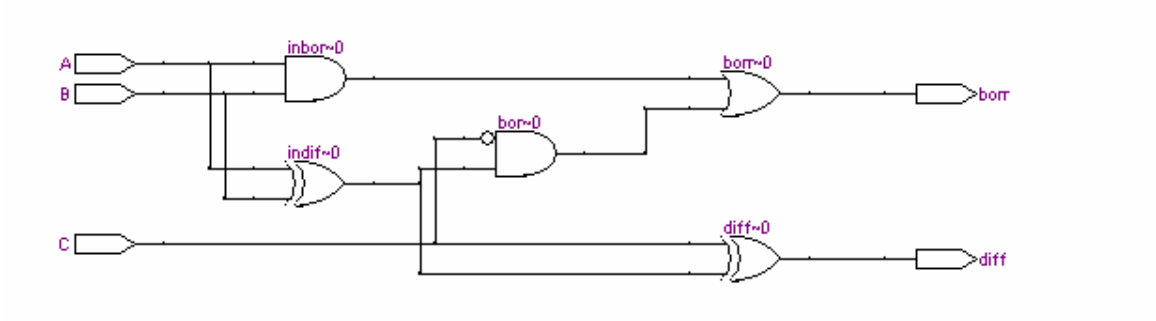


a,b:inputs
c: borrow input.
diff: difference output.
borr: borrow output

VHDL CODE:

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
use ieee.std_logic_unsigned.all;
entity fullsubtrbeh is
port (A,B,C : in std_logic;diff, borr: out std_logic);
end fullsubtrbeh;
architecture BEH of fullsubtrbeh is
begin
process(A,B,C)
variable indif,inbor,bor:std_logic;
begin
indif:=a xor b;
inbor:=a and b;
diff<=indif xor c;
bor:=indif and (not c);
borr<=inbor or bor;
end process;
end BEH;
```

RTL VIEW:



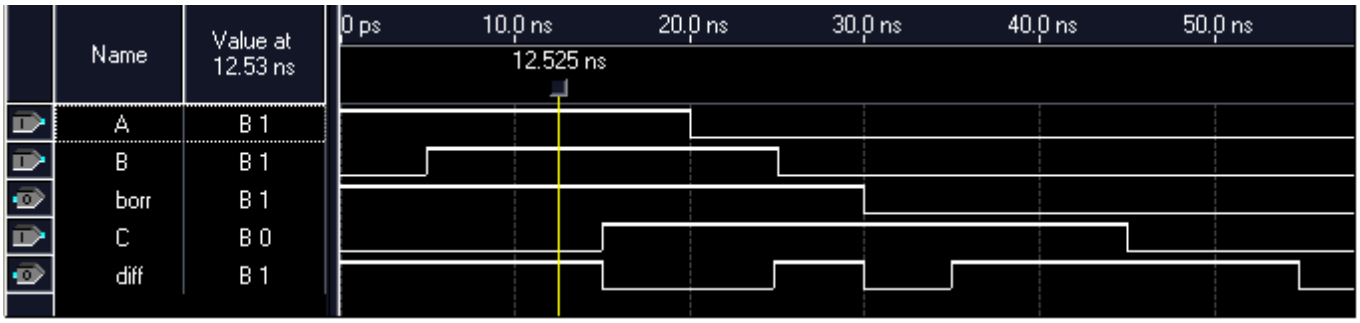
ANALYZER:

Flow Status	Successful - Mon Oct 16 21:27:31 2006
Quartus II Version	5.0 Build 148 04/26/2005 SJ Web Edition
Revision Name	fullsubtrbeh
Top-level Entity Name	fullsubtrbeh
Family	Cyclone
Device	EP1C6Q240C8
Timing Models	Final
Met timing requirements	Yes
Total logic elements	2 / 5,980 (< 1 %)
Total pins	5 / 185 (2 %)
Total virtual pins	0
Total memory bits	0 / 92,160 (0 %)
Total PLLs	0 / 2 (0 %)

TIMING ANALYSIS:

	Type	Slack	Required Time	Actual Time	From	To	From Clock	To Clock	Failed Paths
1	Worst-case tpd	N/A	None	10.009 ns	B	diff			0
2	Total number of failed paths								0

SIMULATION WAVEFORMS:



CONCLUSION:

1. Truth table for all the models of 4 bit subtractor are verified from output waveform.

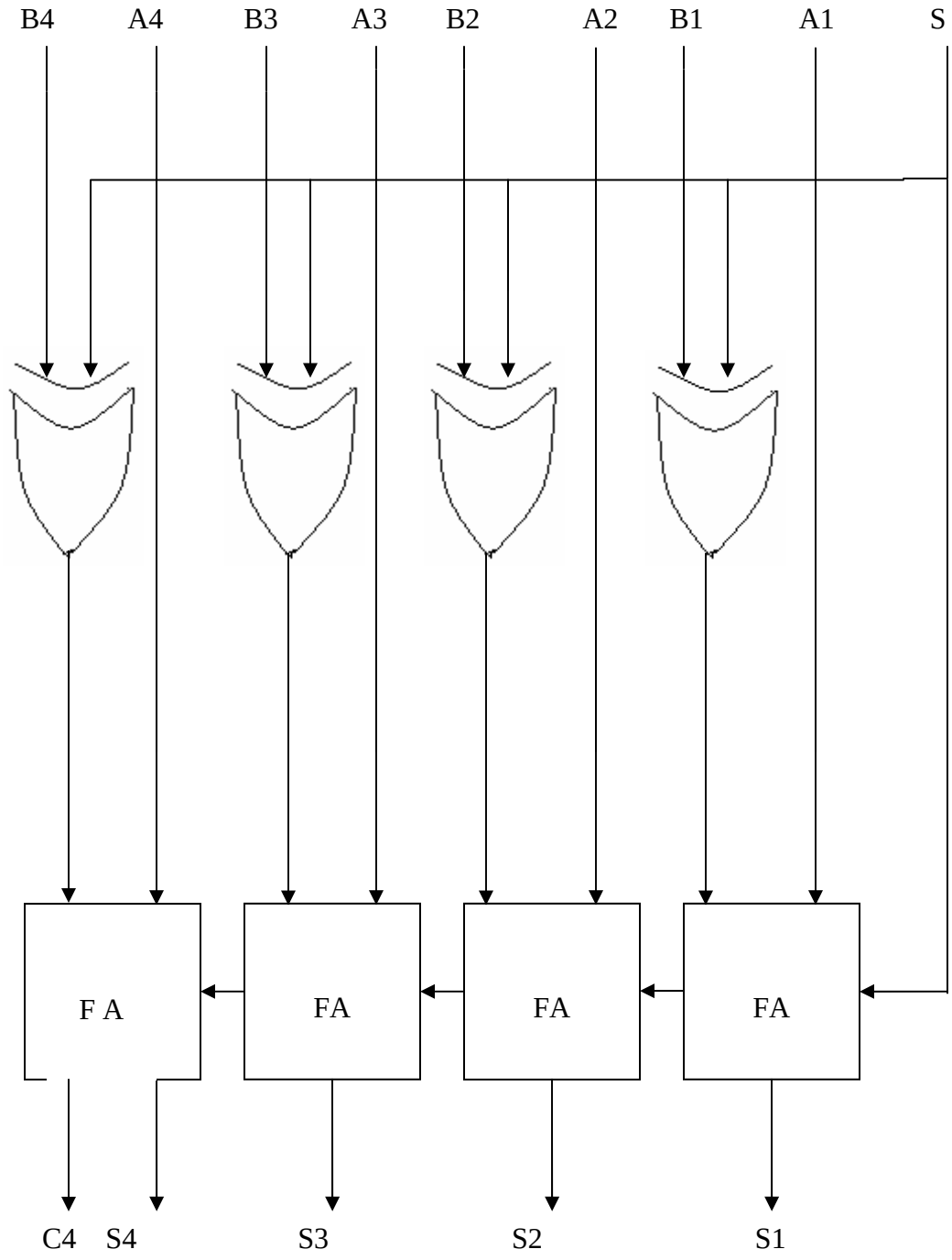
2. RTL viewer and timing analysis of different models are obtained.

4-BIT ADDER CUM SUBTRACTOR

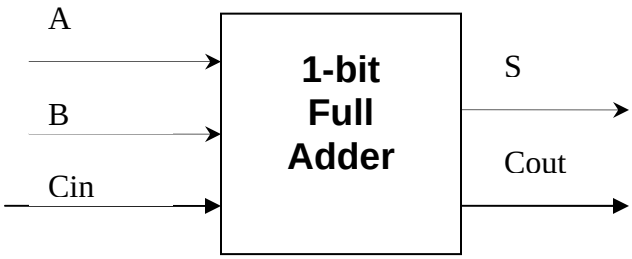
AIM:

To design a 4-bit adder cum subtractor

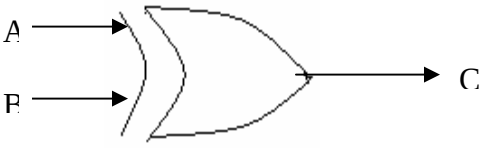
ENTITY:



BASIC COMPONENT :
1)FULL ADDER



2)XOR GATE:



TRUTH TABLES:
1)FULL ADDER

Truth table for full adder

A	B	Cin	S	Cout
0	0	0	0	0
0	0	1	1	1
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

VHDL CODE (Structural):

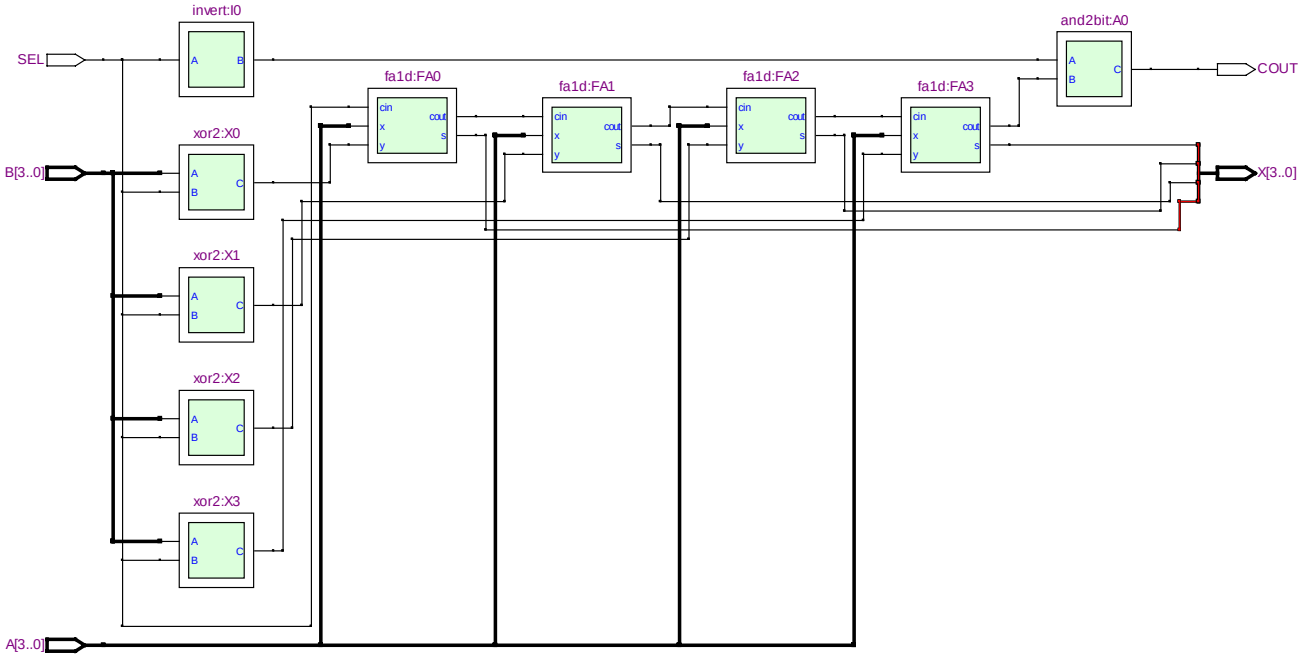
```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;//standard library packages
ENTITY addersub4s IS
PORT(A,B:IN BIT_VECTOR(3 DOWNTO 0);
        SEL:IN BIT;
        COUT:OUT BIT;
        X:OUT BIT_VECTOR(3 DOWNTO 0));//inputs and outputs declaration
END addersub4s;
ARCHITECTURE struc OF addersub4s IS
```

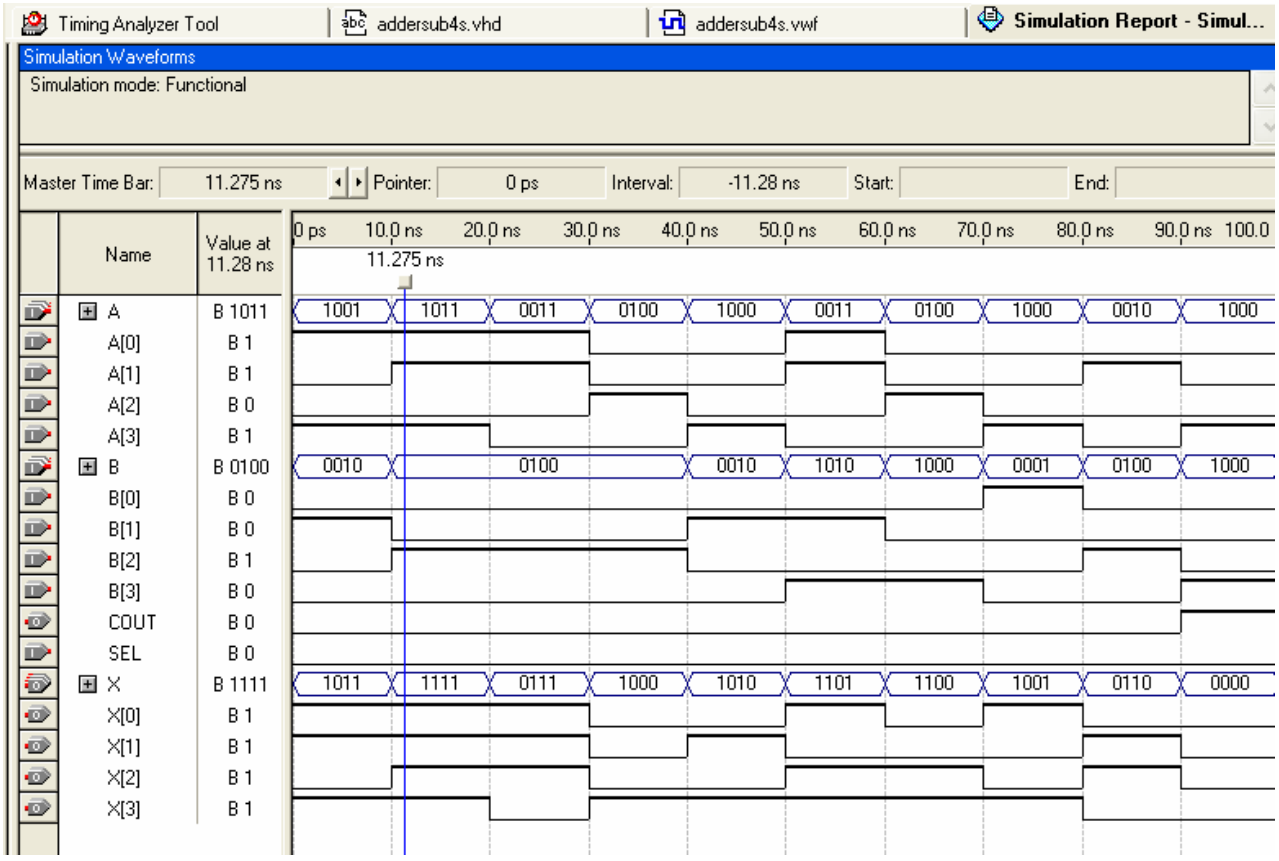
```
COMPONENT fa1d
PORT(X,Y,CIN:IN BIT;
        S,COUT:OUT BIT);
END COMPONENT;
COMPONENT xor2 //basic components used for realization…
PORT(A,B:IN BIT;C:OUT BIT);
END COMPONENT;
COMPONENT and2bit
PORT(A,B:IN BIT;C:OUT BIT);
END COMPONENT;
COMPONENT invert
PORT(A:IN BIT;B:OUT BIT);
END COMPONENT;
SIGNAL E:BIT_VECTOR(3 DOWNTO 0);
SIGNAL C:BIT_VECTOR(3 DOWNTO 1);
SIGNAL carry,non_sel:BIT;
BEGIN//mapping with basic gates
X0:xor2 PORT MAP(B(0),SEL,E(0));
FA0:fa1d PORT MAP(A(0),E(0),SEL,X(0),C(1));
X1:xor2 PORT MAP(B(1),SEL,E(1));
FA1:fa1d PORT MAP(A(1),E(1),C(1),X(1),C(2));
X2:xor2 PORT MAP(B(2),SEL,E(2));
FA2:fa1d PORT MAP(A(2),E(2),C(2),X(2),C(3));
X3:xor2 PORT MAP(B(3),SEL,E(3));
FA3:fa1d PORT MAP(A(3),E(3),C(3),X(3),carry);
I0:invert PORT MAP(SEL,non_sel);
A0:and2bit PORT MAP(non_sel,carry,COUT);
END struc;
```

VHDL Code(Dataflow):

```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
ENTITY addersub4d IS
PORT(A,B:IN BIT_VECTOR(3 DOWNTO 0);
        SEL:IN BIT;
        COUT:OUT BIT;
        X:OUT BIT_VECTOR(3 DOWNTO 0));
END addersub4d;
ARCHITECTURE dataflow OF addersub4d IS
BEGIN
        PROCESS(A,B,SEL)
                VARIABLE S:BIT_VECTOR(3 DOWNTO 0);
                VARIABLE C:BIT_VECTOR(4 DOWNTO 0);
                BEGIN
                C(0):=SEL;
                FOR i IN 0 TO 3 LOOP
                S(i):=A(i) XOR B(i) XOR C(i) XOR SEL;
                C(i+1):=(B(i) XOR SEL) AND A(i);
                END LOOP;
        COUT<=C(4);
        X<=S;
        END PROCESS;
END dataflow;
```

RTL Viewer(Structural)



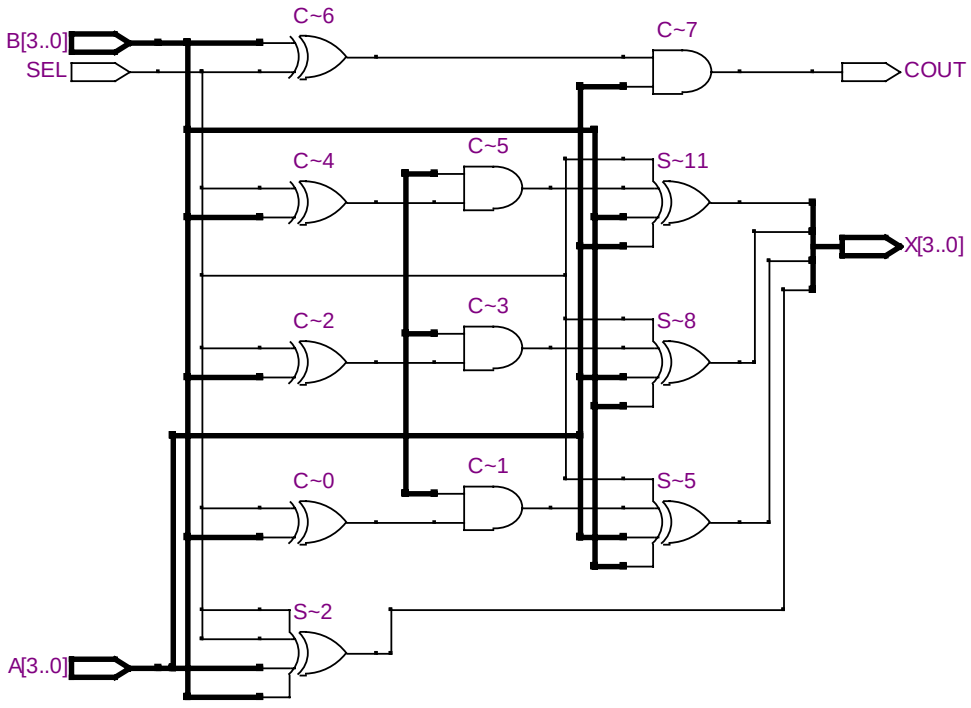


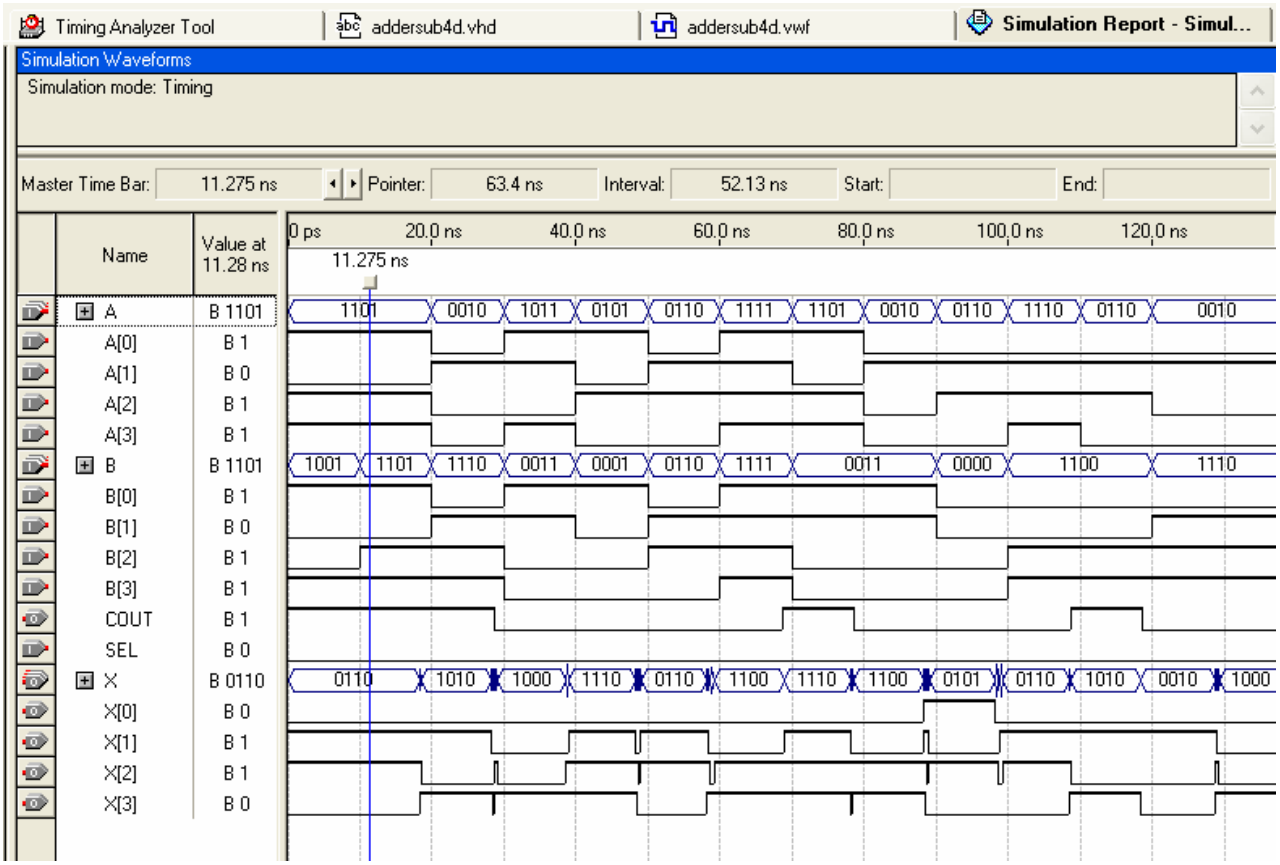
Timing Analyzer Tool | addersub4s.vhd | addersub4s.vwf | Simulation Report - Simulation W...

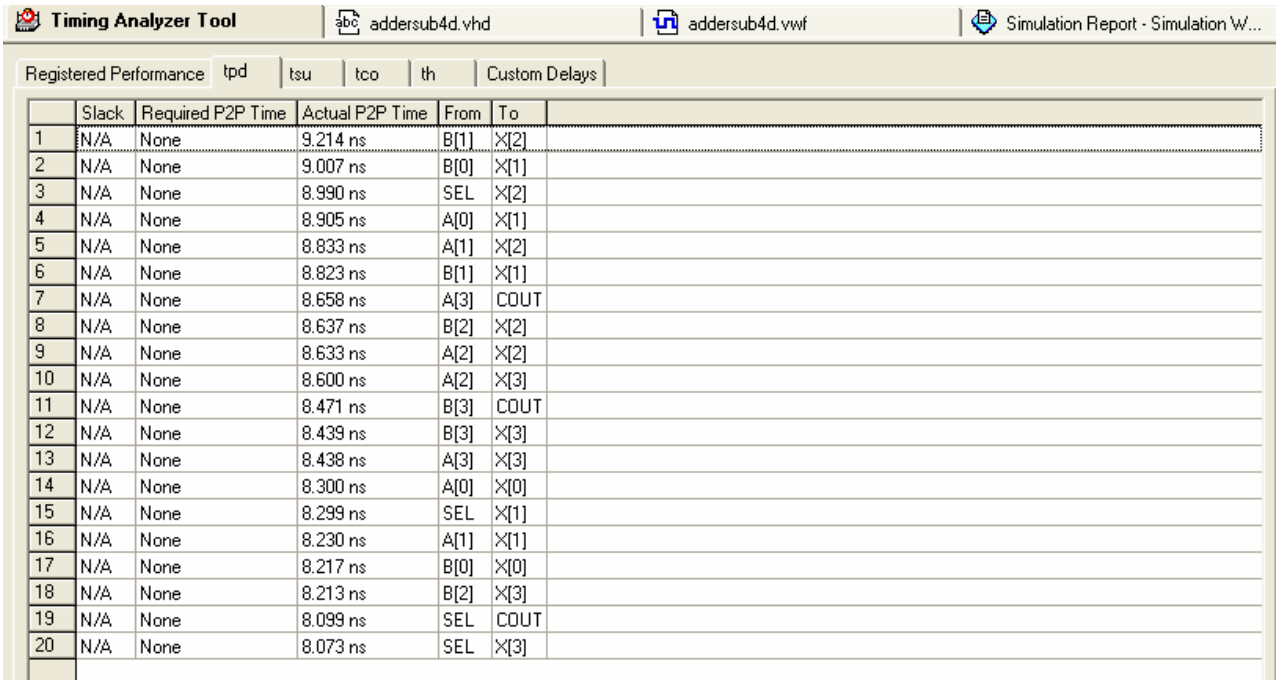
Registered Performance | tpd | tsu | tco | th | Custom Delays

	Slack	Required P2P Time	Actual P2P Time	From	To
1	N/A	None	10.193 ns	A[0]	COUT
2	N/A	None	10.192 ns	A[0]	X[3]
3	N/A	None	10.188 ns	SEL	COUT
4	N/A	None	10.187 ns	SEL	X[3]
5	N/A	None	9.864 ns	B[1]	COUT
6	N/A	None	9.863 ns	B[1]	X[3]
7	N/A	None	9.820 ns	A[0]	X[1]
8	N/A	None	9.816 ns	SEL	X[1]
9	N/A	None	9.693 ns	A[1]	COUT
10	N/A	None	9.692 ns	A[1]	X[3]
11	N/A	None	9.607 ns	A[0]	X[2]
12	N/A	None	9.602 ns	SEL	X[2]
13	N/A	None	9.484 ns	B[1]	X[1]
14	N/A	None	9.305 ns	A[1]	X[1]
15	N/A	None	9.278 ns	B[1]	X[2]
16	N/A	None	9.213 ns	B[2]	COUT
17	N/A	None	9.212 ns	B[2]	X[3]
18	N/A	None	9.107 ns	A[1]	X[2]
19	N/A	None	9.049 ns	B[0]	COUT
20	N/A	None	9.048 ns	B[0]	X[3]
21	N/A	None	8.867 ns	B[3]	COUT
22	N/A	None	8.866 ns	B[3]	X[3]
23	N/A	None	8.782 ns	A[0]	X[0]
24	N/A	None	8.677 ns	B[0]	X[1]
25	N/A	None	8.629 ns	B[2]	X[2]

RTL Viewer(Dataflow):







	Slack	Required P2P Time	Actual P2P Time	From	To
1	N/A	None	9.214 ns	B[1]	X[2]
2	N/A	None	9.007 ns	B[0]	X[1]
3	N/A	None	8.990 ns	SEL	X[2]
4	N/A	None	8.905 ns	A[0]	X[1]
5	N/A	None	8.833 ns	A[1]	X[2]
6	N/A	None	8.823 ns	B[1]	X[1]
7	N/A	None	8.658 ns	A[3]	COUT
8	N/A	None	8.637 ns	B[2]	X[2]
9	N/A	None	8.633 ns	A[2]	X[2]
10	N/A	None	8.600 ns	A[2]	X[3]
11	N/A	None	8.471 ns	B[3]	COUT
12	N/A	None	8.439 ns	B[3]	X[3]
13	N/A	None	8.438 ns	A[3]	X[3]
14	N/A	None	8.300 ns	A[0]	X[0]
15	N/A	None	8.299 ns	SEL	X[1]
16	N/A	None	8.230 ns	A[1]	X[1]
17	N/A	None	8.217 ns	B[0]	X[0]
18	N/A	None	8.213 ns	B[2]	X[3]
19	N/A	None	8.099 ns	SEL	COUT
20	N/A	None	8.073 ns	SEL	X[3]

CONCLUSIONS:

This circuit is used as both an adder and subtractor depending on the selection input. if selection line is 1 then it is subtraction and if zero addition. subtraction is done by complimenting the number and adding. So the basic components used are full adder, xor gate and and gate.

4-bit CARRYLOOK AHEAD ADDER

AIM:

To develop a VHDL code for a four bit carrylook ahead adder.

EQUATIONS FOR 4-BIT CARRY LOOK AHEAD ADDER:

INPUTS:

A=A3A2A1A0 B=B3B2B1B0 Cin=C0

OUTPUTS:

S=S3S2S1S0 Cout=C4

VARIABLES:

P=P0,P1,P2,P3 G=G0,G1,G2,G3

EQUATIONS:

S0=A0 xor B0 xor C0
G0=A0+B0
P0=A0B0
C1= G0+P0C0

S1=A1 xor B1 xor C1
G1=A1+B1
P1=A1B1
C2=G1+P1G0+P1P0C0

S2=A2 xor B2 xor C2
G2=A2+B2
P2=A2B2
C3=G2+P2G1+P2P1G0+P2P1P0C0

S3=A3 xor B3 xor C3
G3=A3+B3
P3=A3B3
C4=G3+P3G2+P3P2G1+P3P2P1G0+P3P2P1P0C0

VHDL Code: Structural Model

```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
USE IEEE.STD_LOGIC_ARITH.ALL;//Standard packages available

ENTITY cla4s IS
PORT(A,B: IN BIT_VECTOR(3 DOWNTO 0);
        CIN:IN BIT;COUT: OUT BIT; //Input and outputdeclaration
        S:OUT BIT_VECTOR(3 DOWNTO 0));
END cla4s;

ARCHITECTURE struc OF cla4s IS
SIGNAL C:BIT_VECTOR(3 DOWNTO 1);
SIGNAL G,P:BIT_VECTOR(3 DOWNTO 0);
SIGNAL E:BIT_VECTOR(9 DOWNTO 0);//Signal declarations
COMPONENT and2bit IS
PORT(A,B:IN BIT;
        C:OUT BIT);
END COMPONENT;
COMPONENT or2bit IS
PORT(A,B:IN BIT;
        C:OUT BIT);
END COMPONENT;
COMPONENT or3bit IS
PORT(A,B,C:IN BIT;
        D:OUT BIT);
END COMPONENT;
COMPONENT or4bit IS
PORT(A,B,C,D:IN BIT;
        E:OUT BIT);
END COMPONENT;
COMPONENT or5bit IS
PORT(A,B,C,D,E:IN BIT;
        F:OUT BIT);
END COMPONENT;
COMPONENT and3bit IS
PORT(A,B,C:IN BIT;
        D:OUT BIT);
END COMPONENT;
COMPONENT and4bit IS
PORT(A,B,C,D:IN BIT;
        E:OUT BIT);
END COMPONENT;
COMPONENT and5bit IS
PORT(A,B,C,D,E:IN BIT;
        F:OUT BIT);
END COMPONENT;
COMPONENT xor3 IS
PORT(A,B,C:IN BIT;
        D:OUT BIT);
END COMPONENT;//Components used..declaration
```

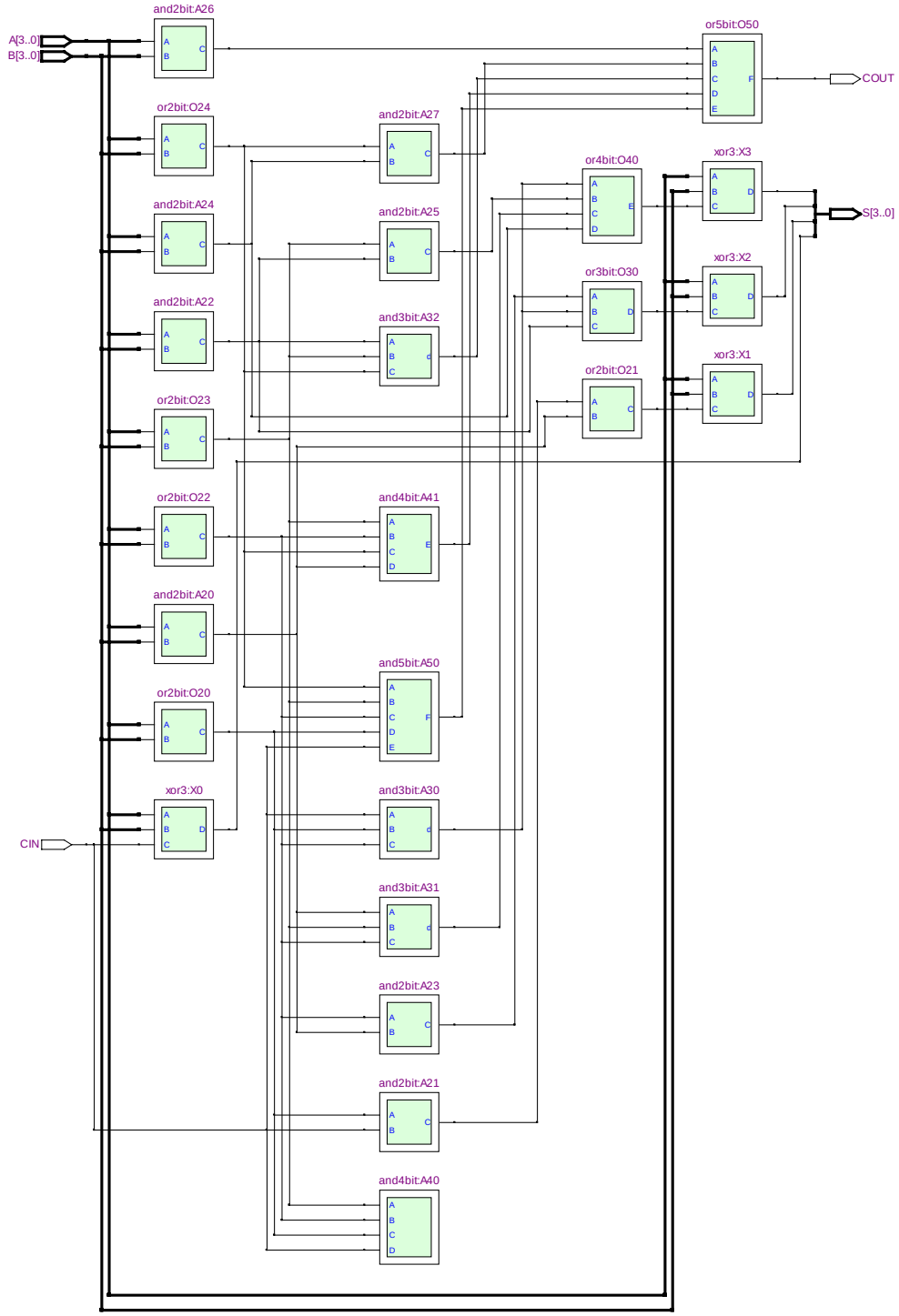
```
BEGIN
        X0:xor3 PORT MAP(A(0),B(0),CIN,S(0));//port mapping with
        A20:and2bit PORT MAP(A(0),B(0),G(0));//basic components
        O20:or2bit PORT MAP(A(0),B(0),P(0));
        A21:and2bit PORT MAP(P(0),CIN,E(0));
        O21:or2bit PORT MAP(E(0),G(0),C(1));
        X1:xor3 PORT MAP(A(1),B(1),C(1),S(1));
        A22:and2bit PORT MAP(A(1),B(1),G(1));
        O22:or2bit PORT MAP(A(1),B(1),P(1));
        A23:and2bit PORT MAP(P(1),G(0),E(1));
        A30:and3bit PORT MAP(CIN,P(0),P(1),E(2));
        O30:or3bit PORT MAP(E(1),E(2),G(1),C(2));
        X2:xor3 PORT MAP(A(2),B(2),C(2),S(2));
        A24:and2bit PORT MAP(A(2),B(2),G(2));
        O23:or2bit PORT MAP(A(2),B(2),P(2));
        A25:and2bit PORT MAP(P(2),G(1),E(3));
        A31:and3bit PORT MAP(G(0),P(2),P(1),E(4));
        A40:and4bit PORT MAP(P(2),P(1),P(0),CIN,E(5));
        O40:or4bit PORT MAP(E(2),E(3),E(4),G(2),C(3));
        X3:xor3 PORT MAP(A(3),B(3),C(3),S(3));
        A26:and2bit PORT MAP(A(3),B(3),G(3));
        O24:or2bit PORT MAP(A(3),B(3),P(3));
        A27:and2bit PORT MAP(P(3),G(2),E(6));
        A32:and3bit PORT MAP(G(1),P(2),P(3),E(7));
        A41:and4bit PORT MAP(P(2),P(1),P(3),G(0),E(8));
        A50:and5bit PORT MAP(P(3),P(2),P(1),P(0),CIN,E(9));
        O50:or5bit PORT MAP(G(3),E(6),E(7),E(8),E(9),COUT);
END struc;
```

VHDL CODE: Data Flow

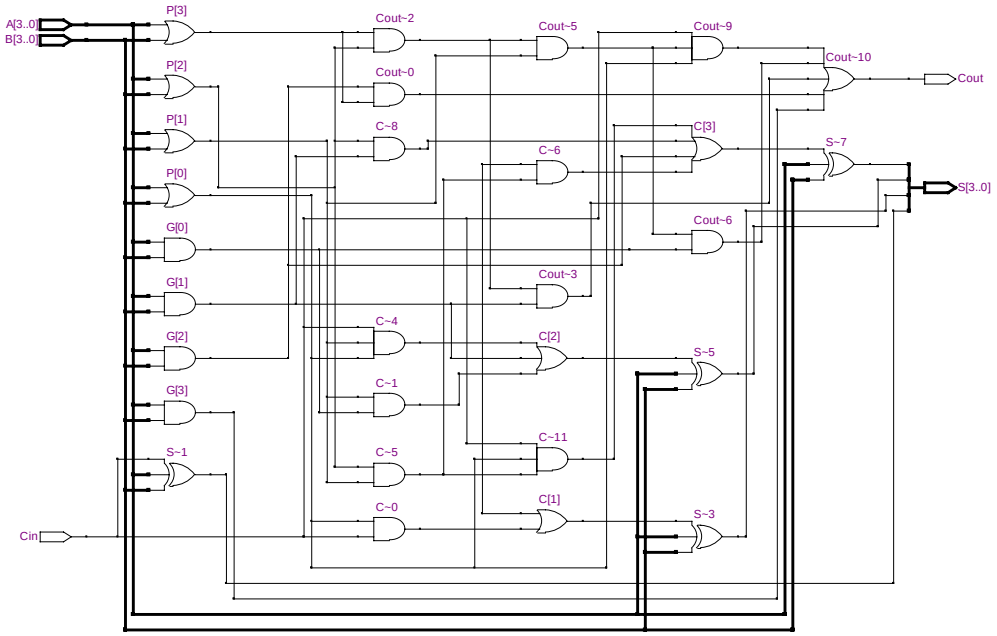
```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
ENTITY counter4d IS
PORT(CLK:IN BIT;
      Q:INOUT BIT_VECTOR(3 DOWNTO 0));
END counter4d;
ARCHITECTURE struc OF counter4d IS
COMPONENT fftasyn
PORT(T,CLK,RESET:IN BIT;Q,QINV:OUT BIT);
END COMPONENT;
COMPONENT and2bit
PORT(A,B:IN BIT;C:OUT BIT);
END COMPONENT;
SIGNAL Q_INV:BIT_VECTOR(3 DOWNTO 0);
SIGNAL A:BIT_VECTOR(2 DOWNTO 0);
BEGIN
        T0:fftasyn PORT MAP('1',CLK,'0',Q(0),Q_INV(0));
        A0:and2bit PORT MAP('1',Q(0),A(0));
        T1:fftasyn PORT MAP(A(0),CLK,'0',Q(1),Q_INV(1));
        A1:and2bit PORT MAP(A(0),Q(1),A(1));
```

```
        T2:fftasyn PORT MAP(A(1),CLK,'0',Q(2),Q_INV(2));
        A2:and2bit PORT MAP(A(1),Q(2),A(2));
        T3:fftasyn PORT MAP(A(2),CLK,'0',Q(3),Q_INV(3));
END struc;
```

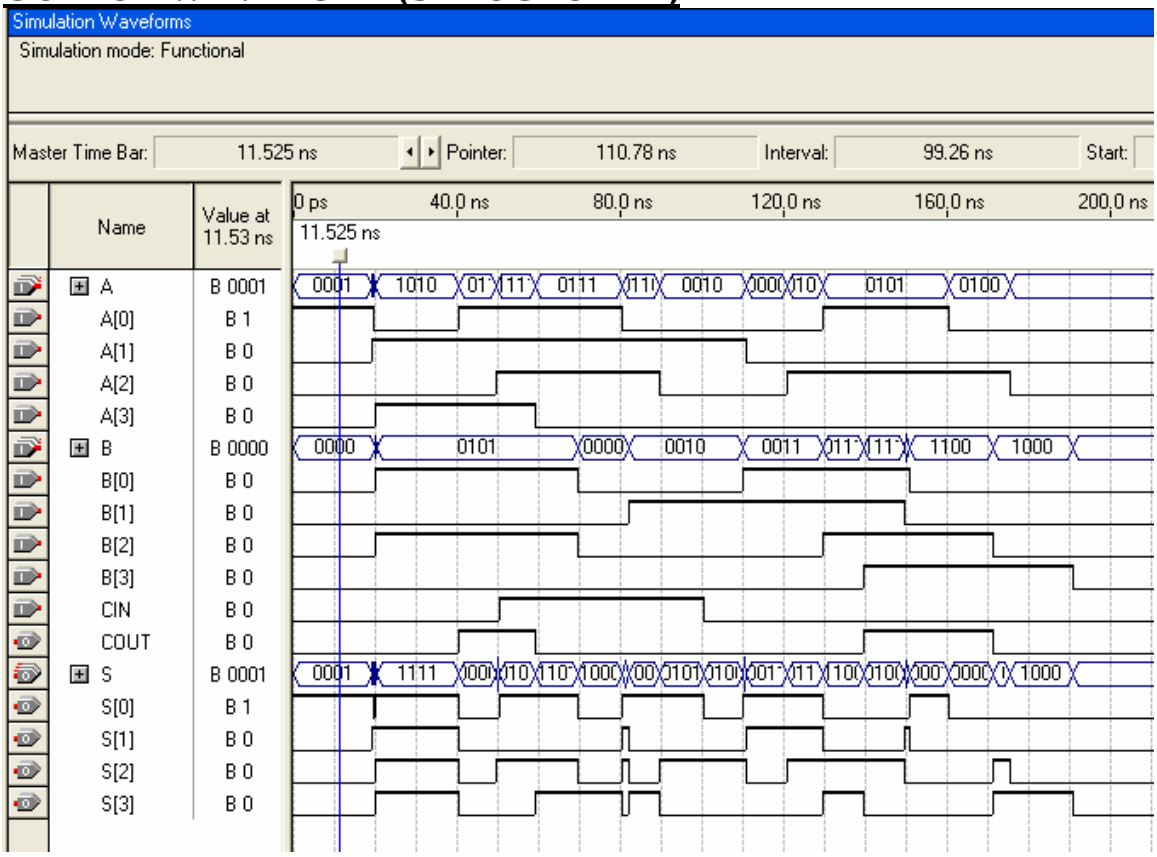
RTL Viewer(STRUCTURAL):



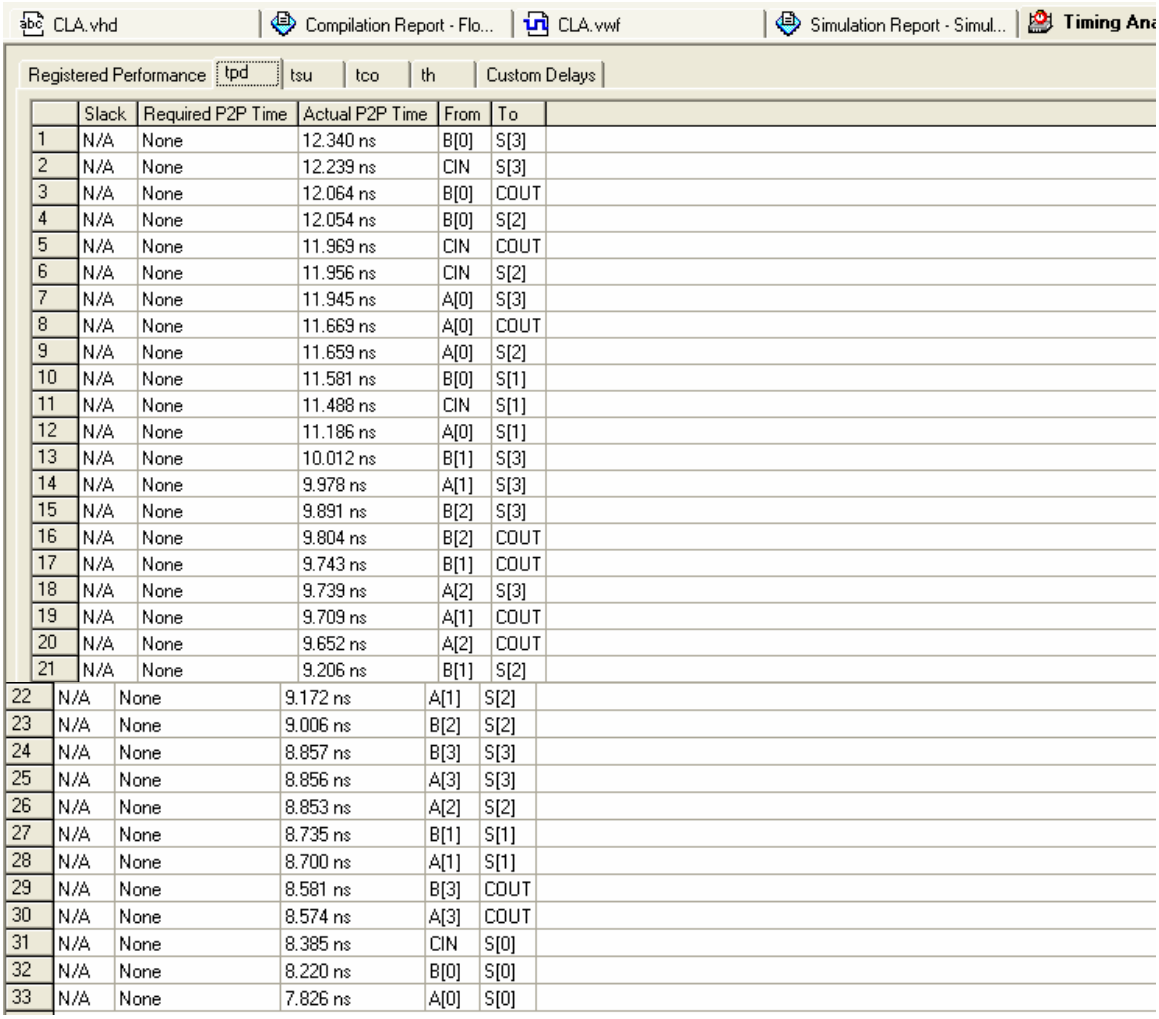
RTL Viewer(Dataflow):



OUTPUT WAVEFORM(STRUCTURAL)

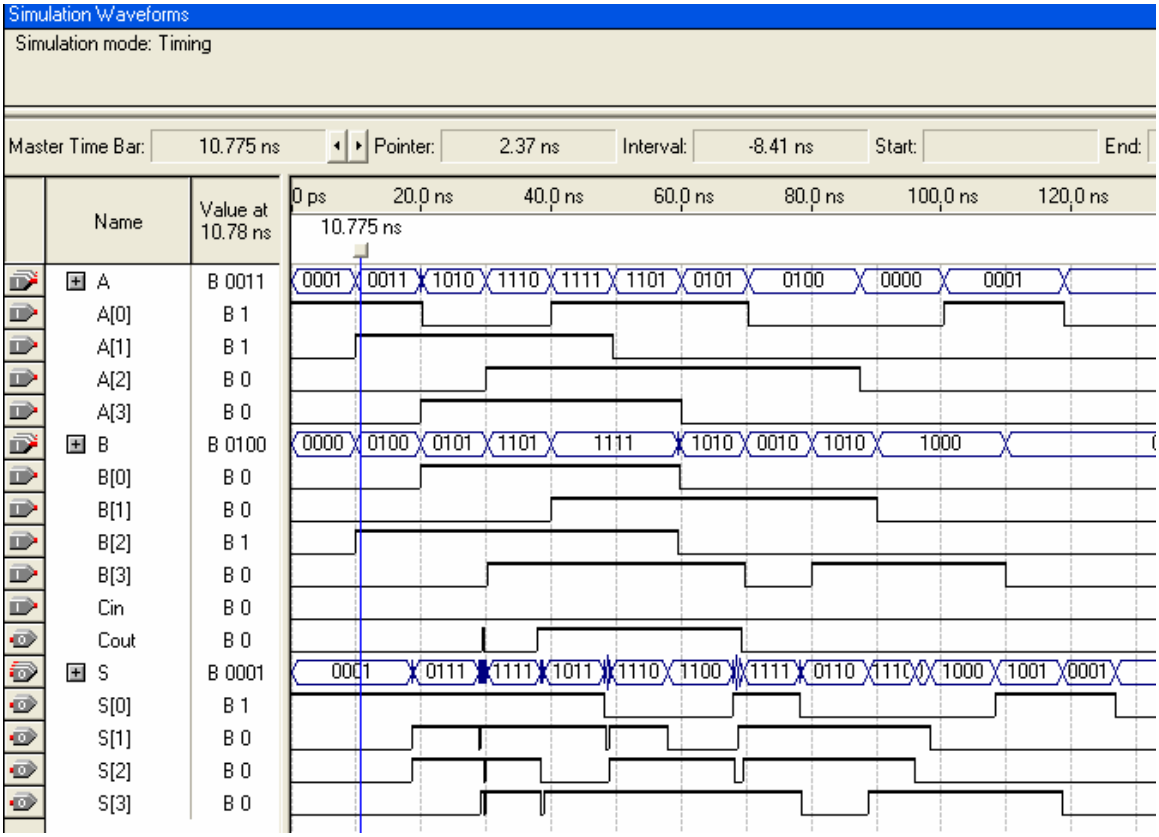


TIMING ANALYSIS(STRUCTURAL):



	Slack	Required P2P Time	Actual P2P Time	From	To
1	N/A	None	12.340 ns	B[0]	S[3]
2	N/A	None	12.239 ns	CIN	S[3]
3	N/A	None	12.064 ns	B[0]	COUT
4	N/A	None	12.054 ns	B[0]	S[2]
5	N/A	None	11.969 ns	CIN	COUT
6	N/A	None	11.956 ns	CIN	S[2]
7	N/A	None	11.945 ns	A[0]	S[3]
8	N/A	None	11.669 ns	A[0]	COUT
9	N/A	None	11.659 ns	A[0]	S[2]
10	N/A	None	11.581 ns	B[0]	S[1]
11	N/A	None	11.488 ns	CIN	S[1]
12	N/A	None	11.186 ns	A[0]	S[1]
13	N/A	None	10.012 ns	B[1]	S[3]
14	N/A	None	9.978 ns	A[1]	S[3]
15	N/A	None	9.891 ns	B[2]	S[3]
16	N/A	None	9.804 ns	B[2]	COUT
17	N/A	None	9.743 ns	B[1]	COUT
18	N/A	None	9.739 ns	A[2]	S[3]
19	N/A	None	9.709 ns	A[1]	COUT
20	N/A	None	9.652 ns	A[2]	COUT
21	N/A	None	9.206 ns	B[1]	S[2]
22	N/A	None	9.172 ns	A[1]	S[2]
23	N/A	None	9.006 ns	B[2]	S[2]
24	N/A	None	8.857 ns	B[3]	S[3]
25	N/A	None	8.856 ns	A[3]	S[3]
26	N/A	None	8.853 ns	A[2]	S[2]
27	N/A	None	8.735 ns	B[1]	S[1]
28	N/A	None	8.700 ns	A[1]	S[1]
29	N/A	None	8.581 ns	B[3]	COUT
30	N/A	None	8.574 ns	A[3]	COUT
31	N/A	None	8.385 ns	CIN	S[0]
32	N/A	None	8.220 ns	B[0]	S[0]
33	N/A	None	7.826 ns	A[0]	S[0]

OUTPUT WAVEFORM(DATAFLOW):



TIMING ANALYSIS:

Registered Performance | tpd | tsu | tco | th | Custom Delays

	Slack	Required P2P Time	Actual P2P Time	From	To
1	N/A	None	9.890 ns	Cin	S[2]
2	N/A	None	9.841 ns	Cin	S[3]
3	N/A	None	9.714 ns	A[0]	S[2]
4	N/A	None	9.691 ns	Cin	Cout
5	N/A	None	9.670 ns	A[0]	S[3]
6	N/A	None	9.644 ns	B[0]	S[2]
7	N/A	None	9.601 ns	B[0]	S[3]
8	N/A	None	9.520 ns	A[0]	Cout
9	N/A	None	9.451 ns	B[0]	Cout
10	N/A	None	9.314 ns	A[3]	S[3]
11	N/A	None	9.283 ns	A[1]	S[3]
12	N/A	None	9.177 ns	Cin	S[1]
13	N/A	None	9.141 ns	A[1]	S[2]
14	N/A	None	9.133 ns	A[1]	Cout
15	N/A	None	9.013 ns	B[1]	S[3]
16	N/A	None	9.006 ns	A[0]	S[1]
17	N/A	None	8.937 ns	B[0]	S[1]
18	N/A	None	8.871 ns	B[1]	S[2]
19	N/A	None	8.863 ns	B[1]	Cout
20	N/A	None	8.636 ns	B[3]	S[3]
21	N/A	None	8.618 ns	A[1]	S[1]

22	N/A	None	8.534 ns	B[2]	S[2]
23	N/A	None	8.483 ns	B[2]	S[3]
24	N/A	None	8.432 ns	A[3]	Cout
25	N/A	None	8.383 ns	A[2]	S[2]
26	N/A	None	8.349 ns	B[1]	S[1]
27	N/A	None	8.340 ns	A[2]	S[3]
28	N/A	None	8.334 ns	B[2]	Cout
29	N/A	None	8.222 ns	Cin	S[0]
30	N/A	None	8.183 ns	A[2]	Cout
31	N/A	None	8.163 ns	B[0]	S[0]
32	N/A	None	8.029 ns	A[0]	S[0]
33	N/A	None	7.751 ns	B[3]	Cout

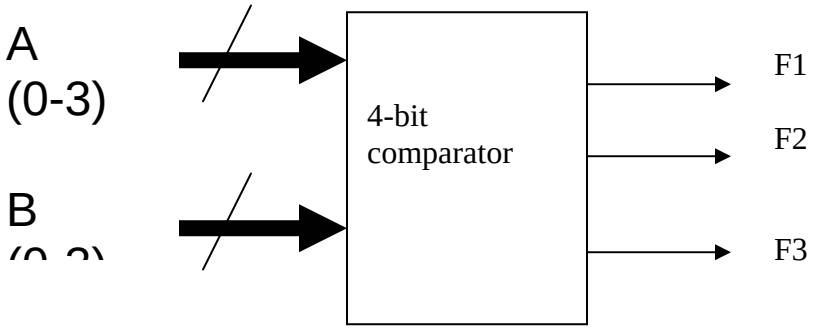
Conclusions:

We find that when 2 4-bit no.s are given as input the output is a 4-bit sum vector and a carry out.The delay in the generation of carry is 5 gates delay and is faster than ripple carry adder.

4-bit comparator

Aim: To design a 4-bit comparator(behavioral and structural model)

ENTITY:

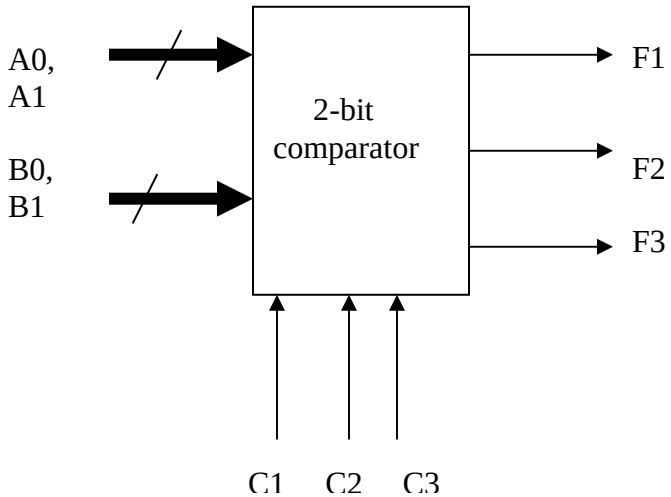


DESCRIPTION:

Comparison between A and B:
F1=1 if A>B
F2=1 if A=B
F3=1 if A<B

BASIC COMPONENT:

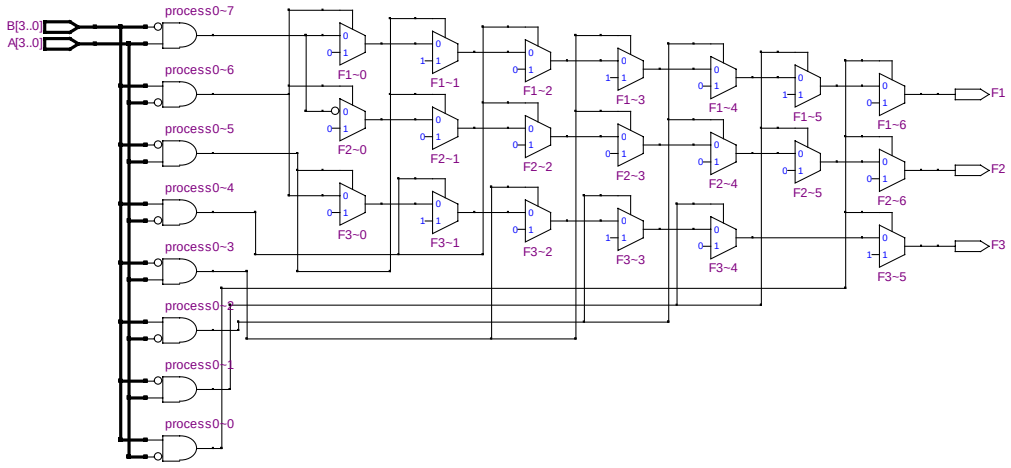
2-bit comparator:



VHDL Code: Behavioral model:

```
LIBRARY IEEE;//standard library
USE IEEE.STD_LOGIC_1164.ALL;//importing the library.

//entity declaration.
ENTITY comp4b IS
PORT(A,B:IN  BIT_VECTOR(3 DOWNTO 0);
        F1,F2,F3:OUT BIT);//inputs and outputs.
END ENTITY; //end of entity declaration.

ARCHITECTURE behav OF comp4b IS
BEGIN
PROCESS(A,B)//sensitivity list.
BEGIN
IF(A(3)='0' AND B(3)='1')THEN F3<='1';F2<='0';F1<='0';
ELSIF(A(3)='1' AND B(3)='0')THEN F1<='1';F2<='0';F3<='0';
ELSE
IF(A(2)='0' AND B(2)='1')THEN F3<='1';F2<='0';F1<='0';
ELSIF(A(2)='1' AND B(2)='0')THEN F1<='1';F2<='0';F3<='0';
ELSE
IF(A(1)='0' AND B(1)='1')THEN F3<='1';F2<='0';F1<='0';
ELSIF(A(1)='1' AND B(1)='0')THEN F1<='1';F2<='0';F3<='0';
ELSE
IF(A(0)='0' AND B(0)='1')THEN F3<='1';F2<='0';F1<='0';
ELSIF(A(0)='1' AND B(0)='0')THEN F1<='1';F2<='0';F3<='0';
ELSE
 F2<='1';F1<='0';F3<='0';
END IF;
END IF;
END IF;
END IF;
END PROCESS;
END behav;//end of architecture.
```

RTL viewer for behavioral.....



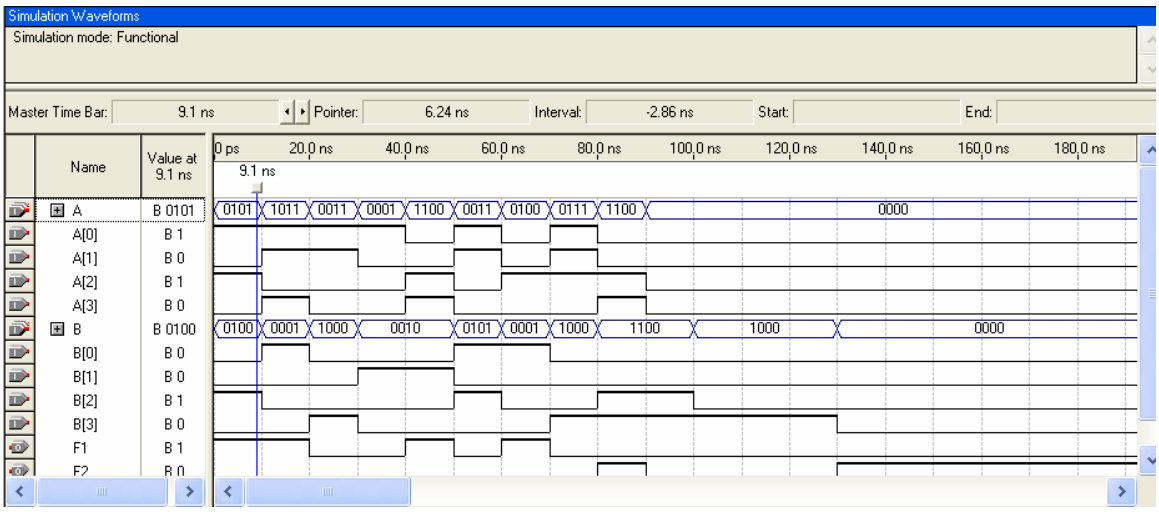
Timing analysis…..

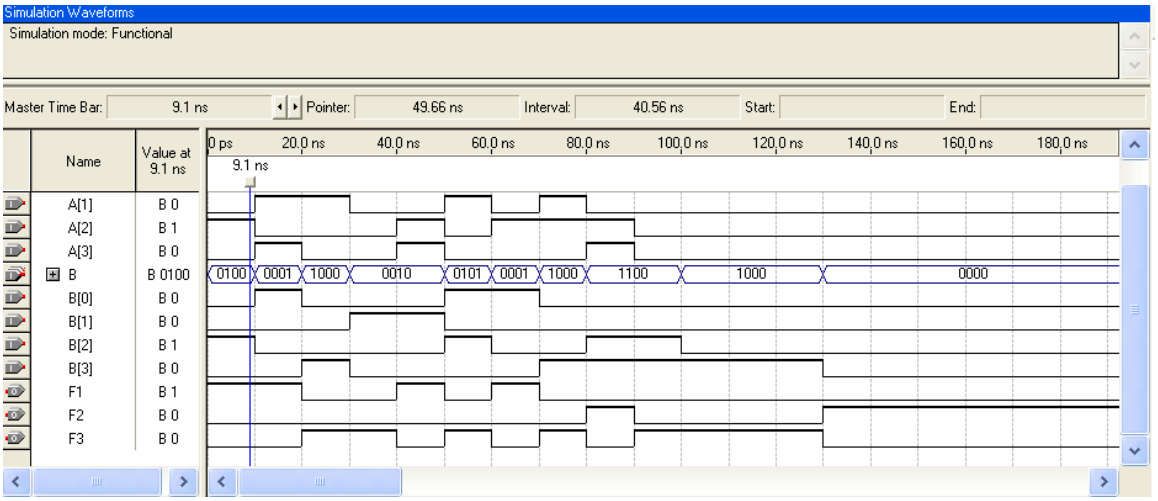
Timing Analyzer Tool — comp4b.vhd

Registered Performance | tpd | tsu | tco | th | Custom Delays

	Slack	Required P2P Time	Actual P2P Time	From	To
1	N/A	None	10.448 ns	A[3]	F2
2	N/A	None	10.144 ns	B[2]	F2
3	N/A	None	9.924 ns	B[3]	F2
4	N/A	None	9.810 ns	A[2]	F2
5	N/A	None	9.743 ns	A[3]	F3
6	N/A	None	9.736 ns	A[3]	F1
7	N/A	None	9.685 ns	A[0]	F2
8	N/A	None	9.469 ns	A[0]	F1
9	N/A	None	9.448 ns	B[1]	F2
10	N/A	None	9.437 ns	B[2]	F3
11	N/A	None	9.433 ns	B[2]	F1
12	N/A	None	9.424 ns	A[1]	F2
13	N/A	None	9.320 ns	B[1]	F1
14	N/A	None	9.313 ns	B[0]	F2

Output waveform…..





VHDL structural model:

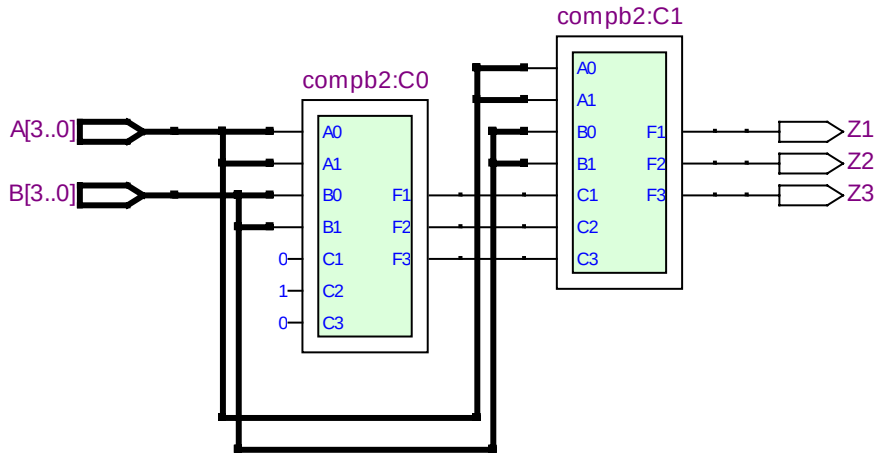
LIBRARY IEEE; ;//*standard library*
USE IEEE.STD_LOGIC_1164.ALL; //*importing the library*.

//*entity declaration.*
ENTITY comp4s IS
PORT(A,B:IN BIT_VECTOR(3 DOWNTO 0);
Z1,Z2,Z3:OUT BIT) ;
END ENTITY; //*end of entity declaration.*

ARCHITECTURE struc OF comp4s IS
//component declaration of two bit comparator.
COMPONENT compb2
PORT(A1,A0,B1,B0:IN BIT;
C1,C2,C3:IN BIT;F1,F2,F3:OUT BIT);
END COMPONENT;*//end of component declaration.*

SIGNAL X:BIT_VECTOR(3 DOWNTO 1);*//signal declaration.*
BEGIN
C0:compb2 PORT MAP(A(3),A(2),B(3),B(2),'0','1','0',X(1),X(2),X(3));
C1:compb2 PORT MAP(A(1),A(0),B(1),B(0),X(1),X(2),X(3),Z1,Z2,Z3);
END struc;*//end of architecture.*

RTL viewer:



Timing analysis:

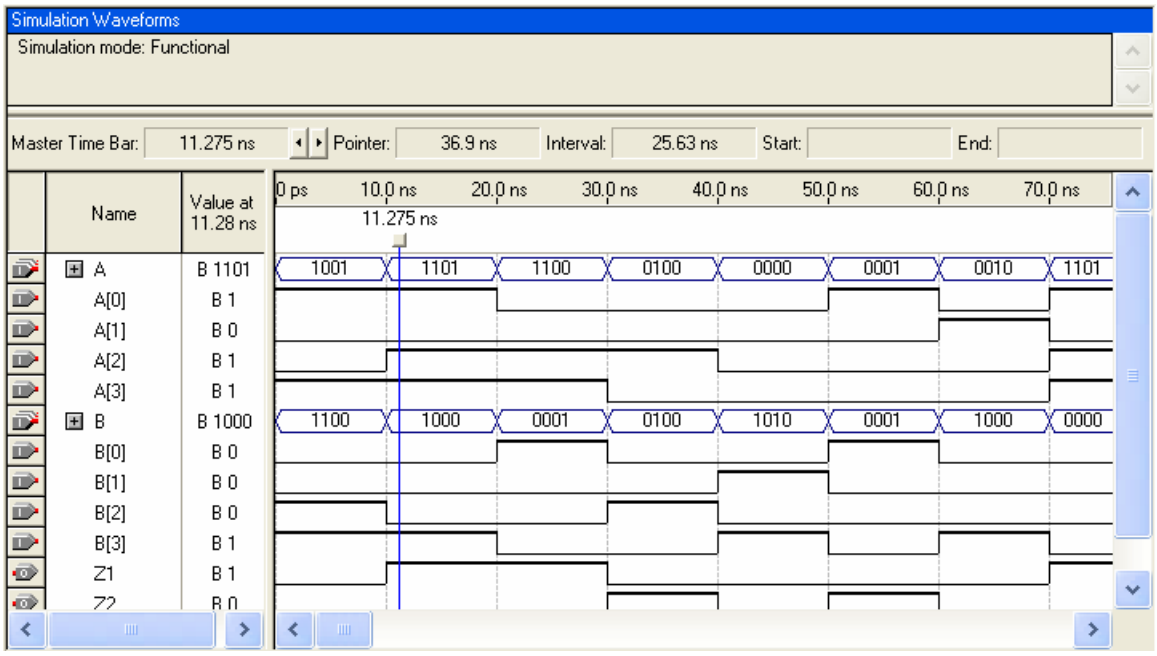
Registered Performance | tpd | tsu | tco | th | Custom Delays

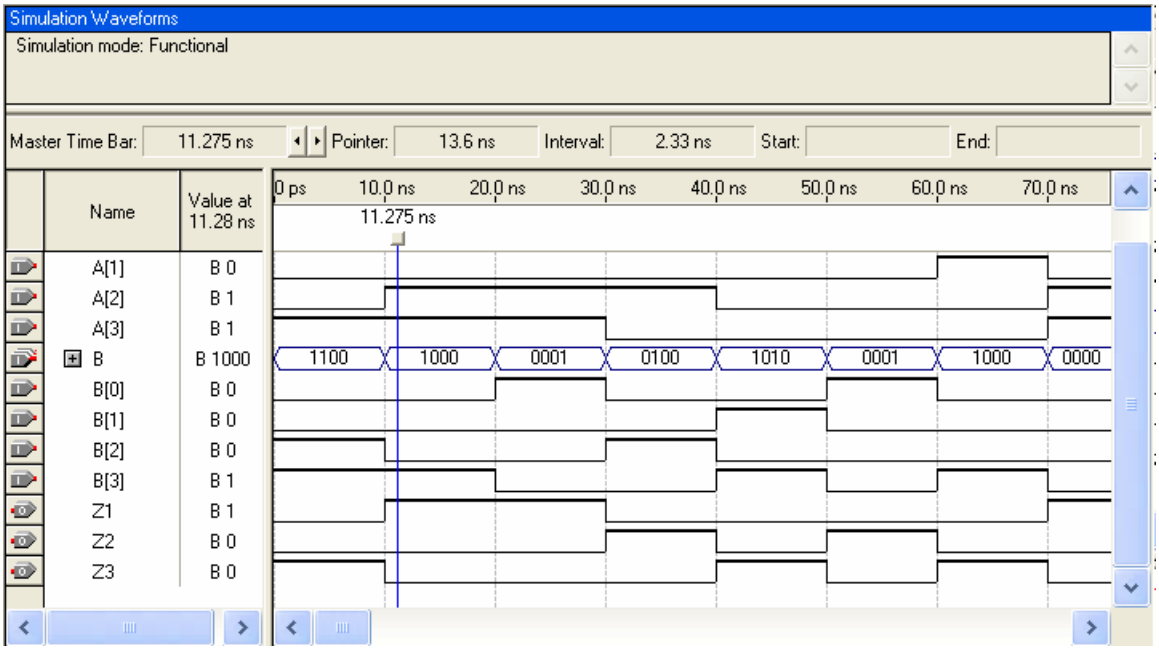
	Slack	Required P2P Time	Actual P2P Time	From	To
1	N/A	None	9.756 ns	B[0]	Z3
2	N/A	None	9.662 ns	B[1]	Z3
3	N/A	None	9.651 ns	A[0]	Z3
4	N/A	None	9.495 ns	A[1]	Z3
5	N/A	None	9.470 ns	A[2]	Z2
6	N/A	None	9.321 ns	B[3]	Z2
7	N/A	None	9.295 ns	B[0]	Z1
8	N/A	None	9.229 ns	A[2]	Z1
9	N/A	None	9.194 ns	B[1]	Z1
10	N/A	None	9.183 ns	A[0]	Z1
11	N/A	None	9.145 ns	A[2]	Z3
12	N/A	None	9.080 ns	B[3]	Z1
13	N/A	None	9.032 ns	A[1]	Z1
14	N/A	None	9.006 ns	B[2]	Z2

Registered Performance | tpd | tsu | tco | th | Custom Delays

	Slack	Required P2P Time	Actual P2P Time	From	To
12	N/A	None	9.080 ns	B[3]	Z1
13	N/A	None	9.032 ns	A[1]	Z1
14	N/A	None	9.006 ns	B[2]	Z2
15	N/A	None	9.002 ns	B[1]	Z2
16	N/A	None	8.840 ns	A[1]	Z2
17	N/A	None	8.803 ns	A[3]	Z2
18	N/A	None	8.765 ns	B[2]	Z1
19	N/A	None	8.717 ns	B[0]	Z2
20	N/A	None	8.661 ns	B[3]	Z3
21	N/A	None	8.606 ns	A[0]	Z2
22	N/A	None	8.583 ns	B[2]	Z3
23	N/A	None	8.562 ns	A[3]	Z1
24	N/A	None	8.065 ns	A[3]	Z3

Output waveform:





CONCLUSION:

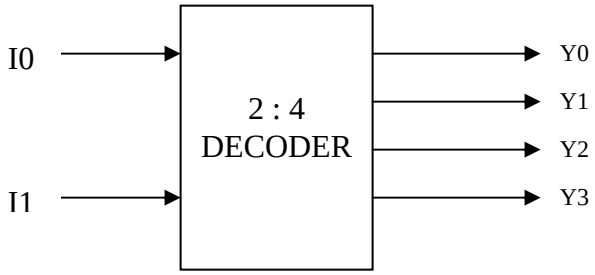
-4-bit comparator is realized using 2-bit comparator in structural method and fully synthesized in behavioral model.

-The model is verified for different combination of inputs as shown in the output waveform.

2:4 DECODER

AIM: To develop a VHDL code for a 2:4 decoder.

ENTITY:



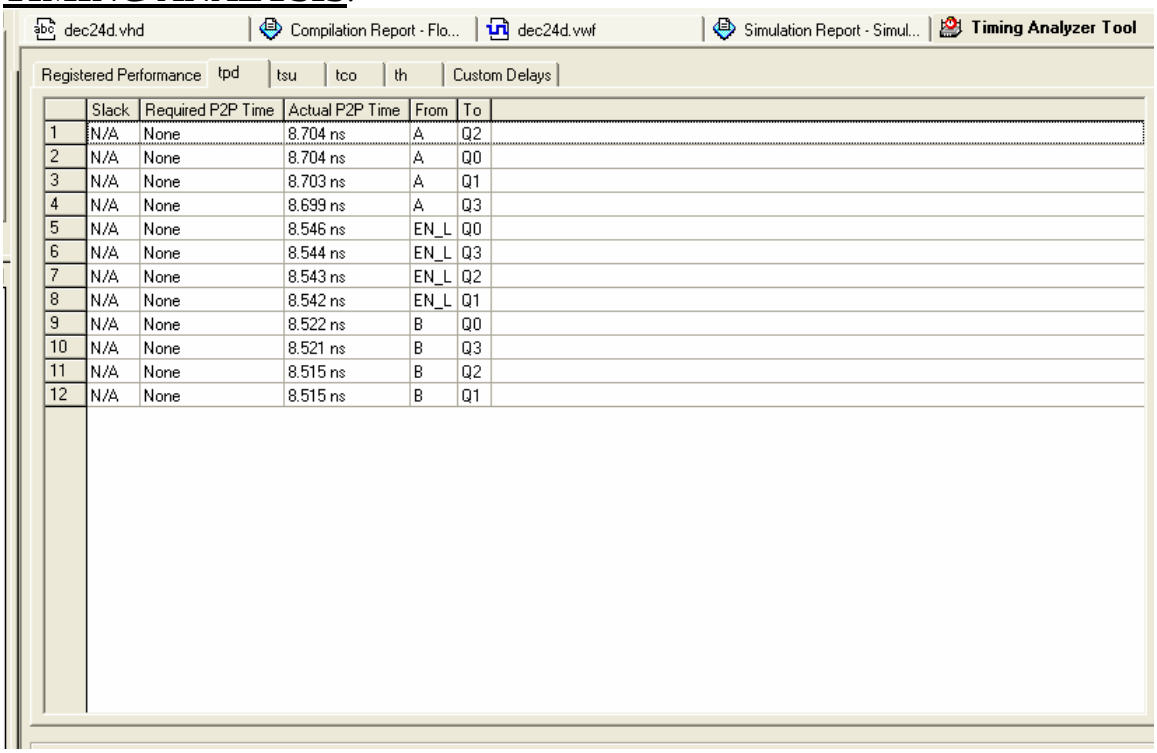
TRUTH TABLE

I1	I0	OUTPUT
0	0	Y0
0	1	Y1
1	0	Y2
1	1	Y3

VHDL code

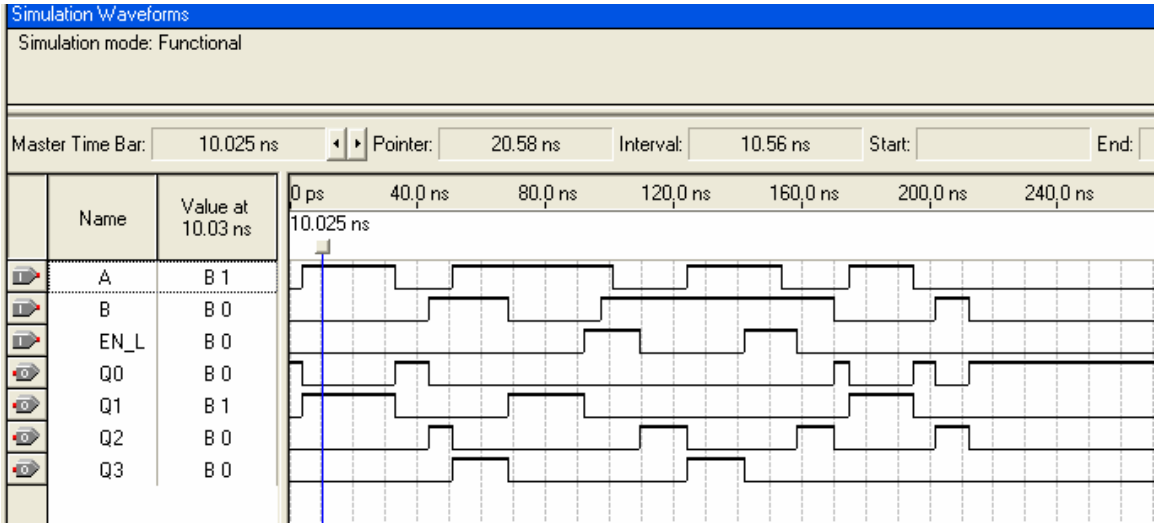
```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
ENTITY dec24d IS
PORT(A,B,EN_L:IN BIT;
        Q0,Q1,Q2,Q3:OUT BIT);
        END ENTITY;

ARCHITECTURE dataflow OF dec24d IS
BEGIN
Q0<=(NOT A)AND (NOT B) AND (NOT EN_L);
Q1<=( A)AND (NOT B) AND (NOT EN_L);
Q2<=(NOT A)AND (B) AND (NOT EN_L);
Q3<=(A)AND (B) AND (NOT EN_L);
END dataflow;
```

TIMING ANALYSIS:

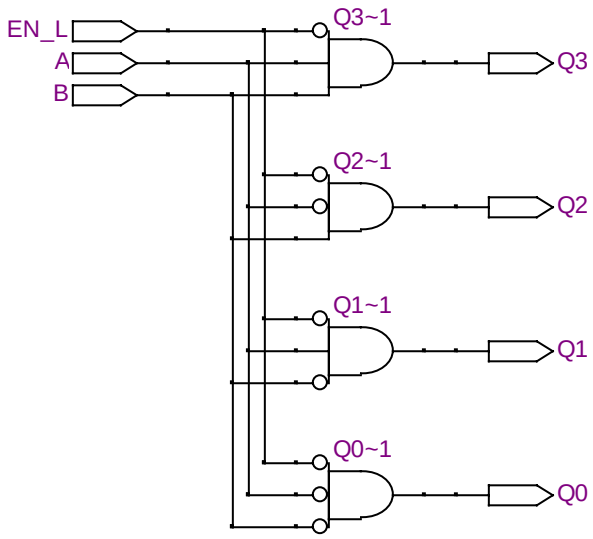


	Slack	Required P2P Time	Actual P2P Time	From	To
1	N/A	None	8.704 ns	A	Q2
2	N/A	None	8.704 ns	A	Q0
3	N/A	None	8.703 ns	A	Q1
4	N/A	None	8.699 ns	A	Q3
5	N/A	None	8.546 ns	EN_L	Q0
6	N/A	None	8.544 ns	EN_L	Q3
7	N/A	None	8.543 ns	EN_L	Q2
8	N/A	None	8.542 ns	EN_L	Q1
9	N/A	None	8.522 ns	B	Q0
10	N/A	None	8.521 ns	B	Q3
11	N/A	None	8.515 ns	B	Q2
12	N/A	None	8.515 ns	B	Q1

OUTPUT WAVEFORMS:



RTL Viewer:



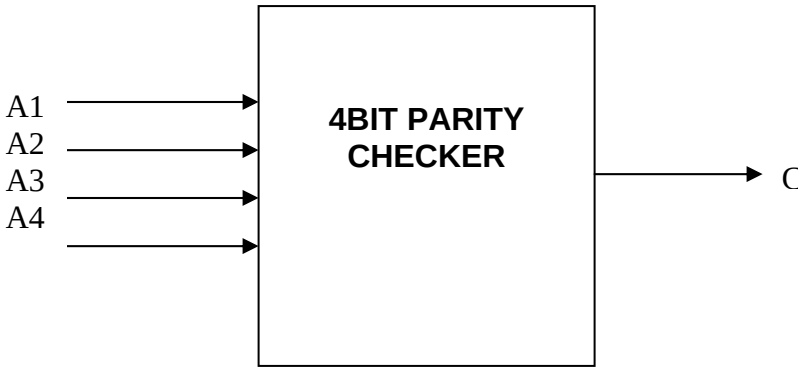
CONCLUSIONS:

-2:4 decoder is realized in dataflow model of architecture and the output waveform
-timing analysis is also obtained.

4 BIT PARITYCHECKER

AIM: To design a 4 bit parity checker.

ENTITY:



TRUTH TABLE FOR PARITY CHECKER:

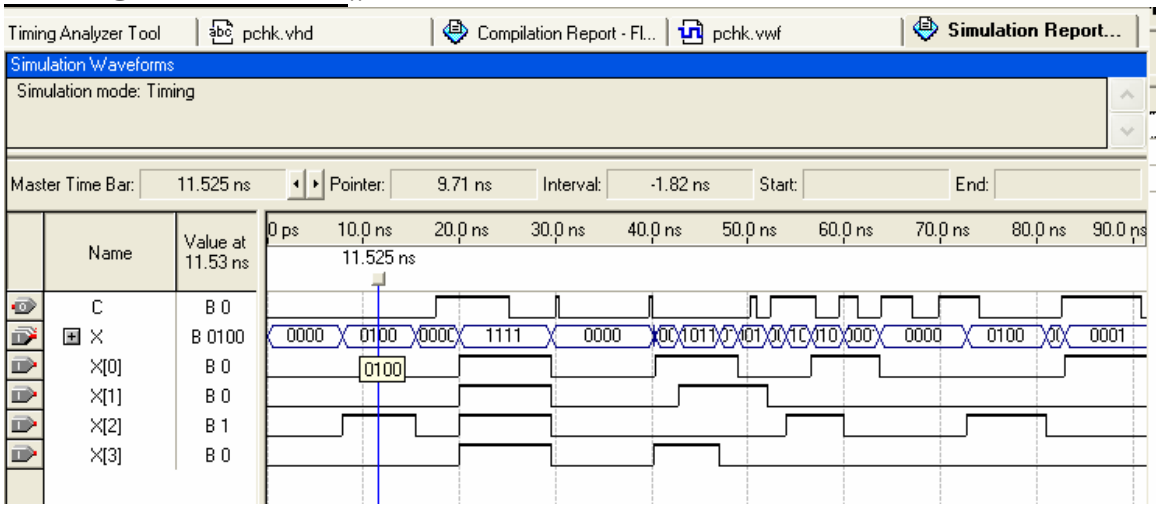
A1	A2	A3	A4	C
0	0	0	0	1
0	0	0	1	0
0	0	1	0	0
0	0	1	1	1
0	1	0	0	0
0	1	0	1	1
0	1	1	0	1
0	1	1	1	0
1	0	0	0	0
1	0	0	1	1
1	0	1	0	1
1	0	1	1	0
1	1	0	0	1
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

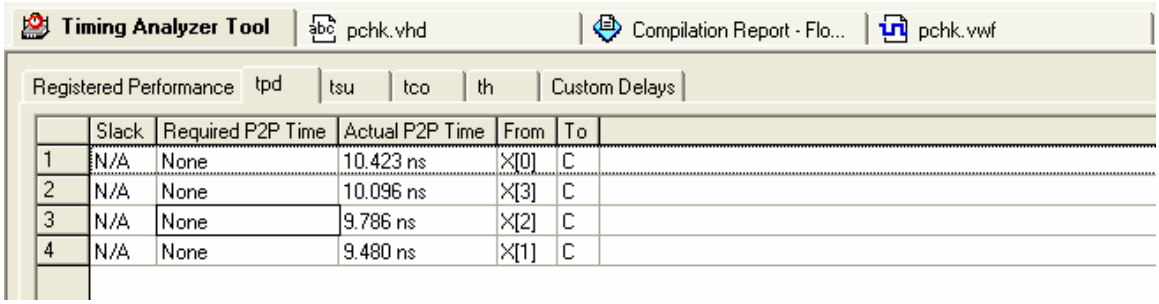
VHDL CODE:

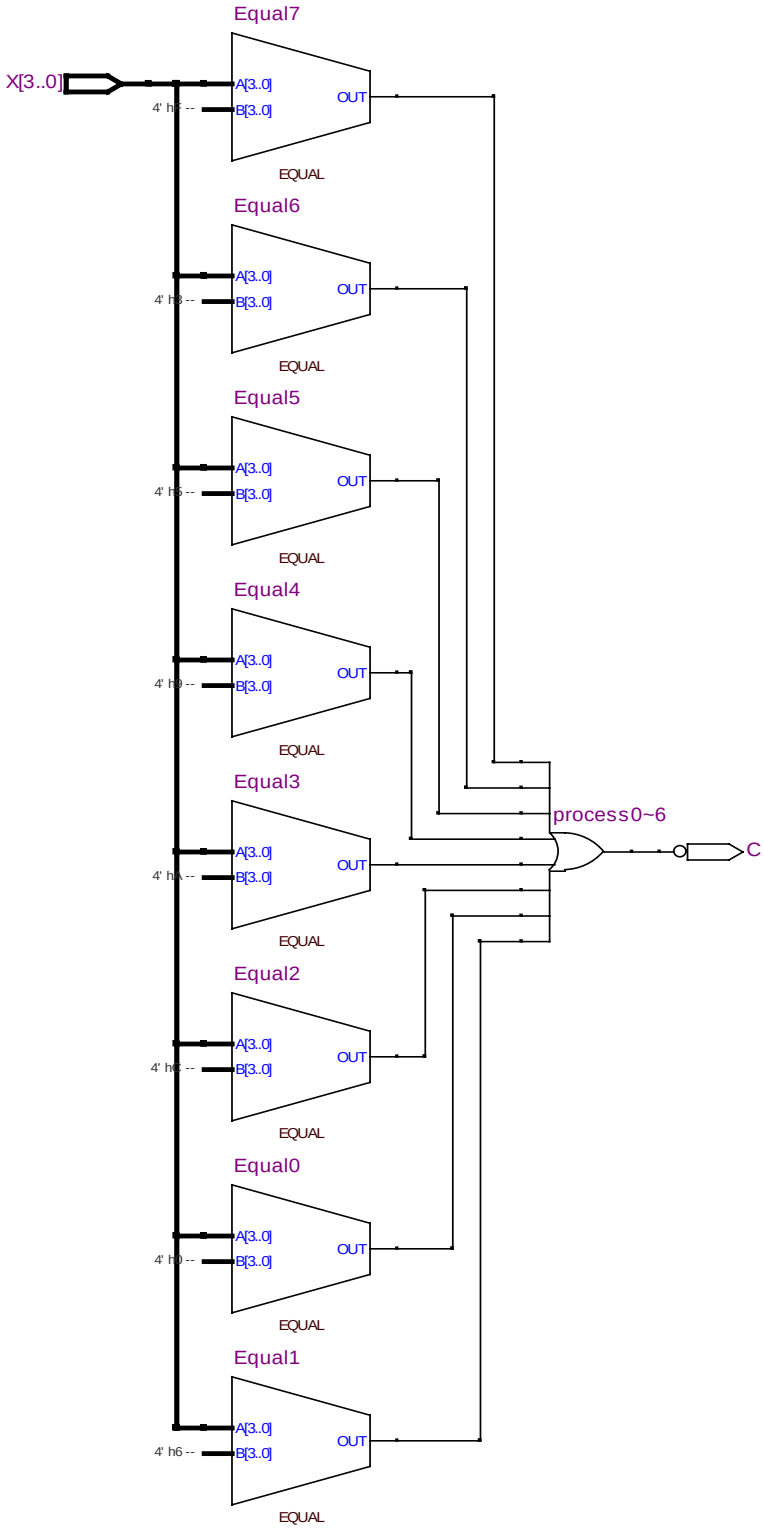
```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;//Standard packages available
ENTITY paritycheck IS
PORT(X:IN BIT_VECTOR(3 DOWNTO 0);
C:OUT BIT);//Input and output declaration
END paritycheck;
```

```
ARCHITECTURE behav OF paritycheck IS//Behavioral modelling
BEGIN
PROCESS(X)//Sensitivity list
BEGIN//Begin process
IF(X="0000" OR X="0110" OR X="1100" OR X="1010" OR X="1001" OR X="0101"
OR X="0011" OR X="1111")
THEN C<='0';
ELSE C<='1';//Assignment of  output values.
END IF;//Syntax for ending if loop
END PROCESS;
END behav;
```

TIMING ANALYZER..







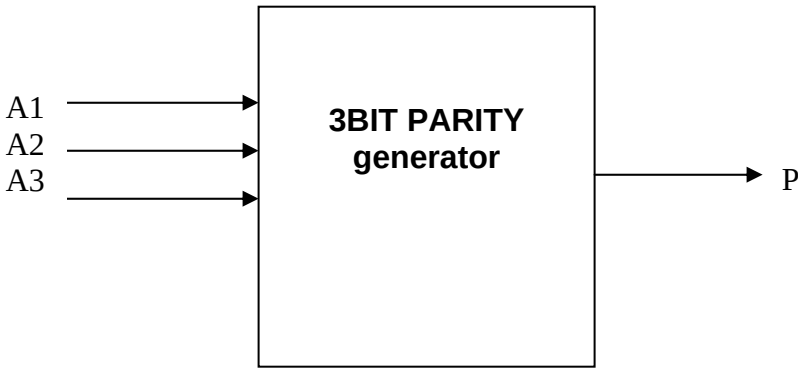
CONCLUSIONS:

We find that when there are odd number of ones in a given number the checker output is one and when there are even number of ones the checker output is zero. This is just a parity checking code that can be used to detect but not correct error.

3-BIT PARITYGENERATOR

AIM: To design a 3 bit parity generator

ENTITY:



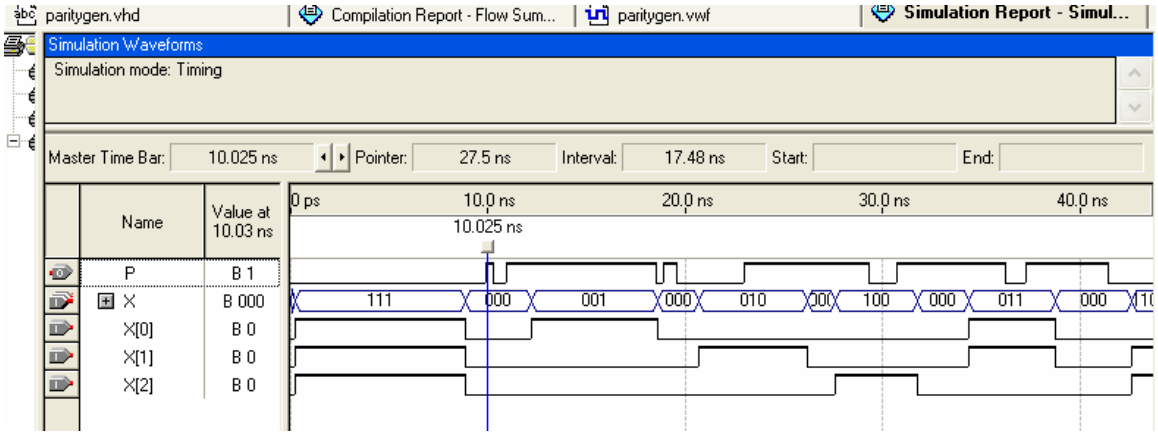
TRUTH TABLE FOR PARITY GENERATOR:

A1	A2	A3	P
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

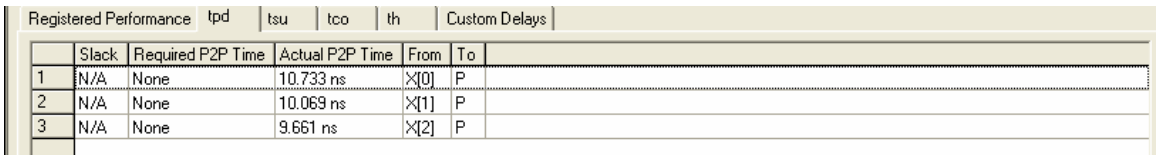
VHDL CODE:

```
LIBRARY IEEE;//standard library.
USE IEEE.STD_LOGIC_1164.ALL;
ENTITY paritygen IS
PORT(X:IN BIT_VECTOR(2 DOWNTO 0);
                                        P:OUT BIT);
END paritygen;
ARCHITECTURE behav OF paritygen IS
BEGIN
PROCESS(X)//sensitivity list
BEGIN
IF(X="000" OR X="011" OR X="110" OR X="101")
THEN P<='0';
ELSE P<='1';
END IF;
END PROCESS;
END behav;
```

Output Waveform



Timing Analysis



	Slack	Required P2P Time	Actual P2P Time	From	To
1	N/A	None	10.733 ns	X[0]	P
2	N/A	None	10.069 ns	X[1]	P
3	N/A	None	9.661 ns	X[2]	P

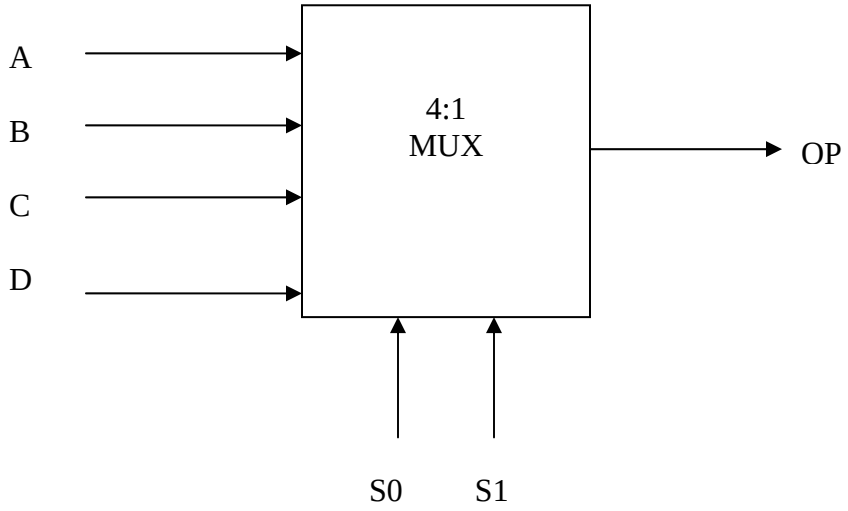
CONCLUSIONS:

We find that when there are odd number of ones in a given number the generator output is one and when there are even number of ones the generator output is zero. This is a parity generating code that can be used to ensure that all numbers have even parity.

4X1 MULTIPLEXER

AIM: To design 4X1 multiplexer.

ENTITY:



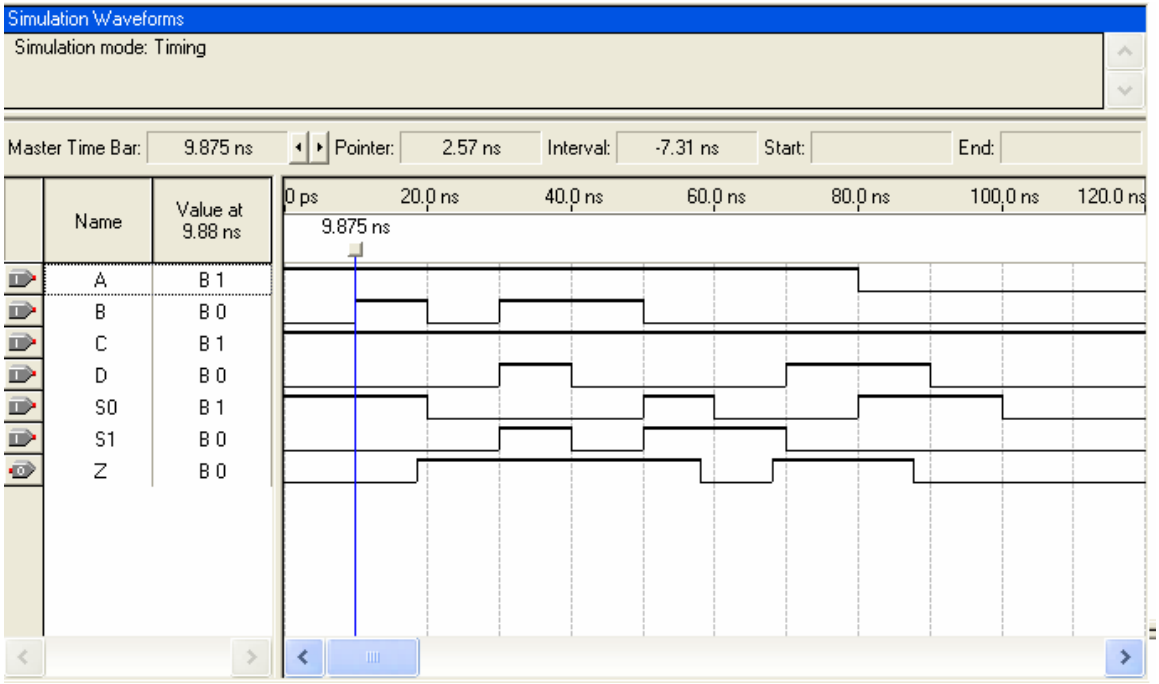
TRUTH TABLE

S1	S0	O/P
0	0	I0
0	1	I1
1	0	I2
1	1	I3

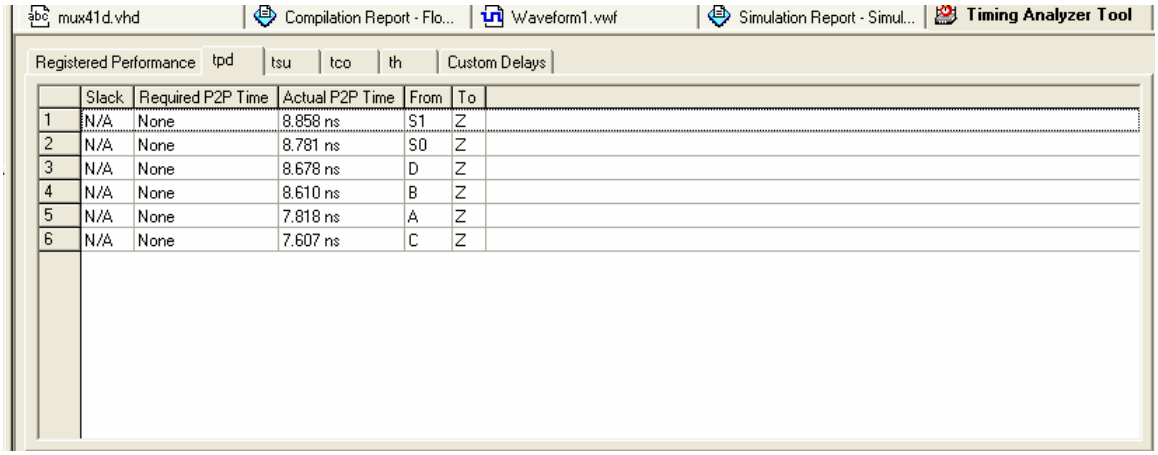
VHDL CODE: (behavioral code)

```
LIBRARY IEEE;//standard library.
USE IEEE.STD_LOGIC_1164.ALL;//importing standard library.
USE IEEE.STD_LOGIC_ARITH.ALL;

//entity declaration
ENTITY 4mux1 IS
PORT(A,B,C,D:IN STD_LOGIC;
        S0,S1: IN STD_LOGIC;
        Q:OUT STD_LOGIC);
END 4mux1;
//end of entity declaration

ARCHITECTURE behave OF 4mux1 IS
BEGIN
        PROCESS(A,B,C,D,S0,S1)//sensitivity list.
                BEGIN
                        IF S0='0' AND S1='0' THEN Q<='A';
                        ELSIF SO='1' AND S1='0' THEN Q<='B';
                        ELSIF SO='0' AND S1='1' THEN Q<='C';
                        ELSE Q<='D';
                        END IF;
        END PROCESS;
END behave;//end of architecture.
```

Output waveform…...

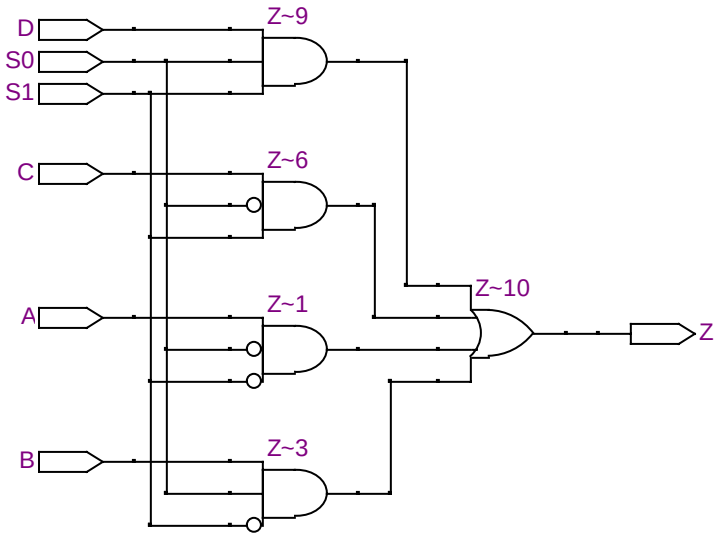


Timing analysis:



	Slack	Required P2P Time	Actual P2P Time	From	To
1	N/A	None	8.858 ns	S1	Z
2	N/A	None	8.781 ns	S0	Z
3	N/A	None	8.678 ns	D	Z
4	N/A	None	8.610 ns	B	Z
5	N/A	None	7.818 ns	A	Z
6	N/A	None	7.607 ns	C	Z

RTL-view:



Conclusions:

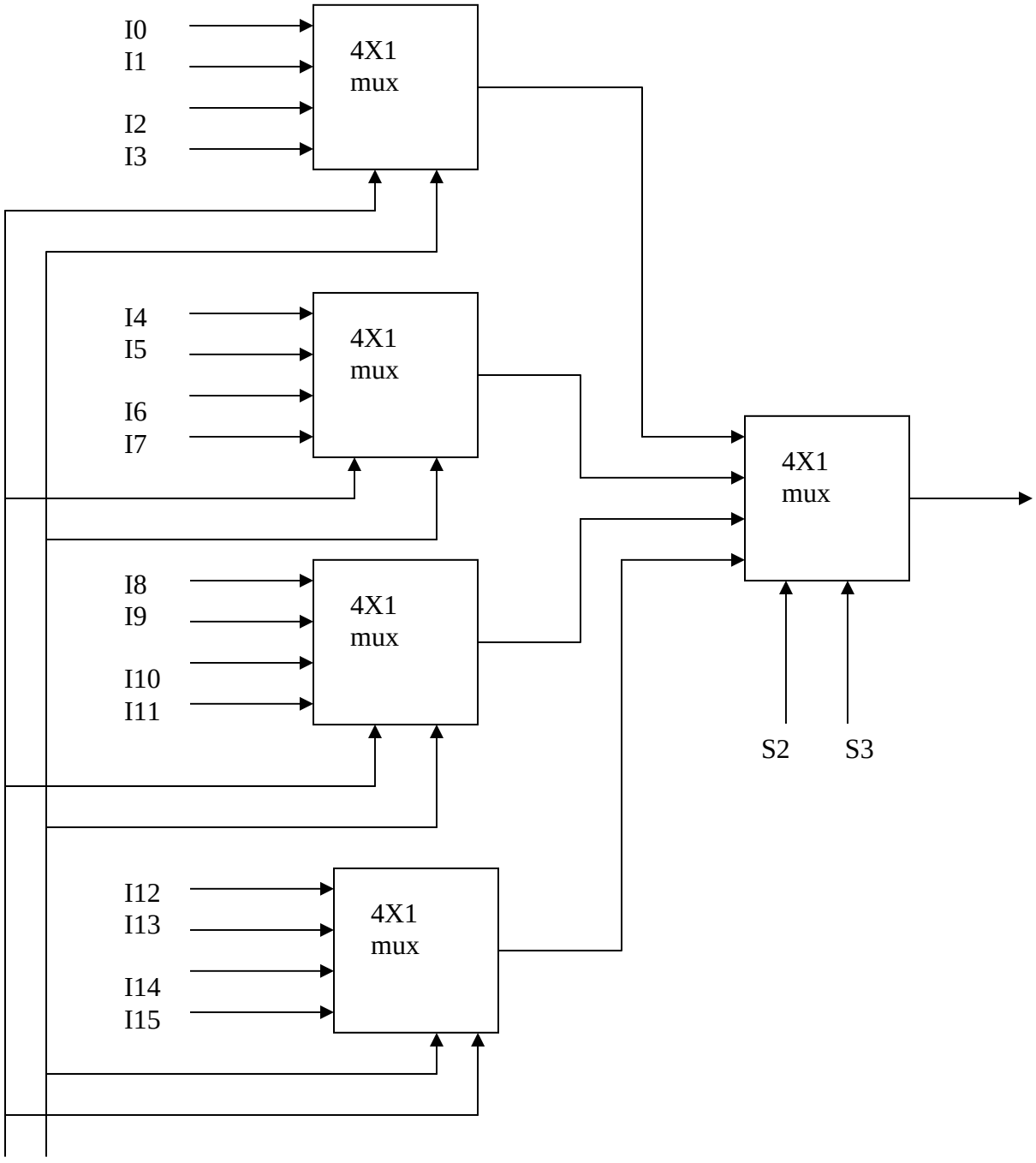
-4X1 multiplexer was realized using behavioral model as the architecture.

- The 4X1 multiplexer verified for various combination of inputs.

16X1 MULTIPLEXER USING 4X1 MULTIPLEXER

AIM: To design 16X1 multiplexer using 4X1 multiplexer.

ENTITY:



S0 S1

BASIC COMPONENT:

4X1 multiplexer:
TRUTH TABLE :
1)BASIC COMPONENT:

S1	S0	O/P
0	0	I0
0	1	I1
1	0	I2
1	1	I3

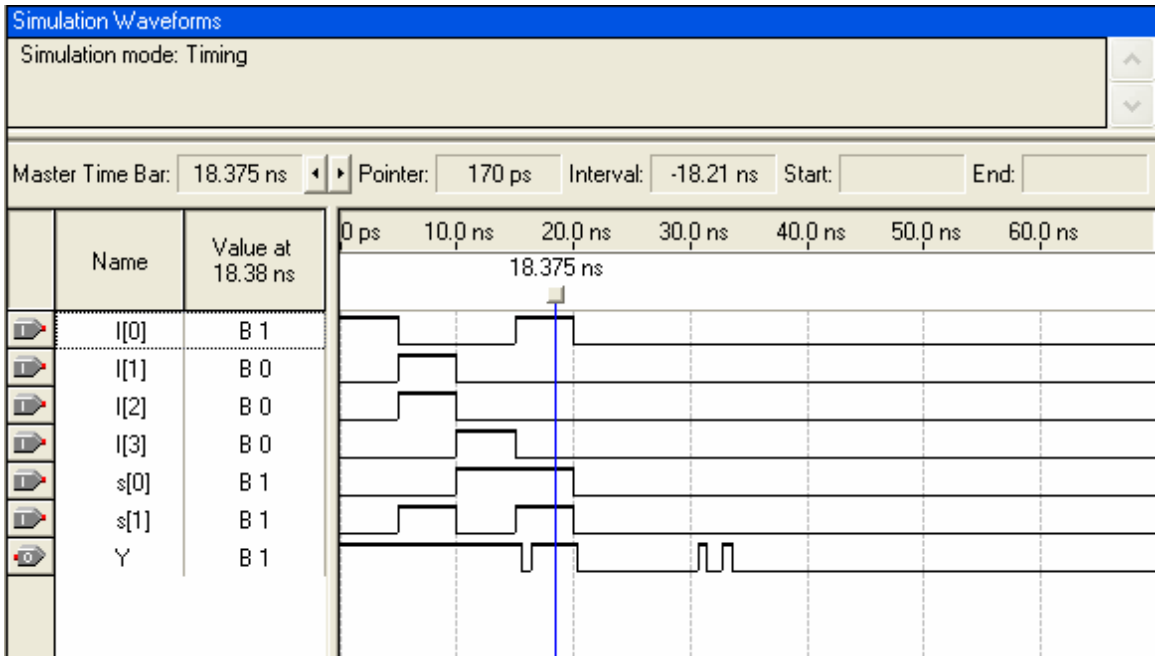
2)ENTITY:

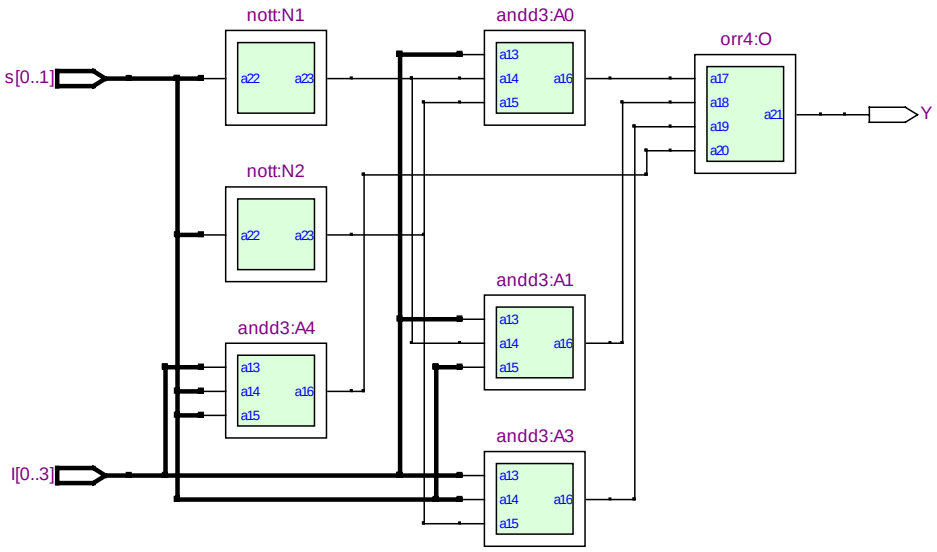
S3	S2	S1	S0	O/P
0	0	0	0	I0
0	0	0	1	I1
0	0	1	0	I2
0	0	1	1	I3
0	1	0	0	I4
0	1	0	1	I5
0	1	1	0	I6
0	1	1	1	I7
1	0	0	0	I8
1	0	0	1	I9
1	0	1	0	I10
1	0	1	1	I11
1	1	0	0	I12
1	1	0	1	I13
1	1	1	0	I14
1	1	1	1	I15

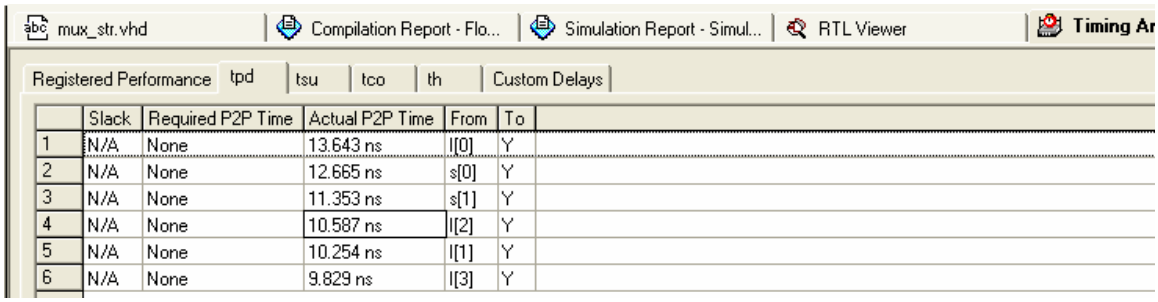
VHDL CODE:

```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
USE IEEE.STD_LOGIC_ARITH.ALL;//Standard library packages
ENTITY MUX161s IS
PORT(A:IN STD_LOGIC_VECTOR(15 DOWNTO 0);
       S: IN STD_LOGIC_VECTOR(3 DOWNTO 0);
       Z:OUT STD_LOGIC);//input and output declaration
END MUX161s;

ARCHITECTURE struc OF MUX161s IS
SIGNAL Z1,Z2,Z3,Z4:STD_LOGIC;
COMPONENT mux41b IS//Basic component
PORT(A,B,C,D,S0,S1:IN STD_LOGIC;
         Q:OUT STD_LOGIC);
END COMPONENT;
BEGIN
```

```
        M1:mux41b PORT MAP(A(0),A(1),A(2),A(3),S(0),S(1),Z1);
        M2:mux41b PORT MAP(A(4),A(5),A(6),A(7),S(0),S(1),Z2);
        M3:mux41b PORT MAP(A(8),A(9),A(10),A(11),S(0),S(1),Z3);
        M4:mux41b PORT MAP(A(12),A(13),A(14),A(15),S(0),S(1),Z4);
        M5:mux41b PORT MAP(Z1,Z2,Z3,Z4,S(2),S(3),Z);//mapping
END struc;
```







	Slack	Required P2P Time	Actual P2P Time	From	To
1	N/A	None	13.643 ns	I[0]	Y
2	N/A	None	12.665 ns	s[0]	Y
3	N/A	None	11.353 ns	s[1]	Y
4	N/A	None	10.587 ns	I[2]	Y
5	N/A	None	10.254 ns	I[1]	Y
6	N/A	None	9.829 ns	I[3]	Y

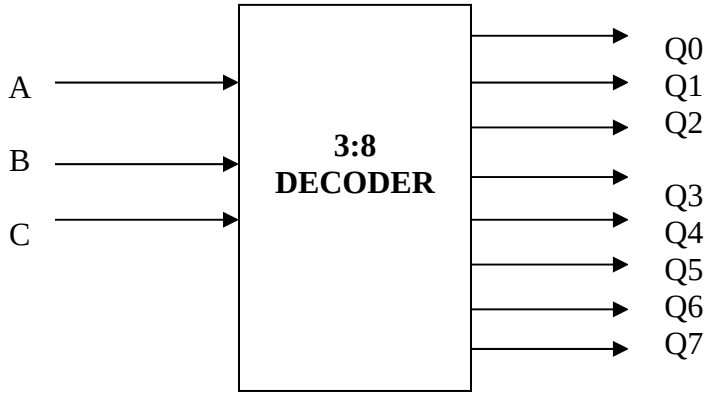
CONCLUSIONS:

We find that we can realize 16:1 mux using four 4:1muxes.The higher order selection lines are used to select one of the outputs from each of the four basic muxes and the lower ones as selection inputs to the basic multiplexers. Thus one of the 16 input lines is selected.

3:8 DECODER USING 2:4 DECODER

Aim: To realize 3:8 decoder using 2:4 decoder.

Entity:



TRUTH TABLE:

3:8 DECODER

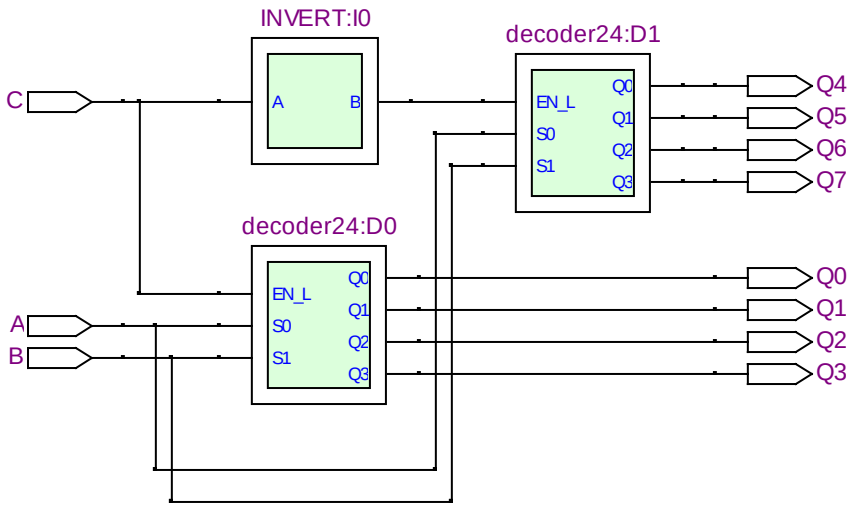
A	B	C	Q0	Q1	Q2	Q3	Q4	Q5	Q6	Q7
0	0	0	1	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	0	0
0	1	1	0	0	0	1	0	0	0	0
1	0	0	0	0	0	0	1	0	0	0
1	0	1	0	0	0	0	0	1	0	0
1	1	0	0	0	0	0	0	0	1	0
1	1	1	0	0	0	0	0	0	0	1

VHDL CODE:

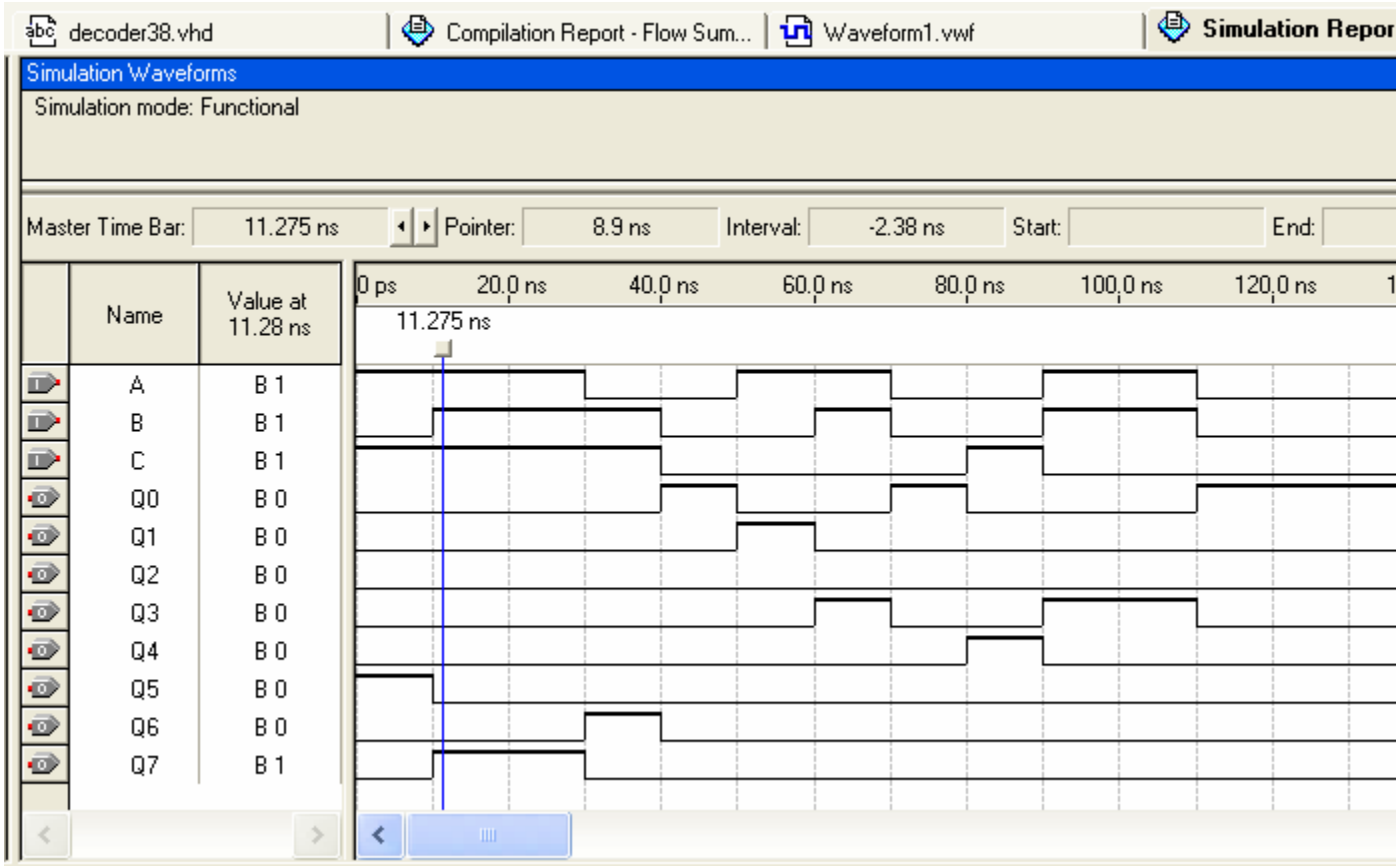
```
LIBRARY IEEE;//standard library
USE IEEE.STD_LOGIC_1164.ALL;//importing the libraray
//entity declaration...
ENTITY decoder38 IS
PORT(A,B,C:IN BIT;
        Q0,Q1,Q2,Q3,Q4,Q5,Q6,Q7:OUT BIT);
END ENTITY;
//end of entity declaration.
```

```
ARCHITECTURE struc OF decoder38 IS

//component declaration..
COMPONENT  decoder24 IS
PORT(S0,S1,EN_L:IN BIT;Q0,Q1,Q2,Q3:OUT BIT);
END COMPONENT;

COMPONENT INVERT
PORT(A:IN BIT;B:OUT BIT);
END COMPONENT;

//signal declaration..
SIGNAL CINV:BIT;

BEGIN
I0:INVERT PORT MAP(C,CINV);
D0:decoder24 PORT MAP(A,B,C,Q0,Q1,Q2,Q3);
D1:decoder24 PORT MAP(A,B,CINV,Q4,Q5,Q6,Q7);
END struc;//
```

RTL View:



Output Waveform:



Timing Analysis:

Registered Performance | tpd | tsu | tco | th | Custom Delays

	Slack	Required P2P Time	Actual P2P Time	From	To
1	N/A	None	9.067 ns	A	Q4
2	N/A	None	9.063 ns	A	Q7
3	N/A	None	9.062 ns	A	Q5
4	N/A	None	9.054 ns	A	Q2
5	N/A	None	8.879 ns	C	Q5
6	N/A	None	8.878 ns	C	Q2
7	N/A	None	8.874 ns	C	Q4
8	N/A	None	8.870 ns	C	Q7
9	N/A	None	8.834 ns	A	Q6
10	N/A	None	8.833 ns	A	Q0
11	N/A	None	8.831 ns	A	Q1
12	N/A	None	8.826 ns	A	Q3
13	N/A	None	8.802 ns	B	Q5
14	N/A	None	8.800 ns	B	Q2
15	N/A	None	8.793 ns	B	Q4
16	N/A	None	8.792 ns	B	Q7
17	N/A	None	8.647 ns	C	Q3
18	N/A	None	8.645 ns	C	Q1
19	N/A	None	8.644 ns	C	Q6
20	N/A	None	8.640 ns	C	Q0
21	N/A	None	8.569 ns	B	Q3
22	N/A	None	8.567 ns	B	Q1
23	N/A	None	8.566 ns	B	Q6
24	N/A	None	8.559 ns	B	Q0

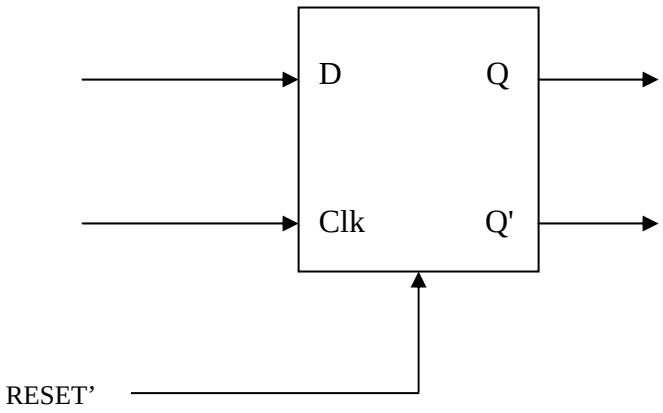
Conclusion:

-we can realize higher order decoder using lower order decoder.
-The decoder truth table was verified by giving different inputs.

D FLIPFLOP

AIM: To develop a VHDL code for D flipflop.

ENTITY:



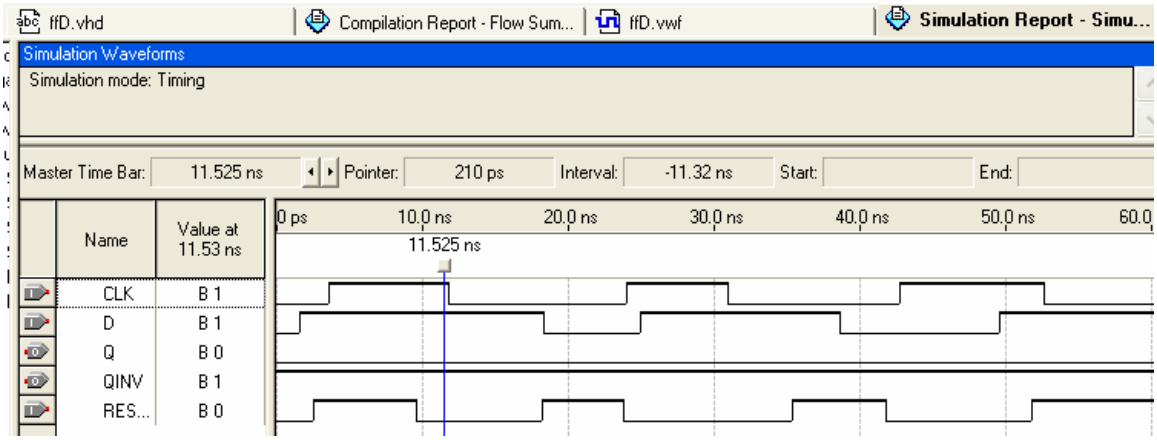
TRUTH TABLE

T	Qin	OUTPUT
0	0	0
0	1	0
1	0	1
1	1	1

VHDL CODE

```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
ENTITY ffD IS
PORT(D,CLK,RESET:IN BIT;
        Q,QINV:OUT BIT);
END ffD;
ARCHITECTURE behav OF ffD IS
BEGIN
        PROCESS
                BEGIN
                WAIT UNTIL CLK='1' AND CLK 'EVENT;

                IF(RESET='1') THEN Q<='0';QINV<='1';
                ELSIF D='1' THEN Q<='1';QINV<='0';
                ELSE Q<='0';QINV<='1';
                END IF;
        END PROCESS;
END behav;
```

Output waveform:



Timing analysis:

ffD.vhd | Compilation Report - Flo... | ffD.vwf | Simulation Report - Simul...

Registered Performance | tpd | tsu | tco | th | Custom Delays

	Slack	Required tsu	Actual tsu	From	To	To Clock
1	N/A	None	3.849 ns	RESET	Q~reg0	CLK
2	N/A	None	3.413 ns	D	Q~reg0	CLK

ffD.vhd | Compilation Report - Flo... | ffD.vwf | Simulation Report - Simul...

Registered Performance | tpd | tsu | tco | th | Custom Delays

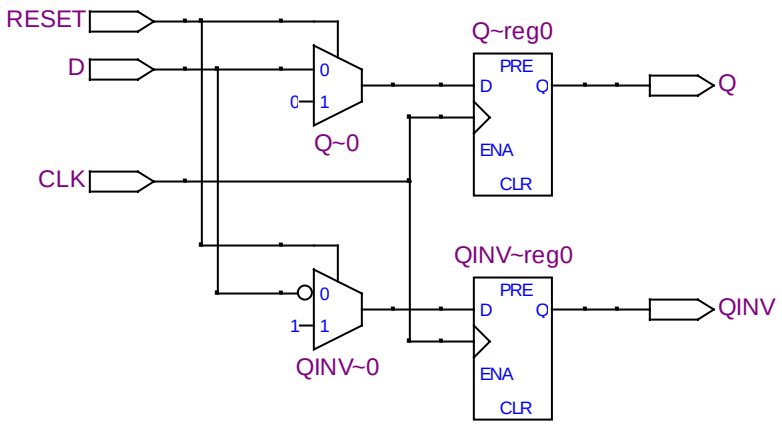
	Slack	Required tco	Actual tco	From	To	From Clock
1	N/A	None	6.916 ns	Q~reg0	QINV	CLK
2	N/A	None	6.916 ns	Q~reg0	Q	CLK

ffD.vhd | Compilation Report - Flo... | ffD.vwf | Simulation Report - Simul...

Registered Performance | tpd | tsu | tco | th | Custom Delays

	Minimum Slack	Required th	Actual th	From	To	To Clock
1	N/A	None	-3.361 ns	D	Q~reg0	CLK
2	N/A	None	-3.797 ns	RESET	Q~reg0	CLK

RTL-viewer:

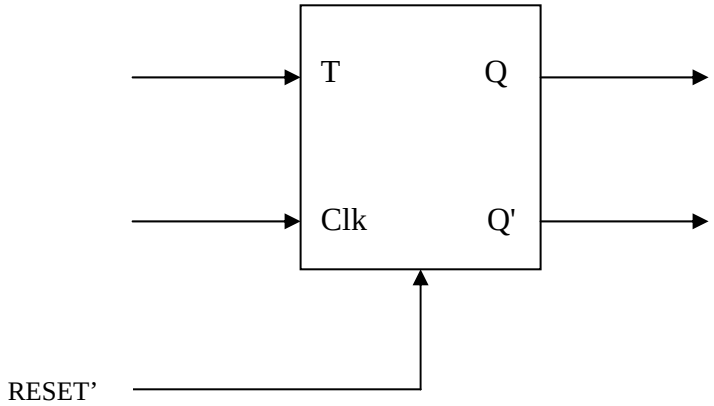


Conclusion:

-The truth table for delay flip-flop is verified.
-also the RTL and timing analysis were obtained.

T -FLIPFLOP

AIM: To develop a VHDL code for T flipflop.
ENTITY:



TRUTH TABLE

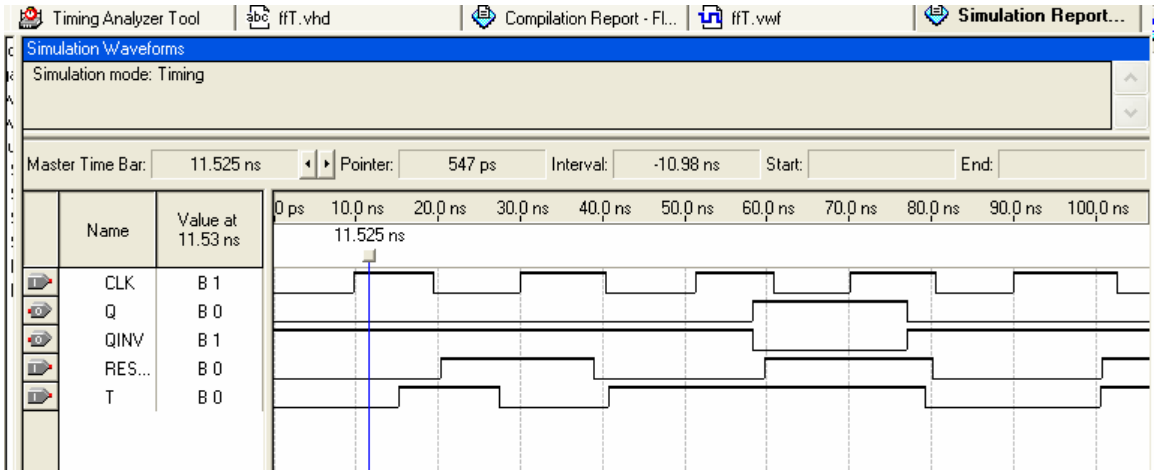
T	Qin	OUTPUT
0	0	0
0	1	1
1	0	1
1	1	0

VHDL CODE

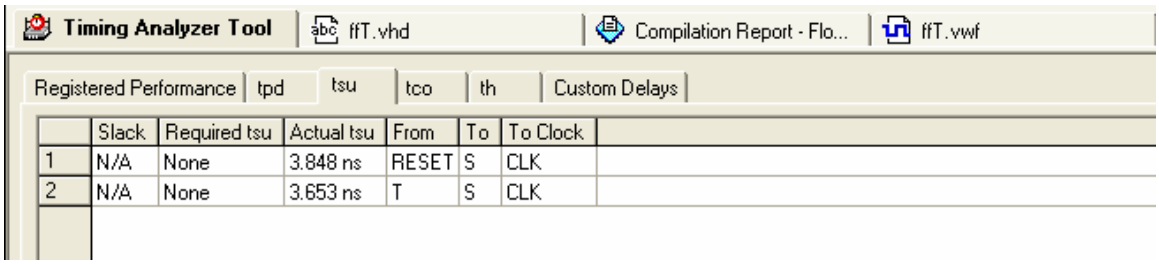
```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
ENTITY ffT IS
PORT(T,CLK,RESET:IN BIT;
        Q,QINV:OUT BIT);
END ffT;
ARCHITECTURE behav OF ffT IS
SIGNAL S:BIT;
BEGIN
        PROCESS
                BEGIN
                        WAIT UNTIL CLK='1' AND CLK 'EVENT;
                        IF(RESET='1')THEN S<='0';
                        ELSIF T='1' THEN S<=NOT S;
```

```
            END IF;
        END PROCESS;
        Q<=S;
        QINV<=NOT S;
END behav;
```

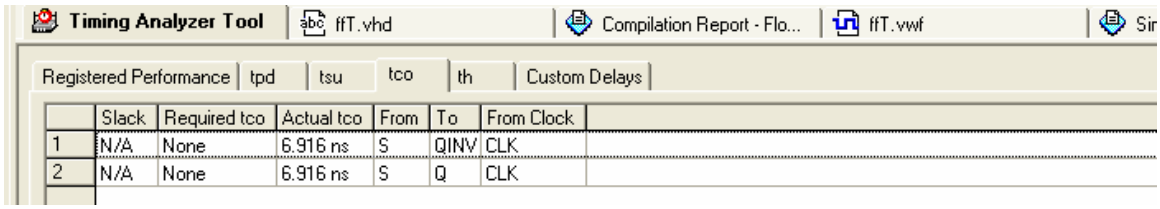
Output waveform:



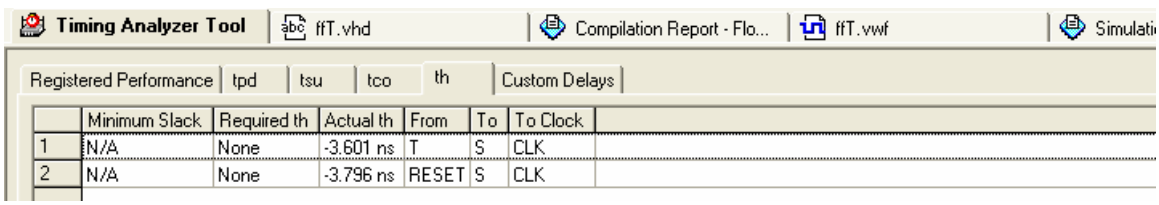
Timing analysis:



	Slack	Required tsu	Actual tsu	From	To	To Clock
1	N/A	None	3.848 ns	RESET	S	CLK
2	N/A	None	3.653 ns	T	S	CLK

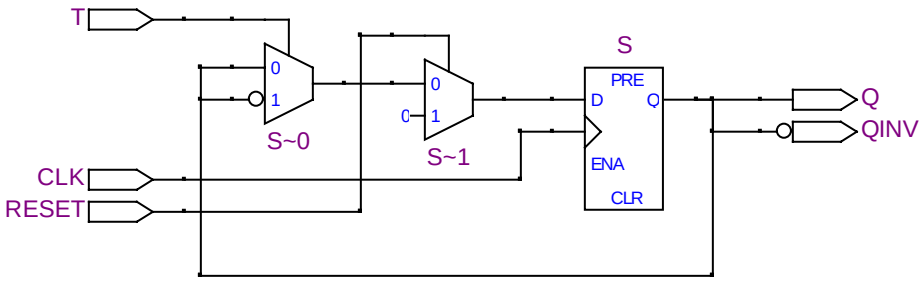


	Slack	Required tco	Actual tco	From	To	From Clock
1	N/A	None	6.916 ns	S	QINV	CLK
2	N/A	None	6.916 ns	S	Q	CLK



	Minimum Slack	Required th	Actual th	From	To	To Clock
1	N/A	None	-3.601 ns	T	S	CLK
2	N/A	None	-3.796 ns	RESET	S	CLK

RTL view:



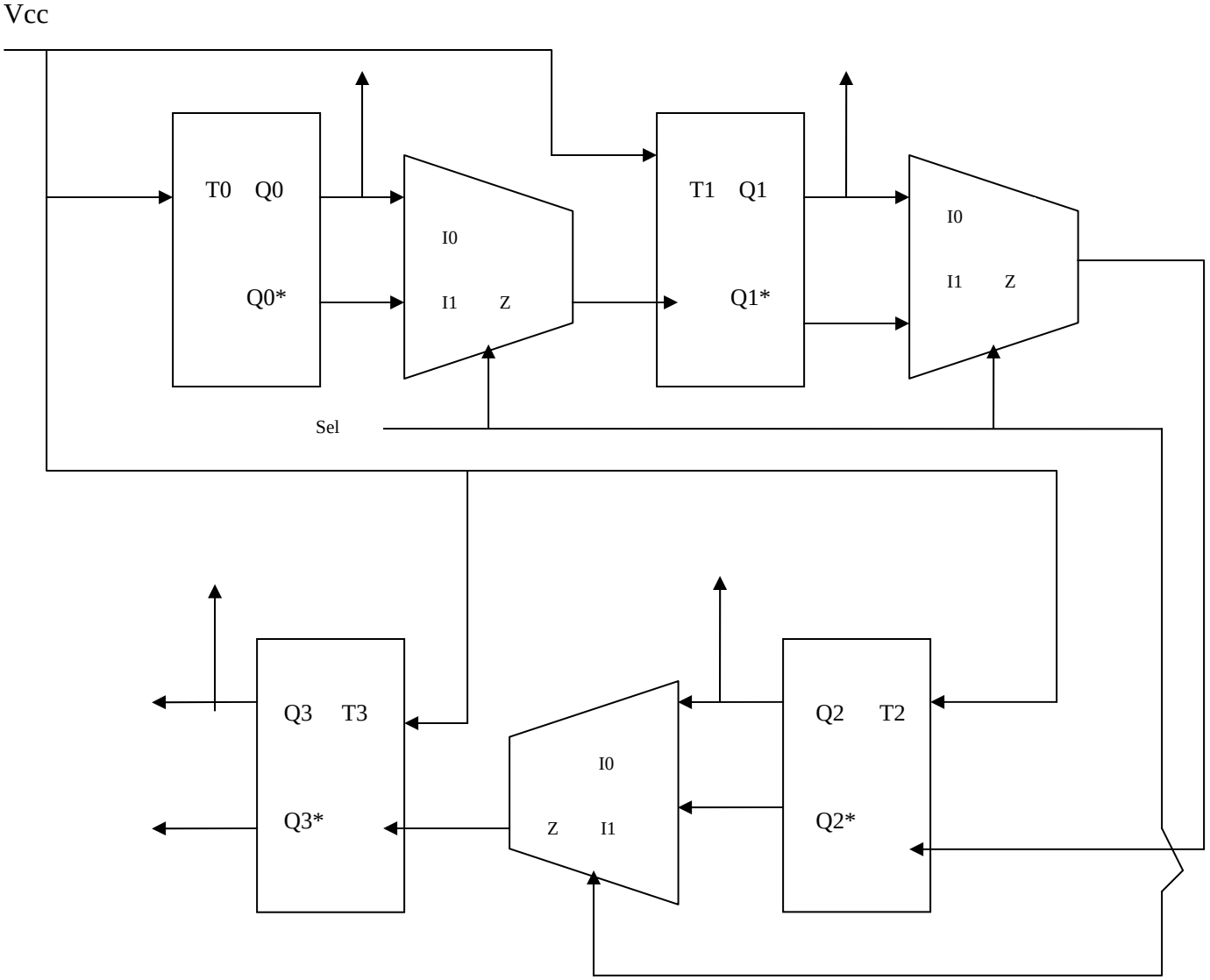
Conclusion:

-The truth table is verified for the toggle flip-flop.
-RTL-viewer and timing analysis is also obtained.

4-bit ASYNCHRONOUS UP-DOWN COUNTER

AIM: To develop a VHDL code for a four bit asynchronous up-down counter.

ENTITY:



BASIC COMPONENTS: (1) T-FLIPFLOP
(2) 2:1 MULTIPLEXER

TRUTH TABLE FOR BASIC COMPONENTS:
T-FLIPFLOP:

I/P	PS	NS
0	Qn	Qn
1	Qn	Qn*

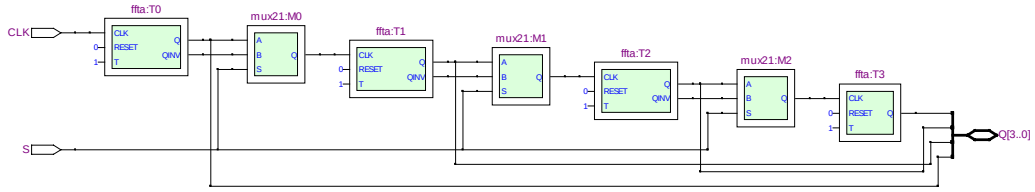
2:1 MULTIPLEXER:
Input lines: I0, I1

Sel	Z
0	I0
1	I1

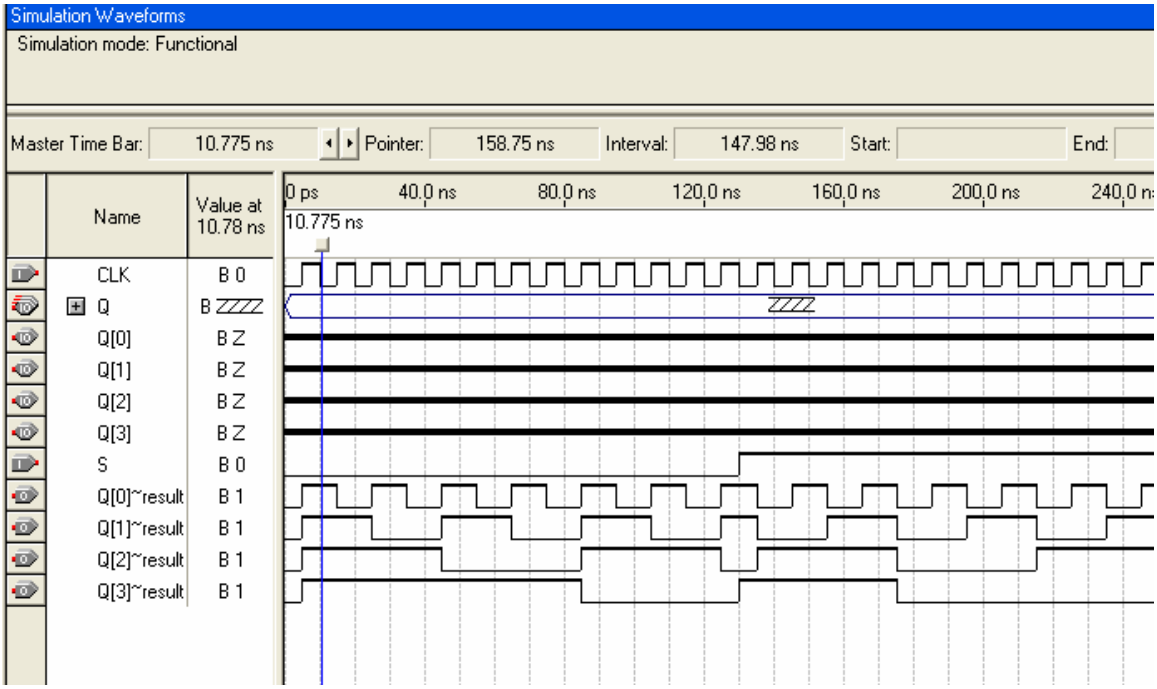
VHDL Code:

```
library ieee;
use ieee.std_logic_1164.all;//standard library packages
entity bit4audcounter is
port(s,clk:in std_logic;q:inout std_logic_vector(3 downto 0));//i/p and o/p declaration
end bit4audcounter;

architecture struc of bit4audcounter is
component mux21//declaration of basic components used
port(a,b,s:in std_logic;y:out std_logic);
end component;
component fft
port(t,clk,reset:in std_logic;q,q_inv:inout std_logic);
end component;
signal m:std_logic_vector(2 downto 0);
signal q_inv:std_logic_vector(3 downto 0);
begin                                   //mapping
t0:fft port map('1',clk,'0',q(0),q_inv(0));
m0:mux21 port map(q(0),q_inv(0),s,m(0));
t1:fft port map('1',m(0),'0',q(1),q_inv(1));
m1:mux21 port map(q(1),q_inv(1),s,m(1));
t2:fft port map('1',m(1),'0',q(2),q_inv(2));
m2:mux21 port map(q(2),q_inv(2),s,m(2));
t3:fft port map('1',m(2),'0',q(3),q_inv(3));
end struc;
```

RTL Viewer:



OUTPUT WAVEFORM:



TIMING ANALYSIS:

Registered Performance | tpd | tsu | tco | th | Custom Delays

	Slack	Required tco	Actual tco	From	To	From Clock
1	N/A	None	11.980 ns	fftA:T3|S	Q[3]	CLK
2	N/A	None	11.003 ns	fftA:T3|S	Q[3]	S
3	N/A	None	10.143 ns	fftA:T2|S	Q[2]	CLK
4	N/A	None	9.166 ns	fftA:T2|S	Q[2]	S
5	N/A	None	8.302 ns	fftA:T1|S	Q[1]	CLK
6	N/A	None	7.325 ns	fftA:T1|S	Q[1]	S
7	N/A	None	6.514 ns	fftA:T0|S	Q[0]	CLK

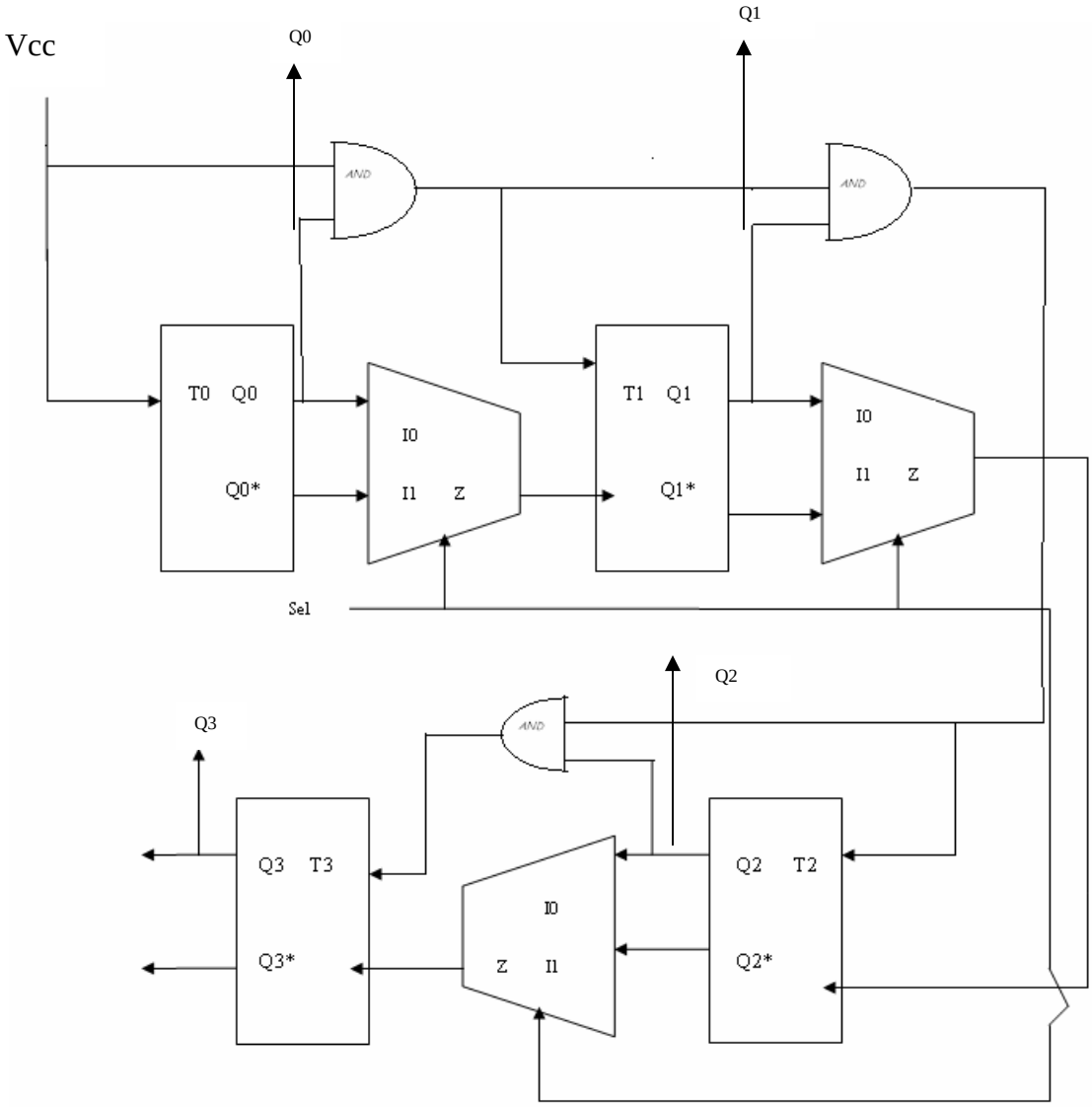
Conclusions:

We find that the counter acts as an up counter when s=1 and down counter when s=0. We also verify that output of one flipflop actys as the clock for the next..thus resulting in uniform counting.

4-bit SYNCHRONOUS UP-DOWN COUNTER

AIM: To develop a VHDL code for a four bit synchronous up-down counter.

ENTITY:



BASIC COMPONENTS: (1) T-FLIPFLOP
(2) 2:1 MULTIPLEXER
(3) AND

TRUTH TABLE FOR BASIC COMPONENTS:
T-FLIPFLOP:

I/P	PS	NS
0	Qn	Qn
1	Qn	Qn*

2:1 MULTIPLEXER:
Input lines: I0, I1

Sel	Z
0	I0
1	I1

2 INPUT AND GATE
Input Lines

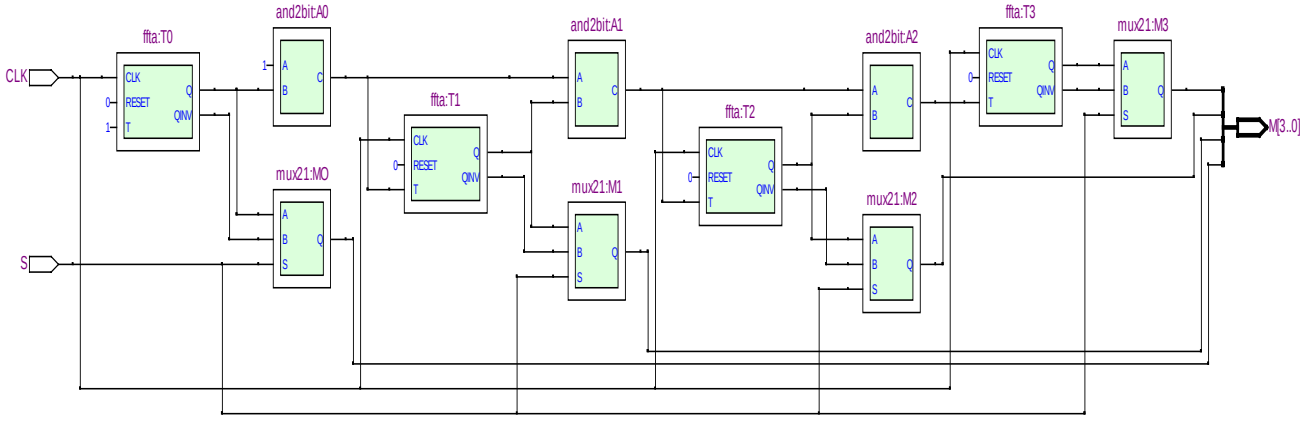
A	B	OUTPUT
0	0	0
0	1	0
1	0	0
1	1	1

VHDL Code:

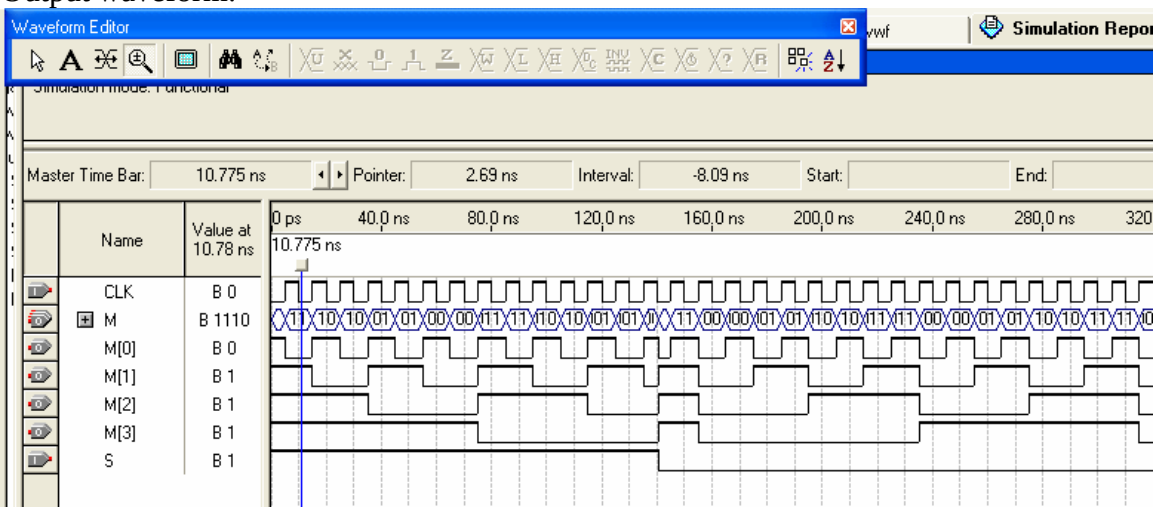
```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;//standard library packages
ENTITY counter4uds IS
PORT(CLK,S:IN BIT;
        M:OUT BIT_VECTOR(3 DOWNTO 0));//i/p and o/p declaration
END counter4uds;
ARCHITECTURE struc OF counter4uds IS
COMPONENT ffta //declaration of basic components
PORT (T,CLK,RESET:IN BIT;Q,QINV:OUT BIT);
END COMPONENT;
COMPONENT mux21 IS
PORT(A,B,S:IN BIT;Q:OUT BIT);
END COMPONENT;
COMPONENT and2bit
PORT(A,B:IN BIT;C:OUT BIT);
END COMPONENT;
SIGNAL Q_INV,Q:BIT_VECTOR(3 DOWNTO 0);
SIGNAL A:BIT_VECTOR(2 DOWNTO 0);
```

```
BEGIN//mapping
        T0:ffta PORT MAP('1',CLK,'0',Q(0),Q_INV(0));
        A0:and2bit PORT MAP('1',Q(0),A(0));
        T1:ffta PORT MAP(A(0),CLK,'0',Q(1),Q_INV(1));
        A1:and2bit PORT MAP(A(0),Q(1),A(1));
        T2:ffta PORT MAP(A(1),CLK,'0',Q(2),Q_INV(2));
        A2:and2bit PORT MAP(A(1),Q(2),A(2));
        T3:ffta PORT MAP(A(2),CLK,'0',Q(3),Q_INV(3));
        MO:mux21 PORT MAP(Q(0),Q_INV(0),S,M(0));
        M1:mux21 PORT MAP(Q(1),Q_INV(1),S,M(1));
        M2:mux21 PORT MAP(Q(2),Q_INV(2),S,M(2));
        M3:mux21 PORT MAP(Q(3),Q_INV(3),S,M(3));
END struc;
```

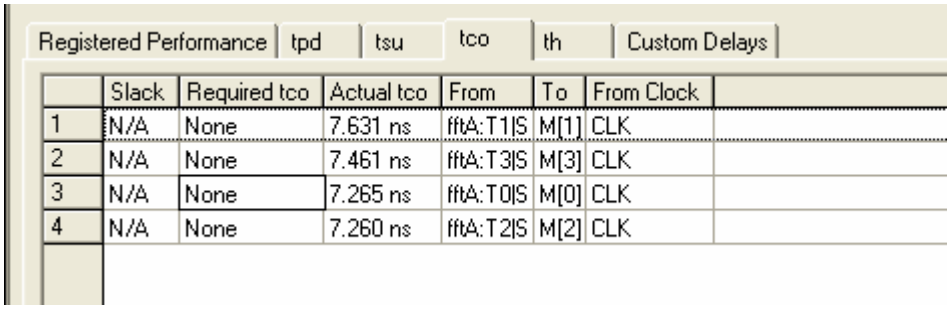
RTL Viewer:



Output waveform:



Timing Analysis:



Registered Performance | tpd | tsu | tco | th | Custom Delays

	Slack	Required tco	Actual tco	From	To	From Clock
1	N/A	None	7.631 ns	fftA:T1|S	M[1]	CLK
2	N/A	None	7.461 ns	fftA:T3|S	M[3]	CLK
3	N/A	None	7.265 ns	fftA:T0|S	M[0]	CLK
4	N/A	None	7.260 ns	fftA:T2|S	M[2]	CLK

CONCLUSIONS:
We find that the synchronous up down counter is driven by a common clock and thus there is uniform propagation delay. We also verify that this counter acts like an up counter when s=0 and down counter when s=1