# CADENCE TUTORIAL

By

Chao You
Mariam Hoseini

ECE 423/623
North Dakota State University
Spring 2007

## Content

## 1. Schematic (Inverter)

### Introduction to Cadence

Cadence Design Systems provides tools for different design styles. In this tutorial you will learn to use three Cadence products: Composer Schematic, Composer Symbol and the Virtuoso Layout Editor. This tutorial will help you to get started with Cadence and successfully create symbol, schematic and layout views of an inverter and even more!

### Getting started

Verify your Cadence setup

- 1. Reboot your computer. Select Linux (Ubuntu).
- 2. Use your Windows' user name and password to login.
- 3. Go to Applications >> Terminal.
- 4. Type "icfb" to start Cadence.

If you have "Command not found" error message, please follow the following steps. In the terminal window, type the following command:

- cp /cad/cds/script/bash.bashrc .bashrc
- cp /cad/cds/script/cds.lib cds.lib
- cp /cad/cds/script/.cdsenv .cdsenv
- cp /cad/cds/script/.cdsinit .cdsinit
- cp /cad/cds/script/bindkeyAMS.il bindkeyAMS.il
- cp /cad/cds/script/display.drf display.drf

Then **restart** your terminal window.

When icfb starts, a new window opens up on the screen, as in figure 1. This new window is called the Command Interpreter Window or CIW.

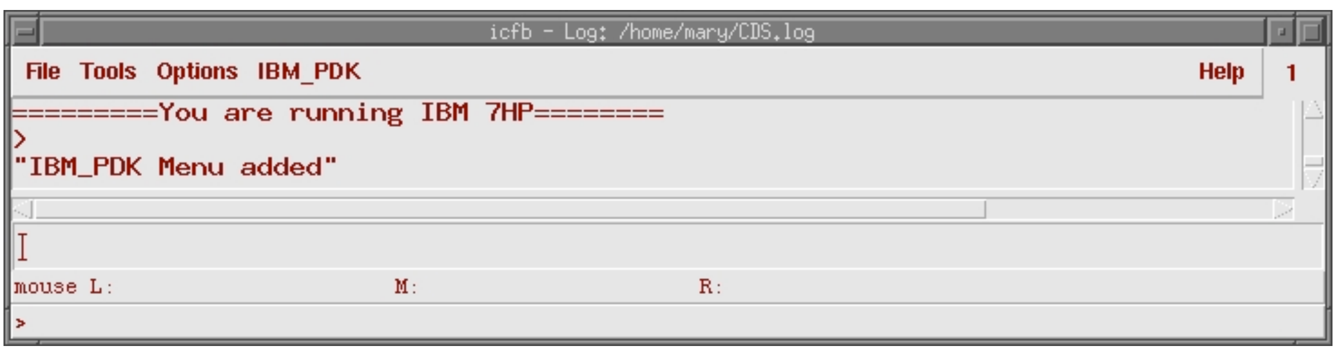


Figure 1- Command Interpreter Window

Now we will learn how to create the schematics, symbol and layout views of a design and how to simulate the design. We will follow our steps for creating the schematic, symbol and layout of an inverter and simulating it.

**Creating a New Library**

Take a look at existing libraries T**ools-> Library manager**. In order to separate your files from those that already exist in the system, you must create a library of your own and place your files in that library. To create a new library from the CIW, do the following: **IBM_PDK-> Library-> Create**, the "New library" window opens.

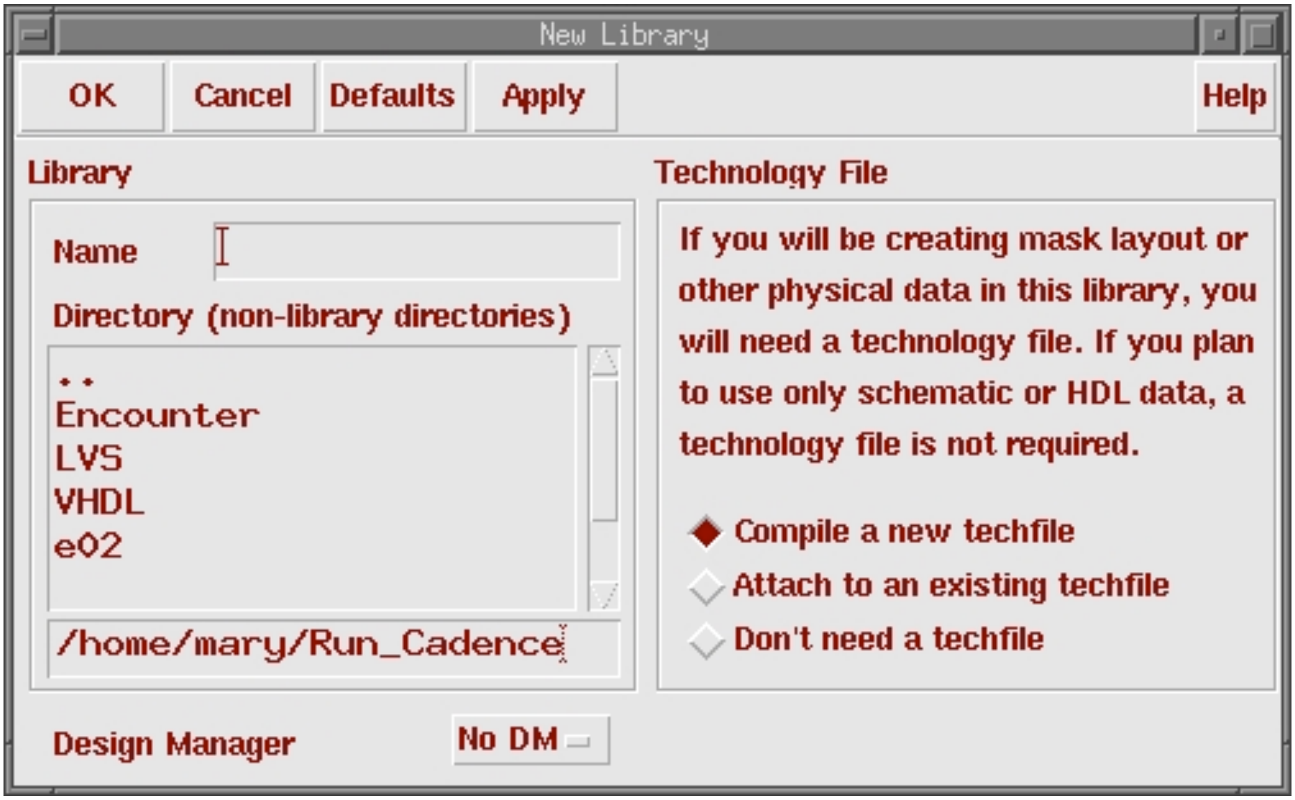

Figure 2- New Library Window

Type the name of the library, we name our library as VLSI, in "Technology file" select "Attach to an existing techfile", then OK. Another window opens, in "Technology library" chose "bicmos7hp" then OK. In new window make sure that the "library name" is "VLSI", "Technology" is "bicmos7hp" and "Number of levels of metal" is "M5" (that is 5 layer of metal), click OK.

**Opening a New Cell View**

To open a new cell view from the CIW, do the following:
**Tools-> Library manager,** then **File-> New -> Cell View.** Library name should be defined to "VLSI"; Type the Cell name "Inverter", View name would be "Schematic" and Tool as "Composer-Schematic", then OK.
The "Virtuoso® Schematic Editing" window opens as can be seen in figure 3.

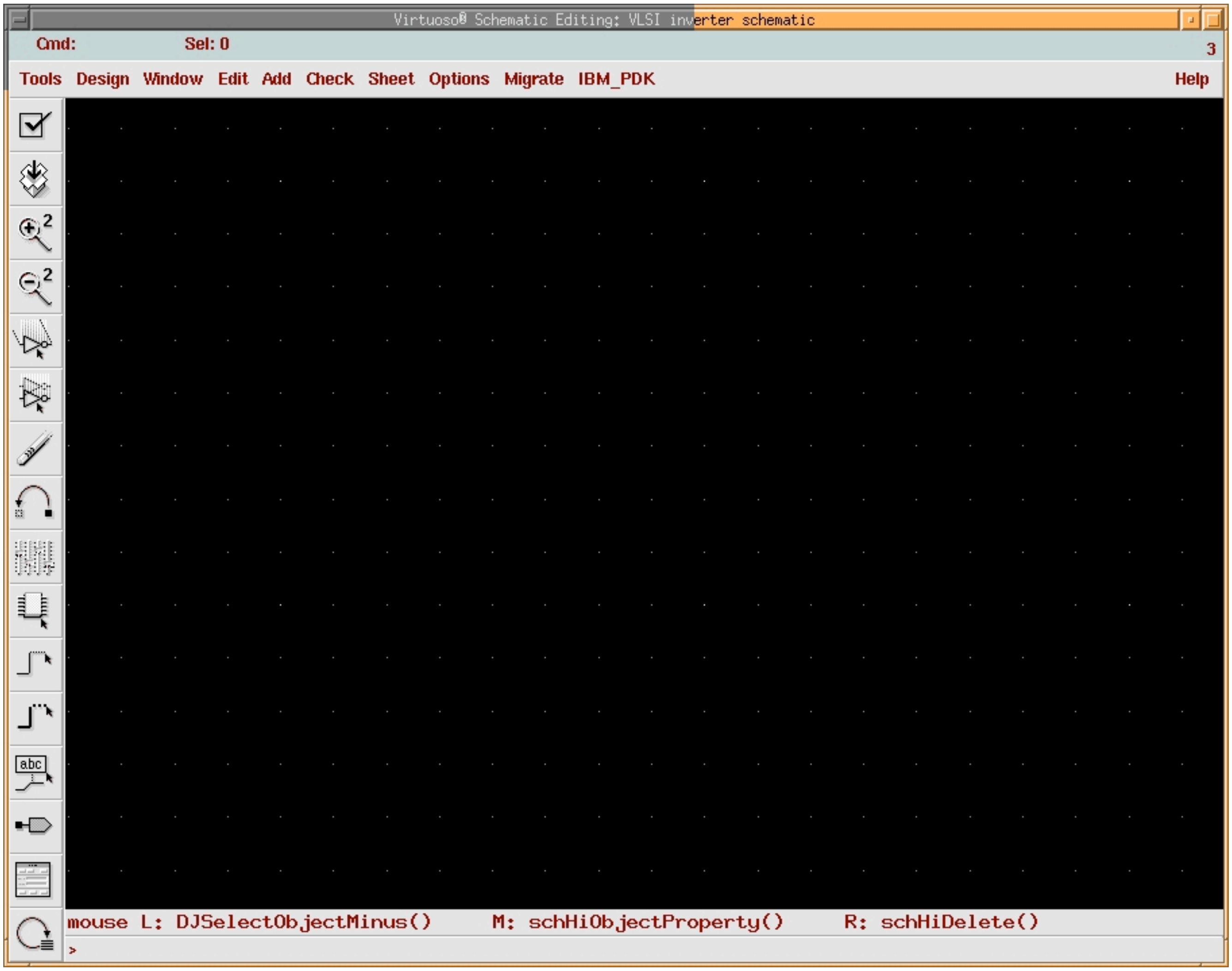

Figure3- Virtuoso® Schematic Editing" window

**Creating a Schematic**

A schematic is the graphical representation of the logic circuit design either by discrete devices (transistors, resistors, etc.) or by logic elements (AND gates, OR gates, etc.). The connectivity information is obtained from the placement of pins and wires. To create a schematic design, you use a "Schematic Editing" window.

Now we are going to make the schematics of the inverter. In the left hand side of the window there is panel, in which some icons can be seen, one of them looks like an IC; that is "Instance", or you can go through **Add-> Instance.** Click on "Browse". The "Library Browser-Add Instance" window opens up. In this window, select **Library=bicmos7hp**, **Category = FET** (if Show Categories is on), **Cell = NFET**, **View = symbol.** Then close.

Move the cursor onto the Virtuoso® Schematic Editing window. You will see the symbol attached to the arrow pointer. If necessary you can modify the properties of NFET in "Add Instance" window. Left click and Place the NFET.

You can browse again and this time chose a PFET, in order to modify the threshold voltage of inverter that is to set $V_m$ to "0.5 * $V_{DD}$", the width of PFET should be greater than NFET, so increase the value

for PFET width 3 times, also select "Add NW contact", then place PFET.
Browse again, chose "subc" and places it too.
Now the window looks like figure 4.

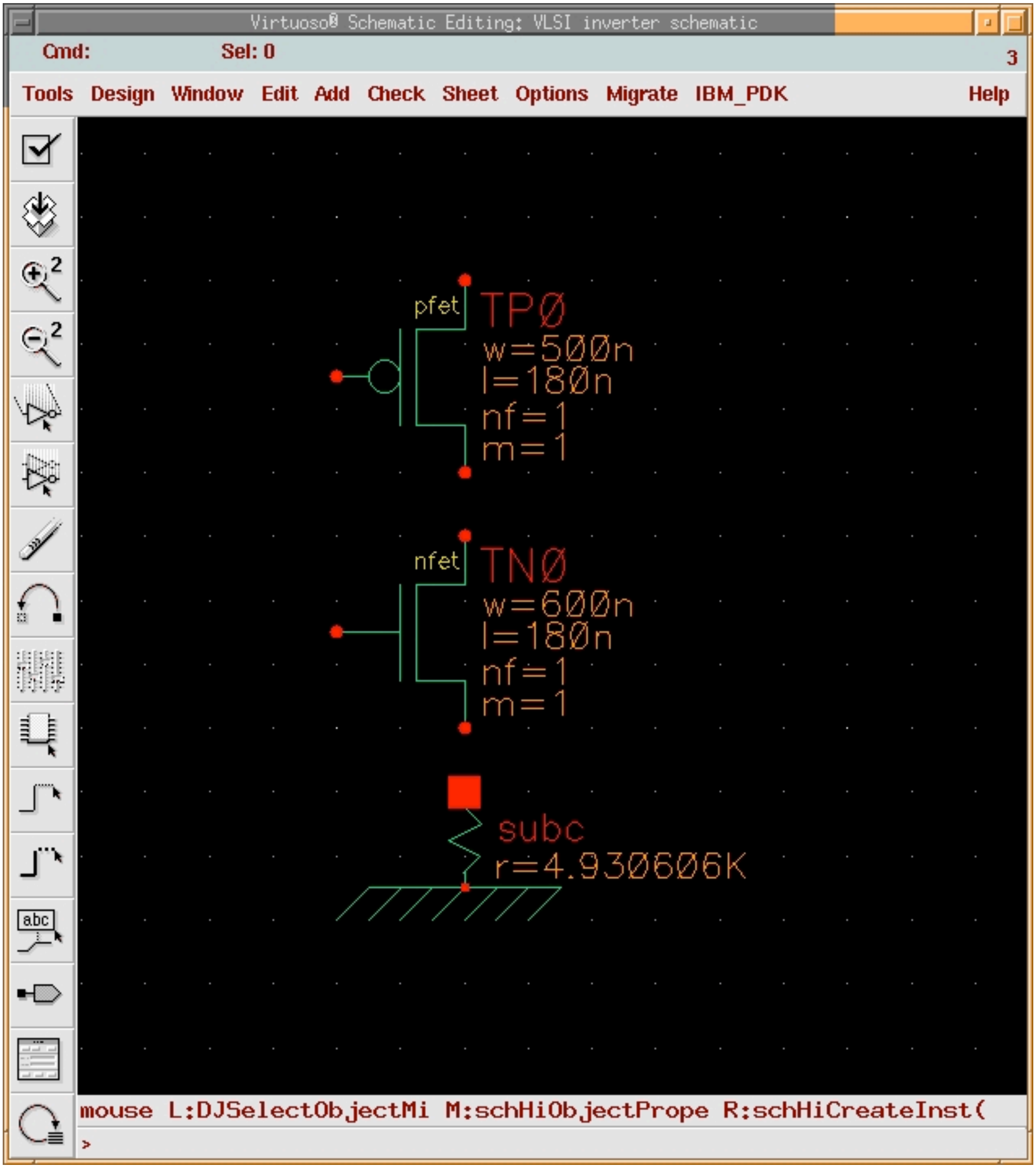


Figure 4

Now from left panel chose **Wire (narrow)** or **Add-> Wire (narrow)** and complete the inverter.

**Note1**. In order to enlarge something select that with right button of the mouse.
**Note2.** The shortcut key for moving is "M" and for stretching is "S".
**Note3**. To stretch, first press "S" then select the thing you want to move with left button of mouse, left click on it, at last move it to wherever you want.
**Note4**. In order to draw wire; single left click on the start point, then single left click on the destination.

Now the inverter looks like figure 5.

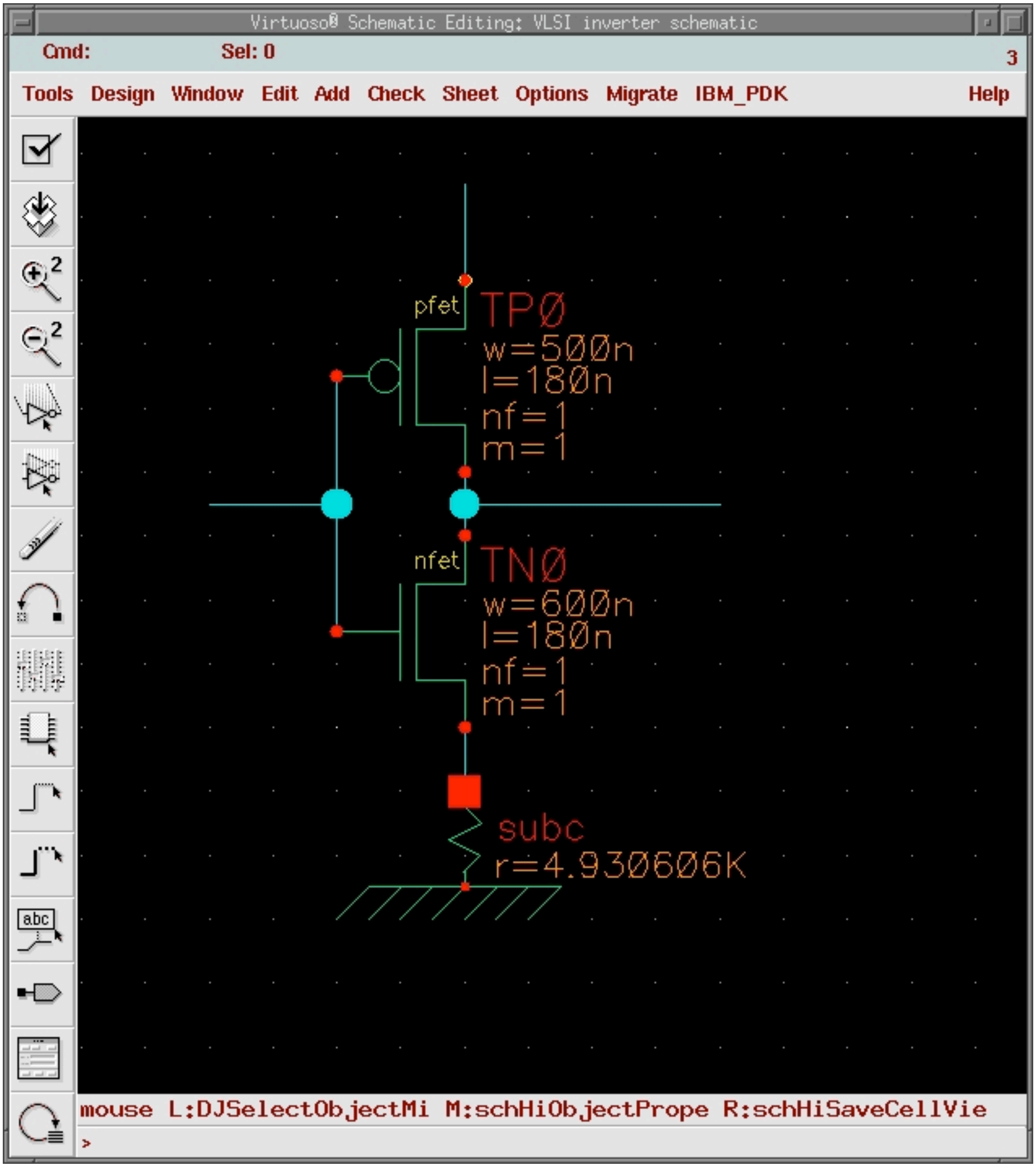


Figure 5

From left panel chose **Wire Name**, in "Names" type "**vdd!"**, "**vss!"**. The exclamation mark means that vdd and vss are global nets. Place "vdd!" and "vss!" on the corresponding wires.

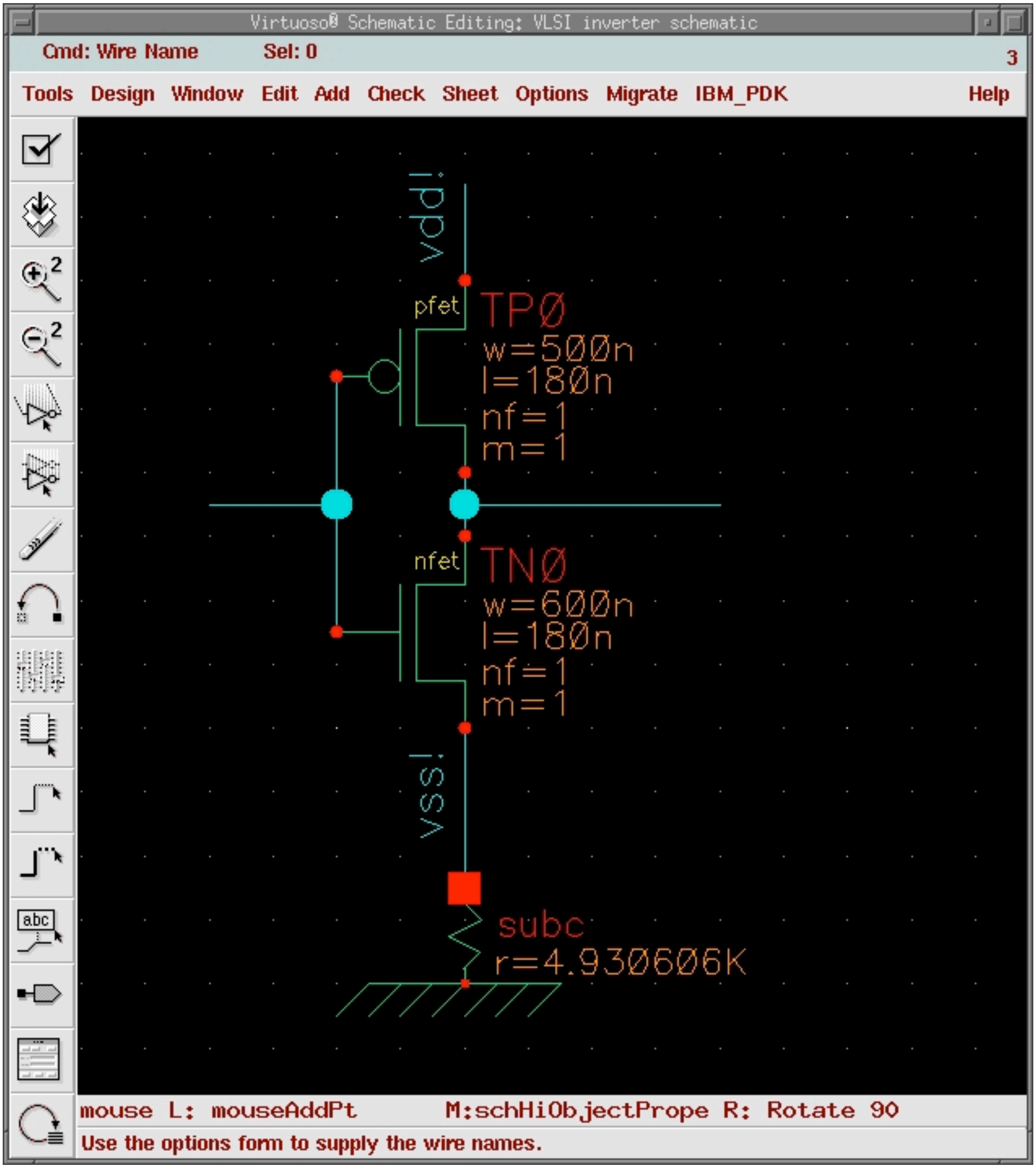


Figure 6

From left panel chose **Pin,** in "Pin Names" type **A** and select the direction as **input.** Place the pin at input. Repeat this step again, in "Pin Names" type **Z** and select the direction as **output.** Place the pin at

output.
Now the schematic of the inverter is complete as can be seen in figure 7.

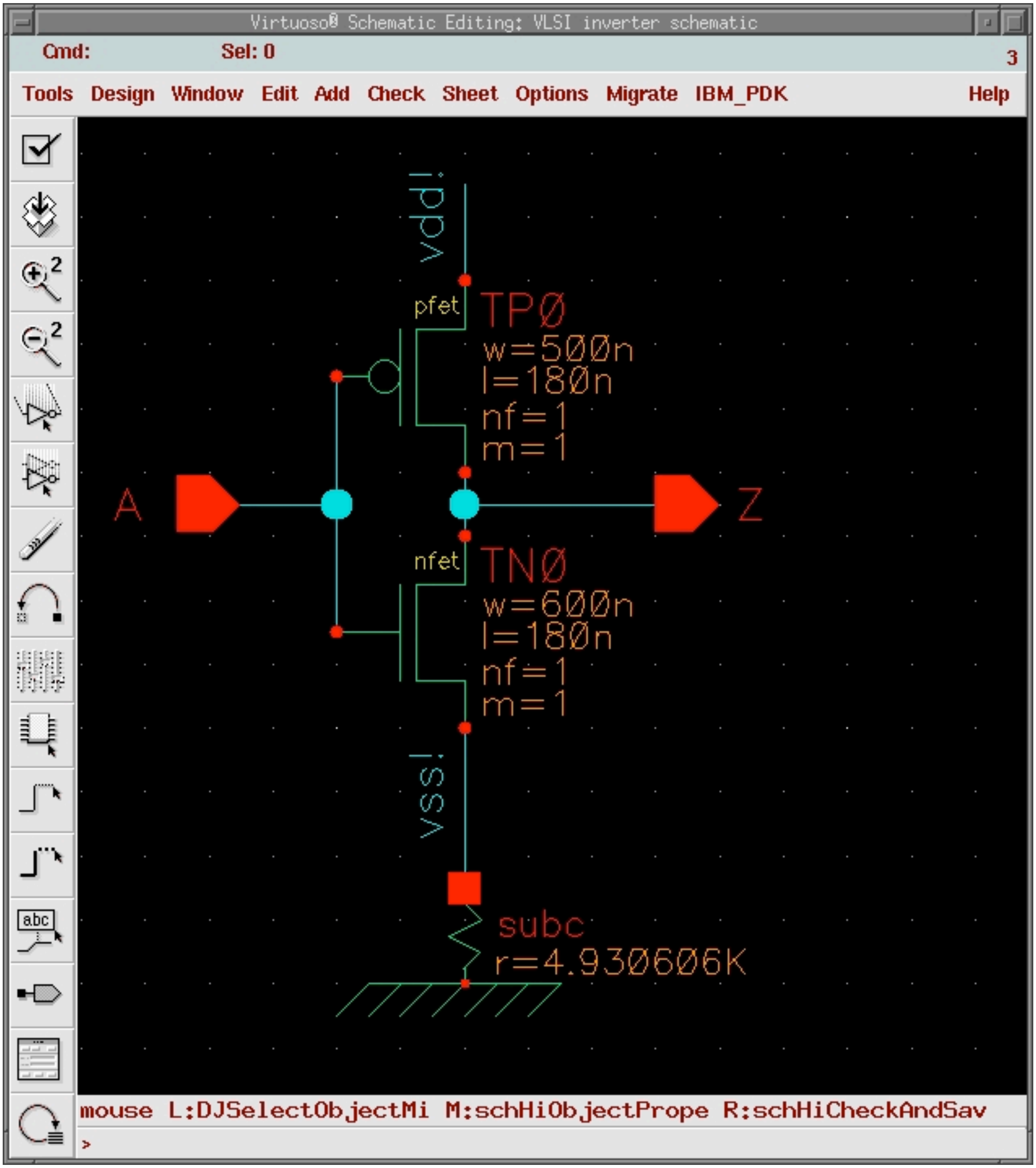

Figure 7- Complete schematic of inverter

From left panel, chose "**Check and Save**". If there were any error in the schematic, it would warn you and you should modify it; otherwise the schematic is saved.

## 2. Symbol (Inverter)

### Creating a Symbol

Now we want to make a symbol for the inverter. In the same “Virtuoso® Schematic Editing” window, chose **Design-> Create Cellview-> From Cellview.** The “library name” should be **VLSI;** “Cell name” should be **inverter** and “From View Name” as **Schematic,** then OK.
The “symbol Generation Options” opens. Make sure you have all input pins in “Left Pins” and all output pins in “Right Pins”. For our inverter we should assign **A** in “Left Pins” and **Z** in “Right Pins”, then OK.
“Virtuoso® Symbol Editing” window opens.

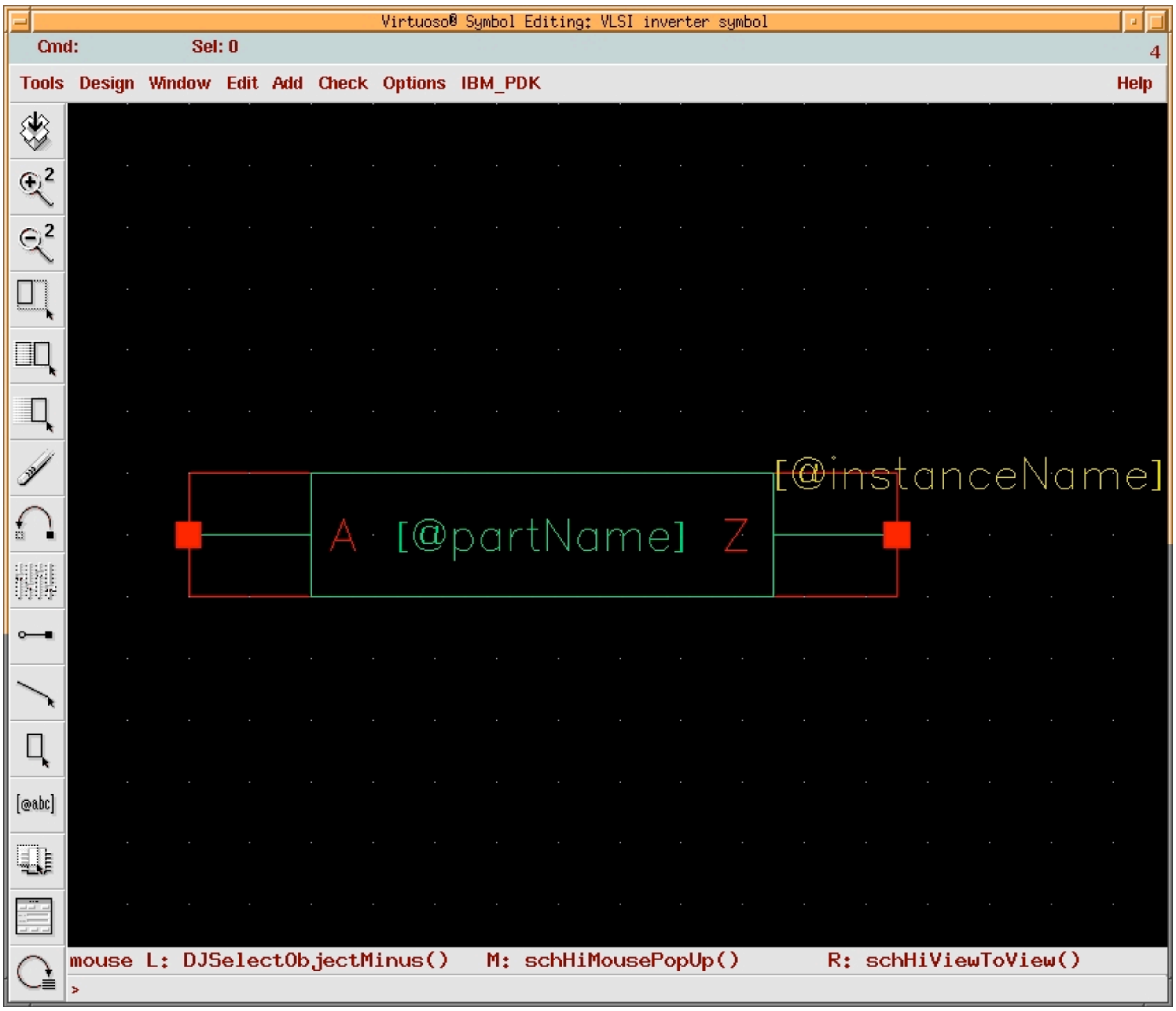

Figure 8-“Virtuoso® Symbol Editing” window

In this Window, your favorite symbol for inverter can be created. So you can delete extra parts and just leave A & Z pins.

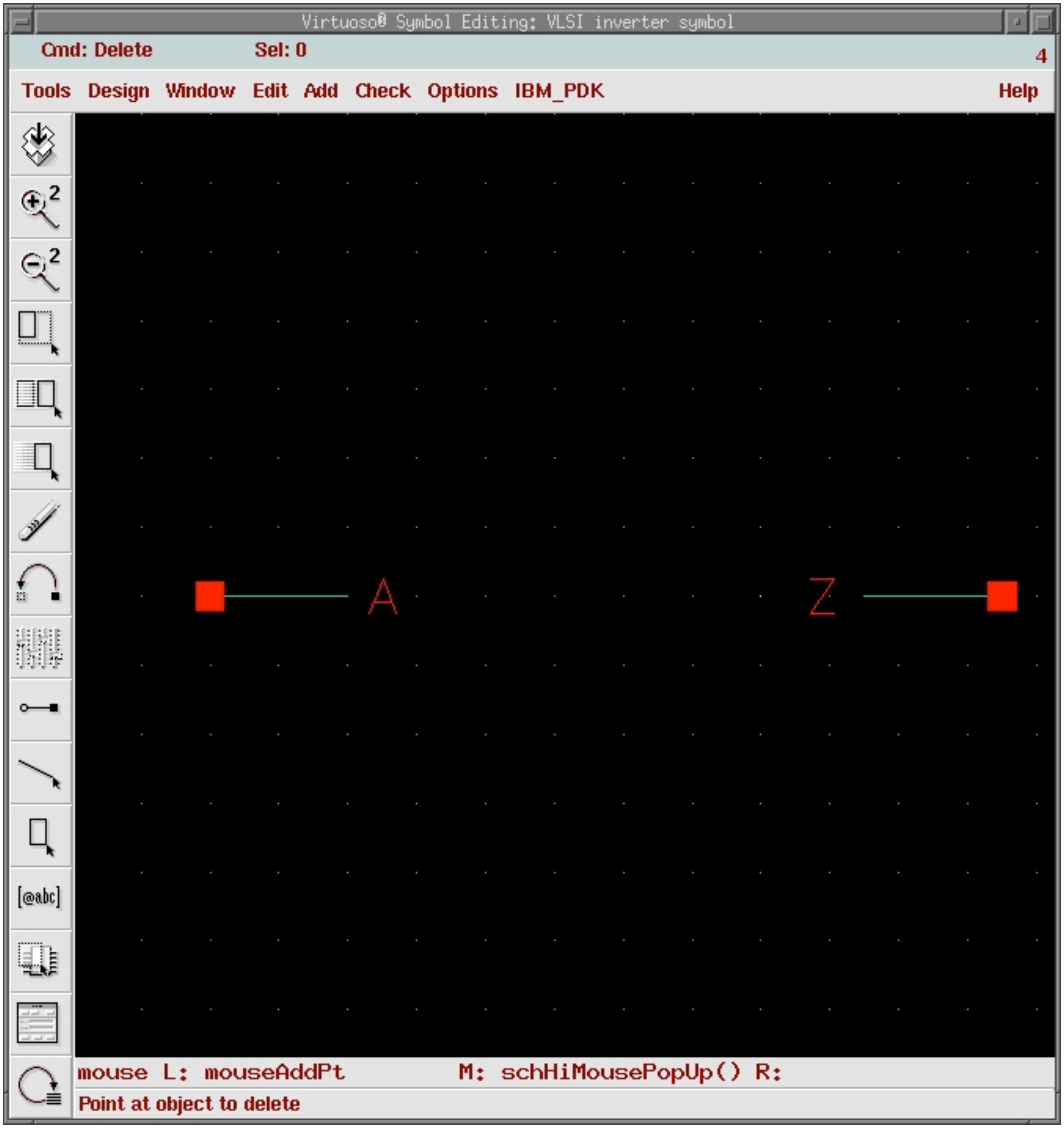


Figure 9

By means of **Line** & **circle**, from **Add-> Shape,** you can complete the symbol of inverter.

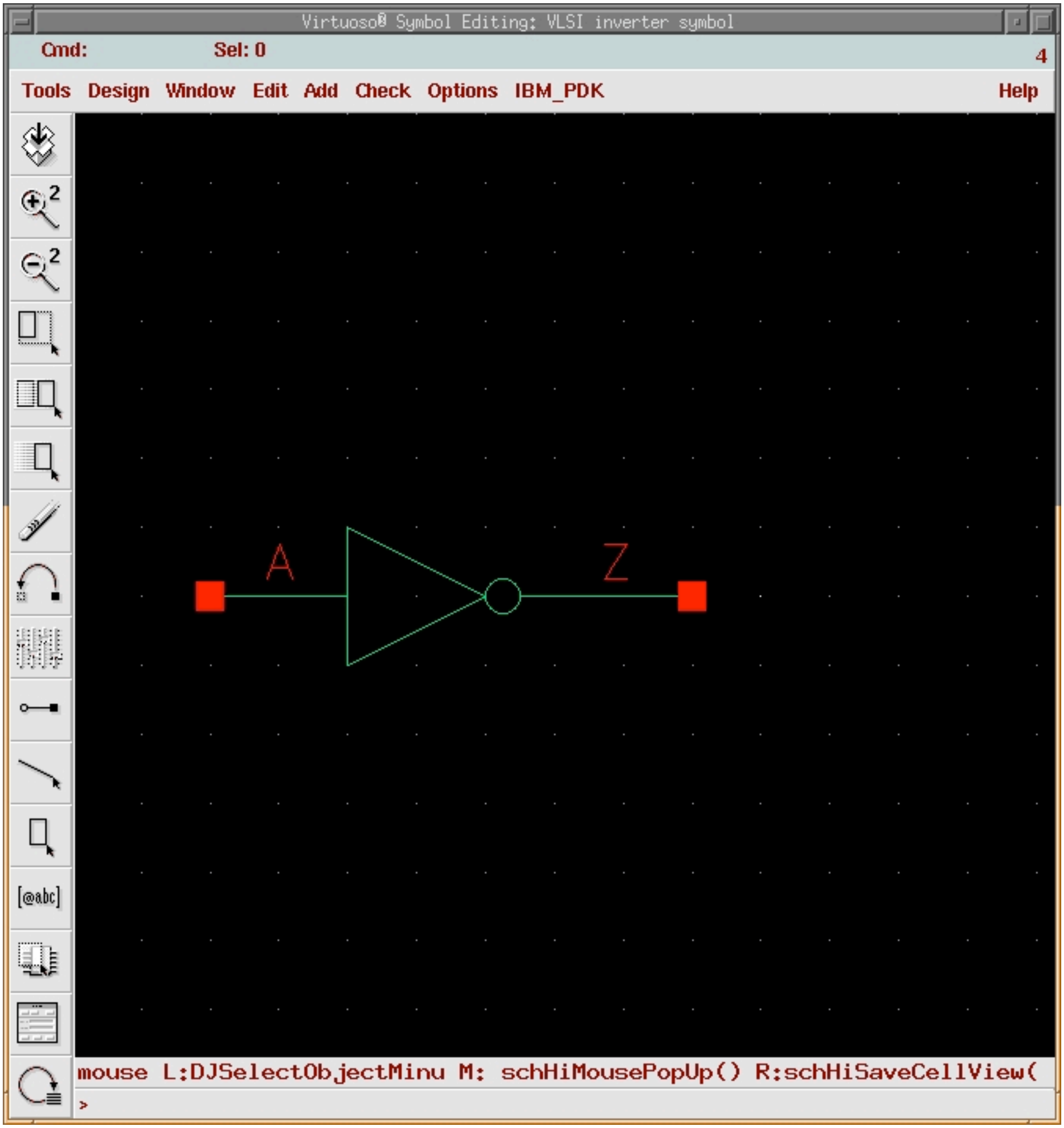


Figure 10- Symbol for inverter

### 3. Simulation (Inverter)

Now we want to test the schematic of inverter and simulate its operation. In order to do this, a new cellview should be created, so do: **Tools-> Library Manager-> File -> New-> Cellview**. Name the cell as "test_inverter", then OK.
Do **Add Instance,** select **Library=VLSI**, **Cell = inverter**, **View = symbol.** Then close.
Add instance again and select **Library=Share, Cell = CMOS_500M**, **View = symbol.** Then close. Repeat this and select **Library=Share, Cell = Power_3V**, **View = symbol.** Then close.

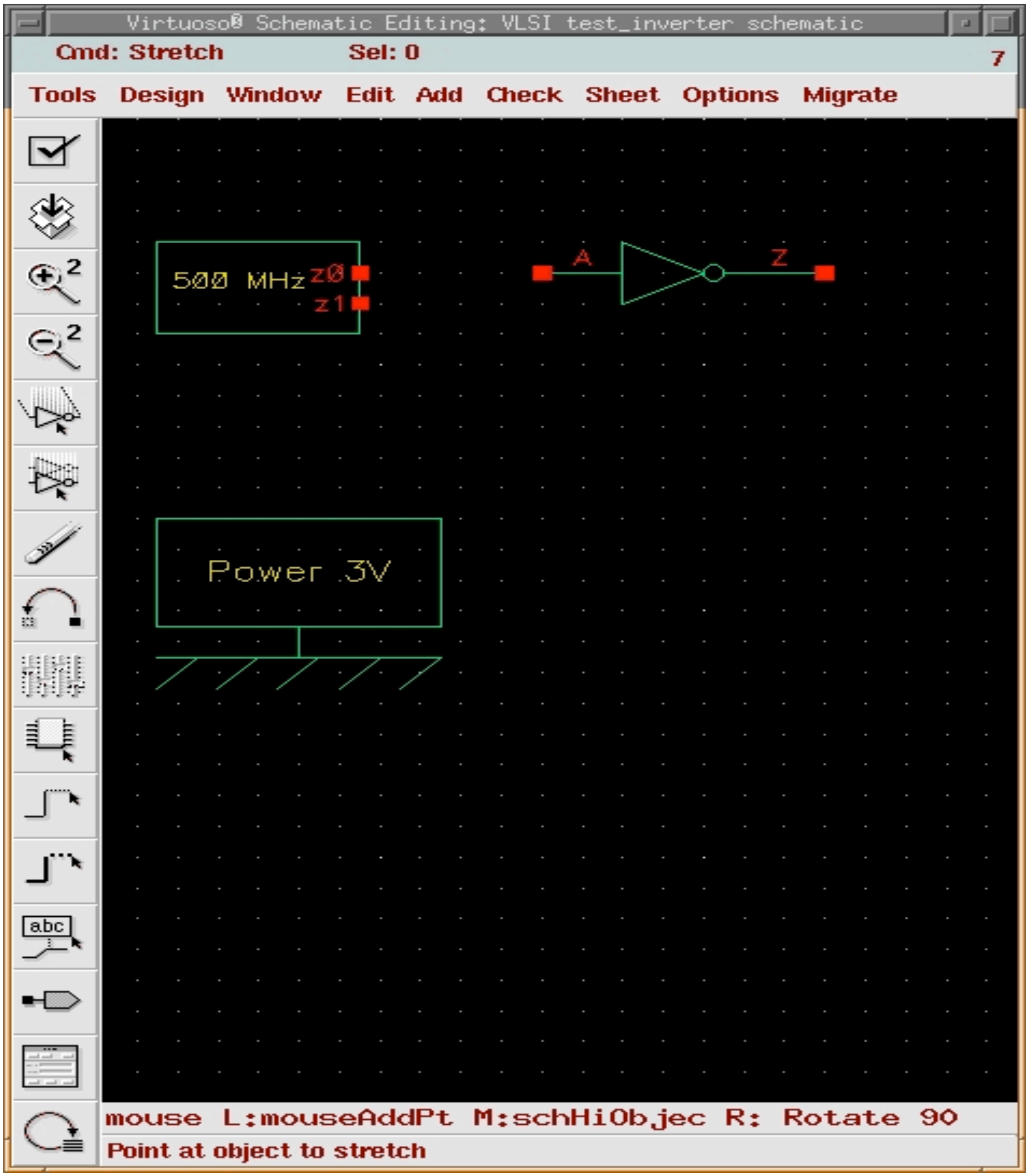


Figure 11

Connect the input of inverter to one of the output of ac source. Name the input and output wire of inverter as “in” &”out”. As we assign vdd and vss by global nets, you don't need to connect power source to the inverter with wires.

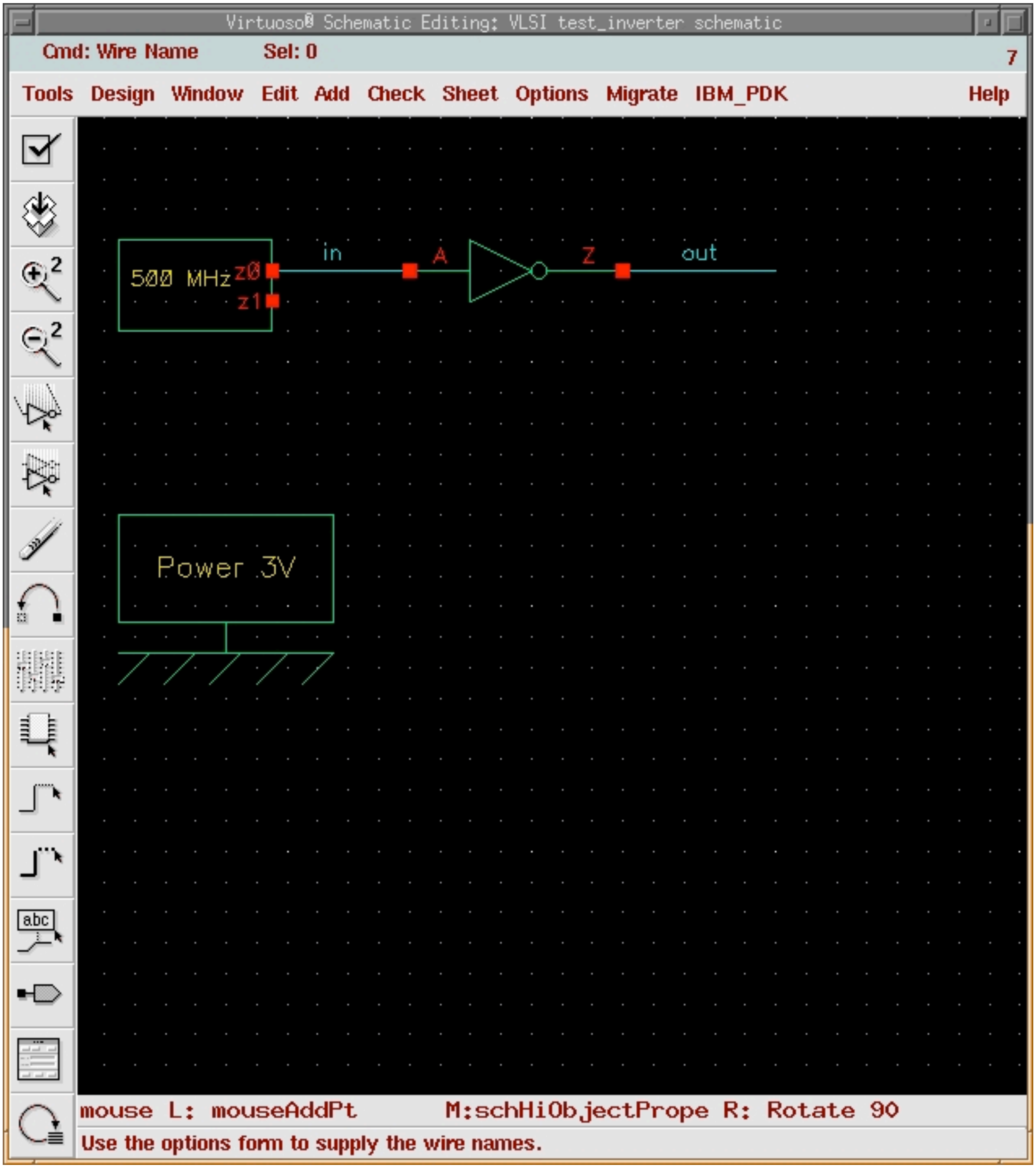


Figure 12- Cell view for testing inverter

Then “Check and Save” the schematic. If you get three warnings considering the unconnected ports Z1

and Z, you can neglect them and continue.
It is time to simulate the design. Select **Tools-> Analog environment.** Virtuoso® Analog Design Environment window opens. In right hand side panel select "Netlist and Run". Other window pop ups, click OK. If in CIW you got the message "Netlister: There were errors, no netlist was produced." you had to go back to the cell view and modify the errors; otherwise it is successful!
In Virtuoso® Analog Design Environment window, chose **Results-> Direct Plot-> Transient Signal**.
In Virtuoso® Schematic Editing window, select input and output wire by single left click, if they are selected properly, they become colorful ,then press **Esc.** The waveform window appears in which input and outputs are plotted, you can see whether the inverter operates well or it needs modification.

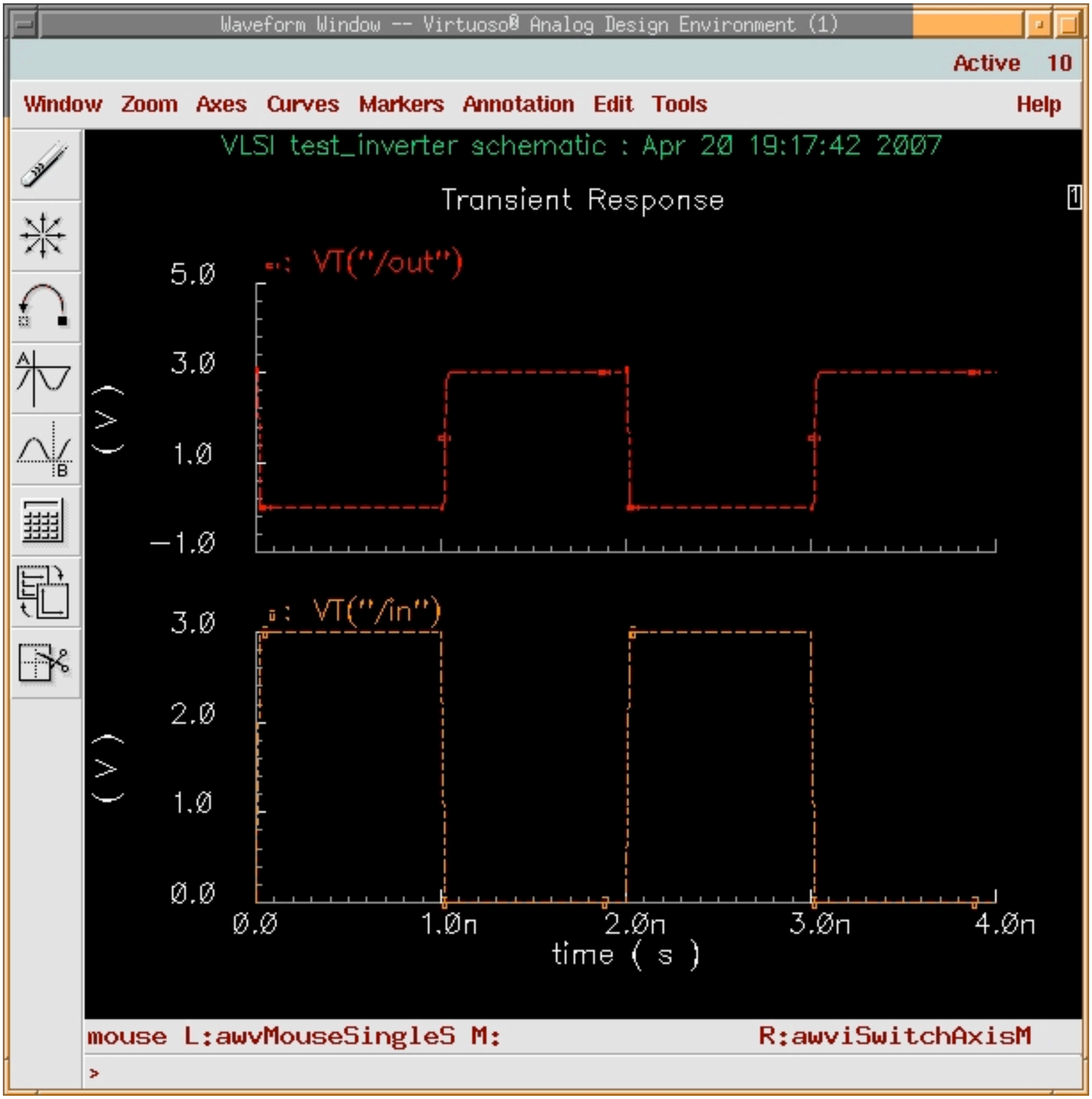


Figure 13- Simulation result of inverter

## 4. Schematic, Symbol and simulation (AND Gate)

At this step we want to create AND gate by means of Inverter and NAND gate. We have already created an inverter and simulated it successfully, so we should only create NAND gate.
Follow all the steps explained in the schematic section for inverter and make the NAND schematic as can be seen in figure 14. Make sure to adjust the width of PFET and NFET.

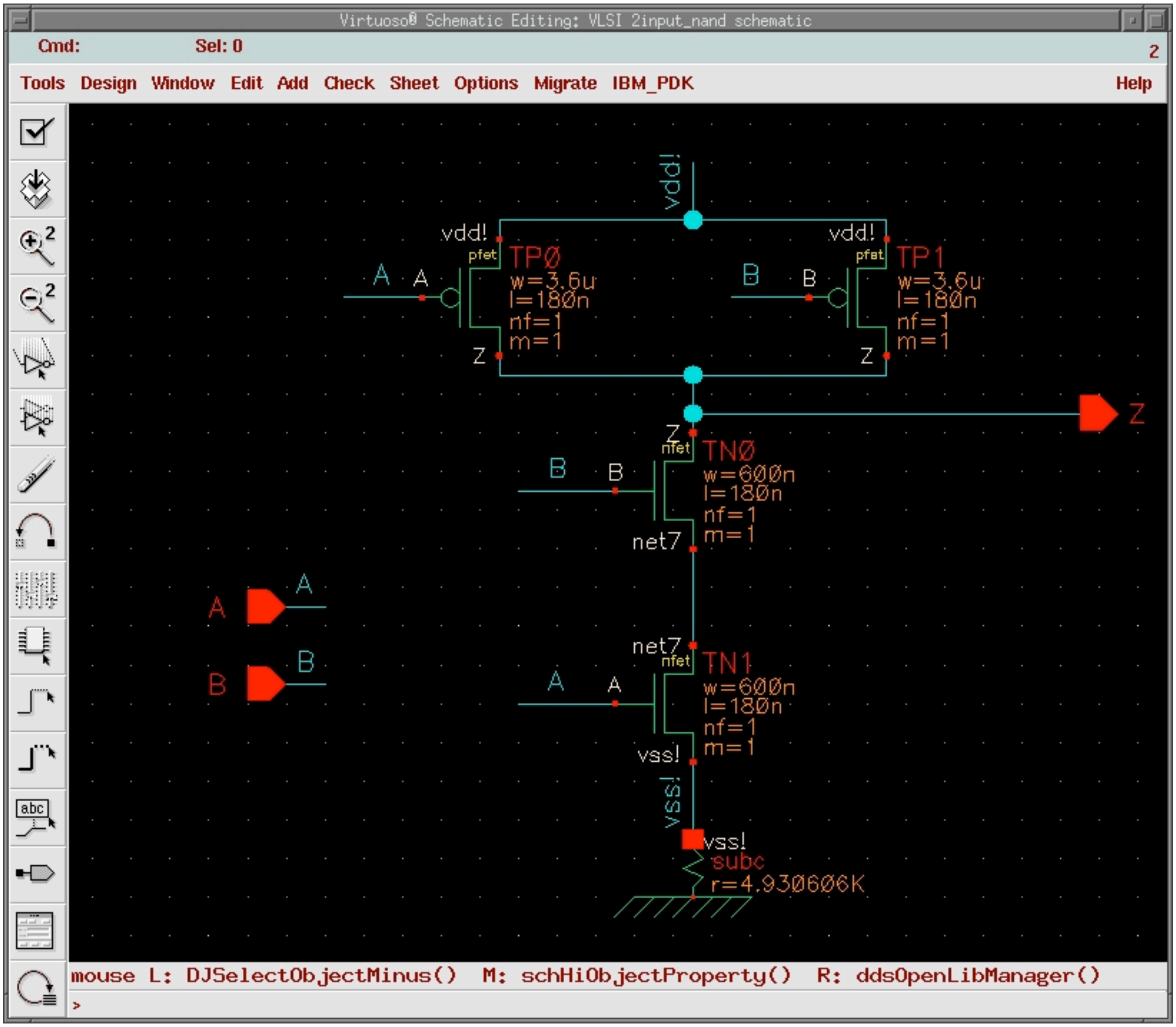


Figure 14- Schematic of NAND gate

Now make the Symbol for NAND gate.

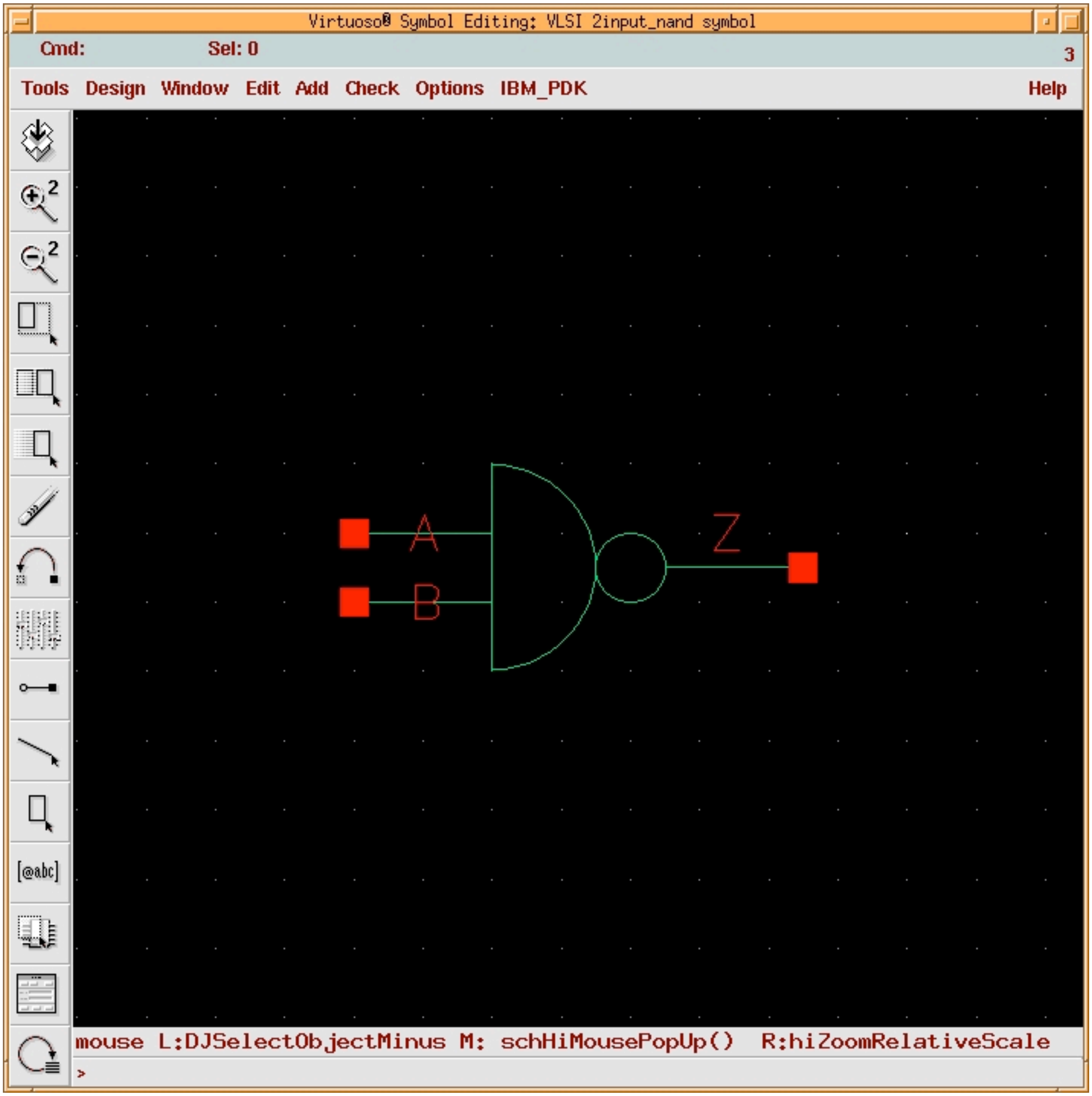


Figure 15- Symbol of NAND gate

Make another cellview to test the NAND gate.

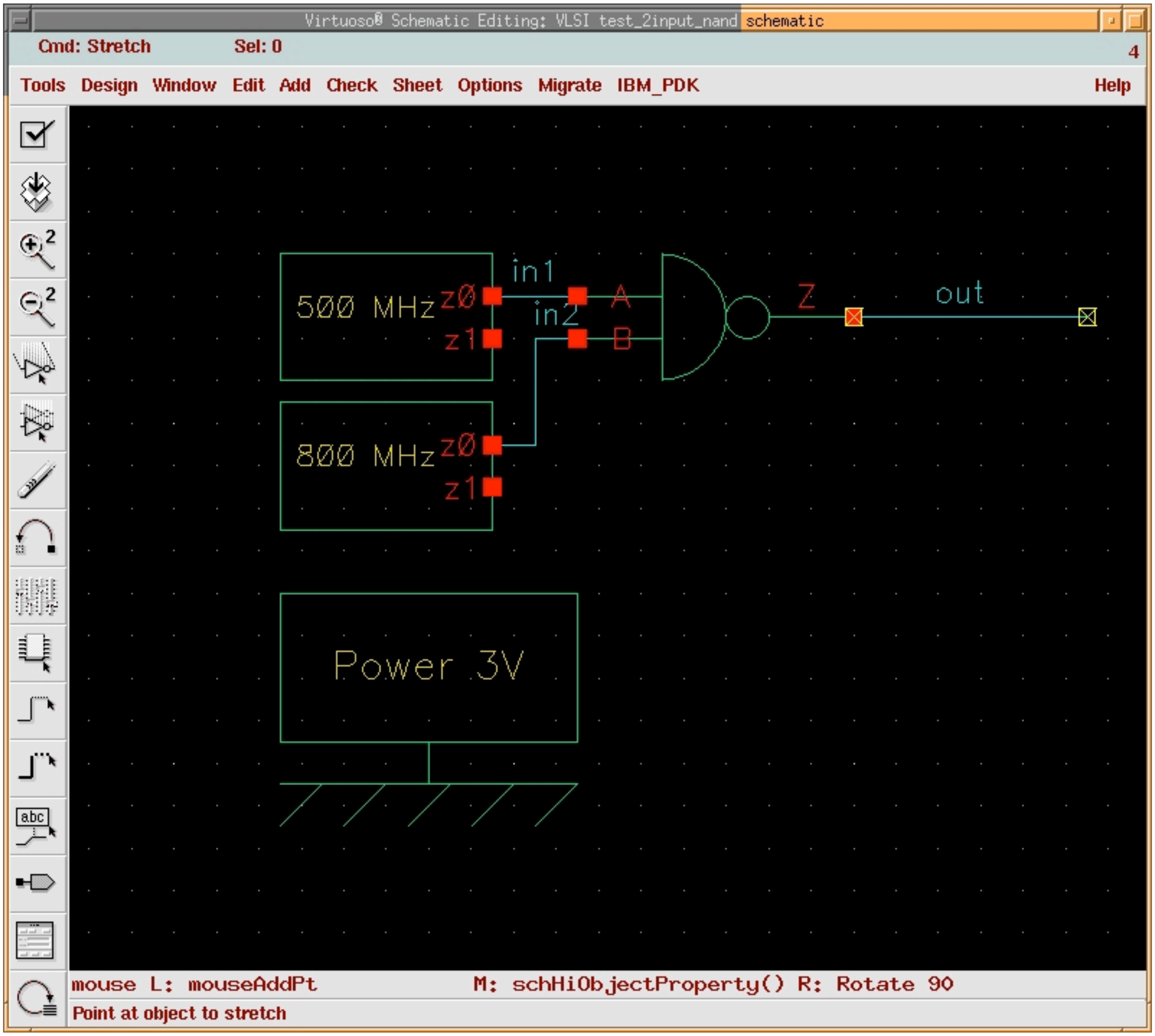


Figure 16- Cell view for testing NAND gate

Now simulate the operation of NAND gate and verify its correctness.

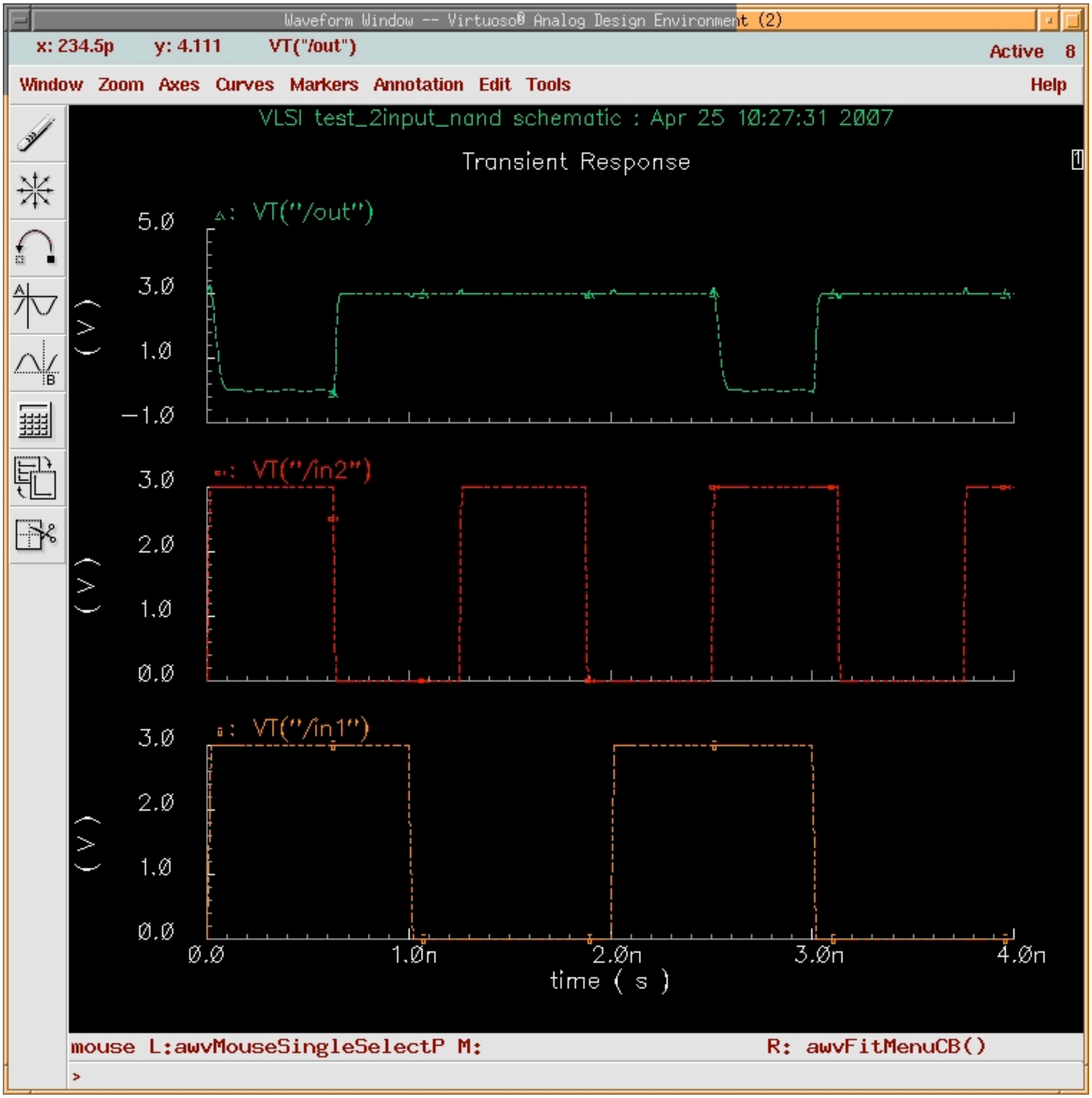


Figure 17-Simulation result of NAND gate

OK! We are done with NAND gate too. Now we can use our NAND and inverter gates to make AND gate.

Make a new cellview and add the Symbol of inverter and NAND gate on it and complete it as can be seen in figure 18.

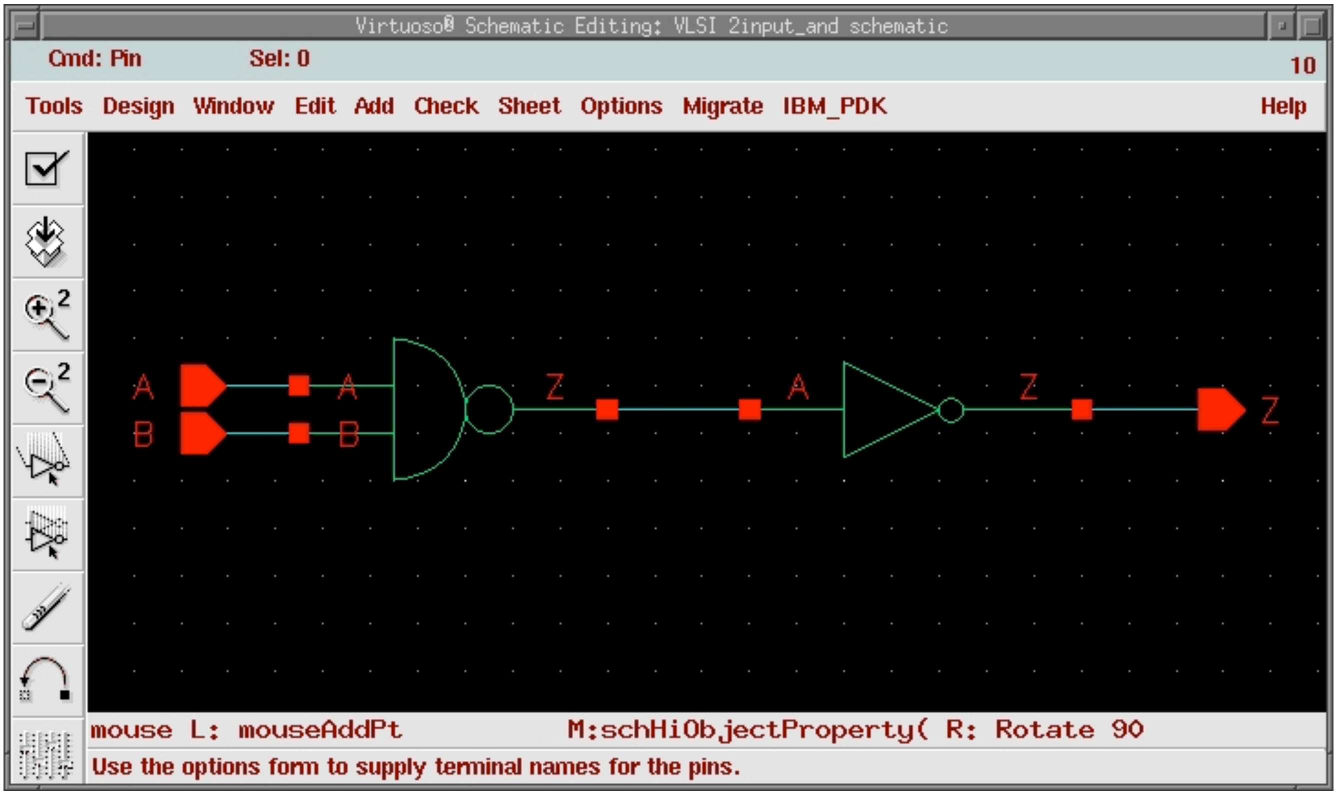

Figure 18-Schematic of AND gate

Make a symbol for AND gate.

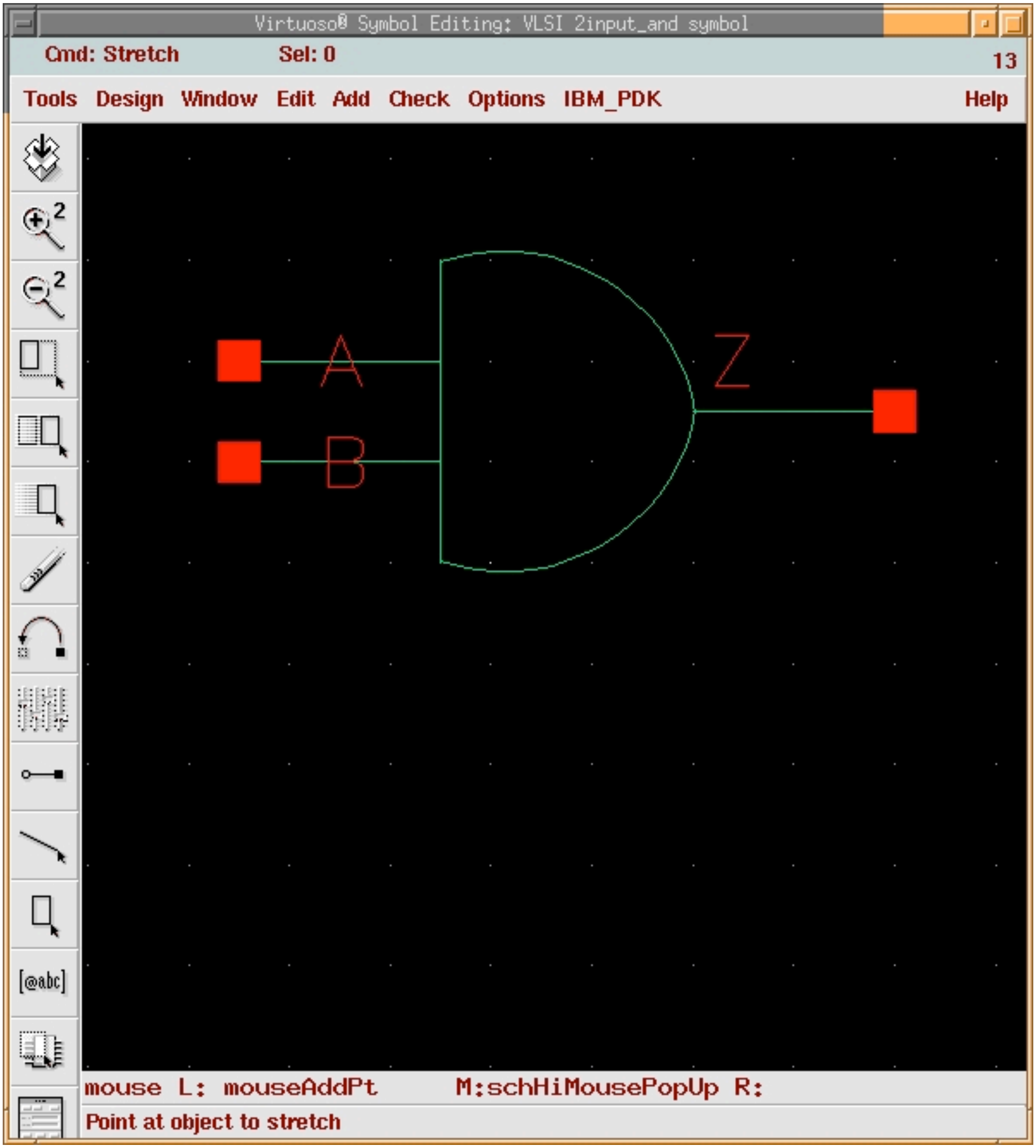


Figure 19- Symbol of AND gate

To test the AND gate and simulate its operation, make another cellview.

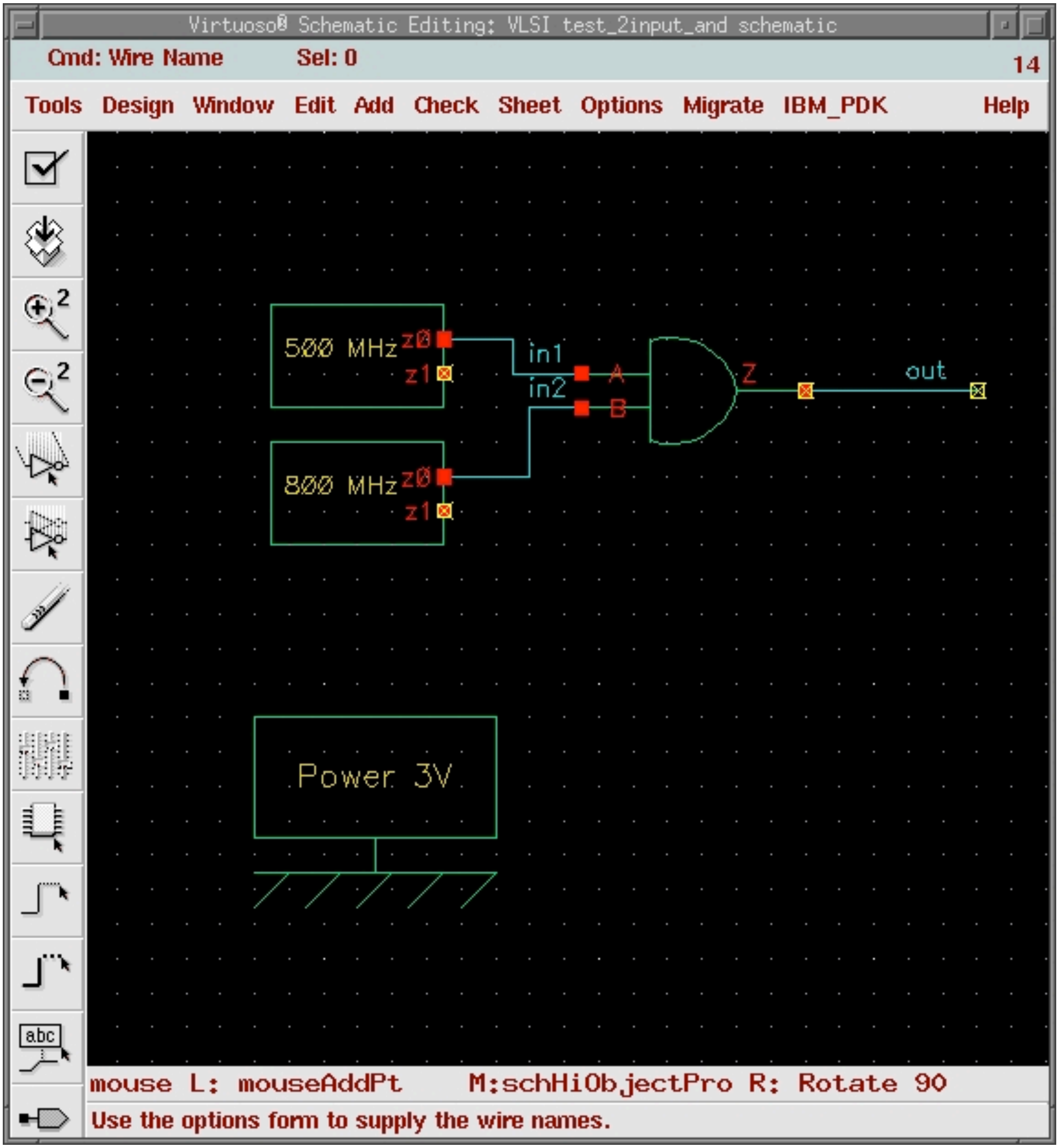

Figure 20 –Cell view for testing AND gate

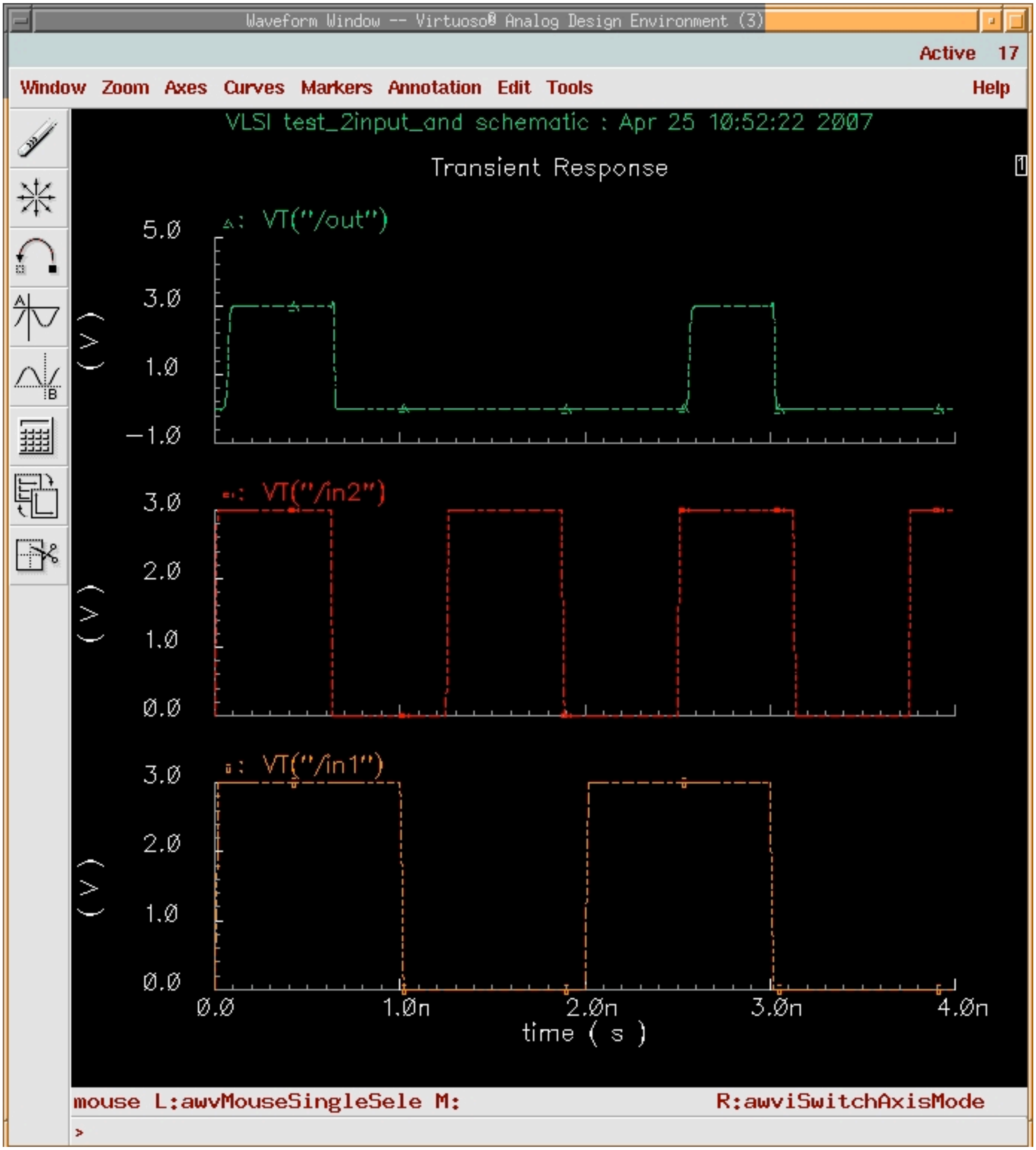

Figure 21- Simulation result of AND gate

## 5. LAYOUT (Inverter)

**Layout Introduction**

This is the most difficult part of the process. It is a good idea to play with the display control and also become familiar with the rules so that you get the idea of the dimensions. The key points you should know are:

Layer Names:

| | |
|---|---|
| ca | metal one to polysilicon contact |
| mc | metal contact |
| rx | thin oxide layer (active transistor area) |
| nw | n-well (used for pfets) |
| dt | deep trench-used to isolate n-well around PFET |
| pc | polysilicon, used for gates and interconnecting gates |
| m1 | metal 1, lowest layer metal |
| v1 | via 1 connects m1 to m2 |
| m2, m3 m4 | other metal layers |
| v2, v3, v4 | other interconnect vias |
| lm | Last metal, use for power rails. Middle click on lm dg and then Window->Redraw to make this layer invisible. |

**Creating a Layout**

We are going to use the schematic of the inverter we've already made to implement the layout of an inverter. So choose **library manager-> VLSI (from Library) -> inverter (from Cell) ->schematic (from View)**. This opens the schematic Editing window.
Now choose **Tools->Design synthesis->layout XL**, The startup Option window appears asking "Create a new or open an existing cell view?" choose **Create New** and press **OK**. "Create new file" window appears. Make sure the library name should be "VLSI", cell name be "inverter", view name be "layout" and the tool be "Virtuoso". Press **OK**. Now the Layout editing and LSW windows open.
In Layout editing window choose **Design->Gen from source.** This opens another window called "Layout generation options". In "layout generation" select **I/O pins**, **Instances** and deselect **Boundary**. In **I/O Pins-> Layer/Master** select **M1-pr** and press **Apply**. In **pin label shape** chose **Label** then in **Pin Label Options** change the **height** to 0.5 or smaller value if you want your labels to appear smaller. Finally press **OK**.

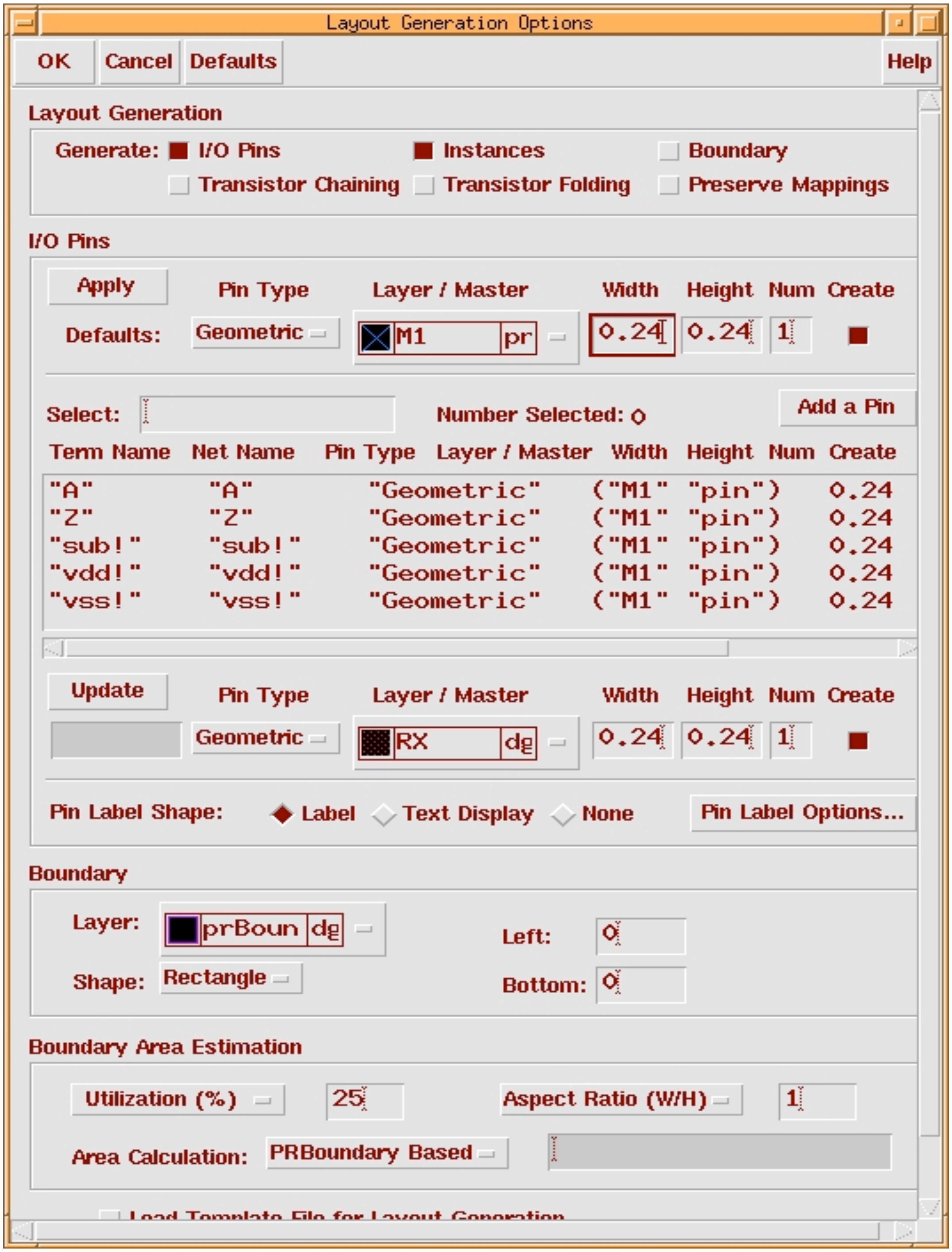


Figure 22- "Layout Generation Option" window

Now you can see a PFET, NFET and substrate appear in layout editing window. As you know the green bar represents Gate and the blue bars represent drain and source.

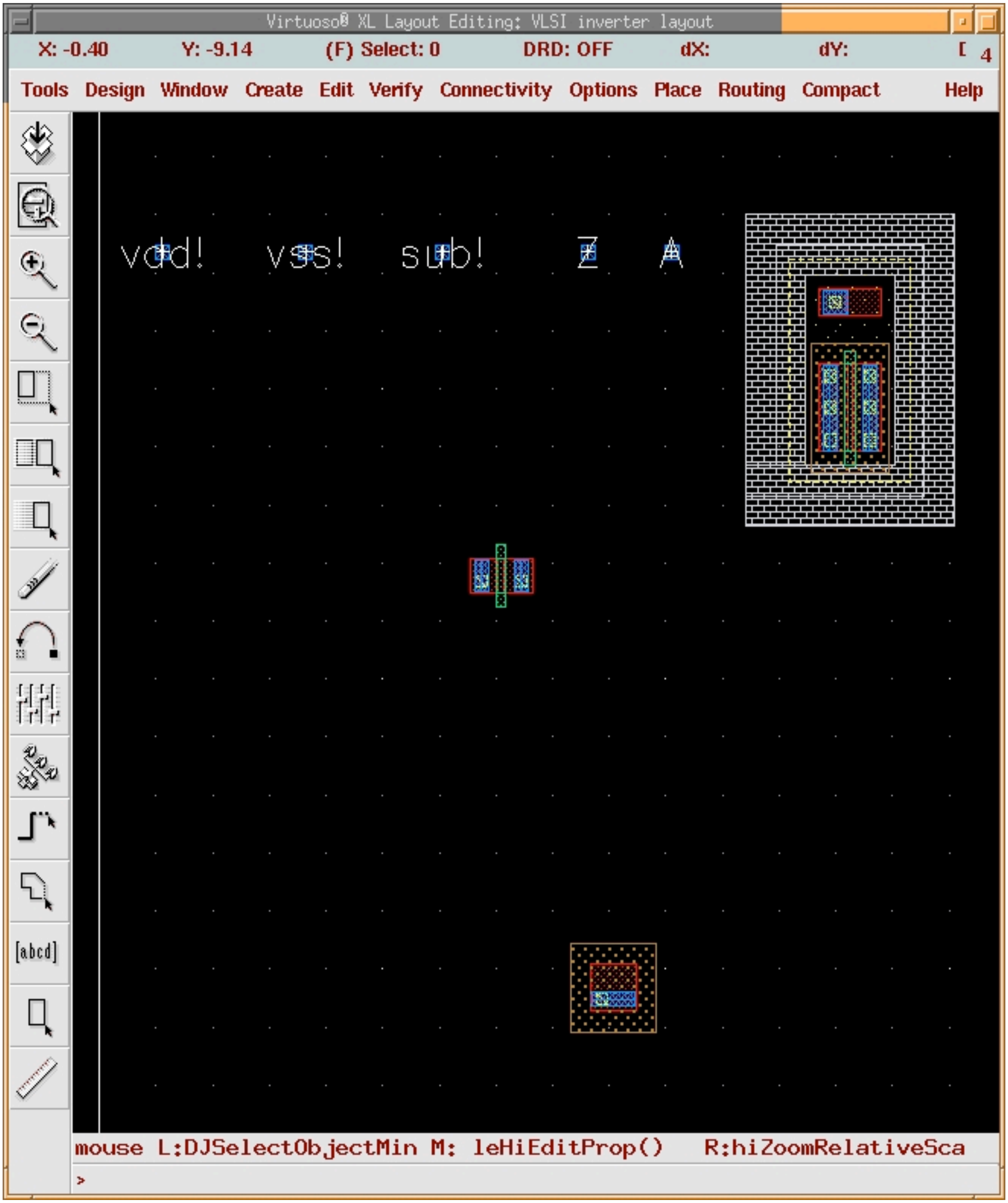
Virtuoso® XL Layout Editing: VLSI inverter layout
X: -0.40 Y: -9.14 (F) Select: 0 DRD: OFF dX: dY:
Tools Design Window Create Edit Verify Connectivity Options Place Routing Compact Help
vdd! vss! sub! Z A
[abcd]
mouse L:DJSelectObjectMin M: leHiEditProp() R:hiZoomRelativeSca
>

Figure 23

Now move the NFET to the bottom of the PFET and move substrate to the bottom of the NFET.

As we are engineer we should do precise works! So use ruler to align the gates of NFET and PFET together. Click on the picture of ruler in left hand side of the window and draw it from PFET gate to NFET gate. Make sure they are aligned. If not just move one of them so that their gates become aligned with each other.

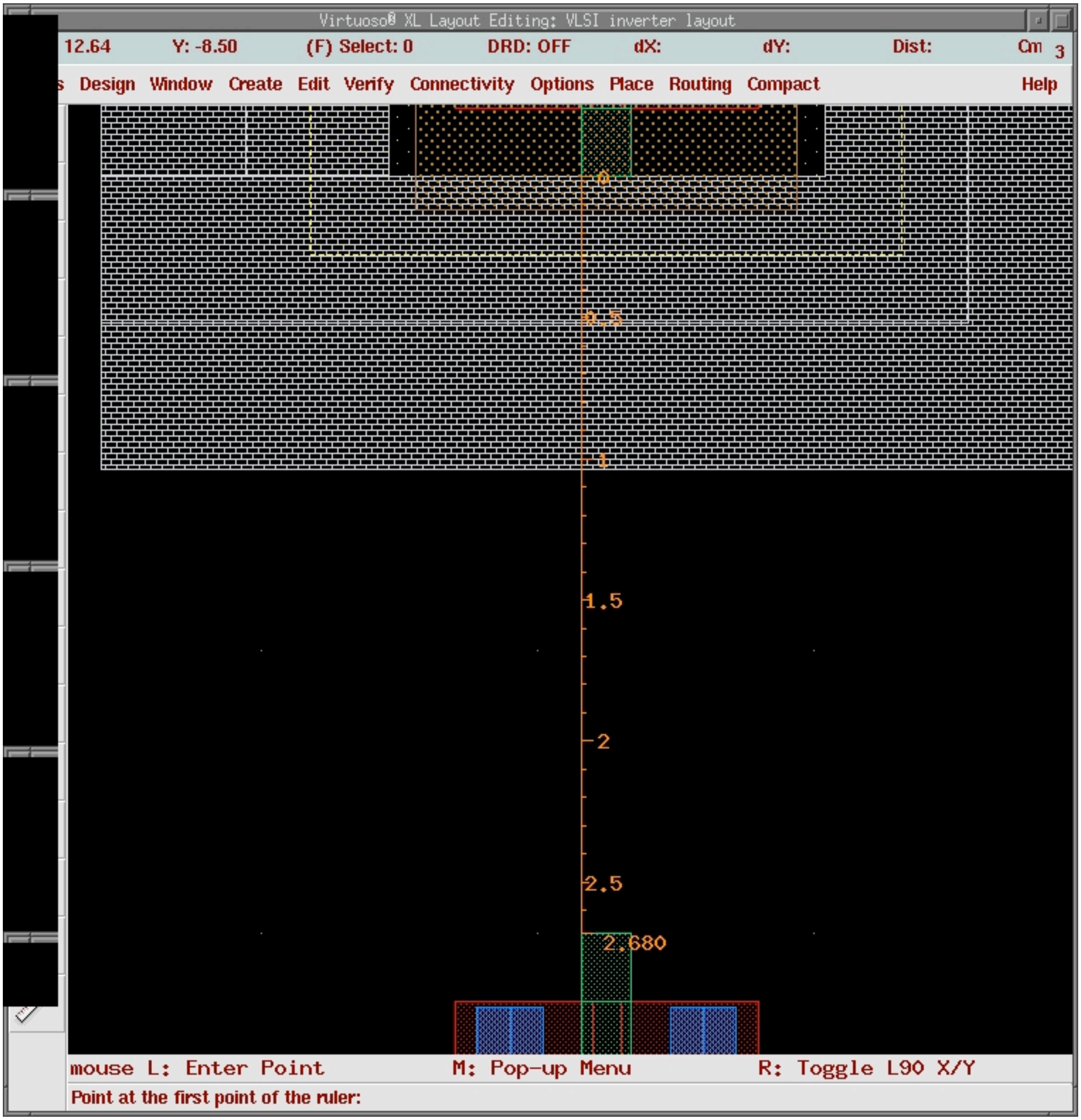


Figure 24- Aligning the gates of PFET and NFET in inverter layout

**Note5**. You can delete the ruler by “**Ctrl+u**”.

As you know in the schematics of the inverter the gates are connected together and the input is applied to gates. So we want to connect the poly silicon layer of NFET and PFET together, which is their gate. In order to do this single click on the polysilicon, press "**W**" and draw a path from PFET gate to NFET gate.

**NOTE6**.To terminate drawing one layer, you should left double click very fast.

**NOTE7**.Make sure the connecting path be aligned with one side of both NFET and PFET gate otherwise it causes error in future!

Move the input (which has the Label A in our inverter) to the left hand side of the connecting poly layer that you have already drawn. Draw a poly path from the connecting layer to “A” but don't attach the layer to A. Press **F3**, “Create path” window opens, in "change to layer" part select **M1-dg** and then hide. A “via” appears. Put it at the end of the poly layer. Click on the metal of “via”, press "**W**" a window opens which ask you about the object. Chose "**Net A on M1 drawing**", then **Ok**. Draw a metal path from via to input A.

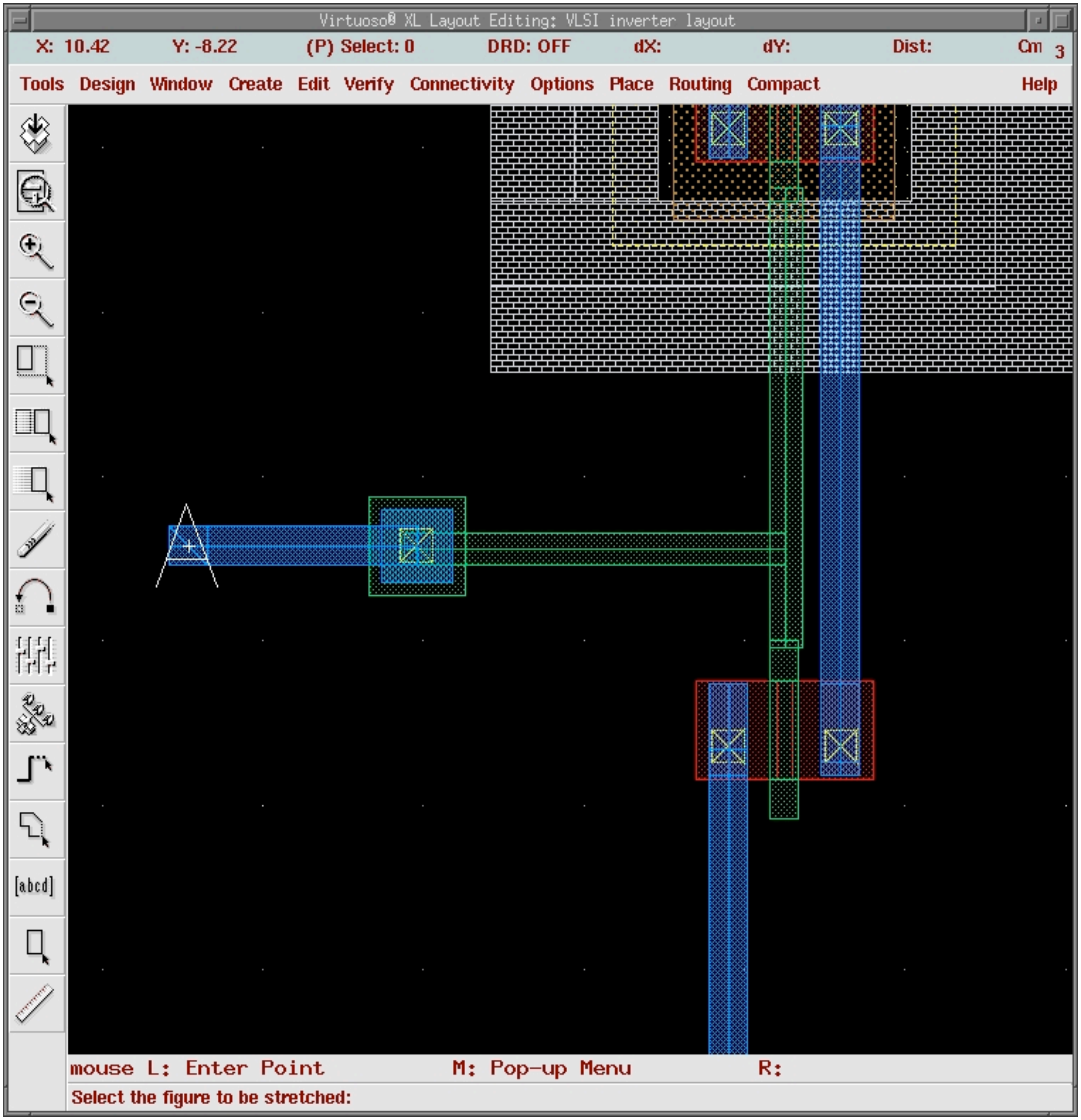


Figure 25- Inverter input in layout

At this point we want to connect the output; therefore we should connect the drain of PFET and NFET together.

Select M1-dg, press "**W**" and draw a path between NFET and PFET sources. Move the output (which has the Label Z in our inverter) to the right hand side of the connecting metal path that you have already drawn and draw a metal path from the connecting layer to Z.

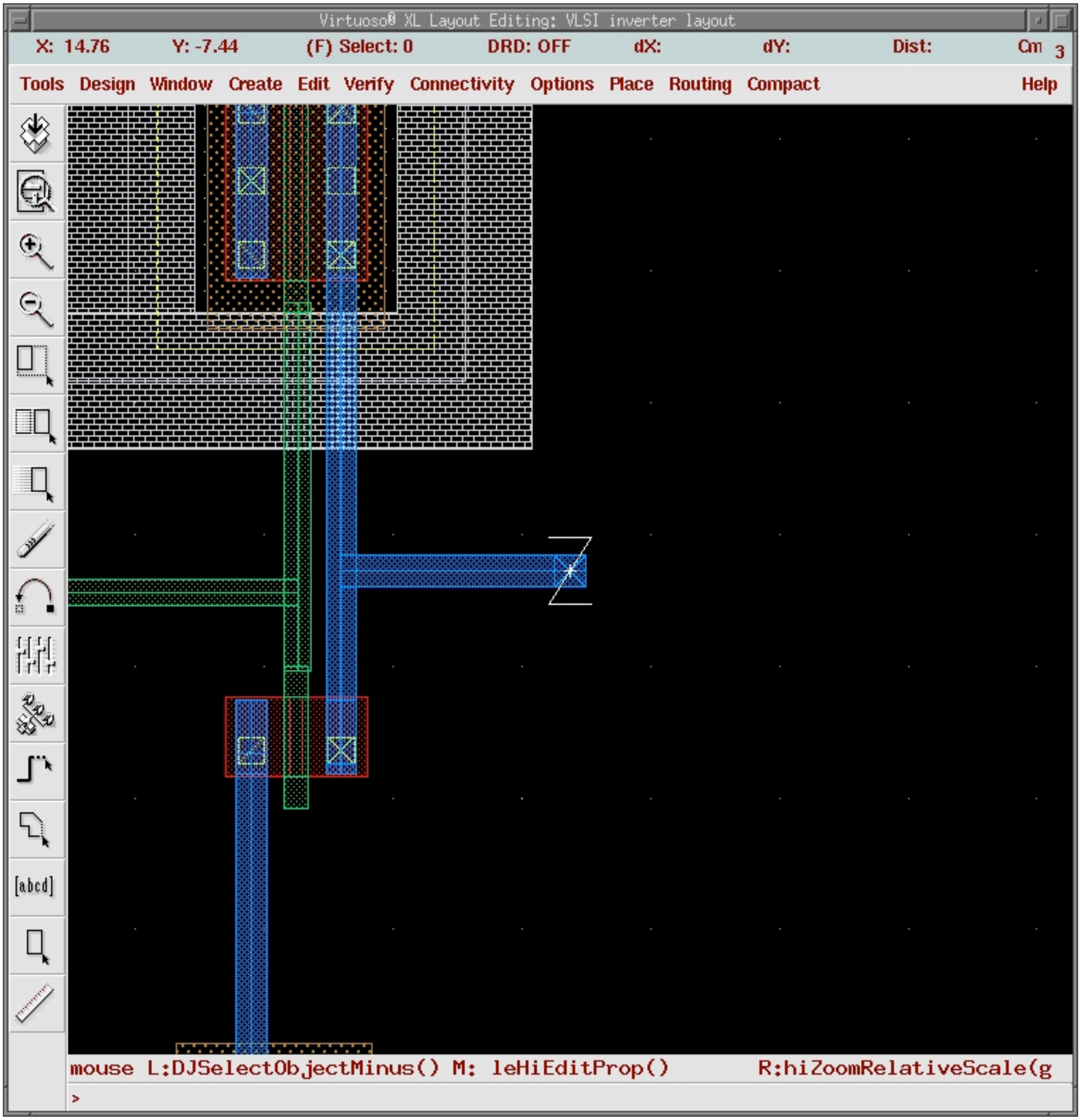

Figure 26- Inverter output in layout

Now move vdd! to top of the PFET and connect the source of PFET to vdd! with **M1-dg** path. Then draw the vdd rail as seen in figure 27.

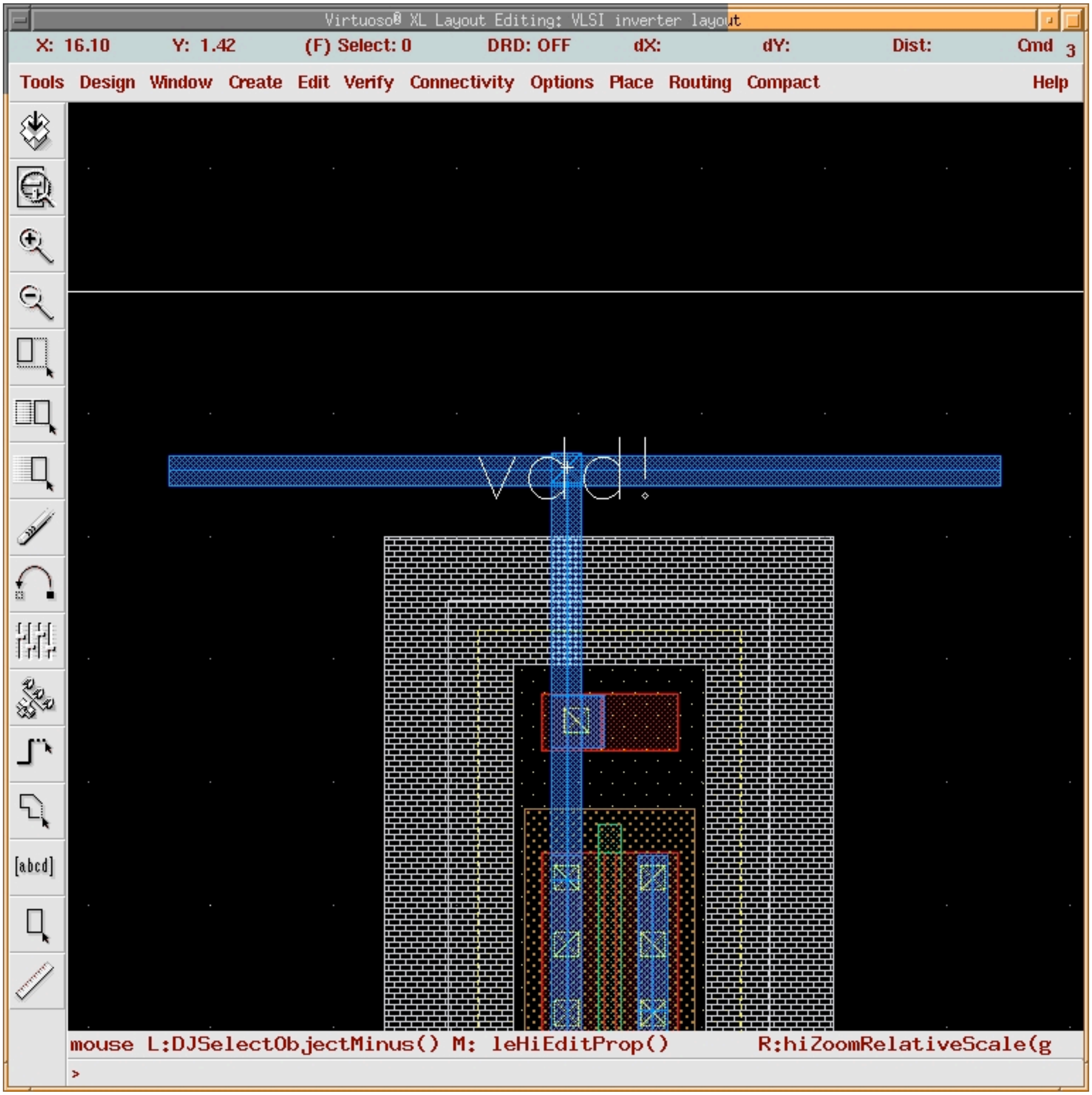


Figure 27- Vdd rail in inverter layout

Now we should connect the source of NFET to ground, do it by M1-dg path. Then move the vss! to the substrate and connect it to the path you have already drawn. Then draw a ground rail as seen in figure 28.

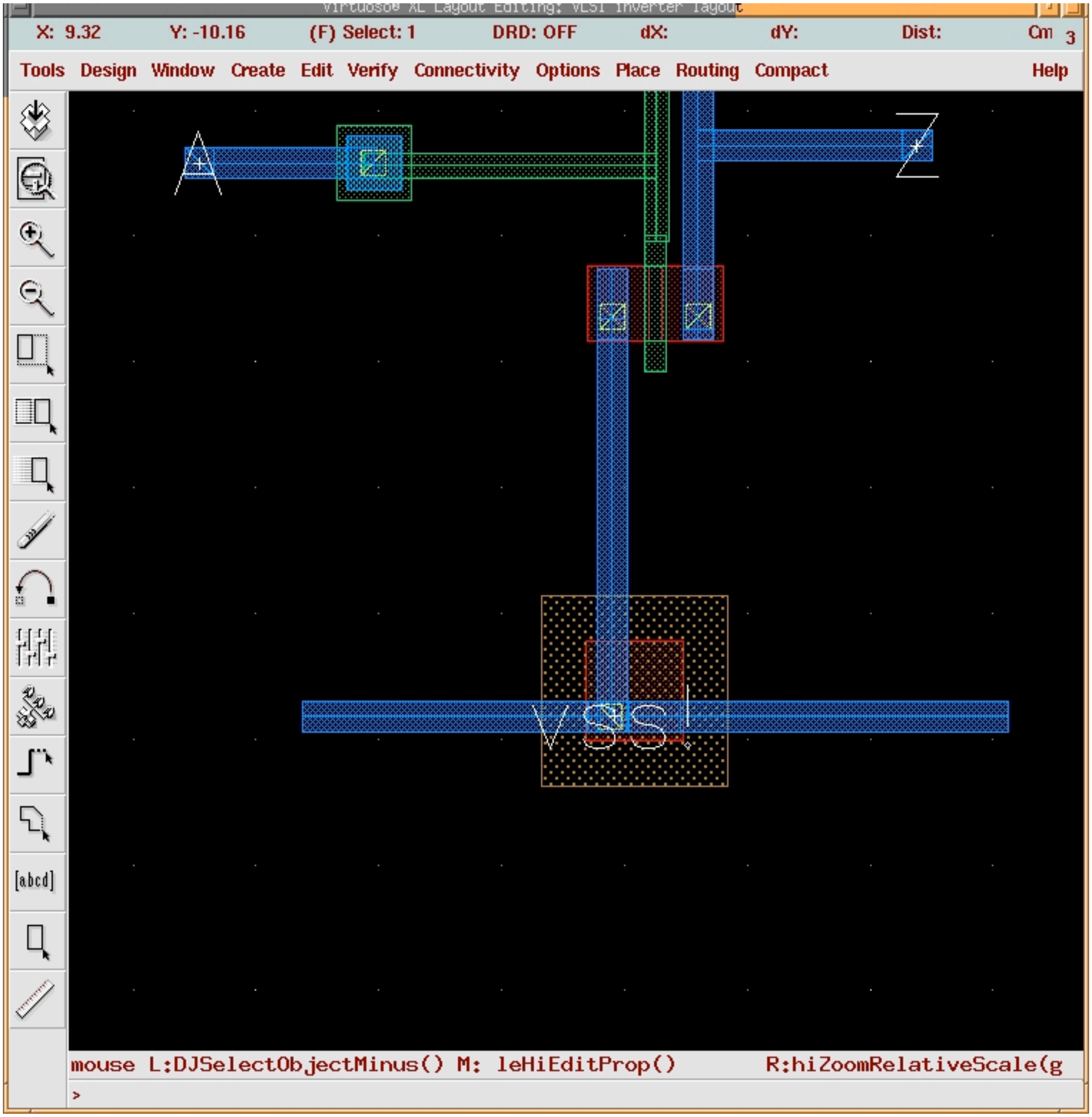


Figure 28- Vss rail in inverter layout

OK! We are done with the layout. Now your layout should look like figure 29.

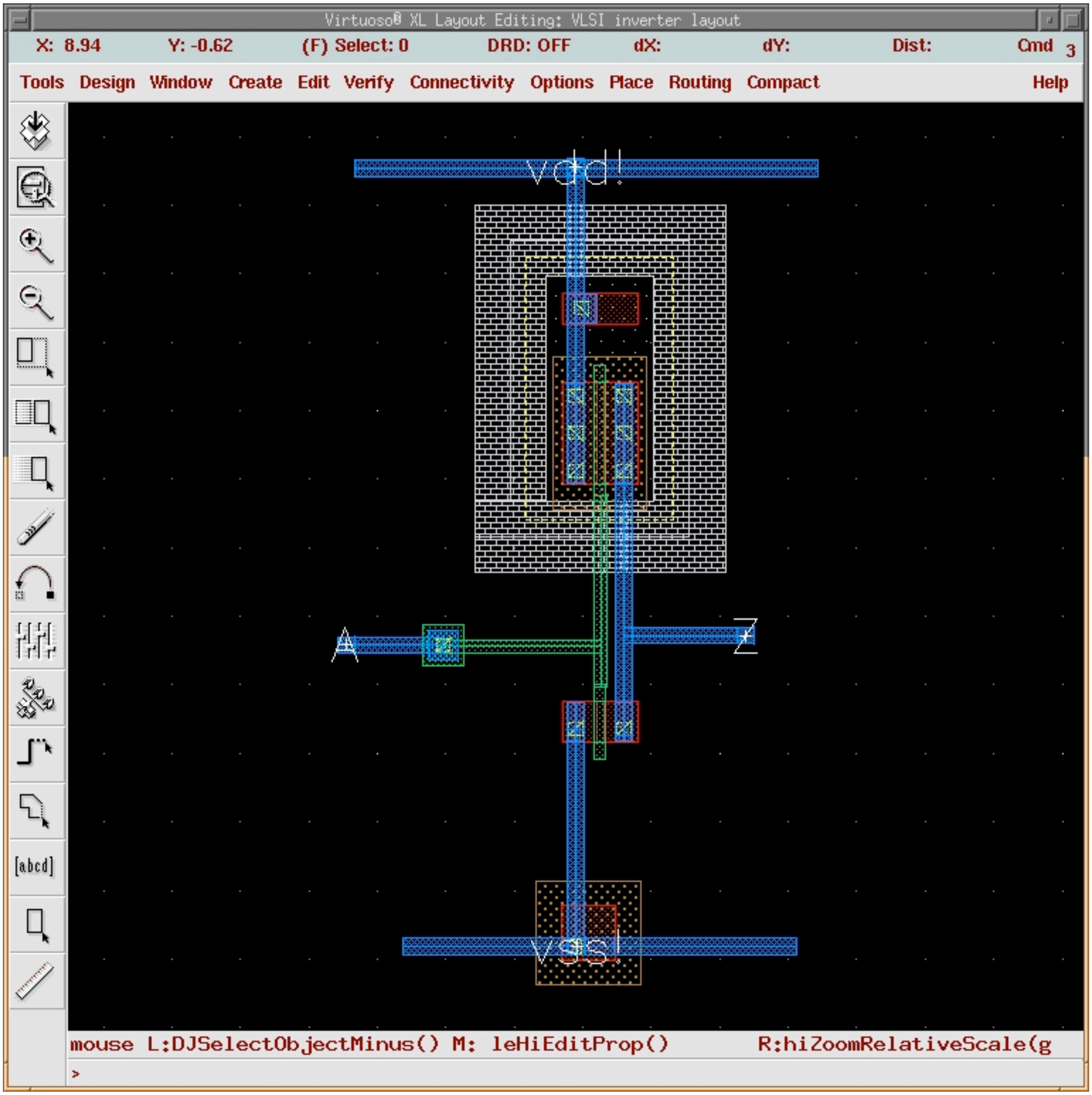
Virtuoso® XL Layout Editing: VLSI inverter layout
X: 8.94
Y: -0.62
(F) Select: 0
DRD: OFF
dX:
dY:
Dist:
Cmd 3
Tools
Design
Window
Create
Edit
Verify
Connectivity
Options
Place
Routing
Compact
Help
vdd!
A
Z
vss!
[abcd]
mouse L:DJSelectObjectMinus() M: leHiEditProp()
R:hiZoomRelativeScale(g

Figure 29

**Checking Design Rules and correcting Errors**

To perform design rule checking, choose **Verify-> DRC**, "DRC" window opens, press **Set Switches** and chose **GridCheck**, deselect **Echo Commands**, press **OK**.

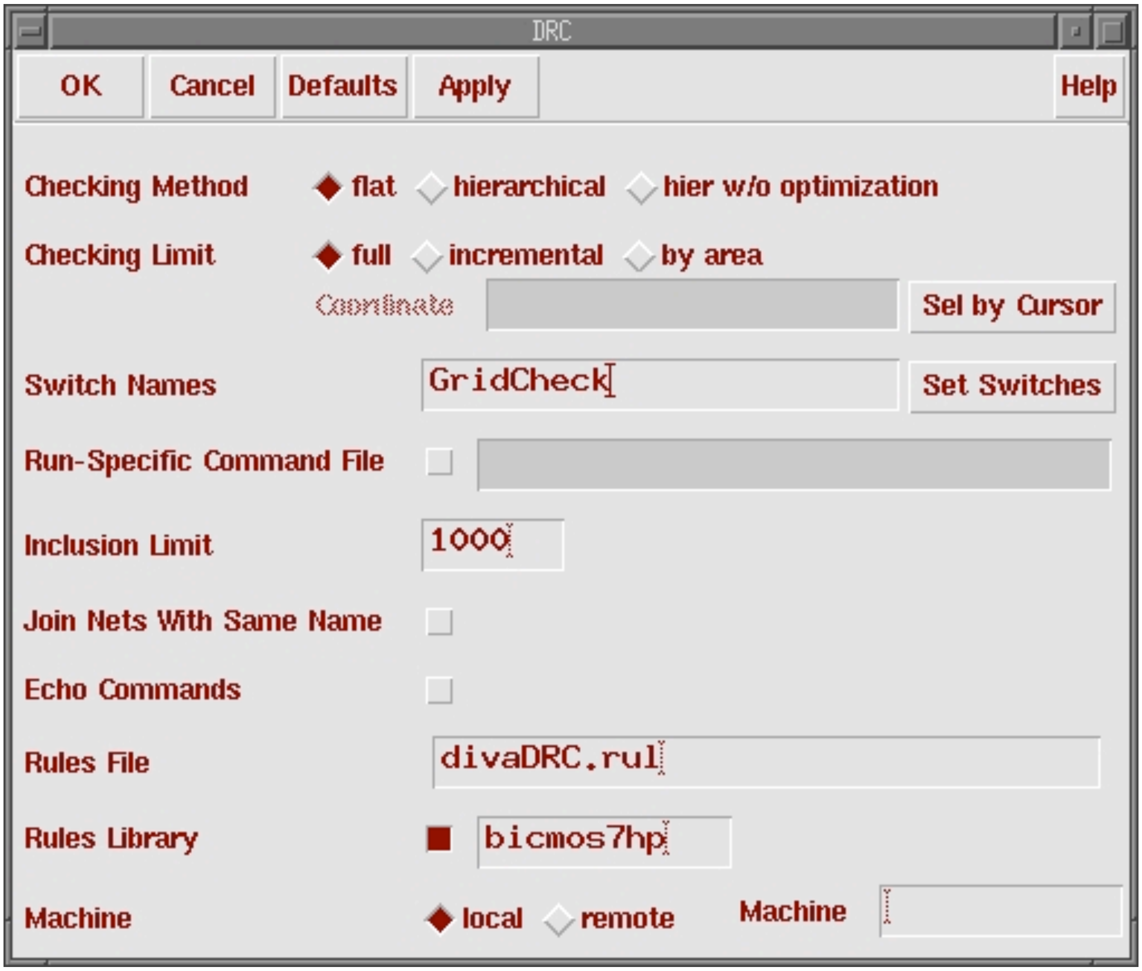

Figure 30- DRC window

Now check the ICW, if you are done with 2 errors that are INFO ones (they are shown in figure 31), you are done, otherwise you should go back to your layout and correct the mistakes.

**NOTE 8**.Pay attention that errors are defined by white cross on the layout so you can easily find and correct them.

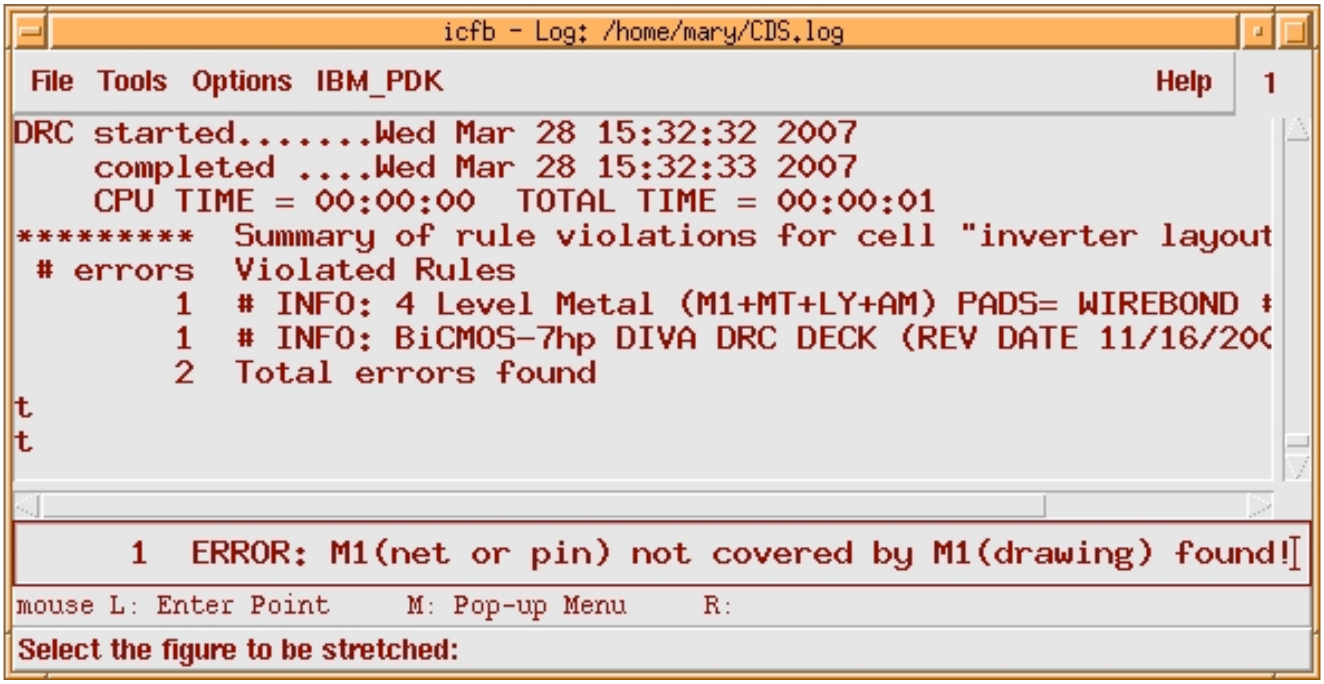

Figure 31- DRC check result in ICW

**Layout vs. Schematic Check (LVS)**

The final step is LVS. The purpose of LVS is to see if the layout you created matches the schematic. It must match in pin names, direction, logical nets and the circuit parameters. A netlist from your schematic is compared to a netlist from your layout. This is called the extracted view. To create an extracted view from the "layout" window chose **Verify-> Extract**, the "Extractor" window opens, deselect **Echo Commands** and in **Rules File** type **divaEXT5.rul**, press **OK**.

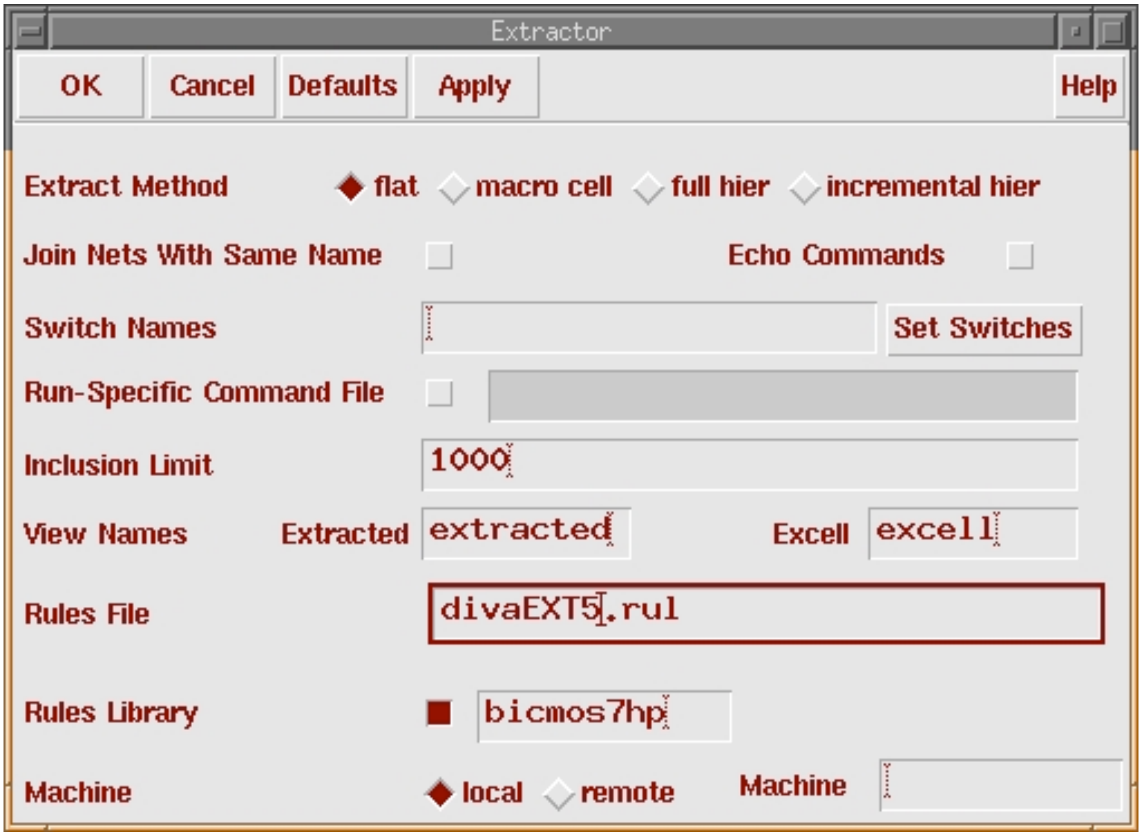


Figure 32- Extractor window

**Note9**. In "Extractor" Window, you can set the switches to **Rparasitic**. It allows you to see the parasitic resistors in extracted file.

Now choose **Verify-> LVS**, in the column of "Create Netlist Extracted" for **Cell** type "**inverter"** (the same name) and for **View** type "**extracted"**.

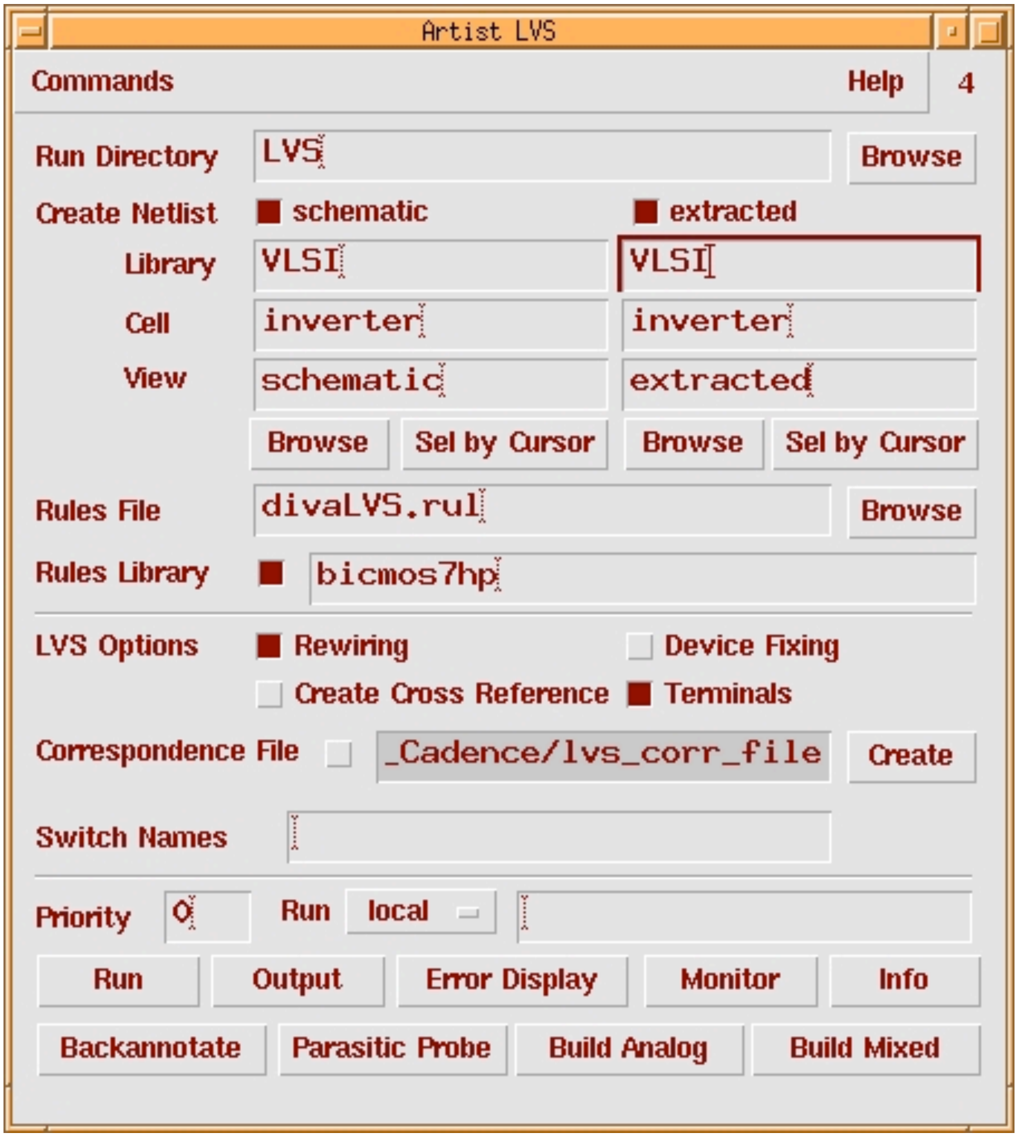


Figure 33- LVS window

Press Run, after you get the message shown in figure 34 press Output.

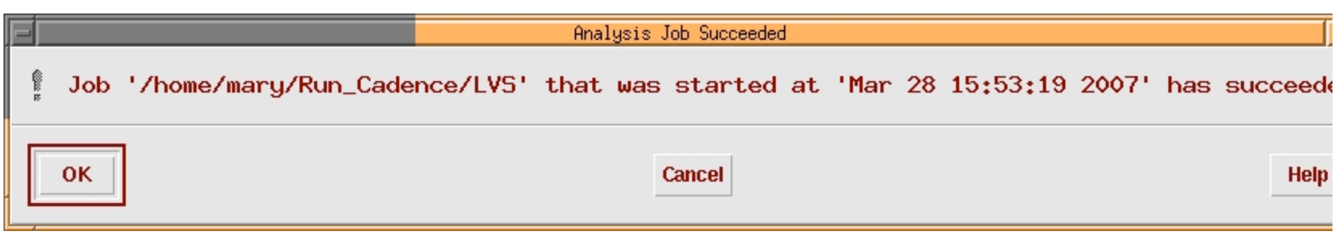


Figure 34

Now another window opens in which you can check if there are any errors in this step or not, if you are done with no error, you see the window like figure 35.

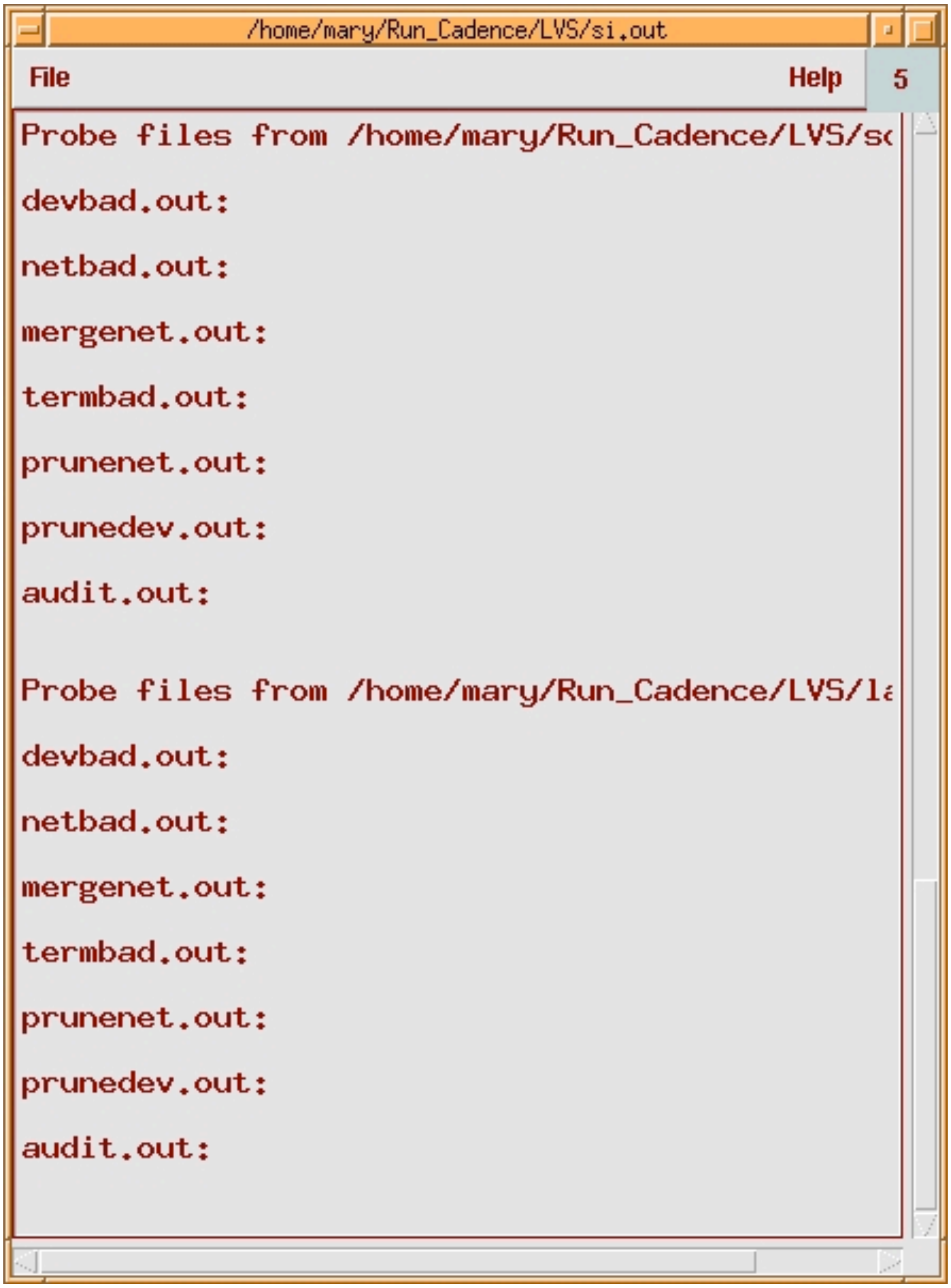


Figure 35- Output result of LVC check

Finally from "Schematic" window, chose **Tools->Analog Environment->Setup->Environment Option**, The "Environment Option" window opens, in **Switch View list** add "**extracted**" before schematic, press **OK**.

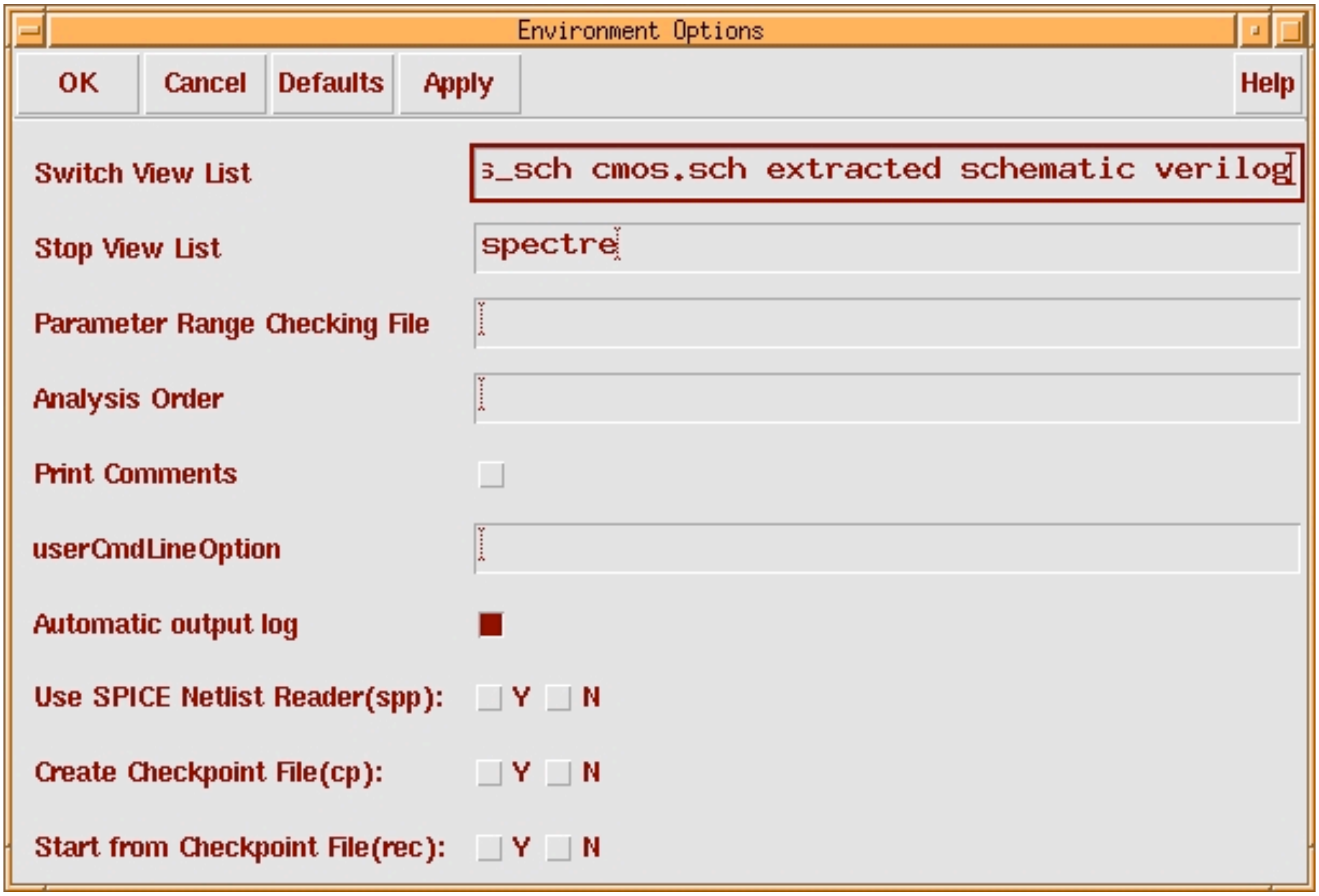


Figure 36- Environment Option window

Now you can check whether your layout works fine or not, if not you should go back to your schematics and modify it again!

### 6. Layout (NAND)

We are going to make the layout of two inputs NAND gate. We created the schematic and symbol of NAND gate before, and we could simulate it successfully.
Then Follow the steps as described in layout tutorial for inverter until you can see NFETs, PFETs and Subc in "Layout Editing" window as seen in figure 37.

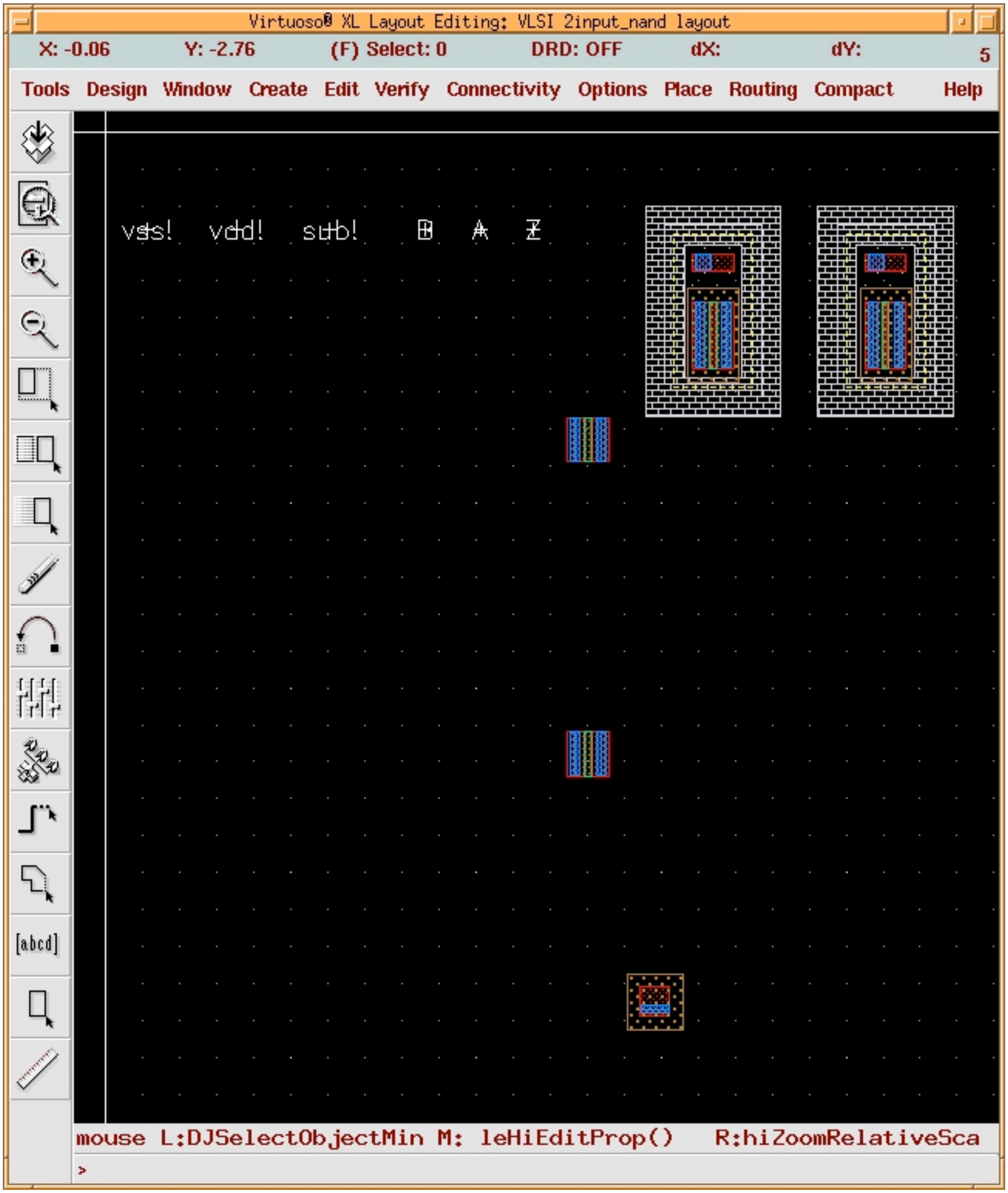


Figure 37

Now you can do the same steps as described before and complete the layout, but that layout won't be the best choice, because you can change the terminals of some nodes and share a point. Remember the best layout is one with most shared points. Fortunately there is a simple way to accomplish this goal; Eular path! The Eular path for NAND gate is drawn in figure 38.

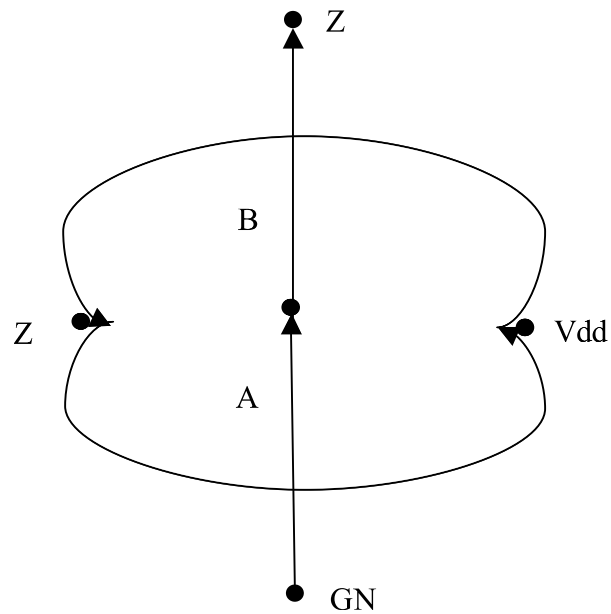


Figure 38- Eular path for two inputs NAND gate

From the Eular path you can easily deduce the stick diagram as shown in figure 39.

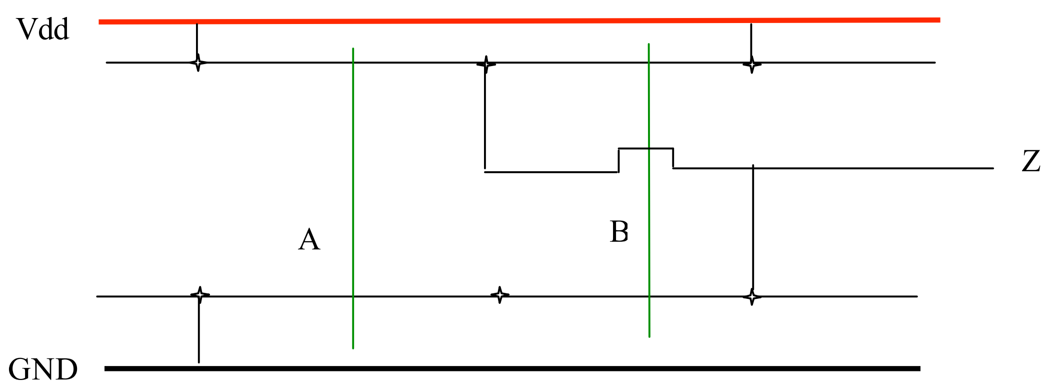


Figure 39- Stick diagram for two inputs NAND gate

OK! You got the concept of what we are going to do with the layout in Cadence. We will apply the same procedure here. In order to do this Cadence give you the chance to use "**finger**". Let's see how? Select one of the PFET; click "**Properties**" from the left panel of the "Layout" window, chose **parameter,** in "Number of fingers" type "**2**". Now you can see the PFET changes to two PFETs with one shared point as in figure 40. You can delete the other PFET.

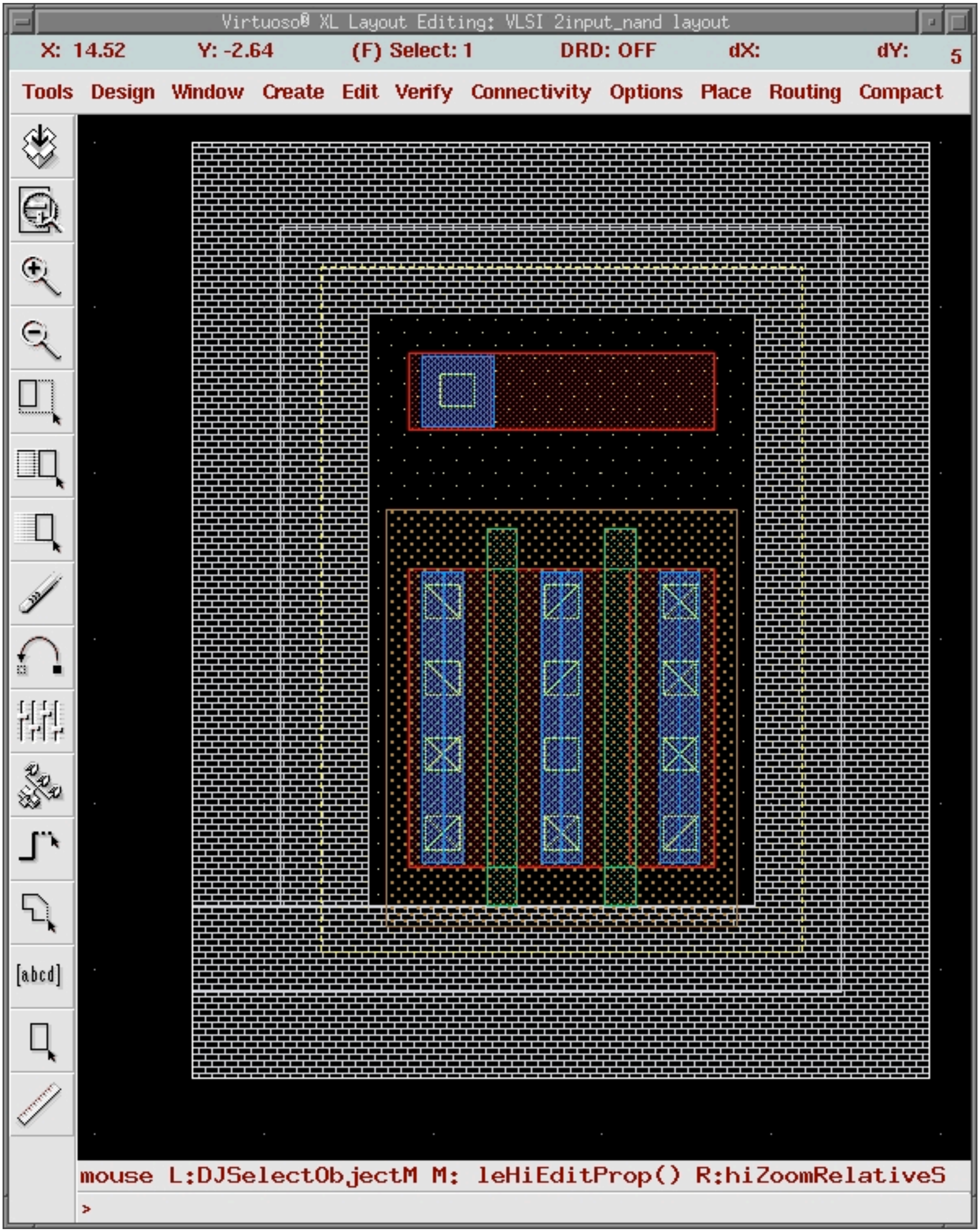

Figure 40- PFET with finger 2

Select one of the NFETs; click "**Properties**" from the left panel of the "Layout" window, chose **parameter,** in "Number of fingers" type "**2**". Now you can see the NFET changes to two NFETs with one shared point as in figure 41. You can delete the other NFET.

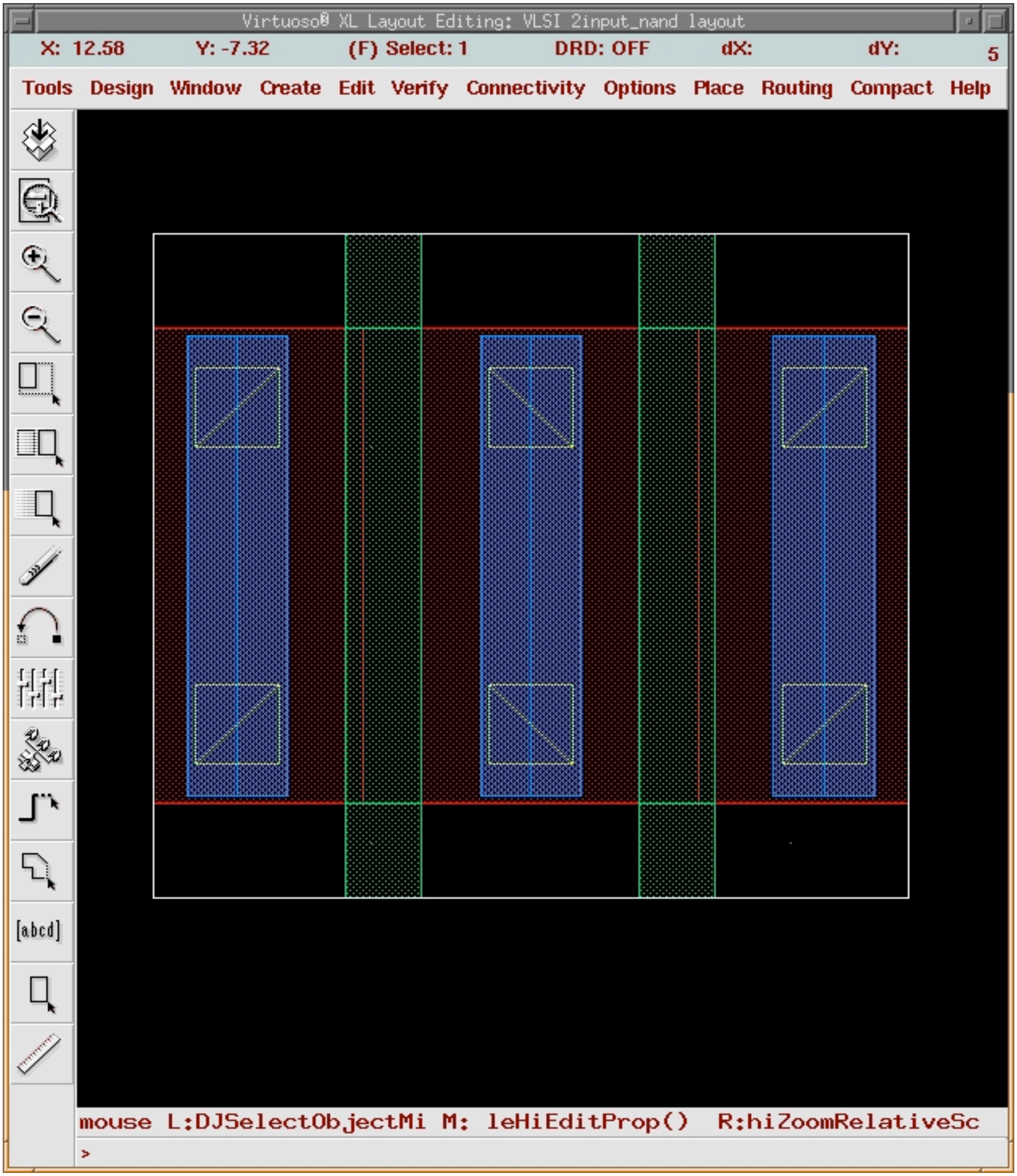


Figure 41- NFET with finger 2

Now you should continue making the layout as explained before. It is a great help to consider the stick diagram while you are making the layout.

**Note10.** After you connect the first gate poly layer to input A, for connecting the second gate poly layer to input B, you should not use metal 1 layer; otherwise the two gates will be connected to each other. The correct way is shown in figure 42.

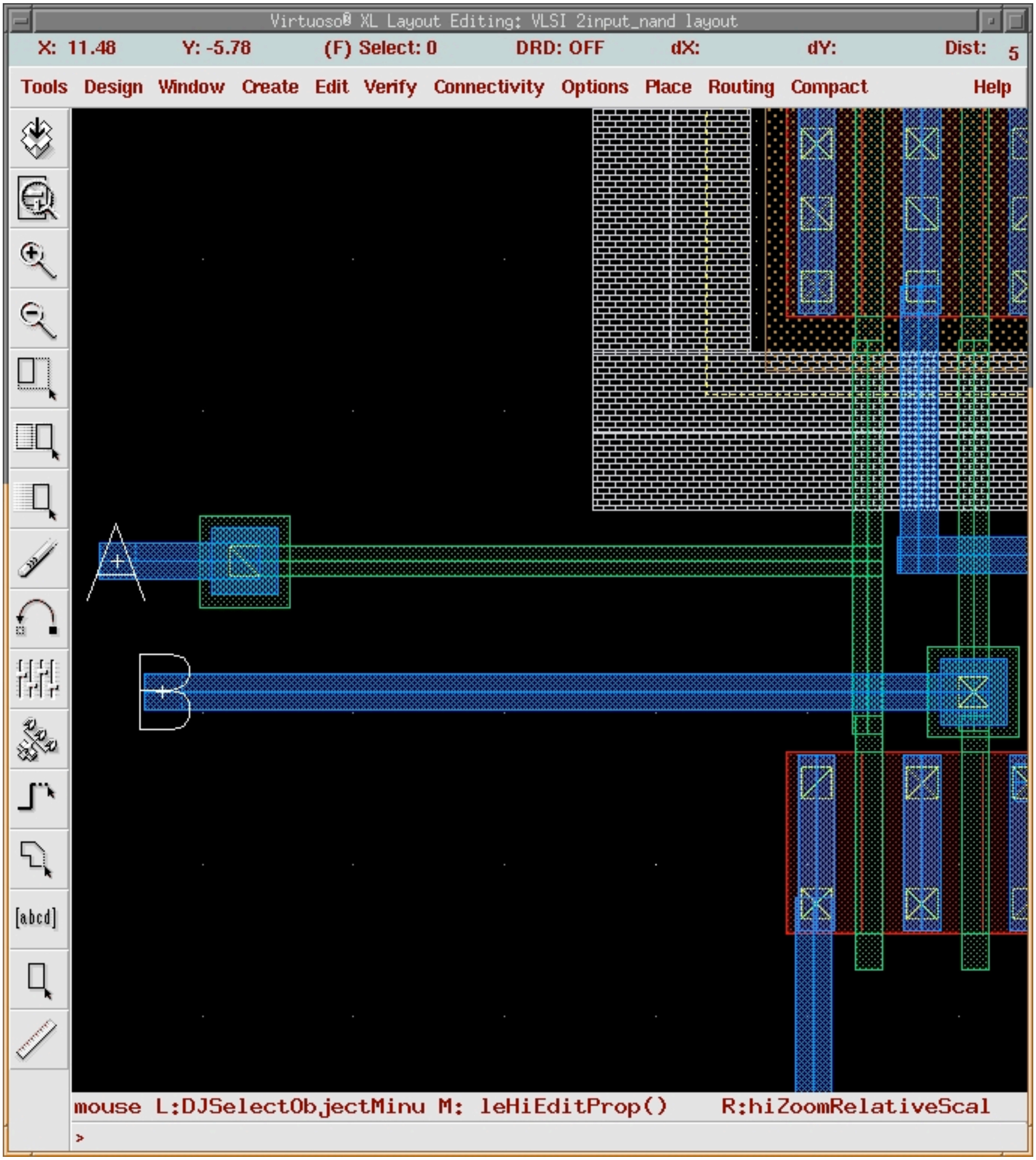


Figure 42- Inputs of NAND gate in layout

**Note11**. Make sure to consider enough space between via of the gate and output, otherwise it will generate DRC error.

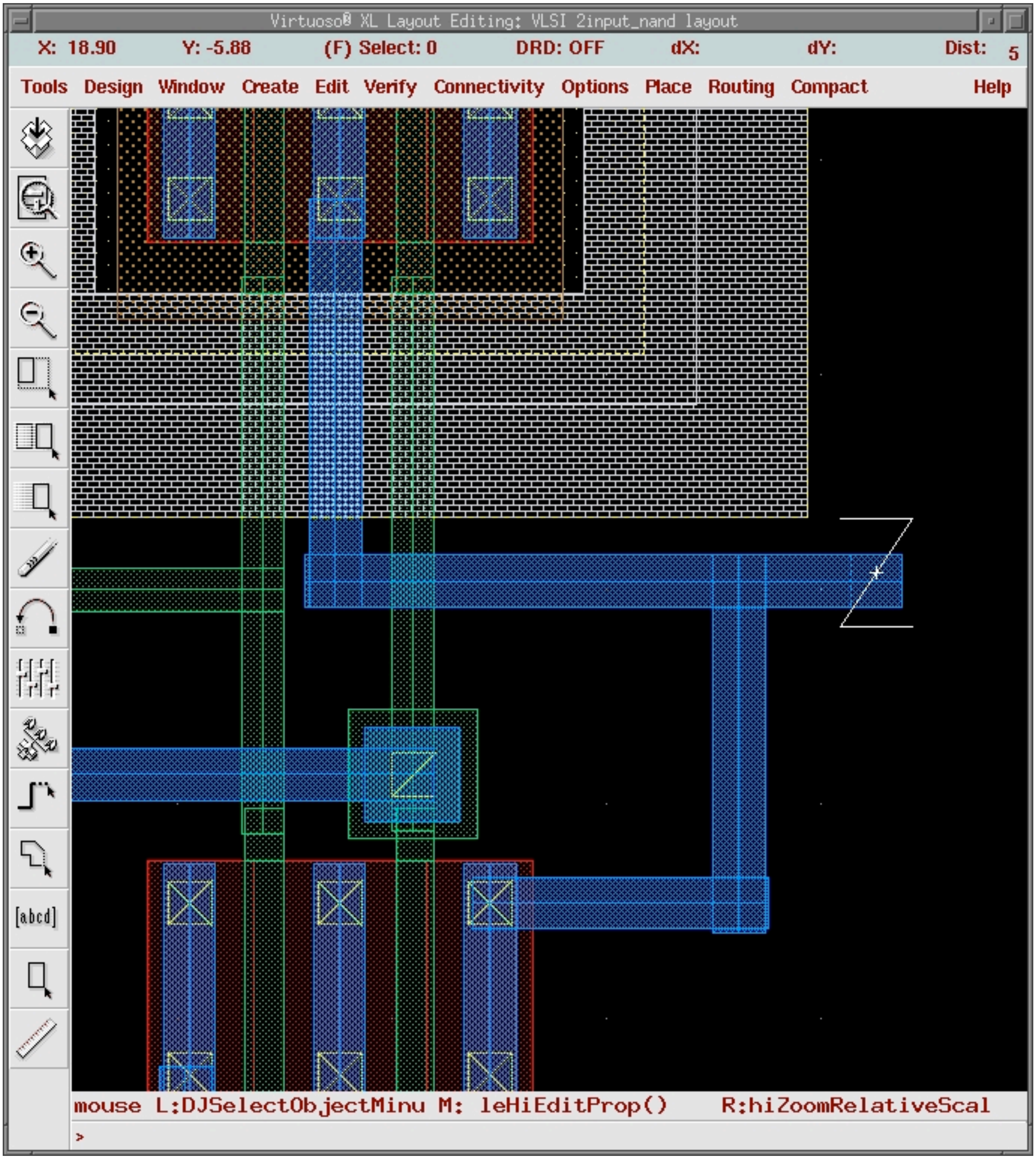


Figure 43- Output of NAND gate in layout

**NOTE12.** The path for vdd and vss rail should be thicker than others. So it would have less resistance.

The complete layout is seen in figure 44.

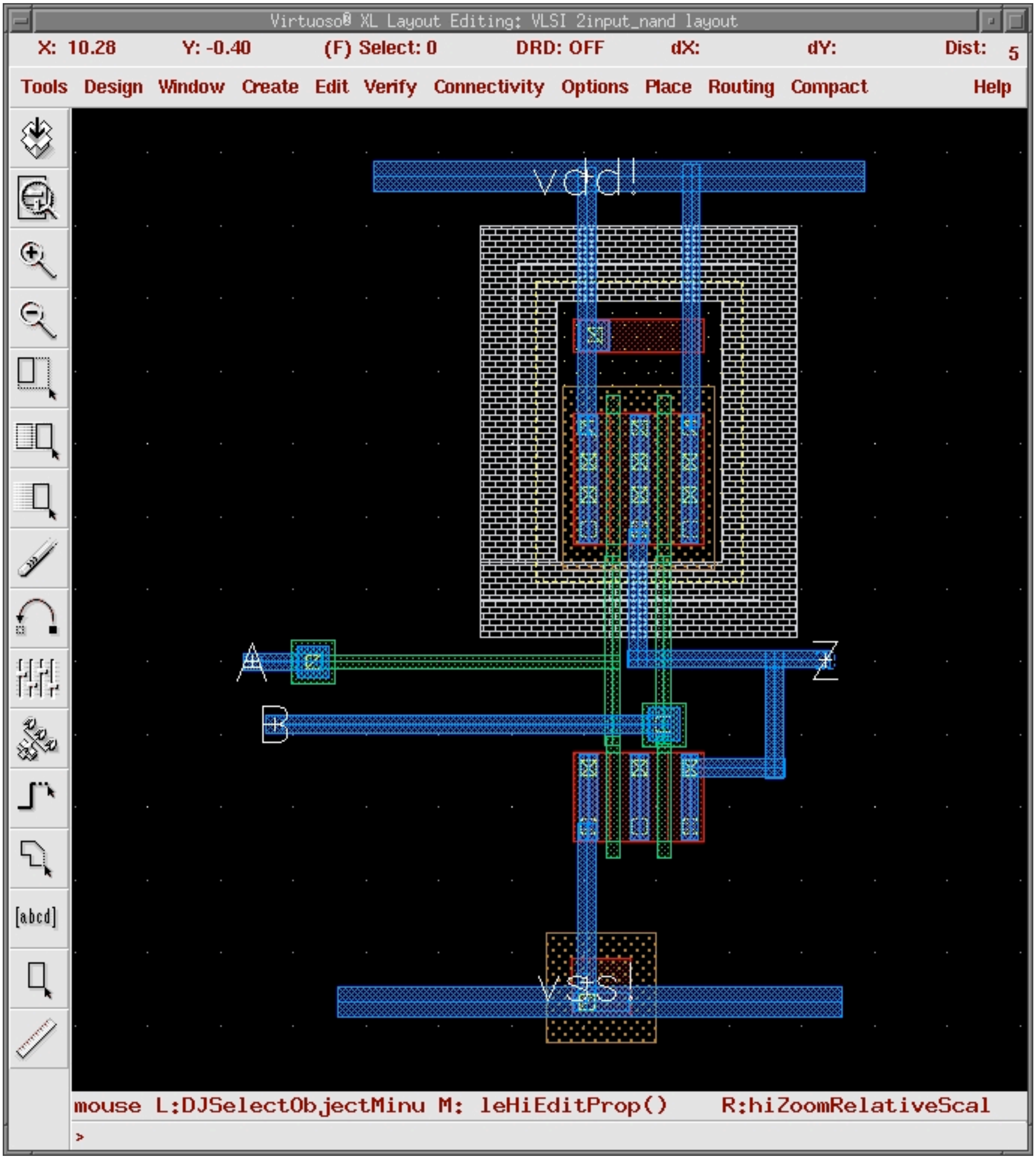


Figure 44- Layout of NAND gate

Then you should verify your layout with DRC, Extract, and LVS (these verifications are described in detail in layout tutorial for inverter). If with each verification, you get any errors, make sure to come back to the layout and modify it.

## 7. From a circuit layout to Chip level

**Chip level schematic, simulation and layout**

A circuit layout is just a part of your design. In order to send out your design and test it, you need to do other things for fabrication purposes. The first thing on your list is to add pads to your design. A pad is the interface between silicon level circuits and the outside world. A design without a pad cannot be powered and tested. It certainly cannot be used later on. The pad that can be used in BiCMOS7hp is bondpadx. To create a chip level schematic, you need connect your inputs/outputs and power/ground signals to your pads. Figure 45 shows the pad properties. You also need a subc to tell Cadence that substrate is tied to the "vss!" Figure 46 shows an example of an inverter circuit at chip level design.

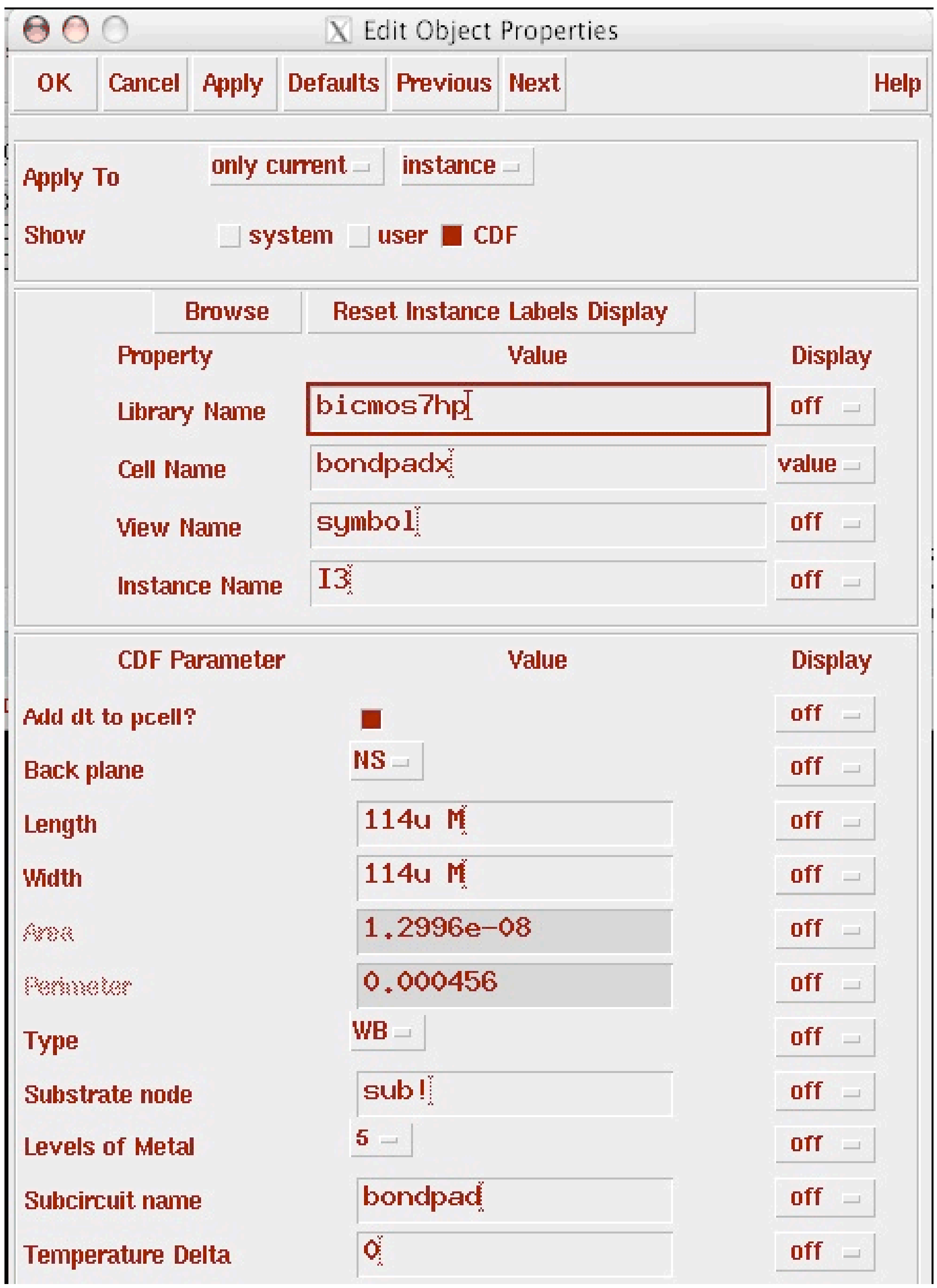


Figure 45-Pad property

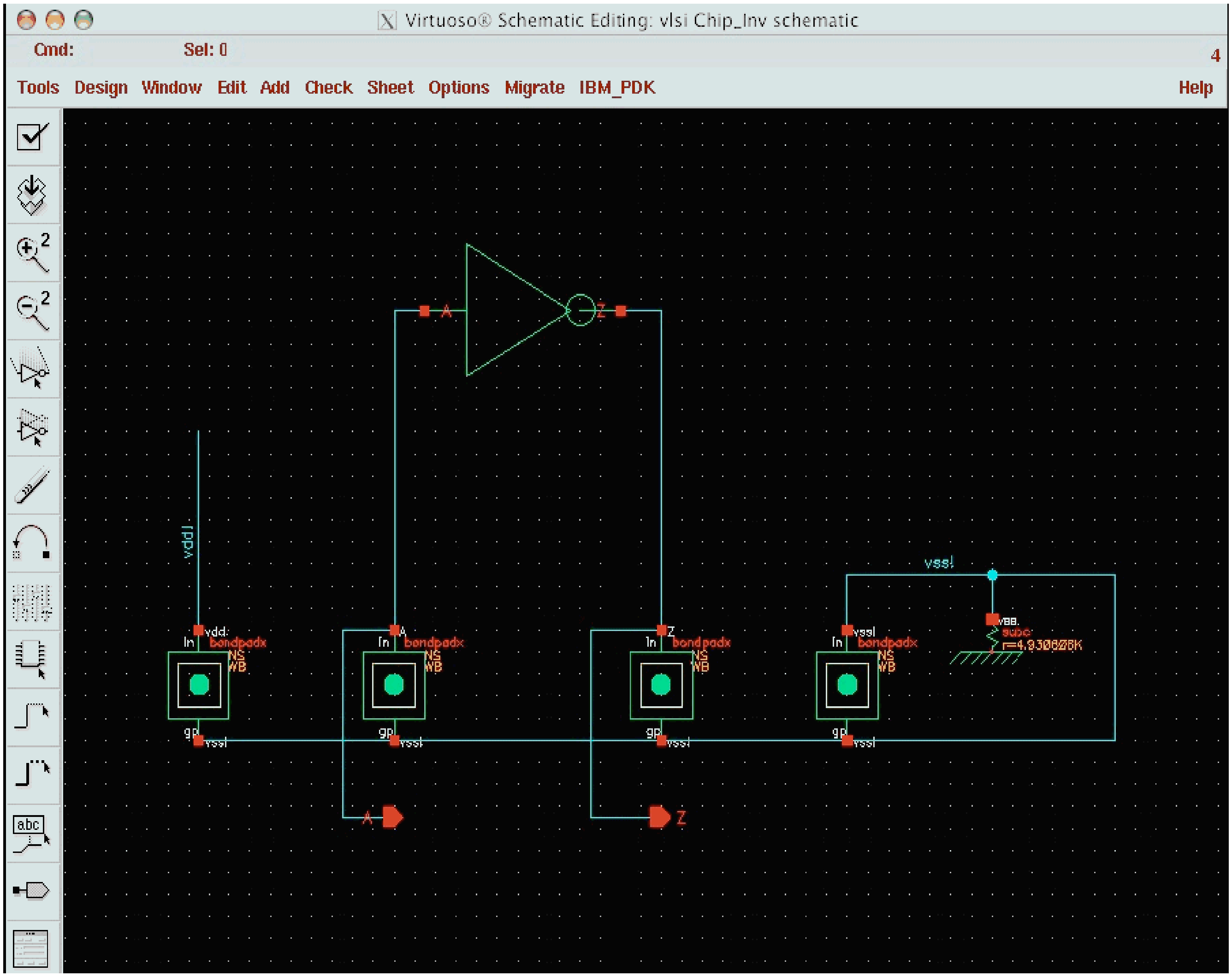


Figure 46- Chip level schematic

Once you have a chip level schematic, you can create a corresponding symbol for the chip. A regular test circuit can be built to simulate your design. After your chip level simulation meets your performance expectation, you can use layout tool to create a layout with chipedge, which is the green box in the following figure. In all IBM documents, IBM uses chipedge as the name. However, in the design library, bicmos7hp, IBM uses "Image" for the name. It is in the "chip" category of the "bicmos7hp" directory. You can specify the size of your chipedge. In the Fig 47, the pads are put at the bottom of the layout. The chipedge covers all the design. Only one side of the pads can be placed close to the edge. There is a 95-um margin on the right and left side of the pads to the chipedge. That margin is to make DRC happy. The substrates of the pads are tied to the ground. In our case, it is vss!. The chipedge needs to be connected to ground also. Therefore, in the example, the vss! is connected to the chipedge directly from the vss! pad (bottom left pad). The substrates of the pads are connected to the chipedge through M1 at the bottom. Since the vss! Pad is connected to the chipedge and to your circuit layout, Cadence will consider that the pad is "prune" from your design and give you a "Device removed because of user's 'prune' request" in LVS log. That mismatch is ok. The pad size and pad distance should match the testing equipment you have in your lab. In our case, you can use 114 um for your pad and 15 um for the distance between pads.

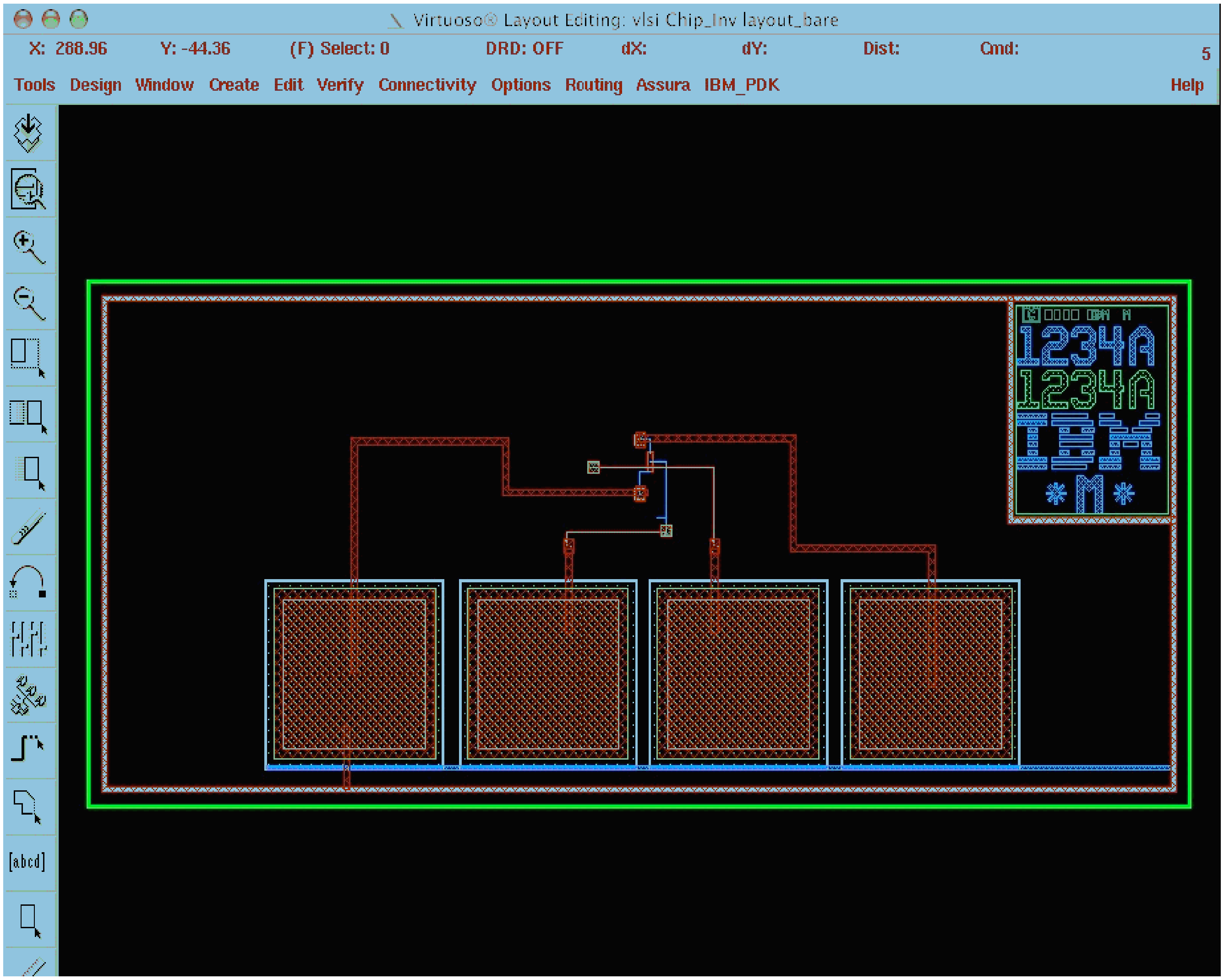


Figure 47- Chip level layout

**After chip level layout**

Ready to send your design? Not yet.
The process to send out a chip to a vender is called a "tapeout," which involves a series of design rule checks. In our current setting, a tapeout has to pass all the design checks above the Assura (not include Assura) as shown in. Every check will check certain things. The CIW window provides feedbacks of the check results.

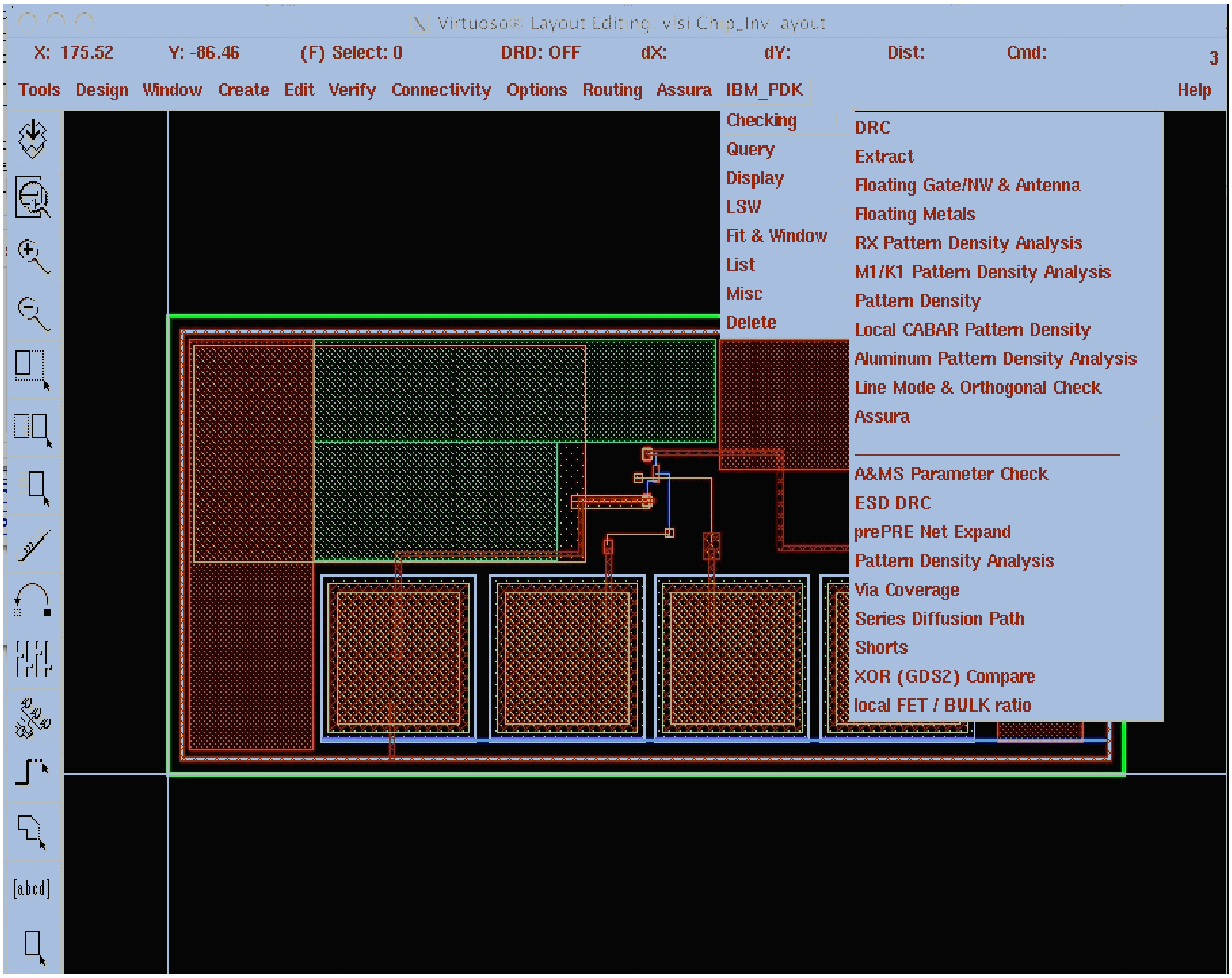

Figure 48- Design rule checks.

Most pattern design checks are checking whether you have enough certain layers in your design. For example, the "Pattern Density Check" checks DV, AM, LY, PC, RX and DT layer in your design. In Figure 49, a pattern density check is done. The CIW window shows the error message as: LY is not enough. At least 23% is expected in your design but your design only has 3.81%.

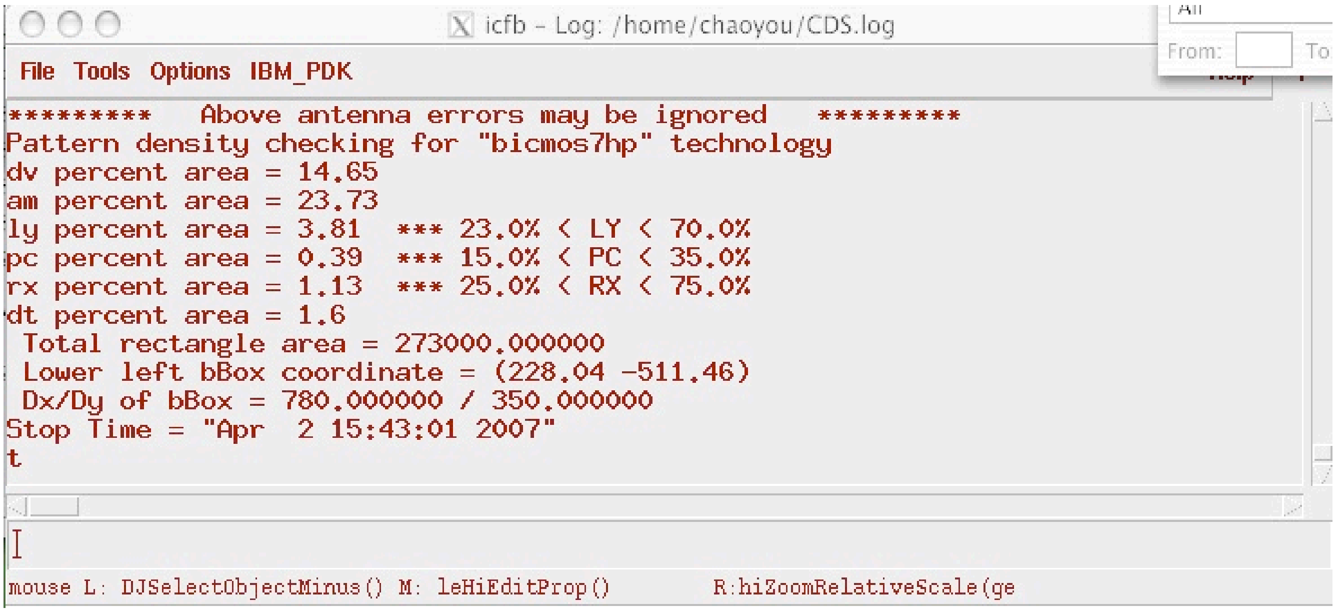

Figure 49- Pattern Design Check

An easy solution for the pattern density check is to add more layers of the same kind to your layout. In the above example, you need more LY, PC and RX layer in your design. The solution is posted in Figure 50, where the red area is RX layer, the green area is PC, and orange area is LY on the top left corner. LY is a top metal layer. PC and RX are bottom layers that used to form PFET and NFET. PC and RX can overlap in this case. LY can overlap any layer as long as it does not create a short.

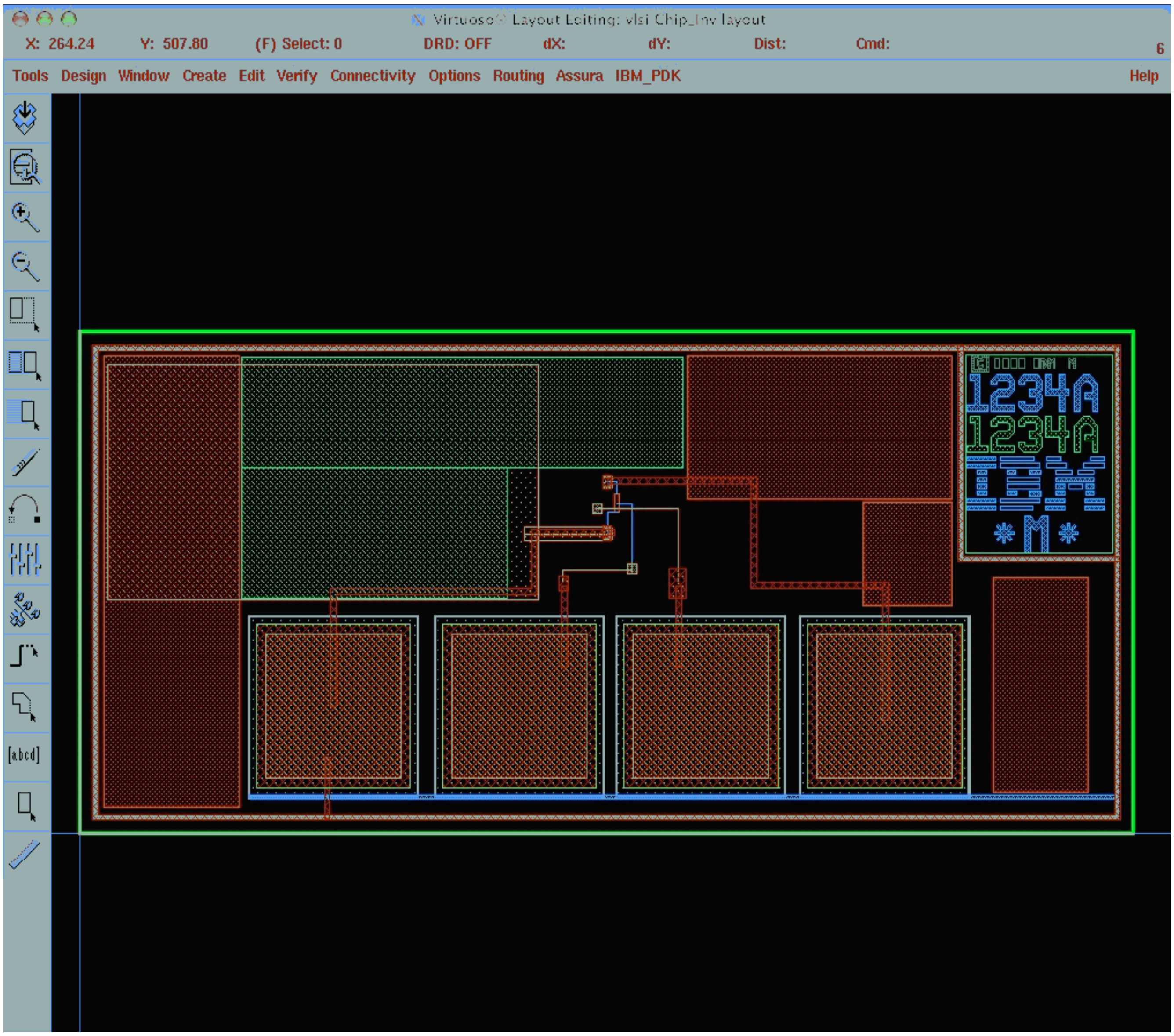


Figure 50-Pattern Density Check Solution

Other pattern design checks look for a density balanced. For example, you cannot have all LY on the left side of your chip. You also need some LY on the right side of your chip. In our example, the LY is balanced since there are LY layers for the power pad connection. Floating gate/metal rules specify that the ratio of the top metal layer to the bottom layer is not too small or too big. In our example, floating gate check makes more troubles. It specifies that all the signals that are connected to AM layer must have an AM/PC ratio between 10 and 150. The ratio is the area of the layers. You can check design rule GR131 for more detail. The solution for this check is to add RX/M1 contact connected to your gate. In Figure 51, a solution is shown. An 8□8 RX/M1 contact is added to connect the M1, which is connected to the gate. Since the gate is connect to the top layer AM, which is a pad. The AM/PC ratio is very small. You need a big RX/M1 contact to correct this check.

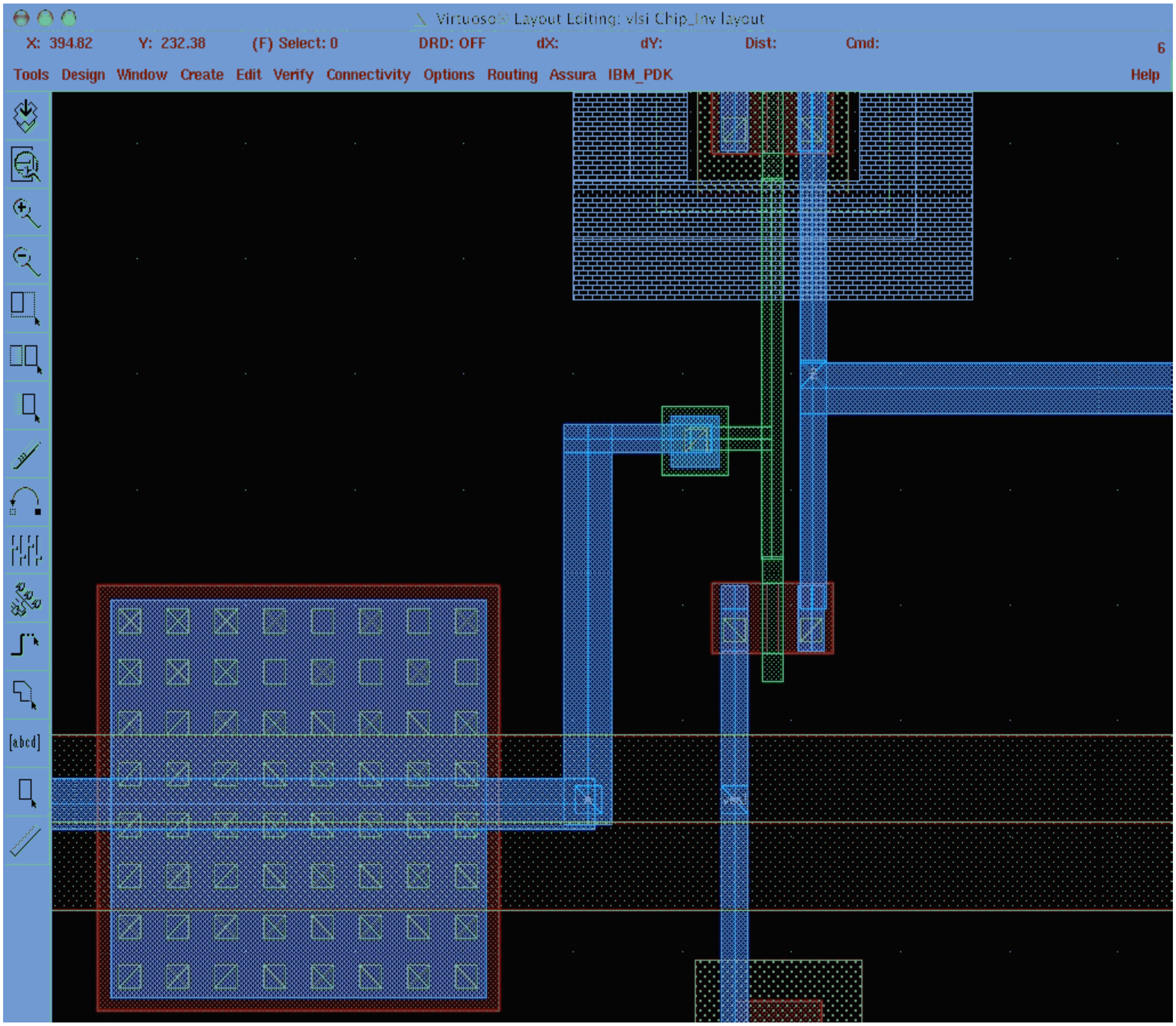

Fig 51- Floating Gate Solution

Every time you made a change in your layout, i.e. adding a layer for pattern density or adding RX/M1 contact for the floating gate. You have to do all checks to make sure everything follows the design rules. Once you think you passed all the checks, you need to backup your log file (CDS.log) in your home directory. All the venders need this log file as a certificate for your design. Now, you are ready to submit. From IBM_PDK ->GDS2/GL1 Translation, specify your library and cell name. The industry standard format is GDSII file. It is a compressed file format contains all the layers' information. You can just email or FTP your gds2 file to your vender for fabrication. Once the vender received your gds2 file, they will use the same form but reverse what you just did. The GDS2 into Cadence option will provide the vender your original layout (just layout, no schematic).

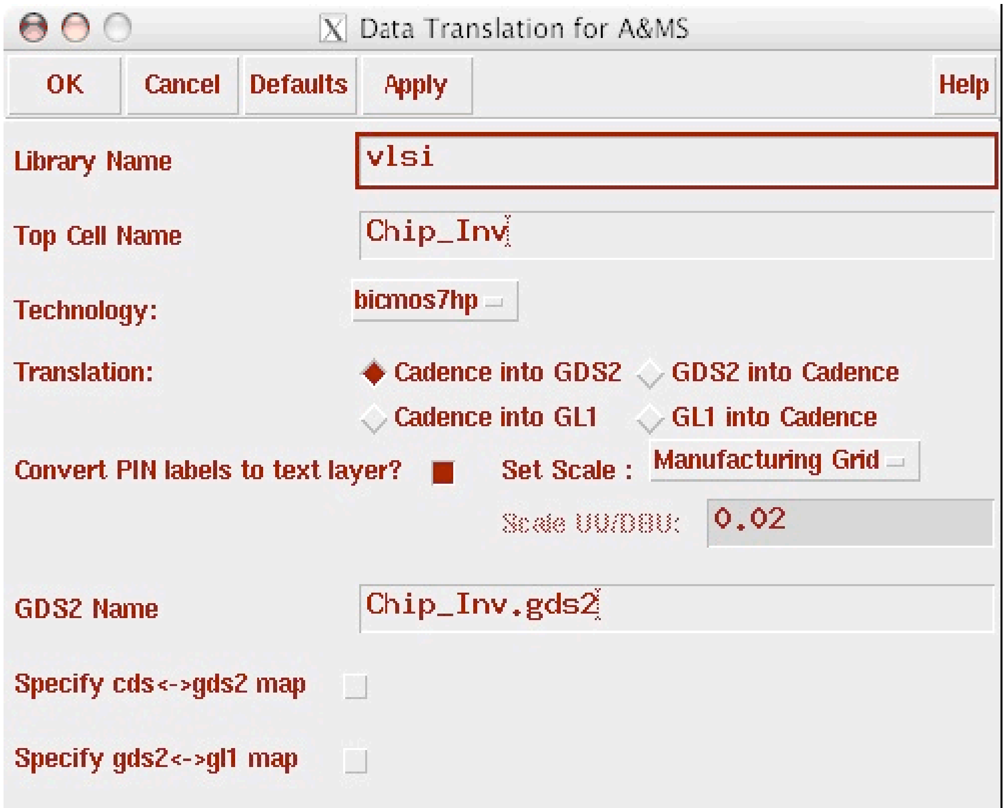

Figure 52-GDSII generation

# Appendix

- **Making screen capture**

To make a better screen capture for your schematic, layout and simulation, you can print to a PostScript (PS) file, open the "ps" file in gimp and save the "ps" file as jpg. For example, in schematic, **Design->Plot->Submit.**

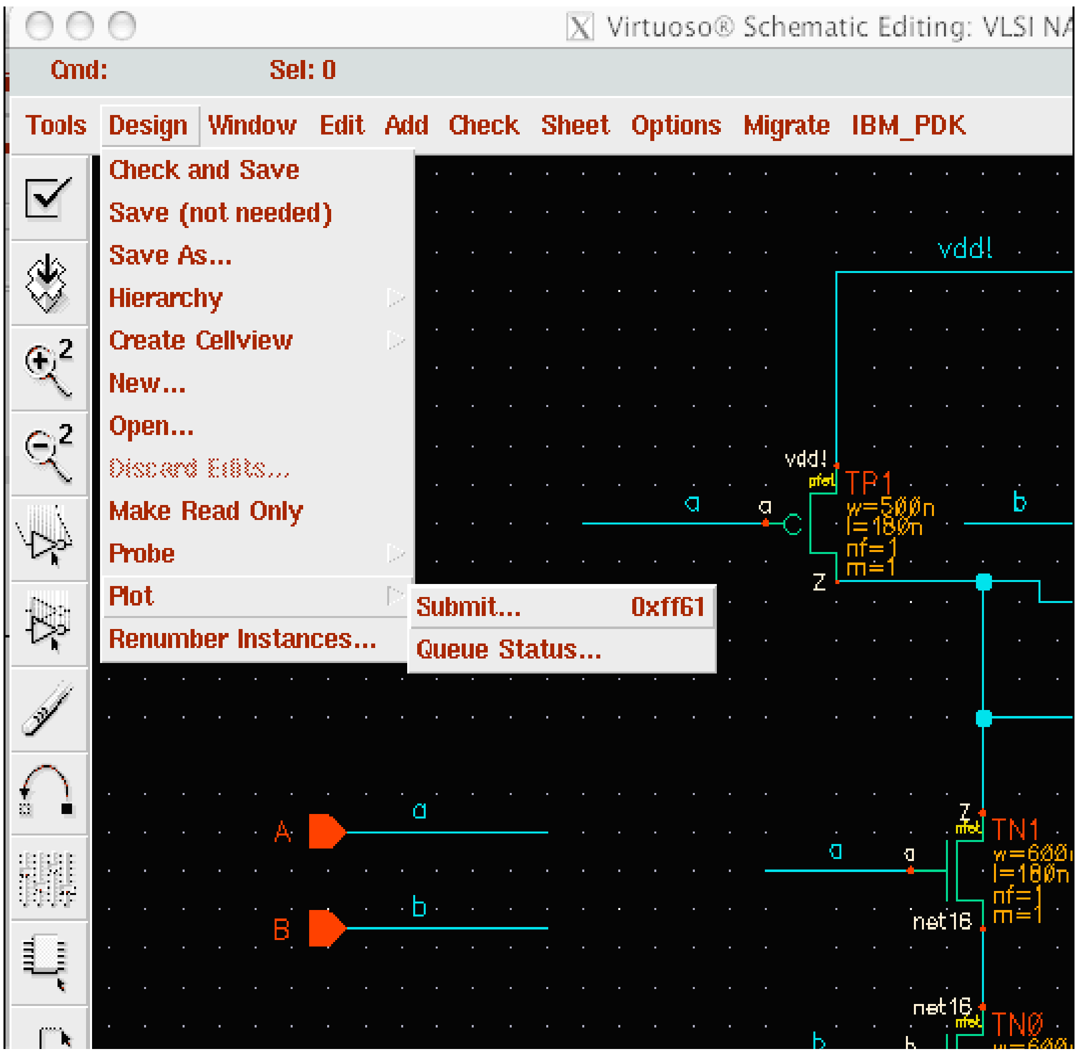


Figure 53

You will get the submit plot window. Click the "**Plot Options**" button, change the local tmp directory to "~/", and send the plot to "**test.ps**".

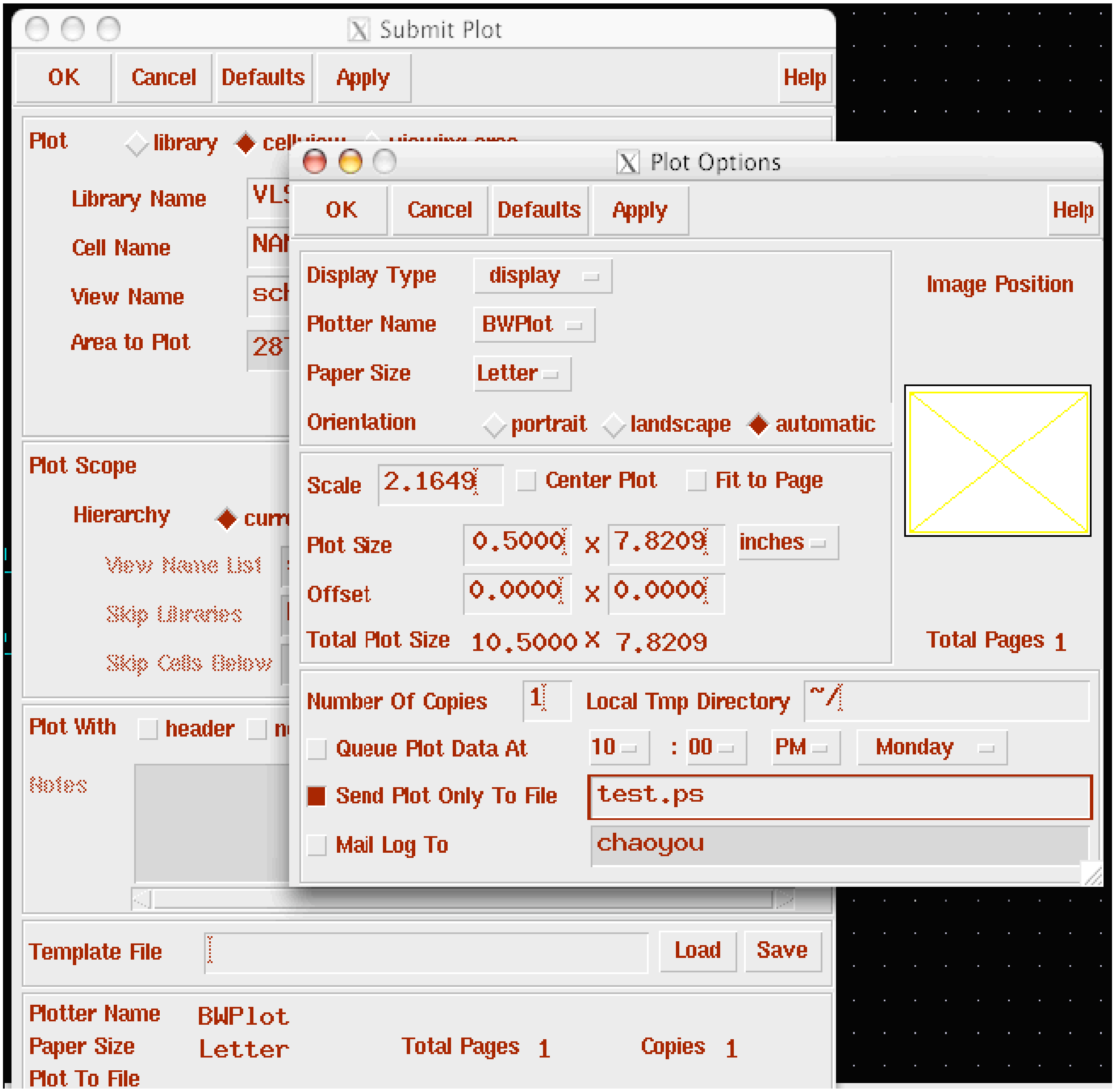


Figure 54

Click **OK** for both "Plot Options" and "Submit Plot" window; you will have a "test.ps" file in your home directory. Open the ps file in gimp (In a terminal window, type "gimp") and save it as a jpg file. In a simulation window, you can use **File->Hardcopy** to get the submit plot window.

- **Restore Cadence Setting**

1. **Cadence settings**

The following script has been placed in the **/cad/cds/script/restore**. The purpose of the script is to restore your Cadence setting in your personal storage. **After you execute the script (type/cad/cds/script/restore in your terminal window), you have to start your icfb from ~/Desktop/Personal_Storage/VLSI, not from your home directory.**

```
#!/bin/bash
#This script copy all the environment files to students personal storage
# by Chao You,
# April 13th, 2007
FILENAME=~/Desktop/Personal_Storage/VLSI
if [ -d $FILENAME ]; then
        echo ==Directory exitis==
        echo ==Copying.........==
else
        mkdir $FILENAME
        echo ==Directory Created==
        echo ==Copying..........==
fi
cp -r /cad/cds/script/* $FILENAME
cp /cad/cds/script/bash.bashrc ~/.bashrc
cp /cad/cds/script/.cds* $FILENAME
source ~/.bashrc
cd ~/Desktop/Personal_Storage/VLSI
```

*FILENAME* sets the path of the Cadence library files and environment settings. The *IF* statement checks if the VLSI library exists. It creates one if the library does not exist. Then the script copies the environment files into your VLSI directory, runs the bash setting, and leaves you in your VLSI directory.

**2. Cadence library**

Cadence library definition file (cds.lib) uses the following format to define a library:

DEFINE_library_name_library_path

In the following example, the cds.lib defines several libraries. For example, it defines the **Share** library in **/cad/cds/Share** directory. It also defines the **bicmos7hp** library in **/cad/cds/IBM_PDK/bicmos7hp/rel/cdslib/bicmos7hp**.

```
DEFINE analogLib $IC_INST_DIR/tools/dfII/etc/cdslib/artist/analogLib
DEFINE basic $IC_INST_DIR/tools/dfII/etc/cdslib/basic
DEFINE sample $IC_INST_DIR/tools/dfII/samples/cdslib/sample
DEFINE US_8ths $IC_INST_DIR/tools/dfII/etc/cdslib/sheets/US_8ths
DEFINE bicmos7hp
/cad/cds/IBM_PDK/bicmos7hp/rel/cdslib/bicmos7hp
DEFINE Share /cad/cds/Share
DEFINE VLSI ~/Cadence
```

If you have a library name other than VLSI, you can change your cds.lib accordingly.

## Reference


- R.Y.Dinakar, B.S.Goda, J.Mayega, “Cadence Design System Tutorial”, ECSE 4220: VSLI Design, Rensselaer Polytechnic Institute, 4 November 2003.