# SystemVerilog Assertions Handbook, 3rd edition

## … for Dynamic and Formal Verification

**Ben Cohen**
**Srinivasan Venkataramanan**
**Ajeetha Kumari**
**...and Lisa Piper**


**VhdlCohen Publishing**
**Los Angeles, California**
**http://www.SystemVerilog.us/**

# SystemVerilog Assertions Handbook, 3rd Edition

## … for Dynamic and Formal Verification

**Published by:**
**VhdlCohen Publishing**
**P.O. 2362**
**Palos Verdes Peninsula CA 90274-2362**
**ben@SystemVerilog.us**
**http://www. SystemVerilog.us/**

---

**Library of Congress Cataloging-in-Publication Data**
**A C.I.P. Catalog record for this book is available from the Library of Congress**

**SystemVerilog Assertions Handbook, 3rd Edition**
***… for Dynamic and Formal Verification***
**ISBN 878-0-9705394-3-6**

**[1] Reprinted with permission from IEEE Std. P1800/D5, 2012 -prelim Standard for SystemVerilog Unified Hardware Design,Specification, and Verification Language, Copyright 2012, by IEEE. The IEEE disclaims any responsibility or liability resulting from the placement and use in the described manner.**

**Items reprinted from the above referenced IEEE document are identified with a prefix [1] and are shown in italic font.**



**Printed on acid-free paper**

**Printed in the United States of America**

# Contents

# FOREWORD, Dennis Brophy

In the decade since the completion and release of the first version of the IEEE SystemVerilog standard, the use of assertions in verification has taken center stage. In as much as design and verification teams have been able to use assertions during functional verification tests, they have proven even more valuable to open the world of formal verification to more users to perform exhaustive block-level and interface tests.

We know debug of electronic systems is an ever increasing challenge given the relentless increase in design complexity and protocols engineers must use in their designs. Many design issues are buried deep in a system and can be difficult to reach or detect in a timely fashion even with today's automated constrained random stimulus generation solutions. The embrace of a design reuse paradigm that allows design teams to pull predesigned blocks into their design has led to the creation of the Verification Intellectual Property (VIP) business. The various protocols and blocks that now come together to comprise a design come with VIP that will include assertions to detect illegal use and conditions that might otherwise be difficult to detect. VIP also lessens the need for design and verification teams to have deeper protocol expertise and knowledge. All this is enabled with assertions.

In 2010, research showed use of SystemVerilog was up more than 233% over prior years with more than 7 out of 10 design and verification engineers using it. Even more telling was the use of the SystemVerilog Assertions (SVA) part of the standard. Research showed that *assertions* enjoyed the same high level of use with 7 out of 10 design and verification engineers adopting SVA.

Assertion based verification (ABV) methodologies has been found to address design and verification challenges and the market use reflects it. The assertions portion of the IEEE SystemVerilog standard has also been enhanced over these years to extend and improve what can be done with them based on the cumulative experiences of the design and verification community to date.

The third edition to the *SystemVerilog Assertions Handbook* comes at a time when the IEEE updates its popular SystemVerilog standard and at a time when the FPGA community is increasing its adoption of SystemVerilog assertions as well. Design and verification engineers will find the handbook useful not just as a resource to begin to adopt assertions, but to apply the latest additions and updates found in the IEEE standard to the ever pressing design and verification challenges.

**Dennis Brophy**
Director of Strategic Business Development
Design Verification Technology Division
Mentor Graphics Corporation
http://www.mentor.com/

# FOREWORD, Sven Beyer

In 2005, SystemVerilog assertions became part of the IEEE 1800 standard. Now, 7 years later, there is no longer any doubt about the fact that Assertion-Based Verification has become a mainstream technology: according to the 2010 Wilson Research Group Functional Verification Study, assertions are used in roughly two thirds of all projects – from rather simple automatically generated assertions to capturing complex bus transactions in assertions. Interestingly, the usage of formal ABV has also increased by 50% from 2007 to 2010, with a projected further increase in the coming years. Nowadays, companies that have not yet adopted assertions feel a strong need to do so in order to catch up with their competition. Finally, SVA is the dominant assertion language, making up the lion's share of 75% of all assertions.

In addition to the growing acceptance and tool support of SVA over the last 7 years, the standard itself has been very much alive with two updates in 2009 and now in 2012, enhancing many existing features and adding numerous new ones. So in summary, more and more engineers are exposed to SVA while at the same time, the standard quickly evolves, trying to address the growing needs of those engineers for more productivity. This definitely calls for a first class reference documentation – and this book, *SystemVerilog Assertions Handbook, 3rd Edition* by Ben Cohen, Srinivasan Venkataramanan, Ajeetha Kumari, and Lisa Piper, provides such a comprehensive reference manual that is suited for both SVA power users and novices. It introduces assertion methodologies and gives a clear idea on what assertions are good for, addressing both coverage and the complementary strengths of dynamic and formal verification. It carefully lays out the numerous SVA language constructs one by one in a way that really gets across to the typical engineer, emphasizing the intended usage, adding telling examples, and listing the counter-intuitive pitfalls that may cost an engineer precious time in debugging. Therefore, this book is sure to find its place on the bookshelf of numerous engineers all over the world, and since it is the first comprehensive reference manual to also address the IEEE 1800-2012 standard, for example with its numerous enhancements to the checker construct, it is sure to remain on this shelf and be extensively used for quite some time.

**Sven Beyer**
Product Manager Design Verification
OneSpin Solutions
http://www.onespin-solutions.com/

# FOREWORD, Stuart Sutherland

The fact that you picked up this book means that you are most likely already aware of the benefits of using temporal assertions to verify hardware functionality in hardware design models. These benefits include, but are not limited to, proving that actual design functionality matches (or does not match) the intent of a design specification, localizing design bugs in large, complex models, and significantly reducing the amount of verification code required to gain confidence that a design is functionally correct.

As one who has been teaching and consulting on Verilog and SystemVerilog for many years, I can attest first-hand to those benefits. On one project, I was contracted late in the design cycle to help create a more robust verification environment. As is often the case, many of the engineers working on the project wore two hats. Early in the project they worked on modeling the RTL code. As the design progressed, they transitioned to verifying the RTL models and the post-synthesis gate-level models. The RTL modeling of design was complete when I arrived on the scene, and the engineering team felt the RTL models that made up the design had been thoroughly tested and worked correctly. As I reviewed the verification process, I noted that no assertions had been used, and saw several places where simple assertions – often just one-line of functional code – could easily be added. I wrote about 20 SystemVerilog temporal assertions, and immediately found several places where the design functionality did not match the design specification. The same verification stimulus was used, but these few temporal assertions detected some real design bugs that the "thorough" verification had missed. The assertions also identified just where in the full design the problem first showed up. There was no need to spend hours tracing back in both logic and time from an incorrect output value to try to troubleshoot the cause of a problem. SystemVerilog Assertions did an excellent job of saying *something has gone wrong, and the problem is right here!* The benefits of assertion based verification are very real.

The complexity of the design we need to verify requires that an assertions language has a robust set of features and capabilities. The SystemVerilog Assertions (SVA) language meets that rigorous requirement. The robustness of SVA also means that it can be challenging to learn to use SVA -- and to use it correctly. The *SystemVerilog Assertions Handbook* is an essential resource for overcoming that challenge. The book examines the use of SVA in the context of verifying true-to-life designs. Thorough explanations of each feature of SVA show the where and how to use SVA correctly, as well as point out pitfalls to avoid. At my company, we feel this book is so essential for understanding and properly using SVA, that we include a copy of the book as part of the standard training materials in all of our "SystemVerilog Assertions for Design and Verification Engineers" training workshops.

Stuart Sutherland
SystemVerilog Training and Consulting Wizard
Sutherland HDL, Inc.
http://www.sutherland-hdl.com

# FOREWORD, Cristian Amitroaie

As usual, Ben keeps up with the latest trends in our industry. This time he focuses on the new SystemVerilog assertion capabilities in the IEEE 1800-2012 standard update, including the checker construct.

The first benefit this book brings is a systematic and clearly organized perspective on SVA, from planning to terminology, from how assertions work and how to debug them, to coverage driven and formal verification using assertions. This includes the language clearly identified rules, and many tables and figures annotated with comments.

Second it offers many concrete examples. Examples are fresh air for engineers when diving into complex topics and this book has plenty, including the mapping between natural language and the corresponding SVA implementation.

Third, it contains guidelines on what to use and what to avoid, based on experience with both SVA and UVM. Knowing and following best practices are essential to engineers these days, when work pressure doesn't leave much time to carefully digest all the implications of the highly sophisticated means we use on a daily basis.

This is a book every engineer should keep handy!

**Cristian Amitroaie**
CEO
Amiq.com
http://www.dvteclipse.com

# PREFACE

## What's new?

*SystemVerilog Assertions Handbook, 3rd Edition* is a follow-up book to the very popular and highly recommended second edition, published in 2010. This 3rd Edition is updated to include the new SystemVerilog assertion features, enhancements, and clarifications presented by the *IEEE 1800-2012 Standard for SystemVerilog Unified Hardware Design, Specification, and Verification Language* (herein referred as IEEE 1800-2012, or LRM – language reference manual).[1] The 2012 LRM changes include several enhancements for properties and sequences, particularly in the area of immediate assertions, data type support, argument passing, vacuity definitions, global clock resolution, and inferred clocking in sequences. Enhancements were also made in vector-analysis system functions, assertion-control system tasks, newer assertion statements, and in the usage and restrictions of property and sequence local variables. There were also changes in the interpretation of some operators. The `checker`, as an encapsulation for SVA, was introduced in 2009 and many significant enhancements were made in the 2012 LRM including module-like programming features with some restrictions. This update includes details on all these new changes to the LRM as well as improvements to the organization and content of the previous release based on feedback received from our customers.

## The creators

This SVA 3rd Edition evolved from many years of practical experiences and studies in the processes / design / verification / and language worlds. This book is an excellent reference in the process and application of SVA. It was created by four authors who came from very strong technical backgrounds, thus putting a lot of synergy in the creation of this book. **Ben** has many years of design, synthesis, and verification of digital designs; he authored 11 books on VHDL, Verilog, design processes, VMM, PSL, and SVA, and has taught several classes in these fields. **Srini** worked at *Intel* as a verification engineer, and at *Synopsys* as an application and verification field engineer; he is now CTO of *CVC Pvt Ltd*, a high-end design-verification consulting company, and provides training in SV, SVA, VMM, OVM/UVM, VHDL, consulting for companies, and sales representation for many EDA products. **Ajeetha** has many years of experience in design and verification using VHDL, SV, SVA, VMM, OVM/UVM; she is the founder, CEO and Managing Director of *CVC*. She has also been consultant for many EDA companies and verification turnkey projects across India, Israel & Taiwan. **Lisa** worked at *Cadence* as a methodology and product engineer supporting assertions in simulation, formal verification, and emulation. She participated in the SVA standardization work for the IEEE 1800-2009 release. She also managed an organization that was responsible for the definition, verification, and support of Telecom IC's, LAN IC's, and ATM IC's at *Lucent Microelectronics*. She now is a technical marketing manager at *Real Intent*.

---

[1] This book is based on P1800/D6, 2012 DRAFT STANDARD FOR SYSTEMVERILOG, which reflects the latest version frozen to further technical changes.
DvCon2012 Proceedings, http://events.dvcon.org/events/proceedings.aspx?id=131-1-P
http://events.dvcon.org/events/proceedings.aspx?id=131-4

## How this book addresses SVA

This book is unique in the application and understanding of SVA for verification. This is because it addresses the assertion language from several viewpoints:

- The process of using assertions in the design of a chip. This includes using assertions throughout the requirements, design, and verification phases, as demonstrated by examples. The value of "process" is something that we deem critical, and was incorporated in Ben's *Component Design by Example* book.
- SystemVerilog as a language for engineers. The book presents many complete and simulatable examples that make use of best coding practices and advanced SystemVerilog constructs, including associative arrays, queues, and classes. It also addresses the use of SystemVerilog with VHDL. Our VHDL experiences and Ben's books on VHDL and Verilog made us more sensitive to the issues a VHDL user may encounter in using SystemVerilog.
- Assertions as a language, and the deep understanding of how assertions are processed. This includes the concepts of *attempts* / *threads* / *clock flow* / *end points* / *automatic variables* / *scheduling semantics* and their relationships on coding styles and efficiency and verification outputs.
- Assertions for real designs. What is reflected in this book's many examples and code writing methodologies and approaches is our vast work experience along with books on PSL, SVA and VMM, and presentation of several papers for DvCon and SNUG, and interactions with customers' requirements.
- Style and coding guidelines. Again, our experiences are reflected in our explanation of the constructs to use and to avoid, and why. The goal is to be able to write assertions that express the intended behavior, and to write them in a style that is efficient for simulation and formal verification.
- Verification and interaction with UVM. Our deep understanding of frameworks (wrote book *A Pragmatic Approach to VMM Adoption*) and usage of UVM, and its application in the verification environment made us more aware of the need to tie-in SVA with UVM. We explain the interactions of SVA with class-based UVM code in the area of error messaging of assertions and in the modification of variable values for use by the class-based control tasks.
- Debug of simulation results. The application of the assertion statements and simulation tool outputs, along with example debug capabilities is explained.
- Coverage. Coverage is a key element in the verification process. We demonstrate data oriented and control oriented coverage in the many complete examples addressing those topics.
- Formal verification. Our understanding and appreciation of formal verification is demonstrated through the explanation of the concepts and through the complete analysis of two design cases from requirements to formal verification.
- Dictionary of models. We felt a need to demonstrate how English requirements can be translated into SVA models. We achieved that by selecting a set of requirements based on documents, our work experience, and inquiries posted by users in several online technical forums.
- Dictionary of terms relating to SystemVerilog assertions. As authors of several books, we felt a need to explain the technical terms used in the field of design and verification and assertions. A dictionary of terms, with explanations and references is presented as an appendix.

## More about the creation of this book

Our goal is to make *SystemVerilog Assertions Handbook, 3rd Edition* an excellent reference manual on the application of SystemVerilog assertions during the design and verification processes. We explain the concepts, coding rules, and guidelines via text/tables/diagrams, images, annotations, complete models, and simulation results.

We validated the many complete examples and test verification code with five major EDA tools to insure accuracy and IEEE compliance with the currently supported features of SystemVerilog (*as of the date of this publication, the 2012 features are not yet supported (NYI))*. The simulation results included in the book are courtesy of *Mentor Graphics* who provided us with access to *QuestaSim* for the simulation of SVA code (*mentor.com*).

In addition, the models used in formal verification were verified with *OneSpin 360™ MV Product Family* of formal verification tools, and the graphical results are also provided on the distribution files (*onespin-solutions.com/*).

The construction of the many examples was greatly facilitated by the use of the *Design and Verification* Tools platform (*DVT, dvteclipse.com*), which is a powerful programming environment for the e language, SystemVerilog and VHDL with support for UVM, OVM, and VMM.

This book represents the collaboration of four authors; Ben was on the Assertions Committee (SV-AC) for the specification of the assertion features of SystemVerilog-2012. Other authors have been part of OVL, SVA 2009/Accellera committees.

## How to read this book

When a child learns a language, he/she first learns, by dense exposure to the words and through multiple passes, concepts, basic vocabulary, and overview before learning the alphabet and the grammar of the language. SystemVerilog is a language, and the assertions aspect is another outbreak of that language. In presenting the material for SVA, we took a similar approach to the learning process of a language. We started with an overview and exposure of the basic concepts, with many examples, without getting into the details of the grammar and rules. We then focused on the details of the sequences and properties, and then moved on to advanced topics with more examples. We followed that by addressing the process of using assertions in all phases of the design and verification cycles, including the requirements, design, and verification phases. We added the application of formal verification with two complete models. We then followed that with coding and usage guidelines, and then a dictionary of models and a dictionary of terms.

When addressing each of those topics, we decided to present applications and information that dealt with the topic at hand (e.g., local variables) but with certain advanced topics presented in later sections (e.g., `first_match` operator)). We clearly identified the language rules and guidelines addressing the individual topics during their contexts. We annotated the rules and code examples with comments and callout boxes. In reading this book, many users may find it easier to first take a look at the annotated code examples and grasp the concepts and style prior to digging deeper into the text that explains the rules. Once a good understanding of the language rules and guidelines is achieved, users may frequently refer to the tables that summarize (with examples) the syntax of the language (e.g., Section 2.1, 2.2, 2.3, 3.1, 3.9, 4.2.3, 4.2.4, 4.5).

Throughout the book we indicated the forward / backward referencing of critical topics. Thus, we envision the reading of this book as a multi-pass process, with appropriate jumps to forwarded material if the reader needs more information on that topic. We believe that this process will help the reader grasp the various concepts, applications, and grammar of the language.

The coding and usage guidelines presented throughout this book emerged from years of doing design and verification, and of using / teaching HDLs and assertion languages and framework libraries. We envision that in near future EDA tools will emerge to enforce these guidelines as sort of lint checks for SVA.

We also strongly recommend writing the exercises at the end of Chapter 3 and reading the answers to those exercises in Appendix A; those answers provide additional information and recommendations about the critical concepts.

## The intent

One of the reasons that we decided to write this handbook on SystemVerilog Assertions is the positive impact Assertion-based Verification (ABV) is providing in the design & verification of complex chips, and we believe that SystemVerilog is setting up a viable and effective standard in the design and verification processes. We also felt that the "assertions" aspect of SystemVerilog needed special emphasis. Thus, we maintain the focus of this book on SystemVerilog Assertions, with usage of many of the new features that SystemVerilog provides. We are assuming that the users are familiar with SystemVerilog, and have access to books that address SystemVerilog language.[2] Assertion-Based Verification is changing the traditional design process because that methodology helps to formally characterize the design intent and expected operations.[3] ABV also quickens the verification task because it provides feedback at the white-box level.[4] As a formal property specification language, SystemVerilog Assertions facilitate automation of common verification tasks that can be exploited across various verification technologies.

As designers and consultants/trainers, we experienced many designs that were weakly specified and documented. The RTL modeling lacked information about properties and design characteristics, and that led to difficulties and/or ambiguities in the maintenance and verification processes. A design specification is helpful in defining requirements. However, specifications are generally defined in an informal language, like English. They lack a standard machine executable representation and cannot be dynamically simulated and/or statically processed by a formal verification tool to ensure compliance to requirement. Adding SVA to the process fills that gap – at least partly for control dominated feature specifications.

## Book Organization

**Chapter 1** provides an introduction to Assertion-Based Verification and serves as an introduction to SystemVerilog Assertions (SVA) concepts with emphasis on properties and assertions, immediate and concurrent. It also addresses the topic of states of an assertion. It prepares the readers for Chapters 2, 3, and 4, which represent the "core" of SystemVerilog Assertions. **Chapter 2** delves into the understanding and application of sequences that represent the real base for the definition of temporal assertions. That chapter extends from chapter 1 the concepts of attempts / threads of assertions; the definition of the sequence operators; and the rules of local variables. **Chapter 3** delves into understanding properties, along with the property operators. **Chapter 4** provides a deeper appreciation of SystemVerilog Assertions by addressing advanced topics for properties and sequences, including assertion-based functions; clocked sequences and assertions across multiple-clock domains; the SystemVerilog scheduling mechanism used in assertions; the assertion directives; the immediate assertions; and binding of verification entities to modules. **Chapter 5** introduces the `checker`. That chapter includes the motivation behind this relatively new entity, the syntax, its contents, the use model, the rules, and its applications by examples. **Chapter 6** addresses the methodologies in using properties / sequences / assertions during the requirement and verification planning phases, in addition to the RTL and testbench levels. It first explains the process, and then demonstrates an application of assertions in the requirements specification and verification plan using a synchronous *First-In First-Out* (FIFO) as an Intellectual Property. SystemVerilog packages, interfaces, modules, and bindings are also demonstrated. **Chapter 7** addresses the formal verification aspects of SystemVerilog Assertions, and introduces the global clocking functions, typically used in formal verification. Chapter 7 focuses on Formal Verification (FV) methodologies for functional verification of RTL designs. It provides two case studies verified with *OneSpin 360™ MV Product Family* of formal

---

2 * *SystemVerilog Language Reference Manual* http://www.systemverilog.org/
* *SystemVerilog for Verification: A Guide to Learning the Testbench Language Feature*s, Chris Spear and Greg Tumbush (Feb 14, 2012)
* *SystemVerilog For Design A Guide to Using SystemVerilog for Hardware Design and Modeling* Stuart Sutherland, Simon Davidmann, Peter Flake, KAP, June 2003, ISBN 1-4020-7530-8

3 *Assertion-Based Design, Second Edition*, Harry D. Foster, Adam C. Krolnik, David J. Lacey June 2004, ISBN 1-4020-8027-1,
*The SystemVerilog Verification Methodology Manual* (VMM), 2005 Springeronline.com

4 *Writing Testbenches: Functional Verification of HDL Models,* Janick Bergeron, Kluwer Academic Publishers

verification tools using as testcases a traffic light controller model (an FSM type design) and the FIFO model (control model with a memory) described in Chapter 6. **Chapter 8** provides a set of guidelines in using SystemVerilog Assertions. These guidelines emerged from experience with usage of Assertion-Based Verification with Accellera's PSL, vendor's recommendations, code reviews, and LRM documentation. **Chapter 9** represents a "dictionary" of classes of application examples that translate English descriptions of properties to SystemVerilog properties**. Appendix A** provides the answers to the exercises asked at the end of Chapter 3. **Appendix B** is a summary of terms and definitions used within this book. **Appendix C** is a list of the system tasks and system functions**.** A list of **reserved words** is also provided. The **Index** provides a page lookup for information available in this book.

## All code is available for download

**http://www.systemverilog.us/BkSva3**
**http://cvcblr.com/BkSva3**
**Use WinZip for .zip files, and "tar xvfz file_name.tgz" for .tgz files**

# DISCLAIMER

Every attempt was made to ensure accuracy in the specifications and implementation of the languages (HDLs and SystemVerilog Assertions) and models. However, all code provided in this book and in the accompanied website is distributed with *ABSOLUTELY NO SUPPORT* and *NO WARRANTY* from the authors. Neither the authors nor any supporting vendors shall be liable for damage in connection with, or arising out of, the furnishing, performance or use of the models provided in the book and website.

# Acknowledgements

*SystemVerilog Assertions Handbook, 3nd Edition* could not have been written without the support and help from several companies who provided us with access to their design and verification tools that support SystemVerilog, along with access to their support groups who provided us with valuable information about SystemVerilog. We also acknowledge the insights of several engineers who helped us in the review process.

We thank Karen Pieper, *Accellera chair for IEEE P1800 Standard for SystemVerilog* for nominating Ben Cohen to represent Accellera in the SV-AC assertions group for the development of SVA'2012. His participation and involvement in this group helped in the specification of the language and allowed us to bring more insights into the best practices and applications of SVA.

We particularly thank *Mentor Graphics*® for providing us licenses of *QuestaSim* (a part of the *Questa*® verification platform) for the verification of assertions through simulation.[5] The ease of use of those tools, and the display of results with concise, but on target, information on the various views helped us in better explaining the behavior of assertions. Of particular interest was the waveform view that displayed the assertion signals, assertion successful attempts, vacuity, pass, and fail. The assertion thread viewer was also of great value as it provided more detailed information about an assertion attempt, its threads, and the values of its local and related variables. Other valuable outputs provided by the tool included the assertion / coverage/ cover / covergroup windows. We thank *Mentor Graphics*® for granting us permission to publish those results in our book and on the distribution files.

We would like to express our gratitude to *OneSpin Solutions* for providing us with formal verification analyses and results of two of our RTL models using *360™ MV*[6], OneSpin's formal assertion-based verification (ABV) solution for ASIC and FPGA designs. *OneSpin's 360™ MV* supports a broad range of formal ABV applications including automatic RTL checks, verification of implementation intent and high-level functional requirements, systematic operation- and transaction-level design verification, as well as automatic detection of verification gaps in assertion sets. The application of *360 MV* uncovered several subtle design and assertion issues in our RTL models that have been missed by previous verifications. The graphical root cause analysis features of *360 MV*™ were very helpful in understanding and correcting these issues. We also thank *OneSpin Solutions* for granting us permission to publish the results in both the book and the distribution files. We also thank Klaus Winkelmann for helping us in the use of formal verification to uncover the issues with our designs.

We thank Cristian Amitroaie and his support group from *AMIQ* for providing us licenses of DVT, an excellent set of Design and Verification tools. DVT provides a complete and easy to use programming environment for the e Language, SystemVerilog with support for SVA and the VMM/OVM/UVM frameworks, and VHDL[7]. The use of DVT allows us to easily code and verify the examples prior to compilation, and copy the formatted code into the book.

---

[5] *Mentor Graphics*® provides software and hardware design solutions that enable companies to develop better electronic products faster and more cost-effectively. They offer numerous products in the area of chip design and verification. In the area of simulation and assertions, *Mentor Graphics* provides *ModelSim DE* and *QuestaSim* simulators. http://www.mentor.com

[6] *OneSpin*'s *360 MV* product family is a comprehensive formal assertion-based verification solution for starters, experienced users and experts. *360 MV* is based on more than a decade of industrial application experience and technology development in formal verification. http://www.onespin-solutions.com/

[7] http://www.amiq.ro/consulting/, http://www.dvteclipse.com/index.html Design and Verification Tools (DVT) The Complete Development Environment for the e Language, SystemVerilog, and VHDL.

In the creation of the 2nd Edition of this book, we received support from several companies, and that information is retained in this edition. Our sincere thanks are due to *Synopsys* for providing us, at that time, access to their *VCS* platform supporting many of the SystemVerilog IEEE 1800-2009 features.[8] In addition, *SpringSoft* supported us by providing a license of the *Verdi™ Automated Debug System*, an advanced solution for debugging digital designs and assertions.[9] *Aldec* was another company who provided an engineering resource for technical review and access to their *Riviera-PRO™* a high-performance verification platform for ASIC and FPGA designs with ABV support.[10]

We thank the *IEEE for* granting us permission to quote material from the *IEEE 1800 LRM*, the document that defines the rules of SystemVerilog and SystemVerilog Assertions.

Several SystemVerilog experts participated in the review process of this book. The review is a necessary step to iron out areas of disagreements, and to provide a piece of work that meets user's requirements in the application of SystemVerilog Assertions. In that endeavor, we sincerely thank the following people and organizations: Dennis Brophy, from *Mentor Graphics* for his full support of our endeavor; Michael Siegel and Klaus Winkelmann, from *OneSpin Solutions* for their help and support in verifying two models through *OneSpin 360™ MV Product Family*, and for valuable feedback on formal verification.

We also thank the following engineers for reviewing our book and providing valuable feedback: Anupam Prabhakar, *Mentor Graphics®*; Sven Beyer, *OneSpin Solutions*.

During the creation of this book there were several language issues and clarifications that needed to be addressed in the IEEE 1800 SVA committee. Several participants of this IEEE committee contributed to our specific questions on some issues; thus we particularly thank Erik Seligman, Dmitry Korchemny, and Ed Cerny.

I (Ben) especially thank my wife, Gloria Jean, for supporting me in this endeavor.

We (Ajeetha & Srini) would like to acknowledge the valuable time our cute little son Adruth and elder son Anirudh have allowed us to spare on this book. I (Srini) would like to personally dedicate this book to my beloved father Sri. K. Venkataramanan who passed away recently; his memories and blessings are my sole inspiration to cross any hurdle in my life.

---

8 The *VCS* solution powerful debug and visualization environment minimizes the turnaround time to find and fix design bugs. http://www.synopsys.com/tools/verification/functionalverification/pages/vcs.aspx

9 The *Verdi* Automated Debug System is an advanced solution for debugging digital designs that provides powerful technology to comprehend complex and unfamiliar design behavior; automate difficult and tedious debug processes; and unify diverse and complicated design environments.
http://www.springsoft.com/products/debug-automation/verdi

10 *Riviera-PRO* is a high-performance verification platform for ASIC and FPGA design teams, equipped with mixed-language simulation engine and advanced debugging tools. Riviera-PRO supports Electronic System Level (ESL) Verification with SystemC and SystemVerilog, Assertions Based Verification (ABV), Transaction Level Modeling (TLM) and VHDL/Verilog Design Rule Checking.
http://www.aldec.com/Products/default.aspx


**Sculpture Created by my Wife Gloria to**
**Express my Long Hours with a Laptop in the Creation of Books**

# About the Authors

**Ben Cohen** is currently a consultant and actively represents Accellera in the IEEE 1800-2012. He has technical experience in digital and analog hardware design, computer architecture, ASIC design, synthesis, and use of hardware description languages for modeling of statistical simulations, instruction set descriptions, and hardware models. He applied VHDL since 1990 to model various bus functional models of computer interfaces. He authored several books in the field of design and verification languages including *VHDL Coding Styles and Methodologies,* first and second edition; *VHDL Answers to Frequently Asked Questions*, first and second editions; *Component Design by Example; Real Chip Design and Verification Using Verilog and VHDL*; *Using PSL/SUGAR with Verilog and VHDL* (first edition, also translated to Japanese)*; Using PSL/Sugar for Formal and Dynamic Verification, 2nd Edition; SystemVerilog Assertions Handbook* (first edition, also translated into Japanese); *SystemVerilog Assertions Handbook* (2nd edition); and *A Pragmatic Approach to VMM Adoption.*

He was one of the pilot team members of the VHDL Synthesis Interoperability Working Group of the Design Automation Standards Committee who authored the *IEEE P1076.6 Standard for VHDL Register Transfer Level Synthesis*. He was a member of the *VHDL* and *Verilog Synthesis Interoperability Working Group of the Design Automation Standards Committees*, and *Accellera OVL and PSL* standardization working groups. He participated in the working group for the development of the new IEEE 1800-2009 LRM for SystemVerilog assertions. He also was a member of the SV-AC assertions group for the development of *IEEE P1800-2012 Standard for SystemVerilog*. He taught several VHDL, PSL, and SVA training classes. He has presented many papers at events such as DVCon and SNUG, including an assertion tutorial on SVA and PSL and VMM.

**VhdlCohen Publishing**
**ben@SystemVerilog.us http://SystemVerilog.us**

**Srinivasan Venkataramanan** Srinivasan Venkataramanan is Chief Technology Officer (CTO) at CVC Pvt Ltd, a high-end Design-Verification consulting firm based in Bangalore - India.

Srinivasan's areas of interest are the advanced verification solutions and methodologies such as SystemVerilog, UVM, OVM, VMM, Assertion-Based Verification, formal verification etc. As part of CVC, he provides support to leading edge semiconductor design companies on their verification methodologies and challenges. CVC has launched *Unleashing UVM* (TM) solution during mid 2012. Under this, he offers solutions such as our various training sessions, solve complex customer problems, such as *time-to-debug*, *qualifying verification effectiveness*, *choosing the right technology* for a given problem etc.

In his previous employment at Synopsys, India Private Ltd., Bangalore, he was a Senior Staff Verification Solutions Engineer where he deployed advanced Verification solutions to many customers across AsiaPac region including Taiwan, China, India and also Israel. He assisted customers in variety of areas, such as evaluating SystemVerilog; optimizing regressions using multi-core technologies; and showcasing value of VCS verification platform to specific domains, such as Image processing, Networking, DSP etc. Prior to joining Synopsys, he worked at Intel, Philips Semiconductors, and RealChip communications in the areas of front-end design and verification of ASICs (leading edge high-speed, multi-million gates ASIC designs) with several HDLs and HVLs, including VHDL, Verilog, Specman, and Vera. He successfully developed complex verification environments using advanced methodologies, such as Coverage-Driven Verification and Constrained-random verification using Verisity's Specman, ABV etc. Srini holds a Masters Degree from the prestigious Indian Institute of Technology (IIT), Delhi in VLSI Design, and Bachelors degree in Electrical engineering from TCE, Madurai. Srini has co-authored the following books: *A Pragmatic Approach to VMM Adoption*; Using *PSL/Sugar, 2nd Edition*; and *SystemVerilog Assertions Handbook 1st and 2nd Editions*.

He presented several papers at conferences and forums such as DesignCon, DVCon, SNUG etc. He has been delivering training sessions on SVA, SVTB, OVM & VMM to customers for more than 5 years.

**CVC Pvt.Ltd.,**
**Bangalore, India**
**http://www.cvcblr.com/ srini@cvcblr.com**

**Ajeetha Kumari** Ajeetha Kumari is the founder and CEO and Managing Director of CVC Pvt Ltd, a high-end Design-Verification consulting firm based in Bangalore - India. At CVC she leads a team of elite, seasoned Verification professionals focused on next generation verification automation and productivity techniques. As CEO, her focus is on business development, new strategic partnerships and exploring new ventures for CVC. More recently she was instrumental in rolling out CVC's various functional verification offerings under a new logo *UnleashingUVM (TM)*. She has been providing consultancy to leading-edge semiconductor houses on various verification challenges for over half-a-decade.

Ajeetha is very well networked and known for close interaction with Design-Verification community on various online forums and events. She runs a popular blog at www.cvcblr.com/blog along with contributions from many others. She presented many papers, tutorials at events such as DVCon, SNUG, CDNLive etc. She has experience with several HDLs and HVLs including Verilog, VHDL, SystemVerilog, PSL, SystemVerilog Assertions, E and Vera. She co-authored the following books: *A Pragmatic Approach to VMM Adoption*; *Using PSL/Sugar, 2nd Edition*; and *SystemVerilog Assertions Handbook 1st and 2nd Editions.*

She received her M.S. in Electrical engineering from the prestigious Indian Institute of Technology (IIT), Madras.

**CEO & Managing Director**
**http://www.cvcblr.com/ akumari@cvcblr.com**

**Lisa Piper** currently works for Real Intent Inc as a senior technical marketing manager for advanced verification products. Her primary focus is applying structural analysis, simulation, and formal techniques as appropriate, to tackle issues caused by X-propagation. Lisa worked for Cadence Design Systems for 10 years where she was involved with using assertions in simulation-based verification, adapting OVL assertions for use in acceleration, and formal verification (a.k.a. model checking). Product definition, training, assertion methodology, and new product introduction were key areas of focus. This also included active participation in IEEE 1800-2009 SVA standardization work.

Prior to that, Lisa spent 10 years managing the definition and applications support teams for Telecom IC's, LAN IC's, and ATM IC's at Lucent Microelectronics. This built upon previous experience at AT&T Bell Labs co-developing the first ISDN S/T Interface chip and designing one of the first ISDN U-Interface phones.

Lisa holds an MSEE from Ohio State University and a BSEE from Purdue University. She co-authored the book *SystemVerilogAssertions Handbook, 2nd Edition.* Lisa presented many papers at events such as DVCon, including an assertion tutorial on SVA and PSL.

**lisa_piper@systemverilog.us**