Presentation Topics Covered

1. Synthesis

2. Inputs of PD & it's Contents

3. SDC Contents

4. Sanity Checks

5. Floorplan & Physical only cells

6. Power plan

7. Placement

8. CTS

9. Routing

10. Congestion Analysis

11. STA Topics (Setup, Hold, Timing DRVs & PVT, OCV & Derates)

12. Sign-Off (Crosstalk, Antenna Effect, Multi-cut Vias, Redundant Via Insertion / Double Via Insertion, IR Drop Analysis, EM, DRC & LVS)

Synthesis

- **Synthesis**

Synthesis is process of converting RTL (Synthesizable Verilog code) to technology specific gate level netlist (includes nets, sequential and combinational cells and their connectivity).

RTL:

RTL stands for Register Transfer Level. Here, we describe the behavioral functionality of the circuit/system that we are designing based on the flow of signals or transfer of data.

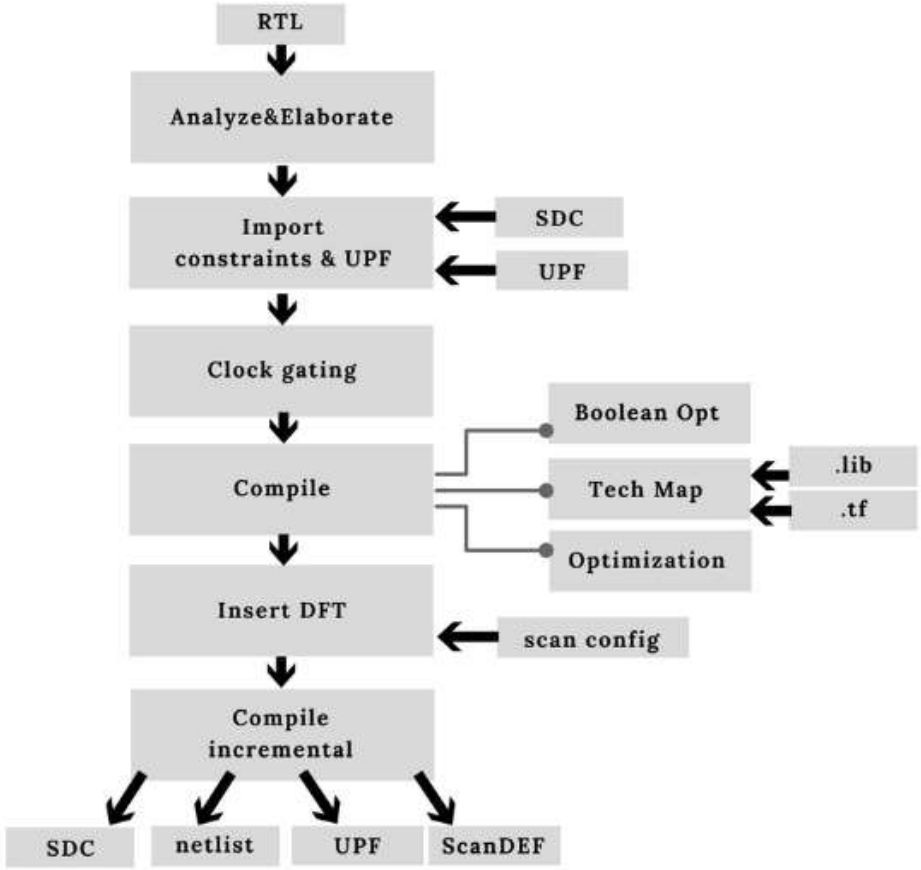
Inputs of Synthesis:

- Netlist (RTL code)
- SDC
- .lib, .lef, Tech file & TLU+ file
- UPF
- Scan config

Goals of Synthesis:

- To get a gate level netlist
- Inserting clock gates
- Logic optimization
- Inserting DFT logic
- Logic equivalence between RTL and netlist should be maintained

Synthesis Steps:



Outputs of Synthesis:

- Netlist (Gate level netlist)
- SDC
- UPF
- Scan DEF - information of scan flops and their connectivity in a scan chain

Inputs of PD & it's Contents

- **Netlist**
- **SDC**
- **.lib (Timing Library)**
- **.lef (Physical Library)**
- **Technology file (.tf)**
- **Tlu+ file**
- **DEF file**
- **Milkyway Library**

• **Netlist**

It is a textual description of a circuit mode of components. Components are generally gates, so a Netlist is a collection of gates. A netlist can also be a connection of resistors, capacitors or transistors. Overall, Netlist is the description of connectivity of an electronic circuit. Synthesis team provides netlist to us.

Eg of a Neltist:

```
Module half-adder (C,S,A,B);
    Input A,B;
    Output C,S;
```

AND2 U1(.Y(C), .A(A), .B(B));

EXOR U2(.Y(S), .I1(A), .I2(B));

Endmodule

- **.lib (Timing Library)**

.lib is basically a timing model contains cell delays, transition, setup and hold time requirements. CCS and NLDM techniques are used to generate .lib files. In CCS (composite current source) current source is used for driver modeling, CCS has 20 variables to account input slew and output load data where as, NLDM uses the voltage source for driver modeling and it has only 2 variables which are not sufficient for modeling the nonlinearity of any circuit. So CCS is more accurate than NLDM. Because of the difference in number of variables used in both the models, size of CCS file is 10X times larger than the NLDM file. Also the run time for CCS is more when compared to NLDM.

The design needs to be tested for certain PVT (process voltage and temperature) corners. But for every PVT corner, the timing of the cells are different. Hence there is a .lib file for every PVT corner.

In .lib file following unit attributes are present

- Time unit
- Voltage unit
- Current unit
- Leakage power unit
- Capacitive load unit
- Slew rate : Lower and upper limit values are defined in terms of percentage for both rise and fall time

- Input threshold at rise and fall time

- Output threshold for rise and fall time

Look Up table templates are defined for different parameters like delay, hold, passive energy, recovery, removal, setup, with different matrix.

For each cell (AND, NAND, Or etc..) following attributes are defined:

- Area of cell

- Leakage power

- Capacitance

- Rise and fall capacitance

- Properties such as capacitance, direction of the pin etc. for each pin (input and output) will be defined.

Example:

```
Cell (AND2_3) {

area : 8.000

pin (o) {

direction : output;

timing () {

related_pin : “A”;
```

rise_propagation () }

rise_transition () }

function : “(A & B)”;

max_cap :

min_cap : }

pin (a) {

dir : input;

cap : ;

}

- **.lef (Physical Library)**

The LEF file is the abstract view of cells. It only gives the idea about PR boundary, pin position and metal layer information of a cell. To get the complete information about the cell, DEF (Design Exchange Format) file is required. In this 3 sections are defined, i.e. technology, site, macros. In the technology part layers, design rules, via definitions and metal capacitance are defined. In the site, site extension is defined and in the macros the information about cell description, dimension, layout of pins and blockages and capacitance are defined.

For every technology the layer and the via statements are different. So for the layer and via, the type of the layer (layer may be routing type, master slice or overlap), width/pitch and spacing, direction, resistance, capacitance, and antenna factor are defined.

It contains name of the pin, pin location, pin layer, direction of the pin, site row, VDD & VSS, height & width of the pin and cell.

Example:

```
Layer m1
  Type routing
      width 0.50;
End m1
Layer via
  Type cut
End via
Macro NAND_1
Foreign NAND_1 0.00.00
Origin 0.00.00
Size 4.5 by 12.0
Symmetry x y;
Site core;
Pin A
  Dir input;
  Port
  Layer m1        End
```

- **Technology file (.tf)**

The technology file contains the physical properties of our fabrication process. For Eg, it would contains the no of metal layers, the design rules, resistances, capacitance as well as the routing grid needed. Tech file should be compatible with both physical & timing libraries.

This file contains specific information such as names, characteristics for each metal layer and design rules. First input for PD is Tech file.

It contains nwell, pwell, metal pitch, width and spacing.

- **TLU+ file**

TLU+ file is a binary table format that stores the RC Coefficients. The TLU+ models enable accurate RC extraction results by including the effect of width, space, density and temperature on the resistance coefficients. The map file matches the layer and via names in the Milkyway technology file with the names in the ITF (Interconnect Technology Format) file.

The main functions of this file can be given as finding:

- R,C parasitics of metal per unit length.
- These parasitics are used for calculation Net delays.
- If TLU+ files are not given, then these are extracted from .ITF file.
- For loading TLU+ files, we have to load three files Max TLU+, Min TLU+ and Map file.
- Map file maps the .ITF file & .tf file of the layer and via names.

- **DEF file**

The DEF file basically contains the placement information of macros, standard cells, I/O pins and other physical entities. DEF is used as an input for various stages. Floorplan DEF is given at the import design stage to provide information about macro location, IO ports and block shape, SCANDEF is given at the import design stage for scan chain reordering which contains the connectivity information of scan flops and it is also an input of scan tracing stage, DEF generated by PnR is used in Star RC extraction.

In detail it contains:

- Die Area
- Tracks
- Components (macros)
- I/O Pins
- Nets
- Blockages & Halo
- Vias
- Row
- Metal layers

- **Milkyway Library**

Milkyway is a Synopsys library format that stores all of circuit files from synthesis through place and route all the way to signoff. Most Synopsys tools can read and write in the Milkyway format including Design Compiler,

IC Complier, StarRCXT, Hercules, Jupiter & Prime time.

The Milkyway database consists of libraries that contain information about your design. Libraries contain information about design cells, std cells, macro cells and so on. They contain physical descriptions and also logical information.

Milkyway provides 2 types of libraries that we can use (i) reference lib and (ii) design lib. Ref lib contains std cells and hard (or) soft macro cells which are typically created by vendors. Ref lib contains physical info such as routing directions and the placement unit tile dimensions which is the width & height of the smallest instance that can be placed. A design lib contains a design cell which contains reference to multiple reference libraries (std cells & macro cells).

The most commonly used Milkyway views are CEL & FRAM. CEL is the full layout view and FRAM is the abstract view for place and route operations.

FRAM view:

Synopsys tool IC Complier use “FRAM” view as PnR abstract. FRAM view is a cell view that has only the PINs and metal and via blockages defined. This makes sure that the interconnection between the PINs can be routed automatically and that the routing tool will not route over existing metal/via areas thus ruling out any shorts. Cadence EDI tools use LEF views which again has only the PINs and blockages defined,

SDC Contents

- **Clock & Attributes**

- create_clock:

It creates a clock and defines its characteristics. It creates a clock in the current design at the declared source and defines its period and waveform. The STA uses this information to propagate the waveform across the clock network to the clock pins of all sequential elements driven by this clock source.

Eg: create_clock – period 6 -waveform {0 3} -name clk [get_ports clk]

- virtual clock:

A virtual clock is a clock without any source. A Virtual clock is a clock that physically does not exist, but it can be used to constrain a block. This clock is defined, but has not been associated with any pin/port. A virtual clock is used as a reference to constrain the interface pins by relating the arrivals at input/output ports with respect to it with the help of input and output delays.

Eg: create_clock -period 5.0 -name CLK_VIR

- create_generated_clock:

It creates an internally generated clock and defines its characteristics. Derived from a main clock. It might be divided by 2, 4 or 1. It creates a generated clock in the current design at a declared source by defining its frequency with respect to the frequency at the reference pin. The static timing analysis tool uses this information to compute and propagate its waveform across the clock network to the clock pins of all sequential elements driven by this source.

Eg: create_generated_clock -divide_by 2 -source [get_ports clk] -name clkdiv [get_registers clkdiv]

- set_clock_latency:

In Sequential designs, each timing path is triggered by a clock signal that originates from a source. The flops being triggered by the clock signal are known as Sinks for the clock. In general, Clock latency or Insertion delay is defined as the amount of time taken by the clock signal in traveling from its source to the sinks.

Clock latency = Source latency + Network latency

Eg: set_clock_latency -source 2.000 [get_clocks SYSCLK]

- set_clock_uncertainty:

It specifies the uncertainty or skew characteristics of a single clock or between two different clocks. The timing analyzer uses this information to determine the worst possible clock arrival times for each timing check. We can specify uncertainty separately for setup and hold, and can specify separate rising and falling clock transitions. The setup uncertainty is subtracted from the data required time for each applicable path, and the hold uncertainty is added to the data required time for each applicable path.

Eg: set_clock_uncertainty -setup -rise_from clk1 -fall_to clk2 200ps

- **I/O Delay Modeling**

- set_input_delay:

Defines the arrival time of an input relative to a clock. This command sets input path delays on input ports relative to a clock edge. This usually represents a combinational path delay from the clock pin of a register external to the current design.

Eg: set_input_delay 1.4 -clock_fall -clock CLK2 –max {IN1}

- set_output_delay:

Defines the output delay of an output relative to a clock. This command sets output path delays on output ports relative to a clock edge. Output ports have no output delay unless you specify.

Eg: set_output_delay 1.4 -clock_fall -clock CLK2 –max {OUT1}

- set_input_transition:

Sets a fixed transition time on input or inout ports. Sets the max_transition_rise, max_transition_fall, min_transition_rise, or min_transition_fall attributes to the specified transition values on the specified input and inout ports.

Eg: set_input_transition [-clock <name>] [-clock_fall] [-fall] [-max] [-min] [-rise]<transition> <ports>

- set_driving_cell:

This command sets attributes on the specified input or inout ports in the current design to associate an external driving cell with the ports. The drive capability of the port is the same as if the specified driving cell were connected in the same context to allow accurate modeling of the port drive capability for nonlinear delay models.

The following example associates the drive capability of the AND2 library cell with the IN1 port.

Eg: set_driving_cell -lib_cell AND2 {IN1}

- set_max_delay:

Specifies a maximum delay target for paths in the current design. This command specifies that the maximum path length for any startpoint in from_list to any endpoint in to_list must be less than delay_value.

Eg: set_max_delay 15.0 -from {ff1a ff1b} -through {u1} -to {ff2e}

- set_max_transition:

Sets the max_transition attribute to a specified value on specified clocks group, ports or designs.Compile attempts to ensure that the transition time for a net is less than the specified value.

Eg: set_max_transition transition_value [-clock_path][-data_path] [-rise][-fall] object_list

- set_max_capacitance:

Sets the max_capacitance attribute to a specified capacitance value on specified input ports or designs. Compile attempts to ensure that the capacitance value for a net is less than the specified value.

Eg: set_max_capacitance capacitance_value [-clock_path][-data_path] [-rise][-fall] object_list

- set_max_fanout:

Sets the max_fanout attribute to a specified value on specified input ports or designs. Compile attempts to ensure that the sum of the fanout_load attributes for input pins on nets driven by the specified ports or all nets in the specified design is less than the given value.

Eg: set_max_fanout fanout_value object_list

- set_load:

Sets the load to a specified value on a specified port. It is usually used to set what capacitance there is on an output (i.e. the external capacitance that exists)

Eg: set_load 25 out_p

- **Clock Exceptions**

- set_false_path:

Identifies paths that are considered false and excluded from the timing analysis. The false timing paths are paths that do not propagate logic level changes. The path starting points are the input ports or register clock pins, and the path ending points are the register data pins or output ports. This constraint disables setup and hold checking for the specified paths.

Eg: set_false_path -from [get_clocks {clk1}] -to reg_2:D

Eg: set_false_path -through U0/U1:Y

- set_multicycle_path:

Defines the path that takes multiple clock cycles. Setting multiple cycle paths constraint overrides the single cycle timing relationships between sequential elements by specifying the number of cycles that the data path must have for setup or hold checks. If you change the multiplier, it affects both the setup and hold checks.

Eg: set_multicycle_path 3 -from [get_pins {reg1}] –to [get_pins {reg2}]

- set_disable_timing:

Disables timing through the specified cells, pins, ports or timing arcs in the current design.

Eg: set_disable_timing -from A -to Z object_list

Sanity Checks

- **Sanity Checks**

We need to perform some sanity checks before we start our physical design flow, to ensure that inputs received from various team such as synthesis team, library team etc are correct. If we missed this checks than it can create problem in later stage.

Below are input fies which we are mainly checking

1. Library Checks
2. Design/Netlist Checks
3. SDC Checks

1. Library Checks:

It performs consistency checks between logical and physical libraries. Suppose, if designer forgot to load either HVT, LVT or .lib, then there will be inconsistency. It checks library qualities in three main areas:

- Physical library quality
- Logic versus physical library consistency
- Logic versus logic library consistency

ICC Command: check_library

This command shows the name of the library, name of the file, library type & version, time unit, capacitance unit, leakage power unit, current unit and it shows the no of cells missing, no of metals or pins missing in physical and logical library.

2. Design/Netlist Checks:

It checks the current design for conistency. It checks the internal representation of the current design for consistency and issues error and warning messages as appropriate.

It checks the quality of netlist and identifies:

- Floating pins
- Multidriven nets
- Black box module
- Undriven input ports
- Unloaded output ports
- Unconstrained pins
- Pin direction mismatches

ICC Commad: check_design

This command shows the particular i/p port is connected to o/p ports and vice-versa and o/p port is connected to logic 0 and a pin on submodule is connected to logic 1 or logic 0.

3. SDC Checks:

PNR tool wont optimize the paths which are not constrained. So, we have to check if any unconstrained paths are exist in the design. This checks:

- Clock reaching all the clock pins of flops or not
- Ports missing input/output delay

- Ports missing slew/load constraints

- Multiple clocks driving same regsiter

- Unconstrained end points

- Combinational loops

ICC Command: check_timing

This command reports unconstrained paths. If there are any unconstrained paths in the design, run the report_timing_requirements command to verify that the unconstrained paths are false path**.**

report_timing:

It displays timing information about a design. This command provides a report of timing information for the current design. By default, this command reports the single worst setup path in each clock group.

This command shows the data arrival time, data required time, library setup time, clock delay, clock uncertainty and slack. Ithas the startpoint and endpoint.

So report_timing violates if slack is -ve value & it does not violates is slack is +ve or 0 value.

set_zero_interconnect_delay_mode true:

This command enables or disables the zero interconnect delay mode. The zero interconnect delay mode forces the timer to ignore contribution on a timing path from any wire capacitance in a design. It forces all wire capacitance to be as trivial as 0, regardless of the wire load model used in the design. Disabling the mode allows the timer to consider the wire capacitances as it did before enabling the mode.

Floorplan & Physical only cells

- **Floorplan**

Floorplan is one the critical & important step in Physical design. Quality of your Chip / Design implementation depends on how good is the Floorplan. A good floorplan can be make implementation process (place, cts, route & timing closure) cake walk. On similar lines a bad floorplan can create all kind issues in the design (congestion, timing, noise, IR, routing issues). A bad floorplan will blow up the area, power & affects reliability, life of the IC and also it can increase overall IC cost (more effort to closure, more LVTs/ULVTs).

Inputs of Floorplan:

- Technology file (.tf)
- Netlist
- SDC
- Library files (.lib & .lef) & TLU+ file

Steps of Floorplan:

- Die size estimation
- I/O port placement
- Macro placement
- Row creation
- Placement blockages
- Power planning
- Adding physical only cells

Macro Placement:

Macro placement is done based on connectivity information of Macro to IO cells and Macro to Macro. Macro placement is very critical for congestion and timing. Macro placement should result in uniform std cell area.

Macro placement requires:

- Fly-line analysis
- Data-flow diagram
- Design module hierarchy analysis
- Channel length calculation

Guidelines of Floorplan:

- Place the macros close to the boundary of the core
- See the macro pin face core side
- Group the macros belonging to same logic block
- Keep sufficient channels between macros
- May have to control the cell placement around macro
- Apply placement blockages in the channels between macros
- Avoid notches in floorplan

Floorplan Qualification:

- No I/O ports short

- All I/O ports should be placed in routing grid

- All macros in placement grid

- No macros overlapping

- Check PG connections (For macros & pre-placed cells only)

- All the macros should be placed at the boundary

- There should not be any notches. If unavoidable, proper blockages has to be added

- Remove all unnecessary placement blockages & routing blockages (which might be put during floor-plan & pre-placing)

Outputs of Floorplan:

- I/O ports placed

- Cell rows created

- Macro placement final

- Core boundary & area

- Pin positions

- Floorplan DEF file

• **Physical only cells**

Placing the cells before asking the tool to place all the std cells. There are few std cells which needs to be placed across the core area to meet the founder requirement. End-cap cells, tap-cells, I/O buffers, spare cells etc... are pre-placed cells.

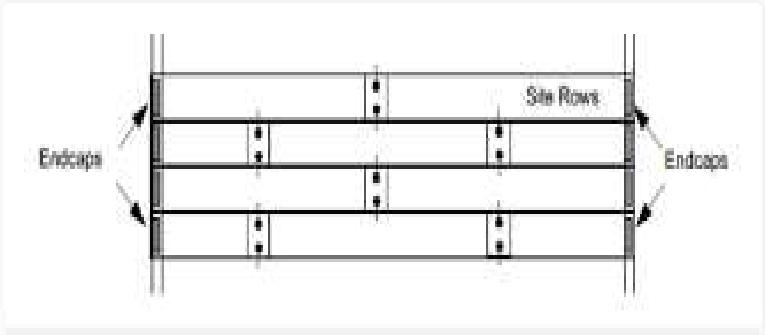
End-cap cells:

The library cells do not have cell connectivity as they are only connected to power and ground rails, thus to ensure that gaps do not occur between well and implant layer and to prevent the DRC violations by satisfying well tie-off requirements for core rows we use end-cap cells.

End-cap cells doesn't have any timing model and contains only base layers. This cell usually placed at the std cell row start and end. This will have VDD& VSS. These are added to know the start and end of the row.

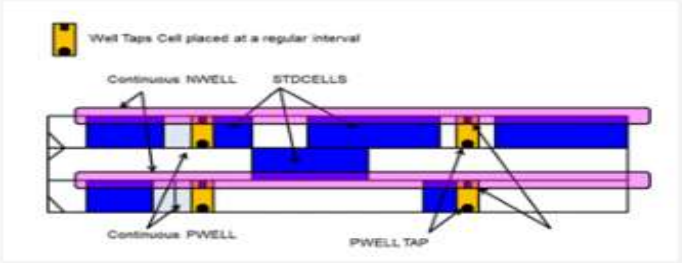
End-cap cells are placed to avoid the cells damages at the start and end of the row to avoid wrong layer wavelength for correct manufacturing. During fabrication, poly variation is more at the edges. To account for that poly variation dummy cells are inserted at the edges of the row so that mfg defects are reduced. During etching process, undesired portion from the surface of a wafer is chemically removed and that should not affect the std cells present in the design. So, by adding end-cap cells prevents such damage.

It also used to address boundary Nwell issues for DRC cleanup. It checks whether nwells are properly terminated.



Tap cells:

Used to prevent Latch-up. This cell connects the power VDD and ground VSS to the nwells and substrate (i.e. nwell to VDD and p-substrate to VSS). It ensures nwell substrate continuity in the design. By placing well taps at regular intervals in design the nwell potential is held constant for proper electrical functionality. It gives reverse bias to the nwell and pwell. The spacing between the well-tap insertion in the design will vary accordingly to the technologies.



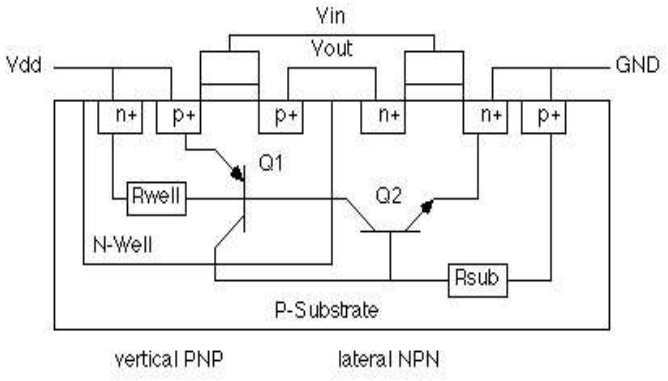
Latch-up:

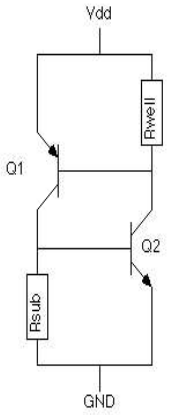
Latch-up refers to short circuit formed between power and ground rails in an IC leading to high current and damage to the IC. Latch-up is the phenomenon of low impedance path between power rail and ground rail due to interaction between parasitic pnp and npn transistors. The structure formed by these resembles as silicon controlled rectifier (SCR, usually known as a thyristor). These form a +ve feedback loop, short circuit the power rail and ground rail, which eventually causes excessive current and can even permanently damage the device.

In a latch-up conduction, the current flows from VDD to GND directly via the two transistors, causing the dangerous condition of a short circuit. The resistors are bypassed and thus excessive current flows from VDD to GND.

Latch-up prevention is easily accomplished

by minimizing Rsub and Rwell.





Decap cells:

Decap cells are temporary capacitors which are added in the design between power and ground rails to counter the functional failure due to dynamic IR drop.

Dynamic IR Drop happens at the active edge of the clock at which a high current is drawn from the power grid for a small duration. To avoid the Dynamic IR drop, charge stores in the cells and release the charge to Nets.

If power source is far from a flop the chances are there that flop can go into metastable state. To overcome decaps are added, when current requirement is high this decaps discharge and provide boost to the power grid.

It is used to remove ground bounce and power bounce. Ground bounce is reduced by placing the more ground I/O pads than power pads.

Decap cells are normally poly gate transistors where source and drain are connected to the ground rail and the gate is connected to the power rail.

More decap cells, then more leakage also will be there. Decap cells are placed as fillers cells in the design.

Filler cells:

Filling 100% of area with normal cells is impossible. Filler cells are used to connect the gaps between the cells after placement. They don’t have the functionality and are used to fill the spaces to prevent the base layer DRC violations.

Filler cells are used to establish the continuity of the Nwells and the implant layers on the standard cells rows, some of the cells also don't have the Bulk Connection (Substrate connection) because of their small size (thin cells).

In those cases, the abutment of cells through inserting filler cells can connect those substrates of small cells to the Power/Ground nets (i.e. those tin cells can use the Bulk connection of the other cells (this is one of the reason why you get stand alone LVS check failed on some cells).

Generally, after placement and routing stage and before LVS& DRC check, these cells are added in the design. If not added, then power/ground open occurs between the std cells.

Tie cells:

Tie cells are used for preventing damage of cells. Tie High cells (Gate one input is connected to VDD, another input is connected to signal net) and Tie low cells (Gate one input is connected to VSS, another input is connected to signal net) are used to connect the gate of the transistor to either Power and Ground.

Directly we can't cannot the pins to VDD/VSS. It has to go through Tie cells. Tie cells are used to enable the logic-0/logic-1 connection. Tie cell should be directly driving and it should not be buffered.

In lower technology nodes, if the gate is connected to Power or Ground. The transistor might be turned "ON/OFF" due to Power or Ground Bounce. These cells are part of the std cell library. The cells which require VDD (Typically constant signals tied to 1) connect to tie high cells. The cells which require VSS/VDD (typically constant signals tied to 0) connect to tie low cells. These cells are part of the std cell library.

Spare cells:

Spare cells are use to implement ECO after Base-To and before Metal-To. If any bugs reported/found after the tape-out, we can use these spare cells to fix the bugs.

Spare cells acts as redundant cells. The key in having spare cells in your design is that you only need to change the metal layers in order to rewire the logic and fix any bugs. This means you only need to pay for new metal masks, thus saving money. The spare cells input pins must be tied to VDD/VSS and output pins left floating. When these cells are required, their inputs are disconnected from VDD/VSS and connected to functional logics in ECO mode.

Spare cells are nothing but the std cell and are placed randomly across the chip for later use. Spare cells are std cells in a design that are not used by the netlist. There would be inclusion of approx. 5% of spare cells in the whole design, not more than that. It basically reduces the mask preparation cost.

Power plan

- **Power plan**

Power planning is done to provide uniform supply voltages to all cells in the design. The primary objective of power planning is to ensure that all on chip components (blocks, memory, I/O cells etc...) have adequate power and ground connections. Three levels of Power Distribution:

1. Rings - Carries VDD and VSS around the chip
2. Stripes - Carries VDD and VSS from Rings across the chip
3. Rails - Connect VDD and VSS to the standard cell VDD and VSS

Core Power Management:

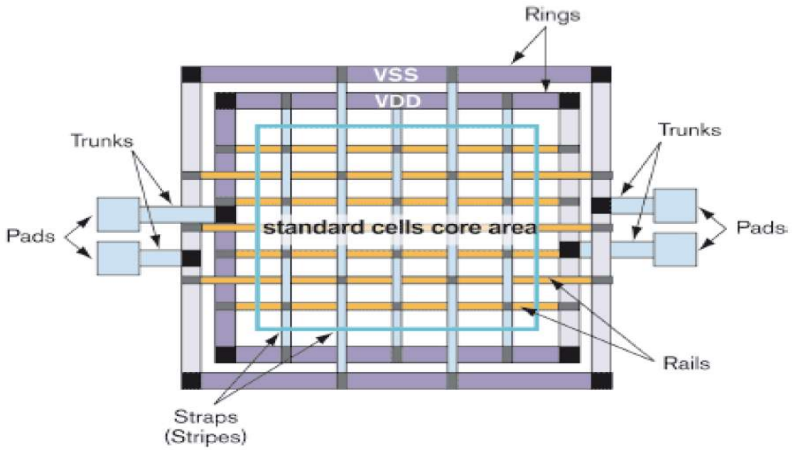
VDD and VSS power rings are formed around the core and macro. In addition to this straps and trunks are created for macros as per the power requirement. Std cell rails are created to tap power from power straps to std cell power/ground pins.

I/O Power Management:

Power rings are formed for I/O cells and trunks are constructed between core power ring and power pads.

Inputs of Power plan:

- Netlist & SDC
- .lib , .lef & tech file
- Tlu+ file
- UPF



UPF Contents:

Power intent specifies:

Power Distribution Architecture:

- Power domains – Group of elements which share a common set of power supply requirements
- Supply rails – Power distribution (ports, nets, sets & switches)
- Shutdown control

Power Strategy:

- Power state tables – Legal combination of states of each power domain
- Operating voltages

Usage of Special cells:

- Isolation cells
- Level shifters
- Power switches
- Retention registers

Isolation Cells:

- Powered off domains do not drive their outputs anymore and these outputs become floating nodes. This could be a problem when other active domains gets these floating nodes as input. It could result in crowbar current which affects the proper functioning of the powered up domain.

- Isolation cells (also called “clamps”) keep the turned off sub-block outputs at a predefined value. This is how the shut-down sub-block does not corrupt other active sub-block functionality.

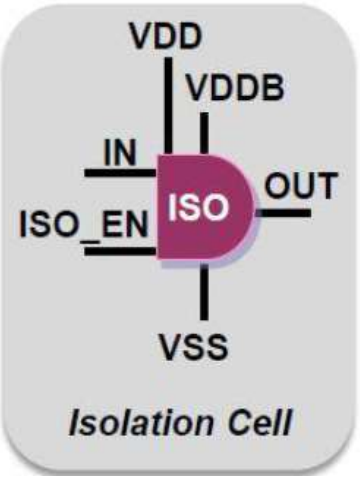
- Isolation cells are powered by a constant supply and drive 0, 1 or latch the old value of the turned off domain.

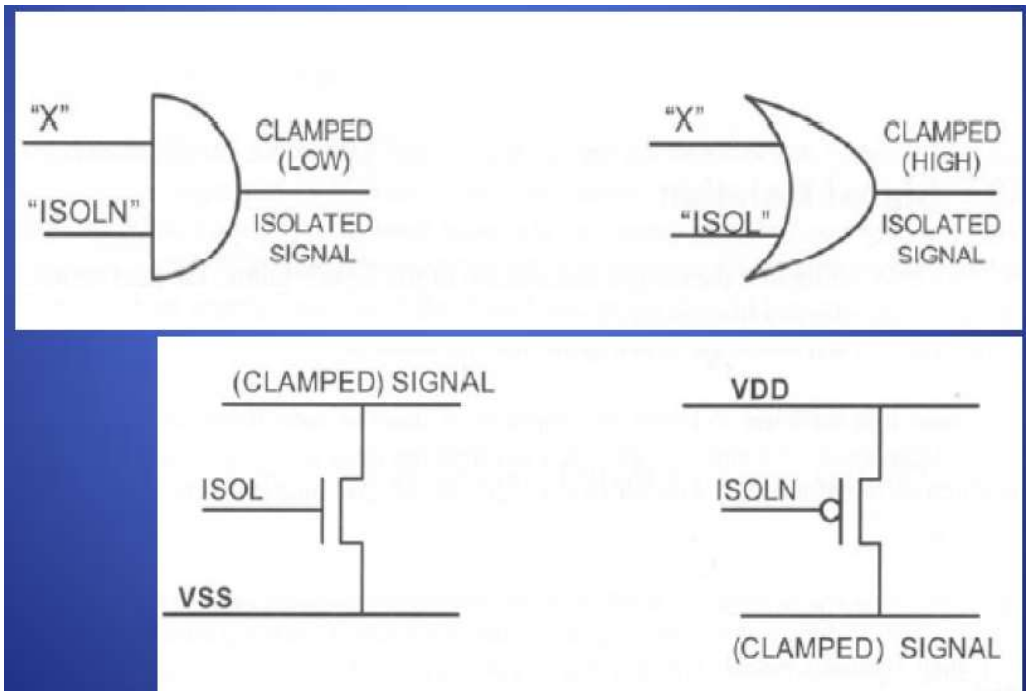
- Isolation cells pass logic values during the normal mode of operation, but clamp it to some specified value when a control signal is asserted.

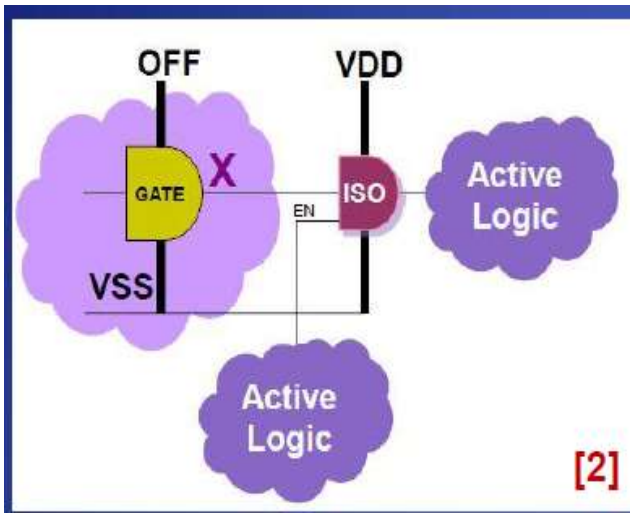
- Isolation cell clamps the output of powered down block to a specified value (‘0’, ‘1’, last)

  - Gate type clamp cells (AND, OR)

  - Transistor type clamp (pull-up, pull-down)







Level Shifters :

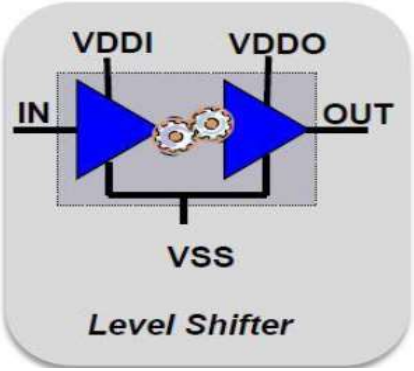
- Level shifters have the minimal functionality of a buffer.

- Necessary as most low-power designs have multi-voltage domains and/or employ dynamic voltage scaling.

- A level shifter swings a logic value in one voltage domain to the **same** logic value in another voltage domain.

- An '**Up**' level shifter swings a logic value from a lower voltage domain to the same logic value in a higher voltage domain.

- A '**Down**' level shifter swings a logic value from a higher voltage domain to the same logic value in a lower voltage domain.



Retention Registers:

- In order to reduce power consumption, memories are shut down where their power domain is switched off or when they are not in use. Registers are corrupted when power is switched off. Corruption is typically represented as 'X' (unknown).

- Some memories need to retain their values for fast wake-up. For these memories, only the memory array stays powered on during the shut-down while the peripheral interfaces are powered off.

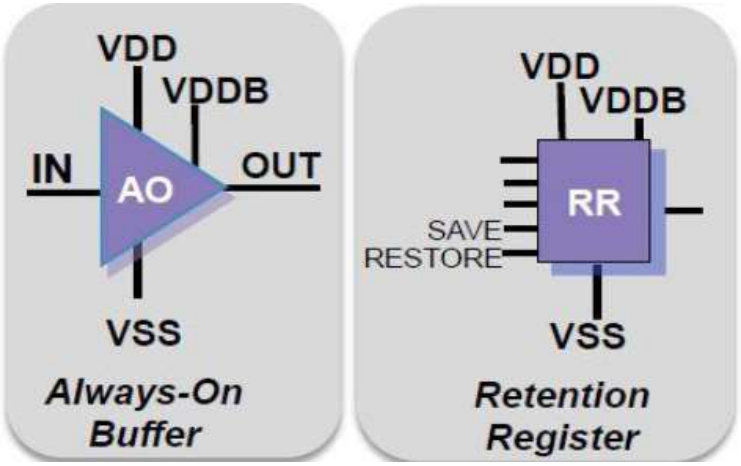
- Retention registers keep their previous active value after being shut down.

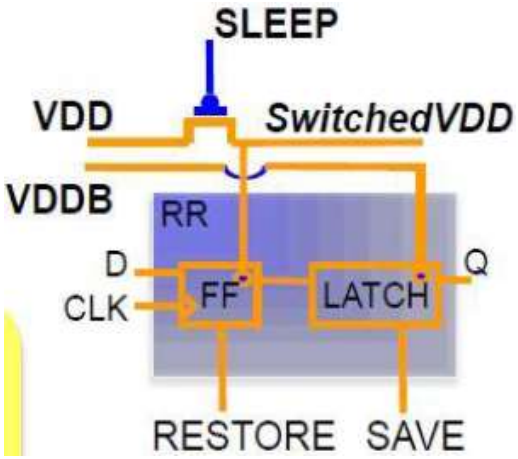
- Retention registers save state information before a power domain is switched off and restore it when the power is turned back on.

- Retention registers comprise of two circuits.

    - Standard register logic, supplied by primary power VDD

    - Shadow latch retention circuitry, with alternate supply VDDB





    - SAVE – transfers FF content into shadow latch during shutdown

    - RESTORE – transfers state from shadow latch to FF when powered back on

Frequently Used Power Reduction Techniques:

Power Gating:

- In a processor chip, certain areas of the chip will be idle and will be activated only for certain operations. But these areas are still provided with power for biasing.

- The power gating limits this unnecessary power being wasted by shutting down power for that area and resuming whenever needed.

- It is used for reducing LEAKAGE POWER or power consumption by switching off power supply to the non operational power domain of the chip during certain mode of operation.

- Header & footer switches, isolation cells, state retention flip flops are used for implementing power gating.

Clock Gating:

- Clock gating limits the clock from being given to every register or flops in the processor. It disables the clock of an unused device. In clock gating the gated areas will still be provided with bias power.

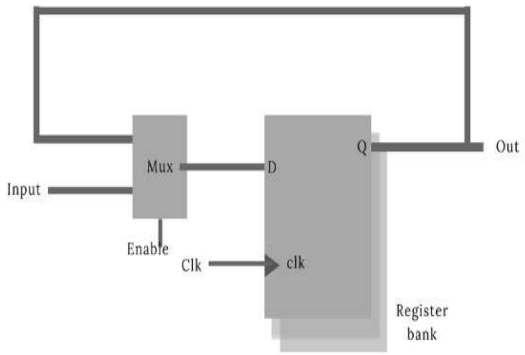
- It is used for reducing DYNAMIC POWER by controlling switching activities on the clock path.

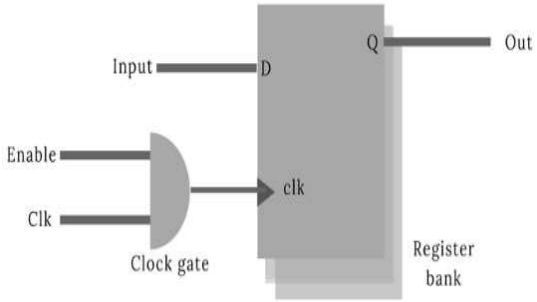
- Generally gate or latch or flip flop based block gating cells are used for implementing clock gating.

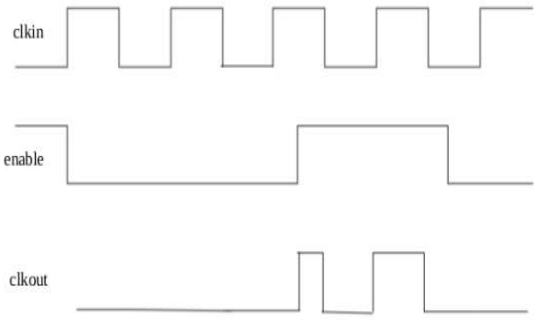
- 50% of dynamic power is due to clock buffer. Since clock has highest toggle rate and often have higher drive strength to minimize clock delay. And the flops receive clocks dissipates some dynamic power even if input and output remains the same. Also clock buffer tree consumes power. One of the techniques to lower the dynamic power is clock gating.

- In load enabled flops, the output of the flops switches only when the enable is on. But clock switches continuously, increasing the dynamic power consumption.

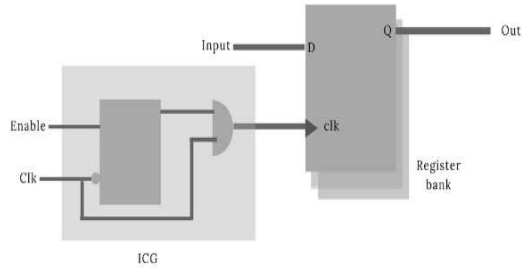
- By converting load enable circuits to clock gating circuit dynamic power can be reduced. Normal clock gating circuit consists of an AND gate in the clock path with one input as enable. But when enable becomes one in between positive level of the clock a glitch is obtained.

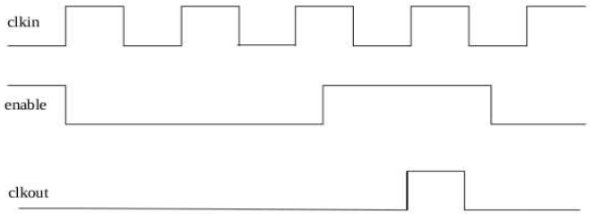






- To remove the glitches due to AND gate, integrated clock gate is used. It has a negative triggered latch and an AND gate.





- Clock gating makes design more complex. Timing and CG timing closure becomes complex. Clock gating adds more gates to the design. Hence min bit width (minimum register bit width to be clock gated) should be wisely chosen, because the overall dynamic power consumption may increase.

Voltage and Frequency Scaling:

- It changes the voltage and clock frequency to match the performance requirements for a given operation so as to minimize leakage.

- Different blocks are operated at variable supply voltages. The block voltage is dynamically adjusted based on performance requirements.

- Frequency of the block is dynamically adjusted. Works alongside with voltage scaling.

Substrate Biasing:

- It changes the threshold voltage to reduce leakage current at the expense of slower switching times.

Multiple Threshold Voltages:

- Uses different Vt in the circuit to reduce leakage but still satisfy timing constraints.

Multiple Supply Voltages:

- Using Multi VDD reduces power consumption by powering down the not used voltage domain. Different blocks are operated at different supply voltages. Signals that cross voltage domain boundaries have to be level shifted.

Memory Partitioning:

- The memory is split into several partitions. Not used ones can be powered down.

Types of Power Dissipation:

The power dissipation is classified in two categories:

- Static power dissipation

- Dynamic power dissipation

Static Power Dissipation:

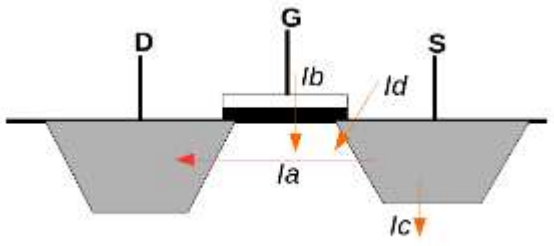
In this class, power will be dissipated irrespective of frequency and switching of the system. It is continuous and has become more dominant at lower node technologies. The structure and size of the device results in various leakage currents. Few reasons for static power dissipation are:

- Sub-threshold current

- Gate oxide leakage

- Diode reverse bias current

- Gate induced leakage



Its hard to find the accurate amount of leakage currents but it mainly depends on supply voltage (VDD), threshold voltage (Vth), transistor size (W/L) and the doping concentration.

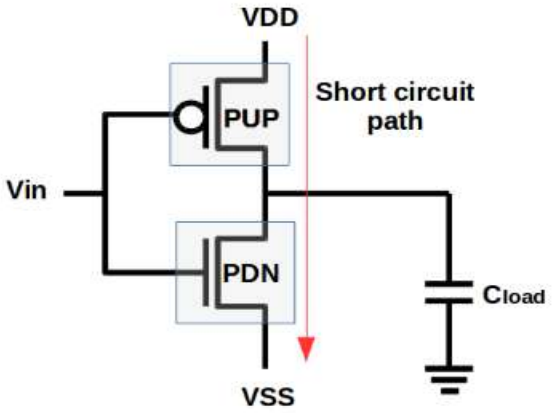
Dynamic Power Dissipation:

There are two reasons of dynamic power dissipation; Switching of the device and short circuit path from supply (VDD) to ground (VSS). This occurs during operation of the device. Signals change their logic state charging and discharging of output mode capacitor.

Short-circuit Power Dissipation:

Because of slower input transition, there will be certain duration of time "t", for which both the devices (PMOS and NMOS) are turned ON. Now, there is a short circuit path from VDD to VSS. This short circuit power is given by:

$$P_{short\text{-}circuit} = V_{dd} . I_{sc} . t$$



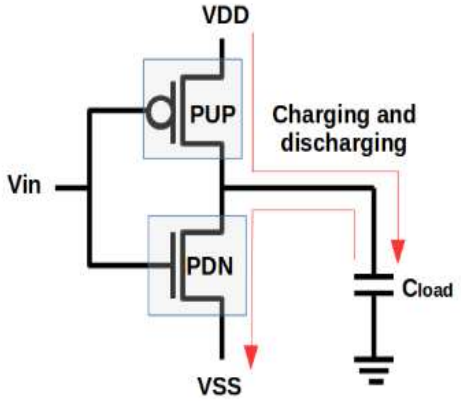
where, Vdd – Supply voltage, Isc – Short-circuit current and t – Short-circuit time.

Short-circuit power is directly proportional to rise time and fall time.

Switching Power Dissipation:

Energy is drawn from the power supply to charge up the output mode capacitance. Charging up of the output cap causes transition from 0V to VDD. So, the power dissipated during charging and discharging of total load [output capacitance + net capacitance + input capacitance of driven cell(s)] is called Switching power dissipation. The switching power is given by:

$$P_{switch} = \alpha \cdot VDD^2 \cdot Cload \cdot f$$



where, α – Switching activity factor, f – Operating frequency,

VDD – Supply voltage & Cload – Load capacitance.

Placement

- **Placement**

In this stage, all the standard cells are placed in the design (size, shape & macro-placement is done in floor-plan). Placement will be driven by different criteria like timing driven, congestion driven, power optimization etc. Timing & Routing convergence depends a lot on quality of placement.

Inputs of Placement:

- Technology file (.tf)

- Netlist

- SDC

- Library files (.lib & .lef) & TLU+ file

- Floorplan & Powerplan DEF file

Goals of Placement:

- Timing, Power and Area optimizations

- Minimum congestion

- Minimal cell density, pin density and congestion hot-spots

- Minimal timing DRVs

Steps of Placement:

1. Pre-placement: Placing the cells before asking the tool to place all the std cells. There are few std cells which needs to be placed across the core area to meet the founder requirement. End-cap cells, tap-cells, I/O buffers, spare cells etc... are pre-placed cells.

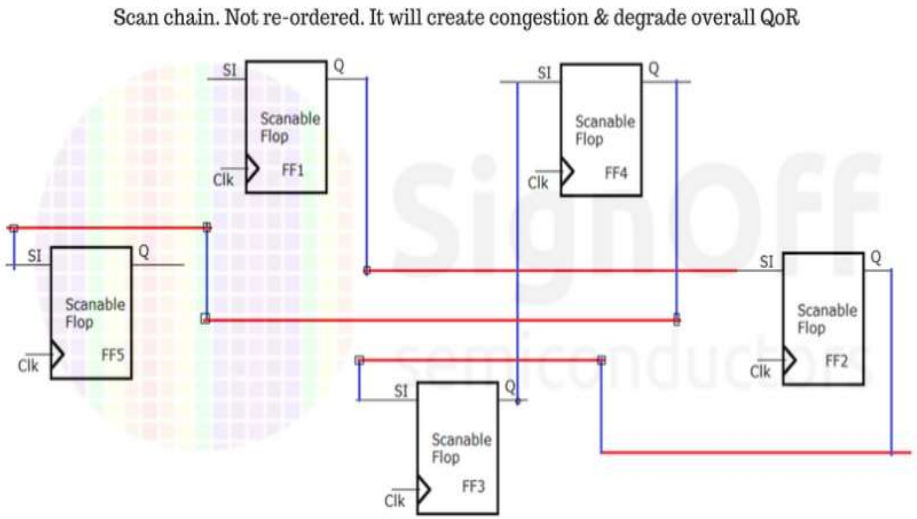
2. Coarse Placement: Initial fast placement of std cells without logical optimization. Makes a good seed for further placement. Some cells may not be legally placed and there can be overlapping.

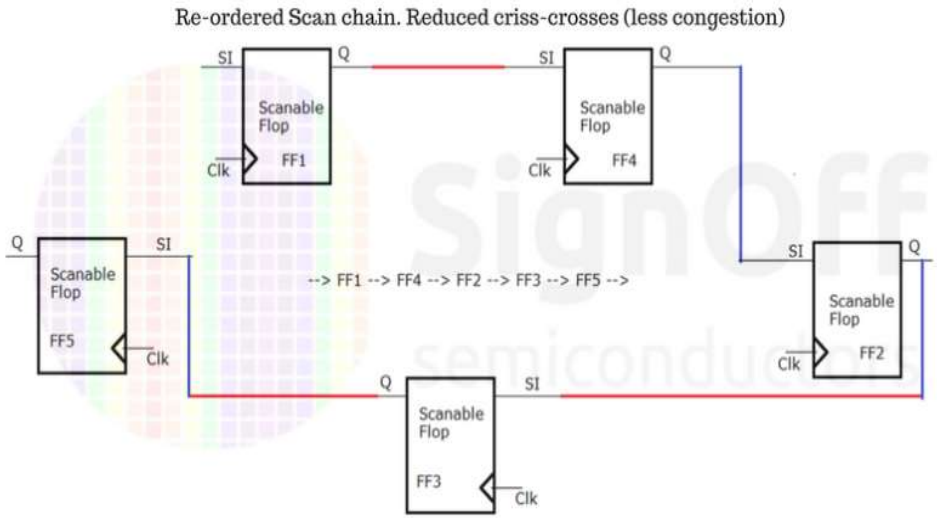
3. Placement Legalization: To avoid overlapping cells need to have legalized locations. Overlapping cells cannot be fabricated, creates shorts etc…

4. HFNs: All high fan-out nets will be synthesized (buffer tree) except clock nets & nets with don't touch attribute. Scan-enable and reset are few examples of high fan-out nets. HFNS honors max fan-out setting. HFNs is the process of buffering the high fan-out nets to balance the load. Too many load affects delay numbers and transition times because load is directly proportional to the delay. BY buffering the HFN the load can be balanced. This is called HFNs.

5. Timing/Power Optimizations: It includes cell sizing, cloning, buffer insertion, area recovery etc…

6. Scan chain reordering: DFT tool flow makes a list of all the scan-able flops in the design, and sorts them based on their hierarchy. In APR tool scan chains are reordered on the basis of placement of flops & Q-SI routing. This is nothing but scan-chain reordering. Scan-chain reordering helps to reduce congestion, total wire-length etc…





Placement Qualification:

- Unplaced cells (should be 0)
- Cells overlap (should be 0)
- Utilization
- Minimal timing issue
- Minimal congestion issue
- Minimal timing DRVs
- Total area after optimization

Outputs of Placement:

- Congestion report
- Timing report
- Design with all std cells placed in core area
- Logs
- Placement DEF file

CTS

- **CTS**

Clock Tree Synthesis (CTS) is one of the most important stages in PnR. CTS QoR decides timing convergence & power. In most of the ICs clock consumes 30-40 % of total power. So efficient clock architecture, clock gating & clock tree implementation helps to reduce power.

The process of distributing the clock and balancing the load is called CTS. Basically, delivering the clock to all sequential elements. CTS is the process of insertion of buffers or inverters along the clock paths of ASIC design in order to achieve zero/minimum skew or balanced skew. Before CTS, all clock pins are driven by a single clock source. CTS starting point is clock source and CTS ending point is clock pins of sequential cells.

Inputs of CTS:

- Technology file (.tf)

- Netlist

- SDC

- Library files (.lib & .lef) & TLU+ file

- Placement DEF file

- Clock specification file which contains Insertion delay, skew, clock transition, clock cells, NDR, CTS tree type, CTS exceptions, list of buffers/inverters etc...

CTS Targets:

- Skew

- Insertion delay

CTS Goals/Constraints:

- Max transition
- Max capacitance
- Max fanout

CTS Flow:

- Read CTS SDC
- Compile CTS using CTS spec file
- Place clock tree cells
- Route clock tree

Clock Latency/Insertion Delay:

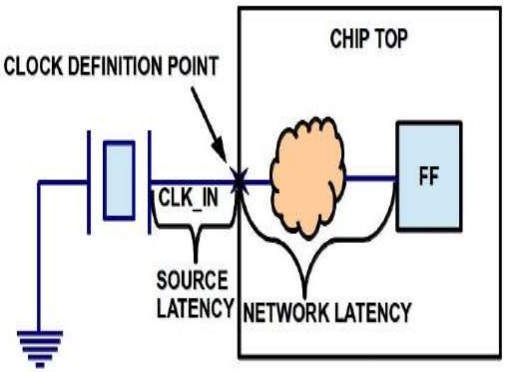
The time taken by the clock to reach the sink point from the clock source is called Latency. It is divided into two parts – Clock Source Latency and Clock Network Latency.

Clock Source Latency:

- The delay between the clock waveform origin point to the definition point.

Clock Network Latency:

- The delay from the clock definition point to the destination/sink point.



Clock Skew:

The difference in the clock latencies of two flops belong to the same clock domain.

- If the capture clock latency is more than the launch clock, then it is positive skew. This helps to meet setup.

- If the capture clock latency is less than the launch clock, then it is negative skew. This helps to meet hold.

Local Skew:

- The difference in the clock latencies of two logically connected flops of same clock domain.

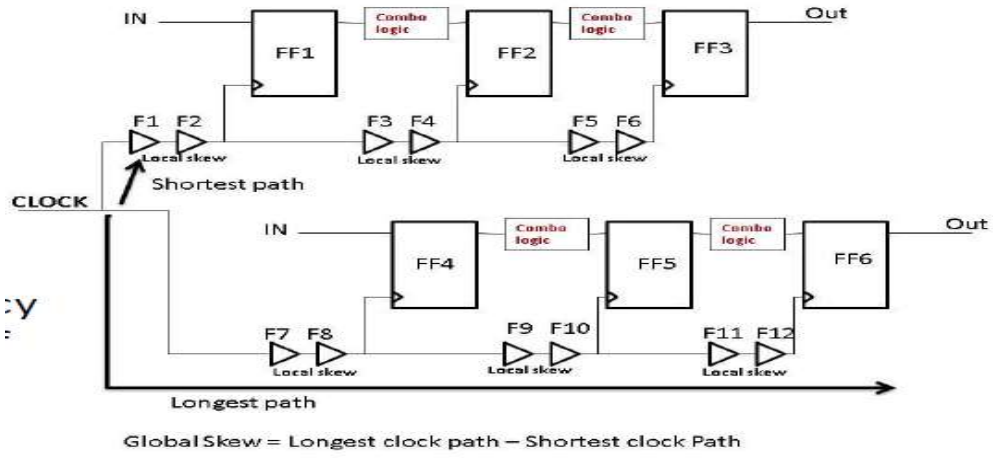
Global Skew:

- The difference in the lowest clock latency and highest clock latency of two flops of same clock domain.

Clock Tree Reference:

By default, each clock tree references list contains all the clock buffers and clock inverters in the logic library. The clock tree reference list is,

- Clock tree synthesis
- Boundary cell insertions
- Sizing
- Delay insertion



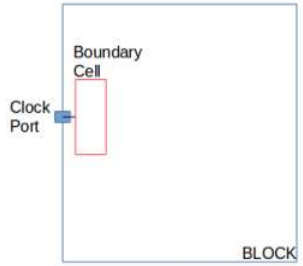
Boundary cell insertions:

- When you are working on a block-level design, you might want to preserve the boundary conditions of the block's clock ports (the boundary clock pins).

- A boundary cell is a fixed buffer that is inserted immediately after the boundary clock pins to preserve the boundary conditions of the clock pin.

- When boundary cell insertion is enabled, buffer is inserted from the clock tree reference list immediately after the boundary clock pins. For multi-voltage designs, buffers are inserted at the boundary in the default voltage area.

- The boundary cells are fixed for clock tree synthesis after insertion; it can't be moved or sized. In addition, no cells are inserted between a clock pin and its boundary cell.



Delay insertion:

- If the delay is more, instead of adding many buffers we can just add a delay cell of particular delay value.

- Advantage is the size and also power reduction. But it has high variation, so usage of delay cells in clock tree is not recommended.

Clock Tree Exceptions:

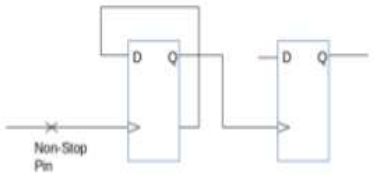
Non-Stop pin:

Non-stop pins trace through the endpoints that are normally considered as endpoints of the clock tree.

Example:

- The clock pin of sequential cells driving generated clock are implicit non-stop pins.

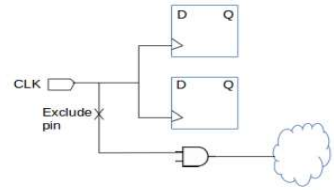
- Clock pin of ICG cells



Exclude pin:

Exclude pin are clock tree endpoints that are excluded from clock tree timing calculation and optimization. The tool considers exclude pins only in calculation and optimizations for design rule constraints. During CTS, the tool isolates exclude pins from the clock tree by inserting a guide buffer before the pin or these pins are need not to be considered during the clock tree propagation.

Example - Non clock input pin of sequential cell

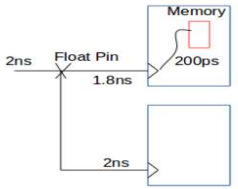


In the above figure, beyond the exclude pin the tool never perform skew or insertion delay optimization but does perform design rule fixing.

Float pin:

Float pins are clock pins that have special insertion delay requirements and balancing is done according to the delay [Macro modeling]. This is same as sync pin but internal clock latency of the pin is taken into consideration while building the clock tree. To adjust the clock arrival for specific endpoints with respect to all other endpoints.

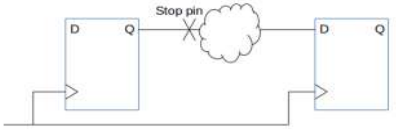
Example - Clock entry pin of hard macros



Stop pin:

Stop pins are the endpoints of clock tree that are used for delay balancing. In CTS, the tool uses stop pins in calculation & optimization for both DRC and clock tree timing.

Example - Clock sink are implicit stop pins



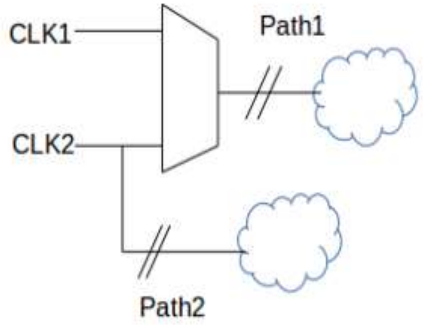
The optimization is done only upto the stop pin as shown in the above fig. The clock signal should not propagate after reaching the stop/sync. This pin needs to be considered for building the clock tree.

Don't Touch Sub-tree:

If we want to preserve a portion of an existing clock tree, we put don’t touch exception on the sub-tree.

- CLK1 is the pre-existing clock and path 1 is optimized with respect to CLK1.

- CLK2 is the new generated clock. Don’t touch sub-tree attribute is set w.r.t C1.



Example:

- If path1 is 300ps and path2 is 200ps, during balancing delay are added in path2.

- If path1 is 200ps and path2 is 300ps, during balancing delay can’t be added on path1 because on path1 don’t touch attribute is set and we get violation.

Don't Buffer Net:

It is used in order to improve the results, by preventing the tool from buffering certain nets. Don’t buffer nets have high priority than DRC. CTS do not add buffers on such nets.

Example - If the path is a false path, then no need of balancing the path. So set don’t buffer net attribute.

Don't Size Cell:

To prevent sizing of cells on the clock path during CTS and optimization, we must identify the cell as don’t size cells.

Specifying Size-Only Cells:

During CTS & optimization, size only cells can only be sized not moved or split. After sizing, if the cells overlap with an adjacent cell after sizing, the size-only cell might be moved during the legalization step.

Clock Tree Structures:

Cluster Based:

Most commonly used approach. Based on location of clock sinks, group them into clusters. It builds tree for all individual clusters. Balance clusters by adding buffers at the root of the cluster.

H-Tree:

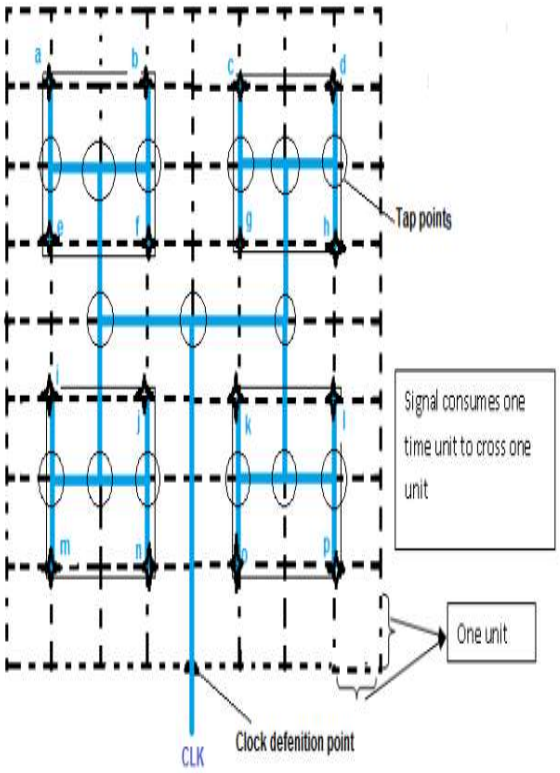
It is called Binary tree. Each driver has 2 symmetric sinks. Can be able to achieve very low skew with reasonable buffer/inverter count. Clock port will be at the centre and skew will be minimal because of its nature (branching kind of model). H-Tree structure will be routed in higher layers.

Advantages:

- Balanced latencies & Low skew

Disadvantages:

- Requires big driver, thus lots of power
- Requires more routing resources



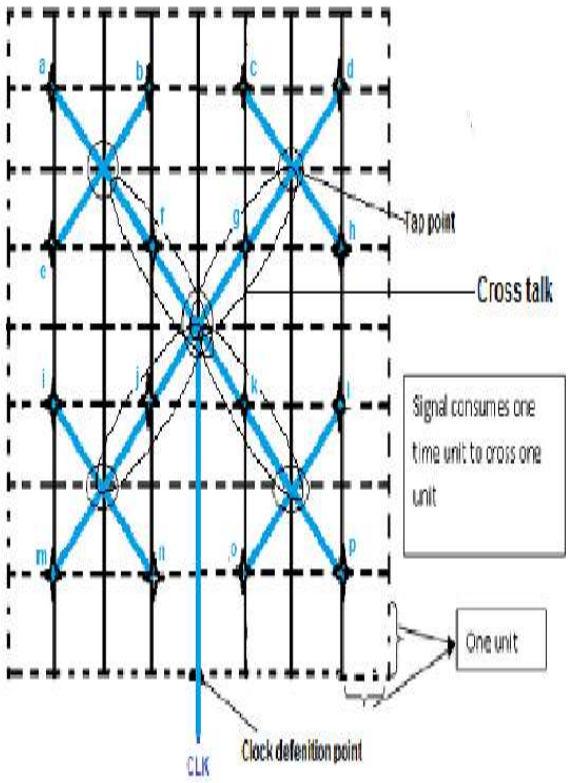
X-Tree:

Advantages:

- Balanced latencies & Low skew

Disadvantages:

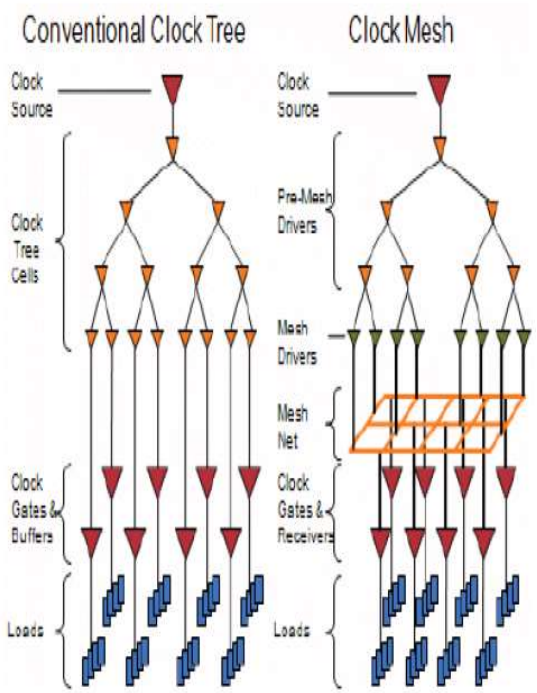
- Crosstalk



Conventional Clock Tree & Clock Mesh:

Generally used on very high speed design like MCUs. The clock mesh includes a clock source, pre-mesh drivers, mesh drivers, the mesh net, clock gates, mesh receivers and loads. The clock gating cells are spread uniformally in design area irrespective of the clock sinks. Based on placement, clock sinks are connected to clock gating cells.

The main difference between conventional clock tree and clock mesh is the presence of the mesh net. Another major difference is that the mesh drivers are connected to the mesh net as a multi-driven net.



Advantages:

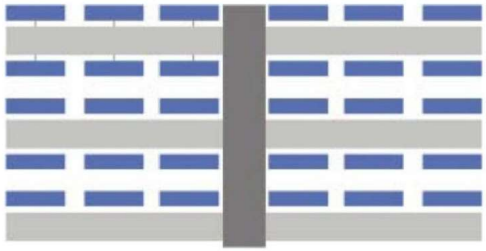
- Low skew
- High OCV tolerance

Disadvantages:

- Requires more power charge the parasitic R&C of mesh
- Requires more routing resources
- Difficult to implement for the designs having more than one clock

Spine Tree:

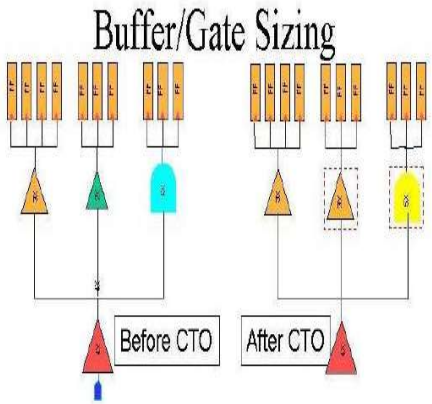
The spine tree (Fish bone) arrangement makes it easy to reduce the skew but it is heavily influenced by process parameters and may have problems with phase delay.



CTS Optimization Techniques:

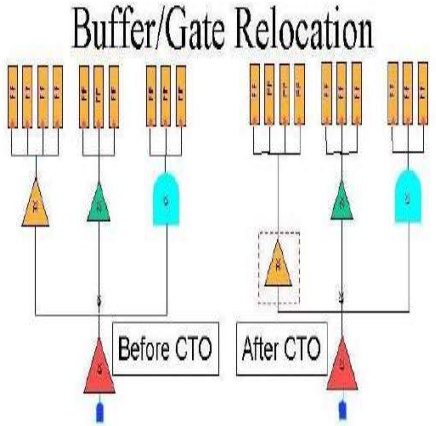
1. Buffer/Gate Sizing:

Sizes up or down buffers and gates to improve both skew and insertion delay.



2. Buffer/Gate Relocation:

Physical location of the buffer or gate is moved to reduce skew and insertion delay.



3. Delay Insertion:

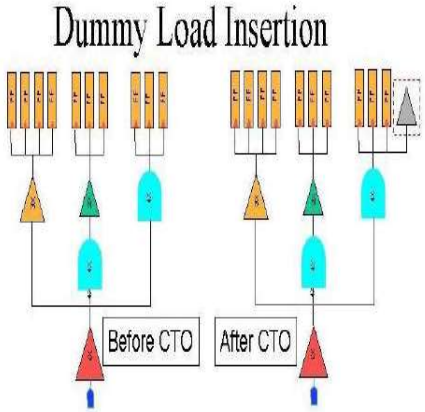
Delay is inserted for shortest paths.

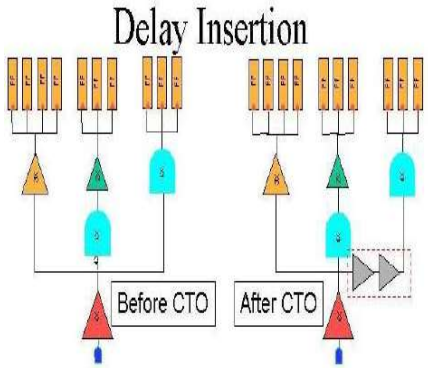
4. Dummy Load Insertion:

Uses load balancing to fine tune the clock skew by increasing the shortest path delay.

Outputs of CTS:

- Timing report
- Congestion report
- Skew report
- Insertion delay report
- CTS DEF file





Routing

- **Routing**

Making physical connections between signal pins using metal layers are called Routing. Routing is the stage after CTS and optimization where exact paths for the interconnection of standard cells and macros and I/O pins are determined. Electrical connections using metals and vias are created in the layout, defined by the logical connections present in the netlist (i.e. Logical connectivity converted as physical connectivity).

After CTS, we have information of all the placed cells, blockages, clock tree buffers/inverters and I/O pins. The tool relies on this information to electrically complete all connections defined in the netlist such that:

- There are minimal DRC violations while routing.

- The design is 100% routed with minimal LVS violations.

- There are minimal SI related violations.

- There must be no or minimal congestion hot spots.

- The Timing DRCs & QOR are met and good respectively.

Inputs of Routing:

- Netlist

- All cells & ports should be legally placed with clock tree structure & CTS DEF file

- NDRs

- Routing blockages

- Technology data (metal layers (lef, tech file etc...), DRC rules, via creation rules, grid rules (metal pitch) etc...)

Goals of Routing:

- Minimize the total interconnect/wire length.
- Minimize the critical path delay.
- Minimize the number of layer changes that the connections have to make (minimizing the number vias).
- Complete the connections without increasing the total area of the block.
- Meeting the Timing DRCs and obtaining a good Timing QoR.
- Minimizing the congestion hotspots.
- SI driven: reduction in cross-talk noise and delta delays.

Routing Constraints:

- Set constraints to number of layer to be used during routing.
- Setting limits on routing to specific regions.
- Setting the maximum length for the routing wires.
- Blocking routing in specific regions.
- Set stringent guidelines for minimum width and minimum spacing.
- Set preferred routing directions to specific metal layers during routing.
- Constraining the routing density.
- Constraining the pin connections.

Routing Flow:

The different tasks that are performed in the routing stage are as follows:

- Global Routing (also performed during placement stage)
- Track assignment
- Detailed Routing
- Search and Repair

Global Routing:

- It divides entire design into routing regions and generates a tentative route for each net. Each net is assigned to a set of routing region.
- Design is divided into small bins and then it identifies the available tracks and assign layers to nets. It calculates how many available tracks are there and it will be overlapped.
- It avoids long detours.
- Repetitive global route runs to reduce congestion.
- It does not specify the actual layout of wires and it is not sensitive to DRV violation.

Track Assignment:

Track assignment is a stage wherein the routing tracks are assigned for each global routes. The tasks that are performed during this stage are as follows:

- Assigning tracks in horizontal and vertical partitions.
- Rerouting all overlapped wires.

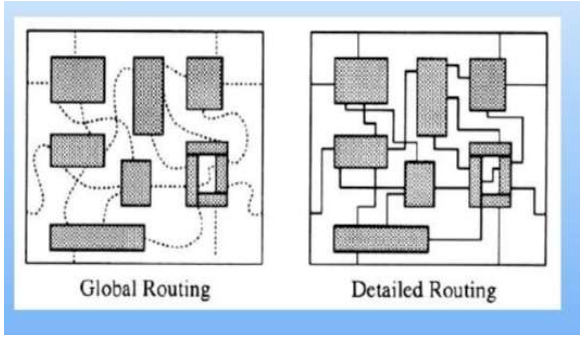
Track Assignment replaces all global routes with actual metal layers. Although all nets are routed(not very carefully), there will be many DRC, SI and timing related violations, especially in regions where the routing connects the pins. These violations are fixed in the succeeding stages.

Basically, it completes first cut routing with all vias inserted between wires.

Detailed Routing:

The main goal of detailed routing is to complete all of the required interconnect without leaving shorts or spacing violations (DRC violations). It follows routing topology from Global/Track route. Here the actual layout of wires is specified.

It completes the connection by adding vias. It routes a small area at a time (sbox) and transverses the whole design box by box until entire routing is complete. Generally routed in 2 phase:



- First make all connections without worrying about DRC.

- Verify DRC and incrementally fix them till DRC count is 0.

Detailed Routing - Incremental Fixes:

- Based on DRC type, tool will either spread the wires or fill a gap. These fixes might cause some more violations like Notch filling can cause spacing violations because of increased width and wire spreading can cause violation in adjacent region.

- Tool will analyze them and fix them. This is iterative loop.

Search and Repair:

The search-and-repair stage is performed during detailed routing after the first iteration. In search-and-repair, shorts and spacing violations are located and rerouting of affected areas to fix all possible violation is executed.

Routing Optimization:

Routing optimization is a step performed after detailed routing in the flow. Inaccurate modeling of the routing topology may cause timing, signal integrity and logical design constraint related violations. This may cause conditions wherein fixing a violation would create other violations and many such scenarios may cascade to make it very difficult for timing closure with no timing DRCs. Hence it is necessary to fix and optimize the routing topology. Routing optimization involves:

- Fixing timing violations & timing DRVs (max transition, max capacitance and max fanout).
- Fixing LVS (opens & shorts) & DRCs.
- Finding & Fixing Antenna violations (using jumpers and antenna diodes).
- Area and Leakage power recovery.
- Fixing SI related issues.
- Redundant via insertion.

Outputs of Routing:

- Routing db file or DEF file with no opens & shorts
- Timing report
- Congestion report
- Skew & Insertion delay report
- Geometric layouts of all nets

Congestion Analysis

- **Congestion**

If the number of routing tracks available for routing in one particular area is less than the required routing tracks then the area said to be congested. There will be a limit for number of nets that can be routed through particular area. Pins inside GRC decide required tracks and Technology decides available tracks.

- **Reasons for Congestion**

  - High Standard cell density in small area
  - Placement of standard cells near macros
  - Pin density due to high fan in cells like AOI,OAI
  - Bad floorplan (No proper blockages, halos and macro placement)
  - Macros/Standard cells might have used the all the metal layers inside and no routing resources
  - Power straps and clock network might have used more routing resources
  - Placing macros at the centre instead of boundary

- **How to fix Congestion?**

1. Higher cell density can cause for congestion. By default the cell density can be upto 95%. We can reduce the cell density at congested areas by using coordinate option.

   ICC Command: set_congestion_options - max_util 0.6 - coordinate {x1 y1 x2 y2}
   set placer_max_cell_density_threshold 0.6 --> set the placement max density

   Here we set the maximum cell density upto 60% and given the coordinates for the particular area.

2. If the design is congested, we can rerun place_opt with the -congestion and -effort high options.

ICC Command: place_opt -congestion-driven -effort high

During congestion driven placement, the cells (Higher cell density) which caused for congestion are spread apart.

3. Reduce local cell density using partial placement blockage. For eg, to define a partial blockage with a maximum allowed cell density of 60 percent (a blocked percentage of 40), enclosed by the rectangle with corners at (10, 20) and (100,200), use the following ICC command.

ICC Command: create_placement_blockage -bbox {10 20 100 200} -type partial -blocked_percentage 40

4. If we have more pin density, which can be reduced by adding cell-padding to the cells which is causing congestion. Cell padding can be applied by setting the keepout margin command.

ICC Command: set_keepout_margin -type soft -outer {x1 y1 x2 y2} [get_selection]

5. By using blockages and halos. They prevent the tool placing cells in that particular locations to give enough space for routing near the macros.

6. Change the Floorplan (macros placement, macros spacing and pin orientation) such that cells will be distributed uniformly.

7. Reordering the scan chain to reduce congestion.

The Congestion report provides a summary of the congestion information, which includes the total congestion, the horizontal congestion, and the vertical congestion.

Both Dirs: Overflow = 48090 Max = 15 (1 GRCs) GRCs = 47264 (0.18%)

H routing: Overflow = 26229 Max = 12 (1 GRCs) GRCs = 26149(0.10%)

V routing: Overflow = 21861 Max = 4 (3 GRCs) GRCs = 21115(0.08%)

In the congestion report:

- The Overflow value is the total number of wires in the design global routing cells that do not have a corresponding track available.

- The Max value corresponds to the highest number of over utilized wires in a single global routing cell.

- The GRCs value is the total number of over congested global routing.

STA

- **What is STA?**

STA is one of the techniques used to verify the timing of a digital design.

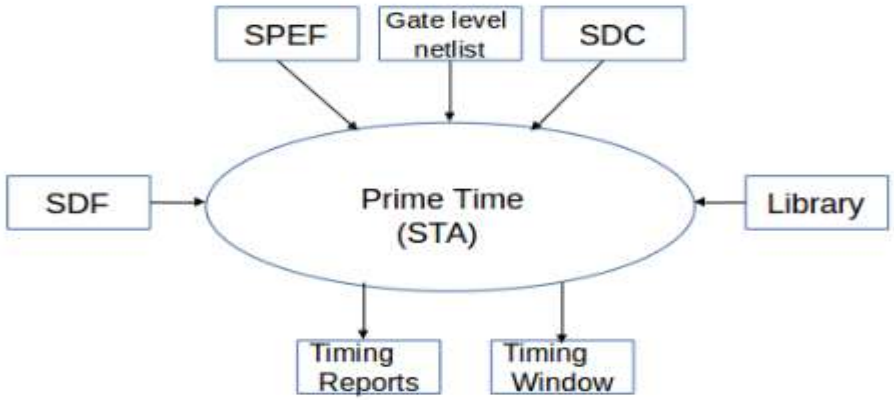
STA provides faster and simpler way of checking and analyzing all the timing paths in the design for any timing violations.

STA tool calculates Arrival time, Required time and Slack.

Difference between DTA & STA

Dynamic Timing Analysis	Static Timing Analysis
Simulation based	Formula based
Highly accurate	More pessimistic
Slow run time	Very fast
Not possible for whole Chip	Most popular for sign off Checks

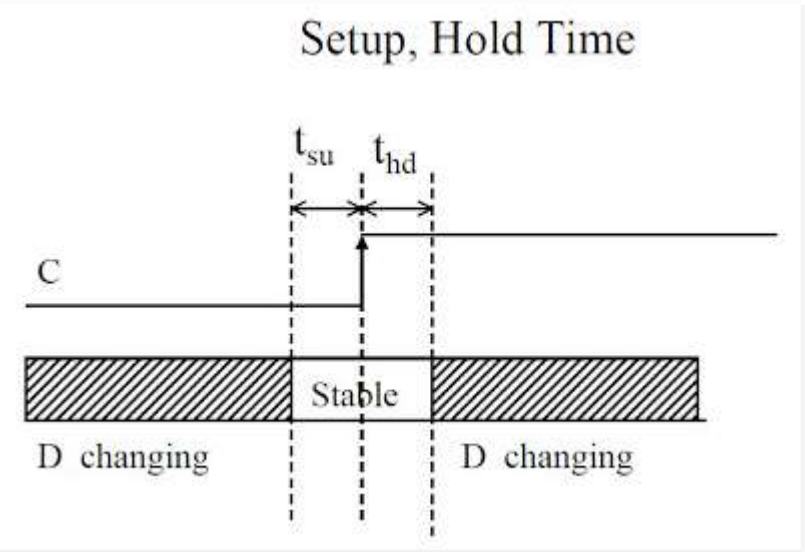
Input & Output Files of STA

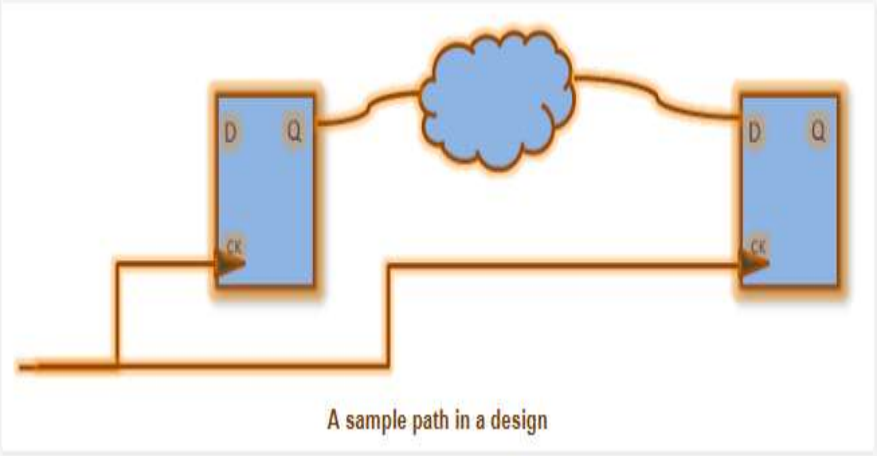


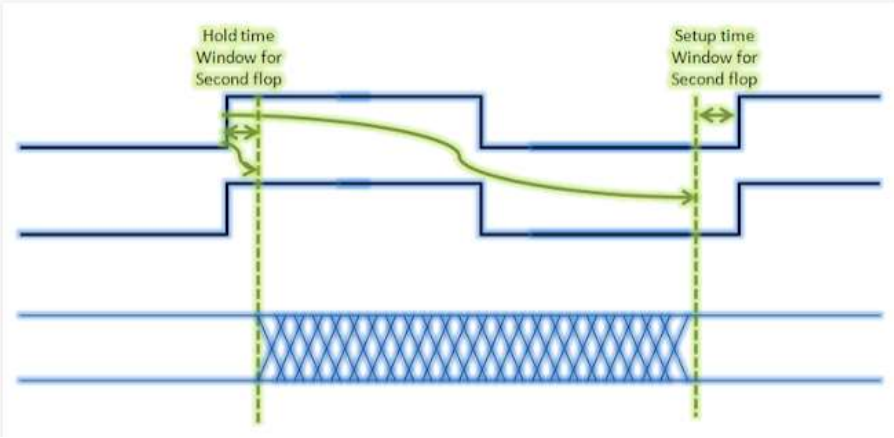
• **Setup & Hold Definition**

Setup time is the minimum amount of time the data signal should be held steady before the clock event so that the data are reliably sampled by the clock.

Hold time is the minimum amount of time the data signal should be held steady after the clock event so that the data are reliable sampled.







- **Setup & Hold Analysis**

Data should be stable Ts time before the posedge of FF2/C

If Ts = 0 ns, then, data launched from FF1 at time = 0ns should arrive at D of FF2 before or at time = 10ns. If data takes too long to arrive, it is reported as setup violation.

If Ts = 1ns, then, data launched from FF1 at time = 0ns should arrive at D of FF2 before or at time = 10 – 1 = 9ns. If data takes too long to arrive, it is reported as setup violation

Data should be stable Th time after the posdge at FF2/C.

To satify the hold condition at FF2 for the data launched by FF1 at 0ns, the data launched at FF1 at 10ns should not reach at FF2/D before 10ns+Th time.

If Th = 0.5ns, then we can say that the data launched from FF1 at time 10 ns doesn't propagate so soon that it reaches at FF2 before time 10.5 ns.

- **Setup & Hold Violation**

If Setup time is Ts for a flipflop and if data is not stable before Ts time from active edge of the clock, there is a Setup violation at that flipflop. So if data is changing in the non-shaded area before active clock edge, then it's a Setup violation.

For Setup check, generally we consider Maximum delay along Data path and Minimum delay along clock path.

Setup Slack = Required time – Arrival time

Arrival time(max) = clock delay FF1(max) + clock to Q-delay FF1(max) + combinational logic delay(max)
Required Time = clock adjust + clock delay FF2(min) – set up time FF2 – clock uncertainty
Here clock adjust = clock period(since setup is analyzed at next edge)

So, Setup violates if Arrival time is more than Required time.

If Hold time is Th for a flipflop and if data is not stable after Th time from active edge of the clock, there is a Hold violation at that flipflop. So if data is changing in the non-shaded area after active clock edge, then it's a Hold violation.

For Hold check, generally we consider Minimum delay along Data path and Maximum delay along clock path.

Hold Slack = Arrival time – Required time

Arrival time(min) = clock delay FF1(min) + clock to Q-delay FF1(min) + combinational logic delay(min)
Required Time = clock adjust + clock delay FF2(max) + hold time FF2 + clock uncertainty
Here clock adjust = 0(since hold is analyzed at same edge)

So, Hold violates if Required time is more than Arrival time.

- **Ways to fix Setup Violation**

During Placement Stage:

1. Timing path groups - We can use this option to resolve Setup timing during placement stage. Groups a set of paths or endpoints for cost function calculations. The delay cost function is the sum of all groups (weight * violation), where violation is the amount for which setup was violated for all paths within the group. If there is no violation within a group, its cost is zero. Groups enable you to specify a set of paths to optimize even though there might be larger violations in another group. When endpoints are specified, all paths leading to those endpoints are grouped.

ICC Syntax: group_path [-weight weight_value] [-critical_range range_value] -name group_name [-from from_list] [-through through_list] [-to to_list]

Eg: group_path -name "group1" -weight 2.0 -to {CLK1A CLK1B}

2. Create Bounds – We can constrain the placement of relative placement cells by defining move bounds with fixed coordinates. Both soft bounds and hard bounds are supported for relative placement cells, and both rectangular bounds and rectilinear bounds are supported. To constrain relative placement by using move bounds, use the create_bounds command.

ICC Command: create_bounds -coordinate {100 100 200 200} "U1 U2 U3 U4" -name bound1

3. If the design is having timing violation, we can rerun place_opt with the -timing and -effort high options.

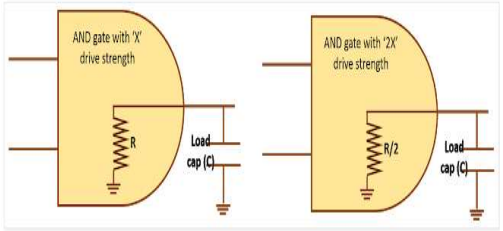
ICC Command: place_opt -timing-driven -effort high

Timing driven placement tries to place cells along timing critical paths close together to reduce net RCs and meet setup timing.

4. Change the Floorplan (macros placement, macros spacing and pin orientation) to meet the better timing.

After Placement Stage:

1. Increase the drive strength of data-path logic gates - A cell with better drive strength can charge the load capacitance quickly, resulting in lesser propagation delay. Also, the output transition should improve resulting in better delay of proceeding stages. A better drive-strength gate will have a lesser resistance, effectively lowering the RC time constant; hence, providing less delay. This is illustrated in figure 1 below. If an AND gate of drive strength 'X' has a pull down resistance equivalent to 'R', the one with drive strength '2X' will have R/2 resistance. Thus, a bigger AND gate with better drive strength will have less delay.



2. Use data-path cells with lower threshold voltages - HVT Swap. Means change HVT cells into SVT/RVT or into LVT. Low Vt decreases the transition time and so propagation delay decreases. So, replace HVT with RVT or LVT will speed up the timing.

3. Buffer insertion – If net length is long, then we insert Buffer to boast. It decreases the transition time, which decreases the wire delay. If the amount of wire delay decreases due to decreasing of transition time > cell delay of buffer, then overall delay decreases.

4. Reduce the amount of buffering in the path – It will reduce the cell delay but increase the wire delay. So, if we can reduce more cell delay in comparison to wire delay, the effective stage delay increases.

5. Route the net using Higher metal layers.

6. Replace buffers with 2 inverters – Adding inverter decreases the transition time 2 times then the existing buffer gate. Due to that, the RC delay of the wire decreases. Cell delay of 1 buffer gate = cell delay of 2 inverter gate.

7. Play with clock skew: Positive skew helps improve the setup slack. So, to fix setup violation, we may either choose to increase the clock latency of capturing flip-flop, or decrease the clock latency of launching flip-flop. However, in doing so, we need to be careful regarding setup and hold slack of other timing paths that are being formed from/to these flip-flops. This is called Useful Skew. So, basically, Useful skew is nothing but adding delay intentionally in the clock path in order to meet the better timing.

•**Ways to fix Hold Violation**

- Hold violation is just opposite of setup violation. Hold violation happens when data is too fast compared to the clock speed. For fixing the hold violation, delay should be increased in the data path.

1. Increase the drive strength of data-path logic gates

2. Use data-path cells with higher threshold voltages

3. Buffer insertion/removal

4. Route the net using Higher metal layers

5. Increase the clk->q delay of launching flip-flop

- **Timing DRVs**

  - Max Tran

  - Max Cap

  - Max Fanout

Causes:

1. HVT cells give slower transition - The HVT cells have larger threshold voltages compared to LVTs and RVTs. Hence, they take more time to turn ON resulting in larger transition time.

2. Weak Driver - The driver won't be able to drive the load resulting in bad transition of the driven cell. Thus the delay increases.

3. Load is more - The driving cell cannot drive load more that what it is characterized for. This is set in .lib using max cap value. If the load that a cell sees increases beyond its maximum capacitance value, then it causes bad transition and hence increases delay.

4. Net length is large - Larger the net length, larger the resistance, worser the transition. Thus results in trans violation. The RC Value of a long net will increase the load seen by a cell causing max cap violations as well.

5. Fanout is too large - If the fanout number increases beyond the limit of what the driver cell in characterized for, it causes max fanout violations. The increased load results in max cap violation which indirectly causes max tran violation as well.

Fixes:

Max Tran:

- Replace HVT cells with LVT cells.

- Up size the driver.

- Reduce the net length by adding buffers. Longer the nets, larger the resistance. Putting a buffer at the middle of a long net splits the resistance into half.

- Reduce the load by reducing fanout and downsizing the driven cell.

Max Cap:

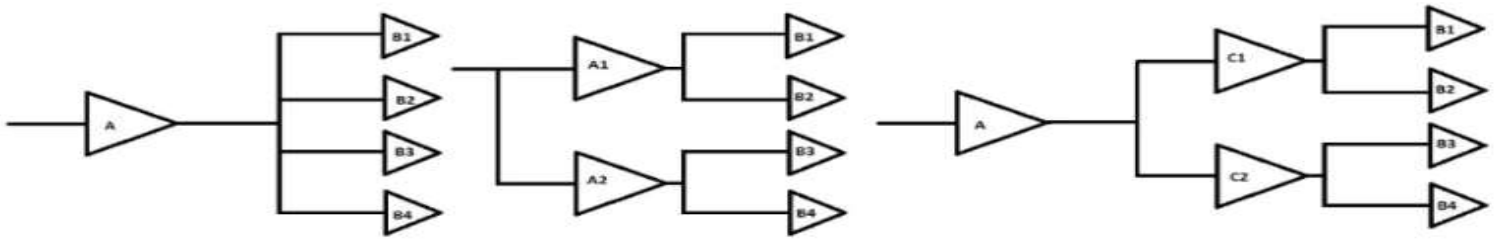
- Up size the driver.

- Split long nets by buffering.

- Reduce the load by reducing the fanout (by load splitting) or by downsizing the driven cell.

Max Fanout:

- Reduce the fanout by load splitting by buffering or cloning.



(a) (b) Cloning (c) Buffering

- Fig. (a) shows a buffer driving four other cells. In fig. (b), the load is split using Cloning. The first buffer is cloned and each buffer now drives half of the load. In fig.(c), the load is split using buffering. Two new buffers are added at the output of buffer A. Now buffer A is driving C1 and C2 and each of them are driving half of the load.

- **PVT**

PVT is abbreviation for Process, Voltage and Temperature. In order to make our chip to work in all possible conditions, like it should work in Siachen Glacier at -40°C and also in Sahara Desert at 60°C, we simulate it at different corners of process, voltage and temperature which IC may face after fabrication. These conditions are called as corners. All these three parameters affect the delay of the cell. We will see each and every parameter and its effect on delay in detail.

Process:

Process variation is the deviation in attributes of transistor during the fabrication. During manufacturing a die, the area at the centre and that at the boundary will have different process variation. This happens because layers which will be getting fabricated can not be uniform all over the die. Below are few important factors which can cause process variation.

- Wavelength of the UV light

- Manufacturing defects

The affects of process variation are listed below;

- Oxide thickness variation

- Dopant and mobility fluctuation

- Transistor width, length etc.

- RC Variation

These variations will cause the parameters like threshold voltage to change its value from expected. Threshold voltage depends on oxide thickness, source-to-body voltage and implant impurities

$$I_D = (1/2)\mu_n C_{ox} (W/L)(V_{GS} - V_{Th})^2$$

The current flowing through the channel directly depends upon mobility (μ_n), oxide capacitance C_{ox} (and hence thickness of oxide i.e. t_{ox}) and ratio of width to length.

Any of these parameters change, it will result in changing the current. In other words, it will affect the delay of the circuit. Delay decreases with increase in current.

Voltage:

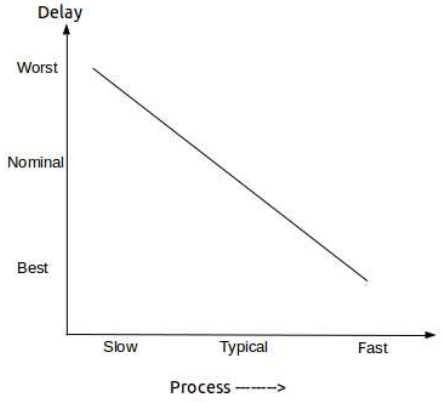
Now a days, supply voltage for a chip is very less. Lets say chip is operating at 1V. So there are chances that at certain instance of time this voltage may vary. It can go to 1.1V or 0.9V

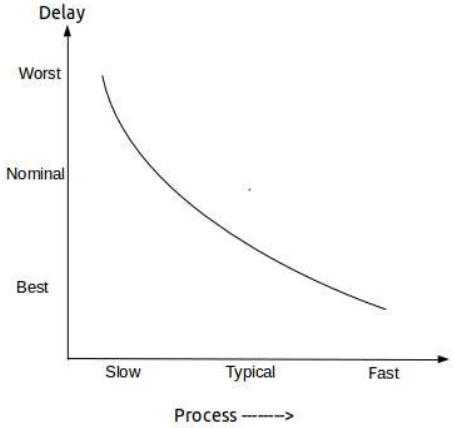
The important reason for supply voltage fluctuations is IR drop. IR drop is caused by the current flow over the parasitic resistance of the power grid. IR drop reduces the supply voltage from the required value. The second important reason for voltage variation is supply noise caused by parasitic inductance in combination with resistance and capacitance. The current through parasitic inductance causes the voltage bounce. Supply voltage that any chip works on is given externally. It can come from DC source or some voltage regulator. Voltage regulator will not give same voltage over a period of time. It can go above or below the expected voltage and hence it will cause current to change making the circuit slower or faster than earlier.

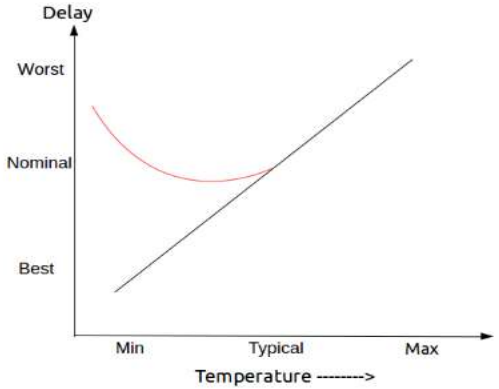
Temperature:

The temperature variation is with respect to junction and not ambient temperature. The temperature at the junction inside the chip can vary within a big range and that's why temperature variation need to be considered. Delay of a cell increases with increase in temperature. But this is not true for all technology nodes. For deep sub-micron technologies this behaviour is contrary. This phenomenon is called as temperature inversion.

Temperature inversion: The delay depends on the output capacitance and I_D current (directly proportional to C_{out} and inversely proportional to I_D). When the temperature increases, delay also increases (due to the variation in carrier concentration and mobility). But when temperature decreases, delay variation shows different characteristics for submicron technologies. For technology nodes below 65nm, the delay will increase with decrease in temp and it will be maximum at -40°C. This phenomena is known as temperature inversion.







RC Variation

RC variation is also considered as corners for the setup and hold checks. RC variation can happen because of fabrication process and the width of metal layer can vary from the desired one.

Critical corners for Setup and Hold check

We always check our chip to work in worst scenarios. We should be very pessimistic about setup and hold checks. So consider worst case scenarios.

Setup violation can be caused if data is coming very slow. So the condition when process is slow, voltage is minimum and temperature is maximum is the worst case for setup check. Also because of temperature inversion at lower technology node, delay will increase as temperature decrease. Hence lowest temperature results in more delay. It is not compulsory that the delay at lowest temperature is always less than delay at highest temperature.

Hold violation is caused if data comes faster. So process should be faster, voltage should be maximum and temperature should be minimum.

Now if setup and hold are checked in worst corners, then the chip should work in every scenario. Still we check them in typical corners because we need to analyse power consumption. Refer following table for the worst case scenarios for setup and hold.

	Mode	Process	Voltage(V)	Temperature(°C)	RC/C
Setup	Functional	SS	0.9	125	RC worst/Cworst
	Functional	SS	0.9	-40	RC worst/Cworst
Hold	Functional	FF	1.1	-40	RC best/Cbest

Table: Worst Scenarios for setup and hold

On Chip Variations (OCV)

Variations are of two types:

1. Global variations:

These are PVT variations that depend on external factors like Process, Supply Voltage and Temperature. ICs are fabricated in batches and hence exhibit die to die variations. Some exhibit strong process (fast switching) and weak process (slow switching). These are known as inter-chip variations.

2. Local variations:

Local variations are also variations in PVT, but these are intra-chip variations known as OCV

Process:

All the transistors in a chip cannot be expected to have the same process. There can be variations in channel length, oxide thickness, doping concentration, metal thickness etc due to imperfections in manufacturing process like mask print, etching etc.

Voltage:

The supply voltage reaching the power pins will not be the same for all standard cells. The power network has a finite resistance. Consider two cells, one which is placed closer, and other placed far. As the interconnect length for the farther cell is more, it has more resistance and results in a higher IR drop, thereby reducing the supply voltage reaching the cell. As the voltage is less, this cell has more delay than the cell which is placed closer.

Temperature:

The transistor density within a chip is not uniform. Some regions of the chip have higher density and higher switching, resulting in a higher power dissipation. Hence the junction temperature at these regions are higher, forming localized hot spots. This variation in temperature across the chip can result in different delays

Derates

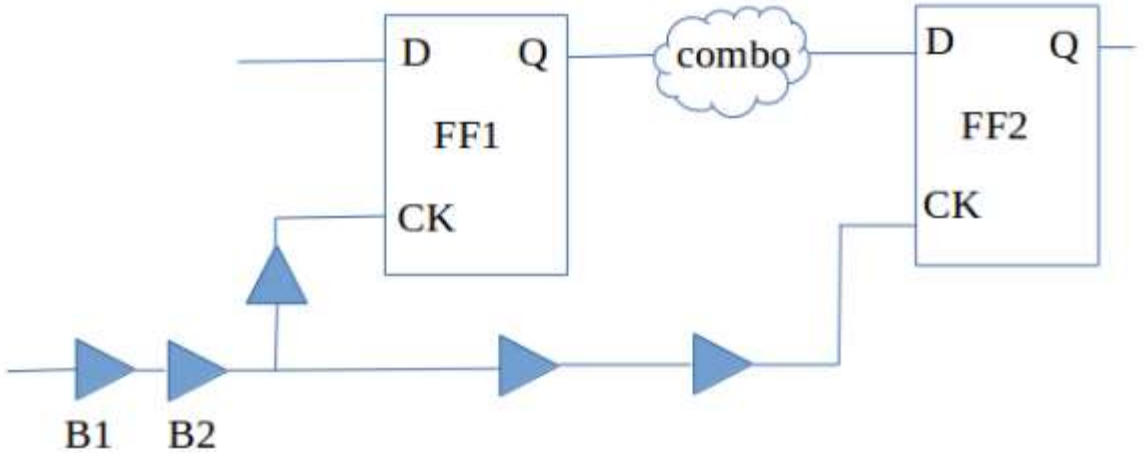
As a result of OCV, some cells may be fast or slow than expected. If these variations are not accounted, results may be pessimistic and can lead to setup or hold violations. In order to model these, we introduce derates. Timing derates are multiplied with the net delay and cell delay for the launch and capture clock paths. This is given as say x%. Let us consider a timing derate of 8% and how it is accounted in setup and hold analysis.

Setup analysis:

Setup check is done in worst case. The setup check is more pessimistic when the launch clock reaches late than the capture clock. Here we multiply the launch path delays with late derate of 1.08 and the capture path delays with an early derate of 0.92.

Hold analysis:

Hold check is done in best case. Hold check is more pessimistic when the launch clock reaches early than the capture clock. Here we multiply the launch path delays with an early derate of 0.92, and capture path delays with a late derate of 1.08.



Consider the above path from FF1 to FF2. For setup analysis, as per the previous example, a late derate of 1.08 is applied for the launch path and an early derate of 0.92 is applied for the capture path.

The launch clock path and the capture clock path share a portion of the clock tree (B1, B2) and then diverge from the common point. This common path delays are multiplied with different derates (early and late), resulting in different delays. These cells have max delay in launch path and min delay in capture path. The same cell cannot have different delays at the same time. This results in additional pessimism which has to be removed. Here comes the need for **Clock Re-convergence Pessimism Removal (CRPR) or Common Path Pessimism Removal (CPPR)**. This pessimism value is the difference between the max and min delay at the common clock path. To reduce pessimism, CRPR is added to required time in setup analysis and subtracted from required time in hold analysis.

ICC Command: set_timing_derate -max -early -net_delay 0.9

set_timing_derate -max -late -net_delay 1.1

Sign-Off

- **Crosstalk**

It occurs when there are parallel running nets of same metal layer for long distance. When a signal on a net switches very fast then, the wires in its surroundings tends to switch too and their logical values or transition can be affected. This is called cross talk. It causes noise bumps in non-switching nets and advancing/delaying of transition in switching nets.

Crosstalk Noise

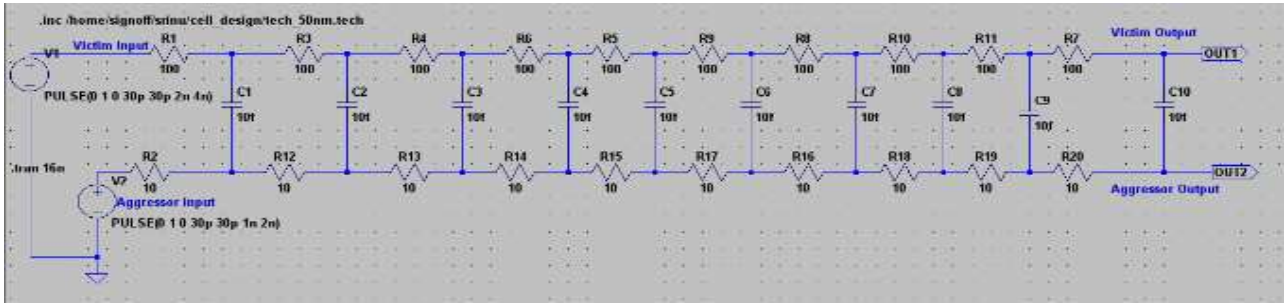
It is undesired change in the logical values of victim due to switching in the input of aggressor. If one net is switching and other is at a constant value , the switching net may cause voltage spikes on other net. This is called as cross talk noise. Cross talk noise is evolving as a key source in degrading performance and reliability of high speed integrated circuits.

Crosstalk Delay

When there is some delay or advancement in output transition of victim due to input transition of aggressor, it is called as cross talk delay. It occurs when some transition is happening in both the nets. Cross talk delay depends on the switching direction of the aggressor and victim nets.

If input transitions in aggressor and victim occur in same direction then output transition of victim becomes faster and it is earlier. It may cause hold violations.

If input transitions occur in opposite directions then output transition of victim becomes slower and it is delayed, which may cause setup violations.



- **<u>Ways to fix Crosstalk effects</u>**

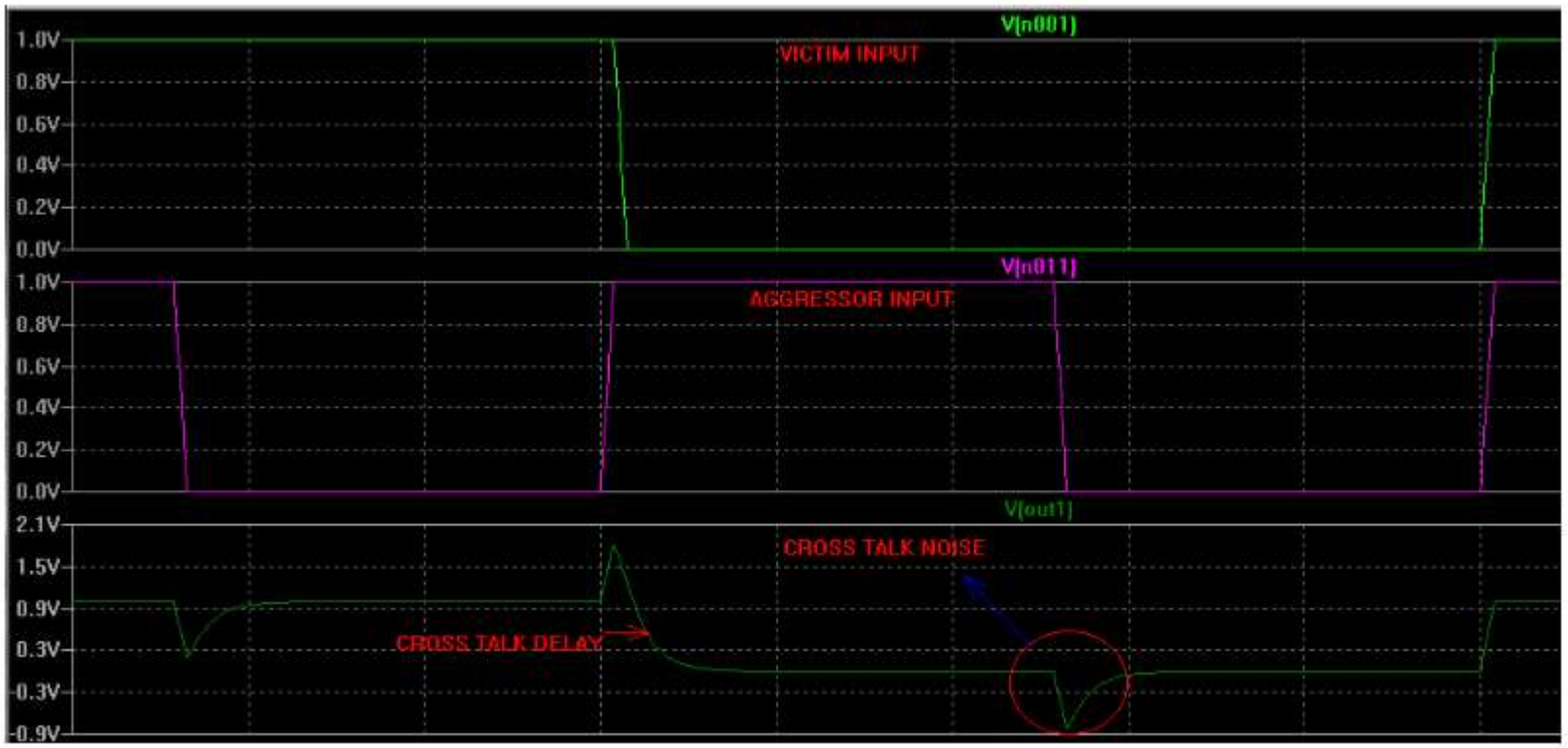
  - Assign Non default routing (NDR) rules – Increase spacing.

  - Insert buffer to split the long nets

  - Jump to different metal layers

  - Shielding – Ground lines run in between the signal lines so that they form coupling capacitance with ground

  - By up-sizing the driver of victim net or downsizing driver of aggressor net



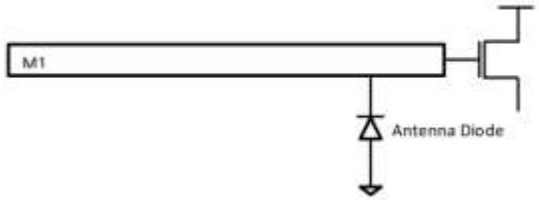
• **Antenna Effect**

During IC fabrication, the wafer usually undergoes various processing steps, such as metalization (laying of metal wires) and etching (to make the surface flat). During the metalization step, few of the nets connecting the gate terminals can be floating as upper metal layers have not been fabricated yet. In the plasma etching process (widely used in recent fabrication processes), there may be accumulation of unwanted electrostatic charges on these floating nets, which act as antennas.

Typically, in ICs nets are driven either from source or drain of the device and connects a receiver gate terminal over a gate oxide. The gate gets ruptured when the amount of these charges are more than threshold and there by leading to a total chip failure. This phenomenon of an electrostatic charge being discharged into the device is known as “antenna effect”. The threshold is decided by metal layer area to gate area ratio.

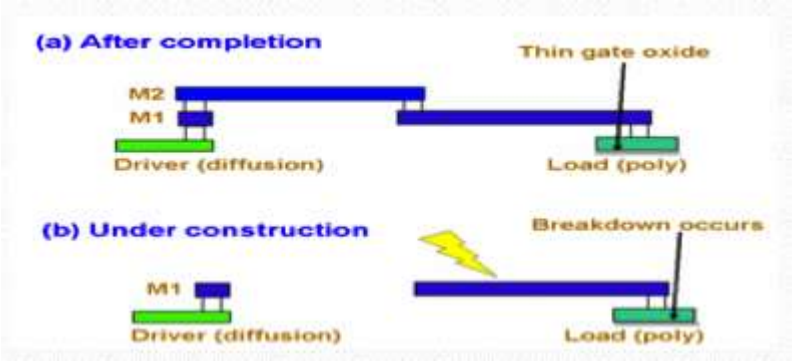
Ways to fix Antenna violation:

1. Use Antenna diode – We use antenna diodes to overcome the antenna effects. Zener diodes will be connected to the metal layers to remove the excess charge. To avoid this deposition of charge at the gate of a transistor, a diode is generally used in reverse biased mode which can drain out the charge without affecting the transistor circuitry. For this we generally make use of n-type diode because p-type diode would need extra biasing of its nwell.



2. Metal hopping - When the metal connected to a gate is long and there is space for a higher metal hop , it is always advisable to do so instead of using an antenna diode in order to avoid antenna violations. Suppose we jog a metal 2 net with metal 3. Then while etching metal 2, the part of the net which is drawn in metal 2 only comes into picture because metal 3 has not been manufactured yet. So the effective charge reduces. This is the reason for using a higher metal as a jumper.

So, it breaks the long net in the same layer and route part of the net using upper metal layer.



3. Decrease the ratio in lef file and do routing

4. Insert the buffers

- **Multi-cut Vias**

Multi-cut vias are a DFM technique to increase the yield. In an IC, a via is used to connect a metal track of one metal layer with a metal track of another metal layer. Generally, for signal lines other than the power lines, one via is provided for each connection point. Such via is called a “single-cut via''.

With decreasing technology nodes in IC industry, the metal width gets reduced and also the cross-sectional area of a via will decrease. Accordingly in the mfg process, it has become difficult to form a via of a desired pattern. In a worst case scenario, an open failure occurs in the single-cut via formation part and thus the desired device operation cannot be implemented, resulting in a lower yield.

Further, as the cross-sectional area of a via decreases, the delay time in signal lines increases and disconnection rate also increases due to EM. This causes lowering of device reliability.

As wayout of these issues, multiple vias may be provided in parallel for each connection point. Such via is called as ''multi-cut via''. If two vias are provided for each connection point, such a via is called a ''double-cut via''. After layout, single-cut vias are replaced with multi-cut vias as many as possible so that the device reliability improves.

• **Redundant Via Insertion / Double Via Insertion**

During IC fabrication, there may be partial or complete via failure due to various reasons such as cut misalignment, electro migration and thermal stress induced voiding.

A partial via failure increases the resistance of the signal nets and may increase the delay causing difficulty in timing closure. Whereas a complete via failure will result in broken nets. This leads to increase in yield loss and hence is critical to fix this issue. To fix yield loss due to via failure, a good practice is the insertion of redundant vias adjacent to single vias for support. Redundant vias are basically extra single vias on minimal width nets. It is to be noted that redundant vias are required for support to increase the yield, unlike muli-cut vias on wide nets for functionality purposes.

Although redundant via insertion is a step in Design for Manufacturability (DFM) it is more advantageous to insert redundant vias in the post-route optimization stage or during the detailed routing stage. This is because during the DFM stage the layout is almost complete and can be modified only slightly without causing other cascaded violations. Hence this restricts the insertion of redundant vias at single via points, resulting in an increasing chance of yield loss due to via failure.

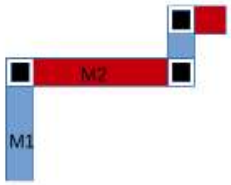
The advantages of redundant vias are:

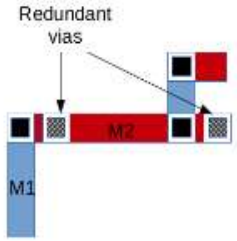
- Nets are less likely to break.

- Yield is improved.

- Decrease in the resistance of the vias.

- Avoids increase of net delay due to partial via failure.





• **IR Drop Analysis**

IR Drop can be defined as the voltage drop in metal wires constituting power grids before it reaches the vdd pins of the cells. IR drop occurs when there are cells with high current requirement or high switching regions. IR drop causes voltage drop which in-turn causes the delaying of the cells causing setup and hold violations. Hold violations cannot be fixed once the chip is fabricated.

There are two types of IR drop analysis namely:

Static IR drop analysis:

- Independent of the cell switching, the drop is calculated with the help of wire resistance.

- Calculates the average voltage drop of entire design assuming current drawn across is constant.

- As average current is calculated this analysis depends on time period. This analysis is good for signoff checks in older technology.

Dynamic IR drop analysis:

- Depends on switching activity of the logic.

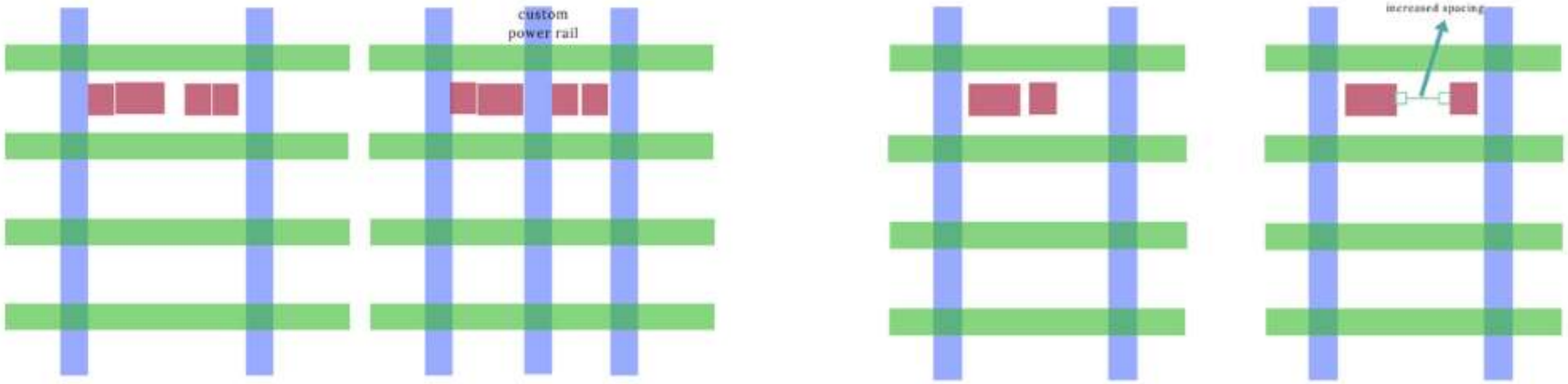
- Is vector dependent .

- Less dependent on clock period as depends on instantaneous current.

- Analysis of peak current demand and highly localized cells.

Methods to reduce IR drop:

1. Robust power mesh – Initial power grid is made based on static IR analysis due to late availability of switching activity. If there is IR drop due to some of the clustered cells then adding a strip will make the power mesh more robust.



2. De-cap - These are decoupling capacitors which are spread across the high switching region to maintain the voltage.

3. Spacing - If clock cells are clustered and causing IR drop, then by spacing them apart near to different power rails will reduce the IR drop. While shifting the cell to next power rail, it should be made sure that the power rail is not driving many cells, because adding another cell may give IR drop.

4. Reducing load - Cells driving more load will be drawing more current. Hence reducing load will reduce IR drop.

5. Downsizing - Cells of smaller size will draw less current. But the transition of cells should not become worse.

6. Increase the width of stripes. If width is already high, increase no of stripes.

7. Check if any via is missing and check if there is any scope to add more vias.

8. The number of power switches can be increased to reduce IR drop.

9. It should be made sure that all the power pins of macros are properly connected to the power rails.

Note: IR drop analysis is done in RC worst corner (corner having more resistance of rails) and FF process, high voltage and high temp corner (PVT corner) because current is drawn more in this corner.

- **EM**

Electro migration (EM) refers to the unwanted movement of materials in a semiconductor. If the current density is high enough, there can be a momentum transfer from moving electrons to the metal ions making the ions to drift in the direction of the electron flow. This results in the gradual displacement of metal atoms in a semiconductor, potentially causing open and short circuits. Due to high current density and resistance of metal in the recent technologies EM has become dominating.

Methods to Fix EM:

1. Widen the wire to reduce current density

2. Keep the wire length short

3. Increase metal width

4. Downsize the drive

5. Add more vias

6. Layer switching is another option as upper metal layers in the technology have higher current driving capability (due to greater thickness)

Note: Clock nets are more prone to EM, as those are more frequently switched signals as compared to other nets. Because of this only, we avoid using highest drive strength clock buffers to build clock tree.

• **DRC**

Design rules are set of parameters provided by semiconductor manufacturers to the designers, in order to verify the correctness of a mask set. It varies based on semiconductor manufacturing process. These rule set describes certain restrictions in geometry and connectivity to ensure that the design has sufficient margin to take care of any variability in manufacturing process.

Design rule checks are nothing but physical checks of metal width, pitch and spacing requirement for the different layers with respect to different manufacturing process. If we give physical connection to the components without considering the DRC rules, then it will lead to failure of functionality of chip, so all DRC violations has to be cleaned up.

After the completion of physical connection, we check each and every polygon in the design, based on the design rules and reports all the violations. This whole process is called Design Rule Check.

Typical DRC rules are:

- Interior
- Exterior
- Enclosure
- Extension

Interior:

Fig1: Distance of interior facing edge for a single layer.

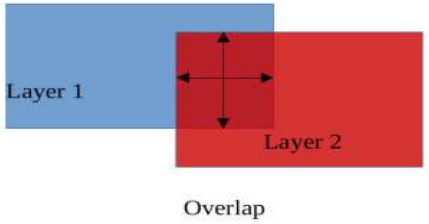


Fig2: Distance of interior facing edge of two layer.

Exterior:

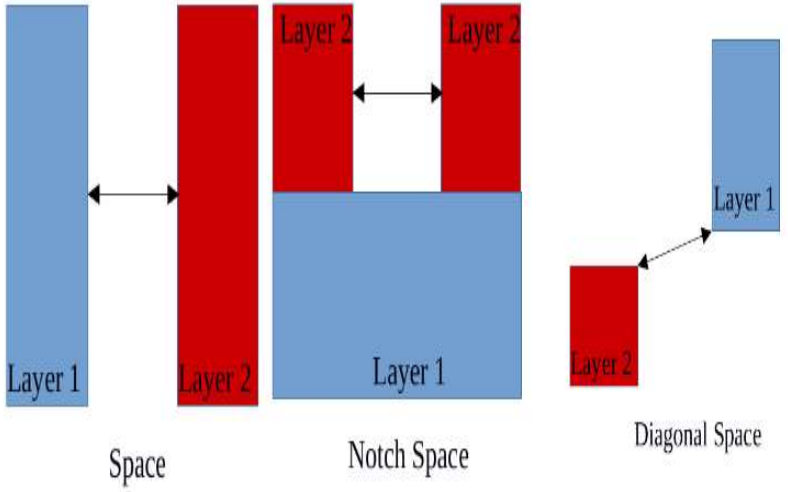


Fig3: Distance of exterior facing edge of two layer

Enclosure:

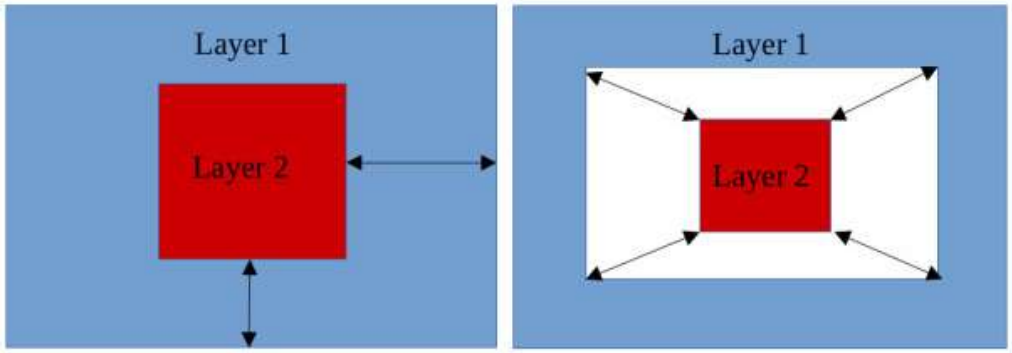


Fig4: Distance between inside edge to outside edge.

Extension:

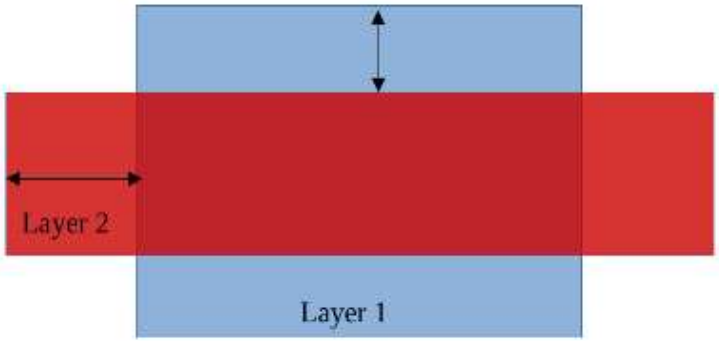


Fig4: Distance between inside edge to outside edge.

DRC's are two categories – Base Layer DRC's and Metal DRC's

Base Layer DRC's:

Base layer means all layers upto contact/metal 1. In Base layer, DRC flow rules will be checked:

- Base DRC's are spacing rules for geometries inside transistor (Well spacing, Poly spacing, Poly width)
- Tap cell requirement
- Well continuity (after routing fill empty space using spare cells)

Fixes:

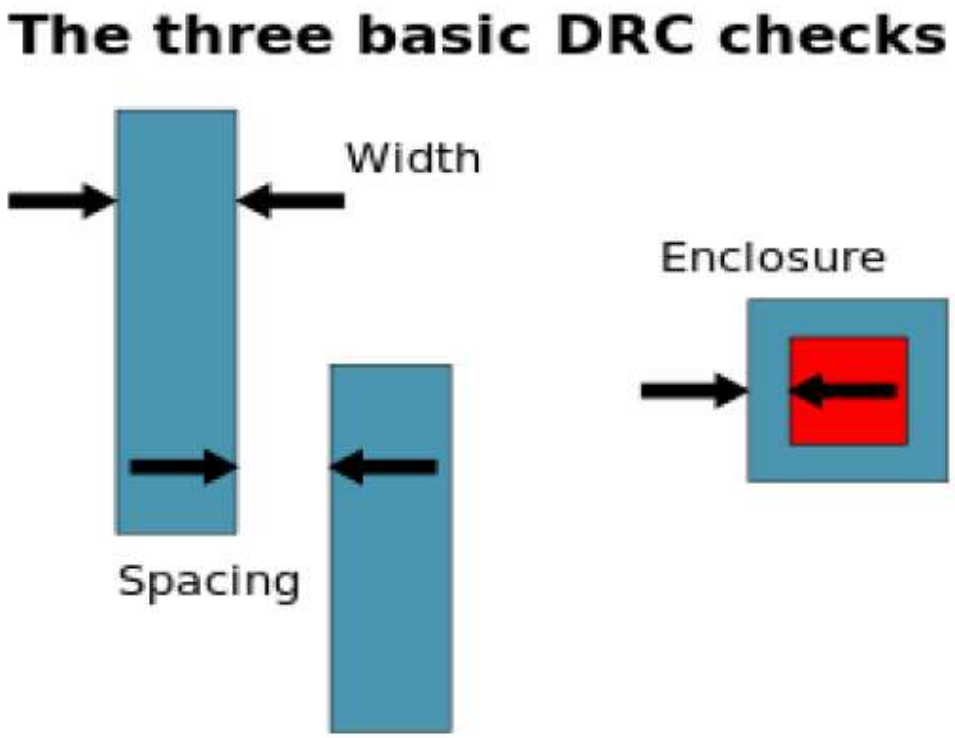
- Make sure design is placed legally
- No cell overlaps & no gaps

Most of the times, base DRC’s will be clean if we ensure above two conditions.

Metal DRC's:

Metal DRC means from contact to all the routing layers. Basic metal DRC's are:

- Width (min & max)
- Spacing (min)
- Via enclosure (size of min cut, min via to via spacing in multi cut Vias)



• **LVS**

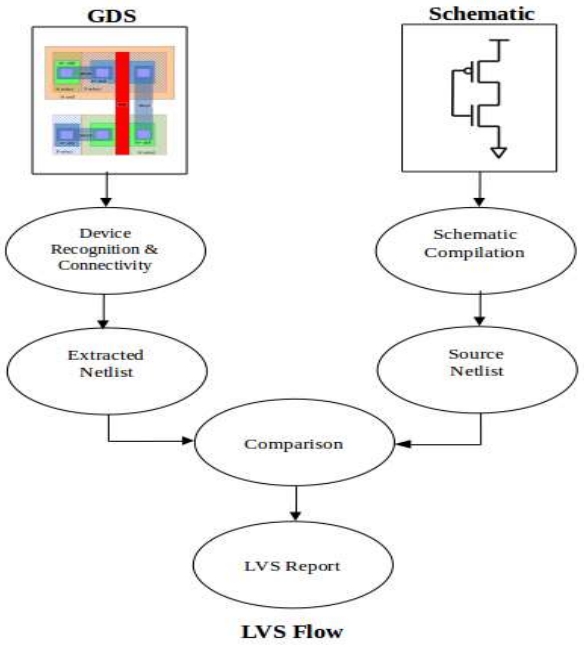
DRC only verifies that the given layout satisfies the design rules provided by the fabrication unit. It does not ensure the functionality of layout. Because of this, idea of LVS is originated. This blog focuses on how LVS works and what all are the common issues faced in LVS.

Inputs of LVS:

- Netlist

- GDS layout database of the design

- LVS rule deck



LVS Flow

LVS rule deck is a set of code written in Standard Verification Rule Format (SVRF) or TCL Verification Format (TVF). It guides the tool to extract the devices and the connectivity of IC's. It contains the layer definition to identify the layers used in layout file and to match it with the location of layer in GDS. It also contains device structure definitions.

Steps of LVS:

1. Extraction: The tool takes GDSII file containing all the layers and uses polygon based approach to determine the components like transistors, diodes, capacitors and resistors and also connectivity information between devices presented in the layout by their layers of construction. All the device layers, terminals of the devices, size of devices, nets, vias and the locations of pins are defined and given an unique identification.

2. Reduction: All the defined information is extracted in the form of netlist.

3. Comparison: The extracted layout netlist is then compared to the netlist of the same stage using the LVS rule deck. In this stage the number of instances, nets and ports are compared. All the mismatches such as shorts and opens, pin mismatch etc.. are reported. The tools also checks topology and size mismatch.

LVS Checks:

- No of devices in schematic and its layout

- Type of devices in schematic and its layout

- No of nets in schematic and its layout

Errors occur:

1. Shorts: Shorts are formed, if two or more wires which should not be connected together are connected.

2. Opens: Opens are formed, if the wires or components which should be connected together are left floating or partially connected.

3. Component mismatch: Component mismatch can happen, if components of different types are used (e.g, LVT cells instead of HVT cells).

4. Missing components: Component missing can happen, if an expected component is left out from the layout.

5. Parameter mismatch: All components has it's own properties, LVS tool is configured to compare these properties with some tolerance. If this tolerance is not met, then it will give parameter mismatch.

Important Questions:

--> Why Setup is checked at next edge and Hold is checked at present edge?

Ans: Data should be stable Ts time before the posedge of FF2/C

If Ts = 0 ns, then, data launched from FF1 at time = 0ns should arrive at D of FF2 before or at time = 10ns. If data takes too long to arrive, it is reported as setup violation.

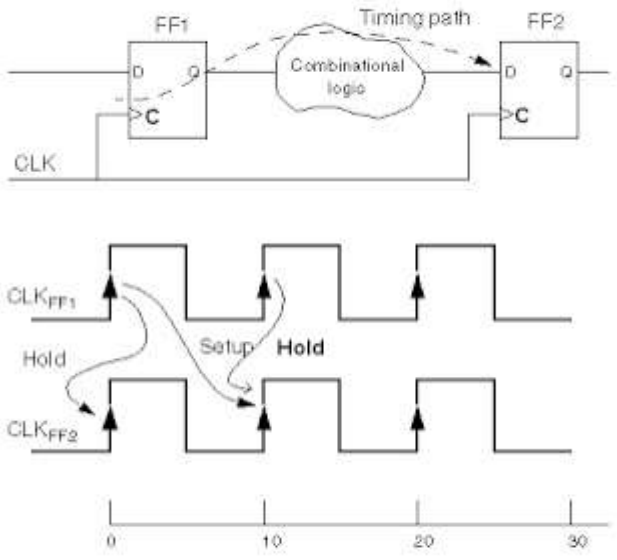
If Ts = 1ns, then, data launched from FF1 at time = 0ns should arrive at D of FF2 before or at time = 10 – 1 = 9ns. If data takes too long to arrive, it is reported as setup violation

Data should be stable Th time after the posdge at FF2/C.

To satify the hold condition at FF2 for the data launched by FF1 at 0ns, the data launched at FF1 at 10ns should not reach at FF2/D before 10ns+Th time.

If Th = 0.5ns, then we can say that the data launched from FF1 at time 10 ns doesn't propagate so soon that it reaches at FF2 before time 10.5 ns.

With the above explanation, we can say Setup is checked at next clock edge and Hold is checked at same clock edge.



--> Why Setup is checked at max corner and Hold at min corner?

Ans: For setup, the required time should be more than the arrival time. And setup violates when arrival time is more. So, setup check is more pessimistic when arrival time is more or when the launch clock reaches late than the capture clock. That means delay is more. So, setup will be checked at max delays.

For hold, the arrival time should be more than the required time. And hold violates when required time is more. So, hold check is more pessimistic when required time is more or when the launch clock reaches early than the capture clock. That means data arrival time is less. So, hold will be checked at min delays.

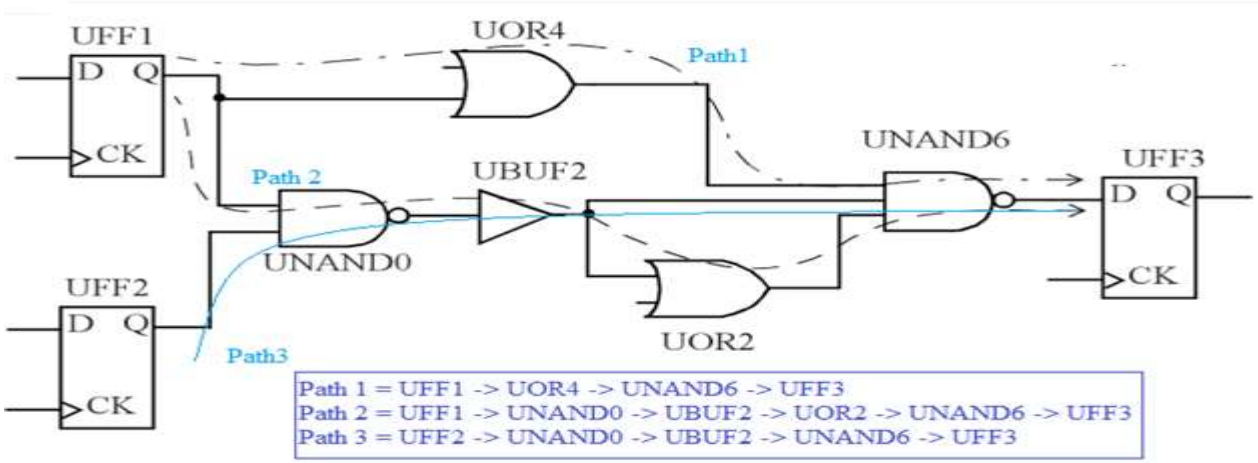
→ Why we are not checking the hold before CTS?

Ans: Before CTS, clock is ideal that means exact skew is not there. All the clocks reaching the flops at the same time. So, we dont have skew and transition numbers of the clock path, but this information is sufficient to perform setup analysis since setup violation depends on the data path delay. Clock is propagted only after CTS (actual clock tree is built, clock buffers are added & clock tree hierarchy, clock skew, insertion delay comes into picture) and that's why hold violations are fixed only after CTS.

→ Can both Setup and Hold violations occur in same start and end points? If yes, how you will resolve?

Ans: Yes, if they have different combo paths. Consider, the below scenario.

Violation Paths		For Fixing the Hold Violation	
Setup	Hold	Dos	DON'Ts
Path 1	Path2	Add delay anywhere in path2 between UNAND0 and UNAND6	Don't add delay between UNAND6 and UFF3 (Common section of both paths)
Path 1	Path3	Add delay anywhere between UFF2 and UNAND6	Don't add delay between UNAND6 and UFF3 (Common section of both paths)
Path 2	Path3	Add delay before UNAND0 in path3. Can add delay in path3 between UBUF2 and UNAND6)	Don't add delay between UNAND0 and UNAND6 (Common section of both paths)



Thank You