VLSI Design Layout Using MAGIC Technology: Skywater 130nm

Marym Shahangian, James E. Stine

Oklahoma State University

VLSI Computer Architecture Reaserch Group

Designing digital systems and Very Large Scale Integration (VLSI) circuits requires advanced tools and techniques due to their complexity and high-performance demands. One of these tools is MAGIC (Multiple Application Graphics Interface Circuit), which is used in conjunction with the Skywater 130nm technology. This document aims to explore the use of MAGIC for VLSI layout design and examine how to leverage the benefits of the Skywater 130nm technology to optimize performance and reduce circuit size. Skywater 130nm, known for its ability to create high-density circuits with low power consumption, is a suitable choice for advanced VLSI designs. This study reviews various design stages including layout drawing, parameter extraction, and functional simulation using MAGIC, and analyzes the results obtained from the practical implementation of these techniques.

Magic is an interactive system for VLSI circuit layout design that uses a color graphics display and input devices like a mouse or graphics tablet. It offers advanced features such as continuous rule-checking, connectivity awareness, hierarchical circuit extraction, and routing tools, simplifying the design process. Magic's streamlined design rules enable faster development and ease of use, making it an essential tool for efficient and reliable VLSI design.

James E. Stine, Jr., Ph.D.

Edward Joullian Endowed Chair and Professor in Engineering

Oklahoma State University

Department of Electrical and Computer Engineering

VLSI Computer Architecture Research Group

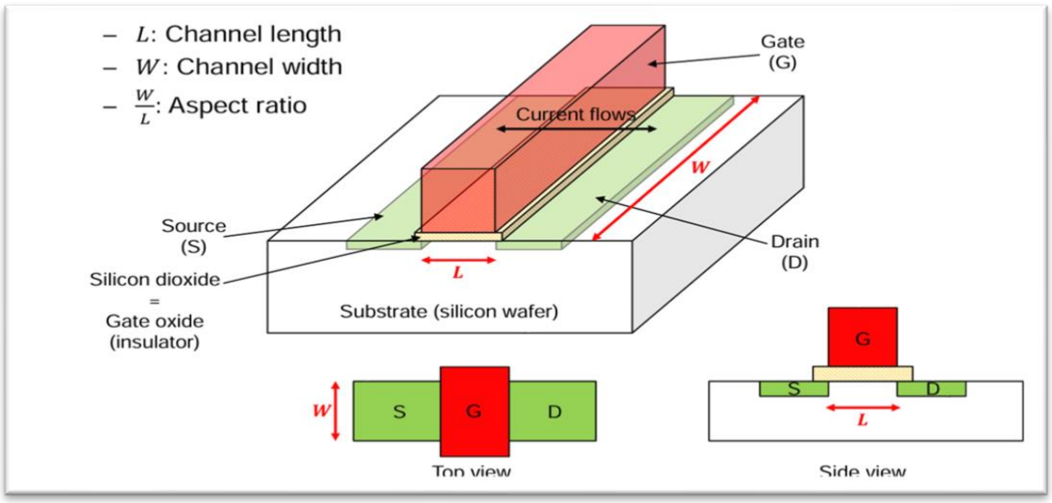
Stillwater, OK 74078 USA

E-mail: james.stine@okstate.edu

Phone: (405) 744-9244

Physical Structure

MOSFETs (Metal-Oxide-Semiconductor Field-Effect Transistors) are fundamental components in CMOS (Complementary Metal-Oxide-Semiconductor) technology. MOSFET is a semiconductor device that uses the electric field effect to control the current.Understanding their physical shape and layer placement is crucial for designing and optimizing integrated circuits.



1. **Substrate (Silicon Wafer)**

   The base material of the MOSFET, usually made of silicon, on which all other layers are built.

2. **Source (S) and Drain (D)**

   Two heavily doped regions in the substrate where current enters (Source) and exits (Drain). These are typically made of n-type or p-type semiconductor material depending on whether the MOSFET is n-channel or p-channel.

3. **Gate (G)**

   A conductive layer placed above the channel, separated by a thin layer of silicon dioxide (SiO_2). The gate controls the current flow through the channel by creating an electric field when a voltage is applied. when voltage is applied to it, creates an electric field in the oxide layer, which either forms or depletes the conduction channel between the source and drain. The gate voltage determines whether the MOSFET is in the "on" or "off" state.

4. **Silicon Dioxide (SiO2)**

   Also known as the gate oxide, this insulator layer lies between the gate and the substrate. It prevents direct current from flowing between the gate and the channel while allowing the electric field to control the channel conductivity.

5. **Channel:** When an appropriate voltage is applied to the gate, a conduction channel forms between the source and drain in the substrate. Current flows through this channel.

   MOSFETs are primarily of two types:

- **n-channel MOSFET:** The conduction channel is formed by electrons.
- **p-channel MOSFET:** The conduction channel is formed by holes.

6. **Channel Length (L)**

   The distance between the source and drain regions. This is a critical dimension as it affects the MOSFET's switching speed and current-carrying capability.

7. **Channel Width (W)**

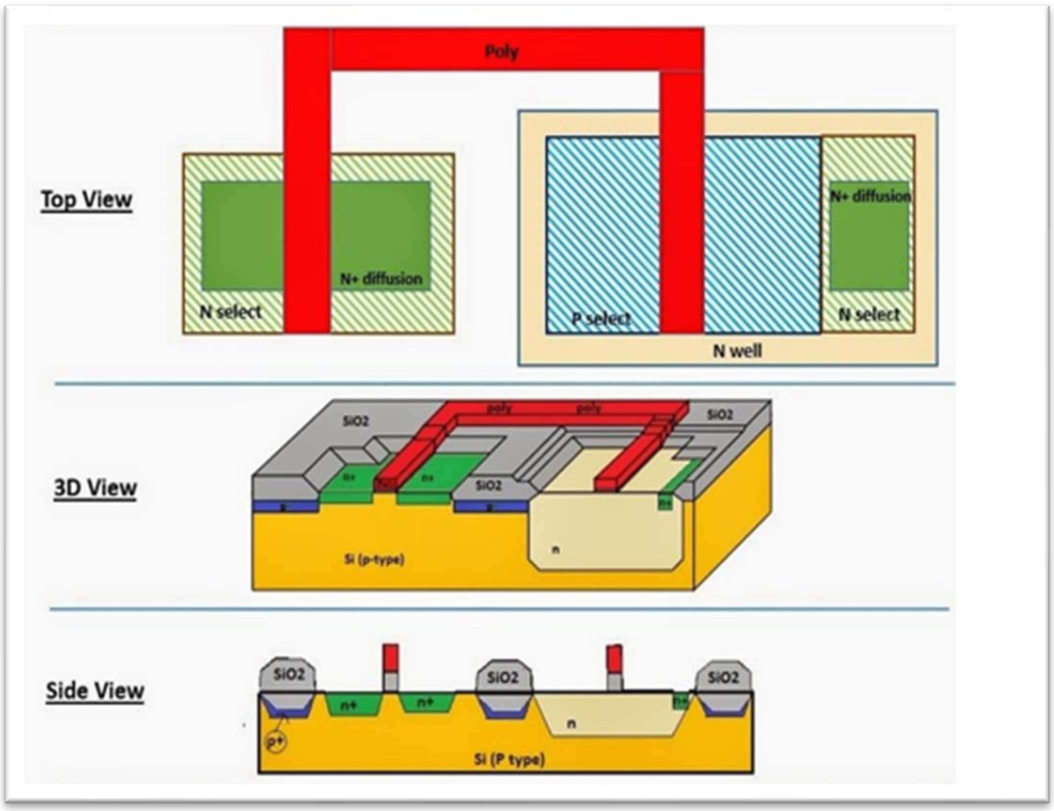
   The width of the region through which current flows between the source and drain. It also influences the current-carrying capability of the MOSFET.

8. **Aspect Ratio ($\frac{W}{L}$)**

   The aspect ratio of the MOSFET is the ratio of the channel width (W) to the channel length (L). This ratio is important for determining the current-driving capability of the MOSFET. A higher aspect ratio generally means a higher current capability.

MOSFET Structure: Top, 3D, and Side Views

For better understanding, the image below shows the structure of a MOSFET from three different perspectives: Top View, 3D View, and Side View. In each of these views, the various layers and regions of the MOSFET are displayed in different colors as follows:



Top View

- **Poly (Polycrystalline Silicon):** The red regions represent the polycrystalline silicon, which forms the gate of the MOSFET. It spans across the n-well and p-well regions.
- **N+ and P+ Diffusion:** The green and blue regions are the heavily doped n-type (N+) and p-type (P+) regions, which form the source and drain of the MOSFETs.
- **N-well and P-well:** The n-well and p-well are regions in the substrate where n-type and p-type transistors are formed, respectively.
- **Select Regions:** Areas marked for N-select and P-select are regions selectively doped to form either n-type or p-type regions, enhancing the transistor formation.

3D View

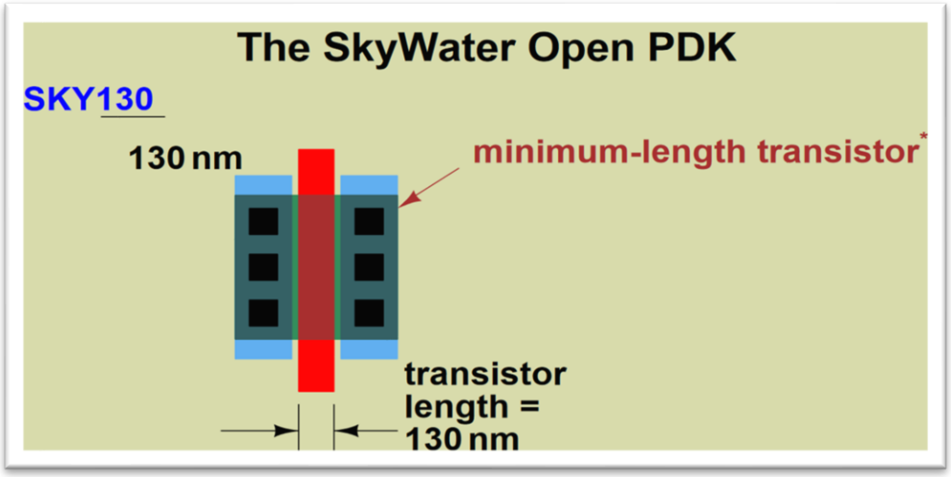
- **Silicon (Si) Substrate:** The yellow area represents the silicon substrate, typically p-type, on which the CMOS structure is built.
- **SiO2 (Silicon Dioxide):** The grey regions are the silicon dioxide insulator layers that electrically isolate different components and act as the gate oxide under the poly gates.
- **N+ and P+ Diffusion:** The green (N+) and blue (P+) regions show the heavily doped areas forming the source and drain of the MOSFETs.
- **Gate (Poly):** The red polycrystalline silicon regions act as the gate electrodes controlling the channel between the source and drain.

Side View

- **Si (P-type) Substrate**
- **N-well and P-well**
- **SiO2 Layers**
- **N+ and P+ Diffusion**
- **Poly Gate**

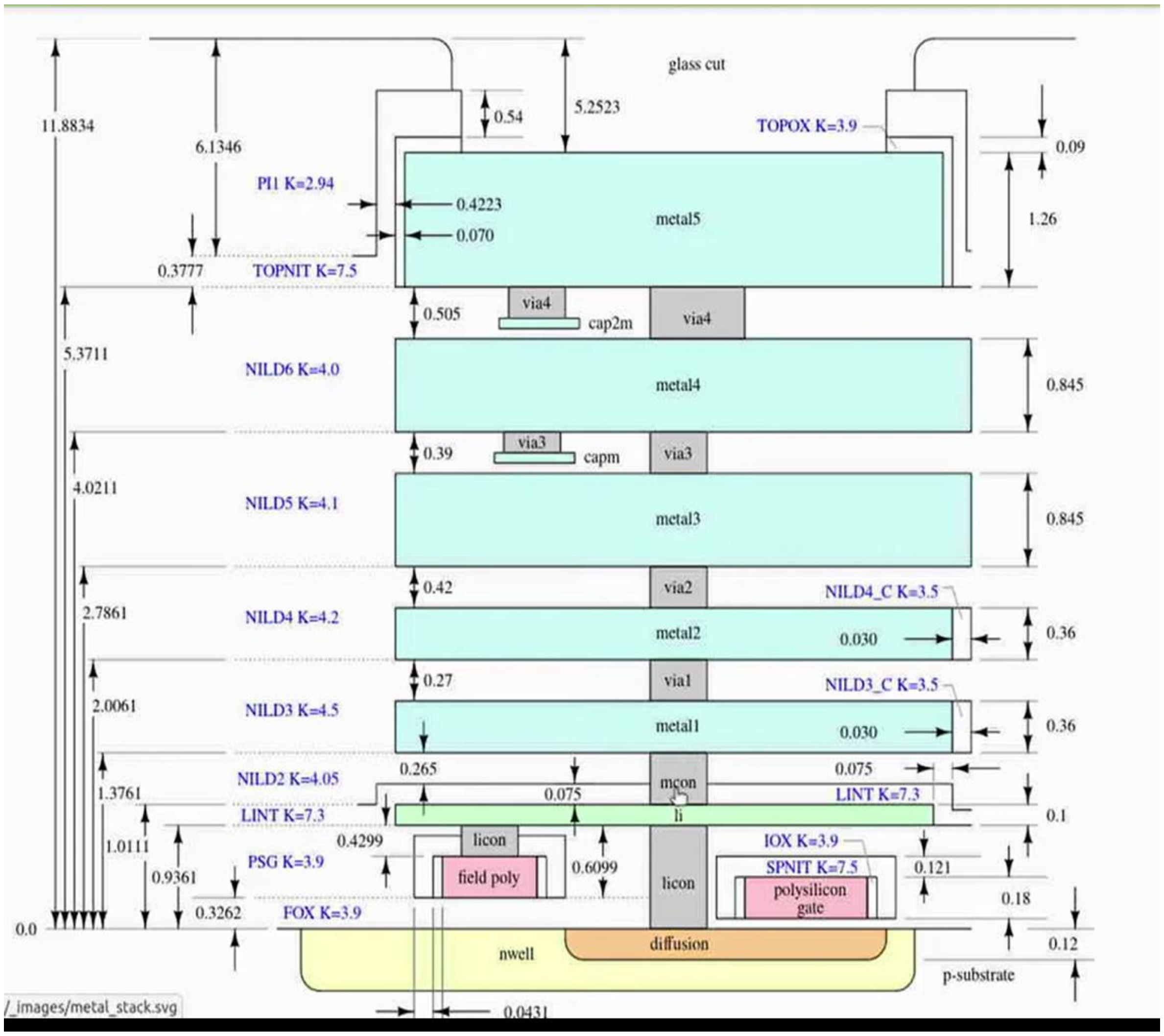
SkyWater PDK

The SkyWater PDK is a collaboration between Google and SkyWater Technology Foundry to provide a fully open source Process Design Kit (PDK), making it possible for the first time, to work off libraries and setups for open source tools build around a real manufacturable process, which can be used to facricate real designs at SkyWater's facility. 130 denoting the minimum feature length (length of the smallest manufacturable transistor in the process) of 130 nanometers.



Sky130 PDK Metal Stack

The metal layers in the Sky130 PDK are structured with progressive thicknesses, starting from the topmost metal layer (metal 5) with a thickness of 1.26 µm, down to the metal 1 layer with a thickness of 0.36 µm. Below the metal 1 layer and above the polysilicon, there is a 0.1 µm layer of titanium nitride (TiN) known as the local interconnect (LI). This stack includes intermediate layers such as metal 4 and metal 3, each with a thickness of 0.845 µm, and metal 2, with a thickness of 0.36 µm. The layers are interconnected by vias(via4,via3, via2, via1, and mcon), which ensure electrical connectivity between the different metal layers.



Introduction to DRC and LVS

Fundamentals of Physical Verification:

Physical verification is a critical process that acts as a safeguard between physical design and potential design failures. Its primary objective is to identify and correct design errors that could impair the functionality or manufacturability of a semiconductor device. The process ensures that the design adheres to the required specifications and can be manufactured without issues. Various tools are employed to confirm and validate the electrical and logical functionality as well as the manufacturability of the design. Physical verification involves several key steps, including:

1. **Design Rule Check (DRC):** This step ensures that the design complies with the manufacturing process rules. It checks for spacing, width, and other geometric constraints to prevent manufacturing defects.
2. **Layout vs. Schematic (LVS):** LVS verifies that the layout matches the schematic design at the netlist level, ensuring that the intended electrical connectivity is maintained.

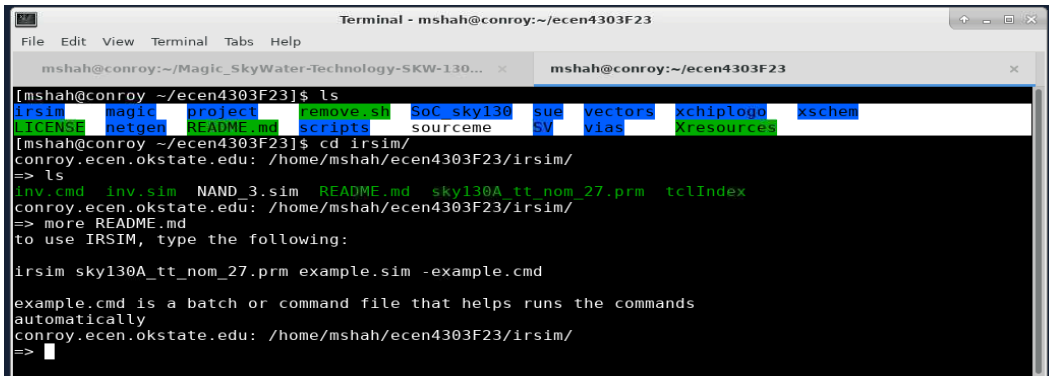
3. **Design for Manufacturability (DFM):** DFM checks optimize the design for manufacturing efficiency and yield, addressing potential manufacturing issues early in the design process.
4. **Antenna Checks:** These checks prevent damage to the transistors during the manufacturing process by identifying and mitigating antenna effects.
5. **Electrical Rule Check (ERC):** ERC verifies that the electrical characteristics of the design, such as voltage and current limits, are within acceptable ranges.

Among these steps, the two most crucial are:

1. **Design Rule Check (DRC):** Ensures geometric compliance with manufacturing constraints.
2. **Layout vs. Schematic (LVS):** Ensures that the physical layout correctly implements the schematic design.

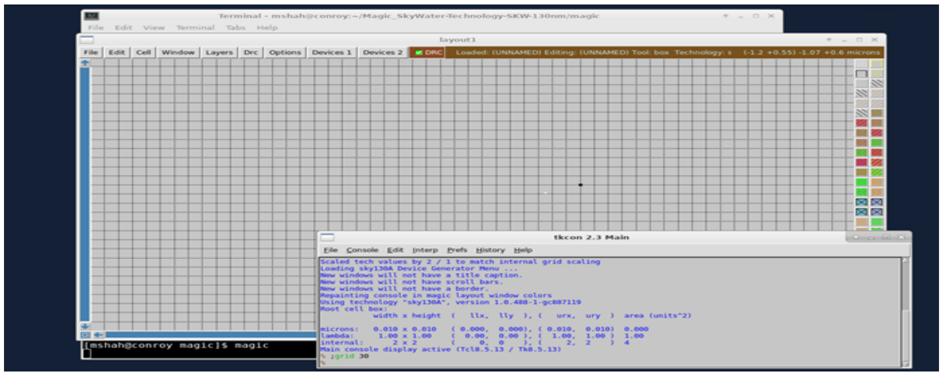
Introduction to Essential Directories and Scripts

Before commencing the design process, we will provide a concise overview of the essential directories and scripts that you need to be acquainted with.


Terminal - mshah@conroy:~/ecen4303F23
File Edit View Terminal Tabs Help
mshah@conroy:~/Magic_SkyWater-Technology-SKW-130...
mshah@conroy:~/ecen4303F23

```
[mshah@conroy ~/ecen4303F23]$ ls
irsim    magic   project    remove.sh  SoC_sky130  sue  vectors  xchiplogo   xschem
LICENSE  netgen  README.md  scripts    sourceme    SV   vias     Xresources
[mshah@conroy ~/ecen4303F23]$ cd irsim/
conroy.ecen.okstate.edu: /home/mshah/ecen4303F23/irsim/
=> ls
inv.cmd  inv.sim  NAND_3.sim  README.md  sky130A_tt_nom_27.prm  tclIndex
conroy.ecen.okstate.edu: /home/mshah/ecen4303F23/irsim/
=> more README.md
to use IRSIM, type the following:

irsim sky130A_tt_nom_27.prm example.sim -example.cmd

example.cmd is a batch or command file that helps runs the commands
automatically
conroy.ecen.okstate.edu: /home/mshah/ecen4303F23/irsim/
=> 
```

❖ **`Magic`** is a versatile VLSI layout tool that provides a user-friendly interface for designing and verifying integrated circuits. When you open Magic, you typically see two main components: a layout window and a console window. Here's a detailed explanation of these components:


Terminal - mshah@conroy:~/Magic_SkyWater-Technology-SKW-130nm/magic
File Edit View Terminal Tabs Help
layout1
File Edit Cell Window Layers Drc Options Devices 1 Devices 2 DRC
Loaded: (UNNAMED) Editing: (UNNAMED) Tool: box Technology: s (-1.2 +0.55) -1.07 +0.6 microns
tkcon 2.3 Main
File Console Edit Interp Prefs History Help

```
[mshah@conroy magic]$ magic
```

```
Scaled tech values by 2 / 1 to match internal grid scaling
Loading sky130A Device Generator Menu ...
New windows will not have a title caption.
New windows will not have scroll bars.
New windows will not have a border.
Repainting console in magic layout window colors
Using technology "sky130A", version 1.0.488-1-gc887119
Root cell box:
          width x height  (   llx,  lly  ), (   urx,  ury  )  area (units^2)

microns:   0.010 x 0.010   ( 0.000,  0.000), ( 0.010,  0.010)  0.000
lambda:     1.00 x 1.00    (  0.00,  0.00 ), (  1.00,  1.00 )  1.00
internal:      2 x 2       (     0,  0    ), (     2,  2    )  4
Main console display active (Tcl8.5.13 / Tk8.5.13)
% ;grid 30
%
```

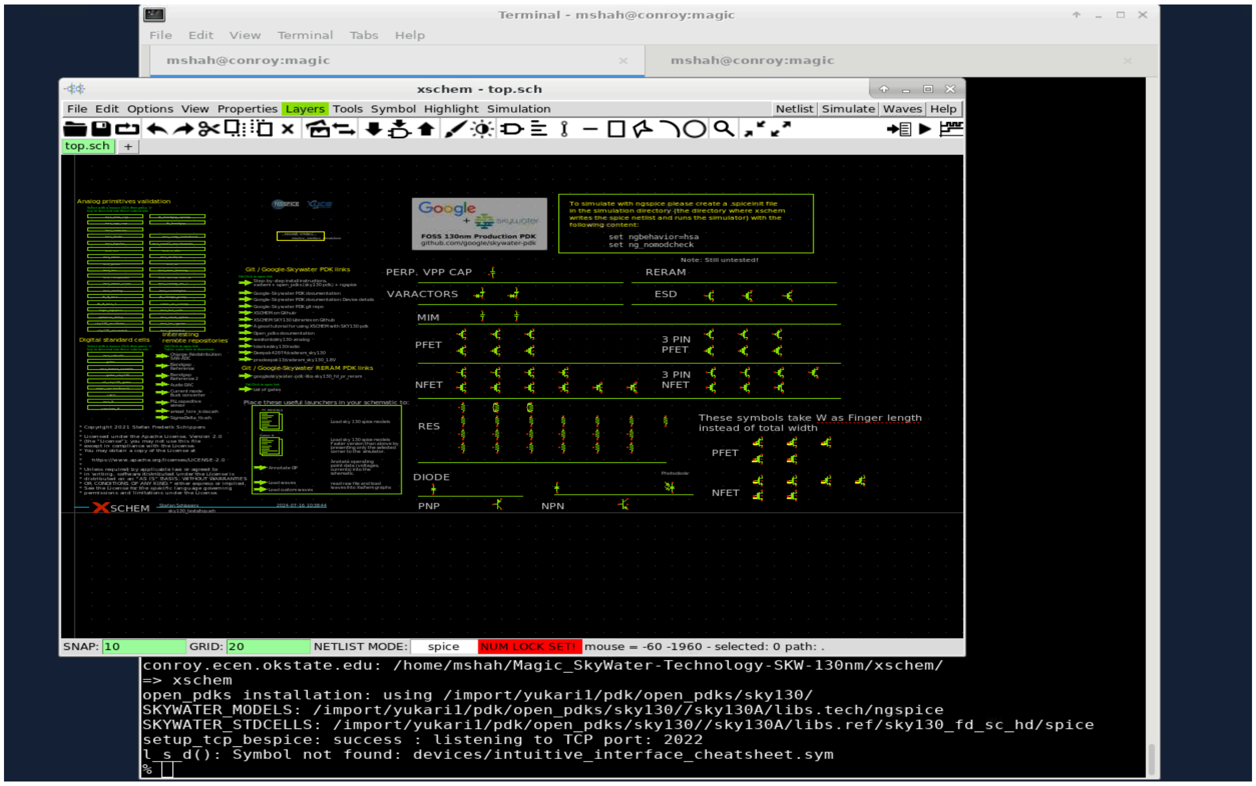
1. **Graphical Window (Layout Window)**

The layout window is where you create and edit the physical design of your integrated circuits. It displays the different layers of the chip, such as diffusion, polysilicon, metal layers, and more.

2. **Console Window (command window)**

The console window provides a command-line interface for interacting with Magic. It allows you to input commands, run scripts, and receive feedback or error messages. The console window may use a special command-line interface called "**tkcon**" (short for Tcl/Tk Console) for executing Tcl/Tk commands and displaying their output. This window offers more features compared to a simple terminal window.

- **xschem**
  is a powerful schematic capture tool used in conjunction with Magic for designing integrated circuits. It provides a graphical user interface (GUI) for creating and editing circuit schematics. The GUI allows you to create and edit schematics visually. You can place components, draw wires, and define circuit connections. xschem includes a comprehensive library of electronic components that you can use in your schematics.


Terminal - mshah@conroy:magic
File Edit View Terminal Tabs Help
mshah@conroy:magic
mshah@conroy:magic
xschem - top.sch
File Edit Options View Properties Layers Tools Symbol Highlight Simulation
Netlist Simulate Waves Help
top.sch
PERP. VPP CAP
VARACTORS
MIM
PFET
NFET
RES
DIODE
PNP
NPN
RERAM
ESD
3 PIN PFET
3 PIN NFET
These symbols take W as Finger length instead of total width
PFET
NFET
FOSS 130nm Production PDK
XSCHEM

```
SNAP: 10   GRID: 20   NETLIST MODE: spice  NUM LOCK SET!  mouse = -60 -1960 - selected: 0 path: .
conroy.ecen.okstate.edu: /home/mshah/Magic_SkyWater-Technology-SKW-130nm/xschem/
=> xschem
open_pdks installation: using /import/yukari1/pdk/open_pdks/sky130/
SKYWATER_MODELS: /import/yukari1/pdk/open_pdks/sky130//sky130A/libs.tech/ngspice
SKYWATER_STDCELLS: /import/yukari1/pdk/open_pdks/sky130//sky130A/libs.ref/sky130_fd_sc_hd/spice
setup_tcp_bespice: success : listening to TCP port: 2022
l_s_d(): Symbol not found: devices/intuitive_interface_cheatsheet.sym
%
```

- **`irsim`** is used for simulating digital circuits at the switch level, where precise modeling of timing and delays based on resistance and capacitance parameters is crucial. This tool is particularly employed in the early stages of digital circuit design and verification.

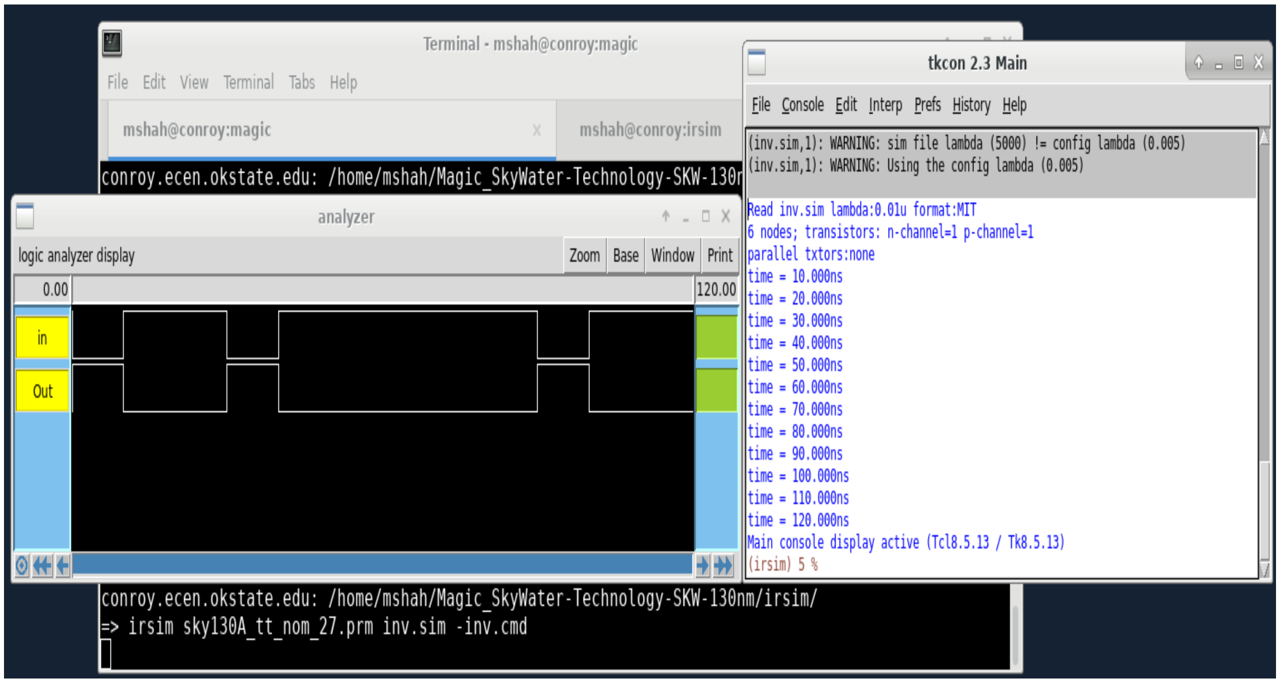
This guide explains how to manipulate a C program to increase the size of the test and ensure the `project.cmd` passes. Running the `create_test.sh` script will create a new `project.cmd`. Please follow the instructions carefully.

Note: You may need to change `project_template.cmd` and/or the C file to ensure your input/output signals match.

Procedure

1. Modify `bit_test.c` to change the loops in line 56.
2. Run `./create_test.sh` to compile and generate a new `project.cmd`.
3. Execute the command: `irsim sky130A_tt_nom_27.prm project.sim -project.cmd`.
4. Any errors will be displayed in the IRSIM output.

   When you run this command "`irsim sky130A_tt_nom_27.prm example.sim -example.cmd`" with the parameters specified for your project, a window titled "Analyzer" opens. This window is part of the IRSIM simulation tool, used for analyzing and viewing simulation results. It allows you to observe and analyze your circuit simulation in the form of waveforms. These waveforms help you analyze the behavior of your circuit over time and evaluate its performance.



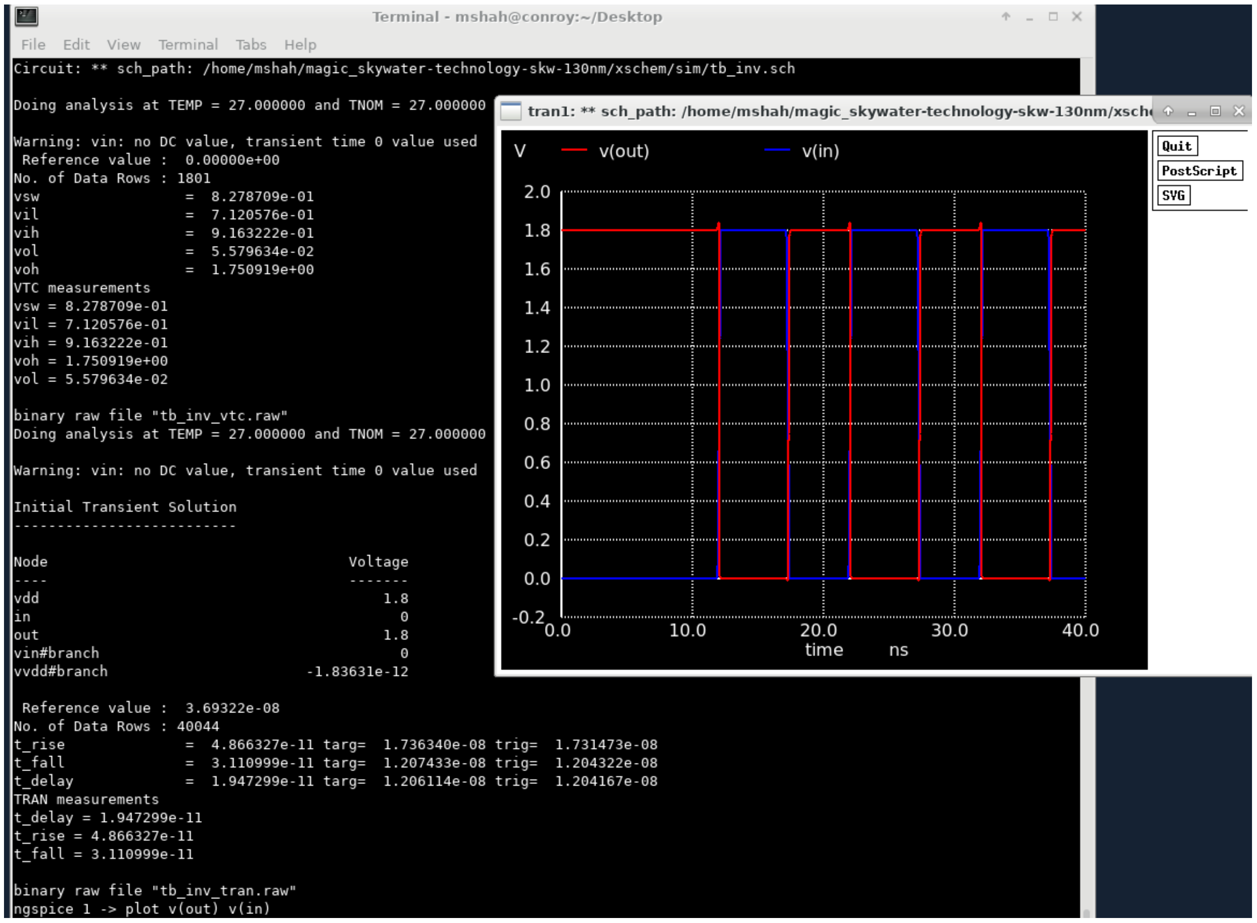
```
conroy.ecen.okstate.edu: /home/mshah/Magic_SkyWater-Technology-SKW-130n
```

```
(inv.sim,1): WARNING: sim file lambda (5000) != config lambda (0.005)
(inv.sim,1): WARNING: Using the config lambda (0.005)
Read inv.sim lambda:0.01u format:MIT
6 nodes; transistors: n-channel=1 p-channel=1
parallel txtors:none
time = 10.000ns
time = 20.000ns
time = 30.000ns
time = 40.000ns
time = 50.000ns
time = 60.000ns
time = 70.000ns
time = 80.000ns
time = 90.000ns
time = 100.000ns
time = 110.000ns
time = 120.000ns
Main console display active (Tcl8.5.13 / Tk8.5.13)
(irsim) 5 %
```

```
conroy.ecen.okstate.edu: /home/mshah/Magic_SkyWater-Technology-SKW-130nm/irsim/
=> irsim sky130A_tt_nom_27.prm inv.sim -inv.cmd
```

- **`ngspice`** is an open-source, mixed-level/mixed-signal electronic circuit simulator. It is based on SPICE (Simulation Program with Integrated Circuit Emphasis) and supports a variety of simulation types, including transient analysis, AC analysis, DC sweep, and more.

you can run this command "`ngspice inv.spice`" "`Plot v(out) v(in)` "


Terminal - mshah@conroy:~/Desktop
File Edit View Terminal Tabs Help
tran1: ** sch_path: /home/mshah/magic_skywater-technology-skw-130nm/xsche
V
v(out)
v(in)
2.0
1.8
1.6
1.4
1.2
1.0
0.8
0.6
0.4
0.2
0.0
-0.2
0.0
10.0
20.0
30.0
40.0
time
ns
Quit
PostScript
SVG

```
Circuit: ** sch_path: /home/mshah/magic_skywater-technology-skw-130nm/xschem/sim/tb_inv.sch

Doing analysis at TEMP = 27.000000 and TNOM = 27.000000

Warning: vin: no DC value, transient time 0 value used
 Reference value :  0.00000e+00
No. of Data Rows : 1801
vsw                 =  8.278709e-01
vil                 =  7.120576e-01
vih                 =  9.163222e-01
vol                 =  5.579634e-02
voh                 =  1.750919e+00
VTC measurements
vsw = 8.278709e-01
vil = 7.120576e-01
vih = 9.163222e-01
voh = 1.750919e+00
vol = 5.579634e-02

binary raw file "tb_inv_vtc.raw"
Doing analysis at TEMP = 27.000000 and TNOM = 27.000000

Warning: vin: no DC value, transient time 0 value used

Initial Transient Solution
--------------------------

Node                                   Voltage
----                                   -------
vdd                                        1.8
in                                           0
out                                        1.8
vin#branch                                   0
vvdd#branch                       -1.83631e-12

 Reference value :  3.69322e-08
No. of Data Rows : 40044
t_rise              =  4.866327e-11 targ=  1.736340e-08 trig=  1.731473e-08
t_fall              =  3.110999e-11 targ=  1.207433e-08 trig=  1.204322e-08
t_delay             =  1.947299e-11 targ=  1.206114e-08 trig=  1.204167e-08
TRAN measurements
t_delay = 1.947299e-11
t_rise = 4.866327e-11
t_fall = 3.110999e-11

binary raw file "tb_inv_tran.raw"
ngspice 1 -> plot v(out) v(in)
```

Another graphical tool for viewing and analyzing waveforms is the " **`gaw`**" tool

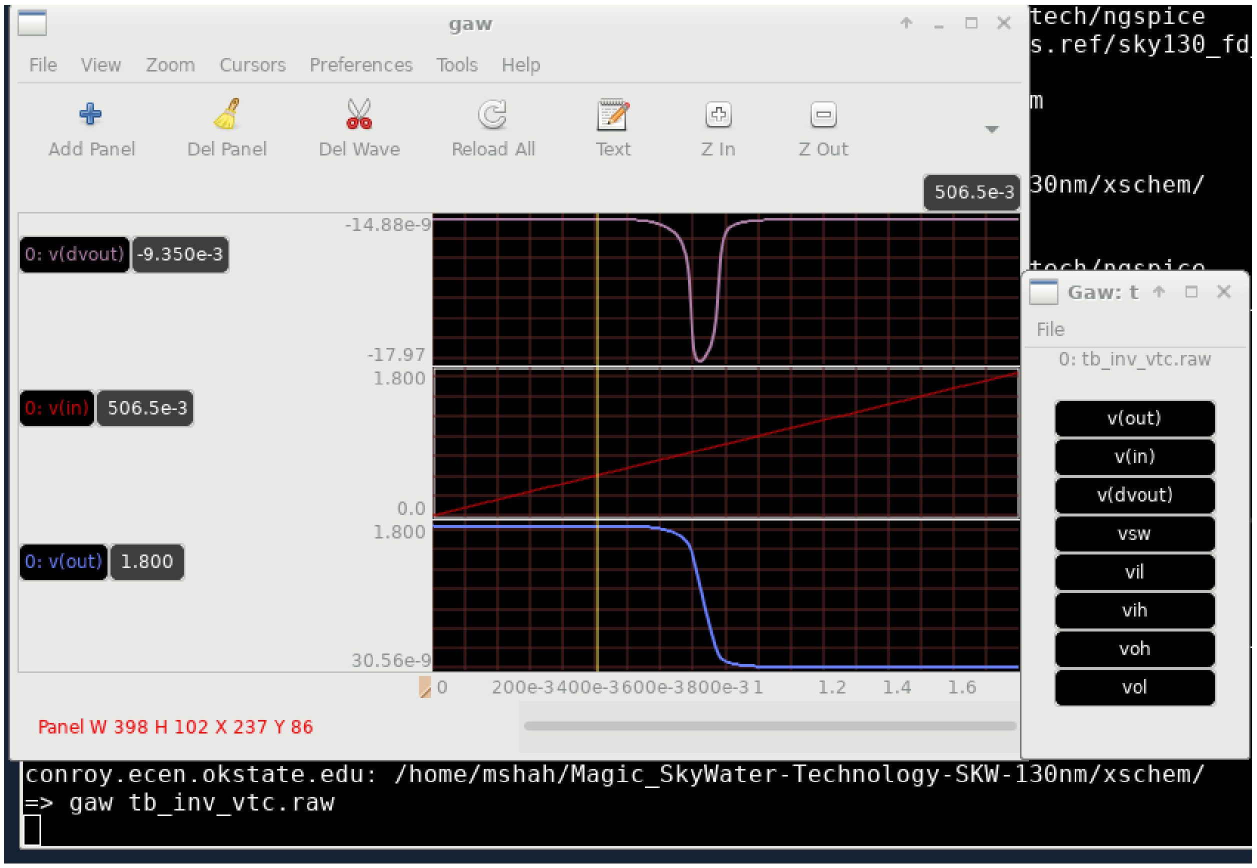
- **`gaw(GTK Analog Waveform Viewer)`**
  Gaw is a Linux software tool for displaying analog waveforms from sampled datas. The Gaw software is licensed under the terms of the GNU General Public License as published by the Free Software Foundation.
  GAW offers unlimited panels, waveforms, and data in a waveform, with features like real-time data capture from a sound card, various export options, drag and drop functionality, grid and labeling display, log scales, measurement cursors, and a powerful zoom facility. The tool supports multiple input/output formats and language localization with gettext.

To simulate the provided files, follow these steps:

1) **Open xschem:** Load the testbench schematic files " xschem tb_inv.sch "

2) **Modify Commands:** Make any necessary modifications to the commands within the schematic.
3) **Generate Netlist:** Click the " Netlist " button in xschem to create the SPICE netlist.
4) **Run Simulation:** Click the "Simulate" button to execute the simulation.

After the simulation completes, you can view the results using GAW: " `gaw tb_inv_vtc.raw`"



```
tech/ngspice
s.ref/sky130_fd
m
30nm/xschem/
tech/ngspice
conroy.ecen.okstate.edu: /home/mshah/Magic_SkyWater-Technology-SKW-130nm/xschem/
=> gaw tb_inv_vtc.raw
```

❖ **The `Netgen`** Netgen is a tool used for comparing netlists and performing LVS (Layout Versus Schematic) simulation, a process that verifies the correspondence between the layout design and the schematic to ensure they are consistent and accurately matched.

" To run netgen, please use the following nomenclature:

```
netgen -batch lvs "../spice/inv.spice inv" "../xschem/inv.spice inv"
$PDK_ROOT/sky130A/libs.tech/netgen/setup.tcl inv.out
```

Or this can be accomplished by running the runNetgen.bash script:

```
./runNetlist.bash inv
```

Keep in mind that the above script utilizes the top-level name in place of the "inv" name shown above if other circuits are desired."

- **The `Sue`** directory likely refers to a schematic design environment or a tool related to electronic circuit design. Sue stands for Schematic User Environment and provides a user interface for designing electronic circuit schematics.

  .**sim files** are used for circuit simulations.

  .**sue files** are schematic files.

  .**sue.BAK** files are backup versions of the schematics.

  .**sp and .sp.cache** files are SPICE netlist files.

  Other files include simulation commands, technology parameters, Tcl indices, and simulation outputs.

- **`XChipLogo`: Creating Chip Graffiti**

To add some chip graffiti to your design, follow the procedure below:

1. **Copy Files:** Copy the necessary files to your directory.
2. **Convert Image:** Use the command `convert -compress none file.png file.pbm` to convert your PNG file to a PBM file.
3. **Generate MAG File:** Run the command `xchiplogo file.pbm file.mag` to create the MAG file from your PBM file.
4. **Insert Graffiti:** Insert your new graffiti into your final design. Ensure it is not connected to any other layout.

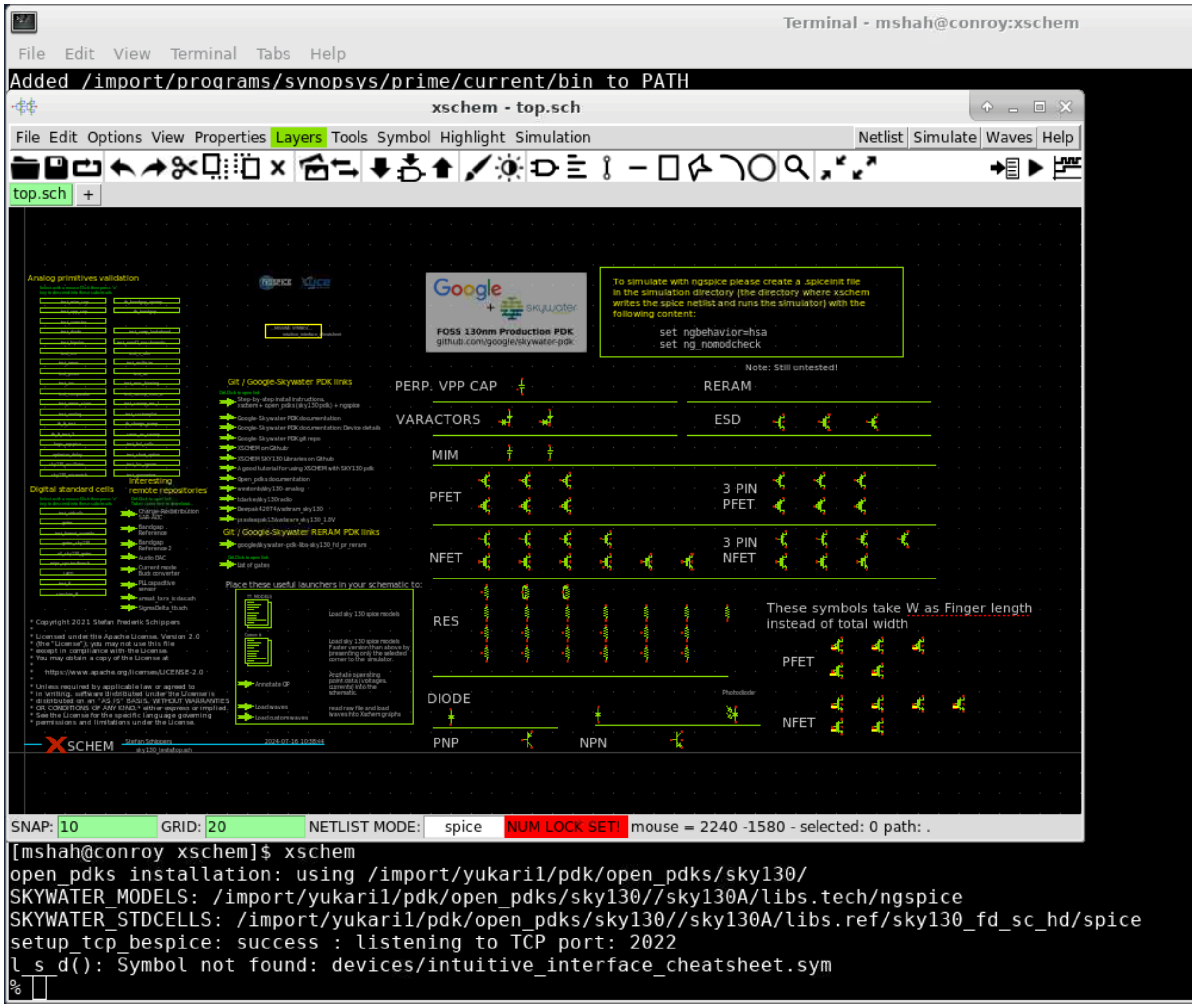
```
cp /home/user/images/logo.png /home/user/project/

cd /home/user/project/

convert -compress none file.png file.pbm

xchiplogo file.pbm file.mag

getcell file.mag
```

By following these steps, you can convert an image (such as a logo or graffiti) into a Magic layout file and add it to your final design. This image will appear as a decoration in your design and should not affect the circuit's functionality.

Note: You can achieve similar functionality in both Magic and Netgen by appending " `noconsole` " to their launch commands. Additionally, you can run Magic without a graphical interface by using "`magic -dnull -noconsole`" a method often employed when executing Magic through a script.

Schematic Drawing

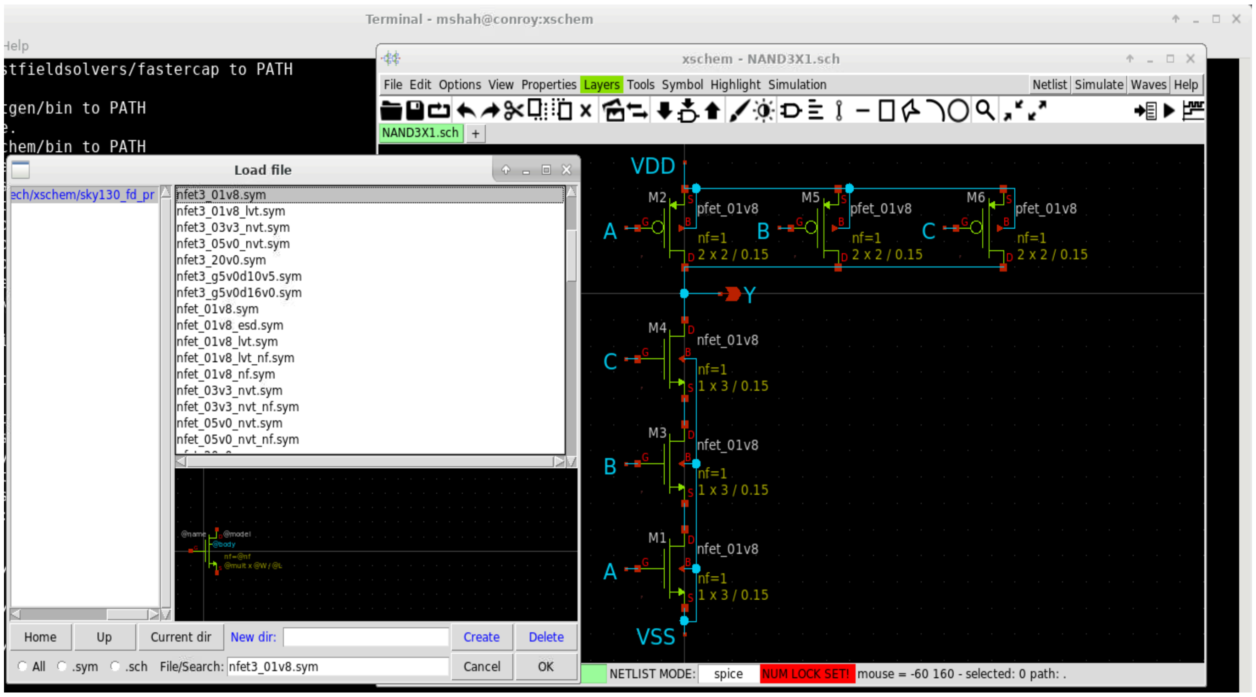
To run xschem, type xschem in the terminal and hit enter. This will open a GUI however unlike magic and netgen, the tcl command line is in the terminal itself. you will now see this:


Terminal - mshah@conroy:xschem
File Edit View Terminal Tabs Help
xschem - top.sch
File Edit Options View Properties Layers Tools Symbol Highlight Simulation
Netlist Simulate Waves Help
top.sch
PERP. VPP CAP
RERAM
VARACTORS
ESD
MIM
PFET
3 PIN PFET
NFET
3 PIN NFET
RES
These symbols take W as Finger length instead of total width
PFET
DIODE
NFET
PNP
NPN
SNAP: 10
GRID: 20
NETLIST MODE: spice
NUM LOCK SET!
mouse = 2240 -1580 - selected: 0 path: .

```
Added /import/programs/synopsys/prime/current/bin to PATH
[mshah@conroy xschem]$ xschem
open_pdks installation: using /import/yukari1/pdk/open_pdks/sky130/
SKYWATER_MODELS: /import/yukari1/pdk/open_pdks/sky130//sky130A/libs.tech/ngspice
SKYWATER_STDCELLS: /import/yukari1/pdk/open_pdks/sky130//sky130A/libs.ref/sky130_fd_sc_hd/spice
setup_tcp_bespice: success : listening to TCP port: 2022
l_s_d(): Symbol not found: devices/intuitive_interface_cheatsheet.sym
%
```

Go to the File menu and select the option to open a new schematic. This will create a blank schematic canvas where you can start building your circuit.(for example: NAND with 3 input) Press the Insert key on your keyboard. This action will allow you to add new components to your schematic. Navigate to the sky130 Directory. Within the sky130 directory, find and select the sky130_fd_pr option. This library includes primitive cells, which are basic building blocks for your circuit.

You should choose the nfet3 and pfet3, both with 01v8.sym types for demonstration purposes. Go to the xschem library and then to devices. Here you need a ipin, opin and an iopin. You can use the “w” button and mouse cursor to connect all wirings and use the “q” key to change names of the labels.


Terminal - mshah@conroy:xschem
xschem - NAND3X1.sch
File Edit Options View Properties Layers Tools Symbol Highlight Simulation
Netlist Simulate Waves Help
NAND3X1.sch
Load file
VDD
M2
M5
M6
pfet_01v8
A
B
C
nf=1
2 x 2 / 0.15
Y
M4
M3
M1
nfet_01v8
1 x 3 / 0.15
VSS
Home Up Current dir New dir: Create Delete
All .sym .sch File/Search: nfet3_01v8.sym Cancel OK
NETLIST MODE: spice NUM LOCK SET! mouse = -60 160 - selected: 0 path: .

```
Help
tfieldsolvers/fastercap to PATH

gen/bin to PATH
e.
hem/bin to PATH
```

```
ech/xschem/sky130_fd_pr
nfet3_01v8.sym
nfet3_01v8_lvt.sym
nfet3_03v3_nvt.sym
nfet3_05v0_nvt.sym
nfet3_20v0.sym
nfet3_g5v0d10v5.sym
nfet3_g5v0d16v0.sym
nfet_01v8.sym
nfet_01v8_esd.sym
nfet_01v8_lvt.sym
nfet_01v8_lvt_nf.sym
nfet_01v8_nf.sym
nfet_03v3_nvt.sym
nfet_03v3_nvt_nf.sym
nfet_05v0_nvt.sym
nfet_05v0_nvt_nf.sym
```

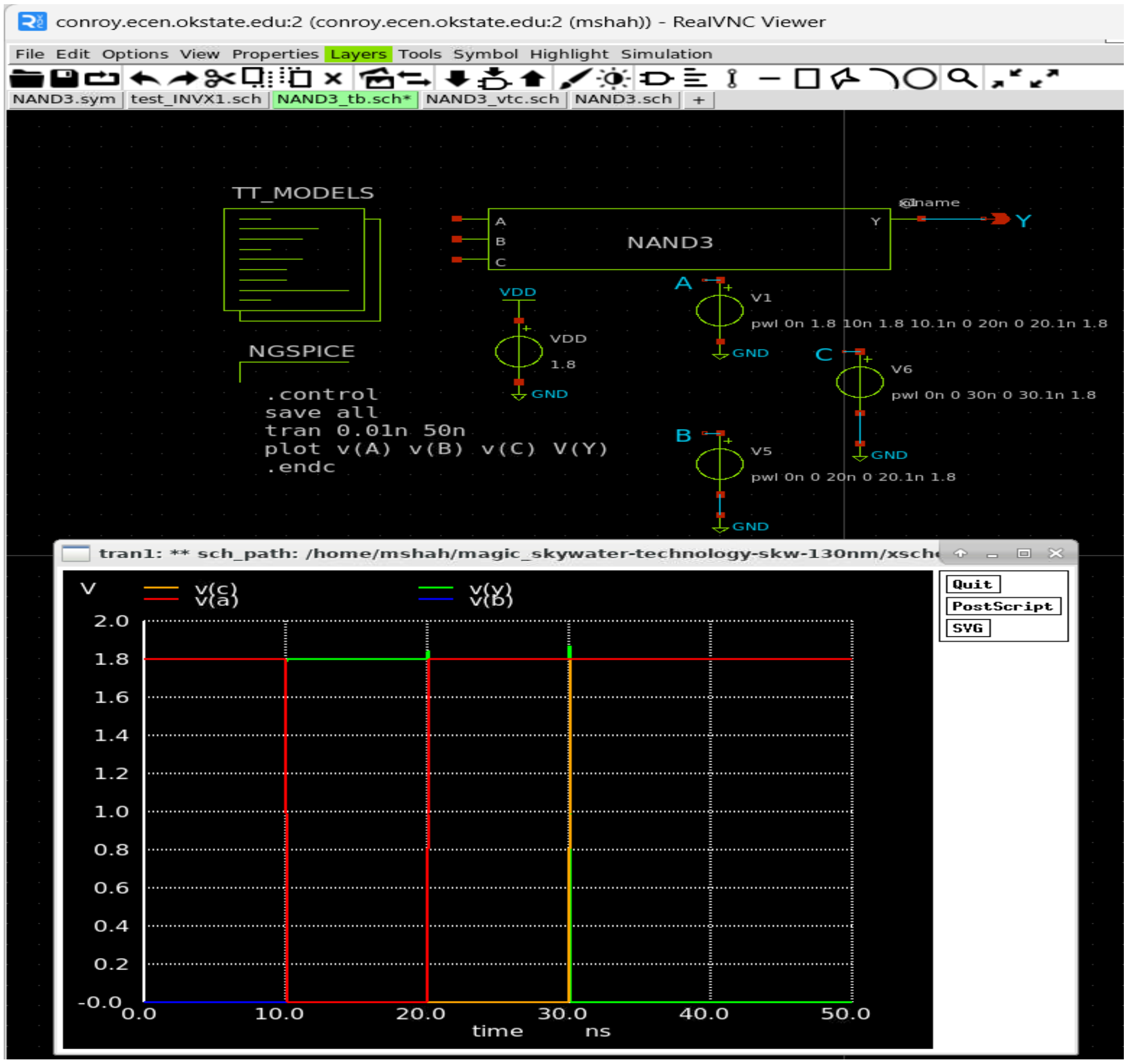
When all the components are in place and the connections are properly made, the schematic is now ready. “ Go to file --> "Save as" and save as NAND.sch and click on OK. ”

The goal is to create a layout for this circuit. First, it must be functionally validated by creating an independent testbench using the schematic as a symbol. To do this, go to the symbol menu and click on “Make symbol from schematic”. Next, start your testbench schematic by opening a new schematic from the File menu and select “NAND.sym”.

The parameters need to be added. Go to Vdd and set the value to 1.8. Go to V2, V3, V4 and set the value to a PWL sweep as shown. Press insert again and search and add TT_MODELS that will tell ngspice where to find the device models for the devices used in the schematic and add code_shown.sym will run a transient analysis and plot the values for Vin and Vout.

Save the file again as NAND_tb.sch and then tap netlist, then, run the ngspice simulation to ensure there are no errors in your setup. This step takes a while so be patient.

- **Analyze Results**: Review the output plots for Vin and Vout to verify the expected behavior. Check other critical nodes and currents as needed.
- **Adjust Parameters**: If the simulation results do not meet expectations, adjust the parameters or model files accordingly and rerun the simulation.


conroy.ecen.okstate.edu:2 (conroy.ecen.okstate.edu:2 (mshah)) - RealVNC Viewer
File Edit Options View Properties Layers Tools Symbol Highlight Simulation
NAND3.sym | test_INVX1.sch | NAND3_tb.sch* | NAND3_vtc.sch | NAND3.sch | +
TT_MODELS
NGSPICE
NAND3
A
B
C
Y
VDD
1.8
GND
V1
pwl 0n 1.8 10n 1.8 10.1n 0 20n 0 20.1n 1.8
V6
pwl 0n 0 30n 0 30.1n 1.8
V5
pwl 0n 0 20n 0 20.1n 1.8
tran1: ** sch_path: /home/mshah/magic_skywater-technology-skw-130nm/xsche
Quit
PostScript
SVG
v(c)
v(a)
v(y)
v(b)
V
time
ns

```
.control
save all
tran 0.01n 50n
plot v(A) v(B) v(C) V(Y)
.endc
```

After confirming the functionality of the NAND schematic, generate a netlist exclusively for the NAND, excluding the entire testbench. Open the NAND from the files menu, navigate to the simulation menu, select “LVS netlist: top level is a subckt” and tap netlist. Finally, exit xschem.

MAGIC Starting Magic (layout) design

In the Magic software, there are several keys and shortcuts designed to increase user efficiency and speed. Below are some of the most important ones:

Mouse Controls:

- **Left Click to Relocate the Box:**
    - to move the white box to a different location.
- **Right Click to Resize the Box:**
    - This allows you to adjust the size of the box as desired.
- **Middle mouse button:**
    - Hover your cursor over one of the colour tabs on the right and either press your middle mouse button to fill the area within the white box with the corresponding colour.

Keyboard Shortcuts:

- **z:** Zoom In to the area selected by the mouse.
- **Shift + z:** Zoom Out from the current area.
- **a:** Select Area
- **d:** delete
- **r:** Move Right
- **q:** Move Left
- **e:** Move up
- **w:** Move Down
- **u:** Undo
- **Shift + u**: Redo
- **x:** expands a selected cell, all details are visible.
- **Shift + x**: un-expands a selected cell, all details are hidden.
- **s**:
    - **Single Press**: Press the s key to select or highlight an element (such as a wire, contact, or diffusion layer) in your design.
    - **Multiple Presses**: Press the s key multiple times to cycle through and highlight all connected elements. Each press will expand the selection to include more connected elements.

The 3-input NAND gate consists of three NMOS transistors and three PMOS transistors. The target fabrication technology is the SkyWater Technology (SKW) 130nm CMOS process, so the channel length (L) of both devices is set to the minimum allowed by the technology: 0.15µm. The channel widths (W) of the NMOS and PMOS devices are set to 0.46µm and 0.58µm, respectively.The sizing of each transistor allows designers to customize the current drive for each gate based on fan-in, fan-out, and other influential factors. Note that the source and the body (p-substrate) of the NMOS transistors are connected to ground (GND node), while the source and the body (n-well) of the PMOS transistors are connected to the positive supply (VDD node), even though these connections are not shown in the diagram.

Start Magic from the Unix prompt:

prompt> `magic NAND`

In the toolbar, go to Window and select “Set Grid to 0.05um.” Or

```
:grid 30
```

➢ Creating the Gate of a Transistor with Polysilicon

Start by painting the polysilicon layer for the transistor gate. (The order in which layers are placed in the graphics window is not important, and so the layout steps described here are by no means unique). With the channel width (W) and length (L) specified, create a box of 0.15µm x 1µm using the mouse. The length of the box is not important at this stage since it will be adjusted later.. Once the box is correctly sized, use the command:

```
:paint poly
```

➢ Creating diffusion for the n/p active regions

The dimensions of these diffusion layers can change depending on what drive strength you want these transistors to have. The wider the diffusion layer is, the more drive strength the transistor will have. This also affects the size of the diffusion contacts that connect the local interconnect metal layer to the diffusion layer. Larger contacts will be needed to drive transistors with larger widths. **The P-type transistor is usually 1.5 to 2 times the size of the N-type transistor because PMOS transistors have lower carrier mobility compared to NMOS transistors. This means that PMOS transistors need to be larger to provide the same drive current as NMOS transistors.**

```
:paint pdiff
```

```
:paint ndiff
```

Please, make sure that the poly hangs over the diffusion by at least 0.13um or else it will cause a DRC.

➢ Creating Contacts within the Diffusion:

Place contacts to connect diffusion layers to the local interconnect. Minimum size for contacts is 0.17µm x 0.17µm. Adjust diffusion and interconnect as needed to avoid DRC errors.

- **Commands**:

  For N diffusion: `:paint ndc`

  For P diffusion: `:paint pdc`

  Paint local interconnect: `:paint li`

Connecting PMOS and NMOS: Connect the source of PMOS to the drain of NMOS using local interconnect to form the output of the NAND. Verify connections with the s key.

➢ Creating Poly Contacts:

Place a contact to connect local interconnect to poly. Ensure sufficient poly and local interconnect on both sides. Use the following commands:

Paint poly contact: `:paint pc`
Paint local interconnect: `:paint li`

➢ Placing Power and Ground Rails:

Place local interconnect above the PMOS for power and below the NMOS for ground.

➢ Substrate Contacts:

- Place N-substrate contacts on the power (VDD) rail and P-substrate contacts on the ground rail.
- Paint substrate diffusion layers under these contacts:

For N-substrate diffusion: `:paint nsd`
For N-substrate contact: `:paint nsc`
For P-substrate diffusion: `:paint psd`
For P-substrate contact: `:paint psc`

- Ensure diffusion layers cover the contact area to avoid DRC errors.
- Typically, place as many contacts as there are transistors, plus one.
- If you need to delete something, you can use the following command in Magic:
  `:erase [object]`
  Replace [object] with the specific item you want to delete, such as a layer, cell, or any other element in your design.

➢ Creating Labels:

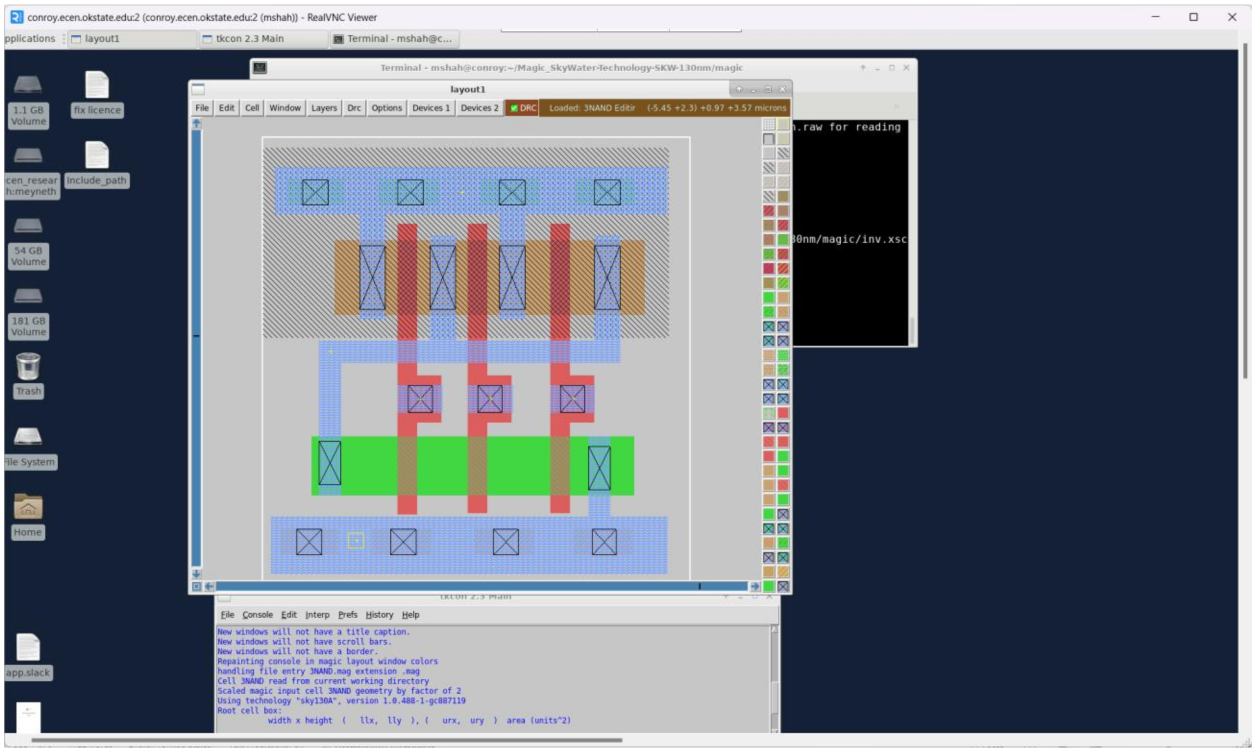
- Label inputs and outputs for LVS verification:

  ➢ Create a label by clicking the left-mouse button followed by the right-mouse button to make a " +" symbol.
  ➢ Use the following commands to label the inputs on the poly:
  ```
  :label A
  :label B
  :label C
  ```
  ➢ Label the output on the local interconnect as "Y":
  ```
  :label Y
  ```

- Ensure all labels match the names used in your schematic.

Saving and Exiting:

- Save the layout: `:save`
- Exit Magic: `:quit`



```
New windows will not have a title caption.
New windows will not have scroll bars.
New windows will not have a border.
Repainting console in magic layout window colors
handling file entry 3NAND.mag extension .mag
Cell 3NAND read from current working directory
Scaled magic input cell 3NAND geometry by factor of 2
Using technology "sky130A", version 1.0.488-1-gc887119
Root cell box:
           width x height  (   llx,  lly  ), (   urx,  ury  )  area (units^2)
```

The next step in the design flow process will be to use following commands to extract ".gds", ".ext", ".sim", ".sp", and ".spice" files. Scripting is very important to Electronic Design Automation Tools. First, navigate to File --> Save and select Autowrite. The process is not complete yet. Open the command window and type the following commands:

```
extract do local
```

```
extract all
```

The command `extract do local` instructs to perform all extractions in the local directory, and `extract all` executes the actual extraction. To ensure the extraction is in the spice format for LVS, run the following commands:

```
ext2spice lvs
```

```
ext2spice
```

```
ext2sim labels on
```

```
ext2sim -R -C

ext2spice scale off

ext2spice -F -R -C -f spice3

calma

| magic -dnull -noconsole

quit
```

Ready to run LVS, move to the netgen folder and execute the following command:

```
netgen  -batch  lvs  "../mag/NAND.spice  NAND"  "../xschem/NAND.spice  NAND"
$PDK_ROOT/sky130A/libs.tech/netgen/setup.tcl inv.out
```

- **Some other options for extraction are:**

1. Parasitic capacitance extraction

```
ext2spice cthresh <threshold>
```

2. Resistance extraction

```
ext2sim labels on

ext2sim

extresist tolerance 10

extresist
```

3. RX Extraction

```
ext2spice lvs

ext2spice cthresh 0.01

ext2spice extresist on

ext2spice
```

Concept of Cells in VLSI Design

Cells are fundamental building blocks in the layout of VLSI (Very-Large-Scale Integration) circuits. Using cells allows for an efficient and organized layout by reusing existing designs that is a powerful technique for managing complex circuits efficiently.When you design a basic component like an inverter, you can save this design as a cell. This cell can then be reused in other designs or multiple times within the same design. Each cell can be part of a larger design, creating a hierarchical structure. Cells can be stored in libraries, either created by you or provided by others, and these cells can be included in new designs.

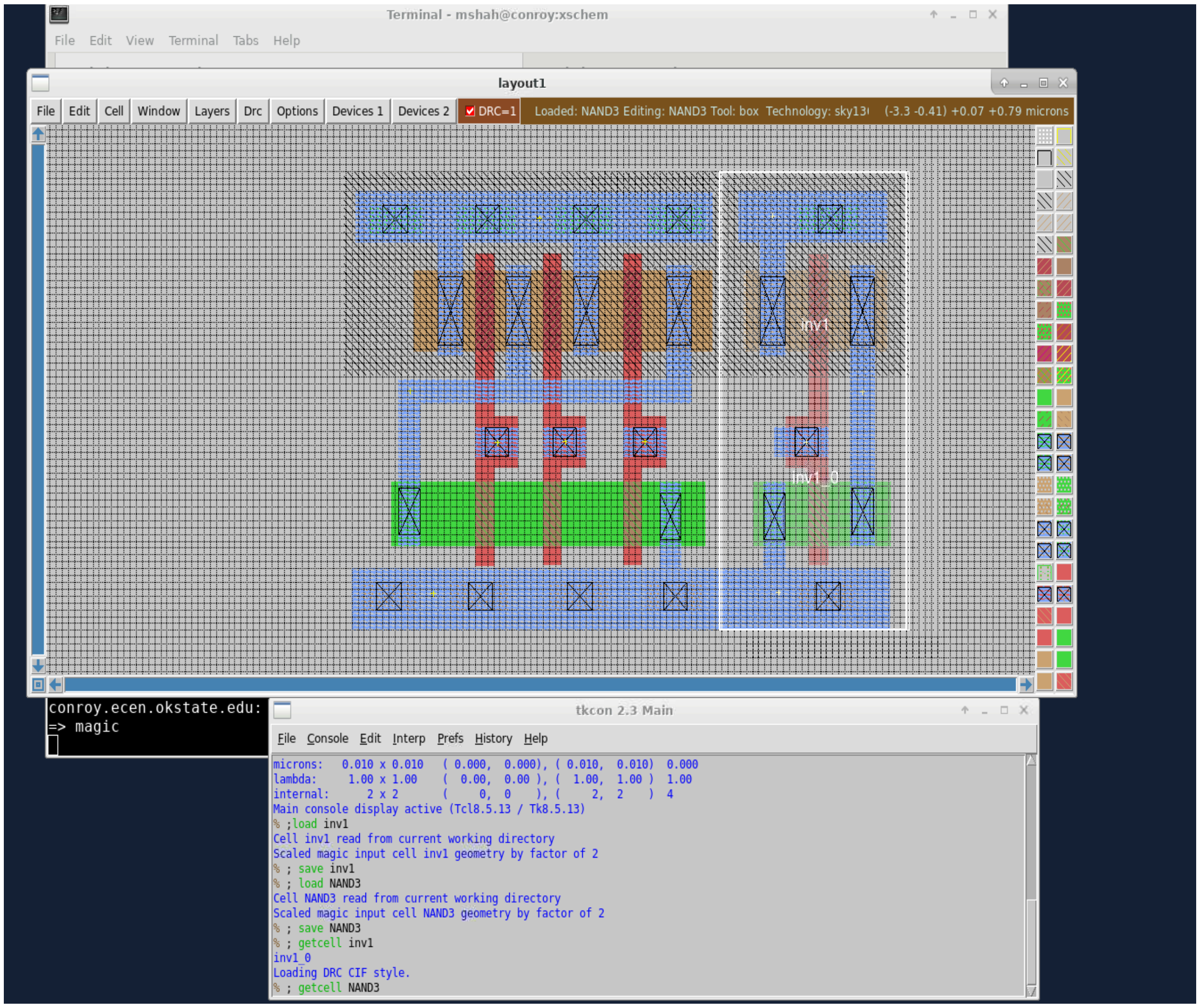
Every time a circuit is created and saved in Magic, it becomes a cell. These cells can be incorporated into the hierarchy of new circuits. By creating reusable building blocks, designers can streamline the design process, maintain consistency, and manage large designs more effectively.

Let's assume you have previously created and saved the layouts for a NAND gate and an inverter in Magic. You can now use these cells to construct an AND gate efficiently by leveraging the hierarchical design.

Load the previously created NAND and inverter cells into your current design environment. Position the NAND cell in your design to serve as the base of the AND gate. Place the inverter cell to invert the output of the NAND gate, thus creating an AND gate.

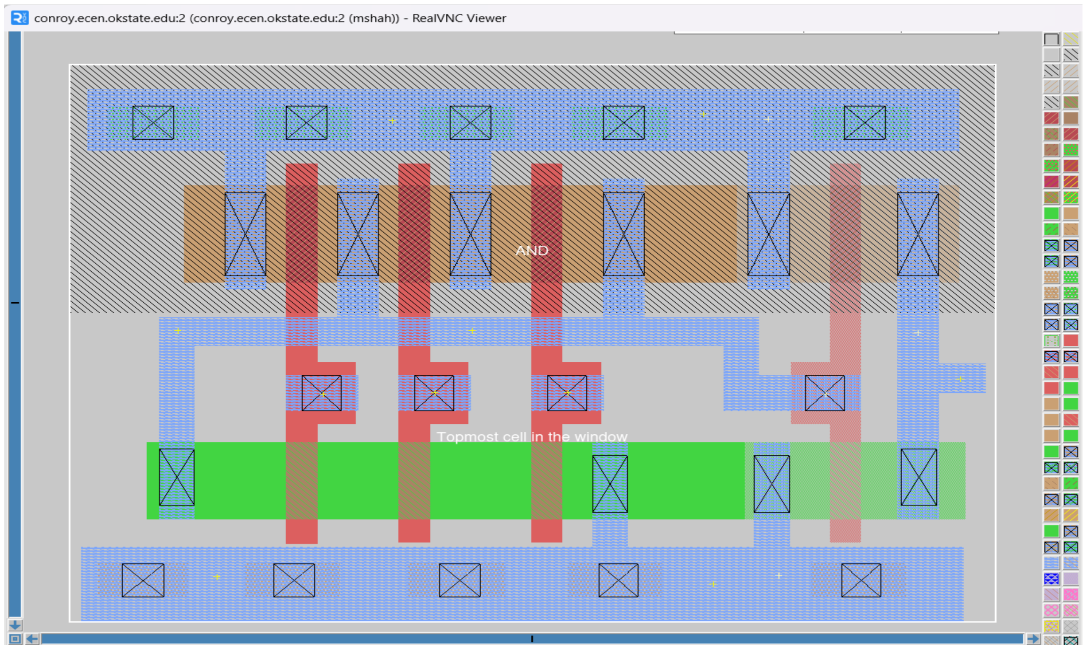
```
:load NAND_gate

:save NAND_cell

:load inverter

:save inverter_cell

:getcell NAND_cell

:getcell inverter_cell
```

Press the 'x' key to expand the NAND cell, revealing all its details and connection points for the inverter.


Terminal - mshah@conroy:xschem
File Edit View Terminal Tabs Help
layout1
File Edit Cell Window Layers Drc Options Devices 1 Devices 2
DRC=1
Loaded: NAND3 Editing: NAND3 Tool: box Technology: sky130 (-3.3 -0.41) +0.07 +0.79 microns
inv1
inv1_0
tkcon 2.3 Main
File Console Edit Interp Prefs History Help

```
conroy.ecen.okstate.edu:
=> magic
```

```
microns:   0.010 x 0.010   ( 0.000,  0.000), ( 0.010,  0.010)  0.000
lambda:     1.00 x 1.00    (  0.00,  0.00 ), (  1.00,  1.00 )  1.00
internal:      2 x 2       (     0,  0    ), (     2,  2    )  4
Main console display active (Tcl8.5.13 / Tk8.5.13)
% ;load inv1
Cell inv1 read from current working directory
Scaled magic input cell inv1 geometry by factor of 2
% ; save inv1
% ; load NAND3
Cell NAND3 read from current working directory
Scaled magic input cell NAND3 geometry by factor of 2
% ; save NAND3
% ; getcell inv1
inv1_0
Loading DRC CIF style.
% ; getcell NAND3
```


conroy.ecen.okstate.edu:2 (conroy.ecen.okstate.edu:2 (mshah)) - RealVNC Viewer
AND
Topmost cell in the window

Connect the output of the NAND gate to the input of the inverter using the local interconnect layer. Ensure that all connections are correctly made and labeled appropriately for LVS verification.

Connect the output of the NAND gate to the input of the inverter using the local interconnect layer.

```
:paint li
```

Label the output of the inverter (which is the output of the AND gate).

```
:label Y
```

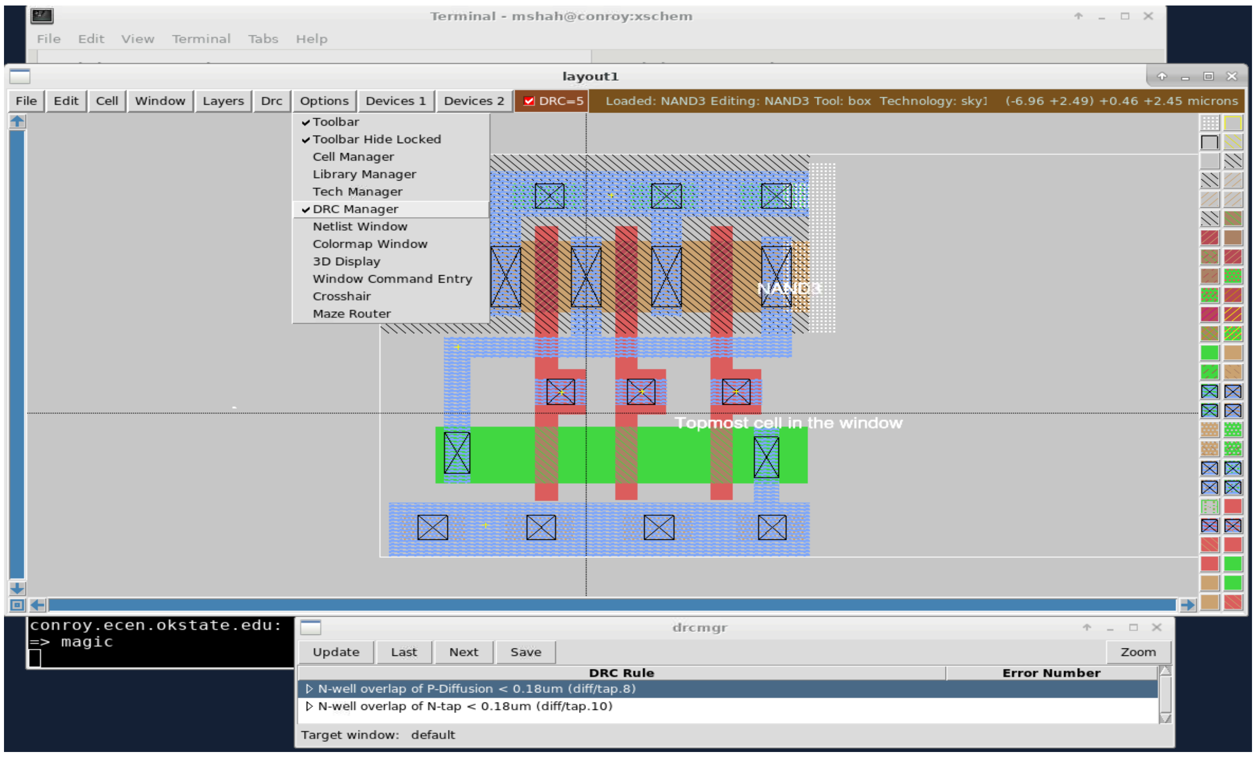
```
:save AND_gate
```

```
:quit
```

If a DRC error occurs, navigate to the options menu and select DRC Manager > Zoom. The error details will be displayed, allowing you to identify the location and resolve the issue effectively. You can also use the following command to understand the reason for the error:

```
:drc why
```

Magic has a veature in that it runs an idle DRC engine process which allows for interactive DRC. It shows this by indiciating a white stripple pattern on the part of the layout where the DRC violation has occurred.



Loaded: NAND3 Editing: NAND3 Tool: box Technology: sky1 (-6.96 +2.49) +0.46 +2.45 microns

```
conroy.ecen.okstate.edu:
=> magic
```

DRC Rule	Error Number
▷ N-well overlap of P-Diffusion < 0.18um (diff/tap.8)	
▷ N-well overlap of N-tap < 0.18um (diff/tap.10)	

Target window: default

By using the previously created NAND and inverter cells, you efficiently constructed an AND gate in Magic. This hierarchical design approach saves time, maintains consistency, and simplifies the layout process, demonstrating the power and flexibility of using cells in VLSI design.

➢ Reading GDS Files in MAGIC:

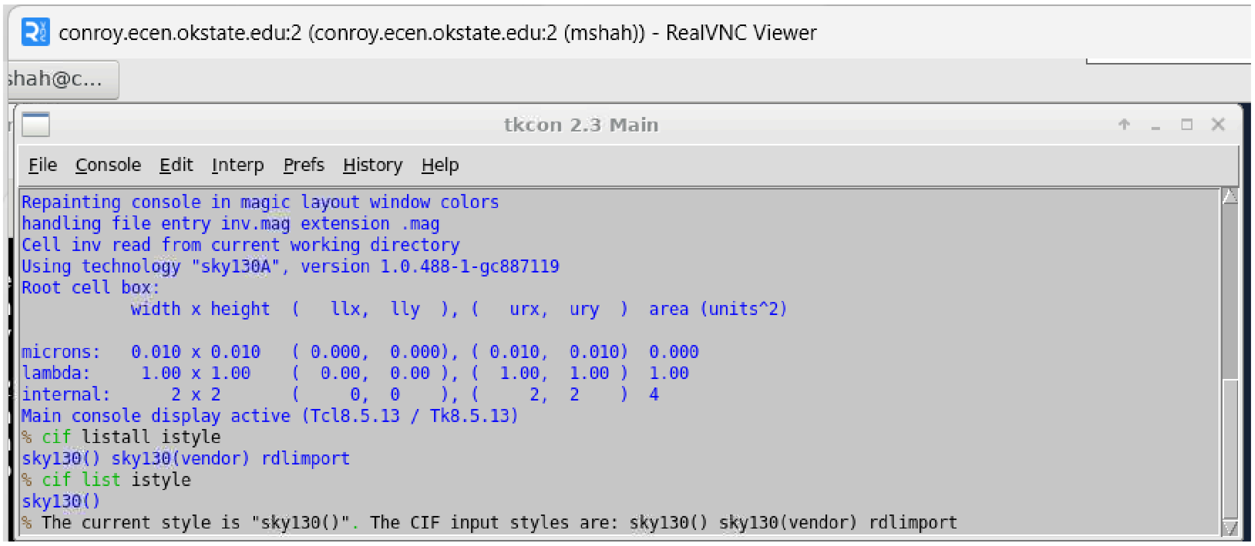
GDS files contain the geometric information of the IC layout, which is essential for fabrication. Reading GDS files into MAGIC allows designers to visualize and verify the final layout before it is sent for manufacturing.

The cif istyle xxx Command in MAGIC

The cif istyle xxx command in the MAGIC software is used to set the CIF (Caltech Intermediate Form) output style to xxx. CIF is a format for representing the layout of integrated circuits. The istyle parameter determines the CIF output style, which can vary depending on the fabrication process or other requirements.

Selecting the appropriate CIF style based on the fabrication process and design needs can improve the accuracy and efficiency of the design process. By using the cif listall istyle command, you can view all available CIF styles and choose the best one for your needs.

- **cif listall istyle**: Displays a list of all CIF styles available in MAGIC.
- **cif list istyle**: Displays the current CIF style in use.


conroy.ecen.okstate.edu:2 (conroy.ecen.okstate.edu:2 (mshah)) - RealVNC Viewer
shah@c...
tkcon 2.3 Main
File Console Edit Interp Prefs History Help

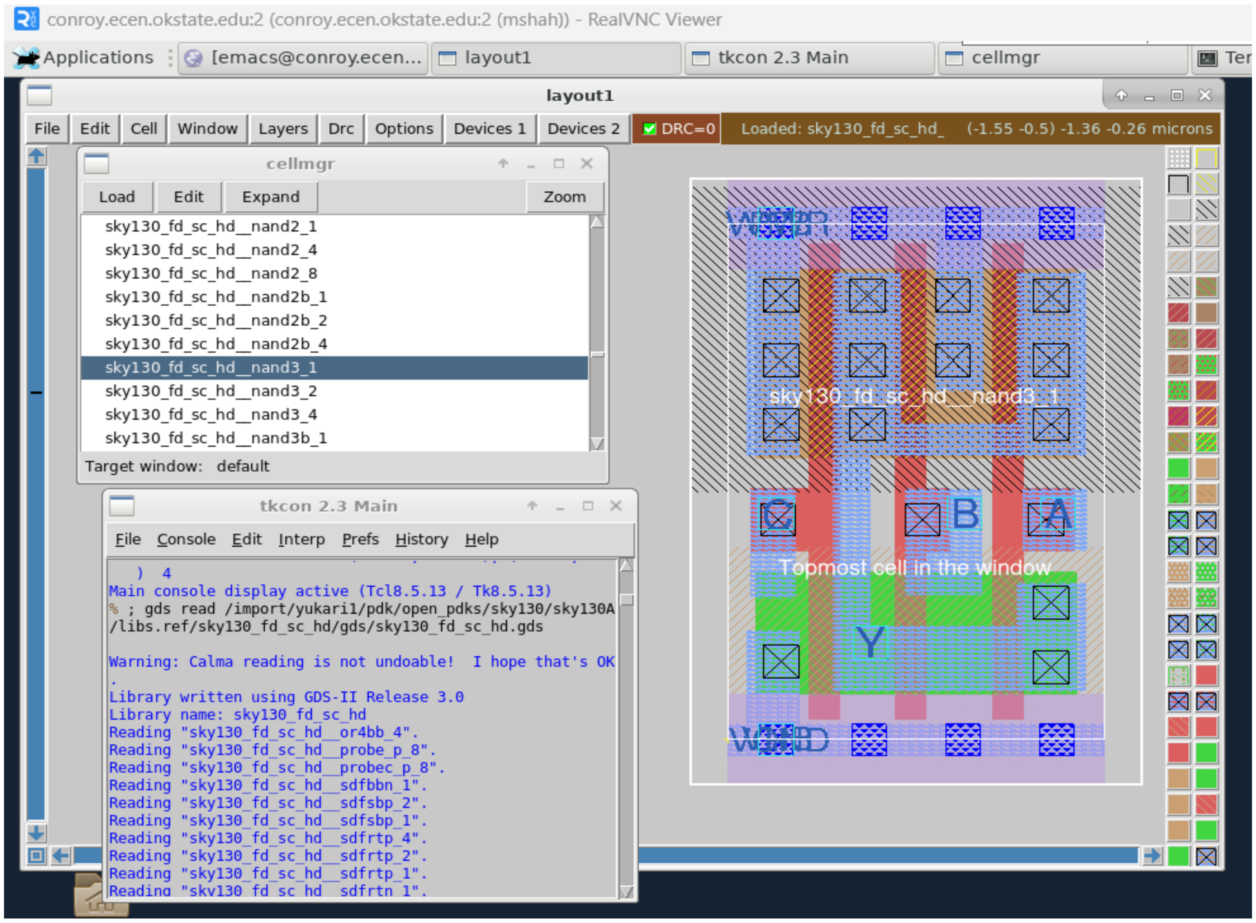
```
Repainting console in magic layout window colors
handling file entry inv.mag extension .mag
Cell inv read from current working directory
Using technology "sky130A", version 1.0.488-1-gc887119
Root cell box:
           width x height  (   llx,  lly  ), (   urx,  ury  )  area (units^2)

microns:   0.010 x 0.010   ( 0.000,  0.000), ( 0.010,  0.010)  0.000
lambda:     1.00 x 1.00    (  0.00,  0.00 ), (  1.00,  1.00 )  1.00
internal:      2 x 2       (     0,  0    ), (     2,  2    )  4
Main console display active (Tcl8.5.13 / Tk8.5.13)
% cif listall istyle
sky130() sky130(vendor) rdlimport
% cif list istyle
sky130()
% The current style is "sky130()". The CIF input styles are: sky130() sky130(vendor) rdlimport
```

If you want to change the current CIF style to `sky130(vendor)`, you can use the following command: `:cif istyle sky130(vendor)`

Now, using the default enabled style, let's try reading a GDS file from sky130 directory.

```
:gds read
/import/yukari1/pdk/open_pdks/sky130/sky130A/libs.ref/sky130_fb_sc_hd/gds/
sky130_fd_hd.gds
```

Now that the GDS file has been read, select a simple cell from Options --> Cell Manager, such as nand3_1.


conroy.ecen.okstate.edu:2 (conroy.ecen.okstate.edu:2 (mshah)) - RealVNC Viewer
Applications
[emacs@conroy.ecen...
layout1
tkcon 2.3 Main
cellmgr
File Edit Cell Window Layers Drc Options Devices 1 Devices 2
DRC=0
Loaded: sky130_fd_sc_hd_ (-1.55 -0.5) -1.36 -0.26 microns
Load Edit Expand Zoom
sky130_fd_sc_hd__nand3_1
Topmost cell in the window
C
B
A
Y
File Console Edit Interp Prefs History Help

cellmgr list:

```
sky130_fd_sc_hd__nand2_1
sky130_fd_sc_hd__nand2_4
sky130_fd_sc_hd__nand2_8
sky130_fd_sc_hd__nand2b_1
sky130_fd_sc_hd__nand2b_2
sky130_fd_sc_hd__nand2b_4
sky130_fd_sc_hd__nand3_1
sky130_fd_sc_hd__nand3_2
sky130_fd_sc_hd__nand3_4
sky130_fd_sc_hd__nand3b_1
Target window:   default
```

```
   )  4
Main console display active (Tcl8.5.13 / Tk8.5.13)
% ; gds read /import/yukari1/pdk/open_pdks/sky130/sky130A
/libs.ref/sky130_fd_sc_hd/gds/sky130_fd_sc_hd.gds

Warning: Calma reading is not undoable!  I hope that's OK
.
Library written using GDS-II Release 3.0
Library name: sky130_fd_sc_hd
Reading "sky130_fd_sc_hd__or4bb_4".
Reading "sky130_fd_sc_hd__probe_p_8".
Reading "sky130_fd_sc_hd__probec_p_8".
Reading "sky130_fd_sc_hd__sdfbbn_1".
Reading "sky130_fd_sc_hd__sdfsbp_2".
Reading "sky130_fd_sc_hd__sdfsbp_1".
Reading "sky130_fd_sc_hd__sdfrtp_4".
Reading "sky130_fd_sc_hd__sdfrtp_2".
Reading "sky130_fd_sc_hd__sdfrtp_1".
Reading "sky130_fd_sc_hd__sdfrtn_1".
```

➢ Reading a LEF Files in MAGIC:

To read a LEF (Library Exchange Format) file in MAGIC, you can follow these steps:

Setting CIF Style:

```
cif istyle xxx
```

Use the lef read command to read the LEF file. Suppose your LEF file is located at /path/to/your/lef/file.lef.

```
; gds read
/import/yukari1/pdk/open_pdks/sky130/sky130A/libs.ref/sky130_fd_sc_hd/lef/
sky130_fd_sc_hd.lef
```

After reading the LEF file, you can use MAGIC's tools to view and edit the layout. Ensure that the LEF file is read correctly and all necessary layers and information are available.

By following these steps, you can read a LEF file in MAGIC and use it for designing and laying out integrated circuits.