Learning to assert yourself

Grant Martin
Tensilica

A variation on an old saying goes something like this: If you give an engineer a set of verification IP components, they can verify one design, but if you teach them to develop their own verification IP they can verify designs for the rest of their lives! Creating Assertion-Based IP, written by two of the foremost experts in the field, Harry Foster and Adam Krolnik, is such a tutorial. If you open this book without knowing very much about creating assertion-based verification IP for real designs, you will close it with a reasonable understanding of how to do so for various classes of design blocks found in today’s systems. This book uses the specific syntax of SystemVerilog Assertions (SVA) and the verification environment of the Advanced Verification Methodology (AVM), which is a subset of the Open Verification Methodology (OVM). However, the same principles apply to other assertion specification languages and within the context of other verification environments.

Perhaps most importantly, this book teaches a way of thinking about creating and using assertion-based verification IP. Another aspect of the book that I particularly like is that it is extremely well-written. It has a natural, flowing style that moves the reader through a topic and between topics in a way that’s easy to follow.

The book is organized into a logical sequence. The first chapter gently introduces the concept of assertion-based IP; properties and assertions; and background on abstractions, reuse, languages, libraries, and use models. The second chapter is well-placed, as it clearly defines the notations and terminology used throughout the rest of the book. Chapter 3, the last introductory chapter, defines a verification process and the assertion-based IP architecture discussed in the remaining chapters. Thus, this chapter develops the general approach of interface-based assertion IP and associated analysis ports that are fundamental to this book.

The next set of chapters are based on a modern bus-based digital design consisting of various types of components: bus interfaces, arbiters, controllers, and datapath components. Chapter 4, which is only three pages long, gives a short overview of the components discussed in Chapters 5 through 8.

All the remaining chapters then follow a common structure: First, the authors introduce components in each class and give an overview description, including well-drawn block diagrams. This is followed by a simple set of properties described in a natural-language format. This property set serves as a typical example of the component class under discussion, although it is not a complete set of properties, and each example set covers an interesting variety of the important properties to be verified. Next, the authors provide assertion examples in SVA based on the use of the AVM infrastructure. Finally, they show how to encapsulate the properties inside a module.

This approach is both tutorial in nature and deliberately repetitive. By using this common style and repeating some of the same language from one component type to another, the authors make it easier for readers to move between sections or to dive into a later section without having to read the entire book from cover to cover. Each example is extensively worked out with code, tables, and timing diagrams, as
well as textual explanations. The explanations go into
detail when required without repeating the relatively
obvious parts. This approach lets readers concentrate
on the more complex pieces of each example.

Two appendices—one a tutorial overview of SVA, the
other a complete OVM-AVM testbench example—
complement the main part of the book. The first
appendix is a quick guide to SVA; the second one
enables readers to apply the examples more quickly,
especially since OVM and AVM are freely downloadable.
Despite being specific to a language and an environ-
ment, the principles and concepts of Chapters 5 through
8 are applicable to other languages and contexts.

Any team wishing to learn how to develop
assertion-based verification IP would benefit from this
book as a central resource for the team and as a basis
for their own experimentation. The authors’ emphasis
on following common processes and their consistency
in keeping definitions of properties, assertions, and
portions of the testbenches relatively orthogonal and
as simple as possible (rather than as intertwined
complex pieces of IP that try to do everything in one
module) is very helpful.

This book could also be used as a text or reference
for university-level verification courses. However, this
brings me to my major problem with the book: its
price. At $129, it is way too expensive for individual
engineers or students to buy. Given its tutorial nature, it
is more of a “teach-me” book to be used for a
relatively short time and then occasionally as a
reference, rather than something to use on a continual
basis or as a deep reference. Being a hardcover book
makes it more expensive than, say, a more natural
format in soft cover. If it were priced well below $60, it
would be easier to see more people being able to buy
an individual copy.

But price notwithstanding, if a course or design
group finds a way to share this book’s valuable
tutorial content, they will certainly derive value from
it. Although it is not aimed at the experienced
assertion-based verification IP creator, it can never-
theless be a good vehicle for helping to develop the
next generation of such engineers.

Direct questions and comments about this depart-
ment to Grant Martin, 2424 Raven Road, Pleasanton,
CA 94566; gmartin@ieee.org.

For further information on this or any other computing
topic, visit our Digital Library at http://www.computer.
org/publications/dlib.