Digital VLSI Design

Lecture 1: Introduction

Semester A, 2016-17

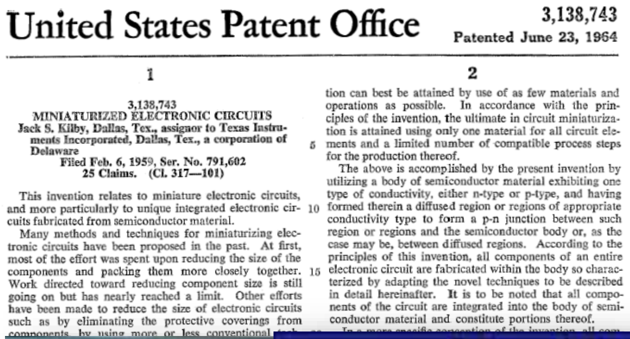
Lecturer: Dr. Adam Teman

6 November 2016

Emerging Nanoscaled Integrated Circuits and Systems Labs

Bar-Ilan University
אוניברסיטת בר-אילן

Motivation



United States Patent Office

3,138,743
Patented June 23, 1964

1

3,138,743
MINIATURIZED ELECTRONIC CIRCUITS
Jack S. Kilby, Dallas, Tex., assignor to Texas Instruments Incorporated, Dallas, Tex., a corporation of Delaware
Filed Feb. 6, 1959, Ser. No. 791,602
25 Claims. (Cl. 317—101)

This invention relates to miniature electronic circuits, and more particularly to unique integrated electronic circuits fabricated from semiconductor material.

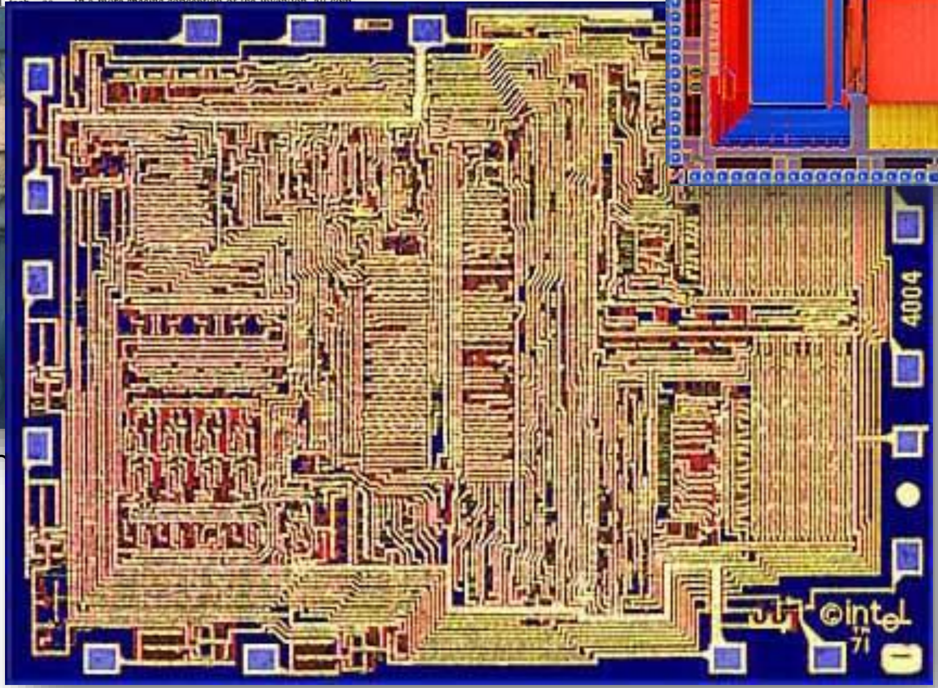
Many methods and techniques for miniaturizing electronic circuits have been proposed in the past. At first, most of the effort was spent upon reducing the size of the components and packing them more closely together. Work directed toward reducing component size is still going on but has nearly reached a limit. Other efforts have been made to reduce the size of electronic circuits such as by eliminating the protective coverings from components, by using more or less conventional

2

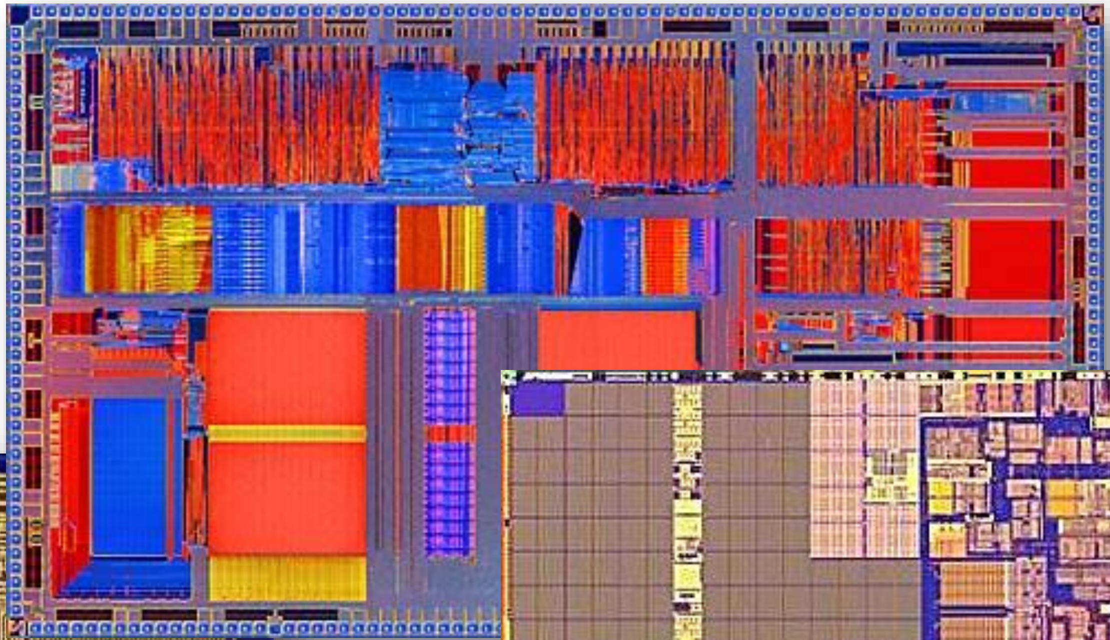
tion can best be attained by use of as few materials and operations as possible. In accordance with the principles of the invention, the ultimate in circuit miniaturization is attained using only one material for all circuit elements and a limited number of compatible process steps for the production thereof.

The above is accomplished by the present invention by utilizing a body of semiconductor material exhibiting one type of conductivity, either n-type or p-type, and having formed therein a diffused region or regions of appropriate conductivity type to form a p-n junction between such region or regions and the semiconductor body or, as the case may be, between diffused regions. According to the principles of this invention, all components of an entire electronic circuit are fabricated within the body so characterized by adapting the novel techniques to be described in detail hereinafter. It is to be noted that all components of the circuit are integrated into the body of semiconductor material and constitute portions thereof.

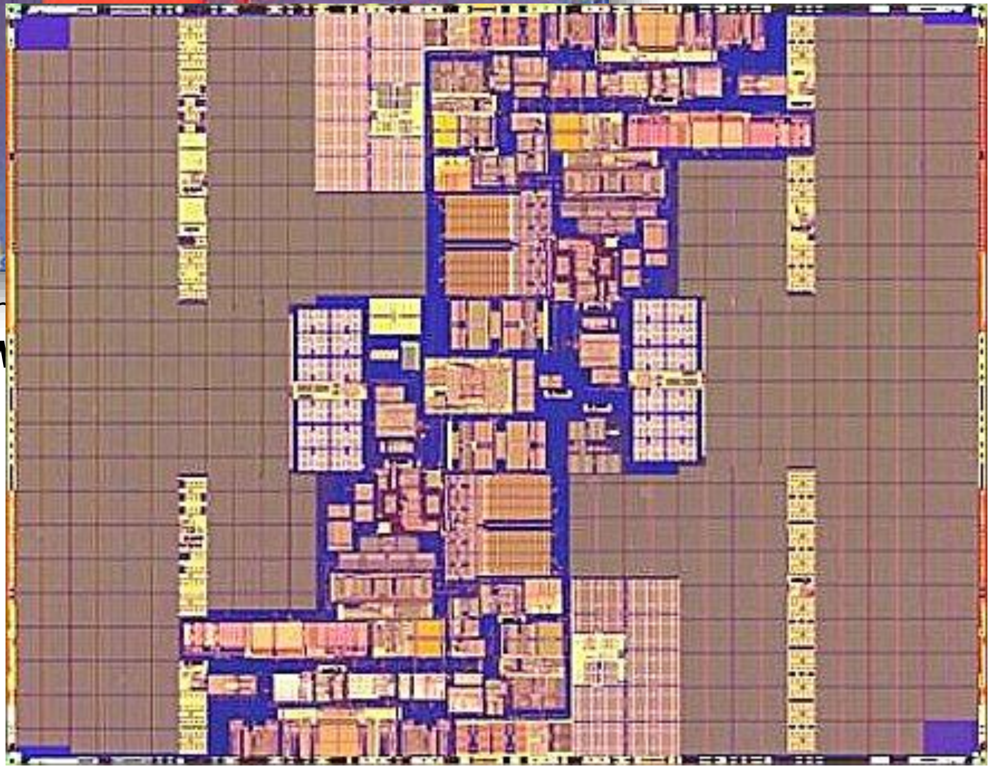
1964 – The In



1971 – The Intel 4004
2,300 Transistors



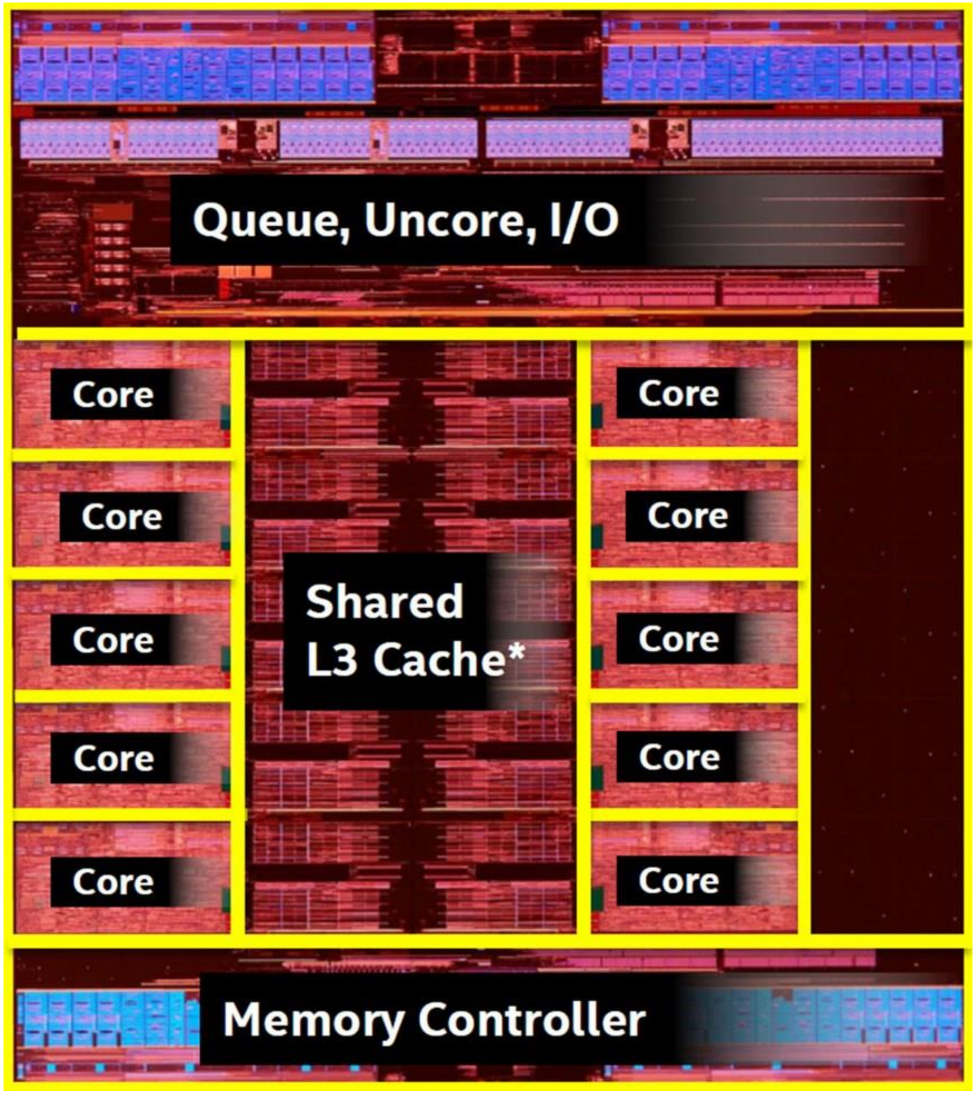
1992 – Th
1.2**M**

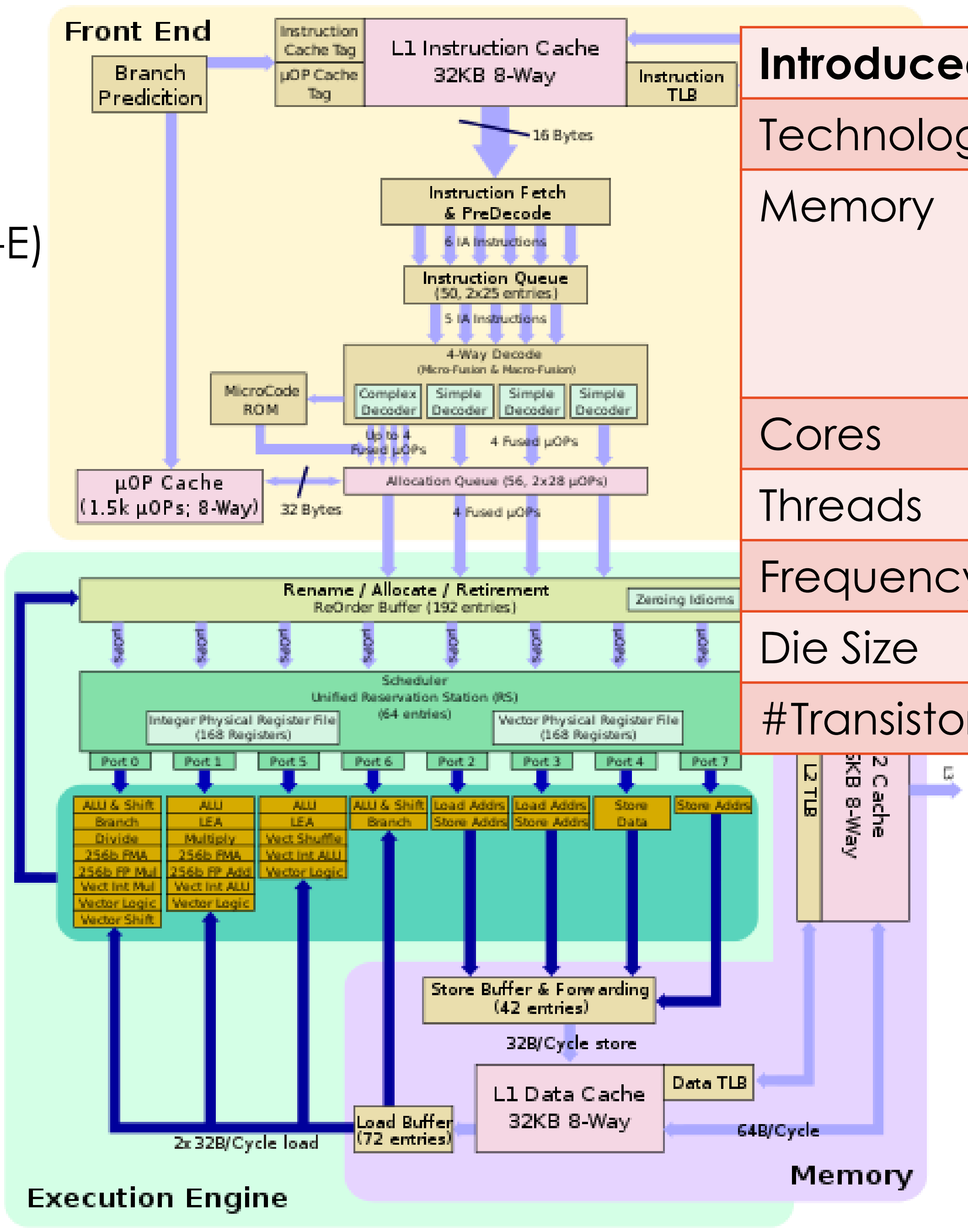


2006 – Itanium 2 "Montecito"
1.7**B** Transistors

Motivation

Core i7-6950X Extreme Edition (Broadwell-E)



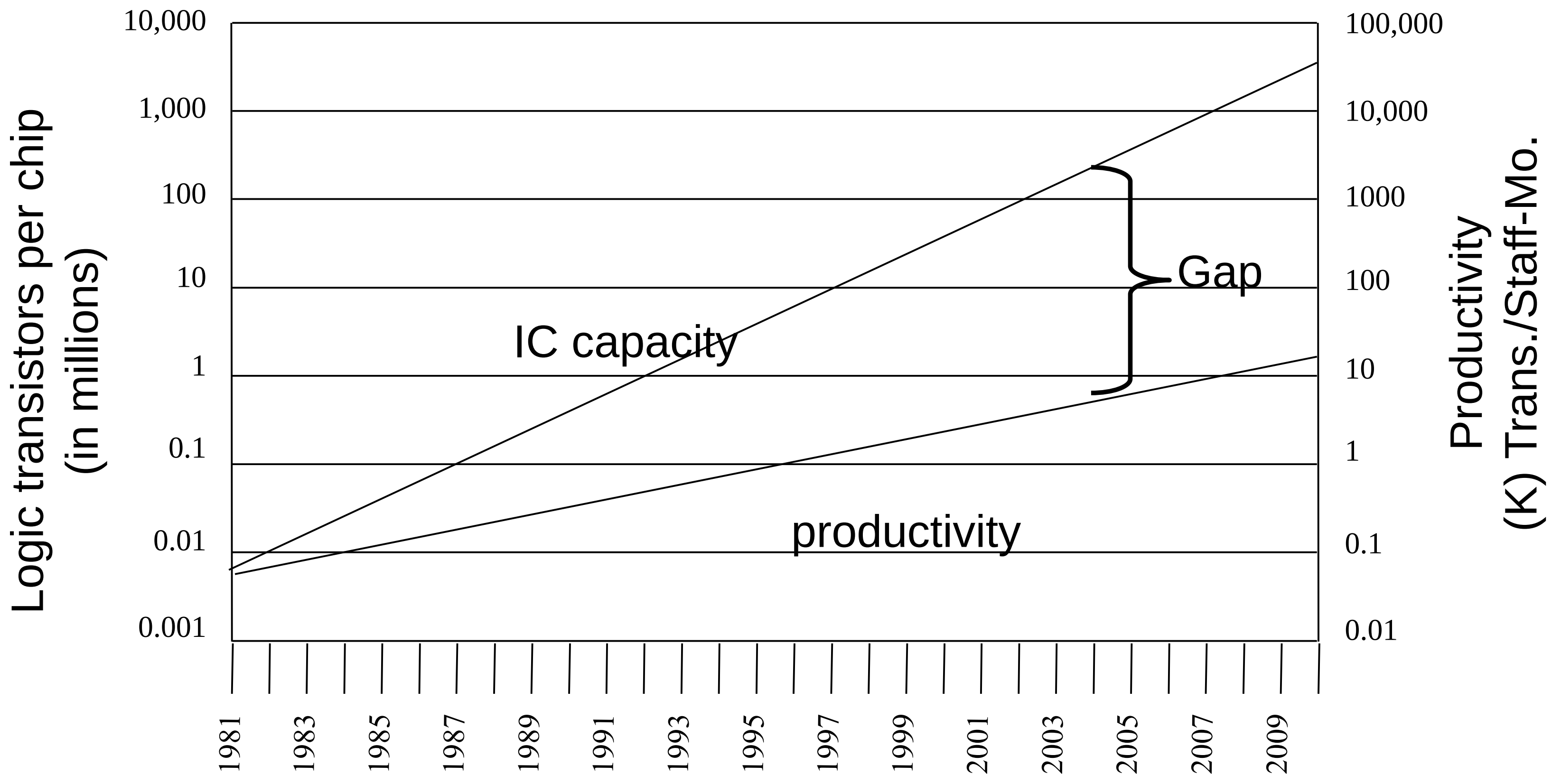


Introduced	May 2016
Technology	14nm FinFET
Memory	320KB L1-I\$ 320KB L1-D\$ 2.56MB L2\$ 25MB L3\$
Cores	10
Threads	20
Frequency	3.0-3.5 GHz
Die Size	246 mm^2
#Transistors	3.2 B

http://en.wikichip.org/wiki/intel/core_i7ee/i7-6950x

Motivation

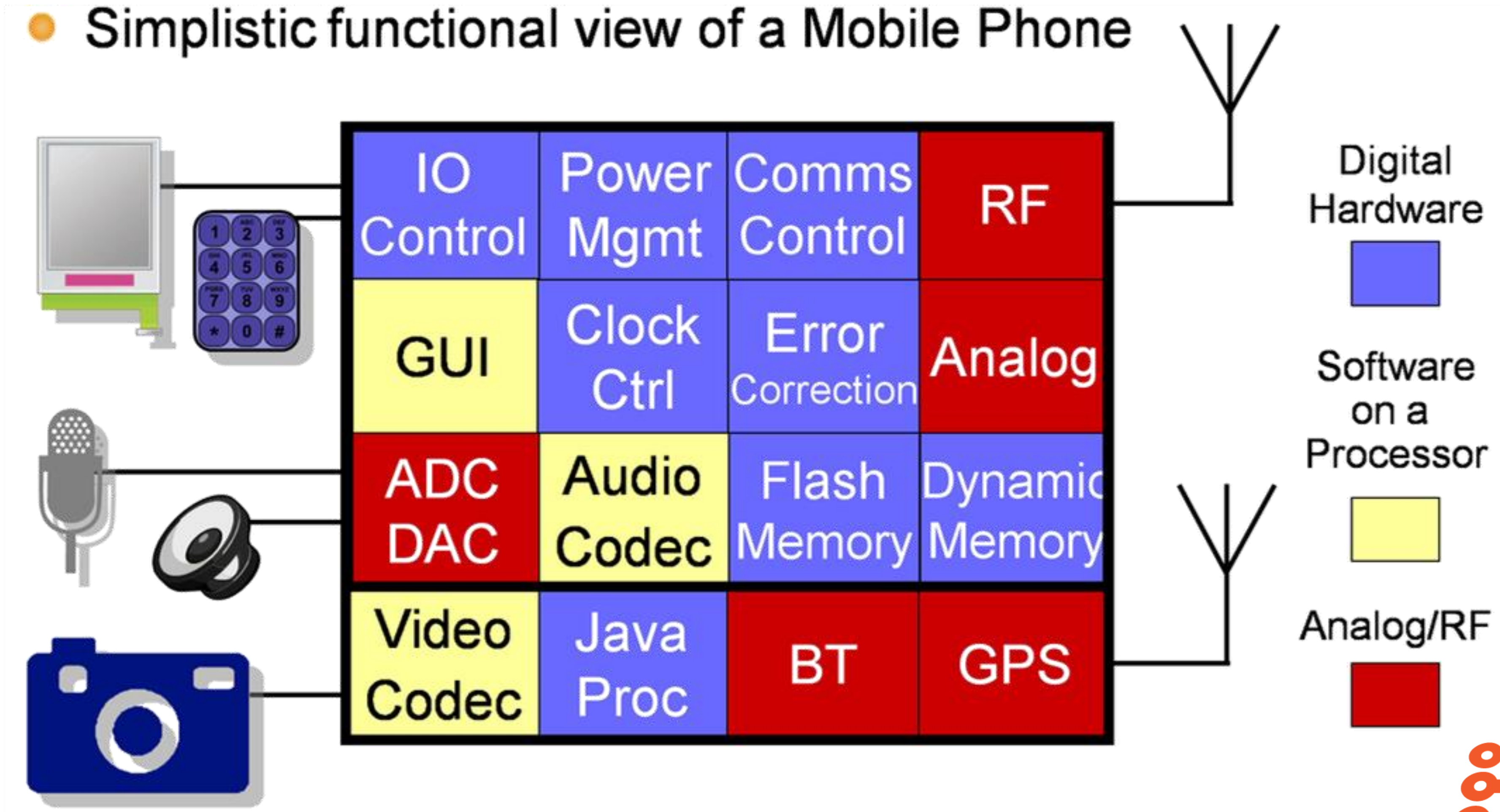
- **Houston, we have a problem…**



"Moore's Law of Engineers"

Motivation

- How on earth do we design such a thing???



The Solution:

Design Abstraction

Design Automation

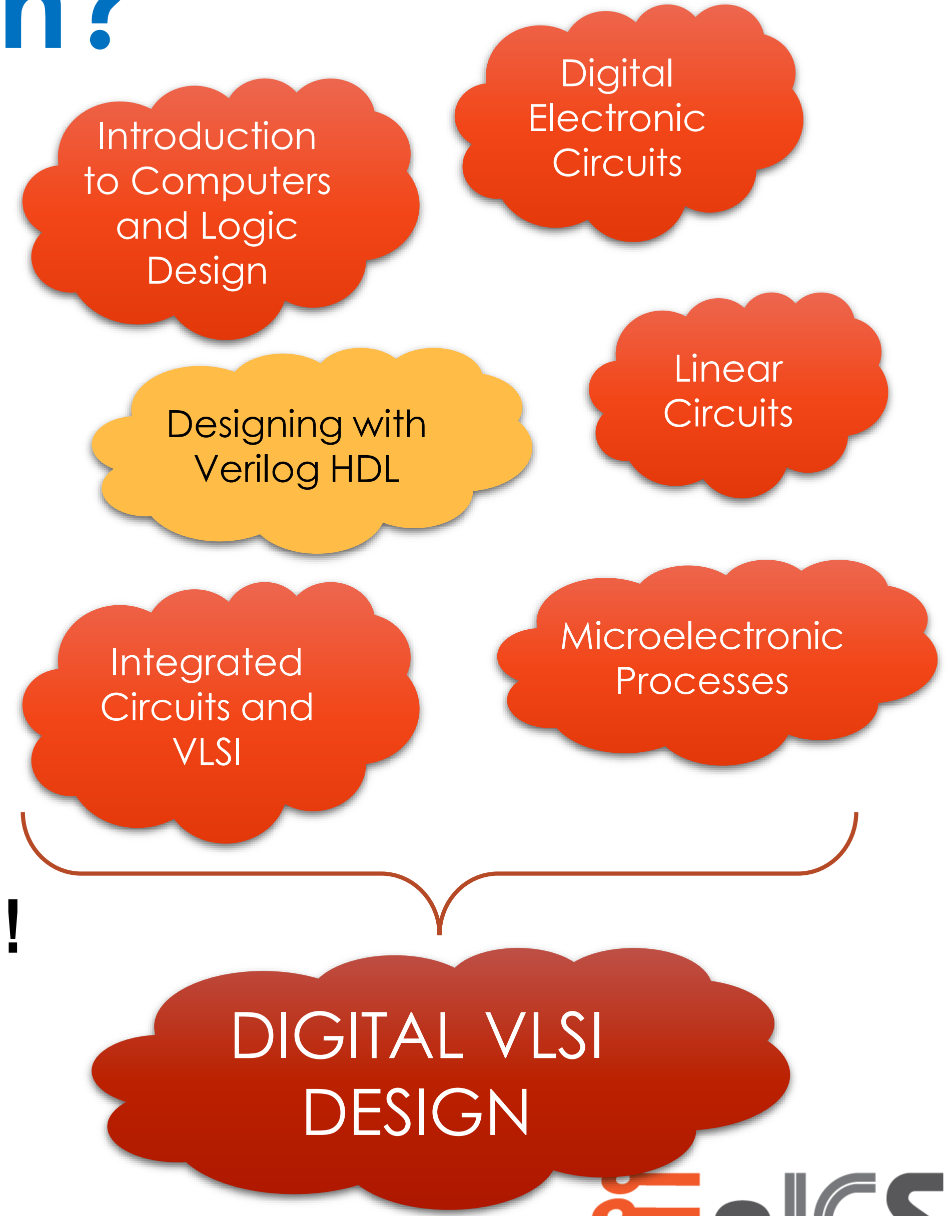
Design Re-use (IP)

Where does this course fit in?

- **In this course, we will learn how to design a chip.**
- **You have already learned:**
  - Build a general purpose micro-processor.
  - How to design a digital gate, analog circuit, and custom-designed component.
  - How to fabricate a chip.

- **Now, we will learn how to fit all of these on a chip**

…and make sure it works!


Introduction to Computers and Logic Design
Digital Electronic Circuits
Designing with Verilog HDL
Linear Circuits
Integrated Circuits and VLSI
Microelectronic Processes
DIGITAL VLSI DESIGN

1 Motivation
2 Course Logistics
3 Building a Chip
4 Design Automation
5 Chip Design Flow

Course Logistics

Course Info

- **Course number:** 83-612, Semester A, 2016-17
  - Hebrew name: מעגלי ומערכות וי.אל.אס.איי. דיגיטליים
  - More appropriate name: Digital VLSI Design
  - 1.5 Credits (2 hrs. Lecture + 1 hr. Discussion/Exercise)
- **Lectures:** Tuesday 09:00-11:00 – Auditorium 2
  - Actual times: 09:05-10:35
  - Lecture will be 1.5 hours with no break
- **Discussion/Exercise:** Tuesday 11:00-12:00 – Room 42
  - Actual times: 11:00-11:45+
- **Presentations and Discussion Handouts will be available before the lecture.**
  - You are advised to print out the handouts!
- **Video will hopefully be available *before* the lecture…**

Course Administration

- **Lecturer:**
  - Dr. Adam Teman – but call me Adi.
  - Office Hours:
    - Faculty of Engineering, 4th Floor, Office 413
    - Tuesday 1300-1400, but please make appointment by email
    - Contact: adam.teman@biu.ac.il
- **Teaching Assistant:**
  - Or Maltabashi
  - Office Hours:
    - EnICS Labs, Floor -1, Room 62 – **by appointment!**
    - Contact: ormaltab@gmail.com
- **USE THE FORUM!!!**
- ***Google* and *Cadence Support* are your best friends!**

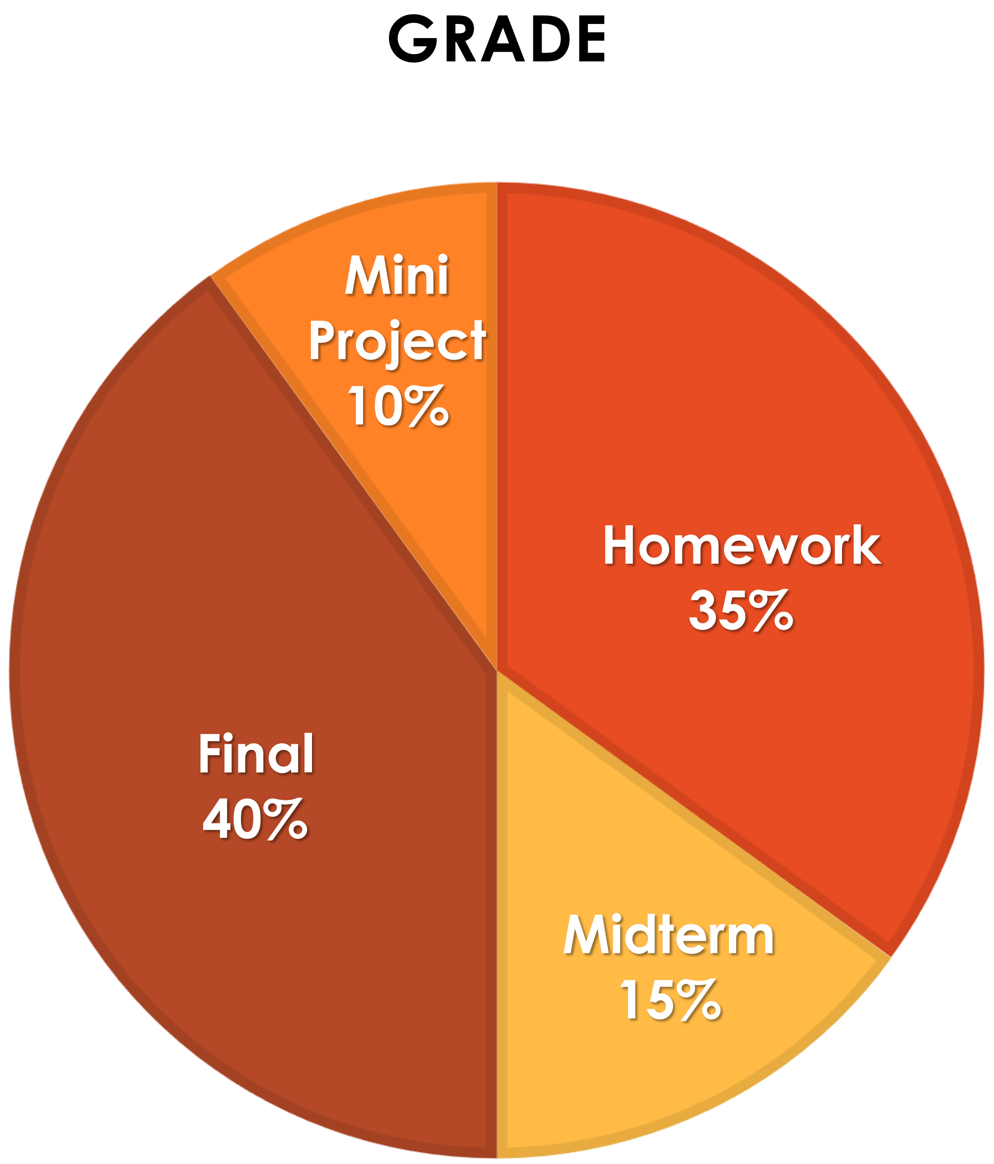
If you have not yet opened a Cadence Support account, now is the time!

Homework/Projects

- **Homework:**
  - Almost every week, you will receive a homework assignment.
  - Deadline for homework is (usually) 2 weeks from publication.
    - **Extensions** will be considered ***IFF*** you ask *at least* 48 hours before the deadline!
  - Homework is done *individually*!
    - Helping each other is encouraged.
    - **Grading** will be according to the *order* of submission!
- **Mini-project:**
  - Each pair of students will suggest a subject for a project towards the end of the semester.
  - Adi will authorize the project subject.
  - Bonus for preparing in LaTeX and/or making a video!

Grading

- **Homework: 35%**
- **Midterm: 15%**
- **Final: 40%**
  - Both the midterm and final will include *many* short questions, often multiple-choice, covering *everything* we discussed in class.
- **Mini-project: 10%**
  - Bonus for preparing in LaTeX!



Syllabus (*not Final*)

- Week 1: Introduction
- Week 2: Verilog
- Week 3: Static Timing Analysis
- Week 4-5: Logic Synthesis
- Week 6: Moving to the Physical Domain
- Week 7: Placement
- Week 8: Clock Tree Synthesis
- Week 9: Routing
- Week 10: Design for Test
- Week 11: Verification

References

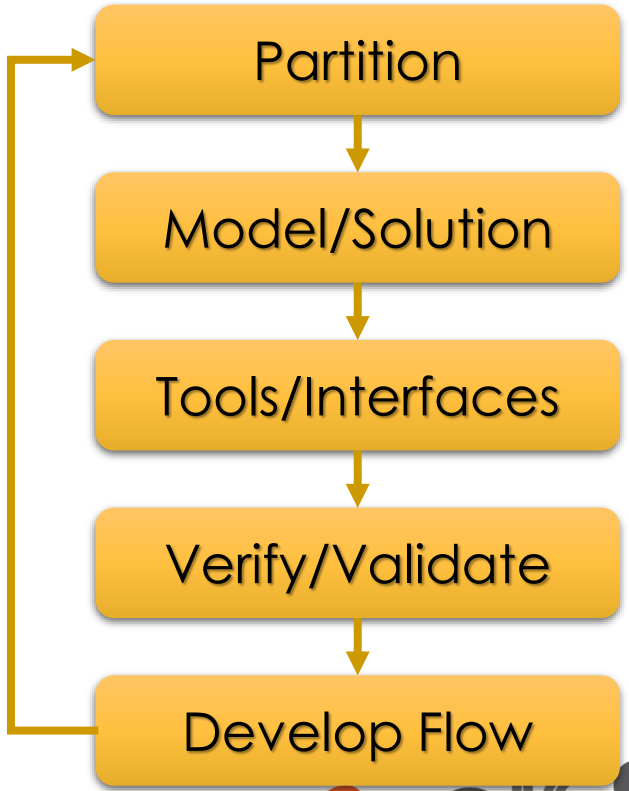
- **Way too many to state all, and hopefully many are cited on the slides themselves, but here are a few:**
  - Rob Rutenbaur – “From Logic to Layout” (available on Coursera)
  - Nir Sever – Low Power Design (BGU)
  - Roy Shor – תכן לוגי (BGU)
  - IDESA Digital Design Course
  - Rabaey “Digital Integrated Circuits” 2nd Edition
  - Weste, Harris “CMOS VLSI Design”
  - Google (oh, thank you Google!)
  - Cadence Support (support.cadence.com)
  - Synopsys SolveNet (solvenet.synopsys.com)

Building a Chip

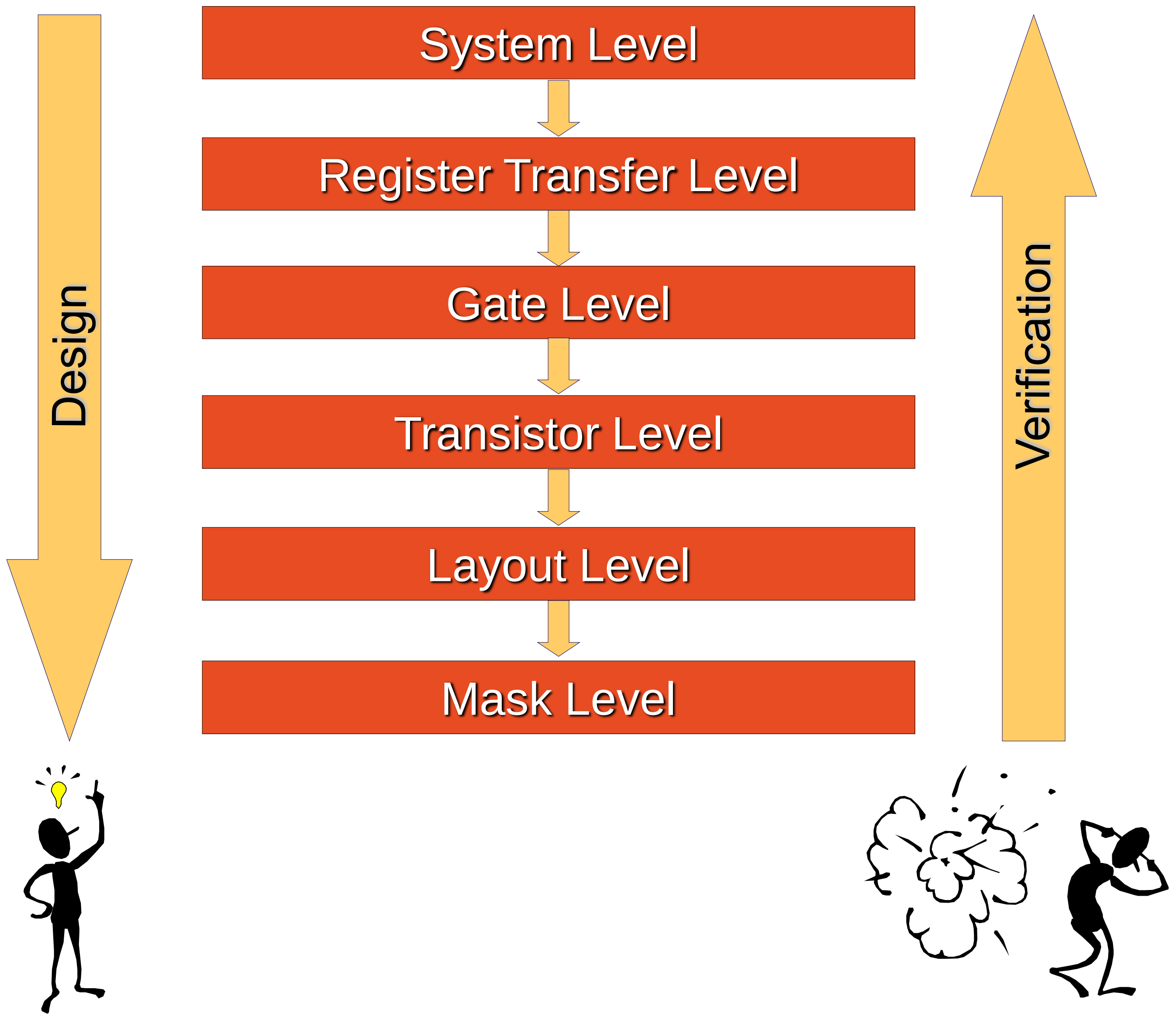
General Design Approach

- **How do engineers build a bridge?**

- **Divide and conquer !!!!**
  - **Partition** design problem into many sub-problems, which are manageable
  - Define mathematical **model** for sub-problem and find an **algorithmic** solution
    - Beware of model limitations and check them !!!!!!!
  - Implement algorithm in **individual design tools**, define and implement general **interfaces** between the tools
  - Implement **checking tools** for boundary conditions
  - Concatenate design tools to general **design flows** which can be managed
  - See what doesn't work and **start over**.



Basic Design Abstraction



Another view:

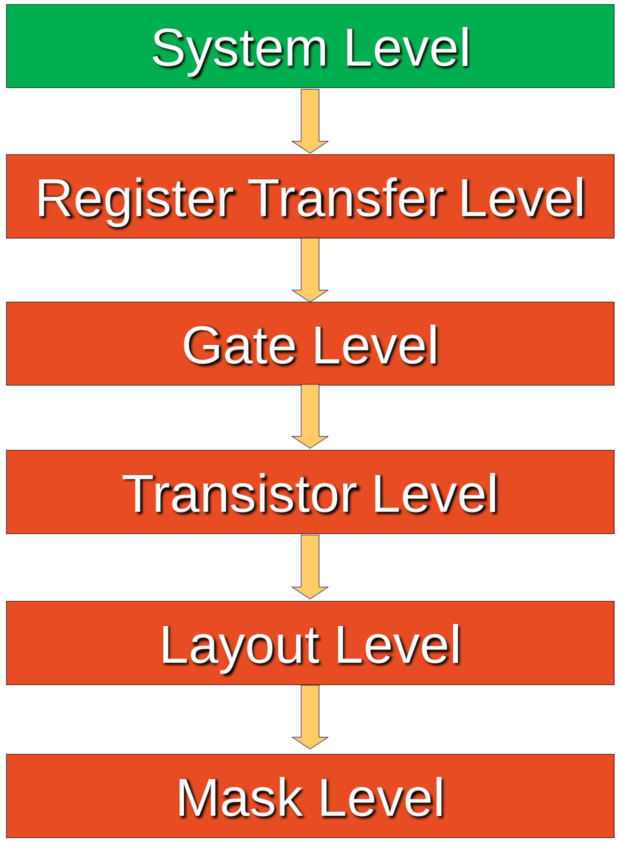
- Application
- Algorithm
- Programming Language
- OS / Virtual Machine
- Instruction Set Architecture
- Microarchitecture
- Register-Transfer Level
- Circuits
- Devices
- Physics

System Level Abstraction

- **Abstract algorithmic description of high-level behavior**
  - e.g., C-Programming language

```
Port* compute_optimal_route_for_packet
          (Packet_t *packet, Channel_t *channel)
   {
      static Queue_t *packet_queue;
      packet_queue = add_packet(packet_queue, packet);
      ...
   }
```

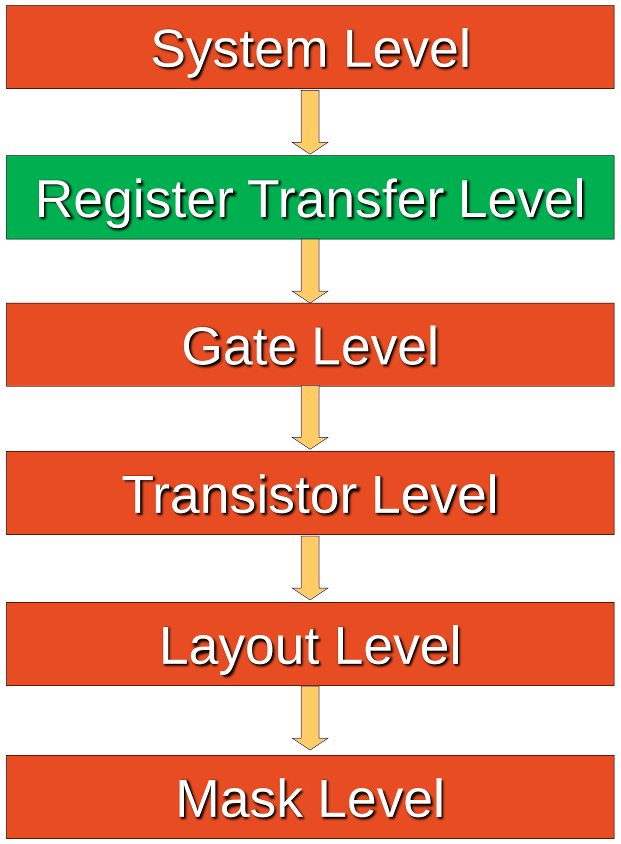
  - Abstract because it does not contain any implementation details for timing or data
  - Efficient to get a compact execution model as a first design draft
  - Difficult to maintain throughout project because no link to implementation



Register-Transfer Level (RTL)

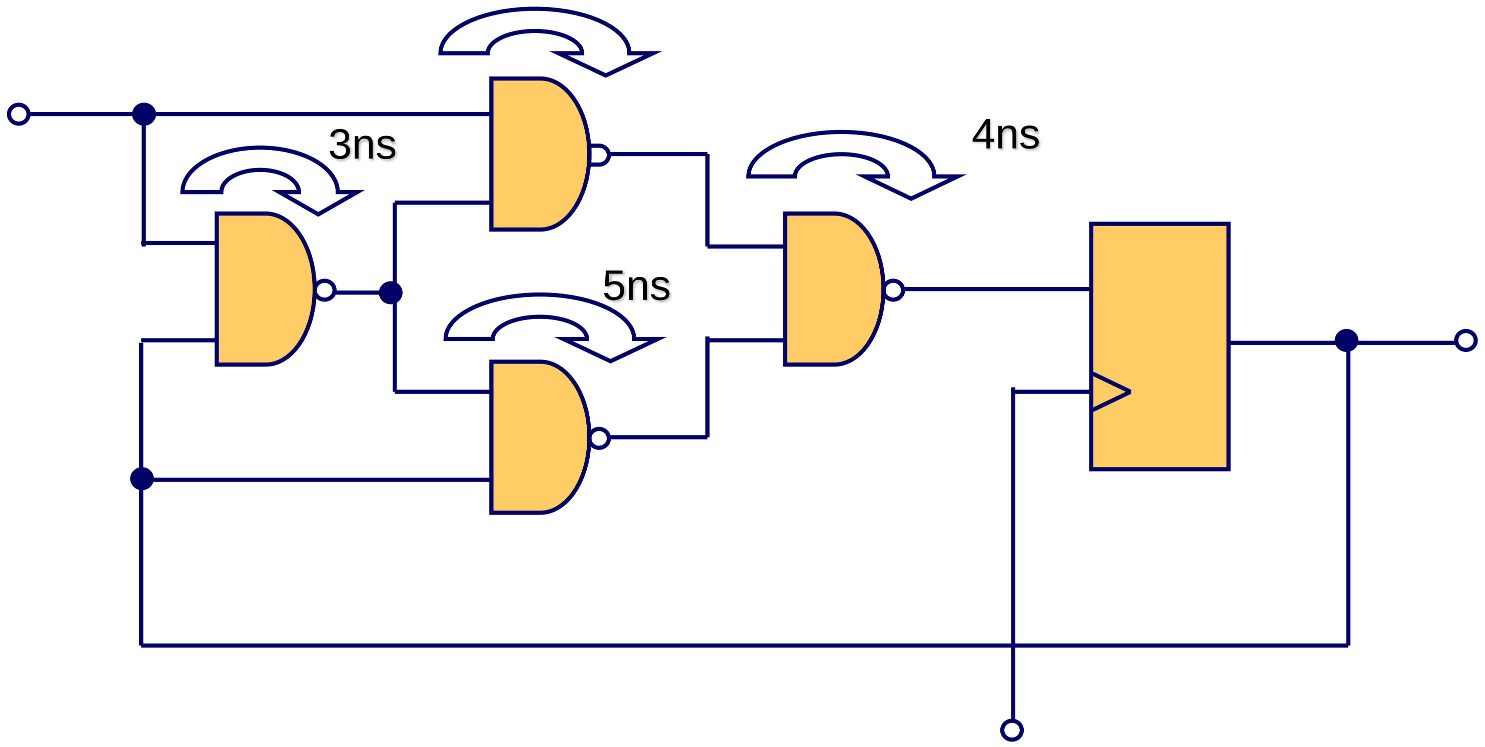
- **Cycle accurate model “close” to the hardware implementation**
  - bit-vector data types and operations as abstraction from bit-level implementation
  - sequential constructs (e.g., `if-then-else`, `while` loops) to support modeling of complex control flow

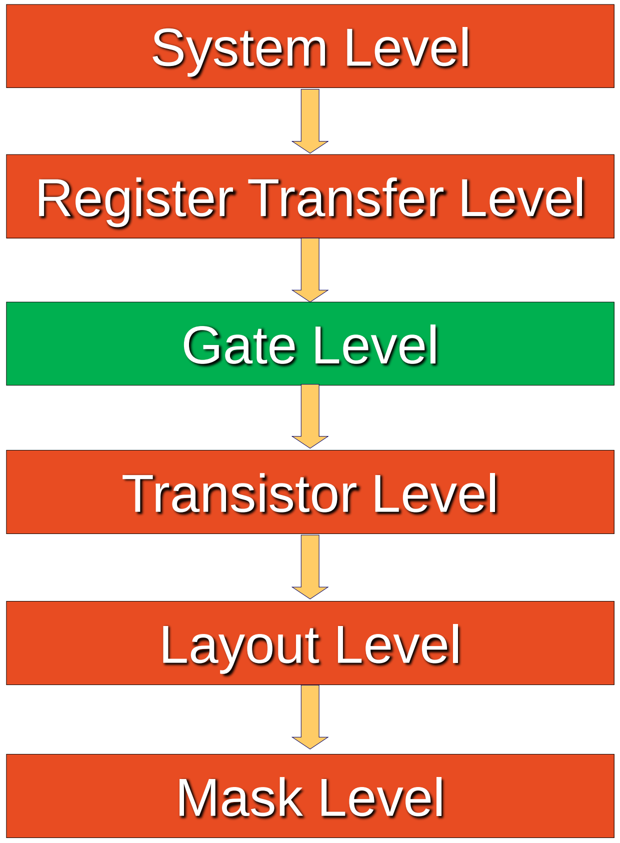
```
module mark1;
  reg [31:0] m[0:8192];
  reg [12:0] pc;
  reg [31:0] acc;
  reg[15:0] ir;

  always
   begin
     ir = m[pc];
     if(ir[15:13] == 3b’000)
        pc = m[ir[12:0]];
     else if (ir[15:13] == 3’b010)
        acc = -m[ir[12:0]];
     ...
   end
endmodule
```



Gate Level Abstraction (GTL)

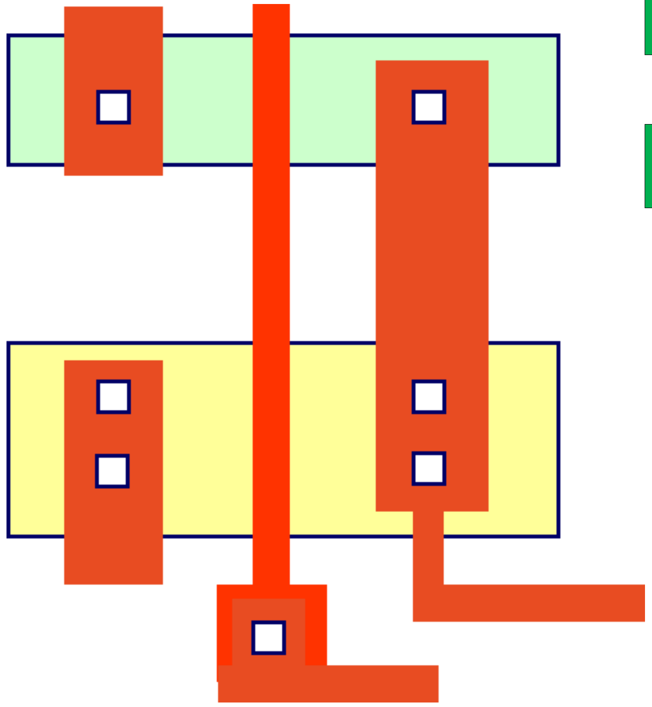
- **Model on finite-state machine level**
  - models function in Boolean logic using registers and gates
  - various delay models for gates and wires

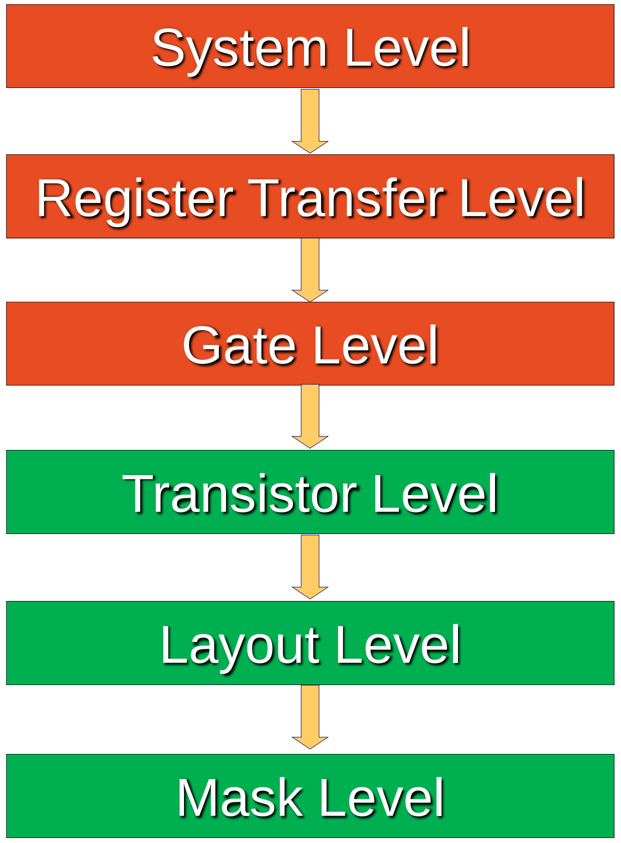




Transistor to Mask Level

- **As we've seen in previous courses:**
  - Transistor Level:
    - Use compact models to enable accurate circuit simulation.
  - Layout Level:
    - Draw polygons to implement the devices and interconnect.
  - Mask Level:
    - Create actual photo-masks for performing lithography during fabrication process.



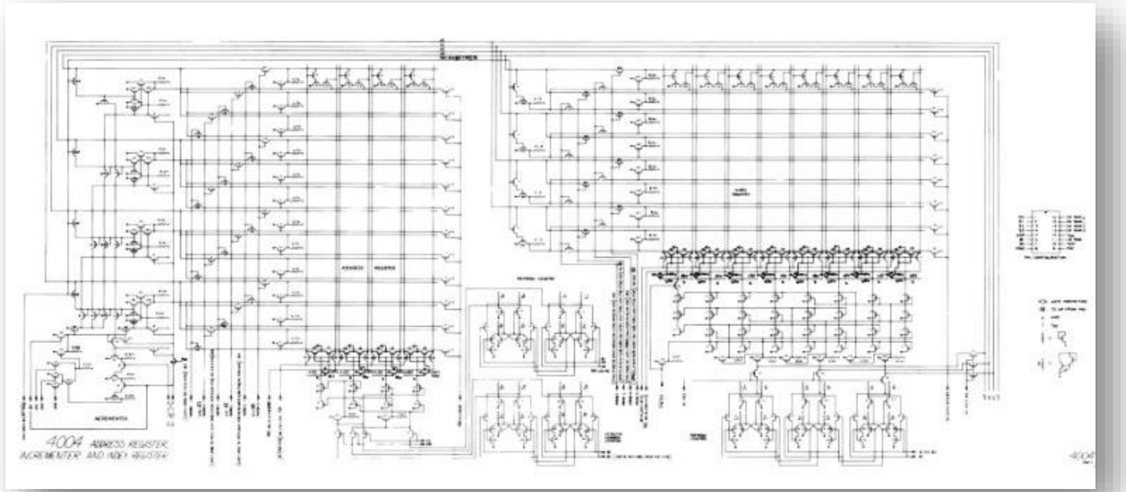


1 Motivation
2 Course Logistics
3 Building a Chip
4 Design Automation
5 Chip Design Flow

Design Automation

The (really) Olden Days

- **Early chips were prepared entirely by hand:**

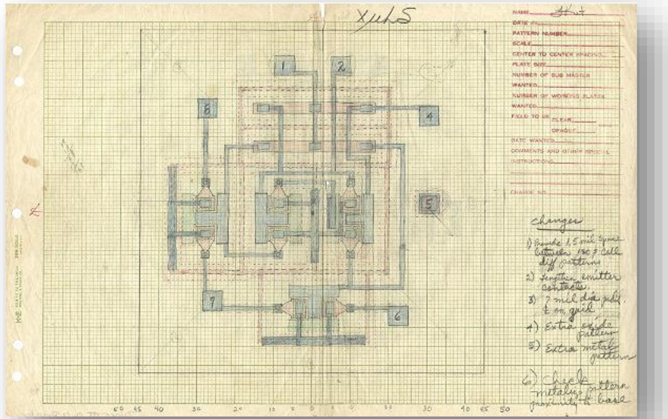

Schematic of Intel 4004 (1971)

Mainframe CAD System (1967)

http://www.computerhistory.org/revolution/digital-logic

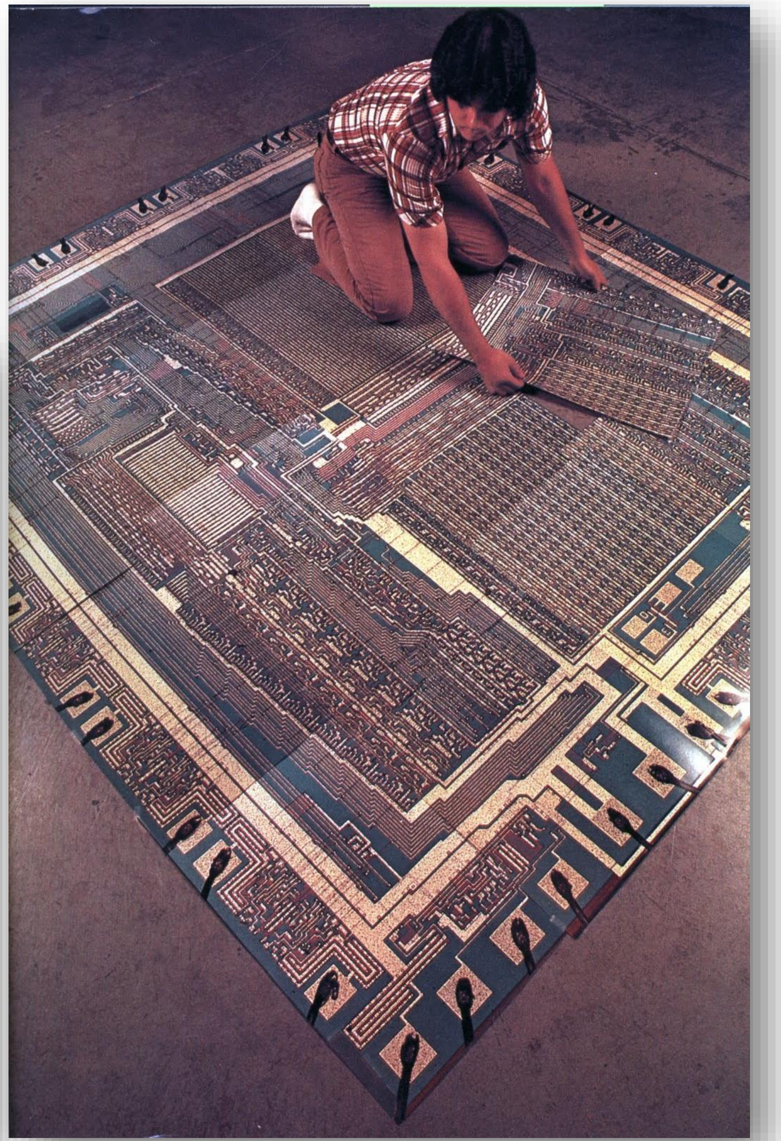
The (really) Olden Days

- **Early chips were prepared entirely by hand:**

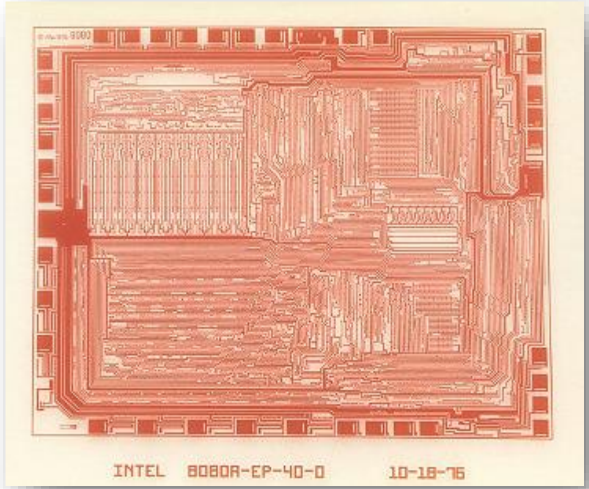


Hand drawn gate layout (Fairchild)

Rubylith Operators (1970)



The original Tape-Out?



8088A Mask Transparent Overlays (1976)

http://www.computerhistory.org/revolution/digital-logic

Design Automation Today

Design:
- High-Level Synthesis
- Logic Synthesis
- Schematic Capture
- Layout
- PCB Design

Simulation:
- Transistor Simulation
- Logic Simulation
- Hardware Emulation
- Technology CAD
- Field Solvers

Validation:
- ATPG
- BIST

Analysis and Verification:
- Functional Verification
- Clock Domain Crossing
- Formal Verification
- Equivalence Checking
- Static Timing Analysis
- Physical Verification

Mask Preparation:
- Optical Proximity Correction (OPC)
- Resolution Enhancement Techniques
- Mask Generation

EDA in this Course

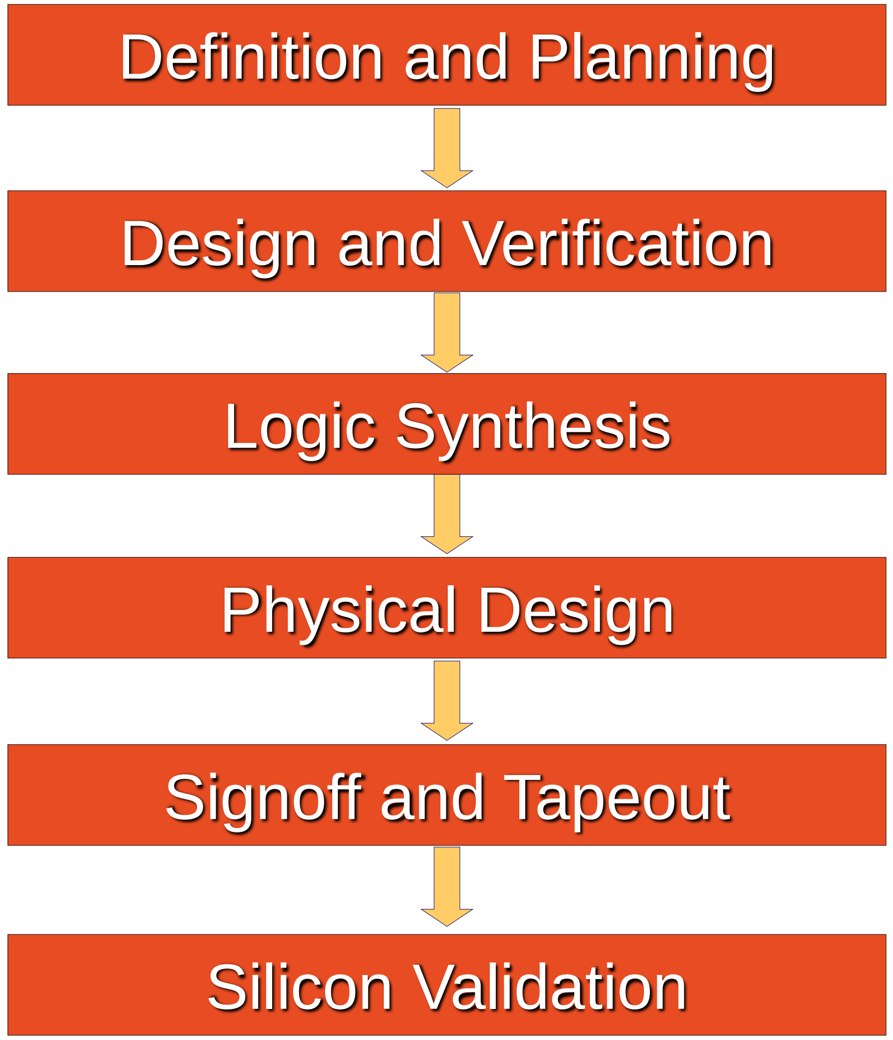
- **We will primarily use a Cadence Digital Implementation flow:**
  - RTL Compiler (Genus) – Synthesis tool
  - Encounter (Innovus) – Place and Route
    - Tempus – Static Timing Analysis
    - Voltus – Power and IR Drop
    - QRC – Parasitic Extraction
    - Ccopt – Clock Tree Synthesis
  - Incisive (irun) – Logic Simulation
  - Conformal – Logic Equivalence

Chip Design Flow

How a chip is built

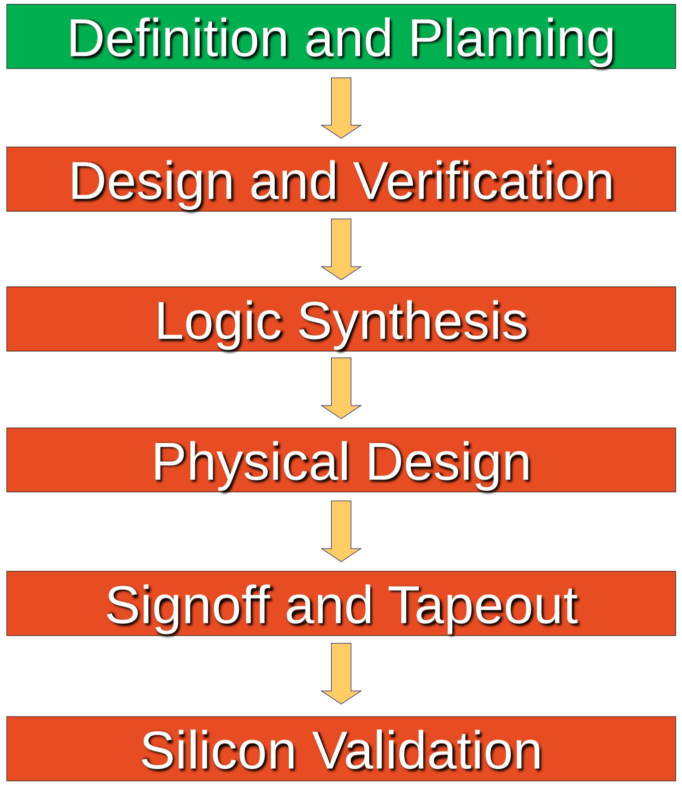
- Definition and Planning
- Design and Verification (Frontend)
- Logic Synthesis (Frontend and Backend)
- Physical Design (Backend)
- Signoff and Tapeout
- Silicon Validation

- Don’t forget package & board design, software design, test plan, etc., etc., etc.



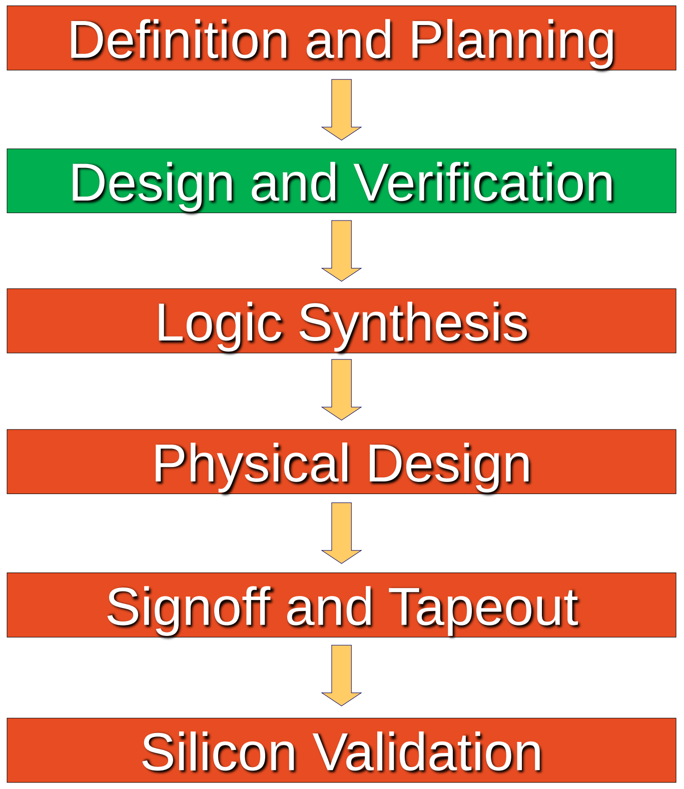
Definition & Planning

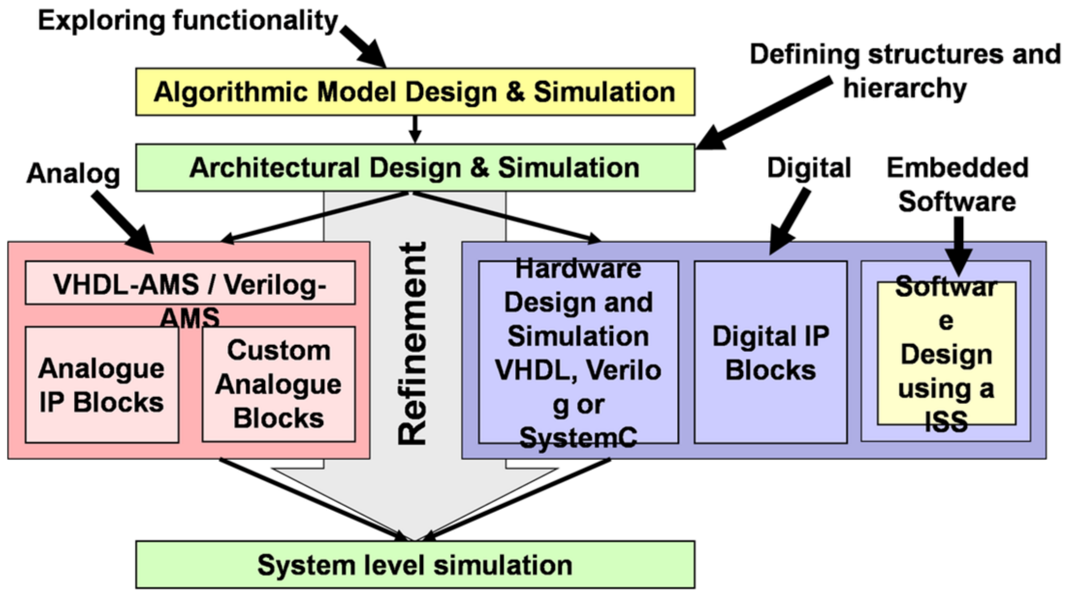
- **Marketing Requirements Document (MRD)**
- **Chip Architecture**
  - Define bus structures, connectivity
  - Partition Functionality
  - High-Level System Model (Bandwidths, Power, Freq.)
  - System partitioning (HW vs SW, #Cores, Memories…)
- **Design Documents**
- **Floorplan/Board Requirements**
- **Process and fab**
- **Project kick-off – transfer to implementation**



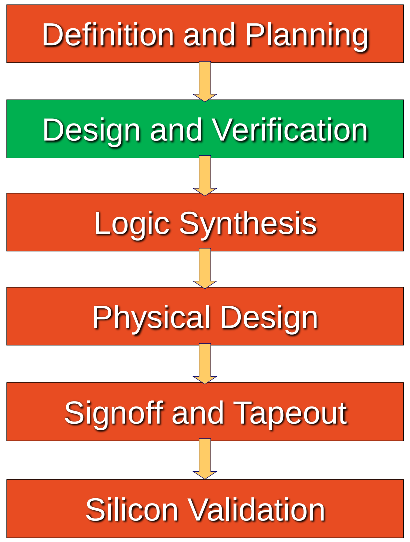
Design and Verification

- **RTL (Register Transfer Level) Design**
- **Integration/Development of IPs**
- **RTL Lint/synthesabilty checks**
- **Formal Verification**
- **Functional verification all the IPs:**
  - unit level
  - sub-system level
  - chip (SOC) level




Exploring functionality
Algorithmic Model Design & Simulation
Defining structures and hierarchy
Architectural Design & Simulation
Analog
Digital
Embedded Software
VHDL-AMS / Verilog-AMS
Analogue IP Blocks
Custom Analogue Blocks
Refinement
Hardware Design and Simulation VHDL, Verilog or SystemC
Digital IP Blocks
Software Design using a ISS
System level simulation

Design and Verification - IP Integration

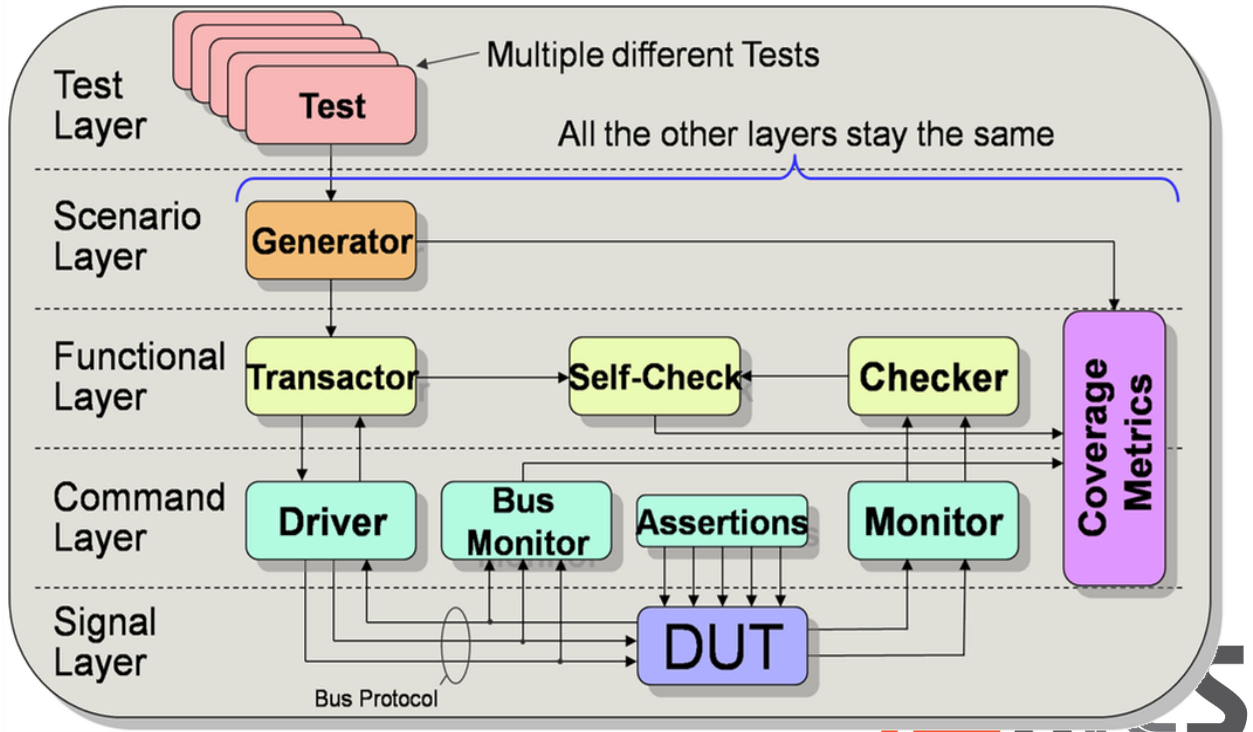


- **Hard IP**
  - IP provided as pre-existing layouts with:
    - Timing models
    - Layout abstracts
    - Behavioral models (Verilog/VHDL)
    - Sometimes with Spice models, full-layouts
  - This is the standard delivery format for custom digital blocks
    - RAMs, ROMs, PLLs, Processors
- **Soft IP**
  - RTL Code
    - Can be encrypted
    - Instantiated just like any other RTL block
  - Sometimes with behavioral models

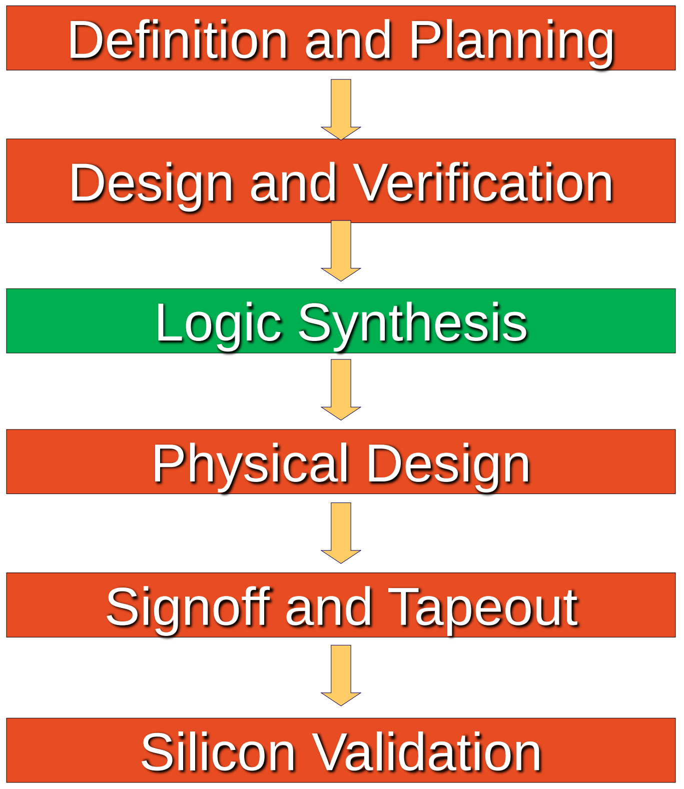
Design and Verification - Prototyping

- **Different levels of verification:**
  - Specification driven testing
  - Bug driven testing
  - Coverage driven testing
  - Regression
- **FPGA Prototyping:**
  - Synthesize to FPGA
  - Speeds up testing where possible.
- **Hardware Emulation:**
  - Big servers that can emulate the entire design.

Definition and Planning
↓
Design and Verification
↓
Logic Synthesis
↓
Physical Design
↓
Signoff and Tapeout
↓
Silicon Validation



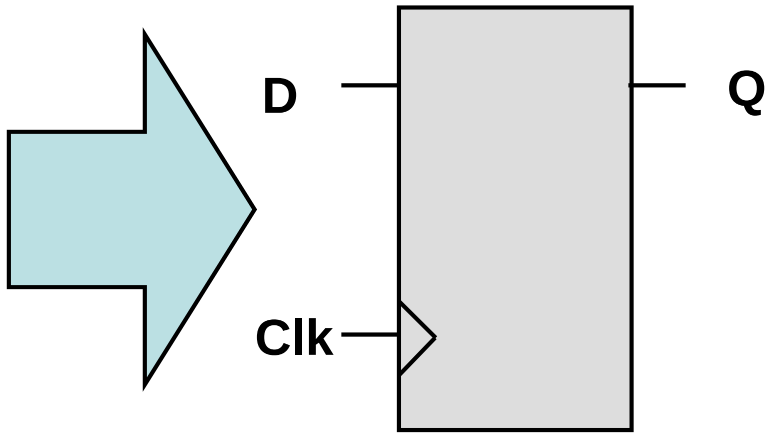
Logic Synthesis



- **Inputs:**
  - Technology library file
  - RTL files
  - Constraint files (SDC)
  - DFT definitions
- **Output:**
  - Gate-level netlist

- **Synthesis**
  - Converting RTL code into a generic logic netlist
- **Mapping**
  - Mapping generic netlist into standard cells from the core library
- **Optimization**
  - To meet Timing / Area / Power constraints

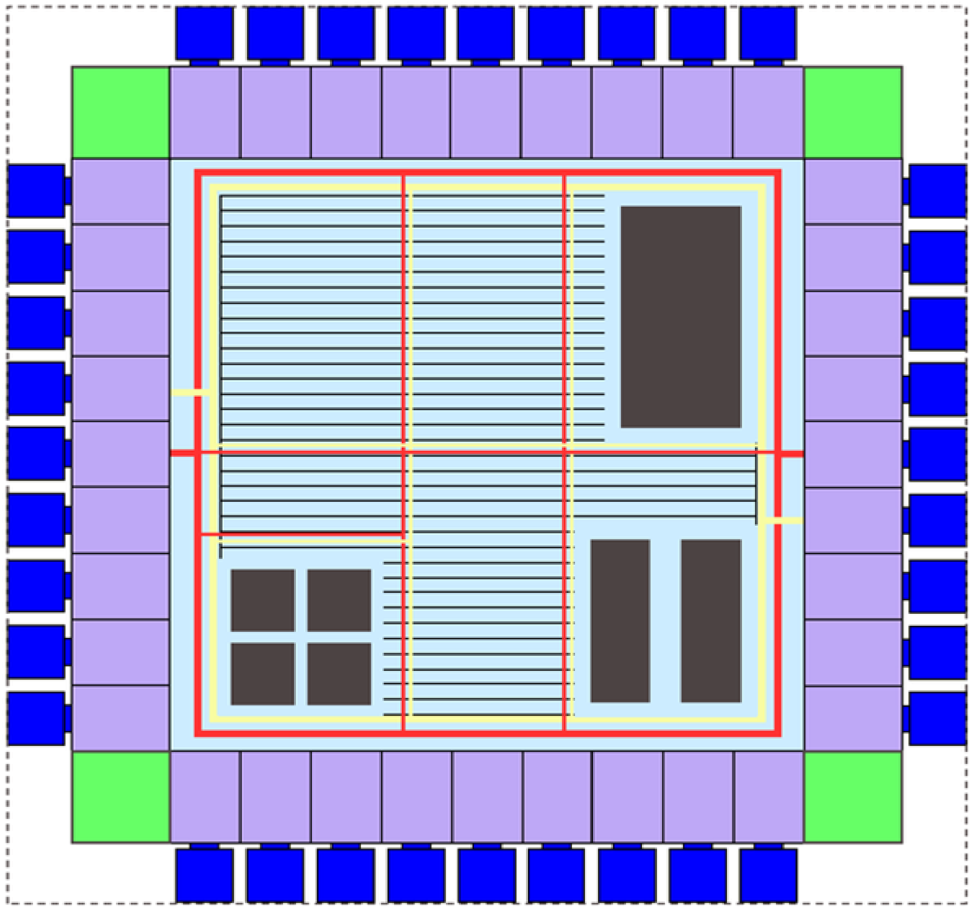
```
module DFF(Clk, D, Q);
  input Clk;
  input D;
  output Q;
  always @(posedge Clk)
    Q <= D;
endmodule
```

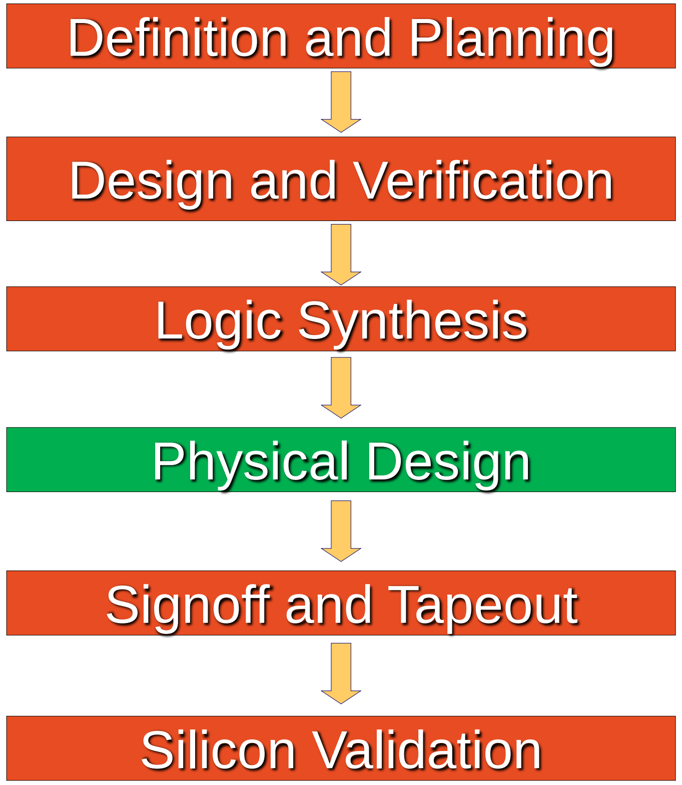


- **Post Synthesis checks**
  - Gate-level simulation
  - Formal verification (Logic Equivalence)
  - Static Timing Analysis (STA)
  - Power/Area estimation

Physical Design (Backend)

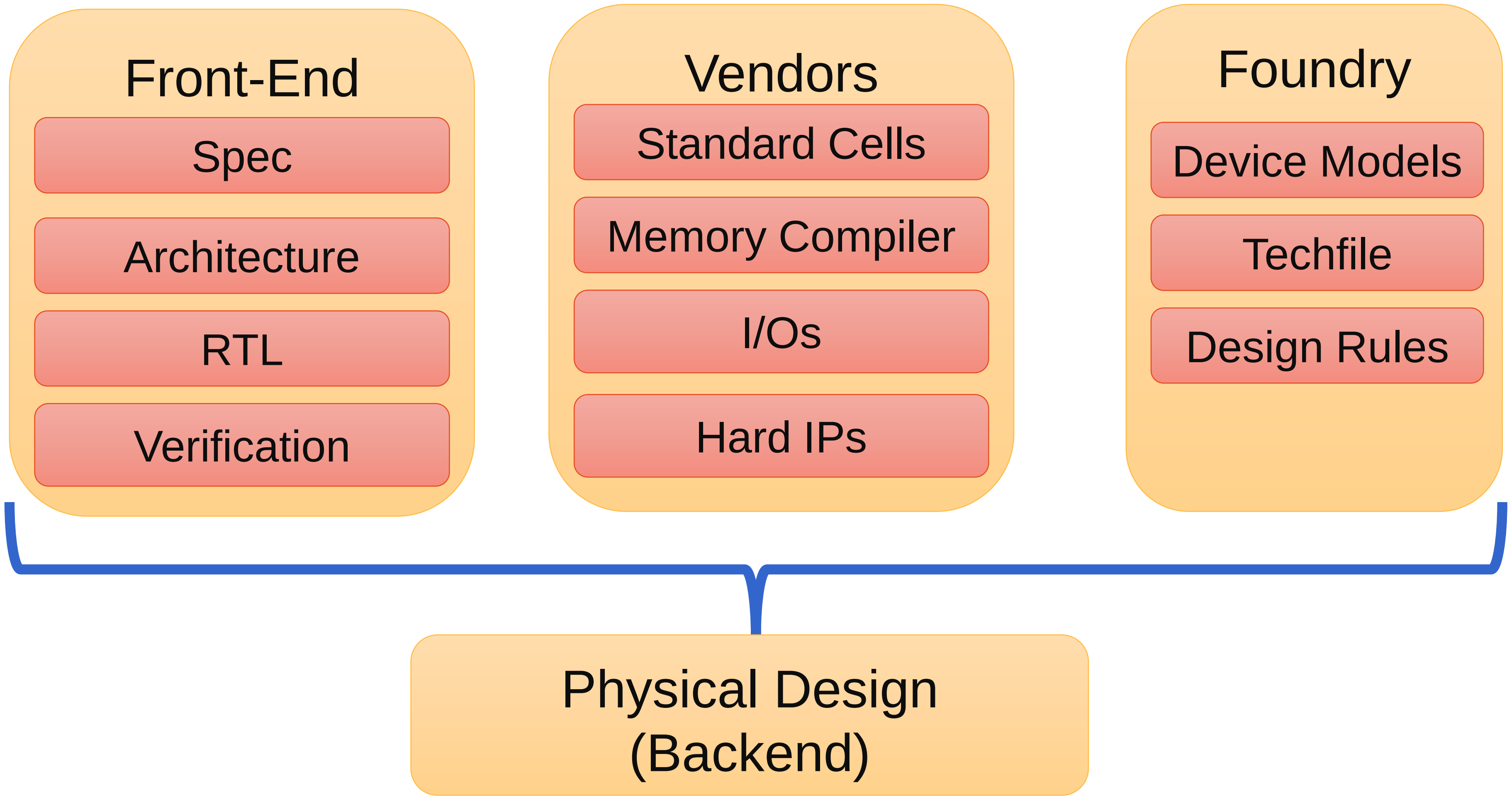
- Floorplan
- I/O Ring
- Power Plan
- Placement
- Clock Tree Synthesis
- Route
- DRC, LVS, Antennas, EM
- LEC, Post-layout

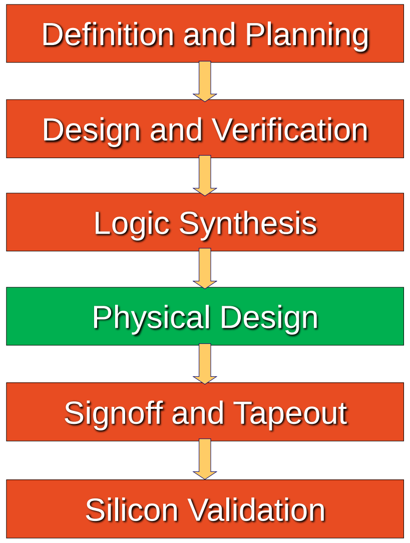




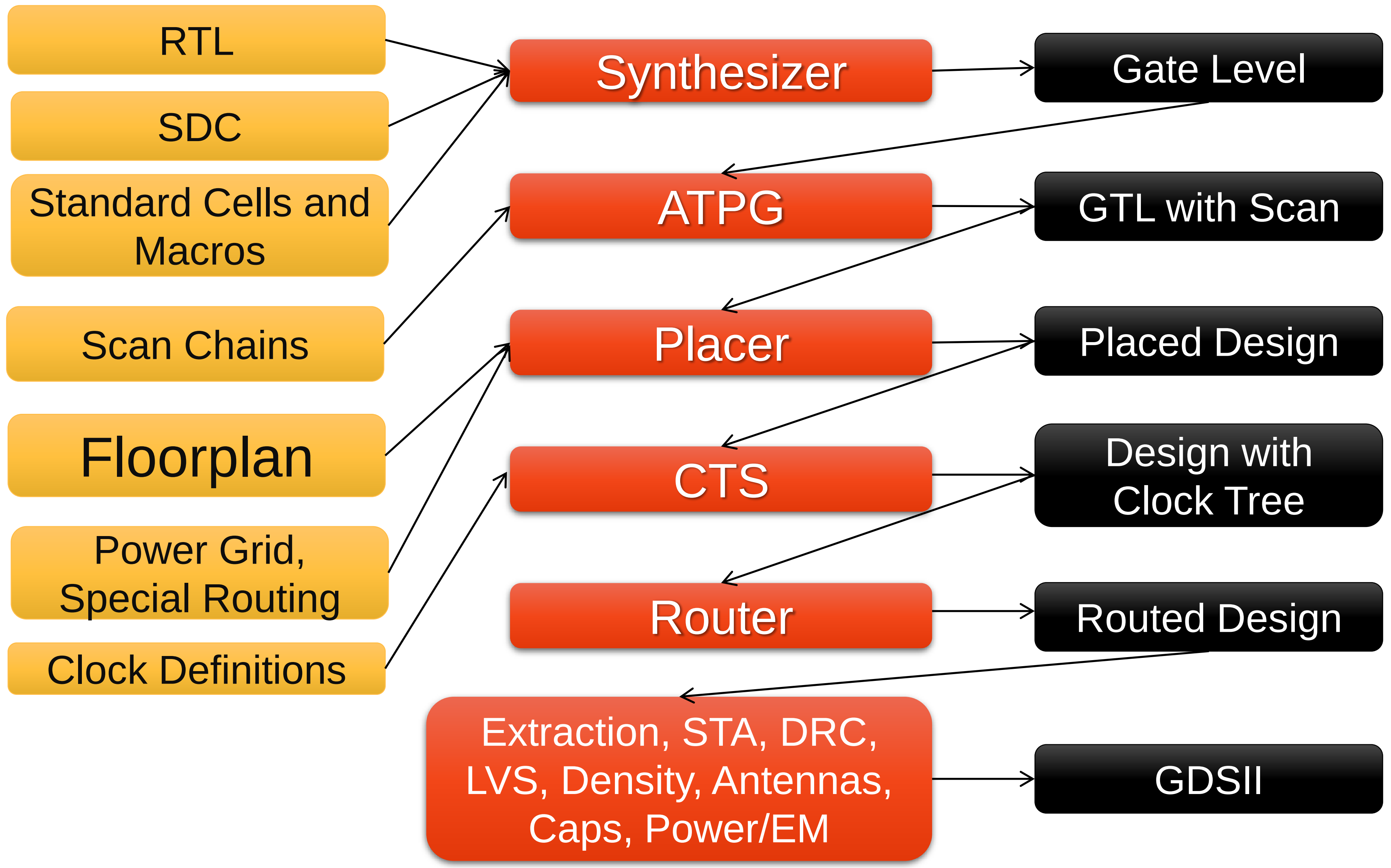
Physical Design – Backend Flow

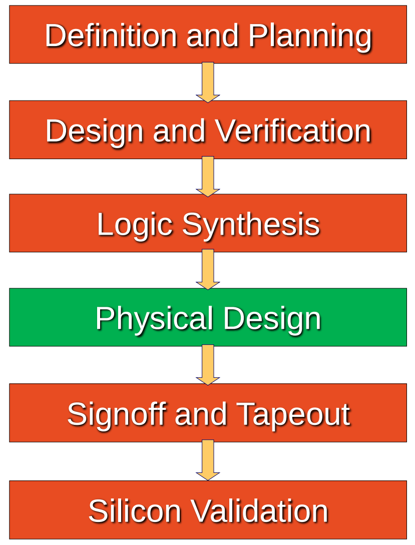
- **Physical Implementation Inputs**





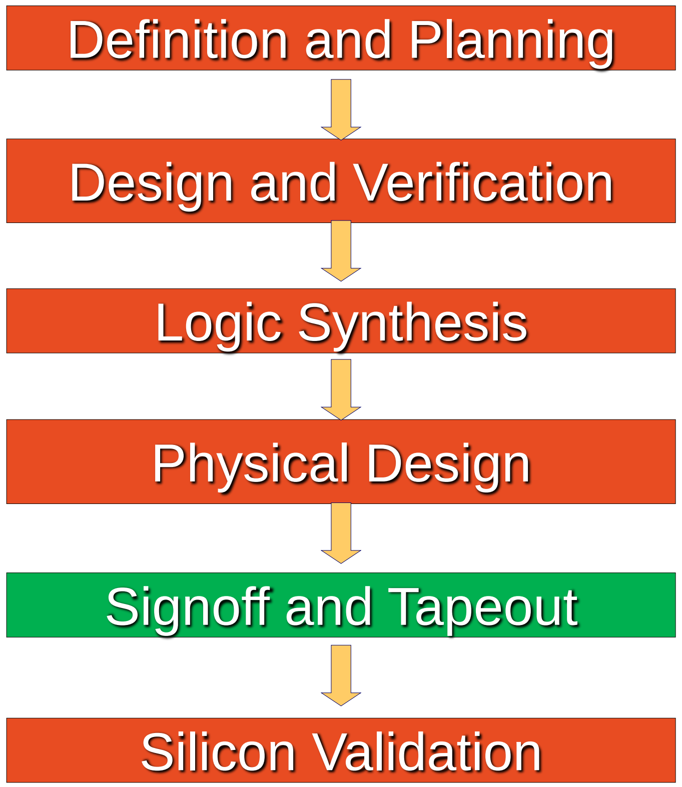
Physical Design – Backend Flow





Signoff and Tapeout

- Parasitic Extraction
- STA with SI
- DRC/LVS/ERC/DFM
- Post-layout Gate-level Simulation
- Power Analysis
- DFT
- Logic Equivalence



Glossary

- **Just to cover most of the terminology of today's lesson:**
  - RTL
  - GTL
  - CAD
  - EDA
  - DFT (ATPG, Scan, BIST)
  - OPC
  - Frontend
  - Backend
  - Verification
  - Signoff
  - Tapeout
  - Hard IP
  - Soft IP
  - FPGA
  - Emulation
  - Lint
  - Formal Verification
  - STA
  - SDC
  - SI
  - DRC, LVS, EM
  - GDSII

Special Thanks to:
Nir Sever
IDESA Digital Course
Marvell Israel
For the knowledge and materials required for preparing this lecture.