

Due Tuesday, Jan. 25th, 2022

Please get screen captures of all schematics, layout, and simulation results/plots. Submit your report electronically through Canvas.

Introduction

Hand-analyzing circuits is demanding and often intractable. For this reason, circuit designers of all types rely on computer aided design (CAD) tools to analyze complicated circuits. Cadence Virtuoso® is the industry standard for integrated circuit design and includes all the features necessary to design a complete system on chip (SoC) from schematic capture to layout and fabrication. This tutorial looks at basic models used for simulating MOSFET transistors, following hand-calculations from homework problems in Gray and Meyer's *Analysis and Design of Analog Integrated Circuits* 5ed.

Linux/Cadence Setup

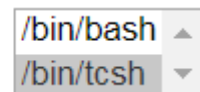
Modern integrated circuit (IC) design tools run exclusively on Linux platforms. This tutorial assumes no Linux or Cadence experience. Even if you are familiar with the tools, please try to organize your directory structure like that in this tutorial to simplify grading and debug.

Pre-Cadence Setup

Before starting Cadence, we need to appropriately configure a working directory and local environment variables. First and foremost, make sure your login shell is tcsh. The default shell is bash. To change this permanently, manage your EE account on the department webpage (link below) and set your login shell to bin/tcsh, like shown:

<https://user.ee.washington.edu/eeuser/account>

Login Shell



Setting up your Linux Environment to launch Cadence Design Tools

Next, we must tell the login shell which environment variables to instantiate on login. This is done with a .cshrc file for the bash shell. To copy a pre-made .cshrc file to your own account, enter the following commands into a Linux terminal:

```
cp /home/lab.apps/vlsiapps/cshrc/EE332/.cshrc ~/.  
source ~/.cshrc
```

Next, create and access your cadence working directory with the following commands:

```
mkdir ~/EE332  
mkdir ~/EE332/cadence  
cd ~/EE332/cadence
```

note: Please use directory names that are commensurate with EE 473 / EE 538. For example, the last command from above should be “cd ~/EE473/cadence”. Please swap out EE473 for EE332 when appropriate for the rest of the tutorial.

Since the cadence directory is empty, use the copy command `cp` to add the setup files for cadence:

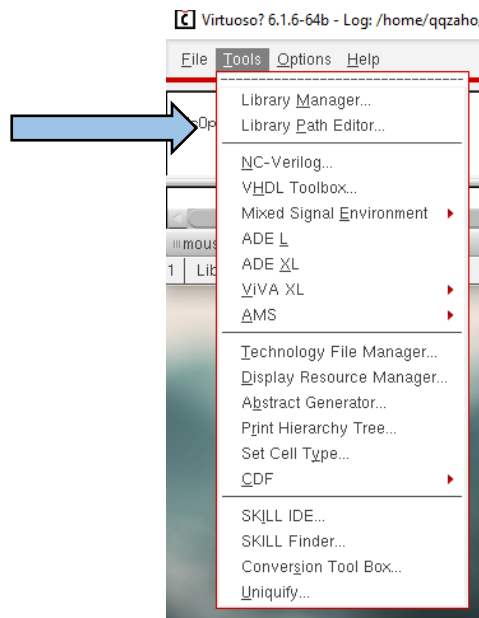
```
cp /home/lab.apps/vlsiapps/cshrc/EE332/.cdsinit ./
cp /home/lab.apps/vlsiapps/cshrc/EE332/display.drf ./
cp /home/lab.apps/vlsiapps/cshrc/EE332/.cdsenv ./
cp /home/lab.apps/vlsiapps/cshrc/EE332/cds.lib ./
```

You can now run Cadence from the EE332/cadence directory just created. Double check that you have the four copied files and start Cadence with the following command (virtuoso is the command to start Cadence):

```
ls -al
virtuoso &1
```

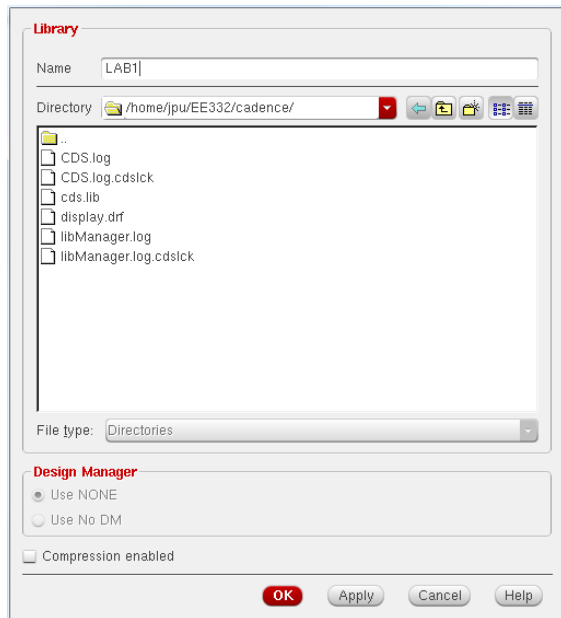
Cadence Library Setup

There is a lot to the Cadence design environment. While Cadence is extremely complex (yet, VERY powerful), for this class we will only describe the features you need to complete the homework and projects. However, feel free to experiment and explore the different Analyses tools available in the Cadence environment. This tutorial will cover the bare bones features necessary for schematic simulation. The first is to create your own library, this is how you will store your circuit schematics, simulations and eventually layout. To create a library for your Lab 1 assignment, go to your Virtuoso window, select the “**Tools**” menu and then select “**Library Manager**”, as shown below:

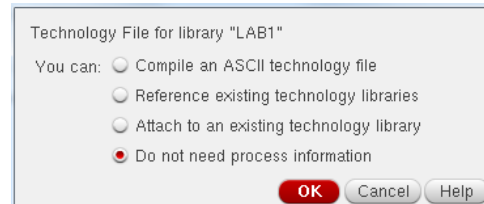


In the Library Manager window, open **File** -> **New: Library** and create a library called **LAB1** in your cadence working directory, as shown below:

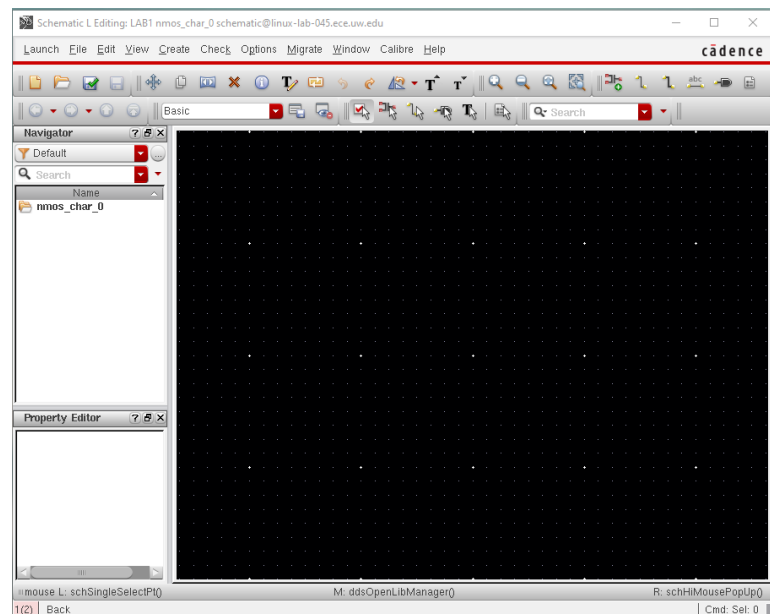
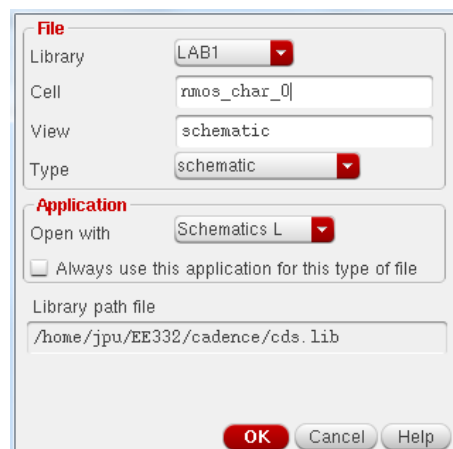
¹ The “&” indicates that you want to run Cadence in parallel with other commands or functions. If you do not include “&” and want to run a command in the terminal, you would need to exit Cadence to run the new command.



You will be asked to attach a technology file.
Select **“Do not need process information,”** as shown below:



With your new library selected, open **File -> New -> CellView** in the Library Manager window. The below popup menu should appear. Create a schematic cellview with a descriptive name, as shown below on the left. Note that spaces and Linux file/directory names do not get along. Use underscores instead of spaces in all Linux names.



Setup is now complete, and a window should open up that looks like the picture on the right. The following steps will guide you through populating your schematic and simulating the transfer characteristics of a MOSFET device.

MOSFET Drain Current Curves

The models used for simulating transistors in modern integrated circuit (IC) processes are governed by hundreds of parameters and match tested device behavior within very close margins. This introduction starts with the simplest MOSFET model, so that the corresponding hand-calculations do not require a degree in mathematics.

This tutorial starts with **Problem 1.15** from *Gray and Meyer's* (another classic circuit textbook):

1.15 An NMOS transistor has parameters $W = 10\text{ }\mu\text{m}$, $L = 1\text{ }\mu\text{m}$, $k' = 194\text{ }\mu\text{A/V}^2$, $\lambda = 0.024\text{ V}^{-1}$, $t_{ox} = 80\text{ }\text{\AA}$, $\phi_f = 0.3\text{ V}$, $V_{t0} = 0.6\text{ V}$, and $N_A = 5 \times 10^{15}\text{ atoms/cm}^3$. Ignore velocity saturation effects.

(a) Sketch the I_D - V_{DS} characteristics for V_{DS} from 0 to 3 V and $V_{GS} = 0.5\text{ V}$, 1.5 V , and 3 V . Assume $V_{SB} = 0$.

(b) Sketch the I_D - V_{GS} characteristics for $V_{DS} = 2\text{ V}$ as V_{GS} varies from 0 to 2 V with $V_{SB} = 0$, 0.5 V , and 1 V .

Problem 1.15.a.

The schematic for testing the transistor for **problem 1.15a** should have an NMOS device, two DC voltage sources (for V_{DS} , and V_{GS}), and a ground reference, looking something like this:

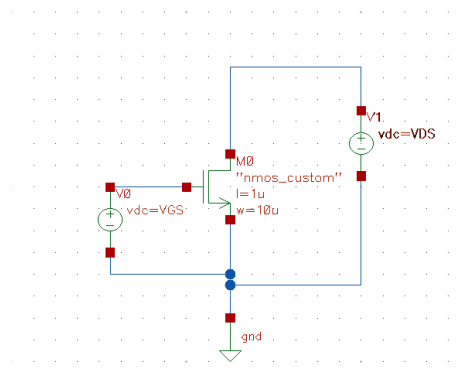
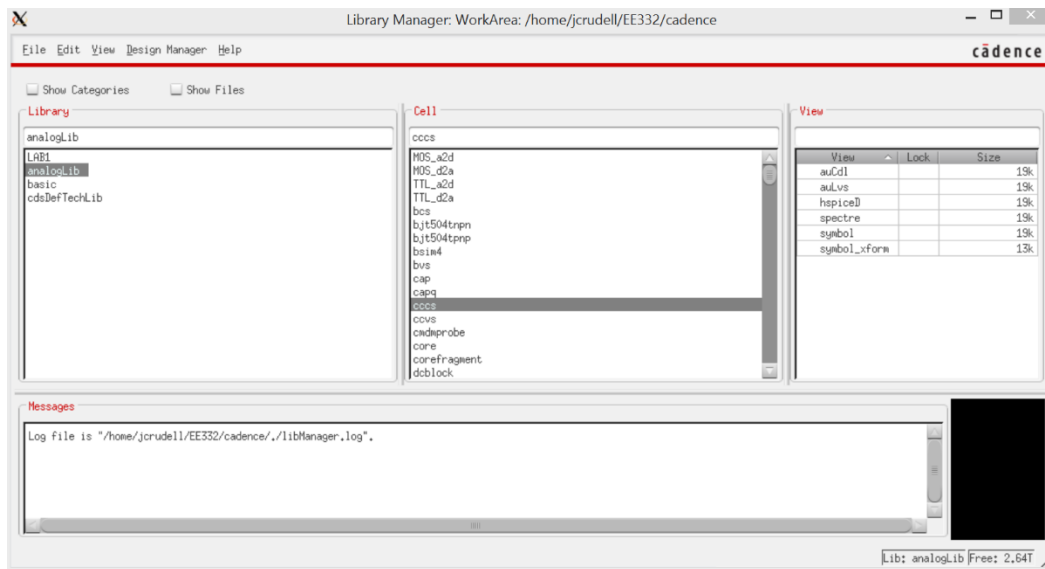


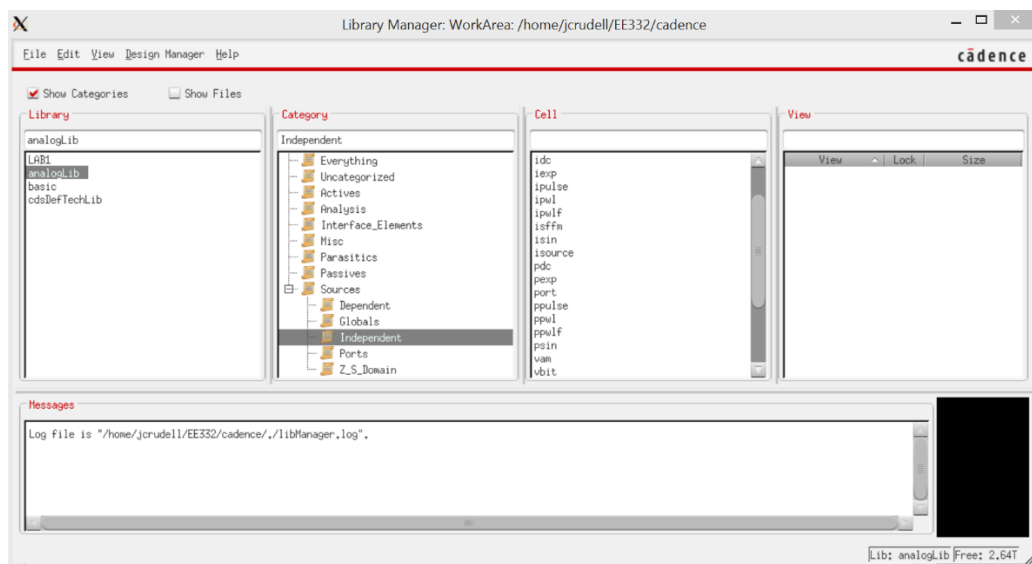
Figure 1: NMOS transistor schematic to simulation “I-V Characteristics”.

Before drawing the schematic above, a quick tutorial on the organization of components and cells will be presented. All the components in this schematic are found in another library which should appear in your “**Library Manager**”. The different libraries you have appear in the left column labelled “**Library**” as shown below. Click on “**AnalogLib**” and you should see a whole bunch of cell names appear in the middle column labeled “**Cell**”, again as shown below:



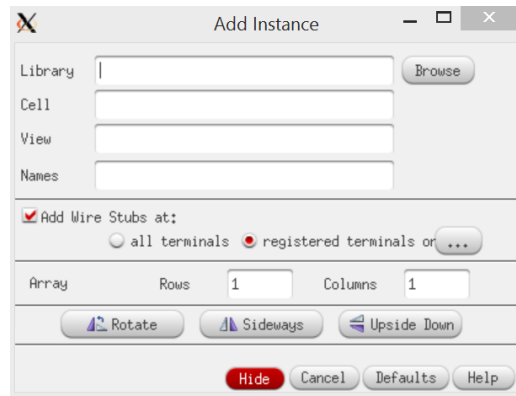
Click on one of the cells in the middle, you will notice something pops up in the column labeled, “**View**”. For the moment, we will not concern ourselves with the different views. However, later when becoming more familiar with Cadence, you will use these different views for simulation purposes.

Another feature which is quite powerful in the Library Manager window is the ability to categorize different cell views. On the upper right, click on the button that says, “**Show Categories**” and a new column appears called “**Category**”. Now all the cell views are organized in categories which makes cell easier to find based on type or classification. For example, click on “**Sources**”, then “**Independent**”. Now in the “**Cell**” column, you will see all the *independent* voltage and current sources that are available in the **analogLib** library, see below. This includes voltage and current pulse generators, sinewaves, DC sources, etc. If you click on the Category, **Sources -> Dependent**, you will see a bunch of different dependent current and voltages sources, such as “**cccs**” which is a *current controlled current source* (cccs).



Now that you guys have graduated to the level of Cadence Librarians, we can move on to build the schematic in **figure 1**. All the components in **figure 1** come from the **analogLib** library. Go back to your schematic named, “**nmos_char_0**”. To place components, or “**instantiate**” them, there are multiple ways of

doing this. The brute force method is to go to the top of your schematic and select **Create - > Instance**. There will be a “popup” menu as shown below.



Let's start by placing the NMOS in the schematic. To do this, fill in the menu with **Library = “analogLib”** (note: the names are case sensitive), **Cell = “nmos”**, and **View = “symbol”**. Now pass the mouse over the schematic and you will see a shadow of a nmos transistor following the cursor. Simply left click your mouse and this will place the transistor in the schematic. It was quite a bit of work to place this transistor and there are faster ways of doing things in the Cadence by using “**Speed keys**”. Rather than going to the “**Create**” menu at the top, try putting the cursor in the schematic then hit the **I** key, you should see the same “**Add Instance**” menu just popup. Again, in the “**Add Instance Form**,” specify the library (analogLib), cell name (nmos), and view name (symbol). The example below shows the form for the NMOS model:

Library: analogLib
Cell: nmos
View: symbol
Names:
☐ Add Wire Stubs at:
☐ all terminals ☒ registered terminals only
Array: Rows: 1 Columns: 1

Model name: nmos_custom
Width: 10u M
Length: 1u M
Source diffusion area:
Drain diffusion area:
Source diffusion periphery:
Drain diffusion periphery:
Drain diffusion res squares:
Source diffusion res squares:
Drain diffusion length:
Source diffusion length:
Multiplier:
Temp rise from ambient:
Source/drain selector:
Additional drain resistance:
Additional source resistance:
Dist. OD _poly(one side):
Dist. OD _poly(other side):
Dist. betn neighbour fingers:
Estimated operating region:
Hot-electron degradation:
Bulk node connection: S

Notice that some of the parameters for the NMOS model have been filled out. This parameter list shows the end-user-changeable parameters for the nmos model. More detailed parameters will be added to the **nmos_custom.scs** model file, to be created later. For now, input the model name and drawn transistor dimensions from **Problem 1.15**.

Place a ground reference (**gnd**), also from the **analogLib** library.

Drawing Wires

Then, wire the voltage sources so that they are referenced to ground and driving the corresponding terminals of the NMOS device. The wiring tool can be accessed with the **W** key. To start a wire, place it over one of the terminals of either the transistor, or one of the sources; you should notice that a small diamond will light up around the terminal. Left click will start the wire, then move the mouse around to the point where you want to connect the wire, click again to complete the wire. You can also make turns with the wire by clicking in open spaces (makes a corner). If you make a mistake with the wire, hit “esc”. If you want to delete the wire, pass the cursor over the wire and click on the section of wire that you want to get rid of. This should highlight that section, then hit the delete key.

Note: for the transistor sizes, just type “10u” which will appear as “10u M” which is 10 micrometers. The “M” appears on its own, you don’t need to add it.

Now, you need to finish the schematic by adding some voltage sources, which are “**vdc**” components in “**analogLib**”. Place these components in the schematic and arrange similar to **fig. 1**.

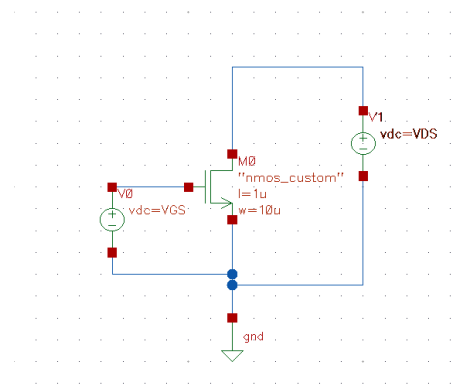
Library: analogLib
Cell: vdc
View: symbol
Names:
☒ Add Wire Stubs at:
☐ all terminals ☒ registered terminals only
Array: Rows: 1 Columns: 1

Noise file name:
Number of noise/freq pairs: 0
DC voltage: VGS V
AC magnitude:
AC phase:
XF magnitude:
PAC magnitude:
PAC phase:
Temperature coefficient 1:
Temperature coefficient 2:
Nominal temperature:

Before placing the “**vdc**” components, add parameters for the DC output voltages in the model under the entry “**DC voltage**”. This can be a number or a variable. We will use variables “**VGS**” and “**VDS**” for the voltage source placed at the gate and drain of the device, respectively. To add a ground symbol, place the components **Library: analogLib, Cell: gnd**.

A couple tips:

- ESC – will exit a popup menu or take you out of any editing mode that you might be in. For example, after placing a device, the mouse might have a new device attached. To get rid of the new component hit ESC. Likewise, if you are drawing wires and want to leave that mode, hit ESC.
- Q – to access the properties of a device. To access a device's parameters (this could be anything, transistors, resistors, capacitors, sources, etc). First, put the cursor over the device, left click to select the device, then hit "Q", this will popup the properties menu of that device. Try this for the voltage sources and transistor that are in your schematic. Note: Cadence is funny about popping up menus. You might hit "Q", and nothing happens because the menu is already opened behind the schematic. If this happens, check behind your schematic and other Cadence windows...again, a feature of Cadence that isn't perfect.
- F – Fits the entire schematic to the size of the open window. Very useful for re-orienting yourself in large designs,



“Check and save” the circuit with this button in the “hotbar” near the top left:



Always “Check and Save” your circuit whenever a change is made. You can just “save” the schematic, but without “Check and Save” you cannot run simulations. In short, Cadence will “check” to make sure the schematic is drawn without any obvious electrical errors before you run a simulation. If everything is drawn correctly then the figure 1 schematic should not throw any warnings or errors.

OK, time to simulate your schematic!

The next step is to make a custom model file for the NMOS device that includes k' , λ , t_{ox} , ϕ_f , V_{t0} , and N_A . The simulator comes pre-packaged with MOSFET models similar to the equations given in *Gray and Meyer*. This is easily done with a single text file. **These model files are typically provided by the foundry for a given integrated circuit process.** This tutorial creates a custom model in order to exactly match the device parameters from the *Gray and Meyer* hand analysis.

Cadence, by default, uses the **Spectre** simulation language. It is very similar to Spice but has been optimized for many specialty simulations.

The syntax for a **Spectre** nmos model is as below:

Spectre:	<code>model mname mos1 type= [n p] [param= value]*</code>
----------	---

Go to your linux terminal and create a file “**nmos_custom.scs**” in your Cadence working directory with your favorite Linux text editor.

For those of you unfamiliar with Linux, you can create and open a text file with a GUI editor with the following command:

```
gedit nmos_custom.scs &
```

Following the Spectre syntax, the file should contain the following text:

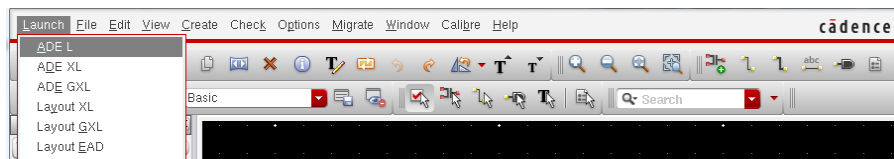
```
1 model nmos_custom mos1 level=1 type=nmos kp= lambda= tox= phi= vto= nsub=
```

Input the parameters given in the problem statement in scientific format. For example, $k' = 194\mu A/V^2$ would look like this:

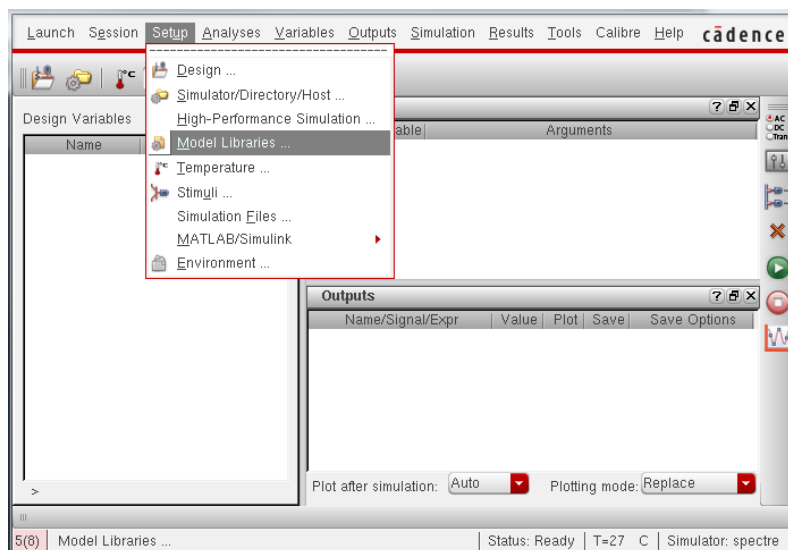
```
kp=1.94e-4
```

Fill in all the device parameters from problem **1.15 of G&M**. Remember that phi is ϕ (problem statement) and nsub is N_A . Finish off and save your model file and return to the Cadence environment.

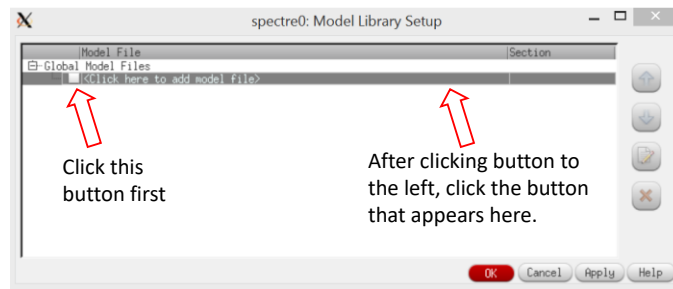
We will reference this file from the Cadence simulation environment. Back in the Cadence schematic window, start the simulation tool with **Launch -> ADE L**, as shown below:



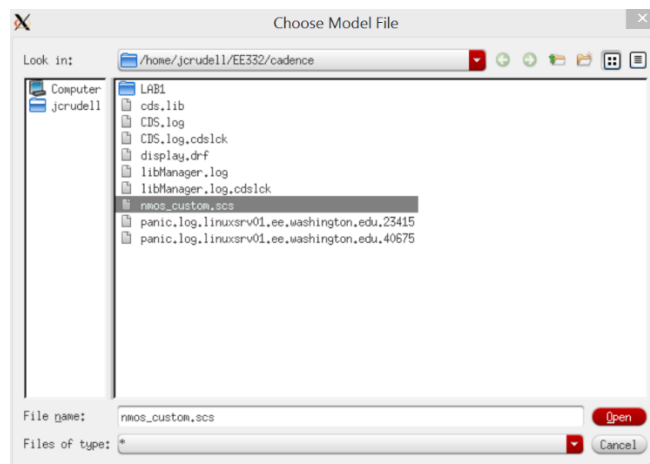
Welcome to the “**Analog Design Environment**” or ADE. This is the main menu to setup and run very sophisticated simulations (although in EE 332 they will not be that complicated). Part of setting up a simulation is to define your models and variables before selecting the type of simulation analysis to run; this is all done using the ADE menu. To reference the model file just created open **Setup -> Model Libraries** and add the “**nmos_custom.scs**” file to the list.



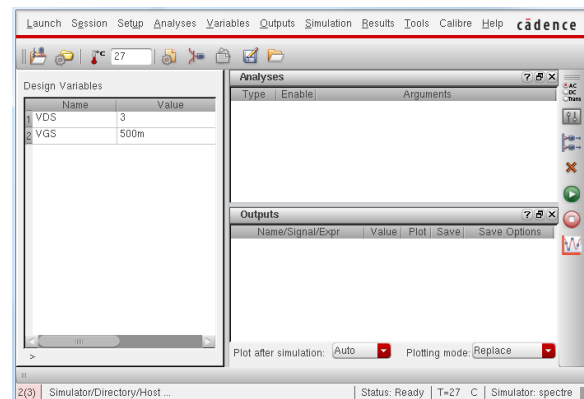
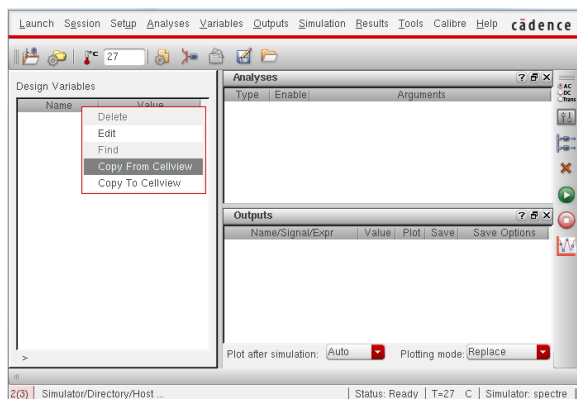
The below menu will appear, click on both the buttons shown.



Now, you should get a file browser that pops up as shown below. Select your model file that is in your Cadence directory.



Now your model file is loaded into ADE. Next, we need to tell ADE that you have variables in your schematic. Cadence will look for all the variables in your schematic and “pull” in these variables from your parameter definitions that you set for your DC voltage sources. To do this, **right-click in the Design Variables** section and select “**Copy from Cellview**” as shown below. This searches your schematic for unset parameters, and they will appear in this menu, as shown on the ADE window on the right. Give VDS and VGS nominal values for now of “3” (3 volts), and “500m” (500mVolts), respectively. Again, this is shown on the right. You can edit the variables VDS and VGS by placing the mouse over the variable, then double click. Again, you’ll get a menu where you define your variable. Later we’ll actually sweep those variables using a simulation technique called parametric sweep.



Analysis: ☐ tran ☒ dc ☐ ac ☐ noise
☐ xf ☐ sens ☐ dcmatch ☐ stb
☐ pz ☐ sp ☐ envlp ☐ pss
☐ pac ☐ pstb ☐ pnoise ☐ pxf
☐ psp ☐ qpss ☐ qpac ☐ qpnoise
☐ qpxf ☐ qpdp ☐ hb ☐ hbac
☐ hbnnoise ☐ hbsp

DC Analysis

Save DC Operating Point ☒
Hysteresis Sweep ☐

Sweep Variable

☐ Temperature
☒ Design Variable Variable Name:
☐ Component Parameter
☐ Model Parameter

Sweep Range

☒ Start-Stop Start: Stop:
☐ Center-Span

Sweep Type

Add Specific Points ☐

Enabled ☒

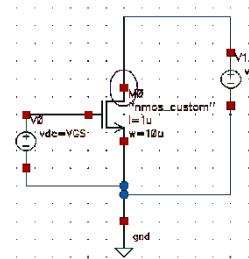
Now that the models and variables are properly setup, we are ready to pick the type of simulation that we'd like to perform. To do this, Go to the “**Analyses**” menu at the top of the ADE window, hit “**Choose**” and the menu shown to the left will pop up. Open Analyses: Choose and select DC analysis. Check the “**Save DC Operating Point**” radiobox, select the Design Variable, VDS, as your Sweep Variable. Sweep from 0 to 3 volts, as defined in the problem statement. We will repeat this simulation several times for each specified value of VGS. Exit the Analyses window.

Now go to the “**Outputs**” menu at the top of your ADE window, select “**Setup**” and the below menu should appear. Hit “**From Schematic**” which allows you to pick the node or net to plot and store results from the simulation.

Click the drain terminal of the nmos device and a blinking circle should appear:

Selected Output

Name (opt):
Expression:
Calculator:
Will be: ☒ Plotted/Evaluated



The drain current should now be on the outputs list. Make sure both Plot and Save are checked, as below:

Launch Session Setup Analyses Variables Outputs Simulation Results Tools Calibre Help cadence

Design Variables

Name	Value
1 VDS	3
2 VGS	500m

Analyses

Type	Enable	Arguments
dc	<input checked="" type="checkbox"/>	t 0 3 Automatic Start-Stop

Outputs

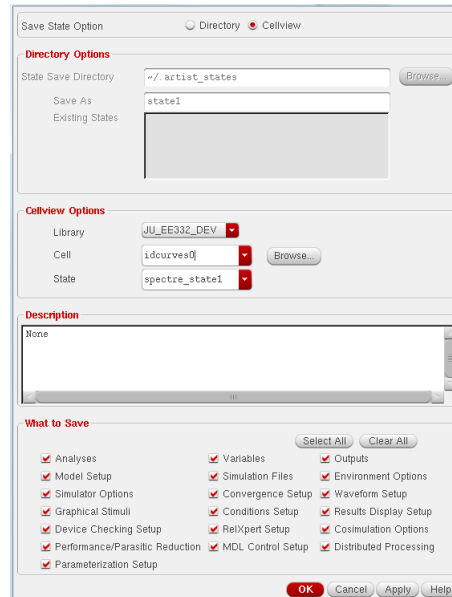
Name/Signal/Expr	Value	Plot	Save	Save Options
M0/D		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	no

Plot after simulation: Plotting mode:

> Select on Schematic Outputs to Be Plotted

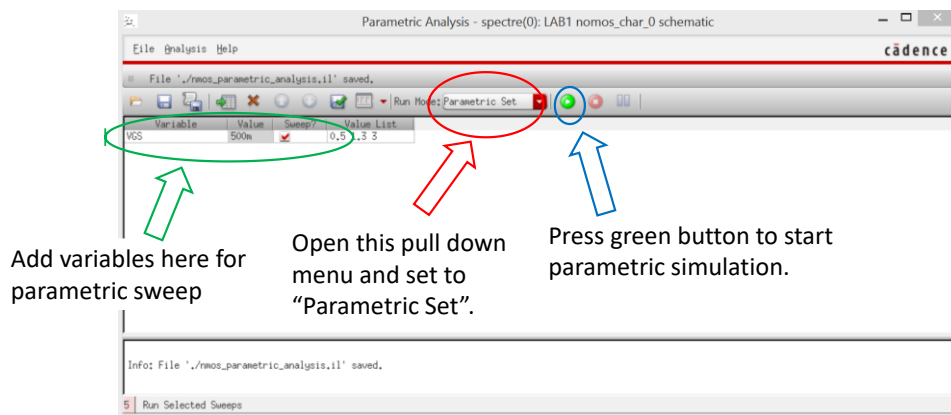
2(3) Model Li | Status: Selecting outputs to be plotted... | T=27 C | Simulator: spectre | State: spectre_state1

It's good practice to save your configurations for later use. At the top of the ADE window, go **Sessions -> Save State** and save the state to the “**cellview**,” as shown below. This saves the ADE configuration with your schematic which is convenient when you want to return to this simulation at a later time. To see this, go back to the Library manager, click on your library “**Lab1**”, then click on your cell “**nmos_char_0**”, on the right, you'll under the View column, the name of your spectre state. Later you can open this up and the ADE menu opens with everything that you just entered including type of simulation analyses, variable, model files, etc.



Finally, we will set up the parametric sweep over the values of VGS given in the problem. Open the **Tools -> Parametric Analysis** window and enter a “**Parametric Set**” type run over VGS, as shown below:

This runs the DC analysis configured earlier for each value of VGS in the Value List.

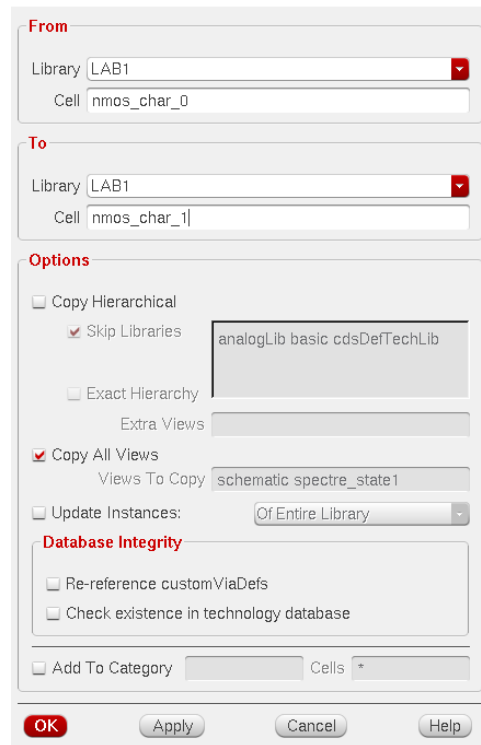


Now it's time to simulate! Press the “**Netlist and Run**” button in the Parametric Analysis window to run the simulation and plot the results. Another window will open and show three I-V curves indicating the different VGS values being used. Save a plot of your results for submission, this can be done with **File -> Save Image** in the waveform viewer window. You may need to change the background color of the plot to something other than black to see the waveforms on the printout.

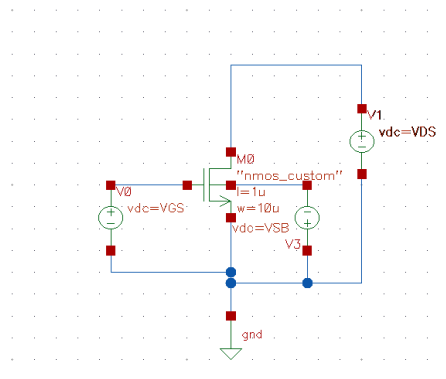
Problem 1.15.b.

This problem asks for a sweep of VGS for different source-body bias voltages. The nmos model used in the previous problem did not have a body terminal, so the schematic requires slight modification.

It is best to organize your work in separate cellviews, to prevent having to re-do any setup later. In the **Library Manager**, copy the cellview used for **Part a** into a different name by right-clicking on it. A window like below should pop up and fill out the window as shown below. This replicates the schematic and ADE setup defined for the previous section.



Open the new schematic and remove the NMOS device. Replace it with a “nmos4” device from the analogLib library, with the same parameters **Part a**. This adds the body terminal to the device model when it was previously assumed that the source and body were tied together. **Add a third voltage source to negatively bias the body with respect to the source, with a parameterized dc voltage, VSB.** The “Quick Key” for rotate is “R”. The schematic should now look like the example below:



Once finished, “**Netlist and Save**” the schematic, and open the ADE L window again. Open **Session -> Load State** to pull in the simulation parameters from before. **Note!** This simulation is still set to simulate with the previous schematic from Part a. Open **Setup -> Design** and select the new schematic, or your simulations will not include the body bias modification. The top of the ADE L window will say “*LAB1 nmos_char_1 schematic*” to indicate its pointing towards the new schematic with the body bias.

To modify the simulation for **Part b**, pull in the VSB parameter in the Design Variables window, as before. In the “Design Variables” section, **set VSB to 0V and VDS to 2V**. Change the DC sweep parameter under “Analyses” to **VGS from 0V to 2V**.

Now, open the Parametric Analysis window again, and set up a sweep for **VSB = 0V, 0.5V, 1V**. Run this parametric sweep to complete Part b.

Make sure to save your plot and simulation setup!

Deliverables

Please print out a copy of the results generated from both problems and add 2 or 3 sentences to describe the result. **Before the next lab session**, please upload it through canvas. The plots should be clearly visible on a white background. This can be done in the **Graph -> Properties** menu, as shown below:

