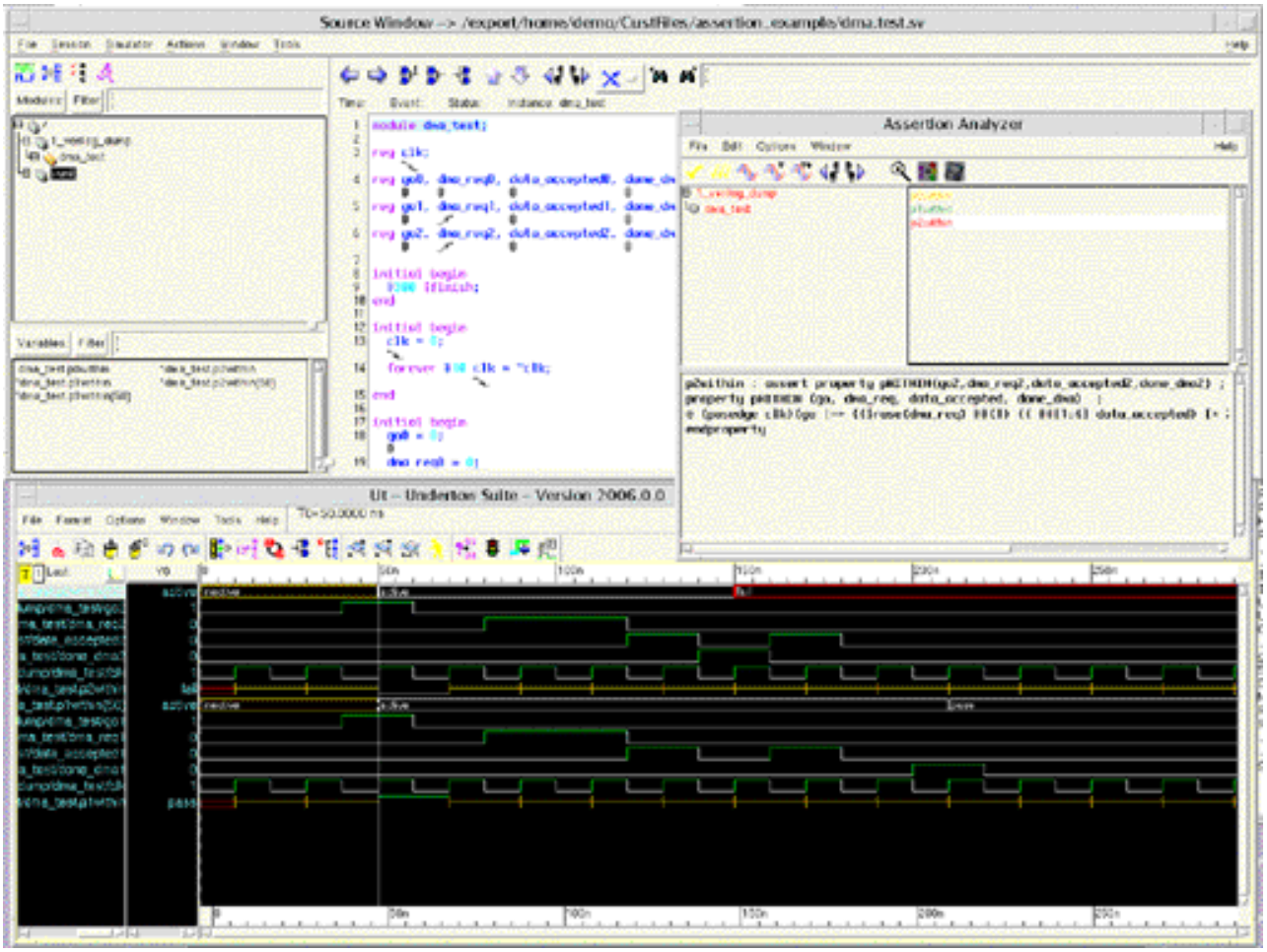
VeritoolsVerifyer

Standalone SystemVerilog Assertion Evaluation, Analysis, and Coverage



SVA Based Functional Verification

- **Manage, design, and evaluate SVAssertions without re-simulating DUT**
- **SystemVerilog Assertions and results are shown in design hierarchy**
- **View timing results for SVAssertion evaluations**
- **View signal components for any SVAssertion evaluation**
- **View timing result for any SVA execution thread along with local variables**

SVA Based Functional Verification Coverage

- **A 'what if' window allows users to create or modify assertions**
- **Users can re-run any assertion evaluation an unlimited number of times without re-simulating**
- **Assertion Coverage metrics provide full Functional Verification Coverage**
- **Test setup and hold times in gate designs using analog simulations result data**

Veritools

Download VeritoolsVerifyer from our web site: www.veritools.com

459 Hamilton Ave., Suite 200, Palo Alto, CA 94301; phone: (650) 462-5590 fax: (650) 462-5593; email: inquiry@veritools.com

Verify your design in a fraction of the time!

Functional Verification

In today's ASIC design environment, the percentage of successful ASIC tape outs has been declining over the past several years, in part due to the fact that ASIC designs have been getting progressively more complex and hence increasingly more difficult to verify and test.

To increase the success rate of new ASIC designs, companies are starting to incorporate SystemVerilog assertions into their design verification process. Many of today's designs have over 30,000 lines of assertion code, the next generation of designs are expected to have over 100,000 lines of assertion code, which will be required just to insure the design under test is working. Design managers and design engineers find that it is getting more difficult to efficiently manage such large amounts of assertion code and to see what the assertion coverage really is. It is even difficult to verify that the assertions are actually testing what design engineers had intended these assertions to test in the first place.

The VeritoolsVerifyer, a stand alone SystemVerilog Assertion analyzer, allows designers to both manage their assertion code, and to quickly verify that the assertions do what the designer had intended them to do.

Manage SVAssertion code; The user's SVAssertion tests and results of evaluations are displayed right in the user's design hierarchy.

Stand alone SystemVerilog Assertion evaluation; The evaluation of the SVAssertions is done directly from a result file generated from a Verilog, VHDL or SystemVerilog simulation. The evaluation of the user's SVAssertions can done if the SVA's are linked to the users design via the "bind" SV function or are embedded into the user's design. The user's SVAssertions can be evaluated an unlimited number of times without requiring any re-simulation. SVAssertions evaluation results are color coded and displayed right in the user's design hierarchy. Yellow indicates the assertion result was vacuously true only, Green, the assertion result was vacuously true or had only evaluated to a "pass", and Red indicates the assertion evaluation failed at least once during evaluation.

Graphical Waveform Result Display; By selecting any assertion, users can display the "assertion results" in the waveform window, and see exactly which assertions reached a pass or fail condition. Assertion result times can then be selected for nay assertion evaluation to display the SVAssertion timing for this SVAssertion evaluation. Users can display on the graphical waveform window the signals that were used in the evaluation for this assertion and can even located and display every independent execution thread for any assertion, along with the local variables that apply to this thread.to find out why any particular evaluation failed. Normally evaluating SVAssertions along with the design result in a significant slow down in simulator performance because SVA assertions having one important dimension not in Verilog or VHDL. These languages have declaration and instantiation, SVAssertions have declaration, instantiation.and execution. Each and every SVAssertion can and sometimes will execute each and every single clock cycle even if it is already being executed. Because this tool uses the same simulation result files that are generated during normal design simulations, there is never any negative impact at all on simulation run times when adding SVAssertions to the user's verification process.

"What If"' Capability; VeritoolsVerifyer provides a "What if" capability to allow the user to find and fix the conditions that may have prevented an assertion from going to a assertion pass condition. If any assertion fails, the user can bring up the assertion execution that fialed, modify the assertion and re-evaluating this assertion. The user can use this window to split apart assertion expressions into several smaller assertion expressions to find out exactly what part of the assertion expression has passed and which part of the complete assertion expression is failing. This process can be repeated an unlimited number of times to quickly find exactly prevented the assertion from reaching the assertion pass condition and to find the correct SVAssertion code that will pass. The assertion expression evaluator in the "What If" window can generally re-evaluate any modified assertions in seconds, even while using very large simulation result files.

In a simulator only based approach users have to;

Run simulation and find the assertions that are failing
Load the source code for the failing assertion into an editor
Edit the failing assertion code
Commit the new code and re-compile the source code
Rerun the design and assertion simulation
Reload the simulation results into an assertion analysis tool
Re-verify that the new assertion is or is not working correctly
Repeat this process if the assertion is still not working

Using the "What if" capability users can;

Run assertion evaluation, find the failing assertions, Press Edit
Edit the SVAssertion code for any assertion that failed
Press "Test Edited Assertion", result showing whether assertion is working displays on waveform window, in most cases in seconds.
Repeat from "Edit the SVAssertion", as many times as necessary.

Users can modify, re-evaluate and re-verify that the new SVAssertion is working or not in most cases in seconds. Using only a simulator-only based approach to verify SystemVerilog Assertion code, each iteration could take as much as several hours. A new simulation run is only required, when using the VeritoolsVerifyer when a change is required to the user's design or test vectors.

SVAssertions and Gate Design: The exact same SVAssertions that are run against the RTL design can be also be used to verify the gate design that was synthesized from this RTL code. Users only have to insure their SVAssertions take into account the actual clock timing seen at the clock input of each FF. Veritools provides an entire set of tools to allow users to use analog simulation results with extracted net lists to achieve the highest possible accuracy in the final SVAssertion analysis of FF set-up and hold timing.

Functional Verification Coverage

SVAssertion Coverage with complete assertion metrics: The VeritoolsVerifyer includes SVAssertion coverage so users and design managers can see the overall coverage of the SVAssertions in their functional verification process. Results are accumulated and tabulated so users can see a percentage of functional coverage for any part of their design, and can quickly go to any part of the design where additional assertion tests may be needed.

No other tools have these features at any price.

All of the features of VeritoolsVerifyer are available in both interactive mode, or in batch mode for use in virtual simulation, without using a simulator license. These tools support **Verilog, Verilog 2001, VHDL, SystemC** and **SystemVerilog**, and in addition supports standalone SystemVerilog assertion evaluations including a complete suite of tools to do assertion metrics for coverage analysis. Veritools products are available on Linux systems, 32- and 64-bit Sun _Solaris, HP700/800, IBM AIX, and Windows 98/NT/2000/XP. Copyright 2006, All Rights Reserved, Veritools, Inc. Trademarks are owned by their respective