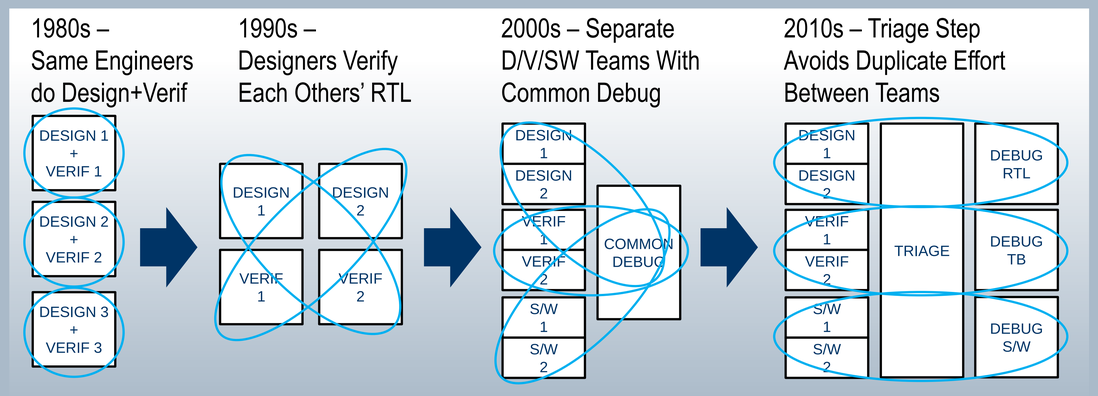
The Evolution of Triage - Real-time Improvements in Debug Productivity

Gordon Allan, Mentor Graphics Corp.

Mentor Graphics Corporation
46871 Bayside Parkway, Fremont, CA 94538

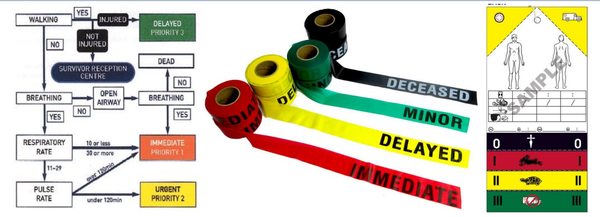
Gordon Allan
Product Manager,
Questa Simulator

Direct: (510)354-5578
Mobile: (510)304-4058
gordon_allan@mentor.com
www.mentor.com/questa



Triage:

microelectronic design/verification : the process of analysis of a discrete set of reported issues of common derivation, including a determination of priority and methodology for further exploration, analysis and ultimately resolution, of each issue, with the ultimate goal of making the most effective utilization of resources to achieve maximum benefit.



10000 tests
1000 fails
100 symptoms
10 root causes

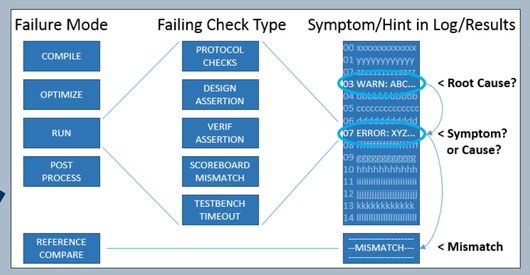
1. **Categorize an individual regression test failure**
2. **Analyze a list of N fails, look for commonality across multiple fails**
3. **Analyze list of NN fails looking deeper for common root causes across multiple fails**
4. **Analyze a single test instance across N regression runs of run history**

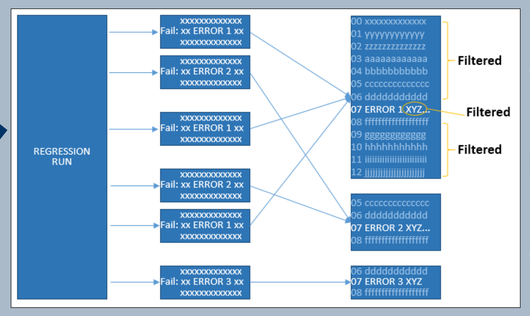
Several techniques can be used here to look at a fail in isolation and make observations about it which will help with other collective failure analysis activity.
The main benefit of this initial preparation in triage is to know which resource or expertise is required and who should dig deeper on this particular fail.

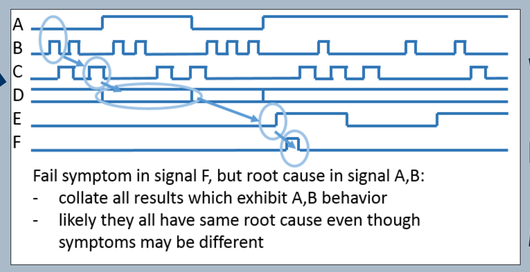
If there are multiple fails, then triage the collection of fails as well as each individual fails, to spot trends and groups. Build or buy a tool or scripting environment that can provide analysis of multiple runs and execute database-query-like analysis on the data:

The symptoms of a failing test may be far removed in time or scope from the root cause of that failure. We may see many failing tests in a regression each with a differing failure symptom, when constrained random stimulus is involved. The ultimate triage productivity benefit is isolating the single biggest root cause, affecting many regression test runs,

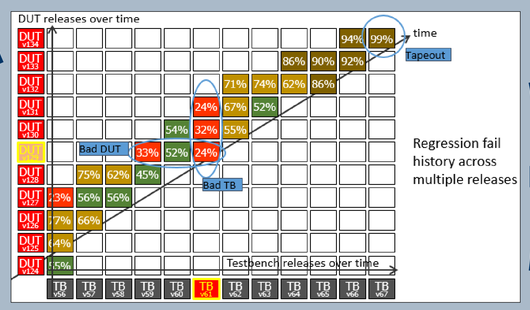
Another way to look at regression data is to look back in time given a common reference point – normally a directed or directed-random test case run with a particular seed and configuration.







Fail symptom in signal F, but root cause in signal A,B:
- collate all results which exhibit A,B behavior
- likely they all have same root cause even though symptoms may be different



Categorize a single regression fail:
- build-time error or a run-time fail condition or postprocessing?
- testbench check or timeout or lack-of-success or mismatch?

Look for root cause indicators:
- find the first failing error message in log, earliest assertion fired
- are there uncharacteristic warnings, early indicators of failure?

Source and significance of the check:
- designer inserted assertion or verification environment assertion?
- scoreboard check or a protocol check - how precise is diagnosis?

What correlation can we see from the point of failure?
- does same error appear in multiple failing test cases or configs?
- can we correlate more cases after filtering out variant data?

Can we add precision to enable tool-based triage analysis?
- categorize assertion behavior across multiple tests
- locate precise signal path and time for further analysis, correlation

If less precision, can we identify interesting commonality?
- which general area of design function or problem signal group?
- anything in common in multiple failing tests to bin them together?

Use available formal technology:
- analyze design around the point of failure
- automatically apply formal property checking to identified region
- spot common design rule violations

Use available debug root cause analysis tools:
- run a causality check in the debug tool
- two pieces of info: signal name, time of failing transition
- derive root cause signal name and time, for further analysis

Repeat results analysis with these 'enhanced' results
- identify common signatures across regression run
- correlate several different symptoms to one root cause

Has this failure mode occurred before?
- when was this observation was last made
- what was deduced about the problem on that occasion?

If this a noisy test?
- does this test fail continuously/regularly or sporadically?
- a new regression in behavior, or is testbench hitting fragility limit?

What can we learn from debug history and records kept?
- when this test/check last failed, what did debug session look like?
- do we already know the likely root cause of this test fail or check?
- who should we talk to to ask, or allocate further debug?

Triage Interactions