FPGA 技术基础工程实践

——基于 Vivado 与 Verilog HDL

胡迎刚　**编著**

西南交通大学出版社

· 成　都 ·

前言

PREFACE

《FPGA 技术基础工程实践》一书是作者在长期从事 EDA 教学、科研工作的经验总结，书中内容紧跟技术发展前沿，紧跟社会对 FPGA 技术人才需求的不断变化而设置，且以应用型人才实践动手能力培养为目标，符合应用型人才培养要求。本书着眼于简单、快速、高效地让读者进入 FPGA 技术世界，轻松掌握 FPGA 的开发设计技术、为广大零基础的读者提供入门级的学习资料，也为有一定基础的 FPGA 使用者提供进阶级的学习指导。

本书创新与特色：

1. 项目贯穿，“做中学”，轻知识点而重知识点应用。

教材以 20 多个难易程度层次递进的工程任务项目贯穿各个章节，没有编写关于 Verilog HDL 语法语句的章节，而是把基本概念知识点隐含于各个项目中，在完成项目过程中，引出项目涉及的语法知识，更有利于学生理解、掌握，真正体现“做中学”特色，学习效果较传统教材内容先概念、后应用具有明显提高。学生学习产出不再是掌握了枯燥的语法语句等概念，关注知识点本身，而是学会了如何应用，大大提高了知识的应用能力。

2. 理实一体，凸显应用能力培养。

教材项目化设置，非常适合理实一体教学，在完成任务过程中，讲解理论知识，再辅以更高层次的应用实践，实践—理论—实践交替进行，充分保证应用型的教学效果。

全书共分为 5 章，主要是以 Xilinx 公司的 FPGA 和最新的集成开发环境 Vivado 2019.2 为核心，深入浅出地介绍了 FPGA 技术基础应用技巧和系统设计方法。第 1 章主要介绍 FPGA 技术概念、内部结构、工作原理，了解 EDA 技术相关概念及设计流程（2 学时）；第 2 章介绍 Vivado 软件的安装和使用方法（4 学时）；第 3 章是全书的重点，通过设置 6 个项目，以案例式讲解 Verilog HDL 的基本语言、结构、常用语句完成典型数字电路设计方法，以达到做中学目的；第 4 章介绍 IP 核的设计使用方法，介绍 Vivado 提供的常用 IP 核定制流程，自定义 IP 核基本方法等；第 5 章通过设置 4 个具体时序设计案例，强化基于 FPGA 和硬件描述语言开发数字系统的能力。

本书适合于电子、通信、自动化等电子信息类专业和计算机技术应用类专业本、专科的 EDA 技术、FPGA 技术开发相关课程教材，也可以作为 FPGA 应用入门的自学参考书。书中的概念和理论参考了的某些参考文献，在此对原著书籍的作者们表示感谢。由于本人水平有限，书中难免出现错误和遗漏，希望读者批评指正。

联系方式：四川省成都市郫都区团结镇学院街 65 号

邮编：611745

邮箱：hyg_scgsxy@126.com

胡迎刚

2020 年 1 月

于四川工商学院

目录

CONTENTS

第 1 章 FPGA 技术概述

本章主要内容包括数字技术发展历史、EDA 技术简介、硬件描述语言介绍、FPGA 基本结构及工作原理。主要为开始 FPGA 学习做必要的概念铺垫，本章内容作为一般了解即可，也可作为学生自学内容。

1.1 数字技术发展历史

在计算机世界中，归根到最底层的计算，只有两种状态，既数字电路的开和关，对应于二进制数字“ 1 ”或“ 0 ”。任何最强大的计算机、最繁杂的计算也最终都是用通过 1 和 0 来实现的。这实际上暗合了中国古典哲学的“ 阴阳 ”，“ 1 ”和“ 0 ”生万物哲学。现在我们就一起来回顾一下数字逻辑电路的发展历程、实现基本理论和典型成果。

数字技术（Digital Technology）是一项与电子计算机相伴相生的科学技术，通过特殊设备，将各种信息（包括图、文、声、像等）转换为电子计算机能识别的二进制数字“ 0 ”和“ 1 ”，然后进行运算、加工、存储、传送、传播和还原的技术。

用数字信号完成对数字量进行算术运算和逻辑运算的电路称为数字电路，或数字系统。由于它具有逻辑运算和逻辑处理功能，所以又称数字逻辑电路。数字技术的发展历程与模拟电路一样，经历了由电子管、半导体分立器件到集成电路的几个时代，但其发展比模拟电路更快。从 20 世纪 60 年代开始，数字集成器件以双极型工艺制成了小规模逻辑器件。随后发展到中规模逻辑器件。20 世纪 70 年代末，微处理器的出现，使数字集成电路的性能产生质的飞跃。

1. 真空管

1904 年，英国人佛来明（J.A. Fleming）博士发明了真空二极管；1907 年，德国人德福雷斯特（Lee Do Forest）将二极管加以改良，发明了真空三极管。在数字电路的发展初期，第一代电子电路都是由抽成真空的巨大的玻璃管组成的，所以叫真空管。真空管是

利用灯丝或电路板的两极来发射电子束来控制电流。然而，并非所有的管都被抽空，一些使用气体的和较小的管使用光敏材料和磁场来控制电子的流动。它们都有共同点：价格昂贵，消耗大量电力并散发出巨大热量。它们也非常不可靠，使用中需要大量的维护。而且它们的尺寸很大，难于制造更小型的“计算机”。

2. 晶体管

1947 年 12 月 23 日，美国贝尔实验室正式成功地演示了第一个基于锗半导体的具有放大功能的点接触式晶体管，标志着现代半导体产业的诞生和信息时代的开启。晶体管可以说是 20 世纪最重要的发明。

严格地说，晶体管泛指一切的单个半导体元件，经常用来指代半导体材料制成的三极管、场效应管，等等。它的英文名字是 transistor，来自 trans-resistance（即 transfer resistance），也就是所谓的“跨阻”，指的是输出端电压变化与输入端电流变化的比值（单位是欧姆），反映了输入对输出的影响能力。

3. 集成电路

集成电路（integrated circuit，见图 1.1）是一种微型电子器件或部件。采用一定的工艺，把一个电路中所需的晶体管、电阻、电容和电感等元件及布线互连一起，制作在一小块或几小块半导体晶片或介质基片上，然后封装在一个管壳内，成为具有所需电路功能的微型结构；其中所有元件在结构上已组成一个整体，使电子元件向着微小型化、低功耗、智能化和高可靠性方面迈进了一大步。1959 年，德州仪器（Texas Instruments）的杰克基尔比（Jack Kilby）是世界上最早向世界展示将很多这些晶体管放在单个晶圆（硅片）上的人之一，他为第一个 IC 或集成电路申请了专利。

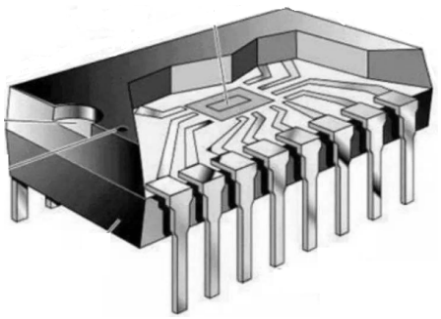


图 1.1　集成电路结构

4. 摩尔定律

20 世纪 50 年代，飞兆半导体和英特尔的联合创始人戈登 · 摩尔（Gordon Moore）发表了一篇论文，指出每个集成电路的元件数量将在未来十年每年增加一倍。1975 年，他回顾了他的预测，并表示组件的数量现在每两年增加一倍。这就是著名的摩尔定律。

几十年来摩尔定律一直被验证是正确的。而且摩尔定律一直在指导芯片制造和设计。

英特尔和 AMD 的研究人员一直以来都是根据摩尔定律设定目标和指标。由于摩尔定律迫使芯片设计的长足发展，计算机也变得越来越小。摩尔定律不仅仅是一种预测，它已成为制造商旨在实现的目标和标准。

1971 年第一个半导体采用的是 10 μm 工艺。到 2001 年，它是 130 nm，是 1971 年的近 1/80。截至 2017 年，最小的晶体管工艺为 10 nm，相比较人头发直径是 100 μm，比晶体管大近 10 000 倍。

摩尔定律危机：随着大规模电路的发展，晶体管越来越小，集成数量成几何级增加，其制造工艺却越来越难了。克服这些技术和工艺壁垒不仅需要大量的时间和人力，还需要大量的资金和投资。因此，摩尔定律中的时间也逐渐放缓，甚至它可能会很快不成立，摩尔定律危机爆发（当然如果没有巨大变革这是必然的）。英特尔花了大约两年半的时间才从 2012 年的 22 nm 工艺发展到 2014 年的 14 nm 工艺，之后 10 nm 的研究和开发一直就问题不断，多次延迟，不过好消息是 AMD 7 nm 的显卡和 CPU 已经上市。因为摩尔定律不是真正的定律，只是一种预测或推测。尽管芯片制造商一直致力于实现并保持目标，但这样做变得越来越困难。

5. 量子计算

随着电子元件越来越小（纳米级），量子特性和效应逐渐显现。随着晶体管尺寸不断缩小，其 PN 结耗尽层的尺寸也越来减小。耗尽层非常重要，其用于阻止电子的流动。研究人员通过计算得出，由于电子在其耗尽区中的隧道效应，小于 5 nm 的晶体管将无法阻止电子流动。由于隧穿，电子将不会感知耗尽区域，直接“跨穿”。如果不能阻止电子流动，晶体管就会失效。此外，我们现在正在慢慢接近原子本身的大小，理论上我们不能建立一个比原子小的晶体管。硅原子的直径约为 1 nm，现在我们制造的晶体管的栅极尺寸约为该尺寸的 10 倍。就算是不考虑量子效应的，我们也将达到晶体管的物理极限，无法做到更小。

未来最可能的解决方案是发展量子计算（Quantum Computers）。像 D-Wave 和 Rigetti Computing 这样的公司正在这个领域广泛开展工作，量子计算已经显示出巨大的前景，它的优势是一次可以拥有多个状态（与其他计算机“0”“1”不同）。目前，有些实验性质的量子计算已经取得很好的成果，比如基于量子技术的真正的随机数算法已经成功。

1.2 EDA 技术简介

随着微电子技术的不断进步，大规模集成电路加工技术的不断提高，即半导体工艺技术的不断提高，现代电子设计技术取得了长足发展。20 世纪 90 年代，电子设计自动化

（Electronic Design Automation，EDA）技术的出现，极大地提高了现代电子系统设计的效率和可靠性，使功能多样化、体积小型化、功耗最低化的当代电子系统设计要求得以满足，EDA 技术也成为现代电子设计技术的核心。

1.2.1 EDA 技术概念

EDA 技术作为现代电子设计技术的核心，它是以微电子技术为物理层面，计算机软件技术为手段，实现集成电子系统或专用集成电路 ASIC（Application Specific Integrated Circuit）设计的一门新兴技术，其最终目标是实现 ASIC 的设计。

EDA 技术从概念上有狭义和广义之分，狭义的 EDA 技术，是指以可编程逻辑器件（Programmable Logic Device，PLD）为设计载体，以硬件描述语言（Hardware Description Language，HDL）为系统逻辑描述的主要表达方式完成设计文件，在计算机及 EDA 软件工具平台上，自动完成系统逻辑编译、化简、分割、综合、优化和仿真，直至下载到可编程逻辑器件，最终实现既定的电子电路设计技术。

EDA 技术主要面向两大类人员使用，一类是专用集成电路 ASIC 的芯片设计研发人员；另一类是广大的电子线路设计人员，即不具备集成电路深层次知识的设计人员。本书所阐述的 EDA 技术以后者为应用对象，这样，EDA 技术可以简单理解为以大规模可编程逻辑器件为设计载体，设计者用硬件描述语言来编写设计文件，并输入给相应的 EDA 开发软件，经过编译和仿真，最终下载到设计载体中，完成系统电路设计任务的一门新技术。

利用 EDA 技术进行电子系统设计，它具有以下几个特点：

（1）用软件的方式设计硬件。

（2）用软件方式设计的系统到硬件系统的转换是由相关的开发软件自动完成的。

（3）设计过程中可用相关软件进行各种仿真测试。

（4）系统可现场编程、在线升级。

（5）整个系统可集成在一个芯片上，体积小、功耗低、可靠性高。

综上所述，EDA 技术将是现代电子设计的发展趋势。

广义的 EDA 技术，除了狭义的 EDA 技术内容外，还包括计算机辅助分析技术（如 MATLAB、EWB 等），计算机辅助设计 CAD（Computer Assist Design）技术（如 Protel、OrCAD 等）。计算机辅助分析和计算机辅助设计均不具备逻辑综合和逻辑适配的功能，因此它们并不能成为严格意义上的 EDA 技术。EDA 技术应该包含三个层次的内容，首先是 EWB、Protel 的学习作为 EDA 的最初级内容；其次是利用 HDL 完成对大规模可编程逻辑器件的开发作为 EDA 的中级内容；最后以 ASIC 的设计作为 EDA 技术的最高级内容。

1.2.2 EDA 技术发展历史

EDA 技术作为现代电子设计技术的核心，不仅在硬件实现方面融合了大规模集成电路制造技术、IC 版图设计技术、ASIC 测试和封装技术、FPGA（Field Programmable Gate Array）/CPLD（Complex Programmable Logic Device）编程下载技术、自动测试技术等；在计算机辅助工程方面融合了计算机辅助设计（CAD）、计算机辅助制造（CAM）、计算机辅助测试（CAT）、计算机辅助工程（CAE）技术以及多种计算机语言的设计概念；还包含了现代电子学中的电子线路设计理论、数字信号处理技术、数字系统建模和优化技术以及基于微波技术的长线技术理论等。因此，EDA 技术不再是某一单一学科的分支，或某种新的技能技术，而是一门综合性学科。它融合多学科于一体，又渗透于各学科之中，模糊了传统软件和硬件间的界限，使计算机软件技术与硬件实现、设计效率和产品性能合二为一，真正代表了电子设计技术和应用技术的发展方向。有专家预言，EDA 技术将会是对 21 世纪产生重大影响的十大科学技术之一。

正因为 EDA 技术丰富的内容以及与电子技术各个学科领域的相关性，其发展历程与计算机、集成电路、电子系统设计的发展是同步的，主要经历了计算机辅助设计（CAD）、计算机辅助工程（CAE）设计和电子系统设计自动化（EDA）3 个阶段。

1. 计算机辅助设计（CAD）阶段

20 世纪 70 年代，随着中、小规模集成电路的出现和应用，传统的手工制图设计印制电路板和集成电路的方法已无法满足设计精度和效率的要求，人们开始利用计算机取代手工劳动，辅助进行集成电路版图编辑、PCB 布局布线等工作，这就产生了第一代 EDA 工具。受当时计算机技术的制约，能支持的设计工作有限且性能也比较差。

2. 计算机辅助工程（CAE）阶段

20 世纪 80 年代，第一个个人工作站（Apollo）计算机平台的出现，推动了 EDA 工具的迅速发展。为了满足电子产品在规模和制作上的需求，出现了以计算机仿真和自动布线为核心技术的第二代 EDA 技术。具有自动综合能力的 CAE 工具代替了设计师的部分设计工作，实现了以软件工具为核心，通过这些软件完成产品开发的设计、分析、生产、测试等各项工作。而在 80 年代末，出现了 FPGA，CAE 和 CAD 技术的应用更为广泛，它们在 PCB 设计方面的原理图输入、自动布局布线及 PCB 分析，以及逻辑设计、逻辑仿真、布尔方程综合和化简等方面担任了重要的角色，特别是各种硬件描述语言的出现及其在应用和标准化方面的重大进步，为电子设计自动化必须解决的电路建模、标准文档及仿真测试奠定了基础。

3. 电子系统设计自动化（EDA）阶段

20 世纪 90 年代，设计师们逐步从使用硬件转向设计硬件，从电路级电子产品开发转向系统级电子产品开发。随着硬件描述语言的标准化得到进一步的确立，计算机辅助工程、辅助分析、辅助设计在电子技术领域获得了更加广泛的应用。与此同时，集成电路

设计工艺的高速发展，已经步入了超深亚微米阶段，近千万门以上的大规模可编程逻辑器件的陆续面世，以及基于计算机技术的面向用户的低成本大规模 ASIC 设计技术的应用，促进了真正 EDA 技术的形成。

进入 21 世纪，随着各 EDA 公司致力于推出兼容各种硬件实现方案和支持标准硬件描述语言的 EDA 工具软件的研究，EDA 技术正向着功能强大、简单易学、使用方便的方向发展。新一代 EDA 开发工具的发布，新一代大规模可编程逻辑器件的问世以及价格的不断降低，使得 EDA 技术正在逐步走进人们生活中的各个领域。

1.2.3 EDA 技术设计流程

运用 EDA 技术对 FPGA/CPLD 进行开发设计的一般流程如图 1.2 所示，主要包括设计输入（原理图/HDL 文本编辑）、设计处理（编译和检查、综合、适配）、仿真、编程下载、硬件测试等几个阶段。整个设计过程，基本上都在 EDA 软件平台上完成，每个阶段都有相应的基于计算机环境的 EDA 工具的支持。常用 EDA 工具大致可以分为 5 个模块：设计输入编辑器、HDL 综合器、适配器、仿真器、下载器。

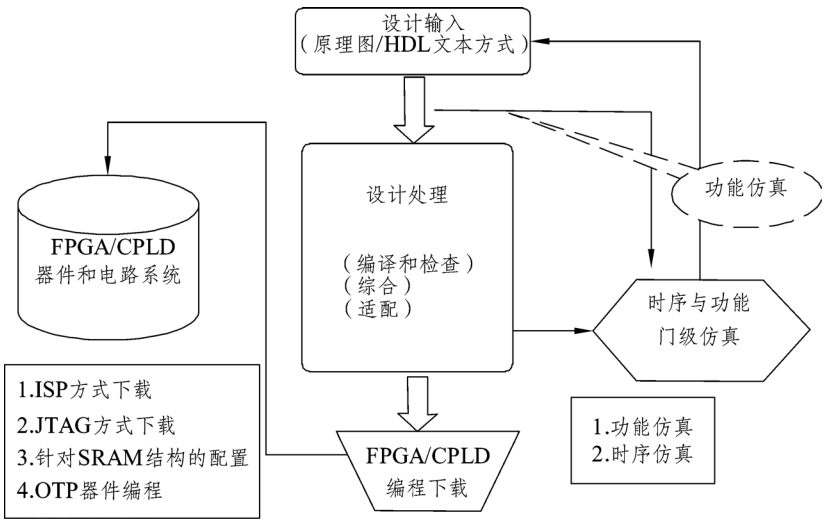


图 1.2　EDA 设计流程

1. 设计输入

从图 1.2 中可以看出，任何一项工程设计均是从设计输入开始，其功能是用一定的逻辑表达手段将设计表达出来，输入计算机及 EDA 开发工具，为后续设计的处理提供一个设计基础。通常，设计输入时使用的 EDA 工具主要包含图形编辑器和文本编辑器两种类型，即设计者可以用图形方式或文本方式将设计表达出来，为逻辑综合做准备。

（1）图形输入。

图形输入通常包含原理图输入、状态图输入和波形图输入三种方式。

原理图输入方法：利用 EDA 工具提供的图形编辑器以原理图的方式进行的输入。这是一种最直接的设计输入方式，它使用软件系统提供的元器件库及各种符号和连线画出设计电路的原理图，原理图由逻辑器件和连线构成，这些器件包括类似与门、或门、非门、触发器、74 系列器件和类似 IP 的功能块等。原理图输入的优点是比较容易掌握，直观且方便，绘图方法类似于 Protel 原理图的绘制（但这种原理图和 Protel 画的原理图有本质的区别），设计者可以利用基本的数字电路的知识便可以进行电子线路在 FPGA 中的实现，而不需要增加诸如 HDL 等新的相关知识。然而，原理图输入法的优点同时也成为它的缺点：① 随着设计规模增大，设计的易读性迅速下降，对于原理图中密密麻麻的电路连线，极难弄清电路的实际功能。② 原理图一旦完成，电路结构的改变将十分困难，因此几乎没有可以再利用的设计模块。③ 由于图形文件的不兼容性，电路模块的移植困难、入档困难、交流困难，因为缺乏一个统一的标准化的原理图编辑器。④ 由于在原理图中已经确定了设计系统的基本电路结构和元件，留给综合器和适配器的优化选择空间十分有限，难以实现面积、速度以及不同风格的综合优化。因此，原理图输入方法大多用于设计者对系统及各个部分电路很熟悉的情况。

状态图输入方法：主要是利用 EDA 工具的状态图编辑器，用绘制状态流程图的方式进行输入。当设置好时钟信号名、状态转换条件、状态机类型等要素后，EDA 编译器和综合器就能将此状态变化流程图编译综合成电路网表，还可以自动生成 VHDL 程序。这种设计方法可以简化状态机的设计难度，常常用于状态机电路的设计。

波形图输入方法：是将待设计的电路看成是一个黑盒子，在波形编辑器中，只需告诉 EDA 工具黑盒子电路的输入和输出时序波形图，EDA 工具便可以据此完成相应功能电路的设计。波形图输入主要用于建立和编辑波形设计文件，输入仿真向量和功能测试向量，适用于时序逻辑和有重复性的逻辑函数。

（2）文本输入。

文本输入是指在 EDA 工具的相应文本编辑器中，将使用了某种硬件描述语言的电路设计文本，如 VHDL 或 Verilog HDL 的源程序代码，进行编辑输入的方式。这种方式与传统的软件语言编辑输入大同小异，只是这里的硬件描述语言是设计硬件电路，而非软件语言的程序设计。文本输入方法可以克服原理图输入方法所存在的所有弊端，它是 EDA 技术中最基本、最有效和最通用的输入方法。

实际上，在 EDA 技术的实际应用中，文本输入和原理图输入是可以混合使用的，即在原理图中的底层元件符号，可以用文本方式设计完成；顶层设计一般均采用原理图设计输入方式。

2. 设计处理

设计处理是 EDA 设计中的核心环节，该过程智能化、自动化的实现从设计输入到最终硬件实现的全过程。在设计处理阶段，编译软件将对设计输入文件进行逻辑化简、综

合、优化并根据设计者选择的具体器件自动进行适配，最终产生一个编程文件。设计处理主要包含设计编译和检查语法错误、逻辑综合、适配、布局和布线、生成编程数据文件等过程。

（1）编译和检查。

设计输入完成后，首先进行编译，在编译过程中进行语法错误检查，如检查原理图的信号线有无漏接，信号线端口名是否有重复，文本程序代码的关键字有无错误，使用语句结构是否规范，是否符合相应语言的语法规则等等。如果有错，会及时标出错误的位置，供设计者修改、纠正。

（2）综合。

经过编译和检查无误的设计输入文件，便可以进入到综合阶段。所谓综合（Synthesis），就是将电路的高级语言（如行为描述、原理图或状态图描述）转换为低级的，可与 FPGA/CPLD 的门阵列基本结构相映射的网表文件或程序。逻辑映射的过程就是将电路的高级描述，针对给定硬件结构组件，进行编译、优化、转换和综合，最终获得门级电路甚至更低层次的电路描述文件。

实际上，综合功能完全是在 EDA 工具——综合器中完成的。显然，综合器就是能够自动将一种设计表述形式向另一种设计表述形式转换的计算机程序，或协助进行手工转换的程序。它可以将高层次的表述转换为低层次的表述，可以将用行为表述的设计文件转换为具体的电路结构，可以将高一级抽象的电路（如算法级）转换为低一级的门级电路，并可以用相应的技术进行实现。在这里，综合的整个过程实际上共分为四步来完成：

① 将自然语言转换到 VHDL 语言算法，即自然语言综合。

② 从算法表示转换到寄存器传输级（Register Transport Level，RTL），即从行为域到结构域的综合——行为综合。

③ 从 RTL 级表示转换到逻辑门（包含触发器）的表示，即逻辑综合。

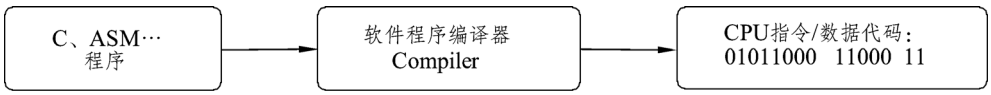
④ 从逻辑门表示转换到版图表示，或转换到 FPGA 的配置网表文件，即结构综合。

为了能更好地理解综合的过程和功能，可以将其与传统的软件程序编译器进行对比。从表面上看，我们熟悉的软件程序代码到可执行文件的产生，是通过编译器完成的，其过程可以理解为软件程序语言到机器语言的“翻译”过程；综合器和编译器都不过是一种“翻译器”，它们都能将高层次的设计表达转换为低层次的表达，但它们却又有着本质的区别，如图 1.3 所示。

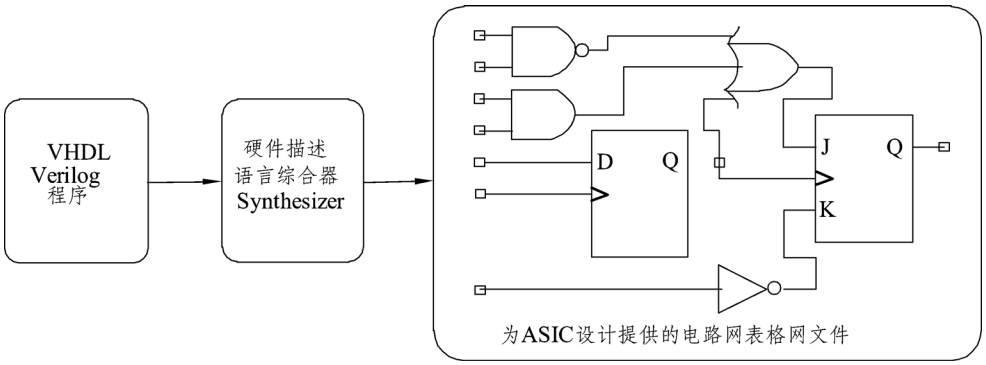
编译器和综合器的主要区别在于编译器是将软件程序翻译成为基于某种特定 CPU 的机器代码，这种二进制代码流仅限于 CPU 能识别而不能移植，并且机器代码不代表硬件结构，属于纯软件开发设计。而综合器则是将硬件描述语言程序代码翻译为一具体的网表文件，这种网表文件代表了特定的硬件结构，因此，EDA 技术中的程序代码属于硬件设计。

（3）适配/实现。

适配是在适配器中完成的，所谓适配就是将由综合器产生的网表文件针对某一具体的目标器件进行逻辑映射操作，即将网表文件配置于指定的目标器件中，使之产生最终的下载文件，如 JEDEC、JAM 格式文件。适配器完成逻辑映射工作具体包含底层器件配置、逻辑分割、逻辑优化、逻辑布局布线等。适配所选用的目标器件（FPGA/CPLD）必须属于原综合器指定的目标器件系列，适配完成后可以利用适配所产生的仿真文件作精确的时序仿真。



（a）软件语言设计流程



（b）硬件语言设计流程

图 1.3　编译器和综合器的功能比较

3. 设计仿真

在编程下载之前往往需要利用 EDA 的仿真工具对适配生成的结果进行模拟测试，这就是所谓的仿真。仿真是 EDA 设计过程中的重要步骤，其过程和实现原理是让计算机根据一定的算法和一定的仿真库对 EDA 设计进行模拟，以验证设计，排除错误。设计过程中的仿真主要有功能仿真和时序仿真。

所谓功能仿真，就是直接对 VHDL/Verilog、原理图描述或其他描述形式的逻辑功能进行模拟测试，以了解其实现的功能是否满足原设计的要求。仿真过程中不涉及任何具体器件的硬件特性（如延时特性），不经历综合与适配阶段，在设计项目编辑编译后即可进入门级仿真器进行模拟测试。功能仿真的好处是设计耗时短，对硬件库、综合器等没有任何要求。

所谓时序仿真，就是将适配器所产生的网表文件送入到仿真器中进行的仿真。这时的仿真是最接近于真实器件运行特性的仿真，因为仿真中已经包含了器件硬件特性参数，因此可以得到精确的时序仿真结果。但时序仿真的仿真文件必须来自针对具体器件的综合器与适配器，综合后所得的 EDIF 等网表文件通常作为 FPGA 适配器的输入文件，产生的仿真网表文件中包含了精确的硬件延迟信息。

对于大规模的设计项目而言，综合和适配在计算机上的耗时是十分可观的，如果每一次修改设计后都进行时序仿真，显然会极大地降低开发效率。因此，通常需要先进行功能仿真，待确定设计文件所表达的功能满足原设计意图时，再进行综合、适配和时序仿真，以提高设计的效率。

4. 编程下载

如果经过编译、综合、适配和仿真等过程都没有问题，即能满足设计的要求，这时便可以把适配后生成的下载或配置文件，通过编程器或编程电缆向 FPGA 或 CPLD 下载，以便进行硬件调试和验证，这就是 EDA 的编程下载。

一般而言，对 CPLD 的下载称为编程（Program），对 FPGA 中的 SRAM 进行直接的下载称为配置（Configure）。这主要取决于所选用器件的结构所决定。

5. 硬件测试

设计的最后便是硬件测试，所谓硬件测试，就是将载入了设计的 FPGA/CPLD 直接用于应用系统中，以便最终验证设计项目在目标系统上的实际工作情况，排除错误，改进设计。

1.3　硬件描述语言（HDL）

1.3.1 硬件描述语言概述

所谓硬件描述语言（Hardware Description Language，HDL），是指专门用于对电子系统硬件行为描述、结构描述、数据流描述的语言。利用这种语言，数字电路系统的设计可以从顶层到底层（从抽象到具体）逐层描述自己的设计思想，用一系列分层次的模块来表示极其复杂的数字系统。然后，利用电子设计自动化（EDA）工具，逐层进行仿真验证，再把其中需要变为实际电路的模块组合，经过自动综合工具转换到门级电路网表。接下来，再用专用集成电路 ASIC 或现场可编程门阵列（FPGA）自动布局布线工具，把网表转换为要实现的具体电路布线结构。

硬件描述语言（HDL）的发展至今已有近 30 年的历史，并成功地应用于设计的各个阶段：建模、仿真、验证和综合等。到 20 世纪 80 年代，已出现了上百种硬件描述语言，对设计自动化曾起到了极大的促进和推动作用。但是，这些语言一般各自面向特定的设计领域和层次，而且众多的语言使用户无所适从。因此，急需一种面向设计的多领域、多层次并得到普遍认同的标准硬件描述语言。20 世纪 80 年代后期，VHDL 和 Verilog HDL 语言适应了这种趋势的要求，先后成为 IEEE 标准。

HDL 一般可用于系统仿真和硬件实现。如果只用于仿真，那么几乎所有的语法和编程方法都可以使用。但如果是用于硬件实现（在 FPGA/CPLD 中），那么就必须保证程序“可综合”（程序的功能可以用硬件电路实现）。不可综合的 HDL 语句在软件综合时将被忽略或者报错。应当牢记一点：所有的 HDL 描述都可以用于仿真，但不是所有的 HDL 描述都能用硬件实现。

硬件描述语言（HDL）的使用与其他的高级语言相似，编写的代码也需要首先经过编译器进行语法、语义的检查，并转换为某种中间数据格式。但与其他高级语言不同之处在于，用硬件描述语言编写程序的最终目的是要生成实际的硬件，经相关软件工具处理后，最终得到的是一个硬件电路。

1.3.2 Verilog HDL

1. Verilog HDL 概述

Verilog HDL 是目前应用最为广泛的一种硬件描述语言，用于从算法级、门级到开关级的多种抽象设计层次的数字系统建模。被建模的数字系统对象的复杂性可以介于简单的门和完整的电子数字系统之间。数字系统能够按层次描述，并可在相同描述中显式地进行时序建模。

Verilog HDL 语言具有下述描述能力：设计的行为特性、设计的数据流特性、设计的结构组成以及包含响应监控和设计验证方面的时延和波形产生机制。此外，Verilog HDL 语言提供了编程语言接口，通过该接口可以在模拟、验证期间从设计外部访问设计，包括模拟的具体控制和运行。

Verilog HDL 语言不仅定义了语法，而且对每个语法结构都定义了清晰的模拟、仿真语义。因此，用这种语言编写的模型能够使用 Verilog 仿真器进行验证。语言从 C 编程语言中继承了多种操作符和结构。Verilog HDL 提供了扩展的建模能力，其中许多扩展最初很难理解。但是，Verilog HDL 语言的核心子集非常易于学习和使用，这对大多数建模应用来说已经足够。当然，完整的硬件描述语言足以对从最复杂的芯片到完整的电子系统进行描述。

Verilog 的设计初衷是成为一种基本语法与 C 语言相近的硬件描述语言。这是因为 C 语言在 Verilog 设计之初，已经在许多领域得到广泛应用，C 语言的许多语言要素已经被许多人习惯。一种与 C 语言相似的硬件描述语言，可以让电路设计人员更容易学习和接受。不过，Verilog 与 C 语言还是存在许多差别。另外，作为一种与普通计算机编程语言不同的硬件描述语言，Verilog 还具有一些独特的语言要素，如向量形式的线网和寄存器、过程中的非阻塞赋值等。总的来说，具备 C 语言的设计人员将能够很快掌握 Verilog 硬件描述语言。

2. Verilog HDL 发展历史

Gateway 设计自动化公司的工程师菲尔·莫比（Phil Moorby）于 1983 年末完成了 Verilog 的主要设计工作。

20 世纪 90 年代初，开放 Verilog 国际（Open Verilog International，OVI）组织（即现在的 Accellera）成立，Verilog 面向公有领域开放。1992 年，该组织寻求将 Verilog 纳入电气电子工程师学会标准。最终，Verilog 成为电气电子工程师学会 IEEE 1364-1995 标准，即通常所说的 Verilog-95。

设计人员在使用这个版本的 Verilog 的过程中发现了一些可改进之处。为了解决用户在使用此版本 Verilog 过程中反映的问题，Verilog 进行了修正和扩展，这部分内容后来再次被提交给电气电子工程师学会。这个扩展后的版本后来成为电气电子工程师学会 1364-2001 标准，即通常所说的 Verilog-2001。Verilog-2001 是 Verilog-95 的一个重大改进版本，它具备一些新的实用功能，如敏感列表、多维数组、生成语句块、命名端口连接等。目前，Verilog-2001 是 Verilog 的最主流版本，被大多数商业电子设计自动化软件包支持。

2005 年，Verilog 再次进行了更新，即电气电子工程师学会 1364-2005 标准。该版本只是对上一版本的细微修正。这个版本还包括了一个相对独立的新部分，即 Verilog-AMS。这个扩展使得传统的 Verilog 可以对集成的模拟和混合信号系统进行建模。容易与电气电子工程师学会 1364-2005 标准混淆的是加强硬件验证语言特性的 SystemVerilog（电气电子工程师学会 1800-2005 标准），它是 Verilog-2005 的一个超集，它是硬件描述语言、硬件验证语言（针对验证的需求，特别加强了面向对象特性）的一个集成。

2009 年，IEEE 1364-2005 和 IEEE 1800-2005 两个部分合并为 IEEE 1800-2009，成为一个新的、统一的 SystemVerilog 硬件描述验证语言（hardware description and verification language，HDVL）。

Verilog HDL 是在用途最广泛的 C 语言的基础上发展起来的，其最大特点就是易学易用，如果有 C 语言的编程经验，可以在一个较短的时间内很快地学习和掌握。

3. Verilog HDL 典型结构

```
`timescale 1ns / 1ps                    //定义时间刻度，不可综合
module  模块名 （端口列表）;             //定义模块（实体）
     端口声明;
     变量声明;
     assign 语句;                       //数据流描述语句结构
     always @（posedge clk）            //行为描述语句结构
       begin
            if (表达式)  语句;
       end
     元件例化语句;                      //电路结构描述语句
endmodule
```

1.3.3 VHDL

超高速集成电路硬件描述语言（Very High Speed Integrated Circuit Hardware Description Language，VHDL）最早是由美国国防部（DOD）发起创建，于 1985 年正式推出，通过 IEEE（The Institute of Electrical and Electronics Engineers）进一步发展，于 1987 年将 VHDL 采纳为 IEEE 1076 标准发布。从此，VHDL 成为硬件描述语言的业界标准之一，也是目前标准化程度最高的硬件描述语言。1993 年 IEEE 对 VHDL 进行了修订，增加了部分新的命令与属性，增强了对系统的描述能力，并公布了新版本的 VHDL，即 IEEE 1076-1993 版本。VHDL 经过近 30 年的发展、应用和完善，以其强大的系统描述能力、规范的程序设计结构、灵活的语言表达风格和多层次的仿真测试手段，在电子设计领域得到了普遍的认同和广泛的接受，已经成为现代 EDA 领域的首选硬件描述语言。

VHDL 作为一个规范语言和建模语言，涵盖面广，抽象描述能力强，能从多个层次对数字系统进行建模和描述，大大简化了硬件设计任务，提高了设计效率和可靠性。VHDL 的基本结构至少包含一个实体和一个结构体，而完整的 VHDL 结构还应包含配置和程序包与库。在应用 VHDL 进行复杂电路设计时，往往采用“自顶向下”结构化的设计方法。其典型的结构如下所示。

```
Library ieee;
use ieee.std_logic_1164.all;          --库声明

ENTITY  实体名  IS                    --实体定义
   port(端口列表);
END  ENTITY  实体名 ;
ARCHITECTURE  结构体名 of  实体名 IS   --结构体定义
 [信号申明；]
begin
      功能描述语句;
END ARCHITECTURE  结构体名;
```

在设计中是选择 VHDL 还是选择 Verilog HDL？这是一个初学者最常见的问题。其实两种语言的差别并不大，它们的描述能力也是类似的。比较而言，VHDL 是一种高级描述语言，适用于电路高级建模，综合的效率和效果较好。Verilog HDL 是一种低级的描述语言，适用于描述门级电路，容易控制电路资源，但其对系统的描述能力不如 VHDL。只要掌握其中一种语言以后，可以通过短期的学习，较快地学会另一种语言。选择何种语言主要还是看周围人群的使用习惯，这样可以方便日后的学习交流。当然，如果您是

集成电路（ASIC）设计人员，则必须首先掌握 verilog，因为在 IC 设计领域，90%以上的公司都是采用 verilog 进行 IC 设计。对于 CPLD/FPGA 设计者而言，两种语言可以自由选择。

1.4　FPGA 结构及工作原理

可编程逻辑器件（Programmable Logic Device，PLD）是一种半定制集成电路，在其内部集成了大量的门和触发器等基本逻辑单元电路，通过用户编程来改变 PLD 内部电路的逻辑关系或连线，从而得到所需要的电路设计功能。这种新型逻辑器件，不仅速度快、集成度高，能够完成用户定义的逻辑功能，还可以加密和重新定义编程，其允许编程次数可以达到上万次。可编程逻辑器件的出现，大大改变了传统数字系统设计方法，简化了硬件系统，降低了成本、提高系统的可靠性、灵活性。因此，自 20 世纪 70 年代问世以后，PLD 就受到广大工程师的青睐，被广泛应用于工业控制、通信设备、仪器仪表和医疗电子仪器等众多领域，为 EDA 技术开创了广阔的发展空间。

常见的 PLD 主要包括 FPGA（Field Programmable Gate Array，现场可编程门阵列）和 CPLD（Complex Programmable Logic Device，复杂可编程逻辑器件）两大类。FPGA 和 CPLD 最明显的特点是高集成度、高速度和高可靠性。高速度表现在其时钟延时可小至纳秒级，结合并行工作方式，广泛应用于超高速领域和实时测控方面；高可靠性和高集成度表现在几乎可以将整个系统集成于一个芯片中，实现所谓片上系统 SOC（System On a Chip）。

对于一个开发项目，究竟是选择 FPGA 还是选择 CPLD 呢？主要看开发项目本身的需要。对于一般规模，且产量不是很大的项目，通常选择 CPLD 较好；对于大规模的 ASIC 设计或片上系统设计，则多选择 FPGA。另外，FPGA 掉电后将丢失原有的逻辑信息，在实际应用中，往往需要为 FPGA 芯片配置一个专用的 ROM，CPLD 掉电后不会丢失数据。

1.4.1　FPGA 概述

FPGA（Field Programmable Gate Array）于 1985 年由 Xilinx 创始人之一 Ross Freeman 发明，虽然有其他公司宣称自己最先发明可编程逻辑器件 PLD，但是真正意义上的第一颗 FPGA 芯片 XC2064 为 Xilinx 所发明，这个时间差不多比摩尔提出著名的摩尔定律晚 20 年左右，但是 FPGA 一经发明，后续的发展速度之快，超出大多数人的想象，近些年的 FPGA，始终引领先进的工艺。

FPGA 的主要应用领域是通信、工控、国防、消费，近年来 FPGA 最引人关注的变化趋势之一就是应用领域不断扩展。

在 FPGA 传统应用市场方面，通信逐步实现高速、复杂协议，消费电子应用则注重低功耗、低成本。此外，FPGA 还广泛应用于医疗电子、安防、视频、工业自动化、语音

网络、半导体制造设备以及家电等领域。

1.4.2 FPGA 内部结构

FPGA 的基本结构都是基于查找表加寄存器结构的，Xilinx、Altera、Lattice、Actel 和 Atmel 公司都是知名的 FPGA 供应商，这些厂商的 FPGA 产品的基本构架都可简化为 6 个部分，分别为可编程输入/输出单元、可编程逻辑块（CLB）、嵌入式块 RAM、丰富的布线资源、底层嵌入式功能单元和内嵌专用硬核等。

1. 可编程输入/输出单元（IOB）

可编程输入/输出（Input/Output）单元简称 I/O 单元，是芯片与外界电路的接口部分，完成不同电气特性下对输入/输出信号的驱动与匹配要求。FPGA 内的 I/O 按组分类，每组都能够独立地支持不同的 I/O 标准。通过软件的灵活配置，可适配不同的电气标准与 I/O 物理特性，可以调整驱动电流的大小，可以改变上、下拉电阻。目前，I/O 口的频率也越来越高，一些高端的 FPGA 通过 DDR 寄存器技术可以支持高达 2 Gb/s 的数据速率。

外部输入信号可以通过 IOB 模块的存储单元输入到 FPGA 的内部，也可以直接输入 FPGA 内部。当外部输入信号经过 IOB 模块的存储单元输入到 FPGA 内部时，其保持时间（Hold Time）的要求可以降低，通常默认为 0。为了便于管理和适应多种电器标准，FPGA 的 IOB 被划分为若干个组（bank），每个组的接口标准由其接口电压 VCCO 决定，一个组只能有一种 VCCO，但不同组的 VCCO 可以不同。只有相同电气标准的端口才能连接在一起，VCCO 电压相同是接口标准的基本条件。

2. 可编程逻辑块（CLB）

CLB 是 FPGA 内的基本逻辑单元。CLB 的实际数量和特性会依器件的不同而不同，但是每个 CLB 都包含一个可配置开关矩阵，此矩阵由 4 或 6 个输入、一些选型电路（多路复用器等）和触发器组成。开关矩阵是高度灵活的，可以对其进行配置以便处理组合逻辑、移位寄存器或 RAM。

Slice 是 Xilinx 公司定义的基本逻辑单位。一个 Slice 由两个 4/6 输入 LUT、进位逻辑、算术逻辑（一个异或门 XORG，一个与门 MULTAND）、存储逻辑和函数复用器组成。

算术逻辑包括一个异或门（XORG）和一个专用与门（MULTAND），一个异或门可以使一个 Slice 实现 2 bit 全加操作，专用与门用于提高乘法器的效率；进位逻辑由专用进位信号和函数复用器（MUXC）组成，用于实现快速的算术加减法操作；4 输入函数发生器用于实现 4 输入 LUT、分布式 RAM 或 16 比特移位寄存器（Virtex-5 系列芯片的 Slice 中的两个输入函数为 6 输入，可以实现 6 输入 LUT 或 64 比特移位寄存器）；进位逻辑包括两条快速进位链，用于提高 CLB 模块的处理速度。

3. 嵌入式块 RAM（BRAM）

多数 FPGA 都具有内嵌的块 RAM，这大大拓展了 FPGA 的应用范围和灵活性。块

RAM 可被配置为单端口 RAM、双端口 RAM、内容地址存储器（CAM）以及 FIFO 等常用存储结构。CAM 存储器在其内部的每个存储单元中都有一个比较逻辑，写入 CAM 中的数据会和内部的每一个数据进行比较，并返回与端口数据相同的所有数据的地址，因而在路由的地址交换器中有广泛的应用。除了块 RAM，还可以将 FPGA 中的 LUT 灵活地配置成 RAM、ROM 和 FIFO 等结构。在实际应用中，芯片内部块 RAM 的数量也是选择芯片的一个重要因素。

单片块 RAM 的容量为 18k 比特，即位宽为 18 比特、深度为 1 024，可以根据需要改变其位宽和深度，但要满足两个原则：首先，修改后的容量（位宽、深度）不能大于 18k 比特；其次，位宽最大不能超过 36 比特。当然，可以将多片块 RAM 级联起来形成更大的 RAM，此时只受限于芯片内块 RAM 的数量，而不再受上面两条原则约束。

4. 丰富的布线资源

布线资源连通 FPGA 内部的所有单元，而连线的长度和工艺决定着信号在连线上的驱动能力和传输速度。FPGA 芯片内部有着丰富的布线资源，根据工艺、长度、宽度和分布位置的不同而划分为 4 类不同的类别。第一类是全局布线资源，用于芯片内部全局时钟和全局复位/置位的布线；第二类是长线资源，用以完成芯片组间的高速信号和第二全局时钟信号的布线；第三类是短线资源，用于完成基本逻辑单元之间的逻辑互连和布线；第四类是分布式的布线资源，用于专有时钟、复位等控制信号线。

在实际中设计者不需要直接选择布线资源，布局布线器可自动地根据输入逻辑网表的拓扑结构和约束条件选择布线资源来连通各个模块单元。从本质上讲，布线资源的使用方法和设计的结果有密切、直接的关系。

5. 底层内嵌功能单元

内嵌功能模块主要指 DLL（Delay Locked Loop）、PLL（Phase Locked Loop）、DSP 和 CPU 等软处理核（SoftCore）。

现在越来越丰富的内嵌功能单元，使得单片 FPGA 成为系统级的设计工具，使其具备了软硬件联合设计的能力，逐步向 SOC 平台过渡。DLL 和 PLL 具有类似的功能，可以完成时钟高精度、低抖动的倍频和分频，以及占空比调整和移相等功能。赛灵思公司生产的芯片上集成了 DCM 和 DLL，Altera 公司的芯片集成了 PLL，Lattice 公司的新型芯片上同时集成了 PLL 和 DLL。PLL 和 DLL 可以通过 IP 核生成的工具方便地进行管理和配置。

1.4.3 FPGA 工作原理

由于 FPGA 需要被反复烧写，它实现组合逻辑的基本结构不可能像 ASIC 那样通过固定的与非门来完成，而只能采用一种易于反复配置的结构，查表（Look Up Tabler，LUT）可以很好地满足这一要求。目前，主流 FPGA 都采用了基于 SRAM 工艺的查找表结构，也有一些军品和宇航级 FPGA 采用 Flash/熔丝/反熔丝工艺的查找表结构。

由布尔代数理论可知，对于一个 N 输入的逻辑运算，最多产生 2^N 个不同的组合。所以，如果预先将相应的结果保存在一个存储单元中，就相当于实现了与非门电路的功能。

FPGA 的原理的实质，就是通过配置文件对查找表进行配置，从而在相同的电路情况下实现了不同的逻辑功能。

1. 4 输入查找表结构

LUT 本质就是一个 RAM，自 FPGA 诞生以来，它大多使用 4 输入的 LUT 结构，所以每个 LUT 可以看成一个包含四位地址线的 RAM。当设计者通过原理图或 HDL 描述了一个逻辑电路后，FPGA 厂商提供的集成开发工具就会自动计算逻辑电路的所有可能结果，并把真值表事先写入到 RAM 中。这样，每输入一个信号进行逻辑运算就等于输入一个地址进行查表，找出地址对应的内容，然后输出内容即可。

下面用一个 4 输入逻辑与门电路的例子来说明 LUT 实现组合逻辑的原理。LUT 描述四输入逻辑与关系见表 1.1。

表 1.1　4 输入与门的真值表

实际逻辑电路		LUT 实现方式	
A，B，C，D 输入	逻辑输出	RAM 地址	RAM 中存储内容
0000	0	0000	0
0001	0	0001	0
⋮			
1111	1	1111	1

从表 1 可以看到，LUT 具有和逻到电路相同的功能，但是 LUT 具有更快的执行速度和更大的规模。与传统化简真值表构造组合逻辑的方法相比，LUT 具有明显的优势，主要表现在：

（1）LUT 实现组合逻辑的功能由输入决定，而不是由复杂度决定。

（2）LUT 实现组合逻辑有固定的传输延迟。

2. 6 输入查找表结构

在 65 nm 工艺条件下，与其他电路（特别是互连电路）相比，LUT 的常规结构大大缩小。一个具有 4 倍比特位的 6 输入 LUT 结构仅仅将所占用的 CLB 面积增加了 15%，但是平均而言，每个 LUT 上可集成的逻辑数量却提高了 40%。当采用更高的逻辑密度时，通常可以降低级联 LUT 的数目，并且改进关键路径延迟性能。

新一代的 FPGA 提供了真正的 6 输入 LUT，可以将它用作逻辑或者分布式存储器，这时，LUT 是一个 64 位的分布式 RAM（甚至双端口或者四端口）或者一个 32 位可编程移位寄存器，每个 LUT 具有两个输出，从而实现了五个变量的两个逻辑函数，存储 32×2 RAM 比特，或者作为 16×2 比特的移位寄存器进行工作。