A Pragmatic Approach to VMM Adoption

… a SystemVerilog Framework for Testbenches

A Pragmatic Approach to VMM Adoption

… a SystemVerilog Framework for Testbenches

Ben Cohen
Srinivasan Venkataramanan
Ajeetha Kumari

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

A Pragmatic Approach to VMM Adoption

... a SystemVerilog Framework for Testbenches

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

A Pragmatic Approach to VMM Adoption
a SystemVerilog Framework for Testbenches

ISBN 0-9705394-9-5

Printed on acid-free paper

Printed in the United States of America

Contents

All code is available for download

FOREWORD, Janick Bergeron

Functional verification has become the most effort-consuming task in bringing a semiconductor product to market. And it is getting worse. The verification aspect of SystemVerilog was designed to bring the power and productivity of Hardware Verification Languages to the industry at large, within the familiar framework of the Verilog language. However, based on my many years of experience using and teaching HVLs, I knew that it would be a slow and difficult process for new adopters to realize all of its potential and benefits unless a clear path was provided. With three other experts in other area of functional verification, we set out to provide that path as the Verification Methodology Manual for SystemVerilog.

Then there is the question of the form the path should take. Should it be a path that takes users from simple, purely directed Verilog testbenches to constrained-random testbenches, exploring along the way how various SystemVerilog constructs are used most effectively? Or should it be a clear definition of the end goal, a set of minimum practices that must be used to create interoperable and reusable verification assets? Should it be a training vehicle or a methodology?

As the name of the VMM suggests, we chose the latter form. A methodology is about what should and should not be done to be as productive as possible. There must be a clear indication of whether or not a testbench or a transactor is compliant to the methodology. A methodology cannot be a training vehicle – nor can a training vehicle claim to be a methodology. A methodology and the training it requires to learn it and apply it effectively are two different things.

If you are not familiar with SystemVerilog – or a Hardware Verification Language – the VMM can be a large pill to swallow. We fully expected that training and introductory material would appear to help adopt the methodology described in the VMM – just like there are several introductory books and training classes on the Verilog language and its usage, despite the fact that it is fully specified in an IEEE standard document. VMM-related conference papers have also already started to appear.

I am pleased to welcome this book to the cannon of VMM literature. I hope you will find it helpful in appreciating the power of the VMM methodology and ease your adoption of it.

Janick Bergeron
Synopsys Scientist
http://www.synopsys.com/

FOREWORD, Stuart Sutherland

In my garage, I have a roll-around mechanic’s tool chest. It has several drawers, and each drawer has many tools neatly laid out side-by-side. I also have a repair manual on how to work on the 35 year old 1971 Volkswagen beetle that is parked in my garage; a project car that my son and I are working on. Unfortunately—especially for the car—having the tools and manuals is not enough to make me a qualified auto mechanic. The shop manual tells me what to do, but not which tool to use. Even if I happen to select the correct tool for the repair at hand, I find myself reading and rereading the same paragraph in the manual in a vain attempt to make the description and pictures in the manual match what I am looking at when lying underneath the car looking up at the greasy underside of the engine or transmission. And having all those tools and a manual do not seem to help a bit when I’m trying to figure out in what order to reassemble the 40 or more pieces that make up the front brake assembly and wheel bearings.

SystemVerilog gives us, as design and/or verification engineers, a huge set of tools to work with. SystemVerilog adds to the programming and verification constructs of Verilog many new data types, new programming statements, dynamic arrays, associative arrays, process synchronization, mailboxes, direct calls to C, and Object-Oriented programming (an entire tool set, in an of itself). The constructs and capabilities of SystemVerilog, if they were mechanical tools arranged neatly in a tool chest, would be the envy of mechanics everywhere.

The *Verification Methodology Manual* (VMM) is, as its name states, a manual, not unlike the repair manual on my old car. The VMM is not a simple manual that a novice engineer can just pick up and make sense of. It shouldn't be. The VMM presents a complex and robust methodology that can handle any size of design and any type of design, and at the same time, a methodology that can scale and be re-used for the verification of many future designs.

SystemVerilog gives us the tools, and the VMM gives us the methodology. But, like the tools and repair manual in my garage, SystemVerilog and the VMM are merely resources. Having the SystemVerilog tool set and the VMM do not magically make us verification engineers. Somewhere along the way, we need to figure out how to pick the right tool or tools from the SystemVerilog tool chest, and use those tools in the way the VMM recommends.

This is where this book, *A Pragmatic Approach to Adopting VMM*, comes into play. This book teaches by example how the constructs in SystemVerilog are used to implement the methodology presented in the VMM. Similar to learning from an experienced mechanic how to use the right tools to repair a car, this book provides a way to learn from experts the right way to use SystemVerilog for verification. With the help of these experts, it is much easier to understand how to apply the methodology presented in the VMM on actual designs.

This book supplements the SystemVerilog language standard and the VMM. This book does not replace having a copy of the Verilog and SystemVerilog standards (or good books about those standards). Nor does this book replace having a copy of the VMM. The language reference books, the VMM and this book should be used together.

Ben Cohen, Srinivasan Venkataramanan, and Ajeetha Kumari have found a way to make it obvious how to correctly use the complex VMM and the vast number of SystemVerilog Object-Oriented verification constructs. The book is an invaluable resource to all engineers who need to apply SystemVerilog in the verification of designs.

Next, I wonder if Ben and his co-authors can write a book to help make it obvious how to make the text and pictures in my auto repair manual map to the greasy underside of my 1971 VW beetle! ☺

Stuart Sutherland, SystemVerilog instructor and President of Sutherland HDL, Inc.
(Sutherland HDL provides expert level training on how to correctly use Verilog and SystemVerilog, with specialized courses on synthesis, verification, and assertions)

SUTHERLAND HDL
training engineers to be Verilog and SystemVerilog wizards
www.sutherland-hdl.com

FOREWORD, Scott Sandler

Verification methodology has always been a perplexing puzzle. Developing semi-structured approaches has consumed countless years of project and people time, costing the semiconductor industry billions. The growth of integrated circuit complexity has driven the effort required beyond all reason. The advent of Hardware Verification Languages brought the promise of more structure and the potential for great savings. SystemVerilog takes things a couple of steps further by combining design and verification in a single language while retaining the simple elegance of Verilog and adding the modern object-oriented programming capabilities that are key to saving time in complex software projects.

Like any powerful medium, the promise and potential of SystemVerilog comes with the risk of getting bogged down in even more complexity. The emergence of VMM has helped to organize and structure the efforts of design and verification teams adopting SystemVerilog to help them avoid that pitfall. Now once again, Ben Cohen and his co-authors have done the industry a real service by making a powerful technique even more accessible. "A Pragmatic Approach to Adopting VMM…" is exactly what the doctor ordered for demystifying the methodology and focusing users on the keys to applying it efficiently.

Our work at Novas is aimed at making the complex more easily understood. We believe that standards and standardized methodologies are organizing principles that allow the entire semiconductor industry to continue moving forward at its remarkable pace. Building support for the whole of SystemVerilog into our products has been a significant effort, and well worthwhile, as it allows our users to leverage VMM and this book to find and resolve more bugs faster!

Scott Sandler
President and CEO
Novas Software, Inc..
http://www.novas.com/

FOREWORD, Alain Raynaud

With the language well supported by the major simulation tool vendors, 2006 will be known as the year of SystemVerilog. As users start adopting SystemVerilog, they are especially curious of the new testbench constructs and frameworks offered by the language. The Verification Methodology Manual (VMM) is a SystemVerilog-based framework that arrived just in time to put some order in a situation that was about to become chaotic. It remains to be seen if the VMM will become the standard for verification testbenches, just like RTL became the standard subset for synthesis, but this book will serve as a stepping stone for all users wanting to learn about the VMM by example.

I cannot emphasize enough how important it is for the EDA industry as a whole to adopt standards and common practices, so that the limited R&D budgets can be spent where it helps the most, instead of supporting many different and incompatible coding styles and frameworks. In the area of hardware-assisted acceleration and emulation especially, off-the-shelf FPGAs have won the battle for speed. However, accelerating the verification of a design is of no value if its testbench can't keep up. But thanks to the work of the Interface Technical Committee (ITC) at Accellera, the technology is ready today to allow testbenches to exchange data with designs emulating at multiple MHz. What had limited the usefulness of those hardware transactors until today was the manual effort required to retrofit them into old, poorly structured testbenches. The VMM in that regard is the missing link to an efficient verification flow, from well-structured, reusable and easy to maintain testbench in simulation to accelerated simulation where both design and testbench run at speeds previously unheard of.

This book can also be seen as a long overdue training on best practices for writing SystemVerilog-based testbenches. Companies should adopt this approach, if only to save cost, as verification engineers have better things to do than redevelop and learn a new testbench environment every time they change projects.

Alain Raynaud
Technical Director
EVE USA, inc.
http://www.eve-team.com/

PREFACE

The Book

This book is intended to help you come up to speed in the design of SystemVerilog transaction-based testbenches that comply with the *Verification Methodology Manual* (VMM).[1] The goals of this book are to help you adopt, with complete, compilable, and executable examples, the VMM methodology in the creation of comprehensive constrained-random and directed verification environments using a transaction-level modeling (TLM) approach. All code examples are available for download. This book is NOT a repeat of the *Verification Methodology Manual for SystemVerilog* book that defines the rules and techniques used to create a VMM compliant testbench. Instead, this book makes use of the most practical rules of VMM to demonstrate and show how to create a VMM-compliant testbench. Not every feature of the VMM methodology is addressed. However, what is demonstrated are the features and techniques that support transactions; generators; command transactors (e.g., Bus Functional Models); logging of messages; and the verification environment. Since SystemVerilog includes an object-oriented programming language (OOP), we provide applications of OOP design patterns such as factories and callbacks. In addition, we address advanced topics that relate to different applications and verification, including the synchronization of events through the notification services; channel broadcast; channel scheduling; and role of coverage

This book is also NOT an explanation of SystemVerilog, its data types, tasks, assertions, coverage, constraints, etc. That information is available in the SystemVerilog LRM, and in many books[2]. However, we use many of the features of SystemVerilog and provide explanations for some because they need emphasis. We assume that the reader either understands SystemVerilog, or has access to material that explains SystemVerilog. Because SystemVerilog is an extension of Verilog-2001, Verilog users will also be able to benefit from this book to get started with VMM.

We use icons in our text and diagrams to represent key Transaction-Level Modeling (TLM) concepts (e.g., transactions, channel, and transactor). We also use Unified Modeling Language (UML) diagrams created with *StartUML*™ for Windows, or with *Umbrello* for Linux[3] to emphasize the relationships between classes, properties, methods, and SV interfaces. Although UML is primarily used as a documentation tool, in some applications it can be used for the creation of software. However, conversion software from UML to SystemVerilog is not yet available. The UML diagrams show the key, but not all, properties (i.e., variables) and methods of the classes.

---

[1] Verification Methodology Manual for SystemVerilog, Bergeron, J., Cerny, E., Hunter, A., Nightingale, A. 2005, ISBN: 0-387-25538-9, Springer

[2] - IEEE P1800 SystemVerilog: Unified Hardware Design, Specification and Verification Language
- *SystemVerilog for Verification A Guide to Learning the Testbench Language Features Spear, Christian B. 2006, ISBN: 0-387-27036-1 Springer*
- *SystemVerilog for Design · A Guide to Using SystemVerilog for Hardware Design and Modeling, Sutherland, S., Davidmann, S. (et al.), 2006, ISBN 0-387-33399-1, Springer*
- *SystemVerilog Assertions Handbook*, Ben Cohen, Srinivasan Venkataramanan, Ajeetha Kumari 2005 VhdlCohen Publishing

[3] http://www.staruml.com/ *StarUML* - The Open Source UML/MDA Platform
http://uml.sourceforge.net/ *Umbrello UML Modeller*

To achieve the goals of constructing VMM compliant testbenches, we use two models to represent the Design Under Test (DUT): a synchronous FIFO controller model with assertions, and a serial to parallel converter design for the advanced chapter to further explain the capabilities of VMM. For laboratory exercises we're also using a loadable counter, thus making this book useful for training. Every chapter ends with a set of questions that touch on the key points covered in that chapter, with answers provided appendix A. All code presented in the book, including laboratory setups and solutions, is available for download and was verified with VCS X-2005.06-SP2.

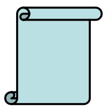
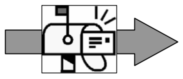
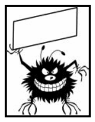
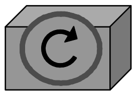
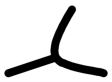
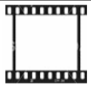
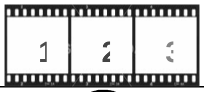
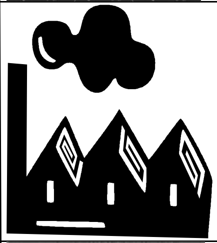
The Intent

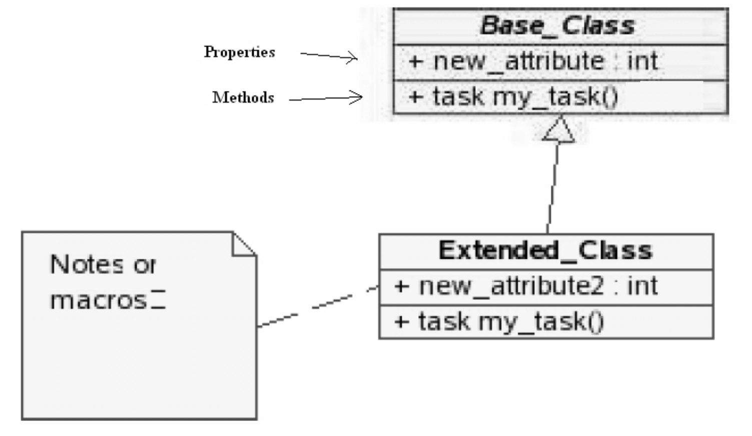
One of the reasons that we decided to write *A Pragmatic Approach to Adopting VMM* is because we truly believe that VMM is a framework technology that allows you to quickly create a fast, reusable, and extendable testbenches using SystemVerilog. This book adds some additional explanations and practical, complete examples to emphasize the features of VMM. This book also presents the results of real-life experiences applying VMM to various problems, providing useful insight and enabling faster VMM adoption.

The Outline

Our approach in explaining the adoption of VMM is to first address in **Chapter 1** the concept of a framework and the role of VMM for verification, including the verification layers. We then delve into key elements of this framework in **Chapter 2,** including the transactions and channels. In **Chapter 3,** we explore two other key elements of VMM, including the transaction generator and the command transactor, also known as bus functional model (BFM). After these elements are explained, we address in **Chapter 4** the construction of the environment where all the different elements/components of VMM are joined together. These first four chapters are fundamental in creating a VMM compliant testbench. **Chapter 5** then adds the use of the factory pattern to provide additional flexibility in the choice of classes to be used by objects. **Chapter 6** covers the concept of callbacks to handle unforeseen functionality through the strategic insertion of callback points, where the implementation can be postponed, if needed. Next, we concentrate in **Chapter 7** on notification services and we apply them to a custom generator with a factory and notification. **Chapter 8** addresses advanced framework topics including the use of scenario generator, channel broadcasters, schedulers, functional coverage, VMM log and formatting of message, and the seeding of randomization. Every chapter has a set of refresher questions and an associated laboratory exercise. **Appendix A** presents the answers to the questions in the chapters. The downloadable code represents all the code in this book, include the laboratory setup and answers to the laboratory exercises. The **Afterword** reflects our opinions on VMM, and in the adoption of SystemVerilog and VMM framework by industry.

The Notation

	**Transaction, a class**		**Monitor, a class**
	**Channel, a class**		**Debug Interface, SystemVerilog interface**
	**Transaction Generator, a class**		**DUT interface SystemVerilog Interface**
	Command Transactor, a class	 	**Directed tests** **Atomic random tests** **Scenario tests**
	Environment, a class		**Factory**
	Callback, a class	1500 HOME VISITOR FOULS FOULS	**Scoreboard, a class**

 Properties Methods Base_Class + new_attribute : int + task my_task() Extended_Class + new_attribute2 : int + task my_task() Notes or macros	**UML diagram** *** Name of class at top.** **Base class shown in italic** **Non-based class in regular font** *** properties (e.g., variables) in middle of block** *** Methods at bottom of block** *** Arrow or lines show relationships**

Acknowledgements

A Pragmatic Approach to Adopting VMM could not have been written without the support of Synopsys who provided us with SystemVerilog tools including *vcs.*[4]

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 for the design of a VMM compliant testbench. We especially thank **Janick Bergeron**, Synopsys Scientist, for his in-depth review of our book, and for sharing with us many useful and insightful guidelines for the application of VMM. We also appreciate his time for all these efforts and for writing a foreword. We thank **Vikas Gautam,** Synopsys Director of CAE and **Chris Spear,** Verification Specialist Synopsys, for making themselves available to provide excellent feedback on SystemVerilog, content, style, and approaches used in the book. We appreciate all the technical and editing comments from **George Bakewell,** Novas Software, Inc, **Alain Raynaud**, EVE USA, and **Stuart Sutherland**, Sutherland HDL. We also are grateful for the forewords from **Scott Sandler**, Novas Software, Inc, **Alain Raynaud**, EVE USA, and **Stuart Sutherland**, Sutherland HDL.

I, Ben Cohen, thank my wife, Gloria Jean, for supporting and bearing with me again in this endeavor.

[4] http://www.synopsys.com/products/simulation/simulation.html

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 an HDL and Property languages (PSL, SystemVerilog Assertions) trainer and 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 (with ECSS and Simscript), instruction set descriptions (with ISPS), and hardware models (VHDL, Verilog). He applied VHDL since 1990 to model various bus functional models of computer interfaces. He authored *VHDL Coding Styles and Methodologies,* first and second editions, and *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, Guide to Property Specification Language for ABV (1st Edition,* also translated to Japanese by Cadence*), Using PSL/Sugar for Formal and Dynamic Verification, 2nd Edition*, and *SystemVerilog Assertions Handbook* (also translated to Japanese). 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 is currently a member of the *VHDL* and *Verilog Synthesis Interoperability Working Group of the Design Automation Standards Committees*, and *Accellera* OVL and PSL and SVA standardization working groups. He taught several VHDL, PSL, and SVA training classes.

VhdlCohen Publishing
Email: ben@systemverilog.us
Web: http://www.systemverilog.us/

Srinivasan Venkataramanan is currently employed as a Corporate Application Engineer (CAE) Manager with Synopsys, India Private Ltd., Bangalore – India. His areas of interest are the emerging verification solutions and methodologies such as SystemVerilog, VMM, Assertion-Based Verification, formal verification etc. He provides support to leading edge semiconductor design companies on their verification methodologies and challenges. In his previous employment with various design houses, he was actively involved in the verification of leading edge high-speed, multi-million gates ASIC designs. He successfully developed complex verification environments using advanced methodologies, such CDV using Verisity's *Specman*, ABV 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 with several HDLs and HVLs, including VHDL, Verilog, *Specman*, and *Vera*. Srini is active in several technical discussion forums, working committees of OVL, SystemVerilog, VHDL-200X Testbench and Verification, 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 book *Using PSL/Sugar for Formal and Dynamic Verification, 2nd Edition* and *SystemVerilog Assertions Handbook.*

Web: http://www.noveldv.com/
Email: sri@noveldv.com

Ajeetha Kumari is running a Bangalore-based Verification Consultancy firm named *Contemporary Verification Consultants (CVC)*, and consults for various customers on high end verification methodologies and languages. CVC offers corporate and educational trainings in areas such as VHDL, SystemVerilog, SVA, PSL, VMM, etc. She co-authored *Using PSL/Sugar for Formal and Dynamic Verification, 2nd Edition* and *SystemVerilog Assertions Handbook.* Her interests include front-end design and verification methodologies including the application of VMM. She has also been involved in EDA tool evaluations. She has experience with several HDLs and HVLs including Verilog, VHDL, SystemVerilog, PSL, SystemVerilog Assertions and Vera. She currently maintains a Verification centric web site. She received her M.S. in Electrical engineering from the prestigious Indian Institute of Technology (IIT), Madras and Bachelors from TCE, Madurai.

Contemporary Vérification Consultants Pvt. Ltd.
Email: ajeetha@noveldv.com ajeetha@abv-sva.org
Web : http://www.noveldv.com/

DISCLAIMER