第1章 可编程逻辑器件设计流程导论

本章简要介绍了可编程逻辑器件设计流程，其主要内容包括：设计流程概述；设计输入和综合；设计实现；设计验证以及 FPGA 设计技巧概述几个部分。

本章的目的是使读者初步了解 Xilinx 的可编程逻辑器件的设计流程，以便读者更好地掌握本书后面的内容。

1.1 设计流程概述

图 1.1 给出了标准的 Xilinx 设计流程，从图中可以看出可编程逻辑器件标准设计流程包括以下步骤：

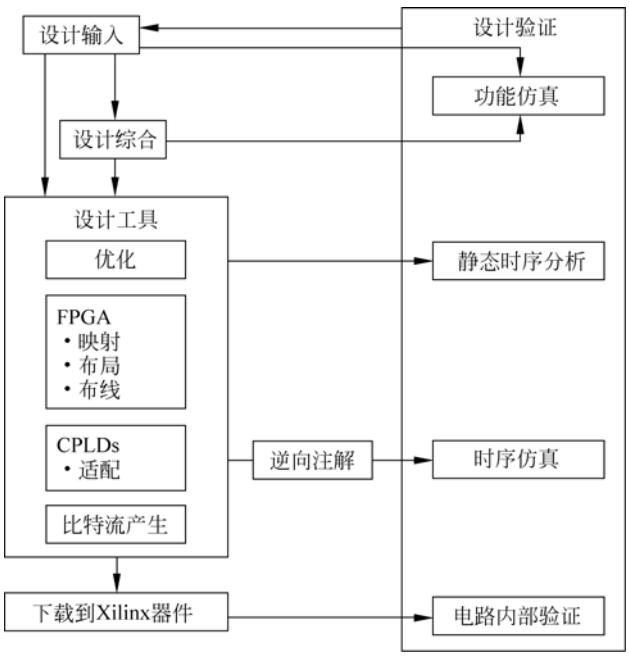

图 1.1 Xilinx 标准的设计流程

1．输入设计和综合

在设计流程的这一步中，通过原理图编辑器、硬件描述语言（HDL）或者两种混合方法来创建自己的设计。如果使用 HDL 创建设计输入，就必须将 HDL 文件综合到一个 EDIF 文件中；如果使用 Xilinx 的综合工具（Xilinx Synthesis Technology，XST），就必须把 HDL 文件综

合到一个 NGC 文件中。

2．设计实现

通过执行特定的 Xilinx 架构，转换逻辑设计文件格式，例如 EDIF，这样就可以将设计输入和综合步骤创建到物理文件格式中。这些物理信息存储于 FPGA 的本地电路说明文件 NGC 和 CPLD 的 VM6 文件中。再通过这些文件创建一个比特流文件，并为随后器件编程创建一个 PROM 或 EPROM 文件。

3．设计验证

使用门级仿真器或者下载电缆，确定设计满足时间要求，并能正常运行。查看 Xilinx 的下载电缆和演示版信息的在线帮助。

图 1.2 给出了详细的 Xilinx FPGA 设计的软件流程图。

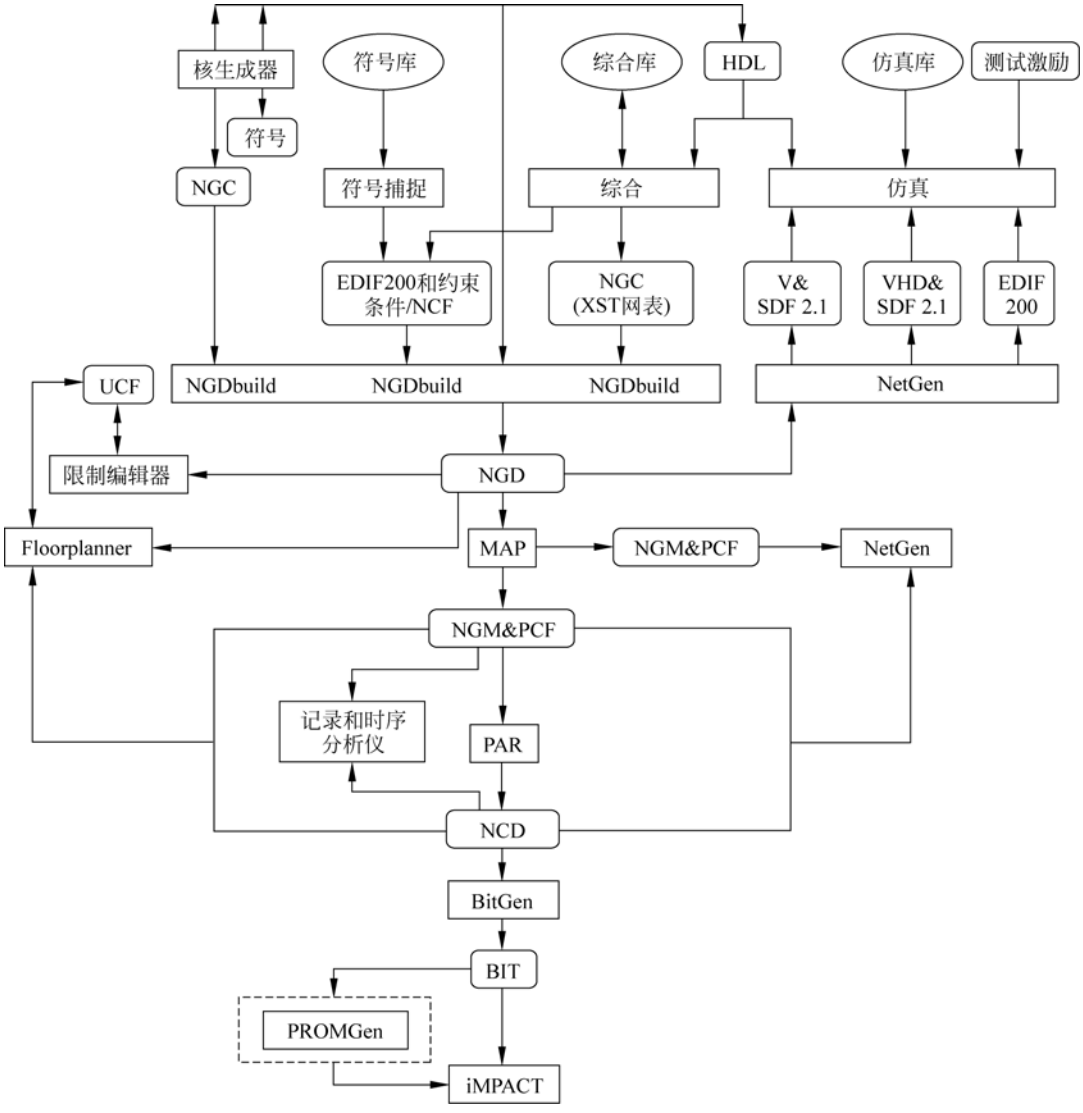

图 1.2　Xilinx FPGA 设计软件流程图

整个设计流程是一个不断地输入、执行、验证，直到设计是正确和完整的过程。Xilinx 的开发系统通过设计流程周期，允许快速反复设计。因为 Xilinx 器件允许无限制的编程，所以调试设计的电路时，不必丢弃已经编程的器件。

图 1.3 给出了详细的 Xilinx CPLD 设计的软件流程图。

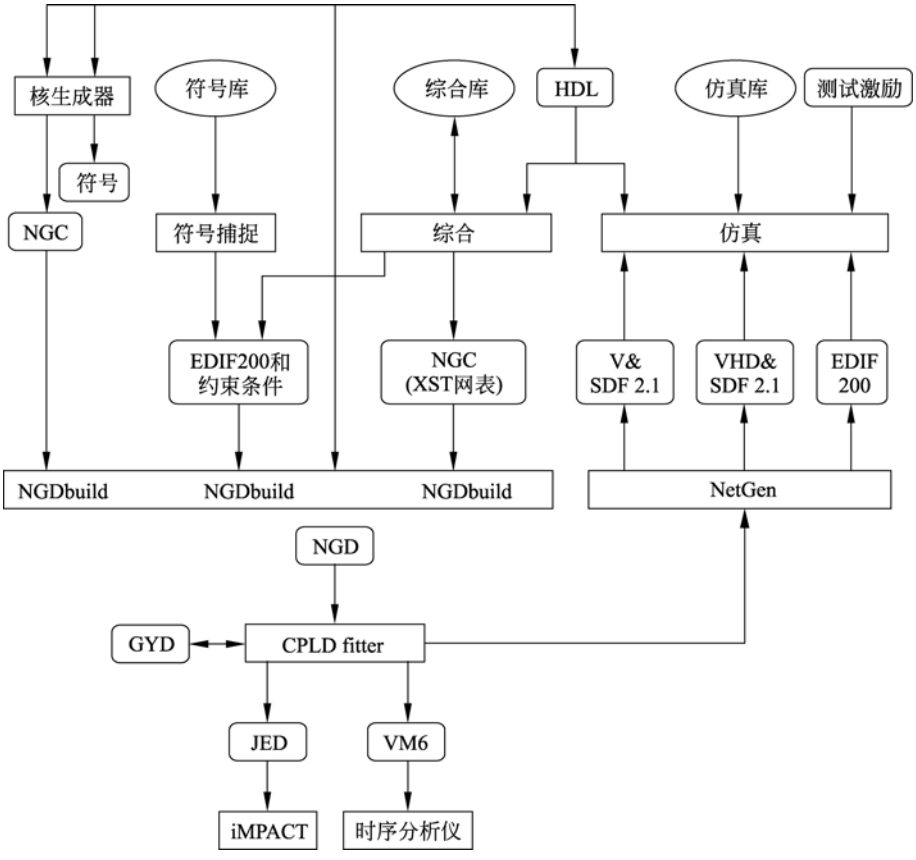

图 1.3　Xilinx CPLD 设计软件流程图

1.2　设计输入和综合

通过原理图编辑器或者文本编辑器工具输入一个设计。设计输入开始于一个设计概念，即使用画图或功能描述来表示设计。从最初的设计，创建网表，再综合并转化本地通用对象（Native Generic Object，NGO）文件。这个文件输入 Xilinx 的软件程序 NGDBuild，该程序产生一个逻辑本地通用数据库（Native Generic Database，NGD）文件。图 1.4 给出了输入设计和综合步骤。

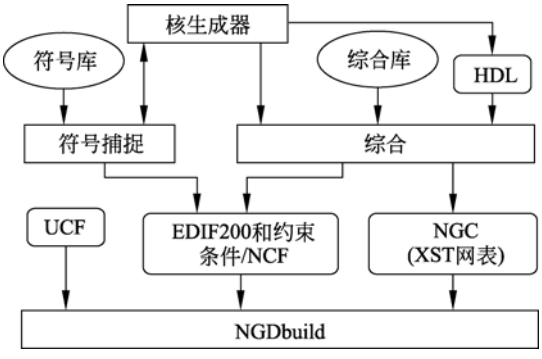

图 1.4　设计输入流程图

1.2.1 层次化设计

层次化设计对于原理图和 HDL 输入都很重要，这是因为：

- 可以将设计概念化。
- 将设计结构化。
- 使调试设计更容易。
- 使设计的不同部分的不同输入设计方法（原理图，HDL，本地编辑）能更容易结合。
- 使更新设计更容易，其中包括设计、实现，以及在设计过程中验证个别元件。
- 减少优化时间。
- 便于并行设计。

在层次化设计中，一个特定的等级名字区分了每个库元件、特定的模块，以及实例创建。

对于设计中的每一个元件和网络都应该命名，这些名称通过 FPGA 编辑工具储存和调用，还用来备份注释并出现在调试和分析工具中。如果设计者没有命名，那么原理图编辑器会自动生成名字。对于自动生成名称的电路很难进行分析，因为这些名称仅对 Xilinx 的软件有意义。

1.2.2 原理图输入

原理图工具为原理图设计输入提供了一个图形界面。通过这些工具将设计中的符号连接在一起，这些符号表示设计中的逻辑元件。设计人员可以使用独立的门来建立设计，或者连接多个门建立功能块。

1．库元件

元件和宏都是元件库中的结构单元。Xilinx 库提供了原语，以及高级的宏功能。原语是基本的电路元件，例如与门和或门。每个原语都有唯一的库名，标号和描述。宏包括多个库元件，这些库元件包含原语和其他宏。Xilinx 的 FPGA 可以使用以下类型的宏：

- 软宏（软核）具有预先定义的功能，但有灵活的映射、布局和布线。软宏可用于所有的 FPGA。
- 相关布局宏（RPMs）有固定映射和相对布局。RPMs 可用于所有芯片，除了 XC9500 系列的 CPLD。

宏不能用于综合，因为综合工具有它们自己的模块生成器并且部要求 RPMs。如果想要覆盖模块生成，设计者可以例化核生成器模块。对于大多数的高级综合工具，这并不能提供优势，除非是对于不能推断的模块。

2．核生成器工具（仅限于 FPGA）

Xilinx 的核生成器传递参数化的核，这个核针对 Xilinx FPGA 进行了优化。这些库包含从简单延迟元件到复杂数字信号处理的滤波器以及多路复用器。

1.2.3 HDL 输入和综合

一个典型的硬件描述语言（HDL）支持混合描述，在这个描述中门和网表结构被用于功能描述。这种混合描述能力可以以高度抽象的形式描述系统结构，并逐步完善设计中的门级实现细节。HDL 语言描述具有下列优点：

- 在设计阶段就可以及早验证设计功能。一个用 HDL 写的设计可以立即进行仿真。在这个高层次和门级的设计仿真，在门级实现之前，就可对结构和设计决策进行评估。
- HDL 描述比网表或者原理图描述更易阅读和理解。对于一个设计，HDL 描述提供独立技术文档以及功能。
- 对于大的设计，HDL 工具比原理图工具更容易处理。

在创建 HDL 设计之后，要对它进行综合。在综合时，HDL 文件中的行为信息被转换为结构网表，并且为 Xilinx 芯片优化设计。Xilinx 支持一些第三方厂商的综合工具。此外，Xilinx 也提供自己的综合技术（Xilinx Synthesis Technology，XST）工具。

1．功能仿真

在创建设计之后，可对其进行仿真。功能仿真是对设计中的逻辑进行测试，以判断其是否正常工作。如果在设计流程中尽早地进行功能仿真，可以为随后的设计步骤节省时间。

2．约束

如果想要对设计中的时间参数或者布局参数进行约束，设计者可以指定映射、块布局以及时间规范。

可以手工输入约束，或者使用约束编辑器 Floorplanner 或者 FPGA 编辑器（Xilinx 提供的图形用户界面工具）；还可以使用时序分析图形界面或者 TRACE 命令行程序来评估电路所违反的这些约束条件（通过生成设计的静态时序分析）。

1）映射约束

在 Spartan 系列和 Virtex 系列的 FPGA 中，可以指定使用 FMAP 来确定逻辑块如何映射到可配置的逻辑块（Configurable Logic Block，CLB）。这些映射符号可在原理图中使用。但是，如果过多地使用这些映射限制，则会使得设计中的布线非常困难。

2）模块布局

块布局可限制在指定位置，可以是多个位置中的其中一个，或者是一个位置区域。具体位置可以用综合工具在原理图中指定出来，或者在用户约束文件（Usr Constraint File，UCF）中指定出来。不恰当的块布局会影响设计的布局和布线。只有 I/O 块要求布局满足外部引脚的要求。

3．时序规范

可以指定设计中路径的时间要求。在对设计进行布局和布线时，PAR 使用这些时间规范来达到最佳性能。

1.3 设计实现

设计实现过程从逻辑设计文件映射或适配到指定的器件开始，到物理设计布线成功并生成比特流文件时结束。在实现阶段也可像在设计初期一样改变约束条件。图 1.5 给出了 FPGA 的设计实现过程。

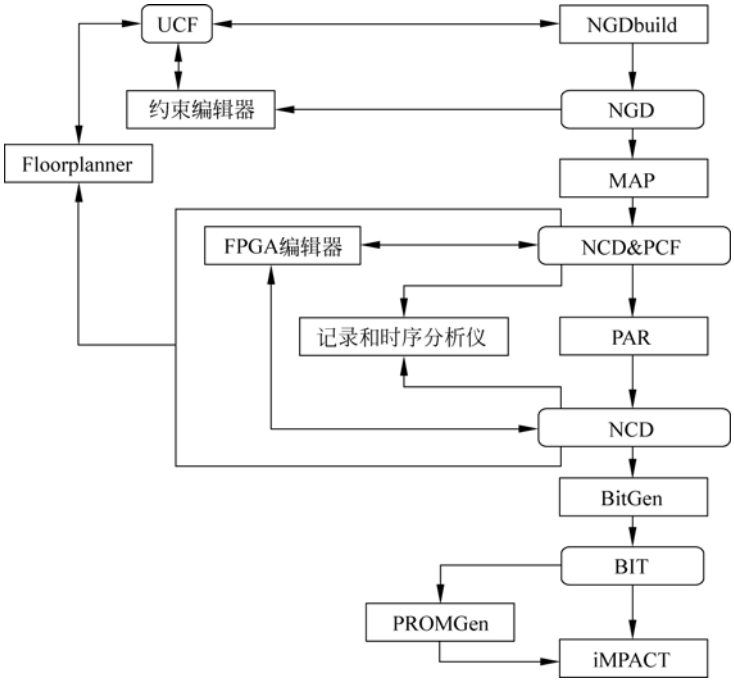

图 1.5　FPGA 的设计实现流程

下面对关键步骤进行说明：

1．映射（MAP，仅限于 FPGA）

对于 FPGA，MAP 命令行程序将逻辑设计映射到 Xilinx 的 FPGA 芯片。MAP 的输入是 NGD 文件，NGD 文件包括设计的逻辑描述，这些逻辑描述的方式包括层次化的元件，低层次的 Xilinx 原语和任意数量的硬布局布线宏（NMC）文件（该文件包含物理宏的定义）。然后 MAP 将逻辑映射到 Xilinx FPGA 内的元件（逻辑单元，I/O 单元和其他元件）。

MAP 的输出设计是一个 NCD 文件，是设计被映射到 Xilinx FPGA 内元件的物理描述。然后使用 PAR 命令行程序，可以对 NCD 文件进行布局和布线。

2．布局和布线（PAR，仅限于 FPGA）

对于 FPGA，PAR 命令行程序用映射的 NCD 文件作为输入，对设计进行布局和布线，输出一个布局布线的 NCD 文件，该文件被比特流生成器 BitGen 使用。当设计变化时，设计者对设计反复进行布局和布线时，NCD 文件可以作为一个向导文件。在以下情况也可使用 FPGA Edit 的 GUI 工具：

- 对于一个完整的设计，在使用自动布局和布线工具前先布局和布线关键的元件。
- 手动修改布局和布线，编辑器允许手动和自动的元件布局和布线。

3．比特流的生成（BitGen，仅限于 FPGA）

对于 FPGA，BitGen 命令行程序为 Xilinx 设备配置生成比特流。BitGen 以完整的布线的 NCD 文件作为输入，并生成一个配置比特流（.bit 扩展名的二进制文件）。BIT 文件包含所有的来自 NCD 文件定义内部逻辑和 FPGA 互连的配置信息，还有来自和目标设备相关的文件。

在生成 BIT 文件之后，使用 IMPACT 图形工具可将其下载到芯片。也可使用 PromGen 命令行程序将 BIT 文件格式化为 PROM 文件，并用 IMPACT 工具下载到 PROM。

图 1.6 给出了 CPLD 的设计实现流程。

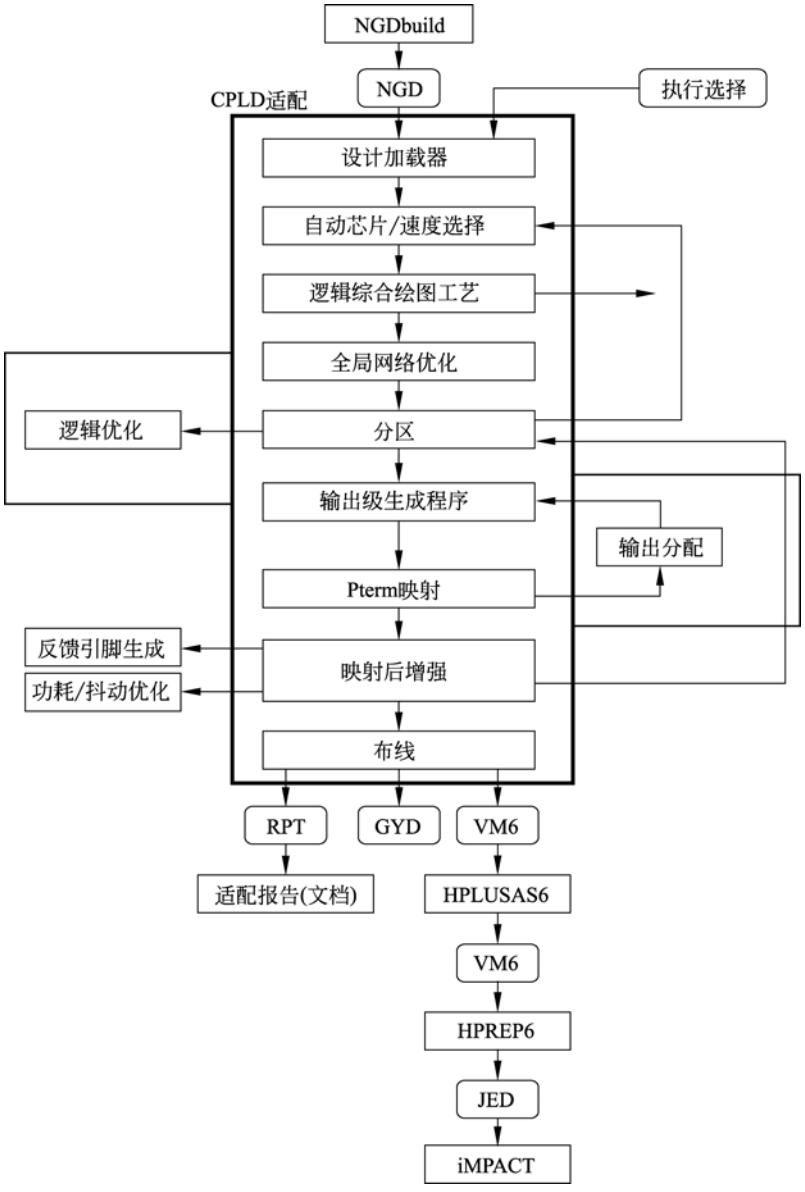

图 1.6　CPLD 的设计实现流程

1.4 设计验证

设计验证是对设计的功能和性能的测试，可以按照以下几个方法对 Xilinx 的设计进行验证：（1）仿真（功能和时间）；（2）静态时序分析；（3）电路验证。表 1.1 列出了不同的设计工具适用的验证类型。

表 1.1 验证工具

验证类型	工具
仿真	第三方模拟器（集成和非集成的）
静态时序分析	TRACE（命令行程序）；时序分析器（GUI）
	Mentor Graphics TAU 和 Innoveda BLAST 软件使用 STAMP 文件格式（仅用于验证输入/输出的时序）
内部电路验证	设计规则检查器（命令行程序）
	下载电缆

设计验证步骤应贯穿设计的全部过程中，图 1.7 给出了 FPGA 设计流程中的三种验证方法。图 1.8 给出了 CPLD 设计流程中的三种验证方法。

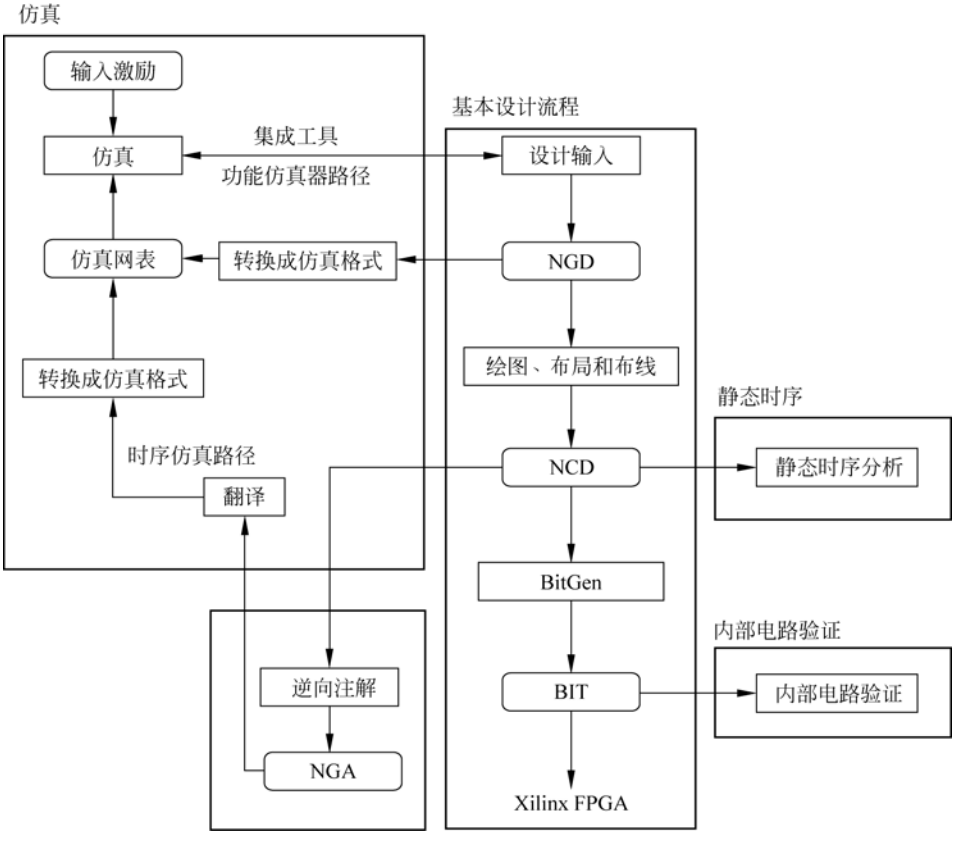

图 1.7 FPGA 设计流程中的三种验证方法

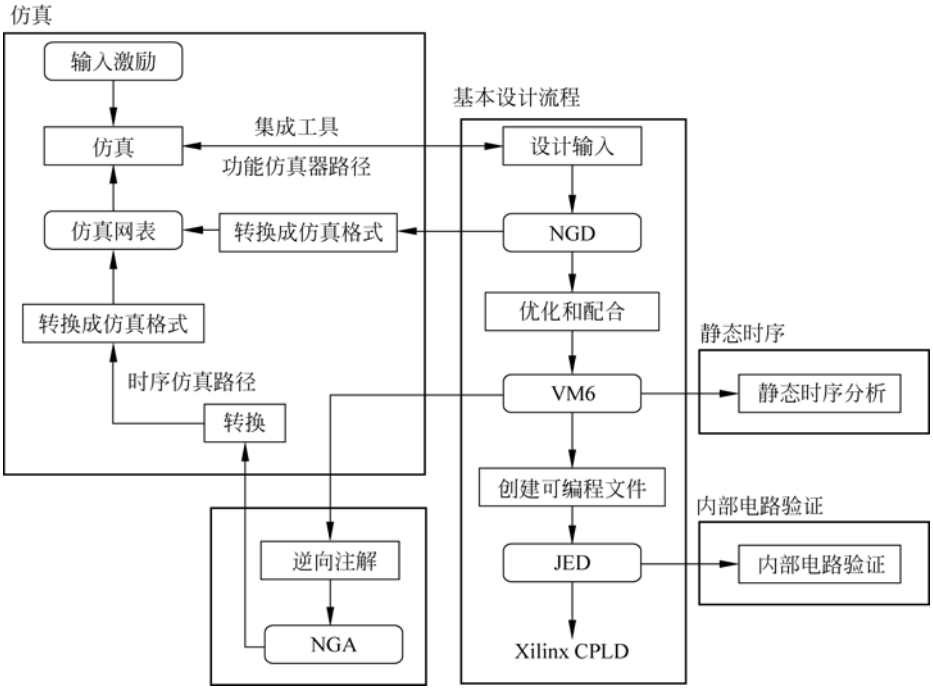

图 1.8　CPLD 设计流程中的三种验证方法

1.4.1　仿真

功能仿真和时序仿真都可用于验证设计。这部分介绍了在时序仿真前的逆向注解的过程（back-annotation）。同时也介绍了对于基于原理图和 HDL 设计的功能和时序仿真的方法。

1）逆向注解

在时序仿真之前，物理设计信息必须转化并分配回逻辑设计。对于 FPGA，使用程序 NetGen 进行逆向注解过程。对于 CPLD，使用 Tsim 时序仿真器进行逆向注解。这些程序建立一个数据库，将逆向注解信息转化为网表格式，并用于时序仿真。

图 1.9 给出了 FPGA 逆向注解的流程，图 1.10 给出了 CPLD 逆向注解的流程。

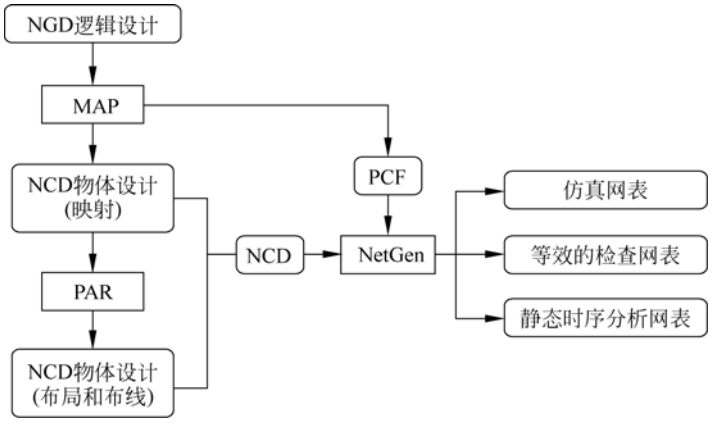

图 1.9　FPGA 的逆向注解流程

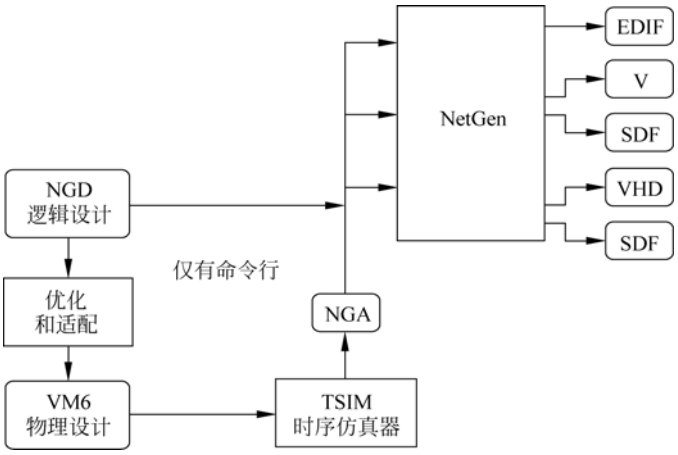

图 1.10　CPLD 的逆向注解流程

2）NetGen

NetGen 是一个命令行程序，负责分配延迟、设置和保持时间、时钟的信息到输出，并将物理 NCD 设计文件中的脉冲宽度回到逻辑 NGD 文件，生成 Verilog 或 VHDL 网表，和静态时序分析工具一起用于支持时序仿真、等价性检验。

NetGen 以 NCD 作为输入。NCD 文件可以仅仅是一个映射设计，或者是部分或完整的布局和布线设计。MAP 所创建的 NGM 文件是可选择的输入源。NetGen 将可选择的 NGM 文件中的映射信息和 NCD 文件中的布局、布线和定时信息合并起来。在 CPLD 设计中，NetGen 以 NGA 文件作为输入，并生成时序仿真网表。

1．基于原理图的仿真

设计仿真涉及使用软件模型来测试设计。在最坏条件下测试设计的功能和性能时，这是最有效的。设计者可以轻松地探测内部节点来检查电路的逻辑行为，再通过这些结果来修改原理图。

仿真使用的是与 Xilinx 开发系统相联系的第三方工具。使用各种 CAE 专用的用户引导界面作为主要参考，包括 Xilinx 支持仿真器的命令和特征。

设计仿真工具中提供的软件模型，用于执行详细的设计描述，可以执行以下的功能仿真和时序仿真。

1）功能仿真

当设计在芯片内实现之前，用功能仿真判断设计中的逻辑是否正确。功能仿真可在设计流程的初始阶段进行。因为在该阶段，对实现的设计进行时序仿真是不可用的，所以对设计的逻辑进行测试的仿真器使用单元延迟。通常，在设计流程中越早进行功能仿真，纠正错误就能更快更容易。

可以使用集成的或非集成的仿真工具。集成的工具，如 Mentor Graphics 或者 Innoveda，通常包含一个内建接口，用于连接仿真器和原理图编辑器，这些工具可以使用同一个网表。在使用集成的工具时，可以直接从设计开始到仿真。

在捕获环境中，在设计输入后，基于原理图工具的功能仿真可以立即执行。如果仿真器没有被集成，原理图捕捉工具要求一个 Xilinx Unified 库，并且仿真器需要一个库。大多数的基