Analysis and Adaptation

Analysis and Adaptation

- Building a good verification plan is the first step for successful verification
  - But, it is not enough!
- Need to constantly:
  - Monitor the verification process
  - Analyze the observations
  - Adapt to address issues identified by the analysis
- Three basic levels of adaptation
  - Change the way the verification environment is activated
  - Change the verification environment
  - Change the verification plan

37

So far, we discussed how to build a good verification plan. This is a necessary first step for successful verification, but a good plan alone is not sufficient for a successful verification. The reason for that is no matter how careful and detailed our plan is, we cannot foresee everything that will happen during the execution of the plan. Specifically, we cannot foresee when and where bugs will be found. In addition, even a perfect plan cannot guarantee success if its execution is not good enough.

Therefore, throughout the execution of the VP, we need to constantly monitor the verification process, analyze the observations we get, and adapt to address the issues that are pointed out by the observations and the analysis.

Adaptation can lead to changes in all aspects of the verification process, starting from its foundations, namely the verification plan. Simply speaking, the adaptation can be at three basic levels. The simplest level is changes in the way the verification environment is activated. Changes at this level can include adding new stimuli specifications for the stimulus generator or simply changes the number of simulation jobs executed for existing specifications. More complex adaptation require changes to verification environment itself, and in some cases the verification plan need to be adapted to address some issues. Note that changes at a basic level may often require changes at the levels above it as well.

Two Types of Analysis

1. Coverage analysis
   - Was included in the lectures on coverage.

2. Failure analysis

38

The observation data and analysis can come from many sources, and many observations analysis and adaptation come from project management in general, engineering projects specifically, and even more specifically, SW development. Here, we concentrate on items that are specific to verification, namely coverage analysis and failure analysis. Coverage analysis is the main vehicle for measuring the progress of verification against the plan; while failure analysis can help us identify problems in the plan itself.

Failure Analysis

Failure Analysis

- During execution of the verification plan (many) failures are observed
- This is not a bad phenomena
  - Remember that the goal of the verification process is to identify faults in the DUV
- The goal of failure analysis is to understand failures, their causes, their relation to one another, and their relation to the verification process

40

During the execution of the verification plan failures occur. It is important to note that this phenomenon is not necessarily bad, because, after all, the goal of the verification process is to demonstrate that the implementation adheres to the requirements and the specification and this is usually done by removing faults or bugs that prevent it from doing so. The goal of failure analysis is to learn as much as possible from failures and their causes. Failure analysis should lead to fixing everything that needs to be fixed, and this is more than just removing the bug from the implementation. In fact, here we focus on analysis that helps fixing the verification plan and its execution.

Failures and Faults

- **Failure** – an observed DUV behavior that violates the specified behavior
- **Fault** – the root cause of a failure
- There can be a many-to-many relationship between faults and failures
  - Mishandling of overflow in the input FIFO can cause:
    - Lost commands in the output port
    - Bad data in the output port
  - Bad data in the output port can be caused by:
    - Mishandling of overflow in the input FIFO
    - Bad selection in the output selector

41

Before talking about analysis, some terminology and the difference between failure and fault. Failure is an observation that the behavior of the implementation is not what it should be. Fault is the root cause of the failure. Permanent logical and functional faults are often called bugs, and in the discussion here we may interchange faults and bugs. For example, if by mistake we replace an *and* gate with an *or* gate in an adder, a possible failure is getting 1 + 1 = 3, while the fault is the gate replacement.

It is important to note that in many cases there is many-to-many relation between faults and failures. For example, a fault of mishandling overflow in the input queue of the DMA engine can lead to lost commands in the output port or bad data there (two different failures) and bad data may be caused by this mishandling of overflow or selecting a wrong data source.

How Failures Are Detected

- Inspection and code review
- Output of formal verification tools or other static analysis tools, such as lint
- Activation of response checkers during simulation
- Analysis of coverage data
- Visual observation of application misbehavior

42

Failures can be detected in many ways ranging from inspection and reviews (not only of code, but also requirements, specification, verification plans, and more) to visual observation of real-life application misbehavior.

Types of Failure Analysis

- Detailed failure analysis
  - Understand the cause and effects of failures and faults on the design, environment, verification process and more
- Statistical failure analysis
  - Identify trends, provide prediction

43

We can divide failure analysis into two main categories. Detailed analysis looks at a failure or fault (or a small group of them) and tries to understand the cause and effects of the failure. Statistical analysis looks at larger sets of failures and tries to extract statistical information out of them, such as trends and predictions. Statistical analysis is similar in many ways to coverage progress analysis.

Detailed Failure Analysis

- The outcome of the analysis
  - The failure is understood and recorded
  - The failure is resolved
  - The verification plan and process are adapted
  - Lessons learned for the future
- Note: In most cases failure analysis—and especially the last two items—are simple and the outcome of the analysis is that we found a failure and a fault when and where expected and because we are doing our job the right way.

44

We start with detailed analysis, and for the detailed analysis we start at the end. Namely, what is the outcome of the analysis? The outcome has several elements in it. First, we need to ensure that the failure is understood and recorded (this can save a lot of time and effort when similar failures are found in the future). Next, the failure needs to be resolved. This does not necessarily mean that the fault leading to it is fixed. The last two elements concern the verification process. First, we need to ensure that the verification plan and process are adapted accordingly, and that any important lessons learned are applied in the future.

While this sounds scary, it is important to note that in most cases the last two elements are very simple. They say that we found a failure because we were looking for it and we did our job properly.

Understanding the Failure

- The goal is to understand the scope and severity of the failure and how the failure can be recreated
- Provides useful information for debugging and other parts of the failure analysis
  - Simplify and generalize the failure conditions
    - Find simper settings / stimuli that recreate the failure
    - Find necessary and sufficient conditions for the failure
  - Localize the fault in terms of place and time
  - Research: Generate easy-to-debug tests

45

The first thing is to understand the failure in term of its effects. An important aspect here is to find how to recreate the failure because this will lead to the underlying fault. When doing so, we would like to extract as much useful information for debugging and other parts of the analysis. Important aspects here is simplifying and generalizing the failure conditions and localizing it as much as possible.

What to Look For

- In simulation
  - Determinism
    - Does the failure always occur in the same settings?
      - With the same seed?
      - With different seeds (or random seed)?
  - Parameters that are correlated with the failure
    - Parameters that cause the failure to disappear
    - Parameters that cause the failure to change
  - Specific parts in the stimuli that are correlated to the failure
- In formal verification
  - Constraints that affect the failure
  - Time bounds that affect the failure

46

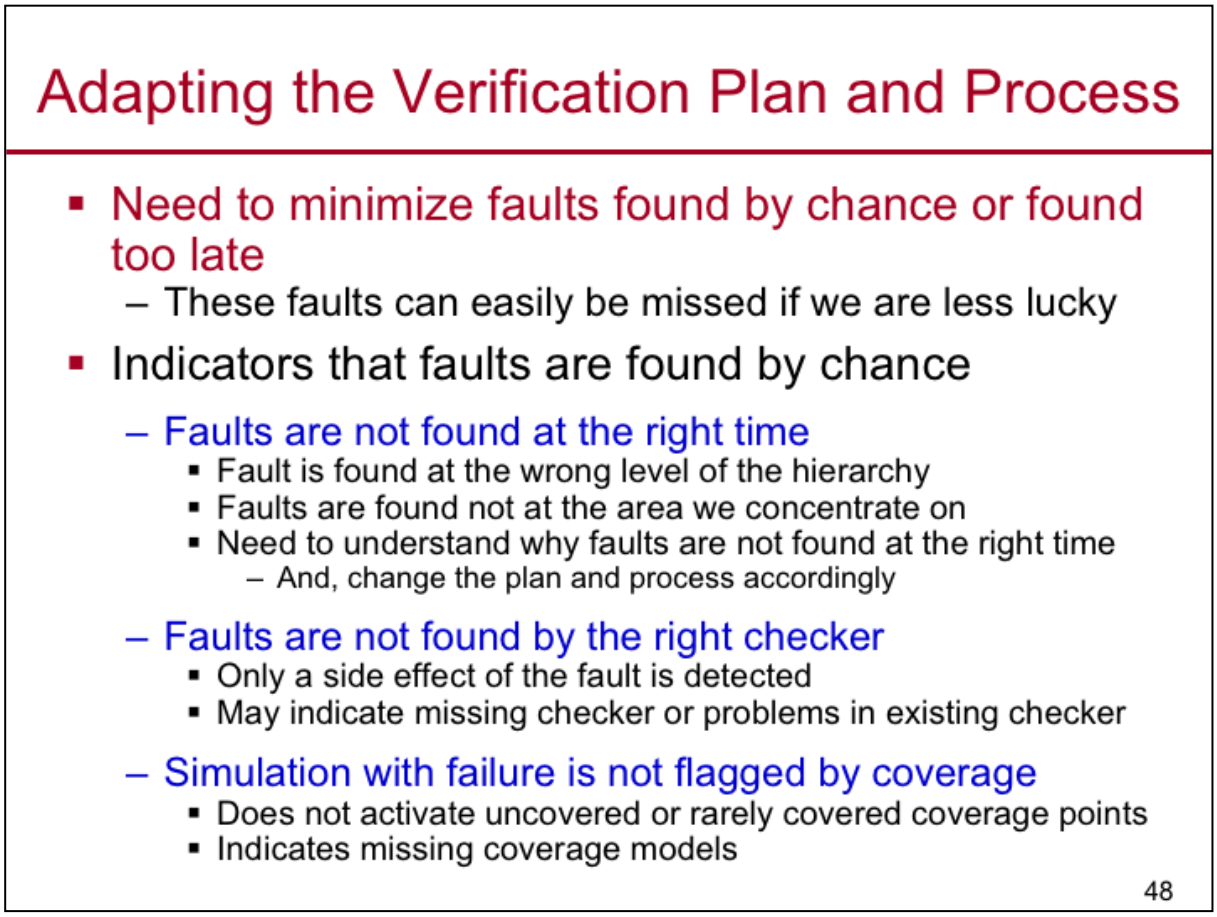
An important aspect in understanding a failure is understanding the conditions in under which it occurs. In simulation, the first thing to check is how deterministic the failure is. In other words, does it always occur in the same settings (with and without the same random seed). Next, it can be useful to understand which parameters in the environment are correlated to the failure. Specifically, we look for parameters that cause the bug to disappear or change behavior. More useful—but in many cases harder to obtain information—is direct correlation between the applied stimulus and the failure. The equivalent to this in formal verification are the constraints and bounds applied in the formal verification process.

Resolving the Failure

- This does not always mean fixing the fault
  - Defer to future tape outs / releases
  - Bypass by software or surrounding modules
  - Record in errata sheets
- Need to ensure that the resolution is complete
  - The fix / bypass is correct
  - All cases are covered
  - No new faults introduced in the process
  - (Similar cases are also handled)
- Mini-verification plan is needed
  - Coverage models
  - Stimuli generation strategy
  - New result checkers

47

The next step is resolving the failure. As mentioned earlier, this does not always mean fixing the bug. No matter what resolution is used, it is important that it is correct and complete and that it did not introduce new failures. It is also important to check that similar faults—sometimes called “cousin bugs” —do not exist or are fixed as well. This calls for a mini-verification plan for the fix (but again this mini-plan can be quite small).

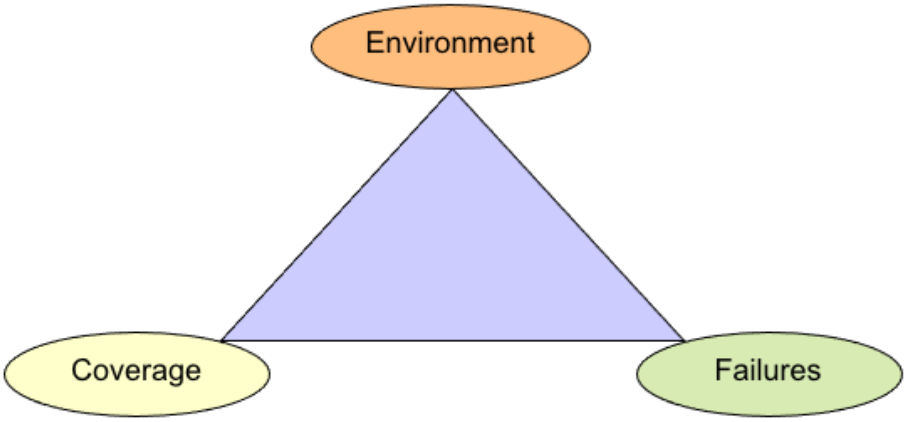


The goal of adaptation is to reduce the number of faults found by chance. Finding faults accidentally is dangerous because these faults can easily be missed if we are less lucky. Therefore, an important part of failure analysis is looking for indications that a fault was found by chance, fixing the verification plan and its implementation to ensure that similar faults are found because we are looking for them. Indications that a fault is found by chance are:

- **Wrong place or time** — wrong level of the hierarchy (e.g., the adder bug is found only at system verification); failures at areas that we do not concentrate on (e.g., the adder bug is found while concentrating on the multiplier). We need to understand why the fault was not caught at the right time and fix the plan accordingly. For example, adding coverage model for the adder operation would have prevented this problem.
- **Fault is found by the wrong checker** — For example the adder bug is found because it caused wrong data length in outgoing message. Here, we need to ensure that proper checkers that catch such failures close to their occurrence are added.
- **Simulation is not flagged by coverage** — If the simulation with failure does not hit an uncovered or rarely covered event, then that simulation is not different from other simulation from coverage point of view. This means that a coverage point or coverage model that corresponds to the failure is missing.

Correlating Coverage and Failures

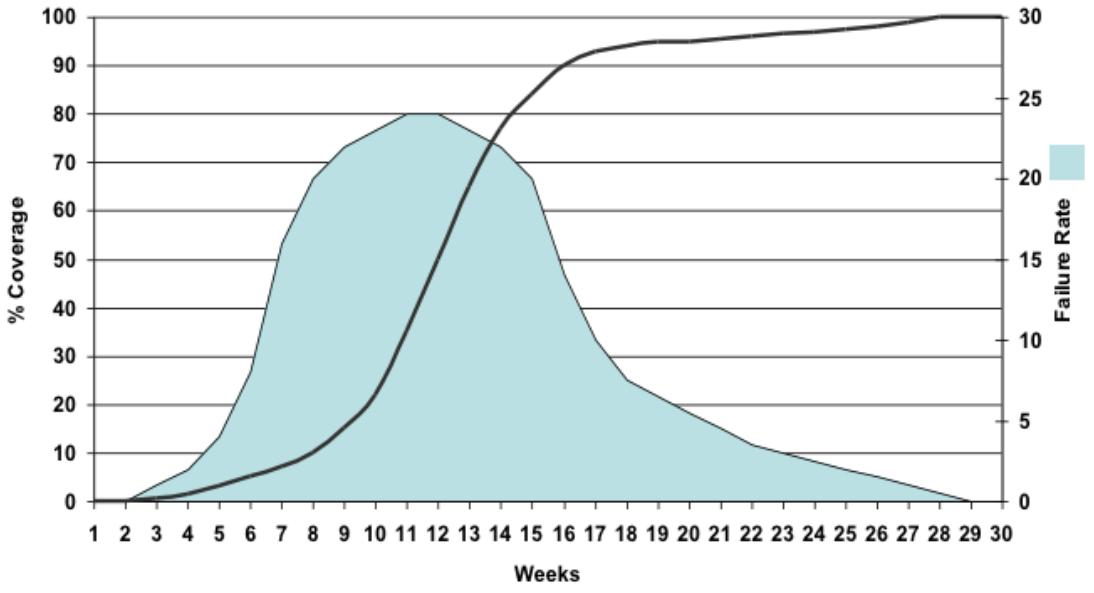
- There is a direct correlation between
  - Changes in the verification environment and the DUV
  - Progress in coverage
  - Detection of new failures



49

An important aspect of analysis and adaptation is the correlation between coverage and failures, correlation between the two of them and changes in the design and environment. For example, activating a new feature in the design should open the door for new bugs to show themselves (as we saw earlier) and also improve coverage.

Correlating Failure Rate and Coverage Progress



50

This plot shows the failure rate (the blue plot, note error in printed version) and coverage progress as a function of time. The plot shows that most failures are exposed when coverage increases at the fastest pace. The explanation for this is simple. When coverage progresses faster, it means that more areas of the design are explored at the same time, and therefore, the potential of finding new problems increases.

Individual Coverage and Failure Correlation

- Correlating a failure to specific coverage can be helpful in the failure analysis and debugging processes
- Rare coverage points exercised by a simulation that fails can hint at the location of the fault that caused the failure
  - Rare coverage points are coverage points rarely, if ever, exercised by passing simulations
  - These coverage points record what happened in the DUV prior to the failure
  - They are very useful if the failure is distant (in logic or time) from the fault or the fault is complex
- If no such rare coverage points are recorded, then it is likely that the failure is found by chance
  - The verification plan needs to be refined to catch these failures

51

Correlating a specific failure with a coverage point can be useful in the failure analysis process. If a failed simulation hits rare coverage points, these points can provide hints about the location of the fault that caused the failure. The rare events are a trail of what happened in the DUV prior to the failure, thus they can indicate where the fault might be. This is especially true if the failure is far from the fault or the fault is complex.

On the other hand, if the failed simulation does not contain rare coverage points, it is likely that the failure was found by chance. In this case we should investigate what is missing in the verification plan and how to refine it so that such failures would not escape.