Guide to Multi-Cell Layout
A 410 Lab Help Document

Layout Rules for Multi-Cell Circuits:

Hierarchical Design

When constructing higher level circuit blocks from lower level cells, you **must must must** maintain a hierarchical structure. That is, you must *instantiate* your cells rather than *copying* them into the new cell. To add a cell instance in the layout editor, press 'i' and then click on 'browse' to select the cell from your library. You can use *shift-F* to peak into the internal layers of the cell and *control-F* to hide the cell polygons. Do not flatten your cells!

You must add power and ground rail pins to each cell, even if it is composed of instantiated cells which already have power/ground pins.

Cell Placement

When combining multiple instances to create higher-level functions, place cells in rows so that the vdd/ground metals are adjacent to (just touching) each other. If the cell boundary guidelines have been followed, this will merge all well/select layers and power rails into continuous layers.

Power Rail Widening

Combining more cells means an increase in the current flowing though the power supply rails. When combining multiple cells into larger blocks, you need to ensure your power rails can handle the peak current load of all circuits attached to the rail. Without a well-thought floorplan of the overall chip, this is difficult to determine. To analyze this is you must estimate the max current though any row of circuits (a row shares power supply rails) and estimate the length of the row. Then, knowing the sheet resistance of metal1 you calculate the max resistance of the power rail (sheet resistance x length/width). Then calculate the maximum voltage drop at max current. Finally you set the height of the power rail so the maximum voltage drop is below 0.1V.

If you are still confused you are in luck. We have analyzed this for you and estimate that the 3μm (10 lambda) power rails used in your basic cells should be sufficient for normal current loads of rows up to 1mm wide. So, you do not have to widen the power supply rails.

Signal Routing

With all the cells placed you can begin routing signals. When possible, keep routing within the cell boundary using poly and metal1. For signals that can not be routed within the cell boundary, you can use vertical metal2 to get the signal out of the cell boundary to a horizontal metal1 routing trace that is placed on the top or bottom of the supply rails. The metal2 has to be used **only in the vertical direction**. To connect from metal1 to metal2, use appropriate via contacts. It is advisable to keep the width of the metal1 and metal2 as 4λ since you will need this size to include contacts (vias) between the two metals. As a general rule, try to place all outside routing either on the top or the bottom of your cell but not both (this facilitates tighter stacking if you horizontally flip alternating rows of cells in higher level circuit blocks.

Connecting Power Rails for Multi-row Layouts

If you build a circuit block that uses more than one row of cells, you MUST connect the power supply rails (VDDs of each row, Grounds of each row) before you can pass LVS. There are two ways you can do this.

1) Virtual connections:

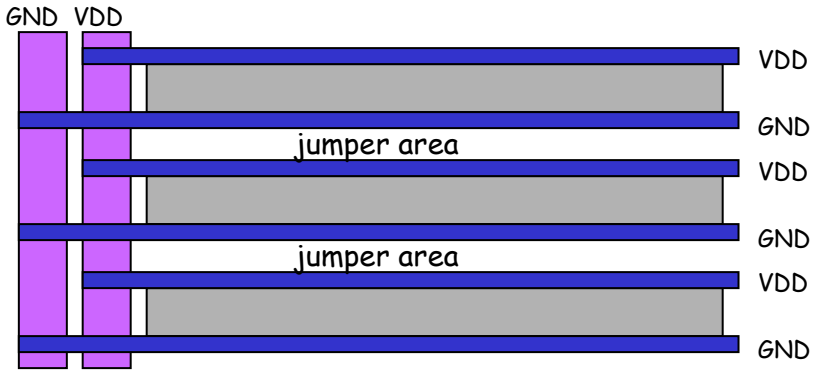
You can label each VDD/Ground rail (typically within a row of cells) individually and join those using virtual connections. This method is preferred if you know single rails are not needed in your cell hierarchy, i.e., that the power rails will all be physically joined at a later stage in the design.

To use virtual connections, when getting the extract view from the layout, check the option "Joint nets with same names" in the extractor window. If you do not do this, you will get LVS errors for having multiple nets with the same name.

2) Physical connection by polygons:

It is sometimes preferable to have all of the VDD/Ground connections within a cell physically joined together. This often requires some knowledge (or plan) of higher levels in the layout hierarchy. If you know the power rails will be joined at a higher level in the design, it would be preferable to use virtual connections than to place temporary physical connections that must be removed later.

To make single power rails for multi-row cells, you can, for example, use two vertical routes of metal2 (one for VDD and one for Ground) to the left or right of the completed cell block and then add horizontal metal1 and vias to connect the individual VDD/Ground rails. This method is illustrated below.



Optional Physical Design Techniques:

Cadence includes a set of tools called '*mosaic*' that will not be taught in the class but can be used, at your option, to simplify some aspects of physical design with multiple cells. Refer to the Layout Editor Manual, linked on the class website, for the physical design using mosaic tools.

Creating multiple instance of the same object	:pp.234-240.
Creating multiple pins	:pp.247-250.
Creating array copies	:pp.283-287.

Edit-in-Place Technique
Important: do not flatten your cells!

When creating a complex layout with hierarchical (non-flattened) instantiations of lower-level cells, you might find that you want to immediately see how making a change in an instantiated layout affects the higher level one. Additionally, you may want to quickly make a change to an instantiated layout without having to open a new window editing window. In these cases, you can use the "Edit in Place" command.

1. Select the instance that you wish to edit by left clicking on it
2. From the menu, select Design => Hierarchy => Edit in Place. Alternatively, you can press "Control-x".
3. This will put an orange box around the instance which you may now edit.
4. If you want to descend further into the hierarchy, repeat steps 1-4 as necessary.
5. When you are done editing and wish to save the changes, check and save the design.
6. Select Design => Hierarchy => Return to return the editor scope to the top-level layout. Alternatively, you can press "b".
7. If you made changes that weren't saved, Cadence will prompt you for a save. If you don't wish to save, make sure to select "No". This will restore the layout to its original form.

Even though you can still see all of the details of the high-level layout, you can only edit within the area of the orange box. Any change you make in this instance (such as moving a layer or adding a pin) is automatically reflected in ALL instances, but not made permanent until you save.

Be very careful while using the tool. Any change you make is saved to the base layout view that you are modifying, meaning any and every instantiation of the cell in your design library will be modified as well. It is generally only a good idea to use this technique if a cell is instantiated only once in your design, or if all instantiations of the cell exist within a single layout. In this way, all changes you make are kept local to the current layout you are working on.

This technique can also be applied to symbols in the schematic editor. When applied to a symbol, the editor changes from the schematic editor to the symbol editor, but maintains the higher level schematic view in the editing window. In this way, you make changes to the symbol in the context of a higher-level schematic.

Traversal of Design Hierarchy: Schematic view
If you have followed a hierarchical design methodology when creating schematics, you'll find that sometimes you'd like read or edit components within an instance without having to open a new window. Additionally, you may just want to quickly double-check that an instance contains the circuit that you expect, and not an old or incorrect one. There are a few shortcuts in Cadence that let you quickly traverse the circuit schematic hierarchy.

1. From the current schematic, select the symbol for the instance that you wish to edit or read by left clicking on it
2. Decide if you want to read from the instance, or edit it.

    a. If you want to read it, select Design => Hierarchy => Descend Read. Alternatively, you can press “x”. Click “OK” or hit enter to confirm.
    b. If you want to read it, select Design => Hierarchy => Descend Edit. Alternatively, you can press “Shift-x”. Click “OK” or hit enter to confirm.
3. This will change the editing tool to the default for the instance into which you’ve descended.
4. If you want to descend further into the hierarchy, repeat steps 1-4 as necessary.
5. When you are done editing, check and save the design. If you are not editing, then this step isn’t necessary.
6. Decide where you wish to return to
    a. If you want to return to one level up, select Design => Hierarchy => Return. Alternatively, you can press “b”.
    b. If you want to return to the top level, select Design => Hierarchy => Return to top. Alternatively, you can press “Shift-b”.
8. Repeat steps 5-7 as necessary.

Note: If you have made changes to a lower-level cell, it is often necessary to save the schematic at the next highest level as well. As you ascend by following step 6a, make sure to check and save every schematic along the way. Failing to do this prior to simulation may result in simulator warnings about using outdated schematics.

Keep in mind that all changes you make to lower level cells accessed this way will be reflected in any and every single instance of that cell in your design. For example, if you descend into a one-bit ALU schematic all the way down to an inverter, and change the schematic for the inverter, you are simultaneously changing all schematics in your design library that use that type of inverter. This would be the same as if you opened the inverter schematic in a new window and edited it that way. This tool is only a shortcut, and does not allow customization of instances. However, it can increase your productivity since you do not need to open and maintain multiple windows for a single design.