**Indian Institute of Information Technology Surat**

****

**Lab Report On**

**DIGITAL VLSI DESIGN**

**EC(602)**

**Submitted by**

**Dev Yadav (ui22ec17)**

**Course Faculty**

**Dr.Tanmay Dubey**

**Department of Electronics and Communication Engineering**

**Indian Institute of Information Technology Surat**

**Gujarat-394190, India**

**Table of Contents**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Exp. No** | **Name of the Experiments** | **Pageno.** | **Date of Experiment** | **Date of Submission** | **Marks Obtained** |
| **1** | Analysis of resistor load inverter circuit | 2-5 |  |  |  |
| **2** | Parametric sweep analysis for resistive load inverter circuit | 6-7 |  |  |  |
| **3** | To design and implement CMOS inverter circuit | 8-10 |  |  |  |
| **4** | To calculate P.D. & power dissipation of CMOS inverter circuit | 11-13 |  |  |  |
| **5** | To design and implement CMOS NAND & NOR logic gates | 14-17 |  |  |  |
| **6** |  |  |  |  |  |
| **7** |  |  |  |  |  |
| **8** |  |  |  |  |  |
| **9** |  |  |  |  |  |
| **10** |  |  |  |  |  |
| **11** |  |  |  |  |  |
| 12 |  |  |  |  |  |
| 13 |  |  |  |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

**LAB1**

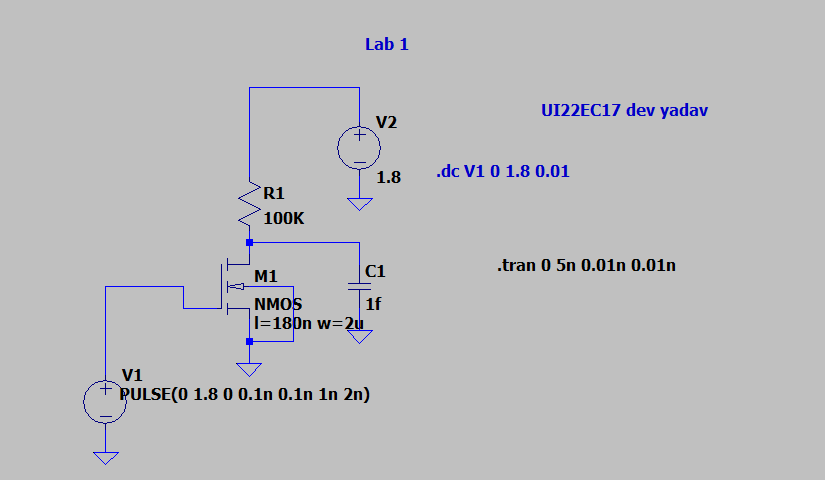
**AIM**: Perform the resistive inverter circuit for dc and transient aim tool theory ckt diagram maths tables waveforms labels will be in format roll no in each circuit.

**APPARATUS**: LTSpice Simulator

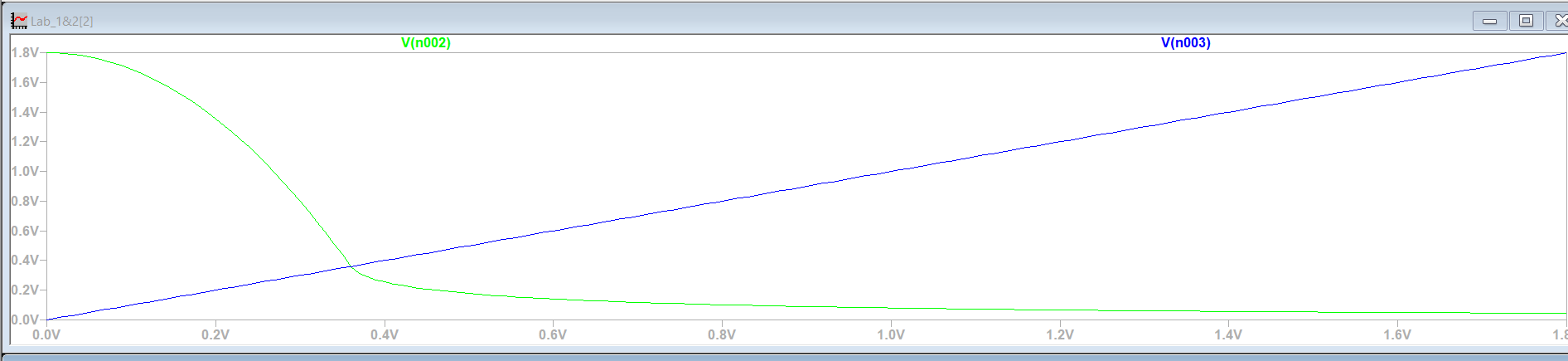
**THEORY**

A **resistive inverter** is one of the simplest forms of a digital inverter circuit, primarily used to demonstrate the fundamental principles of logic inversion in digital electronics. It consists of two main components: a **resistive load** and an **active switching device**, typically a transistor. The output voltage of the circuit depends on the input signal and the characteristics of the load resistance.

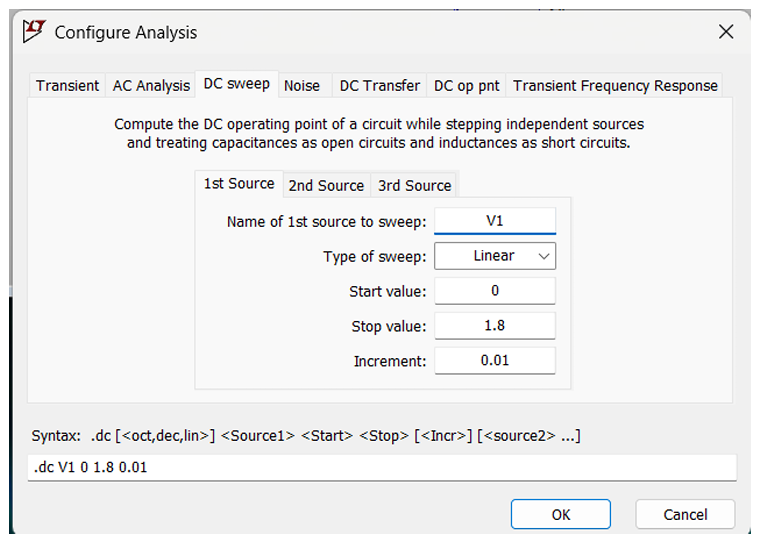
**CIRCUIT**

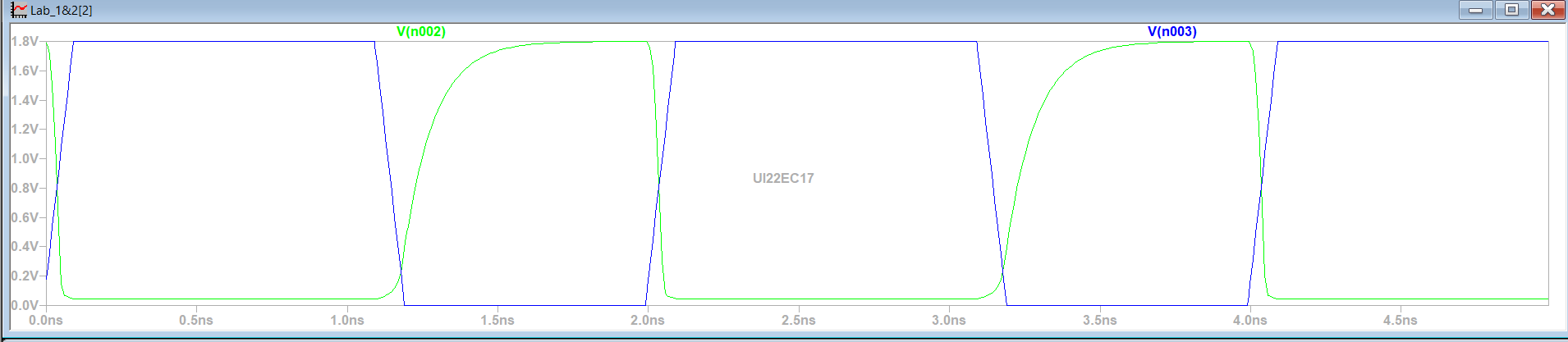
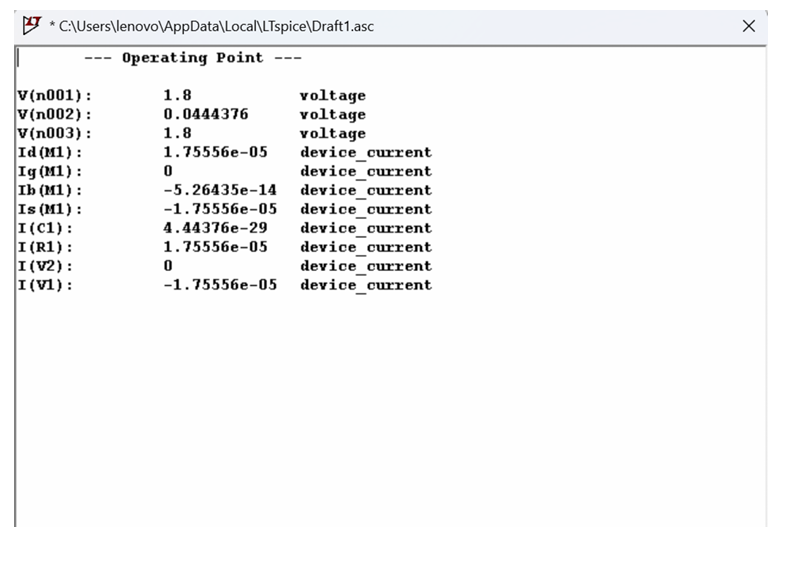


**OUTPUT**

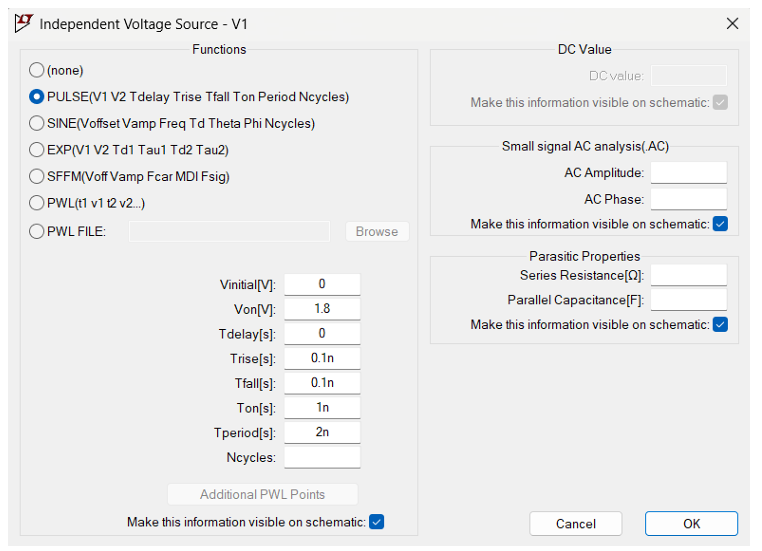


In the figure above, the point where the green line intersects the blue line represents the threshold voltage. The points VIH and VIL can be determined by drawing tangents to the curve; the points where the slope of these tangents equals −1 correspond to VIH and VIL , respectively.





The figure effectively highlights this disparity, showing that the charging time is much longer than the corresponding discharging time. This insight underscores the importance of RC time constant considerations in designing circuits for applications such as filtering, timing, and energy management, where precise control of charge and discharge dynamics is critical.



**Conclusion**: We successfully analyzed and simulated the resistive load inverter circuit, gaining valuable insights into its DC and transient behavior. The experiment allowed us to study the **Voltage Transfer Characteristics (VTC)** curve and compute critical parameters such as VILV\_{IL}VIL​ (input low voltage threshold) and VIHV\_{IH}VIH​ (input high voltage threshold).

The analysis highlighted the inverter's switching behavior, demonstrating how the input voltage determines the output voltage through the resistive load. Additionally, the transient response revealed the charging and discharging dynamics of the circuit, emphasizing the influence of the **RC time constant** on signal propagation and the circuit's speed.

**LAB 2**

**AIM**: To perform the parametric sweep analysis on Resistive Load Inverter to plot the VTC curve for different values of Load Resistance and calculate the various critical parameters of the VTC curve for each cases

**APPARATUS**:- LTSPICE SIMULATOR

**THEORY**

A **Resistive Load Inverter** is a basic digital inverter circuit comprising a resistive load and a transistor. The **Voltage Transfer Characteristics (VTC)** curve of this inverter illustrates the relationship between the input voltage (Vin) and the output voltage (Vout). Performing a **parametric sweep analysis** involves varying the load resistance (Rl) systematically and observing its impact on the VTC curve and critical parameters.

**OH​**: Logic HIGH output voltage.

**VOL​**: Logic LOW output voltage.

**VIH​**: Input voltage threshold for HIGH-to-LOW transition.

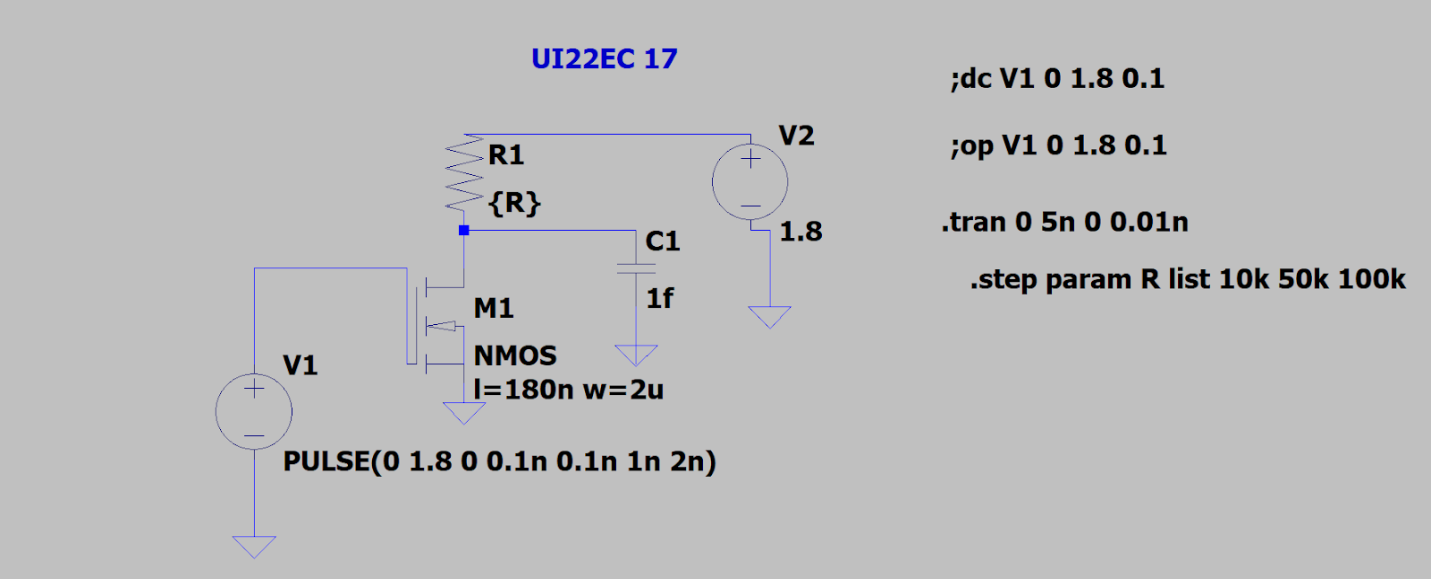
**VIL​**: Input voltage threshold for LOW-to-HIGH transition.

**Noise Margins**:

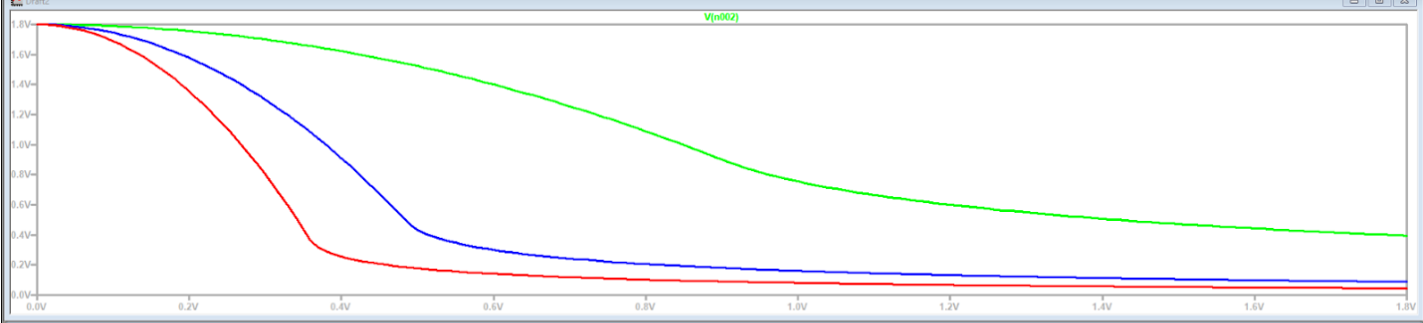
* NMH=VOH−VIH = VOH​−VIH​
* NML=VIL−VOL = VIL​−VOL

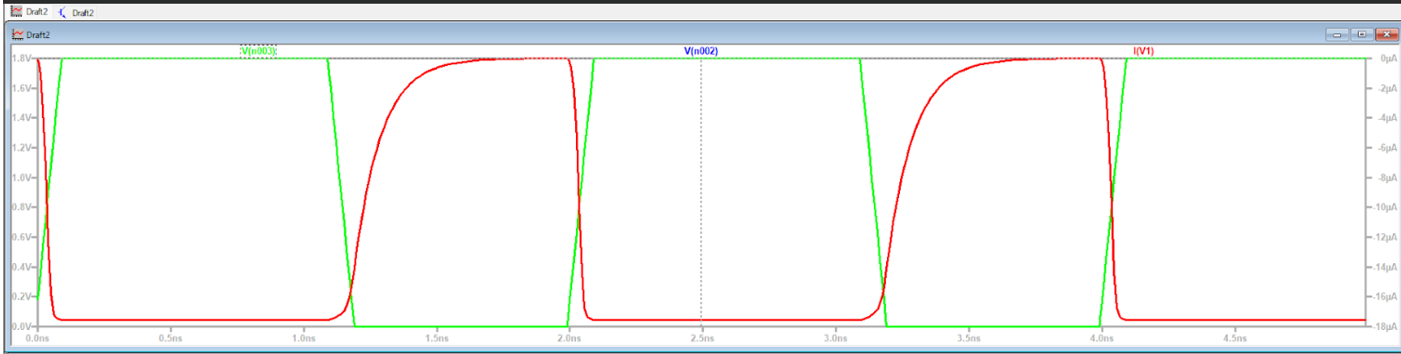
​.

CIRCUIT



OUTPUT





As depicted in the figure, increasing the value of the load resistance has a significant impact on the inverter's behavior. The threshold voltage (VTH ), which is the point where the output voltage transitions sharply, decreases with an increase in resistance. This happens because a higher resistance reduces the current flow, causing the circuit to respond more slowly to input voltage changes, thereby shifting the switching threshold. Moreover, the values of VIH and VIL , which define the high and low logic levels, are also influenced as the slope of the VTC curve changes. The experiment demonstrates the sensitivity of the inverter's performance to load resistance, emphasizing the need to optimize these parameters to achieve desired switching behavior in practical circuit designs.

**Conclusion**

In Lab 2, we investigated how variations in load resistance impact the performance of a resistive load inverter circuit. By conducting a parametric sweep analysis, we examined the changes in the voltage transfer characteristics (VTC) and evaluated critical parameters such as VIL​, VIH​, and VTH​.

The analysis revealed that increasing the load resistance reduces the current flow through the circuit, leading to a decrease in the threshold voltage (VTH​). This shift in VTH​ significantly influenced the switching behavior of the inverter, altering its response to input voltage transitions. Additionally, variations in VIL​ and VIH​ demonstrated the dependence of the inverter's logic levels on resistance, highlighting its sensitivity and tunability.

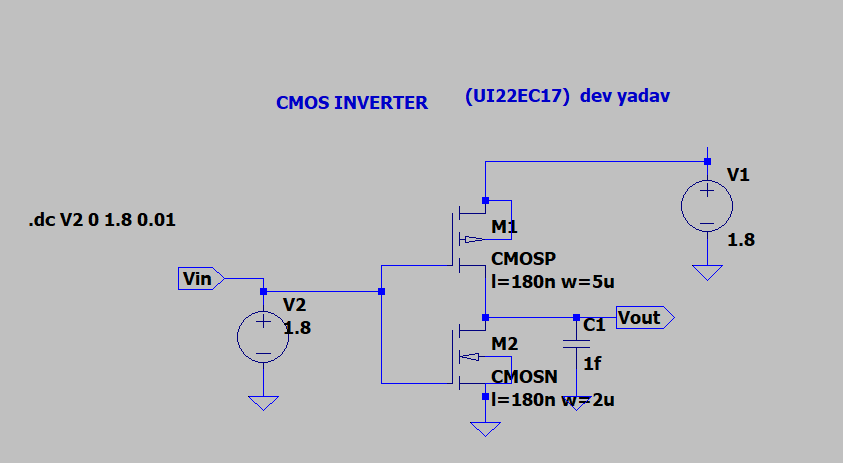
These findings emphasize the importance of load resistance in determining the inverter's operational characteristics, allowing for optimization of switching behavior and logic thresholds for specific design requirements.

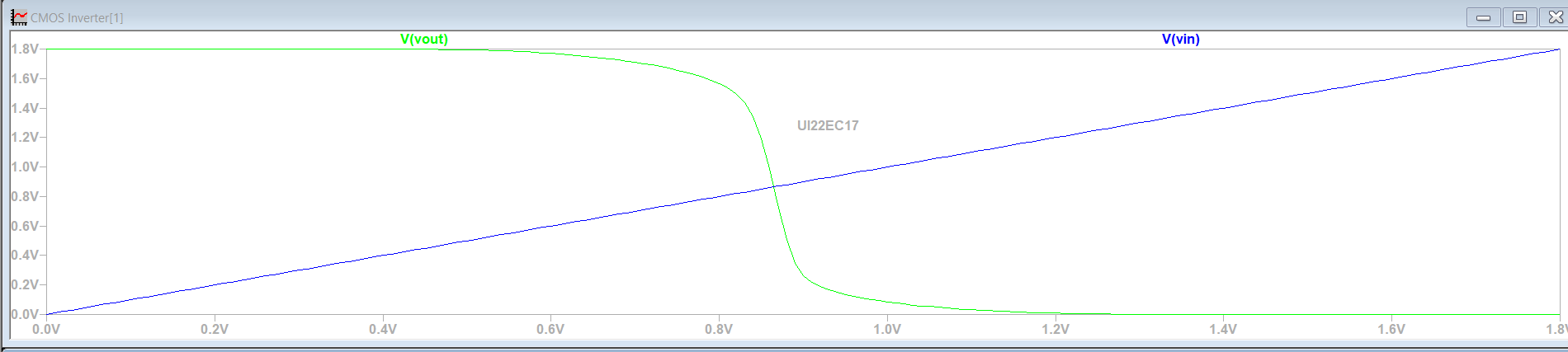
**LAB3**

**AIM**:To implement Cmos inverter circuit and perform the DC analysis and compare the result with Cmos inverter with 180nm technology

**APPARATUS**: LT Spice simulator

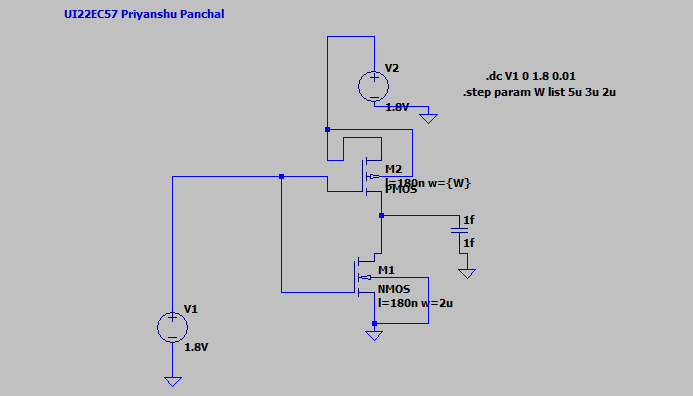
**CIRCUIT DIAGRAM**

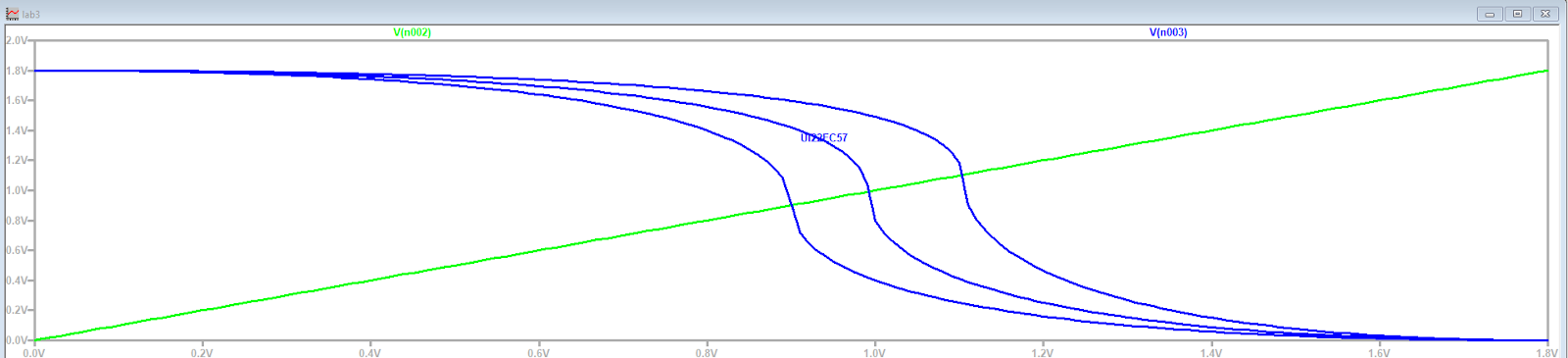




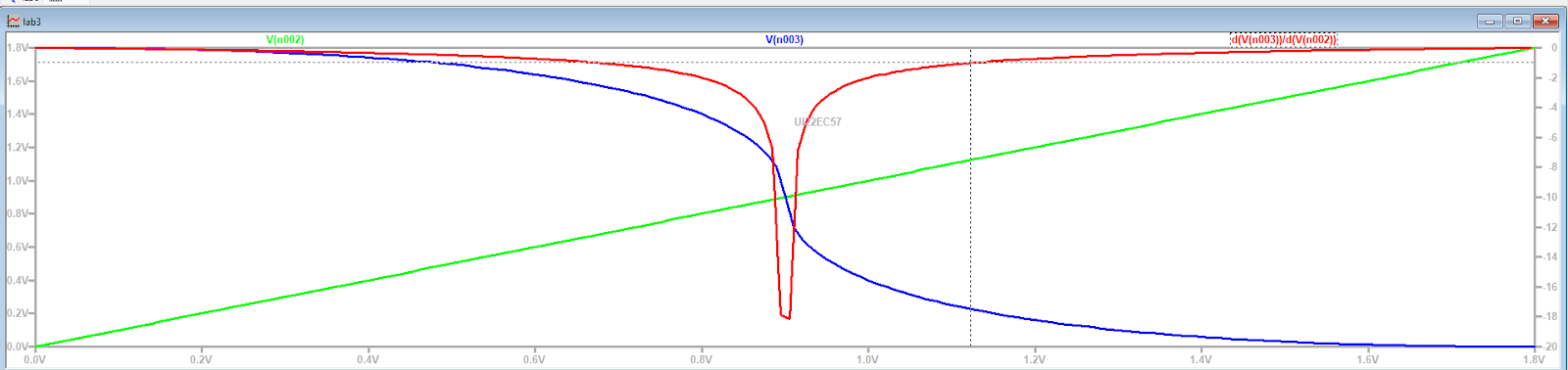
OBSERVATION WITH VARIABLE WIDTH

For w=2u

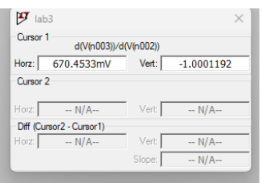




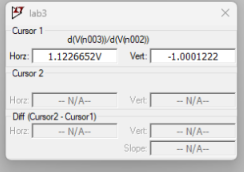
READINGS OF VIL AND VIH



VIH



VIL



From the above results Vil=672.41mv and Vih=1.12v and Voh=Vdd=1.8 v and Vol=0v

**CALCULATION**

**Noise Margin**

Nmh=Voh-Vih

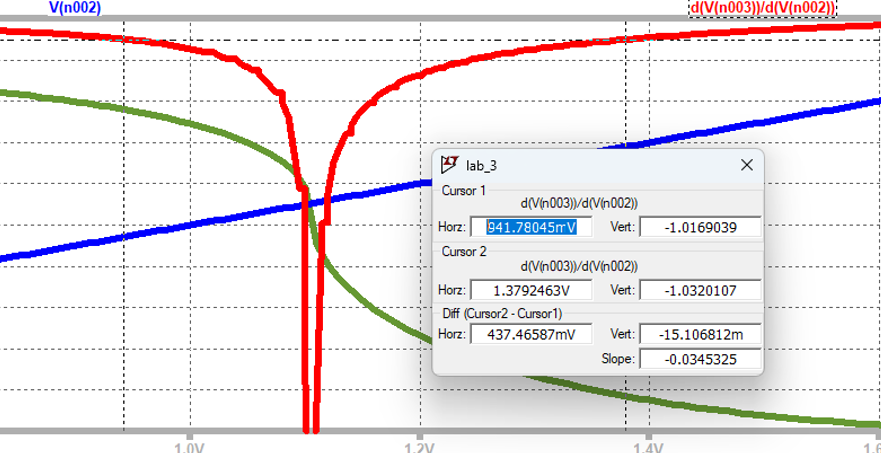
        =1.8-1.12

        =0.68v

Nml=Vil-Vol

        =0.67v

Now for w=5u



From the above results Vil=941.78mv and Vih=1.37v and Voh=Vdd=1.8 v and Vol=0v

**CALCULATION**

Noise margin

Nmh=Voh-Vih

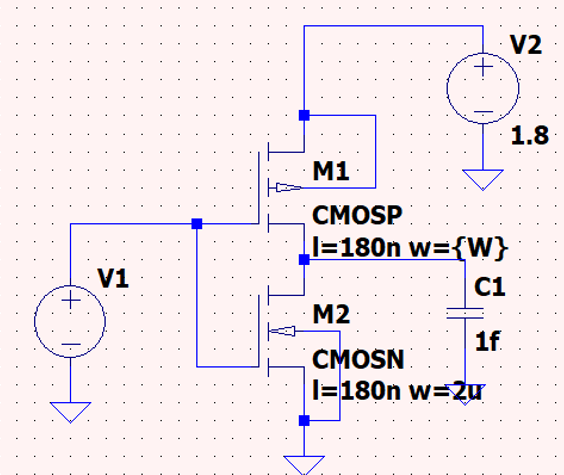
        =1.8-1.37v

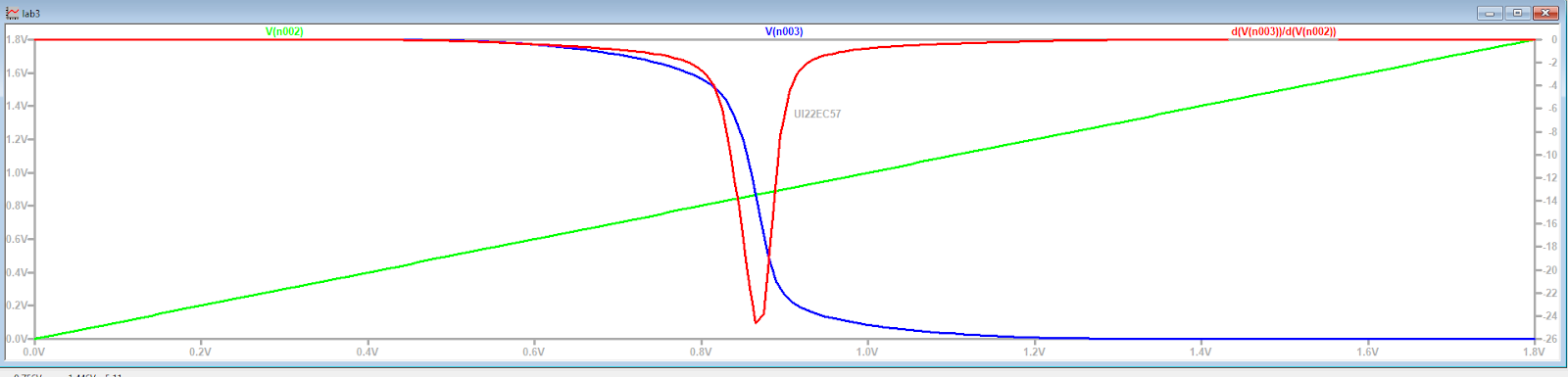
        =0.43v

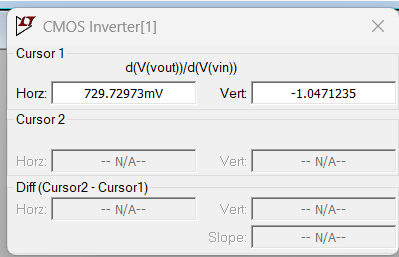
Nml=Vil-Vol

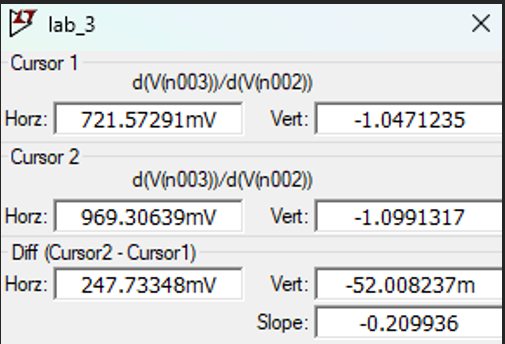
        =0.94v

**NOW FOR 180nm TECHNOLOGY**









From the above results Vil=721.57mv and Vih=962.30mv and Voh=Vdd=1.8 v and Vol=0v

**CALCULATION**

**Noise margin**

Nmh=Voh-Vih

        =1.8-0.962v

        =0.838vv

Nml=Vil-Vol

        =0.962v

**COMPARISON TABLE**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Parameter** | **VIH(V)** | **VIL(V)** | **VOL(V)** | **VOH(V)** | **Noise Margin**  **(NMH)** | **Noise Margin**  **(NMH)** |
| **w=2u** | **1.12** | **0.67** | **0** | **1.8** | **0.68** | **0.67** |
| **w=5u** | **1.37** | **0.94** | **0** | **1.8** | **0.43** | **0.94** |
| **w=5u**  **180nm** | **0.97** | **0.72** | **0** | **1.8** | **0.84** | **0.72** |

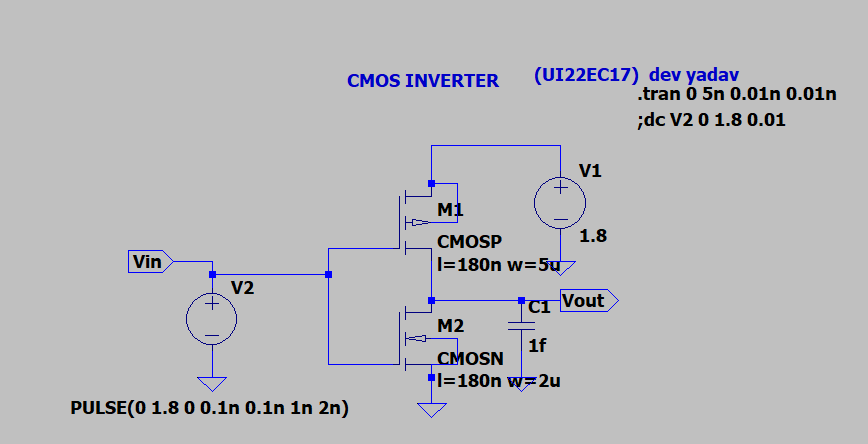
**CONCLUSION**

As the width increases, the input high voltage (VIH) and input low voltage (VIL) increase, while the noise margin for the high state (NMH) decreases for smaller widths, with a slight improvement seen for a larger width (180mm). The output voltage (both high and low) remains stable across all widths. These results suggest that, while larger widths may slightly improve noise margins for the low state, they might cause a reduction in noise margin for the high state.

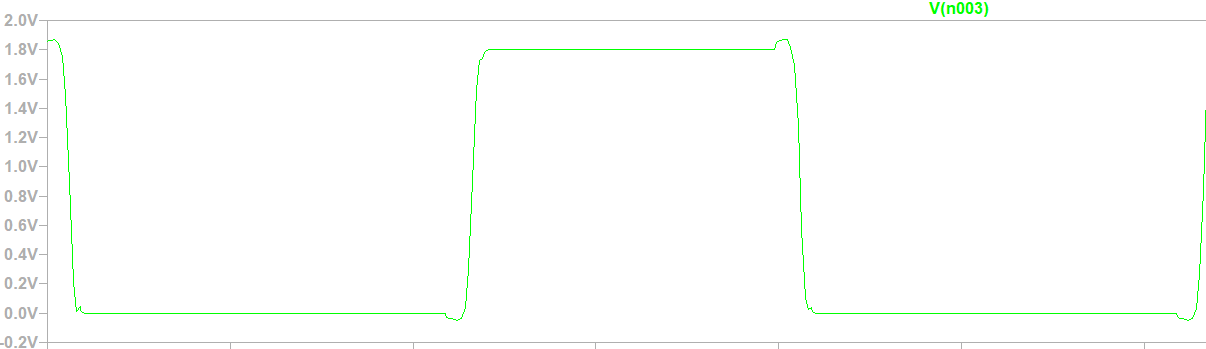
**LAB 4**

**AIM:**To implement CMOS inverter circuit in 180nm technology and perform transient analysis by calculating the delay and power dissipation.

**Circuit diagram**



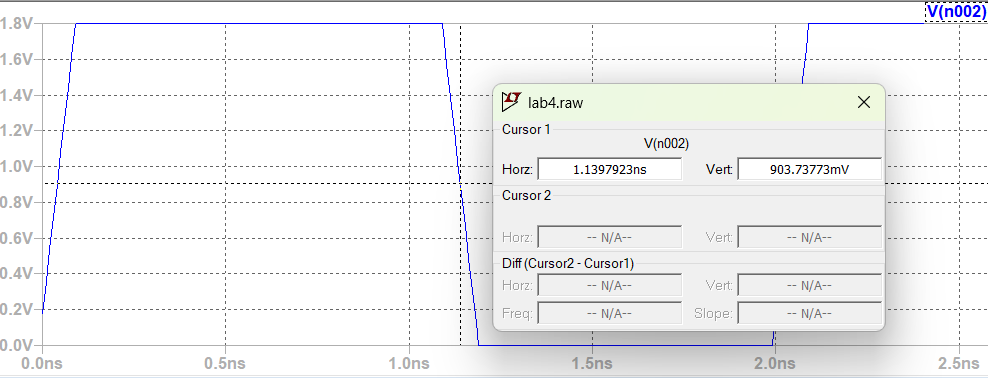
**OUTPUT TRANSIENT ANALYSIS**



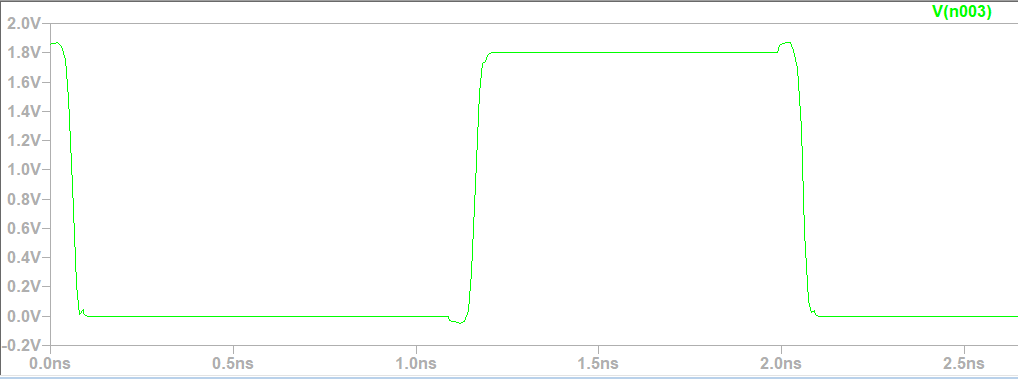
**PROPAGATION DELAY**

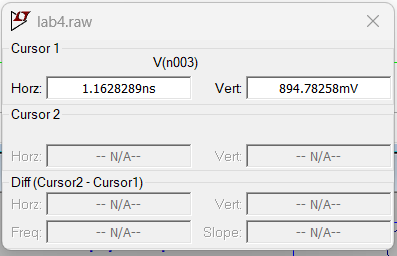
**FROM LOW TO HIGH**

**VIN**



**VOUT**





50% OF INPUT VIN THE TIME TAKEN IS 1.13ns

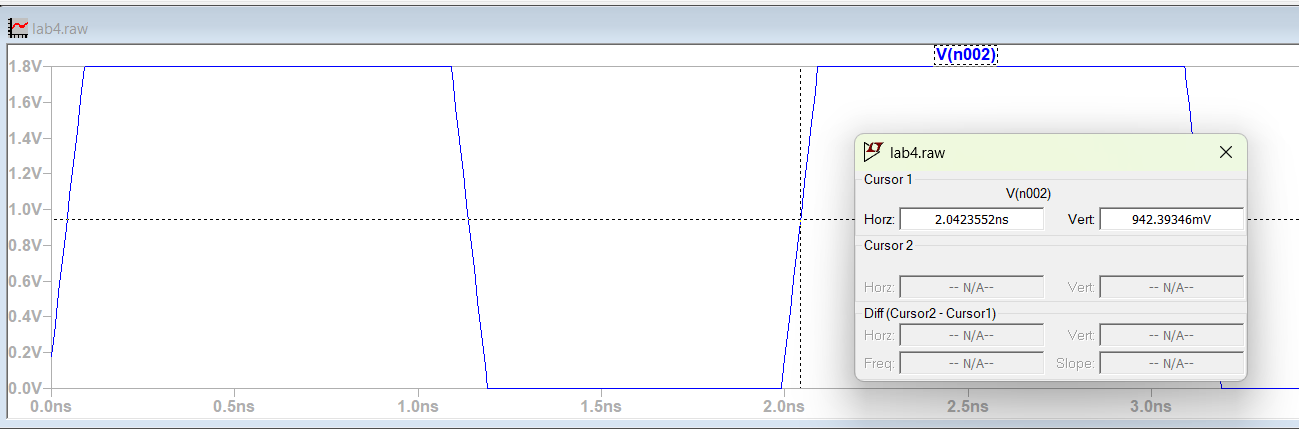
50% of OUTPUT VOUT THE TIME TAKEN IS 1.16ns

 So delay from low to high is the difference of the time taken for input to reach 50% and output to reach 50% that is 24ps

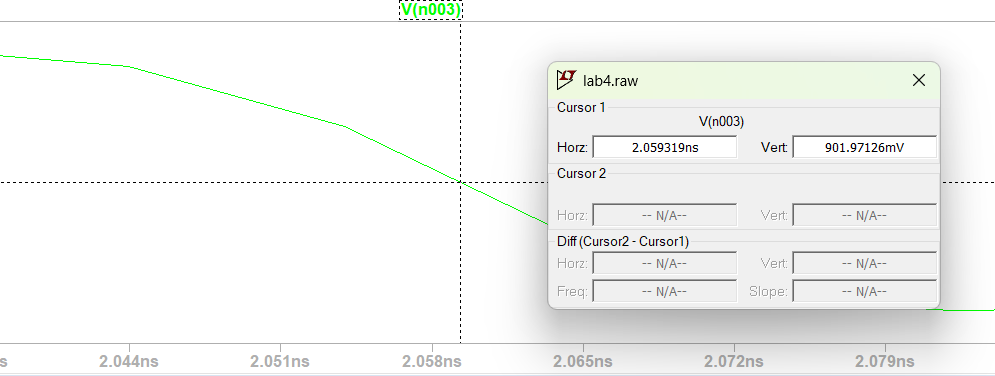
**PROPAGATION DELAY**

**FROM HIGH TO LOW**

**VIN**



**VOUT**



50% OF INPUT VIN THE TIME TAKEN IS 2.04ns

50% of OUTPUT VOUT THE TIME TAKEN IS 2.05ns

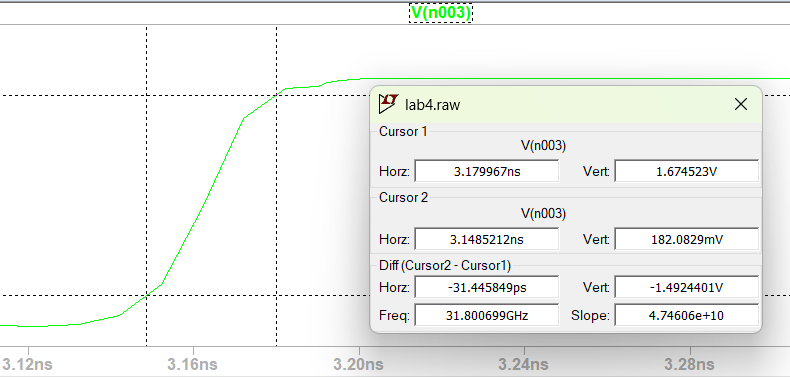
 So delay from low to high is the difference of the time taken for input to reach 50% and output to reach 50% that is 18ps

So the total delay is the average of both the delays that is (18+24)/2=21ps

**RISE TIME**

10% OF VOH IS 0.18V

90%OF VOH IS 1.62V



RISE TIME=10%OF OUTPUT-90%OF OUTPUT

                  =3.17-3.12

                  =31ps

**FALL TIME**

10% OF VOH IS 0.18V

90%OF VOH IS 1.62V

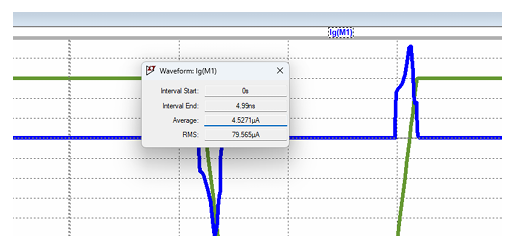
FALL TIME=90%OF OUTPUT -10%OF OUTPUT

                   =2.07-2.04

                   =26ps



**POWER**



Average Current: 4.5271𝜇A

 Average Power: Average Current x Average Voltage

 =8.14μW

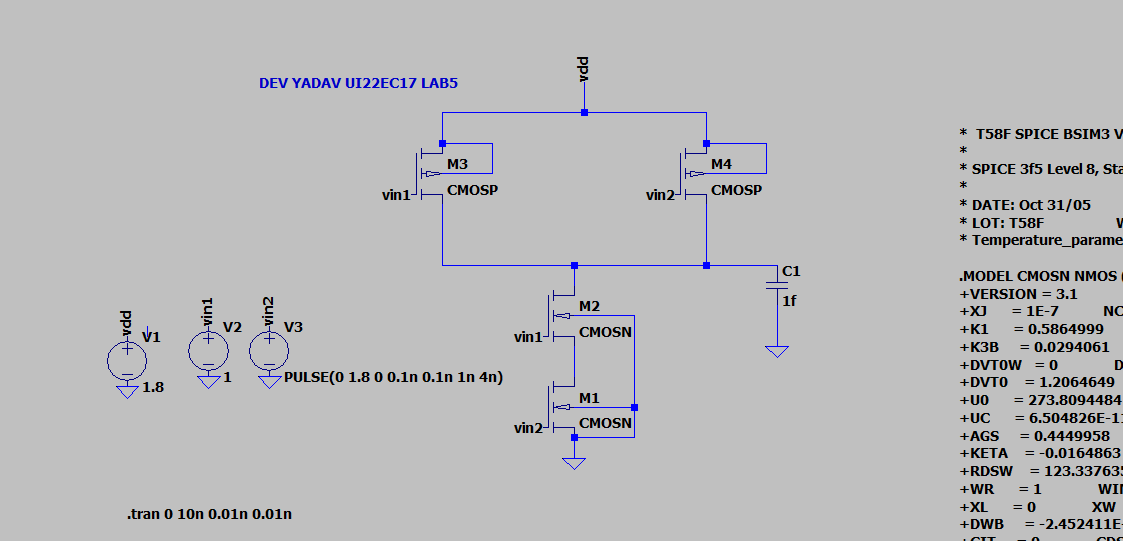
**CONCLUSION**

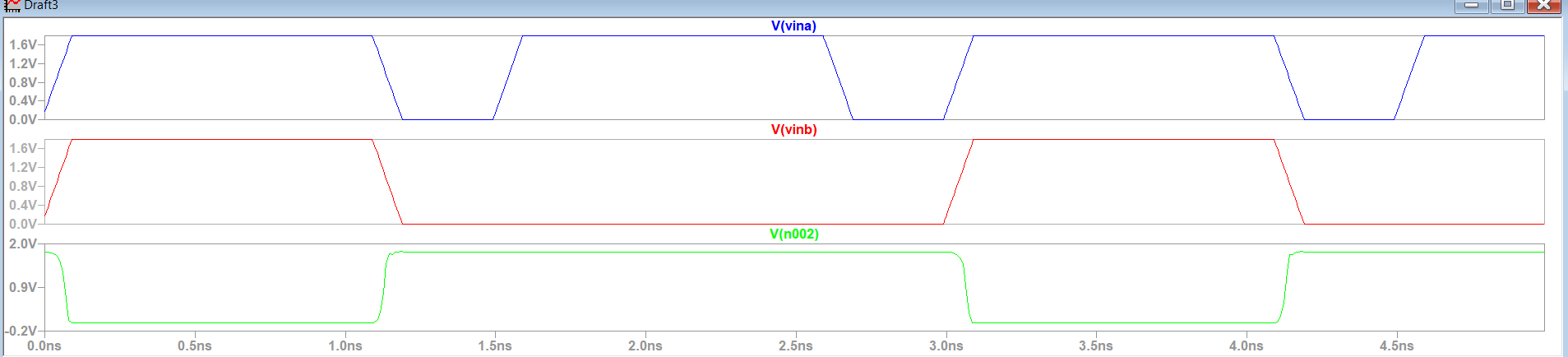
The CMOS inverter implemented in 180nm technology demonstrates excellent performance with low propagation delay (21ps average), fast rise (31ps) and fall (26ps) times, and minimal power dissipation (8.14μW). These characteristics make the inverter ideal for use in high-speed, low-power digital circuits, which is a fundamental benefit of CMOS technology

**Assignment -5**

**Aim:-** To design and implement Two input NAND and NOR gate using CMOS logic and measure its propagation delay with respect to each input considering the worst case scenario.

**Circuit Diagram for NAND logic:-**

****

****

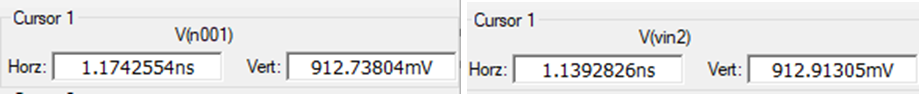
**For NAND Logic:-**

**Case-1**

**A=1 and B=pulse**

**Tplh**

Input fall Output rise

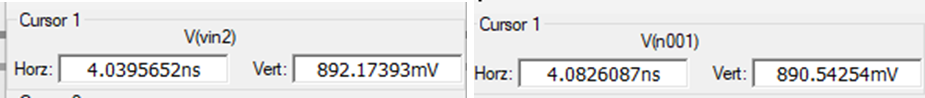
****

**Tplh=Tfall-Trise**

**1.13-1.17=0.04ns**

**Tphl**

**Input rise Output fall**

****

**Tphl =Tfall-Trise**

**4.08-4.03=0.05ns**

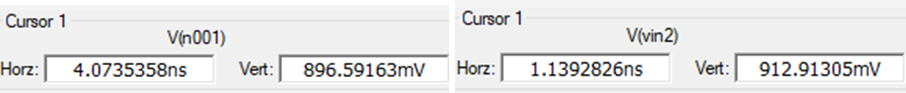
**Tavg=(Tplh+Tphl)/2 =0.04+0.05)/2 =0.045ns**

**Case-2**

**A=pulse and B=1**

**Tplh**

Input fall Output rise

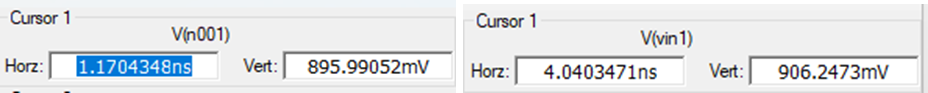
****

**Tplh=Tfall-Trise**

**4.07-1.13=2.94ns**

**Tphl**

**Input rise Output fall**

****

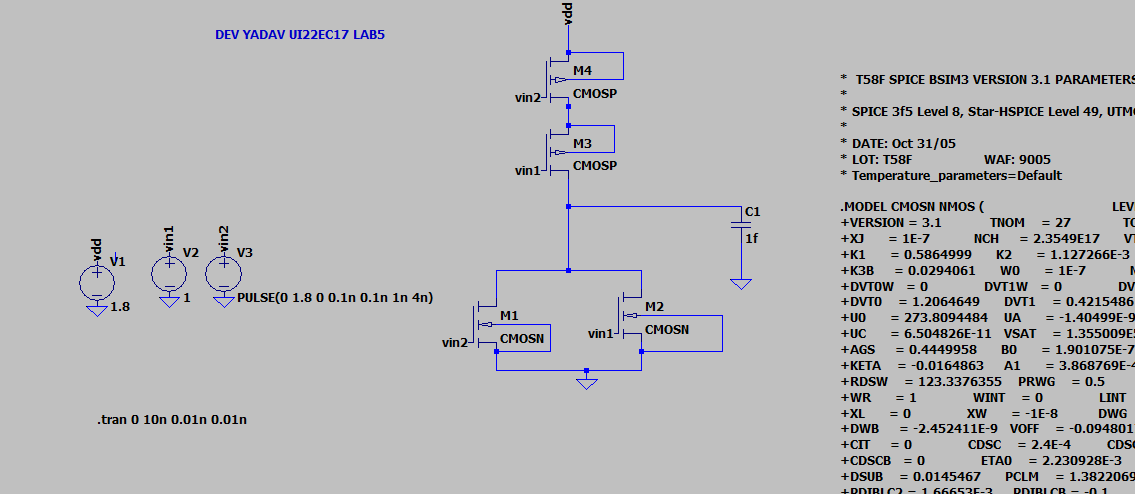
**Tphl =Tfall-Trise**

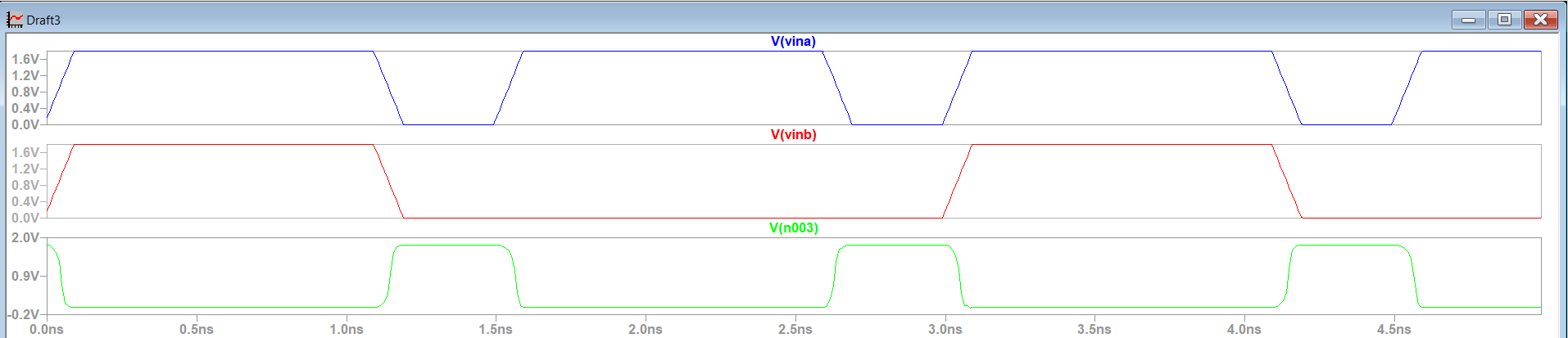
**4.04-1.17=2.87ns**

**Tavg=(Tplh+Tphl)/2 =(2.87+2.94)/2 =2.90ns**

**Tavg for nand gate using cmos=(0.045+2.90)/2=1.47ns**

**Circuit Diagram for NOR logic:-**

****

****

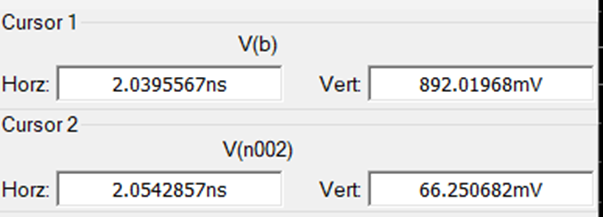
**For NOR Logic:-**

**Case-1**

**A=1 and B=pulse**

**Tplh**

Input fall Output rise

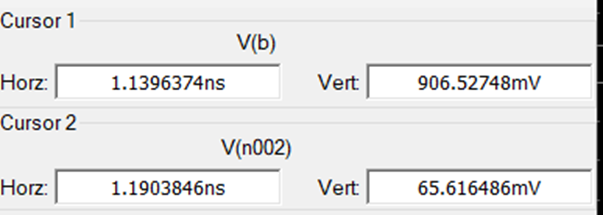
****

**Tplh=Tfall-Trise**

**2.05-2.03=0.02ns**

**Tphl**

**Input rise Output fall**

****

**Tphl =Tfall-Trise**

**1.19-1.13=0.06ns**

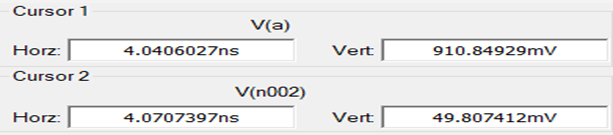
**Tavg=(Tplh+Tphl)/2 =(0.02+0.06)/2 =0.04ns**

**Case-2**

**A=pulse and B=1**

**Tplh**

Input fall Output rise

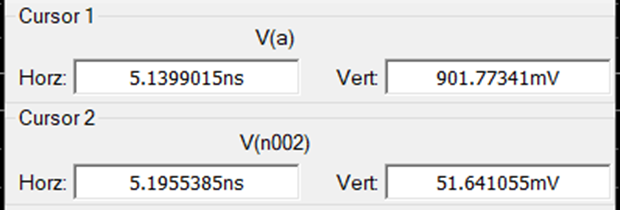
****

**Tplh=Tfall-Trise**

**4.07-4.04=0.03ns**

**Tphl**

**Input rise Output fall**

****

**Tphl =Tfall-Trise**

**5.19-5.13=0.05ns**

**Tavg=(Tplh+Tphl)/2 =(0.03+0.05)/2 =0.04ns**

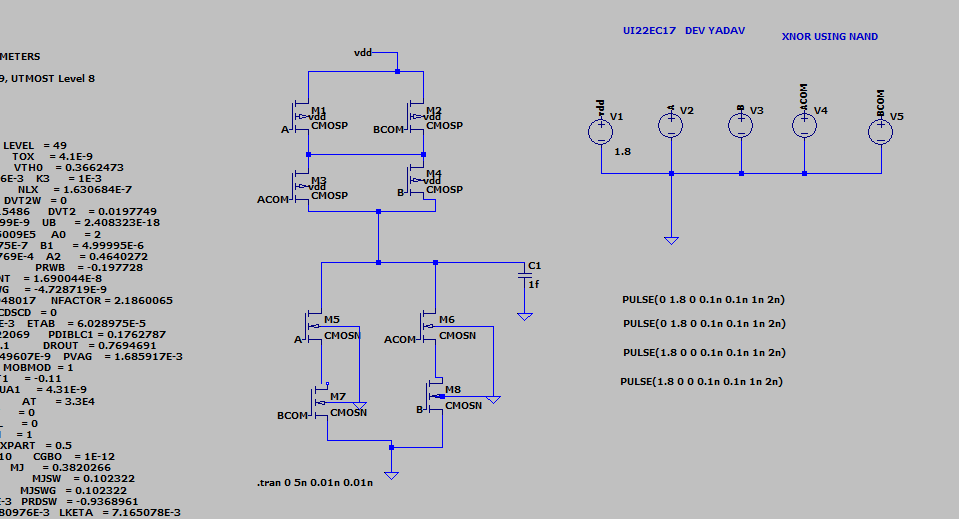
**Tavg for nor gate using cmos=(0.04+0.04)/2=0.04ns**

**Conclusion:-** Learn how to implement NAND and NOR logic using the cmos circuit and calculate their delays for both NAND and NOR logic.

**Assignment -6**

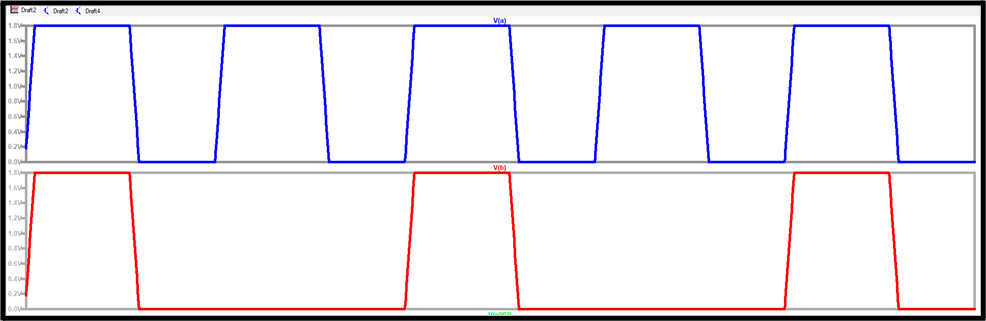
**Aim:-**To design and implement Two input XNOR gate using CMOS logic and observe its propagation delay with respect to each input considering the worst case scenario**.**

**Circuit Diagram for XNOR logic:**

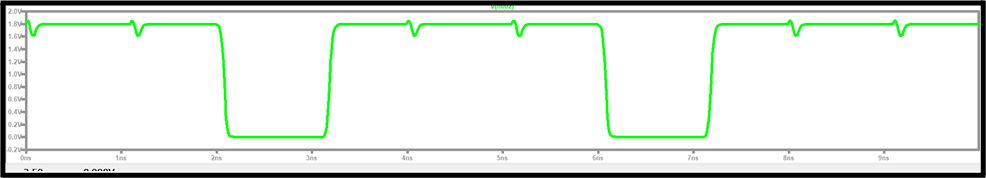
****

**Output:-**

**Input (A and B)**

****

**Output(A⊕B)**

****

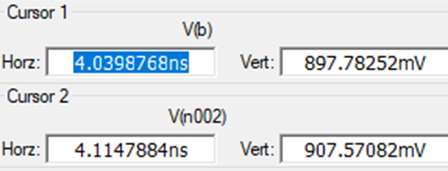
**For XNOR Logic:-**

**Case-1**

**A=1 and B=pulse**

**Tplh**

Input fall Output rise

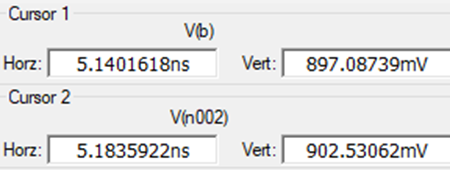
****

**tplh=Tfall-Trise**

**4.114-4.039=0.075ns**

**Tphl**

**Input rise Output fall**

****

**Tphl =tfall-trise**

**5.18-5.14=0.04ns**

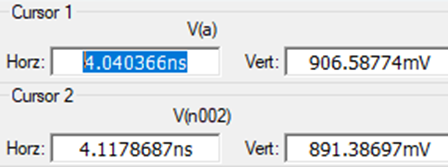
**Tavg=(Tplh+Tphl)/2 =(0.04+0.075)/2 =0.0575ns**

**Case-2**

**A=pulse and B=1**

**Tplh**

Input fall Output rise

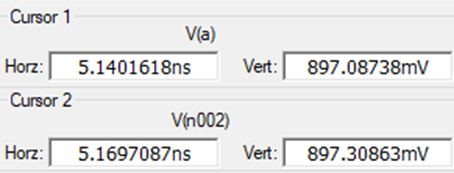
****

**Tplh=Tfall-Trise**

**4.117-4.04=0.077ns**

**Tphl**

**Input rise Output fall**

****

**Tphl =Tfall-Trise**

**5.16-5.14=0.02ns**

**Tavg=(Tplh+Tphl)/2 =(0.077+0.02)/2 =0.0485ns**

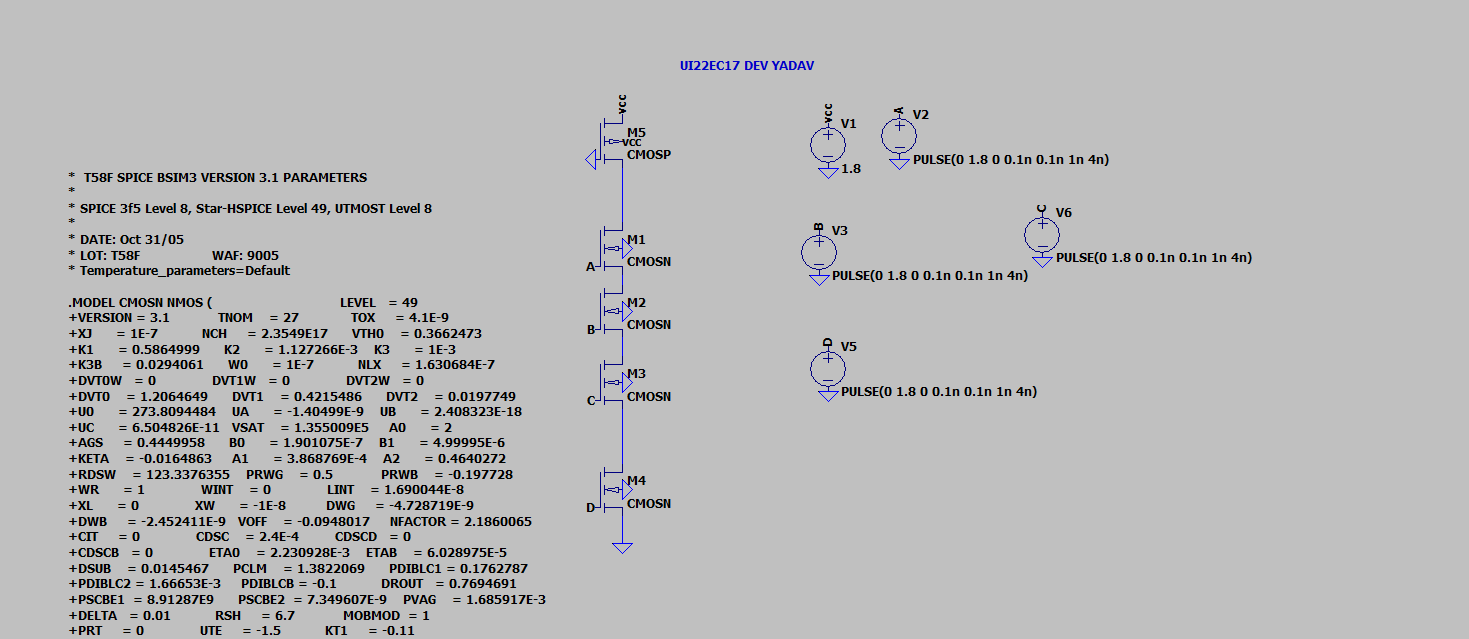
**Tavg for nor gate using cmos=(0.0575+0.0485)/2=0.053ns**

**Conclusion:-** Learn how to implement XNOR logic using the cmos circuit and calculate their delays for both XNOR logic.

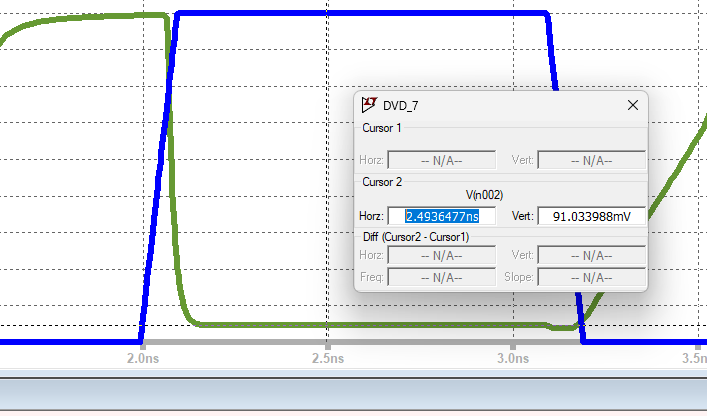
**Assignment 7**

**Aim:** To design and implement four input nand and nor gates using pseudo n-mos logic and observe its delay and power dissipation and voltage swing at the output.

**NAND Gate:**

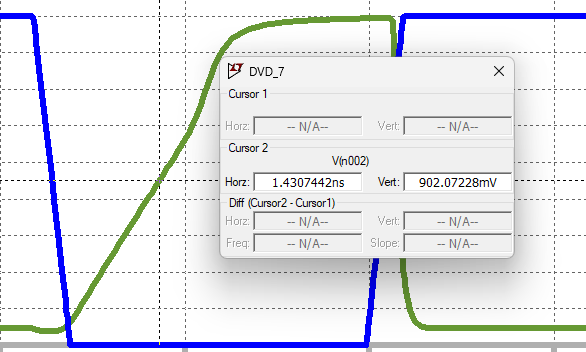


**VOL : 95mV**

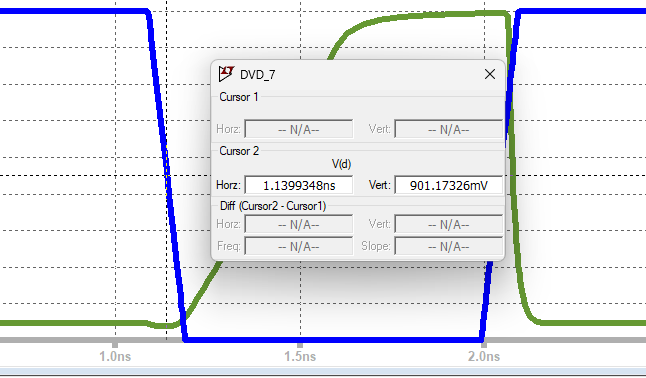


**T(PHL) ->**

Input:



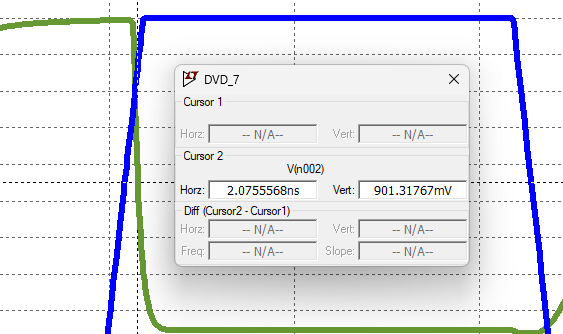
Output:



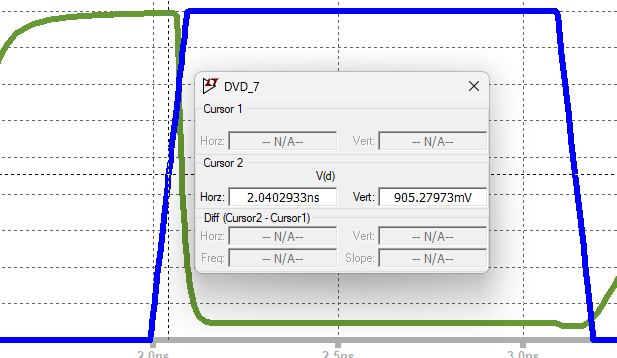
1.4307ns - 1.1399ns   
 -> 290 pico sec

**T(PHL):**

Output:



Input:



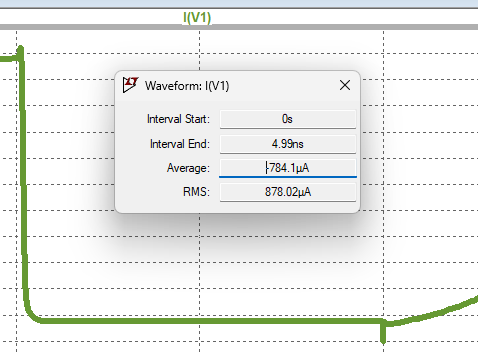
Delay: 2.0402-2.075

-> 35 pico

**Average Delay :**

(35+290)/2 -> 162.5 pico second

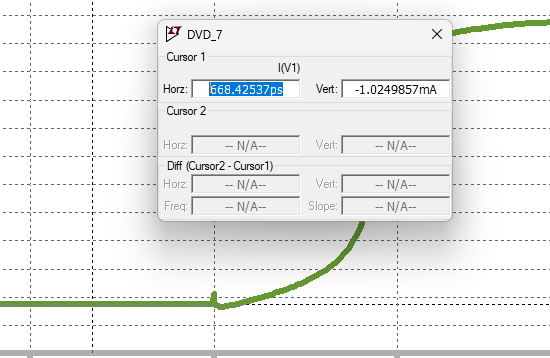
**Average Power:**

****

**Average Power=** -784.1\*1.8

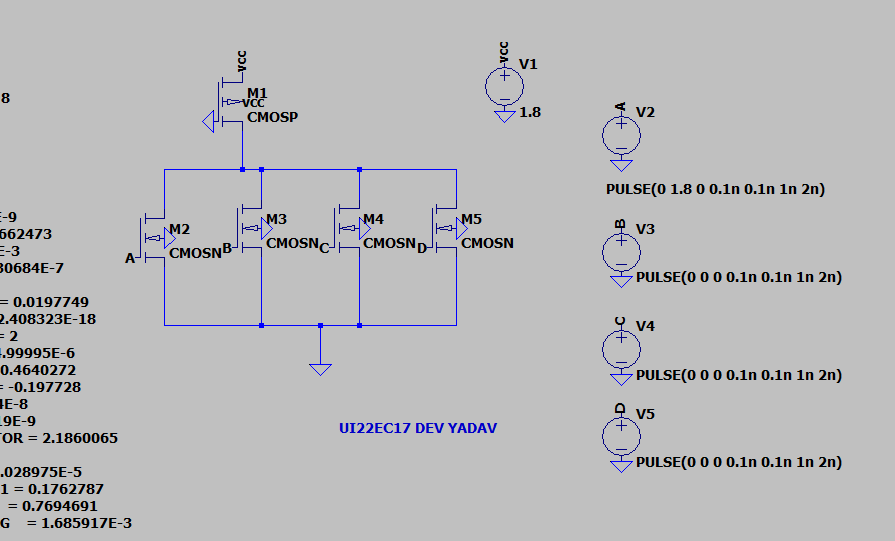
**-1.41mW**

**Static Power: -1.025\*1.8 -> -1.845mW**

****

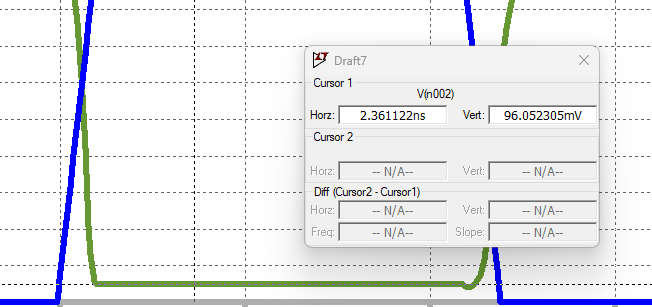
**NOR Gate:**

**Circuit Diagram:**

****

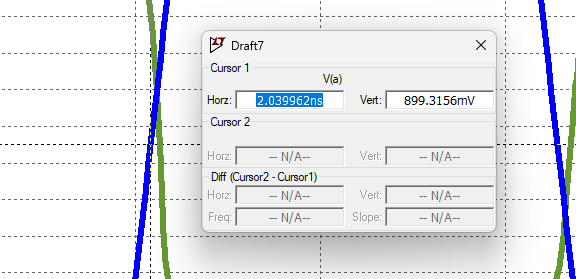
**For A=1.8V and B=C=D=0V**

**VOL=**96.05mV

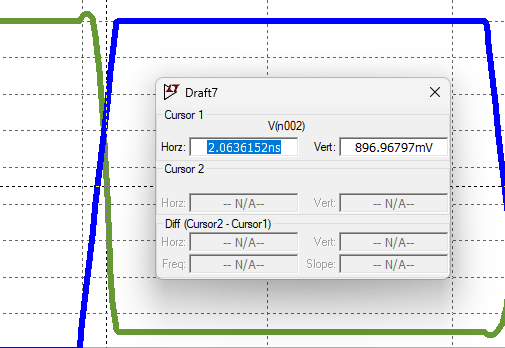
****

**T(PHL) ->**

Input:



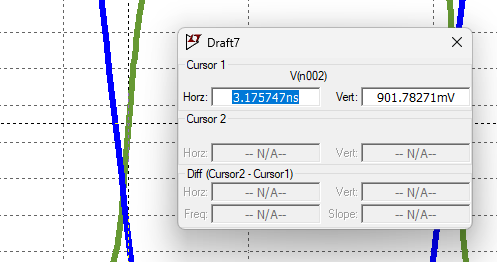
Output:



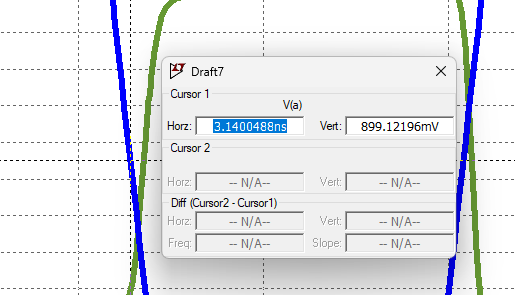
2.0636-2.0399  
 -> 24 pico sec

**T(PLH):**

Output:



Input:



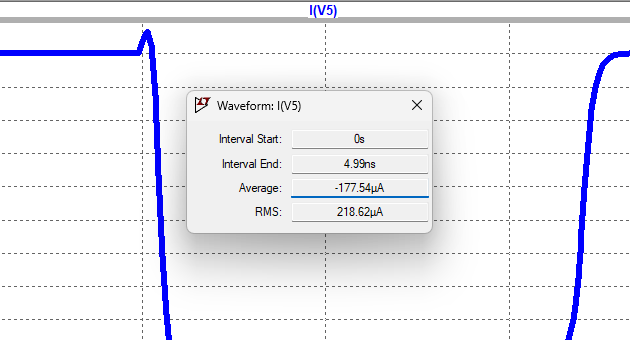
Delay: 3.1757-3.1404

-> 35 pico

**Average Delay :**

(35+24)/2 -> 29.5 pico second

**Average Power:**

****

**Average Power=** -177\*1.8

**-318 microW**

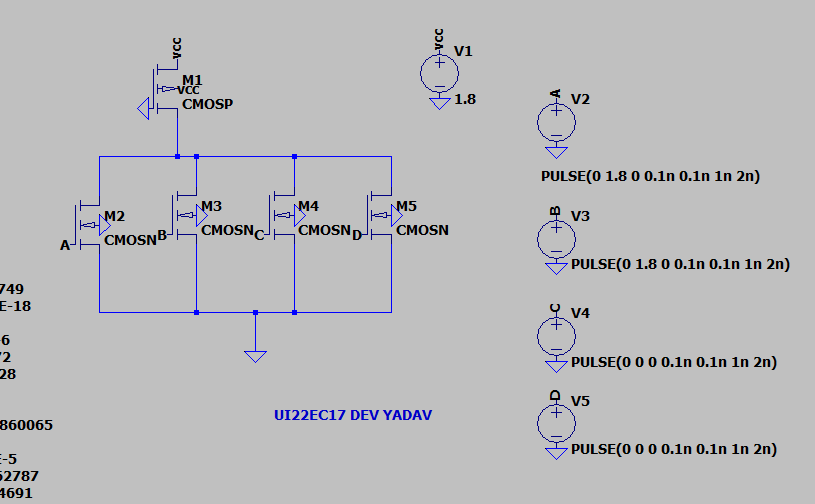
**Static Power:**

****

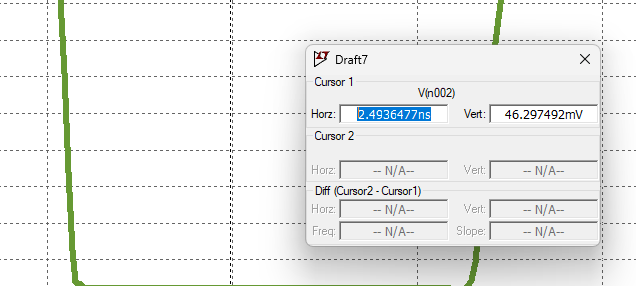
**Static Power=** -273\*1.8

**-491.8 microW**

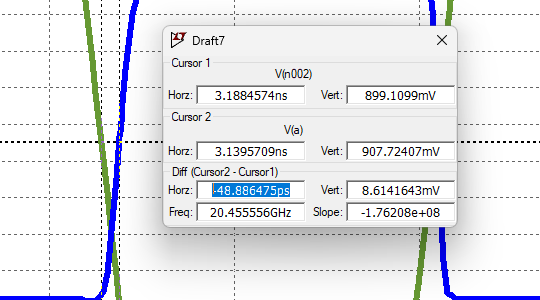
**For A=B=1.8V and C=D=0V**

****

**VOL =** 46mV

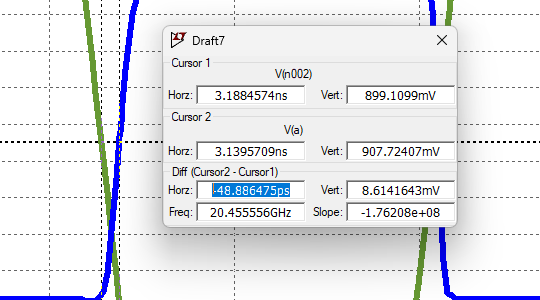
****

**T(PHL) ->**



10.7 PicoSecond

**T(PLH):**

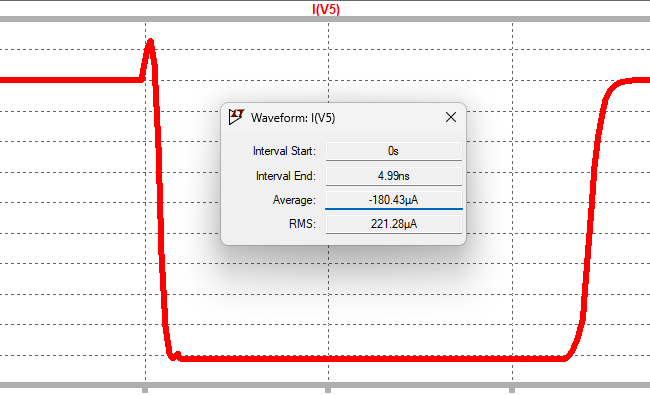


48.8 Picosecond

**Average Delay :**

(10.7+48.8)/2 -> 30 picosecond

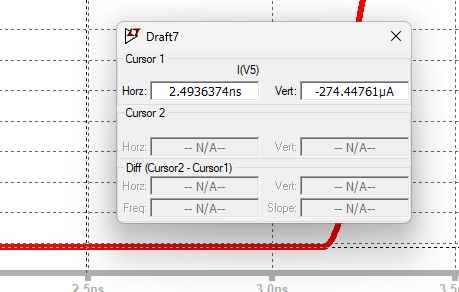
**Average Power:**

****

**Average Power=** -180\*1.8

**-324 microW**

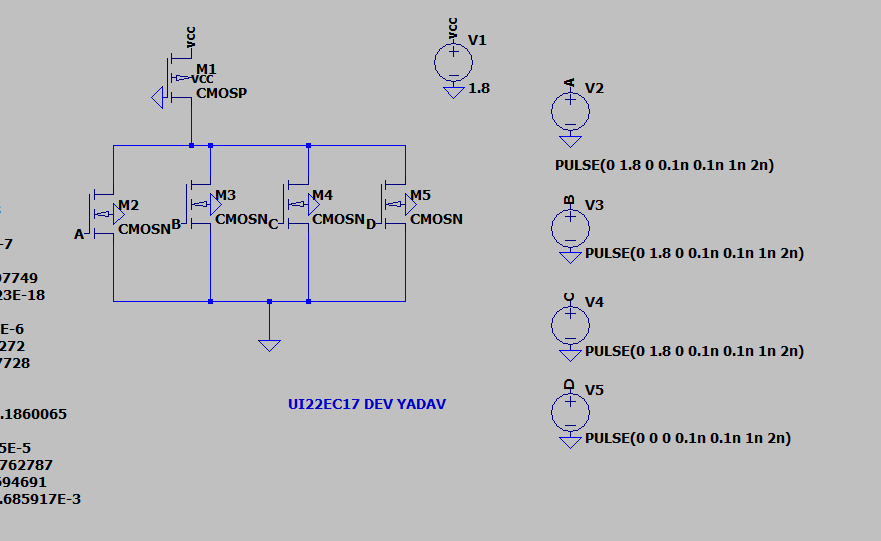
**Static Power:**

****

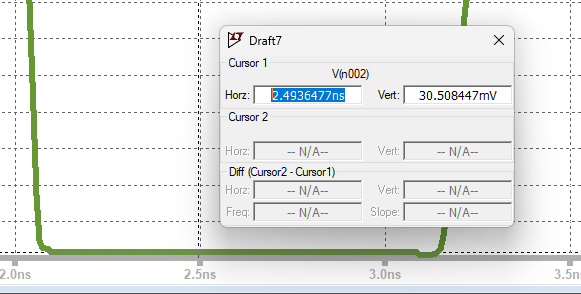
**Static Power=** -274\*1.8

**-493.99 microW**

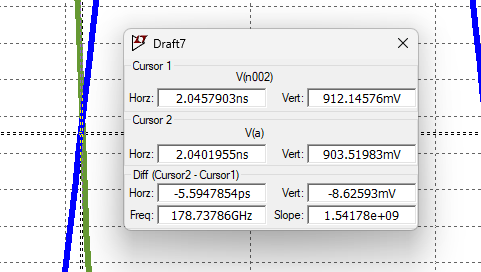
**For A=B=C=1.8V and D=0V**

****

**VOL =** 30mV

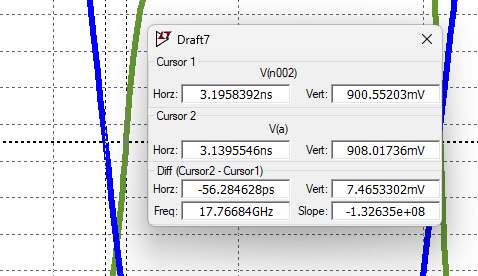
****

**T(PHL) ->**



5.5 PicoSecond

**T(PLH):**

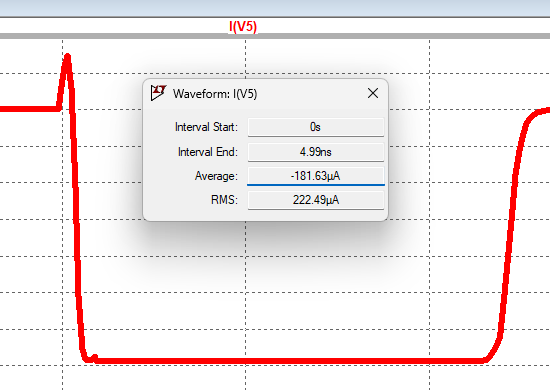


56.2 Picosecond

**Average Delay :**

(5.5+56.2)/2 -> 30.5 picosecond

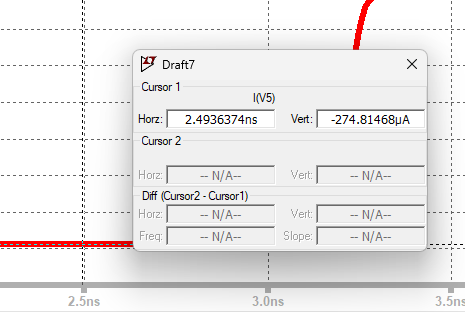
**Average Power:**

****

**Average Power=** -181\*1.8

**-325.5 microW**

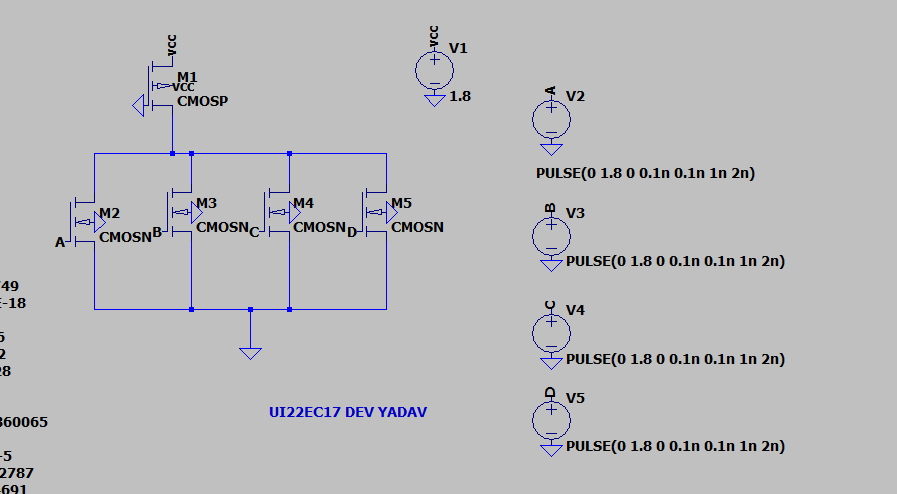
**Static Power:**

****

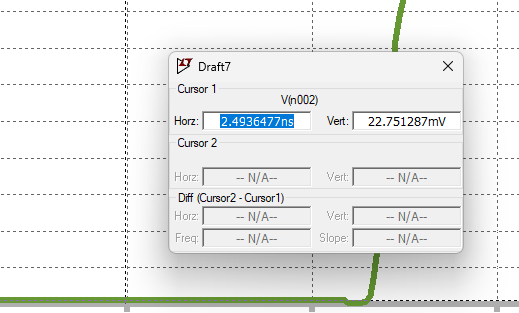
**Static Power=** -274\*1.8

**-494.65 microW**

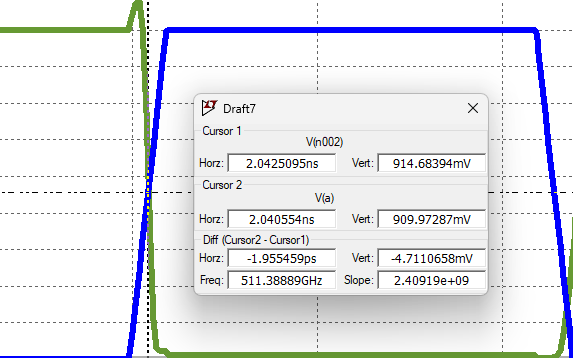
**For A=B=C=D=1.8V**

****

**VOL =** 22mV

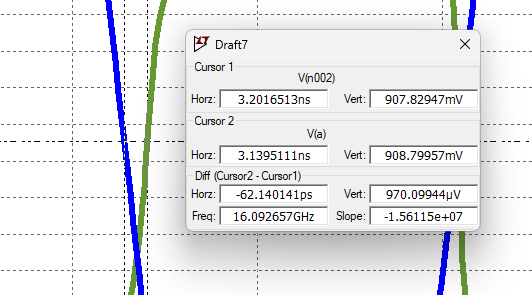
****

**T(PHL) ->**



1.9 PicoSecond

**T(PLH):**

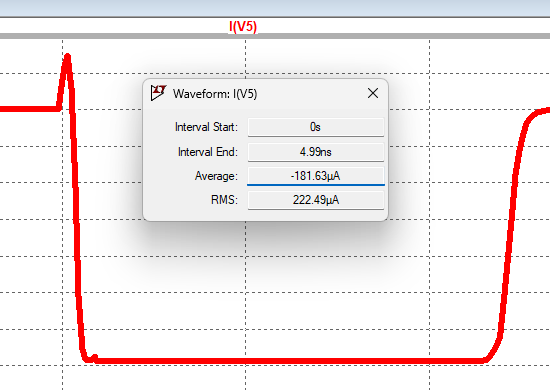


62.1 Picosecond

**Average Delay :**

(1.9+62.1)/2 -> 32 picosecond

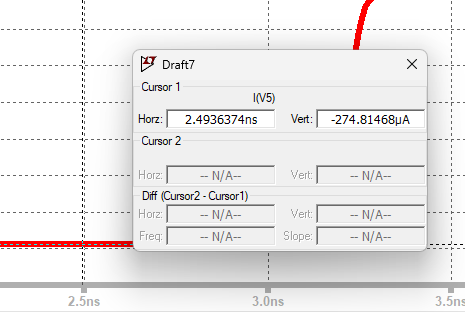
**Average Power:**

****

**Average Power=** -182\*1.8

**-327 microW**

**Static Power:**

****

**Static Power=** -274.96\*1.8

**-494.9 microW**

**Tabular Representation:**

|  |  |  |  |
| --- | --- | --- | --- |
| **ABCD** | **VOL(mV)** | **Delay**  **(picoSecond)** | **Static Power**  **(- microW)** |
| **1000** | 96 | 29.5 | 491.8 |
| **1100** | 46 | 30 | 493.9 |
| **1110** | 30 | 30.5 | 494.6 |
| **1111** | 22 | 32 | 494.9 |

**Conclusion:**

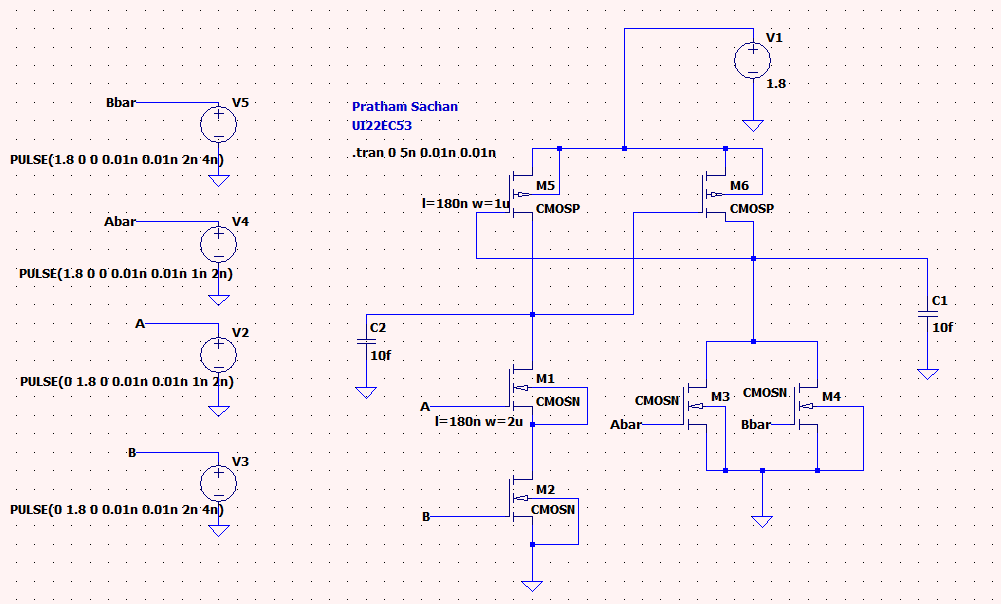
In this experiment we have successfully designed and implemented a 4 input NAND and NOR gate using pseudo NMOS logic. We observed that

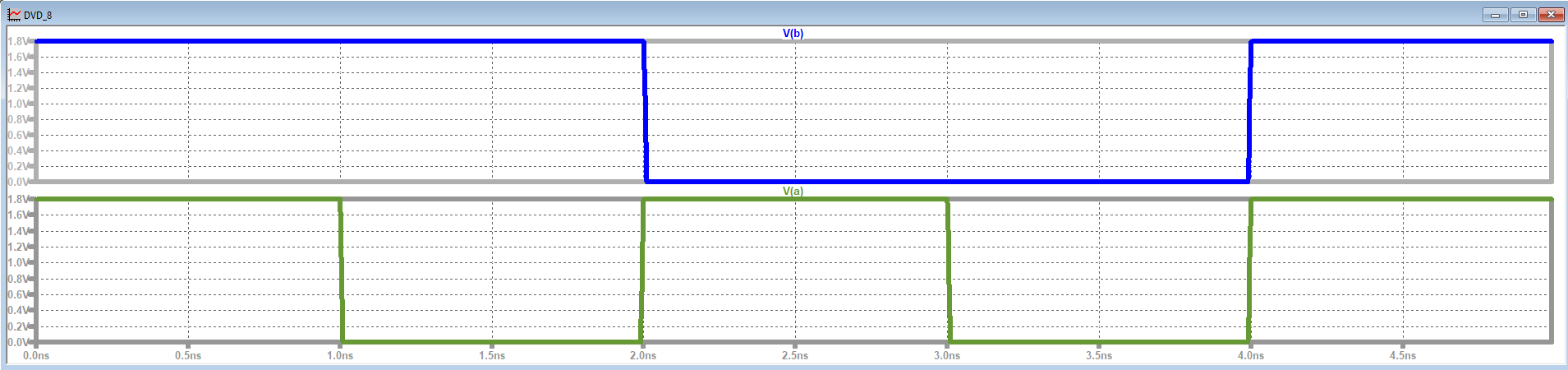
* **Voltage:** Increasing voltage leads to a decrease in delay.
* **Delay:** Higher voltage results in faster system performance with decreasing delay.
* **Static Power:** Static power consumption remains nearly constant across varying voltage levels.

**Assignment: 8**

**Aim:** To design and implement two input AND/NAND gates using differential cascode voltage swing logic and observe its output voltage swing.

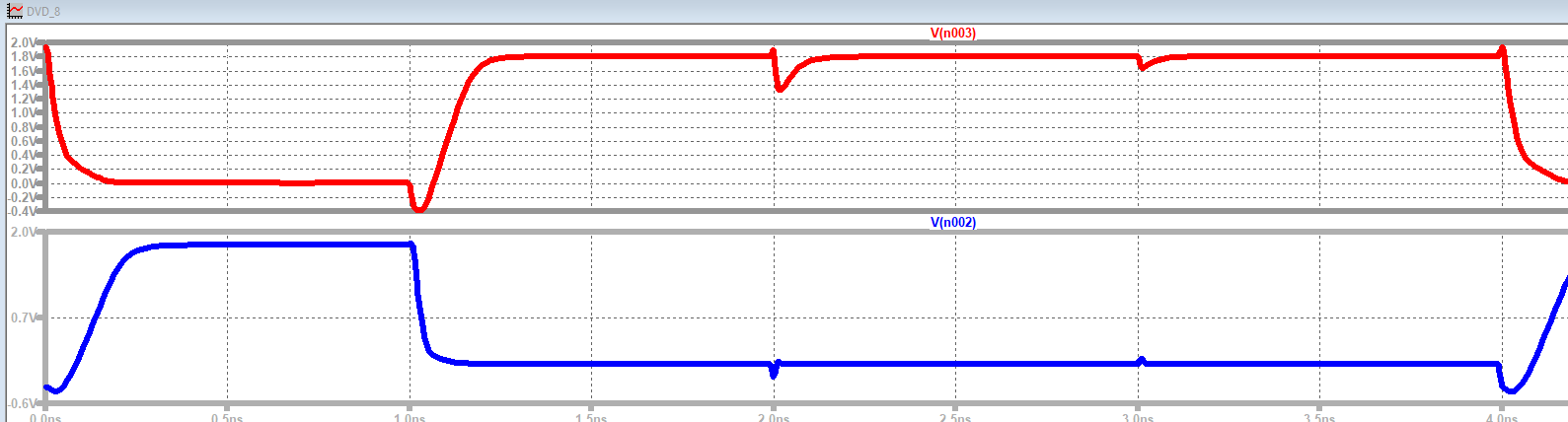
**Circuit:**

****

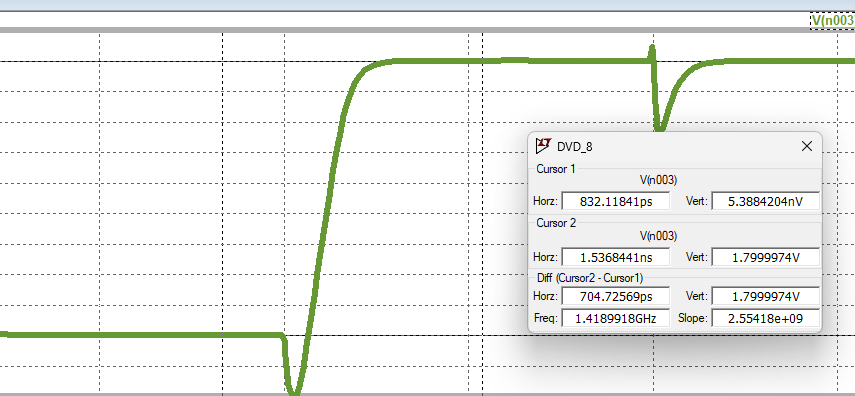
****

**Red: AND**

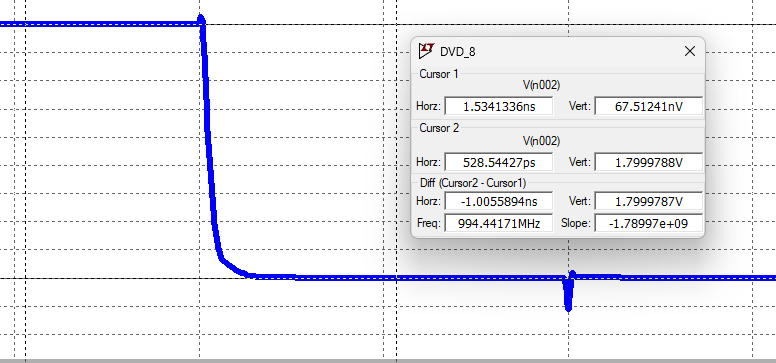
**Blue: NAND**

****

**Voltage Swing NAND:**

****

**Voltage Swing AND:**

****

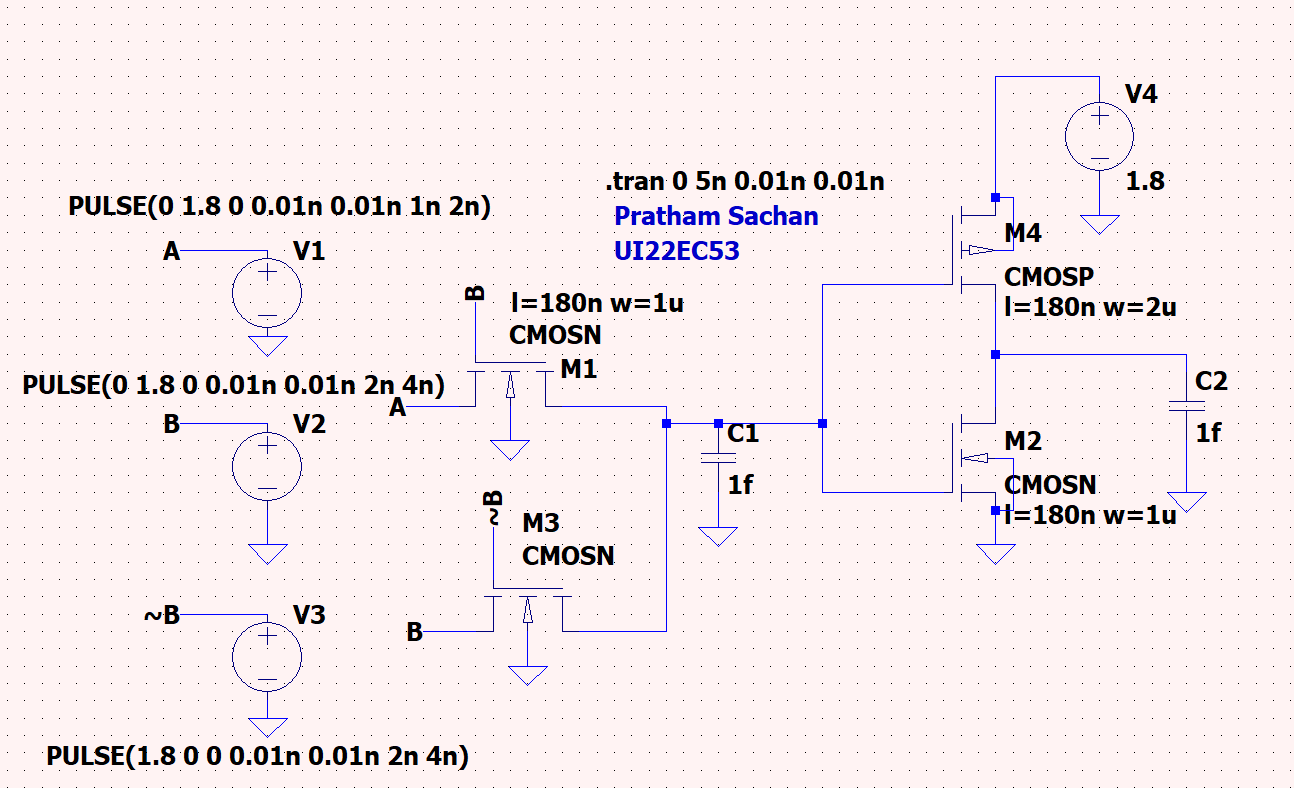
**Conclusion:**

**Assignment: 9**

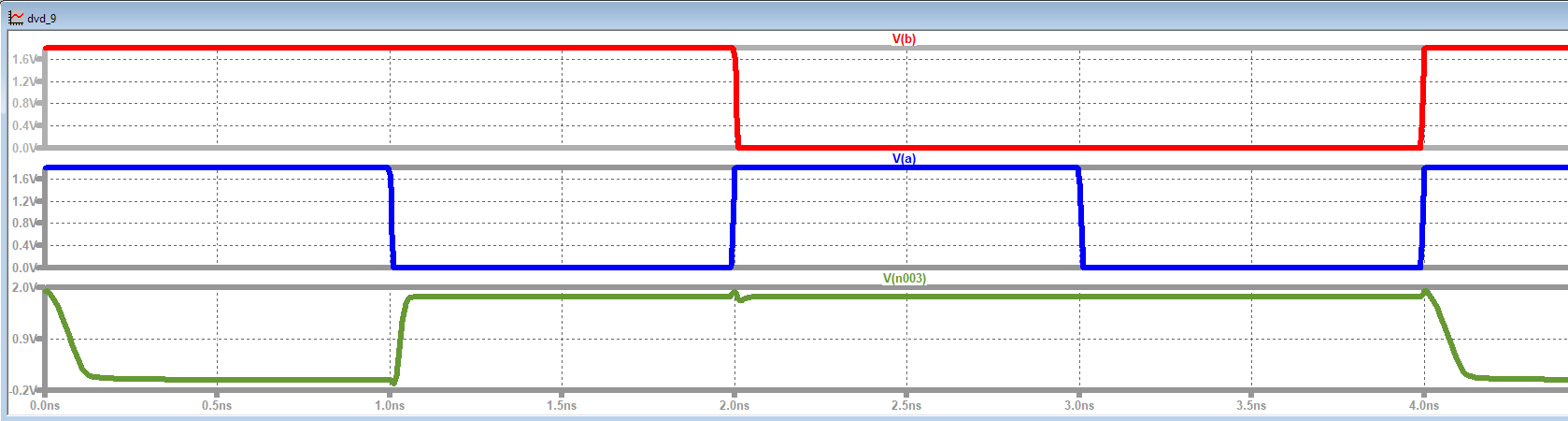
**Aim:** To design and implement basic gates with pass transistor logic and observe the output voltage swing, also restore the output voltage swing by connecting an inverter at the output node and measure static power dissipation.

**Circuit:**

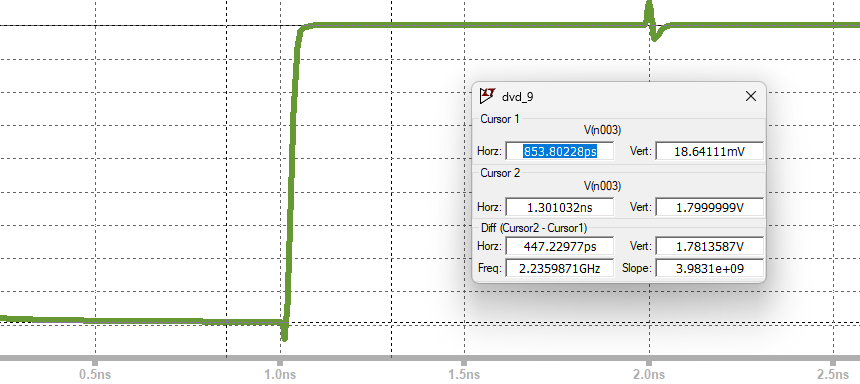
**NAND:**

****

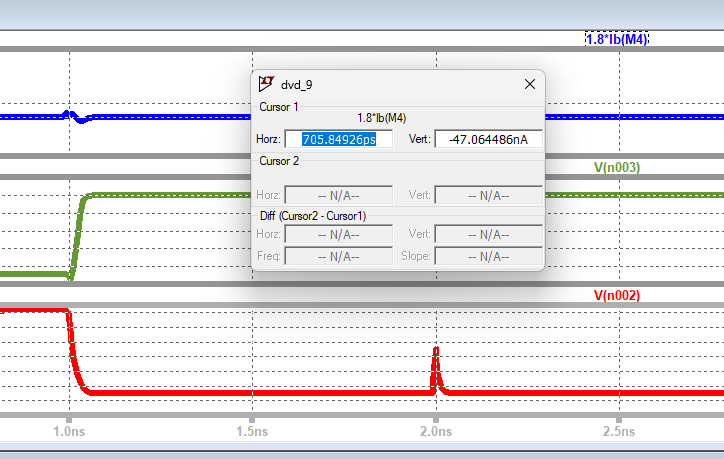
**Output:**

****

**Voltage Swing: 1.78V**

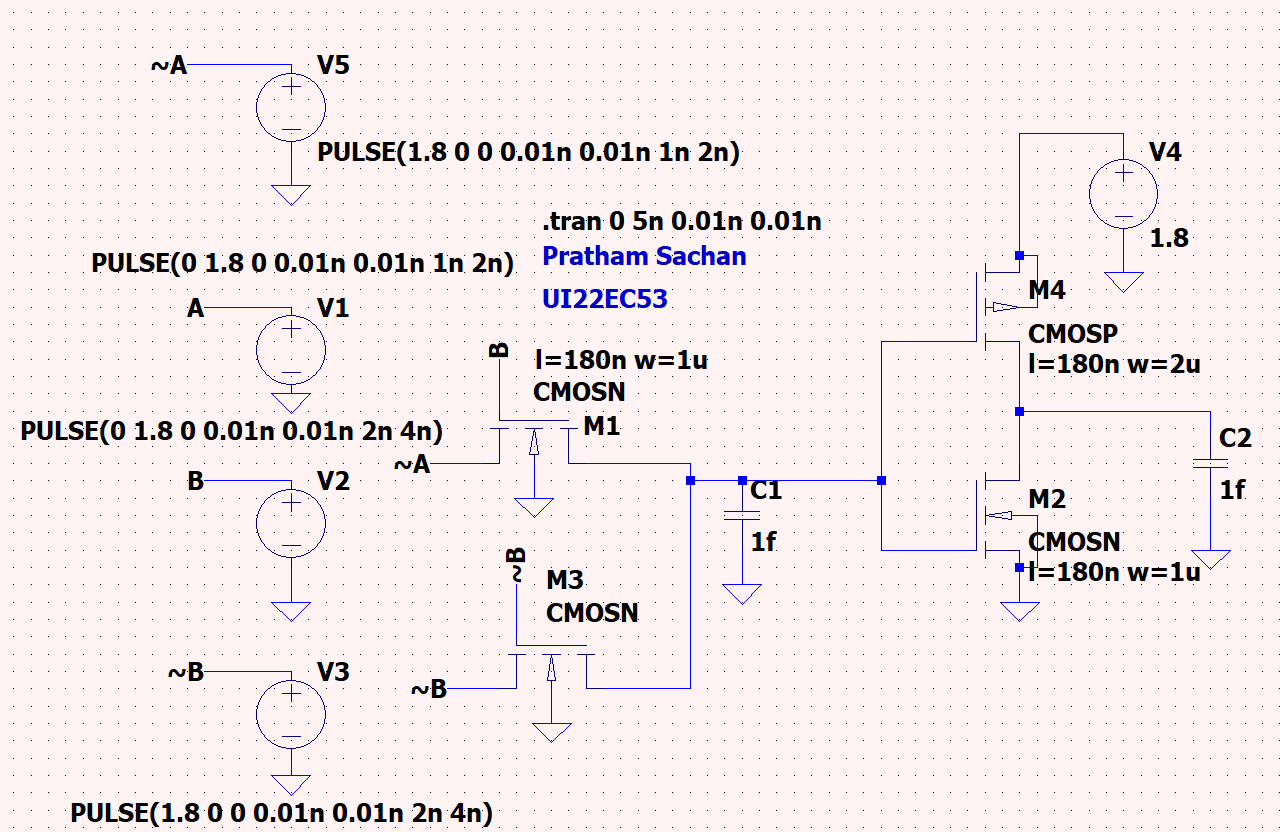
****

**Static Power:**

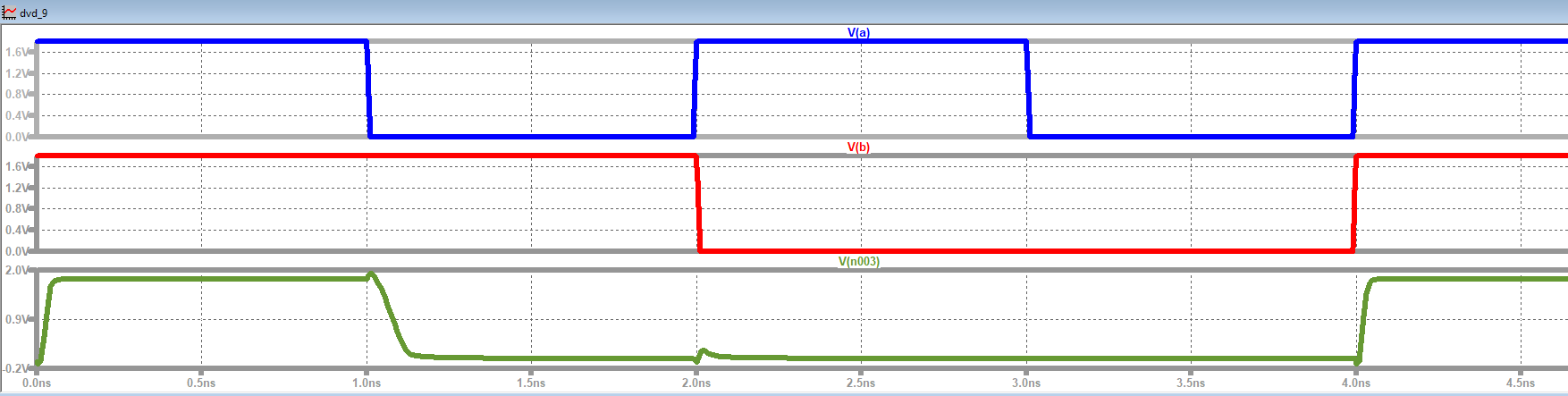
****

**Power:** -47.06nA

**AND Circuit:**

****

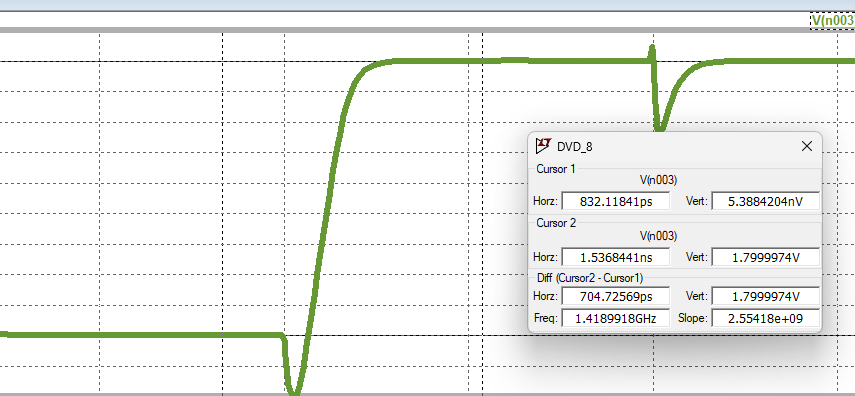
**Output:**

****

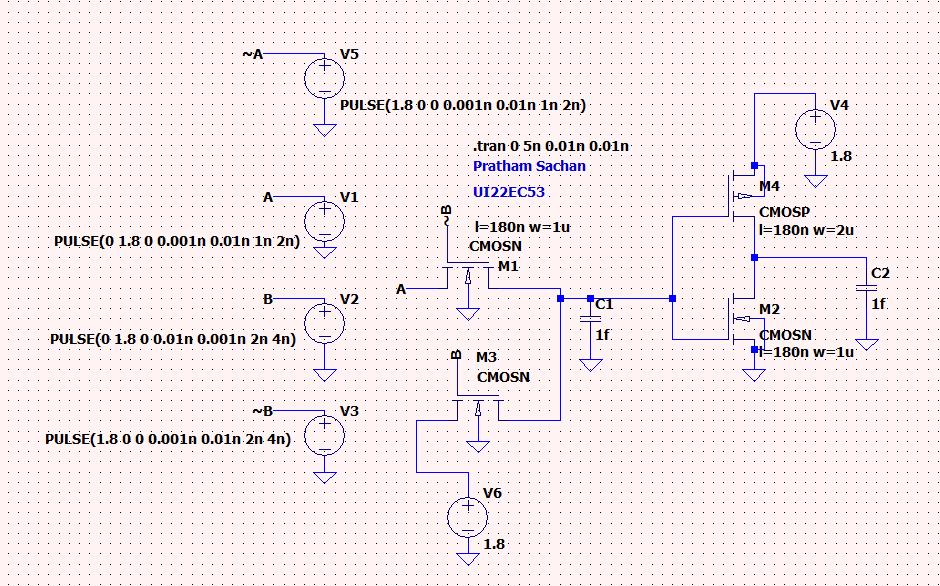
**Static Power: -61.41nA**

****

**Voltage Swing:**

****

**NOR Circuit:**

****

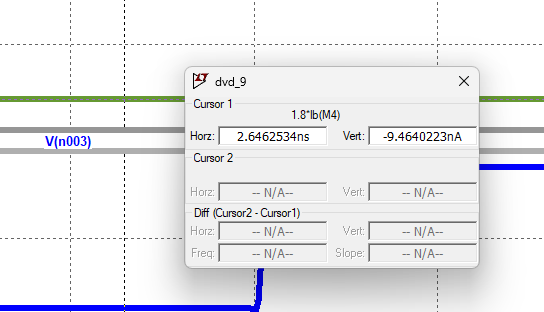
**Output:**

****

**Voltage Swing:**

****

**Static Power:**

****

**Conclusion:-**

In this experiment, we successfully designed a 2-input AND/NAND gate using DCVSL and observed its output voltage swing.

**Observations:**

1. AND Gate: Observed VTH = 0.73V (lower than expected 0.9V) due to stronger pull-down from parallel NMOS.
2. NAND Gate: Observed VTH = 0.93V (slightly higher than 0.9V) due to weaker pull-down from series NMOS.
3. Output Voltage Swing: Maintained between 0V and 1.8V for both gates.

**Assignment 10**

**Aim:**To design and implement a 2x1 MUX and the Boolean function F = AB + C using Transmission Gate Logic (TGL) in LT-Spice and observe the output.

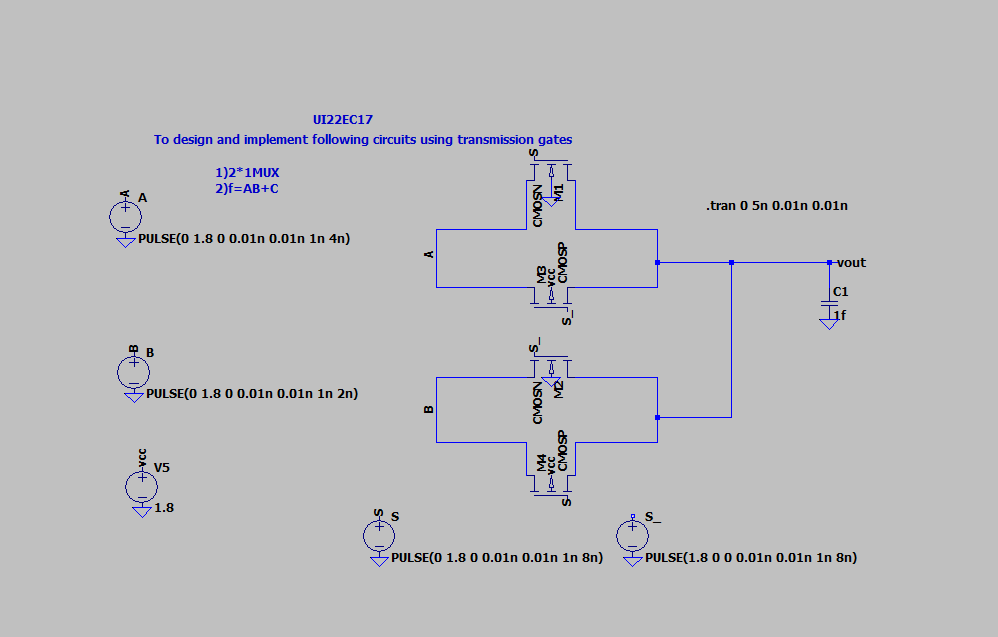
**Components Used:** LT-Spice Software

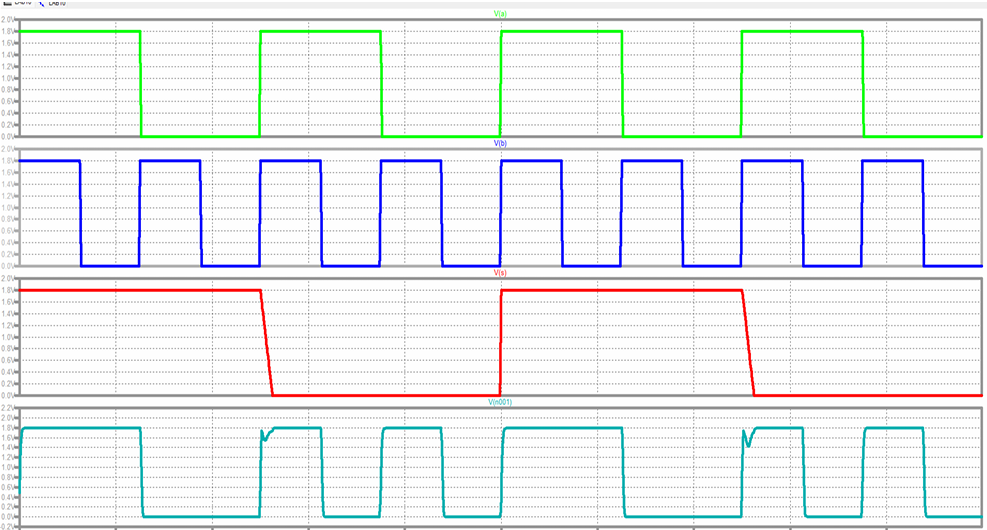
**Key Features of Transmission Gate Logic (TGL)**

* Efficient signal passing with minimal voltage degradation.
* Uses both NMOS and PMOS to pass logic "1" and "0".
* Low power dissipation and high-speed switching.

**Design Approach**

* First transmission gate passes A when S = 1.
* Second transmission gate passes B when S = 0.

**Circuit Diagram**

**Output Waveform  
**

* **Green → A**
* **Blue → B**
* **Red → S (Select Line)**
* **Cyan → Output (Y)**

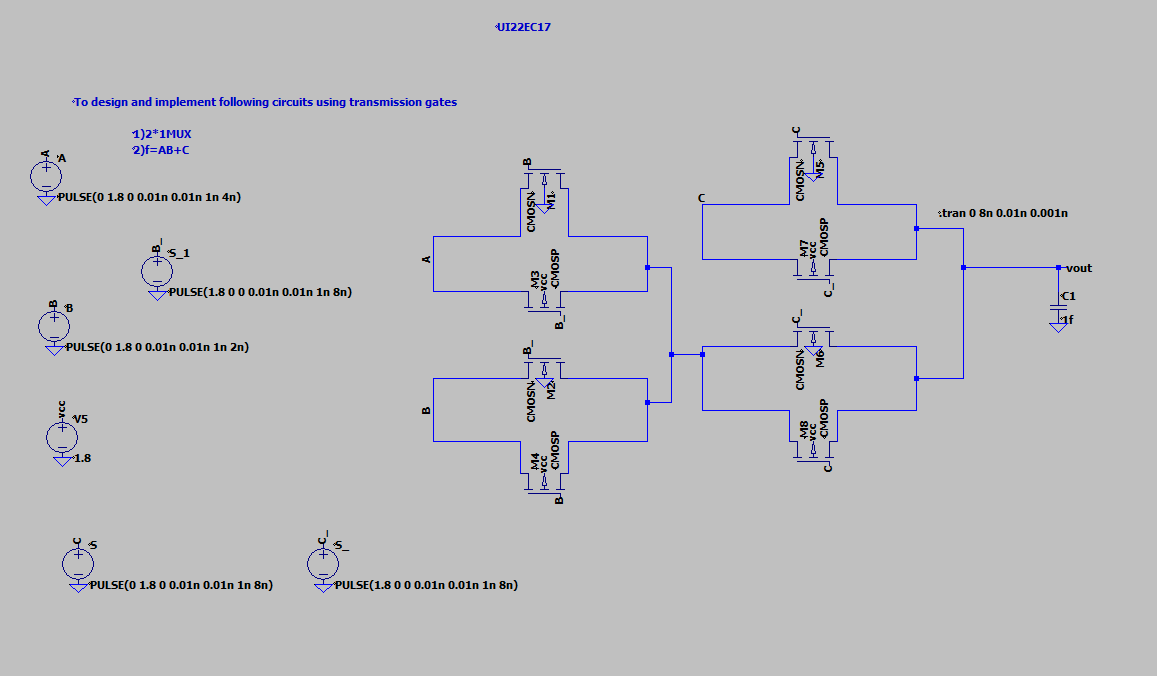
**2. Implementation of Boolean Function F = AB + C**

**Boolean Expression**

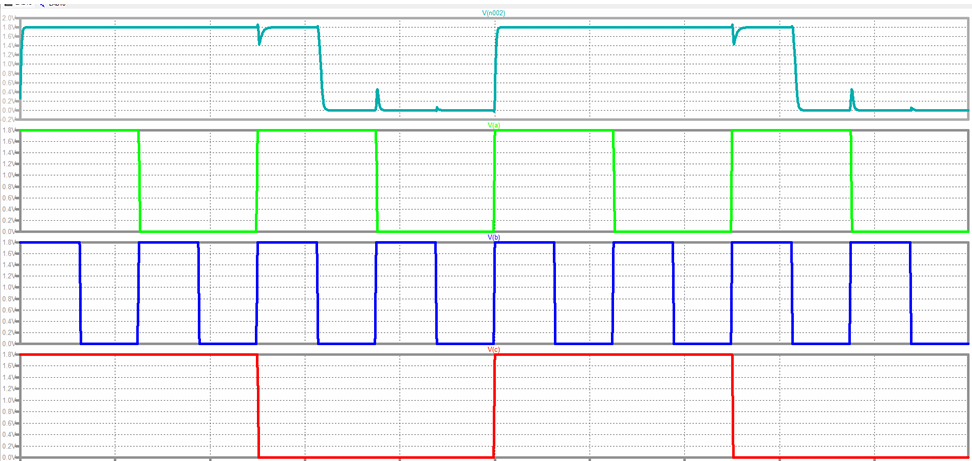
**F=AB+C*F*=*AB*+*C***

**Design Approach**

1. **Implement AB using a MUX.**
2. **Perform OR operation between AB and C using another MUX.**

**Circuit Diagram  
**

***(Insert LT-Spice schematic showing inputs A, B, C, and output F using transmission gates.)***

**Output Waveform  
**

* **Green → A**
* **Blue → B**
* **Red → C**
* **Cyan → Output (F)**

**Conclusion**

* **Both the 2x1 MUX and the Boolean function F = AB + C were successfully implemented in LT-Spice using Transmission Gate Logic (TGL).**
* **The 2x1 MUX correctly selected inputs based on the select line S.**
* **The function F = AB + C was efficiently realized by combining AB and C using transmission gates.**
* **The results confirm the low power dissipation and high-speed switching capabilities of TGL.**

**Assignment 11**

**Aim:**To design and implement dynamic NAND or NOR gates using dynamic logic—both with and without bleeder transistors—and observe their output behavior

**Components Used**

* CMOS ICs (for PMOS and NMOS transistors or their simulation)
* Power Supply and Ground
* Capacitors (for dynamic storage)
* Simulation Tool: **LTSpice**

**Theory**

**Dynamic Logic Overview**

Dynamic logic uses a two-phase clocked approach: precharge and evaluation. It simplifies circuit design by reducing transistor count, mainly using an NMOS pull-down network and a PMOS precharge transistor.

* **Precharge Phase (CLK = 0):**
  + PMOS transistor is ON → charges the output node to logic HIGH.
  + NMOS network is OFF.
* **Evaluate Phase (CLK = 1):**
  + PMOS turns OFF.
  + NMOS network activates and conditionally pulls the output LOW, depending on input values.

**Without Bleeder Transistor**

* During evaluation, if the output is not pulled LOW, it retains the precharged value.
* Over time, charge leakage can degrade this stored voltage, leading to unreliable outputs.

**With Bleeder Transistor**

* A weak PMOS or a high-value resistor from VDD to the output node helps preserve the logic HIGH level.
* It counters charge leakage when no discharge path exists.
* The bleeder is too weak to interfere when NMOS pulls the output LOW.

**1. Dynamic NAND Gate**

**A. Without Bleeder Transistor**

**Circuit Description:**

* PMOS Precharge Transistor: Controlled by CLK; ON during CLK = 0.
* NMOS Evaluation Network: Two NMOS transistors in series (for inputs A and B).
* Output Node: Connected between PMOS and NMOS networks.

**Operation:**

* **CLK = 0 (Precharge):** Output = HIGH.
* **CLK = 1 (Evaluate):**
  + If A = B = 1 → NMOS path conducts → Output = 0.
  + For all other input combinations → NMOS path breaks → Output remains HIGH.

**Issue:**

* If NMOS doesn’t conduct, the output floats and slowly degrades due to leakage.

**B. With Bleeder Transistor**

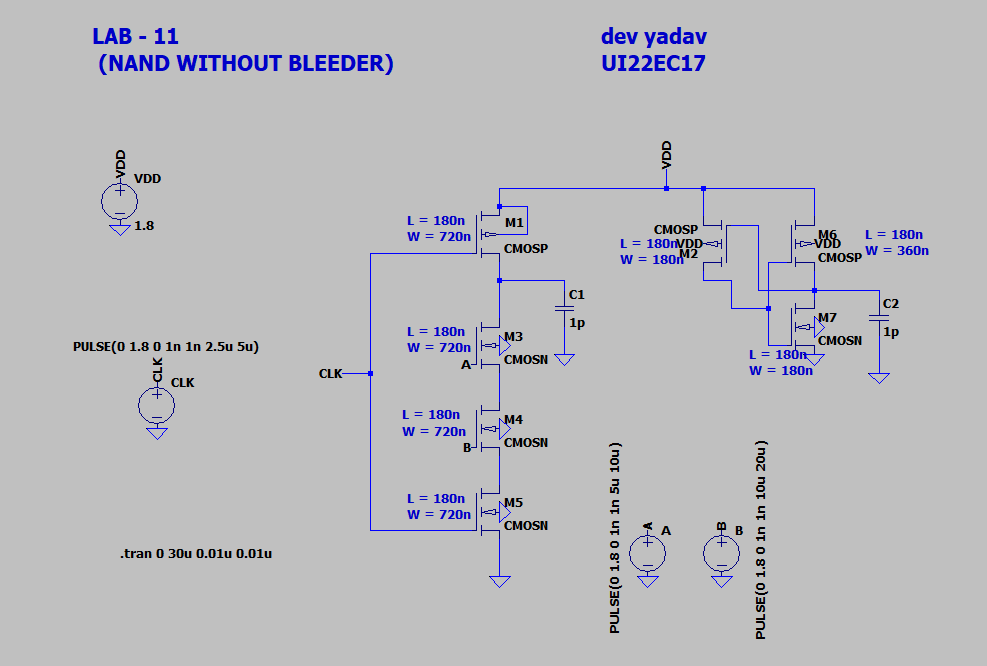
**Circuit Addition:**

* Weak PMOS or high-value resistor connected from VDD to the output node.
* Either always ON or lightly controlled.

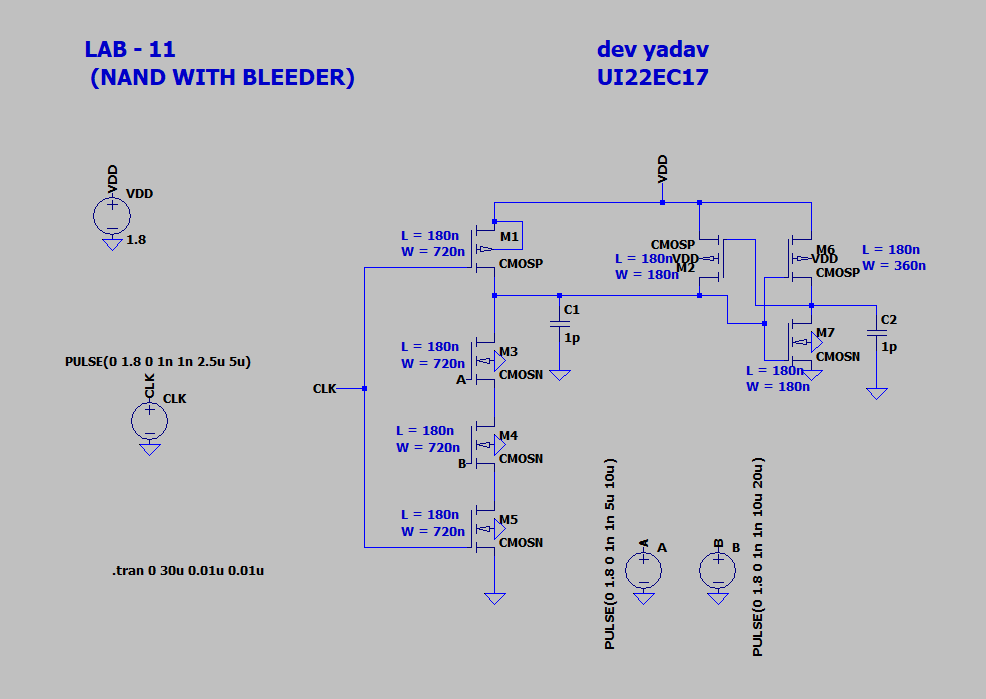
**Operation:**

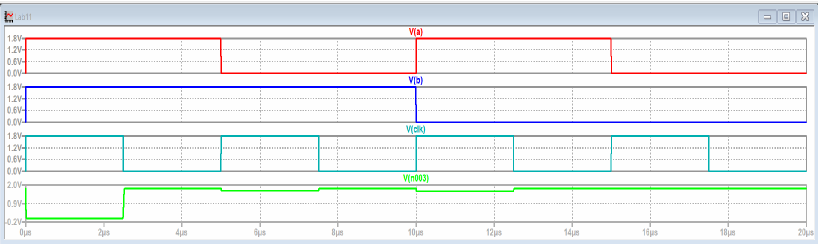
* During CLK = 0 → Precharge PMOS sets output HIGH; bleeder supports charging.
* During CLK = 1:
  + If no discharge path (A ≠ 1 or B ≠ 1) → Bleeder maintains output HIGH.
  + If discharge path (A = B = 1) → NMOS discharges output → Bleeder cannot resist.

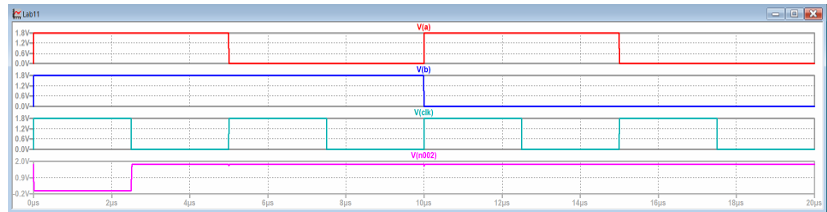
**Circuit Diagram (Without Bleeder):**

****

**Circuit Diagram (With Bleeder):**



**Output (Without Bleeder):**  


**Output (With Bleeder):**

**2. Dynamic NOR Gate**

**A. Without Bleeder Transistor**

**Circuit Description:**

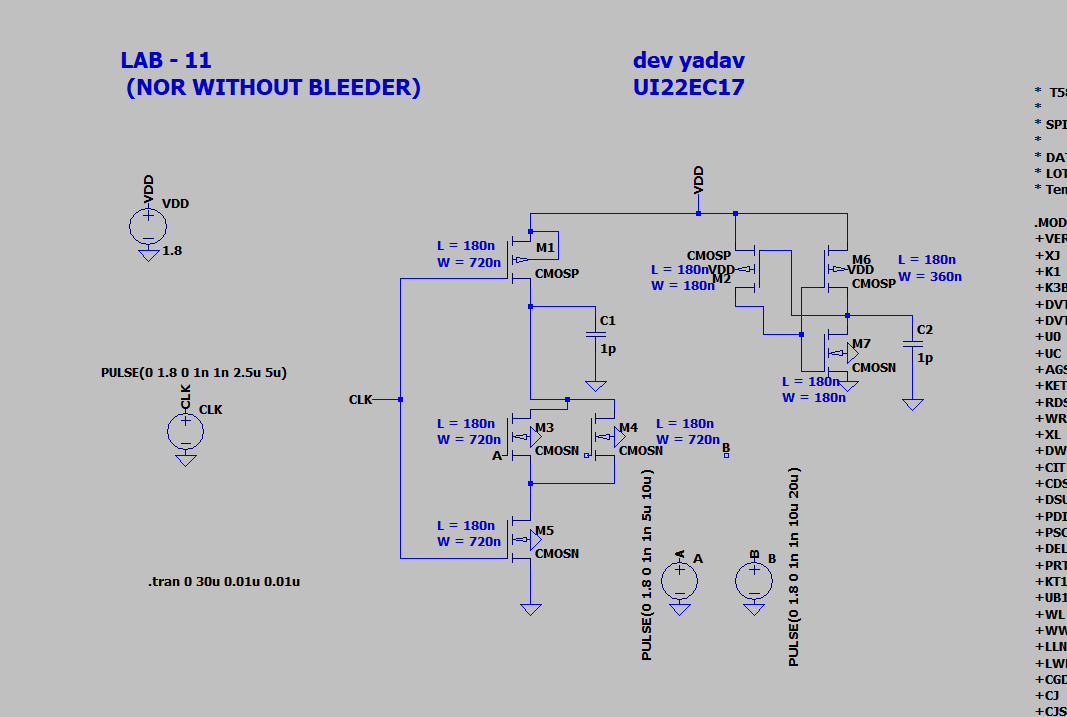
* PMOS Precharge Transistor: Charges output HIGH during CLK = 0.
* NMOS Evaluation Network: Two NMOS transistors in **parallel** for inputs A and B.
* Output Node: Shared between PMOS and NMOS.

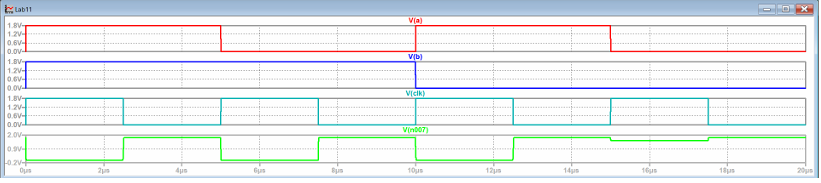
**Operation:**

* **CLK = 0 (Precharge):** Output = HIGH.
* **CLK = 1 (Evaluate):**
  + If A = 1 or B = 1 → Any NMOS conducts → Output = 0.
  + If A = B = 0 → Output retains HIGH.

**Issue:**

* Without a bleeder, charge slowly leaks from output when NMOS doesn’t conduct.

**Circuit Diagram (Without Bleeder):**  
**

**Output (Without Bleeder):**  
**

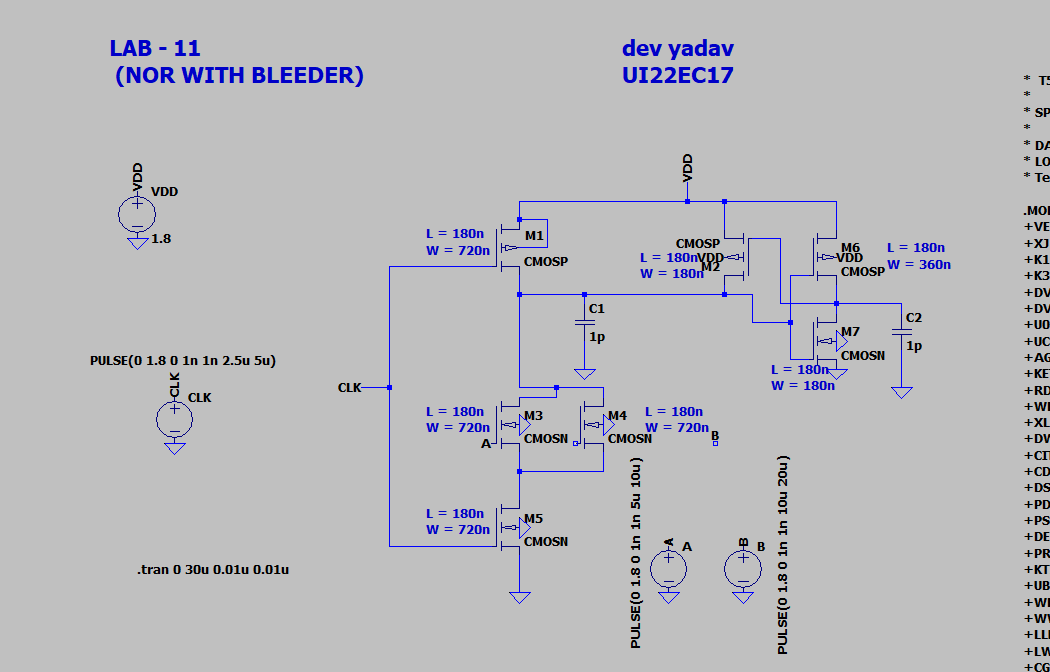
**B. With Bleeder Transistor**

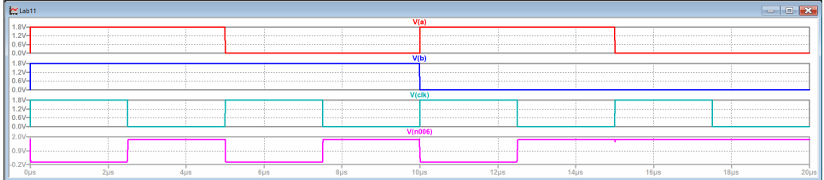
**Circuit Addition:**

* Weak PMOS from VDD to output node.

**Operation:**

* CLK = 0 → Precharge to HIGH; bleeder supports.
* CLK = 1:
  + If A = B = 0 → No NMOS conduction → Bleeder maintains HIGH.
  + If A = 1 or B = 1 → NMOS conducts → Output pulled LOW.

**Circuit Diagram (With Bleeder):**  
**

**Output (With Bleeder):**  
****Conclusion**

* Dynamic logic offers faster operation and reduced transistor usage.
* NAND and NOR gates can be efficiently implemented using NMOS logic evaluation and PMOS precharge.
* Bleeder transistors enhance output stability by compensating for leakage.
* **Limitations:** Sensitive to noise, dependent on clock, and prone to charge loss.
* **Applications:** Widely used in high-speed, low-power integrated circuits like CPUs, memory cells, and digital processors.

**Assignment 12**

**Aim:**To Design and implement 8 transistor implementations of clocked SR latch and analyze its functionalities.

**Components used:-**Pmos transistor,Nmos transistor,dc supply,pulse signal generator,ground connection,measurement tools.

**Theory:**  The SR latch (Set-Reset latch) is a basic building block in digital electronics used for bistable memory storage. A clocked SR latch only responds to inputs when the clock signal is active, adding a level of control.

Overview of SR latch:-

|  |  |  |  |
| --- | --- | --- | --- |
| S | R | Q | Q’ |
| 0 | 0 | Hold | Hold |
| 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | Invalid | Invalid |

A clocked SR latch activates only when a clock signal (CLK) is high.This prevents accidental state changes and synchronizes latch behavior with a system clock.

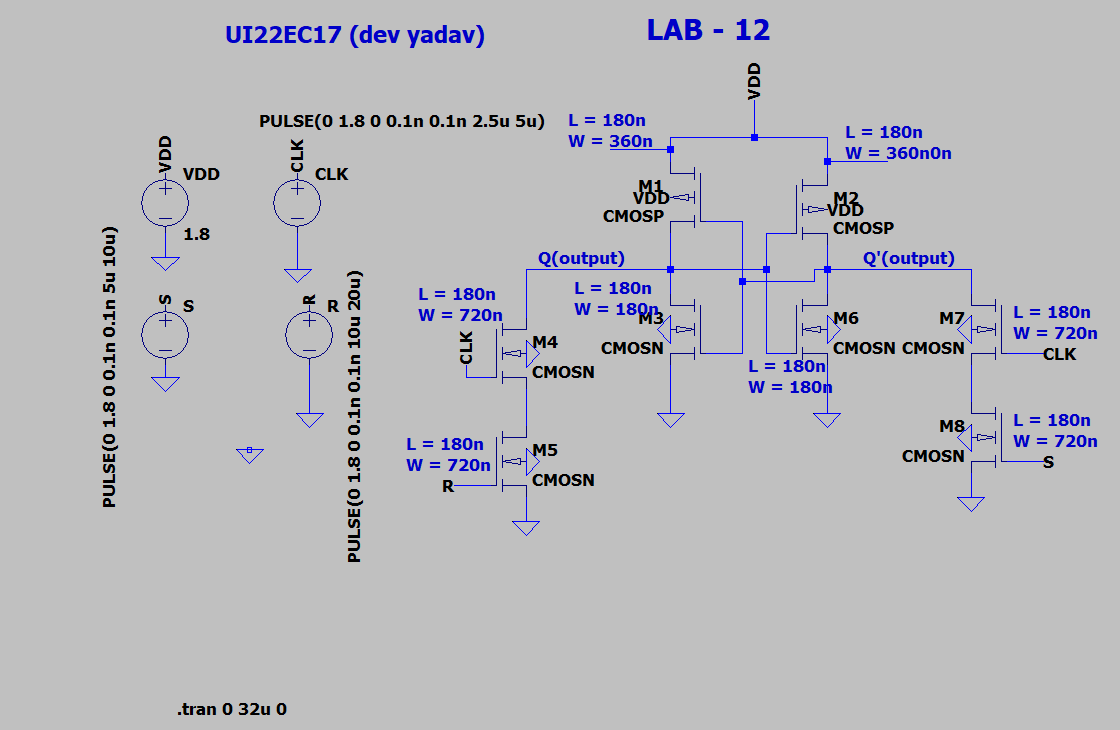
For this the requirement is as follows:-

2 cross-coupled inverters form the bistable core (4 transistors: 2 PMOS, 2 NMOS).

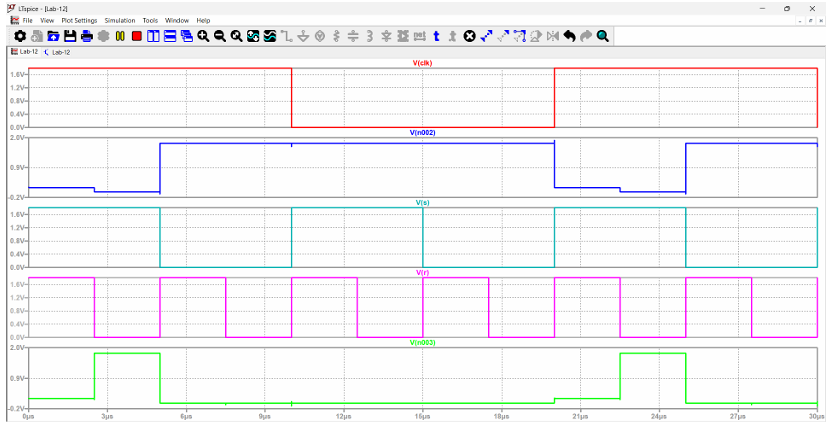
2 clocked NMOS pass transistors to control Set and Reset signals.

2 pull-up PMOS transistors or part of clocked logic for enabling based on CLK.

**Circuit Diagram:-**

****

**Output:-**

****

**Observation:-**

1)S = 1, R = 0, Clock = 1:-

Set condition: Q goes high, Q' goes low

2)S = 0, R = 0, Clock = 1:-

Hold condition: Q and Q' maintain previous state

3)S = 0, R = 0, Clock = 1:-

Hold condition: Q and Q' maintain the reset state

4)S = 1, R = 0, Clock = 1  
Set again: Q goes high, Q' goes low

The latch only responds to S/R changes when the clock is high.When the clock is low, outputs remain stable despite input changes.

**Conclusion:-** This lab demonstrates the correct operation of a clocked SR latch using 8-transistor logic. The latch responds appropriately to set, reset, and hold conditions only when the clock is high, with Q and Q' maintaining complementary outputs. Minor glitches were observed but did not affect overall functionality.