943/U0220 & #901/60010

VLSI Design Automation

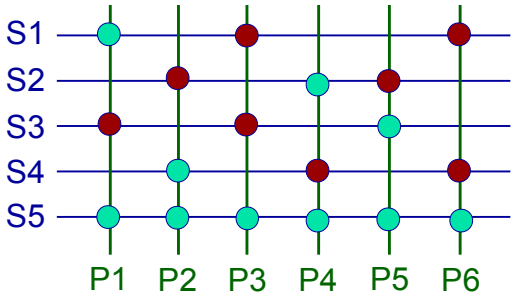
張耀文
Yao-Wen Chang
ywchang@cc.ee.ntu.edu.tw
http://cc.ee.ntu.edu.tw/~ywchang
Graduate Institute of Electronics Engineering
Department of Electrical Engineering
National Taiwan University
Taipei 106, Taiwan
Spring 2004

Administrative Matters

- **Time/Location:** Thursdays 9:10am--12:10pm; EE#2-144.
- **Instructor:** Yao-Wen Chang.
- **E-mail:** ywchang@cc.ee.ntu.edu.tw
- **URL:** http://cc.ee.ntu.edu.tw/~ywchang.
- **Office:** EE#2-548. (Tel) 2363-5251x 548; (Fax) 2364-1972.
- **Office Hours:** Thursdays 2-3pm
- **Teaching Assistants**
  - 陳泰蓁 Tai-Chen Chen (d0943008@ee.ntu.edu.tw)
  - 陳東傑 Tung-Chieh Chen (donnie@eda.ee.ntu.edu.tw)
- **Prerequisites:** data structures (or discrete math) & logic design.
- **Required Text:** S. H. Gerez, *Algorithms for VLSI Design Automation*, John Wiley & Sons, 1999
- **References:** supplementary reading materials will be provided.

Course Objectives

- Study techniques for electronic design automation (EDA), a.k.a. computer-aided design (CAD).
- Study IC technology evolution and their impacts on the development of EDA tools
- **Study problem-solving (-finding) techniques!!!**



Course Contents

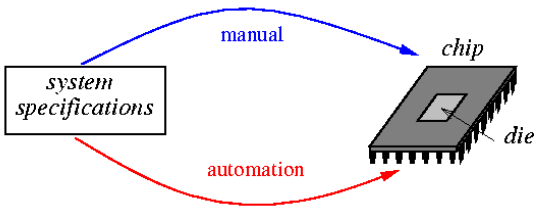
- Introduction to VLSI design flow/styles/automation, technology roadmap, and CMOS Technology (6 hrs)
- (Algorithmic graph theory and computational complexity)* (2 hrs)
- (General-purpose methods for combinatorial optimization)* (1 hr)
- Physical design: partitioning, floorplanning, placement, routing, compaction, deep submicron effects (18 hrs)
- Logic synthesis (6 hrs)
- Formal verification (6 hrs)
- Testing (6 hrs)
- Simulation (3 hrs)
- High-level synthesis (3 hrs)

Grading Policy

- **Grading Policy:**
  - Homework assignments: 25%
  - One in-class open-book, open-note test: 35% (June 17)
  - Programming assignment #1: 20% (due 5pm, April 23)
    - Default programming assignment #1: Any problem of the 2004 MOE IC/CAD contest
    - Contest web site: http://www.cs.nthu.edu.tw/~cad
    - Team work (1--4 persons) is permitted (preferably 2 persons)
  - Programming assignment #2: 20% (due 5pm, June 11)
    - No team work is allowed.
  - Bonus for class participation
- **Homework:** Penalty for late submission: 15% per day.
- **WWW:** http://cc.ee.ntu.edu.tw/~ywchang/Courses/EDA04/eda04.html
- **Academic Honesty:** Avoiding *cheating* at all cost.

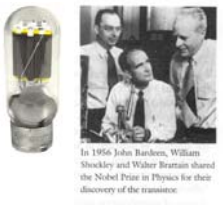
Unit 1: Introduction

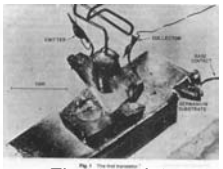
- Course contents:
  - Introduction to VLSI design flow/methodologies/styles
  - Introduction to VLSI design automation tools
  - Semiconductor technology roadmap
  - CMOS technology
- Readings
  - Chapters 1-2
  - Appendix A



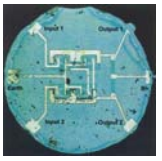
Milestones for IC Industry

- **1947:** Bardeen, Brattain & Shockly invented the transistor, foundation of the IC industry.
- **1952:** SONY introduced the first transistor-based radio.
- **1958:** Kilby invented integrated circuits (ICs).
- **1965:** Moore's law.
- **1968:** Noyce and Moore founded Intel.
- **1970:** Intel introduced 1 K DRAM.




First transistor

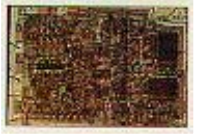

First IC by Kilby


First IC by Noyce

Milestones for IC Industry

- **1971:** Intel announced 4-bit 4004 microprocessors (2250 transistors).
- **1976/81:** Apple II/IBM PC.
- **1984:** Xilinx invented FPGA's.
- **1985:** Intel began focusing on microprocessor products.
- **1987:** TSMC was founded (fabless IC design).
- **1991:** ARM introduced its first embeddable RISC IP core (chipless IC design).


Intel founders


4004

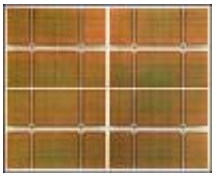

IBM PC

tsmc

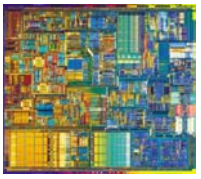
ARM
Advanced RISC Machines

Milestones for IC Industry (Cont'd)

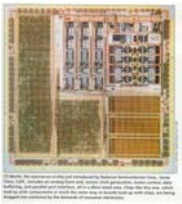
- **1996:** Samsung introduced IG DRAM.
- **1998:** IBM announces1GHz experimental microprocessor.
- **1999/earlier: System-on-Chip (SOC)** applications.
- **2002/earlier: System-in-Package (SIP)** technology.
- An Intel P4 processor contains 42 million transistors (1 billion by 2005)
- Today, we produce > 30 million transistors per person (1 billion/person by 2008).
- **Semiconductor/IC:** #1 key field for advancing into 2000 (*Business Week*, Jan. 1995).



4GB DRAM (2001)



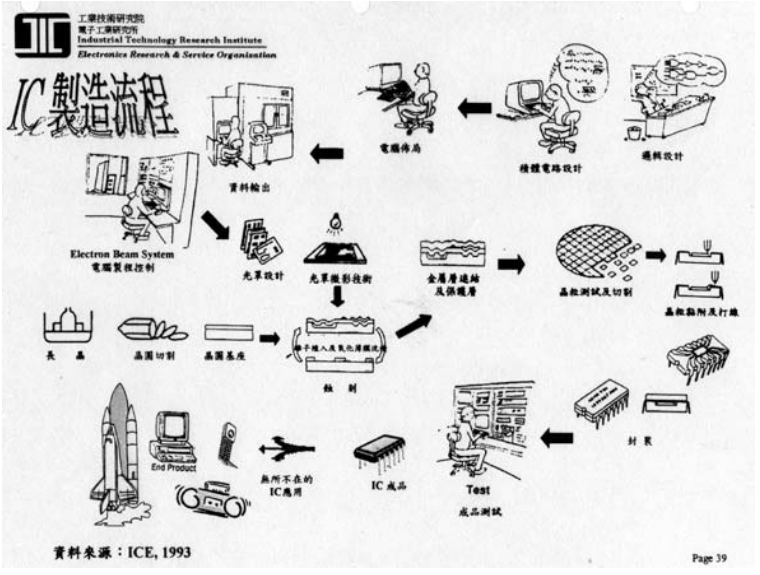
Pentium 4



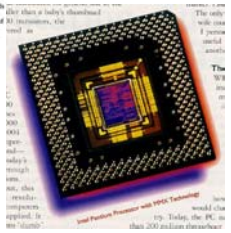
Scanner-on-chip

Blue tooth technology

IC Design & Manufacturing Process



From Wafer to Chip

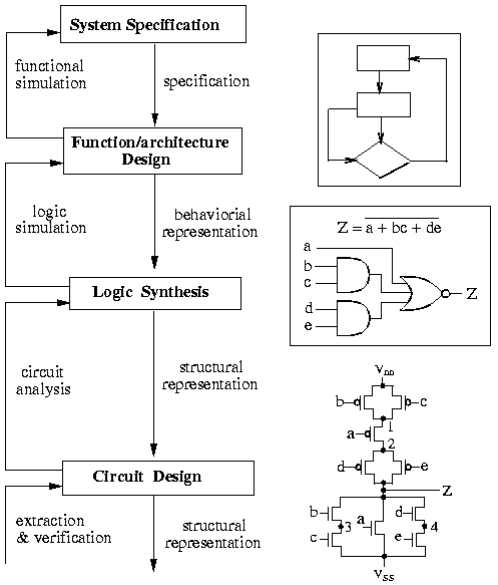


Traditional VLSI Design Cycles

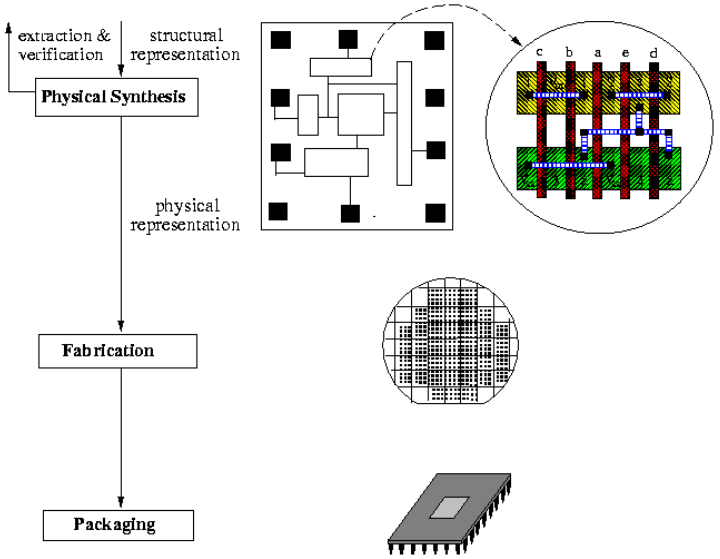
1. System specification
2. Functional design
3. Logic synthesis
4. Circuit design
5. Physical design and verification
6. Fabrication
7. Packaging

- Other tasks involved: testing, simulation, etc.
- Design metrics: area, speed, power dissipation, noise, design time, testability, etc.
- Design revolution: interconnect (not gate) delay dominates circuit performance in deep submicron era.
  - Interconnects are determined in physical design.
  - Shall consider interconnections in early design stages.

Traditional VLSI Design Cycle



Traditional VLSI Design Flow (Cont'd)



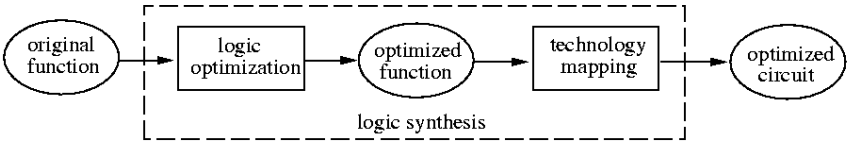
Design Actions

- **Synthesis:** increasing information about the design by providing more detail (e.g., logic synthesis, physical synthesis).
- **Analysis:** collecting information on the quality of the design (e.g., timing analysis).
- **Verification:** checking whether a synthesis step has left the specification intact (e.g., layout verification).
- **Optimization:** increasing the quality of the design by rearrangements in a given description (e.g., logic optimizer, timing optimizer).
- **Design Management:** storage of design data, cooperation between tools, design flow, etc. (e.g., database).

Design Issues and Tools

- System-level design
  - Partitioning into hardware and software, co-design, co-simulation, etc.
  - Cost estimation, design-space exploration
- Algorithmic-level design
  - Behavioral descriptions (e.g. in Verilog, VHDL)
  - High-level simulation
- From algorithms to hardware modules
  - High-level (or architectural) synthesis
- Logic design:
  - Schematic entry
  - Register-transfer level and logic synthesis
  - Gate-level simulation (functionality, power, etc)
  - Timing analysis
  - Formal verification

Logic Design/Synthesis



- **Logic synthesis** programs transform Boolean expressions into logic gate networks in a particular library.
- Optimization goals: minimize area, delay, power, etc
- **Technology-independent** optimization: logic optimization
  - Optimizes Boolean expression equivalent.
- **Technology-dependent** optimization: **technology mapping/library binding**
  - Maps Boolean expressions into a particular cell library.

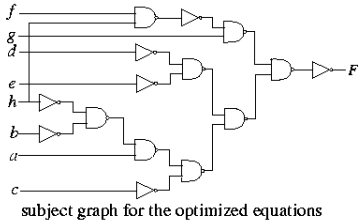
Logic Optimization Examples

- **Two-level:** minimize the # of product terms.
  - $F = \bar{x_1}\bar{x_2}\bar{x_3} + \bar{x_1}\bar{x_2}x_3 + x_1\bar{x_2}\bar{x_3} + x_1\bar{x_2}x_3 + x_1x_2\bar{x_3} \Rightarrow F = \bar{x_2} + x_1\bar{x_3}$.
- **Multi-level:** minimize the #'s of literals, variables.
  - E.g., equations are optimized using a smaller number of literals.

$t1 = a + b\ \ c;$
$t2 = d + e;$
$t3 = a\ \ b + d;$
$t4 = t1\ \ t2 + f\,g;$
$t5 = t4\ \ h + t2\ \ t3;$
$F = t5';$

logic optimization →

$t1 = d + e;$
$t2 = b + h;$
$t3 = a\ \ t2 + c;$
$t4 = t1\ \ t3 + f\,g\,h;$



subject graph for the optimized equations

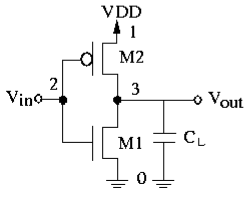
- Methods/CAD tools: Quine-McCluskey method (exponential-time exact algorithm), Espresso (heuristics for two-level logic), MIS (heuristics for multi-level logic), Synopsys, etc.

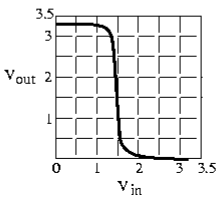
Design Issues and Tools (Cont'd)

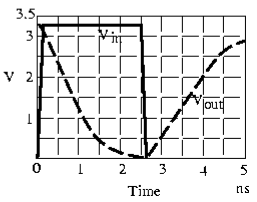
- Transistor-level design
  - Switch-level simulation
  - Circuit simulation
- Physical (layout) design:
  - Partitioning
  - Floorplanning and Placement
  - Routing
  - Layout editing and compaction
  - Design-rule checking
  - Layout extraction
- Design management
  - Data bases, frameworks, etc.
- Silicon compilation: *from algorithm to mask patterns*
  - The *idea* is approached more and more, but still far away from a single *push-buttom* operation

Circuit Simulation of a CMOS Inverter (0.6 μm)

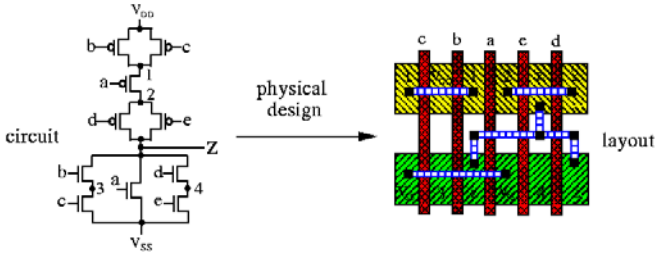
```
M1 3 2 0 0 nch W=1.2u L=0.6u AS=2.16p PS=4.8u AD=2.16p PD=4.8u
M2 3 2 1 1 pch W=1.8u L=0.6u AS=3.24p PS=5.4u AD=3.24p PD=5.4u
CL 3 0 0.2pF

VDD 1 0 3.3
VIN 2 0 DC 0 PULSE (0 3.3 0ns 100ps 100ps 2.4ns 5ns)

.LIB '../mod_06' typical

.OPTION NOMOD POST INGOLD=2 NUMDGT=6 BRIEF
.DC VIN 0V 3.3V 0.001V
.PRINT DC V(3)
.TRAN 0.001N 5N
.PRINT TRAN V(2) V(3)
.END
```





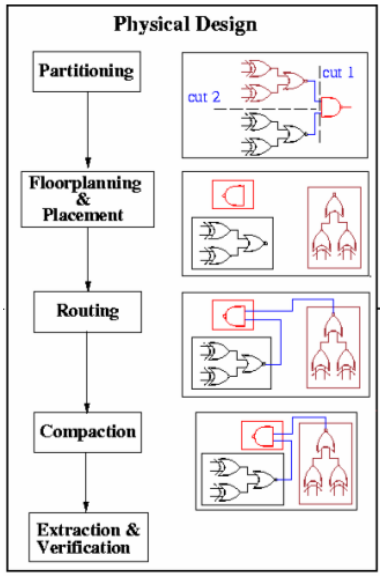


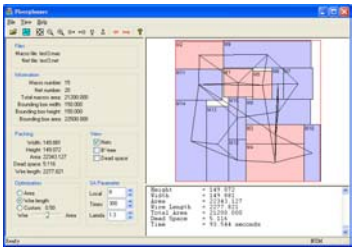
Physical Design



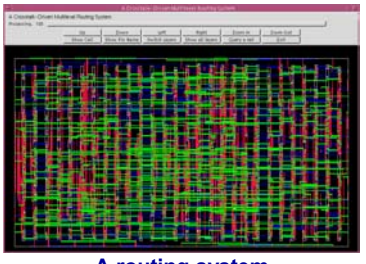
- Physical design converts a circuit description into a geometric description.
- The description is used to manufacture a chip.
- Physical design cycle:
  1. Logic partitioning
  2. Floorplanning and placement
  3. Routing
  4. Compaction
- Others: circuit extraction, timing verification and design rule checking

Physical Design Flow





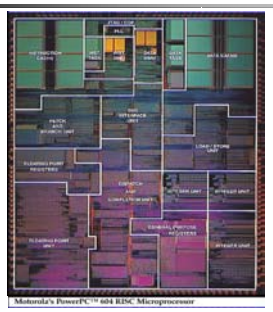
B*-tree based floorplanning system

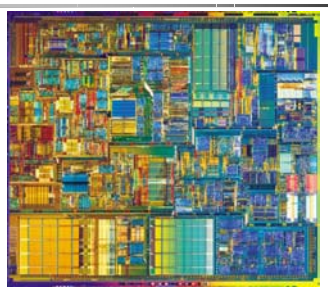


A routing system

Floorplan Examples

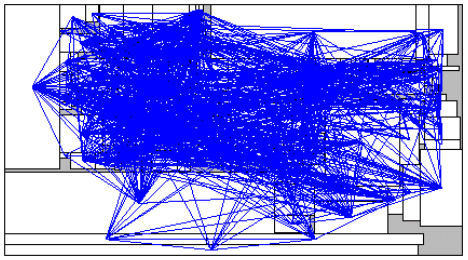
PowerPC 604





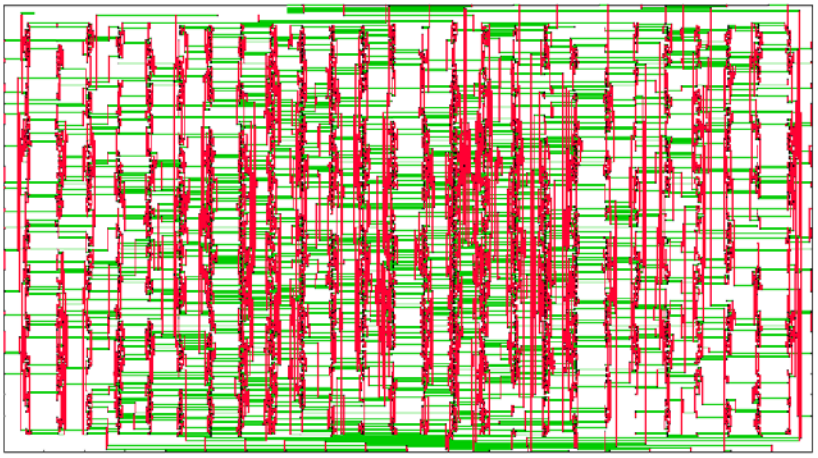
Intel Pentium 4

A floorplan with interconnections

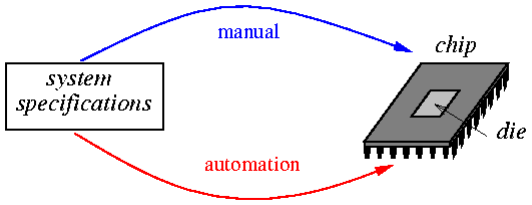


Routing Example

- 0.18um technology, pitch = 1 um, 2774 nets.



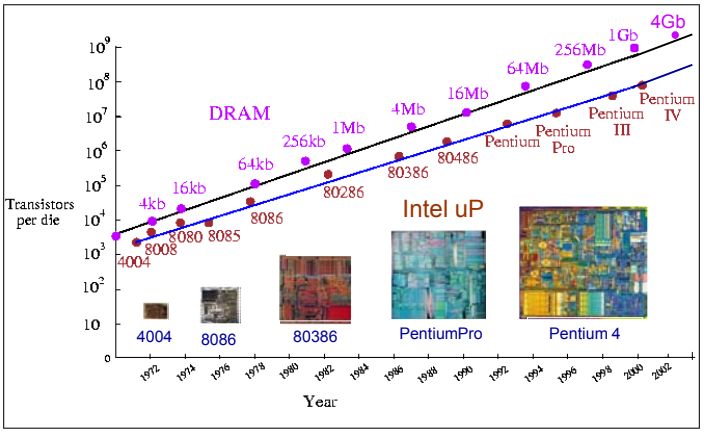
IC Design Considerations



- Several conflicting considerations:
  - **Design Complexity:** large number of devices/transistors
  - **Performance:** optimization requirements for high performance
  - **Time-to-market:** about a 15% gain for early birds
  - **Cost:** die **area**, packaging, testing, etc.
  - Others: power, signal integrity (noise, etc), testability, reliability, manufacturability, etc.

"Moore's" Law: Driving Technology Advances

- Logic capacity doubles per IC at a regular interval.
- Moore: Logic capacity doubles per IC every two years (1975).
- D. House: Computer performance doubles every 18 months (1975)



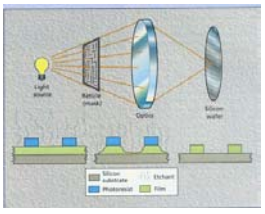
Technology Roadmap for Semiconductors

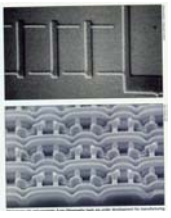
Year	1997	1999	2002	2005	2008	2011	2014
Technology node (nm)	250	180	130	100	70	50	35
On-chip local clock (GHz)	0.75	1.25	2.1	3.5	6.0	10	16.9
Microprocessor chip size (mm^2)	300	340	430	520	620	750	901
Microprocessor transistors/chip	11M	21M	76M	200M	520M	1.40B	3.62B
Microprocessor cost/transistor ($\times 10^{-8}$ USD)	3000	1735	580	255	110	49	22
DRAM bits per chip	256M	1G	4G	16G	64G	256G	1T
Wiring level	6	6–7	7	7–8	8–9	9	10
Supply voltage (V)	1.8–2.5	1.5–1.8	1.2–1.5	0.9–1.2	0.6–0.9	0.5–0.6	0.37–0.42
Power (W)	70	90	130	160	170	175	183

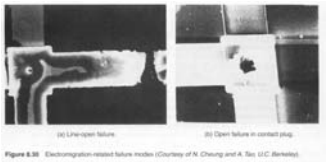
- Source: International Technology Roadmap for Semiconductors (ITRS), Nov. 2002. http://www.itrs.net/ntrs/publntrs.nsf.
- Deep submicron technology: node (**feature size**) < 0.25 μm.
- Nanometer Technology: node < 0.1 μm.

Nanometer Design Challenges

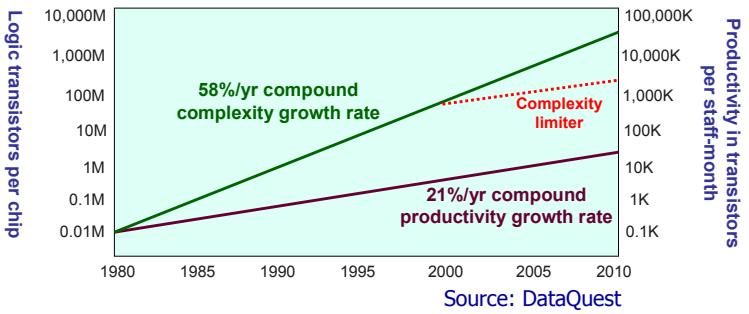
- In 2005, feature size ≈ 0.1 μm, μP frequency ≈ 3.5 GHz, die size ≈ 520 mm^2, μP transistor count per chip ≈ 200M, wiring level ≈ 8 layers, supply voltage ≈ 1 V, power consumption ≈ 160 W.
  - **Feature size ↓ → sub-wavelength lithography (impacts of process variation)? noise? wire coupling? reliability?**
  - **Frequency ↑, dimension ↑ → interconnect delay? electromagnetic field effects? timing closure?**
  - **Chip complexity ↑ → large-scale system design methodology?**
  - **Supply voltage ↓ → signal integrity (noise, IR drop, etc)?**
  - **Wiring level ↑ → manufacturability? 3D layout?**
  - **Power consumption ↑ → power & thermal issues?**







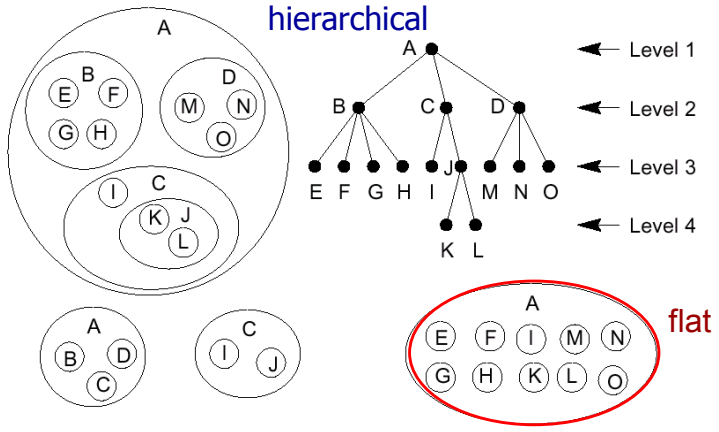
Design Productivity Crisis



- Human factors may limit design more than technology.
- Keys to solve the productivity crisis: CAD (tool & methodology), hierarchical design, abstraction, IP reuse, platform-based design, etc.

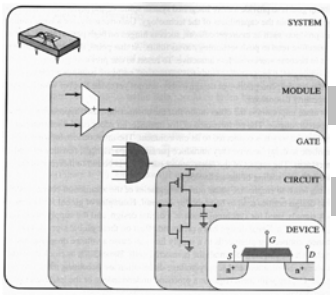
Hierarchical Design

- *Hierarchy:* something is composed of simpler things.
- Design cannot be done in one step ⇒ partition the design hierarchically.



Abstraction

- ***Abstraction:*** when looking at a certain level, you don't need to know all details of the lower levels.



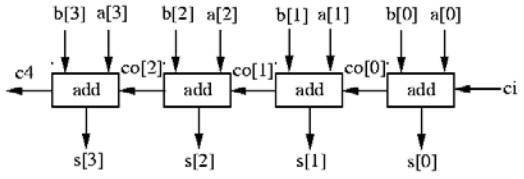
- Design domains:
  - Behavioral: black box view
  - Structural: interconnection of subblocks
  - Physical: layout properties
- Each design domain has its own hierarchy.

Three Design Views

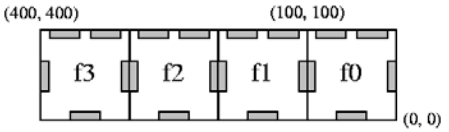
Behavior

```
module add4 (s, c4, ci, a, b);
        input [3:0] a, b;
        input ci;
        output [3:0] s;
        output c4;
        wire [2:0] co;
                add f0 (co[0], s[0], a[0], b[0], ci);
                add f1 (co[1], s[1], a[1], b[1], co[0]);
                add f2 (co[2], s[2], a[2], b[2], co[1]);
                add f3 (c4, s[3], a[3], b[3], co[2]);
endmodule
```

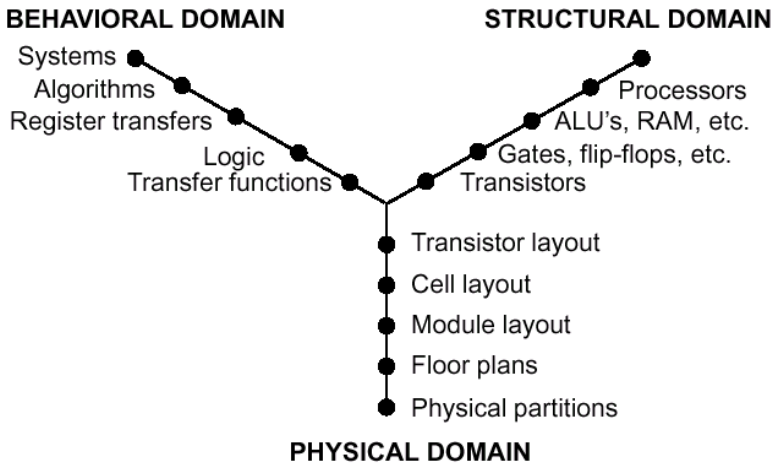
Structural



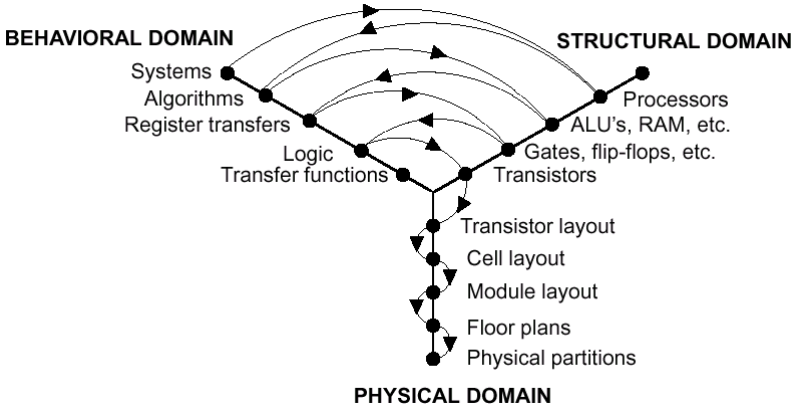
Physical



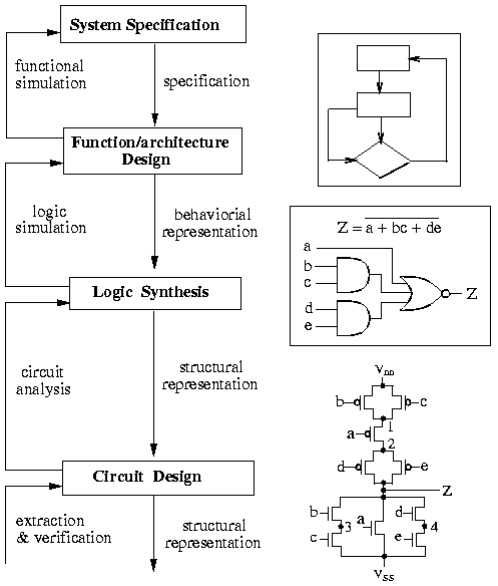
Gajski's Y-Chart



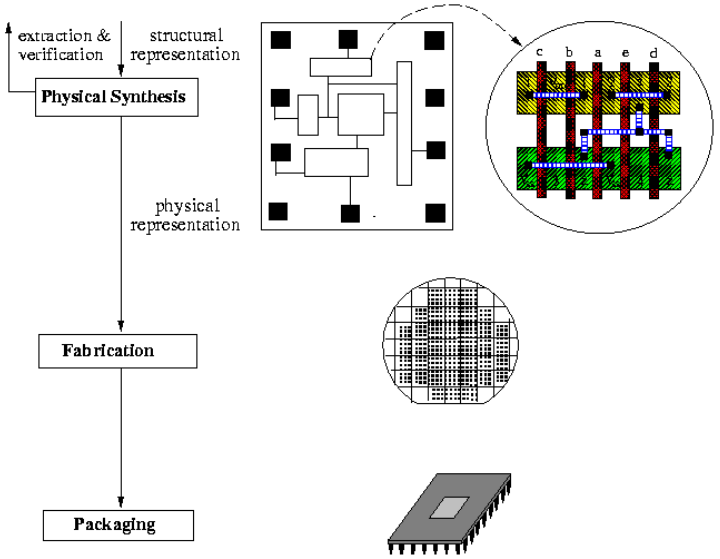
Top-Down Structural Design



Traditional VLSI Design Flow Revisited

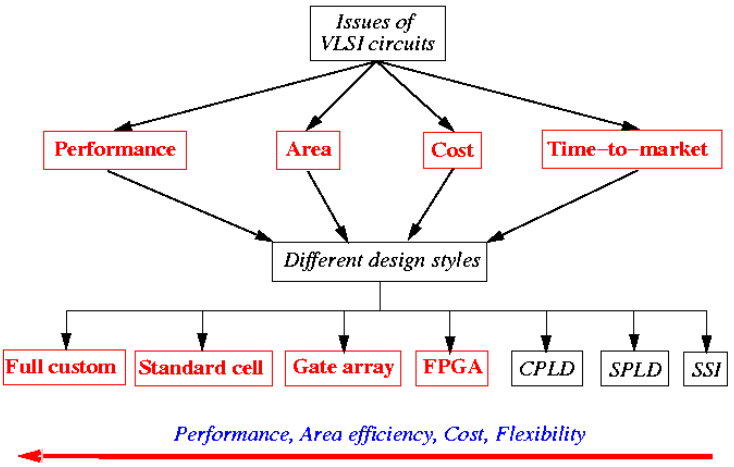


Traditional VLSI Design Flow Revisited (Cont'd)



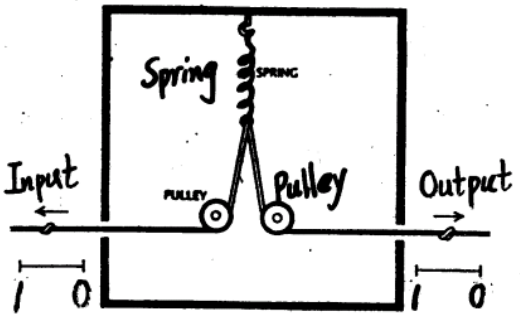
Design Styles

- Specific design styles shall require specific CAD tools



An Ancient Way of Digital Design (1/3)

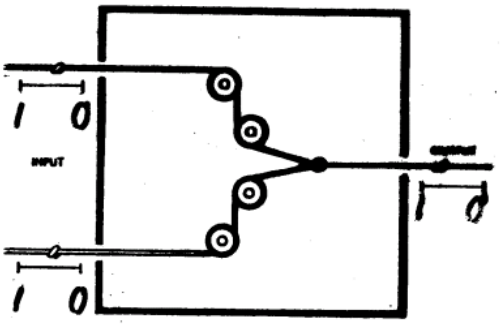
- An ancient rope-and-pulley "computer" on the island of Apraphul (≈ A.D. 850).
  - Joke by A. K. Dewdney, "Computer recreations," *Scientific American*, April 1988.



inverter

An Ancient Way of Digital Design (2/3)

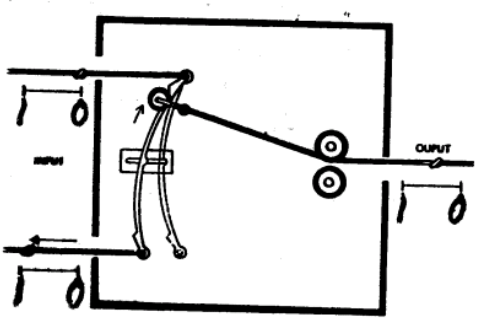
- What is this??



OR gate

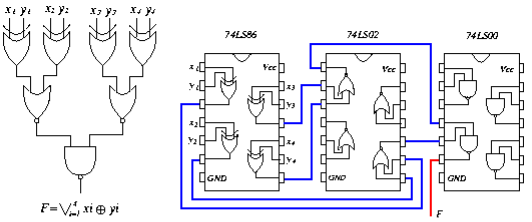
An Ancient Way of Digital Design (3/3)

- What is this??

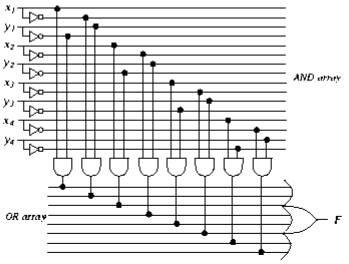


AND gate

SSI/SPLD Design Style



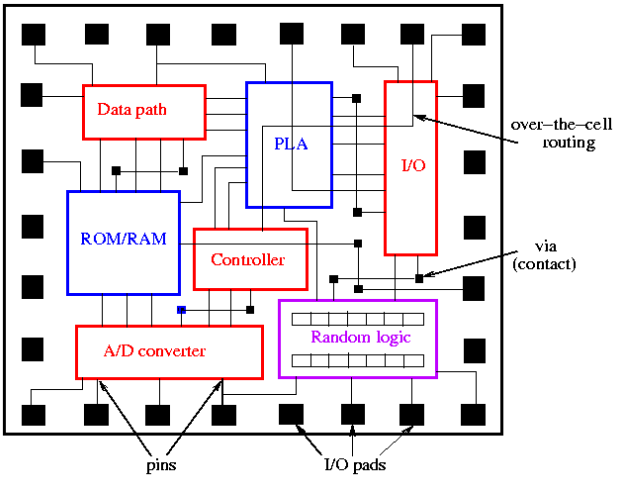
(a) 4–bit comparator. (b) SSI implementation.



(c) SPLD (PLA) implementation.

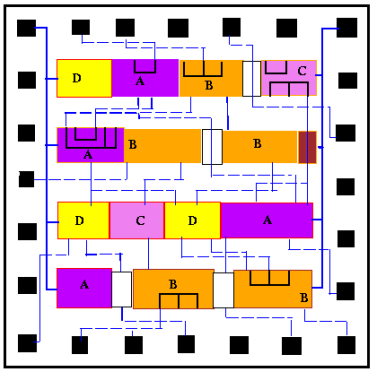
Full Custom Design Style

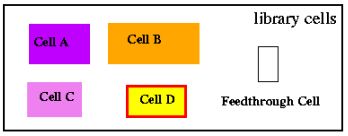
- Designers can control the shape of all mask patterns.
- Designers can specify the design up to the level of individual transistors.



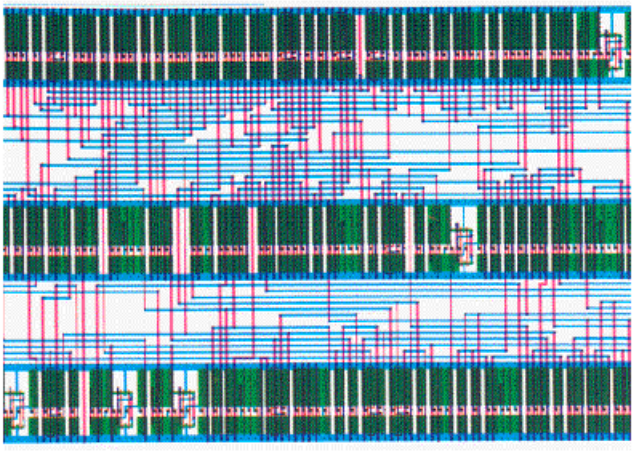
Standard Cell Design Style

- Selects pre-designed cells (of same height) to implement logic





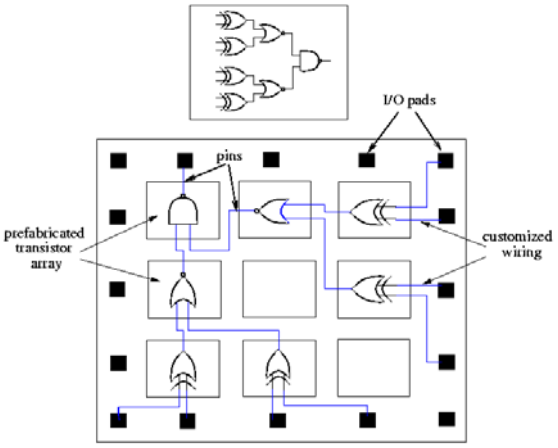
Standard Cell Example



Courtesy Newton/Pister, UC-Berkeley

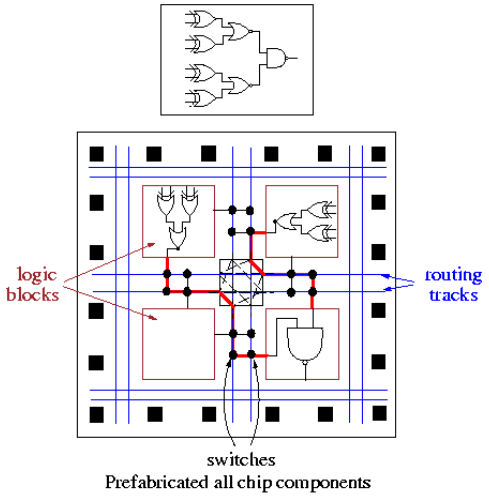
Gate Array Design Style

- Prefabricates a transistor array
- Needs wiring customization to implement logic

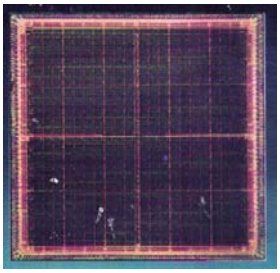


FPGA Design Style

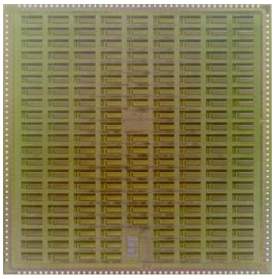
- Logic and interconnects are both prefabricated.
- Illustrated by a symmetric array-based FPGA



Array-Based FPGA Example



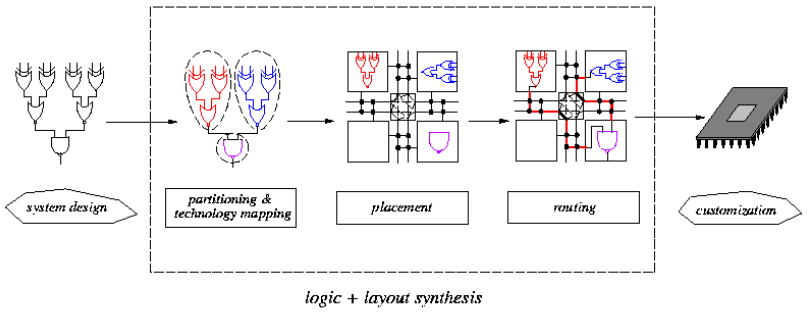
Lucent Technologies 15K ORCA FPGA, 1995
- 0.5 um 3LM CMOS
- 2.45 M Transistors
- 1600 Flip-flops
- 25K bit user RAM
- 320 I/Os



Fujitsu's non-volatile Dynamically Programmable Gate Array (DPGA), 2002

FPGA Design Process

- Illustrated by a symmetric array-based FPGA
- No fabrication is needed



Comparisons of Design Styles

	Full custom	Standard cell	Gate array	FPGA	SPLD
Cell size	variable	fixed height*	fixed	fixed	fixed
Cell type	variable	variable	fixed	programmable	programmable
Cell placement	variable	in row	fixed	fixed	fixed
Interconnections	variable	variable	variable	programmable	programmable

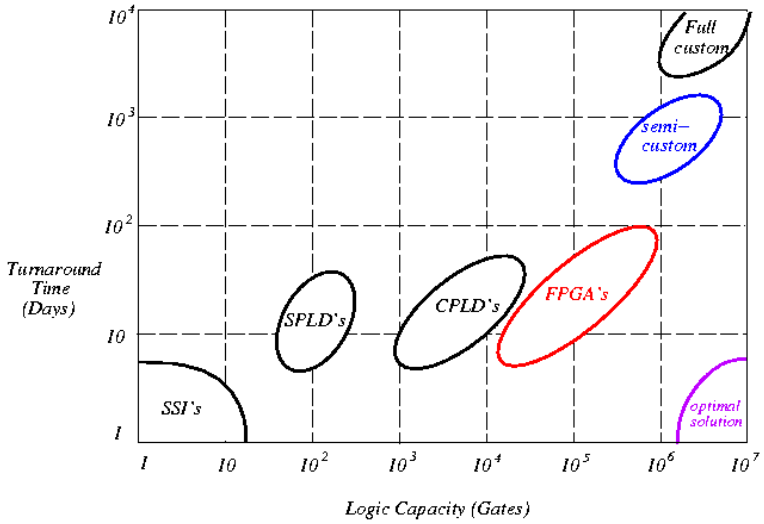
* Uneven height cells are also used.

Comparisons of Design Styles

	Full custom	Standard cell	Gate array	FPGA	SPLD
Fabrication time	− − −	− −	+	+++	++
Packing density	+++	++	+	− −	− − −
Unit cost in large quantity	+++	++	+	− −	−
Unit cost in small quantity	− − −	− −	+	+++	++
Easy design and simulation	− − −	− −	−	++	+
Easy design change	− − −	− −	−	++	++
Accuracy of timing simulation	−	−	−	+	++
Chip speed	+++	++	+	−	−−

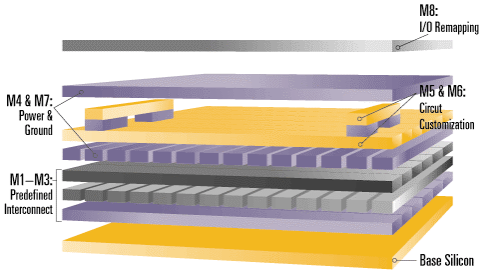
+ desirable; − not desirable

Design Style Trade-offs

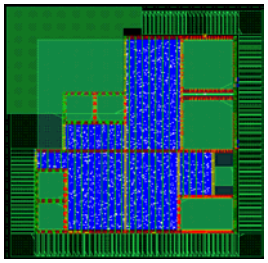


Upcoming: Structured ASIC

- A structured ASIC consists of predefined metal and via layers, as well as a few of them for customization.
- The predefined layers support power distribution and local communications among the building blocks of the device.
- Advantages: fewer masks (lower cost) ; easier physical extraction and analysis.

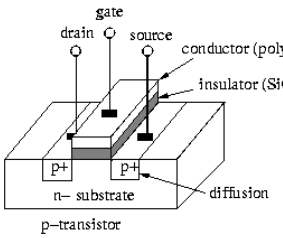


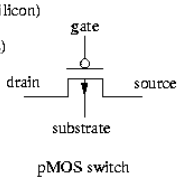
A structured ASIC (M5 & M6 can be customized)



- Green blocks: I/C cells and other hard macro blocks
- Blue area: standard cells
- White points: the spare cells

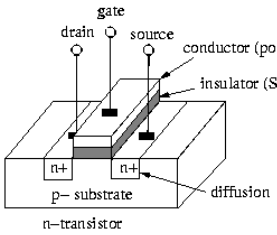
MOS Transistors

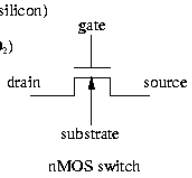




- control input: gate
- switch terminals: drain, source (physically equivalent)
- apply zero voltage to gate ⟹ switch ON

The pMOS switch passes signal "1" well.



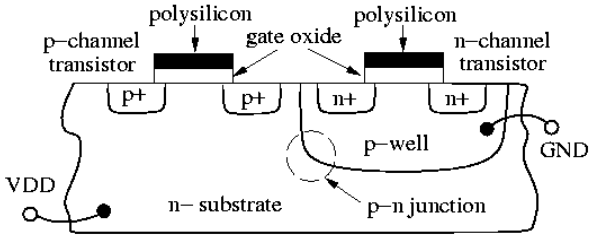


- apply > threshold voltage to gate ⟹ switch ON

The nMOS switch passes signal "0" well.

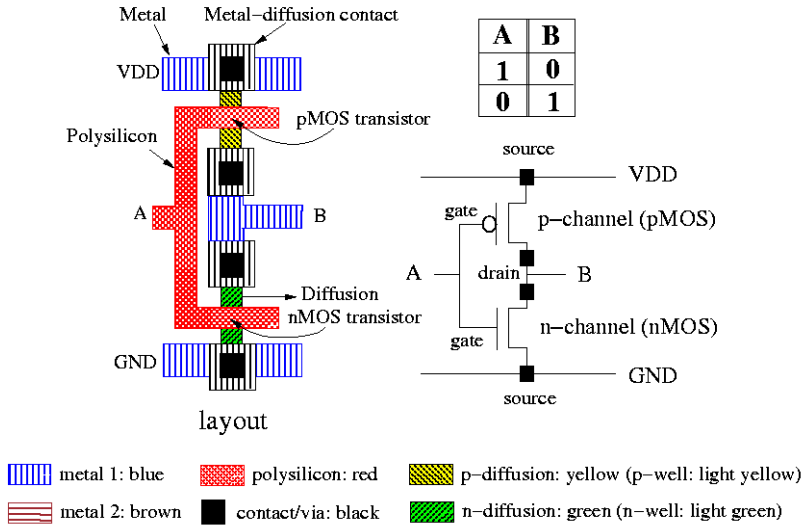
Complementary MOS (CMOS)

- The most popular VLSI technology (v.s. BiCMOS, nMOS).
- CMOS uses both *n*-channel and *p*-channel transistors.
- Advantages: lower power dissipation, higher regularity, more reliable performance, higher noise margin, larger fanout, etc.
- Each type of transistor must sit in a material of the complementary type (the reverse-biased diodes prevent unwanted current flow).



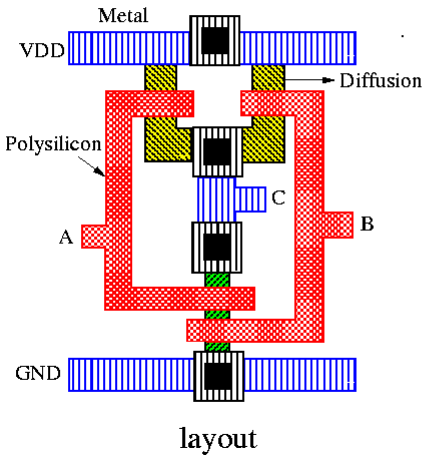
p–well CMOS structure

A CMOS Inverter

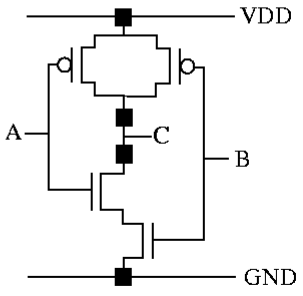


A	B
1	0
0	1

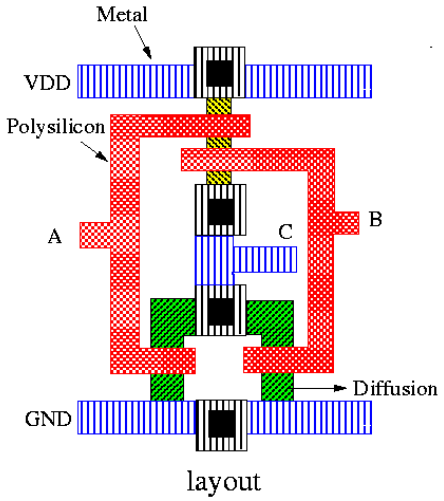
A CMOS NAND Gate



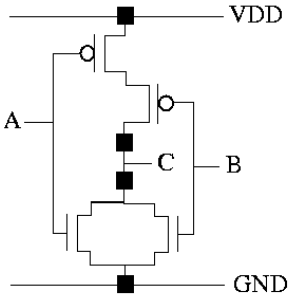
in1 A	in2 B	out C
0	0	1
0	1	1
1	0	1
1	1	0



A CMOS NOR Gate



A	B	C
0	0	1
0	1	0
1	0	0
1	1	0

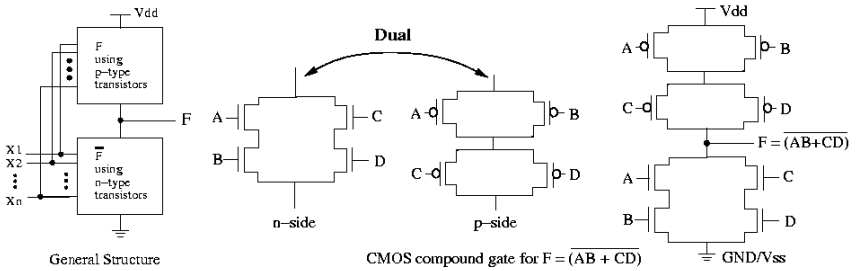


Basic CMOS Logic Library

Name	Distinctive shape	Algebraic equation	Cost (# of transistors)	Scaled gate delay (ps)
AND	X, Y → F	F=XY	6	24
OR	X, Y → F	F=X+Y	6	24
NOT (inverter/ repeater)	X → F	$F=\overline{X}$	2	10
Buffer (driver/ repeater)	X → F	F=X	4	20
NAND	X, Y → F	$F=\overline{XY}$	4	14
NOR	X, Y → F	$F=\overline{X+Y}$	4	14
Exclusive–OR (XOR)	X, Y → F	$F=X\overline{Y}+\overline{X}Y$ $=X\oplus Y$	14	42

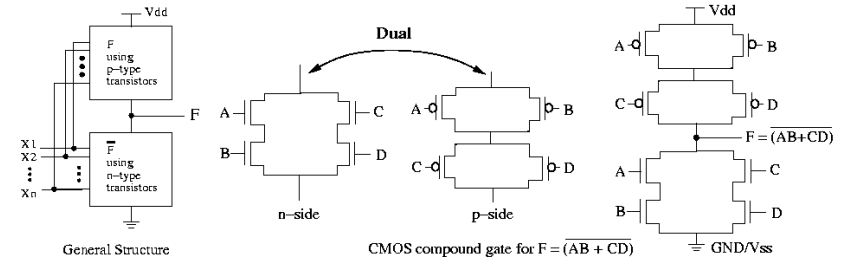
Construction of Compound Gates

- Example: $F = \overline{A \cdot B + C \cdot D}$.
- Step 1 (**n**-network): **Invert** *F* to derive *n*-network $(\overline{F} = A \cdot B + C \cdot D)$
- Step 2 (**n**-network): Make connections of transistors:
  - AND ⇔ Series connection
  - OR ⇔ Parallel connection



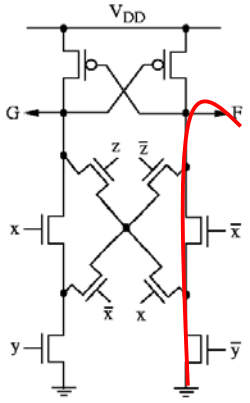
Construction of Compound Gates (cont'd)

- Step 3 (**p**-network): Expand *F* to derive *p*-network
  - $(F = \overline{AB + CD} = \overline{AB} \cdot \overline{CD} = (\overline{A} + \overline{B}) \cdot (\overline{C} + \overline{D}))$
  - **each input is inverted**
- Step 4 (**p**-network): Make connections of transistors (same as Step 2).
- Step 5: Connect the *n*-network to GND (typically, 0V) and the *p*-network to VDD (5V, 3.3V, or 2.5V, etc).



A Complex CMOS Gate

- The functions realized by the *n* and *p* networks must be complementary, and one of the networks must conduct for every input combination.
- Duality is not necessary.



x	y	z	F	G
0	0	0	0	1
0	0	1	0	1
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

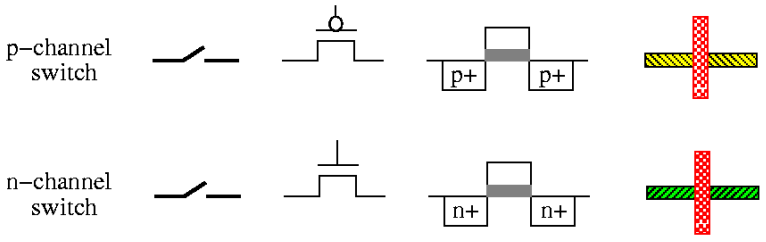
F = ?
G = ?

CMOS Properties

- There is always a path from one supply (VDD or GND) to the output.
- There is never a path from one supply to the other. (This is the basis for the low power dissipation in CMOS--virtually no static power dissipation.)
- There is a momentary drain of current (and thus power consumption) when the gate switches from one state to another.
  - Thus, CMOS circuits have dynamic power dissipation.
  - The amount of power depends on the switching frequency.

Stick Diagram

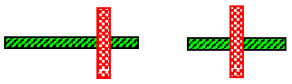
- Intermediate representation between the transistor level and the mask (layout) level.
- Gives topological information (identifies different layers and their relationship)
- Assumes that wires have no width.
- Possible to translate stick diagram automatically to layout with correct **design rules**.



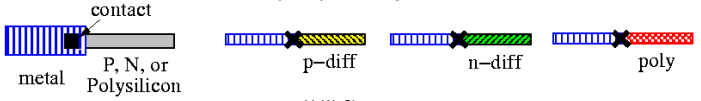
Stick Diagram (cont'd)

- When the same material (on the same layer) touch or cross, they are connected and belong to the same electrical node.

- When **polysilicon** crosses N or P **diffusion**, an N or P transistor is formed.
  - Polysilicon is drawn on top of diffusion.
  - Diffusion must be drawn connecting the source and the drain.
  - Gate is automatically self-aligned during fabrication.

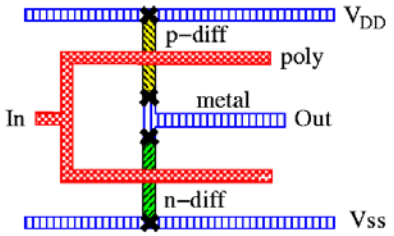


- When a metal line needs to be connected to one of the other three conductors, a **contact** cut (**via**) is required.

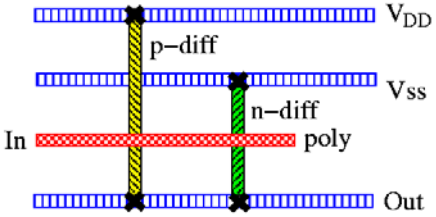


CMOS Inverter Stick Diagrams

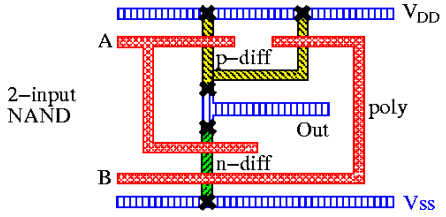
- Basic layout

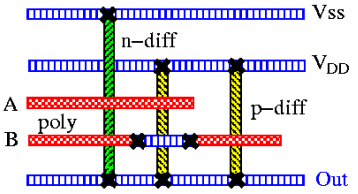


- More area efficient layout

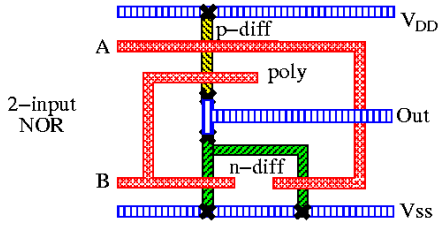


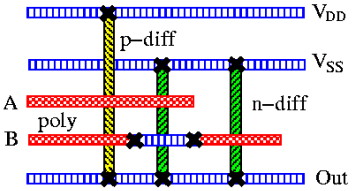
CMOS NAND/NOR Stick Diagrams





area efficient? contacts?

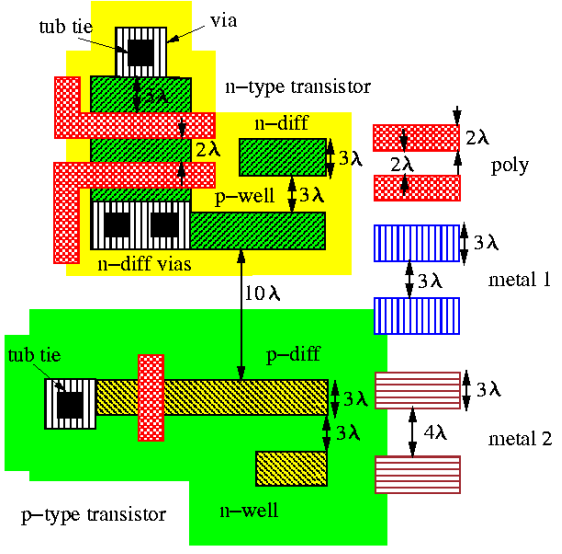




Design Rules

- Layout rules are used for preparing the masks for fabrication.
- Fabrication processes have inherent limitations in accuracy.
- Design rules specify geometry of masks to optimize yield and reliability (trade-offs: area, yield, reliability).
- Three major rules:
  - **Wire width:** Minimum dimension associated with a given feature.
  - **Wire separation:** Allowable separation.
  - **Contact:** overlap rules.
- Two major approaches:
  - **"Micron" rules:** stated at micron resolution.
  - **λ rules:** simplified micron rules with limited **scaling** attributes.
- λ may be viewed as the size of minimum feature.
- Design rules represents a tolerance which insures very high probability of correct fabrication (not a hard boundary between correct and incorrect fabrication).
- Design rules are determined by experience.

SCMOS Design Rules



MOSIS Layout Design Rules

- MOSIS design rules (SCMOS rules) are available at http://www.mosis.org.
- 3 basic design rules: Wire width, wire separation, contact rule.
- MOSIS design rule examples

R1	Min active area width	3 λ
R3	Min poly width	2 λ
R4	Min poly spacing	2 λ
R5	Min gate extension of poly over active	2 λ
R8	Min metal width	3 λ
R9	Min metal spacing	3 λ
R10	Poly contact size	2 λ
R11	Min poly contact spacing	2 λ