# STUDY OF PHYSICAL DESIGN FLOW, CHALLENGES AND ITS SOLUTIONS

A DISSERTATION

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE AWARD OF THE DEGREE OF

MASTER OF TECHNOLOGY
IN
**VLSI DESIGN & EMBEDDED SYSTEMS**


Submitted by:

**LOKESH GAUTAM**
**2K17/VLS/10**

Under the supervision of

Dr. Devanand
Assistant Professor




**ELECTRONICS & COMMUNICATION ENGINEERING**
DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering) Bawana Road,
Delhi-110042

**2017-2019**

# ELECTRONICS & COMMUNICATION ENGINEERING

## DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College of Engineering)

Bawana Road, Delhi-110042

## CANDIDATE'S DECLARATION

I, (LOKESH GAUTAM), Roll No. 2K17/VLS/10 student of M.Tech (VLSI & Embedded systems), hereby declare that the project Dissertation titled "Study of Physical Design flow, Challenges and its Solutions" which is submitted by me to the Department of Electronics and Communication Engineering, Delhi Technological University, Delhi in partial fulfillment of the requirement for the award of the degree of Master of Technology, is original and not copied from any source without proper citation. This work has not previously formed the basis for the award of any Degree, Diploma Associateship, Fellowship or other similar title or recognition.

Place: Delhi **(Lokesh Gautam)**

Date:

# ELECTRONICS & COMMUNICATION ENGINEERING

## DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College of Engineering) Bawana Road,
Delhi-110042

## CERTIFICATE

I hereby certify that the Project Dissertation titled "Study of Physical Design Flow, its Challenges and Solutions" which is submitted by Lokesh Gautam, 2k17/vls/10, Electronics & Communication Engineering, Delhi Technological University, Delhi in partial fulfillment of the requirement for the award of the degree of Master of Technology, is a record of the project work carried out by the student under my supervision. To the best of my knowledge this work has not been submitted in part or full for any Degree or Diploma to this University or elsewhere.

Place: Delhi

Date:

**(Dr. Devanand)**
Assistant Professor
Dept. of ECE, DTU

# Acknowledgments

Every endeavor needs deft guidance and helping hands for its accomplishment, and I was fortunate to have both. I wish to express my indebtedness to my mentors, Dr. Devanand and Mr. Rahul Govindaswamy, without whose guidance, this work would not be possible. I would also like to mention Prof. S. Indu for her constant support throughout. I would like to thank my internal guide at DTU, Dr. Devanand, for his kind guidance, support, motivation and for providing unconstrained freedom to explore. I wish to express my gratitude to my mentor and manager at Qualcomm, Mr. Rahul Govindaswamy, who was always lucid in his explanations, dexterous in his guidance, and generous with his care and support.

I am thankful to Qualcomm, Bangalore, for providing the internship opportunity, which is the backbone of this work. I would like to thank the team members of SoC Physical design WTR team at Qualcomm.

Finally, I would like to thank my parents, family, and friends, for bestowing their love, patience, understanding, and care, which kept me going.

Date: **Lokesh Gautam**

# ABSTRACT

Physical design is all about placing instances defined in the netlist and connecting them by routing through metal layer stack to satisfy design specifications such as performance, power and area (PPA). Current IC designs have multi million instances that are interconnected with several stack of metal layers that connect these instances. Manually performing each step in the design process is not feasible, takes huge amount of time and is error prone.

The complexity in designing a multi-million instance based IC is huge and hence we need dedicated automation flows that complete specific tasks needed to be performed at each step in the design which reduces design time and errors. These flows require knowledge and understanding of EDA tools and scripting languages such as Tcl, Perl or Python.

In addition to complexity, as time to market for chips is decreasing, reuse of IP (Intellectual Property) blocks is highly preferred in each design.

Inputs to physical design are the most important files that you will need to start your design process. If the inputs are read in the EDA tools without any issues (warnings and errors), then your physical design flow goes smooth.

In today's IC design, because of huge design complexity, hierarchical design approach is followed. What this means is, the entire chip is divided into partitions or blocks that are interconnected at a top level module.

Physical Design is all about optimization. Thus, floorplanning is a highly iterative process which takes into account the hard blocks and soft blocks used, memories, IO pads and their placement in the design, routing possibilities between different blocks and inside the blocks, power grid structure for each macro and cell in the design, and also the aspect ratio and IO structure of the entire design.

# CONTENTS

# List of Tables

# List of Figures

Fig. 7.9: Shorter routing wires of sub circuit of hm of core logic

# CHAPTER 1 : INTRODUCTION

The Electronic Design Automation (EDA) industry develops software to support engineers in the creation of new integrated-circuit (IC) designs. Due to the high complexity of modern designs, EDA touches almost every aspect of the IC design flow, from high level system design to fabrication. EDA has made it possible to create high-density interconnect (HDI) circuits.

In this section, well see RTL2GDS flow used in industry and design inputs required at each stage in the flow. There are two kind of input in which the flow starts, one is technology inputs and other is design inputs. Technology inputs are given by respective foundries and design inputs are given by RTL team.

Technology Inputs :

• Standard Cell Library (.lib file): This file is in textual format. This library contains timing, area and power information for all the standard cells.

• Technology file (.lef file (Library Exchange Format)): This file contains the physical properties of the fabrication process. For example, it contains the number of metal layers, the design rules, resistances, etc. It also includes information like routing direction, spacing between 2 metal layers, etc

Given the significance of placement in integrated circuit (IC) physical design, extensive research studies performed over the last 50 years addressed numerous aspects of global and detailed placement [1]. The objectives and the constraints dominant in placement have been revised many times over, and continue to evolve. Additionally, the increasing scale of placement instances affects the algorithms of choice for high-performance tools. We survey the history of placement research, the progress leading up to the state of the art, and outstanding challenges

In literature survey we discuss about history of physical design flow in past few years. Physical design starts with the partitioning. After partitioning next step is floorplanning of the chip. We have core and die area in the chip. Core is the area where our main logic sits. Pre placed cells are cells that have got certain functionality for eg. Adder, multiplier etc. We call them as macros and IPs as well. Locations of pre placed cells are defined according to their connectivity. Locations are based on the netlist.

Chapter 2 discuss about literature survey of physical design flow. It starts with floorplanning steps which starts which defining width and height of core and die area. This is preceeded by placement of pre placed cells. Further chapters discuss about remaining physical design flow. We conclude this work in Chapter 8 with possibility of future scope.

# CHAPTER 2 : LITERATURE SURVEY

## 2.1 FLOORPLANNING STEPS

In this section all the basic steps in floorplanning are being followed. First of all we define the width and height of core and die area. Core is the area where our main logic sits. After that all our preplaced cells are given specific positions which are decided based on application. These steps are followed by decoupling capacitor placement, power planning, pin placement and logical cell placement blockage.

### 2.2.1 Define Width and Height of Core and Die

Core is the area where our main logic sits.

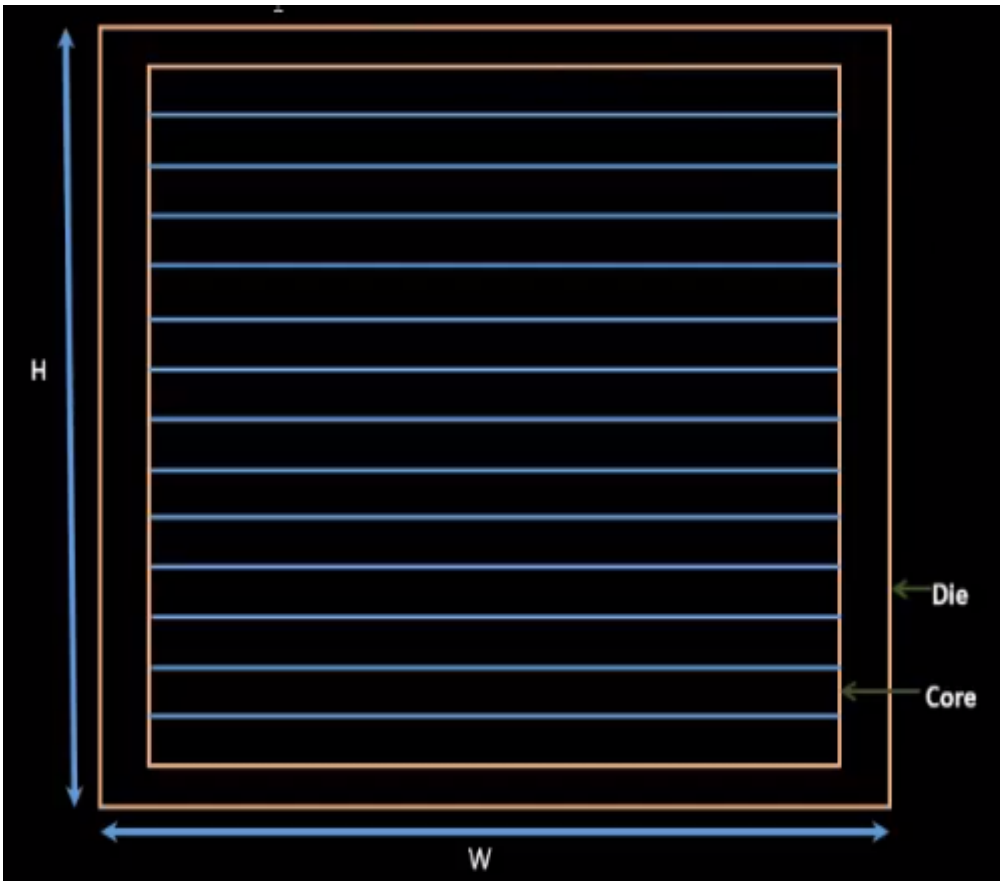



Fig. 2.1 H and W of core and die area [1]

### 2.2.2 Define Locations of Preplaced Cells

Preplaced cells are cells which have got certain functionality for eg. Adder, multiplier etc. We call them as macros and IPs as well. Locations of preplaced cells are defined according to their connectivity. Locations are decided based on complete netlist [2].

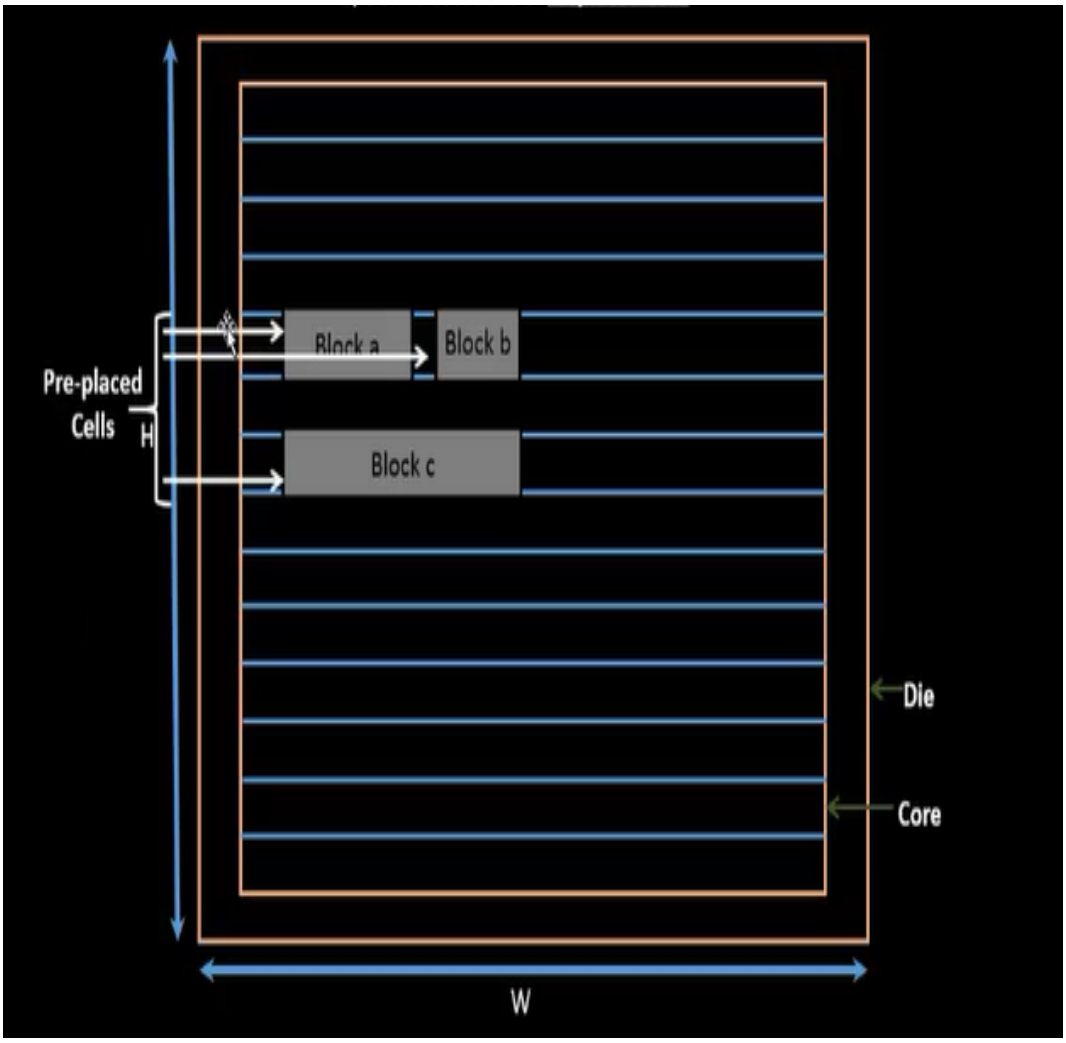



Fig. 2.2 Pre placed cells locations [2]

### 2.2.3 Surround pre placed cells with Decoupling capacitors

When pre placed blocks change logic from 0 to 1 then there is huge amount of current demand which our power supply is not able to cater to [3]. So we place decoupling capacitors near our pre placed cells to cater to their power needs. For rest of the time decoupling capacitors get charged from our main power supply.

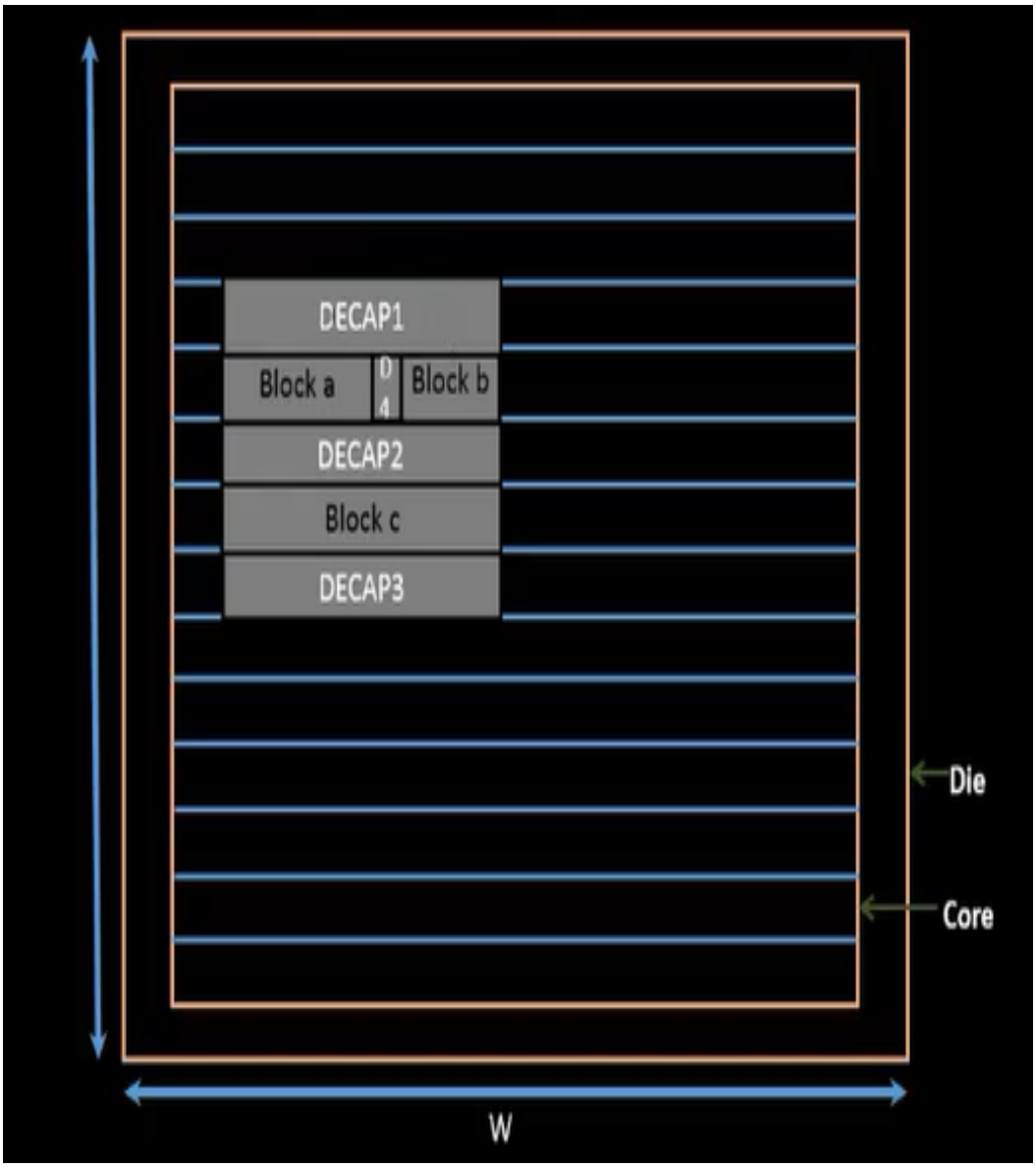



Fig. 2.3 De-Cap cells insertion [2]

### 2.2.4 Power Planning

We create a mesh kind of structure for power supply. Instead of having one source of power supply ,we have multiple sources of power supply. Any cell taps power from its nearest Vdd and ground.

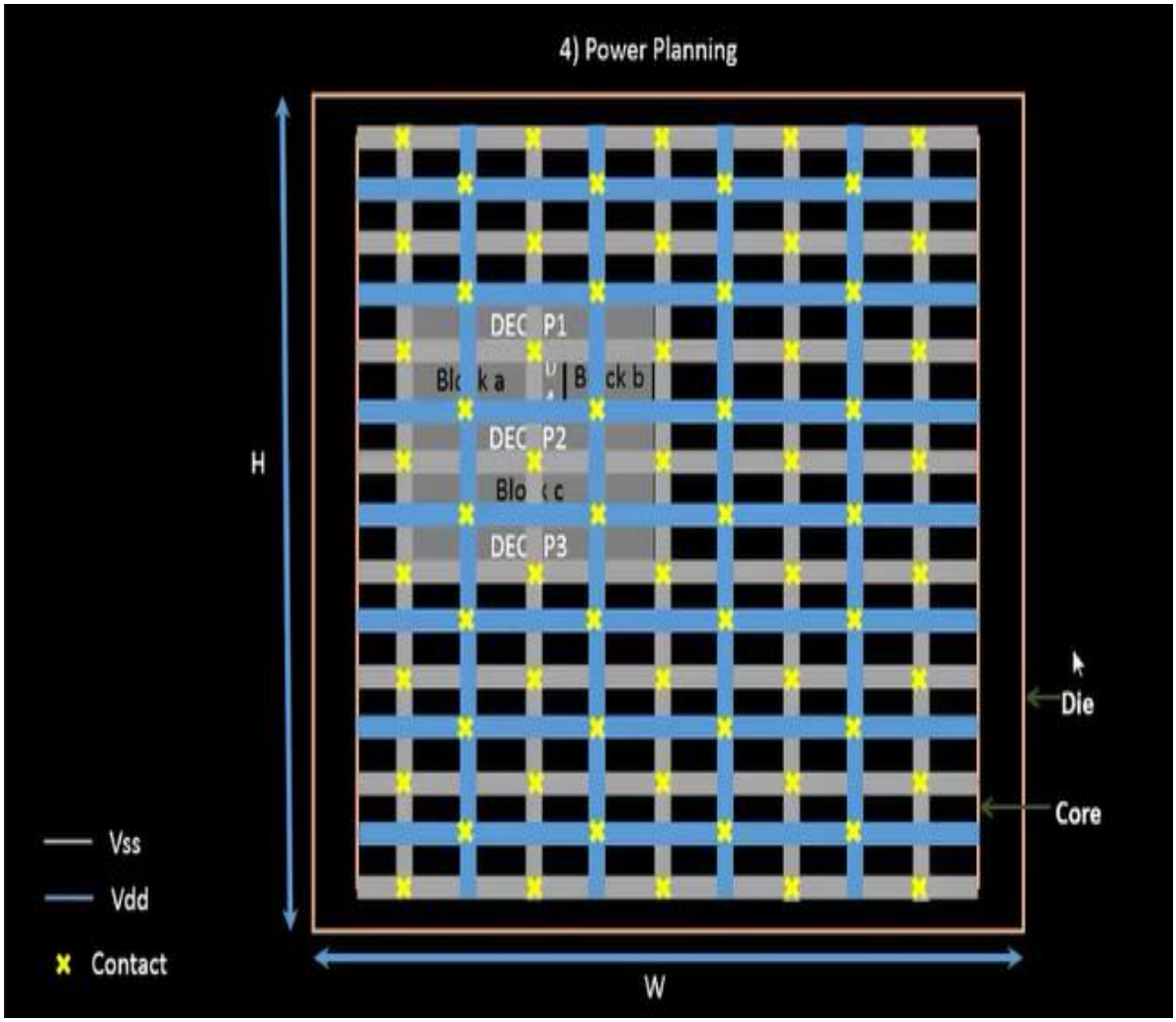



Fig. 2.4 Power Planning [4]

### 2.2.5 Pin Placement

We have to identify input and output pins of the circuitry and place them in the area between core and die.

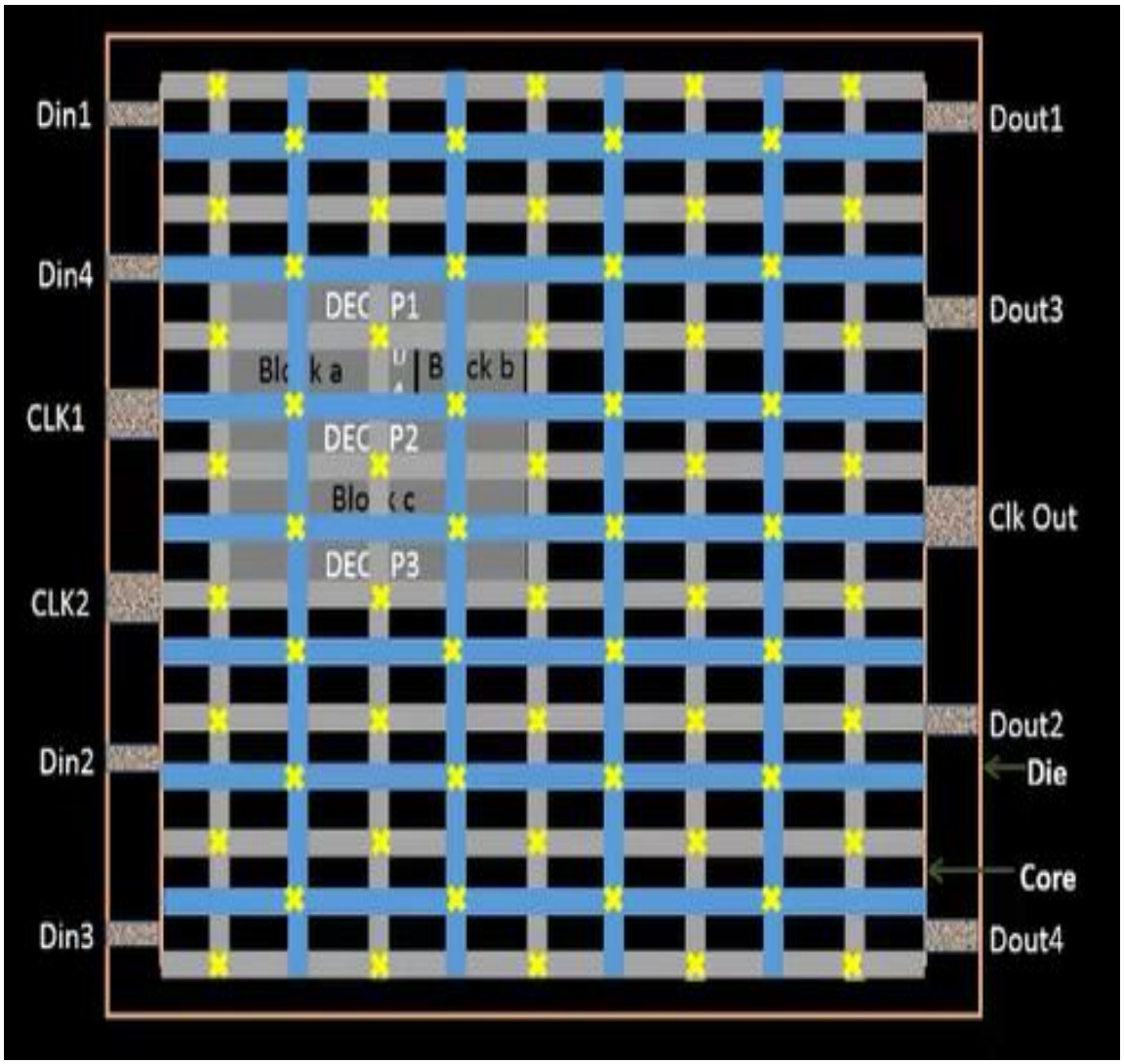



Fig. 2.5 Pin Placement [4]

### 2.2.6 Logical Cell Placement Blockage

We place the blockage in the area between the core and die which ensures there will be no logic cells accidentally placed in this area [5]. This is the final step of floorplan. Now our floorplan is ready for placement and routing step.

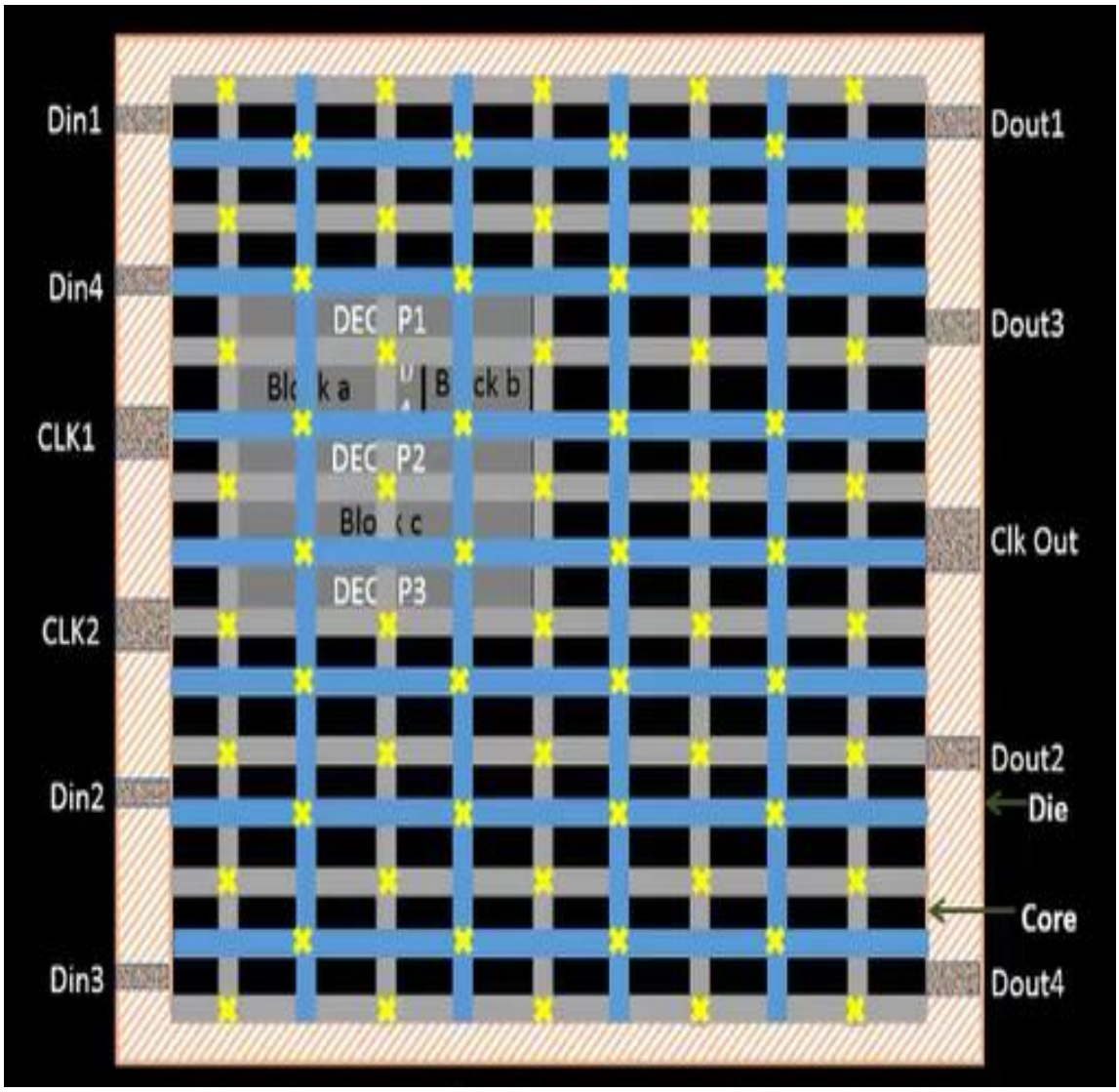



Fig. 2.6 Logical cell placement blockage [5]

## 2.3 PLACE AND ROUTE

### 2.3.1 Bind Netlist with physical cells

To make our design physically realisable, we bind netlist with physical cells [6]. It is the process of binding logical netlist in a physical form.

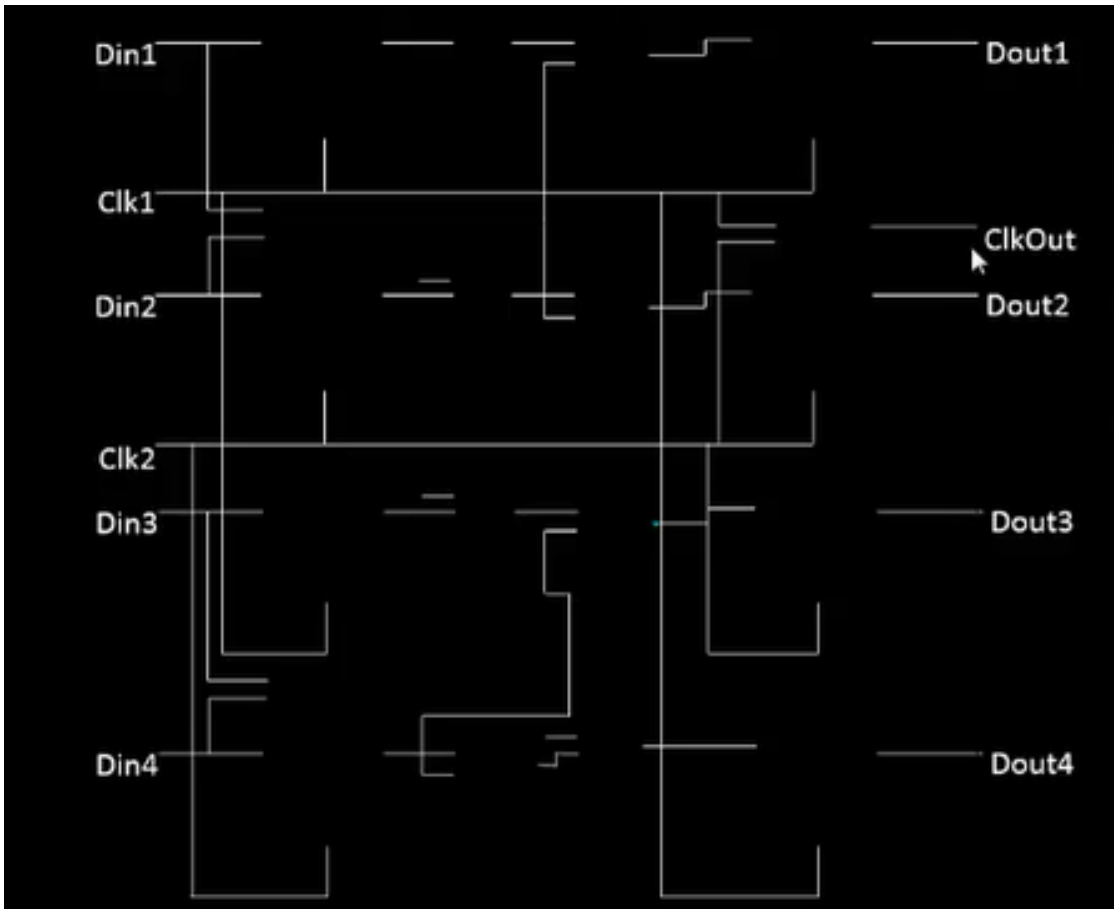



Fig. 2.7 Netlist provided by design team [6]

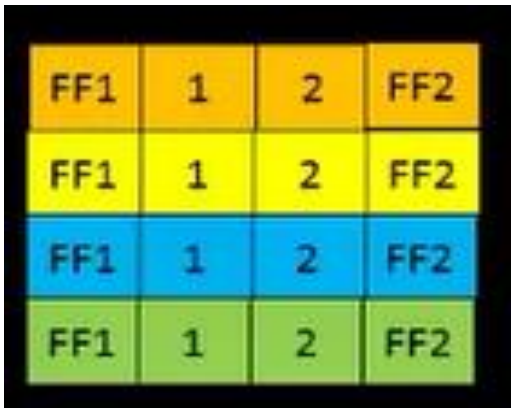



Fig. 2.8 Library of physical cells [6]

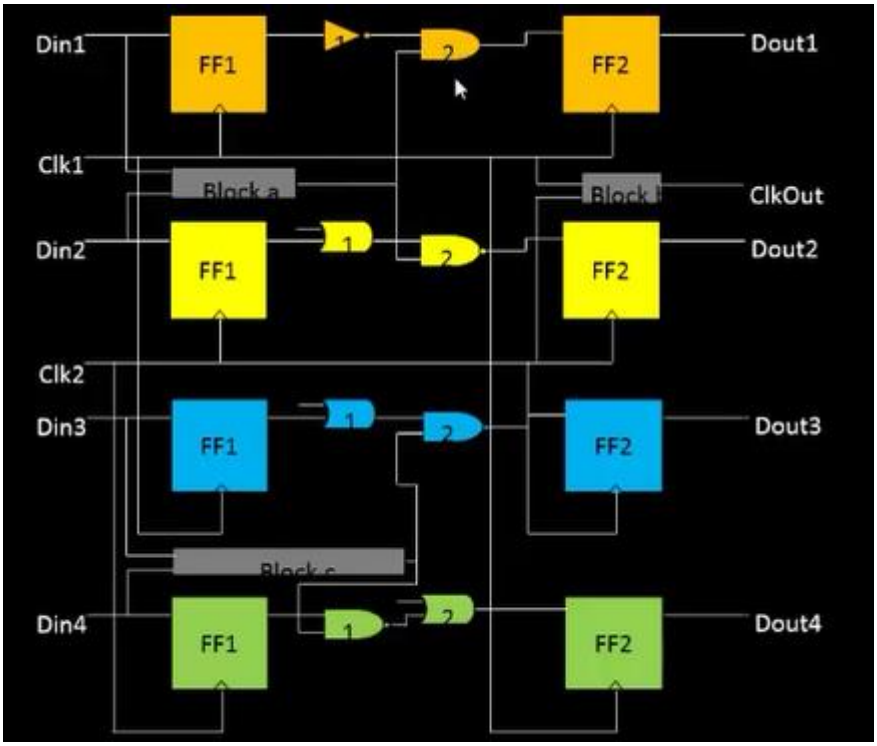



Fig. 2.9 Binding netlist with physical cells [6]

### 2.3.2 Placement

Once we have shapes and sizes of standard cells, next step is to place these cells on the chip. Flip Flop 1 is close to Din1 so it makes complete sense to place FF1 near to Din1. Placement is done in such a manner that there should not be any degradation from input signal to standard cell [7].

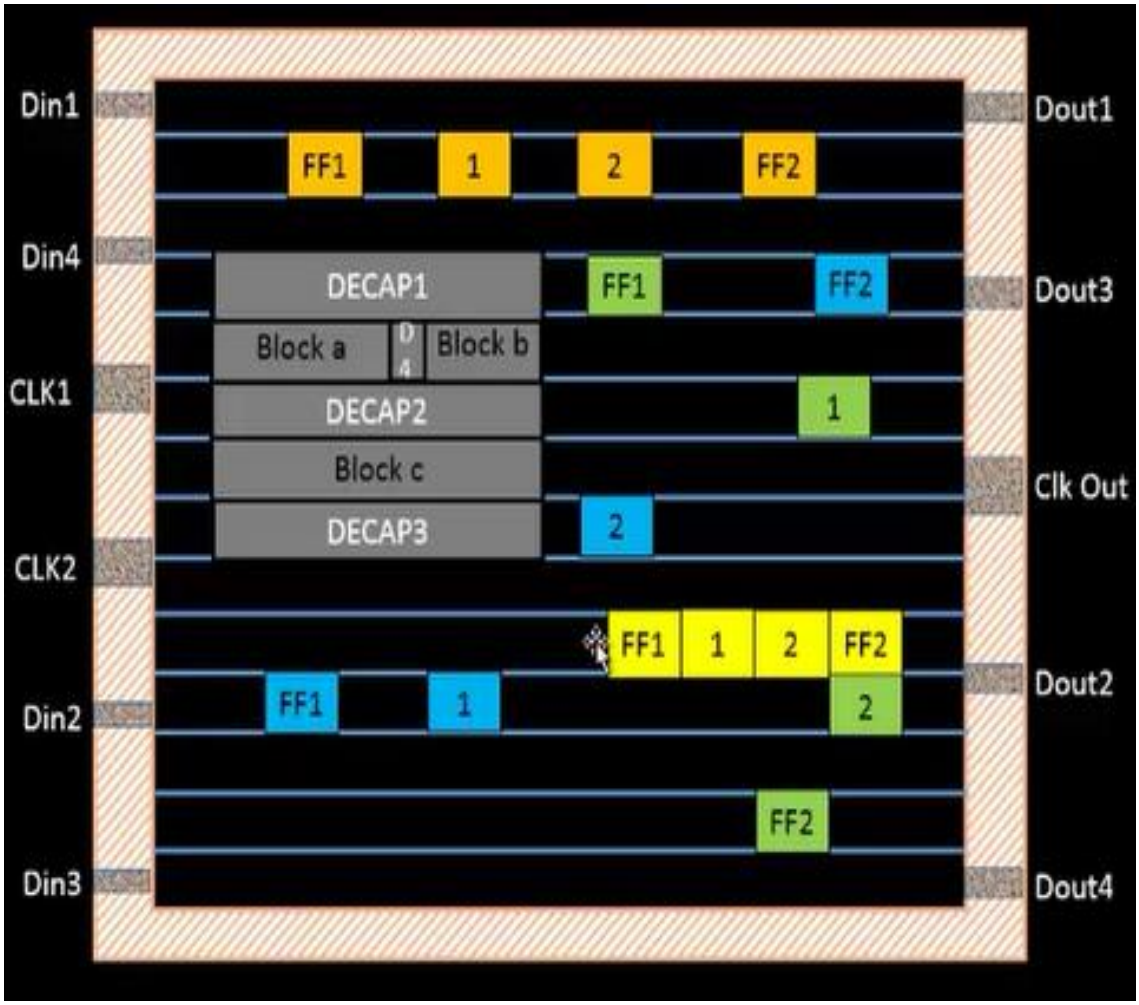



Fig. 2.10 Placement of standard cells [7]

### 2.3.3 Optimize Placement

This is the stage where we estimate wire length and capacitance and based on that we insert repeaters. Repeaters are basically signal conditioner [7]. They reconstruct the original signal and send it ahead. If estimates wire length is small, there is no need to use repeaters as degradation won't be much significant in this case.

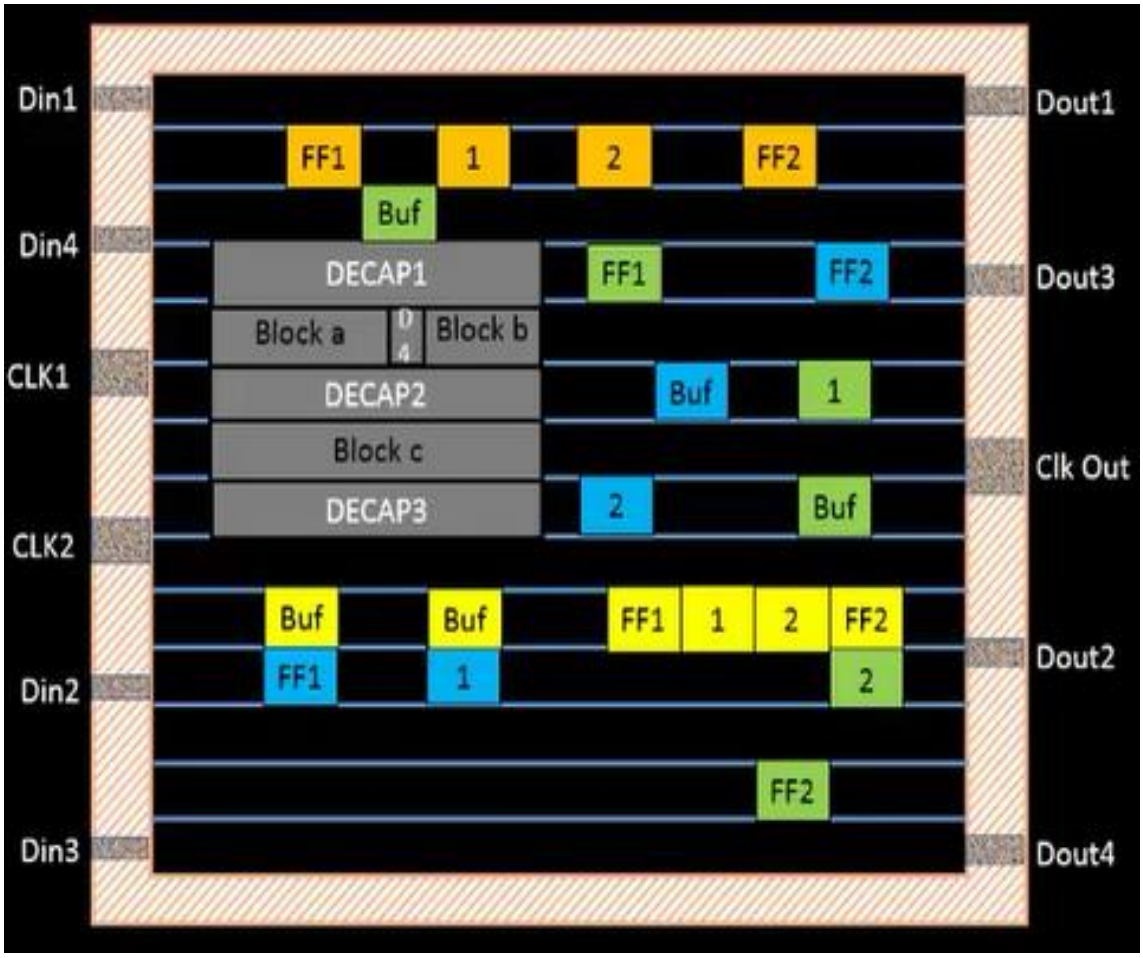



Fig. 2.11 Optimize placement [7]

## 2.4 PLACEMENT TIMING AND CLOCK TREE SYNTHESIS

### 2.4.1 Timing Analysis (with Ideal Clocks)

The next step after the optimize placement stage is to do timing analysis with ideal clock to check how does timing looks like at this stage. If there are any accidental wrong placement done by user or by tool that can be fixed right here. Ideal clock means clock latency is zero. We do timing analysis only on the data path [8].

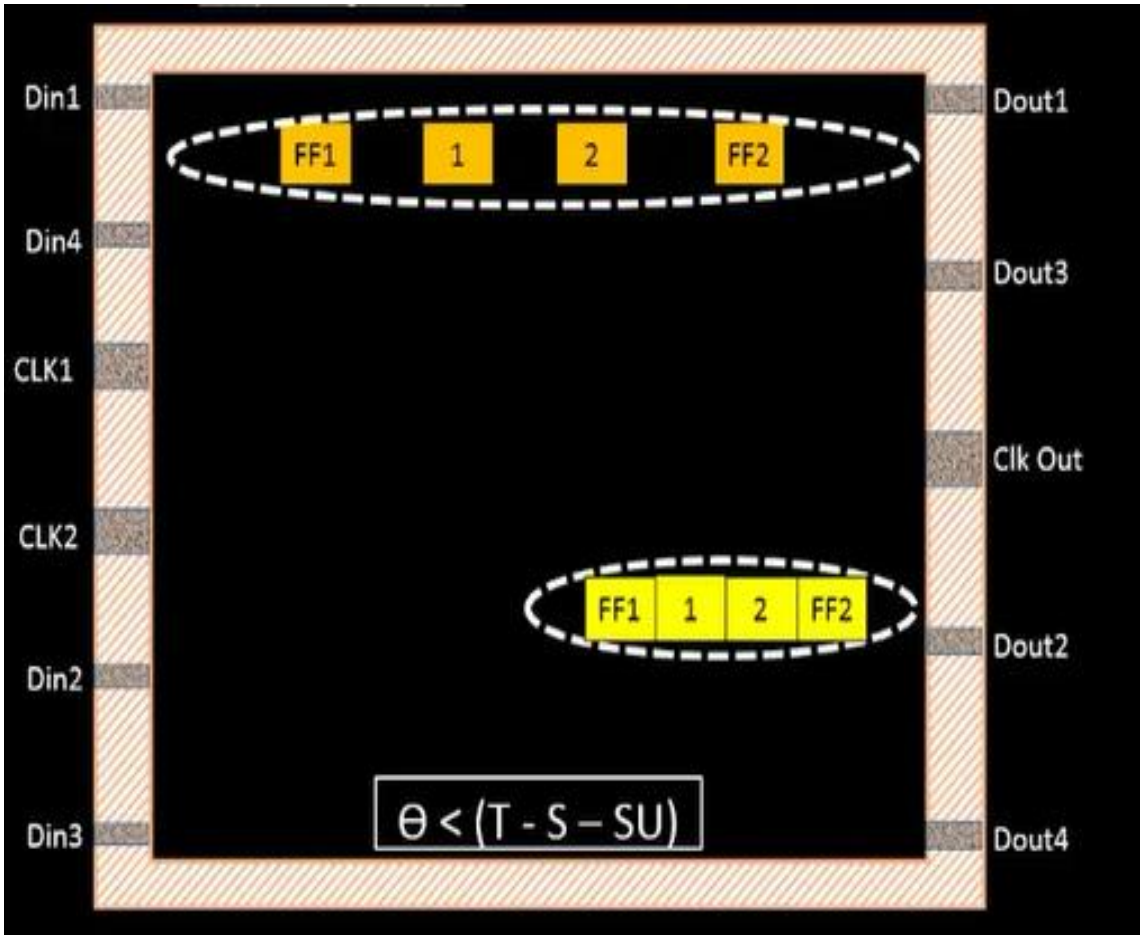



Fig. 2.12 Timing analysis with ideal clocks [8]

### 2.4.2 Data Slew Check

Data slew check basically specifies the data transition check. It is done to find out whether data transitions at the input and output are in the minimal range. There are advantages and disadvantages of having a very high transition and very low transition. That's why data transition is expected to be within a certain limit. If data transition goes below a certain limit then there will be huge current demand within a very short time and there will be huge short circuit current in vice versa case [9].

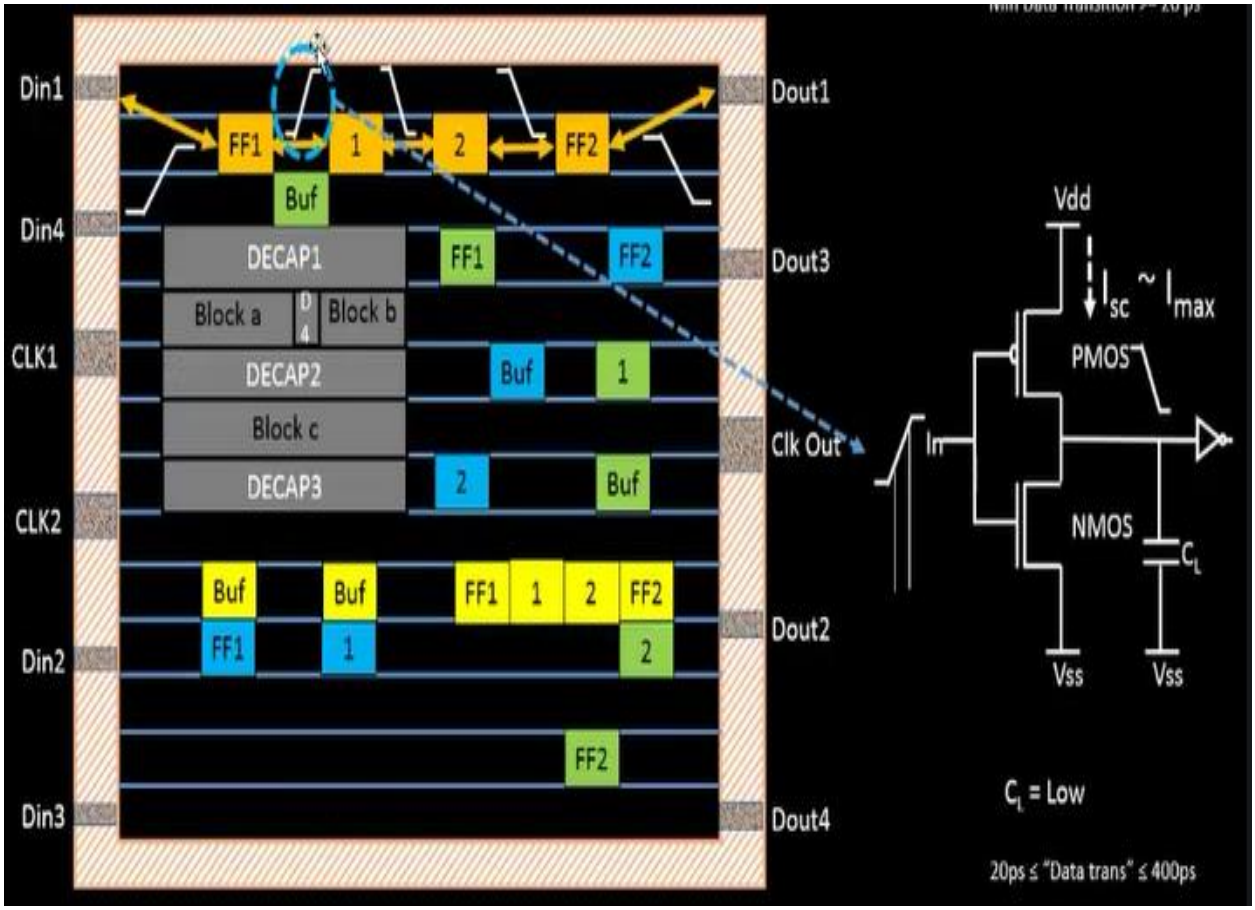



Fig 2.13 Data slew check [9]

### 2.4.3 Clock Tree Synthesis

Clock tree synthesis implies that clock has to reach every flop in the circuit at the respective time [10]. Clock is expected to reach every flip flop in the circuit at the same time. If it is not so then there will be skew which may lead to timing violation. We also add buffers in the clock path when there is huge wire length as the number of capacitors that clock signal has to charge becomes huge and there may be transitions because of that. So to maintain originality of the signal we add repeaters in the path. Signal keeps on getting reconstructed with the help of buffers.

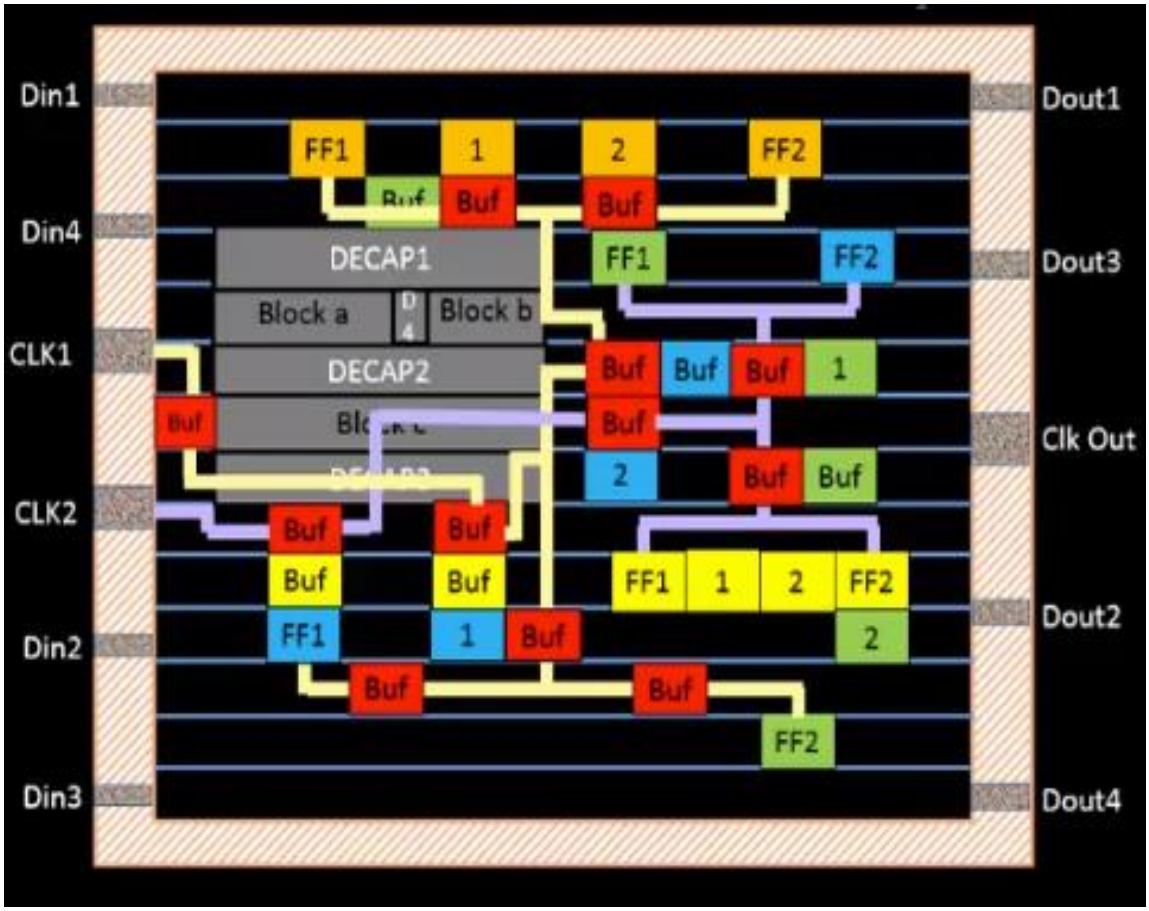



Fig. 2.14 Clock tree synthesis [10]

## 2.5 ROUTE , DRC CLEAN , PARASITICS EXTRACTION , FINAL STA

### 2.5.1 Timing Analysis (with Real Clocks)

Once we are done with the clock net shielding the next step is to do the timing analysis with the real clocks. Since now we have clock tree in place timing will change. With this new logic introduced we need to get idea of the current timing scenario. We have already found combinational delay of the data path. Next step is to find out the launch clock network delay , capture clock network and adjust the equation shown in the figure. We don't touch the clock tree network in the later stages so it makes sense to do a quick timing analysis after clock tree synthesis only [11].

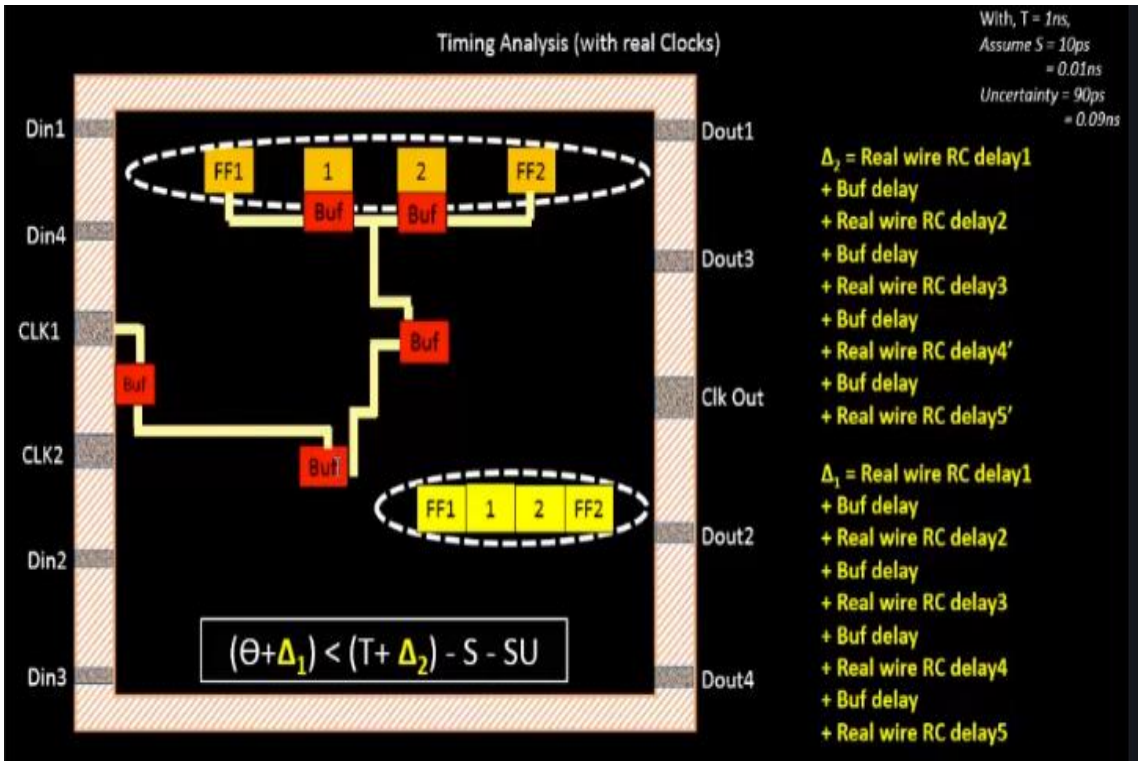



Fig. 2.15 Timing analysis with real clocks [11]

### 2.5.2 Route

In this case we will be getting access to the real wires. There is no more estimated wire. These routes are based on the connectivity from the netlist. We need to route the design and we need to maintain the existing timing scenario [12].

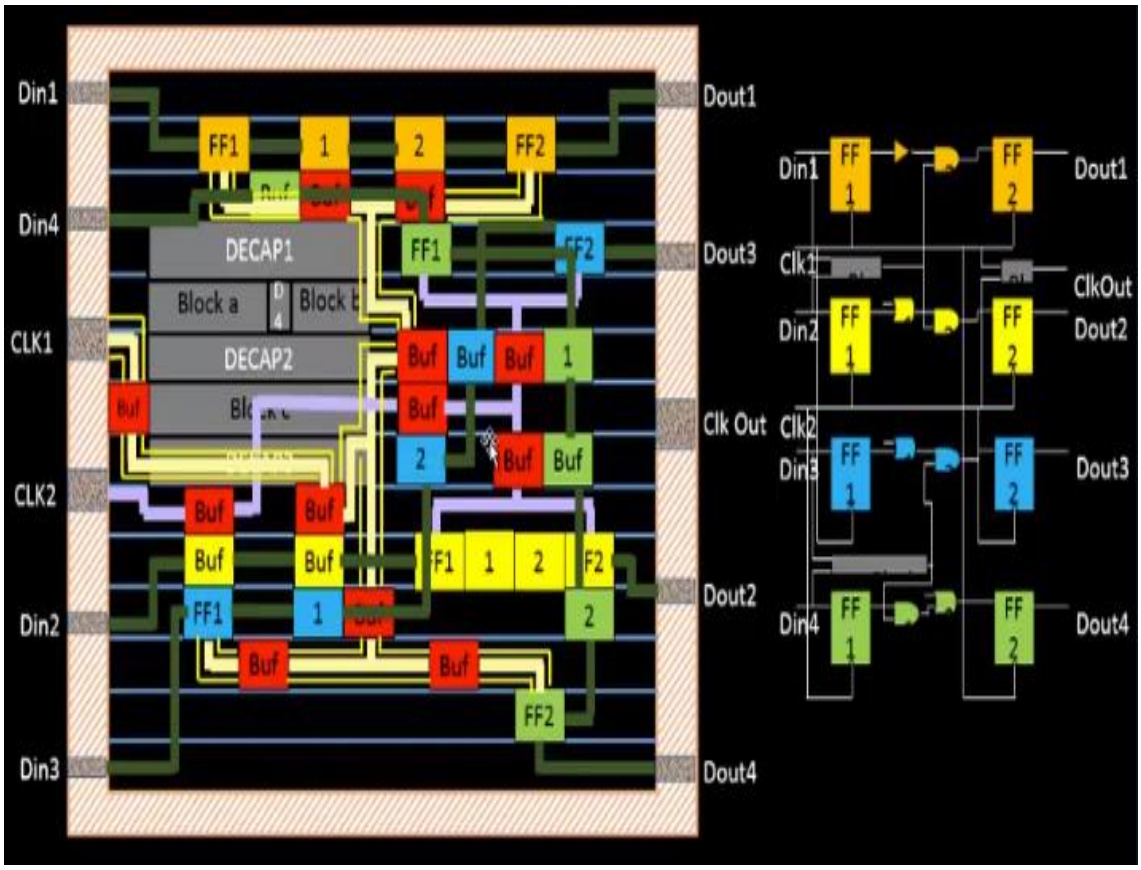



Fig. 2.16 Route [12]

### 2.5.3 DRC Clean

DRC stands for Design Rule Check. Next step after the route is to clean up the DRC's. During route stage there may be some spacing and shorts issues [13]. For example in the circled region in the figure the two routes are very close to each other and we might face a lot of coupling between them and it might not be able to get fabricated by a foundry. Finally this design goes to the foundry to get fabricated and foundry has its own set of rules. For example the minimum spacing between two wires should not be less than some x ums. Also there is a short violation in the circled region as we can see which also needs to be fixed. Also due to DRC clean route wires position might change and thus timing might also change and we need to fix them also if it happens so.

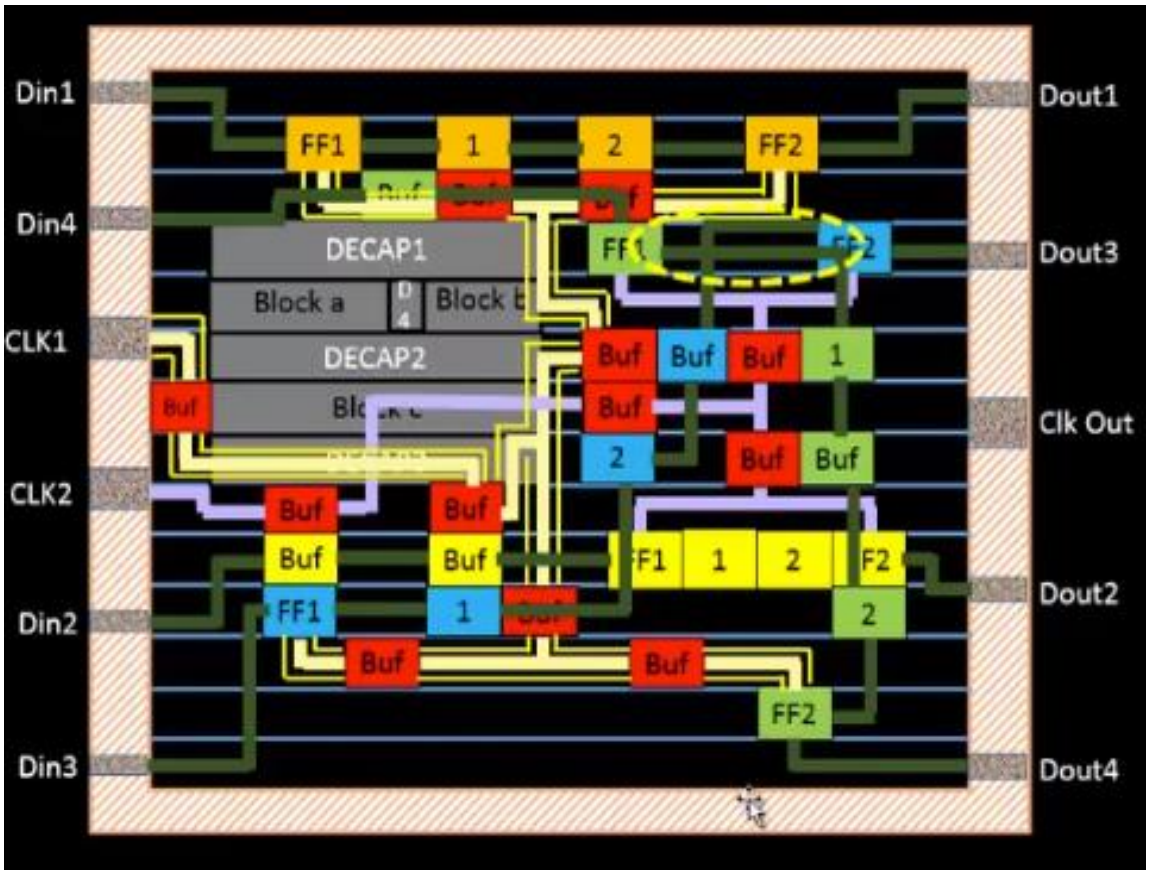



Fig. 2.17 DRC clean [13]

### 2.5.4 Parasitics Extraction

After DRC clean stage we do the full parasitics extraction since we have the data path, clock path and clock net shielding. Resistance and Capacitance of the entire chip is extracted as shown in the figure through some mechanism. Now we do the final round of timing analysis with the extracted spef. There is no more wire estimation. Now we have real wires, real parasitics [14].

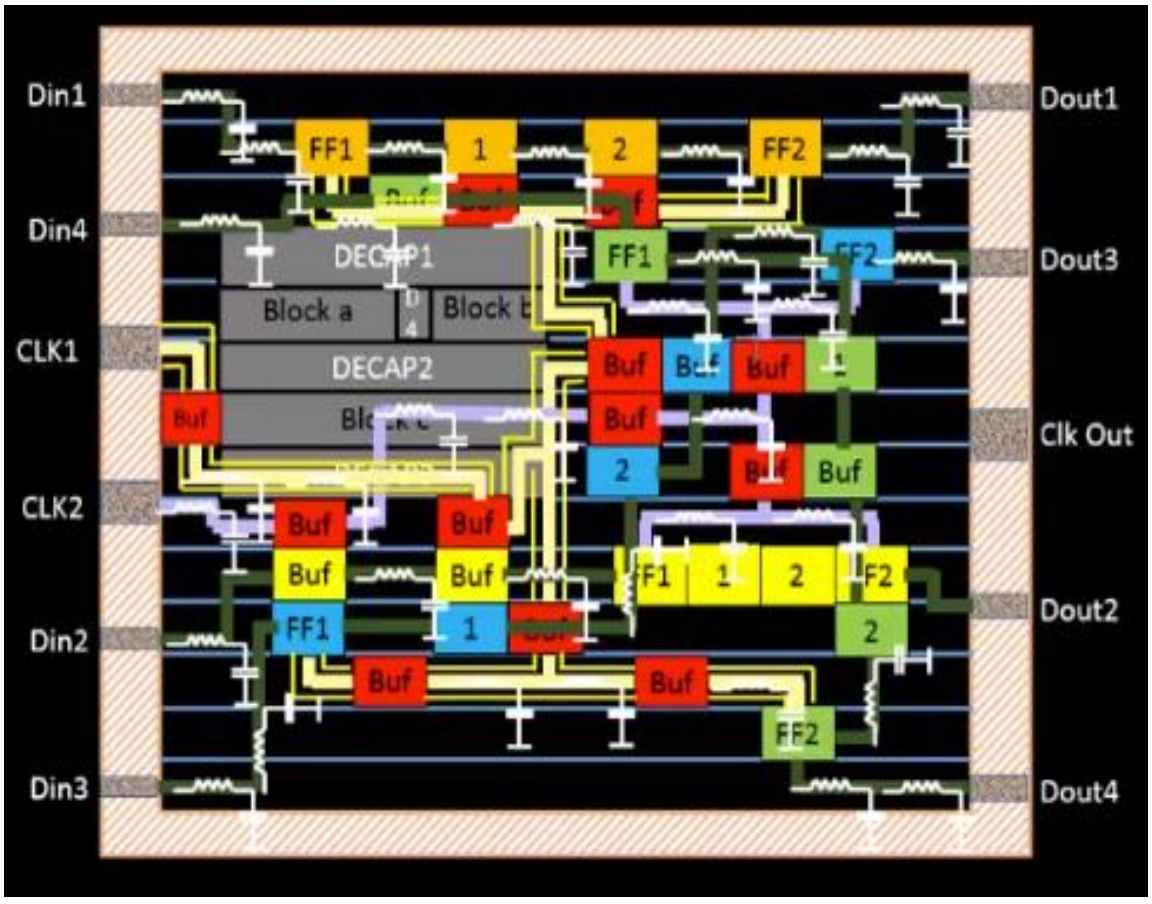



Fig. 2.18 Parasitics extraction [14]

### 2.5.5 Timing Analysis (With Real Clock and Wires)

Here the launch and capture clock network delay remains the same. Combinational delay which is from flip flop 1 to flip flop 2 has now real wire delay rather than the estimated wire delay what we have been doing till now. Final timing equation looks as shown in the figure [15].

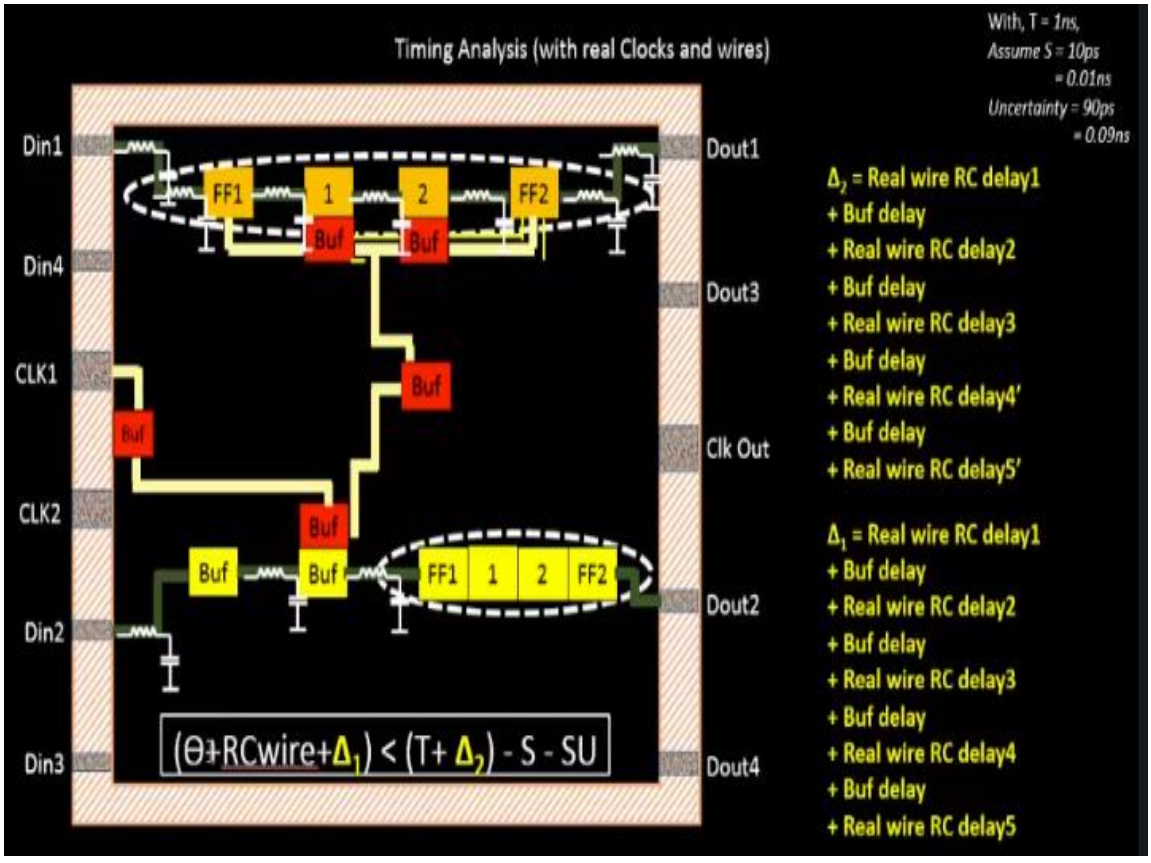



Fig. 2.19 Timing analysis post routing [15]

# CHAPTER 3 : FLOORPLANNING

## 3.1 UTILIZATION FACTOR AND ASPECT RATIO

First step in physical design flow is to define width and height of core and die. Here we will try to understand how do we come up with the values of W and H. Let's take an example of netlist as shown in the figure. While defining the dimensions of the chip we are mostly dependent on the dimensions of the logic gates. Now, let's convert the highlighted symbols into physical dimensions. To find out the dimensions of core and die we find out the dimensions of standard cells and not wires for now.

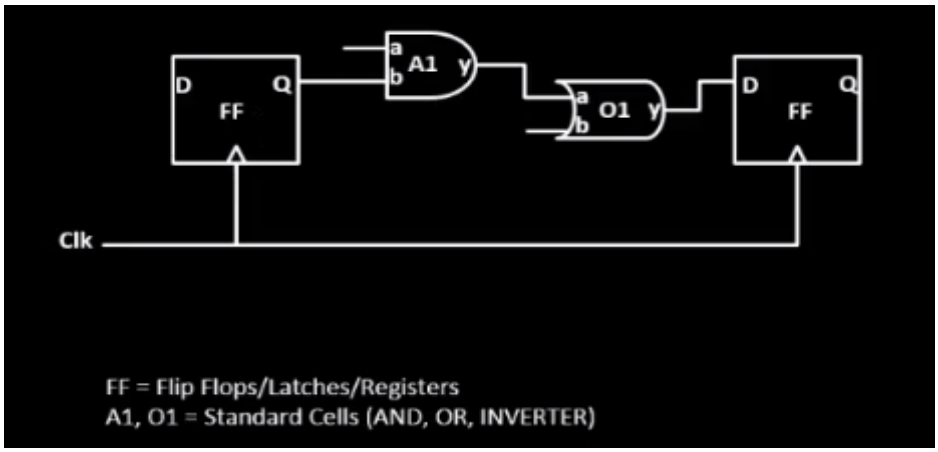
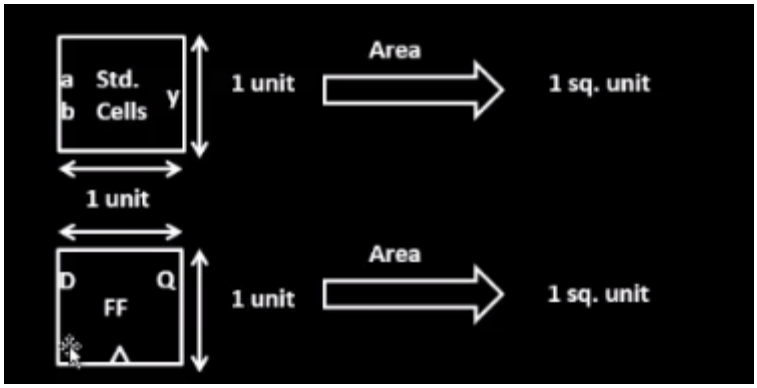



Fig. 3.1 Defining dimensions of logic gates

With the help of this dimensions of standard cell as shown in the figure we try to identify the area which is taken on the silicon wafer. After clubbing all the standard cells without wires we get the area of combined netlist as 4 square units as shown.



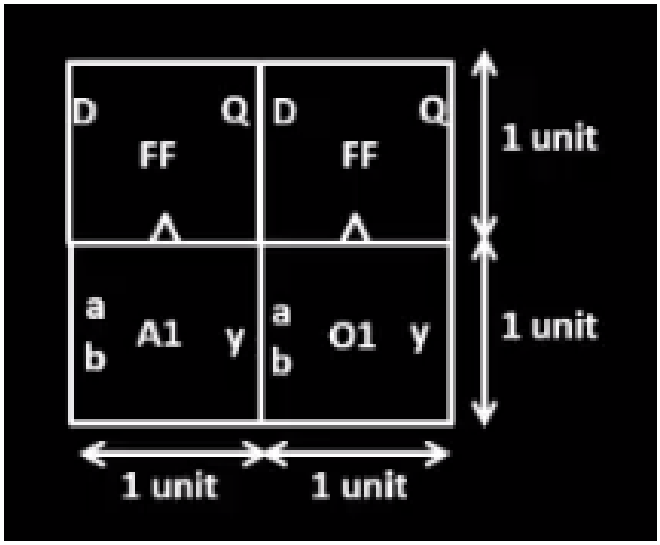
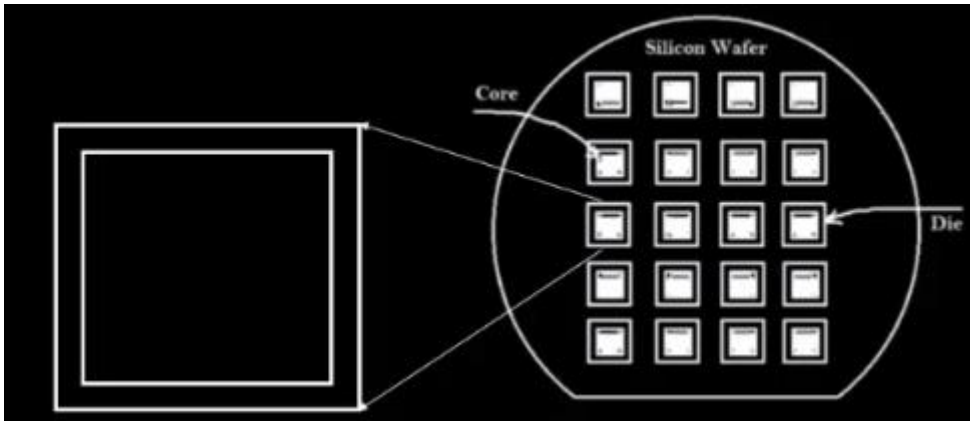




Fig. 3.2 Placing standard cells on the chip

A core is the section of the chip where the fundamental logic of the design is placed. A die, which consists of core, is a small semiconductor material specimen on which the fundamental circuit is fabricated. If the logic cells occupies the complete area of the core then it means we have achieved 100% utilization. Utilization factor is 1 in this case as shown. In ideal case we don't go 100% utilization, we go for 50-60% utilization.

Aspect ratio = (height of core) / (width of core)



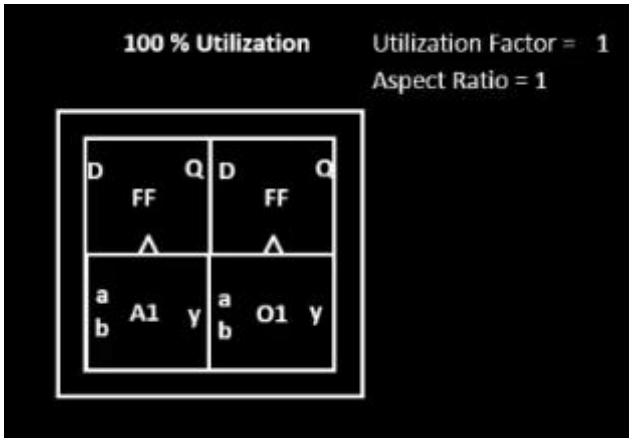




Fig. 3.3 Placing cells to get unity utilization factor

Now let's take the practical case with core height as 2 units and its width as 4 units. Now we place our logic inside it and finds out utilization factor is 0.5 and aspect ratio is also 0.5. Anything other than 1 represents rectangle shape of our core. We can place some additional cells in this area for some optimisation in the later stages for example buffers.

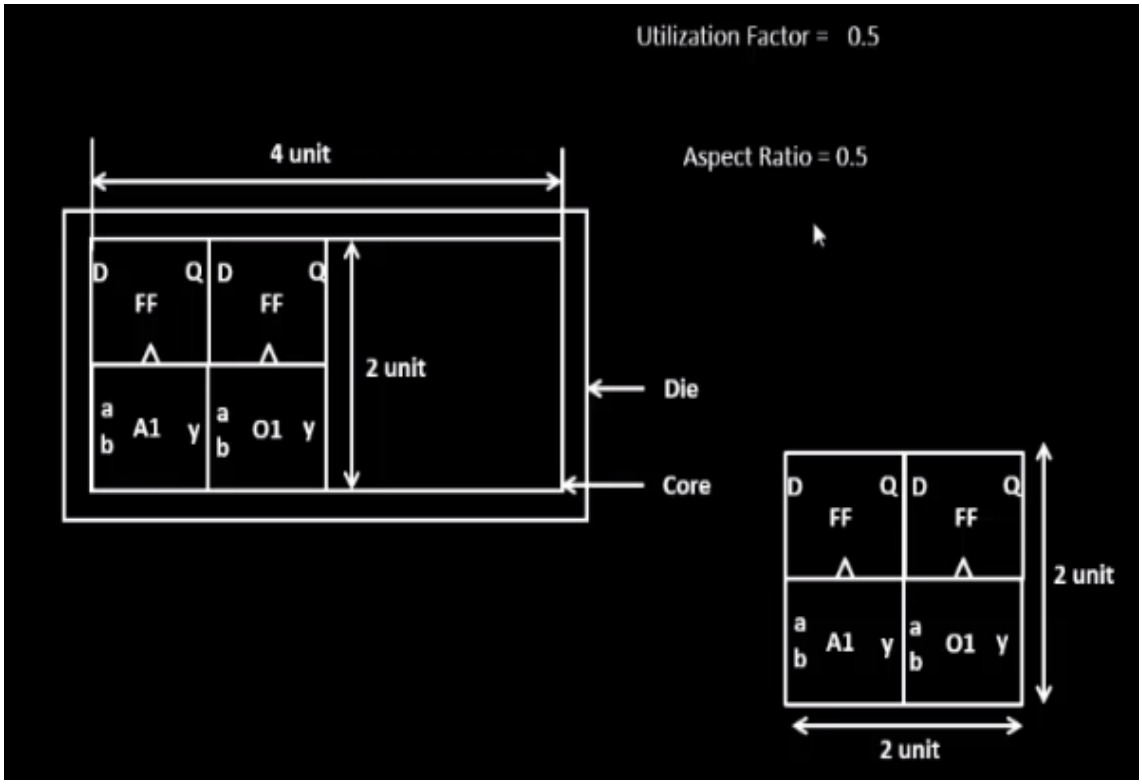



Fig. 3.4 Utilization factor other than unity

## 3.2 CONCEPT OF PREPLACED CELLS

The next step in floor planning is to define the location of preplaced cells. Let's take a combinational logic which performs some function for example a multiplier, a complex mux, clock divider or so. If this combinational logic is a huge circuit for example of some 100k gates. So we don't want to implement this circuitry every time as a part of the main circuit. We granularize this circuit into 2 parts viz. 50k gates each. As shown in the figure we cut the circuit into 2 parts and separate them. Now we treat them as 2 different blocks. These two blocks will be implemented separately. First step is to extend the input – output pins and then black box these boxes.

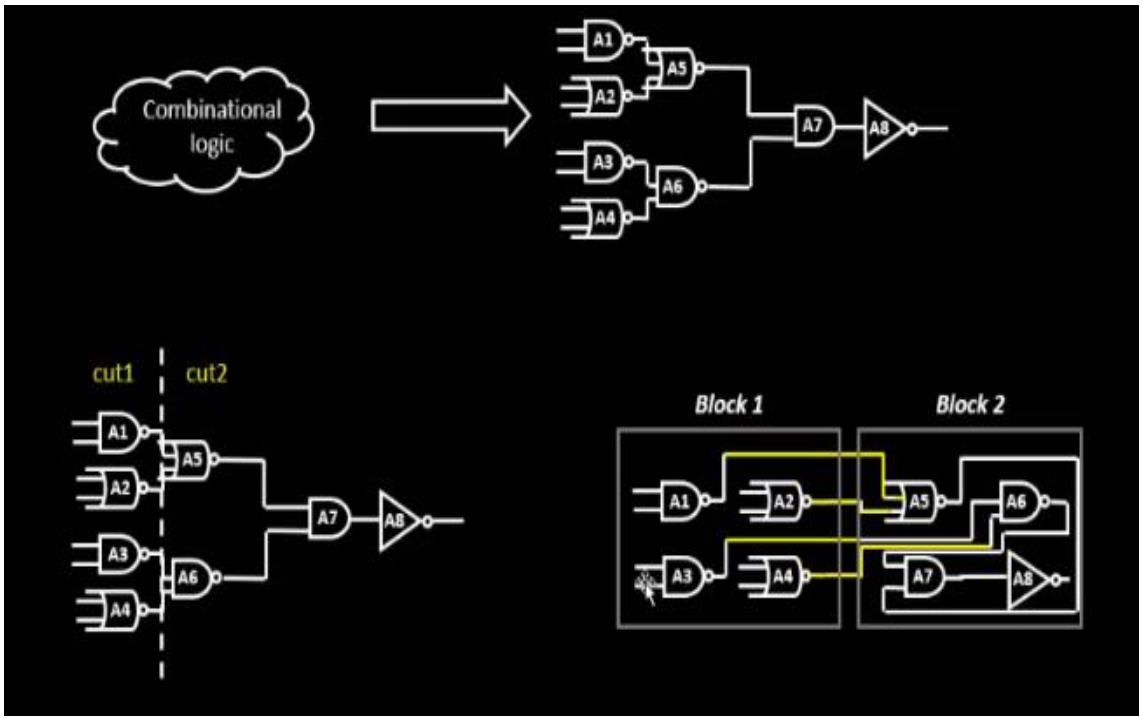



Fig. 3.5 Example of combination logic to be implemented

As shown in the figure first block acts as 4 inputs 4 outputs block and second block acts as 4 inputs 1 output box. Advantage of separating blocks in such fashion is suppose if this block is being replicated multiple times on the netlist to perform some function, we need

not implement it multiple times. We just black box them. So they are implemented only once and they can be reused multiple times as per requirement. Similiarly there are also other IP's available like memory, clock gating cell, comparator, mux etc. The arrangement of these IP's is called Floor planning.

These IP's/blocks have user defined locations, hence are placed in chip before automated placement -and- routing and are called as pre placed cells. Automated placement and routing tools place the remaining logical cells in the design onto chip.

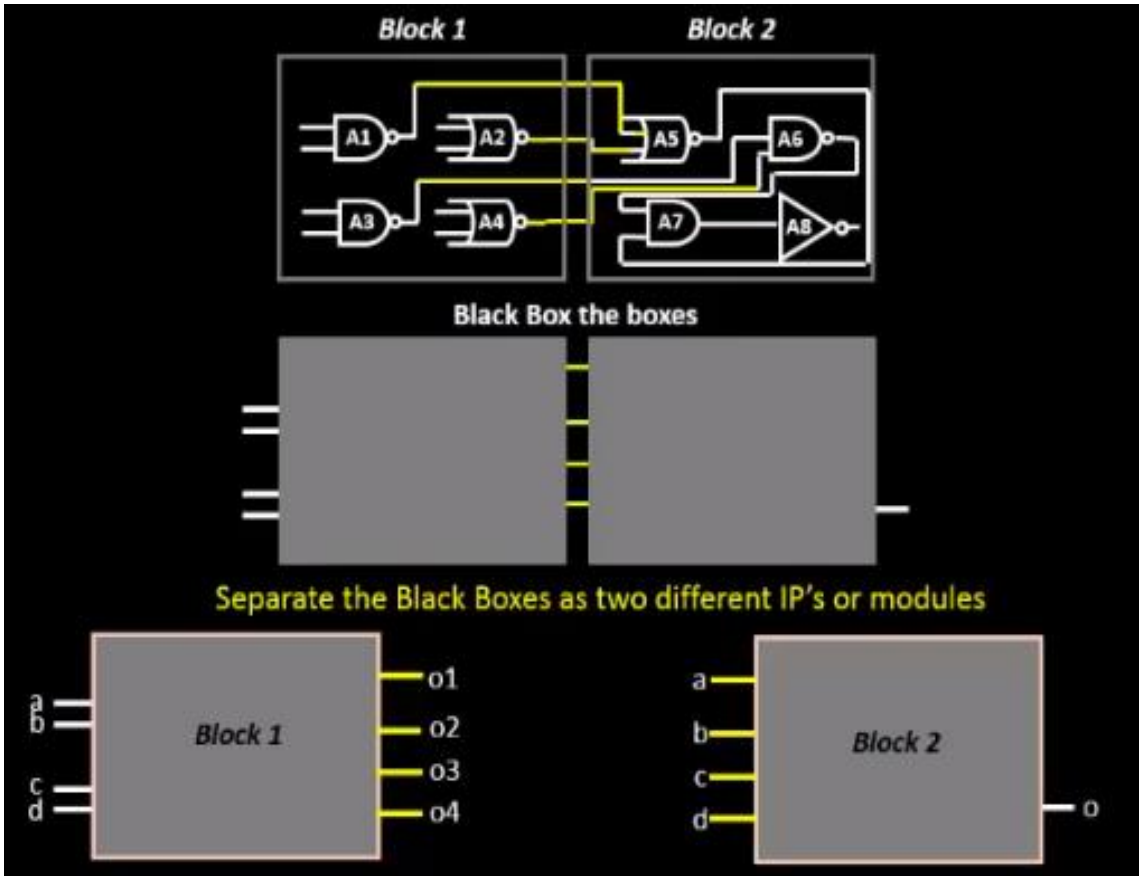



Fig. 3.6 Black boxing the preplaced cells

## 3.3 DE – COUPLING CAPACITORS

We need to surround pre placed cells with de -coupling capacitors. Consider the circuit shown in the figure shown as a part of any of pre placed cells. When the output of AND gate switches from logic 0 to logic 1 it requires huge amount of current. The output capacitance of AND gate has to be completely charged to represent logic 1. It is responsibility of the power supply to send the sufficient amount of current to charge the capacitor to logic 1. Also when it is switching from logic 1 to logic 0 it is the responsibility of Vss to take that amount of charge from the circuit. Now due to presence of Rdd and Ldd ,there will be a voltage drop across them and the voltage at node 'A' would be Vdd' instead of Vdd.

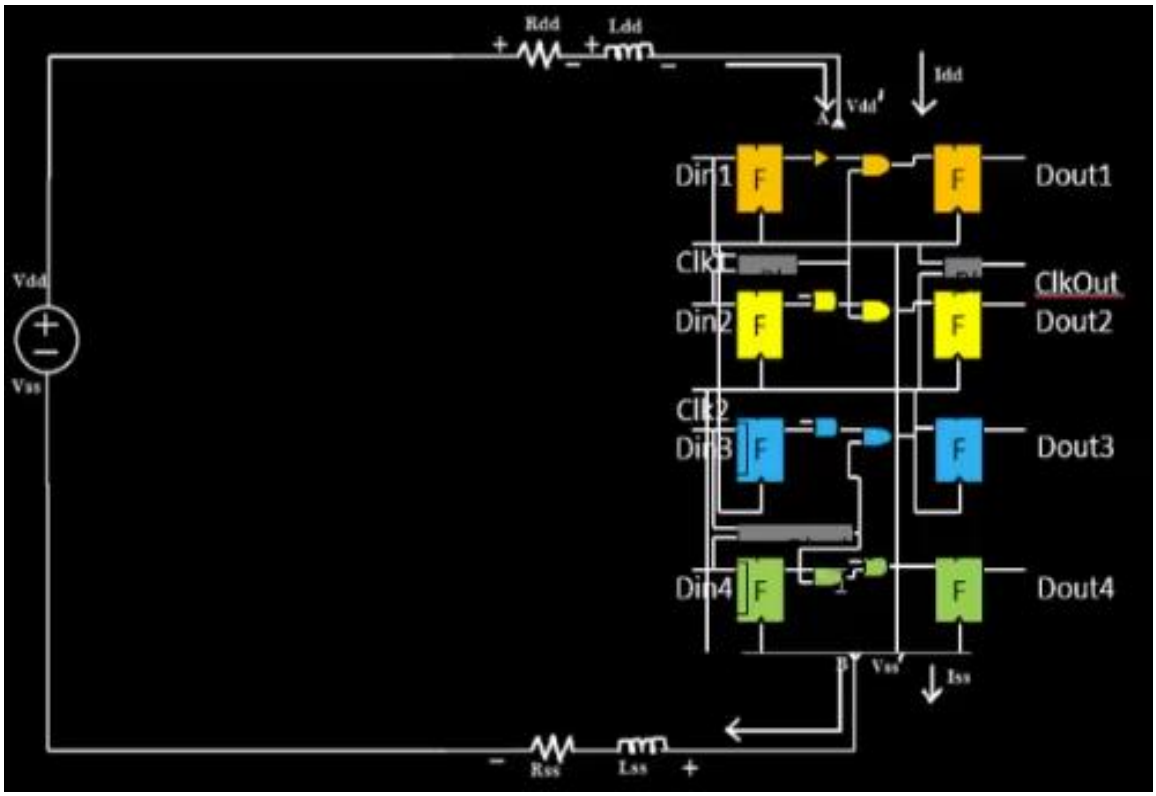



Fig. 3.7 Need for decoupling capacitors

Now suppose this Vdd =1V and Vdd' =0.7V so the output of capacitor can not go beyond 0.7V. Now this 0.7V to be considered as logic 1 it has to be present within noise margin range.

As we can see in noise margin diagram shown any voltage which lies between Vol and Vil is logic '0' , between Vil and Vih as undefined region and between Vih and Voh as logic '1'. So if voltage drop is too high then out Vdd' value may lie in undefined region and we can not be sure whether it can go to logic '0' or logic '1'.

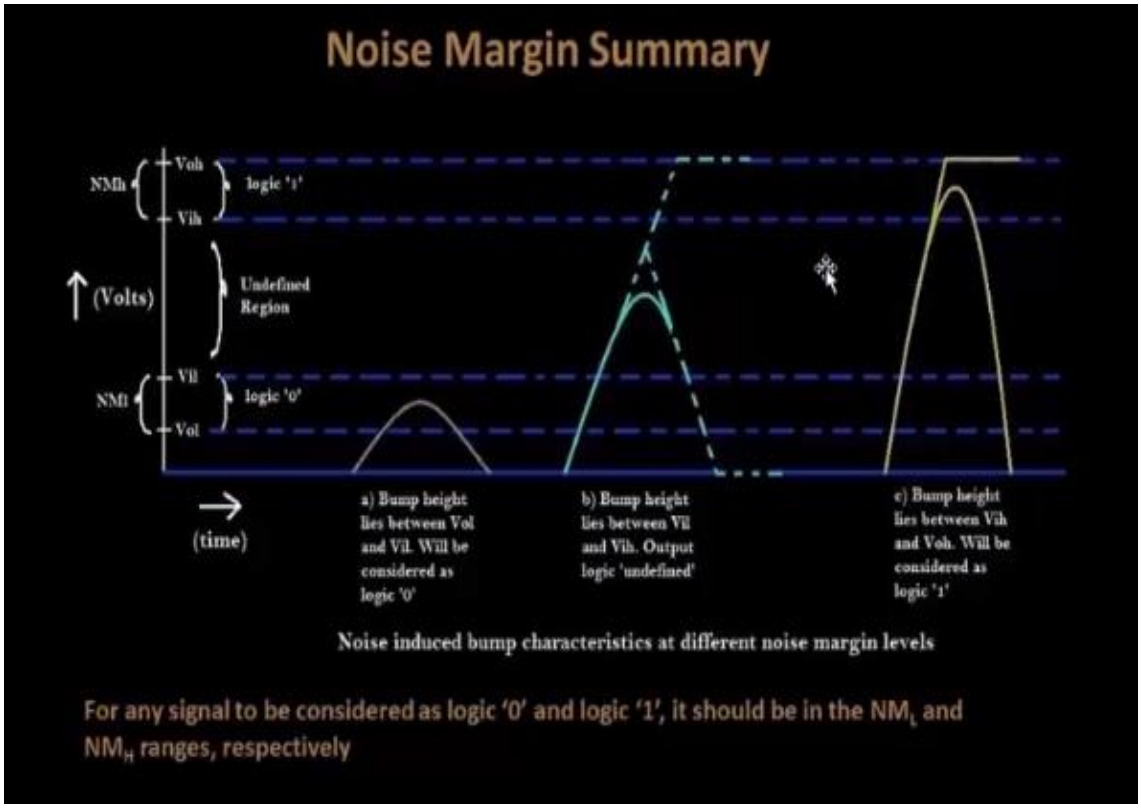
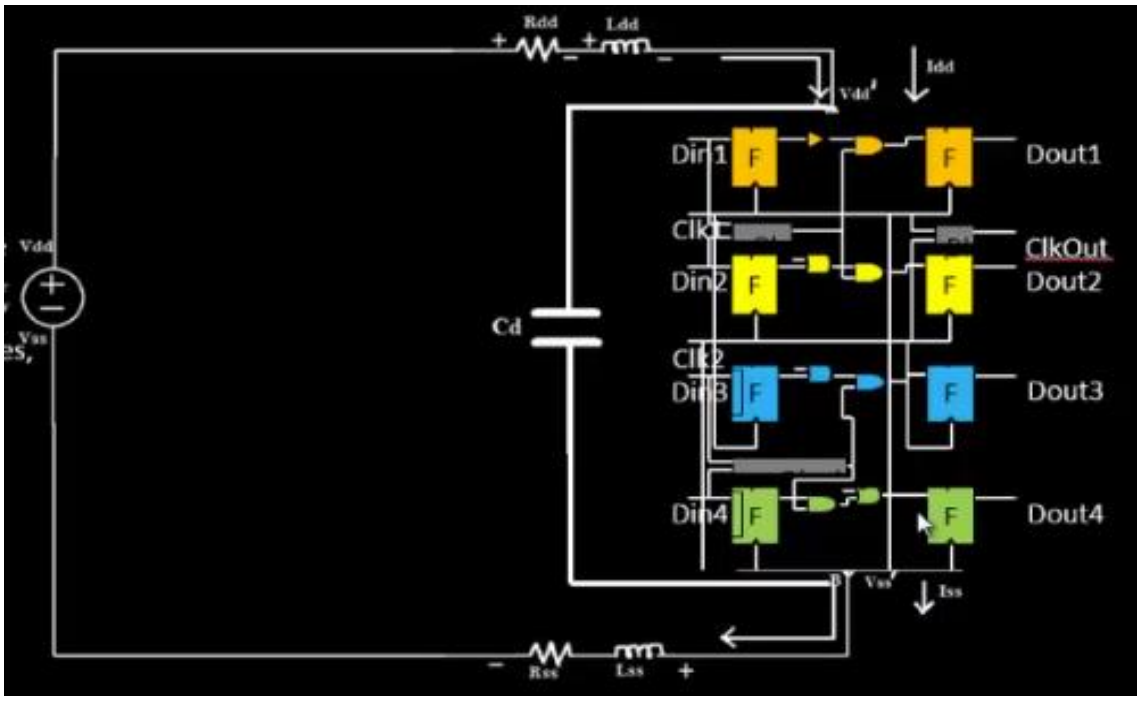



Fig. 3.8 Noise margin concept

This problem can be solved with the help of decoupling capacitors. Decoupling capacitor is a huge capacitor which is completely filled with charge. Equivalent voltage acroos the decoupling capacitor is same as we have of the power supply. Whenever this part of circuit switches it gets its current from the decoupling capacitor.

This capacitor decouples the main circuit from the main supply. Since decoupling capacitor is placed close to this circuit there is hardly andy voltage drop due to physical wire length. When there is no switching activity going on , this decoupling capacitor gets replenished its charge from the main power supply.



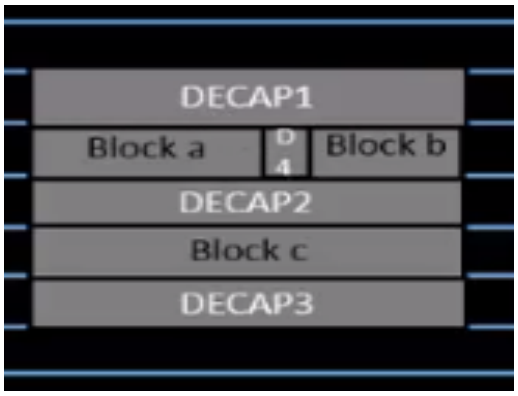




Fig. 3.9 Placement of decap around macros

## 3.4 POWER PLANNING

We have taken care of the local communication. Now, consider below scenario. Now treat each circuitry as black box and consider it a macro. Suppose we have 4 macros in our complete chip as shown Assume that a signal is being sent from driver to load and driver is switching from logic 0 to logic 1. So the interconnect should retain the same signal so that load receives the original signal. As we can see in case of single main power supply it is at a distance from the driver so there is all the chances of voltage drop.

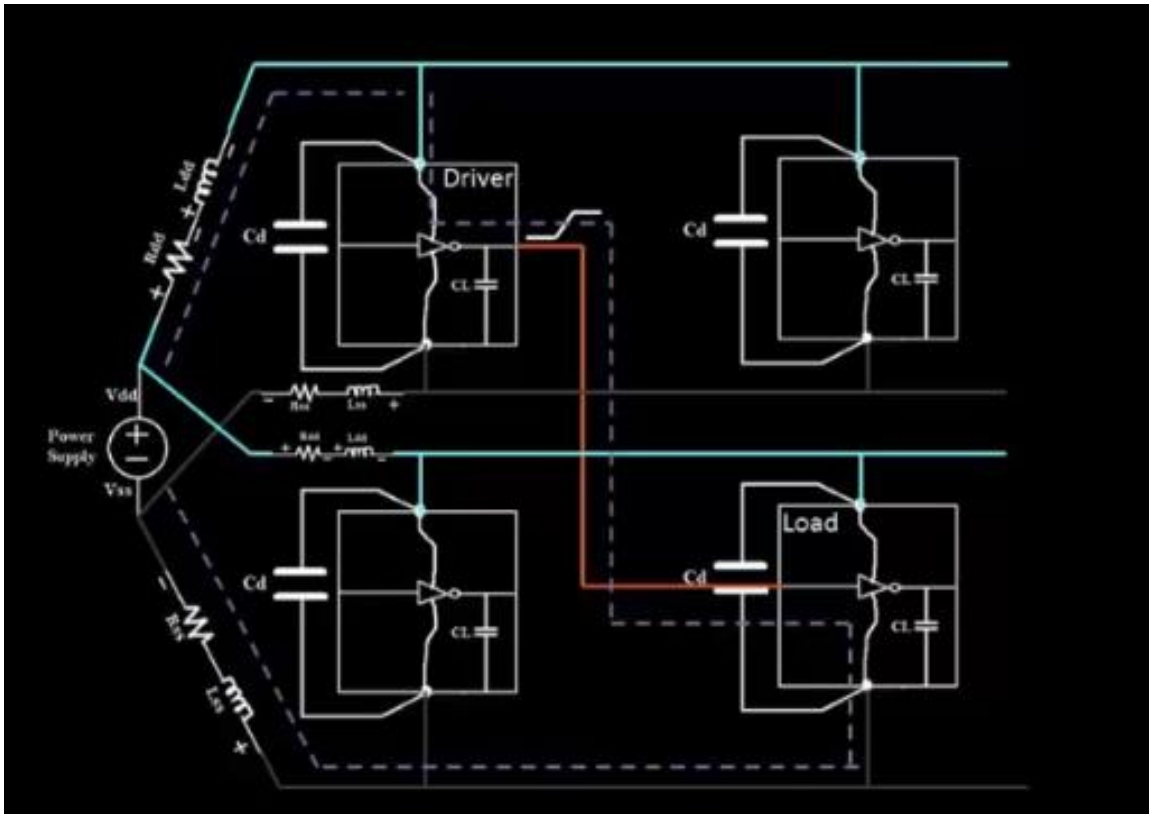



Fig. 3.10 Disadvantages of single power supply

Now instead of having a single source of power supply if we have multiple power supplies we can solve this problem as shown in the figure. So if component of the circuit demands an amount of current , it takes it from the nearest power supply. That's why in modern chips we have multiple pins for Vdd and Vss ports.

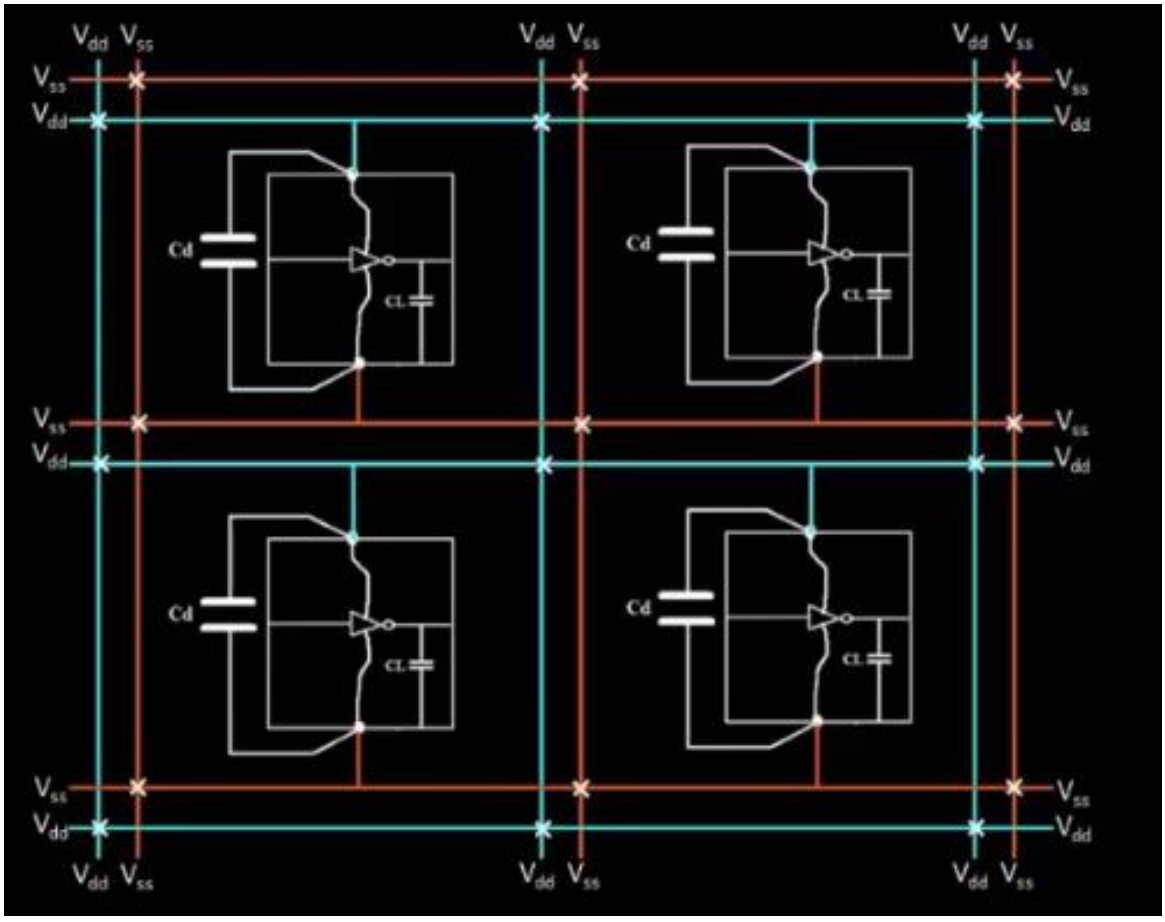



Fig. 3.11 Multiple power supplies

## 3.5 PIN PLACEMENT AND LOGICAL CELL PLACEMENT BLOCKAGE

Before going for pin placement let's take below design for example that needs to be implemented. It is one of sections of that circuit. Similiarly we have 2nd circuit which is driven by clock2 instead of clock1 like circuit 1. We also have some pre placed cells and their connections with the circuit are as shown. Also see the second part of the circuit in which FF1 and FF2 are being driven by different clocks in the same circuit.

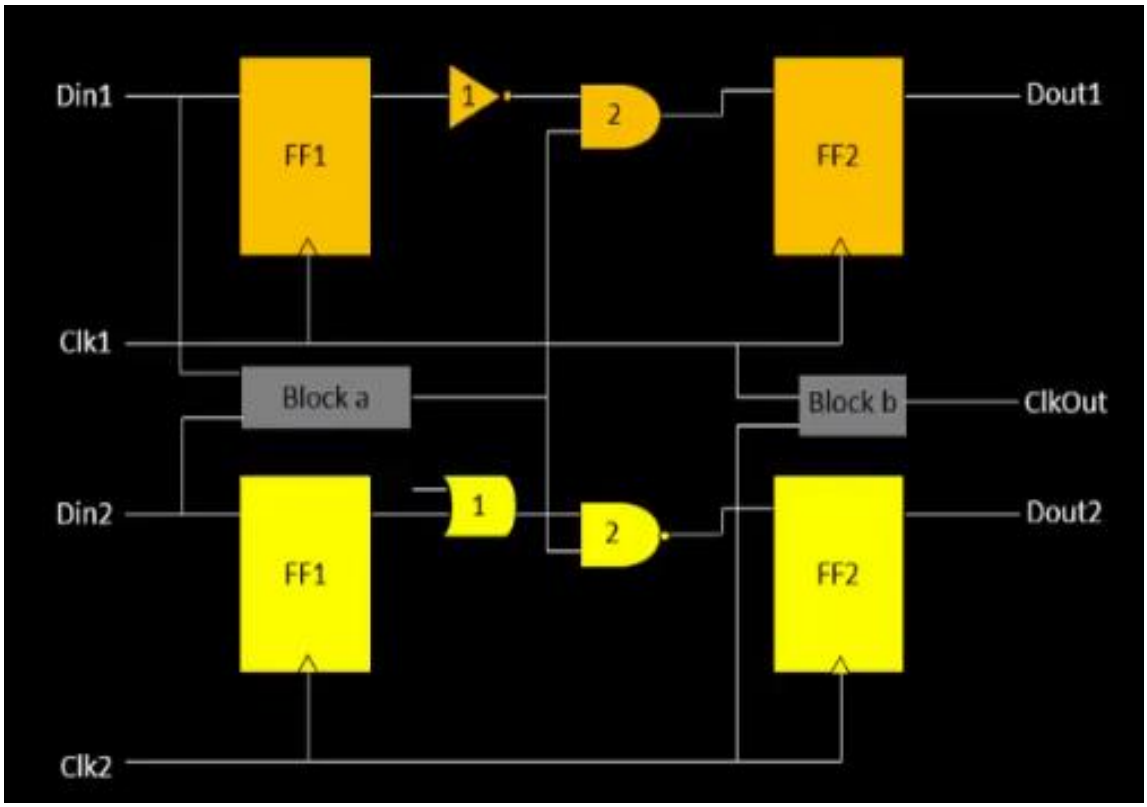



Fig. 3.12 circuit 1

Now look at the complete design as shown below. Now connections are being setup between these two sub circuits according to the clock connections. The connectivity information between the gates is coded using VHDL/Verilog language and is called the "netlist". Here we have 4 input ports Din1, Din2,Din3,Din4 and clk1 and 4 output ports Dout1,Dout2,Dout3,Dout4 and clk out.

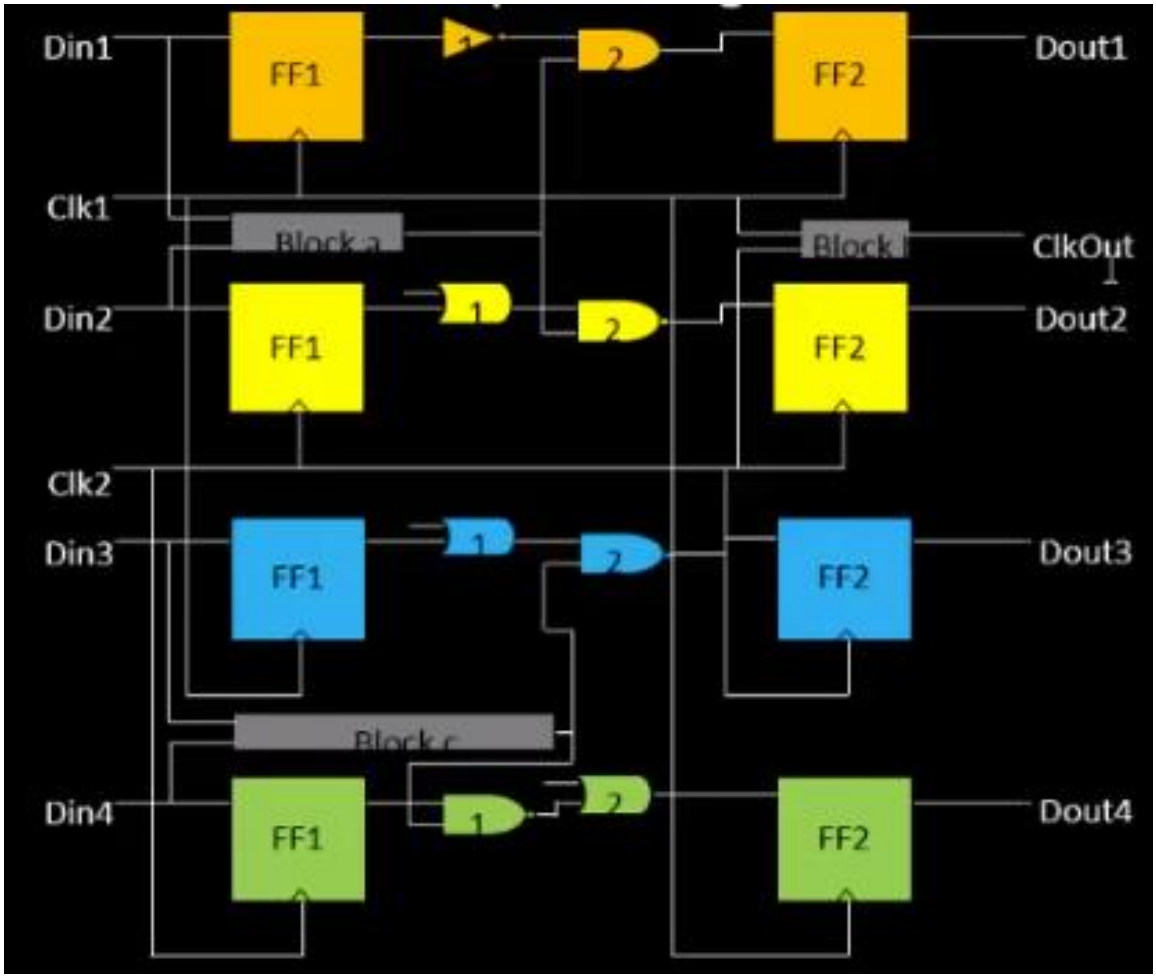



Fig. 3.13 Complete design

Now let's put this design into the chip which we are trying to design. Here we have area between the core and die which is required to be filled with the pin information. In general we put all the input pins on the left hand side, all the output ports on the right hand side but it might vary from design to design. Here ordering of input and output pins are random. Complete knowledge of the functionality of the design is very important for the ordering of the pin placement. This is basically handshaking between front end and back end. As we can observe clock ports are bigger in size than data ports because they drive all the flops, complete chip so we want least resistance in their path.

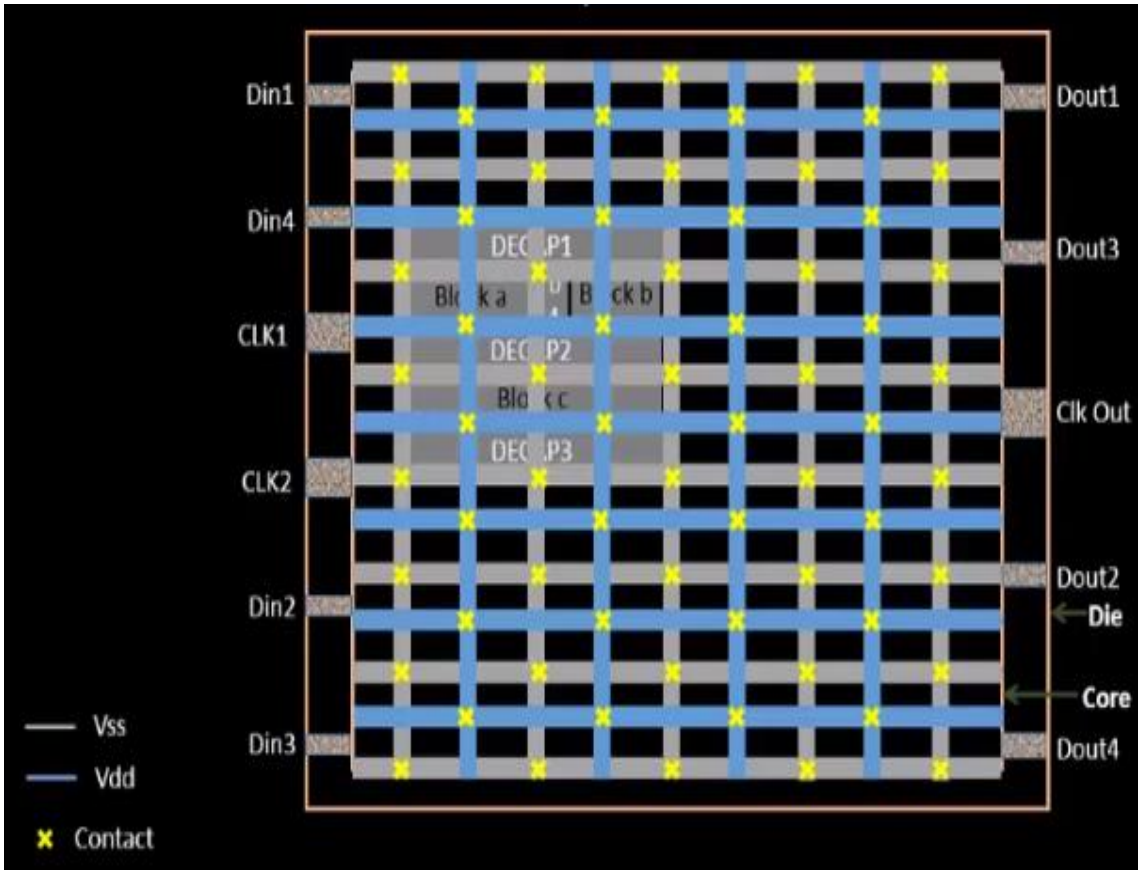



Fig. 3.14 Pin placement on the chip

Now comes the concept of Logical Cell Placement Blockage. As we can see in the figure we block the area with some blockage so that it does not place any standard cell in the area which is reserved for pin placement.

Now after this step our floorplan is ready for placement and route stage.

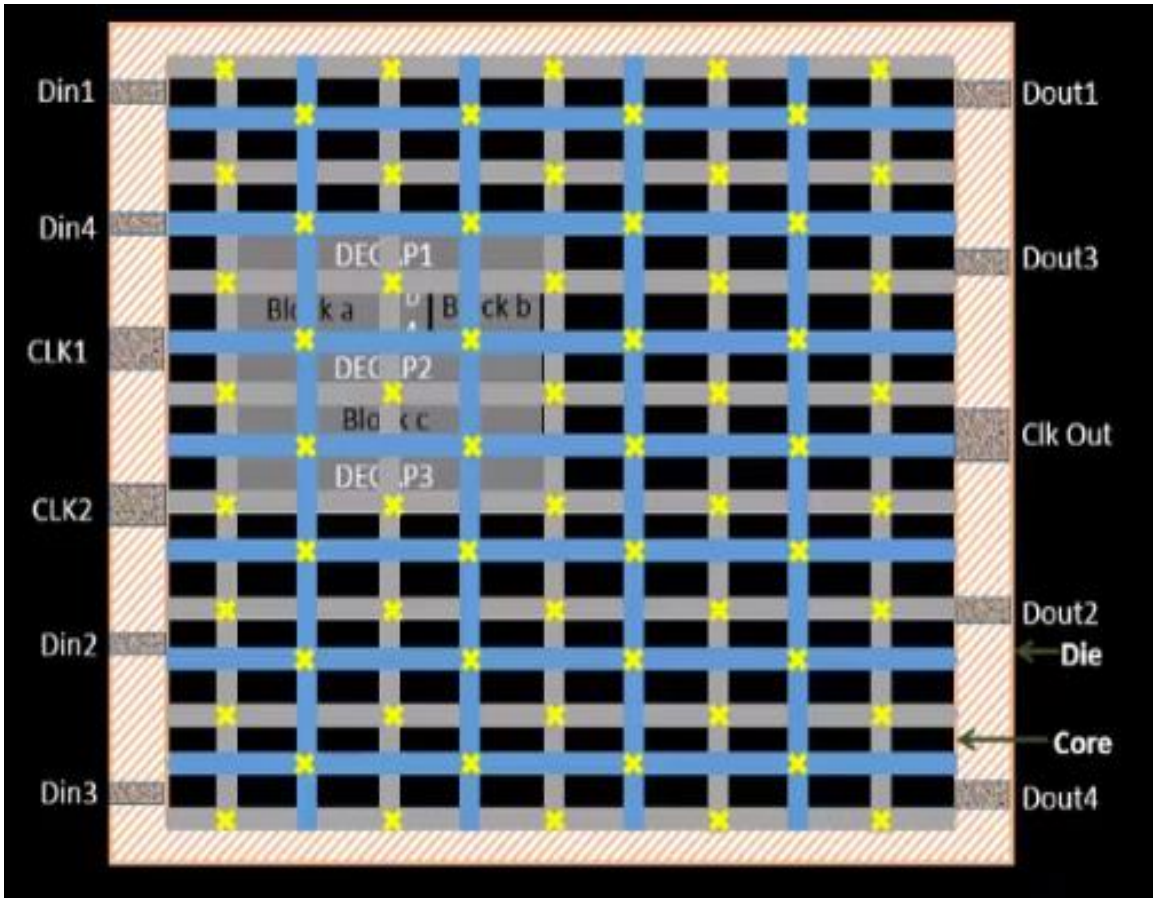



Fig. 3.15 Logic cell placement blockage

# CHAPTER 4 : PLACEMENT

## 4.1 NET-LIST BINDING AND PLACEMENT

First step in this stage is to bind the netlist with the physical cells. Consider the netlist given below with all the gates as shown. Shape of these gates represent the functionality of these gates. But in reality we don't have shapes like these, we have shapes like of a box. So we take each and every component of this netlist and give them proper shape and size by giving them proper height and width. So we have four sections of the netlist as shown.

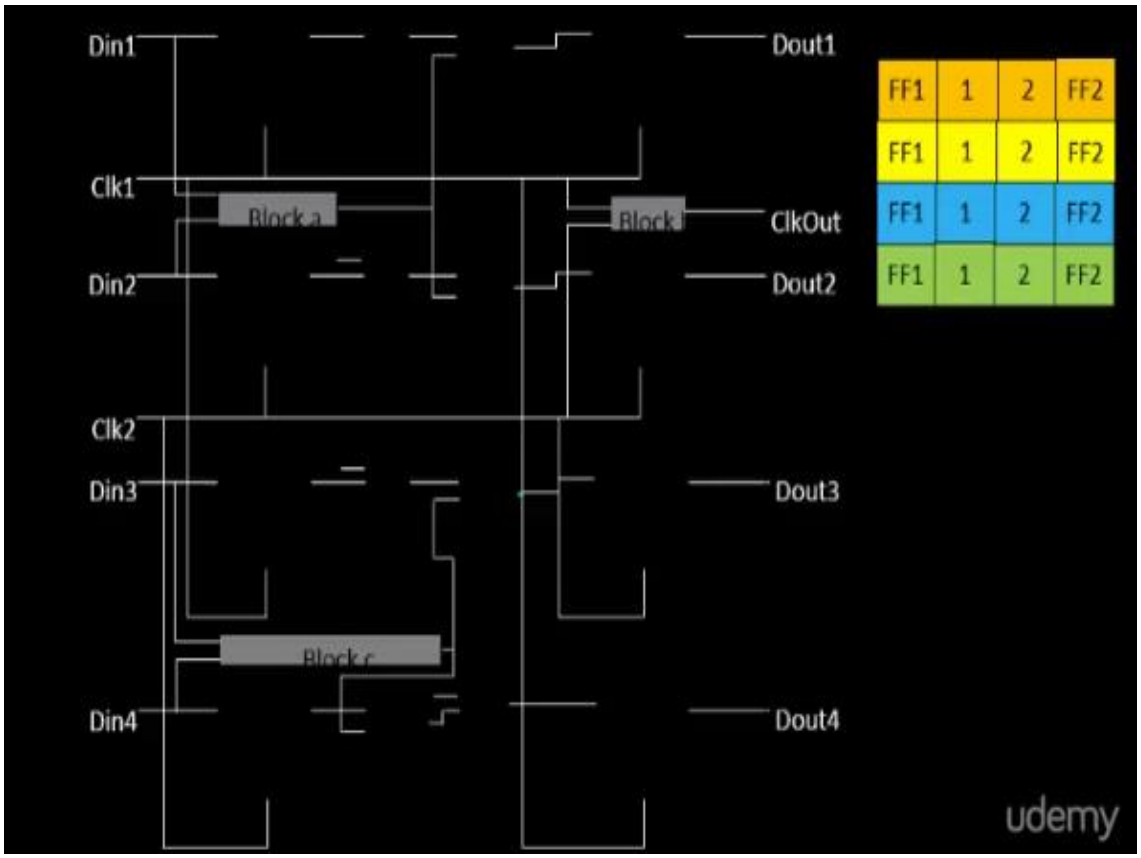



Fig. 4.1 Netlist

Library has all the information of the standard cells and pre placed cells including their timing information for example delay of each gate. Library can also be classified into two. One library contains only shape and size and the other only contains the timing information. It also contains the conditions of the cells. At what condition flip flop 1 emits the output. The when condition and all.

Now as we can see in second figure we have bigger gates in size as compared to the first one and they provide less resistance comparatively. So we have different flavours of the same cell type. We can pick up what we want based on the timing condition and the available space.

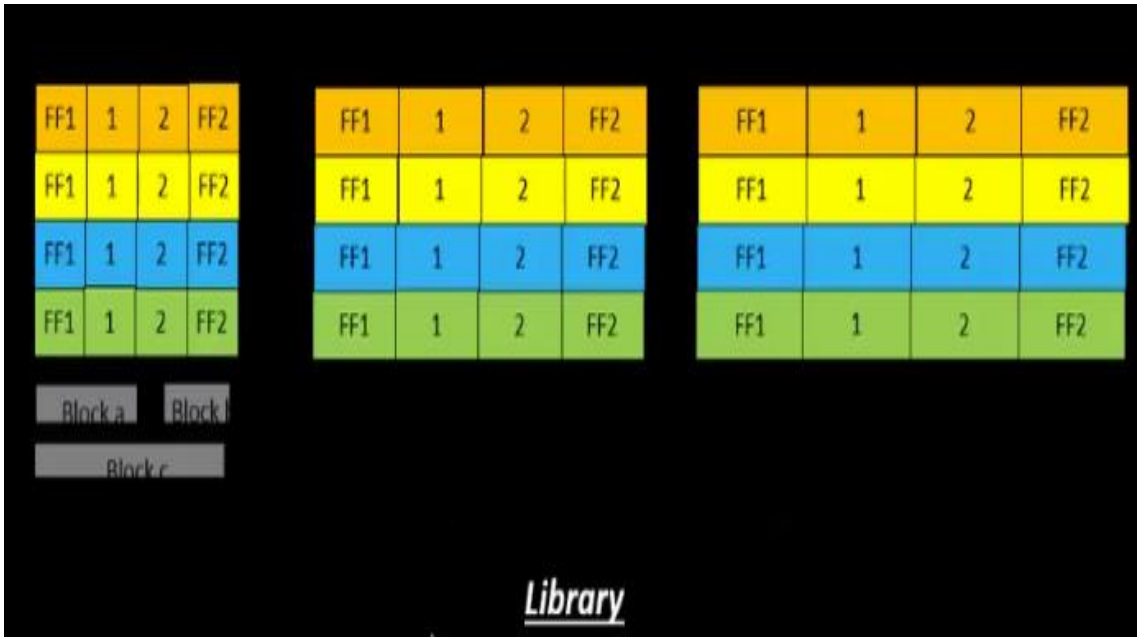



Fig. 4.2 Library of standard cells

Once we have given proper size and shape to all the cells next step is to place it on our readymade floorplan. So now we have our well defined floorplan, netlist and proper defined shape and size of gates which is physical view of the logic gates. Now the task is to place this particular netlist on to the floorplan.

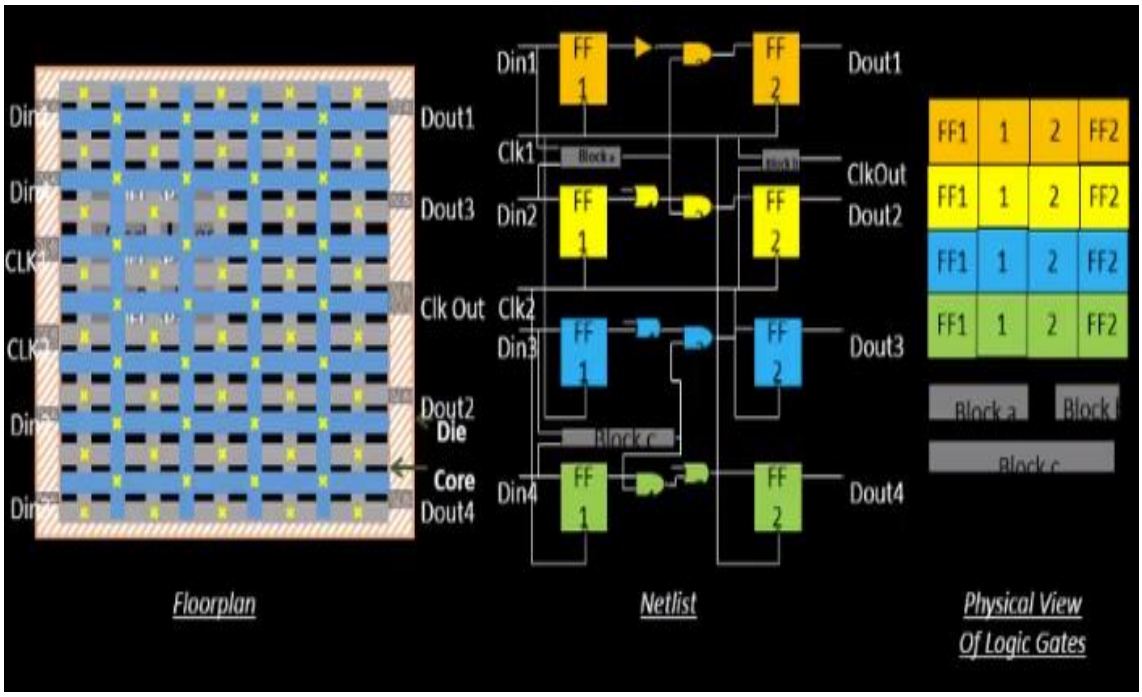



Fig. 4.3 Binding netlist with physical cells

Placement stage makes sure that there will be no placement on the placement blockage area and pre placed cells location would not be affected. We have to take the physical view of the netlist and place it on the floorplan. As we can see FF1 is placed close to Din1 so we place it accordingly on our floorplan. Same goes for FF2 and Dout1. So to reduce the huge timing delay we place all the cells in such fashion.

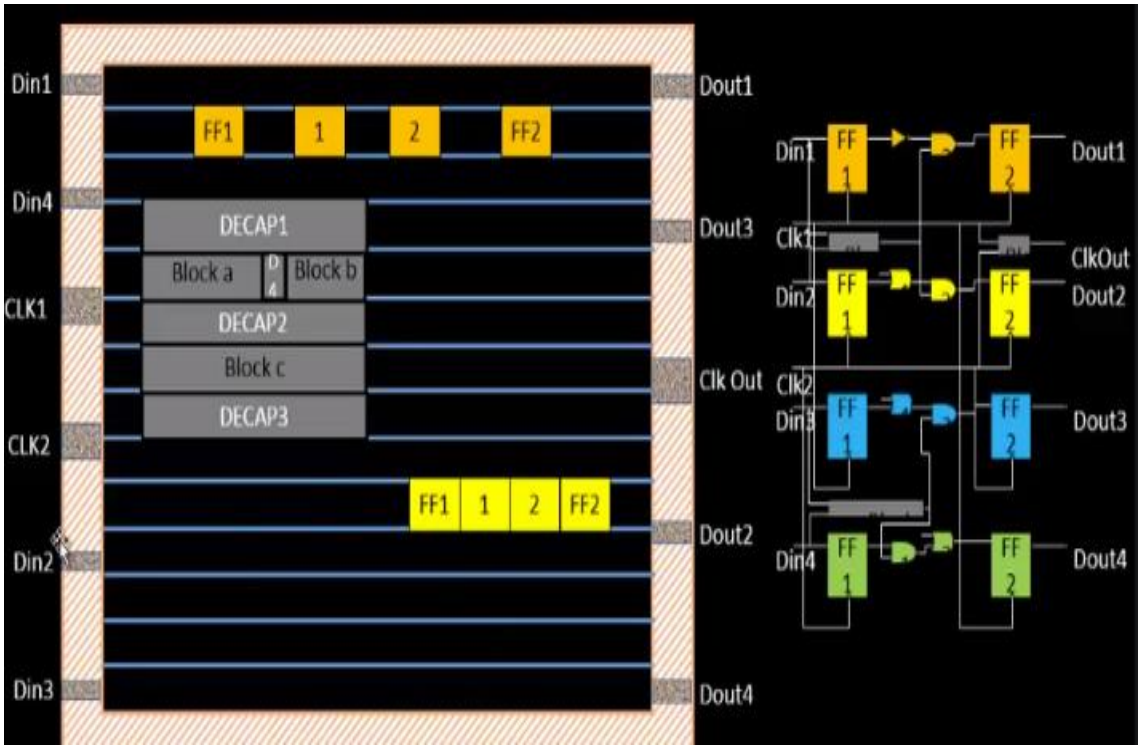



Fig. 4.4 Placement of cells on the chip

## 4.2 OPTIMIZE PLACEMENT

Repeaters are conditioners that repeats our signal, recondition it and gives the output back as the original signal. This way by adding some extra repeaters or extra buffers we maintain the signal integrity but there is loss of area. Based on the wire length between two cells and the capacitance calculations we decide whether we need to add repeaters or not in between them. In case of making a circuit work at very high speed we connect them like FF1 and FF2 as shown. Now as shown in the figure we added repeaters in between the cells where length of the wire is relatively large and signal might face delay and degradation.

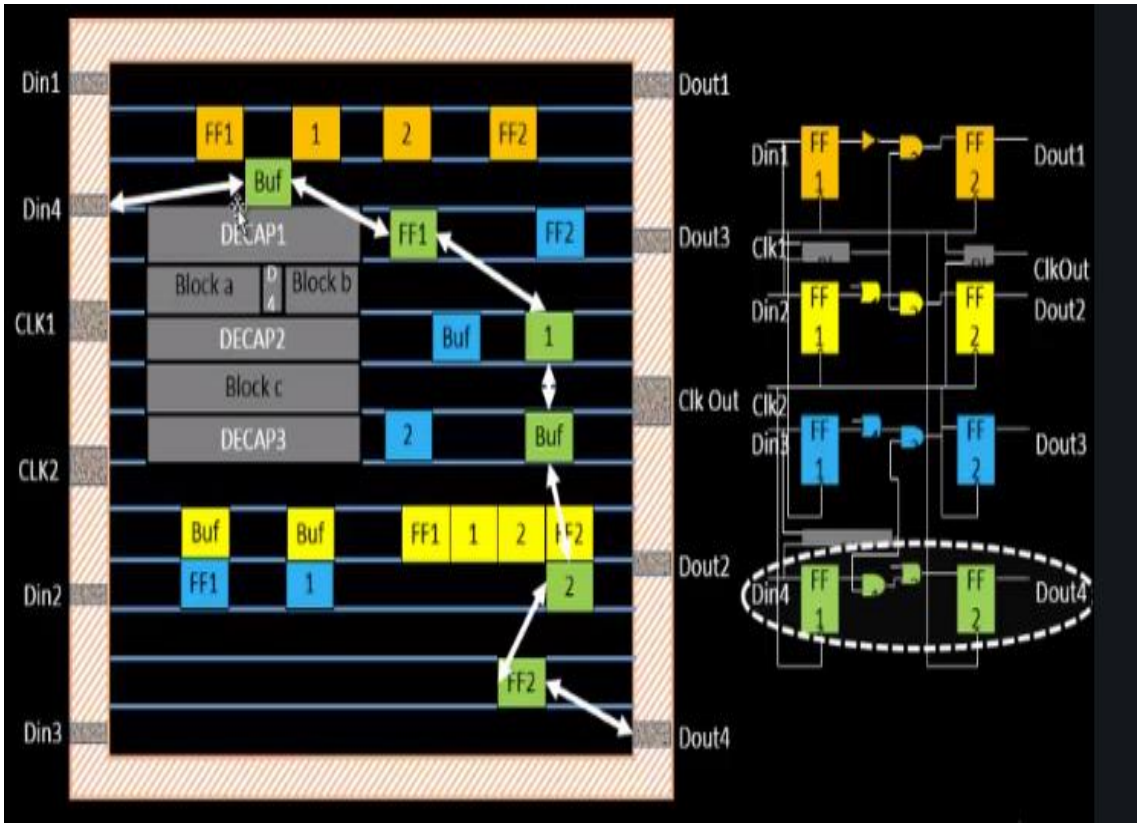



Fig. 4.5 Inserting buffers to optimize placement

# CHAPTER 5 : CLOCK TREE SYNTHESIS

## 5.1 CLOCK TREE ROUTING AND H -TREEE ALGORITHM

As we can see in the netlist clk1 goes to two FF1 and two FF2. Our purpose is to make this clock reach these flops. Let's blindly connect them and see what happens. Clock wire is basically a physical wire. Resistances and capacitances come along with this wire. As shown in the figure we connected the clock wires based on the connectivity information provided in the netlist.

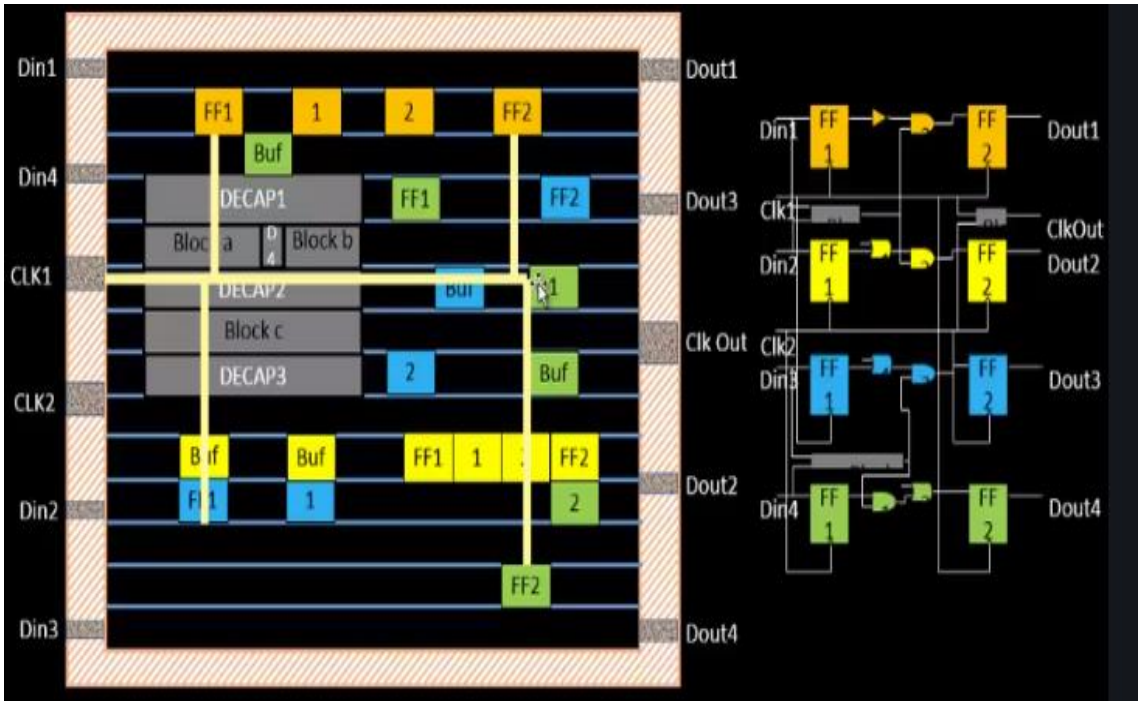



Fig. 5.1 Random clock tree synthesis

Now comes the concept of clock tree buffering. We see all these clock networks are wires with their respective resistances and capacitances associated with them as shown in the figure below. As we can see because of this RC network the original signal of clock is not reproduced properly.

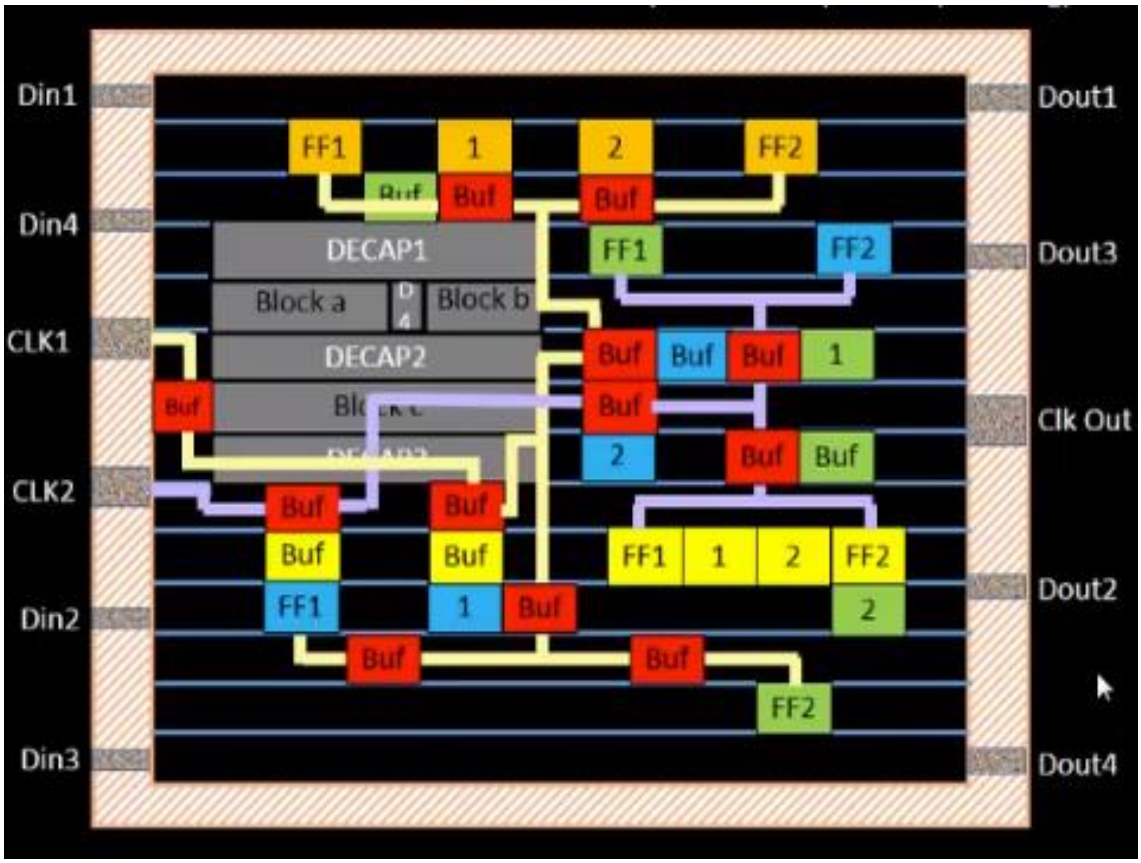



Fig. 5.2 Clock tree buffering

## 5.2 SETUP TIMING ANALYSIS

Till now we built a clock tree network in such a fashion that it maintains zero skew between launch and capture flop. Now after doing clock net shielding and preventing our clock nets from crosstalk effect we go to the next step which is static timing analysis. As shown in the figure below we have buffers in the clock path which offers some delay since we are doing timing analysis with real clocks. Due to these buffer delays the clock edge which was supposed to reach launch flop at $0^{th}$ second now reaches after delay of two buffers. Here 1+2 is launch flop clock network delay and 1+3+4 is capture flop clock network delay. The first edge of clock which was supposed to be at $0^{th}$ is now shifted and similarly the right edge of clock which was supposed to be at T is also shifted. Concept of setup and uncertainty will still hold good and we have to consider them as well. Any circuitry on the chip satisfying this setup equation is ready to work at 1Ghz. Difference of Data required time and Data arrival time is slack. If timing is violated we get negative slack. 0 or positive slack is desired.

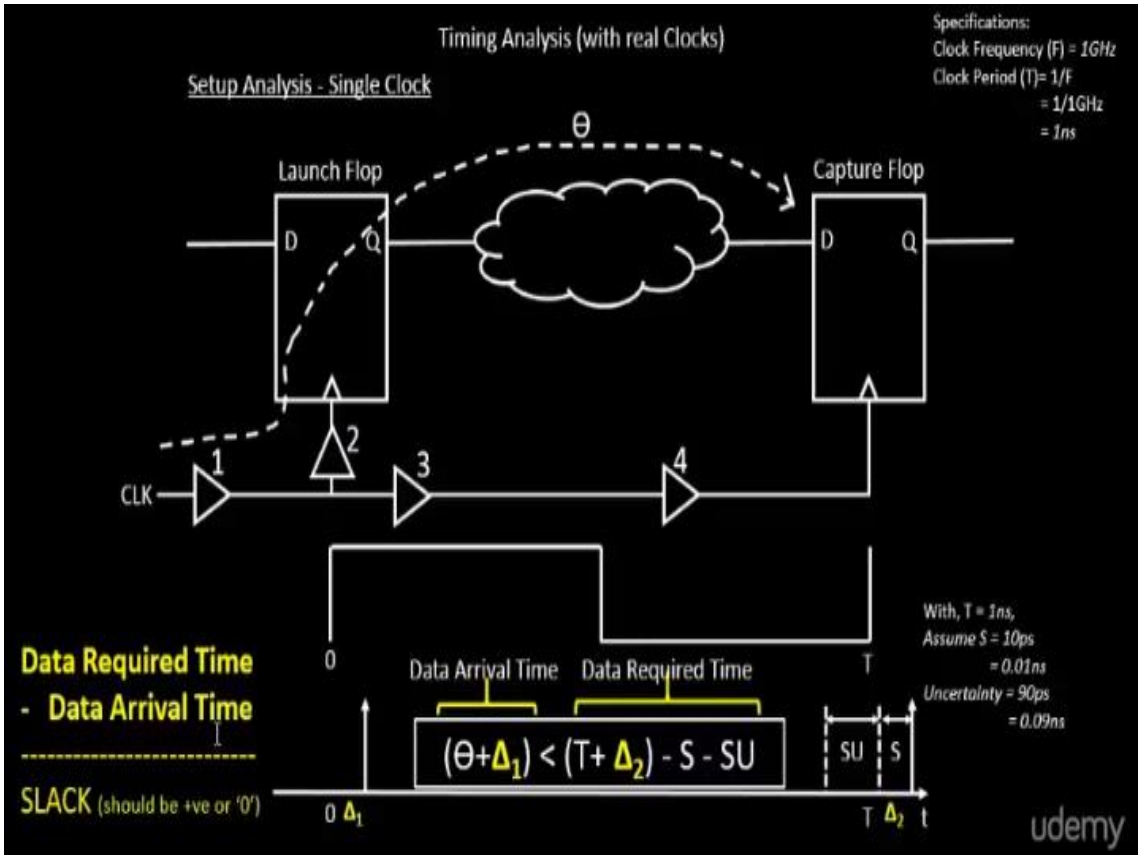



Fig. 5.3 Setup time violation

## 5.3 HOLD TIMING ANALYSIS

It is different from the static timing analysis as the first edge of clock is sent to both launch and capture flop to check for hold timing analysis. Hold condition says that combinational delay of the circuit should be greater than the hold time of the capture flop. By considering real clock we have buffers which offer delay in the clock path and hold time condition changes accordingly. Now clock edges shift by respective amount of time due to buffer delays. Here uncertainty has no meaning as such since it is the same clock edge which is going to launch and capture flop so jitter effect is same for both the flops. That's why uncertainty value is quite low here. Adding hold time of capture flop and uncertainty we arrive at our hold time equation. Here definition of slack is in contrast with the setup time case. Here data arrival time is expected to be greater than data required time.

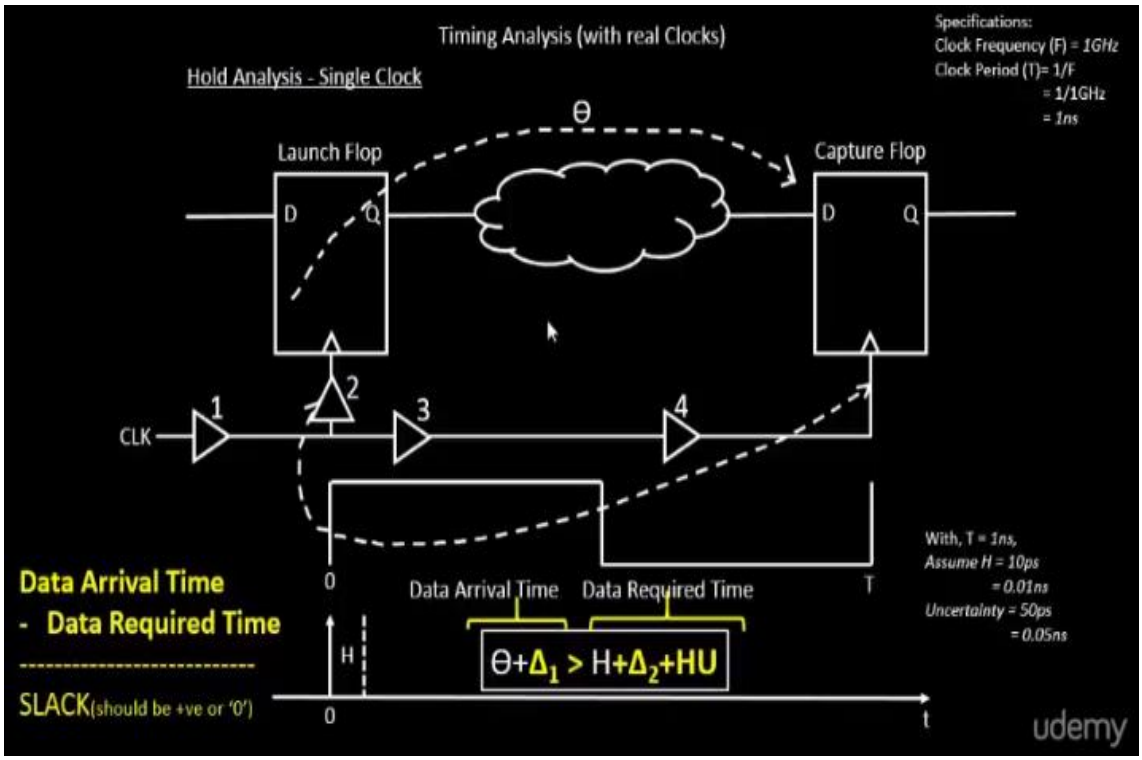



Fig. 5.4 Hold time violation

# CHAPTER 6 : ROUTING AND DESIGN RULE CHECK

## 6.1 MAZE ROUTING LEE'S ALGORITHM

Routing means that two points which are connected in netlist should be connected by physical wires. Routing algorithm has to find the perfect way to connect source and destination. Now algorithm creates a grid at the backend by taking into consideration the two points which are to be routed. Algorithm takes acre that the routing path is not of zigzag shape and it is of L shape. On point of connection is called source and the other target. So according to algorithm we take the source point and mark all the adjacent grids to it as 1. Similiarly we mark all the adjacent grids to all the 1 named grid as 2 as we can see in the figure below.

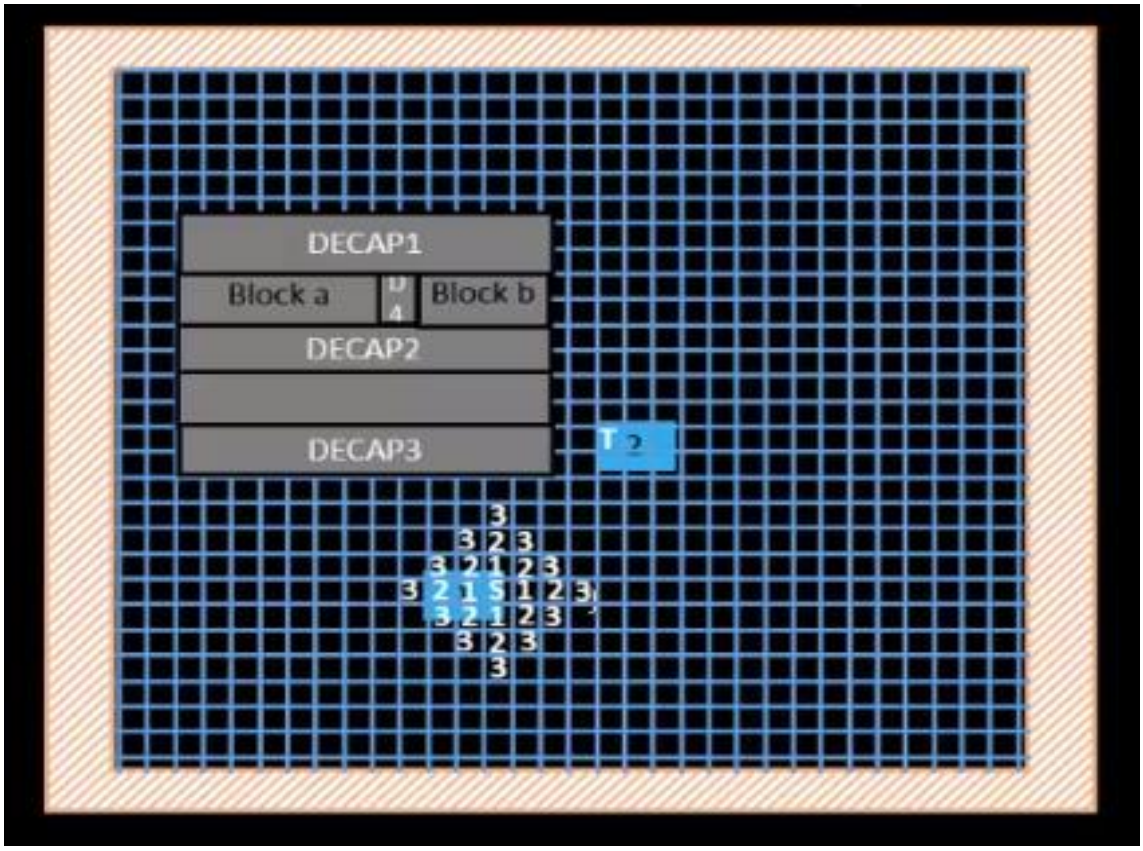



Fig. 6.1 Lee's algorithm

By following the same procedure we have number of paths from source to target as shown in the figure below. So we come down to 2 paths. The first path as shown has more zig zag path. The second path is preferred because it has least number of bends. So it is a L shaped routing and it is preferred in Lee's algorithm. It's disadvantage is that for more number of routing paths as happens in a chip it is very time consuming and it has to store all the grid information so it takes lots of memory.

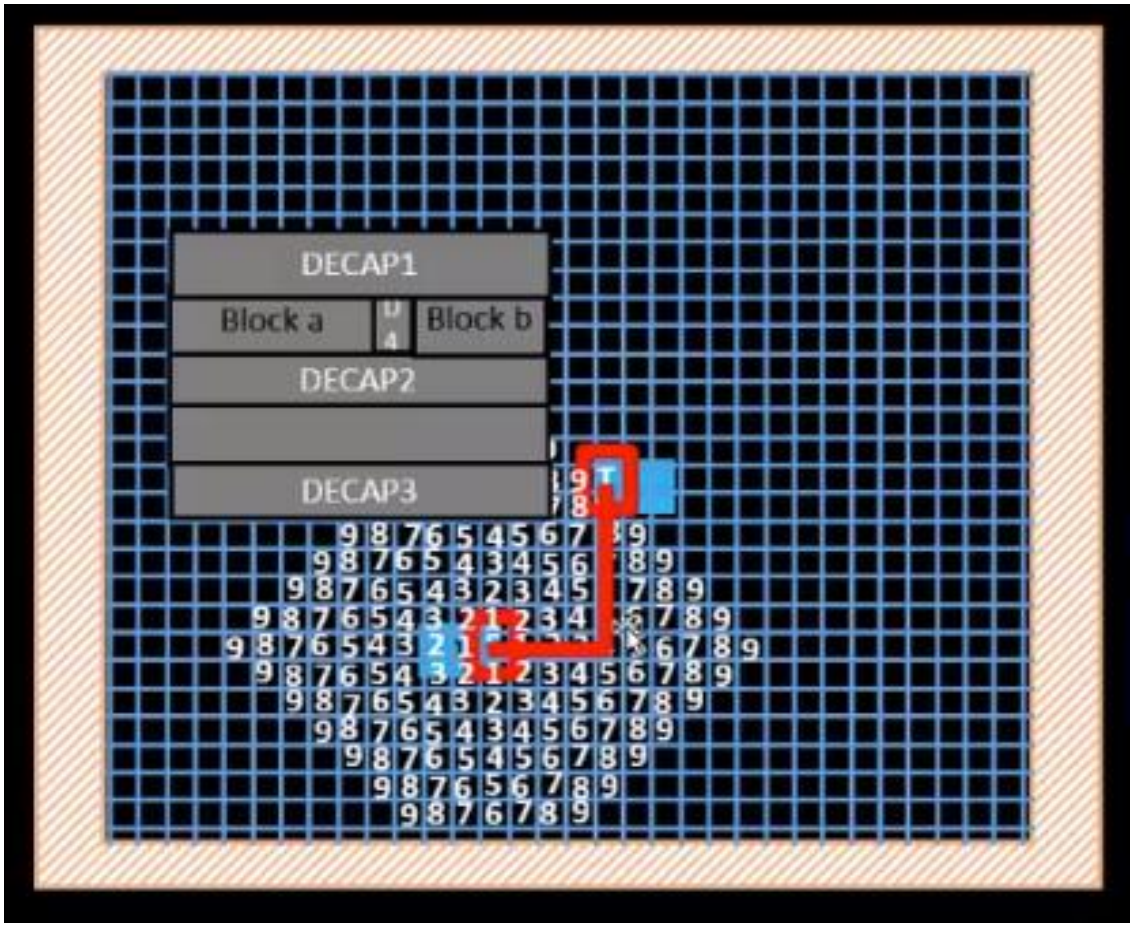



Fig. 6.2 L shaped route

## 6.2 DESIGN RULE CHECK

As shown in the figure two routing wires are being expanded. There are certain rules which we need to follow while doing any kind of routing. We will have a look at few of them in this section. One of basic design rule is to have minimum wire width. Now wire width is decided by optical lithography. Light has got some wavelength which is used to draw the routing wire. Centre to centre distance of two routing wires should also be of some fixed distance which is called wire pitch. Third basic rule is to have minimum wire spacing between two routing wires.

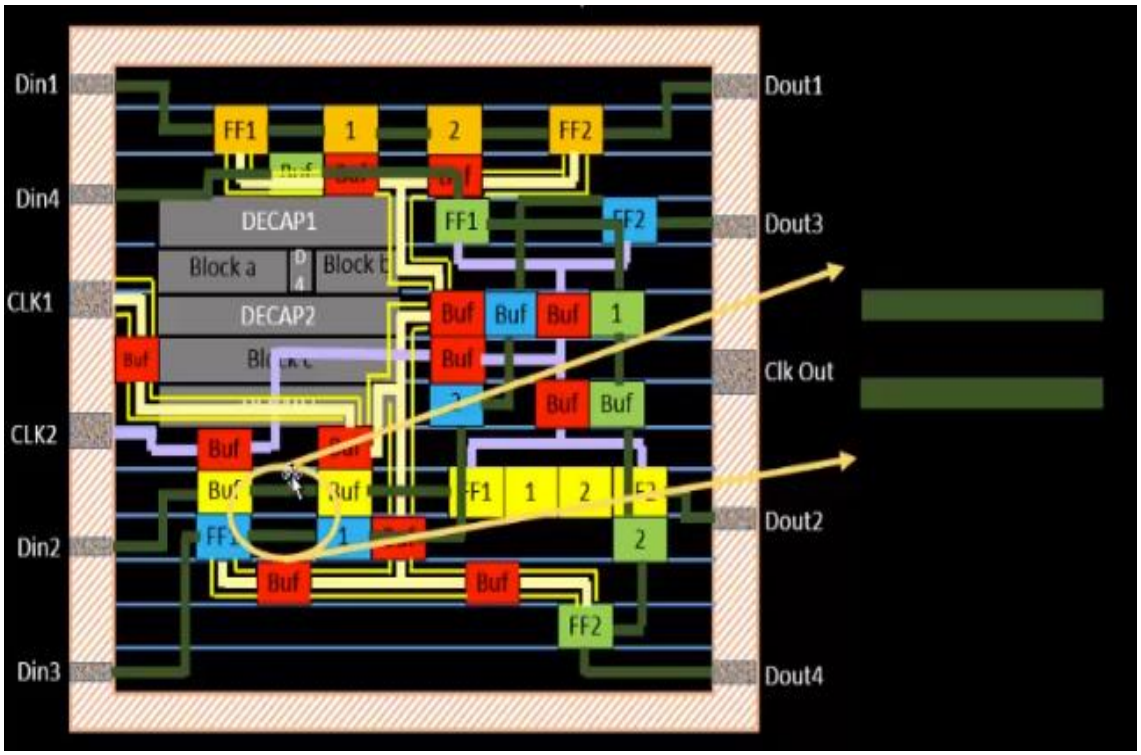



Fig. 6.3 Minimum length between routes

Now look at the different set of problems in DRC cleaning. As shown below this type of problem is called Signal Short. These kind of violations may lead to functional failure. Solution to resolve these DRC shorts is to take one metal layer Mn and we introduce one more metal layer Mn+1 at the top of Mn. While we solve this problem of signal short there are few more DRC rules which come into picture. Two new DRC checks come in to picture. One is the via width and it should be of some minimum value. All DRC rules come from point of view of optical lithography. Second one is to have minimum via spacing.

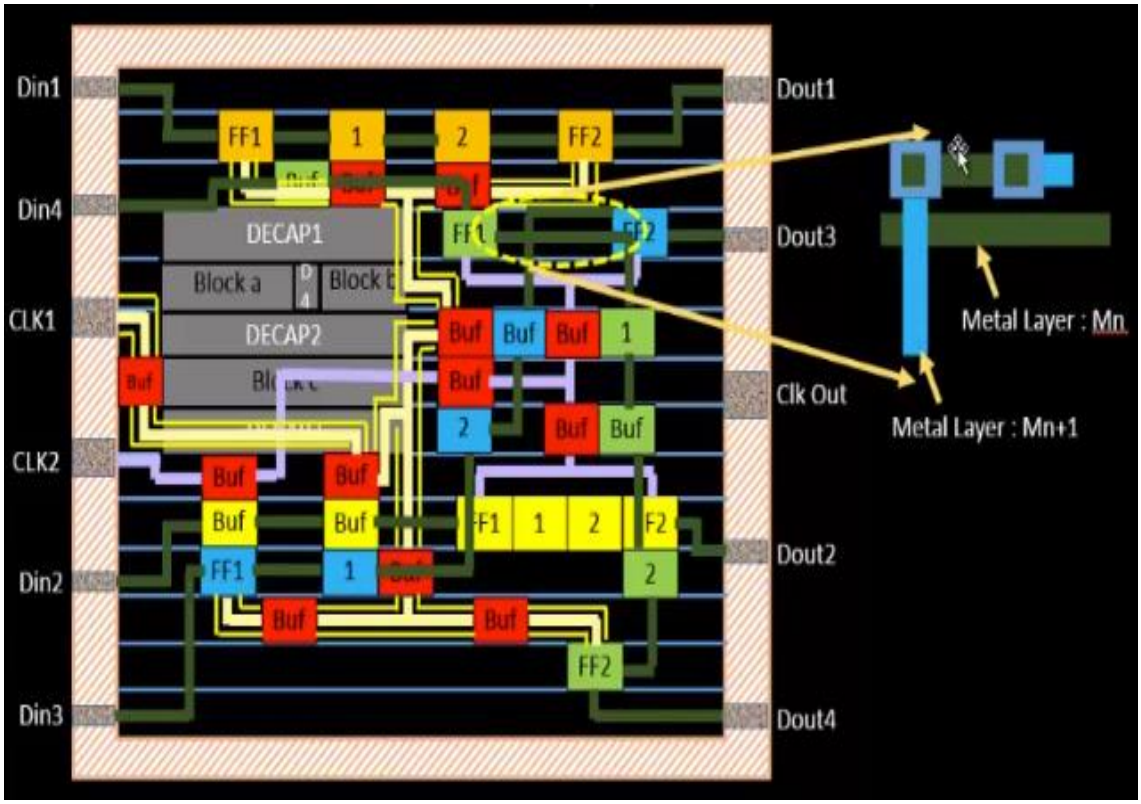



Fig. 6.4 Metal short

# CHAPTER 7 : CHALLENGES AND SOLUTIONS

## 7.1 CHALLENGE 1

As we can see in the netlist clk1 goes to two FF1 and two FF2. Our purpose is to make this clock reach these flops. Let's blindly connect them and see what happens. Clock wire is basically a physical wire. Resistances and capacitances come along with this wire. As shown in the figure we connected the clock wires based on the connectivity information provided in the netlist.

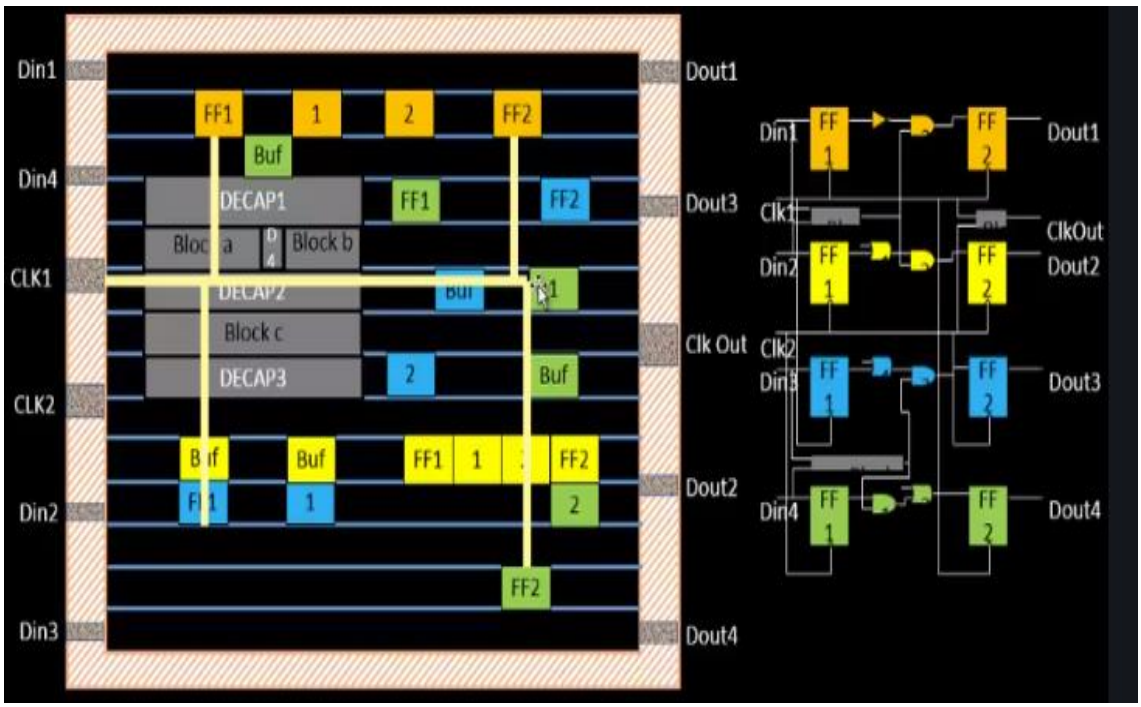



Fig. 7.1 Random clock tree synthesis

Now let's see what is the problem with that kind of clock tree network. Time required from clk1 to FF1 is t1. Based on the physical length of the wires we see time required from clk1 to FF2 is t2. t2 – t1 = skew. Ideally skew should be 0. Since it impacts the timing. That we will see while doing STA. So based on achieving minimal skew this tree is a bad tree.

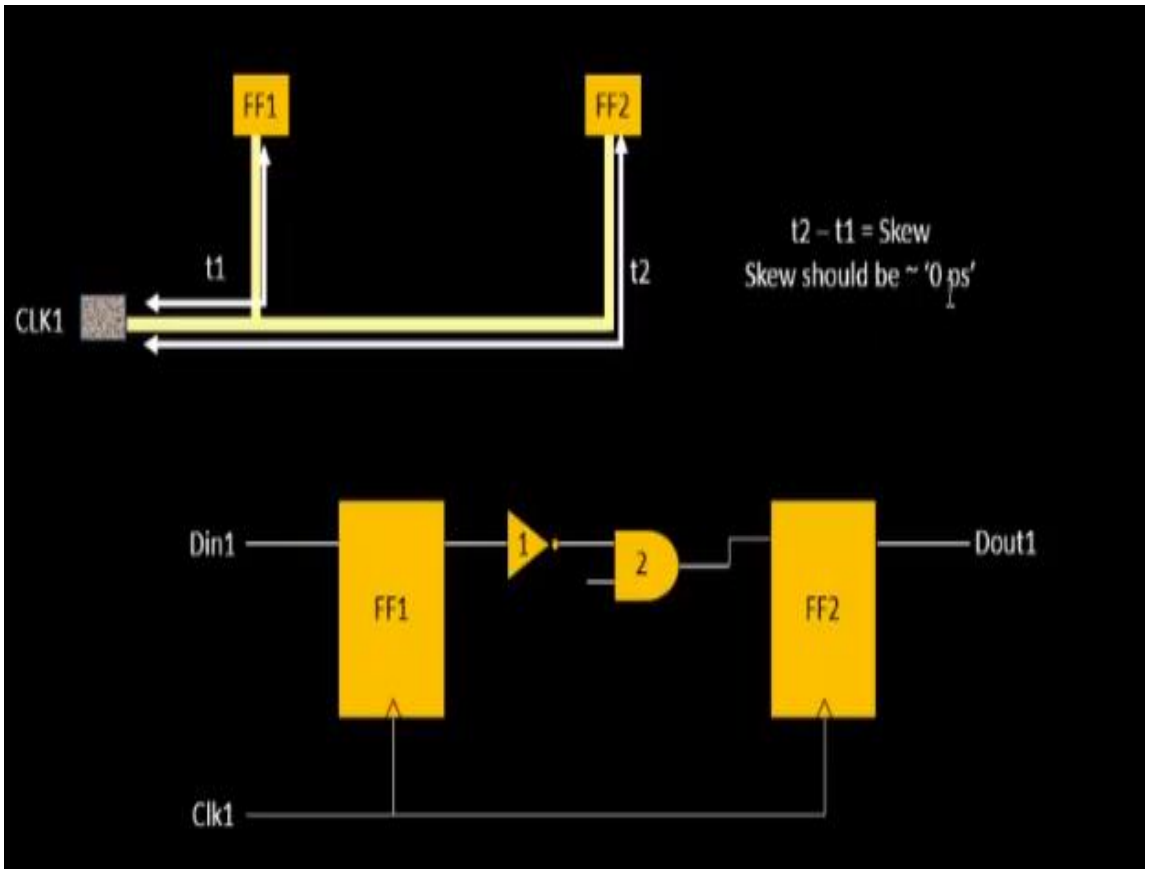



Fig. 7.2 Non zero skew due to random clock tree synthesis

## SOLUTION :

Now there is a algorithm for clock tree synthesis known as H-Tree algorithm. It basically calculates the distance to all the flops and mark the midpoint of that network. With this midpoint strategy now it is almost sure that clock reaches all the flops almost at the same time. As shown in the figure we apply the same H-Tree algorithm strategy to clk2 also. So to recondition clock signal and reproduce the same signal at the flops we add repeaters or buffers at the respective distance in between.

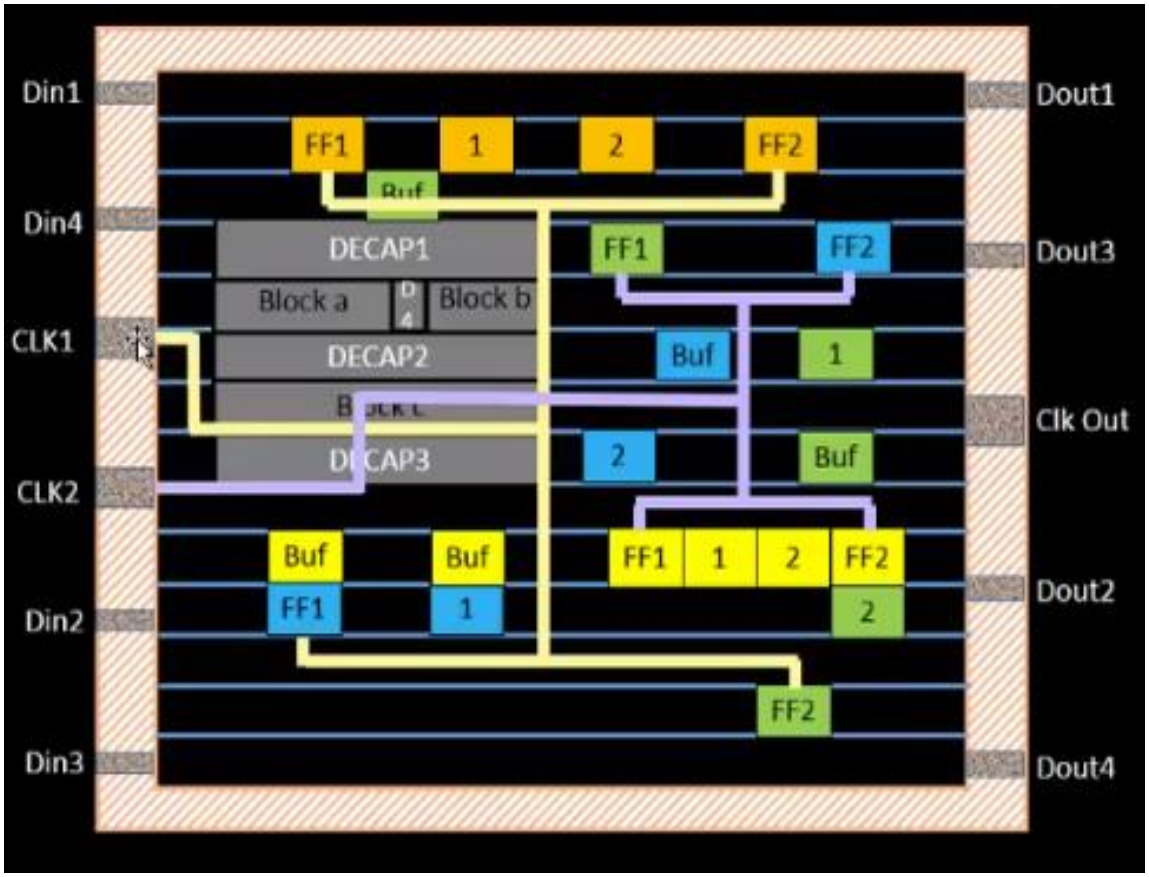



Fig. 7.3 H Tree algorithm

## RESULTS :

Table 1: Comparison of results before and after H tree algorithm

| PARAMETER | BEFORE H TREE ALGORITHM (ns) | AFTER H TREE ALGORITHM (ns) |
|---|---|---|
| CLOCK SKEW | 113 | 0 |
| SETUP TIME SLACK | -412.98 | 2.49 |
| HOLD TIME SLACK | -98.57 | 0.12 |

## 7.2 CHALLENGE 2

Clock tree synthesis implies that clock has to reach every flop in the circuit at the respective time. Clock is expected to reach every flip flop in the circuit at the same time. If it is not so then there will be skew which may lead to timing violation. We also add buffers in the clock path when there is huge wire length as the number of capacitors that clock signal has to charge becomes huge and there may be transitions because of that. So to maintain originality of the signal we add repeaters in the path. Signal keeps on getting reconstructed with the help of buffers. But if there is coupling between two clock nets then there are couple of problems which may occur and we need to deal with them.

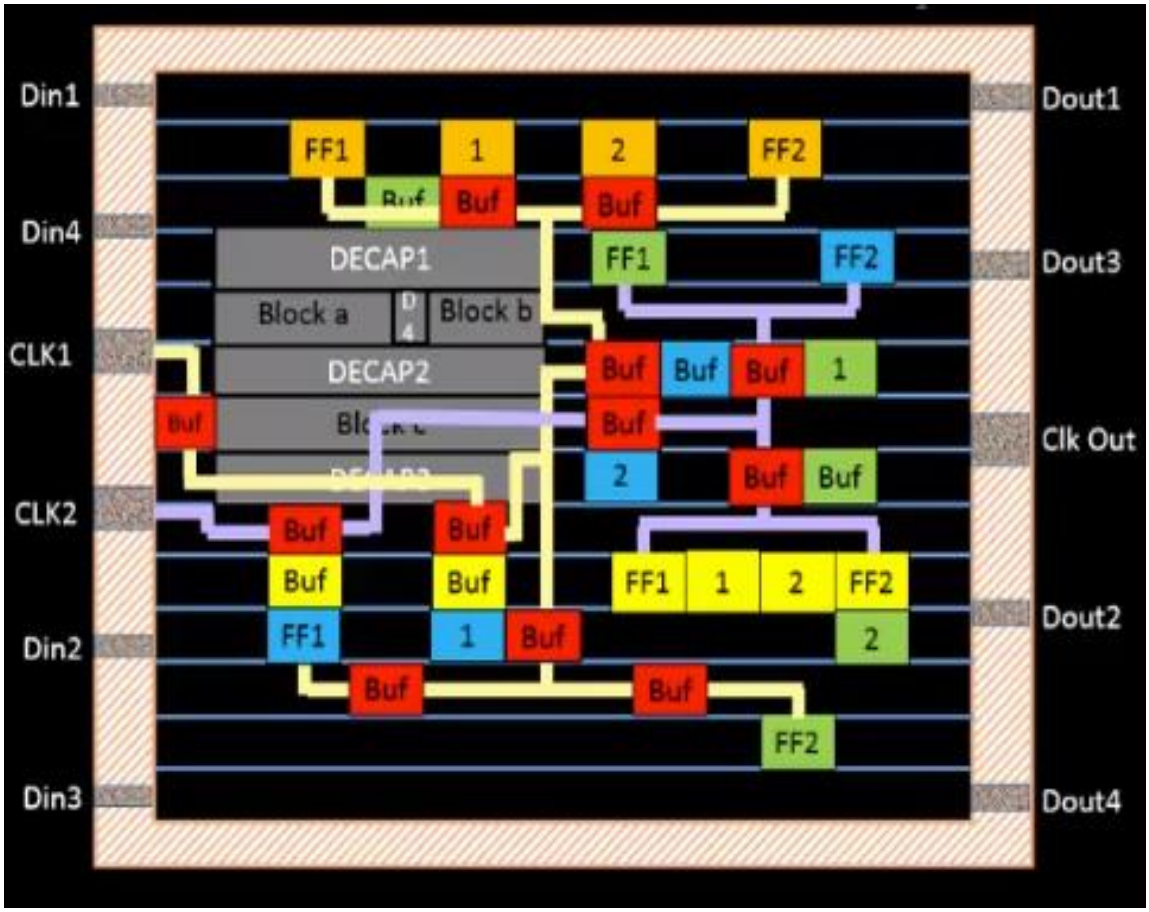



Fig. 7.4 Clock tree synthesis without shielding

First problem that we face due to crosstalk is the glitch. Consider one of the clock net as shown. Whenever there is switching activity happening on the aggressor and coupling capacitance becomes so strong that any activity happening will directly impact the clock net sitting close to it. Our victim was supposed to be at logic 0 but due to aggressor switching from logic 1 to 0 we see there is a dip in the voltage at victim and as a result we see a glitch. Now because of this glitch out reset pin is set high which sets our memory to all 0. This memory could be the part of any critical chip eg. Automobile chip. Incorrect data in memory will result in inaccurate functionality. Due to this complete system goes wrong. So shielding is the process which protects this victim. In shielding we connect our victim to two lines Vdd and ground because these are 2 lines which won't switch and as a result of that our victim is protected.

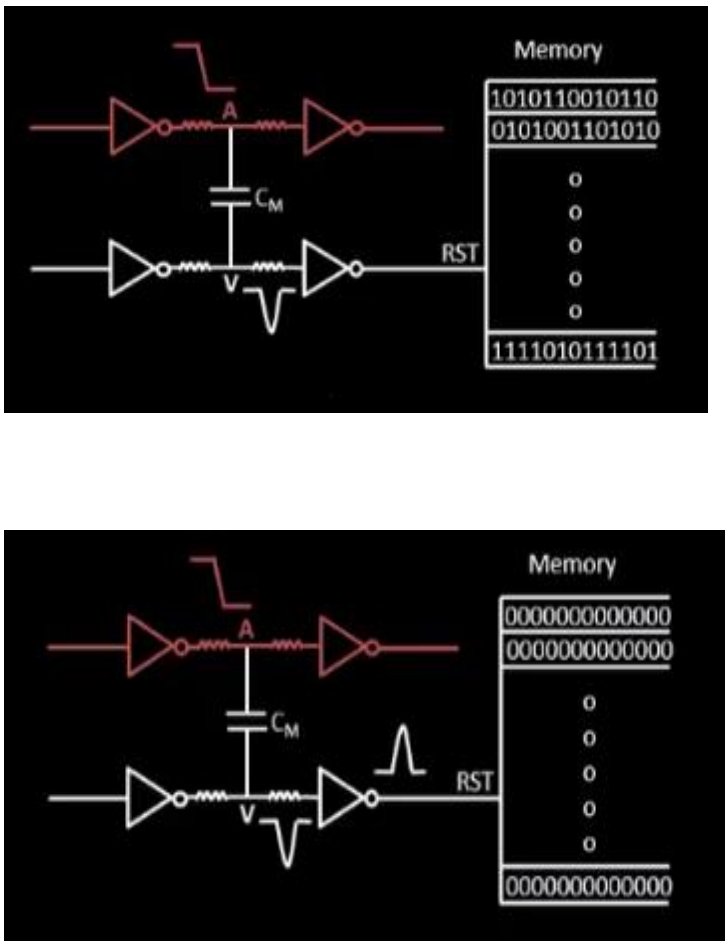



Fig. 7.5 Glitch due to crosstalk

Now second problem due to crosstalk is if victim itself switches and aggressor is also switching. Delta delay comes into picture in such case. Now as we can see in the figure in absence of crosstalk clock latency L1 and L2 are equal. As we can see output of invertor in victim net is going from logic 1 to logic 0. Now because of switching activity on the aggressor it impacts the transition of victim net by some amount of delay which is called Delta delay. Now because of this bump the delay of that particular cell gets impacted by an amount equal to delta delay. Now clock latency becomes L2+delta instead of L2. Now skew is no more 0. It is of some finite value equal to delta. Here we have talked about for a handful of buffers but for a multimillion chip this delta may grow exponentially. Idea is to basically shield the critical nets. Since clock net is critical so we shield them. If there are some data nets which are also critical then we also might need to shield them. It is not possible to shield all the nets because that will increase the routing resources.

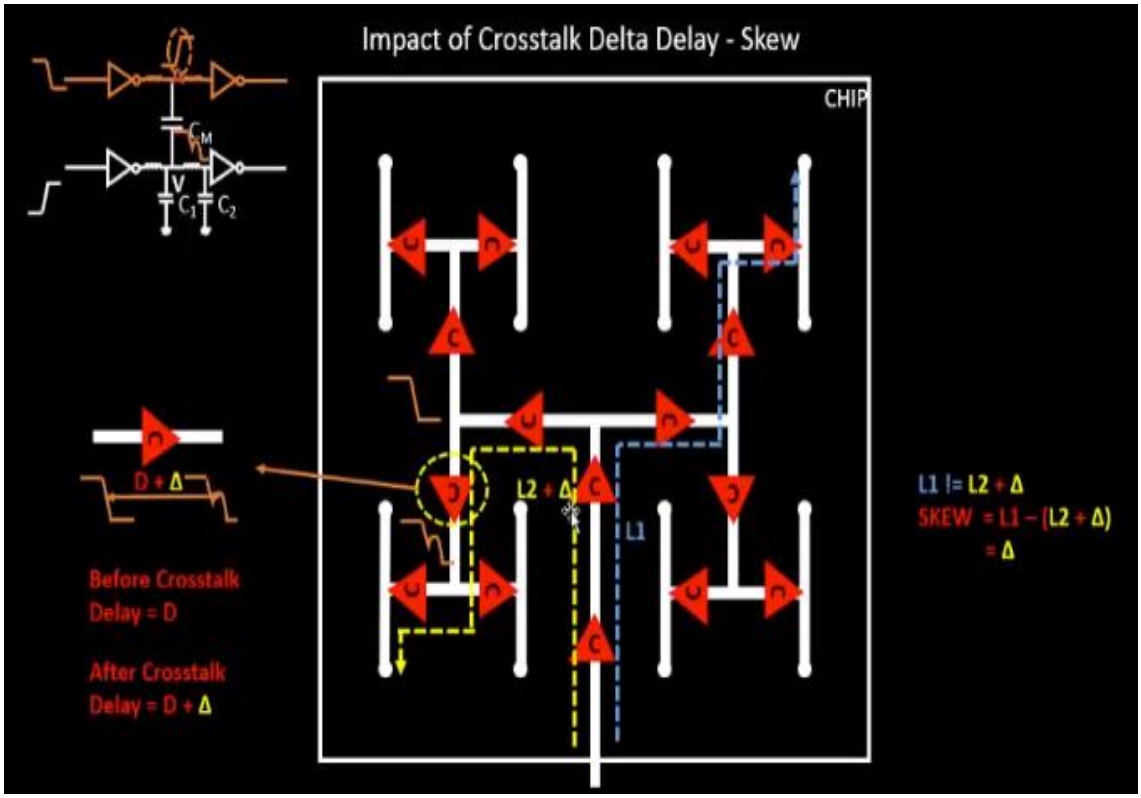



Fig. 7.6 Delta delay due to crosstalk

## SOLUTION :

Till now clock tree has been built in such a fashion that it maintained zero skew between launch and capture flop. Clock tree maintains a zero skew. There is a phenomenon of crosstalk that can happen. To prevent this we take a clock net and shield it. By Shielding we protect the clock net from the outside world. Thus the problems of glitch and delta delay due to crosstalk can be prevented by shielding the clock nets in a fashion as shown in the figure below.

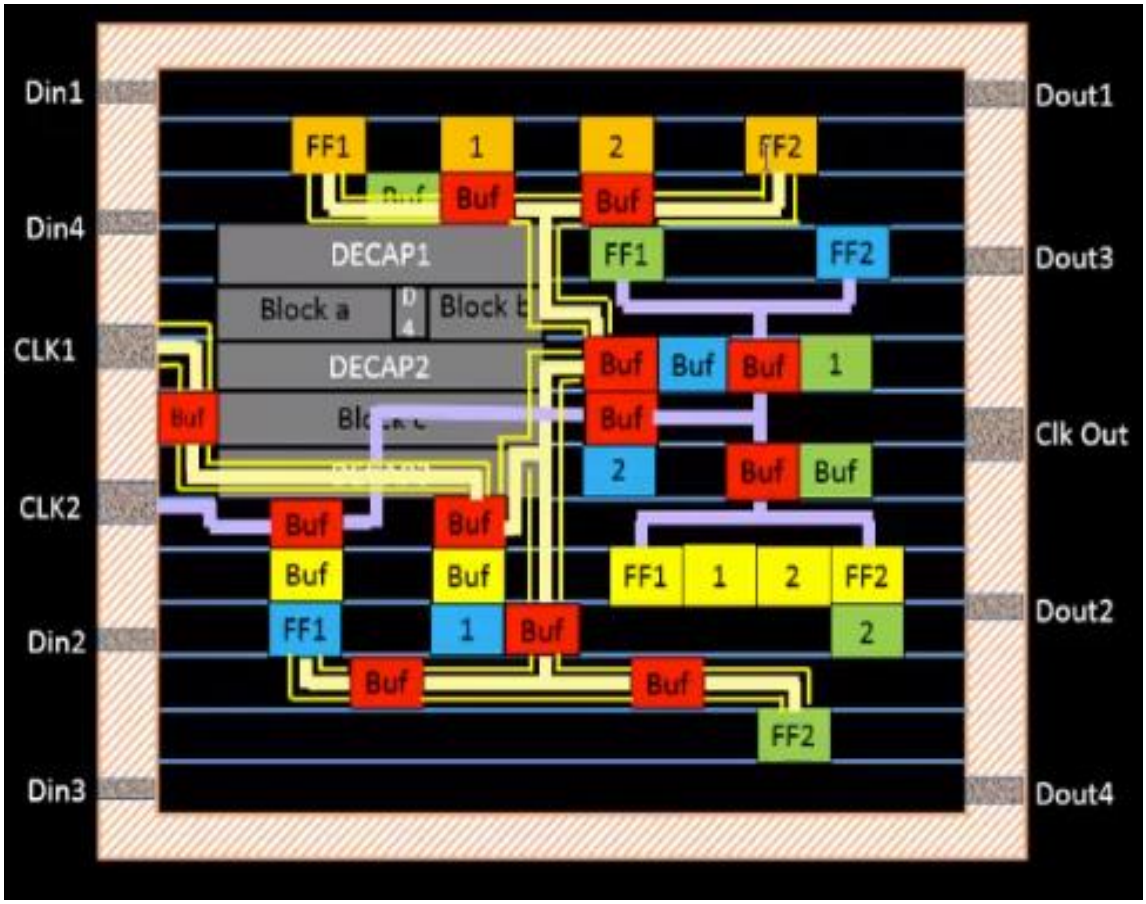



Fig. 7.7 Clock net shielding

**RESULTS :**

Table 2: Comparison of results for before and after clock net shielding

| PARAMETER | BEFORE CLOCK NET SHIELDING (ns) | AFTER CLOCK NET SHIELDING (ns) |
|---|---|---|
| DELTA DELAY (CLOCK SKEW) | 56.78 | 0 |
| SETUP TIME SLACK | -9.99 | 0.05 |
| HOLD TIME SLACK | -10.83 | 1.19 |

## 7.3 CHALLENGE 3

Now look at the circuit shown below. It is a sub circuit of an hm in our core. Here cgc stands for clock gating circuit which is placed between mux and clock divider circuit. Now as we can see the routing length of wires between port 1,2 and mux 1,2 is pretty long. Same is case for the routing length between clock divider 1,2 and mux 3,4 at the output of the circuit as shown. These long routing lengths lead to higher delay in the signal passing through these wires. As a result we face a higher clock latency. Since clock latency increases so we face setup and hold slack issues which in return disturbs the timing of the circuit.

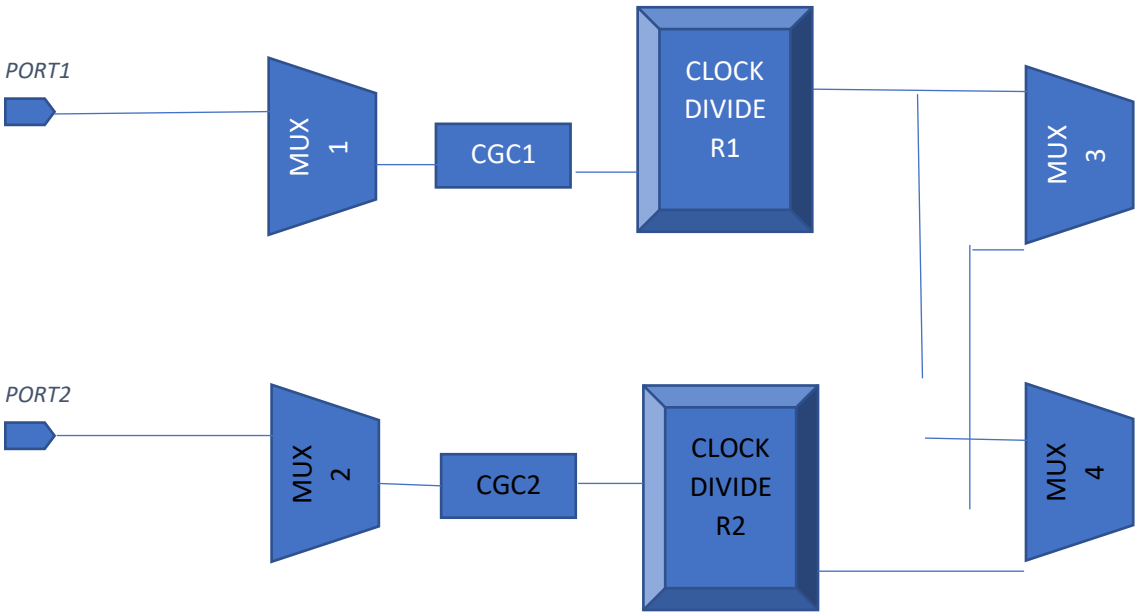



Fig. 7.8 Sub circuit of a hm of our core logic area with long routing wires

## SOLUTION :

Now we look at the solution to fix those timing issues which we faced with the long routing wires at the input and output stages of hm of our core logic in earlier case. As we can see in the figure below we shortened the routing lengths between ports 1,2 and mux 1,2 and similarly the length of routing wires between clock divider 1,2 and mux 3,4 is also reduced by a significant amount in a range which does not cause congestion in the circuit. DRCs are also taken into consideration while shortening the wires of route at these input and output stages of this sub circuit. Thus as a result of shortening the length of wires delay is reduced and thus clock latency is reduced as a result. This avoids any violation in timing of the circuit.

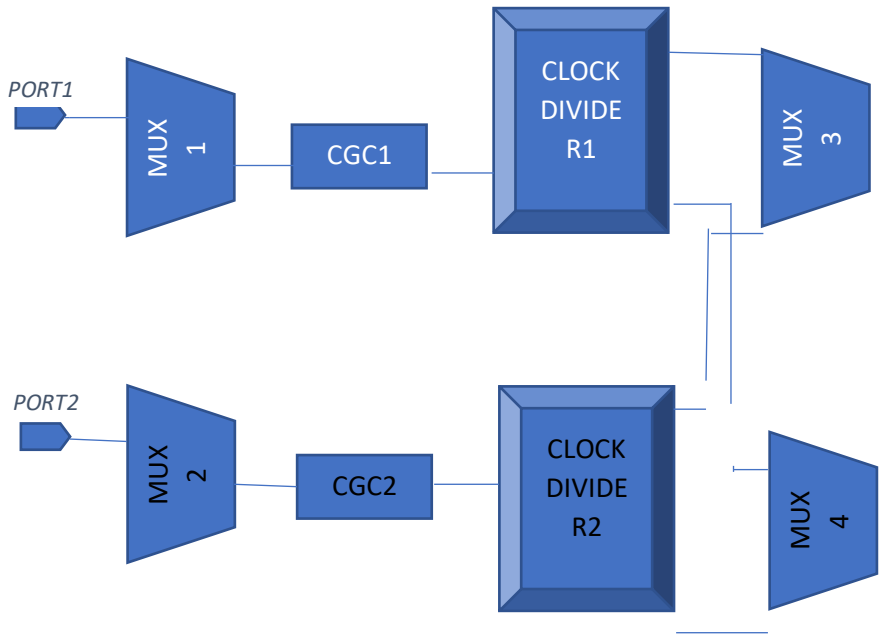



Fig. 7.9 Shorter routing wires of sub circuit of hm of core logic

**RESULTS :**

Table 3: Comparison of results before and after optimization

| PARAMETER | BEFORE OPTIMIZATION | AFTER OPTIMIZATION |
|---|---|---|
| LENGTH OF ROUTING (nm) | 908.08 | 475.71 |
| CLOCK LATENCY (ns) | 1029.90 | 567.20 |
| SETUP TIME SLACK (ns) | -132.99 | 0.08 |
| HOLD TIME SLACK (ns) | -89.06 | 8.83 |

# CHAPTER 8 : CONCLUSION AND FUTURE SCOPE

Physical Design, in simple words, determines the shape, size and power and timing of the design. I tried to learn as many things I could. It all begins with floorplan and it is the base of all the further steps to come. If the floorplan is done intelligently, many problems can be reduced. PRO gives us the actual drcs and the total timing. Accurate timing is obtained in PT run, but most of the changes to reduce this timing errors are to be made in PRO itself. It may look simple flow, but here are lot of things going on in it and many things require manual attention. Scripting does play an important role in all this process.

Thus as with the help of various techniques we were able to tackle various challenges that we faced during the course of internship in physical design team. Future scope of this domain can be discovering few more techniques based on the challenges we face in the future. Hence we can have variety of florplanning, place and route and timing problems which can be solved based on these. It may also help to improve our timing ananlysis so that it improves setup and hold slack positive and helps preventing failure of further stages in fabrication of the chip.

# REFERENCES :


[1] A. Kahng, J. Leinig, I. L. Markov, and J. Hu, "VLSI Physical Design: From Graph Partitioning to Timing Closure," Springer, ISBN 978-90-481-9590-9, May 2011

[2] B.Sowmya, S. M.P., "Minimization of Floorplanning Area and Wire Length Interconnection Using Particle Swarm Optimization," International Journal of Emerging Technology and Advanced Engineering, vol. 3, Issue 8, August 2013.

[3] J. Chen, W. Zhu, and M.M.Ali, "A Hybrid Simulated Aneealing Algorithm for Nonslicing VLSI Floor-planning," IEEE Trans.on Systems,Man and Cybernetics, Part C, vol.41, No.4, July 2011,pp. 544– 553.

[4] P.N.Guo,C.K.Cheng, and T.Yoshimura, "An O-tree representation of non-slicing floorplan and its applications," Proceedings of 36th ACWIEEE Design AutoNation Cont., 1999, pp.268-273.

[5] Y. Ma, S. Dong, X. Hong, Y. Cai, C. K. Cheng, and Jun Gu, "VLSI Floor-planning with Boundary Constraints Based on Corner Block List," IEEE, 2001, pp. 509- 514.

[6] S. N. Adya et al., ''Benchmarking for large-scale placement and beyond,'' IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., vol. 23, no. 4,20014, pp. 472–487.

[7] S. N. Adya and I. L. Markov, ''Combinatorial techniques for mixed-size placement,'' ACM Trans. Design Autom. Electron. Syst., vol. 10, no. 1,2005, pp. 58–90.

[8] U. Brenner and A. Rohe, ''An effective congestion driven placement framework,'' in Proc. Int. Symp. Phys. Design, 2002, pp. 6–11.

[9] U. Brenner and J. Vygen, ''Faster optimal single-row placement with fixed ordering,'' in Proc. Design Autom. Test Eur. Conf. Exhibit., 2000, pp. 117–121

[10] U. Brenner and M. Struzyna, ''Faster and better global placement by a new transportation algorithm,'' in Proc. Design Autom. Conf., 2005, pp. 591–596.

[11] V. Betz and J. Rose, ''VPR: A new packing, placement and routing tool for FPGA research,'' in Field-Programmable Logic and Applications, ser. Lecture Notes in Computer Science, vol. 1304. Berlin, Germany: Springer-Verlag, 1997, pp. 213–222.

[12] C. J. Alpert, ‘‘What makes a design difficult to route,’’ in Proc. Int. Symp. Phys. Design, 2010, pp. 7–12.

[13] T. Lin and C. Chu, ‘‘POLAR 2.0: An effective routability-driven placer,’’ in Proc. Design Autom. Conf., 2014, pp. 1–6

[14] W.-H. Liu, W.-C. Kao, Y.-L. Li, and K.-Y. Chao, ‘‘Multi-threaded collision-aware global routing with bounded-length maze routing,’’ in Proc. Design Autom. Conf., 2010, pp. 200–205.

[15] W.-H. Liu, C.-K. Koh, and Y.-L. Li, ‘‘Optimization of placement solutions for routability,’’ in Proc. Design Autom. Conf., 2013, pp. 1–9

[16] E. G. Friedman, High Performance Clock Distribution Networks, Kluwer Academic Publishers, 1997

[17] Cadence support: https://support.cadence.com , last accessed on 14 June 2019

[18] Synopsys: https://solvnet.synopsys.com , last accessed on 16 June 2019

[19] http://www.vlsi-basics.com/2014/02 , last accessed on 12 June 2019

[20] https://www.techdesignforums.com/practice/sphere/techniques, last accessed on 11 June 2019

[21] http://www.signoffsemi.com/47072-2 , last accessed on 11 June 2019

[22] Udemy website link for figures and related applications of floorplanning and https://www.udemy.com/vlsiacademyphysicaldesignflow/learn/lecture/2162514#overview

# Electronic Design Automation

ORIGINALITY REPORT

| 6% | 4% | 2% | 2% |
|---|---|---|---|
| SIMILARITY INDEX | INTERNET SOURCES | PUBLICATIONS | STUDENT PAPERS |

PRIMARY SOURCES

| | | |
|---|---|---|
| 1 | Igor L. Markov, Jin Hu, Myung-Chul Kim. "Progress and Challenges in VLSI Placement Research", Proceedings of the IEEE, 2015<br>Publication | 1% |
| 2 | Submitted to VIT University<br>Student Paper | 1% |
| 3 | Submitted to CSU, San Jose State University<br>Student Paper | 1% |
| 4 | vlsisystemdesign.com<br>Internet Source | 1% |
| 5 | www.eecs.umich.edu<br>Internet Source | 1% |
| 6 | www.slideshare.net<br>Internet Source | 1% |
| 7 | shareok.org<br>Internet Source | 1% |
| 8 | lmcs.episciences.org<br>Internet Source | <1% |

| | | | |
|---|---|---|---|
| Exclude quotes | On | Exclude matches | < 10 words |
| Exclude bibliography | On | | |