

Riviera™



SystemVerilog in Riviera

Synopsys EDA Interoperability Developers' Forum

Santa Clara, CA

April 7th, 2005

ALDEC – Brief Introduction

- Developer of mixed VHDL, Verilog and EDIF common kernel simulation technology
- Majority of revenue put back into Research & Development
- Specializes in mixed HDL, EDIF and co-simulation, hardware acceleration and co-verification, and team based development and documentation solutions
- Holder of several key patents in verification technology
- European based R&D Team
- Established in 1984
- First HDL simulator released in 1997
- Multi-platform Riviera simulator entered the market in 2001

SV Support Level in Riviera

- ❖ Riviera 2005.04 supports the assertions constructs of SV 3.1a
- ❖ Riviera supports selected SV 3.0 constructs (both testbench and design subset)
- ❖ Additional support for SystemVerilog constructs is planned (level of support depends on SystemVerilog standardization by IEEE)

SV Impact on Hardware Designers

- ❖ Almost all old Verilog designs will work in SystemVerilog. The two exceptions are:
 - Designs using identifiers that became SV keywords
 - Designs that utilized erroneous behaviors of original Verilog corrected in Verilog 2001
- ❖ New designs will benefit from many enhancements:
 - User-defined types and richer set of built-in data types
 - Enhanced behavioral modeling constructs
 - Enhanced structural modeling constructs

SV Behavioral Modeling in Riviera

- ❖ **Explicit Specification of Combinatorial Block of Logic (`always_comb`)**
- ❖ **Explicit Specification of Latch (`always_latch`)**
- ❖ **Explicit Specification of Flip-flop (`always_ff`, `iff`)**
 - No need to worry about specifying correct sensitivity list for the *always* block
 - Both simulator and synthesizer treat this description exactly the same way

SV Behavioral Modeling in Riviera

- ❖ **User-defined Data Types**
(`typedef`)
- ❖ **Enumeration Types**
(`enum`)
- ❖ **Time Units and Precision Tied to Module**
(`timeprecision`, `timeunit`)
 - Clarification designer's intention
 - Code becomes more readable
 - Lower probability of introducing accidental errors during design modifications

SV Structural Modeling in Riviera

- ❖ **Implicit *.name* Port Connections**
(latch U1 (.CK, .D, .Q);)
- ❖ **Implicit *.** Port Connections**
(gates U2 (.*);)

- **Faster than traditional port connection**
- **Clean and safe!**

SV Impact on Design Verification

- ❖ Old Verilog testbenches will work in SystemVerilog
 - New keyword restrictions still apply
- ❖ New testbenches and system-level designs will benefit from many enhancements:
 - New **final** procedural block
 - New loop statements (**do..while**, **break**, **continue**)
 - Simpler syntax of tasks and functions
 - New, C-like operators (**+=**, **-=**, **++**, **--**, etc.)

High-level SV Constructs in Riviera

- ❖ **The final Procedural Block**
 - ❖ **New do...while Loop Statement**
 - ❖ **New break and continue Loop Statements**
-
- **Better handling of 'end of simulation' tasks**
 - **Increased functionality of loops**
 - **Better handling of exceptions in loops**

High-level SV Constructs in Riviera

- ❖ **Simpler Syntax of Functions**
(no `begin..end`, the use of `return`)
- ❖ **Simpler Syntax of Tasks**
(no `begin..end`, the use of `return`)
 - More compact descriptions of tasks and functions
 - More flexible specification of value returned by the function
 - Easier coding of immediate task exit

High-level SV Constructs in Riviera

- ❖ **Interfaces**

- ❖ **Built-in C Types**

(`bit`, `byte`, `int`, `shortint`, ...)

- ❖ **New Operators**

(`++`, `--`, `+=`, `-=`, `*=`, ...)

- Very important in system-level descriptions
- More compact descriptions
- Faster simulation

SystemVerilog Assertions

- ❖ **Full Support of Immediate Assertions**
(`assert (expression) action_block)`
 - Non-temporal expression check
 - Immediate or delayed actions
- ❖ **Constantly Improved Support of Concurrent Assertions**
(`assert property(property_specification / property_instance)`)
 - See next slide

SystemVerilog Assertions

- ❖ **Limited Support of Property Declarations**
 - Recently added implications and *disable iff()*
 - No formals
 - No local assertion variables
- ❖ **Almost Complete Support of Sequences**
 - All delays, ranges, repetitions, etc.
 - Majority of sequence operations (no *within*)

SystemVerilog Assertions

❖ Special Feature

- Backward compatibility with assertion syntax as described in SV 3.0 and SV 3.1

❖ Remaining Implementation Tasks

- Sequence operator `within`
- Assertion variables
- Subroutine calls on sequence match

Roadmap

- ❖ High level constructs:
 - Structures
 - Unions
 - Classes
 - ❖ Coverage
 - ❖ Completion of assertion support
- 👉 **Schedule depends on user demand and IEEE 1800 standardization**

How to Enable SV Support in Riviera

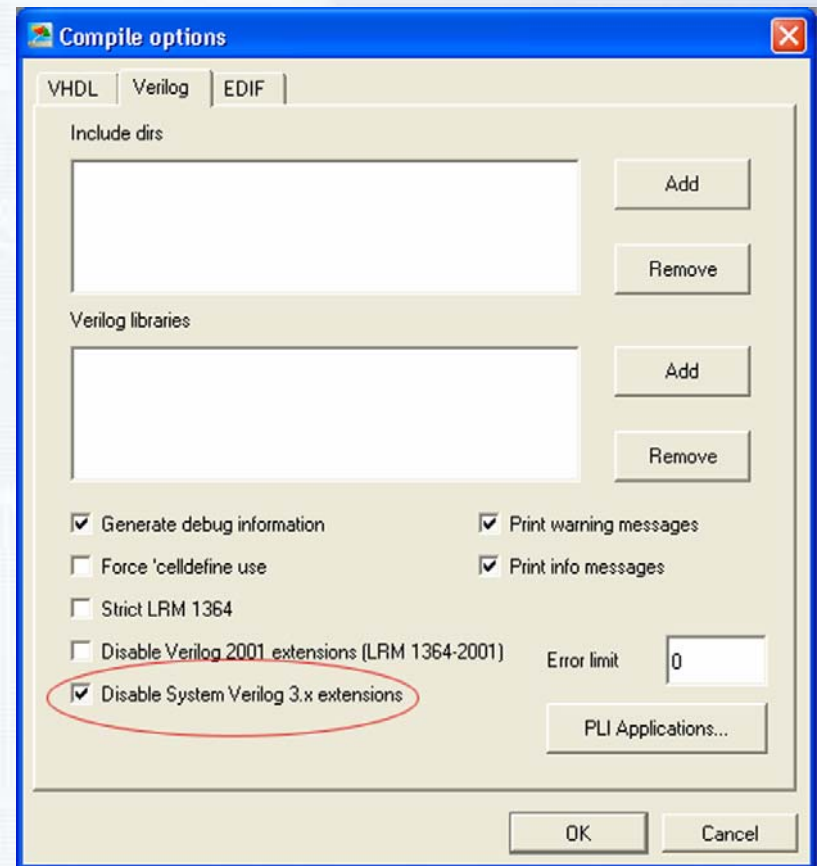
If you are compiling source files from Riviera GUI, go to:

Compilation → **Compilation Options**

in the main menu and clear checkbox named:

Disable SystemVerilog 3.x extensions

in the Verilog tab.



How to Enable SV Support in Riviera

- ❖ If you are compiling source files from the script or command line, remember **not to use** any of the two switches in the **alog** command :
 - v95 Verilog 95 only (disable Verilog 2001 and SystemVerilog)
 - v2k Verilog 2001 only (disable SystemVerilog)
- ❖ If any of the two switches is used, SystemVerilog constructs will be flagged as errors during compilation