Columbia University  
Department of Electrical Engineering  
EE E6930. Tutorial #1.  
Schematic entry and circuit simulation.

This is a first in a series of tutorials using the Columbia Integrated Systems Laboratory VLSI design flow. These tutorials are not intended to teach you everything about the tools but instead provide the essentials needed to get started. For additional detail, please consult the on-line documentation. Don’t be afraid to seek out the TA or instructor for help. Also, please be patient, since glitches and problems with the tools are inevitable. This is the way it is with any complex electronic design automation software environment, and you will find the same glitches and problems in industry.

In this tutorial, you will how to do design entry in Cadence’s Composer and enter and simulate a schematic with HSPICE.

Since you are new to the SUN machines in the VLSI lab, you must do a few things to set up for the class as well as for this tutorial.

1. Please add the line:

   . /usr/tools/init/kshrc

   to your .profile file. This will ensure that you pick up the default environment set up for the class. (Note: You want to add this where it will be run.) ksh will be the only supported shell and is the default shell that has been set up for you.

2. Please create a file called .cdsinit. In this file, please add the line:

   load("/usr/tools/cds/setup/cdsinit");

3. Please copy /usr/tools/cds/setup/cdsenv to .cdsenv in your home directory.

4. Please create a file called cds.lib in your home directory. Edit the file, adding a single line:

   INCLUDE /usr/tools/cds/setup/cds.lib

1
On-line Cadence documentation can be found by typing `openbook` at the
UNIX prompt.
1. **Canned Cadence tutorial.** Please do the Cadence Composer tutorial,
Chapters 1 - 5. You will find the tutorial if you run the `openbook` command
at the UNIX prompt. Once at the main menu, select “Design Entry”. Then
select “Virtuoso Schematic Composer Tutorial”. Please use `icfbz` to bring
up the Cadence software. It will take several hours to work through the
tutorial, but it is worth it!. It is very important that you are familiar with
Composer, so please don’t cut any corners here.
2. **EE E6930 Composer and HSPICE tutorial.** In this tutorial, we
will learn the specifics of the design environment for schematic entry for
this course and how to simulate circuits in HSPICE. To do this, we are
going to create and simulate a simple inverter chain using the TSMC 0.25 μm
technology.

First, create a library to hold your test design. To do this from the CIW,
choose **File → New → Library.** You will get a “New Library” form. In this
form, enter the name “tutorial1” into the Name field. On the technology file
options, choose “Attach to an existing techfile.” The Design Manager option
should remain “No DM.” You will then get an “Attach Design Library to
Technology File” window. Choose the library “tsmc025” from the cyclic field
and click “Ok.” This gets the technology information for your library from the
“tsmc025” library, which is the technology-specific library that we are
using in this course. The technology file information contains all the details
on the technology that are necessary to get all the Cadence tools (particularly
the layout tools) to work properly.

Now, open the library you have just created. From the CIW, choose
**Tools → Library Manager.** Select the “tutorial1” library from the Library
Manager window. We are now going to create an inverter. Choose **File →
New → CellView** from the Library Manager window. In the “Create New
File” form, enter the cell name “inverter” and the view name “Schematic”.
Hit “Ok.” A Composer schematic window will now open for us to create our
inverter.

Let’s first instantiate the pMOS. Choose **Add → Instance.** Click on the
“Browse” button on the “Add Instance” from. A Library Browser window
opens. Choose the “tsmc025” library, the “pmos” cell, and the “symbol”
view. Hit “Close” to close this window. Now, the library, cell, and view
fields on the “Add Instance” form should be filled out nicely for you. Now,
if you scroll down this form, you will see a series of fields that you should fill in to indicate the size of this transistor. At a minimum, you will need to fill in “Width” and “Length.” For now, leave the “Drain diffusion area”, “Source diffusion area”, “Drain diffusion periphery”, “Source diffusion periphery” as zero (these correspond to the AS, AD, PS, and PD parameters in SPICE) and ignore the fields for the “layout type” and ”number of fingers.” In Tutorial 3, we will introduce layout-dependent device type which will “automatically” calculate these source and drain diffusion parameters for more accurate prelayout simulation. For now, we will just fill out “Width” and “Length.” We will use a minimum length device, so in the length field, type 0.24u for 0.24μm. In the width field, type 20u for a 20μm width FET. Hit the “Hide” button to get rid of the “Add Instance” form. Place the transistor and hit the “Esc” button to break out of the add component operation (remember, “Esc” gets you out of anything).

Repeat this operation, instantiating a “nmos” cell from the “tsm025” library. You will also need to grab the “vdd” and “gnd” symbols from the “analogLib” library. You can place the nmos, pmos, vdd, and gnd symbols so that they already connect. If you place them apart, then you will have to wire them up.

Now you need to add a couple of pins and then wire up the pins. To get the pins, choose Add → Pin. Choose the direction “input” and let the pin name be “a”. Use “Hide” to hide the form and “Esc” to break out of the add pin operation. Add an output pin “y” in a similar fashion. Wire up the schematic now using Add → Wire(narrow). When you are done, use Design → Check and Save to check the schematic for errors and save it.

Now, you will need to make a symbol for your inverter. It usually easiest to start out with the “box” symbol that Cadence creates for you automatically and then modify it. We will follow this route. Choose Design → Create Cellview → From Cellview. The “From View Name” should be “schematic” and the “To View Name” should be “symbol.” Hit ”OK.” You will then get a “Symbol Generation Options” form. The default should be fine with pin a on the left and pin y on the right. Hit ”OK.” You now get a symbol editor window and we can now modify the symbol so that it looks like an inverter (rather than a box). To perform the modification, you probably want to delete the box, using Edit → Delete and then selecting the box (Remember to look at the bottom left-hand corner of your design window for prompts of what command if active or what action Composer expects you
to do). Hit “Esc” to break out of delete. Now, create an inverter symbol using **Add → Shape → Line** and **Add → Shape → Circle**. The red box is the bounding box for the symbol. You might want to resize it using **Edit → Stretch**. When you are done, check and save your symbol and exit the symbol window. You can also close the schematic window for the inverter since we are done with this as well.

Now, in your “tutorial1” library, create a new schematic cell view for the new cell “topJlevel.” In the Composer schematic editor, create and wire up an inverter chain consisting of three instantiations of the inverter cell symbol you created above. At the end of the inverter chain, instantiate a load capacitance using the “cap” symbol in the “analogLib” library. You will need to fill out the “Capacitance” field on the “Add Instance” window. Use 20\(\mu F\) for 20\(\mu F\). On the input, we are going to add a source so that we will be able to simulate the design. Choose the “vpwl” symbol from the “analogLib” library. In the “Number of time points” field, enter 4. Now you will need to fill in the four time-value pairs.

- Time 1. 0n
- Voltage 1. 0
- Time 2. 1n
- Voltage 2. 0
- Time 3. 1.1n
- Voltage 3. 3.3V
- Time 4. 3n
- Voltage 4. 3.3V

To get ready for simulation, you will also need to instantiate a “vdc” source between “vdd” and “gnd” with a dc value of 2.5 V (the nominal supply voltage for our technology). Add a border to your schematic to make it look nice by choosing **Sheet → Edit Size**. Choose the A size schematic. Remember to Check and Save your schematic. You can print your schematic on the printer in the VLSI laboratory by choosing **Design → Plot → Submit**.
Now, we will simulate the design using HSPICE. HSPICE is a commercially supported version of SPICE, which means that it is considerably more robust and has much more function that Berkeley SPICE (companies would never even think of using Berkeley SPICE!). Choose Tools → Analog Artist. This opens up an Analog Artist Simulation Window. If you picked up the EE E6930 .cdsenv file, the simulator in the top right corner of the simulation window should be “hspiceS.” If it is not, you will have to choose Setup → Simulator/Directory/Host to change it. The temperature in the top right corner should also be specified as 25°C, which will be the temperature we use for all simulations in this course. You next need to set up the path to the BSIM3v3 device models to be used in this course. We will do this with an include file. Choose Setup → Environment. In the “Include File” field, type

/usr/tech/tsmc025/models/tsmctt

In the “Include/Stimulus File Syntax” field, select “hspice.” Now to set up the analysis, choose Analyses → Choose. Select “tran” and enter a stop time of 3n for 3nsec. To select outputs to be saved as part of the analysis, choose Outputs → To Be Saved → Select On Schematic. Then select on the schematic by clicking on the appropriate nets the outputs to be saved as part of the simulation run. To select outputs to be plotted as part of the analysis, choose Outputs → To Be Plotted → Select On Schematic. Then select on the schematic by clicking on the appropriate nets. Choose Simulation → Run to netlist and run your simulation. When the simulation is completed, it will automatically plot the results selected. If there are any problems, you can view the simulation log by choosing Simulation → Output Log. You can print your waveforms by selecting Window → Hardcopy on the waveform window.

Now, we are going to modify the design to use an inverter whose size is parameterized. Create a cell “pinverter” schematic view by copying the current “inverter” schematic view. You can use the copy functions from the Library Manager for this. Open the “pinverter” schematic and select the nMOS device. Hit ‘q’ to display the Edit Properties form. In the “width” field, enter pPar("size"). Now do the same for the pMOS device, but enter 2.0*pPar("size") into the “width” field. Now create the symbol from the schematic (the reason for doing this is that this will automatically update the base CDF properties for “pinverter” to include the new parameter size).
Now copy the schematic for “top_level” into “top_level_parm.” Delete the instances of the “inverter” cells. Use Add Instance to add instances of “pinverter” instead. You will size a parameter size now appear on the “Add Component” form. Fill this in with 6 u. Rerun HSPICE to verify that you get the same results as before. This parameterized cell scheme allows you to resize a particular cell instance using parameters on the master.