Increasing Predictability in the Design Verification Process

Andrew Piziali, October 24, 2006

cadence™

Overview

- Introduction
- Why Are There Surprises?
- Plan the Verification of the Design
- Use the Plan as a Verification Process User Interface
- Summary

Introduction

- Chips are taped out with insufficient knowledge of their functional quality
- Software is released with even less certainty of its quality
- Market schedule dictates production
- Verification team does the best they can within the schedule
- However, no one can answer the question “Are we done yet?”

Why Are There Surprises?

- Uncertain resource requirements
  - Labor for development, bug discovery, diagnosis and repair
  - Verification software: simulators, formal, VIP
  - Verification hardware: CPUs, emulators, accelerators
- Undefined verification progress metrics
  - What quantitative leading indicators are used?
  - Are past requirements indicative of future needs?
- Ad hoc assessment of verification progress
  - Simulations run
  - Tests written, run and passed
  - Bugs found

Plan the Verification of the Design

- Write a verification plan, not a test plan
- Verification is demonstrating that the intent of a design is preserved in its implementation
- Testing—running trials against the implementation—is one way to verify
- A verification plan has two purposes:
  1. Quantify the scope of the verification problem
  2. Specify the solution to the verification problem
- “Quantify” means choosing metrics and implementing and using measurement tools
- “The solution” is a verification environment functional specification

Quantify the Scope of the Verification Problem

- Kinds of metrics
  - Physical metrics: length, mass, temperature
  - Synthetic metrics: verification coverage, bug rate, proven properties
- Functional verification coverage
  - Structural coverage
  - Assertion coverage
  - Functional coverage
- Functional coverage
  - Defined by a coverage model for each feature
  - Model approximates a feature subset in a declarative fashion
  - Model employs attributes, values and relationships
  - Each model must be designed and implemented

Specify the Solution to the Verification Problem

- Verification environment functional specification
- Dynamic verification
  - Verification IP: in-house and commercial reusable environments
  - Constrained random verification environment
  - Three aspects: stimulus, checking, coverage
  - Top-level design of coverage is part of the scoping process
- Static verification
  - Verification IP: assertion libraries
  - Design properties for each feature

Use the Plan as a Verification Process User Interface

- Write plan in a natural language for human communication
  - Ideas are conceived in our native tongue
  - Abstraction level and ambiguity must be modulated
  - Google “In Defense of Natural Language"
- Machine readable for verification environment linkage
  - Plan-based mapping: plan to verification environment
  - Implementation-based mapping: verification environment back to the plan
- Adapt plan to changing requirements
- Make sure it always reflects your verification goals

Summary

- Unpredictability is rooted in lack of visibility
- Plan the verification of your design
- Increase the value of the verification plan by transforming it into a verification process user interface
- Use the plan to control the verification process
- Questions?