

Topic 4

ASIC Design Flow II

Xinfei Guo
xinfei.guo@sjtu.edu.cn

October 30th, 2024



Formal Verification

Are they equal?

Verification in the Design Flow

Verification is very important!!!

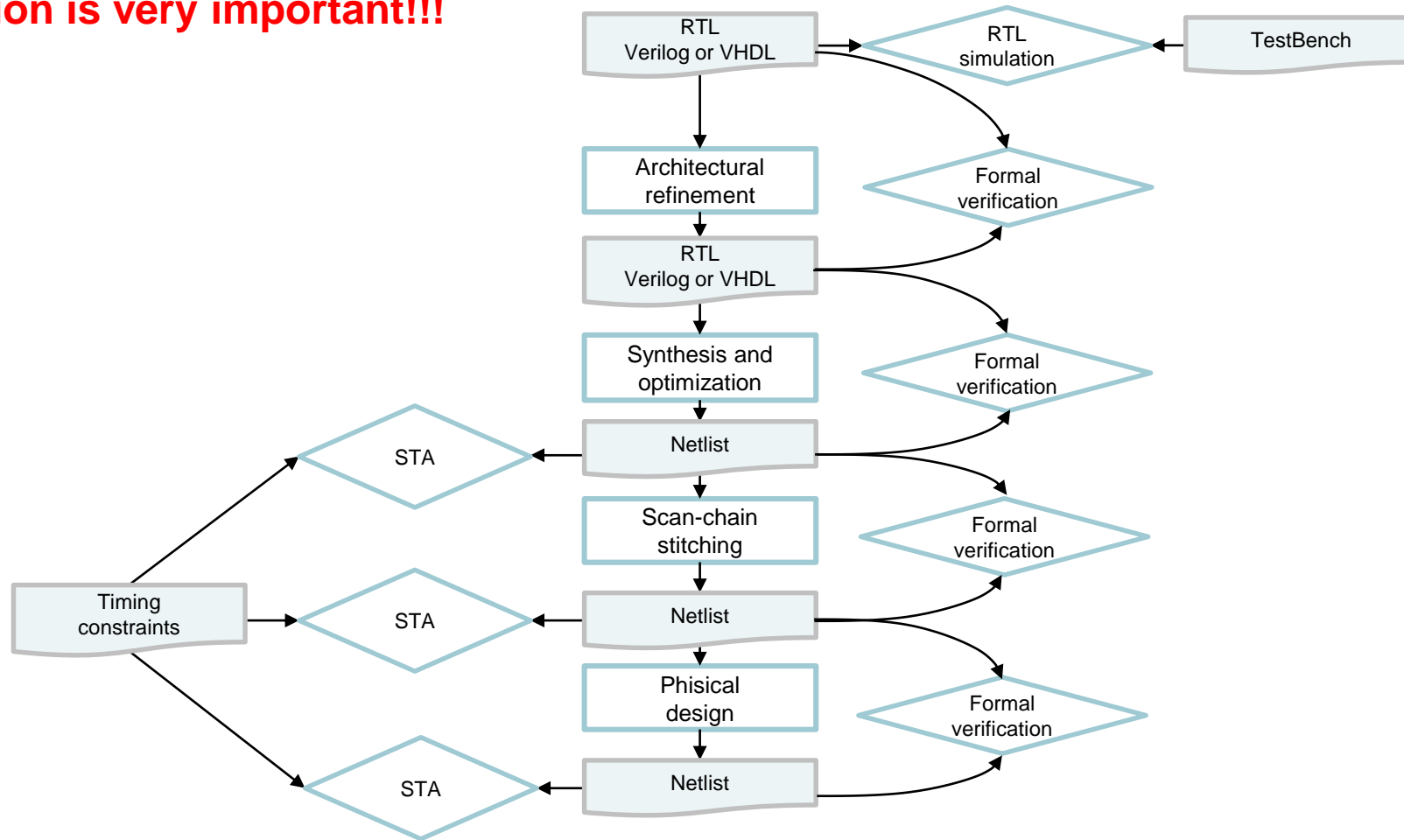


Figure: Synopsys

Traditional IC Simulation

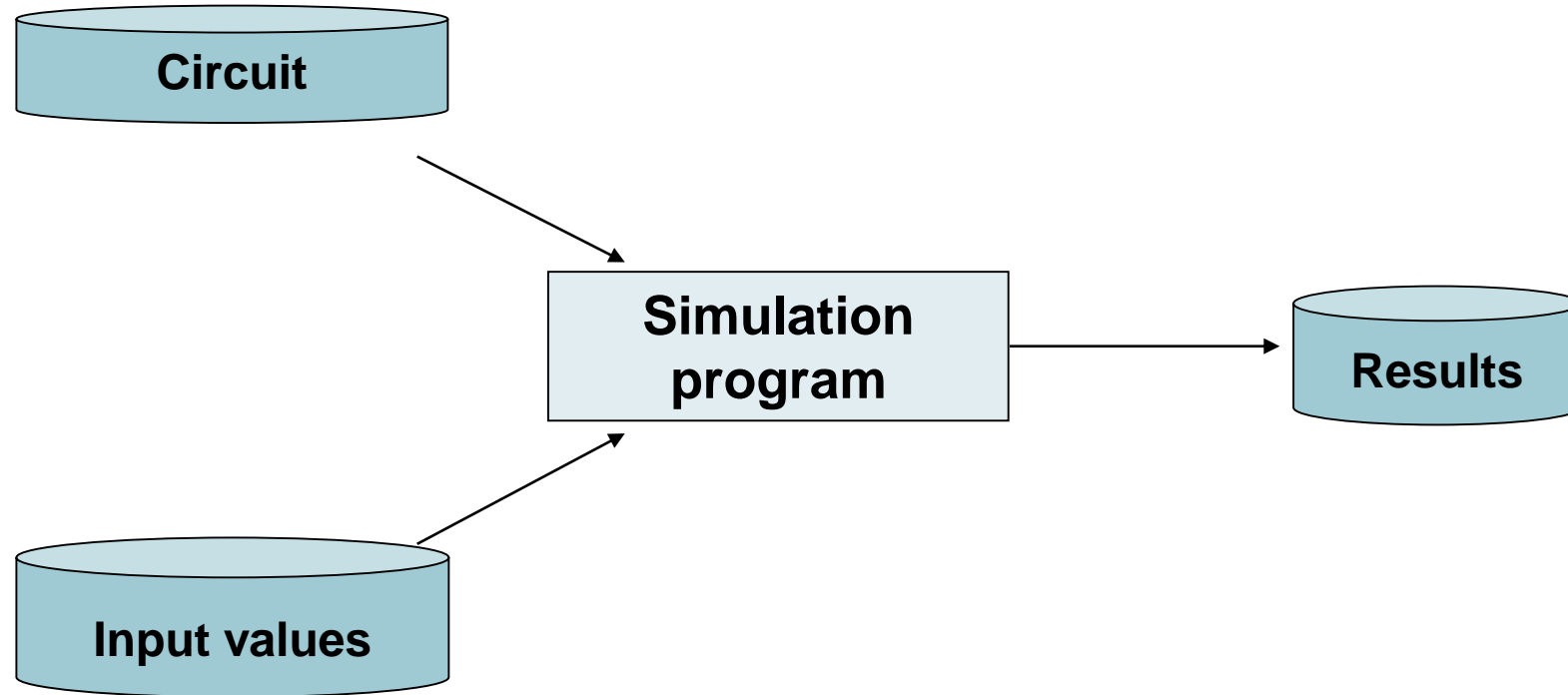


Figure: Synopsys

Formal Verification

- Formal verification checks whether two designs are functionally equivalent or not
- Its purpose is to detect unexpected differences that may have been introduced into a design during development

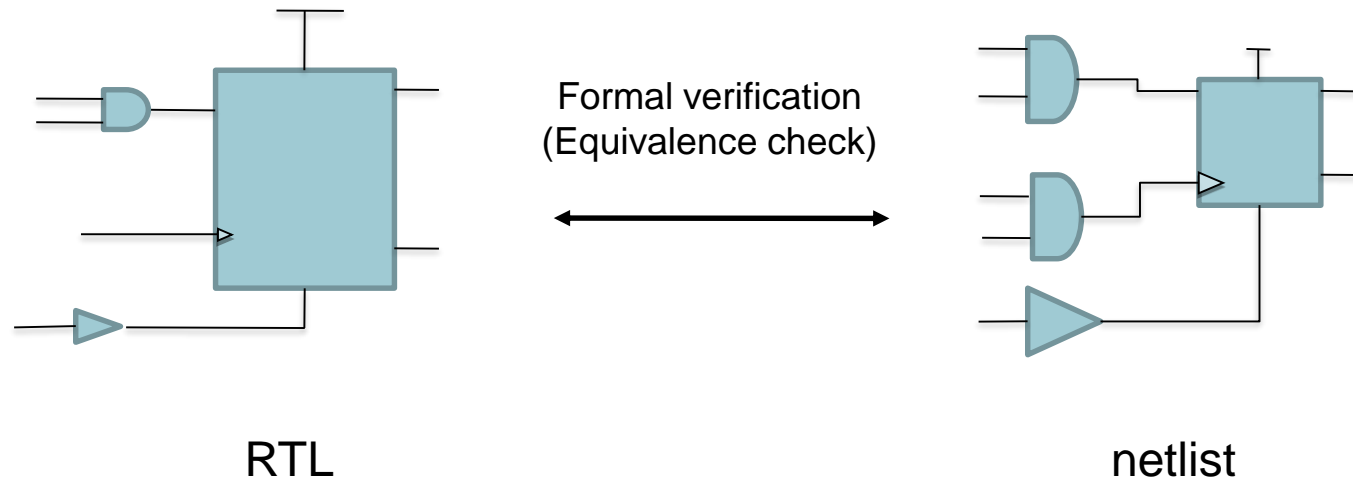


Figure: Synopsys

Formal Verification

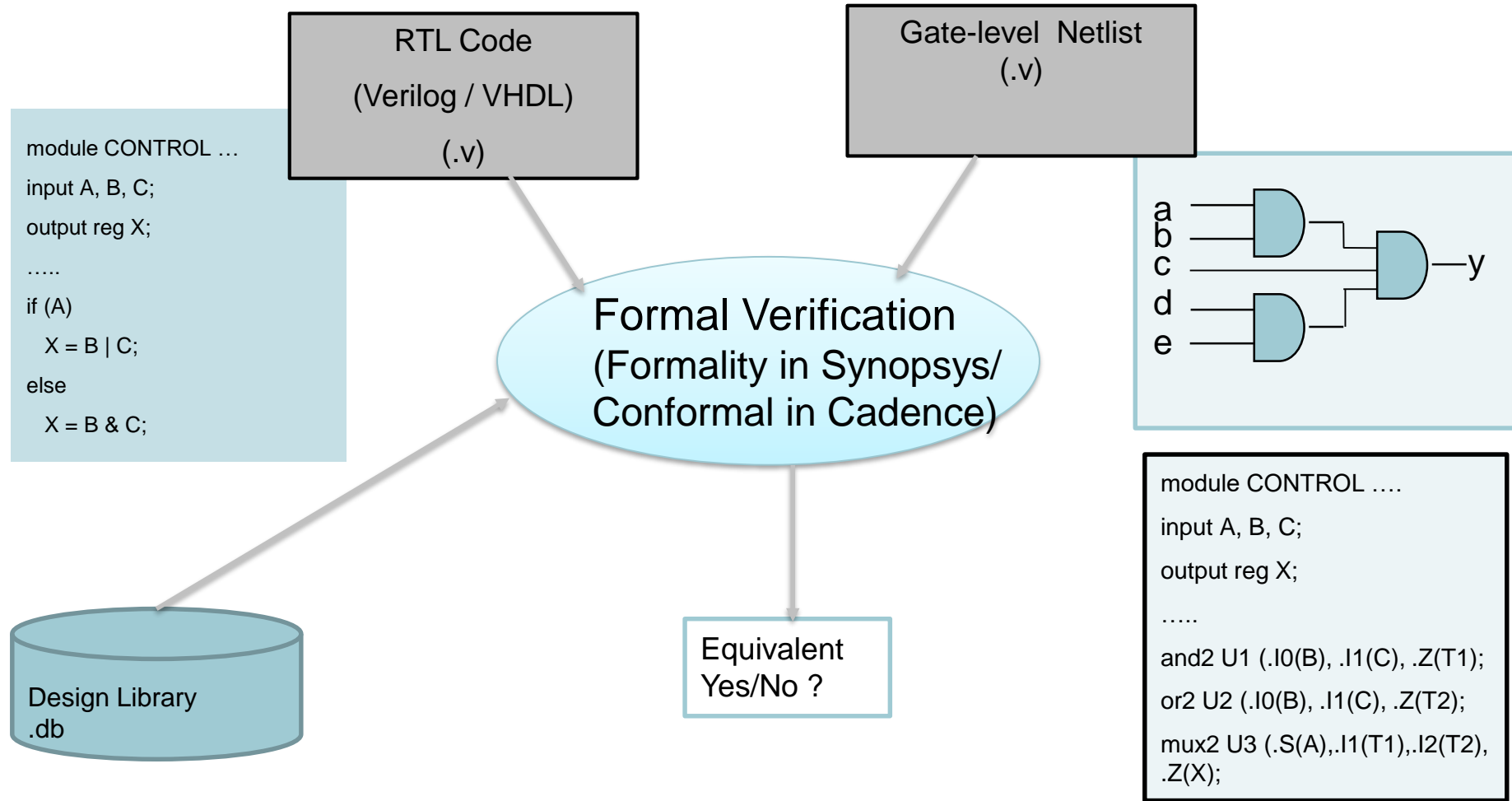
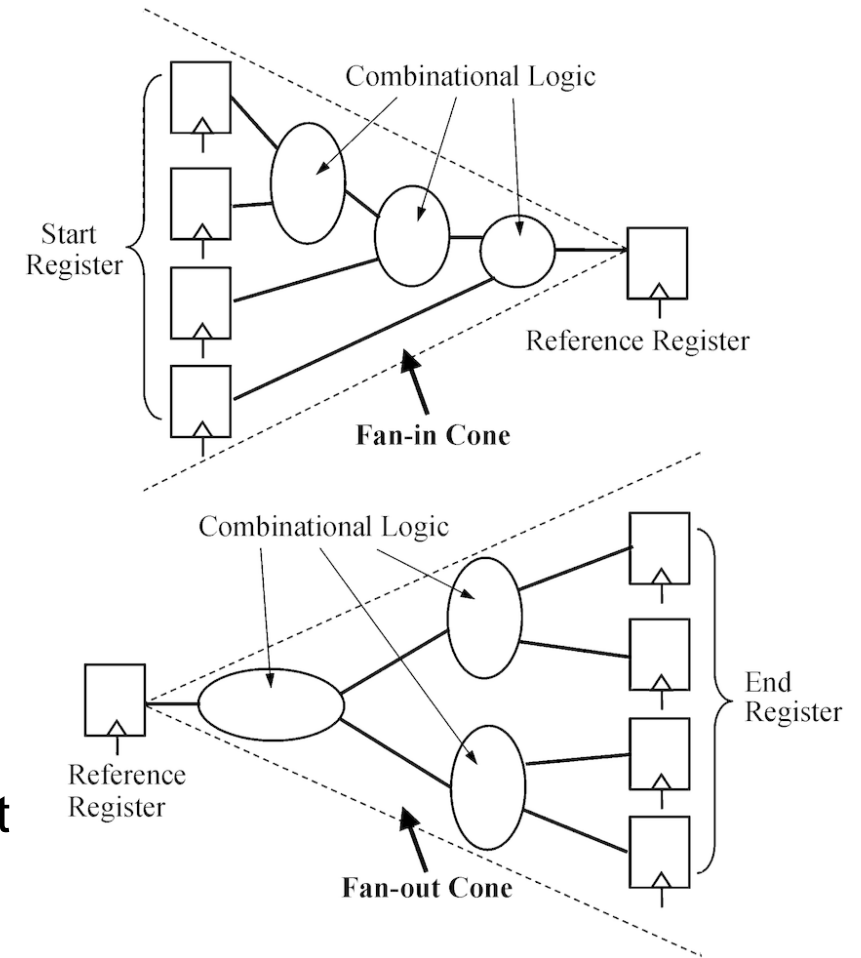


Figure: Synopsys

Key Concepts

- Main concepts in Formality are
 - Compare Point
 - Primary output of a circuit
 - Registers within a circuit
 - Input to black boxes within a circuit
 - Logic Cone
 - A block of combinational logic which drives a compare point
 - Benefits
 - No need to run simulation.
 - Functionality checks can be done by taking a netlist after any stage.
 - Bugs can be easily identified.



Equivalent Checking Verification Process

Equivalence checking is a four-phase process

- Reading and elaborating language descriptions into logical representations
- Setting up prompt for verification
- Mapping of corresponding compare points between pair of designs (matching)
- Comparison of logic cones that drive the compare points (verification)

Example: Input Files of Formality (Synopsys)

Formality supports following input formats:

Input formats	Options
Verilog (synthesizable subset)	- read_verilog
Verilog (simulation libraries)	- read_verilog -vcs
VHDL (synthesizable subset)	- read_vhdl
EDIF	- read_edif
Synopsys binary files	- read_db, read_ddc, read_mdb (*)

Formality Flow Overview

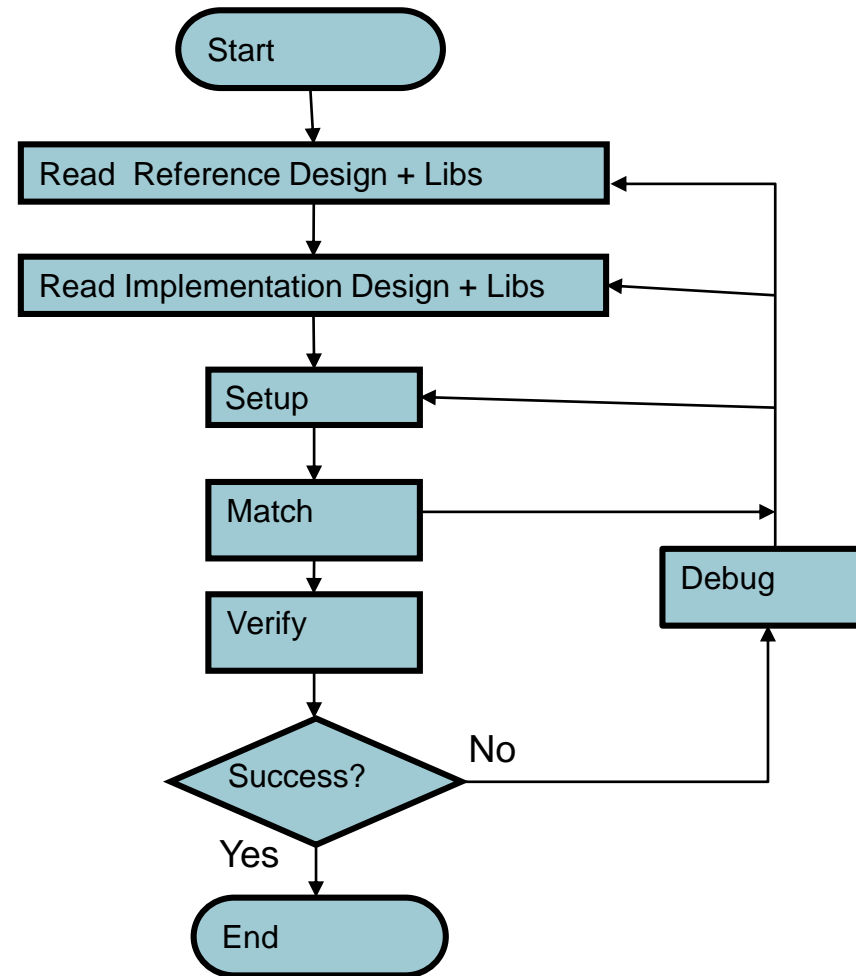


Figure: Synopsys

Match (Matching Compare Points)

- Process of aligning compare points between two designs

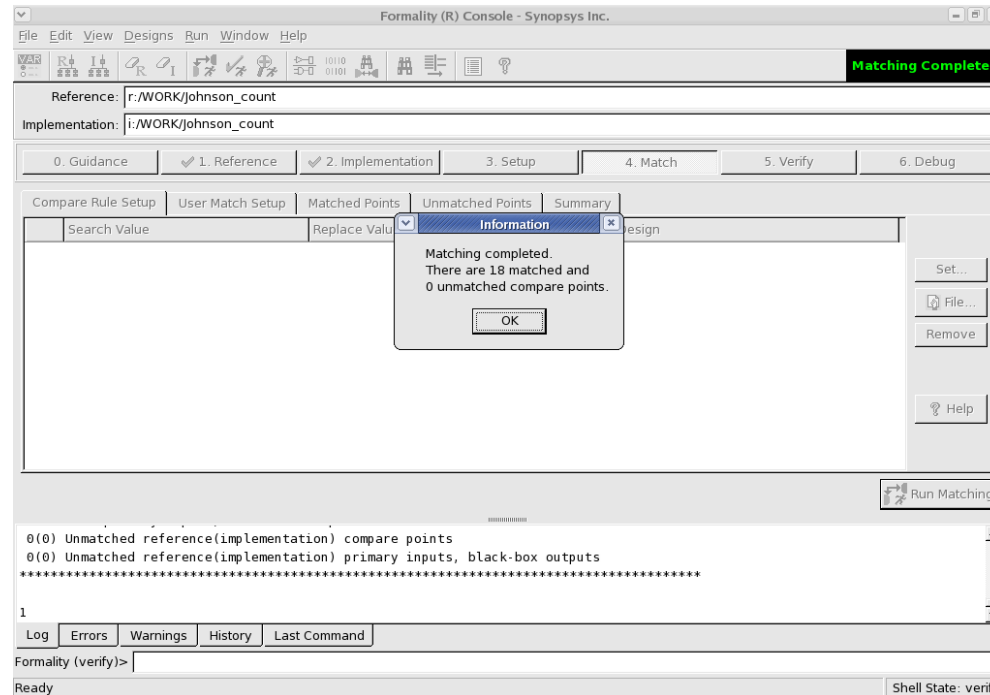


Figure: Synopsys

Debug

- Debugging is part of the process where verification results are used to pinpoint either failing or inconclusive results. During the debug step the user may determine where and possibly why the results were unsuccessful

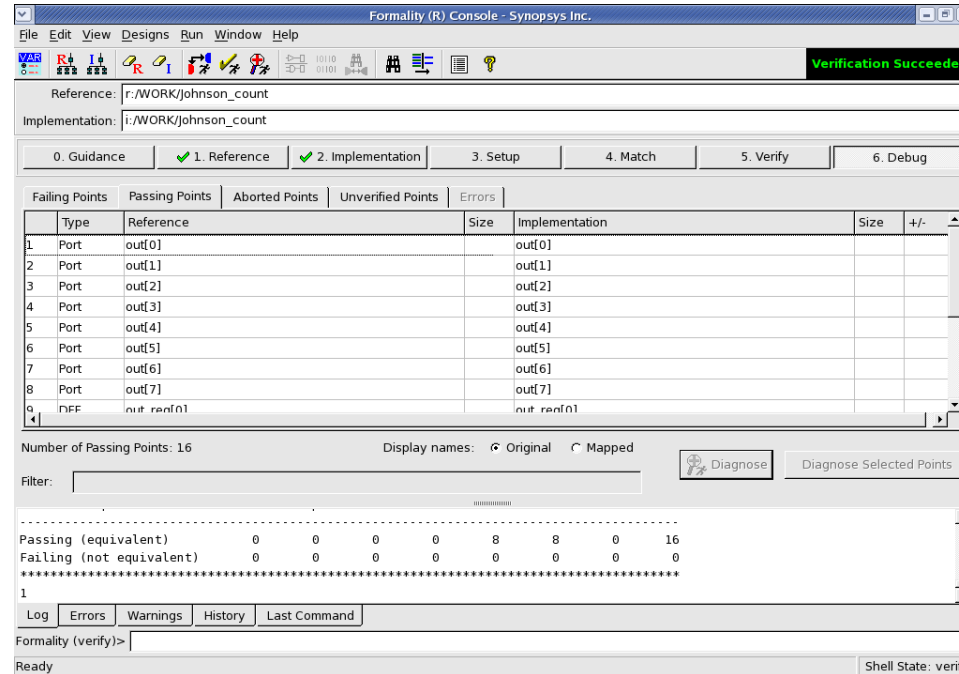
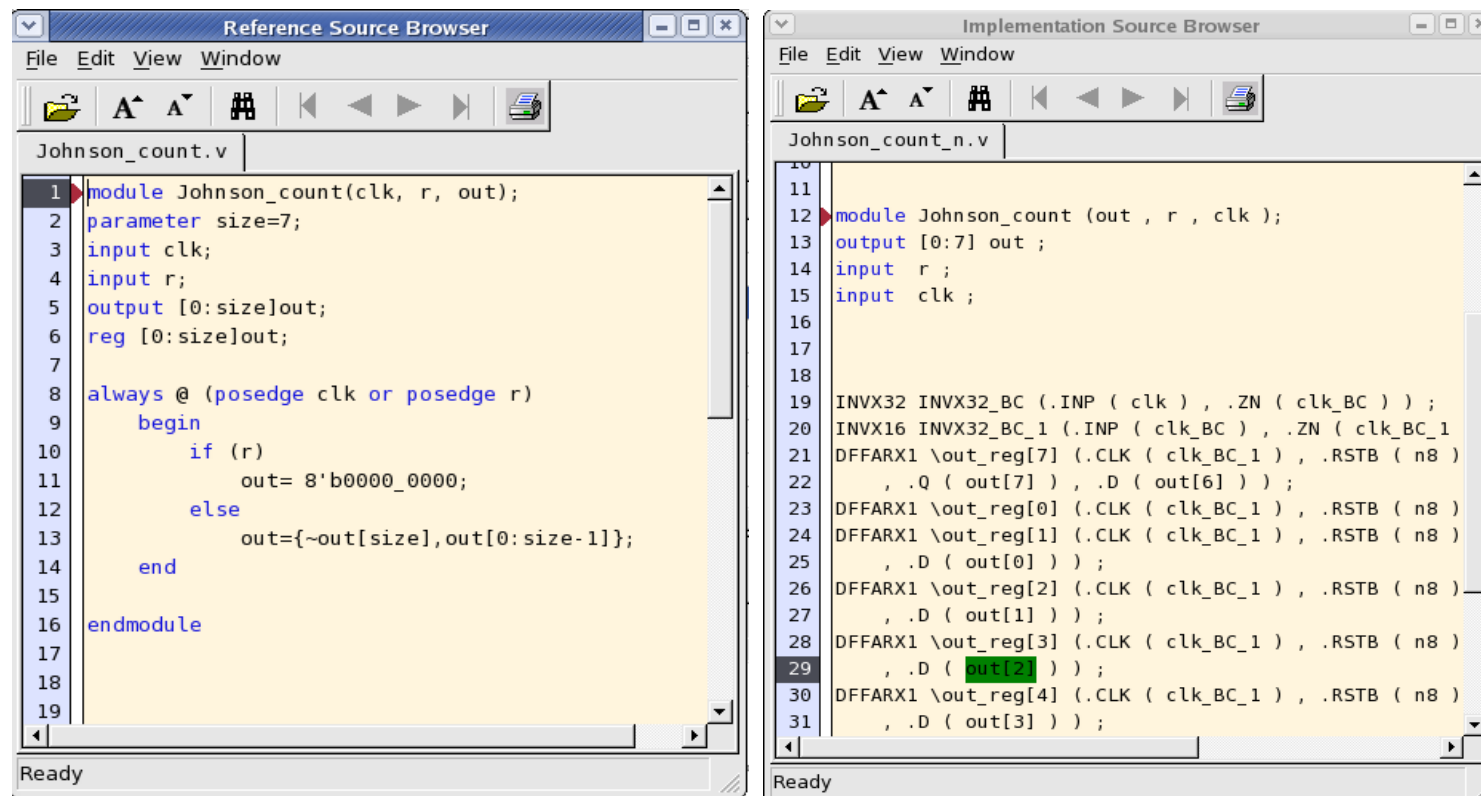


Figure: Synopsys

Debug (2)



The image displays two side-by-side windows from the Synopsys IDE, showing Verilog code for a Johnson counter.

Left Window: Reference Source Browser
File Edit View Window
Johnson_count.v
1 module Johnson_count(clk, r, out);
2 parameter size=7;
3 input clk;
4 input r;
5 output [0:size]out;
6 reg [0:size]out;
7
8 always @ (posedge clk or posedge r)
9 begin
10 if (r)
11 out= 8'b0000_0000;
12 else
13 out={~out[size],out[0:size-1]};
14 end
15 endmodule
16
17
18
19
Ready

Right Window: Implementation Source Browser
File Edit View Window
Johnson_count_n.v
10
11
12 module Johnson_count (out , r , clk);
13 output [0:7] out ;
14 input r ;
15 input clk ;
16
17
18
19 INVX32 INVX32_BC (.INP (clk) , .ZN (clk_BC)) ;
20 INVX16 INVX32_BC_1 (.INP (clk_BC) , .ZN (clk_BC_1)) ;
21 DFFARX1 \out_reg[7] (.CLK (clk_BC_1) , .RSTB (n8)
22 , .Q (out[7]) , .D (out[6])) ;
23 DFFARX1 \out_reg[0] (.CLK (clk_BC_1) , .RSTB (n8)
24 DFFARX1 \out_reg[1] (.CLK (clk_BC_1) , .RSTB (n8)
25 , .D (out[0])) ;
26 DFFARX1 \out_reg[2] (.CLK (clk_BC_1) , .RSTB (n8)
27 , .D (out[1])) ;
28 DFFARX1 \out_reg[3] (.CLK (clk_BC_1) , .RSTB (n8)
29 , .D (out[2])) ;
30 DFFARX1 \out_reg[4] (.CLK (clk_BC_1) , .RSTB (n8)
31 , .D (out[3])) ;
Ready

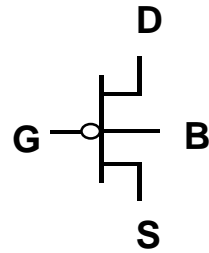
Figure: Synopsys

Floorplanning

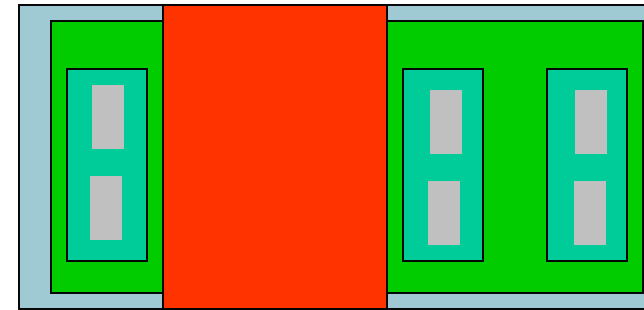
How should we place macros? Where should we place?

Concepts of the Circuit and Layout

Circuit



Layout



Resulting structure in manufactured IC

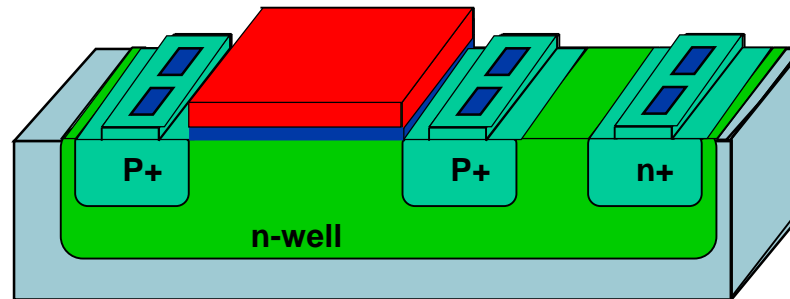
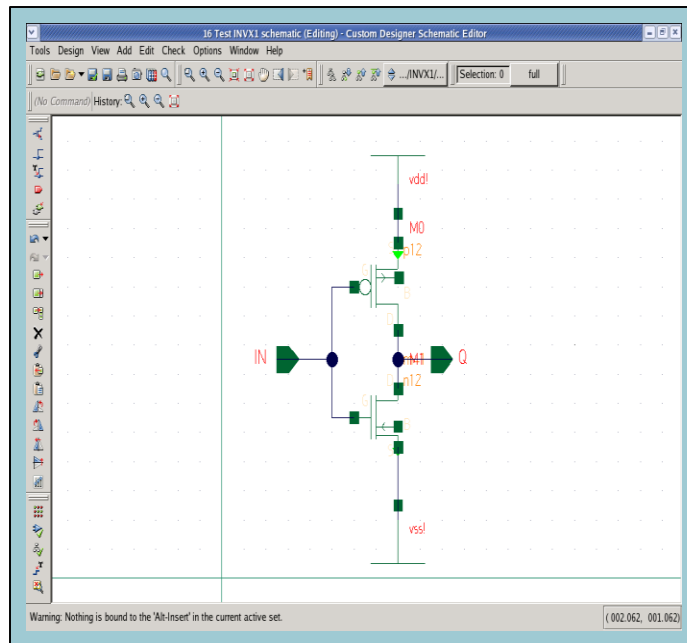
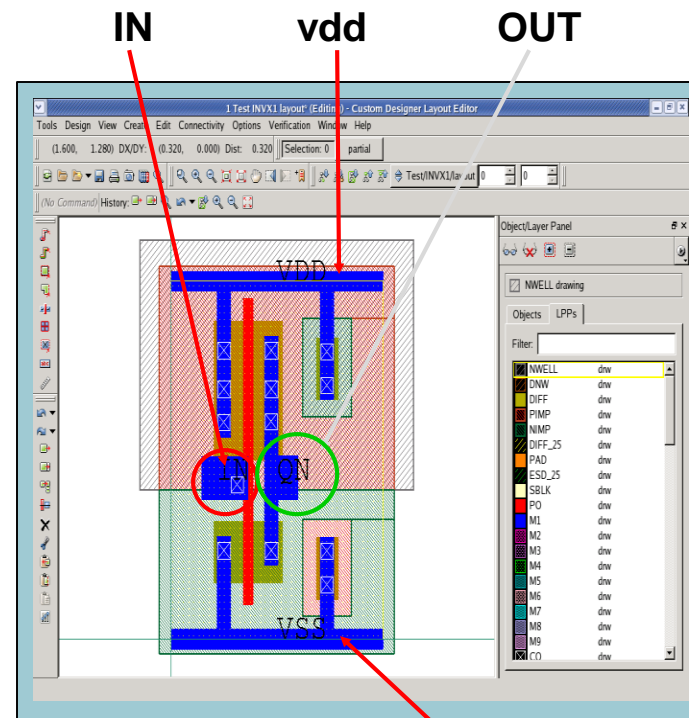


Figure: Synopsys

Circuit and Layout Editors



Circuit

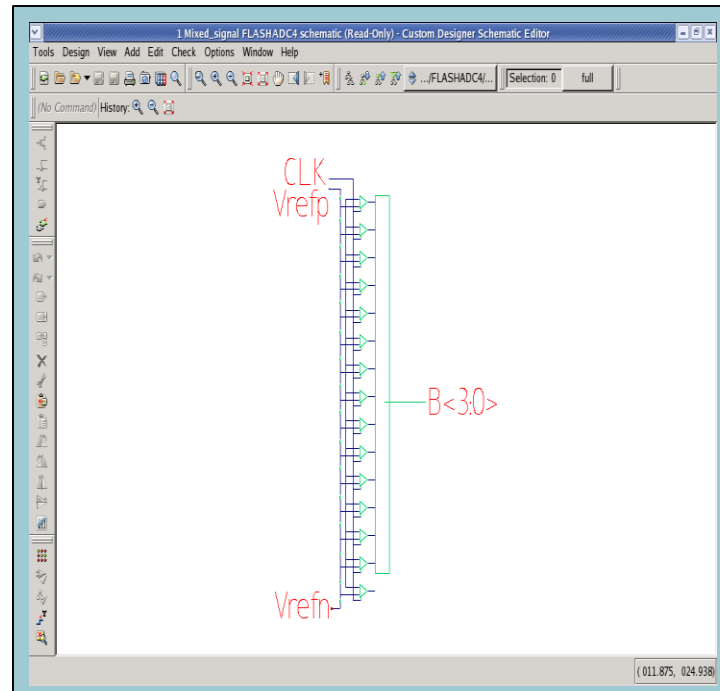


Layout

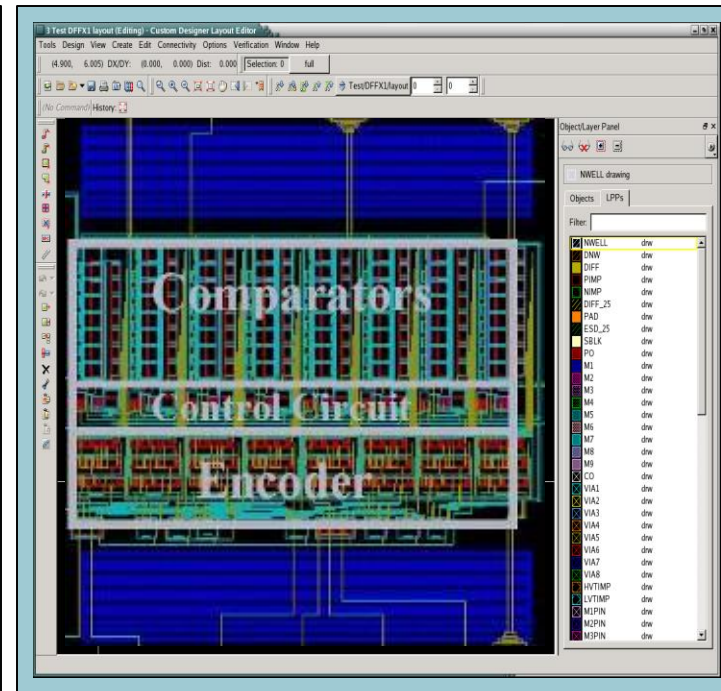
Figure: Synopsys

IP Example

FLASHADC4



Circuit



Layout

Figure: Synopsys

Physical Synthesis

Physical synthesis is the process that produces layout of logic circuit.

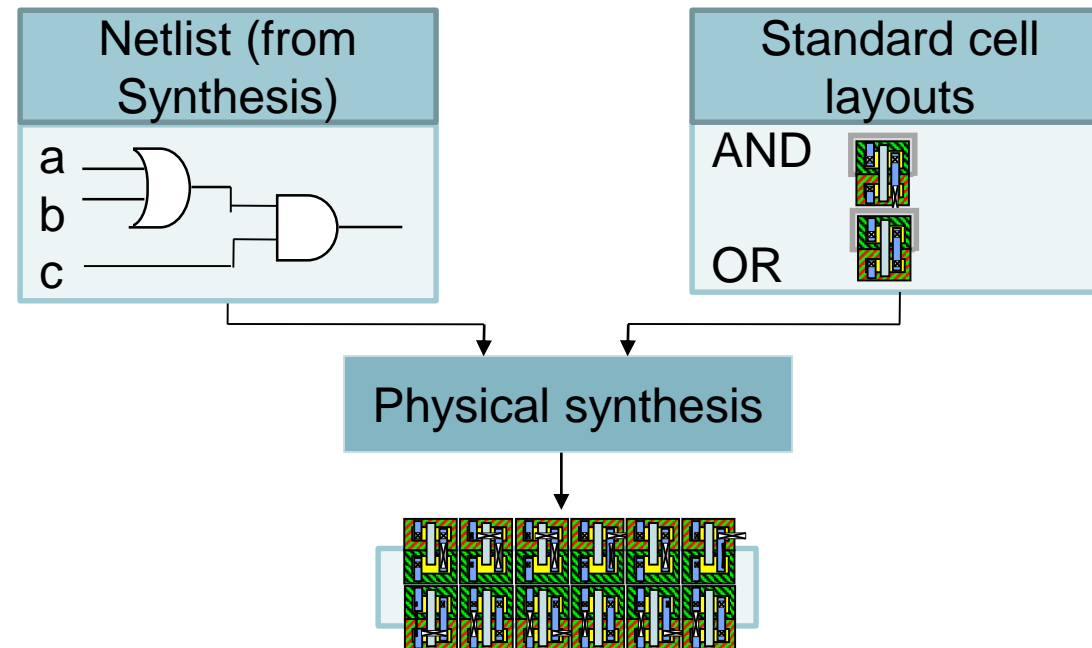


Figure: Synopsys

Physical Synthesis Flow

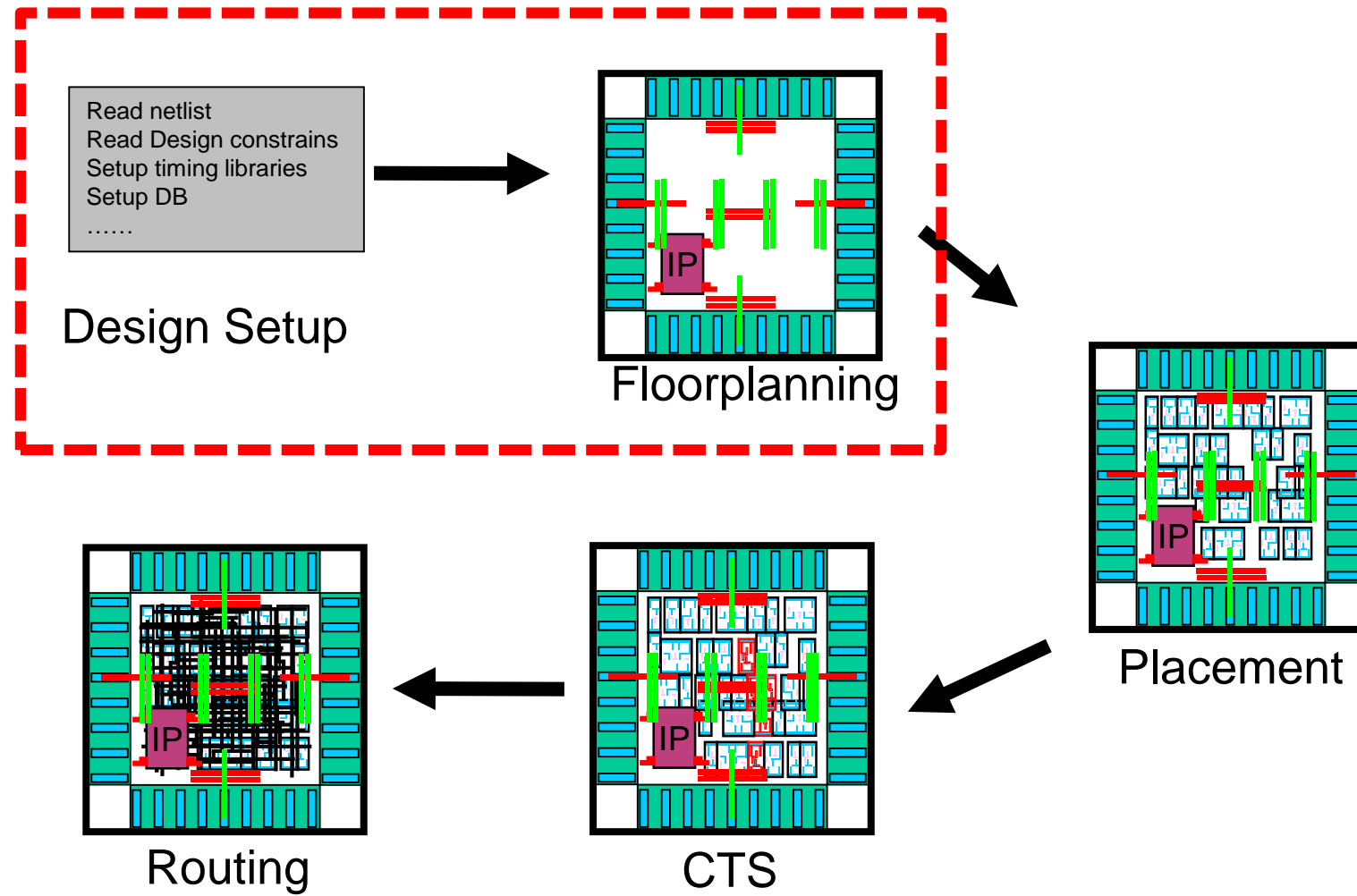
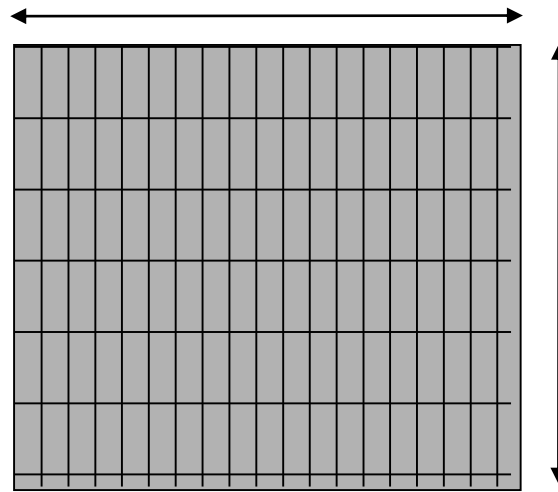


Figure: Synopsys

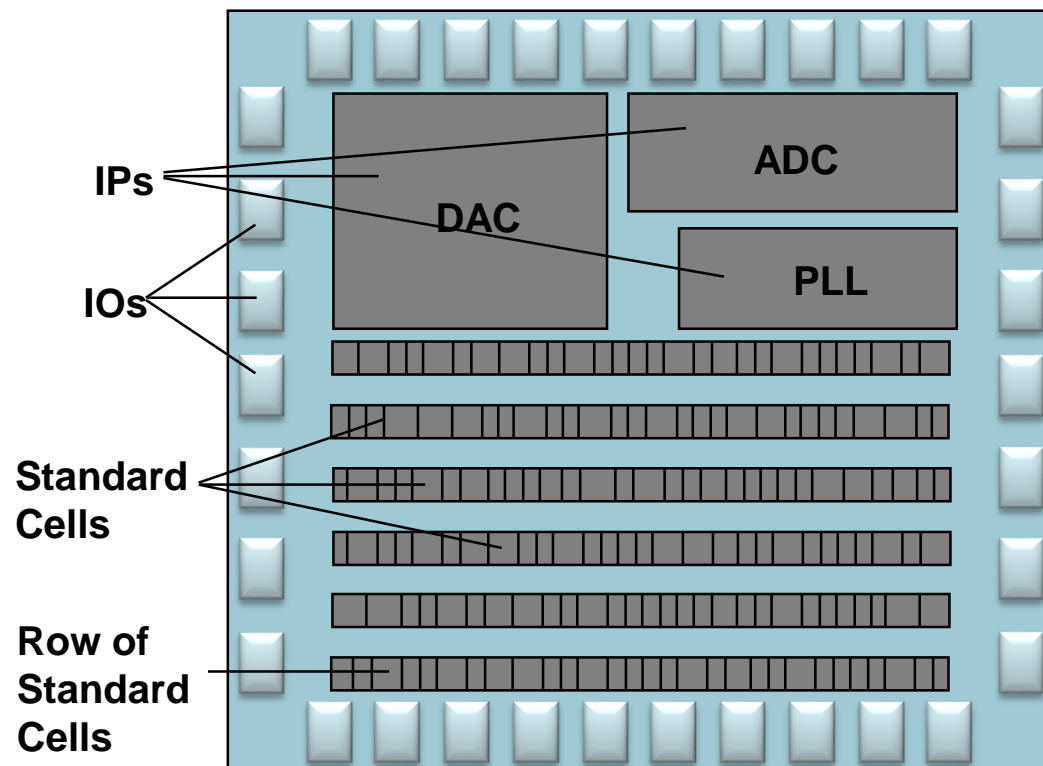
Floorplanning

- Not needed in FPGA flow
- During the floorplanning step the overall cell is defined, including: cell size, supply network, etc.



Floorplan

Floorplanning



- What are done in floorplanning?
 - IO Placement
 - Die Size and Aspect Ratio
 - Special Cell Pre-placemnt
 - IP/Macro pre-placement
 - Power grid generation
 - Blockage definition
 - Standard cells are **NOT** placed yet in this step.

Data Setup for Floorplan

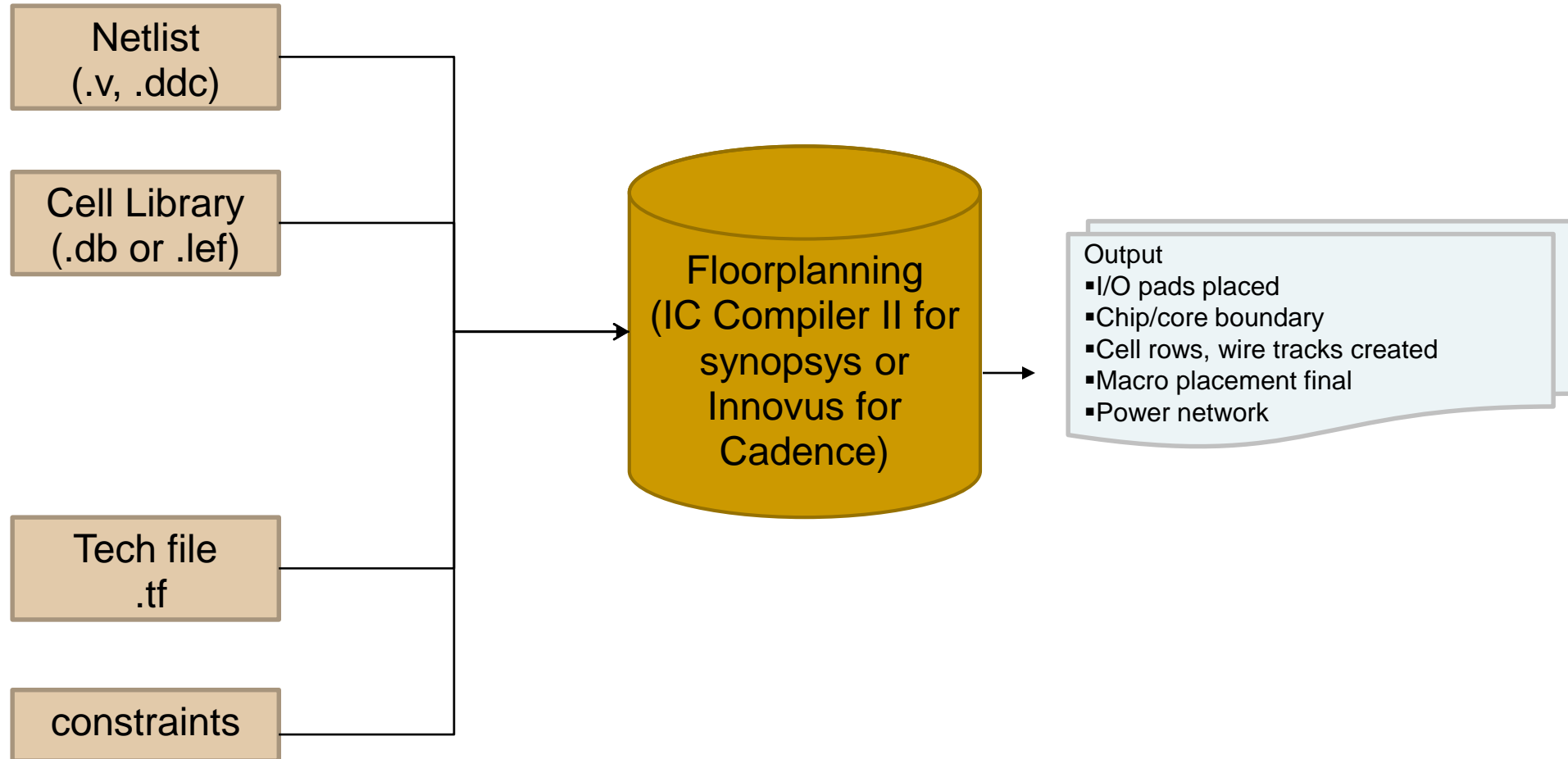
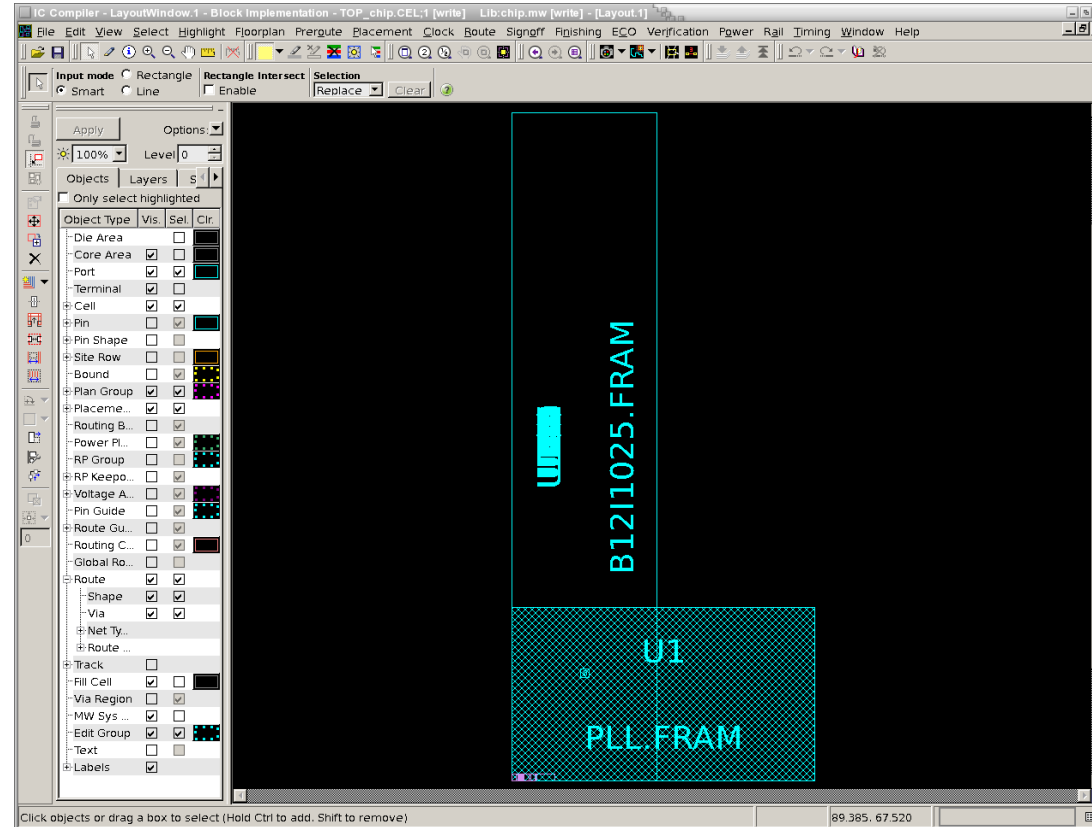


Figure: Synopsys

Design Importing

- Example: Design view from ICC after importing the design into ICC

IC Compiler



Hard Macro

Figure: Synopsys

Initialize Floorplanning

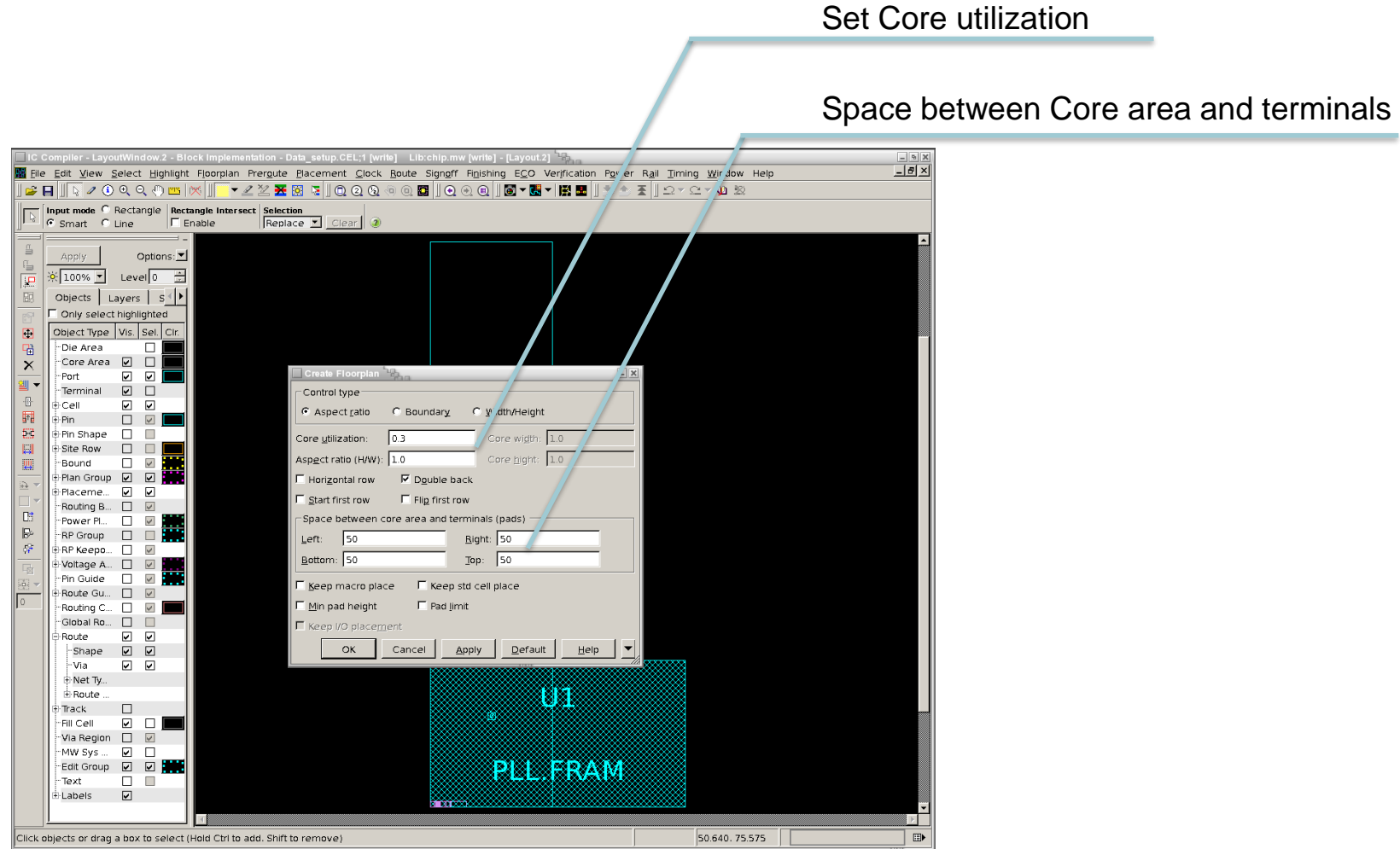


Figure: Synopsys

Initialize the Floorplan

- Generates the basic elements of the FP
 - Place IO pads/pins
 - Create chip/core boundary
 - Create rows and tracks honoring user defined values
 - aspect ratio/width & height/row number boundary
 - core utilization
 - ...

`initialize_floorplan`

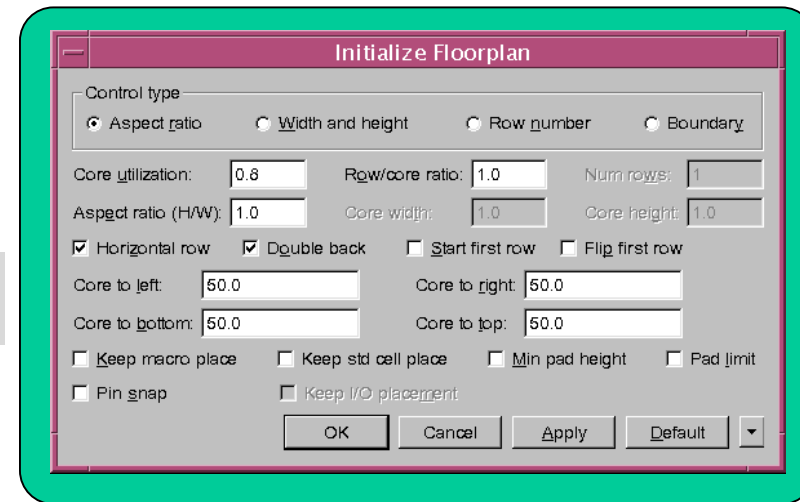


Figure: Synopsys

What is a row?

- Cells are placed in rows, next to each other
- One cells structure continue previous one
- Cells on neighbor rows are flipped so that they can share same supply

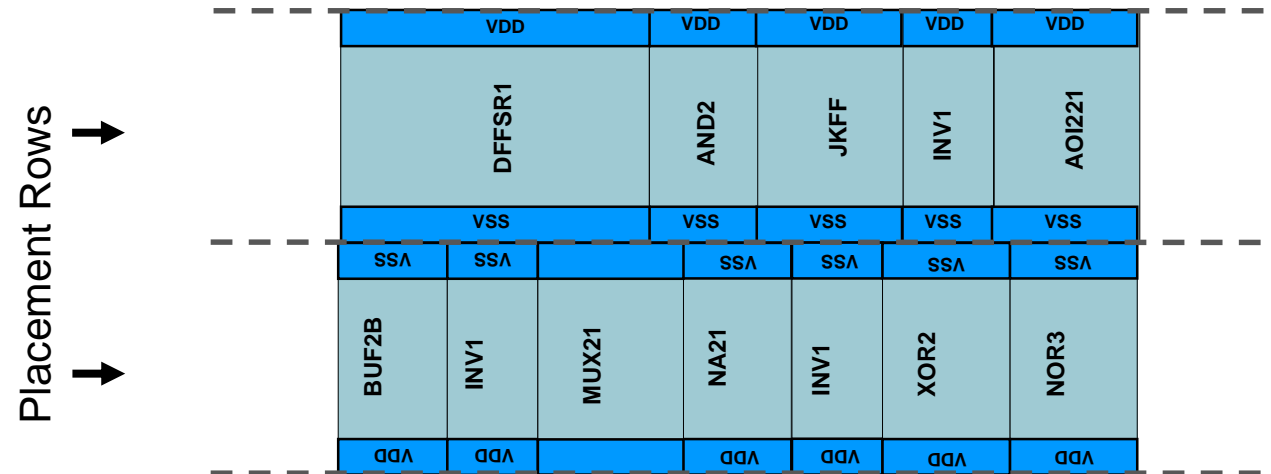
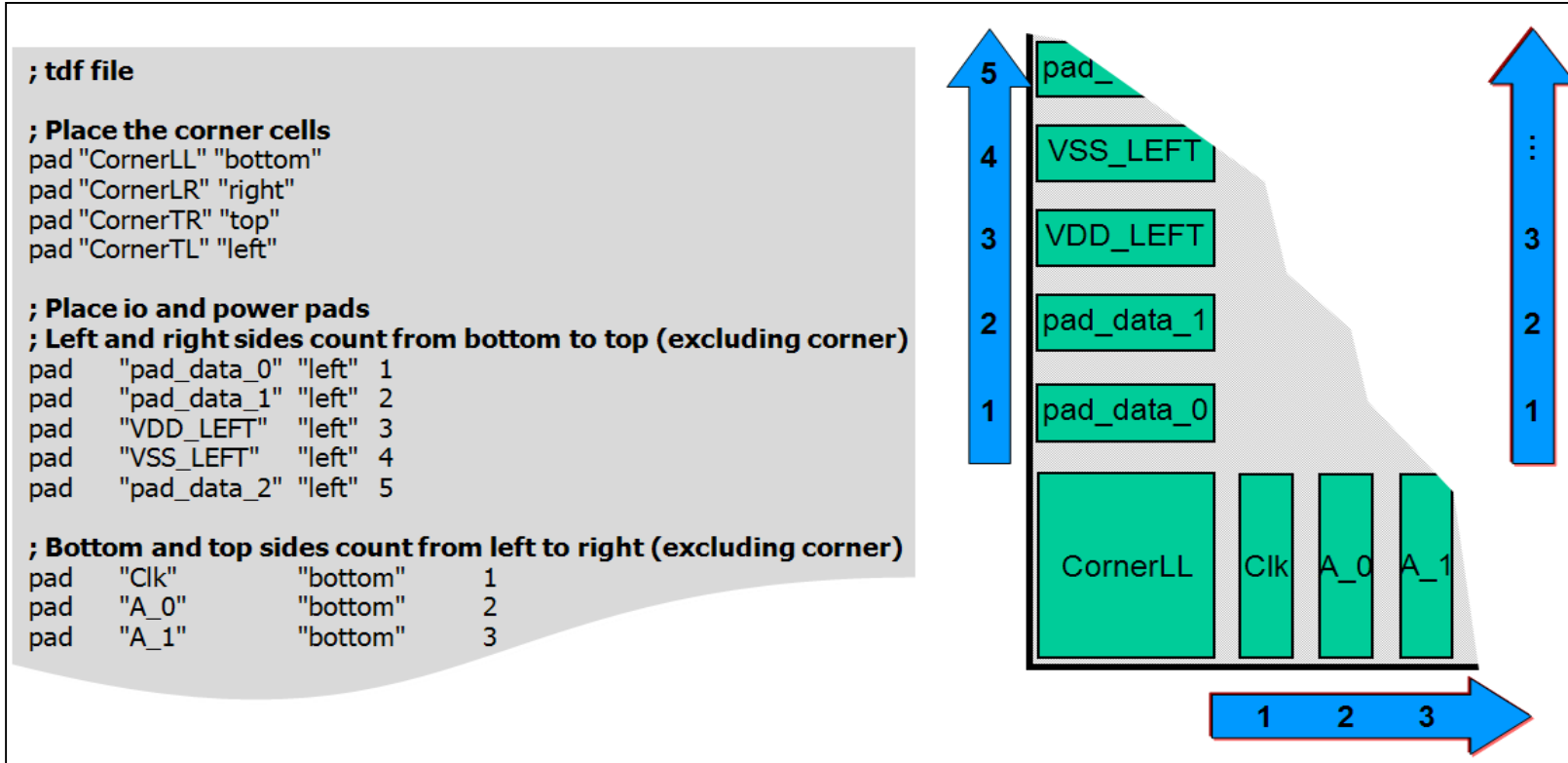


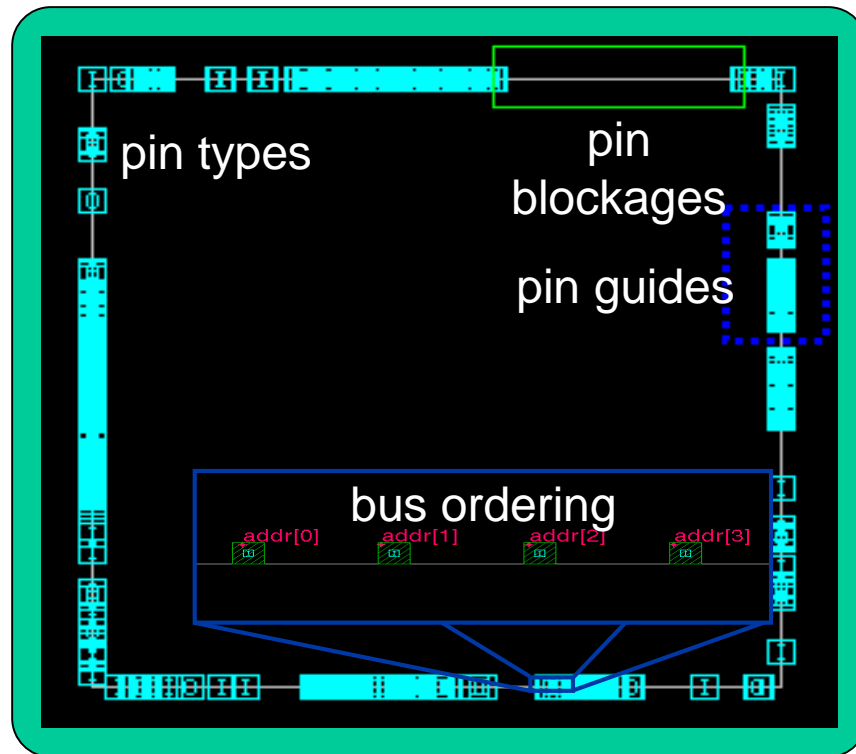
Figure: Synopsys

I/O Order Assignments



By default, ports are evenly distributed on each side

Optimal Pin Placements For Block Level placement



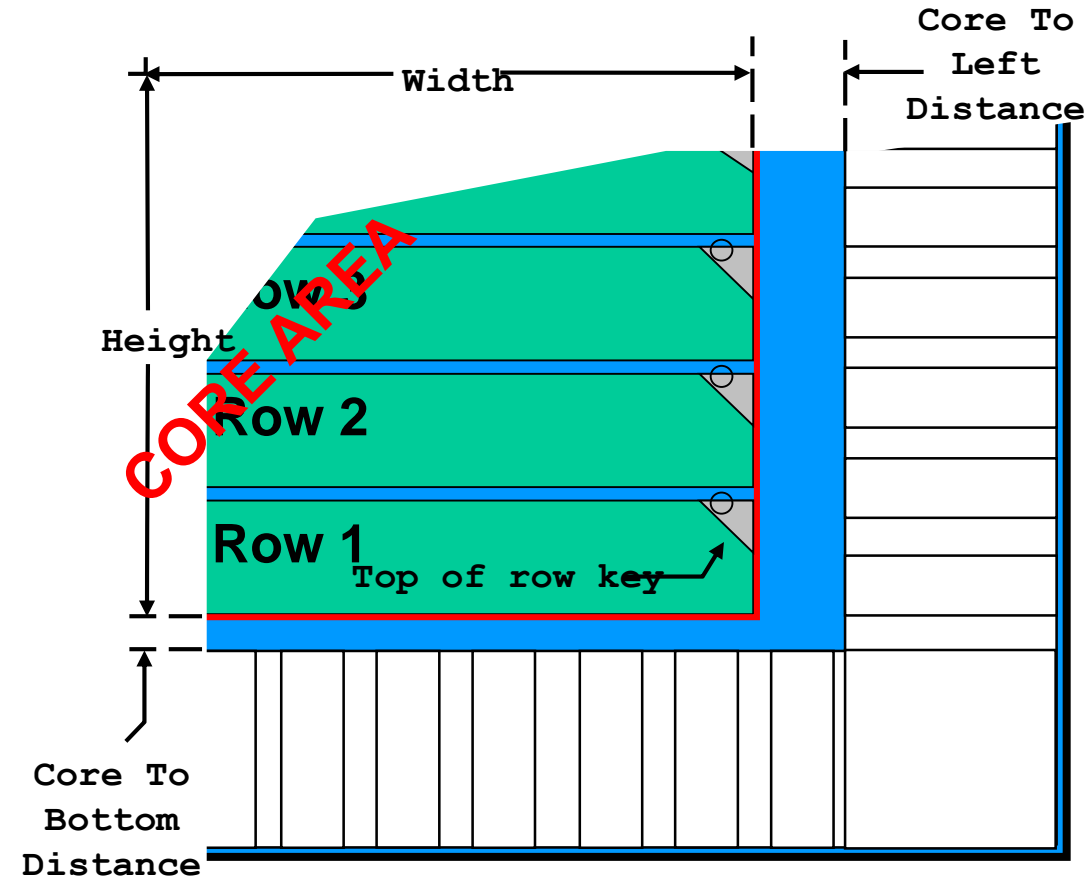
- Rectilinear and rectangular blocks
- Define relative pin locations
 - Drives cell placement
- Or, place cells first
 - Drives pin placement
- Define pin constraints
 - blockages, guides, bus ordering, layers...

Figure: Synopsys

Core Area

Control Parameters

- Aspect Ratio
 - Utilization
 - Aspect ratio (H/W)
 - Row/core ratio
- Width & Height
 - Width
 - Height
 - Row/core ratio
-
-



Example of a horizontal, no double back, no-flip first row with Row/Core < 1

Some operations done in Floorplanning

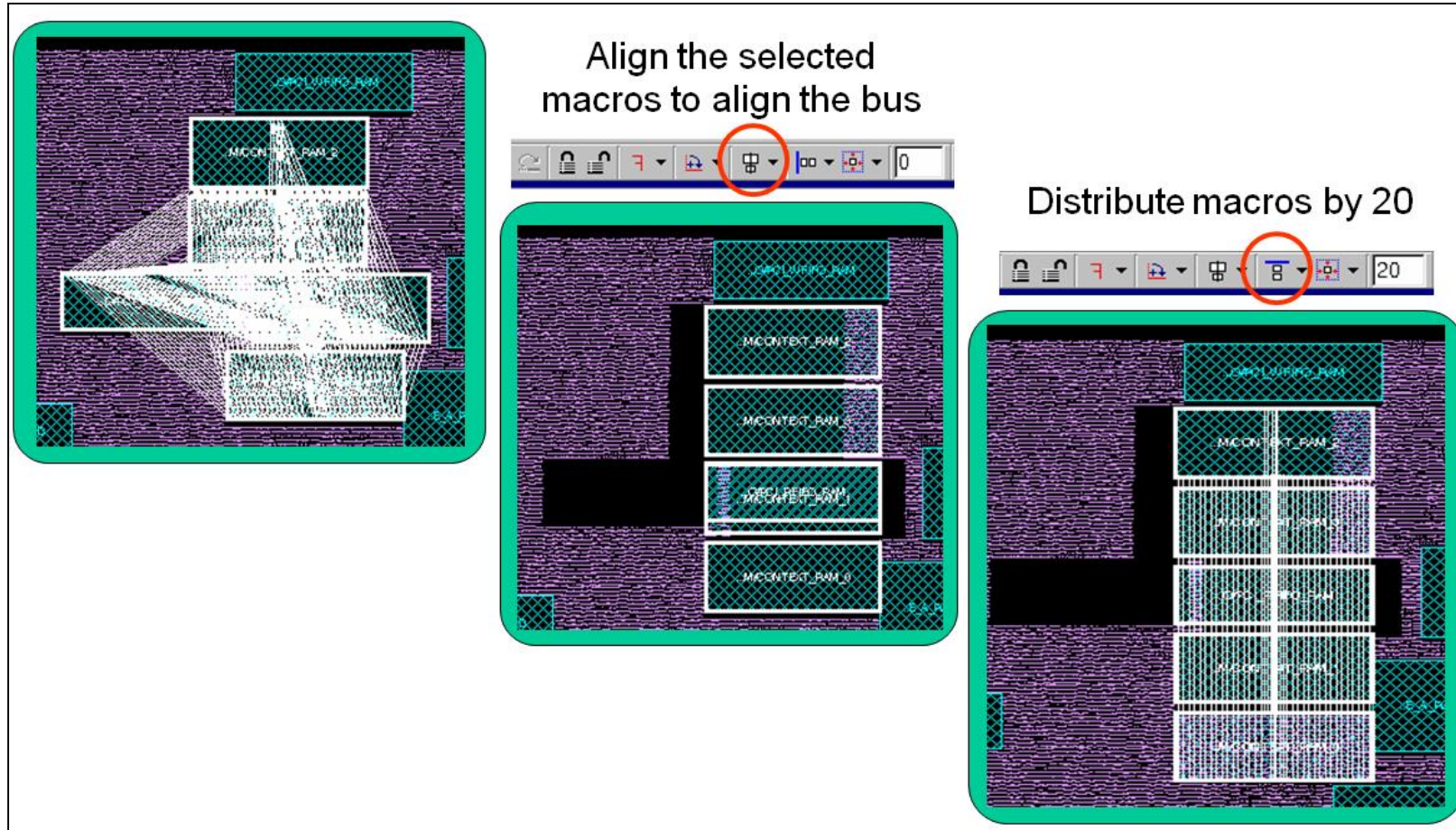
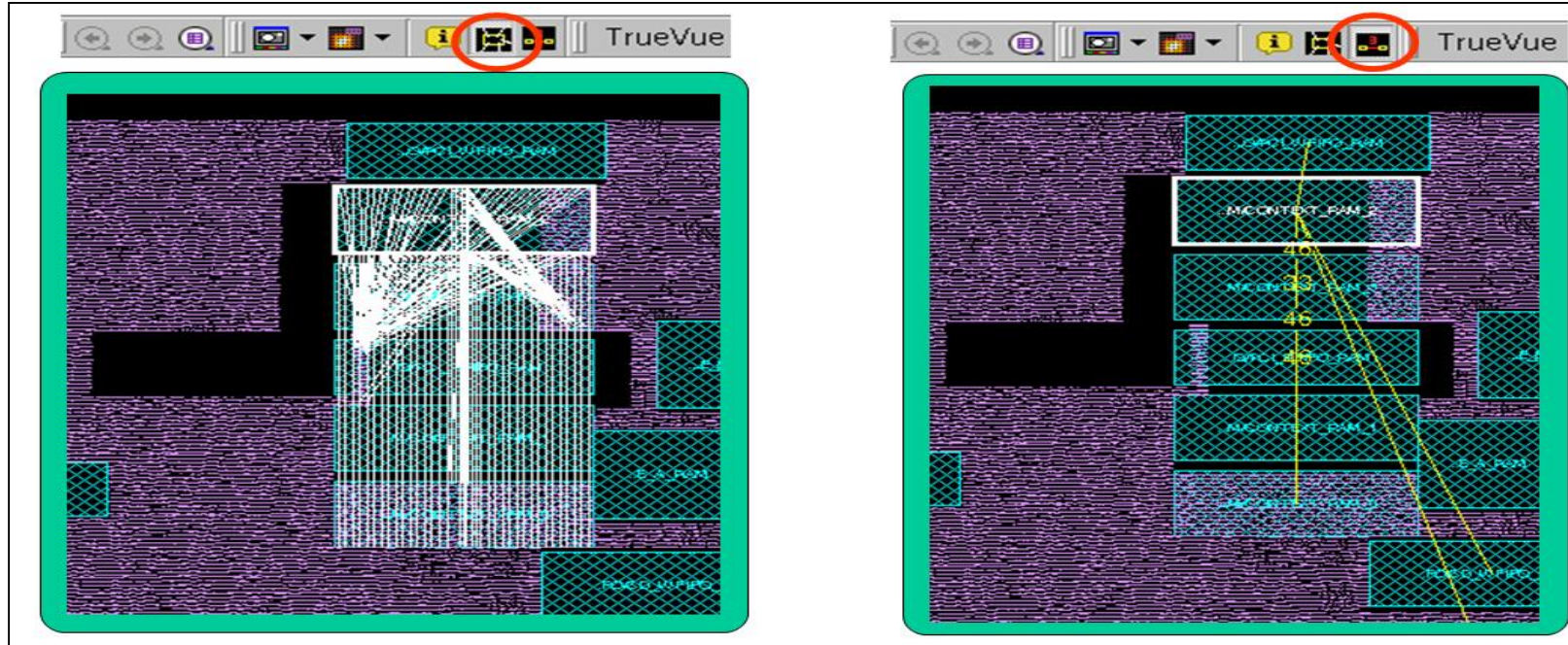


Figure: Synopsys

FP Analysis: Flyline and Net Connectivity



Flyline: a line representing a single net connection

- between two objects
- pins or cell instance objects

Figure: Synopsys

Design View After Floorplanning

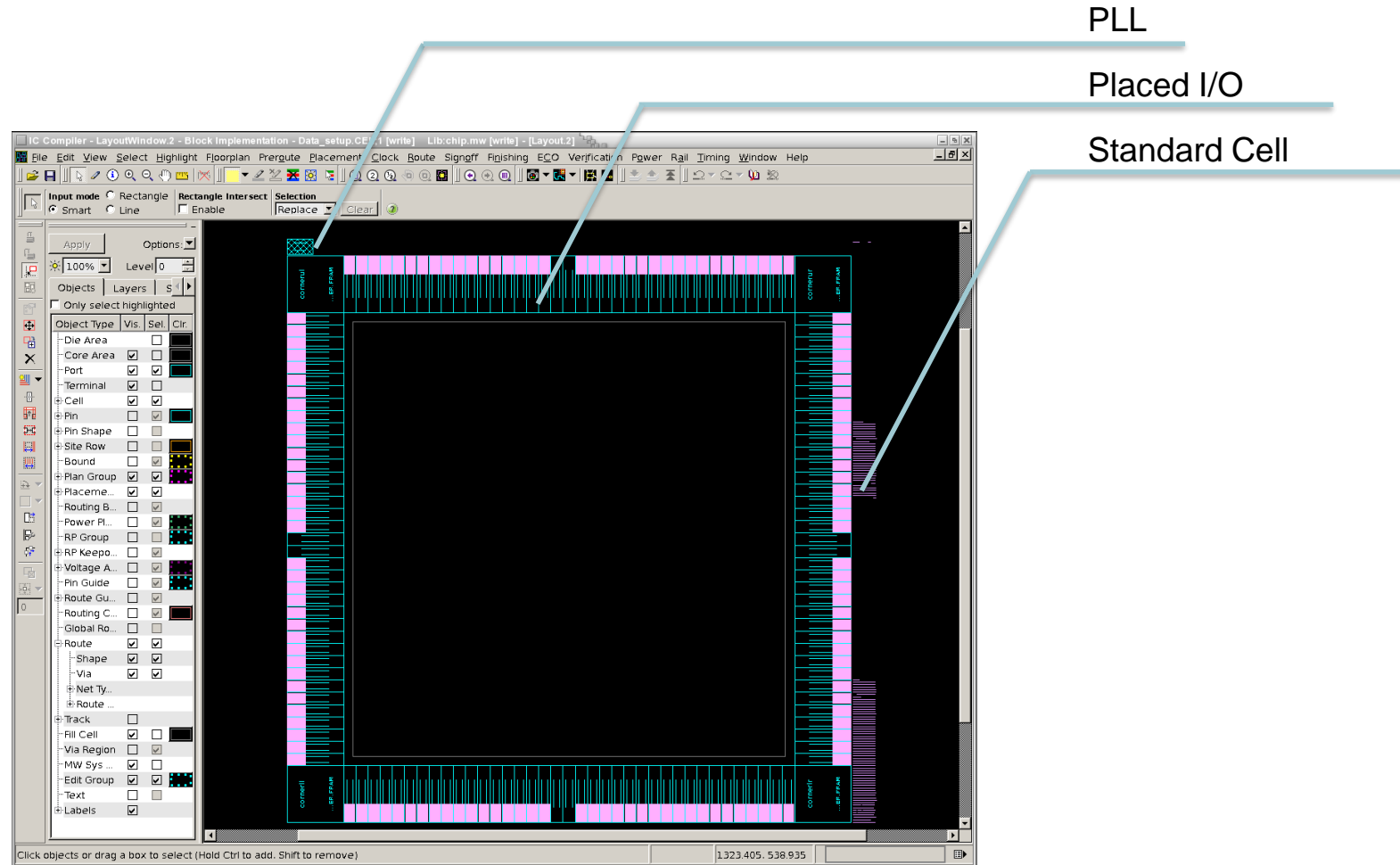
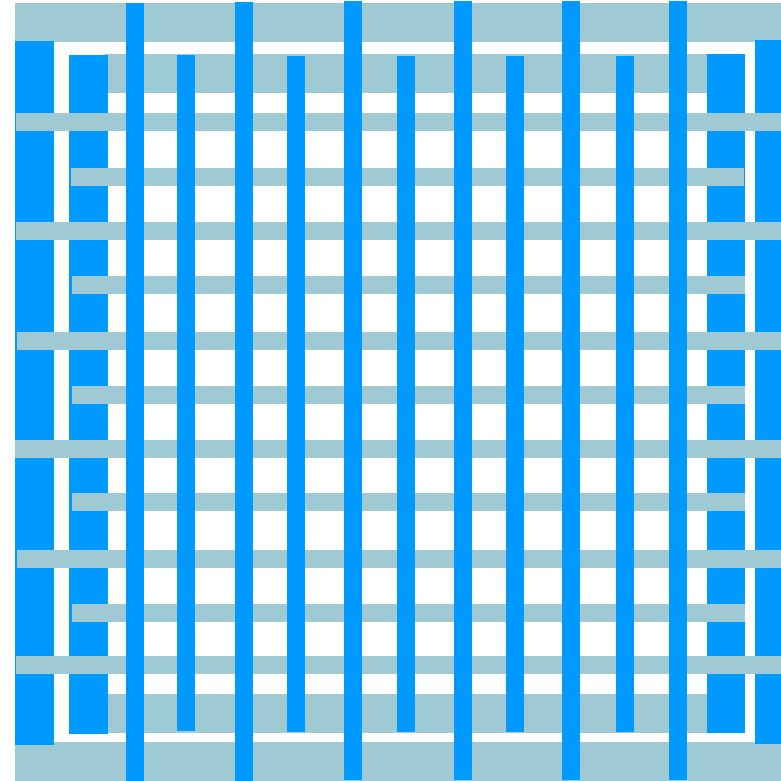


Figure: Synopsys

Power Network Synthesis (PNS)

- Creating power delivery network
 - PNS currently creates a (rectilinear) power plan with/without a core ring connected to a power mesh
 - Designers need to specify
 - Number of straps: min, max
 - Width of straps: min, max
 - Width of ring
 - Layer
 - IR-drop constraint



Trunks are not shown here (they are outside the core rings, between the rings and the pads)

Example: Power Grid

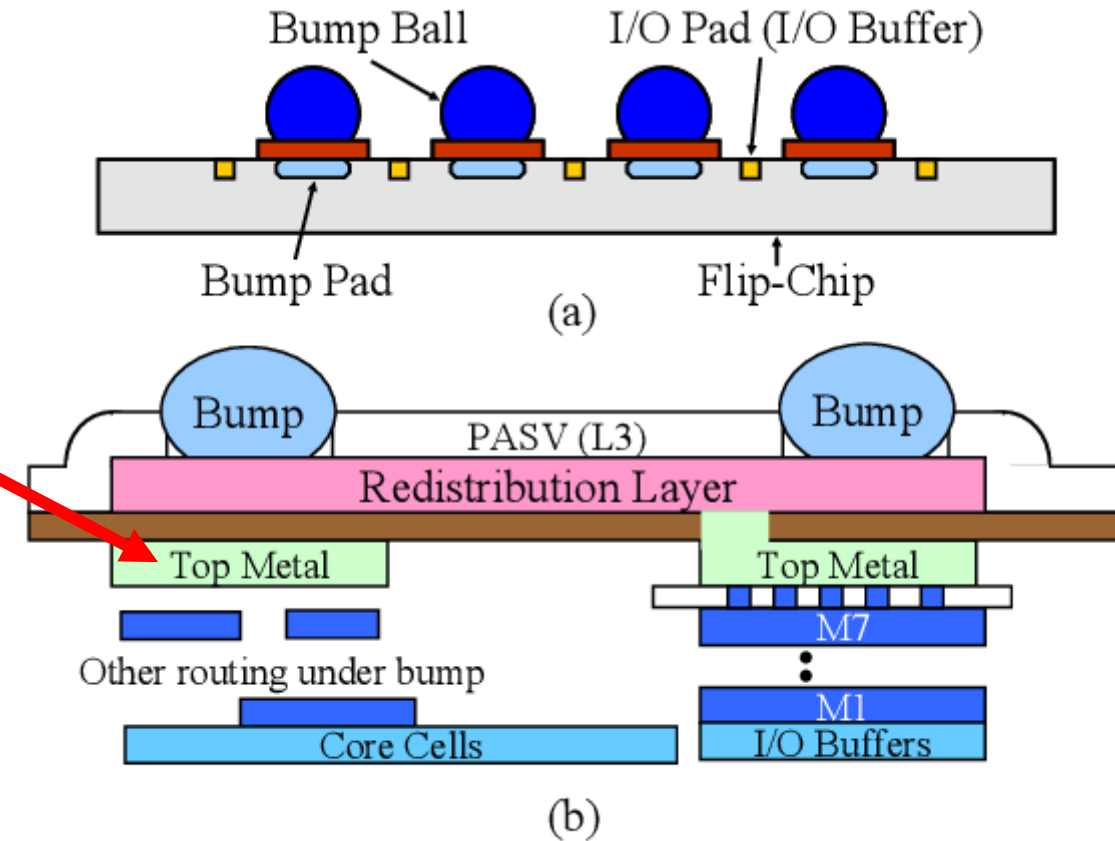
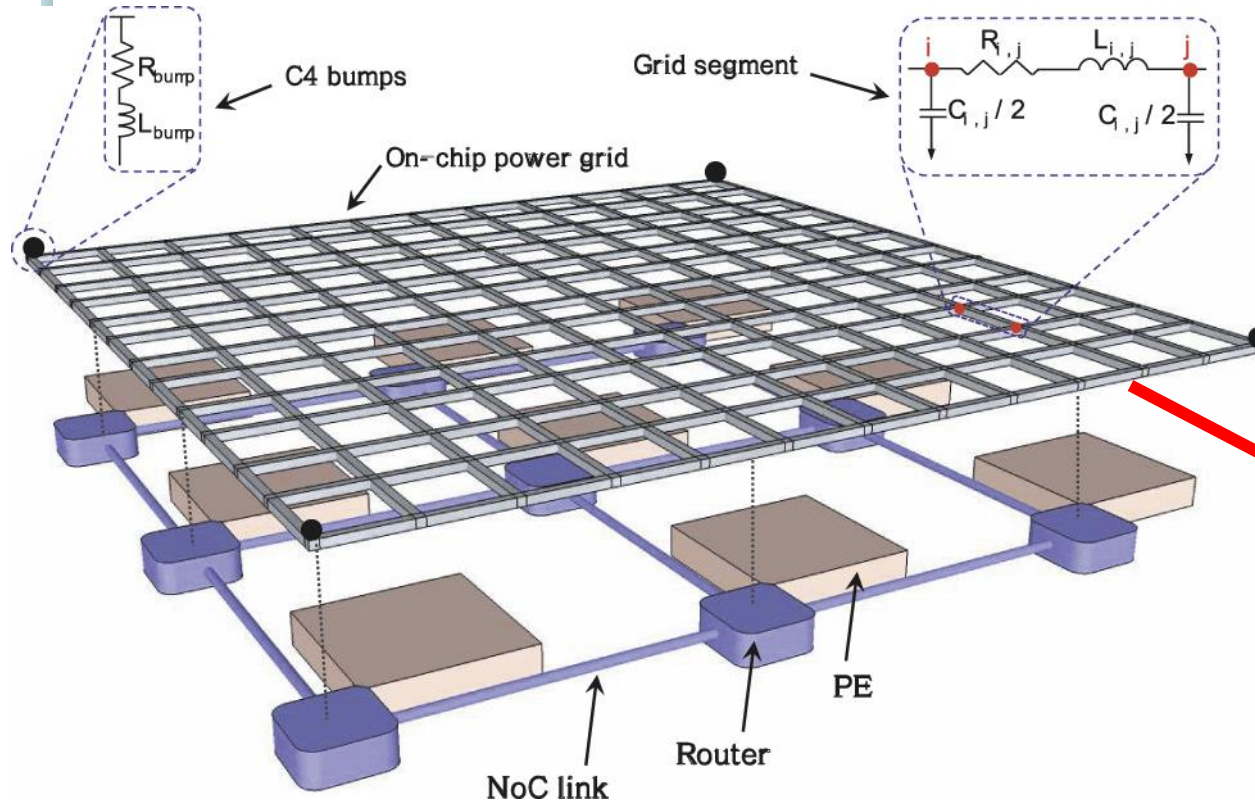


Fig. 1. (a) Flip-Chip Package. (b) Cross Section of RDL.

RDL: extra metal layer on a chip that makes the IO Pads of an IC available in other locations of the chip, for better access to the pads where necessary

Source:

- Dahir N S, Mak T, Xia F, et al. Modeling and tools for power supply variations analysis in networks-on-chip[J]. IEEE Transactions on Computers, 2012, 63(3): 679-690.
- Fang J W, Chang Y W. Area-I/O flip-chip routing for chip-package co-design[C]//2008 IEEE/ACM International Conference on Computer-Aided Design. IEEE, 2008: 518-522.

Macro Placement Constraints

- Macro placement constraints have an impact on placing and further global routing of standard cells
- Some basic constraints of macro placement
 - Alignment of macros by edges
 - Grouping macros in a way that for standard cells rectangular area should be left as much as possible
 - Alignment of macros with core boundary
- Some basic parameters of Macros Placement
 - Routability
 - Timing
 - Wire length
 - Area for standard cells

Creating a User-Defined Array of Hard Macros

Alignment to other macros, pins, and edge

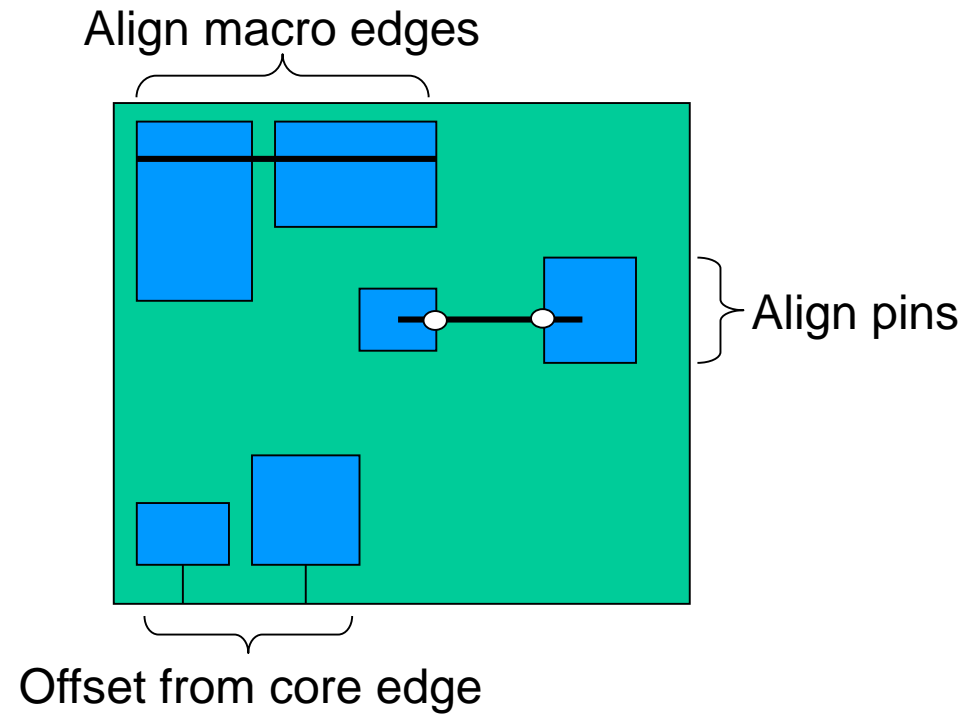


Figure: Synopsys

Rectangular Rings and Power Straps

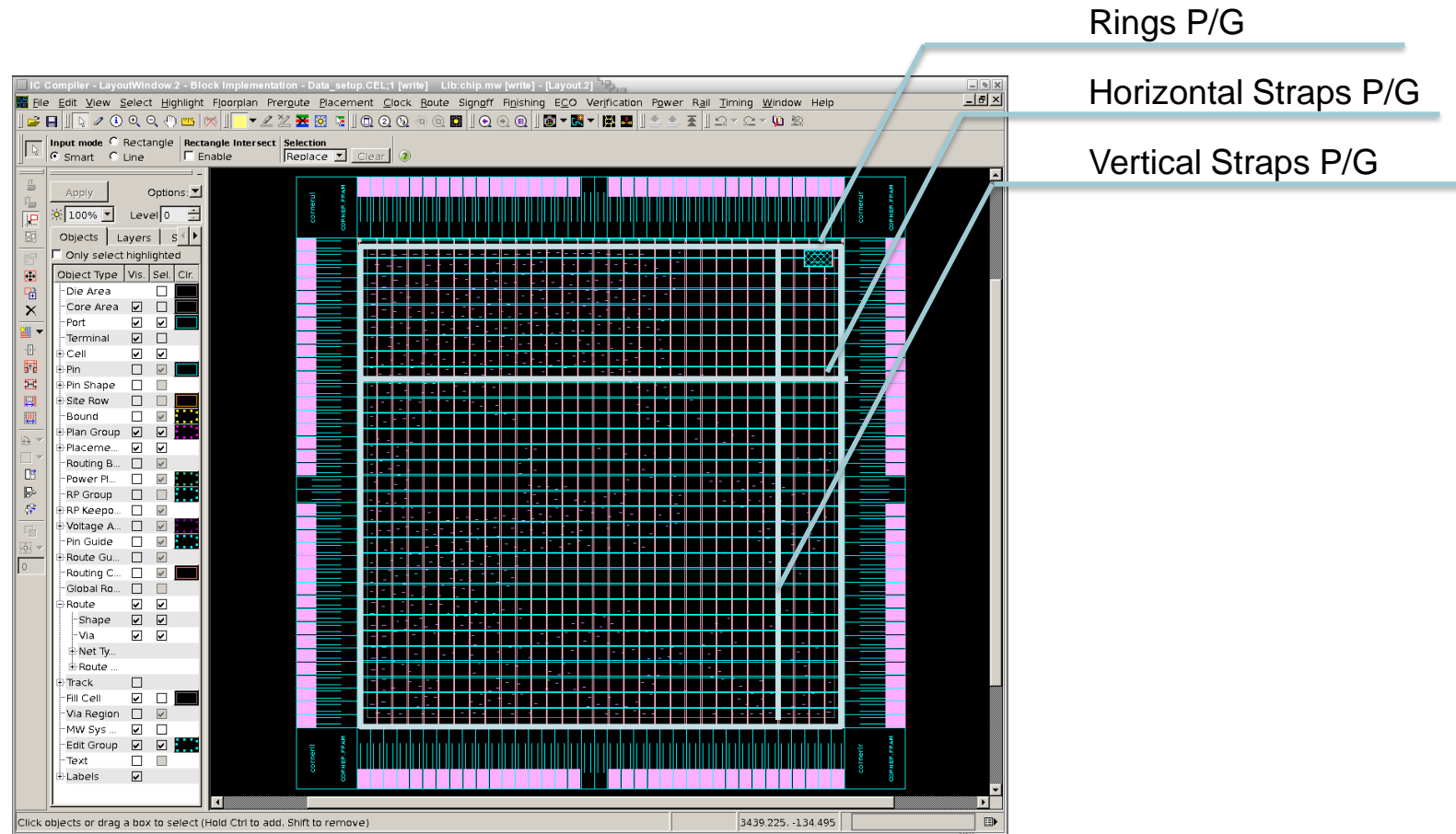


Figure: Synopsys

Analyze Voltage Drop

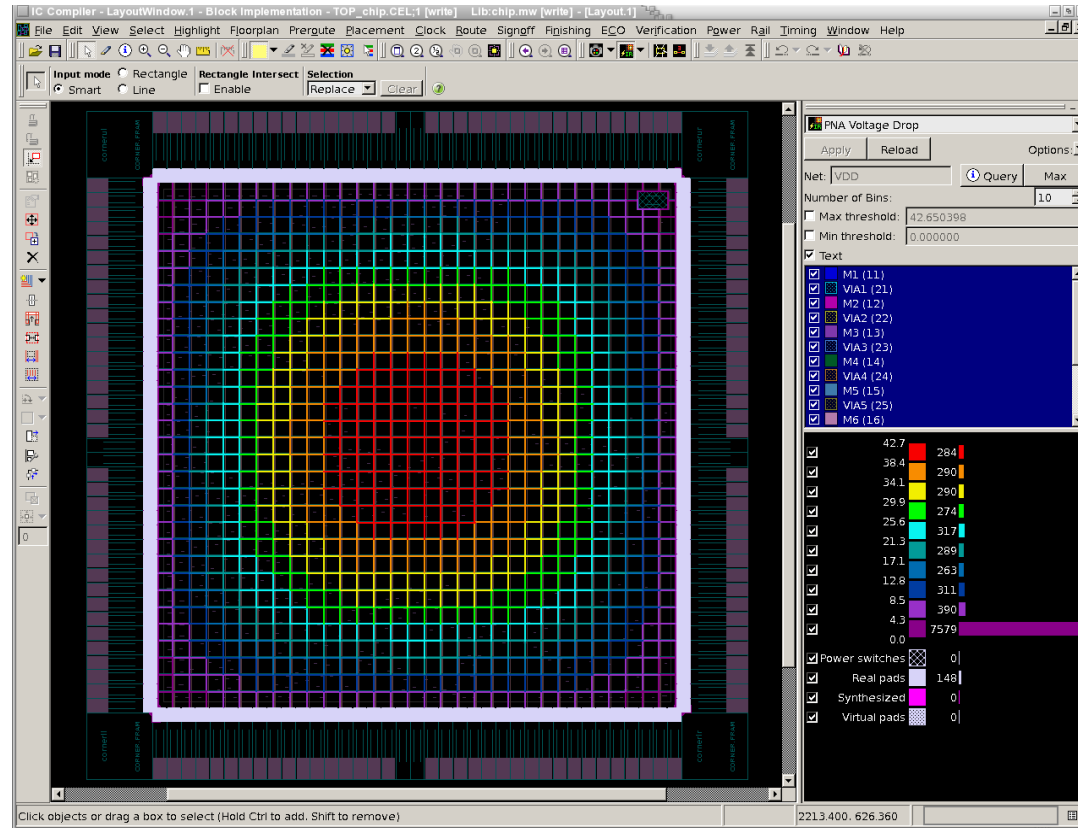
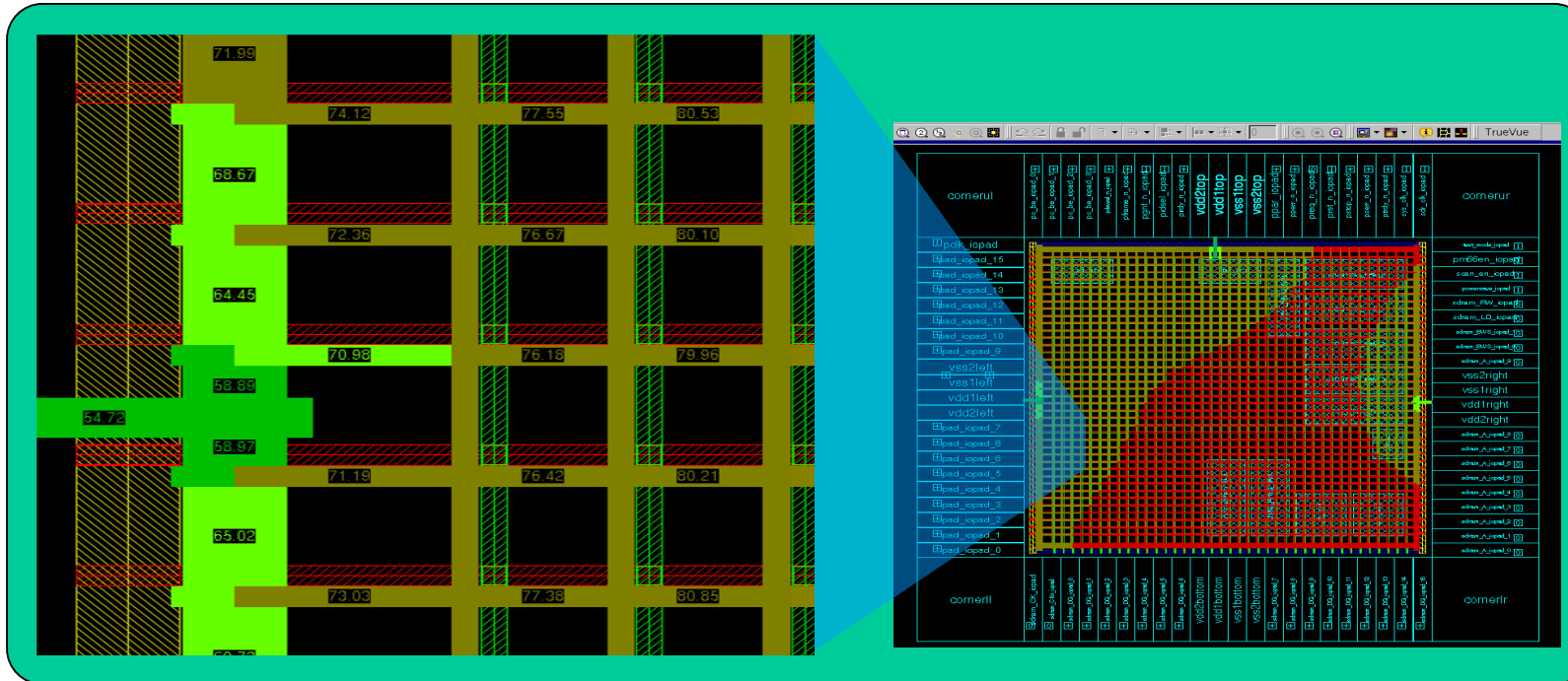


Figure: Synopsys

Display Voltage (IR) Drop



Can continue to add/delete virtual power pads to see if voltage drop map gets better

Floorplanning problem

The floorplanning problem is to plan the positions and shapes of the modules at the beginning of the design cycle to optimize the circuit performance:

- chip area
- total wirelength
- delay of critical path
- routability
- others, e.g., **noise**, **heat** dissipation, etc.

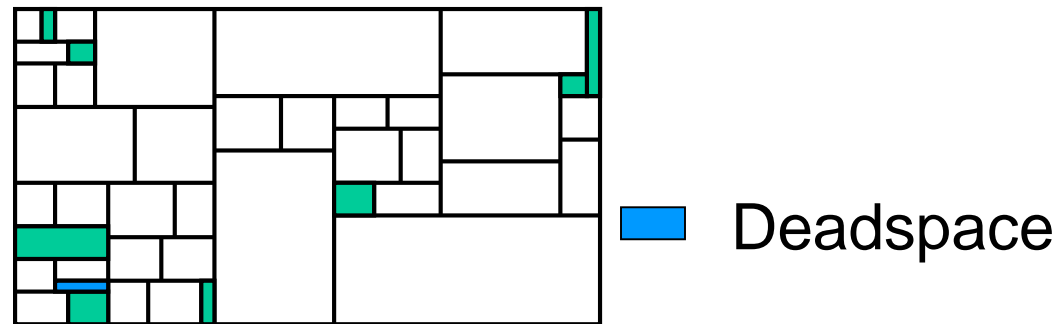


Figure: Synopsys

Floorplanning problem

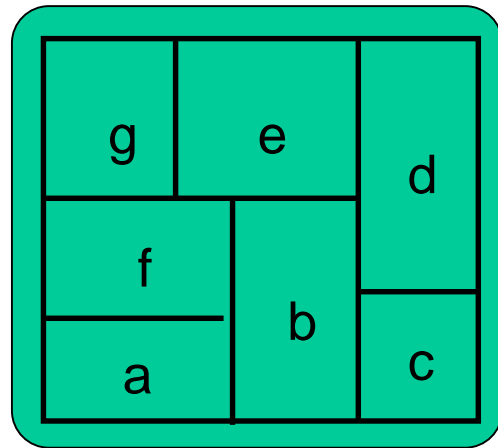
Problem formulation

Input: n Blocks with areas A_1, \dots, A_n

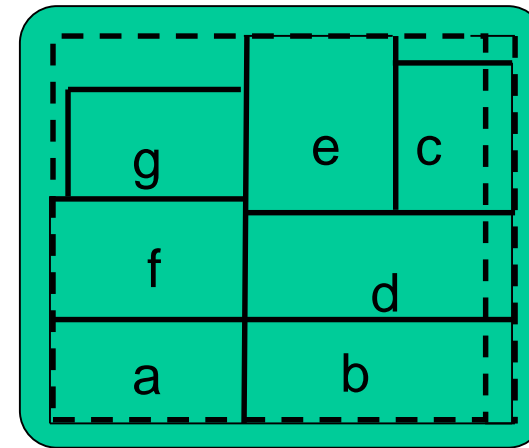
Bounds r_i and s_i on the aspect ratio of block B_i

Output: Coordinates (x_i, y_i) , width w_i and height h_i for each block such that $h_i w_i = A_i$ and $r_i \leq h_i/w_i \leq s_i$

Objective: Optimize the circuit performance.



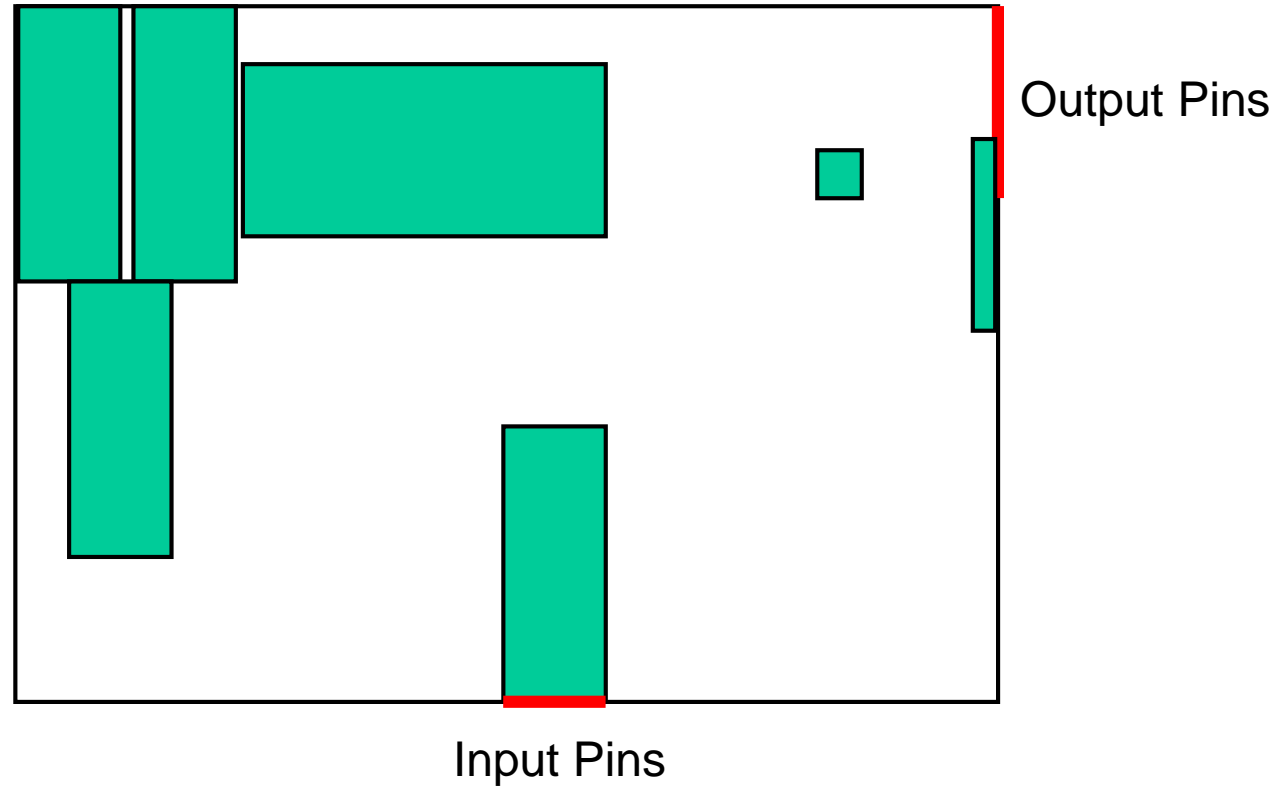
An optimal floorplan in terms of area



A non-optimal floorplan

Figure: Synopsys

A bad floorplan



Placement

Where should we place cells on the floorplan?

Physical Synthesis Flow

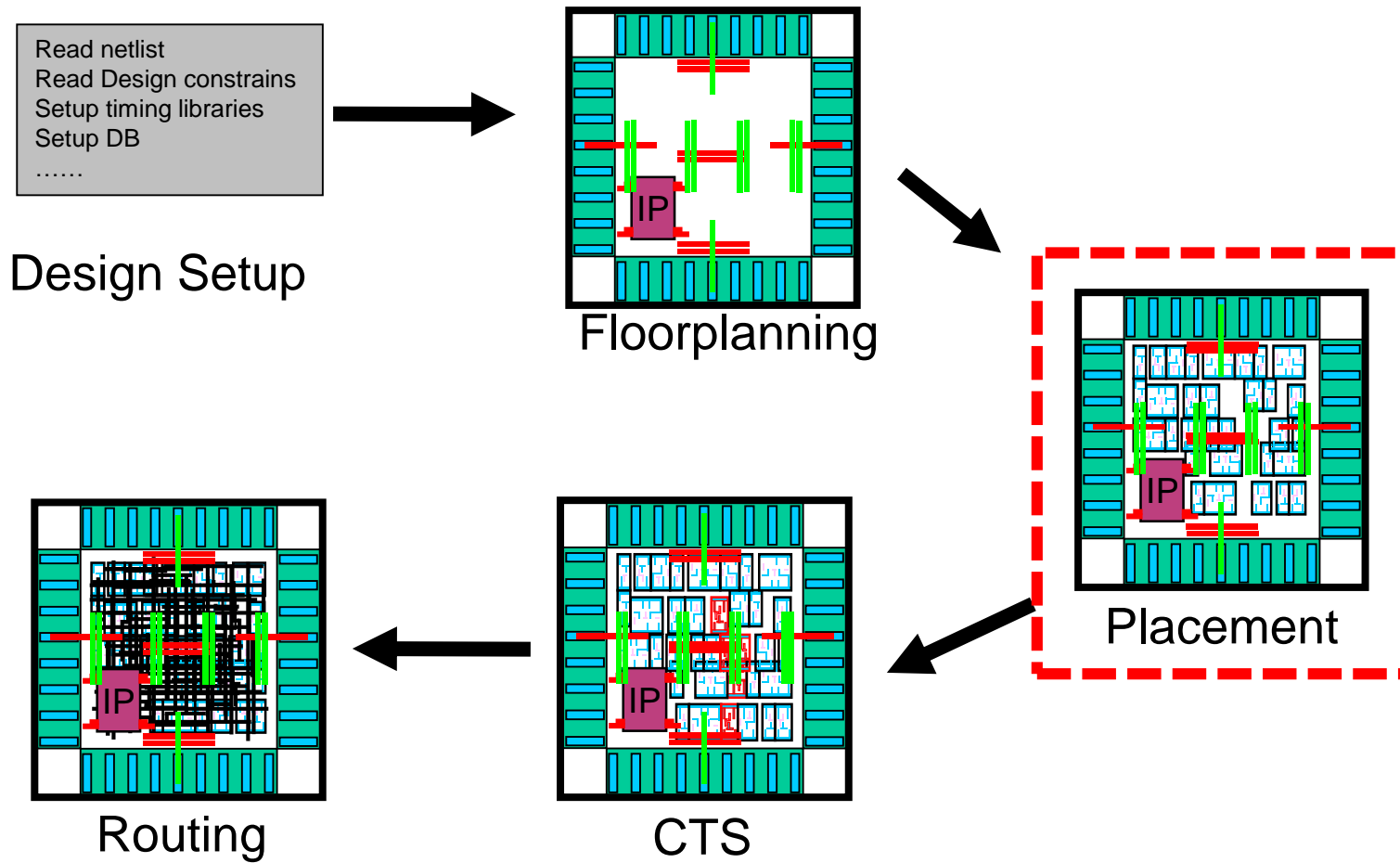
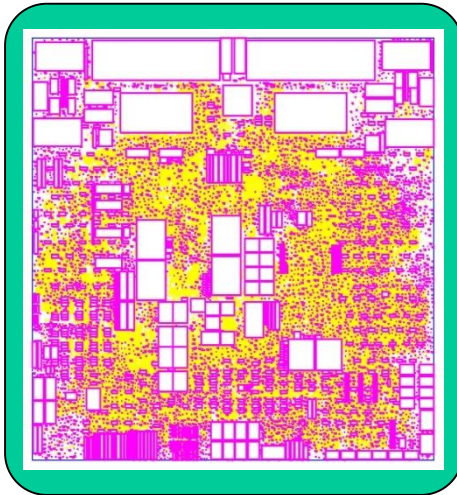


Figure: Synopsys

Placement Problem

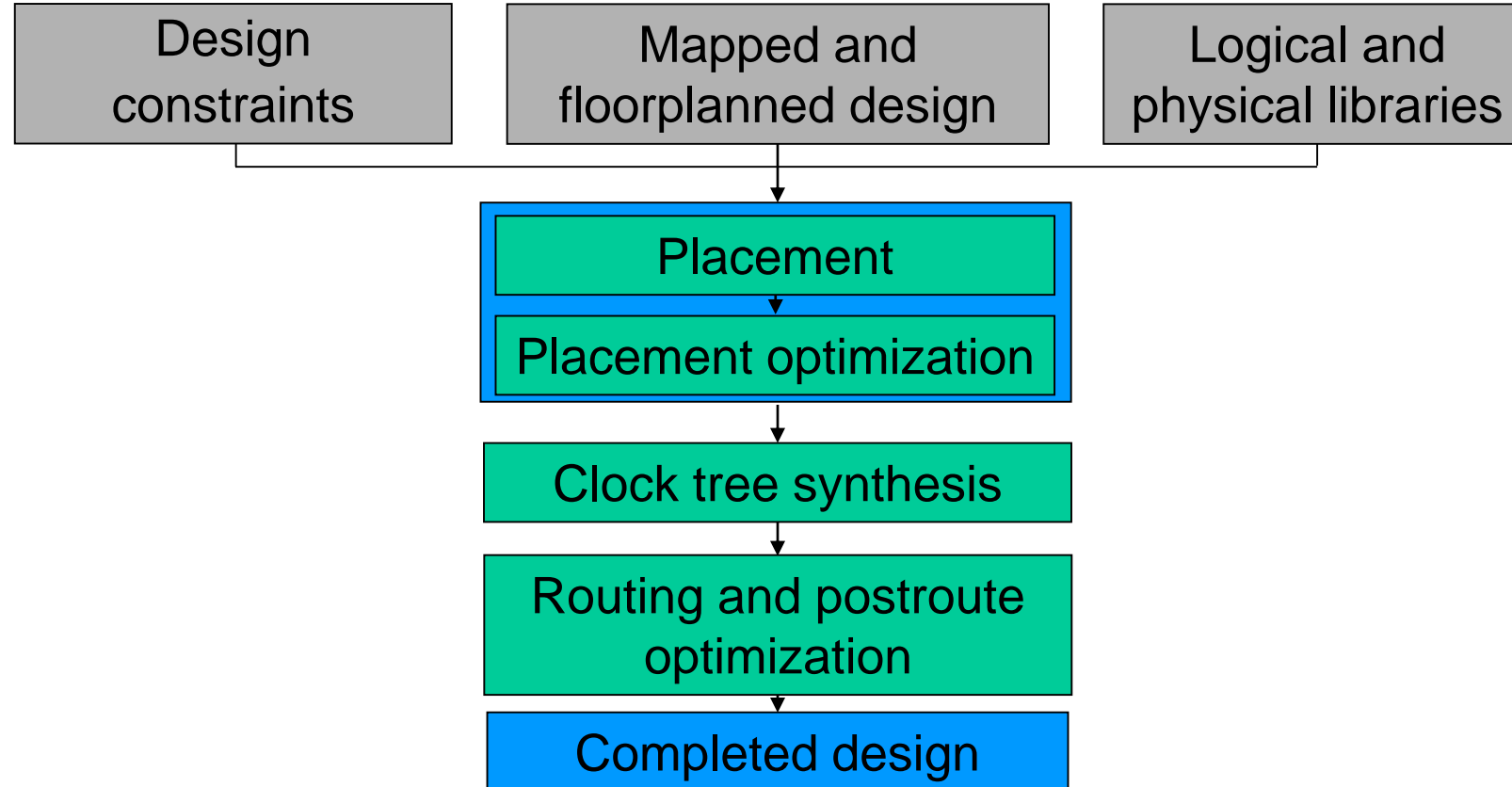
- The goal of placement is to **minimize the total area and interconnect cost.**



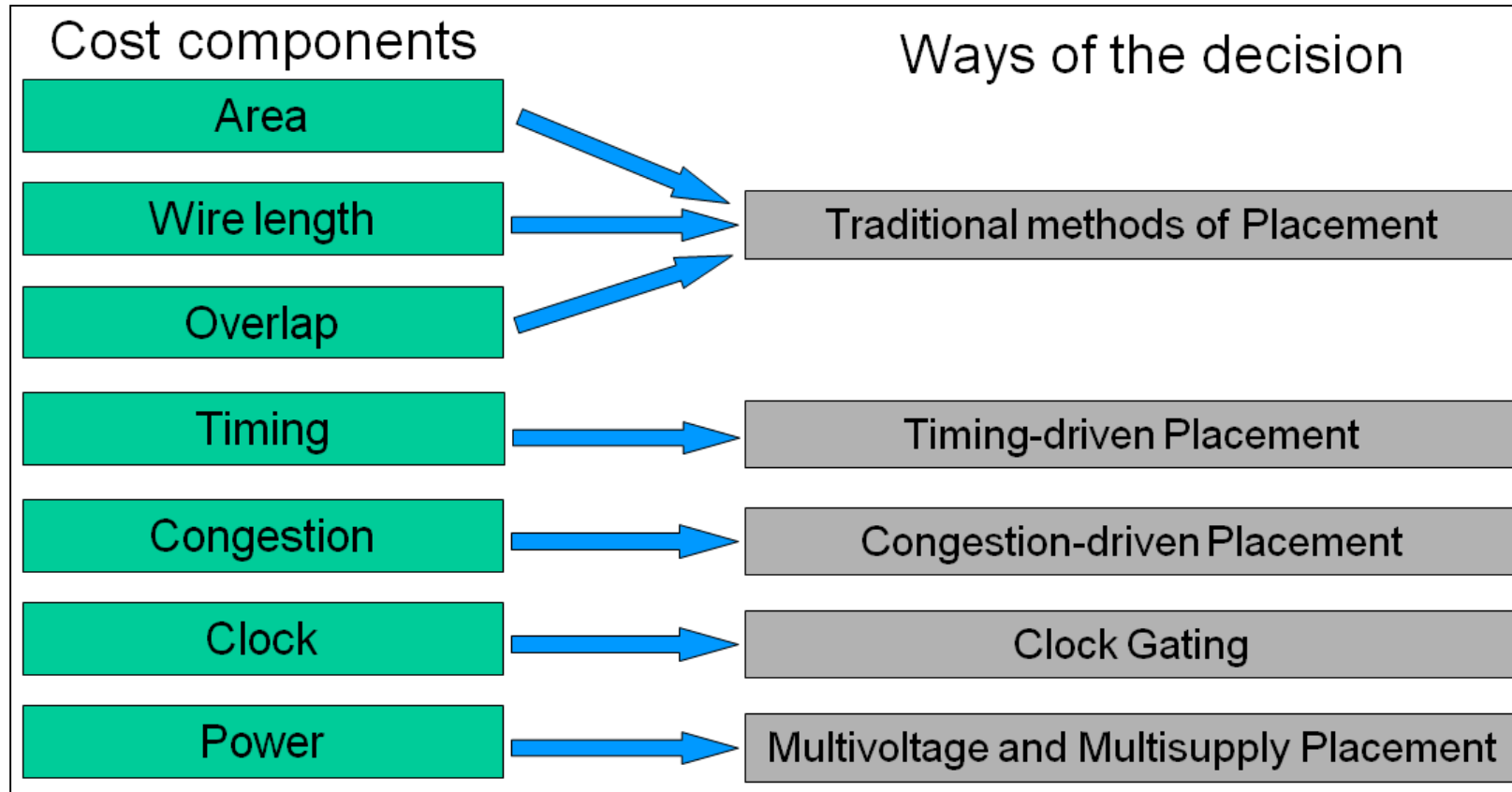
The quality of the attainable routing is highly determined by the placement.

Circuit placement becomes very critical in 90nm and below technologies.

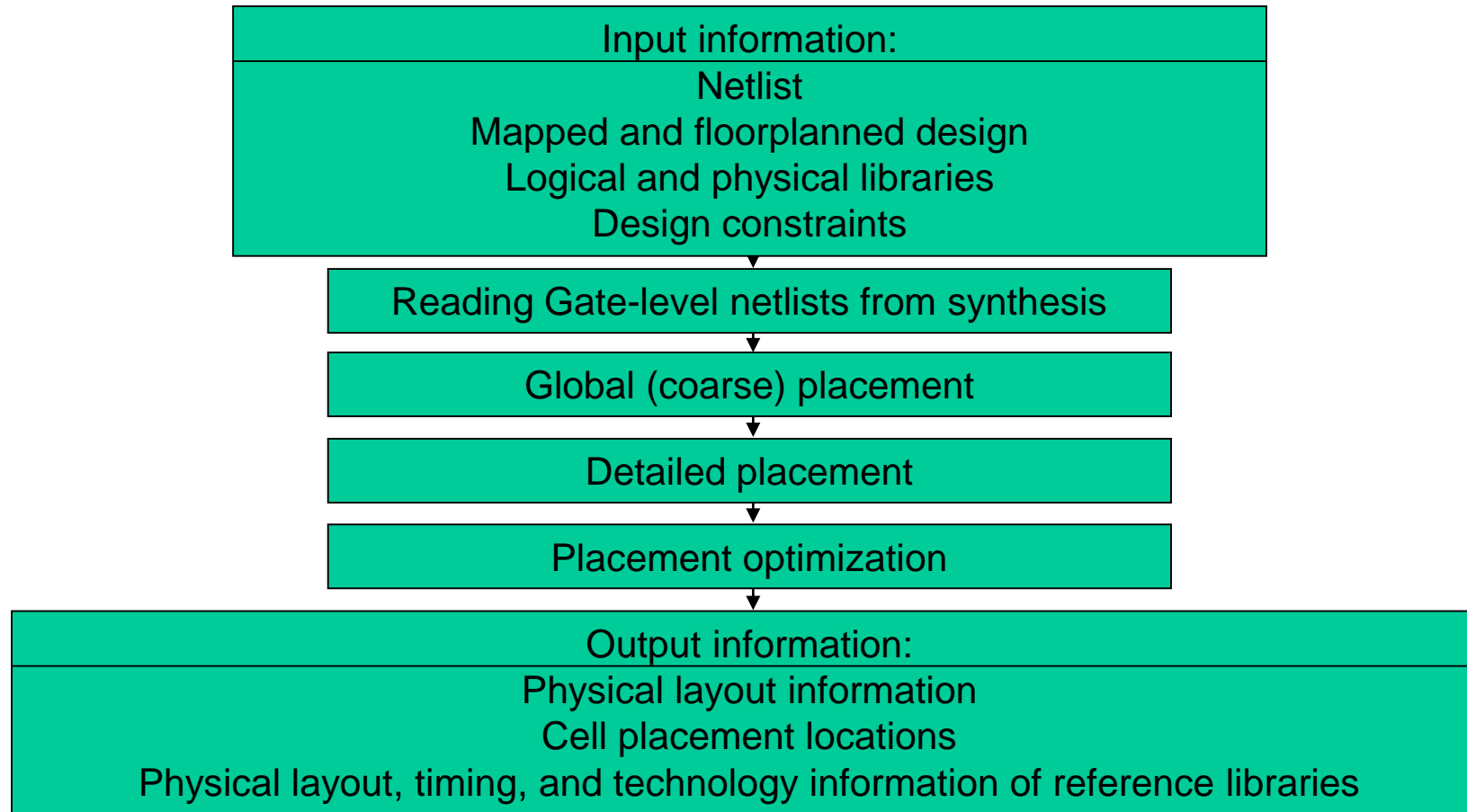
Location of Placement in a Typical IC Physical Design Flow



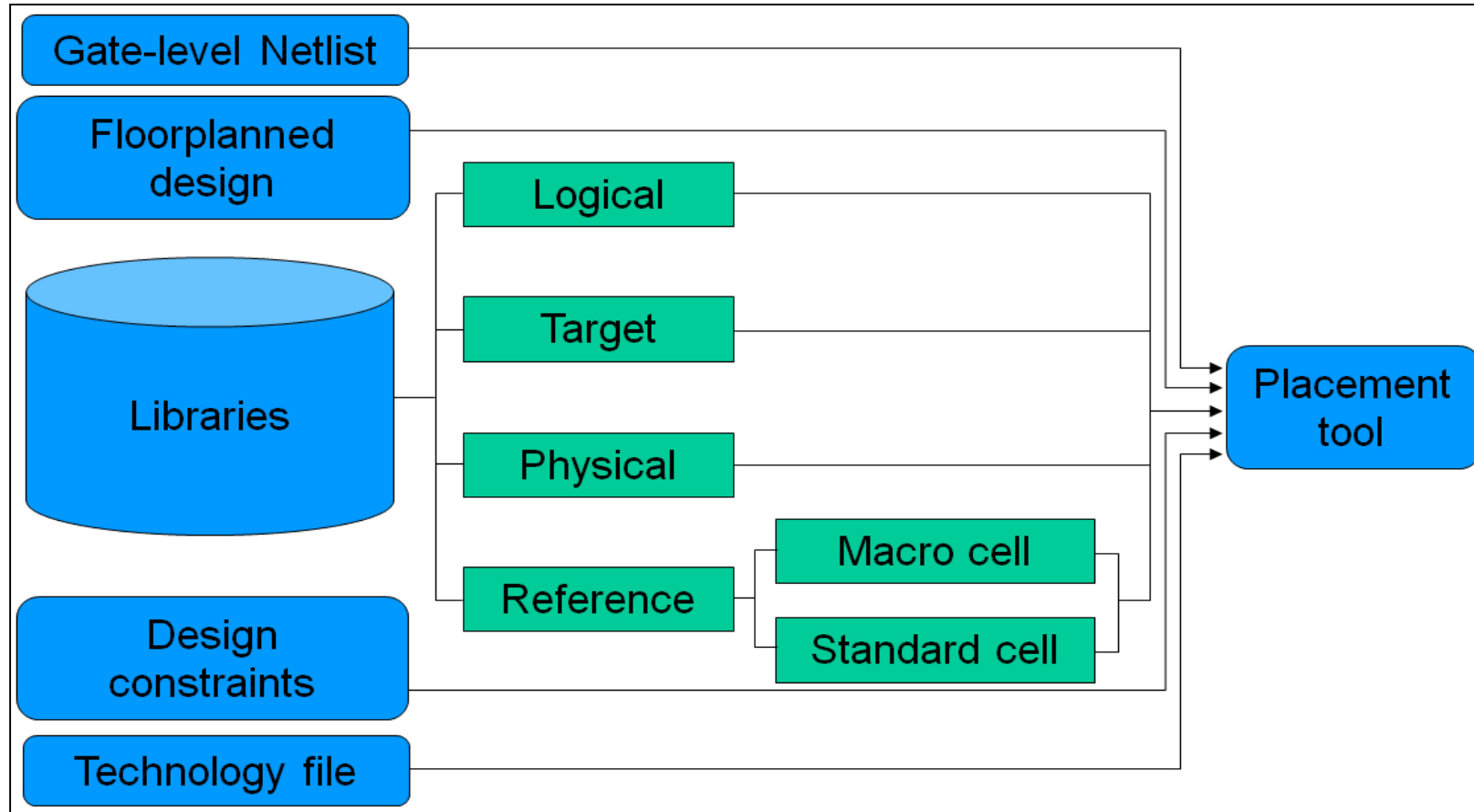
New Tendencies of Physical Designing and Placement Cost Components



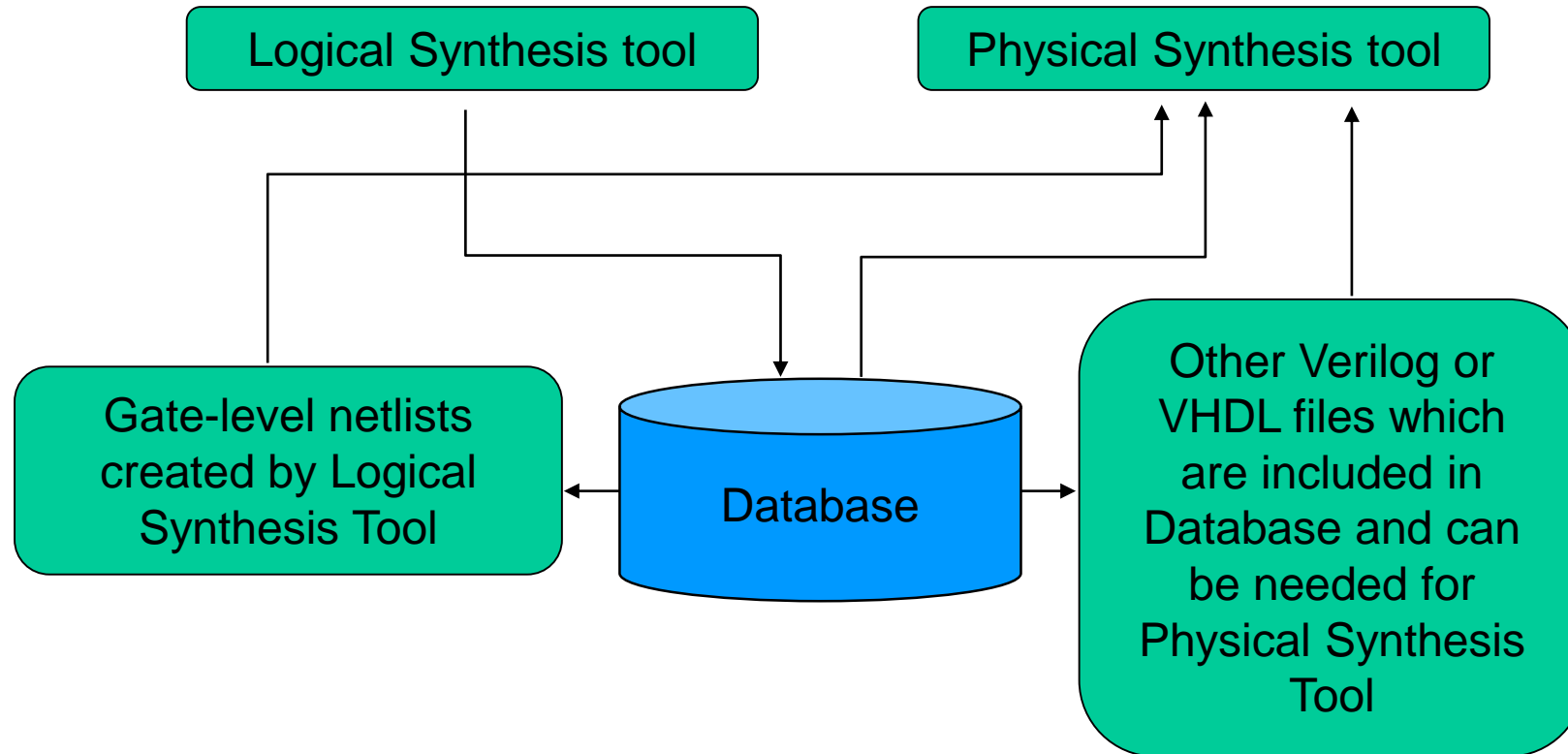
Placement Steps



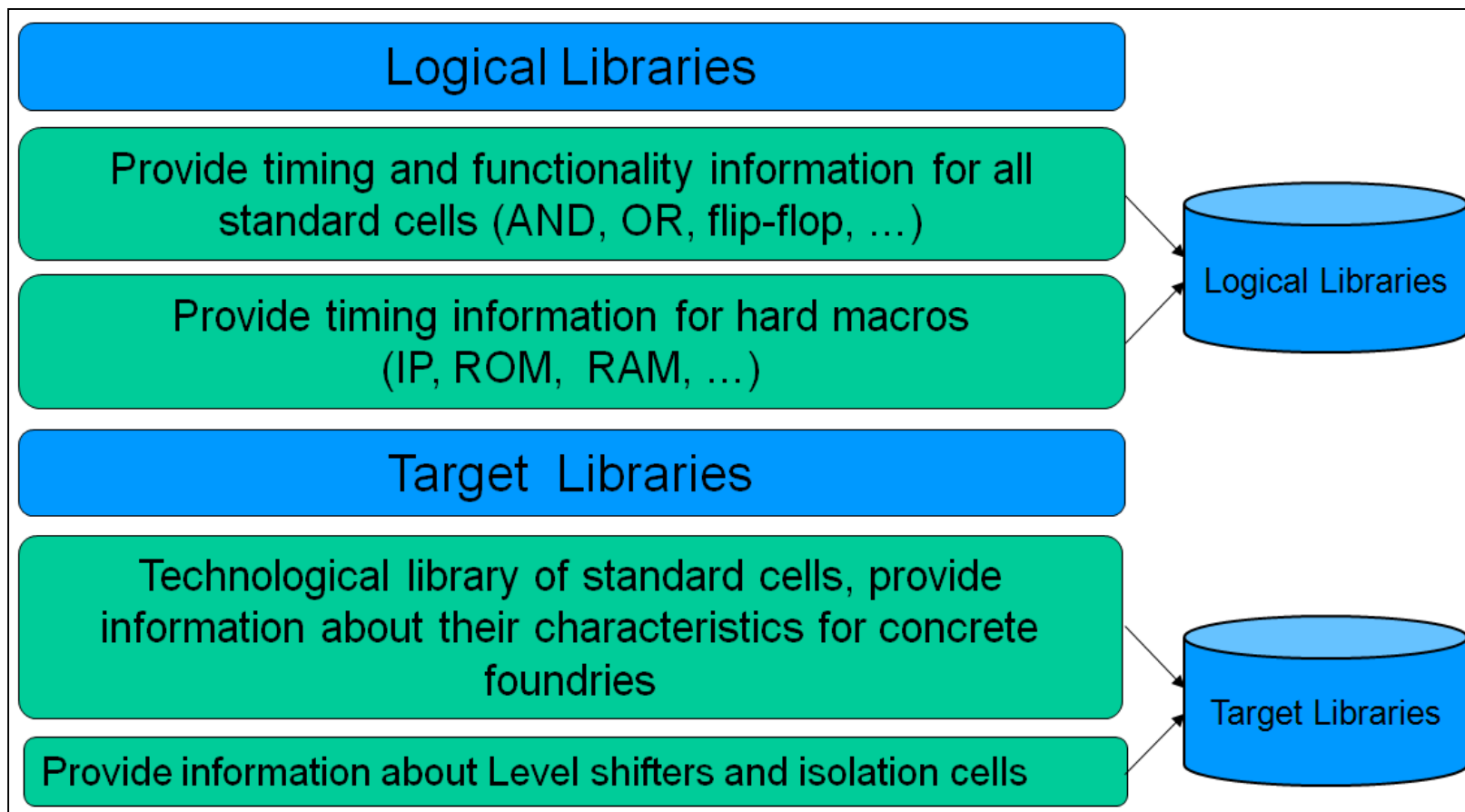
Input Information for Placement Tool



Reading Gate-Level Netlists from Synthesis

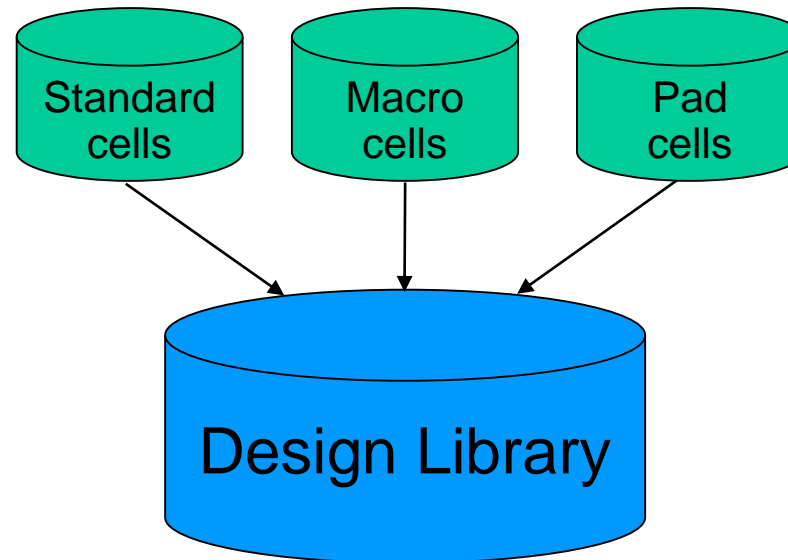


Logical and Target Libraries



Reference Libraries

- Contain subdesigns or cells used by many other designs
- Referenced by pointers in the design library for memory efficiency



Technology File

- Tech File is unique to each technology
- Contains metal layer technology parameters
 - Number and name designations for each layer/via
 - Dielectric constant for technology
 - Physical and electrical characteristics of each layer/via
 - Design rules for each layer/Via (Minimum wire widths and wire-to-wire spacing, etc.)
 - Units and precision for electrical units
 - Colors and patterns of layers for display
 - . . .

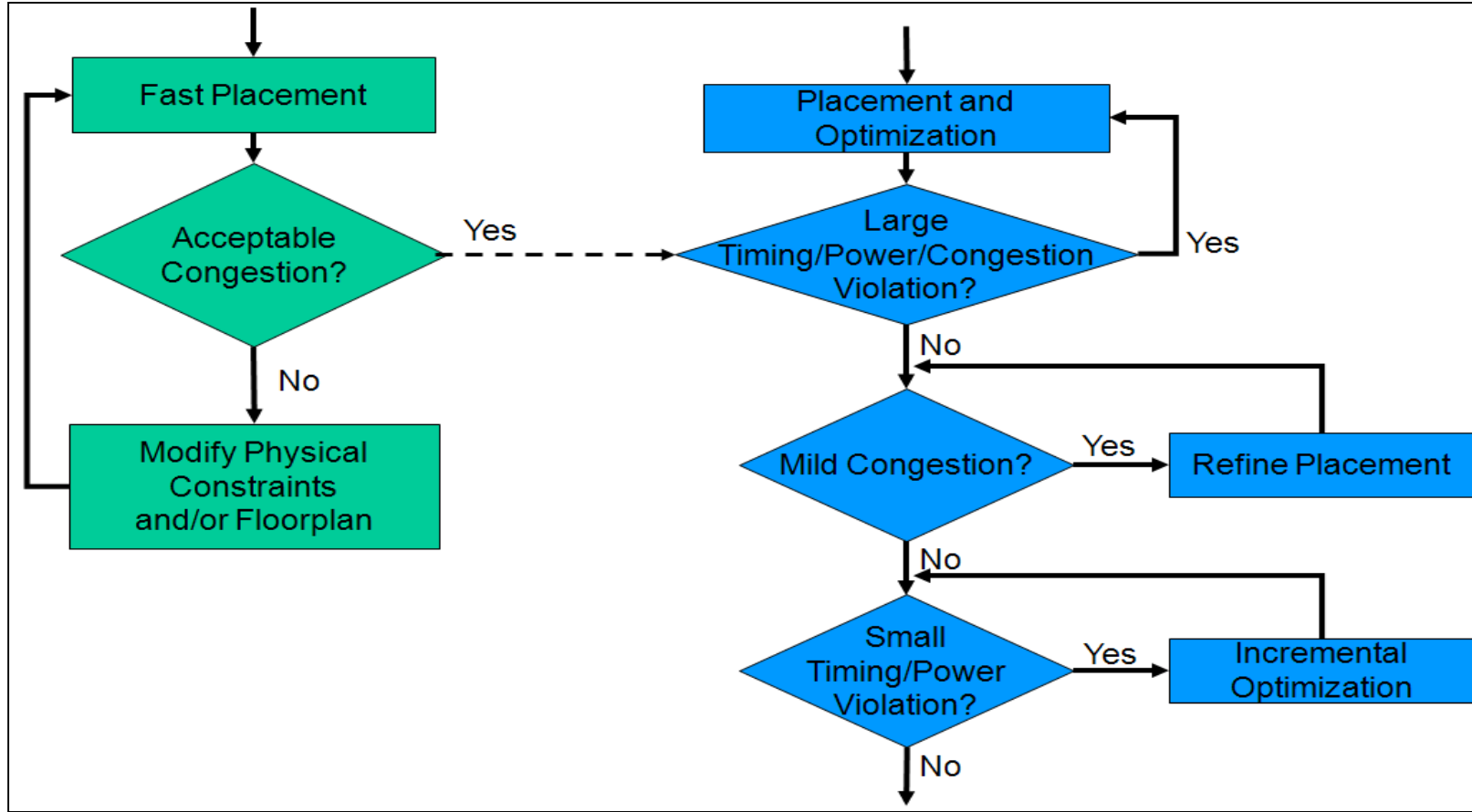
Example of a Technology File

```
Technology {  
  dielectric    = 3.7  
  unitTimeName = "ns"  
  timePrecision = 1000  
  unitLengthName = "micron"  
}  
  
...  
Layer "m1" {  
  layerNumber = 16  
  maskName    = "metal1"  
  pitch       = 0.56  
  defaultWidth = 0.23  
  minWidth    = 0.23
```

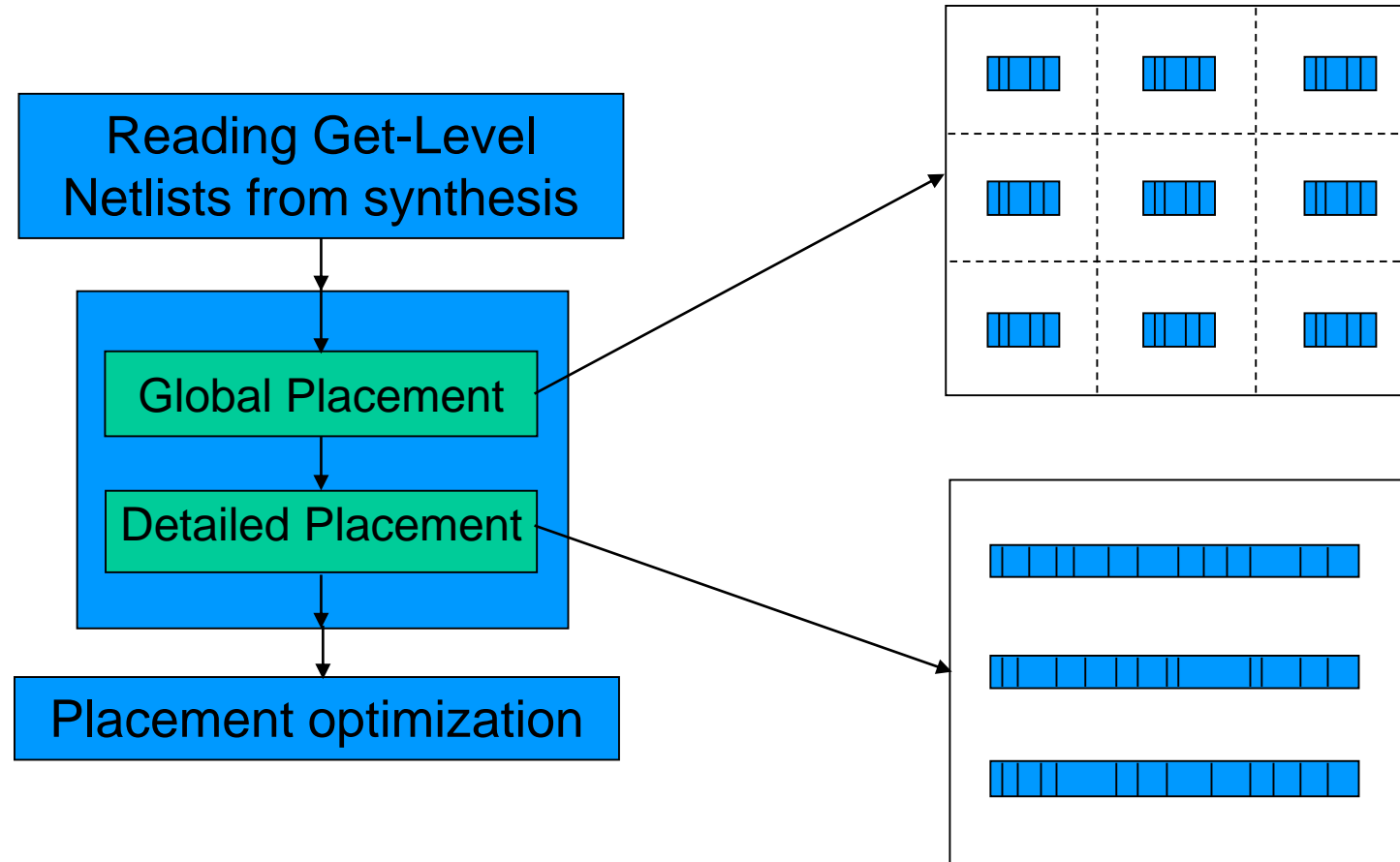
Placement and Optimization Attributes

Typical Attributes	Coarse placement	Detailed placement	Optimization
Fixed	Cannot move cells	Cannot move cell	Cannot move, rotate, or resize cells
Imposed on clock buffers	Cannot move cells	Cannot move cells	Cannot move, rotate, or resize cells
Soft fixed	Cannot move cells	No restrictions	No restrictions
Size only	No restrictions	No restrictions	Can only resize cells
In place size only	Cannot move cells	No restrictions	Can resize cells only if there is room
Imposed on clock sinks	No restrictions	No restrictions	Can resize cells only if there is room
Don't touch	No restrictions	No restrictions	Cannot move, rotate, or resize cells

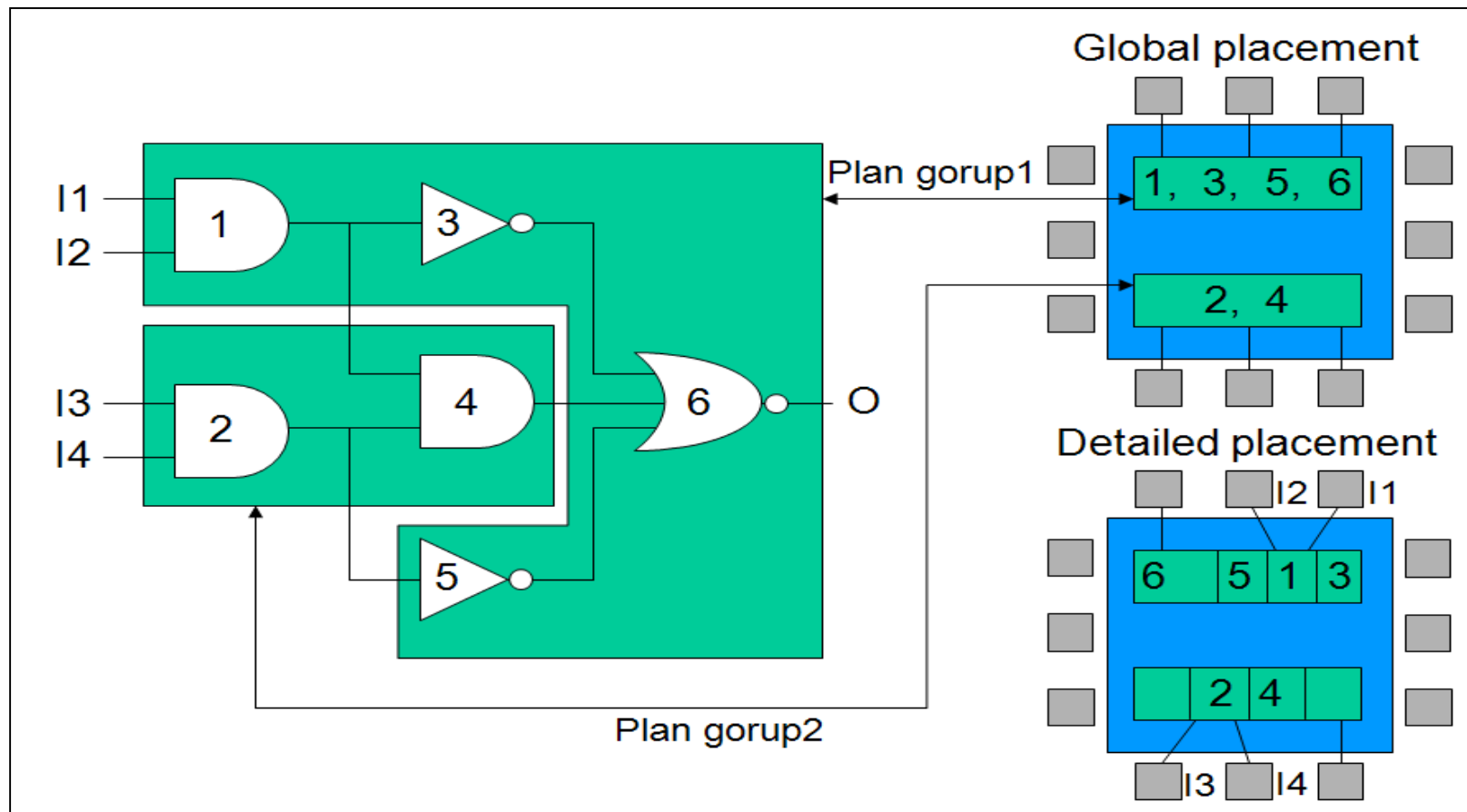
Placement Methodology



Global and Detailed Placement



Partitioning-Based Placement

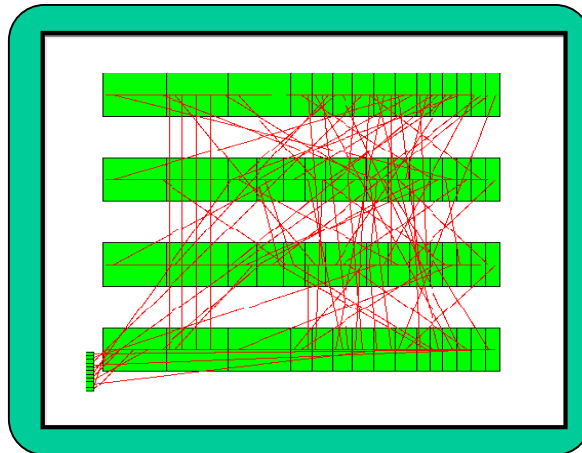


Global Placement

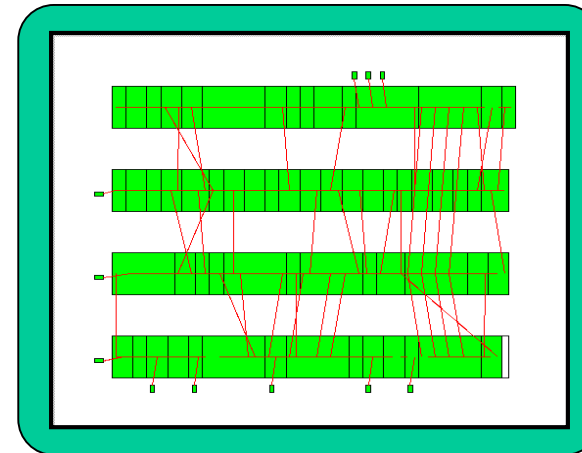
Standard cells must be in groups in such a way that the number of connections between groups is minimum

This issue is solved through circuit partitioning

As a basic criterion, the minimum is taken among group connections

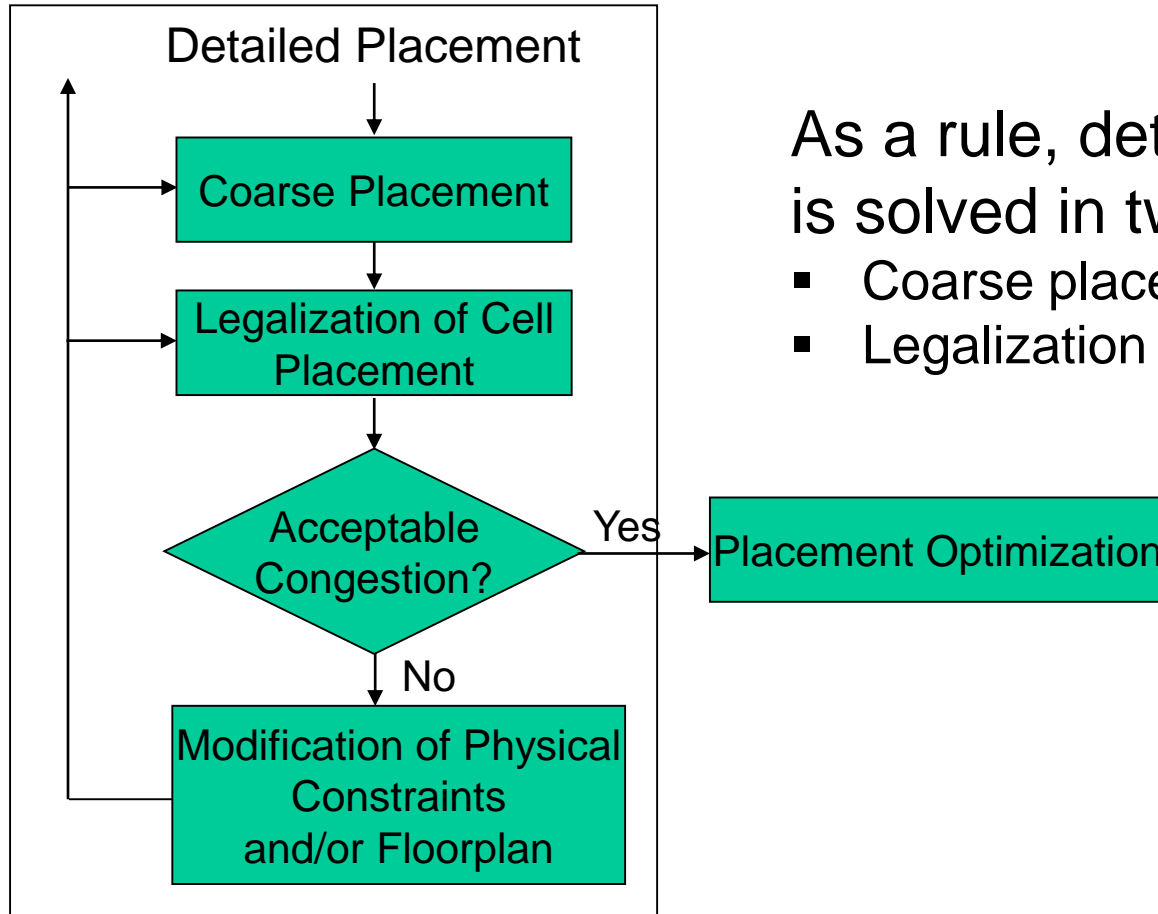


Bad Placement



Good Placement

Detailed Placement



As a rule, detailed placement is solved in two stages:

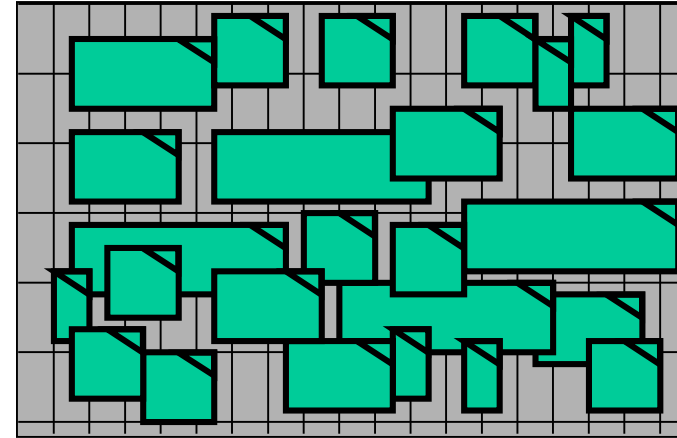
- Coarse placement
- Legalization of cell placement

Coarse Placement

Coarse Placement

All the cells are placed in the approximate locations, but they are not legally placed

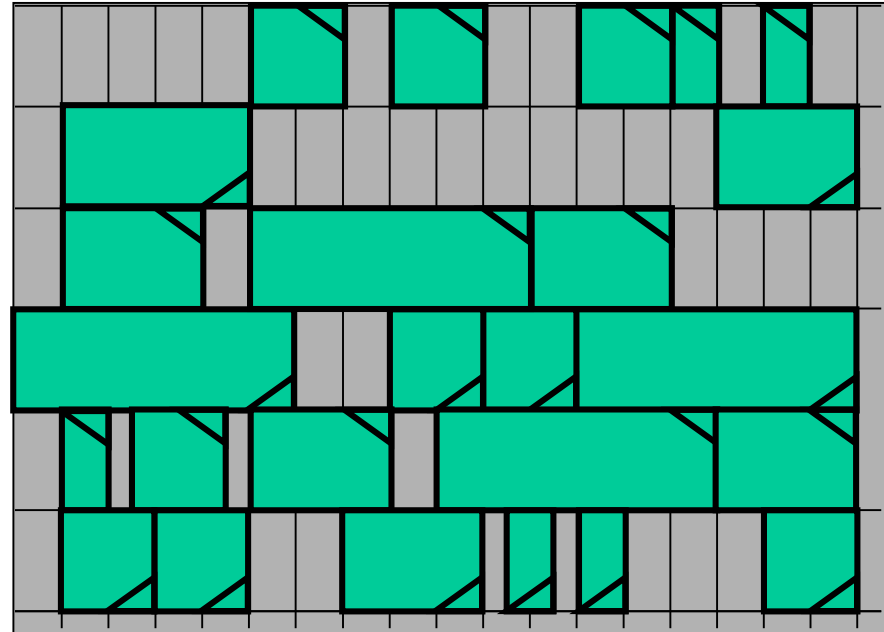
No logic optimization is done



- In a coarse placement all the cells are placed in the approximate locations but they are not legally placed.
- Cells overlap and are not on-grid.
- Large cells (e.g. RAMs) form large placement blockages for other smaller leaf cells.
- Power routing forms routing layer blockages that will also be checked and avoided if specified.

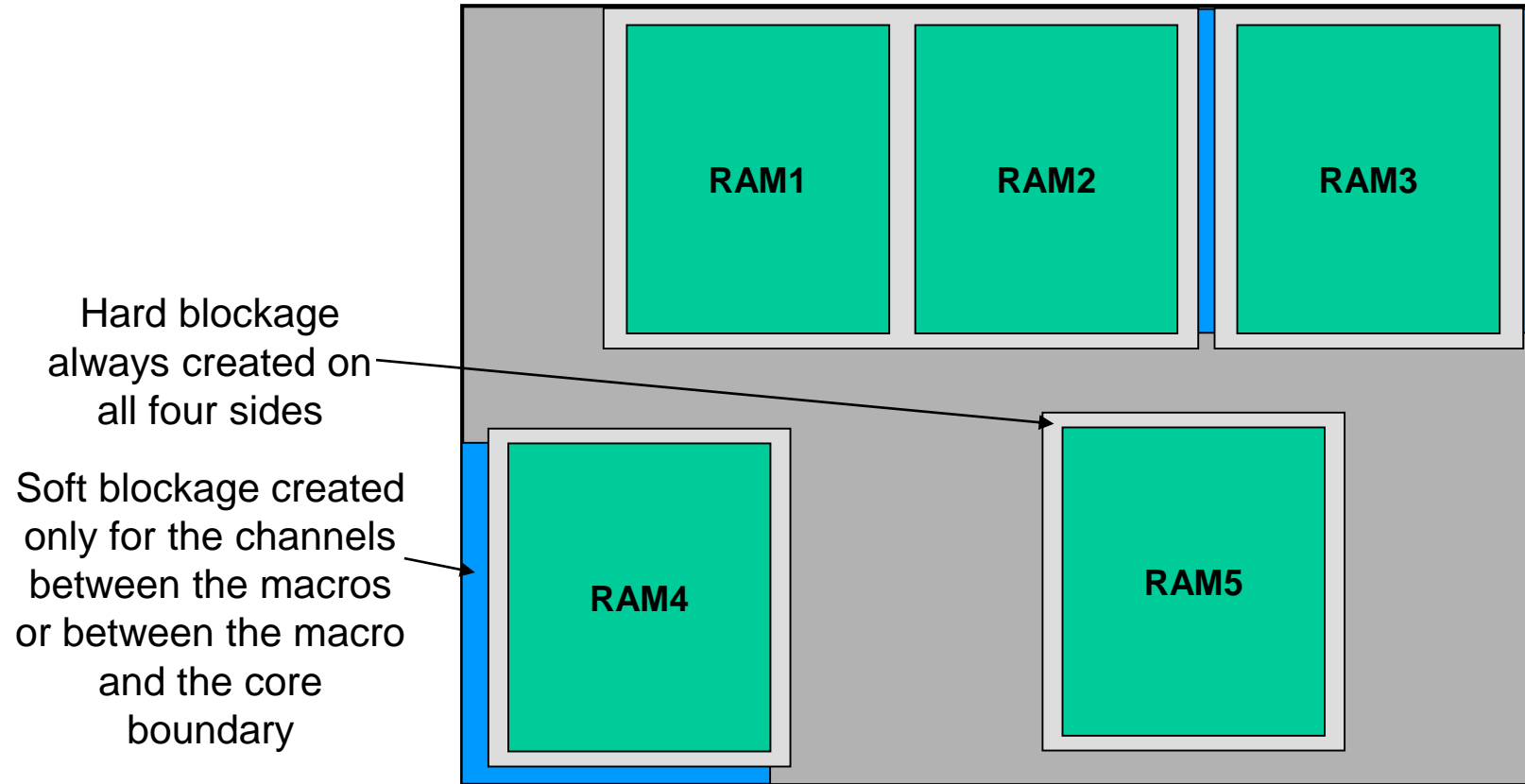
Legalize Cell Placement

Ensure that legal placement is done before saving the design.



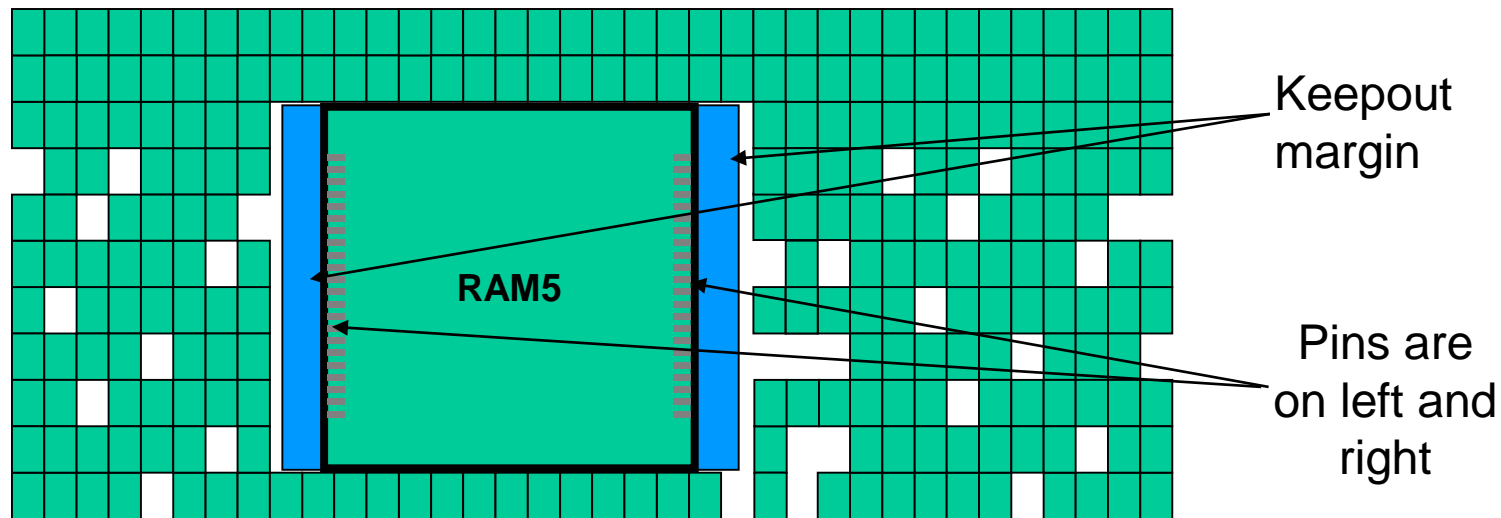
Legal placement of cells is not required for analyzing routing congestion at an early stage

Placement Blockages: Adding or Modifying Global Placement Blockages



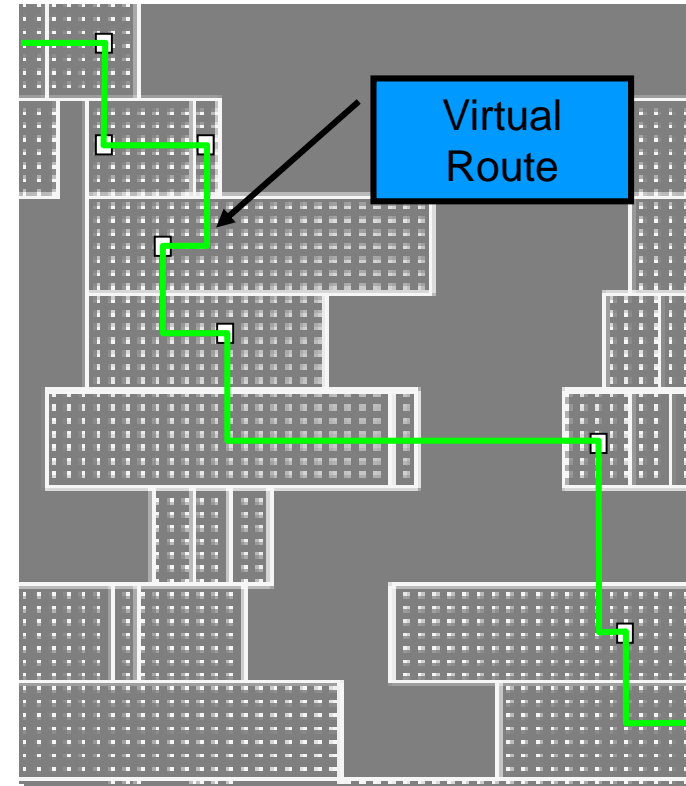
Placement Blockages: Macro Keepout Margin (Padding)

A keepout margin is a region around the boundary of fixed macros in the design in which no other cells are placed.



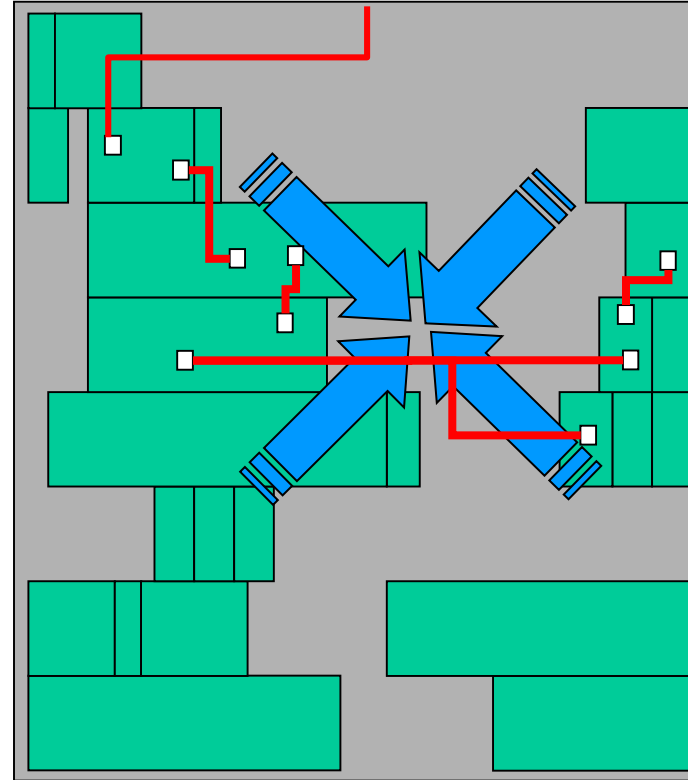
Timing-Driven Placement (1)

- All steps including placement are timing-driven
- Timing-driven placement tries to place critical path cells close together to reduce net RCs and to meet setup timing
- RCs are based on Virtual Route (VR)

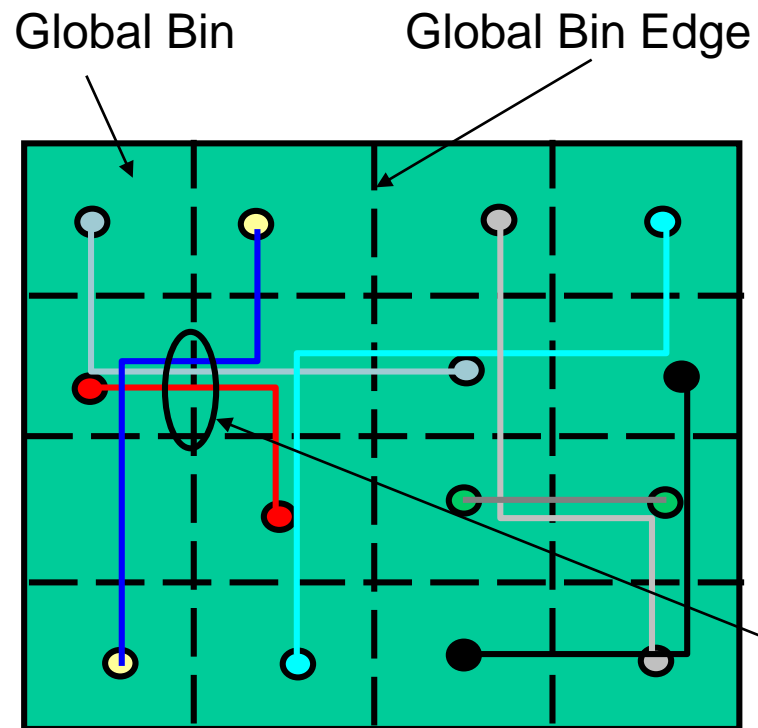


Timing-Driven Placement (2)

- Timing-driven placement based on Virtual Route
 - Tries to place cells along timing-critical paths close together to reduce net RCs and meet setup timing
 - Net RCs are based on Virtual Routing (VR) estimates



Congestion-driven Placement: Congestion

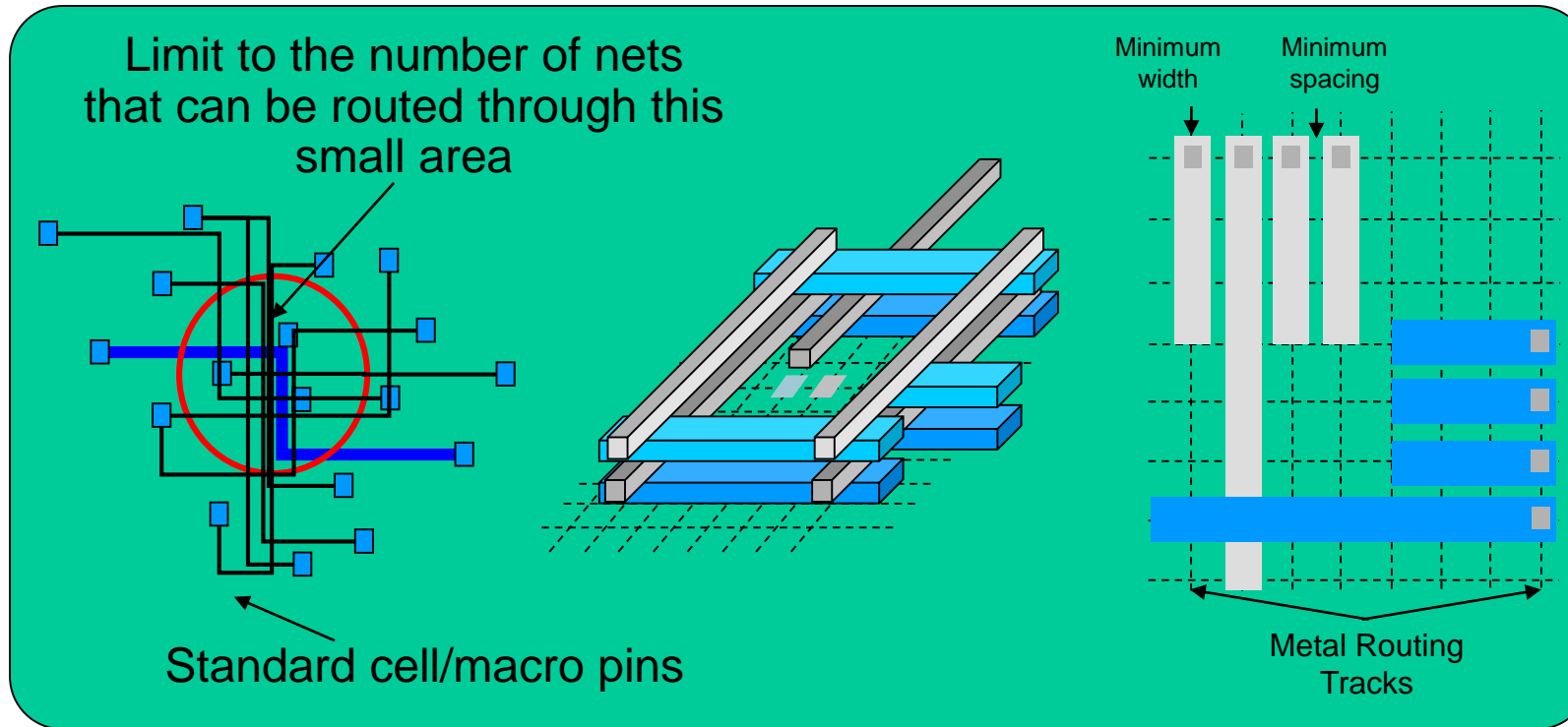


Overflow on each edge =
$$\begin{cases} \text{Routing Demand} - \text{Routing Supply} \\ 0 \text{ (otherwise)} \end{cases}$$

Total Overflow = $\sum_{\text{all edges}} \text{overflow}$

Routing demand = 3
Assume routing supply is 1,
overflow = $3 - 1 = 2$.

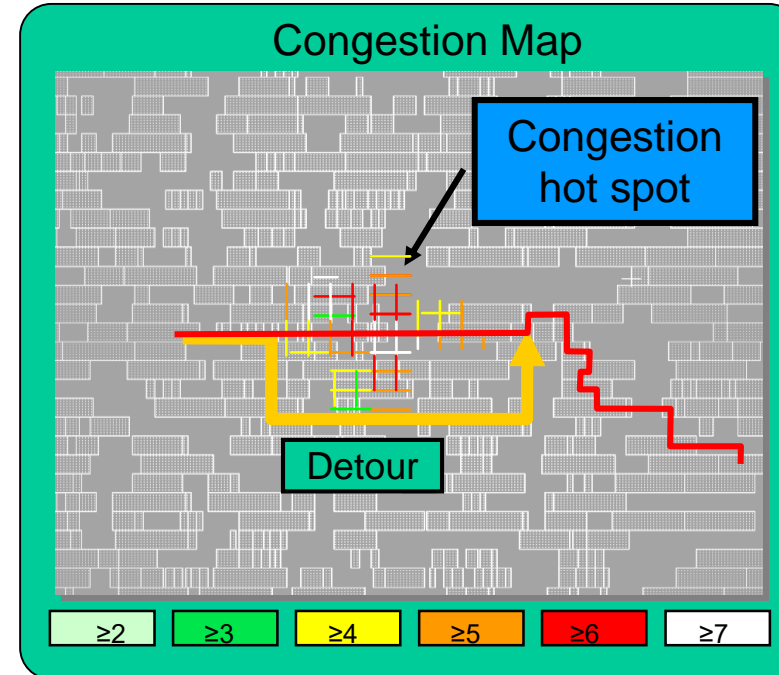
Congestion-driven Placement: Routing resource



When this limit is approached or exceeded, this area is said to be congested.

Placement Issues with Congestion

- If congestion is not too severe, the actual route can be detoured around the congested area
- The detoured nets will have worse RC delay compared to the VR estimates



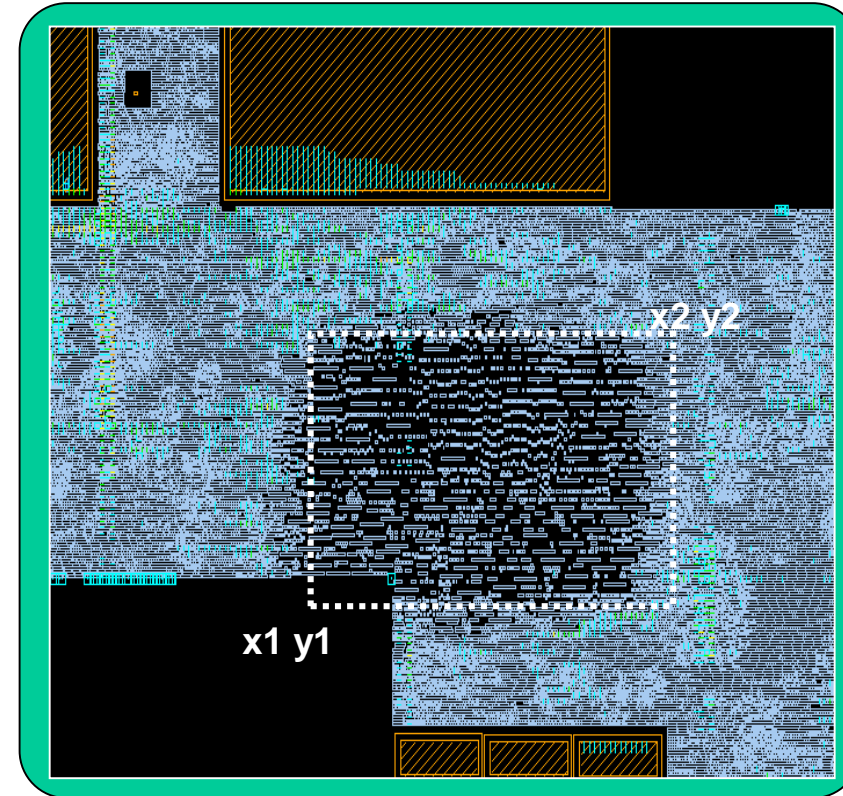
In highly congested areas, delay estimates during placement will be optimistic.

Fix Congestion: Modify Floorplan

- Top-level ports
 - Changing to a different metal layer
 - Spreading them out, re-ordering or moving to other sides
- Macro location or orientation
 - Alignment of bus signal pins
 - Increase of spacing between macros
- Core aspect ratio and size
 - Making block taller to add more horizontal routing resource
 - Increase of the block size to reduce overall congestion
- Power grid: Fixing any routed or non-preferred layers

Modifying Physical Constraints: Cell Density

- Cell density can be up to 95% by default
 - Density level can also be applied to a specific region
- Lower cell density in congested areas using coordinate option



Clock Tree Synthesis (CTS)

How to route the clock?

Physical Synthesis Flow

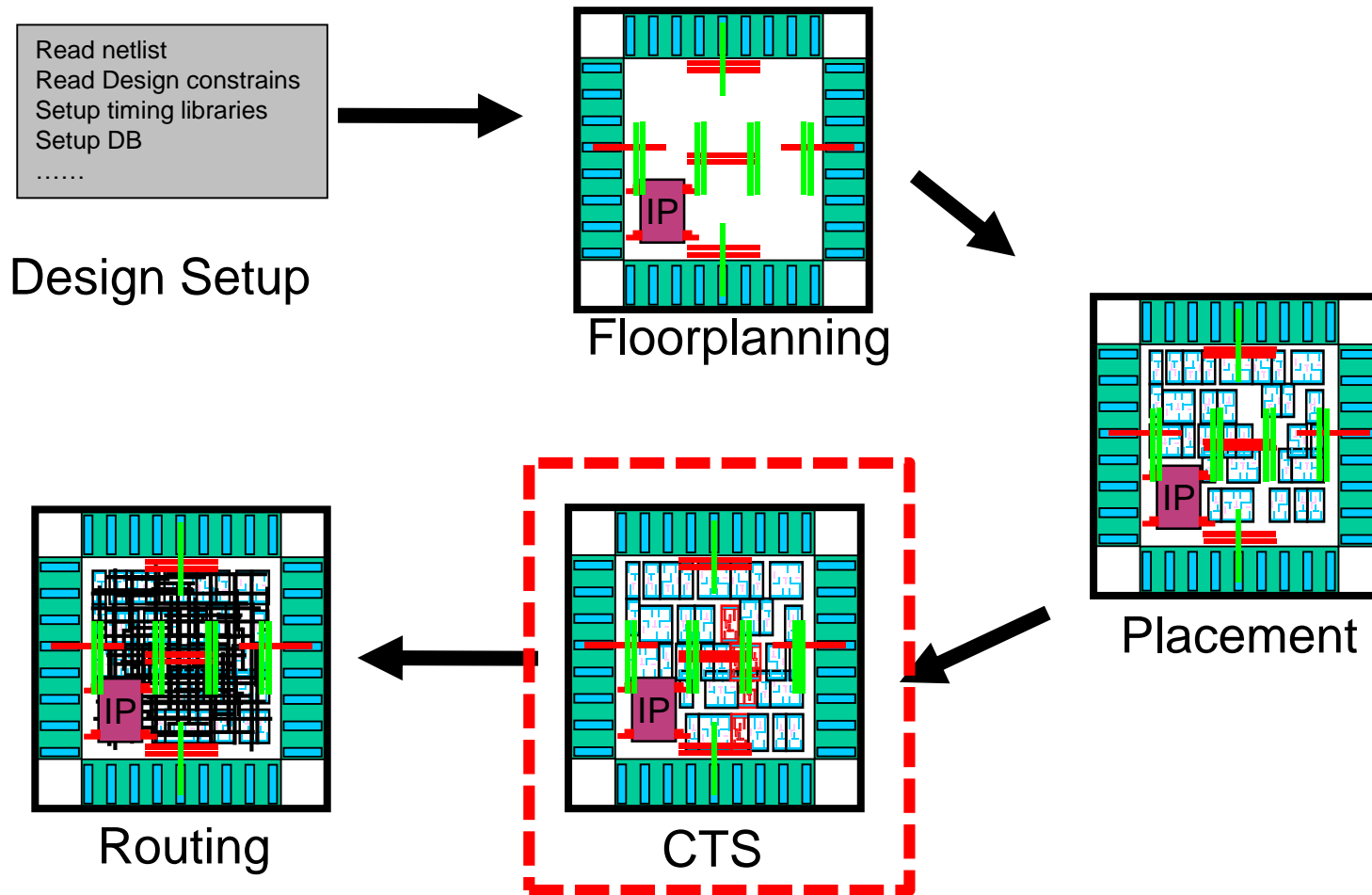
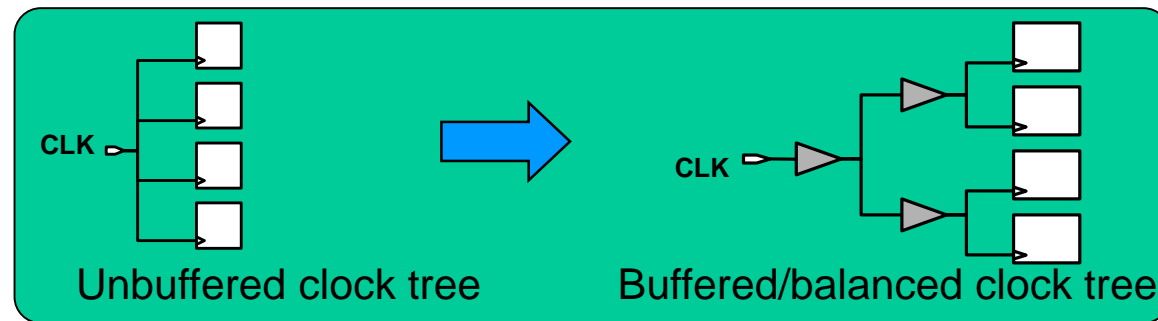


Figure: Synopsys

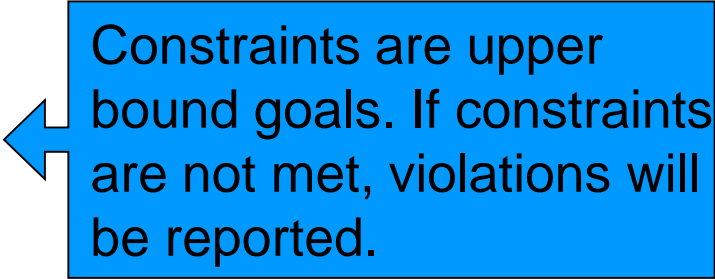
Clock Tree Synthesis (CTS) Problem

- CTS Problem
 - **CTS is the process of distributing clock signals to clock pins based on physical/layout information**
 - After placement of cells the tree of synchronization is synthesized
 - Balanced clock tree is synchronized with the addition of buffers
 - After routing CT optimization is made



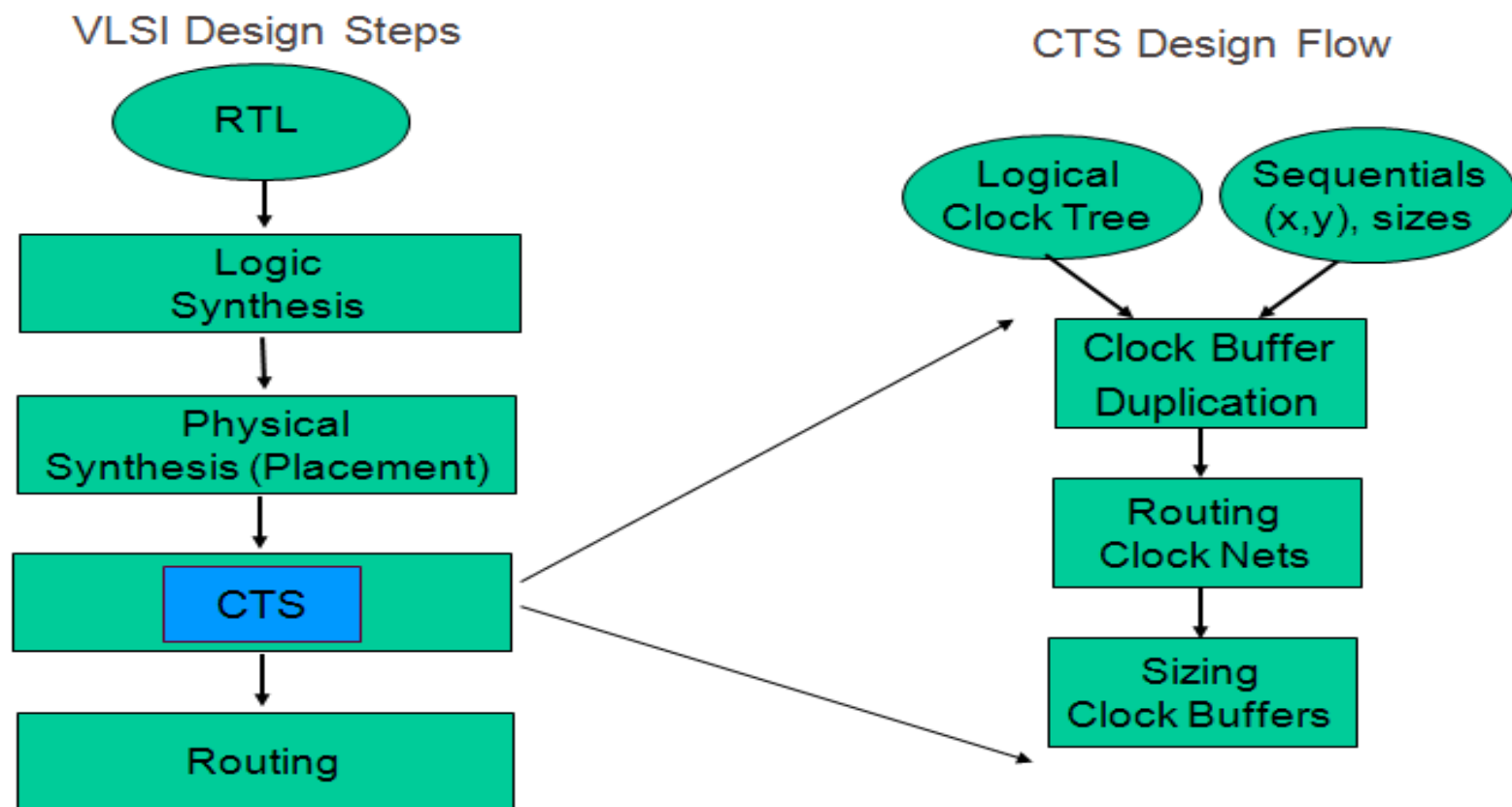
CTS Problem: Clock Tree Synthesis Goals

- Meeting the clock tree design rule constraints
 - Maximum transition delay
 - Maximum load capacitance
 - Maximum fanout
 - Maximum buffer levels
- Meeting the clock tree targets
 - Maximum skew
 - Min/Max insertion delay



Constraints are upper bound goals. If constraints are not met, violations will be reported.

CTS Problem

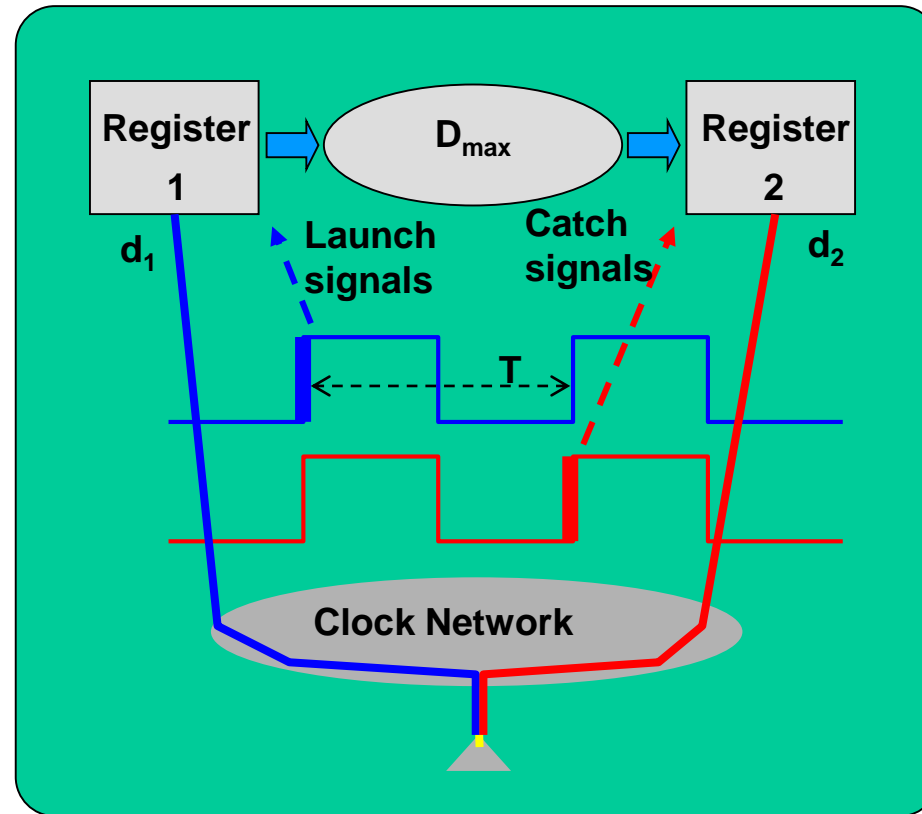


Clock Tree: General Concepts: Clock Distribution Network

$$\text{Skew} = d_1 - d_2$$

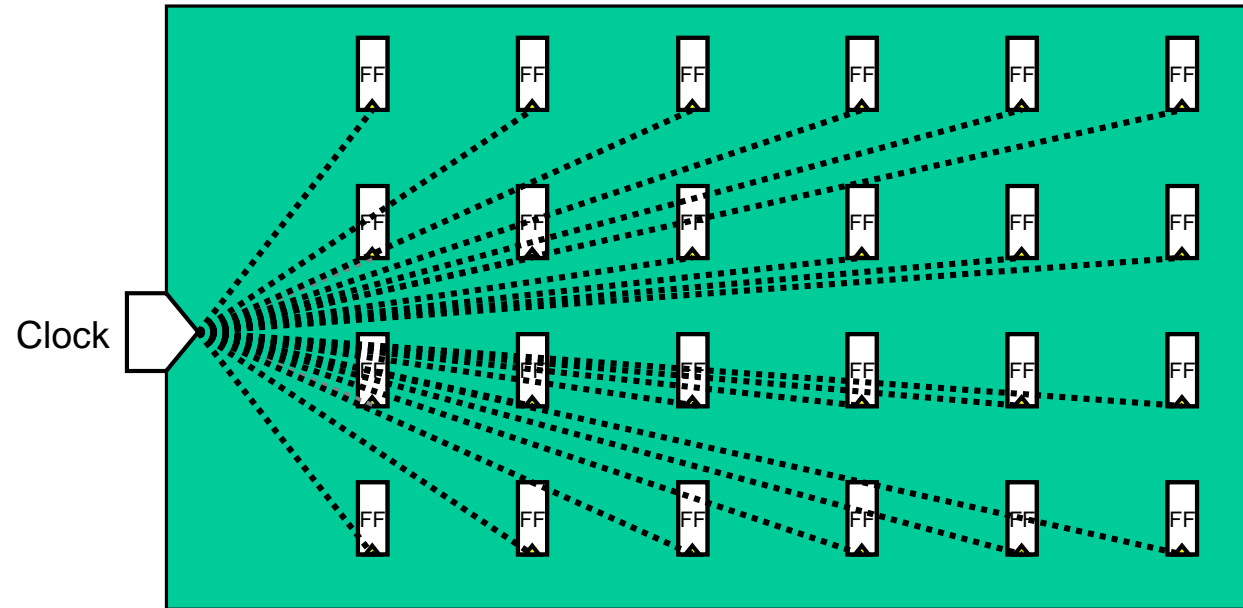
$$\text{Zero skew: } d_1 = d_2$$

$$\text{Useful skew, } d_1 - d_2 = \delta_{12}$$

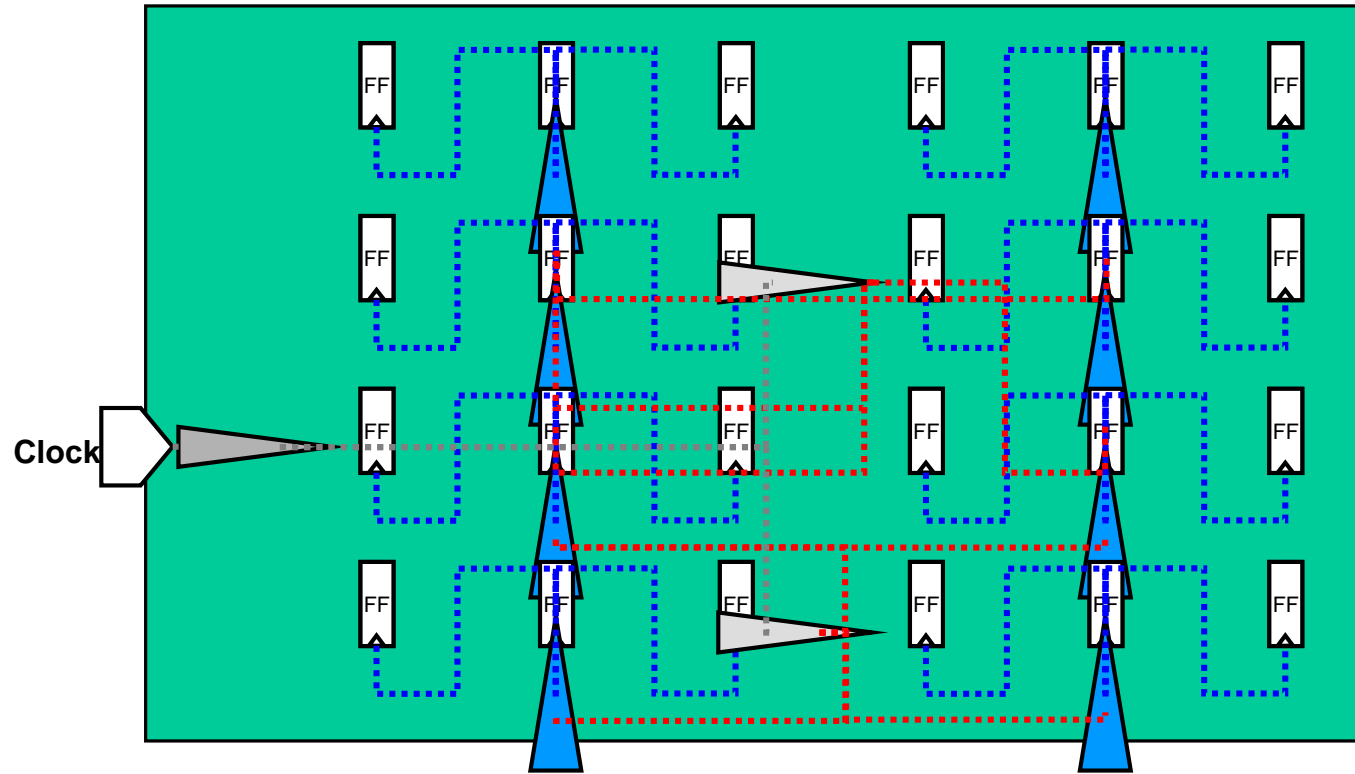


CTS Problem: Starting Point Before CTS

- All clock pins are driven by a single clock source
- All clock pins are from a source of clock pulses in various geometrical distances

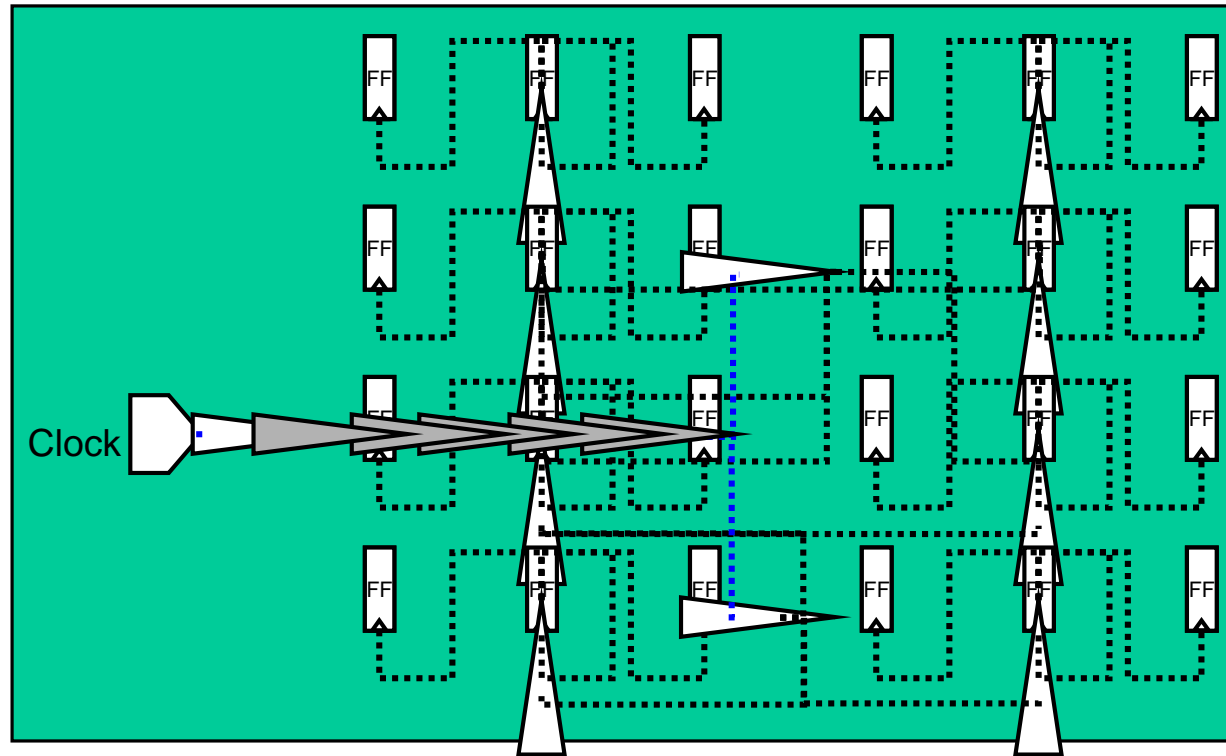


Generated and Gated Clocks: CTS (1)



A buffer tree is built to balance the loads and minimize the skew

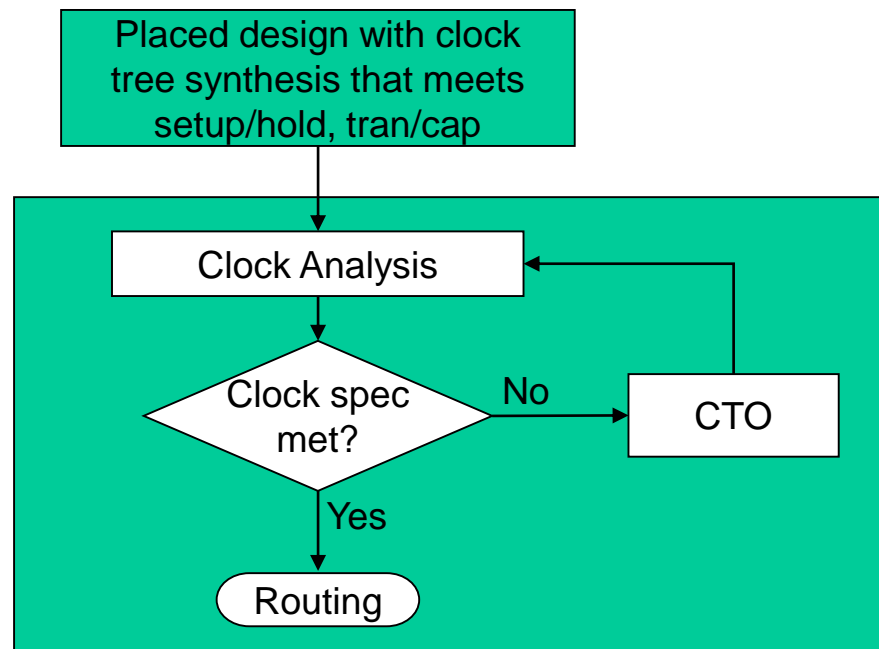
Generated and Gated Clocks: CTS (2)



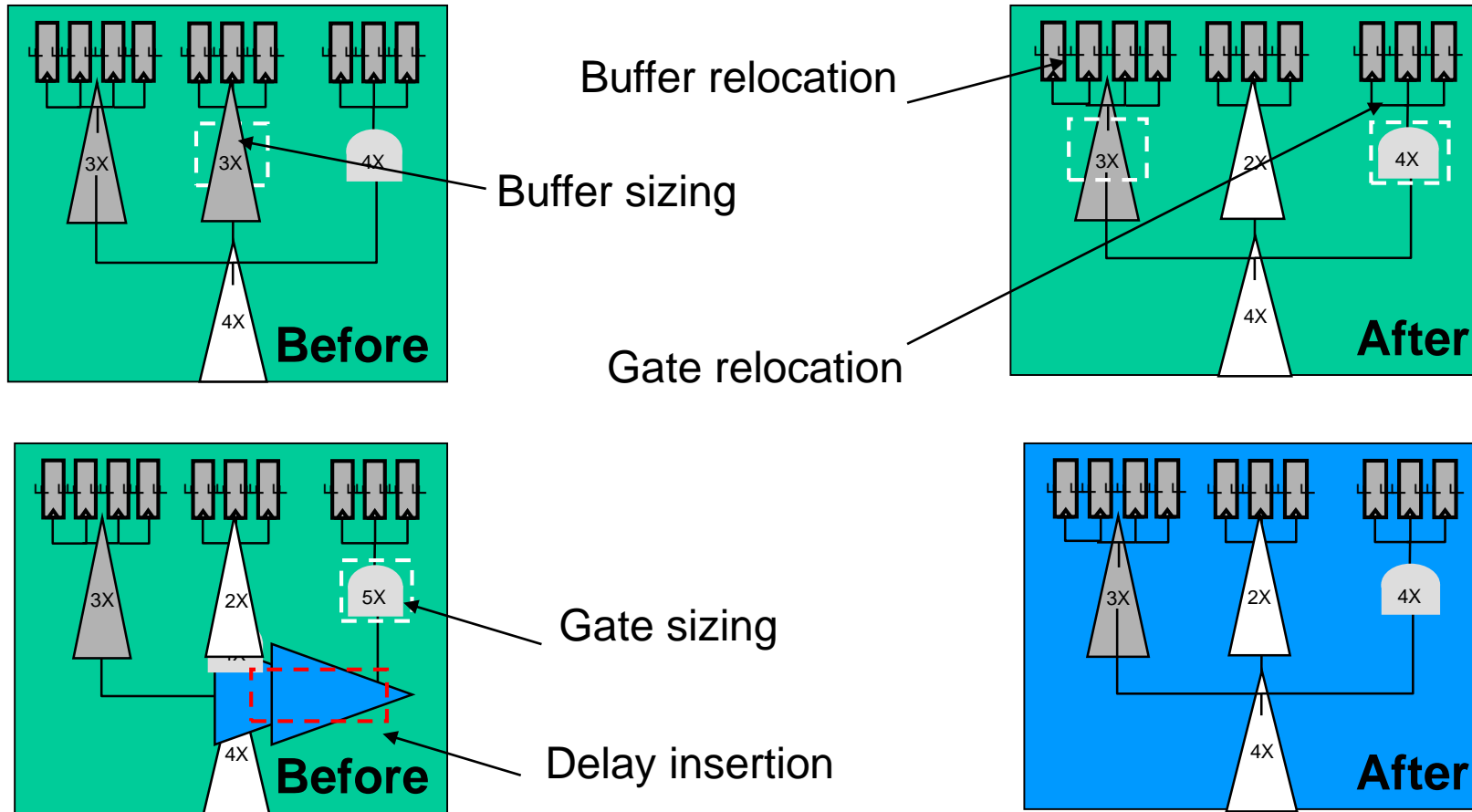
A delay line is added to meet the minimum insertion delay

Clock Tree Optimization (CTO)

Performing additional Clock Tree Optimization as necessary to further improve clock skew



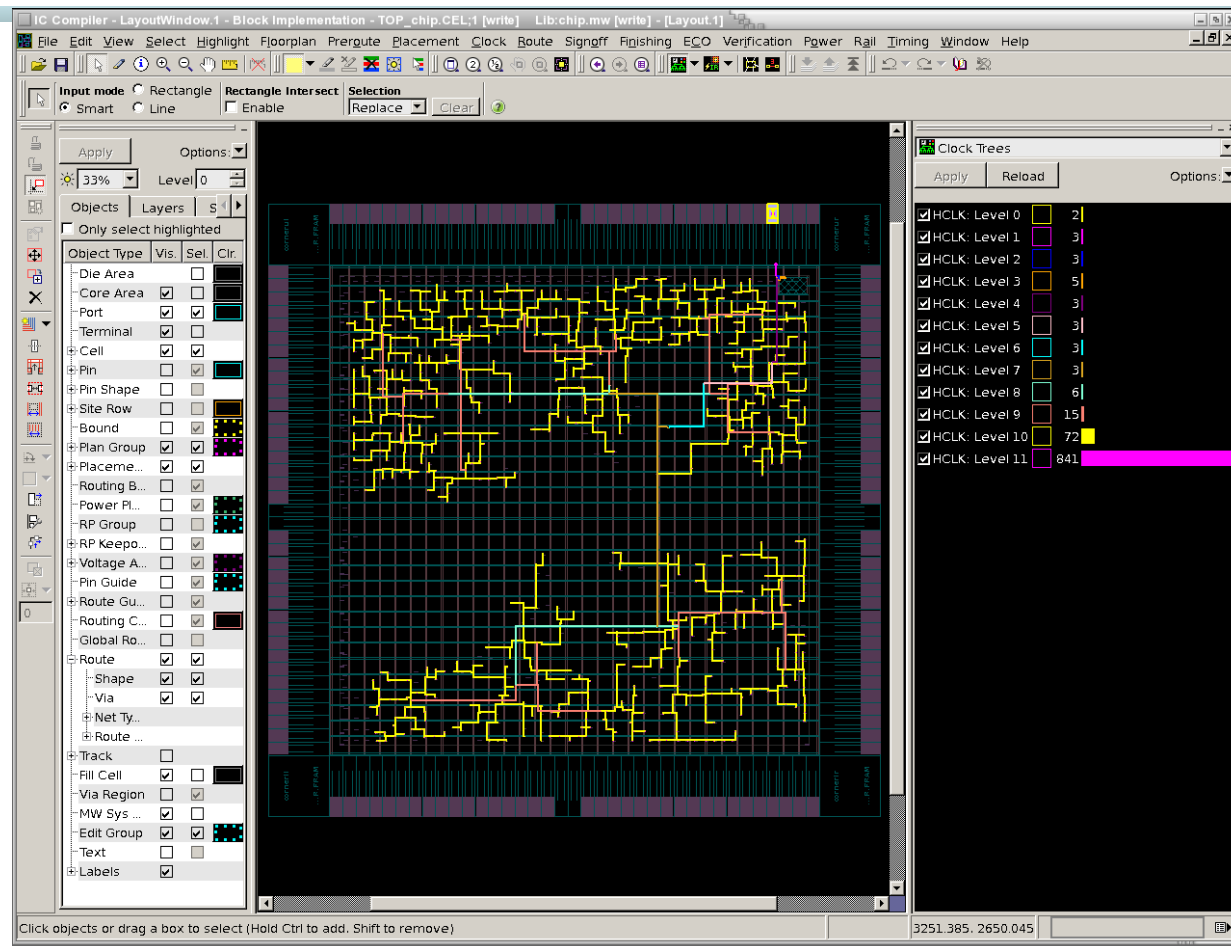
Clock Tree Optimization Options



CTS Problem: Prerequisites for Clock Tree Synthesis

- Before running CTS, the design must meet the following requirements:
 - The design should be placed and optimized
 - Placement – completed; Power and ground nets – prerouted
 - Estimated congestion – acceptable
 - Estimated timing – acceptable (~0ns slack)
 - Estimated max cap/transition – no violations
 - High fanout nets
 - Reset, Scan Enable synthesized with buffers
 - Clocks are still not buffered

Clock Tree



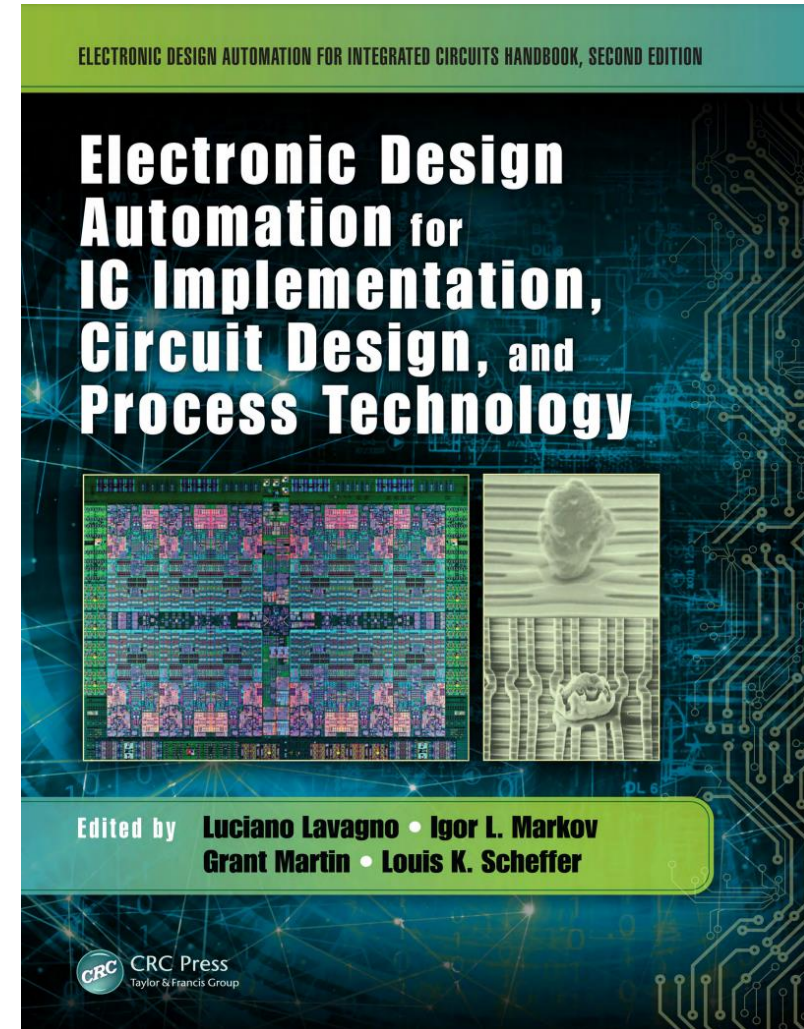
Design view after of CTS

Where are we Heading?

- ASIC Design Flow III

Action Items

- Lab #4 is due!
- Reading Materials
 - Slides
 - Ch. 1 & 2 of Electronic Design Automation for IC Implementation, Circuit Design, and Process Technology (on canvas)



Acknowledgement

Slides in this topic are inspired in part by material developed and copyright by:

- Synopsys Courseware
- Prof. Christopher Batten @ Cornell University