Cross Fertilization Between Testing and Verification

Shmuel Ur

Avi Ziv

Verification Technologies Department

IBM HAIFA LABS

Overview

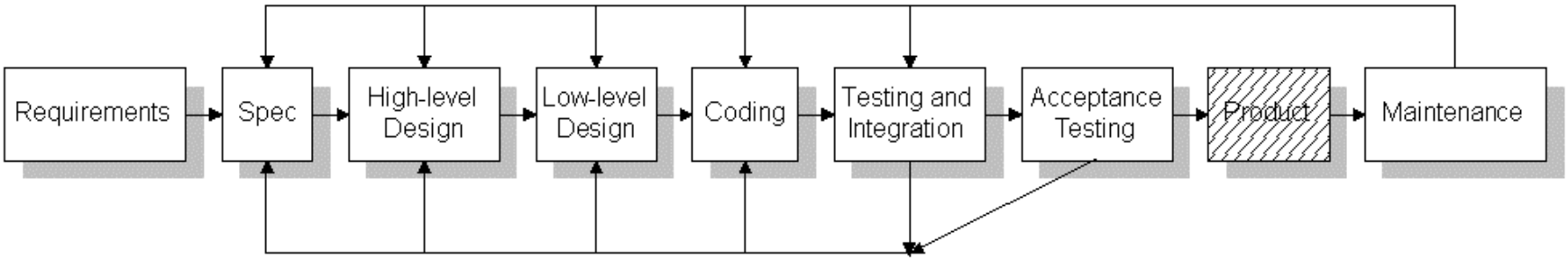
- Motivation
  - Similarities between software and hardware
  - And the differences
- Technology transfer
  - The challenge
  - What worked
  - What did not work
  - Requirements

Hardware and Software Development are Similar

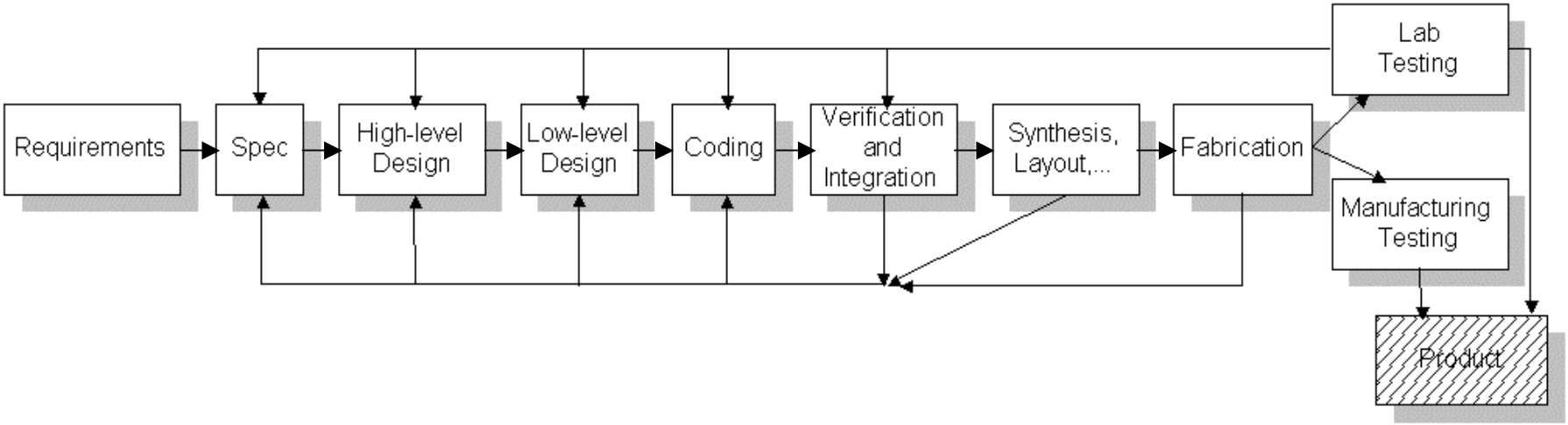
- Similar Development process:
  - Start with high level design
  - Break into smaller units
  - Implement and test each unit
  - Combine the units and test the system
- Even similar languages
  - Hardware design languages share same concepts as software programming languages

The Development Process

Software



Hardware

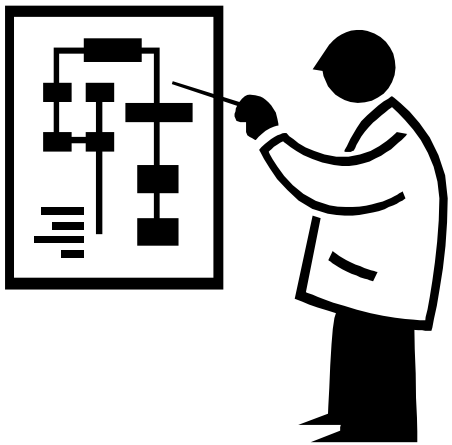


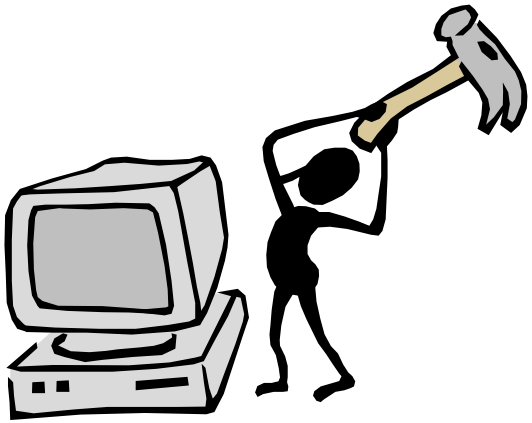
… But They Are Different

- Hardware projects have greater similarities
- The cost of bugs is higher in hardware
- Verification engineers have more sophisticated skills
- Hardware tends to have much longer cycles
- Customer expectations differ

The Big Question

Can the hardware verification and software testing communities share tools and methodologies?





Current State

- Almost no interaction between the communities
  - Separate conferences
  - Different tool companies
  - Different methodology developers
  - No interaction between users

Result: Tool/methodologies developed in one community rarely migrate

The Challenge

- Identify methods and tools that can be used by the other community
- Foster transfer of technology and methodology

Technologies that Crossed

- Code Coverage
  - Tools for C/C++, Java, Verilog, and VHDL
- Minimization of regression suites
  - Used at all levels
- Inspection of code and specifications

Technologies on One Side

- Hardware
  - Model checking
  - Use of behavioral
  - Automatic test generation
  - Functional coverage
- Software
  - Defect analysis

There is hope - crossing continues

Functional Coverage: Technology That Did Not Cross (Yet ?)

- Why?
  - High operation cost
    - Long learning time for methodology and tools
    - Long time to define models
  - Require deep domain knowledge for definition of coverage model
  - Benefits offered in terms of increased quality are often beyond the needs of software systems.

Transfer Requirements

- Tools for software testing need to be:
  - Easy to learn
  - Easy to maintain
  - Affordable
- Tools and methodologies can be used, but not ‘as is’
- Hardware can learn much from software design methodology

Future Work

- Keep an eye on the other side of the tracks
- Convert useful tools
- Common conferences?