VLSI DESIGN FLOW: RTL TO GDS

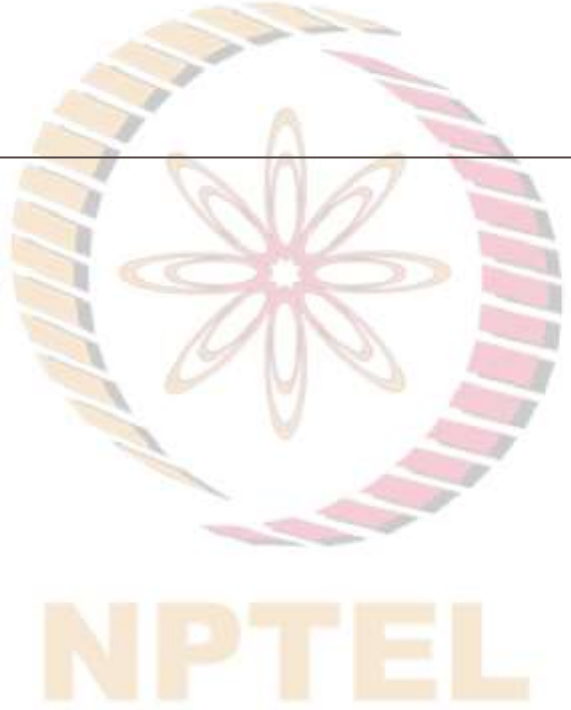
Lecture 7
Overview of VLSI Design Flow: V

Sneh Saurabh
Electronics and Communications Engineering
IIIT Delhi

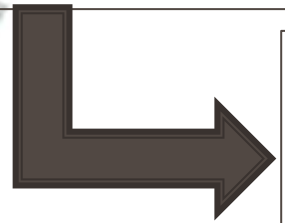
Lecture Plan

Overview of VLSI Design Flow

- Verification
- Testing

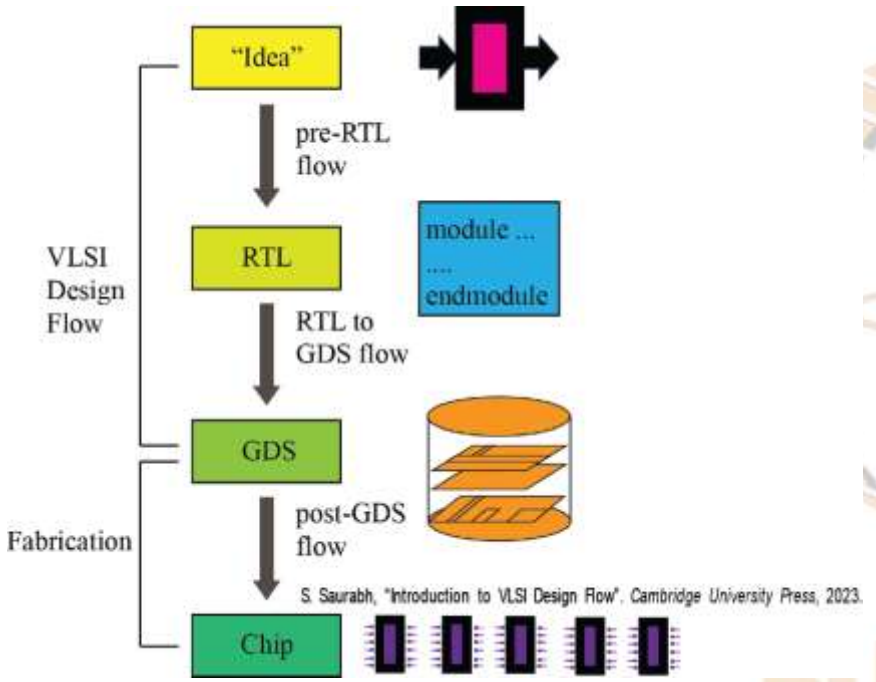


Overview of VLSI Design Flow



Verification

VLSI Design Flow: Verification



Verification:

We perform verification to ensure that the design works as per given functionality

- Verification is done multiple times throughout a VLSI design flow
  - Whenever design undergoes some changes, verification must be done
  - If verification fails remedial action can be taken immediately

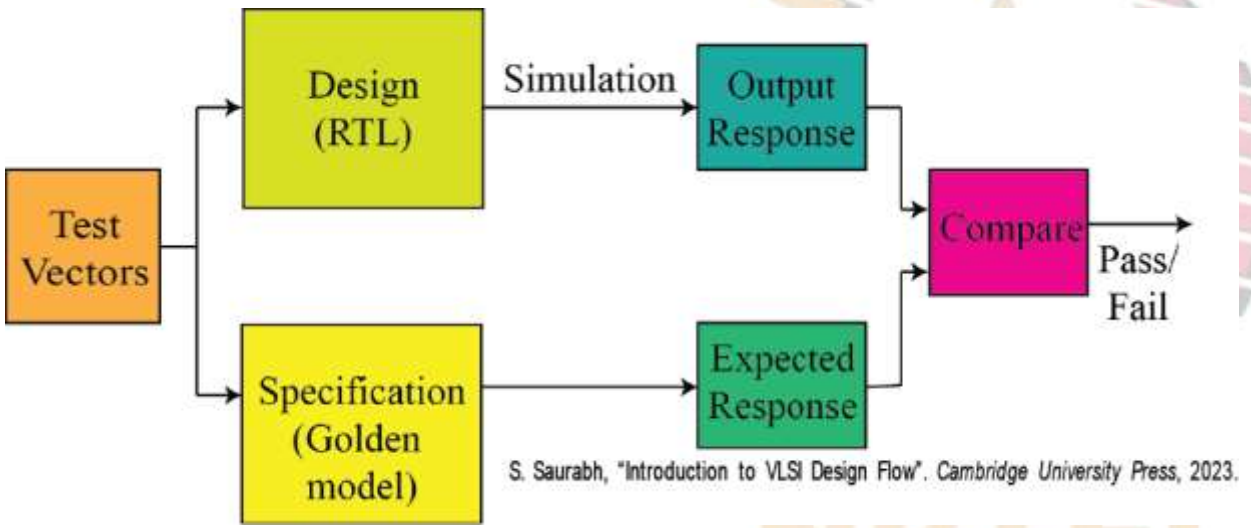
- Significant effort of VLSI design flow is spent on verification

Functional Verification: Simulation

Is design (RTL) producing the same results as given in the specification?

Simulation: technique for ensuring the functional correctness of the RTL using ***test vectors***

- Test vectors: sequence of zeroes/ones and the associated timing information



S. Saurabh, "Introduction to VLSI Design Flow", *Cambridge University Press*, 2023.

- Obtain the response for the given RTL using a simulator
- Expected output is computed using another model (C, C++, MATLAB, etc.)
- Compare output response and expected response

- Merits of simulation:-based verification: fast and versatile
- Demerit of simulation-base verification: incompleteness

Functional Verification: Model Checking

Is design (RTL) producing the same results as given in the specification?

Model checking: technique for ensuring the functional correctness of the RTL using *formal methods*

- Formal methods: establish the proof of a given property using formal mathematical tools such as deductions
- Once we have proven a property mathematically, it is guaranteed to hold for all test stimuli.

Property Checking/Model Checking

- Define *properties* that must be satisfied for a given *specification*
- Check whether *properties* are being satisfied in the implemented design RTL using formal methods

Verification using formal methods:

- Merits: completeness
- Demerits: computationally difficult

Example:

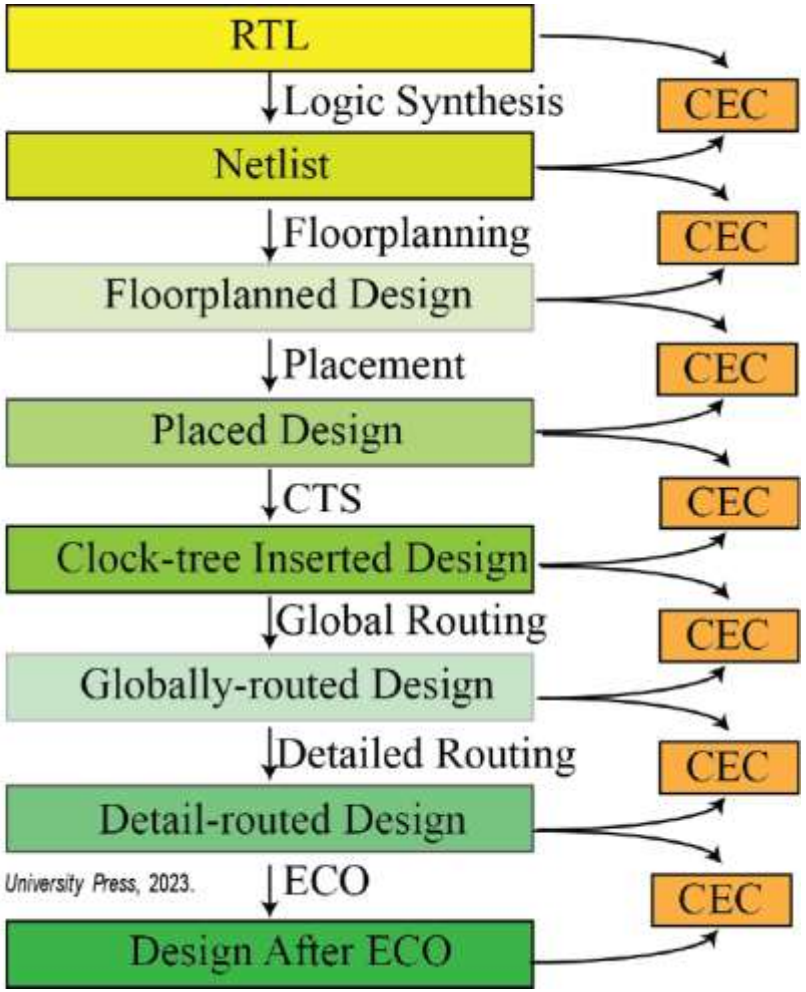
- A traffic controller should generate signals such that "*Not more than one light should be green*"
- In a resource sharing environment: "*Request for a resource should be granted eventually*"

Combinational Equivalence Checking

Will the RTL and the netlist generated by a logic synthesis tool always produce the same (equivalent) output?

Combinational equivalence checking (CEC): establishes the functional equivalence of two models using *formal methods*

- CEC required whenever non-trivial design changes occur
- Carried out multiple times in a design flow



S. Saurabh, "Introduction to VLSI Design Flow", *Cambridge University Press*, 2023.

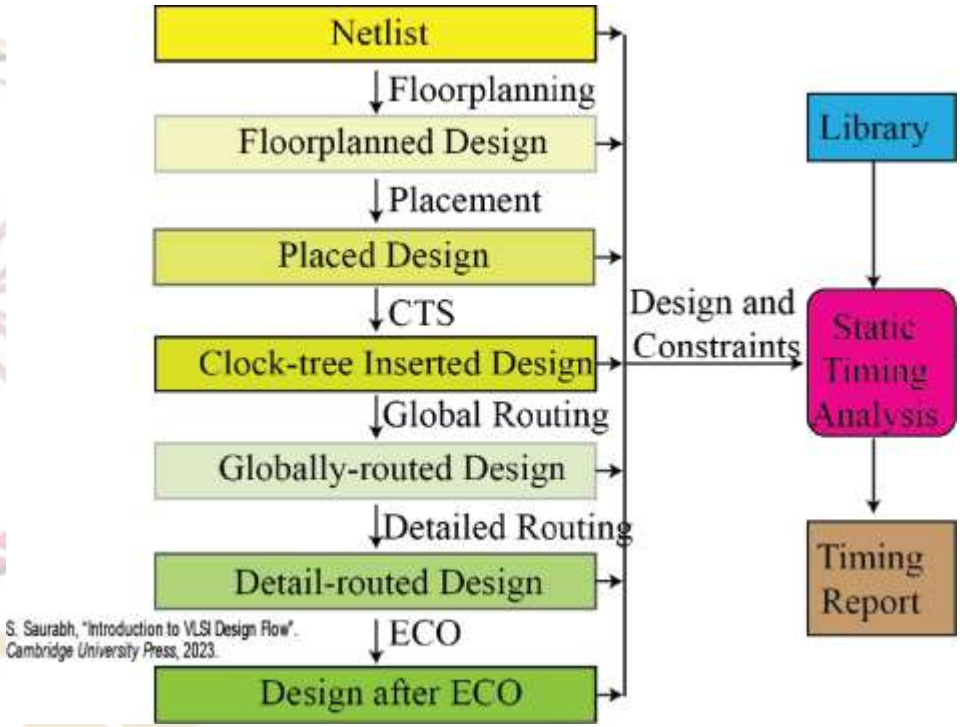
Static Timing Analysis

How to ensure the synchronicity of data transfer between flip-flops in a synchronous design?

Synchronicity: data launched by a flip-flop gets captured in the sequentially adjacent flip-flop in *the next clock cycle*

Static Timing Analysis (STA): ensures deterministic synchronous timing behavior in a circuit

- STA tools consider the worst-case behavior (may be pessimistic), but will always ensure the timing safety
- Carried out multiple times in a design flow



Physical Design Verification

How to ensure that the layout does not have manufacturing and connectivity issues and the yield for a design remains high?

Physical verification:

- Check a set of rules during physical design before sending the layout to the foundry.

Design rule check (DRC):

- Rules are defined by the foundries and depend on the manufacturing technology
- All DRC violations must be fixed before sending it to the foundry for fabrication.

Electrical rule check (ERC):

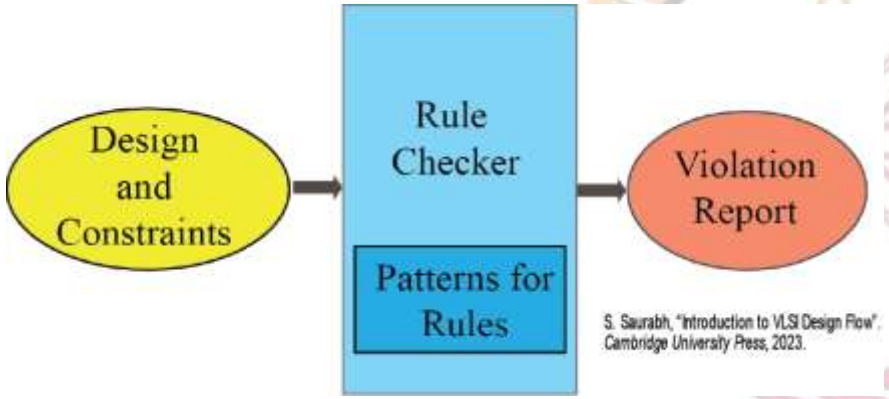
- Rules are defined to ensure proper connectivity (for e.g.: no short circuit between distinct signal lines).

Layout vs. schematic (LVS) check:

- Ensure that the layout is functionally equivalent to the original netlist

Rule Checking

Rules are some restrictions imposed on the ***design entity*** such as RTL, constraints, netlist, layout etc. such that there are no issues in using that design entity downward in the VLSI design flow

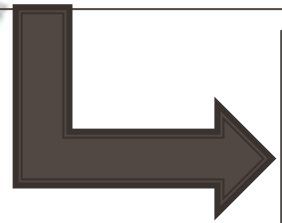


RTL: Ensure that RTL constructs used in the design have no synthesis/simulation issue down the flow

Constraints: Ensure that no conflicting constraints are applied or any constraint is missing

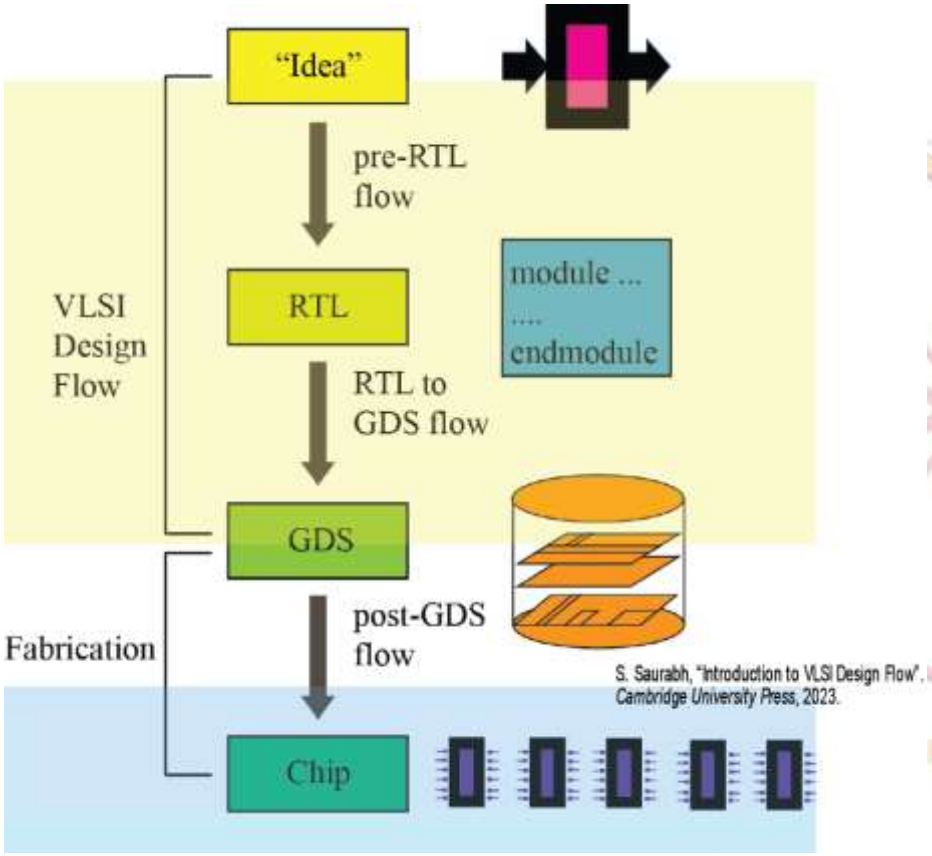
Netlist: Ensure that connectivity of instances do not cause any issue down the flow

Overview of VLSI Design Flow



Testing

VLSI Design Flow: Testing



- **Verification** : Ensures that GDS represents the circuit correctly such that original specification is met.

- **Testing** : Ensures that fabricated chip does not have any manufacturing defect

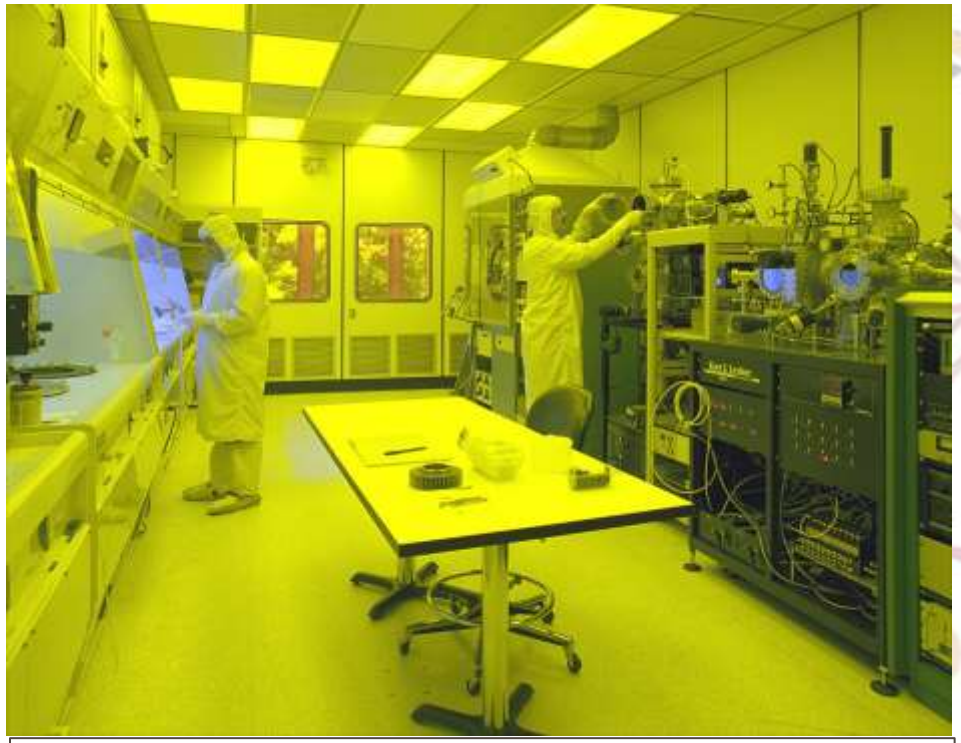
Why testing is important during designing?

- RTL to GDS flow ensures that the chip that is manufactured can be easily tested (**Testability**)
- RTL to GDS flow ensures that if some defect is found then the problem can be easily **diagnosed** (**debugged**) and fixed

Manufacturing Defects: Origin

Defects

- Physical imperfections in a fabricated chip that are of permanent nature



Source: https://commons.wikimedia.org/wiki/File:Clean_room.jpg
Uploaded by Duk 08:45, 16 Feb 2005 (UTC), Public domain, via Wikimedia Commons

Origin of defects

- Statistical deviations in material properties
- Finite tolerances in process parameters
- Airborne particles, and undesired chemicals
- Deviations in mask features

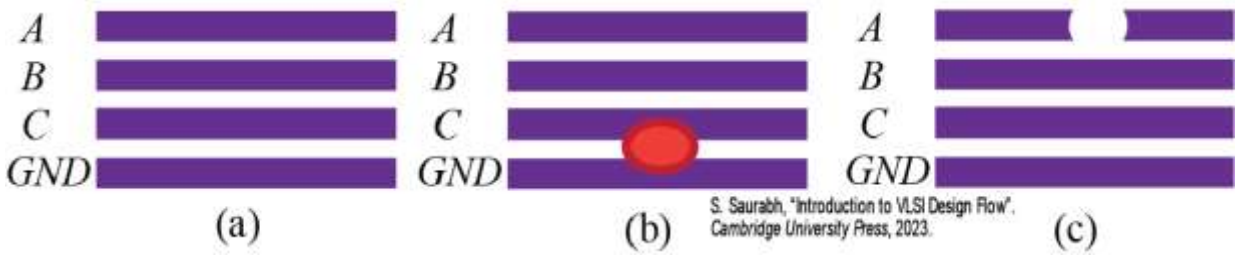
Large area and Spot defects

- Large area defects: simpler to eliminate
- Small area defects, of random nature
  - Inevitably appears in chip fabrication
  - Increases with increase in die area
  - Primary concern for testing

Manufacturing Defects: Manifestation

Manifestation of defects

- Short-circuit and open-circuit (functional failure)
- Change in circuit parameters (such as delay)



S. Saurabh, "Introduction to VLSI Design Flow", Cambridge University Press, 2023.

Faults:

- Testing focuses on finding defects that can be problematic for a circuit behavior
- We model defect (physical phenomenon) using **faults** (circuit model)

Distortions

- Photolithography can produce distorted features on a die
- These are inevitable due to optical effects, etc.
- Needs to be handled using suitable techniques
- **Testing does not detect these distortions**

Inconsequential flaws

- Deviations of a fabricated circuit from the ideal, but not causing any measurable change
- E.g.: if the size of particle is too small
- **Testing does not detect these inconsequential flaws**

Quality of Process: Yield

Yield

- Fraction of die on a wafer that are ***good*** or without any fatal manufacturing defect
- Usually expressed in percentage
- E.g.: If 300 dies are good out of 400 dies manufactured, then yield is:

$$= \frac{300}{400} \times 100\% = 75\%$$

Yield and process technology

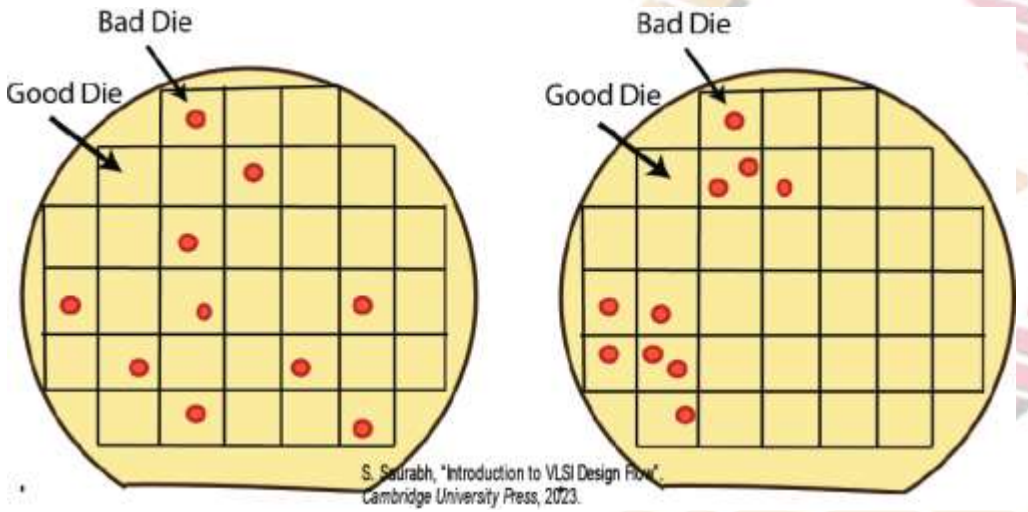
- Function of complexity and maturity of the process
- Desirable $> 50\%$, and often $> 90\%$

Factors affecting yield

1. **Die area**: when area increases, yield falls
2. **Defect density**: average number of defects per unit of chip area (depends on process)
3. **Clustering:** distribution of defects on the chip area

Yield: Dependency on clustering

Which is better for yield, when defects are clustered (defects lying in small region) or unclustered (same number of defects distributed over a larger region)?



S. Saurabh, "Introduction to VLSI Design Flow", Cambridge University Press, 2023.

Let us compute the yield in two cases shown.

Unclustered defects

$$\text{Yield}=\frac{24}{34}\times 100 = 71\%$$

Clustered defects

$$\text{Yield}=\frac{26}{34}\times 100 = 76\%$$

Yield Model

Yield model:

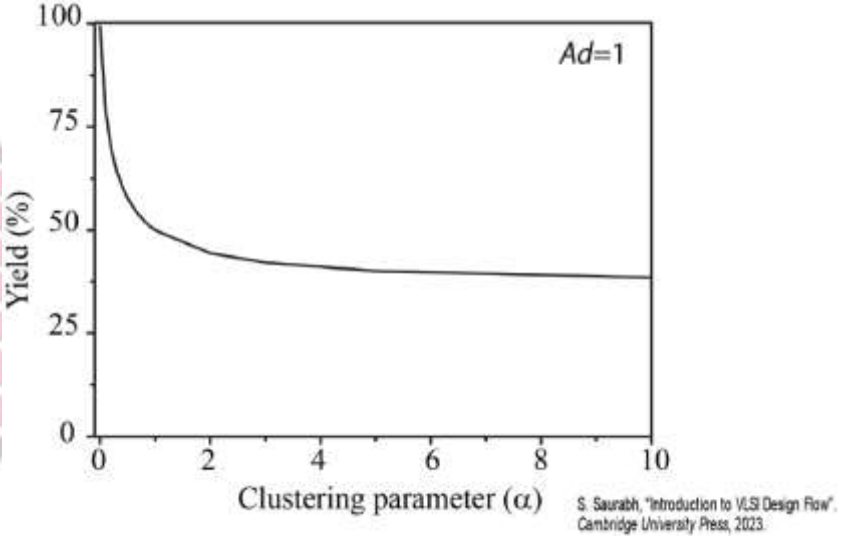
- Required to estimate yield (and hence profitability of IC manufacturing)
- Various models have been proposed with varying accuracy and complexity

- We can assume that the probability of having a defect in a given area increases linearly with the number of defects already present in that area
- One of the models

Yield $\boldsymbol{Y = (1 + Ad/\alpha)^{-\alpha} \times 100\%}$

where

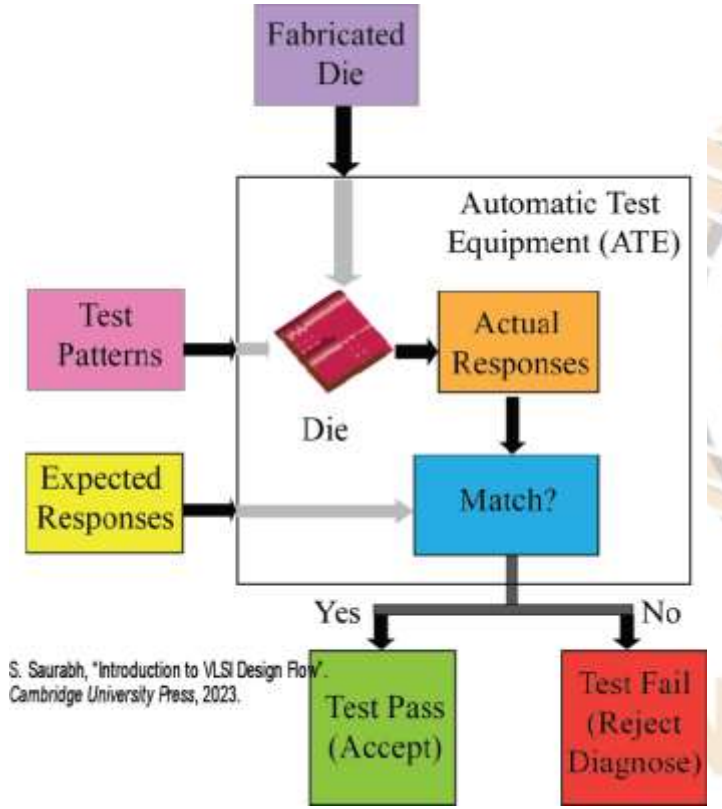
- A is die area,
- d is defect density,
- α is the clustering parameter



Clustering parameter

- $0 < \alpha < \infty$
- $\alpha \to \infty \Rightarrow weak\ clustering$

Testing Technique



S. Saurabh, "Introduction to VLSI Design Flow", *Cambridge University Press*, 2023.

Automatic Test Equipment (ATE)

- Consists of test head, probe cards with probe needles
  - Probe needles make contact with the test pads on the design under test (DUT)
- Test program controls all operation

- Test Patterns are applied to the manufactured chip
- Actual responses are compared with the expected response
- If comparison FAILs ⇒ The chip has some defect(s)

- The failed chip can be diagnosed to find the root cause of the problem
- After diagnosis corrective measures can be taken to reduce the defects

Fault Coverage and Defect Level

We measure the quality of testing using a parameter called **fault coverage**

- **Fault coverage**: measures the ability of the set of test patterns to detect a class of faults
- $Fault\ coverage\ = \frac{\#\ faults\ detectable}{\#\ faults\ possible}$
- Fault coverage is a measure of ***quality of testing***

- Practically achieving 100% fault coverage is challenging
- We try to attain more than 99% coverage
- Some faulty products can reach the end user because the employed set of test patterns may not cover/test all possible faults

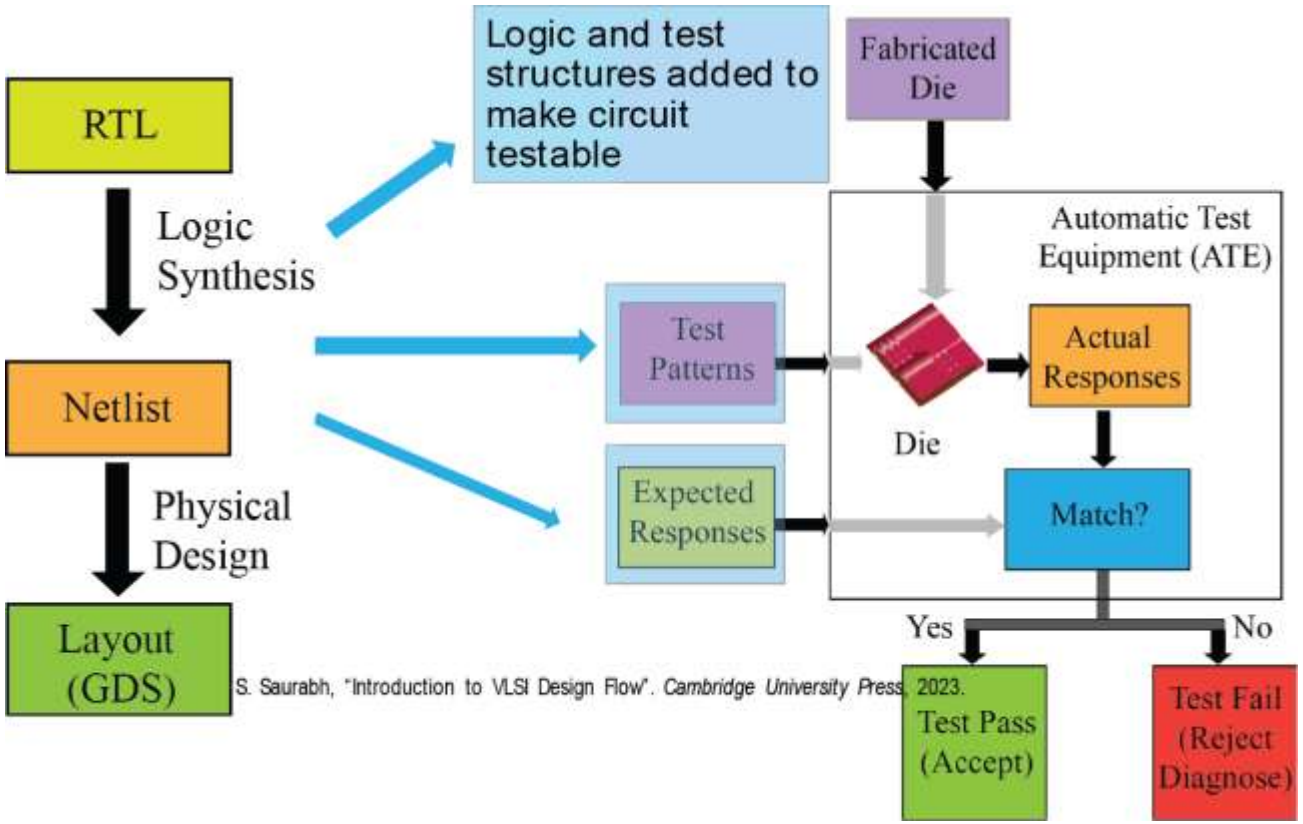
- Perceived quality of a chip strongly depends on: **fault coverage** and **yield**
- Quality of chip is measured by a quantity called ***defect level***

Defect Level (DL)

- Ratio of chips that are "bad" or faulty among the chips that have PASSED the tests
- Measured in parts per million (ppm)

- DL is a measure of effectiveness of the tests (if test is fully effective, then $DL = 0$)
- For commercial chips $DL < 500\ ppm$

Design For Test (DFT)



References

- S. Saurabh, “Introduction to VLSI Design Flow”. Cambridge: *Cambridge University Press*, 2023.