SystemVerilog Assertions Handbook, 2nd 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, 2nd 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, 2nd Edition
… for Dynamic and Formal Verification
ISBN 878-0-9705394-8-7

Printed on acid-free paper

Printed in the United States of America

Contents

FOREWORD, Dennis Brophy

Debug of electronic systems is an ever increasing challenge given the relentless increase in design complexity. Many design issues are buried deep in a system and can difficult to reach or detect in a timely fashion even with today's automated stimulus generation solutions.

Assertion-based verification technology has been found to address these challenges. Design and verification engineers can place assertions in designs or bind assertions to designs to monitor, report and take action when incorrect design behavior is detected. Assertions are the basic elements that enable formal verification where design properties are examined to determine design correctness and facilitate the creation of counter examples to demonstrate design failures.

Recent research has shown more than two-thirds of IC and ASIC designers are using SystemVerilog assertions today with three-fourths to be using them during 2010. FPGA design verification is also facilitated by the use of assertions given the advent of FPGAs with embedded processor cores along with advanced protocol support that would generally require designers to wait until the FPGA was placed in a system to fully debug the system.

The second edition to the *SystemVerilog Assertions Handbook* comes at a time when the IEEE updates its popular SystemVerilog standard (IEEE Std. 1800™-2009) and an FPGA community that is increasing its adoption of SystemVerilog assertions. Design and verification engineers will find the handbook as an excellent resource to begin to adopt assertions, and to apply the latest additions and updates found in the IEEE standard to 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, Shankar Hemmady

When I was asked to review the book, my first thought was, “Don’t we have enough books on SVA already?” However, having worked with Ben and Srini in the past, I was curious to learn more. As I browsed through the book, it became evident that this was a masterpiece in the works. This is the first book to take a design verification engineer’s and manager’s perspective in the assertions arena. It puts the debate on static formal analysis versus dynamic simulation-based assertion checks to rest by providing guidelines on how and where to use these technologies appropriately. The book also includes a lexicon of commonly used temporal requirements/properties in plain English. This is valuable reference material for engineers working in the trenches.

Shankar Hemmady
Principal Engineer,
Synopsys Inc,
Mountain View, California

The new IEEE SystemVerilog Standard has significantly expanded the support for assertions in the language. Some of these extensions add capabilities for sophisticated users, while others make the language easier to use even for beginners. The *System Verilog Assertions Handbook, 2nd Edition*, tackles the critical task of documenting the new (and original) language features with clear explanations and numerous usage examples. I’m certain that the author’s efforts will accelerate the adoption of these valuable new capabilities. I can certainly recommend this book to anyone who wants to get up to speed on the latest System Verilog assertion capabilities for simulation and formal verification.

Dan Benua
Principal CAE
Synopsys Verification Group
Mountain View, California

FOREWORD, Michael Siegel

Assertion-based verification (ABV) and SystemVerilog assertions (SVA) have enjoyed significant adoption in verification projects, both large and small. Assertions, in general, and SystemVerilog assertions in particular, are an effective means to analyze and verify design behavior. They are employed to share design intent, to check expected designs behavior, to enhance error observability, to speed integration verification and to define coverage models, to name but a few applications. Moreover, assertions can be used in simulation and formal verification, enabling engineers to leverage the complementary strengths of these technologies for block, subsystem, and chip-level verification, in order to reduce overall verification effort and to achieve earlier verification closure.

This book, *SystemVerilog Assertions Handbook, 2nd Edition* by Ben Cohen, Srinivasan Venkataramanan , Ajeetha Kumari and Lisa Piper, is a timely, practical and comprehensive reference manual on the use of SystemVerilog assertions, addressing both the existing community of SVA users and people who want to get started with SVA-based ABV, in both (System)Verilog and VHDL designs.

The book – one of the first to cover the new SystemVerilog standard, IEEE 1800-2009 – establishes a conceptual foundation for how to use SVA for assertion-based verification and serves as a practical users guide. It details a broad range of assertion use from the capture of low-level design intent to specification-level requirements, comprehensively covering language features, style guidelines and methodology, taking a step-by-step approach illustrated by many real-world design and assertion examples. Its examples also illustrate the power of using the complementary strengths of dynamic and formal ABV. Experienced SVA users will appreciate the clear presentation and good overview of the new SVA features in the IEEE 1800-2009 standard.

Michael Siegel
Product Marketing Director
OneSpin Solutions
http://onespin-solutions.com/

FOREWORD, Scott Sandler

Like many transformative changes in electronic design, Assertion-Based Verification has taken a long time to mature and become established in the mainstream methodology. And it has followed the typical adoption pattern: early use by very large companies with their own in-house tools and proprietary languages; subsequent emergence of commercial tools, still using proprietary languages; standardization of a language; maturation of tools and techniques; and finally broad acceptance. There is no longer any question that Assertion-Based Verification is a necessary part of the design flow for complex integrated circuits.

Nor is there any uncertainty about the language that will be most widely used to express assertions; it will be SystemVerilog Assertions. This update to the *SystemVerilog Assertions Handbook* is of course very timely. With the recent release of IEEE 1800-2009, SystemVerilog assertions have been greatly enhanced for both power and usability. The new “checker” entity in particular, which gets authoritative treatment in the book, makes thorough verification easier to accomplish and makes testbenches easier to understand.

The approach that Ben Cohen’s team uses to prepare and present highly technical information renders it easily approachable, accessible, and useful. Readers will appreciate the clarity and completeness of the text and examples, and will find they can rely on *the SystemVerilog Assertions Handbook* as a constant companion as they adopt and refine assertion-based verification using SVA.

Scott Sandler
Vice President, Corporate Marketing, SpringSoft, Inc.
President, SpringSoft USA
http://www.springsoft.com/

SpringSoft
accelerating engineers

FOREWORD, Tapan Kapoor

SVA is part of SystemVerilog standard (IEEE 1800). With the latest version of SystemVerilog LRM (IEEE 1800-2009), several new constructs and enhancements have been added to SVA, such as the **checker** construct, new sampled value functions, enhanced local variable support, global clocking, contextual clock inferring, new property operators (**iff**/ **implies**/ followed-by/ **nexttime**/ **always**/ **until**/ **eventually**) and so on.

This book is an excellent enabler for beginners and a detailed guide for advance SVA users. The book explains the syntax and semantics of all the existing and new constructs in SVA, identified with a sidebar, and supplemented with examples, appropriate diagrams, and waveform charts. The book also explains the guidelines for writing assertions that facilitate efficient and effective usage of ABV. This book also addresses, by example, various components of verification—Coverage, Verification Methodology, Verification Planning, and Formal verification in the context of SVA assertions.

At Cadence, we are committed to support SVA in all our tools including simulation, formal verification, and hardware-assisted acceleration. Cadence supports SVA through the generation of native code that is tightly integrated into our *Incisive*™ assertion-based verification environment for simulation and formal verification, which is based on simulation and model checking, respectively. We proactively created all the tools and flows required to support the advanced verification components to work well in different verification environments.

Tapan Kapoor Incisive ABV - R&D Cadence Design Systems, Inc. http://www.cadence.com/	cādence®

FOREWORD, Daniel Mlynek

In my opinion, the book *SystemVerilog Assertions Handbook, 2nd Edition* by Ben Cohen, Srinivasan Venkataramanan, Ajeetha Kumari and Lisa Piper is a perfect source of ABV knowledge. The book will guide the reader step by step through the ABV methodology. Starting from basic knowledge for beginners, it will unveil the power of the SVA. The book describes all language constructs - one by one - with examples and real design applications for learned constructs. It points out the language pitfalls, and explains the grey areas in the language so that you do not spend hours trying to understand why your assertion code doesn't work as expected.

Along with providing a great level of detailed information on the assertion language itself, this book also provides a wider look on the whole verification process and the role of ABV in this process, including from the managers point of view. The book also shows what features can be expected from the tools, and how tools can help us analyze the results. At the end we are given a set of guidelines for the methodology and the language that help avoid confusion. The *2nd edition* of the book has another advantage, which is significant; it is written on the basis of the new SystemVerilog standard - IEEE1800-2009 by people who were involved in the standard development. SystemVerilog is a living language and the new standard introduces new, complex, and powerful features, which can be found in the book.

In conclusion, I'm certain that the readers will not put this book on the shelf after reading, but will be referring to it frequently as a reference manual.

Daniel Mlynek
Leader Application Engineer
http://www.aldec.com/

FOREWORD, Don Mills

SystemVerilog Assertions Handbook, 1st Edition was a primary source for *Sutherland HDL*'s training courses on SystemVerilog Assertions. Students referred to the book often during training classes, for details and reference. It is anticipated that this 2nd Edition will provide yet further enlightenment on the topic of Assertions, for individuals and training courses, for many years to come.

*SystemVerilog Assertions Handbook, 2nd Edi*tion is an excellent reference for learning the basics of the assertion language. Syntax summaries alongside examples help in learning the syntax. There are many examples with graphical representations that demonstrate the concepts. Basic rules are listed, often with quotes from the standard, and then explained. The book goes beyond the standard to demonstrate many subtleties that produce unexpected results and poor performance, and flags the pitfalls to avoid. It is a great refresher for experienced users and for those looking to understand what is new in the SVA language for the IEEE 1800-2009 release. Additional chapters present methodology and application perspectives.

Don Mills
LCDM ENGINEERING.
Consultant, trainer for SystemVerilog and SVA courses.
http://www.lcdm-eng.com
http://www.sutherland-hdl.com

Quote, Erik Seligman

Ben's probing questions have significantly helped those of us on the committee to better understand some implications of the many new language features we added to the IEEE 1800-2009 standard. As a result, I am eagerly looking forward to the arrival of this new book, which I will be using as a key reference on SystemVerilog assertion issues.

Erik Seligman,
Chair of the IEEE p1800 Special Subcommittee on Checkers

PREFACE

The Book
SystemVerilog Assertions Handbook, 2nd Edition is a follow-up book to the first edition, published in 2005. This version addresses the new SystemVerilog assertion features, enhancements, and clarifications presented by the IEEE 1800-2009 Standard for SystemVerilog Unified Hardware Design, Specification, and Verification Language (herein referred as IEEE 1800-2009, or LRM – language reference manual).[1] These new changes in the area of assertions include several new operators for properties and sequences; newer assertion statements; defaults disables; usage and restrictions of property and sequence local variables; changes in the interpretation of some operators; and the definition of a new type of entity called **checker.** The **checker** supports the grouping of several assertion directives and related supporting code, and the inlined instantiation of this grouping in the design. Another significant enhancement to IEEE 1800-2009 was the redefinition of assertions in procedural code. The previous release of the standard was also enhanced to protect against races causing false firings of immediate assertions in procedural code.

Our goal is to make *SystemVerilog Assertions Handbook, 2nd Edition* an excellent reference manual on the use of SystemVerilog assertions. We explain the concepts via text/tables/diagrams, images, annotations, and simulation results. We present, by examples, the coding rules with many simulatable models. We also provide guidelines and recommendations in the use of SVA in the design and verification process. We address formal verification and use two complete examples to demonstrate the value of formal verification. We provide a dictionary of modeling requirements that are translated into assertions, and a dictionary of common terms used in assertions. All new IEEE 1800-2009 features are identified with a bold bar on the left margin.

Many examples include complete test verification code, along with simple testbenches to demonstrate the concepts and show the simulation results. These models, along with the captured waveforms and thread viewer and assertion statistics are available in the distribution files. The simulation results are courtesy of *Mentor Graphics* who provided us with access to *QuestaSim* and *ModelSim DE* for the simulation of SVA code. We also used the ouputs of *Verdi Automated Debug System* to further demonstrate the key points on assertions. 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, courtesy of *OneSpin Solutions*.
This book represents the collaboration of four authors who are experts in SystemVerilog linguistics, system engineering, architecture, and design and verification with hardware description languages (HDLs) and hardware verification languages (HVLs), along with

[1] This book is based on P1800/D9, August 6, 2009 DRAFT STANDARD FOR SYSTEMVERILOG, which reflects the latest version frozen to further technical changes.

experience in teaching and in authoring several books on assertion and verification, thus bringing more synergism to this *SystemVerilog Assertions Handbook, 2nd Edition*.

How to read this book

When a child learns a language, he 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 properties and sequences, and then moved on to advanced topics with more examples. 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** operators). 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 is more interested in that topic. We believe that this process will help the reader grasp the various concepts, applications, and grammar of the language.

Throughout the book we used a coding style notation explained in Chapter 8 on guidelines. Those guidelines emerged from years of doing design and verification, and of using / teaching HDLs and assertion languages. We strongly recommend that the guidelines presented in this chapter be considered. 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 that Assertion-based Verification (ABV) is providing, 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 methodologies.

[2] * *SystemVerilog Language Reference Manual* http://www.systemverilog.org/
* *SystemVerilog For Verification*, Tom Fitzpatrick, Dave Rich, Aturo Salz and Stuart Sutherland, 2005, Springer Springeronline.com
* *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

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 requirements.

Assertion-Based Verification with SystemVerilog Assertions
SVA gives the design architects a standard means of specifying design properties using a concise syntax with clearly defined formal semantics. Similarly, it enables the RTL designers to capture design intent and assumptions in a verifiable form, while enabling the verification engineers to validate that the implementation satisfies its specification through dynamic (i.e., simulation) and formal verification options. Furthermore, it provides a means to measure the quality of the verification process through the creation of functional coverage models built on formally specified properties. It provides a standard means for hardware designers and verification engineers to rigorously document the design specifications using a machine-executable format.

SystemVerilog with assertions improves the quality of digital designs and helps eliminate defects per the Six Sigma methodology[5] because assertions play an important role in a unified verification methodology ranging from requirement definitions through design and verification (see Chapter 6 for discussion on the design process with SystemVerilog Assertions). Assertions express functional design intent and can be used to express assumed input behavior, expected output behavior, or forbidden behavior. Assertions allow the architects or designers to capture the design intent and assumptions in a manner that can be verified in the implementation. Assertions are captured during the development process and are continuously verified throughout the design and verification process. Working in a unified verification methodology, assertions reduce the verification time by detecting bugs earlier, and by isolating where a bug is located (by being closer to the source of error). In addition to detection of property violations, assertions improve the efficiency in a unified methodology by improving reuse, enhancing testbench checking, and capturing coverage information. Per Lionel Benning's experience, designers created fewer initial bugs in the RTL as an ABV methodology forced them to think more clearly and accurately about what to design.[6] Also, properties are more accurate and less prone to misinterpretation than comments in the RTL.

Our experience with the usage of SystemVerilog Assertions for front-end design definitions demonstrated that SystemVerilog Assertions are very powerful in the process of delving into design requirements, design architecture, and definition of restrictions imposed by the architecture. We found the property and assertion definitions more expressive and precise than the use of a natural language, e.g., English. The RTL design and verification tasks were greatly simplified as a result of using this assertion-based methodology because it alleviated the need to write a thorough testbench reference model prior to debugging the model. During simulation the assertions immediately alerted us of design and testbench errors. The use of formal verification tools helped us greatly at quickly detecting errors in the design, along with counterexamples that

[5]http://www.isixsigma.com/sixsigma/six_sigma.asp
Six Sigma is a disciplined, data-driven approach and methodology for eliminating defects (driving towards six standard deviations between the mean and the nearest specification limit) in any process -- from manufacturing to transactional, and from product to service.

[6] *Verifiable RTL Design: A Functional Coding Style Supporting Verification Processes in Verilog*, Lionel Benning and Harry Foster, Kluwer Academic Publishers

demonstrated the problems without any testbench. Corrections of these errors were quickly verified with another run of the formal verification tool.

We strongly recommend the use of ABV with SystemVerilog on design projects. ABV is a very viable methodology for the definition and verification of designs. We must admit though that at times assertions are very frustrating because they (correctly) insisted that our designs were in error while we believed that we had all the necessary fixes!!!

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. It prepares the readers for Chapters 2, 3, and 4, which represent the "core" of SystemVerilog Assertions. **Chapter 2** delves into understanding properties, along with the property operators. **Chapter 3** delves into the understanding and application of sequences that represent the real potential of SystemVerilog Assertions. That chapter addresses the concepts of attempts / threads of assertions; the definition of the sequence operators; and the rules of local variables. **Chapter 4** provides a deeper appreciation of SystemVerilog Assertions by addressing advanced topics for properties and sequences, including assertion-based functions; clocked sequences and multiclocking; the SystemVerilog scheduling mechanism used in assertions; the assertion directives; the immediate assertions; and binding of verification entities to modules. **Chapter 5** introduces the new type of entity, the **checker**. That chapter includes the motivation behind this 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 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 summary 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. 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

DISCLAIMER

Acknowledgements

SystemVerilog Assertions Handbook, 2nd 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, and without the insights of several engineers who helped us in the review process.

We particularly thank *Mentor Graphics®* for providing us licenses of *ModelSim DE* and *QuestaSim* (a part of the *Questa®* verification platform) both used for the verification of assertions through simulation. [7] 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.

Our sincere thanks are due to *Synopsys* for providing us access to their *VCS* platform supporting many of the SystemVerilog IEEE 1800-2009 features.[8] As Synopsys has been in the front line of defining the new standard, the *VCS* simulator started supporting many of the LTL features much before they became part of the IEEE standard. Early access to *VCS* meant that we could validate most of our code, and at times clarify our understanding of some of the corner case behaviors of these new features. This has resulted in several new Mantises being filed for clarification with the SystemVerilog committee during our book writing journey. The recent simulator version, along with support for the new LTL constructs and debug enhancements, is bringing most of the new IEEE 1800-2009 SVA features to be production ready.

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*[9], 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. Surprisingly, 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

[7] *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 Graghics* provides *ModelSim DE* and *QuestaSim* simulators. http://www.mentor.com

[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] *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/

the results in both the book and the distribution files. We also thank Klaus Winkelmann for helping us in the formal verification of our models, and in uncovering the issues with our designs.

We thank *SpringSoft* for supporting us in this endeavor by providing us a license of the *Verdi™ Automated Debug System*, an advanced solution for debugging digital designs and assertions.[10] *Verdi's* automated behavior tracing and *Behavior Analysis* technology, along with its intuitive visualization of multiple threads, temporal annotation on source code, and the timing annotation on temporal ranges helped us in demonstrating the inner works of complex assertions.

We thank *Aldec* for providing us engineering resource for technical review and access to their *Riviera-PRO*™ a high-performance verification platform for ASIC and FPGA designs with ABV support.[11]

We thank the *IEEE for* granting us permission to reproduce some 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 use 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: Dan Benua, *Synopsys*; Daniel Mlynek, *Aldec*; Tapan Kapoor, *Cadence Design Systems*; Roy Subir, *Texas Instruments*.

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

We also thank *Forte Design Systems*[12] for granting us a license of *TimingViewer* as a tool to draw timing diagrams for use in this book.

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 little son Anirudh has 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.

[10] 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

[11] *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

[12] *TimingDesigner* is a flexible, interactive timing analysis and diagram tool.
http://www.timingdesigner.com/ http://www.forteds.com/

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

About the Authors

Ben Cohen is currently a consultant. 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); 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 taught several VHDL, PSL, and SVA training classes.
VhdlCohen Publishing
ben@SystemVerilog.us http://SystemVerilog.us

Srinivasan Venkataramanan is Chief Technical Officer (CTO) at CVC Pvt Ltd, a high-end Design-Verification consulting firm based in Bangalore - India. CVC has been the pioneer in high-end verification consulting and Corporate trainings on many advanced VLSI topics. Starting in 2009, CVC productized an internal, time-tested incubation process of bringing up fresh engineers to be VLSI experts.

Srinivasan's areas of interest are the advanced verification solutions and methodologies such as SystemVerilog, 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 differentiates itself against other consulting firms by solving 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 specifcic 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*.

He presented several papers at conferences and forums such as DesignCon, DVCon, SNUG etc. He has been delivering trainings 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 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. 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 presented many papers, tutorials at events like 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*.

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 is currently an independent consultant for front end verification. 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 aspects of the job. 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 of 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 ISDN phones. Lisa holds an MSEE from Ohio State University.

lisa_piper@systemverilog.us