Introduction to CAD Techniques for Design and Verification

Prof. Chien-Nan Liu
Institute of Electronics
National Chiao-Tung Univ.

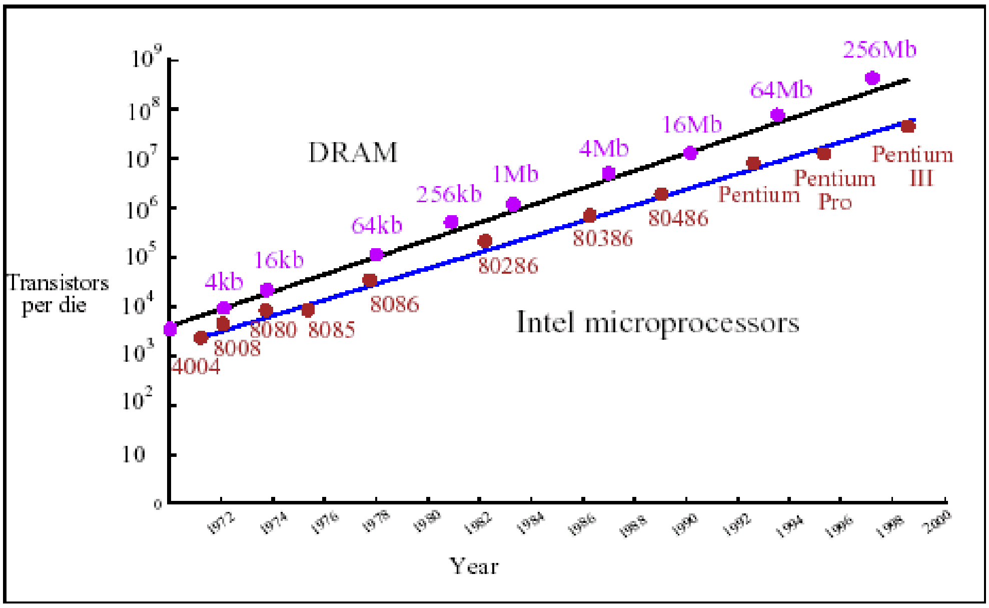
Tel: (03)5712121 ext:31211
E-mail: jimmyliu@nctu.edu.tw
http://www.ee.ncu.edu.tw/~jimmy

Outline

- Moving to SOC
- CAD in SOC Design
- Solutions for Verification Problems
- Solutions for Integration Problems
- Solutions for DSM Problems

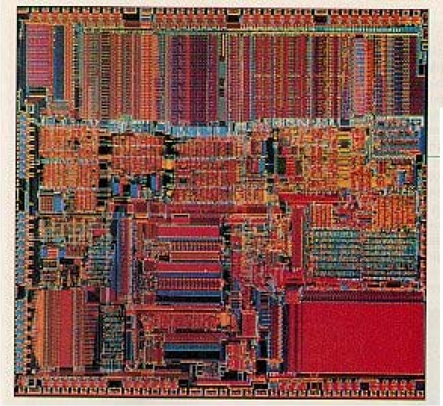
Moore’s Law

- Logic capacity doubles per IC per year at regular intervals (1965)
- Logic capacity doubles per IC every 18 months (1975)

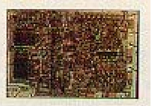


The Dies of Intel CPUs

Pentium Pro

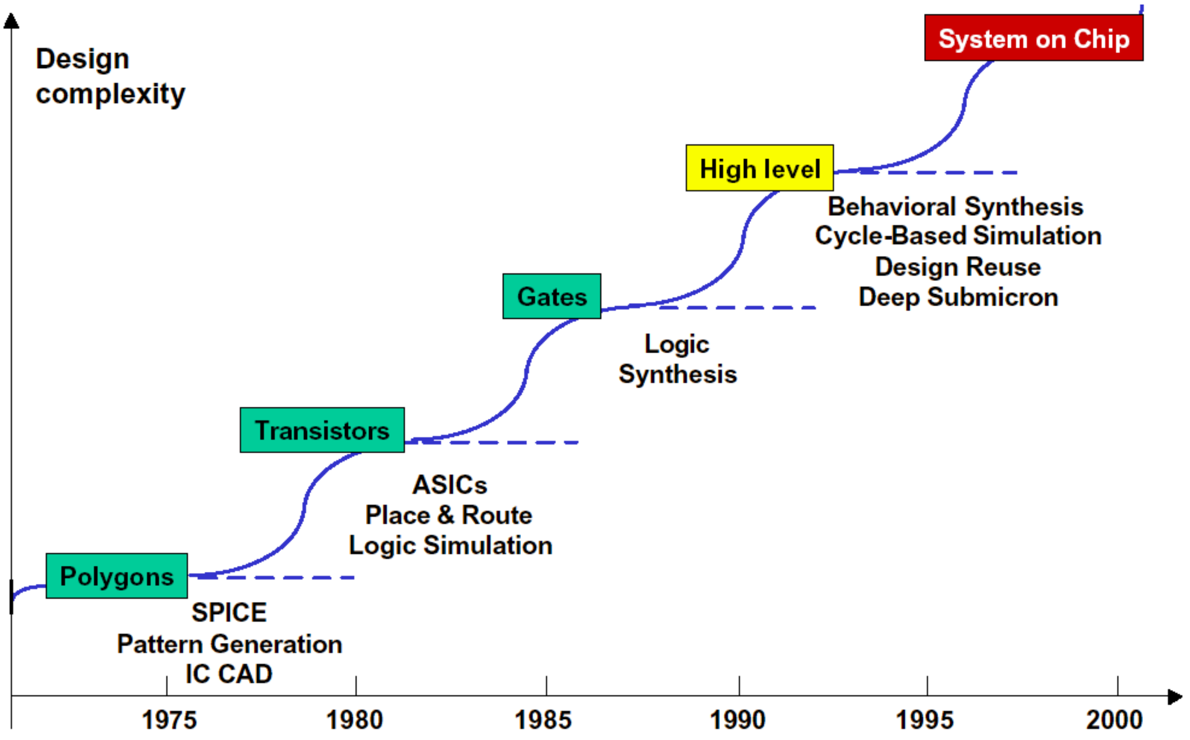


386



4004

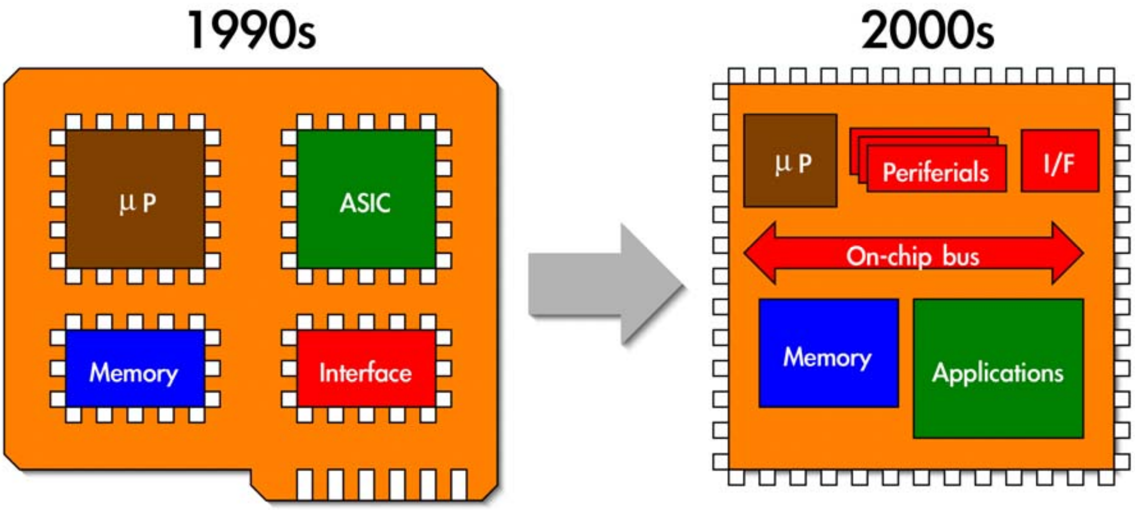
Trends of VLSI Design



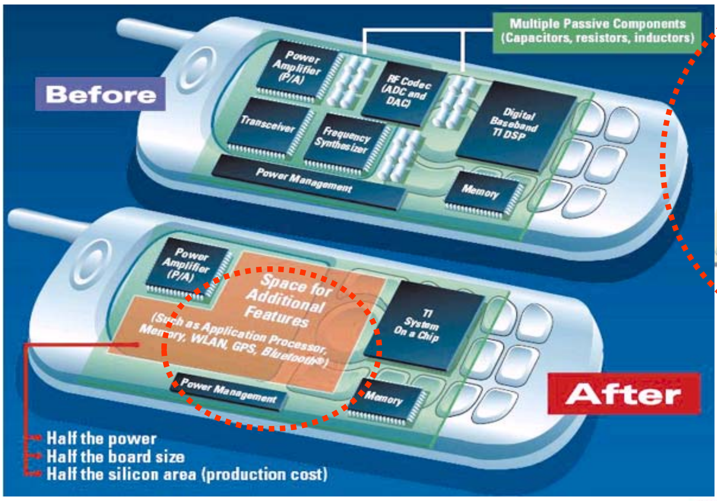
Reference：www.doulos.com

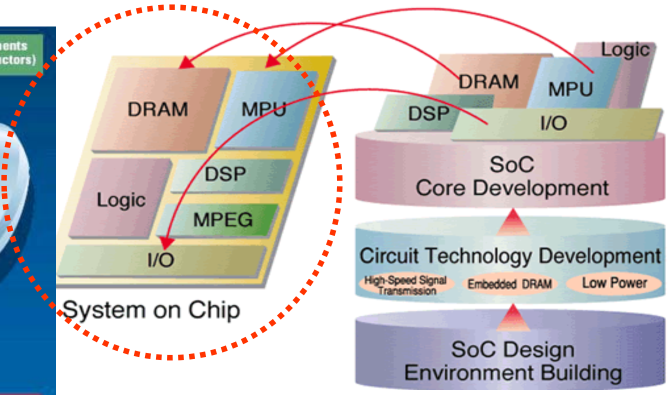
What is SoC?

- System-on-Chip
- An IC that integrates the major functional elements of a complete end-product into a single chip

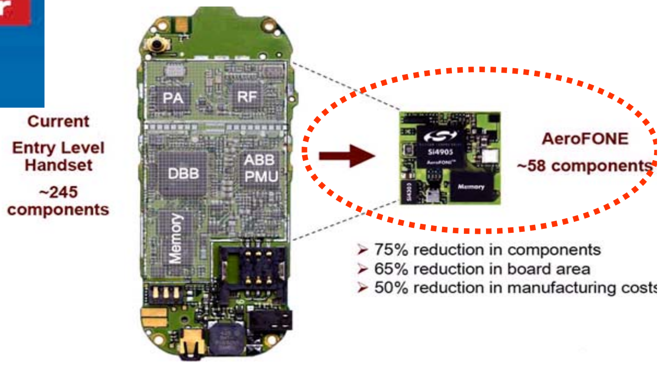


Why Highly Integrated Circuits?





除了體積變小之外，耗電、效能、成本表現均較優



- 75% reduction in components
- 65% reduction in board area
- 50% reduction in manufacturing costs

資料來源： 工研院IEK(2007/09)

SoC Challenges

- Increasing complexity
  - Time-to-market pressure
  - Verification bottleneck
- Integration
  - Hardware v.s. software
  - Digital circuits v.s. analog circuits
  - Testing issues
- Deep submicron effects
  - Timing closure problem
  - Signal integrity problem
  - Reliability problem

Time-to-Market Pressure

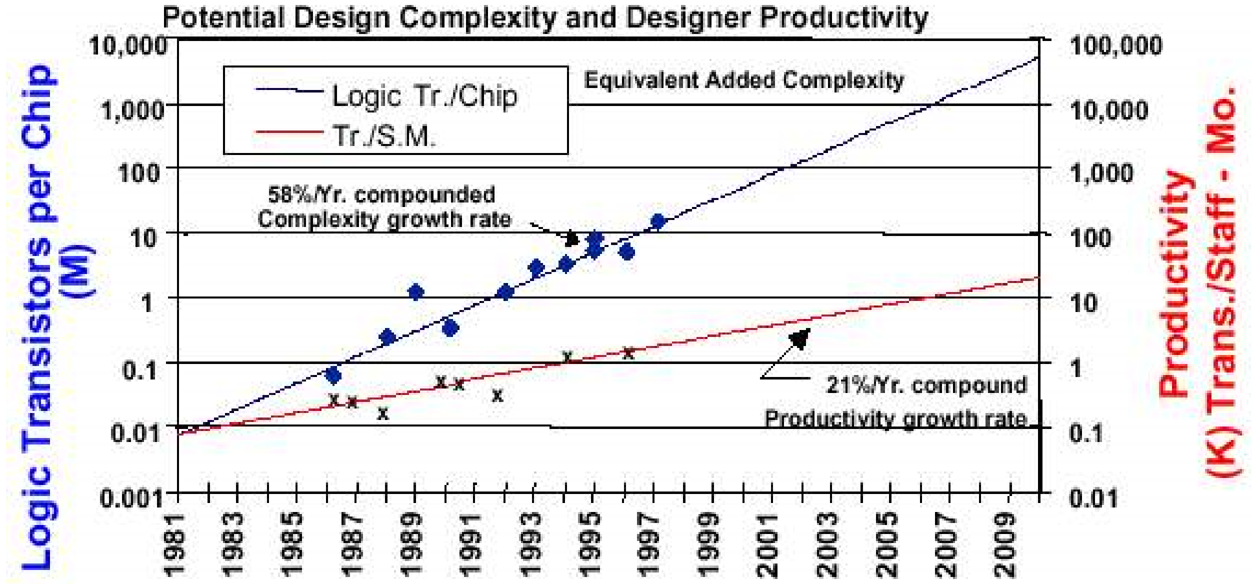
- A lot of pressure from
  - Shorter product lifespan
  - Shrinking design cycles

	1997	1998	1999	2002
Applications	Cellar, PDA, DVD	Set-top boxes, wireless PDA	Internet applications, anything portable	Ubiquitous computing, intelligent, interconnected controllers
Design cycle (month)	18 - 12	12 - 10	10 - 8	8 - 6
Derivative cycle (month)	8 - 6	6 - 4	4 - 2	3 - 2

* Adapted from "Surviving the SOC Revolution."

Productivity Gap

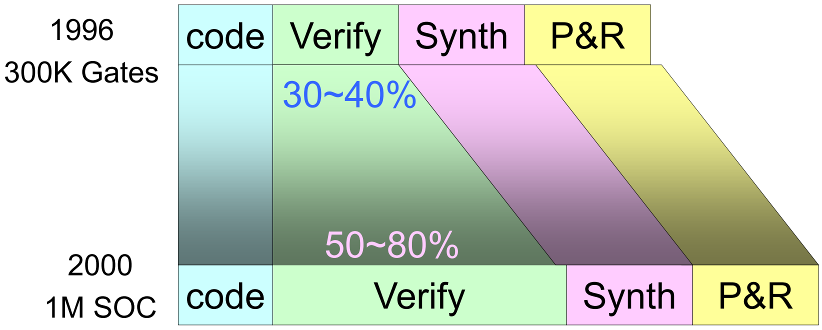
- We do need more productivity !! → **CAD**



Source : ITRS Roadmap 1999 Edition, SIA.

Verification Bottleneck

- Verification becomes the major bottleneck of the modern design flows
  - From 30%-40% to 50%-80%



1996
300K Gates

code | Verify | Synth | P&R

30~40%

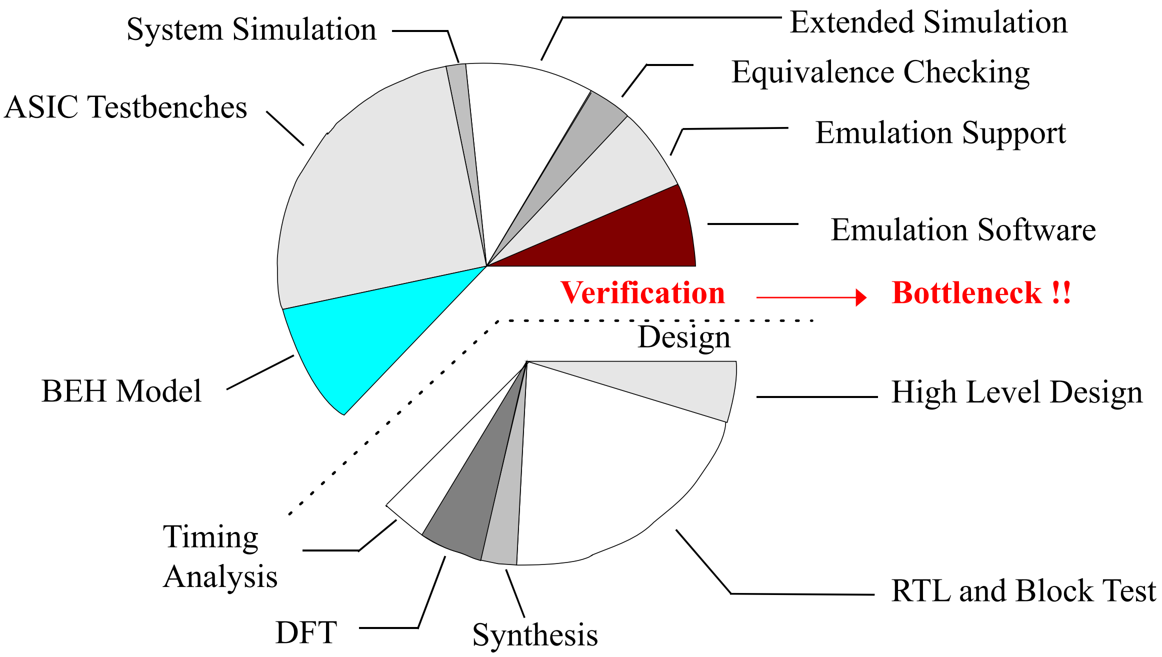
50~80%

2000
1M SOC

code | Verify | Synth | P&R

- An effective verification methodology is also highly desirable

An Industrial Example



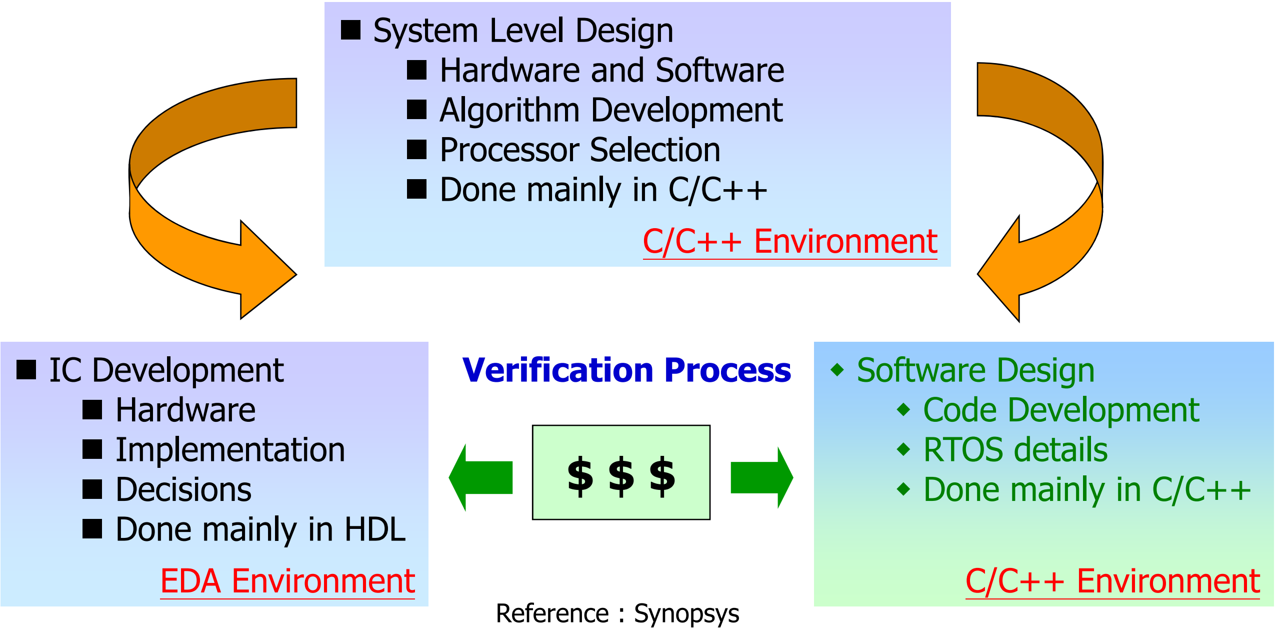
Source : “Functional Verification on Large ASICs” by Adrian Evans, etc., 35th DAC, June 1998.

SoC Challenges

- Increasing complexity
  - Time-to-market pressure
  - Verification bottleneck
- Integration
  - Hardware v.s. software
  - Digital circuits v.s. analog circuits
  - Testing issues
- Deep submicron effects
  - Timing closure problem
  - Signal integrity problem
  - Reliability problem

HW/SW Integration

- Integrating HW/SW at the final step may require high cost to fix inconsistency problems



- System Level Design
  - Hardware and Software
  - Algorithm Development
  - Processor Selection
  - Done mainly in C/C++

C/C++ Environment

- IC Development
  - Hardware
  - Implementation
  - Decisions
  - Done mainly in HDL

EDA Environment

Verification Process

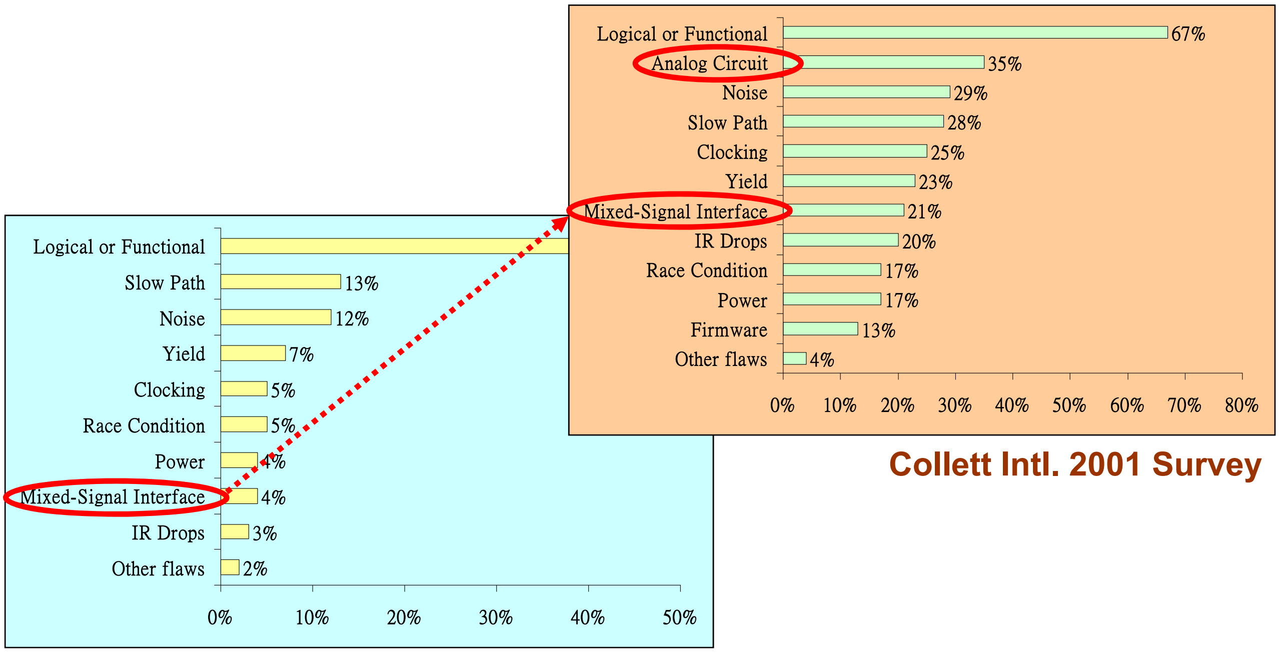
$ $ $

- Software Design
  - Code Development
  - RTOS details
  - Done mainly in C/C++

C/C++ Environment

Reference : Synopsys

Mixed Signals in SoC



Collett Intl. 2001 Survey

Collett Intl. 2000 Survey

Challenges for MS Designs

- Design challenges
  - Chip-level simulation takes too much time
  - Design budgets are not distributed in a well-defined manner
  - Too much time is spent on low-level iterations
  - Design is not completely systematic
  - There is limited or no use of HDL
- Solutions:
  - Use a systematic, top-down design approach to capture design intent
  - Develop some tools for rapid targeting of different design technologies

SoC Testing Challenges

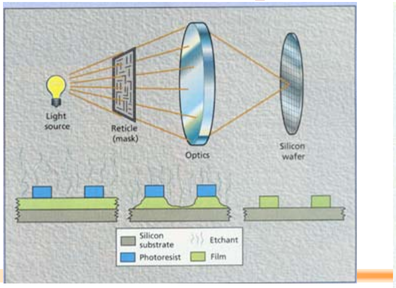
- Distributed design and test
  - Core provider does not know the target environment
  - System integrator is responsible for manufacturing testing
- Test access
  - Difficulties to access deeply embedded cores
  - Bandwidth, I/O pin count limitations
- Test optimization
  - Minimizing test cost while satisfying constraints such as power, resources, coverage, etc.

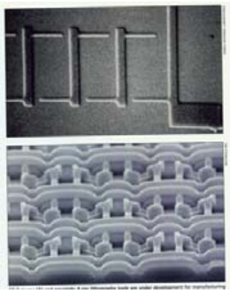
SoC Challenges

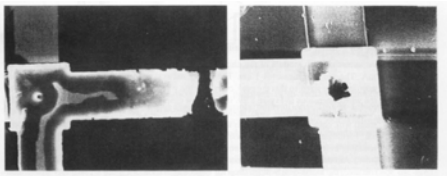
- Increasing complexity
  - Time-to-market pressure
  - Verification bottleneck
- Integration
  - Hardware v.s. software
  - Digital circuits v.s. analog circuits
  - Testing issues
- Deep submicron effects
  - Timing closure problem
  - Signal integrity problem
  - Reliability problem

Nanometer Design Challenges

- In 2005, feature size ≈ 0.1 μm, μ P frequency ≈ 3.5 GHz, die size ≈ 520 mm^2, μ P 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?**





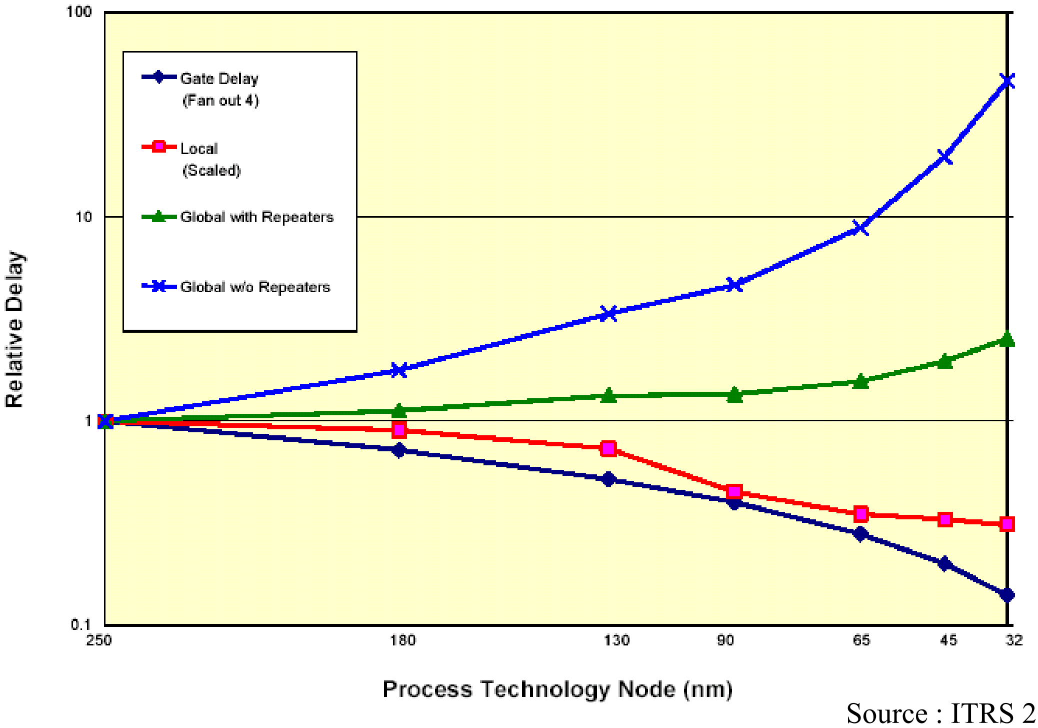


(a) Line-open failure.

(b) Open failure in contact plug.

Figure 8.30 Electromigration-related failure modes (*Courtesy of N. Cheung and A. Tao, U.C. Berkeley*).

Wiring Delay vs. Feature Size



Source : ITRS 2001

Timing Closure Problem

- Wire delay starts to dominate total delay in DSM process
  - Cannot be ignored as taught in digital design course
- Only statistical wire delay model can be used at design phase
  - Lack of physical information about wire length
- Incorrect estimations require long iterations to meeting timing
  - Design schedule will be seriously delayed !!

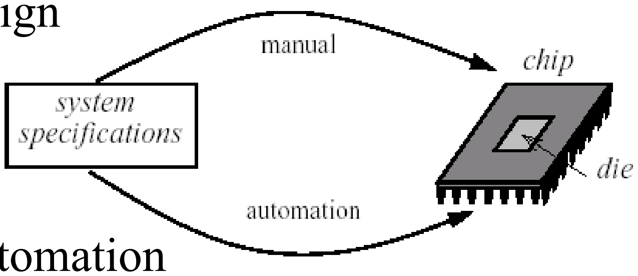
Path name	Pre-layout delay	Post-layout delay
P1	21.72	40.92 (+88.4%)
P2	6.65	7.81 (+17.4%)
P3	11.14	10.43 (-6.4%)
P4	5.03	5.44 (+8.15%)
P5	6.35	13.21 (+108.0%)
P6	6.42	13.20 (+105.6%)

Outline

- Moving to SOC
- CAD in SOC Design
- Solutions for Verification Problems
- Solutions for Integration Problems
- Solutions for DSM Problems

What is CAD ?

- CAD = Computer-Aided Design
  - Exist everywhere in the whole design flow
  - Important but invisible
- EDA = Electronic Design Automation
  - CAD for electronic industry

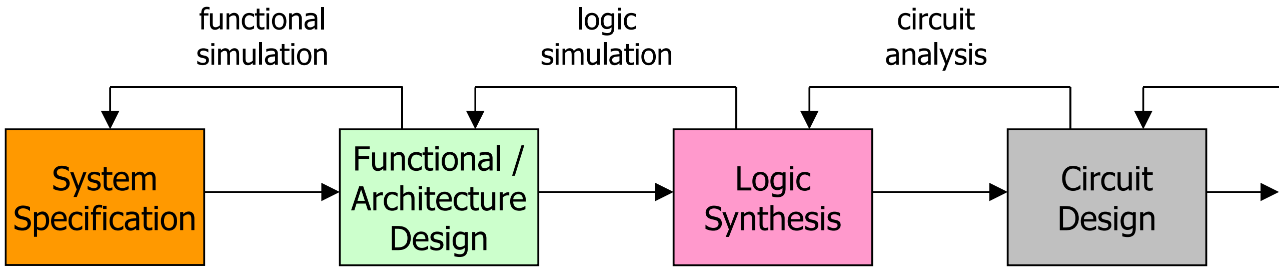


- Typically, more than 50% design time is elapsed on running CAD tools
  - Meet the time to market → faster implementation
  - Reduce costs → less engineers
  - Higher quality → fewer bugs
  - Managing the complexity → larger designs

Human vs. Computer

- Computation time
  - Human: slow
  - Computer: fast
- Correctness of results
  - Human: not guaranteed
  - Computer: always correct
- Working hours
  - Human: 8hrs / day
  - Computer: 24hrs / day
- And more and more advantages …

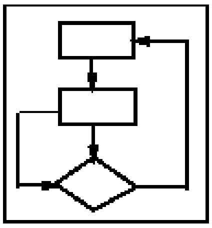
Typical VLSI Design Flow


functional simulation

logic simulation

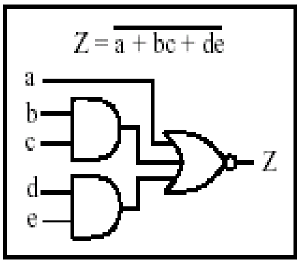
circuit analysis

System Specification → Functional / Architecture Design → Logic Synthesis → Circuit Design →

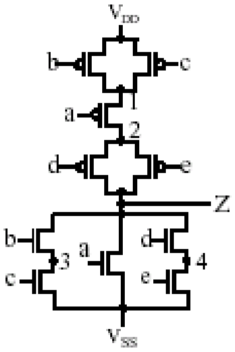

behavior representation

```
module fz(a,b,c,d,e,Z);
input a,b,c,d;
output Z;
assign Z =
  ~(a|(b&C)|(d&e));
endmodule
```

HDL description

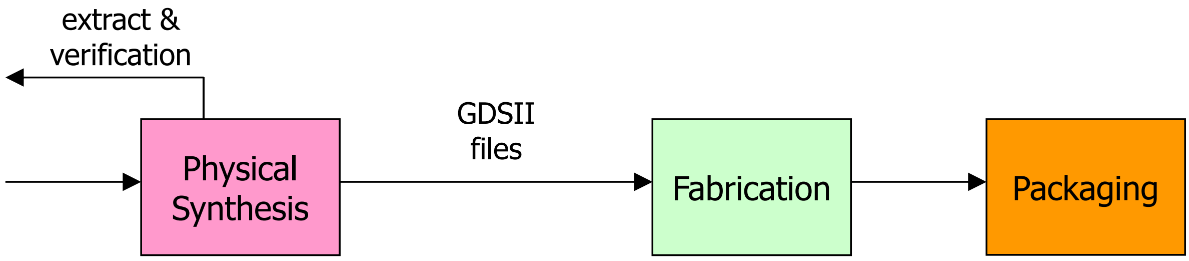


gate-level representation

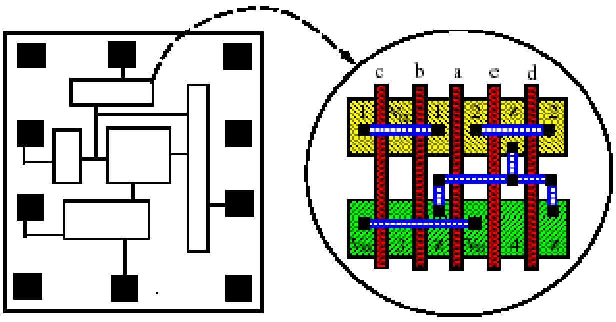


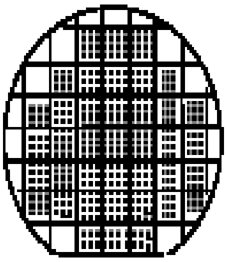
switch-level representation

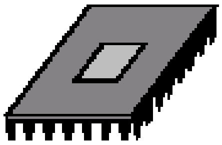
Typical VLSI Design Flow


extract & verification

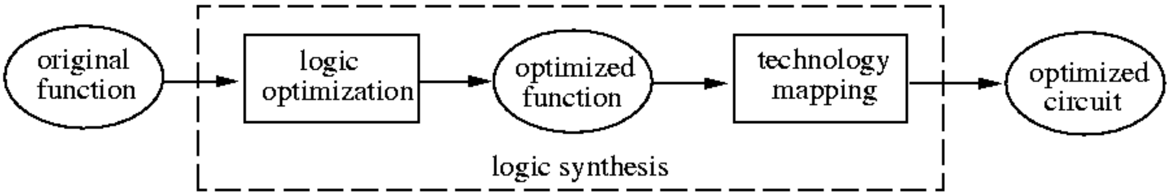
→ Physical Synthesis → GDSII files → Fabrication → Packaging







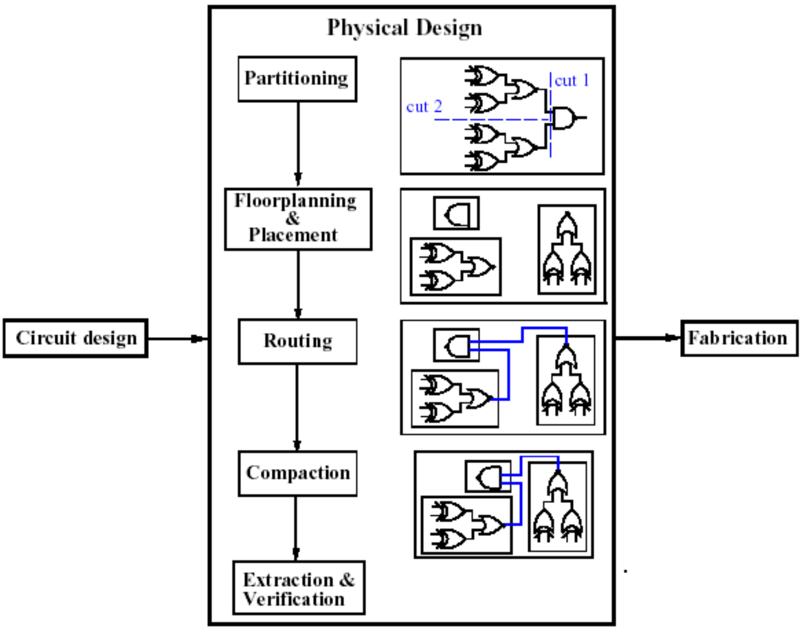
Logic Design/Synthesis



- **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.

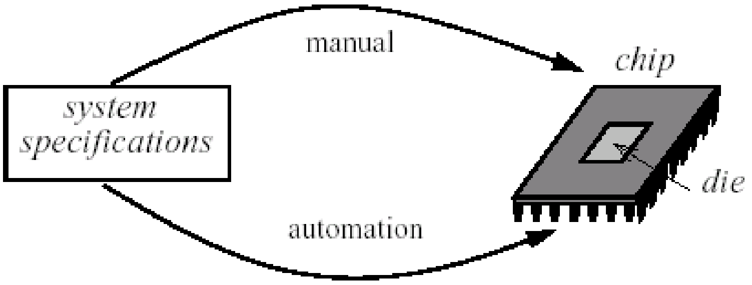
Physical Design Flow

- Physical design cycle:
  1. Logic partitioning
  2. Floorplanning, placement, and pir assignment
  3. Routing (global and detailed)
  4. Compaction
  5. RLC extraction & verification



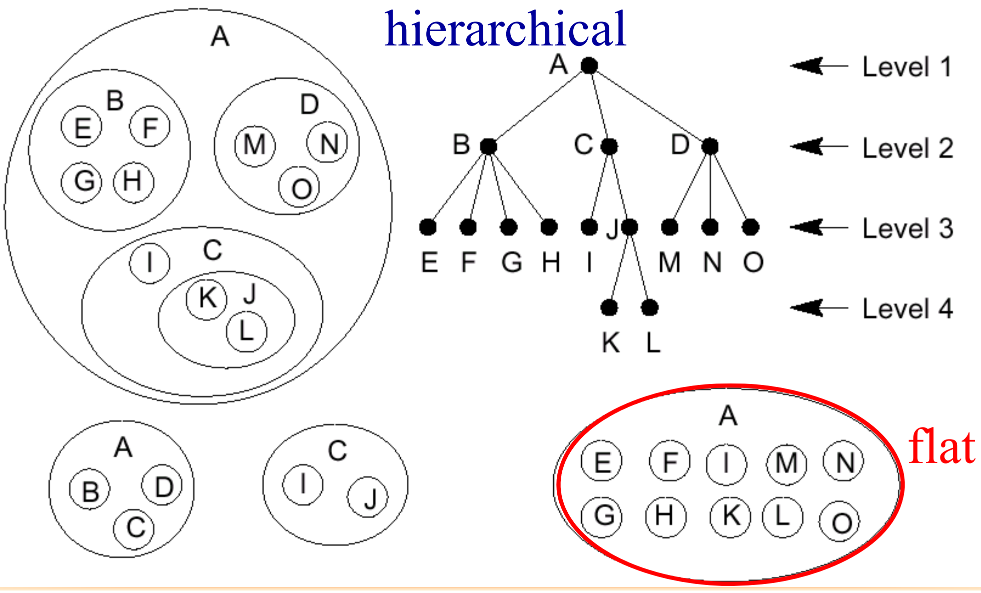
VLSI Design Considerations

- Several conflicting considerations
  - **Design complexity**: large number of devices/transistors
  - **Cost**: die area, packing, testing, etc.
  - **Performance**: optimization requirements for high performance
  - **Time-to-market**: about 15% gain for early birds
  - Others: power, noise, testability, reliability, … etc.
- Keys: hierarchical design, abstraction, design automation



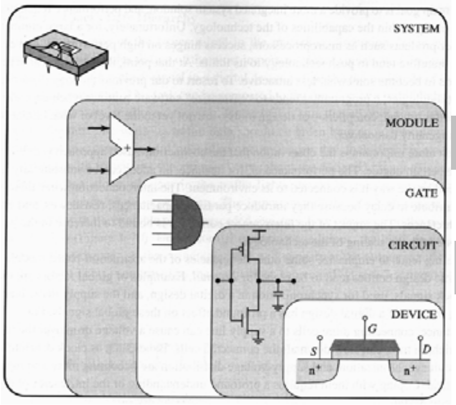
Hierarchical Design

- *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.



system

module

gate

circuit

device

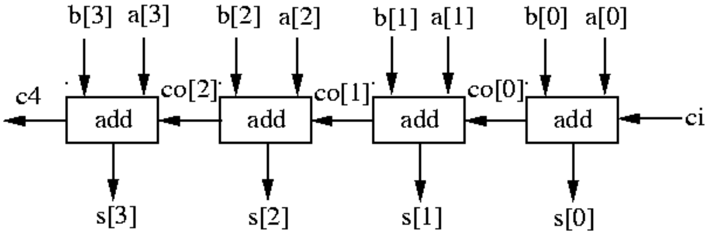
- Design domains:
  - Behavioral: functionality of components
  - Structural: connectivity between components
  - Physical: layout description
- Each design domain has its own hierarchy.

Three Design Views

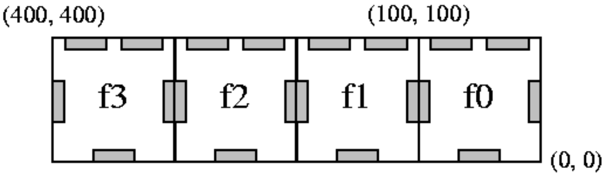
Behavior

```
Module add4 (s, c4, ci, a, b);
    input [3:0] a, b;
    input ci;
    output [3:0] s;
    output c4;
  assign {c4,s} = a + b + ci;
endmodule
```

Structural

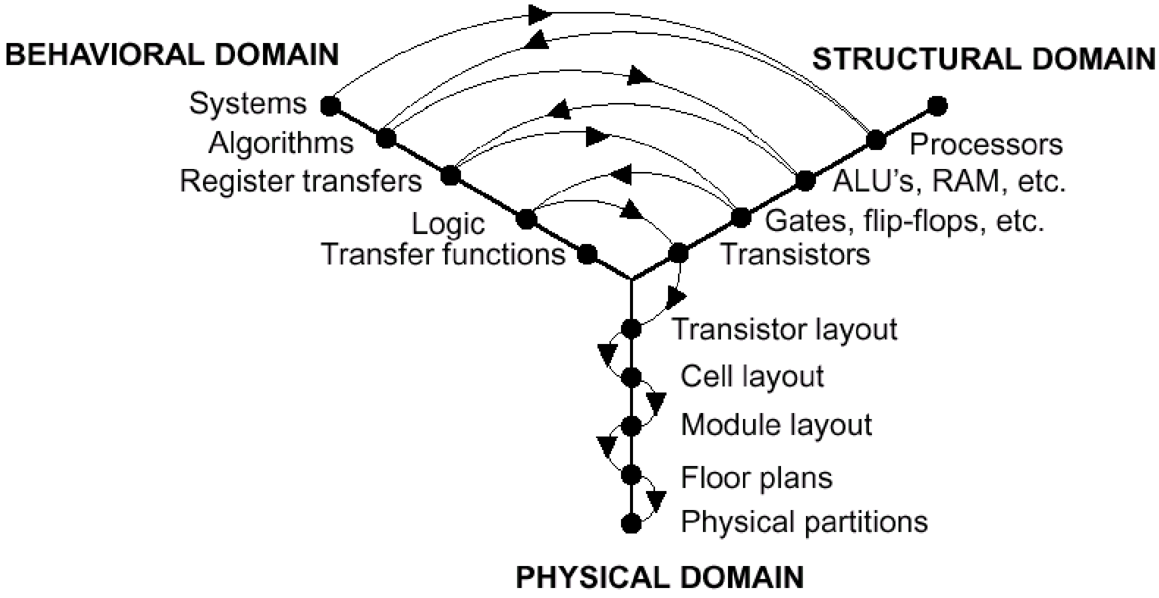


Physical



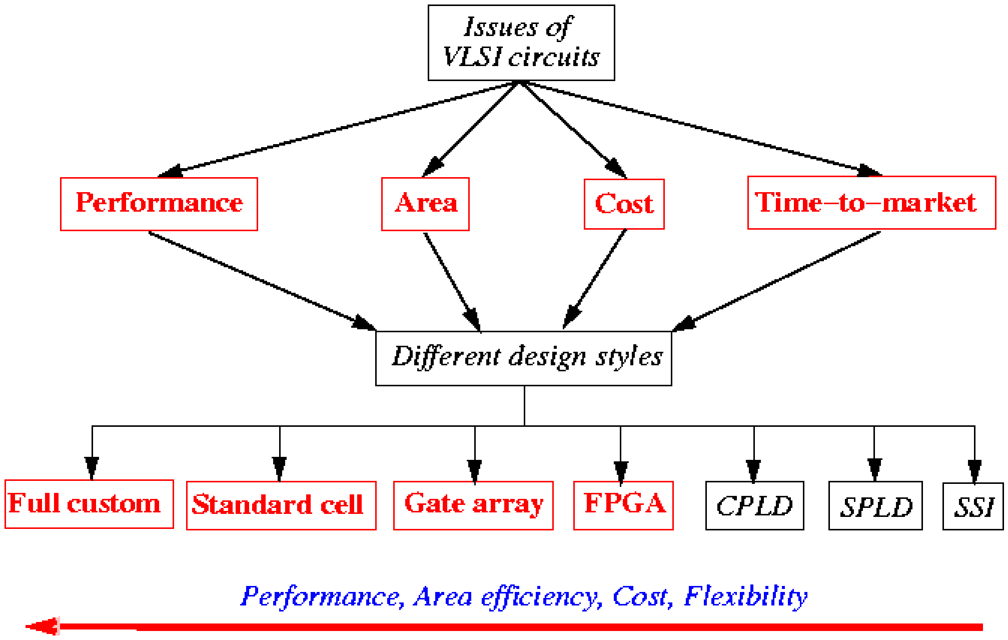
Top-Down Structural Design

- Also known as the Gajski's Y chart

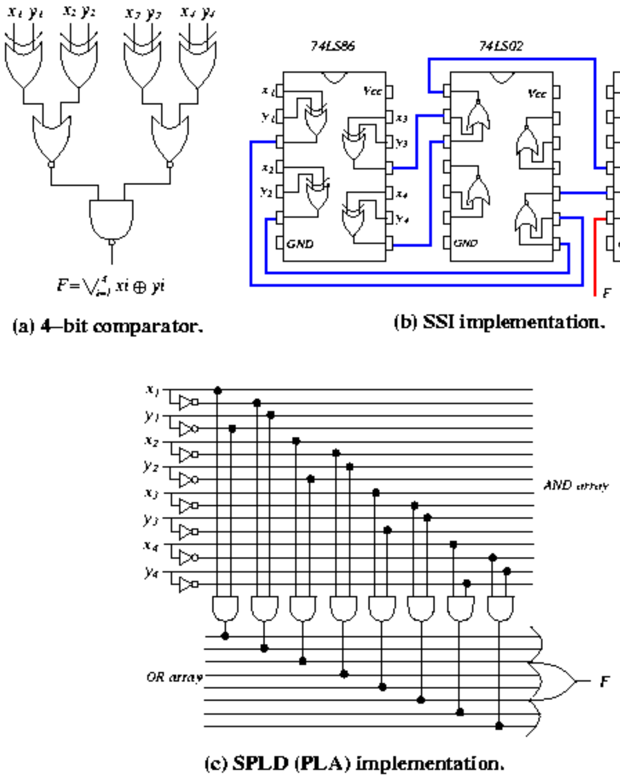


Design Styles

- Specific design styles shall require specific EDA tools



SSI/SPLD Design Style



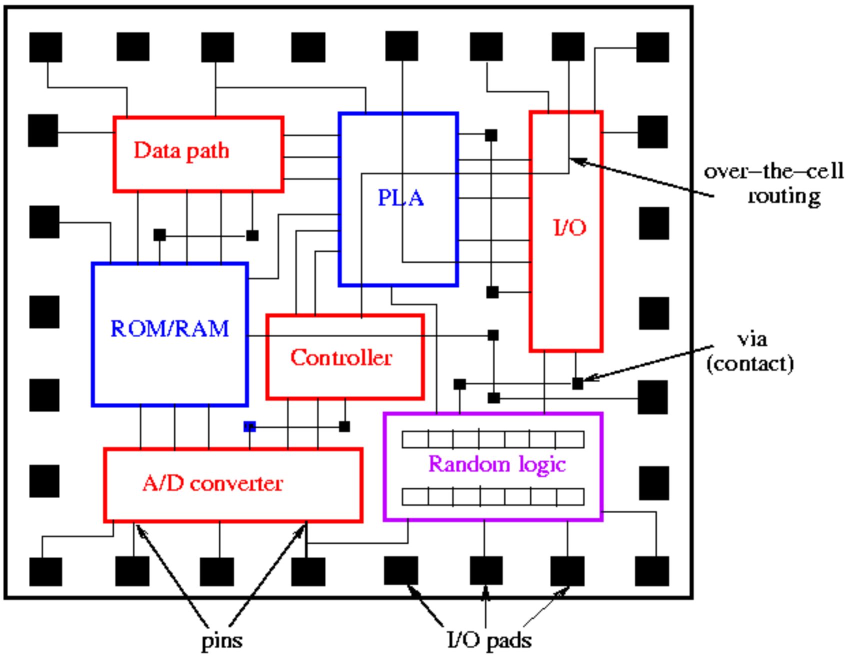
(a) 4–bit comparator.

(b) SSI implementation.

(c) SPLD (PLA) implementation.

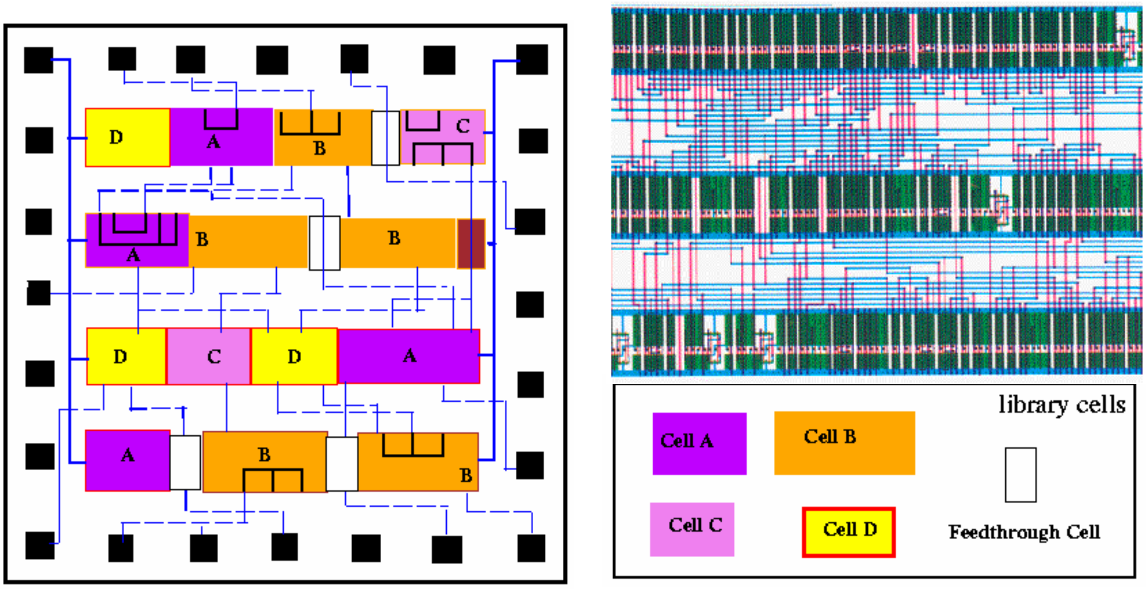
Full Custom Design Style

- 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

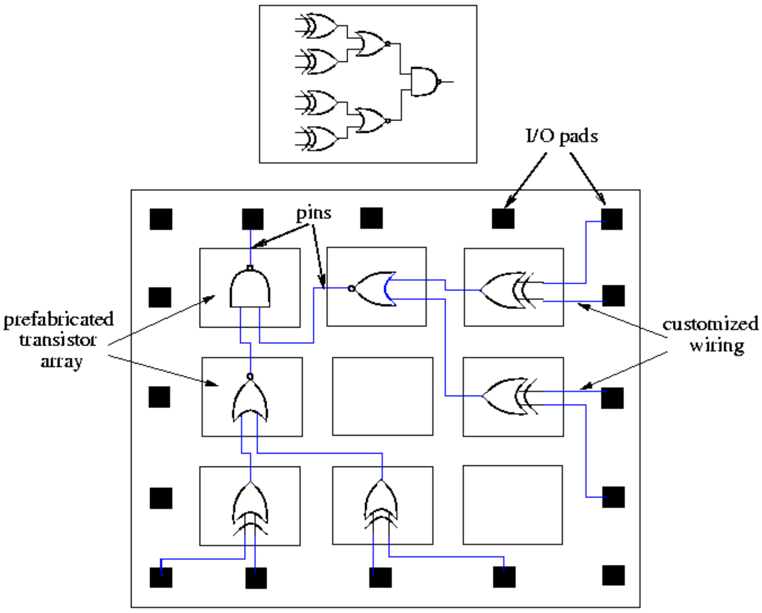
- Select pre-designed cells (of same height) to implement logic
- Characterize and store cells in library



Courtesy Newton/Pister, UC-Berkeley

Gate Array Design Style

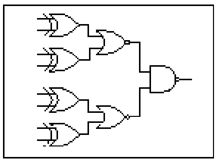
- Prefabricates a transistor array
- Needs wiring customization to implement logic

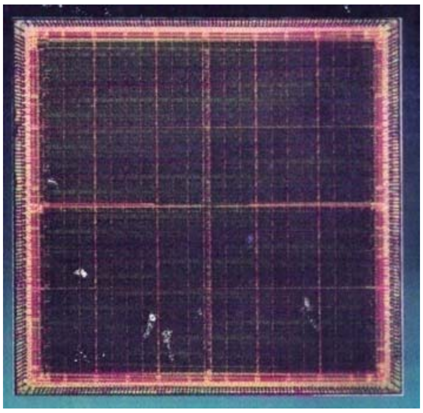


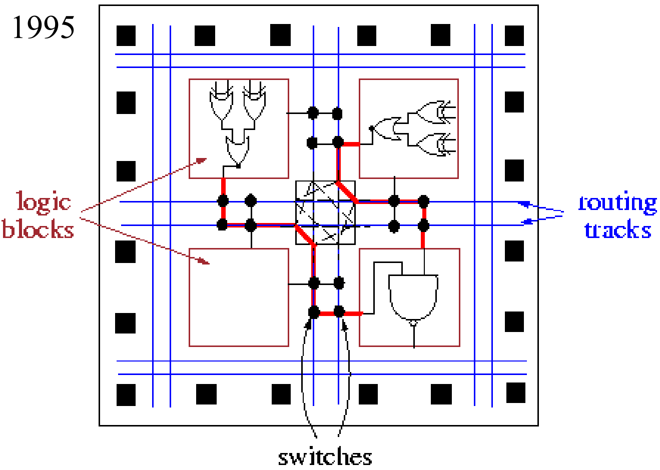
FPGA Design Style

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

Lucent Technologies 15K ORCA FPGA, 1995



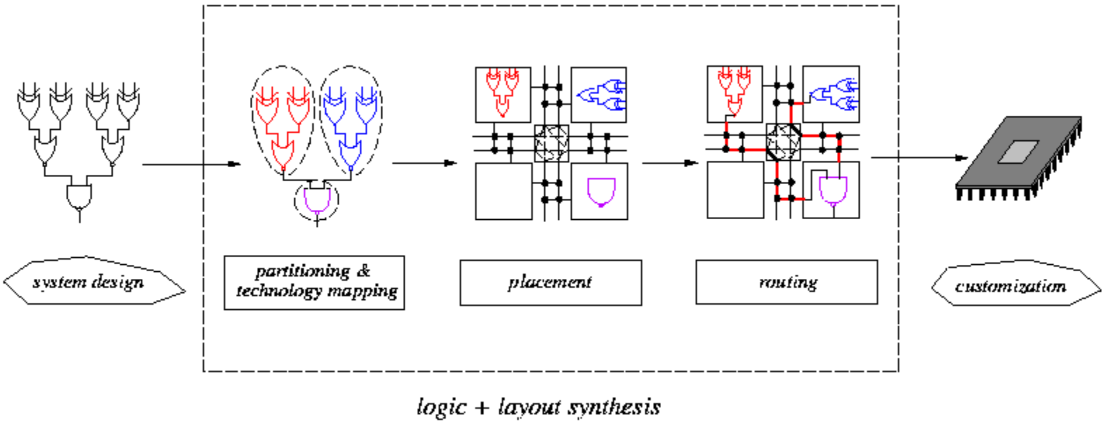




Prefabricated all chip components

FPGA Design Process

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



Comparisons of Design Styles

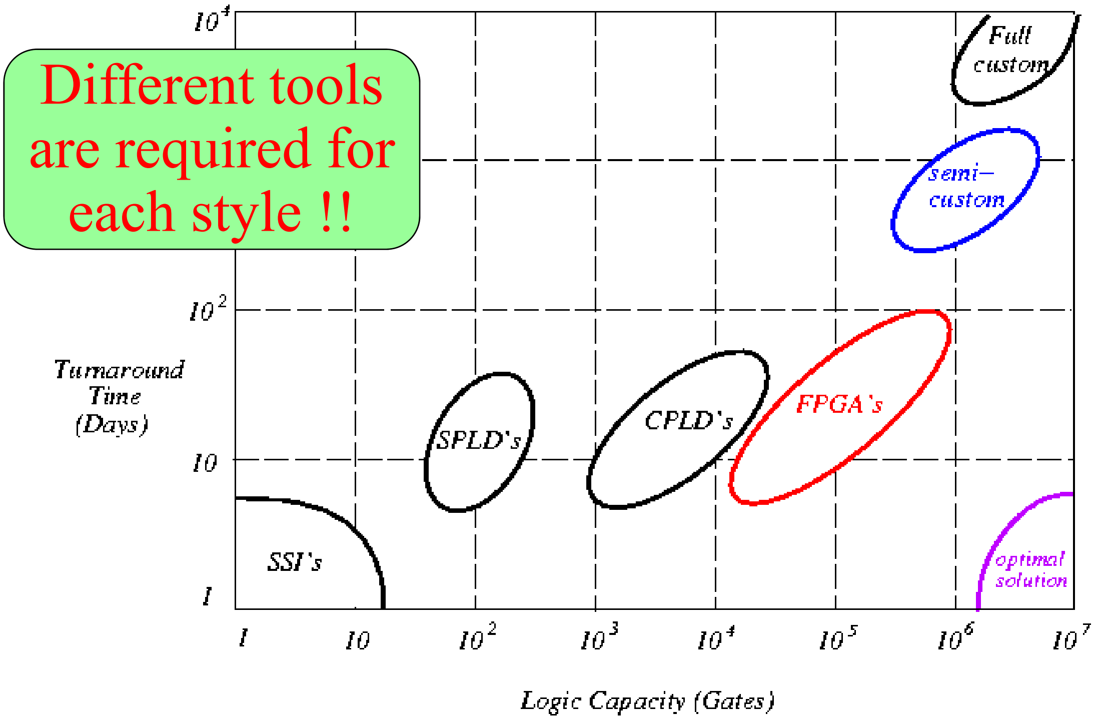
	Full custom	Standard cell	Gate array	FPGA	SPLD
Cell size	variable	fixed height*	fixed	fixed	fixed
Cell type	variable	variable	fixed	programmable	programmable
Cell placement	variable	in row	fixed	fixed	fixed
Interconnections	variable	variable	variable	programmable	programmable

* Uneven height cells are also used.

	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



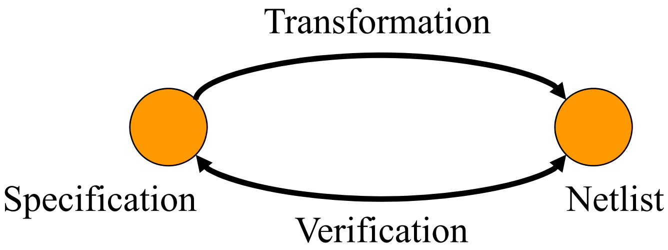
Different tools are required for each style !!

Outline

- Moving to SOC
- CAD in SOC Design
- Solutions for Verification Problems
- Solutions for Integration Problems
- Solutions for DSM Problems

What is Verification ?

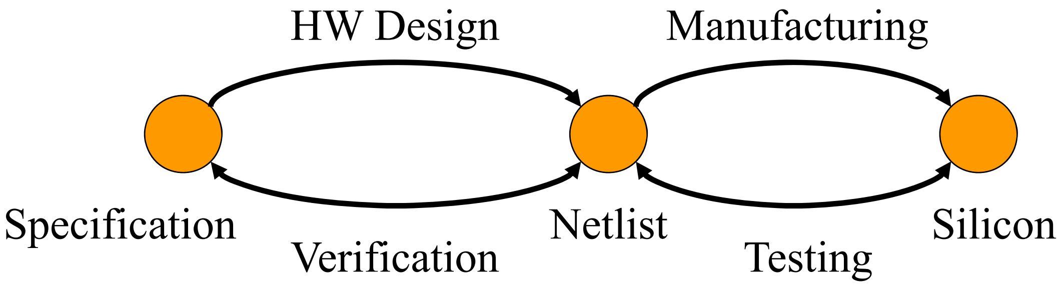
- A process used to demonstrate the functional correctness of a design
- To making sure that you are indeed implementing what you want
- To ensure that the result of some transformation is as expected



Source : "Writing Test Benches – Functional Verification of HDL Models" by Janick Bergeron, KAP, 2000.

Testing v.s. Verification

- Testing verifies manufacturing
  - Verify that the design was manufactured correctly

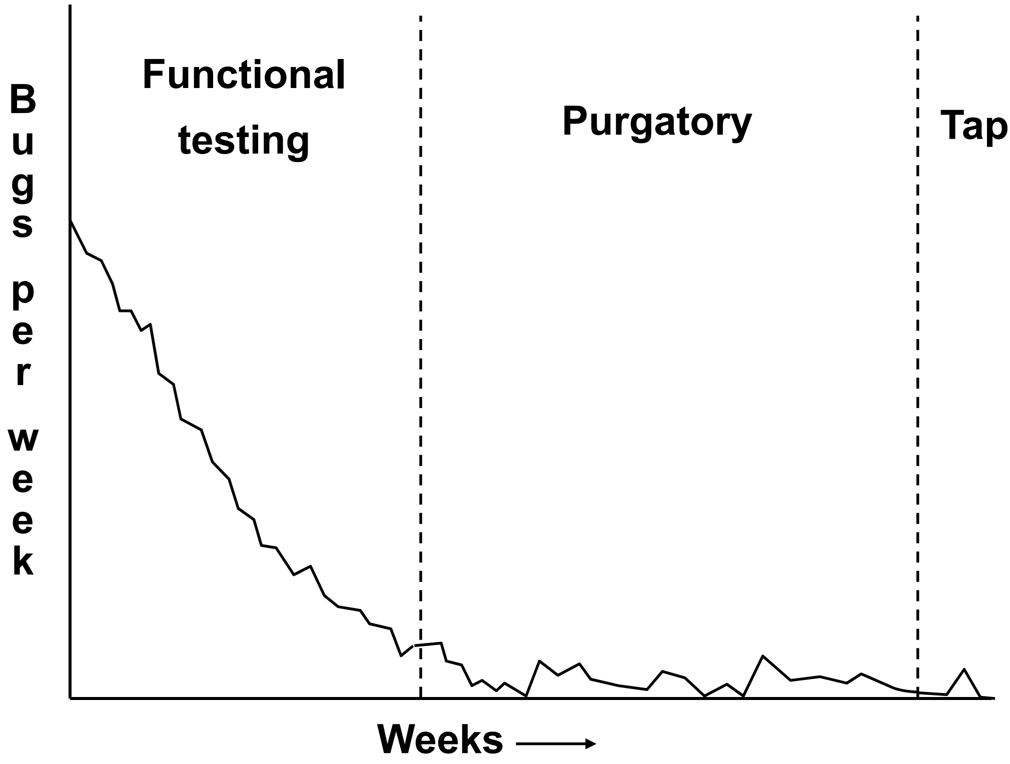


Source : “Writing Test Benches – Functional Verification of HDL Models” by Janick Bergeron, KAP, 2000.

When is Verification Complete ?

- Some answers from real designers:
  - When we run out of time or money
  - When we need to ship the product
  - When we have exercised each line of the HDL code
  - When we have tested for a week and not found a new bug
  - ***We have no idea!!***
- Designs are often too complex to ensure full functional coverage
  - The number of possible vectors greatly exceeds the time available for test

Typical Verification Experience



The Famous Pentium Bug

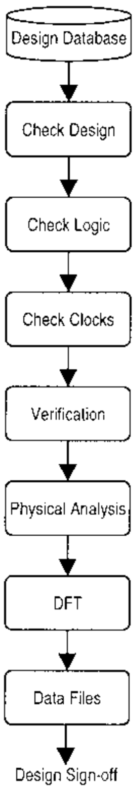
- Summary of the Pentium division bug
  - Pentium was Intel’s mainstream microprocessor
    - 3.3 Million transistors
  - Early versions all had error in floating point division hardware
    - 5 missing transistors, fixed with change to single mask
  - Disclosed largely via Internet
  - Intel ultimately offered replacements to everyone
    - $475 Million charge from 4Q94 revenue

Why Did not Intel Discover it ?

- Standard steps in verifying design
  - Simulate many cases on high-level software model
  - Simulate/emulate final logic design
    - Hardware emulators costing $Millions
    - Run complete chip model at ~ 100Hz (< 10^{-6} X real time)
  - Run tests on initial production chips
    - Feasible to run billions of tests
    - Should have caught error here
  - 1 trillion (10^{12}) test vectors for Pentium chip
- Observations
  - Hard to test all aspects of such a complex system

Design Sign-off

- Sign-off is the final step in the design process
- It determines whether the design is ready to be taped out for fabrication
- No corrections can be made after this step
- The design team needs to be confident that the design is 100% correct
  - Many items need to be checked
- Many verification tools are involved at each step in the sign-off flow



Design Database

Check Design

Check Logic

Check Clocks

Verification

Physical Analysis

DFT

Data Files

Design Sign-off

Ref : “System-on-a-chip Verification – Methodology and Techniques” by P. Rashinkar, etc., KAP, 2001.

Outline

- Moving to SOC
- CAD in SOC Design
- Solutions for Verification Problems
- Solutions for Integration Problems
    - Hardware v.s. software
    - Digital circuits v.s. analog circuits
- Solutions for DSM Problems

Prototyping

- Verify designs using real hardware
    - FPGA, emulator
- Better throughput in handling complex designs
- Software-driven verification
    - Verify SW using HW
- Interfaced with real HW components
- May have a little performance degradation
- May have capacity limitation
- Poor debugging capability
- High cost to fix design problems

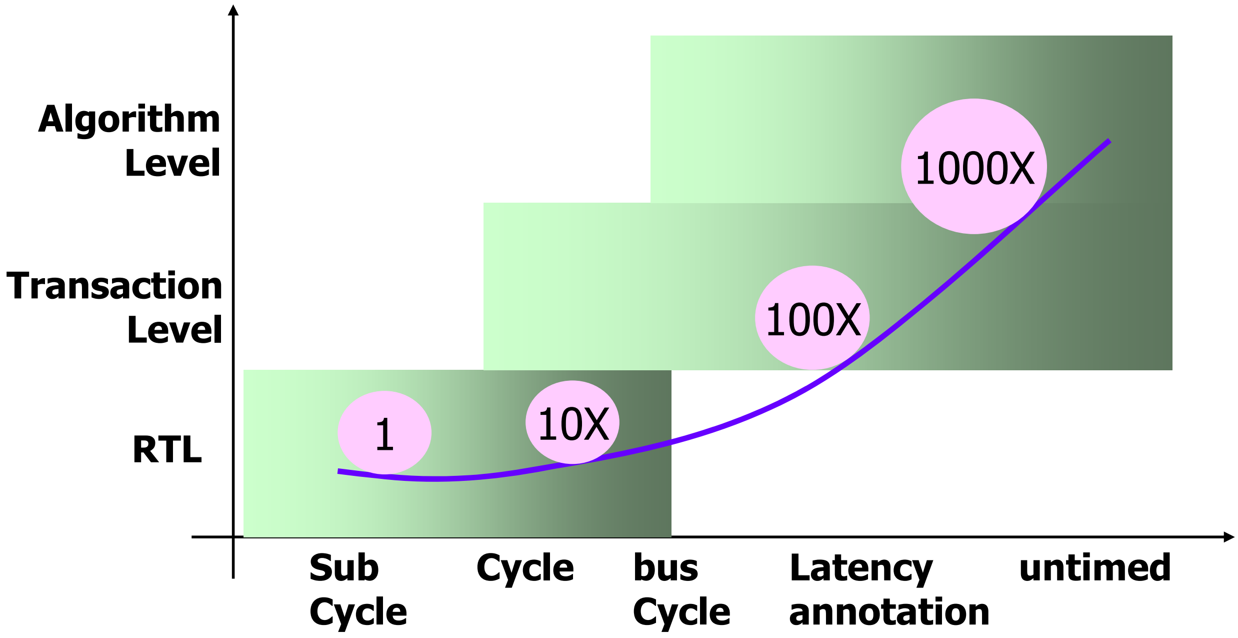
Limited Production

- Even after robust verification process and prototyping, it's still not guaranteed to be bug-free
- Engineering samples
- A limited production for new macro is necessary
  - 1 to 4 customers
  - Small volume
  - Reducing the risk of supporting problems
- Same as real cases but more ***expensive***
- Only used as the final check before mass production

HW/SW Co-Simulation

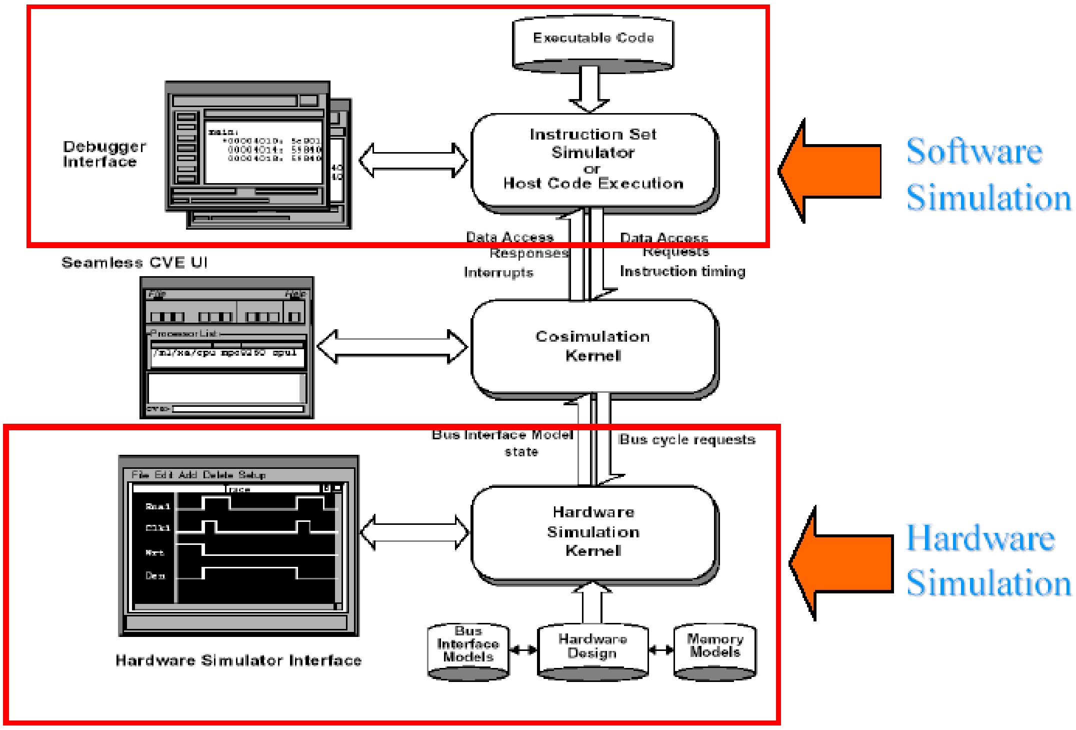
- Couple a software execution environment with a hardware simulator
- Simulate the system at higher levels
  - Software normally executed on an **Instruction Set Simulator** (ISS)
  - A **Bus Interface Model** (BIM) converts software operations into detailed pin operations
- Allows earlier system integration
- Start software development 6 months earlier
- Provide a significant performance improvement for system verification
  - Simulate 100x~1000x faster than RTL

Verification Speed



Reference : Synopsys

Many Available Tools for Co-Sim.

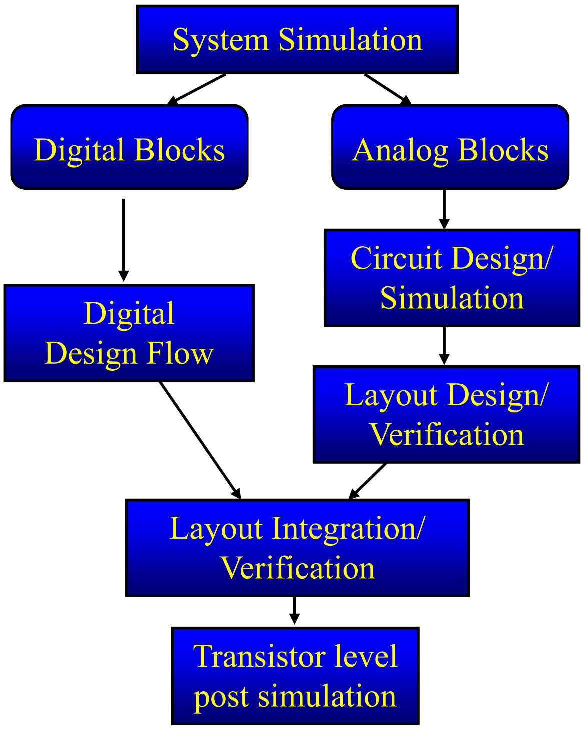

Executable Code
Debugger Interface
Instruction Set Simulator or Host Code Execution
Data Access Responses Interrupts
Data Access Requests Instruction timing
Seamless CVE UI
Cosimulation Kernel
Bus Interface Model state
Bus cycle requests
Hardware Simulation Kernel
Hardware Simulator Interface
Bus Interface Models
Hardware Design
Memory Models
Software Simulation
Hardware Simulation

Reference : Mentor Graphics

Outline

- Moving to SOC
- CAD in SOC Design
- Solutions for Verification Problems
- Solutions for Integration Problems
  - Hardware v.s. software
  - Digital circuits v.s. analog circuits
- Solutions for DSM Problems

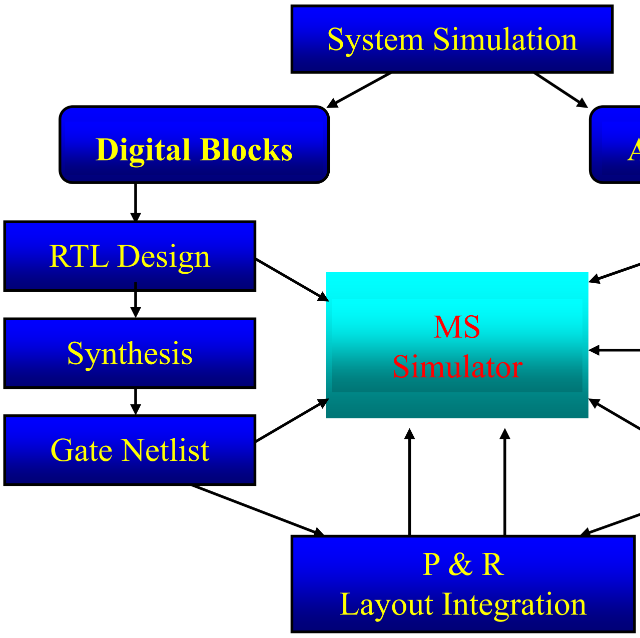
Conventional MS Design Approach


System Simulation
Digital Blocks
Analog Blocks
Digital Design Flow
Circuit Design/ Simulation
Layout Design/ Verification
Layout Integration/ Verification
Transistor level post simulation

- Design and simulate digital and analog circuits separately
  - May miss the effects between each other
- Can perform co-simulation at transistor level only
  - High complexity
  - Too slow

Top-Down MS Design Flow

- Starting from **behavioral models**
  - Can check whole system behavior in advance



Behavioral Modeling

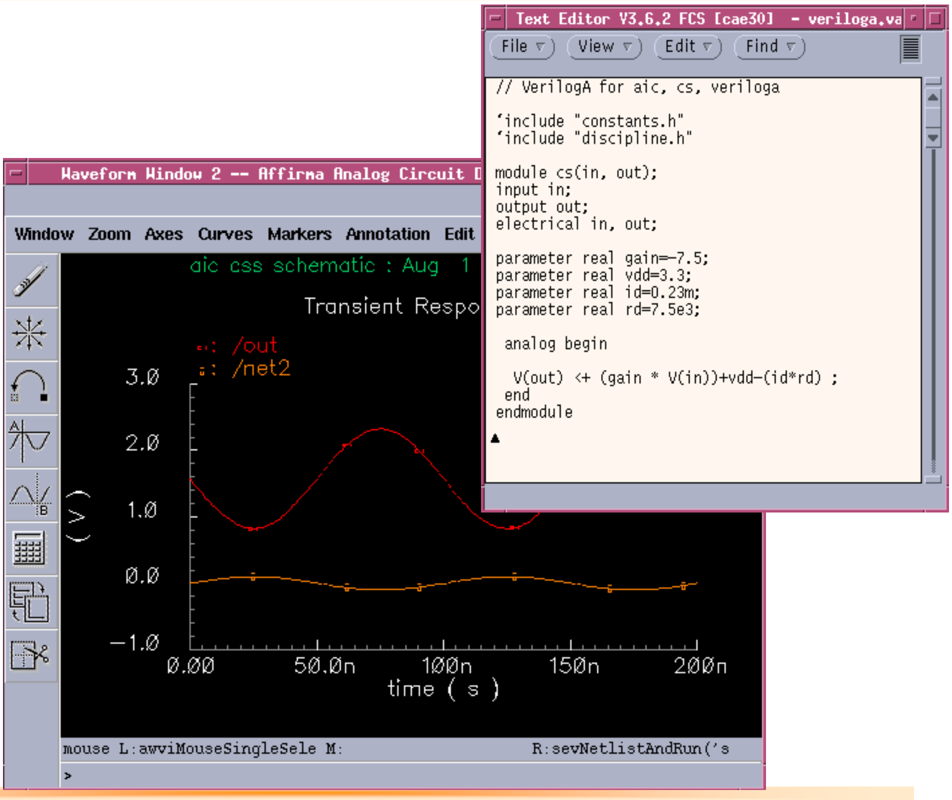
- A mathematical model written in **Hardware Description Language**
  - Verilog-A
  - Verilog-AMS
  - VHDL-A
  - VHDL-AMS
  - ……
- Emulate circuit block functionality by sensing and responding to circuit conditions

Why Behavioral Simulation ?

- **Faster** simulation
- Can verify circuit blocks at **behavioral level**
- Allow **system simulation** with pin-accurate block models
  - Easier to verify the system integration
- Digital blocks can also be simulated together in a mixed-signal environment
- Whole chip simulation becomes feasible
- Have to change the original methodology of analog designers
  - Hard !!

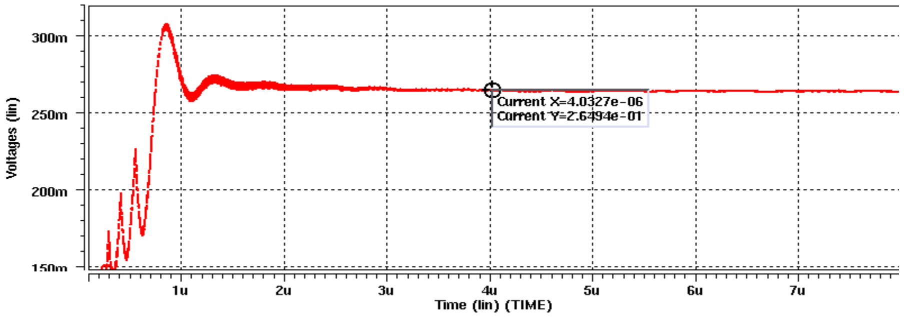
Available Simulation Environment

- Cadence
- Antrim
- Mentor
- Synopsys
- ……



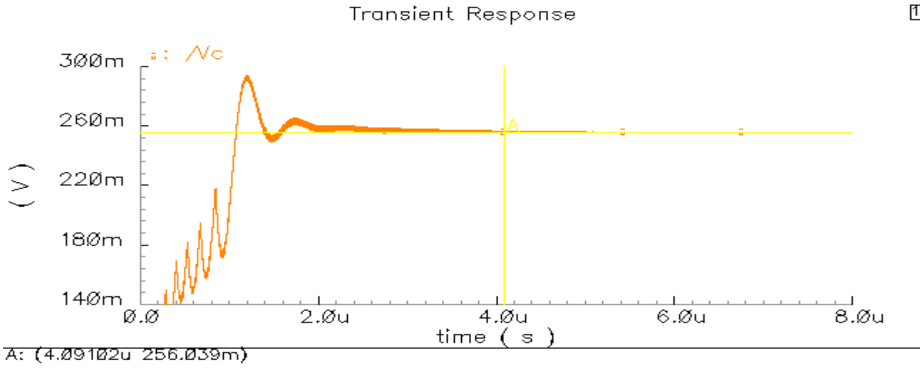
```
// VerilogA for aic, cs, veriloga

'include "constants.h"
'include "discipline.h"

module cs(in, out);
input in;
output out;
electrical in, out;

parameter real gain=-7.5;
parameter real vdd=3.3;
parameter real id=0.23m;
parameter real rd=7.5e3;

 analog begin

  V(out) <+ (gain * V(in))+vdd-(id*rd) ;
 end
endmodule
```

Waveform of PLL Locking



HSpice

Lock time:
4.03 us
Lock voltage:
0.265 v
Simulation time:
22405 sec



Verilog-A

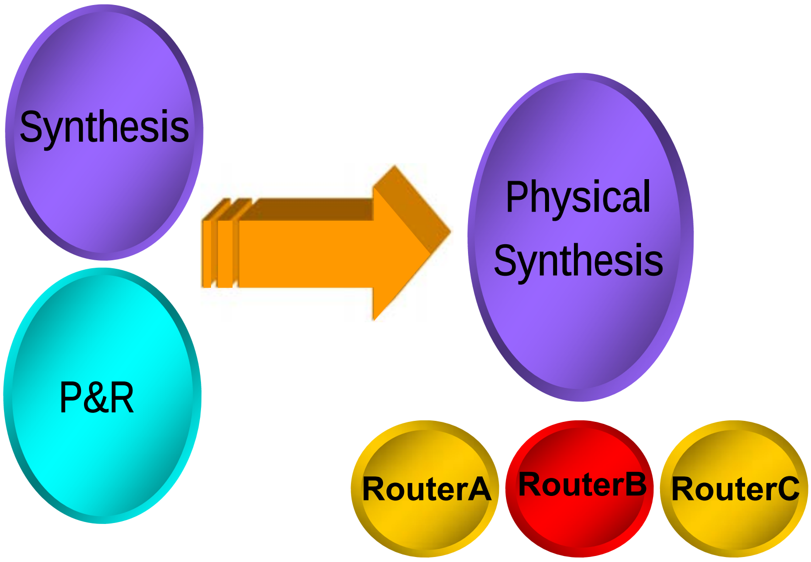
Lock time:
4.09 us
Lock voltage:
0.256 v
Simulation time:
336 sec

Outline

- Moving to SOC
- CAD in SOC Design
- Solutions for Verification Problems
- Solutions for Integration Problems
- Solutions for DSM Problems

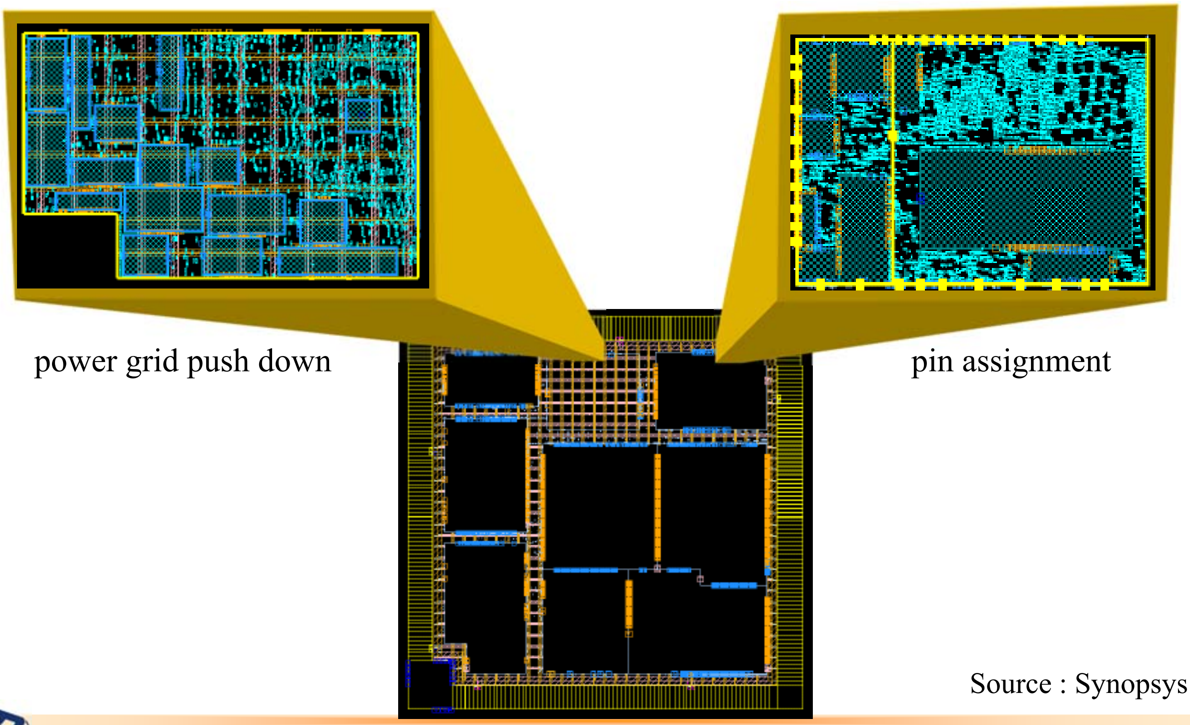
Physical Synthesis

- "Physical synthesis" is a new concept to solve timing closure problem at the synthesis step
- Integrate placement engine with logic synthesis to get more accurate wire length estimation
  - From outer loop to inner loop ??
- Not only database but also timing engine are integrated
  - Consistency makes better results



Hierarchical P&R

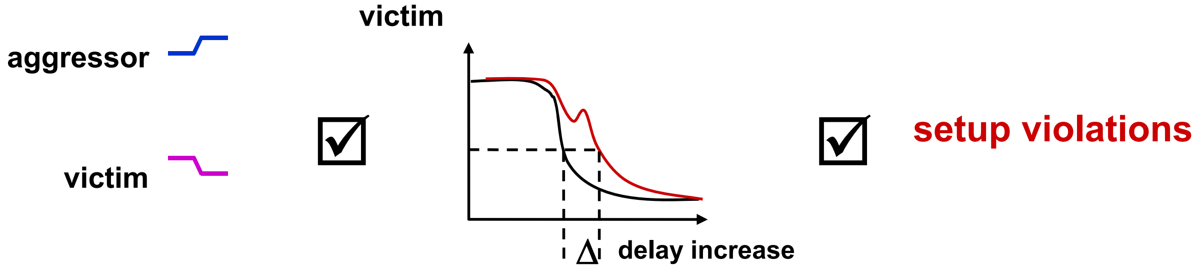
- Hierarchical methodologies are the key to cope high complexity !!



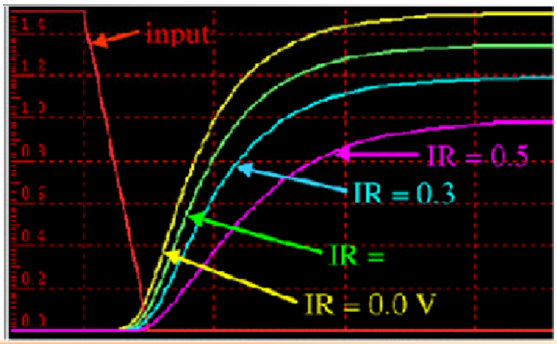
Source : Synopsys

Extra Checks for SI & IR-Drop

- Must be checked for signal integrity (SI) problems before manufacturing

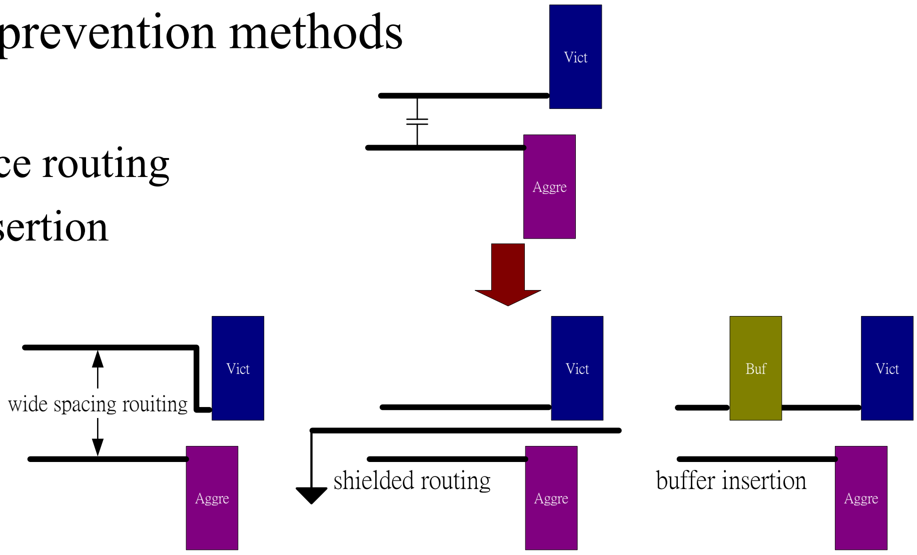


- IR-drop is also an serious problem due to large peak current



SI & IR-Drop Prevention

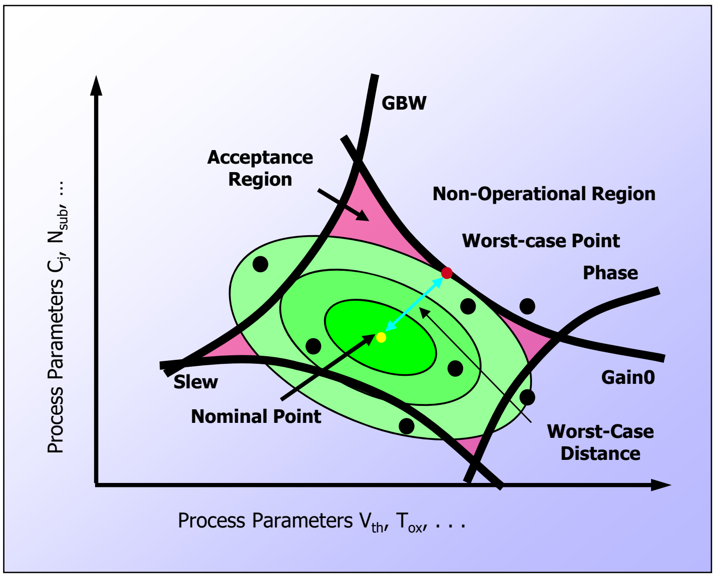
- Typical SI prevention methods
  - Shielding
  - Wide space routing
  - Buffer insertion



- To prevent IR-drop problem, we have to identify the ***current density*** problems with power analysis

Solve Reliability Problems

- There are still some problems that can make chip fail after a period of time
  - Electron migration
  - Hot electron
  - Process variation
  - ……
- Use design-for-manufacturing (DFM) or design-for-Yield (DFY) techniques to obtain suitable design margins



Source : ChipMD

Conclusions

工欲善其事
必先利其器 !!

- SoC is really a big challenge for IC designers
- Design methodology requires a big change
  - Reusing existing IPs and platforms is the key
- More powerful tools are essential to solve those difficult design problems
- ***EDA – Where Electronics Begins***