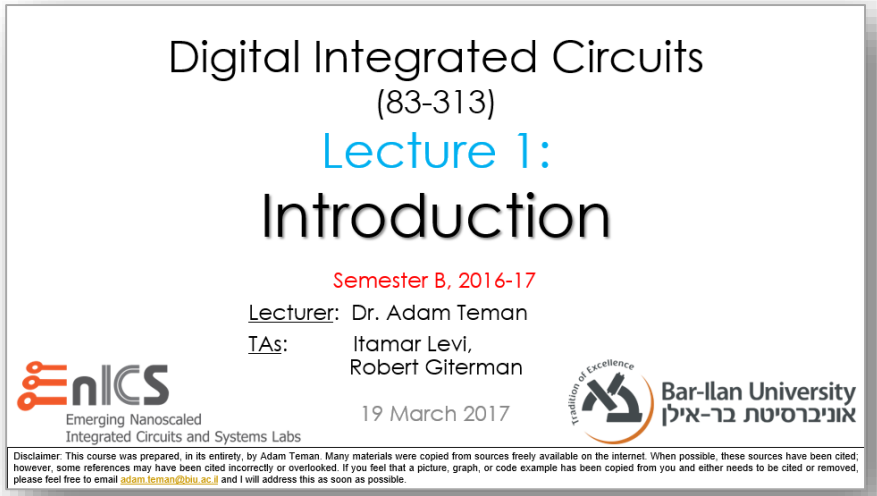
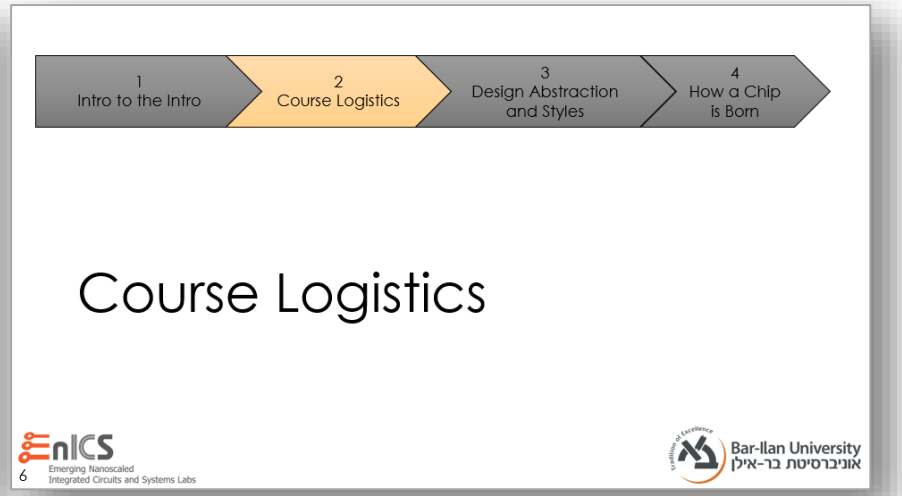
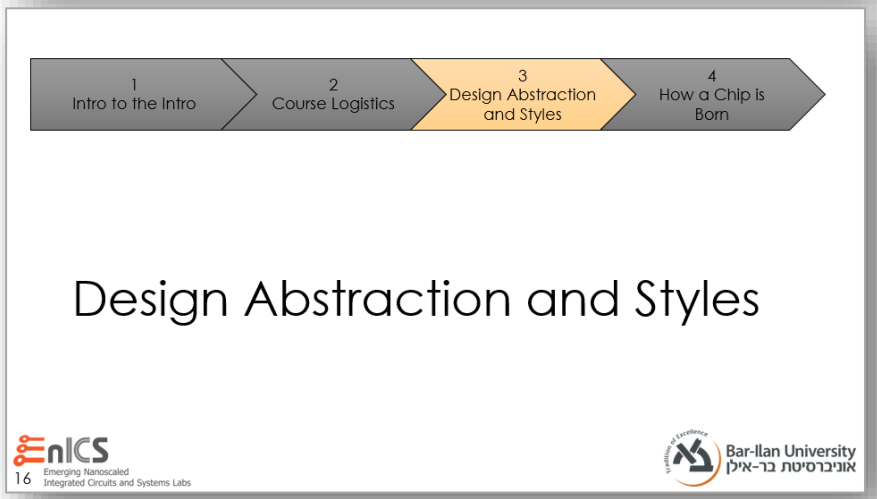
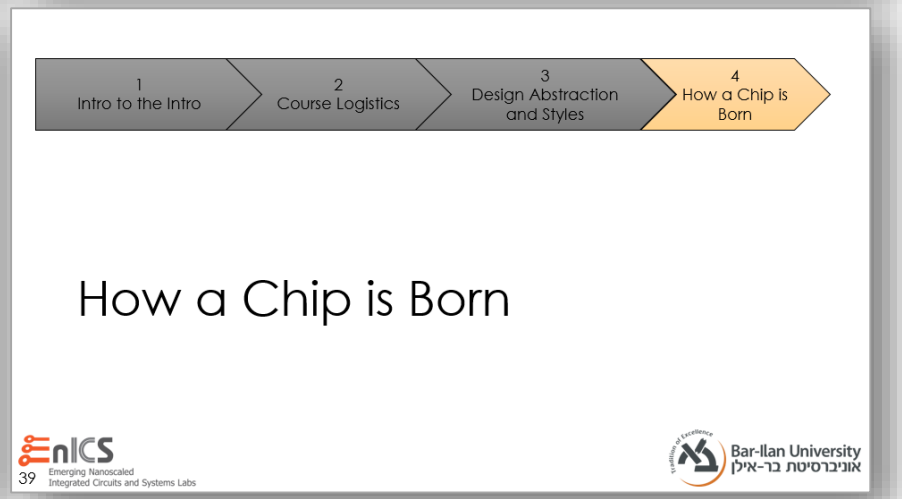
Lecture Outline









Digital Integrated Circuits
(83-313)
Lecture 1:
Introduction

Semester B, 2016-17

Lecturer: Dr. Adam Teman

TAs: Itamar Levi,
Robert Giterman

19 March 2017

Emerging Nanoscaled
Integrated Circuits and Systems Labs

Tradition of Excellence
Bar-Ilan University
אוניברסיטת בר-אילן

What will we do in this Course?

- **In *Digital Electronic Circuits* (83-308), we learned the basics of building a digital gate and got to know a number of methods in digital circuit design.**
- **In the *Digital Circuits Lab* (83-315), we learned how to use CAD/EDA tools to implement basic circuits at a schematic and layout level.**
- **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.

What will we do in this Course?

- **This course deals with the *Circuit Level* and *Gate Level* design of digital components touching on the *Device Level* and *Module Level*.**
- **To get a better "hands-on" understanding, we will use *Cadence Virtuoso* to:**
  - Compile our circuits.
  - Simulate circuit operation.
  - Understand physical implementation and process variations.
- **Upon completion of this course, you will have a good understanding of what's happening "*underneath the HDL*".**

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 Logistics

Class Etiquette (i.e., Behavior!)

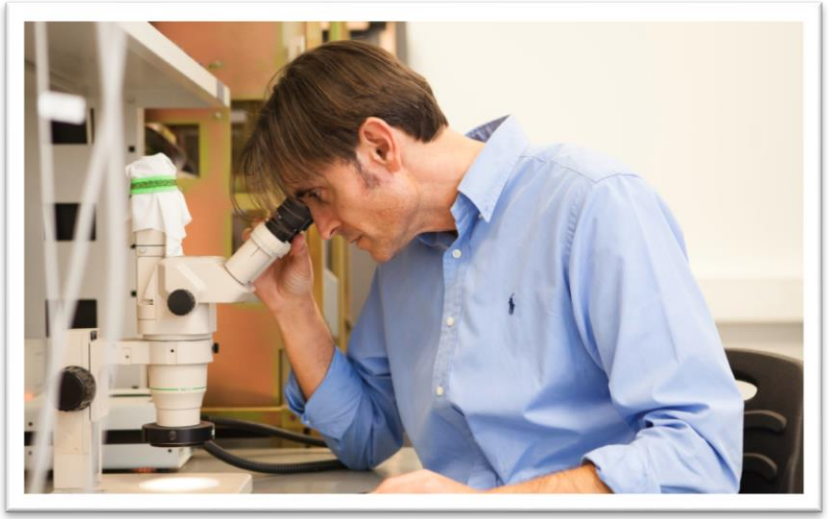
- **Cellphones:**
  - Put all cellphones on silent at the beginning of class.
  - Messaging during class?
- **Talking to your neighbors…**
  - Why not just ask me?
  - If you missed something due to some important whatsapp, I probably won't answer you!
- **Break**
  - No breaks this year!
  - Instead, we have one 1.5 hour lesson and one 45 minute lesson.
  - Please be on time. We have a lot to learn and very little class time.

Course Administration

- **Lecturer: Dr. Adam Teman**
  - Email: adam.teman@biu.ac.il
  - Lectures:
    - Monday 10:00-12:00. Auditorium 042
    - Tuesday: 1500-1600 Auditorium 002
  - Office Hrs: Monday 12:00-1300. Office 413
- **Teaching Assistants:**
  - Itamar Levi: itamar.levi@biu.ac.il
    - Office Hrs: by appointment
  - Robert Giterman: robert.giterman@biu.ac.il
    - Office Hrs: by appointment
- **Labs:**
  - Tuesday 1600-1800. Room 368

Who are we?

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

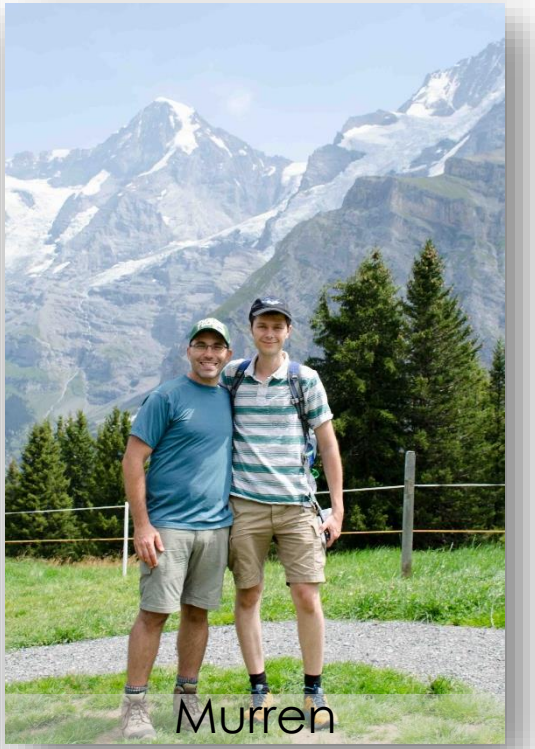


The EnICS Family


San Francisco

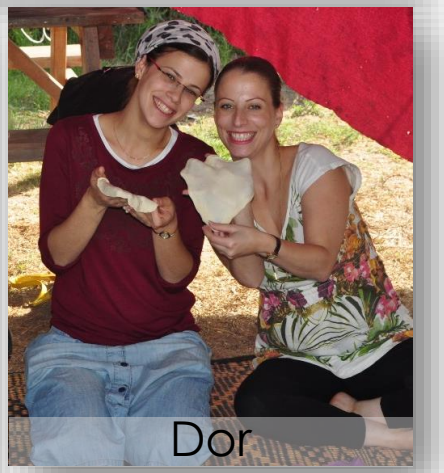

Akko


Grindelwald


Murren


Leysin


Seoul


Dor

Grading

- **Quizzes:**
  - Two Quizzes: 50%
    - May 9, 2017
    - June 27, 2017
  - Quizzes will be held in class (Tuesday lecture)
  - Material will include lectures up to that point.
  - Beware – I ask about *everything*!
    - In other words, you are expected to not only watch the lectures and do the homework, but re-watch the lectures *at least twice* before the quiz.
- **Homework**
  - Homework/Lab Assignments: 50%
  - This will include an assessment of each student in the lab!

Late Submission Policy:
Don't even bother asking for an extension…
Every day you are late will cost you **5 points**.
Be on time – you won't be sorry…

Labs

- **Labs are divided into *frontal* and *work* labs**
  - Frontal labs will include a 1 hour lecture, given twice.
  - Work labs are two hour sessions, during which you can sit and work in the lab with on-site help from Itamar and Robert.
- **Homework is submitted in groups of two!**
  - Since the labs are based on Virtuoso, it is *mandatory* for Computer Engineering students to pair with Electrical Engineering (who already did the Digital Circuits Lab).
  - Homework assignments are *long*. If you wait until the last minute to start working on them, ***you have no chance*** to complete the assignments in time.
- **Use the Moodle Forum**
  - We will happily help you with questions about the homework assignments, but:
    - Do not send us private emails about assignments (we will copy them to the forum)
    - Please provide ***all information*** along with the question, so that we (or your friends) can help.
  - Please share your problems and how you solved them on the forum!
- **Note that, if you haven't signed a CIU, you better do it today!**

Lab Subjects

- **Subject 1: Standard Cell Design**
  - Design and simulation of a complex standard cell in 65nm CMOS.
  - Layout according to Standard Cell design guidelines.
  - Building a timing/power model through library characterization.
  - Creating a place and route abstract with the Abstract Generator.
- **Subject 2: Analog Mixed-Signal Simulation**
  - Setting up an Analog Mixed-Signal Flow
  - Running Simulations with analog devices, Verilog code and Verilog-A models
- **Subject 3: SRAM Design and Characterization**
  - Designing and Measuring SRAM Noise Margins
  - Designing a small SRAM macrocell

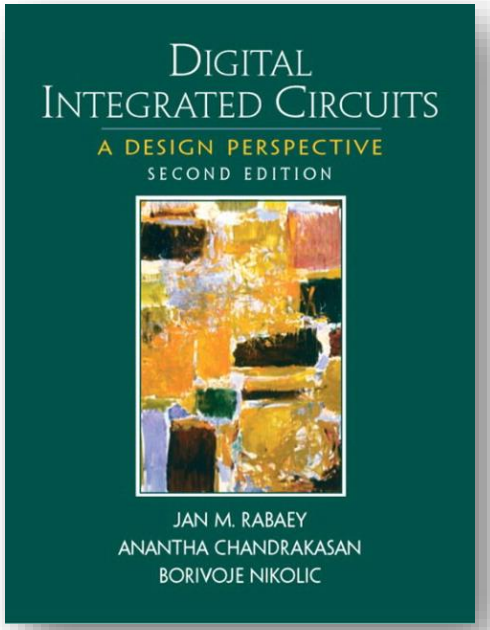
Late Breaking Lab:
We *may* add a lab on
FinFET layout.
If we do, you won't be sorry!

Reading Material


Jan Rabaey


Anantha Chandrakasan


Borivoje Nikolic



- **Main Book:**
  - Rabaey, et al., "*Digital Integrated Circuits: A Design Perspective*", 2nd Edition, 2003.
  - Highly recommended: UC Berkeley ***EE141*** – lectures online (most recent from 2012)
- **But most lectures will have a reading list.**
  - In order to get a better understanding, it is highly advised to use these references!
- **In addition, all lectures will be recorded**
  - Be smart and utilize this resource!


Elad Alon

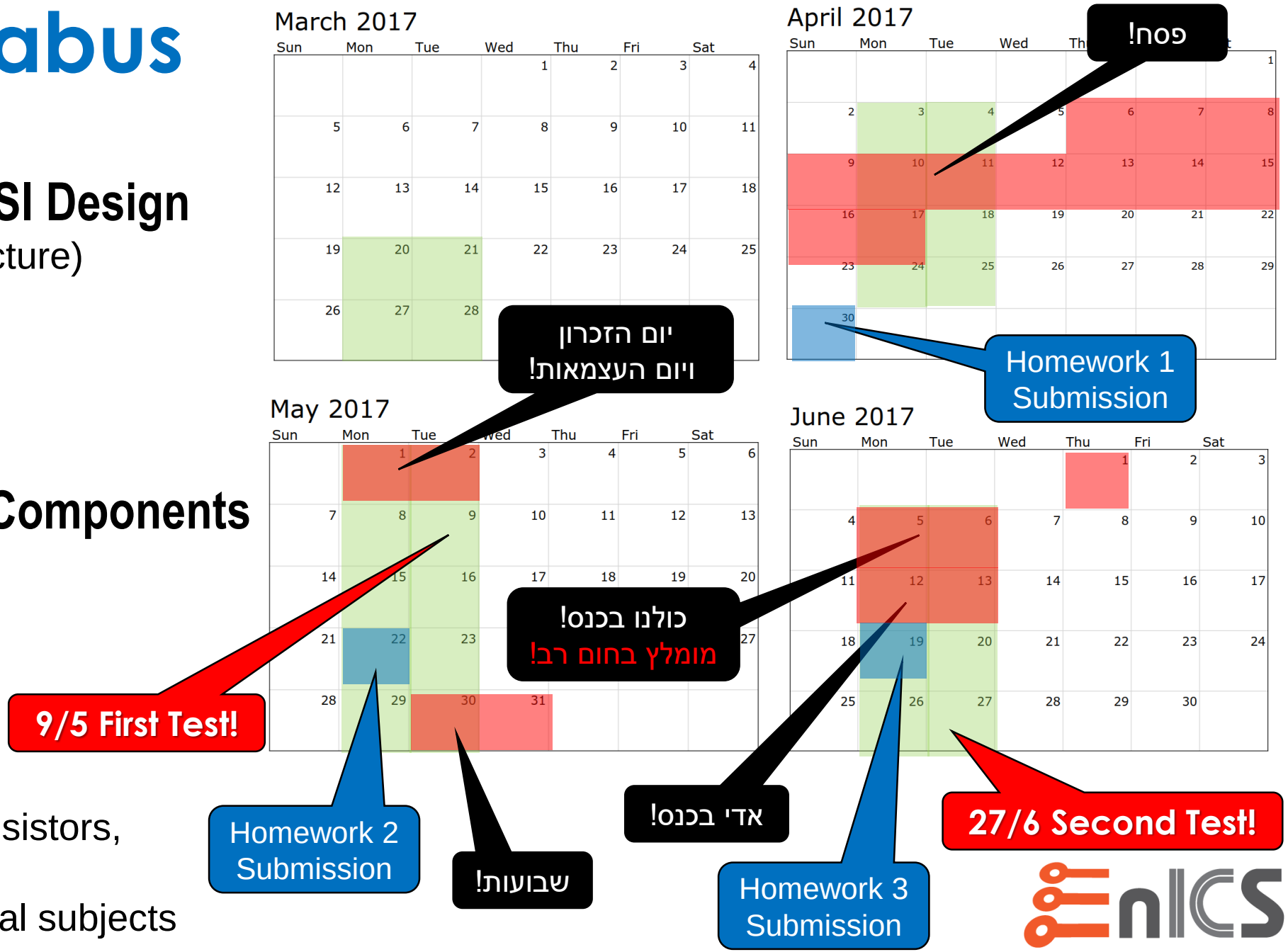
Course Syllabus

- **Part 1:**

  Introduction to VLSI Design
  - Introduction (This Lecture)
  - Standard Cell Design
  - Design Metrics
  - Technology Scaling
- **Part 2:**

  Designing Digital Components
  - Sequential Logic
  - Arithmetic Circuits
  - Memory Design
- **Part 3:**

  VLSI Technology
  - Process, Scaled Transistors, Modeling, Effects
  - FinFETs and additional subjects

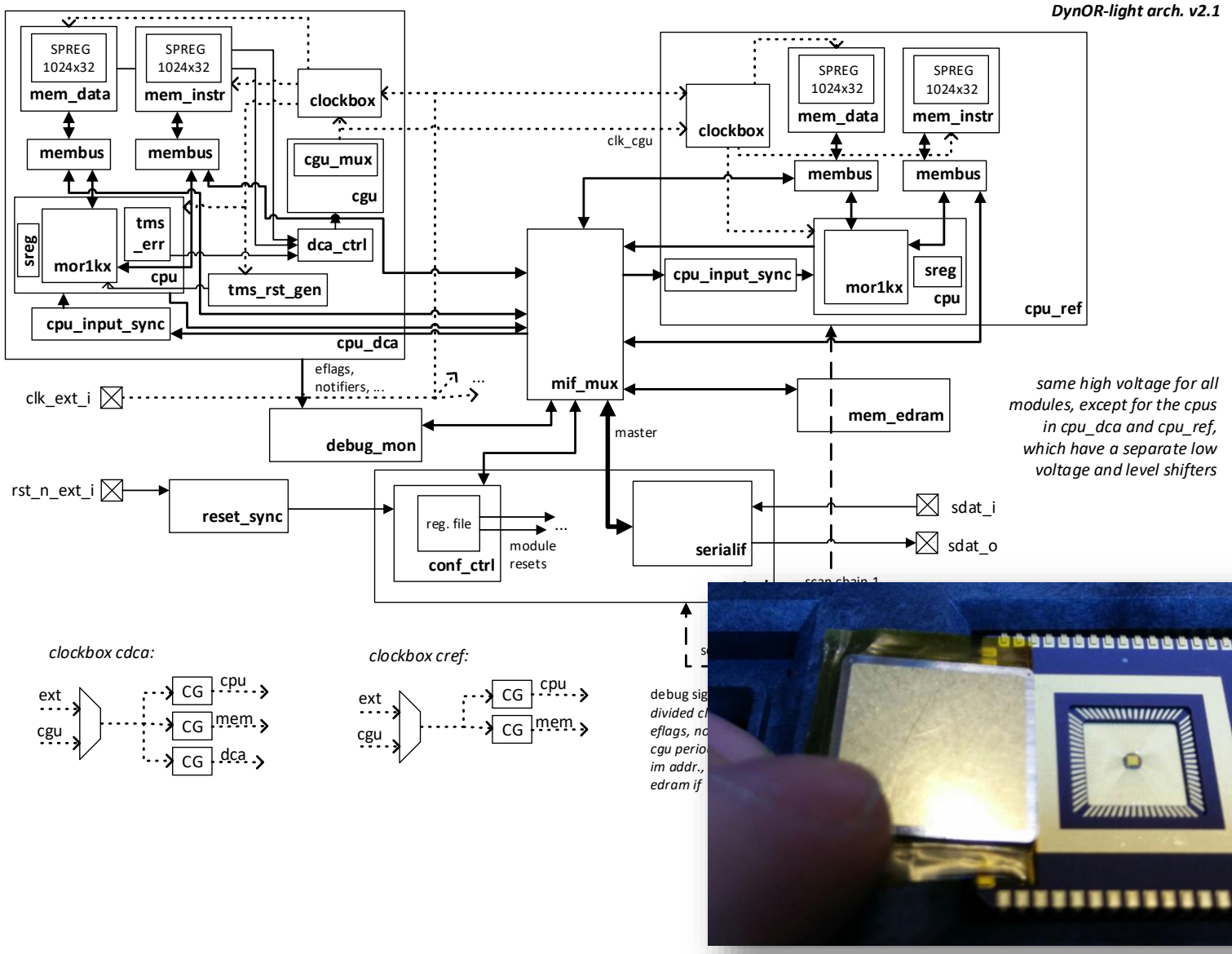


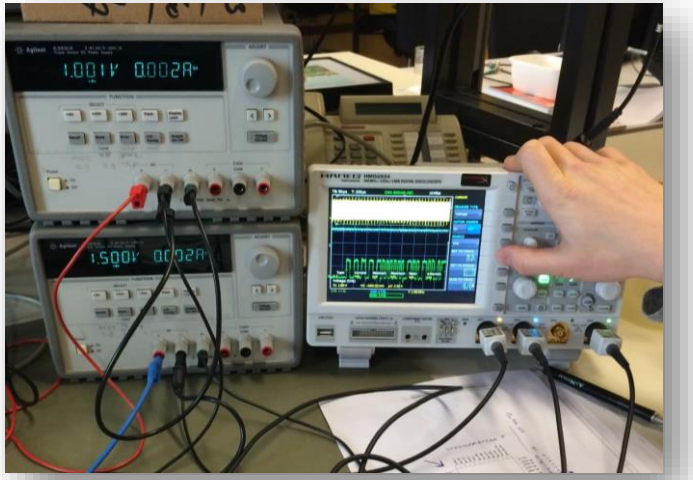
Design Abstraction and Styles

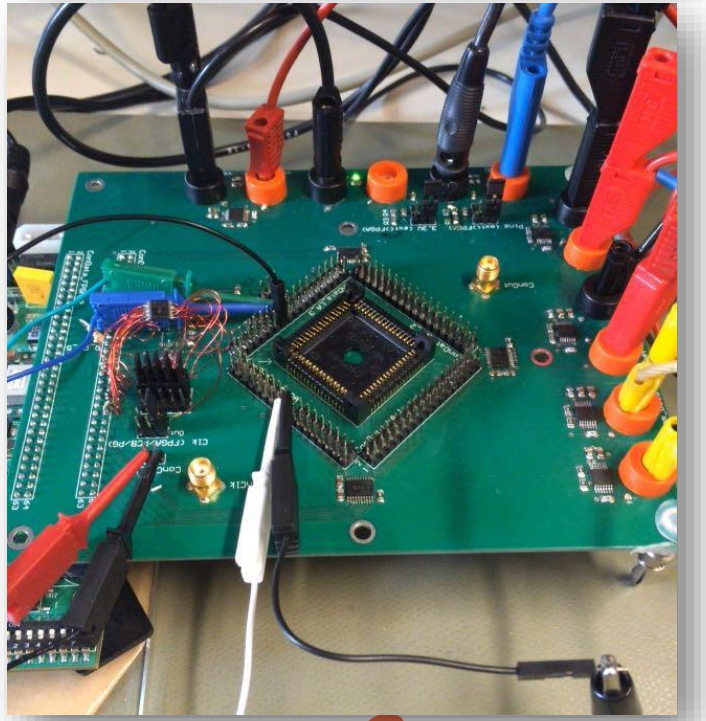
Primer: What is an IC Chip?

Primer: What is an IC Chip?

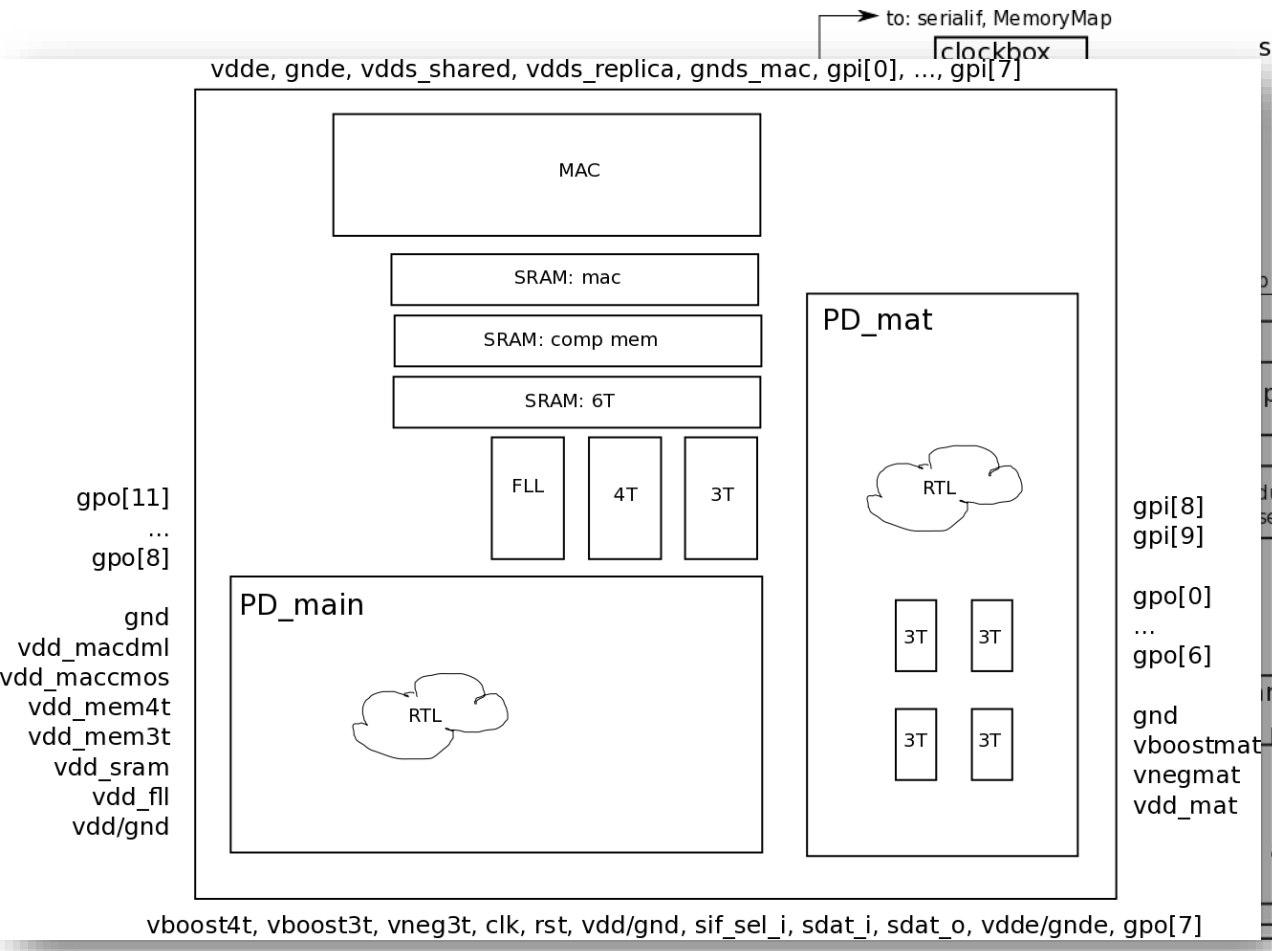
Example: dynOR

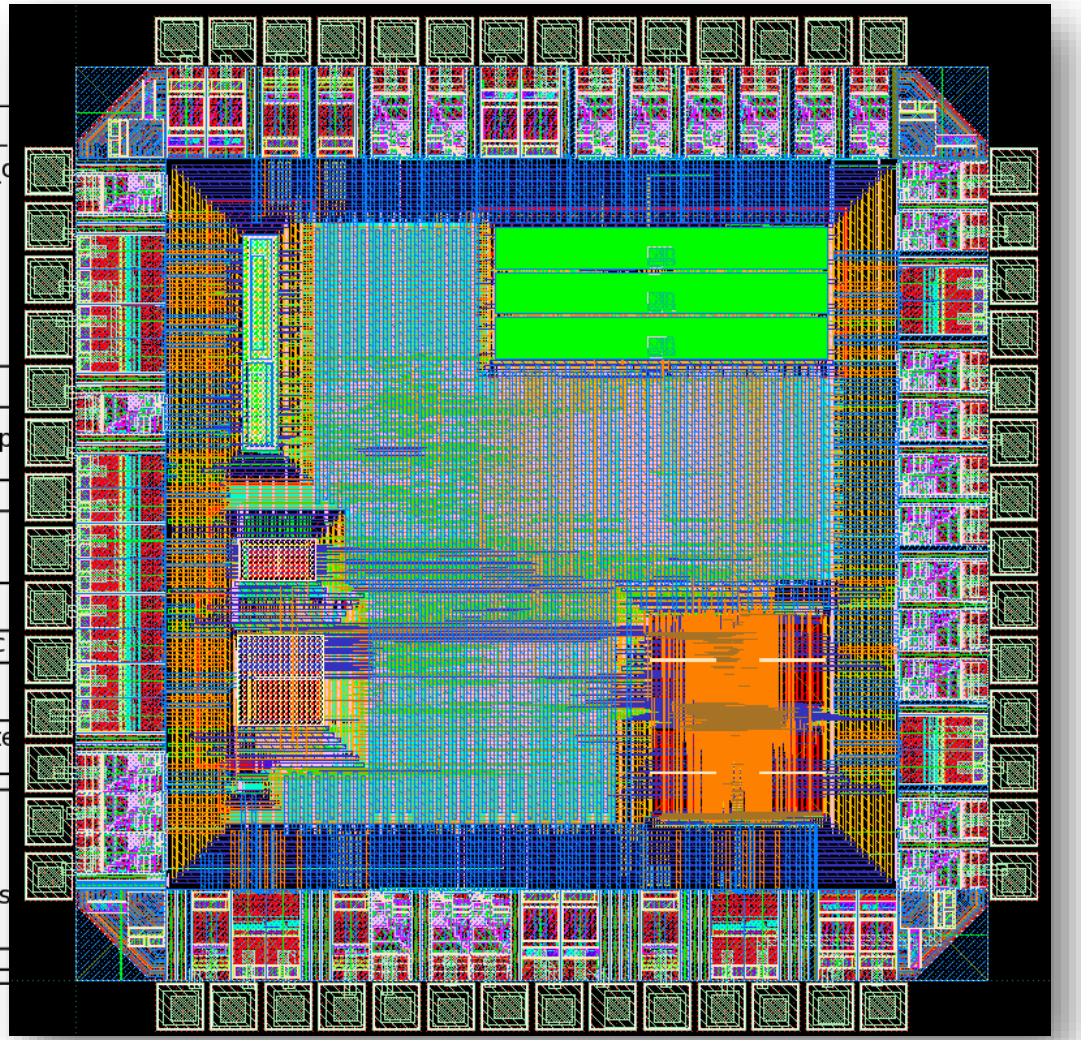






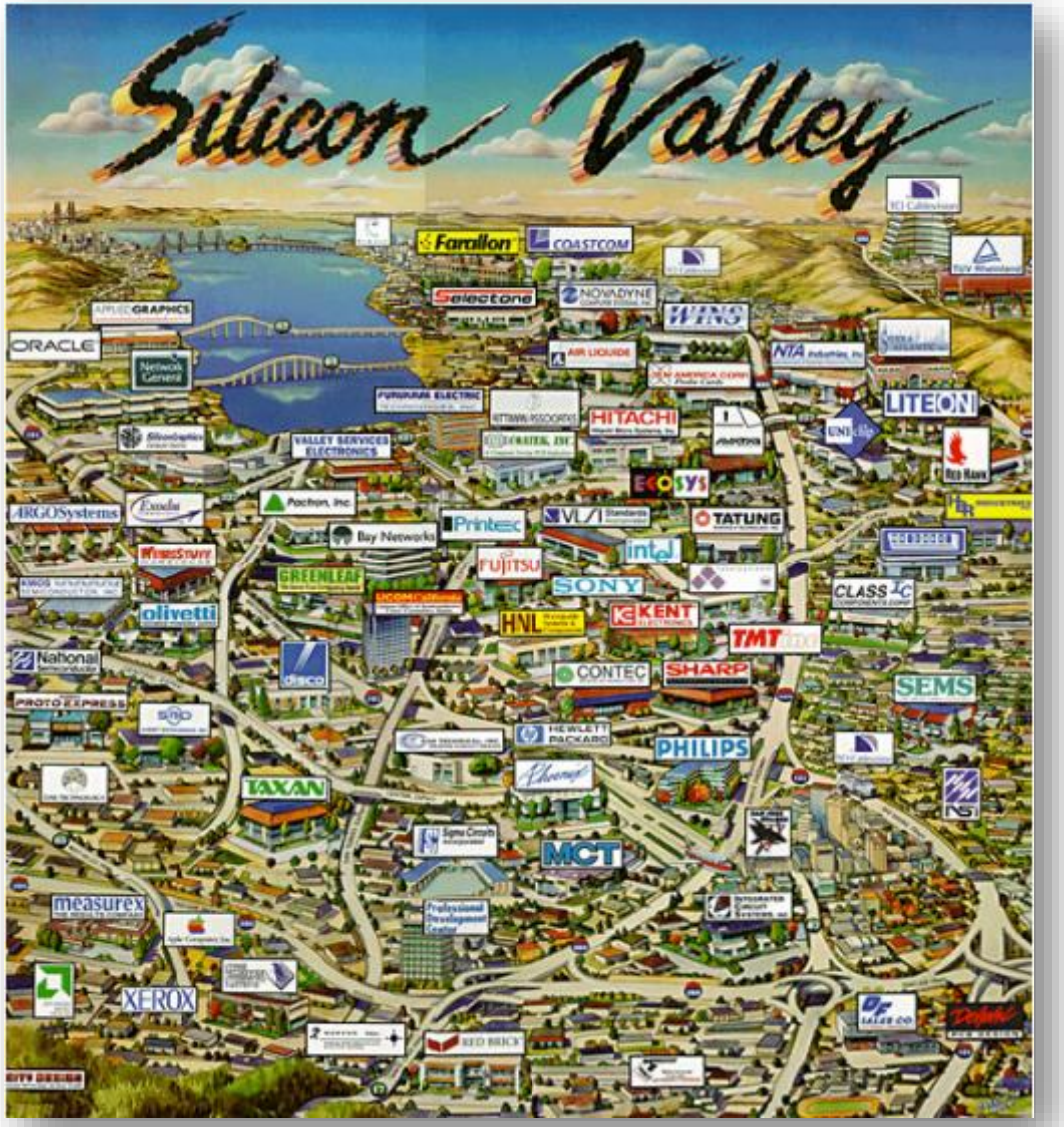
Example 2: BEER





The VLSI Design Ecosystem

Silicon Valley



Digital vs. Analog

- **Digital design enables Abstraction!**
  - Abstraction is only available when there is separation (i.e. no feedback) between blocks.
  - In other words – noise suppression.
- **This is very difficult with Analog design.**

- **In addition, Digital design is *Scalable*, whereas the parasitic effects introduced by scaling cause limitations to analog scaling.**

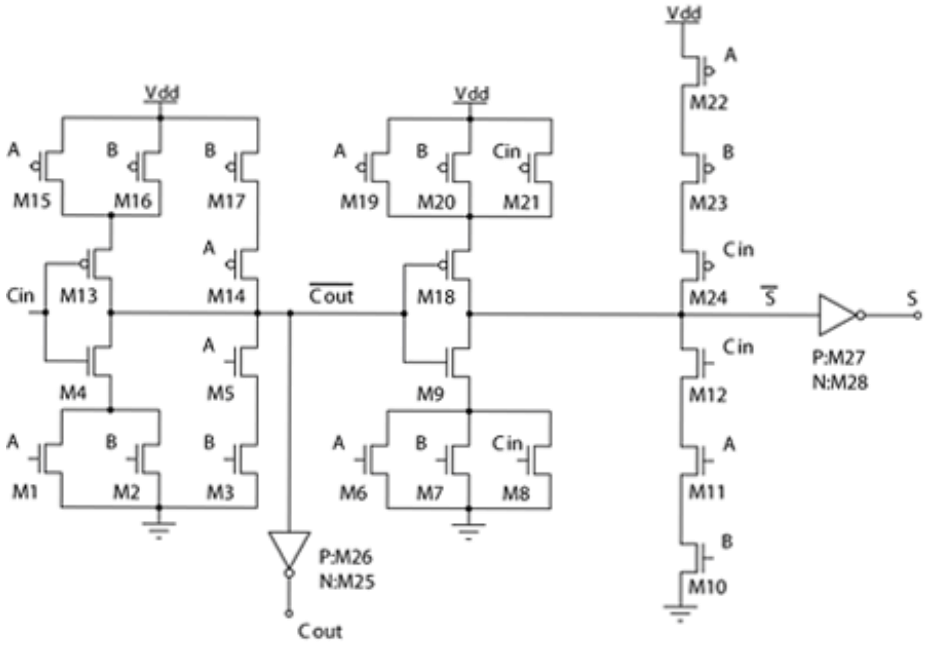
Device Level Abstraction

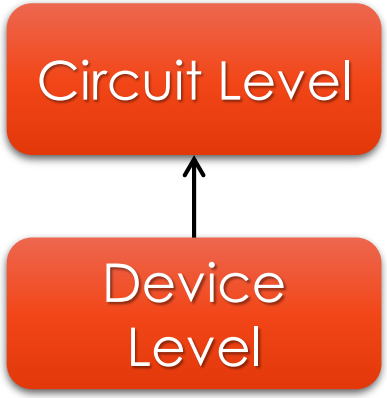
- **Fabrication Plants or *Foundries* supply a *Process Design Kit* (PDK).**
- **The PDK includes:**
  - Devices
    (Transistors, Resistors, Capacitors, Diodes)
  - Layers
  - Rules
- **Various "Flavors" of PDKs are available, e.g.:**
  - General Purpose/High Speed/Low Power
  - RF/Image Sensor
  - Flash/DRAM
  - Various number of Metal Interconnect Layers

Device Level

Circuit Level Abstraction

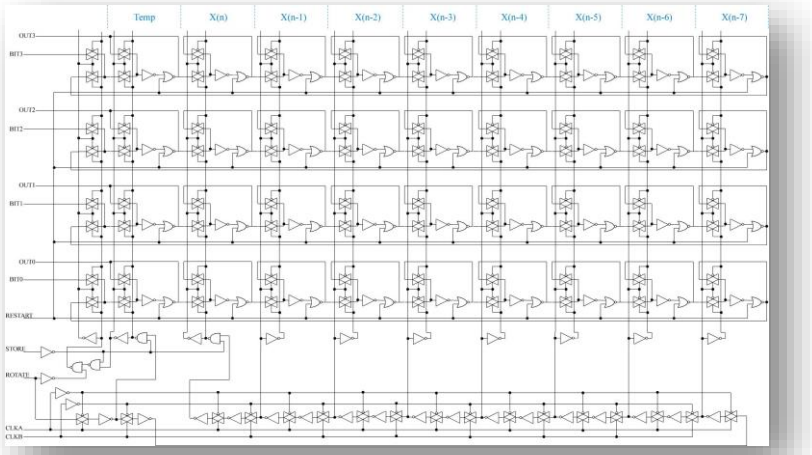
- **Using the devices supplied in the PDK, *Schematics* are drawn.**
- **Simulators, such as *SPICE*, are used to test and optimize the circuits.**
- **Various parameters (called *CDF parameters*) can be modified to optimize the schematic, e.g.:**
  - Length and Width of transistors.
  - Number of Fingers.
  - Capacitances and Resistances.
- **Circuits are drawn with a *Layout Editor* and *parasitics* are extracted for accurate simulation.**



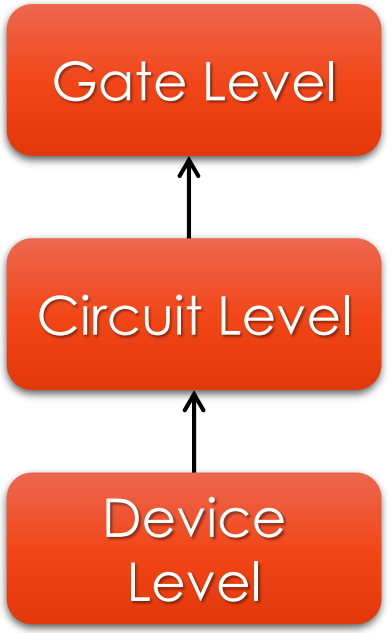


Gate Level Abstraction

- **Drawn circuits are abstracted into a black box for use as gates.**
- **These gates are defined by easy to use characteristics, such as:**
  - Boolean Functionality.
  - Interface (i.e. *pins* or *ports*).
  - Delay and power consumption.
  - Input and output capacitance.
  - Size and geometry.
- **Once a gate is abstracted, it can be used by higher level tools, such as *HDLs*.**

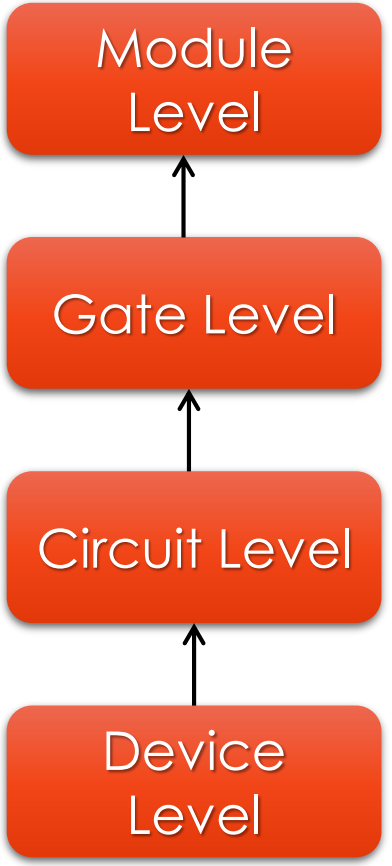


8-bit adder gate-level schematic.
http://pneuhiver.info



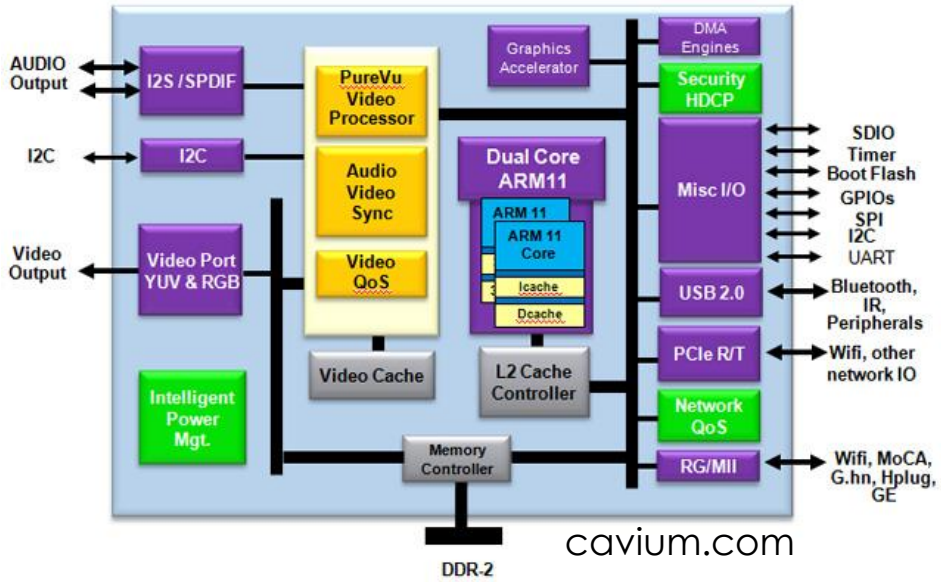
Module Level Abstraction

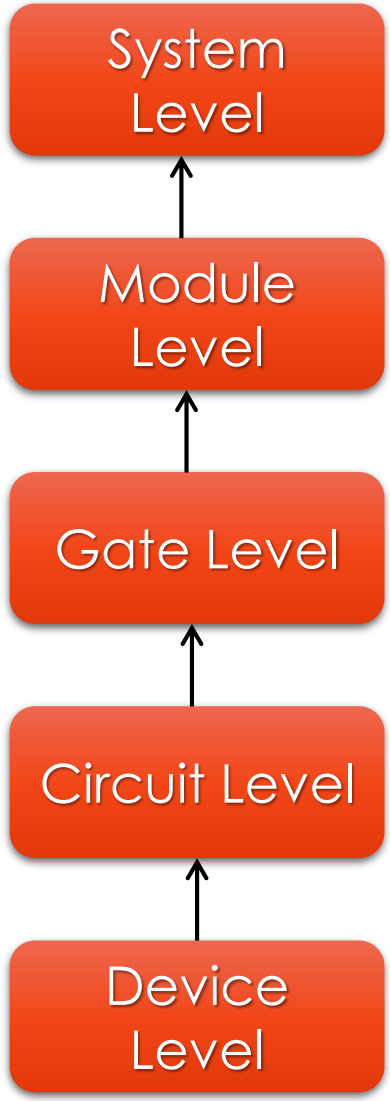
- Gates and other low level circuits are connected together into modules (adders, memories, etc.).
- These are tested for functionality and are abstracted for system integration.
- Analog modules are abstracted from circuit level.
- Digital modules and full systems are defined with HDLs, *instantiating* gates and modules.
- EDA tools are used to verify functionality at all depths of hierarchy.



System Level Abstraction

- **Architectural design defines high level abstractions to build a system.**
- **This abstraction level defines:**
  - Registers
  - Instruction Sets
  - Control Blocks
  - Buses
  - etc.

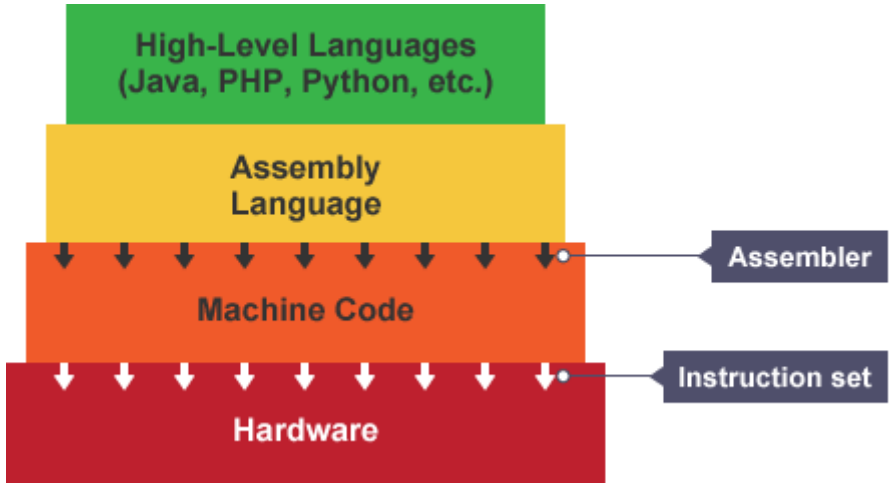




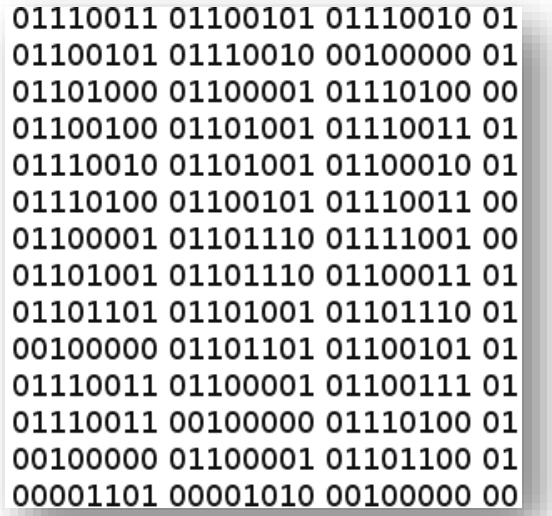
- **Systems are implemented with HDLs and functionality is verified with logical verification.**
- **System are defined to comply with *standards* and implement *protocols*.**

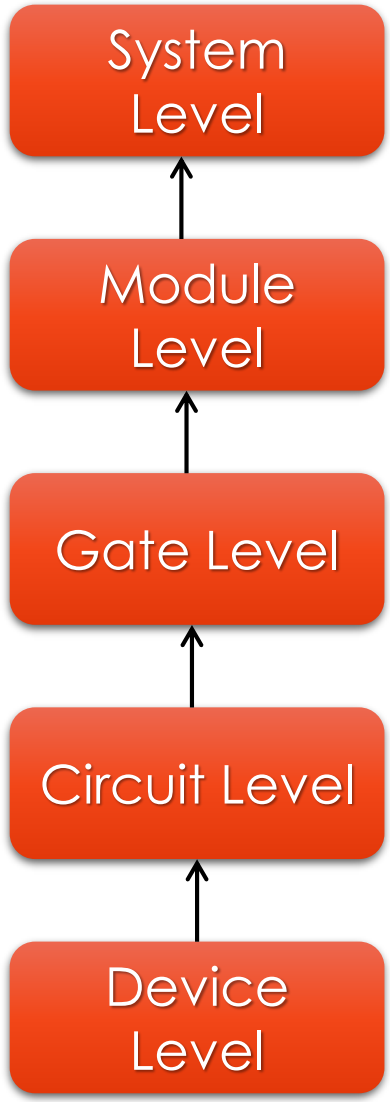
Higher Level Abstraction

- **Programmers write code in a *high level programming language* (C, Java, Perl, etc.)**
- **A *Compiler* translates the code into an *Assembly* language.**
  - Compilers try to optimize the instructions according to the actual architecture they are compiling to.
- **An *Assembler* translates the Assembly into *Machine Language* (i.e. 0's and 1's).**

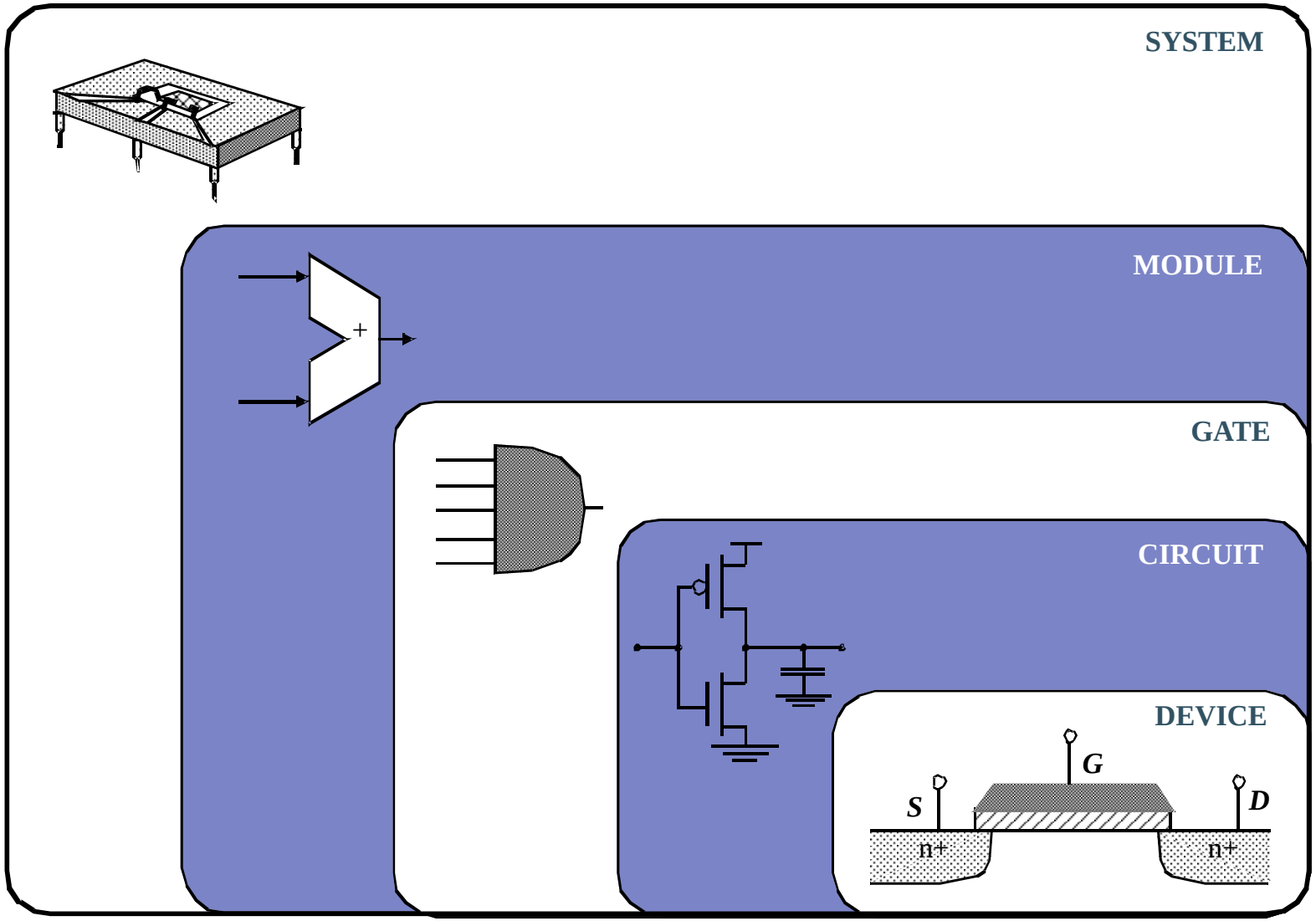


bbc.co.uk





Design Abstraction Levels

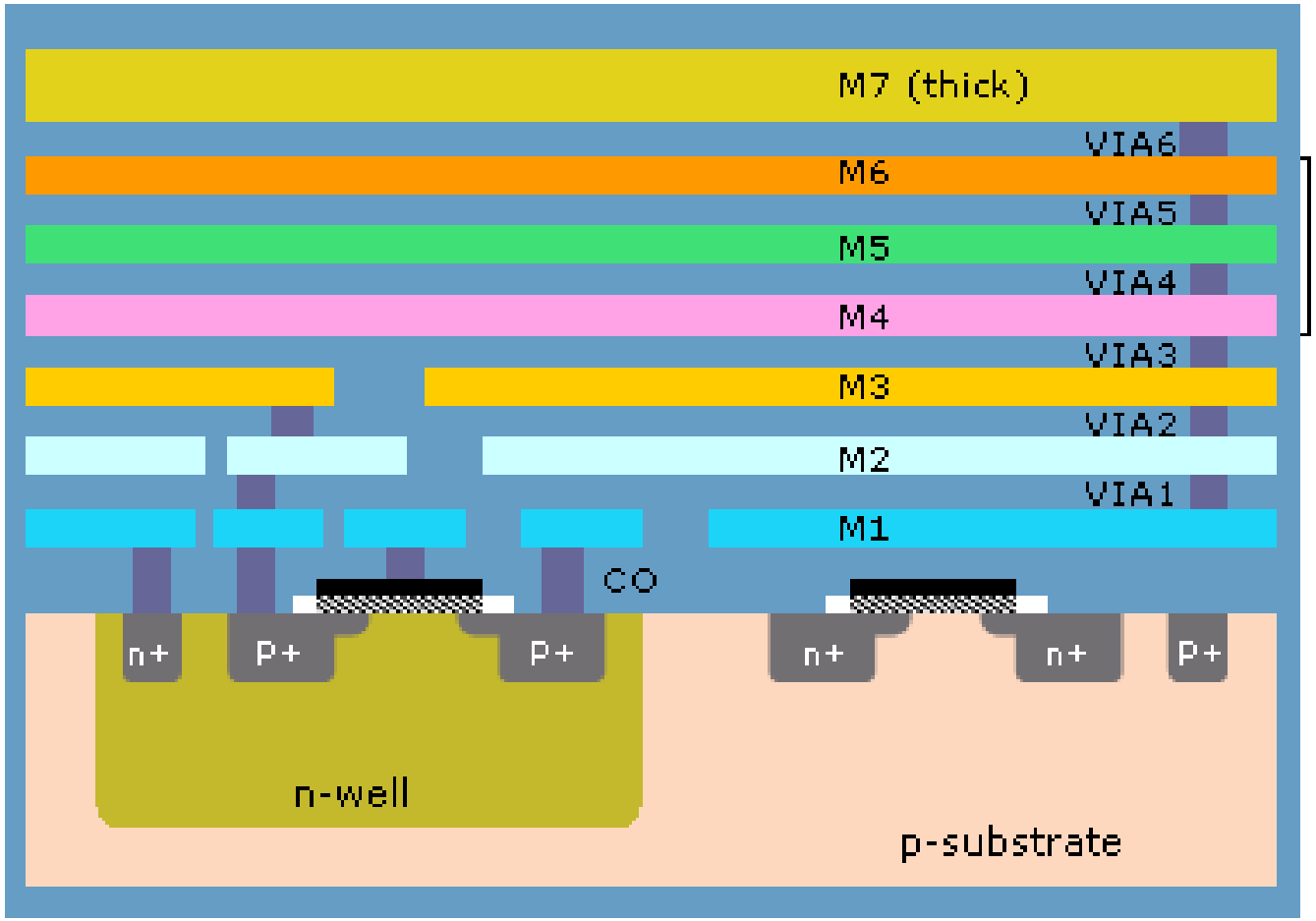


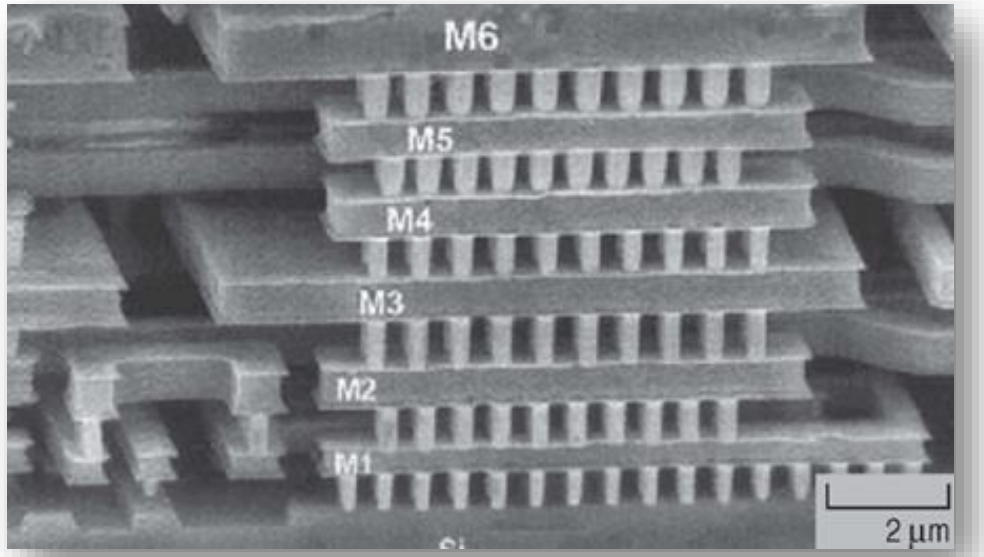
Who actually does this work?

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
  - Gate Array design
  - Field-Programmable Gate Array (FPGA) design

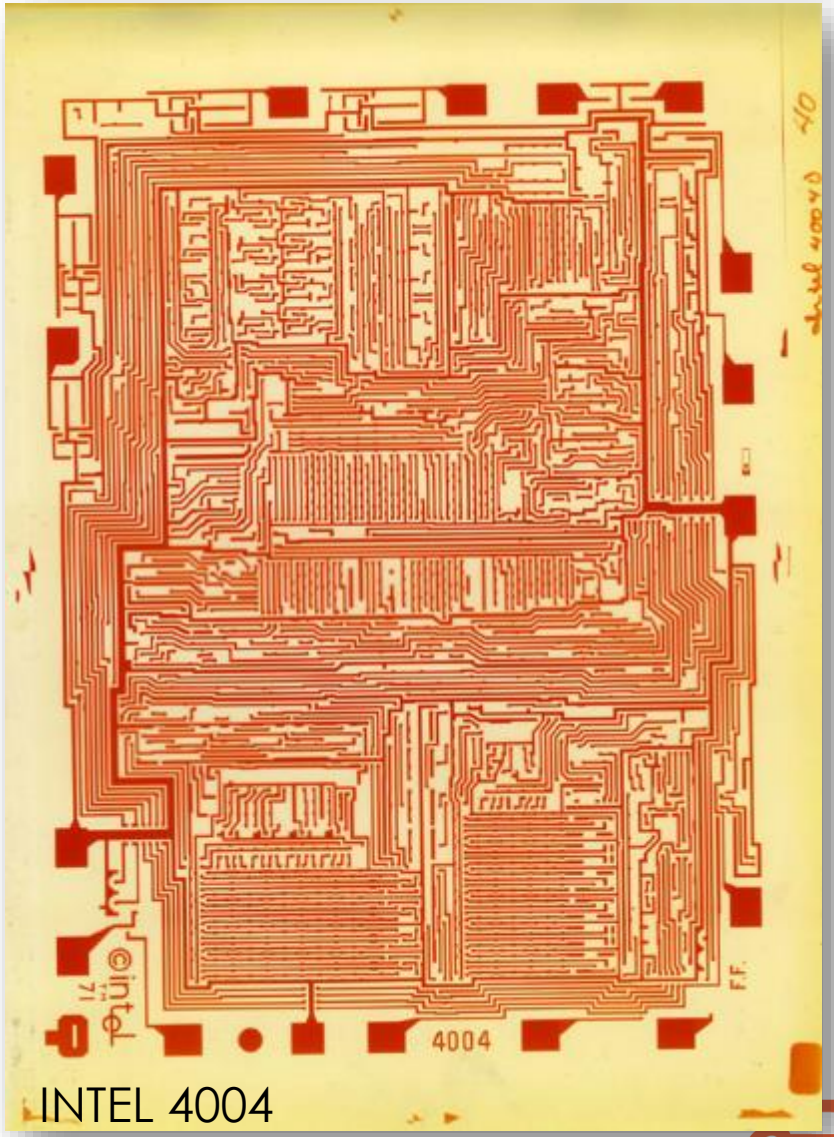
CMOS Technology





VLSI Design Styles – Full Custom

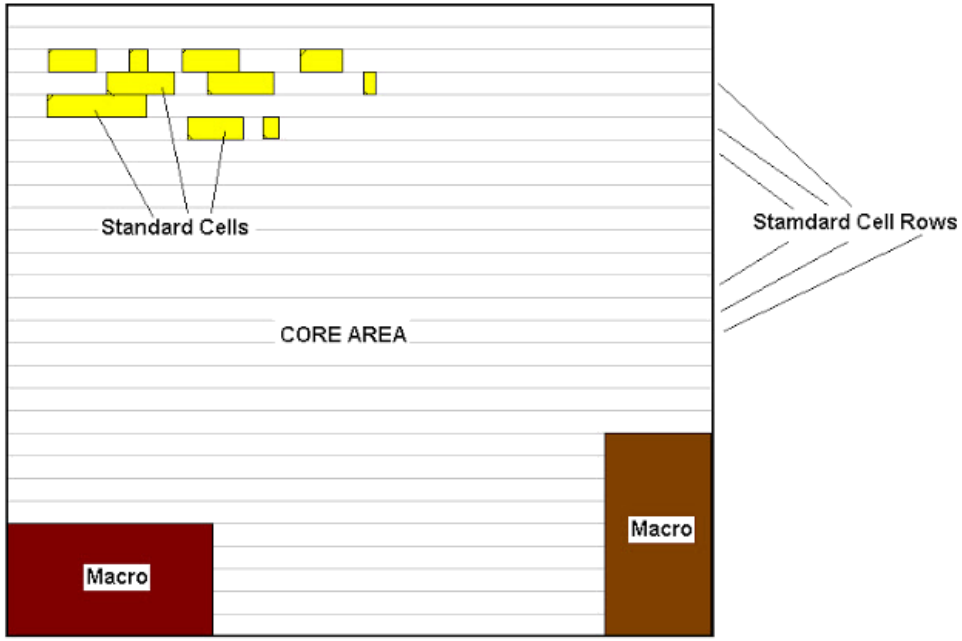
- **Full Custom Design**
  - The original design style.
  - Everything is done at transistor level
  - Rarely used in digital design
  - High cost
  - But, gives significant gain in performance.
  - Analog designs mostly



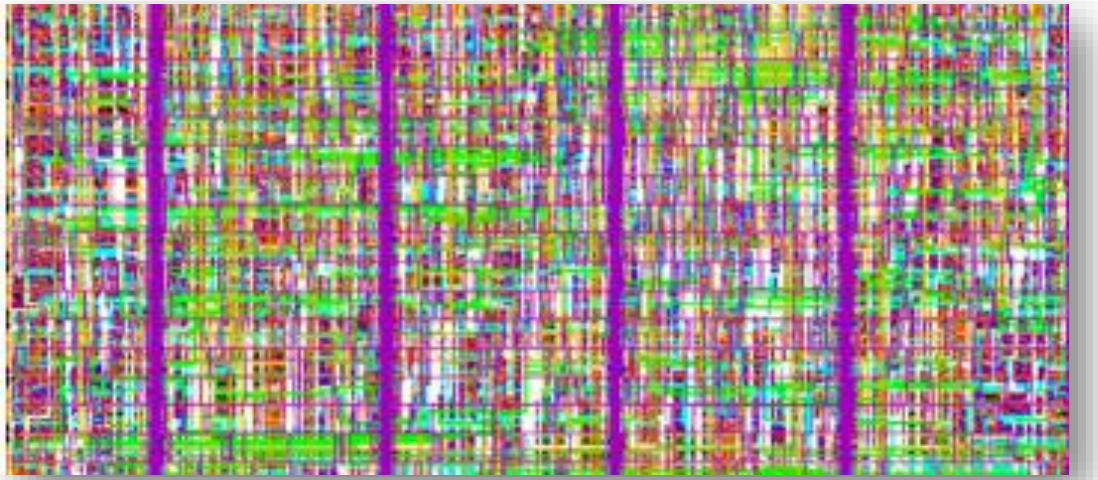
INTEL 4004
www.arstechnica.com

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.

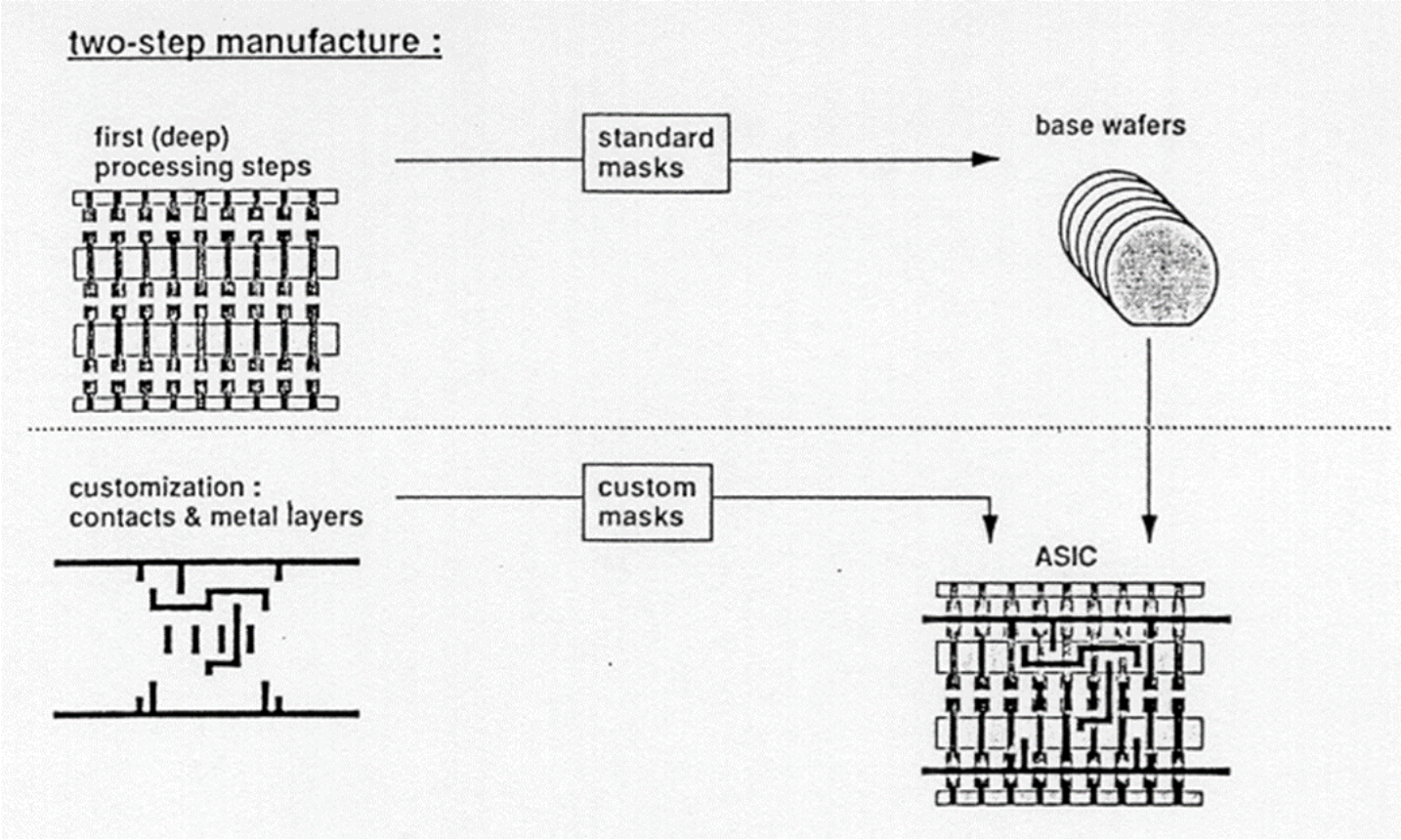


Courtesy: Mantra VLSI



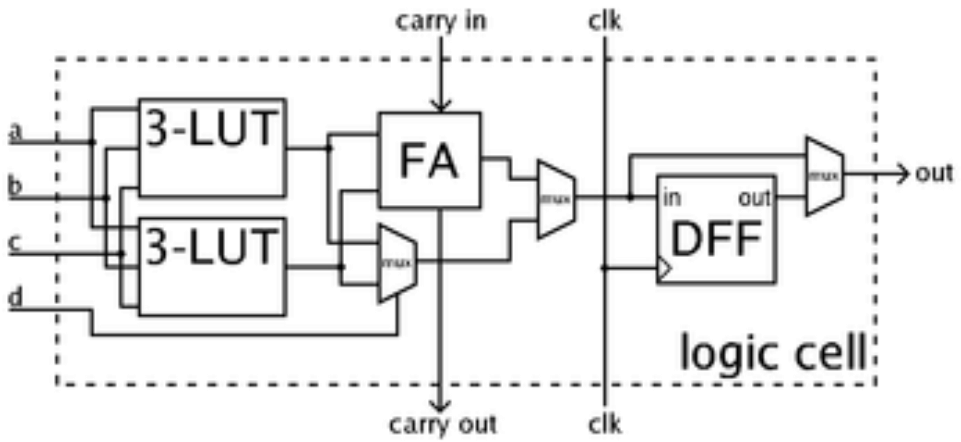
VLSI Design Styles – Gate Array

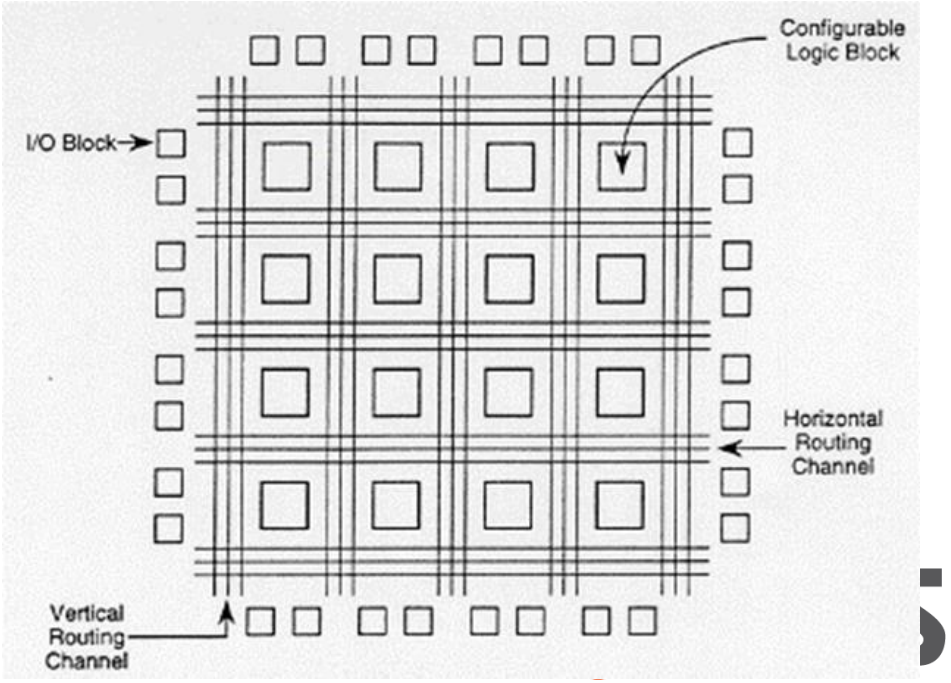
- **Gate Array Design or Structured ASIC**
  - Design implementation is done with metal mask design and processing



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





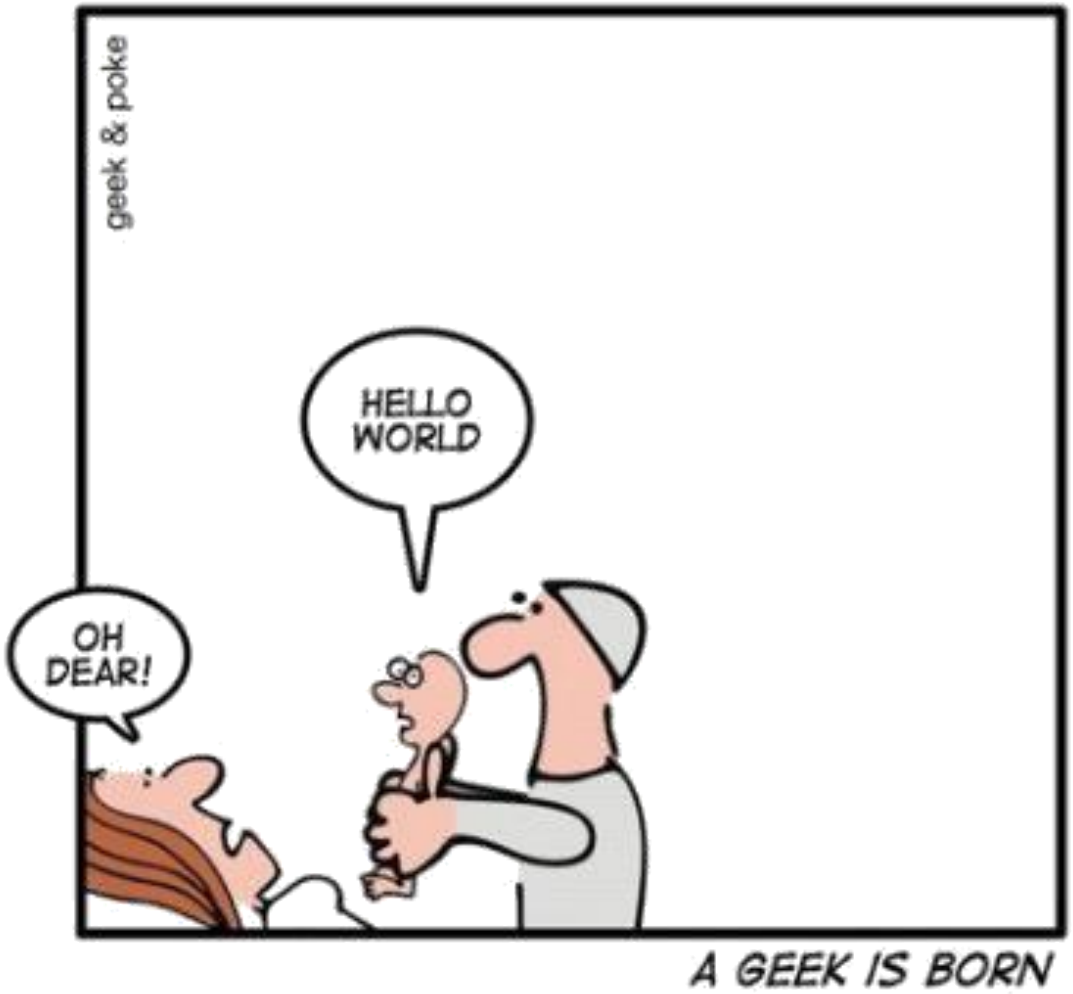
http://lsiwww.epfl.ch/LSI2001/teaching/webcourse

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.
- **Gate Array:**
  - Reduced number of masks → Inexpensive.
  - High percentage of overhead, limited speed.
- **FPGA:**
  - Post silicon configurability, very inexpensive.
  - High percentage of overhead. High cost per chip.

How a Chip is Born

How a chip is born…



A&A and Specification

↓

Logic Design & IP Integration

↓

Logic Verification

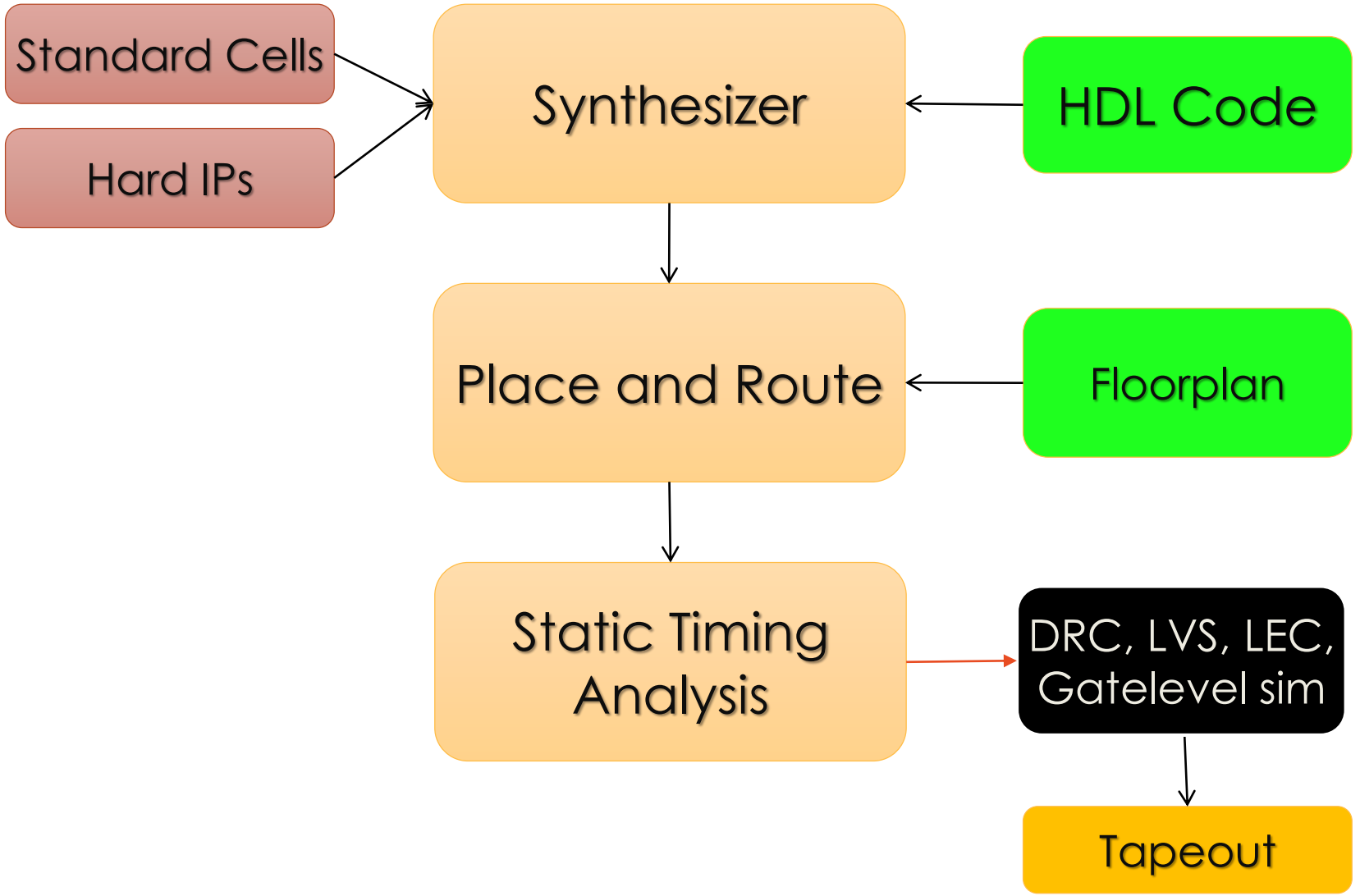
↓

Physical Design

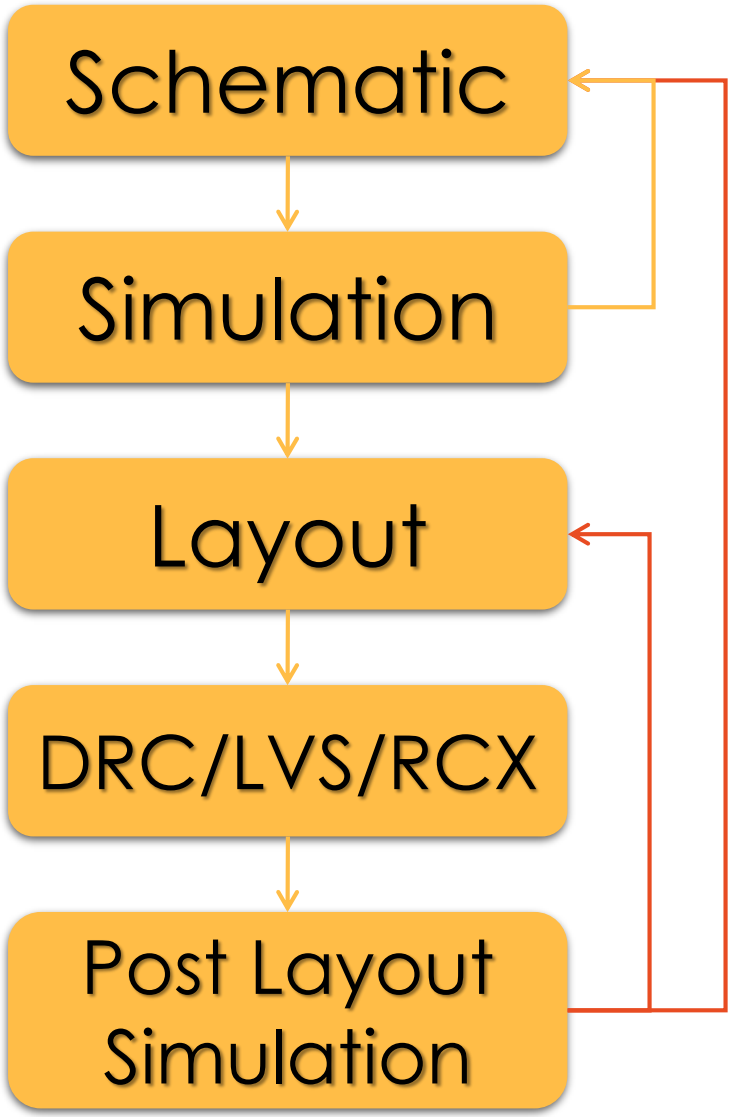
↓

Sign-off & Tape Out

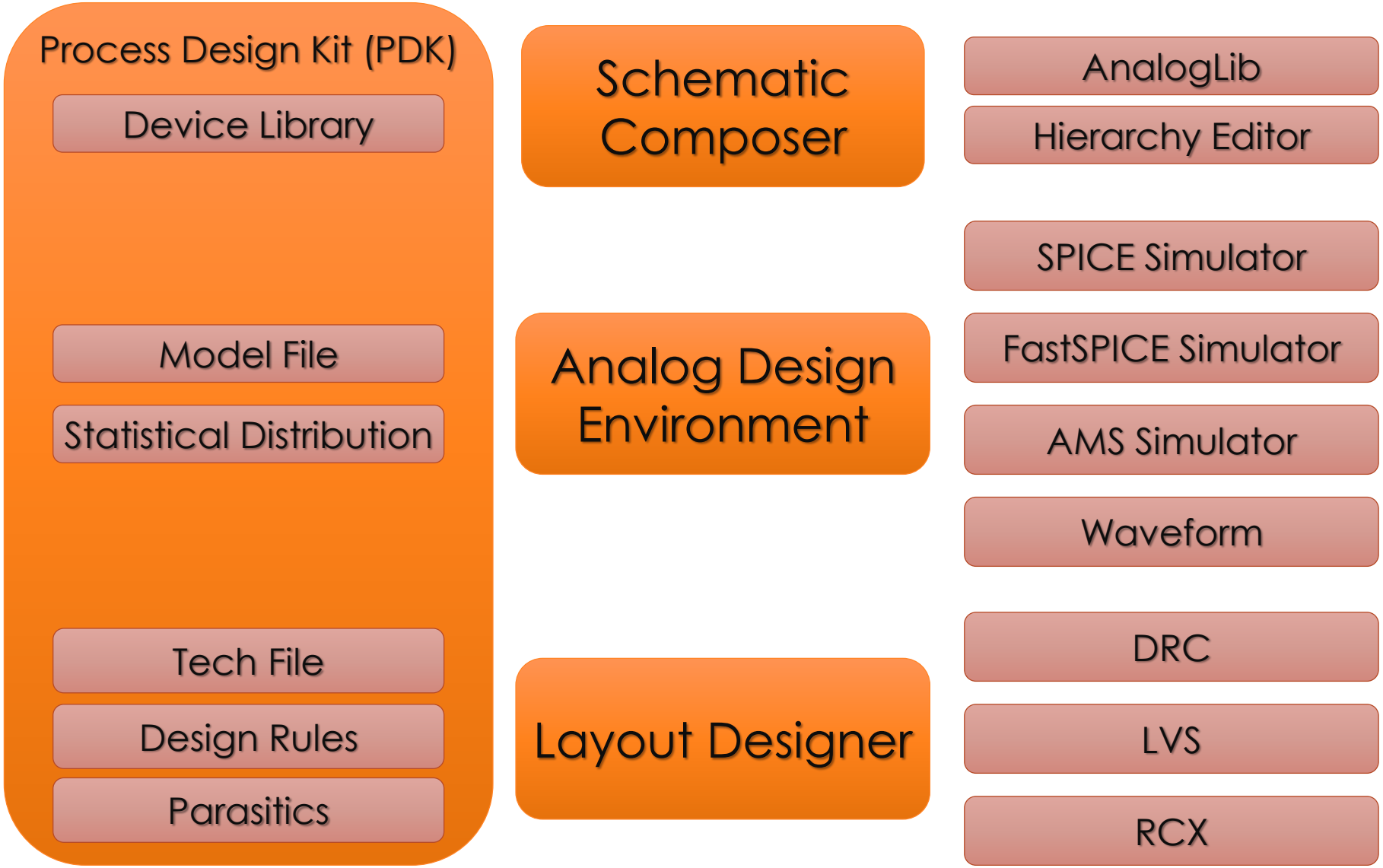
Physical Design Flow



Circuit (Custom) Design Flow

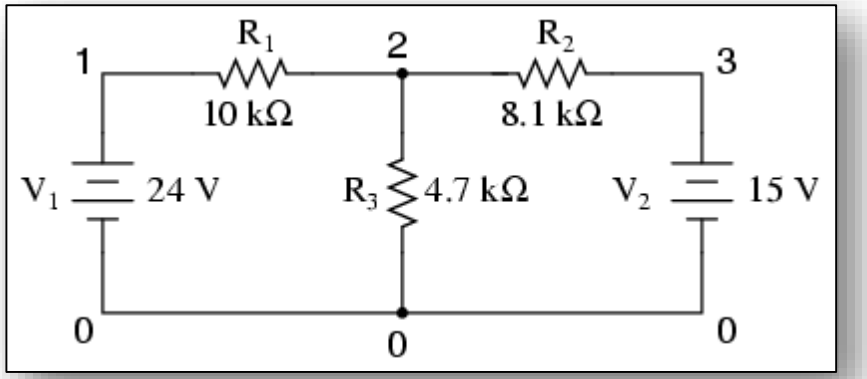


A Little Bit About Transistor Level CAD



Schematic Composer

- **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
```

Analog Design Environment

- **Graphical tool for creating testbenches (ADE, ADE-XL).**
  - 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.**

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

- **Define “variables” to enable parametric sweeps.**
- **Directly translates into a spice netlist or script (“Ocean script”) that runs the simulations and displays the outputs.**

Analog Design Environment

- **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 – Connectivity to Schematics

- **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, Layout-XL uses connectivity between the layout and schematic:**
  - Generate From Source creates “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/RCX

- **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 Assura/PVS
  - Mentor Graphics Calibre

DRC

- **DRC can be applied interactively in Layout-XL by turning on “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

- **RCX is a parasitic extractor that 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 RCX, you must FIRST run LVS.**

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

CAD Tools Used in this Class

- **Cadence Virtuoso 6.1.6 (IC and MMSIM packages)**
  - Virtuoso Schematic Editor
  - Virtuoso Analog Design Environment
  - Spectre Circuit Simulator
  - Virtuoso Visualization and Analysis (Viva)
  - Virtuoso AMS Designer
  - Virtuoso Layout Suite
  - Virtuoso Liberate Characterization Solution
- **Mentor Graphics Calibre**
  - Calibre nmDRC
  - Calibre nmLVS
  - Calibre XRC
- **TSMC 65nm Low Power PDK**
- **Arizona State University 7nm ASAP PTM**

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!