VSD Sky130 OpenLANE Workshop – Advanced Physical Design Using OpenLANE / Sky 130

This repository contains all the information included in the Advanced Physics Design Using OpenLANE / Sky 130 workshop. This workshop helped me in learning ASIC implementation steps from RTL to GDSII flow using open source EDA tools and Google – Skywater technology PDK. Google – Skywater130 PDK is the first manufacturable 130nm opensource PKD.

Table of Content

1. Introduction to OpenLANE

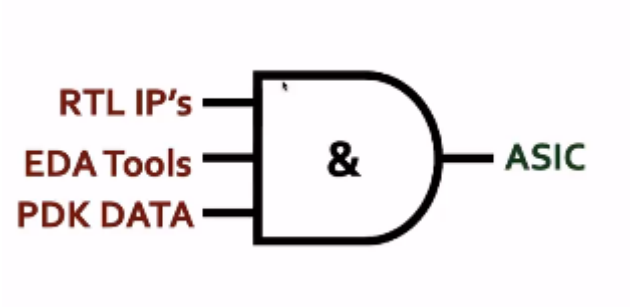
OpenLANE is an automated RTL to GDSII flow which includes different opensource tools like OpenROAD, Yosys, Magic, Netgen, Fault etc., apart from many custom methodology scripts for design exploration and optimization. It is an open-source flow for a true open source tape-out experiment. Its main goal is to produce a clean GDSII with no human interaction. It is tuned for Skywater 130nm Open PDK. It can be used to harden macros & chips.

It operates in 2 modes: Autonomous & Interactive.

2. Day 1: Inception of Open-Source EDA, OpenLANE & Sky130 PDK

Day1 starts with introduction to System on Chip (SoC) basics like chip, PADs, core, die etc., and RISC-V Instruction Set Architecture (ISA). ISA is a language of the computer and the way we talk to the computer. In RISC-V to layout flow, the code in the RISC-V Architecture is implemented using RTL and converts code into Hardware Description Language (HDL) then code is converted to layout using the general RTL to GSD flow.

For open-source implementation of Digital ASIC Design, we need 3 components:

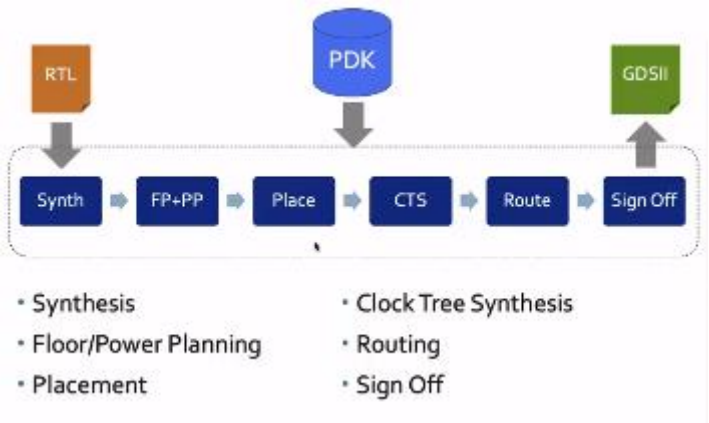


RTL IP's: these are easily available on open-source platforms like librecores.org, opencores.org, github.com etc.,

EAD Tools: QFlow, OpenROAD, OpenLANE are open-source EDA tools available.

PDK DATA: Open-source PDK released by Google + Skywater technology.

Simplified RTL to GDS flow:



- Synthesis

  Design RTL is translated into circuits made out of components from Standard Cell Library (SCL). The resultant circuit is described in HDL and usually refers to as the gate level netlist. The gate level netlist is functionally equal to the RTL.
- Floor Planning and Power Planning (FP & PP)

  Floor planning and power planning varied depending upon is we are implementing on one component (macro floor planning) or a whole chip. The main objective here is to plan a silicon area & create robust power distribution network to power the circuits.

  In Macro floor planning, macro dimension, pin location, rows & routing tracks are defined.

  In power planning, power network is constructed using power pads, power straps and power rings.
- Placement

In Placement, gate level netlist cells are replaced by macros. Placement of the cells on the floorplan, aligned with the sites is called placement, it helps to interconnect the delay and also enable successful routing. Usually placement is done in 2 steps: Global placement and detailed placement.
Global Placement: It tries to find the optimal placements for all cells, such placement is legal which means there can be some cell overlaps.
Detailed Placement: In detailed Placement, positions obtained from the global placement are minimally altered to be legal.

- Clock Tree Synthesis (CTS)

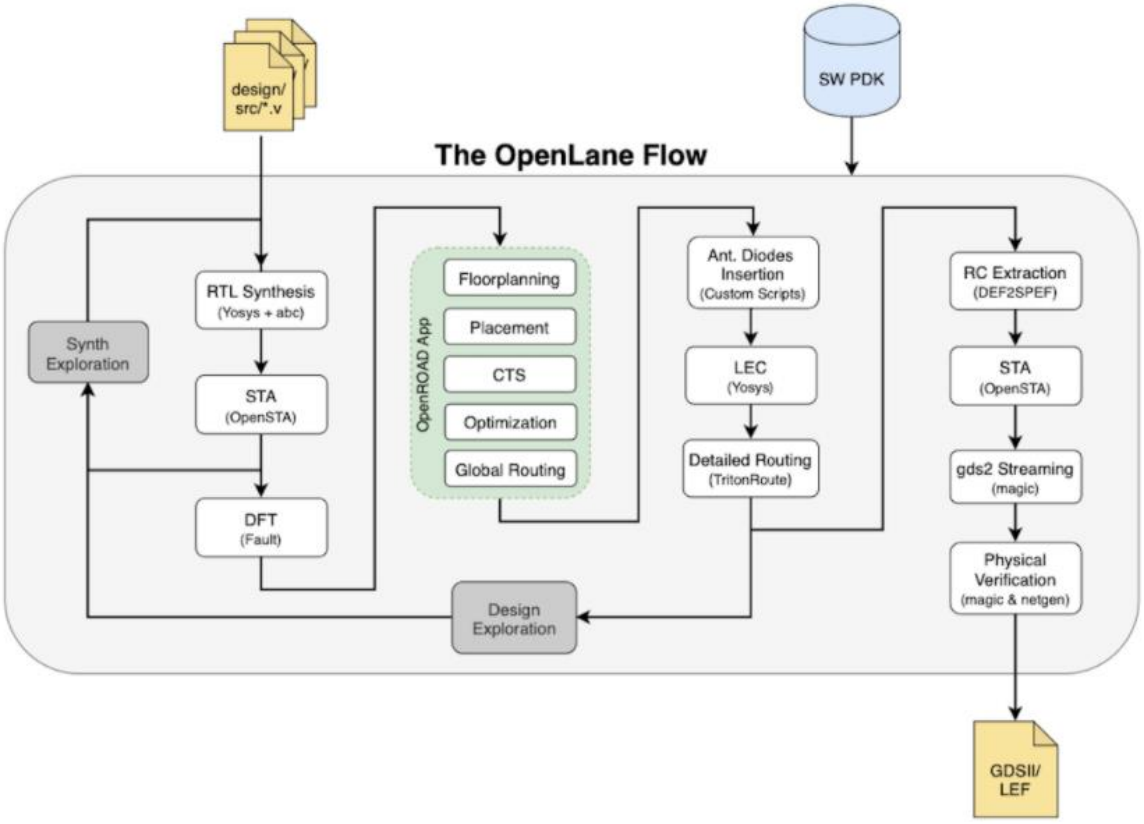
  This step is used to route the clock before the signal are routed. By creating a clock distribution network to deliver the clock to all sequential elements, with minimal skew and in a good shape. The clock network is usually in a shape of tree.

- Routing

  After clock routing now it comes to signal routing. In Routing, valid horizontal and vertical pattern of wires are found to implement net connecting cells together using available metal layers. Routing is performed in 2 steps: Global routing – generates routing guides and detailed routing – uses the routing guides to implement the actual wiring.

- Sign-Off

  In sign-off stage *Physical verification* is done using Design Rule Checking (DRC) & Layout Vs Schematic (LVS) and *Timing verification* is done using Static Timing Analysis (STA).

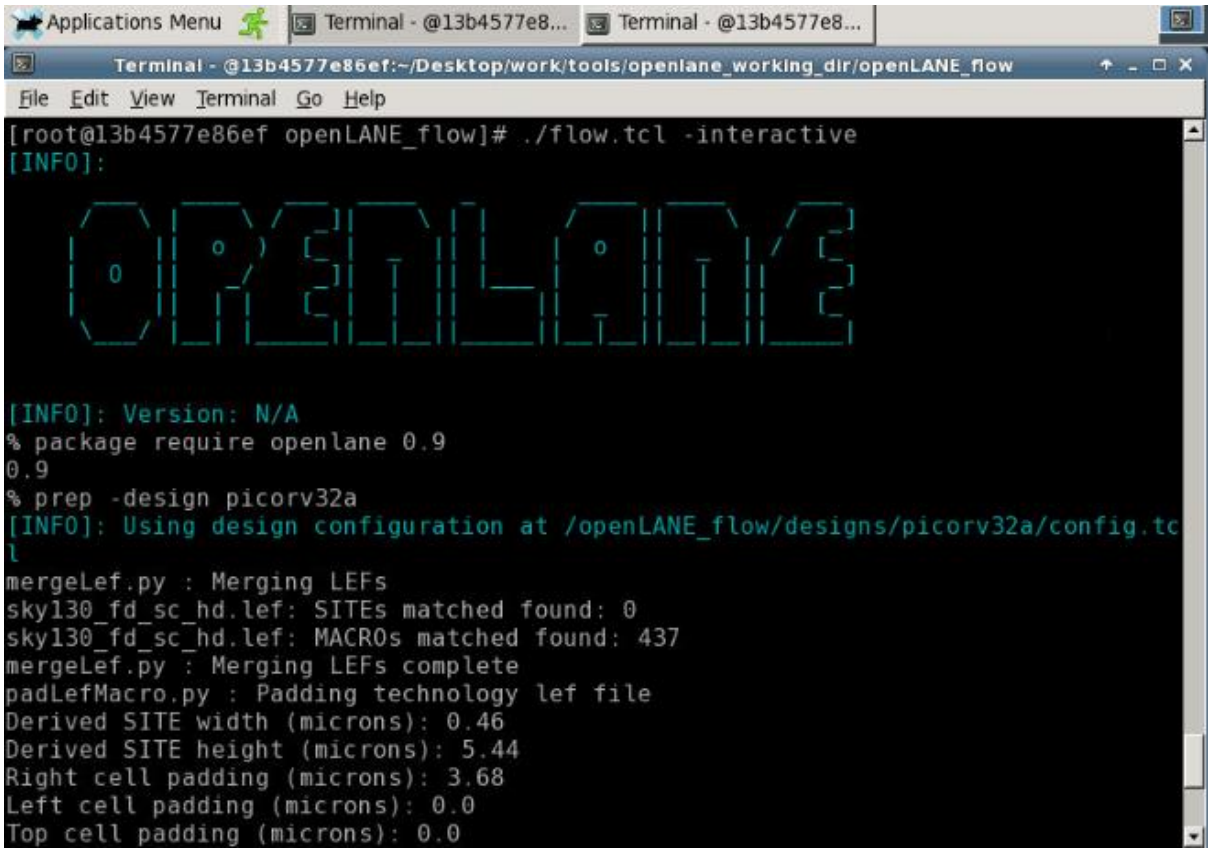


Above picture shows the detailed ASIC design flow. Here new concepts like design exploration, Design for test (DFT), logic equivalence check (LEC), antenna diodes insertion was explained.

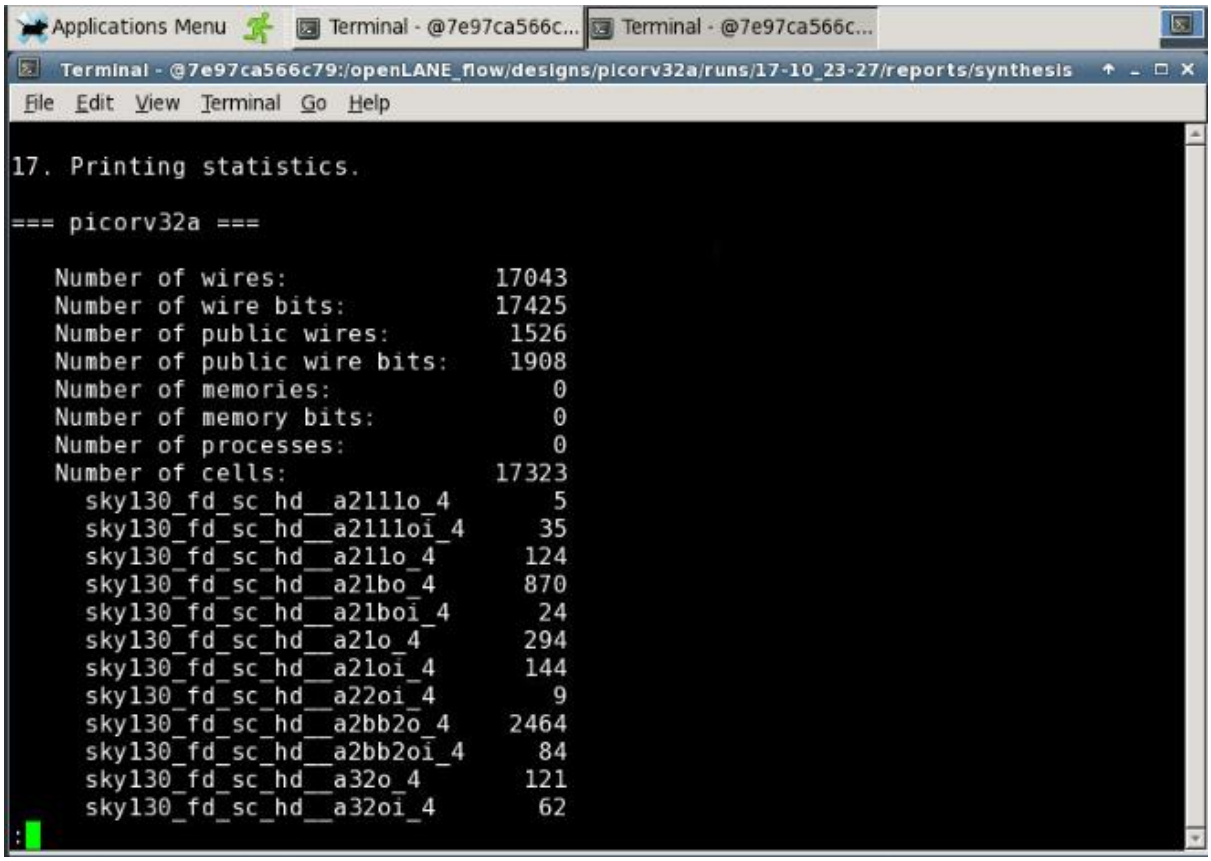
Design exploration helps in finding the best configuration for the design. DFT is useful to make sure our design is ready for testing after fabrication. As optimization is performed in physical design step, which can form change in the gate level netlist (netlist generated by the synthesis step), we need to perform LEC. Every time netlist changes, verification mush be performed. LEC formally confirms the function did not change after modifying the netlist. Antenna Diode problem arises for long nets which is avoided by using fake antenna diodes next to every cell after placement. Later Antenna Checker (Magic) is run. If it reports antenna violations then fake diode replaced by actual one.

LAB:

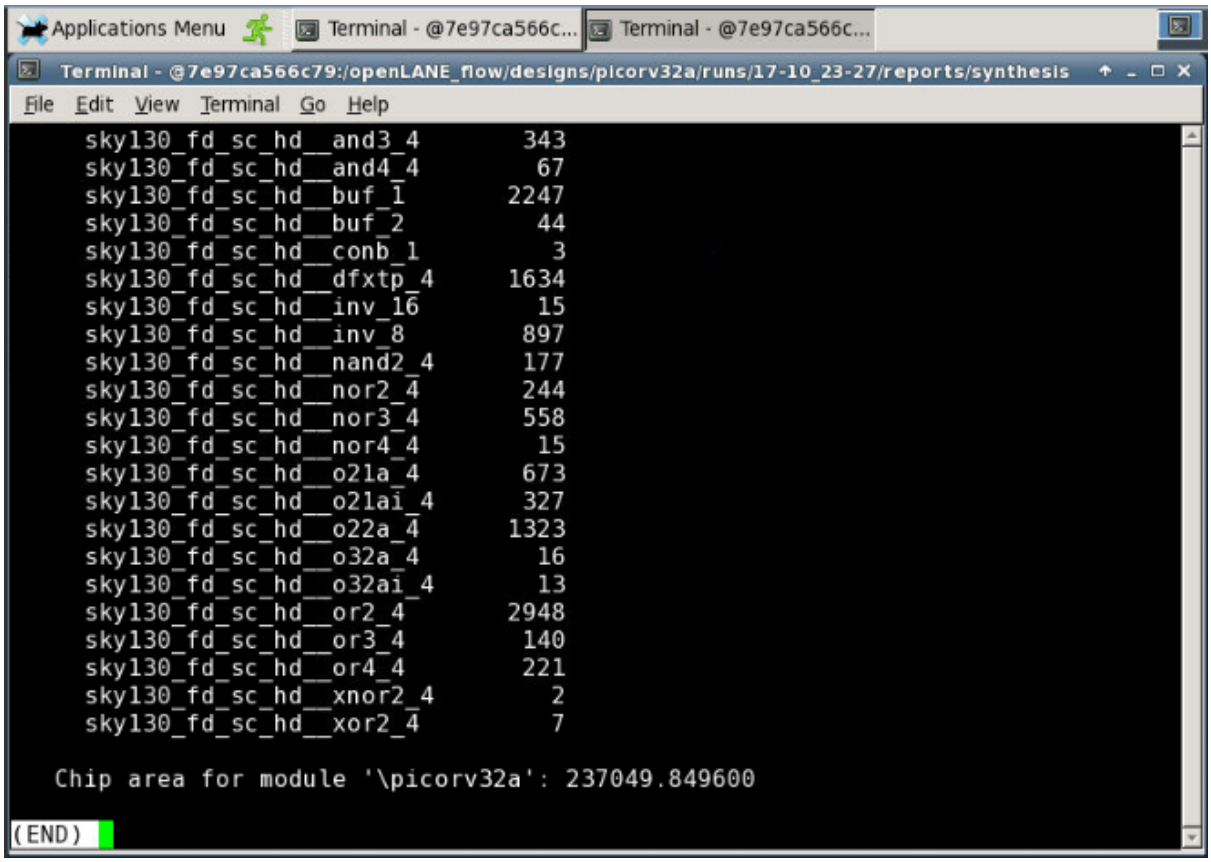
To run OpenLANE



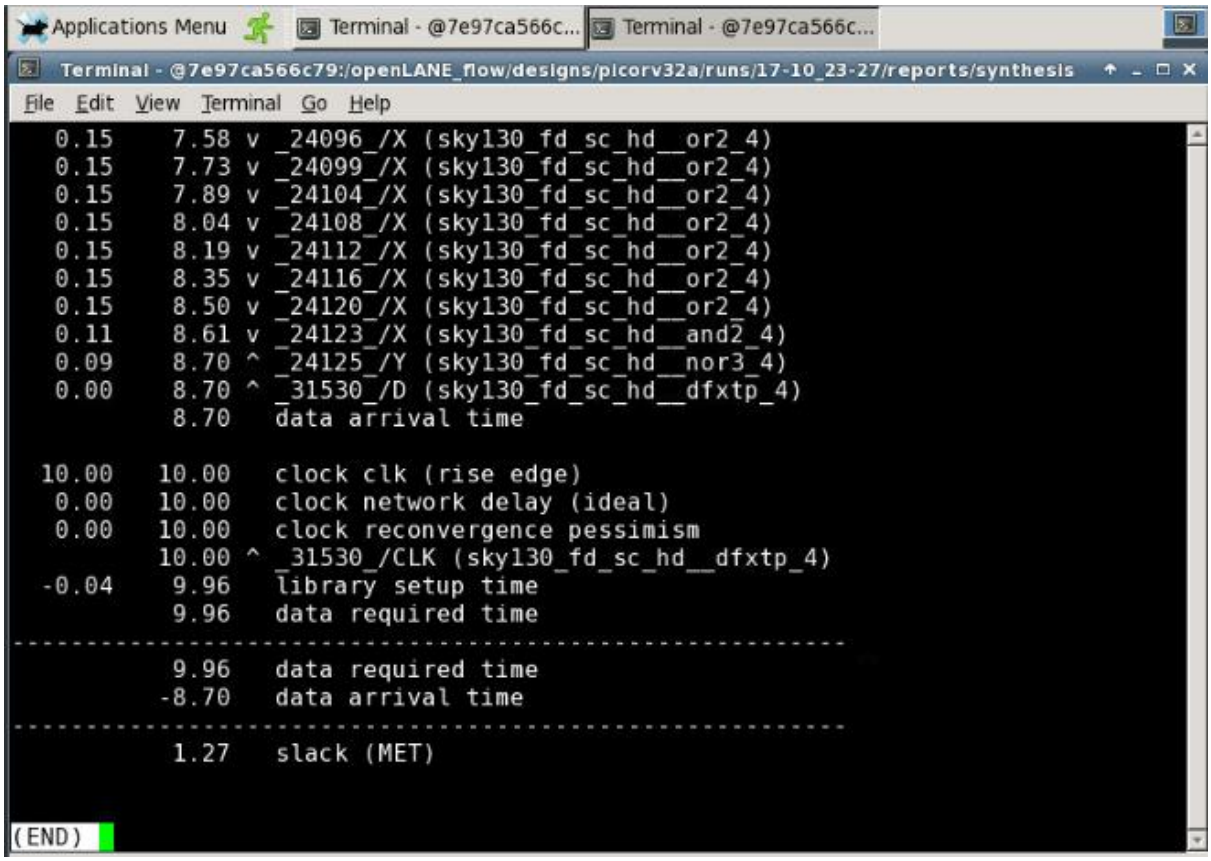
Results of synthesis showing design components



Synthesis showing chip area and number of D F/F’s



Slack calculation of design



3. Day 2: Good Floor planning Vs Bad Floor planning and Introduction to library cells

On Day 2, definition of width and height of core and die were explained. Later, the concepts like *Utilization factor* and A*spect ratio* were explained in detail and their importance to understand the design was also mentioned.

Utilization factor = (Area occupied by netlist) / (Total area of the core)

Ideally, utilization is considered to be 50% - 60% and utilization factor to be 0.5 / 0.6.

Aspect ratio = (Height (of the die)) / (Width (of the die))

Steps involved to define *Pre-placed* cell & its advantage of enhancing reusability and *de-coupling capacitors* and how they help during switching to avoid failure was explained. Once the pre-placed cells are located on the core, they cannot be moved further so these needs to be place carefully. The pre-placed cells are surrounded by De-coupling capacitors. A fully charged *de-coupling capacitor* is placed parallel to circuits to ensure proper supply of peak current I_{peak} by decoupling them from main supply voltage. Hence *de-coupling capacitor* ensures *proper local communication* while multiple V_{dd} & V_{ss} lines lead to *proper global communication* (which is taken care in power planning) avoiding voltage droop and ground bounce conditions. Step of *pin*

placement & logical cell placement blockage is explained. *Logical cell placement* is done to avoid PnR tool to place anything cells and to reserve the area for inputs and output pins.

Cell Design:

Cell design is don't in 3 parts.

- Inputs

  Inputs for cell design flow are PDKs (Process design kits), DRC & LVS rules, SPICE models, library & user-defined specs.
- Design Steps

  Design steps of cell design involve Circuit Design, Layout Design, Characterization. The software GUNA used for characterization. The characterization can be classified as Timing characterization, Power characterization and Noise characterization.
- Outputs

  Outputs of the Design are CDL (Circuit Description Language), GDSII, LEF, extracted Spice netlist (.cir), timing, noise, power.libs, function.

Typical Characterization Flow:

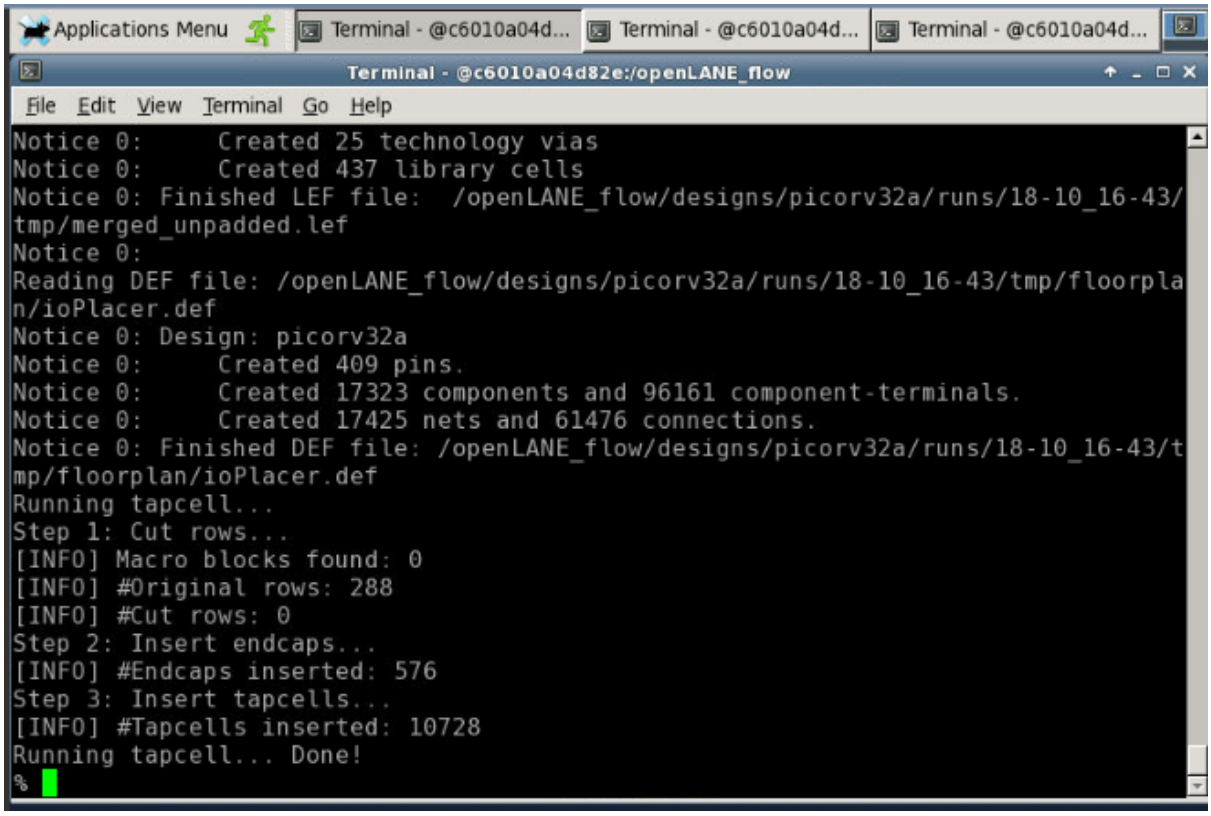
- Reading the Model file of CMOS containing basic property definitions.
- Reading the extracted Spice Netlist.
- Recognize / Define the behaviour of the cell -- buffer cell.
- Read the subcircuits -- of the inverter.
- Attach the necessary power sources.
- Apply an input or the stimulus.
- Provide the necessary output capacitance.
- Provide the necessary simulation commands.

Important parameters of Timing Characterization:

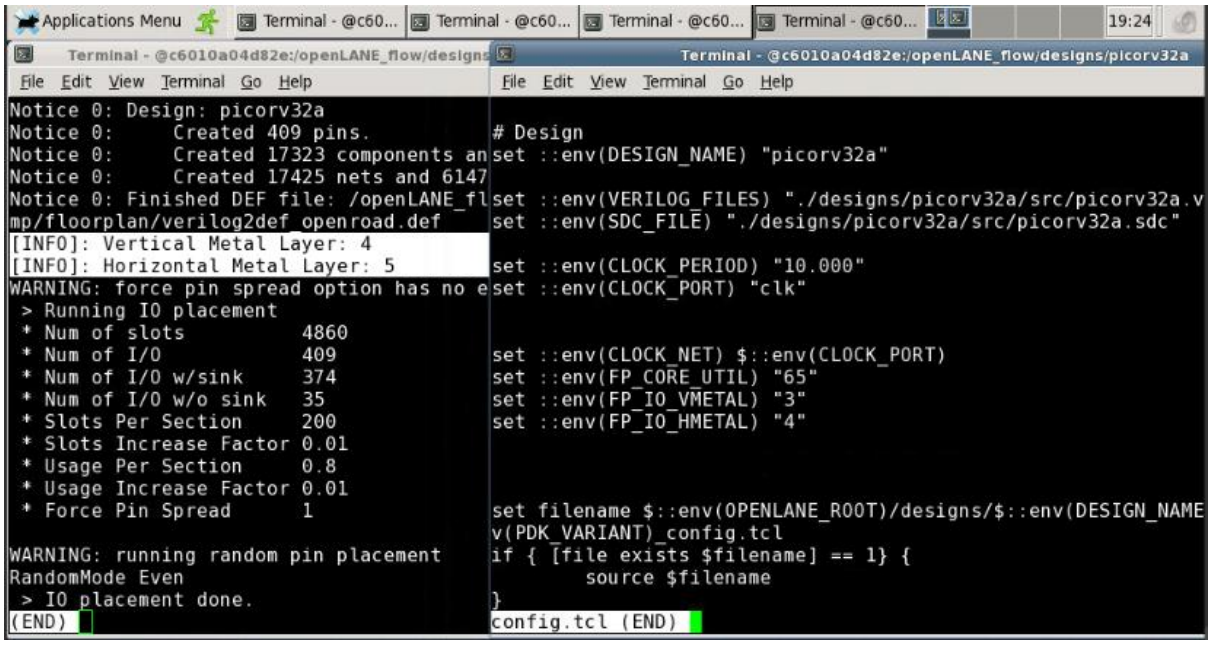
- Rise Delay: Time taken for waveform to rise from 20% to 80% of VDD.
- Fall Delay: Time taken for waveform to fall from 80% to 20% of VDD.
- Propagation Delay: Measured between 50% of Input transition to 50% of Output transition.

LAB:

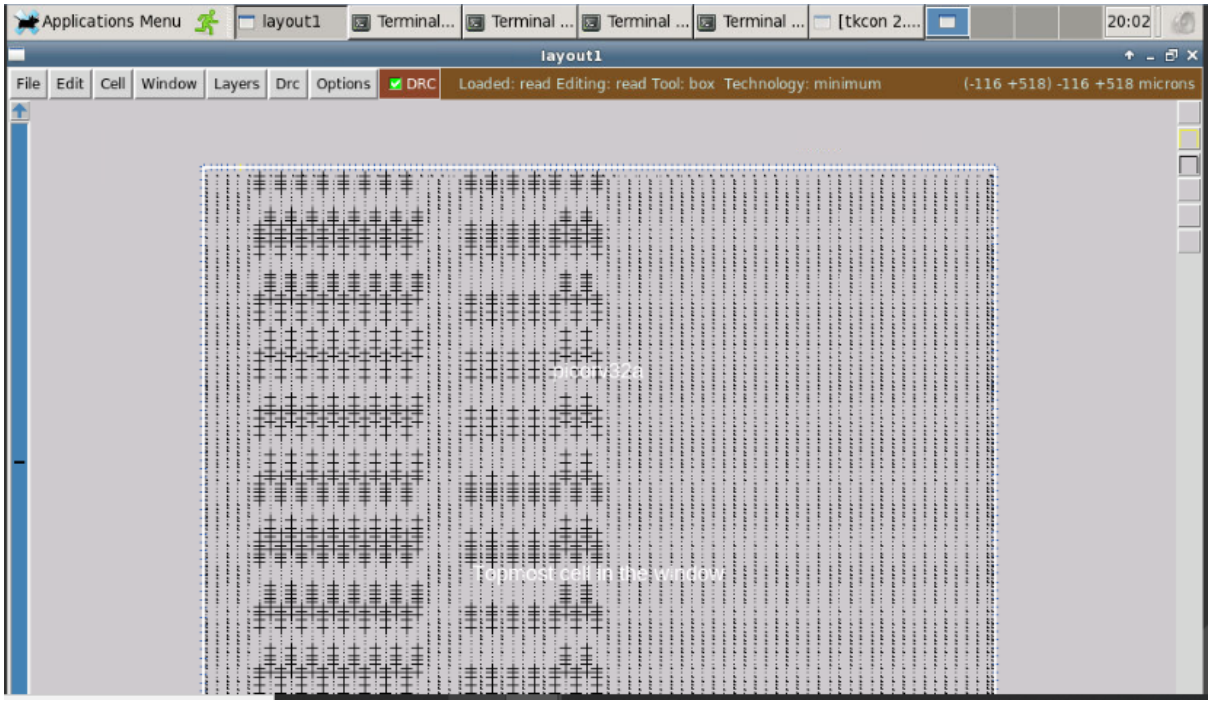
Run Floorplan is completed successfully



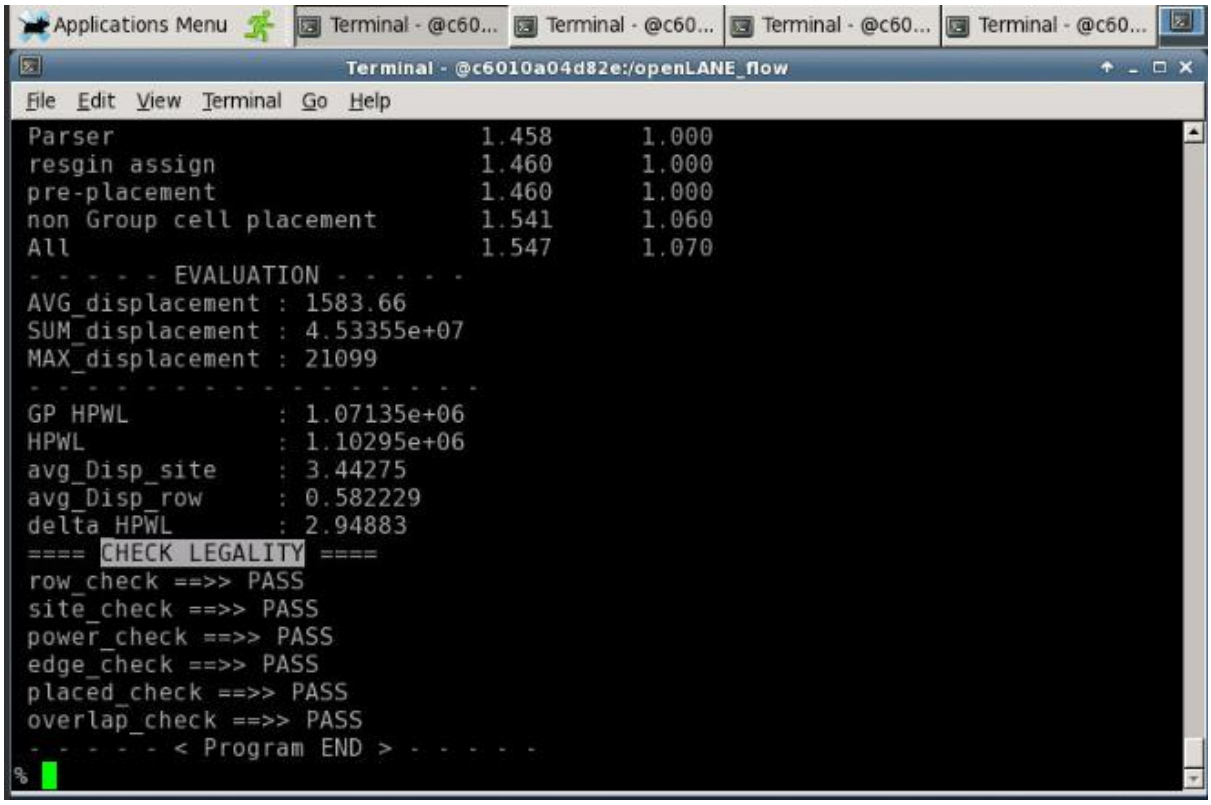
Setting up VMETAL and HMETAL



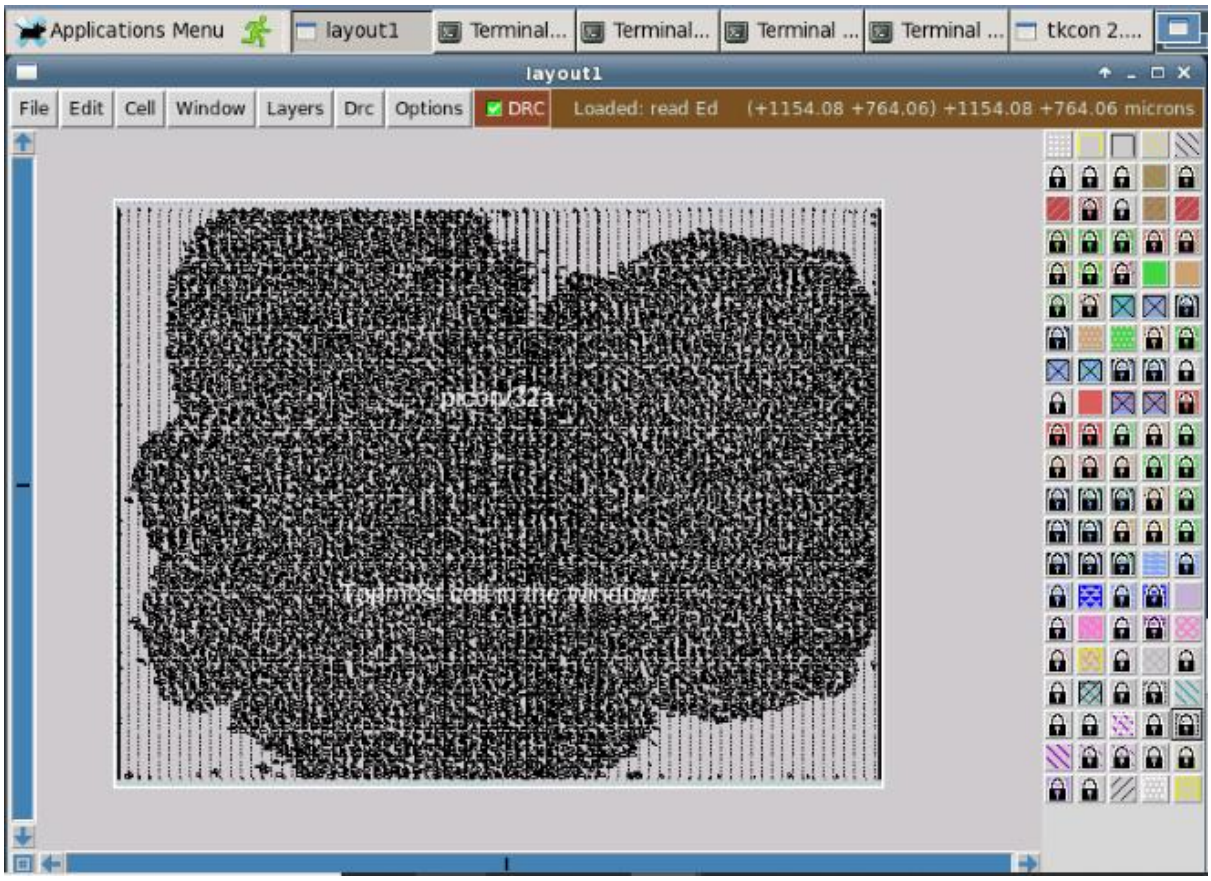
Review floor plan layout in Magic



Run placement is completed successfully



Review placement layout in Magic



4. Day 3: Design library cell using Magic layout and ngSpice characterization

On Day3, SPICE deck creation for an inverter was explained and later, 16-Mask CMOS process was explained step-by-step in detail. OpenLANE offers an interesting feature of making changes into parameters on the go. This helps to deal with issues like congestion. SPICE deck formation contains information's like components connectivity, component values, Identify 'nodes' and name 'nodes'. It was showed how W/L ratio of MOS impacts its conductivity and hence reason of carefully defining W/L ratio of MOS to ensure same *rise* and *fall* delay for clock signals. CMOS robustness defined with the help of parameters that are *Switching Threshold* (V_m). Switching threshold defined by condition $V_{in} = V_{out}$.

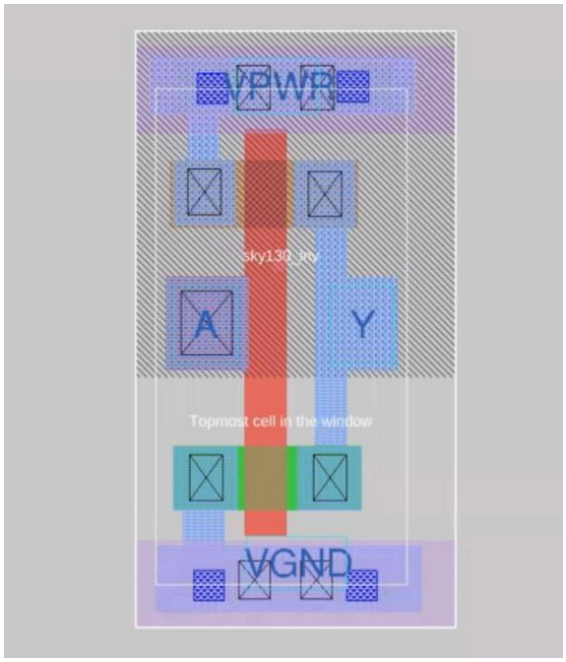
16-Mask CMOS Process steps:

1. Selecting a substrate.
2. Create Active region for transistors.
3. Formation of N-Well and P-Well.
4. Formation of Gate.
5. Lightly Doped Drain (LDD) formation.
6. Source and drain Formation.

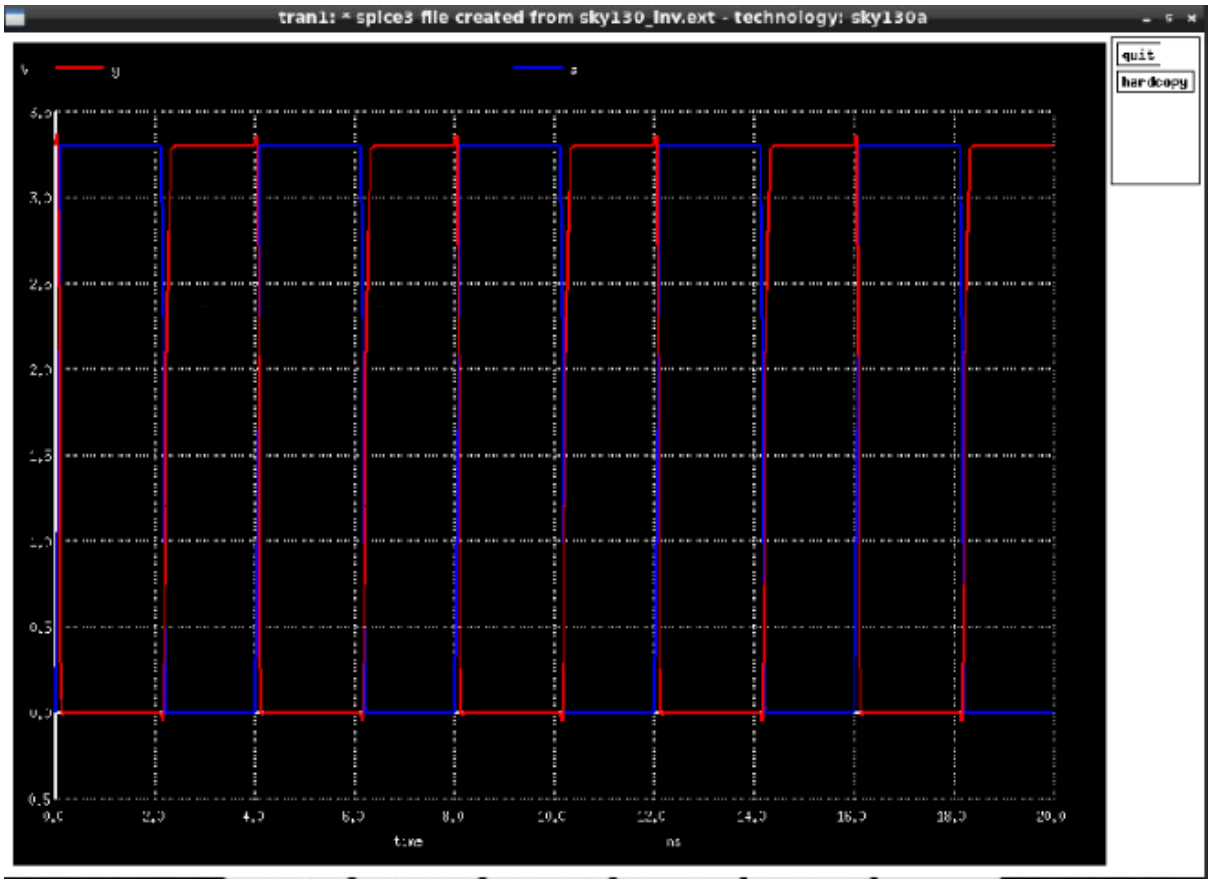
7. Steps to form Contacts and interconnects (local).
8. Higher level formation.

LAB:

Inverter Layout in Magic



Transient analysis of an inverter in ngspice



5. Day 4: Pre-Layout timing analysis and importance of good clock tree

On Day4, Delay tables, setup & hold time concepts were explained. I learned that delay and output transition values of any particular cell are calculated with the help of values of *input transition* and *output load* values. Delay table and output transition table of a cell contain different value for each combination of input trans and output load, represented in form of lookup tables in liberty file.

Setup & Hold Slack Analysis:

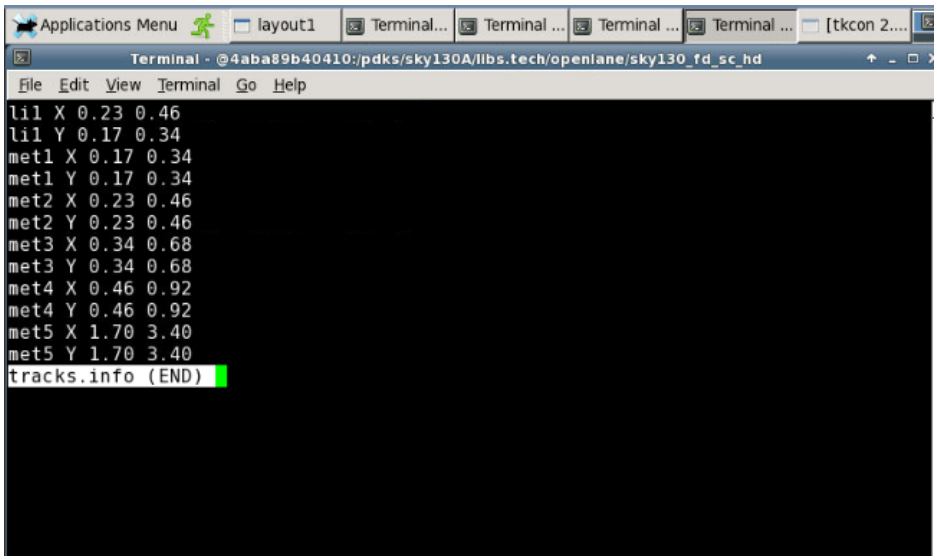
Setup and hold time define a window of time in which our data should remain unchanged for desired data transfer to take place. Factors like uncertainty and skew also play an important role in this. Clock skew is the difference between Source Clock path and Destination Clock path. Slack defined as difference between actual time and required time is monitored. Positive or zero Slack indicates no violation whereas negative slack value indicates violation of timing.

OpenROAD Commands:

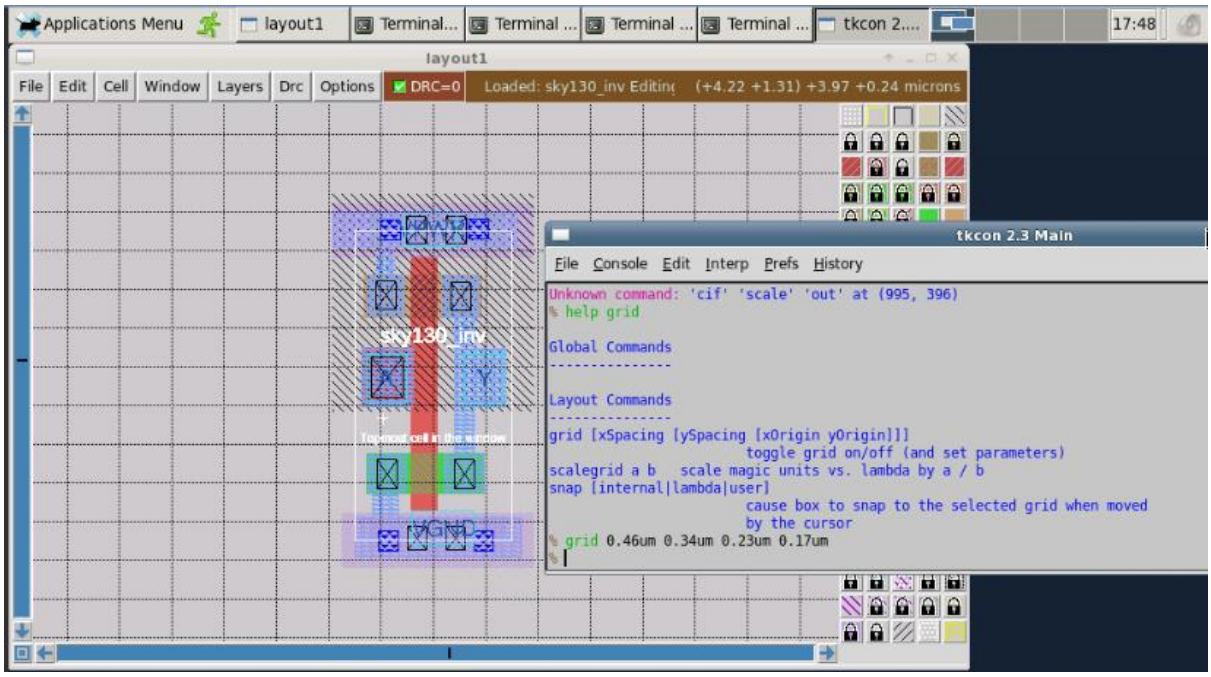
```
read_lef /openLANE_flow/designs/picorv32a/runs/trial/tmp/merged.lef
read_def /openLANE_flow/designs/picorv32a/runs/trial/results/cts/picorv32a.cts.def
write_db pico1.db
read_verilog
/openLANE_flow/designs/picorv32a/runs/trial/results/synthesis/picorv32a.synthesis_cts.v
read_liberty $::env(LIB_SYNTH_COMPLETE)
lik_design picorv32a
read_sdc /openLANE_flow/designs/picorv32a/src/my_base.sdc
set_propagated_clock [all_clocks]
report_checks -path_delay min_max -digits 4
```

LAB:

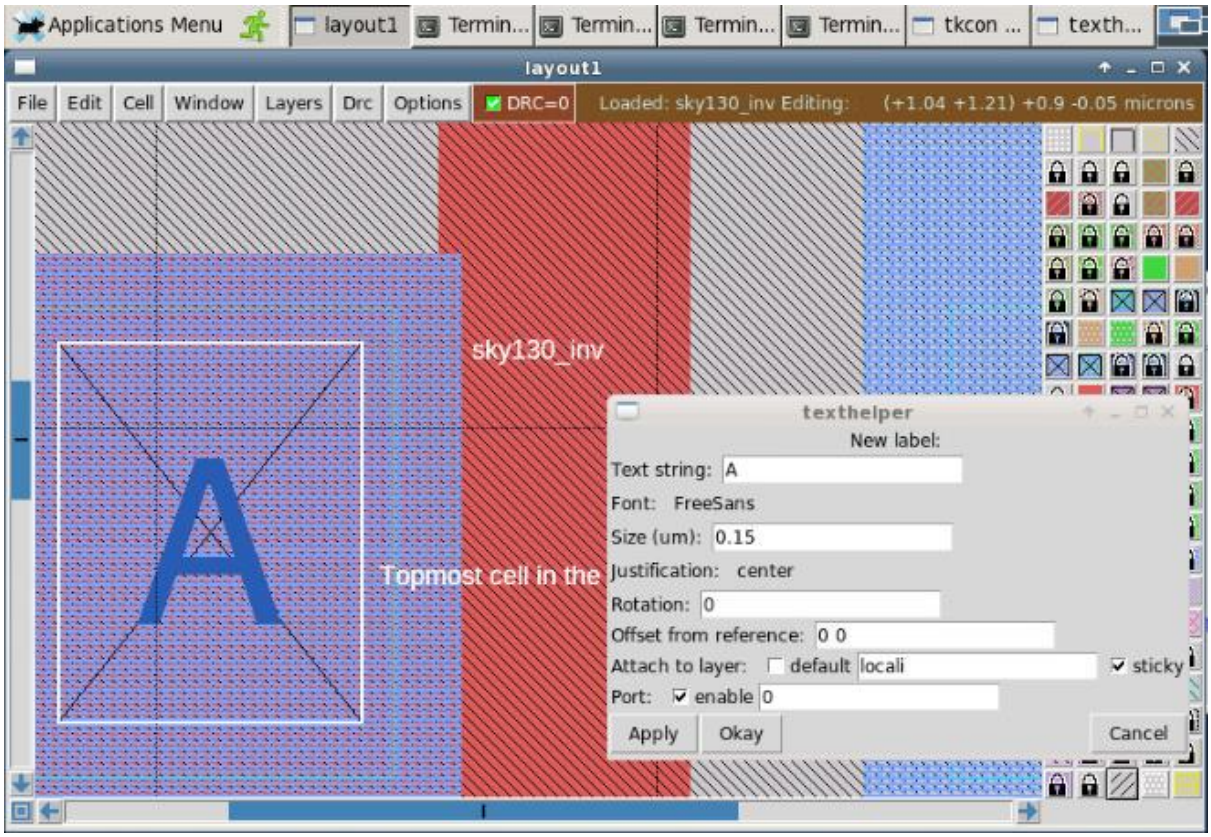
Lab1: Layer information in sky130_fd_sc_hd file



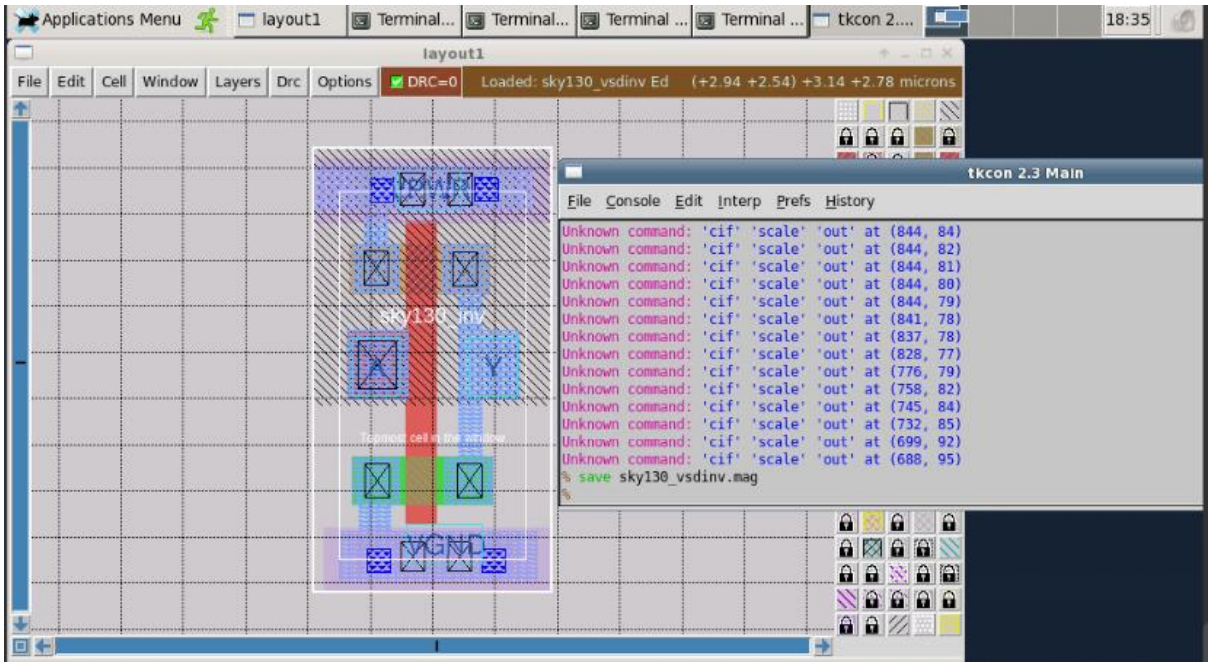
Gird size modified to track size



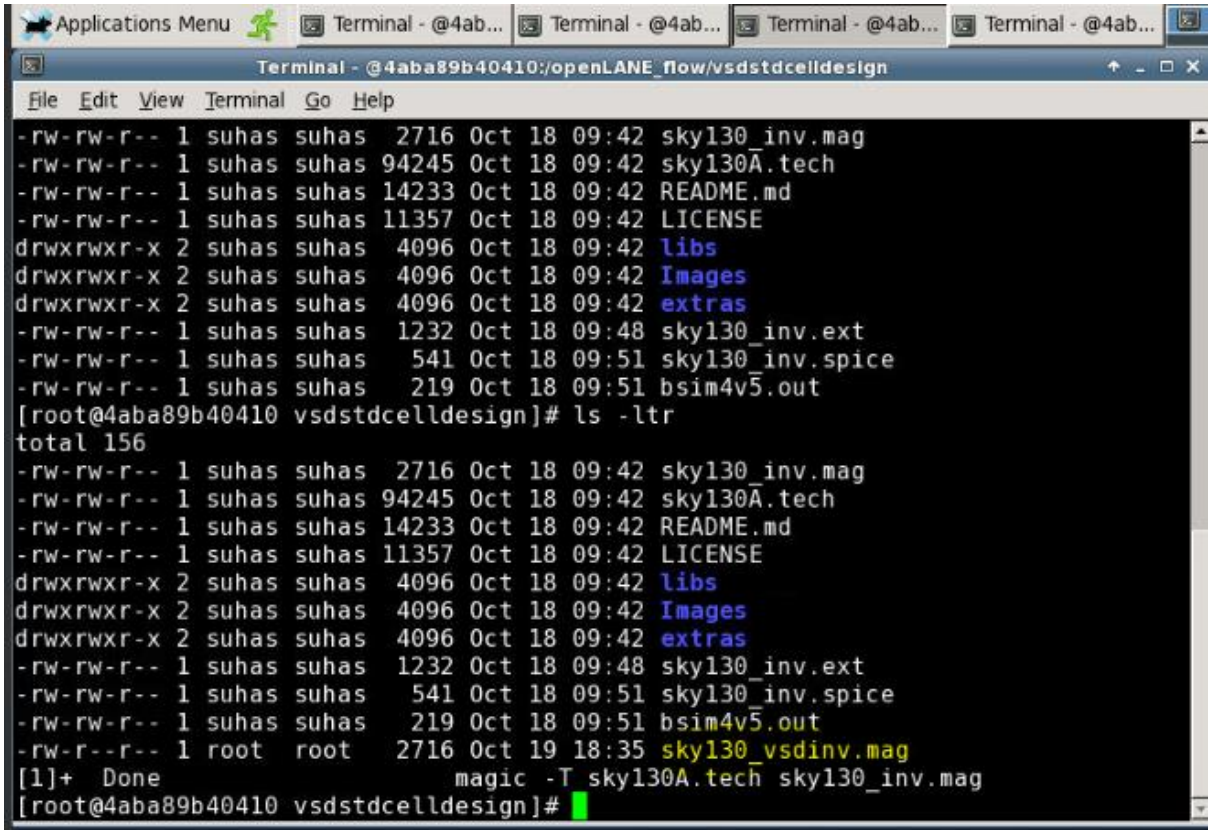
Process to modify labels



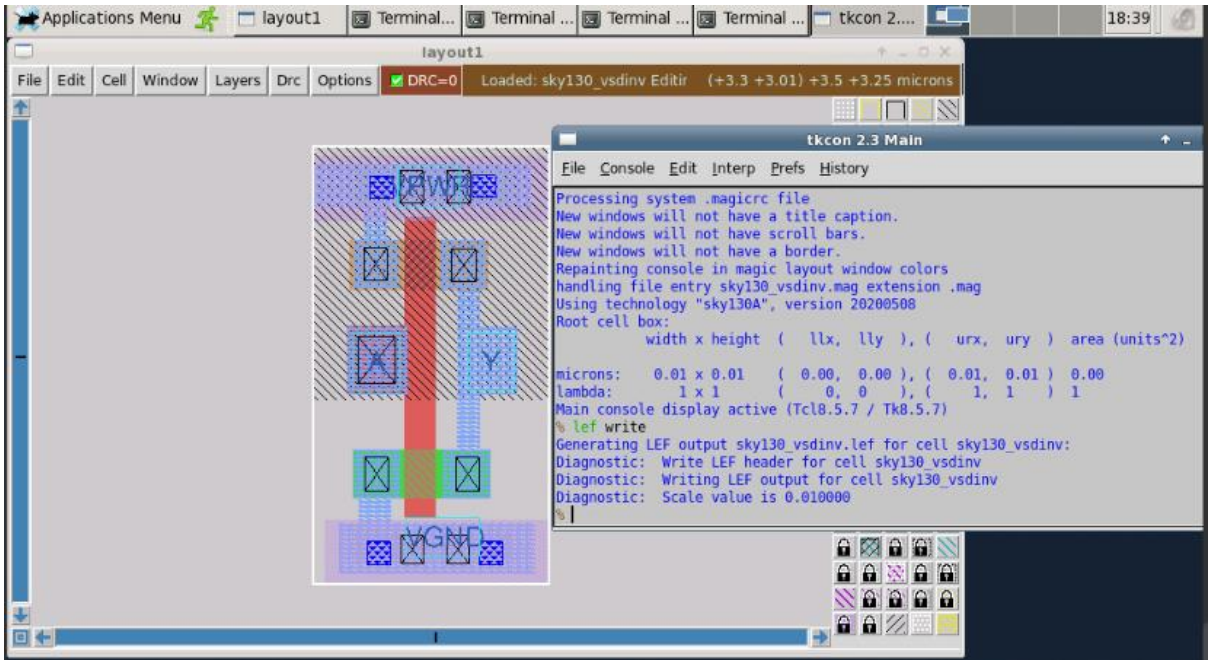
Command to convert Magic layout to .mag



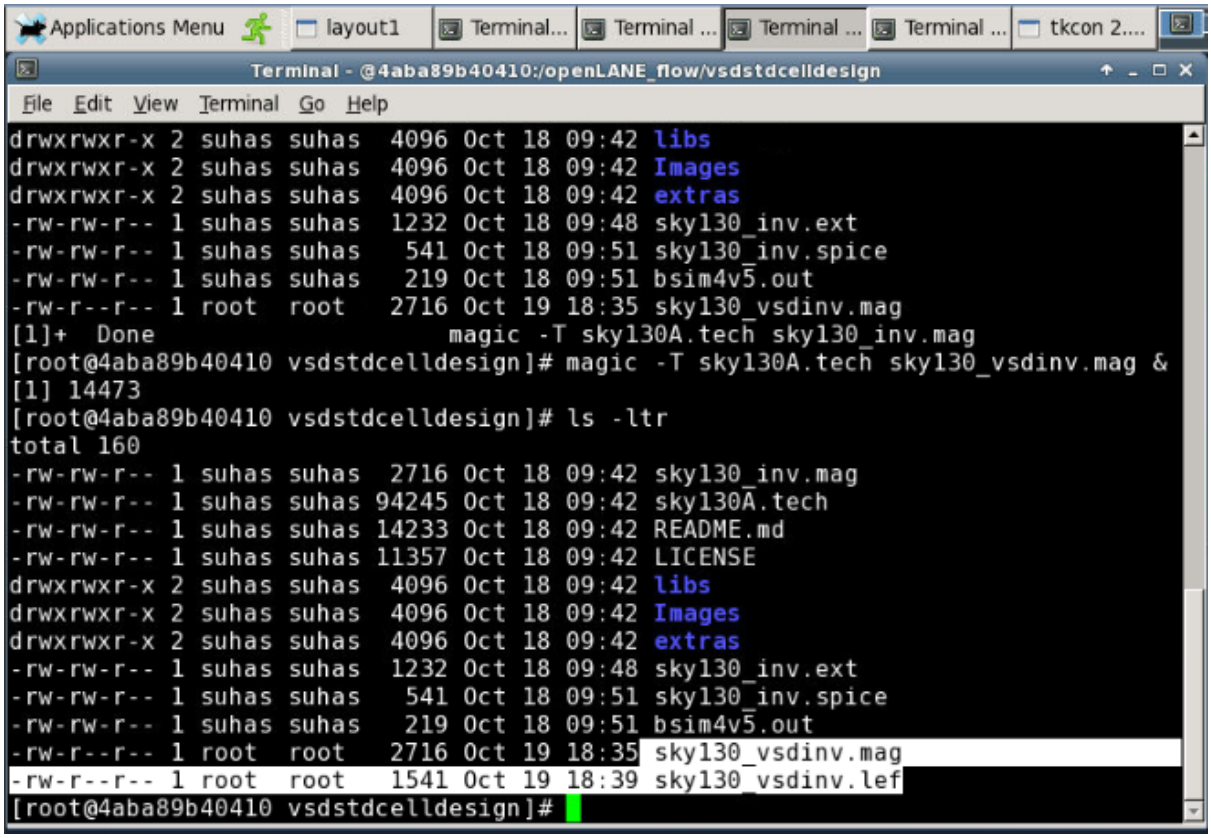
Converting Magic Layout to .mag



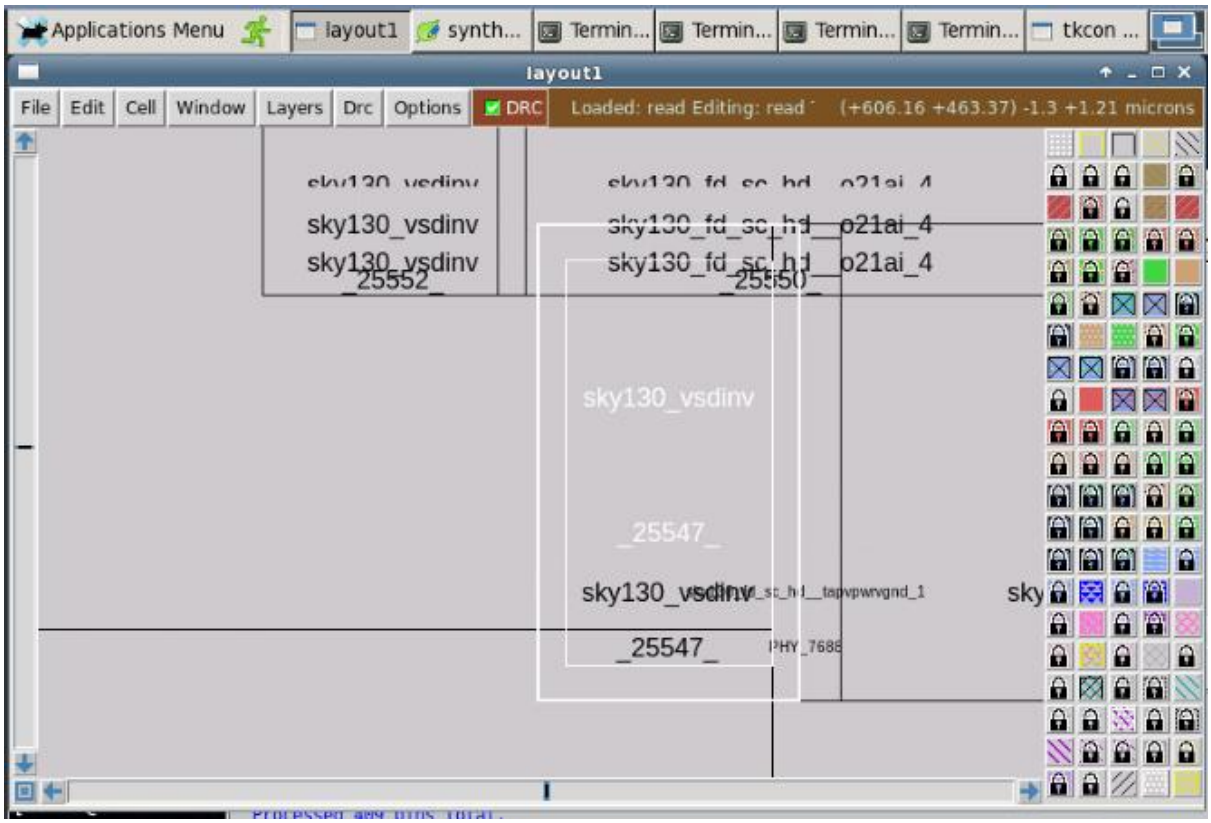
Creating LEF file from Magic layout



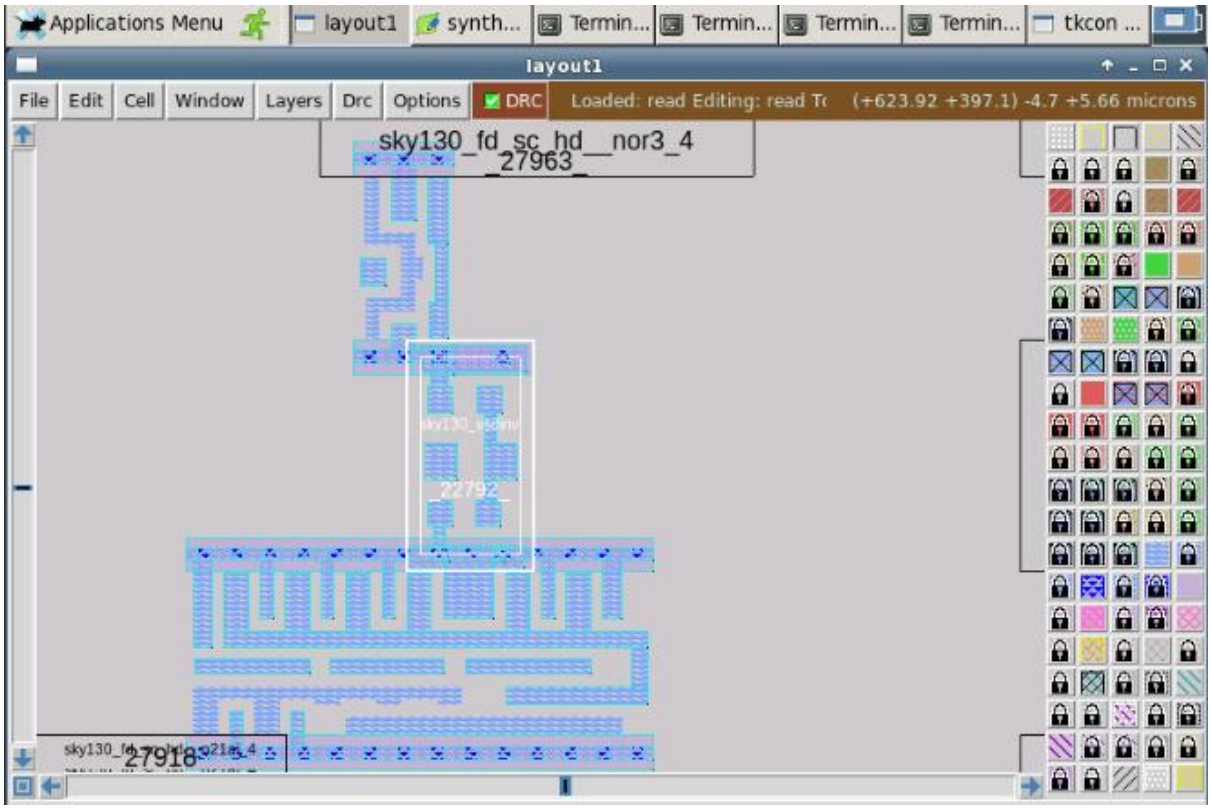
LEF file created



Layout of the sky130_vdsinv Cell



Layout in expanded view



Lab2: Reduction in the Slack Violation and increase in area

synthesis data

File Edit Search Options Help

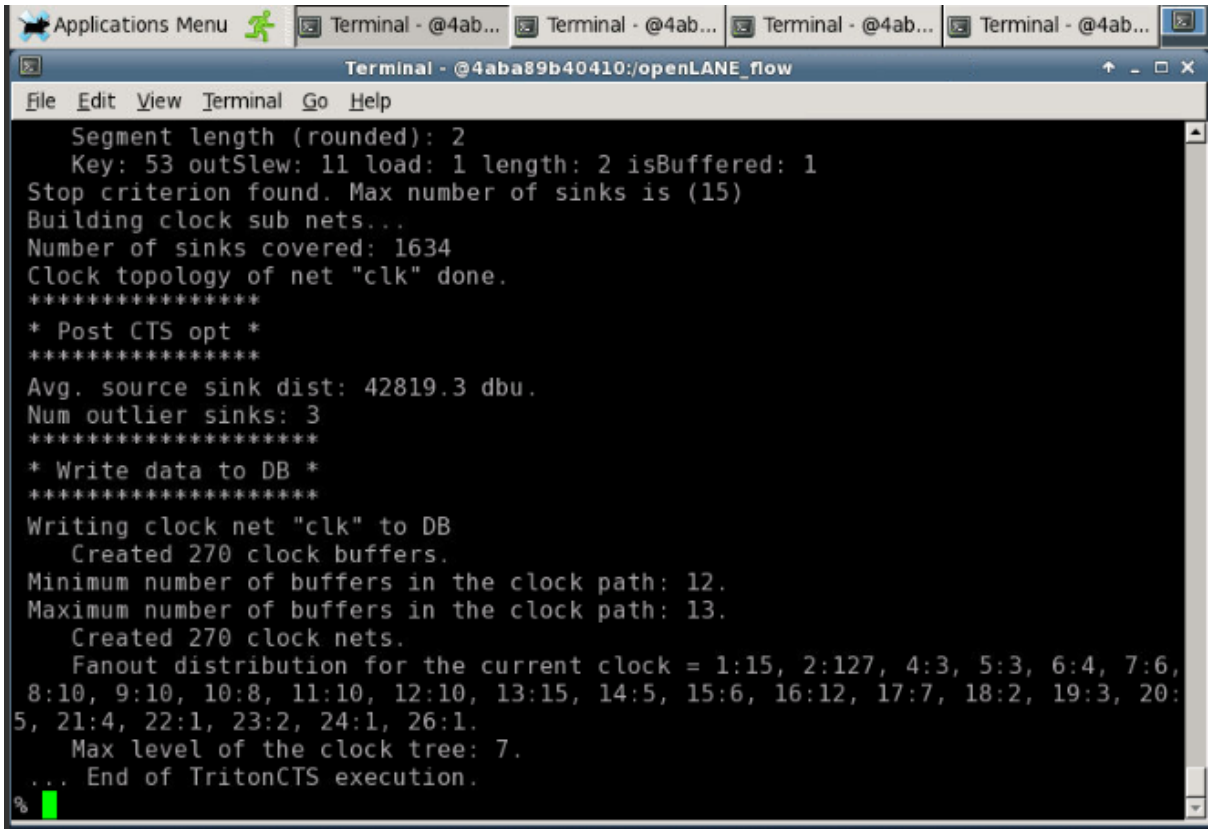
```
before:
Chip area for module '\picorv32a': 225955.459200
-17.96   slack (VIOLATED)
tns -2593.43
wns -17.96


after:
Chip area for module '\picorv32a': 317725.974400
-5.36   slack (VIOLATED)
tns -356.06
wns -5.36


after fanout as 4:
Chip area for module '\picorv32a': 325893.808000
-3.76   slack (VIOLATED)
tns -96.76
wns -3.76

after 2 buffer upsizes:
-0.91 slew
tns -7.71
wns -1.16
```

Terminal - @4aba89b40410:/openLANE_flow

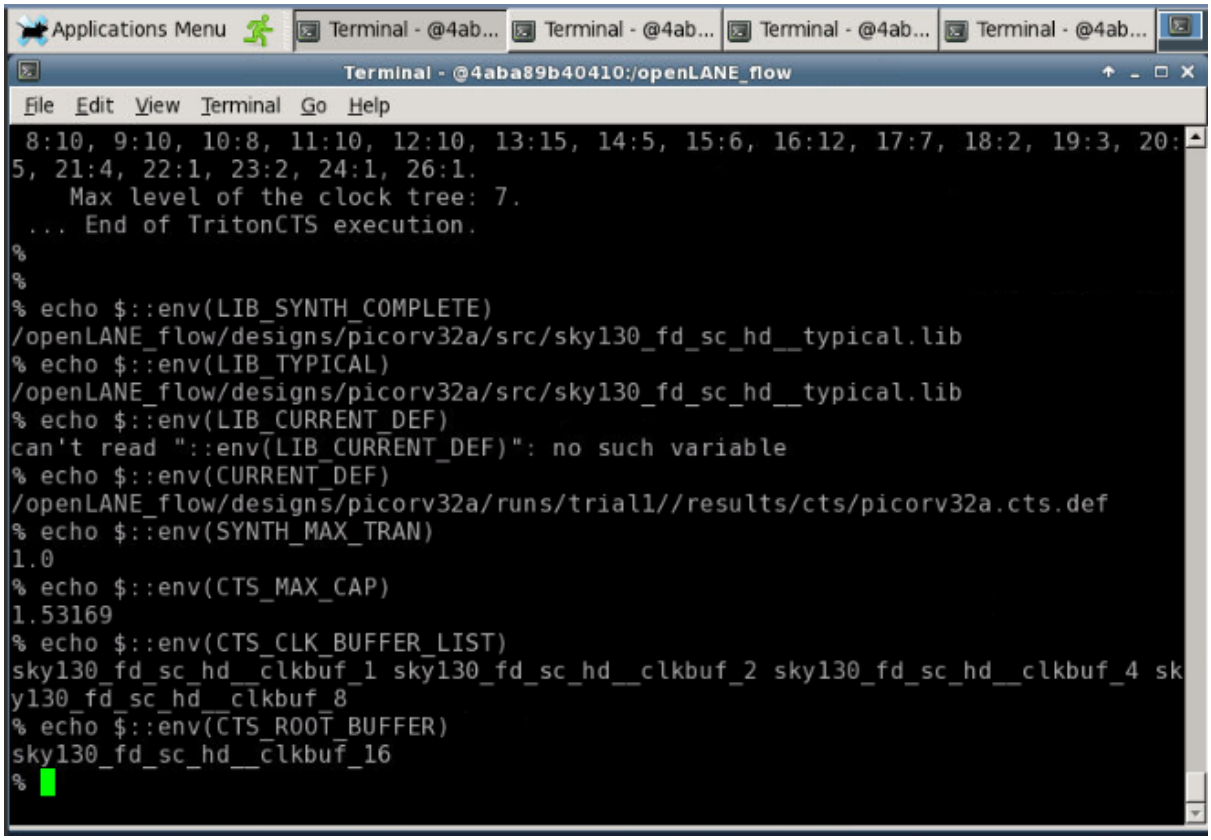
File Edit View Terminal Go Help

```
   0.00    12.62 v _50075_/D (sky130_fd_sc_hd__dfxtp_4)
           12.62   data arrival time

  12.00    12.00   clock clk (rise edge)
   0.00    12.00   clock network delay (ideal)
   0.00    12.00   clock reconvergence pessimism
           12.00 ^ _50075_/CLK (sky130_fd_sc_hd__dfxtp_4)
  -0.28    11.72   library setup time
           11.72   data required time
-----------------------------------------------------------
           11.72   data required time
          -12.62   data arrival time
-----------------------------------------------------------
           -0.91   slack (VIOLATED)


% report_tns
tns -7.71
% report_wns
wns -1.16
%
```

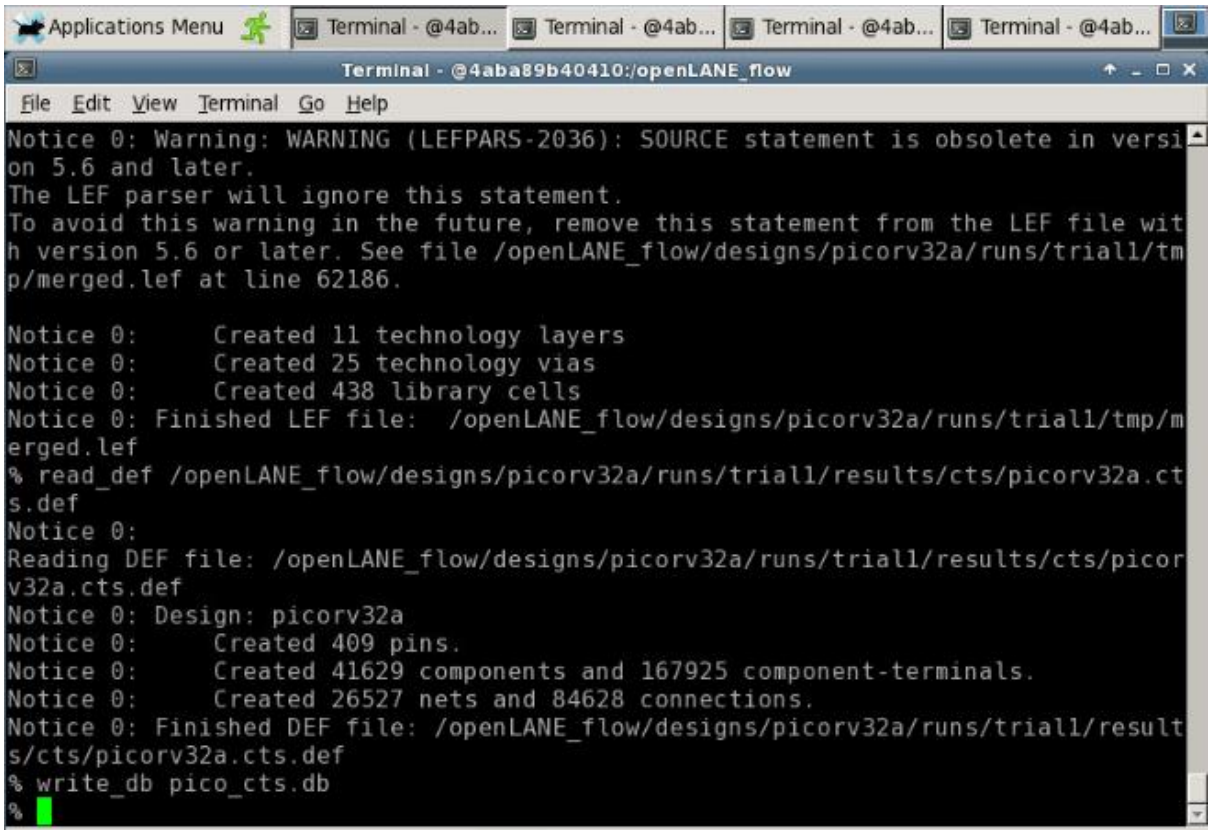
Lab3: Run CTS is completed successfully



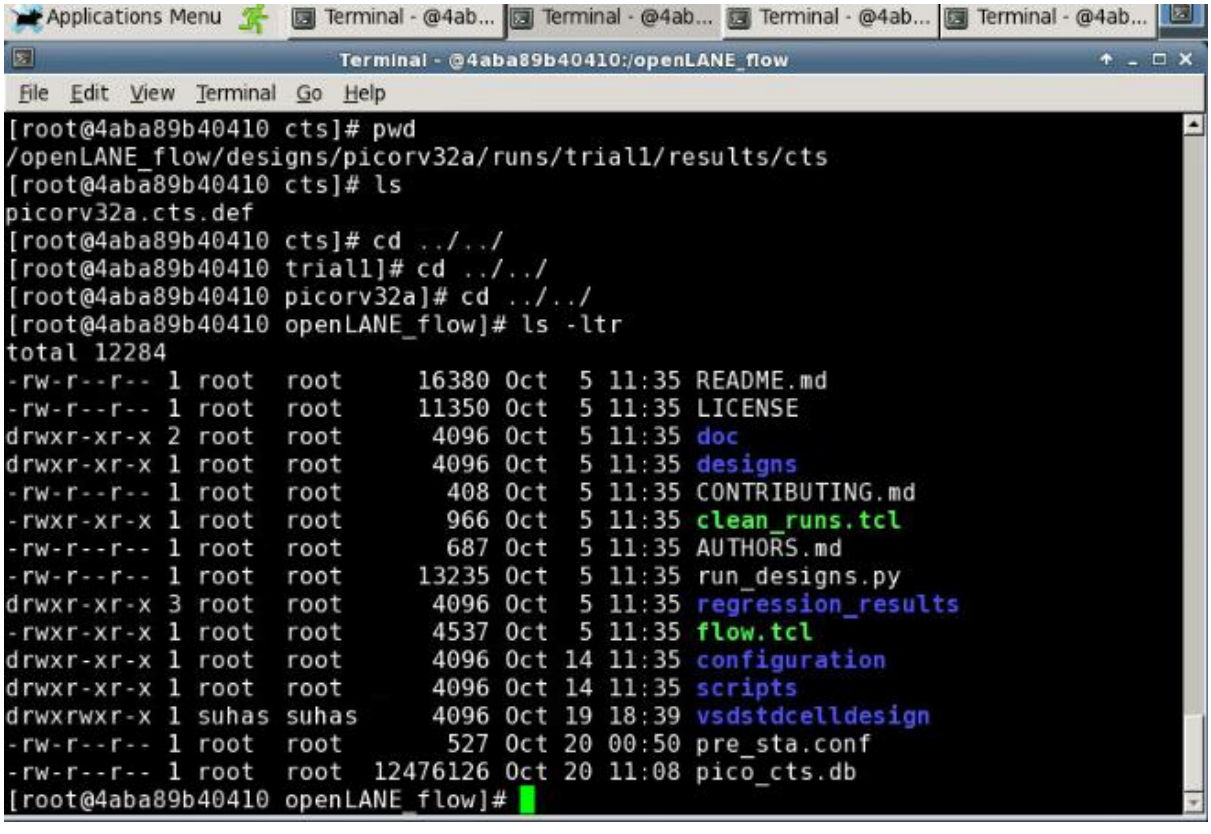
Lab4: CTS values



Creating db file



Db file created in OpenLANE_flow directory



Reading SDC file

```
% read_liberty -min $::env(LIB_MIN)
1
% read_sdc /openLANE_flow/designs/picorv32a/src/my_base.sdc
[INFO]: Setting output delay to: 2.4000000000000004
[INFO]: Setting input delay to: 2.4000000000000004
[INFO]: Setting load to: 0.01765
0
%
```

Slack value MET

Applications Menu | Terminal - @4ab... | Terminal - @4ab... | Terminal - @4ab... | Terminal - @4ab...

Terminal - @4aba89b40410:/openLANE_flow

File Edit View Terminal Go Help

```
   0.3445     3.7926 ^ _29067_/X (sky130_fd_sc_hd__buf_1)
   0.3445     4.1372 ^ _29128_/X (sky130_fd_sc_hd__buf_1)
   0.1776     4.3147 ^ _31098_/X (sky130_fd_sc_hd__buf_1)
   0.1560     4.4707 ^ _31099_/X (sky130_fd_sc_hd__buf_1)
   0.2349     4.7056 ^ _31100_/X (sky130_fd_sc_hd__buf_1)
   0.1725     4.8781 ^ _31101_/X (sky130_fd_sc_hd__or2_4)
   0.0524     4.9306 v _31102_/Y (sky130_fd_sc_hd__o21ai_4)
   0.1955     5.1261 ^ _31109_/Y (sky130_fd_sc_hd__a21oi_4)
   0.0000     5.1261 ^ mem_la_addr[2] (out)
              5.1261   data arrival time

  12.0000    12.0000   clock clk (rise edge)
   0.0000    12.0000   clock network delay (propagated)
   0.0000    12.0000   clock reconvergence pessimism
  -2.4000     9.6000   output external delay
              9.6000   data required time
-------------------------------------------------------------
              9.6000   data required time
             -5.1261   data arrival time
-------------------------------------------------------------
              4.4739   slack (MET)


%
```

Removed clkbuf_1

```
%
% exit
% echo $::env(CTS_CLK_BUFFER_LIST)
sky130_fd_sc_hd__clkbuf_1 sky130_fd_sc_hd__clkbuf_2 sky130_fd_sc_hd__clkbuf_4 sk
y130_fd_sc_hd__clkbuf_8
%
% lreplace $::env(CTS_CLK_BUFFER_LIST) 0 0
sky130_fd_sc_hd__clkbuf_2 sky130_fd_sc_hd__clkbuf_4 sky130_fd_sc_hd__clkbuf_8
% echo $::env(CTS_CLK_BUFFER_LIST)
sky130_fd_sc_hd__clkbuf_1 sky130_fd_sc_hd__clkbuf_2 sky130_fd_sc_hd__clkbuf_4 sk
y130_fd_sc_hd__clkbuf_8
% set ::env(CTS_CLK_BUFFER_LIST) [lreplace $::env(CTS_CLK_BUFFER_LIST) 0 0]
sky130_fd_sc_hd__clkbuf_2 sky130_fd_sc_hd__clkbuf_4 sky130_fd_sc_hd__clkbuf_8
%
```

Inserted clkbuf_1 to the clock buffers

```
% exit
% echo $::env(CTS_CLK_BUFFER_LIST)
sky130_fd_sc_hd__clkbuf_2 sky130_fd_sc_hd__clkbuf_4 sky130_fd_sc_hd__clkbuf_8
%  set ::env(CTS_CLK_BUFFER_LIST) [linsert $::env(CTS_CLK_BUFFER_LIST) 0 sky130_
fd_sc_hd__clkbuf_1]
sky130_fd_sc_hd__clkbuf_1 sky130_fd_sc_hd__clkbuf_2 sky130_fd_sc_hd__clkbuf_4 sk
y130_fd_sc_hd__clkbuf_8
% echo $::env(CTS_CLK_BUFFER_LIST)
sky130_fd_sc_hd__clkbuf_1 sky130_fd_sc_hd__clkbuf_2 sky130_fd_sc_hd__clkbuf_4 sk
y130_fd_sc_hd__clkbuf_8
%
```

Change in slack due to clkbuf_2

slack values

File Edit Search Options Help

```
INITIAL
h slack 0.2929   slack (MET)
s slack 4.4739   slack (MET)

SECOND -- using clock_buf_2
h slack 0.3971  slack (MET)
s slack 4.5941 slack (MET)
hold skew 0.17
setup skew 0.17
```

6. Day 5: Final steps for RTL2GDS using TritonRoute and OpenSTA

On Day5, explanation of how routing process done using TritonRoute and explanation of SPEF file extraction in done. Routing method finds out best possible pattern for connection between two end points, of which one point is *target node* while the other is *source node*. *Maze Routing-Lee's Algorithm* was introduced. In this technique firstly routing grids are created and source & target nodes are identified. Then the blocks adjacent to one under consideration are assigned same numbers and this process is repeated till we reach the target node. Once this done the pattern with minimum number of turns preferably a *l* spaced pattern is finalised for route.

Typical DRC rules for pair of wires:

- Wire width
- Wire pitch
- Wire Spacing

These DRC rules exist because of limitations of the Lithography technique. Deviation takes place in lithography leading to changes which might lead to an unintended open circuit or short circuit and to avoid this the DRC rules are fixed. Another DRC Violation type is Signal short which can be dealt by changing layers.

Routing Technique Classified into 2 steps:

- Global Route: Performed using fast route. In this step route guides are formed.
- Detailed Route: Performed using *TritonRoute*.

TritonRoute performs *initial detailed route* and tries to route within the route guide provided by fast route. It works on MILF-based panel routing with intra-layer parallel route and inter-layer sequential route technique. Input files required for triton route are LEF, DEF and *pre-processed route* guide. Output is in form of detailed routing with optimum wire length and Via count.

Steps for *pre-processed* route guides:

- Initial route guide
- Splitting
- Merging
- Bridging
- Pre-processed Route

LAB:

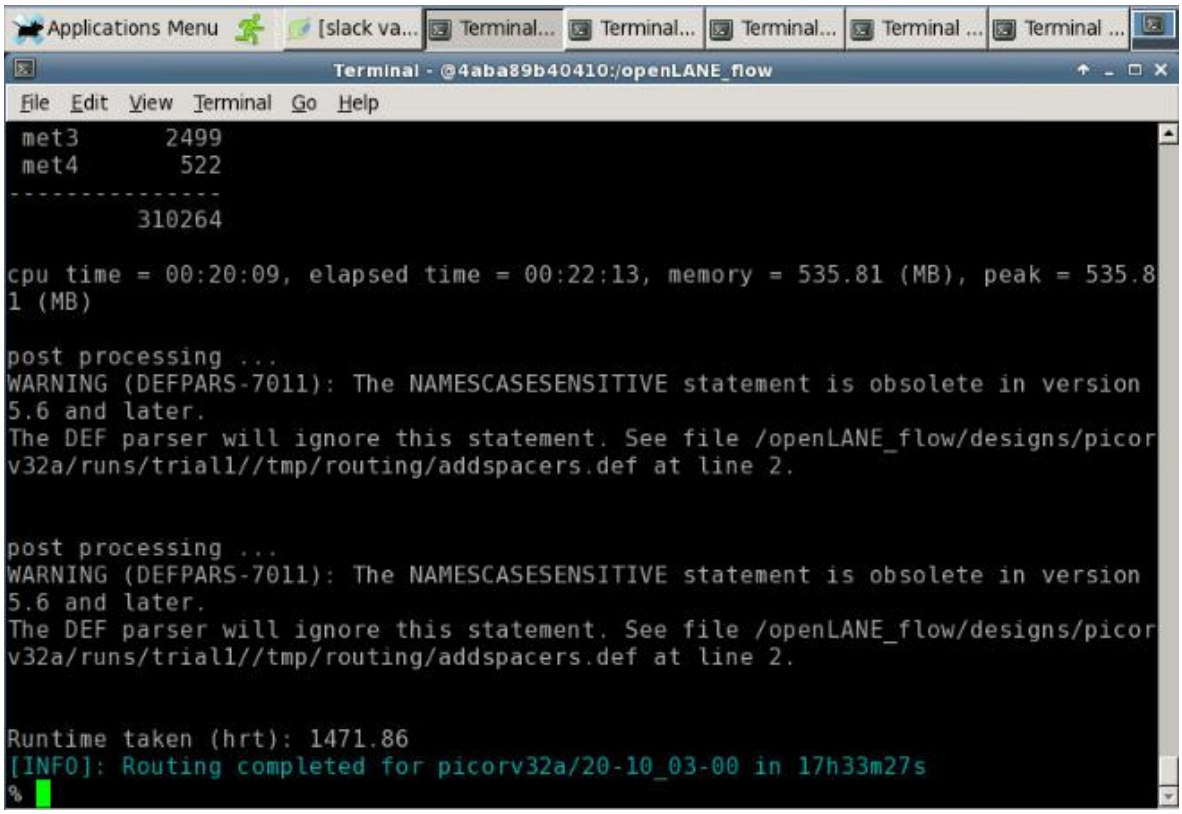
Generating PDN

```
% echo $::env(CURRENT_DEF)
/openLANE_flow/designs/picorv32a/runs/trial1//results/cts/picorv32a.cts.def
% gen_pdn
```

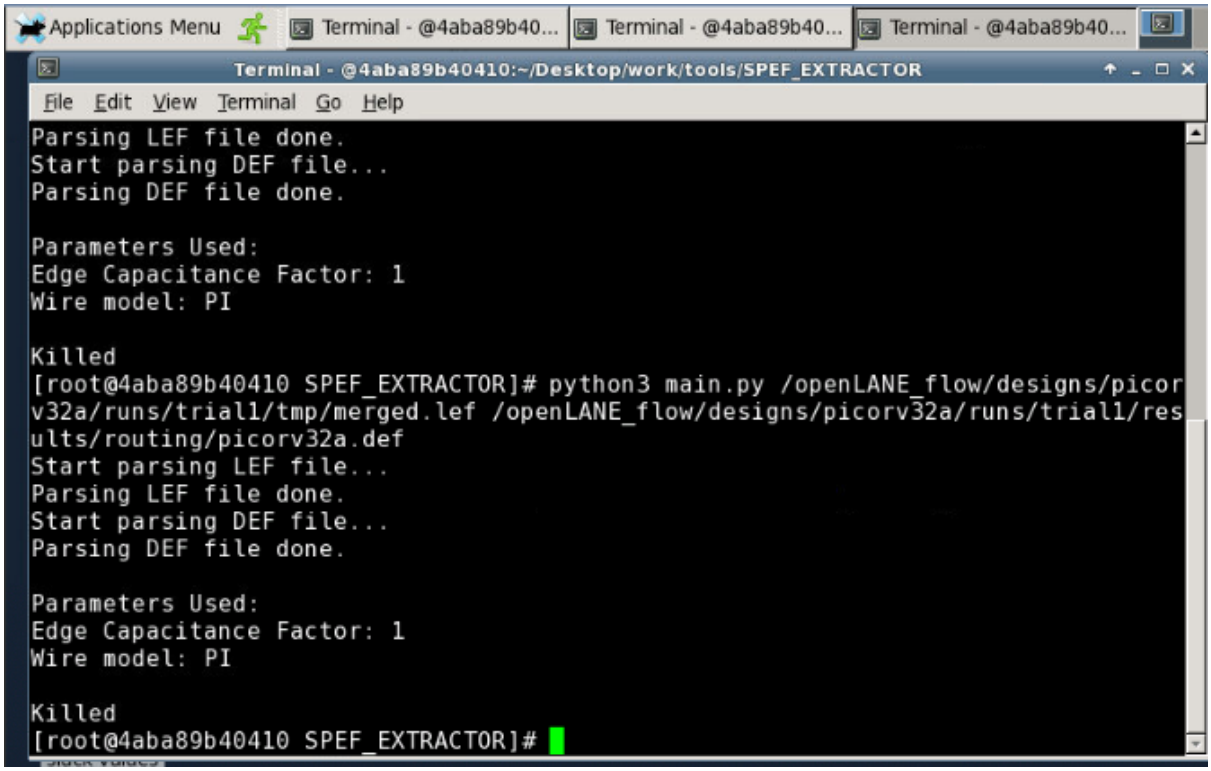
PDN generation completed successfully

```
Applications Menu  [slack va...  Terminal...  Terminal ...  Terminal ...  Terminal ...  Terminal ...
Terminal - @4aba89b40410:/openLANE_flow
File  Edit  View  Terminal  Go  Help
s/cts/picorv32a.cts.def
[INFO] [PDNG-0016] Power Delivery Network Generator: Generating PDN
[INFO] [PDNG-0016]   config: /pdks/sky130A/libs.tech/openlane/common_pdn.tcl
[INFO] [PDNG-0008] Design Name is picorv32a
[INFO] [PDNG-0009] Reading technology data
[INFO] [PDNG-0011] ****** INFO ******
Type: stdcell, grid
    Stdcell Rails
      Layer: met1 -  width: 0.480  pitch: 2.720  offset: 0.000
    Straps
      Layer: met4 -  width: 1.600  pitch: 153.600  offset: 16.320
      Layer: met5 -  width: 1.600  pitch: 153.180  offset: 16.650
    Connect: {met1 met4} {met4 met5}
Type: macro, macro_1
    Macro orientation: R0 R180 MX MY R90 R270 MXR90 MYR90
    Straps
    Connect:
[INFO] [PDNG-0012] **** END INFO ****
[INFO] [PDNG-0013] Inserting stdcell grid - grid
[INFO] [PDNG-0015] Writing to database
% echo $::env(CURRENT_DEF)
/openLANE_flow/designs/picorv32a/runs/trial1//tmp/floorplan/pdn.def
% echo $::env(ROUTING_STRATEGY)
0
% run_routing
```

Run routing is completed successfully



SPEF extraction (mine got killed)



7. Acknowledgements

- Kunal Ghosh, Co-founder (VSD Corp. Pvt. Ltd)