

# Overview of Design Methodologies --- A Review

## Lecture 1

ECE 156B

1

## Hardware Description Languages (HDLs)

- Programming languages used to describe hardware behavior and properties
- Provide different layers of abstraction
  - Structural/Gate
  - Register Transfer Level (RTL)
  - Behavioral
- Serve as input to synthesis
- Support *simulation* and *verification*

ECE 156B

2

## System Verilog

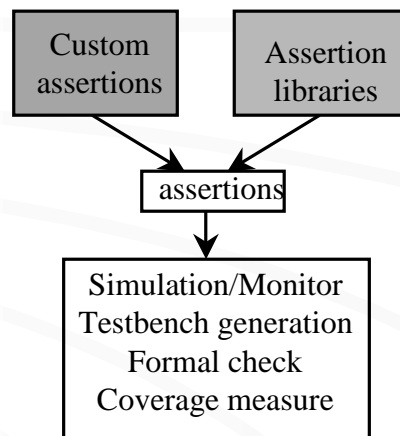
- [www.systemverilog.org](http://www.systemverilog.org)
- Extend Verilog IEEE 2001 to higher abstraction levels for Architectural and Algorithmic Design, and Verification
  - Verilog testbench -> Transaction-level testbench and coverage measurement
  - Verilog assertion -> Allow semi-formal and formal check of design properties
  - Verilog abstraction -> design abstraction, abstract data types, abstract operators
  - Verilog API interface -> C interface, assertion interface, coverage interface

ECE 156B

3

## System Verilog Assertion-based Verification

- Assertions are part of design model
  - Enable verification re-use
  - Improve quality
  - Facilitate testbench generation
  - Serve multiple purposes



ECE 156B

4

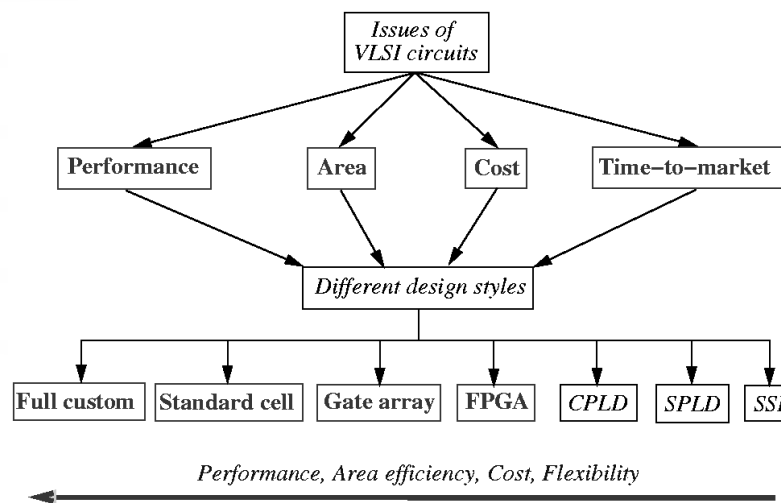
## System C

- [www.systemc.org](http://www.systemc.org)
- "The new System C Verification Standard improves the capability by providing APIs for transaction-based verification, constrained and weighted randomization, exception handling, and other verification tasks" – A Tutorial Introduction on the New SystemC Verification Standard, C. Norris Ip, Stuart Swan
- The language is used to implement the testbench, the transactor, and the design

ECE 156B

5

## Design Styles



ECE 156B

6

## Design Flow

- Derive design specification
- Construct simulation model
  - Behavior and/or RTL
  - Construct *assertions*
- Functional verification of simulation model
- Derive schematics
  - Synthesis (involve logic minimization), or
  - Custom design high performance components
- Equivalence checking (RTL Vs. Schematics)
- Derive gate-level test model
  - Design for test and test generation
- Placement and routing
- Clock tree design and routing
- Performance validation
  - Timing analysis, power analysis, physical design rule check

ECE 156B

7

## Post-Silicon

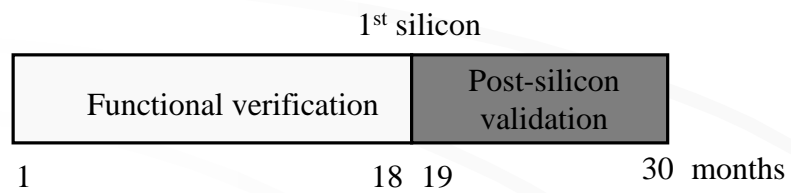
- Tapeout
- Obtain first silicon
- Testing setup
- Yield improvement
  - Diagnosis and silicon debug
  - On-chip repair
- System bring-up test
- Mass production/ Speed binning

ECE 156B

8

## Where Spends Most Resources?

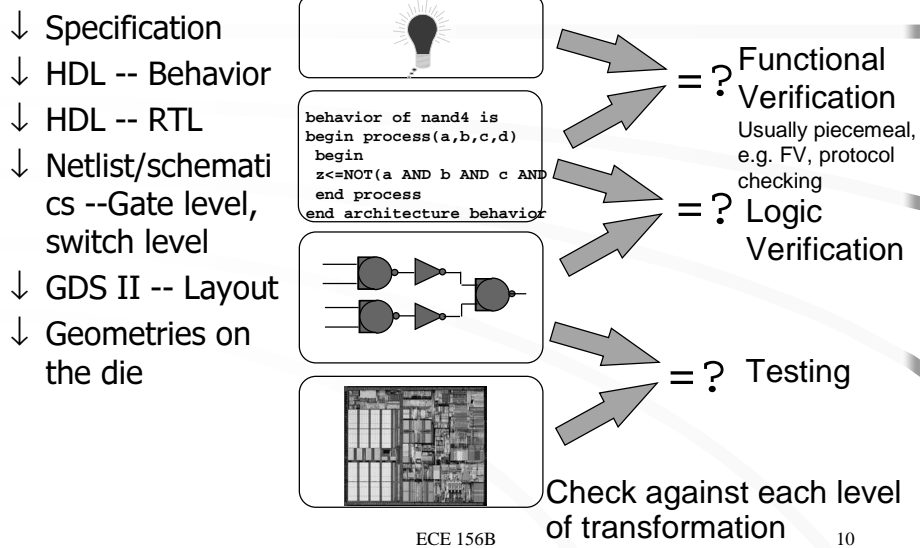
- Functional Verification
- Post-Silicon Validation



ECE 156B

9

## Design Data Hierarchy



ECE 156B

10

## Synthesis (front end)

- Behavioral synthesis
  - Resource allocation (Design automation conference, 1986, pp 474-480)
  - Pipelining (DAC 86, pp. 454-560, 461-466)
  - Control flow parallelization (IEEE Tran. On Computer-aided design of Ics, 1986, No 2, pp 259-269)
  - Communicating sequential processes
  - Partitioning (ICCAD 1989, pp. 280-283)
- Sequential synthesis
  - Register movement/retiming (FOCS, 1981, pp 23-26)
  - State minimization
  - State Assignment (Encoding) (IEEE TCAD, Dec, 1988, 1290-1300)
  - Synthesis for testable design
  - State machine verification

ECE 156B

11

## Synthesis

- Logic synthesis
  - Extracting combinational logic from RTL
  - 2-level logic minimization
  - Algebraic decomposition
  - Multi-level logic minimization
  - Synthesis for test structures (scan)
  - Redundancy removal via ATPG
  - Timing optimization (gate-level)
- Technology mapping
  - Mapping gates to library cells

ECE 156B

12

## Physical design synthesis (back end)

- Placement
  - Global/local
- Routing
  - Global/local
  - clock
- Physical design rule checking
  - Timing analysis (usually cell-based)
  - Layout vs. schematic (LVS) checking
  - Timing sign-off
  - Design rule sign-off

ECE 156B

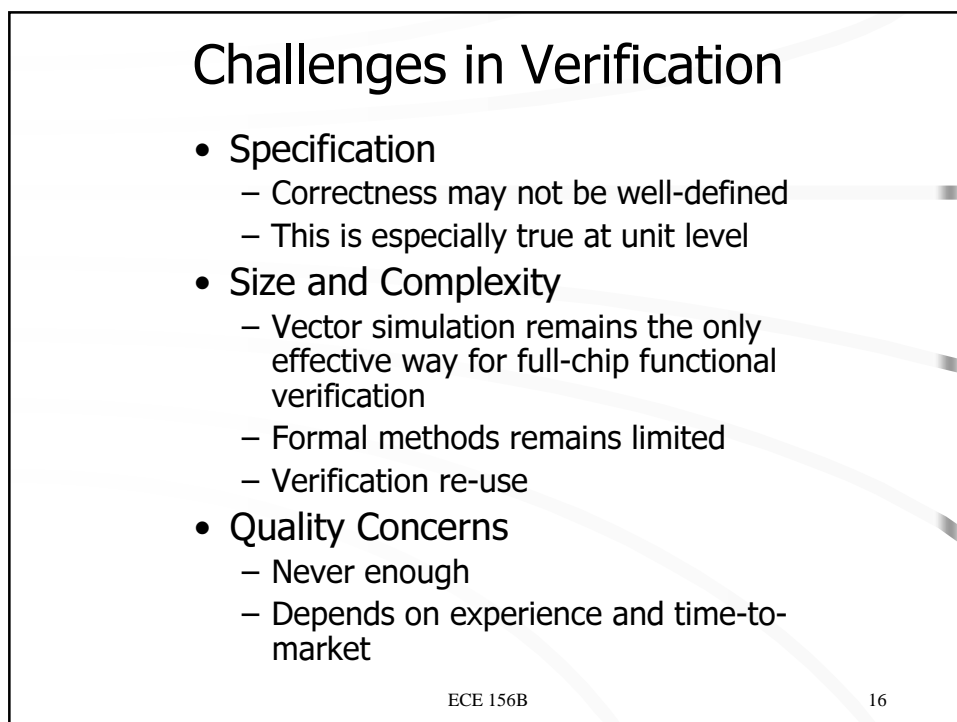
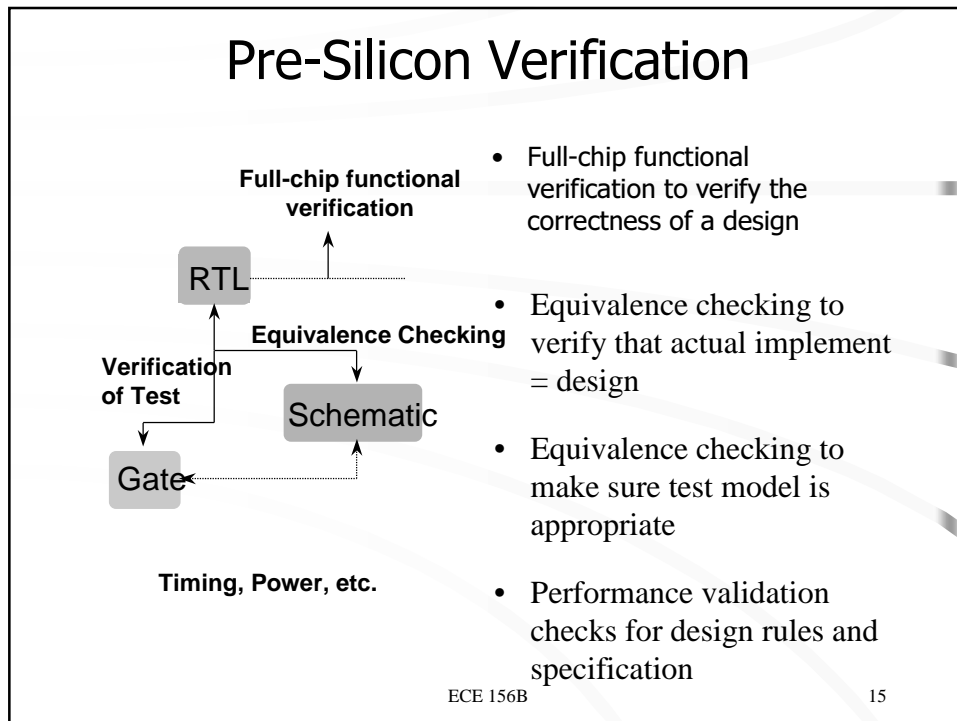
13

## Synthesis is an optimization process

- Area optimization
  - Gate count
  - Physical dimension
- Critical path (timing) optimization
  - Number of gates from PIs to POs
  - Delay from PIs to POs
- Optimized for testability
  - Redundancy removal
- Power optimization
  - Minimizing switching activities

ECE 156B

14



## Post-Silicon Validation

- DV (Design)
  - Verify design sensitivity to various environmental conditions, (e.g. voltage, temperature, frequency, and process variation [to some degree]) with a given test suite -- fullchip level test (diagnostic)
- SV (system)
  - Verify that product is functional in a given system (designed to facilitate debugging) with real peripherals, BIOS, OS and Applications
- CV (Commercial/Compatibility)
  - Verify that product is functional across OEM systems, OSes, applications
- CMV (Circuit Marginality)
  - Verify that product is free from sensitivities to voltage/temp/frequency in a system level operation
- MV (Manufacturing)
  - Verify that a given product can be manufactured in HVM (high volume manufacturing); Yield is not impacted if circuit is manufactured in high volume, over time: process variation (+ all of the above)
- RV (Reliability)
  - Verify that the product has a low infant mortality rate and achieve low FIT (failure in time)

ECE 156B

17

## Challenges in Validation

- Lack of an unified, systematic approach for all the problems
  - Rely on experiences
  - Rely on a collection of approaches
- A subtle problem can cause serious delay in time-to-market
  - It may take months to figure out what is going on for a "bug/defect"
  - The amount of data needed to be processed are extremely large
- New phenomenon appear as we move to very deep sub-micron domain

ECE 156B

18

## In Summary

- No well-defined reference model
  - time-to-market pressure
  - may be changed next time
  - implicitly exist in a simulator
- Too many data formats
  - Increase logic verification burden
- Need better understanding of what we are looking for
  - Fault and error models
- Limitations in tools

ECE 156B

19

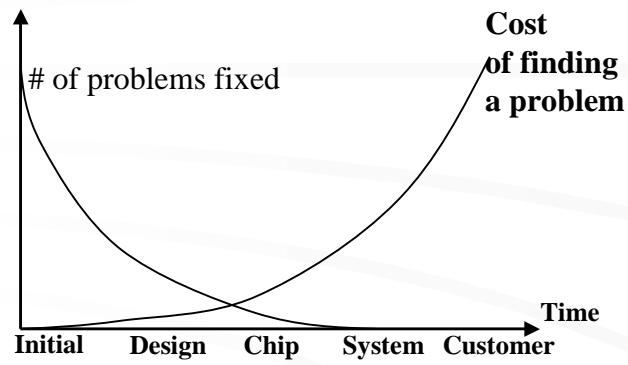
## No Perfect Solution!

- Design automation is achieved by a collection of approaches - no single magic solution!
  - Involve 100 different algorithms
  - Involve tens of different tools
  - Involve different engineers to own different aspects of a design block
  - Apply more formal and systematic approaches

ECE 156B

20

## Do Everything (Possible) Early



- Because the later you have a problem, the higher price it will cost you