Index

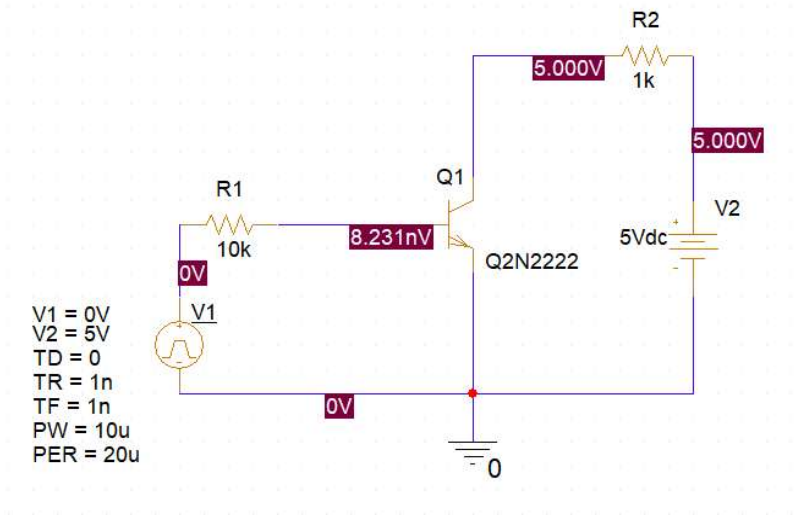
Experiment 1

Aim: (a) Transient Analysis of BJT inverter using step input. (b) DC Analysis (VTC) of BJT inverter.

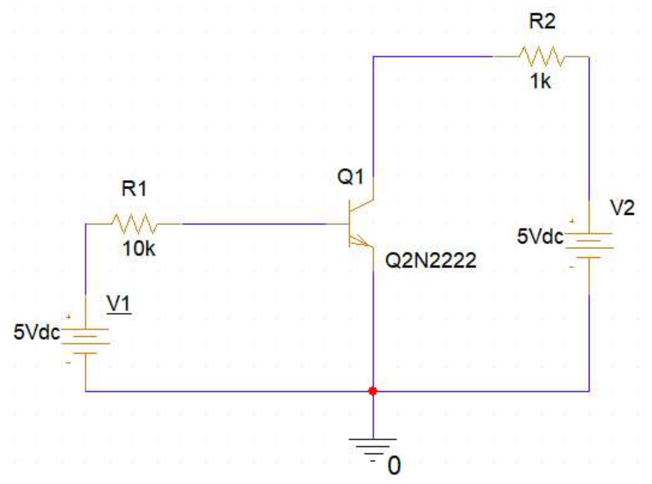
Software Required: Capture CIS

Schematic:

(a)

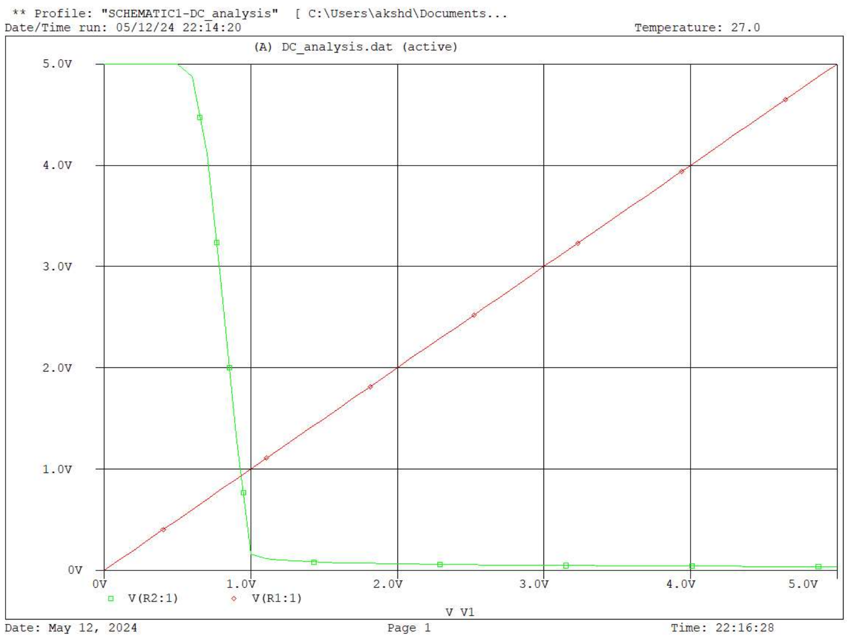


(b)

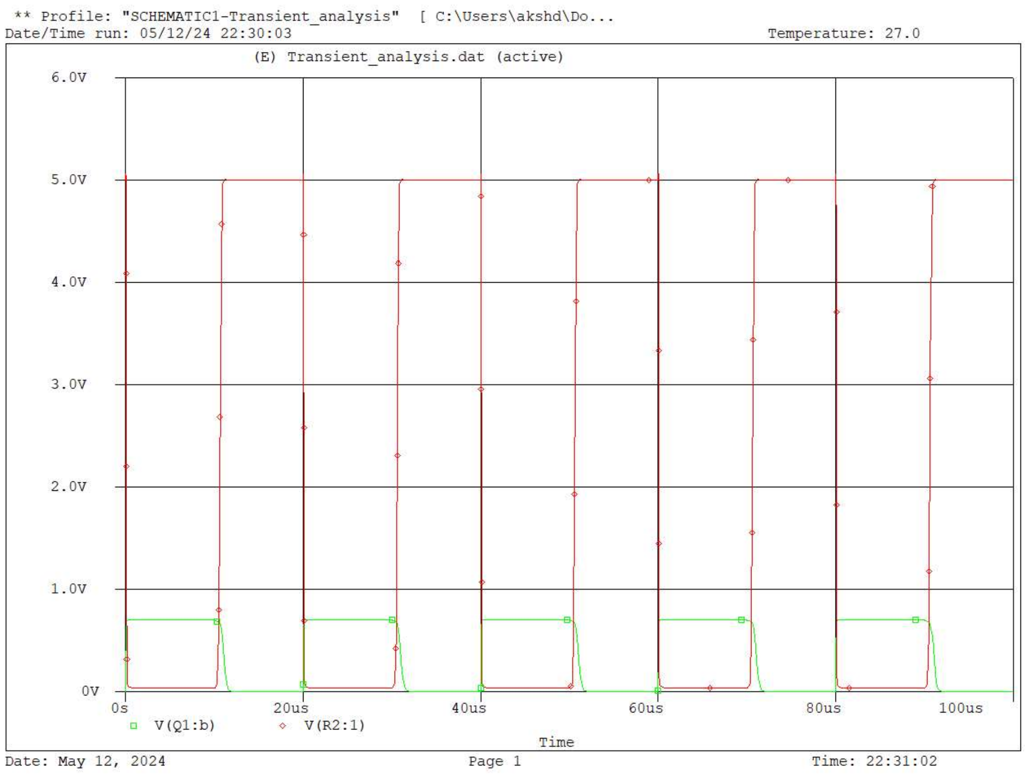


Outputs:

(a)



(b)



Result: Transient and DC analysis of BJT inverter was performed and output waveforms were observed.

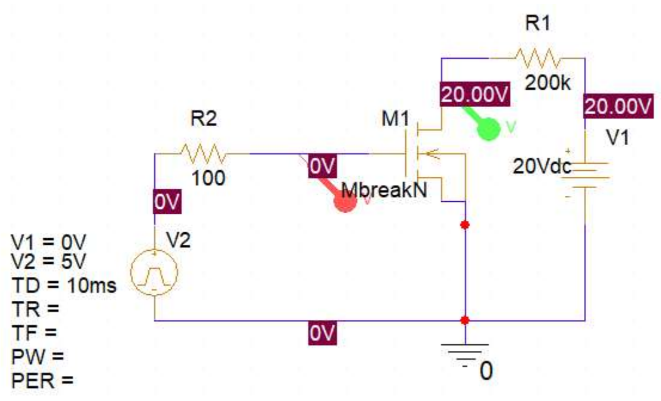
Experiment 2

Aim: (a) Transient Analysis of NMOS inverter using step input. (b) Transient Analysis of NMOS inverter using pulse input. (c) DC Analysis (VTC) of NMOS inverter.

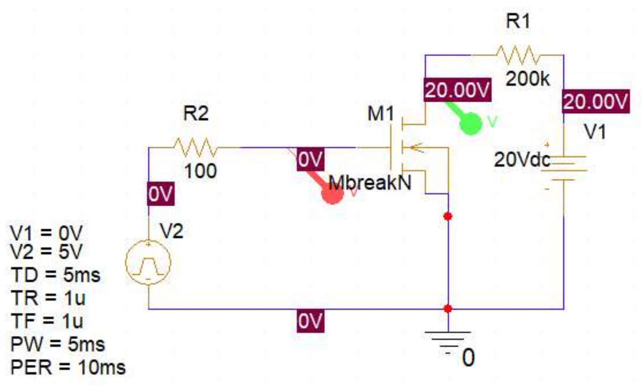
Software Required: Capture CIS

Schematic:

(a)

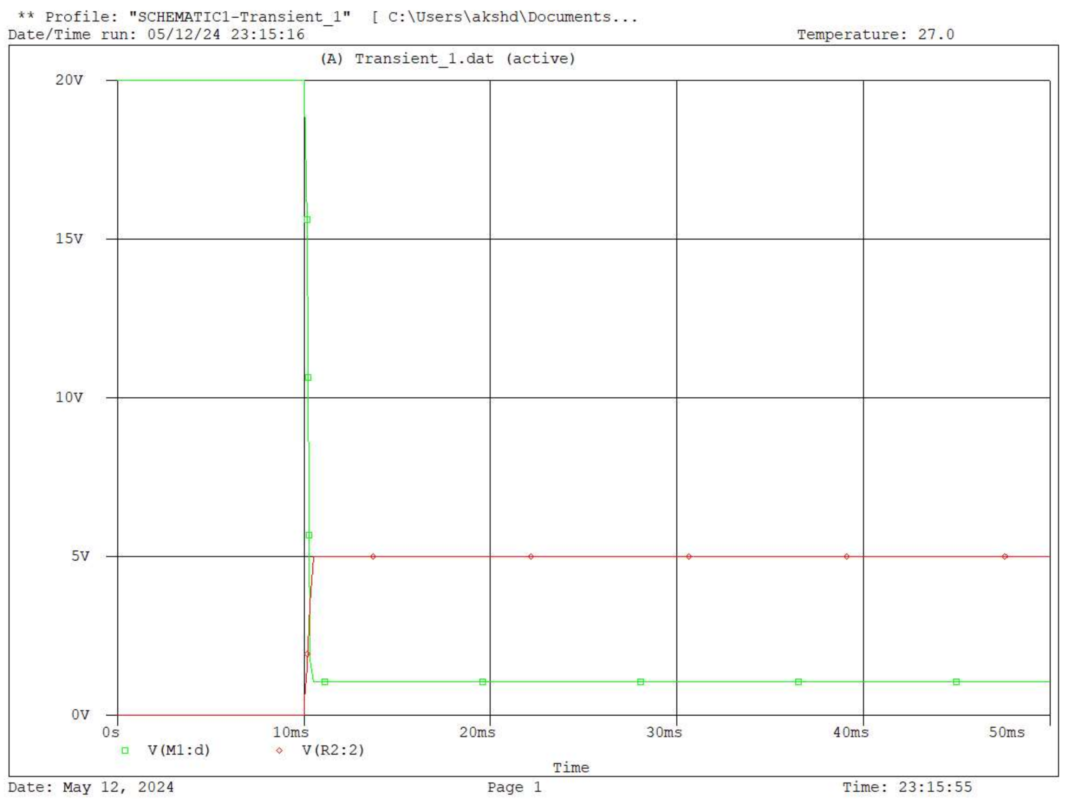


(b)

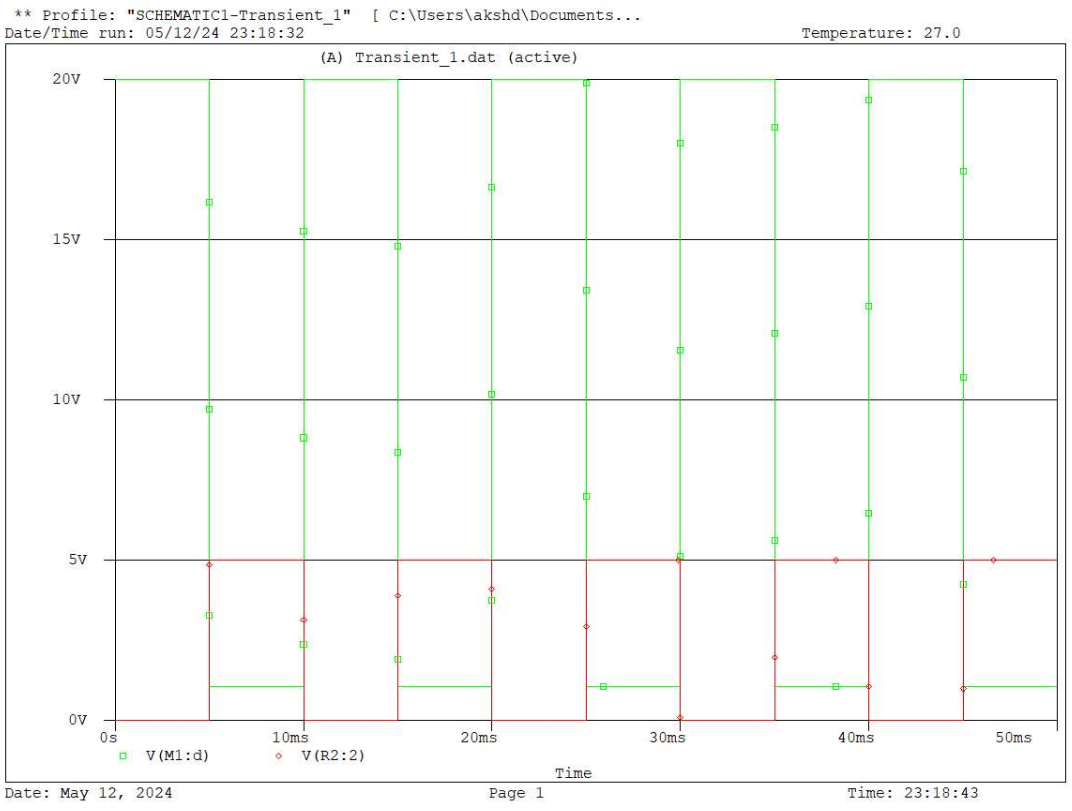


Outputs:

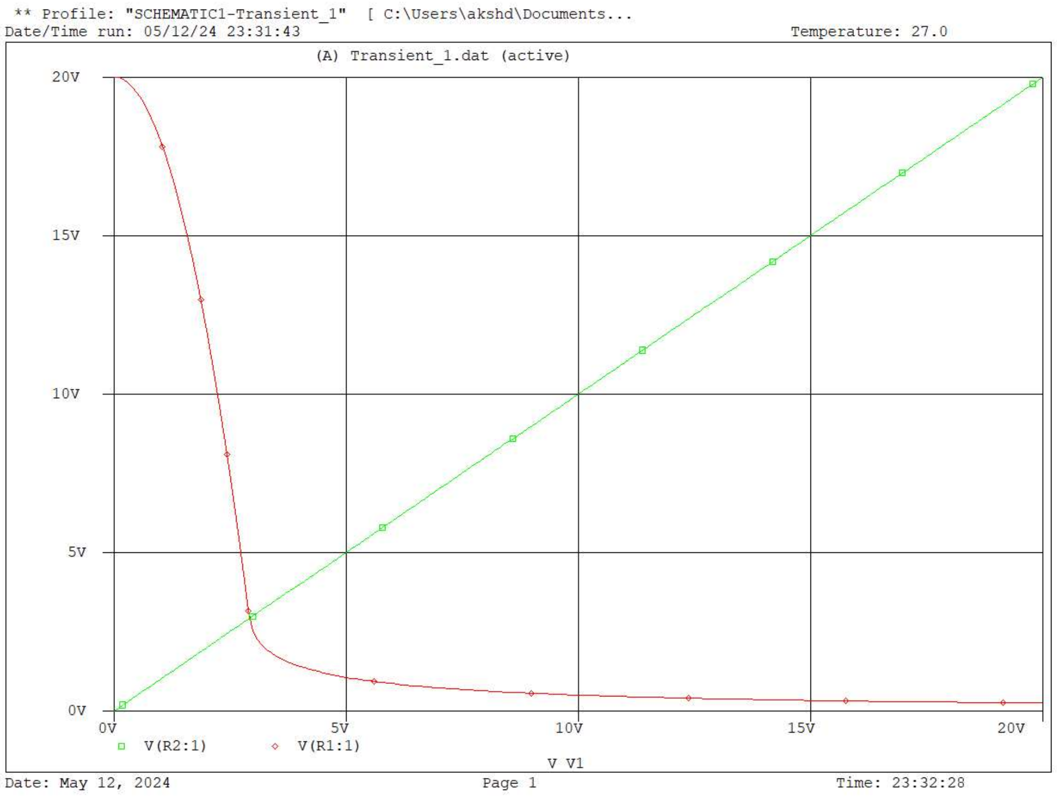
(a)



(b)



(c)



Result: Transient analysis of NMOS inverter using step and pulse input.

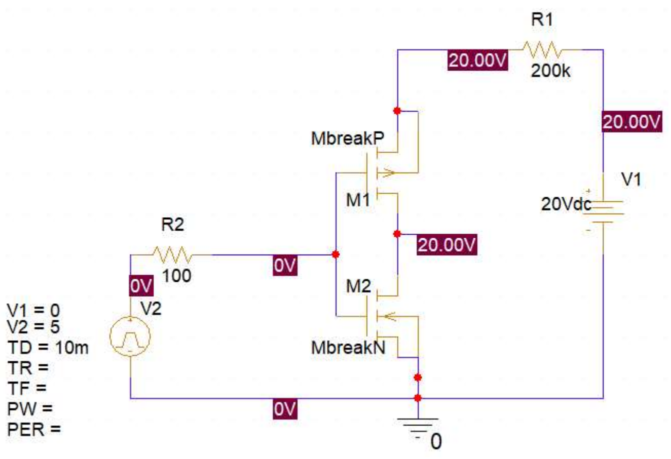
Experiment 3

Aim: (a) Analysis of CMOS inverter using step input. (b) Transient Analysis of CMOS inverter using step input with parameters. (c) Transient Analysis of CMOS inverter using pulse input. (d) Transient Analysis of CMOS inverter using pulse input with parameters. (e) DC Analysis (VTC) of CMOS inverter with and without parameters.

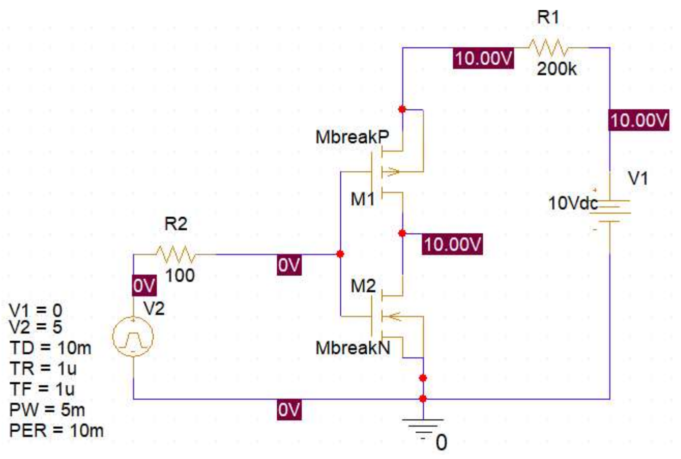
Software Required: Capture CIS

Schematic:

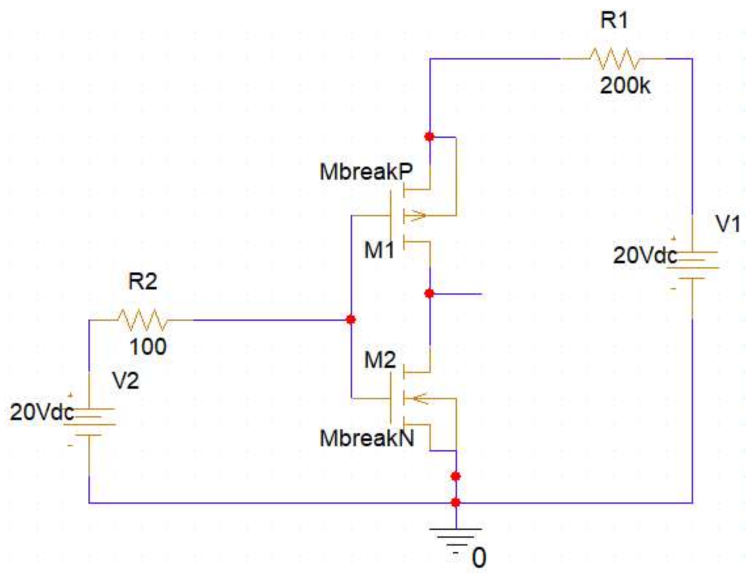
(a) & (b)



(c) & (d)

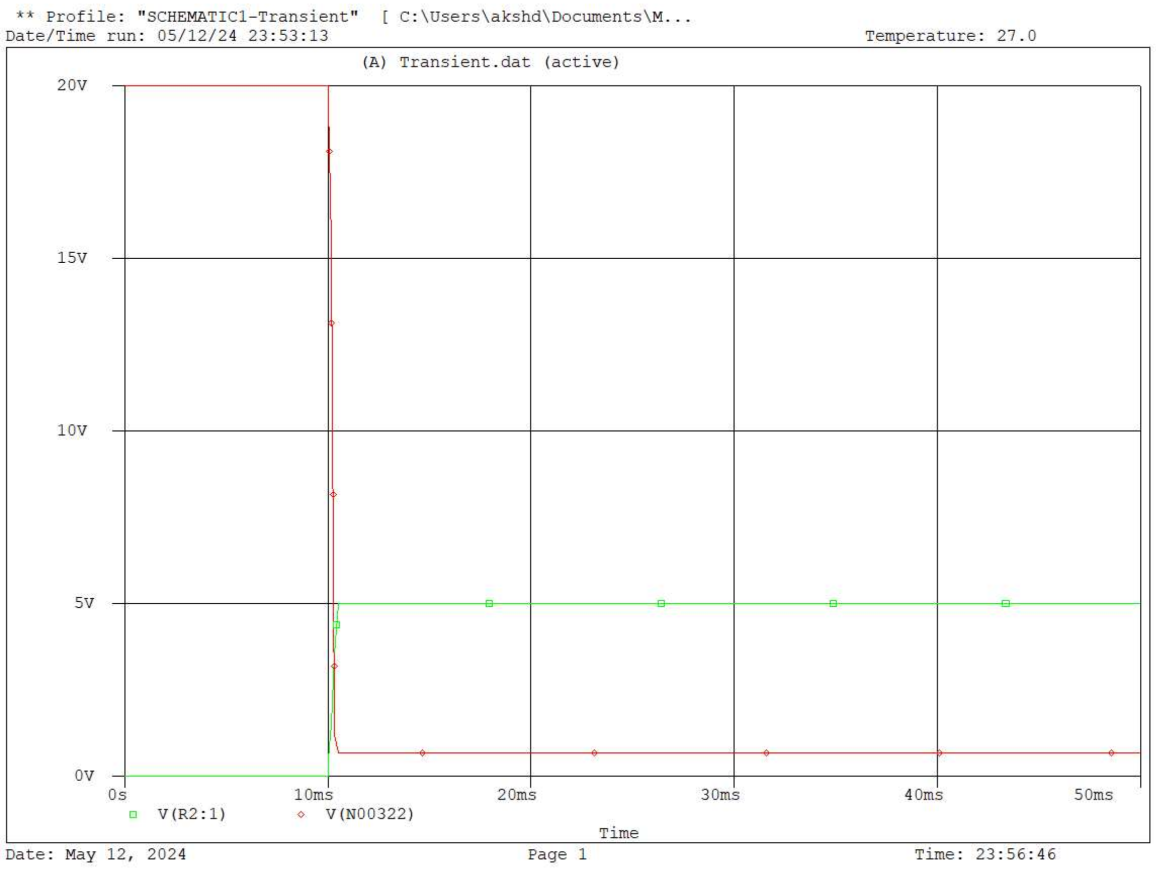


(e)



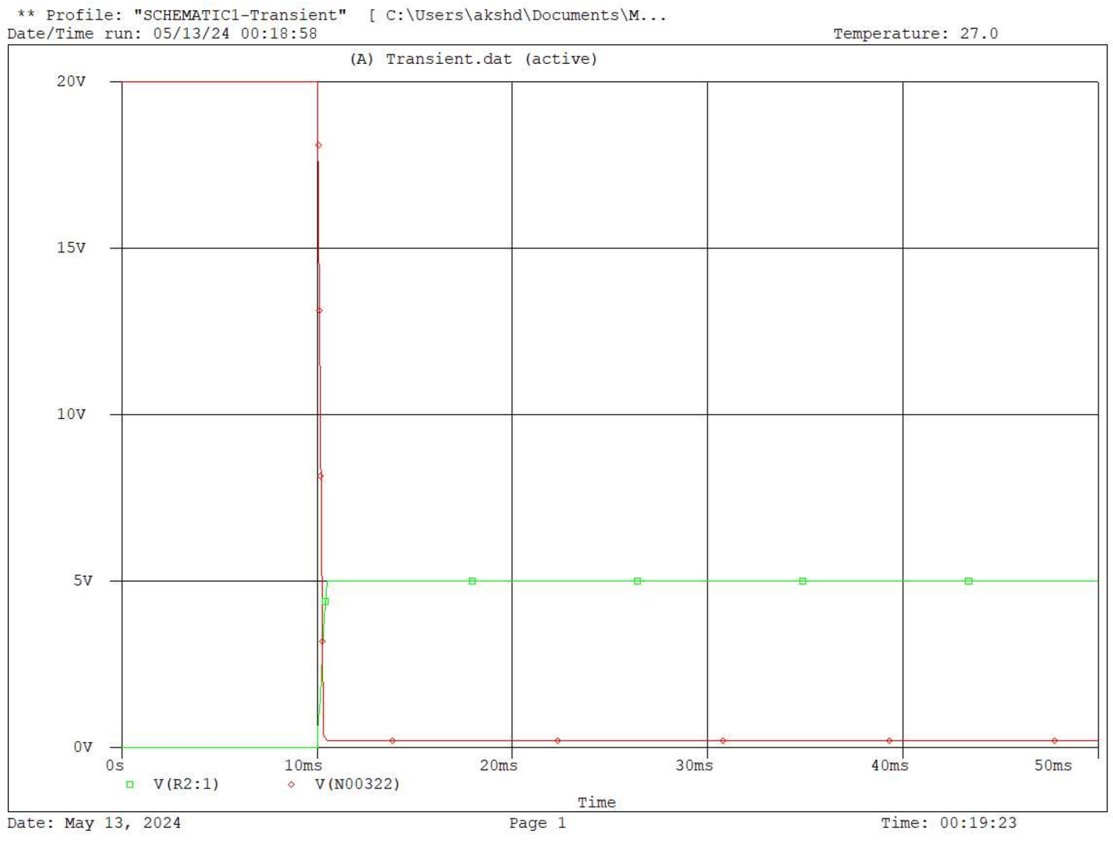
Outputs:

(a)



** Profile: "SCHEMATIC1-Transient" [C:\Users\akshd\Documents\M...
Date/Time run: 05/12/24 23:53:13
Temperature: 27.0
(A) Transient.dat (active)
20V
15V
10V
5V
0V
0s
10ms
20ms
30ms
40ms
50ms
V(R2:1)
V(N00322)
Time
Date: May 12, 2024
Page 1
Time: 23:56:46

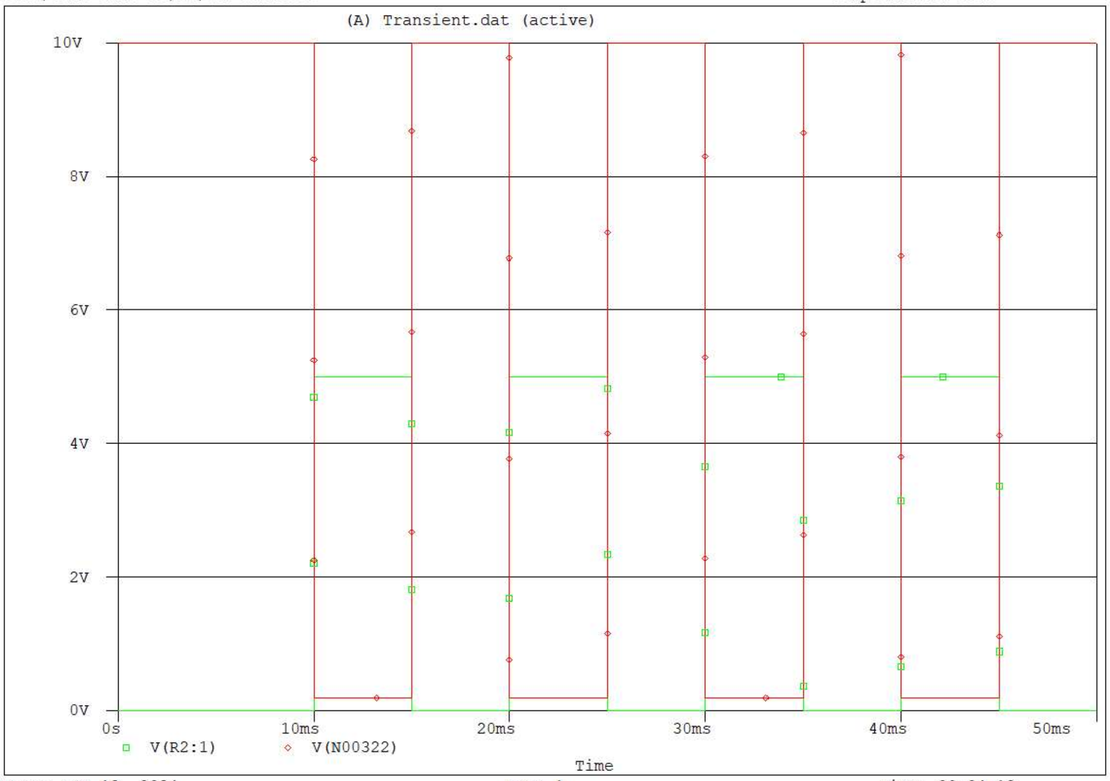
(b)



** Profile: "SCHEMATIC1-Transient" [C:\Users\akshd\Documents\M...
Date/Time run: 05/13/24 00:18:58
Temperature: 27.0
(A) Transient.dat (active)
20V
15V
10V
5V
0V
0s
10ms
20ms
30ms
40ms
50ms
V(R2:1)
V(N00322)
Time
Date: May 13, 2024
Page 1
Time: 00:19:23

(c)

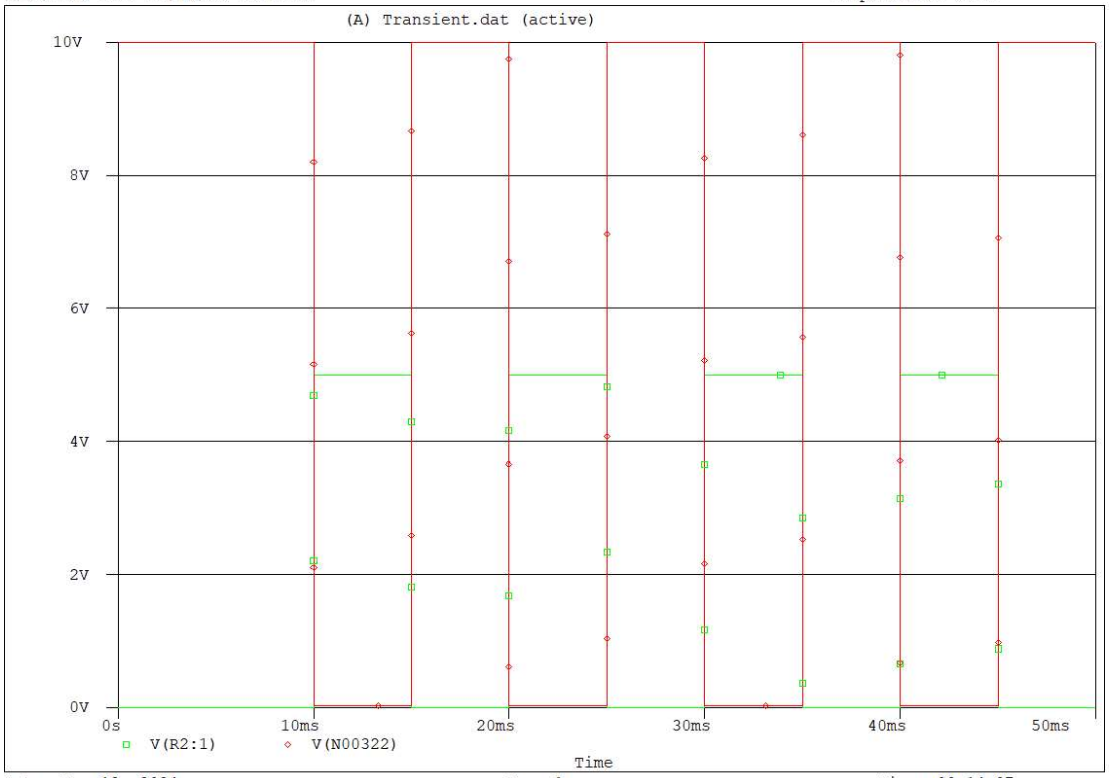
```
 ** Profile: "SCHEMATIC1-Transient"  [ C:\Users\akshd\Documents\M...
Date/Time run: 05/13/24 00:01:57                                        Temperature: 27.0
```



```
Date: May 13, 2024                    Page 1                    Time: 00:04:13
```

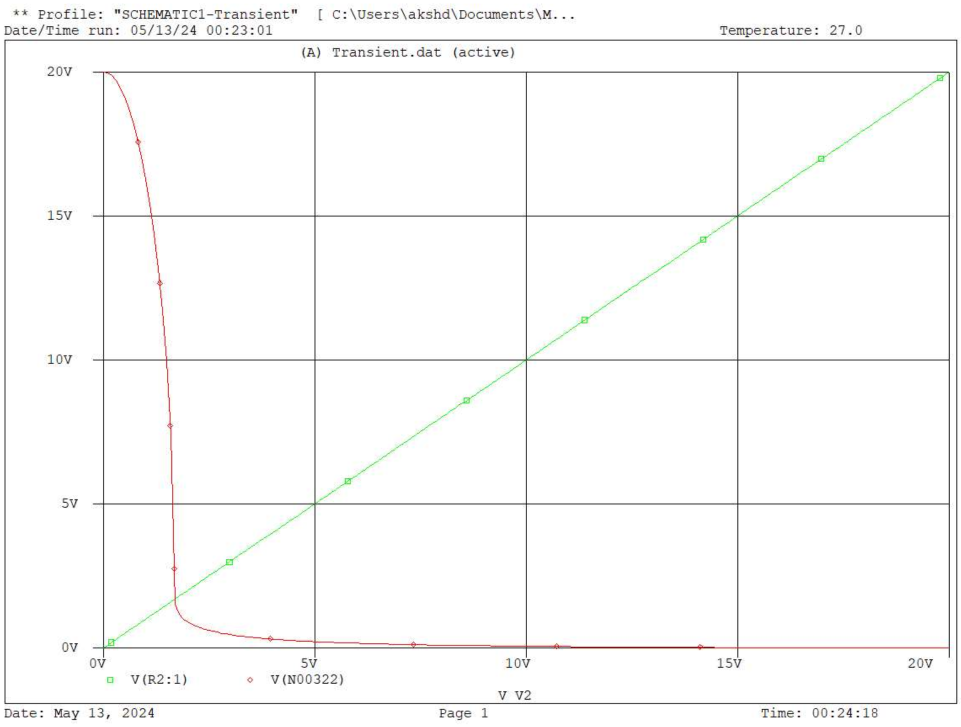
(d)

```
 ** Profile: "SCHEMATIC1-Transient"  [ C:\Users\akshd\Documents\M...
Date/Time run: 05/13/24 00:10:15                                        Temperature: 27.0
```



```
Date: May 13, 2024                    Page 1                    Time: 00:14:07
```

(e)



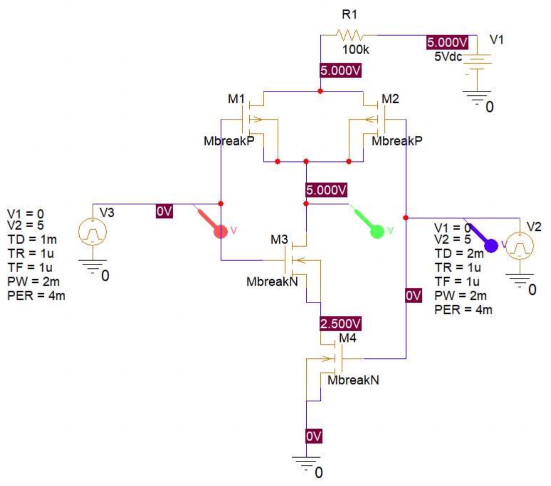
Result: Outputs for all the objectives were plotted and observed.

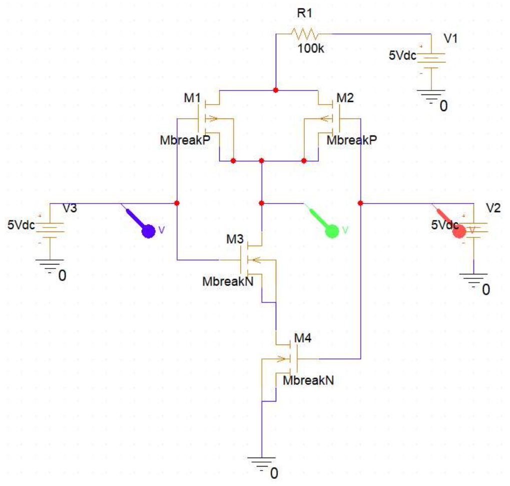
Experiment 4

Aim: Transient & DC Analysis of NAND Gate using CMOS inverter.

Software Required: Capture CIS

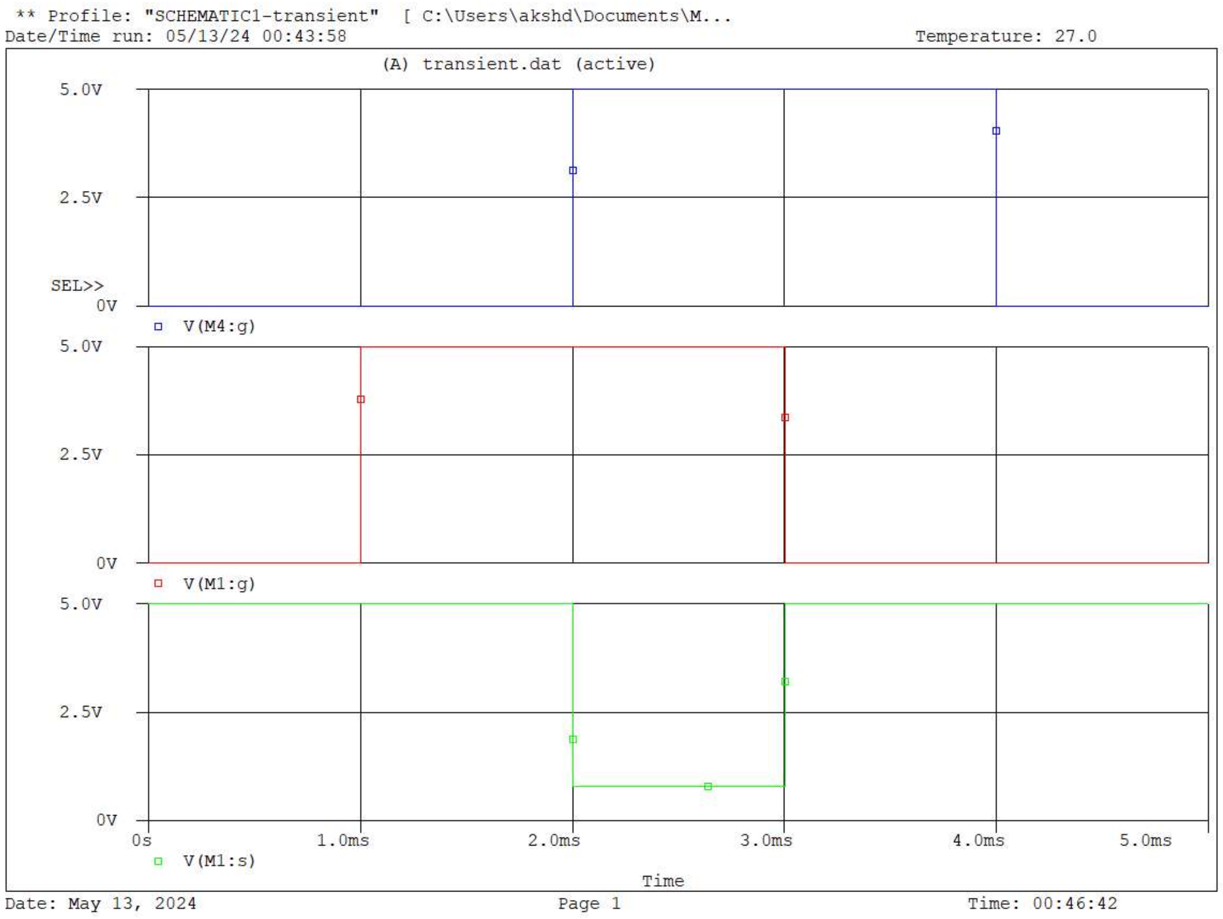
Schematic:



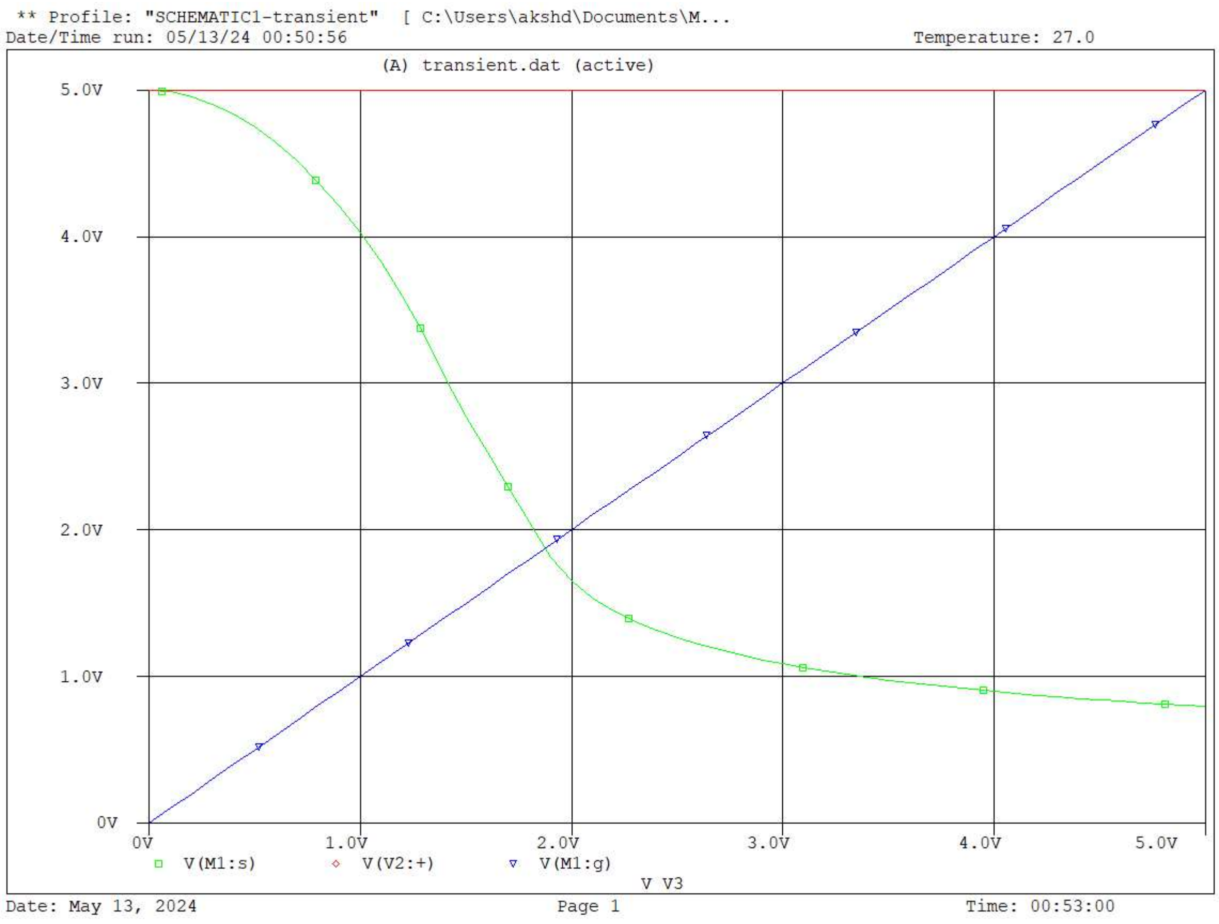


Output:

Transient



DC



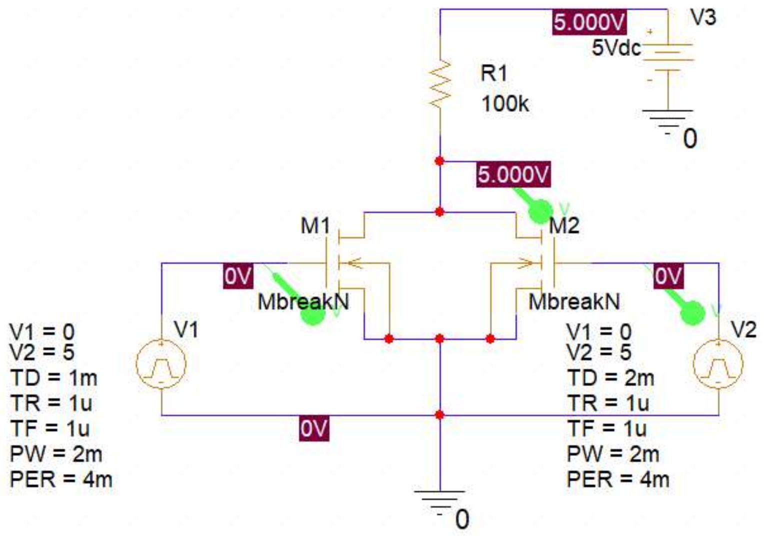
Result: Transient and DC analysis of NAND gate using CMOS inverter were performed and output waveforms were observed.

Experiment 5

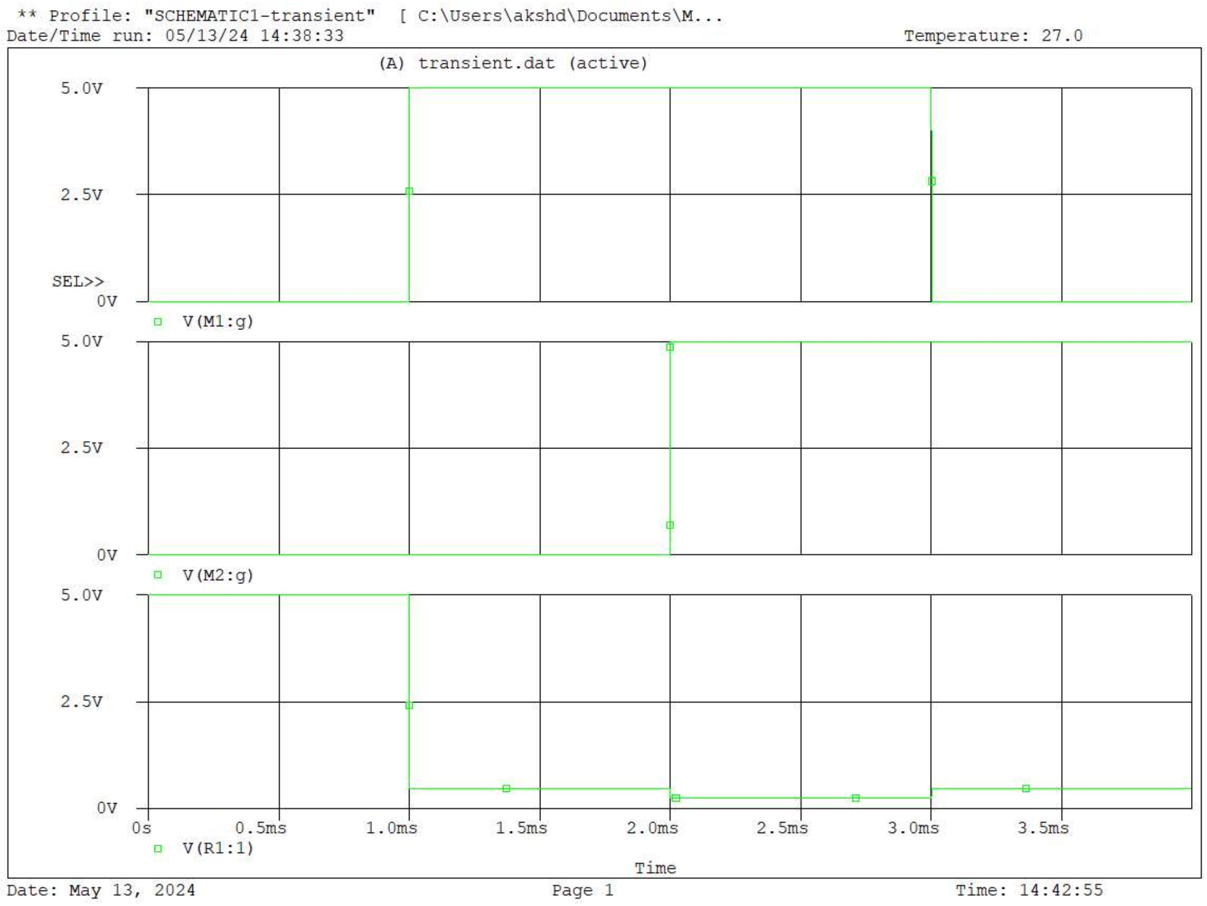
Aim: Transient Analysis of NOR Gate inverter and implementation of XOR gate using NOR gate

Software Required: Capture CIS

Schematic:



Outputs:


```
** Profile: "SCHEMATIC1-transient"  [ C:\Users\akshd\Documents\M...
Date/Time run: 05/13/24 14:38:33                                        Temperature: 27.0
```

```
Date: May 13, 2024                    Page 1                    Time: 14:42:55
```

Result: Transient analysis of NOR gate was performed and observed.

Experiment 6

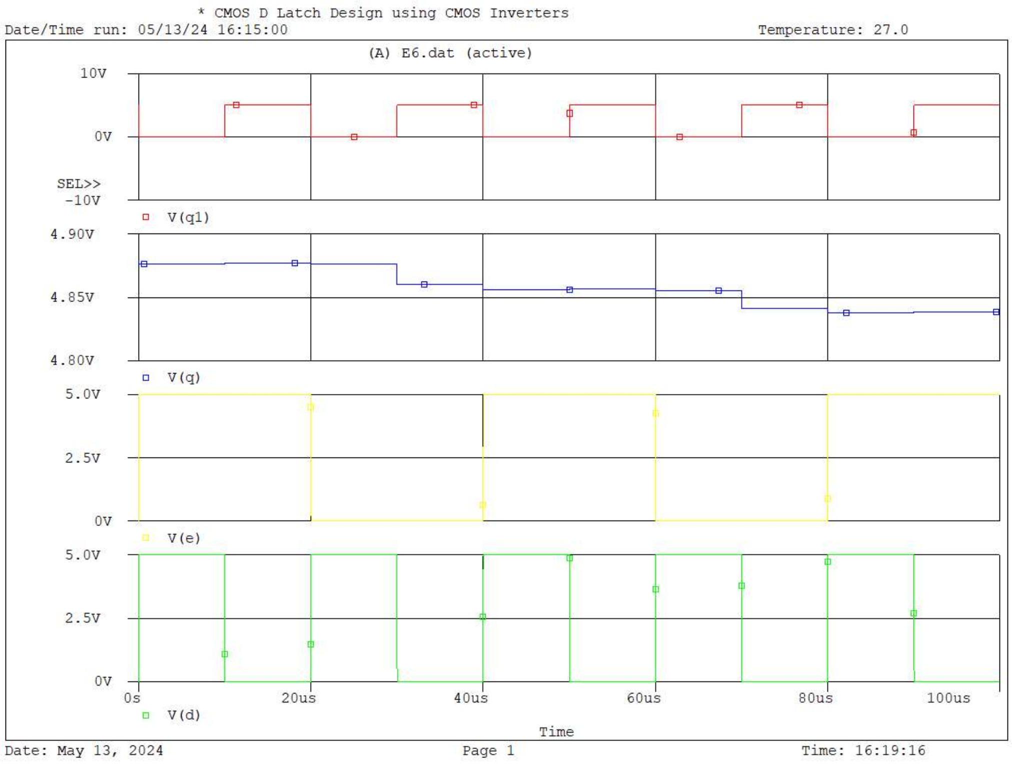
Aim: To design and perform transient analysis of D latch using CMOS inverter.

Software Required: PSpice AD

PSpice Code:

```
*model definitions
<model definition here>

* Define Voltage Sources
Vdd vdd 0 DC 5V
Vd d 0 PULSE(0 5 0 10n 10n 10u 20u)
Ve e 0 PULSE(0 5 0 15n 15n 20u 40u)

* D Latch Circuit Using CMOS
* PMOS transistors
MP1 q1 d vdd vdd P_180n W=2u L=180n
MP2 q2 0 q1 vdd P_180n W=2u L=180n
MP3 q3 q1 q vdd P_180n W=2u L=180n
MP4 q4 q 0 vdd P_180n W=2u L=180n

* NMOS transistors
MN1 q1 d 0 0 N_180n W=1u L=180n
MN2 q2 0 q1 0 N_180n W=1u L=180n
MN3 q3 q1 e 0 N_180n W=1u L=180n
MN4 q4 q e 0 N_180n W=1u L=180n

* Output loading
CL q 0 50p

* Analysis directives
.TRAN 1n 100u
.PROBE
.END
```

Output:



Result: D latch using CMOS inverter was designed and the transient analysis was performed.

<u>**Experiment 7**</u>

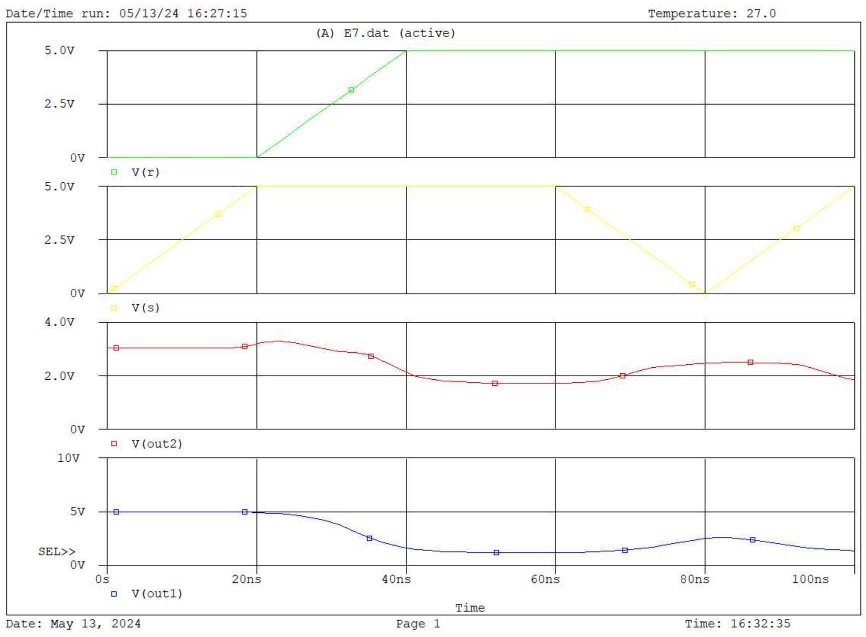
Aim: To design and perform the transient analysis of SR latch circuit using CMOS inverter.

Software Required: PSpice AD

PSpice Code:

```
*model definitions
<model definitions here>

* Define voltage sources
Vdd vdd 0 DC 5V
Vss 0 GND DC 0V
Vs s 0 PULSE(0 5 0 20ns 20ns 40ns 80ns)
Vr r 0 PULSE(0 5 20ns 20ns 40ns 80ns)

* Define the SR Latch using CMOS NOR gates
* First NOR gate
MP1 out1 s vdd vdd P_180n W=3u L=180n
MP2 out1 r vdd vdd P_180n W=3u L=180n
MN1 out1 s out2 GND N_180n W=1.5u L=180n
MN2 out1 r GND GND N_180n W=1.5u L=180n

* Second NOR gate
MP3 out2 r vdd vdd P_180n W=3u L=180n
MP4 out2 out1 vdd vdd P_180n W=3u L=180n
MN3 out2 r out1 GND N_180n W=1.5u L=180n
MN4 out2 out1 GND GND N_180n W=1.5u L=180n

* Output Load Capacitors
Cload1 out1 GND 5pF
Cload2 out2 GND 5pF

* Analysis directives
.TRAN 1ns 100ns
.PROBE
.END
```

Output:



Result: SR latch using CMOS inverter was designed and its transient analysis was performed.

Experiment 8

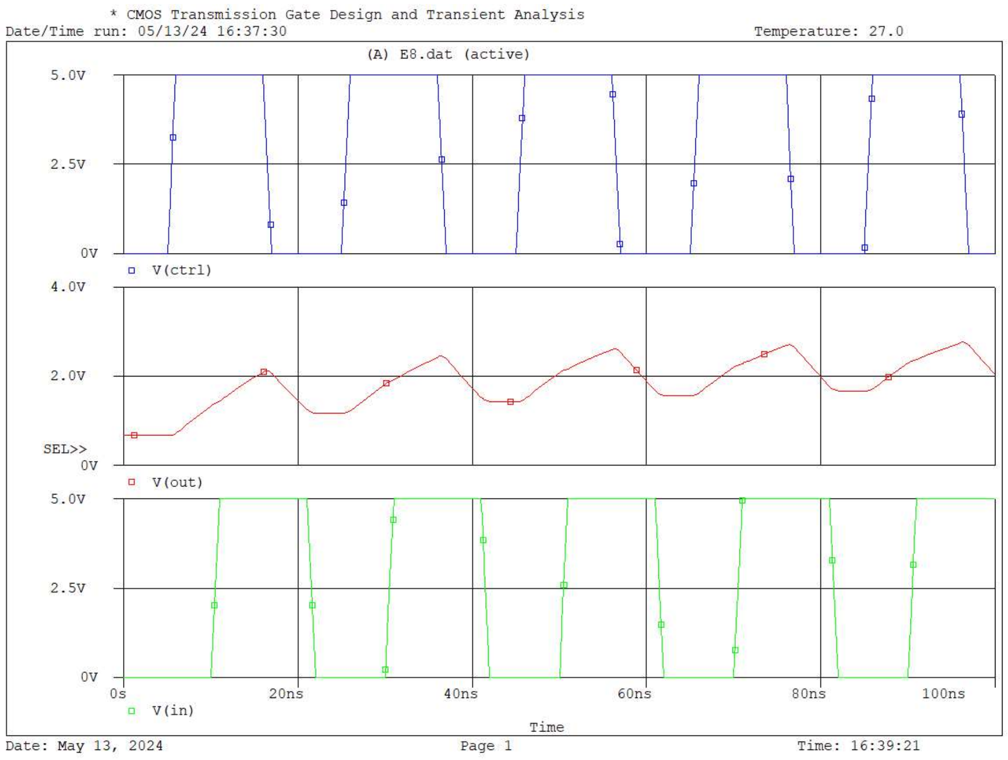
Aim: To design and perform the transient analysis of CMOS transmission gate.

Software Required: PSpice AD

PSpice Code:

```
*model definitions
<model definitions here>

* Define circuit parameters
.PARAM Vdd=5
.PARAM inputPulseHigh=5
.PARAM inputPulseLow=0

* Voltage sources
Vdd vdd 0 DC {Vdd}
Vin in 0 PULSE({inputPulseLow} {inputPulseHigh} 10ns 1ns 1ns 10ns 20ns)
Vctrl ctrl 0 PULSE(0 {Vdd} 5ns 1ns 1ns 10ns 20ns)

* Transmission Gate using NMOS and PMOS
Mn1 out in ctrl 0 N_180n W=1u L=180n
Mp1 out in ctrl vdd P_180n W=1u L=180n

* Output Load
Cload out 0 10pF

* Analysis directives
.TRAN 1ns 100ns
.PROBE

.END
```

Output:



Result: CMOS transmission gate was designed and its transient analysis was performed.

Experiment 9

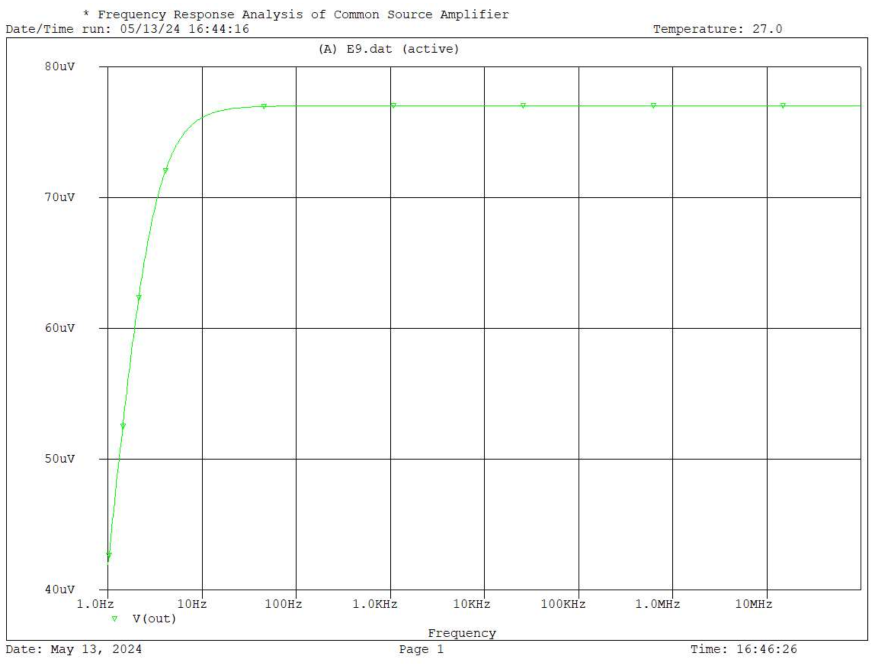
Aim: Analysis of frequency response of Common Source amplifiers.

Software Required: PSpice AD

PSpice Code:

```
*model definitions
<model definitions here>

* Parameters
.PARAM Vdd=5
.PARAM L=180n
.PARAM W=1u

* Voltage Sources
Vdd vdd 0 DC {Vdd}
Vin in 0 AC 1

* NMOS Common Source Amplifier
M1 out in 0 0 N_180n L={L} W={W}

* Biasing the Gate
Vbias gate 0 DC 2.5

* Load and Decoupling Capacitors
CL out 0 10pF
CG gate in 1u

* Analysis directives
.AC DEC 100 1Hz 100MegHz
.PROBE

.END
```

Output:



Result: The frequency response of the common source amplifier was plotted and observed.

<u>**Experiment 10**</u>

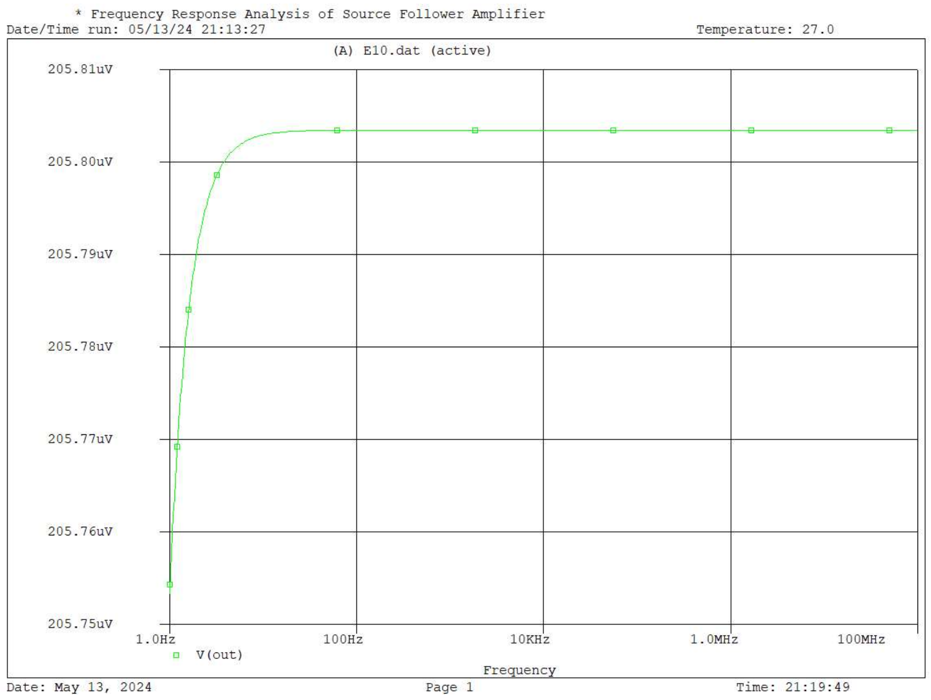
Aim: Analysis of frequency response of Source Follower amplifiers

Software Required: PSpice AD

PSpice Code:

```
*model definitions
<model definitions here>

* Parameters
.PARAM Vdd=5
.PARAM L=180n
.PARAM W=1u

* Voltage Sources
Vdd vdd 0 DC {Vdd}
Vin in 0 AC 1

* NMOS Source Follower Configuration
M1 out in vdd out N_180n L={L} W={W}

* Biasing the Gate
Vbias gate in DC 2.5

* Load and Decoupling Capacitors
CL out 0 10pF
CG gate 0 1u

* Analysis directives
.AC DEC 100 1Hz 100MegHz
.PROBE
.END
```

Output:



Result: Source follower amplifier was designed and its frequency response was observed.

Experiment 11

Aim: Design and Simulation of Full Adder using VHDL program module.

Software Required: Xilinx ISE

VHDL Code:

```
-- Entity declaration for the full adder
entity FullAdder is
   Port (
      A    : in  STD_LOGIC;  -- Input bit A
      B    : in  STD_LOGIC;  -- Input bit B
      Cin  : in  STD_LOGIC;  -- Carry input bit
      Sum  : out STD_LOGIC;  -- Output sum bit
      Cout : out STD_LOGIC   -- Carry output bit
   );
end FullAdder;

-- Architecture declaration of the full adder
architecture Behavioral of FullAdder is
begin
   -- Combinational logic for sum and carry out
   Sum  <= A xor B xor Cin;           -- XOR gate for Sum
   Cout <= (A and B) or (B and Cin) or (A and Cin); -- OR gate for Cout
end Behavioral;
```

Result: The VHDL program module for full adder was simulated and observed.

Experiment 12

Aim: Design and Simulation of 4x1 MUX using VHDL program modul

Software Required: Xilinx ISE

VHDL Code:

```
-- Entity declaration for the 4x1 multiplexer
entity Mux4x1 is
   Port (
      D0   : in  STD_LOGIC;  -- Input data 0
      D1   : in  STD_LOGIC;  -- Input data 1
      D2   : in  STD_LOGIC;  -- Input data 2
      D3   : in  STD_LOGIC;  -- Input data 3
      S    : in  STD_LOGIC_VECTOR(1 downto 0);  -- Selection inputs
      Y    : out STD_LOGIC   -- Output data
   );
end Mux4x1;

-- Architecture declaration of the 4x1 multiplexer
architecture Behavioral of Mux4x1 is
```

```
begin
   -- Process defining the behavior of the multiplexer
   with S select
      Y <= D0 when "00",
          D1 when "01",
          D2 when "10",
          D3 when "11";
end Behavioral;
```

Result: The VHDL program module for 4x1 MUX was simulated and observed.

Experiment 13

Aim: Design and Simulation of BCD to Excess-3 code using VHDL program module

Software Required: Xilinx ISE

VHDL Code:

```
-- Entity declaration for BCD to Excess-3 converter
entity BCD_to_Excess3 is
   Port (
      BCD_in  : in  STD_LOGIC_VECTOR(3 downto 0); -- 4-bit BCD input
      Excess3 : out STD_LOGIC_VECTOR(3 downto 0)  -- 4-bit Excess-3 output
   );
end BCD_to_Excess3;

-- Architecture declaration of the BCD to Excess-3 converter
architecture Behavioral of BCD_to_Excess3 is
begin
   -- Process to handle BCD to Excess-3 conversion
   Excess3 <= BCD_in + "0011";

end Behavioral;
```

Result: The VHDL program module for BCD to Excess-3 code was simulated and observed.

Experiment 14

Aim: Design and Simulation of 3 to 8 decoder using VHDL program module

Software Required: Xilinx ISE

VHDL Code:

```
-- Entity declaration for the 3-to-8 decoder
entity Decoder3to8 is
   Port (
      A   : in  STD_LOGIC_VECTOR(2 downto 0); -- Input vector (3 bits)
      Y   : out STD_LOGIC_VECTOR(7 downto 0)  -- Output vector (8 bits)
```

```
    );
end Decoder3to8;

-- Architecture declaration of the 3-to-8 decoder
architecture Behavioral of Decoder3to8 is
begin
    -- Process to decode the input
    process(A)
    begin
        -- Initialize output to low
        Y <= (others => '0');

        -- Decode input to activate one output
        case A is
            when "000" => Y(0) <= '1';
            when "001" => Y(1) <= '1';
            when "010" => Y(2) <= '1';
            when "011" => Y(3) <= '1';
            when "100" => Y(4) <= '1';
            when "101" => Y(5) <= '1';
            when "110" => Y(6) <= '1';
            when "111" => Y(7) <= '1';
            when others => Y <= (others => '0'); -- Default case
        end case;
    end process;
end Behavioral;
```

Result: The VHDL program module for 3 to 8 decoder was simulated and observed.

<u>Experiment 15</u>

Aim: Design and Simulation of JK Flip-flop using VHDL program module

Software Required: Xilinx ISE

VHDL Code:

```
-- Entity declaration for JK Flip-Flop
entity JKFlipFlop is
    Port (
        J    : in STD_LOGIC;  -- Set input
        K     : in STD_LOGIC;  -- Reset input
        clk   : in STD_LOGIC;  -- Clock input
        clr   : in STD_LOGIC;  -- Asynchronous clear input
        Q     : out STD_LOGIC; -- Output
        Qnot  : out STD_LOGIC  -- Complementary output
    );
```

```
end JKFlipFlop;

-- Architecture declaration of the JK Flip-Flop
architecture Behavioral of JKFlipFlop is
begin
   -- Process with sensitivity list including clk and clr
   process(clk, clr)
   begin
      -- Asynchronous clear
      if clr = '1' then
         Q <= '0';
      elsif rising_edge(clk) then  -- Triggering on the positive edge of the clock
         case (J & K) is
            when "00" =>
               Q <= Q;  -- Hold state
            when "01" =>
               Q <= '0';  -- Reset state
            when "10" =>
               Q <= '1';  -- Set state
            when "11" =>
               Q <= not Q;  -- Toggle state
         end case;
      end if;
      Qnot <= not Q;  -- Complementary output
   end process;
end Behavioral;
```

Result: The VHDL program module for JK Flip-flop was simulated and observed.

<u>Experiment 16</u>

Aim: Design and Simulation of CMOS Inverter using verilog Module

Software Required: Xilinx ISE

Verilog Code:

Verilog module:

```
module CMOSInverter(
   input wire in,    // Input to the inverter
   output wire out   // Output of the inverter
);
```

```
// Behavioral modeling of CMOS inverter
assign out = ~in; // The output is the logical NOT of the input
endmodule
```

Verilog testbench:

```
`timescale 1ns / 1ps
module CMOSInverter_tb;
reg in;     // Input to the inverter
wire out;   // Output of the inverter
// Instantiate the CMOSInverter
CMOSInverter uut (
    .in(in),
    .out(out)
);
initial begin
    // Initialize the input
    in = 0;
    #10; // Wait for 10 ns
    in = 1;
    #10; // Wait for 10 ns
    in = 0;
    #10; // Wait for 10 ns
    $finish; // End the simulation
end
endmodule
```

Result: The Verilog program module for CMOS inverter was simulated and observed.