University of the Philippines
Microelectronics and Microprocessors Laboratory

Lab Module 01: The MOSFET Switch and The Inverter

The inverter is the most primitive logic gate that is commonly used for digital circuits. It's used for both processing data (i.e. inverting for twos complement), generation of clocks, and even buffering of signals. These will become clear in the near future. From the lecture, an inverter consists of MOS transistors that function like switches. In this lab we will study the characteristics of a MOS transistor that acts like a switch. Later, we will it into an inverter where we will also study both its DC and transient characteristics.

What you will learn in this lab?

- Study and characterize the MOS transistor acting as a switch
- Build an inverter and understand its DC and transient characteristics

A few reminders for any lab:

You will also be provided with an answer sheet that matches the current lab.

Each lab will always have **4 main sections**:

- **Review** – this section can be treated like a "pre-lab" where you need to recall some of the previous concepts in your earlier courses or those that were discussed in the lecture.
- **Training** – this section guides you through some skills or concepts that you must learn. They could be some tutorial on how to extract information using the tool, or some interesting design philosophies.
- **Exercises** – this section contains exercises that leave you to do most of the work and thinking. Sometimes you need LTspice for it, and sometimes they could be theoretical. This is to test what you have learned.
- **Supplementary reading** – these are optional content. However, the interested students have the liberty to go through some short essays and exercises. It's to further help you appreciate what IC design is.

There will be special boxes that you need to watch out for!

BLUE BOXES: Aside from the main discussion, these blue boxes contain useful notes and discussions. They can help you understand and appreciate the current topic at hand.

RED BOXES: Contain notes that you need to watch out for. Some of these may be warnings that pertain to some limitations of our simulator. Others could be warnings on how to use particular circuit models. These are important to avoid confusion in some topics.

GREEN BOXES: Contain questions for a particular training task. These are mandatory questions that need to be answered and they are found in Part II of the lab. It's hard to miss them as they are also part of the answer sheets provided to you. These are meant to fortify your understanding of some concepts.

Part I: Review

Let's recall a few concepts from the lecture. Answer the following questions below.

1. List 3 differences between long-channel and short-channel devices. Excluding the length of the transistor.
2. What's the equation for the saturation current for both long-channel and short-channel devices?
3. What's a suitable circuit model of a MOSFET acting as a switch?
4. What's an inverter? Draw the schematic of a CMOS inverter.

Part II: Training

A. Switch Schematic

In this lab we will be investigating both long and short channel transistors but we will focus short channel transistors first. Short-channel transistors are usually better for digital circuits. The long-channel will be an exercise for you to investigate as part of this module.

1. **Create a dedicated folder to store the schematics that will be used in this lab module**. Preferably name the folder as lab01.
2. **IMPORTANT!** Make sure to download the transistor_models.txt file found in this link. This file holds the NMOS and PMOS transistor models for both long and short channels. **Save this in the same directory where you will be saving your schematics**. We will use these transistor models for all our labs. These were taken from http://cmosedu.com/.

DISCUSSION:
It is worthwhile exploring the contents of this file. It consists of the transistor models for the long-channel and short-channel transistors. For the long-channel transistors, we have NMOS (N_1u) and PMOS (P_1u) transistors. If you scroll down a bit you should see the models for short-channel NMOS (N_50n) and PMOS (P_50n) transistors. The short-channel devices have more parameters than the long-channel models. This is because the short-channel devices are more difficult to model due to the number of non-idealities that need to be considered. These models were developed by the Berkeley Short-channel IGFET Model (BSIM) group. They specialize in creating transistor models that can be used by SPICE simulators. More information can be found in their site: https://bsim.berkeley.edu/.

3. **Create a new schematic then save it as "lab01_nmos1u_characterization"**.
4. **Instantiate the transistor seen in Figure 2.1**.
   a. Click on the component symbol and look for the nmos4 component to bring up the NMOS transistor.
   b. Instantiate this into your schematic.

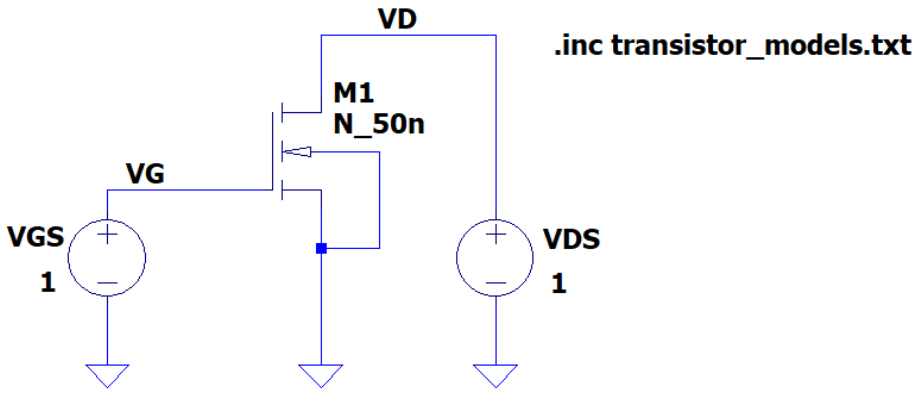


Figure 2.1 DC Analysis Setup for NMOS Transistor

5. **Include the transistor models for this schematic**.
   a. **Click on the SPICE Directive** .op icon in the shortcut icons. Figure 2.2 should appear.
   b. **Fill in the textbox with: ".inc transistor_models.txt"**
   c. **Click OK then place the generated text in an empty space** in your schematic.

You may follow where it's placed in Figure 2.1. This SPICE directive tells LTspice to include the transistor_models.txt file every time we need to simulate our transistors.

WARNING: Do not forget to add the transistor_models.txt file into our simulations. There's a default model in LTspice but it's close to an ideal model and not something that was developed by BSIM.

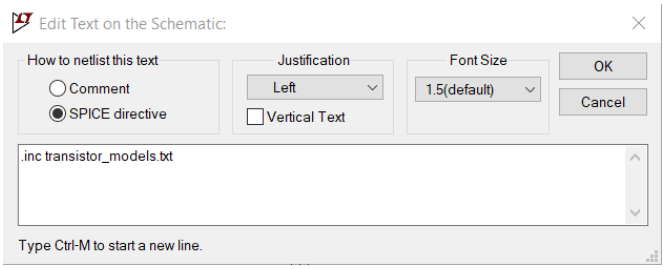


Figure 2.2 Inserting SPICE Directive Textbox

6. **Right-click on the transistor and change the model name NMOS to N_50n**. When we run simulations, LTspice will look for the parameters for the short-channel NMOS transistor with the name N_50n.
7. **Set the length to 50n, and the width to 500n**. The length and widths are relative to a meter, that means 50n pertains to 50 nm while 500n pertains to 500 nm.
8. Instantiate the voltages sources, connect the wires, and label the nodes accordingly.
9. **Don't forget to rename the voltage sources as V_{GS} and V_{DS}.**
10. Set V_{GS} = 1V and set V_{DS} = 1V
11. Great! You're done with the circuit setup.

B. MOS Switch DC Characteristics

First let's study the DC characteristics of the NMOS switch. Particularly we are interested how V_{GS} and V_{DS} control the drain current.

1. **Do a DC sweep** varying VDS from 0V to 1V and in 1mV increments.
2. **Run the simulation then plot the drain current**.
   a. If you hover your mouse close to the drain, a current probe should appear.
   b. You should be able to get Figure 2.3. This plot tells us how the I_{DS} changes with varying V_{DS} and fixed VGS. The background was changed to black for clarity.

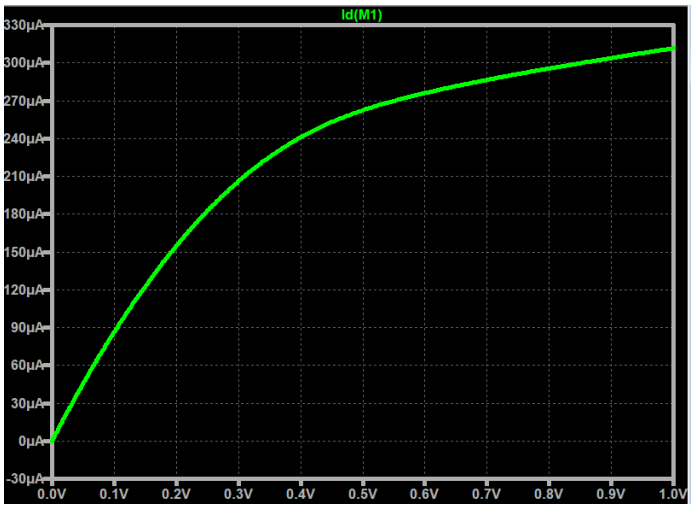


Figure 2.3 Short-channel NMOS IDS vs. VDS with VGS = 0V

Question (Q2.1): Can you tell where $V_{DS,SAT}$ is?

The DC sweep plot only gives us information about how current changes with varying V_{DS} and a fixed V_{GS} input. This also shows us V_{DS} voltages that make the transistor in the linear region and the saturation region. When we get to analog circuits, we will revisit this figure again as this plot also tells us the small-signal output impedance of our circuit. Let's try to plot what happens when we also vary V_{GS} in a parametric DC sweep analysis. Do the following:

1. Using the same schematic, **open the Edit Simulation Cmd again**.
2. **Click on the 2nd source tab and input the following**:
   - Name of 2nd source to sweep: V_{GS}
   - Type of sweep: Linear
   - Start value: 0
   - Stop value: 1
   - Increment: 0.2
3. **Hit OK afterwards, then run the simulation**.
4. **Plot the drain current again and you should arrive with Figure 2.4**. This figure is called **the parametric DC sweep analysis** where the V_{GS} is said to be parametrized from 0V to 1V in 0.2V increments.

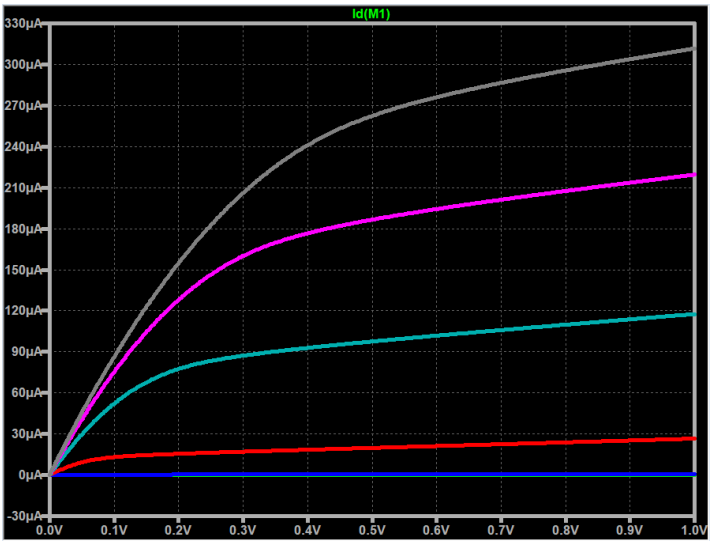


Figure 2.4 Parametric DC Sweep for the Short-channel NMOS Transistor

DISCUSSION:

This figure now gives us very interesting information of how the NMOS transistor behaves. Essentially it tells us how our IDS vs VDS curves change for different VGS inputs. Let's tinker with the plot so that you can explore and internalize what the behavior of our NMOS transistor for the given inputs. **Try out the following**:

- **Click on the name of the plot Id(M1) in your plotting plane to bring up the cursor**. Recall from the first lab, you can scroll the vertical cursor sideways and you should see the values changing. Press the up and down arrow keys to cycle through different plots, you should see the cursor (cross-hair) move from one plot to another. **Try pressing up multiple times until it reaches the top-most plot then go move the cursor side-ways**.
- **To know which V_{GS} plot we ear looking into, right click on either the horizontal or the vertical cursor**. A small window telling you which V_{GS} input you are looking into.

DISCUSSION (continued):
In this figure we can interpret and relate it to the fundamental theories we know. The bottom two plots (V_{GS} = 0.2V and 0V) have very low current compared to the next V_{GS} step (V_{GS} = 0.4V). This indicates that the threshold voltage is somewhere in between 0.4V and 0.2V. One thing for sure is that at V_{GS} = 0.2V the "switch" is still off. Recall that the transistor turns on if and only $V_{GS} \geq V_{THN}$. For higher V_{GS} values, we can estimate V_{THN} by looking at the transition from the linear region into the saturation region. For example, consider the red plot of Figure 2.4. That's when V_{GS} = 0.4V, and if we try to "eye-ball" the point of transition. It would roughly yield to around V_{DS} = 0.1 V. Now recall that $V_{DS,SAT} = V_{GS} - V_{THN}$ for the NMOS transistor. If we take the reading $V_{DS,SAT}$ = 0.1 V, because this is the point of transition, then it follows that $V_{THN} = V_{DS,SAT} - V_{GS} = 0.4 - 0.1 = 0.3$V. The actual threshold voltage (assuming V_{SB} = 0V) is at 0.28 V or 280 mV for the NMOS short-channel transistor. Take note, the "eye-ball" method of estimating the threshold voltage is a poor and unreliable method.

A final note regarding the plot. This plot is more useful for developing analog models because we can determine the output impedance from this figure. However, for a digital model it gives an idea how current behaves in DC when we try to compare the DC characteristics of the inverter later on. There are different ways to plot and extract better information. For example, we can simulate a *diode connected NMOS transistor* where the V_{GS} and V_{DS} are shorted together (here, $V_{GS} = V_{DS}$). Figure 2.5 shows this.

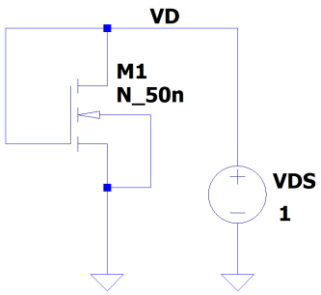


Figure 2.5 Diode connected NMOS transistor

This way we only have 1 DC source to sweep and this plot can give the current and *transconductance* (derivative of the current) exactly at saturation. We will get back to this later on. The point is that there are other ways to plot the current with different setups and we get valuable information depending on what we want to show. There's something noteworthy of an investigation in Figure 2.4. In the region when the transistor is on, how does current increase with increasing V_{GS}? IDS vs VDS (or VGS) for the diode connected transistor is shown in Figure 2.6. It shows a straight line which is consistent to the short-channel equation current:

$$I_D = \mu_n E_{crit} \cdot W \cdot C_{ox} \cdot \left(V_{GS} - V_{DS,SAT} - V_{THN}\right)$$

Figure 2.6 IDS vs VDS (or VGS) curve for diode connected NMOS transistor

Let's simulate the PMOS transistor and re-produce Figures 2.3 and 2.4. Do the following:

1. **Recreate the circuit in Figure 2.7**. Don't forget to include the following setups:
   a. **When you instantiate the PMOS transistor use pmos4 symbol from the component selection**. Then use the P_50n model.
   b. **Use a length of 50n and width of 500n**.
   c. **Go ahead and rotate the transistor twice (Ctrl+R) then flip or mirror it once (Ctrl+E)**. It should look like the one in Figure 2.5.
   d. **Be sure to connect the bulk to ground**!
   e. Do not forget to rename the voltage sources with the names in Figure 2.5.

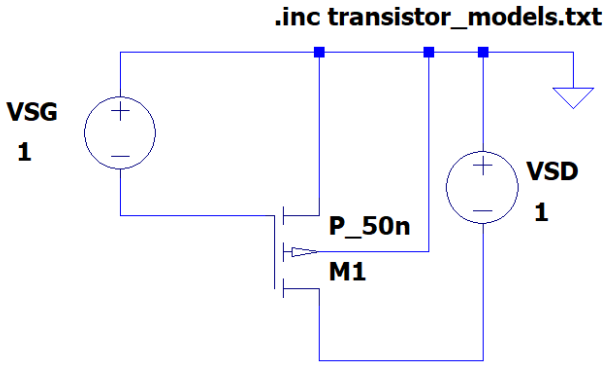


Figure 2.7 DC Analysis Setup for PMOS Transistor

2. First, simulate a DC sweep (not parametrized) of V_{SD} from 0V to 1V in 1mV increments. Let V_{SG} = 1V. Also, let's **get the source current**. You should get Figure 2.8.

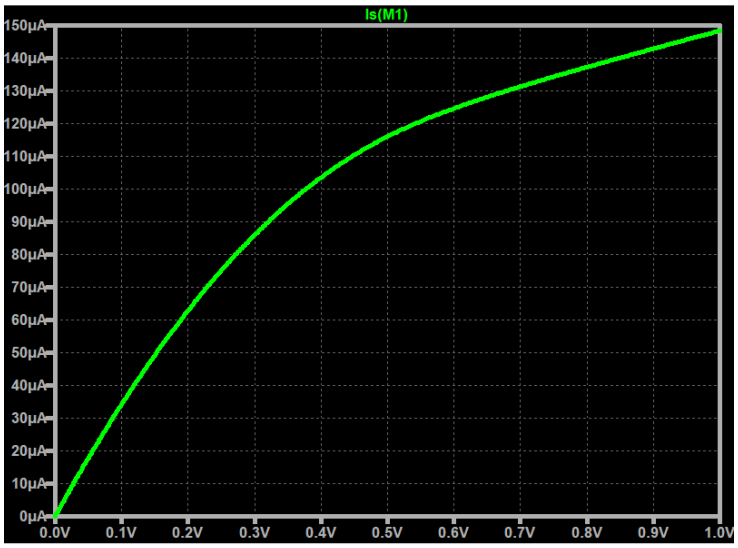


Figure 2.8 Short-channel PMOS ISD vs VSD with VSG = 1V

3. **Do a parametric DC sweep, with the same V_{SD} setting but with V_{SG} from 0V to 1V in 0.2V increments**. You should get Figure 2.9.

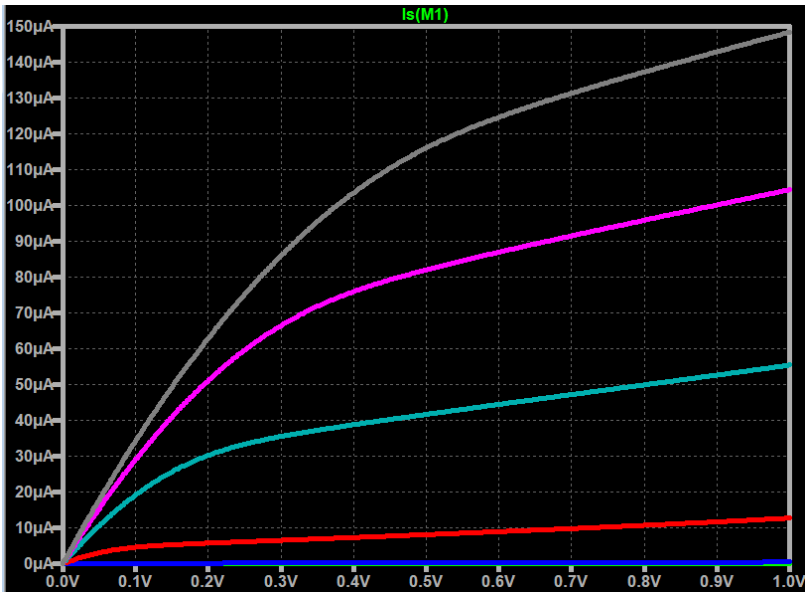


Figure 2.9 Parametric DC Sweep for Short-channel PMOS Transistor

Question (Q2.2):
Using the plots, you generated for the PMOS transistor. Answer the following questions:

1. Does the PMOS transistor switch on with increasing or decreasing V_{SG}?
2. With the figures you generated, does the current move from source to drain (SD) or drain to source (DS)? (Hint: Recall how LTspice defines current direction from Lab01. When you get the drain current, it's just the negative of the source current).
3. Use the "eye-ball" method to estimate V_{THP}. What is V_{THP}?

C. MOS Switch Transient Characteristics

The transient characteristic for a MOS switch is important because we will be characterizing the speed of our inverters later on. Essentially, we need to extract the input capacitance and the output resistance of our transistor. We shall tackle the output resistance of the transistor first. Consider the two circuits shown in Figure 2.10 and Figure 2.11. Figure 2.10 models our NMOS transistor as a switch. Here, the capacitor is initially charged to 1V. (V_C = 1V). When $V_{GS} \geq V_{THN}$, the switch will turn on connecting the resistor and the initially charged capacitor (currently holding V_C = 1V). The capacitor will discharge into the ground passing through the resistor in its path. The speed of discharge is the $R_{NMOS}C_L$ time constant. Eventually, the voltage across the capacitor will reach 0. This is exactly what happens to the NMOS transistor acting as a switch shown in Figure 2.11. Essentially, Figure 2.10 *models* the scenario happening in Figure 2.11. The NMOS transistor consists of both the R_{NMOS} resistor and the switch at the same time. This is depicted in the box in Figure 2.10. Let's try to extract this "modelled" R_{NMOS} resistor.

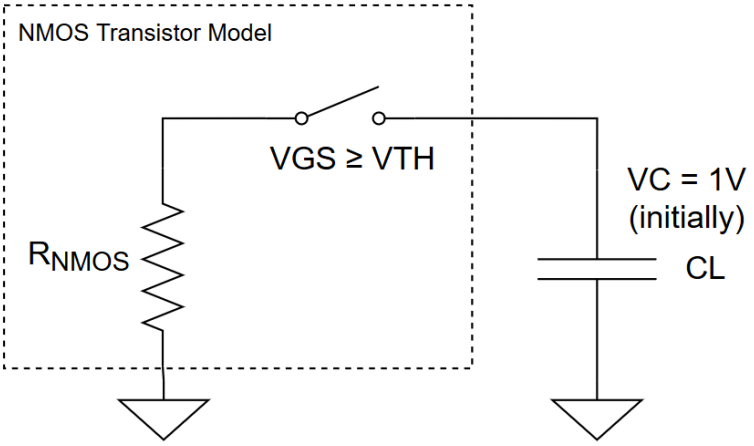


Figure 2.10 NMOS Switch Model

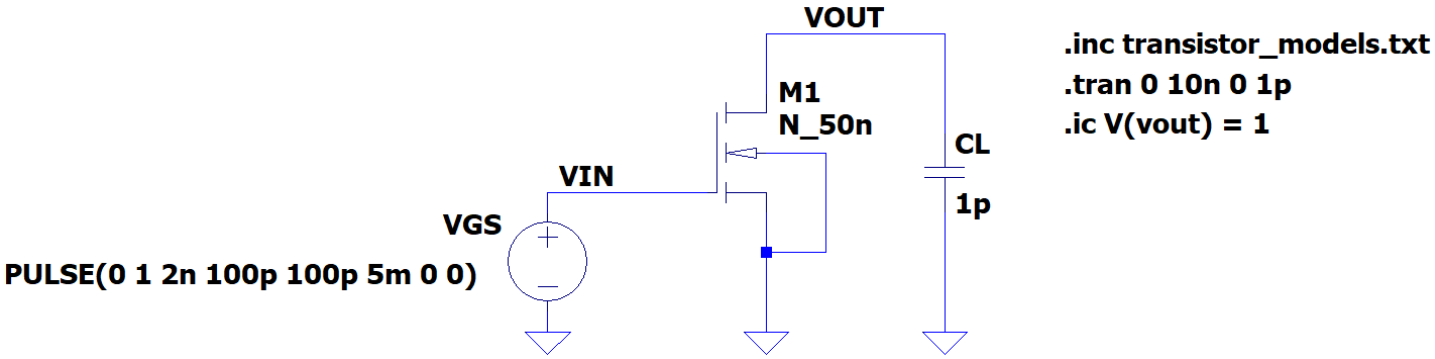


Figure 2.11 NMOS as a Switch

DISCUSSION:
Emphasis is given to the word "modelled" because this does not really describe the exact resistance of the R_{NMOS}. In fact, if you review the lecture slides and refer to our I_{DS} vs V_{DS} curves, the resistance of the transistor changes through time. For example, consider the time when the switch "just turned on". That is when V_{GS} = 1V, and V_{DS} = 1V. The current region of operation of the transistor would be in saturation. Recall Figure 2.4, right-most point in the pink plot. That's V_{GS} =1V, and V_{DS} = 1V. But when the capacitor starts discharging to ground, we are essentially moving along the pink plot moving towards the left until V_{DS} = 0V. At this point we are already at the linear region and the resistance is different from the saturation region. As we move along the pink curve, the effective resistance changes all throughout! Now, that's something difficult to model because we would need to use the equation from the lecture slides:

$$R_{eq} = \frac{\int_{V_1}^{V_2} R(V)dV}{V_2 - V_1} = \frac{\int_{V_{DD}}^{V_{DD}/2} \frac{V}{I_{DSAT} \cdot (1 + \lambda V)} dV}{-V_{DD}/2}$$

Equation 2.1 Effective capacitance of a MOS transistor

Oh no, we don't want that! Just kidding but it is important to internalize the implications of this equation: The equivalent resistance would be the integration of each resistance step of the plot in Figure 2.4. That's the gist of it! Don't worry, we'll use a more practical approach!

Follow the succeeding instructions:

1. **Re-create the circuit in Figure 2.11**. Make sure to include the following:
   - **Don't forget the .inc transistor_models.txt**
   - Use a nmos4, model with length and width as 50n and 500n respectively.
   - The capacitor si 1pF.
   - Create the input to generate a pulse with:
     - Vinitial = 0
     - Von = 1
     - Tdelay = 2n
     - Trise = 100p
     - Tfall = 100p
     - Ton = 5m
     - Tperiod = 0
     - Ncycles = 0
   - Set a transient analysis that stops in 10 ns in 1ps time resolution.
2. **Save this schematic as "lab01_nmos_rc1"**.
3. **We also need to set the initial conditions for the capacitor**.
   a. **Open a SPICE directive** .op **and type in: ".ic V(vout) = 1"**. This is shown in Figure 2.11.
   b. This line specifically tells LTspice that we are setting the voltage VOUT to be at 1V at the start of the simulation. It already assumes that the capacitor is holding enough charge to provide this.
4. **Run the simulation and plot VIN and VOUT**. You should get Figure 2.12. Nice! That wasn't hard right?

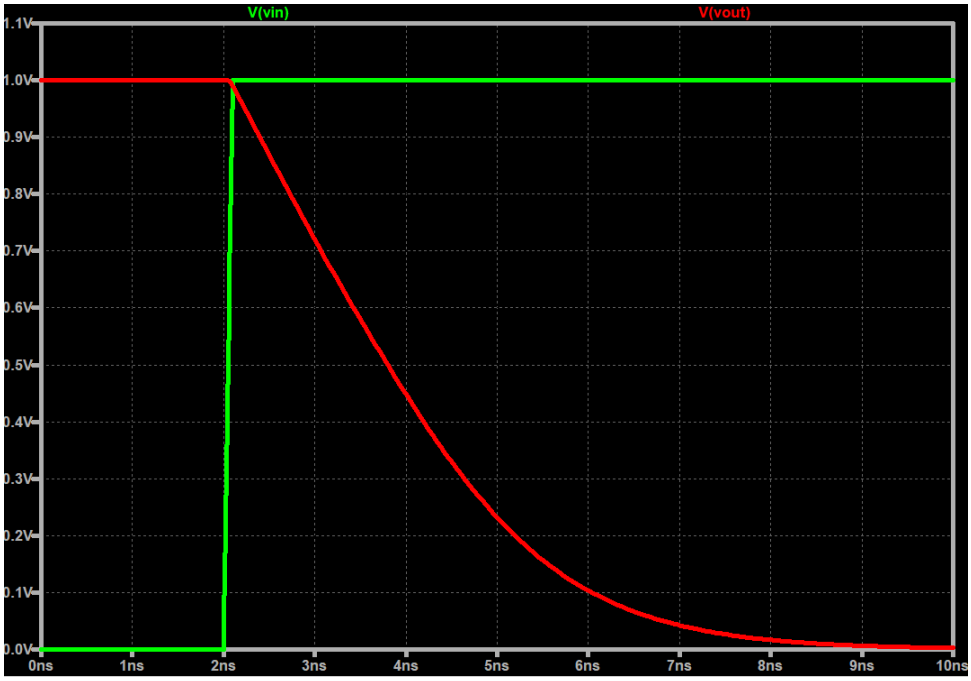


Figure 2.12 Simulation of NMOS Switch

5. **Now, take note of the propagation delay of the circuit**.
   a. Get the time difference between 50% of the output to the 50% of the input.
   b. Set your cursors to get the time when VOUT = 0.5V and VIN = 0.5V. Check the horizontal difference. You should get a reading that is close to 1.75 ns.

Alternatively, let's learn how to use the .MEAS spice directive. In the previous lab this was an optional task but if you did it, that would be great! The .MEAS spice directive helps us measure electrical parameters "automatically". Imagine if you had to do step 4 in multiple circuits? That would be tedious and time consuming. Follow the succeeding steps to learn how to set the .MEAS spice directives to measure the propagation delay.

1. **In the shortcut icons, click on the spice directive** .op.
2. **In the "Edit Text on the Schematic" window, type .meas**. Figure 2.13 shows this.
3. **Click on OK then place it on the side** such that it won't obstruct your circuit.

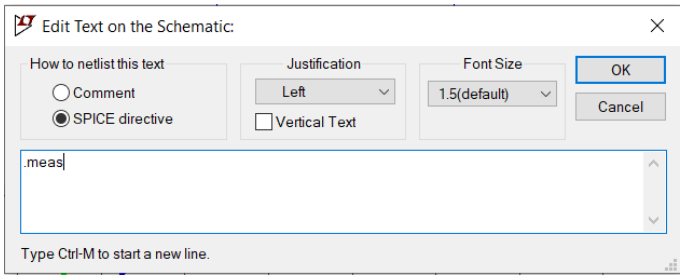


Figure 2.13 Edit Text on the Schematic Window

4. **Right-click on the .meas spice directive** (or text) in your schematic.
5. The ".meas Statement Editor" should appear. This is shown in Figure 2.14.
6. **Input the entries shown in Figure 2.14**. For clarity, this are listed below. Some entries can be selected from the drop-down list boxes. **Click OK when you are finished**.
   - Applicable Analysis: TRAN
   - Result name: t1
   - Genre: FIND
   - Measured Quantity: time
   - Point: WHEN – V(vin)
   - Right Hand Side: 0.5
   - CROSS: 1

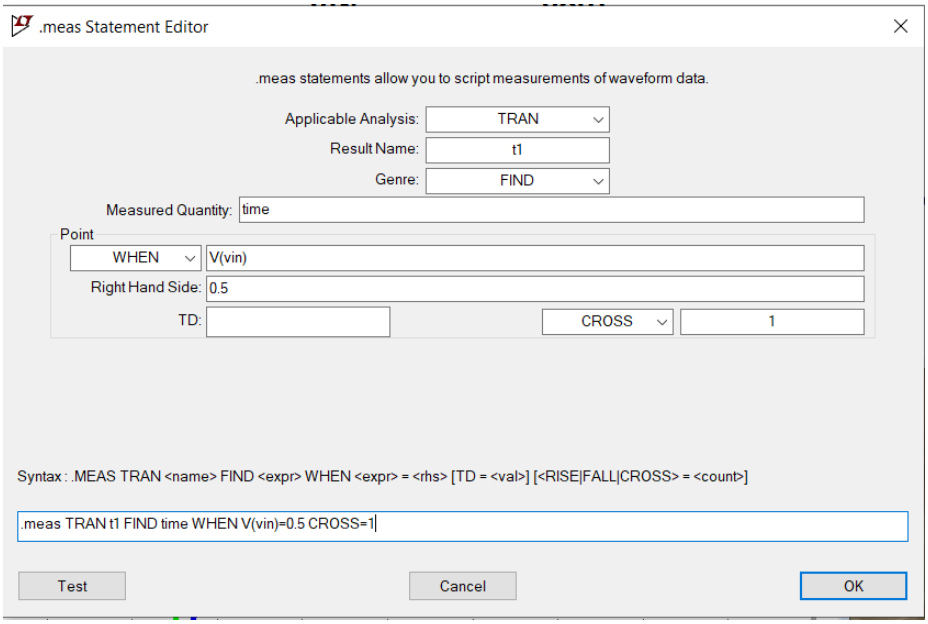


Figure 2.14 .meas Statement Editor

You should get the ".meas TRAN t1 FIND time WHEN V(vin) = 0.5 CROSS = 1". A direct translation of this command would be: "Get measurement in the transient analysis, and set the variable t1 to find the time WHEN v(vin) node is at 0.5 V while crossing the first instance this occurs". In simple English, this code finds the time (starting from t = 0), when we will first encounter (CROSS = 1) the instance that V(vin) = 0.5 V (this is the node of the input voltage), then store this value in the variable t1. Note that t1, is just a name.

7. **Repeat steps 1-4 to create another .meas spice directive with the entries listed below**.
   - Applicable Analysis: TRAN
   - Result Name: t2
   - Genre: FIND
   - Measured Quantity: time
   - Point: WHEN V(vin)
   - Right Hand Side: 0.5
   - CROSS: 1

   Just to make sure you got it right, **Result Name should change to t2**, and the **Point should change to WHEN V(vout)**. Instantiate this command into your schematic as well.

8. **Click the spice directive again but this time fill in the following entries listed below**. Figure 2.15 shows this setup. **When finished, click OK and place it on the schematic**.
   - Applicable Analysis: TRAN
   - Result Name: propagation delay
   - Genre: PARAM
   - Measured Quantity: t2-t1

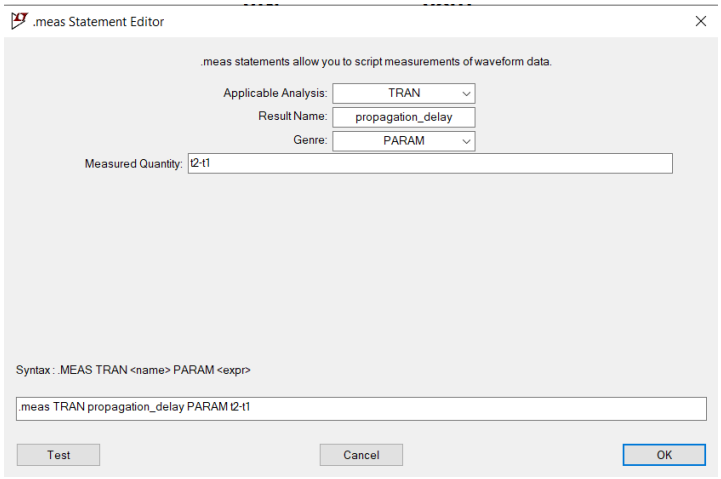


Figure 2.15 Measuring the propagation delay t2-t1

This .meas command gets the difference between t2 and t1. The direct translation would be "In a transient analysis, set the propagation_delay variable to hold the value of t2 – t1". Recall that t1 is the .meas variable extracting the time when VIN = 0.5V while t2 is the .meas variable extracting the time when VOUT = 0.5V. By now you should recognize that we are getting the propagation delay by getting the times when the input and output voltage reached 50% of their final value.

9. **Your circuit should now look like Figure 2.16**. Double check if the .meas spice directives are the same. If not, try to resolve these.

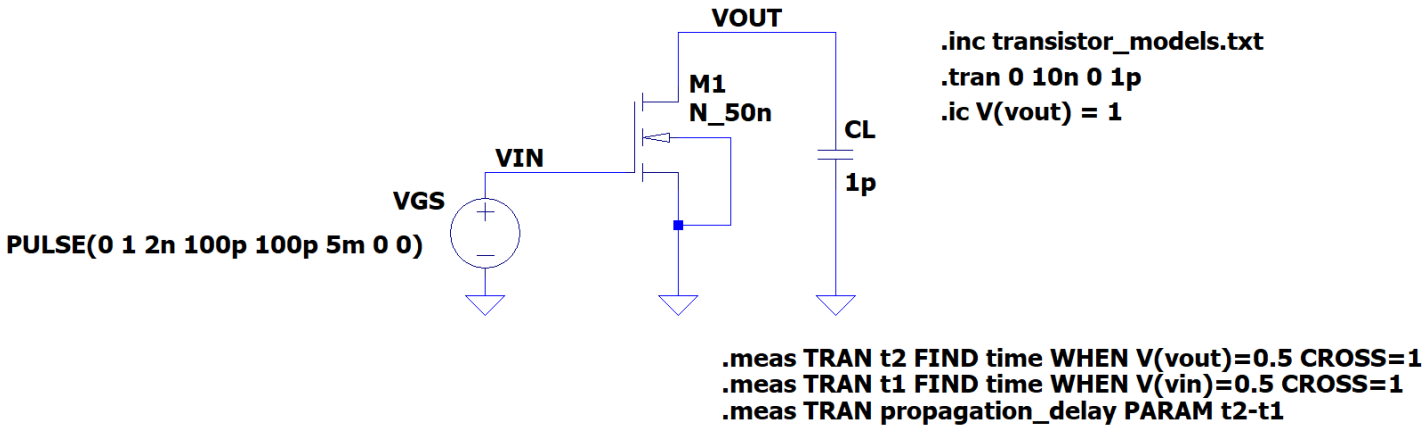


Figure 2.16 NMOS schematic with .meas spice directives

10. **Run your simulation**. Nothing new should appear so don't worry!
11. **Click on View > Spice Error Log**. Figure 2.17 should appear.
    - Take time to internalize and look at the log.
    - You should see the values for t2, t1, and propagation delay appear.
    - Observe that the propagation delay is 1.75 ns. This should be consistent with your calculations and the one you previously extracted using the cursors.
    - Great job on getting this far!

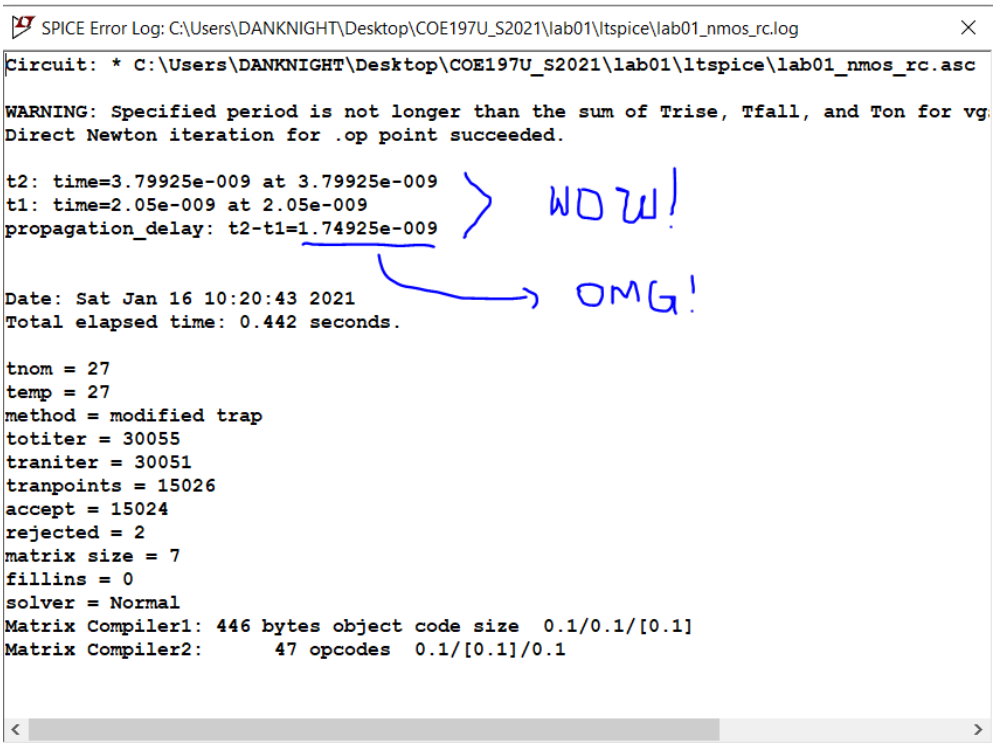


Figure 2.17 Spice Error Log showing t2, t1, and propagation_delay

DISCUSSION:
Recap time! The .meas directive is a useful tool of LTspice to help us automate the extraction of parameters from our circuit. Just to summarize the steps, we first extracted the propagation delay by setting the .meas command to get the time (t1) when VIN = 0.5 V, and also getting the time (t2) when VOUT = 0.5V. The propagation delay is the difference t2 – t1. They are all reported in the SPICE Error Log. This gives us a more accurate answer and is easier to setup than using the cursors. Good job in learning a new skill! You can explore more about this .meas command. Go google it or look directly into the LTspice documentation (Hit F1 and this should appear).

Question (Q2.3):
What is the effective R_{NMOS}? (Hint that leads to the answer: you know that the propagation delay is at 1.75 ns. You also know that the load capacitance is 1pF. You also know the simple equation to compute the propagation delay.)

Question (Q2.4):
Change the transistor width to 250 nm, then get the effective R_{NMOS}. Do it again for a width of 1000 nm. How would you describe the relation of the width size and the effective resistance R_{NMOS}?

DISCUSSION:
If done correctly, you should observe that the effective R_{NMOS} is somewhat linear. If we double the width, the effective R_{NMOS} halves. If we halve the width, R_{NMOS} doubles. There are many ways to interpret this but one of the easiest ways is that the current that is allowed to pass through the transistor is proportional to the width. This is evident from the current equation! So, if there is enough width for the current to pass through, then it also means the capacitor can easily discharge. Another interpretation is that if we treat the channel like a resistor, then it also follows that resistance $R = \rho L/A$. If we increase the width, we increase the area, and hence linearly reducing the effective resistance. You should grasp this concept because it can also be done to characterize the PMOS's effective resistance R_{PMOS}.

Figure 2.10 is quite incomplete because it only considers the equivalent switch resistance R_{NMOS} when the transistor is on. Figure 2.18 is a better version because it also considers the input and output capacitances. Although for now, we will mostly focus on the input capacitance as it plays a larger role when we deal with inverters later. The input of the transistor is a MOS capacitor (metal-oxide-semiconductor). From your previous semiconductor course, the input capacitance is mostly due to the oxide capacitance C_{OX}. Note that C_{OX} changes depending on the operating region of the MOS capacitor. The capacitance changes whether if we are in accumulation, depletion, or in strong inversion. Similar to the problem of R_{NMOS}, the effective capacitance also changes as a function of V_{GS}. We can find a similar equation similar to Equation 2.1, but that would be tedious. Instead, we can determine the effective $C_{IN,NMOS}$ capacitance analogous to how we extract the R_{NMOS}. Follow the succeeding steps to setup our "capacitor" extraction setup. Answer the questions that follow.

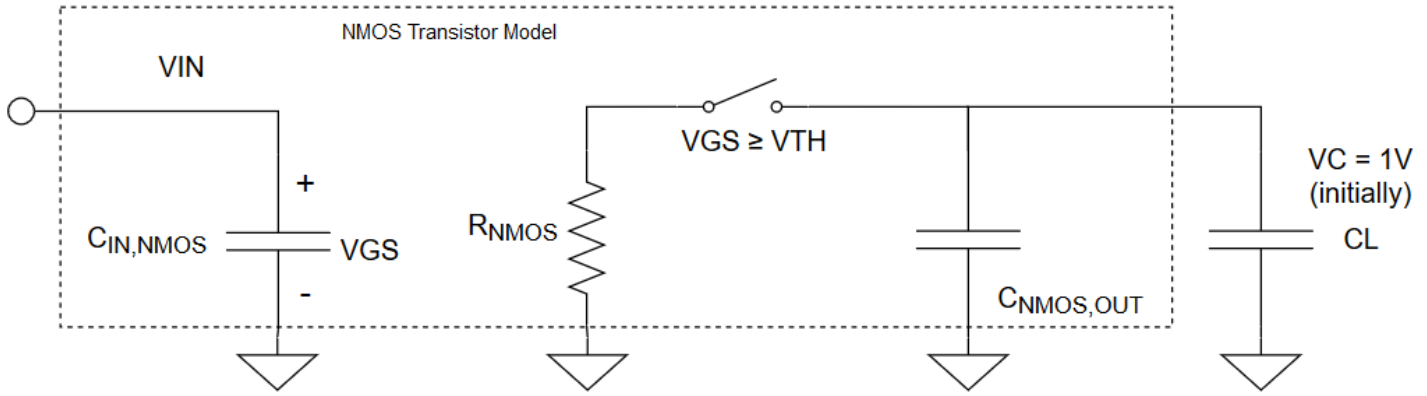
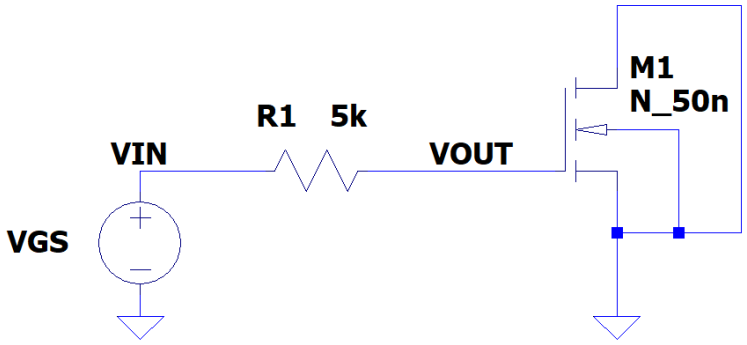


Figure 2.18 Better Model for NMOS Switch

1. **Recreate Figure 2.19 as a new schematic with the name "lab01_nmos_rc2"**. Make sure you implement the following:
   - Be sure to include the .inc transistor_models.txt
   - Use the N_50n model with l = 50n, and width = 500n
   - R1 = 5kΩ
   - Set V_{GS} to be a pulse that starts from 0V and ends at 1V. With 10ps delay, rise, and falling times. Set the Ton = 5ms and leave everything else to 0.
   - Don't forget to label the VIN and VOUT nodes.




```
PULSE(0 1 10p 10p 10p 5m 0 0)
.inc transistor_models.txt
.tran 0 100p 0 0.01p
```

```
.meas TRAN t1 FIND time WHEN V(vin)=0.5 CROSS=1
.meas TRAN t2 FIND time WHEN V(vout)=0.5 CROSS=1
.meas TRAN propagation_delay PARAM t2-t1
```

Figure 2.19 NMOS setup for extracting input capacitance

2. Similar to the R_{NMOS} extracting steps, extract **the propagation delay**, set the .meas spice directives to extract:
   - The time when VIN = 0.5V. Set this to t1.
   - The time when VOUT = 0.5V. Set this to t2.
   - Propagation delay t2-t1
   - The bottom of Figure 2.19 should guide you.
3. **Set a transient simulation to span 80ps in 0.01ps time steps**.
4. **Run your simulation**.
5. **Plot VIN and VOUT both super imposed in one plot, then get the propagation delay**. You should get Figure 2.20 and a propagation delay of 4.62 ps.
6. Great! We're down half-way of our training exercise. Let's try to interpret the results that we have.

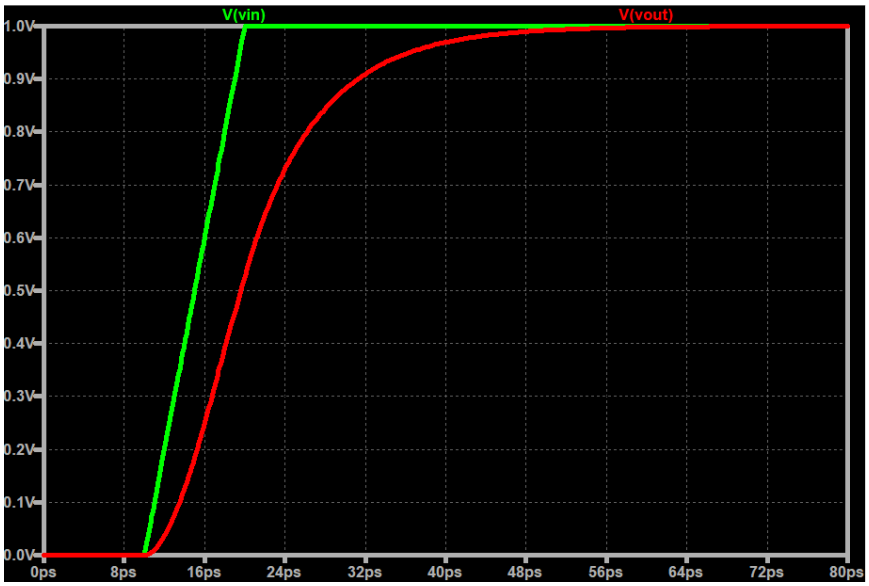


Figure 2.20 Simulation for NMOS switch extracting input capacitance

DISCUSSION:
Looking at our schematic setup in Figure 2.19, it is equivalent to the model shown in Figure 2.21. Again, we emphasize that Figure 2.21 is a "description" or a "model" of Figure 2.19 but it does not perfectly describe what the actual input capacitance is. We are only trying to determine the "effective" input capacitance $C_{IN,NMOS}$ from this model. With this, we can use the same methodology in extracting the effective R_{NMOS} with $C_{IN,NMOS}$ by simply looking at the propagation delay between VIN and VOUT set in Figure 2.16. We hope you get the idea on the method of extracting the effective R_{NMOS} switching resistance and input capacitance of the transistor.

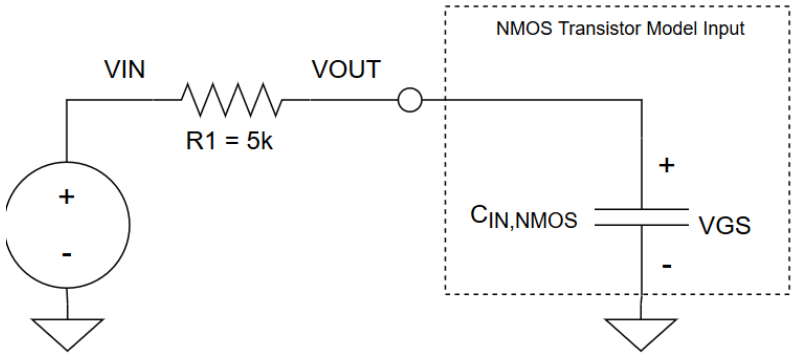


Figure 2.21 Extracting input capacitance

Question (Q2.5):
What is the effective $C_{IN,NMOS}$? (Hint that leads to the answer: you know that the propagation delay is at 4.63 ps. You also know that the resistance is 5kΩ. You also know the simple equation to compute the propagation delay.)

Question (Q2.6):
Change the transistor width to 250 nm, then get the effective $C_{IN,NMOS}$. Do it again for a width of 1000 nm. How would you describe the relation of the width size and the effective capacitance $C_{IN,NMOS}$?

WARNING:
Figure 2.19 not only extracts the sum of C_{GS}, C_{GD}, and C_{OX} capacitances but also all other capacitances that may change due to different operating conditions. This may only be a part of the input capacitances which we will extract later when we get back to inverters. Additionally, there exists parasitic effects like the *Miller Effect* where the C_{GD} gets amplified at the input by the gain of the inverter. We will talk about this in when we get to analog design. What's important here is that the schematic in Figure 2.19 aims to get the effective input capacitance with all other effects considered.

The important lesson in this section is the process of extracting the effective switching resistance and the effective input capacitance. Just the process! To summarize this method, we simply model the input and output as two separate circuits. The output is a simple resistor, while the input is a simple capacitor as shown in Figure 2.15. So, if we want to determine the output resistance R_{NMOS}, we simply model the output of the transistor as a switch with R_{NMOS}. We place a capacitor at the output that is large enough for us to extract the propagation delay. We also know the equations to get the propagation delays (as well as rise and fall times) and then solving for R_{NMOS} is trivial. The same goes for the input of the transistor. It's mostly a capacitive component so we treat it as if it were some kind of capacitor as shown in Figure 2.19. This is indicated in the left side of Figure 2.18. Then we do the same method by attaching a series resistor that is high enough for us to extract the propagation delays. The output capacitance also plays an important role, but in most cases, it is neglected.

D. Inverter DC Characteristics

Let's now study the DC characteristics of the inverter. Figure 2.22 shows the inverter schematic that we will use for our analysis. Figure 2.23 shows the spice schematic that you need to create. By now, you should be comfortable on setting up circuit schematics base on the figure only. Follow the succeeding steps to setup the DC analysis for our simple inverter.

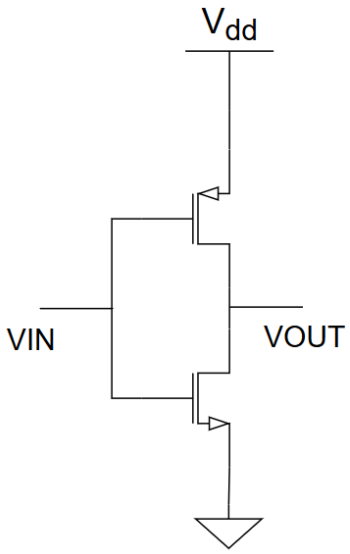


Figure 2.12 The inverter

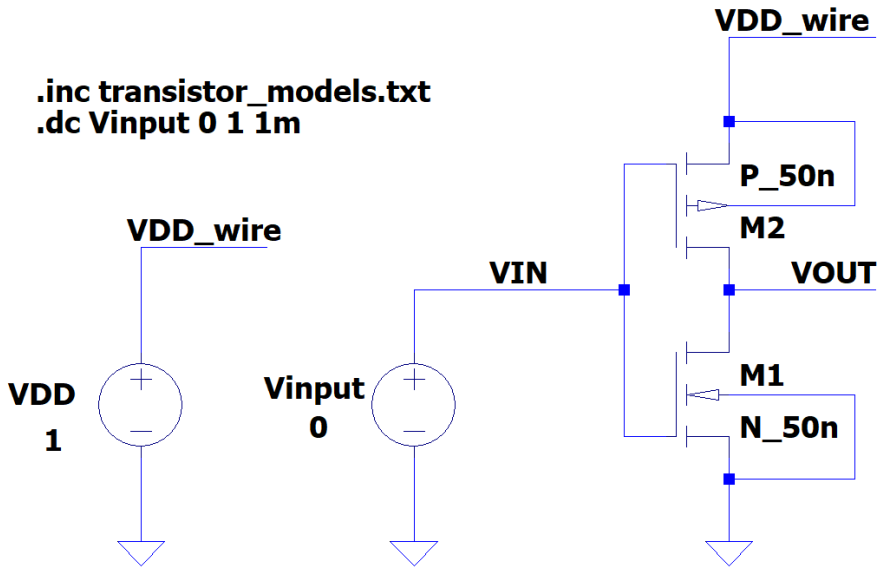


Figure 2.23: Inverter schematic for DC simulation

1. **Recreate Figure 2.20 as a new schematic. Save this as "lab01_inverter_dc"**. It should be clear to you that Figure 2.23 is the same as Figure 2.22. Below is the list of component properties that must be configured:
   - **Don't forget to include the transistor_models.txt**
   - Use N_50n as the model for the NMOS transistor width length = 50n, and width = 500n.
   - Use P_50n as the model for the PMOS transistor width length = 50n, and width = 500n.
   - The PMOS transistor was rotated twice then flipped once such that the source is closer to VDD_wire.
   - Create 2 DC voltage sources with VDD and Vinput as the source names shown in Figure 2.20.
   - **IMPORTANT**! Observe how the + side of the VDD voltage source is just a wire that is cut horizontally. This is fine because we just need to label it as VDD_wire. The source of the PMOS is also connected to a wire labeled as VDD_wire. **Wires that are labeled with the same name are CONNECTED to each other**. Hence, the output of the VDD_source is connected to the source of the PMOS in this setup.
   - Be sure to connect the bulk of the PMOS to VDD_wire while the bulk of the NMOS to ground.
   - Set VDD = 1V, while Vinput = 0V.
   - Don't forget to create a wire in between the PMOS and NMOS. Extend this wire and label it as VOUT.
   - Ignore the .meas spice directives for now.
2. **Set the simulation to do a DC sweep where Vinput starts at 0V, stops at 1V, in 1mV increments**.
3. **Run the simulation and plot the VOUT only**. You should get the fantastic looking inverted S-shape shown in Figure 2.24.

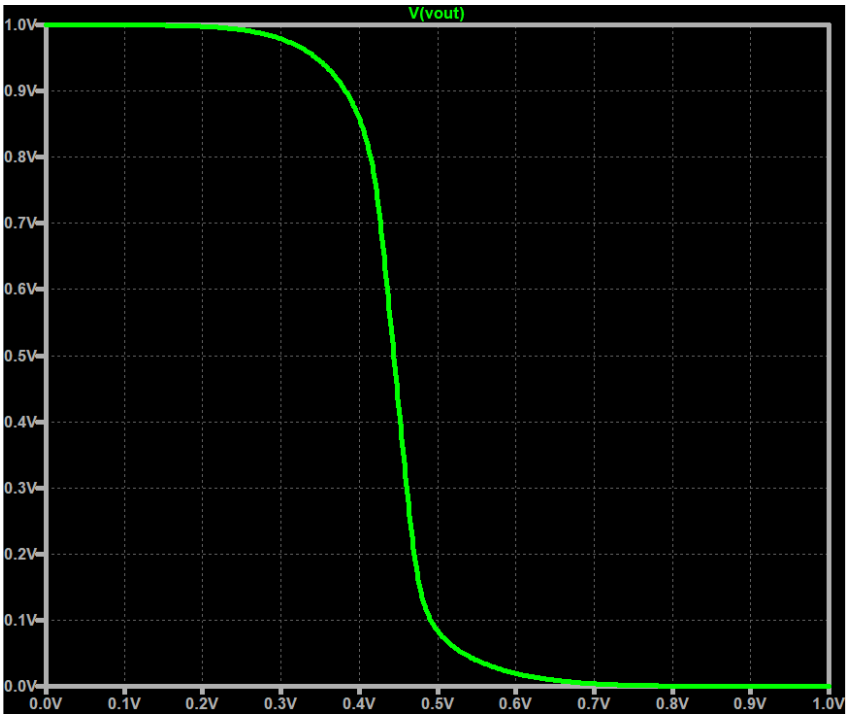


Figure 2.24 Inverter Voltage Transfer Curve

DISCUSSION:

Let's take time and appreciate this S-curve. Figure 2.21 shows the voltage transfer curve for our inverter. It's called a voltage transfer curve because it literally shows how our input changes (transforms) into an equivalent output voltage. For every input, there is an equivalent output voltage. You should know that the x-axis is the input voltage VIN while the y-axis is the output voltage VOUT. For example, when VIN = 0 V then VOUT = 1 V. When VIN = 1 V, VOUT = 0 V. It does what an inverter is supposed to do! That is to convert an input of 1 to 0 and vice versa. What's interesting though is that somewhere in between the range of $0.3V \leq VIN \leq 0.7$ V, the output is not exactly a 1 or a 0. Remember, this is a DC analysis, so say for example, we set VIN = 0.45 V for a very long time (because it's DC) then the resulting output is roughly VOUT = 0.80 V. There is this region that does not produce an exact 1V at the output. From your lecture slides, this is the noisy region where the output cannot determine if the input is not even close to a "good" 1 or 0.

Let's add some interpretation to what we observed in the previous paragraph. As mentioned in the previous sections, the transistor is treated as a switch. Or better yet, a transistor is like a valve where you can control the current that goes through the transistor (from drain to source and vice versa). For example, say VIN = 0V, then the NMOS transistor is off because $VIN = V_{GS,NMOS} = 0V$. However $V_{SG,PMOS} = 1V$ in this scenario making the PMOS transistor on. If that's the case then the VOUT node is shorted to VDD_wire that's why VOUT = 1V. On the other extreme, when $VIN = V_{GS,NMOS} = 1V$, then the NMOS is on connecting VOUT to ground instead. When this occurs $V_{SG,PMOS} = 0V$ making the PMOS transistor off. Clearly, the inverter selects which switch turns on! In the middle of the "noisy" region, both transistors are on at the same time. Such that the PMOS wants to pull VOUT to VDD and the NMOS wants to pull VOUT to GND. There's a "tug-of-war" occurring. Moreover, when both PMOS and NMOS channel exists, there is a straight connection from VDD to GND. This is known as a short-circuit. You can probe the current from the drain of the NMOS and get Figure 2.25.

DISCUSSION (continued):

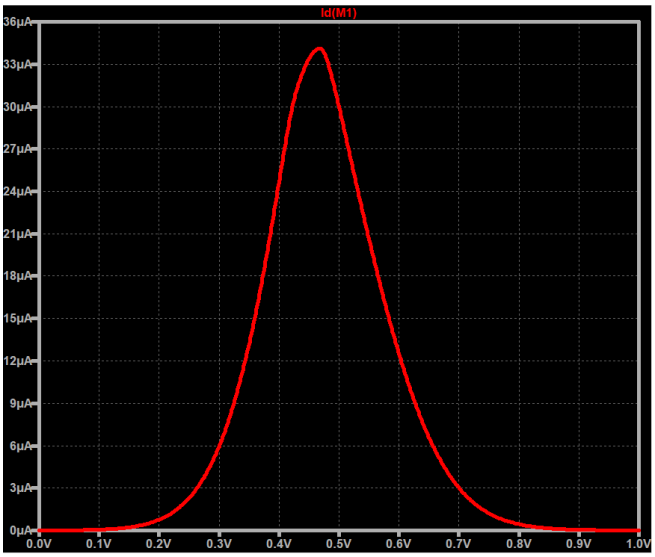


Figure 2.25 Drain current in the NMOS of the inverter

During the tug-of-war, 1 mis-step by either an NMOS or the PMOS will immediately try to shut-off the other. For example, if we set VIN to be 10mV less such that VIN = 0.478V, then the output voltage would be VOUT = 0.628V. That's a 128 mV output change for only a 10mV input step. If we increase VIN by 10mV such that VIN = 0.498 V, we get VOUT = 0.382 V. That's a 118 mV output change for only a 10mV input change. In this "unstable" region, one small step results in a big output change indicating that one of the transistors is dominating the connection! Take some time to understand this phenomenon. It should be evident from your previous class that this change in output for a change in input is also known as the gain of the circuit. Essentially $Gain = \frac{\Delta V_{OUT}}{\Delta V_{IN}}$!

The voltage transfer curve and the concept of gain will be discussed again later when we get to analog design. It's common to see this because it gives us a lot of information. Not only does it describe how our input transforms into a voltage at the output, we can also determine "noisy" regions, it also implies what occurs in our circuits, and it may also provide us information about gain. For now, what's important is that with just this single figure, there is a lot of things we can learn from it. Take time to internalize this and let it sink in ☺. The concept of noise margins was discussed in the lecture slides. We'll live it up to you how as a reading exercise how to get the noise margins.

Question (Q2.7):
There are two figures given below. Figure 2.26 is just Figure 2.24 but labelled with points. The x-axis is the input voltage to an inverter and the y-axis is the output voltage. Figure 2.27 is just Figure 2.4 but with more V_{GS} points. In Figure 2.27, the x-axis is the V_{DS} of the transistor, while the y-axis is the drain to source current. Figure 2.23 only shows the NMOS. Your goal is to match the corresponding point number in Figure 2.27 to the letters in Figure 2.26 when they occur. For example, point A occurs when VGS = 0V and VDS = 1V this is equivalent to point 5 in Figure 2.27. Refer to your answer sheet. No need to re-simulate, you only need Figure 2.26 and Figure 2.27.

A = 5 (Because point A is when VGS = 0, and the VDS = 1V for the NMOS transistor)

Question (Q2.7 CONTINUED):

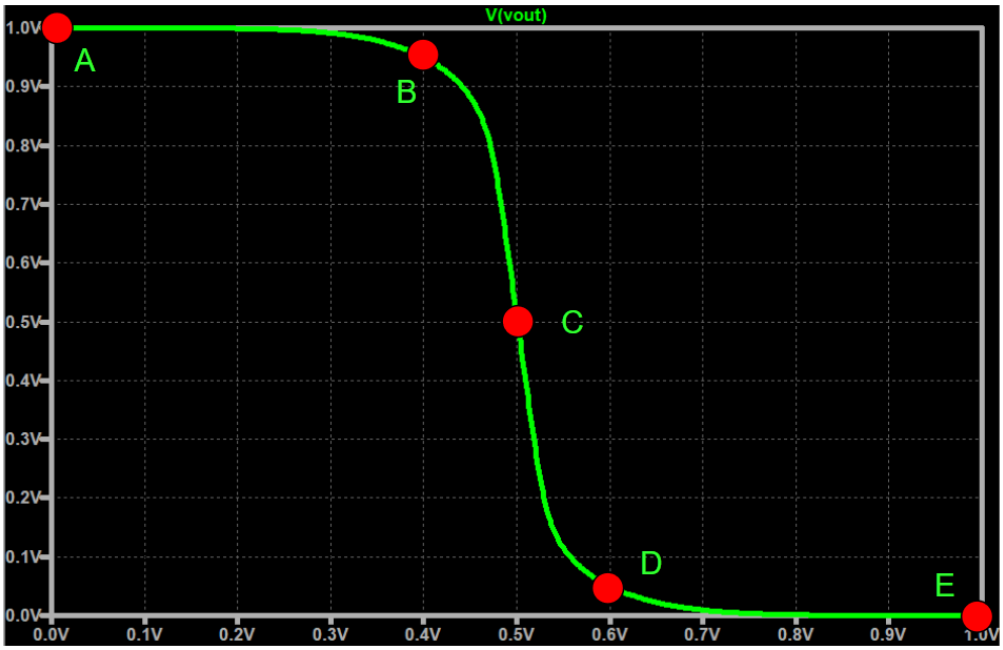


Figure 2.26 Inverter VTC (voltage-transfer-curve) with points

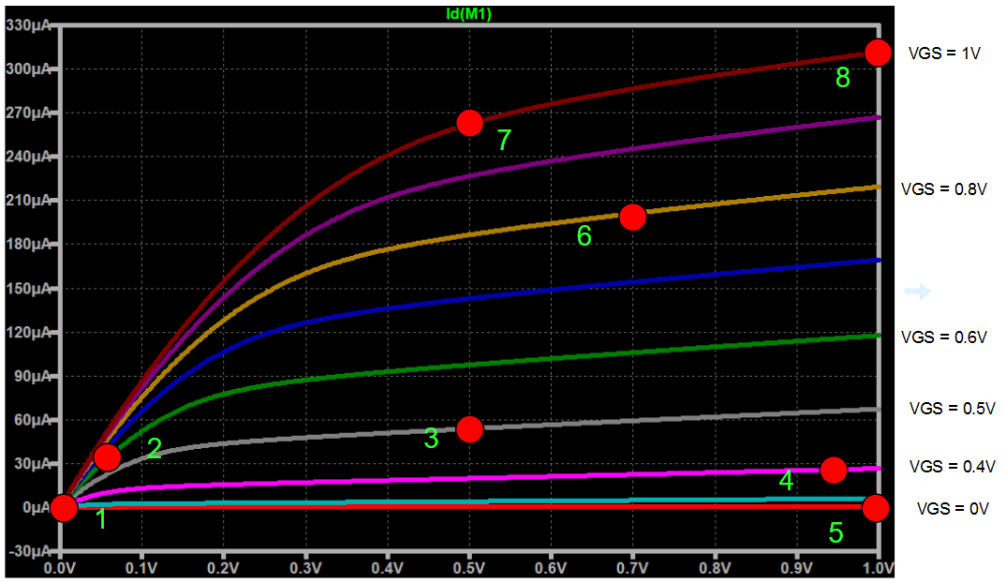


Figure 2.27 IDS vs VDS curves for the NMOS transistor

E. Inverter Transient Characteristics

For the inverter transient characteristics, we only need to get two important parameters, the propagation delay from a high output to a low output transition (t_{PHL}) and the propagation delay form a low input to a high input transition (t_{PLH}). Follow the succeeding steps to setup our test circuit in order to characterize the timing parameters.

1. **Create the schematic shown in Figure 2.28. Save this schematic as "lab01_invert_tran".** You may just copy-paste Figure 2.23, or you may reuse the same circuit. Just for clarity, below is the list of component properties that must be set:
   - **Don't forget to include the transistor_models.txt**
   - Use N_50n as the model for the NMOS transistor width length = 50n, and width = 500n.
   - Use P_50n as the model for the PMOS transistor width length = 50n, and width = 500n.
   - **IMPORTANT!** Observe how the + side of the VDD voltage source is just a wire that is cut horizontally. This is fine! We just need to label it ass VDD_wire. The source of the PMOS is also connected to a cut wire but labeled as VDD_wire. Wires that are labeled with the same name are CONNECTED to each other. The output of the VDD_source is connected to the source of the PMOS in this setup.
   - Be sure to connect the bulk of the PMOS VDD_wire while the bulk of the NMOS to ground.
   - Set VDD = 1V, while Vinput = 0V.
   - Don't forget to create a wire in between the PMOS and NMOS. Extend this a bit and label that as VOUT.
   - **Set the load capacitance CLOAD = 1pF**.

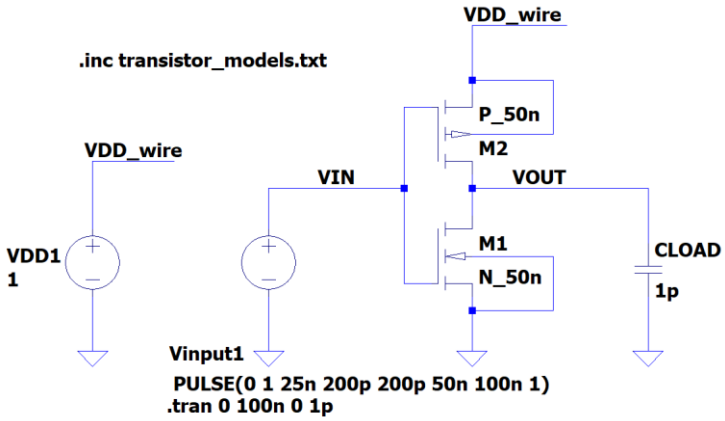


Figure 2.28 Schematic setup for simulating transient characteristics of a simple inverter

2. **Set Vinput to produce a pulse source with the following parameters**:
   - Vinitial = 0V
   - Von = 1V
   - Tdelay = 25n
   - Trise = Tfall = 200p
   - Ton = 50n
   - Period = 100n
   - Ncycles = 1
3. **Set a transient analysis to start at t= 0, and end at t = 100 ns, in 1 ps time steps**.
4. **Run the simulation**
5. **Plot VIN and VOUT in one plotting plane**. You should get Figure 2.29.

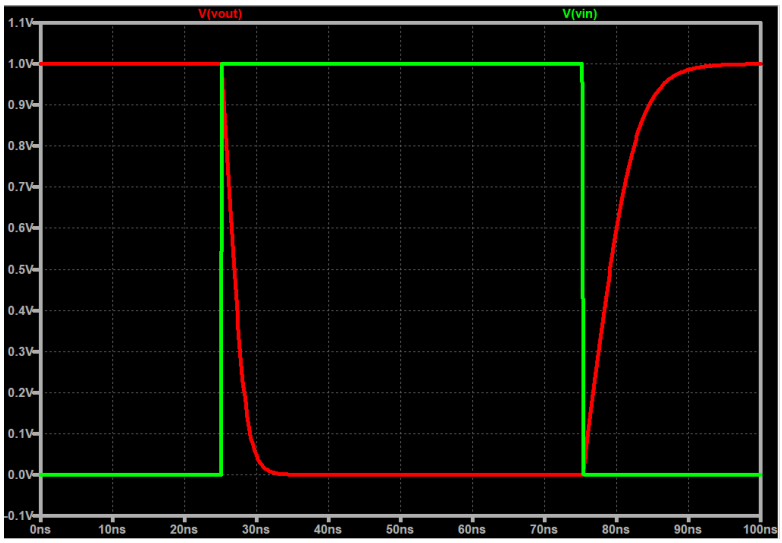


Figure 2.29 Transient simulation for our inverter setup

Just a quick sanity check, again when the input is 0, the output is 1. But when the input moves to 1, the output is slowly pulled down to 0. When the output moves from high-to-low (t_{PHL}), it appears to be faster than when the output moves from low-to-hi (t_{PLH}). Spoiler alert, this has something to do with the mobility of the NMOS and PMOS current. It's more interesting if we can extract the t_{PHL} and t_{PLH} propagation delays. The next steps should help you set this up.

6. Let's first setup the .meas spice directives to determine the t_{PHL} and t_{PLH} delays. Figure 2.30 shows the spice directives for measuring t_{PHL}, while Figure 2.31 shows the spice directives for t_{PLH}. Go ahead and set these up. You may simply copy what is written directly into the spice directive editor. Otherwise, follow the details of each setup.

t_{PHL} setup:

- During transient (TRAN) measure the time it takes WHEN the input voltage rises up to 0.5V (V(vin) = 0.5). This occurs during the 1st edge (CROSS=1). Save the result into the variable named t1_vin_lh.
- During transient (TRAN) measure the time it takes WHEN the output voltage rises up to 0.5V (V(vout) = 0.5). This occurs during the 1st edge (CROSS=1). Save the result into the variable named t2_vout_hl.
- Compute tphl = t2_vout_hl – t1_vin_lh.

```
.meas TRAN t1_vin_lh FIND time WHEN V(vin)=0.5 CROSS=1
.meas TRAN t2_vout_hl FIND time WHEN V(vout)=0.5 CROSS=1
.meas TRAN tphl PARAM t2_vout_hl-t1_vin_lh
```

Figure 2.30 .MEAS spice directives for measuring t_{PHL}

t_{PLH} setup:

- During transient (TRAN) measure the time it takes WHEN the input voltage rises up to 0.5V (V(vin) = 0.5). This occurs during the 2nd edge (CROSS=2). Save the result into the variable named t1_vin_hl.
- During transient (TRAN) measure the time it takes WHEN the output voltage rises up to 0.5V (V(vout) = 0.5). This occurs during the 2nd edge (CROSS=2). Save the result into the variable named t2_vout_lh.
- Compute tplh = t2_vout_lh – t1_vin_hl.

```
.meas TRAN t1_vin_hl FIND time WHEN V(vin)=0.5 CROSS=2
.meas TRAN t2_vout_lh FIND time WHEN V(vout)=0.5 CROSS=2
.meas TRAN tplh PARAM t2_vout_lh-t1_vin_hl
```

Figure 2.31 .MEAS spice directives for measuring t_{PLH}

7. Go ahead and run the simulation again and be sure to record t_{PHL} and t_{PLH}. You should get t_{PHL} = 1.77ns, and t_{PLH} = 3.80 ns.
8. Congratulations! You did a good job getting this far 😊

DISCUSSION:

Let's try to appreciate the results so far. First, as mentioned earlier, it is interesting to note why t_{PLH} is roughly 2x higher than t_{PHL}. If we go back to the current characterization of our PMOS and NMOS transistors, Figure 2.3 and Figure 2.5 shows that the NMOS current is almost twice (or a little bit higher) than that of the PMOS. Also take note that these transistors were originally sized to have the same widths and lengths! If we divide the NMOS saturation current equation and the PMOS saturation current equation, and assuming VTHN = VTHP (don't worry they're relatively close), we get $\frac{\mu_n}{\mu_p}$. The mobility of electrons is 2x higher than holes! This is evident from the I_{DS} vs V_{DS} currents and the t_{PHL} and t_{PLH} results. From here, it should be inferred that the PMOS transistor has 2x more resistance compared to the NMOS transistor. What an interesting property! Keep this in mind because even when PMOS is slower than NMOS, we usually desire a PMOS to give us high resistance. Later, you will learn that this leads to better gain.

Question (Q2.8):
Determine the effective R_{PMOS} and R_{NMOS} from the t_{PHL} and t_{PLH} results. (Hint: look at our circuit, we used a 1pF capacitor)

Part III: Exercise

Let's go a bit further with these exercises. We will be re-using most of the circuits we made earlier. Answer and solve the following problems.

A. Short-Channel vs. Long-Channel

We'll be using Figure 2.1 for this exercise. It's reposted below so you don't have to scroll too far up.

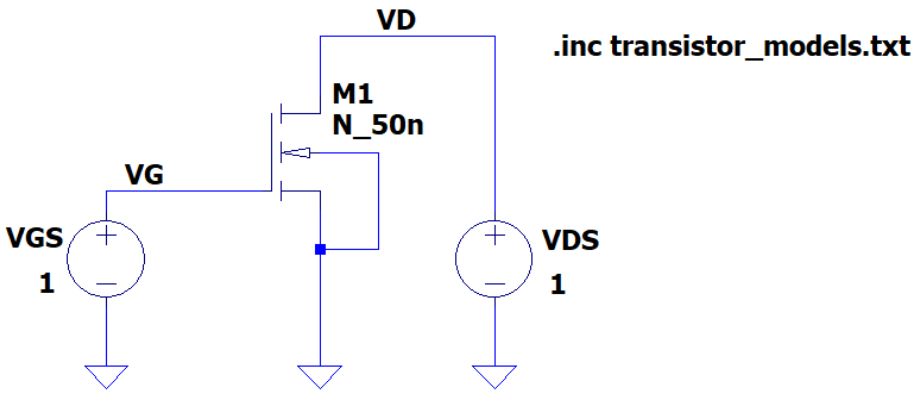


Figure 2.1 DC Analysis Setup for NMOS Transistor

Do the following then answer the succeeding questions that follow:

- Re-simulate a DC sweep analysis with VDS changing from 0 to 1V in 1mV increments. Let V_{GS} = 1V as well. Plot the I_{DS} current as you will need it for comparison later. You should get the same as Figure 2.3 (also reposted below) which you should've already simulated earlier.

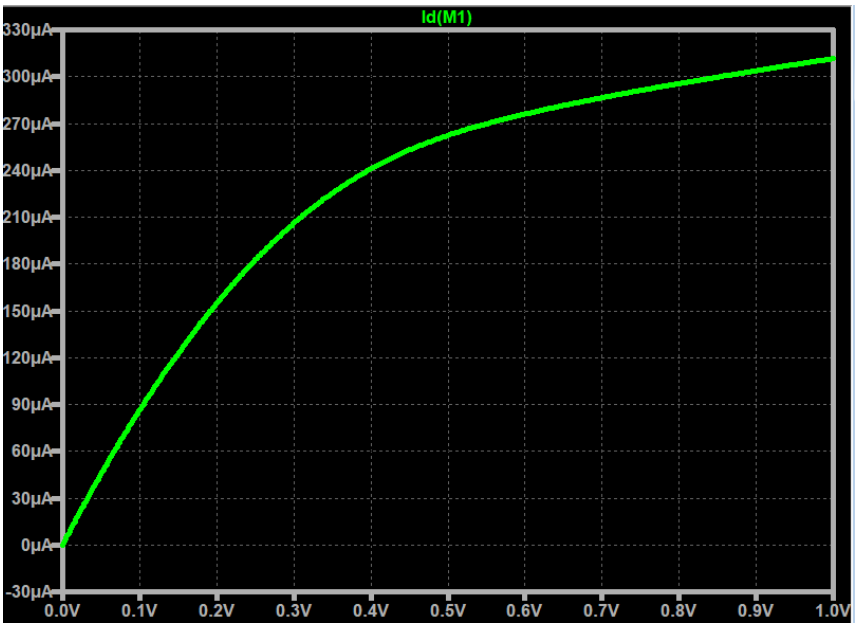


Figure 2.3 Short-channel NMOS I_{DS} vs. V_{DS} with V_{GS} = 1V

- Re-simulate the same setup but this time for a long-channel MOSFET. It is recommended to create a new schematic just so you can compare the succeeding plots simultaneously. Be sure to change the transistor model to use N_1u with length = 1u and width = 10u. Keep all other settings the same. That means your V_{DS} sweep should be from 0 to 1 in 1mV increments and V_{GS} = 1V. Save the plot. Then answer the following questions below.

Questions:

1. Originally, the short-channel NMOS has a $V_{THN,short-channel}$ of roughly 0.3V (the real $V_{THN,short-channel}$ = 0.28 V) and it was difficult to extract due to the linear vs. saturation separation not being too clear. As for the long-channel NMOS, what is $V_{THN,long-channel}$?
2. Compare the current values for both. Why do you think the long-channel has less current?
3. Compare the slope of the saturation region of the short-channel and long-channel currents. What's so different between the two? Can you name something that causes this? (You may need to research a bit).

This time let's make one more set of comparison. For both setups, change the VDS = 3V, VGS = 3V, and the VDS sweep to move from 0V to 3V in 1mV increments. Again, simulate the IDS vs VDS curves for both then answer the questions below.

4. With the new setup, compare the currents between the short-channel and long-channel MOSFETS. Try to elaborate why this happens and compare it to your results and answers from the original setup (i.e. VGS, VDS = 1V).

B. CMOS Tug-of-War

We will reuse Figure 2.23 for this part. It's re-pasted down below for convenience. This time we added a chunk of .meas SPICE directives which measures the middle point of the VTC transition. We will learn a new technique for parametrizing width sizes in a single DC sweep.

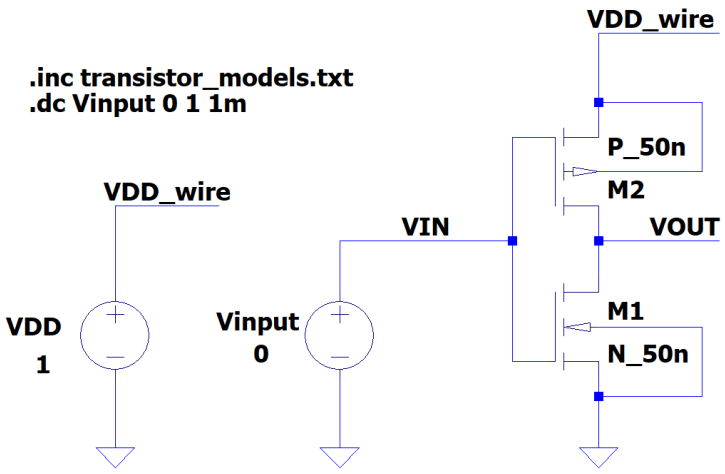


Figure 2.19: Inverter schematic for DC simulation

Originally, the discussion of the DC characteristics of the transistor showed that the PMOS width = NMOS width where both are 500 nm in size. With these widths and lengths set for our transistors, at VIN = 0.5V the output is not perfectly at 0.5V. This affects the effective noise margin and the unequal t_{PHL} and t_{PLH} delays. In some scenarios it is desirable to resize the widths (sometimes lengths too) such that the VOUT = 0.5V when VIN = 0.5V. We want this to occur so that our PMOS and NMOS circuits are symmetric to each other and any transition in the PMOS and NMOS switch will provide the same results, or at least get close enough to have the same results. Do the following:

1. Using the lab01_inverter_dc schematic, change the PMOS width to {500n*k}. See Figure 3.1 for clarity.

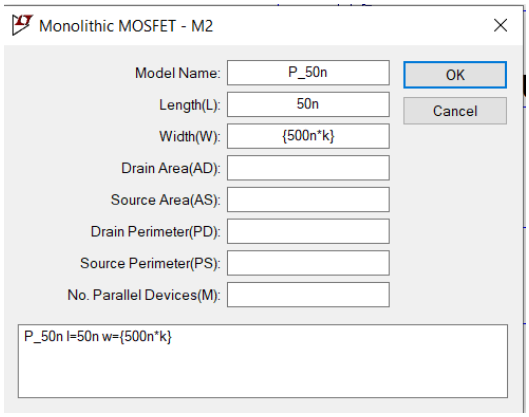


Figure 3.1 Changes for the PMOS width

2. Now add a .step spice directive and paste it on the schematic. Right-click on it to bring up the .step statement Editor. It should look like Figure 3.2. Fill in the following parameters in the editor.

    a. Name of parameter to sweep: k
    b. Nature of sweep: Linear
    c. Start value: 1
    d. Stop value: 2.6
    e. Increment: 0.2

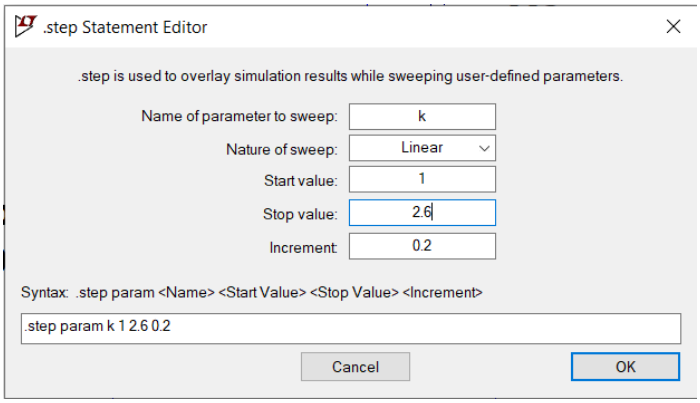


Figure 3.2 .step Statement Editor

The .step statement here extracts the variable k which we set in the width of the PMOS transistor. Then our simulation makes 1 simulation when k = 1, such that the PMOS width = 500 nm. Then it does another simulation with k = 1.2 (because the increment for the next step is 0.2), such that the 2nd DC sweep has a PMOS width of 600 nm. Them it continues until we reach k = 2.6 where the last simulation will be the DC sweep when the PMOS width = 1,300 nm. This simulation provides a parametric analysis that repeats our DC sweep but with varying widths!

3. Run a simulation plot VOUT. You should see this nice looking VTC curves just like the one in Figure 3.3. Feel free to zoom in the area near the 0.5 input region.

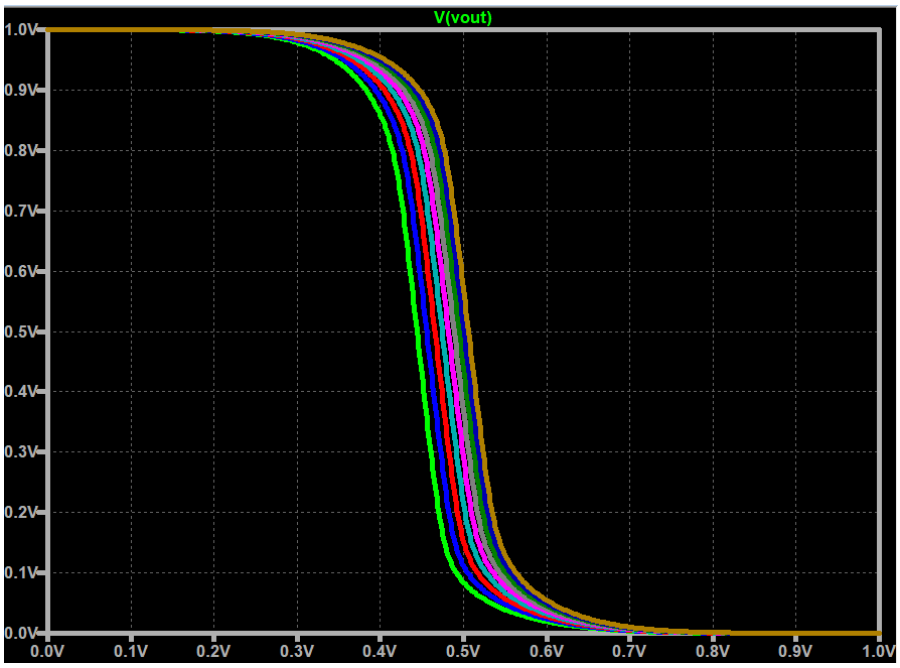


Figure 3.3 Parametric analysis of the VTC curve for varying PMOS widths

Questions:

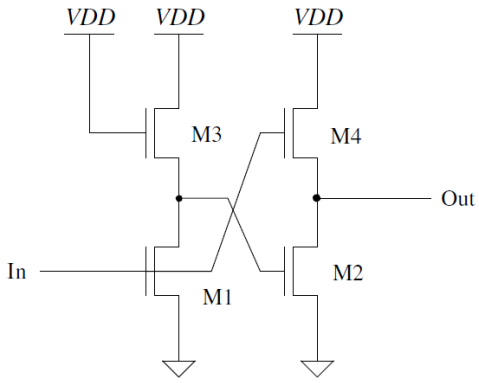
1. What can you observe with the increasing widths of the PMOS? How does the VTC move with increasing PMOS widths?
2. Can you interpret what happens when the widths of the PMOS increase? Answer in terms of, what happens to the PMOS and NMOS currents.
3. What do you think is the best multiplier for the PMOS width such that VOUT is close to 0.5V when Vin = 0.5V? With you answer, what's the best PMOS width to NMOS width ratio?
4. Using the width, you selected in #4, simulate Figure 2.28. Extract the new t_{PHL} and t_{PLH} and compare it to the original results. Did it improve? Make sure to record your t_{PHL} and t_{PLH} readings.

Part IV: Supplementary Notes

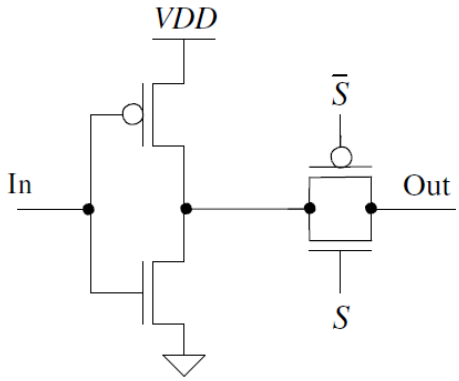
Other inverter configurations

Just for fun, the circuits below are different kinds of inverters. Can you identify / describe what these do? Images were taken from “CMOS Circuit Design, Layout, and Simulation” by Jacob Baker.

1. Inverter circuit 1



2. Inverter circuit 2



3. Inverter circuit 3