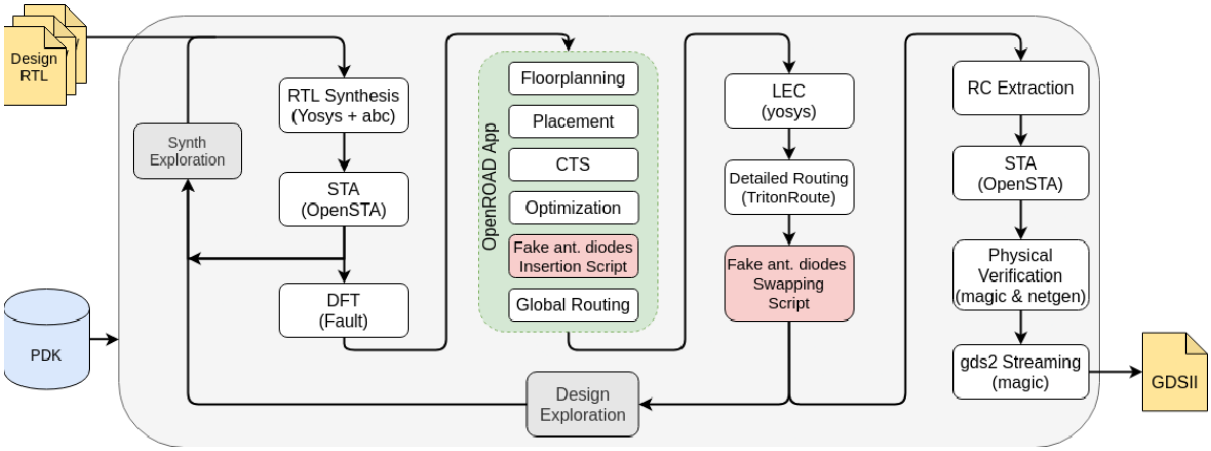
Openlane-Sky130-workshop

An informative workshop on advanced Physical Design using OpenLANE/Sky130 organized by VSD Corp. The workshop focused on building basics by imparting exposure to conceptual as well as practical approach. Learning's: Open source tools, Characterization of Cell, SPICE simulations, Static timing Analysis concepts, Commands of various OpenLANE tools, 16-mask CMOS Fabrication, Antenna Diode Concept, Decoupling Cap, Routing algorithm & Steps, Timing Characterization and familiarity with files like libs/db, LEF, DEF, SDC and SPEF. Tools used: Magic, Ngspice, OpenSTA, TritonRoute, yosys, SPEF extractor and OpenROAD.

Contents

Introduction to OpenLane

OpenLANE is an automated RTL to GDSII flow based on several components including OpenROAD, Yosys, Magic, Netgen, Fault, OpenPhySyn, SPEF-Extractor and custom methodology scripts for design exploration and optimization. It is a tool started for true open source tape-out experience and comes with APACHE version 2.0. The goal of OpenLANE is to produce clean GDSII without any human intervention. OpenLANE is tuned for Skywater 130nm open PDK and can be used to produce hard macros and chips.



The Flow of OpenLane EDA

OpenLane Design Stages

OpenLane flow consists of several stages. By default all flow steps are run in sequence. Each stage may consist of multiple sub-stages.

1. **Synthesis**
    a) Yosys/abc - Perform RTL synthesis and technology mapping.
    b) OpenSTA - Performs static timing analysis on the resulting Netlist to generate timing reports
2. **Floorplaning**
    a) init_fp - Defines the core area for the macro as well as the rows (used for placement) and the tracks (used for routing)
    b) ioplacer - Places the macro input and output ports
    c) pdngen - Generates the power distribution network
    d) tapcell - Inserts welltap and decap cells in the floorplan
3. **Placement**
    a) RePLace - Performs global placement
    b) Resizer - Performs optional optimizations on the design
    c) OpenDP - Performs detailed placement to legalize the globally placed components
4. **CTS**
    a) TritonCTS - Synthesizes the clock distribution network (the clock tree)
5. **Routing**
    a) FastRoute - Performs global routing to generate a guide file for the detailed router
    b) TritonRoute - Performs detailed routing
    c) OpenRCX - Performs SPEF extraction
6. **Tapeout**

a) Magic - Streams out the final GDSII layout file from the routed def

b) KLayout - Streams out the final GDSII layout file from the routed def as a back-up

7. **Signoff**

a) Magic - Performs DRC Checks & Antenna Checks

b) KLayout - Performs DRC Checks

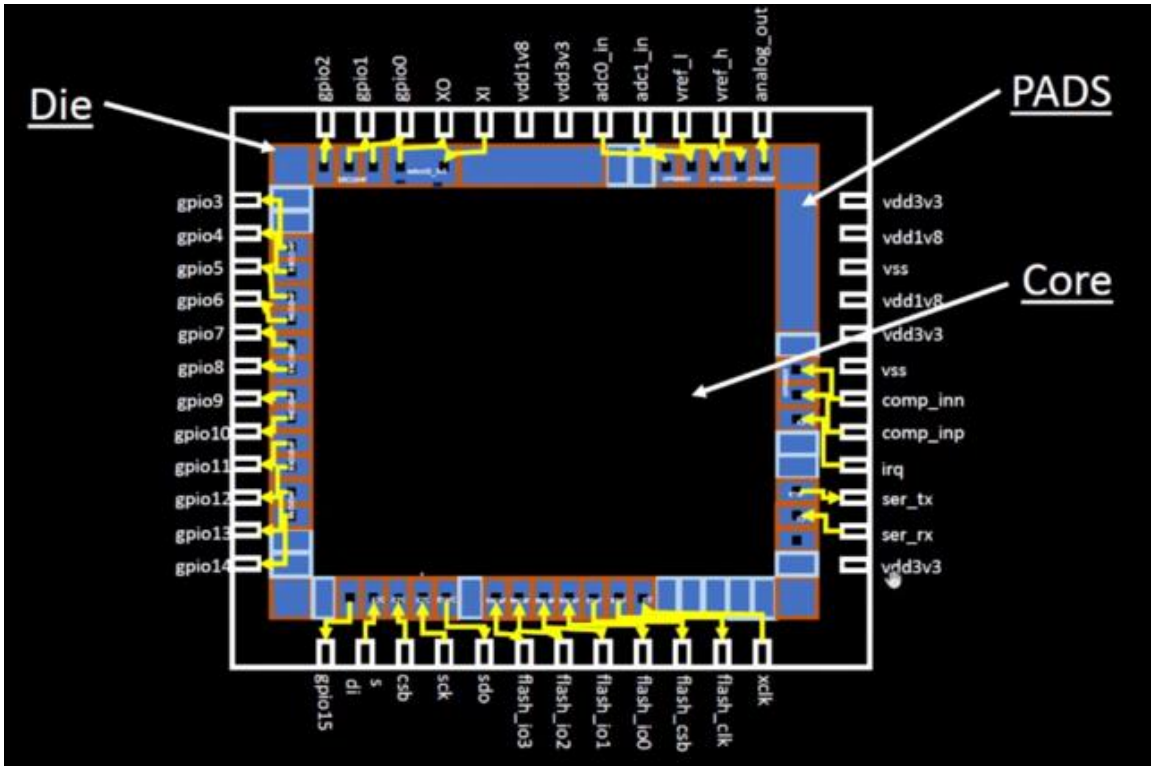
c) Netgen - Performs LVS Checks

d) CVC - Performs Circuit Validity Checks

Day 1: Introduction of chip architecture and insights into Physical Design.

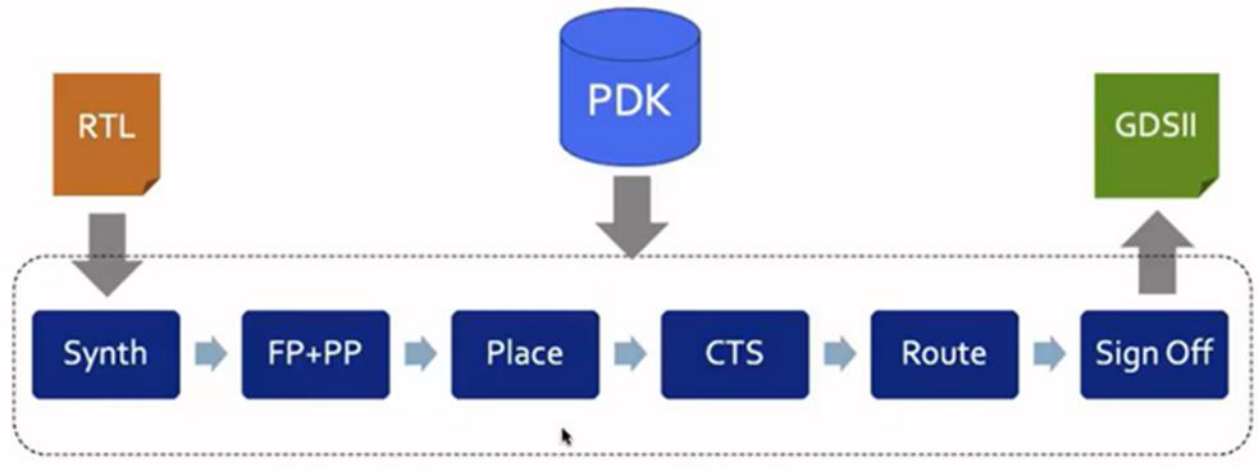
The chip will be sitting at the center of the package. Chip is connected to the package, as shown in the above figure.

- PADS – It is an I/O port which can be used send and receive the signals via PADS.
- CORE – The Foundry IP's as well as many other digital and analog blocks are placed in the chip.
- DIE – The size of the entire chip. Die will be manufactured on the silicon wafer.



Introduction into Physical Design:

In integrated circuit design, **physical design** is a step in the standard design cycle which follows after the circuit design. At this step, circuit representations of the components (devices and interconnects) of the design are converted into geometric representations of shapes which, when manufactured in the corresponding layers of materials, will ensure the required functioning of the components. This geometric representation is called integrated circuit layout. This step is usually split into several sub-steps, which include both design and verification and validation of the layout.



There are mainly six steps in Physical design:

1. **Synthesis :**
   Physical design is based on a netlist which is the end result of the synthesis process. Synthesis converts the RTL design usually coded in VHDL or Verilog HDL to gate-level descriptions which the next set of tools can read/understand. This netlist contains information on the cells used, their interconnections, area used, and other details.

2. **Floorplanning :**
   The second step in the physical design flow is floorplanning. Floorplanning is the process of identifying structures that should be placed close together, and allocating space for them in such a manner as to meet the sometimes conflicting goals of available space (cost of the chip), required performance, and the desire to have everything close to everything else. Based on the area of the design and the hierarchy, a suitable floorplan is decided upon. Floorplanning takes into account the macros used in the design, memory, other IP cores and their placement needs, the routing possibilities, and also the area of the entire design. Floorplanning also determines the IO structure and aspect ratio of the design. A bad floorplan will lead to wastage of die area and routing congestion.

3. **Placement**

Placement uses RC values from Virtual Route (VR) to calculate timing. VR is the shortest Manhattan distance between two pins. VR RCs are more accurate than WLM RCs.
Placement is performed in four optimization phases:

- Pre-placement optimization
- In placement optimization
- Post Placement Optimization (PPO) before clock tree synthesis (CTS)
- PPO after CTS.

Pre-placement Optimization optimizes the netlist before placement, HFNs (High Fanout Nets) are collapsed. It can also downsize the cells. In-placement optimization re-optimizes the logic based on VR. This can perform cell sizing, cell moving, cell bypassing, net splitting, gate duplication, buffer insertion, area recovery. Optimization performs iteration of setup fixing, incremental timing and congestion driven placement. Post placement optimization before CTS performs netlist optimization with ideal clocks. It can fix setup, hold, max trans/cap violations. It can do placement optimization based on global routing. It re does HFN synthesis. Post placement optimization after CTS optimizes timing with propagated clock. It tries to preserve clock skew.

4. **Clock tree synthesis**
   The goal of **clock tree synthesis** (**CTS**) is to minimize skew and insertion delay. Clock is not propagated before CTS as shown in the picture. After CTS hold slack should improve. Clock tree begins at .sdc defined clock source and ends at stop pins of flop. There are two types of stop pins known as ignore pins and sync pins. 'Don't touch' circuits and pins in front end (logic synthesis) are treated as 'ignore' circuits or pins at back end (physical synthesis). 'Ignore' pins are ignored for timing analysis. If clock is divided then separate skew analysis is necessary.

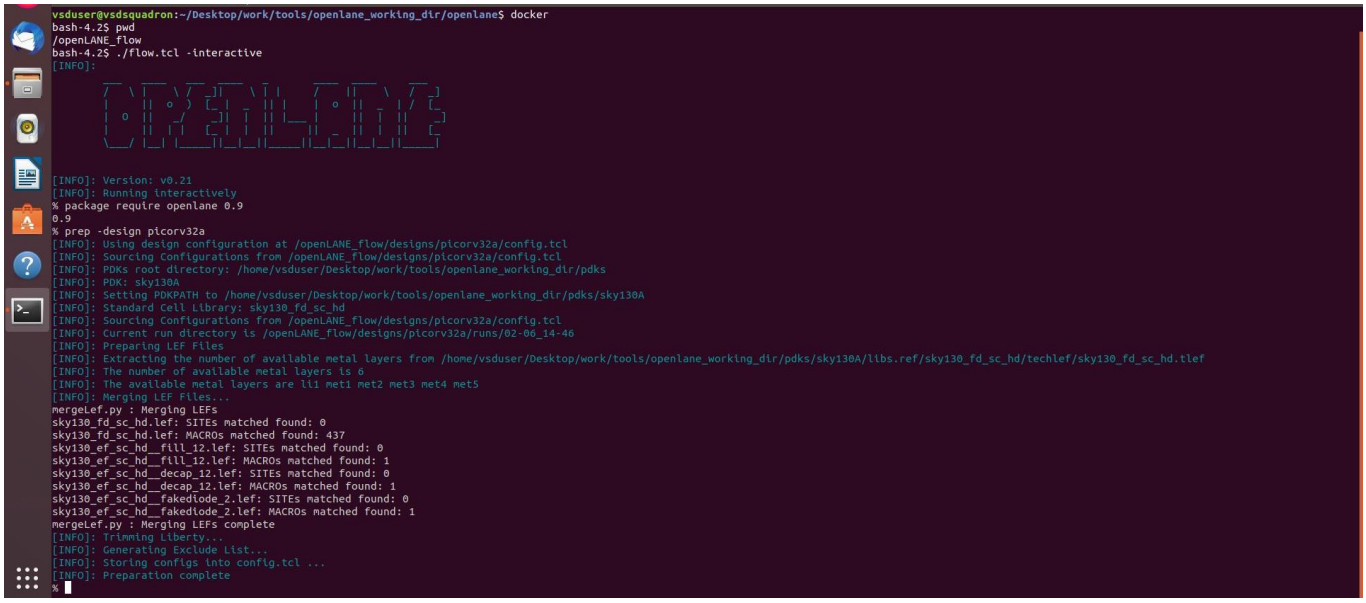
5. **Routing**
   There are two types of routing in the physical design process, global routing and detailed routing. Global routing allocates routing resources that are used for connections. It also does track assignment for a particular net. Detailed routing does the actual connections. Different constraints that are to be taken care during the routing are DRC, wire length, timing etc.

OpenLANE LAB:

The working Directory will be,

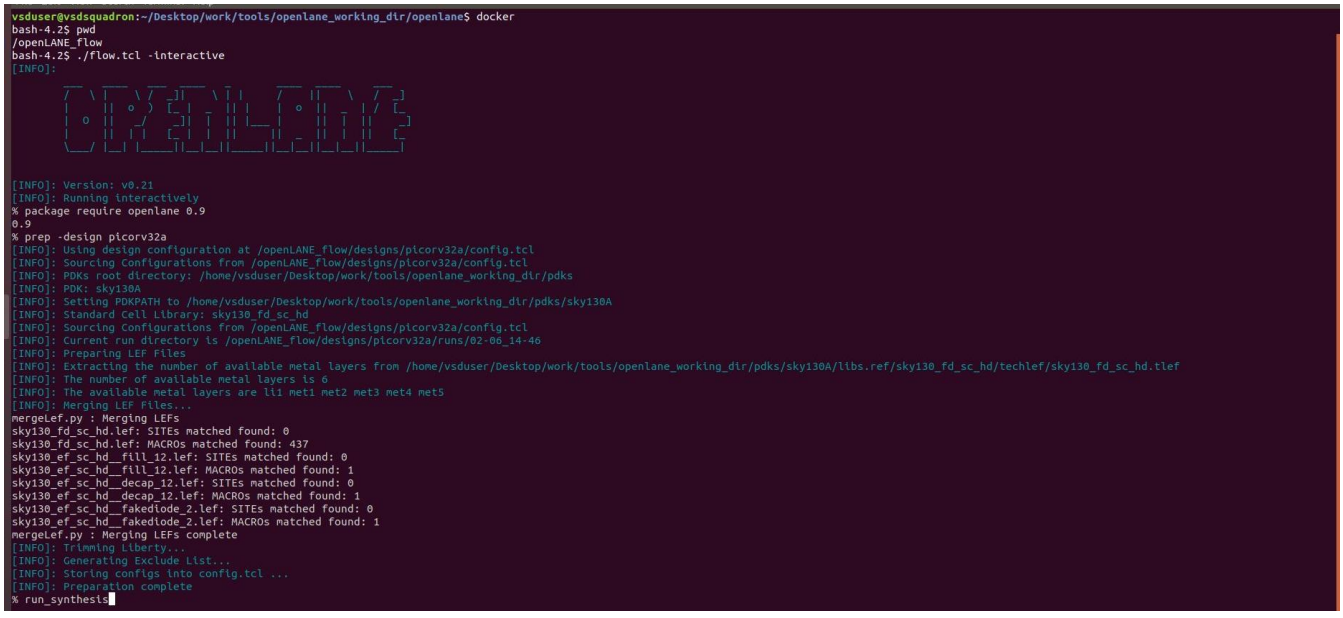
```
vsduser@vsdsquadron:~/Desktop/work/tools/openlane_working_dir/openlane$
```

The commands used enter into OpenLANE directory is,

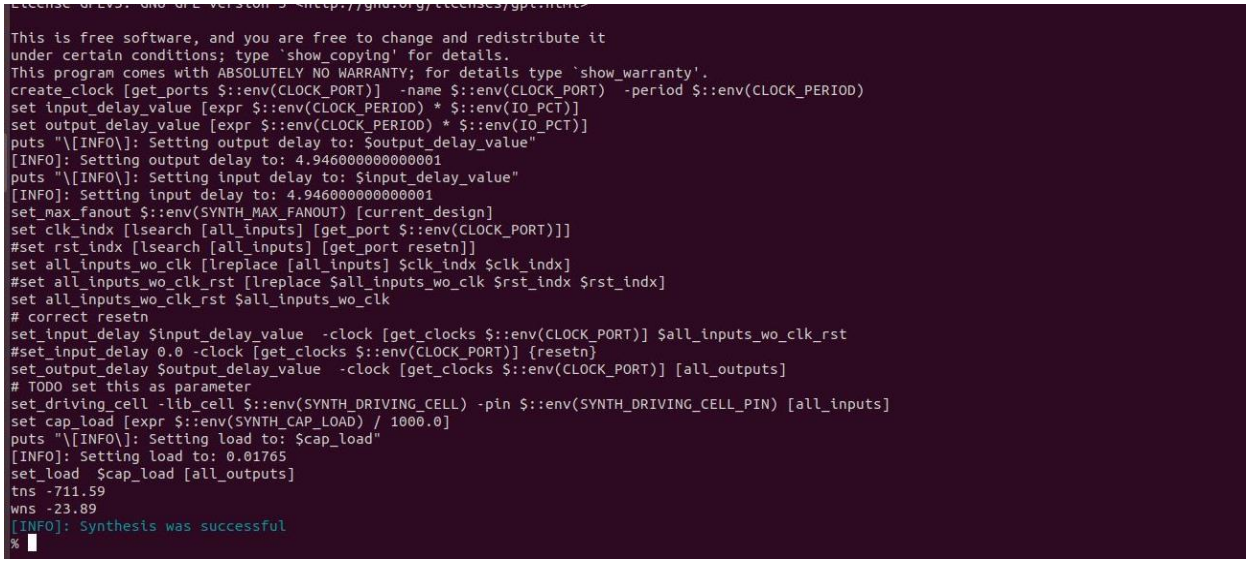


To run synthesis,

Command: run_synthesis



After synthesis step,



Day 2: Chip floorplan and Introduction to Library Cells

On Day 2, definition of width and height of core and die. Factors like Utilization factor and Aspect ratio important to understand a design were introduced and their effects were discussed. Steps involved to define location of pre-placed cell & it's advantage of enhancing reusability and de-coupling capacitor and how they help during switching to avoid failure explained. A fully charged De-coupling cap placed parallel to circuits to ensure proper supply of peak current Ipeak by decoupling them from main supply voltage. Hence de-coupling cap ensures proper local communication while multiple Vdd & Vss lines lead to proper global communication avoiding voltage droop and ground bounce conditions. Step of placing logical cell placement blockage to avoid PnR tool to place anything.

Characterization

Input information required by Characterization software's are PDKs, DRC & LVS rules and spice models. The design steps of it involve *Circuit Design* and *Layout design* characterization. The software GUNA used for characterization. The characterization can be classified as Timing characterization, Power characterization and Noise characterization.

Characterization flow steps

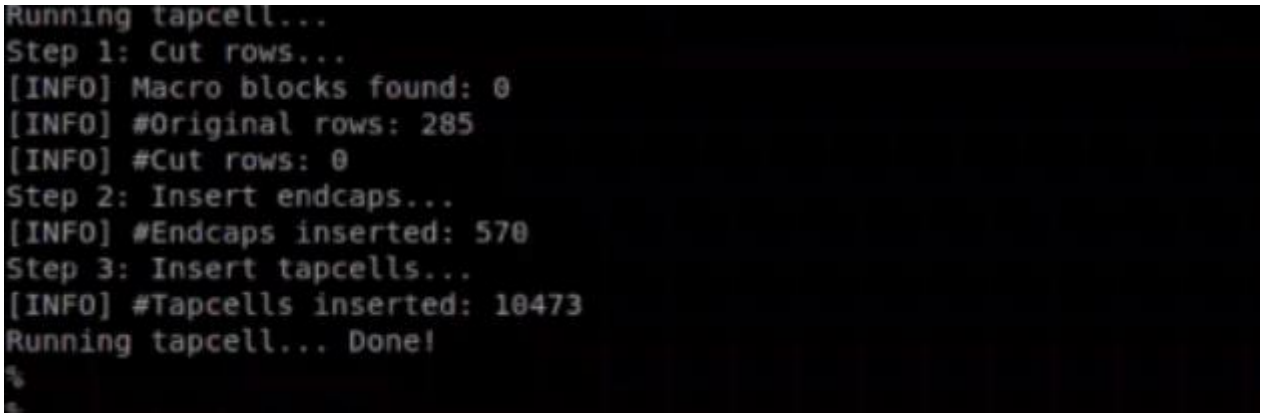
- Model file of CMOS containing basic property definitions
- Read extracted Spice Netlist
- Recognize the behaviour of cell
- Read the subcircuits
- Attract the power sources
- Apply input or stimulus

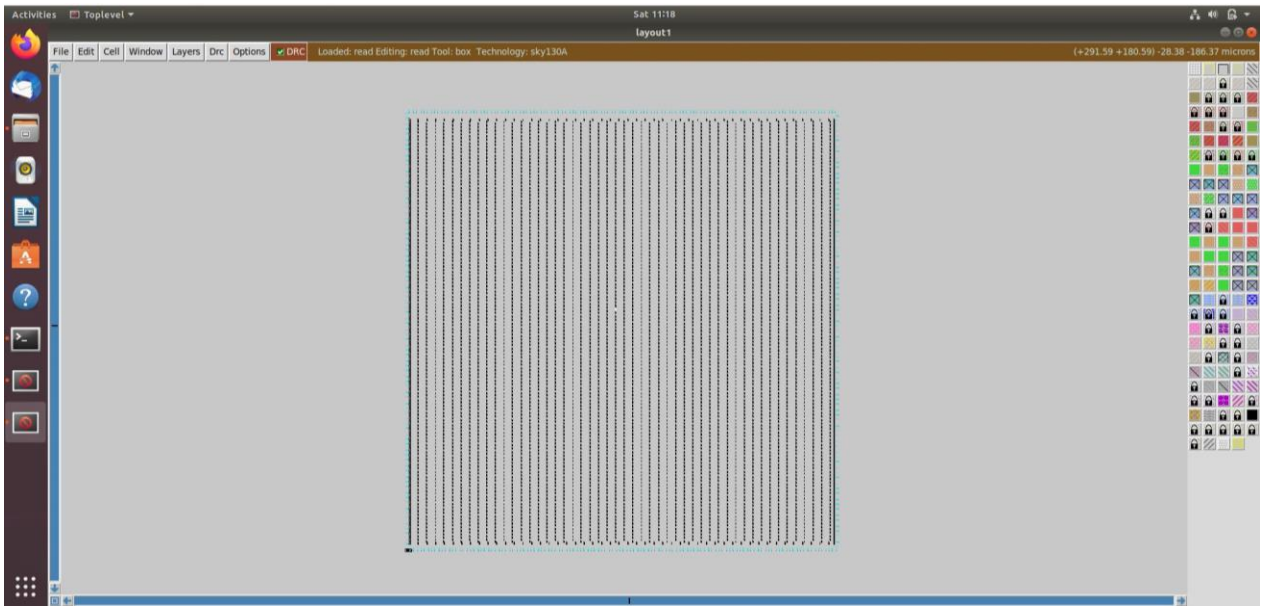
- Provide necessary output capacitance
- Provide necessary simulation commands

To run Floorplan stage

Command : run_floorplan

Floorplan Result





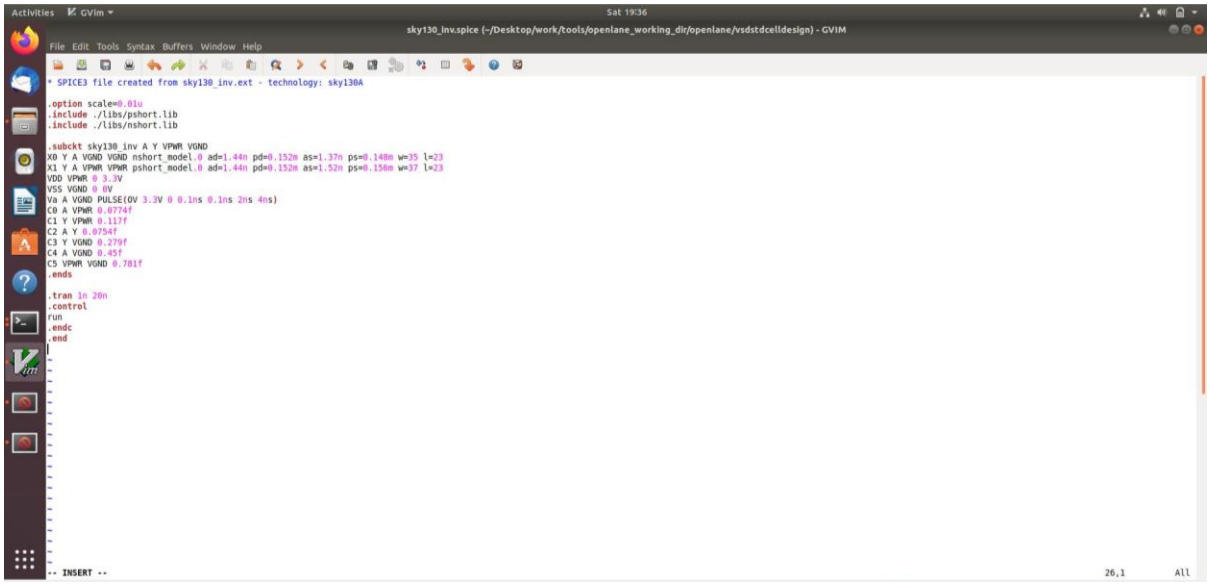
Floorplan of picorv32a

Day 3: Design and Characterization of cells using Magic Layout tool and Ngspice

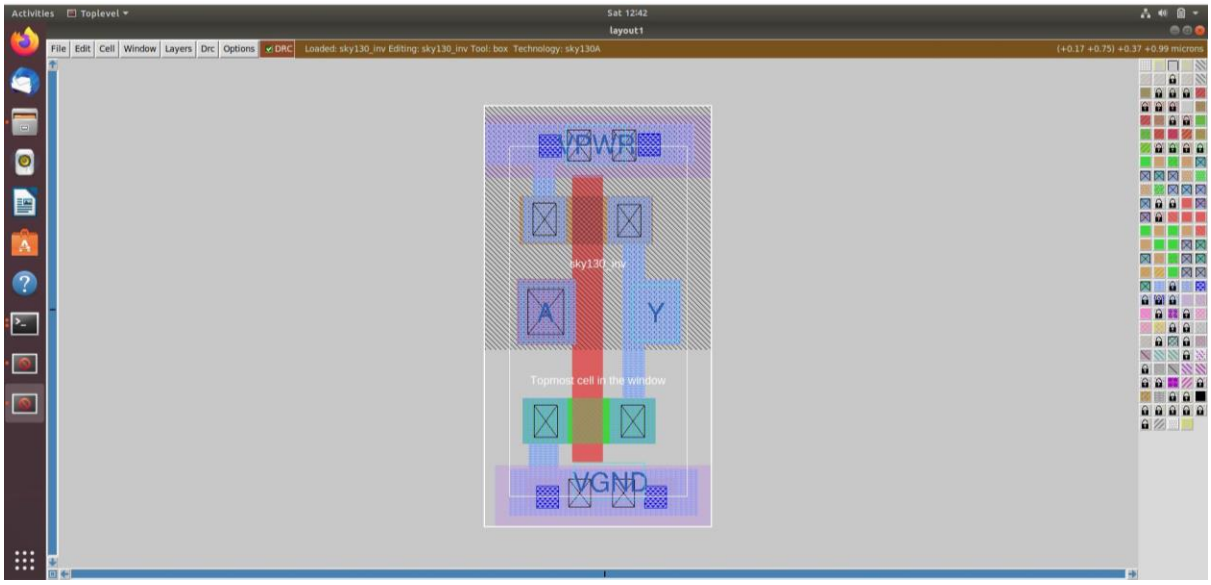
OpenLANE offers an interesting feature of making changes into parameters on the go. This helps to deal with issues like congestion. SPICE deck formation contains information like components connectivity and values and information about nodes. It was showed how W/L ratio of MOS impacts its conductivity 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 parameter that is **Switching Threshold (Vm)**.Here name of inverter is vsdshinv.

Spice deck Description of vsdshinv:



Layout view of vsdshinv:



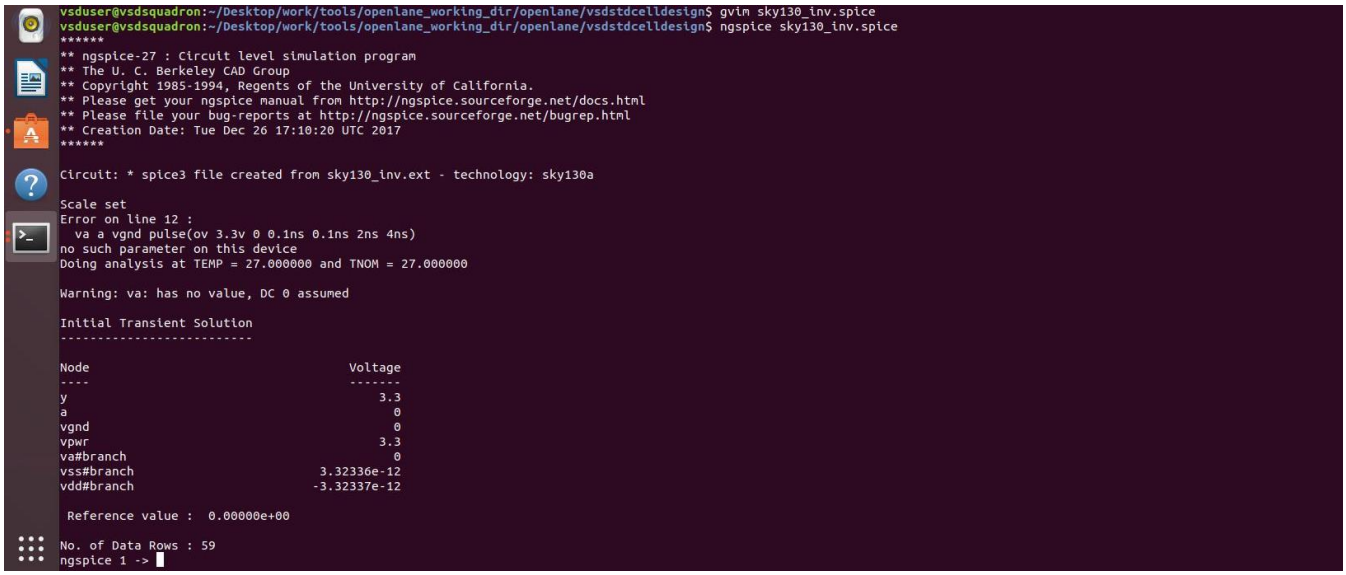
16-Mask CMOS Process Steps

- **Substrate Selection**: Selection of base layer on which other regions will be formed.
- **Create active region for transistors**: SiO2 and si3N2 deposited. Pockets created using photoresist and lithography.
- **Nwell & Pwell formation**: Pwell uses boron and nwell uses phosphorous. Drive in diffusion by placing in high temp furnace.

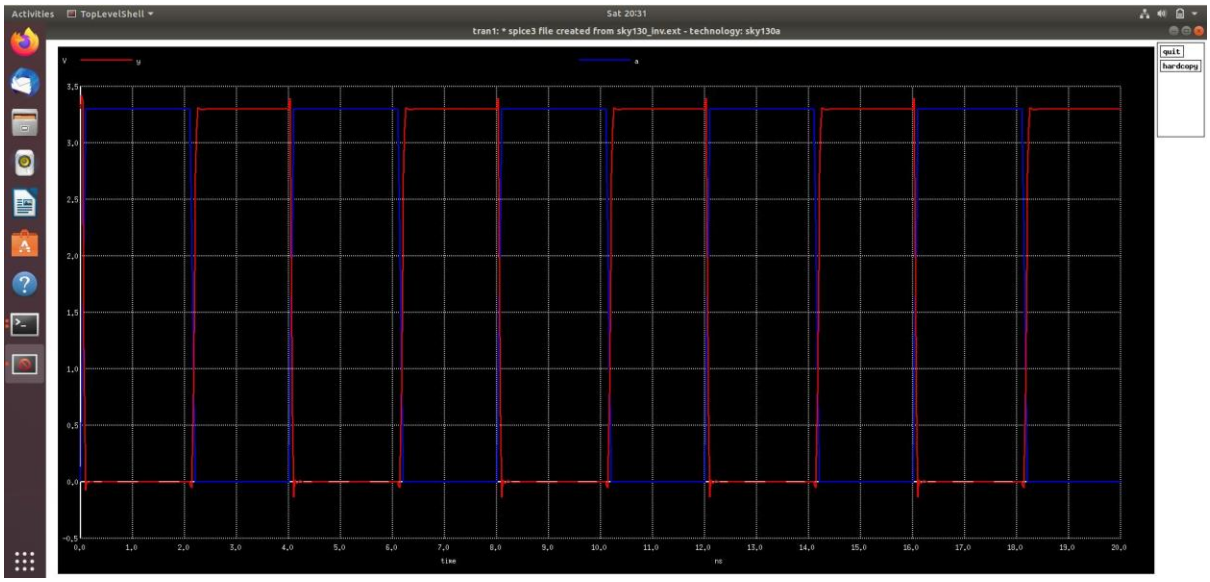
- **Creating Gate terminal**: For desired *threshold value* NA (doping Concentration) and Cox to be set.
- **Lightly Doped Drain (LDD) formation**: LDD done to avoid *hot electron effect* and *short channel effect*.
- **Source and Drain formation**: Forming the source and drain.
- **Contacts & local interconnect Creation**: SiO2 removed using HF etch. *Titanium* deposited using sputtering.
- **Higher Level metal layer formation**: Upper layers of metals deposited.

Ngspice environment:

To enter into Ngspice : command ngspice file_name.spice



Transient analysis of vsdshinv:

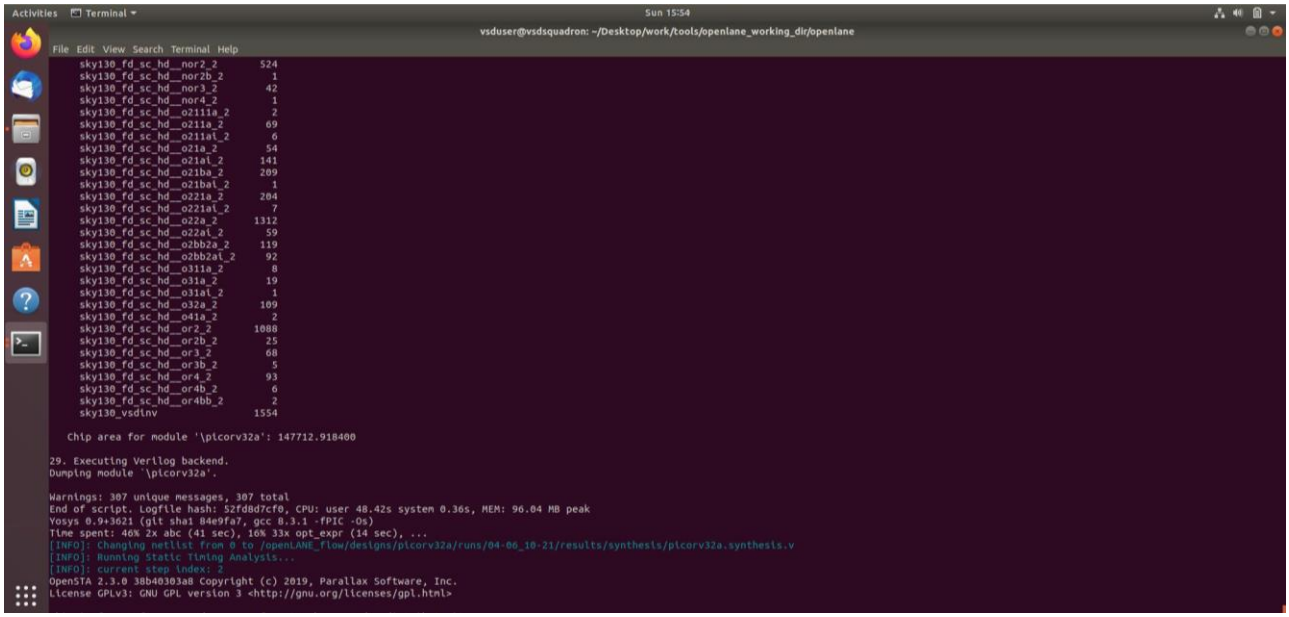


Day 4: Timing Analysis using Triton Clock Tree Synthesis (CTS).

Topic of Delay tables explored on Day 5. 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. With the help of interpolation and extrapolation the values of point in range can also be calculated for precise result.

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.

After addind vsdshinv to merged.lef file. The synthesis and floorplan is done again to get better timing results.



Commands to change configuration variables settings

set ::env(SYNTH_STRATEGY) 1

set ::env(SYNTH_SIZING) 1

OpenROAD Commands

read_lef /openLANE_flow/designs/picorv32a/runs/tmp/merged.lef

read_def /openLANE_flow/designs/picorv32a/runs/results/cts/picorv32a.cts.def

write_db pico12.db

read_verilog /openLANE_flow/designs/picorv32a/runs/results/synthesis/picorv32a.synthesis_cts.v

read_liberty $::env(LIB_SYNTH_COMPLETE)

link_design picorv32a

read_sdc /openLANE_flow/designs/picorv32a/src/my_base.sdc

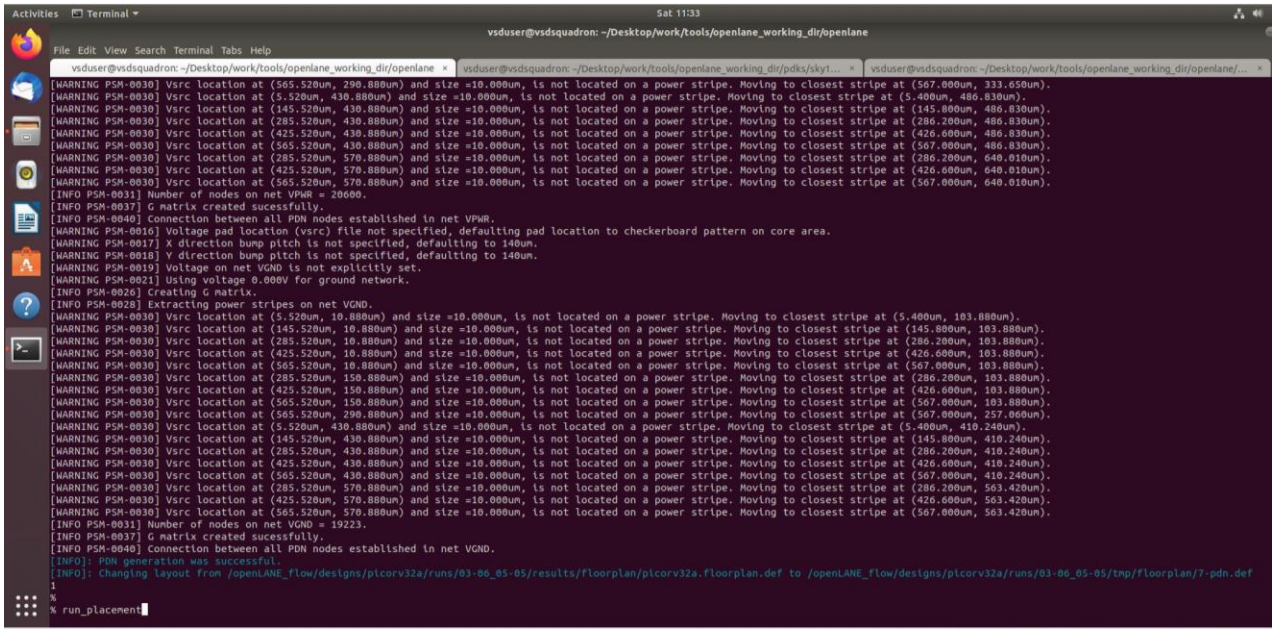
set_propagated_clock [all_clocks]

report_checks -path_delay min_max -digits 4

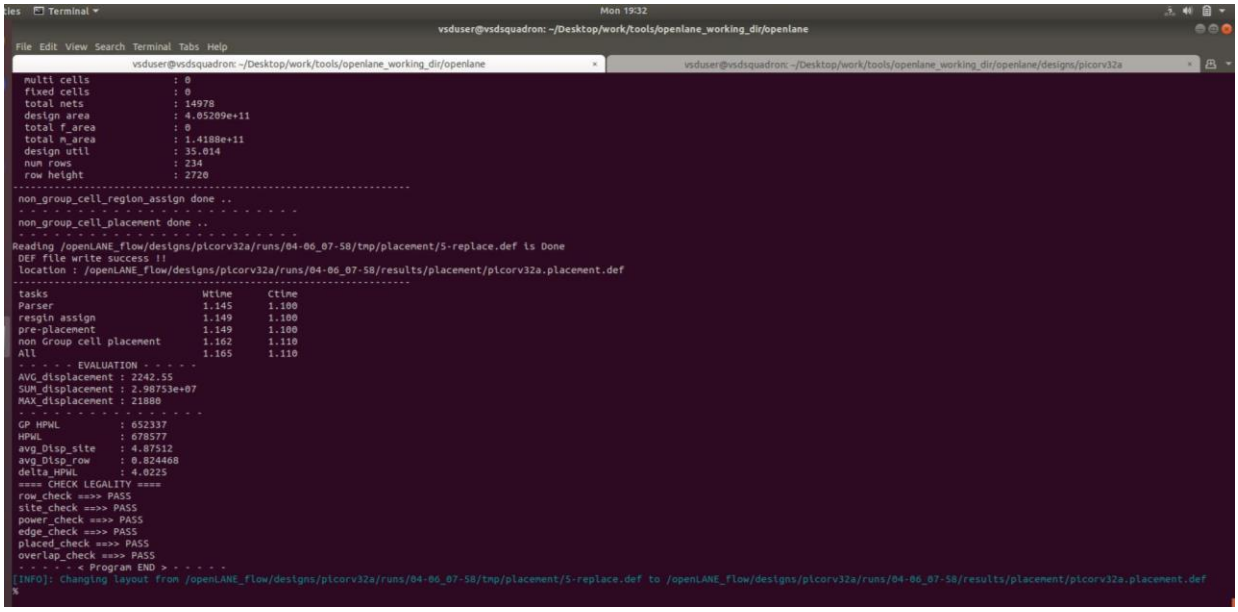
sky130_fd_sc_hd File containing layer information of Cell

```
li1 X 0.23 0.46
li1 Y 0.17 0.34
met1 X 0.17 0.34
met1 Y 0.17 0.34
met2 X 0.23 0.46
met2 Y 0.23 0.46
met3 X 0.34 0.68
met3 Y 0.34 0.68
met4 X 0.46 0.92
met4 Y 0.46 0.92
met5 X 1.70 3.40
met5 Y 1.70 3.40
~
~
```

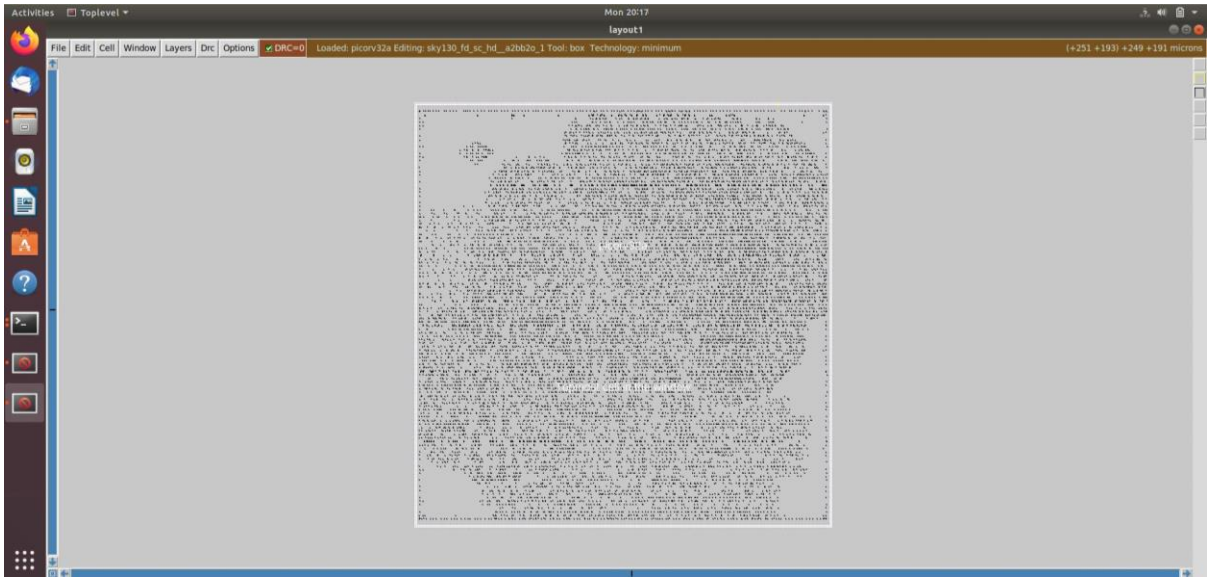
Placement results of picorv32a:



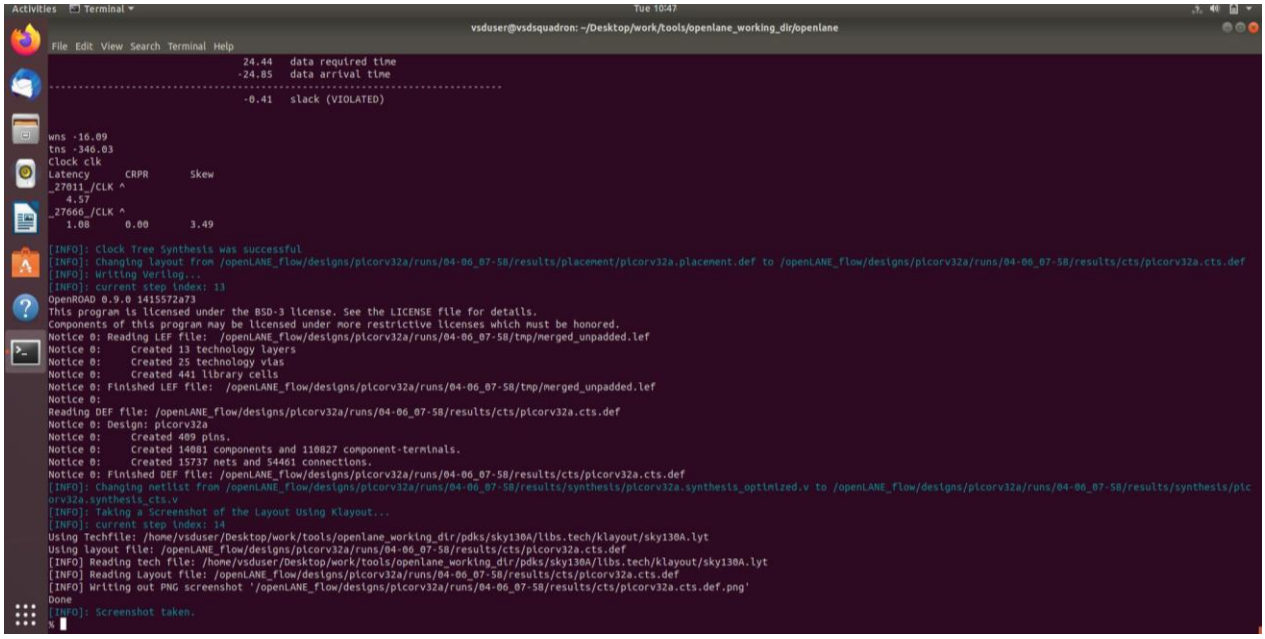
After the run:



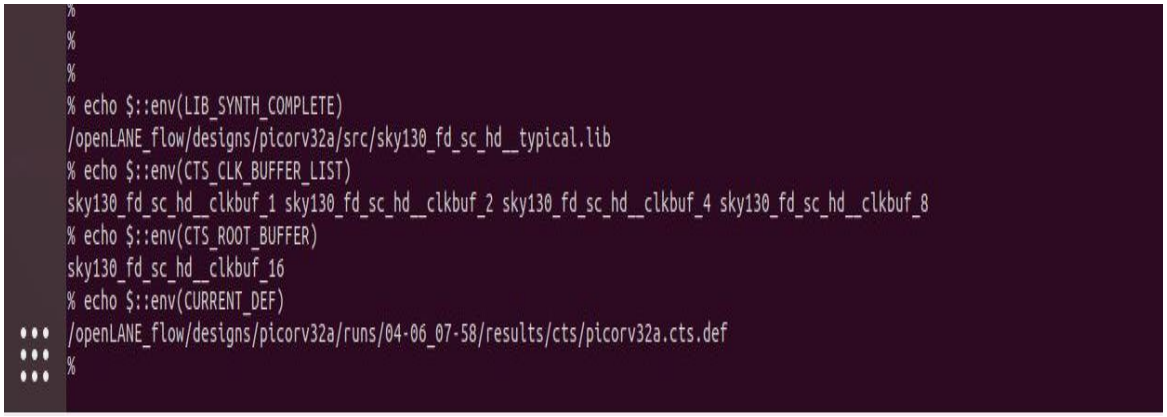
Layout view of placement.def,



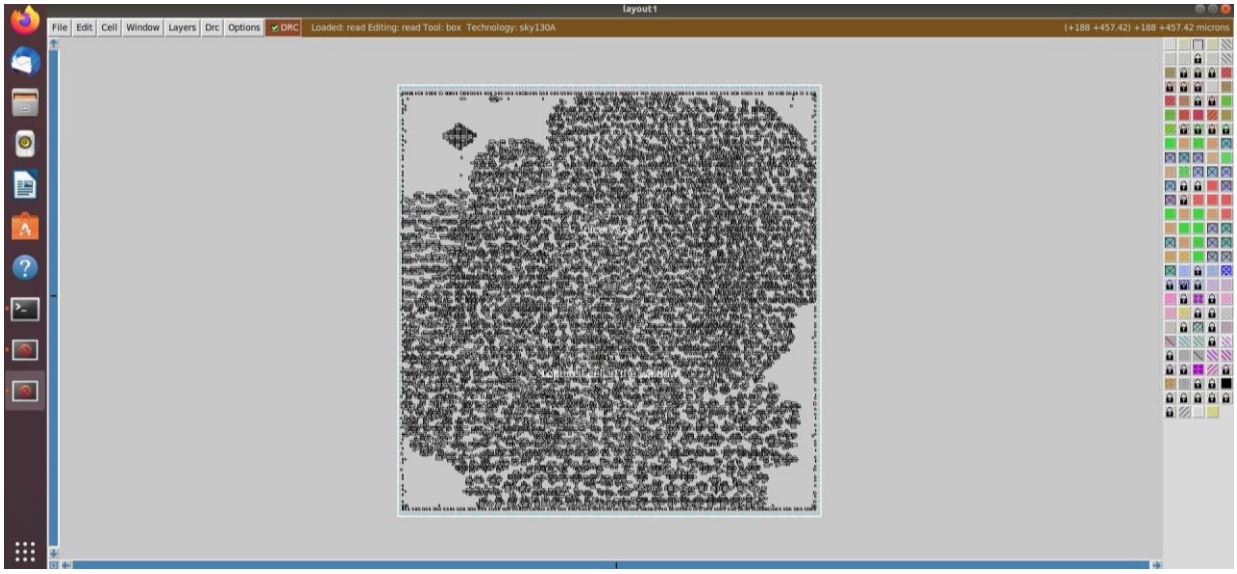
CTS run of Picorv32a:



After CTS run,



Layout view of picorv32a.cts.def



Day 5: Creation of Power Distribution Network, Routing and SPEF Extraction.

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 indentified. 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

1. Wire width
2. Wire pitch
3. 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

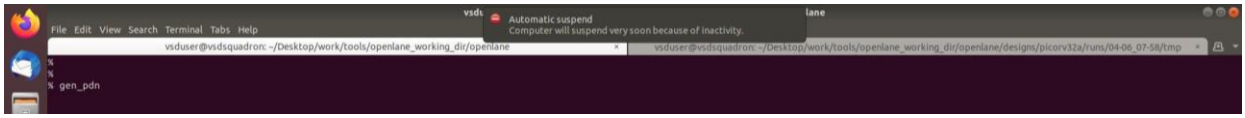
- **Global Route** : Performed using fast route. In this step route guides are formed.
- **Detailed Route** : Performed using *TritonRoute*.

Triton Route It 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

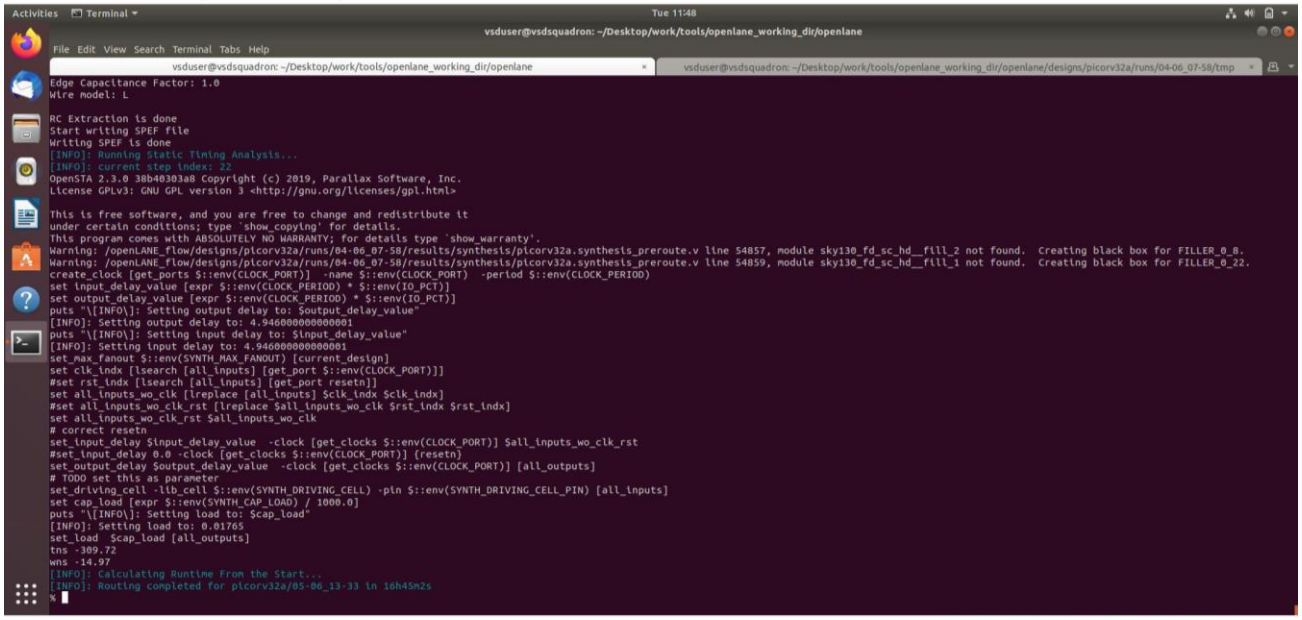
- **Initial route guide** : Basic connecting information using route guides.
- **Splitting** : Routes in non-preferred direction are split into unit width.
- **Merging** : Touching guides have edge orthogonal to the preferred route guide direction are merged.
- **Bridging** : Edges parallel to preferred one are bridged using additional layers.
- **Pre-processed Route** : Now all route guides are in preferred direction as required.

Power Planning of picorv32a,

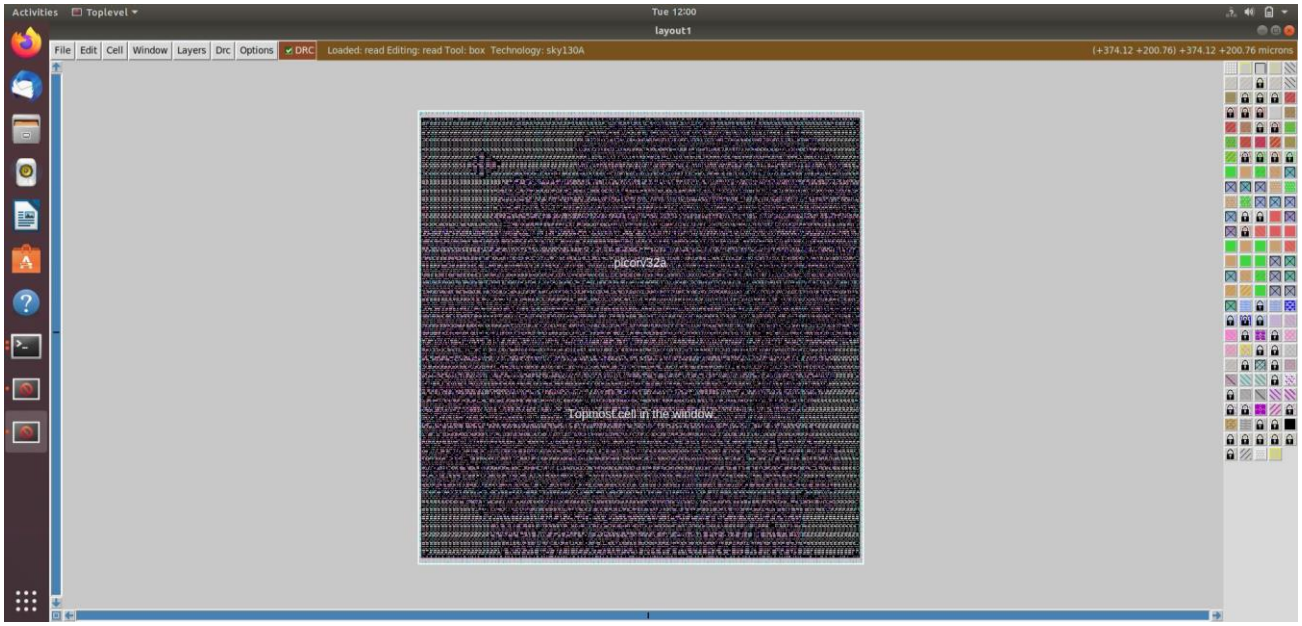


Routing result of picorv32a:

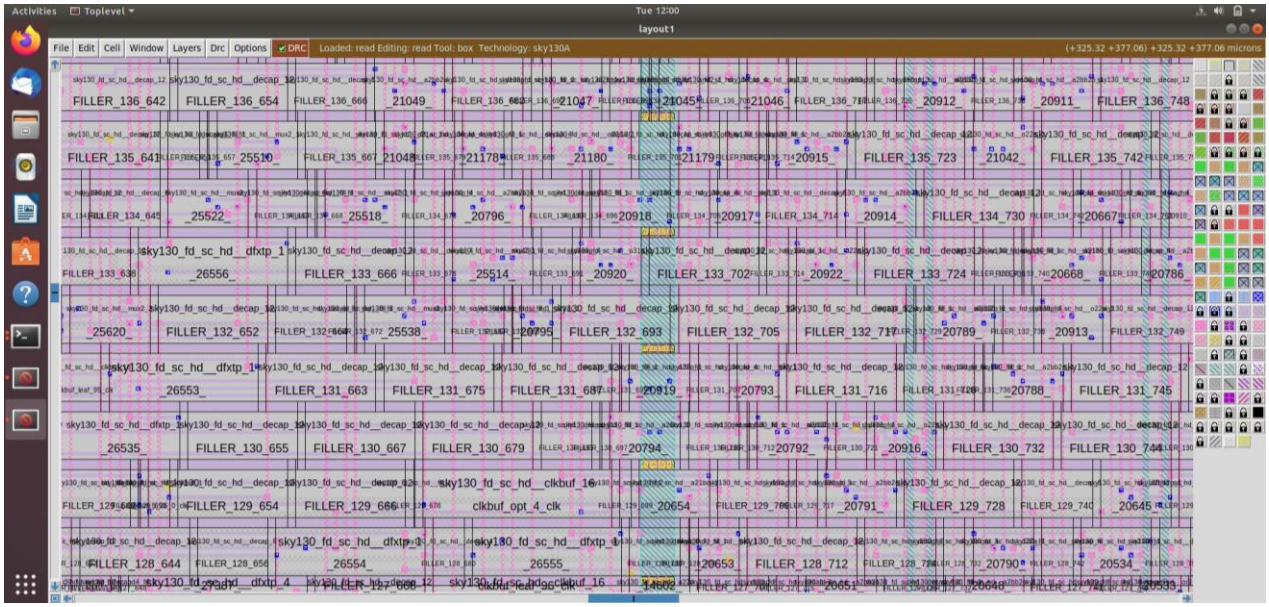
Command : run_routing



Layout view of picorv32a_routing.def,



Detailed view of design,



Acknowledgements

- Kunal Ghosh, Co-founder (VSD Corp. Pvt. Ltd).
- Nickson Jose - Workshop Instructor