第 1 章　数字设计方法概论

电路设计的经典设计方法是依赖于电路原理图的人工设计方法，而现在的大规模复杂电路广泛采用基于计算机语言的现代设计方法。这种实践变革有几方面的原因，其中最重要的原因是没有任何一支设计工程师团队能够用人工方法有效、全面、正确地设计和管理含数百万门级的现代集成电路(IC)。但使用硬件描述语言(HDL)，工程师们能很容易地实现对大型复杂电路系统的设计和管理。即使小规模电路的设计也更多地依赖于基于语言的描述，因为工程师们必须快速设计生产出满足瞬息万变的市场需求的产品。

基于语言的设计易于移植且不依赖于工艺，设计团队也可以重用或修改以前的设计，以保持与更先进工艺的一致性。随着器件物理尺寸的缩小，电路密度的提高，基于原有 HDL 模型进行综合生成的电路同样具有更高的性能。

硬件描述语言也是将各种设计专利成果集成为知识产权核(IP)的一种方便而有效的工具和手段。通过使用这种通用设计语言的描述，电路模块可以根据需要单独或合并进行综合和测试，以缩短设计周期。有些仿真工具还支持基于多种语言的混合描述。

采用 HDL 最显著的优点在于：基于语言描述的电路及其优化可以自动地进行综合，而不用经历人工设计方法中那些费力的步骤(如用卡诺图化简逻辑函数)。

目前，基于 HDL 的综合方法是工业界普遍采用的主流设计方法。设计者可以通过构建一个软件原型或模型来验证其功能，然后利用综合工具自动对所设计的电路进行优化，并且可以生成针对某物理工艺技术的网表(netlist)。

HDL 和综合工具的应用使得工程师们更关注有关功能的设计，而不是具体的单个晶体管或逻辑门的设计；综合得到的电路可以实现预期的功能，并满足面积和/或性能的约束要求。无论是功能模型还是行为模型的 HDL 模型描述，都可综合出不同的结构，并可据此快速对设计进行评估和折中。

HDL 可作为多种设计工具的平台，包括：设计输入、设计验证、测试向量生成、故障分析和仿真、时序分析和/或验证、综合和原理图的自动生成等任务。HDL 这种宽范围的覆盖使得设计者的设计工作通过工具链路时，由于不再需要考虑设计描述在不同工具间的转换过程而大大提高了设计流程的工作效率。

Verilog[1] 和 VHDL[2] 两种语言受到工业界的广泛支持，这两种语言都成为了 IEEE(电气和电子工程师协会)标准，并都得到 ASIC(专用集成电路)和 FPGA(现场可编程门阵列)相关综合工具的支持。模拟电路设计语言，如 Spice[3]，在验证电路的关键时序路径上扮演着重要角色。但由于这些模拟电路描述语言对大型设计来说需要大得惊人的计算量，而且也不支持抽象设计，使得它们在大规模电路设计应用中变得很不实际。混合语言(如 Verilog-A 语言[4])用于设计兼有数字和模拟电路的混合信号系统。近几年还出现了 SystemC[5] 和 Superlog[6] 这样的系统级设计语言，它们能够支持比 Verilog 或 VHDL 语言更高抽象级别的设计。

1.1　设计方法简介

系统级设计 ASIC 和 FPGA 电路的目的是最大限度地确保设计正确，使设计没有致命缺陷并能够进行生产制造。设计者可以按照图 1.1 所示的设计流程进行电路设计。该流程中给出了数

字电路的设计、验证、综合和测试等几个主要步骤的次序。ASIC 设计流程包括了从设计规范和设计输入开始，直到芯片级布局布线以及时序收敛等几个设计进程。在设计中，当所有的信号通路都满足由接口电路、电路时序单元和系统时钟所产生的时序约束条件时，即达到时序收敛。虽然设计流看起来呈现线性关系，但实际上则不然。当发现设计出现错误，设计需求改变，或有不符合设计性能要求及设计约束改变等情况发生时，有可能要重新进行设计流程中的多个步骤。例如，如果一个设计不能满足时序约束，就不得不重新进行布局布线设计，还有可能要对一些关键路径进行重新设计。

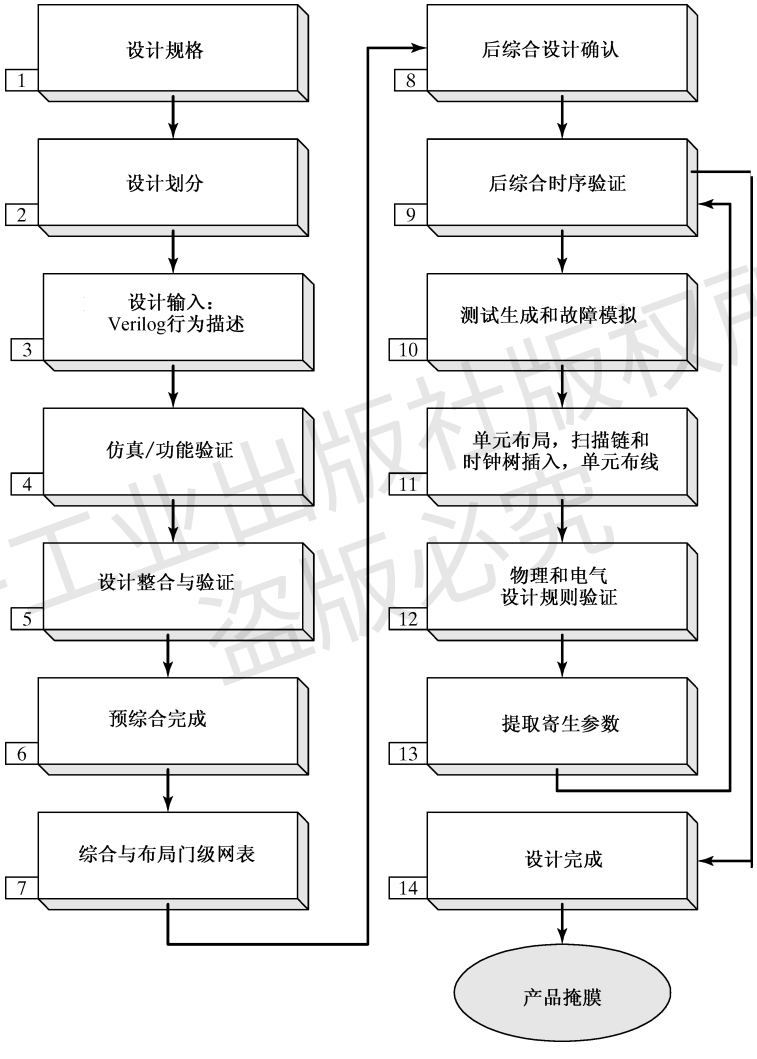

图 1.1 基于 HDL 的 ASIC 设计流程

由于 ASIC 的结构是非固定的，其设计所实现的电路性能取决于晶片上单元电路的物理布局与布线以及底层器件的特性等因素，因此基于标准单元的 ASIC 设计流程要比基于 FPGA 的设计流程更为复杂。在小于 0.18 μm 的亚微米工艺中，互连线延时对电路性能的影响起着关键的作用，其预布局布线的时序估计并不能确保满足布线路径设计的时序收敛要求。下面几节将详细阐述图 1.1 中描述的设计流程的各个步骤。

1.1.1　设计规格

设计流程从设计规格书的撰写开始。设计规格书是一个非常详尽的设计技术描述文档，包括功能、时序、硅片面积，功耗、可测试性、故障覆盖率以及其他指导设计的相关准则等。最简单的设计规格书至少要描述出设计所要实现的功能特性。一般情况下，时序电路用包括状态转移图、时序图和算法状态机(ASM)图表等描述。但规格书的解释说明仍然可能存在某些隐藏问题，这是因为基于 HDL 的模型在具体实现时有可能忽视规格书的某些解释说明。目前正在推广应用的 SystemC[5] 和 Superlog[6] 等高级语言由于其本身就具有设计规格的描述，以及将描述转换为可综合电路的功能，因而可以解决上面所提及的相应问题。

1.1.2　设计划分

现今 ASIC 和 FPGA 实现电路的设计方法论中，把大型电路划分为一个有多个相互关联的功能单元模块所组成的系统结构(architecture)，其中每一个功能单元都能够具有描述其功能的行为模型。把一个复杂系统的设计逐渐划分成规模较小且功能简单的单元电路，这样的划分过程通常被称为自顶向下(top-down design)或层次化(hierarchical design)的设计方法。HDL 支持自顶向下设计的各层次抽象级别的描述，它提供了规范的系统框架来根据需要对一个大型复杂系统进行电路划分、综合和验证。大型复杂系统的各个模块可链接在一起，以进行系统整体功能和性能的验证。经过划分的各个功能模块单元比整个系统更简单，且每个模块单元都可以用基于 HDL 的模型来描述。大型系统的总体设计往往会因其电路规模太大而无法直接综合，但划分后的各个功能单元则可以在合适的时间内进行综合。

1.1.3　设计输入

设计输入是先形成一个基于语言描述的设计，并将其以电子格式的方式存储在计算机中。在现代设计方法中，通常用诸如 Verilog 这样的硬件描述语言来进行描述，与诸如自底向上的手动输入等其他方式相比，采用前者编写一个大型电路的 Verilog 行为描述文件并实现其门级电路综合所花费的时间要少得多，节省下来的时间可用于设计流程的其他部分。Verilog 描述很容易进行编写、修改和替换，因而方便探索电路采用不同的实现结构。此外，综合工具本身也会自动查找具有同样功能的其他实现形式，并能产生描述该设计属性的报告文件。

在把 HDL 描述映射到目标工艺之前，综合工具会创建一个该电路的最佳内部描述格式。在此阶段其内部数据库是通用的，可将一个 HDL 描述映射至各种不同的工艺。例如，综合工具转换引擎技术可以利用内部格式把 FPGA 的设计映射为 ASIC 标准单元库。在此移植过程中不必重新优化其通用描述。

基于 HDL 的设计要比电路原理图设计更容易调试。行为描述方法总结抽象了电路的复杂功能，并隐藏了许多门级的底层细节，因此在功能设计中对出现的问题进行隔离处理，用较少的信息量来实现并简化设计。此外，如果行为描述在功能上是正确的，它就会成为后续的门级电路实现时有价值的设计规范。

基于 HDL 的设计通常在设计中通过使用描述名、加入明晰内容的注释、明确地指明结构的相互关系等文本内容的方式，从而减少了必须保存在其他归档文件中的文档数量和内容。基于 HDL 语言模型的仿真能够清楚地反映出该设计的功能特性。由于 HDL 语言是标准化的，因此不同厂商提供的设计工具平台都支持用它描述的设计文档。

行为建模是工业界用于进行大规模芯片设计的主要描述方法，行为建模描述一个设计的功能特性，即仅指定所设计的电路将要做什么，而无需指出怎样用硬件去构建电路。行为建模只需

描述逻辑电路的输入/输出模型,而不必关注其物理层和门级的实现细节。

行为建模鼓励设计者按下述步骤进行设计:

(1)快速创建一个设计的行为级原型电路(而不要受硬件实现细节的制约);

(2)验证其功能特性;

(3)利用一种综合工具对设计进行优化,并将设计转换成某种物理工艺。

如果这个模型已经写成可综合格式,综合工具将去除其中的冗余逻辑,并且将在其他结构和/或多级等效电路之间进行权衡,最终完成一个能兼顾面积限制或时序约束的设计。行为建模是把设计者的考虑重心放到实现的电路功能上,而不是具体的逻辑门及其互连上。在将设计提交生产之前它还提供了选择其他可选方案的自由度。

除了在综合过程中的重要作用之外,行为建模还可将一个工程设计的各个单元在不同的抽象级别上进行仿真,提高了设计的灵活性。Verilog 语言可用于混合几种抽象级别的描述,使得在门级上实现的设计部分能够与用行为描述表示的其他设计部分整合在一起,并进行仿真。

1.1.4 仿真与功能验证

设计的功能特性能够通过仿真或者形式化方法[7]来进行验证(参见图 1.1 中的步骤 4)。我们主要讨论在此提到的相应规模电路的仿真问题。除非设计的功能特性验证通过,否则设计流程将返回到步骤 3。

整个验证过程分三步进行:

(1)拟定测试方案;

(2)建立测试平台 testbench;

(3)测试执行。

1.1.4.1 拟定测试方案

测试方案要认真组织、编写,以确定什么是要测试的功能特性和如何进行测试。例如,测试方案指明一个算术逻辑单元(ALU)的指令集将在输入特定数据集时,通过对 ALU 行为的详尽仿真来进行校验。对于时序电路的测试方案,因为其状态数目很多,所以必须要有更详尽的描述,以确保设计的高可信度。测试方案应指定激励发生器、响应监测器以及判断被测模型的响应所需的有价值的参考判据。

1.1.4.2 建立测试平台

testbench 是一个 Verilog 模块,在仿真时这个模块中的待测试单元(UUT)已被实例化,同时测试向量发生器也被施加到该模块的输入端。图形显示器或响应监测器都是测试平台的主要组成部分。编写测试平台文件是为了识别在仿真(比如,测试操作码)进行过程中所观测的目标和它的有序活动。若一个设计为多模块结构形式,就要对它的每一个模块分别进行验证,从设计的最低层次开始,然后再对整合设计进行测试来验证模块之间的相互作用是否正确。在这种情况下,测试方案必须描述每个模块的功能特性和将要被测试模块的测试过程,而且这个方案还必须指出怎样对总体设计进行测试。

1.1.4.3 测试执行和模型验证

testbench 可根据测试方案进行完善,而且要对设计的原始指标对应的响应进行验证,例如响应是否与所描述 ALU 相匹配?这一环节的目的主要是找出设计中的错误所在、确定描述语法的正确性、检验习惯用法、为后续的综合工作消除障碍。模型验证要求系统地、彻底地展现出该模型的行为级的验证实例,所以在模型验证完毕之前就进入下一步设计流程是没有意义的。

1.1.5　设计整合与验证

在对已划分设计的每个功能子单元进行验证，并确认其具有正确的功能特性之后，还必须把这些功能子单元重新整合成一个完整的系统，再验证其功能特性正确与否。这就需要对 testbench 重新进行开发，使其激励发生器能够实现上层模块的输入/输出功能，监测端口以及穿越模块边界时总线的活动，并且观测每个内嵌状态机的状态转换情况。这一步骤在设计流程中是至关重要的，必须要完全彻底地执行，以保证设计是在综合正确的情况下完成的。

1.1.6　预综合完成

testbench 要提供全部功能特性的验证实例，而且要保证 Verilog 行为模型的功能特性与设计规范不出现偏差且完全一致。所有已发现的功能错误和问题均被解决后，即完成了预综合。

1.1.7　门级综合与工艺映射

当设计中的所有语法错误和功能错误均已消除且已完成预综合后，通过综合工具创建一个最优布尔描述，并且利用一种有效工艺构建这个设计描述。一般来说，综合工具能够去除冗余逻辑，以寻求能实现功能特性并满足性能(速度)指标要求的最小面积的逻辑电路结构，这一步将产生一个标准单元网表或配置目标 FPGA 的数据库文件。

1.1.8　后综合设计确认

设计确认就是把已综合完的门级描述的响应与行为模型的响应相比较。这可以通过图 1.2 所示的一个有两种模型和一个公共激励发生器的测试平台来完成。其响应能够通过软件或通过虚拟/图形工具进行监测来观察是否两种模型具有同样的功能特性。对于同步设计，在机器周期的边界处必须保持同步——而中间部分则无关紧要。如果行为描述和已综合实现的功能特性不一致，往往就要通过细致的工作研究消除其中的差异。后综合设计确认能够发现行为模型在不同的时钟周期触发产生非预期事件的软竞争情况。下面我们将讨论一种能防止这种情况出现的好的建模方法。①

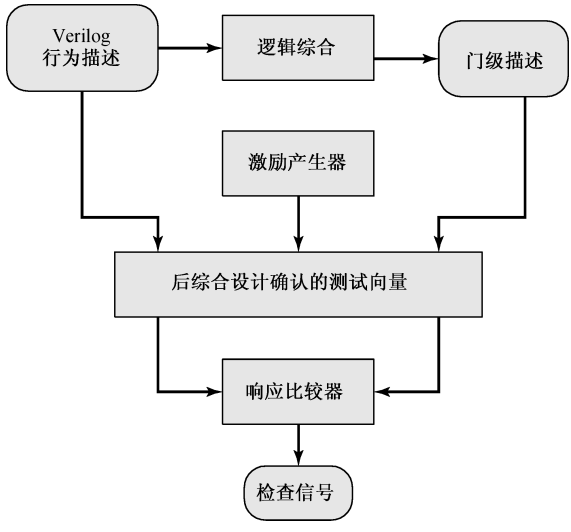

图 1.2　后综合设计确认

① ASIC 设计流程的后综合确认之后，有一个后布局时序验证的步骤。

1.1.9 后综合时序验证

尽管综合过程能产生满足时序规范要求的电路,然而检查电路的时序边界是为了验证关键信号通路上的速度是否达到要求(步骤9)。在设计流程进行到步骤13之后,还需要返回来执行步骤9,这是因为综合工具不能精确地估计版图中连接金属线导致的电容性时延效应。归根结底,还必须从金属材料的特性以及制造掩膜的几何形状等几个方面提取时延参数,并根据所提取的时延参数,利用静态时序分析器来验证最长的通路是否违反时序约束。有时候还不得不通过重新综合电路或是重新布局布线来满足设计规范的要求。对于重新综合的情况,则需要:

(1)重新设计晶体管的尺寸;

(2)改进或替换电路的结构;

(3)替换器件(速度越快面积开销越大)。

1.1.10 测试生成与故障模拟

集成电路在完成制造之后必须进行测试,目的是检验它们有无缺陷,能否正常工作。洁净间中的污染物也会导致电路出现缺陷而使其不能使用。在设计流程的这个步骤中,要用一组测试向量来测量电路的响应。这种测试是针对由制造加工引起的故障而不是设计错误,设计错误应该在预综合结束之前就被检测出来。测试是令人望而生畏的一件事情,一个ASIC芯片可能有数百万个晶体管。但仅有几百个封装引脚可以用来探测内部电路的工作情况。为此,设计者不得不在设计电路中预埋一些额外的特定电路,使得测试器只用几个外部引脚就能单独测试ASIC,或者在印制电路板上测试ASIC的全部内部电路。

验证行为模型的平台能够用来测试由综合产生的电路故障,但对于检测足够高级别的制造缺陷还不够理想。组合电路故障全部都能被检测出来,而时序电路故障的检测则会遇到将在第11章中讲到的问题。故障模拟研究的是从生产线出来的芯片是否能够通过测试来检验其工作情况正常与否。故障模拟的目的是为了确定一组测试向量是否能检测出一组故障。故障模拟的结果用于指导软件工具的使用以便生成更完备的测试模板或测试向量。为了消除那些不能被直接测试到的模块出现故障的可能性,在器件制造前需要生成测试模板或测试向量(如扫描链)并将这些测试模板加入到原始设计中。①

1.1.11 布局与布线

ASIC设计流程中的布局与布线就是将设计单元适当地放置在晶片内,并连接信号通路。在基于单元设计工艺中,需要将各个设计单元整合在一起,形成一个能把全部逻辑门电路刻制在硅晶片上的完整掩膜板图案。这一环节还可能包括在布局中插入时钟树,以便给设计中的时序元件提供一个规则分布的时钟信号。扫描链的插入也是在这一步实现的。

1.1.12 物理和电气设计规则检查

对设计的物理布图进行检查是为了检查线宽、交叠、间隔等约束是否满足要求。电气规则检查是检查扇出约束是否满足,信号的完整性是否没有被电气串扰和电源栅压降所破坏。噪声电

① 扫描链是将电路中的普通寄存器用特定设计的寄存器替换后而形成的,这些特定设计的寄存器在测试模式下能互连在一起并形成移位寄存器。测试模板可以加入到原始设计的电路中。电路的响应可以通过扫描链捕捉到并移位输出以供分析。

平检查以判断电平瞬变特性是否存在问题。功率耗散也要在这一步骤中进行模拟和分析，以便确认芯片产生的热量不会对电路造成损坏。

1.1.13　提取寄生参量

版图所形成的寄生电容能够通过软件工具提取，并用所提取的参数对设计的电气特性和时序性能进行更精确的校验(步骤 13)。利用该提取步骤得到的寄生参量结果来更新时序分析中所用到的负载模型。然后对设计电路的时序约束再次检查，确保在特定的时钟速度下设计方案有效。

1.1.14　设计完成

在所有设计约束都已经满足，也达到了时序约束条件的情况下，就会发出最终设计完成信号。可用于制造集成电路的掩膜集也就具备了。掩膜的格式是由几何数据(通常为 GDS-Ⅱ格式)构成的，这些数据决定了集成电路制造过程中的光刻步骤的顺序。至此，设计者已耗费了大量的人力和物力来确保要制造的芯片在功能和性能上满足设计规范的要求。

1.2　IC 工艺选择

图 1.3 给出了从可编程逻辑器件(PLD)到全定制 IC 制造工艺的、可用来构建数字电路的硅物理实现的各种可选方案。固定架构的可编程逻辑器件适用于低端市场(即低规模且低性能需求)。这些产品相对来说价格低廉，面向小规模设计。

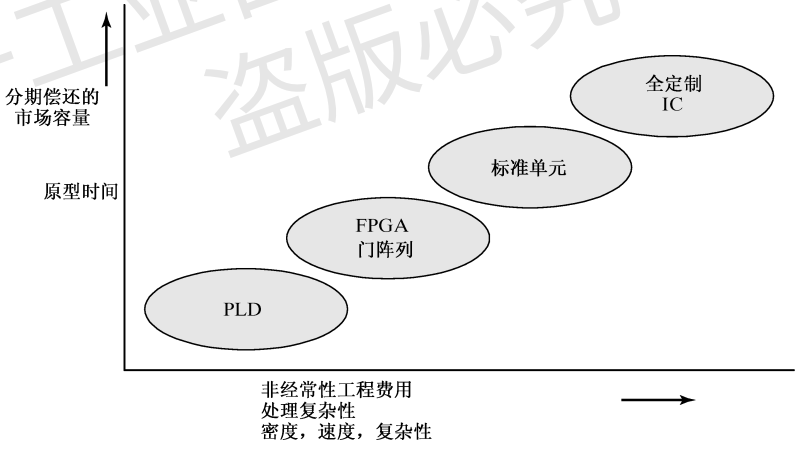

图 1.3　IC 实现的不同工艺

ASIC 设计实现的物理方式有：

(1)高性能电路的全定制版图；

(2)标准单元结构；

(3)门阵列(现场可编程或掩膜可编程)。

究竟采用哪一种实现方式，取决于 ASIC 的市场预期是否能收回其设计成本并达到厂商所需要的利润。用全定制 IC 具有高的性价比，但它需要有足够大的产量或者有足够多的用户群，还要有充足的开发时间和投资以便确保生产出面积最小和速度最高的全定制设计产品。FPGA 具有固定及电可编程的结构，适合用于规模适中的设计实现。设计者可以用支持这种工艺实现方式

的工具,用较短的时间编写并综合一个 Verilog 描述,并在原型机上生成可以运行的物理电路。因此,该设计修改的成本非常低。由于 FPGA 的器件封装和管脚排列都是已知的,因此电路板的版图和电路的研发可以同时进行。在少量的产品原型设计阶段,可采用掩膜可编程和基于标准单元的设计方式实现①。

掩膜可编程门阵列工艺中,晶片上集成的晶体管阵列可以根据需求互连得到逻辑门电路,并实现所期望的功能。晶片上的器件是预先制造好的,用户可以根据需求自行连接金属互连线。晶片上只有金属掩膜是开放的,因此可大大减少完成掩膜工艺所需的时间和成本。其他的 NRE 工程费用(Non-Recurring Engineering,一次性工程费用或非经常性工程成本)可以分摊给硅制造厂家的所有客户。

标准单元工艺技术预先设计并特征化了掩膜层的各个逻辑门,且封装在公用库中。利用布局布线工具把这些单元排放在晶片的相应通道位置,再进行互联,然后整合这些掩膜,这样就可以制造出有特定应用功能的集成电路。该用户的掩膜集是特定于其实现的逻辑功能的,对于大型电路来说,该项费用将超过 50 万美元。但是与该设计有关的 NRE 工程费用和单元库设计费被分摊到所有的客户群。对那些有大量应用的同等规模的集成电路,标准单元制造工艺的单位成本比 PLD 和 FPGA 的单位成本还要低。

1.3 后续内容概览

接下来的几章将涵盖图 1.1 所描述的设计流图中最主要的步骤,但不包括单元布局和布线、设计规则检查和寄生参量提取。这几个步骤是在功能正确的设计已成功综合的情况下,才能进行的设计步骤,它们将通过工作在物理掩膜数据库上的其他工具管理、实施,而不是基于 HDL 设计模型。下面将要描述的步骤是在所有 ASIC 流程中以设计者为主导的关键设计步骤。

在其余各章中,第 2 章和第 3 章将回顾组合和时序逻辑电路的人工设计方法,之后讲述用 Verilog 进行组合逻辑设计(第 4 章)和时序逻辑设计(第 5 章)的方法,并通过举例,将人工方法和基于 HDL 的方法进行对比。这里还介绍了算法状态机(ASM)图,以及算法状态机及数据通道(ASMD)图表的应用,说明了这些图表在编写时序电路的行为模型中是非常有用的。第 6 章讨论了用 Verilog 模型实现组合逻辑和时序逻辑综合,还给设计人员提供了构思最佳综合设计的背景资料,并且给出了可能导致设计失败的常见错误。第 7 章概括了 RISC CPU 和 UART 数据通道控制器的处理问题。第 8 章介绍了 PLD,CPLD,RAM 和 ROM 以及 FPGA 可编程逻辑器件,并在章末的习题中给出了在广泛应用的评估板上实现的设计。第 9 章描述了数字处理器的算法与结构。第 10 章讨论算术指令的实现问题。第 11 章讲述了时序验证、测试生成和故障模拟等后综合设计问题,还包括了 JTAG 和 BIST 等方面的讨论。

学习用 Verilog 语言进行设计有三件事最重要:例程,例程,例程。我们举了一些难度渐进的例程,并且给出了这些例程的 Verilog 描述。在每章的末尾列出了一些要求用 Verilog 进行设计的、颇具挑战性的习题,我们希望在设计中读者能够采纳下面的忠告:精炼、清晰和可验证。

参考文献

1. *IEEE Standard Hardware Description Language Based on the Verilog Hardware Description Language*, Language Reference Manual (LRM), IEEE Std. 1364 - 1995. Piscataway, NJ: Institute of Electrical and Electronic Engineers, 1996.

① 现在业界多采用现场可编程器件 FPGA/CPLD 器件进行原型机设计。——译者注

2. *IEEE Standard VHDL Language Reference Manual* (LRM), IEEE Std, 1076 - 1987. Piscataway, NJ: Institute of Electrical and Electronic Engineers, 1988.
3. Negel LW. *SPICE2: A Computer Program to Simulate Semiconductor Circuits*, Memo ERL-M520, Department of Electrical Engineering and Computer Science, University of California at Berkeley, May 9, 1975.
4. Fitzpatrick D, Miller I. *Analog Behavioral Modeling with the Verilog-A Language*, Boston: Kluwer, 1998.
5. SystemC Draft Specification, Mountain View, CA: Synopsys, 1999.
6. Rich, D., Fitzpatrick, T., "Advanced Verification Using the Superlog Language," Proc. Int. HDL Conference, San Jose, March 2002.
7. Chang H, et al. *Surviving the SOC Revolution*, Boston: Kluwer, 1999.