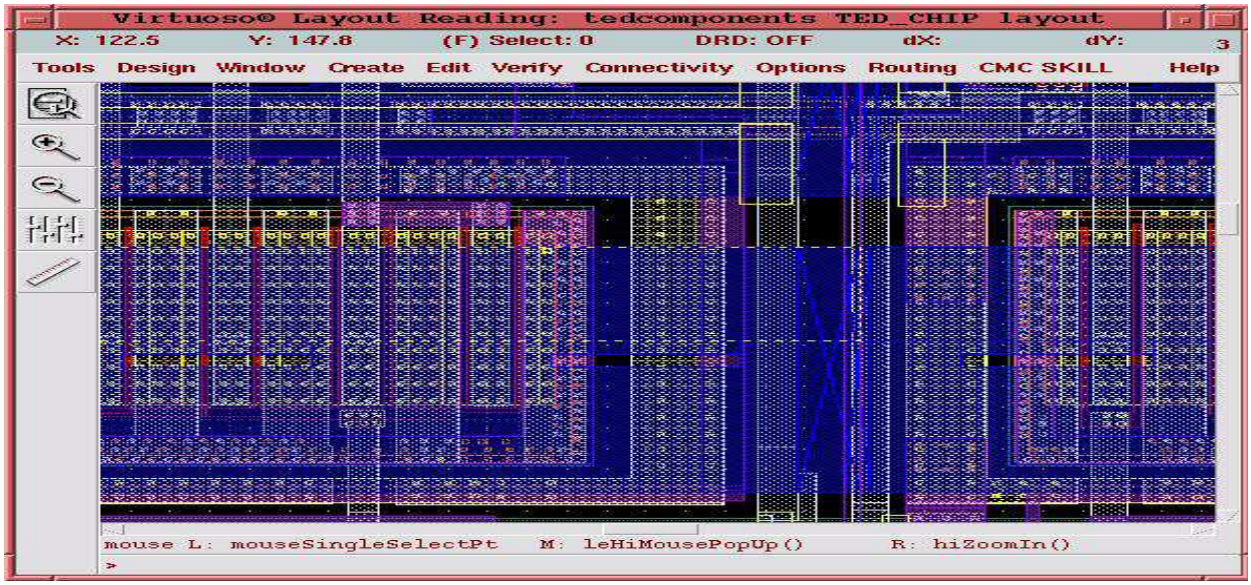
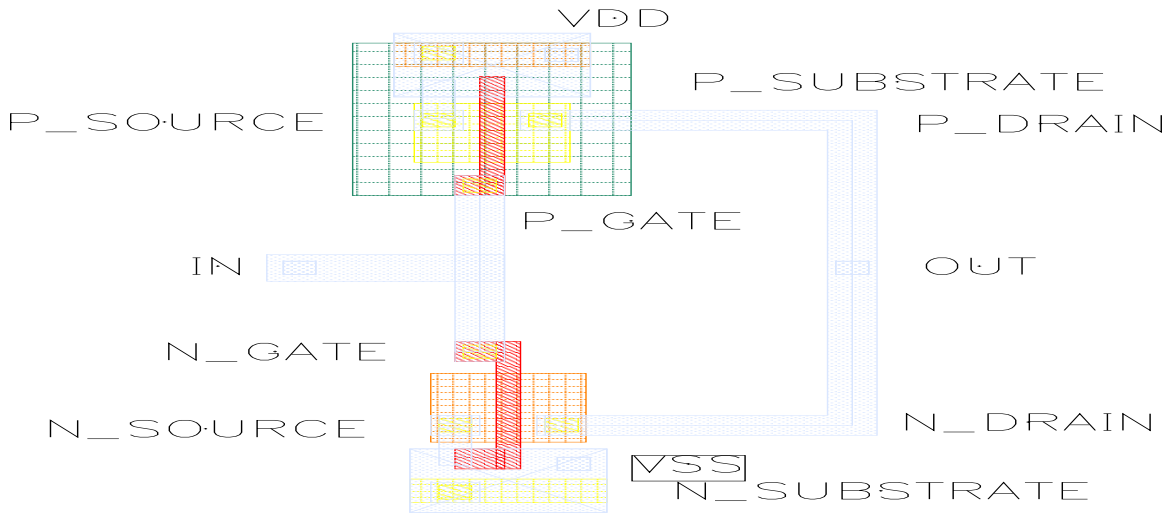
A Tutorial on Using the Cadence® Virtuoso Editor to create a CMOS Inverter with CMOSIS5 Technology





Developed by
Ted Obuchowicz
VLSI/CAD Specialist, Dept. of Electrical and Computer Engineering
Concordia University

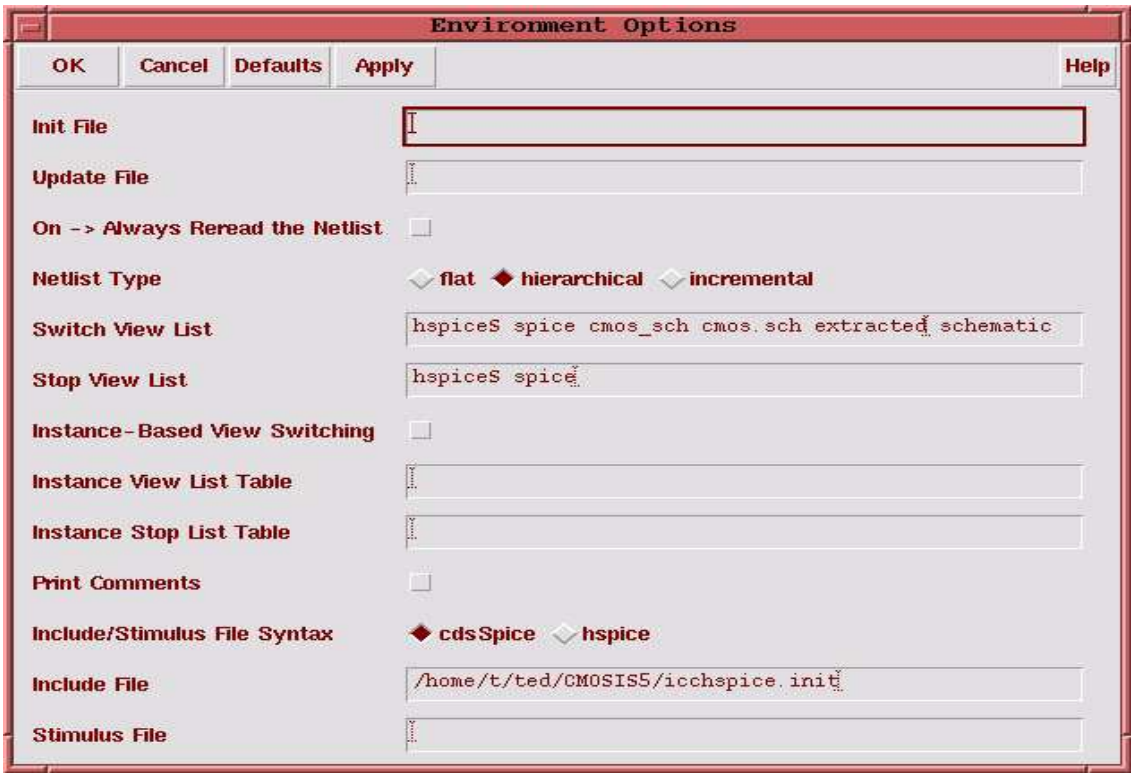
Feb.23, 1998
Revised: May 2006

Revision History

October 26, 2000: updated the procedure to fill out the Extractor form for Cadence 2000a.

Sept. 5, 2003: updated procedure for using non-global power supply pins (VDD and VSS instead of global power supply pins (VDD! and VSS!) since the HSPICE netlister in Cadence 2002a no longer extracts the global pins leading to “no dc path from node XXX to ground” error messages”. Consequently, any schematic which makes use of an extracted symbol view of a layout will have to make explicit connections to a power supply source and the VDD and VSS pins which will now appear in the symbol of the extracted layout.

Sept. 27, 2004: updated procedure on page 16 for simulating an extracted view of a layout since the extracted keyword no longer appears by default in the Switch View List of the Setup -> Environment form. Users MUST ensure that extracted appears before schematic in the Switch View List or else the simulation of the extracted layout will fail. For Cadence 2003a, the Environment form appears as:



May, 2006: updated tutorial to include color Postscript images. Updated images and text to reflect Cadence 2004a.

INTRODUCTION

This tutorial is an introduction to the Layout Editor available from the Cadence design tools and the CMOSIS5 design kit from the Canadian Microelectronics Corporation (CMC). This tutorial is based on the current version of Cadence (2004a). The CMOSIS5 design kit is based on the Hewlett-Packard CMOS14TB process. This is a high-speed, high density 0.5 micron CMOS process which feature a 0.6 micron drawn gate length optimized for 3.3 V operation. The CMOS14TB process is a triple-metal, single poly CMOS process.

I: USING THE VIRTUOSO LAYOUT EDITOR TO CREATE A PMOS TRANSISTOR

This section will explore the use of the Virtuoso Layout editor. A p-type MOS transistor will be designed. A p-type MOSFET transistor is fabricated with the CMOS14TB process by crossing polysilicon and N-Island in a P-Substrate.

1-1: Start the Cadence tools by typing the following command from the UNIX prompt:

% **cmosis5**

The main CIW (Command Interpreter Window) will appear. The next step is to create a new library to hold your work. The library created must be attached to a specific technology file, in this case the CMOSIS5 design kit from the Canadian Microelectronics Corporation.

1-2: To create a new library and attach it to the CMOSIS5 technology file select: **File -> New -> Library**. The New Library window will appear. In this window fill in the following: Name: mylib (or any other suitable name), Technology File: **select the Attach to an existing techfile** button. Select **OK** at the top of the New Library window by left clicking with the mouse. Once you have done this, a new window will appear. This is the Attach Design Library to Technology File as shown in Figure 1 below:

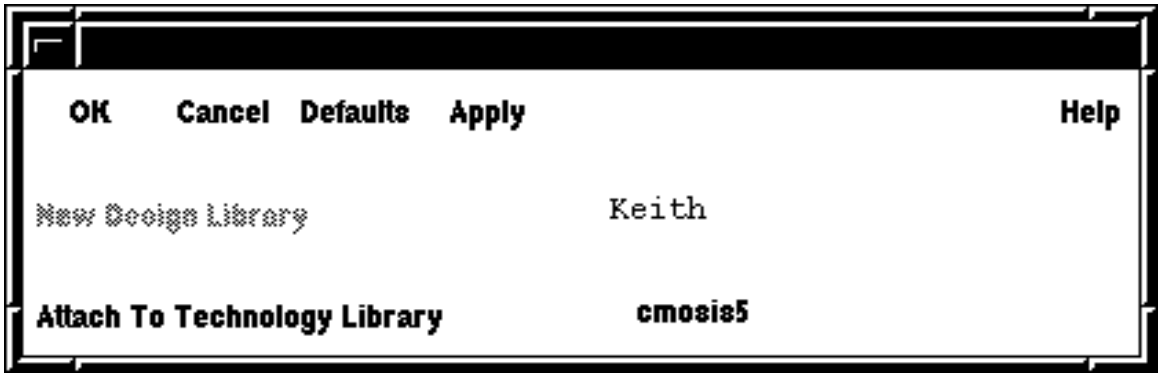


Figure 1: Attach Design Library to Technology File window.

From this window change to default Library from PIC to cmosis5 by left clicking on the button labelled PIC. A pop-up submenu will appear listing the possible choices of technology files. Select cmosis5 by left clicking with the mouse.

1-3: Create a new cell to hold the layout of your pmos transistor. From the main CIW window, select **File -> New -> Cell view**. In the Create New file window fill in the following: **Library Name**: click on this button to select from a list of possible libraries the library you created in the above steps (for example mylib), **Cell Name**: enter the name of your cell, for example **pmos**, **View Name**: enter **layout**, **Tool**: select **Virtuoso-Schematic**. Once you have entered all the values, click on the **OK** button. **NOTE: view names should be entered in lower-case letters, for example: schematic, symbol, layout are valid view names. The system will accept upper case letters in view names, however not all menu items will be selectable. For view names corresponding to schematic, symbol, and layout ALWAYS USE LOWER CASE LETTERS ONLY. Note also cell names should not begin with a digit.**

1-4: The Virtuoso Editing window will appear (see Figure 2). It is in this window that one creates the various layers (polysilicon, N-Well, etc) that make up a design. The Layer Select Window (LSW) will also appear. The LSW window is used to select the particular layer to work with.We will begin the layout of the pmos transistor by creating a rectangle of polysilicon of 0.6 micron by 7 micron. Note: the design in this tutorial is not a minimum sized transistor and its W/L ration is not 1. Note further that although the CMOS14TB process is a 0.5 micron process, the drawn minimum size is 0.6 micron.

From the LSW window, make sure that all the layers are valid and selectable. Left click on the AV and AS buttons. This is a convenient method of selecting which layers are to be displayed in the editing window. For example, by first making all the layers invalid (by clicking NV) and then selecting the particular layer(s) from the list in the window, and then redrawing the layout (select Design -> Refresh from the Virtuoso Editing window) only those layer(s) which you have selected will be drawn.

We will be laying a polysilicon rectangle of 0.6 micron by 7.0 micron. From the LSW, select the poly dg layer by left clicking with the mouse on this choice from the list of layers. Notice how the current active layer is displayed in the top portion of the LSW window.

From the Virtuoso Editing window select **Create -> Rectangle**. You will noticed that attached to the cursor (represented by an arrow) is a small yellow square. Move the arrow to where you want one corner of the rectangle to be located and left click. Next, move the arrow to the OPPOSITE corner of the rectangle you wish to define and left click. A rectangle filled with a red pattern will be drawn. If you wish to create another polysilicon rectangle, you may move the arrow to the new location and repeat the process. To stop creating polysilicon rectangles, press the ESC key. In general. when working in the Virtuoso Editing window, the ESC key is used to terminate a command. You can use the grid dots to determine the approximate size of the rectangle. By default, the cellview window shows a grid of dots. There are two types of grids: the “minor” and the “major” grid. Minor grid points are displayed in white and appear at every micron, the major grid points appear at every 5 microns. To choose different values for the grid points select **Design -> Options -> Display** from the Virtuoso Editing window, the Display Options window (Figure 3)

will appear. In the Grid Controls section fill in the desired units between the minor and major grid points. The X Snap Spacing and Y Snap Spacing do not apply to the visible grid, they control how the cursor snaps to the drawing grid. For example, if you set the minor grid to 1 micron and

Figure 2a: The Layer Select Window.

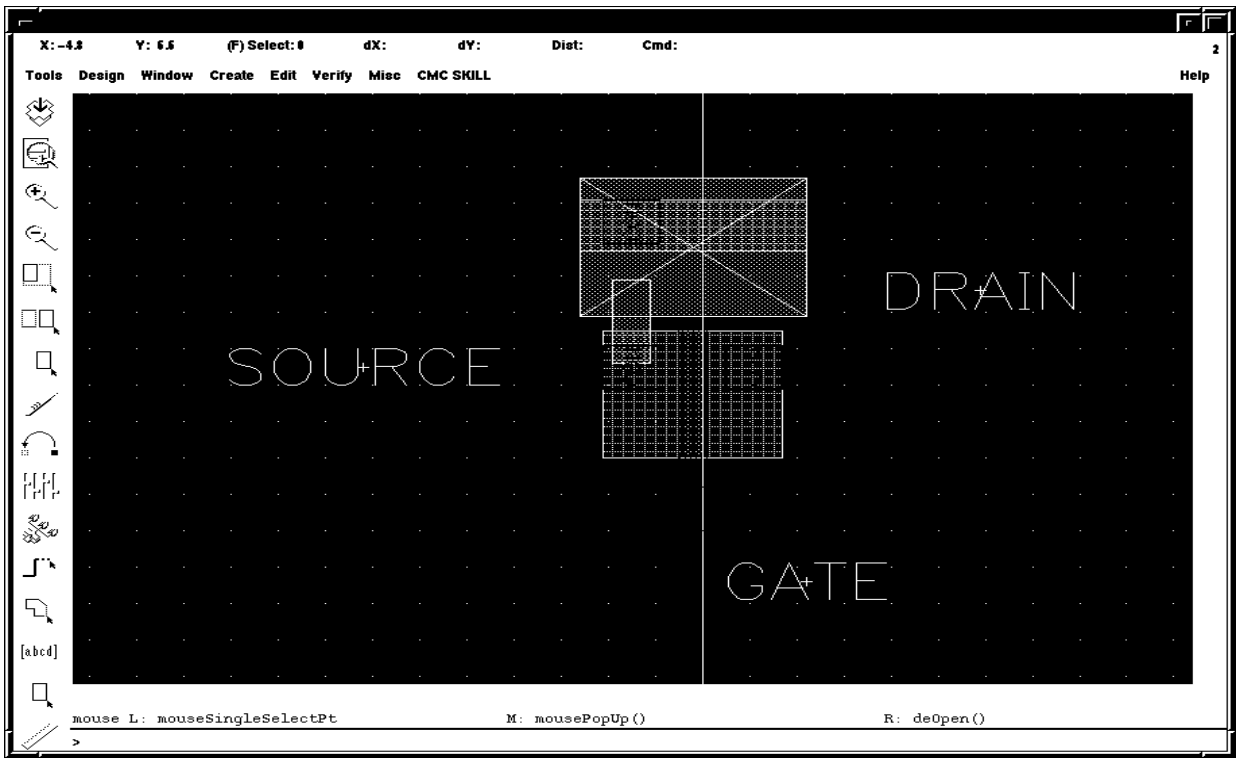


Figure 2b: The Virtuoso- Editing Window.

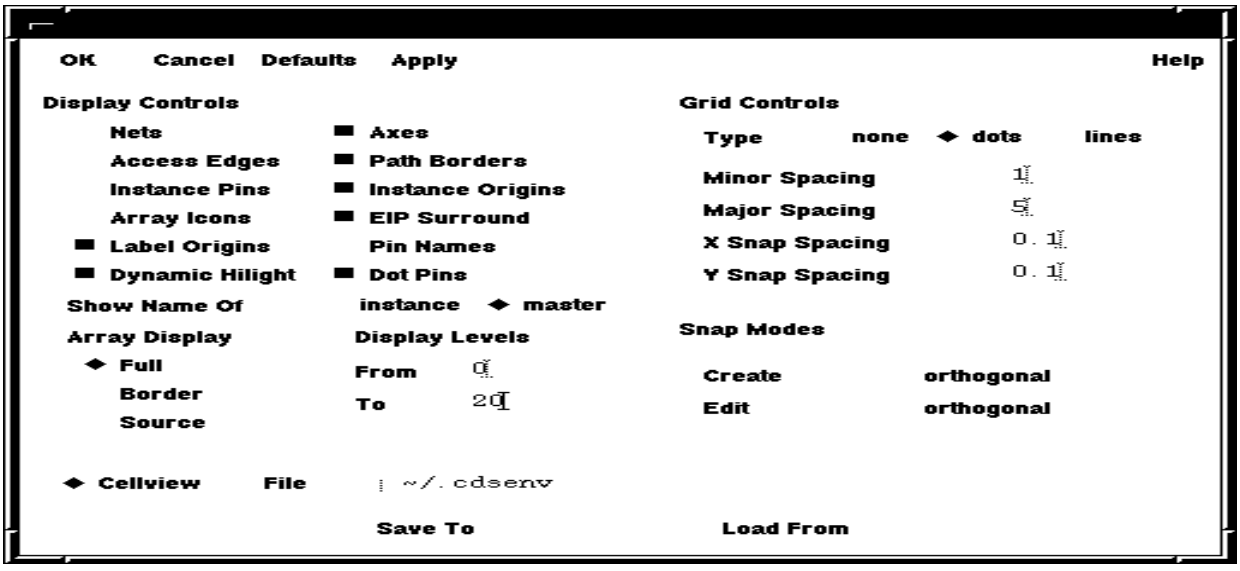


Figure 3: The Display Options Window.

you want to draw objects at 0.5 micron intervals, you would set the X and Y snap spacings at 0.5 microns. The X and Y Snap Spacings define the distance the cursor can move along the x and y axes respectively.

Another method to alter the size of a drawn object is to use the edit feature and to manually change the coordinates of the object. For instance, you first select the object you wish to edit by moving the cursor to it and left clicking the mouse. The object you have selected will become outlined in a white color . Next, select **Edit** from the menu choices in the Virtuoso Editing window. From the popup menu which will appear, select **Properties**. The Edit Properties window will appear listing the selected object's properties (see Figure 4). In the case of a rectangular layer being selected, the type of layer, and the X-Y coordinates of the rectangle's lower left and top right corners will be given. You can change these values to suit your needs. Select **OK** from the Edit properties window to make the changes. Note that the Edit choice from the Virtuoso Editing window has additional choices: **Undo, Redo, Move, Copy, Stretch**, etc. . Experiment with these features. Remember to enter ESC after you have selected and object and performed an Edit operation on it. The selected object stays selected until the ESC key is entered.

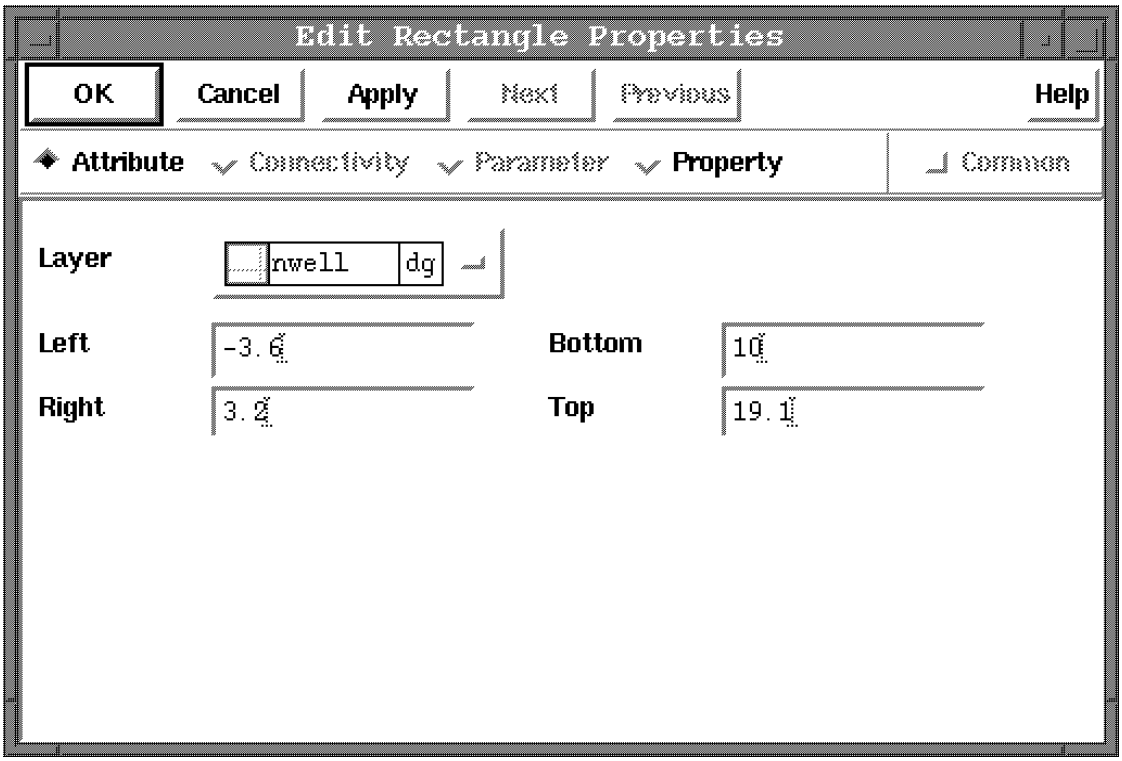


Figure 4: The Edit Rectangle Properties window.

Your polysilicon rectangle (which will eventually become the gate terminal of the pmos transistor should resemble the one shown in Figure 5.

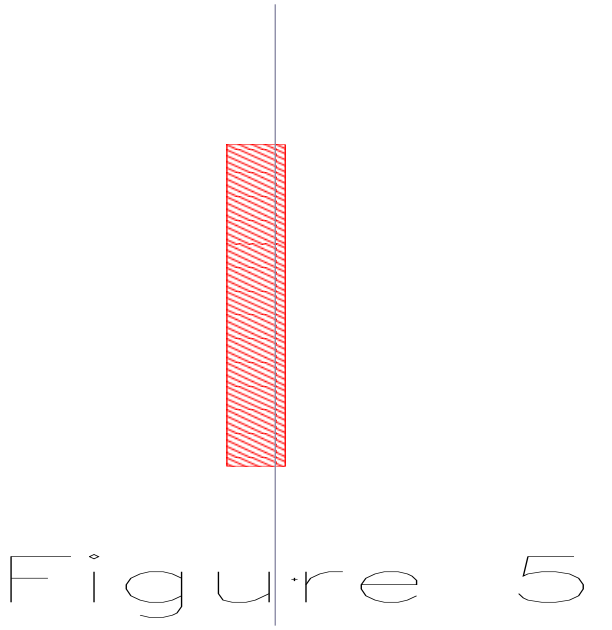


Figure 5.

1-5: The next step is to overlap the polysilicon with a rectangle of pisland. Select from the LSW window the **pisland dg** layer and create a rectangle of approximately 4 x 4 microns. This rectangle should overlap the polysilicon as shown in Figure 6.

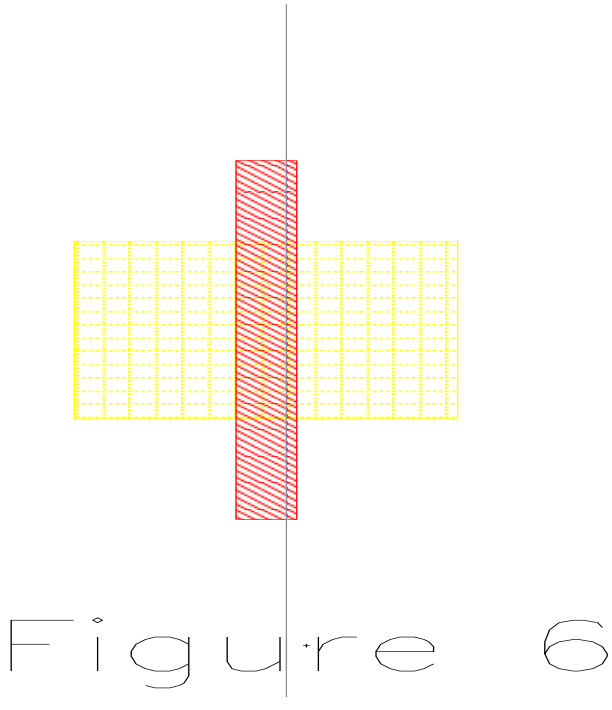


Figure 6.

1-6: Next, place an N-well around the transistor. Select the **nwell dg** layer from the LSW. Create a rectangle of approximately 9.0 x 7.0 microns and place it over the layers you have already drawn. Note: it may be necessary to slightly alter these dimensions if a Design Rule Check (DRC) reports any design rule violations, more on this subject later. Refer to Figure 7 for details concerning the placement of the N-well.

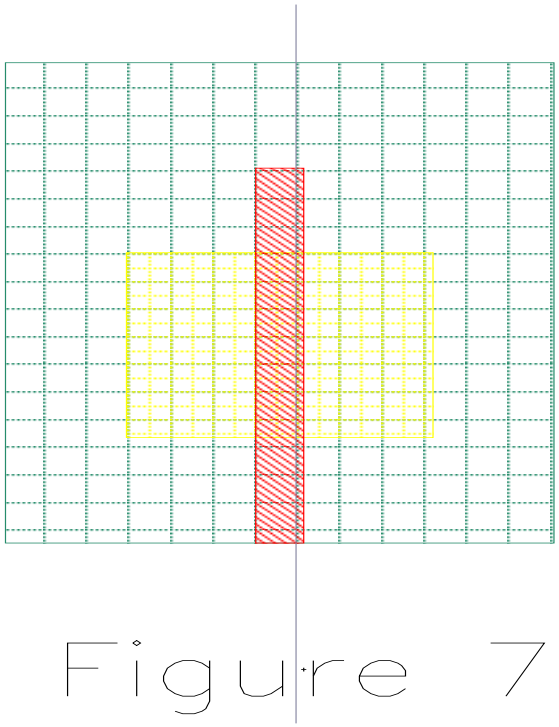


Figure 7.

1-7: Next, we will place a contact at one end of the polysilicon gate. This will make an electrical connection to the gate terminal of the transistor. Contacts may be created manually by first selecting **contact dg** layer (yellow color) and creating a square of size 0.8 micron x 0.8 micron. The design rules state that contacts must be covered by a **metal1** layer. After the contact has been placed, cover it a layer of **metal1**. A more easier method of creating contacts is to use the predefined contact from the hcells library, and to place an instance of this cell. This predefined contact is of the proper size (0.8 micron x 0.8 micron) and is already packaged with a metal1 layer. To obtain an instance of this predefined cell, select **Create -> Instance** from the Virtuoso Editing window. In the Create Instance form which will appear, enter **hcells** in the Library field, **CON** as the Cell name, and **layout** as the View name. A predefined contact will appear attached to your cursor,.place it at one end of the polysilicon gate. Press ESC to stop creating instances of this cell. Refer to Figure 8 for details. When you place the contact, it will appear as a red square with the letters Con in the square. You can toggle between different views of these predefined cells by using the **CTRL-F** and **SHIFT-F** combinations. SHIFT-F will show the layers making up the predefined cell, in this case a yellow contact dg layer and a white metal1 dg overlapping the yellow

contact. CTRL-F will replace this view with the red symbolic representation for the contact.

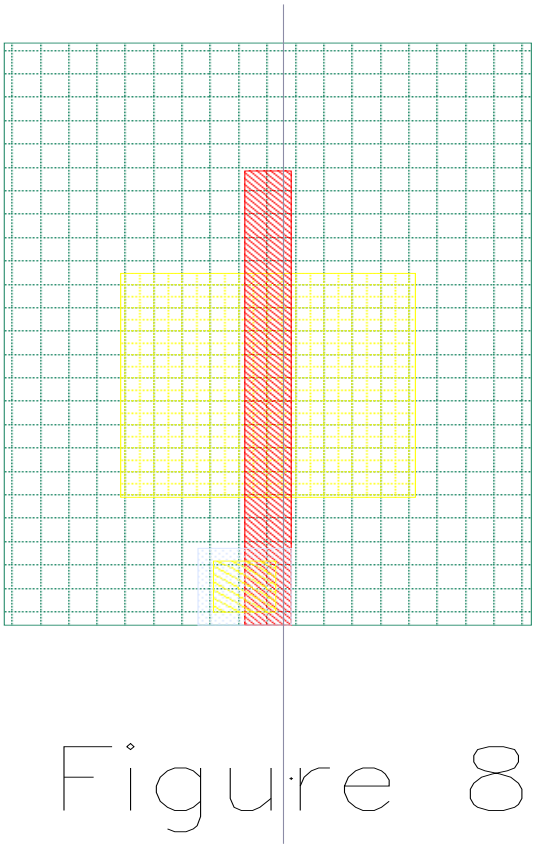


Figure 8.

1-8: We shall perform a DRC Check. It is a good practice to perform DRC checks often when designing a layout. This will make the task of fixing any errors easier. To run a DRC check select, **Verify -> DRC** from the Virtuoso Editing window. The DRC window will appear (Figure 9). Select OK. Reported in the CIW window will be any violations of design rules (if any). The design rule violation we are concerned with is the one reported as :

1 8H: contact must be on island or poly

There may be other errors reported, ignore these for the time being as the layout is not yet complete. To correct the error, we must place polysilicon around the contact and its metal1 overlap. A quick method of locating the errors is to use the **Verify -> Markers -> Find** choice from the Virtuoso-Editing window. In the Find Marker form, if you select **Zoom to Markers**, the error will be reported with a marker symbol and an additional window (see Figure 10) explaining the reason for the error will appear.

Note: for very large designs the DRC will execute faster if you set the Echo Commands button off in the DRC window. The default setting is ON which will result in each rule name being displayed in the CIW window which will slow down the execution of the rule checking.

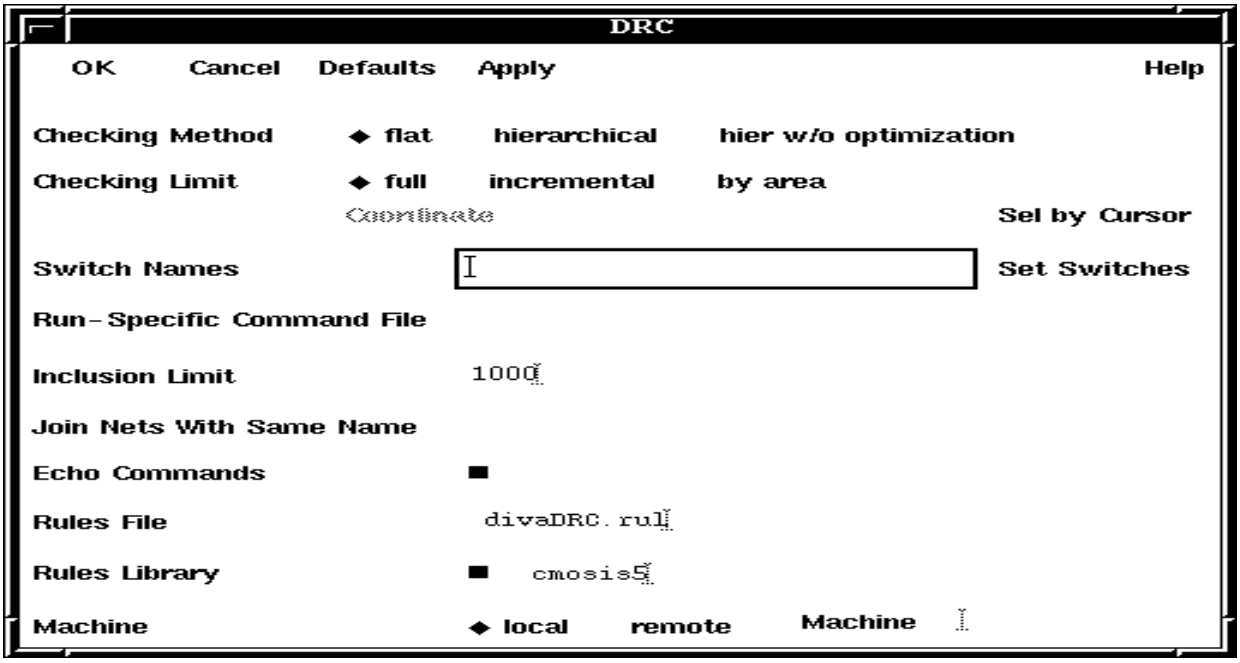


Figure 9: DRC window.

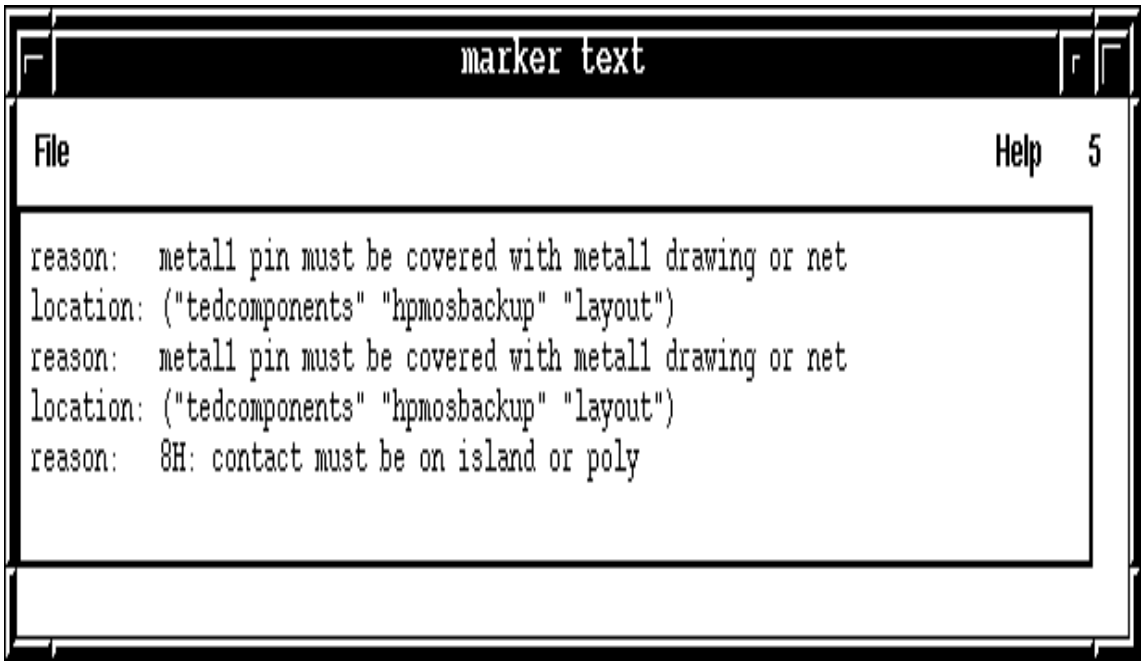


Figure 10: Marker Text window.

1-9: Next, we place two more contacts. In a CMOS transistor the source and drain terminals are identical. It is not until the source to substrate connection is established that the distinction between the source and drain terminals is made. Place two more **CON** cells from the hcells library using the **Create -> Instance** menu. These two contacts should be placed in the yellow pisland. Refer to Figure 11 for details.

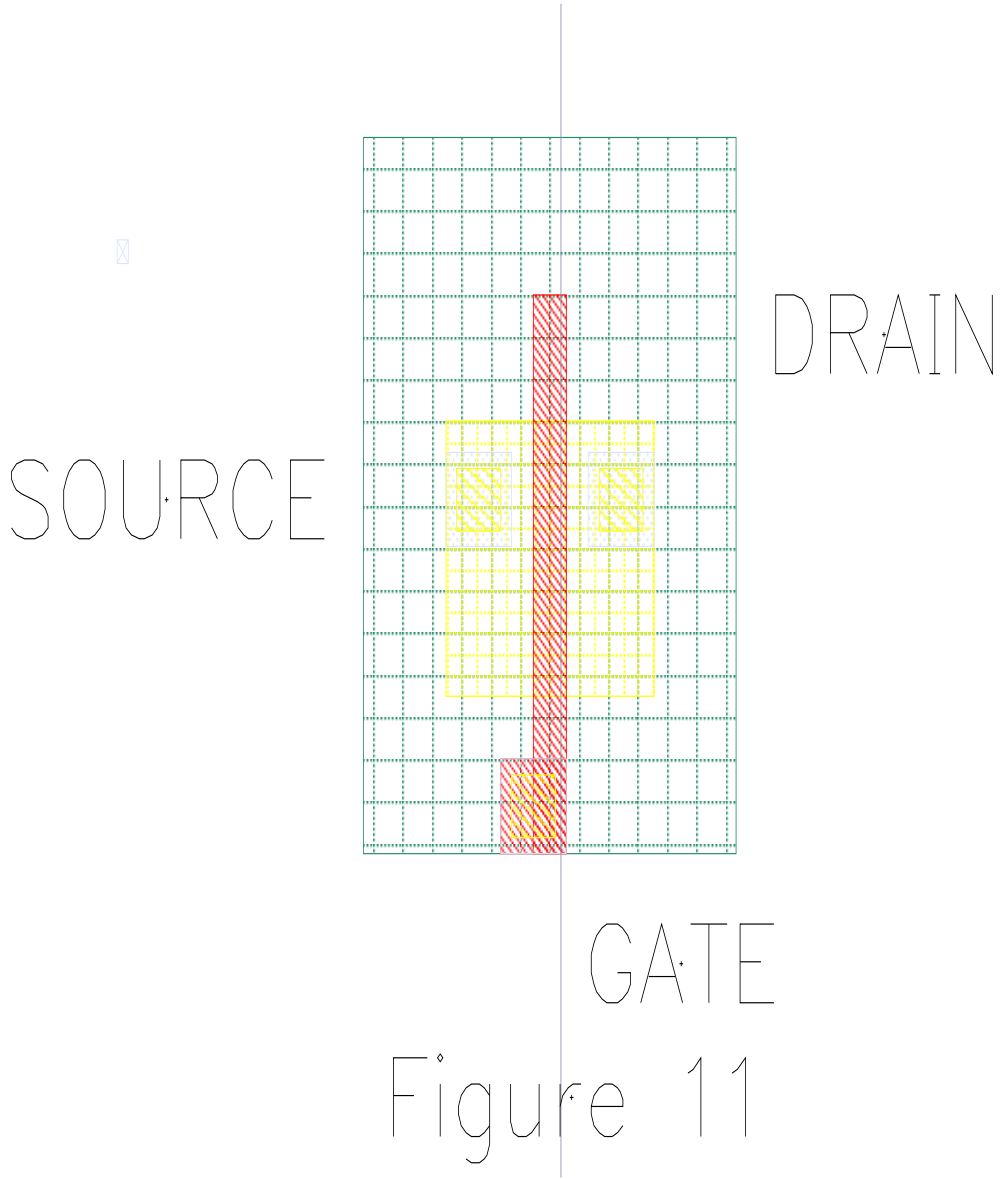


Figure 11.

1-10: We are now ready to make the source-substrate connection. In a p-type of MOSFET, the substrate consists of n-type material. This is the nisland in the CMOSIS5 technology. Select the **nisland dg** layer from the LSW window, and create a rectangle with this layer. Place the nisland rectangle with one edge adjacent to the green nwell of the transistor. Refer to Figure 12.

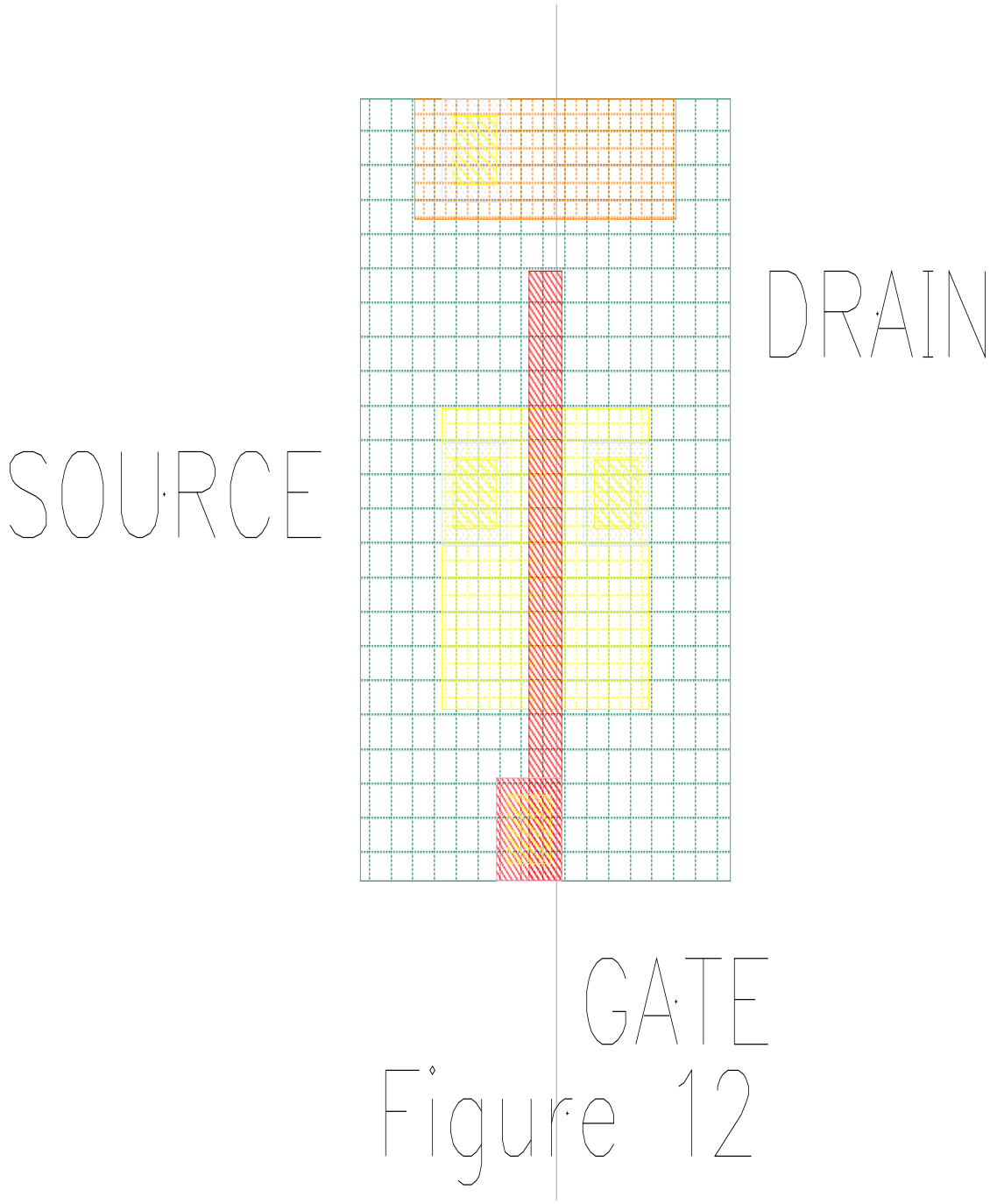


Figure 12.

1-11: Place a contact at one end of the nisland rectangle. Eventually, we will place a metal1 layer connecting this contact to one of the contacts placed in the pisland. This will establish the source-substrate connection, thus distinguishing the source terminal from the drain terminal. Refer to Figure 13.

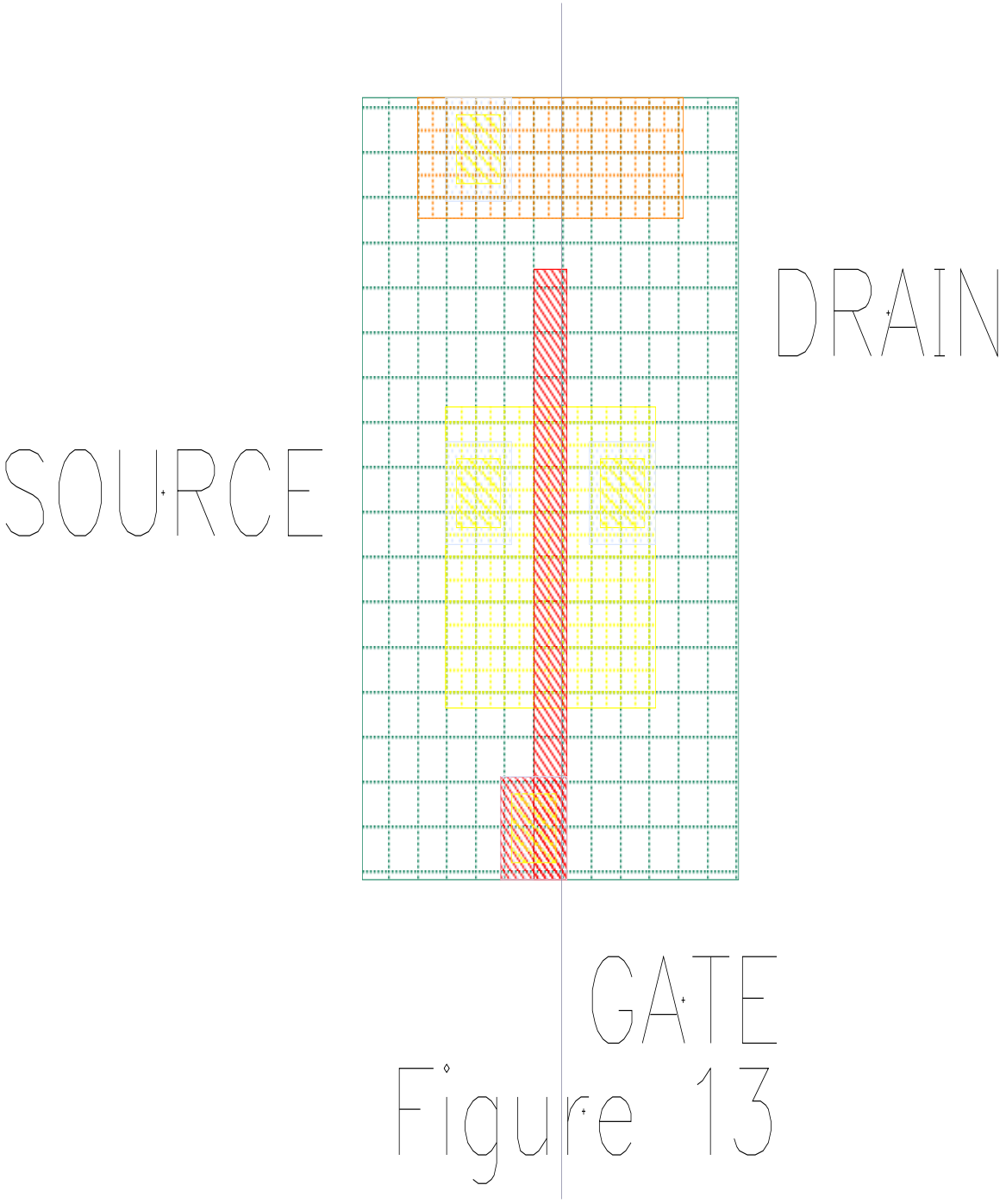


Figure 13.

1-12. Connect the contact you have just placed in the nisland rectangle with one of the contacts in the pisland. Select **metal1 dg** from the LSW window. Create and place a rectangle connecting the two contacts. See Figure 14.

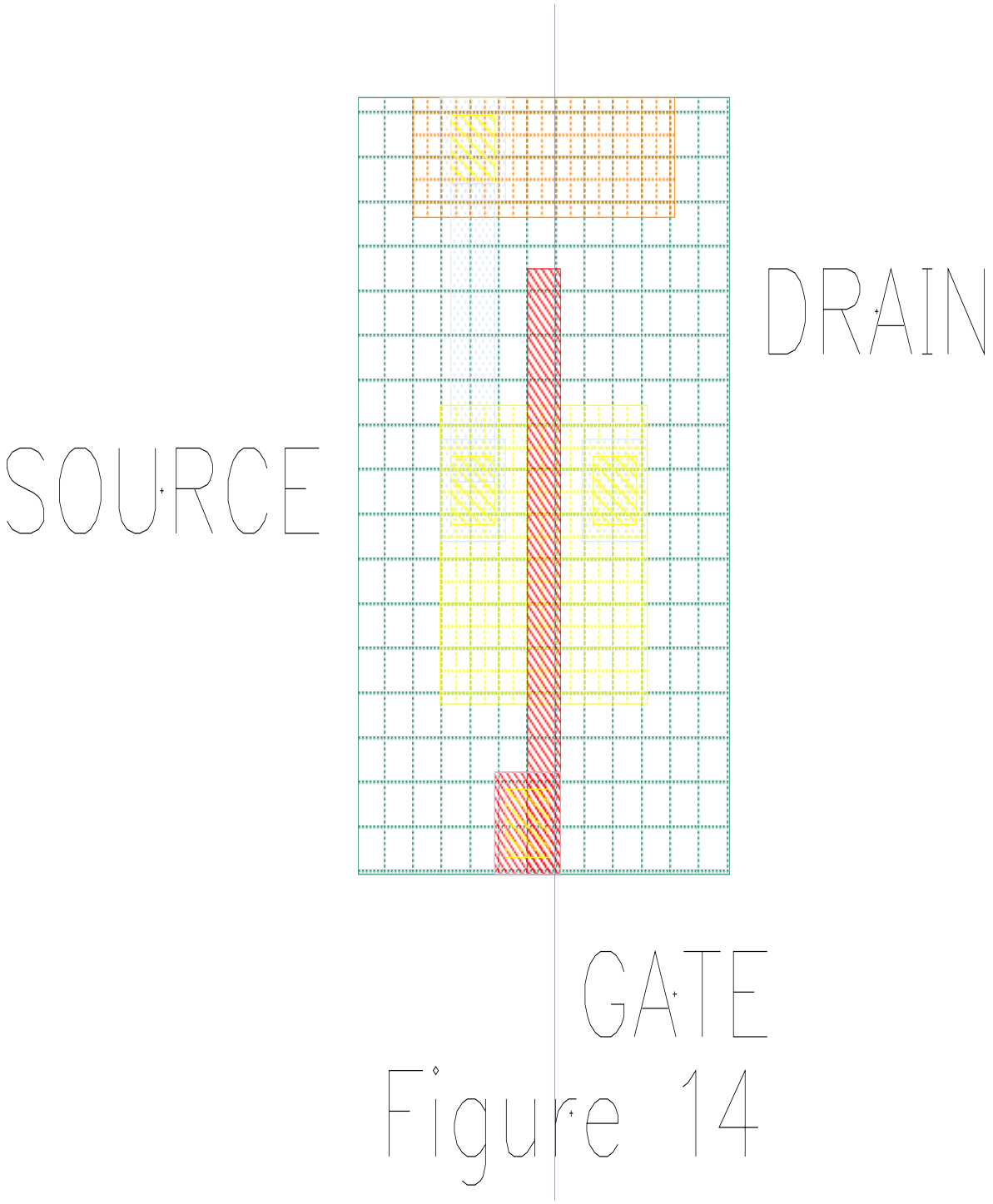


Figure 14.

1-13: This completes the layout of the PMOS transistor. Save your work by selecting **Design -> Save**. As a final check, perform a design rule check. There should be 0 errors reported in the CIW window.

II - USING THE VIRTUOSO LAYOUT EDITOR TO CREATE A NMOS TRANSISTOR

A N-type MOSFET is formed by crossing polysilicon and N-Island in a P-substrate. The following steps outline the procedure of forming a NMOS transistor using the CMOSIS5 design kit.

2-1: From the CIW window, create a new cell called nmos, with View type set to layout. You can use the same Library name as the name you chose to hold your pmos cell. From the CIW window, select File -> New -> Cellview and fill out the Create New File form window (Figure 15).

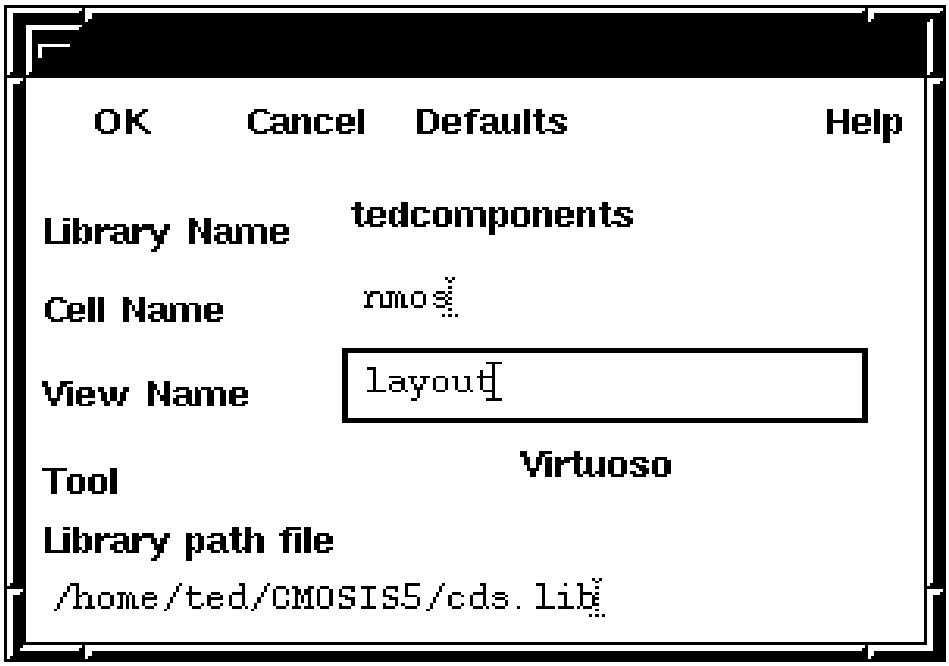


Figure 15: Create New File window.

2-2: A Virtuoso Editing window will appear. Select the poly dg layer from the Layer Select window and draw a rectangle 0.6 microns X 8.0 microns. Refer to Figure 16 for the details of this; note that in this figure there is a small square of polysilicon. Although it is not necessary to draw it at this point in time, it will be necessary at a later point when a contact is added at this end to define the gate terminal.

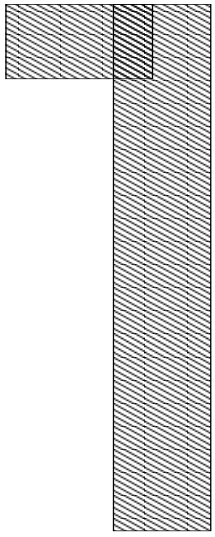


Figure 16.

2-3: Create a rectangle of nisland dg. The intersection of the poly dg and the nisland dg will form the basis of out nmos transistor. Refer to Figure 17 for the details. The nisland dg in this figure is approximately 3.80 microns X 4.10 microns.

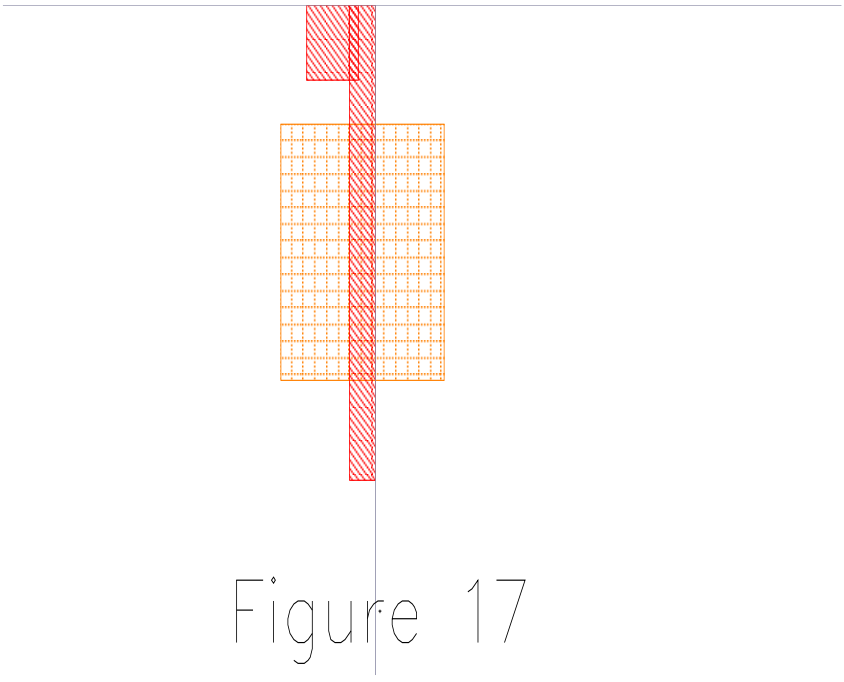


Figure 17.

2-4: Next, place three contacts as indicated in Figure 18. Recall that the hcells library has a predefined contact called CON, this predefined cell comes precovered with the appropriate metal layer. Select Create -> Instance and specify hcells, CON, and layout in the Library, Cell, and View fields in the Create Instance Form.

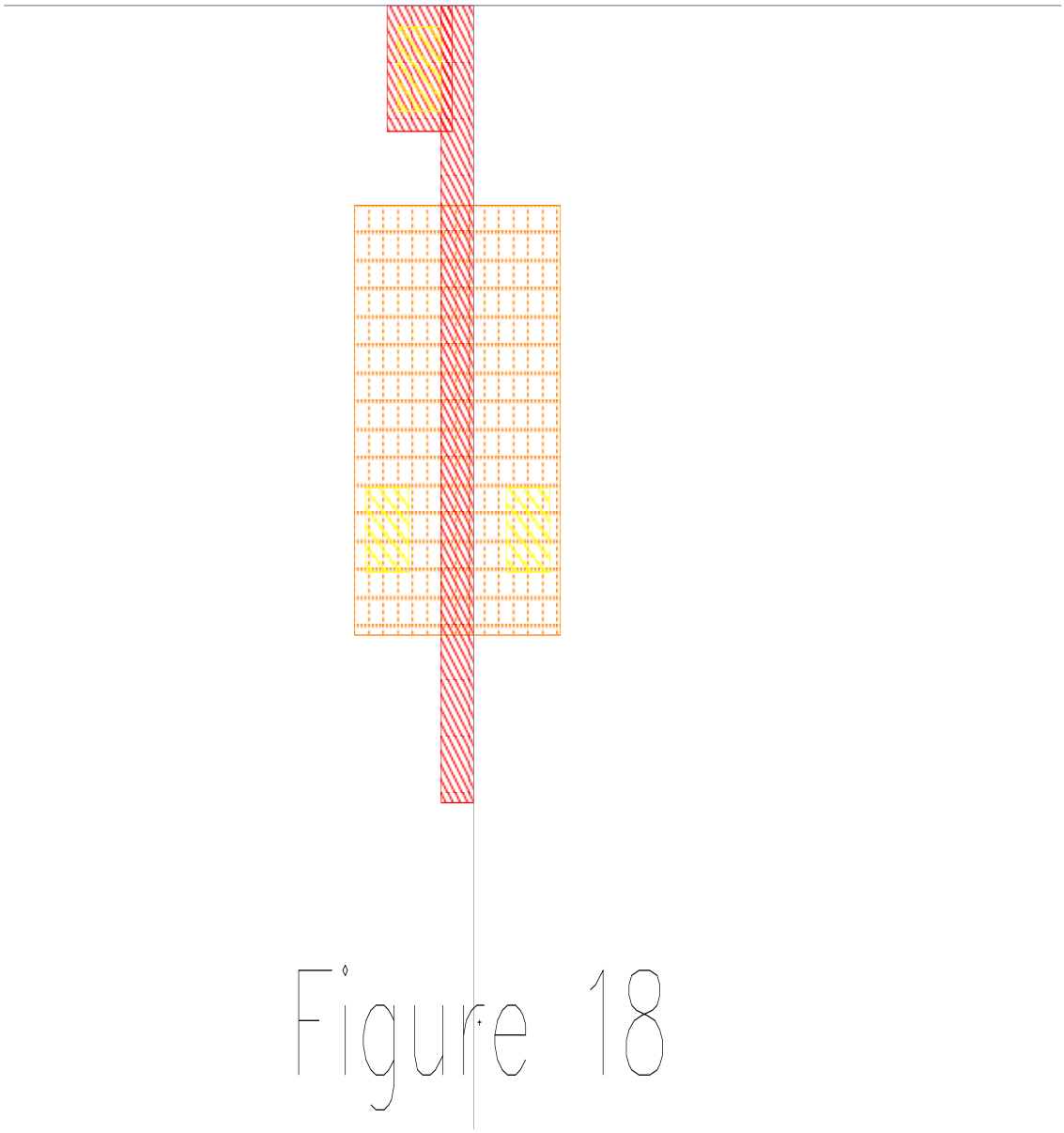


Figure 18.

2-5: The next step is to define the substrate layer. In a nmos transistor this is a pisland dg layer. Select pisland dg from the LSW window and create a rectangle and place this rectangle at the bottom of the transistor, see Figure 19.

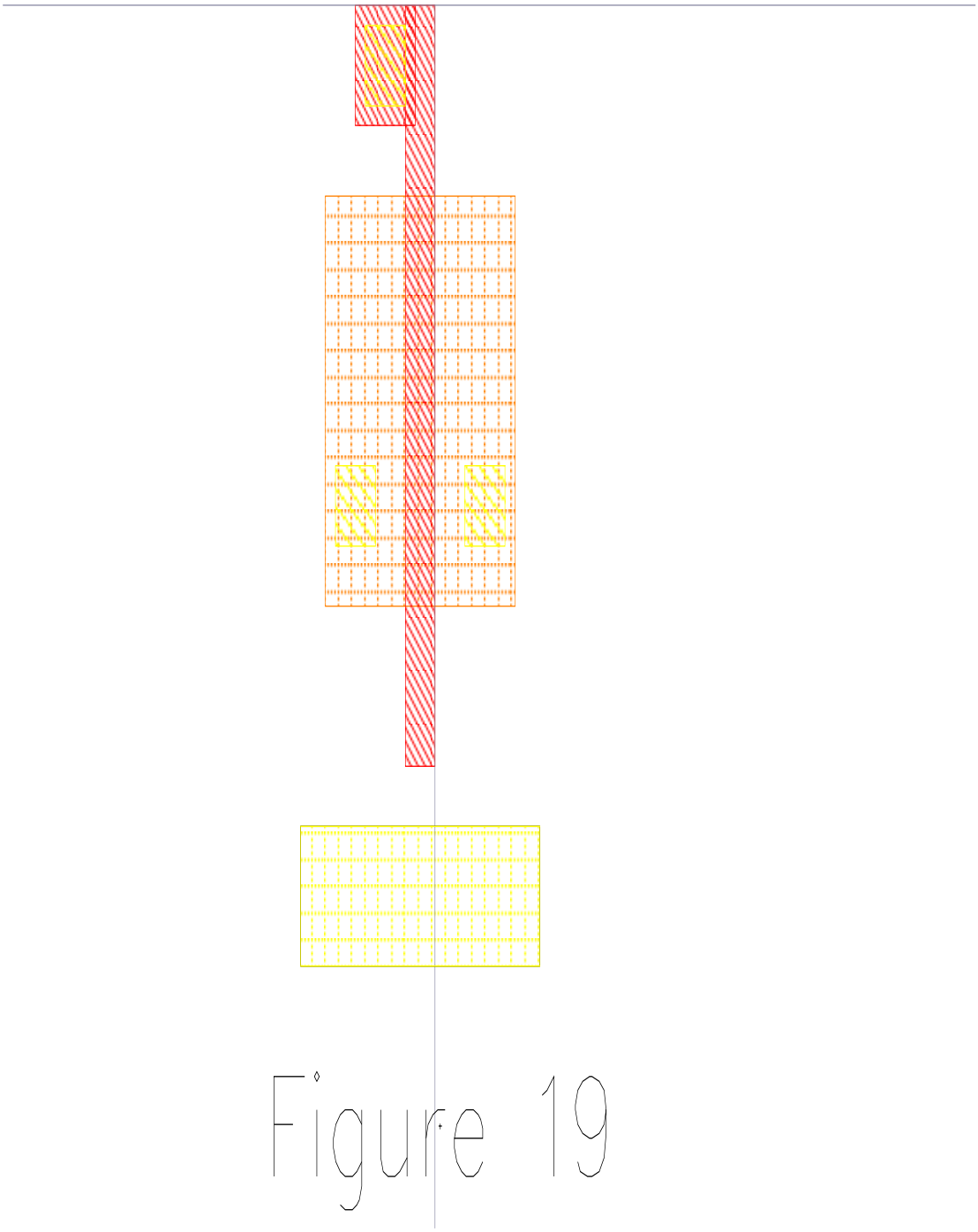


Figure 19.

2-6: Place a contact at one end of this pisland dg rectangle. We will (in the next step) connect this contact with one of the contacts in the nisland dg; this will establish the source-to-substrate connection. See Figure 20.

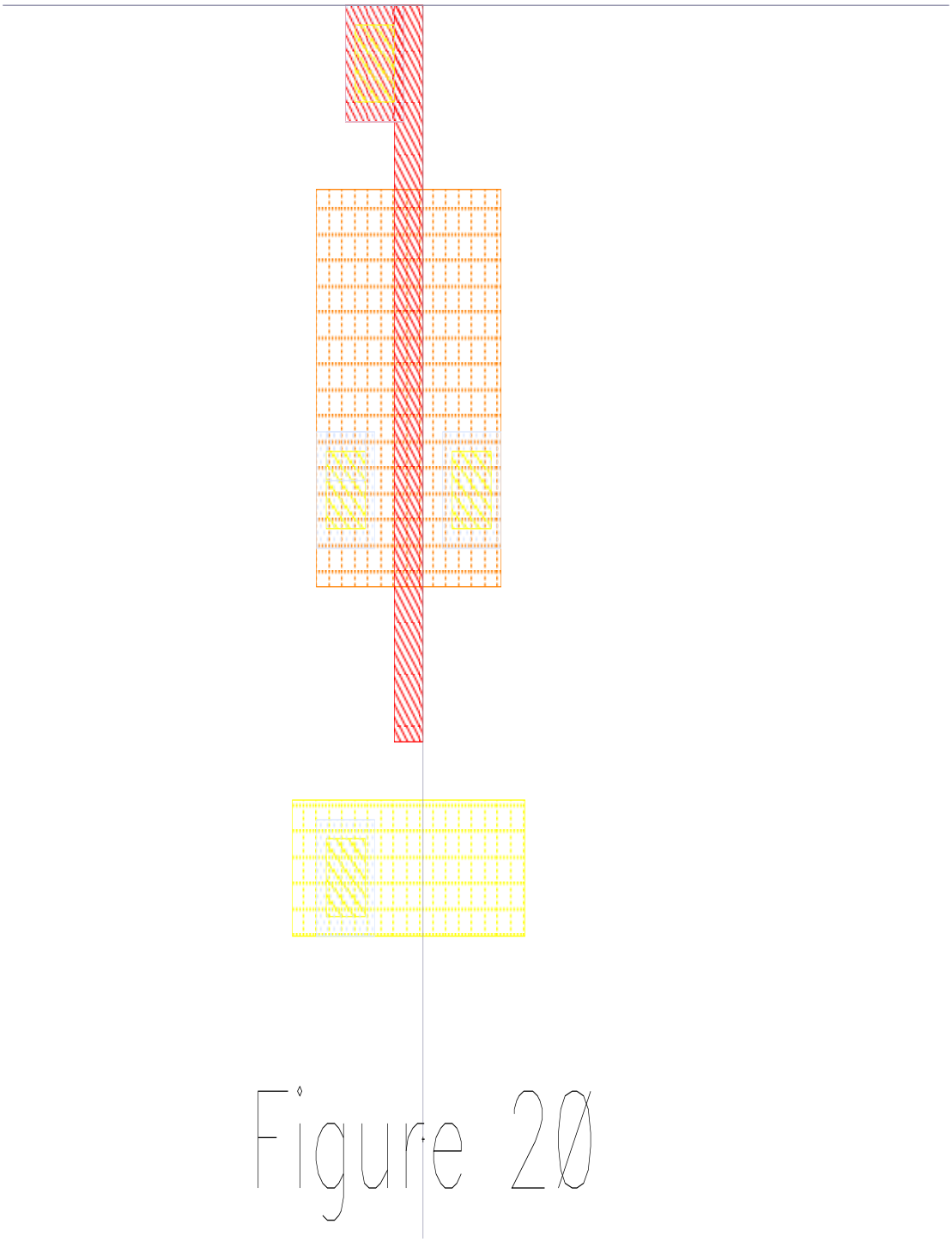


Figure 20.

2-7: Connect one of the contacts in the nisland dg rectangle with the contact you have just placed in the pisland dg substrate with a rectangle of metal1 dg. Refer to Figure 21 for the details. This figure also indicates the terminals of the nmos transistor: Gate, Drain, Source, and Substrate.

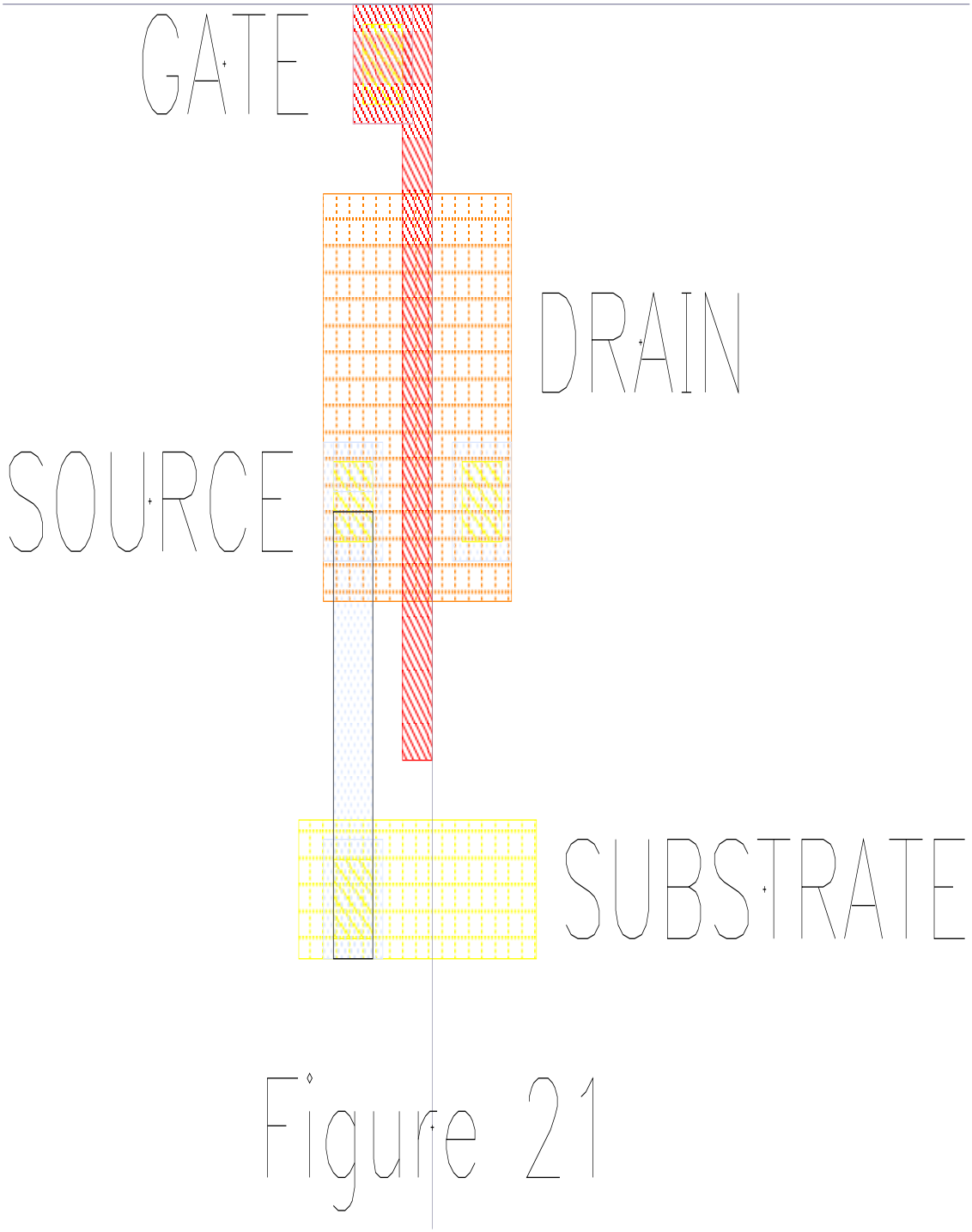


Figure 21.

2-8: Save your work; Select Design -> Save from the Layout window.

2-9: Perform a Design Rule Check of your nmos layout. Select Verify -> DRC from the Layout window and click OK in the DRC window which will appear. The results of the Design Rule Check will appear in the CIW window. If you have any errors, fix them and redo the DRC.

2-10: Once you have finished using the Cadence design tools, remember to quit from them. It is very important to exit from the applications. Merely logging off from the terminal does not quit the application, the processes belonging to the application continue to run in the background. This severely degrades system performance; a few errant processes left running in the background can quickly bring a once quick workstation to a snail pace. Select **File -> Exit** from the main CIW window to exit. A popup window will appear asking **OK to exit icfb?** . Another window will appear stating "**You have edited the way your layers appear. The layer display information is stored in a display.drf file. Do you want to save your changes?** Click on **Cancel** to exit the application.

III: Creating an Inverter using Instances of NMOS and PMOS transistors

This section will explore a bottom-up design methodology to create a CMOS inverter circuit using the two previously created cells (the pmos and nmos cells). The methodology is based upon the fact that once a particular cellview has been created, this cellview can then be inserted into a new cellview. Using this approach, one may construct a library of basic building blocks such as an inverter, a NAND gate, and XOR gate, etc and then use these components to build larger, more complex circuits (i.e. half-adder, full-adder, flip-flop, etc).

3-1: Start Cadence by typing **cmosis5** from the UNIX prompt.

3-2: Create a new cellview to hold your inverter layout. Attach this new cellview to the existing library which you created in the first part of this tutorial. Use the layout view.

3-3: To add an instance of a previously created cellview select **Create -> Instance** from the Virtuoso Layout window. In the Create Instance form which will appear type in the Library name, the Cell name, and the view name for the instance you wish to create. For example:

Library tedcomponents
Cell pmos
View layout

Of course, your names would refer to your actual library and cell names.

3-4: Attached to the cursor will be a yellow outline. Move the cursor to where you want to place this instance and left click with the mouse. The instance will appear as a rectange outlined in red with the instance name inside the rectangle (i.e. my instance was named hpmos). You can use the **Window -> Zoom out by 2** feature in the Layout Window to zoom to a different view. Refer to

Figure 22 for details.

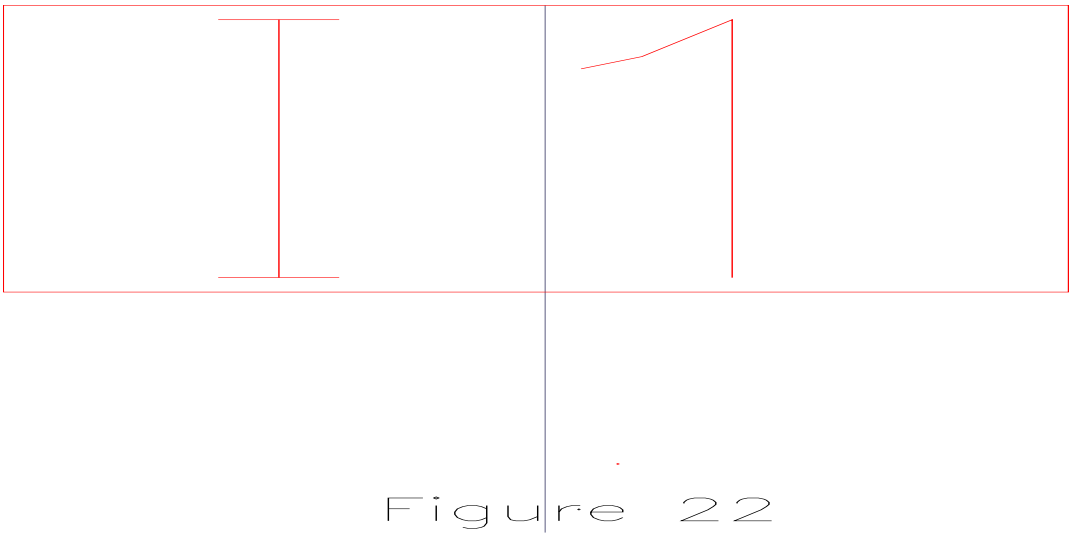


Figure 22.

3-4: Repeat the above two steps to create and place an instance of your nmos cell. Place this instance slightly below the pmos instance. Refer to Figure 23.

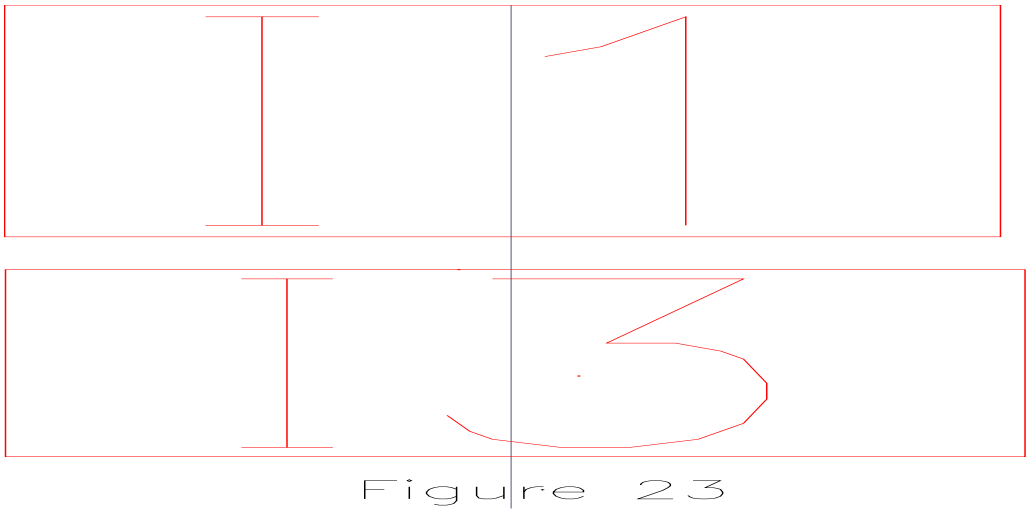


Figure 23.

3-5: You will notice that SHIFT-F will change the view displaying all the layers in the instances.

CTRL-F will change to the outline view. However, before changing views to display all the layers, it is necessary to first **flatten** the instances. By flattening slected instances, one can then go to the objects which compose the instance and edit them (i.e. stretch selected objects, add new objects, etc). To flatten an instance first select the instance by left clicking it with the mouse. Next, select **Edit -> Hierarchy -> Flatten** from the Layout window and click OK in the Flatten popup window. You will notice that the selected instance now appears with all its layers drawn. Refer to Figure 24.

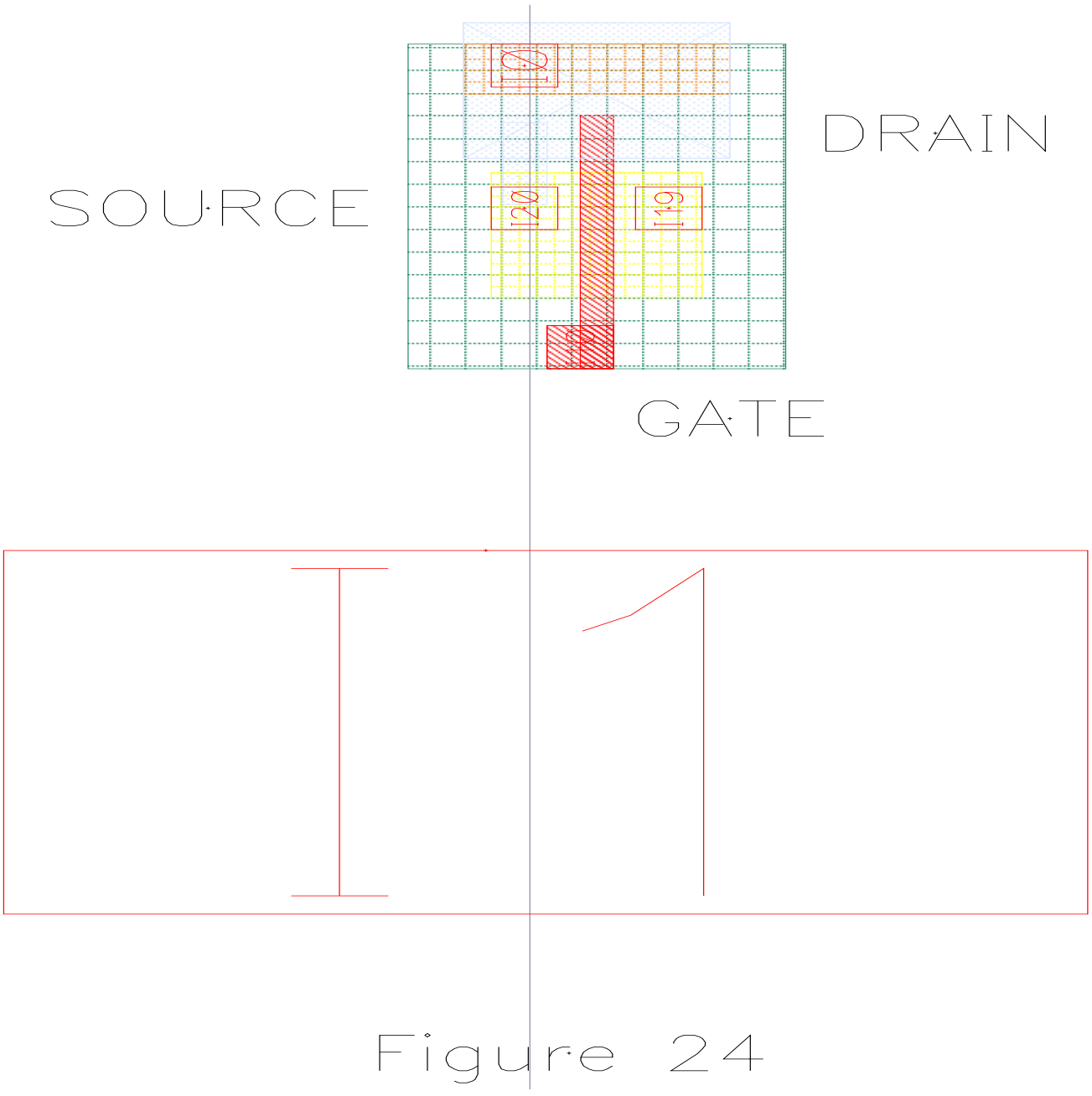


Figure 24.

3-6 Repeat the above procedure to flatten the other instance in your cellview. Refer to Figure 25.

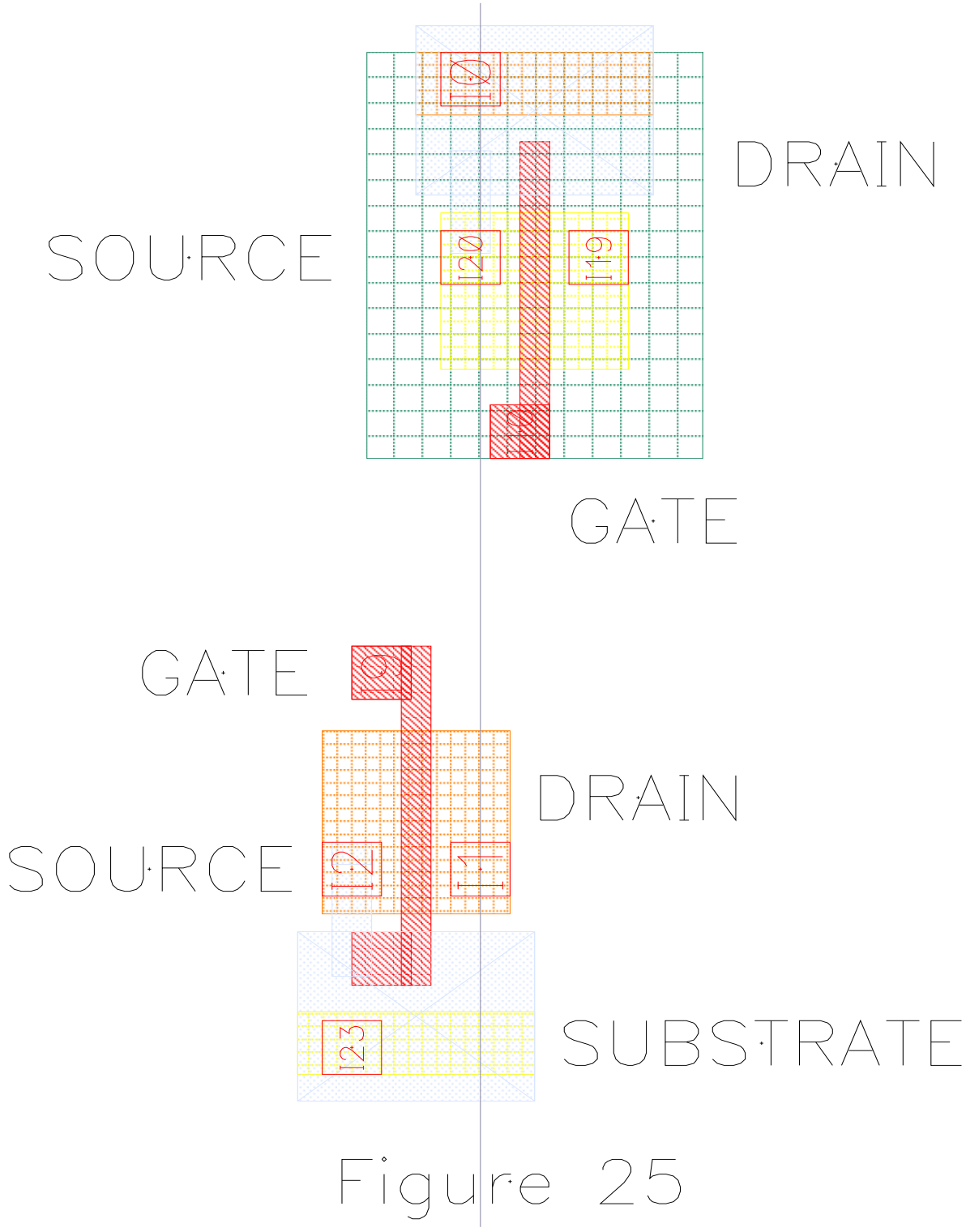


Figure 25.

3-7: You may note that in these figures there are two large white rectangles with an X through them. These are absent from the original cellviews corresponding to the nmos and pmos cells. These areas are metal1 pins covered with a layer of metal1 dg. These establish the positive supply and ground connections for the inverter. We will now add these pins and meatl1 coverings.

3-8: Select metal1 pn from the Layer Select Window, then select **Create -> Rectangle** from the Layout window and place a rectangle over the substrate of the pmos transistor. Next, select **metal1 dg** from the LSW and cover this rectangle with a layer of metal1.

3-9: Repeat the above procedure to make a **metal1 pn** covered with a layer of **metal1 dg** and place this over the substate area of the nmos cell.

3-10: The next step is to place a **symbolic pin** anywhere inside of these two metal1 pins. We will create a symbolic pin for the positive power supply connection (VDD) and one for the ground connection (VSS). Select from the Layout Window **Create -> Pin**. In the **Create Symbolic Pin** Window form which will appear enter **VDD** as the terminal name, i**nputOutput** as the IO type, and **metal1_T** as the pin type. You will note that the pin width is set to 0.8 microns as a default value. Refer ro Figure 26 for the details of the settings for this form.

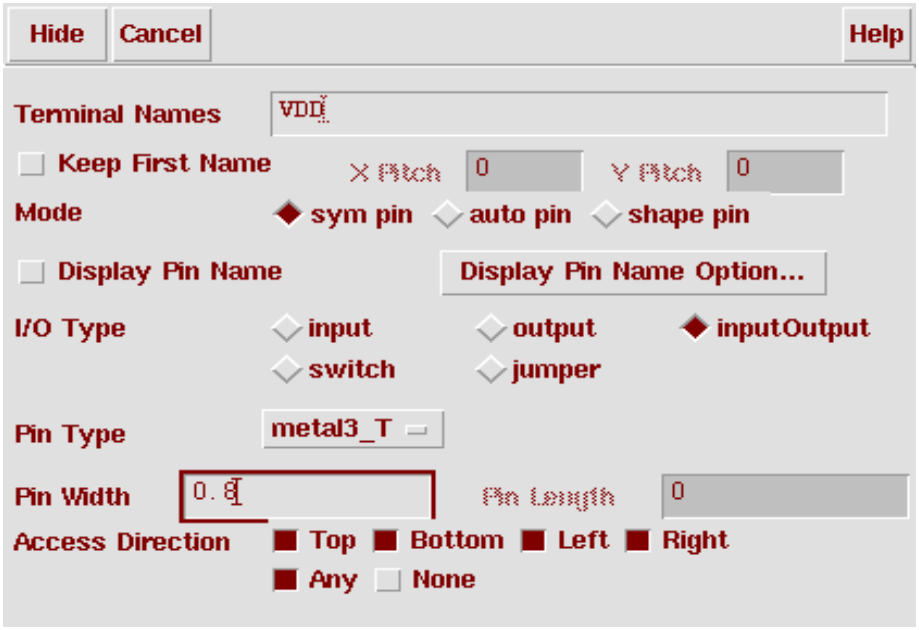


Figure 26: Create Symbolic Pin Form.

3-11: Place this VDD pin anywhere inside the metal1 pn in the pmos transistor.

3-12: Repeat the above to create a symbolic pin with terminal name **VSS** and direction inputOut-

put and Pin Type metal1_T anywhere inside the metal1 pn rectangle in the nmos transistor.

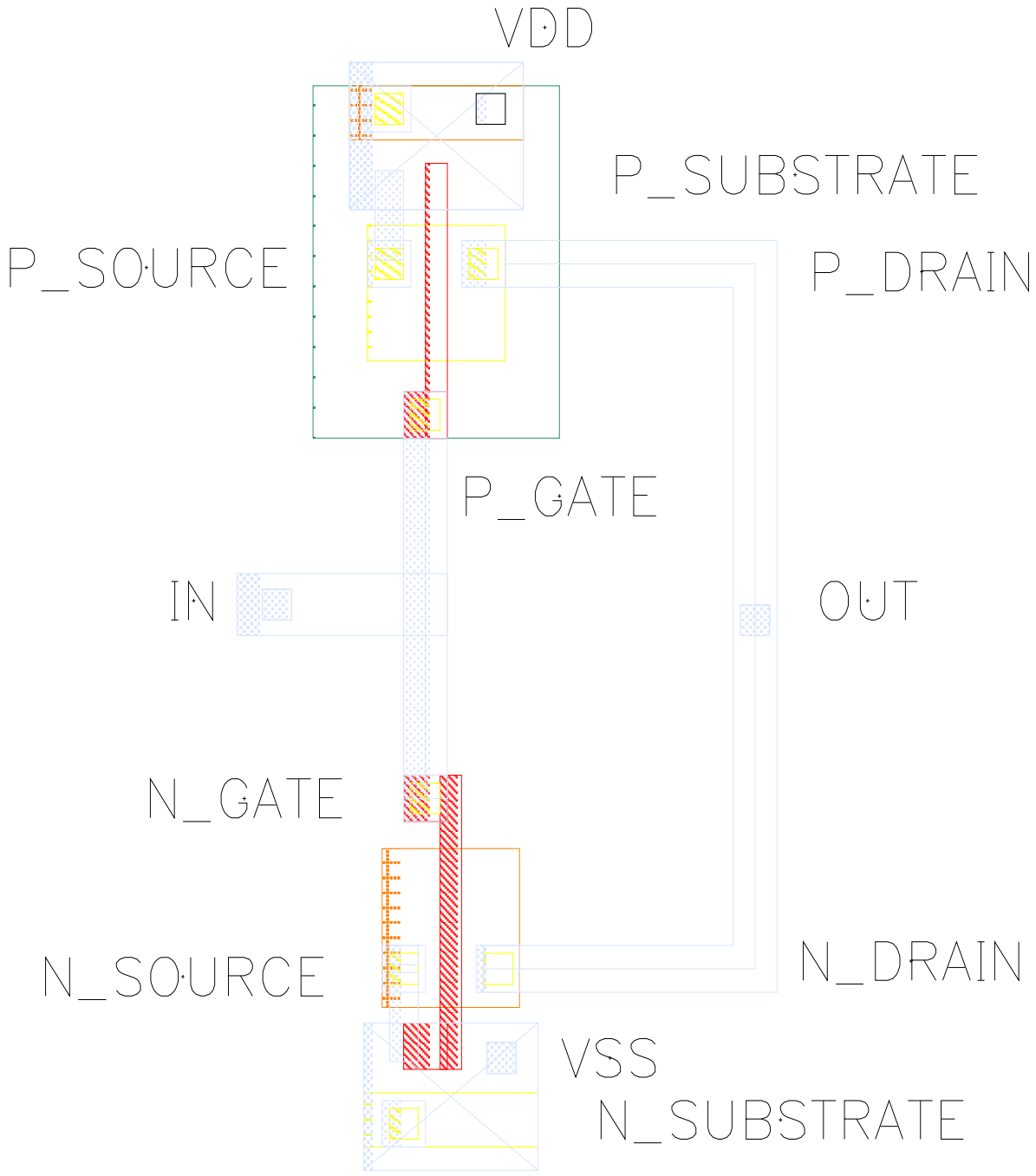


Figure 27

Figure 27.

3-13. Referring to Figure 27, connect the two gates of the two transistors with a metal1 rectangle. Place a **symbolic pin** with terminal name **IN** and direction set to **input** and Pin Type **metal1_T** to define the input pin in this metal1 dg rectangle. Connect the two drains with **metal1 dg**. This can be done by selecting **metal1 dg** from the LSW window, and selecting **Create -> Path** in the Layout Window. Define the path using the mouse, left clicking each time you want to change direction. It may be necessary to alter the snap mode and the path width in the Create Path form to create a suitable path. Place a **symbolic pin** with terminal name **OUT** and direction **output** and Pin Type as **metal1_T** inside this metal1 dg path connecting the two drains.

3-14: Save and your design. Select **Design -> Save**.

3-15: Perform a Design Rule Check and correct any errors if any. Select **Verify -> DRC** from the layout window and click **OK** in the DRC form. Any errors will be reported in the CIW window.

LAYOUT GUIDELINES:

The CMOS14TB process is a triple-metal process. As such, once can define three distinct metal layers. A good design practice is to use one metal to make horizontal interconnections, and different metal layer to make vertical interconnections. Using this approach, one need not worry about unwanted connections between separate metal layers. Use a via of the appropriate type to make a physical connection between separate metal layers.

IV: SIMULATING THE INVERTER LAYOUT

4-1: To simulate the inverter, one can create a symbol of the inverter layout and then create a new schematic containing this symbol connected to power supply/ground sources and input stimuli. The steps to create a symbol from a layout are as follows.

4-2: Extract the layout by selecting from the Virtuoso Layout Editor window **CMC Skill -> Layout/Extract -> Clean Extracted View** . Observe any messages reported in the CIW window: you should see something similar to:

CMOSIS5cleanExtract()
Running CMOSIS5cleanExtract...
... removed (non-supply) jumper pins and terminals
Completed CMOSIS5cleanExtract

4-3: From the Virtuoso Editing Window, select **Verify -> Extract**. The Extractor window will appear. In this form, set the Switch Names to ‘Cparasitic?’. Click on OK in the Extractor window. There should be 0 errors reported in the CIW window. Refer to Figure 28 for details in filling out the Extract form. **Please note that the exact format of these switch setting may vary depending upon the version of Cadence being used**. To obtain the exact names of valid switch setting click on the Set Switches button next to the Switch Names form. A new window will appear listing the available settings.One can then click on each setting as desired and it will be added to the

list of switch names, or you may type it in manually in the Switch names list. **Similar differences may exist with other windows, explore with the window form to find out the differences between tool versions.**

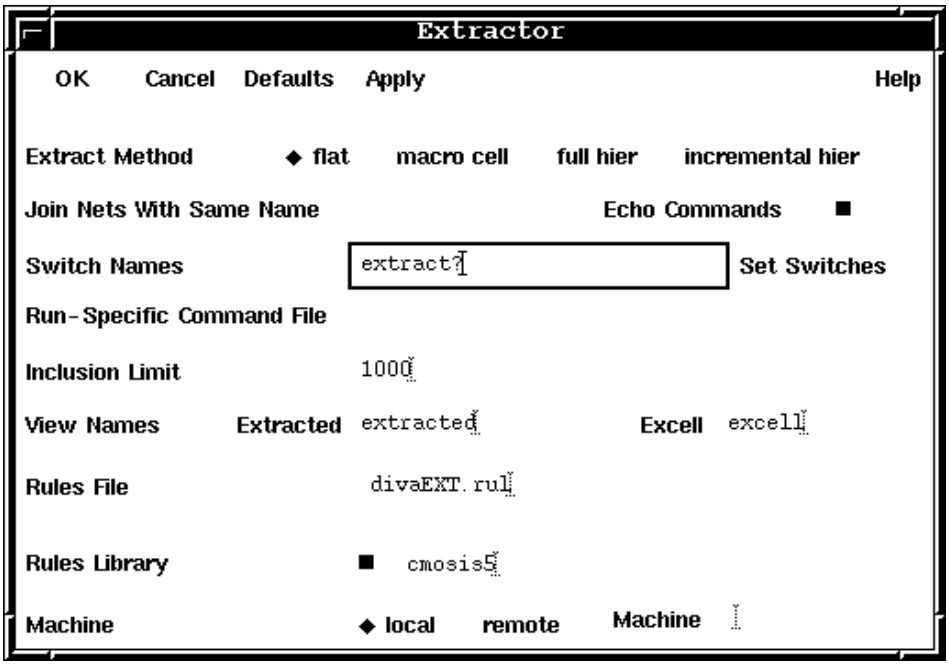


Figure 28: Extract Window with selected parameters.

4-4: The next step is to generate a pin-only schematic. From the Virtuoso Editor window which contains your layout drawing select **CMC Skill -> Layout/Extract -> generate pin-only schematic**. A schematic drawing containing only pins will be generated. This schematic drawing will be used to generate the symbol. The CIW window will report:

CMOSIS5createSchFromWindow()
Ignore the above warning.
Schematic containing only pins for the generation of symbol was created.

4-5: Close the window containg your layout drawing. From the Virtuoso Editing window select **Design -> Save** (in case you have made any changes) then select **Window -> Close**.

4-6: Open the newly created schematic view of your cell. From the CIW window select **File -> Open**. In the Open form, enter the name of your library, the name of your cell, and select schematic as the View name. Select OK in the Open form. A new Composer-Schematic window will appear containing a drawing with only red pins, this is the schematic diagram corresponding to the layout (see Figure 29). The symbol for the layout will be created from this schematic diagram.

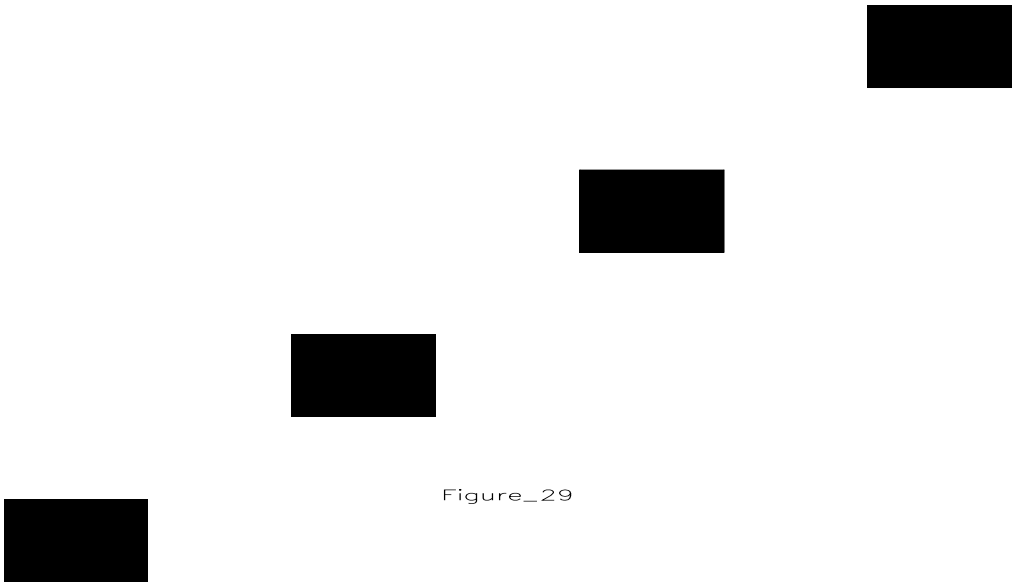


Figure 29.

4-7: From the Composer-Schematic window which just opened up, select **Design -> Create Cellview -> From Cellview**. Ensure that the From field is set as schematic and the To field is set as symbol in the Create Cellview form which will appear. Click on **OK**. Refer to Figure 30.

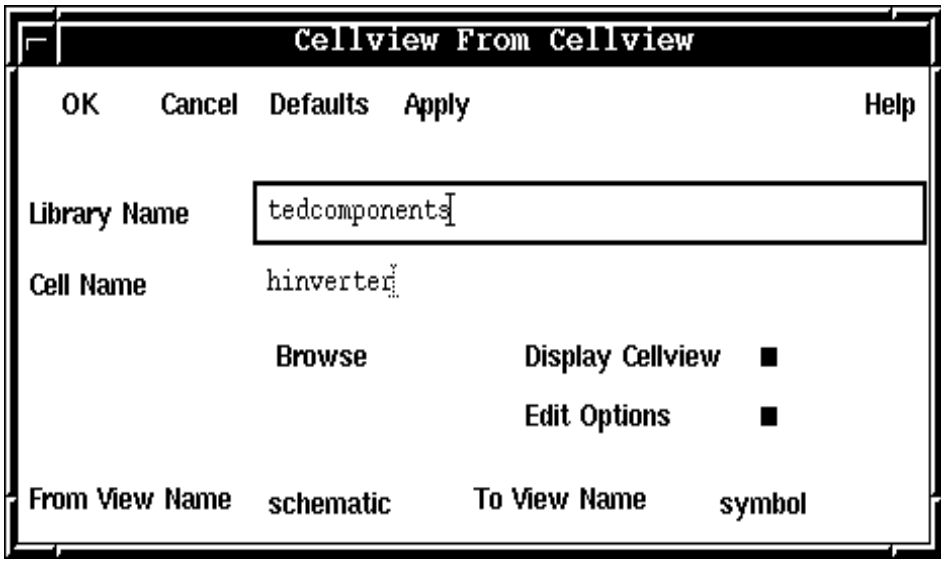


Figure 30: Create Cellview Form.

4-8: A new window will appear. This window will list the pins which will appear in the symbol and their names and orientations . Click on **OK** in this window to generate the symbol for the layout.

4-9: A new schematic diagram can now be drawn using this symbol to create a test circuit as explained in the previous tutorial (A Tutorial on Using Cadence's Schematic Editor and Analog Artist to Create and Simulate a CMOS Inverter at the Transistor Level using CMOSIS5 Technology). That is, add an instance of this symbol, add a power supply source, an input source, a tiedown, VDD and VSS pins, wire up the circuit and simulate it. Figure 31 shows a typical test schematic.

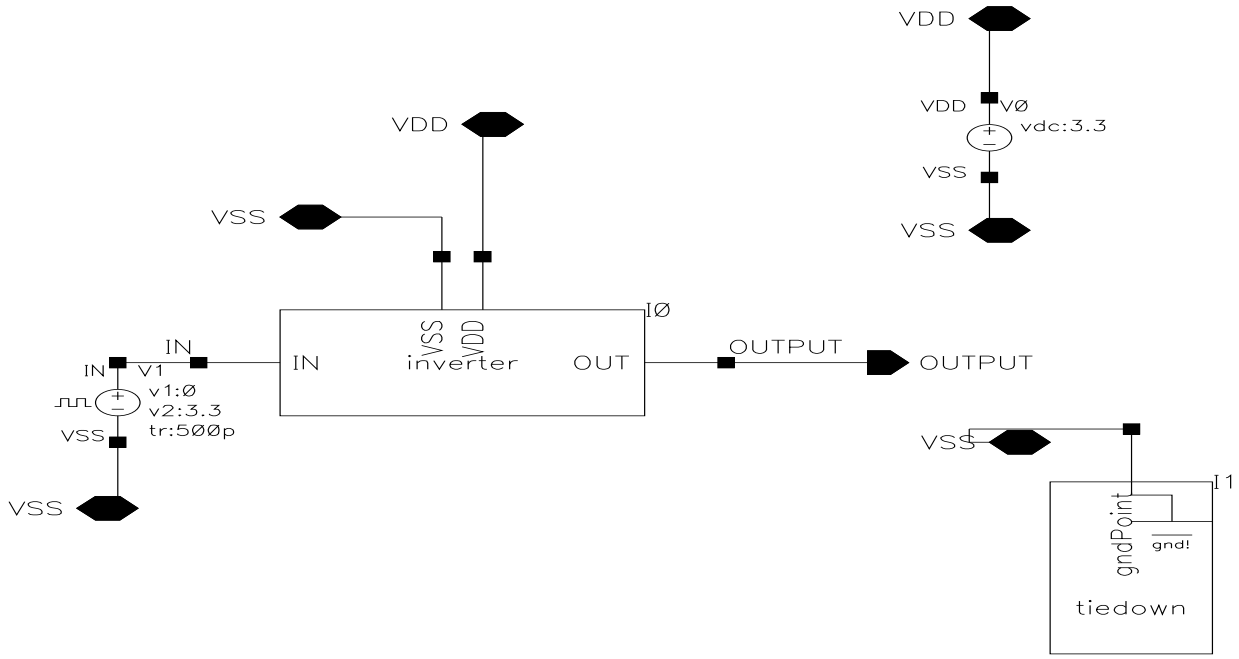


Figure 31.

IMPORTANT : ensure that the `Switch View List` of the Setup -> Environment window contains the *extracted* keyword BEFORE the *schematic* keyword or else the layout simulation will fail (the circuit netlist will not contain the subcircuit consisting of your layout). The switch view list is a list of views that the Cadence software switches into when searching for design information. The software searches through the hierarchical views to be netlisted in the order specified in the list This list must contain the name of the simulator. For Cadence 2004a, the
default switch view list is:

Switch View List : hspiceS spice cmos_sch cmos.sch schematic

You must manually add *extracted* before schematic in the Switch View List:

Switch View List : hspiceS spice cmos_sch cmos.sch **extracted** schematic

V: CONSTRUCTING A PADFRAME AND ROUTING A CIRCUIT

Once a circuit has been completed at the layout level and its functionality tested (by creating a symbol and a test schematic), the final step in a full-custom design is to construct a **padframe** and establish the interconnections between the input/output and power supply pads of the padframe and the inputs/outputs and supply connections of the circuit layout.

In this tutorial we will make use of the following five types of pads available as standard cells from the **cmcpads** library: **padinc**, **padoutc**, **CORNERc**, **padvddrcc**, and **padvssrcc**. A brief description of the functionality of these pads may be found by selecting **CMOSIS5 -> CMOSIS5 documentation -> hcells and IO cells Descriptions** from the main window. Briefly these pads are as follows:

padinc - input pad
padoutc - output pad
CORNERc - corner cell with no pad
padvddrcc - supply pad connected to VDD and VDDCORE and VDDRING
padvssrcc - supply pad connected to VSS and VDDCORE and VSSESD and VSSRING

The CMOS14TB process is a triple-metal process. As such it is possible to make routing interconnects using three separate layers of metal: **metal1 dg**, **metal2 dg**, and **metal3 dg**. Each of these layers is electrically insulated from each other and as such may intersect without short circuiting. This is very helpful for establishing circuit interconnections. Whenever it is necessary to make an interconnection between two intersecting metal layers, a via of the appropriate type must be placed at the intersection point. Two general rules for the use of vias are:

via1: use a via1 to establish a connection between intersecting metal1 dg and metal2 dg layers.

via2: place a via2 at the intersection of a metal2 dg and metal3 dg layer to establish a connection between these two layers.

There are design rules concerning the use of vias, for example a via1 must be a minimum size of 0.8 micron by 0.8 micron and must be covered by metal1 and metal2 with a certain overlap of the metals beyond the boundary of the via. A DRC check will reveal if you have violated any of these rules when creating and placing vias.

When using the standard input/output and supply pads found in the cmcpads library, one must know how to make connections from these pads to one's circuit. This section explains how to tap in and from the pads found in the cmcpads library.

PADINC

Figure 32 illustrates an example of the padinc input pad available in the cmcpads library. It consists of a passivation layer (the blue layer with the square waves on it). It also contains a VDD rail at the top of the pad and a VSS rail at the bottom of the pad. Found at the bottom of the pad near

the right hand side will be a small metal2 square with the label OP near it. This defines the output terminal of this pad. Simply drop a metal2 rectangle on top of this square and route this rectangle to your circuit input. Figure 33 reveals a magnified view of this tap point for the padinc pad.

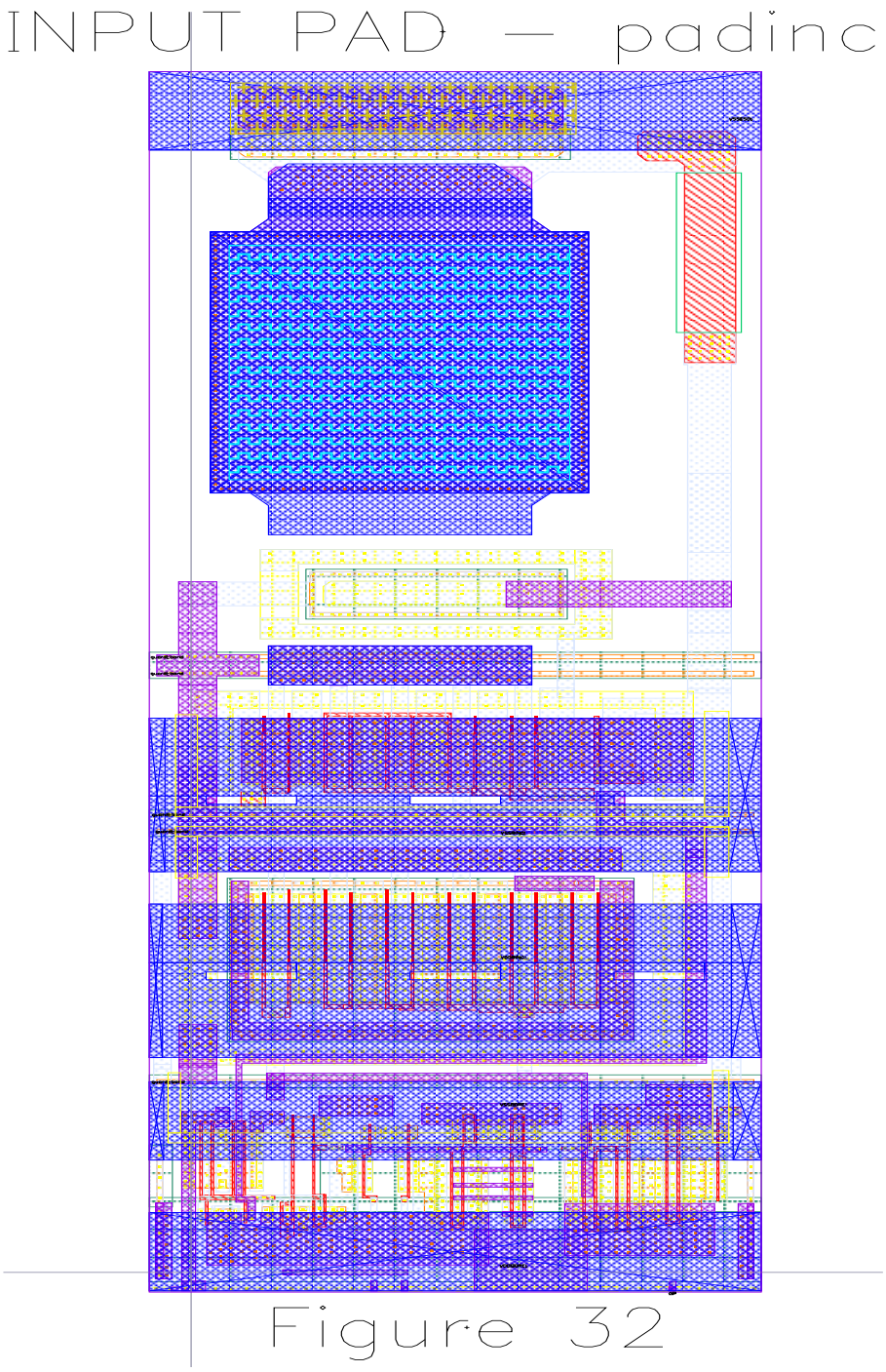


Figure 32.

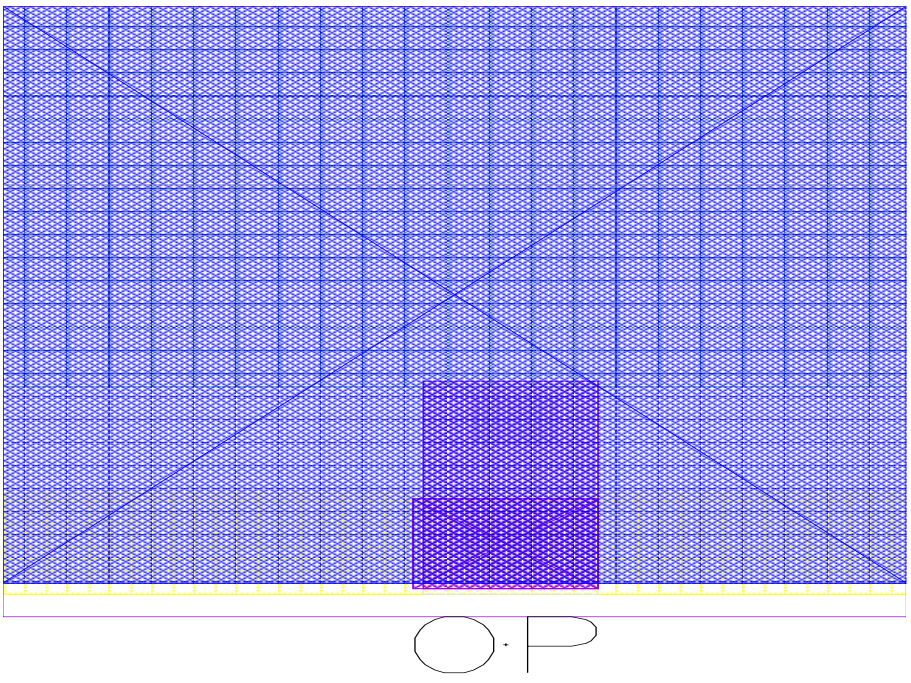


Figure 33

Figure 33.

PADOUTC

To make a connection to a padoutc (see Figure 34), connect metal2 dg to the small metal2 dg sqaure found at the bottom of the pad labelled IP. Figure 35 is a magnified view of this portion of the padoutc.

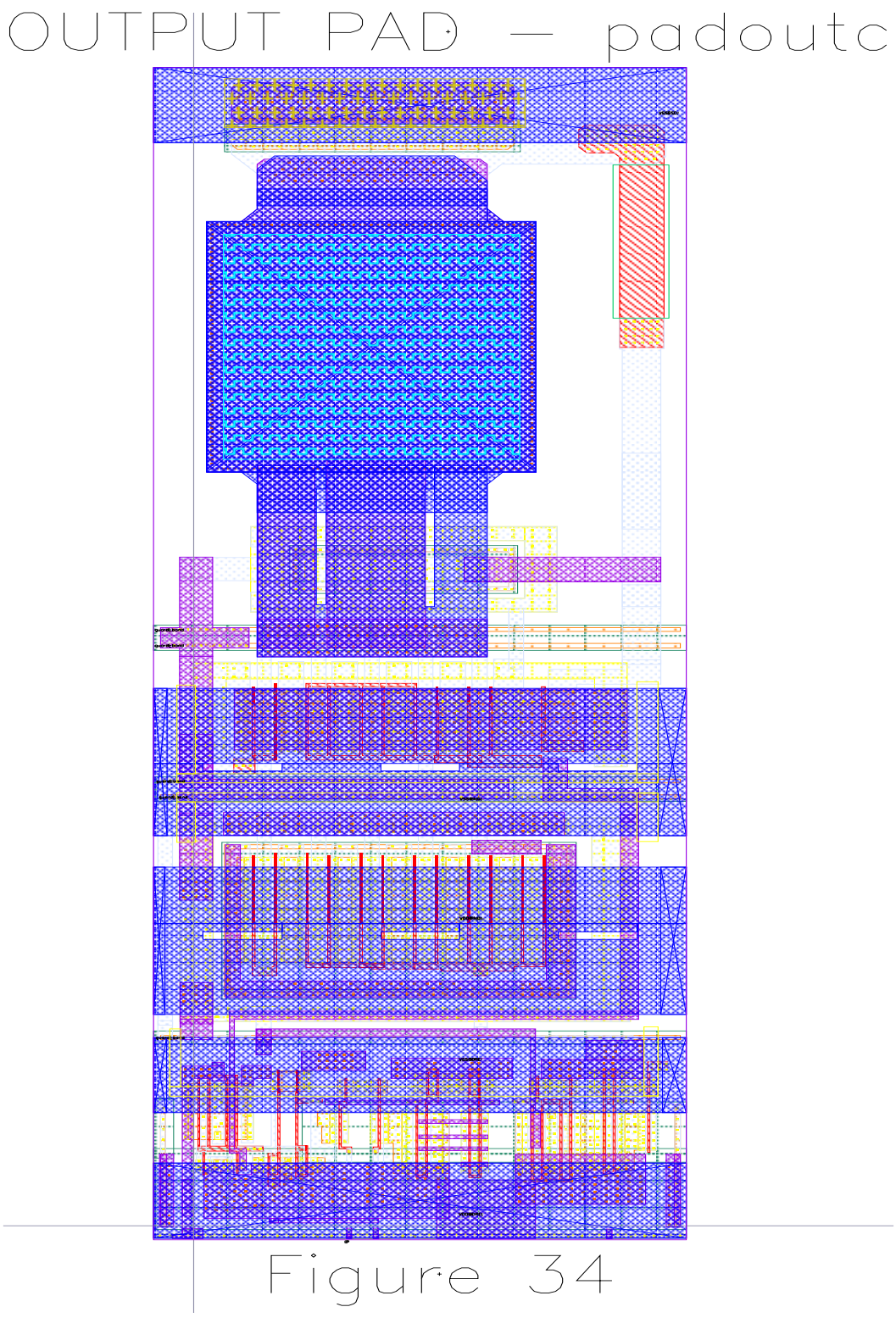


Figure 34.

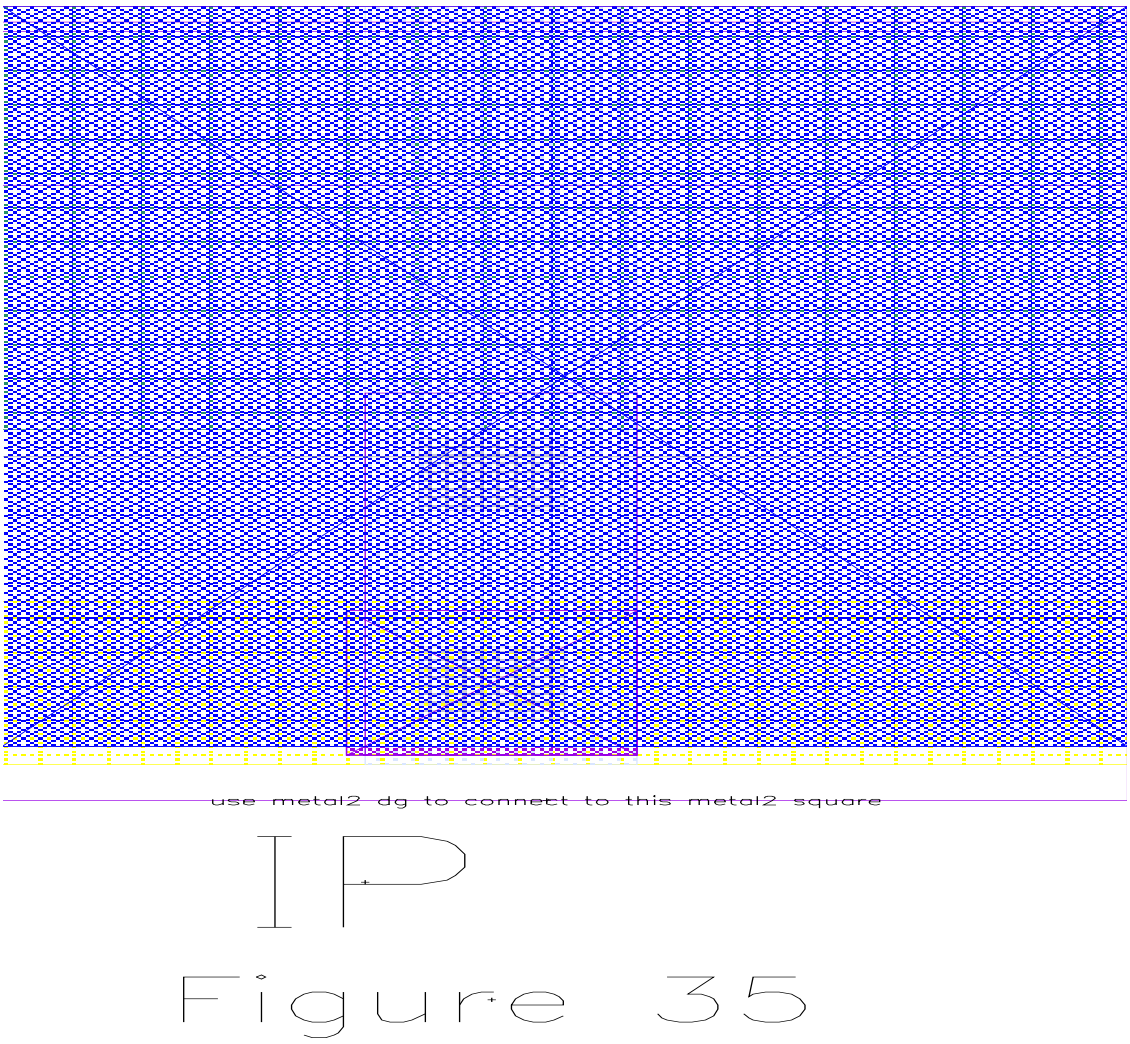


Figure 35.

PADVDDRCC

Figure 36 shows the padvddrcc. The connection point is the small metal1 square (of size 30 x 26.90 microns) found at the bottom of the cell. Use the metal1 dg layer to connect to this square. Then distribute this metal1 to all the VDD points in your layout. Figure 37 is a magnified view showing the connection point.

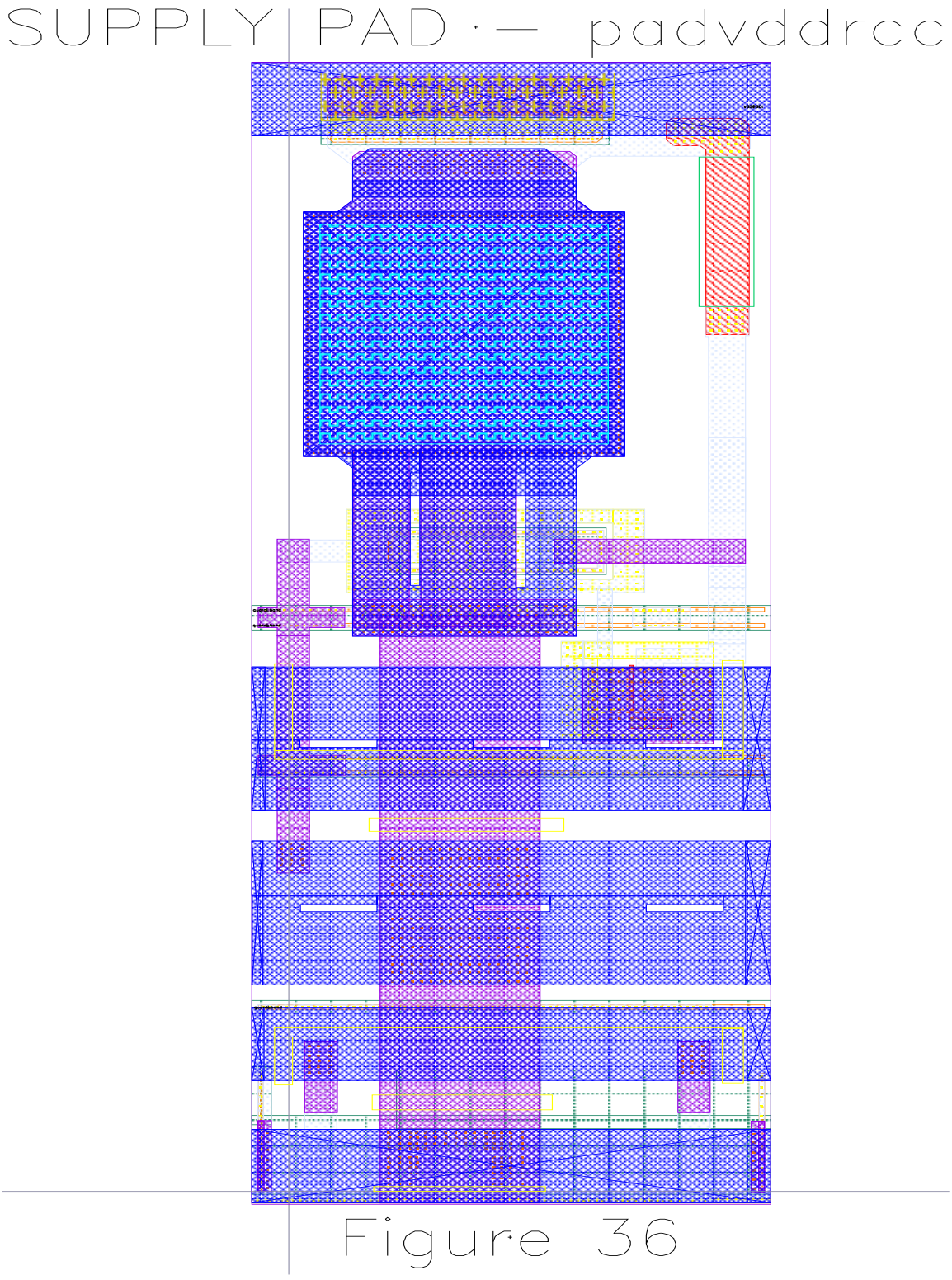


Figure 36.

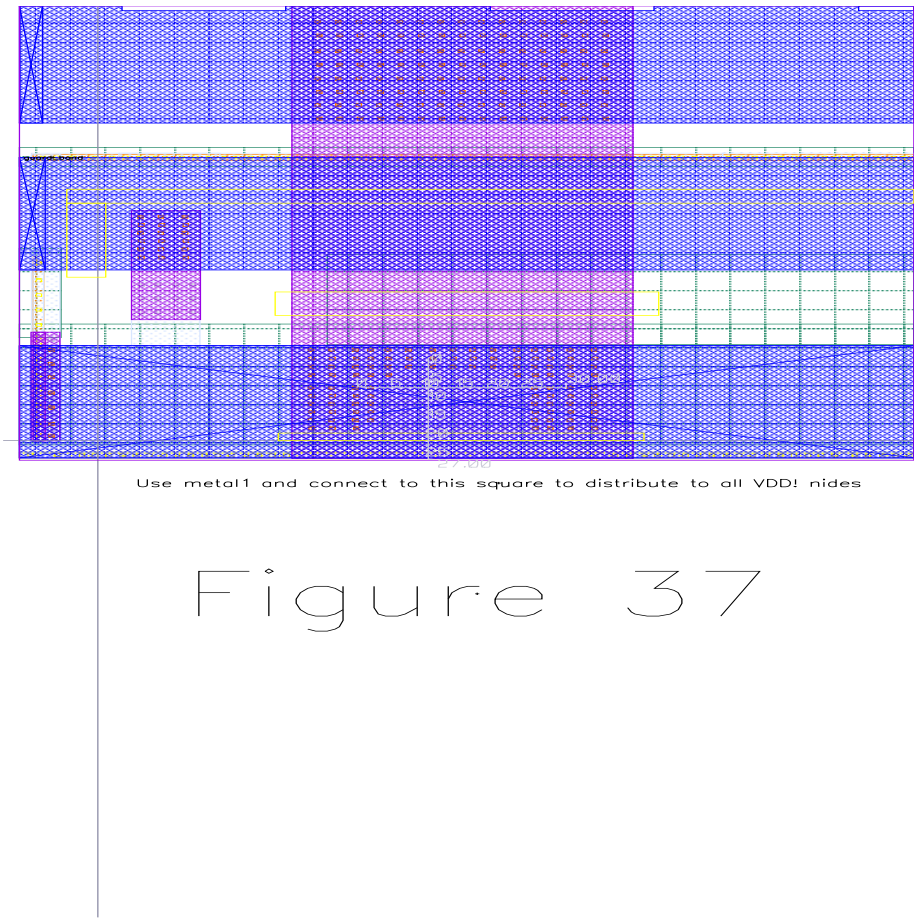


Figure 37.

PADVSSRCC

Figure 38 shows the layout for the padvssrcc cell. The connection method is very similar to that used for the padvddrcc cell. You will find a metal1 square of about 30 x 26.9 microns at the bottom of the cell. This is the tap point, connect some metal1 to this square and distribute it to you VSS nodes in your circuit. Figure 39 shows a magnified view of the connection area.

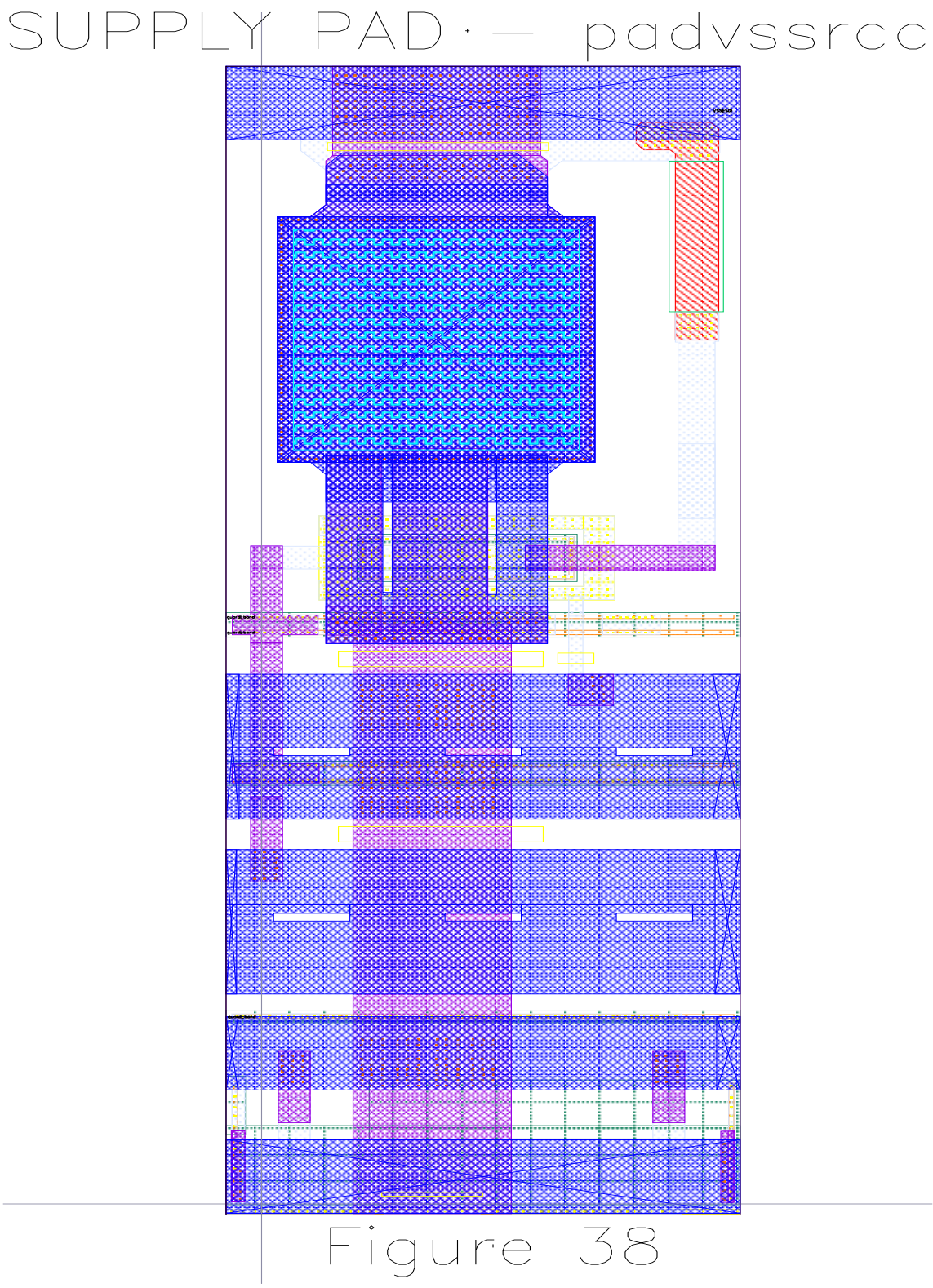


Figure 38.

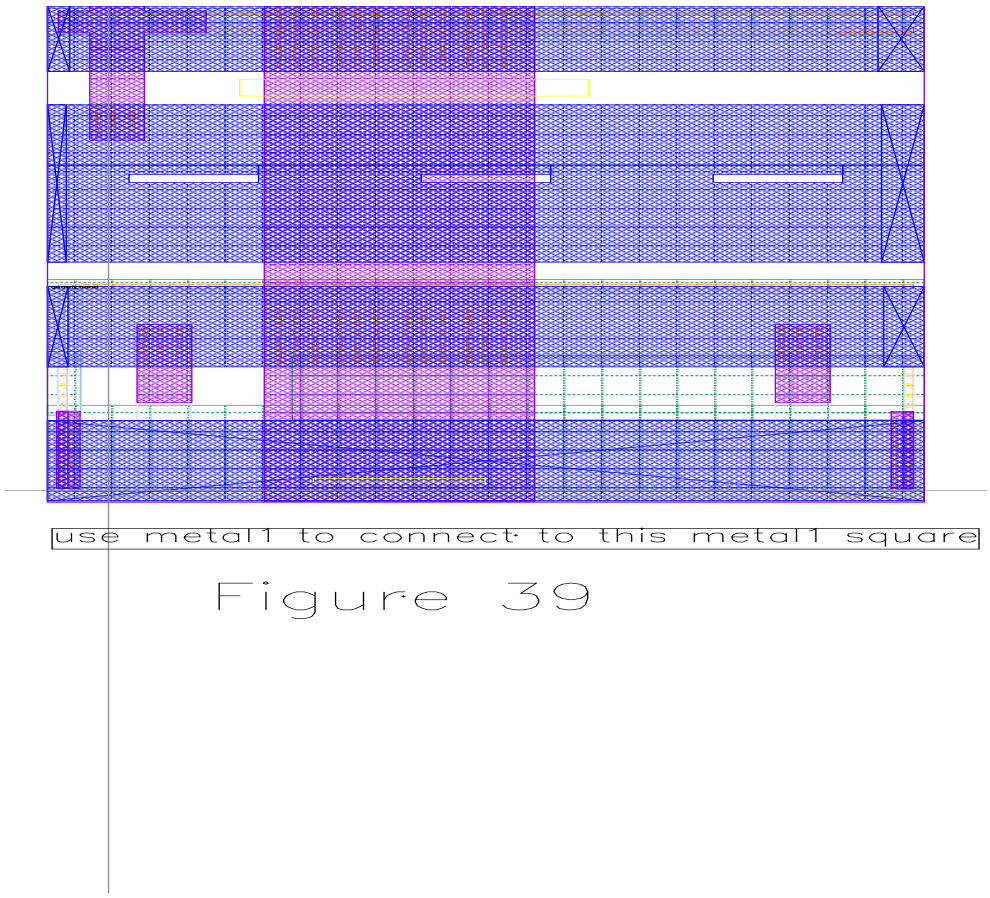


Figure 39.

Constructing the padframe:

5-1: Refer to Figure 40 when constructing a padframe. This padframe consists of two input pads of type padinc, two output pads of type padoutc, four corner cells of type CORNERc, a VDD pad of type padvddrcc and a VSS pad of type padvssrcc. Use a separate input pad for each input in your design and a separate output pad for each output. The pads are found in the cmcpads library. Add appropriate instances of each pad into your layout window. The placement of the pads is critical. Make sure that the pads overlap each other (by about 1 micron or so) to ensure contiguous distribution of the VDD and VSS rings. Also, if the pads are improperly spaced relative to each other, some design rule violations may occur. I found the following to be a good rule of thumb to follow when designing a padframe: offset the supply pads and the i/o pads to the EXTERIOR of the corner cells. For example, for the padframe illustrated in this tutorial the offsets would be as shown in Figure 41. Note: the offset shown in this figure is greatly exaggerated. The actual offset is only 1-2 microns in reality.

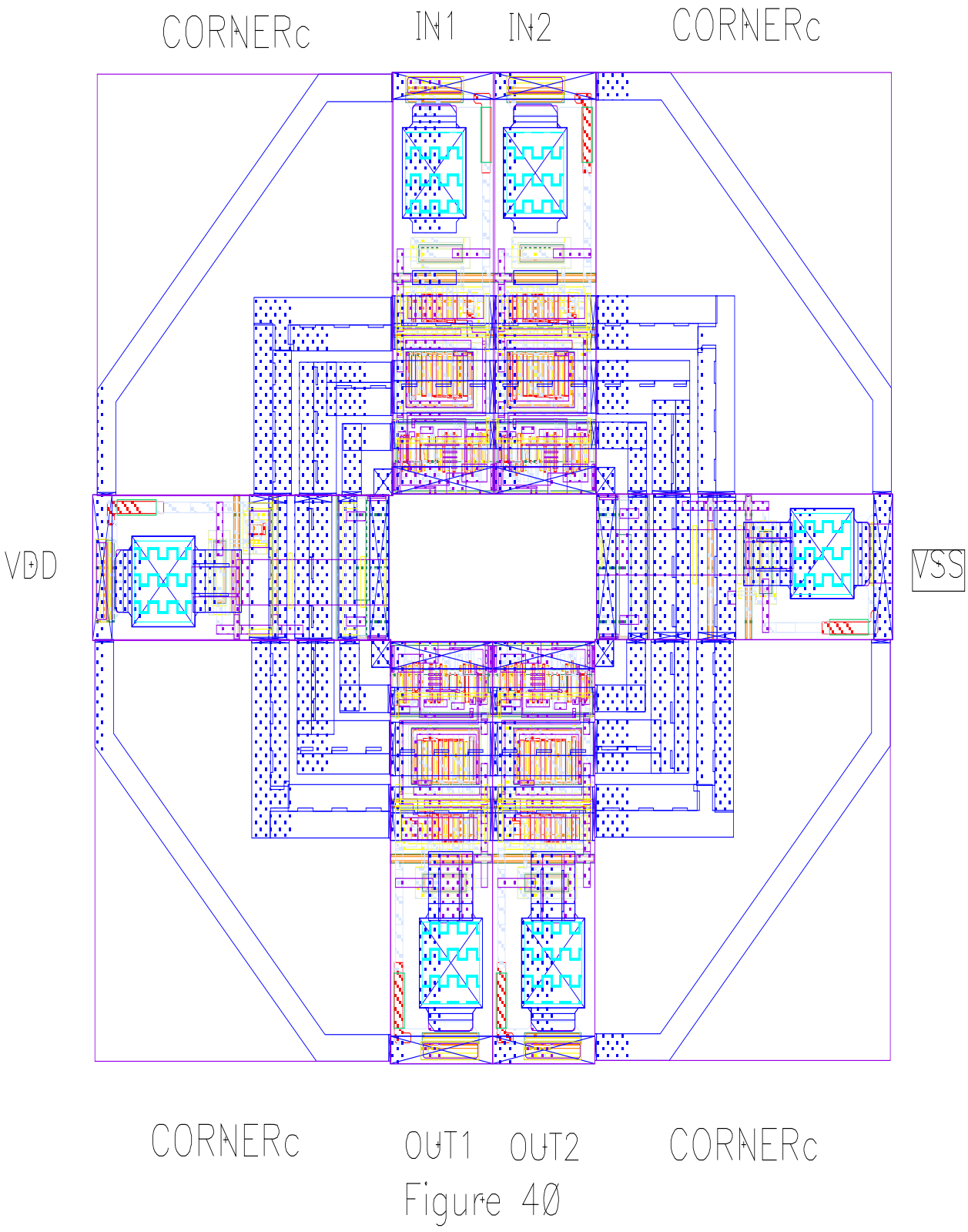

CORNERc
IN1
IN2
CORNERc
VDD
VSS
CORNERc
OUT1
OUT2
CORNERc
Figure 40

Figure 40.

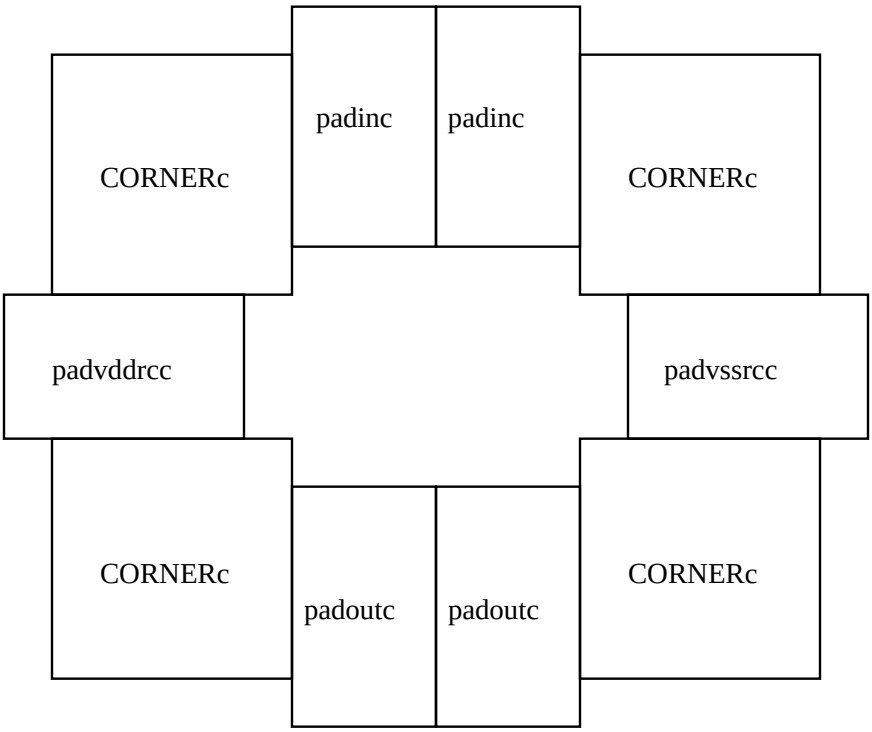


Figure 41: Offset of the pads within a padframe.

5-2: Perform a DRC on the padframe. If there are any errors, correct them by altering the offset of the pads in the frame. Once there are no errors in your padframe, you can add instances of your cells using Create -> Instance and specifying the library name and cell name and view name (layout) of the specific instances you wish to place inside the padframe. This simple example simply consists of two instances of the inverter cell which was created in this tutorial.

5-3: NOTE: if you had created symbolic pins in your instances (as was done in the inverter in order to create a symbol for the inverter and verify its operation), delete these symbolic pins from the instance layout when you bring it inside your padframe. We will add (in a later step) the top-level symbolic pins to the bonding pads on the padframe itself.

5-4: Route the interconnections between the input pads and inputs of your circuit, the outputs of your circuit and the output pads, provide VDD and VSS buses and power your circuit. Refer to Figure 42 and 43 for details. Figure 43 illustrate a method of tapping off from a VDD bus and feeding the VDD terminal of an inverter. Note the use of a via1 to make the necessary interconnection between the two metal layers. Also note the use of distinct metal layers; one layer is chosen for horizontal segments, and another is chosen for vertical interconnect segments.

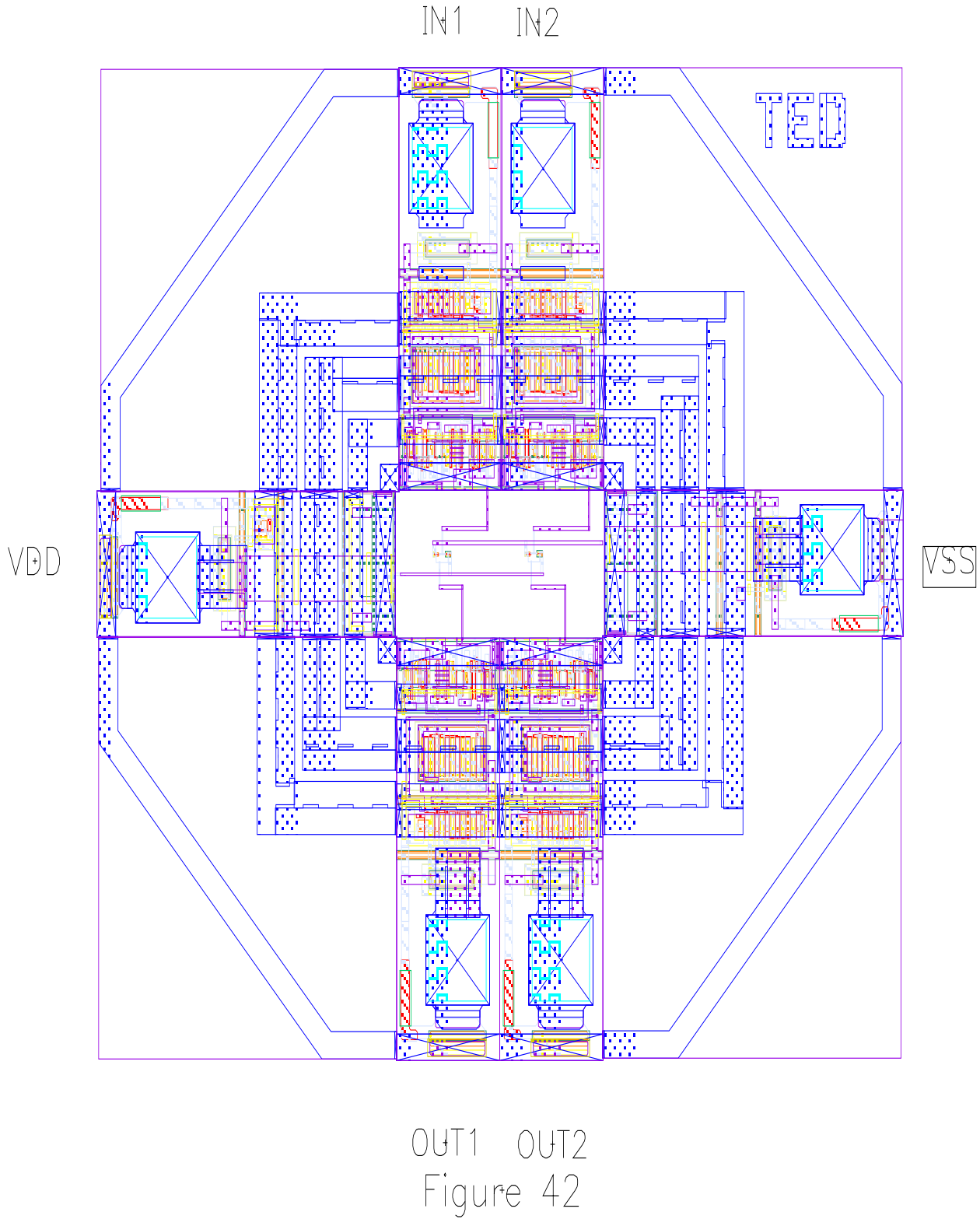


Figure 42

Figure 42.

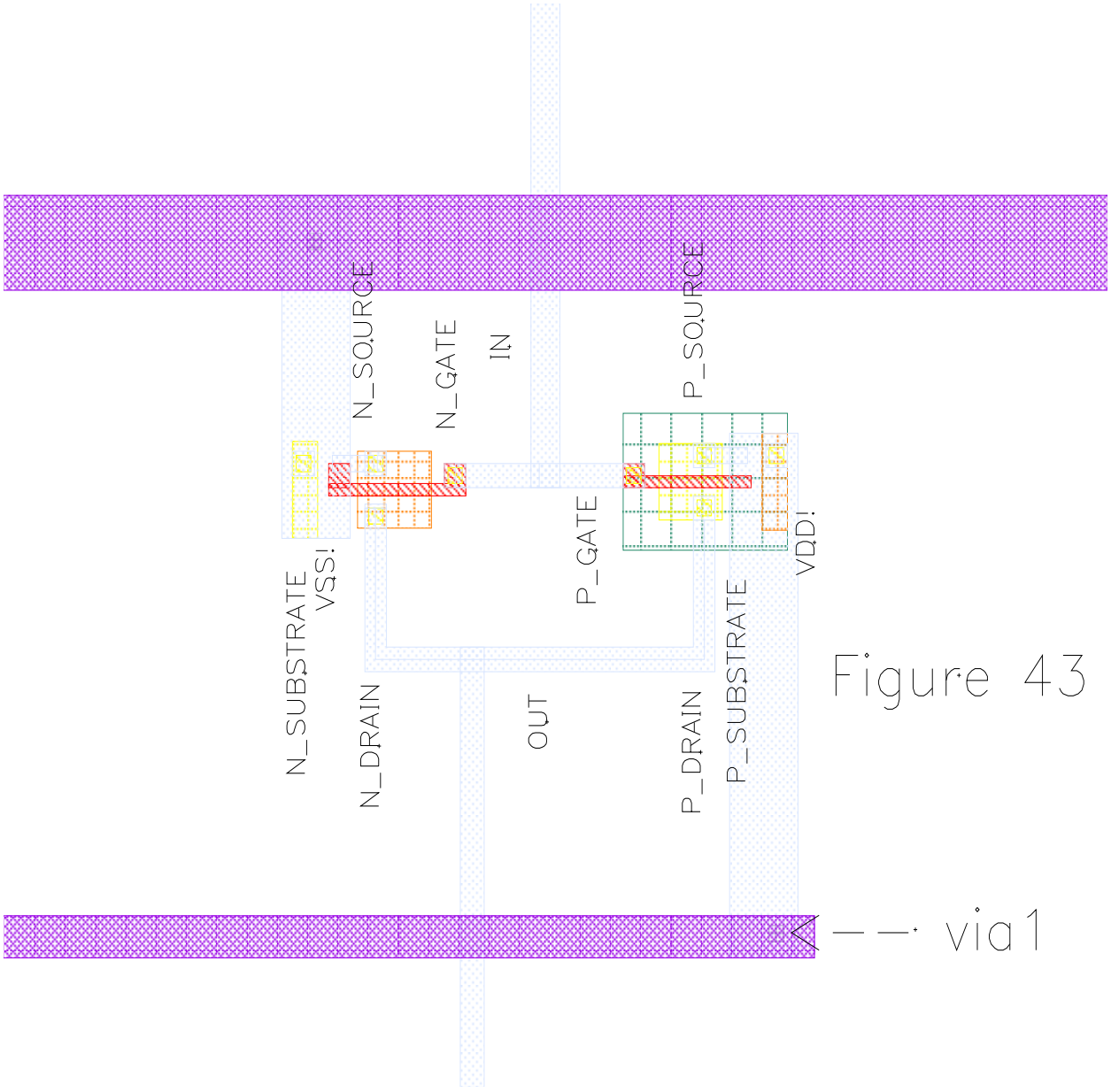


Figure 43.

5-4: The last step is to place a symbolic pin of the appropriate metal type and appropriate direction (in this case metal3_T) on top of each of the bonding pads in each of the pads in the padframe. You will note that each i/o pad and supply pad has a large area which is covered by a set of light blue “square waves”. This is the passivation layer which covers the metal3 layer underneath. The passivation layer determines where the passivation coating (a thin glass layer that is used to coat the entire chip during fabrication) does not cover. This will allow bonding wires to come into contact with the metal pad when the chip is bonded into a package. Place the symbolic pins you

create anywhere inside the passivation layer. Give each symbolic pin a terminal name (for example IN1, IN2, OUT1, OUT2, VDD, VSS and make sure the direction is set as input for input pins, output for output pins, and inputoutput for both the VDD and VSS pins. Place each pin into the appropriate pad within the passivation layer. Use the menu item Create -> Pin from the layout window, fill in the Terminal name for the pin in the Create Symbolic Pin form which will appear and also be sure to set the Pin Type to metal3_T, and the appropriate I/O Type.

5-5: Perform one final Design Rule Check on your completed chip. There should be 0 errors reported.

5-6: A symbol for the chip can now be created (using the same procedure as was done for creating a symbol from a layout). A test schematic may be drawn using this symbol and a suitable test circuit constructed and simulated to verify the overall functioning of the chip.
5-7: Note that if your padframe has any instances of padinc or padoutc cells which are not connected to any node in your circuit an error during simulation will occur. The error message will be: “No DC path from node #??”. A quick remedy is to simply make a dummy connection from an unsed padinc to an unused padoutc. This will however lead to unnecessary power consumption. A more elegant solution is to create custom padblanks which are used as “filler” pads to evenly space out a given padframe.