DIGITAL SYSTEMS DESIGN LAB-17EC66

B.Tech. IV-Sem., ECE

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING,

Lakireddy Bali Reddy College of Engineering (AUTONOMOUS),

L.B.Reddy Nagar, MYLAVARAM – 521230

LAKKIREDDY BALI REDDY COLLEGE OF ENGINEERING
DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING
(Autonomous & Affiliated to JNTUK, Kakinada& Approved by AICTE, New Delhi,
Accredited by NBA, Certified by ISO 9001:2015)
L B Reddy Nagar, Mylavaram-521 230, Krishna District, Andhra Pradesh.

Digital System Design Lab–17EC66

LIST OF EXPERIMENTS

The following experiments are to be simulated using Xilinx ISE 14.7

1. LOGIC GATES – DATA FLOW AND BEHAVIOURAL MODEL
2. FULL ADDERS/ SUBTRACTOR
3. BINARY TO GRAY AND GRAY TO BINARY CODE CONVERTERS
4. 3X 8 DECODER – 74138.
5. 4 BIT MAGNITUDE COMPARATOR – 7485.
6. a.8 X 1 MULTIPLEXER – 74151 b. 1X4 DEMULTIPLEXER – 74155
7. D,J-K,T FLIP-FLOPS
8. DECADE COUNTER – 7490.
9. UP/DOWN COUNTER
10. SHIFT REGISTER-7495.
11. UNIVERSAL SHIFT REGISTERS – 74194/195.
12. RAM (16 x 4) – 74189 (READ AND WRITE OPERATIONS)

1. LOGIC GATES

AIM: To design and simulate logic gates using Xilinx ISE14.7

Tools Required:

Simulator-Xilinx ISE14.7

Synthesis-XST

Preferred language-VHDL

THEORY:

A logic gate performs a logical operations with one or more logic inputs and produces a single logic output to implement Boolean functions.A **logic gate** is an elementary building block of a digital **circuit**.

NOT gate:A NOT gate is more commonly called as an inverter. The circle on the symbol is called a *bubble* and is used in logic diagrams to indicate a logic negation between the external logic state and the internal logic state (1 to 0 or vice versa)

A	Y1
0	1
1	0

`

AND gate:The **AND gate** is a basic digital logic gate that implements logical conjunction

A	B	Y2
0	0	0
0	1	0
1	0	0
1	1	1

ORgateThe *OR gate* behaves the logical inclusive "or." The output is "true" if either or both of the inputs are "true." If both inputs are "false," then the output is "false

A	B	Y3
0	0	0
0	1	1
1	0	1
1	1	1

NAND gate:The *NAND gate* operates as an AND gate followed by a NOT gate. It acts in the manner of the logical operation "and" followed by negation. The output is "false" if both inputs are "true." Otherwise, the output is "true."

A	B	Y4
0	0	1
0	1	1
1	0	1
1	1	0

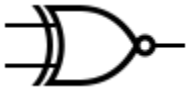
NORgate:The *NOR gate* is a combination OR gate followed by an inverter. Its output is "true" if both inputs are "false." Otherwise, the output is "false

A	B	Y5
0	0	1
0	1	0
1	0	0
1	1	0

EX-OR gate:The *XOR* (*exclusive-OR*) *gate* acts in the same way as the logical "either/or.Another way of looking at this circuit is to observe that the output is 1 if the inputs are different, but 0 if the inputs are the same.

A	B	Y6
0	0	0
0	1	1
1	0	1
1	1	0

EX-NOR gate:The *XNOR (exclusive-NOR) gate* is a combination XOR gate followed by an inverter. Its output is "true" if the inputs are the same, and"false" if the inputs are different



A	B	Y7
0	0	1
0	1	0
1	0	0
1	1	1

PROCEDURE:

- Open ISE project navigator in the Xilinx ISE design suite 14.7 software.
- Go to file -> new project and set the project settings as
    - Product category: All
    - Family: SPARTAN 3E
    - Device: XC3S500E
    - Package:4fg320
    - Synthesis Tool: XST
    - Simulator: ISim
    - Preferred Language: VHDL

- Select the device -> right click -> new source -> VHDL Module and give the input and output ports ,then write the VHDL code in VHDL module.
- Select left top-level view as Implementation
- Compile for syntax errors.
- Select the device -> right click -> new source -> VHDL TEST BENCH and give the TEST input/TEST patterns
- Select left top-level view as Simulation
- Compile for behavioral syntax errors.
- Simulate design using ISE Simulator.

VHDL Program:

```
library IEEE;

use IEEE.STD_LOGIC_1164.ALL;

entity logicgates is

   Port ( A,B : in  STD_LOGIC;

          Y1,Y2,Y3,Y4,Y5,Y6,Y7 : out  STD_LOGIC);

end logicgates;
```

```
architecture Behavioral of logicgates is

begin

   Y1<= NOT A;

   Y2<= A AND B;

   Y3<= A OR B;

   Y4<= A NAND B;

   Y5 <= A NOR B;

   Y6<= A XOR B;

   Y7<=  A XNOR B;

end Behavioral;
```

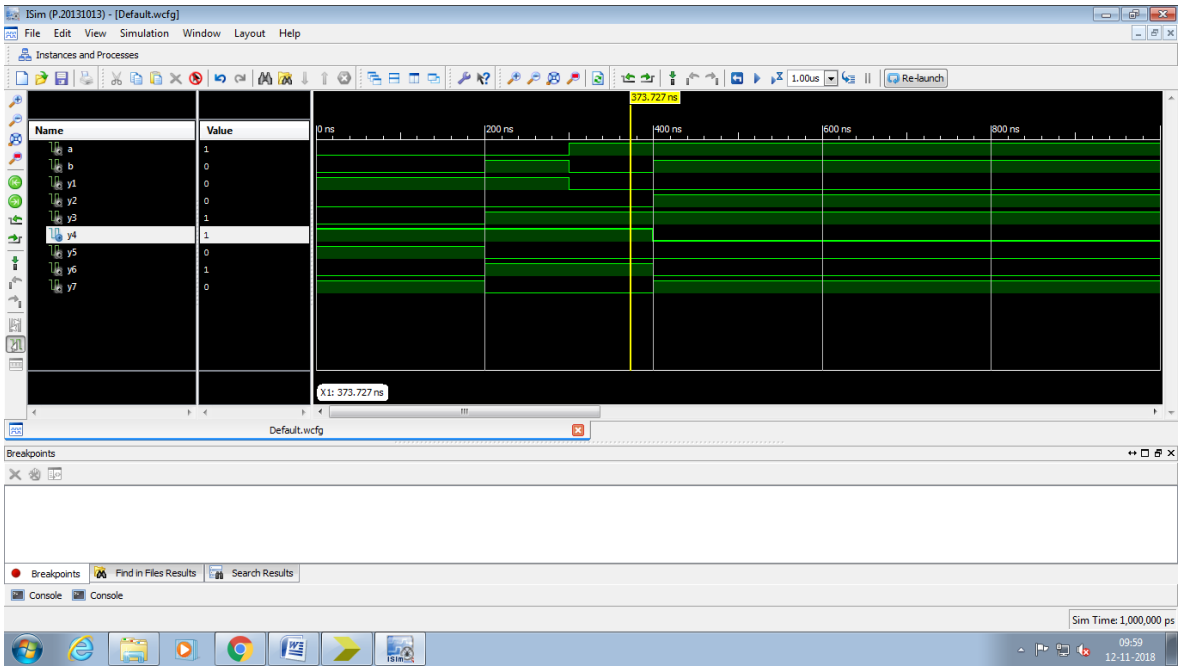
TEST BENCH :

```
LIBRARY ieee;

USE ieee.std_logic_1164.ALL;

 ENTITY logicgatestb IS

END logicgatestb;

 ARCHITECTURE behavior OF logicgatestb IS

    COMPONENT logicgates

   PORT(

        A : IN  std_logic;

        B : IN  std_logic;

        Y1 : OUT  std_logic;

        Y2 : OUT  std_logic;

        Y3 : OUT  std_logic;

        Y4 : OUT  std_logic;

        Y5 : OUT  std_logic;

        Y6 : OUT  std_logic;

        Y7 : OUT  std_logic

       );
```

```
    END COMPONENT;

    --Inputs
   signal A : std_logic := '0';
   signal B : std_logic := '0';
            --Outputs
   signal Y1 : std_logic;
   signal Y2 : std_logic;
   signal Y3 : std_logic;
   signal Y4 : std_logic;
   signal Y5 : std_logic;
   signal Y6 : std_logic;
   signal Y7 : std_logic;
   -- No clocks detected in port list. Replace <clock> below with
  -- appropriate port name
BEGIN
  -- Instantiate the Unit Under Test (UUT)
   uut: logicgates PORT MAP (
          A => A,
          B => B,
          Y1 => Y1,
          Y2 => Y2,
          Y3 => Y3,
          Y4 => Y4,
          Y5 => Y5,
          Y6 => Y6,
          Y7 => Y7
        );
     -- Stimulus process
```

```
 stim_proc: process
 begin
-- hold reset state for 100 ns.
    wait for 100 ns;
  A<='0';B<='0';
wait for 100 ns;
  A<='0';B<='1';
 wait for 100 ns;
  A<='1';B<='0';
wait for 100 ns;
A<='1';B<='1';
 wait;
 end process;
END;
```

SIMULATION RESULTS :



CONCLUSION:

Hence function of all logic gates are simulated using Xilinx ISE 14.7

VIVA Questions with answer.

Q.1What is VHDL?

Ans. VHDL is the VHSIC Hardware Description Language. VHSIC is an abbreviation for Very High Speed Integrated Circuit

Q.2 Name all the basic gates.

Ans. i) AND ii) OR iii) NOT

Q.3 How many architectures are present in VHDL.

Ans . 4, behavior, dataflow, structural and mixed

Q.4 What is the full form of IEEE?

Ans Institute of Electrical and Electronic Engineering.

Q.5 Define Entity

Ans. It is an external view of a design unit.

Q6. Why NAND and NOR are called universal gates?

Ans. Because all the basic gates can be derive from them.

2. FULL ADDER / SUBTRACTOR

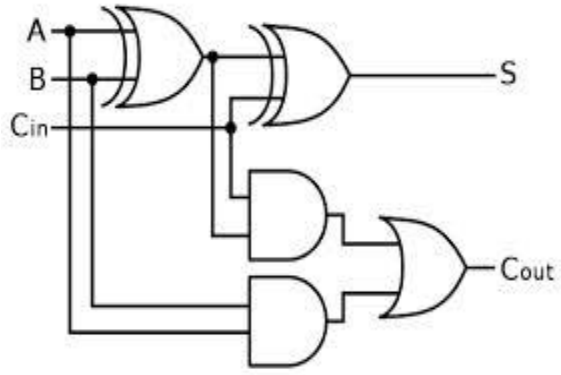
AIM: To Design and Simulate full adder/sub tractor using Xilinx ISE 14.7

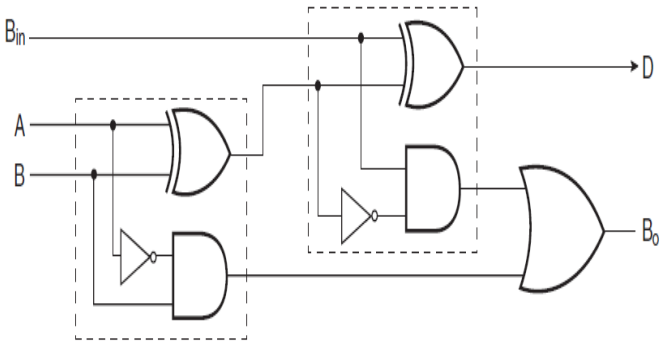
TOOLS REQUIRED:

Tools: Simulator-Xilinx ISE

Preferred language-VHDL

THEORY:





INPUTS			OUTPUT	
A	B	C-IN	C-OUT	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

INPUT			OUTPUT	
A	B	C	DIFF.	BORROW
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

Sum S= A XOR B XOR Cin;

Carry Cout = (A AND B) OR (B AND Cin) OR (Cin AND A);

Difference D = A XOR B XOR BIN;

Borrow BOUT< ((NOT A) AND B) OR ((NOT A AND BIN) OR (B AND BIN):

PROCEDURE:

- Open ISE project navigator in the Xilinx ISE design suite 14.7 software.
- Go to file -> new project and set the project settings as
    - Product category: All
    - Family: SPARTAN 3E
    - Device: XC3S500E
    - Package:4fg320
    - Synthesis Tool: XST
    - Simulator: ISim
    - Preferred Language: VHDL

- Select the device -> right click -> new source -> VHDL Module and give the input and output ports ,then write the VHDL code in VHDL module.
- Select left top-level view as Implementation
- Compile for syntax errors.
- Select the device -> right click -> new source -> VHDL TEST BENCH and give the TEST input/TEST patterns
- Select left top-level view as Simulation
- Compile for behavioral syntax errors.
- Simulate design using ISE Simulator

VHDL PROGRAM:

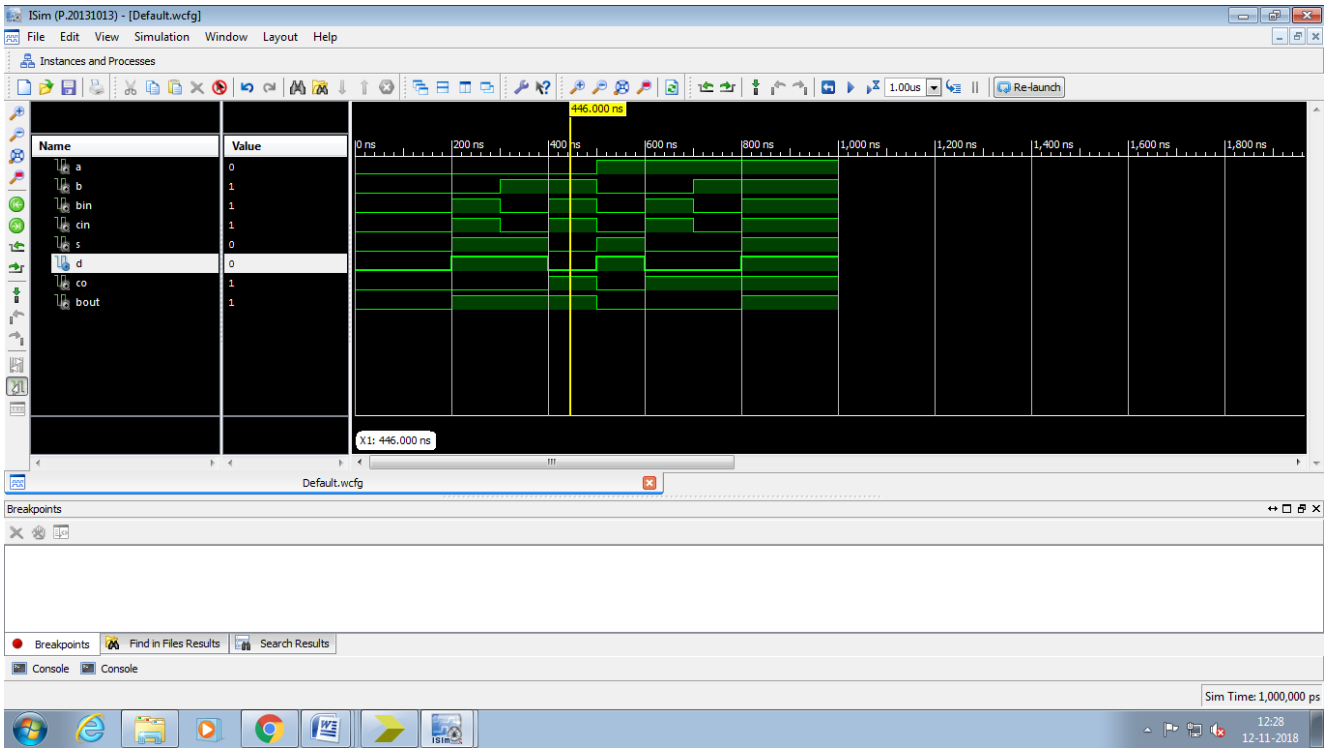
```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
-- Uncomment the following library declaration if using
entity FULLADDER-SUBTRACTOR  is
   Port ( A,B,BIN,CIN : in  STD_LOGIC;
          S,D,CO,BOUT : out  STD_LOGIC);
end FULLADDER-SUBTRACTOR ;

architecture Behavioral of FULLADDER-SUBTRACTOR  is

begin
S<= A XOR B XOR CIN;
D <= A XOR B XOR BIN;
CO <= (A AND B) OR (B AND CIN)OR(CIN AND A);
BOUT<=((NOT A) AND B) OR(( NOT A) AND BIN) OR ( B AND BIN);
end Behavioral;
```

TEST BENCH :

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
 ENTITY FULLADDERTB IS
 END FULLADDERTB;
 ARCHITECTURE behavior OF FULLADDERTB IS
  -- Component Declaration for the Unit Under Test (UUT)
    COMPONENT FULLADDER
   PORT(
       A : IN  std_logic;
       B : IN  std_logic;
       BIN : IN  std_logic;
       CIN : IN  std_logic;
       S : OUT  std_logic;
       D : OUT  std_logic;
       CO : OUT  std_logic;
       BOUT : OUT  std_logic
      );
  END COMPONENT;
    --Inputs
  signal A : std_logic := '0';
  signal B : std_logic := '0';
  signal BIN : std_logic := '0';
  signal CIN : std_logic := '0';
```

```
          --Outputs
 signal S : std_logic;
 signal D : std_logic;
 signal CO : std_logic;
 signal BOUT : std_logic;
 -- No clocks detected in port list. Replace <clock> below with
 -- appropriate port name
BEGIN
-- Instantiate the Unit Under Test (UUT)
 uut: FULLADDER PORT MAP (
       A => A,
       B => B,
       BIN => BIN,
       CIN => CIN,
       S => S,
       D => D,
       CO => CO,
       BOUT => BOUT
     );   -- Stimulus process
 stim_proc: process
 begin
  -- hold reset state for 100 ns.
    wait for 100 ns;

   A<='0'; B<='0'; CIN<='0'; BIN<='0';
  Wait for 100 ns;

   A<='0'; B<='0'; CIN<='1';BIN<='1';
          Wait for 100 ns;

   A<='0'; B<='1'; CIN<='0';BIN<='0';
 Wait for 100 ns;

   A<='0'; B<='1'; CIN<='1';BIN<='1';
 Wait for 100 ns;

   A<='1'; B<='0'; CIN<='0';BIN<='0';
  Wait for 100 ns;

   A<='1'; B<='0'; CIN<='1';BIN<='1';

 Wait for 100 ns;

A<='1'; B<='1'; CIN<='0';BIN<='0';
           Wait for 100 ns;

A<='1'; B<='1'; CIN<='1'; BIN<='1';

  wait;
end process;

END;
```

SIMULATION RESULTS:



CONCLUSION:

Hence function of full adder and subtractor is simulated using Xilinx ISE 14.7

VIVA –QUESTIONS

1. Why HDL is used?
2. What is the difference between sequential and combinational circuits?
3. Construct full adder using half adder.
4. Construct half subtractor
5. What is a test bench in **vhdl**?

3. BINARY TO GRAY AND GRAY TO BINARY CODE CONVERTERS

AIM: To Design and Simulate Binary to Gray and Gray to Binary Code Converters using Xilinx ISE 14.7

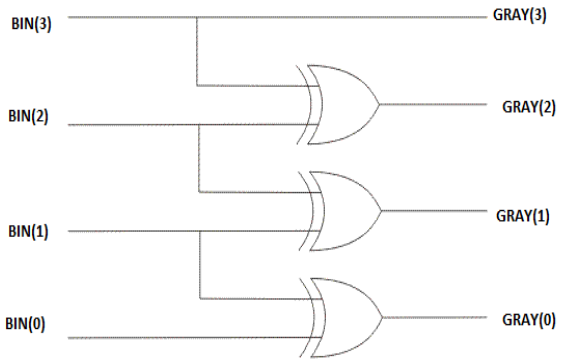
TOOLS REQUIRED:

Tools: Simulator-Xilinx ISE

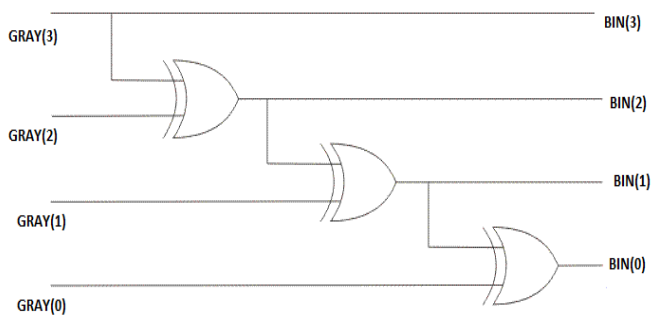
Preferred language-VHDL

THEORY:

Gray codes are non-weighted codes, where two successive values differ only on one bit.



Binary to Gray



Gray to Binary

PROCEDURE:

- Open ISE project navigator in the Xilinx ISE design suite 14.7 software.
- Go to file -> new project and set the project settings as
    - Product category: All
    - Family: SPARTAN 3E
    - Device: XC3S500E
    - Package:4fg320
    - Synthesis Tool: XST
    - Simulator: ISim
    - Preferred Language: VHDL

- Select the device -> right click -> new source -> VHDL Module and give the input and output ports ,then write the VHDL code in VHDL module.
- Select left top-level view as Implementation
- Compile for syntax errors.
- Select the device -> right click -> new source -> VHDL TEST BENCH and give the TEST input/TEST patterns
- Select left top-level view as Simulation
- Compile for behavioral syntax errors.
- Simulate design using ISE Simulator

VHDL PROGRAM:

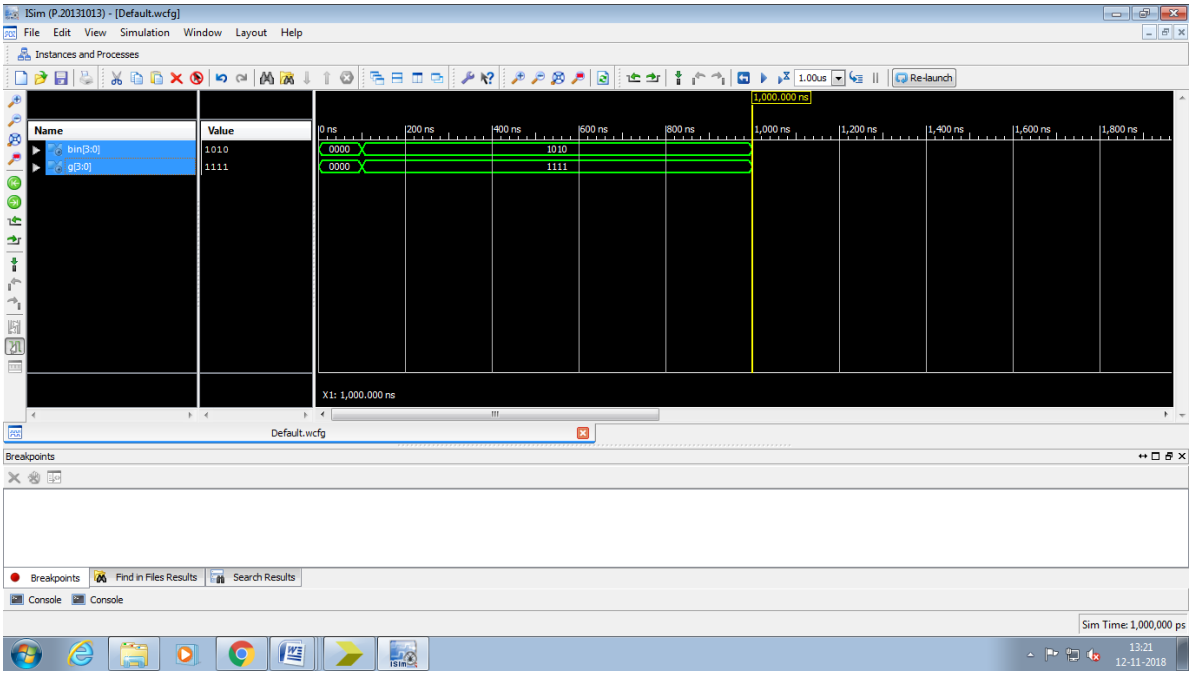
```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
entity graytobinary is
   Port ( bin,gin : in  STD_LOGIC_VECTOR (3 downto 0);
          B,G : out  STD_LOGIC_VECTOR (3 downto 0);
          S : in  STD_LOGIC);
end graytobinary;
architecture Behavioral of graytobinary is
begin
process(bin,gin,s)
begin
if (s='0') then
G( 3)<= bin ( 3);
G (2) <= bin ( 3) xor bin (2) ;
G( 1) <= bin ( 2) xor bin (1);
G ( 0) <= bin ( 1 ) xor bin (0);
```

```
end  if;
if ( s ='1') then
B( 3) <= gin(3);
B( 2) <= gin ( 3) xor gin (2) ;
B( 1) <= gin ( 3) xor gin (2) xor gin ( 1) ;
B( 0) <= gin ( 3) xor gin (2) xor gin ( 1 ) XOR gin(0);
end   if ;
end  process;

end Behavioral;
```

TEST BENCH :

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

--USE ieee.numeric_std.ALL;

ENTITY gtobtog_tb IS
END gtobtog_tb;

ARCHITECTURE behavior OF gtobtog_tb IS
    -- Component Declaration for the Unit Under Test (UUT)
    COMPONENT graytobinary
   PORT(
        bin : IN  std_logic_vector(3 downto 0);
        gin : IN  std_logic_vector(3 downto 0);
        B : OUT  std_logic_vector(3 downto 0);
        G : OUT  std_logic_vector(3 downto 0);
        S : IN  std_logic
       );
   END COMPONENT;

  --Inputs
  signal bin : std_logic_vector(3 downto 0) := (others => '0');
  signal gin : std_logic_vector(3 downto 0) := (others => '0');
  signal S : std_logic := '0';

          --Outputs
  signal B : std_logic_vector(3 downto 0);
  signal G : std_logic_vector(3 downto 0);
  -- No clocks detected in port list. Replace <clock> below with
  -- appropriate port name

BEGIN

          -- Instantiate the Unit Under Test (UUT)
  uut: graytobinary PORT MAP (
        bin => bin,
        gin => gin,
        B => B,
        G => G,
        S => S
       );
```

```
  stim_proc: process
  begin
    -- hold reset state for 100 ns.
    wait for 100 ns;

    bin<= "1100";
                  wait for  100 ns;
                  gin<= "1010";
    -- insert stimulus here

    wait;
  end process;

END;
```



CONCLUSION:

Hence function of gray to binary/binary to gray is simulated using Xilinx ISE 14.7

VIVA-QUESTIONS:

1. write a gray code for the given binary code?

2. write a binary code for the given gray code?

3. Applications of code converters?

4. List out the different code converter with examples?

5. How to differentiate dataflow and behavoral and structural architecture models ?