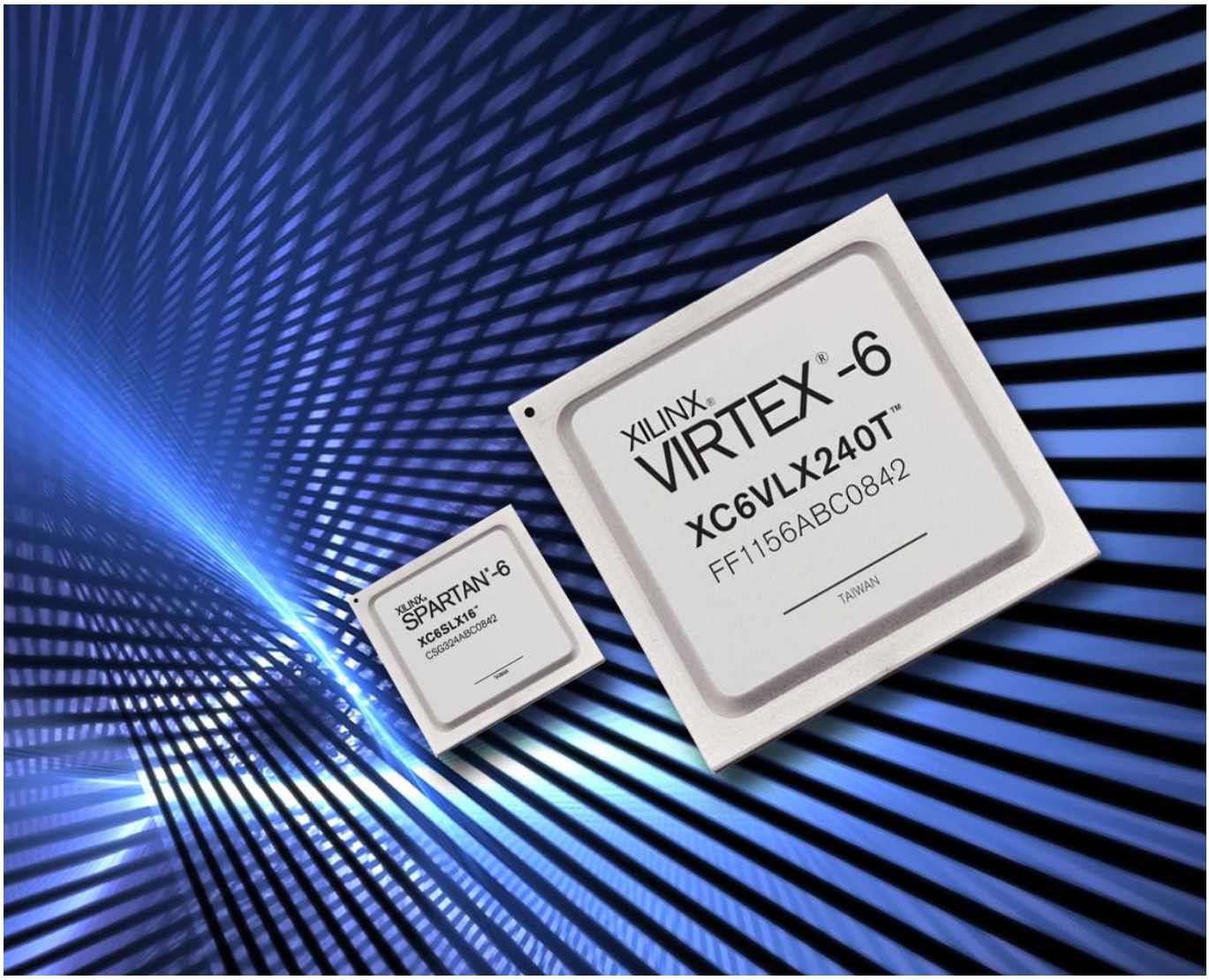
《电子工程师创新设计必备宝典系列之FPGA开发全攻略》



FPGA开发全攻略——

工程师创新设计宝典

上册

基础篇

2009年2月 1.0版

前言

张国斌
电子书主编
2009年2月25日

2008年，我参加了几次可编程器件供应商举办的技术研讨会，让我留下深刻印象的是参加这些研讨会的工程师人数之多，简直可以用爆满来形容，很多工程师聚精会神地全天听讲，很少出现吃完午饭就闪人的现象，而且工程师们对研讨会上展出的基于可编程器件的通信、消费电子、医疗电子、工业等解决方案也有浓厚的兴趣，这和其他器件研讨会形成了鲜明的对比。

Garnter和iSuppli公布的数据显示：2008年，全球半导体整体销售出现25年以来首次萎缩现象，但是，可编程器件却还在保持了增长，预计2008年可编程逻辑器件(PLD)市场销售额增长7.6%，可编程器件的领头羊美国供应商赛灵思公司2008年营业收入预计升6.5%！在全球经济危机的背景下，这是非常骄人的业绩！也足见可编程器件在应用领域的热度没有受到经济危机的影响！这可能也解释了为什么那么多工程师对可编程器件感兴趣吧。

在与工程师的交流中，我发现，很多工程师非常需要普及以FPGA为代表的可编程器件的应用开发知识，也有很多工程师苦于进阶无门，缺乏专业、权威性的指导，在Google上搜索后，我发现很少有帮助工程师设计的FPGA电子书，即使有也只是介绍一些概念性的基础知识，缺乏实用性和系统性，于是，我萌生了出版一本指导工程师FPGA应用开发电子书的想法，而且这个电子书要突出实用性，让大家都可以免费下载，并提供许多技巧和资源信息，很高兴美国赛灵思公司对这个想法给予了大力支持，赛灵思公司亚太区市场经理张俊伟小姐和高级产品经理梁晓明先生对电子书提出了宝贵的意见，并提供了大量FPGA设计资源，也介绍了一些FPGA设计高手参与了电子书的编撰，很短的时间内，一个电子书项目团队组建起来，北京邮电大学的研究生田耘先生和赛灵思公司上海办事处的苏同麒先生等人都参与了电子书的编写，他们是有丰富设计经验的高手，在大家的共同努力下，这本凝结着智慧的FPGA电子书终于和大家见面了！我希望这本电子书可以成为对FPGA有兴趣或正在使用FPGA进行开发的工程师的手头设计宝典之一，也希望这个电子书可以对工程师们学习FPGA开发和进阶有实用的帮助！如果可能，未来我们还将出版后续版本！

目 录

前言…… 2
第一章、为什么工程师要掌握FPGA开发知识？ ……5
第二章、FPGA基本知识与发展趋势 …… 7
2.1 FPGA结构和工作原理…… 7
2.1.1 梦想成就伟业…… 7
2.1.2 FPGA结构…… 8
2.1.3 软核、硬核以及固核的概念…… 15
2.1.4 从可编程器件发展看FPGA未来趋势…… 15
第三章、FPGA主要供应商与产品…… 17
3.1.1 赛灵思主要产品介绍…… 17
第四章、FPGA开发基本流程…… 29
4.1 典型FPGA开发流程与注意事项……29
4.2 基于FPGA的SOC设计方法……32
基于FPGA的典型SOC开发流程为 …… 32
第五章、FPGA实战开发技巧…… 33
5.1 FPGA器件选型常识……33
5.1.1器件的供货渠道和开发工具的支持 ……33
5.1.2 器件的硬件资源 ……33
5.1.3 电气接口标准 …… 34
5.1.4 器件的速度等级 …… 35
5.1.5 器件的温度等级 …… 35
5.1.6 器件的封装 ……35
5.1.7 器件的价格 ……35
5.2 如何进行FPGA设计早期系统规划 ……36
5.3. 综合和仿真技巧……37
5.3.1 综合工具XST的使用…… 37
5.3.2 基于ISE的仿真…… 42
5.3.3 和FPGA接口相关的设置以及时序分析…… 45
5.3.4 综合高手揭秘XST的11个技巧……51
5.4 大规模设计带来的综合和布线问题……52
5.5 FPGA相关电路设计知识……54

5.5.1 配置电路…… 54
5.5.2 主串模式——最常用的FPGA配置模式…… 56
5.5.3 SPI串行Flash配置模式…… 58
5.5.4 从串配置模式…… 62
5.5.5 JTAG配置模式…… 63
5.5.6 System ACE配置方案…… 64
5.6 大规模设计的调试经验……68
5.6.1 ChipScope Pro组件应用实例…… 68
5.7 FPGA设计的IP和算法应用……74
5.7.1 IP核综述…… 74
5.7.2 FFT IP核应用示例…… 75
5.8 赛灵思 FPGA的专用HDL开发技巧……79
5.8.1 赛灵思 FPGA的体系结构特点…… 79
5.8.2 赛灵思 FPGA 芯片专用代码风格…… 79
ISE与EDK开发技巧之时序篇……83
5.10 新一代开发工具ISE Design Suit10.1介绍……85
5.10.1 ISE Design Suit10.1综述…… 85
5.10.2 ISE Design Suit 10.1的创新特性…… 85
5.11 ISE与第三方软件的配合使用技巧……92
5.11.1 Synplify Pro软件的使用 ……92
5.11.2 ModelSim软件的使用…… 99
5.11.3 Synplify Pro、ModelSim和ISE的联合开发流程 …… 104
5.11.4 ISE与MATLAB的联合使用…… 105
5.12 征服FPGA低功耗设计的三个挑战 ……108
5.13 高手之路——FPGA设计开发中的进阶路线…… 111
附录一、FPGA开发资源总汇……112
附录二、编委信息与后记……113
附录三、版权声明……114

第一章、为什么工程师要掌握FPGA开发知识?

作者：张国斌、田耘

2008 年年初，某著名嵌入式系统 IT 公司为了帮助其产品售后工程师和在线技术支持工程师更好的理解其产品，举行了 ASIC/FPGA 基础专场培训. 由于后者因为保密制度而只能接触到板级电路图和 LAYOUT，同时因 ASIC/FPGA 都是典型的 SoC 应用，通常只是将 ASIC/FPGA 当作黑盒来理解，其猜测性读图造成公司与外部及公司内部大量的无效沟通. 培训结束后 , 参与者纷纷表示 ASIC/FPGA 的白盒式剖析极大提高了对产品的理解，有效解决了合作伙伴和客户端理解偏异性问题，参加培训的工程师小 L 表示：“FPGA 同时拥有强大的处理功能和完全的设计自由度，以致于它的行业对手 ASIC 的设计者在做 wafer fabrication 之前 , 也大量使用 FPGA 来做整个系统的板级仿真，学习 FPGA 开发知识不但提升了我们的服务质量从个人角度讲也提升了自己的价值。”

实际上，小 L 只是中国数十万 FPGA 开发工程师中一个缩影，目前，随着 FPGA 从可编程逻辑芯片升级为可编程系统级芯片，其在电路中的角色已经从最初的逻辑胶合延伸到数字信号处理、接口、高密度运算等更广阔的范围，应用领域也从通信延伸到消费电子、汽车电子、工业控制、医疗电子等更多领域，现在，大批其他领域的工程师也像小 L 一样加入到 FPGA 学习应用大军中。未来，随着 FPGA 把更多的硬核如 PowerPC™ 处理器等集成进来，以及采用新的工艺将存储单元集成，FPGA 越来越成为一种融合处理、存储、接口于一体的超级芯片，“FPGA 会成为一种板级芯片，未来的电子产品可以通过配置 FPGA 来实现功能的升级，实际上，某些通信设备厂商已经在尝试这样做了。”赛灵思公司全球资深副总裁汤立人这样指出。可以想象，未来，FPGA 开发能力对工程师而言将成为类似 C 语言的基础能力之一，面对这样的发展趋势，你还能简单地将 FPGA 当成一种逻辑器件吗？还能对 FPGA 的发展无动于衷吗?

电子产品设计趋势的变化

自电子产品诞生之日起，电子产品开发流程和方法就随着电子元器件的不断演进而变化，从最早的电子管器件到晶体管再到集成电路，工程师在设计产品时，所采用的工具和方法都有所不同，但是总的来说贯穿电子设计的统一思路是：使用印刷电路板上的分立、现成元件、连接器或 IC 创建物理平台实现所需要的功能。例如，在 60 年代，如果要设计一个收音机，工程师必须通过在 PCB 板上通过晶体管、电阻、电容、电感、电线、滤波器、二极管等电路搭建出一个物理平台，实现对 RF 信号的调谐、滤波、放大等，最后实现收音机的功能。集成电路出现以后，一些分立器件被集成到一颗芯片上，但是总的设计思路没有变化，还是要在一个 PCB 板上通过无源器件和 IC 搭建出一个物理平台，实现信号的接收、处理和输出。但是，随着 FPGA 等可编程器件的诞生，设计思路正发生着微妙的变化——随着更多功能从分立器件移到可编程领域，各种不同的设计流程交汇到了一起。现在，有效的电子设计是将板卡设计、可编程逻辑设计和软件开发融合在一起，未来，随着 FPGA 融合处理、存储于一体，板卡设计将融合进可编程逻辑设计中，电子产品设计将演变为可编程逻辑设计和嵌入式软件设计，那时，电子设计将更体现一种“软”设计，一种通过开发语言和工具实现的设计，而 FPGA 将成为这种

“软”设计的载体，以 FPGA 形式存在的低成本、大规模可编程器件可以随时随地获得，这使设计者有可能将所有系统核心功能都转移到软设计中，并利用这种设计的优势。

这些“软”设计优势包括：更容易保护系统功能使其不被仿制或逆向工程，编程到设计中的“软”元素容易更新，使设计过程更具连续性。好的工具所设计的软设计不依赖于事先指定的硬件平台。而且，设计可以在最终硬件平台内继续进行，即使产品已经移交客户也仍然可行。即“软”设计将成为电子设计的发展方向。

另一点，现今及未来的电子产品都在追求智能化和个性化，智能化只能通过软件来实现，个性化呢，需要工程师简单地修改就可体现不同的特色，另外也需要保护自己的设计不被仿制，要做到这点，也需要可编程器件。

每个工程师都希望自的产品永远与众不同。与众不同就是要让产品与竞争产品不一样，让购买者选择你的产品而不选择竞争对手的产品。但是，怎么样才能在日益全球化的市场中保持与众不同呢?

不要再指望在硬件上能达到目的，因为现在几乎每个人都能获得同样的芯片。当现有物理硬件中实现的任何功能受到市场的欢迎的同时，大量的仿制就出现了。

所以要将产品的区别建立在编程器件智能上，保护有价值的 IP，并且使竞争对手很难对其进行逆向工程。而且，即使硬件已经制造出来，产品仍可以通过“软”设计进行创新并为产品增值，产品的成功就有了保障。而这些，都离不开可编程器件。可编程器件是实现“软”设计的保障和载体。

电子设计工程师设计方法和设计内容在不断变化

电子设计工程师的设计方法和内容其实也在一直变化，电子管时代，设计工程师要掌握电子管的性能和设计要点，晶体管时代，设计工程师要熟悉跟中电路的作用和搭建，集成电路诞生以后，设计工程师要熟悉 IC 管脚的作用和功能，而设计工具从最早的草稿图、软件辅助设计也发展到电子设计自动化工具 (EDA 软件)，以 FPGA 为代表的可编程器件诞生后，设计工程师不但要设计硬件电路更要熟悉 HDL、Verilog 等 IC 设计语言，此外，还要熟悉接口、数字信号处理、算法、EDA 设计方法学等等，电子工程师要学习的知识日益增多。

未来的硬件工程师是什么样的?

那么，未来的的硬件设计工程师是什么样的? 或者说未来的硬件设计工程是怎样的? 而已这样说：以 VHDL 或者 Verilog 语言来表达设计意图、以 FPGA 做为硬件载体、以计算机为设计开发工具，以 EDA 软件为开发环境、以 SoC、IP 等为综合设计的方法，已经成为硬件设计工程的主要特征。可以预见，FPGA 将成为未来的硬件工程师必用的设计元素之一。

另外，FPGA 在应用中的其他显著优势是可以减少 BOM 整合多个分立的数字器件 (例如一个很小很便宜的 CPLD 可以替换好几个 74 系列芯片)、降低 PCB 布线难度 (MGT/GTP 等串行收发器将原本与需要三五十条线并行数据线替换为少量的串行线路)、可定制性 (可以自己写代码来支持非标准的接口)，可扩展性 (可编程易修改方便升级)、加速面市时间 (只需关心功能实现，不需要再花时间制成专用 IC) 等，这样 FPGA 带给设计的公司的好处已经不是从成本体现了，它可以大幅度提升开发的效率!

综上所述，我们就明白为什么工程师要掌握 FPGA 开发知识了，希望本书有助于大家了解和掌握 FPGA 开发。

第二章、FPGA基本知识与发展趋势

FPGA 是英文 Field Programmable Gate Array 的缩写，即现场可编程门阵列，它是在 PAL、GAL、CPLD 等可编程器件的基础上进一步发展的产物。它是作为专用集成电路 (ASIC) 领域中的一种半定制电路而出现的，既解决了定制电路的不足，又克服了原有可编程器件门电路数有限的缺点。它是当今数字系统设计的主要硬件平台，其主要特点就是完全由用户通过软件进行配置和编程，从而完成某种特定的功能，且可以反复擦写。在修改和升级时，不需额外地改变 PCB 电路板，只是在计算机上修改和更新程序，使硬件设计工作成为软件开发工作，缩短了系统设计的周期，提高了实现的灵活性并降低了成本，因此获得了广大硬件工程师的青睐。

2.1 FPGA 结构和工作原理

2.1.1 梦想成就伟业

1984 年，在硅谷工作的 Bernie Vonderschmitt、Ross Freeman 和 Jim Barnett 共同构建了一个设想，他们梦想创立一家不同于一般的公司。他们希望创建一家在整个新领域内开发和推出先进技术的公司。并且，他们还希望以这种方式领导它：在这里工作的人们热爱他们的工作、享受工作的乐趣，并对他们所从事的工作着迷。

图2-1 Ross Freeman(左)是FPGA的发明人，Bernie Vonderschmitt(右)是赛灵思公司的创始人创造性地推出了“无晶圆半导体”公司的概念。

2009 年 2 月 18 日，Ross Freeman 因他的这项发明——现场可编程门阵列 (FPGA) 而荣登 2009 美国发明家名人堂。

Freeman 先生的发明是一块全部由“开放式门”组成的计算机芯片，其专利号为 4,870,302。采用这种芯片，工程师可以根据需要进行编程，添加新的功能，满足不断发展的标准或规范要求，并可在设计的最后阶段进行修改。

2.1.2 FPGA结构

对 PROM、EPROM、E2PROM 熟悉的人都知道这些可编程器件的可编程原理是通过加高压或紫外线导致三极管或 MOS 管内部的载流子密度发生变化，实现所谓的可编程，但是这些器件或只能实现单次可编程或编程状态难以稳定。FPGA 则不同，它采用了逻辑单元阵列 LCA(Logic Cell Array) 这样一个新概念，内部包括可配置逻辑模块 CLB(Configurable Logic Block)、输出输入模块 IOB(Input Output Block) 和内部连线 (Interconnect) 三个部分。

FPGA 的可编程实际上是改变了 CLB 和 IOB 的触发器状态，这样，可以实现多次重复的编程由于 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。查找表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 或者熔丝与反熔丝工艺的查找表结构。通过烧写文件改变查找表内容的方法来实现对 FPGA 的重复配置。

根据数字电路的基本知识可以知道，对于一个 n 输入的逻辑运算，不管是与或非运算还是异或运算等等，最多只可能存在 2n 种结果。所以如果事先将相应的结果存放于一个存贮单元，就相当于实现了与非门电路的功能。FPGA 的原理也是如此，它通过烧写文件去配置查找表的内容，从而在相同的电路情况下实现了不同的逻辑功能。

查找表 (Look-Up-Table) 简称为 LUT，LUT 本质上就是一个 RAM。目前 FPGA 中多使用 4 输入的 LUT，所以每一个 LUT 可以看成一个有 4 位地址线的 的 RAM。 当用户通过原理图或 HDL 语言描述了一个逻辑电路以后，PLD/FPGA 开发软件会自动计算逻辑电路的所有可能结果，并把真值表 (即结果) 事先写入 RAM，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，然后输出即可。

实际逻辑电路		LUT的实现方式	
a, b, c, d输入	逻辑输出	RAM地址	RAM中存储的内容
0000	0	0000	0
0001	0	0001	0
…	…	…	…
1111	1	1111	1

表2-1 输入与门的真值表

从表中可以看到，LUT 具有和逻辑电路相同的功能。实际上，LUT 具有更快的执行速度和更大的规模。

由于基于 LUT 的 FPGA 具有很高的集成度，其器件密度从数万门到数千万门不等，可以完成极其复杂的时序与逻辑组合逻辑电路功能，所以适用于高速、高密度的高端数字逻辑电路设计领域。其组成部分主要有可编程输入 / 输出单元、基本可编程逻辑单元、内嵌 SRAM、丰富的布线资源、底层嵌入功能单元、内嵌专用单元等，

主要设计和生产厂家有赛灵思、Altera、Lattice、Actel、Atmel 和 QuickLogic 等公司，其中最大的是美国赛灵思公司，占有可编程市场 50% 以上的市场份额，比其他所有竞争对手市场份额的总和还多。

FPGA 是由存放在片内 RAM 中的程序来设置其工作状态的，因此，工作时需要对片内的 RAM 进行编程。用户可以根据不同的配置模式，采用不同的编程方式。

加电时，FPGA 芯片将 EPROM 中数据读入片内编程 RAM 中，配置完成后，FPGA 进入工作状态。掉电后，FPGA 恢复成白片，内部逻辑关系消失，因此，FPGA 能够反复使用。FPGA 的编程无须专用的 FPGA 编程器，只须用通用的 EPROM、PROM 编程器即可。这样，同一片 FPGA，不同的编程数据，可以产生不同的电路功能。因此，FPGA 的使用非常灵活。

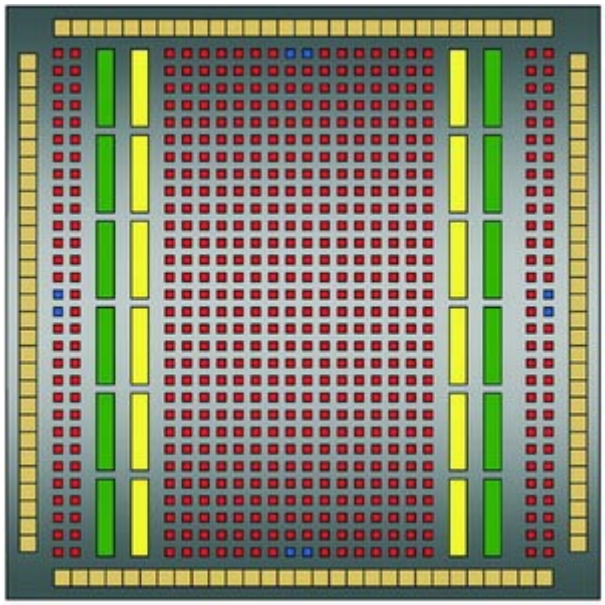


图2–2 被广泛应用的Xilinx Spartan – 3系列FPGA

如前所述，FPGA 是由存放在片内的 RAM 来设置其工作状态的，因此工作时需要对片内 RAM 进行编程。用户可根据不同的配置模式，采用不同的编程方式。Xilinx FPGA 的常用配置模式有 5 类：主串模式、从串模式、Select MAP 模式、Desktop 配置和直接 SPI 配置。

目前，FPGA 市场占有率最高的两大公司赛灵思公司和 Altera 生产的 FPGA 都是基于 SRAM 工艺的，需要在使用时外接一个片外存储器以保存程序。上电时，FPGA 将外部存储器中的数据读入片内 RAM，完成配置后，进入工作状态；掉电后 FPGA 恢复为白片，内部逻辑消失。这样 FPGA 不仅能反复使用，还无需专门的 FPGA 编程器，只需通用的 EPROM、PROM 编程器即可。Actel、QuickLogic 等公司还提供反熔丝技术的 FPGA，具有抗辐射、耐高低温、低功耗和速度快等优点，在军品和航空航天领域中应用较多，但这种 FPGA 不能重复擦写，开发初期比较麻烦，费用也比较昂贵。Lattice 是 ISP 技术的发明者，在小规模 PLD 应用上有一定的特色。早期的赛灵思公司产品一般不涉及军品和宇航级市场，但目前已经有多款产品进入该类领域。

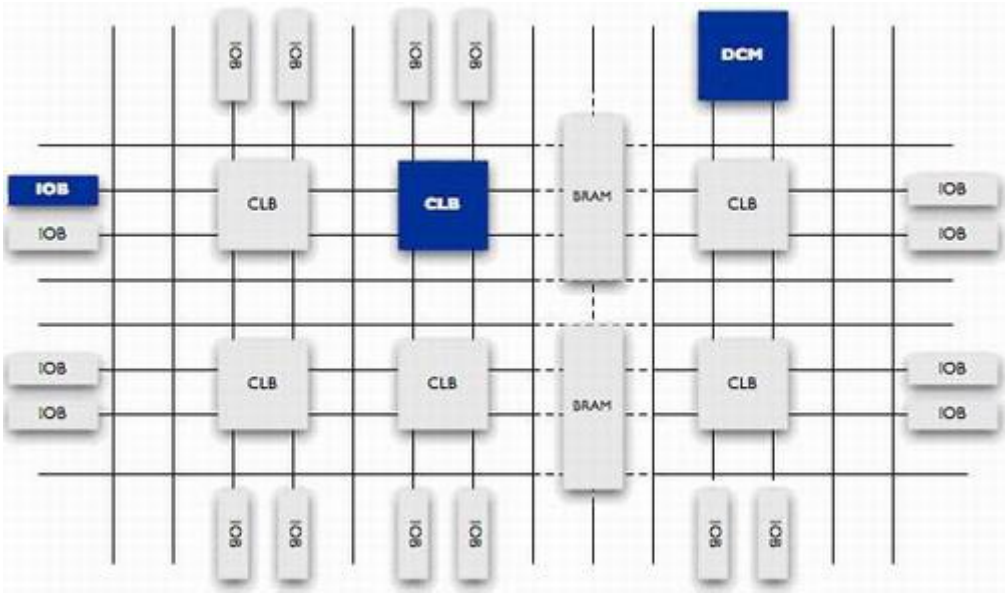


图2-3 FPGA芯片内部结构

FPGA 芯片结构目前主流的 FPGA 仍是基于查找表技术的，已经远远超出了先前版本的基本性能，并且整合了常用功能 (如 RAM、时钟管理和 DSP) 的硬核 (ASIC 型) 模块。如图 2-3 所示 (注：图 2-3 只是一个示意图，实际上每一个系列的 FPGA 都有其相应的内部结构)，FPGA 芯片主要由 6 部分完成，分别为：可编程输入输出单元、基本可编程逻辑单元、完整的时钟管理、嵌入块式 RAM、丰富的布线资源、内嵌的底层功能单元和内嵌专用硬件模块。

每个模块的功能如下：

1. 可编程输入输出单元(IOB)

可编程输入 / 输出单元简称 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，其示意结构如图 2-4 所示。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 2Gbps 的数据速率。

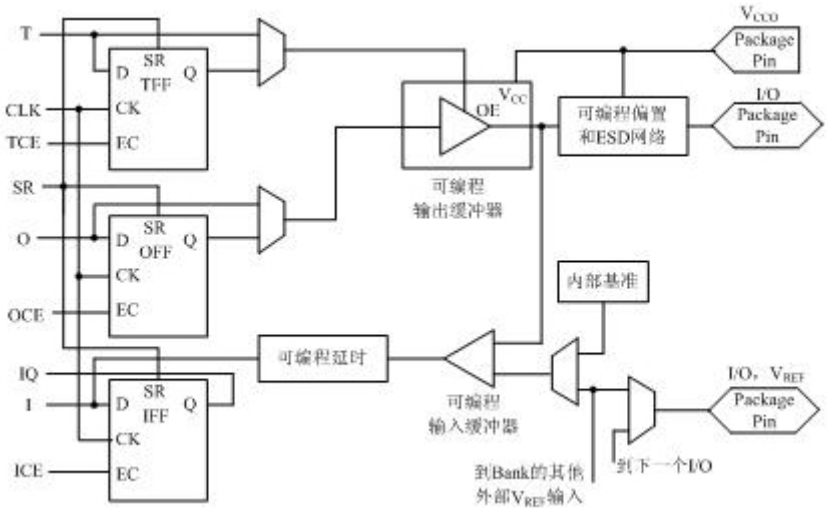


图2-4 IOB内部结构

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 bank 的接口标准由其接口电压 VCCO 决定，一个 bank 只能有一种 VCCO，但不同 bank 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. 可配置逻辑块(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。在赛灵思公司公司的 FPGA 器件中，CLB 由多个 (一般为 4 个或 2 个) 相同的 Slice 和附加逻辑构成，如图 2-5 所示。每个 CLB 模块不仅可以用于实现组合逻辑、时序逻辑，还可以配置为分布式 RAM 和分布式 ROM。

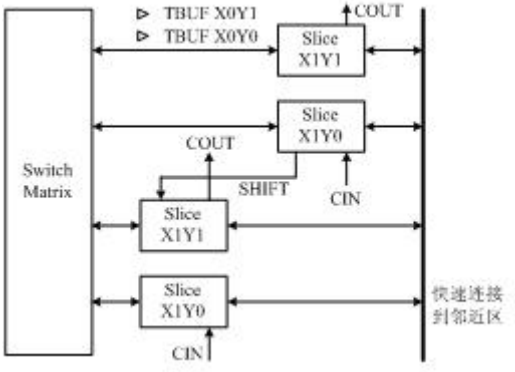


图2-5 典型的CLB结构示意图

Slice 是赛灵思公司公司定义的基本逻辑单位，其内部结构如图 2-6 所示，一个 Slice 由两个 4 输入的函数、进位逻辑、算术逻辑、存储逻辑和函数复用器组成。

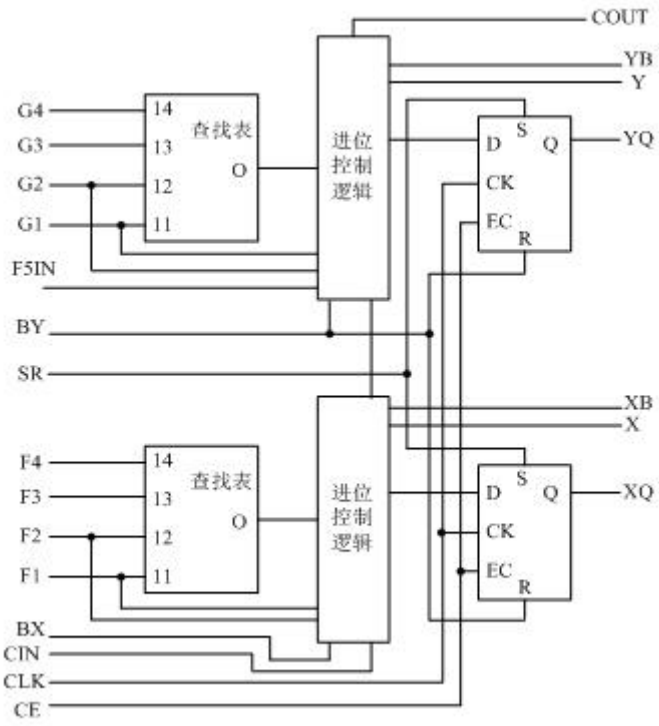


图2-6 典型的4输入Slice结构示意图

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 2bit 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. 数字时钟管理模块(DCM)

业内大多数 FPGA 均提供数字时钟管理 (赛灵思公司的全部 FPGA 均具有这种特性)。赛灵思公司推出最先进的 FPGA 提供数字时钟管理和相位环路锁定。相位环路锁定能够提供精确的时钟综合，且能够降低抖动，并实现过滤功能。

4. 嵌入式块RAM(BRAM)

大多数 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。RAM、FIFO 是比较普及的概念，在此就不冗述。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。

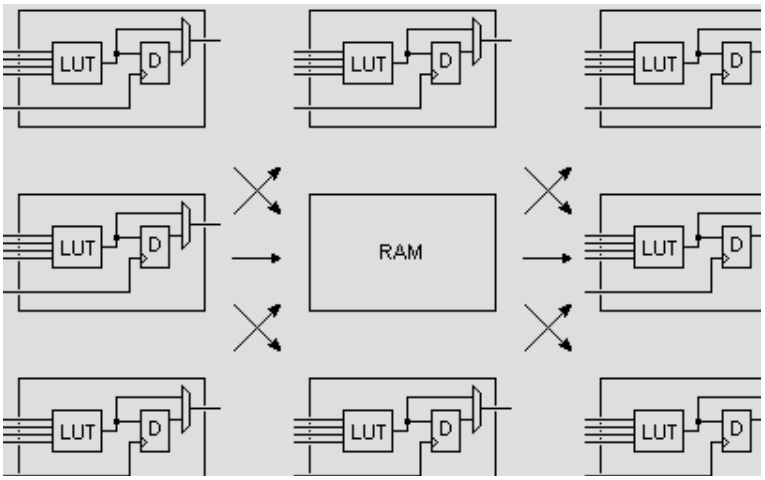


图2-7 内嵌的块RAM

单片块 RAM 的容量为 18k 比特，即位宽为 18 比特、深度为 1024，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。

5. 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 Bank 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。

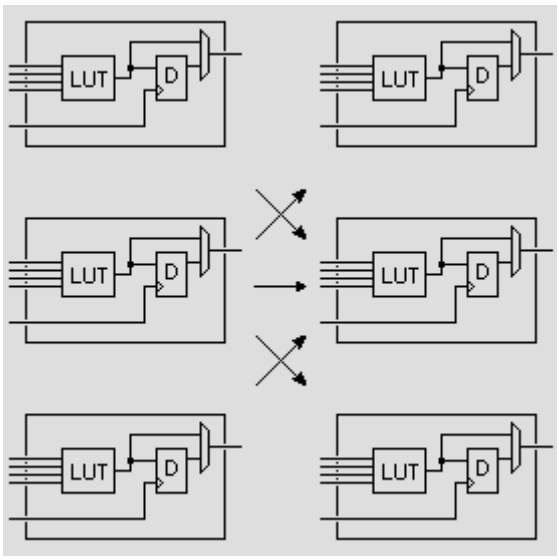


图2-8 FPGA内部互连布线

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。

6. 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 等软处理核 (Soft Core)。现在越来越丰富的内嵌功能单元,使得单片 FPGA 成为了系统级的设计工具,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。DLL 的结构如图 2-8 所示。

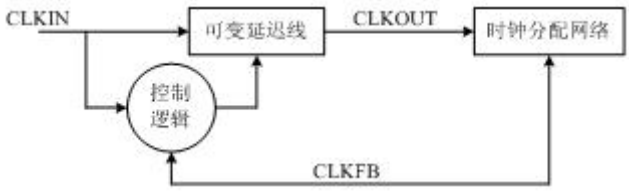


图2-9 典型的DLL模块示意图

7. 内嵌专用硬核

内嵌专用硬核是相对底层嵌入的软核而言的，指 FPGA 处理能力强大的硬核 (Hard Core)，等效于 ASIC 电路。为了提高 FPGA 性能，芯片生产商在芯片内部集成了一些专用的硬核。例如：为了提高 FPGA 的乘法速度，主流的 FPGA 中都集成了专用乘法器；为了适用通信总线与接口标准，很多高端的 FPGA 内部都集成了串并收发器 (SERDES)，可以达到数十 Gbps 的收发速度。

赛灵思公司的高端产品不仅集成了 Power PC 系列 CPU，还内嵌了 DSP Core 模块，其相应的系统级设计工具是 EDK 和 Platform Studio，并依此提出了片上系统 (System on Chip) 的概念。通过 PowerPC™、Miroblaze、Picoblaze 等平台，能够开发标准的 DSP 处理器及其相关应用，达到 SOC 的开发目的。

此外,新推出赛灵思的 FPGA 系列如 Virtex-5 LXT 还内建了 PCI Express 和三态以太网 MAC 硬核 (TEMAC)，与软核实现方式相比，硬核可以把功耗降低 5~10 倍，节约将近 90% 的逻辑资源。

Xilinx 三态以太网 MAC 核是一个可参数化的核，非常适合在网络设备中使用，例如开关和路由器等。可定制的 TEMAC 核使系统设计者能够实现宽范围的集成式以太网设计，从低成本 10/100 以太网到性能更高的 1GB 端口。TEMAC 核设计符合 IEEE 802.3 规范的要求,并且可以在 1000Mbps、100 Mbps 和 10 Mbps 模式下运行。另外，它还支持半双工和全双工操作。TEMAC 核通过 Xilinx CORE Generator™ 工具提供，是 Xilinx 全套以太网解决方案的一部分。

2.1.3 软核、硬核以及固核的概念

IP(Intelligent Property) 核是具有知识产权核的集成电路芯核总称，是经过反复验证过的、具有特定功能的宏模块，与芯片制造工艺无关，可以移植到不同的半导体工艺中。到了 SOC 阶段，IP 核设计已成为 ASIC 电路设计公司和 FPGA 提供商的重要任务，也是其实力体现。对于 FPGA 开发软件，其提供的 IP 核越丰富，用户的设计就越方便，其市场占用率就越高。目前，IP 核已经变成系统设计的基本单元，并作为独立设计成果被交换、转让和销售。

从IP核的提供方式上,通常将其分为软核、固核和硬核这3类。从完成IP核所花费的成本来讲,硬核代价最大；从使用灵活性来讲，软核的可复用使用性最高。

1. 软核(Soft IP Core)

软核在 EDA 设计领域指的是综合之前的寄存器传输级 (RTL) 模型；具体在 FPGA 设计中指的是对电路的硬件语言描述，包括逻辑描述、网表和帮助文档等。软核只经过功能仿真，需要经过综合以及布局布线才能使用。其优点是灵活性高、可移植性强，允许用户自配置；缺点是对模块的预测性较低，在后续设计中存在发生错误的可能性，有一定的设计风险。软核是 IP 核应用最广泛的形式。

2. 固核(Firm IP Core)

固核在 EDA 设计领域指的是带有平面规划信息的网表;具体在 FPGA 设计中可以看做带有布局规划的软核，通常以 RTL 代码和对应具体工艺网表的混合形式提供。将 RTL 描述结合具体标准单元库进行综合优化设计，形成门级网表，再通过布局布线工具即可使用。和软核相比，固核的设计灵活性稍差，但在可靠性上有较大提高。目前，固核也是 IP 核的主流形式之一。

3. 硬核 (Hard IP Core)

硬核在 EDA 设计领域指经过验证的设计版图；具体在 FPGA 设计中指布局和工艺固定、经过前端和后端验证的设计，设计人员不能对其修改。不能修改的原因有两个：首先是系统设计对各个模块的时序要求很严格，不允许打乱已有的物理版图；其次是保护知识产权的要求，不允许设计人员对其有任何改动。IP 硬核的不许修改特点使其复用有一定的困难，因此只能用于某些特定应用，使用范围较窄。

2.1.4 从可编程器件发展看FPGA未来趋势

可编程逻辑器件的发展历史可编程逻辑器件的发展可以划分为 4 个阶段，即从 20 世纪 70 年代初到 70 年代中为第 1 阶段，20 世纪 70 年代中到 80 年代中为第 2 阶段，20 世纪 80 年代到 90 年代末为第 3 阶段，20 世纪 90 年代末到目前为第 4 阶段。

第 1 阶段的可编程器件只有简单的可编程只读存储器 (PROM)、紫外线可擦除只读存储器 (EPROM) 和电可擦只读存储器 (EEPROM)3 种，由于结构的限制，它们只能完成简单的数字逻辑功能。

第 2 阶段出现了结构上稍微复杂的可编程阵列逻辑 (PAL) 和通用阵列逻辑 (GAL) 器件，正式被称为 PLD，能够完成各种逻辑运算功能。典型的 PLD 由“与”、“非”阵列组成，用“与或”表达式来实现任意组合逻辑，

所以 PLD 能以乘积和形式完成大量的逻辑组合。

第 3 阶段赛灵思和 Altera 分别推出了与标准门阵列类似的 FPGA 和类似于 PAL 结构的扩展性 CPLD，提高了逻辑运算的速度，具有体系结构和逻辑单元灵活、集成度高以及适用范围宽等特点，兼容了 PLD 和通用门阵列的优点，能够实现超大规模的电路，编程方式也很灵活，成为产品原型设计和中小规模 (一般小于 10000) 产品生产的首选。这一阶段，CPLD、FPGA 器件在制造工艺和产品性能都获得长足的发展，达到了 0.18 工艺和系数门数百万门的规模。

第 4 阶段出现了 SOPC 和 SOC 技术，是 PLD 和 ASIC 技术融合的结果，涵盖了实时化数字信号处理技术、高速数据收发器、复杂计算以及嵌入式系统设计技术的全部内容。赛灵思和 Altera 也推出了相应 SOCFPGA 产品，制造工艺达到 65nm ，系统门数也超过百万门。并且，这一阶段的逻辑器件内嵌了硬核高速乘法器、Gbits 差分串行接口、时钟频率高达 500MHz 的 PowerPC™ 微处理器、软核 MicroBlaze、Picoblaze、Nios 以及 NiosII，不仅实现了软件需求和硬件设计的完美结合，还实现了高速与灵活性的完美结合，使其已超越了 ASIC 器件的性能和规模，也超越了传统意义上 FPGA 的概念，使 PLD 的应用范围从单片扩展到系统级。未来，赛灵思高层透露，该公司正在研制采用全新工艺的新型 FPGA，这种 FPGA 将集成更大的存储单元和其他功能器件，FPGA 正向超级系统芯片的方向发展！ 2 月 5 日，赛灵思发布了采用 40nm 和 45nm 的 Spartan – 6 和 Virtex – 6 FPGA 系列，并开启了目标设计平台这一新的设计理念，相信 FPGA 的应用会得到更大的发展！

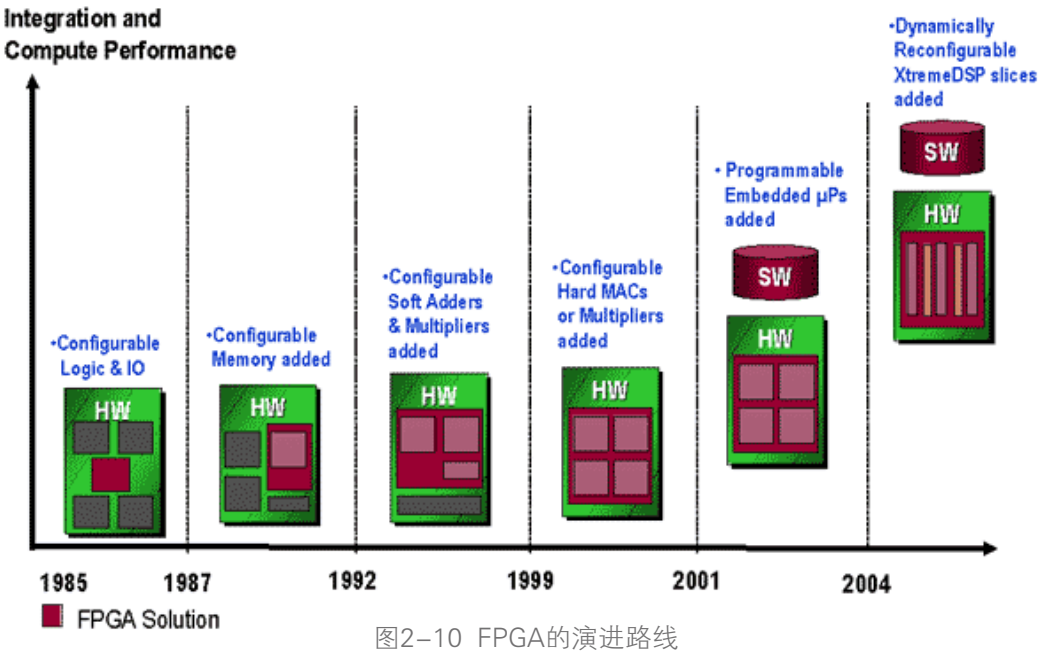


图2–10 FPGA的演进路线

第三章、FPGA主要供应商与产品

1984 年，赛灵思发明了现场可编程门阵列 (FPGA)，同时它成为全球首家无晶圆半导体公司的鼻祖，赛灵思 通过不断应用尖端技术来长久保持它的行业领袖地位：赛灵思 是首家采用 180nm、150nm、130nm、90nm 和 65nm 工艺技术的企业，目前提供约占世界 90% 的高端 65nm FPGA 产品。据 iSuppli 的统计数据，2007 年它拥有世界 51% 以上的可编程器件市场份额。目前 FPGA 领域主要的供应商是赛灵思、Altera、Actel 和 Lattice。其中 Altera 和赛灵思主要生产一般用途 FPGA，其主要产品采用 RAM 工艺。Actel 主要提供非易失性 FPGA，产品主要基于反熔丝工艺和 FLASH 工艺。

由于赛灵思一直在 FPGA 开发领域方面拥有领先优势和最大份额，故本文主要介绍赛灵思公司的 FPGA 产品。

3.1.1 赛灵思主要产品介绍

目前赛灵思公司有两大类 FPGA 产品：Spartan 类和 Virtex 类，前者主要面向低成本的中低端应用，是目前业界成本最低的一类 FPGA；后者主要面向高端应用，属于业界的顶级产品。这两个系列的差异仅限于芯片的规模和专用模块上，都采用了先进的 0.13 、90 甚至 65 制造工艺，具有相同的卓越品质。

1. Spartan类FPGA

Spartan 系列适用于普通的工业、商业等领域，目前主流的芯片包括：Spartan–2、Spartan–2E、Spartan–3、Spartan–3A、Spartan–3E 以及最新的 Spartan – 6 等种类。其中 Spartan–2 最高可达 20 万系统门，Spartan–2E 最高可达 60 万系统门，Spartan–3 最高可达 500 万门，Spartan–3A 和 Spartan–3E 不仅系统门数更大，还增强了大量的内嵌专用乘法器和专用块 RAM 资源，具备实现复杂数字信号处理和片上可编程系统的能力。

(1)Spartan–2 系列

Spartan–2 在 Spartan 系列的基础上继承了更多的逻辑资源，达到更高的性能，芯片密度高达 20 万系统门。由于采用了成熟的 FPGA 结构，支持流行的接口标准，具有适量的逻辑资源和片内 RAM，并提供灵活的时钟处理，可以运行 8 位的 PicoBlaze 软核，主要应用于各类低端产品中。其主要特点如下所示：采用 0.18 工艺，密度达到 5292 逻辑单元；系统时钟可以达到 200MHz；采用最大门数为 20 万门，具有延迟数字锁相环；具有可编程用户 I/O；具有片上块 RAM 存储资源；

Spartan–2 系列产品的主要技术特征如下表所示。

型号	系统门数	SLICE数目	分布式RAM容量	块RAM容量	专用乘法器数	DCM数目	最大可用I/O数	最大差分I/O对数
XC2S15	15k	216	6144b	16k	0	0	86	0
XC2S30	30k	486	13824b	24k	0	0	132	0
XC2S50	50k	864	24567b	32k	0	0	176	0
XC2S100	100k	1350	38400b	40k	0	0	196	0
XC2S150	150k	1944	55296b	48k	0	0	260	0
XC2S200	200K	2646	75264b	56k	0	0	284	0

表3–1 Spartan–2系列 FPGA主要技术特征

(2)Spartan-2E 系列

Spartan-2E 基于 Virex-E 架构，具有比 Spartan-2 更多的逻辑门、用户 I/O 和更高的性能。赛灵思还为其提供了包括存储器控制器、系统接口、DSP、通信以及网络等 IP 核，并可以运行 CPU 软核，对 DSP 有一定的支持。其主要特点如下所示：采用 0.15 工艺，密度达到 15552 逻辑单元；最高系统时钟可达 200MHz；最大门数为 60 万门，最多具有 4 个延时锁相环；核电压为 1.2V，I/Q 电压可为 1.2V、3.3V、2.5V，支持 19 个可选的 I/O 标准；最大可达 288k 的块 RAM 和 221K 的分布式 RAM；

Spartan-2E 系列产品的主要技术特征如下表所示。

型号	系统门数	SLICE数目	分布式RAM容量	块RAM容量	专用乘法器数	DCM数目	最大可用I/O数	最大差分I/O对数
XC2S50E	50k	964	24576b	32k	0	0	182	83
XC2S100E	110k	1350	38400b	40k	0	0	202	86
XC2S150E	150k	1944	55296b	48k	0	0	265	114
XC2S200E	200k	2646	752645b	56k	0	0	289	120
XC2S300E	300k	3456	98304b	64k	0	0	329	120
XC2S400E	400K	5400	153600b	160k	0	0	410	172
XC2S600E	600k	7776	221184b	288k	0	0	514	205

表3-2 Spartan-2E系列 FPGA主要技术特征

(3)Spartan-3 系列

Spartan-3 基于 Virtex-II FPGA 架构，采用 90 技术，8 层金属工艺，系统门数超过 5 百万，内嵌了硬核乘法器和数字时钟管理模块。从结构上看，Spartan-3 将逻辑、存储器、数学运算、数字处理器处理器、I/O 以及系统管理资源完美地结合在一起，使之有更高层次、更广泛的应用，获得了商业上的成功，占据了较大份额的中低端市场。其主要特性如下：采用 90 工艺，密度高达 74880 逻辑单元；最高系统时钟为 340MHz；具有 的专用乘法器；核电压为 1.2V，端口电压为 3.3V、2.5 V、1.2V，支持 24 种 I/O 标准；高达 520k 分布式 RAM 和 1872k 的块 RAM；具有片上时钟管理模块(DCM)；具有嵌入式 Xtrema DSP 功能，每秒可执行 3300 亿次乘加。

Spartan-3 系列产品的主要技术特征如下表所示。

型号	系统门数	SLICE数目	分布式RAM容量	块RAM容量	专用乘法器数	DCM数目	最大可用I/O数	最大差分I/O对数
XC3S50	50k	864	12k	72k	4	2	124	56
XC3S200	200k	2260	30k	216k	12	4	173	76
XC3S400	400k	4032	56k	288k	16	4	264	116
XC3S1000	1000k	8640	120k	432k	24	4	391	175
XC3S1500	1500k	14976	208k	576k	32	4	487	221
XC3S2000	2000k	23040	320k	720k	40	4	565	270
XC3S4000	4000k	31104	432k	1728k	96	4	712	312
XC3S5000	5000k	37440	520k	1872k	104	4	784	344

表3-3 Spartan-3系列 FPGA主要技术特征

(4)Spartan-3A/3ADSP/3AN 系列

Spartan-3A 在 Spartan-3 和 Spartan-3E 平台的基础上，整合了各种创新特性帮助客户极大地削减了系统总成本。利用独特的器件 DNA ID 技术，实现业内首款 FPGA 电子序列号；提供了经济、功能强大的机制来防止发生窜改、克隆和过度设计的现象。并且具有集成式看门狗监控功能的增强型多重启动特性。支持商用 flash 存储器，有助于削减系统总成本。其主要特性为：采用 90 工艺，密度高达 74880 逻辑单元；工作时钟范围为 5MHz~320MHz；领先的连接功能平台，具有最广泛的 IO 标准 (26 种，包括新的 TMDS 和 PPDS) 支持；利用独特的 Device DNA 序列号实现的业内首个功能强大的防克隆安全特性；五个器件，具有高达 1.4M 的系统门和 502 个 I/O；灵活的功耗管理。

Spartan-3A 系列产品的主要技术特征如下表所示。

型号	系统门数	SLICE数目	分布式RAM容量	块RAM容量	专用乘法器数	DCM数目	最大可用I/O数	最大差分I/O对数
XC3S50	50k	864	12k	72k	4	2	124	56
XC3S200	200k	2260	30k	216k	12	4	173	76
XC3S400	400k	4032	56k	288k	16	4	264	116
XC3S1000	1000k	8640	120k	432k	24	4	391	175
XC3S1500	1500k	14976	208k	576k	32	4	487	221
XC3S2000	2000k	23040	320k	720k	40	4	565	270
XC3S4000	4000k	31104	432k	1728k	96	4	712	312
XC3S5000	5000k	37440	520k	1872k	104	4	784	344

表3-4 Spartan-3A系列FPGA主要技术特征

Spartan-3ADSP 平台提供了最具成本效益的 DSP 器件，其架构的核心就是 XtremeDSP DSP48A slice，还提供了性能超过 30GMAC/s、存储器带宽高达 2196 Mbps 的新型 XC3SD3400A 和 XC3SD1800A 器件。新型 Spartan-3A DSP 平台是成本敏感型 DSP 算法和需要极高 DSP 性能的协处理应用的理想之选。其主要特征如下所示。 采用 90 工艺，密度高达 74880 逻辑单元； 内嵌的 DSP48A 可以工作到 250MHz；采用结构化的 SelectRAM 架构,提供了大量的片上存储单元； VCCAUX 的电压支持 2.5V 和 3.3V,对于 3.3V 的应用简化了设计；低功耗效率，Spartan-3A DSP 器件具有很高的信号处理能力 4.06 GMACs/mW。

Spartan-3ADSP 系列产品的主要技术特征如下表所示。

型号	系统门数	SLICE数目	分布式RAM容量	块RAM容量	专用乘法器数	DCM数目	最大可用I/O数	最大差分I/O对数
XC3S1800A	1800k	18770	260k	1512k	84	8	519	227
XC3S3400A	3400k	26856	373k	2268k	126	8	469	213

表3-5 Spartan-3ADSP系列 FPGA主要技术特征

Spartan-3AN 芯片为最高级别系统集成的非易失性安全 FPGA，提供下列 2 个独特的性能：先进 SRAM FPGA 的大量特性和高性能以及非易失性 FPGA 的安全、节省板空间和易于配置的特性。Spartan-3AN 平台是对空间要求严苛和 / 或安全应用及低成本嵌入式控制器的理想选择。Spartan-3AN 平台的关键特性包括：业界

首款 90nm 非易失性 FPGA，具有可以实现灵活的、低成本安全性能的 Device DNA 电子序列号；业内最大的片上用户 Flash，容量高达 11Mb；提供最广泛的 I/O 标准支持，包括 26 种单端与差分信号标准 灵活的电源管理模式，休眠模式下可节省超过 40% 的功耗 五个器件，具有高达 1.4M 的系统门和 502 个 I/O。

Spartan–3AN 系列产品的主要技术特征如下表所示

型号	系统门数	SLICE数目	分布式RAM容量	块RAM容量	专用乘法器数	DCM数目	最大可用I/O数	最大差分I/O对数
XC3S50AN	50	792	11k	54k	3	2	108	50
XC3S200AN	200	2016	28k	288k	16	4	195	90
XC3S400AN	400	4032	56k	360k	20	4	311	142
XC3S700AN	700	6624	92k	360k	20	8	372	165
XC3S1400AN	1400	12672	176k	576k	32	8	502	227

表3–6 Spartan–3AN系列 FPGA主要技术特征

(5)Spartan–3E 系列

Spartan–3E 是目前 Spartan 系列最新的产品，具有系统门数从 10 万到 160 万的多款芯片，是在 Spartan–3 成功的基础上进一步改进的产品，提供了比 Spartan–3 更多的 I/O 端口和更低的单位成本，是赛灵思公司性价比最高的 FPGA 芯片。由于更好地利用了 90 技术，在单位成本上实现了更多的功能和处理带宽，是赛灵思公司新的低成本产品代表，是 ASIC 的有效替代品，主要面向消费电子应用，如宽带无线接入、家庭网络接入以及数字电视设备等。其主要特点如下：采用 90 工艺；大量用户 I/O 端口，最多可支持 376 个 I/O 端口或者 156 对差分端口；端口电压为 3.3V、2.5 V、1.8V、1.5V、1.2V；单端端口的传输速率可以达到 622，支持 DDR 接口；最多可达 36 个 的专用乘法器、648 块 RAM、231 分布式 RAM；宽的时钟频率 以及多个专用片上数字时钟管理 (DCM) 模块。

Spartan–3E 系列产品的主要技术特征如下表所示。

型号	系统门数	SLICE数目	分布式RAM容量	块RAM容量	专用乘法器数	DCM数目	最大可用I/O数	最大差分I/O对数
XC3S50AN	50	792	11k	54k	3	2	108	50
XC3S200AN	200	2016	28k	288k	16	4	195	90
XC3S400AN	400	4032	56k	360k	20	4	311	142
XC3S700AN	700	6624	92k	360k	20	8	372	165
XC3S1400AN	1400	12672	176k	576k	32	8	502	227

表3–7 Spartan–3E系列FPGA主要技术特征

Spartan-3A 延伸系列平台对比列表

系统要求	3A	3AN	3A DSP
多功能嵌入式处理	✓	✓	✓
最佳存储器架构	✓	✓	✓
低成本安全性	✓	✓	✓
集成式 Flash 存储器		✓	
单芯片非易失性		✓	
高性能 DSP 功能			✓
增强的电源管理	✓	✓	✓

表3-8 Spartan－3A延伸系列品台性能对比

(6)Spartan-6 系列

作为 Spartan FPGA 系列的第六代产品，Spartan-6 FPGA 系列采用可靠的低功耗 45nm 9 层金属布线双层氧化工艺技术生产。 这一新系列产品实现了低风险、低成本、低功耗以及高性能的完美平衡。 Spartan-6 FPGA 系列的高效双寄存器 6 输入 LUT(查找表) 逻辑结构利用了可靠成熟的 Virtex 架构，支持跨平台兼容性以及优化系统性能。 丰富的内建系统级模块包括 DSP 逻辑片、高速收发器以及 PCI Express® 接口内核，也源于 Virtex 系列，能够提供更高程度的系统级集成。

Spartan-6 FPGA 系列专门针对成本和功率敏感的市场 (如汽车娱乐、平板显示以及视频监控) 采用了特殊技术。 新的高性能集成存储器控制器支持 DDR、DDR2、DDR3 和移动 DDR 存储器，硬内核的多端口总线结构能够提供可预测的时序和高达 DDR2/DDR3 800 (400MHz) 的性能。 在设计向导的支持下，为 Spartan-6 FPGA 构建存储控制器的过程变得非常简单和直接。

先进功率管理技术方面的创新以及可选的 1.0v 低功耗内核使得 Spartan-6 FPGA 能够比前一代 Spartan 系列功耗降低多达 65%。 快速灵活的 I/O 支持超过 12Gbps 的存储器访问带宽，兼容 3.3v 电压并且采用了更为绿色的 RoHS 兼容无铅封装。

Spartan-6 系列产品的主要技术特征如下表所示：

Part Number		LX4	LX9	LX16	LX25	LX45	LX100	LX150	LX25T	LX45T	LX100T	LX150T
Logic Cells		3.4K	9K	15K	24K	43K	101K	147K	24K	43K	101K	147K
Maximum Distributed RAM (Kbits)		32	90	136	228	401	975	1,358	228	401	975	1,358
Block RAM (18K bits each)		8	32	32	52	116	268	268	52	116	268	268
Total Block RAM (Kbits)		144	576	576	936	2,088	4,824	4,824	936	2,088	4,824	4,824
Clock Manager Tiles (CMT)		1	2	2	2	4	6	6	2	4	6	6
DSP48A1 Slices		4	16	32	38	58	182	182	38	58	182	182
PCI Express® Endpoint Block		–	–	–	–	–	–	–	1	1	1	1
Memory Controller Blocks		0	2	2	2	2	4	4	2	2	4	4
GTP Low-Power Transceivers		–	–	–	–	–	–	–	2	4	8	8
Package	Size (Pitch)	Maximum User I/O: Select IO™ Interface Pins (GTP transceivers)										
TQ144	20 x 20 mm (0.5 mm)	100	100									
CS225	13 x 13 mm (0.8 mm)	120	160	160	160							
FT256	17 x 17 mm (1.0 mm)			186	186							
CS324	15 x 15 mm (0.8 mm)		200	232	232				174 (2)	174 (4)		
FG484	23 x 23 mm (1.0 mm)				264	354	354	354	264 (2)	296 (4)	296 (4)	296 (4)
FG676	27 x 27 mm (1.0 mm)					370	498	498		370 (4)	396 (8)	396 (8)

表3-9 Spartan-6系列FPGA主要技术特征

2.Virtex系列FPGA

Virtex 系列是赛灵思的高端产品，也是业界的顶级 FPGA 产品，赛灵思公司正是凭借 Vitex 系列产品赢得市场，从而获得 FPGA 供应商领头羊的地位。可以说赛灵思以其 Virtex–5、Virtex–4、Virtex–2 Pro 和 Virtex–2 系列 FPGA 产品引领现场可编程门阵列行业。主要面向电信基础设施、汽车工业、高端消费电子等应用。目前的主流芯片包括：Vitrex–2、Virtex–2 Pro、Vitex–4 和 Virtex–5 等种类。

(1)Vitrex–2 系列

Vitrex–2 系列具有优秀的平台解决方案，这进一步提升了其性能；且内置 IP 核硬核技术，可以将硬 IP 核分配到芯片的任何地方，具有比 Vitex 系列更多的资源和更高的性能。其主要特征如下所示：采用 0.15/0.12 工艺；核电压为 1.5V，工作时钟可以达到 420MHz；支持 20 多种 I/O 接口标准；内嵌了多个 硬核乘法器，提高了 DSP 处理能力；具有完全的系统时钟管理功能，多达 12 个 DCM 模块。

Virtex–2 系列产品的主要技术特征如下表所示。

型号	系统门数	SLICE数目	分布式RAM容量	块RAM容量	专用乘法器数	DCM数目	最大可用I/O数	最大差分I/O对数
XC2V40	40k	256	8k	72k	4	4	88	0
XC2V80	80k	512	16k	144k	8	4	120	0
XC2V250	250k	1536	48k	432k	24	8	200	0
XC2V500	500k	3072	96k	576k	32	8	264	0
XC2V1000	1000k	5120	160k	720k	40	8	432	0
XC2V1500	1500K	7680	240k	864k	48	8	528	0
XC2V2000	2000K	10752	336k	1008k	56	8	624	0
XC2V3000	3000K	14336	448k	1728k	96	12	720	0
XC2V4000	4000K	23040	720k	2160k	120	12	912	0
XC2V6000	6000K	33792	1056k	2592k	144	12	1104	0
XC2V8000	8000K	46592	1456k	3024k	168	12	1108	0

表3–10 Virtex–2系列 FPGA主要技术特征

(2)Virtex–2Pro 系列

Virtex–2 Pro 系列在 Virtex–2 的基础上，增强了嵌入式处理功能，内嵌了 PowerPC™405 内核，还包括了先进的主动互联 (Active Interconnect) 技术，以解决高性能系统所面临的挑战。此外还增加了高速串行收发器，提供了千兆以太网的解决方案。其主要特征如下所示：采用 0.13 工艺；核电压为 1.5V，工作时钟可以达到 420MHz；支持 20 多种 I/O 接口标准；增加了 2 个高性能 RISC 技术、频率高达 400MHz 的 PowerPC™ 处理器；增加多个 3.125Gbps 速率的 Rocket 串行收发器；内嵌了多个 硬核乘法器，提高了 DSP 处理能力；具有完全的系统时钟管理功能，多达 12 个 DCM 模块。 Virtex–2 Pro 系列产品的主要技术特征如下表所示。

型号	SLICE	分布式	块RAM容量	PowerPC	专用乘	DCM数目	Rocket	最大可用I/O数
	数目	RAM容量			法器数		I/O	
XC2VP2	1408	44k	216k	0	12	4	4	204
XC2VP4	3008	94k	504k	1	28	4	4	348
XC2VP7	4928	154k	792k	1	44	8	8	396
XC2VP20	9280	290k	1584k	2	88	8	8	564
XC2VP30	13696	428k	2448k	2	136	8	8	644
XC2VP40	19392	606k	3456k	2	192	8	12	804
XC2VP50	23616	738k	4176k	2	232	8	12	852
XC2VP70	33088	1034k	5904k	2	328	8	16或20	996
XC2VP100	44096	1378k	7992k	2	444	12	20	1164
XC2VP125	55616	1738k	10008k	4	556	12	20或24	1200

表3-11 Virtex-2 Pro系列 FPGA主要技术特征

(3)Vitex-4 系列

Virtex-4 器件整合了高达 200,000 个的逻辑单元，高达 500 MHz 的性能和无可比拟的系统特性。Vitex-4 产品基于新的高级硅片组合模块 (ASMBL) 架构，提供了一个多平台方式 (LX、SX、FX)，使设计者可以根据需求选用不同的开发平台;逻辑密度高，时钟频率能够达到 500MHz;具备 DCM 模块、PMCD 相位匹配时钟分频器、片上差分时钟网络；采用了集成 FIFO 控制逻辑的 500MHz SmartRAM 技术，每个 I/O 都集成了 ChipSync 源同步技术的 1 Gbps I/O 和 Xtreme DSP 逻辑片。其主要特点如下：采用了 90 工艺，集成了高达 20 万的逻辑单元；系统时钟 500MHz；采用了集成 FIFO 控制逻辑的 500MHz Smart RAM 技术；具有 DCM 模块、PMCD 相位匹配时钟分频器和片上差分时钟网络； 每个 I/O 都集成了 ChipSync 源同步技术的 1Gbps I/O；具有超强的信号处理能力，集成了数以百计的 XtremeDSP Slice，单片最大的处理速率为 。 Vitex-4 LX 平台 FPGA 的特点是密度高达 20 万逻辑单元，是全球逻辑密度最高的 FPGA 系列之一，适合对逻辑门需求高的设计应用。

Virtex-4 SX 平台提高了 DSP、RAM 单元与逻辑单元的比例，最多可以提供 512 个 XtremeDSP 硬核，可以工作在 500MHz，其最大的处理速率为 ，并可以以其创建 40 多种不同功能，并能多个组合实现更大规模的 DSP 模块。与 Vitex-2 Pro 系列相比，还大大降低了成本和功耗，具有极低的 DSP 成本。SX 平台的 FPGA 非常适合应用于高速、实时的数字信号处理领域。

Virtex-4 FX 平台内嵌了 1~2 个 32 位 RISC PowerPC™ 处理器，提供了 4 个 1300 Dhrystone MIPS、10/100/1000 自适应的以太网 MAC 内核，协处理器控制器单元 (APU) 允许处理器在 FPGA 中构造专用指令，使 FX 器件的性能达到固定指令方式的 20 倍;此外,还包含 24 个 Rocket I/O 串行高速收发器,支持常用的 0.6Gbps、1.25 Gbps、2.5 Gbps、3.125 Gbps、4 Gbps、6.25 Gbps、10 Gbps 等高速传输速率。FX 平台适用于复杂计算和嵌入式处理应用。

Virtex-4 系列产品的主要技术特征如下表所示。

型号	SLICE	分布式	块RAM容量	PowerPC	XtremeDSP	DCM数目	Rocket	以太网
	数目	RAM容量			Slice		I/O	MAC
XC4VLX15	6144	96k	864k	0	32	4	0	0
XC4VLX25	10572	168k	1296k	0	48	8	0	0
XC4VLX40	18432	288k	1728k	0	64	8	0	0
XC4VLX60	26624	416k	2880k	0	64	8	0	0
XC4VLX80	35840	560k	3600k	0	80	12	0	0
XC4VLX100	49152	768k	4320k	0	96	12	0	0
XC4VLX160	67584	1056k	5184k	0	96	12	0	0
XC4VLX200	89088	1392k	6048k	0	96	12	0	0
XC4VSX25	10240	160k	2304k	0	128	4	0	0
XC4VSX35	15360	240k	3456k	0	192	8	0	0
XC4VSX55	24576	384k	5760k	0	320	8	0	0
XC4VFX12	5472	86k	648k	1	32	4	0	2
XC4VFX20	8544	134k	1224k	1	32	4	8	2
XC4VFX40	18624	291k	2592k	2	48	8	12	4
XC4VFX60	25280	395k	4176k	2	128	12	16	4
XC4VFX100	42176	659k	6768	2	160	12	20	4
XC4VFX140	63168	987k	9936k	2	192	20	24	4

表3-12 Virtex-4系列 FPGA主要技术特征

Virtex-4 平台对比表

特性/性能	LX	SX	FX
逻辑	✓	✓	✓
先进的系统时钟管理	✓	✓	✓
完整的存储器资源	✓	✓	✓
最高的设计安全性	✓	✓	✓
超高性能 DSP 功能	✓	✓	✓
终极并行连接功能	✓	✓	✓
全面的串行连接功能			✓
嵌入式处理			✓

表3-13 Virtex-4系列平台性能对比

(5)Virtex-5 系列

Virtex®-5 FPGA 是世界上首款 65nm FPGA 系列，采用 1.0v、三栅极氧化层工艺技术制造而成，并且根据所选器件可以提供 330,000 个逻辑单元、1,200 个 I/O 引脚、48 个低功耗收发器以及内置式 PowerPC™ 440、PCIe® 端点和以太网 MAC 模块。已经提供了 5 种系列平台，分别是 LX、LXT 、SXT、FXT、TXT，每种平台都在高性能逻辑、串行连接功能、信号处理和嵌入式处理性能方面实现了最佳平衡。例如 LX 针对高性能逻

辑进行了优化，LXT 针对具有低功耗串行连接功能的高性能逻辑进行了优化，SXT 针对具有低功耗串行连接功能的 DSP 和存储器密集型应用进行了优化。Virtex–5 FXT 则用于实现具有速率最高的串行连接功能的嵌入式处理，Virtex–5 TXT 可用于实现超高带宽应用，如有线通信与数据通信系统内的桥接、开关和集聚。

现有的 Virtex–5 系列产品的主要技术特征如下表所示。

表 1：Virtex–5 FPGA 系列器件

器件	可配置逻辑块 (CLB)			DSP48E Slice[2]	Block RAM 模块			CMT[4]	PowerPC 处理器模块	PCI Express 端点模块	以太网 MAC[5]	最大 RocketIO 收发器数 [6]		I/O Bank 总数[8]	最大用户 I/O 数[7]
	阵列（行 x 列）	Virtex–5 Slice[1]	最大分布式 RAM (Kb)		18 Kb[3]	36 Kb	最大 (Kb)					GTP	GTX		
XC5VLX30	80 x 30	4,800	320	32	64	32	1,152	2	不适用	不适用	不适用	不适用	不适用	13	400
XC5VLX50	120 x 30	7,200	480	48	96	48	1,728	6	不适用	不适用	不适用	不适用	不适用	17	560
XC5VLX85	120 x 54	12,960	840	48	192	96	3,456	6	不适用	不适用	不适用	不适用	不适用	17	560
XC5VLX110	160 x 54	17,280	1,120	64	256	128	4,608	6	不适用	不适用	不适用	不适用	不适用	23	800
XC5VLX155	160 x 76	24,320	1,640	128	384	192	6,912	6	不适用	不适用	不适用	不适用	不适用	23	800
XC5VLX220	160 x 108	34,560	2,280	128	384	192	6,912	6	不适用	不适用	不适用	不适用	不适用	23	800
XC5VLX330	240 x 108	51,840	3,420	192	576	288	10,368	6	不适用	不适用	不适用	不适用	不适用	33	1,200
XC5VLX20T	60 x 26	3,120	210	24	52	26	936	1	不适用	1	2	4	不适用	7	172
XC5VLX30T	80 x 30	4,800	320	32	72	36	1,296	2	不适用	1	4	8	不适用	12	360
XC5VLX50T	120 x 30	7,200	480	48	120	60	2,160	6	不适用	1	4	12	不适用	15	480
XC5VLX85T	120 x 54	12,960	840	48	216	108	3,888	6	不适用	1	4	12	不适用	15	480
XC5VLX110T	160 x 54	17,280	1,120	64	296	148	5,328	6	不适用	1	4	16	不适用	20	680
XC5VLX155T	160 x 76	24,320	1,640	128	424	212	7,632	6	不适用	1	4	16	不适用	20	680
XC5VLX220T	160 x 108	34,560	2,280	128	424	212	7,632	6	不适用	1	4	16	不适用	20	680
XC5VLX330T	240 x 108	51,840	3,420	192	648	324	11,664	6	不适用	1	4	24	不适用	27	960
XC5VSX35T	80 x 34	5,440	520	192	168	84	3,024	2	不适用	1	4	8	不适用	12	360
XC5VSX50T	120 x 34	8,160	780	288	264	132	4,752	6	不适用	1	4	12	不适用	15	480
XC5VSX95T	160 x 46	14,720	1,520	640	488	244	8,784	6	不适用	1	4	16	不适用	19	640
XC5VSX240T	240 x 78	37,440	4,200	1,056	1,032	516	18,576	6	不适用	1	4	24	不适用	27	960
XC5VTX150T	200 x 58	23,200	1,500	80	456	228	8,208	6	不适用	1	4	不适用	40	20	680
XC5VTX240T	240 x 78	37,440	2,400	96	648	324	11,664	6	不适用	1	4	不适用	48	20	680
XC5VFX30T	80 x 38	5,120	380	64	136	68	2,448	2	1	1	4	不适用	8	12	360
XC5VFX70T	160 x 38	11,200	820	128	296	148	5,328	6	1	3	4	不适用	16	19	640
XC5VFX100T	160 x 56	16,000	1,240	256	456	228	8,208	6	2	3	4	不适用	16	20	680
XC5VFX130T	200 x 56	20,480	1,580	320	596	298	10,728	6	2	3	6	不适用	20	24	840
XC5VFX200T	240 x 68	30,720	2,280	384	912	456	16,416	6	2	4	8	不适用	24	27	960

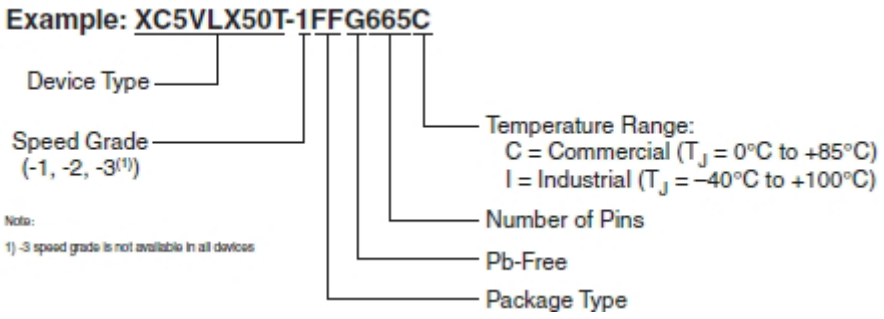
表3–14 Virtex–5系列 FPGA主要技术特征

其主要特点如下：

采用了最新的 65 工艺，结合低功耗 IP 块将动态功耗降低了 35%；此外，还利用 65nm 三栅极氧化层技术保持低静态功耗；利用 65nm ExpressFabric 技术，实现了真正的 6 输入 LUT，并将性能提高了 2 个速度级别。内置有用于构建更大型阵列的 FIFO 逻辑和 ECC 的增强型 36 Kbit Block RAM 带有低功耗电路，可以关闭未使用的存储器。 逻辑单元多达 330,000 个，可以实现无与伦比的高性能；I/O 引脚多达 1,200 个，可以实现高带宽存储器 / 网络接口，1.25 Gbps LVDS；低功耗收发器多达 24 个，可以实现 100 Mbps – 3.75 Gbps 高速串行接口；核电压为 1V，550 MHz 系统时钟；550 MHz DSP48E slice 内置有 25 x 18 MAC，提供 352 GMACS 的性能，能够在将资源使用率降低 50% 的情况下，实现单精度浮点运算；利用内置式 PCIe 端点和以太网 MAC 模块提高面积效率；更加灵活的时钟管理管道 (Clock Management Tile) 结合了用于进行精确时钟相位控制与抖

动滤除的新型 PLL 和用于各种时钟综合的数字时钟管理器 (DCM)；采用了第二代 sparse chevron 封装，改善了信号完整性，并降低了系统成本；增强了器件配置，支持商用 flash 存储器，从而降低了成本。

注：一个 Virtex–5 Slice 具有 4 个 LUT 和 4 个触发器，而一个前文所提及的常规 Slice 只包含 2 个 LUT 个 2 个触发器。每个 DSP48E 包含一个 25*18 位的硬核乘法器、一个加法器和一个累加器。



Virtex–5 FPGA 订购信息适用于所有封装，包括无铅封装。

(6)Virtex–6 FPGA 系列

基于采用第三代 Xilinx ASMBL™ 架构的 40nm 制造工艺，Virtex–6 FPGA 系列还拥有新一代开发工具和早已针对 Virtex–5 FPGA 而开发的广泛 IP 库支持。这些都为多产的开发和设计移植提供了强大的支持。 与竞争厂商提供的 40nm FPGA 产品相比，新的 Virtex–6 FPGA 系列器件性能提高 15%，功耗降低 15%。新器件在 1.0v 内核电压上操作，同时还有可选的 0.9v 低功耗版本。 这些使得系统设计师可在设计中采用 Virtex–6 FPGA，从而支持建设“绿色”中心办公室和数据中心。对于电信行业这一点特别重要，因为该行业正在扩展对因特网视频和富媒体内容的支持。

Virtex– 6 FPGA 系列包括三个面向应用领域而优化的 FPGA 平台，分别提供了不同的特性和功能组合来更好地满足不同客户应用的需求：

Virtex–6 LXT FPGA—优化目标应用需要高性能逻辑、DSP 以及基于低功耗 GTX 6.5Gbps 串行收发器的串行连接能力。

Virtex–6 SXT FPGA—优化目标应用需要超高性能 DSP 以及基于低功耗 GTX 6.5Gbps 串行收发器的串行连接能力。

Virtex–6 HXT FPGA—作为优化的通信应用需要最高的串行连接能力，多达 64 个 GTH 串行收发器可提供高达 11.2Gbps 带宽 。

Virtex–6 FPGA 把先进的硬件芯片技术、创新的电路设计技术以及架构上的增强完美结合在一起，与前一代 Virtex 器件以及竞争 FPGA 产品相比，功耗大大降低，性能更高并且成本更低。表 3–15 显示了 Virtex–6 FPGA 系列主要技术特征。

Part Number		LX75T	LX130T	LX195T	LX240T	LX365T	LX550T	LX760	SX315T	SX475T
Logic Cells		74.5K	128K	200K	241K	364K	550K	759K	315K	476K
Maximum Distributed RAM (Kbits)		1,045	1,740	3,040	3,650	4,130	6,200	8,280	5,090	7,640
Block RAM/FIFO (36Kbits each)		156	264	344	416	416	632	720	704	1,064
Total Block RAM (Kbits)		5,616	9,504	12,384	14,976	14,976	22,752	25,920	25,344	38,304
Mixed Mode Clock Manager (MMCM)		6	10	10	12	12	18	18	12	18
DSP48E1 Slices		288	480	640	768	576	864	864	1,344	2,016
PCI Express® Interface Blocks		1	2	2	2	2	2	0	2	2
10/100/1000 Ethernet MAC Blocks		4	4	4	4	4	4	0	4	4
GTX Low-Power Transceivers		12	20	20	24	24	36	0	24	36
Package	Size (Pitch)	Maximum User I/O: Select IO™ Interface Pins (GTX Transceivers)								
FF484	23 x 23 mm (1.0 mm)	240 (8)	240 (8)							
FF784	29 x 29 mm (1.0 mm)	360 (12)	400 (12)	400 (12)	400 (12)					
FF1156	35 x 35 mm (1.0 mm)		600 (20)	600 (20)	600 (20)	600 (20)				
FF1759	42.5 x 42.5mm (1.0 mm)			720 (24)	720 (24)	720 (24)	840 (36)		720 (24)	840 (36)
FF1760	42.5 x 42.5mm (1.0 mm)						1,200 (0)	1,200 (0)		

表3–15 Virtex–6 FPGA系列主要技术特征

(7)Xilinx PROM 芯片介绍

赛灵思公司的 Platform Flash PROM 能为所有型号的 Xilinx FPGA 提供非易失性存储。全系列 PROM 的容量范围为 1Mbit 到 32Mbit，兼容任何一款 Xilinx FPGA 芯片，具备完整的工业温度特性 (–40° C 到 +85° C)，支持 IEEE1149.1 所定义的 JTAG 边界扫描协议。

PROM 芯片可以分成 3.3V 核电压的 系列和 1.8V 核电压的 系列两大类，前者主要面向底端引用，串行传输数据，且容量较小，不具备数据压缩的功能；后者主要面向高端的 FPGA 芯片，支持并行配置、设计修订 (Designing Revisioning) 和数据压缩 (Compression) 等高级功能，以容量大、速度快著称，其详细参数如下表所示。

型号	容量	V_{CCINT}	封装	JTAG ISP 配置	串行 配置	并行 配置	设计 修订	数据 压缩
XCF01S	1Mbit	3.3V	VO20/VOG20	√	√			
XCF02S	2Mbit	3.3V	VO20/VOG20	√	√			
XCF04S	4Mbit	3.3V	VO20/VOG20	√	√			
XCF08P	8Mbit	1.8V	VO48/VOG48/FS48/FSG4	√	√	√	√	√
XCF16P	16Mbit	1.8V	VO48/VOG48/FS48/FSG4	√	√	√	√	√
XCF32P	32Mbit	1.8V	VO48/VOG48/FS48/FSG4	√	√	√	√	√

表3–16 赛灵思公司PROM芯片总结 (截至2008年11月数据)

该系列包含 XCF01S、XCF02S 和 XCF04S(容量分别为：1Mb、2Mb 和 4Mb)，其共同特征有 3.3V 核电压，串行配置接口以及 SOIC 封装的 VO20 封装。 内部控制信号、数据信号、时钟信号和 JTAG 信号的整体结构如图 3–2 所示。

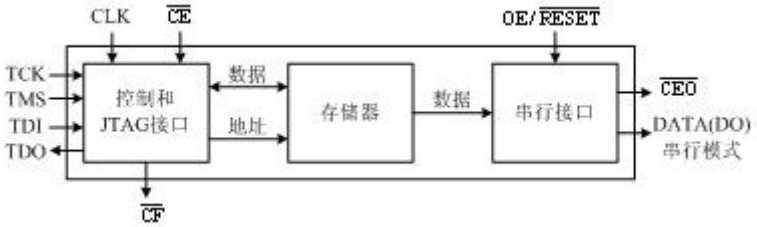


图3–2 XCF01S/XCF02S/XCF04S PROM结构组成框图

系列有 XCP08P、XCF16P 和 XCF32P(容量分别为：8Mb、16Mb 和 32Mb)，其共同特征有 1.8V 核电压、串行或并行配置接口、设计修订、内嵌的数据压缩器、FS48 封装或 VQ48 封装和内嵌振荡器。 内部控制信号、数据信号、时钟信号和 JTAG 信号的整体结构如图 3–3 所示，其先进的结构和更高的集成度在使用中带来了极大的灵活性。

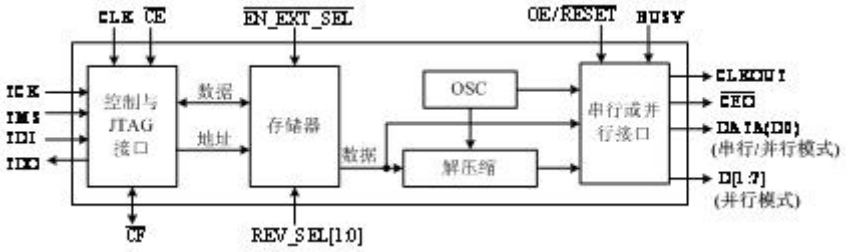


图3–3 XCP08P/XCF16P/XCF32P PROM结构组成框图

值得一提的是系列设计修正和数据压缩这两个功能。设计修订功能在 FPGA 加电启动时改变其配置数据，根据所需来改变 FPGA 的功能，允许用户在单个 PROM 中将多种配置存储为不同的修订版本，从而简化 FPGA 配置更改，在 FPGA 内部加入少量的逻辑，用户就能在 PROM 中存储多达 4 个不同修订版本之间的动态切换。数据压缩功能可以节省 PROM 的空间，最高可节约 50% 的存储空间，从而降低成本，是一项非常实用的技术。当然如果编程时在软件端采用了压缩模式，则需要一定的硬件配置来完成相应的解压缩。

第四章、FPGA开发基本流程

FPGA 是可编程芯片，因此 FPGA 的设计方法包括硬件设计和软件设计两部分。硬件包括 FPGA 芯片电路、存储器、输入输出接口电路以及其他设备，软件即是相应的 HDL 程序以及最新才流行的嵌入式 C 程序。

目前微电子技术已经发展到 SOC 阶段，即集成系统 (Integrated System) 阶段，相对于集成电路 (IC) 的设计思想有着革命性的变化。SOC 是一个复杂的系统，它将一个完整产品的功能集成在一个芯片上，包括核心处理器、存储单元、硬件加速单元以及众多的外部设备接口等，具有设计周期长、实现成本高等特点，因此其设计方法必然是自顶向下的从系统级到功能模块的软、硬件协同设计，达到软、硬件的无缝结合。

这么庞大的工作量显然超出了单个工程师的能力，因此需要按照层次化、结构化的设计方法来实施。首先由总设计师将整个软件开发任务划分为若干个可操作的模块，并对其接口和资源进行评估，编制出相应的行为或结构模型，再将其分配给下一层的设计师。这就允许多个设计者同时设计一个硬件系统中的不同模块，并为自己所设计的模块负责；然后由上层设计师对下层模块进行功能验证。

自顶向下的设计流程从系统级设计开始，划分为若干个二级单元，然后再把各个二级单元划分为下一层次的基本单元，一直下去，直到能够使用基本模块或者 IP 核直接实现为止，流行的 FPGA 开发工具都提供了层次化管理，可以有效地梳理错综复杂的层次，能够方便地查看某一层次模块的源代码以修改错误。

在工程实践中，还存在软件编译时长的问题。由于大型设计包含多个复杂的功能模块，其时序收敛与仿真验证复杂度很高，为了满足时序指标的要求，往往需要反复修改源文件，再对所修改的新版本进行重新编译，直到满足要求为止。这里面存在两个问题：首先，软件编译一次需要长达数小时甚至数周的时间，这是开发所不能容忍的；其次，重新编译和布局布线后结果差异很大，会将已满足时序的电路破坏。因此必须提出一种有效提高设计性能，继承已有结果、便于团队化设计的软件工具。FPGA 厂商意识到这类需求，由此开发出了相

应的逻辑锁定和增量设计的软件工具。例如，赛灵思公司的解决方案就是 PlanAhead。

Planahead 允许高层设计者为不同的模块划分相应 FPGA 芯片区域，并允许底层设计者在所给定的区域内独立地进行设计、实现和优化，等各个模块都正确后，再进行设计整合。如果在设计整合中出现错误，单独修改即可，不会影响到其它模块。Planahead 将结构化设计方法、团队化合作设计方法以及重用继承设计方法三者完美地结合在一起，有效地提高了设计效率，缩短了设计周期。

不过从其描述可以看出，新型的设计方法对系统顶层设计师有很高的要求。在设计初期，他们不仅要评估每个子模块所消耗的资源，还需要给出相应的时序关系；在设计后期，需要根据底层模块的实现情况完成相应的修订。

4.1 典型 FPGA 开发流程与注意事项

FPGA 的设计流程就是利用 EDA 开发软件和编程工具对 FPGA 芯片进行开发的过程。典型 FPGA 的开发流程一般如图 4.1.1 所示，包括功能定义 / 器件选型、设计输入、功能仿真、综合优化、综合后仿真、实现、布线后仿真、板级仿真以及芯片编程与调试等主要步骤。

1、功能定义/器件选型

在 FPGA 设计项目开始之前，必须有系统功能的定义和模块的划分，另外就是要根据任务要求，如系统的功能和复杂度，对工作速度和器件本身的资源、成本、以及连线的可布性等方面进行权衡，选择合适的设计方案和合适的器件类型。 一般都采用自顶向下的设计方法，把系统分成若干个基本单元，然后再把每个基本单元划分为下一层次的基本单元，一直这样做下去，直到可以直接使用 EDA 元件库为止。

2、 设计输入

设计输入是将所设计的系统或电路以开发软件要求的某种形式表示出来，并输入给 EDA 工具的过程。常用的方法有硬件描述语言 (HDL) 和原理图输入方法等。原理图输入方式是一种最直接的描述方式，在可编程芯片发展的早期应用比较广泛，它将所需的器件从元件库中调出来，画出原理图。这种方法虽然直观并易于仿真，但效率很低，且不易维护，不利于模块构造和重用。更主要的缺点是可移植性差，当芯片升级后，所有的原理图都需要作一定的改动。目前，在实际开发中应用最广的就是 HDL 语言输入法，利用文本描述设计，可以分为普通 HDL 和行为 HDL。普通 HDL 有 ABEL、CUR 等，支持逻辑方程、真值表和状态机等表达方式，主要用于简单的小型设计。而在中大型工程中，主要使用行为 HDL，其主流语言是 Verilog HDL 和 VHDL。这两种语言都是美国电气与电子工程师协会 (IEEE) 的标准，其共同的突出特点有：语言与芯片工艺无关，利于自顶向下设计，便于模块的划分与移植，可移植性好，具有很强的逻辑描述和仿真功能，而且输入效率很高。 除了这 IEEE 标准语言外，还有厂商自己的语言。也可以用 HDL 为主，原理图为辅的混合设计方式，以发挥两者的各自特色。

3、 功能仿真

功能仿真也称为前仿真是在编译之前对用户所设计的电路进行逻辑功能验证，此时的仿真没有延迟信息，仅对初步的功能进行检测。仿真前，要先利用波形编辑器和 HDL 等建立波形文件和测试向量 (即将所关心的输入信号组合成序列)，仿真结果将会生成报告文件和输出信号波形，从中便可以观察各个节点信号的变化。如果发现错误，则返回设计修改逻辑设计。常用的工具有 Model Tech 公司的 ModelSim、Sysnopsys 公司的 VCS 和 Cadence 公司的 NC-Verilog 以及 NC-VHDL 等软件。

4、 综合优化

所谓综合就是将较高级抽象层次的描述转化成较低层次的描述。综合优化根据目标与要求优化所生成的逻辑连接，使层次设计平面化，供 FPGA 布局布线软件进行实现。就目前的层次来看，综合优化 (Synthesis) 是指将设计输入编译成由与门、或门、非门、RAM、触发器等基本逻辑单元组成的逻辑连接网表，而并非真实的门级电路。真实具体的门级电路需要利用 FPGA 制造商的布局布线功能，根据综合后生成的标准门级结构网表来产生。为了能转换成标准的门级结构网表，HDL 程序的编写必须符合特定综合器所要求的风格。由于门级结构、RTL 级的 HDL 程序的综合是很成熟的技术，所有的综合器都可以支持到这一级别的综合。常用的综合工具有 Synplicity 公司的 Synplify/Synplify Pro 软件以及各个 FPGA 厂家自己推出的综合开发工具。

5、综合后仿真

综合后仿真检查综合结果是否和原设计一致。在仿真时，把综合生成的标准延时文件反标注到综合仿真模型中去，可估计门延时带来的影响。但这一步骤不能估计线延时，因此和布线后的实际情况还有一定的差距，并不十分准确。目前的综合工具较为成熟，对于一般的设计可以省略这一步，但如果在布局布线后发现电路结构和设计意图不符，则需要回溯到综合后仿真来确认问题之所在。在功能仿真中介绍的软件工具一般都支持综合后仿真。

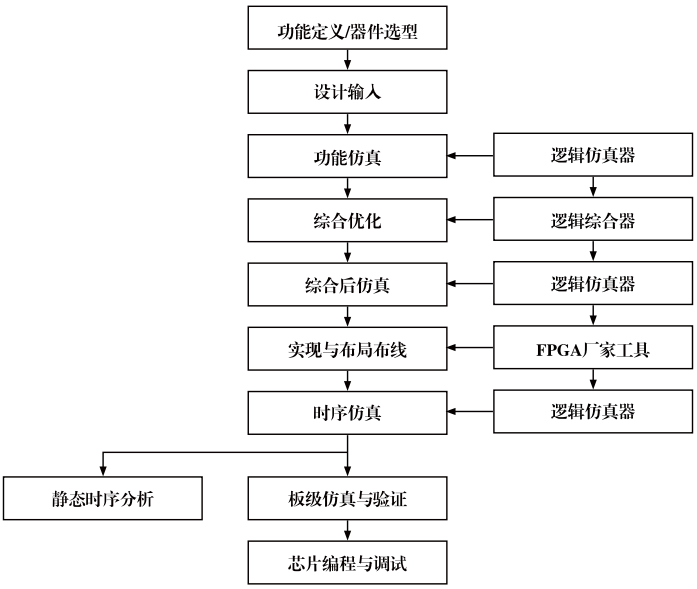


图4-1 FPGA典型设计流程

6、实现与布局布线

布局布线可理解为利用实现工具把逻辑映射到目标器件结构的资源中，决定逻辑的最佳布局，选择逻辑与输入输出功能链接的布线通道进行连线，并产生相应文件（如配置文件与相关报告），实现是将综合生成的逻辑网表配置到具体的 FPGA 芯片上，布局布线是其中最重要的过程。布局将逻辑网表中的硬件原语和底层单元合理地配置到芯片内部的固有硬件结构上，并且往往需要在速度最优和面积最优之间作出选择。布线根据布局的拓扑结构，利用芯片内部的各种连线资源，合理正确地连接各个元件。目前，FPGA 的结构非常复杂，特别是在有时序约束条件时，需要利用时序驱动的引擎进行布局布线。布线结束后，软件工具会自动生成报告，提供有关设计中各部分资源的使用情况。由于只有 FPGA 芯片生产商对芯片结构最为了解，所以布局布线必须选择芯片开发商提供的工具。

7、时序仿真

时序仿真，也称为后仿真，是指将布局布线的延时信息反标注到设计网表中来检测有无时序违规（即不满足时序约束条件或器件固有的时序规则，如建立时间、保持时间等）现象。时序仿真包含的延迟信息最全，也最精确，能较好地反映芯片的实际工作情况。由于不同芯片的内部延时不一样，不同的布局布线方案也给延时带来不同的影响。因此在布局布线后，通过对系统和各个模块进行时序仿真，分析其时序关系，估计系统性能，以及检查和消除竞争冒险是非常有必要的。在功能仿真中介绍的软件工具一般都支持综合后仿真。

8、板级仿真与验证

板级仿真主要应用于高速电路设计中，对高速系统的信号完整性、电磁干扰等特征进行分析，一般都以第三方工具进行仿真和验证。

9、芯片编程与调试

设计的最后一步就是芯片编程与调试。芯片编程是指产生使用的数据文件 (位数据流文件，Bitstream Generation)，然后将编程数据下载到 FPGA 芯片中。其中，芯片编程需要满足一定的条件，如编程电压、编程时序和编程算法等方面。逻辑分析仪 (Logic Analyzer，LA) 是 FPGA 设计的主要调试工具，但需要引出大量的测试管脚，且 LA 价格昂贵。目前，主流的 FPGA 芯片生产商都提供了内嵌的在线逻辑分析仪 (如 Xilinx ISE 中的 ChipScope、Altera QuartusII 中的 SignalTapII 以及 SignalProb) 来解决上述矛盾，它们只需要占用芯片少量的逻辑资源，具有很高的实用价值。

4.2 基于 FPGA 的 SOC 设计方法

目前，由于 FPGA 性能提升价格下降，同时嵌入越来越多内核，很自然地，很多 IC 设计公司将 FPGA 用于 ASIC 原型验证，把 FPGA 可编程的优点带到了 SOC 领域，其系统由嵌入式处理器内核、DSP 单元、大容量处理器、吉比特收发器、混合逻辑、IP 以及原有的设计部分组成。

SOC 平台的核心部分是内嵌的处理内核，其硬件是固定的，软件则是可编程的；外围电路则由 FPGA 的逻辑资源组成，大都以 IP 的形式提供，例如存储器接口、USB 接口以及以太网 MAC 层接口等，用户根据自己需要在内核总线上添加，并能自己订制相应的接口 IP 和外围设备。

基于FPGA的典型SOC开发流程为：

1、芯片内的考虑

从设计生成开始，设计人员需要从硬件 / 软件协同验证的思路入手，以找出只能在系统集成阶段才会被发现的软、硬件缺陷。然后选择合适的芯片以及开发工具，在综合过程得到优化，随后进行精确的实现，以满足实际需求。由于设计规模越来越大，工作频率也到了数百兆赫兹，布局布线的延迟将变得非常重要。为了确保满足时序，需要在布局布线后进行静态时序分析，对设计进行验证。

2、板级验证

在芯片设计完毕后，需要再进行板级验证，以便在印刷电路板 (PCB) 上保证与最初设计功能一致。因此，PCB 布局以及信号完整性测试应被纳入设计流程。由于芯片内设计所做的任何改变都将反映在下游的设计流程中，各个过程之间的数据接口和管理也必须是无误的。预计 SOC 系统以及所必须的额外过程将使数据的大小成指数增长，因此，管理各种数据集本身是急剧挑战性的任务。

第五章、FPGA实战开发技巧

5.1 FPGA 器件选型常识

作者：童 鹏、胡以华/中科院上海技术物理研究所

FPGA 器件的选型非常重要，不合理的选型会导致一系列的后续设计问题，有时甚至会使设计失败；合理的选型不光可以避免设计问题，而且可以提高系统的性价比，延长产品的生命周期，获得预想不到的经济效果。

FPGA 器件选型有以下7个原则:器件的供货渠道和开发工具的支持、器件的硬件资源、器件的电气接口标准、器件的速度等级、器件的温度等级、器件的封装和器件的价格。

5.1.1器件的供货渠道和开发工具的支持

目前，主要的 FPGA 供应商有赛灵思公司、Altera 公司、Lattic 公司和 Actel 公司等，FPGA 的发展速度非常快，很多型号的 FPGA 器件已不是主流产品，为了提高产品的生命周期，最好在货源比较足的主流器件中选型。Xllinx 公司的主流器件有 Spartan–3E、Spartan–3A、Virtex–4LX、Virtex–4 SX、Virtex–4 FX、Virtex–5 LX、Virtex–5SX、Virtex–5 FX、Spartan–6 和 Virtex – 6 等系列，其中 Spartan–3E 和 Spartan–3A 系列主要应用于逻辑设计和简单数字信号处理，Virtex–4 LX 和 Virtex–5 LX 系列主要应用于高速逻辑设计，Virtex–4 SX 和 Virtex–5 SX 系列主要应用于高速复杂数字信号处理，Virtex–4 FX 和 Virtex–5 FX 系列主要应用于嵌入式系统。

赛灵思公司有集成开发环境 ISE，Altera 公司有集成开发环境 Quartus Ⅱ，两个集成开发环境支持本公司所有器件的设计和开发。该集成开发环境不仅功能强大、界面友好，而且有很多第三方合作伙伴提供相应的技术支持，能使器件获得更高的性能。因此，如果没有特殊应用要求，建议最好在这两家公司进行器件选型。

5.1.2 器件的硬件资源

硬件资源是器件选型的重要标准。硬件资源包括逻辑资源、I / O 资源、布线资源、DSP 资源、存储器资源、锁相环资源、串行收发器资源和硬核微处理器资源等。

逻辑资源和 I / O 资源的需求是每位设计人员最关心的问题，一般都会考虑到，可是，过度消耗 I / O 资源和布线资源可能产生的问题却很容易被忽视。主流 FPGA 器件中，逻辑资源都比较丰富，一般可以满足应用需求。可是，在比较复杂的数字系统中，过度 I / O 资源的消耗可能会导致 2 个问题：FPGA 负荷过重，器件发热严重，严重影响器件的速度性能、工作稳定性和寿命，设计中要考虑器件的散热问题；局部布线资源不足，电路的运行速度明显降低，有时甚至使设计不能适配器件，设计失败。根据本人的应用经验：

(1) 在做复杂数字信号处理时，位数比较高的乘法器和除法器对全局布线资源的消耗量比较大；

(2) 在做逻辑设计时，双向 I / O 口对局部布线资源的消耗量比较大；

(3) 在利用存储器资源设计滤波器的应用场合，局部布线资源的消耗量比较大；

(4) 在电气接口标准比较多，而逻辑比较复杂的应用场合，局部布线资源的消耗量比较大。

在做乘法运算比较多而且对速度性能要求比较高的应用场合，最好能选用带 DSP 资源比较多的器件，例如，Altera 公司的 Statix Ⅱ和 Statix Ⅲ系列，赛灵思公司的 Virtex–4 SX 和 Virtex–5 SX 系列等。

器件中的存储器资源主要有 2 种用途：作高性能滤波器；实现小容量高速数据缓存。这是一种比较宝贵的硬件资源，一般器件中的存储器资源都不太多，存储器资源较多的器件逻辑容量也非常大，用得也比较少，供货渠道也不多，器件价格也非常高。因此，在器件选型时，最好不要片面追求设计的集成度而选用这种器件，可以考虑选用低端器件 + 外扩存储器的设计方案。

目前，主流 FPGA 中都集成了锁相环，利用锁相环对时钟进行相位锁定，可以使电路获得更稳定的性能。赛灵思公司提供的是数字锁相环，其优点是能获得更精确的相位控制，其缺点是下限工作频率较高，一般在 24 MHz 以上；Altera 公司提供的是模拟锁相环，其优点是下限工作频率较低，一般在 16 MHz 以上，其主流器件 Statix Ⅱ和 Statix Ⅲ系列中的增强型锁相环工作频率只要求在 4 MHz 以上，其缺点是对时钟相位的控制精度相对较差。

在通讯领域里，用光纤传输高速数据是一个比较常用的解决方案。A1tera 公司的 Statix Ⅱ GX 和 Statix Ⅲ GX 系列，赛灵思公司的 Virtex–4 FX 和 Virtex–5 FX 系列都集成了高速串行收发器 (注意：赛灵思 V5 带 T 的产品都有高速串行收发器，V4、V2P 某些型号也有高速串行收发器)，这种器件价格一般都比较高。目前，National 和 Maxim 等公司提供的高性能专用串行收发芯片价格都不高，因此，如果只是进行光纤数据传输没计，大可不必选用这种器件；如果是光纤数据传输 + 逻辑或算法比较复杂的应用场合，最好是将两种方案进行比较，然后考虑是否选用该器件。

利用集成硬核微处理器的 FPGA 器件进行嵌入式开发，代表嵌入式应用的一个方向。赛灵思公司提供集成 PowerPC™ 的 Virtex–4 FX 和 virtex–5 FX 系列器件。随着器件价格不断下降，在很多应用场合，在不增加成本的情况下，选用该器件和传统 FPGA+MCU 的应用方案相比，能大幅度提高系统性能和降低硬件设计复杂程度。此时，选用该器件是比较理想的。

5.1.3 电气接口标准

目前，数字电路的电气接口标准非常多。在复杂数字系统中，经常会出现多种电气接口标准。目前，主流 FPGA 器件支持的电气接口标准有：1. 5 V，1. 5–V 等，可以满足绝大部分应用设计需求。

HSTL ClassI, 1. 5 - V HSTL Class Ⅱ , 1. 8 V, 1. 8 - V
HSTL ClassI, 1. 8 - V HSTL Class Ⅱ , 2. 5 V, 3. 3 - V
PCI, 3. 3 - V PCI - X, 3. 3 - V PCML, AGP 1X, AGP 2X,
Compact PCI, CTT, Differential 1. 5 - V HSTL ClassI, Differential 1. 8 - V HSTL ClassI, Differential 1. 8 - V HSTL Class Ⅱ , LVPECL, LVDS, GTL, GTL +, Hyper Transport, LVCMOS, LVTTL, SSTL - 18 ClassI, SSTL - 18 Class Ⅱ , SSTL - 2 ClassI, SSTL - 2 Class Ⅱ , SSTL - 3 ClassI, SSTL - 3 Class Ⅱ , Differential SSTL - 2

赛灵思公司的 FPGA 几乎所有的管脚都支持 SSTL–2 Class Ⅱ电气接口标准，此时选用赛灵思公司的 FPGA 是比较理想的。

5.1.4 器件的速度等级

关于器件速度等级的选型，一个基本的原则是：在满足应用需求的情况下，尽量选用速度等级低的器件。该选型原则有如下好处：

(1) 由于传输线效应，速度等级高的器件更容易产生信号反射，设计要在信号的完整性上花更多的精力；

(2) 速度等级高的器件一般用得比较少，价格经常是成倍增加，而且高速器件的供货渠道一般比较少，器件的订货周期一般都比较长，经常会延误产品的研发周期，降低产品的上市率。

5.1.5 器件的温度等级

某些应用场合，对器件的环境温度适应能力提出了很高的要求，此时，就应该在有工业级甚至是军品级或宇航级的器件中进行选型。据调研，Altera 公司每种型号的 FPGA 都有工业级产品；Xllinx 公司每种型号的 FPGA 都有工业级产品，部分型号的 FPGA 提供军品级和宇航级产品。

5.1.6 器件的封装

目前，主流器件的封装形式有：QFP，BGA 和 FB–GA，BGA 和 FBGA 封装器件的管脚密度非常高，设计中必须使用多层板，PCB 布线相当复杂，设计成本比较高，器件焊接成本比较高，因此，设计中能不用尽量不用。不过，在密度非常高，集成度非常高和对 PCB 板体积要求比较高的应用场合，尽量选用 BGA 和 FBGA 封装器件。还有一种情况，在电路速度非常高的应用场合，最好选用 BGA 和 FBGA 封装器件，这 2 种封装器件由于器件管脚引线电感和分布电容比较小，有利于高速电路的设计。

5.1.7 器件的价格

器件集成度不断提高，性能不断上升，而价位不断下降是 FPGA 器件发展的普遍趋势，因此，在不断推出的新型器件中选型是一个基本规律。以赛灵思公司刚推出的 Virtex–5 为例，性能比 Virtex–4 提高 30%，而相对价位却降低 35%。

5.2 如何进行 FPGA 设计早期系统规划

作者：Ricky Su (www.rickysu.com)

这篇文章讲述了如何用工具提高效率的方法，适用程度因人而异。

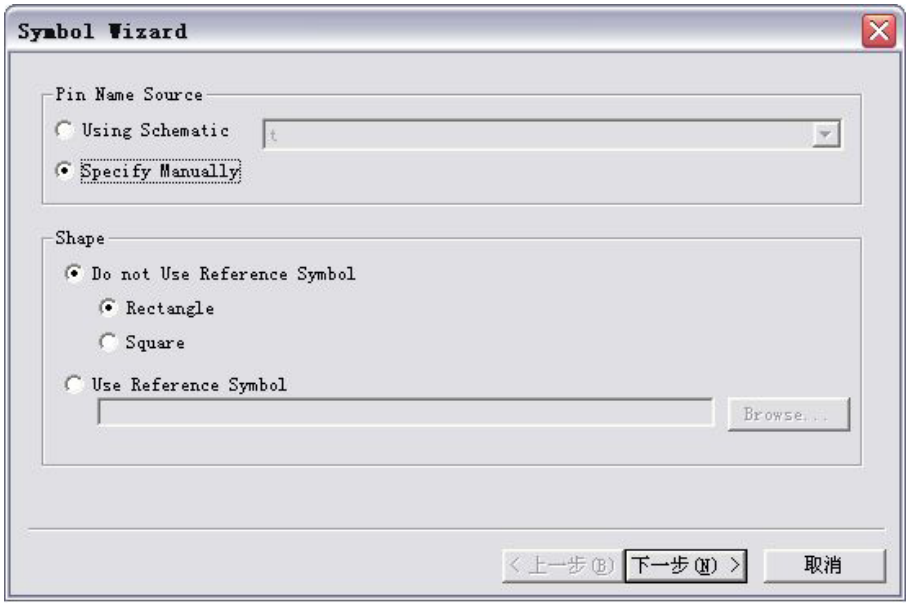
Situation: 在对 FPGA 设计进行最初步的系统规划的时候，需要进行模块划分，模块接口定义等工作。通常，我们起初会在纸上进行设计，到了一定阶段的定稿可能会输入 Visio 等工具，方便在 Team 内部交流和审阅。虽然在纸上我们可以很随意地书写，而用纸画的不方便就在于，如果想对某一个模块进行一些改动或者重画模块，那么常常因为留出的空余纸张不够，而导致拿一张新的白纸重新画一遍，比较浪费时间。对于电子化的 Visio 来说，方便修改是好处，但他不是专为设计 FPGA 系统而设计的，添加输入输出端口没那么方便，也不会根据定义的模块自动生成 HDL 文件。

Question: 我们能不能使用更好软件进行系统规划呢?

Solution: 答案是可以的。下面以 ISE 10.1 为例作说明：

1) 画一个空模块，仅定义端口 – 新建 Schematic，选择 Tools –> Symbol Wizard，里面可以定义 Symbol 名和端口属性。完成后生成 sym 格式的 Symbol。如果端口是一个 bus，那么可以用 A(4:0) 的形式。

2) 将 Symbol 添加到原理图 – 在 Schematic 的 Symbol 页面，选择 Categories 为工程文件夹，在 Symbols 列表中就可以看到刚刚新建的 Symbol。将它添加到原理图中。



3) 重复 1–2 步骤，建立所有 Symbol，并连接端口。如果需要修改连线的名字或者模块的例化名，可以选择需要修改名字的元件然后按右键 ––> Object Properties –––> 在 Name/InstName 窗格中填入需要的名字。

4) 如需修改 Symbol，可以直接在 sym 文件中修改 – 可以按右键 –> Add –> Pin 等等添加，也可以 Copy 已存在的 Pin，然后改变 PinName。但是 ISE10.1 的 Symbol Editor 对 Add Pin 有一些 Bug。因此在 UltraEditor 打开这个 sym 文件，在里面修改可能是更好的办法。sym 文件格式很易懂。改变 Symbol 端口后需要 Update

Schematic。在点到 Schematic 后会自动弹出 Update 对话框。

5) 生成原理图对应的 HDL 文件 – 点击 "Sources in Project" 列表中的 sch 文件，在 "Process" 窗口选择 "View HDL Functional Model"。这样会自动生成 Schematic 对应的 HDL 文件，其中例化了上面的各个模块。要改变 HDL 文件类型，可以改变 Project 属性中的 "Generated Simulation Language" 属性。

6) 生成 Symbol 对应的 HDL 文件 – 在打开一个 sym 文件时，选择 Tools –> Generate HDL Template from Symbol。此时可以选择生成 VHDL 还是 Verilog 的文件。

至此，我们已经生成了顶层文件和待开发的子模块文件，我们已经可以在它的基础上进行开发了。在开发过程中我们可能还会碰到这些问题：

1. 我想把设计图打印下来 – 除了 ISE 自带的打印功能外，要打印好看的图纸，还可以使用 Synplify Pro 或 PlanAhead。由于以上流程生成的代码都是可综合的，带有端口信息的 HDL 会被综合工具认为是一个 blackbox 的 wrapper，因此我们可以用 ISE 或 Synplify 将这些代码综合，综合工具会生成比较好看的综合模块图 (RTL Schematic)。除了可以用 ISE 和 Synplify 打开这些综合网表产生 RTL Schematic 之外，也可以用 PlanAhead 打开综合网表，它的 Schematic 显示功能更为强大。

2. 我要修改某些模块的端口，并添加连线修改模块端口是否还需要在原来的 Schematic 上编辑修改呢? 这是仁者见仁智者见智的问题了。我个人在生成了带端口信息的 HDL 后还是偏好修改 HDL –-> 综合 –-> 在 PlanAhead 中产生需要的连接图 –-> 打印 –-> 在打印稿上继续思考写写划划 –-> 继续修改 HDL 这样的流程。

5.3. 综合和仿真技巧

作者：田耘/云创工作室

5.3.1 综合工具XST的使用

所谓综合，就是将 HDL 语言、原理图等设计输入翻译成由与、或、非门和 RAM、触发器等基本逻辑单元的逻辑连接 (网表)，并根据目标和要求 (约束条件) 优化所生成的逻辑连接，生成 EDF 文件。XST 内嵌在 ISE 3 以后的版本中，并且在不断完善。此外，由于 XST 是赛灵思公司自己的综合工具，对于部分赛灵思芯片独有的结构具有更好的融合性。

完成了输入、仿真以及管脚分配后就可以进行综合和实现了。在过程管理区双击 Synthesize–XST，如图 5–1 所示，就可以完成综合，并且能够给出初步的资源消耗情况。图 5–2 给出了模块所占用的资源。

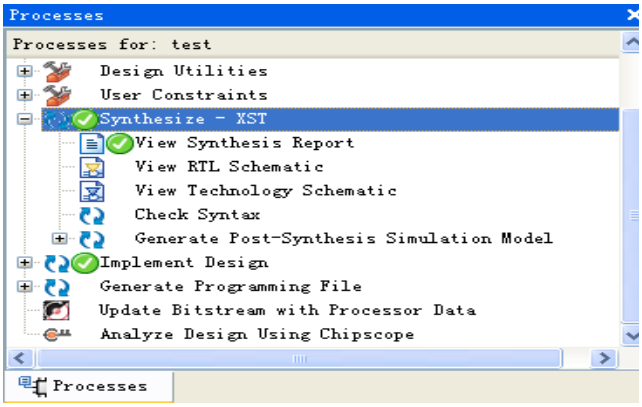


图5-1　设计综合窗口

Device Utilization Summary				
Logic Utilization	**Used**	**Available**	**Utilization**	**Note(s)**
Number of 4 input LUTs	8	30,720	1%	
Logic Distribution				
Number of occupied Slices	5	15,360	1%	
Number of Slices containing only related logic	5	5	100%	
Number of Slices containing unrelated logic	0	5	0%	
Total Number of 4 input LUTs	9	30,720	1%	
Number used as logic	8			
Number used as a route-thru	1			
Number of bonded IOBs	17	448	3%	
Number of BUFG/BUFGCTRLs	1	32	3%	
Number used as BUFGs	1			
Number used as BUFGCTRLs	0			
Total equivalent gate count for design	115			
Additional JTAG gate count for IOBs	816			

图5-2 综合结果报告

综合可能有 3 种结果：如果综合后完全正确，则在 Synthesize-XST 前面有一个打钩的绿色小圈圈；如果有警告，则出现一个带感叹号的黄色小圆圈；如果有错误，则出现一个带叉的红色小圈圈。综合完成之后，可以通过双击 View RTL Schematics 来查看 RTL 级结构图，察看综合结构是否按照设计意图来实现电路。ISE 会自动调用原理图编辑器 ECS 来浏览 RTL 结构。对于一个计数器，其 RTL 结构图如图 5-3 所示，综合结果符合设计者的意图，调用了加法器和寄存器来完成逻辑。

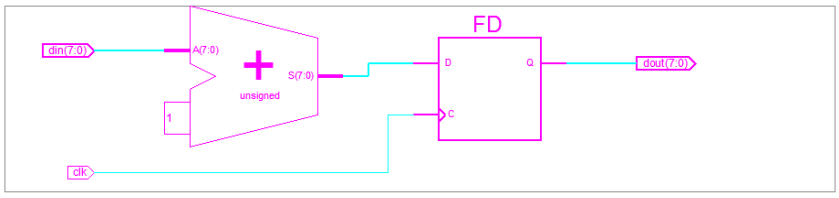


图5-3　经过综合后的RTL级结构图

2. 综合参数的设置

一般在使用 XST 时，所有的属性都采用默认值。其实 XST 对不同的逻辑设计可提供丰富、灵活的属性配置。下面对 ISE9.1 中内嵌的 XST 属性进行说明。打开 ISE 中的设计工程，在过程管理区选中“Synthesis -XST”并单击右键，弹出界面如图 5-4 所示。

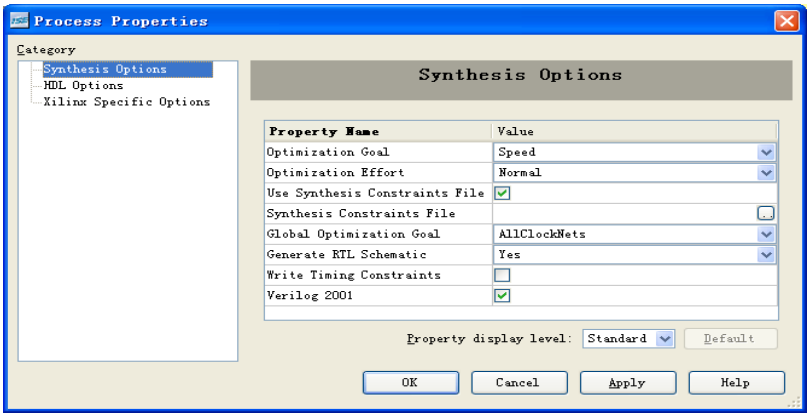


图5-4 综合选项

由图 5-4 可以看出，XST 配置页面分为综合选项 (Synthesis Options)、HDL 语言选项 (HDL Options) 以及赛灵思特殊选项 (Xilinx Specific Options) 等三大类，分别用于设置综合的全局目标和整体策略、HDL 硬件语法规则以及赛灵思特有的结构属性。

1) 综合选项参数

综合参数配置界面如图 5-4 所示，包括 8 个选项，具体如下所列：

【Optimization Goal】：优化的目标。该参数决定了综合工具对设计进行优化时，是以面积还是以速度作为优先原则。面积优先原则可以节省器件内部的逻辑资源，即尽可能地采用串行逻辑结构，但这是以牺牲速度为代价的。而速度优先原则保证了器件的整体工作速度，即尽可能地采用并行逻辑结构，但这样将会浪费器件内部大量的逻辑资源，因此，它是以牺牲逻辑资源为代价的。

【Optimization Effort】：优化器努力程度。这里有【normal】和【high】两种选择方式。对于【normal】，优化器对逻辑设计仅仅进行普通的优化处理，其结果可能并不是最好的，但是综合和优化流程执行地较快。如果选择【high】，优化器对逻辑设计进行反复的优化处理和分析，并能生成最理想的综合和优化结果，在对高性能和最终的设计通常采用这种模式；当然在综合和优化时，需要的时间较长。

【Use Synthesis Constraints File】：使用综合约束文件。如果选择了该选项，那么综合约束文件 XCF 有效。

【Synthesis Constraints File】：综合约束文件。该选项用于指定 XST 综合约束文件 XCF 的路径。

【Global Optimization Goal】：全局优化目标。可以选择的属性包括有【AllClockNets】、【Inpad To Outpad】、【Offest In Before】、【Offest Out After】、【Maximm Delay】。该参数仅对 FPGA 器件有效，可用于选择所设定的寄存器之间、输入引脚到寄存器之间、寄存器到输出引脚之间，或者是输入引脚到输出引脚之间逻辑的优化策略。

【Generate RTL Schematic】：生成寄存器传输级视图文件。该参数用于将综合结果生成 RTL 视图。

【Write Timing Constraints】：写时序约束。该参数仅对 FPGA 有效，用来设置是否将 HDL 源代码中用于控制综合的时序约束传给 NGC 网表文件，该文件用于布局和布线。

【Verilog 2001】：选择是否支持 Verilog 2001 版本。

HDL 语言选项

HDL 语言选项的配置界面如图 5-5 所示，包括 16 个选项，具体如下所列：

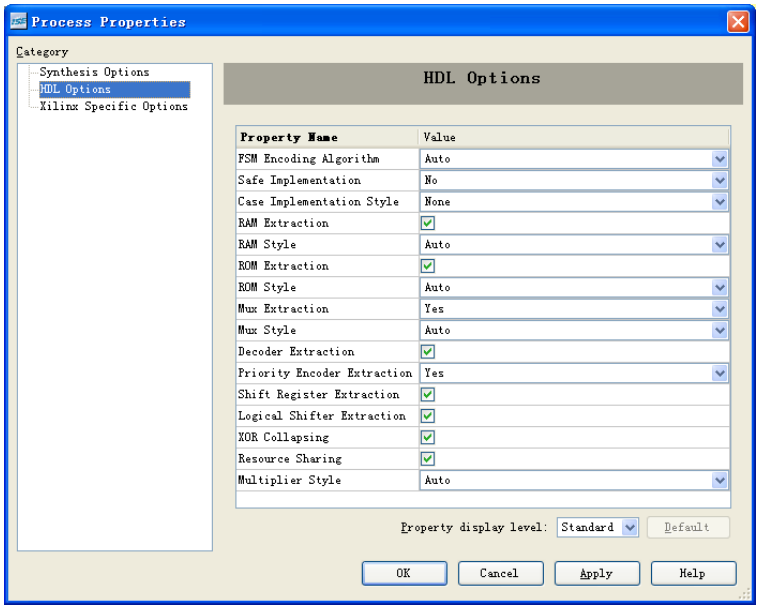


图5-5 HDL语言选项的配置界面选项

【FSM Encoding Algorithm】:有限状态机编码算法。该参数用于指定有限状态机的编码方式。选项有【Auto】、【One-Hot】、【Compact】、【Sequential】、【Gray】、【Johnson】、【User】、【Speed1】、【None】编码方式，默认为【Auto】编码方式。

【Safe Implementation】：将添加安全模式约束来实现有限状态机，将添加额外的逻辑将状态机从无效状态调转到有效状态，否则只能复位来实现，有【Yes】、【No】两种选择，默认为【No】。

【Case Implementation Sytle】：条件语句实现类型。该参数用于控制 XST 综合工具解释和推论 Verilog 的条件语句。其中选项有【None】、【Full】、【Parallel】、【Full-Parallel】，默认为【None】。 对于这四种选项，区别如下：(1)【None】，XST 将保留程序中条件语句的原型，不进行任何处理；(2)【Full】，XST 认为条件语句是完整的，避免锁存器的产生；(3)【Parallel】，XST 认为在条件语句中不能产生分支，并且不使用优先级编码器；(4)【Full-Parallel】，XST 认为条件语句是完整的，并且在内部没有分支，不使用锁存器和优先级编码器。

【RAM Extraction】：存储器扩展。该参数仅对 FPGA 有效，用于使能和禁止 RAM 宏接口。默认为允许使用 RAM 宏接口。

【RAM Style】：RAM 实现类型。该参数仅对 FPGA 有效，用于选择是采用块 RAM 还是分布式 RAM 来作为 RAM 的实现类型。默认为【Auto】。

【ROM Extraction】：只读存储器扩展。该参数仅对 FPGA 有效，用于使能和禁止只读存储器 ROM 宏接口。默认为允许使用 ROM 宏接口。

【ROM Style】：ROM 实现类型。该参数仅对 FPGA 有效，用于选择是采用块 RAM 还是分布式 RAM 来作为 ROM 的实现和推论类型。默认为【Auto】。

【Mux Extraction】：多路复用器扩展。该参数用于使能和禁止多路复用器的宏接口。根据某些内定的算法，

对于每个已识别的多路复用 / 选择器，XST 能够创建一个宏，并进行逻辑的优化。可以选择【Yes】、【No】和【Force】中的任何一种，默认为【Yes】。

【Mux Style】：多路复用实现类型。该参数用于胃宏生成器选择实现和推论多路复用 / 选择器的宏类型。可以选择【Auto】、【MUXF】和【MUXCY】中的任何一种，默认为【Auto】。

【Decoder Extraction】：译码器扩展。该参数用于使能和禁止译码器宏接口，默认为允许使用该接口。

【Priority Encoder Extraction】：优先级译码器扩展。该参数用于指定是否使用带有优先级的编码器宏单元。

【Shift Register Extraction】：移位寄存器扩展。该参数仅对 FPGA 有效，用于指定是否使用移位寄存器宏单元。默认为使能。

【Logical Shifter Extraction】：逻辑移位寄存器扩展。该参数仅对 FPGA 有效，用于指定是否使用逻辑移位寄存器宏单元。默认为使能。

【XOR Collapsing】：异或逻辑合并方式。该参数仅对 FPGA 有效，用于指定是否将级联的异或逻辑单元合并成一个大的异或宏逻辑结构。默认为使能。

【Resource Sharing】：资源共享。该参数用于指定在 XST 综合时，是否允许复用一些运算处理模块，如加法器、减法器、加 / 减法器和乘法器。默认为使能。如果综合工具的选择是以速度为优先原则的，那么就不考虑资源共享。

【Multiplier Style】：乘法器实现类型。该参数仅对 FPGA 有效，用于指定宏生成器使用乘法器宏单元的方式。选项有【Auto】、【Block】、【LUT】和【Pipe_LUT】。默认为【Auto】。选择的乘法器实现类型和所选择的器件有关。

2) 赛灵思特殊选项

赛灵思特殊选项用于将用户逻辑适配到赛灵思芯片的特殊结构中，不仅能节省资源，还能提高设计的工作频率，其配置界面如图 5–6 所示，包括 10 个配置选项，具体如下所列。

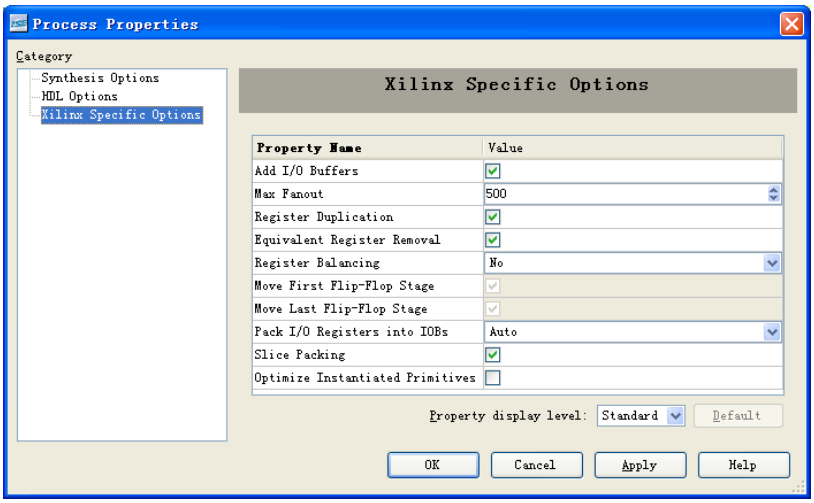


图5–6　赛灵思指定的选项

【Add I/O Buffers】：插入 I/O 缓冲器。该参数用于控制对所综合的模块是否自动插入 I/O 缓冲器。默认为自动插入。

【Max Fanout】：最大扇出数。该参数用于指定信号和网线的最大扇出数。这里扇出数的选择与设计的性能

有直接的关系，需要用户合理选择。

【Register Duplication】：寄存器复制。该参数用于控制是否允许寄存器的复制。对于高扇出和时序不能满足要求的寄存器进行复制，可以减少缓冲器输出的数目以及逻辑级数，改变时序的某些特性，提高设计的工作频率。默认为允许寄存器复制。

【Equivalent Register Removal】：等效寄存器删除。该参数用于指定是否把寄存器传输级功能等效的寄存器删除，这样可以减少寄存器资源的使用。如果某个寄存器是用赛灵思的硬件原语指定的，那么就不会被删除。默认为使能。

【Register Balancing】:寄存器配平。该参数仅对FPGA有效,用于指定是否允许平衡寄存器。可选项有【No】、【Yes】、【Forward】和【Backward】。采用寄存器配平技术，可以改善某些设计的时序条件。其中,【Forward】为前移寄存器配平,【Backward】为后移寄存器配平。采用寄存器配平后，所用到的寄存器数就会相应地增减。默认为寄存器不配平。

【Move First Flip-Flop Stage】：移动前级寄存器。该参数仅对FPGA有效，用于控制在进行寄存器配平时，是否允许移动前级寄存器。如果【Register Balancing】的设置为【No】，那么该参数的设置无效。

【Move Last Flip-Flop Stage】：移动后级寄存器。该参数仅对FPGA有效，用于控制在进行寄存器配平时，是否允许移动后级寄存器。如果【Register Balancing】的设置为【No】，那么该参数的设置无效。

【Pack I/O Registers into IOBs】：I/O寄存器置于输入输出块。该参数仅对FPGA有效，用于控制是否将逻辑设计中的寄存器用IOB内部寄存器实现。在赛灵思系列FPGA的IOB中分别有输入和输出寄存器。如果将设计中的第一级寄存器或最后一级寄存器用IOB内部寄存器实现，那么就可以缩短IO引脚到寄存器之间的路径，这通常可以缩短大约1~2ns的传输时延。默认为【Auto】。

【Slice Packing】：优化Slice结构。该参数仅对FPGA有效，用于控制是否将关键路径的查找表逻辑尽量配置在同一个Slice或者CLB模块中，由此来缩短LUT之间的布线。这一功能对于提高设计的工作频率、改善时序特性是非常有用的。 默认为允许优化Slice结构。

【Optimize Instantiated Primitives】：优化已例化的原语。该参数控制是否需要优化在HDL代码中已例化的原语。默认为不优化。

5.3.2 基于ISE的仿真

在代码编写完毕后，需要借助于测试平台来验证所设计的模块是否满足要求。ISE提供了两种测试平台的建立方法，一种是使用HDL Bencher的图形化波形编辑功能编写，另一种就是利用HDL语言，相对于前者使用简单、功能强大。下面介绍基于Verilog语言建立测试平台的方法。

首先在工程管理区将“Sources for”设置为Behavioral Simulation，在任意位置单击鼠标右键，并在弹出的菜单中选择“New Source”命令，然后选中“Verilog Test Fixture”类型，输入文件名为“test_test”，再点击“Next”进入下一页。这时，工程中所有Verilog Module的名称都会显示出来，设计人员需要选择要进行测试的模块。

用鼠标选中 test，点击“Next”后进入下一页，直接点击“Finish”按键，ISE 会在源代码编辑区自动显示测试模块的代码：

```
`timescale 1ns / 1ps
module test_test_v;
    // Inputs
    reg clk;
    reg [7:0] din;
    // Outputs
    wire [7:0] dout;

    // Instantiate the Unit Under Test (UUT)
    test uut (
            .clk(clk),
            .din(din),
            .dout(dout)
    );
    initial begin
            // Initialize Inputs
            clk = 0;
            din = 0;
            // Wait 100 ns for global reset to finish
            #100;
            // Add stimulus here
    end
    endmodule
```

由此可见，ISE 自动生成了测试平台的完整架构，包括所需信号、端口声明以及模块调用的完成。所需的工作就是在 initial…end 模块中的“// Add stimulus here”后面添加测试向量生成代码。添加的测试代码如下：

```
forever begin
    #5;
    clk = !clk;
if(clk == 1)
            din = din + 1;
    else
```

```
        din = din;
    end
```

完成测试平台后。在工程管理区将“Sources for”选项设置为 Behavioral Simulation，这时在过程管理区会显示与仿真有关的进程，如图 5-7 所示。

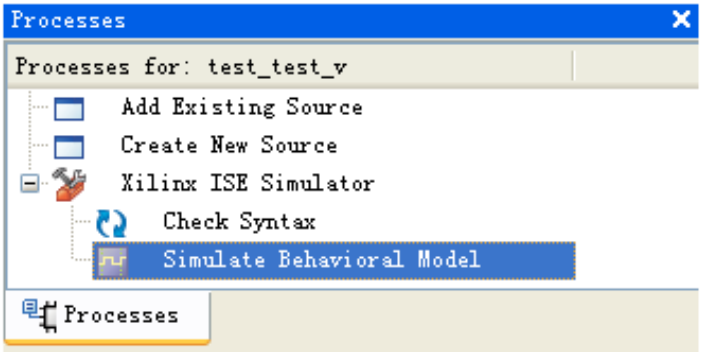


图5-7 仿真过程示意图

选中图 5-7 中 Xilinx ISE Simulator 下的 Simulate Behavioral Model 项，点击鼠标右键，选择弹出菜单的 Properties 项，会弹出如图 5-8 所示的属性设置对话框，最后一行的 Simulation Run Time 就是仿真时间的设置，可将其修改为任意时长，本例采用默认值。

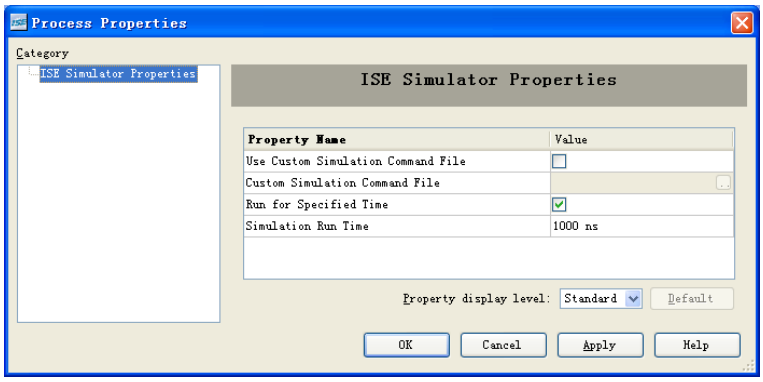


图5-8 仿真属性设置对话框

仿真参数设置完后，就可以进行仿真了，直接双击 ISE Simulator 软件中的 Simulate Behavioral Model，则 ISE 会自动启动 ISE Simulator 软件，并得到如图 5-9 所示的仿真结果，从中可以看到设计达到了预计目标。

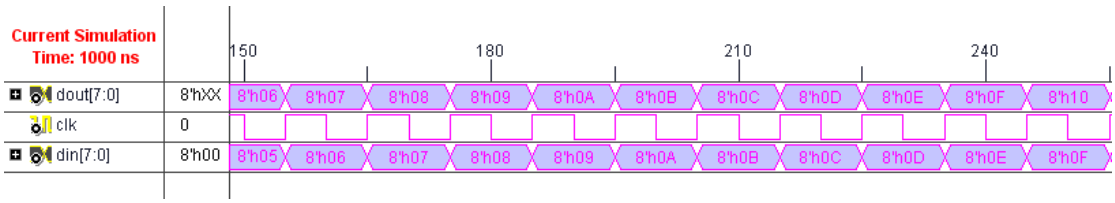


图5-9 test模块的仿真结果

5.3.3 和FPGA接口相关的设置以及时序分析

5.3.3.1 使用约束文件添加时序约束

一般来讲，添加约束的原则为先附加全局约束，再补充局部约束，而且局部约束比较宽松。其目的是在可能的地方尽量放松约束，提高布线成功概率，减少 ISE 布局布线时间。典型的全局约束包括周期约束和偏移约束。

在添加全局时序约束时，需要根据时钟频率划分不同的时钟域，添加各自的周期约束；然后对输入输出端口信号添加偏移约束，对片内逻辑添加附加约束。

1. 周期约束

周期约束是附加在时钟网路上的基本时序约束，以保证时钟区域内所有同步组件的时序满足要求。在分析时序时，周期约束能自动处理寄存器时钟端的反相问题，如果相邻的同步元件时钟相位相反，则其延迟会被自动限制为周期约束值的一半，这其实相当于降低了时钟周期约束的数值，所以在实际中一般不要同时使用时钟信号的上升沿和下降沿。

硬件设计电路所能工作的最高频率取决于芯片内部元件本身固有的建立保持时间，以及同步元件之间的逻辑和布线延迟。所以电路最高频率由代码和芯片两部分共同决定，相同的程序，在速度等级高的芯片上能达到更高的最高工作频率；同样，在同一芯片内，经过速度优化的代码具有更高的工作频率，在实际中往往取二者的平衡。

在添加时钟周期之前，需要对电路的期望时钟周期有一个合理的估计，这样才不会附加过松或过紧的周期约束，过松的约束不能达到性能要求，过紧的约束会增加布局布线的难度，实现的结果也不一定理想。常用的工程策略是：附加的时钟周期约束的时长为期望值的 90%，即约束的最高频率是实际工作频率的 110% 左右。

附加时钟周期约束的方法有两个：一是简易方法，二是推荐方法。简易方式是直接将周期约束附加到寄存器时钟网线上，其语法如下所示：

[约束信号] PERIOD = { 周期长度 } {HIGH | LOW} [脉冲持续时间];

其中，[] 内的内容为可选项，{} 中的内容为必选项，“|”表示选择项。[约束信号] 可为“Net net_name”或“TIMEGRP group_name”，前者表示周期约束作用到线网所驱动的同步元件上，后者表示约束到 TIMEGRP 所定义的信号分组上 (如触发器、锁存器以及 RAM 等)。{ 周期长度 } 为要求的时钟周期，可选用 ms、s、ns 以及 ps 等单位，默认值为 ns，对单位不区分大小写。{HIGH | LOW} 用于指定周期内第一个脉冲是高电平还是低电平。[脉冲持续时间] 用于指定第一个脉冲的持续时间，可选用 ms、s、ns 以及 ps 等单位，默认值为 ns，如果缺省该项，则默认为 50% 的占空比。如语句：

Net “clk_100MHz” period = 10ns High 5ns;

指定了信号 clk_100MHz 的周期为 10ns，高电平持续的时间为 5ns，该约束将被添加到信号 clk_100MHz 所驱动的元件上。

推荐方法常用于约束具有复杂派生关系的时钟网络，其基本语法为：

TIMESPEC “TS_identifier” = PERIOD “TNM_reference” { 周期长度 }

{HIGH | LOW} [脉冲持续时间];

其中，TIMESPEC 是一个基本时序相关约束，用于标志时序规范。“TS_identifier”由关键字 TS 和用户定义的 identifier 表示，二者共同构成一种时序规范，称为 TS 属性定义，可在约束文件中任意引用，大大地丰富了派生时钟的定义。在使用时，首先要定义时钟分组，然后再添加相应的约束，如：

NET “clk_50MHz” = “syn_clk”;

TIMESPECT “TS_syn_clk” = PERIOD “syn_clk” 20 HIGN 10;

TIMESPEC 利用识别符定义派生时钟的语法为：

TIMESPEC “TS_identifier2” = PERIOD “timegroup_name” “TS_identifier1”

[* | /] 倍数因子 [+| –] phasevalue [单位]

其中，TS_identifier2 是要派生定义的时钟，TS_identifier1 为已定义的时钟，“倍数因子”用于给出二者周期的倍数关系，phasevalue 给出二者之间的相位关系。如：

定义系统时钟 clk_syn：

TIMESPEC “clk_syn” = PERIOD “clk” 5ns;

下面给出其反相时钟 clk_syn_180 以及 2 分频时钟 clk_syn_half：

TIMESPEC “clk_syn_180” = PERIOD “clk_180” clk_syn PHASE + 2.5ns;

TIMESPEC “clk_syn_180” = PERIOD “clk_half” clk_syn / 2;

2. 偏移约束

偏移约束也是一类基本时序约束，规定了外部时钟和数据输入输出引脚之间的相对时序关系，只能用于端口信号，不能应用于内部信号，包括 OFFSET_IN_BEFORE，OFFSET_IN_AFTER，OFFSET_OUT_BEFORE，OFFSET_OUT_ AFTER 等 4 类基本约束。偏移约束的基本语法为：

OFFSET = [IN | OUT] “offset_time” [units] {BEFORE | AFTER} “clk_name”

[TIMEGRP “group_name”];

其中 [IN | OUT] 说明约束的是输入还是输出。“offset_time”为数据和有效时钟沿之间的时间差，{BEFORE | AFTER} 表明该时间差是在有效时钟之前还是之后，“clk_name”为有效时钟的名字，[TIMEGRP “group_name”] 是用户添加的分组信号，在缺省时，默认为时钟 clk_name 所驱动的所有触发器。偏移约束通知布局布线器输入数据的到达时刻，从而可准确调整布局布线的过程，使约束信号建立时间满足要求。

1)“OFFSET IN”偏移约束

“OFFSET IN ”偏移约束是输入偏移约束，有 OFFSET_IN_AFTER 和 OFFSET_IN_BEFORE 两种，前者定义了输入数据在有效时钟到达多长时间后可以到达芯片的输入管脚，这样可以得到芯片内部的延迟上限，从而对那些与输入引脚相连的组合逻辑进行约束；后者定义数据比相应的有效时钟沿提前多少时间到来，是与其相连的组合逻辑的最大延时，否则在时钟沿到来时，数据不稳定，会发生采样错误。输入偏移的时序关系如图 5–10 所示。

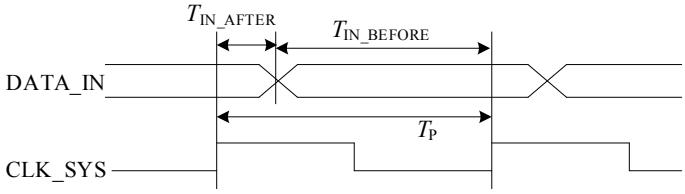


图5-10 输入偏移的时序关系

例如：

NET “DATA_IN” OFFSET = IN 10.0 BEFORE “CLK_50MHz”；

表明在时钟信号 CLK_50MHz 上升沿到达前的 10ns 内，输入信号 DATA_IN 必须到达数据输入管脚。

NET “DATA_IN” OFFSET = IN 10.0 AFTER “CLK_50MHz”；

表明在时钟信号 CLK_50MHz 上升沿到达后的 10ns 内，输入信号 DATA_IN 必须到达数据输入管脚。

2)“OFFSET OUT”偏移约束

“OFFSET OUT”偏移约束是输出偏移约束，有 OFFSET_OUT_AFTER 和 OFFSET_OUT_BEFORE 两种，前者定义了输出数据在有效时钟沿之后多长时间稳定下来，是芯片内部输出延时的上限；后者定义了在下一个时钟信号到来之前多长时间必须输出数据，是下一级逻辑建立时间的上限。输出偏移的时序关系如图5.3.11所示。

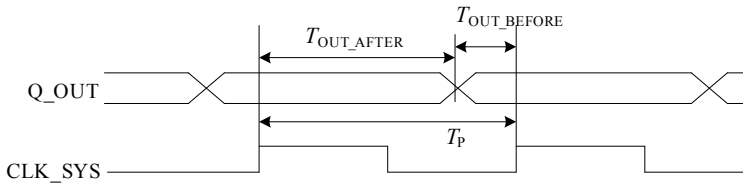


图5-11 输出偏移的时序关系

例如：

NET “DATA_OUT” OFFSET = OUT 10.0 BEFORE “CLK_50MHz”；

表明在时钟信号 CLK_50MHz 上升沿到达前的 10ns 内，输出信号 DATA_OUT 信号必须离开数据输出管脚。

NET “DATA_OUT” OFFSET = OUT 10.0 AFTER “CLK_50MHz”；

表明在时钟信号 CLK_50MHz 上升沿到达后的 10ns 内，输出信号 DATA_OUT 信号必须一直保持在数据输出管脚上。

3. 分组约束

分组约束可有效管理大量的触发器、寄存器和存储器单元，将其分为不同的组，每组附加各自的约束，在大型设计中有着广泛的应用。

1)TNM/TNM_NET 约束

TNM/TNM_NET 约束用于选出可构成一个分组的元件，并对其重新命名，然后整体添加约束。除了 IBUFG 和 BUFG 外，所有的 FPGA 内部元件都可以用 TNM 来命名，其语法规则为：

{NET|INST|PIN} “ob_name” TNM = “New_name”；

其中“ob_name”为 NET、INST 以及 PIN 的名称，New_name 为分组的名称。例如：

INST ff1 TNM = MY_FF1;

NIST ff2 TNM = MY_FF1;

将实例 ff1 与 ff2 添加到新分组 MY_FF1 中。

此外，TNM 语法也支持通配符“? ”和“*”，提高了在大规模设计中添加分组约束的效率。

当 TNM 约束附加在线网上时，则该路径上所有的同步元件都会被添加到分组中，但不会穿过 IBUFG 组件；当 TNM 约束附加到宏或原语的管脚上，则被该引脚驱动的所有同步元件会被添加到新分组中；当 TNM 约束附加到原语或宏上，则将原语或宏添加到新的分组中。

TNM_NET 约束专门用来完成网线的分组，与 TNM 不同的是，TNM 可以穿越 IBUFG/BUFG。因此，如果把 TNM 约束添加到端口上，则只能定义该端口；而要是把 TNM_NET 添加到端口上，则可穿越 BUFG，受该端口驱动的所有组件都将被添加到分组中。

2)TIMEGRP 约束

TIMEGRP 用于分组合并和拆分，将多个分组形成一个新的分组。其合并分组的语法为：

TIMEGRP “New_group” = “Old_group1” “Old_group2” … ;

其中，New_group 为新建的分组，而 Old_group1 和 Old_group2 以及…为要合并的已有分组。

拆分分组的语法为：

TIMEGRP “New_group” = “Old_group1” EXCEPT “Old_group2” ;

其中 Old_group2 是 Old_group1 的子集，New_group 为 Old_group1 中除去 Old_group2 之外所有的部分。

3)TPSYNC 约束

TPSYNC 用于将那些不是管脚和同步元件的组件定义成同步元件，以便可以利用任意点来作为时序规范的终点和起点。其相应的语法为：

{NET|INST|PIN} “ob_name” TPSYNC= “New_part” ;

将 TPSYNC 约束附加在网线上，则该网线的驱动源为同步点；附加在同步元件的输出管脚上，则同步元件中驱动该管脚的源为同步点；附加在同步元件上，则输出管脚为同步点；附加在同步元件的输入管脚上，则该引脚被定义成同步点。

4)TPTHRU 约束

TPTHRU 用于定义一个或一组路径上的关键点，可使用户定义出任意期望的路径。其相应的语法为：

{NET|INST|PIN} “ob_name” TPTHRU = “New_name” ;

例如，在图 5–12 所示场景中，从 A1 到 A2 有两条路径，其中逻辑 1 的延迟很大，需要提取出来完成特定的约束：

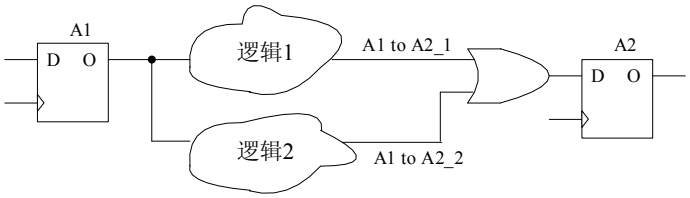


图5-12 TPTHRU约束示例场景

INST “A1” TNM = “S” ;

INST “A2” TNM = “E” ;

NET “A1toA2_1” TPTHRU = “M” ;

TIMESPEC “SME” = FROM “S” THRU “M” TO “B” 10;

其中第三句指令利用 TPTHRU 定义了中间点 “M”，然后第 4 句在此基础上定义了通过 M 点的整条路径，从两条平行的路径中挑出了期望路径。

4. 局部约束

局部约束包括 FROM_TO 约束、最大延时约束、最大偏移约束、虚假路径、系统时钟抖动约束、多周期路径和多时钟域约束等。在实际开发中，正如本章前沿所述，时序是设计出来，而不是靠约束自动得到的，因此这里不再对局部约束作过多讨论。

5.3.3.2 提高时序性能的手段

时序性能是 FPGA 设计最重要的指标之一。造成时序性能差的根本原因有很多，但其直接原因可分为三类：布局较差、逻辑级数过多以及信号扇出过高。下面通过时序分析实例来定位原因并给出相应的解决方案。

1. 布局太差及解决方案

图 5-13 所示时序报告，其中附加的周围约束为 3ns，实际周期为 3.027ns，逻辑时间只有 0.869ns，而布线延迟竟达到 2.203ns，很明显失败的原因就是布局太差。

```
Data Path: source to dest
    Delay type         Delay(ns)  Logical Resource(s)
    -------------------------  -------------------
    Tcko                  0.272   source
    net (fanout=7)        0.325   net 1
    Tilo                  0.146   lut 1
    net (fanout=1)        1.500   net 2
    Tilo                  0.146   lut 2
    net (fanout=1)        0.174   net 3
    Tilo                  0.146   lut 3
    net (fanout=1)        0.204   net 4
    Tas                   0.159   dest
    -------------------------  -----------------------------
    Total                 3.072ns (0.869ns logic, 2.203ns route)
                                  (28.3% logic, 71.7% route)
```

图5-13 布局太差的时序报告示意图

相应的解决方案有：

1) 在 ISE 布局工具中调整布局的努力程度 (effort level)；

2) 利用布局布线工具的特别努力程度 (extra effort) 或 MPPR 选项；

3) 如果用户熟悉区域约束，则利用 Floorplanner 相对区域约束 (RLOC)，重新对设计进行布局规划。

2. 逻辑级数过多及解决方案

在 FPGA 设计中，逻辑级数越高，意味着资源的利用率就越高，但对设计工作频率的影响也越大。在图 5-5 所示的例子中，附加的周围约束为 3ns，实际周期为 3.205ns，逻辑时间为 1.307ns，已经对设计的实际性能造成了一定的影响。对于这种情况，ISE 实现工具是没有任何改善的，必须通过修改代码来提高性能，相应的解决方案有：

1) 使用流水线技术，在组合逻辑中插入寄存器，简化原有的逻辑结构；

2) 检查该路径是否是多周期路径，如果是，添加相应的多周期约束；

3) 具备良好的编码习惯，不要嵌套 if 语句或 if、case 语句，并且尽量用 case 语句代替 if 语句。

```
Data Path: source to dest
    Delay type         Delay(ns)  Logical Resource(s)
    -------------------------  --------------------
    Tcko                  0.272   source
    net (fanout=7)        0.521   net 1
    Tilo                  0.146   lut 1
    net (fanout=1)        0.180   net 2
    Tilo                  0.146   lut 2
    net (fanout=1)        0.223   net 3
    Tilo                  0.146   lut 3
    net (fanout=1)        0.123   net 4
    Tilo                  0.146   lut 4
    net (fanout=1)        0.310   net 5
    Tilo                  0.146   lut 5
    net (fanout=1)        0.233   net 6
    Tilo                  0.146   lut 6
    net (fanout=1)        0.308   net 7
    Tas                   0.159   dest
    -------------------------  ---------------------------------
    Total                 3.205ns (1.307ns logic, 1.898ns route)
                                  (40.8% logic, 59.2% route)
```

图5-14 逻辑级数太多的时序报告示意图

3. 信号扇出过高及解决方案

高扇出会造成信号传输路径过长，从而降低时序性能。如图 5-15 所示，附加的周期约束为 3ns，而实际周期为 3.927ns，其中网线的扇出已经高达 187，从而导致布线时延达到 3.003ns，占实际时延的 77.64%。这种情况是任何设计所不能容忍的。

```
Data Path: source to dest
    Delay type         Delay(ns)  Logical Resource(s)
    -------------------------  --------------------
    Tcko                  0.272   source
    net (fanout=7)        0.125   net 1
    Tilo                  0.146   lut 1
    net (fanout=187)      2.500   net 2
    Tilo                  0.146   lut 2
    net (fanout=1)        0.174   net 3
    Tilo                  0.146   lut 3
    net (fanout=1)        0.204   net 4
    Tas                   0.159   dest
    -------------------------  ---------------------------
    Total                 3.872ns (0.869ns logic, 3.003ns route)
                                  (22.4% logic, 77.6% route)
```

图5-15 扇出太高的时序报告示意图

相应的解决方案有：

1) 通过逻辑复制的方法来降低信号的高扇出，可在 HDL 代码中手动复制或通过在综合工具中设置达到目的；

2) 可利用区域约束，将相关逻辑放置在一起，当然本方法仅限于高级用户。

5.3.4 综合高手揭秘XST的11个技巧

作者：Ricky Su(www.rickysu.com)

技巧 1、XST 主要参考资料：XST User Guide (ISE 安装目录 doc 中的 xst.pdf)

技巧 2、辅助参考资料：WP231 – HDL Coding Practices to Accelerate Design Performance

技巧 3、特别注意之一：请给 XST 加时序约束。

通常我们会为工程添加 UCF 约束指定时序要求和管脚约束。但是 UCF 约束是给 MAP，PAR 等实现使用的，综合工具 XST 并不能感知系统的时序要求。而为 XST 添加 XCF 约束却是使实现结果拥有最高频率的关键。其原因是显而易见的：实现工具只能在综合网表的基础上布局布线，而综合工具却可以根据要求调整综合网表，使实现工具更容易满足时序要求。如果不将时序目标告知综合器，将导致我们对性能的要求得不到体现。

XCF 约束语法与 UCF 类似并且在 XST User Guide 中有详细描述。其实常用的 Period、Offest 等约束和 UCF 的语法是一模一样的，可以直接使用在 XCF 中。

给设计添加 XCF 约束的方法是 Synthesize – XST –-> 右键 –-> Synthesis Constraint File = 指定路径

技巧 4、特别注意之二：仔细察看综合报告中的 Warning。切记要仔细查看综合报告中的所有 Warning 并确认是否是可以安全忽略的。综合器产生 Error 会使工具停止工作，但是 Warning 经常会被用户忽略。其实 Waning 可以提示很多潜在的逻辑问题，比如某些信号声明了，被使用了，却没有被赋值，或者综合器发现了 Latch 但却不是期望的结果等等。

技巧 5、常用选项之一：keep_hierarchy – 保持层次。在初始设计 /debug 的时候很有用。XST 根据层次来综合，不打破层次优化，所有的寄存器名字都以名字排列，UCF 约束可以很方便得找到需要约束的对象。如果选择 soft，则在综合时保持层次，而在 map 时工具会打破层次来优化，但是 instance 的名字还是保留的。

技巧 6、常用选项之二：register_duplication + max_fanout + equivalent_register_removal + resource_sharing – 允许自动复制寄存器，设置最大扇出，禁止资源共享。当 Timing 不满足时使用复制寄存器的方法通常能改善一些瓶颈。综合器为了节省面积而做出的某些优化可能导致对时序不利，因此关闭 equivalent_register_removal 和 resource_sharing 可能可以改善时序。

技巧 7、常用选项之三：Add IO Buffers – 自动插入缓冲器。当我们的设计作为顶层使用时，通常让

工具自动插入 IO buffer；当需要将设计作为模块插入别的设计中时，就需要禁止自动插入 IO Buffer。

技巧 8、常用选项之四：Number of Clock Buffers 和 buffer_type 约束：当综合结果中的 BUFG 不是像想象中一样时，我们可以通过下面两种方法来解决：

– 用 buffer_type 约束对该信号所使用的 Buffer 类型定义。具体使用方法在 XST User Guide

– 手动插入 BUFG，然后设置允许使用 BUFG 的数量，那么手动插入的将拥有高优先级而先占用了 BUFG，工具就不会再自动插 BUFG 了。

技巧 9、BlackBox：调用其它已经综合好的网表需要使用 BlackBox。BlackBox 说白了就是只有端口说明的 HDL 文件。更多的 BlackBox Tip 请参考我的博客 (注：为 RickySu 的博客)。

技巧 10、XST 的命令行模式：XST 支持使用命令行模式进行批量操作。

命令行的 XST 支持两种模式：

Shell 方式 – 在 cmd 下输入 xst，然后在 xst 的 shell 环境中一条一条打命令；

Script 方式 – 在 cmd 下用 xst –ifn script.scr 运行 script.scr 内的命令，或者在 xst shell 中用 script 命令调用 script.scr 中的内容。在此之前，会需要先准备好 compile_list.prj。EDK 其实就使用这种方法调用 XST。更详细的语法参考 XST User Guide。

技巧 11、要查看综合后的网表，除了 XST 自带的 RTL Schematic 工具和 Technology Schematic 工具，还可以使用 PlanAhead。他的显示 / 查找能力更为强大，而且他会先合并所有的综合网表，不会因为某个模块式预先综合好的而不能察看内部状况。

5.4 大规模设计带来的综合和布线问题

FPGA 设计的时序性能是由物理器件、用户代码设计以及 EDA 软件共同决定的，忽略了任何一方面的因素，都会对时序性能有很大的影响。本节主要给出大规模设计中，赛灵思物理器件和 EDA 软件的最优使用方案。

1)IO 约束技巧

优秀设计必须要考虑 IO 约束的技巧。对于赛灵思器件来讲，进位链是垂直分布的、逻辑排列块之间也有水平方向上三态缓冲线的直接连接、且硬核单元基本都是按列分布 (在水平方向就具备最短路径)，因此最优的方案为：将用于控制信号的 I/O 置于器件的顶部或底部，且垂直布置；数据总线的 I/O 置于器件的左部和右部，且水平布置，如图 5–16 所示。

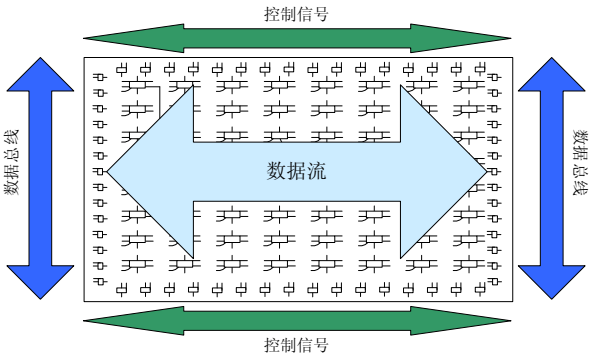


图5–16 赛灵思器件的最佳IO布局示意图

这种IO分配方式充分利用了Xilinx FPGA芯片架构的特点，如进位链排列方式以及块RAM、硬核乘法器位置。进位链的结构如图 5–17 所示。能解决多位宽加法、乘法从最低位向最高位的进位延时问题；块 RAM 和硬核乘法器可节约大量的逻辑资源且保证时序，二者在 FPGA 芯片中都是自上而下成条状分布，因此数据流水平、控制流垂直可最大限度地利用芯片底层架构。当然，在实际系统设计中，可能无法完全做到上述要求，但还是应该尽可能地将高速率、多位宽的信号布置在芯片左右两侧。

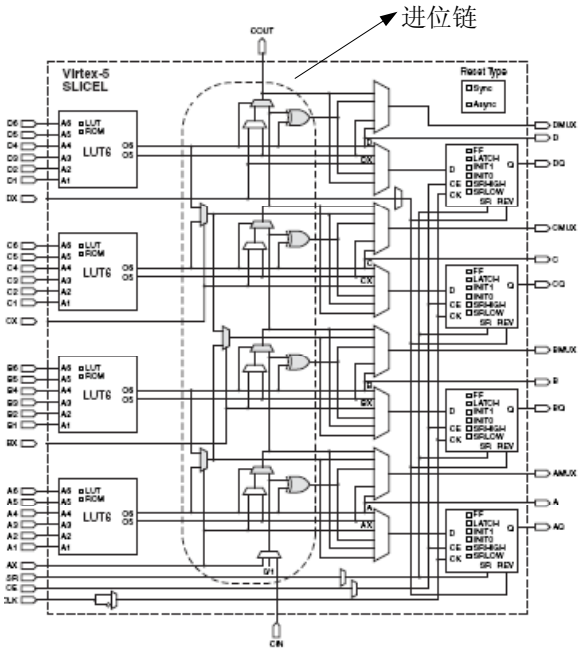


图5–17 赛灵思器件的进位链结构示意图

2)ISE 的实现工具

ISE 中集成的实现工具具备不同的努力程度 (Effort Level)，当然使用最高级别的，可以提高时序性能，而不必采取其它措施 (如施加更高级的时序约束,使用高级工具或者更改代码等),但需要花费很长的计算时间。为此，赛灵思推荐的最佳流程如图 5–18 所示。

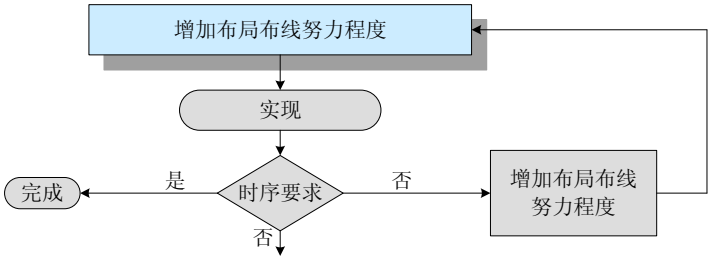


图5–18 赛灵思实现工具的最佳使用策略

在第一遍实现时，使用全局时序约束和缺省的实现参数选项，如果不能满足时序要求，则可尝试以下方法：

(1) 尝试修改代码，如使用合适的代码风格，增加流水线等 ；

(2) 修改综合参数选项，如 Optimization Effort，Use Synthesis Constraints File，Keep Hierarchy，Register Duplication，Register Balancing 等 ；

(3) 增加实现工具的努力程度 ；

(4) 在综合和实现时采用指定路径时序约束的方法。

实现工具分为映射 (MAP) 和布局布线 (PAR) 两部分，和 PAR 一样，也可使用 Map–timing 参数选项针对关键路径进行约束。如 ：参数“Timing–Driven Packing and Placement”给关键路径以优先时序约束的权利 ；用

户约束通过翻译 (Translate) 过程从 User Constraints File (UCF) 中传递到设计中。

5.5 FPGA 相关电路设计知识

FPGA 的相关电路主要就是 FPGA 的配置电路，其余的应用电路只要将外围芯片连接到 FPGA 的通用 I/O 管脚上即可。

5.5.1 配置电路

FPGA 配置方式灵活多样，根据芯片是否能够自己主动加载配置数据分为主模式、从模式以及 JTAG 模式。典型的主模式都是加载片外非易失 (断电不丢数据) 性存储器中的配置比特流，配置所需的时钟信号 (称为 CCLK) 由 FPGA 内部产生，且 FPGA 控制整个配置过程。从模式需要外部的主智能终端 (如处理器、微控制器或者 DSP 等) 将数据下载到 FPGA 中，其最大的优点就是 FPGA 的配置数据可以放在系统的任何存储部位，包括：Flash、硬盘、网络，甚至在其余处理器的运行代码中。JTAG 模式为调试模式，可将 PC 中的比特文件流下载到 FPGA 中，断电即丢失。此外，目前赛灵思还有基于 Internet 的、成熟的可重构逻辑技术 System ACE 解决方案。

(1) 主模式

在主模式下，FPGA 上电后，自动将配置数据从相应的外存储器读入到 SRAM 中，实现内部结构映射；主模式根据比特流的位宽又可以分为：串行模式 (单比特流) 和并行模式 (字节宽度比特流) 两大类。如：主串行模式、主 SPI Flash 串行模式、内部主 SPI Flash 串行模式、主 BPI 并行模式以及主并行模式，如图 5–19 所示。

(2) 从模式

在从模式下，FPGA 作为从属器件，由相应的控制电路或微处理器提供配置所需的时序，实现配置数据的下载。从模式也根据比特流的位宽不同分为串、并模式两类，具体包括：从串行模式、JTAG 模式和从并行模式三大类，其概要说明如图 5–20 所示。

(3)JTAG 模式

在 JTAG 模式中，PC 和 FPGA 通信的时钟为 JTAG 接口的 TCLK，数据直接从 TDI 进入 FPGA，完成相应功能的配置。

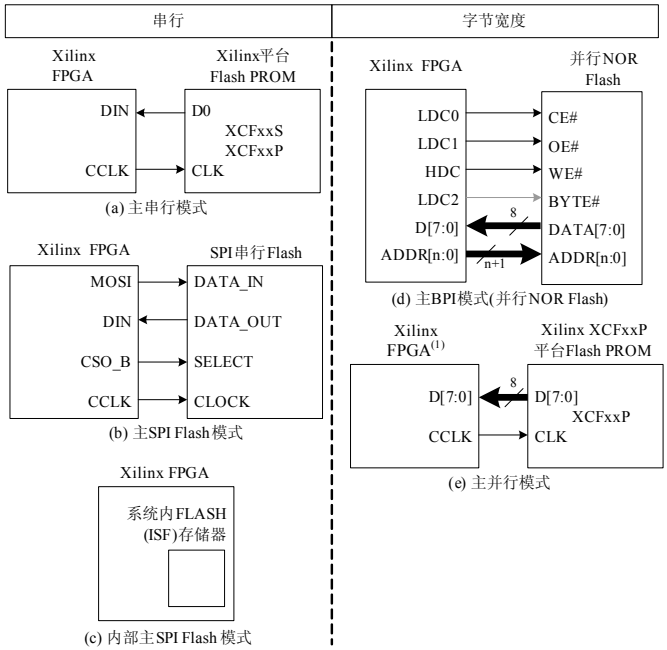


图5-19 常用主模式下载方式示意图

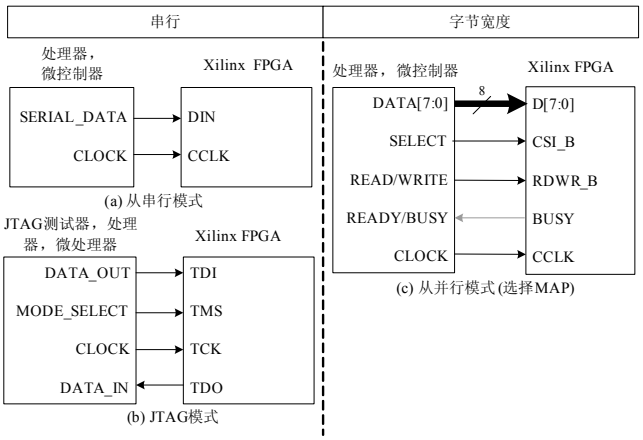


图5-20 常用的从模式下载方式示意图

目前，主流的 FPGA 芯片都支持各类常用的主、从配置模式以及 JTAG，以减少配置电路失配性对整体系统的影响。在主配置模式中，FPGA 自己产生时钟，并从外部存储器中加载配置数据，其位宽可以为单比特或者字节；在从模式中，外部的处理器通过同步串行接口，按照比特或字节宽度将配置数据送入 FPGA 芯片。此外，多片 FPGA 可以通过 JTAG 菊花链的形式共享同一块外部存储器，同样一片 / 多片 FPGA 也可以从多片外部存储器中读取配置数据以及用户自定义数据。

Xilinx FPGA 的常用配置模式有 5 类：主串模式、从串模式、Select MAP 模式、Desktop 配置和直接 SPI 配置。在从串配置中，FPGA 接收来自于外部 PROM 或其它器件的配置比特数据，在 FPGA 产生的时钟 CCLK 的作用下完成配置，多个 FPGA 可以形成菊花链，从同一配置源中获取数据。Select MAP 模式中配置数据是并行的，是速度最快的配置模式。SPI 配置主要在具有 SPI 接口的 FLASH 电路中使用。下面以 Spartan-3E 系列芯片为例，给出各种模式的配置电路。

5.5.2 主串模式——最常用的FPGA配置模式

1. 配置单片FPGA

在主串模式下，由 FPGA 的 CCLK 管脚给 PROM 提供工作时钟，相应的 PROM 在 CCLK 的上升沿将数据从 D0 管脚送到 FPGA 的 DIN 管脚。无论 PROM 芯片类型 (即使其支持并行配置)，都只利用其串行配置功能。Spartan3E 系列 FPGA 的单片主串配置电路如图 5–21 所示。主串模式是赛灵思公司各种配置方式中最简单，也最常用的方式，基本所有的可编程芯片都支持主串模式。

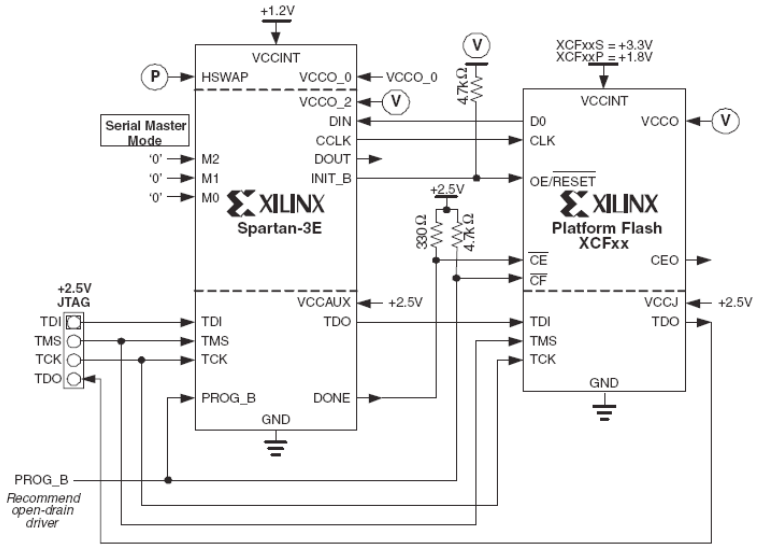


图5–21 Spartan–3E主串模式配置电路

2. 配置电路的关键点

主串配置电路最关键的 3 点就是 JTAG 链的完整性、电源电压的设置以及 CCLK 信号的考虑。只要这 3 步任何一个环节出现问题，都不能正确配置 PROM 芯片。

(1)JTAG 链的完整性

FPGA 和 PROM 芯片都有自身的 JTAG 接口电路，所谓的 JTAG 链完整性指的是将 JTAG 连接器、FPGA、PROM 的 TMS、TCK 连在一起，保证从 JTAG 连接器 TDI 到其 TDO 之间，形成 JTAG 连接器的“TDI → (TDI~TDO) → (TDI~TDO) → JTAG 连接器 TDO”的闭合回路，其中 (TDI~TDO) 为 FPGA 或者 PROM 芯片自身的一对输入、输出管脚。图 5–12 中配置电路的 JTAG 链从连接器的 TDI 到 FPGA 的 TDI，再从 FPGA 的 TDO 到 PROM 的 TDI，最后从 PROM 的 TDO 到连接器的 TDO，形成了完整的 JTAG 链，FPGA 芯片被称为链首芯片。也可以根据需要调换 FPGA 和 PROM 的位置，使 PROM 成为链首芯片。

(2) 电源适配性

如图 5–22 所示，由于 FPGA 和 PROM 要完成数据通信，二者的接口电平必须一致，即 FPGA 相应分组的管脚电压 Vcco_2 必须和 PROM Vcco 的输入电压大小一致，且理想值为 2.5V，这是由于 FPGA 的 PROG_B 和 DONE 管脚由 2.5V 的 Vccaux 供电。此外，由于 JTAG 连接器的电压也由 2.5V 的 Vccaux 提供，因此 PROM

的 VCCJ 也必须为 2.5V。因此，如果接口电压和参考电压不同，在配置阶段需要将相应分组的管脚电压和参考电压设置为一致;在配置完成后，再将其切换到用户所需的工作电压。当然，FPGA 和 PROM 也可以自适应 3.3V 的 I/O 电平以及 JTAG 电平，但需要进行一定的改动，即添加几个外部限流电阻，如图 5–22 所示。在主串模式下，XCFxxS 系列 PROM 的核电压必须为 3.3V，XCFxxP 系列 PROM 的核电压必须为 1.8V。

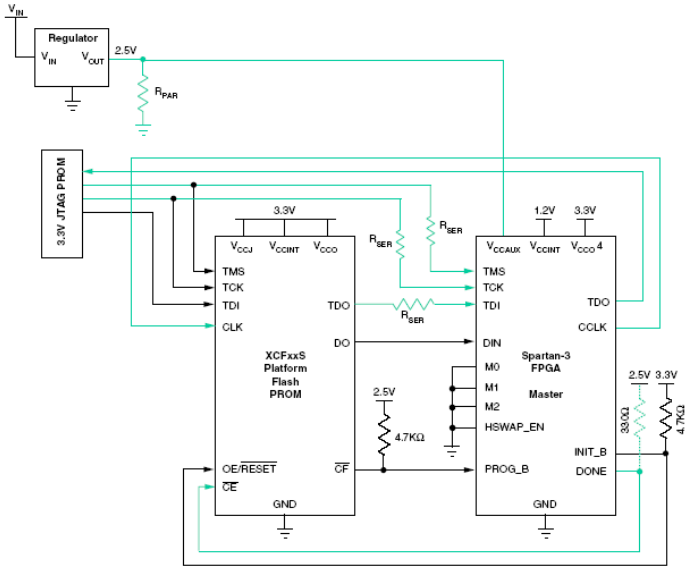


图5–22 3.3V的JTAG配置电路示意图

图 5–22 中的 RSER、RPAR 这两个电阻要特别注意。首先，RSER= 68Ω 将流入每个输入的电流限制到 9.5 mA；其次，N = 3 三个输入的二极管导通，

RPAR = VCCAUX min/ NIIN = 2.375V/(3*9.5mA)

=83 Ω 或 82 Ω（与标准值误差小于 5% 的电阻）

(3)CCLK 的信号完整性

CCLK 信号是 JTAG 配置数据传输的时钟信号，其信号完整性非常关键。FPGA 配置电路刚开始以最低时钟工作，如果没有特别指定，将逐渐提高频率。CCLK 信号是由 FPGA 内部产生的，对于不同的芯片和电平，其最大值如表 F–1 所示。

PROM 芯片型号	I/O 电压 (FPGA Vcco2 或 PROM Vcco)	SPARTAN–3E FPGA 最大 配置时钟频率 (MHz)
XCF01S XCF02S XCF04S	3.3V or 2.5V	25
	1.8V	12
XCF08P XCF16P XCF32P	3.3V, 2.5V, or 1.8V	25

表5–1 不同PROM芯片的最大配置时钟频率

3. 配置多片FPGA

多片 FPGA 的配置电路和单片的类似，但是多片 FPGA 之间有主 (Master)、从 (Slave) 之分，且需要选择不同的配置模式。两片 Spartan 3E 系列 FPGA 的典型配置电路如图 5–23 所示，两片 FPGA 存在主、从地位之分。

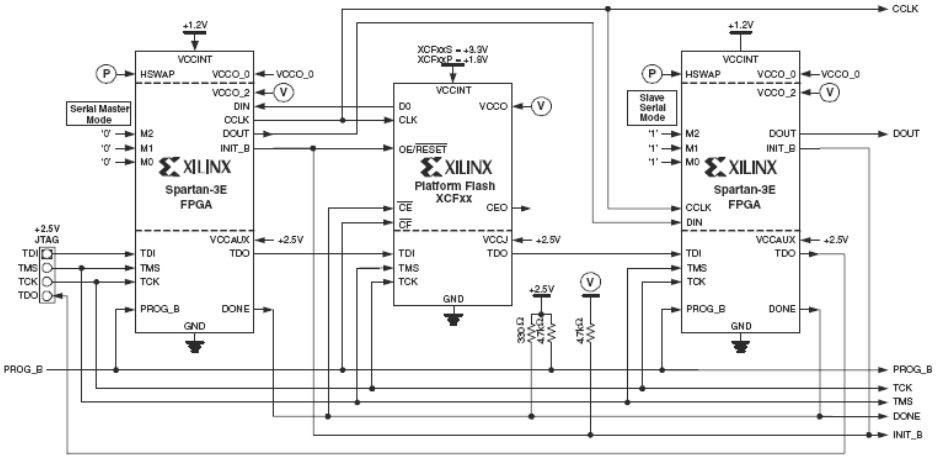


图5–23 主从模式下两片FPGA的配置电路

5.5.3 SPI串行Flash配置模式

1. SPI串行配置介绍

串行 Flash 的特点是占用管脚比较少，作为系统的数据存贮非常合适，一般都是采用串行外设接口 (SPI 总线接口)。Flash 存贮器与 EEPROM 根本不同的特征就是 EEPROM 可以按字节进行数据的改写，而 Flash 只能先擦除一个区间，然后改写其内容。一般情况下，这个擦除区间叫做扇区 (Sector)，也有部分厂家引入了页面 (Page) 的概念。选择 Flash 产品时，最小擦除区间是比较重要的指标。在写入 Flash 时，如果写入的数据不能正好是一个最小擦除区间的尺寸，就需要把整个区间的数据全部保存另外一个存贮空间，擦除这个空间，然后才能重新对这个区间改写。大多数 Flash 工艺更容易实现较大的擦除区间，因此较小擦除区间的 Flash 其价格一般会稍贵一些。此外，SPI 是标准的 4 线同步串行双向总线，提供控制器和外设之间的串行通信数据链路，广泛应用于嵌入式设备中。

赛灵思公司的新款 FPGA 都支持 SPI 接口。SPI 总线通过 4 根信号线来完成主、从之间的通信，典型的 SPI 系统中常包含一个主设备以及至少一个从设备，在 FPGA 应用场合中，FPGA 芯片为主设备，SPI 串行 FLASH 为从设备。4 个 SPI 接口信号的名称和功能如表 5–2 所示。

SPI 接口信号名	信号功能描述
SCLK	SPI 接口工作的串行时钟
MOSI	从主机到从机的数据信号，用于将主机的执行代码和数据发送到从机上
MISO	从从机到主机的数据信号，用于收集从机的所传输的数据信号
SS_n	从机片选信号，低电平有效。当其为高电平时，放弃对从机的控制，并将 MISO 端口置为高阻状态

表5-2 SPI接口信号列表

一个主芯片和一个从芯片的通信接口如图 5-24 所示。FPGA 通过 SCLK 控制双方通信的时序，在 SS_n 为低时，FPGA 通过 MOSI 信号线将数据传送到 FLASH，在同一个时钟周期中，FLASH 通过 SOMI 将数据传输到 FPGA 芯片。无论主、从设备，数据都是在时钟电平跳转时输出，并在下一个相反的电平跳转沿，送入另外一个芯片。

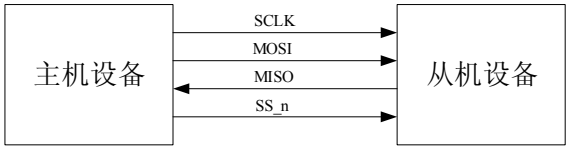


图5-24 SPI接口连接示意图

其中 SCLK 信号支持不同的速率，一般常采用 20MHz。通过 SPI 接口中的 CPOL 和 CPHA 这两个比特定义了 4 种通信时序。其中，CPOL 信号定义了 SCLK 的空闲状态，当 CPOL 为低时，SCLK 的低电平为空闲状态，否则其空闲状态为高电平；CPHA 定义了数据有效的上升沿位置，当其为低时，数据在第 1 个电平跳转沿有效，否则数据在第 2 个电平跳转沿有效。其相应的时序逻辑如图 5-25 所示。

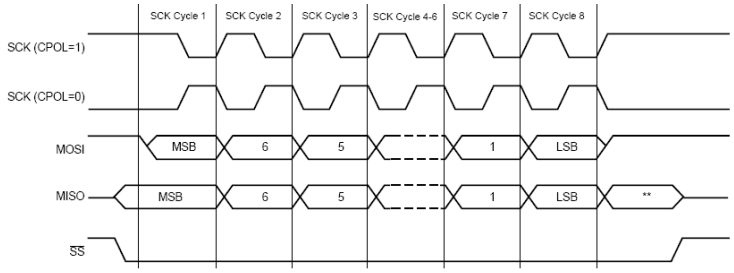


图5-27 CPHA为低时SPI的总线时序示意图

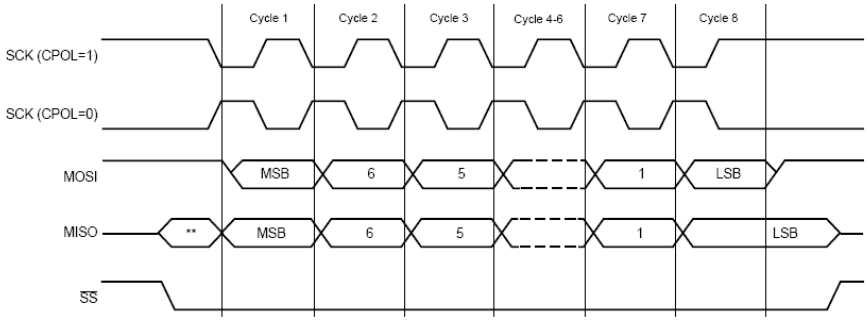


图5-28 CPHA为高时SPI的总线时序示意图

可以通过增加片选信号 SS_n 的位宽来支持多个从设备，SS_n 的位宽等于从设备的个数。对于某时刻被选中的从设备和主设备而言，其读写时序逻辑和图 5-29 一样。

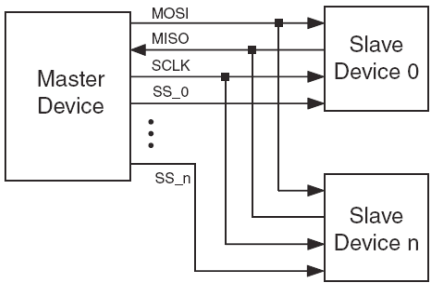


图5-29 多个从芯片的连接电路图

SPI 串行 FLASH 作为一种新兴的高性能非易失性存储器，其有效读写次数高达百万次，不仅引脚数量少、封装小、容量大，可以节约电路板空间，还能够降低功耗和噪声。从功能上看，可以用于代码存储以及大容量的数据和语音存储，对于以读为主，仅有少量擦写和写入时间的应用来说，支持分区（多页）擦除和页写入的串行存储是最佳方案。

2. SPI串行FLASH配置电路

SPI 串行配置模式常用于已采用了 SPI 串行 FLASH PROM 的系统，在上电时将配置数据加载到 FPGA 中，这一过程只需向 SPI 串行发送一个 4 字节的指令，其后串行 FLASH 中的数据就像 PROM 配置方式一样连续加载到 FPGA 中。一旦配置完成，SPI 中的额外存储空间还能用于其它应用目的。

1)SPI 配置电路

虽然 SPI 接口是标准的 4 线接口，但不同的 SPI FLASH PROM 芯片采用了不同的指令协议。FPGA 芯片通过变量选择信号 VS[2:0] 来定义 FPGA 和 SPI FLASH 的通信方式、FPGA 的读指令以及在有效接收数据前插入的冗余比特数。常用 SPI FLASH 与 FPGA 的有效操作配置如表 5-3 所示，其余的 VS[2:0] 配置留有它用。

SPI Flash 型号		ISE 是否支持	读命令			容量 (Mbits)								
			快速读(0X0B)	读(0X03)	读阵列(0XE8)									
			FPGA VS[2:0] 设置											
厂家	系列		1:1:1	1:0:1	1:1:0	0.512	1	2	4	8	16	32	64	128
STMicro	M25P	*	*	*		*	*	*	*	*	*	*	*	*
	M25PE	*	*	*			*	*	*	*				
	M45PE	*	*	*			*	*	*	*				
Atmel	AT45DB_D	*	*	*	*		*	*	*	*	*	*	*	
	AT45DB_B	*			*		*	*	*	*				

表5-3 赛灵思芯片所支持的SPI FLASH存储器以及配置列表

从整体上看来，控制 SPI 串行闪存比较容易，只需要使用简单的指令就能完成读取、擦除、编程、写使能 / 禁止以及其它功能。所有的指令都是通过 4 个 SPI 引脚串行移位输入的。

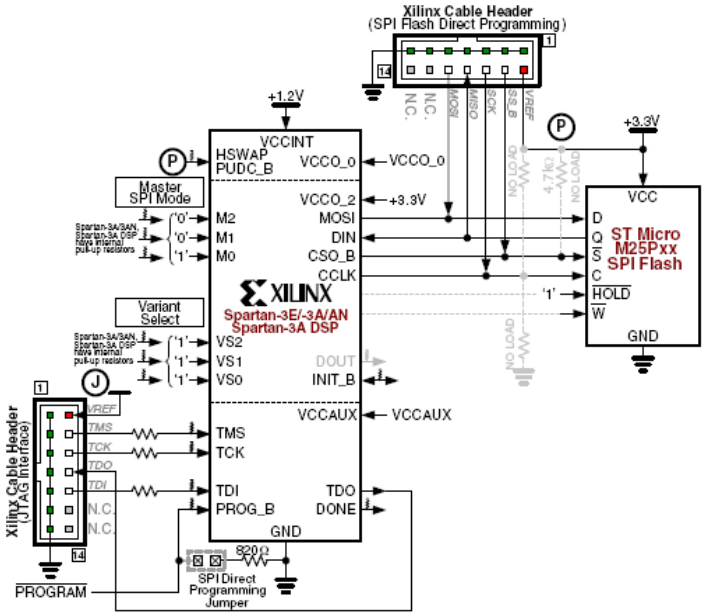


图5-30 支持快读写的串行FLASH配置电路示意图

不同型号的 FPGA 芯片具有数目不同的从设备片选信号，因此所挂的串行芯片数目也就不一样。例如：Spartan-3E 系列 FPGA 芯片只有 1 位 SPI 从设备片选信号，因此只能外挂一片 SPI 串行 FLASH 芯片。在 SPI 串行 FLASH 配置模式下，M[2:0]=3’b001。FPGA 上电后，通过外部 SPI 串行 FLASH PROM 完成配置，配置时钟信号由 FPGA 芯片提供时钟信号，支持两类业界常用的 FLASH。

图 5-30 给出了 Spartan3E 系列 FPGA 支持 0X0B 快速读写指令的 STMicro 25 系列 PROM 的典型配置电路。其中的 Flash 芯片需要 Flash 编程器来加载配置数据；单片的 FPGA 芯片构成了完整的 JTAG 链，仅用来测试芯片状态，以及支持 JTAG 在线调试模式，与 SPI 配置模式没有关系。

从中可以看出，SPI Flash 容量大，适合于大规模设计场合。但由于 SPI 配置需要专门的 Flash 编程器，且操作起来比较麻烦，不适合在产品研发阶段调试 FPGA 芯片，因此一般还会添加 JTAG 链专门用于在线调试。

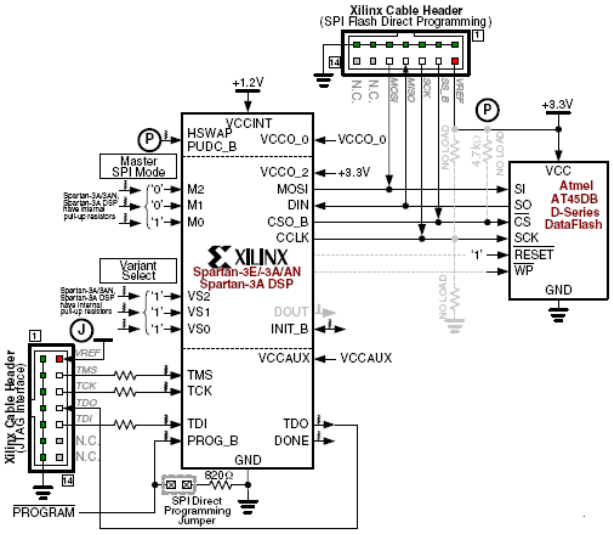


图5–31 Atmel SPI串行FLASH配置电路示意图

图 5–31 给出了 Spartan3E 系列 FPGA 支持 SPI 协议的 Atmel 公司“C”、“D”系列串行 Flash 芯片的典型配置电路。这两个系列的 FLASH 芯片可以工作在很低温度，具有短的时钟建立时间。同样，单片的 FPGA 芯片构成了完整的 JTAG 链，仅用来测试芯片状态，以及支持 JTAG 在线调试模式，与 SPI 配置模式没有关系。

表 5–3 给出了 SPI 配置接口的连线说明，每个 SPI Flash PROM 采用的名字略有不同，SPI Flash PROM 的写保护信号和保持控制信号在 FPGA 配置阶段是不用的。其中 HOLD 管脚在配置阶段必须为高，为了编程 Flash 存储器，写保护信号必须为高。

5.5.4 从串配置模式

在串行模式下，需要微处理器或微控制器等外部主机通过同步串行接口将配置数据串行写入 FPGA 芯片，其模式选择信号 M[2:0]=3’b111。典型的 Spartan 3E 系列 FPGA 单片配置电路如图 5.5.11 所示。DIN 输入管脚的串行配置数据需要在外部时钟CCLK信号前有足够的建立时间。其中单片 FPGA 芯片构成了完整的 JTAG 链，仅用来测试芯片状态，以及支持 JTAG 在线调试模式，与从串配置模式没有关系。外部主机通过下拉 PROG_B 启动配置并检测 INIT_B 电平，当 INIT_B 为高时，表明 FPGA 做好准备，开始接收数据。此时，主机开始提供数据和时钟信号直到 FPGA 配置完毕且 DONE 管脚为高，或者 INIT_B 变低表明发生配置错误才停止。整个过程需要比配置文件大小更多的时钟周期，这是由于部分时钟用于时序建立，特别当 FPGA 被配置为等待 DCM 锁存其时钟输入。

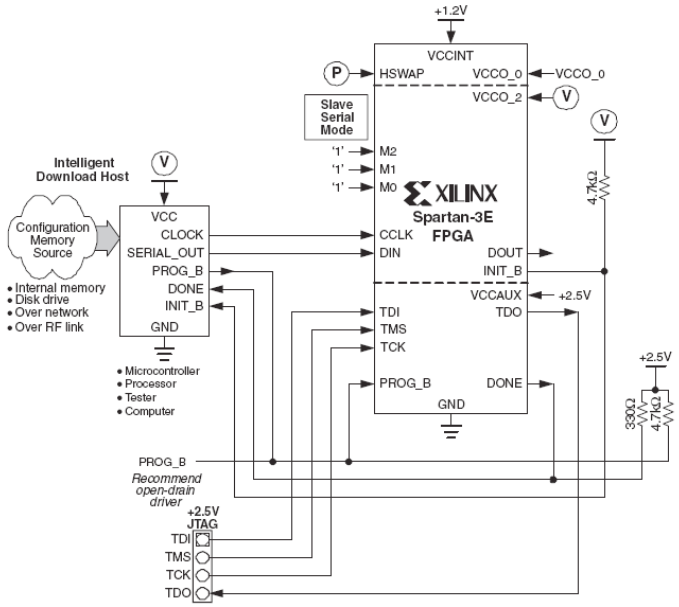


图5–32　FPGA从串配置电路示意图

此外，从串配置模式也可配置多片 FPGA 芯片，典型的两片 Spartan 3E 系列 FPGA 的从串配置电路如图 5–33 所示。所有芯片的 CCLK 信号都有主控设备提供，靠近主控设备的 FPGA 要充当桥梁的作用，将配置数据转发到第二个 FPGA 芯片。可以看到采用从串配置的好处主要在于节省电路板面积，并使得系统具备更大的灵活性。

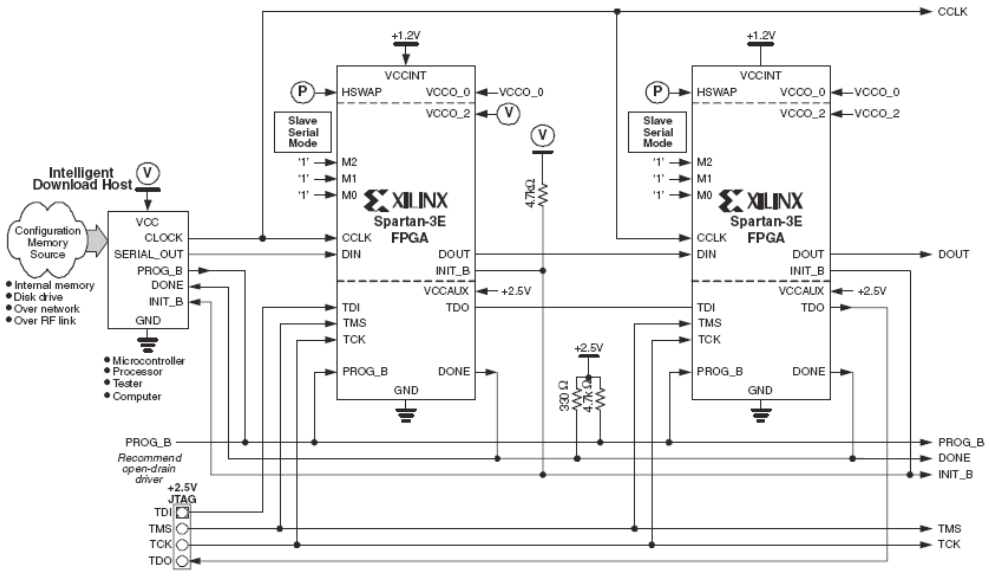


图5–33　多片FPGA从串模式配置电路

5.5.5 JTAG配置模式

1. JTAG配置电路

赛灵思公司的 FPGA 芯片具有 IEEE 1149.1/1532 协议所规定的 JTAG 接口，只要 FPGA 上电，不论模式选

择管脚 M[2:0] 的电平，都可用采用该配置模式。但是将模式配置管脚设置为 JTAG 模式，即 M[2:0]=3’b101 时，FPGA 芯片上电后或者 PROG_B 管脚有低脉冲出现后，只能通过 JTAG 模式配置。JTAG 模式不需要额外的掉电非易失存储器，因此通过其配置的比特文件在 FPGA 断电后即丢失，每次上电后都需要重新配置。由于 JTAG 模式已更改，配置效率高，是项目研发阶段必不可少的配置模式。典型的 Spartan 3E 系列芯片的 JTAG 配置电路如图 5–34 所示。

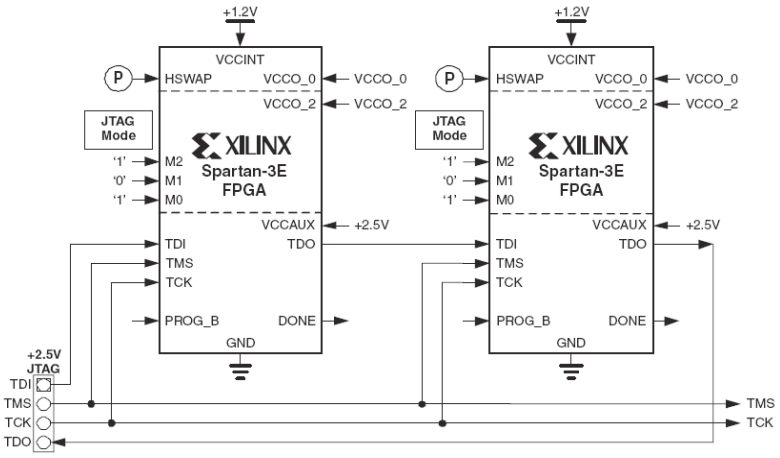


图5–34 JTAG模式配置电路示意图

5.5.6 System ACE配置方案

随着 FPGA 成为系统级解决方案的核心，大型、复杂设备常需要多片大规模的 FPGA。如果使用 PROM 进行配置，需要很大的 PCB 面积和高昂的成本，因此很多情况下都利用微处理由从模式配置 FPGA 芯片，但该配置方案容易出现总线竞争且延长了系统启动时间。为了解决大规模 FPGA 的配置问题，赛灵思公司推出了系统级的 System ACE(Advanced Configuration Environment) 解决方案。

System ACE 可在一个系统内，甚至在多个板上，对赛灵思的所有 FPGA 进行配置，使用 Flash 存储卡或微硬盘保存配置数据，通过 System ACE 控制器把数据配置到 FPGA 中。目前，System ACE 有 System ACE CF(Compact Flash)、System ACE SC(Soft Controller) 以及 System ACE MPM(Muti–Package Module) 三种。读者需要注意的是：System ACE SC/MPM 是和 System ACE CF 独立的解决方案。典型的 ACE 接口以及系统组成如图 5–35 所示。

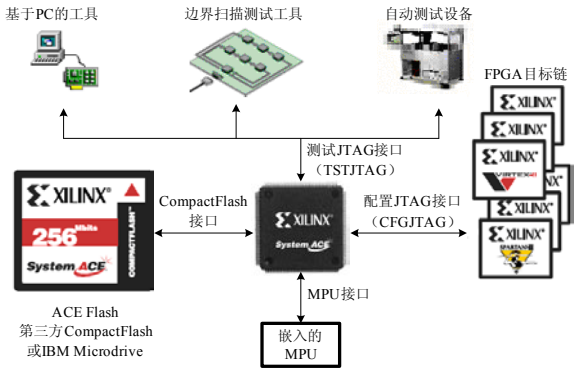


图5–35 典型的ACE接口以及系统组成示意图

1. System ACE CF解决方案

System ACE CF 的核心是 System ACE CF 存储设备和 System ACE 控制器芯片。System ACE CF 存储设备包括赛灵思的 ACE Flash 卡或其它厂家的 Compact Flash 卡以及 IBM 的微硬盘。Compact Flash 卡的容量为 32MB~4GB，微硬盘的容量为 2GB~6GB，至少可配置数百片 FPGA 芯片。

System ACE CF 控制器提供了存储单元和 FPGA 器件之间的接口，PC 和存储器的标准 JTAG 接口。控制器芯片默认的配置模式也是通过边界扫描的方式将数据配置到 FPGA 链中，同样可由边界扫描链的测试和编程接口来辅助进行系统原形的调试，其主要特点有：

– 支持赛灵思所有 FPGA 芯片的配置；

– 以最小的 PC 板空间实现多达 8Gb 的配置；

– 包括高达 152Mbps 的配置速率；

– 利用带有嵌入式处理器核的 FPGA 进行系统调节；

– 管理多个比特流 (全部或部分)，并按需要对其进行激活；

– 包含处理器核初始化；

– 软件存储加密；

– 可移动存储器件；

– 降低了定制配置系统的成本，支持大多数 CompactFlash 卡，包括 Microdrive 单元；包含内置式微处理器接口，可以直接调整 FPGA 配置；释放设计资源。

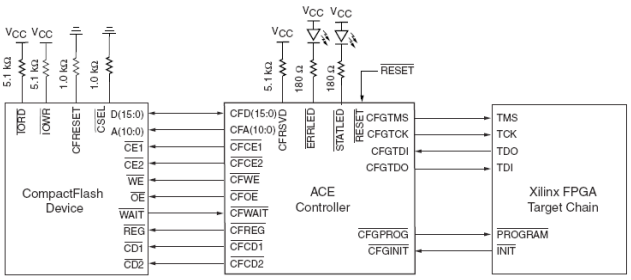


图5–36 System ACE CF配置电路示意图

Compact Flash 接口是 ACE 控制器的关键接口，可连接 Compact Flash 卡、标准的 Compact Flash 模块以及 IBM 微硬盘。Compact Flash 可以进行拆卸，因此对存储内容进行修改和升级以及更换容量都非常方便。Compact Flash 接口由 Compact Flash 控制器和 Compact Flash 仲裁器两部分组成。由 System ACE CF 配置 FPGA 的接口电路如图 5-36 所示。

2. System ACE SC解决方案

System ACE SC 为用户提供了自主性，用户可以自由地选择每一部分的元件，可将其置于电路板的任何位置，且所有的功能在一个独立的 FPGA 中完成，并不需要整合其他组件。System ACE SC 有 4 个主要接口：边界扫描 JTAG 接口、系统控制接口、Flash 存储器接口以及 FPGA 接口，如图 5-37 所示。

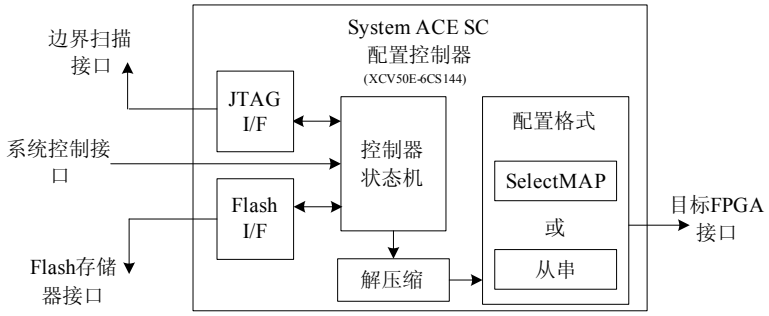


图5-37 System ACE SC接口示意图

其中 JTAG 接口主要提供边界扫描测试和对具有 JTAG 接口的 Flash 存储器通信；Flash 接口主要和外边的 Flash 芯片通信，读取存储器内的内容以及对存储器进行编程；系统控制接口主要提供输入时钟、配置控制信号和配置状态信号等；FPGA 接口主要用于配置 FPGA，可通过从串、从并以及 Selec tMAP 等配置模式。

System ACE SC 和 System ACE CF 的主要区别在于，System ACE SC 的控制器是一个软核逻辑，而不是芯片，需要和设计一起下载到 FPFA 中。其余区别如表 5-4 所列。

特性	System ACE CF	System ACE SC
存储器容量	高达8GB	16/32/64M比特
能否压缩	否	能
FPGA配置模式	JTAG	SelectMAP
最大配置速度	30Mbps	152Mbps
最大设计数	无限制	8
存储媒介	Compact Flash	AMD Flash
器件数目	2	3

表5-4 System ACE CF和System ACE SC的区别

典型的 System ACE SC 配置电路如图 5-38 所示。

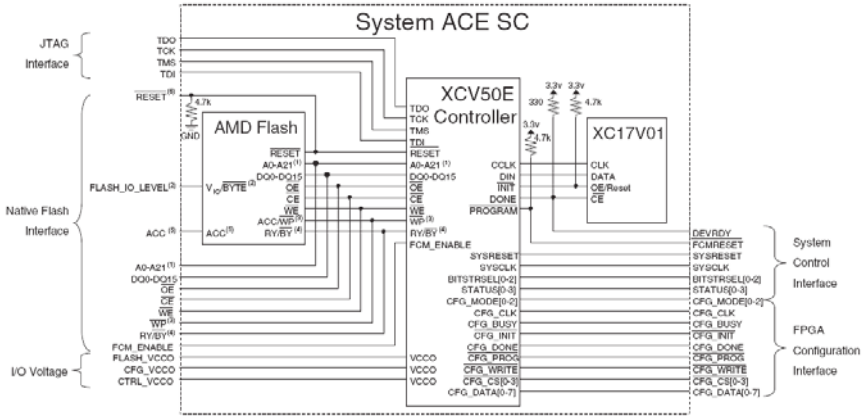


图5-38 System ACE SC配置电路示意图

3. System ACE MPM解决方案

System ACE MPM 是一个整合的组件解决方案，包括 FPGA 和 PROM 组成的配置控制组件和一个 Flash 存储组件，并封装为一个模块，通过尽可能少的组件来实现配置电路。赛灵思公司有 16M、32M 以及 64M 位低密度的 System ACE MPM。System ACE MPM 有 4 个主要接口，和 System ACE SC 的接口一样，其特征和功能也与 System ACE SC 一样。二者的区别在于：System ACE MPM 封装了整个配置模块，而 System ACE SC 允许用户自行配置，其接口电路如图 5-39 所示。

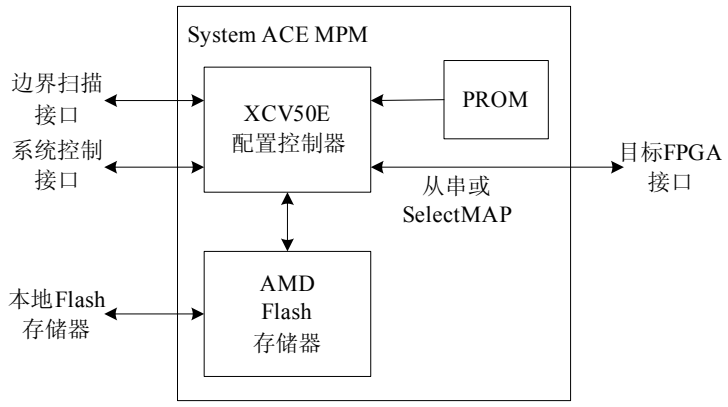


图5-39 System ACE MPM接口电路示意图

System ACE MPM 是赛灵思公司第一个支持位流压缩的配置方案，支持多种配置模式，同时可多达 8 个 FPGA 链的从串配置模式和多达 4 个 FPGA 的 Select MAP 配置模式，最大配置速率为 152Mbps，同时又可最大限度地减小电路板空间和连线。典型的 System ACE MPM 配置电路如图 5-40 所示。总之，System ACE 技术简化了大型 FPGA 系统的配置方案，令开发人员将精力主要集中在系统性能的提高和开发时间的缩短。

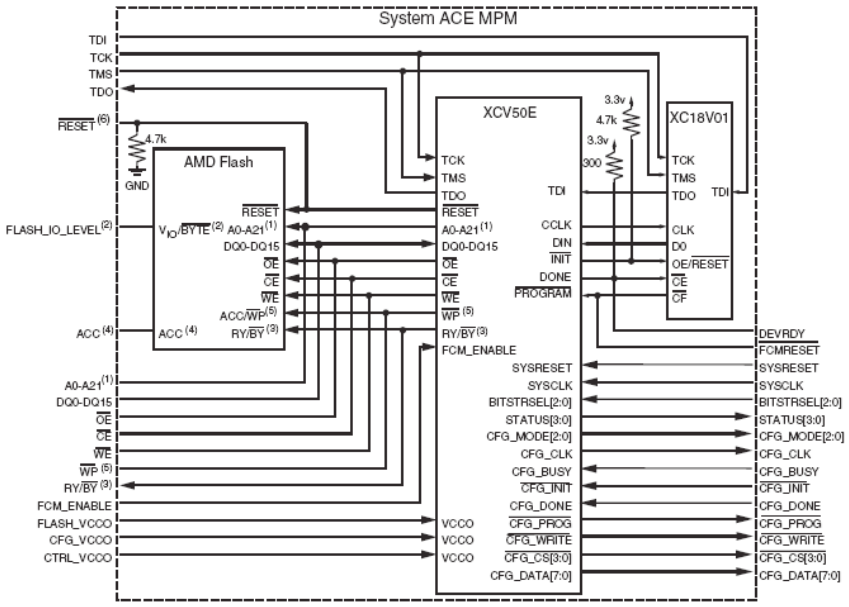


图5–40 System ACE MPM配置电路示意图

5.6 大规模设计的调试经验

在大规模设计的调试应该按照和设计理念相反的顺序，从底层测试，主要依靠 ChipScope Pro 工具。下面主要介绍 ChipScope Pro、FPGA Editor 组件的使用方法。

5.6.1 ChipScope Pro组件应用实例

在赛灵思软件设计工具中，ISE 可集成赛灵思公司的所有工具和程序。ChipScope Pro 也不例外，在 ISE 中将其作为一类源文件，和 HDL 源文件、IP Core 以及嵌入式系统的地位是等同的。本节在 Xilinx Spartan3E–D 开发板上实现一个计数器模块，基于该模块详细介绍如何在 ISE 中新建 ChipScope 应用以及观察、分析数据的详细操作。

例 5.6.1：在 ISE 中实现一个 8 比特计数器，利用 ChipScope 分析其逻辑输出。

(1) 新建用户工程，添加 mycounter.v 的源文件，其内容如下所列：

```
module mycounter(clk, reset, dout);
    input clk;
    input reset;
    output [7:0] dout;
            reg [7:0] dout;
            always @(posedge clk) begin
  if (reset == 0)
          dout <= 0;
```

```
        else
            dout <= dout + 1;
    end
endmodule
```

然后根据电路连接，添加相应的管脚约束。

(2) 综合工程，然后在 ISE 工程管理区，单击右键，选择“Add New Source”命令，在弹出的对话框中选择“ChipScope Definition and Connection File”类型，并在“File Name”栏输入 ChipScope 设计名称 mychipscope，如图 5-41 所示。

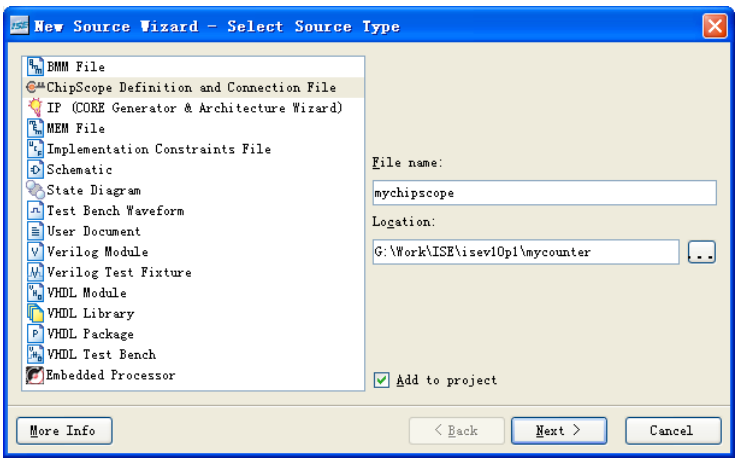


图5-41　添加ChipScope设计示意图

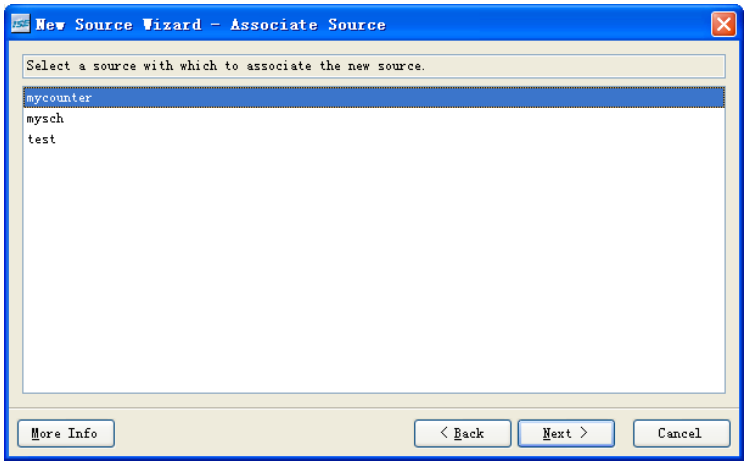
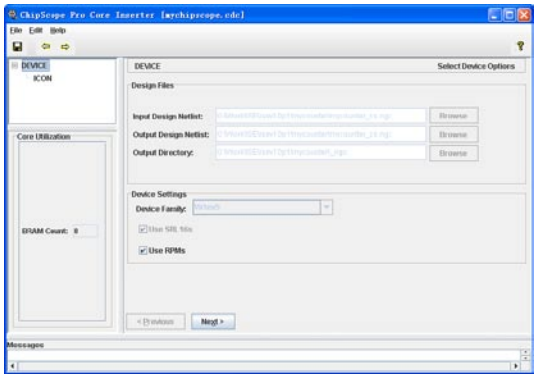


图5-42　测试模块选择界面

单击“Next”按键，进入分析文件选择界面，这里会将该文件夹里所有的 HDL 设计、原理图设计都罗列出来 (包括顶层模块和全部底层模块)，供用户挑选，用鼠标单击即可选中，本例选择 mycounter，如图 5-42 所示。单击“Next”按键进入小结页面，单击“Finish”按键完成添加。

(3) 双击工程区 mycounter.v 下的子模块 mychipscope.cdc，可自动打开 Chipscope Pro Core Insterser 软件，添加触发单元和触发位宽。其中触发类型选为 Basic，位宽为 8 比特；设置采样深度为 4096，各步骤如图 5-43

到图 5-46 所示。



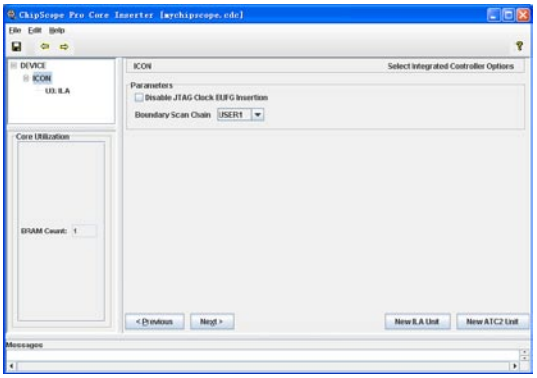
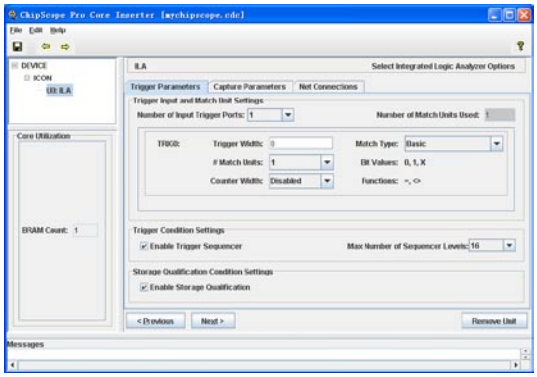


图5-43 调试工程配置界面　图5-44 ICON核配置界面



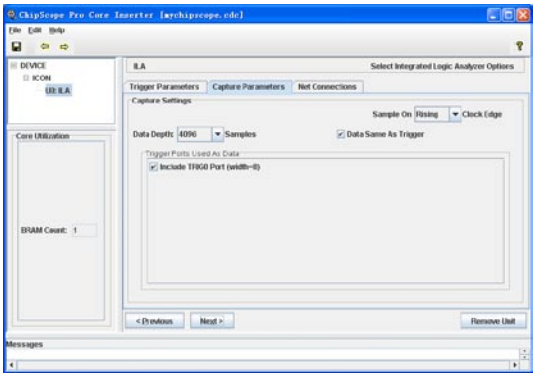


图5-45 触发信号配置界面　图5-46 采集深度配置界面

(4) 点击“Next”进入网表连接显示页面，如图 5-47 所示。其中如果用户定义的触发和时钟信号线有未连接的情况，则图中“UNIT”、“CLOCKPORT”以及“TRIGGERPORTS”等字样以红色显示；正确完成连接后则变成黑色。

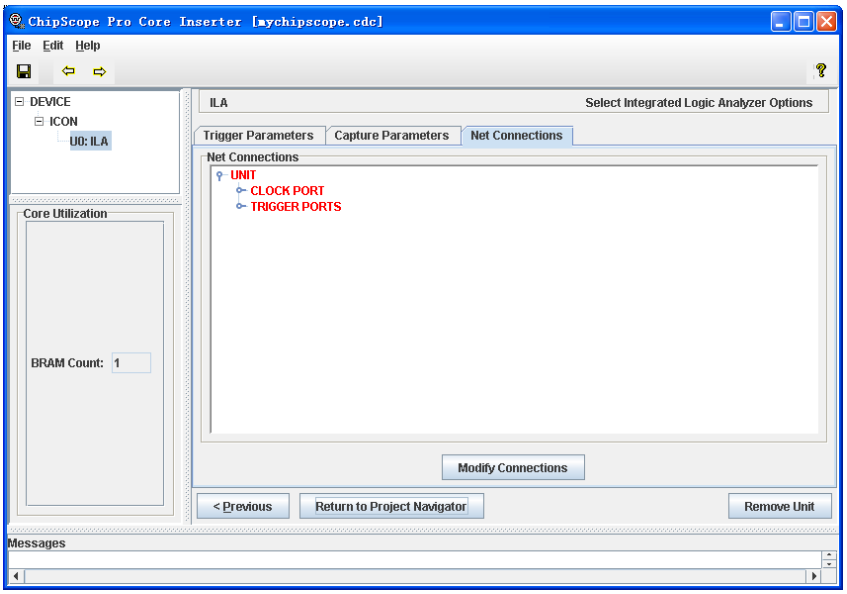


图5-46 网表连接提示界面

点击图 5-47 中“Modify Connection”的按键，进入连接页面，时钟和数据的连接如图 5-48、图 5-49 所示。需要注意的是，ChipScope Pro 只能分析 FPGA 设计的内部信号，因此不能直接连接输入信号的网表，所以输入信号网表全部以灰色显示。如果要采样输入信号，可通过连接其输入缓冲信号来实现，时钟信号选择相应的 BUFGP，普通信号选择相应的 IBUF。如图 5-48 中所示，选择采样时钟时，选择了 CLK_BUFGP。

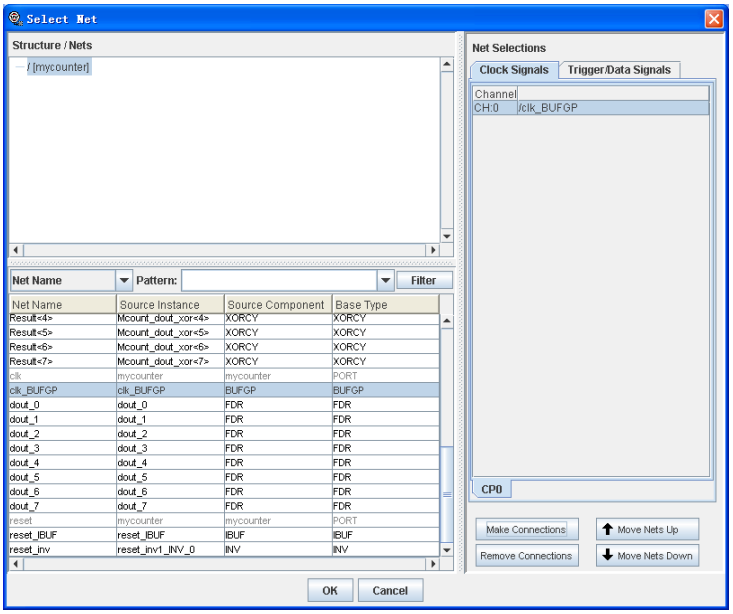


图5-48 时钟网表连接界面

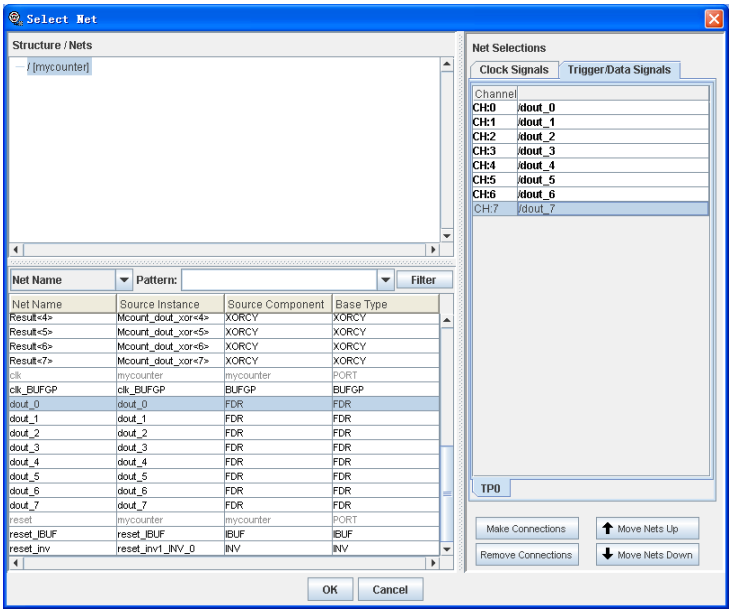


图5-49 触发网表连接界面

连接完成后，单击“OK”按键返回连接显示界面，发现所有提示字符“UNIT”、“CLOCKPORT”以及“TRIGGERPORTS”没有红色，则单击“Return Project Navigator”，退出 Chipscope，返回到 ISE 中。否则需要再次点击“Modify Connection”按键重新连接。

(5) 在工程中加入 UCF 文件，约束时钟、数据管脚位置。为了简化也可以只添加 clk 和 reset 这两个控制信号的管脚约束，其内容如下：

NET "clk" LOC = "C9" | IOSTANDARD = LVCMOS33 ;

Define clock period for 50 MHz oscillator (40%/60% duty-cycle)

NET " clk " PERIOD = 20.0ns HIGH 40%;

NET "reset" LOC = "H13" | IOSTANDARD = LVTTL | PULLDOWN ;

(6) 在 ISE 过程控制区中双击“Implement Design”和“Generate Programming File”，可以完成实现以及生成可编程文件，并将设计人员插入的各类核也将被包含在比特文件中。生成配置文件后，双击图 5-50 所示的“Analyze Design Using Chipscope”图标，可自动打开 Chipscope Pro Analyzer 软件。

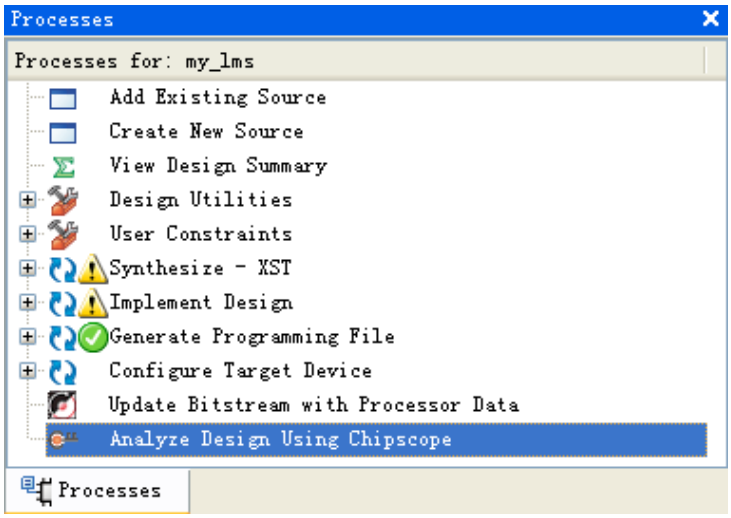


图5-50 Chipscope Pro Analyzer启动操作示意图

(7) 在 Chipscope Analyzer 用户界面上点击工具栏上图标“”，初始化边界扫描链。等扫描完成后，单击“Device”菜单下“DEV: 0 My Device0(XC3S500E) → Configure”命令选择 .bit 文件配置 FPGA。

(8) 芯片配置完成后，选择“File”菜单的“Import”命令，可弹出 CDC 文件加载页面，选择相应的 CDC 文件，将会把所有以“Dataport<n>”的名称修改为综合后的线网名称。

(9) 组合 cnt 总线信号。可按住“Ctrl”键，选择多个总线信号，单击右键，选择“Add to Bus”命令，将其组合成相应的总线信号，如图 5-51 所示。

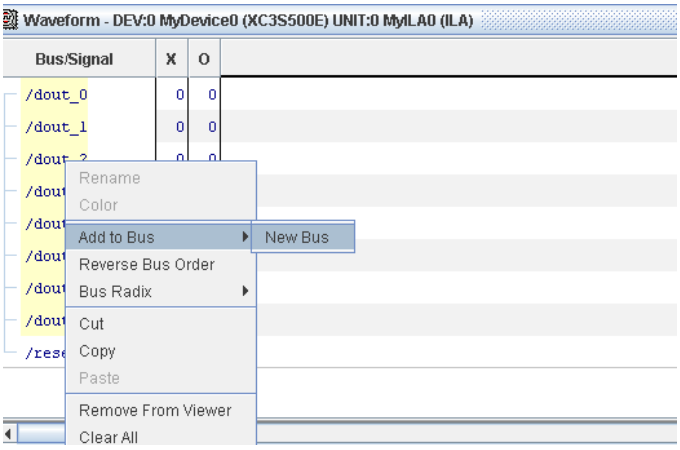


图5-51 添加总线操作示意图

(10) 不设定触发条件采集数据。点击工具栏的“▶”图标，开始采集数据。整体结果如图 5-52 所示，单击工具栏的“🔍”按键，可放大信号，局部结果如图 5-53 所示。从分析结果可以看出，本设计在 FPGA 中成功地完成了 8 比特计数器的功能。

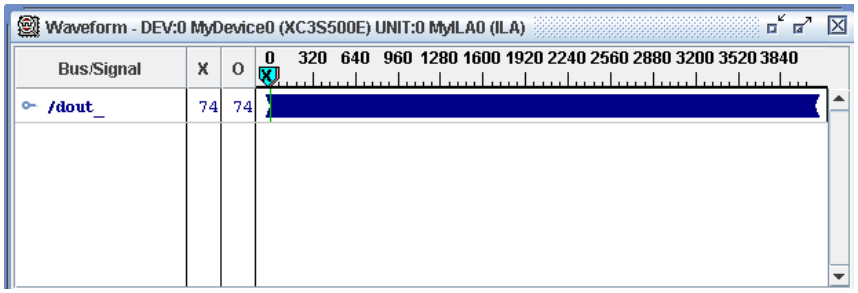


图5-52 Analyzer分析结果整体示意图

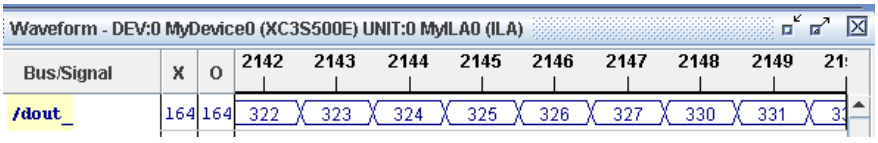


图5-53 Analyzer分析结果局部示意图

(11) 设定触发条件采集数据。在“Trigger Setup”栏 Match 区域的“M0: Trigger Port0”行的 Value 列输入触发条件“0000_0000”，如图 5-54 所示。

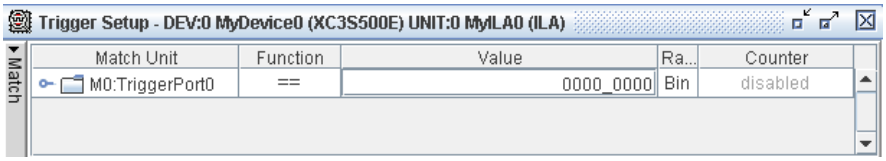


图5-54 触发条件设置界面

点击工具栏的“▶”图标，开始采集数据，可以看到，采集结果的第一个数为 0，如图 5-55 所示。当然，用户可以根据需要设置更复杂的触发条件。

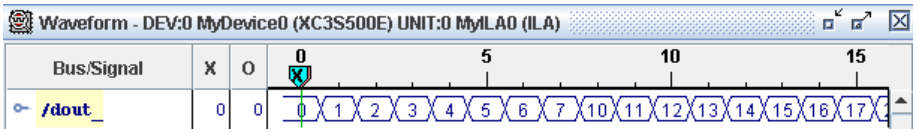


图5-55 触发条件设置界面

(12) 利用 Bus Plot 功能绘制输出信号波形。在工程区双击“Bus Plot”命令，然后在弹出窗口的“Bus Selection”区域选中“dout”，则会将采集数据以图形方式显示出来，如图 5-56 所示。由于本设计是 8 比特加 1 计数器，因此其波形就是幅度为 0 到 255 的锯齿波。

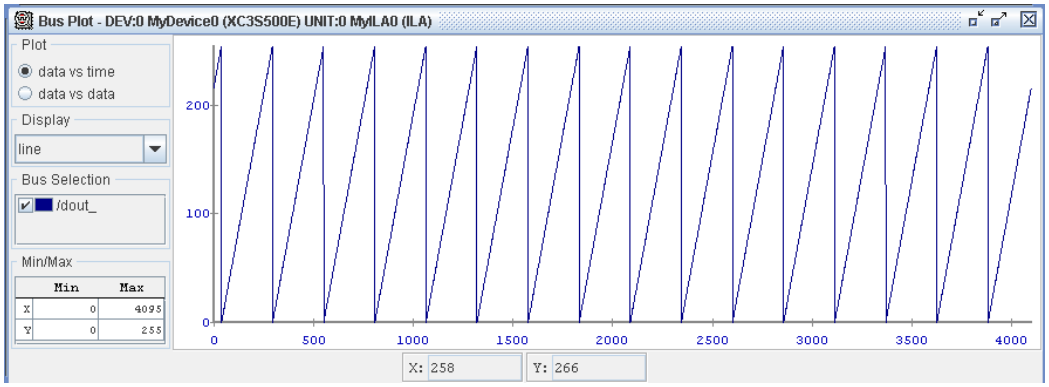


图5-56 8计数器的波形示意图

5.7 FPGA 设计的 IP 和算法应用

基于 IP 的设计已成为目前 FPGA 设计的主流方法之一，本章首先给出 IP 的定义，然后以 FFT IP 核为例，介绍赛灵思 IP 核的应用。

5.7.1 IP核综述

IP(Intelligent Property) 核是具有知识产权核的集成电路芯核总称，是经过反复验证过的、具有特定功能的宏模块，与芯片制造工艺无关，可以移植到不同的半导体工艺中。到了 SOC 阶段，IP 核设计已成为 ASIC 电路设计公司和 FPGA 提供商的重要任务，也是其实力体现。对于 FPGA 开发软件，其提供的 IP 核越丰富，用户的设计就越方便，其市场占用率就越高。目前，IP 核已经变成系统设计的基本单元，并作为独立设计成果被交换、转让和销售。

从IP核的提供方式上，通常将其分为软核、硬核和固核这3类。从完成IP核所花费的成本来讲，硬核代价最大；从使用灵活性来讲，软核的可复用使用性最高。(这部分内容前面已经阐述，这里再重申一下)

软核 (Soft IP Core)

软核在 EDA 设计领域指的是综合之前的寄存器传输级 (RTL) 模型；具体在 FPGA 设计中指的是对电路的硬件语言描述，包括逻辑描述、网表和帮助文档等。软核只经过功能仿真，需要经过综合以及布局布线才能使用。

其优点是灵活性高、可移植性强，允许用户自配置；缺点是对模块的预测性较低，在后续设计中存在发生错误的可能性，有一定的设计风险。软核是 IP 核应用最广泛的形式。

固核 (Firm IP Core)

固核在 EDA 设计领域指的是带有平面规划信息的网表;具体在 FPGA 设计中可以看做带有布局规划的软核，通常以 RTL 代码和对应具体工艺网表的混合形式提供。将 RTL 描述结合具体标准单元库进行综合优化设计，形成门级网表，再通过布局布线工具即可使用。和软核相比，固核的设计灵活性稍差，但在可靠性上有较大提高。目前，固核也是 IP 核的主流形式之一。

硬核 (Hard IP Core)

硬核在 EDA 设计领域指经过验证的设计版图；具体在 FPGA 设计中指布局和工艺固定、经过前端和后端验证的设计，设计人员不能对其修改。不能修改的原因有两个：首先是系统设计对各个模块的时序要求很严格，不允许打乱已有的物理版图；其次是保护知识产权的要求，不允许设计人员对其有任何改动。IP 硬核的不许修改特点使其复用有一定的困难，因此只能用于某些特定应用，使用范围较窄。

IP Core 生成器 (Core Generator) 是 Xilinx FPGA 设计中的一个重要设计工具，提供了大量成熟的、高效的 IP Core 为用户所用，涵盖了汽车工业、基本单元、通信和网络、数字信号处理、FPGA 特点和设计、数学函数、记忆和存储单元、标准总线接口等 8 大类，从简单的基本设计模块到复杂的处理器一应俱全。配合赛灵思网站的 IP 中心使用，能够大幅度减轻设计人员的工作量，提高设计可靠性。

Core Generator 最重要的配置文件的后缀是 .xco，既可以是输出文件又可以是输入文件，包含了当前工程的属性和 IP Core 的参数信息。

5.7.2 FFT IP核应用示例

ISE提供了FFT/IFFT的IP Core,可以完成实数、复数信号的FFT以及IFFT运算。FFT的IP Core提供三种结构，分别为：

(1) 流水线，Streaming I/O 结构：允许连续的数据处理；

(2) 基 4，Burst I/O 结构:提供数据导入／导出阶段和处理阶段。此结构拥有较小的结构，但转换时间较长；

(3) 基 2，Burst I/O 结构:使用最少的逻辑资源，同 Radix-4 相同，提供两阶段的过程。其配置界面有 3 页，第一页如图 5-57 所示，主要用于配置实现结构；第二页配置数据位宽以及数据处理操作；第三页配置数据缓存空间。

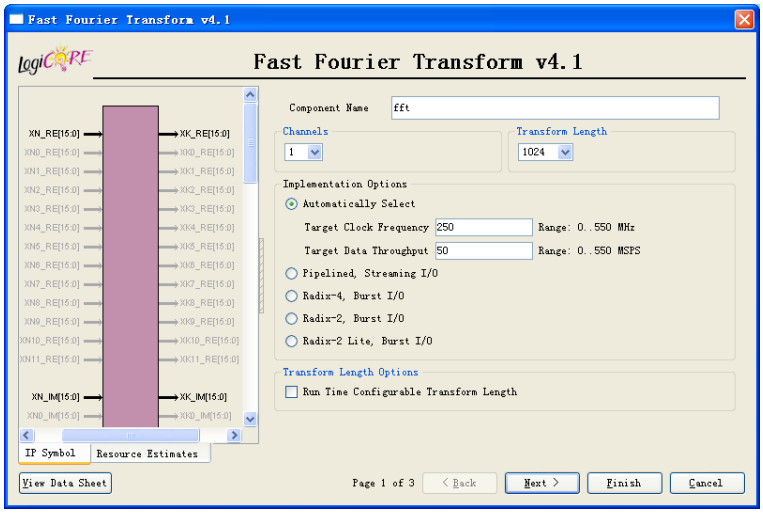


图5-57 FFT IP core的用户界面

在实际硬件操作中，模块的执行速度是很重要的参数，所以本文分析第一种结构，即流水线 Streaming I/O 结构，以进行连续的数据处理。在进行当前帧的 N 点数据时，可加载下一帧的 N 点数据，同时输出前一帧的 N 点数据。此结构由多个基 2 的蝶形处理单元构成，每个单元都有自己的存储单元来存储输入和中间处理的数据，其结构如图 5-58 所示。

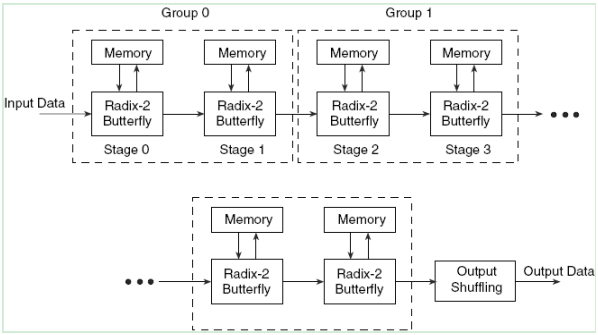


图5-58 FFT模块的流水线，Streaming I/O结构

FFT 的计算单元具有丰富的控制信号，其详细说明见下文。

XN_RE、XN_IM：输入操作数，分别为实部和虚部，以 2 的补码输入。在使用时应当确定其位宽。

START：FFT 开始信号，高有效。当此信号变高时，开始输入数据，随后直接进行 FFT 转换操作和数据输出。一个 START 脉冲，允许对一帧进行 FFT 转换。如果每 N 个时钟有一个 START 脉冲或者 START 始终为高，，则都可以连续进行 FFT。如果在最初的 START 前，还没有 NFFT_WE，FWD_INV_WE，SCALE_SCH_WE 信号，则 START 变高后就使用这些信号的默认值。由于此 IP Core 支持非连续的数据流，因此在任何时间输入 START，即可开始数据的加载。当加载 N 个数据结束后，就开始 FFT 转换运算。

UNLOAD：对于 Burst I/O 结构，此信号将开始输出处理的结果。对于流水线结构和比特逆序输出的情况，此端口不是必要的。

NFFT：此端口只对实时可配置应用时有用。

NFFT_WE：此端口是 NFFT 端口的使能信号。

FWD_INV：用以指示 IP Core 为 FFT 还是 IFFT，其等于 1 时 IP Core 进行 FFT 运算，否则进行 IFFT 运算。至于采用哪种转换运算是可以逐帧变化的。这一端口给 FFT 的使用提供了很大的方便。

FWD_INV_WE：作为 FWD_INV 端口的使能信号。

SCALE_SCH:(1) 在 IP Core 设计时，如果选择在计算过程中进行中间数据的缩减，那么此信号才可起作用；(2) 输入的位宽等于 2*ceil(NFFT/2)，其中 NFFT = log2(point size)。(3) 流水线结构中，将每个基 2 的蝶形处理单元视为一个阶段，每个阶段进行一次数据的缩减，缩减的比例以此输入中对应阶段的两比特表示。(4) 每阶段的两比特数可以是 3,2,1 或 0：它们表示了数据所需要移动的比特数。

SCALE_SCH_WE：作为 SCALE_SCH 的使能信号。

SCLR：可选端口。

Reset：重置信号端口。Reset=1 时，所有工作都停止且初始化。但内部的帧缓存保留其内容。

CE：可选端口。

CLK：输入时钟。

XK_RE，XK_IM：输出数据总线，以 2 的补码输出。SCALE_SCH_WE 有效时，输出位宽等于输入；否则，输出位宽 = 输入位宽 +NFFT+1。

XN_INDEX：位宽等于 log2(point size)，输入数据的下标。

XK_INDEX：位宽等于 log2(point size)，输出数据的下标。

RFD：数据有效信号，高有效，在加载数据时为高电平。

BUSY：IP Core 工作状态的指示信号，在计算 FFT 转换时为高电平。

DV：数据有效指示信号，当输出端口存在有效数据时变高。

EDONE：高有效。在 DONE 信号变高的前一个时钟变为高电平。

DONE：高有效。在 FFT 完成后变高，且只存在一个时钟。在 DONE 变高后，IP Core 开始输出计算结果。

BLK_EXP：当使用 Burst I/O 结构时可用，若选择流水线，则此端口无效

OVFLO：算法溢出指示。在数据输出时，如每帧有溢出，此信号变高。在每帧开始处，此信号重置。

例 5.7.1 使用 IP Core 实例化一个 16 点、位宽为 16 位的 FFT 模块。

IP Core 直接生成的乘法器的 Verilog 模块接口为：

```
module fft16(sclr, fwd_inv_we, rfd, start, fwd_inv, dv, scale_sch_we, done, clk, busy, edone, scale_sch,
xn_re, xk_im, xn_index, xk_re, xn_im, xk_index);
  input sclr , fwd_inv_we, start, fwd_inv, scale_sch_we, clk;
  input [3 : 0] scale_sch;
input [15 : 0] xn_re;
  output rfd, dv, done, busy, edone;
  output [15 : 0] xk_im;
```

```
  output [3 : 0] xn_index;
  output [15 : 0] xk_re;
  input [15 : 0] xn_im;
  output [3 : 0] xk_index;
……
endmodule
```

在使用时，直接调用 multiply 模块即可，如

```
module fft16(sclr, fwd_inv_we, rfd, start, fwd_inv, dv, scale_sch_we, done, clk, busy,
edone, scale_sch, xn_re, xk_im, xn_index, xk_re, xn_im, xk_index);
  input sclr , fwd_inv_we, start, fwd_inv, scale_sch_we, clk;
  input [3 : 0] scale_sch;
input [15 : 0] xn_re;
  output rfd, dv, done, busy, edone;
  output [15 : 0] xk_im;
  output [3 : 0] xn_index;
  output [15 : 0] xk_re;
  input [15 : 0] xn_im;
  output [3 : 0] xk_index;
fft fft1( // 调用 FFT 的 IPCore
    .sclr(sclr), .fwd_inv_we(fwd_inv_we),          .rfd(rfd), .start(start), .fwd_inv(fwd_inv),
    .dv(dv), .scale_sch_we(scale_sch_we), .done(done), .clk(clk), .busy(busy),
    .edone(edone), .scale_sch(scale_sch), .xn_re(xn_re), .xk_im(xk_im),
    .xn_index(xn_index), .xk_re(xk_re), .xn_im(xn_im), .xk_index(xk_index));

endmodule
```

经过仿真测试得到的功能波形图如图 5-59 所示：

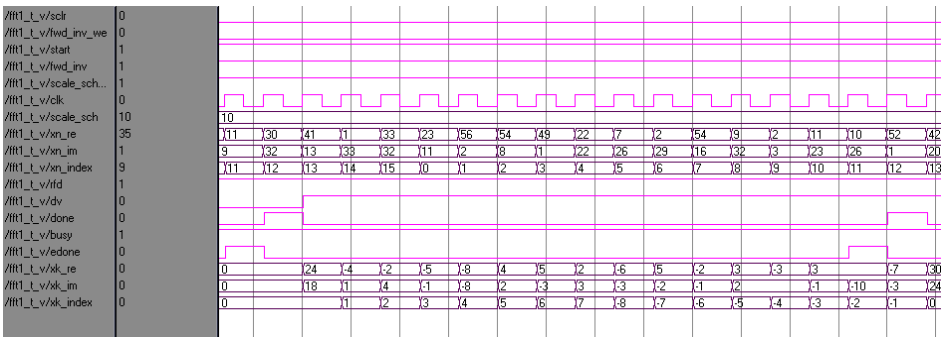


图5-59 FFT的IP core仿真波形

5.8 赛灵思 FPGA 的专用 HDL 开发技巧

专用代码风格是指从 FPGA 器件特征角度考虑，尽可能利用芯片结构以及内嵌的底层宏单元，以取得最佳的综合和实现效果。对于同一个设计，使用适合于 FPGA 体系结构特点的优化设计方法，可以大大提高芯片利用率和设计实现速度。

5.8.1 赛灵思 FPGA的体系结构特点

赛灵思 FPGA 芯片的三种可构造单元是：(1) 可编程输入、输出块 IOB，主要为逻辑阵列与外部芯片引脚之间提供一个可编程接口；(2) 可编程逻辑块 CLB，CLB 主要由一个组合逻辑、几个触发器、若干个多选一电路和控制单元组成，若干个 CLB 有规则地组成 FPGA 逻辑单元阵列结构，以完成用户指定的逻辑功能；(3) 各种连线资源，包括可编程的开关矩阵，内部连接点和金属线。它们位于芯片内部的逻辑块之间，经编程后形成连线网络，以连接芯片内的逻辑块及传递逻辑信息。优秀的设计应在芯片架构的基础上，合理使用组合逻辑、触发器以及块 RAM 等内迁硬核单元。

组合逻辑功能通过用户可编程的查找表实现。查找表是由静态存储器构成函数发生器，在此基础上再增加触发器来形成的，它是既可实现组合逻辑功能又可实现时序逻辑功能的基本逻辑单元电路。SRL16 是赛灵思器件中独有的一种移位寄存器查找表，有 4 个输入用来选择输出序列的长度，能够以极少的硬件资源实现数据缓存和组合逻辑。

每个 CLB 中包含两个触发器，CLB 的组合逻辑功能较少，触发器资源十分丰富。每个 CLB 中包含一个高速进位逻辑。专用进位电路速度远远大于采用传统的加速方法所能增加的速度。算术进位逻辑为有关算术运算中许多新的应用问题提供了有效的解决途径。

同时，赛灵思提供了片上 RAM，特别是大量块 RAM，可以配置成双口 RAM 或 ROM，存储量大、速度快，且不占用逻辑资源，在设计实现中有着广泛的应用。

内嵌的宏单元包括硬核乘加器、硬核处理器、数字时钟处理模块以及高速串行接口，处理能力强，处理能力为片上最高，且不存在时序问题。如果合理地利用宏单元，可达到事半功倍的效果。

5.8.2 赛灵思 FPGA 芯片专用代码风格

1)时钟信号的分配策略

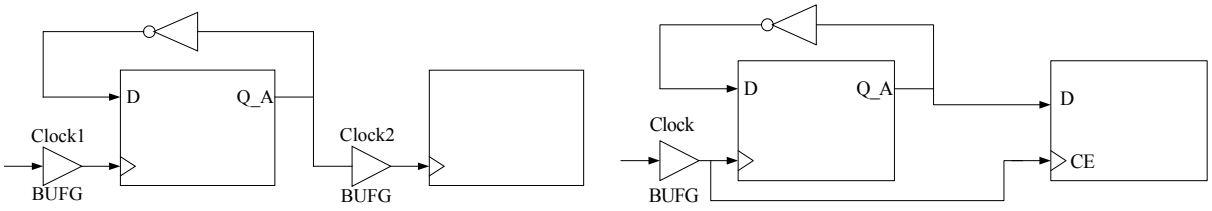
时钟分配网络是 FPGA 芯片中的特殊布线资源，由特定的引脚和特定的驱动器驱动，只能驱动芯片上触发器的时钟输入端或除了时钟输入端外有限的一些负载，其目的是为设计提供小延迟偏差和扭曲可忽略的时钟信号。

首先、使用全局时钟，可为信号提供最短的延时和可忽略的扭曲。全局网线由全局缓冲器 BUFG 来驱动，使用 BUFG 时，时钟信号经 BUFG 驱动后通过长线同时接到每个触发器的时钟端，减少传输延迟。如不使用 BUFG，时钟信号按一般布线连接到不同 CLB，时钟信号到达各触发器的延迟不一致，使同步时序电路出现不同步的现象。

其次、FPGA 特别适合于同步电路设计，尽可能减少使用的时钟信号种类。如 TTL 电路设计中，经常采用的由组合逻辑生成多个时钟分别驱动多个触发器的设计方法对 FPGA 的设计不适用。因为这样做使得时钟种类很多，不能利用专用的时钟驱动器和专用的时钟布线资源，时钟信号只能由通用的布线资源拼凑而成，各个负载点上的时钟延迟偏差很大，会引起数据保持时间问题，降低工作速度。

第三、减小时钟摆率的另一种更有效方法是使用一个时钟信号，生成多个时钟使能信号，分别驱动触发器的时钟使能端，所有触发器的数据装入都由同一个时钟控制，但只有时钟使能信号有效的触发器才会装入数据，时钟使能信号无效的触发器则保持数据。这种方法充分发挥了 FPGA 器件体系结构的优势。

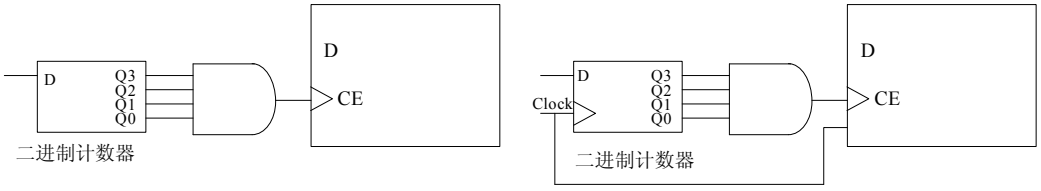
图 5-60 所示电路为分频器的一般设计和优化设计的对比。两种设计方法相比较，图 (a) 电路使用两个全局缓冲，实现两个触发器的异步控制。图 (b) 电路将异步控制转化为同步控制，只占用一个全局缓冲，利用时钟使能信号 CE 控制触发器的动作。因此图 (b) 电路占用更少的全局时钟资源，而且使用一种时钟信号更利用同步控制和减小时钟摆率。



(a) 分频器的普通设计　　(b) 分频器的优化设计

图5-60 分频器两种设计电路的比较

第四、要避免时钟信号毛刺。由图 5-61 所示电路可知，当二进制计数器从 0111 向 1000 变化时，必然会出现一个 1111 的过渡过程，D 触发器的时钟就会产生毛刺，毛刺出现的时间很短，但对于高速处理来讲，足以使触发器误动作。图 (b) 所示电路就是使用同步时钟，并使用时钟使能 CE 端避免时钟信号毛刺的设计电路。



(a) 会产生毛刺的传统设计　　(b) 能消除毛刺的优化电路

图5-61 两种设计电路的比较

2)SRL16 的使用

SRL16 是一种基于查找表 (LUT) 的移位寄存器，可用于构建高密度 DSP 结构 (如滤波器)，能够大幅削减硬件资源。在 Virtex-5 芯片中为 6 输入的查找表，在其余系列芯片中都为 4 输入查找表。其位宽 (B) 和深度 (D) 可以任意配置，最大的特点就是占用 Slice 资源特别少。其占用 Slice 资源 M 的计算公式为 ：

$$M = B(R[D/16]+1)$$

其中，R[] 函数的意义是取整。从上式可以看出，深度为 1 的移位寄存器和深度为 16 的移位寄存器所占用的 Slice 资源是一样的。

例 5.8.1：使用 SRL16 生成位宽为 8，深度为 10 的移位寄存器。

```
module lut_ram(clk, d, q);
 input clk;
 input [7 : 0] d;
 output [7 : 0] q;
 srl16_based_ram srl16_based_ram(
      .clk(clk),
      .d(d),
      .q(q)
 );
endmodule
```

其中 srl16_based_ram 例化了赛灵思提供的 RAM_Based_Shiftreg 的 IP Core，使用 SRL16 结构完成了移位寄存器。上述程序经过 Synplify Pro 综合后，得到的 RTL 结构如图 5-62 所示。

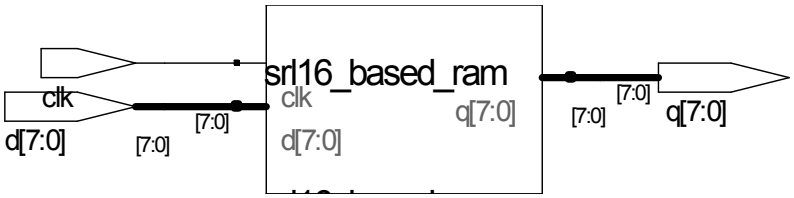


图5-62 SRL16结构移位寄存器的RTL结构示意图

在 ModelSim 6.2b 中完成仿真，其结果如图 5-63 所示

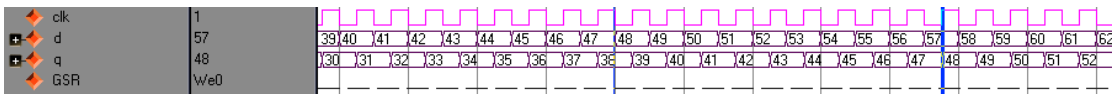


图5-63 SRL16结构移位寄存器的仿真结果示意图

3) 触发器资源的分配技术

由于 FPGA 是一种触发器密集型可编程器件，因此系统的逻辑设计就应该充分利用触发器资源，尽可能降低每个组合逻辑操作的复杂度。

首先，应尽量使用库中的触发器资源。因为 FPGA 触发器资源丰富，而且开发系统在划分逻辑块时，对 D 触发器等元件直接利用 CLB 中的触发器，而对自建触发器则认为是组合电路，需要使用 CLB 中的组合逻辑电路构成，这样既占用更多的 CLB，又浪费了 CLB 的触发器资源。将两种方法进行比较，每使用一个自建 D 触发器比使用库中 D 触发器的电路多占用二至三个 CLB。

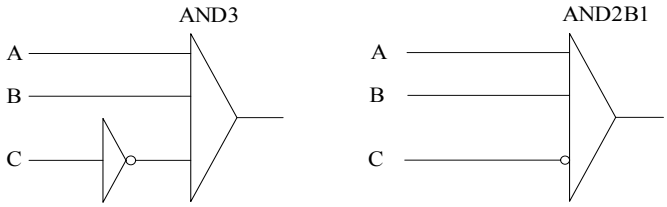
其次，在设计状态机时，应该尽量使用 ONE-HOT 状态编码方案，不用二进制状态编码方案。ONE-HOT

状态编码方案是表示每个状态由 1 位触发器来表示，而二进制状态编码方案是用 LgN/Lg2 位触发器来表示 N 个状态。由于二进制状态编码的稳定度较低，ONE-HOT 状态编码方案对于触发器资源丰富的 FPGA 芯片十分适用。

4) 信号反相的处理策略

在处理反相信号时，设计时应尽可能地遵从分散反相原则。即应使用多个反相器分别反相，每个反相器驱动一个负载，这个原则无论对时钟信号还是对其它信号都是适用的。因为在 FPGA 设计中，反相是被吸收到 CLB 或 IOB 中的，使用多个反相器并不占用更多的资源，而使用一个反相器将信号反相后驱动多个负载却往往会多占资源，而且延迟也增加了。

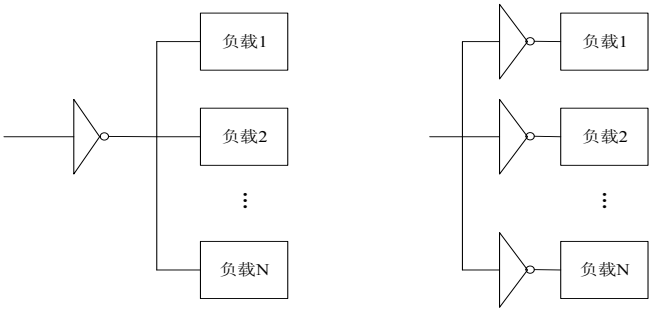
首先，如果输入信号需要反相，则应尽可能地调用输入带反相功能的符号，而不是用分离的反相器对输入信号进行反相。例如，如图 5-64 所示电路是对逻辑 $Y= AB\overline{C}$ 的两种设计电路。两者对比，最优的方案为采用如图 (b) 所示电路，即直接调用 AND2B1，而不要用如图 (a) 所示的电路，用分离的非门对输入信号 C 反相后，再连接到 AND3 的输入。因为在前一种作法中，由于函数发生器用查表方式实现逻辑，C 的反相操作是不占资源的，也没有额外延迟；而后一种作法中，C 的反相操作与 AND3 操作可能会被分割到不同的逻辑单元中实现，从而消耗额外的资源，增加额外的延迟。



(a) 非优化反相设计　　(b) 优化反相设计

图5-64 两种反相设计电路比较

其次，如果一个信号反相后驱动了多个负载，则应将反相功能分散到各个负载中实现，如图 5-65(b) 图所示。而不能采用传统 TTL 电路设计，采用集中反相驱动多个负载来减少所用的器件的数量，如图 5-65(a) 图所示。因为在 FPGA 设计中，集中反相驱动多个负载往往会多占一个逻辑块或半个逻辑块 ,，而且延迟也增加了。分散信号的反相往往可以与其它逻辑在同一单元内完成而不消耗额外的逻辑资源。



(a) 非优化集中反相　　(b) 优化分散反相

图5-65 两种反相驱动电路比较

ISE与EDK开发技巧之时序篇

作者：Ricky Su(www.rickysu.com)

问：一个 FPGA 设计项目需要用哪些评判标准来检验?

一曰功能正确；

二曰时序收敛；

三曰资源消耗少。

时序收敛，即 Timing Closure，意思是使设计的各项时序指标能满足设计前所制定要求。因此，整个过程分为两部分：

1. 制定时序要求；2. 满足时序要求

制定时序要求通常是由整个系统电路的外部环境来决定的，比如：

– 整个电路系统提供给 FPGA 的时钟速度为多快

– FPGA 输入数据是同步信号还是异步信号以及它的频率

– FPGA 输出数据所需的频率

– 输入 / 输出数据与时钟的相位关系

总结以上各种需求情况，得出 FPGA 芯片对外的三种时序约束：

– Period(时钟周期约束)：约束用同一时钟驱动的寄存器 (或同步器件) 所能使用的最低时钟频率来保证 FPGA 内部同步信号的采样时间与保持时间。

– Offset：约束用时钟采样数据 (offset in) 或用时钟打出数据 (offset out) 时时钟与数据的相位差来保证 FPGA 采样数据的建立时间与下一级芯片得到数据的采样时间。

– Pad to Pad：当输入数据进入 FPGA 后没有经过任何同步器件 (即由时钟驱动的器件如寄存器、BRAM 等)，只经过组合逻辑后就输出片外时，Pad to Pad 的 From...To.. 约束用以保证内部的延迟时间。

有了以上三种约束类型，就可以描述外界的任何可能条件，并清楚的对最终设计需要满足的时序要求作出说明，FPGA 实现工具就会依据此要求进行布局布线，并试图满足要求。赛灵思有许多文档对怎样书写时序约束进行了说明。在此要强调的一点是：时序约束首先是对外界环境的一个反映，其次才是对布局布线工具的要求。时序约束向工具说明上游器件所给的信号是怎样的，下游器件又要求怎样的输入，FPGA 实现工具才好依照此标准来综合、布局、布线，时序收敛的设计才可能在真正的电路环境中正常工作。

这里有一个误区需要澄清：多数人认为 Timing 约束是写在 UCF 文件中的，其实 UCF 中的 Timing 约束只有在布局布线过程中才起作用。为了达到最好的时序性能，我们应该从综合开始就使用约束。不管是赛灵思 XST，还是 Synplify 或者其他综合工具都可以添加时序约束。在综合过程就添加时序约束可以使综合器努力综合出合适的网表，这样在布局布线时就更容易满足时序要求了。

设计时序不收敛通常有以下的现象：

– par 报告布线完成，但是有 timing error ；

– par 报告由于不可能达到时序收敛而停止布局布线 ；

– Timing Analyzer 报告显示设计的 timing score 不为 0 ；

– 实际电路板上给定时钟速率 FPGA 工作不正常，降低时钟速率 FPGA 工作正常

如果降低时钟速率能让 FPGA 工作正常，而 Timing 报告又没有显示时序错误，那么有足够的理由怀疑时序约束没有完全约束到所有片内路径，需要仔细研究并完整约束整个设计。

那么设计中的 Timing Error 我们该怎么解决呢?

最简单的，两眼一抹黑，让工具解决 ：把 map, par 等工具的 effor level 提到最高，但通常情况下对结果的提升是不明显的。我们需要有选择地针对不同的情况使用不同的方法。以下来分析几种常见的情况 ：

– Timing 报告显示某一段 net 走线延时特别长 ：

通过在 FPGA Cross Probing 中找到这根 net。如果输入输出距离的确比较长，那么是由于 Place 问题造成的，要解决 Place 问题，需要检查为什么工具会把两个 LUT/FF 放得那么远，是相关的逻辑布局问题，还是因为引脚锁定导致无法移动逻辑的问题。

常用的解决方法可以对前级寄存器做复制寄存器的操作。参考 Xilinx AR9410。

如果是因为输入 / 输出端连接的寄存器被 Pack 到 IOB 中导致寄存器无法移动，那么可以使用 IOB=false 约束将寄存器放在 Slice Logic 中。

– Timing 报告显示逻辑层次比较多，而这些层次中没有延时特别长的 ：

如果是 LUT 到 LUT 的层次太多，那么可以先使用 XST 的 register balancing 功能。如果还是无法满足，可能需要手动调整组合逻辑，在中间插一级寄存器，并修改其他相关的代码，使得相关数据的 latency 一致。其他方法参考 Xilinx AR9417。如果是进位链太长，那么就要考虑使用两个小一点的计数器 / 加法器级联。当考虑到进位逻辑是纵向排列的，当超出一列时，进位会导致延时变长很多时，更需要注意进位链的长度。参考 Xilinx AR9412。

– Hold Violation

Hold Violation 通常都是由 Gated Clock 引起。检查设计中没有使用门控时钟。门控时钟通常会由计数器分频产生。尽量都使用 FPGA 提供的时钟资源，尽量使用 DCM 做 deskew。

– Offset 约束不满足

首先必须保证 offset 写得是正确的。

然后保证输入 / 输出数据一进 FPGA 就用寄存器打一拍，中间不要加组合逻辑。寄存器 Pack 到 IOB 中能最大限度得保证 Offset 约束被满足。(同理，如上所述，不把寄存器放在 IOB 中将有利于 Period 约束。)

如果还是满足不了，可能需要调整一下时钟和数据的相位。可以使用 DCM Phase Shift 调整时钟相位或 IDELAY 调整数据相位。

在制定 Pinout 时可以有意地将一组总线按内部 IOB 的位置排列，低有效位在下方，高有效位在上方，而不是按外部 Pinout 的位置排列。如果以上方法都已经使用并且离目标还差一点点，那么可以试图使用工具的某些

属性，比如：

map –– Timing Driven Packing, Effort Level, Extra Effort, Global Optimization, Allow Logic Optimize Across Hierarchy, Combinational Logic Optimization, Cost Table

par –– Effort Level, Extra Effort

也可以使用 MPPR 或 Xplorer 跑多次实现挑最好的结果。如果所有的尝试都无法满足先前制定的时序目标，那么可能是时候重新考虑一下目标是否合理了。

5.10 新一代开发工具 ISE Design Suit10.1 介绍

赛灵思推出了新一代的整体设计工具套件 ISE Design Suite 10.1 来迎合 FPGA 芯片的改变，并进一步促进 FPGA 器件迅速地从预案性设计平台直接跨入产品生产领域，涵盖逻辑设计、DSP 处理以及嵌入式应用三大方面，使得 FPGA 成为通信、汽车、计算机、医疗、消费电子等要求苛刻且价格敏感领域的首选方案。（编者注：赛灵思即将发布性的开发工具，届时我们将推出更新版）

5.10.1 ISE Design Suit10.1综述

目前，FPGA 设计人员希望设计工具不仅支持先进的生产工艺 (65nm)，还要同时提供更好的工具性能、更高的效率和更丰富的功能，更快实现设计时序收敛和设计反复，快速解决时序以及低功耗等问题。此外，由于工程浩大，必须通过团队合作来完成设计，因此要求设计工具满足团队设计所有要求，通过一个集成常见应用环境的工具来提高团队成产力，并通过片上系统 FPGA 促进真正的系统级解决方案。鉴于此，赛灵思推出了新一代 ISE Design Suite 10.1 版设计套件，从正面解决 FPGA 设计师所面临的严峻挑战，并且第一次提供了一个统一了逻辑、DSP 以及嵌入式应用设计人员需要的解决方案。ISE Design Suite 10.1 为设计的每一步提供了直观的生产力增强工具，覆盖从系统设计探索、软件开发和基于 HDL 硬件语言设计、直到验证、调试和 PCB 设计集成的全部设计流程。

5.10.2 ISE Design Suit 10.1的创新特性

在过去的几年内，ISE 设计工具一直被用户评为业界最佳解决方案，ISE Design Suite 10.1 继承了 ISE 以前版本的全部优点；此外，它还具备以下 8 个创新特点，为大规模、复杂 FPGA 设计提供更高的性能和更高的生产力。

1. 一个套件统一提供全面的客户解决方案

和以往版本相比，ISE Design Suite 10.1 的最大特点就是融合了赛灵思公司发布的所有软件包，为不同用户提供了统一的开发平台，同时支持逻辑、DPS 和嵌入式设计的全面设计环境，且具备完全的互操作能力。同时，协调提供了完整的客户解决方案，无缝集成了系统级设计、IP 核、RTL 设计、功能验证、RTL 综合、布局布线、功率分析、ChipScope 调试、嵌入式系统软硬设计以及完善的第三方 EDA 工具。通过电子化交付流程保证用户快速方便地获得所有产品的最新更新和评估。更为重要的是，ISE Design Suite 10.1 提供了一个可定制的环境，可通过定制来适合设计人员的不同需要；无论用户选择何种定制方案，都通过一个单一序列号来管理，无须像以前版本中的 ISE Foundation、EDK 以及 ChipScope 等采用独立的序列号控制。

2. 编译速度提高两倍

ISE Design Suite 10.1 以平均运行速度提高两倍的特性极大地加快了设计实施速度，使得设计人员可在一天内完成多次反复设计。这对于团队设计是非常重要的，更快的运行速度极大地节约了开发时间，加快了产品的上市速度。

3. SmartXplorer技术提供的设计性能提升高达38%

ISE Design Suite 10.1 的另一个重要意义是及时采用了 SmartXplorer 技术，这一技术专门为解决设计人员所面临的时序收敛和生成力这两大艰巨挑战而开发。SmartXplorer 技术支持在多台 Linux 主机上进行分布式处理，可在一天时间内完成多次实现过程。通过分布式处理和多种实施策略来确定最优策略，性能（最高时钟频率）可以提升多达 38%。同时，SmartXplorer 技术还为用户利用独立时序报告监控每个运行实例提供了相应的工具，通过多组策略支持进入更广泛深入的实现探索。此外，还通过对主流和大规模密度器件进行算法优化和微调，改善大规模 (DSP48、块 RAM) 的布局，利用总线敏感的 IO 布局工具，将总线布置在一起，使性能平均再提高 8%。

4. 集成的PlanAhead Lite提供了终极生产力

集成的 PlanAhead Lite 工具，为用户提供了强大的布局规划和分析功能，并提供动态局部重配

置的功能。首先，简化管理目标 FPGA 和 PCB 之间接口的复杂度，并提高用户设计的性能，当与 ISE 结合使用时，可获得 30% 的性能提升。同时，支持设计分析和布局规划，为用户提供了可视化显示关键路径和布局规划以提高性能。 其次，PlanAhead 技术支持在设计早期阶段智能地实现管脚定义 (PinAhead)，从而避免了在设计后期经常发生的与引脚布局相关的修改。在过去，这种修改通常需要利用交互式引脚布局才能完成设计规模检查。在 PlanAhead 工具中，引脚分配完成后，还可使用逗号分割值 (CSV) 文件或通过 VHDL 或 Verilog 头文件输出 IO 端口信息。 第三，提供了从前端到后端的动态局部重配置的简化方案，动态局部重配置技术是 FPGA 领域的最新技术之一。在 FPGA 运行时，通过 JTAG 或 SelectMAP(ICAP) 重新配置部分区域，而不影响非重配置区域的正常工作。

5. 添加了新的基于策略的设计实施

ISE Design Suite 10.1 推出的基于策略的设计，进一步简化了确定最优实现设置的过程，让尝试优化的工作由工具来完成，不必用户输入复杂的设计。设计人员可规定和设置自己独特的设计目标，可以是性能最大、优化器件利用、降低动态功耗或者是实施时间最短。利用这一资源面积优化策略，逻辑资源利用情况平均可减少 10%。

6. 与第三方EDA厂家广泛合作，可提供更优化的验证能力

ISE Design Suite 10.1 还受益于赛灵思公司同业界领先 EDA 供应商的良好合作，通过广泛联合提供更好的设计验证能力。赛灵思公司和 Mentor Graphics 公司的强强联合，提供了业界第一个 IEEE IP 加密硬 IP 模型，使运行速度缩短多达一倍。新的性能优化 BRAM、DSP 以及 FIFO 仿真模型进一步将 RTL 仿真运行时间缩短了一倍。由于在设计中，有 80% 左右的时间是用来验证设计的，因此上述改进无疑加快了产品面市时间。

7. 简化系统设计

为帮助用户更快实现嵌入式和 DSP 设计，ISE Design Suite 10.1 还对赛灵思嵌入式和 DSP 工具进行了进一步的易用性简化。例如：统一的互操作性保证了用户可以在 ISE Design Suite 10.1 中容易地增添 System Generator 模块。EDK 和 System Generator 技术之间不同工具的集成得到进一步的增强，提供了集成领域专用的设计环境，可在 System Generator 中导入 / 导出 EDK 项目用于硬件协同仿真、方便地将 DSP 设计从 System Generator 设计集成到 ISE 中以及利用 System Generator 自动生成用于 EDK 的 DSP 加速器，为同时涉及嵌入式和信号处理的更复杂 FPGA SoC 设计提供支持。

8. 增强功率分析和优化功能

业界研究表明，满足功率预算是 FPGA 设计人员面临的一项越来越大的挑战，特别是工艺几何尺寸的不断缩小进一步加剧了这一问题。ISE Design Suite 10.1 为用户提供了在设计过程中尽早分析功率要求的功能，同时还可以在设计过程中优化动态功率。第二代 XPower 功率分析工具提供了改善的用户接口，按照模块、结构层次、电源轨和使用的资源来分析功率更为容易，因此进一步增强了功率估算功能。信息可以通过文本和 HTML 报告格式给出。与其它逻辑供应商提供的静态估算网页相比，这是一项巨大进步，同时在提供准确的功耗信息方面是一个飞跃。ISE Design Suite 10.1 提供了便捷全面的功率优化功能，利用集成的“功率优化设计目标”功能，用户可以简单地完成功率优化流程。通过映射和布局布线算法的改进，对于采用 65nm Virtex–5 器件和 Spartan–3 Generation FPGA 的设计动态功率平均可降低 10% 和 12%。

ISE Design Suite 10.1 主要组件

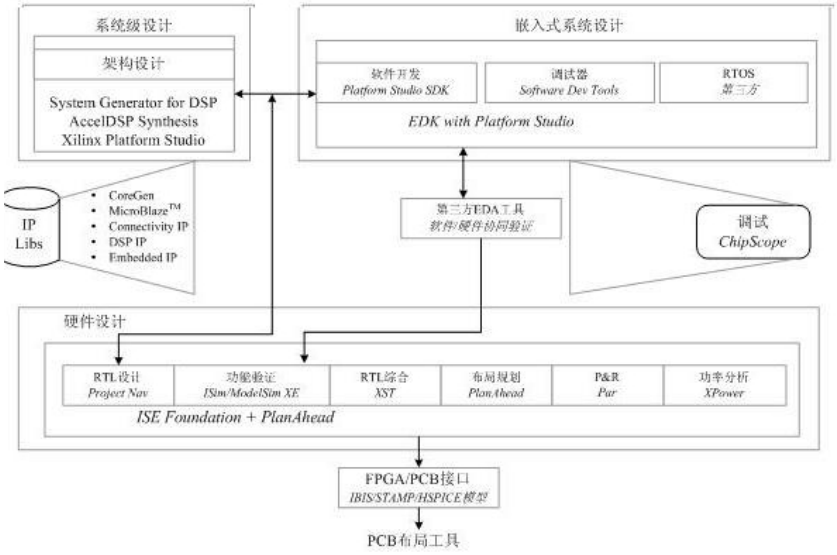


图5–66 ISE Design Suite的完整功能

ISE Design Suite FPGA ISE Design Suite 涉及了 FPGA 设计的各个应用方面，包括逻辑开发、数字信号处理系统以及嵌入式系统开发等 FPGA 开发的主要应用领域，主要包括 ISE Foundation、嵌入式开发套件 (EDK)、System Generator、AccelDSP 综合工具、ChipScope Pro 分析仪、PlanAhead 设计和分析工具等组成部分，其完整的开发功能如图 5–66 所示。

5.10.3.1 ISE Foundation

ISE Foundation 软件是赛灵思公司推出的 FPGA/CPLD 集成开发环境，不仅包括逻辑设计所需的一切，还具有大量简便易用的内置式工具和向导，使得 I/O 分配、功耗分析、时序驱动设计收敛、HDL 仿真等关键步骤变得容易而直观。

ISE 的主要功能包括设计输入、综合、仿真、实现和下载，涵盖了 FPGA 开发的全过程，从功能上讲，其工作流程无需借助任何第三方 EDA 软件。

(1) 设计输入：ISE 提供的设计输入工具包括用于 HDL 代码输入和查看报告的 ISE 文本编辑器 (ISE Text Editor)，用于原理图编辑的工具 ECS(Engineering Capture System)，用于生成 IP Core 的 Core Generator，用于状态机设计的 StateCAD 以及用于约束文件编辑的 Constraint Editor 等。

(2) 综合：ISE 的综合工具不但包含了赛灵思自身提供的综合工具 XST，同时还可以内嵌 Mentor Graphics 公司的 LeonardoSpectrum 和 Synplicity 公司的 Synplify，实现无缝链接。

(3) 仿真：ISE 本身自带了一个具有图形化波形编辑功能的仿真工具 HDL Bencher，同时又提供了使用 Model Tech 公司的 Modelsim 进行仿真的接口。

(4) 实现：此功能包括了翻译、映射、布局布线等，还具备时序分析、管脚指定以及增量设计等高级功能。

(5) 下载：下载功能包括了 BitGen，用于将布局布线后的设计文件转换为位流文件；还包括了 ImPACT，功能是进行设备配置和通信，控制将程序烧写到 FPGA 芯片中去。

使用 ISE 进行 FPGA 设计的各个过程可能涉及到的设计工具如表 5–5 所示。

设计输入	综合	仿真	实现	下载
HDL 文本编辑器 ECS 原理图编辑器 StateCAD 状态机编辑器 Core Generator Constraint Editor	XST FPGA Express （Synplify LeonardoSpectrum）	HDL Bencher （ModelSim）	Translate MAP Place and Route Xpower	BitGen IMPACT

表5-5 ISE设计工具表

5.10.3.2 EDK开发工具

嵌入式系统包括硬件开发和软件开发两部分，因此嵌入式开发工具指可生成硬件平台，并能编辑、编译、链接、加载和调试高级编程语言（通常是 C 或 C++），最终将其运行在处理器引擎上的综合开发工具。EDK 就是赛灵思公司推出的基于 FPGA 的嵌入式开发工具，包括了嵌入式硬件平台开发工具 (Platform Studio)、嵌入式软件开发工具 (Platform Studio SDK)、嵌入式 IBM PowerPC™ 硬件处理器核、Xilinx MicroBlaze 软处理器核、开发所需的技术文档和 IP，为设计嵌入式可编程系统提供了全面的解决方案。

EDK 10.1 版还包括了最新的 IP 内核以优化系统设计。同时还包括了 SPI、DDR2、DMA、PS2 和支持 SGMII 的三模式以太网 MAC 等外设，FlexrayTM 外设选项，以及用于 DMA 的 PCI Express 驱动支持。改进后的多端口存储器控制器以及存储器接口生成器 (MIG) 工具为存储密集应用提供了更为强大和丰富的接口选择。此外，对软核处理器 MicroBlaze 的内核 IP 进一步优化和更新，从而可以提供更大的缓存接口灵活性。

此外，在利用 Virtex-5 FXT 平台进行嵌入式处理系统架构开发和编程的过程中，EDK 10.1 对其进行了进一步的简化。首先，添加了自动设计向导，为设计人员实施高性能 128 位处理器局域总线 (PLB) 提供了一步步的指导，使得配置支持共享式和点到点系统连接非常简便。其次，提供了新的辅助处理器单元控制器 (APU) 工具，对 PowerPC™ 440 处理器模块提供协处理支持。APU 还可以用来建立与高速 FPGA 硬件的直接接口，完成 PowerPC™ 440 处理器代码映射并提供支持软件和硬件优化的分析。

5.10.3.3 DSP工具

目前的 FPGA 芯片不再扮演胶合逻辑的角色，而成为数字信号处理系统的核心器件。在芯片内，不仅包含了逻辑资源，还有多路复用器、存储器、硬核乘加单元以及内嵌的处理器等设备，并且还具备高度并行计算的能力，使得 FPGA 已成为高性能数字信号处理的理想器件，特别适合于完成数字滤波、快速傅立叶变换等。

但遗憾的是，FPGA 并未在数字信号处理领域获得广泛应用，主要原因就是：首先，大部分 DSP 设计者通常对 C 语言或 MATBLAB 工具很熟悉，却不了解硬件描述语言 VHDL 和 Verilog HDL；其次，部分 DSP 工程师认为对 HDL 语言在语句可综合方面的要求限制了其编写算法的思路。基于此，赛灵思公司推出了简化 FPGA 数字处理系统的集成开发工具 DSP tool, 快速简易地将 DSP 系统的抽象算法

转化成可综合的、可靠的硬件系统，为 DSP 设计者扫清了编程的障碍。DSP Tools 主要包括 System

Generator 和 Accel DSP 两部分，前者和 Mathworks 公司的 Simulink 实现无缝链接，后者主要针对 c/.m 语言。

1. System Generator

System Generator 是赛灵思公司的系统级建模工具，在很多方面扩展了 MathWorks 公司的 Simulink 平台，提供了适合硬件设计的数字信号处理 (DSP) 建模环境，加速、简化了 FPGA 的 DSP 系统级硬件设计。System Generator 提供了系统级设计能力，允许在相同的环境内进行软、硬件仿真、执行和验证，并不需要书写 HDL 代码。此外，System Generator 工具还能完成高级提取，自动编译生成 FPGA 代码，也可通过低级的提取，对 FPGA 的底层资源进行访问，从而实现高效率 FPGA 设计构建。目前，基于 System Generator 的设计方法已在复杂系统实现中展现了强大的潜能，必将成为未来流行的 FPGA 开发技术之一。

2. Accel DSP

AccelDSP 是一款第三方综合软件，可将 MATLAB 浮点算法转换成为可综合 RTL 代码。 Xilinx AccelDSP 是目前业界唯一能够将 MATLAB 浮点算法转换成为可综合 RTL 代码的开发工具。该工具可自动地进行浮点 – 定点转换，生成可综合的 VHDL 或 Verilog 代码，并创建用于验证的测试平台，同时还可以生成定点 C++ 模型或由 MATLAB 算法得到 System Generator 块。AccelDSP 综合工具是 Xilinx XtremeDSP 解决方案的重要组成部分。AccelDSP 产品体系由两个主要模块构成：AccelDSP 综合器和 AccelWare IP。 其中，AccelDSP 综合器是一个综合和验证的环境，可以自动将 MATLAB 浮点代码转换成为定点代码，然后生成可综合的 VHDL 或 Verilog 代码，为设计者提供了验证算法和实现算法的功能。

5.10.3.4 ChipScope Pro

逻辑分析仪 (Logic Analyzer) 是 FPGA 调试阶段不可缺少的工具，但是传统逻辑分析仪有两个弊端：首先价格昂贵；其次需要使用大量探头，不仅不合实际且操作麻烦。赛灵思公司为了解决用户的这两个难题，推出了在线逻辑分析仪 (ChipScope Pro)，通过软件方式为用户提供稳定和方便的解决方案。该在线逻辑分析仪不仅具有逻辑分析仪的功能，而且成本低廉、操作简单，因此具有极高的实用价值。ChipScope Pro 既可以独立使用，也可以在 ISE 集成环境中使用，非常灵活，为用户提供方便和稳定的逻辑分析解决方案，支持 Spartan 和 Virtex 全系列 FPGA 芯片。

ChipScope Pro 将逻辑分析器、总线分析器和虚拟 I/O 小型软件核直接插入到用户的设计当中，可以直接查看任何内部信号或节点，包括嵌入式硬或软处理器。信号在操作系统速度下或接近操作系统速度下被采集，并从编程接口中引出，再将采集到的信号通过 ChipScope Pro 逻辑分析器进行分析，从而为设计解放了更多的引脚。利用 FPGA 的可重编程性能，可以在几分钟或几小时内确定设计问题并修改设计；内置的软件逻辑分析器可以用来识别设计问题并进行调试，包括高级触发、过滤和显示选项，无需重新综合即可改变探针指向；可利用远程控制 (从办公室到实验室，或在全球范围内) 通过互联网连接进行调试；此外还包括 Agilent 科技推出的、用于实现功能强大的验证功能的逻辑分析器可选配件，可以探测包括从 FPGA 内部到板上任何地方的交叉互联信号。

5.10.3.5 PlanAhead

PlanAhead 工具简化了综合与布局布线之间的设计步骤，能够将大型设计划分成较小的、更易于管理的模块，并集中精力优化各个模块。此外，还提供了一个直观的环境，为用户设计提供原理图、平面布局规划或器件图，可快速确定和改进设计的层次，以便获得更好的结果和更有效地使用资源，从而获得最佳的性能和更高的利用率，极大地提升了整个设计的性能和质量。PlanAhead 的主要功能包括：

1. 轻松实现引脚规划的 PinAhead 技术

PlanAhead 包含 PinAhead 技术，可以帮助用户更好地处理引脚分配的复杂性问题。PinAhead 提供了一个以全自动或半自动方式将 I/O 端口分配到物理封装引脚上的环境。

2. 整合了 ExploreAhead

ExploreAhea 是一种实现探索工具，通过管理多个实现运行。ExploreAhead 允许用户根据他们指定的策略或者作为工厂默认方法发售的预定策略，执行多个实现操作。在 Linux 环境下，ExploreAhead 具有在远程主机上运行设计的能力。

3. 可完成基于模块的增量设计

PlanAhead 提供了层次化、基于模块的、模块化和增量设计方法，让设计者只需改变一部分设计，而保持其它部分的完整性，从而缩短了设计迭代。即使是在经常改动的情况下，它也能让用户保持所需的性能。

4. 提高设计的信号完整性

PlanAhead 提供了检查加权平均 SSO(WASSO) 分析限制的功能。这使得设计者能够更轻松地限制 FPGA 输出处的触地反弹数量，并能够防止发生 FPGA 引起的其它器件的操作失误。

5. 支持部分重配置

PlanAhead 简化了针对部分重配置的、功能强大但复杂的设计流程。部分重配置是一种独特的方法，可以在静态部分仍然工作的情况下改变设计的动态部分。部分重配置可以让用户减小设计的尺寸、重量、成本和功耗。

6. 基于 TimeAhead 的延时估计

TimeAhead 是一种灵活的、集成到 PlanAhead 中的时序分析器，它让用户在进行布局和布线之前就可以估计布线延迟。采用基于 PlanAhead 模块的方法，可在完成布局和布线的同时，提高时序估计的准确度。

5.11 ISE 与第三方软件的配合使用技巧

5.11.1 Synplify Pro软件的使用

在 FPGA 设计中，许多设计人员都习惯于使用综合工具 Synplify Pro。虽然 ISE 软件可以不依赖于任何第三方 EDA 软件完成整个设计，但 Synplify Pro 软件有综合性能高以及综合速度快等特点，无论在物理面积上还是工作频率都能达到较理想的效果。因此如何在 ISE 中调用 Synplify Pro 综合工具，并进行无缝的设计连接仍然是设计人员需要解决的一个设计流程问题。

1.Synplify Pro综合软件的安装

下面介绍 Synplify Pro 的安装步骤。运行安装程序，欢迎界面过后，将出现如图 5-67 所示的安

装选择界面，可以根据自己的需要选择相应的组件。然后按照默认选项继续即可完成安装。在 Synplify 安装完后,还需要安装 Identify。在开始 程序 Synplify 菜单栏中会出现“Identify 211 Installation”,双击即开始安装，一般来讲，可以按照默认选项继续，直至安装完毕。安装完之后需要添加授权的 License 文件，才能正常使用。

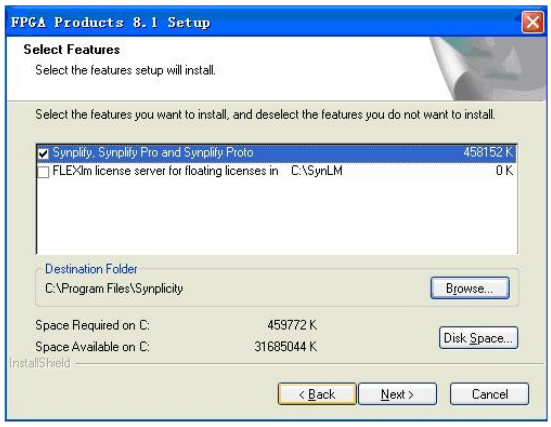


图5-67　Synplify的安装选择界面

2. 关联ISE和Synplify Pro

图5-68 选择Preference菜单项　图5-69 ISE集成工具设定页面

完成了 Synplify Pro 安装后，需要将其和 ISE 软件关联后才能使用 Synplify Pro 进行综合。运行 ISE 软件，在主界面中选择“EditlPreference”菜单项，进行“Reference”设定如图 5–68 所示。在弹出的 Preference 对话框中选择“Integrated Tools”选项卡。该选项卡用于设定与 ISE 集成的软件的路径，第三项的 Synplify Pro 就用于设定 Synplify Pro 仿真软件的路径，如图 5–69 所示。单击 Synplify Pro 文本框后面的按钮，会弹出一个文件选择对话框，选择 Synplify Pro 安装路径下 bin 目录下的“synplify_pro.exe”文件即可。注意：在“Integrated Tools”选项卡中还可以看到其他几个可以和 ISE 进行无缝链接的第三方软件，如 ModelSim、synplifyLeonardoSpectrum、Chip Scope Analyzer 等软件。

3.Synplify Pro的使用方法简介

1） **Synthesis 简单地说就是将 HDL 代码转化为门级网表的过程，其对电路的综合包括以下 3 个步骤：**

首先，HDL compilation 把 HDL 的描述编译成已知的结构元素；其次，运用一些算法，对设计进行面积优化和减小时延。在没有目标库的情况下，Synplify 只能执行一些最基本的优化措施；最后，将设计映射到指定厂家的特定器件上，并执行一些附加的优化措施，包括根据由器件供应商提供的专用约束进行优化。工程文件以 *.prj 作为扩展名，以 tcl 的格式保留了以下信息：设计文件、约束文件、综合时开关选项的设置情况等。 1)Synplify Pro 用户界面介绍 Synplify Pro 是标准的 windows 应用程序，所有功能均可以通过菜单选择来实现。下面按照图 5–70 中数字所标示的次序，对其界面作简要介绍。图中 1 表示 Synplify 的主要工作窗口，在这个窗口中可以详细显示设计者所创建工程的详细信息，包括工程的源文件，综合后的各种结果文件。同时如果综合完成后，每个源文件有多少错误或者警告都会在这个窗口显示出来。图中 2 表示 TCL 窗口，在这个窗口中设计者可以通过 TCL 命令而不是菜单来完成相应的功能。图中 3 表示观察窗口，在这里可以观察设计被综合后的一些特性，比如最高工作频率等。图中 4 是状态窗口，它表示现在 Synplify 所处的状态，比如下图表示 Synplify 处于闲置状态，在综合过程中会显示编译状态、映射状态等等。图中 5 所示的一些复选框，可以对将要综合的设计的一些特性进行设置。Synplify 可以根据这些设置对设计进行相应的优化工作。图中 6 是运行按钮，当一个工程加入之后，按这个 RUN 按钮，Synplify 就会对工程进行综合。图中 7 所示的是 Synplify 的工具栏。

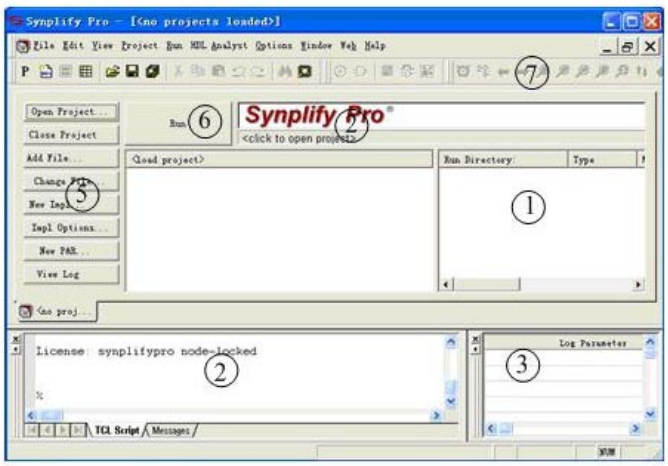


图5–70 Synplify Pro综合工具示意图

2) 建立工程

添加源文件 建立工程首先需要打开 Synplicity Pro。点击“开始”菜单，依次选择“程序 Synplicity Synplify Pro”，启动 Synplify Pro。在工程窗口中包含了以下内容：源文件信息、结果文件信息和目标器件信息。

缺省情况下，当 Synplify 启动时将自动建立一个新工程。这时，可以选择将工程，以新名字保存。如果结束了一个工程的操作，想新建一个工程，则可以选择“FILE NEW”；然后选择工程文件，就可以建立一个新的工程。这项操作也可以通过工具条来进行，单击工具条的 P 图标，则在弹出对话框选择工程文件即可。

新建工程之后，需要将源文件添加进来。点击“ADD FILE”按钮。添加源文件和约束文件。Synplify Pro 把最后编译的“module/entity and the architecture”作为顶层设计，所以需要把顶层设计文件用左键拖拉到源文件菜单的末尾处或者点击“Impl Options”按钮，在 Verilog 属性页中设置顶层模块的名称。

3) 工程属性设置

添加完源文件后需要设置工程属性，点击“Impl option”按钮出现属性页对话框，如图 5-71 所示。下面介绍常用的芯片设置、综合选项、约束设置以及实现结果选项等参数的配置。

图5-71 设置器件属性页

首先，设置 FPGA 芯片信息。打开“Device”属性页，分别设置器件厂家器件

型号、速度级别和封装信息。根据设计的速度和面积要求。可以设置最大扇出系数，缺省是 10000。根据该工程所属模块是否和片外有信号联系，选中或者不选中“Disable I/O insert”，如果选中该选项，则 Synplify Pro 不会为输入输出信号加缓冲，缺省为不选。

设置通用综合选项。点击“options”属性页，选中“Symbolic FSM Compiler”，

Synplify Pro 会在综合过程中启动有限状态机编译器，对设计中的状态机进行优化。选中“Resource Sharing”选项，则启动资源共享；设置了资源共享后，设计的最高工作频率会低于不选中的情况，但是资源会节约很多，因此在设计能够满足时钟频率要求的情况下，一般选中以节省资源。选中“Use FSM Explorer”选项，即可以用 synplify 内置的状态机浏览器观察状态机的各种属性。选中“Pipelining”选项，即启动流水，在高速时钟设计中，如果其他措施都不能达到目标频率则最好选中此项。

设置约束选项。点击“Constraints”属性页，设置模块最高工作频率以及添加约

束文件 (.sdc)。过严或是过松的约束都达不到最佳的效果。一般可先尝试通用的约束，如时钟扇出限制等；如果没有达到要求，可加入一些严格的具体约束，同时注意放松一些可以放松的约束。需要注意的是，综合约束的结果是估计值，应该以布局布线的结果为准。

设置实现结果。点击“Implementation Results”属性页，设置综合结果放置的目

录，综合结果的文件名称。同时一定要将“Write Vendor Constraint File”和“Write Verification Interface Format”选项选中。

4) 时序约束

定义时间约束是为了让综合结果满足预期的时序要求，时间约束通常分为两类：是

通用时间约束，用于目标结构的时序要求；二是黑盒时间约束，用于在设计中指定为黑盒的模块时间约束。在 Synplify Pro 中，可通过 SCOPE、约束文件以及综合属性和指示等 3 种方法添加时序。本节主要介绍利用约束文件添加约束的方法。

约束文件采用 Tcl 语言，以 *.sdc 保存，用来提供设计者定义的时序约束、综合属性以及 FPGA 生产商定义的属性等。约束文件既可以通过 SCOPE 创建编辑也可以使用正文编辑器创建编辑可被添加到在工程窗口的代码菜单中也可以被 Tcl 脚本文件调用。

5) 综合属性和指示

(1) 综合属性和指示简介

综合指示用于控制综合中编译阶段的设计分析，因而必须加入到源代码中。属性是在编译后读入的，因而既可以在源程序中说明，也可以在约束文件中说明。约束文件提供了较大的灵活性，使得可以仅修改约束而不用重新编译源程序，因而是强烈推荐采用的方法。在 Verilog 源程序中，说明指示或属性采用注释的方法语法如下：

// synthesis directivelattribute = "value"

或 /* synthesis directive lattribute = "value" */

(2) 综合指示

综合指示用于通知 Synplify Pro 软件某些用户定制的设置，常以注释的形式出现在源代码后面，Synplify 软件会自动识别相应的说明，按照用户指令完成综合。常用的综合只是如下：

① black_box_pad_pin

声明用户定义的黑盒管脚作为外部环境可见的 I/O pad。如果有不止一个端口列在双引号内,则以逗号分开。由于 Synplify 提供了预定义的 I/Os，一般不需要这一属性。其语法如下：

object /* synthesis syn_black_box black_box_pad_pin = "port_list" */ ;

例如：

module BS(D,IN,PAD,Q) /*synthesis syn_black_box black_box_pad_pin="PAD" */;

② block_box_tri_pins

声明黑盒的一个输出端口是三态，如不止一个列在双引号内，则以逗号分开。其语法如下：

object /* synthesis syn_black_box black_box_tri_pins = "port_list" */ ;

例如：module BBDLHS(D,E,GIN,GOUT,PAD,Q) /* synthesis syn_black_box

black_box_tri_pins="PAD" */ ;

③ full_case

仅用于 Verilog 中的 case 语句,表明所有可能的状态都已经给出,不需要其他逻辑保持信号的值,其语法如下：

object /* synthesis full_case */

其中 object 可以是 case、casex、casez、statements 和 declaration。

④ parallel_case

仅用于 Verilog 中 case 语句，表明生成一个并行的多路选择结构而不是一个优先译码结构。其语法如下：

object /* synthesis parallel_case */

其中 object 可以是 case、casex、casez、statements 和 declaration。

⑤ syn_block_box

说明一个模块或组件为黑盒，仅利用其界面进行综合，而不管内部是否为空，也不进行优化。一般应用于厂家原语或宏或 IP 等用户定义的宏。其语法如下：

object /* synthesis syn_black_box */ ;

其中 object 可以是 module 和 declaration。

⑥ syn_encoding

强制选择自动机实现的方式，其可选值 (value) 如下：

default：综合根据状态的数量选择编码方式编码方式可以是 onehot gray sequential；

onehot：采用 onehot 编码方式；

gray：采用格雷码；

sequential：采用自然码；

safe：如果不能到达任一个状态时让其回到复位态。

syn_encoding 的语法如下：

object /* synthesis syn_encoding = "value" */ ;

其中 object 是状态寄存器定义。

⑦ syn_isclock

说明黑盒的一个输入是时钟信号。对名字为 clk 、rclk 、wclk 的黑盒输入信号，软件自动当作时钟，可以用这个属性说明任意输入信号为时钟信号。其语法如下：

object /* synthesis syn_isclock = 0|1 */ ;

其中 object 是黑盒的 input port。

例如：

```
module ram4(myclk, out, opcode, a, b) /* synthesis syn_black_box*/;
output [7:0] out;
input myclk /* synthesis syn_isclock = 1 */;
input [2:0] opcode;
input [7:0] a, b;
/* Other coding */
```

⑧ syn_keep

保证被指定的 wire 在综合中保持不动，不会被优化掉，常用于用 define_multicycle_path 或 define_false_path，用了 –through 选项。如果你用了这一属性，将生成一个 keepbuf，可对其定义时间约束，且这个 Buffer 只占用一个位置，不出现在门级网表里。其语法如下：

object /* synthesis syn_keep = 0|1 */ ;

其中 object 是 wire 或 reg 声明。

⑨ syn_noprune

用来保持一个或多个 component 的实例，而不管其输出能否完成映射。一般在没有该指示的情况下，未用输出端口的实例会从 EDIF 文件中删除。syn_noprune 可被置于约束文件中，其语法如下：

.sdc 文件中：

```
define_attribute {modulelinstance} syn_noprune {0|1}
```

Verilog 中：

object /* synthesis syn_noprune = 0|1 */ ;

其中 object 可以是 module、declaration，也可以是实例。

⑩ syn_preserve

用在某些独立的寄存器上或模块，使模块中的所有寄存器在优化时保持不动，也可用于保持某个自动机在优化时不动。其语法如下：

object /* synthesis syn_preserve = 0|1 */ ;

其中 object 可以是寄存器定义信号，也可以是 Module。 syn_sharing

确定综合时是否对运算符进行资源共享。缺省值是禁止，也可以在 project 视窗里设置这一选项。其语法如下：

object / * synthesis syn_sharing = " onloff " */ ;

其中 object 可以是 module 定义语句。 syn_state_machine

对设计中的某组状态寄存器进行自动机优化，其语法如下：

object /* synthesis syn_state_machine = 0|1 */ ;

其中 object 是该组状态寄存器。 syn_tco<n>

提供黑盒的输出延迟信息，其语法如下：

object /* syn_tcon = "[!]clock –> bundle = value" */ ;

其中 bundle 是总线或标量信号的集合。 syn_tpd<n>

提供穿过黑盒的组合逻辑的传输延迟信息，其语法如下：

object /* syn_tpdn = "[!]clock –> bundle = value" */ ;

其中 bundle 是总线或标量信号的集合。 syn_tristate

指定黑盒的一个输出端口为三态端口，其语法如下：

object /* synthesis syn_tristate = 0|1 */ ;

其中 object 可以是黑盒的 output port。 syn_tsu<n>

说明一个黑盒的输入要求的建立时间，其语法如下：

object /* syn_tsun = “[!]clock –> bundle = value” */ ; translate_on/translate_off 用于与其他综合软件的兼容，这两者经常配对使用。在这两个指示中间的所有代码将在综合时被忽略，也可以用于在源代码中插入一段仿真代码。其语法如下：

/* synthesis translate_off */

综合时忽略的代码

/* synthesis translate_on */

6) 综合报告解读

综合报告主要由 3 部分组成：编译报告、映射优化报告以及时序报告，但是该报告是冗长的，不容易快速找出用户所关心的结果。因此，Synplify 公司提供了综合报告观察窗，如图 5–70 中第 3 部分所示，可从综合报告文件中取出重要的信息。该窗口的使用非常简单，点击空白的参数显示栏，在下拉栏中选择要查看的项目，则会在同行的右侧显示出结果，如图 5–72 所示。

Log Parameter	dpd_v5
dpd_v5 Part	xc4vsx35ff668-12
Worst Slack	996.968
System - Slack	998.238
System - Estimated Frequency	567.5 MHz
dpd_v5 I/O primitives	194
dpd_v5 Total Luts	1028 (3%)
dpd_v5 Register bits (Non I/O)	2723 (8%)

图5–72 Synplify综合结果示意图

5.11.2 ModelSim软件的使用

ModelSim 软件是一款强大的仿真软件，具有速度快、精度高和便于操作的特点，此外还具有代码分析能力，可以看出不同代码段消耗资源的情况。其功能侧重于编译和仿真，不能制定编译的器件和下载配置的能力，所以需要和 ISE 等软件关联。

1. ModelSim 仿真软件的安装

下面介绍一下 ModelSim 的安装步骤。

1) 运行安装程序后，出现图 5.11.7 所示的界面，如果拥有有效的 License，可以选择完全版 (Full Product) 安装，反之，则应当选择评估版 (Evaluation Edition) 安装。

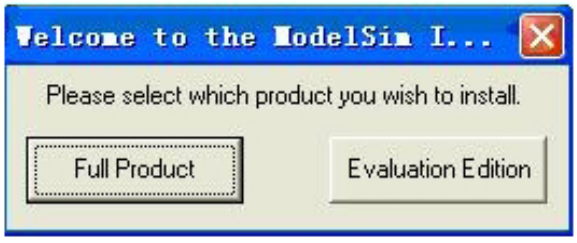


图5-73 ModelSim版本选择窗口

2) 选择完安装类型之后，下一个步骤就是设定安装路径，如图 5-74 所示。

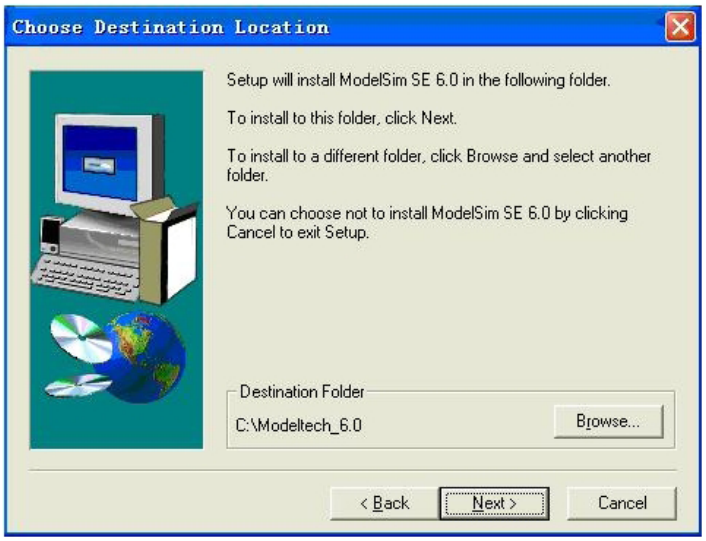


图5-74 ModelSim安装路径选择窗口

3) 如果选择的是完全版本，安装之后会出现 Liscense Wizard 对话框，如图 5-75 所示。

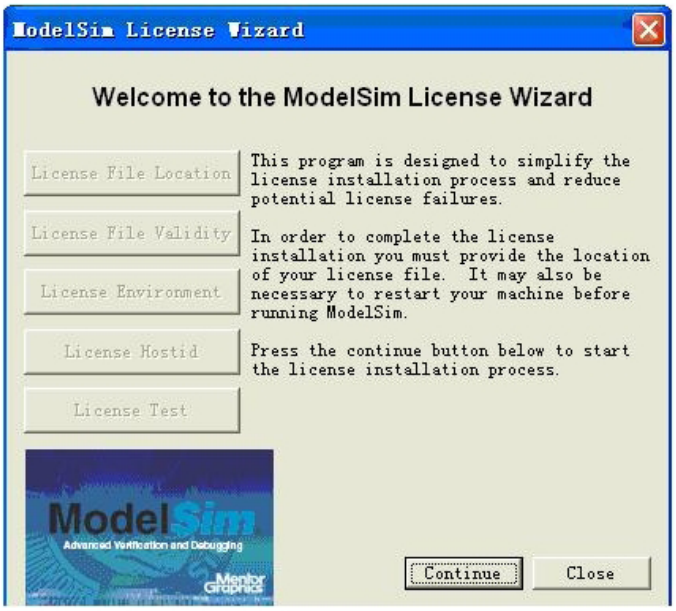


图5–75 ModelSim软件License管理向导

4) 单击 Continue 按钮后，会出现 License 文件选择的对话框，选择有效的 License 文件。单击 OK 按键后，系统会自动进行一系列的有效性检查，只有合法的 License 文件，才能使 ModelSim 正常工作。

2. 关联ISE和ModelSim

完成了 ModelSim 安装后，需要将其和 ISE 软件关联后才能使用 ModelSim 进行仿真。运行 ISE 软件，在主界面中选择“EditlPreference”菜单项,进行 Reference 设定。在弹出的“Preference”对话框中选择“Integrated Tools”选项卡。该选项卡用于设定与 ISE 集成的软件的路径，第一项的“Model Tech Simulator”就用于设定 ModelSim 仿真软件的路径，单击“Model Tech Simulator”文本框后面的按钮，会弹出一个文件选择对话框，选择 ModelSim 安装路径下 win32 目录下的“modelsim.exe”文件即可。

3. 在ModelSim中指定赛灵思的仿真库

ModelSim SE 版在发行时是不带任何 FPGA 厂家的仿真库，因此用户必须手动编译这些库，由此面临的一个问题就是怎样建立各 FPGA 器件的仿真库，目前各 FPGA 厂家都支持用户编译库，所以实现比较简单。 在 ModelSim 中编译赛灵思仿真库有很多方法，下面介绍一种比较常用的方法，分为 3 步来完成。

1、点击“开始 / 运行”按钮，执行下面的命令 ：

“compxlib –s mti_se –f all –l all –o f:\modeltech_6.2b\xilinx_libs –p f:\Modeltech_6.0\win32”

其中，f:\modeltech_6.2b 为 ModelSim 软件的安装目录，用户可根据自己的目录来替换。等待上述命令运行完毕，其运行时间较长，用户不要中途中断。

2、在赛灵思本地库编辑成功后，在相应的目录下，会自动生成 Modelsim.ini 的文件，用任何一个文本编辑器将该文件中 [Library] 目录下 (除 others 以外) 的内容添加到硬盘上相应的另外的 ModelSim 安装目录下同名

“modelsim.ini”文件中的相应 [Library] 位置。

3、最后进入 ISE 主界面，点击“Edit”下拉菜单按纽，选中下拉菜单中的“Prefrence”选项，再选中“Intergal Tools”页面，重新指定 ModelSim 可执行文件即可。退出所有软件，以后再对赛灵思的设计进行仿真都不需要进行库的处理了。

4. ModelSim使用方法简介

本节主要简单介绍 ModelSim SE 6. 6.2b 的使用方法，主要包括建立工程和基本 Verilog 仿真，更多的操作方法需要在实际应用中熟悉并掌握。 1) 建立工程 使用 ModelSim 建立工程主要包括 5 个基本步骤：

<1> 启动ModelSim，选择菜单“File New Poject”，会打开“Creat Project”对话框，如图5.11.11所示。在“Creat Projec”t 对话框中填写“Project Name”为“test”，然后在“Project Location”栏中选择 Project 文件的存储目录，保留“Default Library Name”的设置为 work。点击 OK 按键确认，在 ModelSim 软件主窗口的工作区中即增加了一个空的 Project 标签，同时弹出一个“Add items to the Project”对话框，如图 5–76 所示。

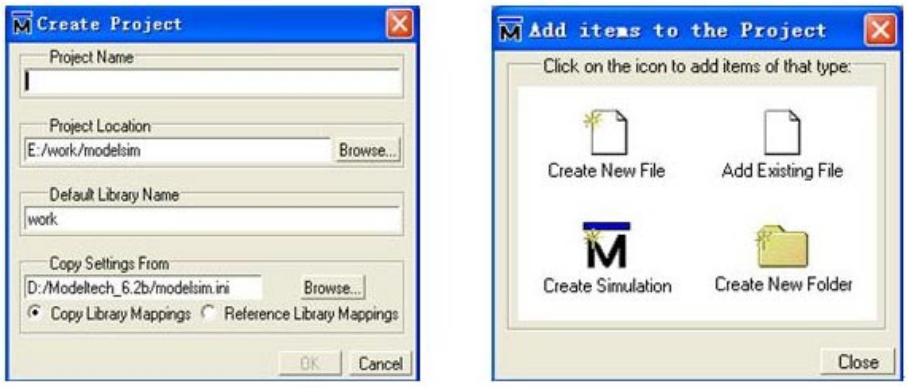


图5–76 ModelSim新建工程窗口图5–77 添加文件到工程向导示意图

<2>. 添加包含设计单元的文件。直接点击 Add items to the Project 对话框以后，在对话框中利用“Add Existing File”或“Create New File”选项，可以在工程中加入已经存在的文件或建立新文件。本节我们选择“Add Existing File”，弹出“Add file to the Project”对话框，如图 5–78 所示。点击对话框中的 Browse 按键，打开 ModelSim 安装路径中的 examples/tutorials /verilog/compare/ 目录，选取 sm.v 和 sm.v 文件 (选中多个文件时，只需要一直按住 Ctrl 按键，用鼠标点击即可)，再选中对话框下面的“Reference from current location”选项，然后点击 OK 按键确认。

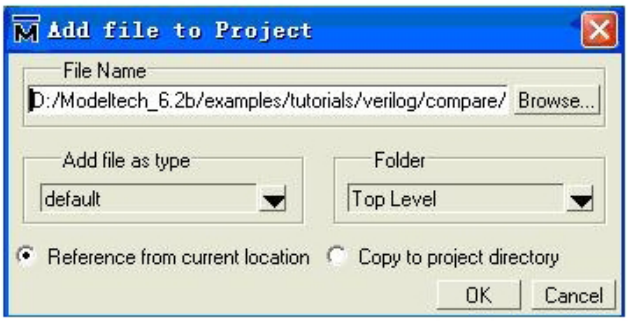


图5-78 ModelSIm添加文件选项示意图

<3>. 在工作区中的 Project 标签页中可以看到新加入的文件，单击右键，选取“Compile Compile All”命令对加入的文件进行编译，如图 -79 所示。

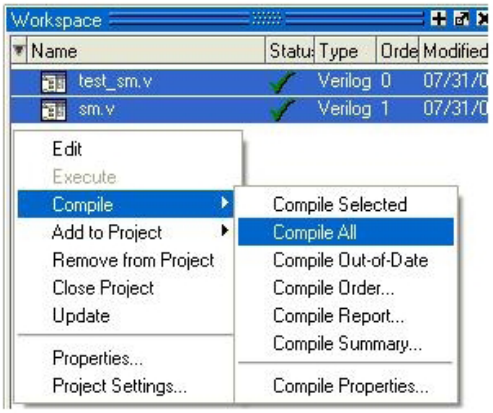


图5-79 ModelSim软件中的工程编译窗口

<4>. 两个文件编译完后，用鼠标点击“Library”标签栏。在标签栏中用鼠标点击 work 库前面的“+”，展开 work 库，就会看到两个编译了的设计单元。如图 5-80。

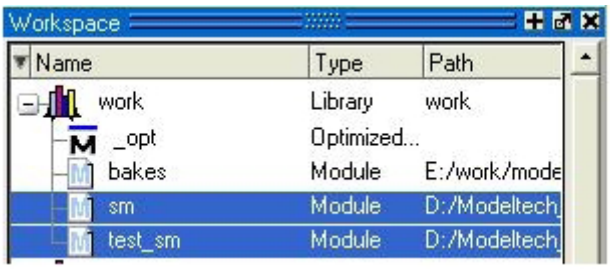


图5-80 编译后的设计单元示意图

<5>. 导入设计单元。双击 Library 标签页中的“test_sm”，在工作区中将会出现 sim 标签，并在右边的对象窗口列出了 test_sm 单元所用到的信号，如图 5-81 所示。

到此，一个工程就已经建立好了，接下来的就是开始运行仿真、分析和设计调试了。选择“File Close Poject”可以关闭当前目录。 2) 基本 Verilog 仿真 在准备仿真的时候，需要完成 (1) 中的所有步骤。然后继续进行下面的步骤：

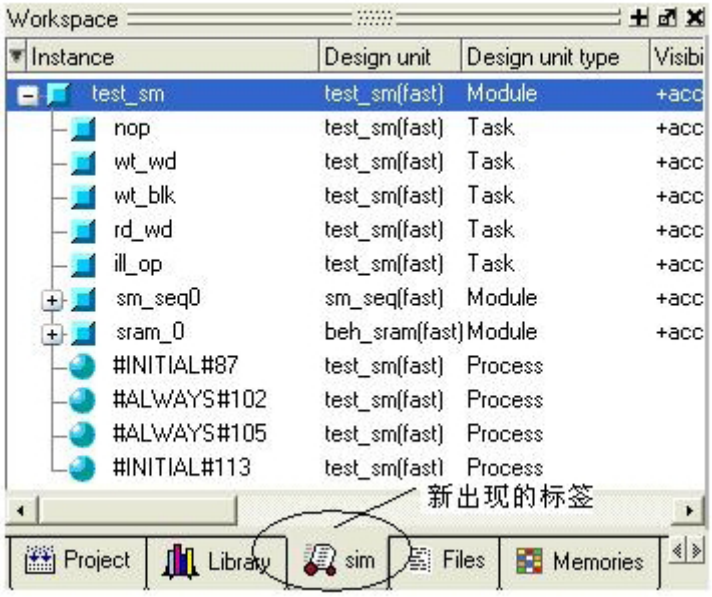


图5-81 将test_sm模块加入工作区示意图

<1>. 通过选择“View < 窗口名 >”调出 signal、list 和 wave 窗口。 也可以通过在主窗口命令行操作区的 VSIM 提示符下输入下面的命令：view signals list wave(回车)

<2>. 向 wave 窗口添加信号。在 signal 窗口中，单击右键，在弹出的菜单中选择“Add to Wave”选项中的“Signal in design”，将设计中用到的所有信号都列在 Wave 窗口中，如图 5-82 所示。

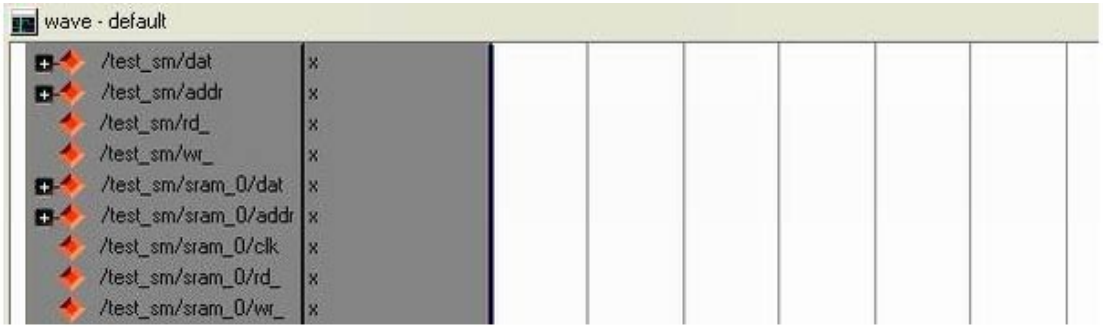


图5-82在Wave窗口中添加信号

<3>. 导入设计的时候，会在工作区打开一个新的 sim 标签，点击“+”展开设计层次结构，可以看到实例 test_sm、sm 等模块。点击 sim 标签中的顶层行保证 test_sm 模块显示在 source 窗口中。

<4>. 点击主窗口工具条的 Run 启动仿真，默认仿真长度为 100ns。或者在命令行输入 run。可以在仿真途中点击“Break”中断运行，在 source 窗口中查看中断时执行的语句。

<5>. 仿真完成，观察仿真波形如图 5-83。确认无误后退出仿真，如果有错则返回 source 区域修改代码。

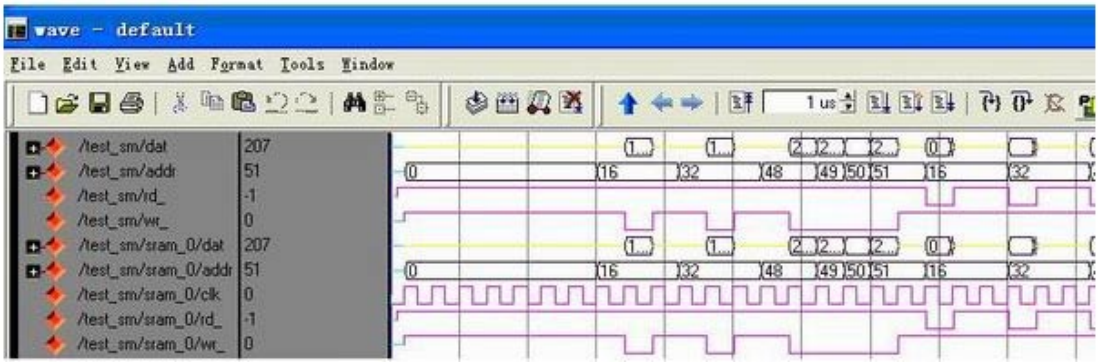


图5-83 test_sm模块的仿真结果示意图

5.11.3 Synplify Pro、ModelSim和ISE的联合开发流程

利用 Synplify Pro、ModelSim 和 ISE 进行联合开发的步骤基本如下：当工程设计完成后，首先需要利用 ModelSim 软件完成功能仿真，然后利用 Synplify Pro 进行综合优化，再在 ISE 中完成映射与布局布线，并借助于 ModelSim 完成布局布线后的时序仿真，最后在 ISE 中完成 .bit 文件的生成和 FPGA 芯片的配置。

通常上述流程有两种实现方法：第一，将前两者作为 ISE 的第三方插件，在 ISE 中通过按键自动调用；其次，就是通过相应的接口文件，手动完成各个流程。自动调用比较简单，只需要在 ISE 中进行简单的设计即可。

1. 自动调用流程

完成了 Synplify Pro 和 ModelSim 的安装，并在集成工具设定页面完成与 ISE 的关联后，如图 5.11.3 所示，单击 ModelSim、Synplify /Synplify Pro 文本框后面的按钮，会弹出一个文件选择对话框，选择 ModelSim、Synplify /Synplify Pro 安装路径中 bin 目录下的 *.exe 文件即可。

需要注意的是，并不是任意的 ISE 版本和任意的 ModelSim、Synplify/Synplify Pro 版本都可以实现无缝连接，只有在 ISE 发布后出现的第三方软件才能和 ISE 无缝连接，否则 Synplify/Synplify Pro 软件只能够用于综合逻辑，而不能识别赛灵思提供的 IP core，ModelSim

则不能用于嵌入式开发环境 (EDK) 设计的仿真。根据个人操作的结果，和 ISE 9.1 匹配的 Synplify /Synplify Pro 应当是 8.8 及其更高版本，相应的 ModelSim 应该为 6.1f 以后的版本。

完成了上述设定后，就可以直接在设计中调用 ModelSim 和 Synplify。在工程管理区的设计芯片上，点击右键，选择“Property”命令，即可打开用户设计的综合和仿真工具选择界面，如图 5.11.19 所示。在“Synthesis Tool”的下拉框中选择 Synplify(Verilog)、在“Simulator”中选择 ModelSin-SE Mixed。

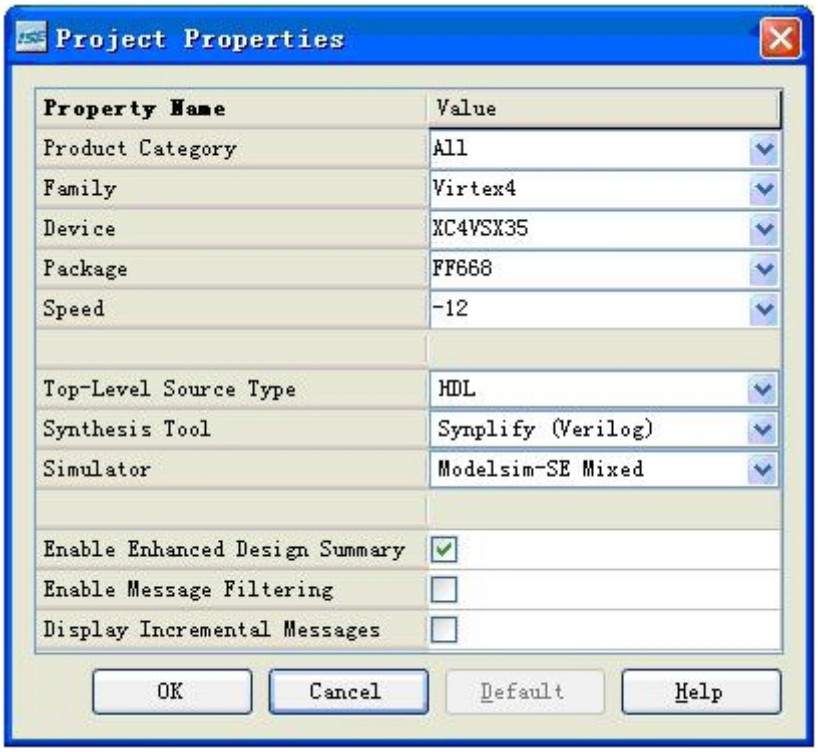


图5-84 设计工具选择界面

2. 手动调用

由于 ModelSim 只是完成功能验证，也 ISE 没有直接的数据交互，手动操作就是分开单独操作。手动调用 Synplify Pro 的方法比较灵活，但操作起来比较麻烦，用户可根据需要自行选择。首先单独启动 Synplify Pro 完成综合过程，再输出符合 ISE 格式的 EDIF 网表，在设置

工程属性时选择 EDIF 设计流程，ISE 仅完成网表的转换、映射和布局布线等操作。同时可在 Synplify Pro 中添加时序约束。

综合完成后，生成的后缀为 .edif 的文件就是综合输出的重要文件，是实现过程的输入，直接将其导入 ISE 即可。

5.11.4 ISE与MATLAB的联合使用

本节主要介绍 MATLAB 设计、ISE 实现以及二者联合测试的 FPGA 开发流程，这是全书的核心思想之一，也是目前最流行的设计方法。

MATLAB 软件是 MathWorks 公司的核心产品，具有用法简单、扩展性好、资源库丰富以及与其他软件接口方便的特点，已成为从事电子信息和信号处理领域人员必备的工具软件之一。MATLAB 和 ISE 的联合使用主要通过以下两个途径来实现：即 MATLAB 辅助 ISE 的方法，以及利用接口软件 System Generator 的方法。本书主要围绕着第一种方法展开讨论，但 System Generator 作为一种新兴的设计模式，具有强大的发展势头，这里对其也作了简单介绍。

1. MATLAB辅助ISE完成FPGA

所谓辅助，就是利用 MATLAB 来加速浮点算法的实现和功能测试。即在进行 FPGA 设计之前，先用 MATLAB 实现浮点算法，分析出算法的瓶颈所在，将程序的串行结构改造成并行结构，接着利用 MATLAB 完成定点仿真，得到满足性能需求的最小定点位宽以及中间步骤计算结果的截取范围，然后在 ISE 中完成设计。最后再利用 MATLAB 的定点仿真结果对设计进行功能验证，整个流程如图 5–85 所示。

关于怎样完成定点仿真以及模块输出结果截取的原理和方法将在以后的内容中展开详细关系讨论。此外，利用 MATLAB 对 FPGA 设计进行功能仿真是比较关键的，下面介绍如何将 MATLAB 和 ModelSim 结合起来使用。

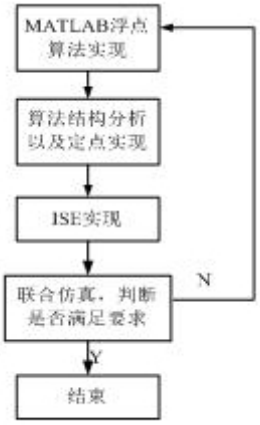


图5–85 MATLAB辅助ISE完成设计的流程图

首先，在 MATLAB 中产生仿真所需的输入信号，以十六进制的形式存放在数据文件中，通常放在后缀为 .txt 的文本文件中；其次在 ModelSim 中用 Verilog 编写仿真测试文件，并通过系统函数 $readmemh 将上述仿真数据文件中的测试向量读入，在 ModelSim 中做功能仿真和时序仿真，并调用 $fopen 函数打开另外一个数据文件，用 $fdisplay 函数将仿真结果写入；再次，在 MATLAB 中将 ModelSim 仿真输出数据文件中的数据读入一个数组中，可以作图分析或者利用统计手段来分析。此外，还可以将 ModelSim 的仿真输出与 MATLAB 的浮点性能作对比，来验证设计的性能。这样比利用其他方式要方便、直观，并且具有更高的正确性。

下面举例介绍 Verilog 语言中文件输入 / 输出函数的使用方法。

系统函数 $fopen 用于打开一个文件，并返回一个整数的文件指针。然后，$fdisplay 就可以使用这个文件指针在文件中写入信息。写完后，用 $fclose 关闭文件。

其语法格式为：integer <file_desc>; <file_desc> = $fopen("<file_name>", "<file_mode>");

$fwrite(<file_desc>, "<string>", variables);

$fclose(<file_desc>); 例如：integer W_file; // 定义文件指针

W_file = $fopen("W_file.txt");

$fdisplay (W_file, "@%h\n%h", a, b);

$fclose (W_file); 以上语句将“a”和“b”分别显示在“@%h\n%h”中的两个 %h 位置，并写入 Write_out_file 指针所指的文件 W_file.txt 中。

2) 读文件操作通过 $readmemh 和 $readmemb 来完成，分别对应的数据文件为 16 进制和二进制。其语法格式为：

```
reg [<memory_width>] <reg_name> [<memory_depth>];
 $readmemh ("<file_name>", <reg_name>);
```

例如：reg [15:0] c [0:15];

$readmemh (“R_file.txt”, c);

上例就是将 R_file 中的数据读入数组 c 中，然后就可以直接使用这些数据了。其中数据文件的格式为：

```
@2c
 45
```

其中，@2c 表示地址，为 16 进制数，45 表示该地址的数据。

2. System Generator工具简介

System Generator 工具由 MathWorks 与 赛灵思 合作开发而成，DSP 设计人员可使用 MATLAB 和 Simulink 工具在 FPGA 内进行开发和仿真来完善 DSP 设计。新型 8.2 版本 System Generator 使 DSP 系统和算法开发商不用写 VHDL 或 Verilog 编程，只需要利用 MATLAB 及 Simulink 来开发他们的设计。一旦浮点建模完成，设计工程师采用赛灵思的比特及周期精确工具箱对其进行量化并自动生成 HDL/RTL、用于 Xilinx FPGA 的网表或完整的比特流，包括新的 Virtex-5 LX 和 LXT 器件。最后，设计工程师在 Simulink 环境内采用高带宽硬件环境来验证并调试实际 FPGA 上的设计。

System Generator 的关键特性主要包括：

1、DSP 建模。利用包含信号处理 (如 FIR 滤波器、FFT)、纠错 (如 Viterbi 解码器、Reed-Solomon 编码器 / 解码器)、算法、存储器 (如 FIFO、RAM、ROM) 及数字逻辑功能的赛灵思模块集，在 Simulink 内构建和调试高性能 DSP 系统。 赛灵思模块集提供的模块可以使您导入 MATLAB 功能模块 (如创建控制电路) 及 HDL 模块 (System Generator 为 Mentor Graphics 的 ModelSim 和 Xilinx ISE 仿真器提供了 HDL 协仿真接口)。

2、Simulink 的 VHDL 或 Verilog 的自动代码生成。从 赛灵思 模块集实现行为 (RTL) 生成与对象明确的 赛灵思 IP 核。

3、硬件协同仿真。创建“FPGA 在环路 (FPGA-in-the-loop)”仿真对象是代码生成选项，允许您验证工作硬件并加速 Simulink 与 MATLAB 中的仿真。System Generator 支持以太网 (10/100/ 千兆位)、PCI、Cardbus 及硬件平台与 Simulink 之间的 JTAG 通信。

4、嵌入式系统的硬件 / 软件协设计。为 Xilinx MicroBlaze™ 32 位 RISC 处理器构建和调试 DSP 协处理器。System Generator 提供了 HW/SW 接口的共享存储器提取功能，自动生成 DSP 协处理器、总线接口逻辑、软件驱动器及协处理器使用方面的软件技术文档。 一个典型的 System Generator 开发实例如图 5-86 所示，图中形如赛灵思公司标志的图标就是 System Generator，只需要双击图标，就可以将浮点算法自动转化成 FPGA 实现。

赛灵思公司网站上提供了 System Generator 完整的使用手册和丰富的实例，读者如有兴趣，可自行阅读。

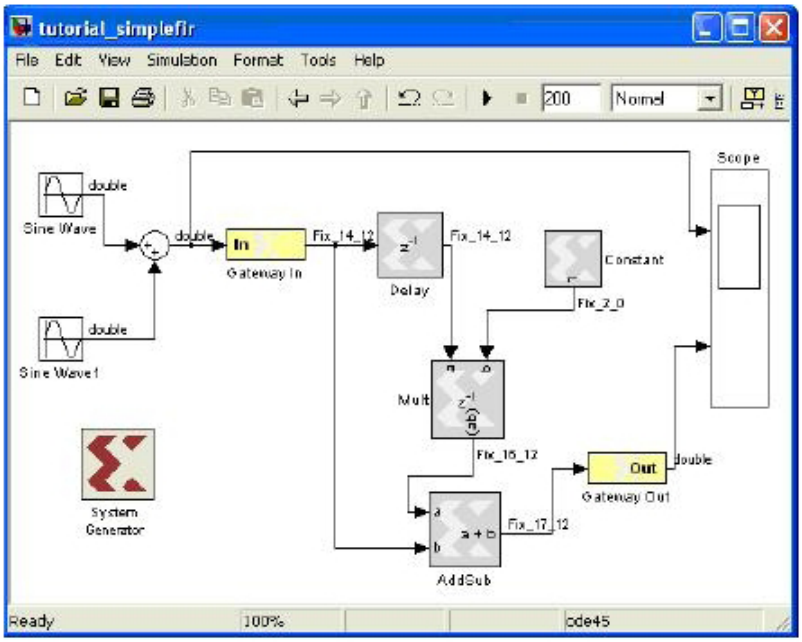


图5–86 System Generator开发实例

5.12 征服 FPGA 低功耗设计的三个挑战

功耗问题一直是 FPGA 被攻击的软肋之一，所以 FPGA 厂商都在投入大量研发来降低功耗，赛灵思在此领域有丰富的经验，这里介绍赛灵思公司降低功耗的主要技术。

随着芯片技术发展到 90nm 及以下，功耗已成为系统设计中的烫手问题。 此时，泄漏在总功耗中起着更重要的作用，使用新型介电材料的、更小的互连尺寸也会影响动态功耗。

系统设计者面临的另一个问题是设计功耗预算更紧了。 这并不局限于某些类型的系统，而是对大多数设计者都有影响。 带有众多板或模块的大型系统，以及便携式和消费类产品，都面临功耗预算问题。 大型系统中，功耗预算通常是针对整个系统的，也针对每个板或每个模块基础上的功率调节。 现在，由于每个板上都有多个电源，所以要提高一个板的功耗预算、同时不影响整个系统的功率分配计划，并不是件容易事。 在线路供电的消费类产品中，设计目的通常是尽可能利用最小、最廉价的电源来控制成本。 即便功率很轻微地超过某个特定型号电源的能力，也会导致使用更大型、更昂贵的电源，而从系统总成本来看，这是不能接受的。 设计者宁愿设计更多的特性以使产品差异化，也不愿使用较大的电源。

在便携式消费类产品中，压倒一切的目的是尽可能延长电池寿命。 对于此类产品而言，更长的电池寿命—工作和待机模式下—就是明显的竞争优势。

面对这些挑战，功耗问题为当今的系统设计者拉响了警报也就不足为奇了。

1、系统设计挑战

当今系统设计者所面临的功率使用、控制挑战主要有三个方面：静态功耗、动态功耗和起动功耗。 每种功耗代表不同的问题，需要使用不同的方法来计算和管理功耗。 静态功耗是器件处于无任何输入信号的静态条件下的功耗。也称为稳态或待机功耗。在当今 90nm 技术器件中，晶体管中的漏电流在静态功耗中占最大的比例。这通常也是便携式设备设计者需要考虑的关键参数，原因是它们会影响电池寿命，对于那些有大量时间处于待机状态，等待来自外部输入的器件尤其如此。

动态功耗是正常操作时的功耗，也称为负载功耗。 动态功耗取决于工作信号频率、互连电容和工作电压。由于电压相关性是平方函数，所以 90nm 器件中的电压下降大大降低了多种器件中的负载功耗。 不过，对于大型的、高性能的、高工作频率的系统而言，动态功耗仍是整个系统功率的主要组成部分。

起动功耗是器件上电时需要的功率，也称为上电或启动功耗，或加电浪涌功耗 (或电流)。 某些器件起动时需要的功率是正常工作下功率的很多倍，因此，对系统电源提出了很高的要求。 在严格控制电源尺寸和成本的消费类系统中，确保起动功耗不超过正常工作功耗是一个关键的设计目标。

2、赛灵思如何帮助管理系统功耗?

Virtex®–5 FPGA

Virtex®–5 FPGA 是世界上首款 65nm FPGA 系列，采用 1.0v、三栅极氧化层工艺技术制造而成，并且根据所选器件可以提供 330,000 个逻辑单元、1,200 个 I/O 引脚、48 个低功耗收发器以及内置式 PowerPC® 440、PCIe® 端点和以太网 MAC 模块。

功耗优化策略

1、利用 RocketIO GTP 收发器实现了 100 Mbps – 3.75 Gbps 低功耗串行连接功能；功耗低于 100 mW(在 3.2 Gbps 的速度下)

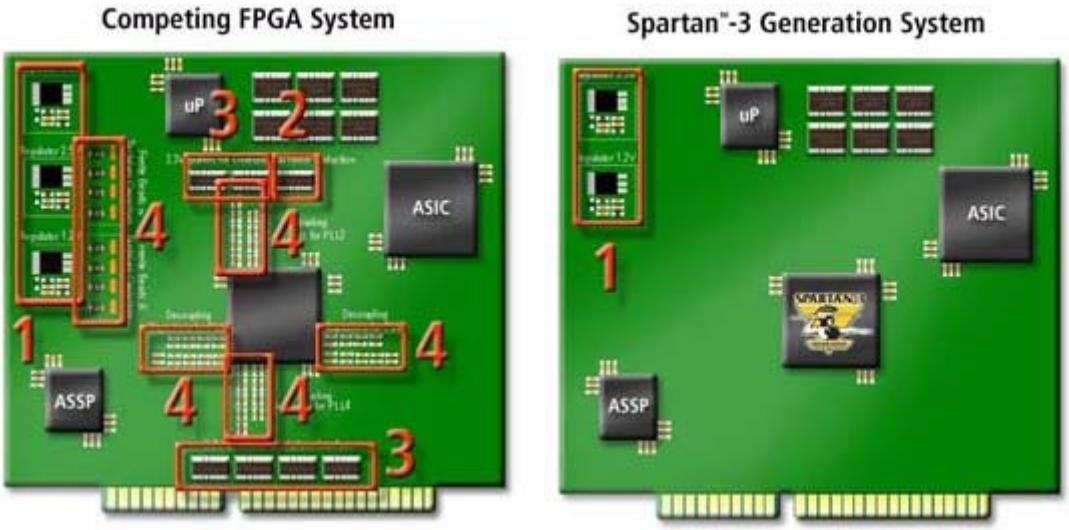
2、利用 65nm ExpressFabric™ 技术和低功耗 IP 模块将动态功耗降低了 35%。

3、三栅极氧化层技术使采用 65nm 工艺时的静态功耗同采用 90nm 工艺的 Virtex–4 FPGA Spartan®–3A 延伸系列

Spartan®–3A 延伸系列 FPGA 解决了众多大批量、成本敏感型电子应用中的设计挑战，

该 FPGA 系列具有 50,000 至 3,400,000 个系统门，可以提供包括总系统成本最低的集成

式 DSP MAC 在内的大量选项。



低功耗策略

集成特性和仅 2 个电源轨即可将外部元件数量降至最低

最低的静态功耗和获奖的功耗管理模式

强大的低成本安全性协议套件可以加速设计开发

CoolRunner™–II 1.8V CPLD

CoolRunner™–II 1.8V CPLD 系列号称是世界上成本最低、功耗最低的 CPLD，凭借其高性能、低功耗的特点，居于行业领先地位。CoolRunner–II CPLD 增加了 DataGATE、高级的 I/O 技术、业界尺寸最小的封装等突破性特点，为应对当今的设计挑战提供了终极的系统解决方案。

低功耗策略

利用业内功耗最低的 1.8V CPLD 达到您的系统功耗预算、利用全数字核和 FZP 工艺技术降低功耗 、低至 28.8 μW 的超低功耗 、16 μA 的典型待机电流、无需额外付费 、利用 DataGATE 实现超低功耗 、功耗管理 (占空比、时钟输入和输入引脚)、利用热插拔阻止插入过程中的输入管脚 、利用锁存通过调试阻止逻辑和快照信号部分 、通过阻止访问器件内的各种储存代码改善安全性 、CoolRunner–II 可以降低整个板功耗、利用 Fast Zero Power(FZP) 技术实现无可比拟的低功耗标准 ，同时提供高性能和低功耗，以及业内最低的待机电流，而无需使用掉电模式 。

此外，它还利用高级特性降低系统成本，例如 2–4 个 I/O 组，用于电压集成，高级的 I/O 和时钟管理 和 On the fly 重配置等。

5.13 高手之路——FPGA 设计开发中的进阶路线

从技术层面来讲，可编程逻辑领域是目前和未来半导体行业最活跃的领域之一，不再是单一地用于 IC 设计的原型验证，更多地用于提供集成的系统级解决方案。现代的 FPGA 不再仅仅是可编程逻辑，而是介于 ASIC 和 FPGA 之间的混合芯片，包含微处理器、收发器以及许多其它单元。所以对 FPGA 设计人员的要求也越来越高，已超出单一的逻辑设计范畴。因此，对于 FPGA 初学者来讲，需要明确个人的进阶路线，进而掌握快速开发的方法。下面给出作者个人的一些观点。

首先，熟悉一门硬件设计语言 (VHDL 或 Verilog HDL)，因为不管在哪种应用领域，HDL 语言都是 FPGA 开发的基础。目前国内使用 Verilog HDL 语言的开发人员较多一些，因此推荐读者学习 Verilog HDL。正因如此，本书的实例都通过 Verilog HDL 实现，并在附录中给出其简要的语法说明。要深入学习 Verilog HDL 语言的读者，可参阅参考文献 [2]。

其次，掌握 ISE Design Suit 相关软件的使用方法。ISE 软件可以完成设计输入、综合、仿真、实现和下载，涵盖了 FPGA 开发的全过程，从中读者可以真切体会到 FPGA 开发全过程。对于嵌入式开发人员，还需要掌握 EDK 软件操作。当掌握软件的基本用法后，可以深入了解各工具组件，如综合工具 XST、布局布线工具 PAR 等的运行机制，以便更好地在设计中利用其特性。本书以及其姐妹篇《ISE Design Suit 10.1 开发指南 (DSP 和嵌入式开发)》则定位于这一阶段。

第三，熟悉赛灵思 FPGA 芯片，包括不同类型资源的性能特点和使用方法。此时，赛灵思所发布的文档是首要参考资料。赛灵思针对每个系列的 FPGA 都提供了丰富而全面的文档，所以在开始任何一个系列的 FPGA 设计前，最好到赛灵思网站 (www.xilinx.com)，将该系列 FPGA 的页面上将所有的文档都下载下来，然后有针对性的做参考。

第四，参考赛灵思推出的开发板以及相应的参考设计，这是向高级进阶最有价值的部分。赛灵思在网上针对每个系列的 FPGA 都有文档说明，并都给出原理图。其开发板的文档说明非常详细详细，也很规范，有很大的参考价值。此外，在那些开发板里也有众多的外围接口电路，基本涵盖了常用的应用场合。参考外围电路芯片的数据手册，仔细体会设计的细节和应用方法。作为硬件工程师，阅读手册是一项基本技能。当然，在具备硬件平台的基础上，参考赛灵思网上的开发板是进阶路线中捷径的捷径。

第五，动手调通一块板子。有 PCB 设计能力的读者，可自行设计；否则可购买相应的开发板，将上面所有的硬件外设调通，并参照类似的开发板，独立完成赛灵思官方的参考设计。完成这一步，就步入高级设计的大门了。

第六，由于 FPGA 芯片以及开发技术发展很快，因此不仅要在工作中累积经验，还应该关注该行业的新技术和新动向，只有这样才能始终站在高处。

整体看来，FPGA 开发入门简单，进阶阶段不仅难度较大、所需知识面广，还是一个繁琐的工作。同时如果想从底层更深入的理解硬件设计，还需需要深厚的理论支持。因此 FPGA 开发是一条平坦但十分陡峭的路。

附录一、FPGA开发资源总汇

一、官网大全

1、赛灵思官方网站 http://china.xilinx.com/

2、赛灵思大学计划官方网站 http://china.xilinx.com/univ/

3、赛灵思开放源码硬件社群 http://www.openhw.org

4、赛灵思网上技术支持中心 http://china.xilinx.com/support/mysupport.htm

5、赛灵思开发工具下载中心 http://china.xilinx.com/support/download/index.htm

6、赛灵思 IP 核中心 http://china.xilinx.com/ipcenter/

7、赛灵思第三方合作伙伴信息 http://china.xilinx.com/alliance/index.htm

8、赛灵思技术解决方案大全 http://china.xilinx.com/technology/index.htm

9、赛灵思重要合作伙伴安富利 http://www.avnet.com/

10、赛灵思官网论坛 http://forums.xilinx.com/xlnx/

专业网站、FPGA开发博客信息

1、《FPGA 实用开发教程》田耘等编辑 http://www.eefocus.com/html/08-11/415501121236t2kE.shtml

2、RickySu 的博客 www.rickysu.com

3、FPGA 专业书籍网店 www.china-pub.com

4、EDA 中国门户网 www.edacn.net

5、电子创新网 www.eetrend.com

6、电子工程专辑 www.eetchina.com

7、电子设计技术 www.ednchina.com

8、电子系统设计 www.ed-china.com

9、21IC 中国电子网 www.21ic.com

10、电子产品世界 www.eepw.com.cn

附录二、编委信息与后记

《FPGA 开发全攻略》经过紧张有序的协作后终于面市了，在此特别向参与此书编撰的作者们表示衷心的感谢，他们是：

赛灵思高级产品经理　梁晓明

赛灵思亚太区市场经理　张俊伟

云创工作室北京邮电大学信息工程学院学生　田耘、徐文波等

赛灵思 PAE　苏同麒

中科院上海技术物理研究所　童 鹏　胡以华

希望这本追求基础、实用与深度的电子书能给 FPGA 开发者带来实际的帮助！也欢迎工程师朋友就 FPGA 开发中的需求与我们交流，以便我们可以开发后续版本提供更好的内容，提升本土 FPGA 应用创新水平！

再次感谢大家的支持！

张国斌
richard@eetrend.com
《FPGA 开发全攻略》电子书主编

附录三、版权声明

1、《FPGA 开发全攻略》著作权属于张国斌等人共同所拥有；

2、本着开源、服务大众的思想，我们授权任何对 FPGA 有兴趣的工程师免费下载并自由复制、传播该书；

3、非经作者 (张国斌为代表) 书面同意，不可以加以切割、编辑及部分内容传播；

4、任何商业用途必须得到作者 (张国斌为代表) 的书面同意。

5、作者联系方式 richard@eetrend.com