国外数字系统设计经典教材系列

A SystemC Primer, Second Edition

[美] J.BHASKER 著

SystemC 入门

夏宇闻 甘伟 译

北京航空航天大学出版社

CD-ROM INCLUDED

国外数字系统设计经典教材系列

SystemC 入门(第 2 版)

A SystemC Primer, Second Edition

[美] J. BHASKER 著

夏宇闻 甘 伟 译

北京航空航天大学出版社

内 容 简 介

SystemC 既是系统级语言，也是硬件描述语言。本书介绍的是 SystemC 2.0 标准，主要介绍 SystemC 有关硬件建模方面的语法特性，换言之，是介绍 SystemC 的 RTL 可综合子集。其主要内容包括：SystemC 数据类型、组合逻辑建模、同步逻辑建模、三态驱动器建模、常用的设计函数模型、测试平台的编写及系统级建模的功能等。随书附带 1 张光盘，内含本书所有例子的代码。本书所有例子都经过 SystemC 2.0.1 的验证。

本书可作为想要了解和学习 SystemC 的设计工程师和系统工程师的参考书，也可用做大学讲授体系结构、数字设计或系统设计课程的教材。

图书在版编目(CIP)数据

SystemC 入门：第 2 版/(美)巴斯克(Bhasker,J.)著；夏宇闻，甘伟译. —北京：北京航空航天大学出版社，2008.9

ISBN 978-7-81124-249-2

Ⅰ.S… Ⅱ.①巴…②夏…③甘… Ⅲ.C 语言—程序设计 Ⅳ.TP312

中国版本图书馆 CIP 数据核字(2008)第 096079 号

SystemC 入门(第 2 版)
A SystemC Primer, Second Edition
[美] J. BHASKER 著
夏宇闻 甘 伟 译
责任编辑 宋淑娟
*
北京航空航天大学出版社出版发行
北京市海淀区学院路 37 号(100083) 发行部电话：010-82317024 传真：010-82328026
http://www.buaapress.com.cn E-mail：bhpress@263.net
涿州市新华印刷有限公司印装 各地书店经销
*
开本：787 mm×960 mm 1/16 印张：16.5 字数：370 千字
2008 年 9 月第 1 版 2008 年 9 月第 1 次印刷 印数：5 000 册
ISBN 978-7-81124-249-2 定价：36.00 元(含光盘 1 张)

译者序

J. Bhasker 在数字集成电路设计界具有很好的声誉。包括美国和中国在内的各国数字电路和系统设计者，无论是新手，还是从 VHDL 转到 Verilog 的老手，其中很多人都阅读过 J. Bhasker编写的 *A Verilog HDL Primer*。J. Bhasker 的教材以简明扼要、清晰易懂著名。*A SystemC Primer*一书是介绍 SystemC 基础知识的教科书，符合他编书的一贯风格，特别适合作为入门教材。

SystemC 对许多中国读者来说，还是一种新的硬件描述语言。近年来，由于基于平台设计方法学的推广，系统验证已成为设计工作中的瓶颈。而 SystemC 由于是一种基于 C++ 的语言，具有学习方便的优点，更重要的是其高速的仿真性能已得到多家设计公司的认可，因此被广泛用做 SoC 系统验证语言。由 OSCI 组织的推荐，2004 年 SystemC 2.0 被正式批准为 IEEE 标准，与 SystemVerilog 并列成为最主要的系统验证语言之一。

本书是根据 J. Bhasker 编写的 *A SystemC Primer* 第 2 版翻译的。2004 年起原书在数字集成电路验证界逐渐走红并非偶然。在世界各地的大学中，教师和学生们所熟悉的主要编程语言和环境是 C++。而 SystemC 是 C++ 的一个子集合，不存在编程环境和学习基础的问题，所以比 SystemVerilog 更便于推广和应用。

本书的翻译工作安排如下：第 2 版序言、序言、前言、第 1～6 章及附录 A、B 和索引等由夏宇闻负责，第 7～9 章由神州龙芯 IC 设计公司的甘伟工程师负责。全书最后的审校与定稿由夏宇闻负责。

在神州龙芯 IC 设计公司工作的工程师和实习研究生樊荣、洪雷、周鹏飞、刘家正、陈岩、李鹏、宋成伟、邢志成、管丽、徐伟俊、杨鑫、苏宇、张云帆、邢小地、李鹏、李琪、陈岩等认真阅读了最后完成的翻译稿，并提出了许多改进意见，使翻译工作的质量有了显著提高。在翻译稿最后完成之际，谨向他们表示诚挚的感谢。

全书翻译稿完成之后，我以前的学生、上海澜起 IC 设计公司的技术总监山岗、威盛 IC 设计公司的高级设计师杨柳女士和田玉文女士等认真阅读了全书，并提出了宝贵的修改意见，在

此一并表示感谢。

2006 年我从北京航空航天大学退休后,受曾明总裁的邀请到神州龙芯 IC 设计公司担任顾问。本书的翻译工作是在他的支持下完成的。该公司不但为我提供了舒适的办公条件和自由宽松的工作时间,而且还为我安排了既谦虚好学又能干的年轻工程师甘伟担任助手。没有曾明总裁的支持,本书的翻译工作不可能既快又高质量地完成。在本书出版之际,让我向曾明总裁、甘伟工程师和神州龙芯 IC 设计公司的全体员工表示衷心的感谢。

夏宇闻

北京航空航天大学教授

2008 年 4 月 1 日于神州龙芯 IC 设计公司

第2版序言

自第1版出版以后，又公布了SystemC的更新版V2.0.1，其中包括已更新的主从库。SystemC验证工作小组开发了一个SystemC的验证库，这是一个源代码开放的C++库，它提供了用SystemC构建高级验证组件和测试平台的标准方法。该验证库支持基于事务的验证、事务的监视和记录，以及有权重和受约束的随机化。

第2版书中附带了一张光盘(CD)，其中录有本书所有例子的代码。在光盘内的README.txt文件中，提供了与一些可用的SystemC试用软件的链接。第2版中还增加了一些新的例子，对SystemC的语法特征进行了许多扩展，使得这些概念更加清晰易懂。书中练习题也有所增加。

第2版书中的所有例子都经过SystemC 2.0.1的验证。

请继续使用我的电子邮箱jbhasker@esilicon.com或通过我的出版商把意见直接反馈给我。

J. Bhasher

2003年11月

序　言

您为什么挑选本书？我认为这是为了学习 SystemC 所做的正确选择。让我们切入话题，本书将直接把读者引入 SystemC 领域。只须阅读第 2 章和附录 A，读者就能在很短的时间内开始编写 SystemC 的模型并进行仿真。

SystemC 既是系统级语言，也是硬件描述语言。只须使用这一种语言就能为硬件和软件系统建模。说 SystemC 是硬件描述语言，是因为这种语言可以为寄存器传输级 RTL (Register Transfer Level)的设计建模。说它是描述系统级规范的语言，是因为它可以在算法级为设计建模。读者也可以用 SystemC 为自己设计的整个系统建模，就像编写软件程序那样，描述该系统的行为。虽然 SystemC 甚至能描述门级网表，但并不提倡那样做，因为用它描述门级网表来建模很麻烦，也没有效率。

SystemC 是建立在 C++ 编程语言基础之上的。它扩展了 C++ 的表达能力，使其能描述硬件。SystemC 添加了诸如并发、事件和数据类型等重要概念，这些能力是通过类库提供的。该类库提供了功能强大的新机制，如可以用所组成的硬件元件、并发行为以及反应性行为为整个系统架构建模。这些只不过是一些建立在 C++ 编程语言类结构顶层的新机制。SystemC 提供了一个仿真内核，允许用户对设计或系统的可执行规范(executable specification)进行仿真。

本书描述了 SystemC 2.0 标准，该标准是由开放 SystemC 提案 OSCI(Open SystemC Initiative)协会的语言工作小组制订和维护的。其目标是在不久的将来将 SystemC 变成一个 IEEE(Institute of Electrical and Electronics Engineers)的标准。读者可从以下网址得到有关 SystemC和 OSCI 的更多信息。

http://www.systemc.org

完整的 SystemC 2.0 标准采用功能规范来描述，读者可从上述网址下载属于 SystemC 2.0软件包一部分的 SystemC 2.0 的用户指南。

开放 SystemC 提案协会在许多家公司的协作和精心策划下于 1999 年成立。1999 年 9 月 SystemC 的第一个版本,即 SystemC 0.9 作为开放源代码向社会公开发布。SystemC 1.0 于 2000 年 3 月向社会公开发布。该版本只限于行为和寄存器传输级的建模,缺少了许多系统级建模的特性。

发布于 2001 年 10 月的 SystemC 2.0 包含了许多系统级的建模特性。这些新特性包括通道、接口和事件等。本书是基于 SystemC 2.0 版本编写的。

本书主要向读者介绍 SystemC 有关硬件建模方面的语法特性,换言之,是介绍 SystemC 的 RTL 可综合子集。用该子集编写的模型可以首先被综合为逻辑门,然后再用硬件来实现这个模型。本书把重点放在硬件建模方面主要出于以下三方面的考虑:

① 当今了解 VHDL(VHSIC Hardware Description Language,IEEE Std 1364)和 Verilog HDL(Verilog Hardware Description Language,IEEE Std 1076)的硬件工程师可能想了解和学习 SystemC。随着抽象级别从寄存器传输级提高到更抽象的层次,目前用 RTL 建模的设计工程师将不得不学习系统级建模。本书以一种很自然的方式通过介绍 SystemC 为这些想学习的设计师架起了桥梁。

② 编写高级算法软件模型的系统设计师需要理解 RTL 综合级,以便将算法模型不断地细化到寄存器传输级,从而将编写的模型转变为具体的逻辑门。

③ 使用 RTL 可综合子集,模型的编写者可以开发 SystemC 的知识产权 IP(Intellectual Property)模型。这样做可使 IP 模块成为可综合的模块,并使它们得到重复利用。

本书是特别针对想要了解和学习 SystemC 的设计工程师和系统工程师编写的。

这是一本为初学者编写的入门书,因此不涉及某些高级 SystemC 话题。例如,主从通信库,SystemC 定义它为方法学的专用库,这些内容超出了本书的范围。

本书可以用做大学和学院讲授体系结构、数字设计或系统设计课程的教材。用 SystemC 作为大学教材最大的好处在于,SystemC 仿真器和 SystemC 所有的类库都有公开的源代码,每个人都可以免费使用。这一点对许多大学教授很有吸引力,这是因为讲解和理解这种新的系统级设计语言不需要投入新的资金。正因为 SystemC 是一种源代码开放的语言,所以大学里进取心强的学生就可以通过直接修改源代码,在语言特性和程序优化两方面不断地扩展 SystemC 现有的功能。

本书所描述的 SystemC 的 RTL 可综合子集是基于我对可综合的 SystemC 子集的理解。该子集与符合 IEEE 标准的 Verilog 和 VHDL 语言的 RTL 可综合子集[①]之间有着密切的对应关系。这两种语言在我的领导下已经(正在)完成了标准化的工作。当前可用的综合工具中有些支持也有些不支持这个可综合子集。至于某个具体的综合工具究竟支持子集中的哪些特

① VHDL 语言的 RTL 可综合子集的 IEEE 标准号是 IEEE 1076.6。Verilog 语言的 RTL 可综合子集的标准草案号是 IEEE P1364.1。

性，建议读者查阅有关工具的文档。

阅读本书并理解 SystemC 时所需要的背景知识有哪些？首先，读者必须知道 C++ 的编程基础，这是最基本的要求。其次，读者还应该有逻辑设计的背景知识。若您早就掌握了 VHDL 或者 Verilog HDL 这两种最常用的硬件描述语言，则选用本书来学习 SystemC 会觉得非常容易（我在本书中，凡用到 SystemC 模型的任何地方，故意不列出等价的 VHDL 和 Verilog HDL 模型）。如果您对 C++ 编程语言了如指掌，那么就会发现用 SystemC 编写高级系统级模型是一件轻松愉快的工作，同时您也能进一步理解 SystemC 内部的诀窍。然而，在读本书之前，读者并非必须掌握 VHDL 或者 Verilog HDL。如果您想充分发挥 SystemC 的强大功能，就必须是一个有很深造诣的高级 C++ 用户。但是如果您只想为硬件建模，或者只想理解 RTL 可综合子集，则只须了解 C++ 编程的基础知识即可。学习 C++ 编程语言的一本好书是 *C++ Primer, Third Edition*，作者是 Stanley B. Lippman 和 Josee Lajoie，由 Addision-Wesley出版社于 1998 年出版。

本书描述的所有模型都在 Solaris 计算机上进行过测试和仿真。综合后的逻辑图是通过手工方法将 SystemC 模型转换为与其等价的可综合的 Verilog 模型，然后再用 Ambit Build-Gates 综合工具对 Verilog 模型进行综合后得到的。

SystemC 的未来

SystemC 语言仍处于不断修改的过程中，在成为 IEEE 标准之前，这种修改会以更快的速度继续下去。即使标准出现后，有用的标准还会继续进化。计划中的 2.X，3.0 和 4.0 版将会陆续公布。2.X 版计划将包括行为层的语法 fork，join 和 interrupt/abort；还将包括支持性能建模和支持时序规范的描述。3.0 版计划支持抽象的 RTOS(Real Time Operating System)建模和调度建模。4.0 版计划支持模拟混合信号系统建模。

致 谢

下面名单中的这些人，帮助我审阅了本书的初稿，为我提供了建设性的反馈意见和新的想法，他们的工作显著提高了本书的质量，在此向他们表示深深的谢意。他们是：

Mike Baird
Abhijit Ghosh
Thorsten Groetker
Jeff Hantgen
Kurt Heinz
Sven Heithecker
Xiaoyan Huang

Martin Janssen
M. N. V. Satya Kiran
David Long
Grand Martin
Dale Mehl
Sanjiv Narayan
Smail Niar
Bernhard Niemann
Stuart Swan
Kartik Talsania
Punitha Thandapani
Yves Vanderperren
Jean Witinski

衷心地感谢你们对本书做出的贡献。

最后,我再说一句发自内心的话:若没有我妻子 Geetha 和我的三个孩子 Arvind,Vinay 和 Vishnu 的不断支持,本书是不可能完成的。

欢迎对本书的任何建议和提出阅读中发现的任何错误。请通过我的出版商或我的电子邮箱 jbhasker@cadence. com 把建议和错误发给我。

J. Bhasker
2002 年 4 月

前　言

SystemC 是电子设计界中多种相关紧迫问题相互作用的副产品。最值得注意的两个“应力点”是：

① 电子产品推向市场的时间要求日益缩短；

② 电子产品越来越复杂，以及基于平台设计方法学日趋成熟。

SystemC 是为了帮助解决上述两个和其他一些“应力点”而诞生的，下面将对此进行简单的讨论。

电子产品推向市场的时间要求日趋缩短

电子产品制造厂商满足消费者要求的方式已经把顾客宠坏了，顾客不断提出越来越苛刻和奢侈的要求。不久前还只能在科幻小说中出现的电子产品现已摆满了处理品商店的货架。大家只须看一下手机或者便携式计算机的大小和性能就会明白这一点。

此外，无论对电子产品的生产厂商还是对销售商而言，利润空间已经非常狭小，因此厂商们有极其强大的动力，不断地向市场引入更高级的先进产品，期望消费者来购买今年新推出的成功产品，以替换去年还是新奇的那些老产品。制造商之间的激烈竞争，消费者对品牌缺乏忠诚度，都激发了螺旋式地不断引入功能更强大和价格更低廉的电子新产品。假如你不能及时满足顾客的需求，那么别人就会很快地满足他们。

制造商们可以采用多种方法来缩短新产品的上市时间。加快把设计思想转变为现实产品的一种根本方法是缩短设计周期。尽可能早地发现设计错误的方法学显然能缩短设计周期，以避免或者减少为了纠正这些错误而多次反复设计。然而，缩短生产集成电路所需时间的最根本途径还在于缩短设计流程各个阶段所花费的时间。

正如已被事实所证明的那样，功能验证，换言之并从更广的角度考虑，也就是对电子产品是否能完成所要求的设计功能进行验证。它不但是每个设计过程的关键步骤，而且也是实质

上的瓶颈。这种功能验证可能涉及产品质量的检查,例如当掌上电脑运行字处理程序时,是否还能继续播放 MP3 录音文件? 是否会听到讨厌的音频噪声? 换言之,有必要确认所设计的电子产品是否已经达到双方一致同意的质量标准,例如是否已达到 802.11b 无线网络标准。最后,也有可能要确定:当若干不正常事件碰巧同时发生时,所设计的电子产品是否仍能正常工作而不出现故障,例如同时按下"启动"和"停止"两个按钮时会发生什么情况。

如果想保持产品的质量,这种类型的功能验证肯定是至关重要的。但不幸的是,由于复杂电路的行为是用软件建模的,所以在软件验证的系统环境下,这种模型固有的执行速度非常慢。事实上,如果设计模型能将被设计的器件在每一个时钟周期下的每一个状态都表现出来,那么逐一验证该器件性能所花费的时间很可能超过几百年。

采用 SystemC 可以加快这种功能验证,因为 SystemC 允许将设计先编写成与时间无关的形式,换言之,先不考虑实际器件中所采用的时钟方案。这样做能显著加快仿真的速度,这是因为时钟信号的变化占据了由事件驱动仿真器在仿真期间必须处理的大部分事件。事实上,这种与系统时钟无关却仍旧能进行仿真的想法是由于引入了 VHDL 变量赋值的概念后才得以实现的。但遗憾的是,所谓的"行为 VHDL",其语言表达能力十分有限;而 SystemC 是建立在相当通用的 C++ 基础之上的一种语言,所以避免了这种行为表达的局限性。

本书的亮点是使用了 C++,如果只依靠 SystemC 本身,并不能显著加快仿真的速度。如果设计中所有时钟的跳变沿都用 SystemC 来表示,那么其仿真速度可能比相应的 VHDL 或 Verilog 模型还要慢或者相差无几。在这种场合下,仿真速度的任何提高都必须依赖仿真器开发者对程序进行优化。然而,依赖仿真程序的优化所增加的仿真速度实在是微不足道,根本无法与在更高的抽象级别上描述设计所增加的仿真速度相比拟。

电子产品日益增长的复杂性和基于平台的设计

人们通常以巨大的热情关注单个硅芯片上能够集成的三极管的数目,即大众出版物经常引用的"摩尔定律"。本书对单芯片上三极管的数目并不十分关心,而把关注的重点放在单个能运转的芯片上集成更多的功能部件。今天"系统芯片"这个技术词汇已经成为现实。全功能系统芯片已经被制造出来了,实际上这是由多个复杂处理器和它们的外围设备、数字信号处理器、多层总线、多个存储器,以及在过去可能都是独立的专用集成电路(ASIC)芯片(例如 MPEG 模块)等组成的单个硅芯片。

这是一个巨大的技术进步,因为系统芯片允许多个器件互相"交谈"而不必付出芯片之间数据交流固有的必须降低通信速度的代价。另外,将整个系统集成到单个硅片上可以使电子产品的体积大大缩小,这一点对于消费类电子产品特别重要。

真实的系统芯片的出现以及快速跟上市场步伐的需求使得一个新的技术词汇——"基于平台的设计"开始频频出现。基于平台的系统,其基本思路蕴涵着用一种全新的分工合作来进

行复杂系统设计的概念，即由硅片生产厂商开发一个基本的硅平台，上面有完整的电路部件，包括各种处理器(控制用的处理器和数字信号处理器)，各种档次的存储器和各种可能用到的专用模块。系统厂商即使有能力开发这个平台，通常也没有开发这个基本硅片平台的必要(动力)，他们只是在这个平台上可编程的部分做一些增值的开发工作，例如为某种处理器编写一些新的软件，或者在基本硅平台上的 FPGA 部分(如果该平台有 FPGA 的话)编写一些专用的硬件程序。

下面举一个简化的具体例子来说明这一点。某硅片制造厂商开发一个硅片设计平台，应用此平台可进行第三代蜂窝电话核心处理器的开发。有一家以销售这种电话为主业务的系统公司选用了该基本平台，在该平台的基础上对该公司正在开发的移动电话样机进行编程，这种编程可以是软件编程，也可以是通过 FPGA 进行的硬件编程，最终开发出了具有该系统公司特色的(移动电话)手机。例如，该系统公司可能添加了一些用户接口软件，使得该公司的手机非常容易使用；还可能将某些节能 IP 编写到该平台的 FPGA 电路块中，延长了电池的使用寿命。经过这些编程，功能增强的平台(即新手机的样机系统)又返回到硅片制造厂商，然后硅片制造厂商为该系统公司生产具有该公司特色的手机。

这样的交易对业务双方都是非常有利的，因为硅制造厂商使得其生产流水线上的任务饱满，而系统公司则集中精力开发有别于其他公司的特色产品。事实上，系统公司只要在该硅开发平台上，为同一系列、型号不同的产品编写各自不同的软/硬件程序就可以创建整个系列产品。低端产品的芯片可能不具有功耗监控装置，而高端产品的芯片通过编程可以添加无数的功能，同一系列中，档次低的产品只能具有档次高的产品的部分特性。而采用同一个硅平台进行设计是该系列产品的基础。

然而，上述方案中仍潜伏着一个问题：如何才能将这个开发平台的特性介绍给开发客户样机的系统设计师呢？在硅开发平台上，设计通常是用硅电路来表示的，例如版图布局、门级描述等，这种传统的表示方法对系统设计师而言可能很难理解。用 VHDL 或者 Verilog 语言编写的寄存器传输级别(RTL)的描述也存在同样的问题。事实上，即使采用如英语那样的自然语言进行文字描述，其用处也极其有限，因为篇幅实在太长。

显然，我们所需要的是一种“可执行技术规范”(executable specification)。这种可执行的规范是在该开发平台上可以运行的仿真模型，通过输入测试平台(test benches)(即公司自己编写的或者由硅平台提供者编写的测试代码)，可以执行该仿真模型。通过观察该硅开发平台不同部分对不同输入激励所产生的响应行为和查看该设计平台不同部分的源代码，系统工程小组就能得到准备研发样机的有关技术规范知识。

这样做是可行的；然而，只有当“可执行技术规范”的仿真模型的运行速度足够快时，才能有效地观察到其行为。而且系统设计小组需要阅读该技术规范，所以编写的技术规范说明书必须让别人能够明确地理解。换言之，描述的抽象层次应当合适。因此，类似的情况，例如在网络设计中，块之间的通信最好用分组报文的形式描述，而不用底层的信号变化来描述。

必须建立能够高速执行的模型,必须建立一个以较高的抽象层次来表示低层次的结构的模型,正是这两个原因,所以必须使用以未定时方式描述的 C++ 代码来表示这个设计平台。如上所述,用 C++ 编写的未定时模型执行速度快,而且 C++ 这种语言非常适合定义易于被系统设计师理解的抽象数据类型。

此外,采用 C++ 编写可执行技术规范还有一个附带的好处,即它可以很自然地与系统设计小组为开发客户样机平台而编写的那些 C++ 软件模型衔接。这样做就能允许创建一个更大规模的可执行技术规范,即在原来硅设计平台基础上又加上了为客户样机平台编制的软件。系统公司和硅设计平台的供应厂商都可以使用这个规范来理解客户样机平台的行为。实际上,若 EDA 厂商开始开发可以将 C++ 代码综合成 FPGA 网表的工具,则描述客户硬件的平台模型也可以很容易地与初始平台的技术规范衔接。

对上述情况了解之后,则对无论是硅设计平台的生产者还是使用者都乐于采用包括使用 SystemC 建模在内的设计风格便不足为怪了。

为何要采用 SystemC?

如果读者能够接受上述观点,那么就能清楚地理解对于设计方法学正在日益增长的需求,而使用 C++ 建模就是设计方法学之一。然而,本书并不是讨论由开放 SystemC 提案 OSCI 协会(Open SystemC Initiative)所定义的 SystemC。C++ 是一种相当开放和容易扩展的语言,任何公司都可以根据其人力资源的水平,定义使用 C++ 的风格,以满足上述需求。事实上,许多公司已经这么做了。

但由 OSCI 协会定义的 SystemC 有其合理性。若每个公司(或者由几个公司组成的公司小组)都自行定义使用 C++ 进行设计的方法学,则各公司之间互相便利地交流设计模块的能力会极大地削弱。当在公司之间共享设计模块时,设计模块的接受者必须首先理解该设计模块提供者所用的方言,显然这会妨碍在硅平台供应商和使用者之间进行设计模块的交流。

另外,这样做也会破坏第三方知识产权(IP)提供者的能力。举例说明如下。对于处理器模型的开发者,他们开发的模型可以被许多公司采用。若使用多种 C++ 方言来编写设计代码,则完整系统的开发者不得不完全依靠自己来编写所有子系统的模块,或者不得不将第三方提供的模块改造成符合自己要求的 C++ 标准。设计进度的快慢对项目的成功至关重要,在这样的市场环境中,这两种不得不采用的手段显然都毫无吸引力。

现在只剩下两条出路:①允许描述设计模型的 C++ 子集的"事实"标准在应用过程中慢慢地成熟;②硅供应商、系统公司、IP 开发者、EDA 工具供应商和 C++ 用户一起合作,开发出大家一致同意的、描述设计模型的 C++ 标准子集。第一条出路并非不可能,这是由于Verilog 的发起者 CADENCE 设计系统公司的多年推广,使得 Verilog 已经成为非常成功的硬件描述语言的事实标准;在 C++ 领域角逐硬件描述语言的情形与此类似,C++ 子集逐渐成为事实标准确实是有可能的。另一条出路,VHDL 过去之所以能成为工业标准,正是因为整个 IC 工业

界当时已清楚地意识到，开发一种可在行业内统一使用的语言是极其必要的。OSCI 清晰地呈现出沿着 VHDL 模式发展的思路，这是由于等待合适的事实标准逐渐成熟的风险实在太大了，所以不能等待，只能合作开发 SystemC 标准。

实在没有时间等待 C++ 事实标准子集的逐渐成熟和出现，所以别无其他退路。电子工业的强烈需求要求我们尽快制订出使用 C++ 对硬件进行描述的标准方法学，并要求该新标准能够满足尽可能多的潜在用户的需求。OSCI 协会曾是基于这种想法而成立的，目前也正是基于这个共识，代表半导体公司、系统公司、IP 供应商和 EDA 公司的成员正在不断地加入到 OSCI 协会中来，使得协会继续健康发展。

我为什么推荐本书？

我认识 Bhasker 差不多有 20 年了，那时他刚从明尼苏达大学获得博士学位，加入我在 Honeywell 实验室的课题组。当时他那么年轻就已经具有了洞察深奥技术问题的杰出能力，并能给那些比他更年轻的组员们透彻地讲解这些难题。讲解艰深技术资料的能力随着他的工作而逐年变得更强。Bhasker 已出版了许多本关于 VHDL、Verilog 和逻辑综合等方面的教材，这些教材已被世界各地大学广泛选用。他的教材组织严密，并以学生们容易理解的方式把电子设计界的“原始材料”清晰地呈现给学生。他的教材已经帮助培养了一代工科学生。

《SystemC 入门》这本独特的教材一定会像 Bhasker 以前所编写的书籍一样，对 SystemC 领域产生同样重大的影响。本书并不是为高级研究人员和语言学家编写的，而是为 SystemC 的入门者编写的基础课本。本书参照常用数字设计的概念，循序渐进地向读者介绍 SystemC 的各种复杂特性。作者采用通俗易懂的讲述风格，使得读者能够很快理解 SystemC 的基础要点，不但可以帮助读者采用 SystemC 语言开始进行设计，而且还可以帮助读者进一步研究该语言的更高级的特性。

最终会有那么一天，SystemC 就像今天的 VHDL 和 Verilog 那样变成设计领域中应用最广泛的语言，那时 SystemC 才能成为真正有用的标准。《SystemC 入门》的出版是促使这一天早日到来的最好催化剂。

Stanley J. Krolikoski
OSCI 协会主席
加州 圣何塞
2002 年 3 月

目　录