CAD OF ELECTRONICS LAB MANUAL

(EEC-653)

DEPARTMENT OF ELECTRONICS&COMMUNICATION ENGINEERING

27, Knowledge Park-III, Greater Noida, (U.P.)
Phone : 0120-2323854-58
website :- www.dronacharya.info

CONTENTS

SYLLABUS

(AS PER PRESCRIBED BY MTU, NOIDA.)

EEC-553 CAD OF ELECTRONICS LABL T P

0 0 3

PSPICE EXPERIMENTS

1. To verify the characteristics of Basic Digital Gates.
2. To verify the characteristics of Low pass and High pass filter.
3. Verification of Half–Wave and Full-Wave Rectifier.
4. To verify the characteristics of CE Amplifier.

VHDL EXPERIMENTS

1. Synthesis and simulation of Full Adder.

2. Synthesis and Simulation of Full Subtractor.

3. Synthesis and Simulation of 3 X 8 Decoder.

4. Synthesis and Simulation of 8 X 1 Multiplexer.

STUDY AND EVALUATION SCHEME

SESSIONAL EVALUATION:-

CLASS TEST	**:**	**10 MARKS**
TEACHER’S ASSESMENT	**:**	**10 MARKS**
EXTERNAL EXAM	**:**	**30 MARKS**
TOTAL	**:**	**50 MARKS**

LIST OF EXPERIMENTS

PSPICE EXPERIMENTS

1. To verify the characteristics of Basic Digital Gates.
2. To verify the characteristics of Low pass and High pass filter.
3. Verification of Half–Wave and Full-Wave Rectifier.
4. To verify the characteristics of CE Amplifier.

VHDL EXPERIMENTS

1. Synthesis and simulation of Full Adder.

2. Synthesis and Simulation of Full Subtractor.

3. Synthesis and Simulation of 3 X 8 Decoder.

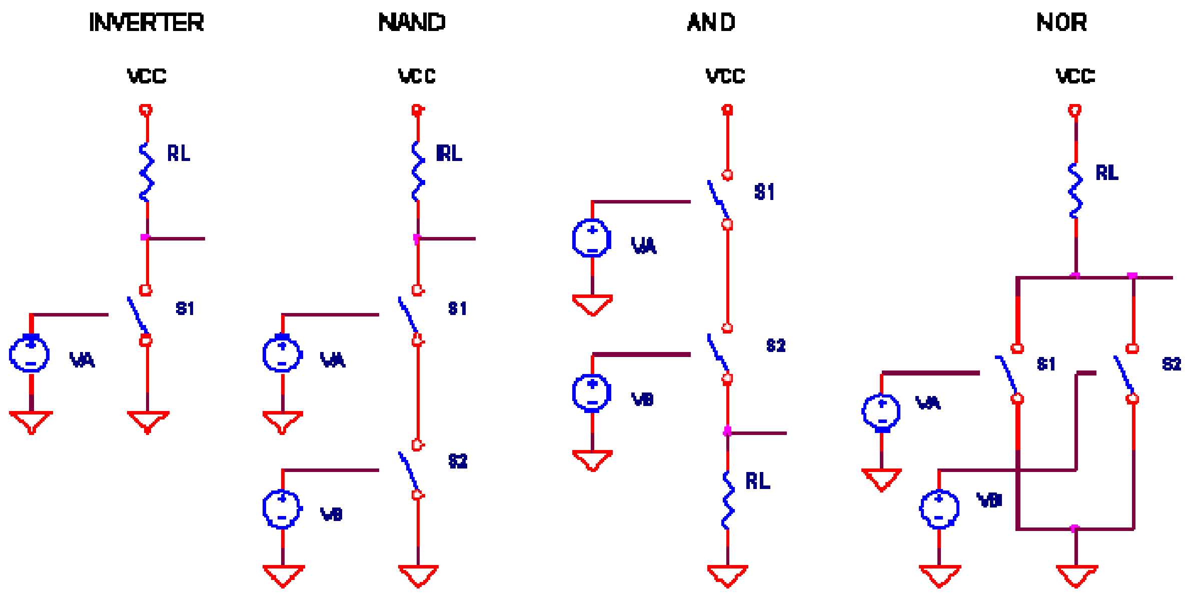
4. Synthesis and Simulation of 8 X 1 Multiplexer.

PSPICE EXPERIMENTS

EXPERIMENT NO. 01

AIM: To verify the characteristics of Basic Digital Gates

CIRCUIT :



You're simulating a circuit, it requires several digital gates, but you don't have a mixed-mode simulator. What to do? One solution involves creating simplified versions of the logic functions. To do this, we look to the NMOS transistor implementation of logic gates where the transistor acts like a voltage-controlled switch. But, instead of the transistor, we'll use the SPICE switch. Just like the transistor, the switch is defined to turn ON when the input voltage goes HI.

By placing the these switches in parallel or series, a variety of basic logic functions can come to life. Here's some helpful hints for logic circuit building:

FUNCTION OUTPUT

AND - Switches in Series

OR - Switches in Parallel

INVERTED - Pull-Up Resistor

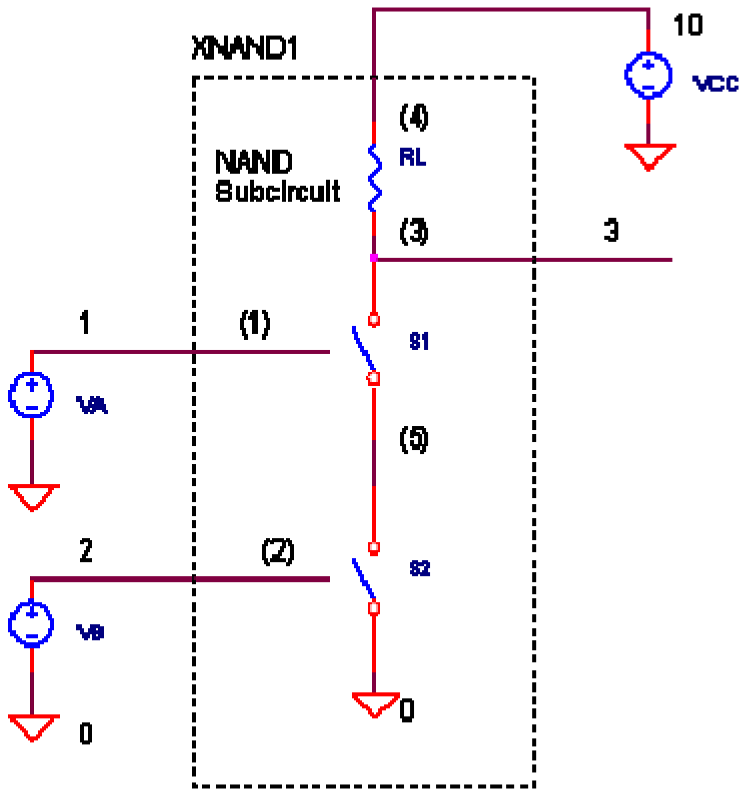
NON-INVERTED - Pull-Down Resistor

THE NAND GATE

So let's have a go at simulating the NAND gate. How do you describe its function? When both A and B are HI, the output is LO. Or stated another way - it's the AND function with an inverted output. The Boolean expression looks like

The circuit appears below. S1 and S2 in series create the AND function; RL in the pull-up position inverts the output. Defining the NAND gate as a subcircuit makes it easy to insert it into a few locations if you wish. The subcircuit nodes are listed in parenthesis.

S1.

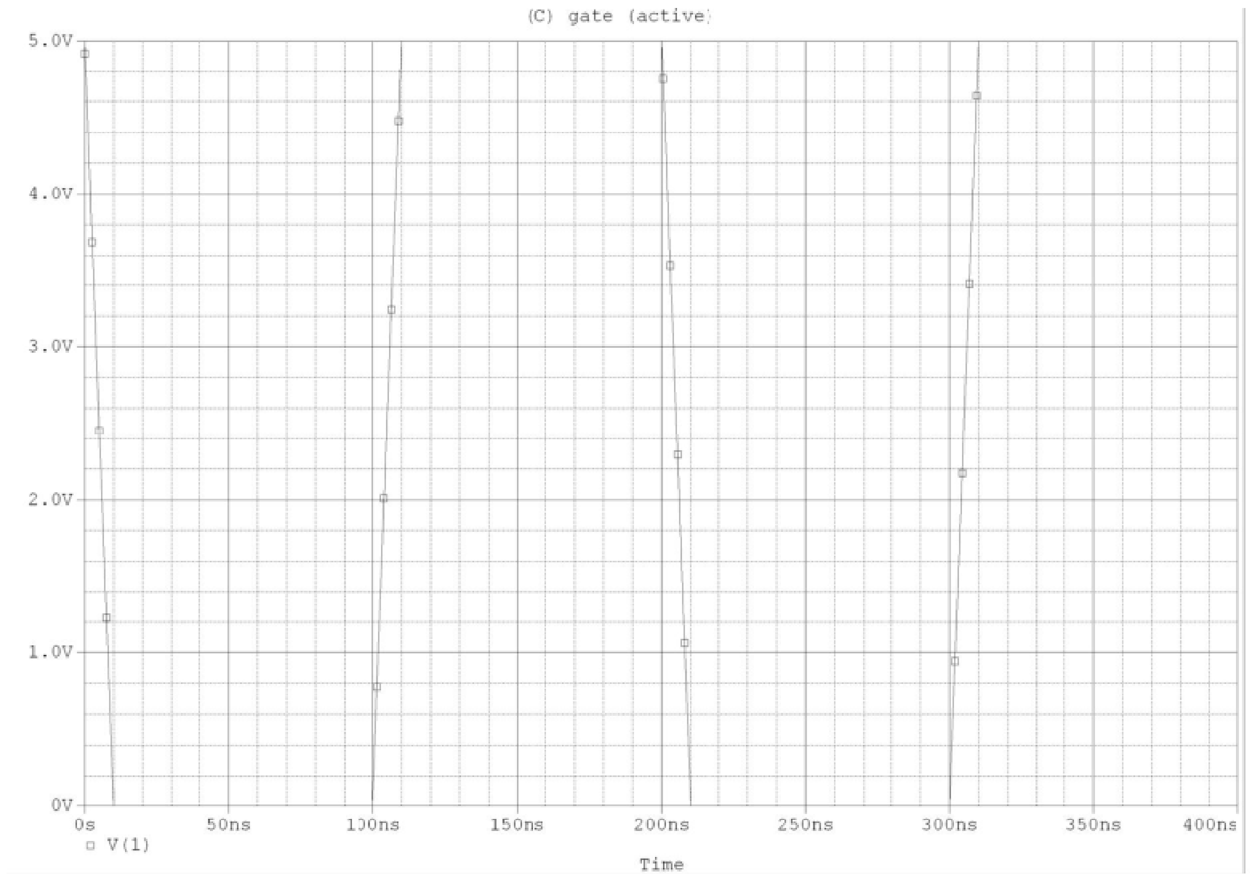


S1 and S2 are defined by RON = 10 _ and ROFF = 1 M_. Compared to the other resistances in the circuit, these should look like an ideal switches. More on the SPICE switch below.

Simulate the SPICE circuit named LOGIC_SW.CIR. VA and VB create two binary signals that form the sequence 00, 01, 10 and 11. VCC = +5V supplies power to the logic gate. Plot the inputs V(1), V(2) and the output V(3). For a clearer view, you might want to plot V(3) in a separate plot window. Does the output go LO when V(1) and V(2) are HI?

Note the finite rise and fall times of V(1) and V(2). You may have also noticed that the output V(3) quickly changes when the inputs pass through 2.5 V. This is the approximate logic threshold level defined for the SPICE switches.

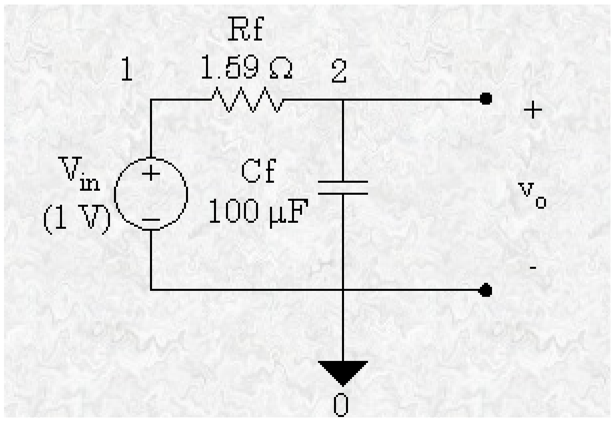
RESULT:



EXPERIMENT NO. 02

AIM: To verify the characteristics of Low pass and High pass filter.

CIRCUIT:



Here's a simple circuit for you to dive into running SPICE simulations and plotting results. What is the purpose of this circuit? Basically it has two roles: to **pass** the desired low frequency signals and **stop** the unwanted high frequency signals.

CUTOFF FREQUENCY

As stated above, the circuit has two roles: to pass the desired low frequency signals and stop the unwanted high frequency signals. But at what frequency does the filter change its behavior from passing the low ones to stopping the high ones. This is called the cut-off frequency.

$$fc = \frac{1}{2\pi \times R1 \times C1}$$

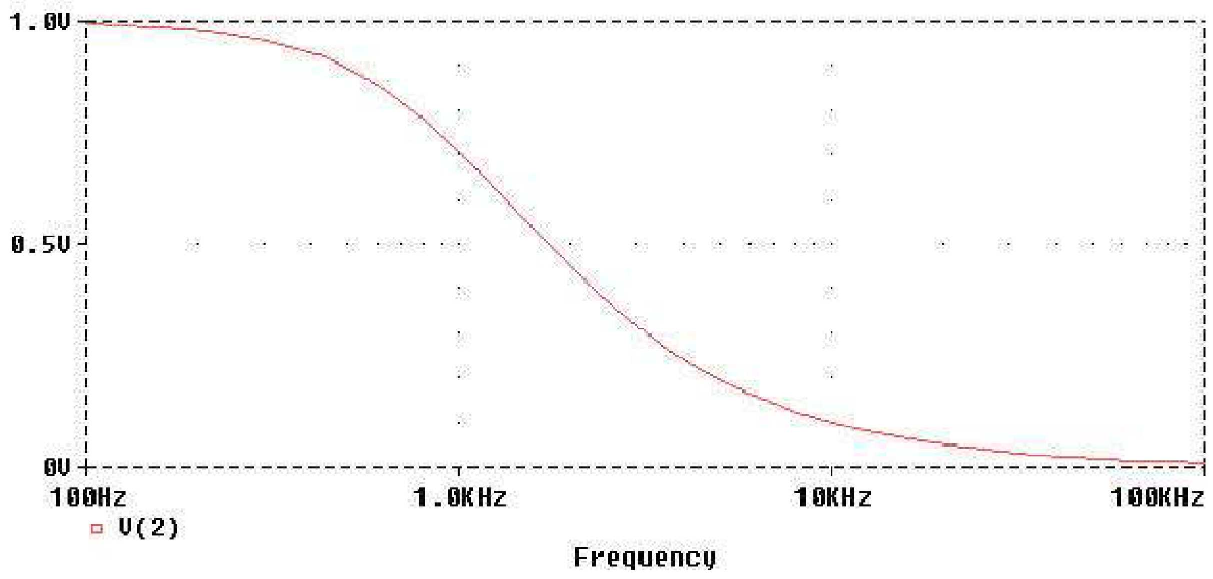
For R1=1k and C1=0.032uF you get fc = 5kHz. Run a simulation. Plot the AC (frequency) sweep results for the output magnitude VM(2) and phase VP(2). What does the magnitude look like before and after 5kHz?

SPICE FILE

```
VIN 1 0 AC 1V
```

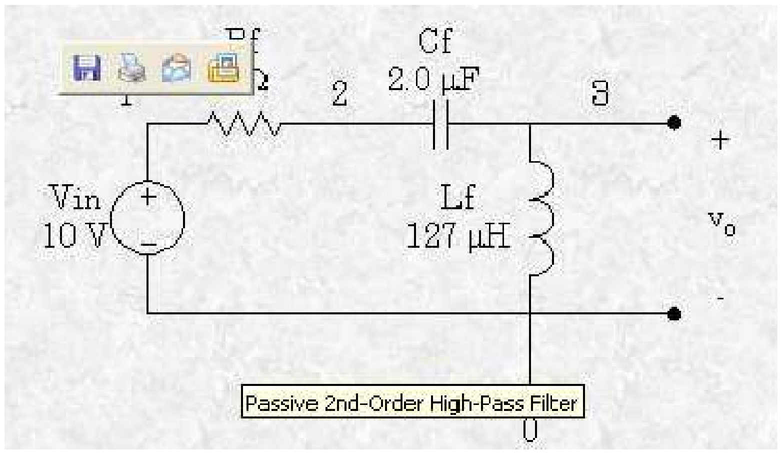
```
RF 1 2 1.59
CF 2 0 100UF
.AC DEC 20 100HZ 100KHZ
.PROBE
.END
```

After running this in PSpice, we start PROBE, choose "Add" from the "Trace" menu and plot the output voltage. PROBE provides the following graph.



HIGH PASS FILTER

CIRCUIT :

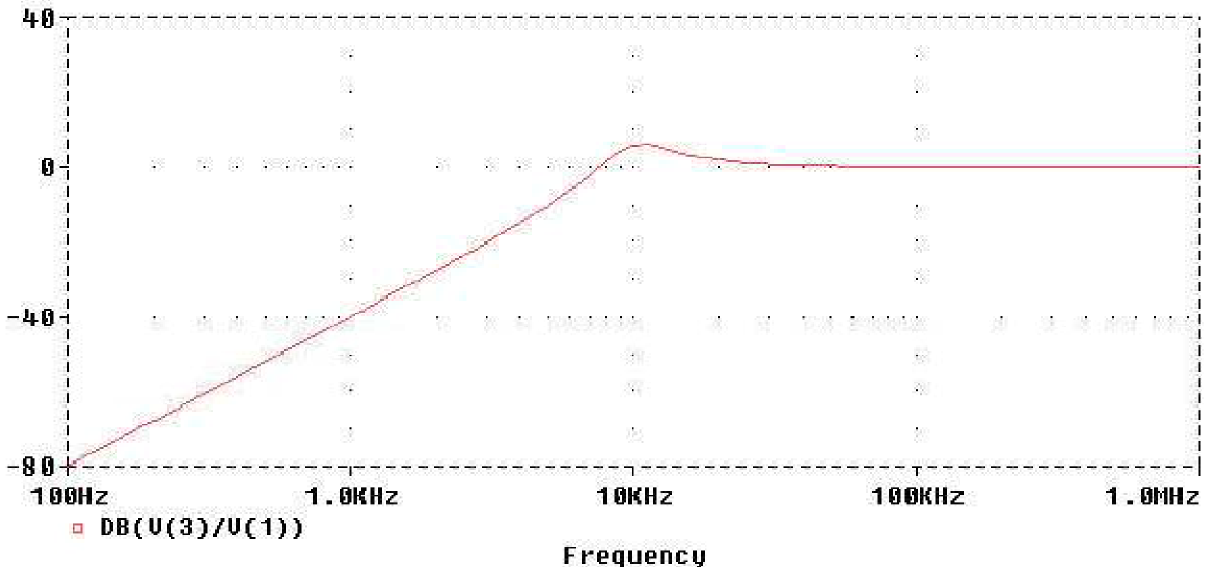


Here's a simple circuit for you to dive into running SPICE simulations and plotting results.

What is the purpose of this circuit? Basically it has two roles: to **pass** the desired high frequency signals and **stop** the unwanted low frequency signals.

```
SPICE FILE
Vin 1 0 AC 10V
Rf 1 2 4.0
CF 2 3 2.0uF
Lf 3 0 127uH
.AC DEC 20 100Hz 1MEG
.PROBE
.END
```

This time we did not use 1V for the input voltage. Therefore, we will need to have PROBE actually divide the input into the output to get the gain. We show this gain in decibels.



Notice that the gain below the resonant frequency of 10 kHz slopes upward at 40 dB/decade. When we plot the phase shift of this filter, we only need to specify the phase angle of the output voltage since the input voltage was specified at 0 degrees.

EXPERIMENT NO.03

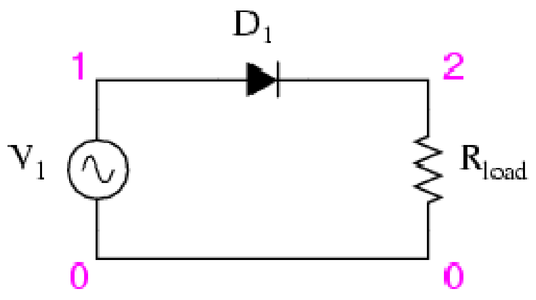
AIM: Verification of Half–Wave and Full-Wave Rectifier.

RECTIFIER

Before the development of silicon semiconductor rectifiers, vacuum tube diodes and copper(I) oxide or selenium rectifier stacks were used. High power rectifiers, such as are used in highvoltage
direct current power transmission, now uniformly employ silicon semiconductor devices
of various types. These are thyristors or other controlled switching solid-state switches which effectively function as diodes to pass current in only one direction.

HALF WAVE RECTIFIER

CIRCUIT:



In half wave rectification, either the positive or negative half of the AC wave is passed, while the other half is blocked. Because only one half of the input waveform reaches the output, it is very inefficient if used for power transfer. Half-wave rectification can be achieved with a single diode in a one-phase supply, or with three diodes in a three-phase supply. Half wave rectifiers yield a unidirectional but pulsating direct current.

SPICE FILE

```
V1 1 0 SIN(0 10V 100HZ)
R 1 2 1K
DA 0 2 D1
.MODEL D1 D
.TRAN 0.01MS 20MS
```

```
.PROBE
.END
```

MODEL WAVE FORM

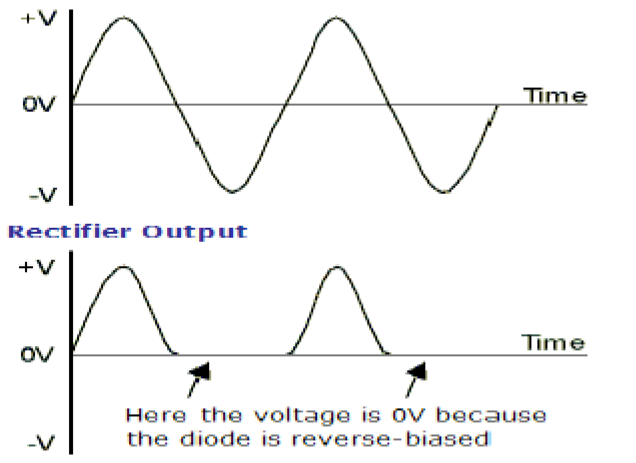
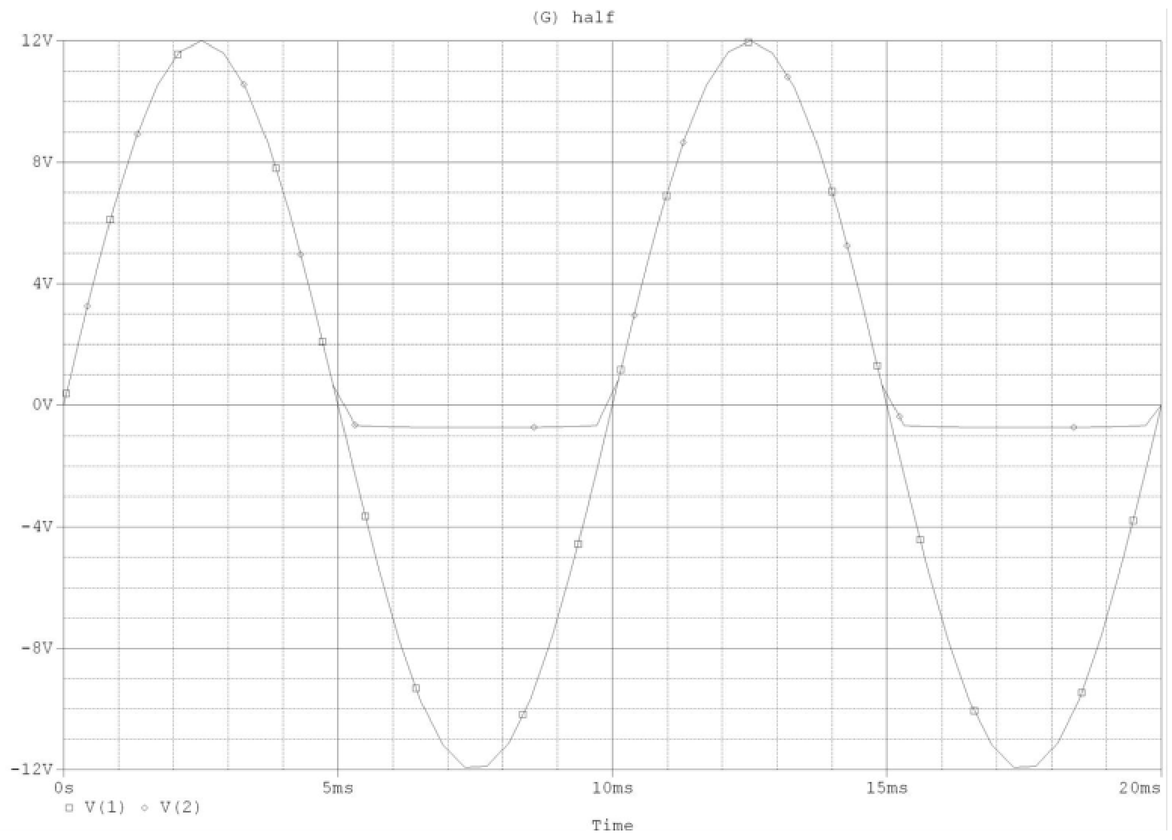


Figure : Half-wave rectification

RESULT:



FULL WAVE RECTIFIER

A full-wave rectifier converts the whole of the input waveform to one of constant polarity (positive or negative) at its output. Full-wave rectification converts both polarities of the input waveform to DC (direct current), and is more efficient.

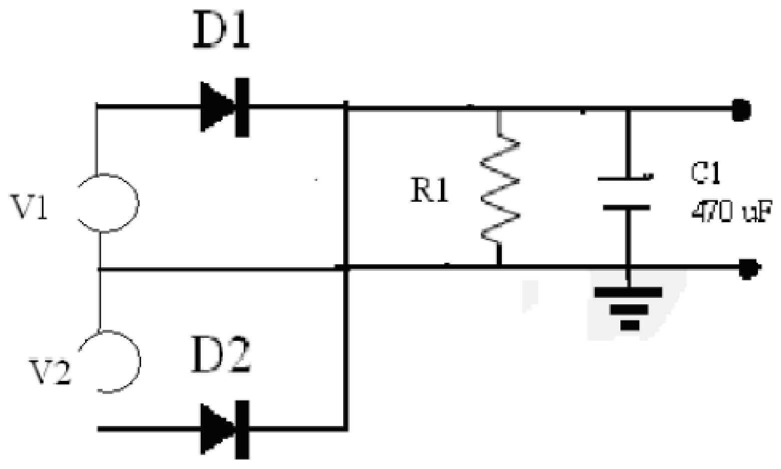


Figure : Full Wave Rectifier

SPICE FILE

```
V1 1 0 SIN(0 10V 100HZ)
R 2 3 1K
C 2 3 1N
D1 1 2 MOD1
D2 0 2 MOD1
.MODEL MOD1 D
.TRAN 0.01MS 20MS

.PROBE
.END
```

MODEL WAVEFORM:

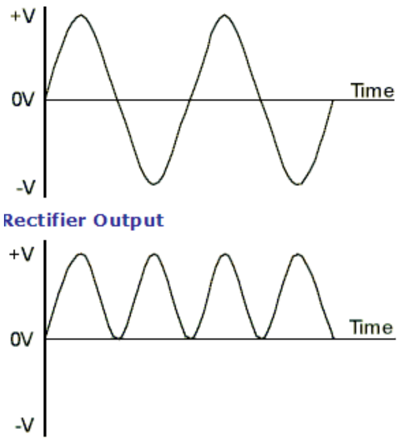
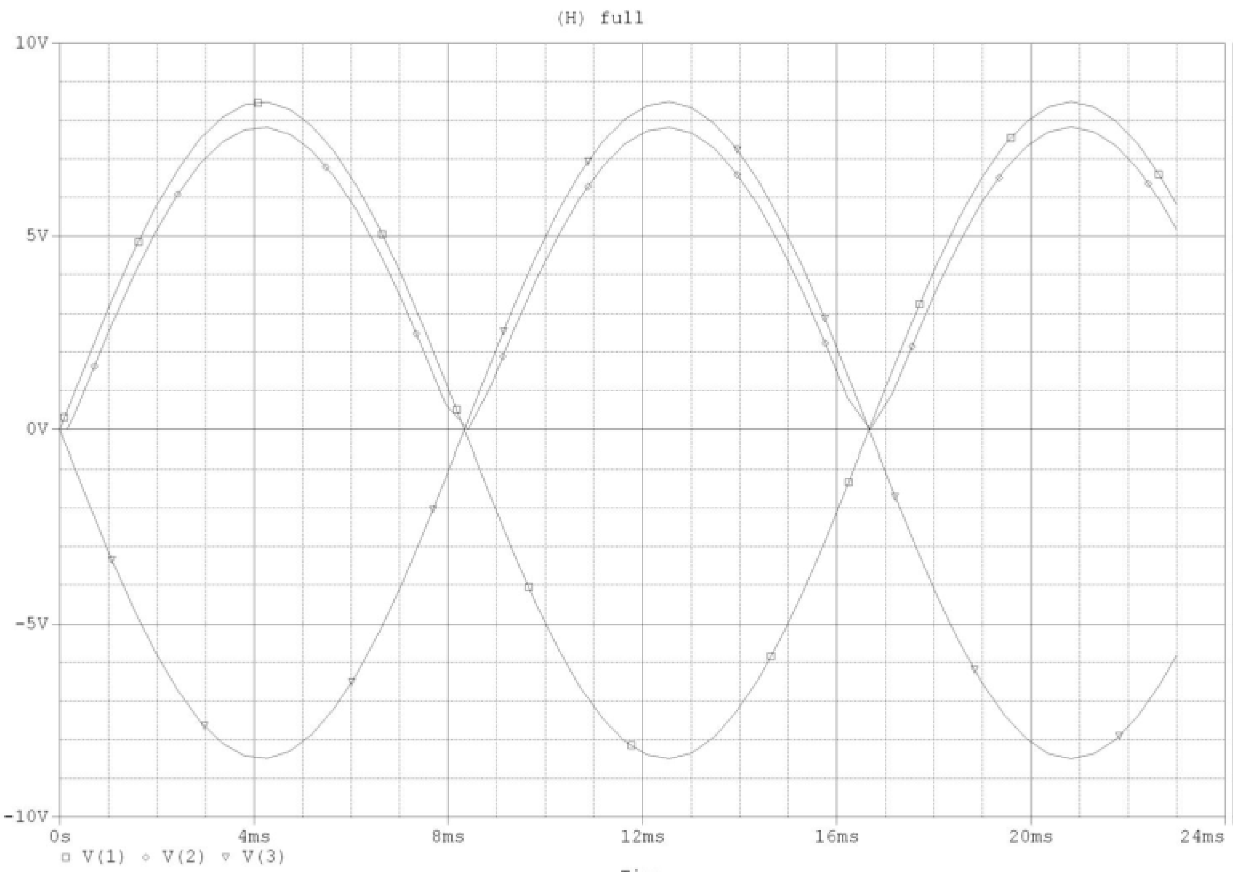


Figure : Full-wave rectification

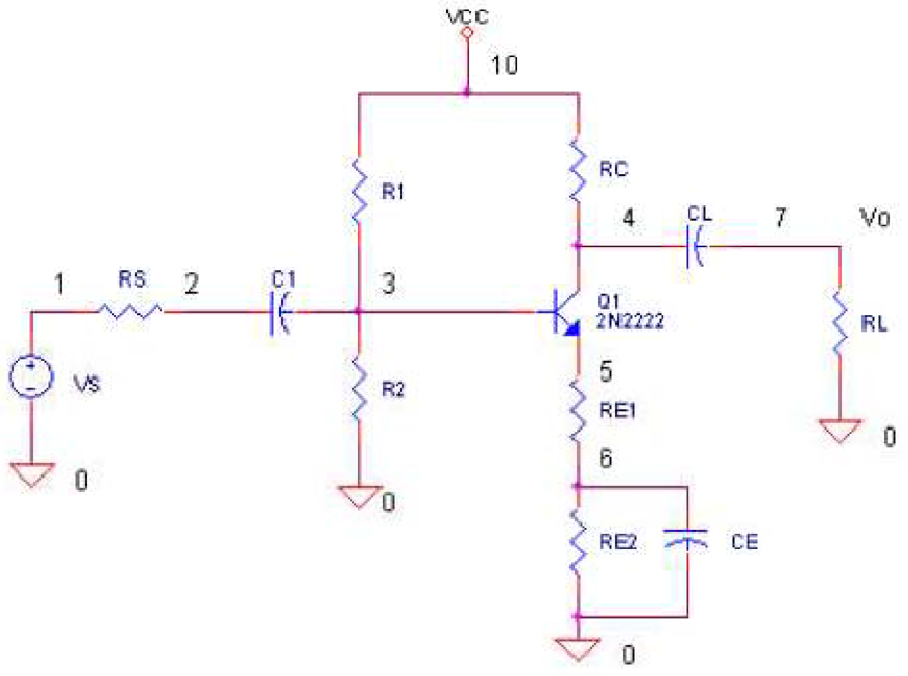
RESULT:



EXPERIMENT NO. 04

AIM: To verify the characteristics of CE Amplifier.

CIRCUIT:



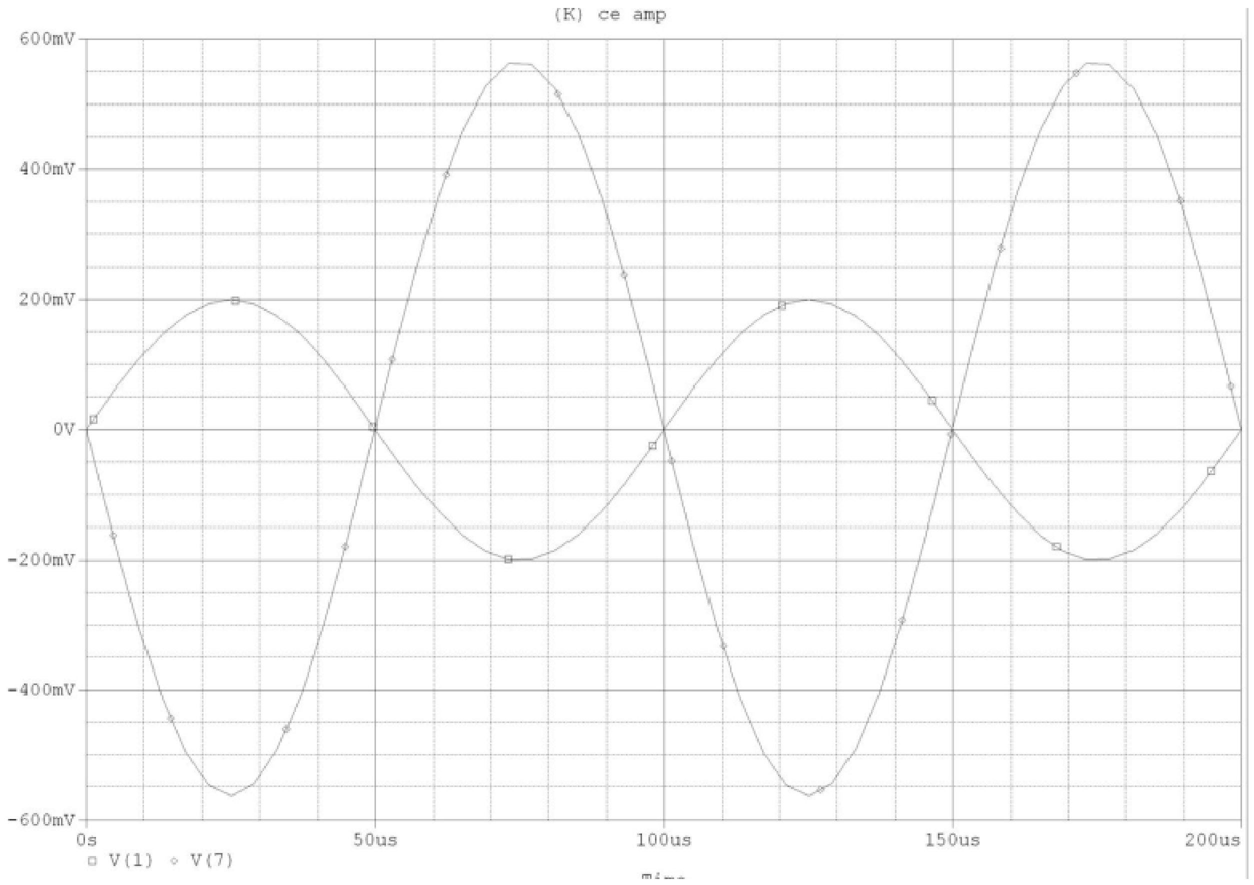
The COMMON-EMITTER CONFIGURATION (CE) is the most frequently used configuration in practical amplifier circuits, since it provides good voltage, current, and power gain. The input to the CE is applied to the base-emitter circuit and the output is taken from the collector-emitter circuit, making the emitter the element "common" to both input and output. The CE is set apart from the other configurations, because it is the only configuration that provides a phase reversal between input and output signals.

SPICE FILE

```
VIN 1 4 SIN(0 1.5V 2KHZ)
VB 4 0 2.3V
RL 3 0 15K
V1 2 0 15V
Q1 2 1 3 MOD1
```

```
.MODEL MOD1 NPN
.TRAN 0.02MS 0.78MS
.PROBE
.END
```

RESULT:



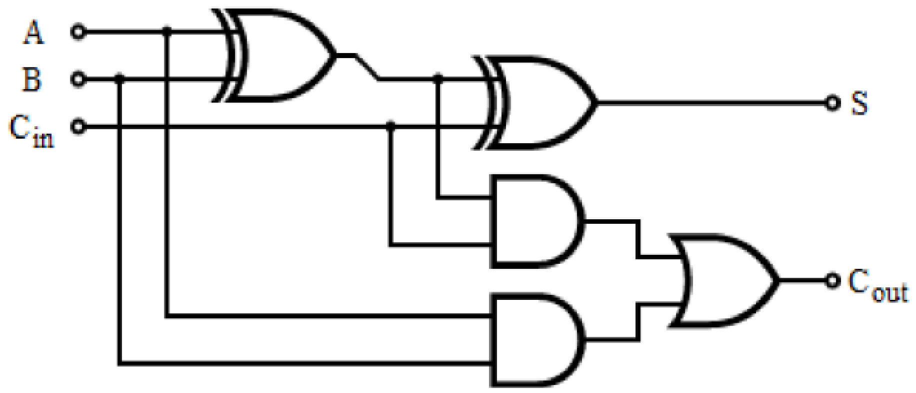
VHDL EXPERIMENTS

EXPERIMANT NO.01

AIM: Synthesis and Simulation of Full Adder.

SOFTWARE USED: Xilinx ISE 13.3i

BLOCK DIAGRAM



TRUTH TABLE

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**

VHDL CODE

```
library IEEE;

use IEEE.std_logic_1164.all;
```

entity Full_ Adder is

port (

A: in STD_LOGIC;

B : in STD_LOGIC;

C_{in} : in STD_LOGIC;

C_{out} : out STD_LOGIC;

S : out STD_LOGIC);

endFull_Adder ;

architecture behavioral of Full_Adder is

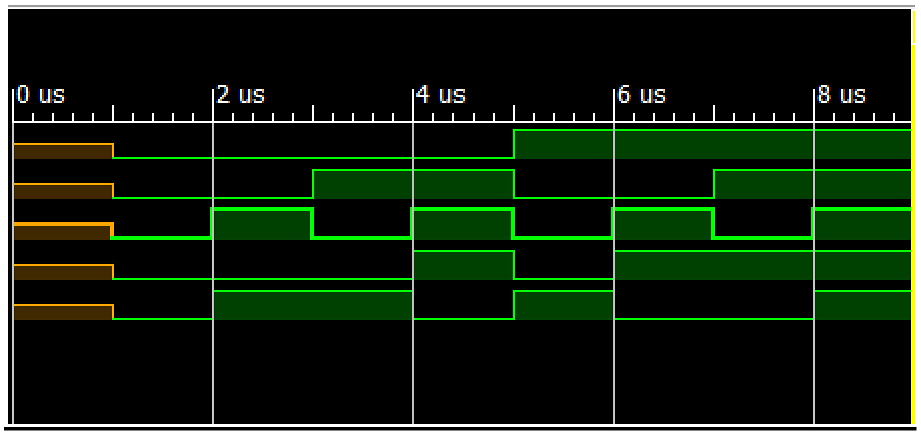
begin

s <= A xor B xorC_{in};

C_{out}<= (A and B) or ((A or B) and C_{in});

endbehavioural;

<u>WAVEFORM</u>

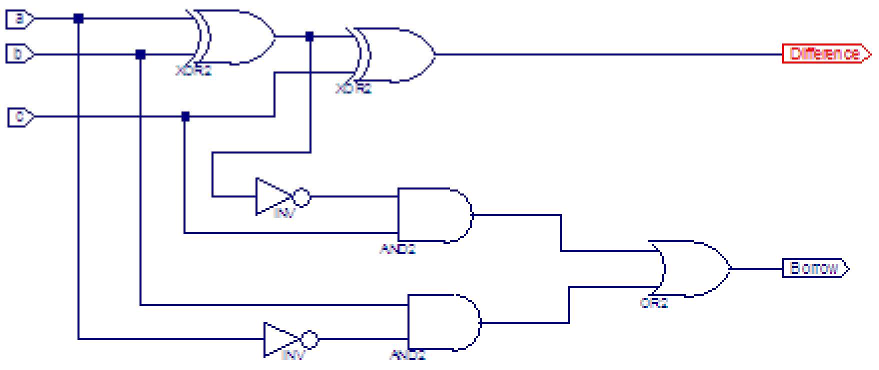


EXPERIMANT NO.02

AIM: Synthesis and Simulation of Full Subtractor.

SOFTWARE USED: Xilinx ISE 13.3i

BLOCK DIAGRAM



TRUTH TABLE

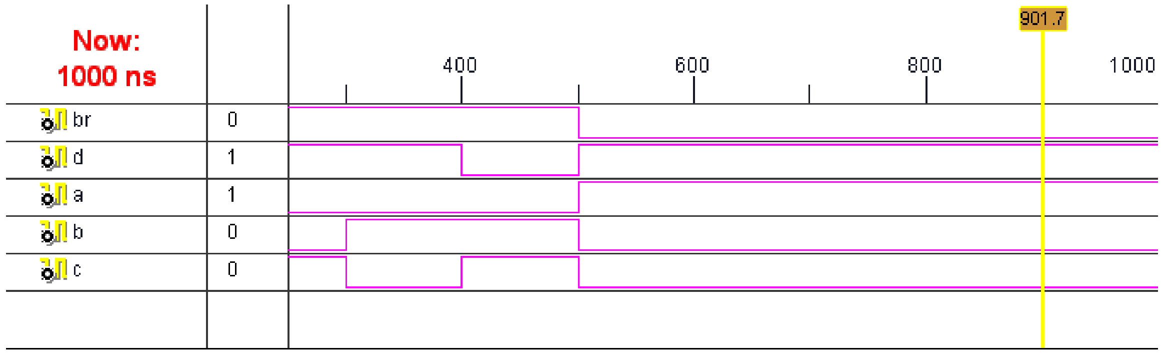
Input			Output	
A	B	C	Difference	Borrow
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

VHDL CODE

```
libraryieee;
use ieee.std_logic_1164.all;
entityflsub_select is
port(a:inbit_vector(2 downto 0);
s:out bit_vector(1 downto 0));
endflsub_select;
architecturebeh of flsub_select is
begin
with a select
```

```
s<=("00") when "000",
("11") when "001",
("11") when "010",
("01") when "011",
("10") when "100",
("00") when "101",
("00") when "110",
("11") when "111";
endbeh;
```

WAVEFORM

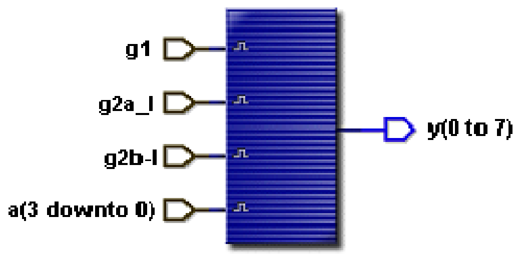


EXPERIMANT NO.03

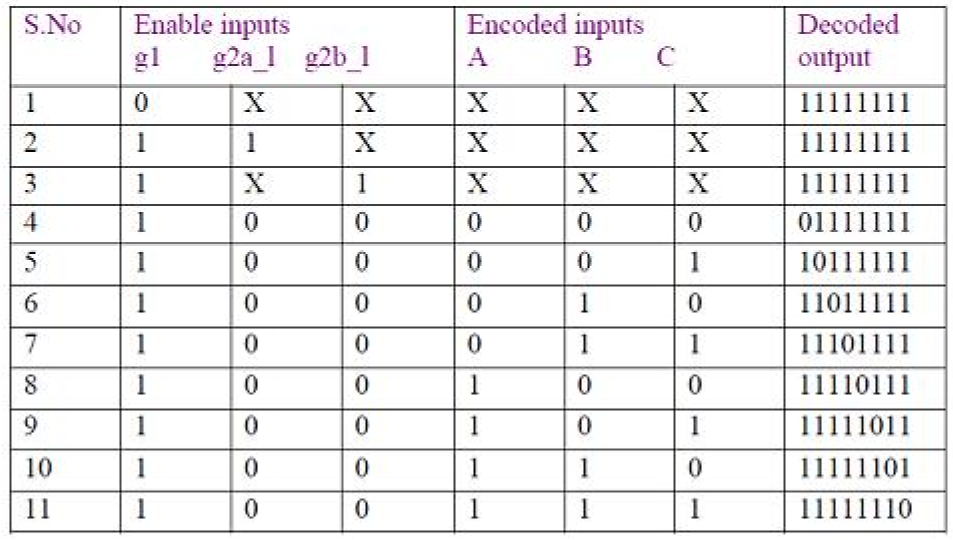
AIM: Synthesis and Simulation of 3 X 8 Decoder.

SOFTWARE USED: Xilinx ISE 13.3i

BLOCK DIAGRAM:



TRUTH TABLE:

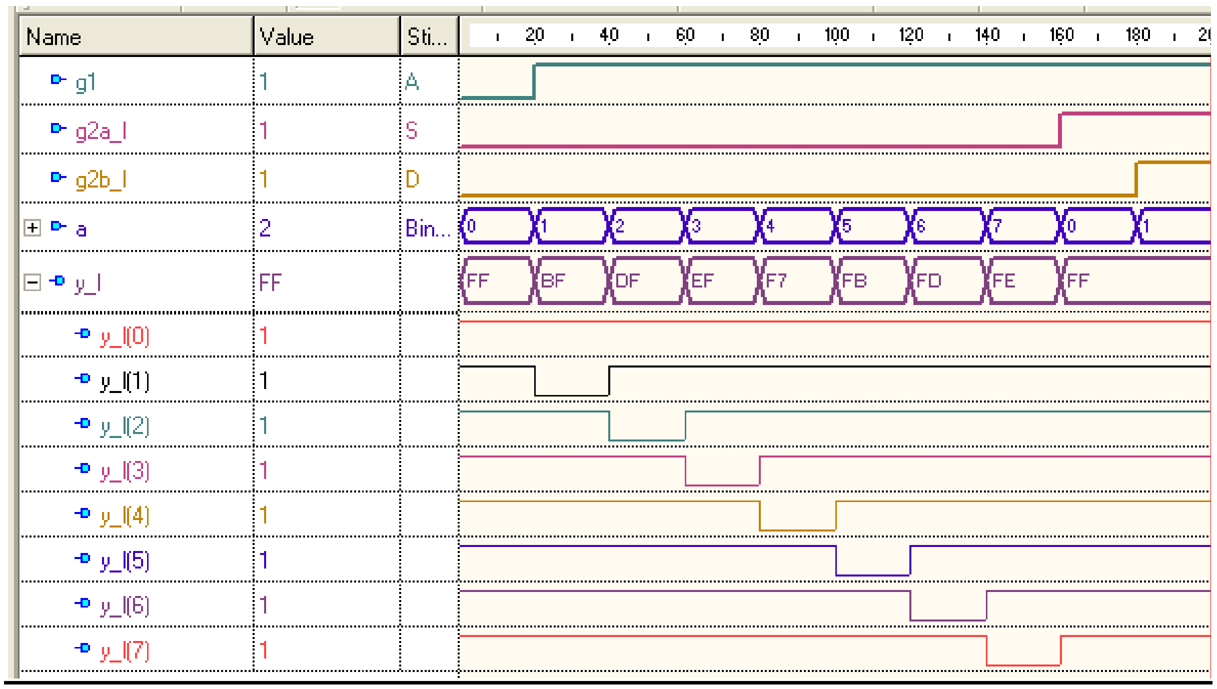


S.No	Enable inputs			Encoded inputs			Decoded output
	g1	g2a_l	g2b_l	A	B	C	
1	0	X	X	X	X	X	11111111
2	1	1	X	X	X	X	11111111
3	1	X	1	X	X	X	11111111
4	1	0	0	0	0	0	01111111
5	1	0	0	0	0	1	10111111
6	1	0	0	0	1	0	11011111
7	1	0	0	0	1	1	11101111
8	1	0	0	1	0	0	11110111
9	1	0	0	1	0	1	11111011
10	1	0	0	1	1	0	11111101
11	1	0	0	1	1	1	11111110

VHDL CODE:

```
library IEEE;

use IEEE.std_logic_1164.all;

entity decoder3X8 is

port (

g1 : in STD_LOGIC;--g1, g2a_l, g2b_l cascade i/ps

g2a_l : in STD_LOGIC;
g2b_l : in STD_LOGIC;

a : in STD_LOGIC_VECTOR (2 downto 0);

y_l : out STD_LOGIC_VECTOR (0 to 7)

);

end decoder3X8;

architecture deco38 of decoder3X8 is

begin

process (a,g1,g2a_l,g2b_l)

begin

if (g1 and not g2a_l and not g2b_l)='1'then

if a <= "000"then y_l<= "01111111";

elsif a <= "001"then y_l<= "10111111";

elsif a <= "010"then y_l<= "11011111";

elsif a <= "011"then y_l<= "11101111";

elsif a <= "100"then y_l<= "11110111";

elsif a <= "101"then y_l<= "11111011";

elsif a <= "110"then y_l<= "11111101";

elsif a <= "111"then y_l<= "11111110";

else y_ l<= "11111111";

end if;

elsey_l<= "11111111";

end if;

end process;

end deco38;
```

WAVEFORMS



VIVA QUESTIONS

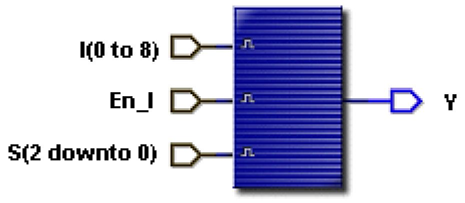
1. Write the behavioral code for the IC 74x138.
2. Write the VHDL code for the IC 74x138 using CASE statement.
3. Write the VHDL code for the IC 74x138 using WITH statement.
4. Write the VHDL code for the IC 74x138 using WHEN--ELSE statement.
5. Write the structural program for IC 74x138.
6. What does priority encoder mean?
7. How many decoders are needed to construct 4X16 decoder?
8. What is the difference between decoder and encoder?
9. Write the syntax for exit statement?
10. Explain briefly about next statement?
11. How to specify the delay in VHDL program?
12. Write the syntax for component declaration.

EXPERIMANT NO.04

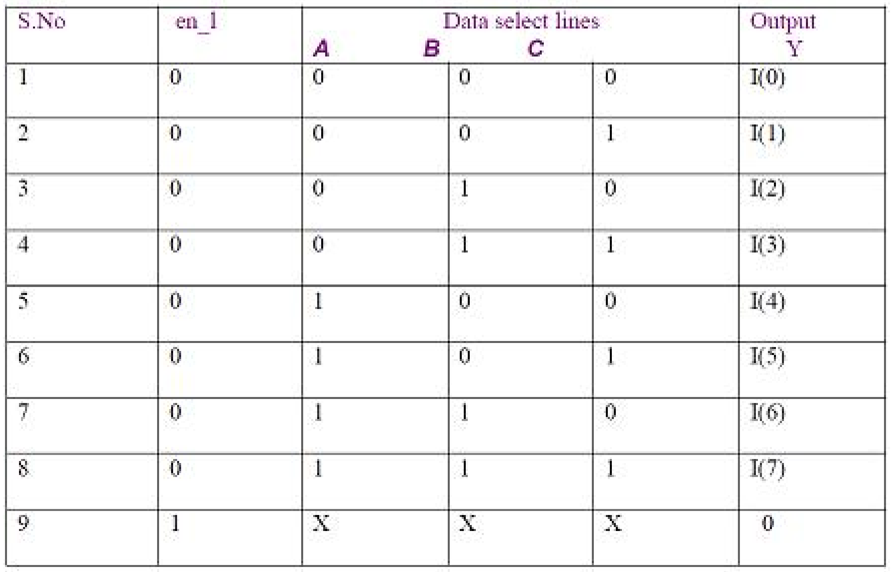
AIM: Write a VHDL code for IC74151—8x1 multiplexer.

SOFTWARE USED: Xilinx ISE 13.3i

BLOCK DIAGRAM



TRUTH TABLE

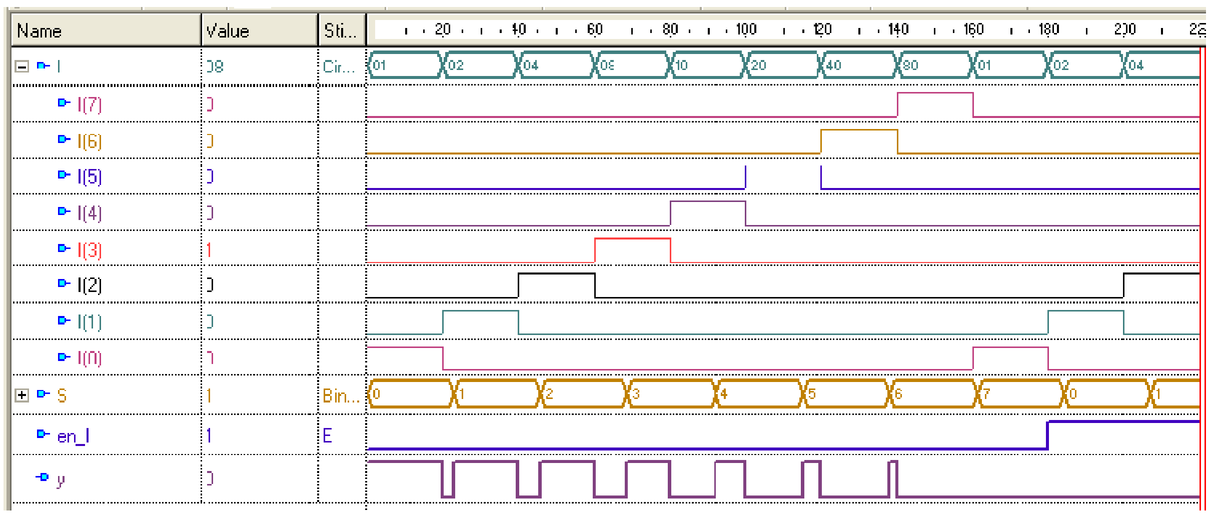


S.No	en_1	Data select lines			Output
		A	B	C	Y
1	0	0	0	0	I(0)
2	0	0	0	1	I(1)
3	0	0	1	0	I(2)
4	0	0	1	1	I(3)
5	0	1	0	0	I(4)
6	0	1	0	1	I(5)
7	0	1	1	0	I(6)
8	0	1	1	1	I(7)
9	1	X	X	X	0

VHDL CODE

```
library IEEE;

use IEEE.std_logic_1164.all;

entity mux151 is

port (

I :in STD_LOGIC_VECTOR (7 downto 0); --8 i/p lines

S :in STD_LOGIC_VECTOR (2 downto 0); --3 data select lines
en_l:in STD_LOGIC; --active low enable i/p

y :out STD_LOGIC --output line

);

end mux151;

architecture mux151 of mux151 is

begin

process (I,s,en_l)

begin

ifen_l='0' then

case s is

when "000" => y <= I(0);

when "001" => y <= I(1);

when "010" => y <= I(2);

when "011" => y <= I(3);

when "100" => y <= I(4);

when "101" => y <= I(5);

when "110" => y <= I(6);

when "111" => y <= I(7);

when others=>null;

end case;

else y <= '0'; --y=0 when en_l=1

end if;

end process;

end mux151;
```

WAVEFORM



VIVA QUESTIONS

1. Write the behavioral code for the IC 74x151.
2. Write the VHDL code for the IC 74x151 using IF statement.
3. Write the VHDL code for the IC 74x151 using WITH statement.
4. Write the VHDL code for the IC 74x151 using WHEN--ELSE statement.
5. Write the structural program for IC 74x151.
6. What is meant by multiplexer?
7. What does demultiplexer mean?
8. How many 8X1 multiplexers are needed to construct 16X1 multiplexer?
9. Compare decoder with demultiplexer?
10. Design a full adder using 8X1 multiplexer?
11. What are the two kinds of subprograms?
12. What are the difference between function and procedure?
13. Explain briefly about subprogram overloading?