系统设计

IP

RTL

SystemC™ 基础教程

A SystemC™ Primer

嵌入式软件

验证

J. Bhasker 著

孙海平等 译

清华大学出版社

系统设计

IP

RTL

嵌入式
软件

验证

J. Bhasker 著
孙海平等 译

SystemC™ 基础教程

A SystemC™ Primer

清华大学出版社
北京

J. Bhasker

A SystemC™ Primer

EISBN: 0-9650391-8-8

Original English language edition published by Star Galaxy Publishing.

图书在版编目(CIP)数据

SystemC™基础教程/巴斯克尔(Bhasker, J.)著;孙海平等译. —北京:清华大学出版社,2004

书名原文:A SystemC™ Primer

ISBN 7-302-08418-1

Ⅰ. S… Ⅱ. ①巴… ②孙… Ⅲ. 硬件描述语言,SystemC—程序设计—教材 Ⅳ. TP312

中国版本图书馆 CIP 数据核字(2004)第 028029 号

出 版 者:清华大学出版社 地 址:北京清华大学学研大厦

http://www.tup.com.cn 邮 编:100084

社 总 机:010-62770175 客户服务:010-62776969

责任编辑:张 靓

封面设计:何凤霞

印 装 者:北京鑫海金澳胶印有限公司

发 行 者:新华书店总店北京发行所

开 本:185×230 印张:14 字数:294 千字

版 次:2004 年 5 月第 1 版 2004 年 5 月第 1 次印刷

书 号:ISBN 7-302-08418-1/TP·6053

印 数:1~4000

定 价:26.00 元

译 者 序

20世纪下半叶，微电子技术得到了长足发展。其中，电子设计自动化（Electronic Design Automation，EDA）技术起到了至关重要的作用。EDA技术引发了整个电子领域的一场螺旋式革命。半导体制造工艺的巨大进步已经使集成电路设计从晶体管的集成发展到逻辑门的集成，现在又发展到知识产权芯核（Intellectual Property core，IP core）的集成，即芯片系统（System-on-a-Chip，SoC）。芯片系统不再是一种功能单一的单元电路，而是将信号采集、处理和输出等完整的系统集成在一起，是在单芯片上广泛采用预先设计好的IP模块，通过各种重用技术快速开发出来的集成电路。这为电子设计带来了前所未有的新挑战，系统设计不得不同时考虑软件和硬件两部分的开发，也不得不考虑知识产权芯核的集成。传统的硬件描述语言将难以胜任诸如系统级建模、软硬件协同设计与验证等艰巨的任务。

近年来新兴的SystemC就是电子设计和EDA领域的十多家大公司为解决SoC时代的新挑战而提出来的。它建立在强大、灵活的C++语言基础之上，具有很强的数据表达能力和计算能力。同时，由于它引入了并发、定时、事件等硬件概念，因而不仅具备了寄存器传输级硬件描述能力，还具备了在系统级抽象层次上进行描述的能力；不仅可以用于软件建模，还可以用于硬件建模，使软硬件协同设计、仿真和验证得以有效实现，从而在设计阶段早期就能够发现和解决问题，减少了设计反复。此外，SystemC的出现也为更好地开发和集成IP芯核提供了可能和保障。这一切都能够有效地帮助设计人员缩短设计周期，赢得更好的市场效益。

目前开放式SystemC联盟（Open SystemC Initiative，OSCI）的各成员单位，如Synopsys、Cadence等公司，都已经推出众多功能强大的、基于SystemC的设计工具，并获得了广泛应用。SystemC还有望成为IEEE标准，届时它将像VHDL和Verilog那样在电子设计领域得到普遍应用。

目前，我国IC产业界和学术界也在广泛讨论SoC设计、IP集成、软硬件协同建模仿真等论题。本书作者J. Bhasker作为负责制定VHDL和Verilog HDL语言综合标准的IEEE工作组主席，对硬件描述语言有着长期深入的研究和全面准确的把握。这本“A SystemC™ Primer”也同“Verilog® HDL Synthesis，A Practical Primer”（该书中文翻译版《Verilog® HDL综合实用教程》已由清华大学出版社出版）一样，内容全面，深入浅出，适用面广，对于打算学习和使用SystemC的学生和工程设计人员来说是一本极好的入门书，可作为SoC设计等相关课程的教材，对于ASIC设计工程师和系统设计工程师也有很高的参考价值。

本书由孙海平、何伟、徐学迅和刘方海主译，孙海平统稿，何伟和郑静等进行了修订和校对。在此，谨向为本书的出版付出辛勤劳动的所有人员致以诚挚的感谢！

译者在翻译过程中力求准确，但限于水平，一定存在错误和不足之处，恳请读者通过电子邮件 hp.sun@263.net 向译者提出批评和指正。

译　者

2003 年 12 月

原 书 序

SystemC 是电子设计领域多个相关的“受力点”相互作用的产物。其中，最有意义的两个“受力点”是：

- 电子产品的面市时间日益缩短。
- 电子产品和大有前途的基于平台设计方法，其复杂度都在不断增长。

SystemC 的诞生，就是用来适应以上这些及其他“受力点”，下面将对此进行简单的讨论。

电子产品的面市时间日益缩短的需求

现在，电子产品制造商们把用户都宠坏了。商店中的廉价电子产品琳琅满目，在不久之前，这常常还只是科幻小说中的景象。只要看看移动电话和手提计算机的大小和性能，你就会明白了。

然而，由于对制造者和销售商来说利润率很低，因此就有了强大的动力促使他们不断将更先进的产品引入市场，期待着顾客以今年的成功产品淘汰那些去年还很奇妙的产品。制造商之间竞争激烈、用户缺乏忠诚度，都加剧了电子产品以迅速推陈出新和以更低价格提供更强大性能的方式螺旋式发展：如果你做不到在顾客需要的时候及时提供其所需，那么别人就会提供。

制造商可以采用很多方法来缩短面市时间。加快将设计思想转变为现实产品的一种根本性方法就是缩短设计周期。缩短设计周期必然涉及在设计过程中尽早发现错误，因此要消除或减少为了修改错误而进行的“循环往复”。但是，缩短生产集成电路所需时间的更根本性的方法，是缩短设计流程中每一个步骤所需的时间。

现实情况是，广义上的功能验证也就是验证产品是否按照设计目标工作，这不仅是设计流程中的关键环节，也是其真正的瓶颈所在。功能验证可能包括产品的质量检查，例如当使用掌上电脑开始文字处理时，是否还能继续播放 MP3 文件，抑或会听到刺耳的噪音？另外，还要求电子设备确实符合公认的标准，如 IEEE 802.11b 等无线网络标准。最后，要求确定电子产品尽量可用，发生异常的操作组合时（如同时按下了“开始”和“停止”按钮），设备也不至于失效。

要保证产品质量，进行功能验证显然是非常重要的。不幸的是，对复杂电路建立软件模型，以及在软件验证系统中运行该软件模型原本就都很慢。实际上，如果将模型设计成将产品在每个时钟周期上的状态都表示出来，那么采用穷举法验证其性能可能会需要若干个世纪的时间。

利用 SystemC 有助于加快功能验证，因为它允许在设计初期编写未定时的模型，即

不必考虑产品实际将采用的时钟方案。这将大幅度地提高验证速度，因为在验证过程中，事件驱动仿真器所处理的事件中很大一部分是时钟信号的变化。实际上，让设计对象独立于系统时钟进行仿真的思路就是将变量赋值引入 VHDL 的幕后动机。不幸的是，所谓的“行为级 VHDL”的表达能力相当有限，而 SystemC 之所以能避免这种局限，原因在于它建立在应用广泛的 C ++ 语言基础之上。

需要说明一点，采用 C ++ ，甚至是 SystemC，其本身并不能额外地大幅度提高仿真速度。如果在 SystemC 设计对象中将所有时钟跃变都表示出来，那么得到的仿真速度就会更加接近其相应的 VHDL 和 Verilog 模型。此时，得到的任何额外的加速将是仿真器研发商优化的结果。但是，这种优化与在更高抽象层次上仿真所带来的加速相比就逊色多了。

电子产品和基于平台设计的复杂度不断增长

人们总是对在单个硅片中能够集成的晶体管数目很感兴趣——科普读物甚至将此称为“摩尔定律”。但是，相对于集成的晶体管数目，人们更感兴趣的是在单个硅片中集成的功能器件的数目。如今，“系统芯片”已成为现实，实现了包括复杂的处理器（及其外设）、数字信号处理器、多层总线、多个存储器以及其他过去可能是独立 ASIC 的电路块（比如 MPEG 模块）在内的全部系统功能。

这是巨大的进步，因为这样能够让多个器件相互“交谈”时不必以芯片间通信必然导致的降速为代价。更何况，将整个系统放到单个芯片中还迎合了产品（尤其是消费类产品）微型化的需求。

以更快的面市时间推出真正的系统芯片，这就导致了“基于平台设计”这一概念的诞生。基于平台的系统，其基本思路孕育着这样一种新的分工：硅片生产商开发基本的硅设计平台，包括处理器（控制逻辑和 DSP）、多层存储器以及各种可能的专用电路块；系统公司不打算（通常也没有能力）开发这种基本的硅设计平台，而是通过支配该平台的可编程部分为平台增加应用价值，也就是说，要么添加处理器上运行的软件，要么对该平台的 FPGA 部分进行编程以实现专用硬件。

因此，以一个简化的方案而言，硅制造商开发硅设计平台，围绕此平台可进行第三代蜂窝电话的核心开发。研制这种电话的系统公司采用该基本平台，并对其进行编程（既包括软件编程，也包括通过 FPGA 进行硬件编程），这些工作内容与系统公司的身份是相称的。比如，系统公司可能会添加用户接口软件以使移动电话非常易于使用；还可能将某些节能 IP 编写到该平台里的 FPGA 电路块中以延长电池寿命。经过这些编程，增强了平台功能，交付给硅制造商就可以投入生产了。

这对于交易的双方是“双赢”的，因为硅制造商保证生产线顺畅地生产，而系统公司集中精力去设计其产品有别于其他产品的特色功能。实际上，系统公司也许会通过对该平台进行不同的编程，从而得到一个全系列产品。低成本产品中的芯片可能不具备电源监控装置，而高端产品中的芯片可能通过编程实现了无数功能——介于高端和低端之间的其他产品的功能都只具备其高端产品的一部分功能。系列产品的基础在于它们采用的是同一个硅设计平台。

然而，此方案背后的问题是：如何才能让那些对平台进行定制的设计人员领会平台特性呢？它是一个硅设计平台，而系统级设计人员可能还没有很好地理解版图(layout)、门级描述等传统的硅设计表示方法。类似地，可能也没有真正理解采用 VHDL 或 Verilog 等语言在寄存器传输级描述的设计对象。实际上，即使采用英语这样的自然语言进行文字描述，其作用也很有限——因为篇幅太长。

很明显，所需要的是“可执行规约”。可执行规约是设计平台中的仿真模型，使用者可以利用规约自身的测试平台和(或)由平台供应商所提供的测试平台来执行该规约。观察设计平台各部分对不同输入激励的响应行为，查看设计平台各部分的源代码，这样系统工程师团队就能获得所定制的设计平台的各方面的知识。

上述做法是可行的，但是，只有可执行规约执行得足够快，才能够有效地观察其行为。而且，系统设计团队需要阅读该规约，所以编写的规约应当能够让人们明确理解，即应当在适当的抽象层次上编写。因此，在网络设计对象中，最好能够在报文传输的层次描述电路块的通信，而不是在底层的信号变化层次进行描述。

以上两点表明需要建立高速的可执行模型。以较高的抽象层次表达较低层次的电路结构的可执行模型，必然导致以未定时方式的 C ++ 描述来表示设计平台。如上所述，采用 C ++ 编写的未定时模型执行速度快，而且 C ++ 语言适合于定义那些便于设计人员理解的抽象数据类型。

此外，采用 C ++ 作为可执行规约有一个额外的优点，即它可以很自然地与系统设计团队为定制设计平台而编写的那些 C ++ 软件模型衔接。这样就能建立更大的可执行规约(即在最初的硅设计平台中再加上所定制的软件)，系统公司和硅设计平台供应商都可用它来理解定制平台的行为。事实上，有些 EDA 厂商正在着手开发将 C ++ 综合成 FPGA的工具，即使那些在硬件方面用于定制设计平台的模型，也可以轻松地与设计平台最初设计规约相衔接。

如上所述，硅设计平台的生产商和用户都采用包括 SystemC 建模在内的设计方式当然是合乎情理的。

为何采用 SystemC?

如果能接受以上观点，那么就能够理解目前对于采用 C ++ 进行建模在内的各种设计方法的需求正在日益增长。但是，到目前为止，还没有开始讨论开放式 SystemC 联盟(Open SystemC Initiative，OSCI)所定义的 SystemC。C ++ 是开放的和相当容易扩展的语言。任何公司都可以根据其劳动力水平定义采用 C ++ 的实际方式，以满足上面所说的各种需求。实际上，确实有很多公司是这么做的。

由 OSCI 定义 SystemC 有一定的合理性。如果每个公司(甚至是公司中的各个设计小组)都自行定义使用 C ++ 进行设计的方法，那么将会极大地降低公司之间彼此轻松协作的能力。公司共享设计对象时，拿到设计对象的那一方必须先理解设计方的 C ++“方

言”。显然，这将妨碍硅设计平台供应商和用户之间进行设计对象的交易。

此外，这还削弱了第三方知识产权(IP)供应商(比如为各种公司提供处理器模型的研发商)的能力。必须解决多种 C ++ 方言并存的问题，这就要求完整系统的研发商要么自己开发所有子系统的模型，要么想办法使第三方模型能适应自己的 C ++ 标准。在设计速度是关键的市场状况下，这两种选择看起来都毫无吸引力。

这样就剩下两条出路，要么让使用 C ++ 的事实模型在应用过程中慢慢浮现出来，要么开发出包括硅供应商、系统公司、IP 研发商和 EDA 工具供应商在内的所有 C ++ 用户都认同的用法模型。前一种做法并非不存在任何问题：Verilog 能够成为 HDL 的事实标准，倚仗于其发明者 Cadence Design Systems 公司发展得很成功。C ++ 领域的角逐也是类似的情形。另一方面，VHDL 能作为工业开发的标准出现，恰恰是因为人们意识到需要一种整个工业界都可以统一使用的通用语言。正因为等待恰当的事实标准在实践过程中浮现出来过于冒险，OSCI 决定沿着 VHDL 模型的技术路线进行研究。

关键在于实在没有时间等待可能的事实标准浮现出来。电子工业的需求表明，应尽快定义使用 C ++ 的标准方法，该标准应尽量满足众多潜在用户的需要。基于这种思路，OSCI 成立了，不断有代表半导体公司、系统公司、IP 供应商和 EDA 公司的成员加入，它们彼此健康的协作推动 OSCI 不断茁壮成长。

为何选用本书？

从 Bhasker 在明尼苏达大学获得博士学位，到进入我在 Honeywell 实验室的研究小组以来，我认识他差不多有 20 年了。甚至在那个时候，他就已经具备了很强的洞悉技术难题的能力，并且善于向更年轻的团队成员把难题解释清楚。这些年来，他解释难于理解的技术材料的能力还在不断增强，与此同时，他出版了许多有关 VHDL、Verilog 和逻辑综合等方面的教材，这些教材在全世界的大学中得到了广泛采用。这些教科书采用严密但可理解的方式展示电子设计领域中的各种“原料”，帮助培养了新一代工科学生。

“A SystemC™ Primer”这本独特的教材，一定会像作者以往的那些书一样，在 SystemC 领域产生较大的影响。本书并不是面向高级研究人员或语言学专家，而是一本 SystemC 的入门教材，它通过引用常见的数字设计概念，循序渐进地引导读者学习 SystemC 的各种复杂特性。他的一贯通俗易懂的著述风格便于读者迅速理解 SystemC 的基础内容，帮助读者开始采用 SystemC 作为设计语言，并且进一步研究更高级的语言要素。

最后，SystemC 只有像现在的 VHDL 和 Verilog 那样在设计领域得到普遍使用，才能成为真正有用的标准。本书的出版对于这一天的早日到来无疑是很好的催化剂。

Stanley J. Krolikoski

OSCI 主席

2002 年 3 月于加利福尼亚州圣何塞

前　言

为什么你会选择本书？我认为你是要学习 SystemC。那么，你选对了！来吧，本书将带你进入 SystemC 领域。只要读完第 2 章和附录 A，你很快就能开始编写 SystemC 模型并对它们进行仿真了。

SystemC 既是系统级描述语言，又是硬件描述语言。它是可以对硬件和软件系统进行建模的一种语言。说它是硬件描述语言，是因为它可以在寄存器传输级(Register Transfer Level，RTL)进行建模；说它是系统级规约语言，是因为它可以在算法级进行建模。还可以采用 SystemC 对整个系统进行建模，并像软件编程那样描述系统的行为。尽管它能描述门级网表，但其本意并非如此使用，因为使用和建立门级网表模型，既繁琐又低效。

SystemC 建立在 C++ 语言基础之上。SystemC 将并发、定时事件和数据类型等重要概念引入 C++，从而扩展了 C++ 的功能以对硬件进行建模描述。SystemC 的这一能力是通过类库提供的，类库提供了利用硬件元素、并发和反应行为对系统体系结构进行建模的强大新机制。这些机制都只是建立在 C++ 编程语言的类结构基础之上。SystemC 还提供了仿真内核，可用来对设计对象或系统的可执行规约进行仿真。

本书介绍的是 SystemC 2.0 标准。该标准由开放式 SystemC 联盟(Open SystemC Initiative，OSCI)语言工作组建立，主旨是在不久的将来使其成为一项 IEEE 标准。关于 SystemC 和 OSCI 更多的信息，可以访问网站 http://www.systemc.org。从该网站可以下载 SystemC 2.0 软件包，软件包中的功能规约和用户指南全面地介绍了 SystemC 2.0 标准。

1999 年，很多家公司精诚合作成立了 OSCI。同年 9 月，第一个 SystemC 版本——SystemC 0.9 版以开放源代码和免费获取的方式发布了。2000 年 3 月发布了 SystemC 1.0 版。该版本还仅限于行为级和寄存器传输级建模，而缺少很多用于系统级建模的语言要素。

2001 年 10 月发行的 SystemC 2.0 版本具有许多系统级建模要素。新增加的要素包括信道、接口和事件。本书是根据 SystemC 2.0 版本编写的。

本书主要介绍 SystemC 的硬件建模方面，即 SystemC 的 RTL 可综合子集。采用该子集编写的模型可以综合成逻辑门电路，进而进入模型的硬件实现阶段。集中介绍硬件建模方面的原因有三点：

• 首先，当今熟知 VHDL(即 VHSIC 硬件描述语言，IEEE 1076 标准)和 Verilog HDL 语言(IEEE 1364 标准)的硬件设计人员可能愿意去了解和学习 SystemC。因为抽象层次正从寄存器传输级转移到更高层次上，所以目前进行 RTL 建模的设计人员将不得不学习系统级建模。本书将以十分自然的方式向设计人员介绍 SystemC，以帮助他们跨越这种抽象层次的鸿沟。

• 其次，编写高层次算法模型的系统设计人员需要理解 RTL 综合，这样他们就能对模型反复细化至寄存器传输级，从而将模型综合成门电路。

• 第三，模块设计人员能够采用 RTL 可综合子集来开发知识产权（Intellectual Property，IP）模型，这样就能实现可重用和可综合的 IP 功能块。

本书专门面向那些希望学习和了解 SystemC 的设计工程师和系统工程师。

本书适合于初学者，因此并没有涉及 SystemC 的某些高级论题。比如，没有讨论在 SystemC 中定义的主-从通信库，因为设计方法的专用库超出了本书的讨论范围。

本书可以作为高校的体系结构、数字设计和系统设计等课程的教材。可采用本书来介绍 SystemC 概念。采用 SystemC 作为课程的一部分，一个很大的优点在于 SystemC 仿真器和所有 SystemC 类库都是开放源代码的，每个人都可以免费获取和使用。因为不需要任何资金投入，所以吸引了很多大学教授讲授和理解这种新的系统级设计语言。开放源代码这一事实，允许有热情的大学生通过直接修改源代码，从语言要素和优化等方面扩展 SystemC 的能力。

本书是基于笔者个人的理解来介绍 SystemC 的 RTL 可综合子集，该子集与 IEEE 标准中的 Verilog 和 VHDL 的 RTL 可综合子集（IEEE 1076.6 标准和标准草案 IEEE P1364.1）吻合，笔者作为主席领导了那两项标准化进程。目前的 SystemC 综合工具是否支持该 SystemC 可综合子集还不是很肯定。关于综合工具所支持的特定功能要素，读者应查阅对应的工具文档。

阅读本书和理解 SystemC 需要哪些背景知识呢？首先必须了解 C++ 语言的基础知识，这是不可或缺的。其次还应该有逻辑设计的背景。如果已经了解 VHDL 和 Verilog HDL 这两种广为流传的硬件描述语言中的任何一种，那么利用本书学习 SystemC 将会如鱼得水（本书并未刻意将 VHDL 和 Verilog HDL 模型与 SystemC 模型等同起来）。如果对 C++ 语言相当了解，就会发现采用 SystemC 编写高级系统级模型异常轻松——同样也能够理解 SystemC 的本质了。然而，了解 VHDL 或 Verilog HDL 并不是阅读本书的必要前提。若想驾驭 SystemC 的全部功能，需首先学会使用 C++ 的高级功能。然而，如果只进行硬件建模和理解 RTL 可综合子集，那么只需要了解 C++ 编程语言的基础知识。Addison-Wesley 于 1998 年出版的由 Stanley B. Lippman 和 Josee Lajoie 撰写的“C++ Primer，Third Edition”就是一本学习 C++ 语言的好书。

本书给出的所有模型都已经在 Solaris 系统上通过测试和仿真。所有图中综合出的逻辑，是先手工将 SystemC 模型转化为等价的 Verilog 描述，然后用 Ambit BuildGates 综合工具综合 Verilog 模型而得到的。

SystemC 的前景

SystemC 这门语言还在发展变化之中——它仍将继续以很快的速度发展，直至成为 IEEE 标准。即使到了那个时候，我希望它作为有用的标准仍将继续发展下去。开发

SystemC 2.X、3.0 和 4.0 发行版本的计划已经制订完毕。2.X 版本将包括分叉/汇合(fork/join)、行为层次的中断/异常中止、性能模型支持以及定时规约支持。3.X 版本将支持抽象 RTOS 建模和调度程序建模。4.X 版本将支持数模混合信号系统建模。

致谢

十分感谢以下人员对本书作出的贡献,他们不仅审阅了初稿,还提出了很多建设性的意见和崭新的思路,从而使本书有了显著的改进。

Mike Baird
Abhijit Ghosh
Thorsten Groetker
Jeff Hantgen
Kurt Heinz
Sven Heithecker
Xiaoyan Huang
Martin Janssen
M. N. V. Satya Kiran
David Long
Grant Martin
Dale Mehl
Sanjiv Narayan
Smail Niar
Bernhard Niemann
Stuart Swan
Kartik Talsania
Punitha Thandapani
Yves Vanderperren
Jean Witinski

由衷感谢!

最后,还要说的是,如果没有我的妻子 Geetha 以及三个孩子 Arvind、Vinay 和 Vishnu 的支持,就不可能有本书的问世。

欢迎对本书提出任何指正和建议,请发电子邮件至 jbhasker@cadence.com 或通过我的出版商与我联系。

J. Bhasker
2002 年 4 月

目　录