Lab 2: Design and DC Simulation of an Inverter

For information on how to setup your account and Cadence environment, and on how to start running Cadence with GPDK045 cadence design kit, please refer the previous lab material, Cadence Lab-1.

For a full custom cadence design (as opposed to a semi-custom or coded/synthesized design), the design process starts by creating a schematic design. The schematic is then simulated to verify operation and optimize performance. As the next step, a layout design of the schematic is generated and checked for physical Design Rule Check (DRC). A netlist is then extracted from the layout design, and a Layout vs. Schematic (LVS) comparison is run to ensure the layout matches the schematic. Finally with the extracted netlist from the layout, the layout design is simulated to observe the effect of *parasitics*. This is called *post layout simulation*, and it reflects more close performance of the design when it comes in a fabricated chip.

As the first step for a full custom cadence design, this lab will deal with a schematic design for the simplest CMOS logic gate, inverter, and perform DC simulations on the schematic to verify its functionality and Voltage Transfer Curve (VTC).

1) **Launch Cadence**:
    a. Connect to the cadence server (cadence03.rowan.edu), using your Rowan network ID and password (refer previous lab material for more info.).
    b. After logging in, set the Cadence environment by sourcing cds.setup file:
    >> source cds.setup
    c. **Move to your Cadence work directory** and then **start Cadence** there:
    >> cd cadence
    >> virtuoso &
    d. You will get two windows, **CIW** and **Library Manager**, and the Library Manager should list three GPDK045 libraries and the one you created, e.g., **Lab**, during the last lab. If the Library manager doesn't open it can be opened from the CIW window.
2) **Draw a Schematic** for a CMOS inverter:
    a. From the **Library Manager**, create a new **Schematic** view, with a name **invx**, *under your work library*. Choose **Schematics XL** for the *Application/Open with*, and **check** for the *checkbox.*

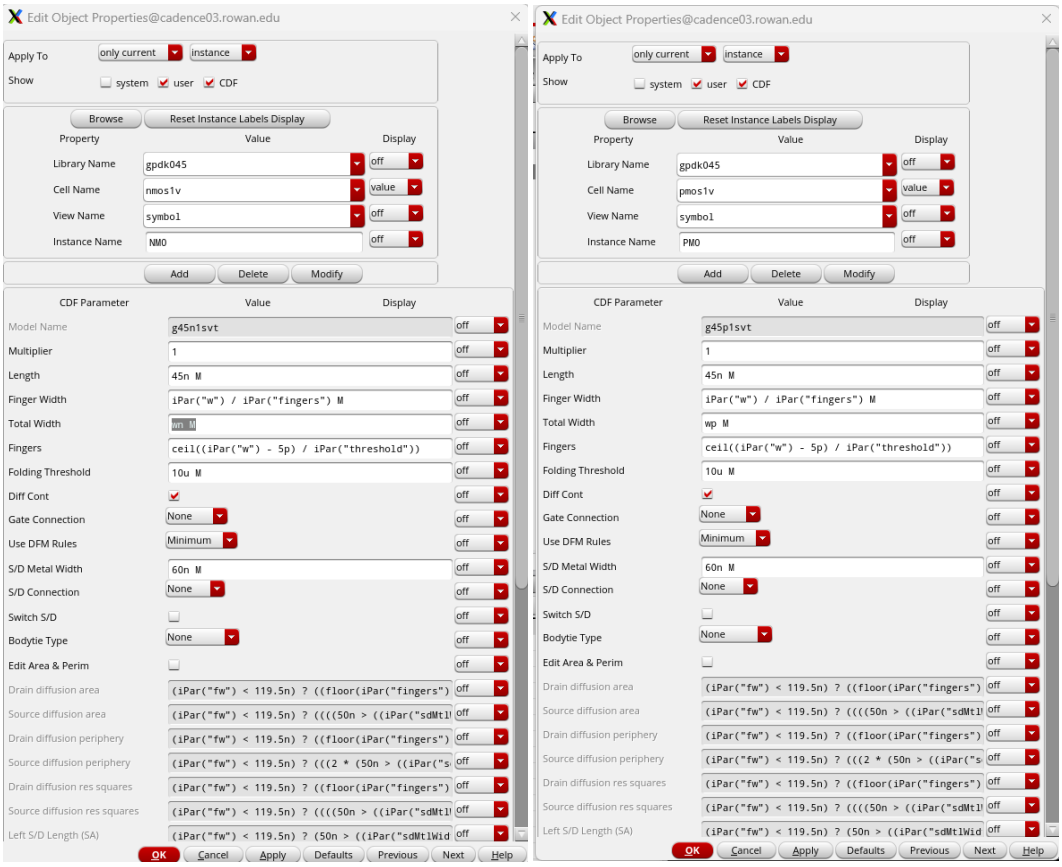
b. From the menu bar, choose *Create* -> *Instance*, or type its hotkey ***i***. Then navigate for the components you want to place, specify its property, and then place it onto the *Schematic Editor*.

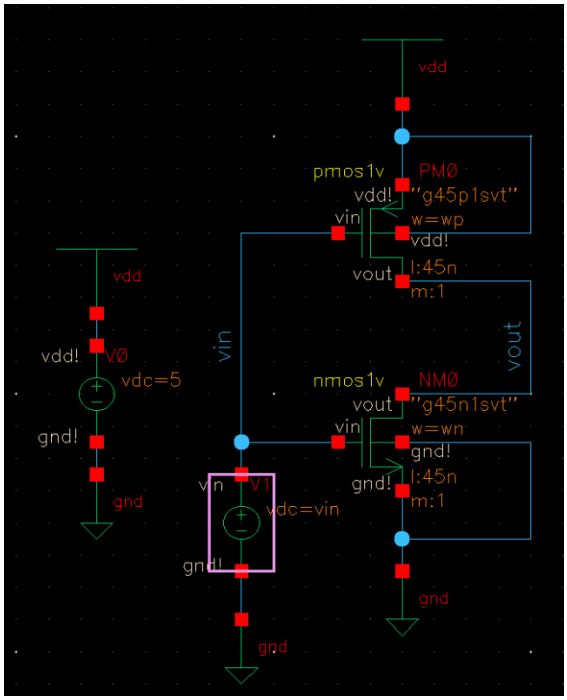
Use the components **nmos1v** and **pmos1v** from the library **GPDK045** for nMOS and pMOS transistors, and choose other ideal components from the library **analogLib**.

Change your gate source voltage to vin.

Below are example properties chosen for transistors with variables ***wn*** & ***wp*** for widths of nMOS & pMOS, respectively. You do not need to change every parameter under that instance, once you change the total width the rest of the values will adjust.

(Note that the minimum transistor width for the gpdk 45nm CMOS is 120nm)





c. Use hotkey "L" (not case sensitive) to change the net names to "vin" and "vout" to make the calculations later in the lab easier. (note: These are case sensitive)

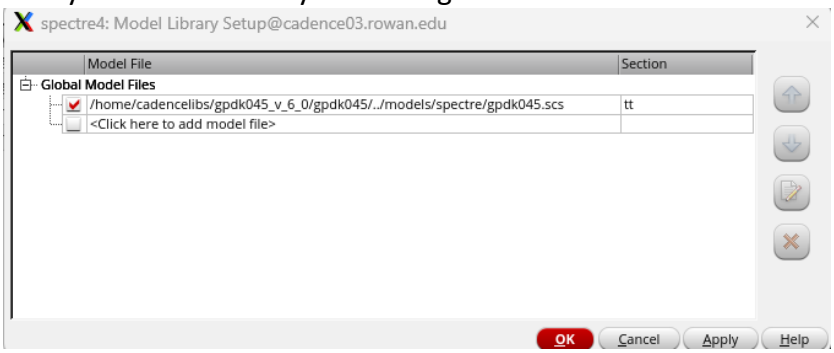
d. Don't forget to do **Check and Save** the schematic view, and make sure there are no errors.

Useful Editing Hot Keys:

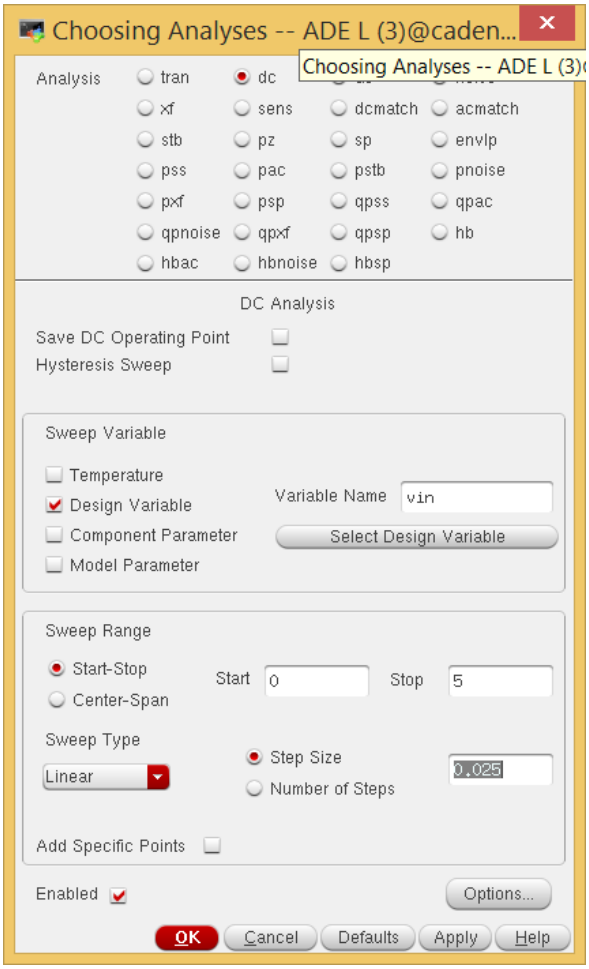
The following "hot keys" are available for the schematic editing tool.

- Press '**i**' to add instances (components)
- Press '**q**' on the device/instance selected to edit properties of the device
- Press '**w**' to add wires
- Press '**f**' fit the schematic in your schematic window
- Press '**l**' to label a wire
- Press '**u**' to undo an operation in the schematic window
- Press '**m**' to move objects
- Press '**DEL**' key to delete objects
- Press '**p**' to add pins
- Press '**z**' to zoom in the window
- Press '**shift+z**' to zoom out the window
- Press '**Up**' and '**Down**' arrow keys to move up and down within a schematic
- Press '**ESC**' key to terminate an operation in the schematic window

3) Launch **Spectre simulator and setting** it up:
   a. In the **Schematic Editor**, go to Toolbar, and Launch **ADE EXPLORER** and create a new view.
   b. Verify the model library and change section to “tt”.



   c. Load and define variables:
      From **ADE Editor**, choose *Variables -> Copy From Cellview*. It will get all the variables you defined in the Schematic Editor, ***vin*** , ***wp***, and ***wn***. Set their default values, e.g., vin=1.5V, wp=240n, and wn=120n.
   d. Simulation setup:
      From **ADE Editor**, choose *Analyses -> Choose*. You can choose a type of simulation and set conditions. Below is for a DC simulation over a range of variable sweep (***vin***=0:0.025:5).

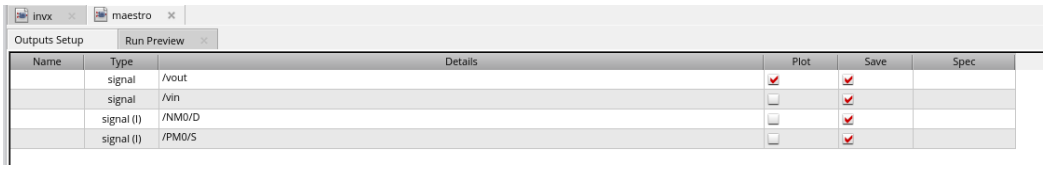


e. Set outputs:

From **ADE Editor**, choose *Outputs ->To Be Plotted -> Select On Design.* This will activate the Schematic Window allowing you to pick which signals (nets/wires/pins) you would like to have plotted during the simulation. Alternatively, you can select *Outputs ->To Be Saved -> Select On Design,* to capture data that you will need for later calculations but don't need to plot.

In the Schematic Window click on the wire and/or pin that are the input and the output of the inverter. Typing the **ESC** key will complete the output selection.

Below setting is for a case where the nets **vin** & **vout** and the pins **P0/S** (source of pMOS) & **N0/D** (drain of nMOS) are chosen to plot or save the input & output voltages and the currents through pMOS & nMOS, respectively. Among the outputs to be saved, the voltage ***vout*** is chosen to be plotted after each simulation.



f. **Save the simulation setup**:

The ADE state can be saved so that the settings do not need to be manually entered each time you want to simulate the design.

- To save the setting, in the **ADE Editor**, select *Session -> Save*.
  The "Save As" field does NOT have to be a unique name for all designs, since the state is saved within the directory of the current cell view.
- To load a saved setting, in the **ADE Editor**, select *Session -> Load State*.
  You can load any saved state, by browsing and choosing the one you want to load. You will like this to save your time later on.

4) Run the simulation

a. Extract netlist:

Before performing the simulation, you should create a netlist of your circuit, and it can be done in **ADE Editor**, by choosing *Simulation -> Netlist -> Recreate*. It will pop a window up that lists the netlist. You can close the netlist window.

b. Run simulation:

For a single simulation, you can simply push the ***green run button*** in the **ADE Editor** window. (Tip: The green run button is for 'netlist and run' which means clicking on the button firstly extracts the netlist and then runs the defined simulation, so you may skip the previous step of netlist extraction.)
When the simulation is completed, it will open a result window for the chosen outputs, like below:

5) **Analysis with Calculator** (Noise Margin):

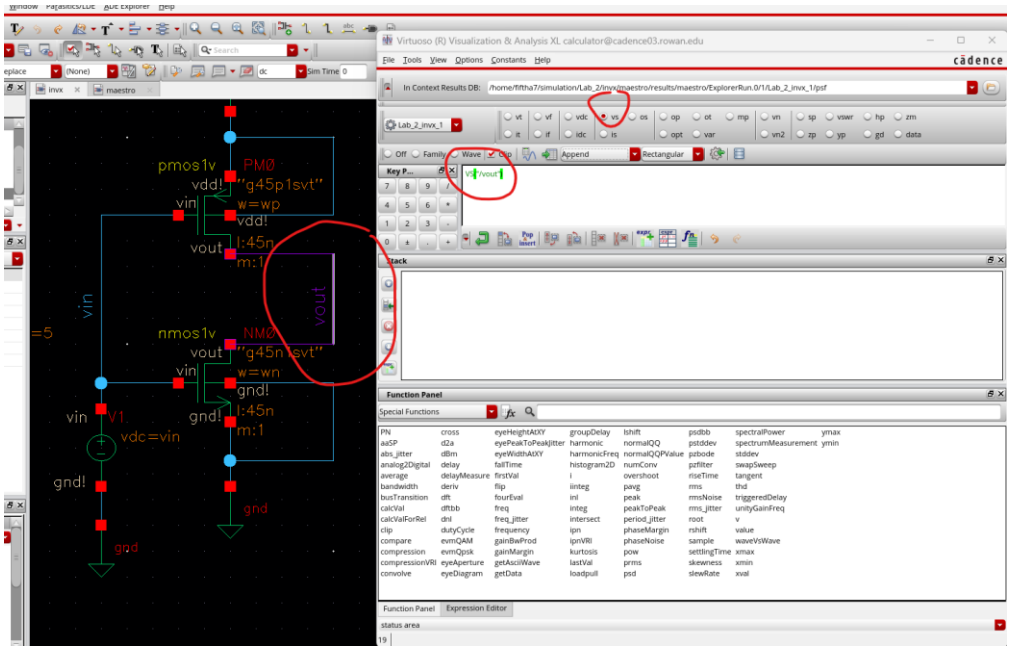
We're going to use the Calculator to compute the noise margin from the Voltage Transfer Curve (VTC) of the inverter.

[Pro Tip: The calculator utilizes information from the output files generated by the simulation, not the circuit. Remember those pins and nodes we didn't plot but saved, we now need those to perform calculations on.]

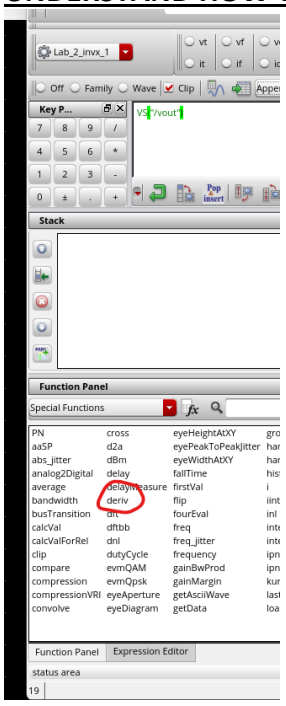
To open with the Calculator tool, from the **waveform window** choose *Tools-> Calculator*. *Tips*: The Calculator works with a "stack" in which you first input one or more operands (waveforms) and then you perform an operation on them, which implies that in order to compute '3-2' you should stack '2' first then '3', and then compute '3-2' by pressing '-'.

To calculate the *Noise Margin*, you will need to handle the VTC. (For details about the definitions of noise margins, please refer the lecture material.)

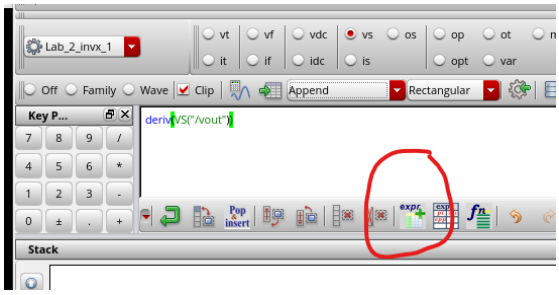
Find and select ***vs*** from the Calculator, and then click the ***vout*** net from the schematic window. That will display **VS("/vout")** in the Calculator.



Then search for a special function ***deriv*** and click on it. **(DO NOT JUST TYPE THESE EQUATIONS IN, USE THE FUNCTION PANEL AND REMAKE THEM SO YOU UNDERSTAND HOW TO USE THE CALCULATOR IN THE FUTURE.)**



That will display **deriv(VS("/vout"))**. Then add this to the expression editor.



If you hit enter in the calculator you can save these expressions in the stack and recall them later.

To calculate V_{IL}, V_{IH}, V_{OH}, and V_{OL}, following expressions can be used (the first 2 are fairly straign forward but I added pictures for the last 2):

V_{IL}: *cross(deriv(VS("/vout")) -1 1 "either" nil nil)*

⇨ Finds the lower input voltage at which the VTC slope equals -1

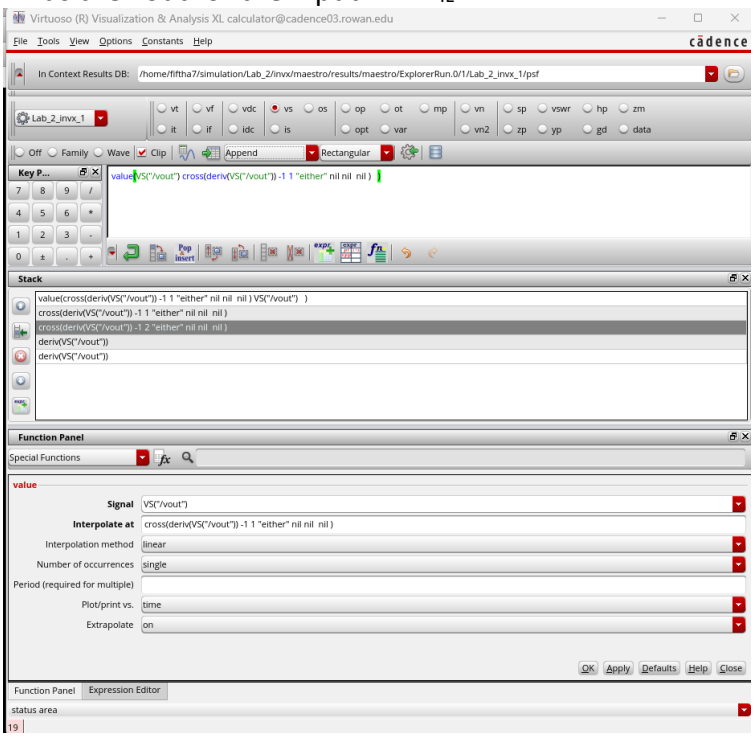
V_{IH}: *cross(deriv(VS("/vout")) -1 2 "either" nil nil)*

⇨ Finds the upper input voltage at which the VTC slope equals -1

V_{OH}: *value(VS("/vout") cross(deriv(VS("/vout")) -1 1 "either" nil nil))*

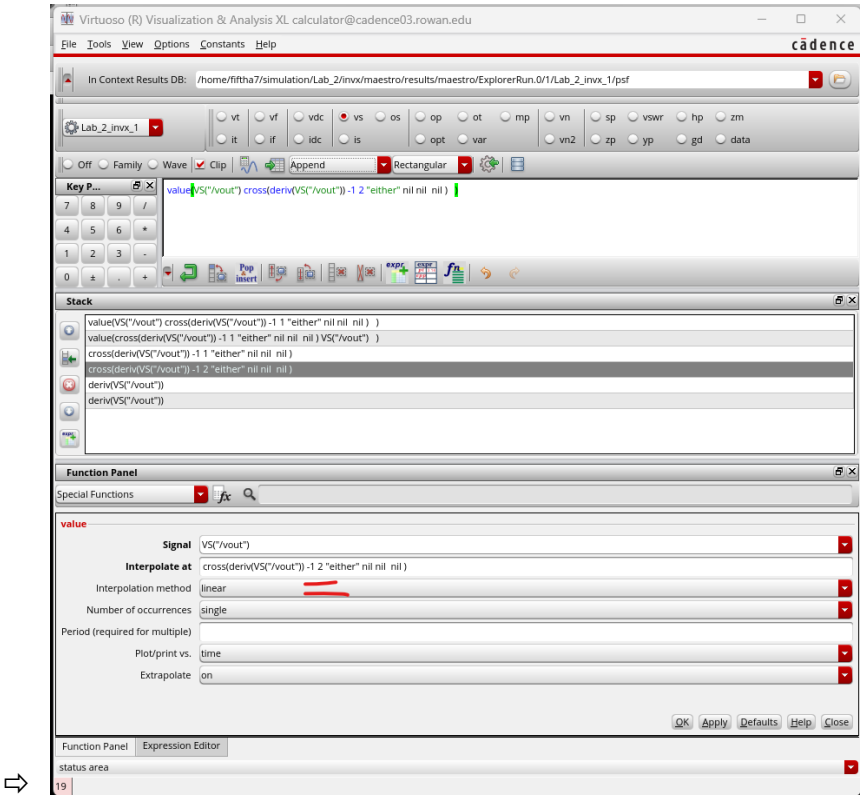
⇨ Finds the vout for the input vin=V_{IL}

⇨

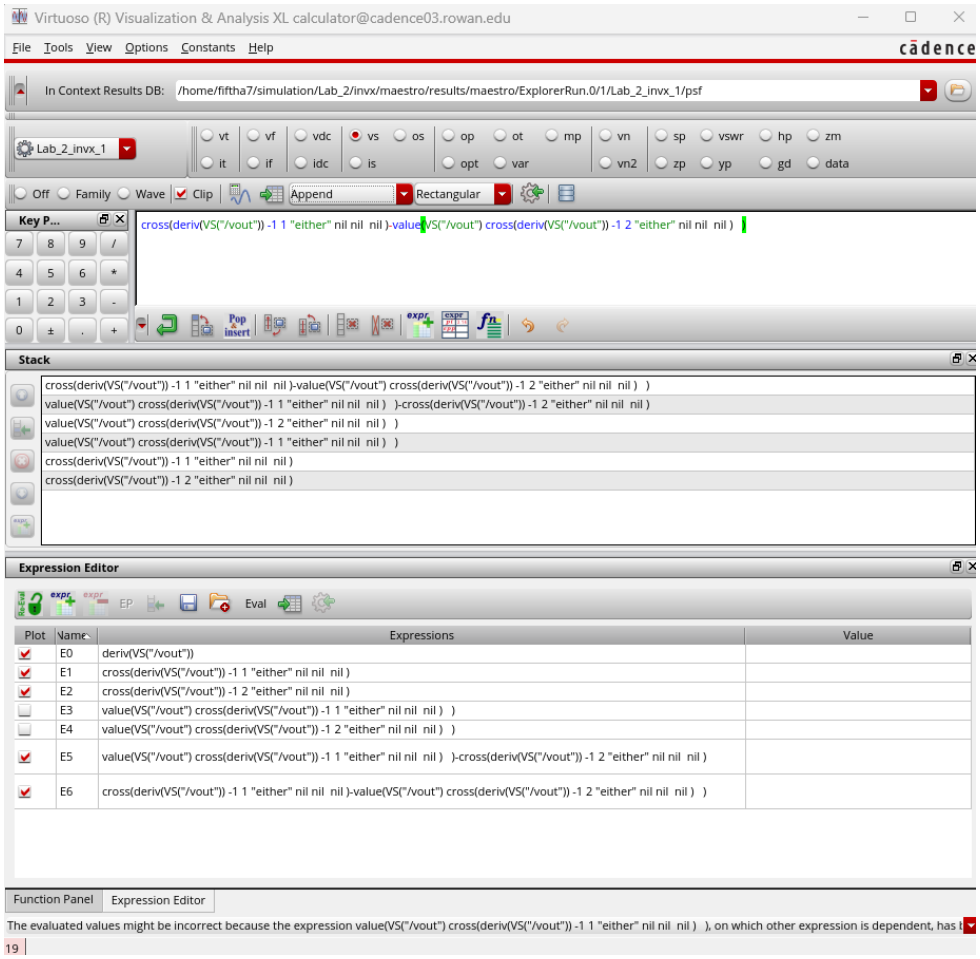


V_{OL}: *value(VS("/vout") cross(deriv(VS("/vout")) -1 2 "either" nil nil))*

⇨ Finds the vout for the input vin=V_{IH}

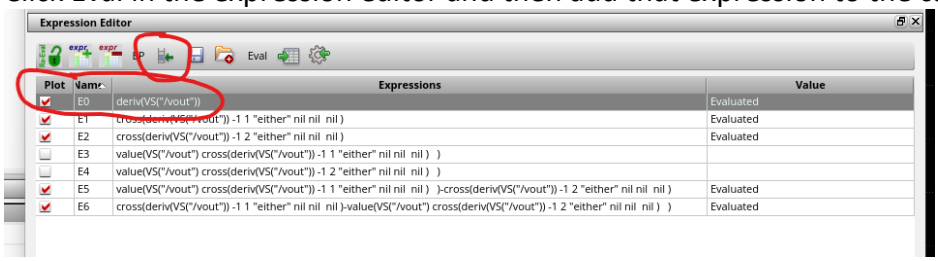


Then the noise margins can be calculated as: $\mathbf{NM_L = V_{IL}-V_{OL}}$, $\mathbf{NM_H = V_{OH}-V_{IH}}$. Your calculator should look like the image below when completed. Notice I saved each calculation in my Expression editor but didn't plot 2 intermediate steps, that's okay those were needed for the calculation.

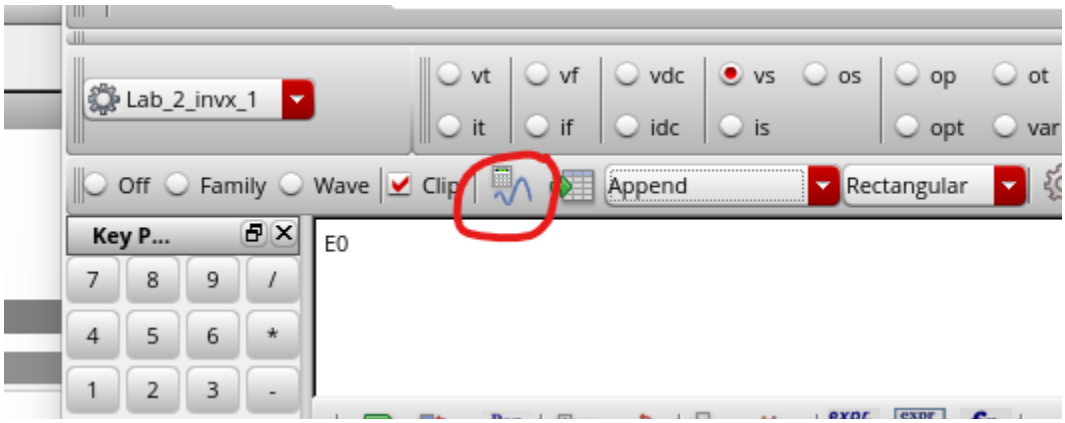


Computed results can be printed in the Calculator window or plotted in the waveform window. Meaning that if there is a single value it will just display that number and if there is a plot it will plot.

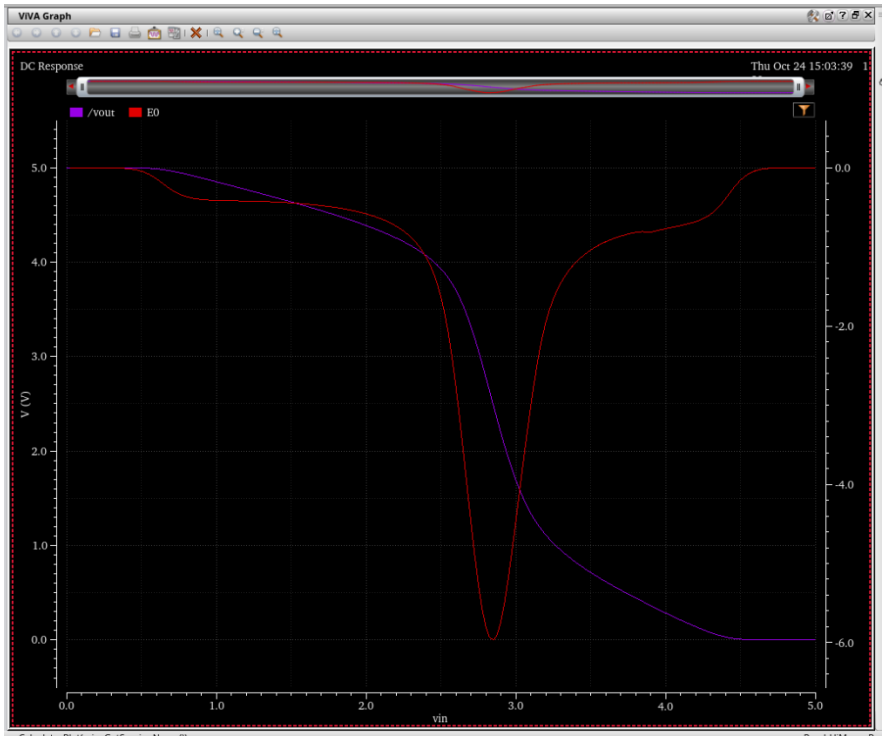
Click Eval in the expression editor and then add that expression to the calculator.



Then you can plot.



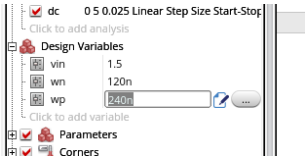
Below is a case where the results of *deriv(VT("/vout"))* is appended in blue to the *VT("/vout")* waveform in red.



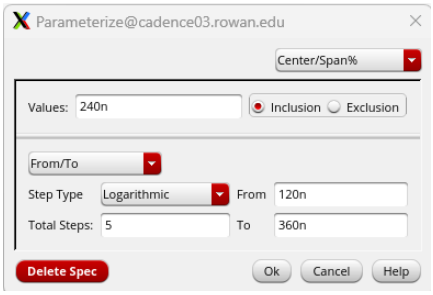
Q: What voltage is the switching point located? Why is it not Vdd/2?

Activity: Find the Noise Margins for logic levels LOW and HIGH.

6) **Parametric Analysis**:
Now we want to see how the noise margin changes over a range of transistor sizes. To run the simulation with a parameter sweep, select the 3 dots next to the design variable you want to adjust.

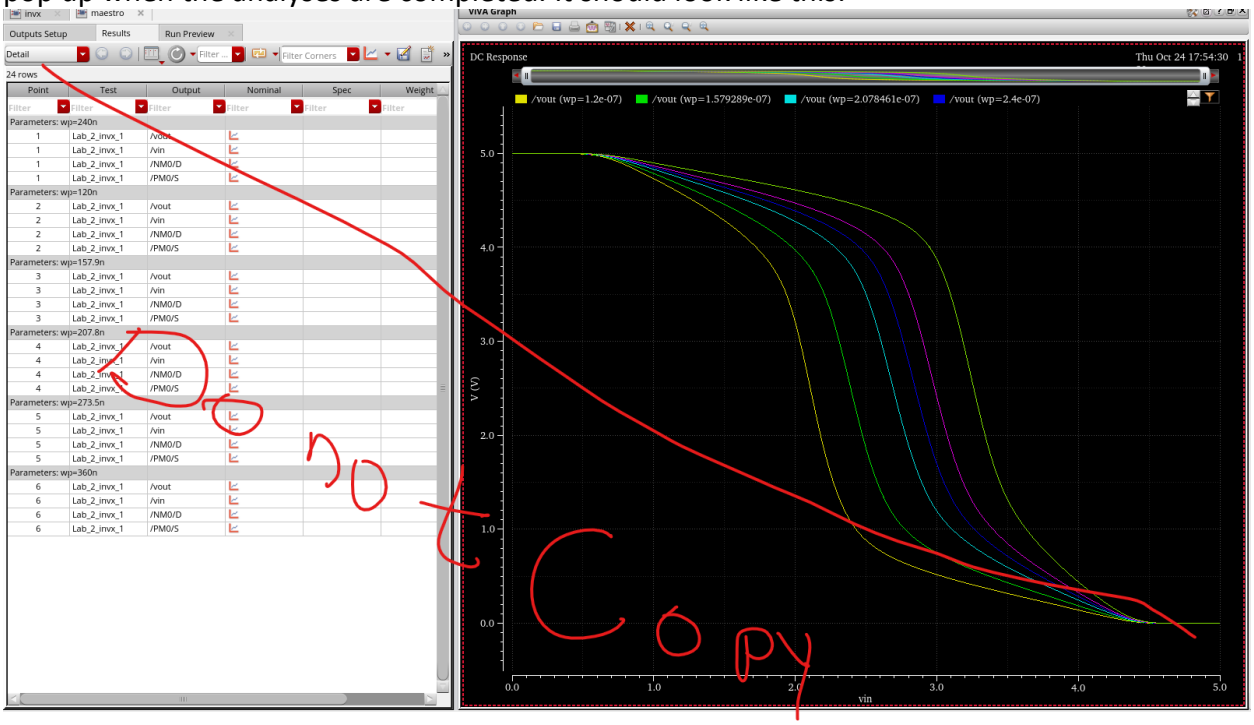


Below is the Parametric Analysis setting for the ***wp*** sweep.



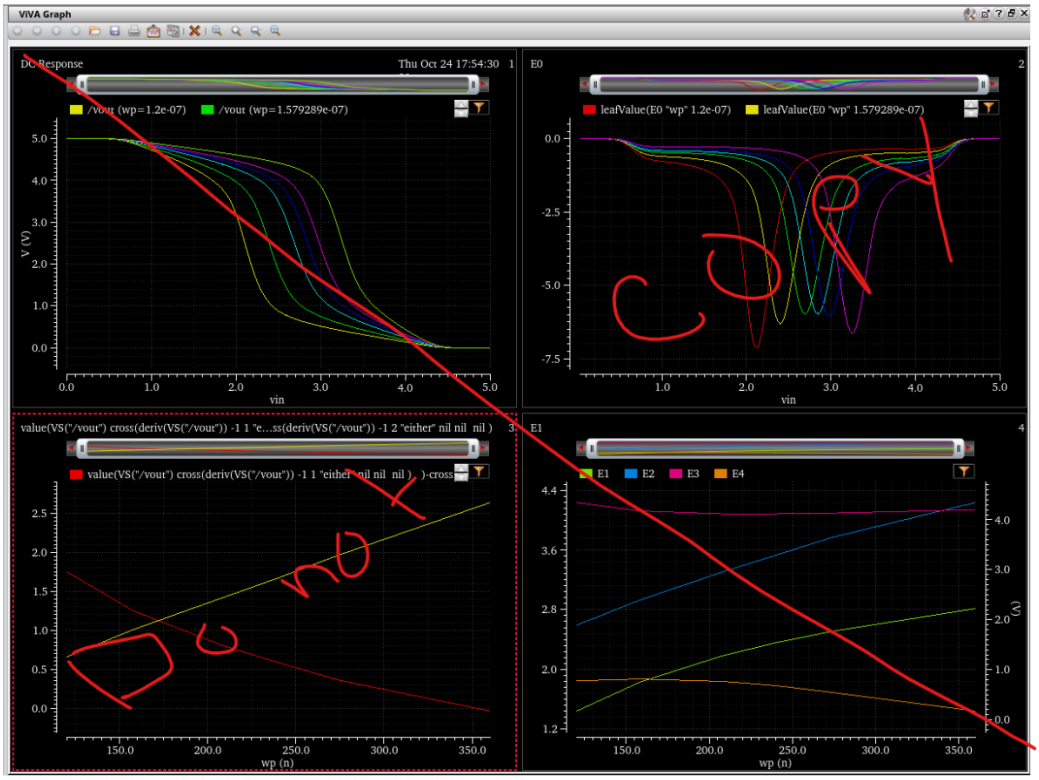
Obviously a 120n size will make the n:p ratio 1:1 instead of the normal 1:2 and the 360n will give a 1:3 ratio. You should choose sizes that make sense for your application.

After creating netlist from the **ADE Editor** (*Simulation -> Netlist -> Recreate*), you can run the parametric simulation by clicking the ***green run button***. The waveform window will pop up when the analyses are completed. It should look like this:



With the expressions stacked in the Calculator, you can now draw the trend of noise margin over the range of transistor size. The calculator has a “Append” setting with a dropdown menu, select “New Subwindow” to save a plot to a new window. You will need to change this back to “Append” for adding plot likes to a single plot.

In the below waveform window, the first section displays the VTC over the sweep range, the 3rd panel shows the slopes of VTC (in other words, gain), the 2nd one shows the computed V_{IL}, V_{IH}, V_{OH}, and V_{OL}, and the 4th one shows the NM_H and NM_L.
[Note: These equations have already been created, you only need to apply them to the current setup with a changing "wp".]



If you successfully get similar graphs showing the trends of noise margins, you're done for this lab. Please summarize what you've learned in this lab and put all relevant results into your report, and qualitatively compare the results you got against the theoretical expectations.

Lab report Submission:

Please use the AppNote format provided on Canvas. Please do not make your own 2 column report, I can see the difference. **Don't use google docs, the formatting is not saved when converting, use word. If you do not have word on your personal computer, you're in luck, your virtual student desktop has the full 365 suite… please use it.**

Part Reference list for the gpdk045 library:

11.1 Mosfets

- nmos1v - 1.1 volt nominal Vt NMOS transistor
- nmos1v_3 - 1.1 volt nominal Vt NMOS transistor with inherited BULK
- nmos1v_hvt - 1.1 volt high Vt NMOS transistor
- nmos1v_hvt-3 - 1.1 volt high Vt NMOS transistor with inherited BULK
- nmos1v_lvt – 1.1 volt low Vt NMOS transistor
- nmos1v_lvt_3 – 1.1 volt low Vt NMOS transistor with inherited BULK
- nmos1v_nat – 1.1 volt native Vt NMOS transistor
- nmos1v_nat_3 – 1.1 volt native Vt NMOS transistor with inherited BULK
- nmos2v – 1.8 volt nominal Vt NMOS transistor
- nmos2v_3 – 1.8 volt nominal Vt NMOS transistor with inherited BULK
- nmos2v_nat – 1.8 volt native Vt NMOS transistor
- nmos2v_nat_3 – 1.8 volt native Vt NMOS transistor with inherited BULK
- pmos1v – 1.1 volt nominal Vt PMOS transistor
- pmos1v_3 – 1.1 volt nominal Vt PMOS transistor with inherited BULK
- pmos1v_hvt – 1.1 volt high Vt PMOS transistor
- pmos1v_hvt_3 – 1.1 volt high Vt PMOS transistor with inherited BULK
- pmos1v_lvt – 1.1 volt low Vt PMOS transistor
- pmos1v_lvt_3 – 1.1 volt low Vt PMOS transistor with inherited BULK
- pmos2v - 1.8 volt nominal Vt PMOS transistor
- pmos2v_3 - 1.8 volt nominal Vt PMOS transistor with inherited BULK