Digital Integrated Circuits

(83-313)

Lecture 1:

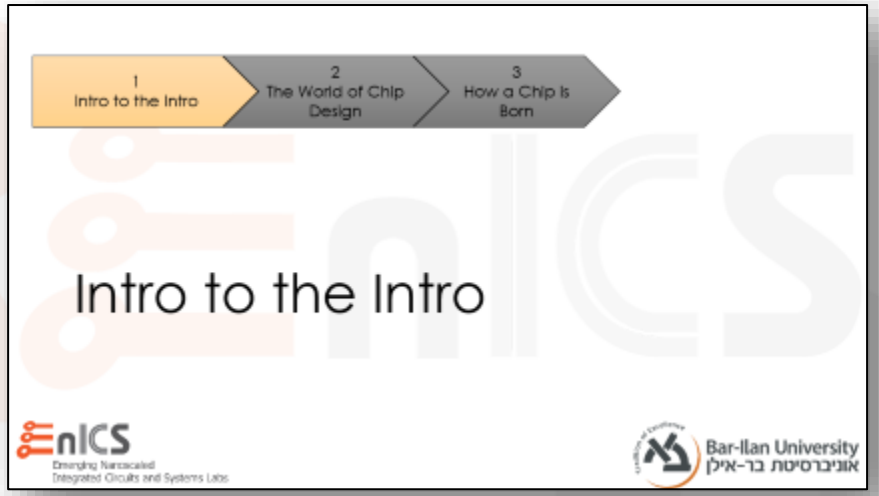
Introduction

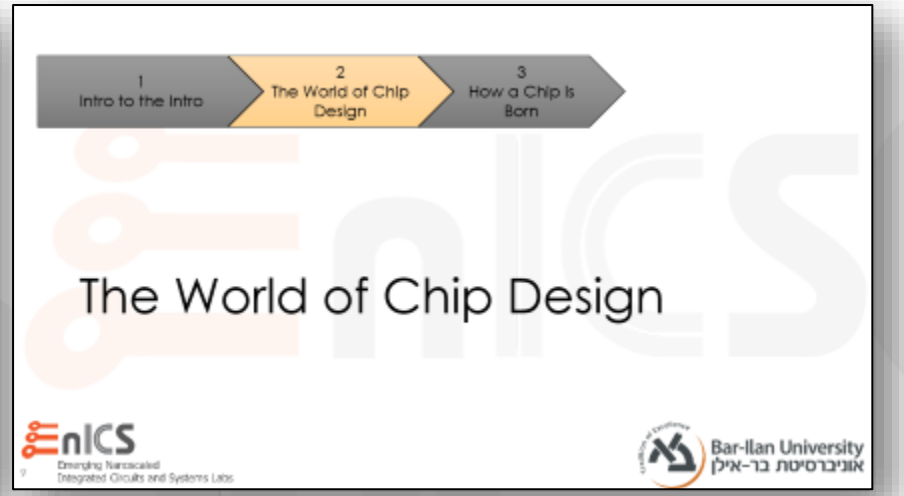
18 March 2020

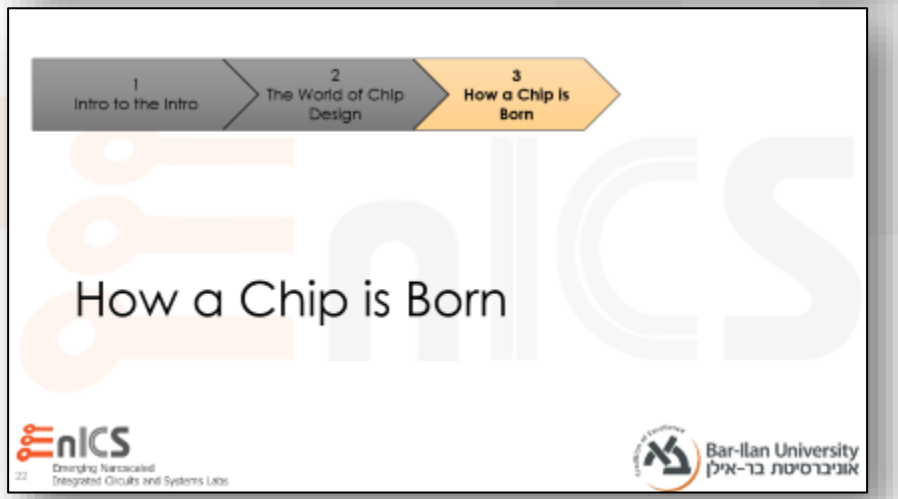
Emerging Nanoscaled
Integrated Circuits and Systems Labs

Bar-Ilan University
אוניברסיטת בר-אילן

Lecture Overview







Intro to the Intro

What will we do in this Course?

- **In *Digital Electronic Circuits* (83-308), we learned the basics of building a digital gate.**
- **In this course we will step up a level, including:**
  - Learning about designing actual digital gates for use in a full digital abstracted implementation flow.
  - Deepening our understanding of the trade-offs in VLSI design.
  - Learning how to design bigger and more complex components, such as arithmetic circuits and memories.
  - Learning about parasitics that are inherent to integrated circuit design.
  - Learning about technology scaling, how this affects us at the process, modeling and design levels, and study basic methods to deal with the resulting effects.
  - Learning about designing in the nanoscale era (i.e., what's going on today).
  - Starting to learn how to "put-it-all-together" towards compiling a VLSI chip.

There's No Free Lunch!

- **The basic motto of this course (and engineering in general) is to understand, design and optimize the *trade-offs* between:**

Performance

Cost

Reliability

Power

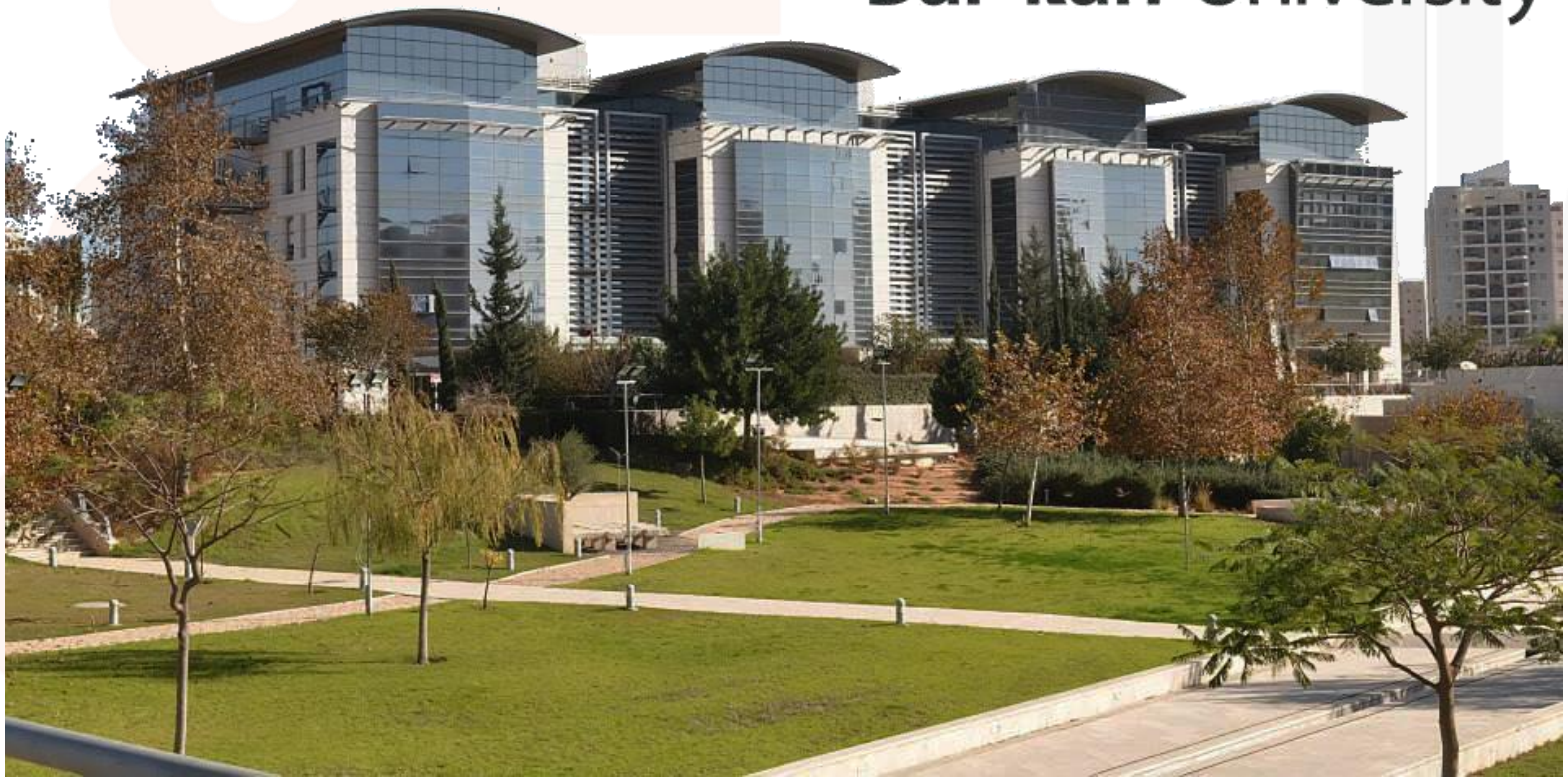
Course Syllabus

- **Part 1: How transistors actually come to this world**
  - Short Introduction
  - Manufacturing Process
  - MOSFET Models
  - Process Variations
- **Part 2: Honey, I Shrunk the Transistors**
  - Abstraction in VLSI Design
  - Design Metrics
  - Process Scaling
  - The NanoScaled Transistor
  - Interconnect
- **Part 3: Let's go out to play**
  - Sequential Logic
  - Arithmetic Circuits
  - Memory Design
  - FinFETs and additional subjects

Who are we?

Tradition of Excellence

Bar-Ilan University



Faculty of Engineering, Bar-Ilan University (BIU)

EnICS

Emerging Nanoscaled
Integrated Circuits and Systems Labs

Who are we?

- **EnICS – Emerging Nanoscaled Integrated Circuits and Systems Labs**

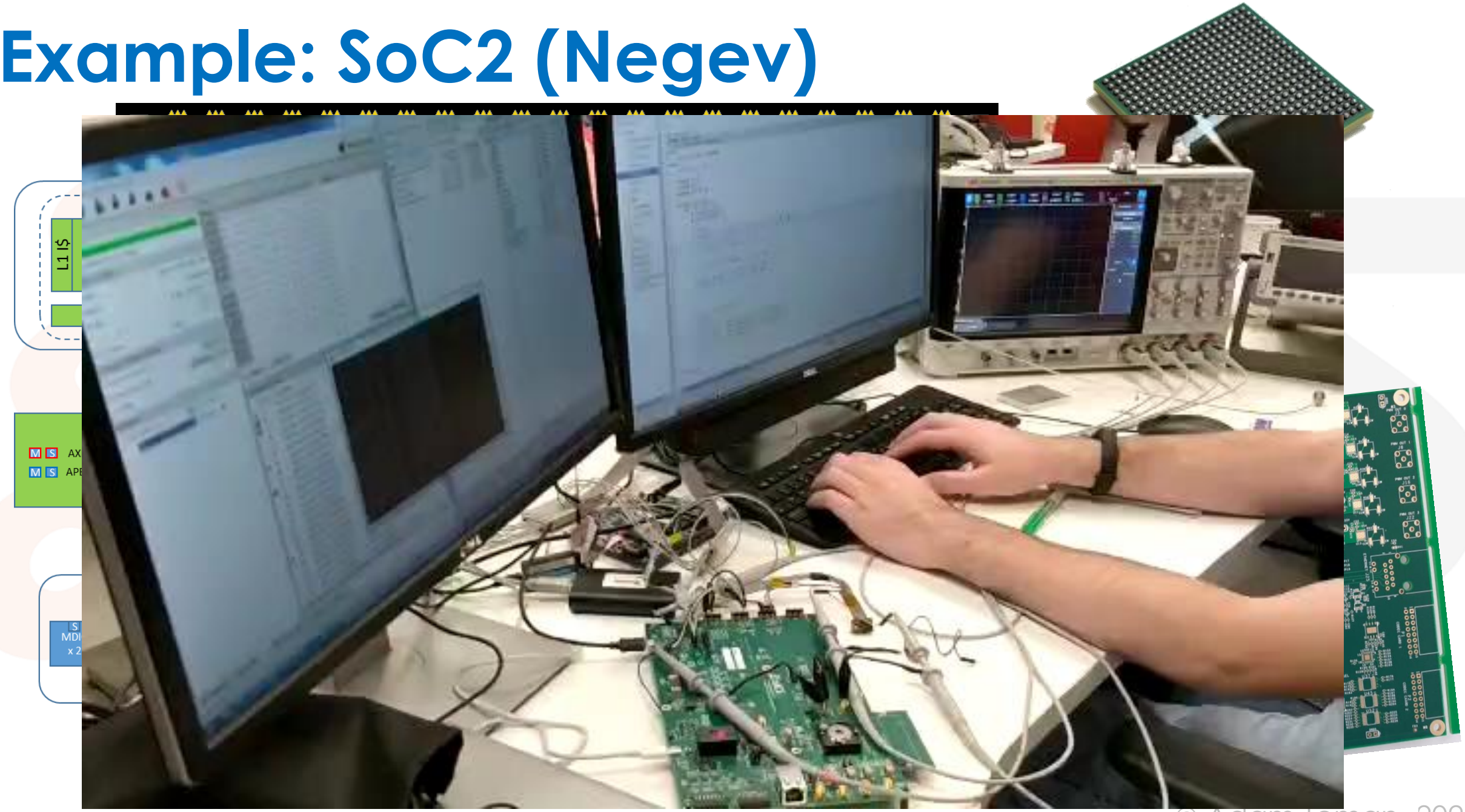
The World of Chip Design

The World of Chip Design

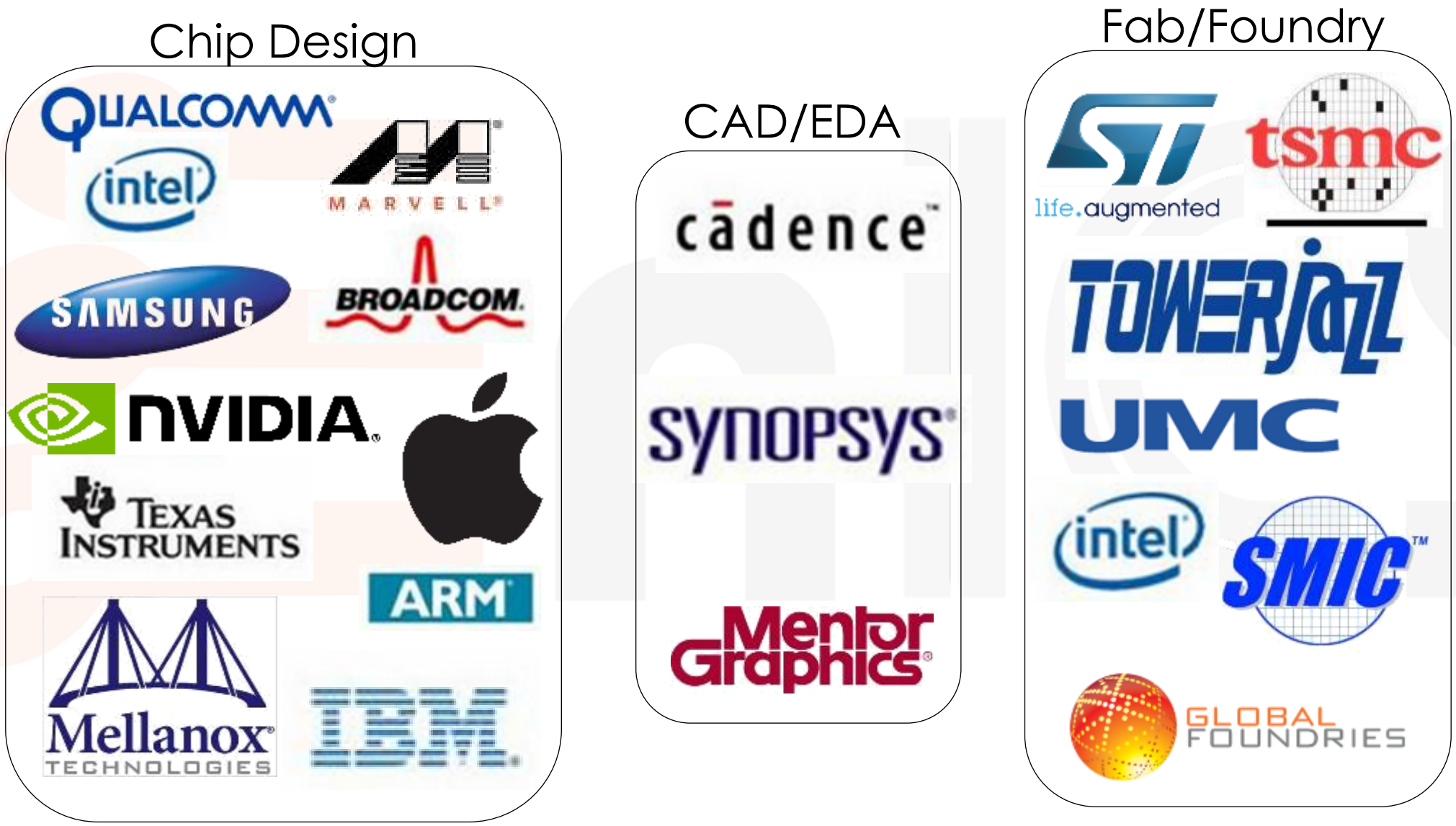
Primer: What is inside a computer?

Primer: What is an IC Chip?

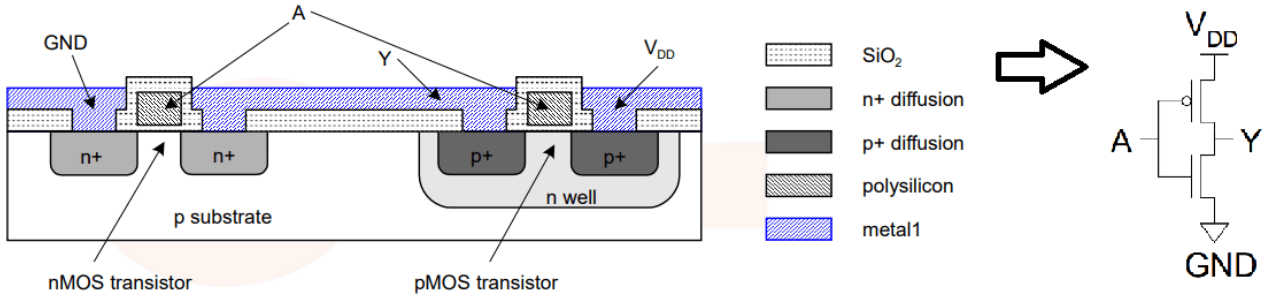
Example: SoC2 (Negev)

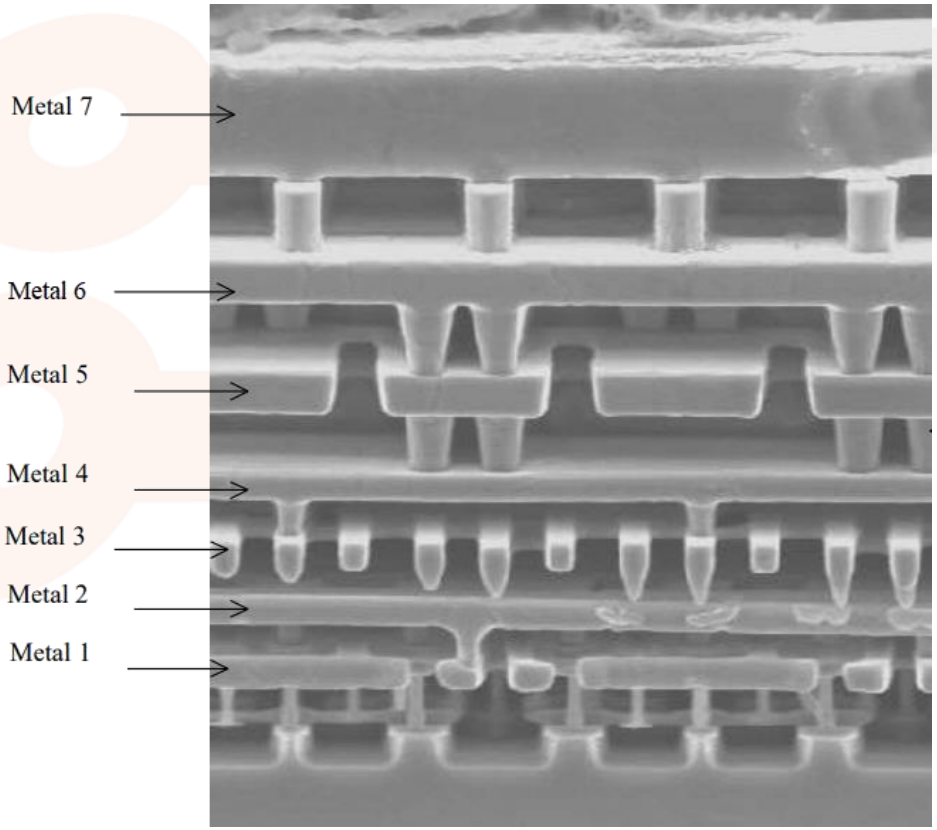


The VLSI Design Ecosystem



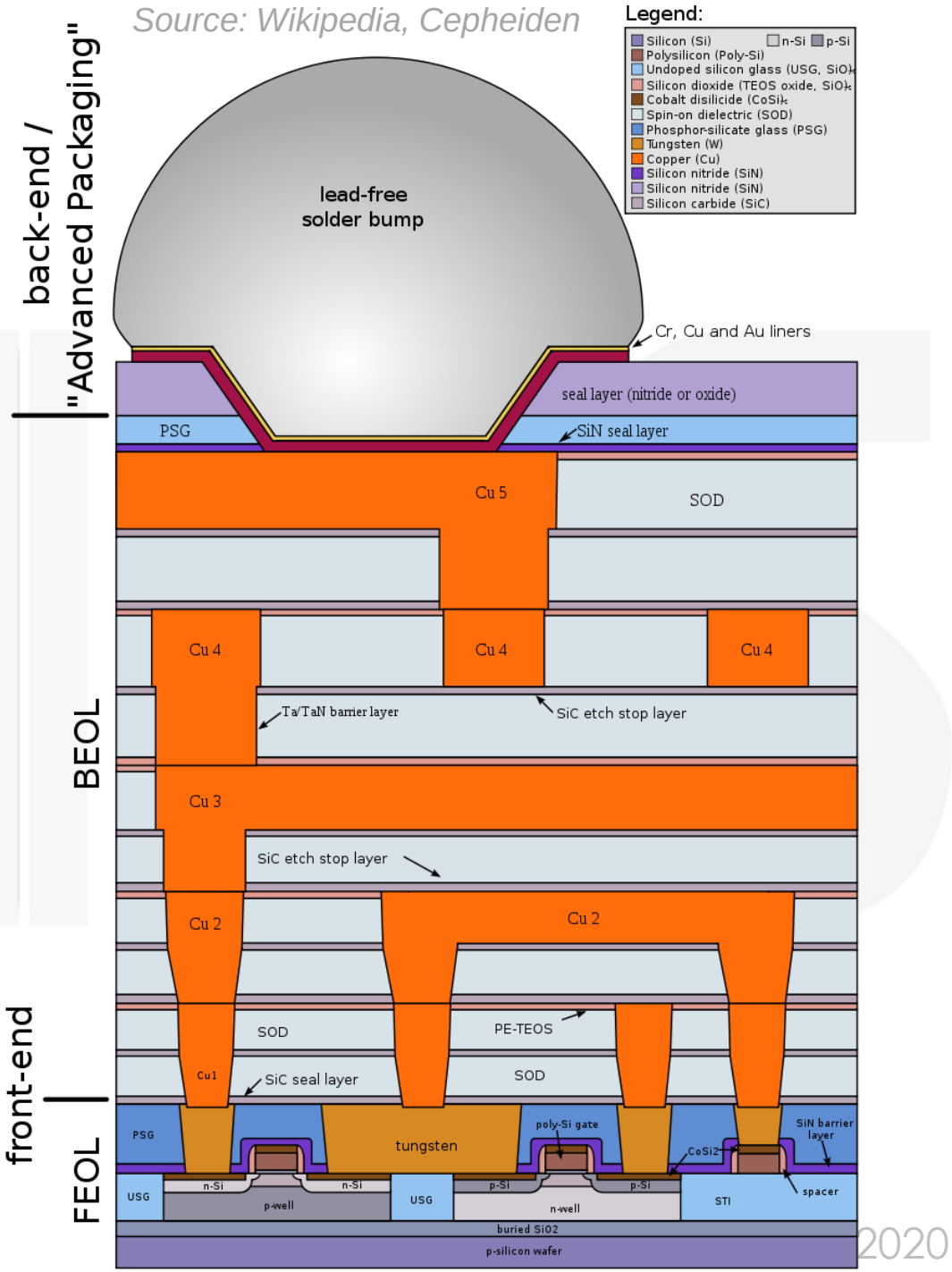
CMOS Technology





Source: techspot.com

Source: Wikipedia, Cepheiden

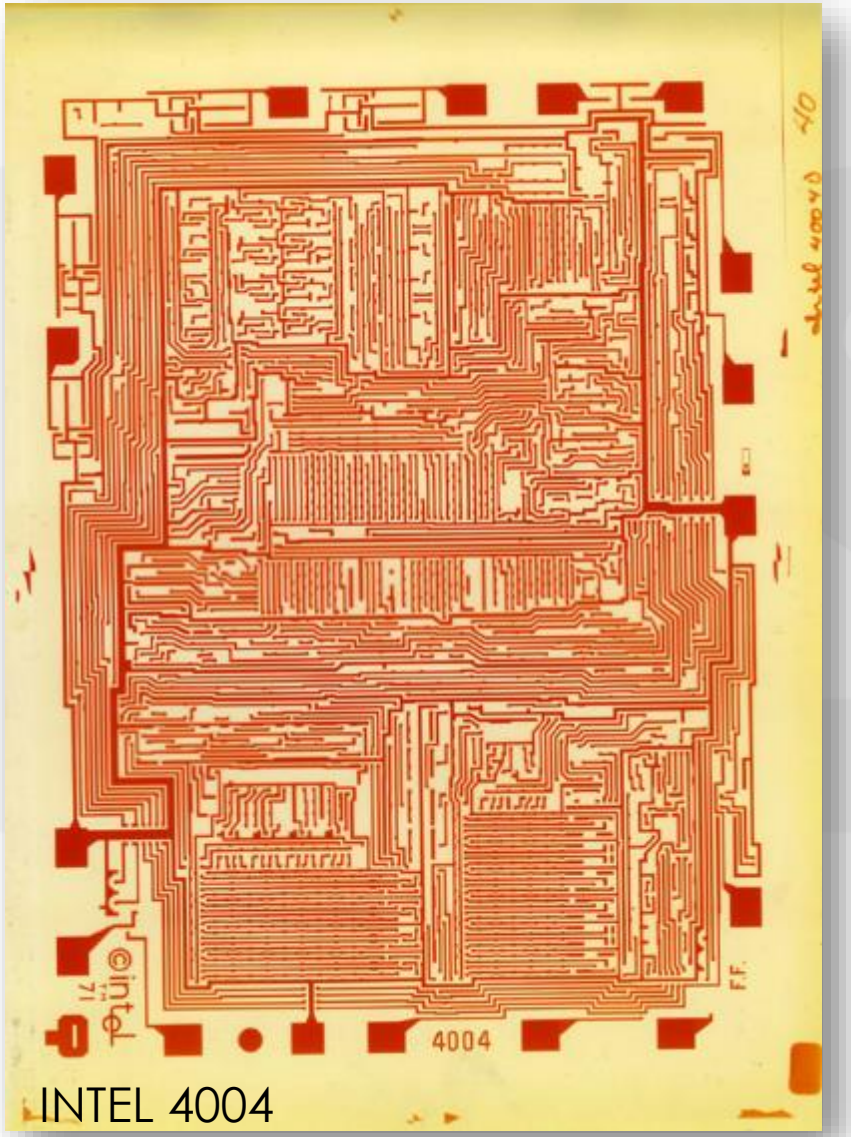


VLSI Design Styles

- **Due to time and cost constraints, very few teams and/or companies develop products from device level through to system level.**
- **Various *Design Styles* are available to shorten the *time-to-market* and *development cost*.**
- ***Trade-offs* are taken into consideration, as abstractions are usually designed *generically* and therefore come with some overhead.**
- **In the following slides, we will briefly discuss:**
  - Full custom design
  - Standard Cell based ASIC design
  - Field-Programmable Gate Array (FPGA) design
  - Microprocessor (Software)

VLSI Design Styles – Full Custom

- **Full Custom Design**
  - The original design style.
  - Everything is done at transistor level
  - Rarely used in digital design
  - High cost
  - But, gives significant gain in performance.
  - Analog designs mostly

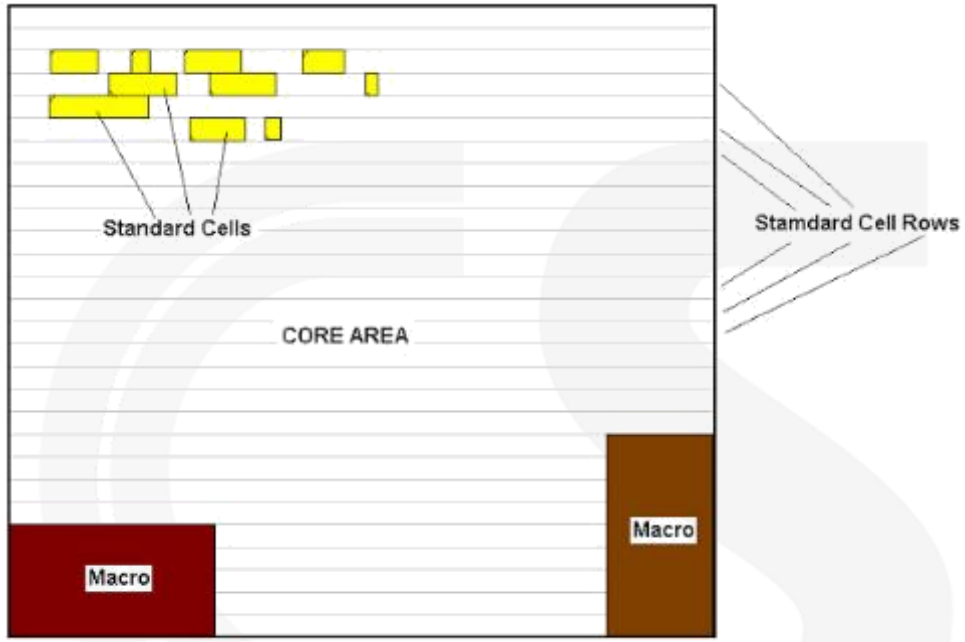


INTEL 4004

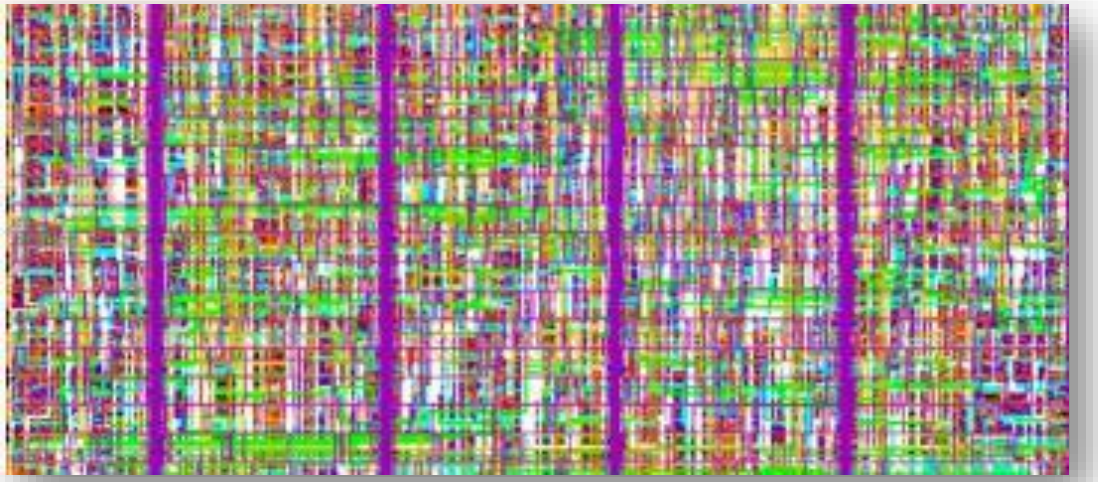
source: www.arstechnica.com

VLSI Design Styles – Standard Cells

- **ASICs (Application Specific Integrated Circuits) are usually built with Standard Cells.**
  - A library of standard cells enables digital implementation of Boolean functions.
  - HDLs are mapped to Libraries
  - Standard cell layouts meet placement guidelines for easy physical implementation.



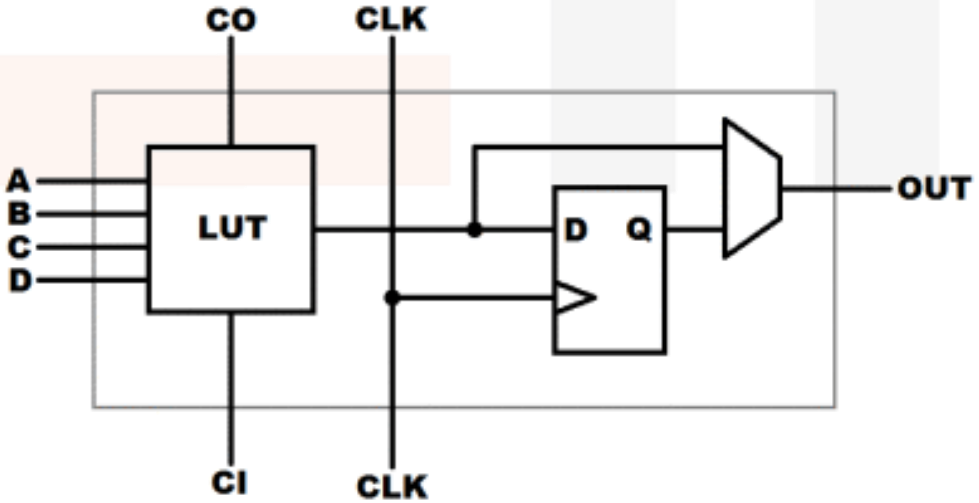
Source: Mantra VLSI

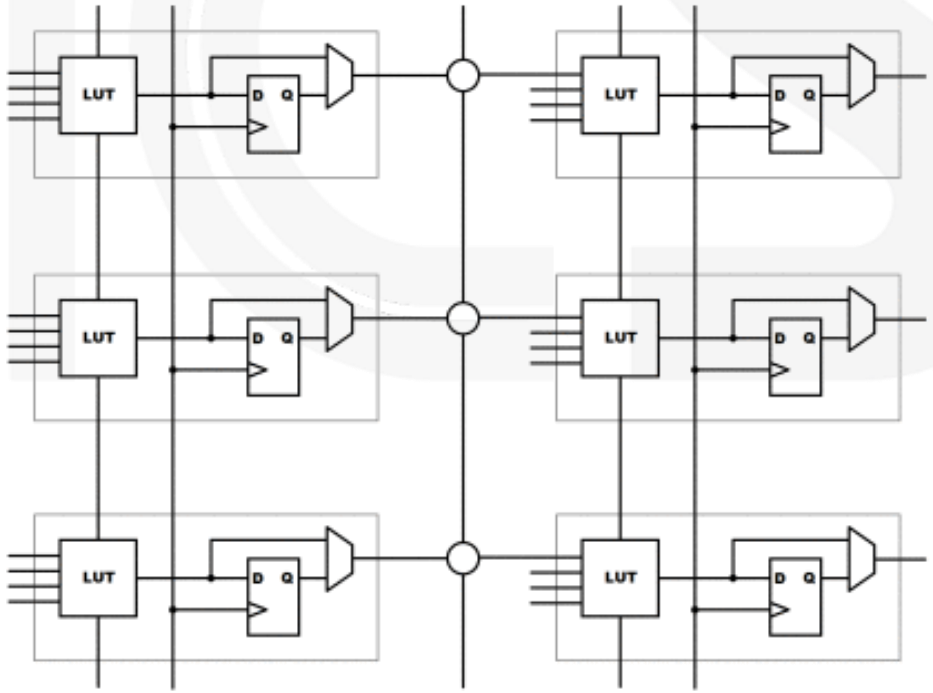


VLSI Design Styles - FPGA

- **FPGA – Field Programmable Gate Array**
  - Array of configurable logic blocks, and programmable interconnect structures
  - Fast prototyping, cost effective for low volume production
  - HDL (hardware description language) – is used

Source: wikipedia





Source: breadboardgremlins.wordpress.com/

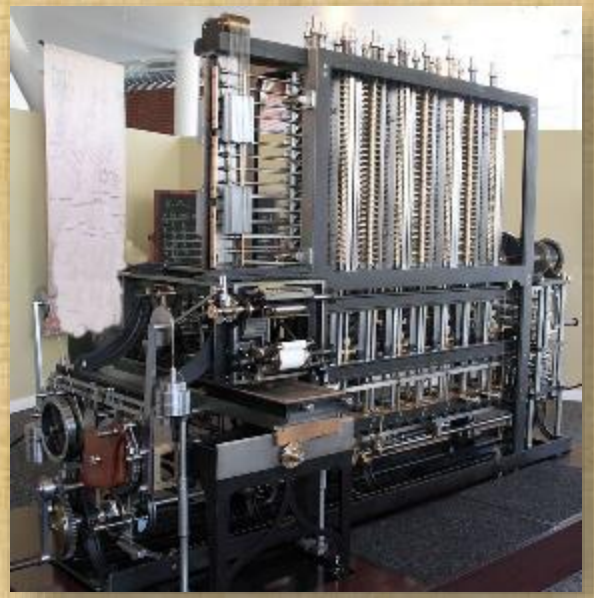
Design Styles – Pros and Cons

- **Full Custom Design:**
  - Customization for optimized power, performance, area.
  - High complexity = cost, time-to-market, high risk.
- **Standard Cell:**
  - Simple, fast, reliable.
  - Only Digital designs. Excess power, wirelength, etc.
- **FPGA:**
  - Post silicon configurability, very inexpensive.
  - High percentage of overhead. High cost per chip.
- **Microcontroller (Software)**
  - Programmable, very inexpensive
  - Very slow compared to hardware implementation.

The Computer Hall of Fame

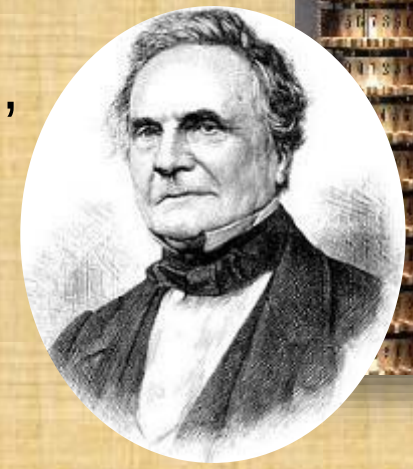
- **The parents of all computers:**

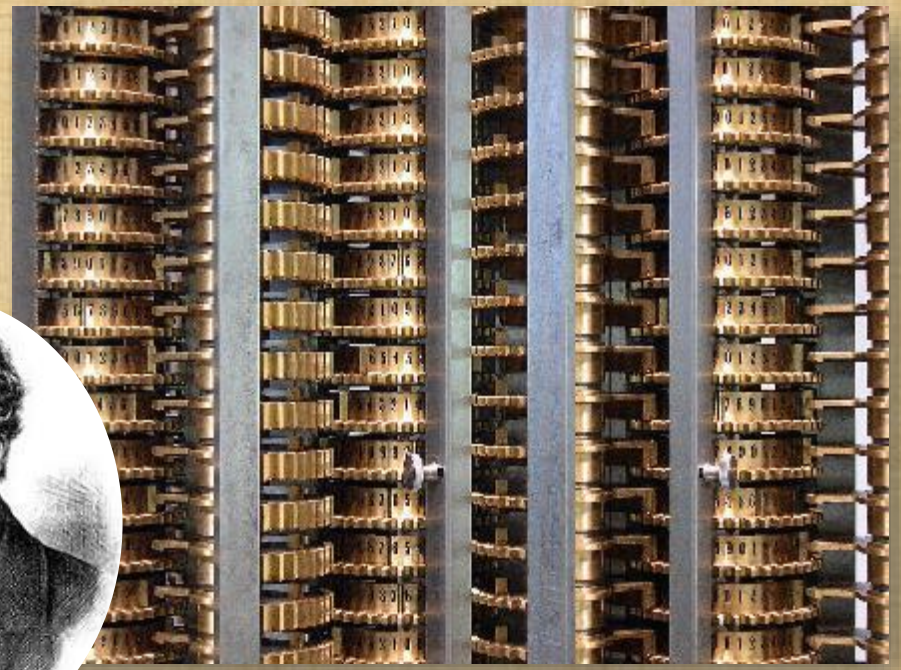
The Difference Engine
The Analytical Engine



Source: wikipedia

- Developed by Charles Babbage in 1822 & 1837
- The Difference Engine never built due to a lack of funding (until 1991).
- The Analytical Engine featured an ALU, basic flow control, punch cards and integrated memory.
- Only a portion was built in 1910, after Babbage's death.

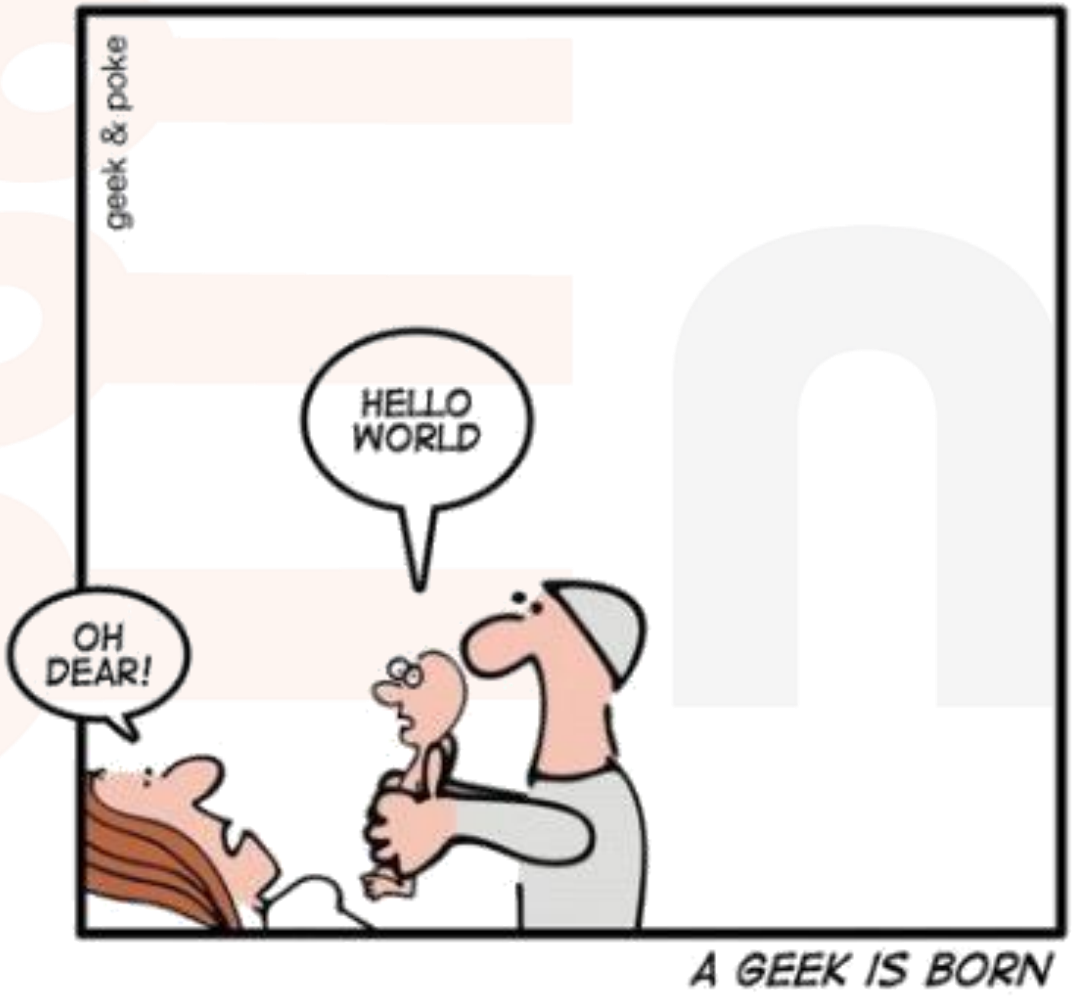




Source: wikipedia

How a Chip is Born

How a chip is born…



A&A and Specification

↓

Logic Design & IP Integration

↓

Logic Verification

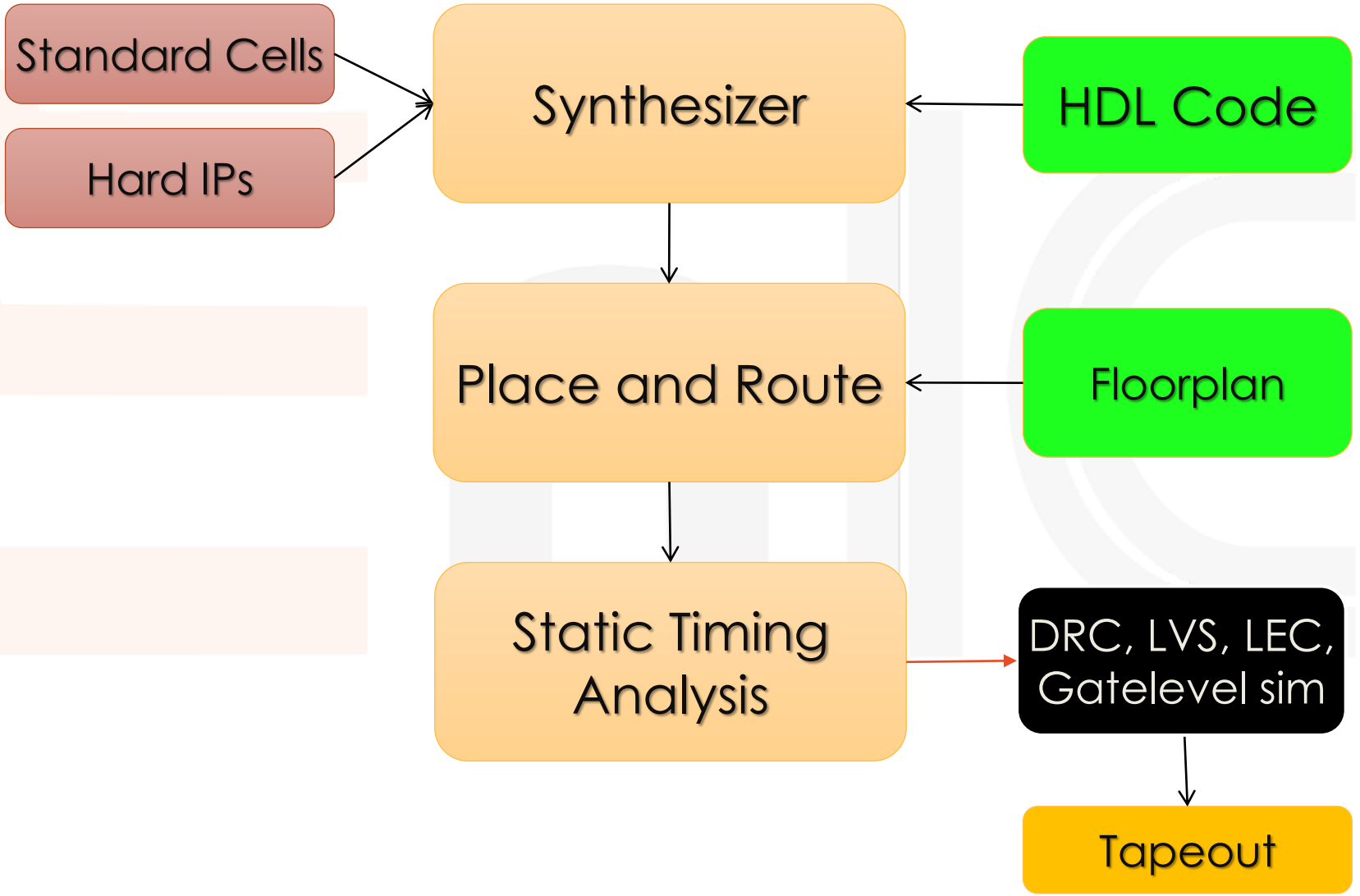
↓

Physical Design

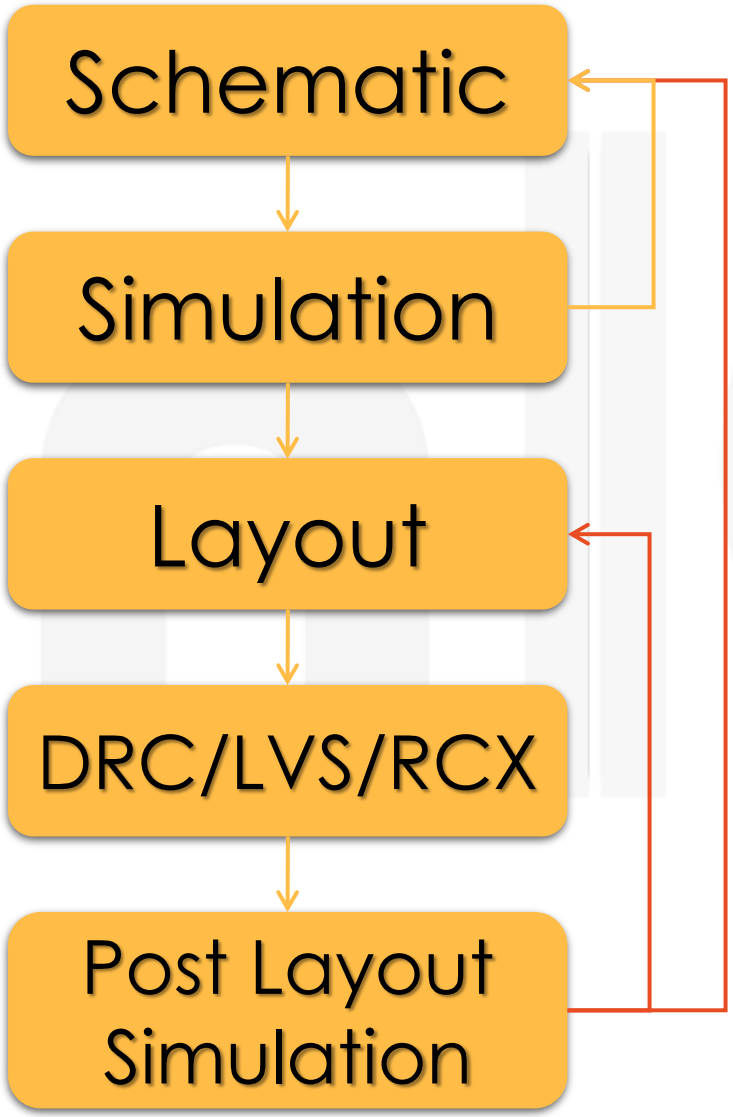
↓

Sign-off & Tape Out

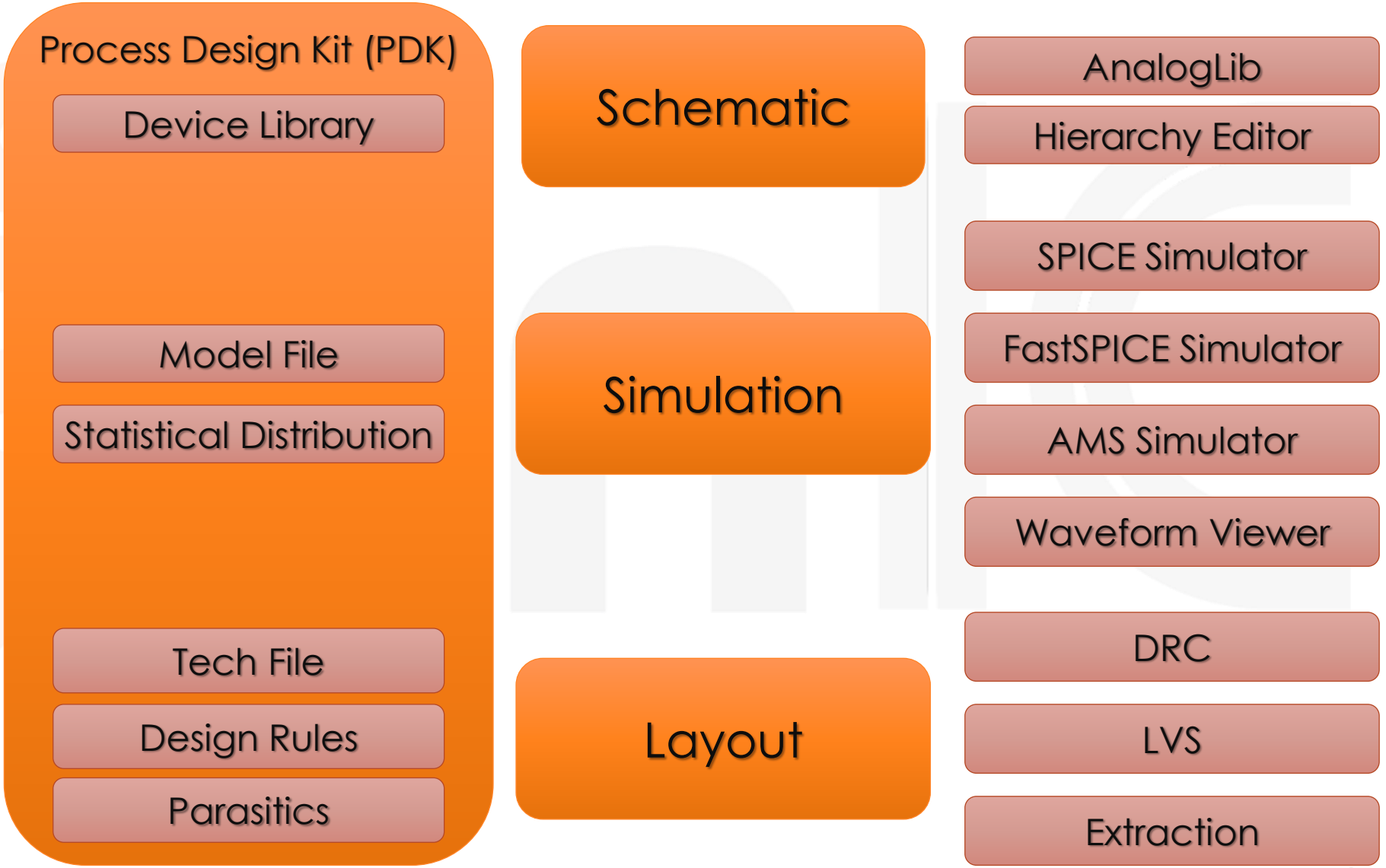
Physical Design Flow



Circuit (Custom) Design Flow

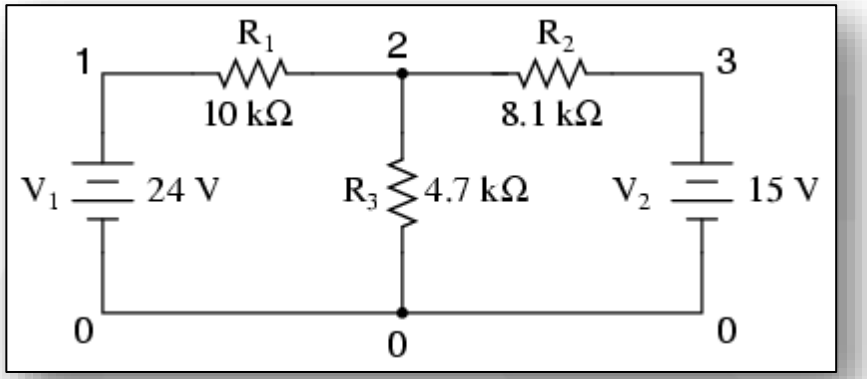


A Little Bit About Transistor Level CAD



Schematics

- **Graphical tool for drawing electrical circuits.**
- **Uses an abstract representation for:**
  - Devices
  - Sources
  - Wires
  - All wires are ideal → $R=C=0$
- **Directly translates into a netlist:**
  - SPICE netlist
  - Spectre netlist




```
V1 1 0 dc 24
V2 3 0 dc 15
R1 1 2 10k
R2 2 3 8.1k
R3 2 0 4.7k
.end
```

Simulation

- **Define your testbench and the results you'd like to measure.**
  - Create "tests" that run simulations on a "top-level" netlist.
- **Main types of tests:**
  - DC Operating Point
  - DC Sweep
  - Transient Analysis
  - AC Analysis
- **Define "outputs" to be printed or plotted.**
- **Define "variables" to enable parametric sweeps.**
- **Directly translates into a spice netlist or script (e.g., Ocean) that runs the simulations and displays the outputs.**

```
.dc V1 0 24 1
.print dc v(1) v(2,3)
.tran .05 1
.print tran v(1,2)
```

Simulation

- **Several simulators can be used in your simulation:**
  - SPICE (Spectre)
  - FastSpice (UltraSim)
  - APS
  - AMS (XPS)
- **Various models can be used to describe devices.**
  - Corner simulation changes models to simulate global process variations.
  - Monte Carlo simulations use statistical distributions to simulation local process variations.

Layout

- **After reaching optimal circuit topology, the layout editor is enacted to produce a physical representation of the circuit.**
- **The process “techfile” describes the various design layers and design rules.**
- **In order to assist the layout designer in circuit compilation, use connectivity between the layout and schematic:**
  - Utilize “p-cells” of the schematic devices.
  - Net names connect between the schematic and layout.
  - Fly-lines and Wires automatically connect nodes on the same net.
  - Pin layers are used to describe schematic ports/pins.

Layout Verification – DRC/LVS/Extraction

- **Design Rule Check (DRC) ensures that the layout meets the design rule requirements.**
- **Layout Vs Schematics (LVS) ensures that the layout truly represents the circuit.**
- **Parasitic Extraction (RCX or PEX) extracts the parasitic resistance, capacitance and inductance of the layout.**
- **The most common tools for performing layout verification are:**
  - Cadence PVS/QRC/Pegasus
  - Mentor Graphics Calibre
  - Synopsys StarXT

DRC

- **DRC can be applied interactively in the Layout editor (DRD)**
- **The DRC checker tests complex geometric patterns for each layer as described in the PDK's rule file.**
- **DRC warnings include:**
  - Circuit level rules
  - Chip level rules
  - Recommended Rules (For DFM)

LVS

- **LVS uses a greedy algorithm to compare netlists.**
  - The "***Source***" netlist is the schematic (already in netlist format).
  - The "***Target***" netlist is the layout as compiled by an extractor that finds devices and connects them.
- **The starting point for the LVS checker is the interface (pins/ports). If these are incorrect, you will get "strange" errors.**
- **LVS also checks various design requirements such as bulk connections.**
  - These are also known as ERC (Electrical Rules Check)

RCX

- **The parasitic extraction tool creates a netlist that includes the non-idealities of the layout.**
  - Actual device sizes, such as Ldiff.
  - Resistance and Capacitance of wires.
  - Coupling Capacitance
- **In order to run extraction, you must FIRST run LVS.**

- **The output of is a list of net parasitics and a netlist that can be plugged back into the circuit simulator for comparison to pre layout results.**
- **Extraction tools include Cadence QRC, Mentor Graphics Calibre, and Synopsys Star-XT.**

Post Layout Simulation

- Pre-layout simulation is done according to generic models with average device parameters and ideal wires.
- After creating a layout of your design, it is essential to re-simulate your design with the actual parasitics – for better or for worse.
- Your circuit will now have a new “view” in addition to its “schematic” view. This view is called “extraction” or “calibre” or something similar.
- The common way to tell the simulator to use this view is by invoking the *Hierarchy Editor.*

Important Concepts From This Lecture

- Abstraction
- Scaling
- Flavors
- Foundry
- Interconnect
- Schematics
- Simulator
- Layout
- Parasitics
- Extraction
- Interface
- Instantiate
- Hierarchy
- Verification
- Standards & Protocols
- Design Styles
- Trade-offs
- Full Custom Design
- Standard Cell Design
- Libraries

Important Concepts From This Lecture

- Gate Array
- Sign Off
- Tapeout
- Synthesis
- Floorplan
- Place and Route
- Static Timing
- Post Layout Simulation
- Corners
- Monte Carlo
- Ocean Script
- Virtuoso
- Calibre
- Assura
- SPICE
- Netlist
- Testbench
- DC Operating Point
- Transient Analysis
- DC Sweep
- AC Sweep
- Process Variations

“Three Letter Words”

- PDK – Process Design Kit
- HDL – Hardware Description Language
- EDA – Electronic Design Automation
- ASIC – Application Specific Integrated Circuit
- FPGA – Field Programmable Gate Array
- IP – Intellectual Property
- DRC – Design Rule Check
- LVS – Layout Vs. Schematics
- RCX – Resistance/Capacitance Extraction
- ADE – Analog Design Environment

Further Reading

- J. Rabaey, "*Digital Integrated Circuits*" 2003, Chapter 1.3
- E. Alon, Berkeley *EE-141*, Lecture 2 (Fall 2009)
  http://bwrc.eecs.berkeley.edu/classes/icdesign/ee141_f09/
- …a number of years of experience!