ECE1388 – TUT01 – Virtuoso Schematic Editor

This tutorial is helping you regarding the Virtuoso Schematic Editor in Cadence. In this one, you will learn about the following topics:

- Set the user preferences
- Passing parameters in a design
- Multiple-Bit Wire Connections
- Multiple-Bit Wire Naming Conventions
- Patchcord Connections and Patchcord Naming Conventions
- Inherited Connections

Setting options

You can set up options from the Command Interpreter Window (CIW); choose *options – User Preferences.* In this form, you can set different window or command controls. As an example for the command controls, enabling *Infix* would make the editor to use the current location of the pointer as the first data point for any command that you start.

If you want to make permanent changes, you need to save the defaults or modify the .cdsinit file.

Passing Parameters

Passing parameters in your hierarchical design from a symbol view to the schematic would be helpful at the design stage when you need to modify/sweep component values regularly. In order to do this, you need to set up a CDF parameter for the parameter of the component of interest.

As an example, we consider designing an NMOS switch:

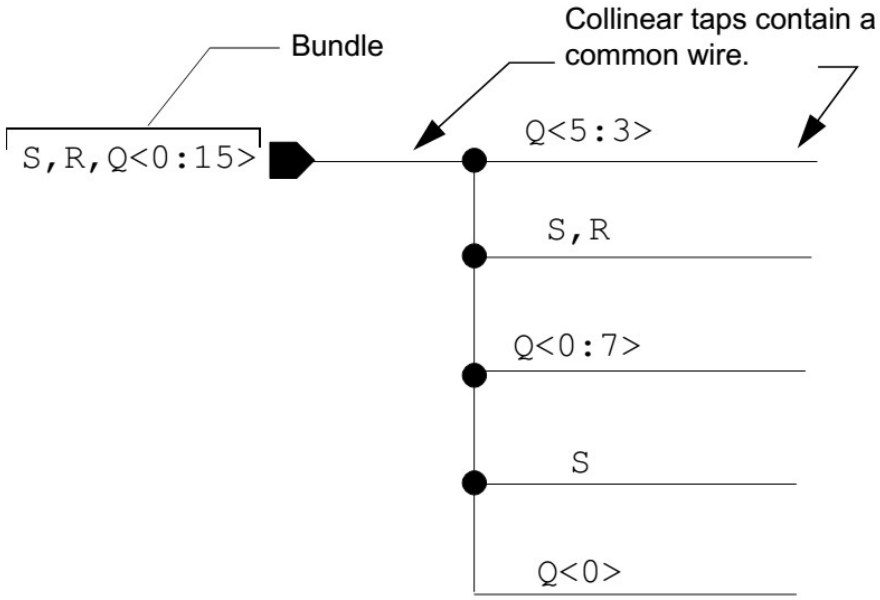
- Make a new cell schematic for your switch
- In the schematic, insert an nmos device. In the Edit Object Properties form, insert `pPar("Wn")` in the width field and `pPar("Ln")` in the length field.
- Create symbol view and save it.
- In case you already have a symbol for the cell, you need to add a CDF parameter by going to *CIW - tools – CDF – Edit*. Then select your cell and make sure the CDF type is BASE, and click *Add* in the *Component Parameters* section to add the component parameter.
    - Param Type → string
    - parseAsNumber → yes
    - units → don't use (you can select an appropriate value as well)
    - parseAsCELL → yes
    - name → put the parameter name
    - Prompt → this value will be displayed as the prompt.
    - defValue → a proper default value
    - Click OK on all forms

Multiple-Bit Wire Connections

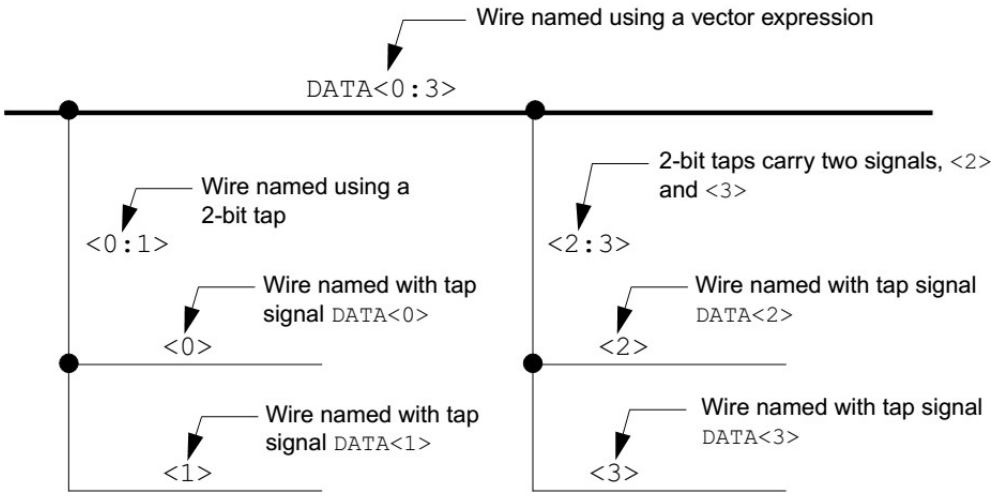
Different ways of connections of multiple-bit wires in your design:

Tapping Multiple bits of a Bundle

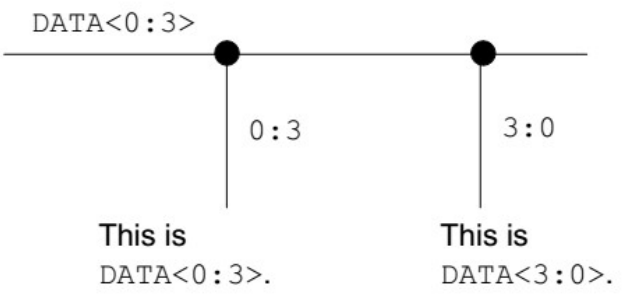
A bundle is a pin or a wire consisting of different names separated by commas; for example S,R. A bus is a single name with a vector expression; for example, Q<0:15>.



Tapping Multiple Bits of a Bus



Designating Tap Size and Bit Order



- ❖ Q1: How can we tap a wire with the name A<0, 0, 1, 1, 0, 1, 0, 1> from a 4-bit bus wire named A<0:3>?

Multiple-Bit Wire Naming Conventions

Using Vector Expressions

- ✓ A list of the individual bit numbers separated by commas and enclosed with angle brackets.
  - Example: DATA<2,1,0> → DATA<2>,DATA<1>,DATA<0>
- ✓ baseName < lowerBound : upperBound : [incrValue] >

  A range of numbers containing a lower and upper bound and, optionally, an increment value.
  - Example: DATA<1:5:2> → DATA<1>,DATA<3>,DATA<5>

Using Prefix Repeat Operators

- ✓ Use the prefix repeat operator <*n> to repeat a single-signal name.
  - Example: <*2>A,B,C → A,A,B,C
- ✓ Use the prefix repeat operator <*n> and parentheses to repeat a group of signal names.
  - Example: <*2>(A,B),C → A,B,A,B,C

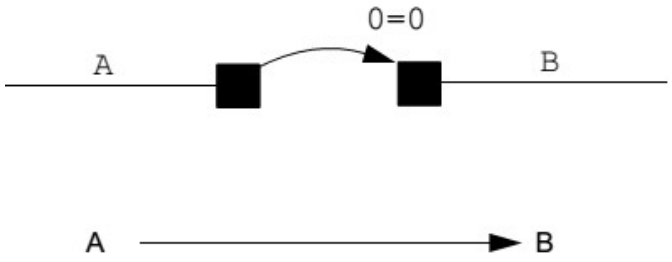
Using Suffix Repeat Operators

- ✓ Use the suffix repeat operator <*n> to repeat each bet in a group of bit names before expanding the vector term.
  - Example: A<0:2*2> → A<0*2,1*2,2*2> → A<0,0,1,1,2,2>
- ✓ Use the suffix repeat operator and parentheses to repeat the sequence of bit names.

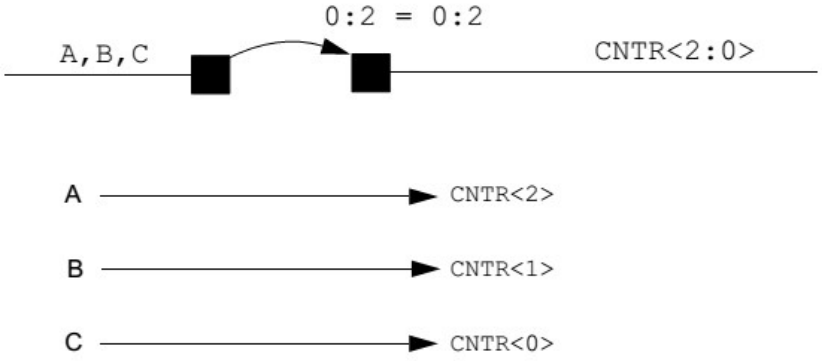
- ❖ Q2: what is the resulting wire name for A<(0:2)*2>?

Patchcord Connections

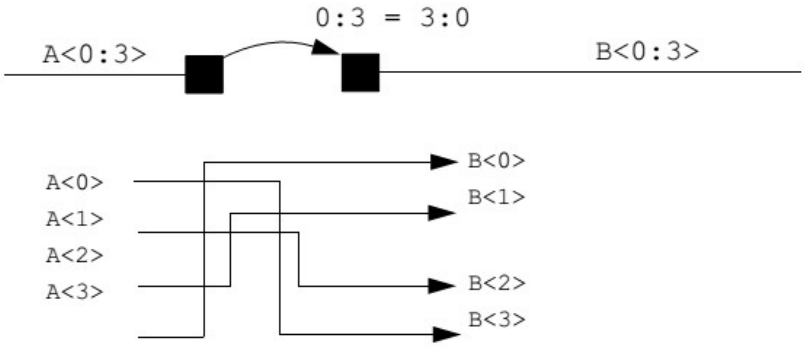
Patchcords are used to alias nets or busses together. This cell can be found in the *basic* library under the name *patch*. As an example, nets A and B can be aliased as shown below.



Or in the following example, singals A, B and C are aliased to CNTR<2>, CNTR<1> and CNTR<0> respectively.



Another example is to rename the bus and flip the order as in the following figure.



- Q3: Consider you have a 16-bit bus in your design (SYSBUS<15:0>). How can you make two new 8-bit buses with different names consisting of the lower and upper 8-bits (DATA<7:0> and ADDR<7:0>?

Inherited Connections:

Global signals are useful in designs where you have only one power domain let's say *vdd!* for power and *gnd!* for ground. Multiple power and ground domains in a design increases the chance of conflicting global signal names.

Inherited connections are specifically useful in designs that multiple power/ground domains are needed, even in the same voltage range. The possibility of overriding the values at different hierarchy levels is crucially important for such designs.

Inherited connections consist of two parts:

- Net expression defining a property name and a default global net name
- One or more *netSet* properties that provide a new signal name that overrides the default global net name.

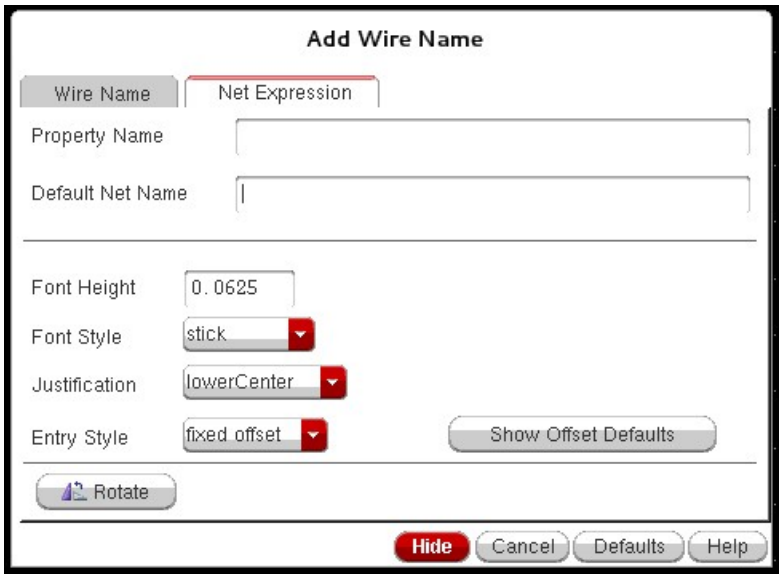
The definition of an inherited connection is done by a net expression with the following format:

[@myPropertyName:%:defaultGlobalSignalName]

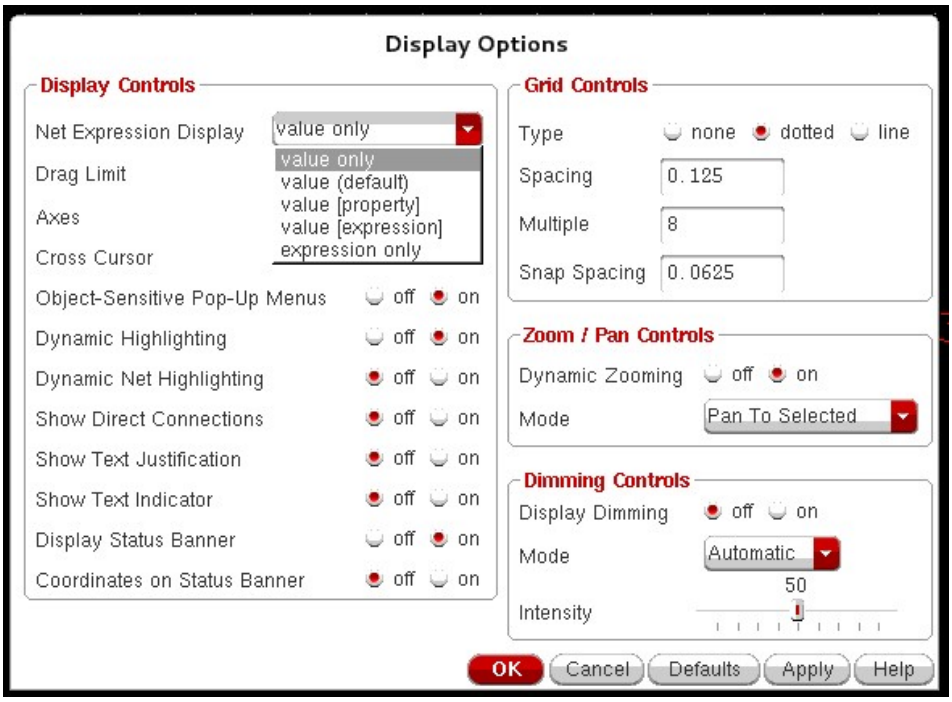
For example: [@myPower:%:vdd!]

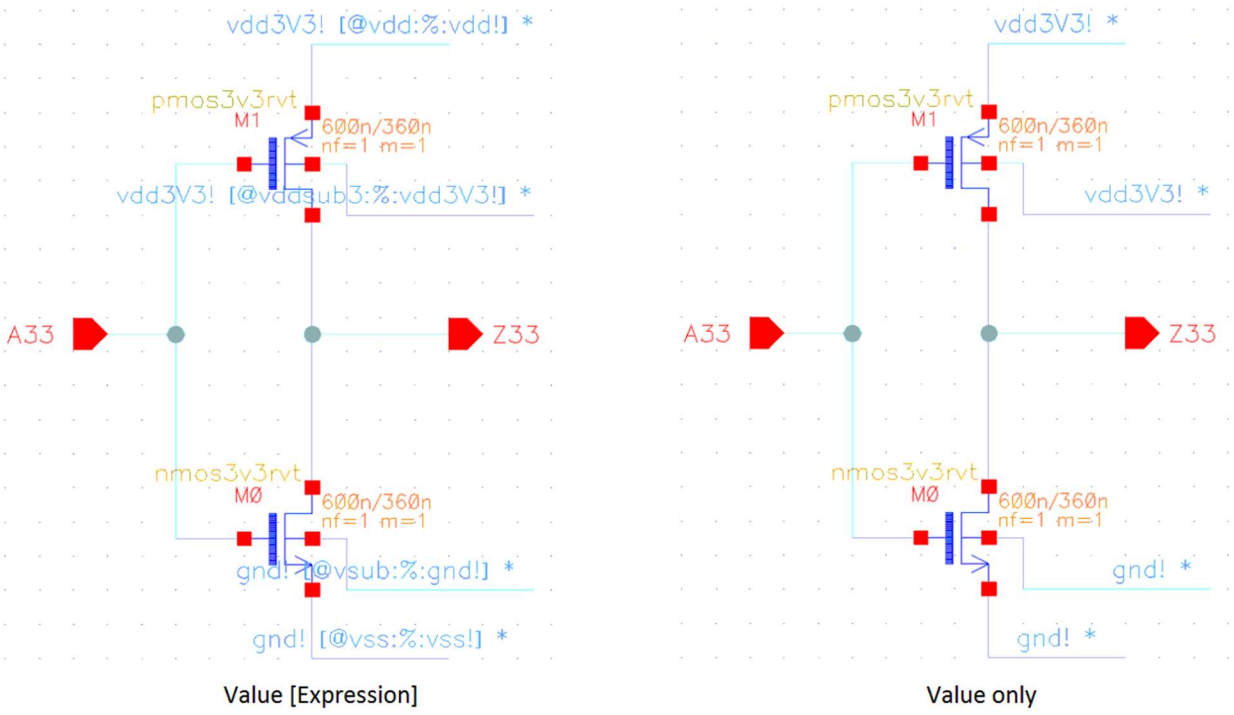
You need to define a *netSet* to property at a higher hierarchy level to override the default value with the *netSet* value. The new value does not need to be a global signal and the new value assignment doesn't work if in a lower hierarchy level, you have assigned another value to that connection.

You can define the net expressions the same way as to assigning a label to a net (hotkey: l). You only need to choose the *Net Expression* tab in the window as shown in this snapshot.

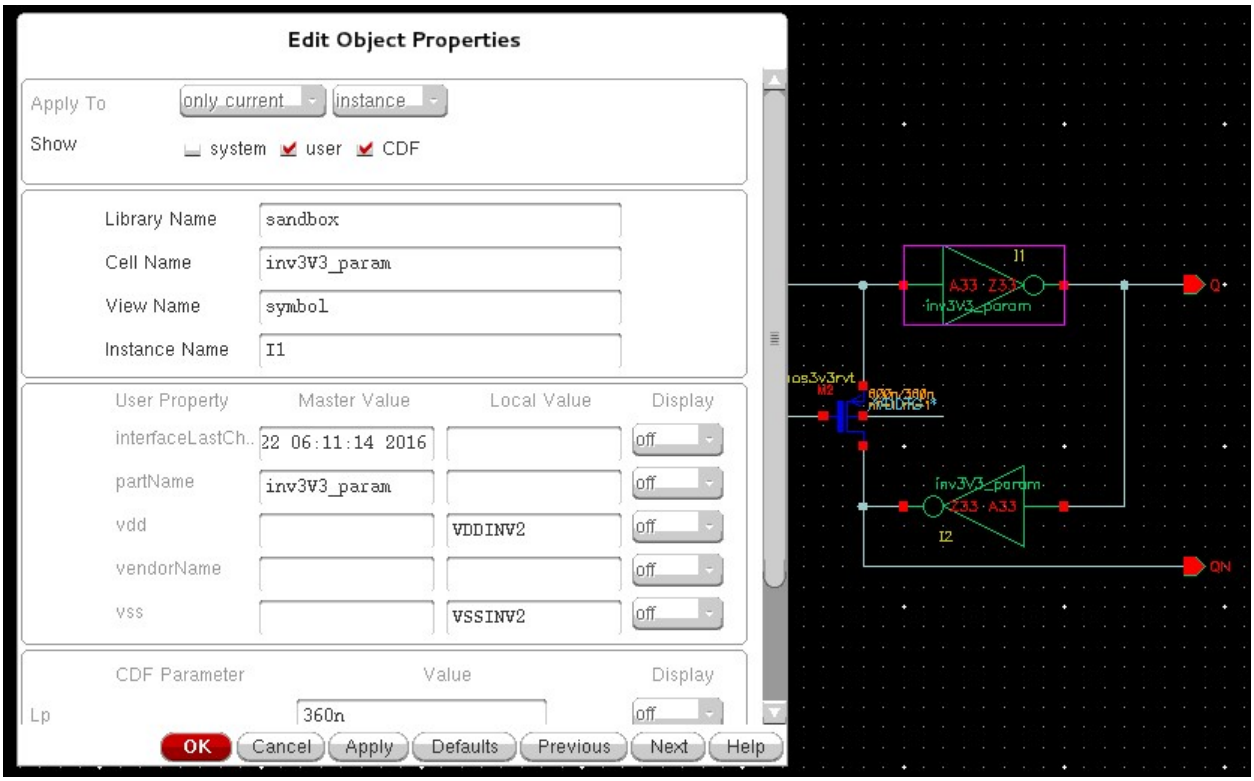


In the display options (hotkey: o) you can determine how to determine how to show the net expressions. Usually the default is "*value only*" where only the assigned value will be shown, by changing to "*value [expression]*" you can see the full net expression. Please see the following figures for comparison.

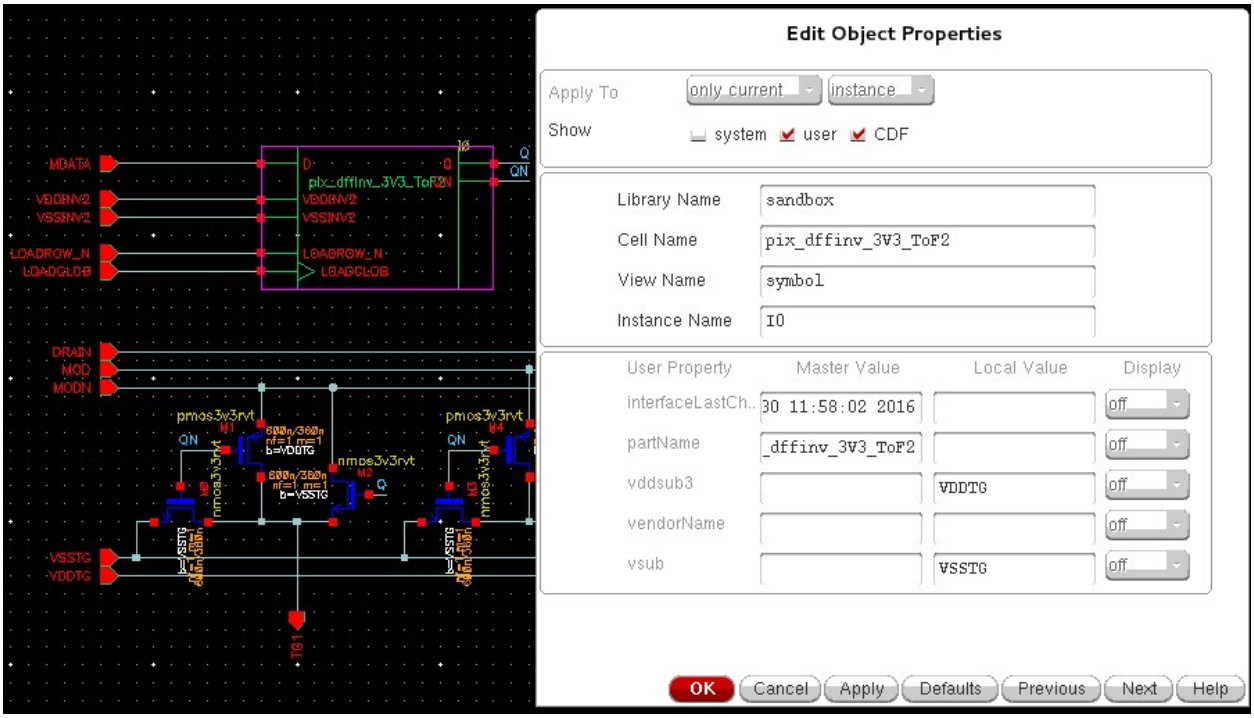




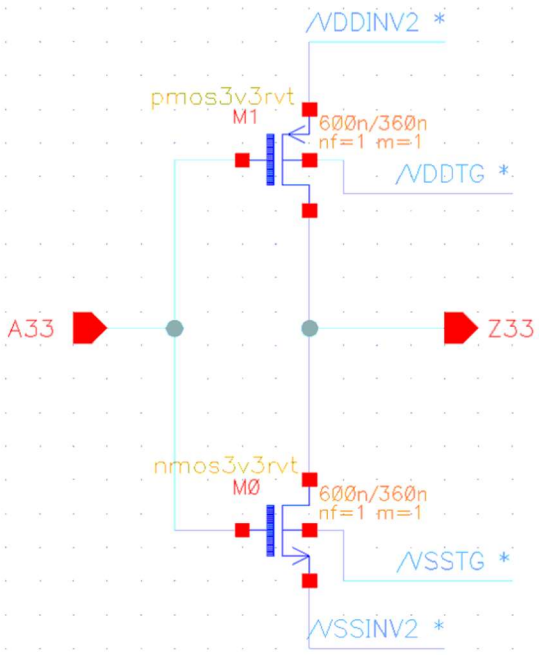
And at one higher hierarchy level we can define the power/ground rails by adding a *netSet* to the properties of the cell as shown in the picture below:



And at a different hierarchy we define the substrate connections as shown below:

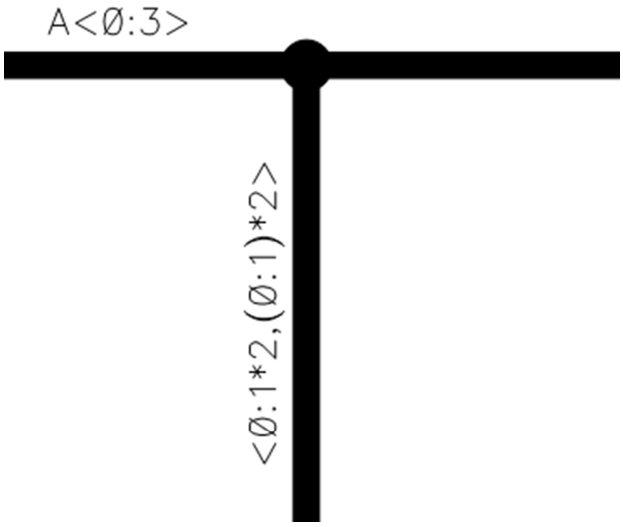


Now in this design if we descend into the inverter schematics, we see the net values as shown here:

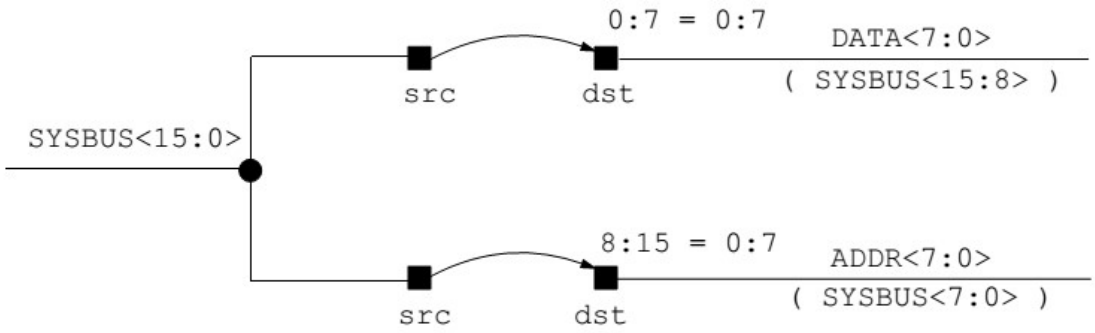

/VDDINV2 *
pmos3v3rvt
M1
600n/360n
nf=1 m=1
/VDDTG *
A33
Z33
nmos3v3rvt
M0
600n/360n
nf=1 m=1
/VSSTG *
/VSSINV2 *

Answer to the Qs

- Q1:



- Q2: A<0,1,2,0,1,2>
- Q3



Reference

Virtuoso Schematic Composer User Guide, by Cadence, Product Version 5.0, Oct 2002.