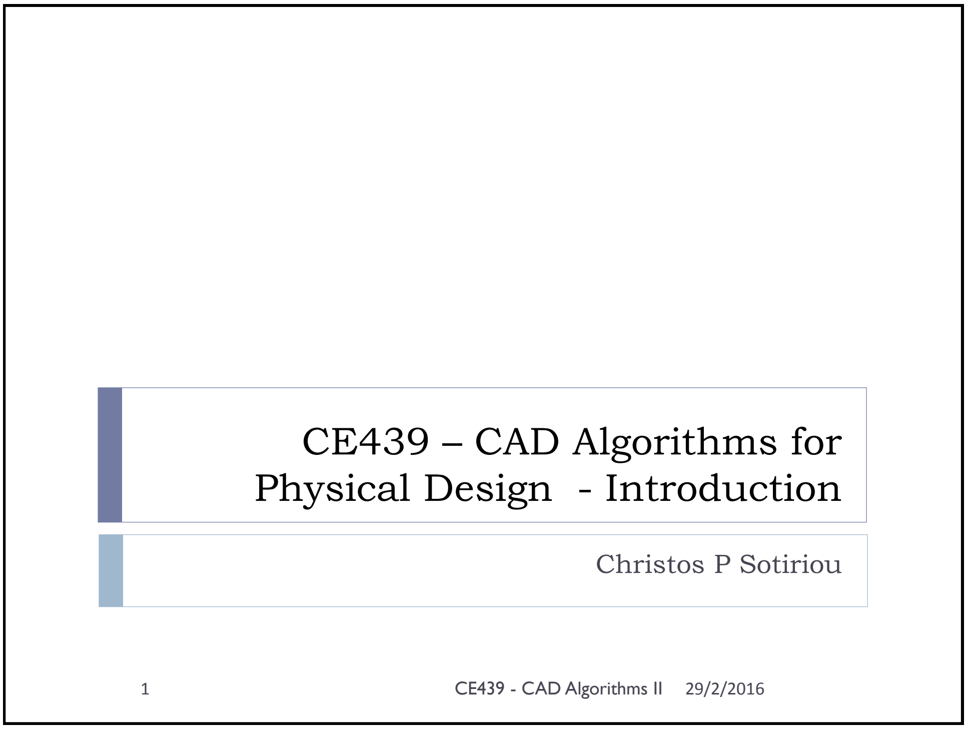
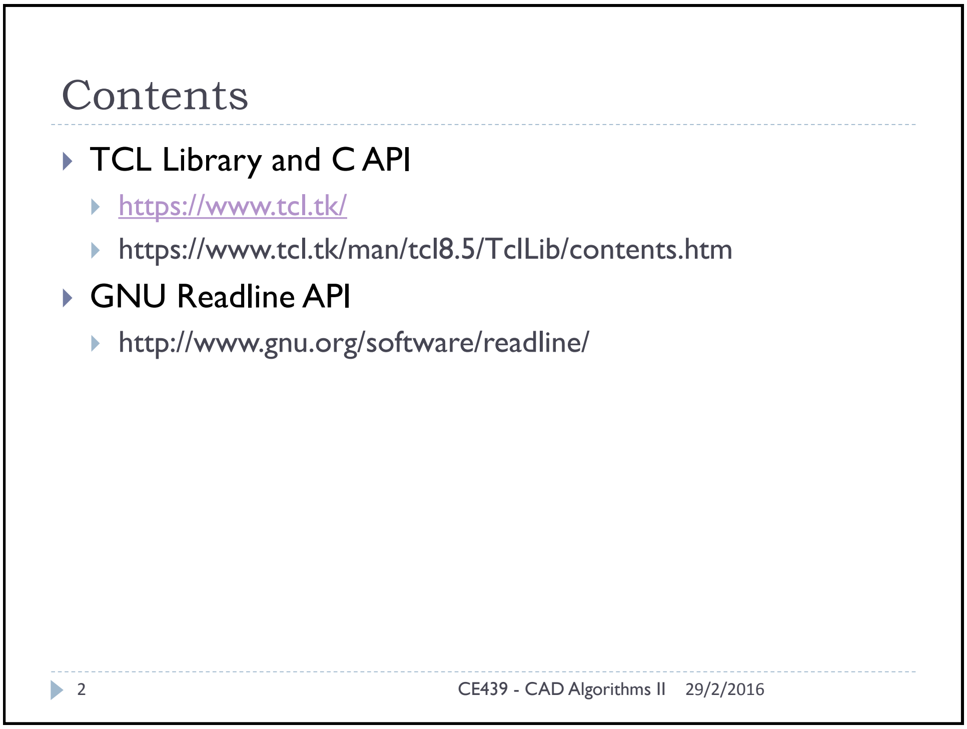


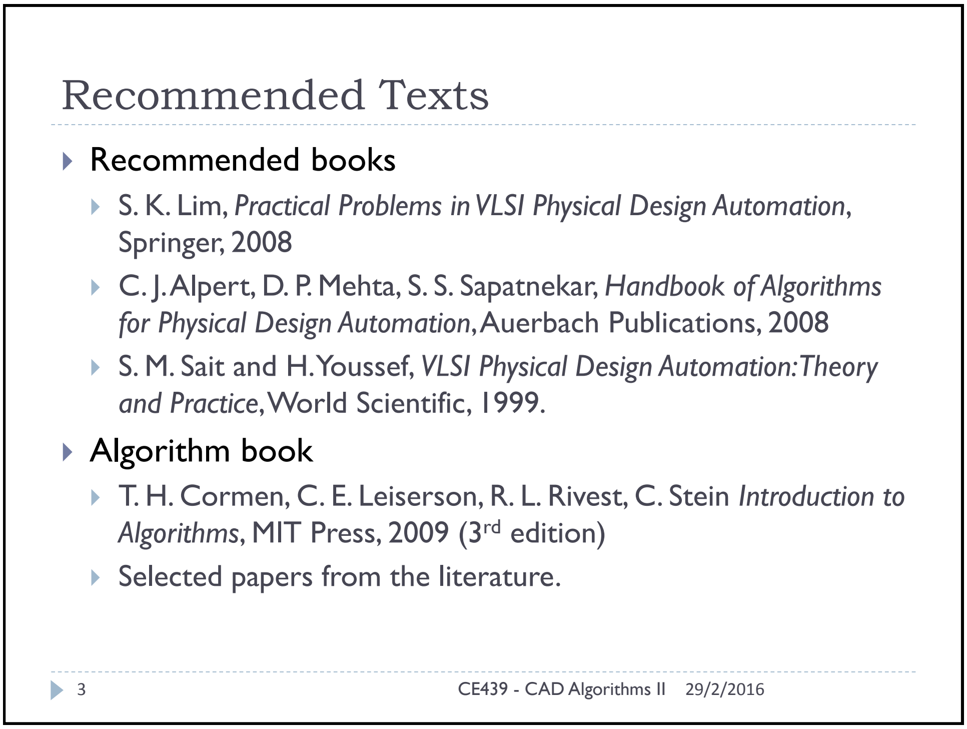
CE439 – CAD Algorithms for Physical Design - Introduction

Christos P Sotiriou



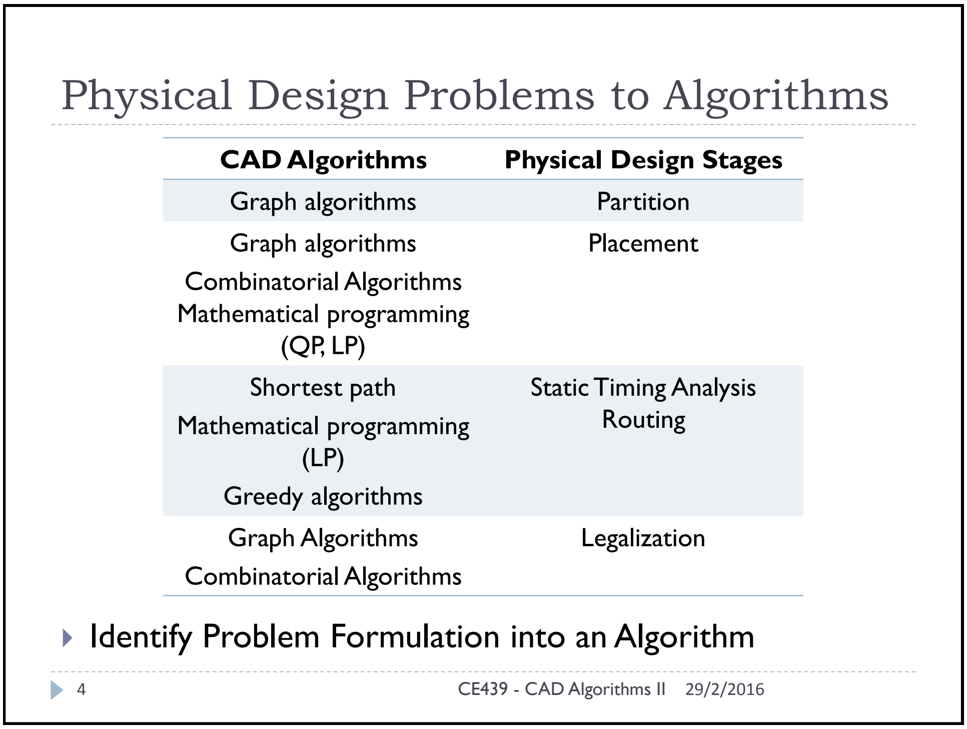
Contents

- TCL Library and C API
  - https://www.tcl.tk/
  - https://www.tcl.tk/man/tcl8.5/TclLib/contents.htm
- GNU Readline API
  - http://www.gnu.org/software/readline/



Recommended Texts

- Recommended books
  - S. K. Lim, *Practical Problems in VLSI Physical Design Automation*, Springer, 2008
  - C. J. Alpert, D. P. Mehta, S. S. Sapatnekar, *Handbook of Algorithms for Physical Design Automation*, Auerbach Publications, 2008
  - S. M. Sait and H. Youssef, *VLSI Physical Design Automation: Theory and Practice*, World Scientific, 1999.
- Algorithm book
  - T. H. Cormen, C. E. Leiserson, R. L. Rivest, C. Stein *Introduction to Algorithms*, MIT Press, 2009 (3rd edition)
  - Selected papers from the literature.



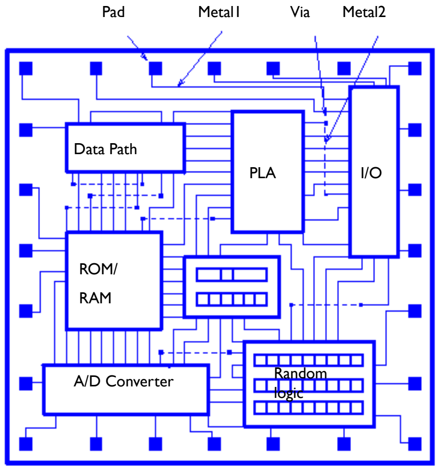
Physical Design Problems to Algorithms

CAD Algorithms	Physical Design Stages
Graph algorithms	Partition
Graph algorithms Combinatorial Algorithms Mathematical programming (QP, LP)	Placement
Shortest path Mathematical programming (LP) Greedy algorithms	Static Timing Analysis Routing
Graph Algorithms Combinatorial Algorithms	Legalization

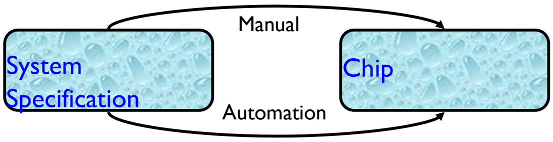
- Identify Problem Formulation into an Algorithm

Physical Design Structure

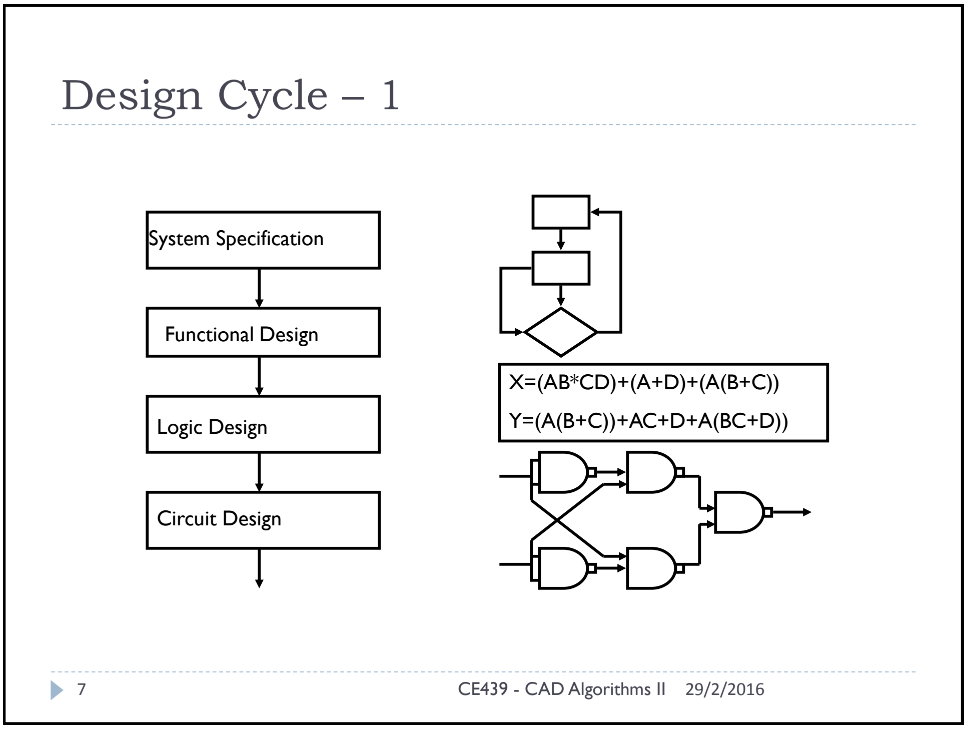
- Devices
  - Transistors
  - Logic gates and cells
  - Function blocks
- Interconnects
  - Local signals
  - Global signals
  - Clock signals
  - Power/ground nets



Physical Design Cycle



- Large number of devices
- Optimization requirements for high performance
- Time-to-market competition
- Power (and other) constraints



Design Cycle – 1

System Specification

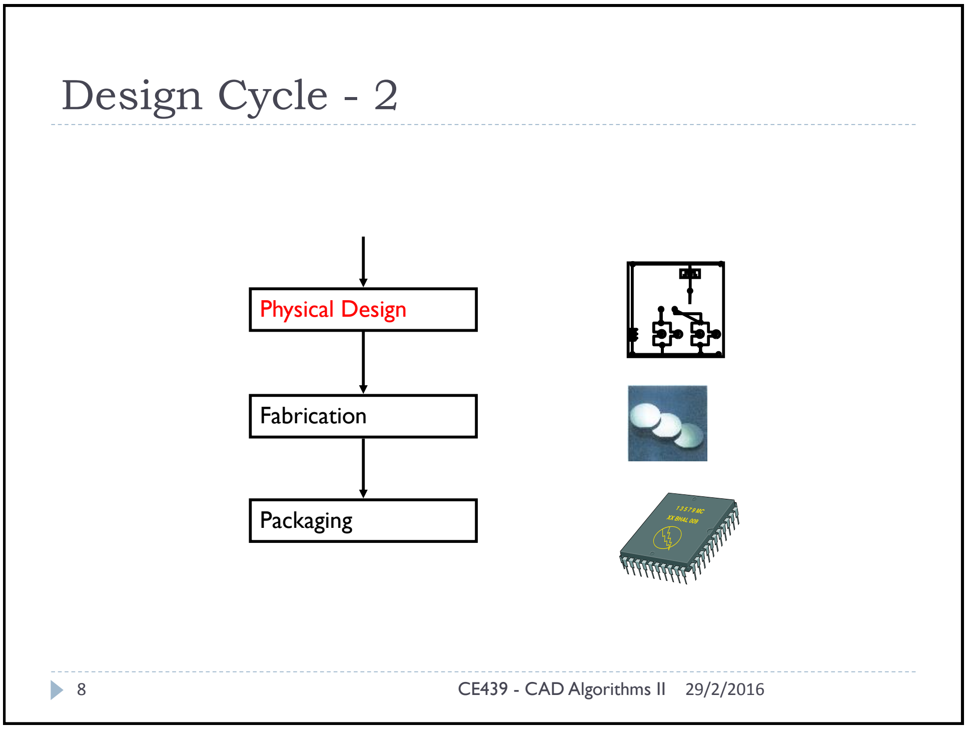
Functional Design

Logic Design

Circuit Design

X=(AB*CD)+(A+D)+(A(B+C))

Y=(A(B+C))+AC+D+A(BC+D))



Design Cycle - 2

Physical Design

Fabrication

Packaging

Physical Design Steps

- Design Steps in More Detail
  - Partitioning/Clustering
  - Floorplanning
  - I/O Pin Assignnent
  - Placement
  - Clock Tree Synthesis
  - Global Routing
  - Detail Routing

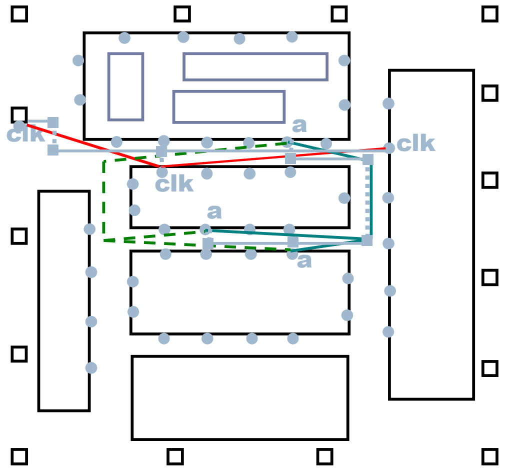
Physical Design Steps

Design Steps:

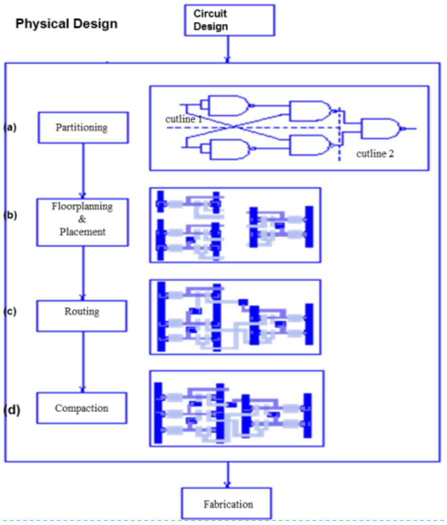
Partition & Clustering
Floorplan & Placement
Pin Assignment
Global Routing
Detailed Routing

Methodology:

Divide-and-Conquer



Physical Design Steps



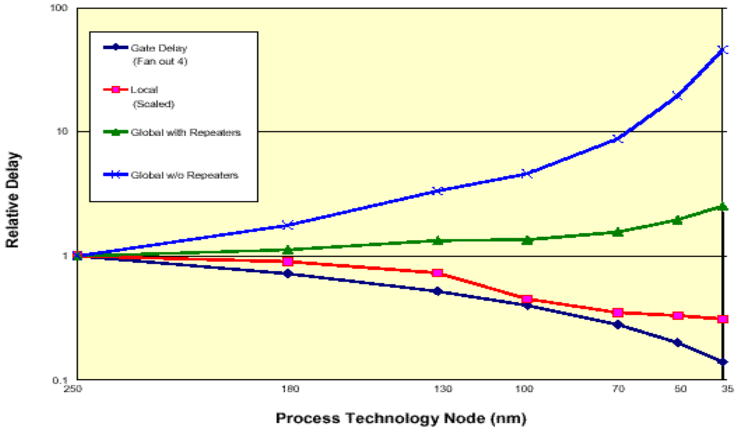
Complexities of Physical Design

- More than 100 million transistors
- Performance driven designs
- Power-constrained designs
- Time-to-Market
- PPA (Power-Performance-Area) is key metric

Why is Physical Design Important?

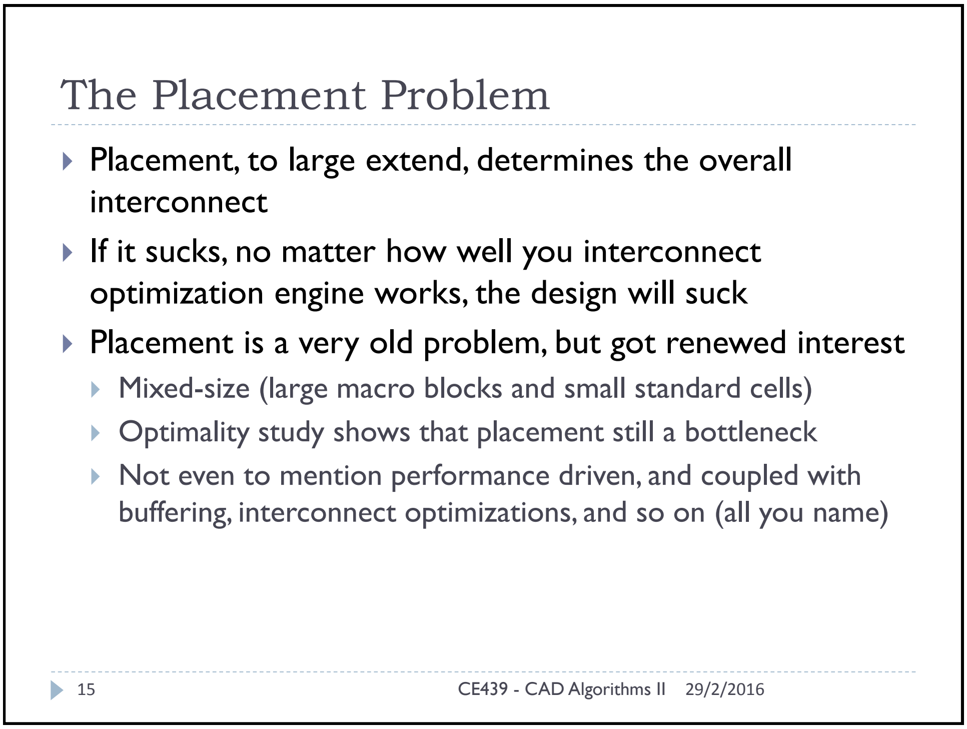
- Many existing solutions are still very suboptimal
  - E.g., placement
- Interconnect dominates
  - No physical layout, no accurate interconnect
- More new physical and manufacturing effects pop up
  - Crosstalk noise, …
  - OPC (manufacturability), etc.
- More vertical integration needed
- Physical design is the KEY linking step between higher level planning/optimization and lower level modeling

Technology Trends and Challenges



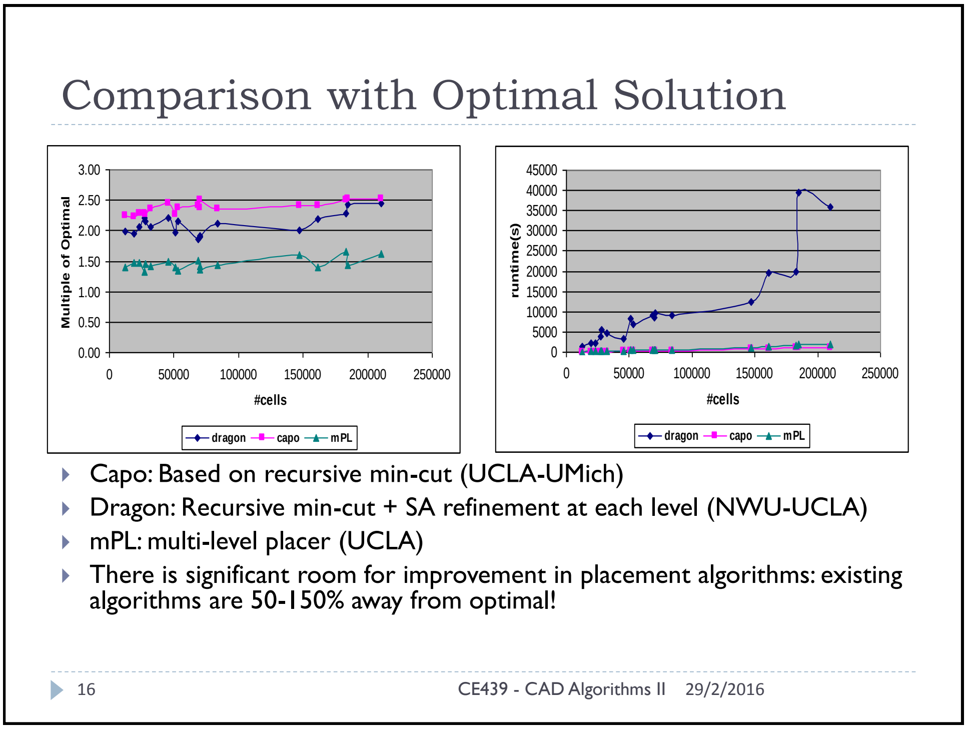
Source: ITRS

- Interconnect determines the overall performance
- In addition: noise, power => Design closure
- Furthermore: manufacturability => Manufacturing closure



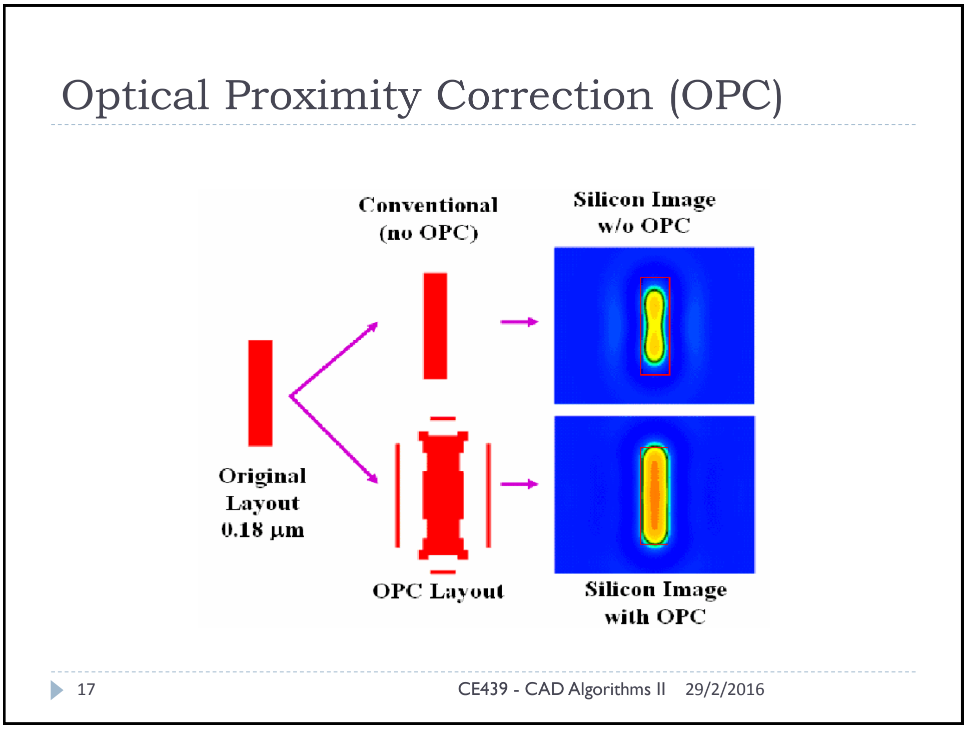
The Placement Problem

- Placement, to large extend, determines the overall interconnect
- If it sucks, no matter how well you interconnect optimization engine works, the design will suck
- Placement is a very old problem, but got renewed interest
  - Mixed-size (large macro blocks and small standard cells)
  - Optimality study shows that placement still a bottleneck
  - Not even to mention performance driven, and coupled with buffering, interconnect optimizations, and so on (all you name)

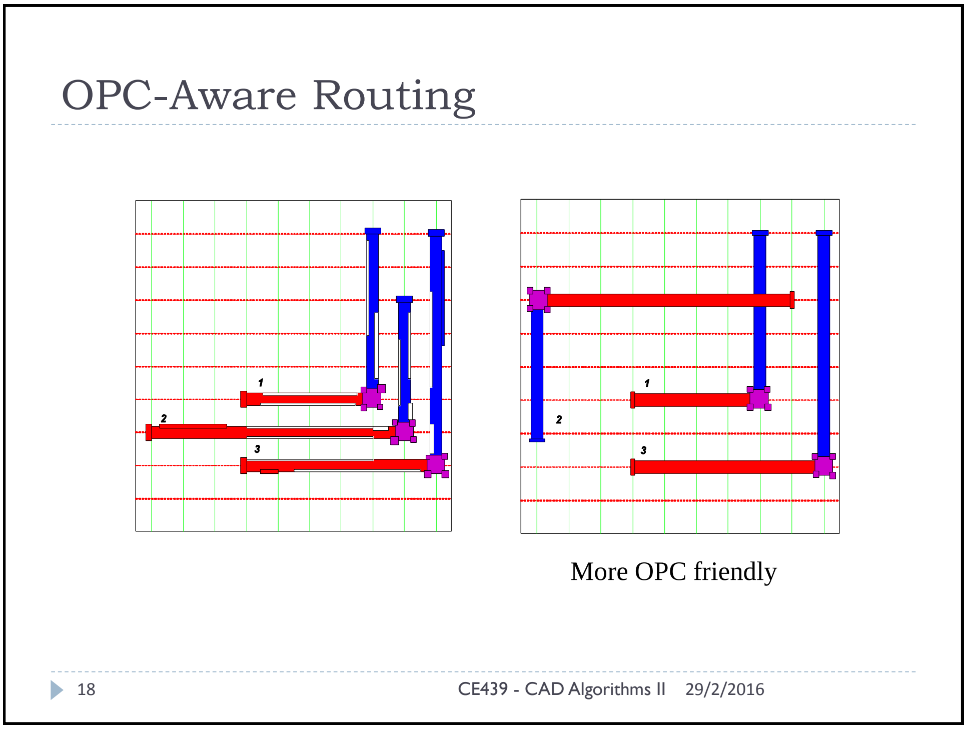


Comparison with Optimal Solution

- Capo: Based on recursive min-cut (UCLA-UMich)
- Dragon: Recursive min-cut + SA refinement at each level (NWU-UCLA)
- mPL: multi-level placer (UCLA)
- There is significant room for improvement in placement algorithms: existing algorithms are 50-150% away from optimal!



Optical Proximity Correction (OPC)



OPC-Aware Routing

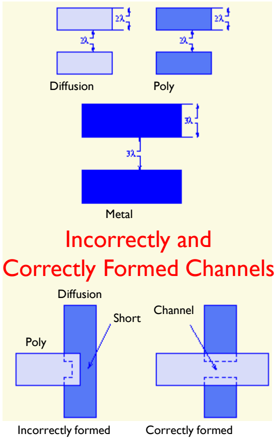
More OPC friendly

Basic Design Rules

- 1. Size Rules
- 2. Separation Rules
- 3. Overlap Rules

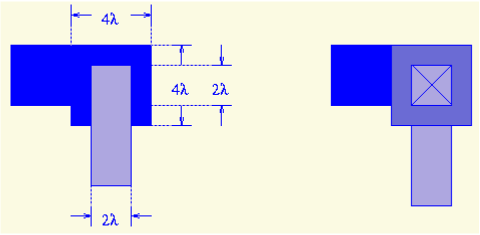
Diffusion Region Width	2λ
Polysilicon Region Width	2λ
Diffusion-Diffusion Spacing	3λ
Poly-Poly Spacing	2λ
Polysilicon Gate Extension	2λ
Contact Extension	λ
Metal Width	3λ

Size, Separation and Overlap Rules



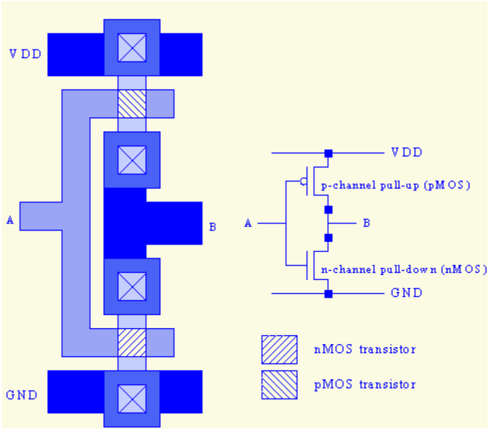
Incorrectly and Correctly Formed Channels

Overlap Rules for Contact Cuts

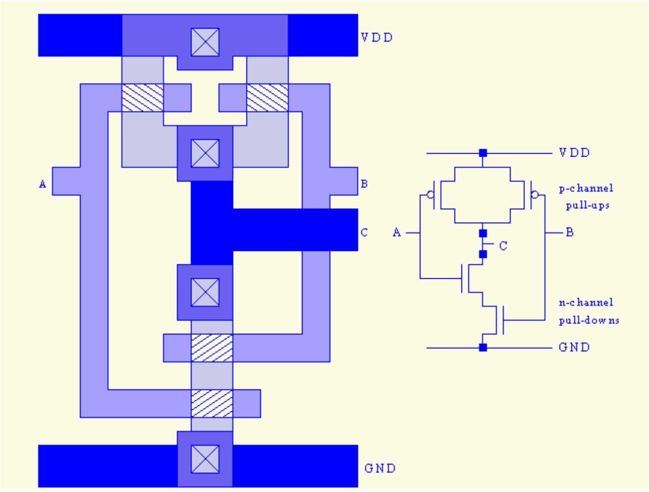


Layout of Basic Devices

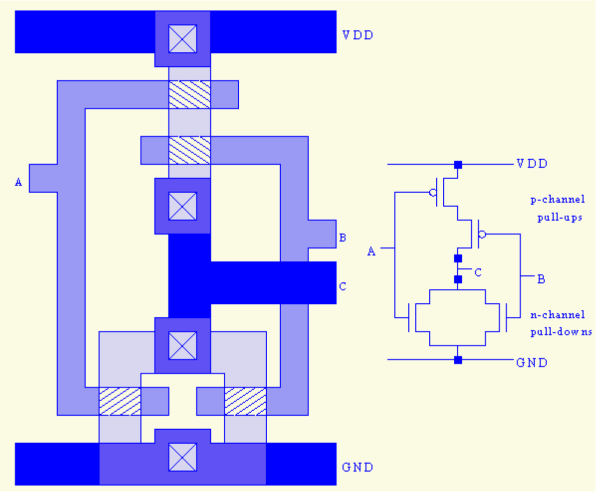
- CMOS Inverter



CMOS NAND Gate



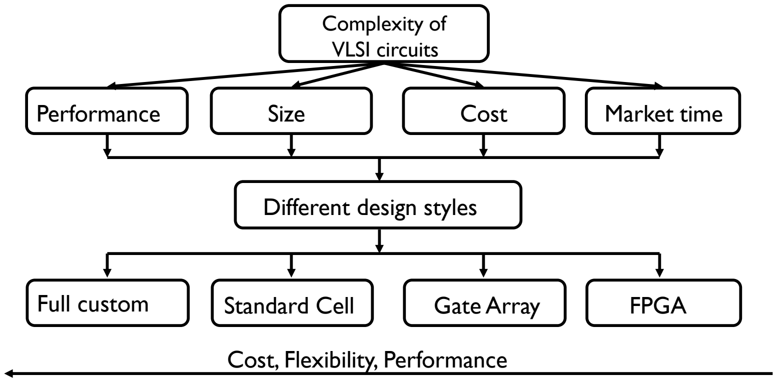
CMOS NOR Gates



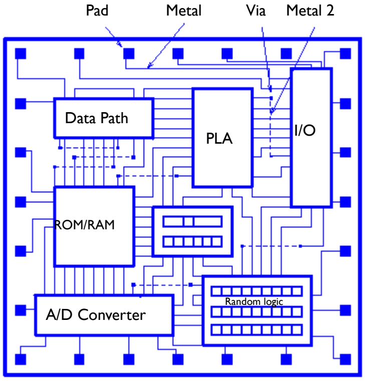
Additional Fabrication Issues

- Scaling
- Parasitic Effects
- Yield Statistics and Fabrication Costs
- Delay Computation
- Noise and Crosstalk
- Power Dissipation

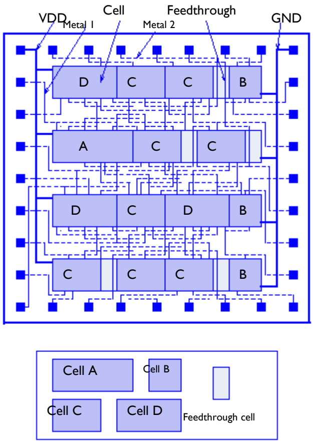
Design Styles



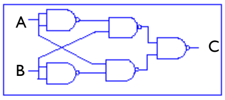
Full Custom Design Style

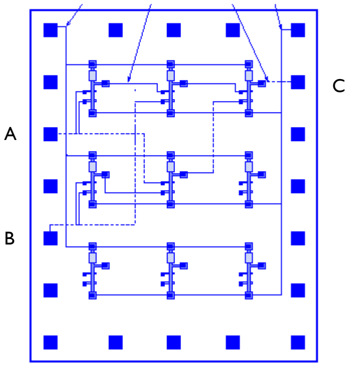


Standard Cell Design Style



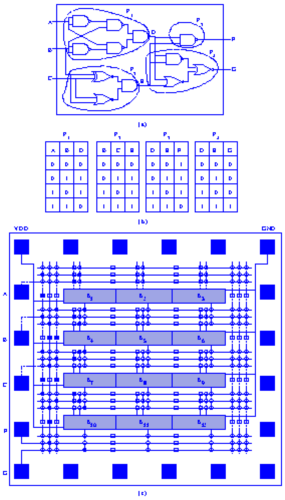
Gate Array Design Style





- Structured ASICs (hot topics nowadays) are essentially gate array

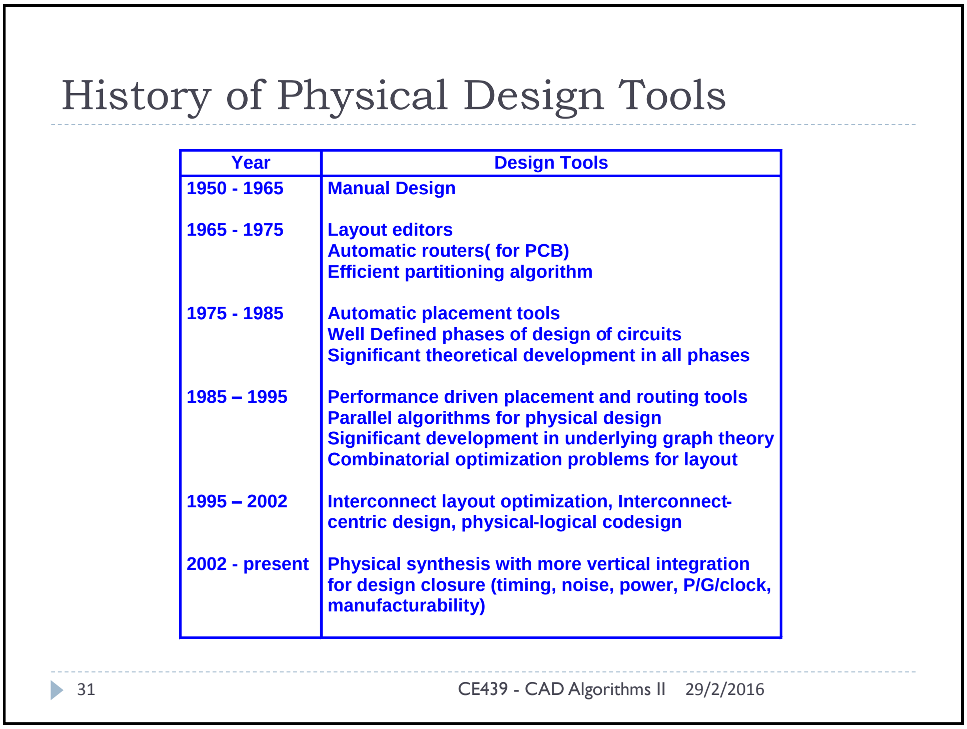
FPGA Design Style



Design Styles Comparison

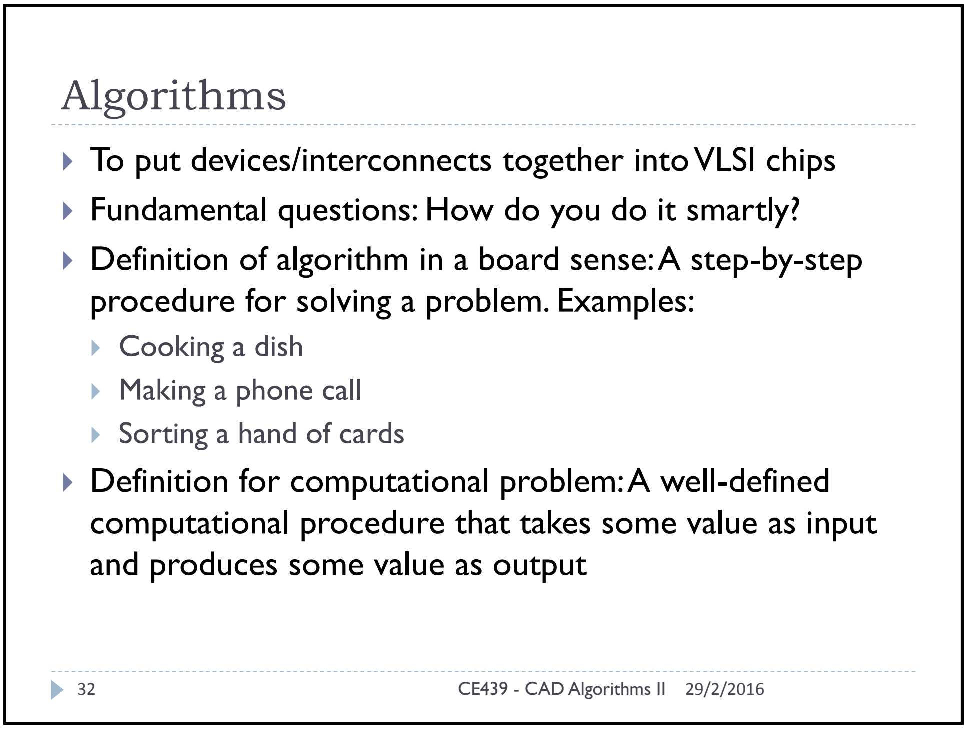
	style			
	full-custom	standard cell	gate array	FPGA
cell size	variable	fixed height *	fixed	fixed
cell type	variable	variable	fixed	programmable
cell placement	variable	in row	fixed	fixed
interconnections	variable	variable	variable	programmable

	style			
	full-custom	standard cell	gate array	FPGA
Area	compact	compact to moderate	moderate	large
Performance	high	high to moderate	moderate	low
Fabrication layers	ALL	ALL	routing layers	none



History of Physical Design Tools

Year	Design Tools
1950 - 1965	Manual Design
1965 - 1975	Layout editors Automatic routers(for PCB) Efficient partitioning algorithm
1975 - 1985	Automatic placement tools Well Defined phases of design of circuits Significant theoretical development in all phases
1985 – 1995	Performance driven placement and routing tools Parallel algorithms for physical design Significant development in underlying graph theory Combinatorial optimization problems for layout
1995 – 2002	Interconnect layout optimization, Interconnect-centric design, physical-logical codesign
2002 - present	Physical synthesis with more vertical integration for design closure (timing, noise, power, P/G/clock, manufacturability)



Algorithms

- To put devices/interconnects together into VLSI chips
- Fundamental questions: How do you do it smartly?
- Definition of algorithm in a board sense: A step-by-step procedure for solving a problem. Examples:
  - Cooking a dish
  - Making a phone call
  - Sorting a hand of cards
- Definition for computational problem: A well-defined computational procedure that takes some value as input and produces some value as output

Some Algorithm Design Techniques

- Greedy
- Divide and Conquer
- Dynamic Programming
- Network Flow
- Mathematical Programming (e.g., linear programming, integer linear programming)

Algorithm Analysis

- There can be many different algorithms to solve the same problem.
- Need some way to compare 2 algorithms.
- Usually run time is the most important criterion used
  - Space (memory) usage is of less concern now
- However, difficult to compare since algorithms may be implemented in different machines, use different languages, etc.
- Also, run time is input-dependent. Which input to use?
- Big-O notation is widely used for asymptotic analysis
  - N→inf

Big-O Notation

- Consider run time for the worst input
  - upper bound on run time.
- Express run time as a function input size n.
- Interested in the run time for large inputs.
- Therefore, interested in the growth rate.
- Ignore multiplicative constant.
- Ignore lower order terms.

- $3n^2+6n+2.7$ is $O(n^2)$.
- $n^{1.1}+10000000000n$ is $O(n^{1.1})$.
- $n^{1.1}$ is also $O(n^2)$, but to be more precise, it is $O(n^{1.1})$

Growth Rates of Some Functions

$$O(\log n) < O(\log^2 n) < O(\sqrt{n}) < O(n)$$
$$< O(n \log n) < O(n \log^2 n) < O(n^{1.5}) < O(n^2)$$
$$< O(n^3) < O(n^4)$$

$$O(n^c) = O(2^{c \log n}) \text{ for any constant } c$$

Polynomial Functions

$$< O(n^{\log n}) = O(2^{\log^2 n})$$
$$< O(2^n) < O(3^n) < O(4^n)$$
$$< O(n!) < O(n^n)$$

Exponential Functions

NP-Complete Problems

- The class NP-Complete is the set of problems which we believe there is no polynomial time algorithms.
- Therefore, it is a class of hard problems.
- NP-Hard is another class of problems containing the class NP-Complete.
- If we know a problem is in NP-Complete or NP-Hard, there is no hope to solve it efficiently.

Algorithm Solution Types

- Polynomial time algorithms
- Exponential time algorithms
- Special case algorithms
- Approximate algorithms
- Heuristic algorithms