

Unifying coverage closure when using different verification techniques

Darren Galpin, Infineon Technologies UK Ltd

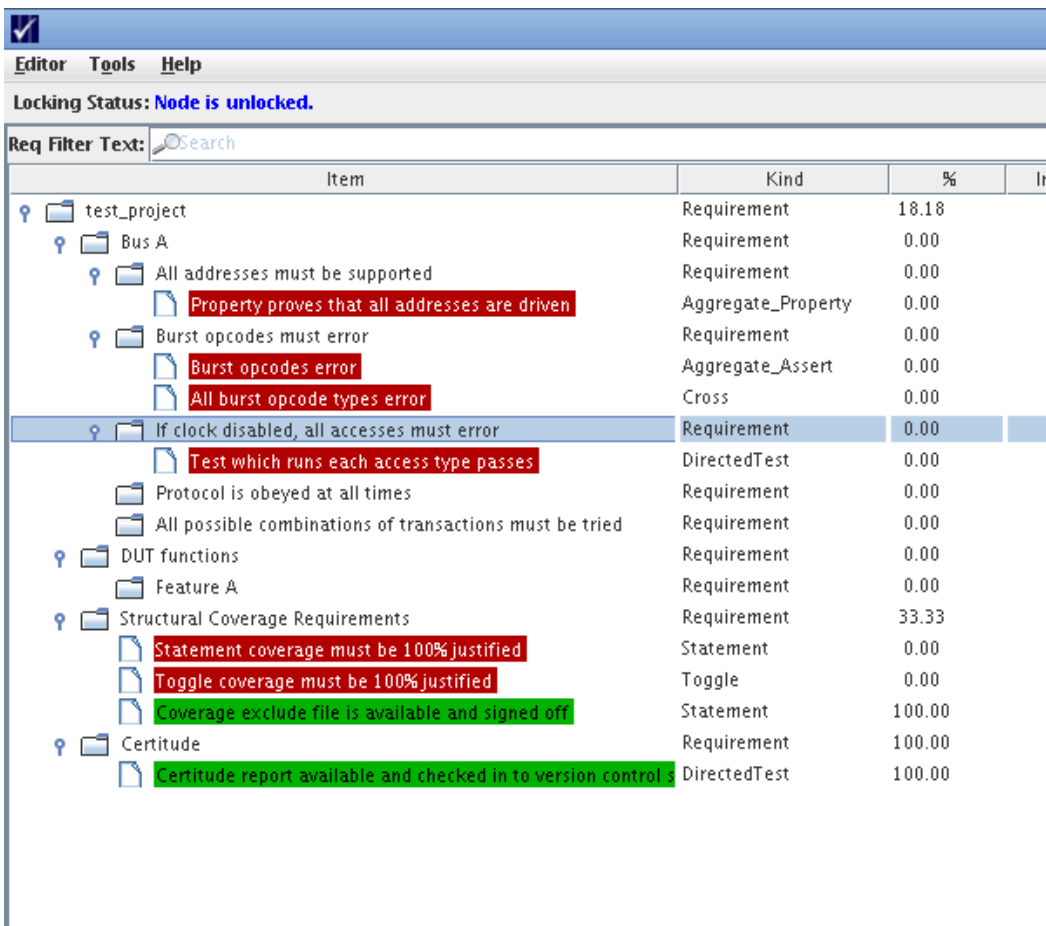


First steps

- › Verification uses different approaches and techniques
 - Random
 - Formal
 - Directed testing
 - Fault insertion
 - Other....
- › All are aiming to do the same thing:
 - Prove given feature or requirement works
 - Prove over a range of conditions
 - Have a plan where if all the checks were switched off and fully random stimuli were applied, your plan doesn't pass.

The plan....

- › Create a plan which is independent of verification tooling.
 - Example from AsureSign (no backhanders from TVS, honest....)



The screenshot shows the AsureSign tool interface. The top menu bar includes 'Editor', 'Tools', and 'Help'. Below the menu, it indicates 'Locking Status: Node is unlocked.' and a search bar for 'Req Filter Text:'. The main area displays a tree view of requirements for a project named 'test_project'. The tree is expanded to show 'Bus A' and its sub-requirements. A table below the tree lists the requirements with columns for 'Item', 'Kind', and '%'. The table data is as follows:

Item	Kind	%	lr
test_project	Requirement	18.18	
Bus A	Requirement	0.00	
All addresses must be supported	Requirement	0.00	
Property proves that all addresses are driven	Aggregate_Property	0.00	
Burst opcodes must error	Requirement	0.00	
Burst opcodes error	Aggregate_Assert	0.00	
All burst opcode types error	Cross	0.00	
If clock disabled, all accesses must error	Requirement	0.00	
Test which runs each access type passes	DirectedTest	0.00	
Protocol is obeyed at all times	Requirement	0.00	
All possible combinations of transactions must be tried	Requirement	0.00	
DUT functions	Requirement	0.00	
Feature A	Requirement	0.00	
Structural Coverage Requirements	Requirement	33.33	
Statement coverage must be 100% justified	Statement	0.00	
Toggle coverage must be 100% justified	Toggle	0.00	
Coverage exclude file is available and signed off	Statement	100.00	
Certitude	Requirement	100.00	
Certitude report available and checked in to version control	DirectedTest	100.00	

- › Break down design into features and/or requirements.
- › All features have at least one check (needed for ISO26262, to prove requirement tested).
- › Add goals which then map to coverage.
- › Assertions used – these can be SVA/PSL/RTL, or from SV/e checks.
- › Properties for more general proofs.
- › Use directed tests where they make more sense.
- › Use manual sign-off to reference reports/files which do not import via UCIS.
- › Will load any UCIS compliant result – structural coverage, functional coverage and formal can be loaded this way.

Coverage closure

- › Now have a combined view of all coverage from all tools in one location.
- › A regression is now the combined set of coverage from all tools, and should be tied to a fixed RTL release.
- › May be covering the same check in more than one approach – save time/effort by mapping to the most efficient one to achieve.
- › If doing full formal proof and can get structural coverage results from the tooling (e.g. OneSpin Quantify), can map to that for the (sub-)blocks proven. Do not then need to do the same in random testbench, for example.
- › Can load in pass/fail results from running tests, allowing directed test mapping if that is more efficient.
- › Can view results over time to see progress in achieving coverage.

Summary

- › Coverage closure best done in a tool independent way.
 - Avoids vendor lock-in, licence deals change.....
 - Allows data from various tools to be brought together for a coherent overview.
- › Coverage closure using combined data allows minimisation of overlap. If one technique is fully covering and checking something, do not need to duplicate in another technique to fulfil different sign-off criteria.
- › Bringing the data together makes it easier to prove that all requirements have been tested. Can generate one report for proof, which makes ISO26262 sign-off and traceability easier.



Part of your life. Part of tomorrow.

