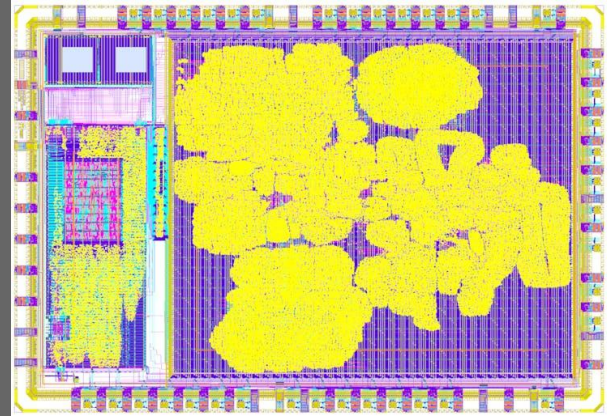


ASIC Design Intro

And how it differs from FPGA design



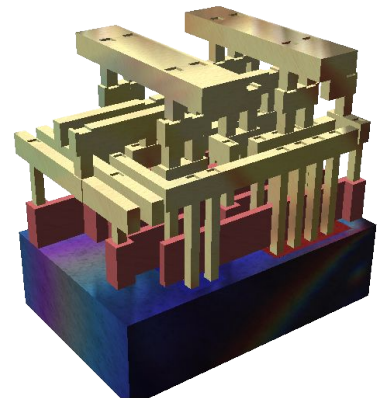
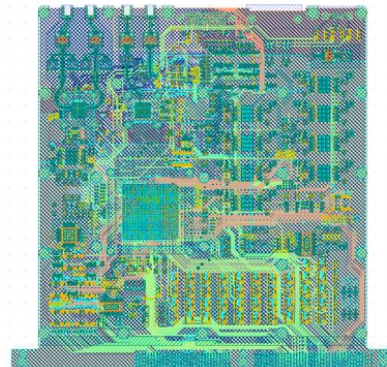
EDA Design Flows

- Goal: Turn our design into a chip
 - RTL to **GDSII**
 - Requires 50+ steps
 - Too complex to do manually
- Failures can occur at any step of the process
 - Our job is to find and fix failures until we hit **Design Closure**

Glossary

GDSII: “Graphic Design System” 2, is a database that contains the final manufacturable design of a chip

Design Closure: meeting all performance, timing, power, functionality, and validation requirements



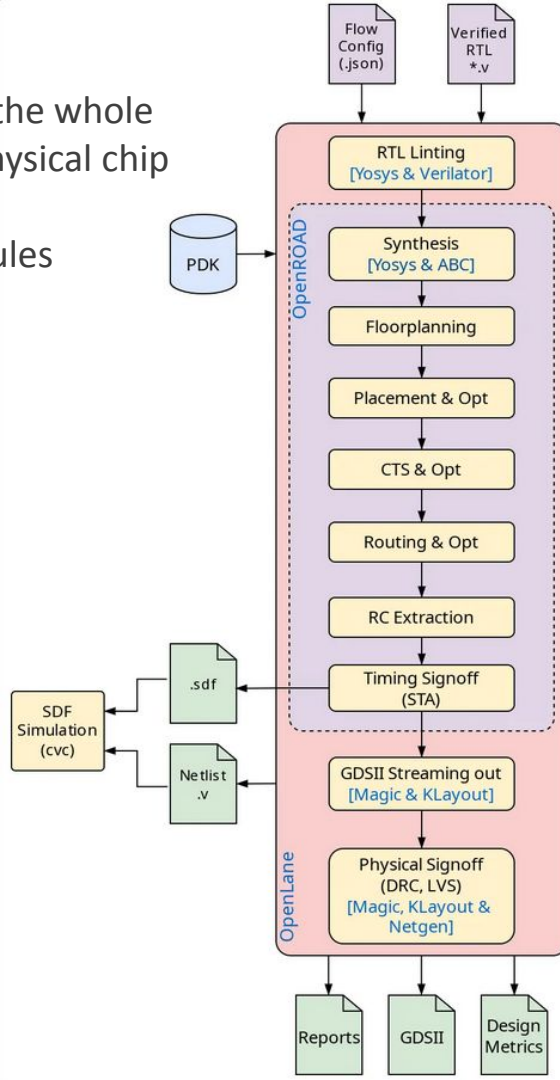
OpenLane2

Glossary

VLSI: very large scale integration, the whole process of going from design to physical chip

PDK: process design kit, manufacturer-specific gates and rules

- AKA LibreLane
 - Our main **VLSI** tool flow
 - Combines many different open source tools
- Optimized for SKY130 **PDK**
 - ASIC designs are manufacturer specific
 - Skywater is US foundry that has open sourced their process



Linting

- Does your code have syntax errors in it
 - Missed semicolon
 - Widths don't match
- Differences from FPGA:
 - Vivado → Not strict, can get away with technically illegal code
 - Verilator → Strict
 - all warnings are errors unless waived
 - idea is find issues early on



Simulation

- For both FPGA and ASIC, start by testing code in simulation
 - FPGA → Only a qualifier/check before hardware
 - ASIC → Primary Method of Development
 - IN DEPTH VERIFICATION
 - Often use multiple simulators

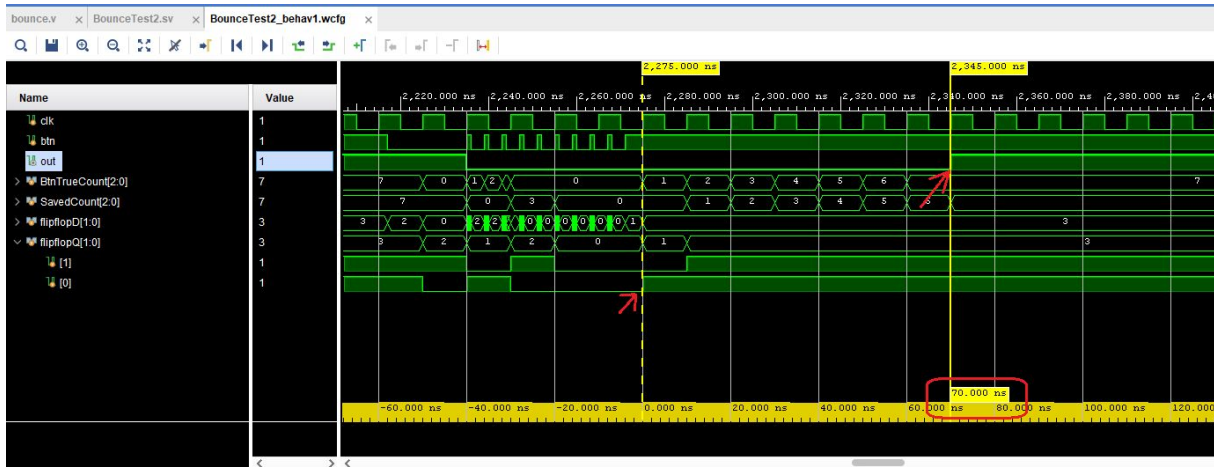


Slow and Accurate



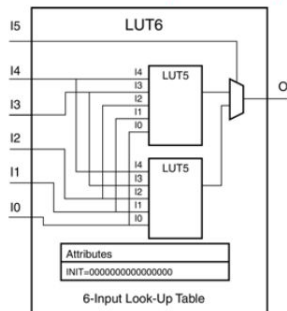
VERILATOR

Fast but Less
Accurate



Synthesis

- Converts behavioral logic into lowest-level logical operations
 - Nand, Nor, Xor, etc.
- FPGA → LUTs
 - Homogenous
 - DSP Slices for $\ast$ / $\div$
- ASIC → Standard Cells
 - **Technology mapping** converts logic into simple or complex cells



X1094

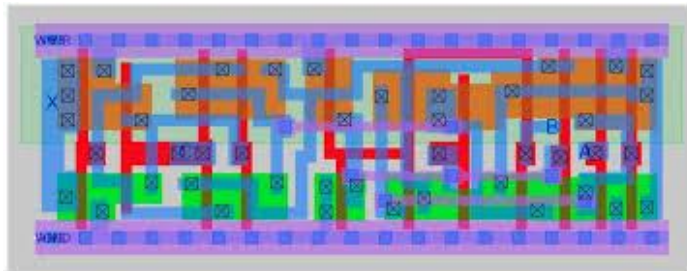
Glossary

Standard Cell: manufacturer-specific implementation of a logic gate

Technology Mapping: converting logical operations into process specific standard cells



Standard Cells



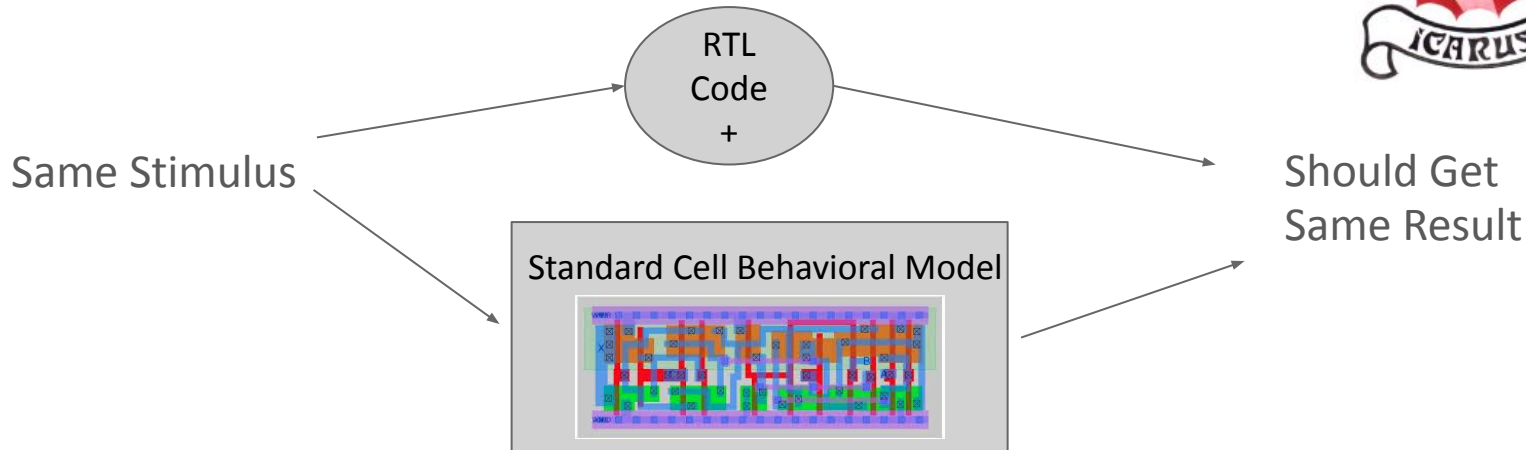
Gate Level Simulation

- At this point in FPGA land, we would just test on FPGA...
- But for ASIC:
 - Re-Run simulation and testing on gate-level **netlist**
 - Ensure synthesized version matches RTL

Glossary

Netlist: verilog file of connected standard cells

Stimulus: set of inputs over time to device



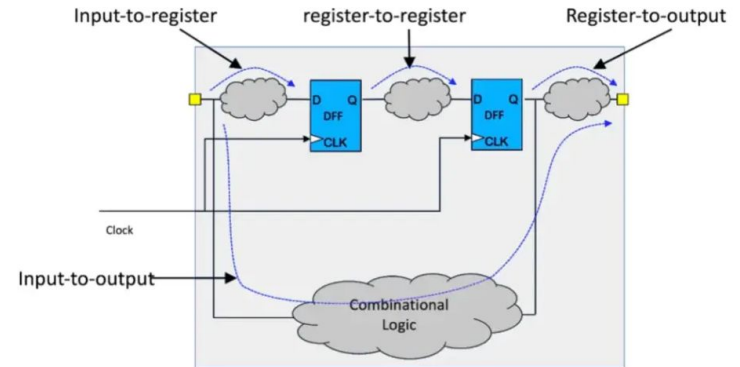
Static Timing Analysis: OpenSTA

Step 12

Glossary

STA: static timing analysis

- Use **STA** to make sure that design meets desired timing constraints
- Figure out how fast logic is based on longest path
 - Does this meet target frequency?



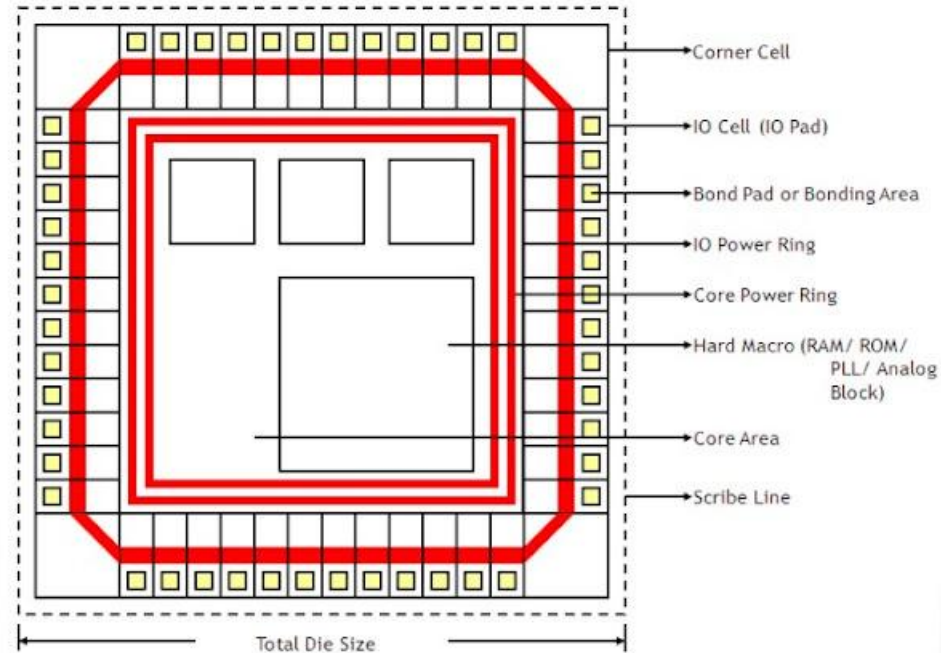
Floorplanning

- Floorplanning decides generally how big chip is and where things go
 - **Macros** placed manually
 - Standard cell locations estimated
- **Core Utilization** (FP_CORE_UTIL)
 - Averages 50-60%
 - **Higher** utilization → greater logic density, more gates
 - Might fail in routing
 - **Lower** utilization → less gates, but easier to route
- Difference from FPGA: not a set amount of LUTs, must decide on logic density
 - Can use Macros → entire giant blocks of a chip are reused

Glossary

Macros: pre-laid-out circuits to use as a component in a design. Often memory, IP, or analog circuitry

Core Utilization: percentage of area used for logic versus kept empty for routing

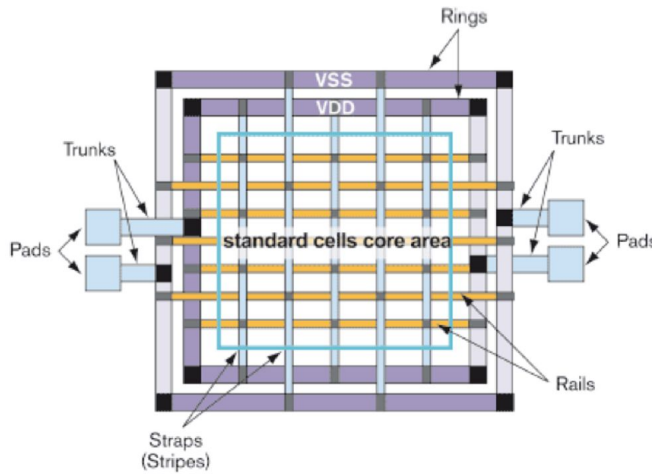


Power Connections

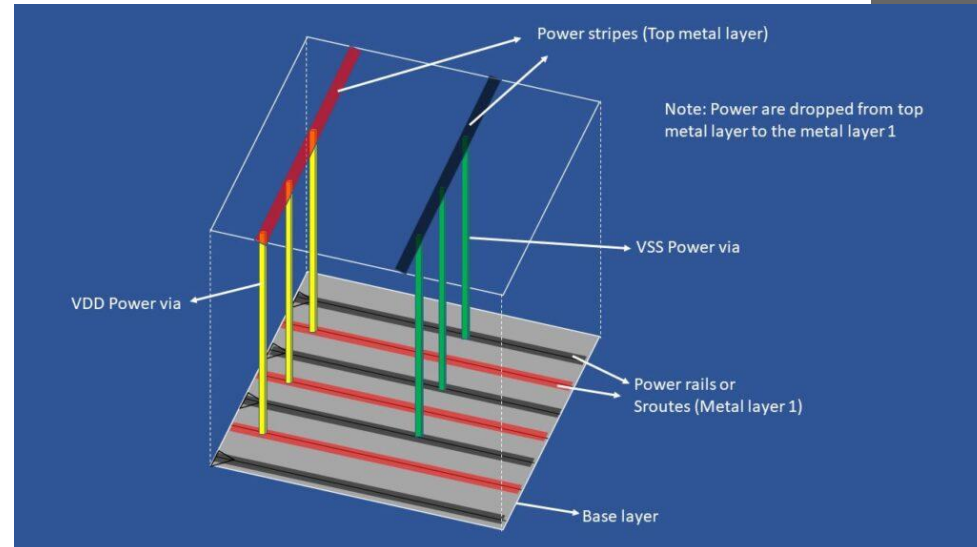
Glossary

PDN: Power Distribution Network

- Every standard cell needs power
 - **PDN** used to globally route power
- **FPGAs** → Care about power for \$\$\$ reasons
- **ASIC** → Also care about power so your **chip doesn't melt**
 - Put more effort into turning off parts of design

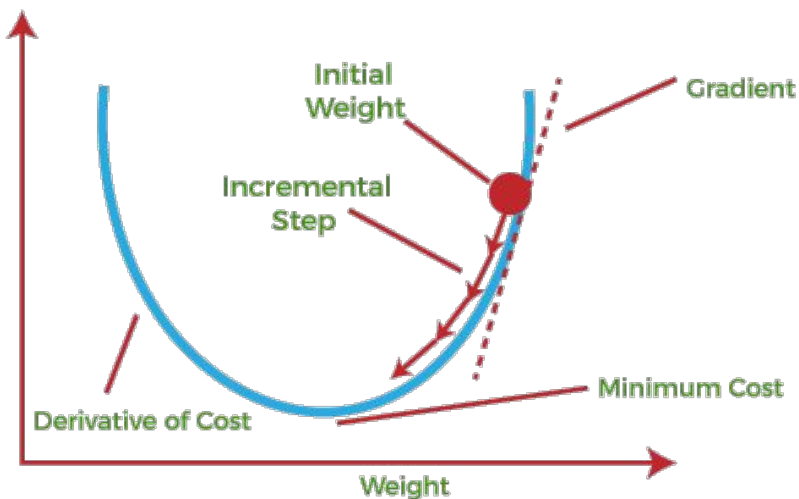
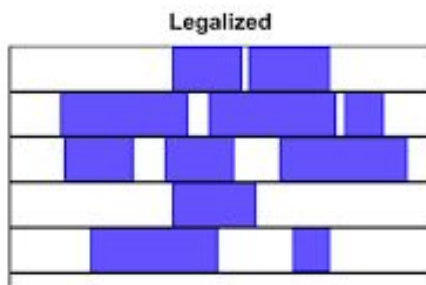
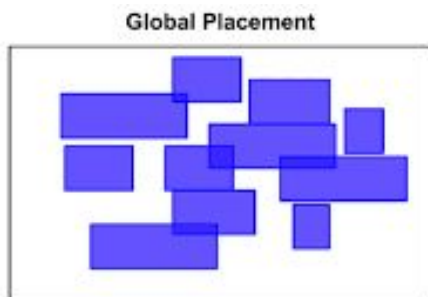


This figure show the complete power planning.



Placement

- Tools must place every standard cell, intelligently
 - Different from FPGA → preplaced LUTs get connected
 - More iterative control over where logic goes
- **Legalization:** move standard cells from general positions into strict rows



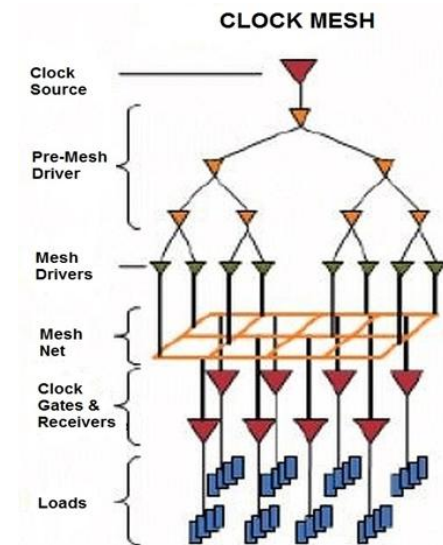
Clock Tree Synthesis

- Important that clock signals globally arrive at the same time
 - Difference in arrival times is clock skew
- **Clock Skew** makes all timing violations worse
- Use specialized structures to distribute clock signal
 - H Trees -> fractals
 - Same propagation delay to every endpoint

Glossary

CTS: Clock Tree Synthesis

Clock Skew: difference in clock signal arrival time between points

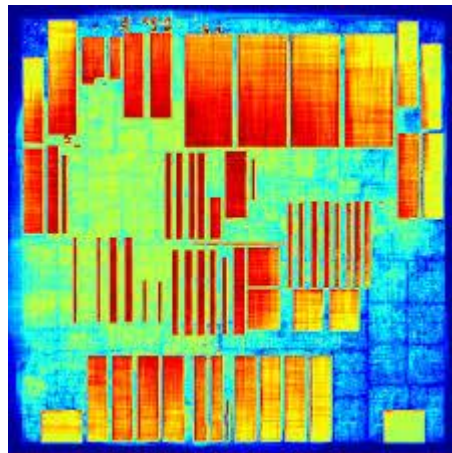


Routing

Glossary

Congestion: ratio of how much available routing space is being used

- Modern chips are wire limited, not logic limited
 - Can fit all the gates we want
 - Wiring them up becomes much harder
- Routing: uses metal layers and vias to connect gates as needed
- Run into **congestion** issues in dense areas
 - 100% congestion → no more wires in an area
 - Turn down core utilization
- Why no RISC-V 256?
 - More parallel wires get exponentially more difficult.



Congestion Heatmap: Visualizes areas with many interconnects, where wire space is all used up

Gate Level Sim and STA AGAIN

Glossary

PnR: place and route,
entire process of
placement and routing

SDF: standard delay format

- After **PnR**, get a new netlist
- Re-Run simulation and testing on placed netlist
 - Will include buffers, power wires, etc
- Re Run timing analysis
 - More accurate now
- Put them together!
 - Timing annotated (**SDF**) simulation!
 - Simulate the gates and their delays



Example

```
(INSTANCE x)
(Delay
  (ABSOLUTE
    (COND b (IOPATH a y (0.21) (0.54) ) )
    (COND -b (IOPATH a y (0.27) (0.34) ) )
    (COND a (IOPATH b y (0.42) (0.44) ) )
    (COND -a (IOPATH b y (0.37) (0.45) ) )
  )
)
```

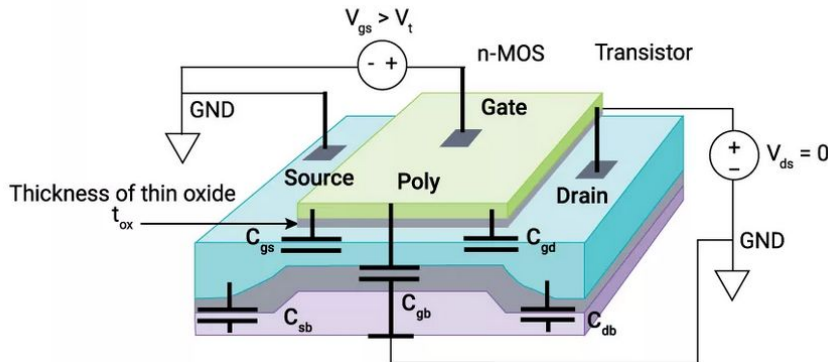
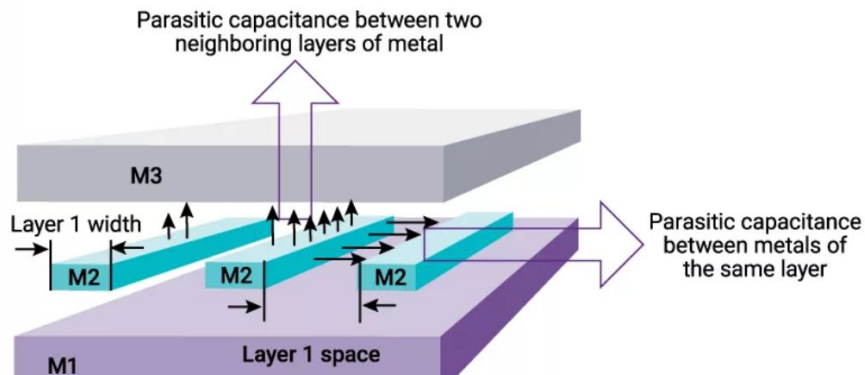
RC Extraction for STA

- Extract Resistance and Capacitance values of each cell and wire using **PEX**
 - Analyze wire and gate area, length
 - Calculate **parasitics**
- Re-run **STA** using higher-accuracy model
 - Now takes account for all gate sizes and wire lengths

Glossary

PEX: Parasitic Extraction

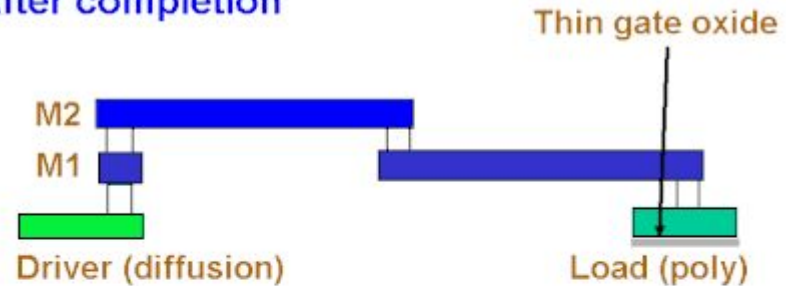
Parasitics: unintended resistance, capacitance, etc. inherent to physical design



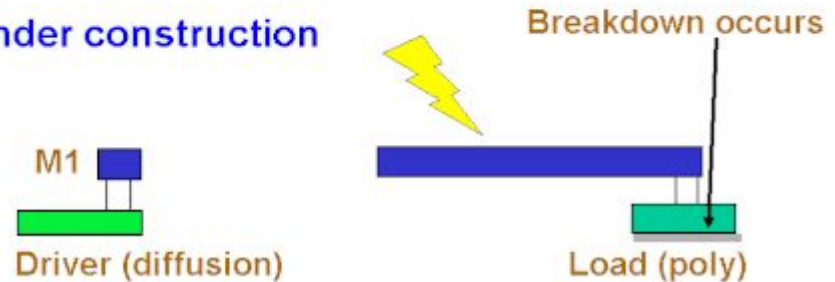
Antennae Checking

- Can your design be manufactured without breaking?
 - Not Noise Related!
- “plasma induced gate oxide damage”
 - Transistor gates are sensitive to high voltage
 - Will spark and destroy gate
- Means long wires can physically break your design

(a) After completion

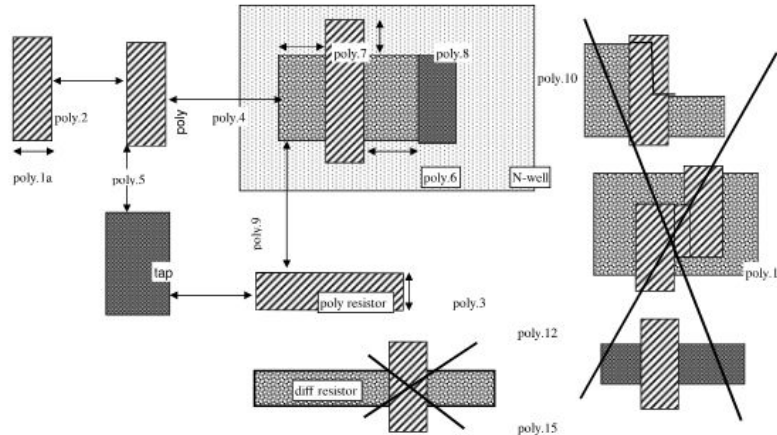


(b) Under construction



DRC

- Ensures manufacturability
- Enforces rules set by the PDK about what is allowed for each layer
- Most computationally intensive
 - Needs to calculate geometry of thousands to millions of shapes across hundreds of layers
 - Can take hours

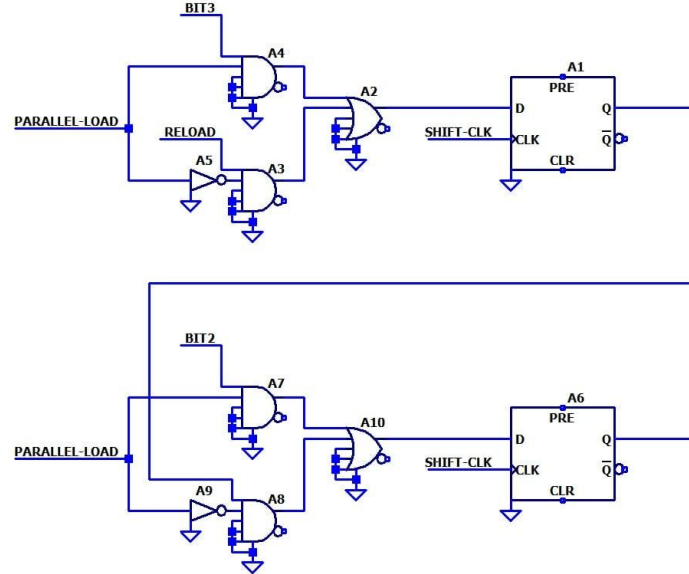


SPICE Extraction

Glossary

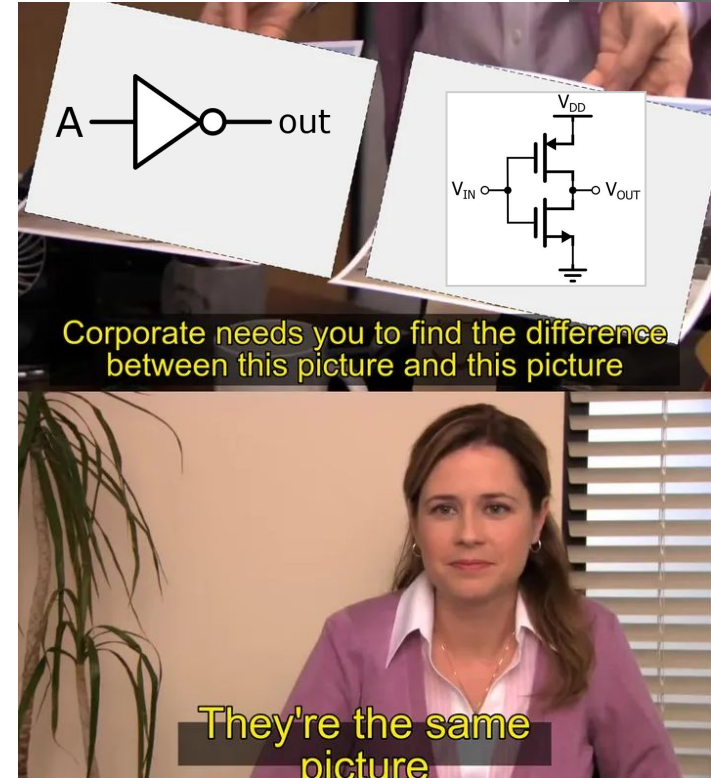
SPICE: Simulation Program with Integrated Circuit Emphasis, electrical simulation language used in LTSpice, NGSpice, etc.

- Use geometry of wires and gates to assemble an electrical netlist (a **SPICE** model)
 - Metal layers connected by vias become single nets
- Assemble transistor-level model of system
 - Can do electrical simulation of small scale systems
 - VERY SLOW



Layout Versus Schematic

- **LVS** compares the electrical **SPICE** netlist of the final circuit with the expected synthesized logical netlist
- Should match if the tools did their job
 - Doesn't match? LVS Error
- Common Issues with LVS:
 - Power / Ground rails not properly connected
 - Macro Pins not connected



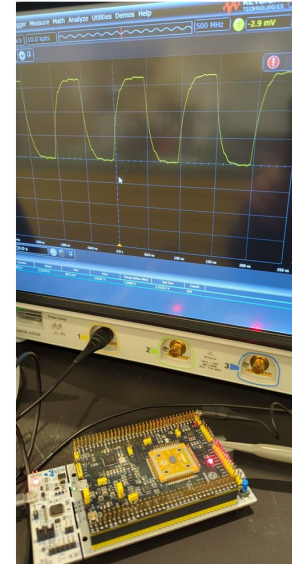
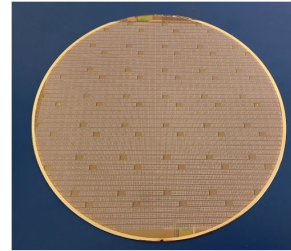
Tapeout

Glossary

Signoff: completing all final verification checks before tapeout

Tapeout: sending final design (GDSII) to manufacturer

- Your design is clean, time for **signoff** and **tapeout**!
- Now you have a chip! ...
 - In 3-6 months
 - You will probably have hundreds of chips
- Do your chips work?
 - Some of them!
 - Bring Up often takes a huge % of budget!



References

- <https://digitalsystemdesign.in/placement-and-routing-for-asic/>
- <https://physicaldesign-asic.blogspot.com/2020/06/floorplanning.html>
- <https://www.semiconductor-digest.com/clock-tree-optimization-methodologies-for-power-and-latency-reduction/>
- <https://www.synopsys.com/glossary/what-is-parasitic-extraction.html>
- <https://openlane2.readthedocs.io>
- <https://openlane.readthedocs.io/en/latest/>
- <https://openroad.readthedocs.io/en/latest/>