ECE201A: Fundamentals of Computer-Aided Design of VLSI Circuits and Systems

Puneet Gupta

Some notes adopted from
Andrew B. Kahng
Lei He
Igor Markov
Mani Srivastava

Instructor Info

- Puneet Gupta
  - http://nanocad.ee.ucla.edu
  - puneetg@ucla.edu
  - 6730C Boelter Hall
- Office hours
  - On Campus: 3pm-4pm Thursdays
  - MSOL: 5pm-6pm Thursdays on Zoom
- "TA": George Karfakis

Syllabus

- RTL (Verilog) to GDSII (layout) flow for large-scale digital designs
  - Logic Synthesis
  - **Physical Design** (partitioning, floorplanning, placement, routing)
  - Test and design for test
- Focus on CAD algorithms
- No textbook. Recommended texts:
  - Assigned paper readings
  - "VLSI Physical Design: From Graph Partitioning to Timing Closure", Springer
  - "Handbook of Algorithms for Physical Design Automation", CRC Press
  - "EDA for IC System Design, Verification, and Testing", CRC Press
  - "EDA for IC Implementation, Circuit Design, and Process Technology"

Prerequisites

- Familiarity with digital circuit design
  - E.g., EE216A, EE115C, EE116B
- Familiarity with basic algorithms and data structures
  - E.g., CS180, CS32
- Reasonable programming skills in C++
  - We will use OpenAccess C++ API for programming throughout the course
- Familiarity with Linux
- Please attempt the following small quiz on your own. If you get less than 7/10, you should reconsider taking the class.

- Basic knowledge prerequisite quiz for ECE201A Survey (surveymonkey.com)

Useful Websites

- EETimes, EDN, …
- DeepChip
- USPTO, Google Patents
- Companies
  - EDA: Synopsys, Cadence, Mentor, …
  - Foundry: TSMC, UMC, GlobalFoundries, SMIC, …
  - IP: ARM, Dolphin, CEVA, …
- DAC, ICCAD, DATE, ASP-DAC, …
- ITRS (public.itrs.net)

Grading

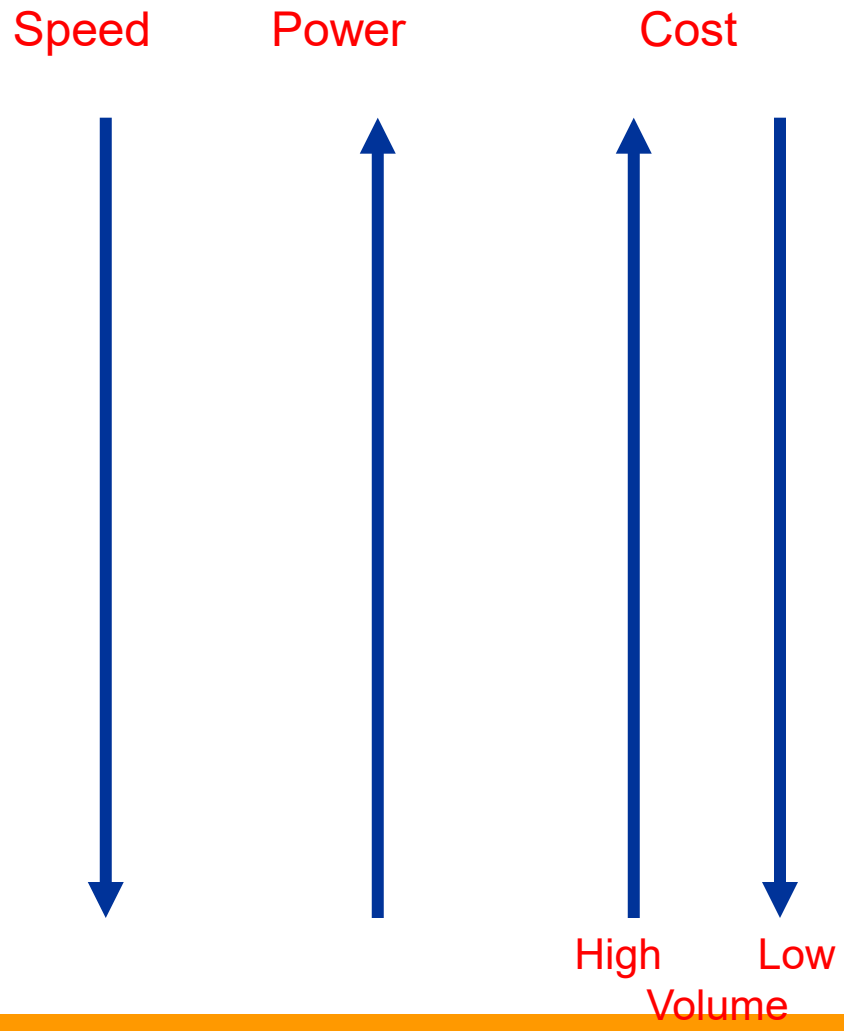
- 5-6 Programming assignments/labs + 1-2 homeworks: 40%
  - All homeworks to be done individually
- Occasional paper per week of reading assignment
  - Homeworks/Quizzes may contain questions pertaining to reading
- Final Project: 40%: Teams of 2 (or 1)
  - Projects assigned in Week 4.
  - Midterm project report due mid-Week 8 (10%)
  - Final project report due end of Week 10
  - 5% points bonus if you do project by yourself (max is still 40%)
- Two in-class quizzes (done on Gradescope): 20%
- If you are taking the course for S/U grade, do NOT submit the project (i.e., only labs/homeworks + quizzes need to be done)

Logistics

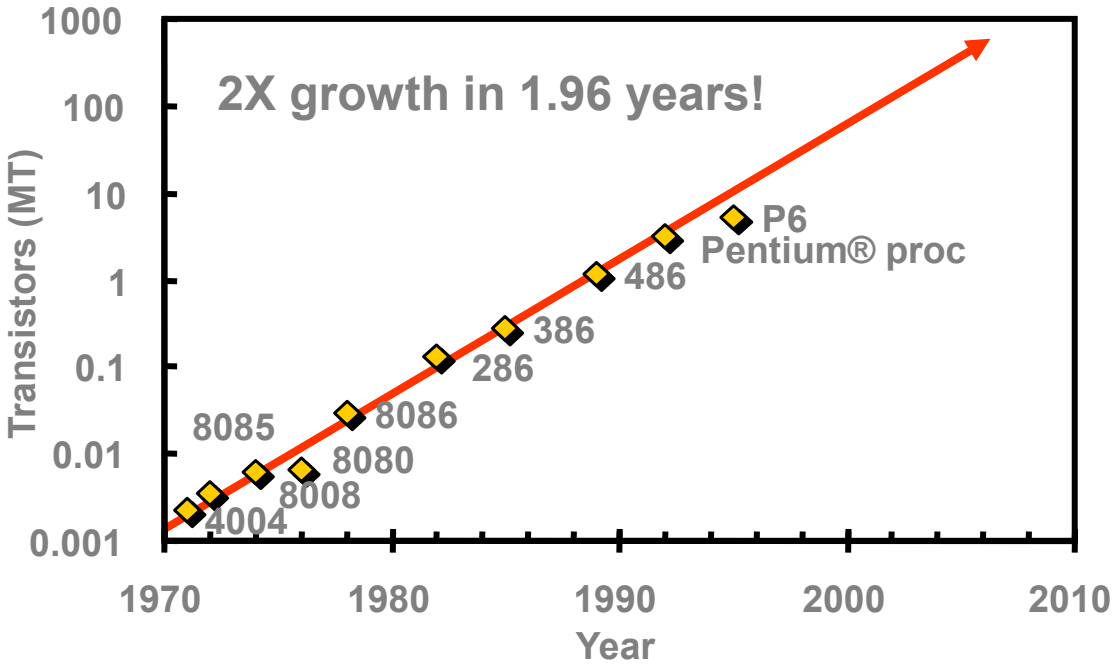
- We will use Piazza and BruinLearn for the class.
  - Grading will be on Gradescope
- HW/Lab report submissions
  - Put the code, etc in a clearly marked readable directory; email the pointer to the TA.
  - For homeworks we will use online submission on Gradescope
  - Only PDFs acceptable
  - Absolutely no email attachments
- Please make sure to have working SEAS logins
  - You have little disk space for course. Use it carefully.
  - Make sure you can login to eeapps.seas.ucla.edu
- PTE Policy
  - If you are PhD student, email me for a PTE#
  - If you are M.S. student, make sure you try the quiz; *and* you should have taken ECE216A (or equivalent) and be very comfortable with C++ programming. Only CES M.S. students, please.

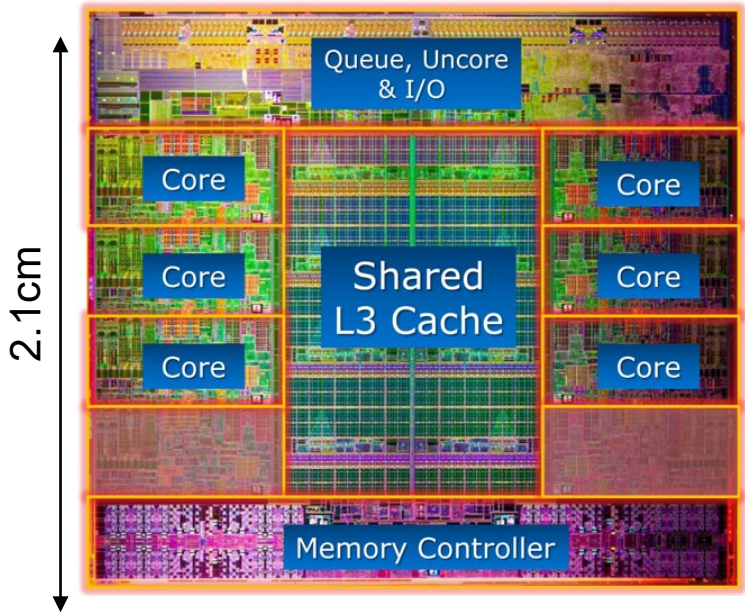
System Implementation Choices

- General Purpose Microprocessors
- Domain-specific processors
  - DSP
  - Network processors
  - Graphics
- FPGA
- Gate Array (structured ASIC)
- ASIC: single purpose chips
  - Full-custom
  - Cell-based



IC Cost and Integration Drivers





2011 Xeon "Sandy Bridge"
2.2 Billion Transistors
32nm technology

- Moore's Law is about *cost*
  - *What is Moore's Law ?*
  - 0.7X dimension (2X area) scaling every technology generation
- Increased integration → decreased cost → more possibilities for semiconductor-based products

Silicon Complexity Challenges

UCLA

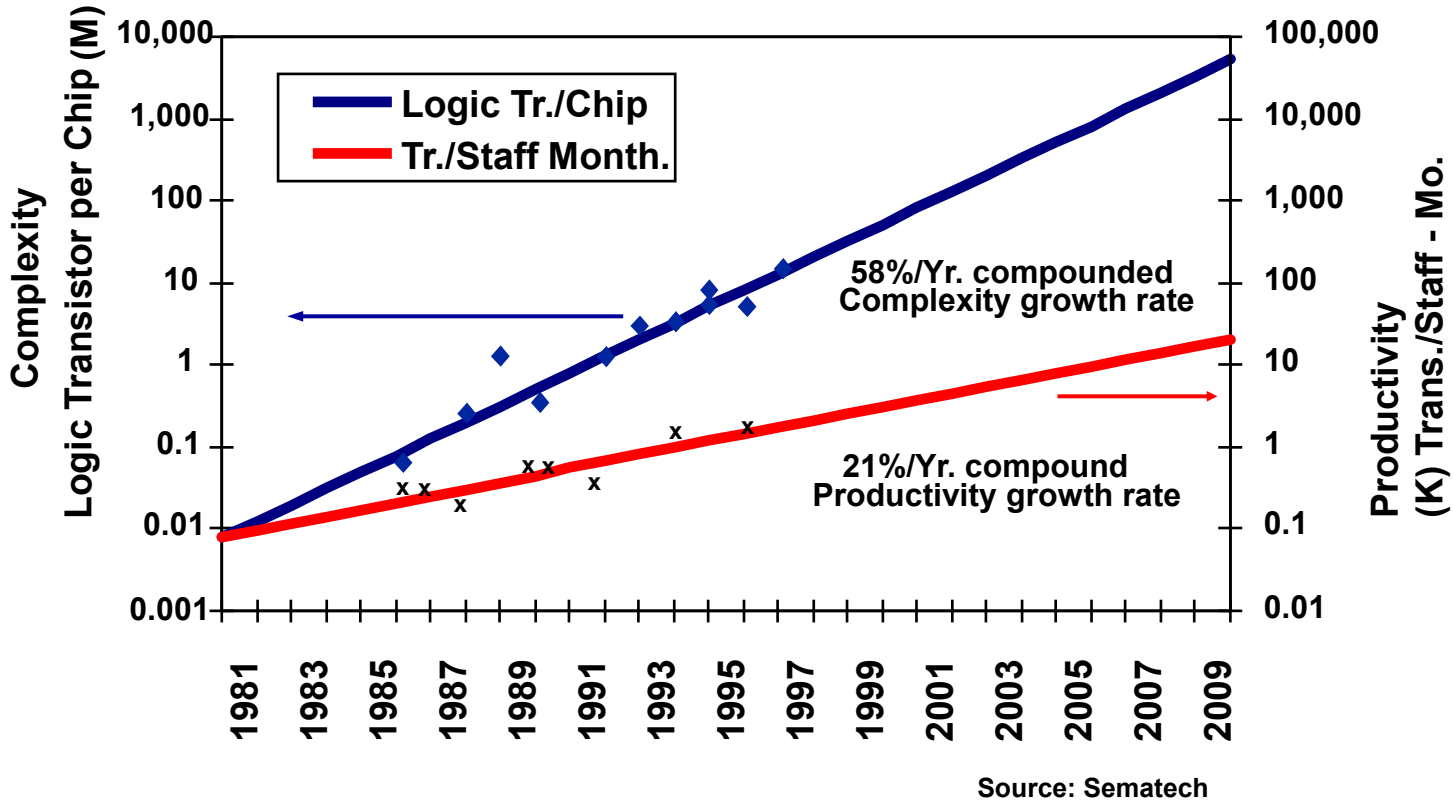
- Silicon Complexity = impact of process scaling, new materials, new device/interconnect architectures
  - Non-ideal scaling (leakage, power management, circuit/device innovation, current delivery)
  - Coupled high-frequency devices and interconnects (signal integrity analysis and management)
  - Manufacturing variability
  - Scaling of global interconnect performance (communication, synchronization)
  - Decreased reliability (SEU, NBTI, electromigration, etc)
  - Complexity of manufacturing handoff (reticle enhancement and mask writing/inspection flow, manufacturing NRE cost)

System Complexity Challenges

UCLA

- System Complexity = exponentially increasing transistor counts, with increased diversity (mixed-signal SOC, …)
  - Reuse (hierarchical design, reuse of verification/test/IP)
  - Verification and test (design for verifiability/test, verification reuse, system-level and software verification, test reuse)
  - Cost-driven design optimization (manufacturing cost modeling and analysis, quality metrics, die-package co-optimization)
  - Embedded software design (platform-based system design methodologies, software verification/analysis, codesign w/HW)
  - Reliable implementation platforms (predictable chip implementation onto multiple fabrics)
  - Design process management (team size / geog distribution, data mgmt, collaborative design)

The Design Problem: Complexity Outpaces Design Productivity

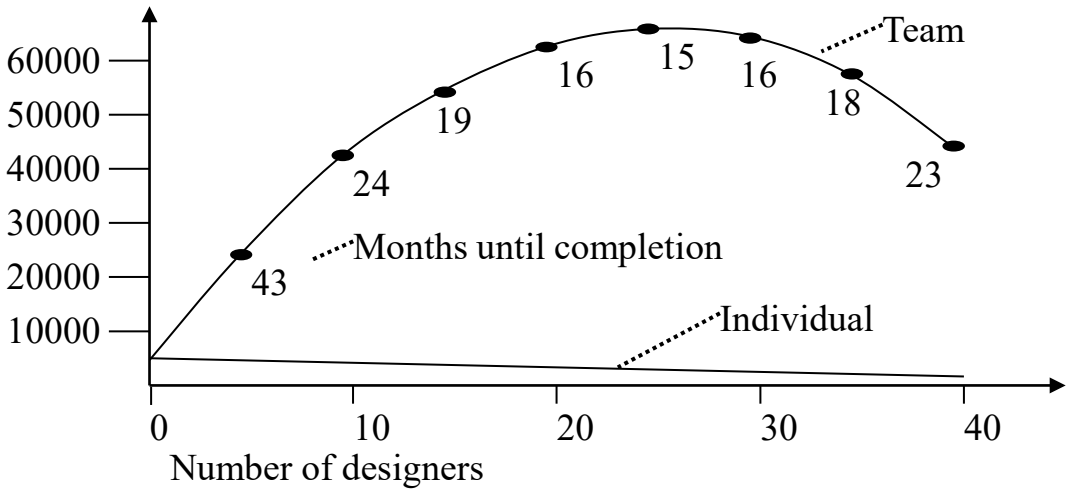


A growing gap between design complexity and design productivity

Throwing More Designers Not the Answer: The Mythical Man-month

- In theory, adding designers to team reduces project completion time
- In reality, productivity per designer decreases due to complexities of team management and communication
- In the software community, known as "the mythical man-month"
- At some point, can actually lengthen project completion time! ("Too many cooks")

- 1M transistors, 1 designer=5000 trans/month
- Each additional designer reduces for 100 trans/month
- So 2 designers produce 4900 trans/month each



[Adapted from Embedded Systems Design: A Unified Hardware/Software Introduction, Copyright 2000 Vahid & Givargis]

Profound Impact on the way VLSI is Designed

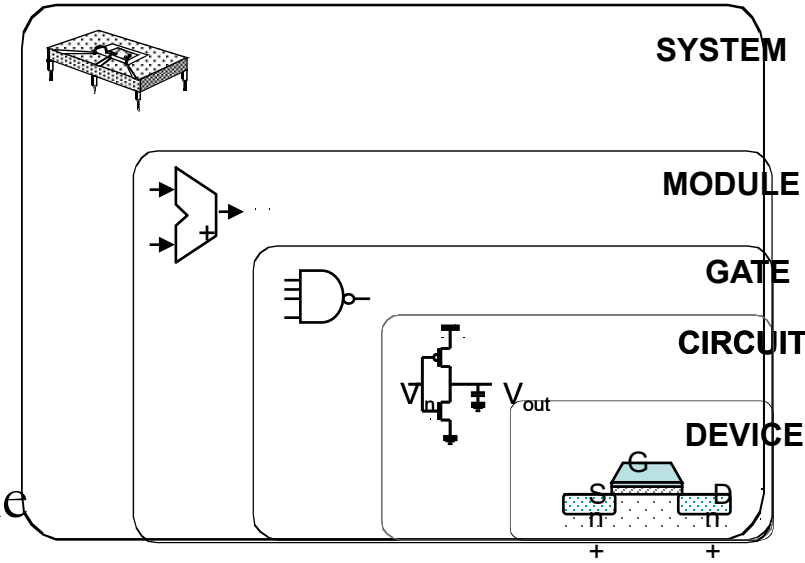
UCLA

- The old way: manual transistor twiddling
  - expert "layout designers"
  - entire chip hand-crafted
  - okay for small chips… but cannot design billion transistor chips in this fashion
- The (not so) new way: using CAD tools at high level
  - tools do the work…
  - high levels of abstractions
    - synthesis from a description of the behavior
  - libraries of reusable cores, modules, and cells

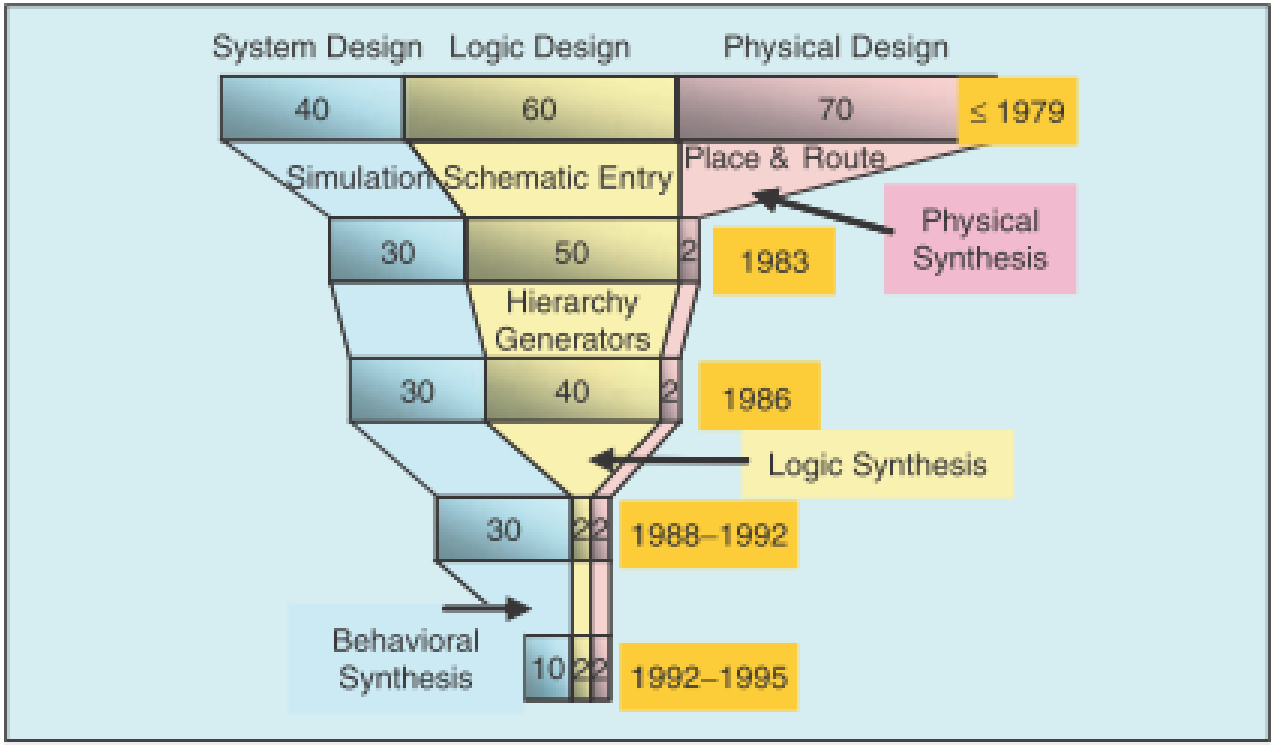
Chip design increasingly like object-oriented software design!

Design Principles

- Partition the problem → divide and conquer, hierarchy
  - Different abstraction levels: RTL, gate-level, transistor-level
- Orthogonalize concerns
  - Logic vs. timing
- Constrain the design space to simplify the design process
  - Balance between design complexity and performance
  - E.g., standard-cell methodology vs. custom design



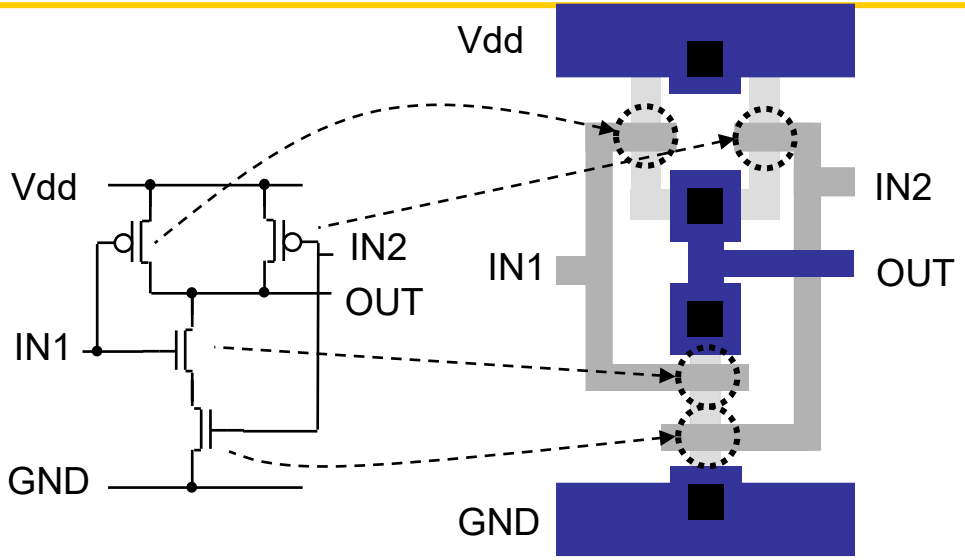
History of Progress in EDA

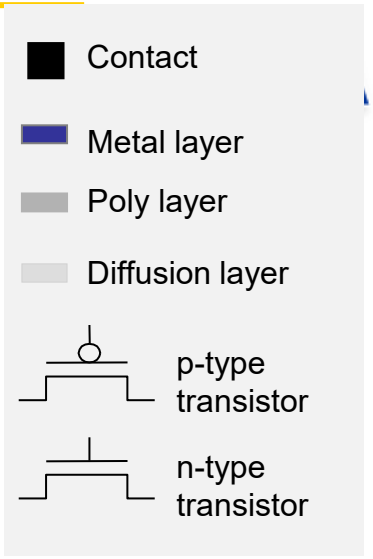


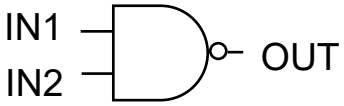
- Reading uploaded on Piazza
  - History of EDA at IBM

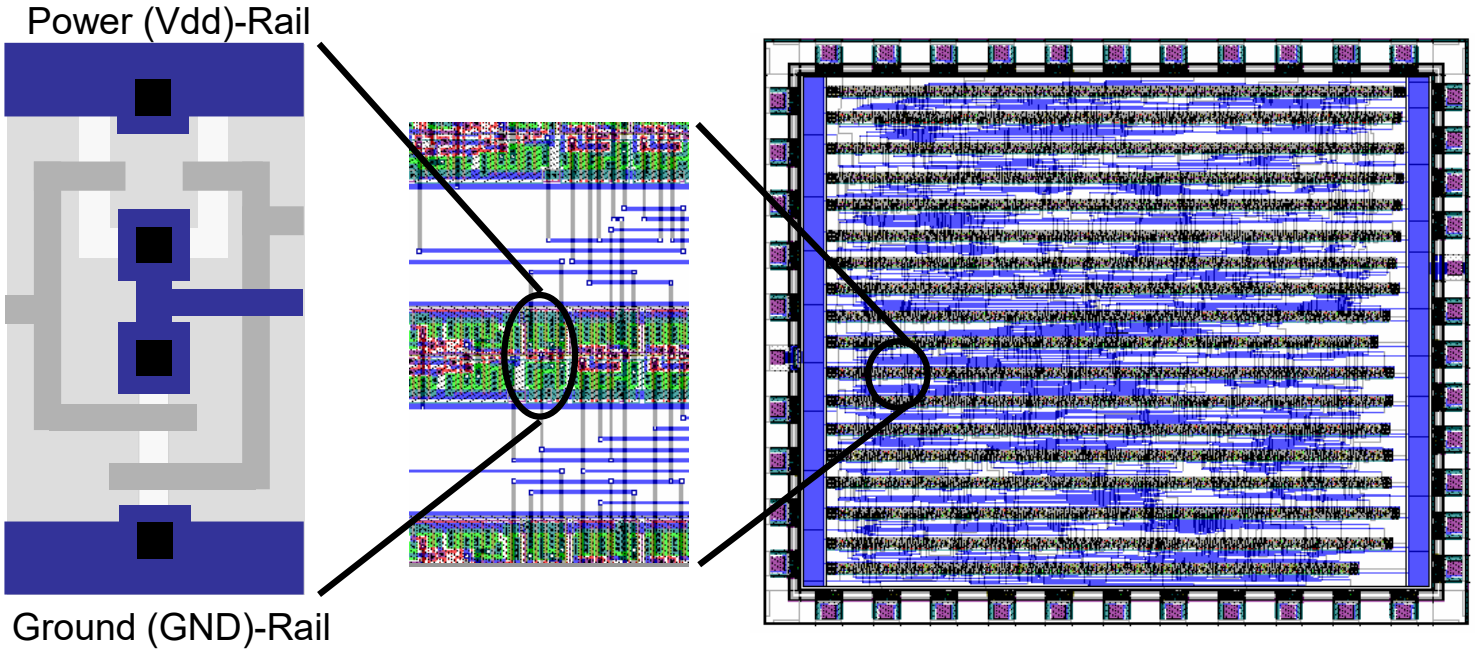
5 min break

The Layout


Vdd
IN2
OUT
IN1
GND
Vdd
IN2
IN1
OUT
GND

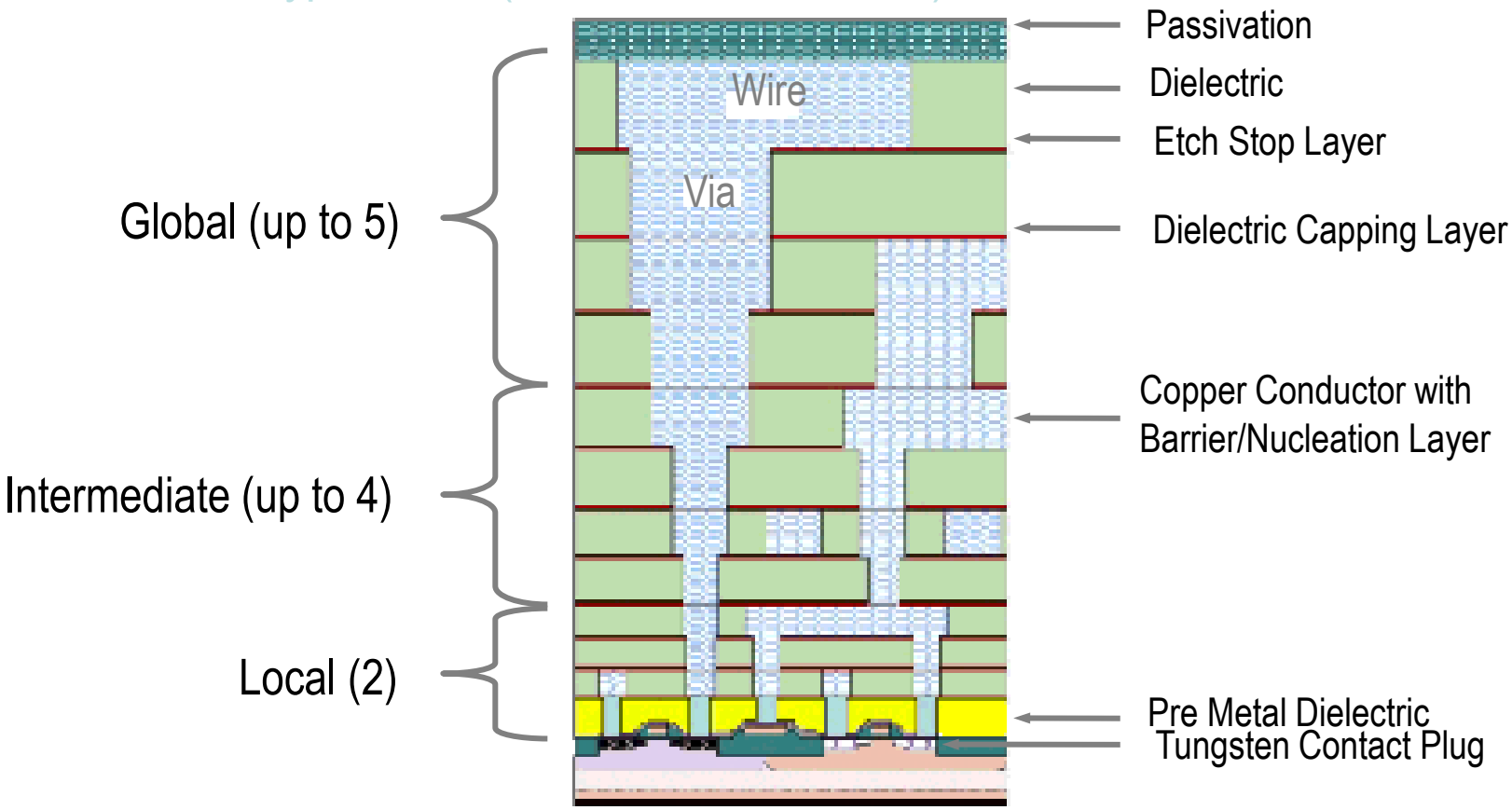

Contact
Metal layer
Poly layer
Diffusion layer
p-type transistor
n-type transistor


IN1
IN2
OUT


Power (Vdd)-Rail
Ground (GND)-Rail

How Is It Connected?

SEMATECH Prototype BEOL ("back end of the line") metal stack, 2000



- *Reverse-scaled* global interconnects
  - Growing interconnect complexity
  - Performance-critical global interconnects

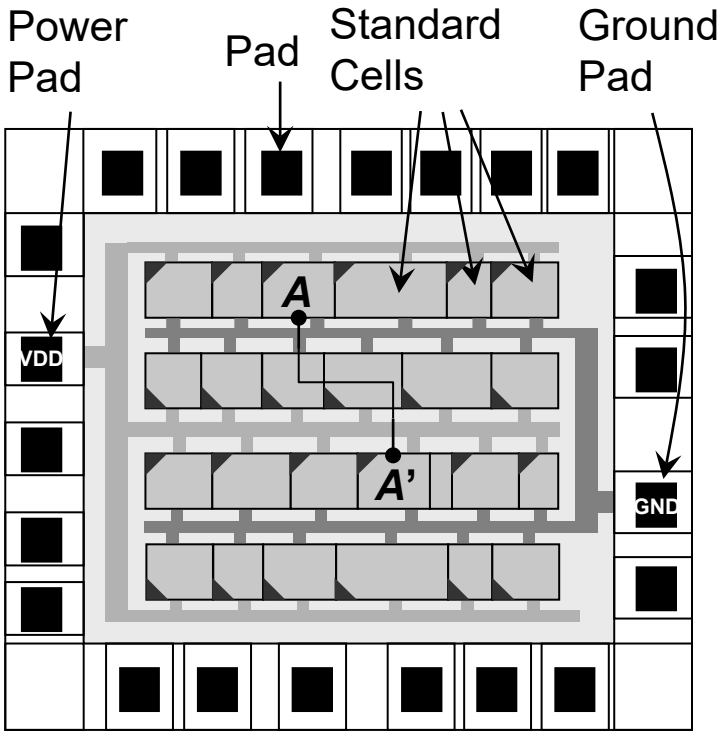
Comparisons of Design Styles

	full-custom	standard cell	FPGA
cell size	variable	fixed height *	fixed
cell type	variable	variable	programmable
cell placement	variable	in row	fixed
interconnections	variable	variable	programmable

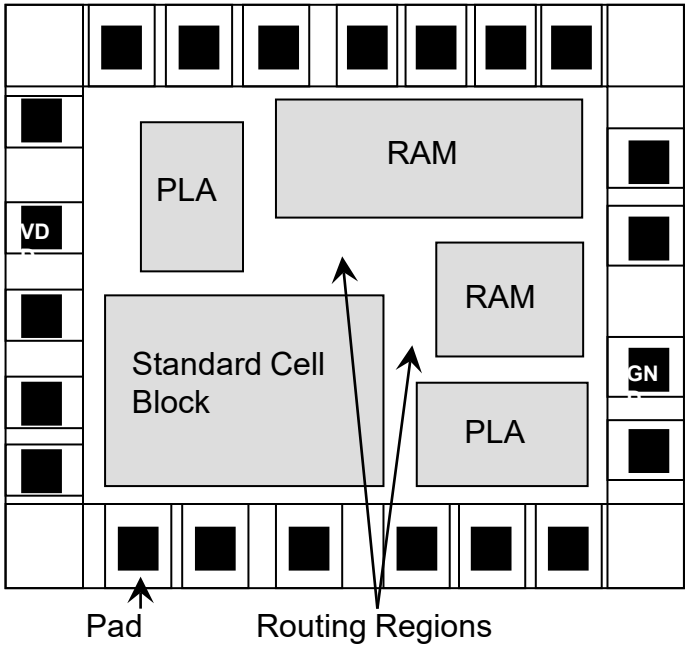
* uneven height cells are also used

Standard Cell Layouts

Standard cell layout using over-the-cell (OTC routing

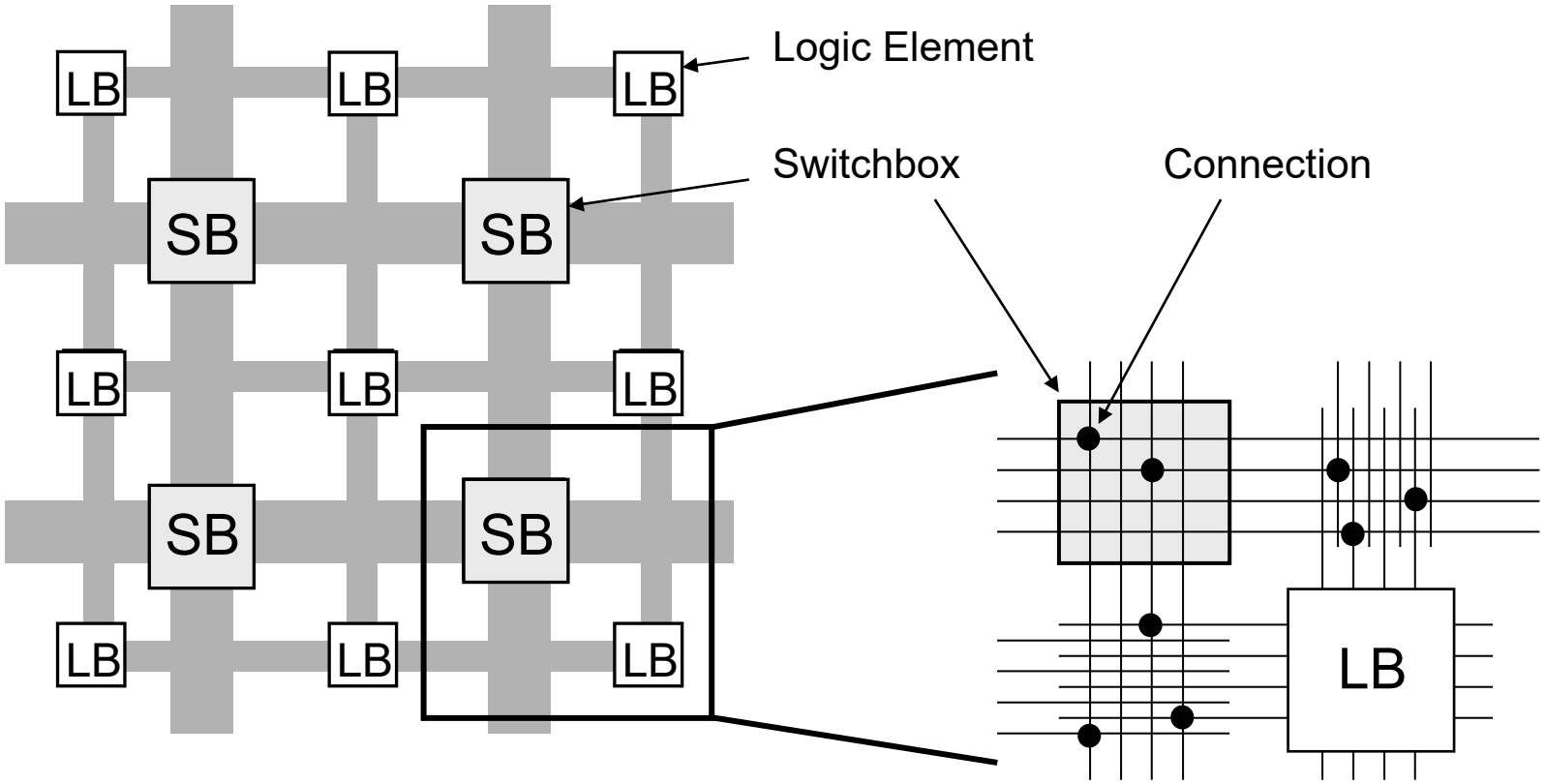


Layout with macro cells



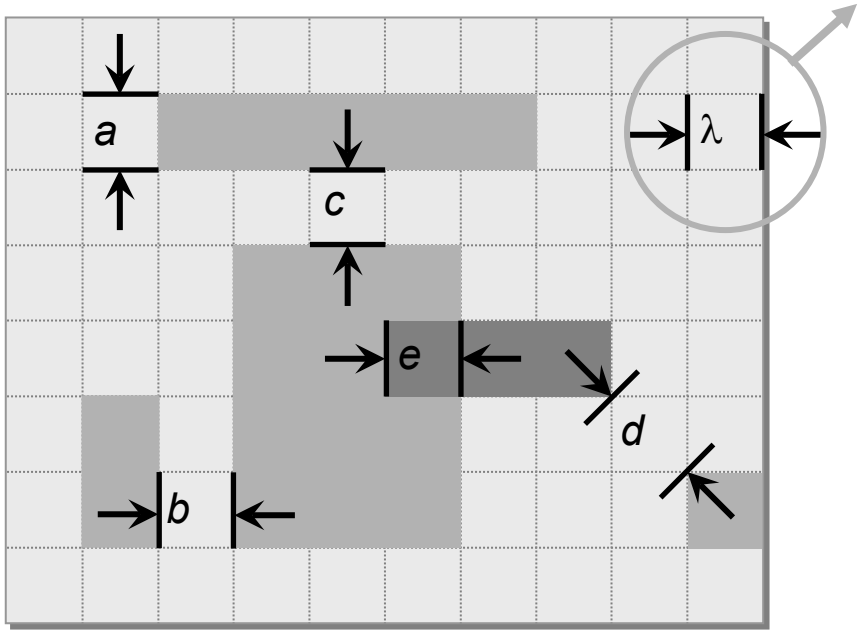
FPGA Layout

Field-programmable gate array (FPGA)



Layout Layers and Design Rules

Categories of design rules



λ: smallest meaningful technology-dependent unit of length

Minimum Width: *a*

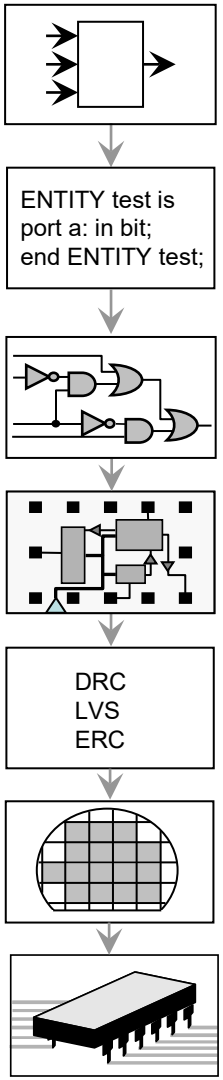
Minimum Spacing: *b*, *c*, *d*

Minimum Overlap: *e*

Library and Design Data

- Models and technology data required to execute the design flow
- Power, timing: Liberty, UPF/CPF
- Layout: LEF/DEF, GDSII, OASIS
- Delays and path timing, constraints, parasitics: SDF, SDC, DSPF, SPEF, SPICE
- Layout rules: "DRM" = design rule manual, captured as a "deck" in Calibre, Hercules
- Design data model: OpenAccess, MilkyWay
  - Hierarchy, connectivity, geometry, timing, …

VLSI Design Flow



System Specification

Architectural Design

Functional Design and Logic Design

Circuit Design and Logic Synthesis

Physical Design

Physical Verification and Signoff

Fabrication

Packaging and Testing

Chip

Partitioning

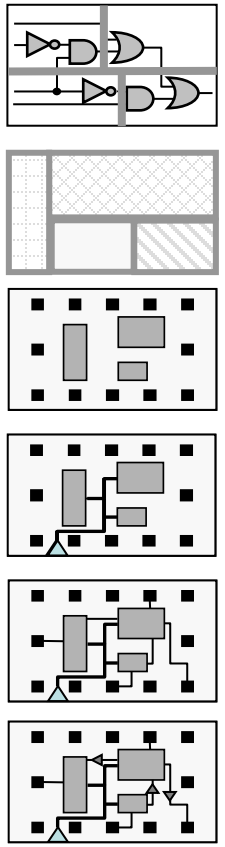
Floorplanning

Placement

Clock Tree Synthesis

Signal Routing

Timing Closure



Behavior → RTL

- High-level synthesis ("C to Verilog")
  - Scheduling
    - Assignment of each operation to a time slot corresponding to a clock cycle or time interval
  - Resource allocation
    - Selection of the types of hardware components and the number for each type to be included in the final implementation
  - Module binding
    - Assignment of operation to the allocated hardware components
  - Controller synthesis
    - Design of control style and clocking scheme
  - Compilation
    - of the input specification language to the internal representation
  - Parallelism extraction
    - usually via data flow analysis techniques
- RTL simulation
  - RTL code, written in Verilog, VHDL or a combination of both, is simulated using testbenches to verify functional correctness
  - Scripted comparison (automated) or Waveform displays (interactive) can be used to verify the outputs

RTL→ Gates

- Logic synthesis: Conversion of RTL to gate-level netlist
  - Targeted to a foundry-specific library
  - Can be performed hierarchically (block by block)
  - Logic minimization done here
  - Timing-driven
    - Clock information (clock is assumed to be ideal zero skew)
    - Primary input arrival times, primary output required times
    - Input driving cells, output loading
    - False paths, multi-cycle paths
    - Interconnect delay may be calculated based on a "wireload model" which uses fanout to estimate delay
- Verification of synthesis results
  - Formal verification
    - RTL description and gate level netlist are compared to verify functional equivalence, thereby verifying the synthesis results
  - Gate-level simulation
    - Covers both functionality and timing
    - Correctness is only as good as the test vectors used
    - Delays are backannotated from the synthesis run

Gates → Layout (Physical Design)

UCLA

- Physical Floorplanning: Defines the basic chip layout architecture
  - Define the standard cell rows and I/O placement locations
  - Place memories and other macros
  - Define power distribution structures such as rings and stripes
  - Allow space for power, clock, major buses
- Place & Route
  - Automatically place the standard cells
  - Generate clock trees
  - Add any remaining power bus connections
  - Route clock lines
  - Route signal interconnects
  - Design rule checks on the routes and cell placements
  - Timing driven tools
    - Require timing constraints and analysis algorithms similar to those used during the static timing analysis step

Analyses

- Static Timing Analysis
  - Verifies that design operates at desired frequency in an input-independent pessimistic fashion
  - Used throughout synthesis, placement, routing
- RC(L) Extraction
  - Calculate resistance and capacitance (and inductance) of interconnects
    - Based on placement of cells
    - Routing segments
  - Drive delay calculation, signal integrity analysis (crosstalk, other noise), static timing
- Physical Verification
  - DRC – Design Rule Check
    - Polygon/Layer spacing rules
    - Verifies the design rules (DRC)
  - LVS – Layout Versus Schematic
    - Verifies that layout and netlist are equivalent at the transistor level
- Functional Verification
  - LEQ – Logical Equivalence Check
    - Check whether two netlists are logically the same
  - Simulation, emulation, ….

Release to Manufacturing

- Final edits to the layout are made and final design rules and other manufacturing constraints are checked
- Manufacturing information such as scribe lanes, seal rings, mask shop data, part numbers, logos and pin 1 identification information for assembly are also added
- 'Tapeout' documentation is prepared prior to release of the GDSII to the foundry
  - GDSII (Stream Format)
    - Final merge of layout, routing and placement data for mask production
- Pad location information is prepared, typically in a spreadsheet
- Manufacturing steps
  - generation of masks
  - silicon processing
  - wafer testing
  - assembly and packaging
  - manufacturing test

Logistics

- Week 1 paper reading (uploaded to Piazza):
  - "EDA in IBM: Past, Present, and Future"
- Next lecture: quick overview of common algorithms and terminology in CAD