



# SystemVerilog – From a User's Perspective

Neil Korpusik

Staff Engineer

Front-end Technologies – ASICs and Processors

10/21/04



## History of Verilog

- Verilog-95
  - first version standardized by the IEEE, 1364-1995
- Verilog-2001
  - a few minor adjustments to Verilog-95 (a.k.a. V2K)
  - didn't create much of a stir within Sun
- SystemVerilog
  - a major overhaul to V2K
  - “When can we start using SystemVerilog?”  
I keep getting asked this question...



## Verilog usage at Sun

- In use since 1987-1988
  - before Cadence placed Verilog in the public domain
  - the move to Verilog is viewed as a positive one
- Verilog exclusively
  - no VHDL
  - all projects use Verilog
- All levels of design
  - ASIC
  - Sparc microprocessors
  - Board level
  - FPGA
- Used in the following product lines
  - SunRays
  - Workstations
  - Servers
  - Network storage products

## Testbench Progression at Sun

- Verilog
  - used initially
  - ran out of steam for complex test benches
    - re-entrant tasks
    - dynamic threads of execution
- Vera
  - initial development was done at Sun, 1994
  - productized by Systems Science and Synopsys
- Native testbench
  - Vera capability built into VCS
- SystemVerilog
  - 3.0 very few Vera capabilities (V2K re-entrant tasks)
  - 3.1 many Vera capabilities – some key features are missing
  - 3.1a has all the necessary testbench features

## Assertions at Sun

- Long history of assertions
  - Verilog monitors
  - 0-in directives
  - OpenVera Assertions (OVA)
  - Targeting SystemVerilog Assertions (SVA)
- Example usage
  - Simulation monitor
  - Isolating corner case bugs
  - Verifying bug fixes
- Number of assertions
  - A few thousand up to 10's of thousands

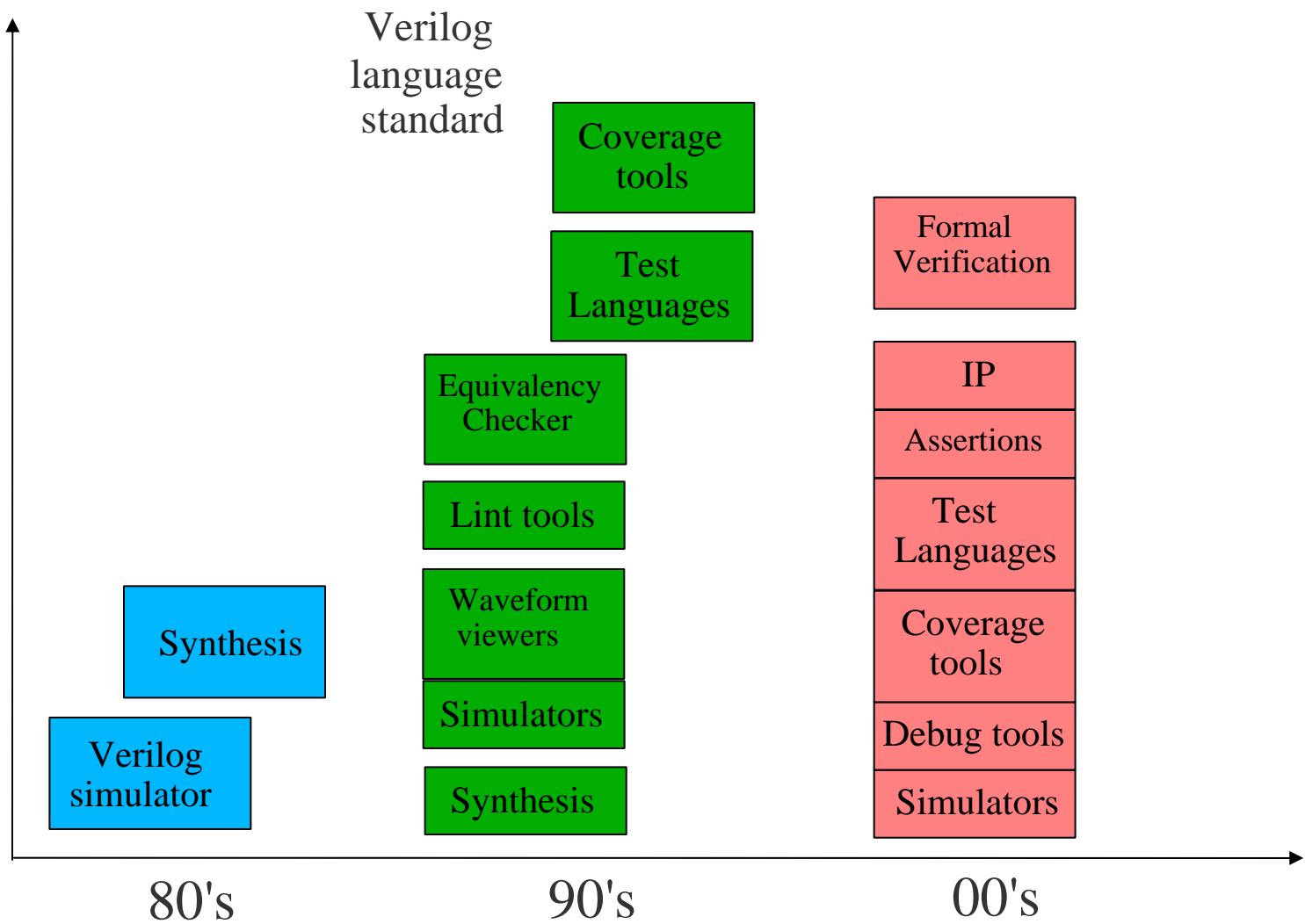
## SystemVerilog features Sun Expects to Use (most features)

- Testbench
  - program block
  - clocking block
  - classes
  - functional coverage
  - fork/join
  - fine-grain process control
  - ref argument passing
  - random constraints
  - interprocess synchronization and communication
  - packed array, dynamic array, associative array
  - streaming operators (pack/unpack)
- Assertions
- interfaces
- enum
- struct
- queue
- New operators (++, --, +=, etc.)
- DPI and APIs

## Why SystemVerilog?

- One integrated language
  - performance boost – much less VPI/PLI usage
  - designer productivity boost
  - fewer tool integration issues
- Natural progression of tool/language capabilities already in use at Sun
  - testbench
  - assertions
  - Verilog
  - “Verilog on steroids” - Stuart Sutherland
- Potential for spawning a new wave of capabilities
  - Verilog + synthesis was an industry enabler
  - SystemVerilog has a similar potential

## How the industry is evolving



## *Obstacles to adoption*

- EDA tool availability
  - *the language is in good shape, it's the tools that are lagging*
- Learning curve – there are a large number of enhancements

## History of SystemVerilog

- Accellera
  - 3.0 – design constructs – Superlog donation
  - 3.1 – testbench, assertions, DPI
  - 3.1a – functional coverage, vpi, features missing from 3.0
  
- IEEE P1800
  - Starting with 3.1a as a basis
  
- Sun Involvement
  - Accellera 3.1 and 3.1a
    - svec - testbench
    - svac - assertions
  - P1800
    - errata sub-committee
  - Strong advocate of SystemVerilog

## Messages to vendors

We want SystemVerilog support and we want it now!

- Lack of SystemVerilog support could potentially impact tool adoption
- Areas of opportunity
  - linter – not just for the design subset, must also cover testbench
  - others?