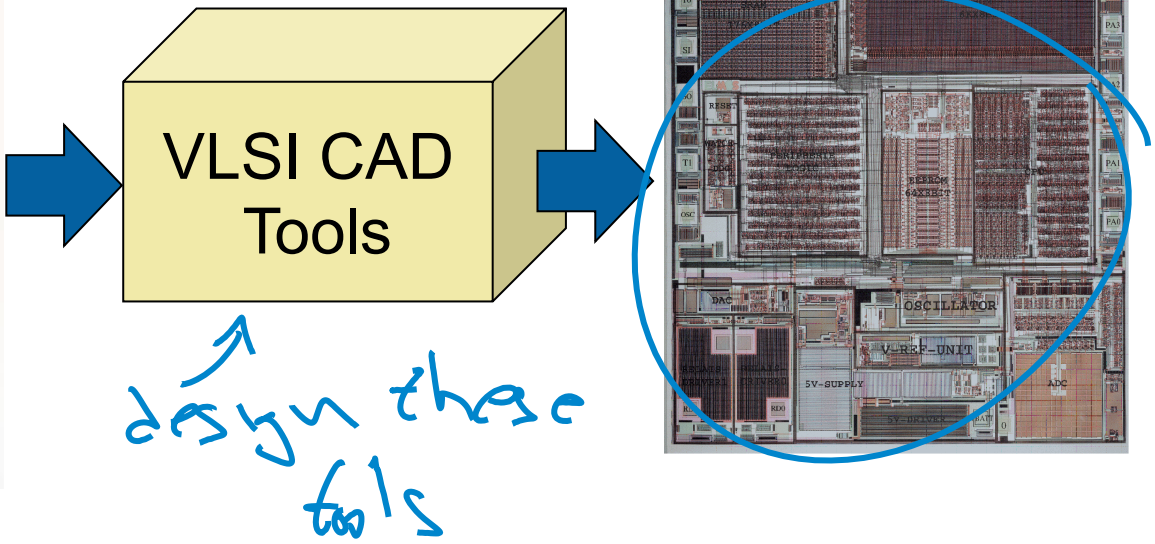

VLSI CAD:
Logic to Layout
Lecture 1
Welcome & Introduction
Rob A. Rutenbar
University of Illinois
Chris Knapton/Digital Vision/Getty Images

Welcome!

- **What we are about in this class…**



Class Logistics

- **10 weeks = 8 weeks of lectures + 1 free week + final exam**
- **Videos every week**
  - 2-3 hours in total
- **8 Problem Sets (i.e., homework assignments)**
  - 8 weeks of video material → 8 assignments
  - Leave Week-9 open for you to finish stuff
- **4 Programming Assignments (i.e., you write some code/scripts)**
  - Some conventional coding
  - Some 'scripts' run thru CAD-centric tools running on our servers

Class Grading: Two Tracks

- **Basic Track**
  - **8** Problem Sets = **75%**
    - Single submission; late submission allowed after deadline for 50% of credit
  - **1** Final exam = **25%**
    - Single submission.
- **Idea: Do this is you don't have time to do the code**

- **Advanced Track**
  - **8** Problem Sets = **40%**
    - Same single submit policy
  - **4** Program Assignments = **40%**
    - Multiple submissions ok; late submission allowed after deadline for 50% of credit
  - **1** Final exam = **20%**
    - Same single submission
- **Idea: Do this for *deepest* understanding of course**

Other Important Stuff

- **Honor code**
  - OK to talk with and work with other people in the class
  - BUT – what you submit must be **your own work**, for homework and for any code
  - AND – please do **NOT** post solutions to any assignments on Coursera site, or share these solutions face to face, in email, via the web, with others in this course

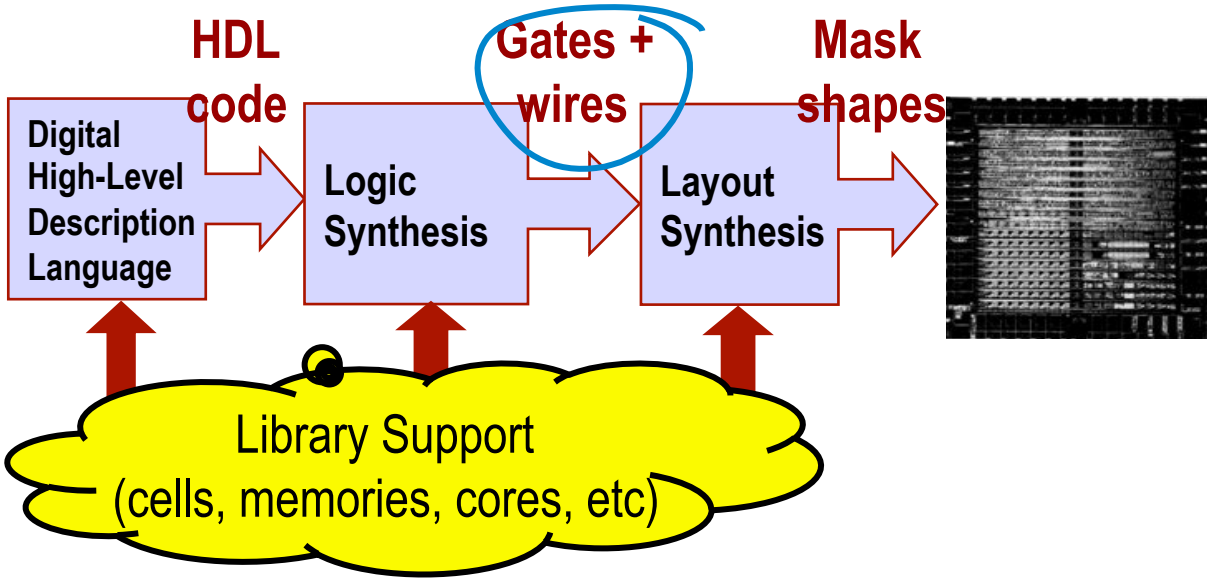
- **Use Coursera interaction mechanisms**
  - Coursera supports discussion forums to ask questions, etc.
  - We will make use of these to help connect you to us (and to each other)

What Background Do You Need?

- **Computer science**
  - Basic programming skills
  - Data structures
- **Computer engineering**
  - Basic digital design (gates, flip flops, Boolean algebra, Kmaps)
  - Combinational and sequential design (finite state machines)
- **Mathematics**
  - Discrete: Basic sets, functions, careful notation
  - Exposure to graph theory is nice but not essential
  - Continuous: Basic calculus, derivatives, integrals, matrices
- **Basic VLSI knowledge**
  - Some chip layout exposure is nice, but not essential

So What is the Course All About...?

- **CAD for semi-custom ASICs**
  - **ASIC** = application-specific integrated circuit
  - **Semi-custom** = try to design reusing some already designed parts
  - **CAD** = flow through a sequence of design steps and software tools



Some Useful Acronyms

- **Semi-custom ASIC**
  - **Application-specific IC** - design a chip for a specific task, using mostly semi-custom techniques
  - Do not expect to make a zillion of them, so cannot afford full custom
  - Not quite as dense (transistors / area) or as fast (GHz) as full custom
- **Semi-custom vs. full-custom**
  - **Semi-custom:** designs mostly from pre-existing parts (gates, memories)
  - **Full-custom:** designs right down at the individual transistor level
  - Today, only things like microprocessors are "full custom"
  - And in fact, even these chips have huge semi-custom parts on them

One More: CAD vs. EDA

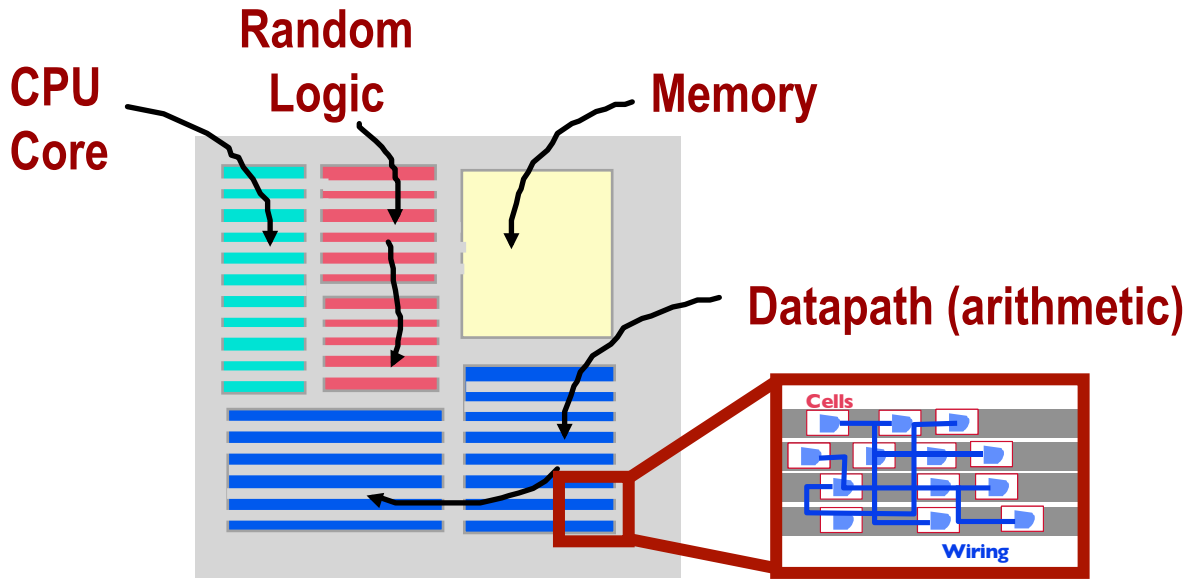
- **CAD: Computer-Aided Design**
  - What we all used to call this world of tools for chip design
  - Problem: other people do "CAD" too, like mechanical engineers, architects, etc.

- **EDA: Electronic Design Automation**
  - What most "insider" chip folks call it. More accurate, more descriptive name
  - Problem: people outside the business not always clear what it means.

- **So, I called this class "VLSI CAD," but it's really "VLSI EDA"**

More Acronyms: System-on-a-Chip ASIC

- **SOC: Integrates many blocks of function on one big chip**
  - Most common: row-based standard cells = gates + flops in rows; and big SRAM memories; and perhaps pre-designed blocks like CPUs



Example: Small SOC Controller Design

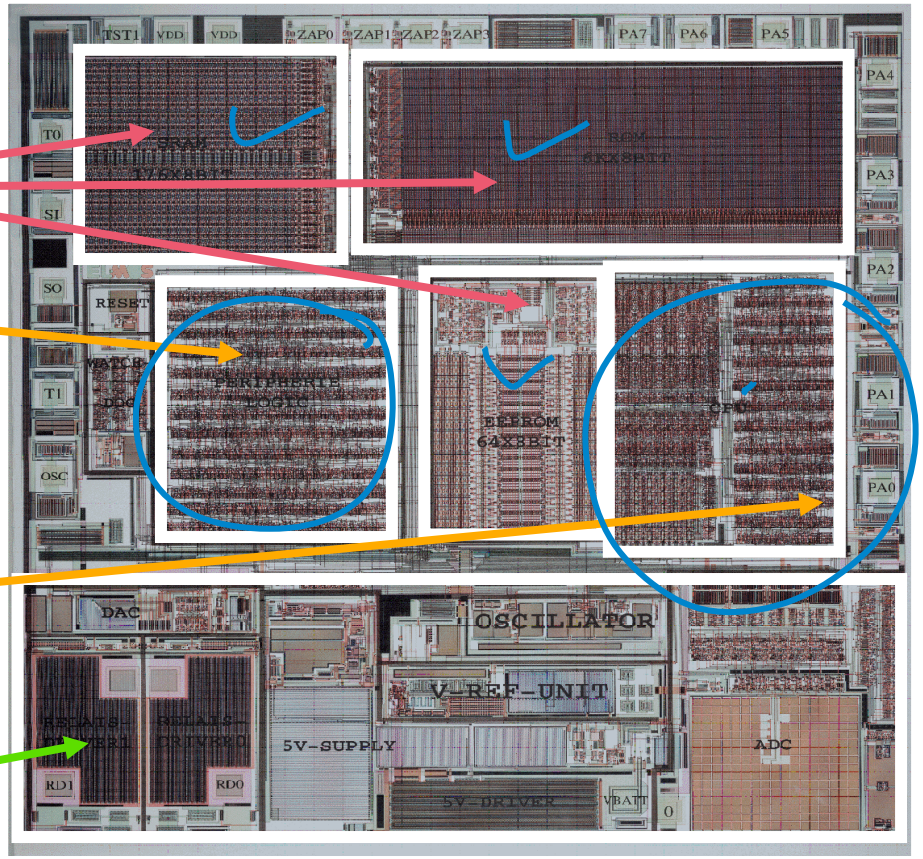
- **Look at blocks**

Memories

Random control logic

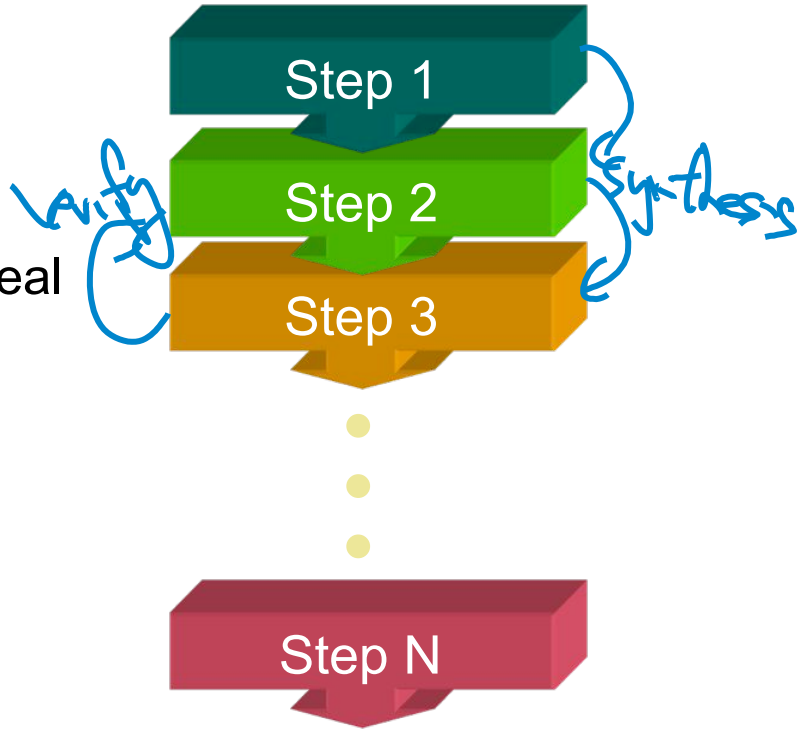
CPU core

Analog interface to external world

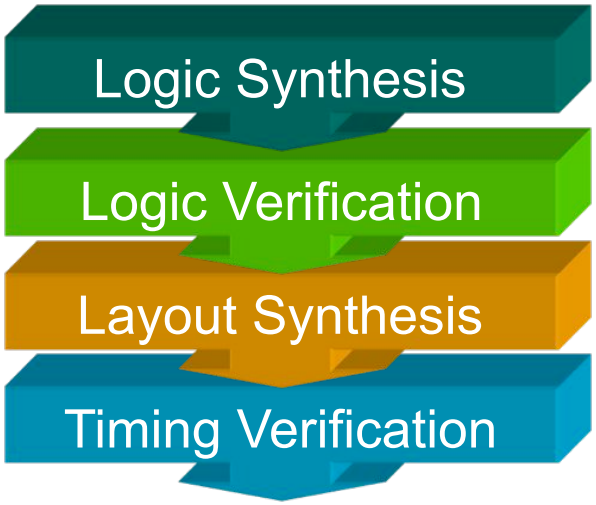


Another Important Term: CAD Flow

- **How to attack big designs like these?**
- **Big idea: Levels of abstraction**
  - Break problem down into smaller steps
  - Each step renders design a little more real
- **Synthesis steps:**
  - Go forward in design: Make new stuff
- **Verification steps:**
  - Look backward: Check that it worked
- **Complete set of steps called: A Flow**



Our Class CAD Tool Flow



- **Start with some Boolean / logic design description…**
- **…end with gates+wires, located at (x,y) coordinates on chip**
- **Note: very over-simplified**
- **Big goal(s) for class**
  - Explain the critical algorithms, data structures & modeling assumptions used in each of these big steps