Analog Labs Manual

Revision 1.0

IC614

ASSURA410

MMSIM 101

Developed By

University Support Team

Cadence Design Systems, Bangalore

Objective

Objective of this lab is to learn the Virtuoso tool as well learn the flow of the Full Custom IC design cycle. You will finish the lab by running DRC, LVS and Parasitic Extraction on the various designs. In the process you will create various components like inverter, NAND gate, XOR gate, Full adder, Latch, SRAM register cell, grey to binary code converter and PLL, differential amplifier we won't be designing every cell, as the time will not be sufficient, instead we will be using some ready made cells in the process.

You will start the lab by creating a library called "**myDesignLib**" and you will attach the library to a technology library called "**gpdk180**". Attaching a technology library will ensure that you can do front to back design.

You will create a new cell called "**Inverter**" with schematic view and hence build the inverter schematic by instantiating various components. Once inverter schematic is done, symbol for "**Inverter**" is generated. Now you will create a new cell view called "**Inverter_Test**", where you will instantiate "**Inverter**" symbol. This circuit is verified by doing various simulations using spectre. In the process, you will learn to use spectre, waveform window options, waveform calculator, etc...

You will learn the Layout Editor basics by concentrating on designing an "**Inverter**" through automatic layout generation. Then you will go ahead with completing the other layouts. After that, you will run DRC, LVS checks on the layout, Extract parasitics and back-annotate them to the simulation environment.

After completing the parasitic back- annotation flow, design is ready for generating GDSII.

Table of Contents

General Notes

There are a number of things to consider before beginning these lab exercises. Please read through this section completely, and perform any needed steps in order to ensure a successful workshop. These labs were designed for use with Incisive Unified Simulator82, IC613 and Assura32.

Before running any of these labs, ensure that you've set up IUS92, IC614, MMSIM101 and Assura41 correctly:

%> setenv CDSHOME <*IC614-installation-home*>

%> setenv MMSIMHOME <MMSIM101-installation-home>

%> setenv PVHOME <Assura41-*installation-home*>

%> setenv AMSHOME <*IUS92-installation-home*>

You will also need to ensure that the IUS92 is setup correctly for lab 5.

To setup the lab environment, please perform the following steps:

1. Ensure the software mentioned above is correctly setup.
2. Source the C-Shell related commands file i.e. (cshrc file).

These labs were designed to be run using Cadence Virtuoso tool and Assura tool.

Lab Getting Started

1. Log in to your workstation using the username and password.The home directory has a **cshrc** file with paths to the Cadence installation.

2. In a terminal window, type **csh** at the command prompt to invoke the C shell.

>csh

>source cshrc

3. To verify that the path to the software is properly set in the **cshrc** file, type the below command in the terminal window and enter:

>which virtuoso

It gives the complete path of IC614 tool Installation.

>**which spectre**

It gives the complete path of MMSIM101 tool Installation.

>**which assura**

It gives the complete path of Assura410 tool Installation.

Starting the Cadence Software

Use the installed database to do your work and the steps are as follows:

1. Change to the course directory by entering this command:

> **cd ~/Database/cadence_analog_labs_613**

You will start the Cadence Design Framework II environment from this directory because it contains cds.lib, which is the local initialization file. The library search paths are defined in this file.

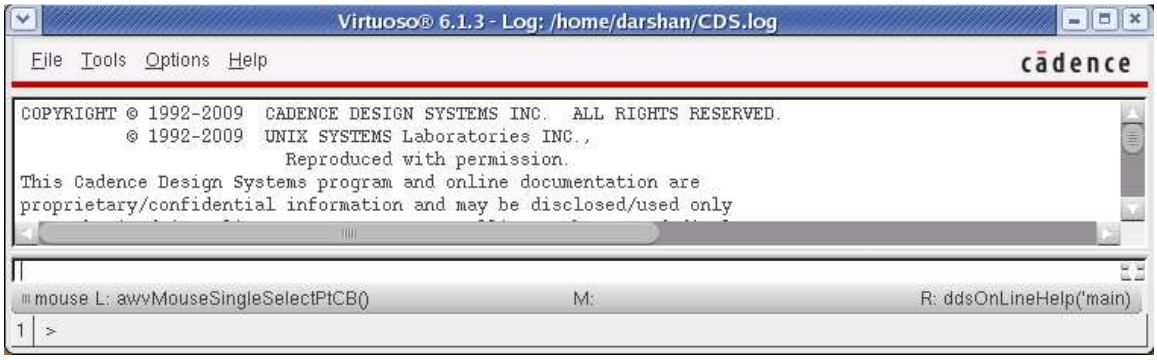
The **Cadence_Analog_labs_613** directory contains Solutions folder and also Work folder. Inside Work folder you can create new cell **/** modifications of the cell locally without affecting your Source cell present inside Solutions directory.

Directory	Directory
. /Solutions	Contains a local copy of all the lab experiments including test circuit for simulation.
. /libs.cdb	Contains a technology library for the design (gpdk180nm).
. /models	Contains spectre models of components for simulation in gpdk180nm technology.
. /stream	Contains layer map file for GDSII format
. /pv	Containing the Assura and Diva verification files
. /techfiles	Contains ASCII versions of the oa22 techfiles
. /dig_source	Contains verilog codes for SAR register and clock
. /cds.lib	File containing pointer to the Cadence OA22 initialization file**.**
. /hdl.var	File defines the work library for AMS simulation
. /docs	Reference manual and user manual for gpdk180nm technology.

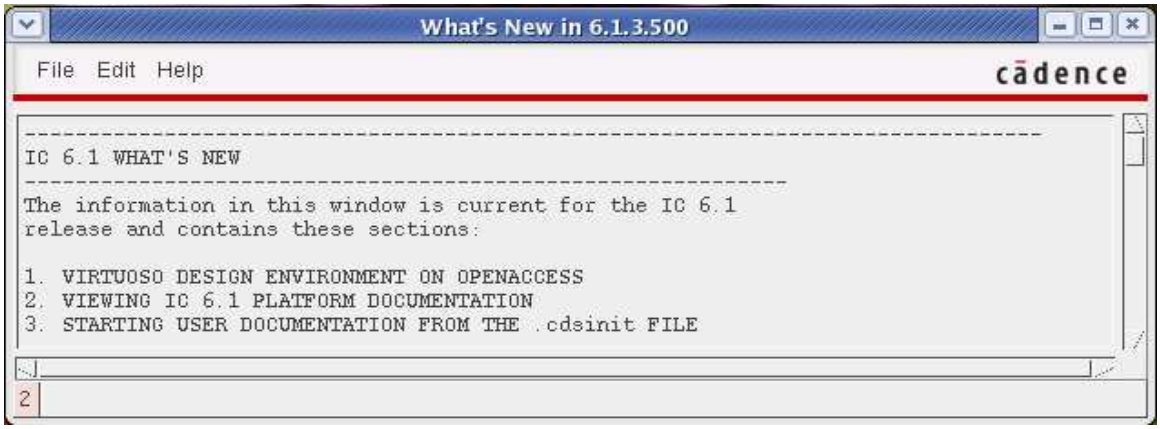
2. In the same terminal window, enter:

> **virtuoso &**

The virtuoso or Command Interpreter Window (CIW) appears at the bottom of the screen.



3. If the "What's New ..." window appears, close it with the **File— Close** command.



4. Keep opened CIW window for the labs.

End of General Notes

Lab: 1 Layout Design Rules

- The physical mask layout of any circuit to be manufactured using a particular process must conform to a set of geometric constraints or rules, which are generally called ***layout design rules***.

- These rules usually specify the minimum allowable line widths for physical objects on-chip such as metal and polysilicon interconnects or diffusion areas, minimum feature dimensions, and minimum allowable separations between two such features.

- ***The main objective of design rules*** is to achieve a high overall yield and reliability while using the smallest possible silicon area, for any circuit to be manufactured with a particular process.

- The layout design rules which are specified for a particular fabrication process normally represent a reasonable optimum point in terms of yield and density.

- A layout which violates some of the specified design rules may still result in an operational circuit with reasonable yield, whereas another layout observing all specified design rules may result in a circuit which is not functional and/or has very low yield.

- To summarize, we can say, in general, that observing the layout design rules significantly increases the probability of fabricating a successful product with high yield.

- The design rules are usually described in two ways :

  - ✓Micron rules, in which the layout constraints such as minimum feature sizes and minimum allowable feature separations, are stated in terms of absolute dimensions in micrometers, or,

  - ✓Lambda rules, which specify the layout constraints in terms of a single parameter (?) and, thus, allow linear, proportional scaling of all geometrical constraints.

Lambda-based layout design rules were originally devised to simplify the industry-standard micron-based design rules and to allow scaling capability for various processes. It must be emphasized, however, that most of the submicron CMOS process design rules do not lend themselves to straightforward linear scaling. The use of lambda-based design rules must therefore be handled with caution in sub-micron

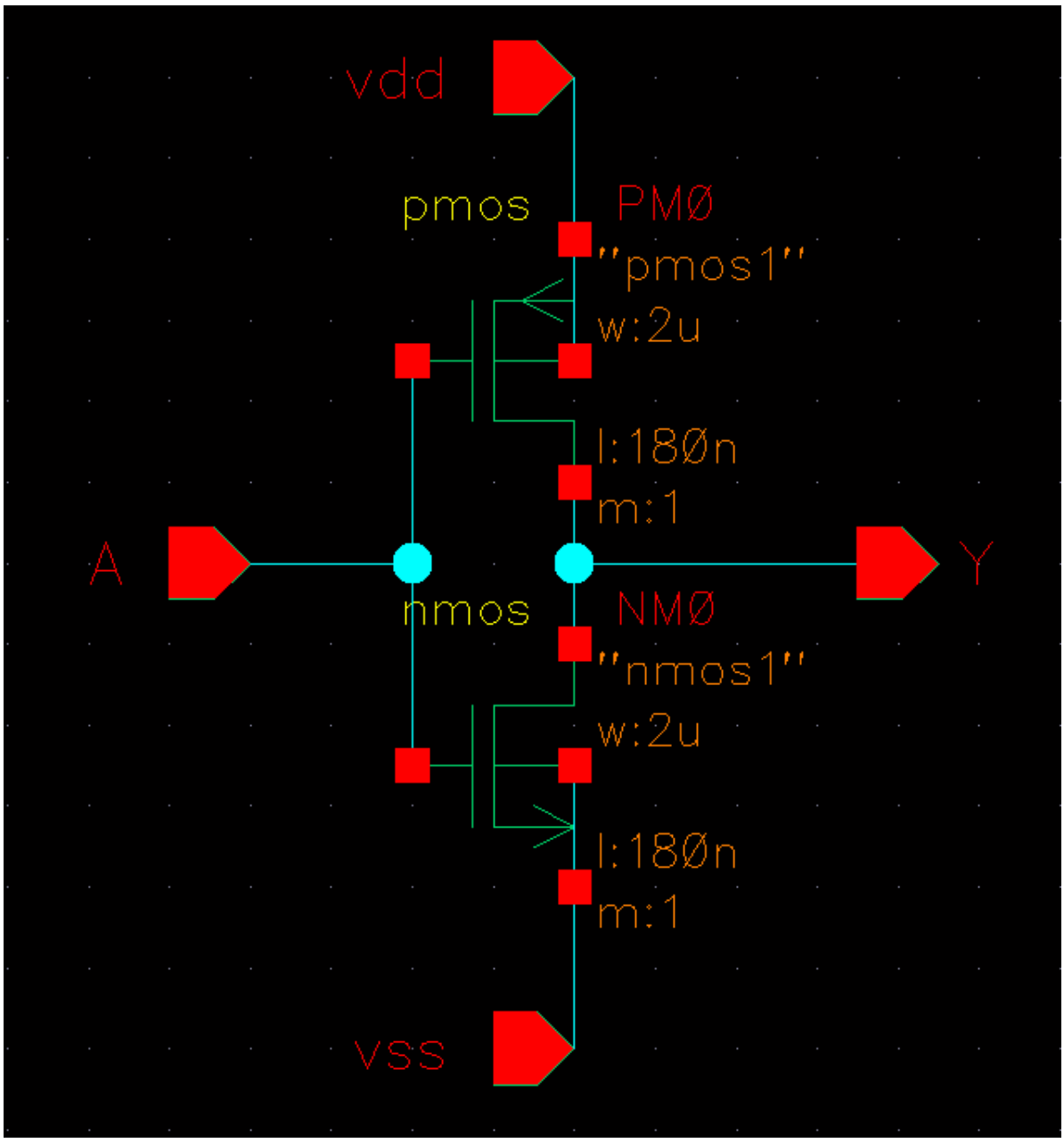
geometries. In the following, we present a sample set of the lambda-based layout design rules devised for the MOSIS CMOS process.

MOSIS Layout Design Rules (sample set)

Rule number	Description	L-Rule
R1	Minimum active area width	3 L
R2	Minimum active area spacing	3 L
R3	Minimum poly width	2 L
R4	Minimum poly spacing	2 L
R5	Minimum gate extension of poly over active	2 L
R6	Minimum poly-active edge spacing (poly outside active area)	1 L
R7	Minimum poly-active edge spacing (poly inside active area)	3 L
R8	Minimum metal width	3 L
R9	Minimum metal spacing	3 L
R10	Poly contact size 2 L	
R11	Minimum poly contact spacing	2 L
R12	Minimum poly contact to poly edge spacing	1 L
R13	Minimum poly contact to metal edge spacing	1 L
R14	Minimum poly contact to active edge spacing	3 L
R15	Active contact size	2 L
R16	Minimum active contact spacing (on the same active region)	2 L
R17	Minimum active contact to active edge spacing	1 L
R18	Minimum active contact to metal edge spacing	1 L
R19	Minimum active contact to poly edge spacing	3 L
R20	Minimum active contact spacing (on different active regions)	6 L

Lab 2.1: AN INVERTER

Schematic Capture



Schematic Entry

Objective: To create a library and build a schematic of an Inverter

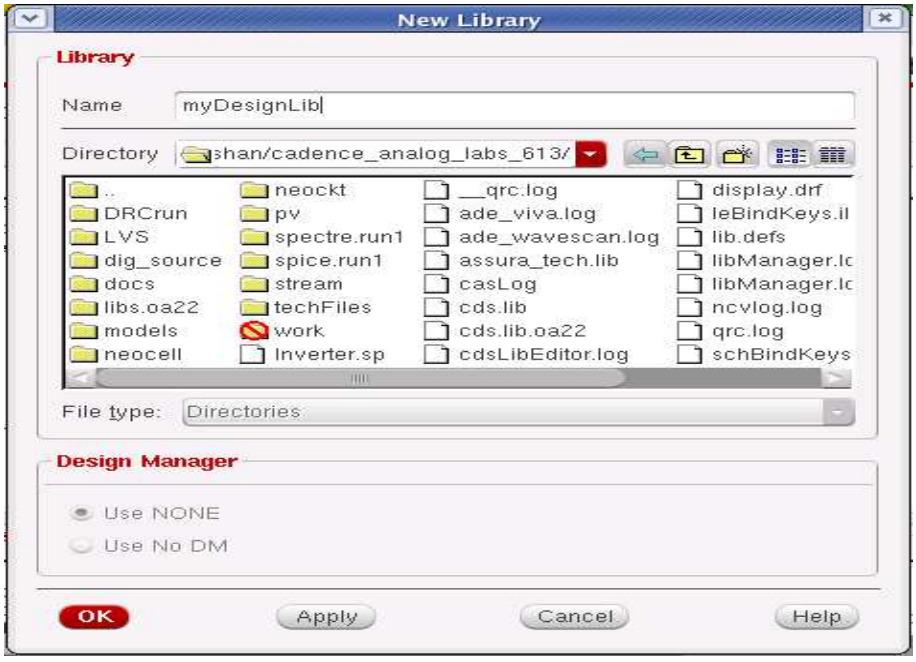
---

Below steps explain the creation of new library "**myDesignLib**" and we will use the same throughout this course for building various cells that we going to create in the next labs. Execute **Tools – Library Manager** in the CIW or Virtuoso window to open Library Manager.

Creating a New library

1. In the Library Manager, execute **File - New – Library**. The new library form appears.

2. In the "New Library" form, type "**myDesignLib**" in the Name section.



3. In the field of Directory section, verify that the path to the library is set to **~/Database/cadence_analog_labs_613** and click **OK**.

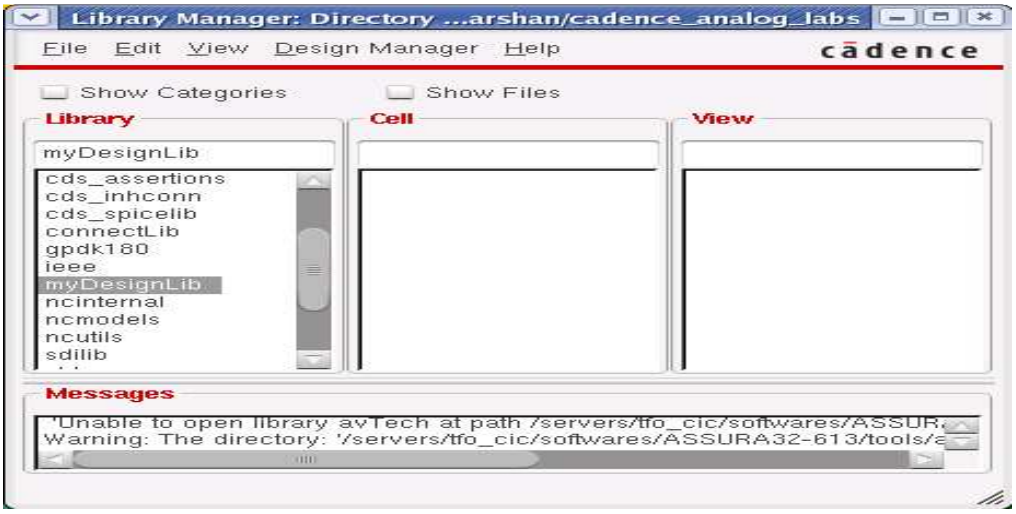
Note: A technology file is not required if you are not interested to do the layouts for the design.

4. In the next "**Technology File for New library**" form, select option **Attach to an existing techfile** and click **OK**.

5. In the "**Attach Design Library to Technology File**" form, select **gpdk180** from the cyclic field and click **OK**.

6. After creating a new library you can verify it from the library manager.

7. If you right click on the "**myDesignLib**" and select properties, you will find that **gpdk180** library is attached as techlib to "**myDesignLib**".

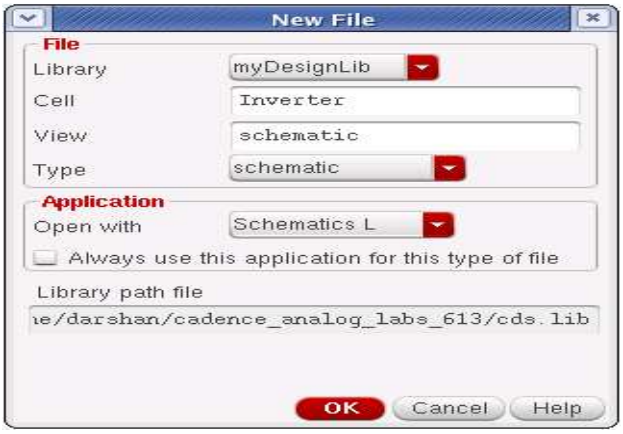


Creating a Schematic Cellview

In this section we will learn how to open new schematic window in the new **myDesignLib**" library and build the inverter schematic as shown in the figure at the start of this lab.

1. In the CIW or Library manager, execute **File – New – Cellview**.

2. Set up the New file form as follows:



Do not edit the **Library path file** and the one above might be different from the path shown in your form.

3. Click **OK** when done the above settings. A blank schematic window for the **Inverter** design appears.

Adding Components to schematic

1. In the Inverter schematic window, click the **Instance** fixed menu icon to display the Add Instance form. 

Tip: You can also execute **Create — Instance** or press **i**.

2. Click on the **Browse** button. This opens up a Library browser from which you can select components and the **symbol** view .

You will update the Library Name, Cell Name, and the property values given in the table on the next page as you place each component.

3. After you complete the Add Instance form, move your cursor to the schematic window and click **left** to place a component.

This is a table of components for building the Inverter schematic.

Library name	**Cell Name**	**Properties/Comments**
gpdk180	pmos	For M0: Model name = pmos1, **W= wp**, L=180n
gpdk180	nmos	For M1: Model name = nmos1, W= 2u, L=180n

If you place a component with the wrong parameter values, use the **Edit— Properties— Objects** command to change the parameters.

Use the **Edit— Move** command if you place components in the wrong location.

You can rotate components at the time you place them, or use the **Edit— Rotate** command after they are placed.

4. After entering components, click **Cancel** in the Add Instance form or press **Esc** with your cursor in the schematic window.

Adding pins to Schematic

1. Click the **Pin** fixed menu icon in the schematic window. You can also execute **create — Pin** or press ***p***. 

The Add pin form appears.

2. Type the following in the Add pin form in the exact order leaving space between the pin names.

Pin Names	**Direction**
vin	Input
vout	Output

Make sure that the direction field is set to **input/output/inputOutput** when placing the **input/output/inout** pins respectively and the Usage field is set to **schematic**.

3. Select **Cancel** from the Add – pin form after placing the pins. In the schematic window, execute **Window— Fit** or press the **f** bindkey. 

Adding Wires to a Schematic

Add wires to connect components and pins in the design.

1. Click the **Wire (narrow)** icon in the schematic window. 

You can also press the ***w*** key, or execute **Create — Wire (narrow).**

2. In the schematic window, click on a pin of one of your components as the first point for your wiring. A diamond shape appears over the starting point of this wire.

3. Follow the prompts at the bottom of the design window and click **left** on the destination point for your wire. A wire is routed between the source and destination points.

4. Complete the wiring as shown in figure and when done wiring press **ESC** key in the schematic window to cancel wiring.

Saving the Design

1. Click the ***Check and Save*** icon in the schematic editor window.

2. Observe the CIW output area for any errors.

Symbol Creation

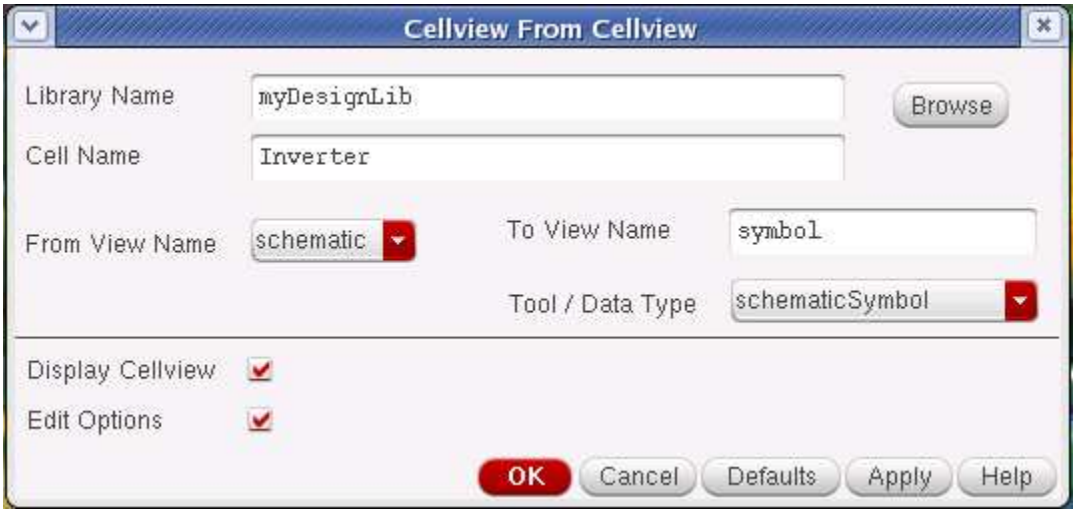
Objective: To create a symbol for the Inverter

In this section, you will create a symbol for your inverter design so you can place it in a test circuit for simulation. A symbol view is extremelyimportant step in the design process. The symbol view must exist for the schematic to be used in a hierarchy. In addition, the symbol has attached properties (cdsParam) that facilitate the simulation and the design of the circuit.

1. In the Inverter schematic window, execute **Create — Cellview— From Cellview**.

The **Cellview From Cellview** form appears. With the Edit Options function active, you can control the appearance of the symbol to generate.

2. Verify that the **From View Name** field is set to **schematic**, and the **To View Name** field is set to **symbol**, with the **Tool/Data Type** set as **SchematicSymbol**.



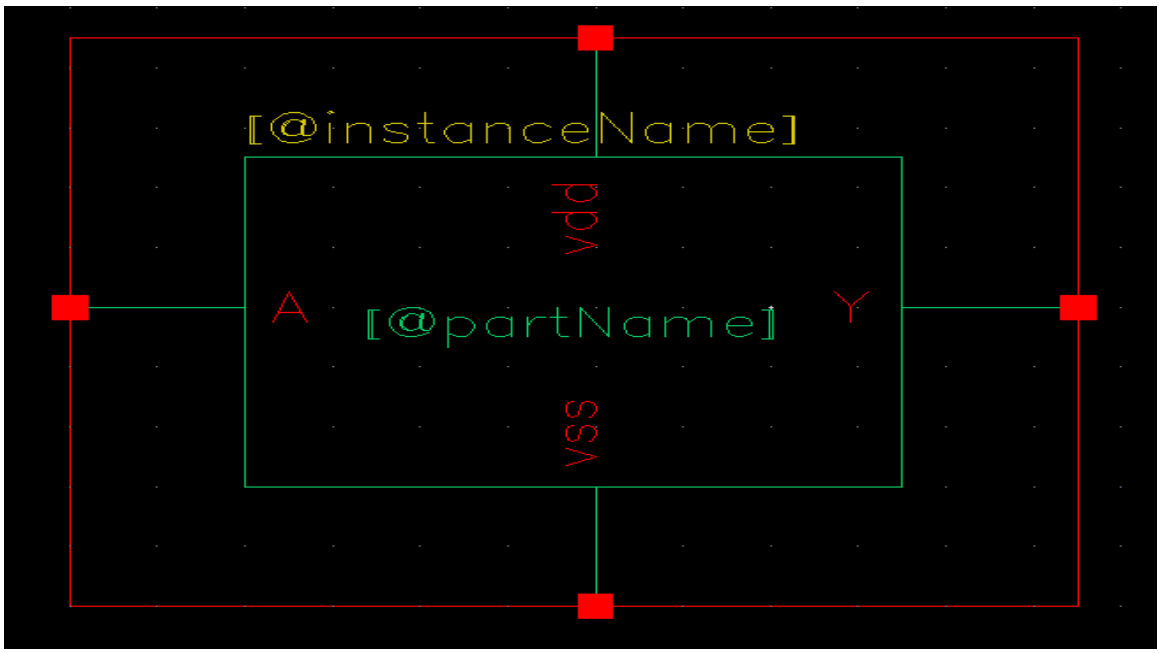
3. Click **OK** in the **Cellview From Cellview** form.

The Symbol Generation Form appears.

4. Modify the **Pin Specifications** as follows:

5. Click **OK** in the Symbol Generation Options form.

6. A new window displays an automatically created Inverter symbol as shown here.



Editing a Symbol

In this section we will modify the inverter symbol to look like a Inverter gate symbol.

1. Move the cursor over the automatically generated symbol, until the green rectangle is highlighted, click **left** to select it.

2. Click **Delete** icon in the symbol window, similarly select the red rectangle and delete that.

3. Execute **Create – Shape – polygon**, and draw a shape similar to triangle.

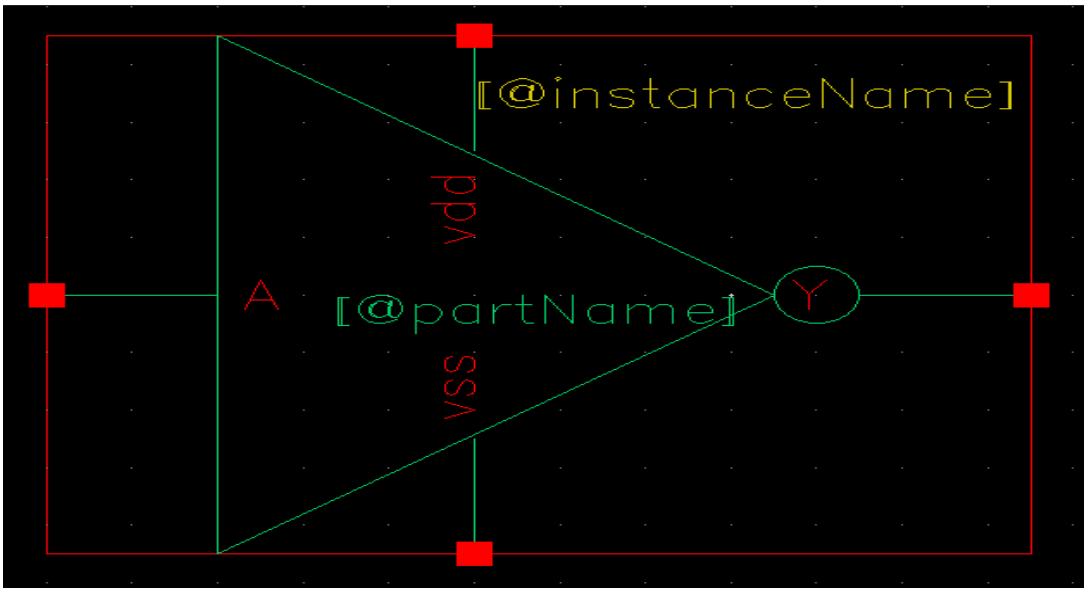
4. After creating the triangle press ***ESC*** key.

5. Execute **Create – Shape – Circle** to make a circle at the end of triangle.

6. You can move the pin names according to the location.

7. Execute **Create — Selection Box**. In the Add Selection Box form, click **Automatic**.A new red selection box is automatically added.

8. After creating symbol, click on the ***save*** icon in the symbol editor window to save the symbol. In the symbol editor, execute **File — Close** to close the symbol view window.



Building the Inverter_Test Design

Objective: To build an Inverter Test circuit using your Inverter

Creating the Inverter_Test Cellview

You will create the Inverter_Test cellview that will contain an instance of the Inverter cellview. In the next section, you will run simulation on this design

1. In the CIW or Library Manager, execute **File— New— Cellview**.

2. Set up the New File form as follows:

3. Click **OK** when done. A blank schematic window for the **Inverter_Test** design appears.

Building the Inverter_Test Circuit

1. Using the component list and Properties/Comments in this table, build the **Inverter_Test** schematic.

Library name	Cellview name	Properties/Comments
myDesignLib	Inverter	Symbol
analogLib	vpulse	v1=0, v2=1.8,td=0 tr=tf=1ns, ton=10n, T=20n
analogLib	vdc, gnd	vdc=1.8

Note: Remember to set the values for **VDD** and **VSS**. Otherwise, your circuit will have no power.

2. Add the above components using **Create — Instance** or by pressing **I**. 

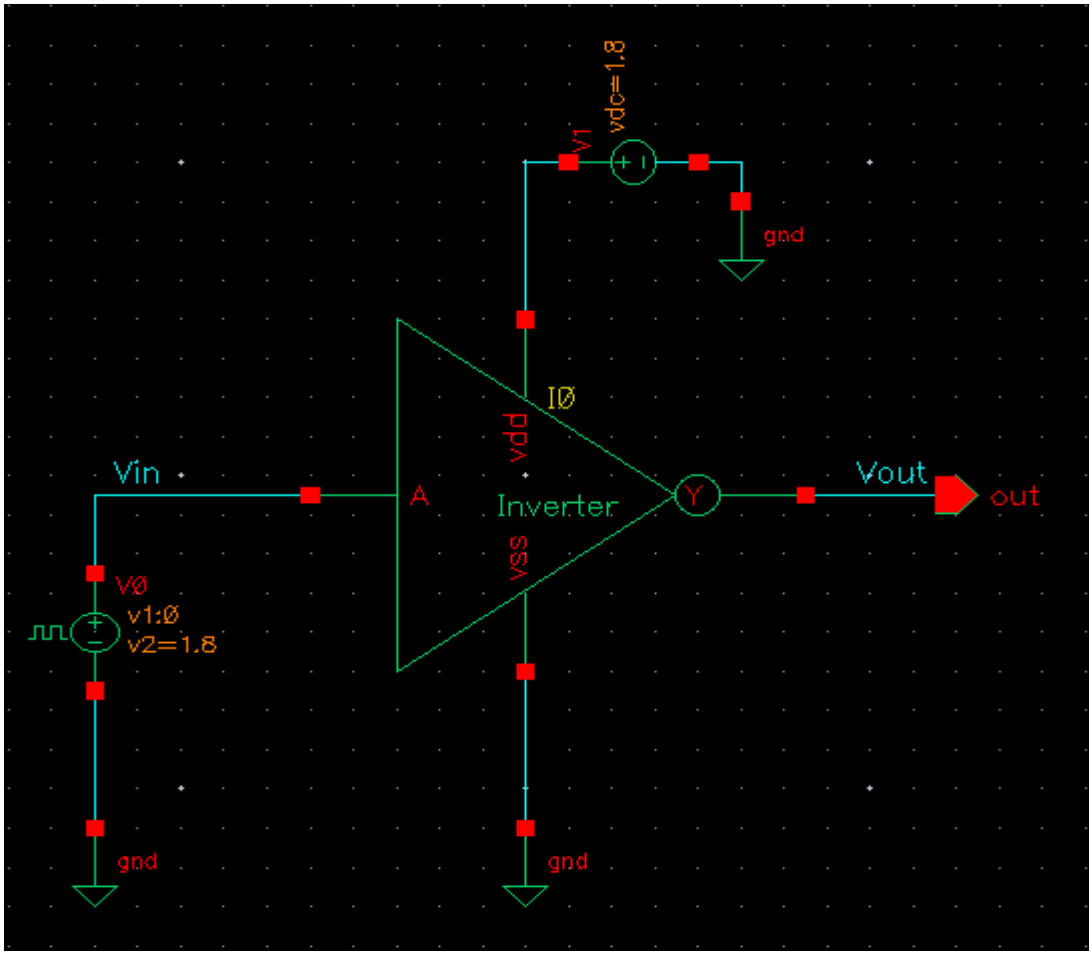
3. Click the **Wire (narrow)** icon and wire your schematic. 

Tip: You can also press the **w** key, or execute **Create— Wire (narrow)**.

4. Click **Create — Wire Name** or press **L** to name the input **(Vin)** and output **(Vout)** wires as in the below schematic. 

5. Click on the ***Check and Save*** icon to save the design. 

6. The schematic should look like this.



7. Leave your **Inverter_Test** schematic window open for the next section.

Analog Simulation with Spectre

Objective: To set up and run simulations on the Inverter_Test design

---

In this section, we will run the simulation for Inverter and plot the transient, DC characteristics and we will do Parametric Analysis after the initial simulation.

Starting the Simulation Environment

Start the Simulation Environment to run a simulation.

1. In the **Inverter_Test** schematic window, execute

Launch – ADE L

The **Virtuoso Analog Design Environment (ADE)** simulation window appears.

Choosing a Simulator

Set the environment to use the **Spectre® tool**, a high speed, highly accurate analog simulator. Use this simulator with the **Inverte*r_Test*** design, which is made-up of analog components.

1. In the simulation window (ADE), execute

Setup— Simulator/Directory/Host.

2. In the Choosing Simulator form, set the Simulator field to **spectre**

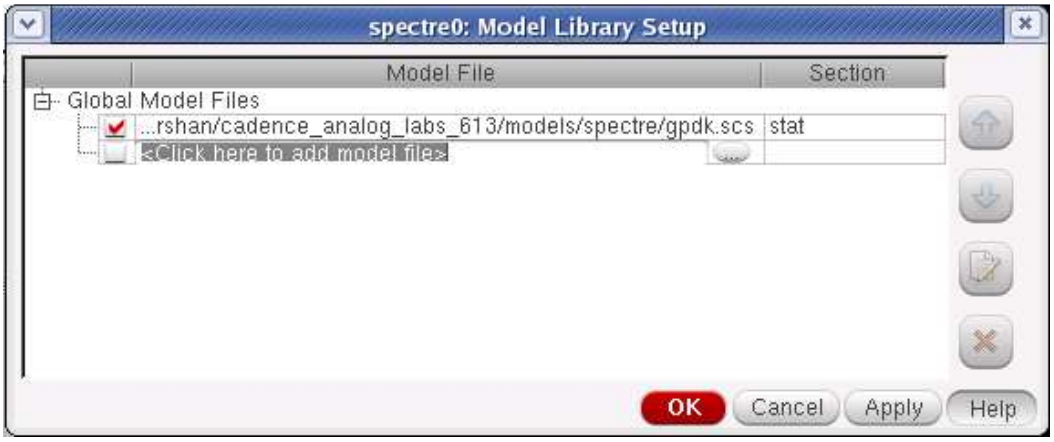
(Not spectreS) and click **OK**.

Setting the Model Libraries

The Model Library file contains the model files that describe the nmos and pmos devices during simulation.

1. In the simulation window (ADE), Execute **Setup - Model Libraries.** The Model Library Setup form appears. Click the **browse** button  to add **gpdk.scs** if not added by default as shown in the **Model Library Setup** form.

Remember to select the section type as **stat** in front of the gpdk.scs file. Your Model Library Setup window should now looks like the below figure.



To view the model file, highlight the expression in the Model Library File field and

Click **Edit File**.

2. To complete the Model Library Setup, move the cursor and click **OK**. The Model Library Setup allows you to include multiple model files. It also allows you to use the Edit button to view the model file.

Choosing Analyses

This section demonstrates how to view and select the different types of analyses to complete the circuit when running the simulation.

1. In the Simulation window (ADE), click the **Choose - Analyses** icon.

You can also execute **Analyses - Choose.**

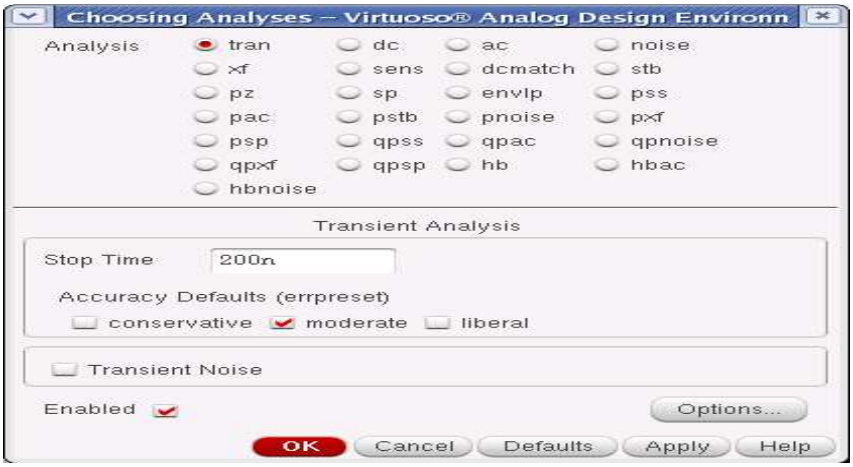
The Choosing Analysis form appears. This is a dynamic form, the bottom of the form changes based on the selection above.

2. To setup for transient analysis

a. In the Analysis section select **tran**

b. Set the stop time as **200n**

c. Click at the **moderate** or **Enabled** button at the bottom, and then click **Apply**.



3. To set up for DC Analyses:

a. In the Analyses section, select **dc**.

b. In the DC Analyses section, turn on **Save DC Operating Point.**

c. Turn on the **Component Parameter**.

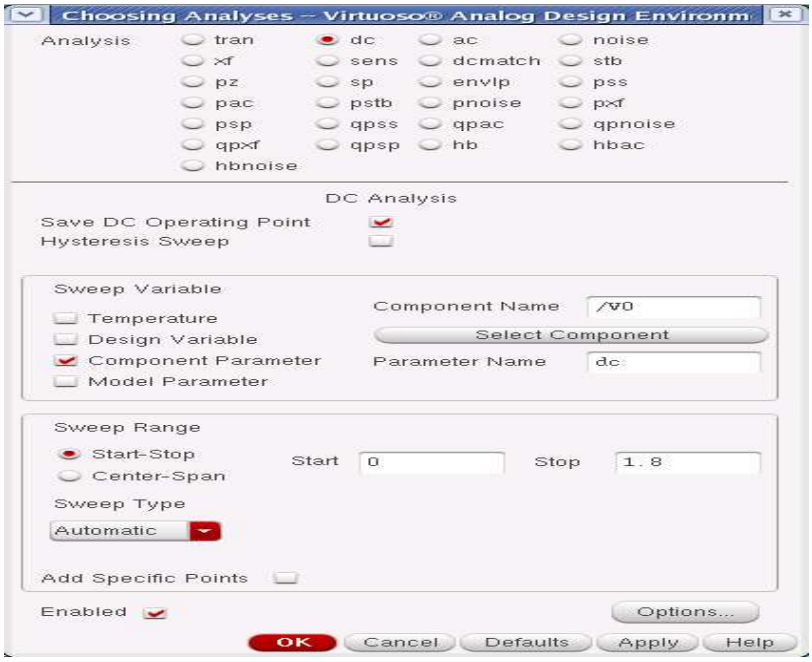
d. Double click the **Select Component**, Which takes you to the schematic window.

e. Select input signal **vpulse source** in the test schematic window.

f. Select **"DC Voltage"** in the **Select Component Parameter** form and click OK.

f. In the analysis form type **start** and **stop** voltages as **0** to **1.8** respectively.

g. Check the enable button and then click **Apply.**



4. Click ***OK*** in the Choosing Analyses Form.

Setting Design Variables

Set the values of any design variables in the circuit before simulating. Otherwise, the simulation will not run.

1. In the Simulation window, click the **Edit Variables** icon. 

The Editing Design Variables form appears.

2. Click **Copy From** at the bottom of the form. The design is scanned and all variables found in the design are listed. In a few moments, the **wp** variable appears in the Table of Design variables section.

3. Set the value of the **wp** variable: With the **wp** variable highlighted in the Table of Design Variables, click on the variable name **wp** and enter the following:

Value(Expr)	2u

Click **Change** and notice the update in the Table of Design Variables.

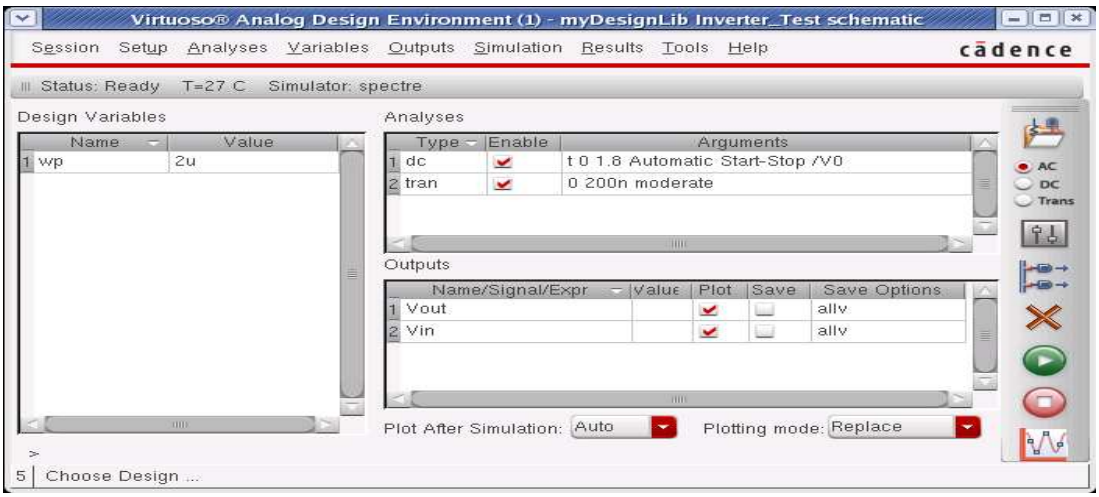
3. Click **OK** or **Cancel** in the Editing Design Variables window.

Selecting Outputs for Plotting

1. Execute **Outputs – To be plotted – Select on Schematic** in the simulation window.

2. Follow the prompt at the bottom of the schematic window, Click on output net **Vout**, input net **Vin** of the Inverter. Press **ESC** with the cursor in the schematic after selecting it.

Does the simulation window look like this?

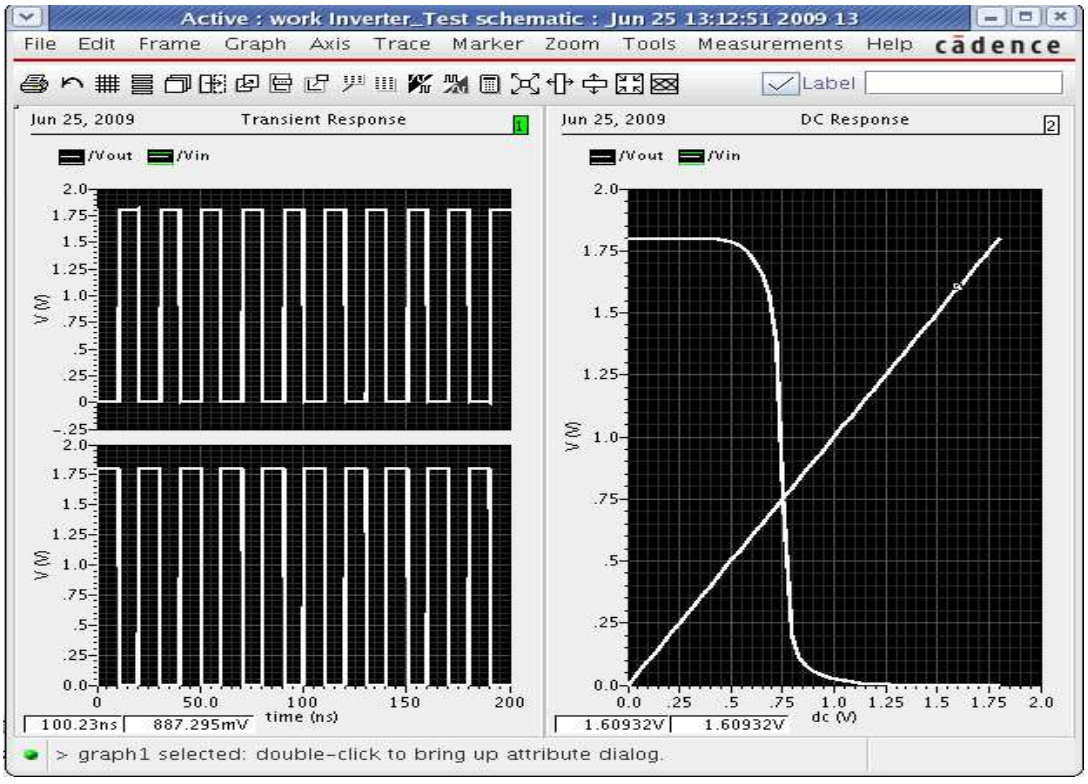


Running the Simulation

1. Execute **Simulation – Netlist and Run** in the simulation window to start the 

Simulation or the icon, this will create the netlist as well as run the simulation.

2. When simulation finishes, the Transient, DC plots automatically will be popped up along with log file.



Saving the Simulator State

We can save the simulator state, which stores information such as model library file, outputs, analysis, variable etc. This information restores the simulation environment without having to type in all of setting again.

1. In the Simulation window, execute **Session – Save State**. The Saving State form appears.

2. Set the **Save as** field to **state1_inv** and make sure all options are selected under what to save field.

3. Click **OK** in the saving state form. The Simulator state is saved.

Loading the Simulator State

1. From the ADE window execute **Session – Load State.**

2. In the Loading State window, set the State name to **state1_inv** as shown

3. Click **OK** in the Loading State window.

Parametric Analysis

Parametric Analysis yields information similar to that provided by the Spectre® sweep feature, except the data is for a full range of sweeps for each parametric step. The Spectre sweep feature provides sweep data at only one specified condition.

You will run a parametric DC analysis on the **wp** variable, of the PMOS device of the Inverter design by sweeping the value of **wp**.

Run a simulation before starting the parametric tool. You will start by loading the state from the previous simulation run.

Run the simulation and check for errors. When the simulation ends, a single waveform in the waveform window displays the DC Response at the **Vout** node.

Starting the Parametric Analysis Tool

1. In the Simulation window, execute **Tools—Parametric Analysis**. The Parametric Analysis form appears.

2. In the Parametric Analysis form, execute **Setup—Pick Name For Variable—Sweep 1**.

A selection window appears with a list of all variables in the design that you can sweep. This list includes the variables that appear in the Design Variables section of the Simulation window.

3. In the selection window, double click left on **wp**.

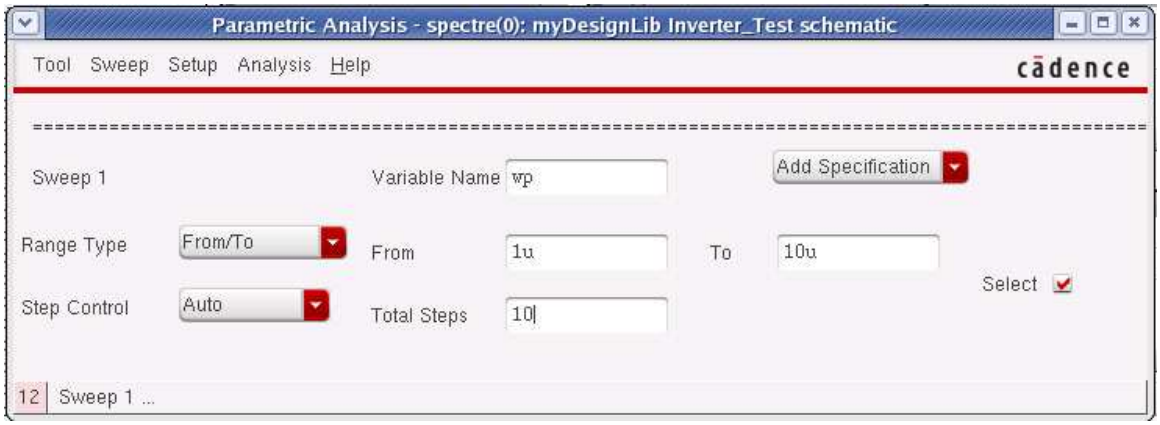
The Variable Name field for Sweep 1 in the Parametric Analysis form is set to **wp**.

4. Change the Range Type and Step Control fields in the Parametric

Analysis form as shown below:

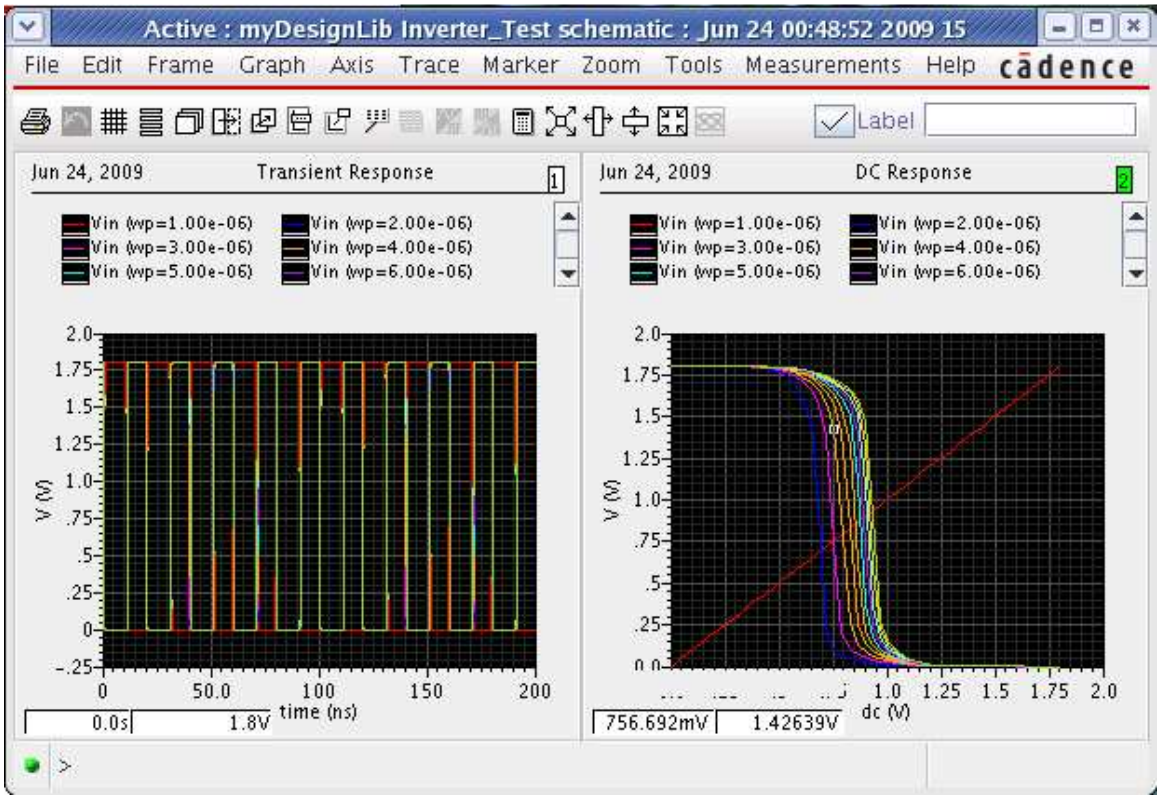
Range Type	**From/To**	From	**1u**	To	**10u**
Step Control	**Auto**	Total Steps	**10**		

These numbers vary the value of the **wp** of the pmos between 1um and 10um at ten evenly spaced intervals.



5. Execute **Analysis—Start**.

The Parametric Analysis window displays the number of runs remaining in the analysis and the current value of the swept variable(s). Look in the upper right corner of the window. Once the runs are completed the wavescan window comes up with the plots for different runs.



Note: Change the wp value of pmos device back to 2u and save the schematic before proceeding to the next section of the lab. To do this use edit property option.

Creating Layout View of Inverter

1. From the **Inverter** schematic window menu execute

Launch – Layout XL. A **Startup Option** form appears.

2. Select ***Create New*** option. This gives a New Cell View Form

3. Check the Cellname **(Inverter)**, Viewname **(layout).**

4. Click **OK** from the New Cellview form.

LSW and a blank layout window appear along with schematic window.

Adding Components to Layout

1. Execute **Connectivity – Generate – All from Source** or click the icon  in the layout editor window, **Generate Layout** form appears. Click **OK** which imports the schematic components in to the Layout window automatically.

2. Re arrange the components with in PR-Boundary as shown in the next page.

3. To rotate a component, Select the component and execute **Edit –Properties**. Now select the degree of rotation from the property edit form.

4. To Move a component, Select the component and execute **Edit -Move** command.

Making interconnection

1. Execute **Connectivity –Nets – Show/Hide selected Incomplete Nets** or click  the icon in the Layout Menu.

2. Move the mouse pointer over the device and click **LMB** to get the connectivity information, which shows the guide lines (or flight lines) for the inter connections of the components.

3. From the layout window execute **Create – Shape – Path/ Create wire** or **Create – Shape – Rectangle (**for vdd and gnd bar**)** and select the appropriate Layers from the **LSW** window and Vias for making the inter connections

Creating Contacts/Vias

You will use the contacts or vias to make connections between two different layers.

1. Execute **Create — Via** or select  command to place different Contacts, as given in below table

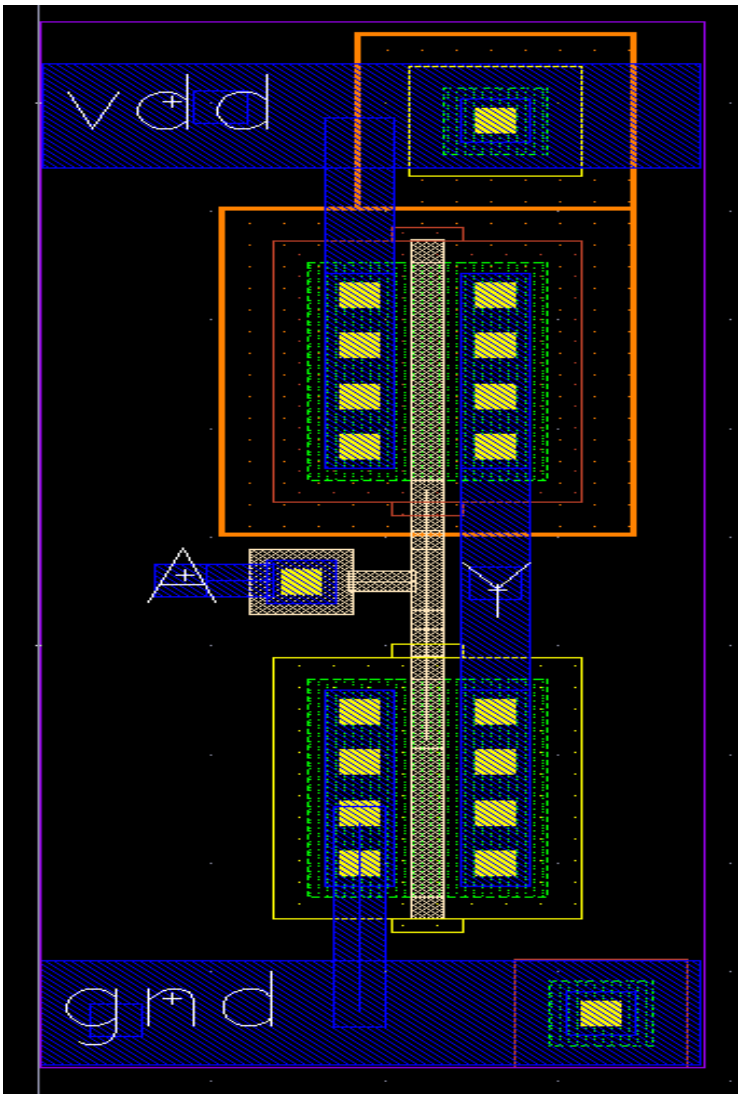
Connection	Contact Type
For Metal1- Poly Connection	Metal1-Poly
For Metal1- Psubstrate	Metal1-Psub

Connection

For Metal1- Nwell Connection Metal1-Nwell

Saving the design

1. Save your design by selecting **File — Save** or click  to save the layout, and layout should appear as below.



Physical Verification

Assura DRC

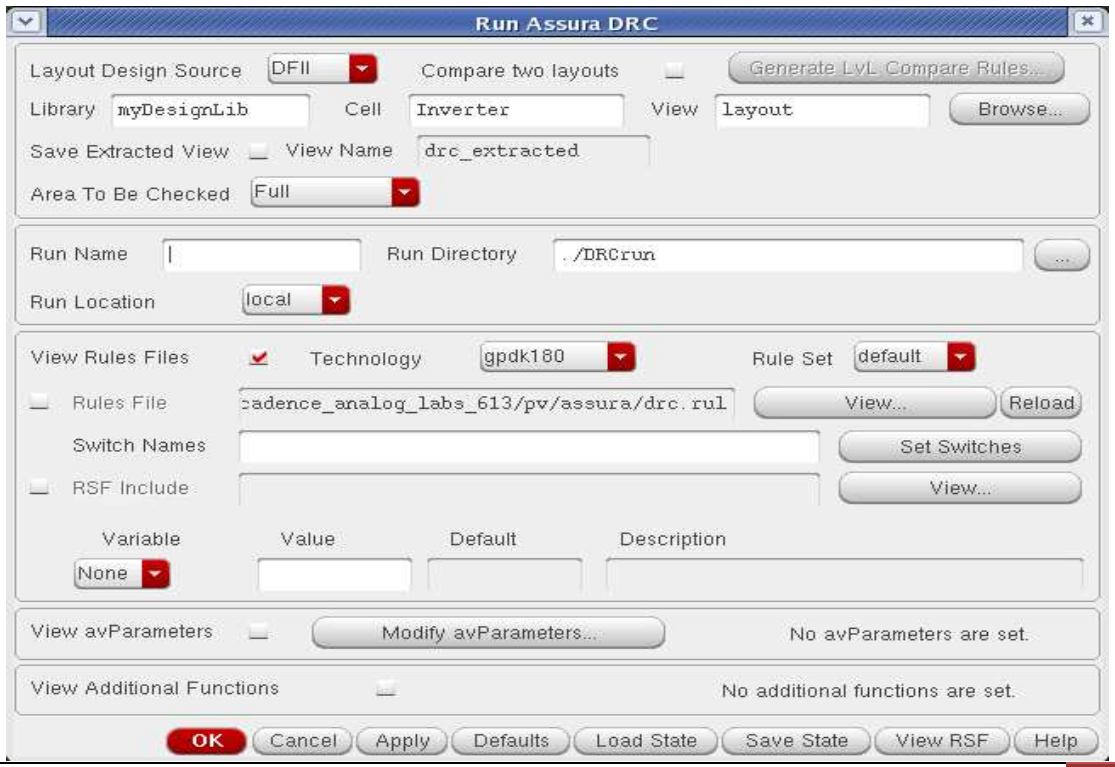
Running a DRC

1. Open the Inverter layout form the CIW or library manger if you have closed that.

Press **shift – f** in the layout window to display all the levels.

2. Select **Assura - Run DRC** from layout window. The DRC form appears. The Library and Cellname are taken from the current design window, but rule file may be missing. Select the Technology as **gpdk180.** This automatically loads the rule file.

Your DRC form should appear like this



3. Click **OK** to start DRC.

4. A Progress form will appears. You can click on the watch log file to see the log file.

5. When DRC finishes, a dialog box appears asking you if you want to view your DRC results, and then click **Yes** to view the results of this run.

6. If there any DRC error exists in the design **View Layer Window** (VLW) and **Error Layer Window** (ELW) appears. Also the errors highlight in the design itself.

7. Click **View – Summary** in the ELW to find the details of errors.

8. You can refer to rule file also for more information, correct all the DRC errors and **Re – run** the DRC.

9. If there are no errors in the layout then a dialog box appears with **No DRC errors found** written in it, click on **close** to terminate the DRC run.

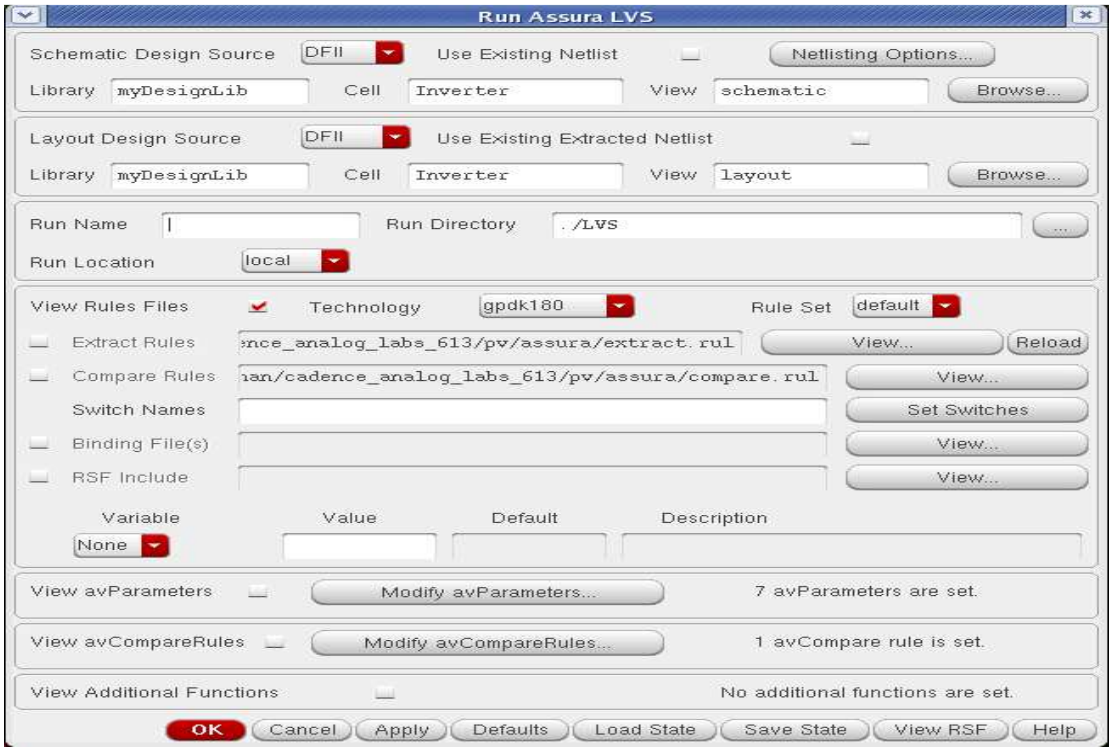
ASSURA LVS

In this section we will perform the LVS check that will compare the schematic netlist and the layout netlist.

Running LVS

1. Select **Assura – Run LVS** from the layout window. The Assura Run LVS form appears. It will automatically load both the schematic and layout view of the cell.

2. Change the following in the form and click **OK**.



3. The LVS begins and a Progress form appears.

4. If the schematic and layout matches completely, you will get the form displaying **Schematic and Layout Match**.

5. If the schematic and layout do not matches, a form informs that the LVS completed successfully and asks if you want to see the results of this run.

6. Click **Yes** in the form LVS debug form appears, and you are directed into LVS debug environment.

7. In the **LVS debug form** you can find the details of mismatches and you need to correct all those mismatches and **Re – run** the LVS till you will be able to match the schematic with layout.

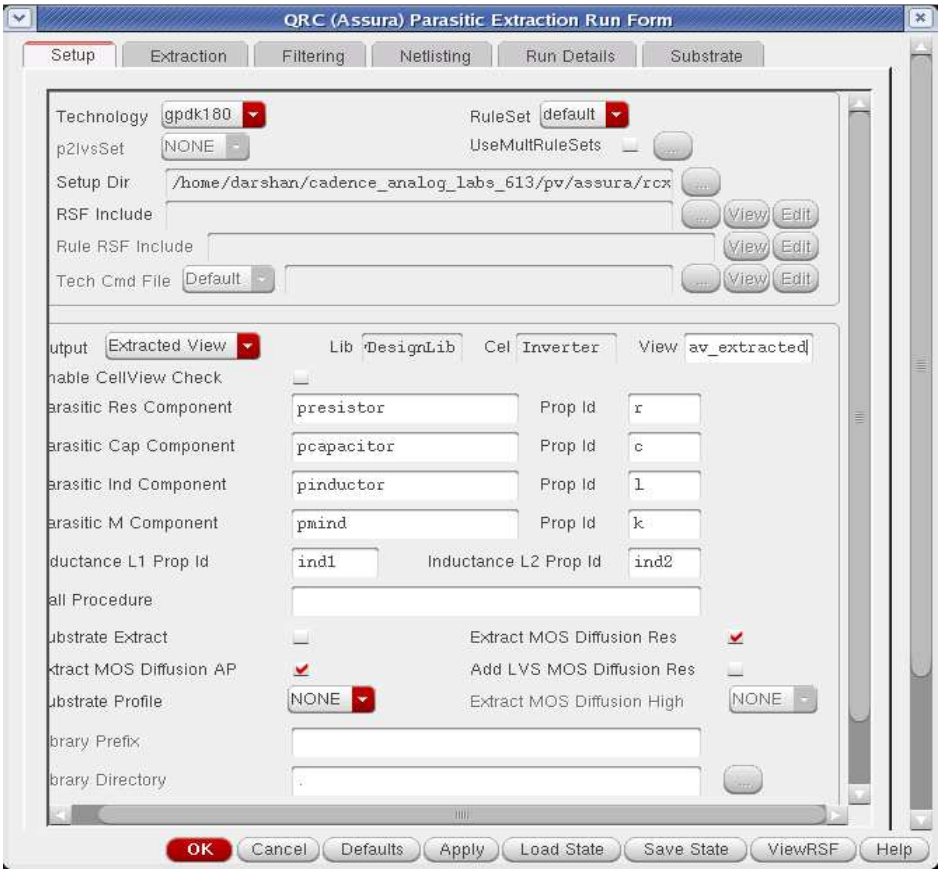
Assura RCX

In this section we will extract the RC values from the layout and perform analog circuit simulation on the designs extracted with RCX. Before using RCX to extract parasitic devices for simulation, the layout should match with schematic completely to ensure that all parasites will be backannoted to the correct schematic nets.

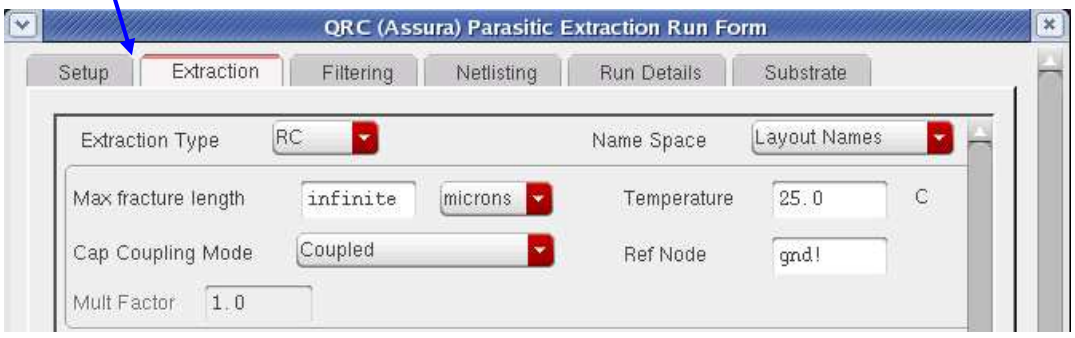
Running RCX

1. From the layout window execute **Assura – Run RCX**.

2. Change the following in the Assura parasitic extraction form. Select **output** type under **Setup** tab of the form.



3. In the **Extraction** tab of the form, choose Extraction type, Cap Coupling Mode and specify the Reference node for extraction.



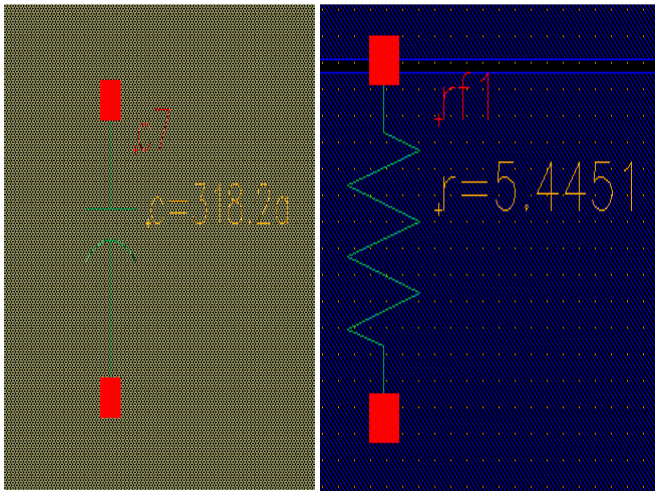
4. In the **Filtering** tab of the form, **Enter Power Nets** as **vdd!, vss!** and **Enter Ground Nets** as **gnd!**

5. Click **OK** in the Assura parasitic extraction form when done. The RCX progress form appears, in the progress form click **Watch log file** to see the output log file.

5. When RCX completes, a dialog box appears, informs you that **Assura RCX run**

Completed successfully.

6. You can open the **av_extracted** view from the library manager and view the parasitic.



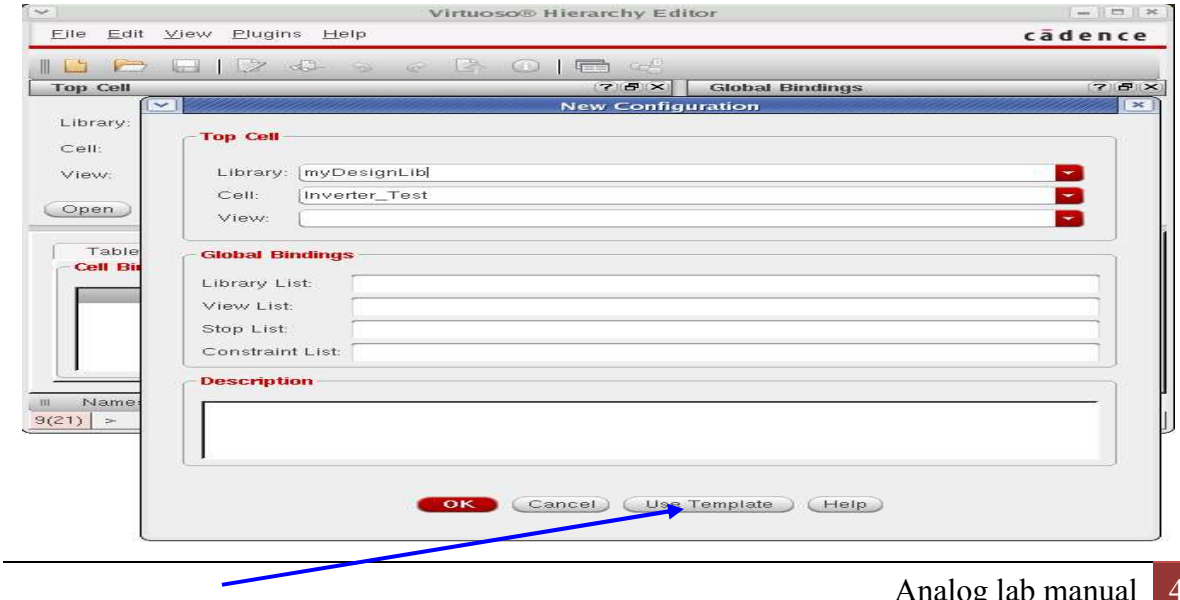
Creating the Configuration View

In this section we will create a config view and with this config view we will run the Simulation with and without parasitic.

1. In the CIW or Library Manager, execute **File – New – Cellview**

2. In the Create New file form, set the following:

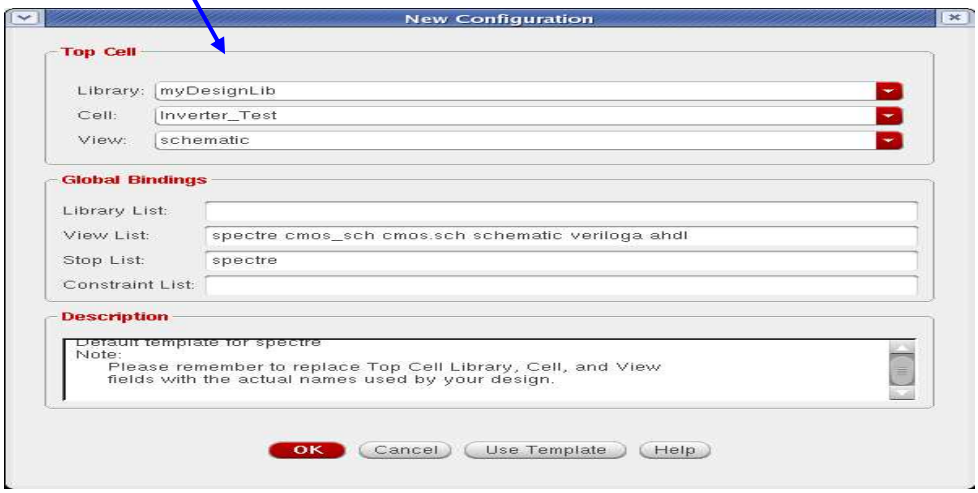
3. Click **OK** in create **New File** form. The **Hierarchy Editor** form opens and a **New Configuration** form opens in front of it.



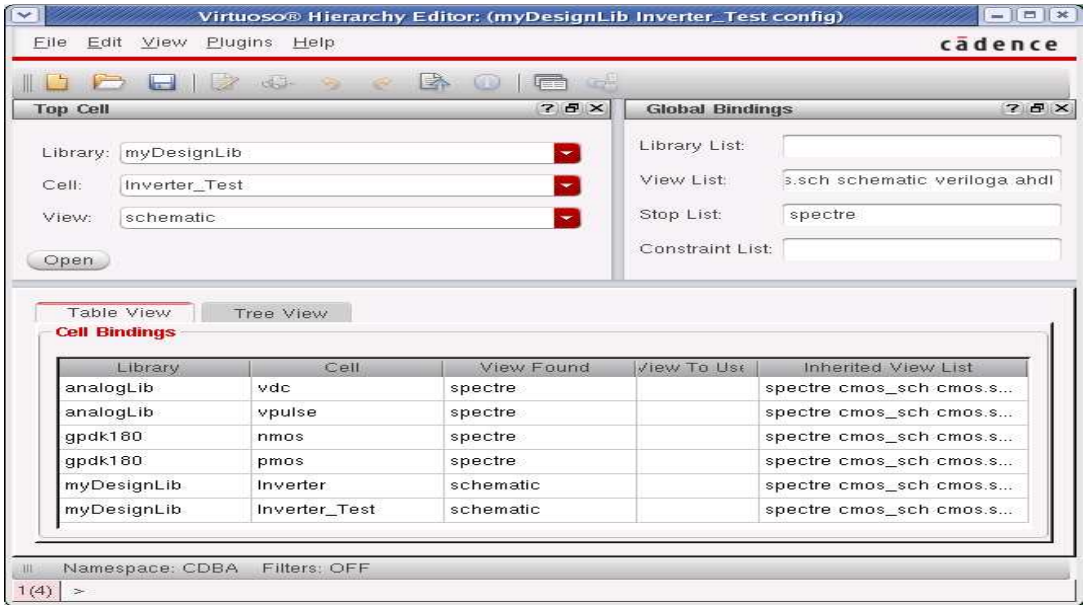
4. Click **Use template** at the bottom of the **New Configuration** form and select **Spectre** in the cyclic field and click **OK**. The Global Bindings lists are loaded from the template.

5. Change the **Top Cell** View to **schematic** and remove the default entry from the **Library List** field.

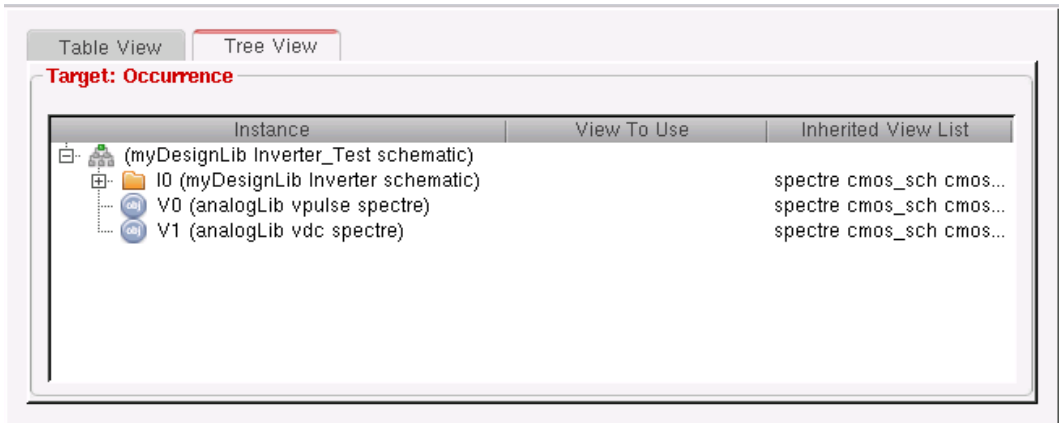
6. Click **OK** in the New Configuration form.



The hierarchy editor displays the hierarchy for this design using table format.



7. Click the **Tree View** tab. The design hierarchy changes to tree format. The form should look like this:



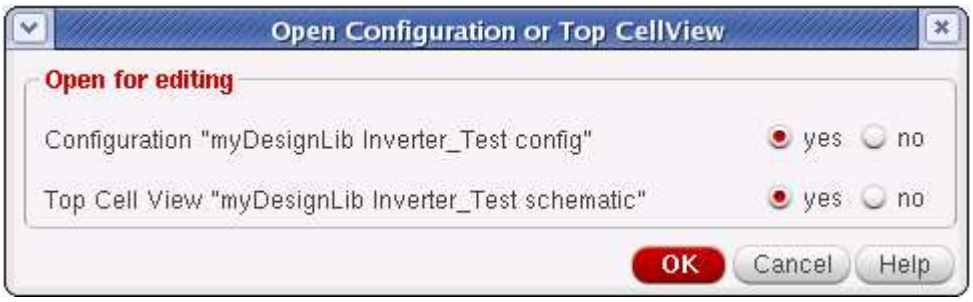
8. ***Save*** the current configuration. 

9. Close the Hierarchy Editor window. Execute **File – Close Window**.

To run the Circuit without Parasites

1. From the Library Manager open **Inverter_Test** Config view.

Open Configuration or Top cellview form appears.



2. In the form, turn on the both cyclic buttons to **Yes** and click **OK.**

The Inverter_Test schematic and Inverter_Test config window appears. Notice the window banner of schematic also states **Config: myDesignLib Inverter_Test config.**

3. Execute **Launch – ADE L** from the schematic window.

4. Now you need to follow the same procedure for running the simulation. Executing **Session– Load state,** the Analog Design Environment window loads the previous state.

5. Click **Netlist and Run** icon to start the simulation. 

The simulation takes a few seconds and then waveform window appears.

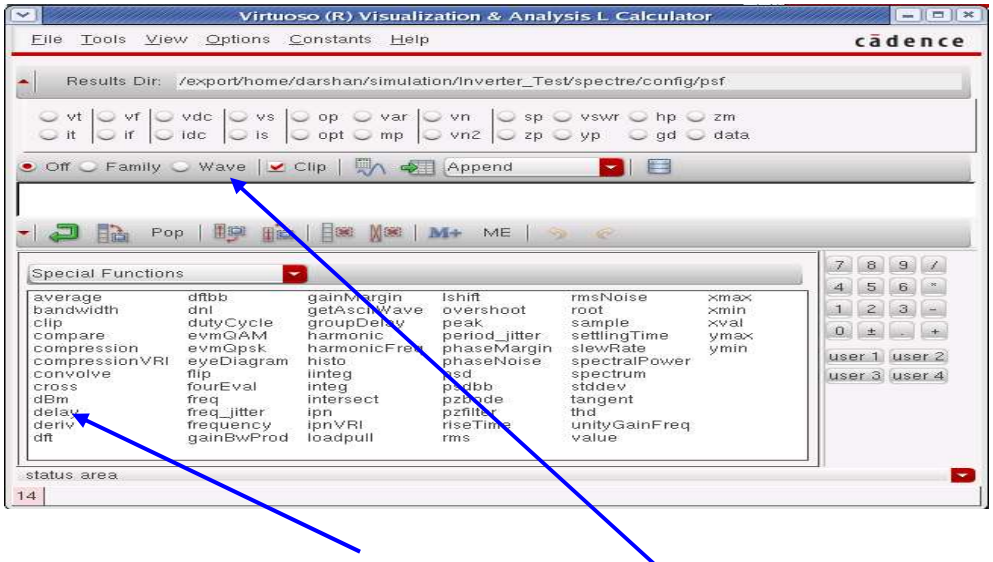
6. In the CIW, note the netlisting statistics in the **Circuit inventory** section. This list includes all nets, designed devices, source and loads. There are no

parasitic components. Also note down the circuit inventory section.

Measuring the Propagation Delay

1. In the waveform window execute **Tools – Calculator.**

The calculator window appears.



2. From the functions select **delay,** this will open the delay data panel.

3. Place the cursor in the text box for Signal1, select the **wave** button and select the input waveform from the waveform window.

4. Repeat the same for Signal2, and select the output waveform.

5. Set the **Threshold value 1** and **Threshold value 2** to 0.9, this directs the calculator to calculate delay at 50% i.e. at 0.9 volts.

6. Execute **OK** and observe the expression created in the calculator buffer.

7. Click on **Evaluate the buffer icon**  to perform the calculation, note down the value returned after execution.

8. Close the calculator window.

To run the Circuit with Parasites

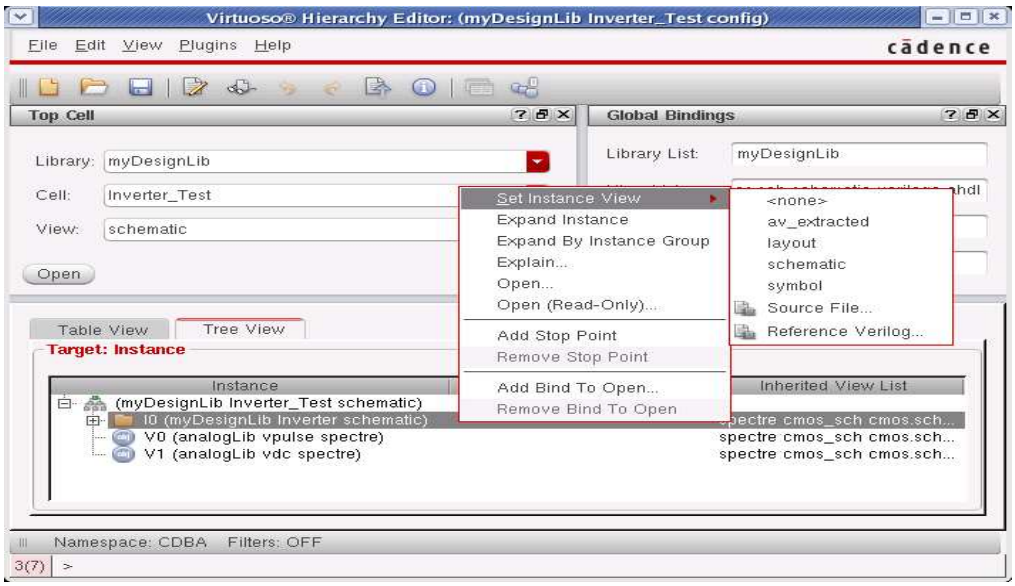
In this exercise, we will change the configuration to direct simulation of the **av_extracted** view which contains the parasites.

1. Open the same Hierarchy Editor form, which is already set for Inverter_Test config.

2. Select the **Tree View** icon: this will show the design hierarchy in the tree format.

3. Click **right** mouse on the Inverter schematic.

A pull down menu appears. Select **av_extracted** view from the **Set Instance view** menu, the View to use column now shows av_extracted view.



4. Click on the **Recompute the hierarchy** icon,  the configuration is now updated from schematic to av_extracted view.

6. From the **Analog Design Environment** window click **Netlist and Run** to 

start the simulation again.

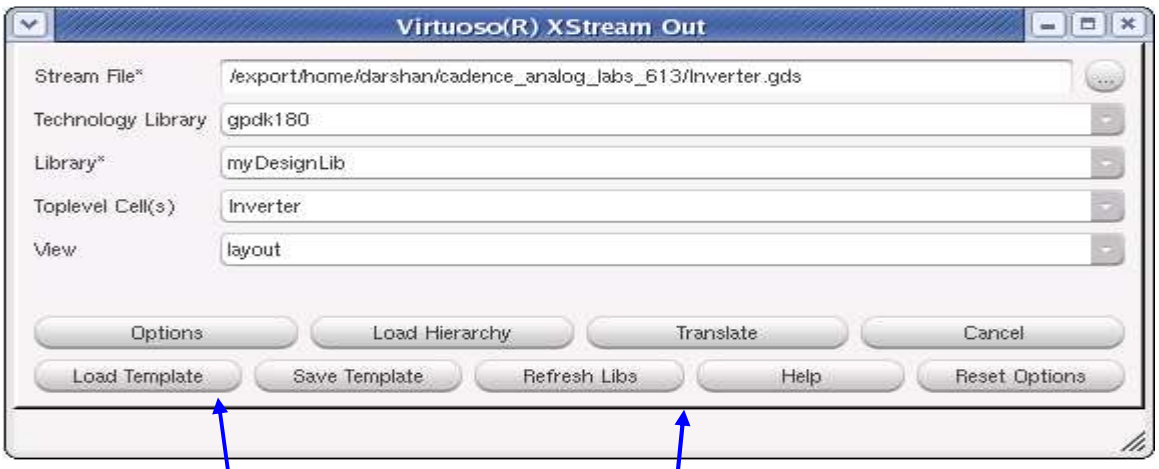
7. When simulation completes, note the **Circuit inventory conditions**, this time the list shows all nets, designed devices, sources and parasitic devices as well.

8. Calculate the delay again and match with the previous one. Now you can conclude how much delay is introduced by these parasites, now our main aim should to minimize the delay due to these parasites so number of iteration takes place for making an optimize layout.

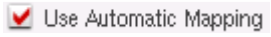
Generating Stream Data

Streaming Out the Design

1. Select **File – Export – Stream** from the CIW menu and **Virtuoso Xstream out** form appears change the following in the form.



2. Click on the **Options** button.

3. In the **StreamOut-Options** form select 

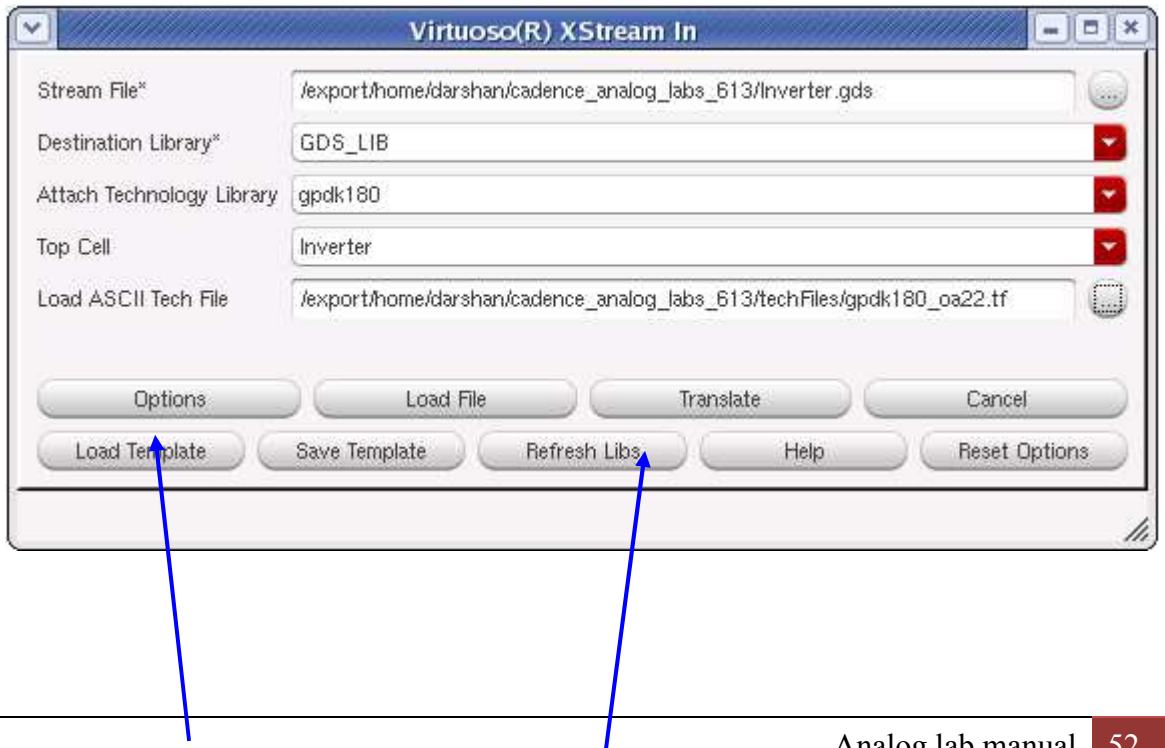
under **Layers** tab and click **OK.**

4. In the **Virtuoso XStream Out** form, click **Translate** button to start the stream translator.

5. The stream file Inverter.gds is stored in the specified location.

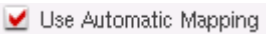
Streaming In the Design

1. Select **File – Import – Stream** from the CIW menu and change the following in the form.



You need to specify the **gpdk180_oa22.tf** file. This is the entire technology file that has been dumped from the design library.

2. Click on the **Options** button.

3. In the **StreamOut-Options** form select 

under **Layers** tab and click **OK.**

4. In the **Virtuoso XStream Out** form, click **Translate** button to start the stream translator.

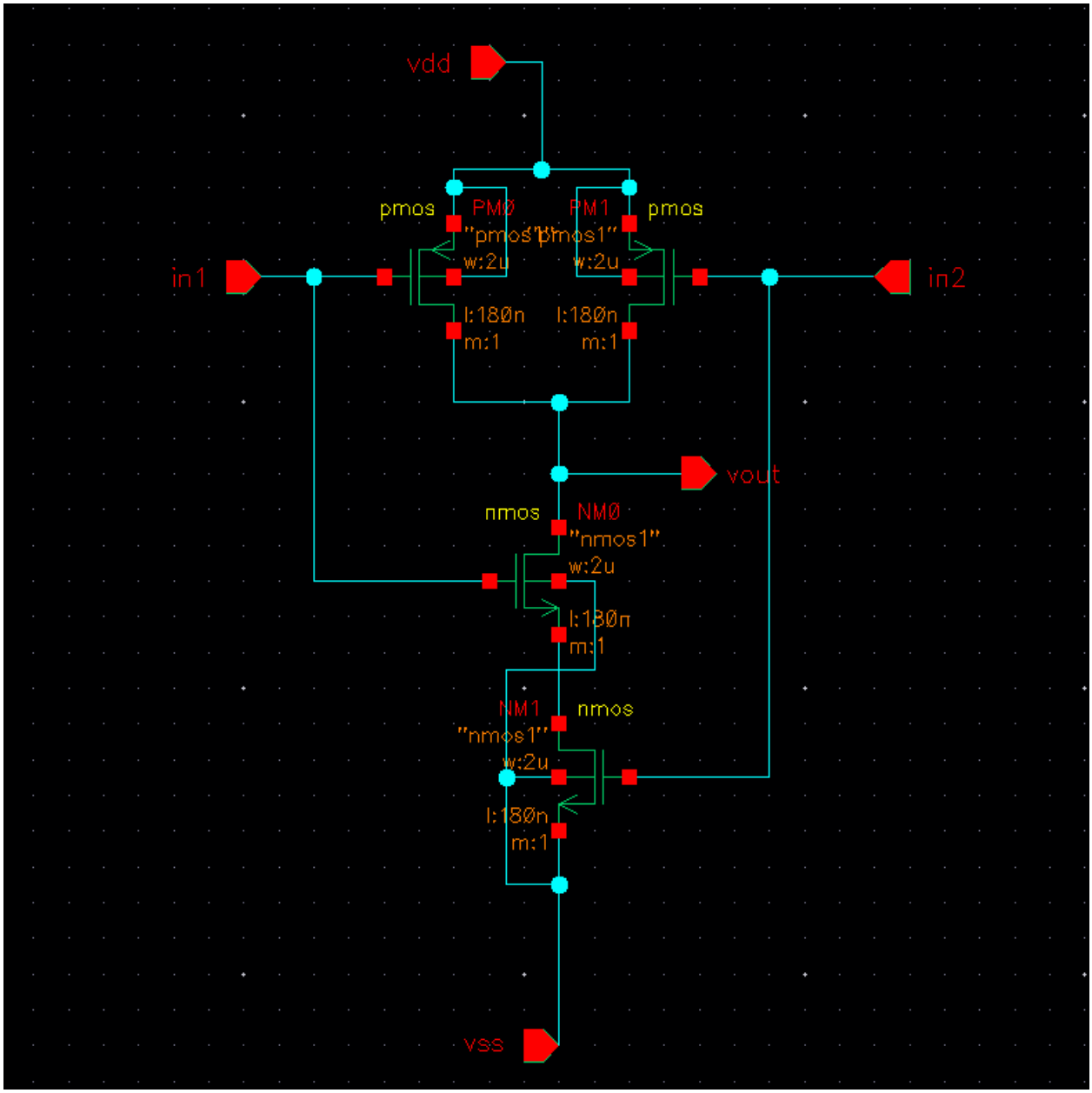
5. From the Library Manager open the **Inverter** cellview from the **GDS_LIB** library and notice the design.

6. Close all the windows except CIW window, which is needed for the next lab.

END OF LAB 2.1

Lab 2.2: A NAND GATE

Schematic Capture



Schematic Entry

Objective: To create a new cell view and build A NAND gate

Use the techniques learned in the Lab2.1 to complete the schematic of NAND gate.

This is a table of components for building the nand gate schematic.

Library name	Cell Name	Properties/Comments
gpdk180	Pmos	Model Name = pmos1,pmos2;
gpdk180	Nmos	Model Name =nmos1,nmos2;

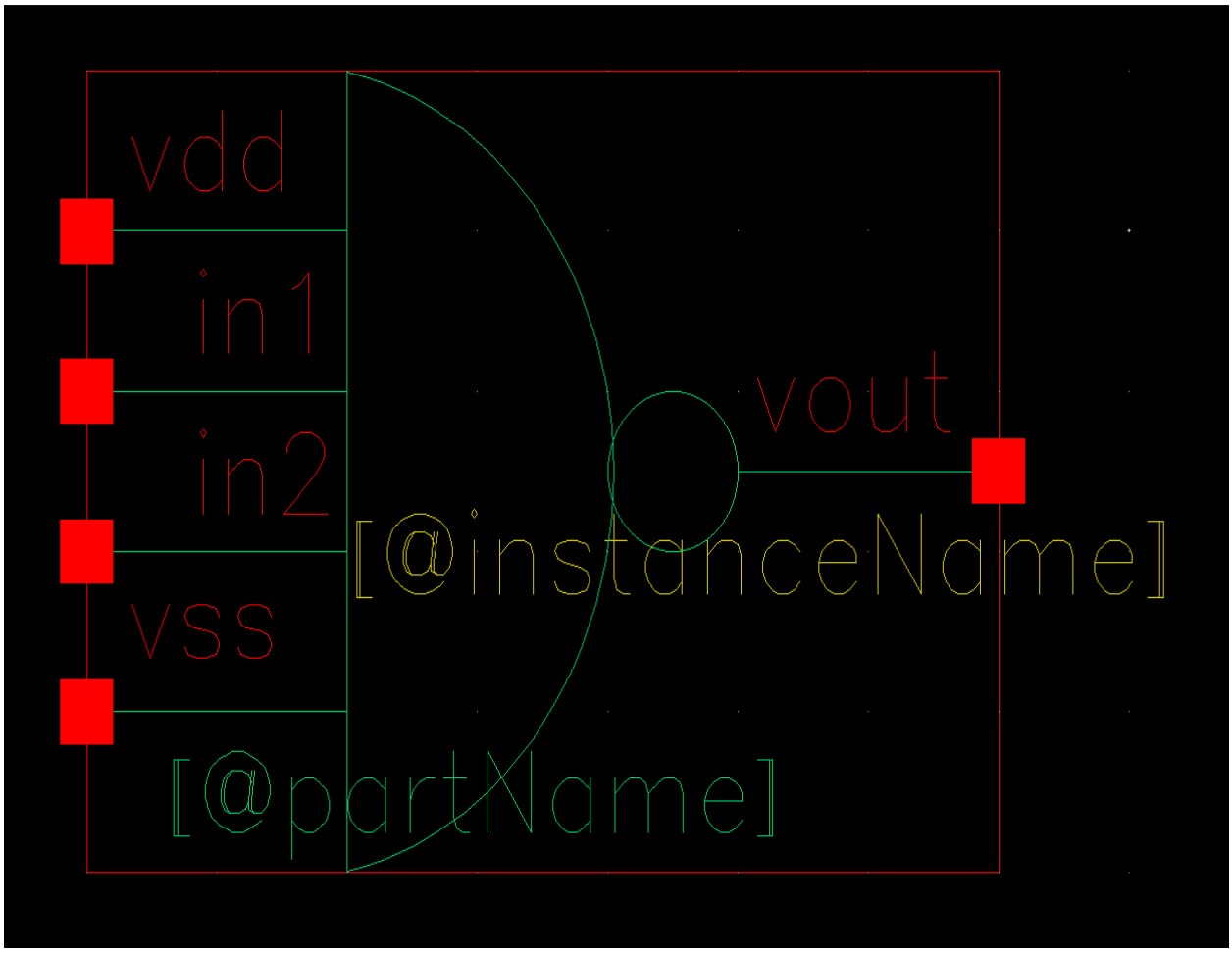
Type the following in the ADD pin form in the exact order leaving space between the pin names.

Pin Names	Direction
Vin1 vin2	Input
vout	Output
vdd vss	Input

Symbol Creation

Objective: To create a symbol for the nand gate

Use the techniques learned in the Lab2.1 to complete the symbol of NAND gate



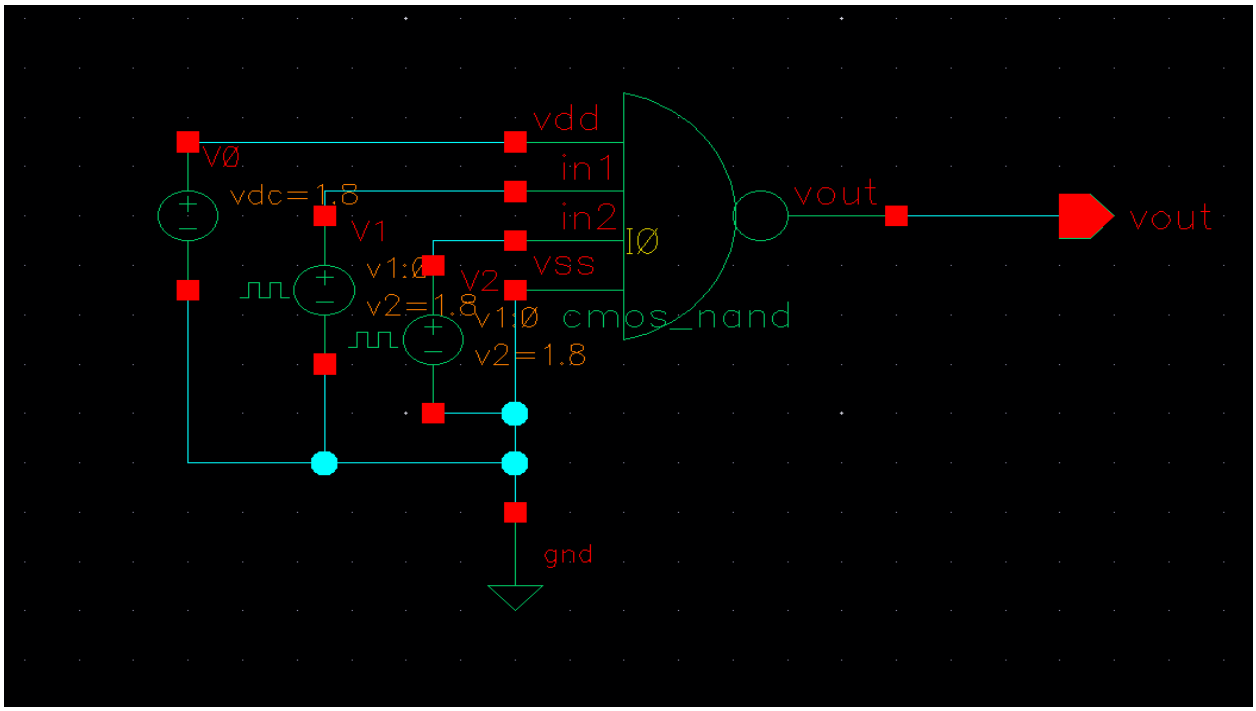
Building the NAND Test Design

Objective: To build NAND_test circuit using your NAND gate

Using the component list and Properties/Comments in the table,

build the cs-amplifier_test schematic as shown below.

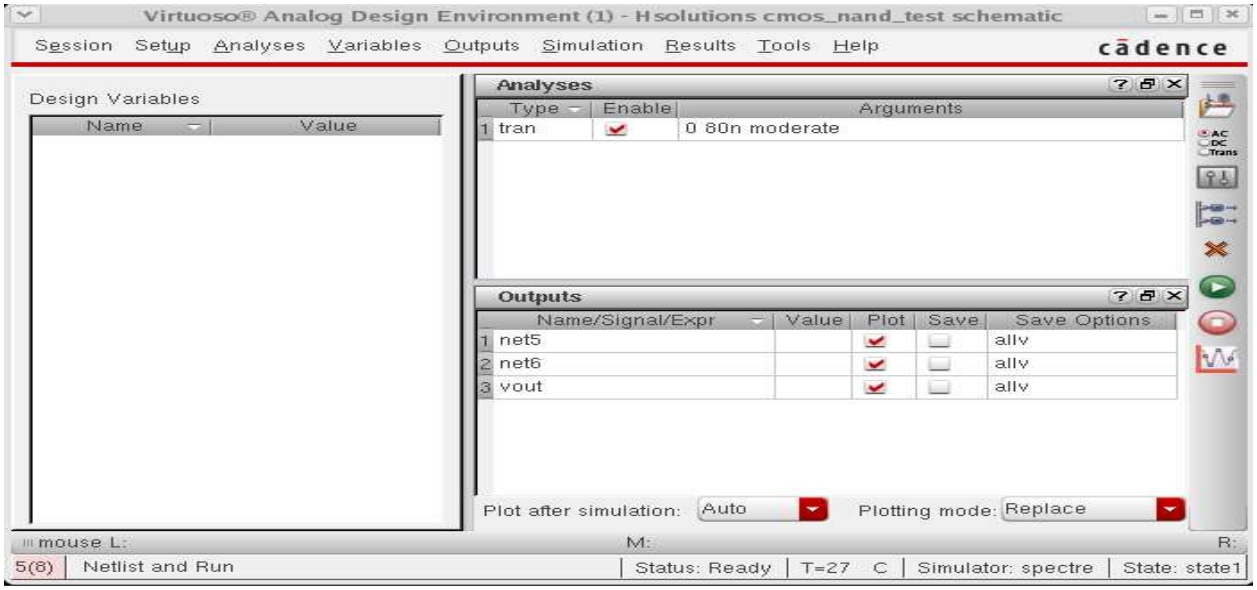
Library name	Cellview name	Properties/Comments
myDesignLib	cmos_nand	Symbol
analogLib	vpulse	Define pulse specification as In lab 2.1
analogLib	vdd,vss,gnd	vdd=1.8 ; vss= 1.8

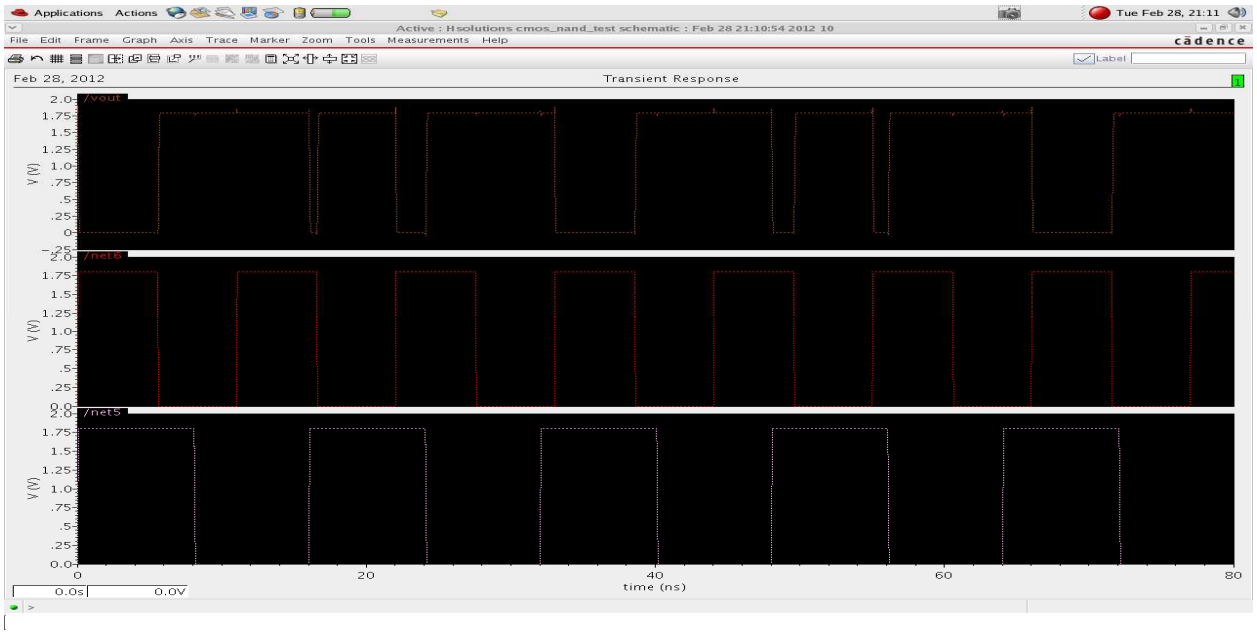


Analog Simulation with Spectre

Objective: To set up and run simulations on the NAND gate design.

Use the techniques learned in the Lab2.1 to complete the simulation of NAND gate, ADE window and waveform should look like below.



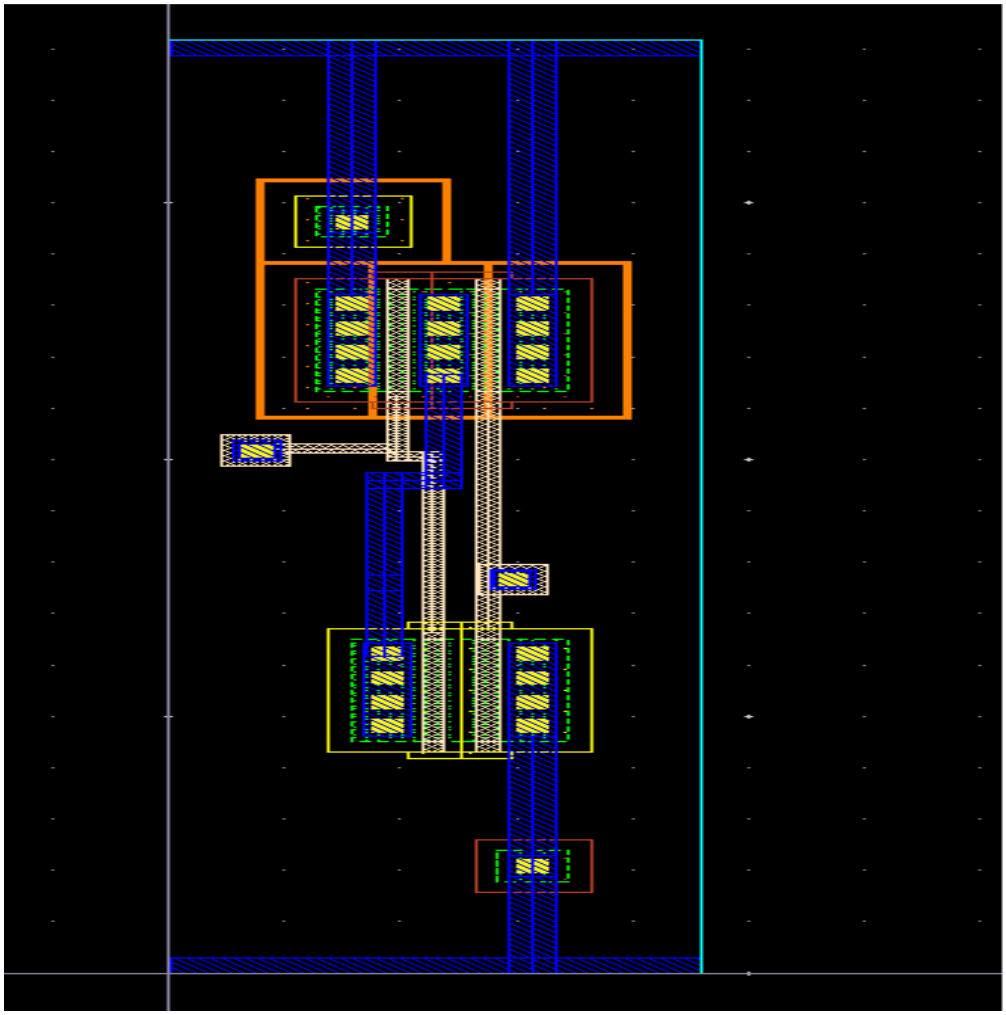


Creating a layout view of NAND gate

Use the techniques learned in the Lab2.1 to complete the layout of NAND gate.

Complete the DRC, LVS check using the assura tool.

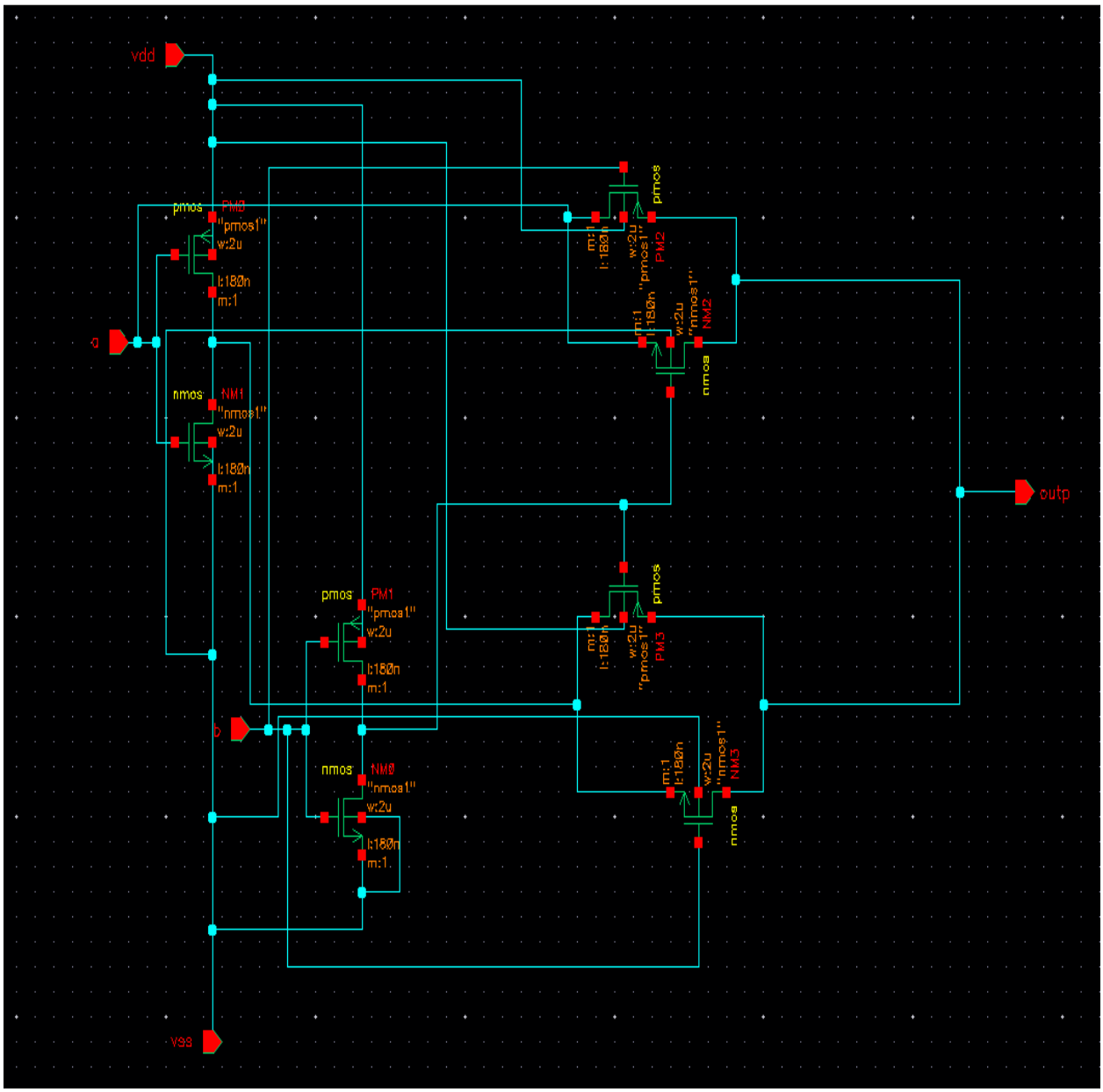
Extract RC parasites for back annotation and Re-simulation.



END OF LAB 2.2

Lab 2.3: A XOR GATE

Schematic Capture



Schematic Entry

Objective: To create a new cell view and build A XOR gate

Use the techniques learned in the Lab2.1 to complete the schematic of XOR gate.

This is a table of components for building the XOR gate schematic.

Library name	Cell Name	Properties/Comments
gpdk180	Pmos	Model Name = pmos1,pmos2,pmos3,pmos4;
gpdk180	Nmos	Model Name =nmos1,nmos2,nmos3,nmos4;

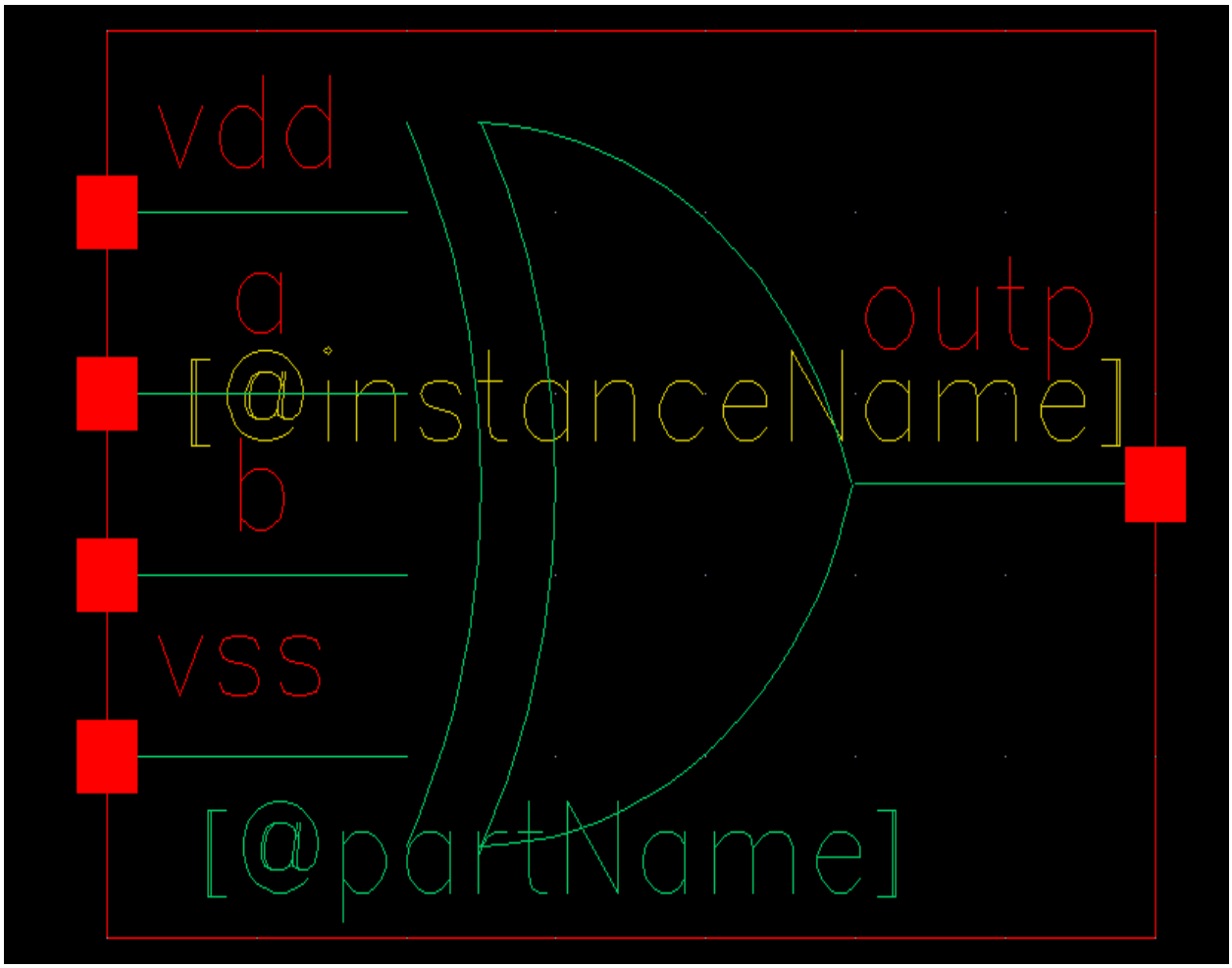
Type the following in the ADD pin form in the exact order leaving space between the pin names.

Pin Names	Direction
Vin1 vin2	Input
vout	Output
vdd vss	Input

Symbol Creation

Objective: To create a symbol for the XOR gate

Use the techniques learned in the Lab2.1 to complete the symbol of XOR gate



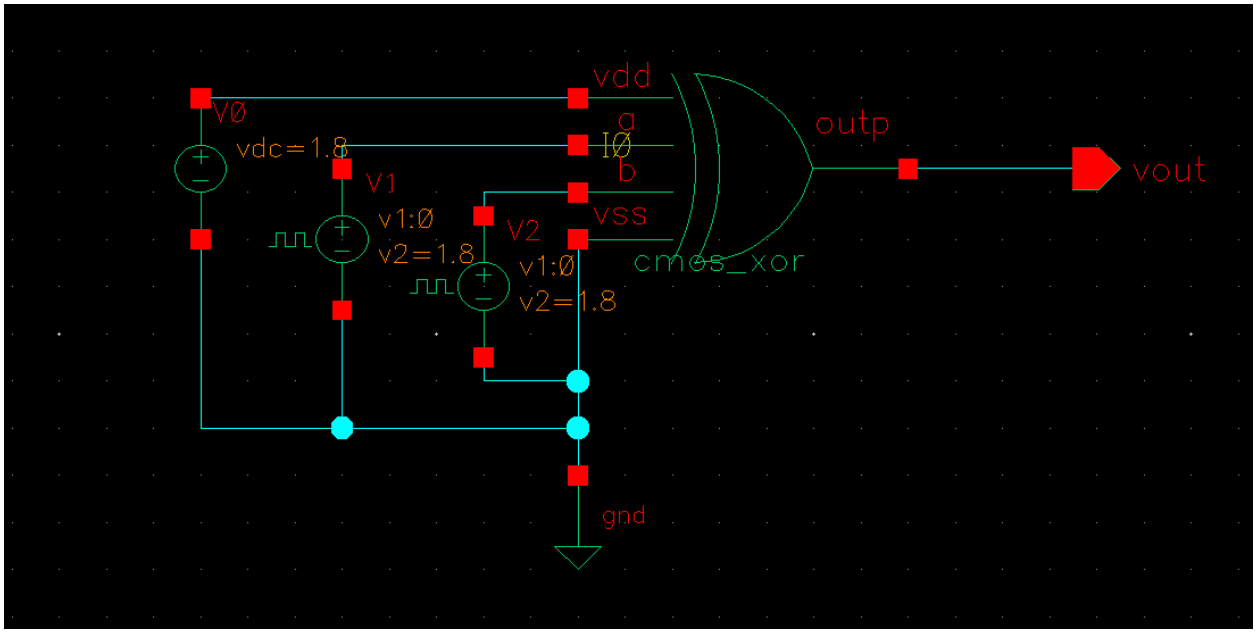
Building the XOR Gate Test Design

Objective: To build cmos_xor_test circuit using your cmos_xor

Using the component list and Properties/Comments in the table,

build the cs-amplifier_test schematic as shown below.

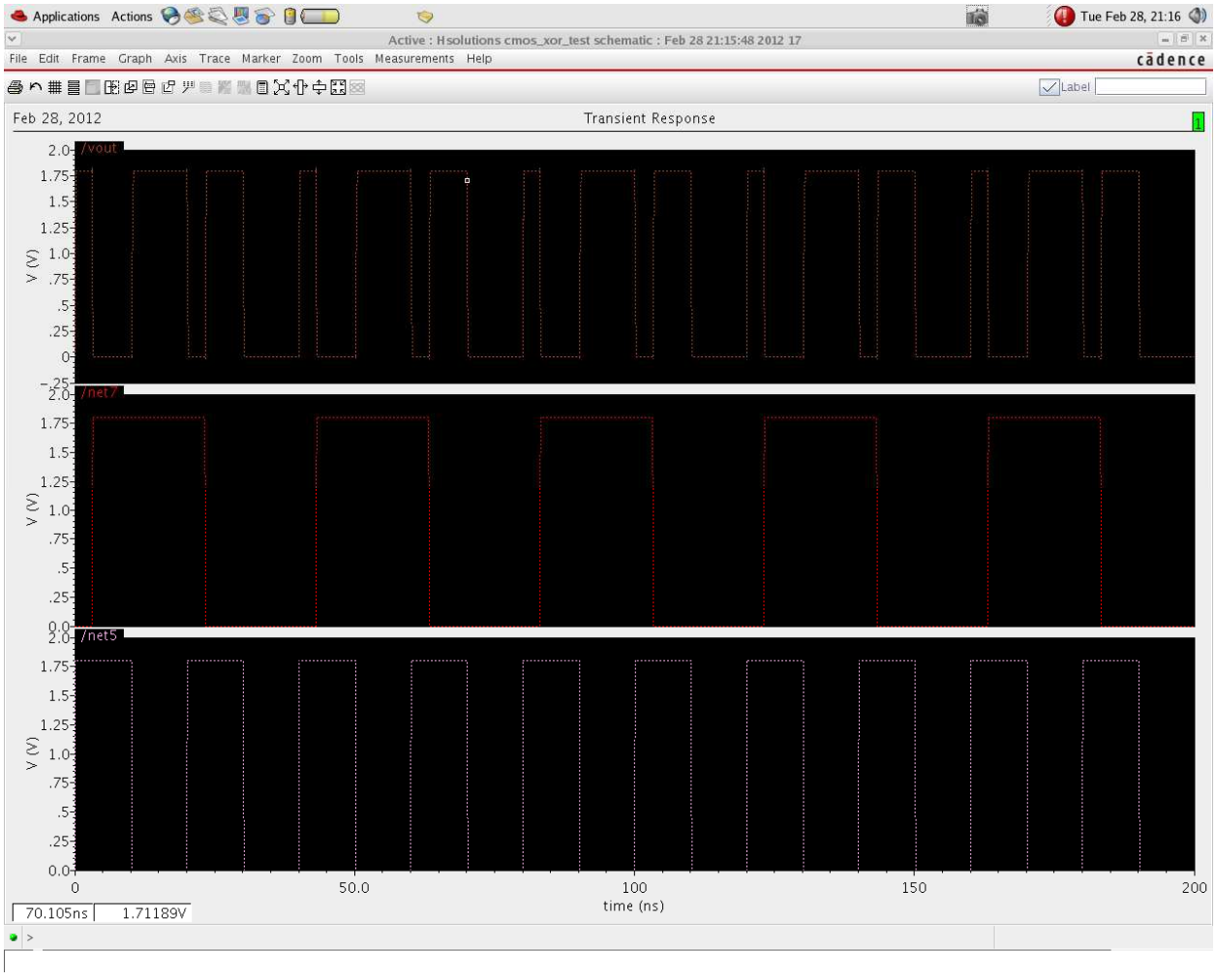
Library name	Cellview name	Properties/Comments
myDesignLib	cmos_XOR	Symbol
analogLib	vpulse	Define pulse specification as In lab 2.1
analogLib	vdd,vss,gnd	vdd=1.8 ; vss= 1.8



Analog Simulation with Spectre

Objective: To set up and run simulations on the XOR gate design.

Use the techniques learned in the Lab2.1 to complete the simulation of XOR gate, ADE window and waveform should look like below.

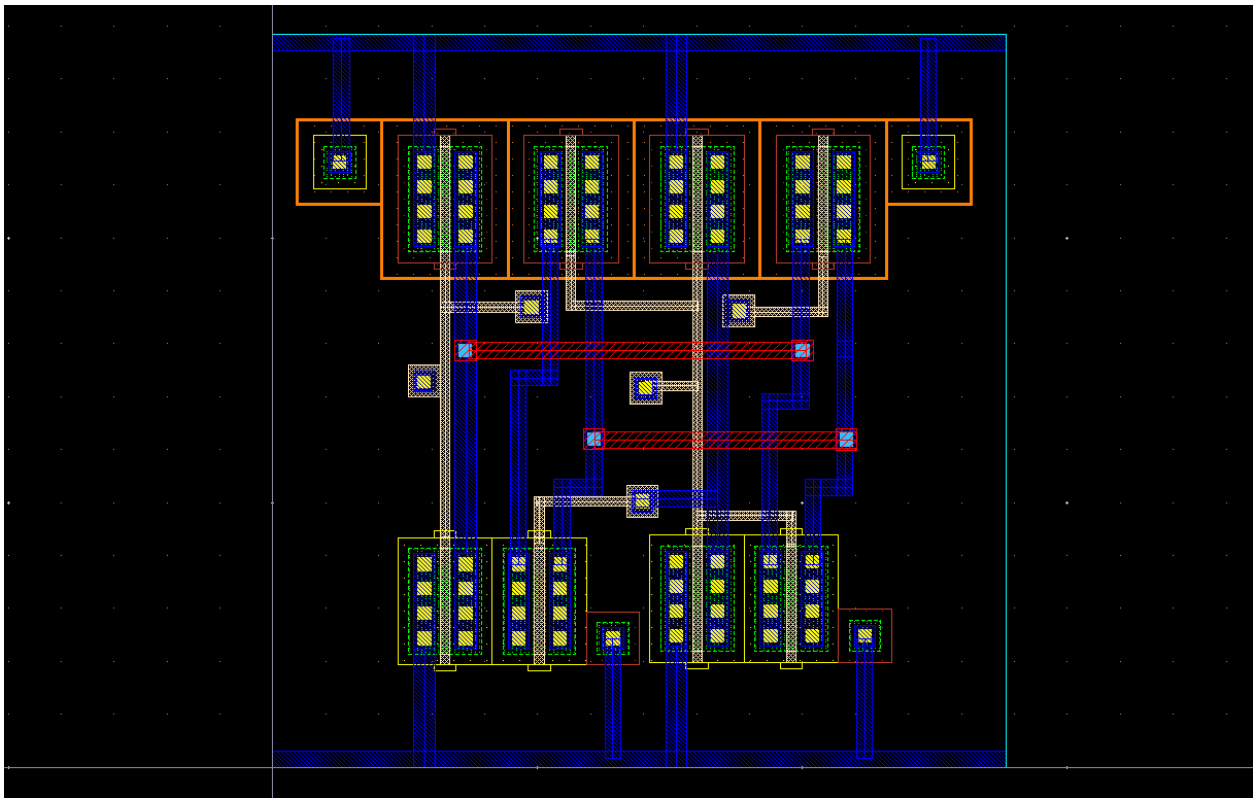


Creating a layout view of XOR gate

Use the techniques learned in the Lab1 and Lab2 to complete the layout of XOR gate.

Complete the DRC, LVS check using the assura tool.

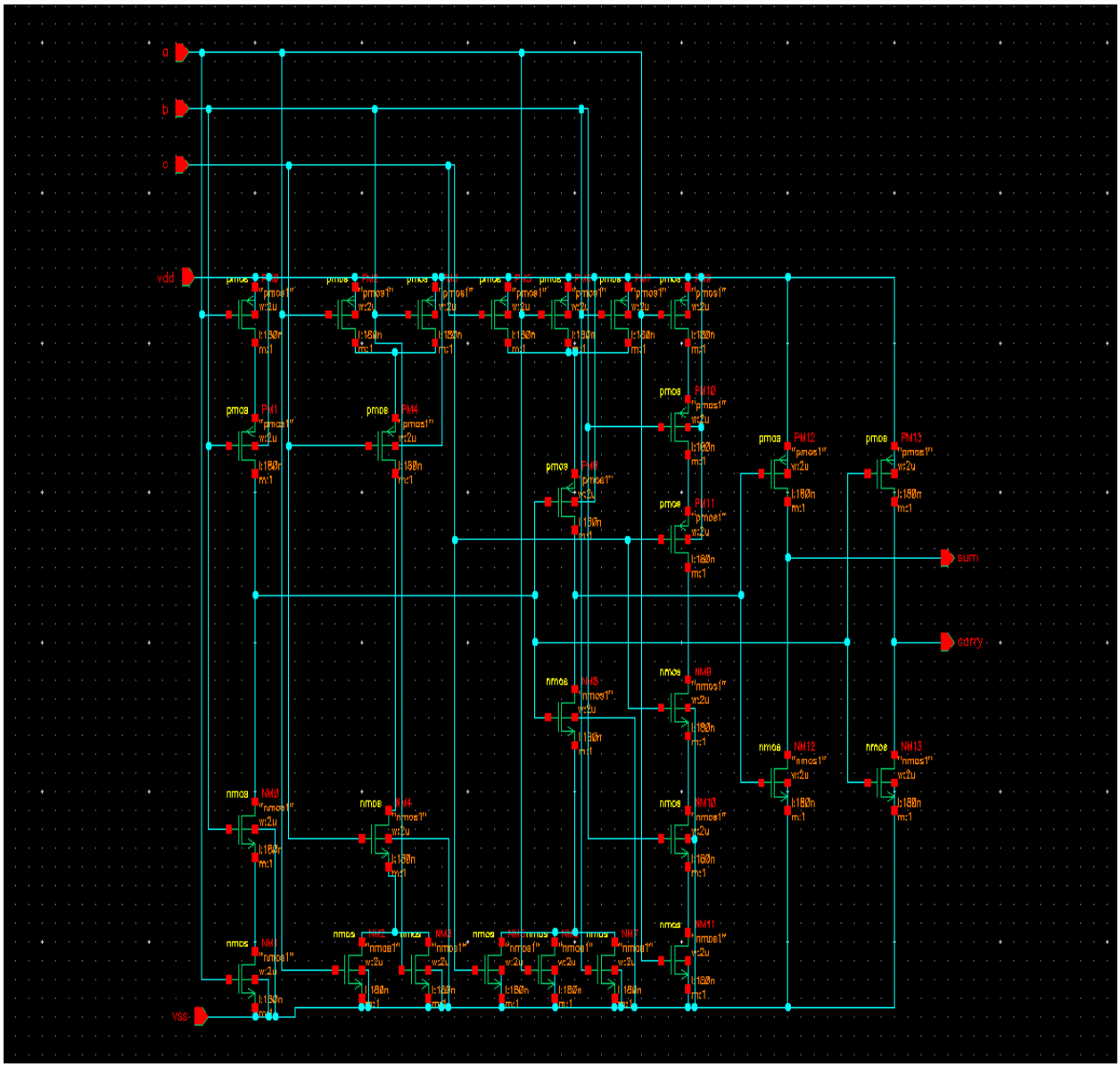
Extract RC parasites for back annotation and Re-simulation.



END OF LAB 2.3

Lab 2.4: A FULL ADDER

Schematic Capture



Schematic Entry

Objective: To create a new cell view and build A FULL ADDER gate

---

Use the techniques learned in the Lab2.1 to complete the schematic of FULL ADDER gate.

This is a table of components for building the FULL ADDER gate schematic.

Library name	Cell Name	Properties/Comments
gpdk180	Pmos	Model Name = pmos1,pmos2,pmos3,pmos4;
gpdk180	Nmos	Model Name =nmos1,nmos2,nmos3,nmos4;

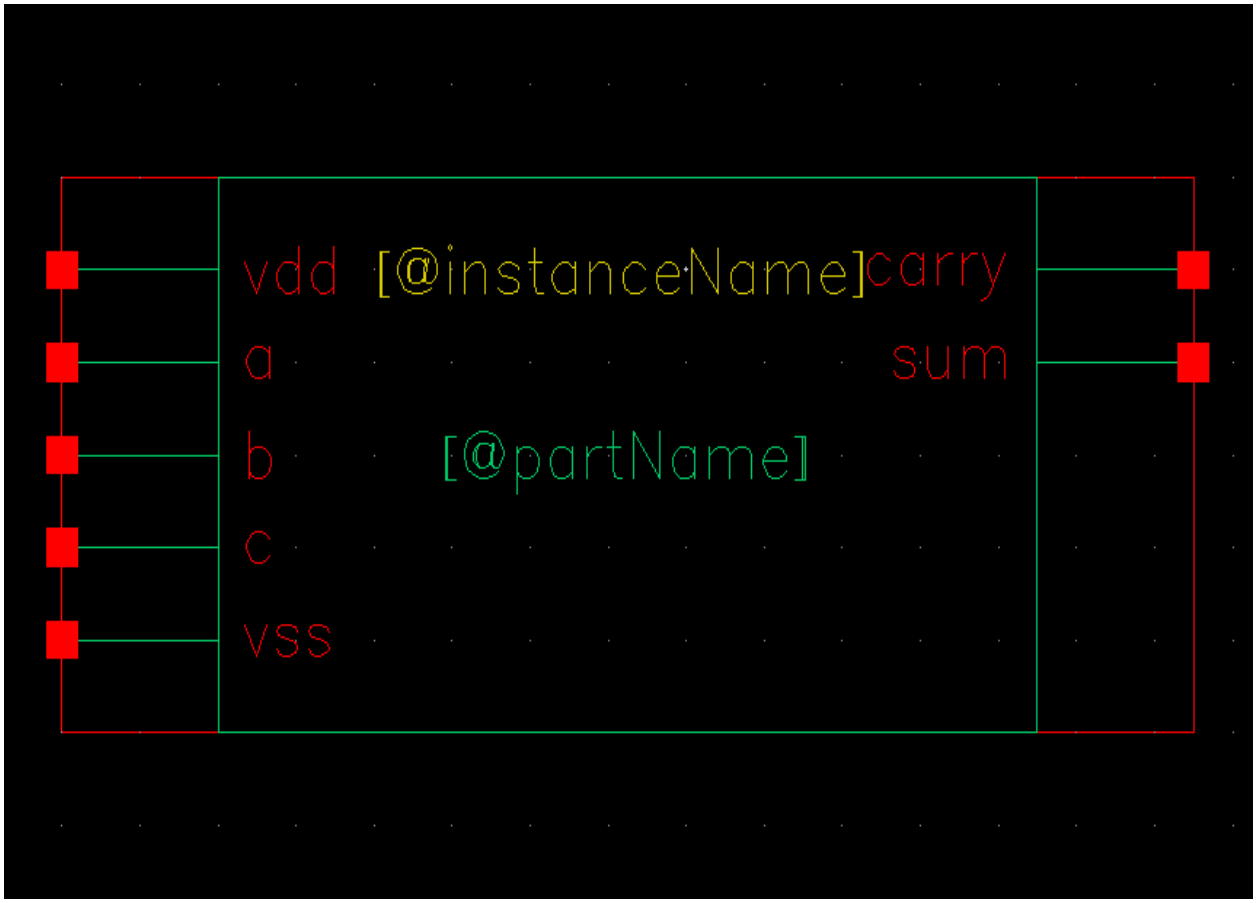
Type the following in the ADD pin form in the exact order leaving space between the pin names.

Pin Names	Direction
Vin1 vin2	Input
vout	Output
vdd vss	Input

Symbol Creation

Objective: To create a symbol for the FULL ADDER

Use the techniques learned in the Lab2.1 to complete the symbol of FULL ADDER



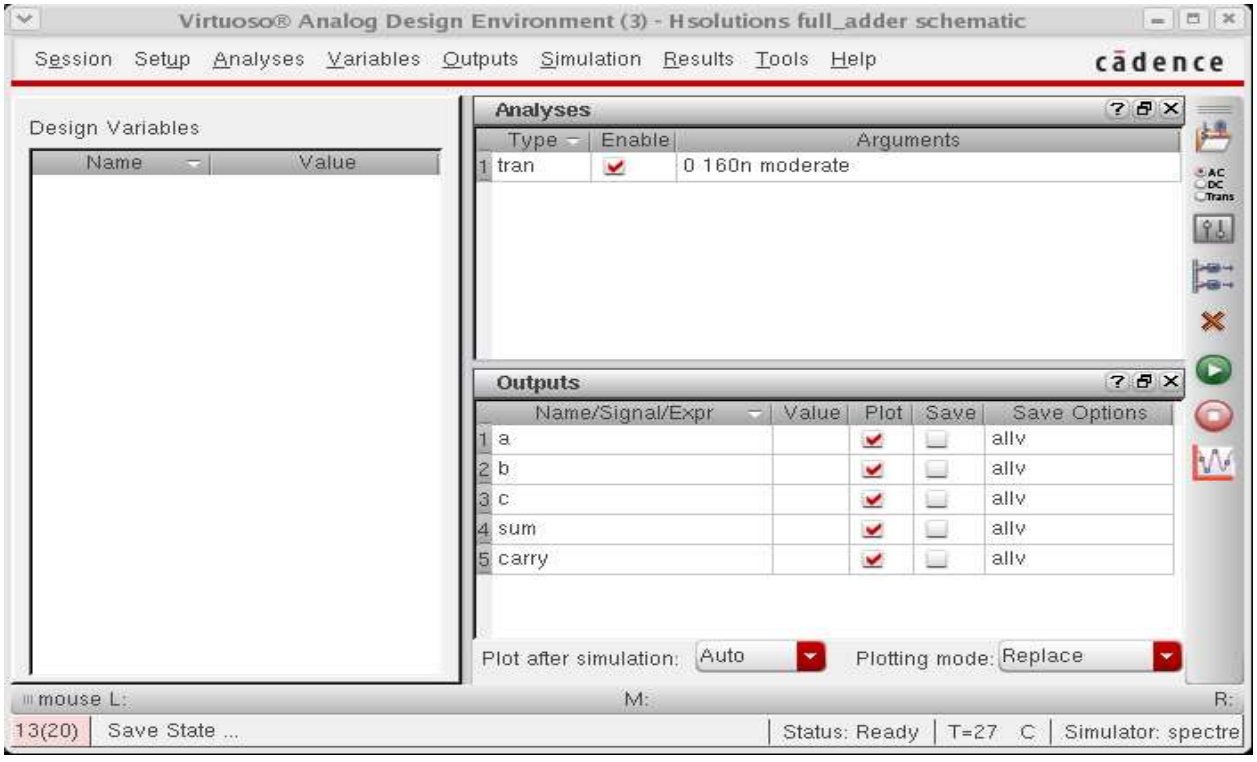
Building the Full Adder Test Design

Objective: To build full_adder_test circuit using your full_adder

Using the component list and Properties/Comments in the table,

build the Full adder_test schematic as shown below.

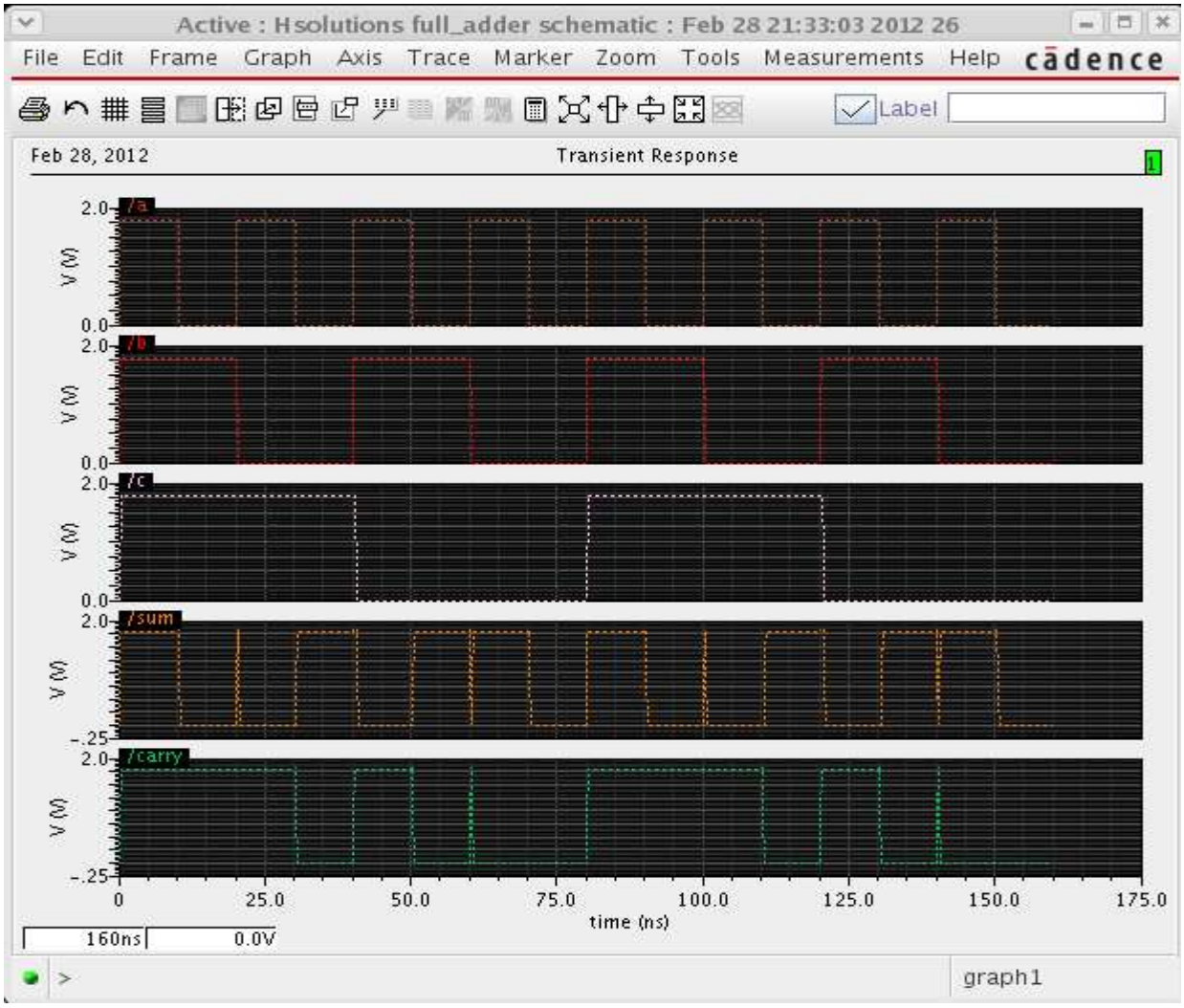
Library name	Cellview name	Properties/Comments
myDesignLib	cmos_FULL ADDER	Symbol
analogLib	vpulse	Define pulse specification as In lab 2.1
analogLib	vdd,vss,gnd	vdd=1.8 ; vss= 1.8



Analog Simulation with Spectre

Objective: To set up and run simulations on the FULL ADDER design.

Use the techniques learned in the Lab2.1 to complete the simulation of FULL ADDER, ADE window and waveform should look like below.

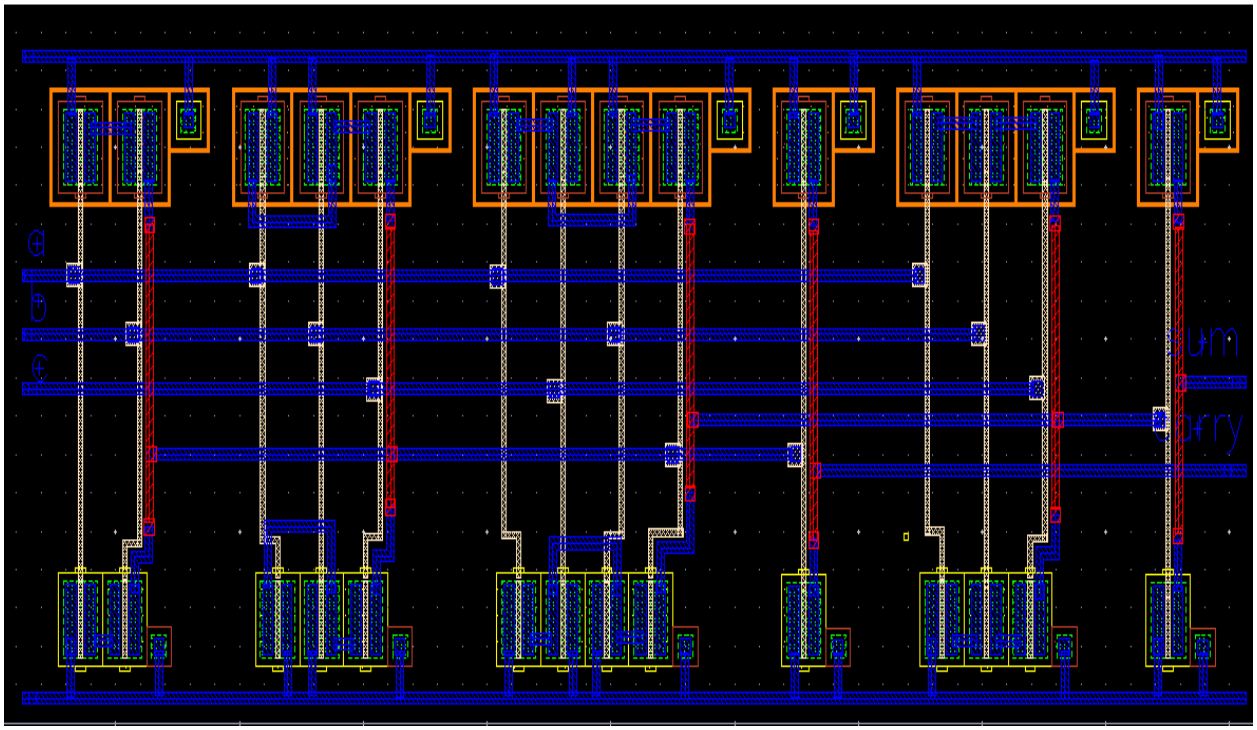


Creating a layout view of FULL ADDER

Use the techniques learned in the Lab1 and Lab2 to complete the layout of FULL ADDER.

Complete the DRC, LVS check using the assura tool.

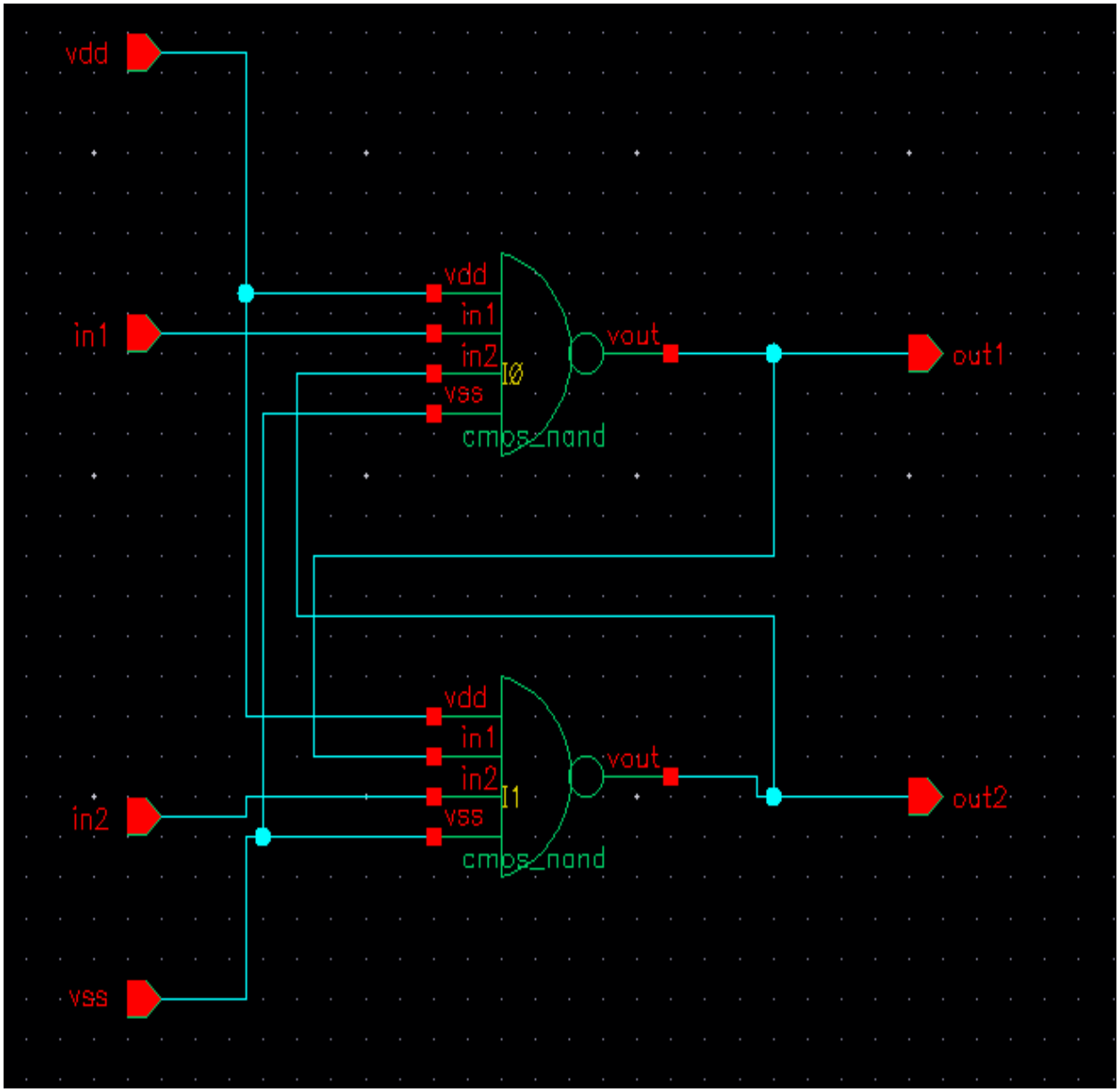
Extract RC parasites for back annotation and Re-simulation.



END OF LAB 2.2

Lab 2.5: A LATCH

Schematic capture



Schematic Entry

Objective: To create a new cell view and build A LATCH

Use the techniques learned in the Lab2.1 to complete the schematic of LATCH.

This is a table of components for building the LATCH schematic.

Library name	Cell Name	Properties/Comments
gpdk180	Pmos	Model Name = pmos1,pmos2,pmos3,pmos4;
gpdk180	Nmos	Model Name =nmos1,nmos2,nmos3,nmos4;

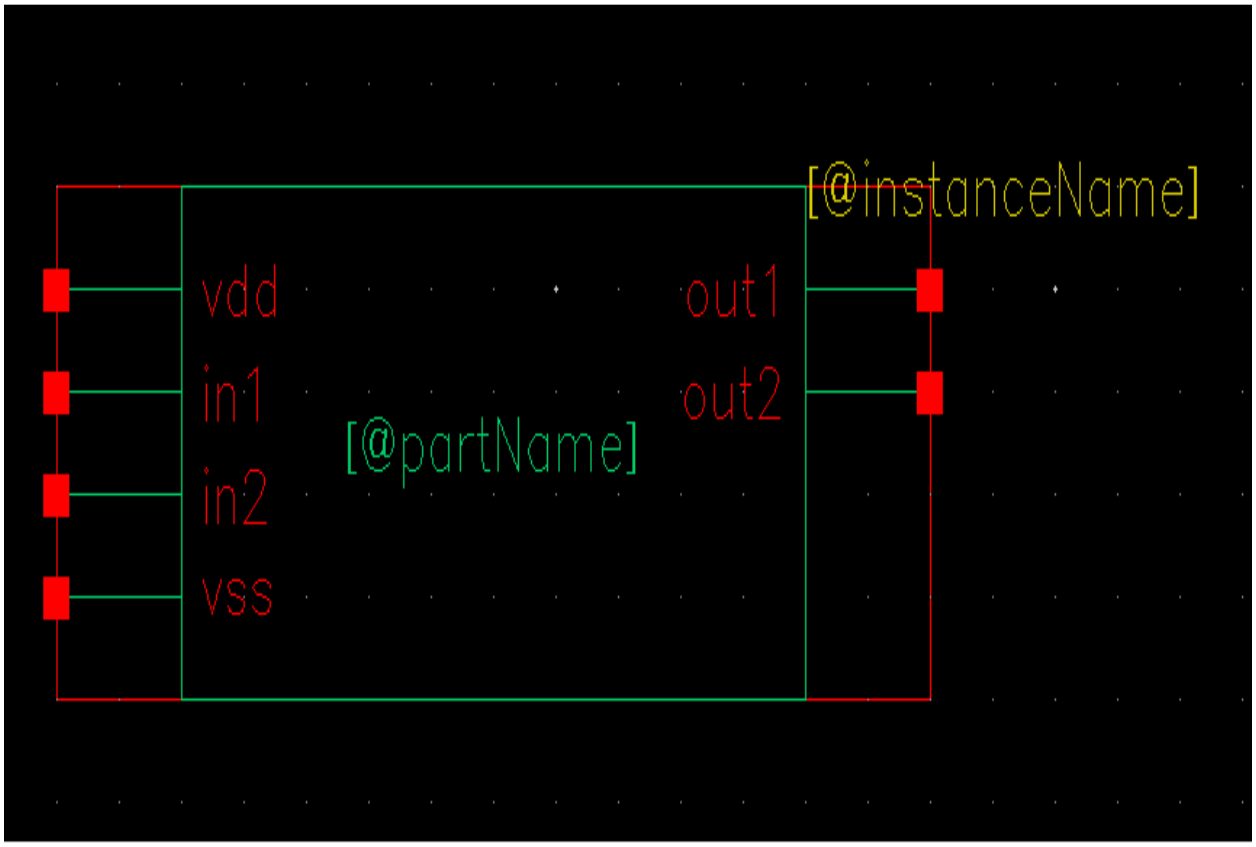
Type the following in the ADD pin form in the exact order leaving space between the pin names.

Pin Names	Direction
Vin1 vin2	Input
vout	Output
vdd vss	Input

Symbol Creation

Objective: To create a symbol for the LATCH

Use the techniques learned in the Lab2.1 to complete the symbol of LATCH



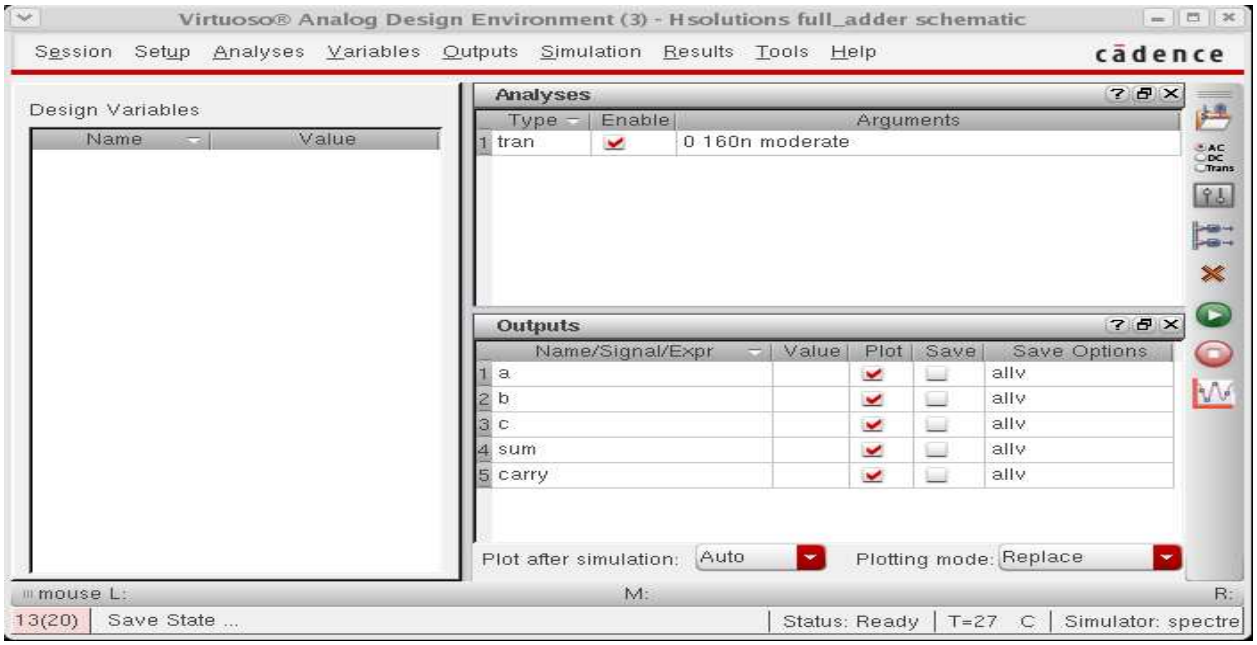
Building the latch Test Design

Objective: To build latch_test circuit using your latch

Using the component list and Properties/Comments in the table,

build the latch_test schematic as shown below.

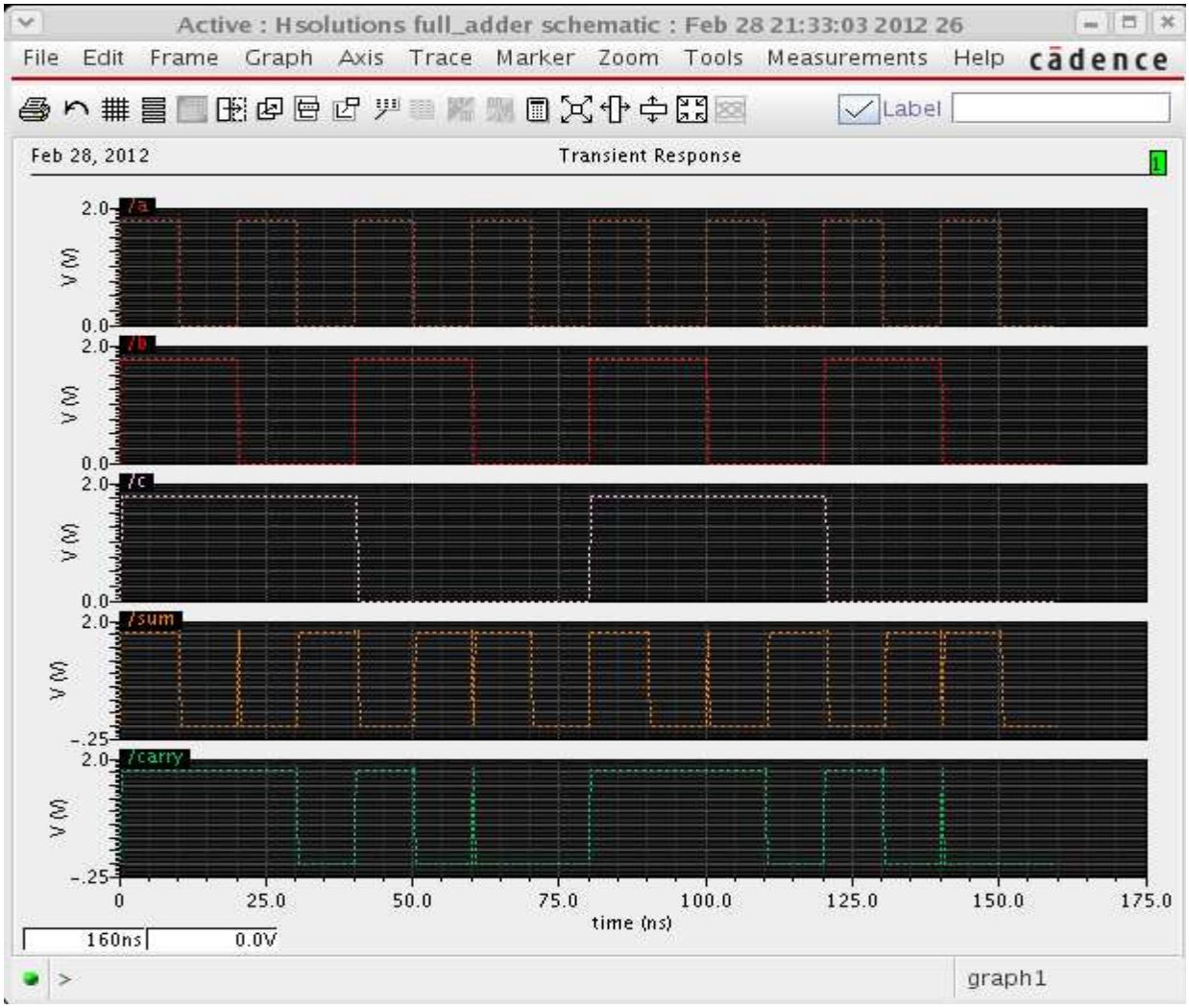
Library name	Cellview name	Properties/Comments
myDesignLib	cmos_LATCH	Symbol
analogLib	vpulse	Define pulse specification as In lab 2.1
analogLib	vdd,vss,gnd	vdd=1.8 ; vss= 1.8



Analog Simulation with Spectre

Objective: To set up and run simulations on the LATCH design.

Use the techniques learned in the Lab2.1 to complete the simulation of LATCH, ADE window and waveform should look like below.

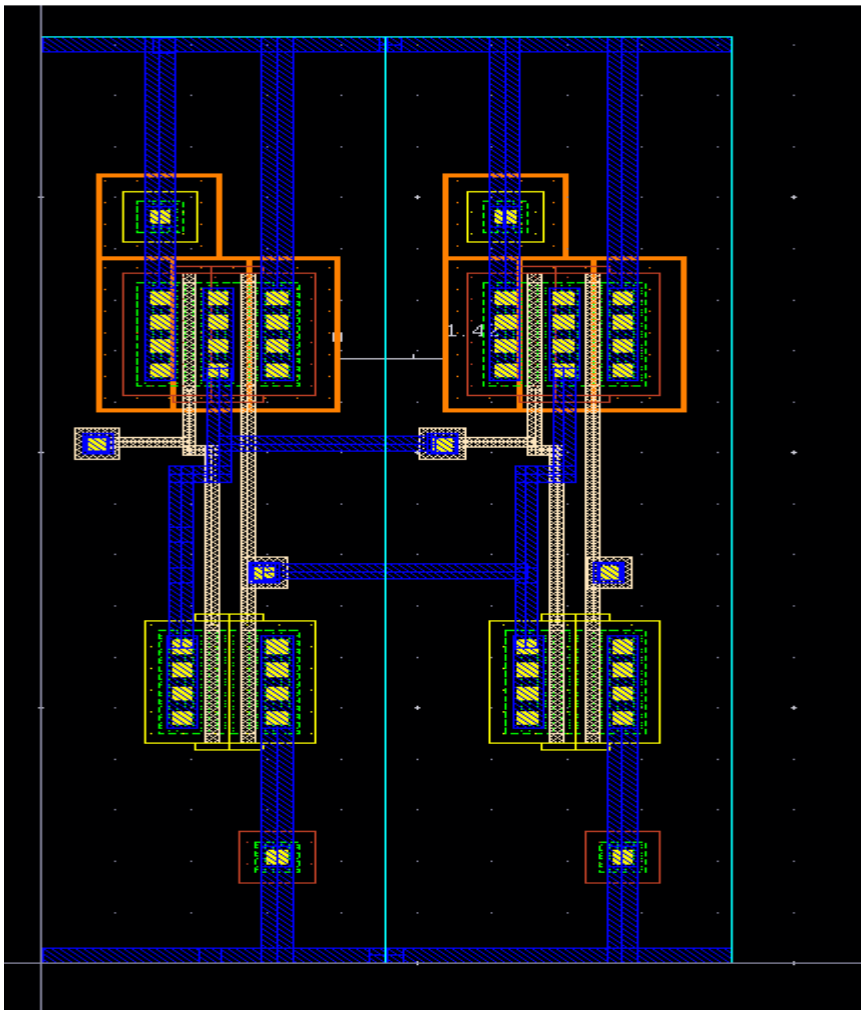


Creating a layout view of LATCH

Use the techniques learned in the Lab1 and Lab2 to complete the layout of LATCH.

Complete the DRC, LVS check using the assura tool.

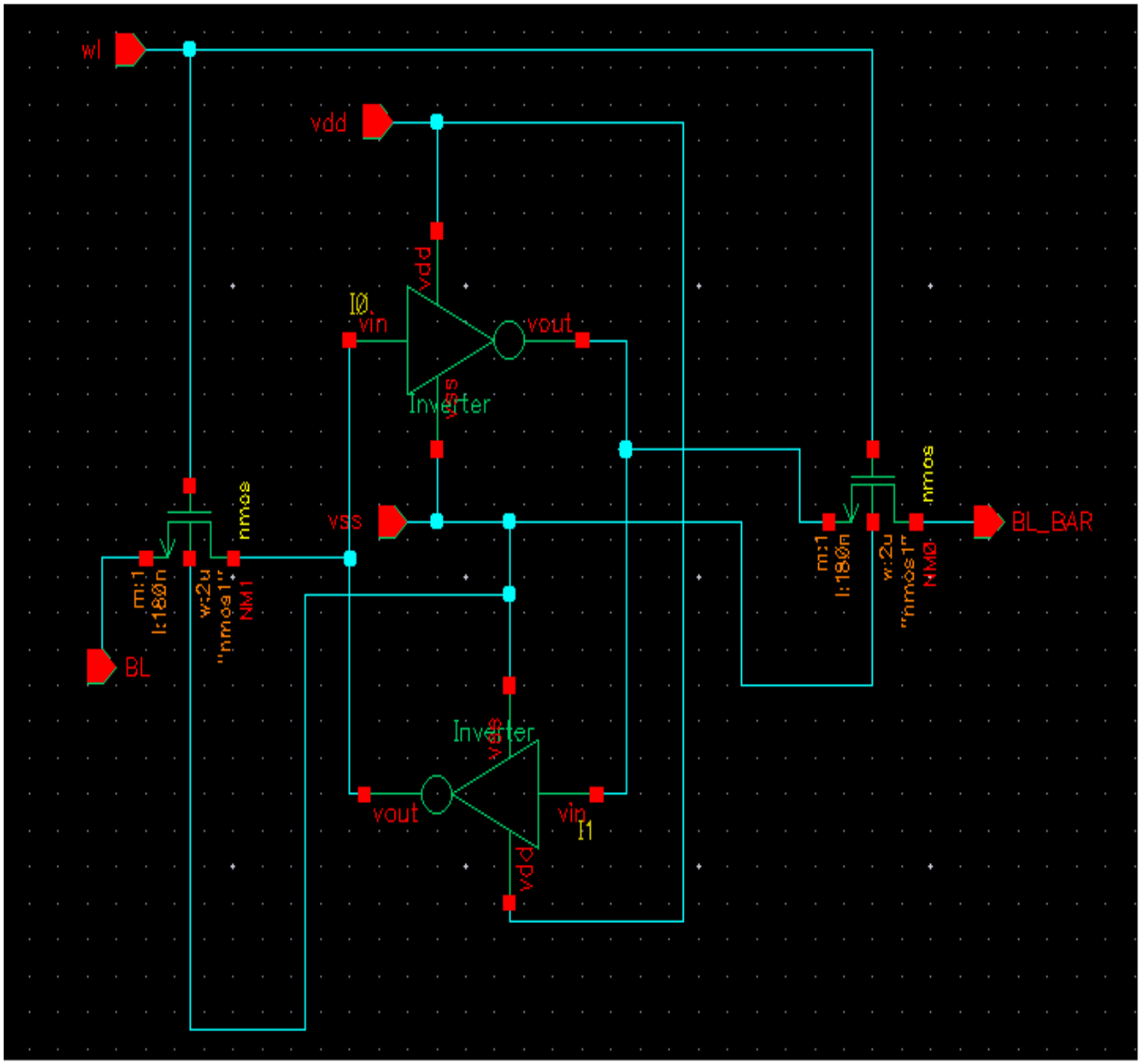
Extract RC parasites for back annotation and Re-simulation.



END OF LAB 2.5

Lab 2.6: A SRAM

Schematic capture



Schematic Entry

Objective: To create a new cell view and build A SRAM

Use the techniques learned in the Lab2.1 to complete the schematic of SRAM.

This is a table of components for building the SRAM schematic.

Library name	Cell Name	Properties/Comments
gpdk180	Pmos	Model Name = pmos1,pmos2,pmos3,pmos4;
gpdk180	Nmos	Model Name =nmos1,nmos2,nmos3,nmos4;

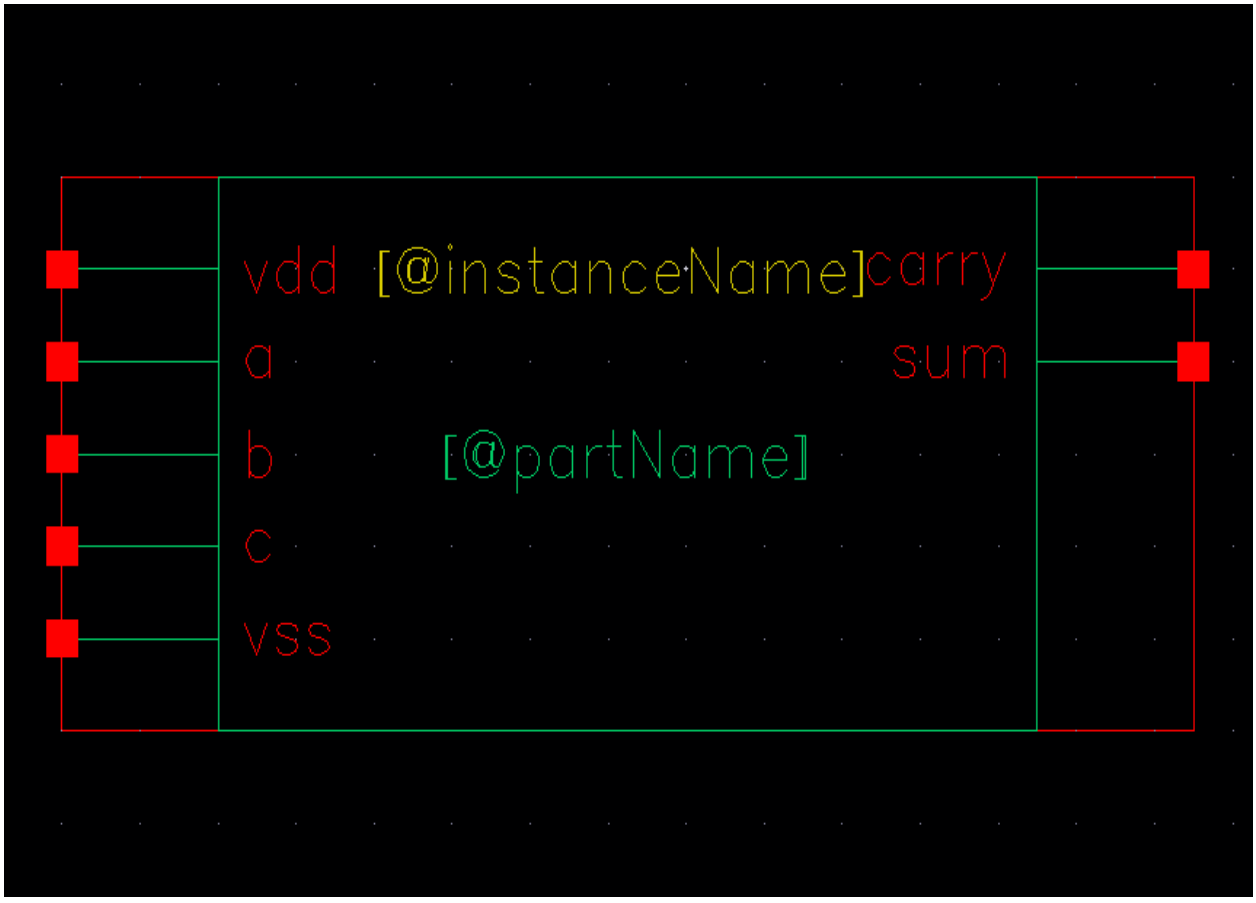
Type the following in the ADD pin form in the exact order leaving space between the pin names.

Pin Names	Direction
Vin1 vin2	Input
vout	Output
vdd vss	Input

Symbol Creation

Objective: To create a symbol for the SRAM

Use the techniques learned in the Lab2.1 to complete the symbol of SRAM

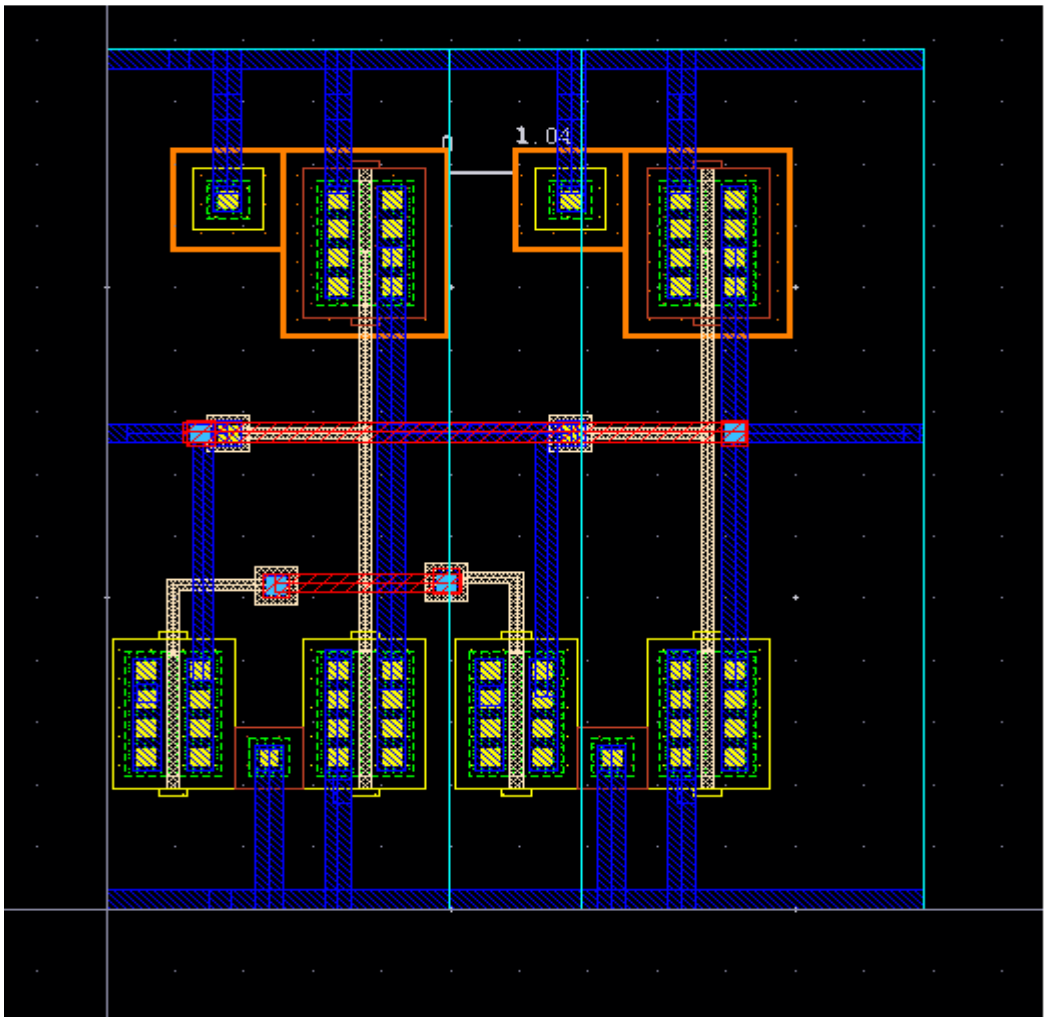


Creating a layout view of SRAM

Use the techniques learned in the Lab1 and Lab2 to complete the layout of SRAM gate.

Complete the DRC, LVS check using the assura tool.

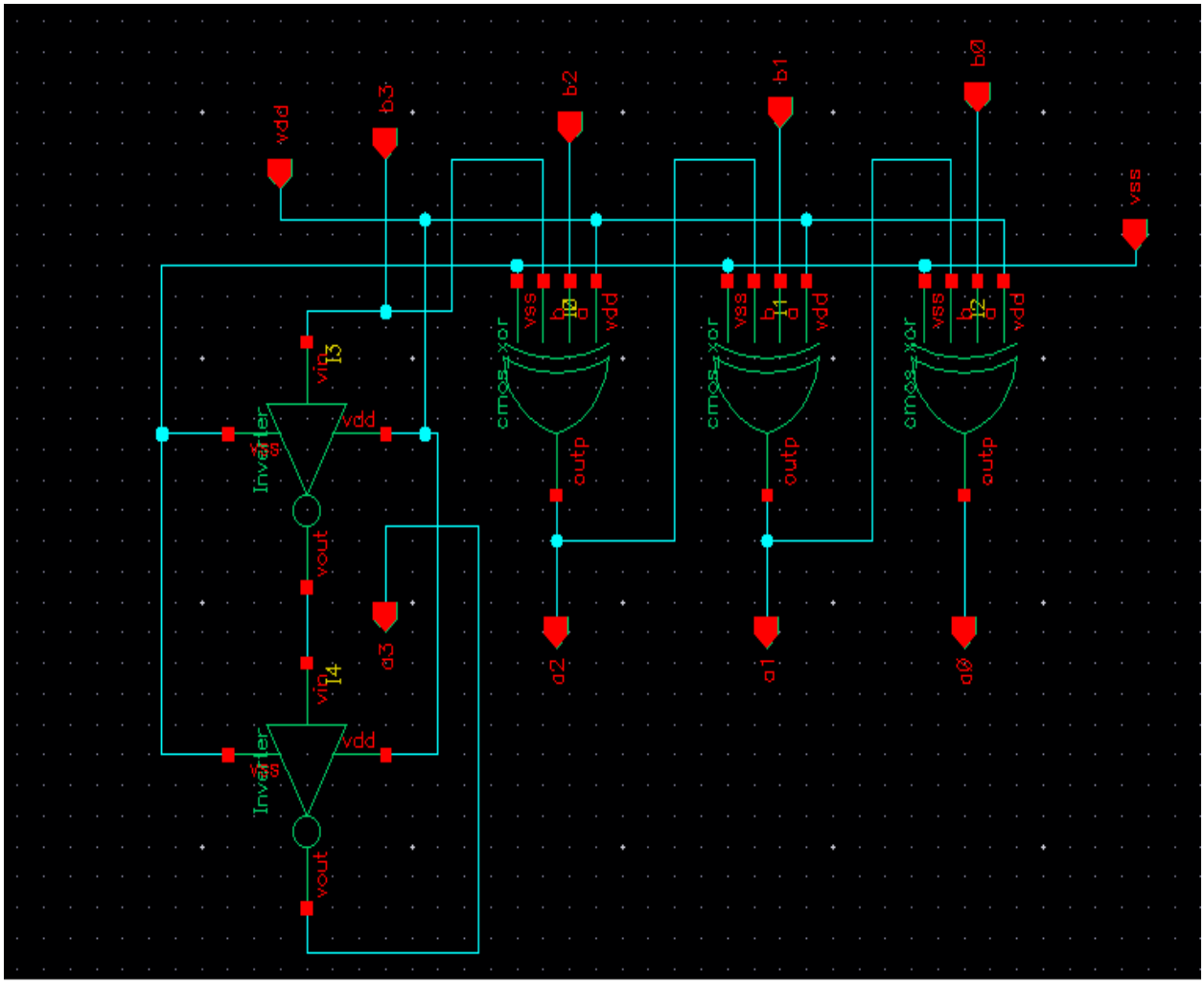
Extract RC parasites for back annotation and Re-simulation.



END OF LAB 2.6

Lab 3: A GREY TO BINARY CODE CONVERTER

Schematic capture



Schematic Entry

Objective: To create a new cell view and build A GREY TO BINARY CODE CONVERTER

Use the techniques learned in the Lab2.1 to complete the schematic of GREY TO BINARY CODE CONVERTER.

This is a table of components for building the GREY TO BINARY CODE CONVERTER schematic.

Library name	Cell Name	Properties/Comments
gpdk180	Pmos	Model Name = pmos1,pmos2,pmos3,pmos4;
gpdk180	Nmos	Model Name =nmos1,nmos2,nmos3,nmos4;

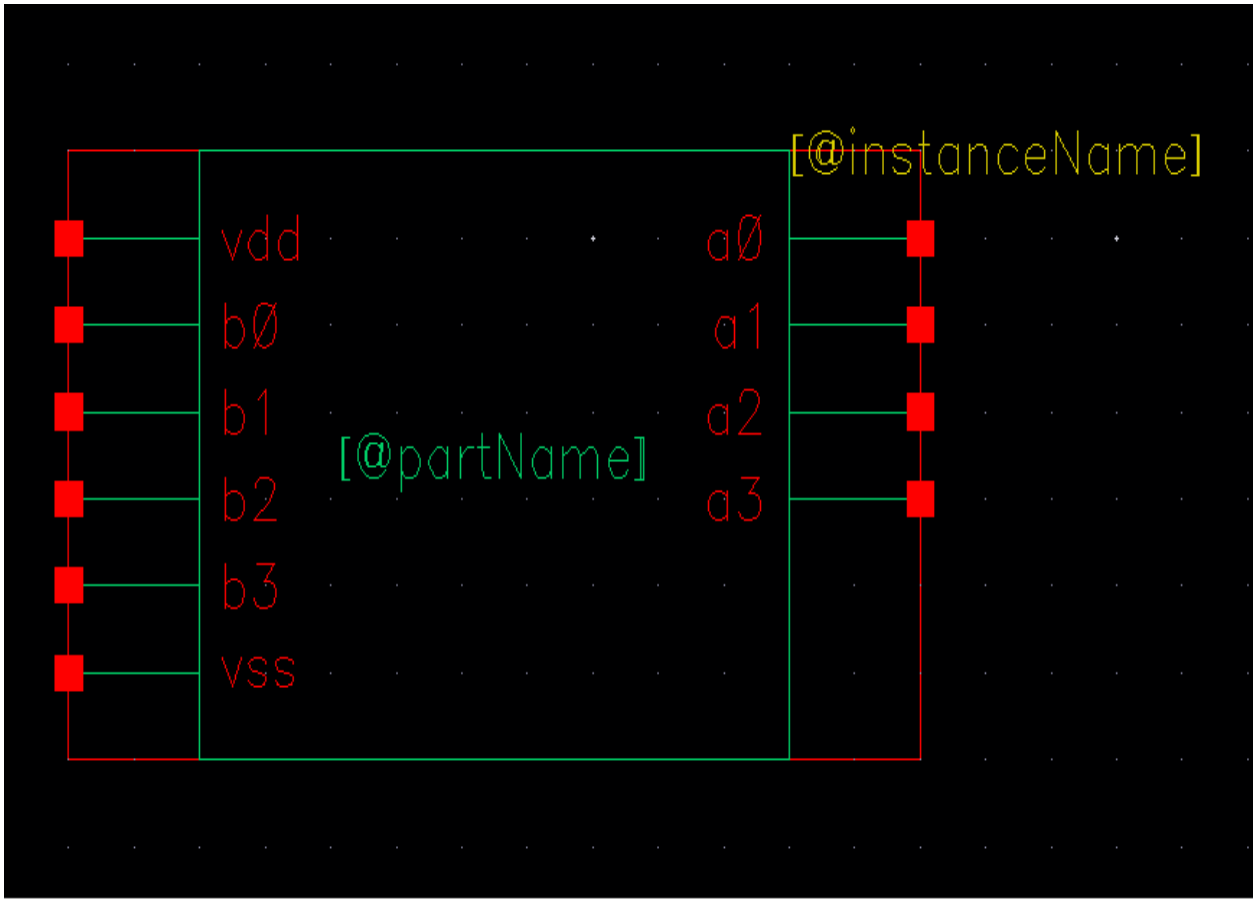
Type the following in the ADD pin form in the exact order leaving space between the pin names.

Pin Names	Direction
Vin1 vin2	Input
vout	Output
vdd vss	Input

Symbol Creation

Objective: To create a symbol for the GREY TO BINARY CODE CONVERTER

Use the techniques learned in the Lab2.1 to complete the symbol of GREY TO BINARY CODE CONVERTER gate

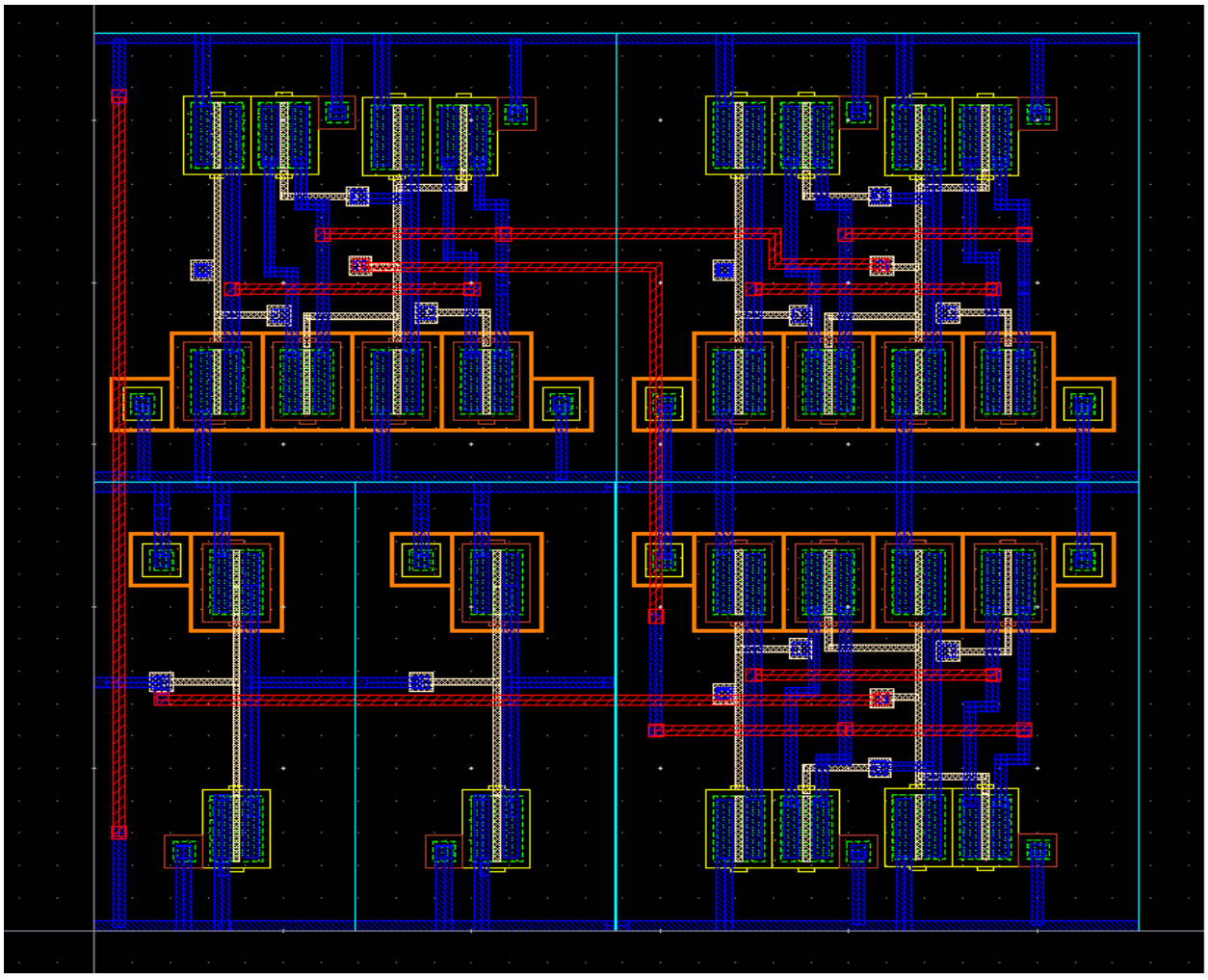


Creating a layout view of GREY TO BINARY CODE CONVERTER

Use the techniques learned in the Lab1 and Lab2 to complete the layout of GREY TO BINARY CODE CONVERTER.

Complete the DRC, LVS check using the assura tool.

Extract RC parasites for back annotation and Re-simulation.



END OF LAB 3

4. Introduction to SPICE Simulation and coding of NMOS/PMOS Circuits.

SPICE is a powerful general purpose analog circuit simulator that is used to verify circuit designs and to predict the circuit behavior. This is of particular importance for integrated circuits. It was for this reason that SPICE was originally developed at the Electronics Research Laboratory of the University of California, Berkeley (1975), as its name implies:

Simulation **P**rogram **f**or **I**ntegrated **C**ircuits **E**mphasis.

A SPICE input file, called source file, consists of three parts.

1. Data statements: description of the components and the interconnections.

Ex: Voltage source statement: Vname N1 N2 Value Type

V1 (2 0) vsource dc=10 type=dc

N1 is the positive terminal node

N2 is the negative terminal node

Type can be DC, AC or TRAN, depending on the type of analysis

Value gives the value of the source

2. Control statements: Tells SPICE what type of analysis to perform on the circuit.

Ex: DC Statement: This statement allows you to increment (sweep) an independent source over a

Certain range with a specified step. The format is as follows:

DC SRCname START STOP STEP

dc dc dev=V1 param=dc start=0 stop=1.8 oppoint=rawfile maxiters=150 maxsteps=10000 annotate=status

In which SRC name is the name of the source you want to vary; START and STOP are the starting

And ending value, respectively; and STEP is the size of the increment.

3. Output statements: specifies what outputs are to be printed or plotted.

Ex: Save vout vin

Coding of MOS:

The MOS transistor name (Nname) has to start with a M; ND, NG, NS and NB are the node numbers of the Drain, Gate, Source and Bulk terminals, respectively. ModName is the name of the transistor model. L and W are the length and width of the gate.

Spice code of NMOS: NMO (vout vin 0 0) nmos1 w= (2u) l=180n as=1.2p ad=1.2p ps=5.2u pd=5.2u

m= (1)*(1)

Spice code of PMOS: PMO (vout vin vdd vdd) pmos1 w=(2u) l=180n as=1.2p ad=1.2p ps=5.2u pd=5.2u m=(1)*(1)

In which ad and as are the areas of source and drain diffusion.

pd and ps are the value of the perimeter of the source and drain.

End of lab 4

Lab:5.1 SPICE Simulation of Basic Analog Circuits: Inverter.

A) Inverter

1. Log in to your workstation using the username and password.

The home directory has a cshrc file with paths to the Cadence installation.

2. In a terminal window, type csh at the command prompt to invoke the C shell.

>csh

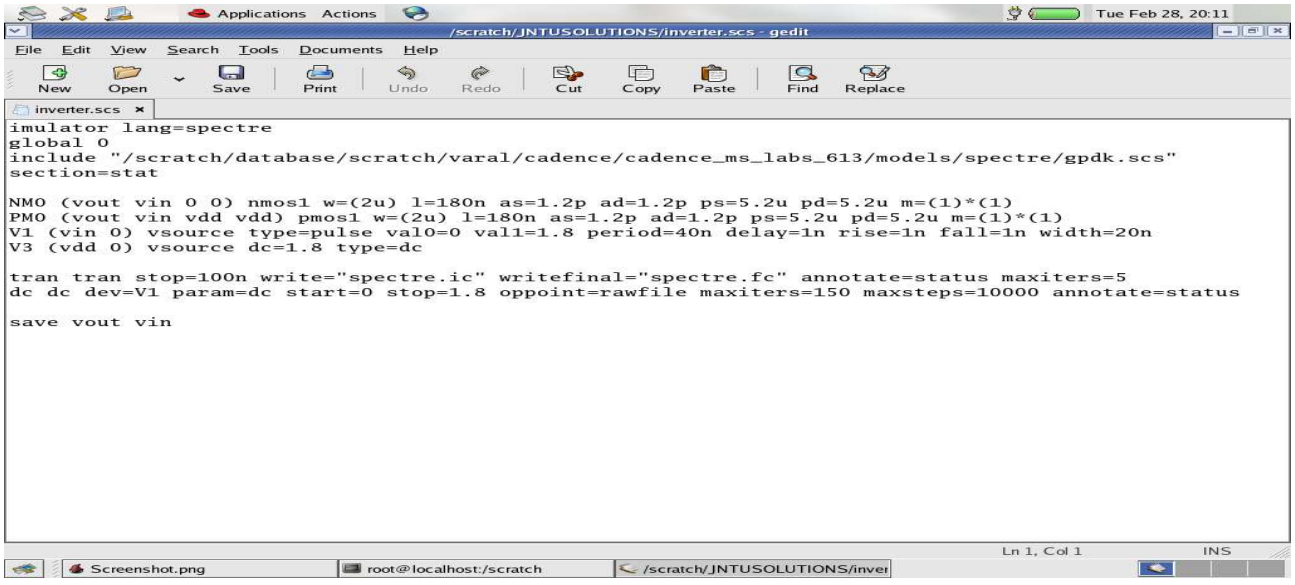
>source cshrc

3. Change to the course directory by entering this command:

> cd ~/Database/cadence_analog_labs_613

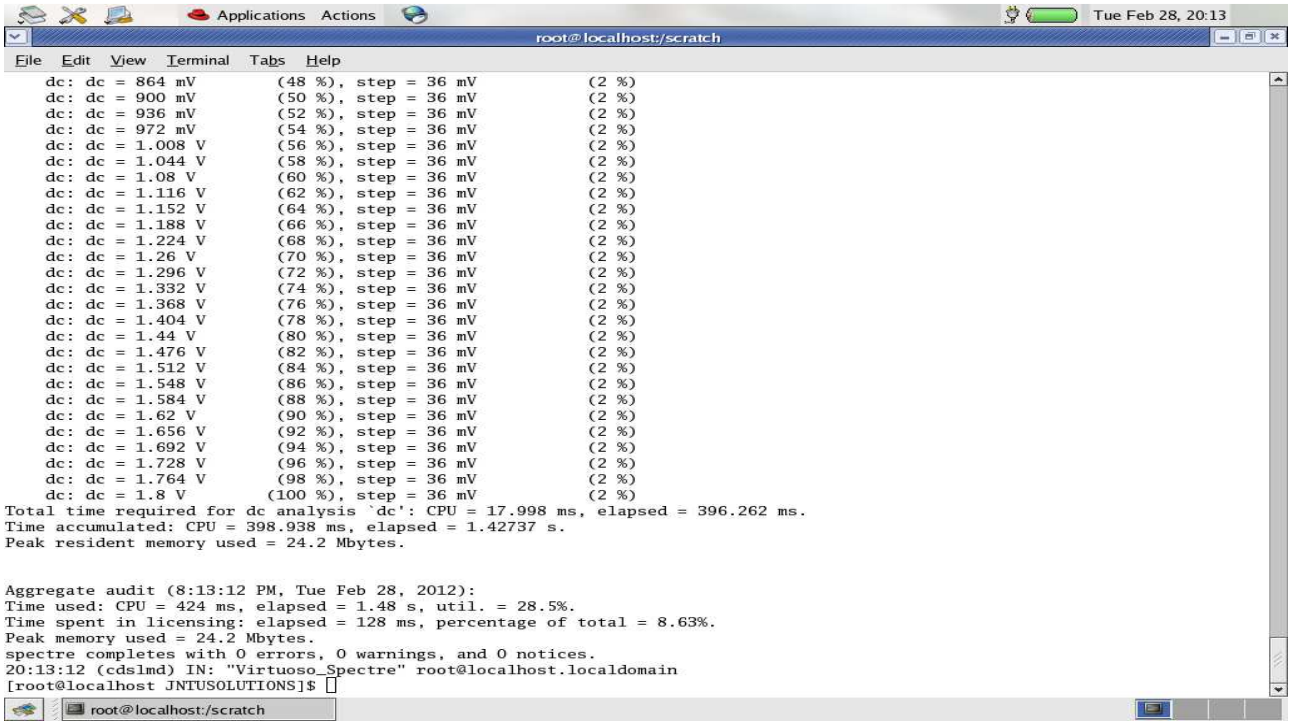
4. In the same terminal window, observe the code of inverter and close.

> gedit inverter.scs



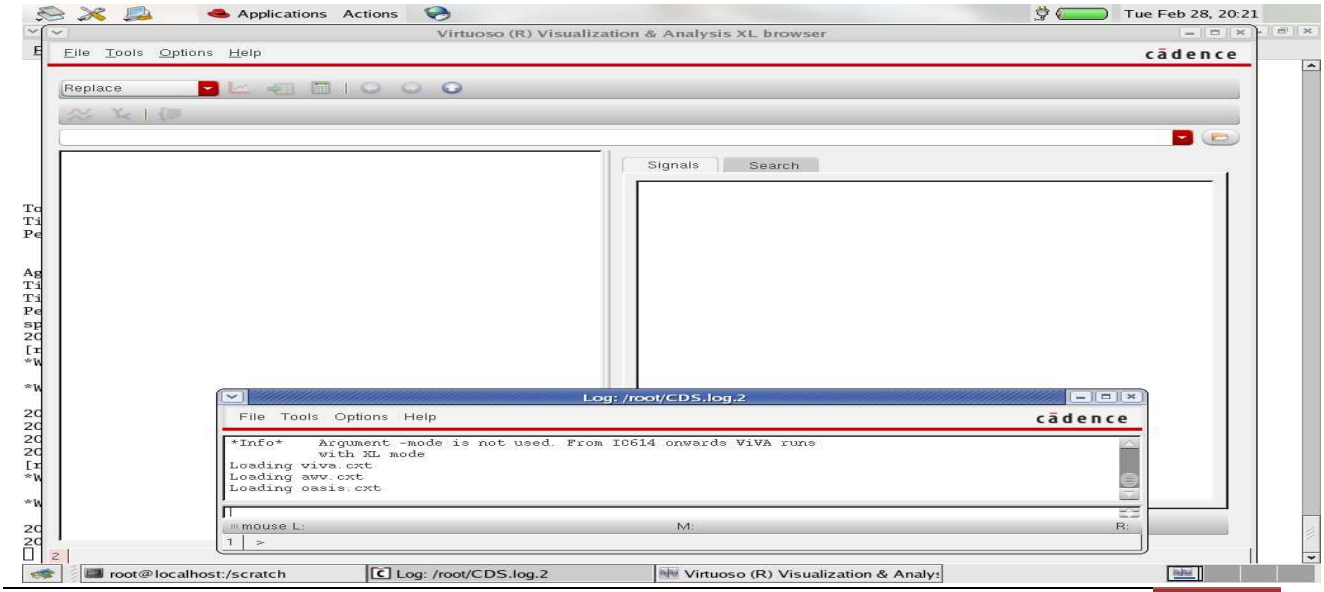
5. Compile the inverter.scs code

>spectre inverter.scs

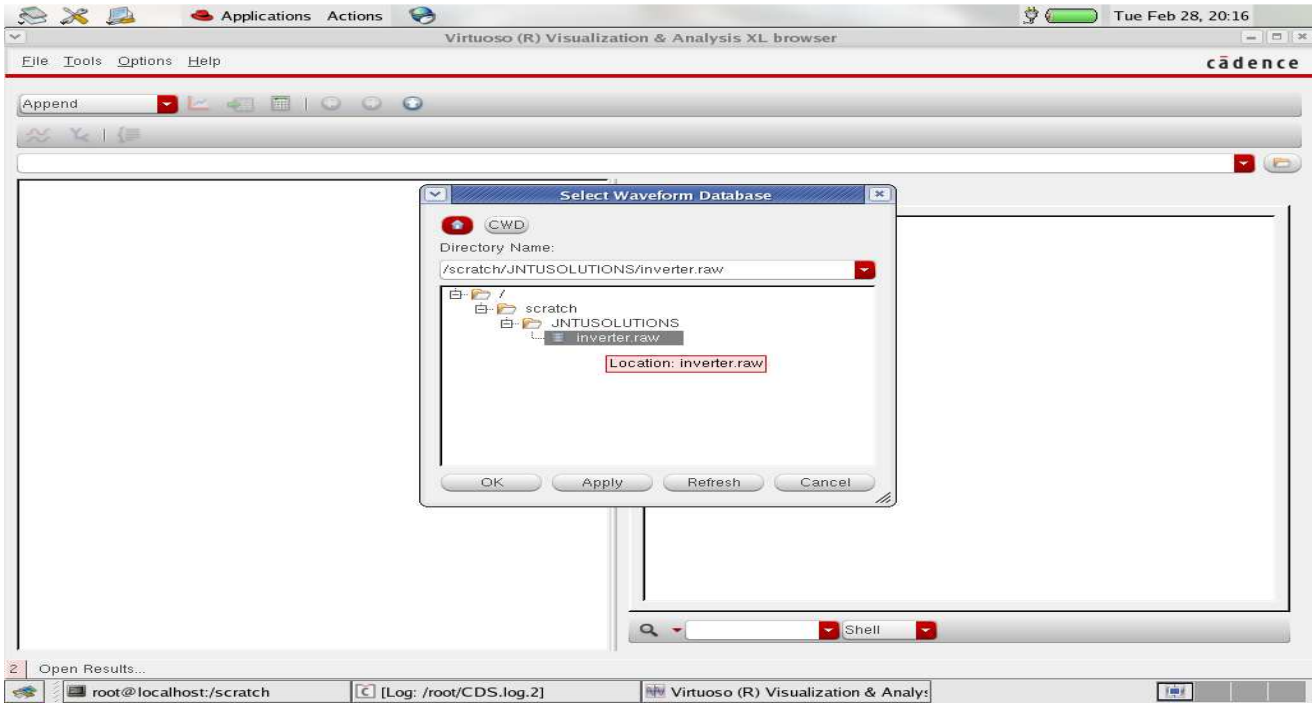


6. Open Virtuoso visulization & Analysis browser.

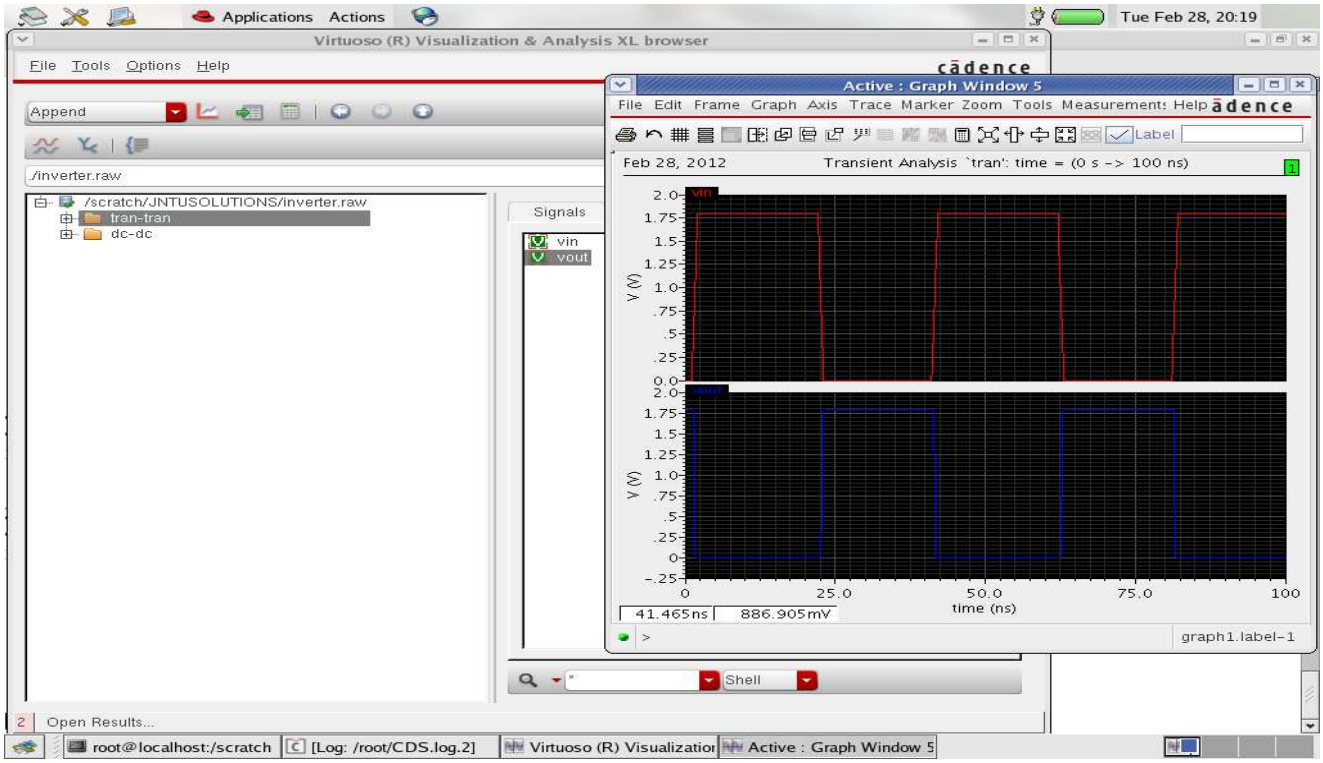
>viva inverter.scs



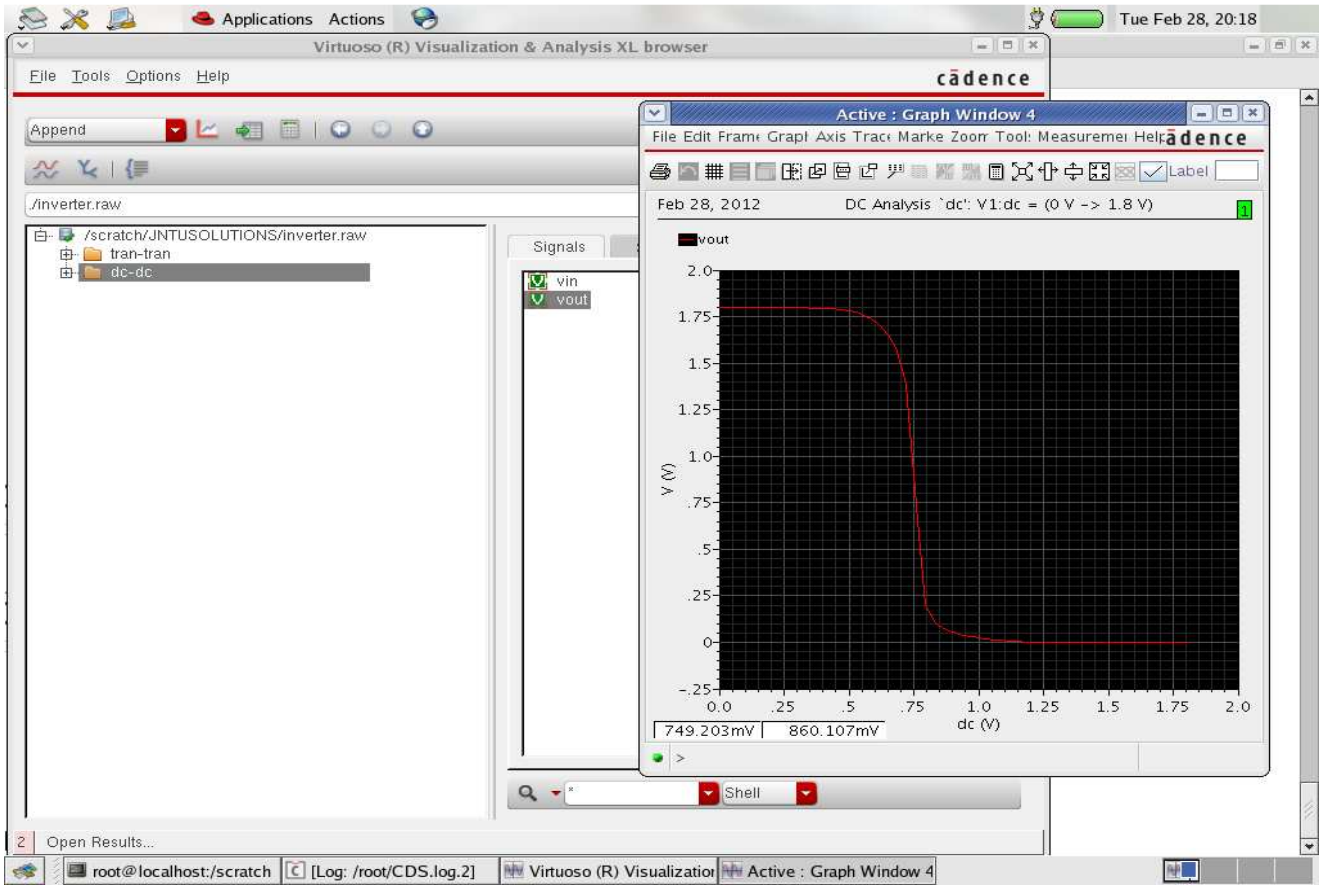
7. Examine the visulization & Analysis browser. a) Select the Append and open the File /Database/cadence_analog_labs_613/inverter.raw



8) For transient analysis: Click on trans-trans , vout and vin



9) For DC analysis: Click on dc-dc, vout.



End of lab 5.1

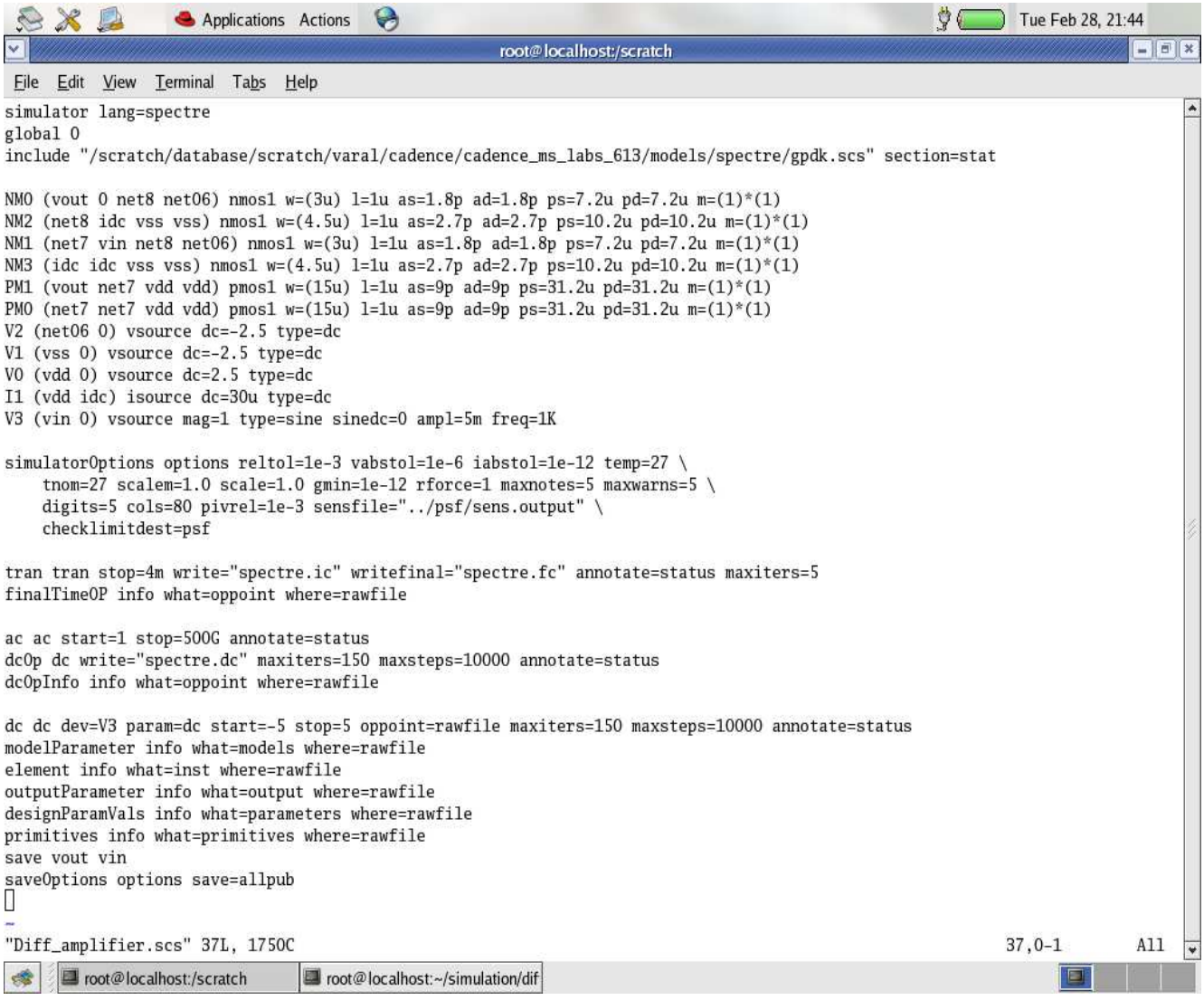
Lab:5.2 SPICE Simulation of Basic Analog Circuits: Differential amplifier

1) Change to the course directory by entering this command:

> cd ~/Database/cadence_analog_labs_613

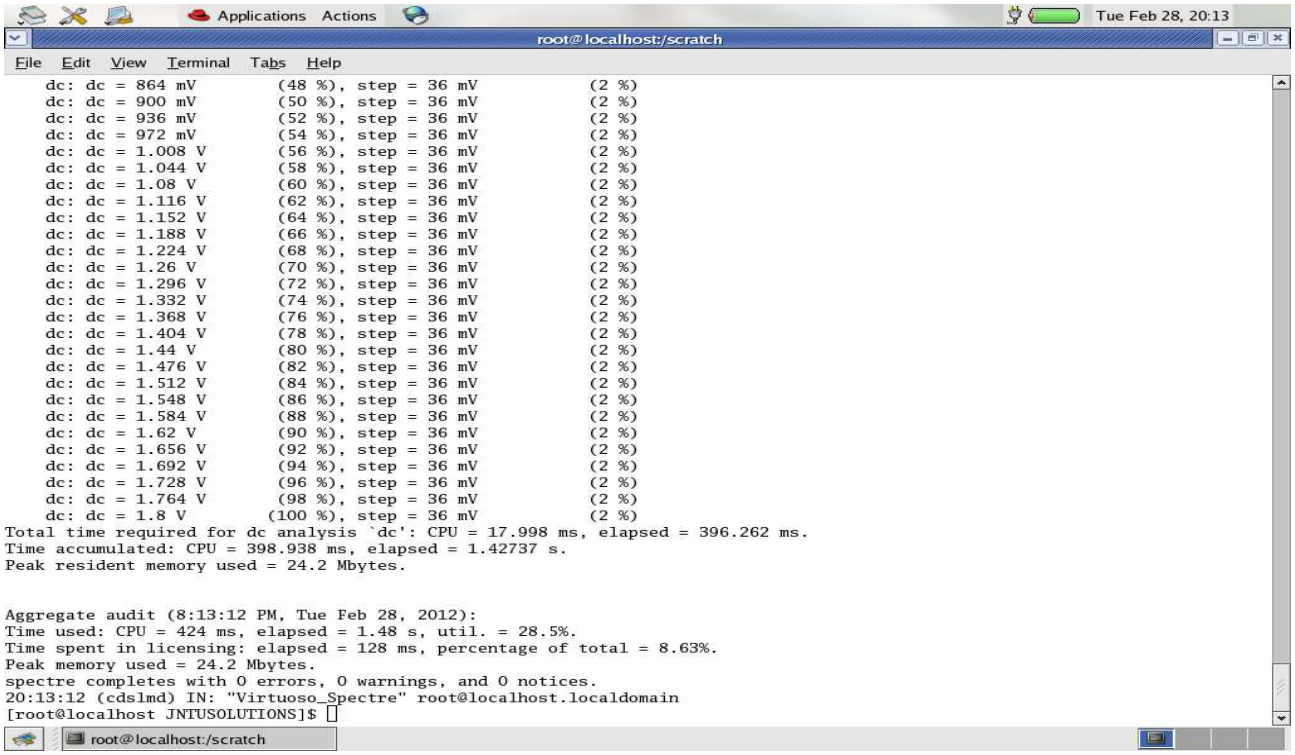
2) In the same terminal window, observe the spice code of differential amplifier and close.

> gedit Diff_amplifier.scs



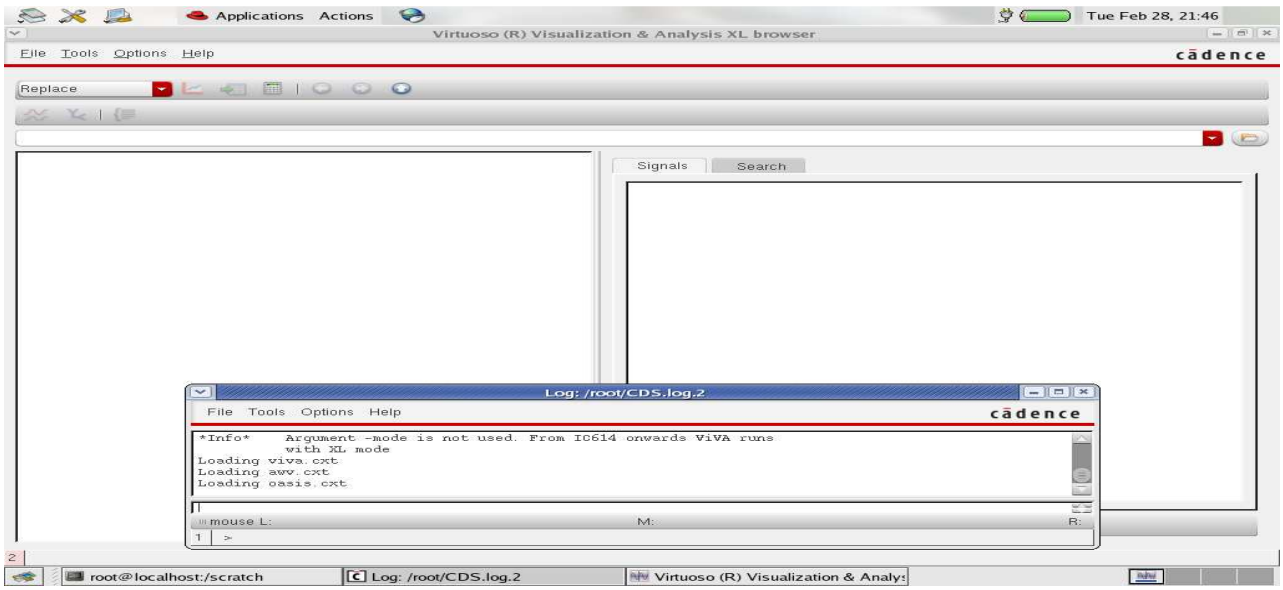
3. Compile the inverter.scs code

>spectre Diff_amplifier.scs

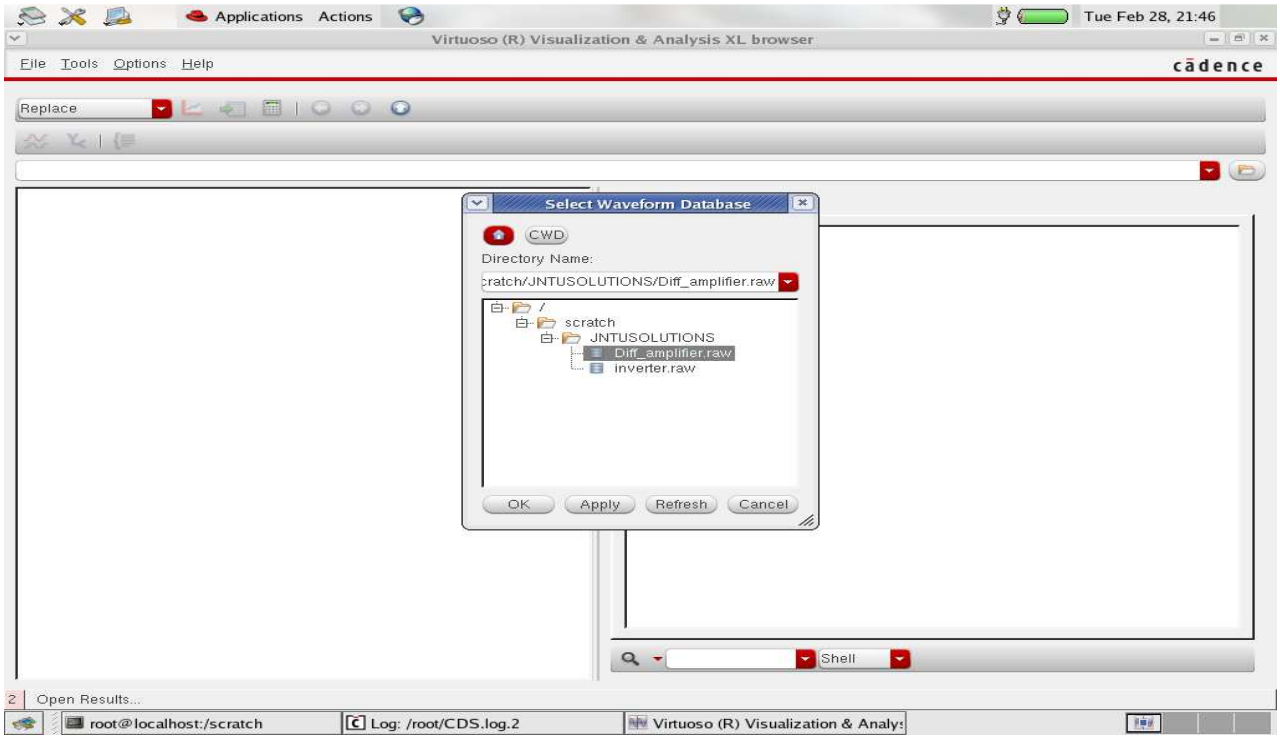


4. Open Virtuoso visulization & Analysis browser.

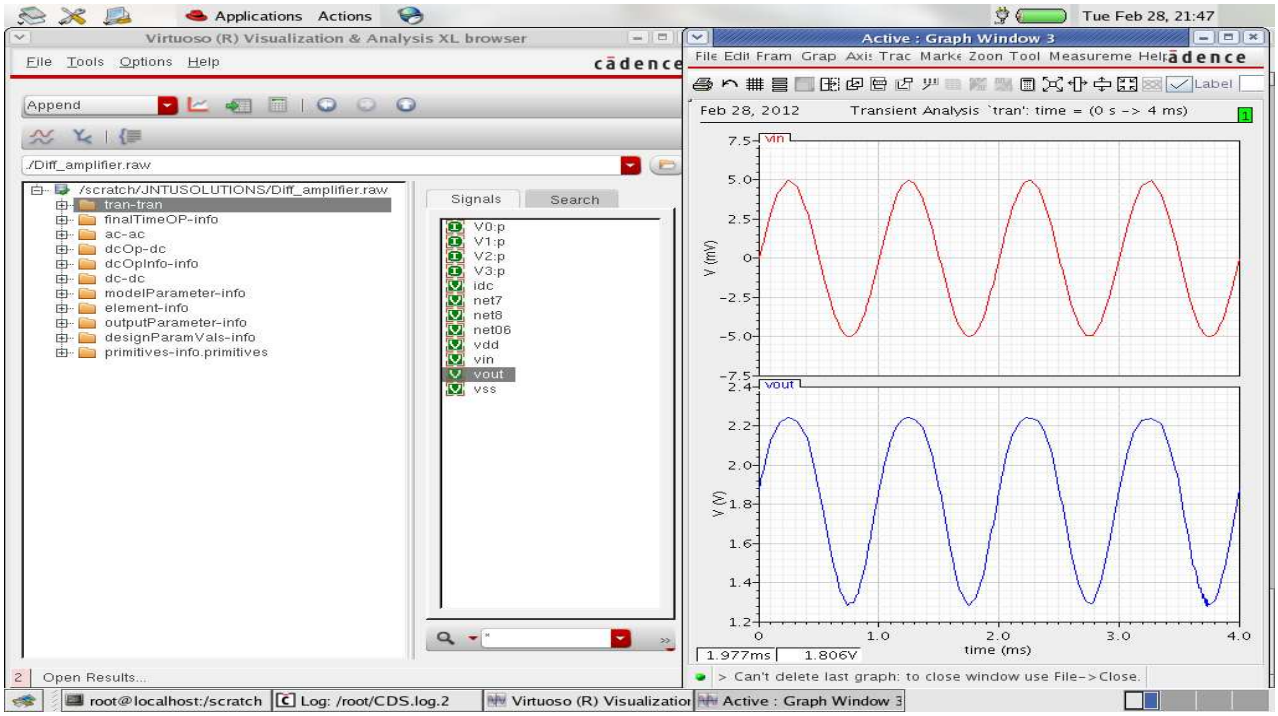
>viva Diff_amplifier.scs



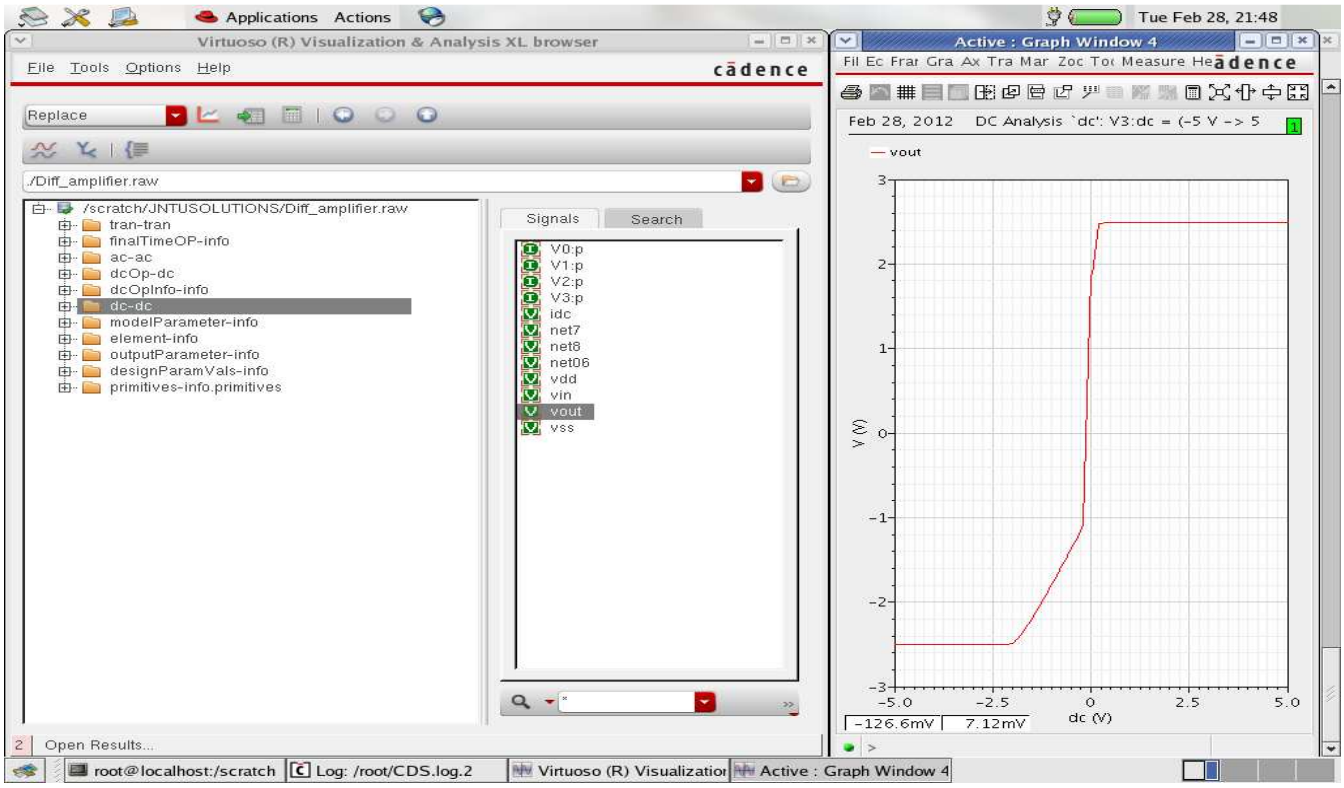
5. Examine the visulization & Analysis browser. a) Select the Append and open the /Database/cadence_analog_labs_613/Diff_amplifier.raw file



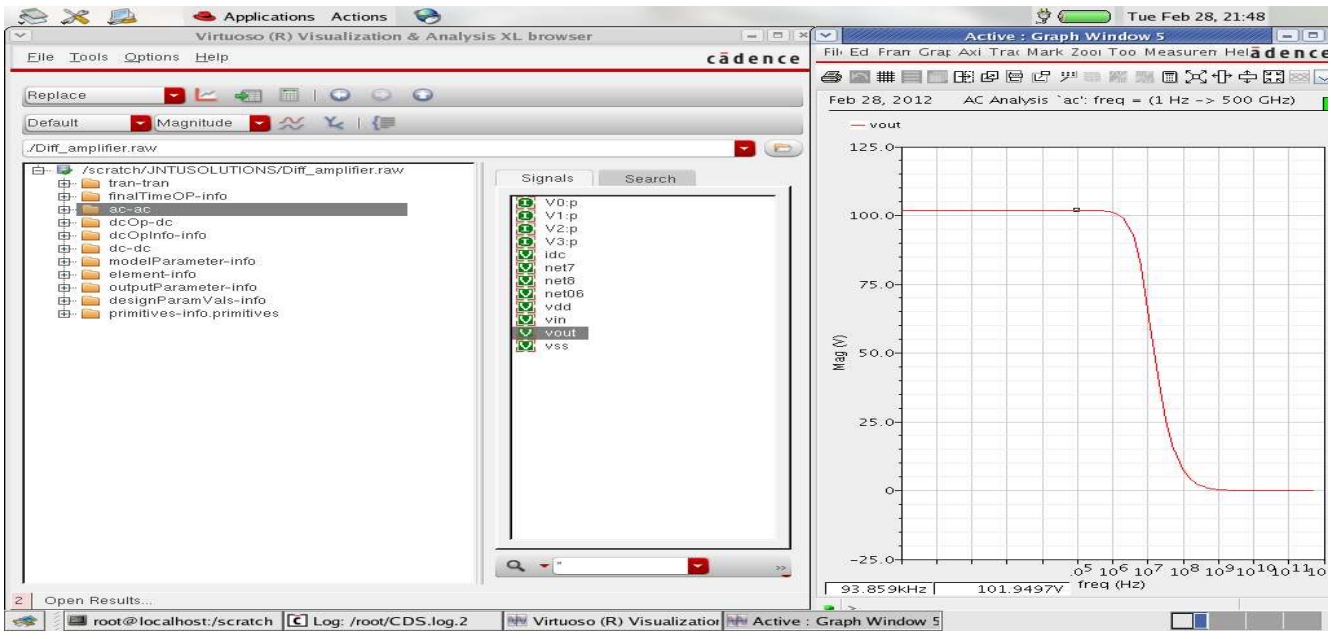
8) For transient analysis: Click on trans-trans, vout and vin



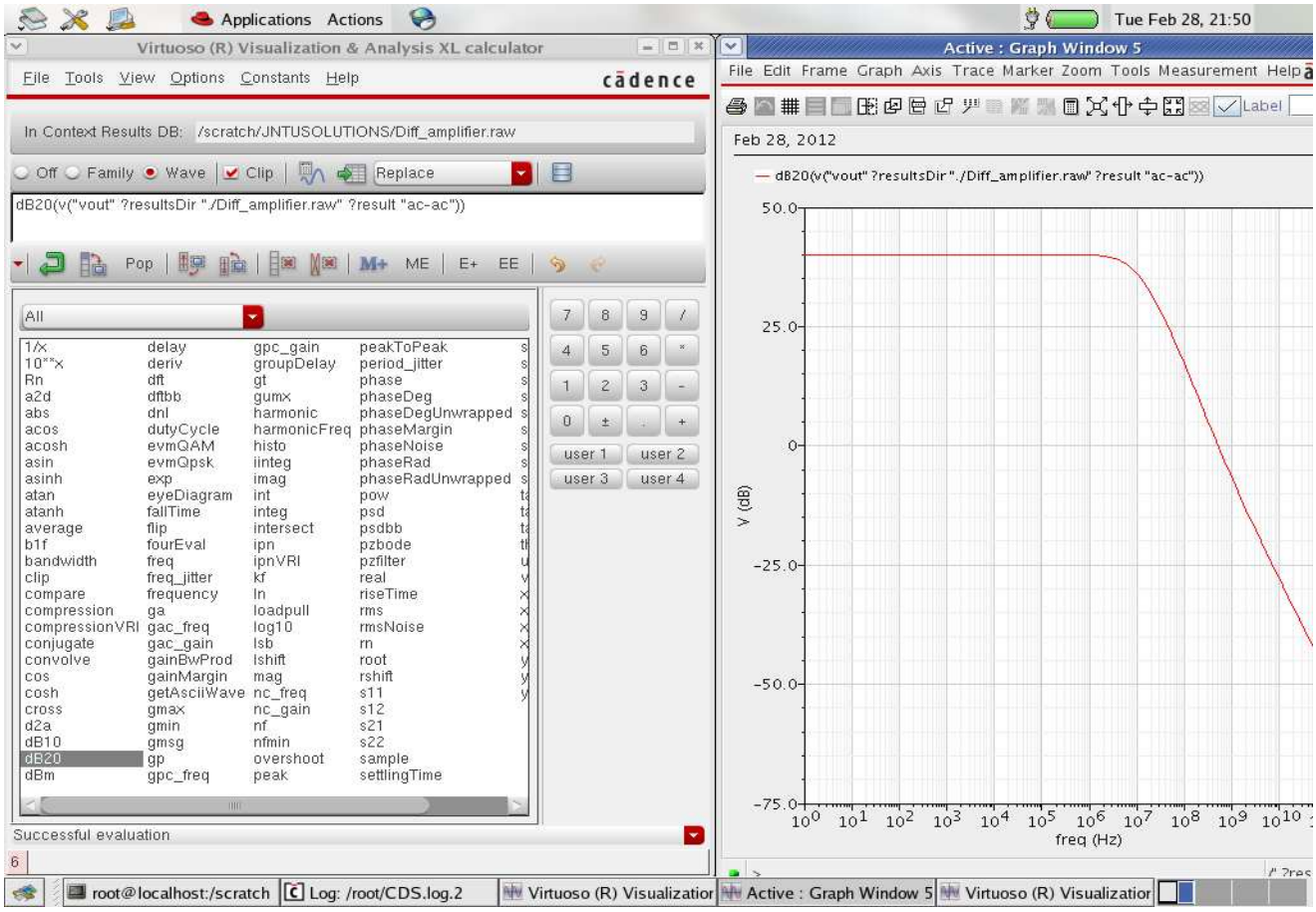
9) For DC analysis: Click on dc-dc, vout.



10) For AC analysis: Click on ac-ac and vout



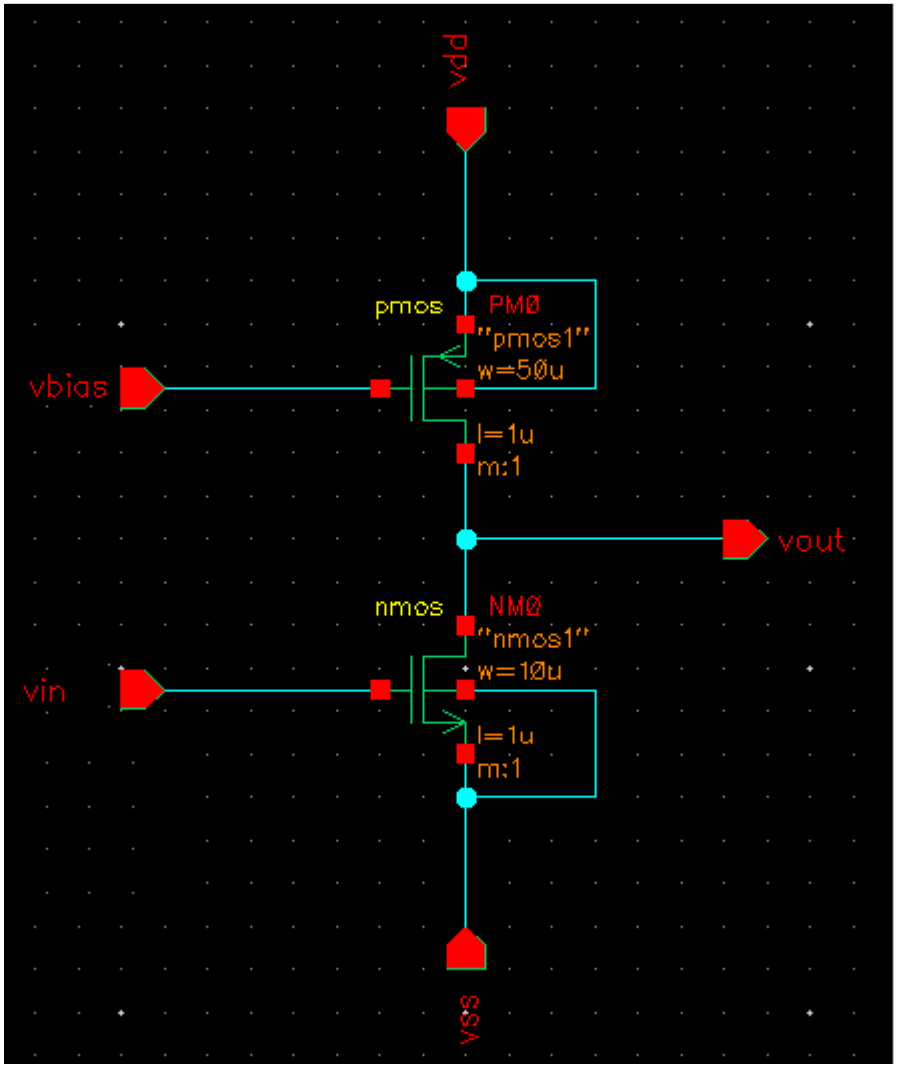
11) To Calculate the gain of Differential pair: select calculator from ac output wave form then select the ac wave form and DB20.



End of lab 5.2

Lab:6 COMMON SOURCE AMPLIFIER

Schematic Capture



Schematic Entry

Objective: To create a new cell view and build Common Source Amplifier

---

Use the techniques learned in the Lab1 and Lab2 to complete the schematic of Common Source Amplifier.

This is a table of components for building the Common Source Amplifier schematic.

Library name	Cell Name	Properties/Comments
gpdk180	Pmos	Model Name = pmos1; W= 50u ; L= 1u
gpdk180	Nmos	Model Name =nmos1; W= 10u ; L= 1u

Type the following in the ADD pin form in the exact order leaving space between the pin names.

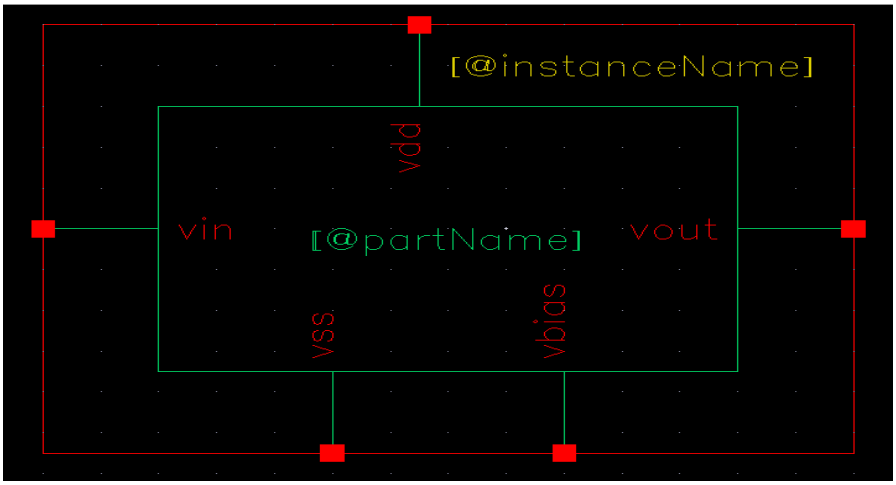
Pin Names	Direction
vin vbias	Input
vout	Output
vdd vss	Input

Symbol Creation

Objective: To create a symbol for the Common Source Amplifier

---

Use the techniques learned in the Lab1 and Lab2 to complete the symbol of cs-amplifier



Building the Common Source Amplifier Test Design

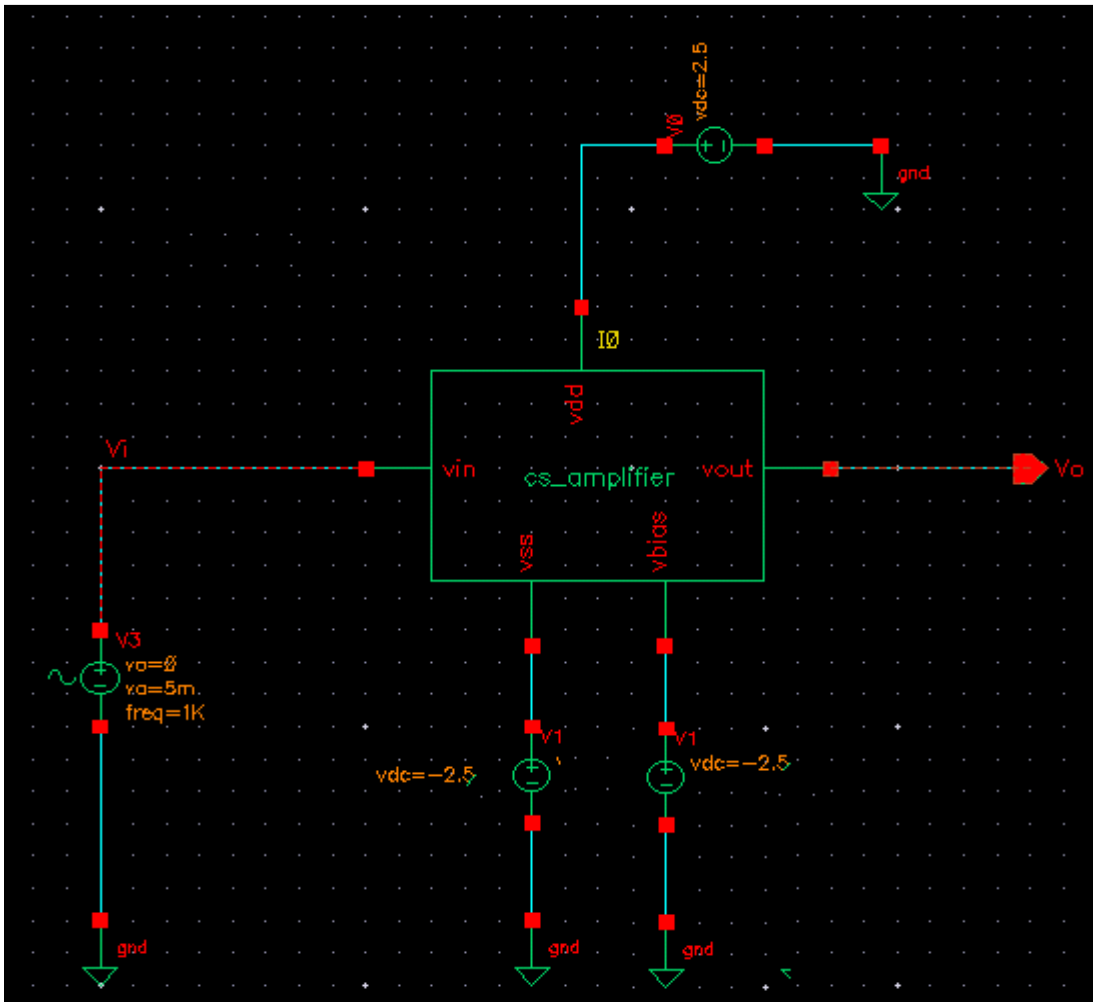
Objective: To build cs_amplifier_test circuit using your cs_amplifier

---

Using the component list and Properties/Comments in the table,

build the cs-amplifier_test schematic as shown below.

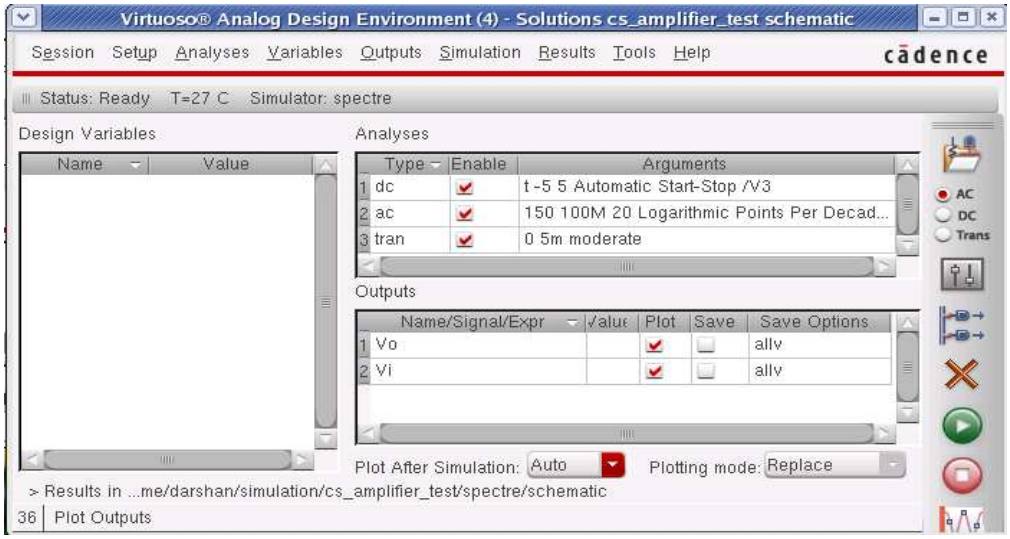
Library name	Cellview name	Properties/Comments
myDesignLib	cs_amplifier	Symbol
analogLib	Vsin	Define pulse specification as AC Magnitude= 1; DC Voltage= 0; Offset Voltage= 0; Amplitude= 5m; Frequency= 1K
analogLib	vdd,vss,gnd	vdd=2.5 ; vss= -2.5 vbias=-2.5

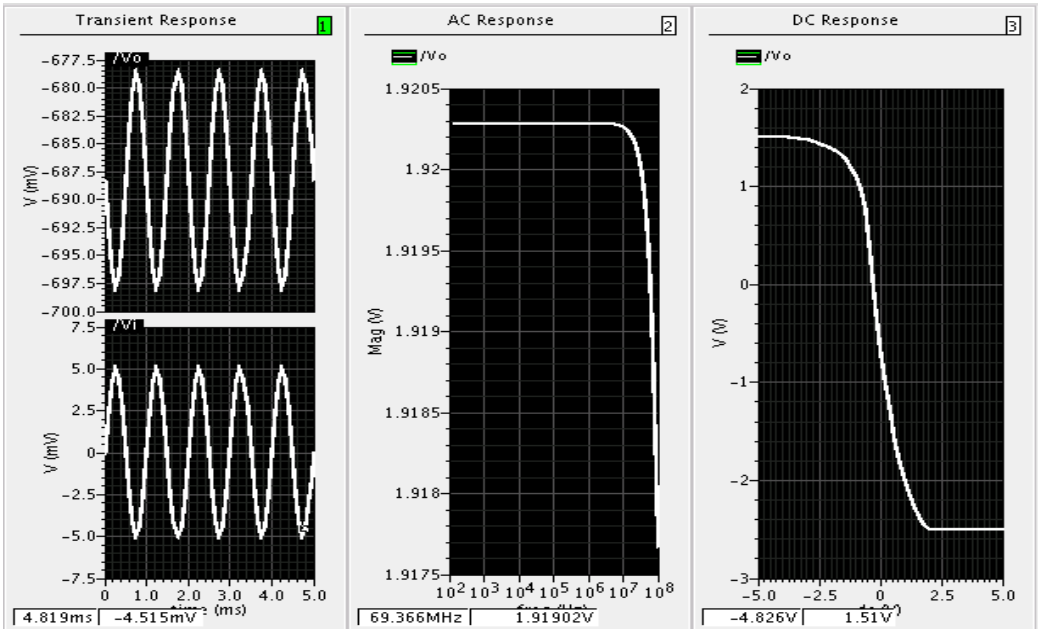


Analog Simulation with Spectre

Objective: To set up and run simulations on the cs_amplifier_test design.

Use the techniques learned in the Lab1 and Lab2 to complete the simulation of cs_amplifier, ADE window and waveform should look like below.



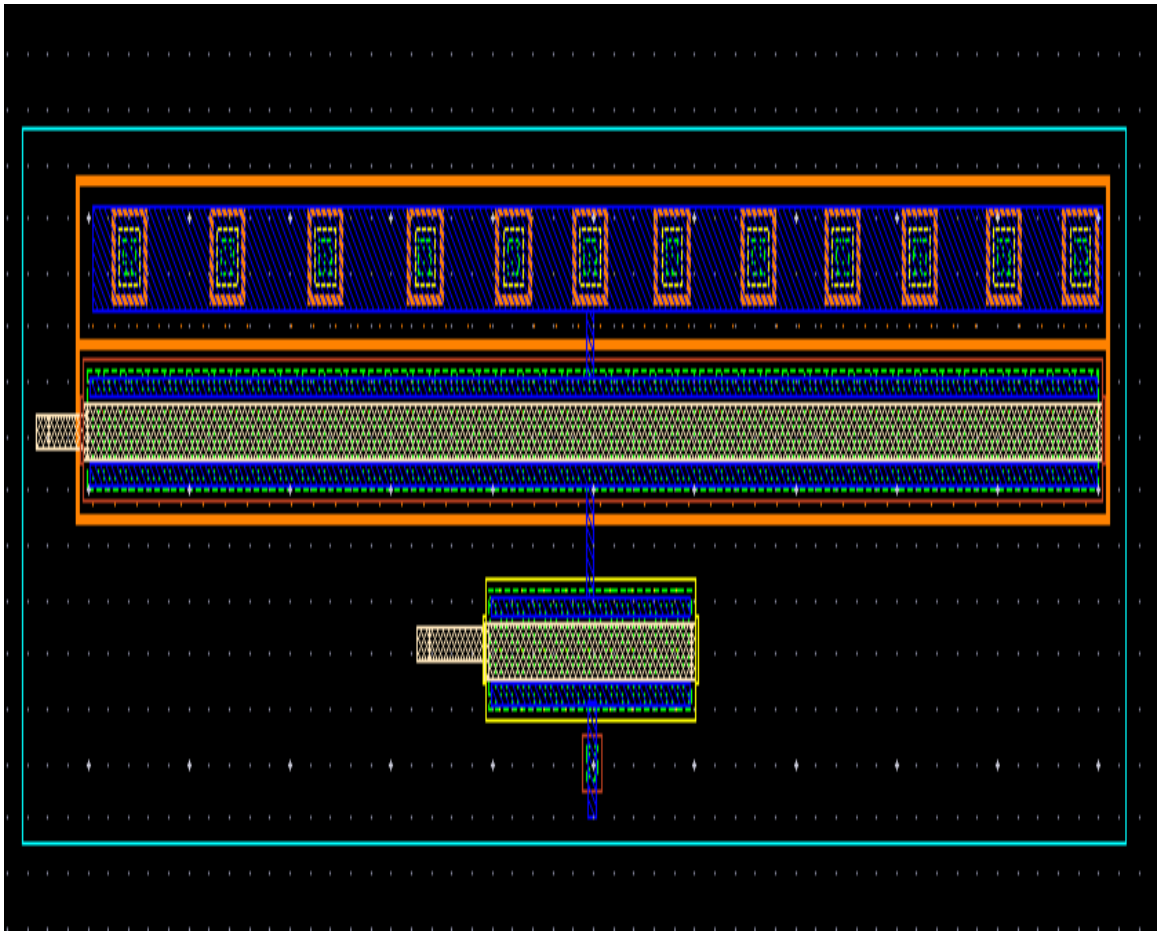


Creating a layout view of Common Source Amplifier

Use the techniques learned in the Lab1 and Lab2 to complete the layout of cs_amplifier.

Complete the DRC, LVS check using the assura tool.

Extract RC parasites for back annotation and Re-simulation.



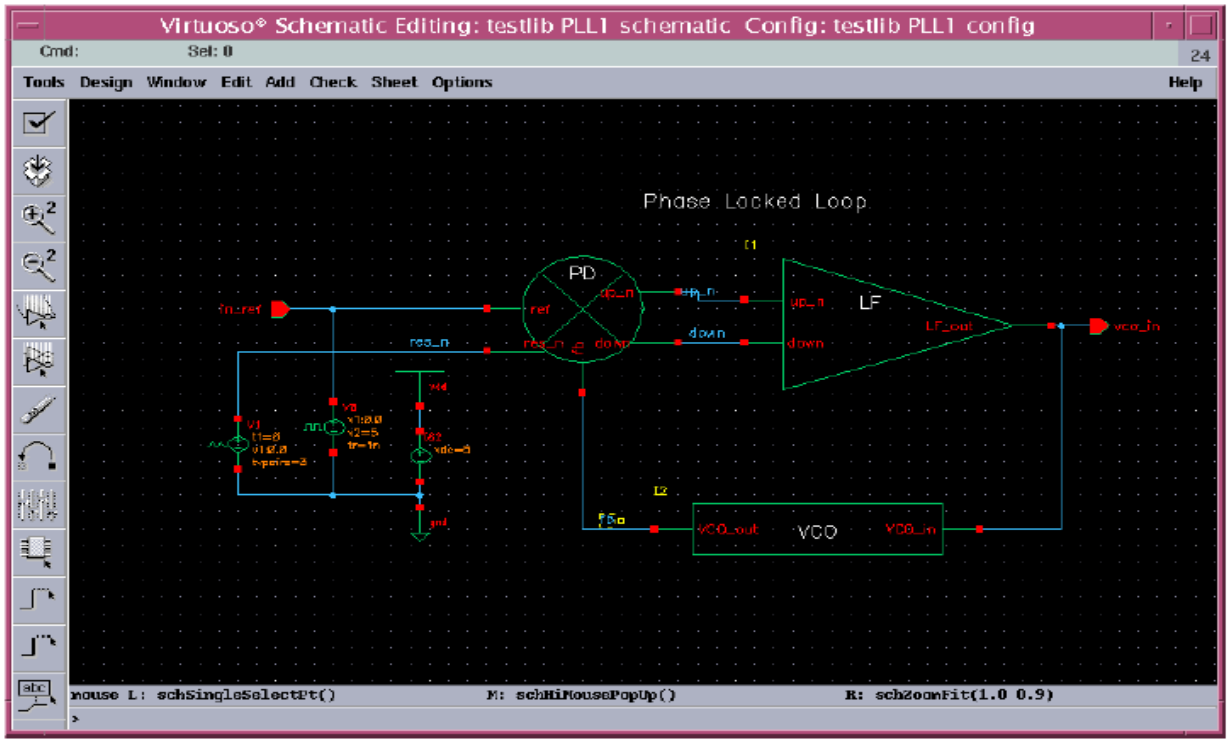
END OF LAB 6

Lab7:Design of PLL:

A phase-locked loop (PLL) generates a clock that is in sync with an input signal. If the input signal changes, the phase detector (PD) detects the difference in frequency and phase between the input and the output and sends a filtered voltage to a voltage-controlled oscillator (VCO) to either raise or lower the output clock frequency as necessary to bring it back into sync with the input signal. PLLs are used widely in data recovery and frequency generation.

The PLL in this lab was designed as a schematic with both schematic and Verilog® views of the phase detector and the schematic and Verilog-AMS views of the VCO. A configuration is used to tell the simulator to use Verilog views for the phase detector and the schematic view of the VCO. Control of the simulation is done using the Virtuoso® Analog Design Environment (ADE), which may be familiar to analog designers.

Schematic:



Compiling the Connect Rules and Modules

1. Change to the *ADElab* directory. If the directory in which the AMS Designer directory is located is not your login directory, then replace the ~ with the correct path.

cd ~/AMSDesigner/ADElab
The contents of this directory are the design contained in the *PLL_lib* directory, a model file in the models directory, a *cds.lib* file, a directory containing Connect Modules and rules, and an *.artist_states* file which can be used to load the ADE form to save time.

2. Compile the Connect Rules and Modules by running the *compileConnect* script. The Connect Rules and Modules will be used to do the level translation between analog ports and digital ports.

. /compileConnect

Starting the Software

1. Start *virtuoso*.
virtuoso &
Note: The ampersand (&) places the command into a background task so the window can be reused.

The Command Interpreter Window (CIW) appears. You may close the What's New window if it appears (**File—Close**).
2. Open the Library Manager from the CIW (**Tools—Library Manager**).

Setting Up and Running the Simulation

1. From the *PLL_lib* library, select the *config* view of the *PLL1* cell and double-click on it. The *config* view is a specific configuration of the design that binds selected views to the cells, and reflects the settings in the Hierarchy Editor.

Select the radio buttons to open both the Configuration (Hierarch Editor) and the Top Cell View (schematic) of the PLL1 cell and click **OK**.

2. In the Hierarchy Editor, note that the PD (phase detector) is set to use the *verilog* view and the VCO1 (voltage-controlled oscillator) is set to use the *schematic* view.
In this lab, the Hierarchy Editor will only be used to view the configuration, not to control the simulation.

3. In the Virtuoso Schematic Editor, which displays the PLL1 configured schematic (*Config:PLL_lib PLL1 config*), execute **Launch—ADE L** to open the Virtuoso Analog Design Environment (ADE).

You may close the What's New in Analog/Mixed Signal window (**File—Close Window**).

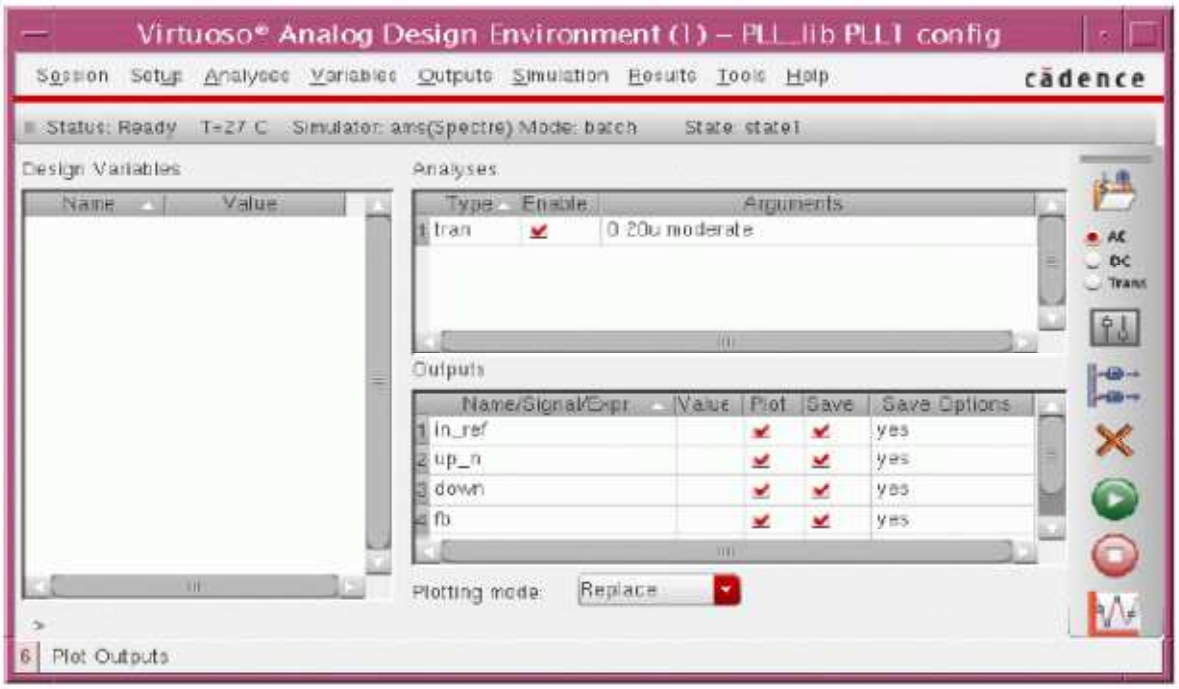
4. In the ADE window, execute **Setup—Simulator/Directory/Host** and, for Simulator, verify that *ams* is selected. Click **OK**.

Note: The simulator was set to *ams* by a line in the *.cdsinit* file:

envSetVal("asimenv.startup" "simulator" 'string "ams")

Note also that this form allows you to change the output directory name and to set up distributed hosts, if you have them available. For this lab, leave it set to a local host.

5. In the ADE window, execute **Session—Load State**. In the Loading State form, select **state1** and click **OK**.
This will pre-load the ADE setup forms with the proper values for this simulation, including a transient simulation of 20us.

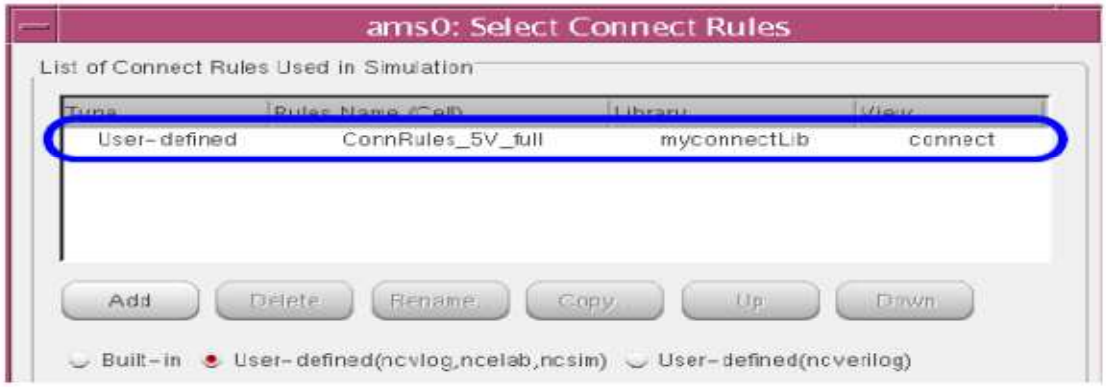


a. Verify the MOSFET models are set to *./models/basicMos.scs* under **Setup—Model Libraries**.

b. Select **Setup—Connect Rules** and select **User-defined** Connect Rules.
Note: If a pop-up display reports an error that you cannot select the built-in rules because the *cds.lib* file does not include a reference to connectLib, you can ignore the message because
you will be using locally defined connect rules. Close the error window.

c. In the Select Connect Rules form in the User-defined rules section, browse to find *ConnRules_5V_full* in the *myconnectLib* library. Close the browser window after highlighting
ConnRules_5V_full, which will copy it to the Select Connect Rules form.

d. Click on the **Add** button to add the *ConnRules_5V_full* Connect Rules to the simulation. The form should look like this:



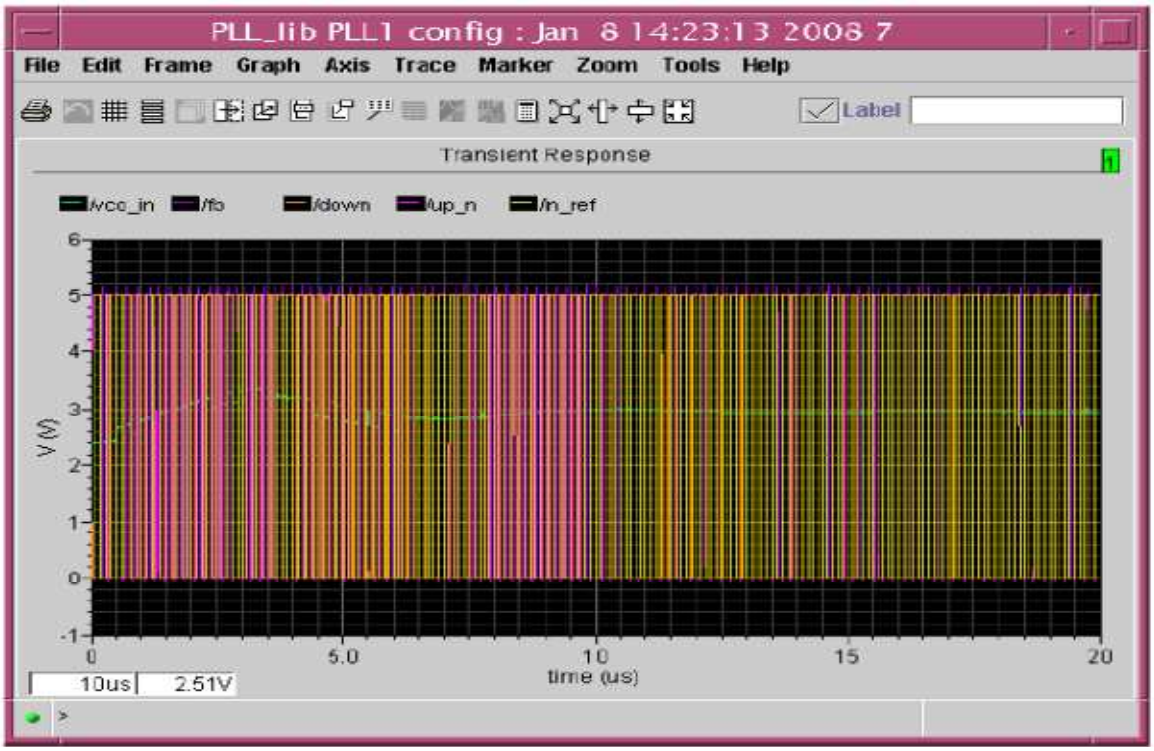
e. Click **OK**.

6. Execute **Simulation—Netlist and Run**, or click on the Netlist and Run icon, to start the simulation.

You may close the netlist and log files when the simulation is finished.

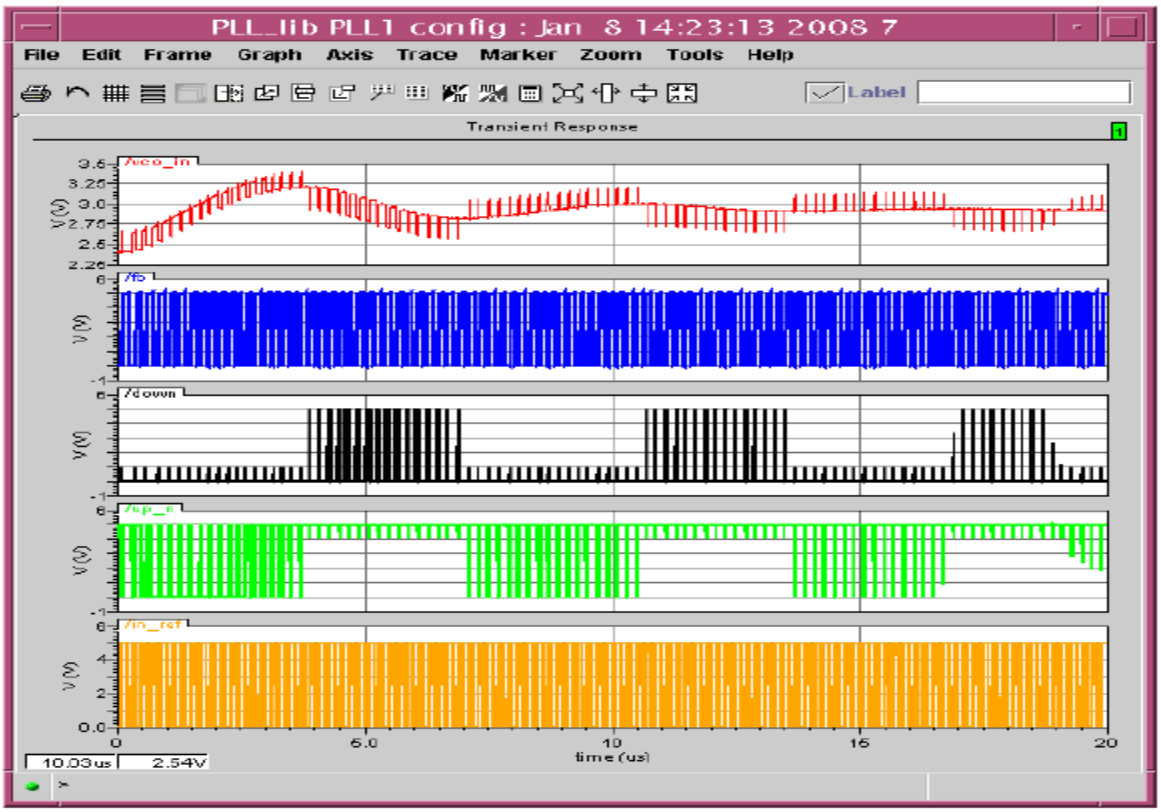
Examining the Results

1. When the simulation completes, ViVA will appear displaying the overlaid traces.



The waveforms need to be formatted to see what is happening in the loop as it locks to the input frequency.

2. Change the analog traces to Strip mode by clicking on the **Strip mode** icon. If all four analog strips are not displayed, select one of them, execute **Trace—Edit**, and change Strip Chart Visible Rows to a higher number. This form also allows you to change trace colors and style. After changing colors for better printing, the ViVA display looks something like this:



Note how the *vco_in* voltage changes in response to *pump-down* pulses (the positive-going pulses) and *pump-up* pulses (the negative-going pulses), and finally settles out after 10 us when the pump-down and pump-up pulse areas are approximately equal. That indicates that syncing to the input signal *(in_ref)* has been achieved.

End of lab 7