EE 332 CAD Lab 1 Autumn 2022

Due at the beginning of next lab session.

Please contact Cindy Liu (xindil@uw.edu) for questions. Please get screen captures of all schematics 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 similar to that in the tutorial to simplify grading and debug.

Pre-Cadence Setup

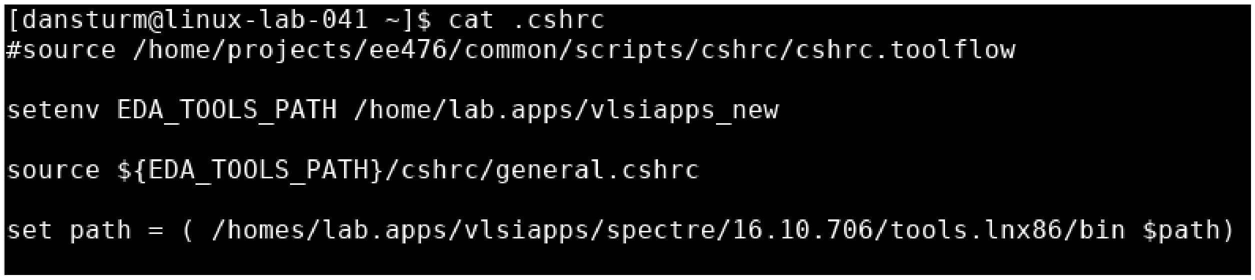
Before starting Cadence, we need to appropriately configure a working directory and local environment variables. First and foremost, make sure your login shell is tcsh by entering "echo $0" (the response should be "tsch"). If it is not, please go back to Lab 0 and complete this step.

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 tcsh shell. A .cshrc file is a file in your home directory ("/home/<netID>") that is automatically executed every time you open a terminal, and we will use this to instantiate the right environment variables. We will do this by including in the .cshrc file instructions to execute an existing .cshrc file. Perform the following steps:

1. Navigate to your home directory (cd ~).
2. Enter "vi .cshrc" into the terminal. If you already have a .cshrc file in this folder, this will open it. If you don't, this will create it. You can check if you already have a .cshrc file by typing "ls -a" into the terminal (note the "-a" – this is necessary for seeing hidden files). You may have a .cshrc file with some text in it if you are in another class that requires usage of Linux servers (for example EE 476)
3. If you already have a .cshrc file, you will see some lines of text in it. Comment out these lines by making the first character a "#" symbol (see photo below). **Important – if you are doing this, make sure to uncomment these lines at the end of this lab, and comment out the lines we are about to enter.** Review lab 0 for some help using the vim editor.
4. Enter the following lines into your .cshrc file:
   a. *setenv EDA_TOOLS_PATH /home/lab.apps/vlsiapps_new*
   b. *source ${EDA_TOOLS_PATH}/cshrc/general.cshrc*
   c. *set path = (/homes/lab.apps/vlsiapps/spectre/16.10.706/tools.lnx86/bin $path)*

Your ~/.cshrc file should now look like this, except perhaps without the first (commented out) line. (Also the netID should be different):



```
[dansturm@linux-lab-041 ~]$ cat .cshrc
#source /home/projects/ee476/common/scripts/cshrc/cshrc.toolflow

setenv EDA_TOOLS_PATH /home/lab.apps/vlsiapps_new

source ${EDA_TOOLS_PATH}/cshrc/general.cshrc

set path = ( /homes/lab.apps/vlsiapps/spectre/16.10.706/tools.lnx86/bin $path)
```

Run **"source .cshrc"** in the terminal to run the script (or, close and reopen terminal). Test that it worked by running **"which spectre"** and **"which virtuoso"**. Next, create and populate your cadence working directory with the following commands:

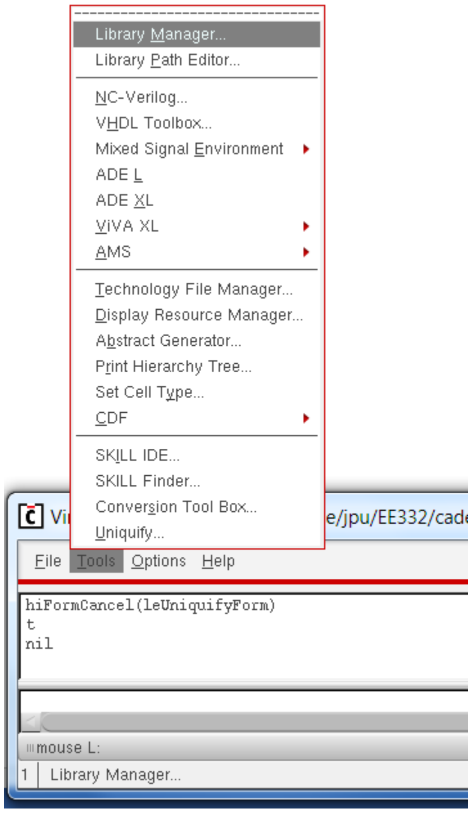
```
mkdir ~/EE332
mkdir ~/EE332/cadence
cp /home/lab.apps/vlsiapps/cshrc/EE332/.cdsinit  ~/EE332/cadence/.
cp /home/lab.apps/vlsiapps/cshrc/EE332/display.drf  ~/EE332/cadence/.
cp /home/lab.apps/vlsiapps/cshrc/EE332/.cdsenv  ~/EE332/cadence/.
cp /home/lab.apps/vlsiapps/cshrc/EE332/cds.lib  ~/EE332/cadence/.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

You can now run Cadence from the EE332/cadence directory just created. Navigate to this directory and start Cadence with the following commands (virtuoso is the command to start Cadence):

```
cd ~/EE332/cadence
virtuoso &
```

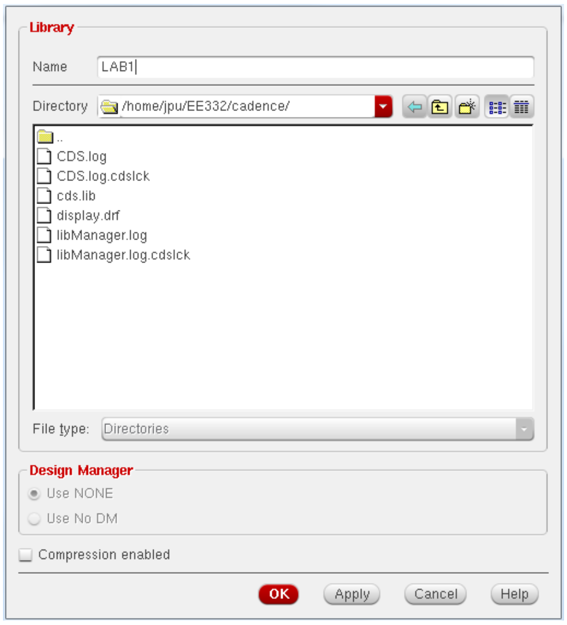
Cadence Library Setup

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:



```
hiFormCancel(leUniquifyForm)
t
nil
```

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



Library

Name: LAB1

Directory: /home/jpu/EE332/cadence/

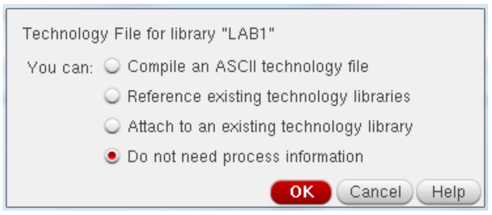
```
..
CDS.log
CDS.log.cdslck
cds.lib
display.drf
libManager.log
libManager.log.cdslck
```

File type: Directories

Design Manager: [✓ Use NONE] Use No DM

Compression enabled

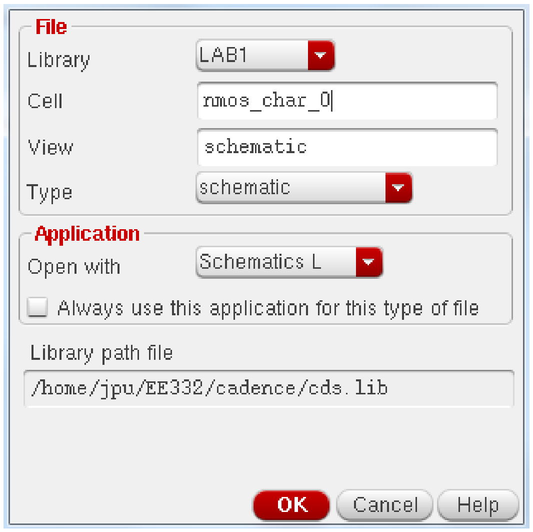
You will be asked to attach a technology file. Select "**Do not need process information**," as shown below:



Technology File for library "LAB1"

You can: Compile an ASCII technology file / Reference existing technology libraries / Attach to an existing technology library / [✓ Do not need process information]

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. Note that spaces and Linux file/directory names do not get along. Use underscores instead of spaces in all Linux names.


File

Library: LAB1

Cell: nmos_char_0

View: schematic

Type: schematic

Application

Open with: Schematics L

☐ Always use this application for this type of file

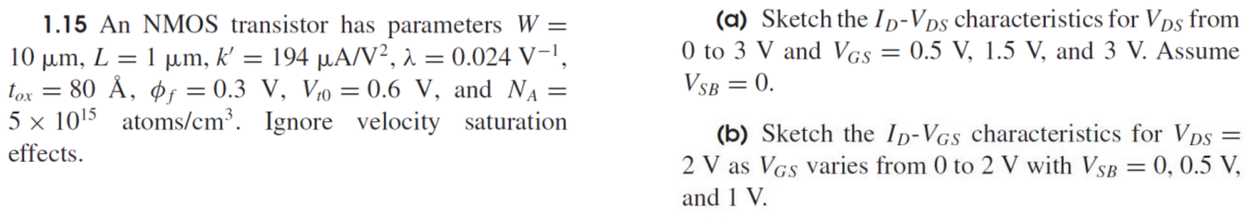
Library path file: /home/jpu/EE332/cadence/cds.lib

Setup is now complete. 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 $W = 10\ \mu\text{m}$, $L = 1\ \mu\text{m}$, $k' = 194\ \mu\text{A/V}^2$, $\lambda = 0.024\ \text{V}^{-1}$, $t_{ox} = 80$ Å, $\phi_f = 0.3$ V, $V_{t0} = 0.6$ V, and $N_A = 5 \times 10^{15}$ atoms/cm^3. Ignore velocity saturation effects.

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

(b) Sketch the I_D-V_{GS} characteristics for $V_{DS} = 2$ 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:

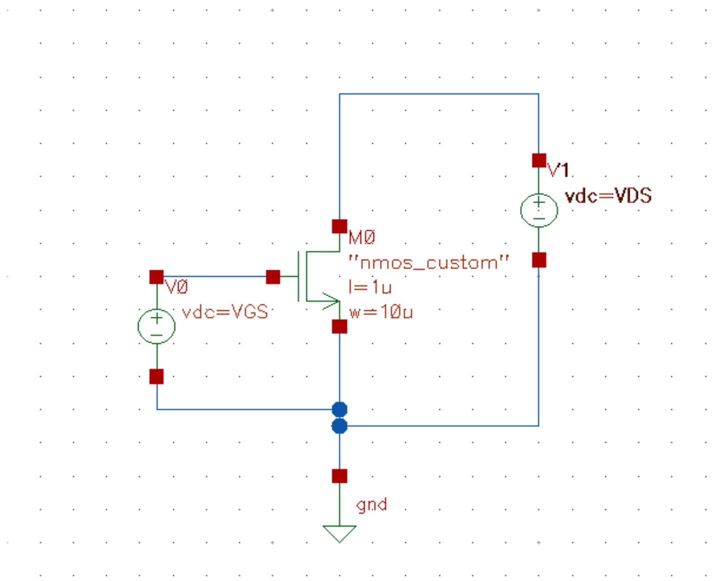
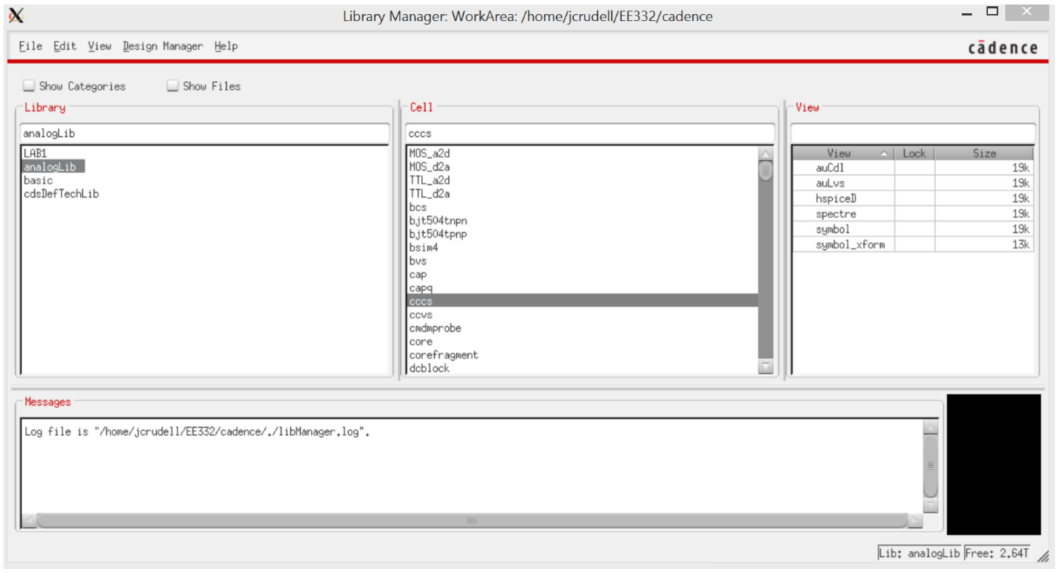


Figure 1: NMOS transistor schematic to simulation "I-V Characteristics".

The next step is to draw the schematic shown above. All the components in this schematic are found in another library which should appear in your "**Library Manager -> WorkArea**". The different libraries you have appear in the left column 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:



Library Manager: WorkArea: /home/jcrudell/EE332/cadence
File Edit View Design Manager Help
cādence
Show Categories
Show Files
Library
Cell
View

```
Library: analogLib
LAB1
analogLib
basic
cdsDefTechLib
```

```
Cell: cccs
MOS_a2d
MOS_d2a
TTL_a2d
TTL_d2a
bcs
bjt504tnpn
bjt504tpnp
bsim4
bvs
cap
capq
cccs
ccvs
cmdmprobe
core
corefragment
dcblock
```

View	Lock	Size
auCdl		19k
auLvs		19k
hspiceD		19k
spectre		19k
symbol		19k
symbol_xform		13k

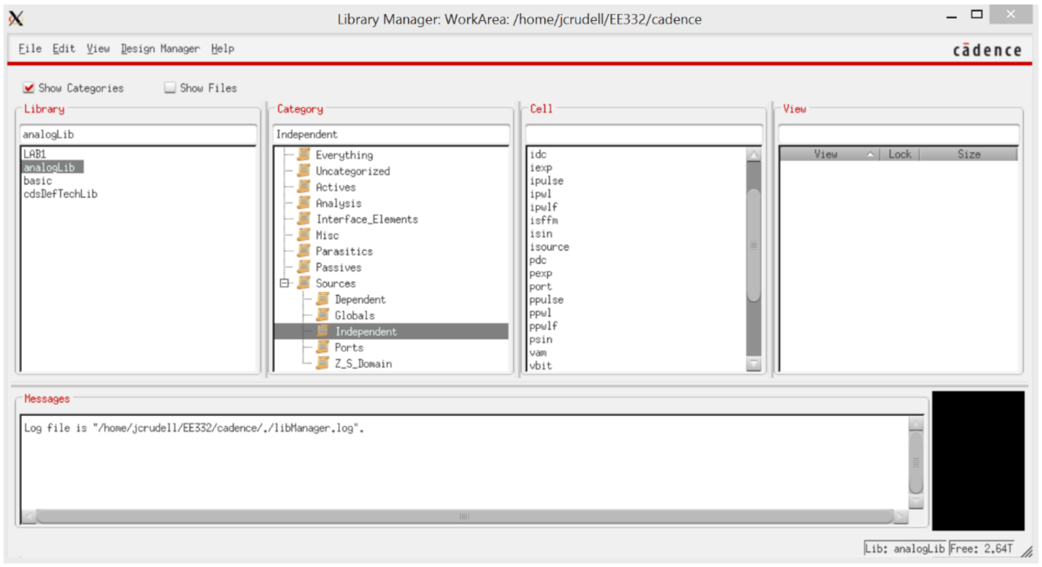
Messages

```
Log file is "/home/jcrudell/EE332/cadence/./libManager.log".
```

Lib: analogLib Free: 2.64T

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 left, 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).


Library Manager: WorkArea: /home/jcrudell/EE332/cadence
File Edit View Design Manager Help
cādence
Show Categories
Show Files
Library
Category
Cell
View
Lock
Size
Messages

Library: analogLib
- LAB1
- analogLib
- basic
- cdsDefTechLib

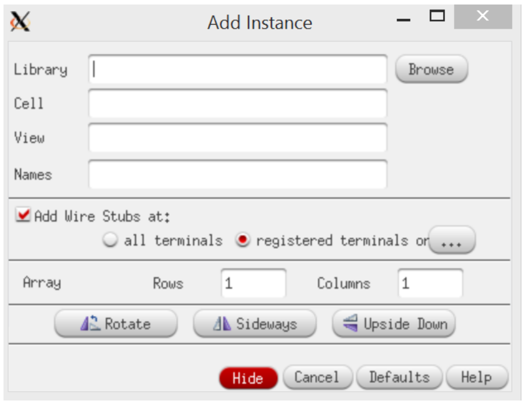
Category: Independent
- Everything
- Uncategorized
- Actives
- Analysis
- Interface_Elements
- Misc
- Parasitics
- Passives
- Sources
  - Dependent
  - Globals
  - Independent
  - Ports
  - Z_S_Domain

Cell:
- idc
- iexp
- ipulse
- ipwl
- ipwlf
- isffm
- isin
- isource
- pdc
- pexp
- port
- ppulse
- ppwl
- ppwlf
- psin
- vam
- vbit

```
Log file is "/home/jcrudell/EE332/cadence/./libManager.log".
```

Lib: analogLib Free: 2.64T

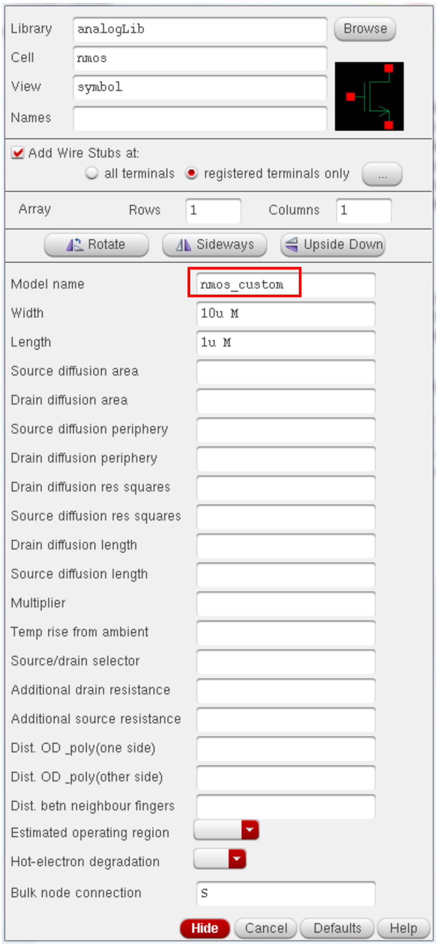
Now that you all have graduated to the level of Cadence Librarians, we can move on to build the schematic in **figure 1**. All the components in **figure 1** come from the **analogLib** library. Go back to your schematic named, "**nmos_char_0**". To place components, or "instantiate" them, there are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



Add Wire Stubs at: [✓] — [ ] all terminals [✓ registered terminals]

Array: Rows 1, Columns 1

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]

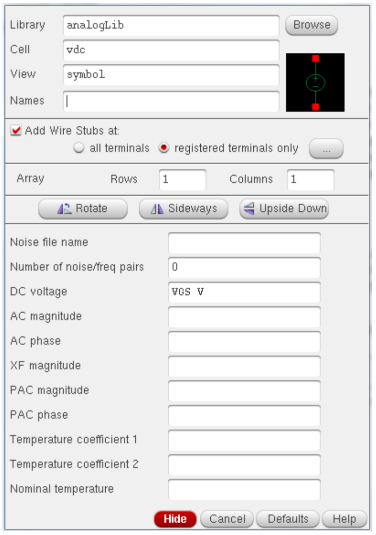
Array	Rows 1	Columns 1	

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.

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	

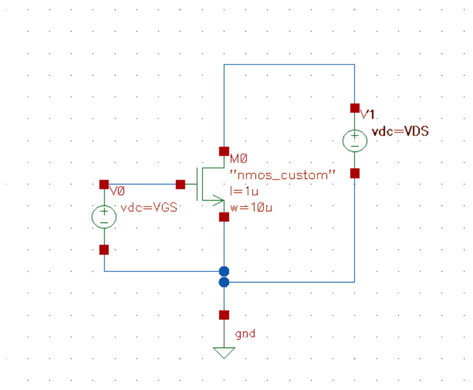
Before placing the "**vdc**" components, add parameters for the DC output voltages in the model under the entry "**DC voltage**". This can be a number or a variable. We will use variables "**VGS**" and "**VDS**" for the voltage source placed at the gate and drain of the device, respectively. To add a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

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.

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 (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,

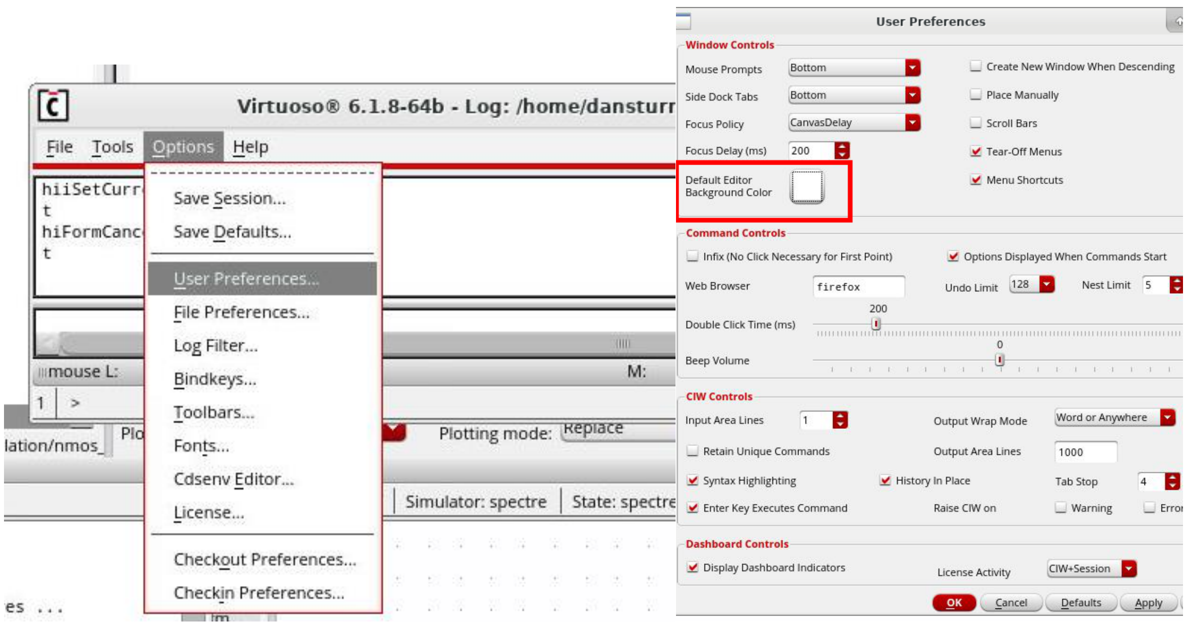


.

"**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.

Please take a picture of your schematic and include it in your lab report (take a screenshot from the computer you are using to access the server). In order to make sure all components and wires are easily visible, please make the background color of the schematic white. This can be done by going to the main Virtuoso window (the one that first opened; that we used to open the library manager), going to **options->User Preferences**, and then changing the color under "window controls".



```
hiiSetCurr
t
hiFormCanc
t
```

Window Controls
- Mouse Prompts: Bottom
- Side Dock Tabs: Bottom
- Focus Policy: CanvasDelay
- Focus Delay (ms): 200
- Default Editor Background Color: ⬜
- ☐ Create New Window When Descending
- ☐ Place Manually
- ☐ Scroll Bars
- ✓ Tear-Off Menus
- ✓ Menu Shortcuts

Command Controls
- ☐ Infix (No Click Necessary for First Point)
- ✓ Options Displayed When Commands Start
- Web Browser: firefox
- Undo Limit: 128
- Nest Limit: 5
- Double Click Time (ms): 200
- Beep Volume: 0

CIW Controls
- Input Area Lines: 1
- Output Wrap Mode: Word or Anywhere
- ☐ Retain Unique Commands
- Output Area Lines: 1000
- ✓ Syntax Highlighting
- ✓ History In Place
- Tab Stop: 4
- ✓ Enter Key Executes Command
- Raise CIW on: ☐ Warning ☐ Error

Dashboard Controls
- ✓ Display Dashboard Indicators
- License Activity: CIW+Session

Running a Simulation of the Schematic

OK, time to simulate your schematic!

The next step is to make a custom model file for the NMOS device that includes $k', \lambda, t_{ox}, \phi_f, V_{t0}$, 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:	**model** *mname* **mos1 type=** [**n** | **p**] [param= *value*]*

Create a file "**nmos_custom.scs**" in your Cadence working directory with your favorite Linux text editor.

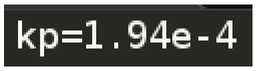
For those of your unfamiliar with Linux, you can create and open a text file with a GUI editor with the following command:

```
gedit nmos_custom.scs &
```

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

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

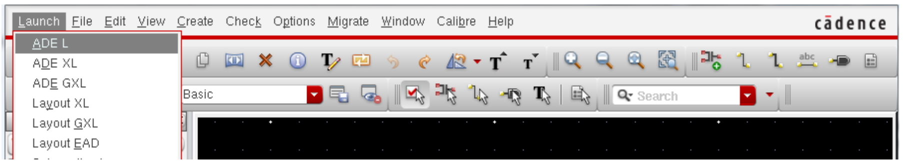
Input the parameters given in the problem statement in scientific format. For example, $k' = 194\mu A/V^2$ would look like this:



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

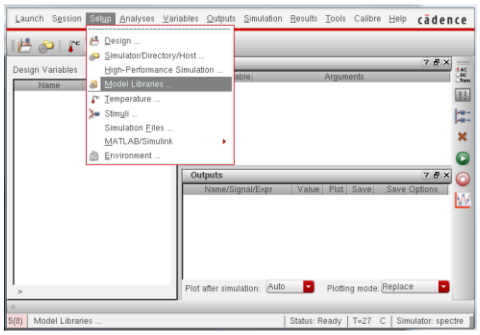
Fill in all the device parameters from problem **1.15 of G&M**. Remember that phi is ϕ (problem statement) and nsub is N_A. Finish off your model file and return to the Cadence environment.

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:



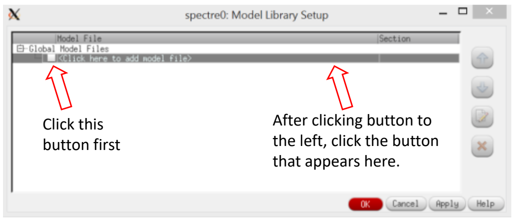
Launch File Edit View Create Check Options Migrate Window Calibre Help
cādence
ADE L
ADE XL
ADE GXL
Layout XL
Layout GXL
Layout EAD
Basic
Search

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.



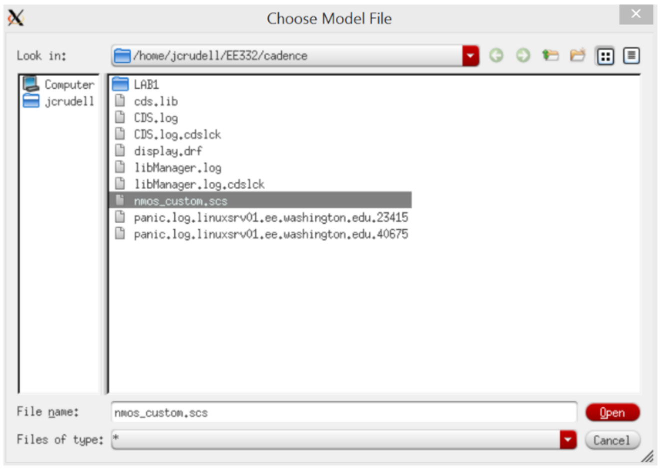
Launch Session Setup Analyses Variables Outputs Simulation Results Tools Calibre Help
cādence
Design ...
Simulator/Directory/Host ...
High-Performance Simulation ...
Model Libraries ...
Temperature ...
Stimuli ...
Simulation Files ...
MATLAB/Simulink
Environment ...
Design Variables
Name
Arguments
Outputs
Name/Signal/Expr Value Plot Save Save Options
Plot after simulation: Auto
Plotting mode: Replace
Model Libraries ...
Status: Ready T=27 C Simulator: spectre

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



spectre0: Model Library Setup
Model File
Section
Global Model Files
<Click here to add model file>
Click this button first
After clicking button to the left, click the button that appears here.
OK
Cancel
Apply
Help

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



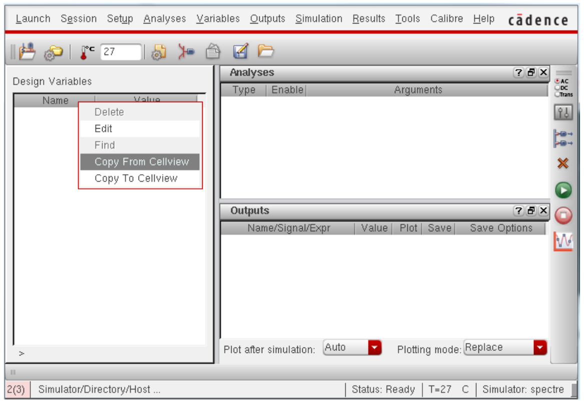
Look in: /home/jcrudell/EE332/cadence

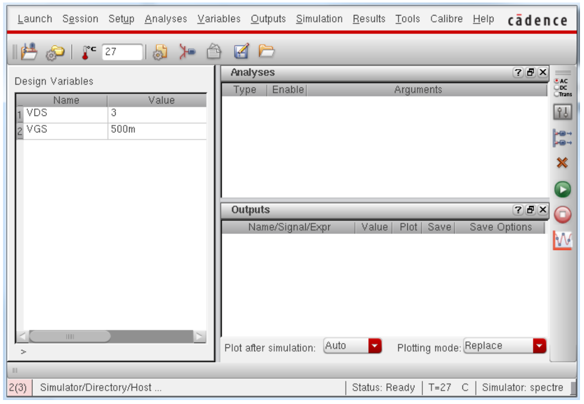
```
LAB1
cds.lib
CDS.log
CDS.log.cdslck
display.drf
libManager.log
libManager.log.cdslck
nmos_custom.scs
panic.log.linuxsrv01.ee.washington.edu.23415
panic.log.linuxsrv01.ee.washington.edu.40675
```

File name: nmos_custom.scs

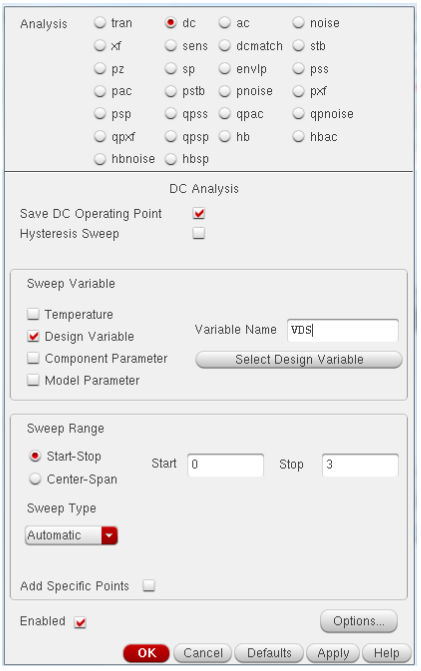
Files of type: *

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.


Launch Session Setup Analyses Variables Outputs Simulation Results Tools Calibre Help cādence
Design Variables
Name Value
Delete
Edit
Find
Copy From Cellview
Copy To Cellview
Analyses
Type Enable Arguments
Outputs
Name/Signal/Expr Value Plot Save Save Options
Plot after simulation: Auto
Plotting mode: Replace
Status: Ready T=27 C Simulator: spectre


Launch Session Setup Analyses Variables Outputs Simulation Results Tools Calibre Help cādence
Design Variables
Analyses
Type Enable Arguments
Outputs
Name/Signal/Expr Value Plot Save Save Options
Plot after simulation: Auto
Plotting mode: Replace
Status: Ready T=27 C Simulator: spectre

	Name	Value
1	VDS	3
2	VGS	500m



DC Analysis

Save DC Operating Point ✓
Hysteresis Sweep

Sweep Variable

Temperature
✓ Design Variable
Component Parameter
Model Parameter

Variable Name VDS

Select Design Variable

Sweep Range

● Start-Stop
Center-Span

Start 0 Stop 3

Sweep Type Automatic

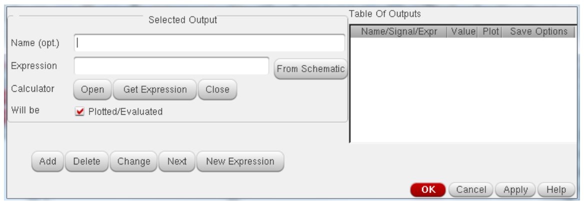
Add Specific Points

Enabled ✓

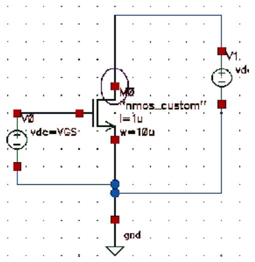
Now that the models and variables are properly setup, we are ready to pick the type of simulation that we'd like to perform. To do this, Go to the "**Analyses**" menu at the top of the ADE window, hit "**Choose**" and the menu shown to the left will pop up. Open Analyses: Choose and select DC analysis. 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 to plot and store results from the simulation.

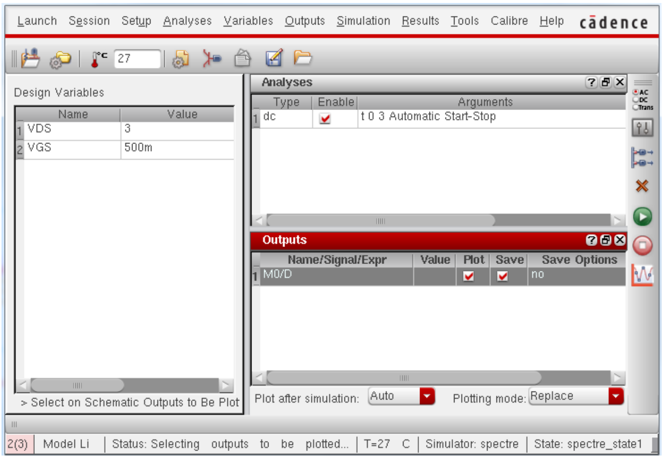
Then, click the drain terminal of the nmos device:



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:



Design Variables

	Name	Value
1	VDS	3
2	VGS	500m

Analyses

	Type	Enable	Arguments
1	dc	✓	t 0 3 Automatic Start-Stop

Outputs

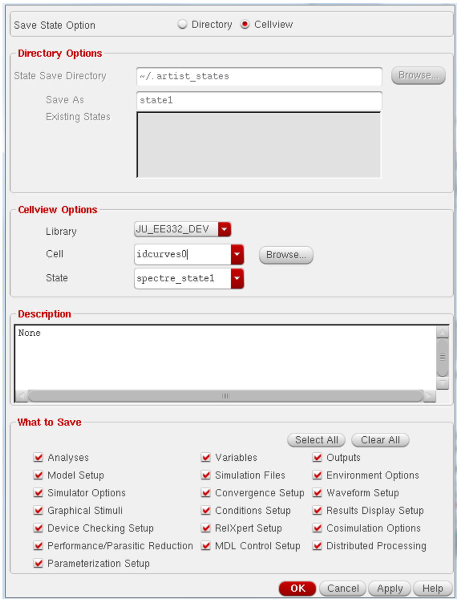
	Name/Signal/Expr	Value	Plot	Save	Save Options
1	M0/D		✓	✓	no

> Select on Schematic Outputs to Be Plot

Plot after simulation: Auto   Plotting mode: Replace

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.



Save State Option ( ) Directory [✓ Cellview]

Directory Options

State Save Directory	~/.artist_states
Save As	state1
Existing States	

Cellview Options

Library	JU_EE332_DEV
Cell	idcurves0
State	spectre_state1

Description

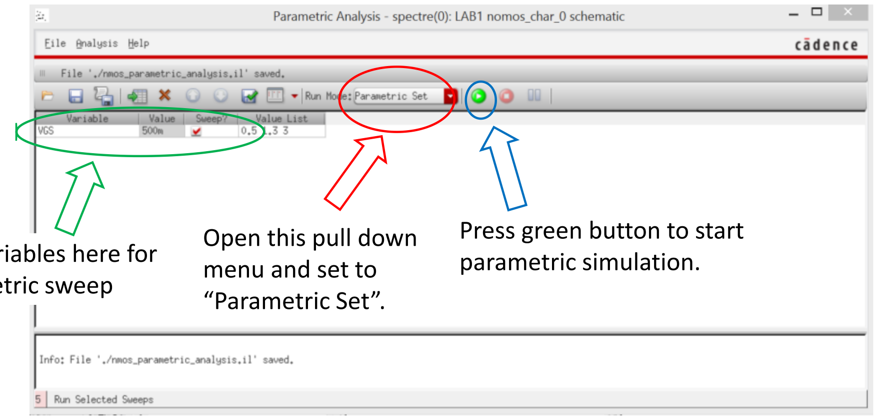
None

What to Save

✓ Analyses	✓ Variables	✓ Outputs
✓ Model Setup	✓ Simulation Files	✓ Environment Options
✓ Simulator Options	✓ Convergence Setup	✓ Waveform Setup
✓ Graphical Stimuli	✓ Conditions Setup	✓ Results Display Setup
✓ Device Checking Setup	✓ RelXpert Setup	✓ Cosimulation Options
✓ Performance/Parasitic Reduction	✓ MDL Control Setup	✓ Distributed Processing
✓ Parameterization Setup		

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.



Parametric Analysis - spectre(0): LAB1 nomos_char_0 schematic

File './nmos_parametric_analysis.il' saved.

Run Mode: Parametric Set

Variable	Value	Sweep?	Value List
VGS	500m	✓	0.5 1.3 3

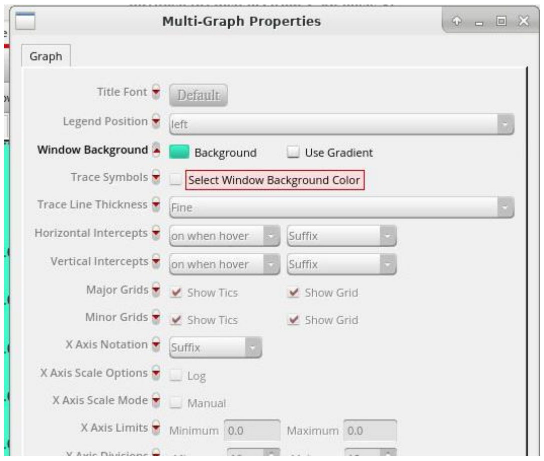
Info: File './nmos_parametric_analysis.il' saved.

5 Run Selected Sweeps

File Analysis Help
cādence
Add variables here for parametric sweep
Open this pull down menu and set to "Parametric Set".
Press green button to start parametric simulation.

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. **Save a plot of your results for submission.** For this week, you can do this by taking a screenshot from the computer you use to access the server – in future weeks we will ask you to generate a plot using Cadence and then transfer it to your computer. Please make the background of the plots white to make sure all traces are easily visible. This can be done in the **Edit -> Multi-Graph**

Properties menu, as shown below (make sure to click on the arrow pointing down to turn it up, then click on the color displayed). Note that graph background color cannot be edited from **Graph -> Properties**.


Multi-Graph Properties

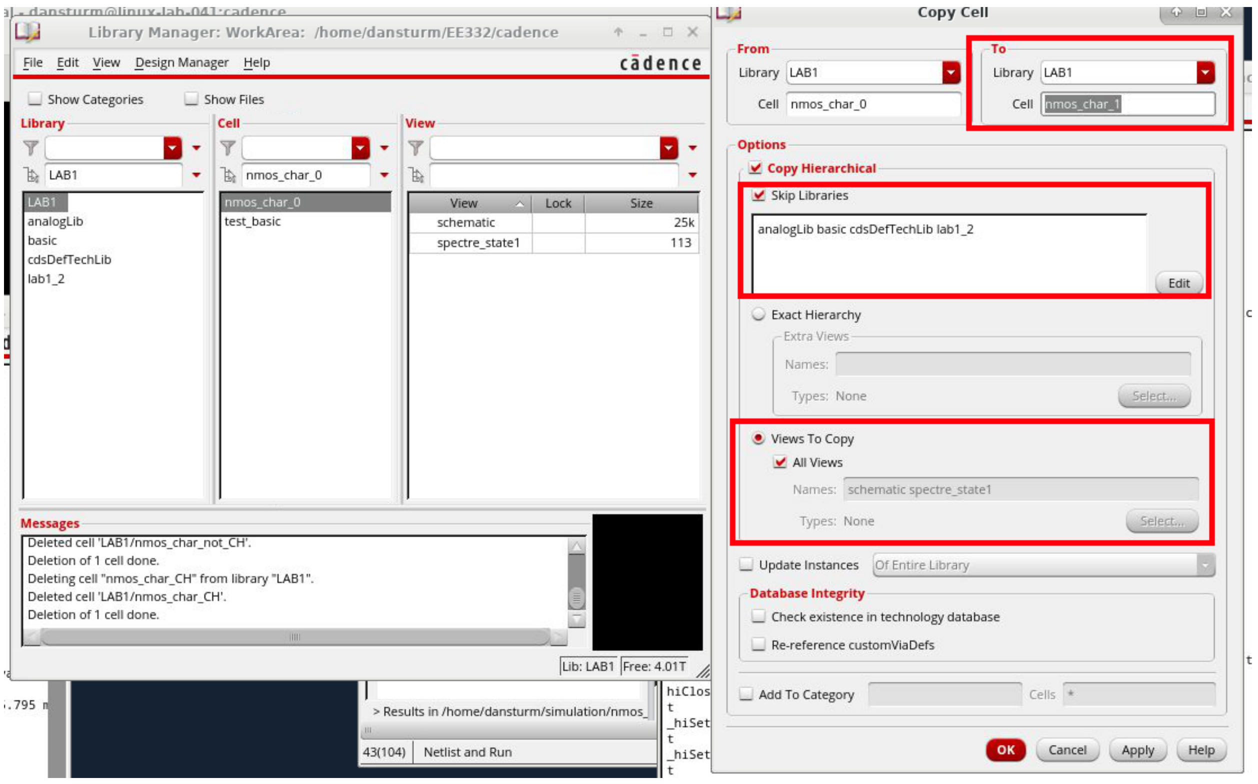
Graph

Title Font	Default	
Legend Position	left	
Window Background	🟩 Background	☐ Use Gradient
Trace Symbols	☐ Select Window Background Color	
Trace Line Thickness	Fine	
Horizontal Intercepts	on when hover	Suffix
Vertical Intercepts	on when hover	Suffix
Major Grids	✓ Show Tics	✓ Show Grid
Minor Grids	✓ Show Tics	✓ Show Grid
X Axis Notation	Suffix	
X Axis Scale Options	☐ Log	
X Axis Scale Mode	☐ Manual	
X Axis Limits	Minimum 0.0	Maximum 0.0

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, as shown below. This replicates the schematic and ADE setup defined for the previous section. Make sure that your cellview for the first part of the lab (probably “nmos_char_0”) is selected and not one of the views, so that the entire cellview is selected (and not just one of the views). Make sure to select “skip libraries” and the rest of the configurations shown below. Give your new cell a new name. You’ll see that both the schematic and the saved ADE configuration are copied. After the copy is complete, make sure the views of the new cell match those of the old cell.


Library Manager: WorkArea: /home/dansturm/EE332/cadence

File Edit View Design Manager Help

☐ Show Categories ☐ Show Files

Library	Cell	View	Lock	Size
LAB1	nmos_char_0	schematic		25k
analogLib	test_basic	spectre_state1		113
basic				
cdsDefTechLib				
lab1_2				

Messages
```
Deleted cell 'LAB1/nmos_char_not_CH'.
Deletion of 1 cell done.
Deleting cell "nmos_char_CH" from library "LAB1".
Deleted cell 'LAB1/nmos_char_CH'.
Deletion of 1 cell done.
```
Lib: LAB1 Free: 4.01T

> Results in /home/dansturm/simulation/nmos_

43(104) Netlist and Run

Copy Cell

From: Library LAB1, Cell nmos_char_0

To: Library LAB1, Cell nmos_char_1

Options

☑ Copy Hierarchical

☑ Skip Libraries: analogLib basic cdsDefTechLib lab1_2 (Edit)

○ Exact Hierarchy — Extra Views: Names: — Types: None (Select...)

● Views To Copy — ☑ All Views — Names: schematic spectre_state1 — Types: None (Select...)

☐ Update Instances: Of Entire Library

Database Integrity

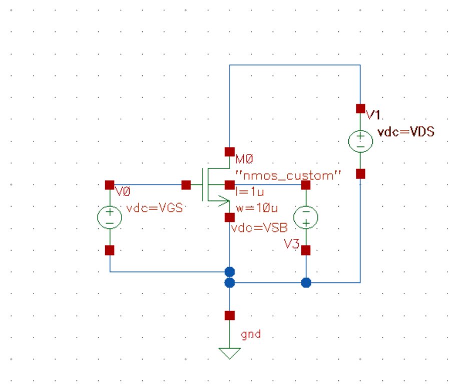
☐ Check existence in technology database

☐ Re-reference customViaDefs

☐ Add To Category — Cells: *

OK Cancel Apply Help

Open the new schematic and remove the NMOS device. Replace it with a nmos4 device from the analogLib library, with the same parameters as the NMOS device before. 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 schematic should now look like the example below:



Once finished, "**Netlist and Save**" the schematic, and again **please take a picture of your schematic (with white background) and include it in your lab report**. 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 (the saved ADE configuration in the new cellview will still "point" to the schematic in the old cellview, because it is a direct copy of the saved ADE configuration in the old cellview). Open **Setup -> Design** and select the new schematic, or your simulations will not include the body bias modification. Save the simulation settings so that Cadence remembers this change.

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

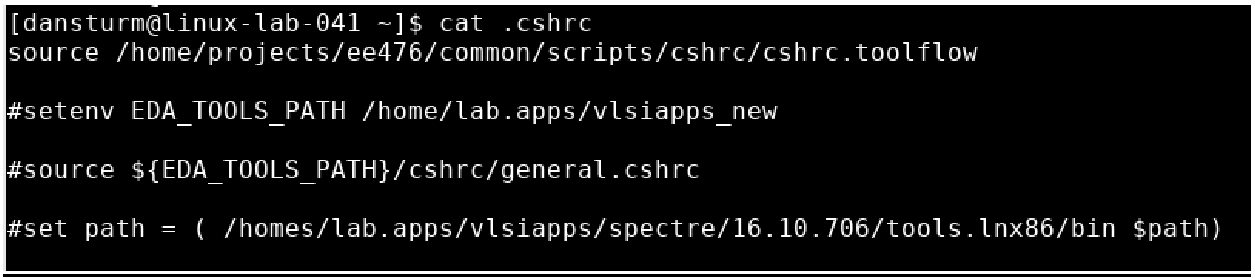
Now, open the Parametric Analysis window again, and set up a sweep for **VSB = 0V, 0.5V, 1V**. Run this parametric sweep to complete Part b.

Make sure to save your plot and simulation setup! Again, **please take a picture of your plot (with white background) and include it in your lab report.**

Deliverables

Please show screenshots of schematics and results generated from both problems and add 2 or 3 sentences for each problem to describe the results. Explain why you see these results, based on device physics and the voltage applied. If you find the results surprising, given the device physics/voltages, explain why. **Before the next lab session**, please upload it through canvas.

Important – if your .cshrc file has commands in it other than the ones added in this lab, make sure that after you are done with this lab you uncomment the pre-existing commands and comment out the ones from today's lab. For example:



```
[dansturm@linux-lab-041 ~]$ cat .cshrc
source /home/projects/ee476/common/scripts/cshrc/cshrc.toolflow

#setenv EDA_TOOLS_PATH /home/lab.apps/vlsiapps_new

#source ${EDA_TOOLS_PATH}/cshrc/general.cshrc

#set path = ( /homes/lab.apps/vlsiapps/spectre/16.10.706/tools.lnx86/bin $path)
```