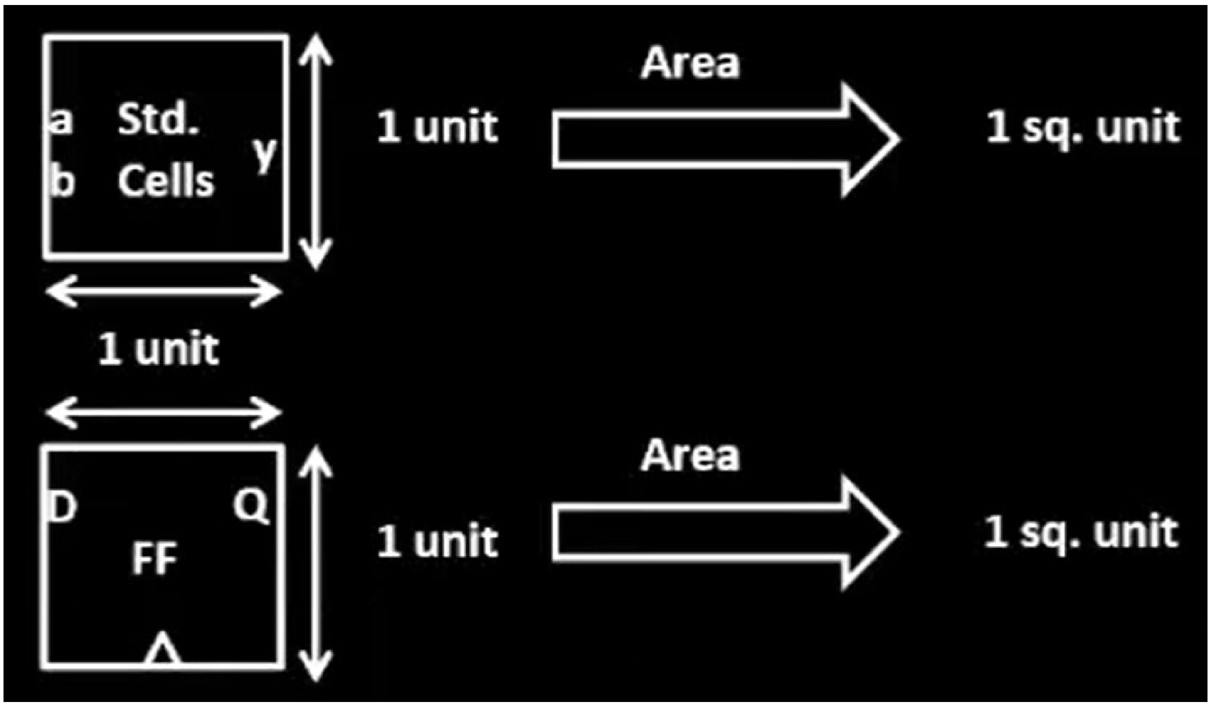
SKY130 Day 2: Good vs Bad Floorplan and Introduction to Library Cells

Chip Floor Planning Considerations

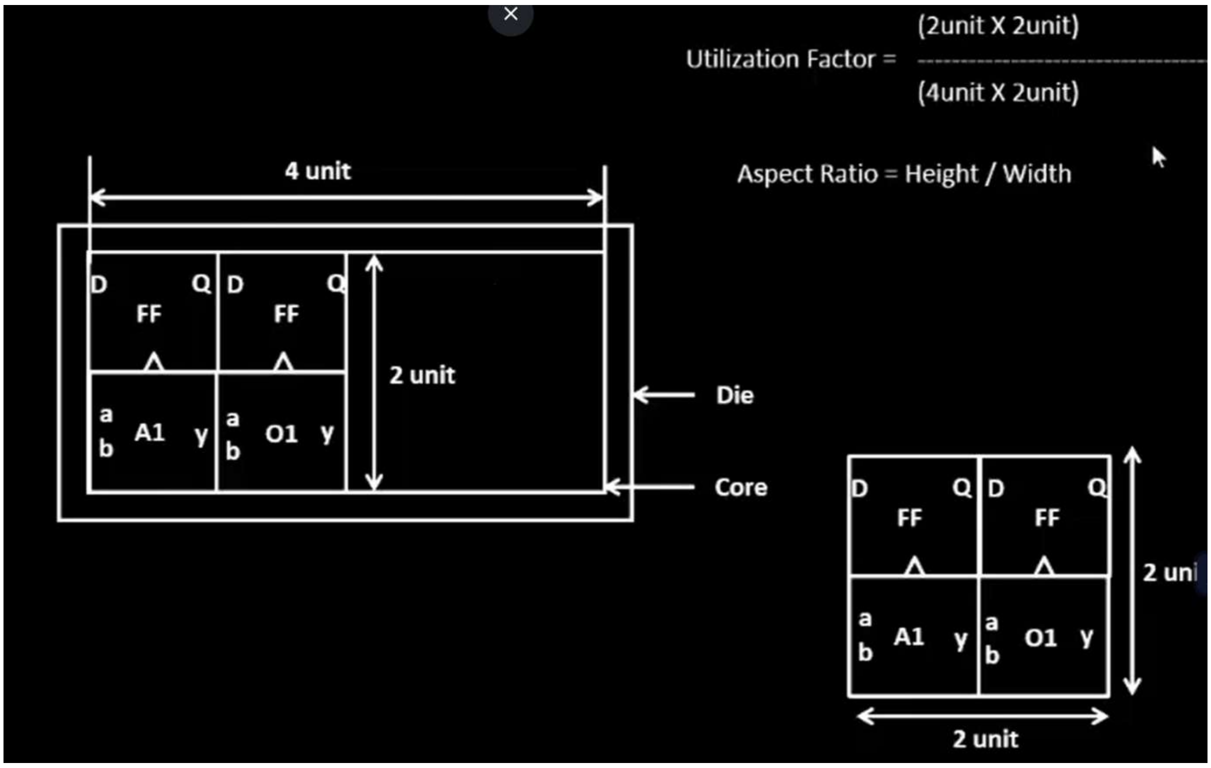
Utilisation Factor and Aspect Ratio

The first step in physical design is defining the core and die size. Starting with a simple netlist, we create a basic symbolic diagram, later converting it into a physical design. Each cell (gates, flip-flops) is assigned rough dimensions—1×1 unit per cell. With 4 gates/flip-flops, the total silicon area required is 4 square units.



NOTE : Here, we are ignoring the wires

All logical cells are placed inside the core, which is part of the die. If the core is fully occupied, it has 100% utilization. Utilization factor = (Netlist area) / (Core area). Here, it is 100%, but in practice, it's around 50%. The aspect ratio is the height-to-width ratio—1 for a square chip, otherwise rectangular.

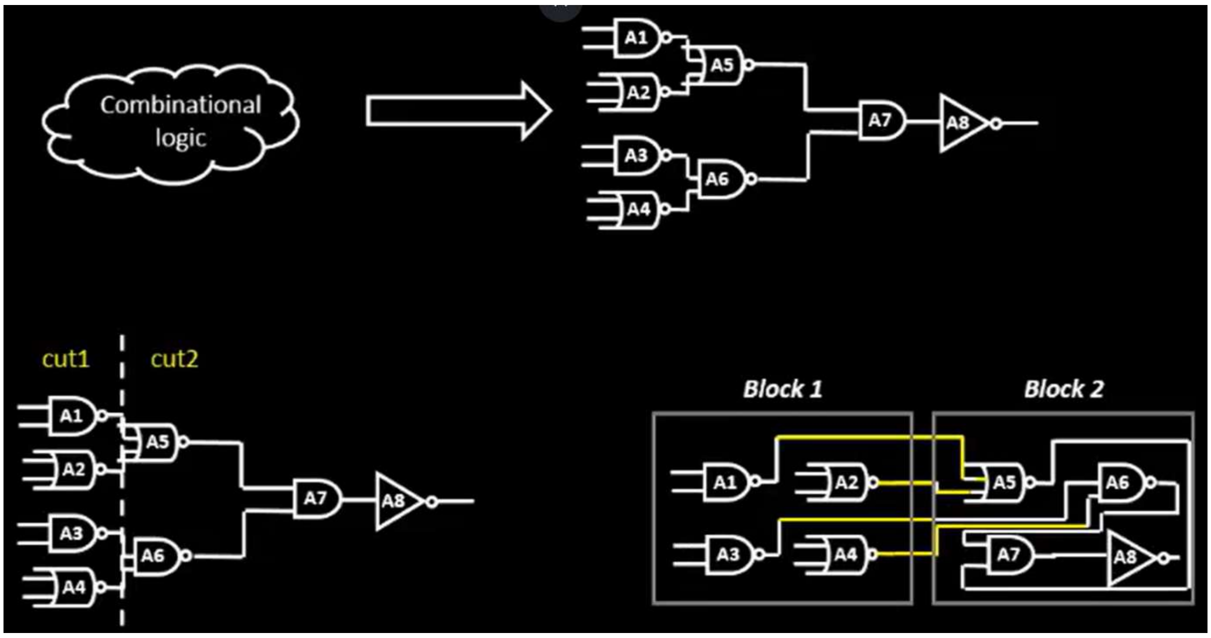

Utilization Factor = (2unit X 2unit) / (4unit X 2unit)
Aspect Ratio = Height / Width
4 unit
2 unit
D
Q
FF
a
b
A1
O1
y
Die
Core
2 unit

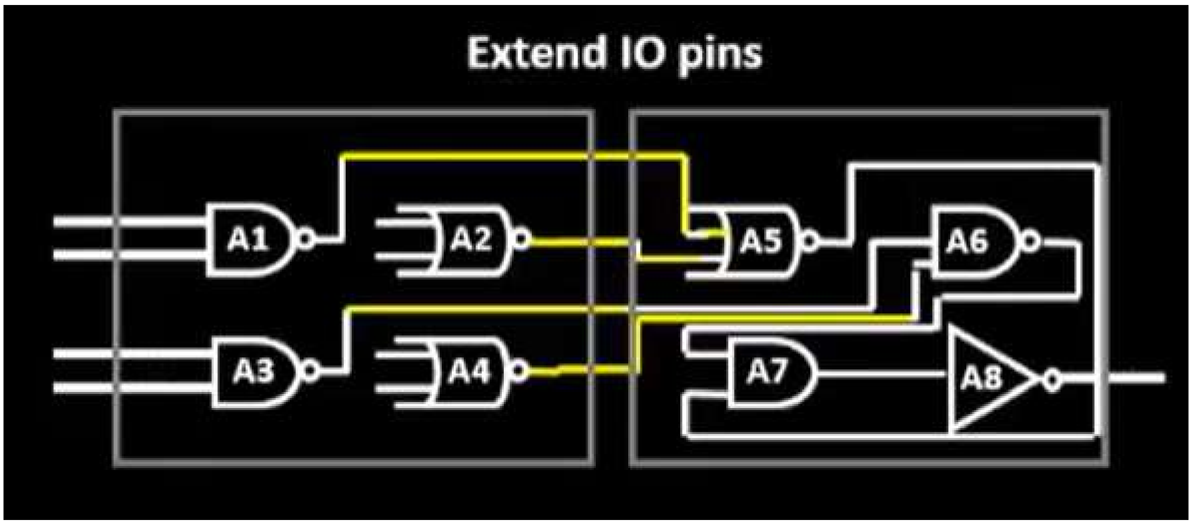
Utilisation factor = 50 %

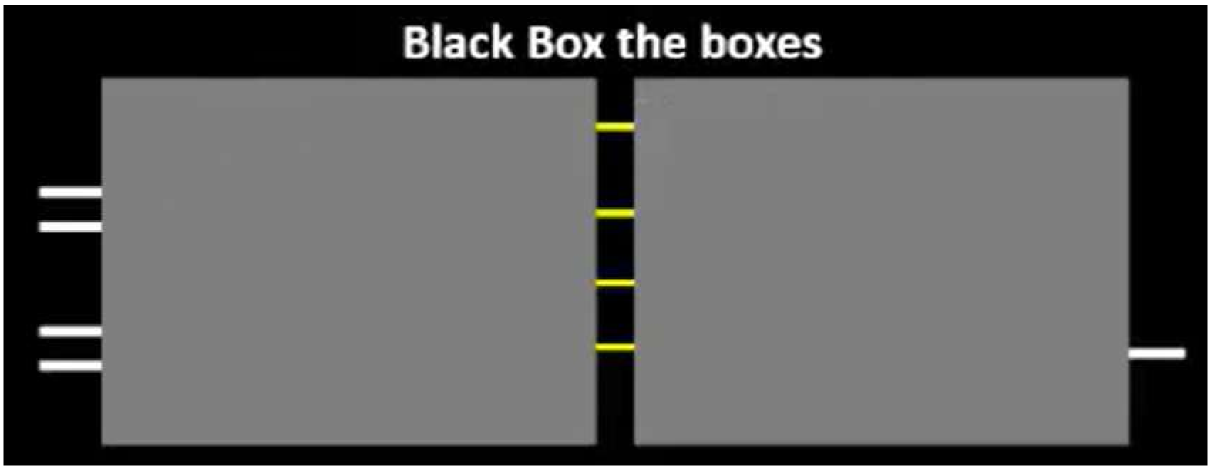
Aspect Ratio = 2 : 4 = 1 : 2 = .5

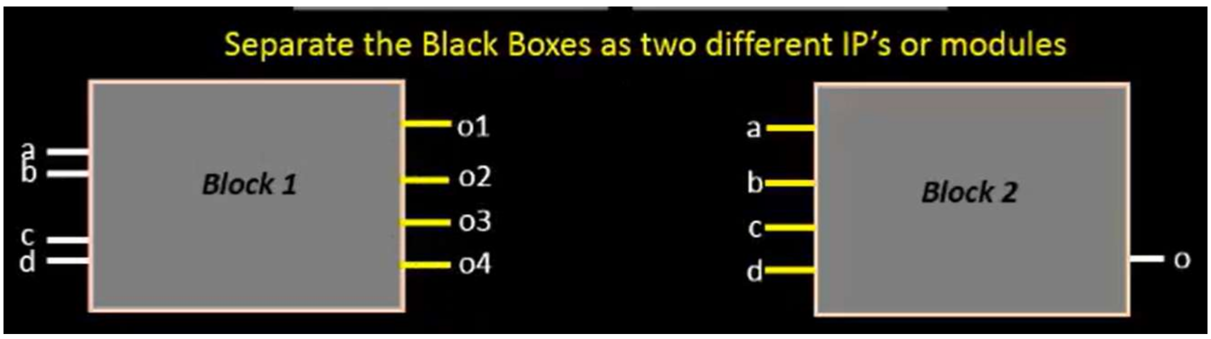
Concept of Pre-Placed Cells

Pre-placed cells are reusable complex logic blocks that Auto Place & Route tools cannot modify, requiring careful design. Placement is user-controlled. A netlist represents combinational logic, which can be divided into black-boxed blocks while maintaining connectivity. If only a black box is needed, it can be directly handed to the designer. Common pre-placed blocks include memory, clock-gating cells, comparators, and MUX. Arranging these blocks in a chip is called floorplanning.









De-Coupling Capacitors

We surround pre-placed cells with de-coupling capacitors. If we think of a circuit to be part of a Pre-placed cells are surrounded by decoupling capacitors to maintain stability. When a circuit is switched on, current is drawn from Vdd, and when switched off, it discharges through the ground. Due to resistance, inductance, and capacitance in the wires, the supplied voltage (Vdd) slightly reduces to Vdd'. Vdd' must stay within the noise margin (Vih to Voh) to ensure circuit stability. If it falls outside this range, the circuit becomes unstable, mainly because of the physical distance between the voltage supply and the circuit.

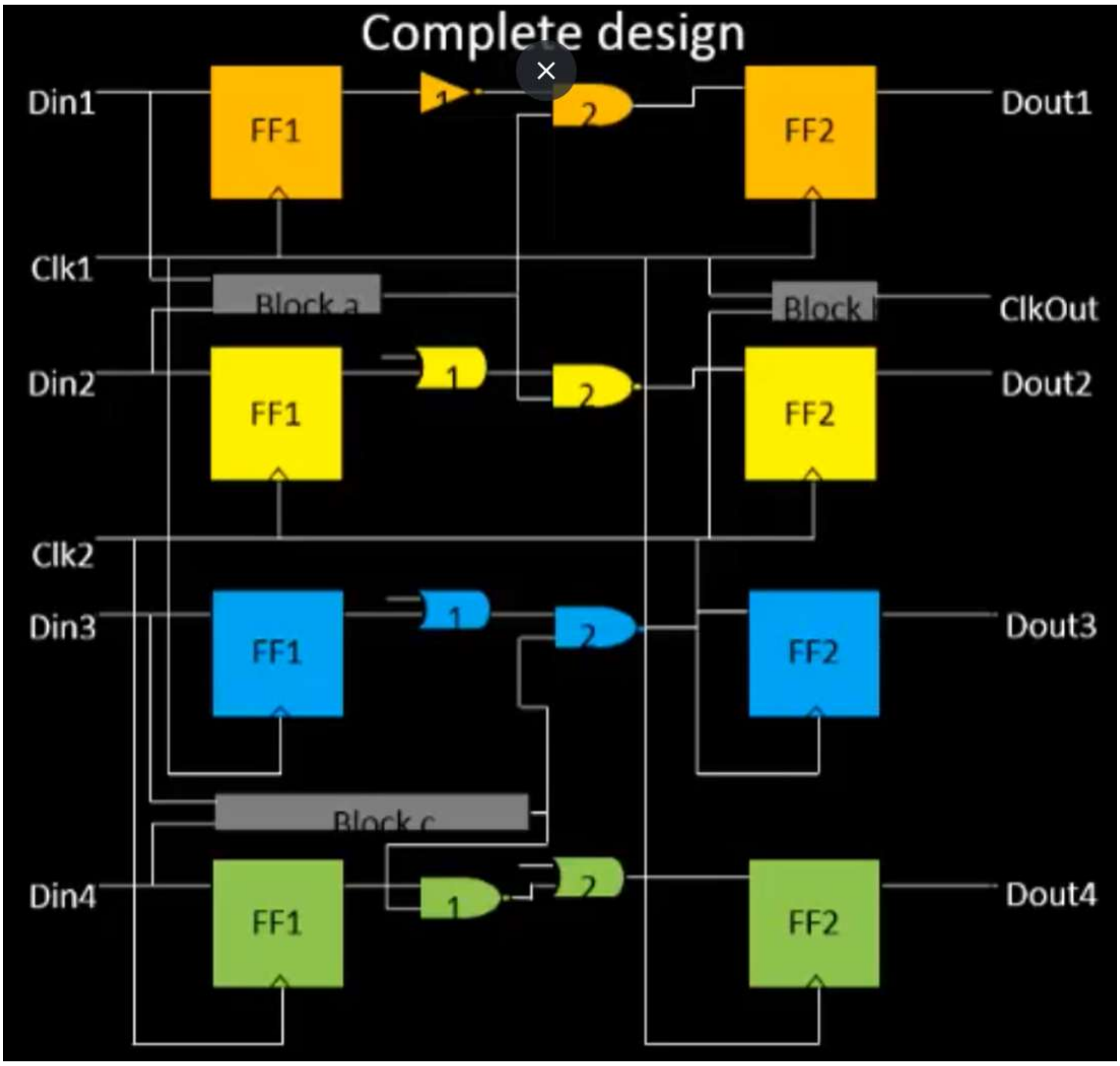
Decoupling capacitors solve the issue of voltage drops by acting as large capacitors filled with charge. The voltage across the capacitor matches the main supply voltage. These capacitors decouple the circuit from the main power supply, ensuring that pre-placed cells receive stable power from the capacitors, maintaining stability and reliability in the circuit.

Power Planning

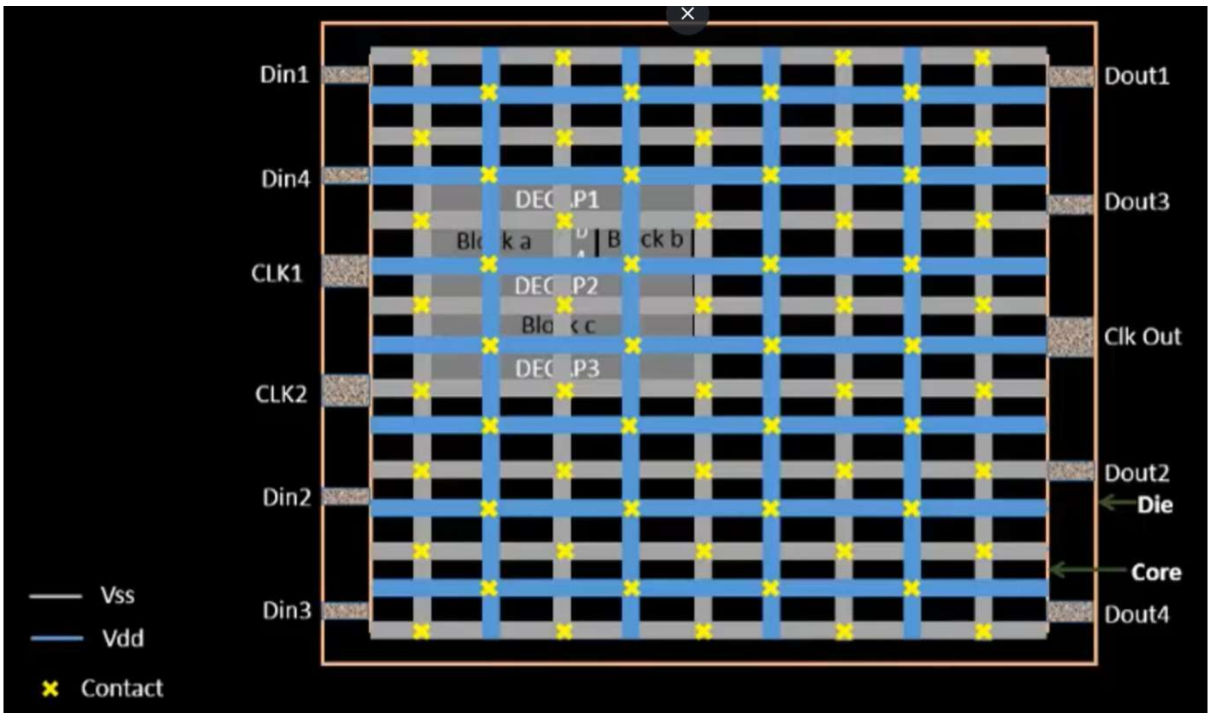
When a logic block, or macro, is repeated multiple times on a chip and requires a lot of voltage, it needs to be powered through decoupling capacitors. However, it's not practical to add decoupling capacitors to the entire circuit. Instead, only the critical elements of the design are decoupled to ensure stable power supply and avoid voltage instability.

When a 16-bit bus is connected to an inverter, all capacitors may discharge simultaneously, causing Ground Bounce due to the large voltage demand. This can also lead to Voltage Drop if there isn't enough current. Both issues can result in the voltage falling outside the noise margin, causing instability. To resolve this, a power mesh is used. This involves multiple power source taps and ground sources, allowing capacitors to source current from the nearest Vdd and sink current to the nearest ground, ensuring stable voltage throughout the circuit.

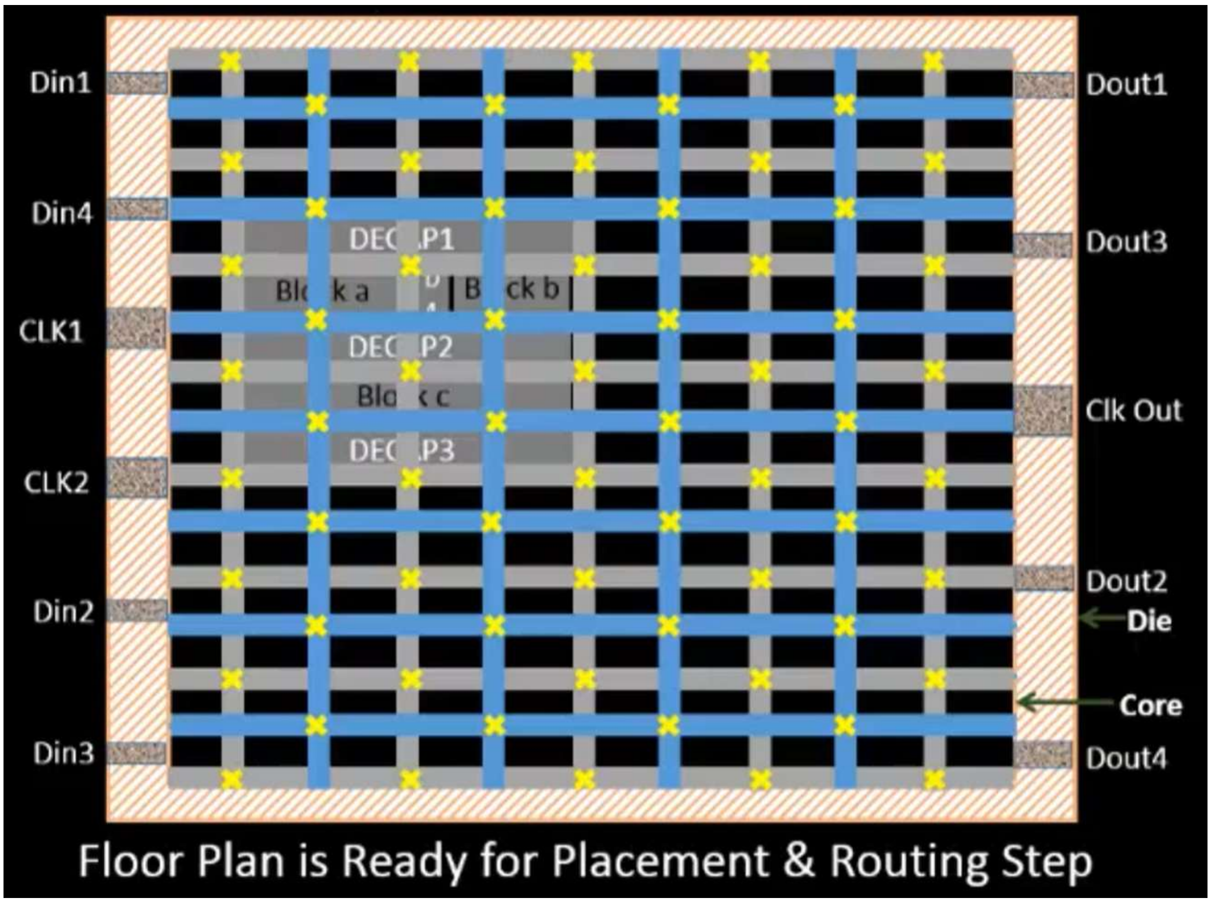
Pin Placement and Logical Cell Placement Blockage



In the example netlist, the connections are defined using VERILOG (or VHDL). The input and output ports are positioned on the left and right sides, respectively, between the core and the die. These port placements are specific to each cell. Since clocks continuously drive the cells, clock ports are larger than data ports. To ensure efficient clock signal delivery, it's crucial to minimize resistance along the clock paths. The size of the clock paths is inversely proportional to the resistance—larger paths reduce resistance and improve performance.

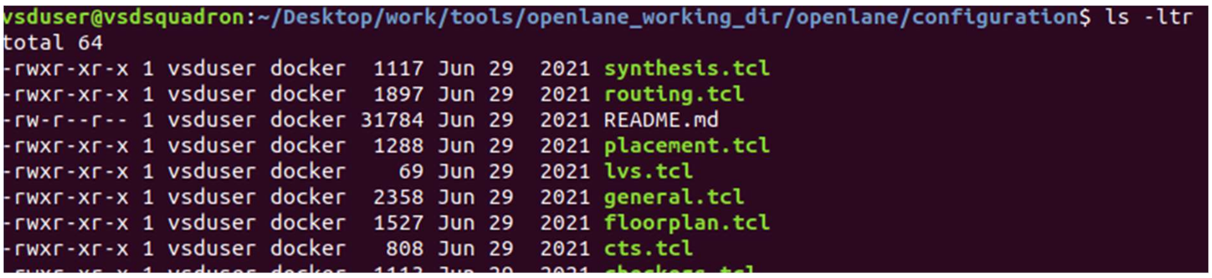


After placing the pins, we create Logical Cell Placement Blockages to prevent the Auto Place and Route (APR) tool from placing any cells on the pin locations. This ensures that the pins remain unobstructed and accessible, allowing proper connections without interference from other cells.



Steps to Run Floorplan Using OpenLANE

1. The first step is setting the configuration variables. Before running the floorplan, the configuration variables or switches must be defined. These can be found in the openlane/configuration file, where various settings related to the design and layout are specified.



```
vsduser@vsdsquadron:~/Desktop/work/tools/openlane_working_dir/openlane/configuration$ ls -ltr
total 64
-rwxr-xr-x 1 vsduser docker  1117 Jun 29  2021 synthesis.tcl
-rwxr-xr-x 1 vsduser docker  1897 Jun 29  2021 routing.tcl
-rw-r--r-- 1 vsduser docker 31784 Jun 29  2021 README.md
-rwxr-xr-x 1 vsduser docker  1288 Jun 29  2021 placement.tcl
-rwxr-xr-x 1 vsduser docker    69 Jun 29  2021 lvs.tcl
-rwxr-xr-x 1 vsduser docker  2358 Jun 29  2021 general.tcl
-rwxr-xr-x 1 vsduser docker  1527 Jun 29  2021 floorplan.tcl
-rwxr-xr-x 1 vsduser docker   808 Jun 29  2021 cts.tcl
```

The **`README.md`** file contains all the configuration variables, organized by stage. The `.tcl` files include the default OpenLANE settings, which are applied unless modified by the user. These files help configure the design flow and specify settings for each stage of the process.

2. All configurations and switches for the current run are taken from the openlane/designs/[design-date]/config.tcl file. This file contains the specific settings for the design, and it overrides the default settings if necessary, allowing customization for the design flow.

There is a order of priority -:

- **openlane/designs/[design-date]/sky130A_sky130_fd_sc_hd_config.tcl**
- **openlane/designs/[design]/config.tcl**
- **openlane/configuration/floorplan.tcl**

In OpenLANE, the number of vertical and horizontal metal layers is always set to one more than what is specified. For instance, if you specify 3 vertical metal layers, the system will actually use 4 layers. This ensures that there is an extra layer for routing and design flexibility.

3. Floorplan is to be run on OpenLANE through the command :- *run_floorplan*

Review Floorplan Files and Steps to Review Floorplan

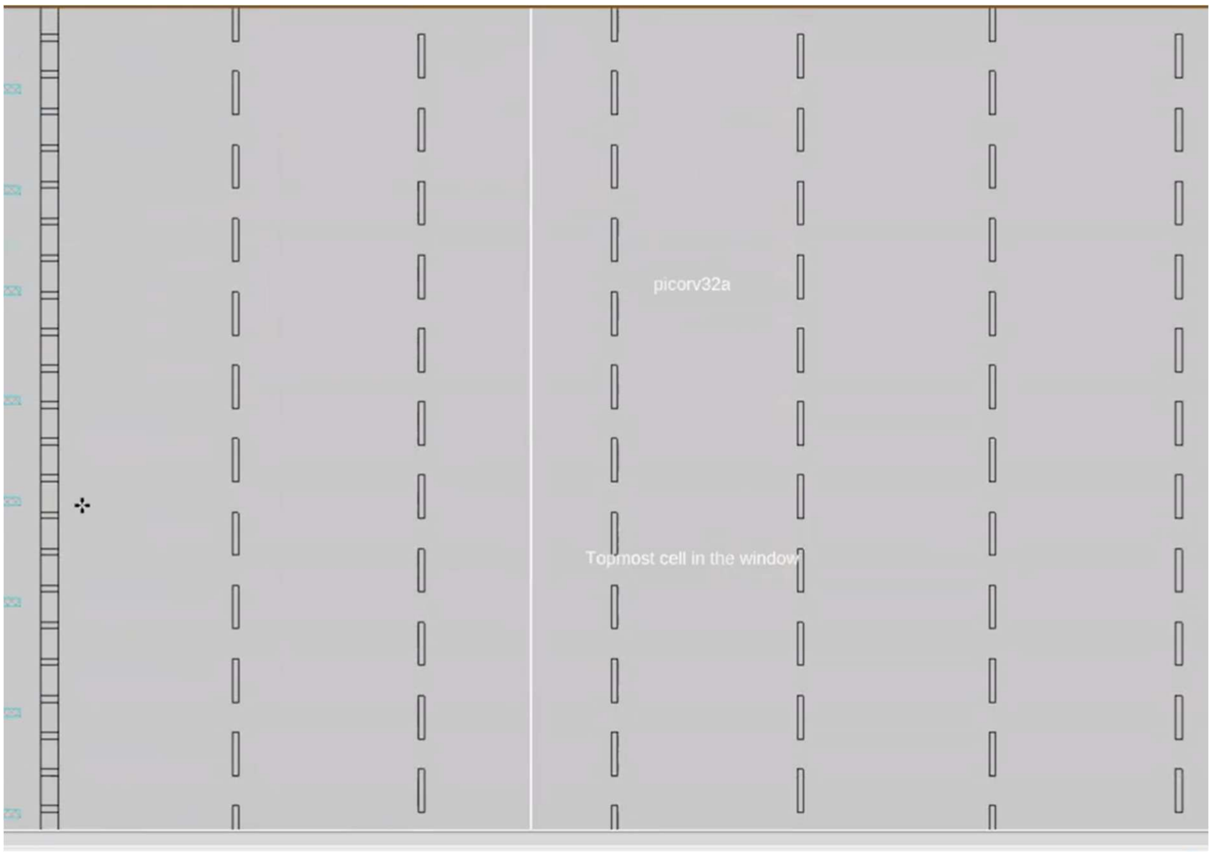
After running the floorplan, the results are stored in a Design Exchange Format (DEF) file. This file includes the die area and positions. The die area in the DEF file is given in database units, where 1 micron equals 1000 database units. For example, if the die dimensions are 554570 units by 565290 units, the die area in microns is calculated as:

Die Area = (554570 / 1000) × (565290 / 1000) = 311829.1653 sq.µm

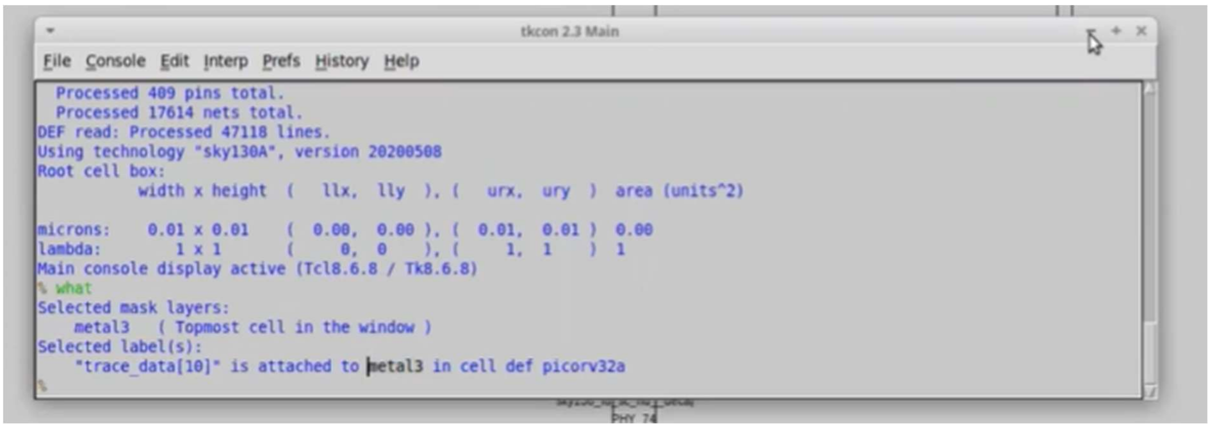
Floorplan Layout in Magic

5. The command *magic -T* **/home/vsduser/Desktop/work/tools/openlane_working_dir/pdks/sky130A/libs.tech/magic/sky130A.tech lef read ../../tmp/merged.lef def read picorv32a.floorplan.def &** should be typed to view the file.

6. Sure! Here's a shorter version:
7. Press the **S** key to select the entire die, then **V** to center the view, and **Z** to zoom in. You'll see that the IO pins are placed randomly and equidistant from each other, based on the **FP_IO_MODE = 1** setting in **openlane/configuration/floorplan.tcl.**



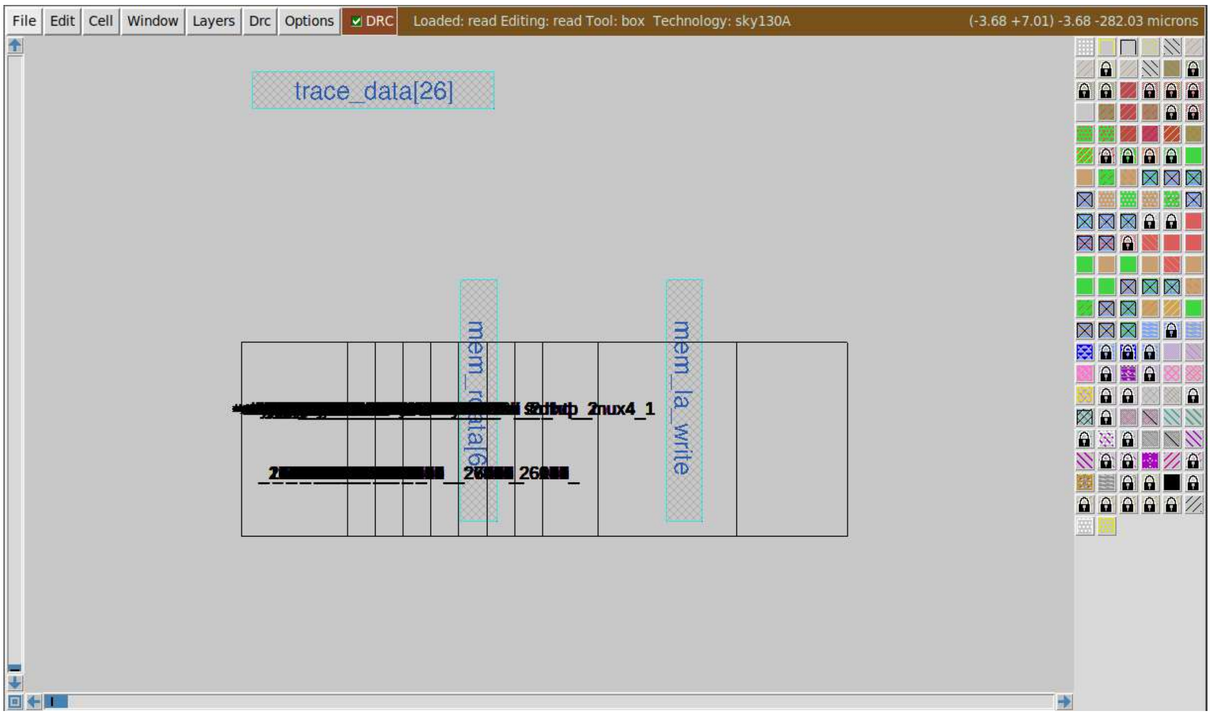
7. After this, typing ***what*** on the **`tkcon`** window will give the layer of the selection.



```
  Processed 409 pins total.
  Processed 17614 nets total.
DEF read: Processed 47118 lines.
Using technology "sky130A", version 20200508
Root cell box:
           width x height  (   llx,  lly  ), (   urx,  ury  )  area (units^2)

microns:    0.01 x 0.01    (  0.00,  0.00 ), (  0.01,  0.01 )  0.00
lambda:        1 x 1       (     0,  0    ), (     1,  1    )  1
Main console display active (Tcl8.6.8 / Tk8.6.8)
% what
Selected mask layers:
    metal3   ( Topmost cell in the window )
Selected label(s):
    "trace_data[10]" is attached to metal3 in cell def picorv32a
%
```

Standard cells are not placed but can be viewed at the bottom left corner of the layout

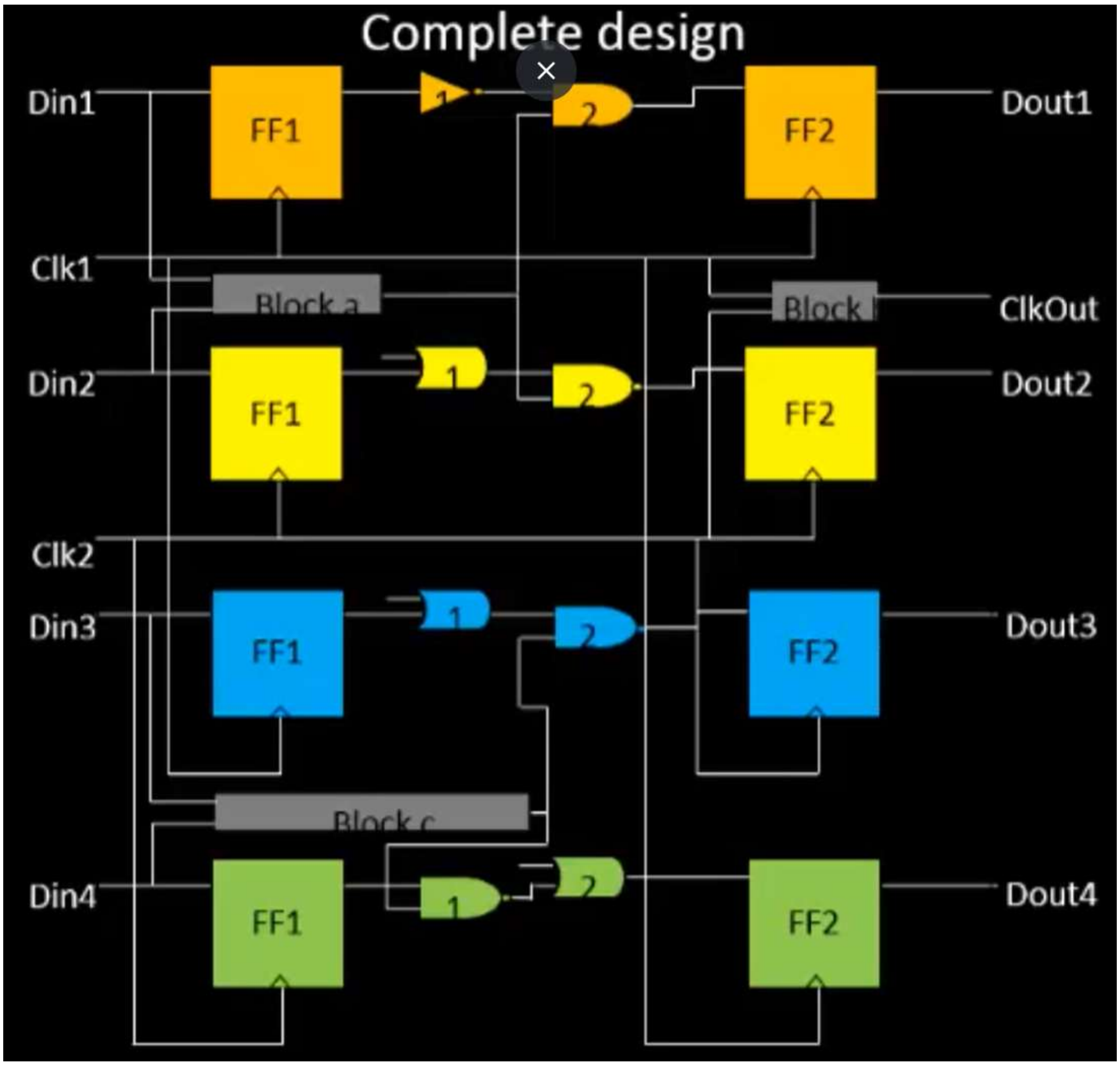

File Edit Cell Window Layers Drc Options DRC Loaded: read Editing: read Tool: box Technology: sky130A (-3.68 +7.01) -3.68 -282.03 microns
trace_data[26]
mem_rdata[6
mem_la_write
2mux4_1

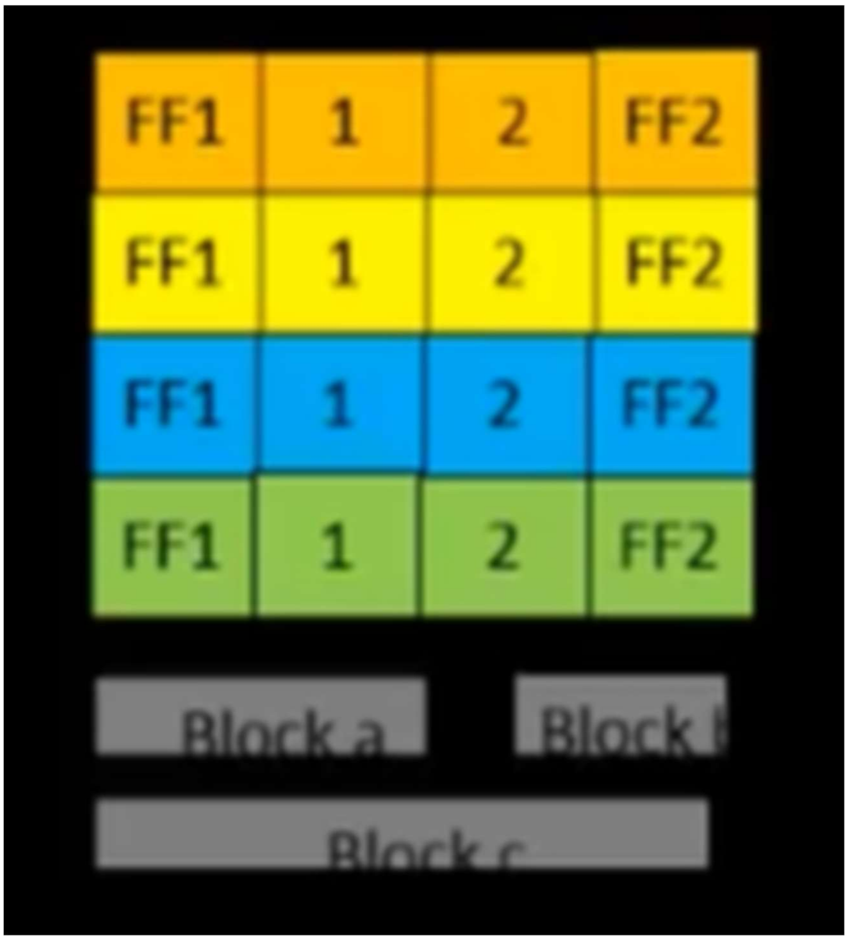
Library Binding and Placement

Netlist Binding and Initial Place Design

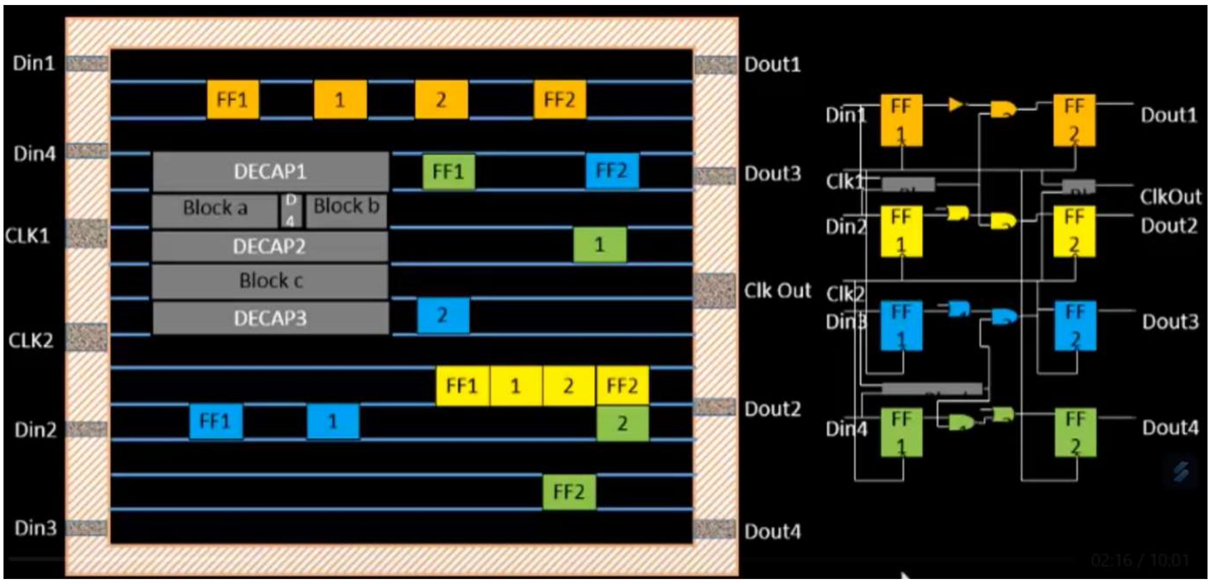
- The first step is to bind the netlist with physical cells, which are cells with real dimensions. While the netlist contains various gates that may appear as simple shapes in the schematic, in production, these gates are typically square or rectangular. Each gate is assigned a specific shape, and once placed, the physical design will look very different from the original netlist, as it reflects the actual dimensions of the cells used in the layout.

These blocks are sourced from a "shelf" called a library. The library contains cells with various shapes, dimensions, and also includes delay information. It offers different sizes of cells with the same functionality, as larger cells typically have lower resistance, improving performance. The library allows designers to select the most suitable cells based on size, speed, and power requirements.


Complete design
Din1
FF1
1
2
FF2
Dout1
Clk1
Block a
Block b
ClkOut
Din2
FF1
1
2
FF2
Dout2
Clk2
Din3
FF1
1
2
FF2
Dout3
Block c
Din4
FF1
1
2
FF2
Dout4


FF1
1
2
FF2
FF1
1
2
FF2
FF1
1
2
FF2
FF1
1
2
FF2
Block a
Block b
Block c

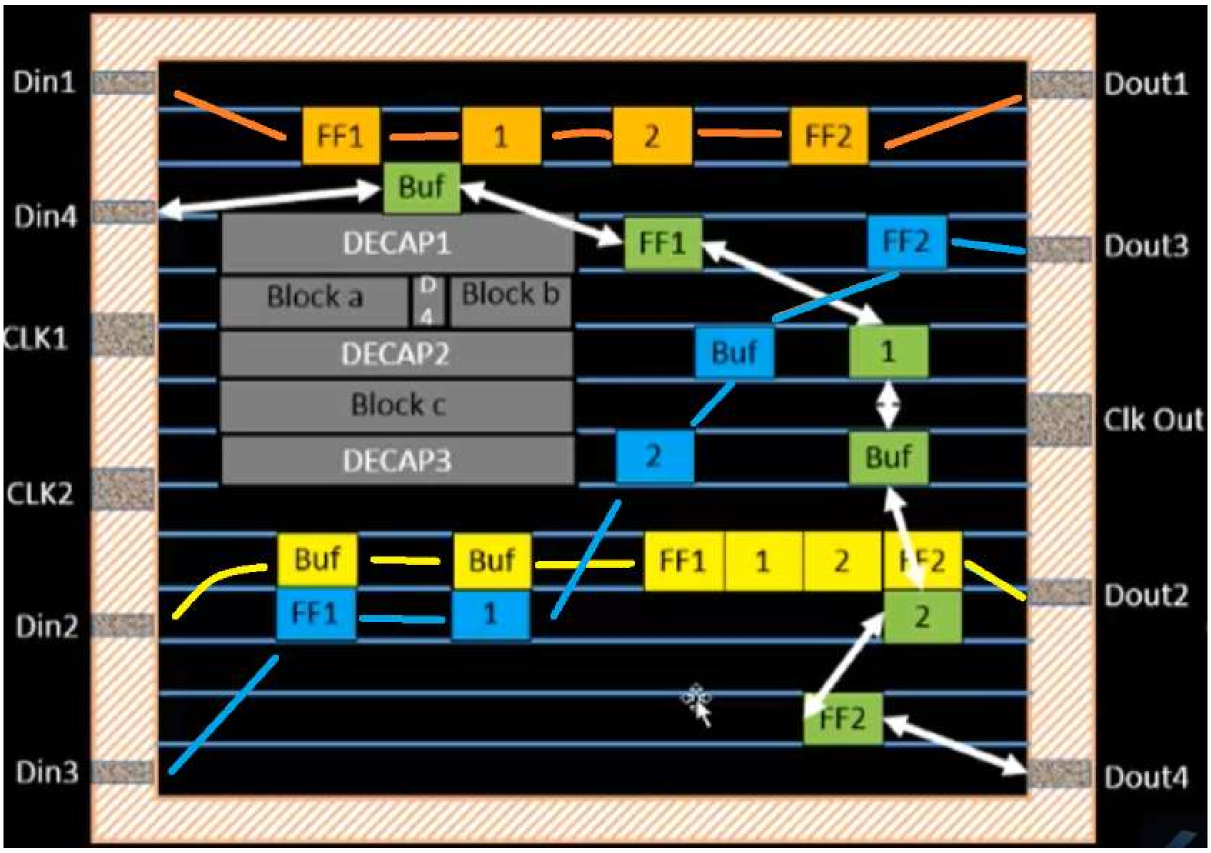
- The second step is **placement**, which is done based on connectivity. For example, Flip-Flop 1 (FF1) is placed near the Din1 pin, and Flip-Flop 2 (FF2) is placed near the Dout1 pin. Combinational cells are then placed close to FF1 and FF2 to minimize delay and improve signal timing, ensuring efficient data flow between components. The goal is to reduce wire lengths and interconnect delays for optimal performance.



Optimise Placement Using Estimated Wire-Length and Capacitance

In this step, we estimate the wirelength needed to connect the components. If the wirelength is too long, repeaters may be required. Long wires can cause signal degradation over distance, so repeaters are used to recondition the signal, restoring it to its original strength before it travels further. This helps maintain signal integrity and ensures reliable data transmission across the chip.

Final Placement Optimization



Congestion Aware Placement Using RePLACE

The command to run placement in OpenLANE, run_placement, is a wrapper that performs three main functions:

1. **Global Placement (using RePlace tool):** This step places cells roughly based on connectivity, without legalizing the placement. The objective here is to minimize the Half-Perimeter Wire Length (HPWL), which is a measure of the routing complexity.
2. **Optimization (using Resier tool):** After global placement, optimization is performed to further improve the placement in terms of wirelength, timing, and power by adjusting the positions of the cells.
3. **Detailed Placement (using OpenDP tool):** This step legalizes the placement, ensuring that standard cells are placed in rows without overlap, adhering to design rules and spacing constraints.

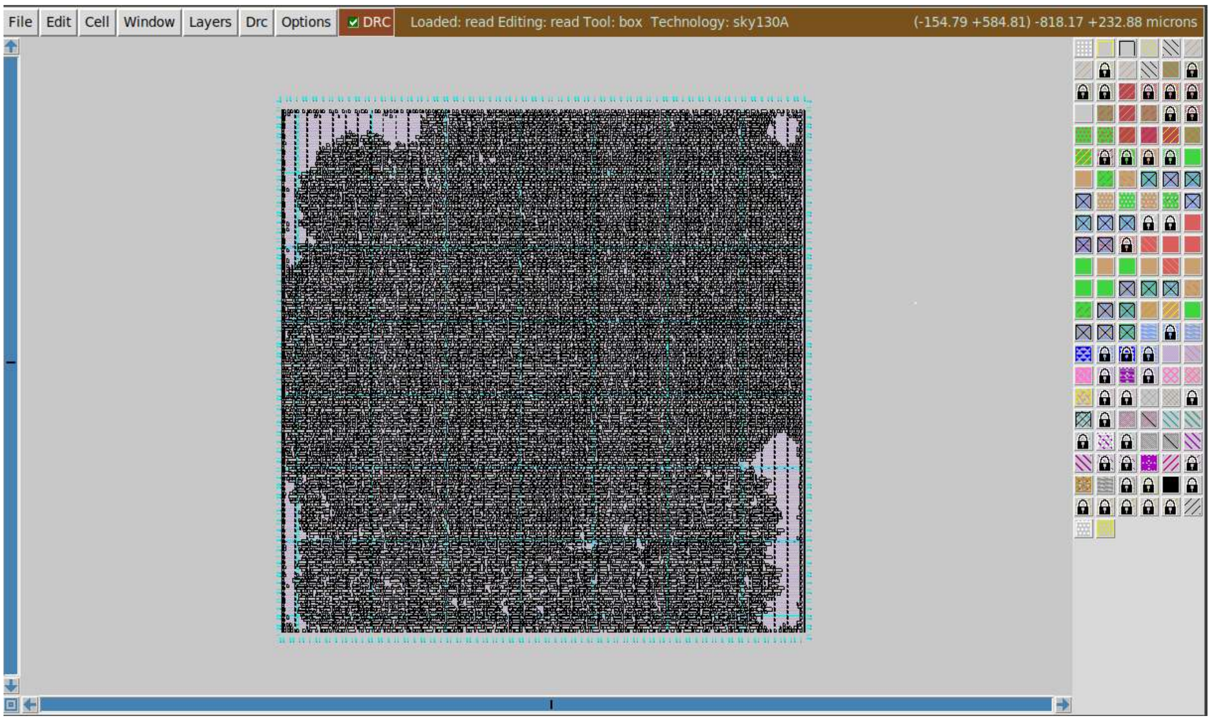
   Placement aims to **converge the overflow value**.

Together, these tools ensure efficient and valid cell placement while optimizing the overall design for performance and manufacturability.

NOTE: If placement will be sucessful and the designs will converge, the overflow value will progressively reduce during the placement.

After running the placement, output is generated in this folder ***openlane/designs/picorv32a/runs/[design - date]/results/placement/picorv32a.placement.def***

Then, we can type the command : ***magic -T /home/vsduser/Desktop/work/tools/openlane_working_dir/pdks/sky130A/libs.tech/magic/sky130A.tech lef read ../../tmp/merged.lef def read picorv32a.placement.def &*** to view it in Magic:


File
Edit
Cell
Window
Layers
Drc
Options
DRC
Loaded: read Editing: read Tool: box Technology: sky130A
(-154.79 +584.81) -818.17 +232.88 microns

Cell Design and Characterisation Flows

Inputs for Cell Design Flow and Circuit and Layout Design Step

Standard cells, like AND gates, OR gates, buffers, etc., are stored in the standard cell library. This library contains various types of cells, each with different variations in drive strength, functionality, and voltage. Larger cells generally have a higher drive strength, making them better suited for driving longer wires. Additionally, cells with a higher threshold voltage (Vth) will take longer to switch than those with a lower Vth, as higher Vth requires more time to change the state of the transistor. This trade-off between drive strength and switching speed is important when choosing the appropriate cells for the design.

The standard cell design flow is as follows:-

INPUTS (PDKS : DRC and LVS rules, SPICE models, library and user defined specs)

PROCESSES (circuit, layout design and charecterisation)

OUTPUTS (Circuit Description Language, GDSII, lef, timing, noise etc)

DRC & LVS Rules contain tech files and poly substrate parameters

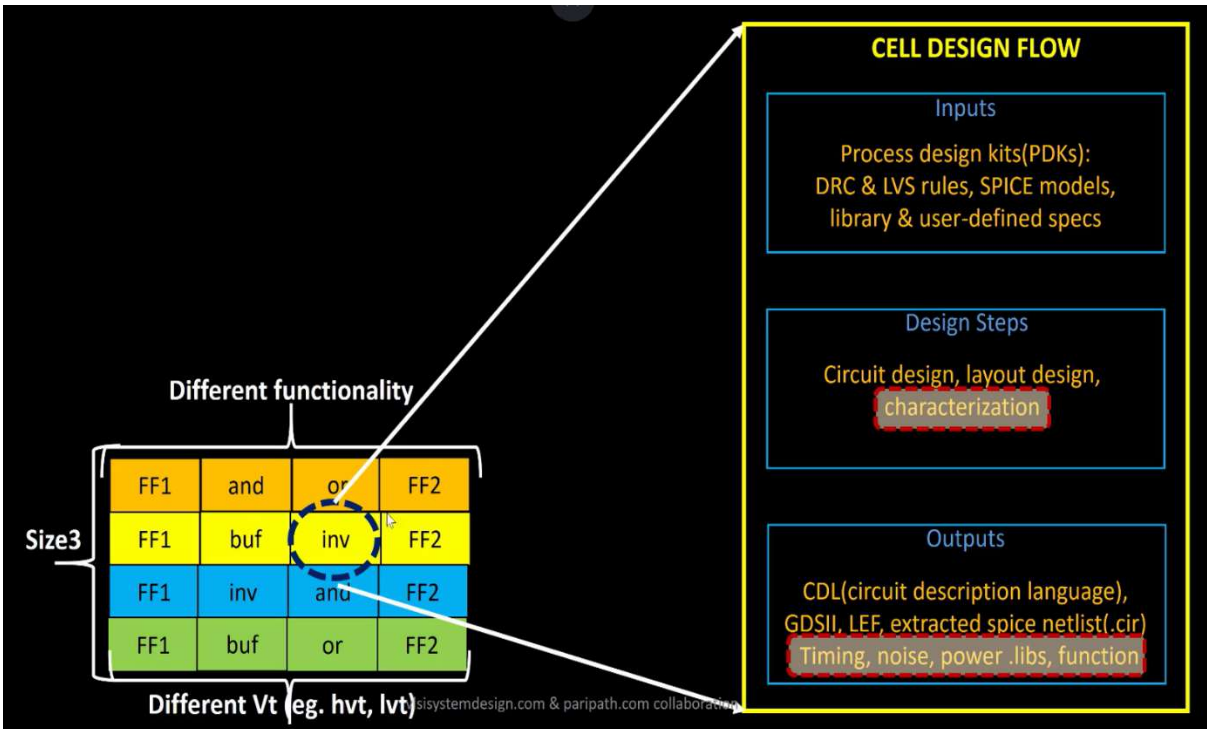
SPICE Models contain threshold, linear regions, saturation region equations with added foundry parameters, including NMOS and PMOS parameters

User defined specifications include cell height and cell width, supply voltage, pin locations, and metal layer requirement

IMPORTANT: The standard cell library developer must adhere to the rules given by the foundry so that when the cell can be used on a real design without any errors

Circuit design is accomplished by modeling the PMOS and NMOS transistors to meet the input library requirements. This ensures that the logic functions properly within the specified parameters of the standard cell library.

For layout design, the **Magic layout tool** is used, which employs **Euler's path** and **stick diagrams**. Euler's path is a method used to ensure that all necessary connections are made without repeating paths, while stick diagrams provide a high-level representation of the layout, showing the connections between different components and layers in a simplified form. Together, these tools help create the physical design of the circuit.


CELL DESIGN FLOW
Inputs
Process design kits(PDKs): DRC & LVS rules, SPICE models, library & user-defined specs
Design Steps
Circuit design, layout design, characterization
Outputs
CDL(circuit description language), GDSII, LEF, extracted spice netlist(.cir)
Timing, noise, power .libs, function
Different functionality
Size3
FF1
and
or
FF2
FF1
buf
inv
FF2
FF1
inv
and
FF2
FF1
buf
or
FF2
Different Vt (eg. hvt, lvt)
vlsisystemdesign.com & paripath.com collaborating

Typical Characterisation Flow

Steps of Characterisation Flow:-

1. Reading of SPICE module files
2. Reading of netlist extracted by SPICE

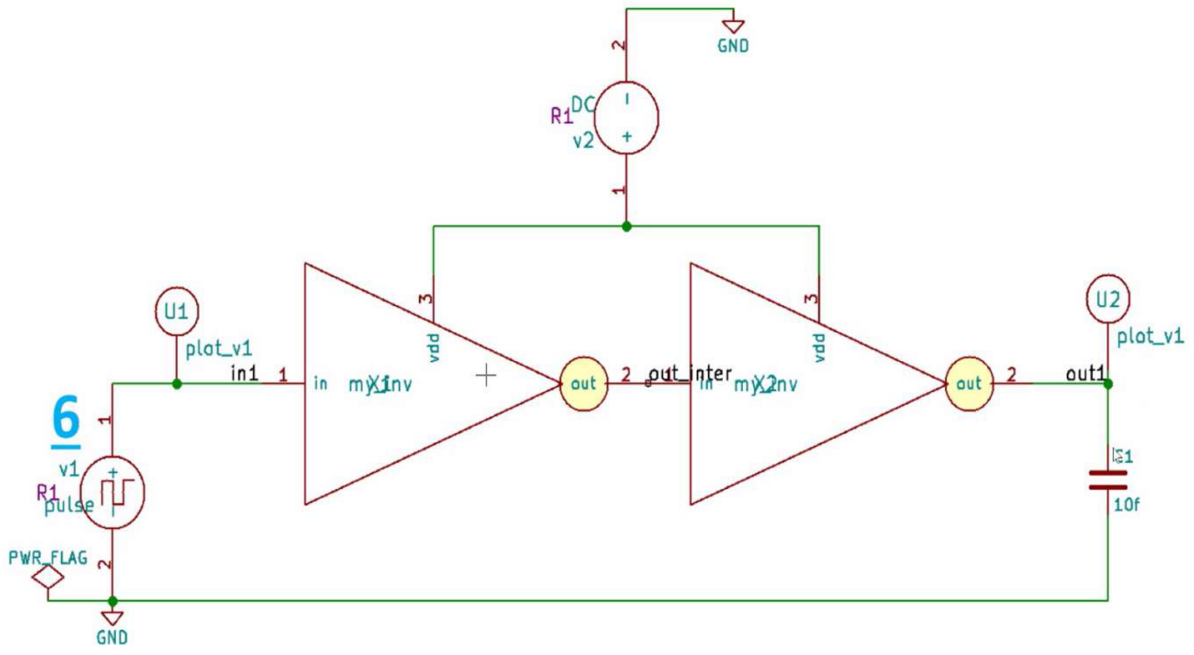
3. Recognising buffer behaviour
4. Reading subcircuits
5. Attaching neccessary power sources
6. Applying stimulus
7. Provision of of neccessary output capacitance
8. Provision of simulation command

These steps are given to the CHARECTERISATION SOFTWARE KNOWN AS **GUNA** in the form of a configuration file, which will generate timing, noise and power models in the form of .*libs* files.

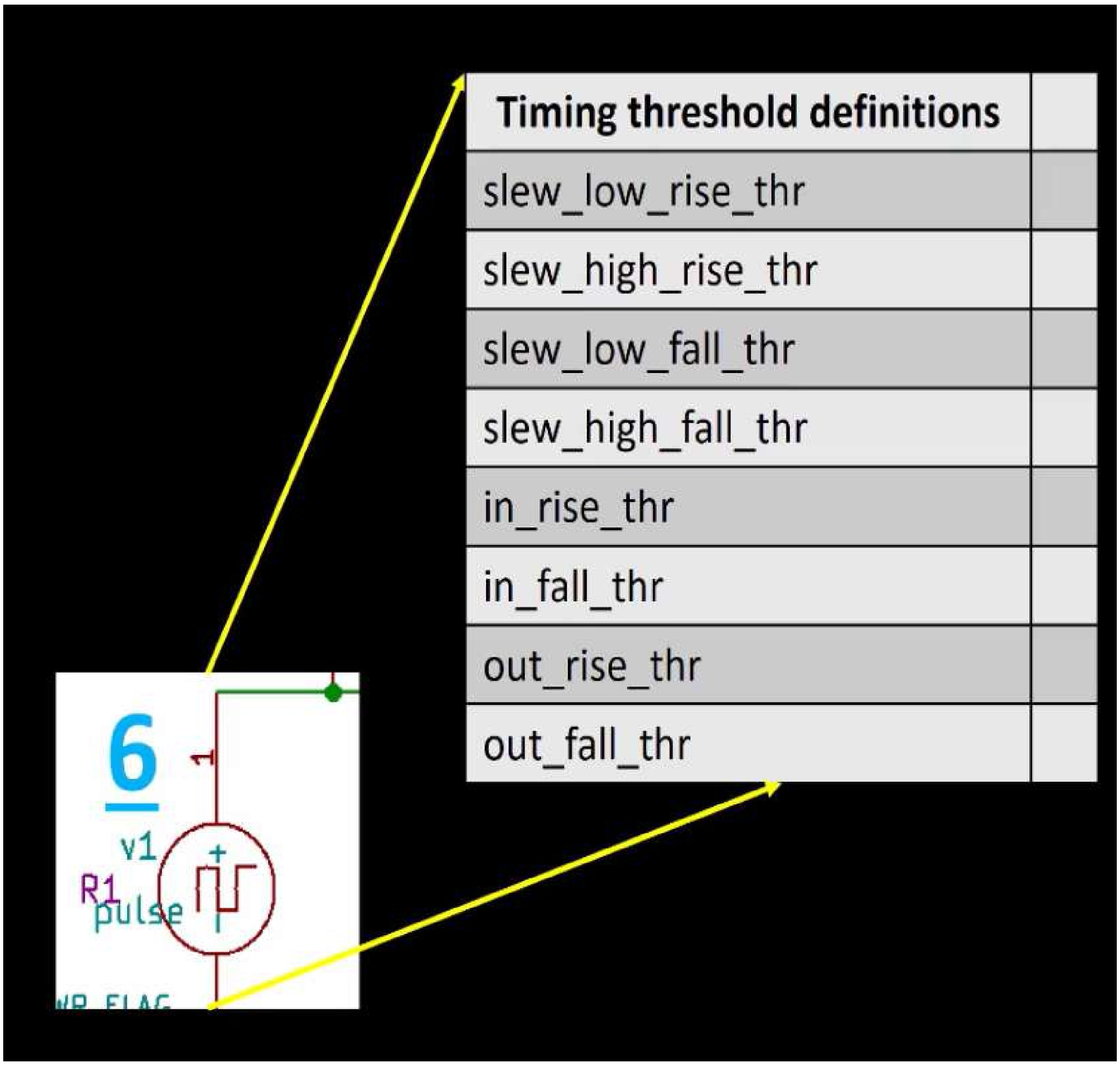
General Timing Characterisation Parameters

Timing Threshhold Definitions

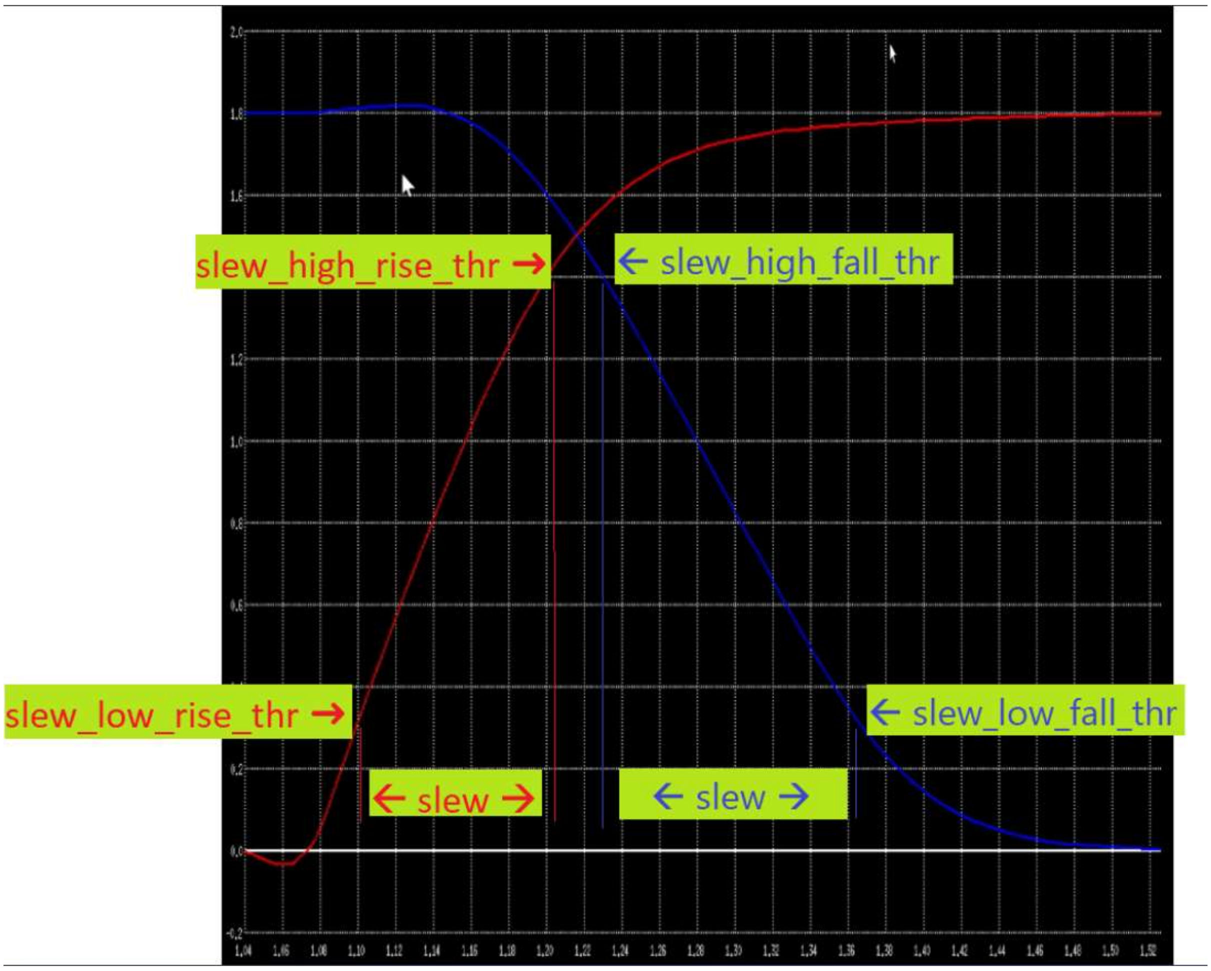
Here, we will talk about the semantics of the various .*libs* files generated by GUNA. To do this, we will take this circuit as an example:

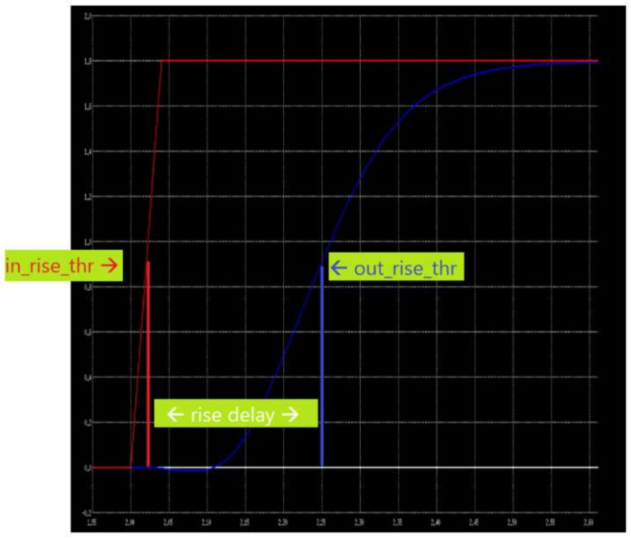


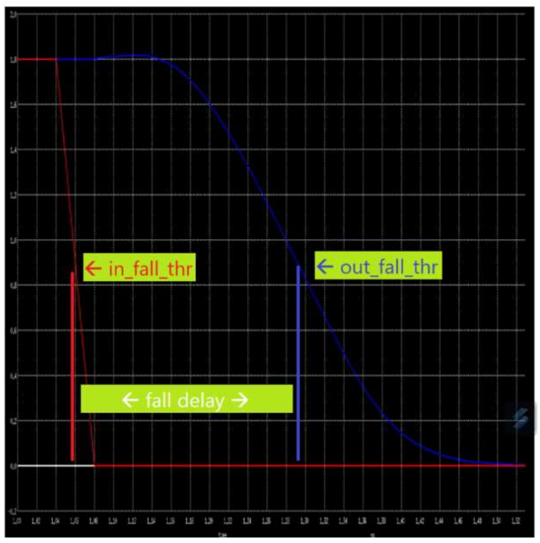
Here, the red line is output of first inverter and blue is output of second inverter.



Timing threshold definitions	
slew_low_rise_thr	
slew_high_rise_thr	
slew_low_fall_thr	
slew_high_fall_thr	
in_rise_thr	
in_fall_thr	
out_rise_thr	
out_fall_thr	



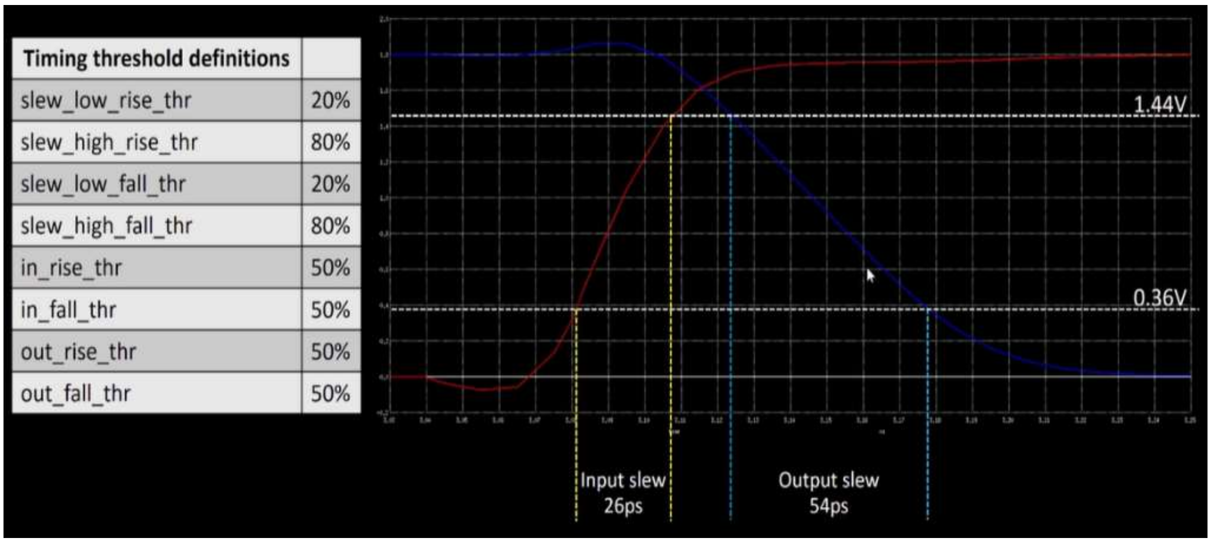




Propogation Delay and Transition Time

Propogation delay is calculated as = time(out_x_thr) - (time_x_thr). If the propogation delay is negative, it can cause quite unexpected results - as an output is generated before the input. Hence, threshhold values should be selected properly. Delay threshold is usually 50% and slew rate threshold is usually 20%-80%.

Transition time is calculated as = time(slew_high_x_thr) - time(slew_low_x_thr)



Timing threshold definitions	
slew_low_rise_thr	20%
slew_high_rise_thr	80%
slew_low_fall_thr	20%
slew_high_fall_thr	80%
in_rise_thr	50%
in_fall_thr	50%
out_rise_thr	50%
out_fall_thr	50%