Lab 1 Part 1: Schematic Design and Simulation

Report Due Date: February 26, 2018 at 6:00 pm

© 2018 Christoph Studer (studer@cornell.edu)

- You will create a schematic and symbol for a CMOS inverter in Cadence Virtuoso.
- You will perform a transient simulation for your inverter.

---

1 Create an Inverter Schematic

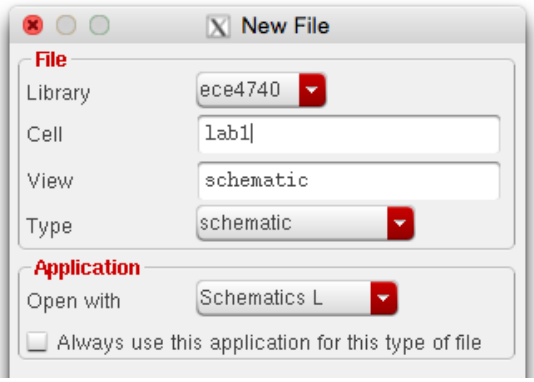
The technology we will use in ECE 4740 is the Generic Process Design Kit ("GPDK090") 90nm CMOS process. This is nominally called a 90nm technology (gate pitch), but the minimum transistor length is 100nm (minor gate length expansion, which is common in more recent technologies). In Part I, you will create a schematic for a CMOS inverter. Size the total widths of `pmos` and `nmos` to 240nm and 120nm, respectively, so that the inverter has a $W_p : W_n = 2 : 1$ ratio for a more balanced rise and fall output time. Always keep the transistor lengths at the minimum, i.e., L =100nm. Add input and output pins to your schematic.

Note: anytime Cadence asks you about checking out a license, select 'always'.

1. We will start creating the schematic for our CMOS inverter. Start out by selecting File→New→Cellview from the Cadence command window.
2. In the following window, set the following fields to the appropriate values:

   Library Name: `ece4740`
   Cell Name: `lab1`
   Type: `schematic`
   Open with: `Schematics L`

   Click OK.

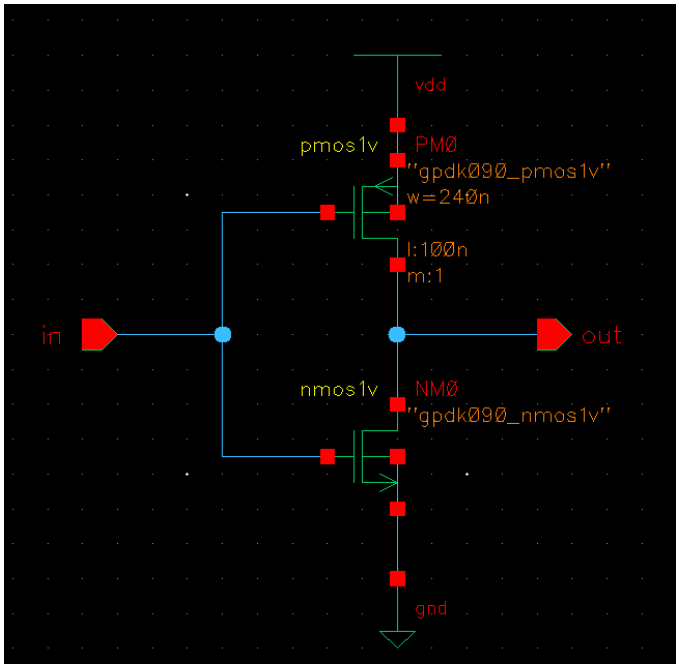


3. You should now notice a new window that with the title "Virtuoso Schematic Editing." Now let's start placing components. Create a new instance by either clicking on the "instance" toolbar button on the left or by pressing "**i**," which stands for "instance."

4. A new window should appear asking for a few values. Notice the first two fields labeled "Library" and "Cell". Library indicates the technology library that contains the component of which you wish to make an instance. Cell indicates the name of the cell view. Note that in order to add an instance of a cell view to a schematic, there needs to be a symbol of that specific cell view. Let's add an `nmos` transistor in the `gpdk090` technology:

5. Click "Browse" next to "Library". In the resulting window select the following:
   Library: `gpdk090`
   Cell: `nmos1v`
   View: `symbol`
   And click Close. You will notice that there is now a transistor symbol attached to your mouse cursor in the schematic view window. If you left-click once, you will place a single instance of this transistor on your schematic. You could have also just entered in the appropriate fields in the "Add Instance" window without having to browse for it. You can also adjust the parameters, such as width and length, of your `nmos` by changing them in the "Add Instance" window under the appropriate parameter fields. If you later need to modify instance parameters, click on the device you wish to modify and press "**q**" standing for "qualities."

6. In order to make an inverter, we need to also add the components `pmos`, `vdd` and `gnd` as shown in the following figure. Use the same method as before to add these components. The `pmos` transistors can be found in the `gpdk090` library; `vdd` and `gnd` will be in the analogLib library. We will add the pins and wires in the next steps.



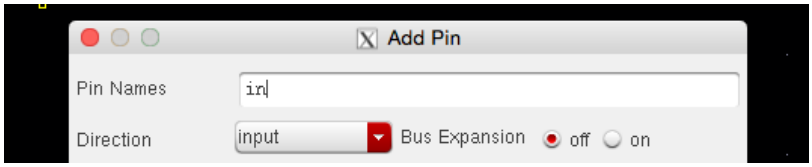
The names `vdd` and `gnd` are most often used as the power rails in CMOS. However, legacy names are summarized in the table below.

	BJT, TTL	FET, CMOS
Positive supply voltage or nominally highest voltage	V_{cc}, V_+, V_{s+}	V_{dd}, V_+
Negative supply voltage or nominally lowest voltage	V_{cc}, V_-, V_{s-}	V_{ss}, GND, V_-

7. Create input and output pins as shown in the figure by selecting Add→Pin or by pressing "**p**". Now create an input pin by doing the following:

   Pin Names: in
   Direction: input

   Click OK.



   You should see an arrow attached to your mouse cursor, much like you saw when adding an `nmos` transistor. Place this pin accordingly.

8. Use the same method to add an output pin: make sure to set the "Direction" field to "output". We don't need to create pins for `vdd` and `gnd` because they're globally defined (hence the net names `vdd!` and `gnd!`).

9. Click the "Check and Save" button on the top of you toolbar. You should click this often to make sure you don't lose your work.

10. Next, we need to wire all components as shown in the figure. Press "**w**" to start drawing a wire. click once on the starting point and draw the wire, then click again on the stopping point. You can also use "**s**" for the starting point and then "**s**" again at the end point. You can also use "**l**" to create labels to connect wires via naming; a must for cleaning up otherwise messy schematics.

11. The transistors in the `gpdk090` library are four terminal devices (Source, Drain, Gate, and Bulk). For this lab, we will tie the bulk connection of `nmos` to ground and the bulk of `pmos` to `vdd`. This approach minimizes leakage to the substrate by avoiding forward biases at the S/D contacts.

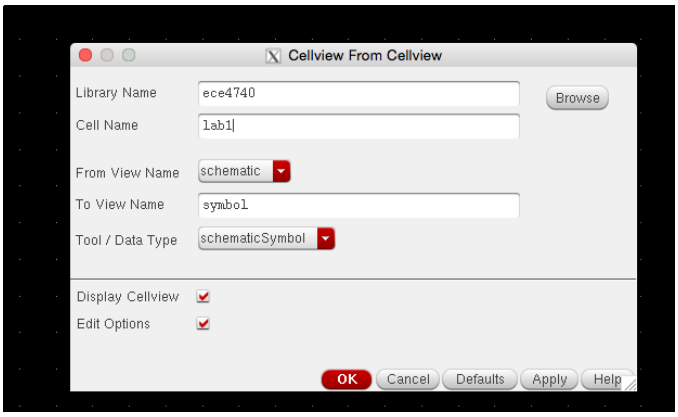
2 Symbol Generation

1. Now that you have created a CMOS inverter schematic, let's create a symbol of it so that it can be instantiated within other schematics just like the `nmos` or `pmos` symbols in the `gpdk090` library.

2. Select Create→Cellview→From Cellview

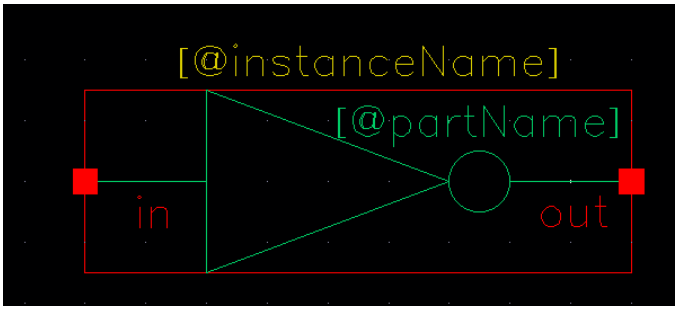
   Now set the following parameters:
   Library Name: `ece4740`
   Cell Name: `lab1`
   To View Name: `symbol`

   And click OK. Click OK in the next window as well.



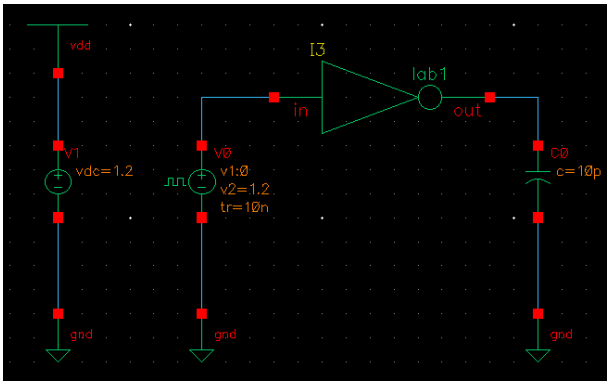
3. Use the drawing tools to create a symbol for the inverter symbol.

You *could* just use the automatically created box symbol, but that's just plain embarrassing for an inverter (and you will lose points). We recommend drawing a triangle with the line tool, then a circle, and then move the pins into the right place. Move the selection box over your inverter when you're done.
Make sure to check and save your symbol.

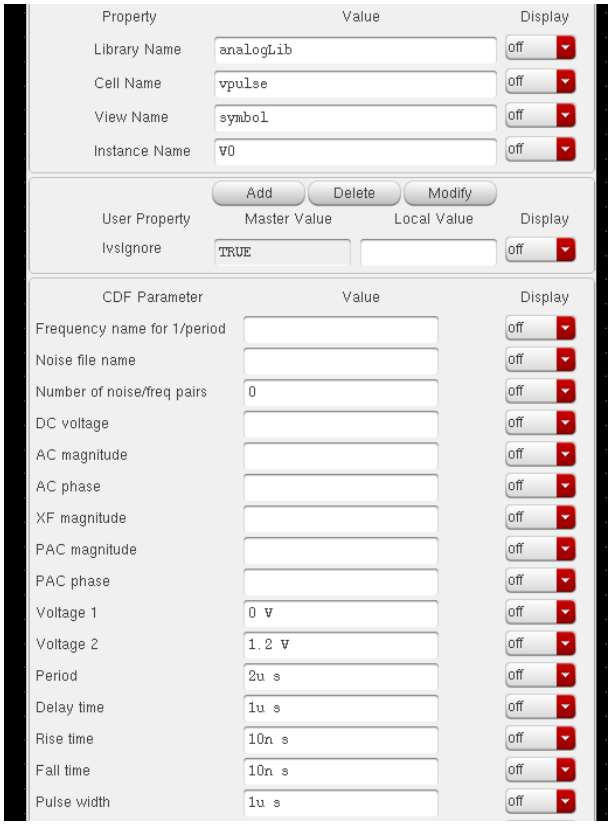


3 Transient Simulation

1. Now let's execute a transient simulation. To do this, let's first create a schematic in which to run the simulation. Create a new schematic named `lab1_sim` as shown in the following figure. Everything you need should be in the `analogLib` or `ece4740` libraries. For the input square wave, use the `vpulse` cellview. For the `vpulse` parameters, it should look like the following:



Make sure to check and save your schematic.



2. Don't forget to set the values specified in the figure for the capacitor and `vdc`.

3. Now that we have the simulation schematic, let's setup the simulation files. Select Launch→ADE L.

4. In the popup window, select Setup→Simulator/Directory/Host and set the following fields:
Simulator: `spectre`
Project Directory: `~/Cadence/simulation`
Host Mode: `local`
Click OK. This will set Cadence to use the Spectre simulator and will create a folder within your

Cadence directory for storing simulation files. For this simple simulation, not many simulation files will be generated, so it may not yet be a problem, but when you simulate much larger projects, this may fill up your disk quota. To solve this, you can redirect your project directory to the temp directory on the server at `/tmp/YourNetID/simulation`. Since this is your temp directory, it will be cleared once you log off, but you really shouldn't need to access your simulation files anyways. If you need results again, just rerun the simulation.

5. Select Setup→Temperature and set it to 25 degrees Celsius.

6. Select Setup→Environment and set:
   Switch View List: `spectre cmos_sch cmos.sch schematic veriloga`
   View List: `spectre`
   Check "Automatic output log"

7. Now save your simulation state by selecting Session→Save State. Load this state for future simulations and modify as necessary (You can save to either "Directory" or "Cellview").

8. Now let's set up the transient analysis. Select Analyses→Choose from the Analog Environment window or click the first button from the top on the toolbar on the right.

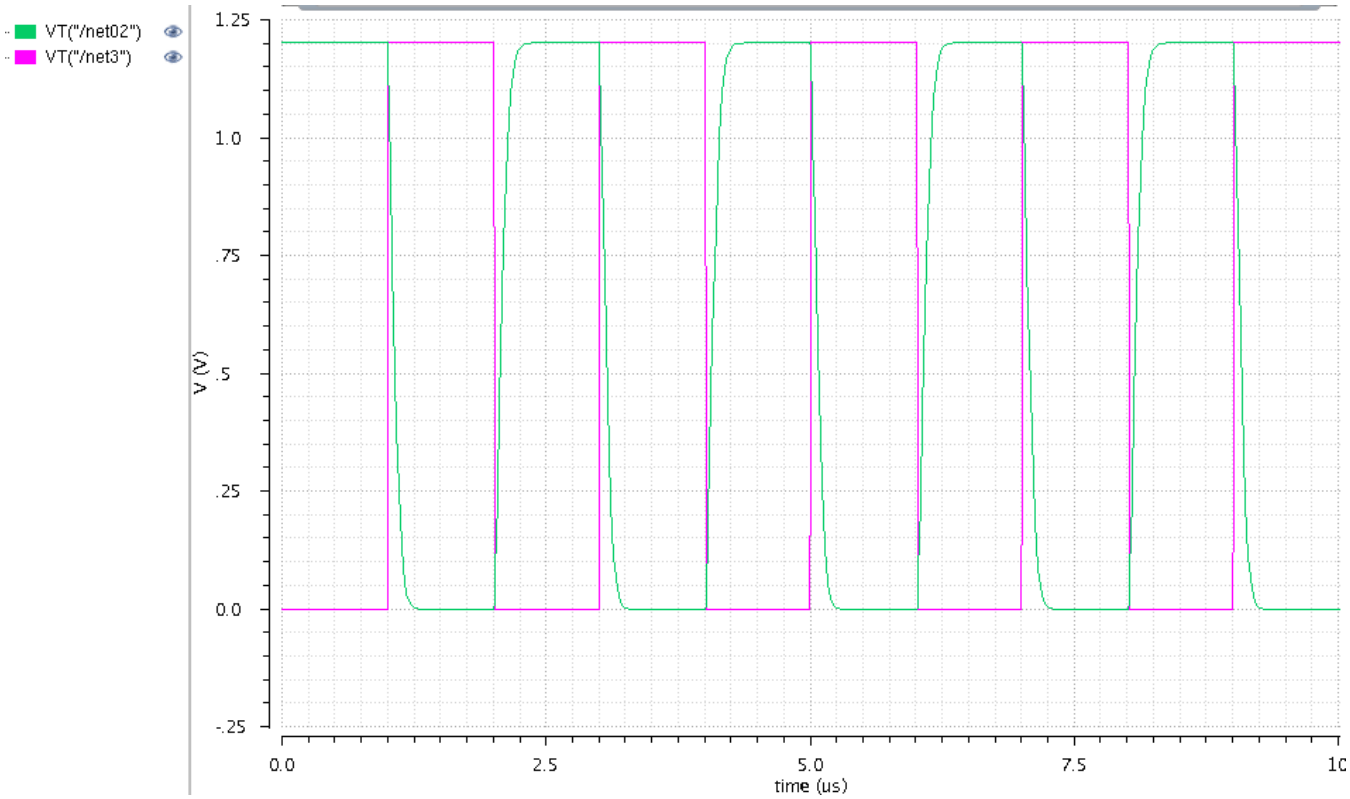
9. Check `tran`. Enter a stop time of 10u for 10μs. Check `moderate` under accuracy defaults. Make sure "Enabled" is checked at the bottom and click OK.

10. Now select Simulation→Netlist and Run or click the icon with the green stoplight in the right-hand-side toolbar. This will create a SPICE netlist of your design and run the selected simulations. Make sure there were no errors by looking through the Cadence command window and verifying that everything was successful.

11. To plot your results, go to Results→Direct Plot→Main Form. A small window will open up. In that window, select `tran`, then select "Voltage" under function. Now, without closing the window select a `net` or a wire, on your schematic. This will plot the voltage on the node as a function of time. Let's make sure the input is the square pulse that we wanted by clicking on that node. Then use the same method to see the output voltage as a function of time. This will overlay both plots on the same axis. If you want to create a new subwindow, click the appropriate icon on the top of the newly generated plot window. Your plots should look something like the one on the following page. The amplitude of the signal should be 1.2 V if $V_{DD} = 1.2$ V.

    For taking screenshots, please use a white background. You can change this by going to Edit→Properties or pressing "**q**".

12. You can also place markers on your plots by selecting Marker→Create Marker (or by pressing "**a**" and then "**b**" for the difference marker) in the plot window. Use this to find the time difference between when the input signal switches and when the output voltage equals roughly half of the supply voltage for both rising and falling output signals. These are respectively called the low-to-high and high-to-low propagation delays, commonly denoted as t_{pLH} and t_{pHL}. Finally, print and save a copy of the plot by selecting File→Save Image and then name the file as you wish. Make sure to add the file type in your filename (e.g., *.jpg, *.png). Or make a nice screenshot of your plot (Alt + Print Screen) that clearly shows the axes and waveform. Do not forget to label all your plots properly (axes, units, etc.); this also applies to all your homework sets.


VT("/net02")
VT("/net3")
1.25
1.0
.75
.5
.25
0.0
-.25
V (V)
0.0
2.5
5.0
7.5
10
time (us)

Lab 1 Part 2: Layout Design and Simulation

Report Due Date: February 26, 2018 at 6:00 pm

- You will learn how to create a layout of the inverter from Part 1.
- You will check the design rules and verify the layout of your inverter.

---

4 Overview: What is Layout?

The schematic view that you have created earlier is good for conceptualizing the circuit. However, the circuit must now be put in silicon, and this is what layout is about. The layout gives you a top-down view of the various components on your chip. Since it is difficult to view more than one layer in a top down view, the various layers are of different colors. To see which layer is which color, refer to the "LSW" window, which provides you with the color-to-layer mapping.

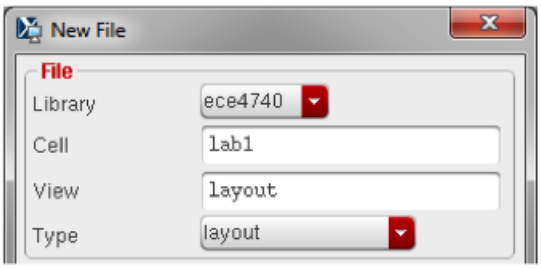
To make an electrical connection between different layers, for example to make a Metal1 layer touch a Metal2 layer, a so-called "via" must be created. A via is nothing more than a plug (or connection) between layers that forms a connection between two metals on different layers. Note that changing any of the dimensions of the via will lead to a DRC error, so leave the size alone.

If you've never done layout before, you are probably wondering why the connections between the various components exist in the schematic but the components seem to be sitting by themselves in the layout view. The main idea of layout is for you to complete the necessary connections, also known as routing the circuit. This is done by drawing rectangles in the metal layers and by making sure everything that is supposed to be connected in schematic is connected in layout.

1. **Starting Layout**

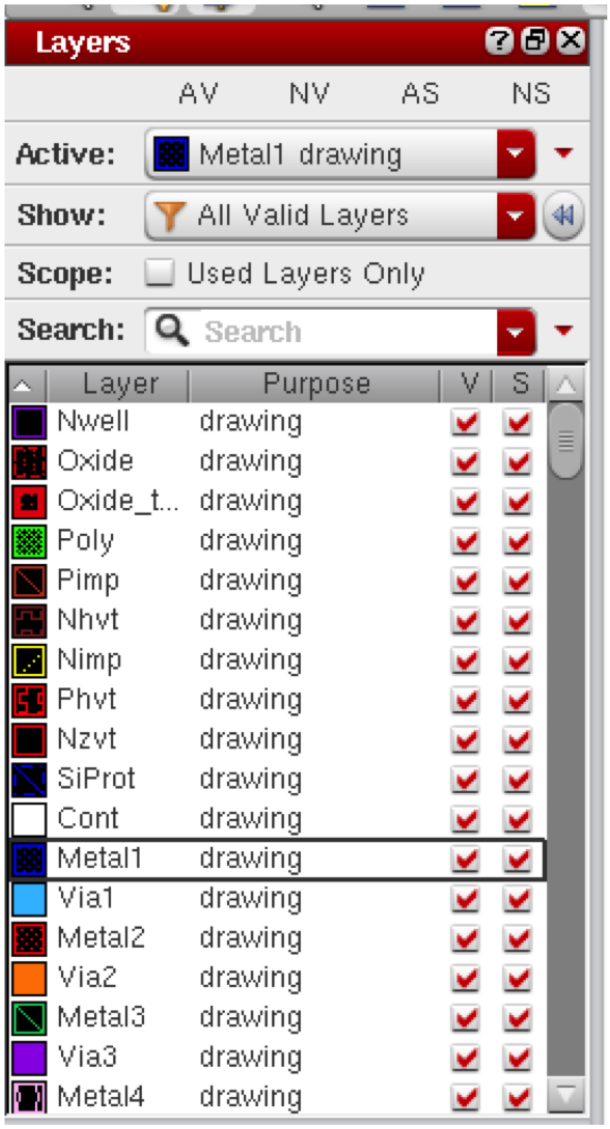
   In the Virtuoso Schematic window of your inverter, select Launch→Layout XL. Layout "Create New", Configuration "Automatic", Click OK.

   Type layout in the View field. Click OK. You should see "Virtuoso XL layout" window open. On the very top of the Virtuoso Layout XL window the title bar should say "Virtuoso Layout Suite XL Editing: ece4740 lab1 layout". This means that you are editing layout view of lab1 cell from `ece4740` library.



2. **LSW sidebar**

The LSW sidebar is shown to the right. This is your command toolbar, and is very important in facilitating and completing your layout task successfully. While it looks quite complicated at first, its function is very simple. The top box, currently set to Metal1 drawing, shows the active layer. This is the layer that gets drawn on when you draw a rectangle. The other buttons to note are the four buttons that say AV, NV, AS, NS. AV means All View, and will allow all layers to be seen. NV means No View, which will hide all layers except the currently selected one. After clicking NV, layers can be turned on by left clicking their button. AS and NS are similar and mean All Selectable and None Selectable respectively, and allows or disallows selection of particular layers. Strategically hiding and showing layers during routing will make your task much easier to finish.



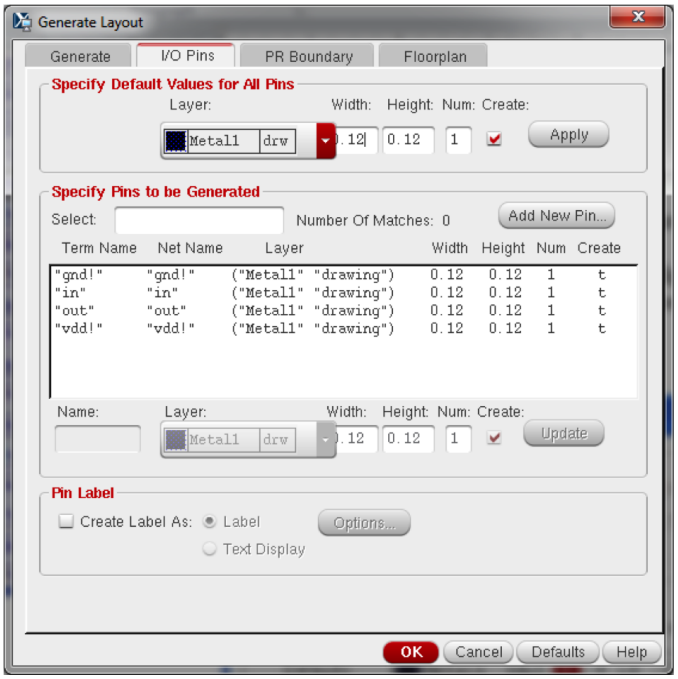
3. **Generating Layout from Schematic**

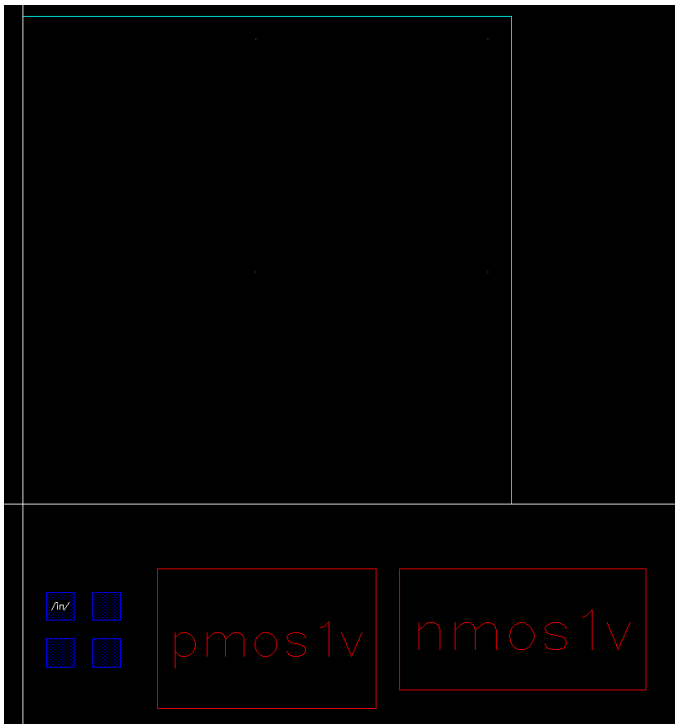
In the Virtuoso XL Layout window Select Connectivity→Generate→All From Source.

In the I/O Pins Section under Defaults. Make sure you select Layer = `Metal1 drw` and click "Apply".

There are several Metal1 layers, but only the `drw`, drawing layer is what we need. This is also true for all other layers we will use in the layout. Leave all other defaults. Click OK.

The initial pin and transistor placement in layout will look like the image below. The small blue squares are the `input`, `output`, `vdd`, and `gnd` pins created in Metal1 as you specified in the previous step.





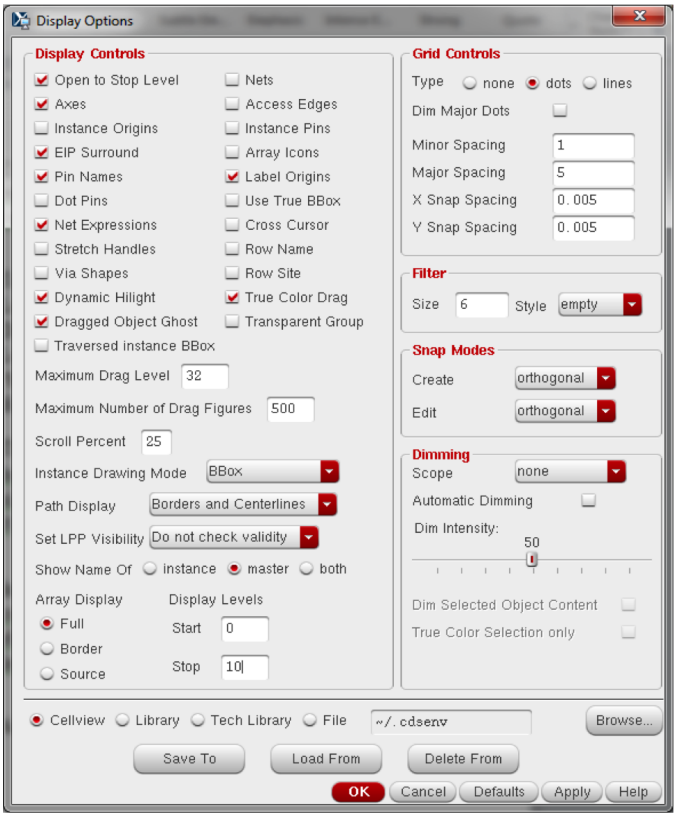
The transistors and pins are shown outside the PR bounding box (the aqua colored square), which is an estimate of the optimum size of the final layout. The bounding box may need to be re-sized to accommodate all components. An important concept to keep in mind during resizing is that standard cells typically have fixed height (so that power/ground rails line up correctly for routing purposes).

Currently, in the layout you will only see outlines of the layout cells for the `nmos` and `pmos`.

To see all layers: from the Virtuoso Layout XL window select Options→Display (or hit the "**e**" key) and set Display Levels: Stop = 10. Or you can press "Shift+F" and "Ctrl+F" to alternate between different stop levels.

The X/Y Snap Spacing defines how fine an increment you can size your layout shapes. Set these values to 0.005.

Now click OK.



4.1 Turn Off Gravity

Before you begin drawing metal, be sure to turn off the Gravity option. Press "**g**" to toggle gravity on and off (The Cadence command window will show a message that gravity has been turned on or off). Alternatively, go to Options→Editor (or press "Shift+E") and uncheck Gravity On. This will allow you to move your mouse freely and create a metal rectangle anywhere you want.

4.2 Tip to avoid DRC error

Select Options→DRD Edit. Set "DRD Mode" to Notify. Also, set the Hierarchy Depth to Current & Below, so you get notification of infractions on all sub-levels of the hierarchy. Finally, select the "Halos", "Rule Text", "Arrows" and "Violation Edges" check boxes in the "Interactive Display" section.

4.3 Zooming In/Out

As you continue working with Virtuoso Layout Editor, you will often find it useful to zoom parts of your layout. You can use the mouse wheel to zoom in and out, or simply remember these useful hotkeys:

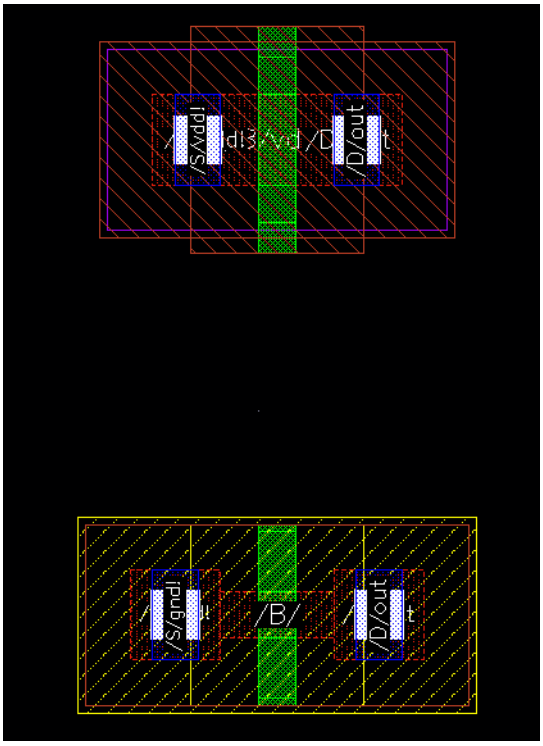
z	Zoom In selected area (hit "**z**", left-click and define the zoom area)
Ctrl+z	Zoom In by 2x
Shift+z	Zoom Out by 2x
f	Zoom fit

You should now be able to see multiple layers in the transistor instances as shown below. Another way to view and hide hierarchical layers is to use shortcuts: "Ctrl+f" for hide, "Shift+f" for display.

Move the `pmos` on top of the `nmos` to prepare for layout.

In general you should plan where the final location of your transistors will be.

Move the transistor instances to roughly their final location in the layout. This will make routing easier later on.



4.4 Transistor Layers

All the colors in the transistors represent the multiple layers that are involved in the fabrication process to produce them. The `nmos` and `pmos` are made of the following layers.

The `nmos` transistor (zoom in if you need a closer look) is made from following layers:

Layer description (LSW name) Visual appearance
n+ implant (layer name: Nimp) yellow diagonal stripes
p-well indicator (layer name: PWdummy) red rectangle just inside the yellow box
oxide (layer name: Oxide) red stripes
poly (layer name: Poly) green area
S/D contacts (layer name: Cont) white contacts
M1 S/D (layer name: Metal1) blue metal1 lines for S/D regions

Similarly, the `pmos` is composed from the following layers:

Layer description (LSW name) visual appearance
p+ implant (layer name: Pimp) red diagonal stripes
n-well indicator (layer name: Nwell) purple rectangle just inside the red box
oxide (layer name: Oxide) red stripes
poly (layer name: Poly) green area
S/D contacts (layer name: Cont) white contacts
M1 S/D (layer name: Metal1) blue metal1 lines for S/D regions

4.5 Drawing Wires

We can now begin routing metal to make all the necessary connections.

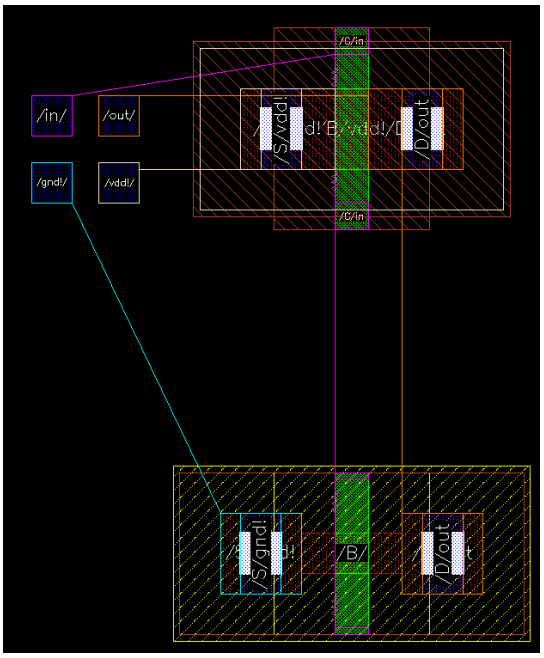
Drawing rectangles: Drawing rectangles in cadence is very simple, and forms the core of your wiring task. Each rectangle you draw in a metal (M1-M9) layer causes a sheet of metal to be formed in that layer. Drawing the rectangles correctly will, of course, correctly route your system. The rectangles must follow the design rules, so they cannot have too small an area, be too narrow, be too close to each other, be too big, etc, etc. For the exact rules, check the `gpdk090` rules manual. To draw a rectangle, press "**r**" while you are in the layout window, then point to one point of the rectangle, click, drag your mouse to the opposite corner and click again. This will draw a rectangle in your currently active layer. Alternatively you can create a path by tapping "**p**". Click once every time you want to make a turn in your path and double click to end the path.

As you know both `nmos` and `pmos` are symmetrical devices. This means that either terminal can function as source or drain. So you may think that it does not matter how you connect your transistors together, but this is not the case. Since your layout is based on a schematic, their connectivity has to match. Otherwise, you will not pass LVS (Layout vrs Schematic) later on.

To make it easier to see what nets should be connected together you may want to visualize your unconnected nets. To do so, select: Connectivity→Incomplete Nets→Show/Hide All...

You should now see different color lines connecting different nets in your layout, indicating they should be connected.

Connect the `nmos` and `pmos` gates together using the poly layer, and connect the drains together in Metal1. The source and drain of your transistors should already have a Metal1 layer contact.



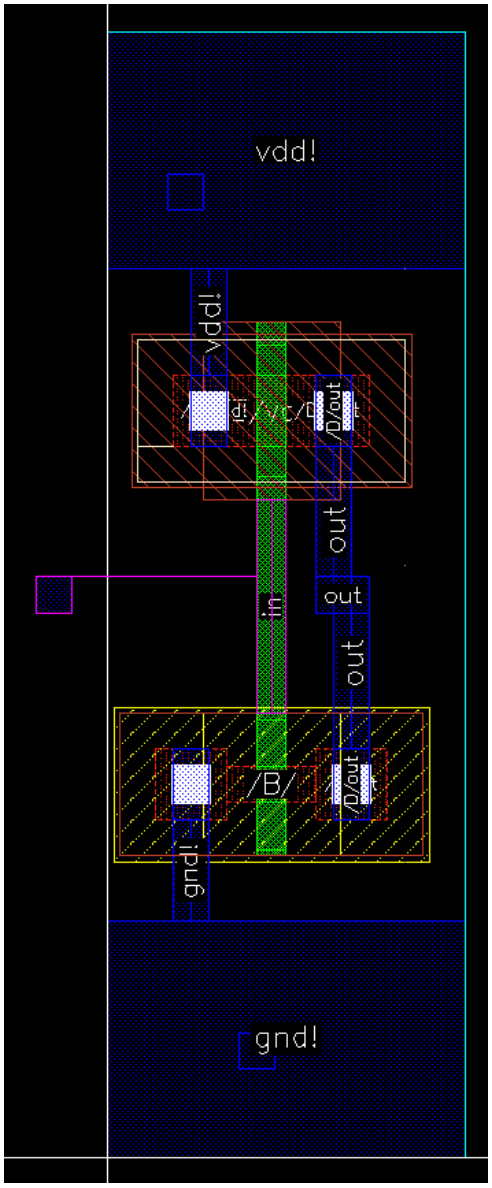
Stretch Tool - After drawing a short strip of metal the stretch tool can be very useful for extending the length of the metal. Press the "**s**" key and then mouse over the edge of the rectangle you want to extend, click and drag to extend or shorten its length. Make sure to not have the object you want to stretch already selected, by clicking on an empty space far from any layer. If you cannot select the edge by moving the mouse over it, try dragging around the edge that you want to select. In general it is easier to select what you want when there are fewer things to select around it. This is where hiding layers (as mentioned above) you are not working with comes in useful.

4. **Connecting Pins, Vdd and Gnd Rails**

   Connect the gates of `nmos` and `pmos` together with a strip of poly. Connect the two drains together in Metal1, this is the output of the inverter. Drag some Metal1 out to the right and connect it to the output pin.

We want to create `Vdd` and `Gnd` rails to make cascading of standard cells easier. In Metal1 draw large `Vdd` and `Gnd` rails above the `pmos` and below the `nmos` respectively. Make the rails around 0.8μm wide; this is so that we can fit substrate contacts later. Use the ruler tool to measure off distances, hit "**k**" then click and drag to measure distances. To clear all ruler marking use "**K**" ("Shift+k"). Connect each transistor's source to the appropriate rail with Metal1. Now place the `Vdd` and `Gnd` Pins that were generated over the respective rails.

Your layout should look like the one to the right.



We now need to connect the input pin to the gates of our `nmos` and `pmos` transistors. If you haven't already noticed the input pin is in Metal1 while the gate of the FETs are in the poly layer. We can connect these two layers through the use of vias.

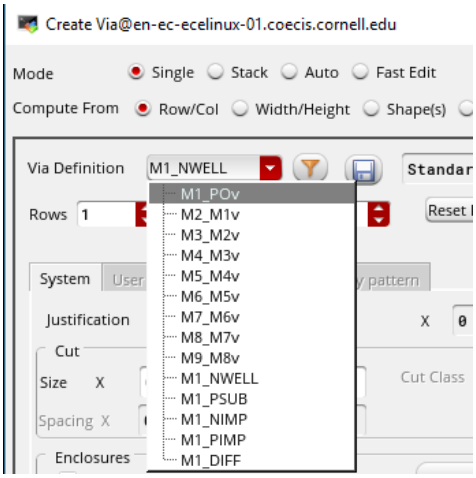
5. **Vias**

   Vias connect two different metal layers that are next to each other. If you want to connect layers that are further than one layer away select "stack".

   To add a via go to Create→Via or press the "**o**" key.

   The "Create Via" box will pop up.

   Under the Via Definition drop down menu select "`M1_POv`". This means Metal1 to Poly via. As you can see you can add a via to connect any two adjacent metal layers, plus a few others we will see later.

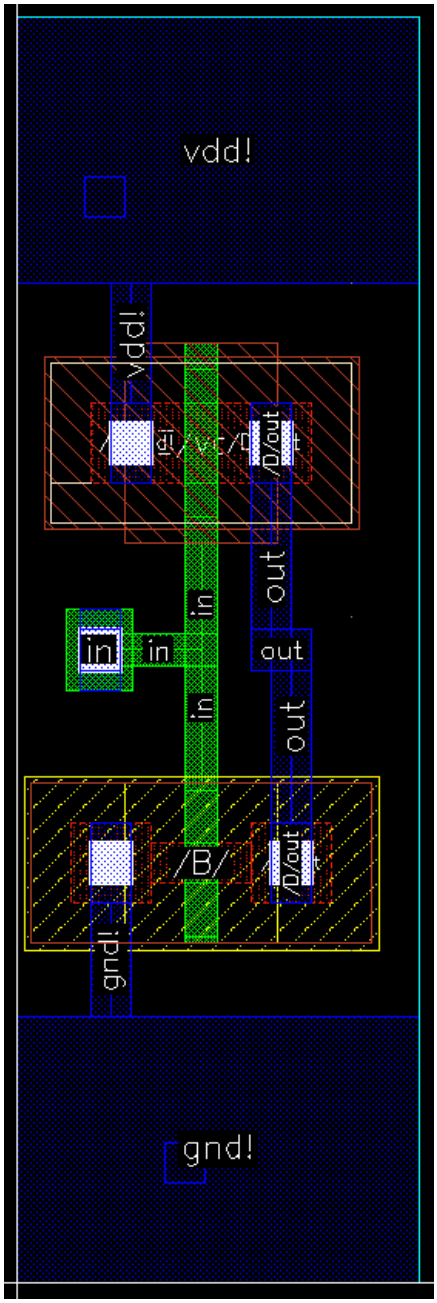


Mouse over the layout window and click to add an instance of the via to your layout. You can continue clicking to add as many vias as you need. When done adding vias hit escape.

Here we will only need one via. Place it between the input pin and the gate.

The white part is the actual via, a "tunnel" between metal layers. You can see there is also green and blue; these are actual rectangles of poly and Metal1 layers.

Connect the input pins and poly gate to the via to finish your connections. Your layout should now look something like the one to the right.



6. **A Note about Routing**
   As you go up in metal layers they are thicker and less resistive, therefore they can propagate a signal faster. BUT the vias that connect metal layers are very resistive and slow. The slowdown of the vias may not make up for the speed up of higher metal layers. A general rule of thumb is to route local signals in low metal layers and global signals in higher metal layer. Another useful rule of thumb is to make all even metal layers east to west only and all odd metal layers north to south only, or vice versa. This will make routing of more complex circuits much easier as you will not have to think about how to route around some metal that is in the way of your path.

7. **Nwell, Substrate and Well Traps**

   We are working in an N-well process. The wafer is a p-type substrate, i.e. you can consider all the black area in the background as a p-substrate. N-mosfets lie in a p-well, so we can simply use the wafer substrate as this p-well. P-mosfets lie in an n-well so we need to create an n-well for our `pmos` to lie in.

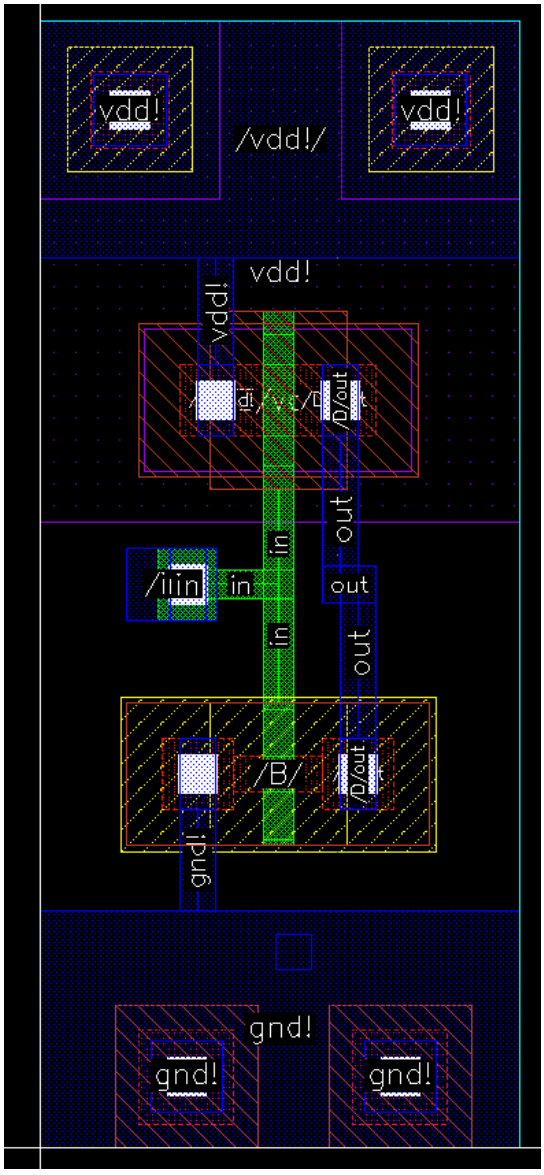
   In the LSW window select the N-well layer and draw a rectangle around the `pmos`, also make sure to draw the N-well around the `Vdd` rail since this is where we will place our n-well tap.

   Now we must connect our substrates to `Gnd` and `Vdd`. This is basically connecting the bulk of our nmos and pmos to `Gnd` and `Vdd` respectively. The substrate taps are a via that you can select from the create via menu just like other vias.

   To add a `pmos` body substrate, tap "**o**" and select `M1_NWELL`. Place two taps within the `Vdd` rail boundry. Make sure this tap is connected to both Vdd and the nwell layer.

   Now let's add substrate contacts for the `nmos` body. Click "**o**" and select `M1_PSUB`. Place two taps within the `Gnd` rail boundary. Make sure this tap is connected to `Gnd`.

   A general rule of thumb is to add many substrate taps when possible. Your layout should now look like the one to the right.



8. **DRC (Design Rule Check)**

   Now that we have finished drawing our layout we must check that our layout meets all the design rules. The design rules are provided to increase the chances of correct fabrication of your circuit in the presence of inevitable process variations, such as misalignment of masks, variations across wafers, etc. The tool that does this check is called DRC. DRC checks the layout to ensure that the layout conforms to the design rules for the technology. For instance, the metals lines must be separated by a certain distance to accommodate process variations and limitations. It is a good idea to run DRC often as you go to ensure that each small change of your design conforms to all of the rules.

   To run DRC, go to Verify→DRC. Make sure the Rules File field is set to `divaDRC.rul` and the Rules

Library field is checked and set to `gpdk090` . Click OK. Check the main Virtuoso command window to see the results of DRC. Here you will see a printout of each DRC error. In the layout window you will see flashing white lines indicating where an error exists.

To be able to find the location of each error it is useful to use Verify→Markers→Find. Check the Zoom to Markers checkbox and click next to move to the next marker. Sometimes the flashing white lines can get annoying! To delete DRC markers go to Verify→Markers→Delete All.

Fix all DRC errors. Run a final DRC to make sure your layout is DRC clean.

9. **LVS (Layout versus Schematic)**

LVS compares the extracted layout with your schematic(s). When the design grows large, it is quite common to make silly mistakes such as shorting two lines which are not supposed to be connected, leaving floating nodes, etc. LVS helps you to find these problems by comparing the schematic net-list with the extracted net-list to make sure there is a match.

The first step to run LVS is to *extract* the circuit that is represented by our layout, so it can be compared with the circuit from our schematic. To do so, go to Verify→Extract. Make sure that the Rules File is set to `divaEXT.rul`, and the Rules Library field is checked and set to `gpdk090`. Click OK. This will create a new view in our cell called `extracted`.

Now, we are ready to run LVS. Go to Verify→LVS. Make sure both schematic and extracted are checked. The Library and Cell field should be `ece4740` and lab1 respectively (or whatever you named your library and schematic/layout). View should be `schematic` and `extracted`, which we just created.
Make sure the Rules File is set to `divaLVS.rul`, and the Rules Library field is checked and set to `gpdk090`.
Click Run in the Diva LVS window. When LVS has finished a dialog box should pop up saying your job has succeeded.
In the Diva LVS window click the Output button. This will open up the output log of the LVS results. Scroll down and look for the words "The net-lists match".

If your net-lists match you have done layout successfully. If you get unmatched net-lists, then your layout does not match the schematic. There will be details about what net-lists don't match, use this to determine what needs to be fixed. Another useful tool is the Error Display button in the Diva LVS window. Here you can set a highlighting color and then click next to zoom into the location of your layout that does not match. After fixing your layout, do not forget to run Verify→Extract again before repeating LVS.

Last updated: January 29, 2018 by Oscar Castañeda/cj/cs

Lab 1 Part 3: Extraction Tips of CMOS drivers

Report Due Date: February 26, 2018 at 6:00 pm

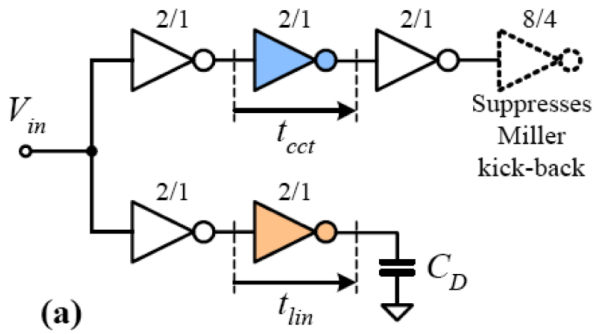
- You will perform parameter sweeps in Cadence Virtuoso.

As you may well know, the accuracy of hand analysis for even relatively simple digital systems depends on the accuracy of our circuit models, which may often be quite involved. A good model must be reasonably accurate, yet tractable in order to provide intuition into the operation of a circuit without getting lost in the complicated physics and mathematics that clutter the design process. Once a first-pass design has been achieved through simplified models, then second and third order effects may be taken into account by simulation to optimize the design. This handout is meant to provide exercises in helping you extract meaningful linear models from Cadence.

Assumptions: For these exercises, you may use VDD $= 1.2$ V and a square wave input signal with the following parameters: $t_{\text{rise}} = t_{\text{fall}} = 10$ ps, $T_{\text{period}} = 20$ us, delay time $= 5$ ps, pulse width $= 10$ us.

Problem 1: Delay and Energy Capacitance

In this problem, we want to model the equivalent input capacitance for a unit sized inverter (in this process $W_{\text{min,p}} = 240$ nm, $W_{\text{min,n}} = 120$ nm, $L_{\text{min}} = 100$ nm). As shown in figure below $W_{\text{p}}/W_{\text{n}}$ is specified for each inverter. For example, $8/4$ sizing for last inverter means $W_{\text{p}} = 8 \times 120$nm and $W_{\text{n}} = 4 \times 120$nm. We have the following experimental setup to determine the delay through an inverter:



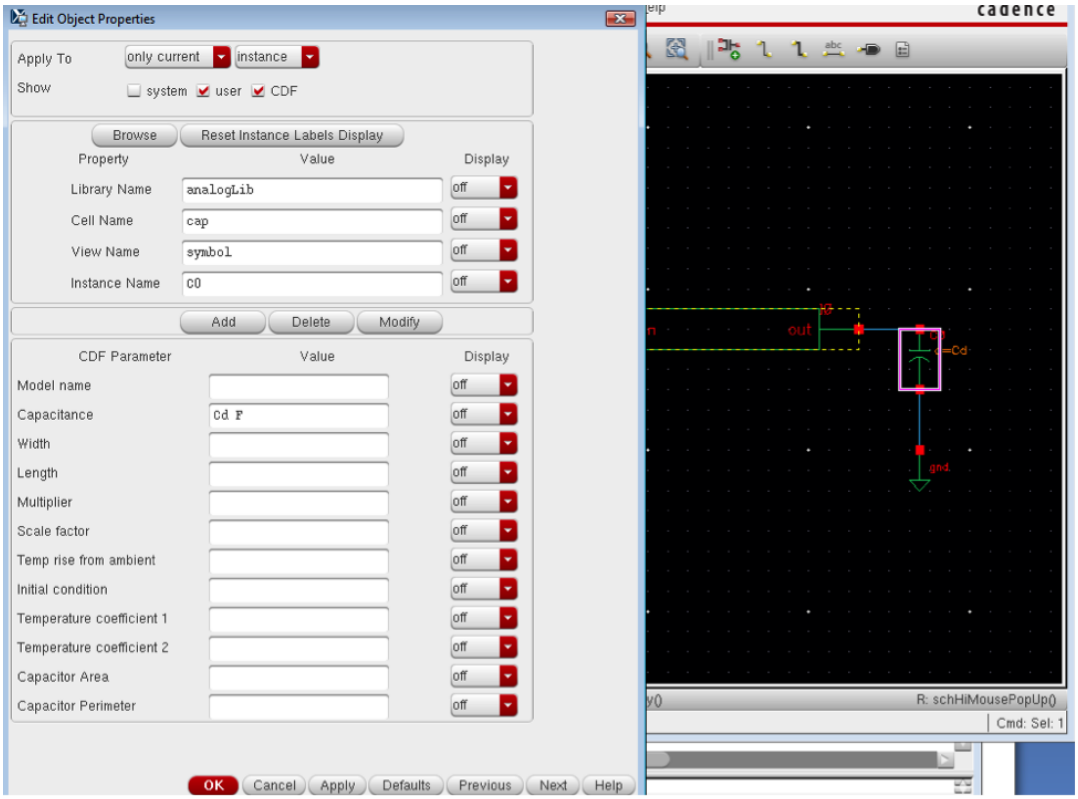
We wish to extract the equivalent delay capacitance of a minimum-sized inverter. More specifically, we will look at the delay of the second inverter in order to determine the equivalent input capacitance of the third stage, which has minimum size. In order to simulate a realistic Miller effect on the input capacitance of the third stage, we add a large fourth inverter stage to act as a large capacitance to reasonably slow down

the output transition of the third stage. The first inverter stage is added in order to simulate a realistic input signal that might be found somewhere on any reasonably designed circuit. Use your simulation setup from Part 1 for the following part.

(a) Determine the capacitance C_D from Figure (a) in such a way as to match the average propagation delay of the blue inverter (i.e., an inverter loaded with another inverter with that of an inverter loaded with a large capacitance).

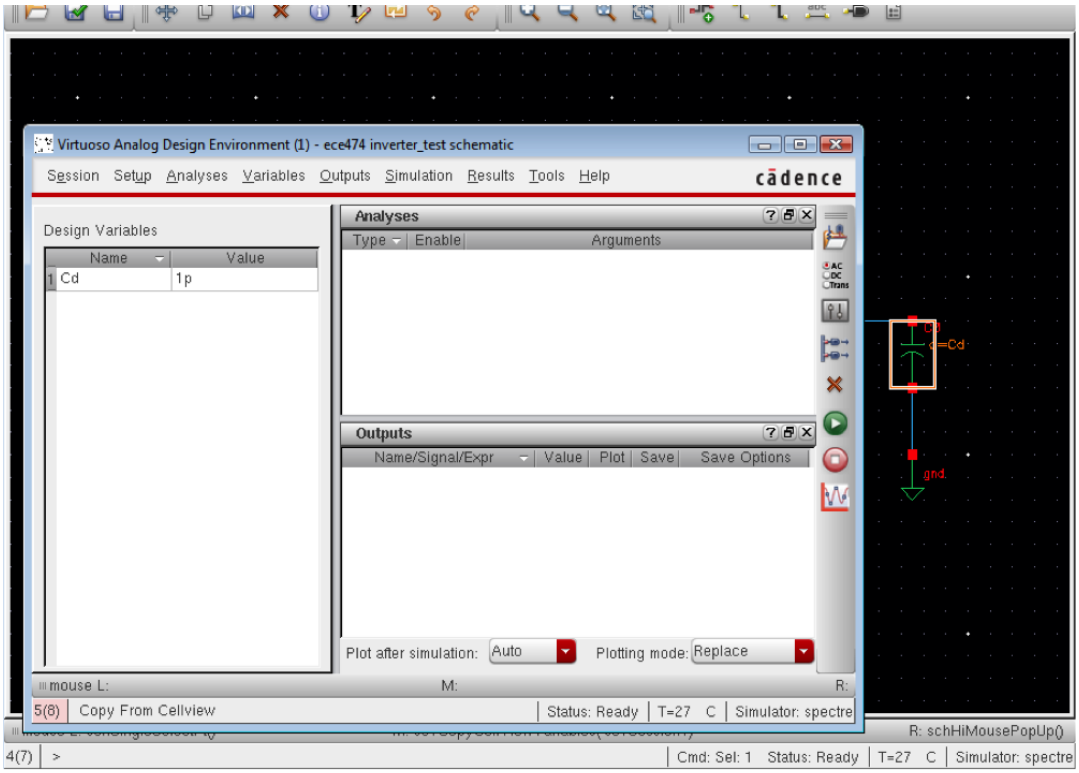
Hint: You will need to make good use of the parametric sweep simulator (In Analog Environment Window: Tools→Parametric Analysis) and the calculator in this part. To start off, we are going to set up and run a parametric transient analysis with parameter C_D.

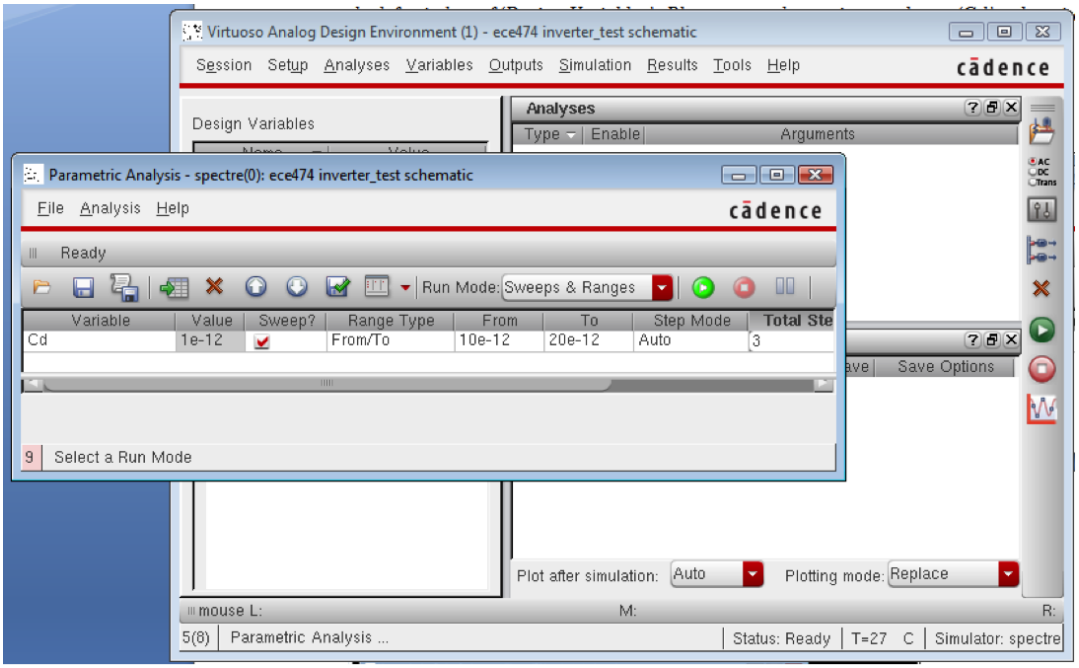
To do this, assign a variable to `CD` *in schematic, like Cd, shown in the following figure.*



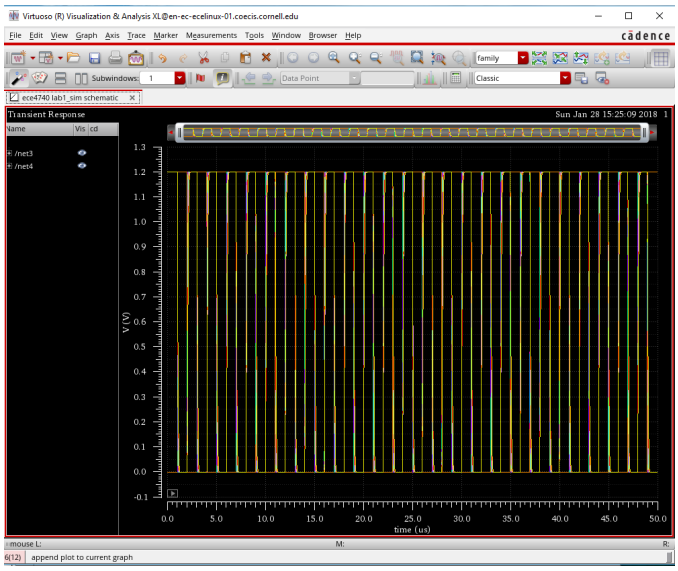
Then go to ADE and load the simulation state that was created in Lab 1 Part 1. Click Variables→Copy from Cellview. You will see the "Cd" appear in the left window of Design Variables. Please remember to assign a value to "Cd" (you can use 0). Otherwise Cadence will not allow you to start the simulation. Before opening the parametric sweep screen, make sure you set up a transient analysis from the ADE screen with a run time of 50µs and moderate accuracy. Also, select the output voltage of the blue and orange inverters as outputs of the simulation. To do so, select Outputs →Setup. A window will open. Click "From Design", then you can select the net you wish to plot its output. This will add the net to a table on the right side of the window. Select the net in the table to change its name (this is optional); click Apply to save your changes. You can do the same for plotting more outputs.

To do the parametric sweep, click Tools→Parametric Analysis. A window to do the parameter sweep will pop out, as shown in the second figure on next page. Then you can choose the variables you want to sweep and also the sweep range in the analysis. Try to use the same range and step count from the example pictures and then you can play around as needed. Make sure to sweep C_D *with a fine enough resolution.*

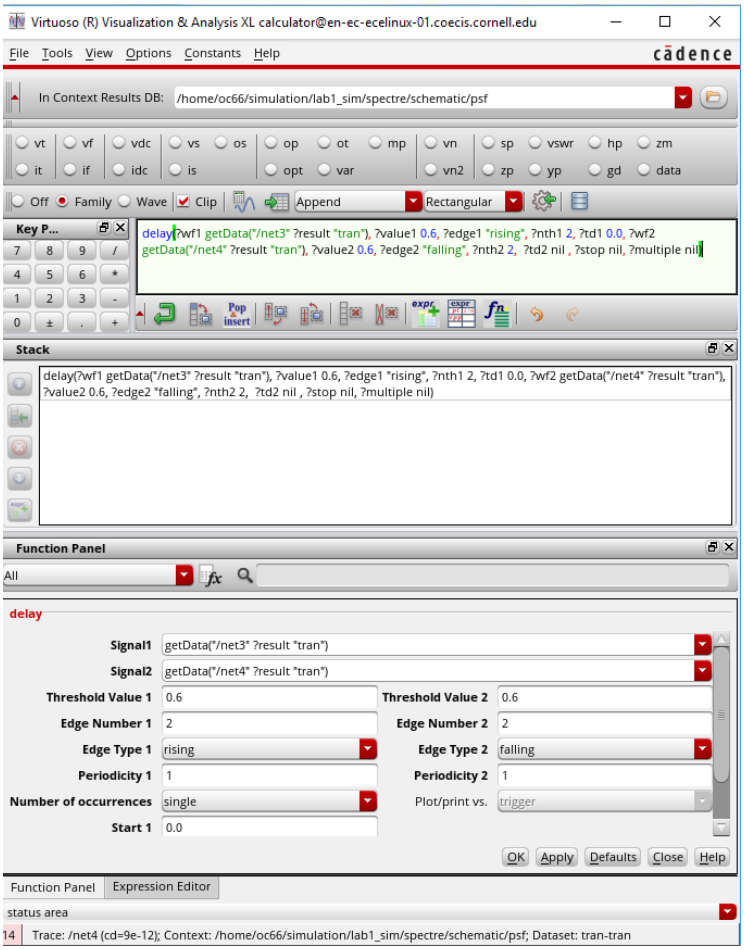




After running the parametric sweep with the green stoplight from the "Parametric Analysis" window, you will need to plot the average propagation delay as a function of C_D. After making the plot that is below (on the next page), click Tools→Calculator to see the Calculator window.



There is a "delay" function in the calculator that asks for parameters as shown in the following figure. Make sure that, in the radio buttons with the options "Off", "Family" and "Wave", the option "Family" is selected. Inside the "delay" function, select the Signal1 *field and then, in the plot window, click on the waveform of the input signal (faster signal). Repeat the same with the* Signal2 *field and the waveform of the output signal (delayed signal).*



Threshold Value 1 and *Threshold Value 2* should be set to 0.6.

Edge Type 1 and *2* correspond to the type of edge you wish to examine for each respective signal. If you want to find the high-to-low delay, set Edge Type 1 to "rising" and Edge Type 2 to "falling" and vice versa for the low-to-high delay.

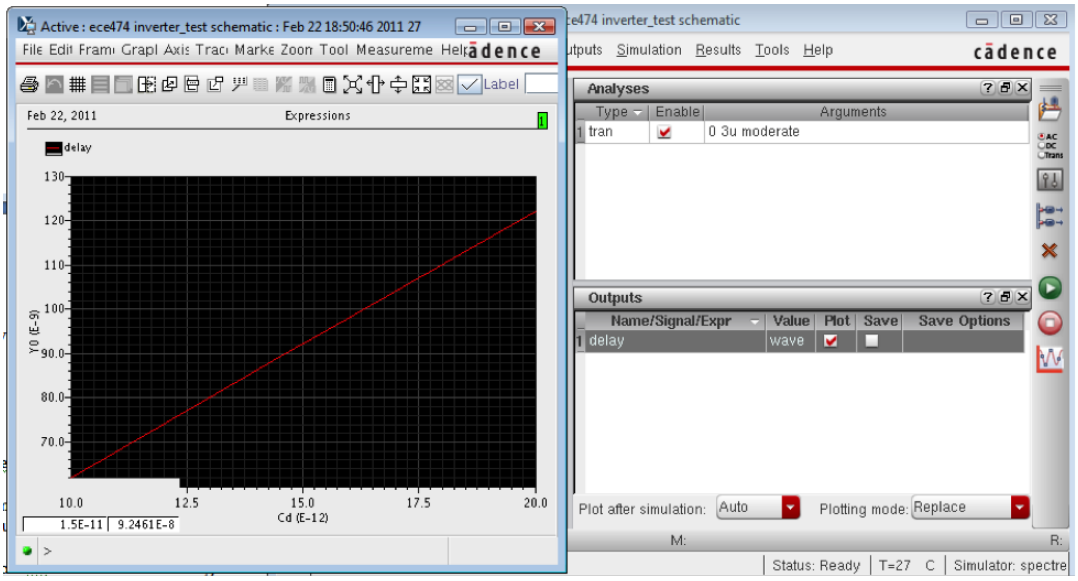
Edge Number 1 and *2* correspond to the occurrence of the given edge in the parameter stated under Edge Type. The Edge Number you pick should be the same for both signals. For example, if you wish to examine the delay from the 2nd rising input edge to the 2nd falling output edge, you would pick *Edge Number 1* and *2* to be 2 and *Edge Type 1* to be "rising" with *Edge Type 2* set as "falling".

Setting these parameters appropriately should give you the delay across one inverter.

Refer to the Help menu for further information about the calculator.

You will need to use this to find the average propagation delay across the inverters of interest as a function of C_D. After that, you may find the intersection of the delays to find the value of C_D that gives equal delays across each inverter.

After you finish setting all the parameters, you would see an expression in the buffer when you click "Apply", shown in the above figure. Then you click the icon for "Evaluate the buffer", which is to the right of the "Off", "Family", "Wave" radio buttons and the "Clip" checkbox. This will add to your current Plot window a new plot of the delay versus the "Cd" value.

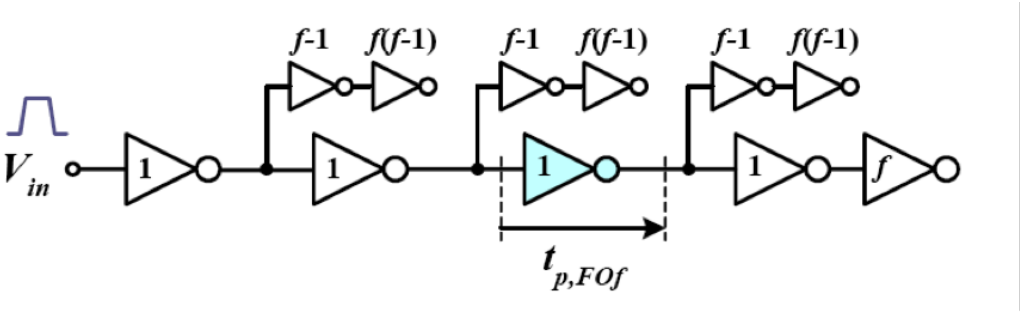


For the repeatable use of this expression, you can save this expression to your outputs. Go to "Outputs" in ADE, choose "Setup". Then click "Get Expression" to get the expression shown in the buffer of calculator.

Save the plot of the parametric sweep from which you can determine C_D.

Problem 2. Delay Model Calibration

The aim of this exercise is to develop a linear delay model for the propagation delay of a minimum sized inverter. We know that there will never be an isolated inverter in any practical circuit and therefore, we cannot model the intrinsic delay of an inverter by finding its delay in isolation. We will use the following test circuit for extraction of the linear propagation delay model:



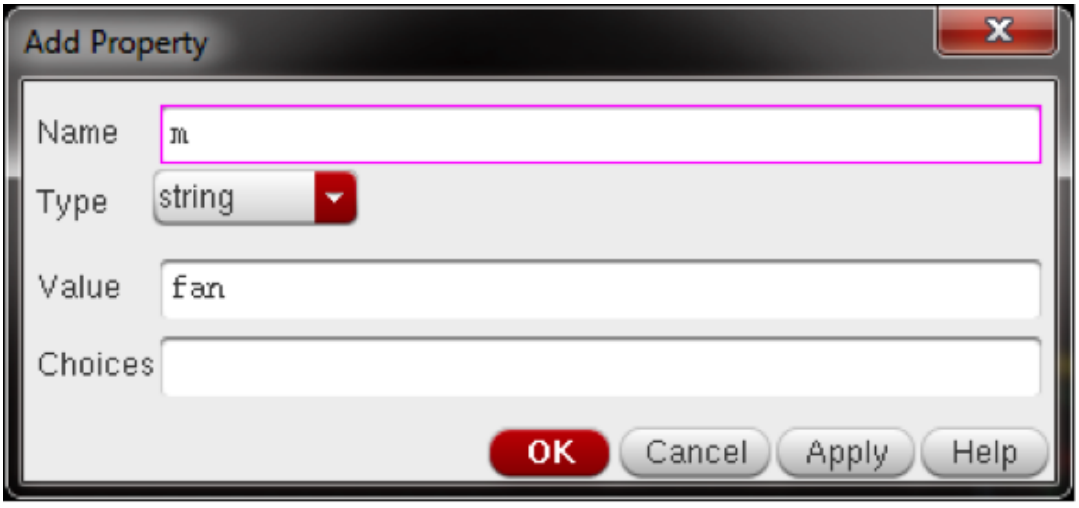
Using this circuit, we can get an accurate approximation of the real propagation delay of an inverter by finding the delay through the third stage. As before, the first two stages provide a realistic input signal, while the last stage suppresses extreme nonlinearities caused by the Miller effect on the input capacitance of the fourth inverter. The branching stages are sized as $f - 1$ to make the total fanout of any inverter in the chain equal to f while still allowing each inverter in the chain of interest to be minimum sized.

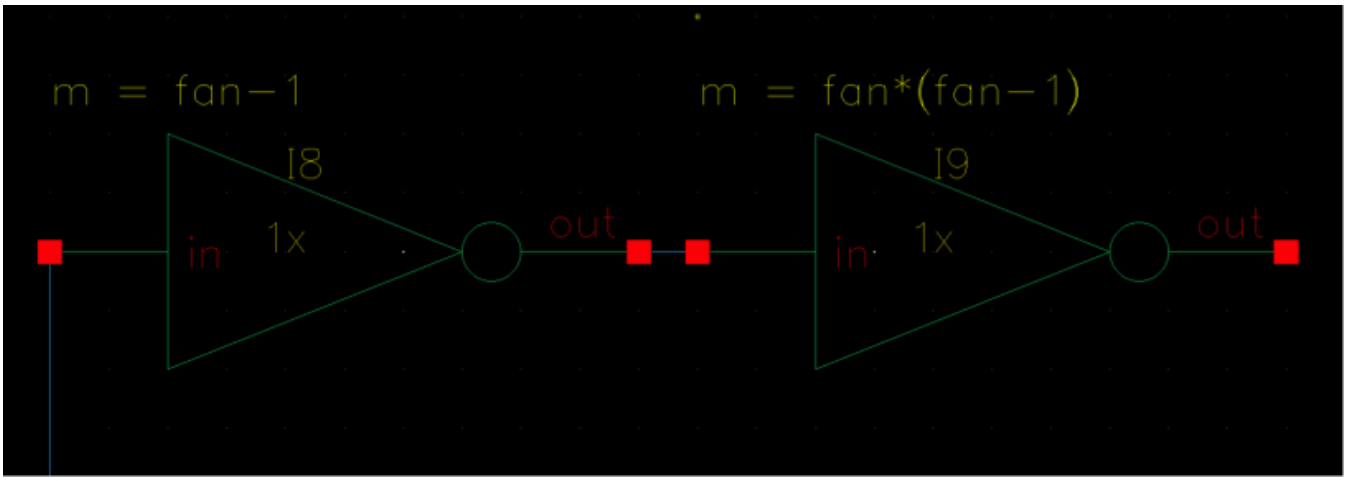
From this setup, we can now find the required parameters of our linear delay model by finding the average propagation delay as a function of f:

$$t_\mathrm{p} = t_\mathrm{p0}\,(1 + f/\gamma)$$

Use this to find t_p0 and γ. You will need these values for your project.

*Hint: You will again need to make use of the parametric analysis simulator and delay function in your calculator. To try other fanout values, add the "**m**" (multiplicity) property to your unit inverter. Select an inverter in the schematic, push "**q**", and select "Add". Give the property a variable name in the "Value" field if you want to use it for parametric analysis. Set the "Display" menu to "both" to keep track of the effective size of each inverter. You may need to manually add the variable to ADE.*



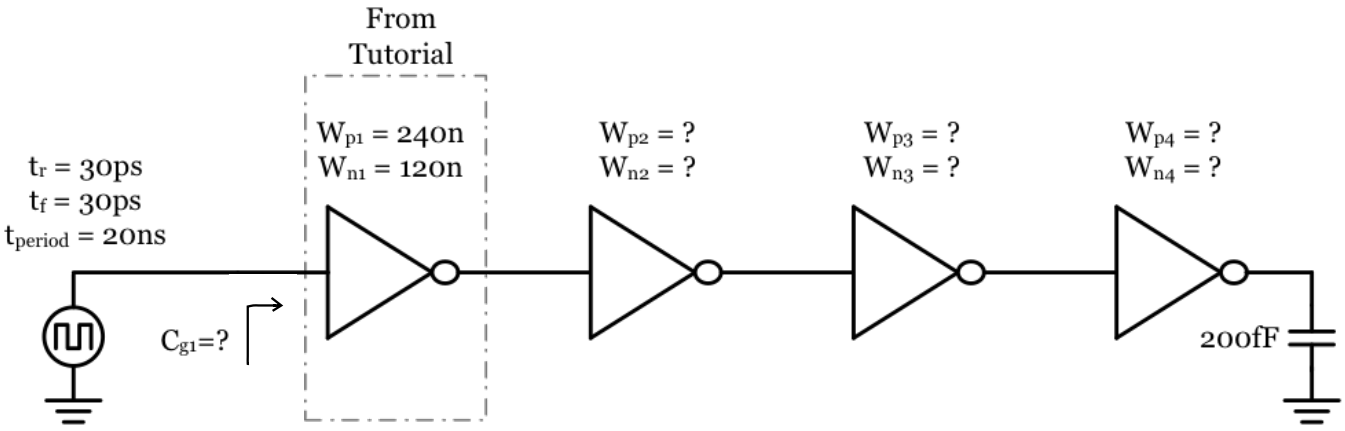

m = fan−1
I8
in
1x
out
m = fan*(fan−1)
I9
in
1x
out

Lab 1 Part 4: Demo Parameters and Report

Report Due Date: February 26, 2018 at 6:00 pm

- You will design an inverter (buffer) chain and use Virtuoso to minimize delay and area.
- You have to complete a PowerPoint report template.

Your goal is to design and layout a four-stage buffer (4 series inverters) with minimum delay and area (we are going to assess the area–delay product). The buffer will drive an external load of 200 fF. The input will be driven by a clock source switching from 0 to VDD and back to 0 with a linear rise/fall time of 30 ps without a source resistance. You will use your C_d value from the previous part for C_{g1}.



Area: The area is calculated by the smallest rectangular bounding box in μm^2.
Delay: Calculated by the average of t_{pHL} and t_{pLH} using the 50 % level definition.
Benchmark: The area–delay product in units of $\mu m^2 \cdot s$.

- The first stage must be a minimally sized inverter (240 nm /120 nm)
- Use only minimum length transistors for your design
- Design the inverters in a hierarchical fashion (i.e. layout for every inverter)
- Assume VDD = 1.2V
- Power rails must be at least 400 nm wide
- Do not use integrated body tie-downs

- Designs must pass DRC and LVS
- Make sensible symbols for each inverter and the entire buffer chain

Hints:

- Use fingers
- Use an orthogonal metal convention (e.g. M2 goes left/right, M3 goes up/down)
- The transient simulation should be between 50 ns and 100 ns (whatever is clearest)

5 Grading consists of a real-time evaluation by a TA and a report

Real-time Evaluation (to take place in PHL 314 on February 26, 2018, 8:40pm – 9:55pm):

An evaluation time will be scheduled with your TA. First, the TA will verify the buffer delay, area, DRC and LVS. Then the TA will ask you **individually** to perform one of the tasks below. The selection will be random and not volunteered. Possible tasks:

- Rotate the layout of the entire buffer by 90° and then pass DRC and LVS
- Re-simulate the delay using a resistive load or a different VDD
- Perform a transient simulation with a sine wave input
- Add one of the inverters into the signal chain and re-simulate delay

You will receive an individual grade of 0-5 based on the evaluation:

5: finished task quickly without TA help
4: finished task with little TA help
3: finished task with significant TA help
2: some clues of how to perform the task
1: little clue of how to perform the task
0: absent without reason; doesn't know how to start

Report: Fill out the PowerPoint report template posted on blackboard. Make sure your schematics and plots are clear and legible. You can label Cadence screenshots or draw the schematics in a program of your choice. The objective is to create a concise report that can be quickly evaluated by a TA.

Turn in a zip file with your *schematics, symbols, layouts, and the report*. Name the zip file **Lab1-(NetId).zip**. The report is due on February 26, 2018 at 6:00 pm.

Important: *Do not cheat!* We are going to randomly try out 10 out of all the submitted designs and compare it to the reported numbers. In case you fudge on benchmark numbers, copy the schematic and layout from previous years' lab, or the design does not pass DRC or LVS, you will get ZERO points for the entire lab.

Last updated: January 29, 2018 by Oscar Castañeda/cj/cs