VERISSIMO

SystemVerilog Testbench Linter

Thorough audit of your testbenches

BENEFITS

- Improves testbench code quality and reliability.
- Prevents incorrect functionality and performance issues.
- Automates coding guidelines checking, including UVM Compliance.
- Simplifies code maintenance.
- Identifies dead code and copy & paste code.
- Accelerates language and methodology learning.
- Ensures best coding practices are followed.

OVERVIEW

SystemVerilog provides powerful constructs and a high level of programming flexibility, at the same time introducing new challenges in code development.

For example, the ability to implement the same functionality in multiple ways may impact the simulation performance or lead to unexpected behavior.

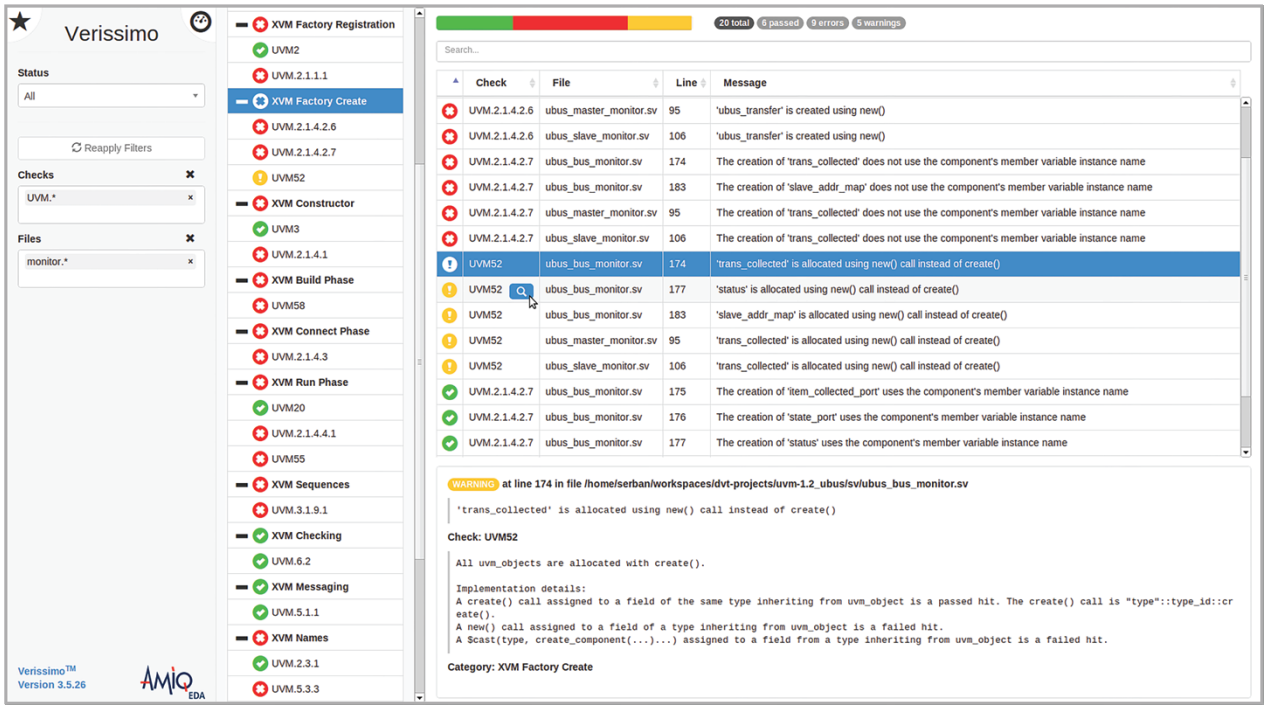
A SystemVerilog compiler checks whether the source code follows the Language Reference Manual (LRM) rules and as such, it flags only language-specific syntactic and semantic errors. However, the absence of compilation errors does not give any indication of code reliability and maintainability. Nor does it imply that best coding practices have been implemented and compliancy with recommended methodologies has been met.

Verissimo SystemVerilog Testbench Linter is a coding guideline and verification methodology compliance checker that enables engineers to perform a thorough audit of their testbenches. With this tool, users can check for language pitfalls, semantic and style issues, and compliance with the appropriate methodologies. Verissimo can be customized to check specific group or corporate coding guidelines to ensure consistency and best practices in code developing.

TYPES OF CHECKS

Verissimo performs a thorough static analysis of the source code. It checks the following areas:

- Suspicious language usage such as non-standard syntax, problematic delta cycle usage, and prohibited system calls.
- Semantic issues that are not caught by the SystemVerilog compiler, for example, an overridden non-virtual method, which will likely result in unexpected behavior.
- Improper styling such as confusing declaration order and naming conventions.
- Verification methodology violations such as inappropriate object creation, missing calls, and constructs that should be avoided.
- Unused code elements such as variables that are never read or written, or functions that are never called.
- Performance issues like not passing arrays by reference to avoid useless copies.
- Copy & paste code duplication.



Verissimo HTML Report

Built-in Repository of Generic SystemVerilog and UVM Checks

Verissimo provides a comprehensive library of generic SystemVerilog and Universal Verification Methodology (UVM) checks. The UVM compliance-checking rules are written in accordance with the verification methodology guidelines from the UVM World (*www.uvmworld.org*).

Customizing the Checks

Users can select from the hundreds of built-in checks in the linter's library only those that correspond to their specific requirements.

Users can customize existing rules by tuning their parameters or create new rules by using a dedicated Java application programming interface (API) that provides access to the linter's internal database.

Integration with the DVT IDE

Verissimo runs both in GUI and batch modes. In the GUI mode, users can perform linting and then visualize the results in the DVT integrated development environment (IDE), which offers an effective way to read and understand the error and warning messages.

Users can quickly jump to the problematic source line to fix the issue flagged by the linter. Then, all they have to do is reapply one or more rules in order to validate the fix.

HTML Report

The Verissimo linter features a report generator that can be used to save the results of a linting session as a text or HTML file.

The HTML report includes a dashboard that shows an overview of the linting status, including information like pass/fail percentage and top failed checks.

The report also comes with advanced functionality for searching and filtering failures. One can use for example author or file filtering to focus the failure analysis on a specific code section.

SUMMARY

The Verissimo linter signals SystemVerilog improper language, semantics, and styling usage, as well as verification methodology violations. It helps improve testbench code reliability, functionality, and maintainability. Verissimo can be customized to meet the demands of small teams, up to larger verification groups or global companies. Ultimately, the Verissimo linter is a tool that allows companies to implement the best coding practices in verification.

The seamless integration between the Verissimo testbench linter as a code analysis tool and the DVT IDE as a code development tool, further improves the verification productivity and quality. It also contributes to decreasing the significant costs associated with code maintenance.

Contact AMIQ

SUPPORT & EVALUATION
support@amiq.com

SALES
sales@amiq.com

WEBSITES
www.dvteclipse.com / www.amiq.com

VER-0617-A4