EZCheck Static Analysis

EZVerify : Verification Productivity with [EZCheck] [EZReport]

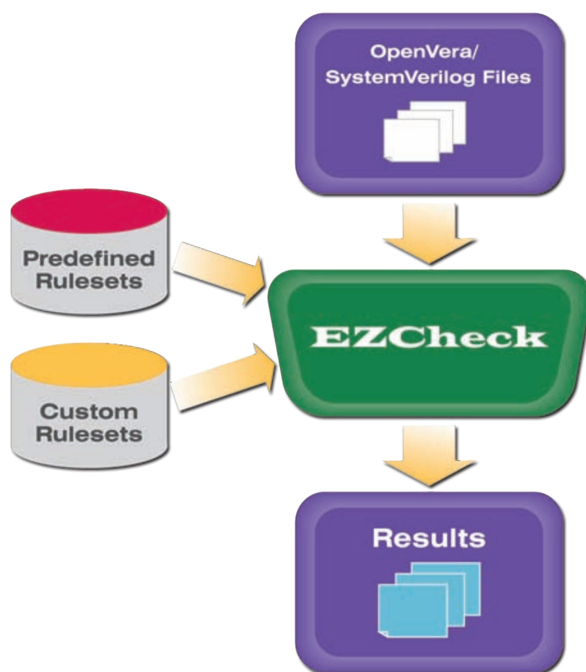
Early Checks Enable Efficient Verification

Hardware verification languages (HVL) have gained widespread popularity in recent times. As in any programming language based flow, HVL-based verification teams should be watchful of errors in code that can lead to deadlock situations, unnecessary debugging cycles and inefficient code. EZCheck statically analyzes code to detect such errors in HVL modules.

The advantage of static analysis extends far beyond traditional error-detection. For example, static analysis data can be used to enable users to put together reusable modules, create portable code or to implement company-wide coding policies.

EZCheck enables users to exploit the power of static analysis for efficient design and verification by providing 325+ predefined rules and several "rulesets" that target error-free code development, best practices for functional coverage model design, object-oriented programming methodology, assertion-based verification and SystemVerilog® Migration.

Use Model



Benefits

- Detect verification errors
- Analyze functional coverage model and implement suggestions for improvement
- Use built-in checks to implement Object-oriented programming policy
- Incorporate best practices for custom assertions in an Assertion-based verification environment
- Provide platform for OpenVera™->SystemVerilog® and Verilog->SystemVerilog® migration
- Implement a corporate-wide coding policy to generate correct, consistent and reusable HVL modules
- Integrate geographically dispersed verification teams

Features

- Full language support for OpenVera™ and SystemVerilog®
- Text and HTML output
- Elegant violation control mechanism to create manageable reports
- Provides in-memory knowledge base that can be accessed with an intuitive API
- Highly extensible - robust API provides foundation for infinite user extensions
- Fits into existing flow
- High-performance - runs very fast
- Platforms - Solaris and Linux

Customizability

- **Input:** Ignore certain files, disable specific rules, provide multiple rulesets and use inlined "pragmas" to inactivate rules
- **Rules:** Specify custom requirement by changing or providing values for predefined rules
- **Rulesets:** Create an infinite number of rulesets by combining predefined and user-defined rules.

VeriEZ Solutions, Inc. is a privately held company that develops and markets solutions that **enable efficient chip verification**. It is the developer of EZVerify and EZTranslate Tool Suites. EZVerify is the industry's first HVL productivity tool suite. EZTranslate is an OpenVera™-to-SystemVerilog® migration tool suite. **More information at www.veriez.com**

EZCheck

Unlock the Power of Rules

EZCheck consists of 325+ rules that users may use to require a certain coding style, enforce a design-wide naming policy and/or restrict usage of certain constructs.

Verification errors: Rules in this category detect errors in HVL code that are not detected by a standard compiler. Examples include incomplete (or missing) assignments to function outputs, uninitialized variables, truncation errors and incomplete case statements.

Language rules: Rules in this category allow users to restrict or disallow the usage of certain language constructs. This feature is significant because a) some language constructs are more error-prone than others and b) downstream tools used in the verification flow may not be able to handle certain constructs.

Naming rules: Rules in this category allow users to establish a module-wide naming policy. Consistent naming schemes will result in manageable and reusable HVL components. For example, enumeration item names can require the enumeration declaration name as a prefix to prevent clashes in the design namespace.

Usage rules: Rules in this category enable the creation of consistent, reusable and easily maintainable code. For example, rules to encourage consistent class design are especially useful for engineers who come from a hardware background.

Assertion rules: Rules in this category enable the creation of consistent and easily maintainable custom assertions. For example, low-performance or non-synthesizable assertions can be detected. User code can be checked against industry best practices.

Enhanced Productivity with Pre-packaged Rulesets

A subset of rules can be assimilated into a "ruleset", enhancing modularity and enabling the use of different rulesets for different applications. Some rulesets included with EZCheck:

Verification Errors ruleset: This ruleset is a collection of rules that warns the user of critical errors in the design. Sample checks are as follows:

- Incomplete assignments to function outputs
- Uninitialized variables, truncation errors
- Mistakes in use of concurrency constructs, missing timeouts
- Use of X or Z in relational, arithmetic and case operations

SystemVerilog® Portability ruleset (OpenVera™ only): This ruleset is a collection of rules that allows the user to check for SystemVerilog® portability of OpenVera™ input. Sample checks are as follows:

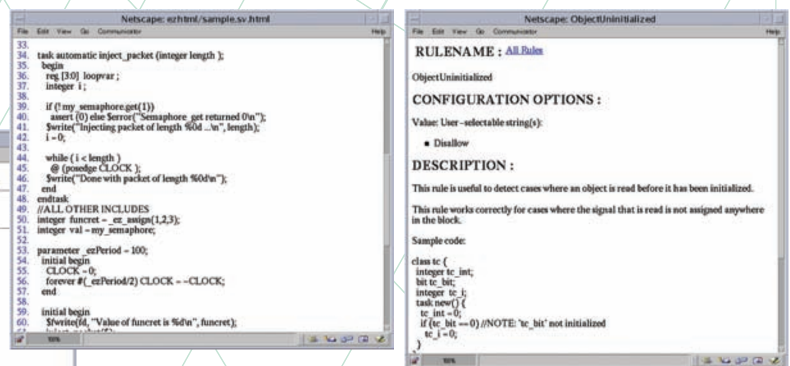
- Blocking constructors and functions
- Use of relatively new language constructs such as aspects, force and release assignments

Assertions ruleset (SystemVerilog only): This ruleset is a collection of rules that allows the user to check for quality assertions in SystemVerilog® code. Sample checks are as follows:

- Incorrect "eventuality" operator usage
- Non-synthesizable operator usage
- Implication in negated property
- $past() with large delays, '0' in delay operators

SystemVerilog® Guidance ruleset (Verilog only): This ruleset is a collection of rules that guides the user to incorporate new SystemVerilog® constructs whenever applicable. Sample checks are as follows:

- Use of new 'always_' process qualifiers
- Branch statements qualified with 'unique' or 'priority'
- Use of clocking and program blocks



The input HVL modules are analyzed for consistency. Violations are reported in an easy-to-understand portable format.

VeriEZ Solutions, Inc. 2086 Walsh Av. #C2, Santa Clara CA 95050 ◈ T: 408 988 1604 ◈ www.veriez.com

All trademarks and registered trademarks are the property of their respective holders.