

Coverage Driven Verification with SystemVerilog



Ace Verification

Functional Coverage

- Definition: The functionality required to be tested to achieve the quality goals
 - Test Definition is “how to exercise”
 - Functional Coverage “what needs to be exercised”
- Use: To assess that the Random and Focused tests adequately exercise the design

SystemVerilog Functional-Coverage Example

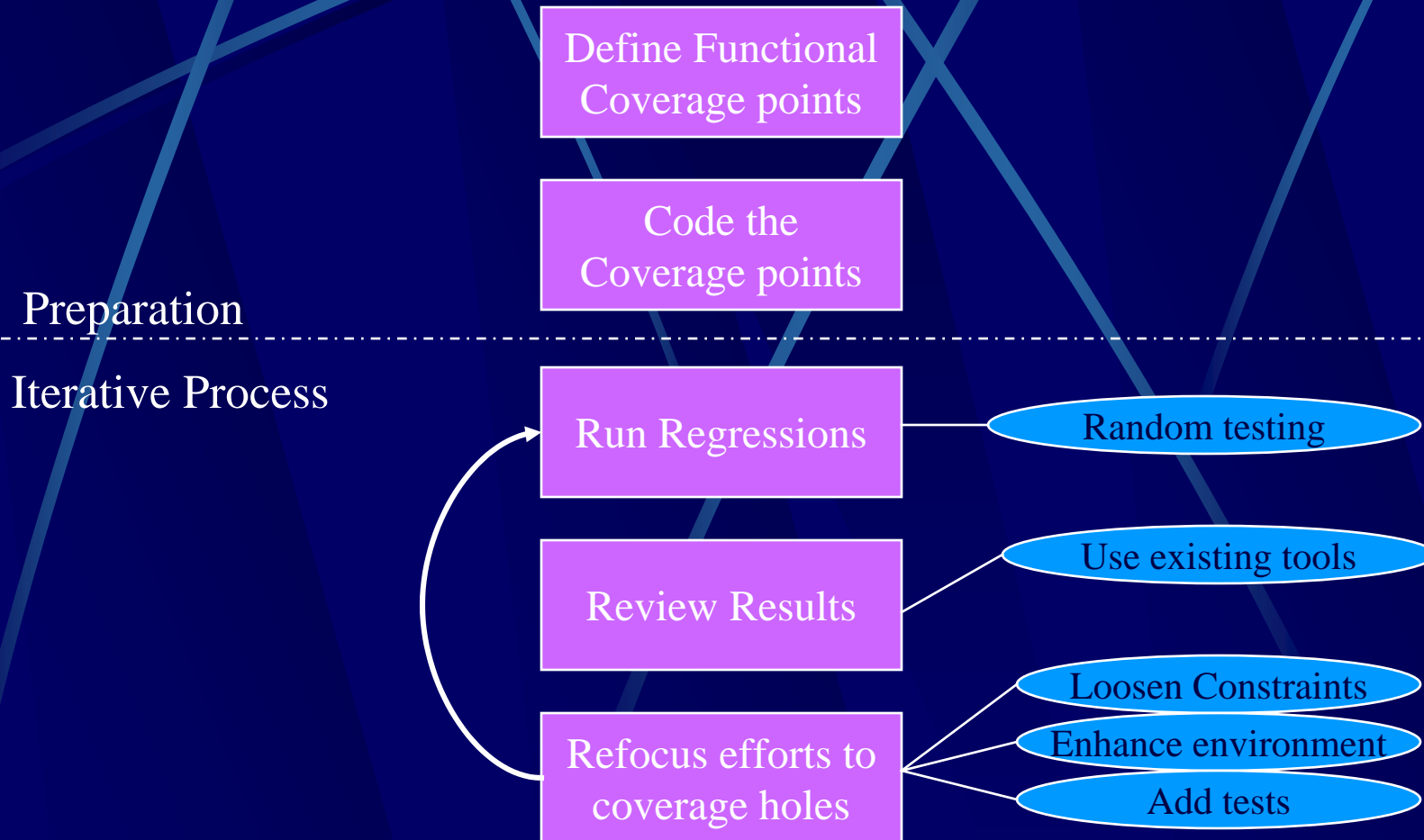
Example Functional Coverage Definition:

IP and IPX packets are sent in both Tx and RX path of the design

SystemVerilog Functional-Coverage Example

```
class s_packet;
  event pkt_sent;
  enum {IP, IPX, RAW} typ ;
  enum {RX, TX, OTHER} direction ;
  covergroup pkt_sent_cvr @(pkt_sent);
    coverpoint typ iff (typ != RAW) ;
    coverpoint direction
    cross typ, direction {
      illegal_bins invalid =
        (direction == OTHER);
    }
  endgroup: pkt_sent_cvr
endclass: s_packet
```

CDV – Flow Chart



Why is the industry moving to Coverage Driven Verification?

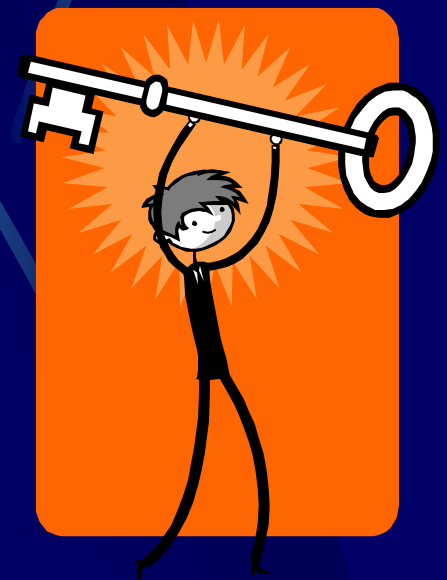
- Focused tests are inadequate in delivering quality designs to fabrication
- Random needs to be aimed in order to be effective
 - Have I covered all of my must-haves?
 - Do I need to focus the random on additional areas
- => CDV facilitates effective Random Verification

CDV – Additional benefits

- Teams define the success criteria for the verification up-front
- The measure of progress is objective (provided by the tools)

Keys to effective CDV

- Goal Definition
- Tracking
- Problem Solving
- Best Known Methods
- Transparency



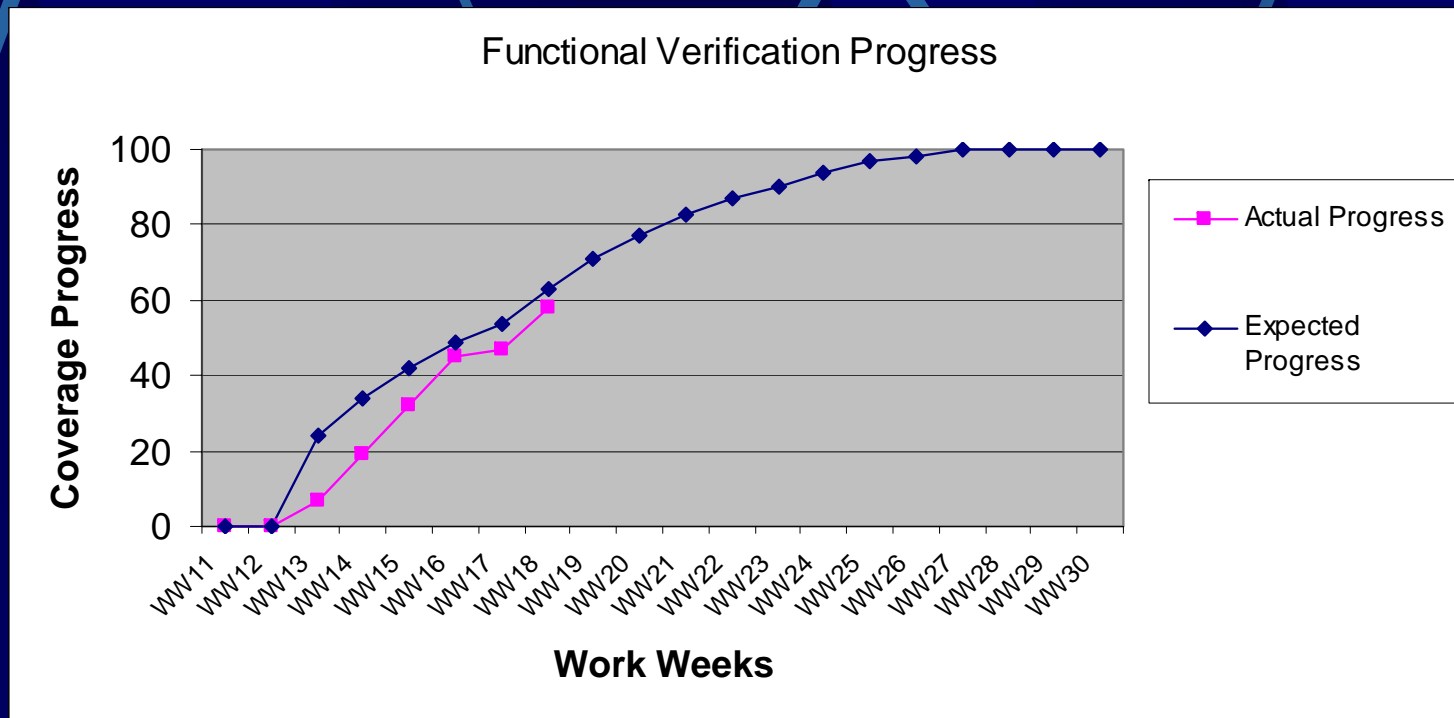
Goal Definition

- Part 1 – Project Goal
- Part 2 – Details
 - Functional Coverage Goals
 - Apply VMM methodology
 - Code in SystemVerilog
 - Assertions and Checks
 - Use SystemVerilog Assertions
 - Enhance with coding styles from VMM

Tracking

- Use functional coverage as the primary metric
- Track planned vs. executed
- Track to 100%

Standard Progress Chart



Problem Solving

- Requirements
 - Clear understanding of problem
 - Functional Coverage report will show lack of progress in feature
 - Cost
 - Overall impact of the problem on overall progress

Best Known Methods (BKMs)

- Use standard interfaces between blocks
 - OCP, AHB etc.
- Use the standard components in VMM
 - Base classes, Messaging, Interfaces
 - vmm_env, vmm_channel etc.
- Use design paradigms
 - Scoreboard, Register Mirroring, Test Flow
 - Etc.

Transparency

- Motivation
- Achieving successful transparency
 - Build an expect graph up front
 - Present it at the same time each week
 - Show progress on individual features
 - No cover-up
 - Allow all stake-holders view of progress

How the keys work together

- Clearly defined goals keep people focused
- The indicator charts give an objective view on overall progress (toward 100%)
 - Verification goal -> Project Goal
- The reports identify problems at the root cause and they get resolved quickly
- Using BKMs shortens design time and reduces the risks of rework
- All stake-holders use the transparency to both boost confidence and preempt problems

Summary

- Coverage Driven Verification
 - Enables effective Random Verification
 - Can be used to achieve predictable schedules

More information

- www.aceverification.com
- “Coverage Revealed” Advanced Training