HZ BOOKS

电子与电气工程丛书

PRENTICE HALL PTR

硬件设计验证

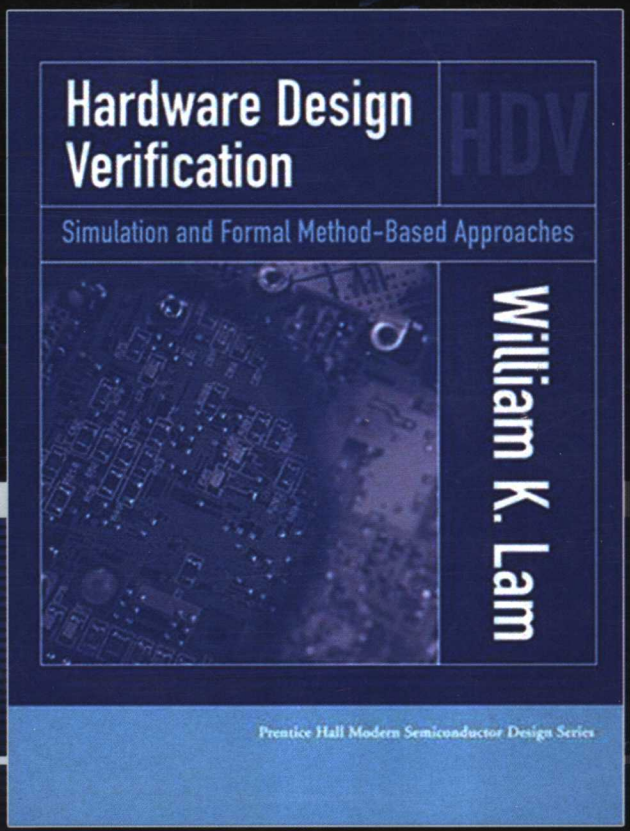
基于模拟与形式的方法

Hardware Design Verification

Simulation and Formal Method-Based Approaches

（美）William K. Lam 著

王维维 译



机械工业出版社
China Machine Press

电子与电气工程丛书

硬件设计验证

基于模拟与形式的方法

Hardware Design Verification:

Simulation and Formal Method-Based Approaches

（美）William K. Lam 著

王维维 译

机械工业出版社
China Machine Press

本书全面介绍硬件系统设计验证的技术和方法，主要涉及基于模拟和形式验证的方法，内容涵盖静态检验、模拟器体系结构、测试基准设计、模拟规划与策略、调试进程与验证周期、形式验证背景知识、判定图与SAT问题、符号计算与模型检验。书中汇集大量设计验证的基本概念与技术，内容深入浅出，叙述详尽，既讨论一般的测试原则又展示具体的实践方法，包含作者多年实践经验，实用性强。每章最后还配有各类习题，读者可用来巩固所学的知识。

本书可作为高等院校电子科学与技术、计算机科学与技术等专业高年级本科生或低年级研究生教材，也可供相关专业工程师参考。

本书版权登记号：图字：01-2005-1853

图书在版编目（CIP）数据

硬件设计验证：基于模拟与形式的方法/（美）兰姆（Lam，W.K.）著；王维维译．
—北京：机械工业出版社，2007.1

（电子与电气工程丛书）

书名原文：Hardware Design Verification: Simulation and Formal Method-Based Approaches

ISBN 7-111-19502-7

Ⅰ.硬…　Ⅱ.①兰…　②王…　Ⅲ.硬件—设计—验证　Ⅳ.TP303

中国版本图书馆CIP数据核字（2006）第073516号

机械工业出版社（北京市西城区百万庄大街22号　邮政编码　100037）

责任编辑：吴　怡

北京诚信伟业印刷有限公司印刷·新华书店北京发行所发行

2007年1月第1版第1次印刷

184mm×260mm·23.5印张

定价：45.00元

凡购本书，如有倒页、脱页、缺页，由本社发行部调换

本社购书热线：（010）68326294

译 者 序

过去几十年，微电子技术高速发展所取得的巨大成就和人们对信息技术的大量需求，使得现代集成电路系统的规模和复杂程度日趋提高。这必将导致对未来集成电路系统的设计和制造提出更高的要求。

一方面，要在包含数以亿计的晶体管的电路中避免出现设计错误，这是一项非常艰巨的任务。据报道，1994 年 Intel 公司生产的 Pentium 芯片中的浮点除法运算错误是在其大量进入商用后才被一位用户发现的，虽然这种错误发生的概率为几亿分之一，但这个设计错误还是给 Intel 公司造成 4.75 亿美元的经济损失和难以估量的形象损失。另一方面，从集成电路设计角度看，系统设计是否满足规范的需求，以及在设计的各个阶段的等价性检验等都对设计验证提出很高的要求。

William K. Lam 博士所著的这本书正是全面而系统地阐述如何应对上述两方面挑战的入门读物。它汇集并整理大量的数字硬件设计验证的技术和方法，提出许多设计验证的原则及实际算法；更值得一提的是，本书叙述深入浅出，为想进入形式验证这一具有“高深数学门槛”的领域的读者提供了一种很好的学习方法。

本书的主要内容围绕着对硬件设计进行验证的两大方法（基于模拟和形式验证的方法）展开。第 1～6 章论述基于模拟的验证。在引入设计验证的一些基本概念以后，作者提出硬件设计代码的编码风格准则及排除静态错误的方法；第 3 章介绍模拟器的基本体系结构与操作；第 4 章则对测试基准的组成与设计进行全面阐述；第 5、6 章分别讨论进行模拟时的测试规划和测试构想，以及研究错误查找周期中不同环节的方法和过程。第7～9章对形式验证这一当前技术热点的若干主题进行循序渐进的讨论。第 7 章为不熟悉有关形式验证所需的数学预备知识的读者提供很好的入门材料；第 8 章则花费大量篇幅讨论二叉判定图（BDD）的构造、简化及相应的检验技术；第 9 章详细讨论模型检验、符号模拟技术，并对所用到的时态逻辑知识点作了精心的安排，使读者能更好地理解这些非常抽象的符号。当然，如果读者事先具备命题逻辑和一阶逻辑的背景知识则更好。

本书作为为数不多的硬件设计验证的系统论著，有其鲜明的特点：对内容的叙述非常详尽。比如，既讨论一般的原则也展示具体的实践方法，有些内容还提供伪代码形式的算法，读者只需简单地改写为具体的程序设计语言，即可上机调试。此外，本书作者 William K. Lam 博士是 Sun 公司的资深经理及高级工程主管。他在 Sun 和惠普公司任职期间，致力于设计验证、低功耗设计及分析、逻辑综合及自动测试生成（ATPG）等工作。正因为这样，本书不是枯燥无味的从理论到理论的陈述，而是包含大量实例，从而更为实用。

正如作者在前言中所说，本书的读者可以是在校的学生，也可以是企业的验证工程师；他们都可以从本书中获得“一站式”的现代硬件设计验证指南。

据译者所知，目前在国内还没有关于硬件设计验证的论著出版。因此，引入这方面的国

外论著是一件非常有意义的工作。本书的翻译遵从忠实于原著的原则，但对于原著中出现的一些明显错误做了修正。另外，还根据原书作者提供的勘误表改正了原文的一些错误。由于本书涉及大量电子科学、计算机科学及数学的一些知识，译文中对一些汉语中没有对应现成译名的名词（尤其是集成电路领域的术语）都给出原文，供读者对照；另外，译者还专门编写了原书中出现的术语缩写词英汉对照表。

在翻译本书的过程中，得到家人的理解和支持。机械工业出版社华章分社的编辑做了大量工作。此外，还得到我的同事俞水根先生的协助。在此，一并对他们表示衷心感谢。

虽然尽了最大努力，但由于水平有限，译文中难免会有错误和不当之处，敬请读者不吝赐教。我的邮件地址为：wangww@vlsi.zju.edu.cn。

王维维

2006 年 3 月于浙江杭州

前　言

对于一个成功的设计项目而言，有两组人员是不可或缺的：一组是设计人员，另一组是验证人员。设计人员通常受过学校的正规教育。许多学院开设关于逻辑设计的详细课程，其范围从数字设计导论一直到高等计算机体系结构。与此相反，大多数的验证工程师是从工作中学到相关技术的。虽然许多学校开始讲授验证的课程，但是很少有高等院校专门培养验证工程师。事实上，大多数验证工程师出身为设计师但逐渐转向设计验证。与设计使用的技术与方法不同，验证所需的知识广博而又组织松散，并且是通过日常的动手实践而获得。此外，验证的范围正在快速扩展：验证技术的全景图每 6 个月就有新技术、标准和工具出现。然而，经过时间检验并且证明是验证的奠基石的原理和技术并没有改变。

本书汇集并系统介绍业界常见的使用广泛的数字设计验证技术和方法，主要讨论数字逻辑设计和验证，但并不包括混合信号或射频元件电路的验证。本书的目标之一是将大量的验证知识传授给大学学生和工程师，使他们能够更好地为就业作准备，并加速他们的学习进度。写一本 10 分钟速成的验证工程师手册，列出可以在工作中立即可用且详细的实用诀窍是非常诱人的，但是这种救急的窍门往往很快就过时。另外一方面，只是讲解理论对于工程师来说也是非常不现实的。因此，本书在两者之间取得了平衡，不仅讨论常用的实践方法同时也提出验证的原理。作者的观点是，只有通过理解原理才能真正掌握实践的要旨，并在使用时体现出创造性。

致读者

本书的读者对象包括三、四年级的本科生和低年级的研究生。本书假设这些读者对一门硬件描述语言非常了解，最好是 Verilog。因为 Verilog 语言在大多数教科书中作为描述的目的而使用。如果另外还对逻辑设计有初步了解则更好。本书是设计验证的入门教程。通过学习，学生们将会学到在基于模拟验证中使用的主要思想、工具和方法，还将学会形式验证的内在原理。本书所提供的材料是经业界测试及广泛使用的。在每章的末尾安排习题，可帮助复习章内所覆盖的知识。对于想深入了解某些专题的读者，请参考本书参考书目中所列出的资源。

本书的读者对象还包括拥有一些验证经验的验证工程师，这些读者也许需要系统地了解验证的不同领域并理解形式验证的基本原理。这些读者也需要掌握基本的系统设计知识和一门硬件描述语言。本书的第一部分详细描述基于模拟的验证方法，并可以作为专业验证师的复习材料或入门读物。对于大多数实践工程师而言，形式验证工具似乎是一种需要深奥的数学知识才能理解的魔术。本书对这些工具和形式验证的工作原理作了解释。在深入讨论形式验证之前，第 7 章介绍了为充分理解验证算法所需要的基本数学知识和计算机算法。

与任何技术一样，没有什么东西能取代动手的经验。这里鼓励读者通过运行本书中的一

些例子和习题，或通过设计一个能使用所论述的工具的项目来认识验证工具。诸如 Verilog 模拟器、测试基准开发工具以及波形观察软件等自由 CAD 软件可以在 www. verilog. net/free. html 上找到。

致教师

本书由两部分组成。第一部分论述传统的验证策略，即基于模拟的验证；第二部分论述在学术界建立完善、并在业界已得到证实的形式验证的各个方面。这两个部分是自成一体的，但可以根据需要而独立讲授。

第一部分描述许多验证工具。由于这些工具的专长因厂商的不同而不同，并且随着时间的推移也不同，所以本书只使用伪通用命令。为了强化学习这些工具，建议把业界的工具作为验证实验的一部分（比如，模拟器、波形观察程序、覆盖工具、错误跟踪系统和修改控制系统）。同样，为巩固形式验证的知识，应当在实验环节中使用商用的形式验证工具，比如硬件描述语言的 linter 程序、模型检验程序和等价检验程序。

本书有配套的教师手册，其内容含有每章末尾所有习题的解答。需要的教师可以填写书后面的“教学支持说明”，然后与出版社联系。

本书的组织

本书作为设计验证的入门书籍，仍需要读者对基本的 Verilog 结构有所了解。虽然本书把 Verilog 语言作为硬件描述语言，但还是尽量给出独立于 Verilog 的思想。在不可避免需要使用 Verilog 的地方使用最简单的 Verilog 结构，让不熟悉该语言的读者掌握其主要思想。

正如前所述，本书由两部分组成。第一部分讲解基于模拟的验证，第二部分讨论形式验证。基于模拟的验证是迄今为止应用最为广泛的方法，是对所有验证工程师的一项最低要求。形式验证则相对是一项新技术，是基于模拟验证的补充。作者相信，要最好地使用一项技术，人们必须先对该项技术内部的原理有深入的理解。因此，本书不仅在用户层次上讨论验证工具的操作（这个内容比较适于写成用户手册），还用很多篇幅讲解模拟和形式技术的基本原理。

第一部分模拟技术由第 2～6 章组成。这些章节的安排顺序与通常模拟验证过程的操作顺序相似。我们从第 2 章开始，对静态错误进行检测。这些错误无须输入向量即可探测出来，并且必须在更广泛的模拟开始之前消除。在第 3 章，我们研究模拟器的基本体系结构。首先提出事件驱动和基于周期的模拟算法，随后讨论模拟器的操作以及协同模拟、对设计的描述、常见模拟器选项和用户界面的应用。

在人们能开始模拟之前，必须建立一个用于系统设计的测试基准。第 4 章讨论测试基准设计、初始化、激励生成、时钟生成网络、错误注入、结果评估及测试配置。当某个系统设计不存在静态错误、并被嵌入一个测试基准以后，模拟即可开始。但是，应当如何模拟系统设计呢？第 5 章提出模拟什么以及如何度量模拟的质量。我们将考察测试规划设计、测试规划中测试项目的生成、输出响应评估、断言（特别是 SystemVerilog 断言）和验证覆盖。

在对电路进行模拟并发现错误之后，下一步就是对在模拟过程中发现的问题进行调试。第 6 章提出广泛使用的调试技术，包括范围压缩、检测指示、错误跟踪、踪迹转储及正向和反向调试。此外，第 6 章还考察系统设计的 4 种基本视图：RTL 代码视图、图式视图、波

形视图和有限状态机视图，然后考查错误修复之后的构想（scenario）。所讨论的工具和方法包括修改控制系统、回归测试机制及流片（tape-out）标准。

本书的第二部分形式验证由第 7～9 章组成。本书第一部分的若干章节可以与第二部分一起学习。比如，第 2、4、5、6 章也适用于形式验证。

理解形式验证的关键在于理解其内在的理论。第 7 章提供后续章节所需的基本数学背景。这些材料涵盖布尔函数及表示、对称布尔函数、有限状态机和等价算法，以及诸如深度优先搜索和宽度优先搜索、强连通分量的图算法。

第 8 章给出判定图的概况，并重点论述二叉判定图。然后学习 SAT（可满足性）问题作为判定图的另一种替代方法。该章最后考察判定图和 SAT 问题在等价性检验和符号模拟中的应用。

第 9 章对符号模型检验进行深入研究。首先给出自动机与计算树逻辑作为模拟有公平性约束的时态行为的方法，然后讨论在一个时态规范对比下的模型检验。根据模型检验算法，给出有效的符号模型检验算法，其中的图运算通过布尔函数计算完成。接着，再次阐述对不存在一一对应的一般电路的等价性检验问题。最后，研究更有效地处理符号计算的算法。

勘误

如此篇幅的书籍必然包括错误和需改进之处。请将错误报告和评论发送至作者的电子邮件：williamlamcmail@gmail.com。

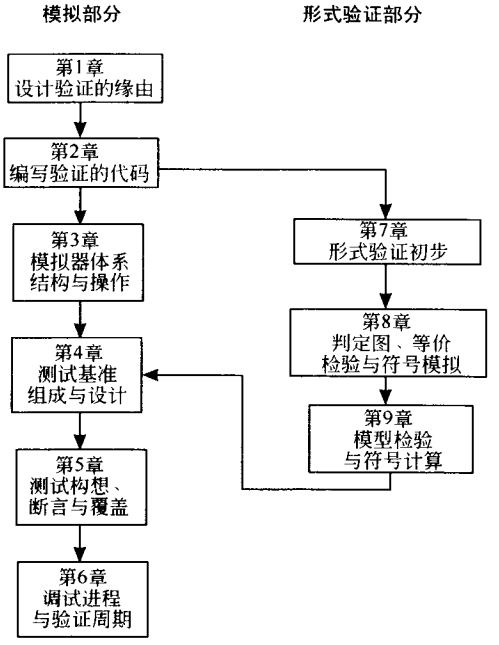


图 P-1　本书各章概览

致　　谢

首先和最重要的是感谢我的妻子 Serene，感谢她的耐心、理解和鼓励。她的爱与支持支撑着我度过这一段漫长且孤独的旅程。我也要感谢我的母亲 Grace 和父亲 James Lam，感谢他们一贯的指导与支持。

感谢所有的书评者花费宝贵的时间阅读本书的手稿，并指出错误及改进的方向。特别感谢 Rajeev Alur、K. C. Chen、Thomas Dillinger、Manoj Gandhi、Yu-Chin Hsu、Sunil Joshi、Shrenik Mehta、Vigyan Singhal、Ed Liu、Paul Tobin 及 John Zook 对本书真知灼见的评论、建议和勉励。Verilog 的自由软件的站点是 Thomas Dillinger 建议的。此外，非常感激 Bernard Goodwin 的热情与鼓励，感谢 Prentice Hall 出版公司编辑的敬业精神。

目　录

第 1 章　设计验证的缘由

本章要点

- 什么是设计验证
- 验证的基本原理
- 验证方法学
- 基于模拟的验证与形式验证的比较
- 形式验证的局限性
- Verilog 语言调度和执行语义简介

从项目团队的组成中可以看到设计验证的重要性。通常，在一个典型的项目团队中，设计工程师和验证工程师的人数相当。有时，验证工程师超出设计工程师的两倍。这些都源自以下的事实：要对一个设计进行验证，验证者不仅首先要像了解系统设计一样了解设计规范，更重要的是要得到基于该规范的一个不同的设计方法。应当强调的是，验证工程师所采用的方法不同于设计工程师的方法。假如验证工程师遵循与设计工程师相同的设计方式，那么如果有错，两者将犯同样的错误，也不能得到预期的验证结果。

从项目的开发周期中能够了解设计验证的困难性。统计数据表明，设计验证约占项目开发周期的 70%。设计工程师通常是依据设计规范的代表事例进行设计，但验证工程师则必须对设计的所有事例进行验证，而有可能存在无限的事例（或者看上去是无限的）。对于一块中等复杂度的芯片而言，即使在验证上投入了大量的资源，但在投产收益之前通常要经历多次流片（tape-out）。

我们不能低估全面设计验证的影响。一块有缺陷的芯片不仅当其返工（respin）时增加设计费用，而且会拖延其上市时间，冲击产品收益，缩小市场份额，使公司陷入穷于应付的困境。因此，人们在能够设计完美的芯片，或者芯片制造成本降低且生产周期缩短之前，设计验证是必不可少的。

1.1　什么是设计验证

设计过程是将一组设计规范转换为规范实现的过程。在规范层次上，规范指明了设计所执行的功能但并不涉及如何去执行。规范的实现方案给出如何提供其功能的细节。规范与其实现方案均为其功能描述的一种形式，但它们的具体性和抽象性是处于不同层次的。描述的抽象层次越高，给出的细节就越少；因而规范的抽象层次要比实现方案要高。在表示设计过程抽象性的图谱中，我们看到抽象性不断降低：即功能规范，算法描述，寄存器传输级（RTL），门级网表，晶体管级网表，布图（见图 1-1）。沿着图谱的任意一层次，都可以给出一个更低层次的多种形式的描述。例如，在门级，就有无限种类的电路可以实现相同的 RTL 描述。当向图的下层移动时，在保持高层次的描述不变的情况下，原先较少的抽象描

述要添加更多的细节。把较抽象的描述转变为较具体的描述过程称为细化。所以，设计的过程就是一个对一组规范进行细化，并产生不同层次的具体实现方案。

设计验证是一个与设计相反的过程，它从一个实现方案开始，并且确认该实现方案是否满足其设计规范。于是，在设计的每一步骤都有验证步骤与之对应。例如，把功能规范转换为算法实现的设计步骤要求有验证的步骤，以确保其算法能执行规范中的功能。与此类似，物理设计层从门级网表生成的布图也需要验证以确保布图与门级网表对应。总之，设计验证包括许多方面，比如，功能验证、时序验证、布图验证及电学验证等。本书只限于研究功能验证，并称之为设计验证。图 1-2 展示了设计过程与验证过程之间的关系。

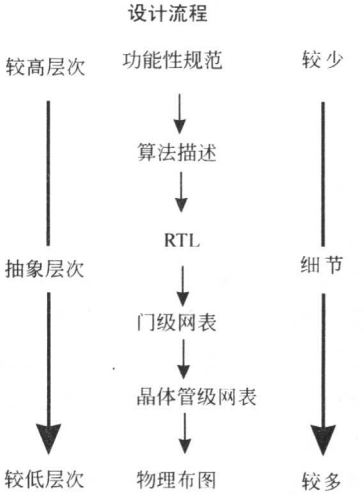


图 1-1 设计的抽象层次图

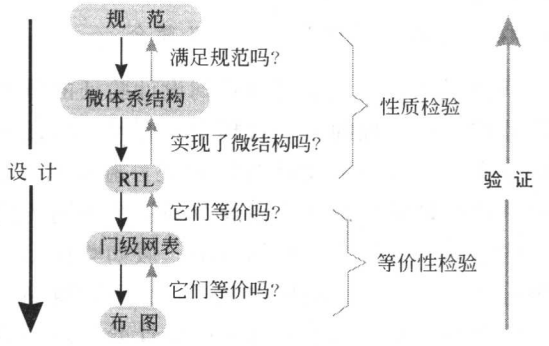


图 1-2 设计和验证的关系

从更为细微的角度看，设计验证还可以进一步分为两种类型。第一种类型是证明设计的两个版本是否在功能上等价。我们把此类验证称为等价性检验。等价性检验的一种常见方案是在相同抽象层次上比较电路的两个版本。例如，把某个电路门级网表的前置扫描与其后置扫描版本进行对比，以确保在正常的操作模式下两者是等价的。

然而，设计的两个版本随着抽象的层次不同而有所差异。例如，一个版本位于设计的规范层次而另外一个版本来自门级网表层次。当两个版本因抽象层次而存在实质性的差异时，两者的功能并不等价，这是因为较低层次的实现方案可能包含了较高层次所允许、但并未指定的许多细节。例如，在原始的规范中并不包括某个实现方案的时间约束。在这种情形下，我们采用的是验证该实现方案满足规范，而不验证两个版本的功能等价性。请注意，等价性检验是一种双向的验证，而这种验证是单向验证，这是由于某个规范在实现时可能并不满足某个未说明的性质。这种类型的验证称为实现验证、性质检验或者模型检验。用性质检验的术语来说，规范是实现方案必须满足的性质。用模型检验的术语来说，实现方案或者设计是电路的一个模型，而规范是性质。因而，模型检验就是针对规范性质的模型的检验。

1.2 验证的基本原理

设计错误主要有两种。第一种错误是在实现的过程中引入的，它是实现错误而不是规范

错误。对设计功能的人为错误解释就是属于这方面的例子。为了防范此类错误的发生，我们可以利用软件的程序直接从设计规范中综合出一个实现方案。虽然这种方法能消除大多数人为错误，但是软件程序中的缺陷仍然可能导致错误的产生；或者，由于软件的使用不当而引发错误。此外，这种综合方法在实践中具有很大的局限性，原因有两点。第一，大多数的规范是以会话语言的随意形式描述的（例如英语），而不是以精确的数学语言（例如 Verilog 或者C++)的形式描述的。我们知道，从一个不严格的语言自动综合出的结果是不可行的。事实上，本书写作时尚没有一种既能满足功能描述又能满足时间描述要求的高级形式化语言。原因之一是由于高层次的功能性需求并不适应于时间需求，而在实现层次上时间需求是更为直观的。所以，当诸如像延时和功耗之类的时间需求与高层次的功能性规范混合在一起时，这些规范明显不准确，使得人们把时间规范降低到较低的抽象层次。第二，即使用精确的数学语言来描述规范，也很少有软件程序能够生成满足所有需求的实现方案。通常，软件程序能从一组功能规范中综合需求，但却不能满足时间需求。

另外一种广泛使用的方法是利用冗余性发现这一类型的错误。即，使用不同的方法两次或者多次实现同样的规范，并加以比较。从理论上说，规范实现的次数和方法越多，验证所产生的可信度就越高。但实际上，我们很少会使用两种以上的方法去实现，这是因为在每一种可选的验证方法中都可能引入更多的错误；此外，成本和时间也是不可逾越的障碍。

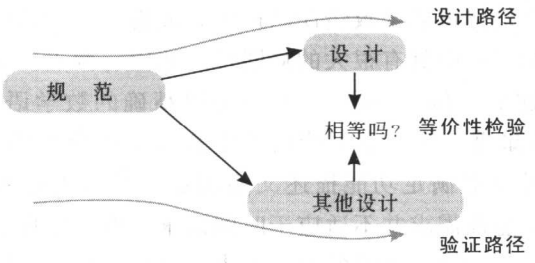
设计的过程可看做是把一组规范转换为一个实现方案的路径。验证的基本原理包括两个步骤。在第一个步骤中，把一个规范转换为一个实现方案。我们把这一步骤称为*验证转换*。在第二个步骤中，把验证所产生的结果与设计所产生的结果进行对比从而检出错误。请见图 1-3a。通常，验证转换的结果发生在验证工程师的头脑中，其形式是从规范推导中得到的性质。例如，验证工程师可以根据规范和计算获得模拟输入向量的预期结果，这个结果就是另一种实现方案。

显然，如果验证工程师经历的步骤与设计工程师完全相同，那么两者将很可能得到相同的结论，避免或者犯相同的错误。因而，设计和验证所经历的轨迹差别越大，验证的可信度就越高。要提高验证工程师工作的可信度，一种办法是使用与设计不同的语言把规范转换为实现方案模型。对应于设计语言，这种语言被称为*验证语言*。Vera、C/C++以及 e 等语言都属于验证语言。一种可行的验证策略是验证模型使用 C/C++语言，而设计模型使用 Verilog/VHSIC 硬件描述语言（VHDL)。

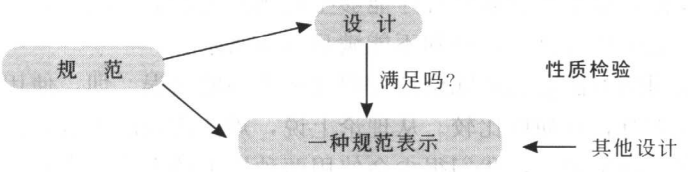
在验证的第二个步骤中，我们对比两种形式的实现方案，方法是把两种形式的实现方案用同一种中间形式表达出来，这样可以有效地进行等价性检验。在某些情况下，这种对比机制可能会很复杂。例如，要比较两个网络中到达的数据包是否出错。此时，常见的形式是将到达的数据包以预先定义的形式排序。另外一种比较机制的例子是确定在晶体管级的电路和 RTL 实现级之间的等价性。常见的共同中间表达形式是二叉判定图（binary decision diagram)。

这里，我们看到典型的基于模拟验证的模型是符合验证原理的。基于模拟验证的模型包含 4 个组成部分：电路、测试模式、参考输出和比较机制。电路在测试模式下被模拟，并把其测试结果与参考输出加以对比。由设计路径产生的实现方案就是电路，由验证路径产生的实现方案则是测试模式及参考输出。之所以考虑把测试模式与参考输出作为验证路径的实现结果，是因为在确定从测试模式产生的参考输出的过程中，验证工程师把基于规范的测试模式转换为参考输出，而这个过程正是一个实现的过程。最后，比较机构对模拟结果进行抽样

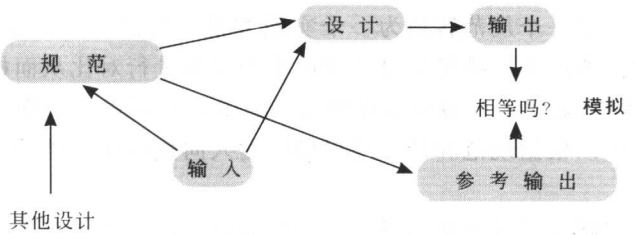
以确定它们与参考输出是否相同。基于模拟的验证的原理见图 1-3c。



a) 利用冗余性验证的基本方法



b) 一个来自模型检验的基本方法的变体



c) 模拟方法对基于冗余的验证的影响

图 1-3 设计验证的基本原理

基于冗余性的验证是一把双刃剑。一方面，它可以发现两种方法之间的不一致性；另一方面，它也可以在两种方法之间引起不相容的差异和一些常见的验证错误。例如，利用C/C++模型对 Verilog 的设计进行验证可能会迫使验证工程师去解决两种语言之间根本的差异，而这种差异本来是可以避免的。由于两种语言的不同，存在着一种语言能准确模拟而同时另一种语言却不能准确模拟的区域。例如对 C/C++模型而言，模拟时间和并行性都是有缺陷的。因为设计代码易受错误影响，验证代码同样也容易出现错误，因此，验证工程师必须像排除验证错误一样去排除设计错误。因此，如果不小心因冗余策略最终使得工程师所要排除的错误比设计中的错误还要多（设计错误加上验证错误），将会引起巨额的验证费用开销。

正如前面所述，第一种错误是在实现过程中引入的，而第二种错误存在于设计规范之中。这些错误可能是未能说明的功能性描述，也可能是互相矛盾的需求以及不现实的功能。惟一能够检测这类错误的途径是利用冗余性，这是由于规范已经位于抽象层次的顶层，因此不存在与之参考的检测模型。举行设计评审会并建立一支详细检查整个设计体系结构的团队也是一种通过冗余性进行验证的有效方式。除了直接检验冗余性之外，对设计成为产品之后