Overview
The objective of this brief tutorial is to provide you with some exposure to the Cadence Virtuoso analog IC design tools. In this tutorial you will gain experience with:

- Schematic capture including hierarchical design and sub-circuit symbol generation
- Simulation through ADE XL (ac, dc, tran)
- Parametric sweep simulation
- Monte Carlo simulation accounting for process variation and/or mismatch

Today we will have time for an introduction to only the front-end Cadence tools, encompassing the design flow through schematic-level circuit simulation. Also included with Cadence’s Virtuoso design environment are tools for performing circuit layout, design-rules checking (DRC), layout-versus-schematic checking (LVS), and parasitic extraction for performing post-layout simulations.

This tutorial will step you through the creation of set of schematics for a very simple linear differential pair CMOS amplifier implemented in a generic 180nm process. The schematics will be created hierarchically. The top-level circuit will be a test bench schematic in which the symbol for the diff pair amplifier sub-circuit will be instantiated. Also instantiated in the top-level test bench will be a bias voltage generation sub-circuit. We will run a variety of simulations of the test bench circuit, including parametric sweep and Monte Carlo simulations, and we will learn how to easily substitute different variations of the sub-circuits used for netlisting.

Getting Started
The following will step you through the process of starting up the Cadence tools.

1) Log into a lab computer then log into LATS.
2) Open a terminal window.
3) In your home directory, create a directory called ‘cadence’. (Type: mkdir cadence)
4) Navigate to the new directory. (Type: cd cadence)
5) Run the cadence setup script by typing: setcadencedemo
6) Launch the Cadence tools by typing: virtuoso
7) The CIW window (along with a “what’s new” window, which can be closed) will appear.
8) Tools → Library Manager

The Library Manager window will open. This is the window you will use to access your designs from within Cadence. It is simply a Cadence-specific file browser. A few libraries in the list are of interest today:

- **analogLib** – this is a generic Cadence library containing ideal circuit components, such as resistors, capacitors, inductors, sources, a ground symbol, etc.
• **gpdk180** – this is the generic 180nm design kit that has been installed specifically for this tutorial.

• **demo** – this library has also been installed specifically for today’s tutorial it contains some of the circuit schematics we’ll be using today.

Under each library are a number of cells. The **cells** are individual circuits. Each cell may have several **views**, such as symbol, schematic, layout, etc. In the next portion of this tutorial you’ll be creating a new cell in the demo library. You’ll first create a schematic, and will then create a sub-circuit symbol to contain that schematic.

**Schematic Creation**

In the Library Manager window, highlight the demo library in the list of libraries, then do the following.

1) File → New → Cell View...

2) Fill out the form as shown:

![New File Form](image)

3) Click ‘OK’. A new schematic window will open.

You’ll be drawing the circuit schematic shown in Figure 1. A few of the operations you’ll need to perform can be accessed through the toolbar buttons shown in Figure 2 or by using the shortcut keys described in Table I.
Figure 1. Differential amplifier schematic.
Begin drawing the circuit schematic by instantiating the various components shown in the schematic of Figure 1:

4) type ‘i’, bringing up the Add Instance form:

![Add Instance Form](image)
5) Click Browse to bring up the Library Browser. Select Show Categories, then select the gpdk180 library, the mos category, the nmos cell, and the symbol view, as shown below:

![Library Browser - Add Instance](image)

6) An NMOS symbol will now be attached to your mouse cursor in the schematic window and the Add Instance form will be populated with all the default parameters for the selected device.

7) For each component, fill out the properties as indicated on the schematic of Figure 1, and in more detail in Table II.

Table II. Diff amp component parameters.

<table>
<thead>
<tr>
<th>Component</th>
<th>Library</th>
<th>Cell</th>
<th>Properties</th>
</tr>
</thead>
</table>
| PMOS loads              | gpdk180   | pmos   | Multiplier = 1<br>\( L = 180\text{nm} \)
|                         |           |        | \( W_{\text{total}} = 200\text{u} \)
|                         |           |        | \( W_{\text{finger}} = 20\text{u} \)
|                         |           |        | \( N_{\text{fingers}} = 10 \) (set this value before W values) |
| Diff Pair NMOS          | gpdk180   | nmos   | Multiplier = 1<br>\( L = 180\text{nm} \)
|                         |           |        | \( W_{\text{total}} = 200\text{u} \)
|                         |           |        | \( W_{\text{finger}} = 20\text{u} \)
|                         |           |        | \( N_{\text{fingers}} = 10 \) (set this value before W values) |
| Current source NMOS     | gpdk180   | nmos   | Multiplier = 1<br>\( L = 2\mu \text{m} \)
|                         |           |        | \( W_{\text{total}} = 500\text{u} \)
|                         |           |        | \( W_{\text{finger}} = 20\text{u} \)
|                         |           |        | \( N_{\text{fingers}} = 25 \) (set this value before W values) |
| Degeneration resistors  | gpdk180   | polyres| \( R = 100\Omega \)
|                         |           |        | \( W = 5\mu \text{m} \) |
| Current source ballast resistor | gpdk180 | polyhres | \( R = 200\Omega \)
|                         |           |        | \( W = 2\mu \text{m} \) |
8) Finish drawing the schematic by adding components and wiring them together.
9) Add input and output pins to the schematic by clicking the pin button or typing ‘p’. This will bring up the following window:

![Add Pin Window]

10) When you enter pin names, a pin symbol will be attached to the cursor in the schematic window. You may enter multiple pin names at once. They will be placed on at a time in the order they are entered. Be sure to select the appropriate direction for each pin.

Your schematic should now be complete. The next step, outlined in the following section, is to create a symbol for the schematic you have just drawn.

**Symbol Creation**
Creating a symbol for the amplifier circuit will allow you to place it as a sub-circuit in a hierarchical schematic. The following steps will walk you through the process of generating a symbol.

1) Pull-down menu: Create → Cellview → From Cellview...
2) The following window appears. It will be filled out as shown. Click ‘OK’.

![Cellview from Cellview Window]
3) The Symbol Generation Options dialog will appear. It will be pre-populated with the pin names from the pins on your schematic, though they’ll be in the wrong locations. Rearrange them as shown below. Click ‘OK’.

4) A symbol editor window will open with an automatically-generated symbol. If necessary, edit the symbol to look like the one shown below.

5) Check and save the symbol and close the symbol editor.

**Test bench Schematic**
The next step is to create the top-level test bench schematic. This is the schematic you’ll use for running simulations of amplifier circuit. A test bench named DiffAmp_tb has been partially created for you. It exists in the demo library. You’ll need to make a few simple edits to it before moving on to the simulation phase.
1) In the Library Manager window, select the demo library, the DiffAmp cell, and double click on the schematic view to open it.

2) Instantiate the DiffAmp symbol that you created into the DiffAmp_tb schematic as shown below.
   a) Click on the Create Instance button (the MOSFET symbol) or type ‘i’.
   b) Fill in the fields in the Add Instance dialog as: Library = demo, Cell = DiffAmp, View = symbol, Names = I0.
   c) The symbol will now be attached to your cursor. Place it in the schematic. You may need to adjust some wiring depending on the size and pin spacing of your symbol.

3) The last edit that needs to be made to the test bench schematic is to properly configure the source that supplies the input signal to the amplifier circuit. Select the source, V1, and type ‘q’ to modify its properties. Fill out the component parameters as shown below. Click ‘OK’.
4) Check and save the test bench schematic (Shift+x).

**Navigating the Hierarchy**

You can navigate into the hierarchical sub-circuit block present in the top-level schematic. To push down into the DiffAmp sub-circuit, do the following.

1) Select the DiffAmp symbol.
2) Type Shift+e. This will bring up the following form. Fill it out as shown and click ‘OK’.

3) You are now at the DiffAmp sub-circuit level in the hierarchy and could edit the schematic from here if necessary.
4) To ascend through the hierarchy and return to the top level, type Ctrl+e.

**Working with the Hierarchy Editor**

Oftentimes, particularly when working with a complex design, it is desirable to have multiple versions of the same sub-circuit blocks used in a full circuit. For example, at the early stages of design of a complex
circuit, you may want to model some amplifier blocks (which will eventually be implemented with transistors) with idealized behavioral models. These behavioral models may be implemented with ideal, dependent voltage sources and other components, or with verilogA descriptions. This allows you to focus on the design of other circuit blocks, while still simulating them within the context of the overall system. Even at more advanced stages of the circuit design, it is often desirable to replace transistor-level circuit blocks with behavioral models to reduce simulation time. Cadence provides a tool to facilitate this process, called the Virtuoso Hierarchy Editor.

Just as the schematic editor opens and edits cell views called schematics, and the symbol editor is used to open and edit symbol views, the hierarchy editor operates on a specific type of cell view called a config view. A config view for the DiffAmp_tb cell has already been created for you to use in this tutorial.

1) Close any open schematic or symbol editor windows.
2) In the Library Manager, select the demo library, the DiffAmp_tb cell, and double-click on the config view.
3) Open both the config and schematic views by selecting ‘yes’ for both and clicking ‘OK’.

The Hierarchy Editor and the DiffAmp_tb schematic should now be open. The Hierarchy Editor lists all of the components that are present in all levels of your design. The View Found column lists the default view to that will be used for each cell when a simulation is run. To use a different view, you will enter the name of that view in the View to Use column.

A variant of the DiffAmp schematic that you drew has also been created for you to use in this tutorial. In this alternate view, called idealBias, the amplifier’s tail current source has been replaced with an ideal current source. You can tell the Hierarchy Editor that you want to use this alternate view when you run a simulation by doing the following.

4) Right-click on the View Found entry (schematic) next to the DiffAmp cell, Set Cell View idealBias. Now the View to Use entry for the DiffAmp cell is set to idealBias.
5) Save the config view (click the save button in the Hierarchy Editor).
6) In the schematic window, select the DiffAmp block. Descend into this block (Shift+E). Notice that the view field on the Descend form is set to idealBias. Click ‘OK’.
7) You’ve now descended into the idealBias view of the DiffAmp cell. Note the difference from the schematic view (ideal tail source).
8) Ascend back to the top level (Ctrl+E).
9) Set the View to Use field for the DiffAmp cell back to schematic.
Also included in the top-level test bench schematic (DiffAmp_tb) is a block called VCSgen. This is a bias voltage generation circuit. The VCSgen cell also includes two variant views: schematic and ideal. Verify that both are available in the View to Use popup menu. Descend into each view in from the test bench schematic.

**Simulation with ADE XL**

The Cadence tool you’ll use to simulate the circuit is ADE XL. Similar to the other tools we’ve looked at, ADE XL operates on a specific type of view, called adexl. An adexl view for the DiffAmp_tb schematic has already been created for you to use in this tutorial. Launch ADE XL, opening the adexl view, as follows.

1) In the schematic editor window, Launch → ADE XL.
2) Select ‘Open Existing View’, click ‘OK’. Had a view not already been created for you, you would select ‘Create New View’.
3) The Open ADE XL View form appears. It should be configured as shown below. Click ‘OK’.

![Open ADE XL View Form]

ADE XL now opens in a new tab in the schematic window. The upper-left portion of the ADE XL window, after expanding the Tests and Global Variables items, is shown in Figure 3. Simulations in ADE XL are organized into **tests**, which may contain multiple analyses of different types (e.g. ac, tran, etc.) and different output setups (signals to plot following simulation). The global variables are user-defined variables, used for parameter definitions in your schematics.

This adexl view includes one test that has already been defined. It has been named ‘tran’ and includes two analyses: a dc operating point analysis and a transient simulation. You can double-click on the analyses to modify their parameters. You can add a new analysis to a test, and you can add new tests by clicking in the indicated places. We’ll add a test later after running the existing test, tran.
Also shown in the ADE XL window is the Outputs Setup tab, as shown in Figure 4. This is a list of user-defined simulation outputs – signals or mathematical expressions using signals – to be calculated and plotted following a simulation run.

Figure 3. Data View portion of the ADE XL window.

Figure 4. Output Setup tab in the ADE XL window.
A few of the more important controls present in the ADE XL Outputs Setup tab are indicated in Figure 4. The simulation type selection allows you to select other types of simulations, such as Monte Carlo analysis and design optimization.

At this point you are ready to run a simulation.

4) Click the run button, and a simulation should begin running. Once the simulation has completed the output signals should be plotted automatically in a new window.

5) Click on the Results tab. A summary of the simulation results is shown here.

6) Also from within the results tab, you can select the plotting mode of the simulator to be either append, replace, new subwindow, or new window. ‘Append’ means that the results from successive simulation runs will be overlaid in the waveform plotting window. ‘Replace’ means that each new set of simulation output signals will replace those from the previous run. Set the plotting mode to ‘Append’.

7) Play around with running the transient simulation. Try varying some of the global variable values. Also, try running simulations using different views for the VCSgen and DiffAmp cells, overlaying the simulation results.

Adding a Test

Next, we’ll add another test to the adexl view.

1) In the Data View portion of the ADE XL window, click on the text that says ‘Click to add test’.

2) The Choosing Design dialog will appear. Make sure it is configured as shown below. Click ‘OK’.

3) The Test Editor window appears next. Click on the ‘Choose Analyses’ Button as indicated below.
4) The Choose Analyses dialog will then appear. Select the DC analysis radio button and select ‘Save DC Operating Point’. Select the AC analysis radio button, then fill out the form as shown. Click ‘OK’.
5) Close the Test Editor window. A new test should now be listed under the with the other test (tran) in the ADE XL window. It will be names something like ‘demo:DiffAmp_tb:1’. Rename it by clicking on it twice slowly (not double-clicking), and typing a new name, such as ‘DiffAmp_ac’, or whatever you’d like.

6) Select this test and deselect the other test. The ADE XL window should look as shown below.

![ADE XL window with DiffAmp_tb selected]

7) Now run a simulation. You have not yet defined any outputs for this test, so no waveforms will be plotted.

8) In the Outputs Setup tab, click the button circled below, and choose ‘Show Enabled Tests’.

![Button to add new output]
9) Click the Add new output button. A new output assigned to the new test will appear. Name it Gain.

10) Click in the Expression field, then click on the ellipsis (...). This will bring up a calculator window.

11) Select the VF radio button at the top of the calculator. That tells the calculator you want to enter a frequency-response voltage signal. The schematic window is brought to the front. Select the output signal, Vo. On the function panel of the calculator click on ‘dB20’. This converts the frequency response at Vo to dB magnitude. If you don’t see ‘dB20’, make sure the ‘All’ (not ‘Special Functions’) is selected as the Function Set, as shown below.

12) Back in the Output Setup tab of the ADE XL window, click on the Expressions field again and click on the arrow button. The expression you entered in the calculator will now be entered here. Click the save button.

13) Add another output, called ‘DcGain’. In the Calculator window, with the previously-defined Gain expression still in the buffer, click on ‘value’ in the Function Panel. Enter ‘1k’ in the Interpolate At field (i.e. calculate the value of the Gain expression at 1kHz), and click ‘OK’.

14) Move the expression from the Calculator buffer to the Expressions field in the Outputs Setup tab of the ADE XL window, as you did previously for the Gain expression. This is an example of a scalar, as opposed to waveform output.

15) Now, run the analysis again. The amplifier’s frequency response should be plotted in the waveform viewer, and the DC gain value will be calculated and displayed in the Results tab of the ADE XL window.

The calculator allows you to define any output signal you wish, both waveforms and scalars. For example, for an AC analysis you could define outputs that calculate bandwidth of the amplifier, or the amount of peaking, etc. For a transient simulation you may want to define outputs that calculate parameters such as risetime, voltage swing, or overshoot, etc.

**Parametric Sweeps**

It is often desirable to run a set of simulations over some range of a parameter or parameters. As an example of this, we’ll now investigate how input common-mode voltage affects the amplifier’s frequency response by performing a parametric sweep simulation, varying the global design variable Vicm.
1) In the Data View section of the ADE XL window, expand the Global Variables item. Double-click on the ‘1’ value next to ‘Vicm’. Click on the ellipsis (...).
2) The Parameterize form appears. Enter the five values, as shown below: 800m 900m 1 1.1 1.2. Click ‘OK’. Save the design.

![Parameterize form](image)

3) Run a simulation (click the Run button). Five separate simulations will now run. When they finish, five separate frequency responses will be plotted, illustrating how the response varies as the input common-mode voltage varies. The scalar output, DcGain, is also plotted as a function of Vicm. This type of analysis can be extremely useful.

**Monte Carlo Simulation**

When designing an IC it is critical to ensure that your design works acceptably over process, voltage (supply voltages), and temperature (PVT). Two types of process variation are of concern: variation from one chip, wafer, or wafer lot to the next (this we refer to as *process variation*), and variation between nominally-matched devices on the same chip (this we refer to as *mismatch*).

The models provided with a design kit will include both worst-case corner models, as well as statistical information characterizing device variation in terms of both process and mismatch. It is possible to run worst-case corners simulations, as well as statistical, or *Monte Carlo* simulations. Here we will run a simple Monte Carlo analysis to see how we can expect our amplifier to vary over process.

1) Set the run mode to ‘Monte Carlo Sampling’.

![Monte Carlo Sampling](image)

2) Click the Simulation Options button and fill out the form as shown. This configures the Monte Carlo simulation to account for process variation, and to run 20 separate simulations, each sampling device parameters at a different point on the process variation distribution. Click ‘OK’. Save.
3) Before running a Monte Carlo simulation we need to remove the parametric sweep on the Vicm variable. Change its value back to 1.

4) Run a simulation (click the run button).

**Conclusion**

That concludes today’s tutorial. The goal of this tutorial was not to teach you how to use the Cadence IC design tools (that, unfortunately will take significantly longer than two hours), but to provide you with some idea of their capabilities and usefulness, as well as their complexity. Hopefully this exposure has prepared you to begin the process of learning to use Cadence as an IC design tool.