



10年口碑积累，成功培养50000多名研发工程师，铸就专业品牌形象

华清远见的企业理念是不仅要良心教育、做专业教育，更要做受人尊敬的职业教育。

《FPGA 应用开发入门与典型实例》(修订版)

作者：华清远见

专业始于专注 卓识源于远见

第 6 章 FPGA 设计开发软件 ISE 使用技巧

本章目标

- 熟悉 ISE 软件的安装与启动
- 掌握 ISE 下 FPGA 的设计流程
- 掌握 ISE 下创建工程的方式
- 掌握 ISE 下如何编译和仿真
- 掌握在线调试工具 ChipScope Pro 的使用技巧
- 掌握 ISE 常用的增量式设计技巧

6.1 ISE 软件简介

6.1.1 ISE 软件简介

Xilinx 作为当世界上最大的 FPGA/CPLD 生产商之一，长期以来一直推动着 FPGA/CPLD 技术的发展。其开发的软件也不断升级换代，由早期的 Foundation 系列逐步发展到目前的 ISE 9.x 系列。

ISE 是集成综合环境的缩写，它是 Xilinx FPGA/CPLD 的综合性集成设计平台，该平台集成了设计、输入、仿真、逻辑综合、布局布线与实现、时序分板、芯片下载与配置、功率分析等几乎所有设计流程所需工具。ISE 系列软件分为 4 个系列：WebPACK、BaseX、Foundation 和 Alliance。ISE WebPACK 系列可以在 www.xilinx.com 网站上直接下载，是一个免费软件，支持一些常用的器件族；ISE BaseX 系列的器件最大规模不超过 700k 系统门；ISE Foundation 系列是最早期 Foundation 系列的延伸；ISE Alliance 系列支持的器件族最全，功能强大，是 Xilinx 的主推设计平台，所以推荐安装 ISE Alliance 系列。

ISE 的主要特点如下。

1. 优良的集成环境

ISE 是一个集成环境，可以完成整个 FPGA/CPLD 开发过程。ISE 集成了很多著名 FPGA/CPLD 设计工具，根据设计流程合理应用这些工具，可以大大提高产品设计效率。

2. 简洁流畅的界面风格

ISE 界面风格简洁流畅，易学易用。ISE 的界面秉承了可视化编程技术，界面根据设计流程而组织，整个设计过程只需按照界面组织结构依次单击相应的按钮或选择相应的选项即可。

3. 丰富的在线帮助信息

ISE 有丰富的在线帮助信息，结合 Xilinx 的技术支持网站，一般设计过程中可能遇到的问题都能得到很好的解决。Xilinx 的官方网站上提供了相关软件（可供下载）、软件使用说明、软件更新、硬件资料、参考设计以及使用过程中常遇到的问题的解决等，此外还提供了大量的视频教程，便于用户学习。

4. 强大的设计辅助功能

ISE 秉承了 Xilinx 设计软件的强大辅助功能。在编写代码时可以使用编写向导生成文件头和模块框架，也可使用语言模板（Language Templates）帮助编写代码，在图形输入时可以使用 ECS 的辅助项帮助设计原理图。

另外，ISE 的 Core Generator 和 LogiBLOX 工具可以方便地生成 IP Core（IP 核）与高效模块为用户所用，大大减少了设计者的工作量，提高了设计效率与质量。

目前 ISE 的最新版本为 ISE 9.1i。Xilinx ISE 9.1 于 2007 年 3 月发布，是业界最完整的可编程逻辑设计解决方案，用于实现最优性能、功率管理、降低成本和提高生产率。ISE 9.1i 利用新 SmartCompile 技术，来帮助用户在更少的时间内实现业内最快速的 FPGA 性能。

6.1.2 ISE 7.1i 特点

由于本书中所涉及的例程都是在 ISE 7.1i 下完成的，这里对 ISE 7.1i 的特点做重点介绍。Xilinx 于 2005 年

3月推出针对 Xilinx Virtex-4 和新推出的 Spartan-3E 系列 FPGA 产品而优化的集成软件环境 (ISE) 7.1i 版。其相比与从前的版本有以下新特性。

1. 易用性有所提高

ISE 7.1i 中新的易用性特色可以加快工程师的设计过程。在设计流程中的每一步, ISE 7.1i 都提供了显而易见的实施结果。新的设计摘要视图 (Design Summary View) 和消息过滤 (Message Filtering) 功能突出了重要的设计信息, 从而减少了在详细的报告文件中搜索的需求。新的技术指示器 (Technology Viewer) 通过易于浏览的示意图表来显示合成后的实施结果。

2. 集成了两款新的仿真器

ISE 7.1i 中还集成了两款新的仿真器, ISE Simulator 和 ModelSim Xilinx Edition-III, 从而可实现更快的仿真和更大的设计容量。通过利用实时芯片上 (in-silicon) 调试功能来加强仿真能力, ChipScope Pro 和 ISE 7.1i 可使实时验证所需要的时间仅为 ASIC 或竞争 FPGA 验证流程的一半。ChipScope Pro 现在还允许设计人员从全球任何地方通过网络连接对系统进行验证和调试。

3. 通用性能提高

通过支持 64 位 Linux, ISE 7.1i 为更高密度的设计和开发创造了优越条件。重要的是, ISE 7.1i 可直接插入到现有 EDA 设计流程中, 与第三方 EDA 合作伙伴的合成、仿真、HDL 分析和验证等设计工具紧密集成。

4. 支持 Spartan-3E FPGA 系列和超低功耗 Spartan-3L FPGA

对于设计人员来说, 成本也是一项重大的挑战。ISE 7.1i 支持 Spartan-3E FPGA 系列和超低功耗 Spartan-3L FPGA, 因而可支持额外的大批量设计, 可大大节约设计者的成本。

总得来说, ISE 7.1i 独特的集成度、高速度以及易用性可以帮助设计人员解决所面临的最紧迫的一些挑战。新版工具集成了主要功耗分析、分层设计、仿真和调试等功能, 还支持目前应用越来越多的基于 Linux 的设计环境。工具中还包括了针对在所有性能领域全球都最快的 FPGA Virtex-4 系列的新速度文件。

与竞争解决方案相比, ISE 7.1i 的逻辑构造性能优势高达 70%, 同时在 DSP、嵌入式处理和连接功能方面也遥遥领先。设计人员可在设计中充分享受这些优势。ISE 7.1i 中还包括了对新推出的全球成本最低的 FPGA 产品 Spartan-3E 系列完全支持的功能。

6.2 ISE 软件的安装与启动

6.2.1 ISE 软件的安装

ISE 的安装改变了 license 管理方式, 在安装后并不需要任何 license 支持, 仅仅是在这安装过程中输入 ISE 的注册序列号 (Register ID) 即可。ISE 7.1i 安装启动界面如图 6.1 所示。

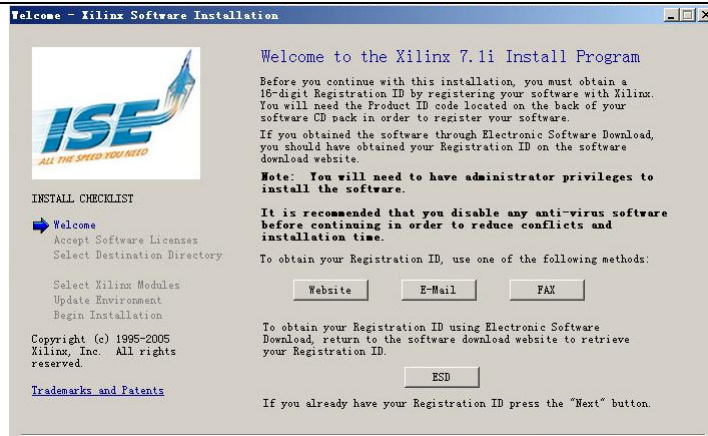


图 6.1 ISE 7.1i 安装启动界面

安装 ISE 时只需要根据所选的版本是在 PC 机或工作站上，然后根据软件的提示安装即可，这里不做详细叙述，只对安装的几个问题进行说明。

1. 环境变量的设置

安装过程结束后，为了能正常使用 ISE，还需要设置 ISE 的环境变量。假设 PC 机上 ISE 的安装目录为 C:\Xilinx。

(1) 如果操作系统是 Windows 98，需要在 austoexec.bat 文件中加入：

```
set Xilinx = c:\Xilinx           //设置环境变量
set PATH = %Xilinx%\bin\nt     //设置系统路径
```

(2) 如果操作系统是 Windows 2000，右键单击“我的电脑”，选择“属性”/“高级”/“环境变量”选项，在环境变量中加入：

```
变量名: Xilinx
变量值: C:\Xilinx
```

如图 6.2 所示。

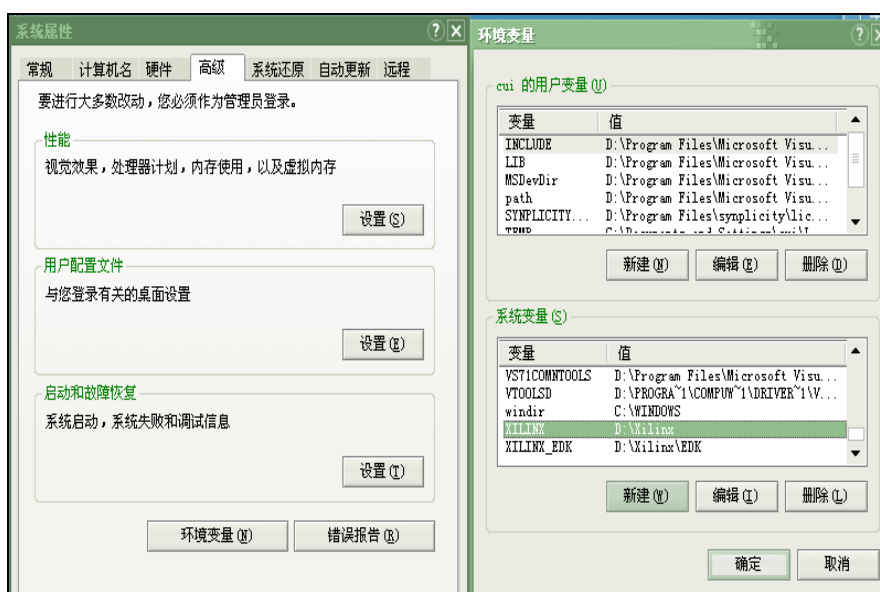


图 6.2 Windows 2000 环境变量配置

(3) 如果操作系统是 Windows NT，加入过程与 Windows 2000 相似。右键单击“我的电脑”图标，选择“属性”/“高级”/“环境变量”选项，在环境变量中加入：

变量名: Xilinx
变量值: C:\Xilinx

(4) 如果操作系统是 Window XP, 可以不用做上述环境变量设置, 安装完毕后, 系统会自动将 Xilinx 的环境变量添加到系统中。

(5) 如果操作系统是 Linux, 需要编辑 WINE 配置文件 /usr/local/ect/wine.conf, 加入 “Xilinx = C:\<your_Xilnx_dir>”, 并且设置环境变量 “setenv XIL_LINUX_WINE=1”。

2. 第三方软件的安装

Xilinx 公司的 ISE 集成开发环境中给许多第三软件预留了接口, 通过这个接口可以从 ISE 集成环境中直接启动第三方软件, 常用的第三方软件 (如: 仿真工具 ModelSim) 和综合工具 (Synplify/Synplify Pro)。在安装这些第三方软件时需要注意两个问题。

(1) 先安装 ISE, 后安装第三方工具。

在这种情况下, ISE 会自动捕获第三方软件的路径, 可以在 ISE 中直接启动第三方工具, 如果有的用户要更换第三方软件版本, 或先安装了第三方工具, 需要在 ISE 集成开发环境的“EDIT”\“Preferences”\“Integrated Tools”选项下进行设置, 如图 6.3 所示。

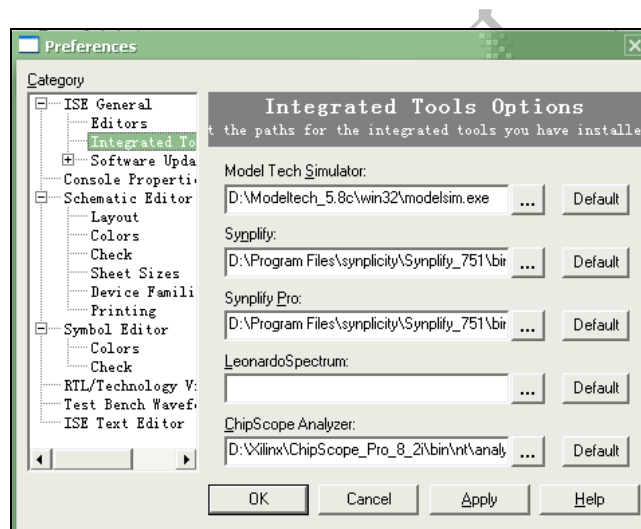


图 6.3 第三方软件接口路径设置

在该对话框中可以设置第三方软件 ModelSim、Synplify、Synplify Pro 的路径以及 ChipScope Analyzer, 用户可以根据软件的安装路径进行设置和修改。

(2) 第三方软件一般需要申请相应的 license, 并设置 license 管理项目。

6.2.2 ISE 软件的启动

ISE 软件的启动可以通过桌面的快捷方式, 如图 6.4 的左图所示: 双击 Xilinx ISE 7.1i 的图标即可打开 ISE 软件。也可以选择“开始”/“所有程序”/“Xilinx ISE 7.1i”/“Project Navigator”选项来打开, 如图 6.4 的右图所示。



图 6.4 ISE 软件的启动方式

ISE 软件启动后的界面如图 6.5 所示。

图 6.5 为 Project Navigator 界面，由图中可以看出，Project Navigator 的主界面由标题栏、菜单栏、工具栏、资源管理窗、当前资源操作窗、工程管理窗、HDL 编辑器工作区、信息显示窗和状态栏等部分组成。Project Navigator 是 ISE 的工程管理器，它是 ISE 所有集成工具的连接纽带。通过使用工程管理器，设计者可以创建、组织和管理自己的设计。

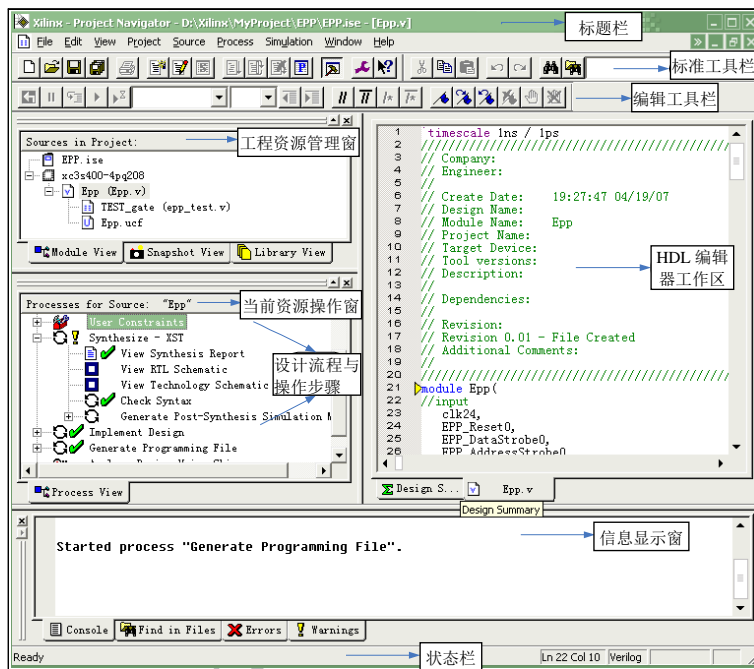


图 6.5 ISE 软件界面

6.3 ISE 软件的设计流程

Xilinx 公司的 ISE 软件是一套用以开发 Xilinx 公司的 FPGA&CPLD 的集成开发软件，它提供给用户一个从设计输入到综合、布线、仿真、下载的全套解决方案，并很方便地同其他 EDA 工具接口。

其中，原理图输入用的是第三方软件 ECS；状态图输入用的是 StateCAD；HDL 综合可以使用 Xilinx 公司开发的 XST、Synopsys 公司开发的 FPGA Express 和 Synplicity 公司的 Synplify/Synplify Pro 等；测试激励可以是图形化的 HDL Bencher，也可以由用户提供测试代码；通过 ModelSim XE(Xilinx Edition)或 ModelSim SE 进行仿真。

Xilinx 为 ModelSim 预留了接口，可以直接在 ISE 环境中打开，使用非常方便。并且 ModelSim 支持综合前、后仿真，以及时序仿真，功能很强大。

除了上述软件以外，也可以使用其他公司的相关 EDA 软件产品。

本节将对 ISE 的软件设计流程做一个全面的介绍。一般来说完整的 ISE 软件设计流程包括：电路设计与输入、功能仿真、综合、综合后仿真、实现、布局布线后仿真与验证以及下载调试等主要步骤，如图 6.6 所示。

具体讲解如下。

1. 设置工作环境

这一步并不是总是需要。通常用在第一次使用 ISE 或需要对某些项目进行修改时，一般有以下几项需要设置：这些设置主要是在“Edit” / “Preferences”下完成的，如图 6.7 所示。

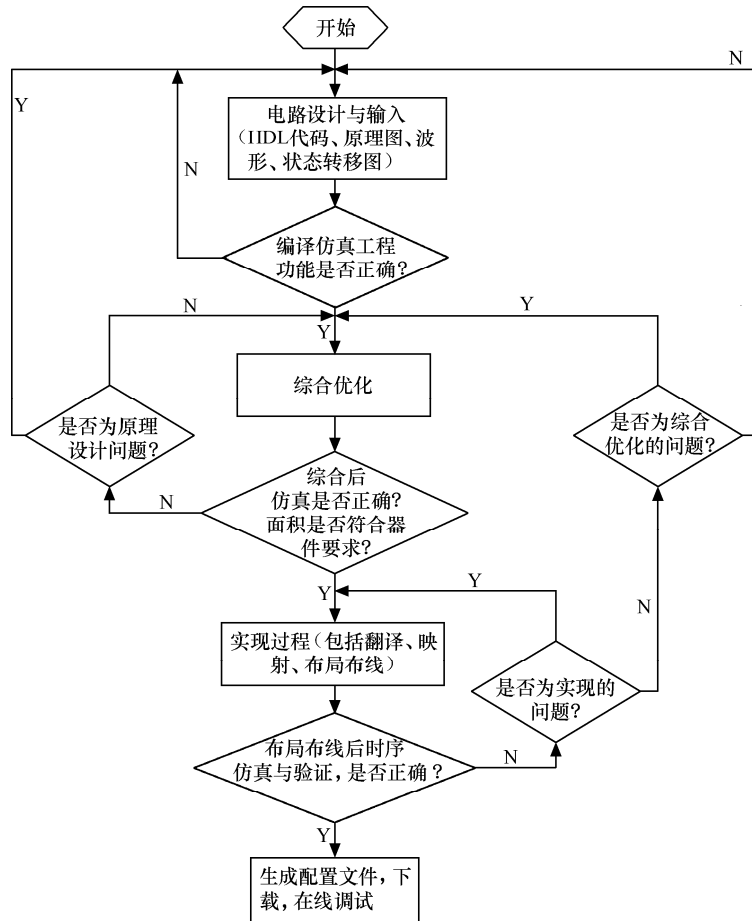


图 6.6 ISE 下 FPGA 设计流程图

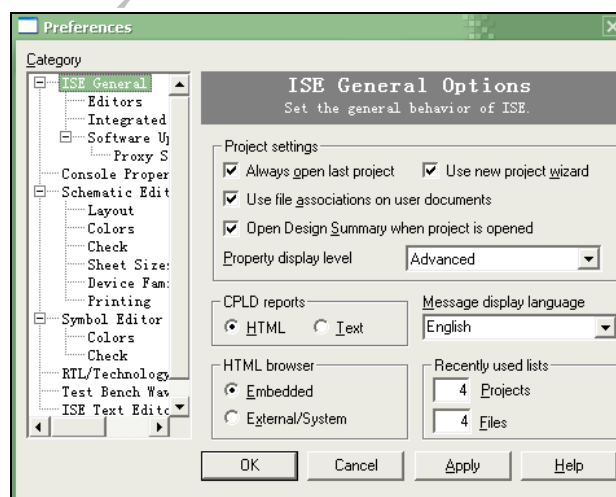


图 6.7 “Preferences”对话框

在 ISE General 中主要有下面几个选项可以进行设置。

(1) 常用的。这主要是设置项目管理器中文件的显示方式、字体、窗口的显示方式等，一般用默认值就行。

(2) 图形编辑器 (Schematic Editor)。这里可设置跳格键 (Tab) 的字符个数、编辑器的字体等，另外除了直接用 ISE 提供的 HDL 编辑器外，也可以采用第三编辑器，如 gvim 就是一款非常优秀的代码编辑器，有

感兴趣的读者可以在网上查阅相关资料。

(3) 流程设置。

(4) 工具设置。主要设置仿真器 ModelSim、HDL 测试台生成工具 HDL Bencher、状态图输入工具 State CAD 的工作目录。其实要设置的就是 ModelSim 的工作目录，因为后两项通常在安装完后 ISE 已经设好了。

2. 新建工程

这一步和其他的软件开发一样，ISE 要求在对文件进行综合或布线之前必须要有一个存在的工程，在新建工程时，需要设置以下几点。

(1) 工程名，最好用英文不要有汉字，因为 ISE 下有些工具对于含有汉字的文件目录支持的不是很好。

(2) 工程所在目录。ISE 所产生的输出文件将全部放在该目录下，但对源文件的目录没有要求。

(3) 器件家族。即设计中所采用的 FPGA 是 Xilinx 的哪一大类。

(4) 器件型号。具体大类下的哪一种器件，此外还要设置封装和速度等级等，这些信息都可以从芯片上提供的信息直接得到。

(5) 综合软件。由于 ISE 预置了 4 种可选的综合器接口(XST 为 Xilinx 自己开发,FPGA Express 是 Synopsys 公司的 OEM 版，在安装 ISE 时就已经装好了。而 Synplify/Synplify Pro 则需要另外购买并安装)，所以必须选择一种作为该工程的综合器。4 种综合器全部支持 Verilog 和 VHDL。但有一点必须注意：如果设计中用到原理图，则只能选择 XST 或 FPGA Express 作综合器，因为 Synplify Pro 不支持原理图方式。

3. 添加设计源文件

如果已有源文件，直接加入即可，否则可采用原理图方式或写 HDL 代码方式新建文件再加入。ISE 下支持多种新建资源类型，包括：User Document（工程说明文件）、VHDL module（VHDL 源代码文件）、VHDL Library（VHDL 库）、VHDL Package（VHDL 包）、VHDL Test Bench（VHDL 测试激励）、Verilog Module（Verilog 源代码文件）、Corgen Generator（IP 核生成工具）、Schematic（原理图）、Test Bench Waveform（测试激励波形）、BMM 文件（块 RAM 映射文件）、State Diagram（状态转移图）、UCF（用户约束文件）等。每种类型的资源在 ISE 都有对应的处理工具。

4. 写测试文件

这一步可以利用 HDL Bencher 工具自动产生。测试台的主要功能是给被测实体加上输入激励，再比较其输出是否与期望值一致，并给出一些提示信息。但推荐大家自己写测试代码，通过写代码的测试灵活性更强，而且对于比较复杂的设计，有时在仿真时还需要读取文件数据，或将最终的仿真结果写进文件或打印，这在 HDL Bencher 下是很难完成的。

5. 功能仿真

利用 ModelSim 来检查电路仿真结果是否正确。如果编译有错，则先将错误更改。如此反复直到仿真正确为止。这只是用于检查代码有无错误，ModelSim 最主要的作用在于通过仿真观察波形来验证设计的功能是否正确，这部分工作对于一个设计来说是非常重要的，因为如果前期的功能仿真做的不到位，会直接影响到最终电路功能的实现，必须在确保功能仿真没有问题的前提下，再进行下面的步骤。

6. 综合

通过这一步将设计转换成具体的电路图。如果设计有错，有可能综合通不过。这就要求用户必须按照可综合代码的风格来设计。另外，综合有很多属性是可以设置的，如果对设计中的某些项目（如速度）有要求的话需要预先设置好。在 ISE 的高级版本中，XST 已经支持 Verilog 和 VHDL 混合语言代码输入，XST 的输出文件是 NGC 网表。

7. 综合后仿真

综合实现后要进行综合后仿真，检验综合后功能是否符合设计要求，如果不符合要求要判断是否是原理图设计的问题，进一步对代码进行优化，直到符合设计要求为止。

8. 设计用户约束文件

用户约束文件主要包括时序约束文件和管脚约束文件，时序约束可以在 ISE 自带的 Constraints Editor 下完成，管脚约束是在 PACE（约束编辑器）下完成的。约束设计完毕生成 UCF 文件。

9. 实现

所谓实现（Implement）是将综合输出的逻辑网表翻译成所选器件的底层模块与硬件原语，将设计映射到器件结构上，进行布局布线，达到在选定器件上实现设计的目的。实现主要分为 3 个步骤：翻译（Translate）逻辑网表、映射（Map）到器件单元与布局布线（Place&Route）。

10. 布局布线后仿真

所谓布局布线仿真，是将 Xilinx 布线所产生的延迟加反标到电路后进行仿真。如果采用 ModelSim 仿真，在 ISE 下提供了 4 个仿真级别：Simulate Behavioral Model（行为仿真）、Simulate Post-Translate Verilog Model（翻译后仿真）、Simulate Post-Map Verilog Model（映射后仿真）、Simulate Post-Place&Route Verilog Model（布局布线后仿真）。

布局布线后生成的仿真时延文件包含的时延信息最全，不仅包含门延时，还包括实际布线延时，所以布线后仿真最准确，能较好地反映芯片的实际工作情况。

11. 下载

把 Xilinx 布线后产生的结果转换成配置文件后置入 FPGA 中。下载成功后就可以测试实际电路了。如果需要脱机配置，则必须将配置文件写入外置存储器中。下载时需要将器件用 JTAG 线与 PC 机连接完成下载。其实上面的 10 个步骤并不是一定要按部就班，这取决于设计者的熟练程度和设计水平。例如测试台的编写、前仿真、后仿真并不是必须的。但为了保证设计的正确性和节约查错所耗的时间，推荐设计者一步一步操作，这样能够及时发现错误及时更正。

ISE 软件一个特点就是具有良好的开发界面，新建一个工程后，在“Processes for Source”对话框中即罗列出了 ISE 下 FPGA 设计流程中的各个环节，如图 6.8 所示，读者可将图中标注的内容与上面介绍的 FPGA 的设计流程中的各个步骤相对应，可以看出，界面上列出了设计流程的各个步骤，设计时只需要按照顺序依次实现就可以了，操作很简便。

由于 ISE 提供的集成开发环境非常好，当然也可以在各自成上述步骤。

例如在 Synplify 中建工程、输入文件和 ModelSim 中新建工程并做仿真，用 Manager（这个程序在 ISE 中是找不到过命令行方式调用该程序使其在后台运目录下找到它），再运行 JTAG 编程器下本节对 ISE 的 FPGA 设计流程做了一个可对照 2.6 节加深对这一流程的理解。

6.4 创建设计工程

本节将重点讲述如何在 ISE 下创建一个设计，第一步要做的就是新建一个工程有以下几个步骤。

(1) 打开 Project Navigator，启动 ISE ISE 的启动请参见 6.2 节。

(2) 选择“File”/“New Project”菜单对话框。

会弹出如图 6.9 的对话框。

如图 6.9 所示，新建工程时需要设置工程名称和新建工程的路径，还要设置顶层模块的类型，具体各个类型解释如表 6.1 所示。

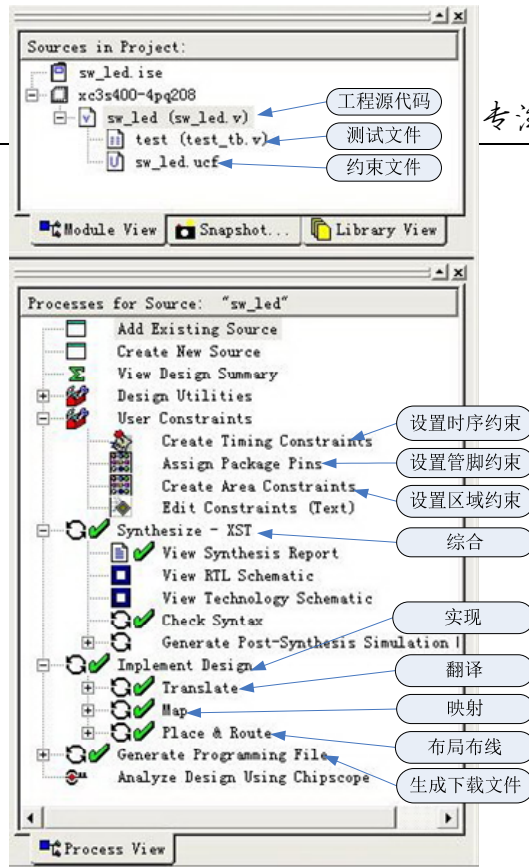


图 6.8 设计流程在 ISE 界面下的体现

用，故推荐从 ISE 的软件环境中完

综合，在 Xilinx 的 Design 的，因为 ISE 是通，但可以在安装载。

简单的介绍，读者

新的工程。要完成工程。具体创建一

集成环境。

项，启动新建工程

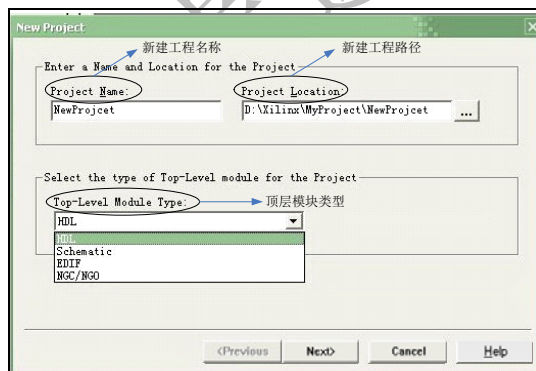


图 6.9 新建工程对话框

表 6.1 顶层模块类型说明

顶层模块类型	类型说明
HDL	硬件描述语言（Verilog 或 VHDL），用描述语言将各底层模块连接起来
Schematic	原理图，顶层模块可以用原理图将各底层模块连接起来，比较直观
EDIF	工业标准网表格式
NGC/NGO	综合后输出的文件格式，可以直接被 NGBuild 读取

(3) 设置工程属性。

启动新建工程对话框后，单击“下一步”按钮进入工程属性对话框设置，如图 6.10 所示。

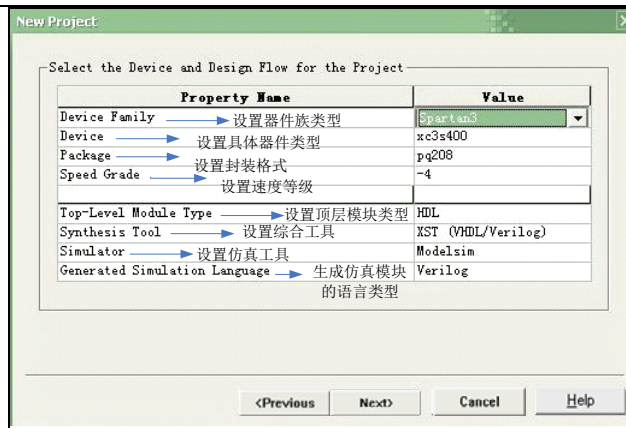


图 6.10 设置工程属性对话框

如图 6.10 所示，需要设置如表 6.2 所示内容。

表 6.2 新建工程属性说明

设置选项	设置内容
Device Family	设置 FPGA 是哪一系列的，如 Spartan3、Spartan3E、Virtex 等
Device	设置 FPGA 的具体型号，每个系列的 FPGA 下都有很多型号，要根据实际工程中应用的 FPGA 进行选择
Package	设置 FPGA 的封装格式，如 PQ208、FG456 等，对不同的封装格式，用户在约束引脚时会有所不同
Speed Grade	设置速度等级，如-4、-5、-6，数字越大速度越快
Top-Level Module type	设置顶层模块的类型
Synthesis Tool	设置设计中采用的综合工具，可以是 Xilinx 自带的 XST，如果安装了第三方工具，也可以选择第三方综合工具，如 Synplify/Synplify Pro 等
Simulator	设置设计中采用的仿真工具，可以是 ISE 自带的 ISE simulator，如果安装了第三方工具 ModelSim，也可以设置为 ModelSim
Generated Simulation Language	如果采用 ISE 自带的仿真工具 ISE Simulator，利用 HDL Bencher 可以在图形界面下编辑测试波形，直接生成测试激励文件，这里是设置生成测试激励文件的语言类型

(4) 为工程新建资源。

设置完工程属性后，单击“下一步”按钮，出现为工程新建资源的对话框，如图 6.11 所示。

新建工程时可以直接为新建的工程新建资源，单击“New Source”按钮会弹出新建资源的对话框，如图 6.12 所示。

这一步在新建工程时并不是必须的，如果在创建新的工程时没有为工程新建资源，可以在以后设计中再新建。如果工程创建完毕后需要新建资源，可以选择“Project” / “New Source”选项，也会弹出如图 6.12 所示的新建资源的对话框。

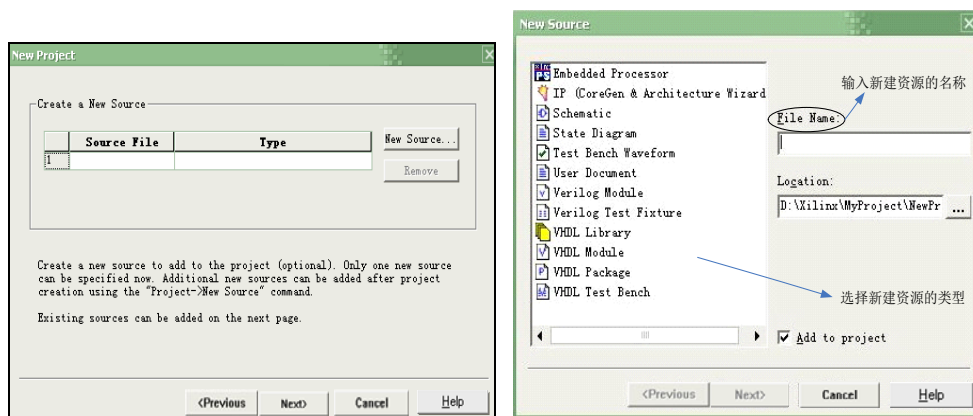


图 6.11 为新建工程新建资源对话框

图 6.12 新建资源对话框

具体新建资源的类型说明如表 6.3 所示。

表 6.3 新建资源类型说明

新建资源类型	类型说明	ISE 中对应的处理工具
Embedded Processor	嵌入式处理器	需要安装 Xilinx EDK 工具
IP (Corgen&Architecture Wizard)	IP 核	IP 核生成器 (Core Generator)
Schematic	原理图	原理图生成器 (ECS)
State Diagram	状态转移图	状态图编辑器 (StateCAD)
Test Bench Waveform	测试激励波形	测试激励生成器 HDL Bencher
User Document	工程说明文件	文本编辑器
Verilog Module	Verilog 源代码	HDL 语言编辑器 (HDL Editor)
Verilog Test Fixture	Verilog 测试激励	HDL 语言编辑器 (HDL Editor)
VHDL Library	VHDL 库	HDL 语言编辑器 (HDL Editor)
VHDL Module	VHDL 源代码	HDL 语言编辑器 (HDL Editor)
VHDL Package	VHDL 包	HDL 语言编辑器 (HDL Editor)
VHDL Test Bench	VHDL 测试激励	HDL 语言编辑器 (HDL Editor)

(5) 为工程添加现有资源。

单击“下一步”按钮，出现如图 6.13 所示对话框。如果工程的源代码已经编辑好，可以单击“Add Source”按钮为新建的工程添加资源，这一步与上一步一样，都不是必需的。如果工程已经创建完毕，可以通过选择“Project”/“Add Source”选项为工程添加资源。

(6) 单击“完成”按钮，创建工程。

单击“下一步”按钮，会出现一个对话框，显示工程的相关信息，如图 6.14 所示，如有设置错误可单击“上一步”按钮进行修改。上述各步骤均设置好后，单击“完成”按钮就可以创建工程了。

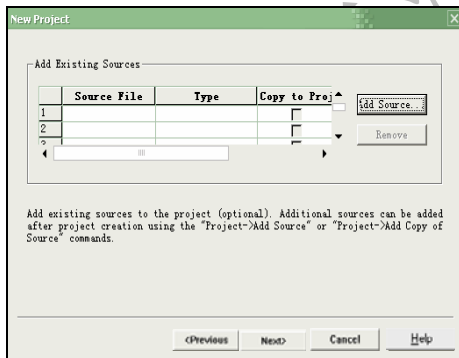


图 6.13 为新建工程添加资源对话框

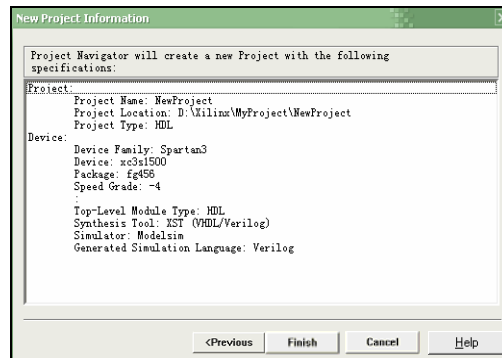


图 6.14 工程信息对话框

按上述步骤就可以完成一个新工程的创建。新工程创建完毕后，就可以通过新建或添加现有文件为工程添加资源，按照 ISE 的 FPGA 设计流程进行设计。

工程创建完毕后，如果需要修改工程的属性可以右键单击 图标。选择“Properties”选项，如图 6.15 所示，可以对工程的属性进行修改。

如果想更改工程的名称，同样右键单击 图标，选择“Properties”选项，弹出如图 6.16 所示对话框，可以更改工程的名称。

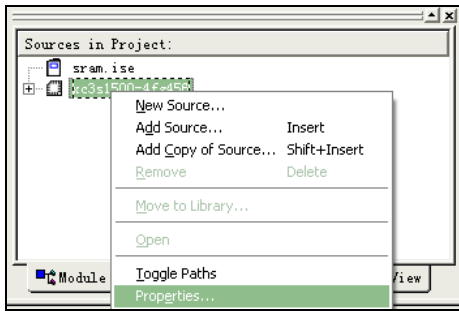


图 6.15 更改工程属性

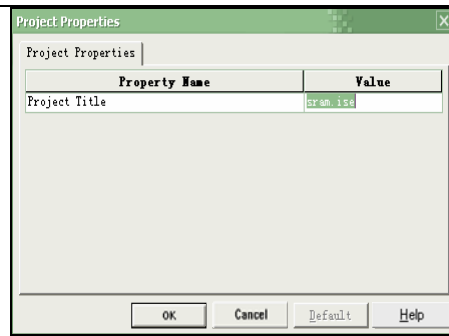


图 6.16 更改工程名称

6.5 编译与仿真设计工程

编写代码完成之后，一个很重要的工作就是验证代码功能的正确性，这就需要对代码进行编译与仿真。编译主要是为了检查代码是否存在语法错误，仿真主要为了验证代码实现的功能是否正确。

编译和仿真设计工程在整个设计中占有很重要的地位。因为代码功能不正确或代码的编写风格不好对后期的设计会有很大的影响，所以需要花很多时间在设计工程的仿真上。

在这一节中将通过一个具体的实例来介绍如何对编译工程代码以及如何使用 ISE 自带的仿真工具 ISE Simulator 进行仿真。

1. 编译工程代码

编译主要是为了检测代码是否存在语法错误。在 ISE 下，源代码的编写是在 HDL Editor 下完成的，但在 HDL Editor 下没有专门用于编译代码的选项。不过在 HDL Editor 下完成代码的编写后，单击“保存”按钮，HDL Editor 就会自动对代码进行编译。如果代码存在语法错误，就会在信息显示窗中显示出来，用户可以根据显示的提示，查找语法错误并修改。

如图 6.17 所示为在输写代码时忘记分号，保存后就会有提示信息。

当不存在错误时，提示信息就不会出现“Warning”。ISE 下对于代码的编译功能并不是很强大，有很多错误是检测不出来的。例如在编写 Verilog 代码时，写 case 语句时漏写了 end case 语句，在 HDL Editor 下是检查不出来的。但这些错误在仿真或综合阶段是可以检测出来的，因此即使完成了编译没有错误，也一定要进行仿真，检查是否还存在其他的错误。

2. 仿真设计工程

这里以一个具体的实例来介绍 ISE 下自带的仿真工具 ISE Simulator 的使用，代码参见本书实例代码。该例程的主要功能是根据拨码开关（sw）输入的值在数码管（seg_led）上显示相应的数值。

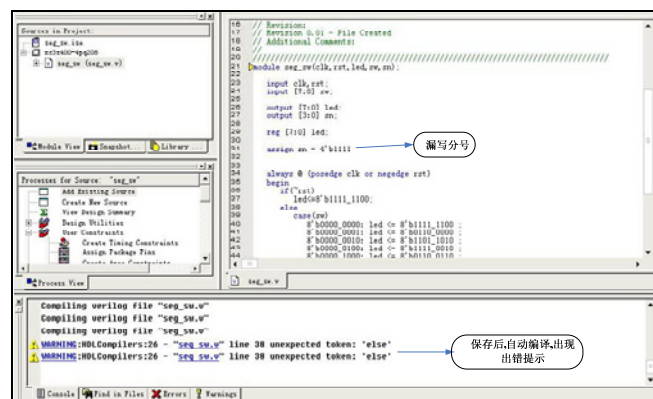


图 6.17 编译后的出错提示

ISE Simulator 的使用主要是借助于 ISE 的辅助设计工具 HDL Bencher（测试激励生成器）来完成的。用户将 VHDL 源代码、Verilog 源代码或 ECS 原理图等设计输入导入工程后，用户可以在图形界面下编辑测试波形，HDL Bencher 可以根据用户编辑的测试波形自动生成测试激励文件，然后调用 ISE 中的 ISE Simulator 进行仿真。可见使用 ISE Simulator 进行仿真主要分两步。

- (1) 调用 HDL Bencher，编辑测试波形，生成测试激励文件。
- (2) 调用 ISE Simulator 对工程文件进行功能仿真和时序仿真。

下面对实例做详细的介绍，具体步骤如下。

- (1) 启动 HDL Bencher。

首先打开实例工程，选择“Project”/“New Source”，弹出新建资源的对话框，选择“Test Bench Waveform”，输入测试激励波形文件名，如图 6.18 所示。

单击“下一步”按钮为测试激励文件选择源文件，如图 6.19 所示，选择要测试的源代码，单击“下一步”按钮。

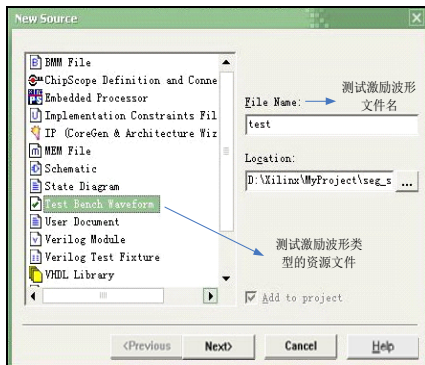


图 6.18 新建测试激励波形文件

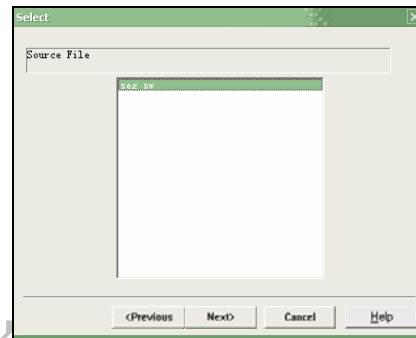


图 6.19 为被测试的源文件对话框

单击“完成”按钮确认新建资源信息，HDL Bencher 会自动启动。

- (2) 波形编辑。

HDL Bencher 启动后首先出现如图 6.20 所示的时钟设置对话框。

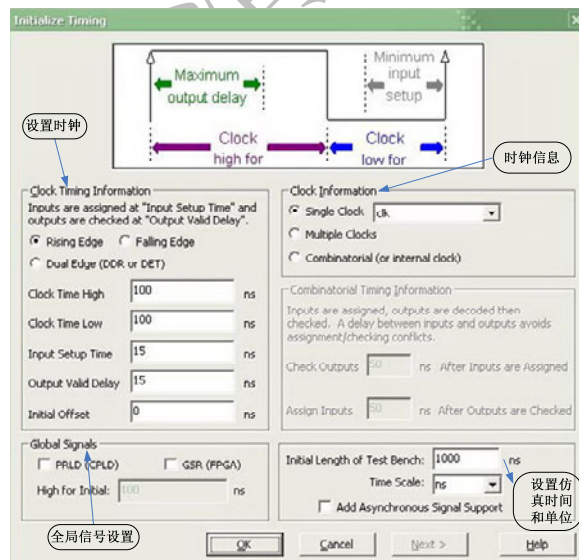


图 6.20 时钟设置对话框

如图 6.20 所示，时钟设置包括：时钟的高电平持续时间（Clock Time High）、低电平持续时间（Clock Time Low）、建立时间（Input Setup Time）和保持时间（Output Valid Delay），系统为单时钟（Single Clock）系统、多时钟（Multiple Clocks）系统以及仿真时间和单位。这里设置的仿真时间表示仿真将持续多长时间后自动停止。

如果设计存在异步时序（Asynchronous Signal Support），还要对异步时钟做相应的设置。这个设计中，只有同步时序，所以不选，与上表设置不一致之处请用户自行修改。设置完毕单击“OK”按钮后系统会自

动生成时钟的波形，如图 6.21 所示。

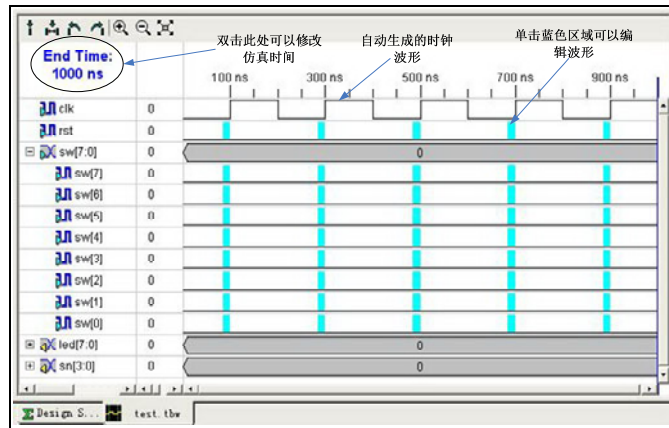


图 6.21 波形编辑界面

如图 6.21 所示，设置好时钟后，系统会自动生成时钟波形。在如图 6.18 所示的对话框中设置的“Initial Length of Test Bench”为 1000ns，这里可以看到仿真在进行了 1000ns 后就自动停止了，双击“End Time”，会弹出如图 6.22 所示的对话框，可以对仿真时间进行修改。

在设置好时钟频率、时钟建立时间和保持时间后，如果要修改，选择“Test Bench”/“Rescale Timing”会弹出如图 6.23 所示对话框，可以对时钟设置进行修改。

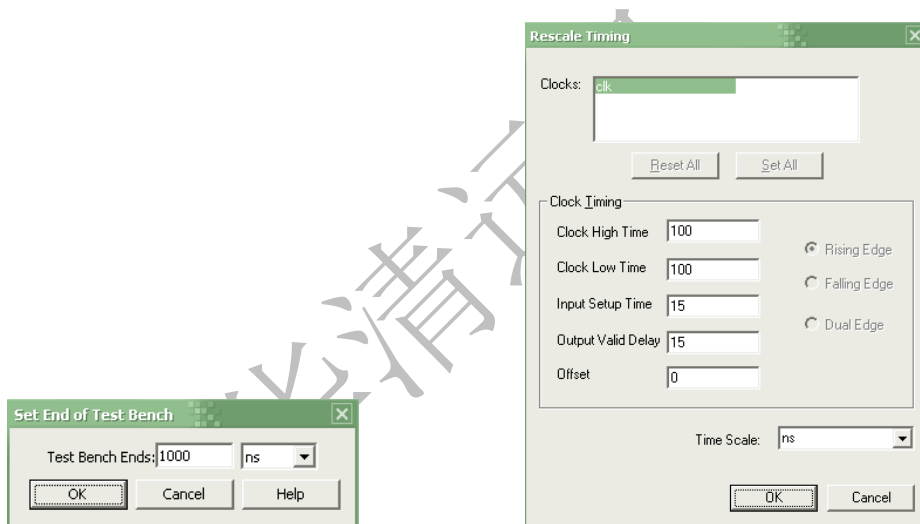


图 6.22 修改仿真时间对话框

图 6.23 修改时钟设置对话框

这此设计中除时钟信号外，rst 和 sw[7:0]为输入信号，需要对 rst 和 sw 的波形进行编辑，编辑方法也比较简单，单击图中的蓝色区域就可以改变波形，根据仿真需要可任意设置波形，在这里设置仿真波形如图 6.24 所示。

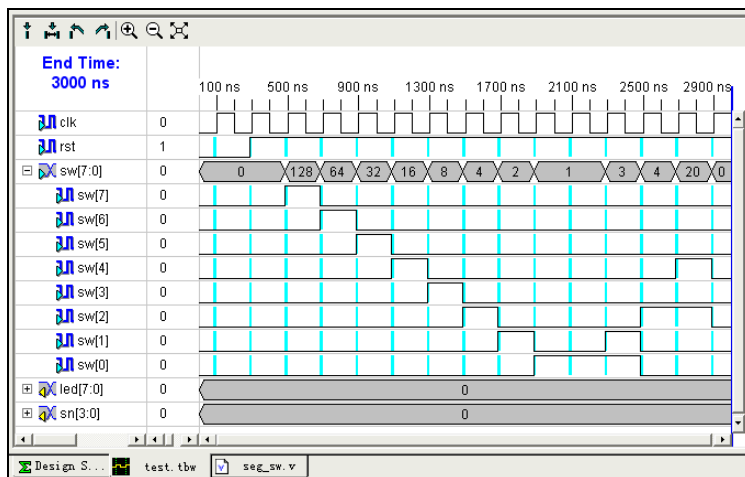


图 6.24 编辑波形图

波形图编辑完毕后，单击“保存”按钮，系统会为工程自动添加“test.tbw”文件。选中此文件，在当前资源操作（Process for Source）视窗中，可以看到“View Generated Test Bench As HDL”选项，如图 6.25 所示。

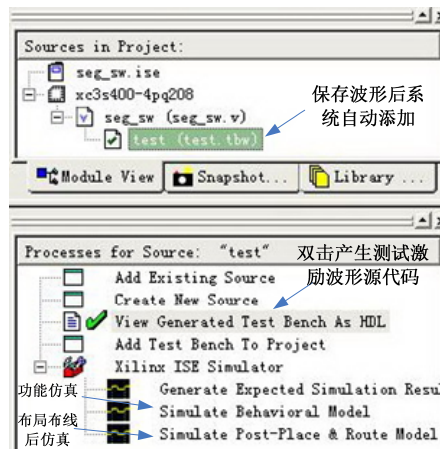


图 6.25 生成测试激励波形文件后的资源视窗

双击此选项，系统就会根据设置的波形自动生成测试激励文件的源代码，如图 6.26 所示。

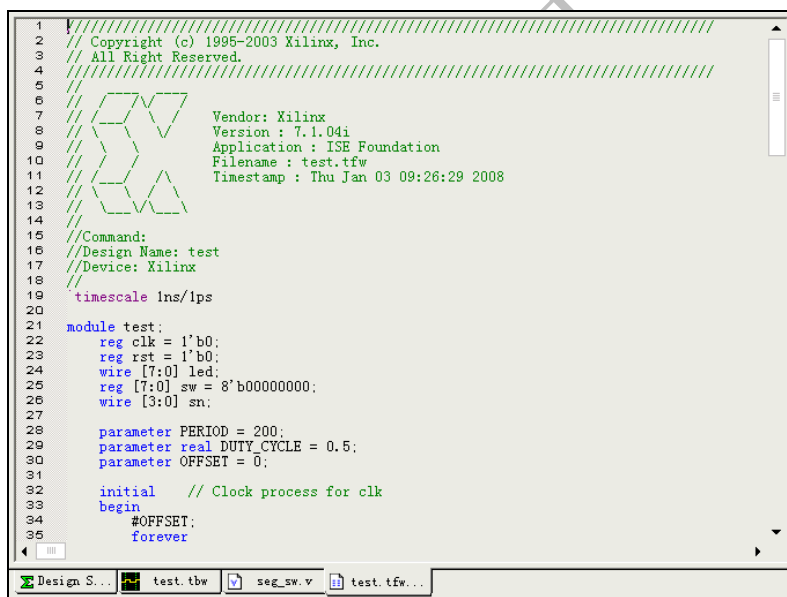


图 6.26 系统根据编辑的波形生成的测试代码

(3) 对设计工程进行功能仿真。

ISE Simulator 中提供了两种级别的仿真：功能仿真和布局布线后仿真，功能仿真可以验证代码功能的正确性，布局布线后生成的仿真时延文件包含的时延信息最全，不仅包含门延时，还包括实际布线延时，所以布线后仿真最准确，能较好地反映芯片的实际工作情况。

按上述步骤用 HDL Bencher 生成测试激励波形文件后，就可对设计工程进行仿真了，首先验证设计功能的正确性，先对工程进行功能仿真。

功能仿真时在工程资源（Sources in Project）视窗中选择波形文件（test.tbw），在当前资源操作（Process for Source）视窗中，双击 Simulate Behavioral Model（如图 6.25 所示），得到功能仿真结果如图 6.27 所示。

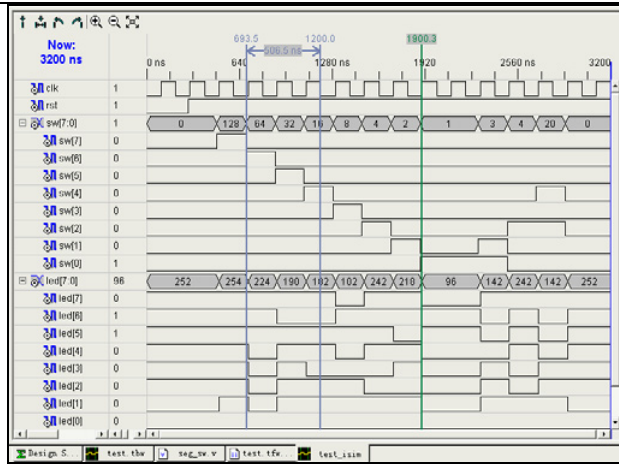


图 6.27 功能仿真结果

观察波形时，左上角有几个按键。单击可放置一条线，用于观察某一特定时刻的值。用于测量时间间隔。用于寻找前一个或下一个信号上升沿，主要用于在测量时间间隔时，定位测量直线。

另外，ISE Simulator 在仿真过程中会自动判断仿真过程中是否有错误发生，如果有错误发生，会在 TX_ERROR 下显示出来，TX_ERROR 会自动统计错误的个数并显示，并且在 SimConsole 信号视窗中显示正确的数值和实际仿真得到的数据，用户可根据这些信息，对源代码进行修改。

(4) 布局布线后仿真。

功能仿真测试功能正确后，就可以按照 ISE 下 FPGA 的设计流程对工程进行综合及布局布线。布局布线后，就可以对工程进行布局布线后仿真。

布局布线仿真时在工程资源 (Sources in Project) 视窗中选择波形文件 (test.tbw)。在当前资源操作 (Process for Source) 视窗中，双击 Simulate Post-Place&Route Model (如图 6.25 所示)，得到布局布线后仿真结果如图 6.28 所示。

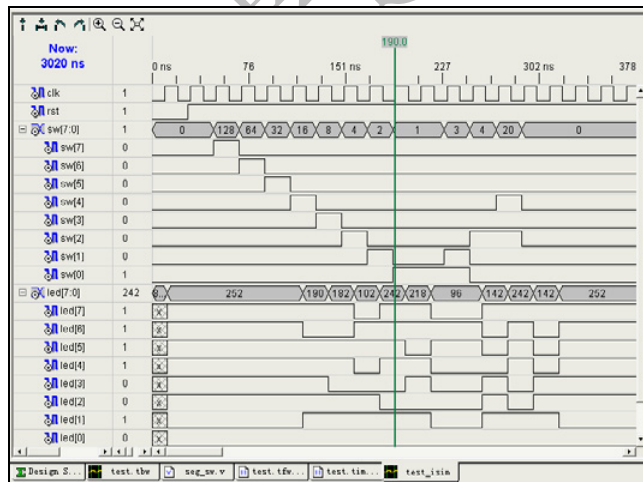


图 6.28 布局布线后仿真结果

如图 6.28 所示，布局布线后仿真结果与功能仿真结果相比，有了较大的延迟，这主要与代码中组合逻辑设置有关，在布局布线仿真后，由于加入了延时信息可能会造成功能的不正确，用户可以通过改写代码或添加相应的时序约束，来优化设计，布局布线后仿真在工程设计中占有非常重要的地位，需要花很长时间来不断地仿真和优化设计。

3. 小结

本节通过一个具体的实例介绍了 ISE 下自带仿真工具 ISE Simulator 的使用。ISE Simulator 的特点是可以通 HDL Bencher 在图形界面下编辑波形，不需要用户编写测试代码，使用方便。用户也可以在 HDL Bencher 下完成波形编辑生成测试代码后，调用 ModelSim 仿真。

通过这个例子，希望用户能够掌握 ISE Simulator 的基本用法，熟悉仿真的基本流程。仿真在 FPGA 的设计当中占有重要位置，可以说设计者的大部分工作都是在做仿真，只有仿真结果达到要求了，才会映射到实际电路，然后在线调试。

在这里推荐大家自己编写测试激励文件。因为在一些复杂的设计当中，使用图形界面编辑激励波形是很难满足设计要求的，无法对工程进行完整的测试。

另外，ModelSim 是一款功能很强大的仿真软件，支持混合仿真。在 ISE 下为 ModelSim 预留了接口，使用也很方便，目前应用比较广泛。在初始学习阶段可以使用 ISE Simulator 作一些简单的仿真，熟悉 FPGA 的设计流程，等熟练之后，建议在 ModelSim 下完成仿真。

6.6 增量式设计 (Incremental Design) 技巧

本节将对 ISE 下增量式设计做一个全面的介绍。FPGA 作为一种现场可编程逻辑器件，其现场可重编程特性能够提高调试速度。每次硬件工程师可以很方便地改变设计，重新进行综合、实现、布局布线，并对整个设计重新编程。

然而当设计算法比较复杂时，每一次综合、实现、布局布线需要花很长的时间。即使仅仅改变设计中的一点点，也会使综合编译的时间成倍增加。而且更为麻烦的是如果整个工程的运行频率很高，对时序的要求也很严格，这样重新布线往往会造成整个时序错乱。

运用增量式设计可以有效地解决这一问题。一方面大大节约综合、布局布线的耗时，另一方面可以继承前一设计中已有的成果，是一种比较常用的设计流程。

6.6.1 增量式设计的必要性

增量式设计 (Incremental Design) 方法是一种能在小范围改动情况下节约综合、实现时间并继承以往设计成果的设计手段。作为一个流程，增量设计能够极大地减小布局布线时间，并且当对一个近似完整的设计作小的变动，可以保持整个系统的性能。

在增量设计中每一个逻辑分组在 Xilinx 的 FPGA 里受到约束以使之只占有自己的空间。在设计中，对对其中之一的逻辑分组做改动时，一个增量设计流程可以确保未做改动的逻辑分组在进行综合输出时不变化。接着布线工具对改动了的逻辑分组重新进行布局布线，而未改动的逻辑分组则继续以前的布局布线结果，这使得整个设计的布局布线时间得以削减。

增量式设计对一处复杂的设计来说是非常必要的，主要是因为增量式设计有以下两个方面的优点。

1. 减小综合、布局布线的耗时

当仅对大型设计工程的局部进行改动时，增量设计流程仅仅改动的部分重新编译，如果改动模块的接口设计恰当，将不会影响其余部分的综合与实现结果，布局布线时也只对改动部分重新布线，未改动的部分保持不变，从而节约了整个编译、布局布线与优化的耗时。

2. 能够很好地继承未修改区域的实现成果

这一点对于对时序要求很严格的设计来说是很有用的。如果一个设计经过多次调试，附加合适的约束，设置恰当的参数达到了最佳实现成果。但是因为对某个细节进行了修改，就需要全部重新综合、布局布线，这样可能前面所做的精心调整工作都无效了。

通过增量式设计，可以解决这一问题。对于已达到设计要求的部分将其保持不变，仅对修改的部分重新编译、布局布线，从而保证在最大程度上继承以往的实现结果。

6.6.2 增量设计流程

具体的增量设计流程如图 6.29 所示。
增量设计的流程可归纳如下。

1. 创建逻辑分组 (Create Logic Group)

在增量设计中为了实现减小综合、布局布线耗时，极改区域的成果，必须要求将设计分成多个逻辑分组。分配一定的逻辑区域，当某一逻辑分组的内容发生改变在该逻辑分组分配的逻辑区域内对其进行重新综合和影响到其他的逻辑分组。

所谓“逻辑分组”，是惟一的逻辑层次中的若干逻辑实层逻辑层次中每个子模块即为一个逻辑分组。在代码“module (Verilog)”和“entity (VHDL)”设计的子设计中往往将实现的不同功能设置为不同的模块，然中实例化所有这些不同功能的模块，从而实现一个完不同功能的模块就可以看作是不同的逻辑分组。

在进行逻辑分组时，需要考虑以下因素。

(1) 设计中所有逻辑除了 IOB 和时钟逻辑，都应该包含在逻辑分组当中。

(2) 顶层模块不应该包含复杂逻辑，仅仅包含一些 I/O 定义、时钟分配逻辑和所有子模块的实例化，直正的功能实体用子模块的逻辑描述。增量设计方法希望将所有的逻辑实体分割到子模块中去，而顶层模块不含任何实际的逻辑功能，以便于做相应的区域约束。

顶层包含实际逻辑功能的缺点在于：当顶层改变时，相关的 Logic Group 的接口将发生变化，从而影响 Logic Group 的结构，在做编译和布局布线时，会影响增量设计的效能。

(3) 逻辑模块分组必须以寄存器输出，即用寄存器分割模块。这一点其实不仅仅是增量设计的需求，也是合理划分模块的一个基本要求。

如果采用同步时序方式设计电路，用寄存器分割逻辑模块，模块间的接口尽量简单，则时序优化路径集中在同一模块内部而不是模块之间的边界上。这样能够使综合器完整地掌握需要时序优化的路径，从而避免了因一个模块内部改变而通过边界影响到其他模块的时序这种不利于增量设计的情况发生。

(4) 每个逻辑分组为其附加区域分组约束。

2. 增量综合 (Incremental Synthesis)

所谓增量综合是指只有改变的部分重新综合，而对未改变部分保持原有的综合结果的一种综合技术。传统的综合技术即使有微小的改动，也会对整个设计重新综合。

如果要实现增量综合必须对综合工具做相应的设置。在这里主要讲述 ISE 自带综合工具 XST 是如何实现增量综合的，对于其他综合工具如：Synplify/Synplify Pro 和 Leonardo Spectrum 综合工具，在这里不做详细介绍。

XST 支持单一工程的模块级增量综合 (BLSI)。实现的方法为在 XST 的约束文件 (扩展名为 xcf) 中附加逻辑分组约束，从而告知 XST Logic Group 的边界。

XST 在综合时，所有的编译与优化都不超越用户在 XCF 文件中约定的 Logic Group 的边界，以达到在细微修改后仅仅对 Logic Group 内部进行重新综合的目的。这样一个逻辑分组 HDL 源代码的改变就不会对其他逻辑分组造成影响。

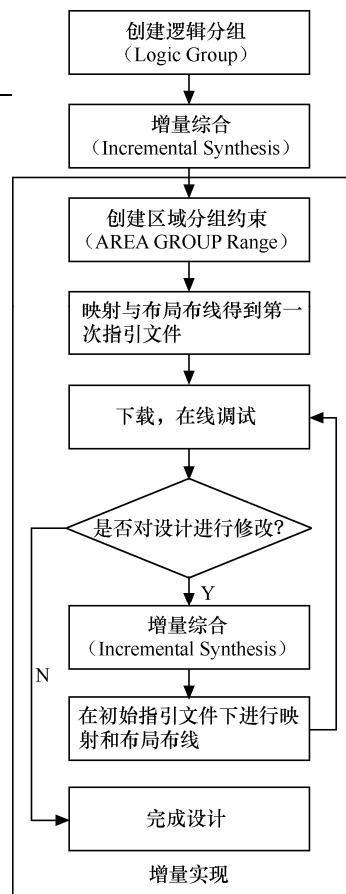


图 6.29 增量设计流程

大程度地继承未修每个逻辑分组应该变时，增量设计可以布局布线，而不会影

体的划分。比如在顶中即为顶层中模块的实体。在一个后在一个顶层模块整的功能，那么这些

对于 VHDL 来说，XST 可以自动检测出哪一个逻辑分组的内容发生了改变。但对于 Verilog 来说，必须应用“resynthesize”属性。例如下面就是一个逻辑分组 A 的 Verilog 源代码发生细微的改变时的 XCF 文件范例。

```
MODEL "top" incremental_synthesis = yes; //对 top 使能增量设计
MODEL "A" incremental_synthesis = yes; //对 A 使能增量设计
MODEL "B" incremental_synthesis = yes; //对 B 使能增量设计
MODEL "C" incremental_synthesis = yes; //对 C 使能增量设计
MODEL "top" resynthesize = no; //通知综合工具哪个模块发生改变
MODEL "A" resynthesize = yes; //no 为未改变，yes 为已改变
MODEL "B" resynthesize = no;
MODEL "C" resynthesize= no;
```

对于 Verilog 设计工程，当某一逻辑分组发生改变时，要为其附加相应的综合约束，才能实现增量综合。另外需要注意的问题是，增量综合是在保留结构层次模式（Keep Hierarchy）下完成的，在进行增量综合时，在“Processes for Source”中选择“Synthesize-XST”，单击右键，设置综合属性如图 6.30 所示。如图 6.30 所示，要完成增量综合，要完成两种设置，选择“Keep Hierarchy”设置为 YES 或 Soft，选择“Synthesis Constraints File”设置综合约束文件的路径。

增量综合后，需要检查综合报告。



图 6.30 增量综合属性设置对话框

3. 创建区域分组约束

创建区域分组约束是增量设计中最重要的一步。区域分组约束做得不好会增加综合、布局布线耗时，甚至有可能导致布局布线无法完成。区域分组约束的创建是利用 ISE 下的 PACE 工具，在 PACE 下完成区域分组约束后，ISE 会自动将其写入 UCF 文件中。

为增量设计创建合理的区域分组约束要遵循以下原则。

- 所有 I/O 引脚位置必须锁定。
- 将与 I/O 端口联系密切的区域分组布置在相应 I/O 端口的附近。
- 区域分组约束的范围不应该重叠。
- 尽量保证每个区域的资源利用率基本一致，避免出现某一区域利用率达到 99%，而另一区域的利用率只有 10% 的情况。
- 如果一个区域分组中包含 FPGA 的多种资源，如：Slice、Block RAM、TBUF、Multipliers 等，这时很有必要将不同的资源设置在不同的区域，然后将几个不同位置区域拼合为一个区域分组。可以使用 PACE、Floorplanner 等工具完成，也可以在约束文件中使用如下约束：

```
INST Logic_Group_A AREA_GROUP = AG_Logic_Group_A ; //AREA 分组
```

```

AREA_GROUP "AG_Logic_Group_A" RANGE = SLICE_X0Y20:SLICE_X20Y30 ;//SLICE 约束
AREA_GROUP "AG_Logic_Group_A" RANGE = RAMB16_X0Y2:RAMB16_X0Y2 ;//RAM 约束
AREA_GROUP "AG_Logic_Group_A" RANGE = MULT18X18_X0Y1:MULT18X18_X0Y1; //MULT 约束
AREA_GROUP "AG_Logic_Group_A" RANGE = TBUF_X0Y0:TBUF_X1Y0; //TBUF 约束
    
```

4. 增量实现 (Incremental Implement)

第 3 步的创建区域分组约束也可以作为增量实现的内容，增量实现的步骤如下。

- (1) 设置区域分组约束。
- (2) 映射、布局布线得到初始指引文件。
- (3) 下载调试，需要修改时，对细微细节进行修改，然后实现增量综合。
- (4) 增量综合后在初始指引文件下进行第二次映射与布局布线。
- (5) 下载调试，如仍有问题，重复步骤 (3) ~ 步骤 (5)，直到符合设计要求为止。

在增量实现过程中，必须要对布局布线的属性进行相关的设置。首先应该使能增量设计，具体设置如图 6.31 所示。在“Processes for Source”中选择“Implement Design”，单击右键，出现属性对话框。选择“Incremental Design Properties”，然后选择“Enable Incremental Design Flow”，使能增量设计。

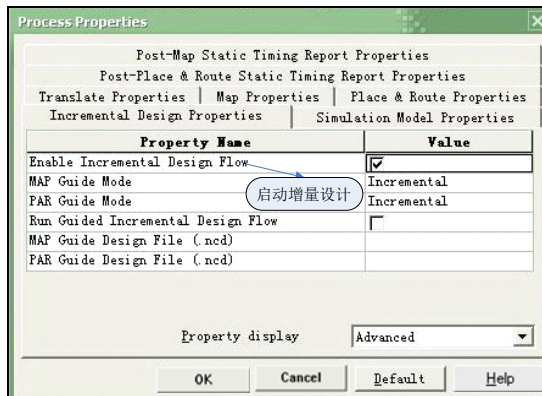


图 6.31 项目管理器中启动增量设计功能对话框

在增量设计中，如果是用 Synplify/Synplify Pro 等第三方综合工具，那么需要由综合工具生成向导文件指导实现工具进行实现流程。选择如图 6.31 中的“Run Guided Incremental Design Flow”选项，表示将由指定的向导文件来指导实现工具进行实现流程。

向导文件可提供 Map.ncd 文件和 Par.ncd 文件。其中，Map.ncd 文件包含了 slices、IOBs 等映射信息。Par.ncd 文件包含了布局和布线信息。实现工具根据这两个文件的信息来确定没有改变的模块和已经改变的模块，把没有改变的模块放进向导文件所记录的上次综合实现的位置。对布线器来说，该布线资源同时也将被保留，其布线关系也不会改变。

使用向导文件，要对实现具的属性做部分修改。将“Map Guide Mode”和“PAR Guide Mode”分别改成增量型 (Incremental)，如图 6.32 和图 6.33 所示。



图 6.32 使用指引文件指引映射

图 6.33 使用指引文件引导布局布线

进行了上述设置后，需要指定映射和布局布线的指引文件。

整个增量实现步骤完成后，需要检查映射报告（扩展名为 map）和布局布线报告（扩展名为 par），看是否真的完成了增量设计。

6.6.3 小结

本节对增量式设计方法的概念以及设计流程做了全面的介绍，希望读者能够掌握增量设计的基本流程，并应用到自己的工程设计当中。在 6.9 节中会通过一个具体的实例来说明增量设计的整个流程。

增量设计在大型的工程设计中是很有用的，可以为设计节约大量的时间。增量设计中一个难点在于分组区域约束的设置，这不但需要对整个工程有全面的把握而且要求对于 FPGA 器件的内部结构有一定的了解，这需要在实践中不断积累经验。

6.7 片上逻辑分析仪（ChipScope Pro）使用技巧

在 FPGA 的调试阶段，传统的方法在设计 FPGA 的 PCB 板时，保留一定数量的 FPGA 管脚作为测试管脚。在调试的时候将要测试的信号引到测试管脚，用逻辑分析仪观察内部信号。

这种方法存在很多弊端：一是逻辑分析仪价格高昂，每个公司拥有的数量有限，在研发期间往往供不应求，影响进度；二是 PCB 布线后测试脚的数量就确定了，不能灵活地增加，当测试脚不够用时会影响测试，测试管脚太多又影响 PCB 布局布线。

ChipScope Pro 是 ISE 下一款功能强大的在线调试工具。面对这些问题，ChipScope Pro 都可以有效地解决。

6.7.1 ChipScope Pro 概述

ChipScope Pro 是针对 Xilinx Virtex-II pro/ Virtex/ Virtex-II/ Virtex-EM/ Spartan-III/ Spartan-III 系列 FPGA 的在线片内信号分析工具。它的主要功能是通过 JTAG 口，在线实时读取 FPGA 的内部信号。

ChipScope Pro 的基本原理是利用 FPGA 中未使用的 BlockRam，根据用户设定的触发条件将信号实时地保存到这些 BlockRam 中，然后通过 JTAG 口传送到计算机，最后在计算机屏幕上显示出时序波形。ChipScope Pro 应用的框图如图 6.34 所示。

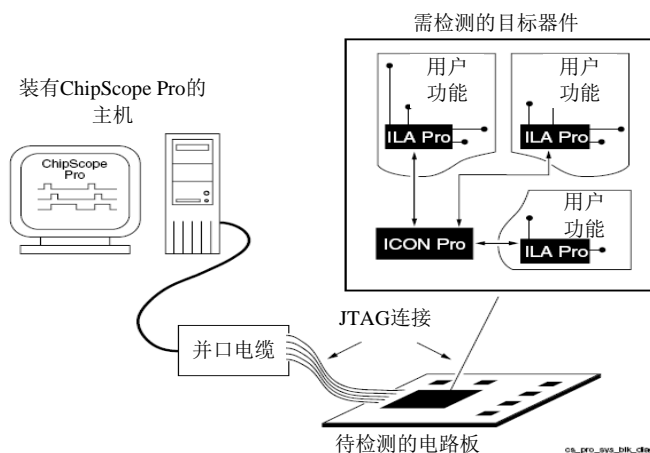


图 6.34 ChipScope Pro 应用框图

其中 ILA、ICON 是为了使用 ChipScope Pro 观察信号而插入的核。ChipScope Pro 工作时一般需要用户设计中实例化两种核：一是集成逻辑分析仪核（ILA core, Integrate Logic Analyzer core），该核主要用于提供触发和捕获的功能；二是集成控制核（ICON core, Integrated Contorlller core），负责 ILA core 和边界扫描端

口 (JTAG) 的通信。

一个 ICON core 可以连接 1~15 个 ILA core。ChipScope Pro 工作时, ILA core 根据用户设置的触发条件捕获数据, 然后在 ICON core 控制下, 通过边界扫描端口上传到计算机, 最后用 ChipScope Pro Analyzer 显示信号波形。

6.7.2 ChipScope Pro 设计流程

ChipScope Pro 工具箱中包含了 3 个工具: ChipScope Pro Core Generator、ChipScope Pro Core Inserter、ChipScope Pro Analyzer, 使用 ChipScope Pro 在线调试工具的 FPGA 设计流程如图 6.35 所示。

由上述流程可知, ChipScope Pro 有两种使用方法。
 第一种是由 ChipScope Pro Core Generator 根据设定条件生成在线逻辑分析仪 IP 核, 包括 core、ILA/ATC core 和 IBA/OPB core 在原 HDL 代码中实例化这些核, 然后综合、布局、下载配置文件, 就可以利用 Analyzer 设定的触发条件, 观察信号。
 第二种是原代码完成综合后, 由 Inserter 工具插入 ICON core 和 ILA 核, 然后综合、布局、下载配置文件, 就可以利用 Analyzer 设定的触发条件, 观察信号。
 第二种是原代码完成综合后, 由 Inserter 工具插入 ICON core 和 ILA 核, 然后综合、布局、下载配置文件, 就可以利用 Analyzer 设定的触发条件, 观察信号。
 第二种是原代码完成综合后, 由 Inserter 工具插入 ICON core 和 ILA 核, 然后综合、布局、下载配置文件, 就可以利用 Analyzer 设定的触发条件, 观察信号。
 第二种是原代码完成综合后, 由 Inserter 工具插入 ICON core 和 ILA 核, 然后综合、布局、下载配置文件, 就可以利用 Analyzer 设定的触发条件, 观察信号。

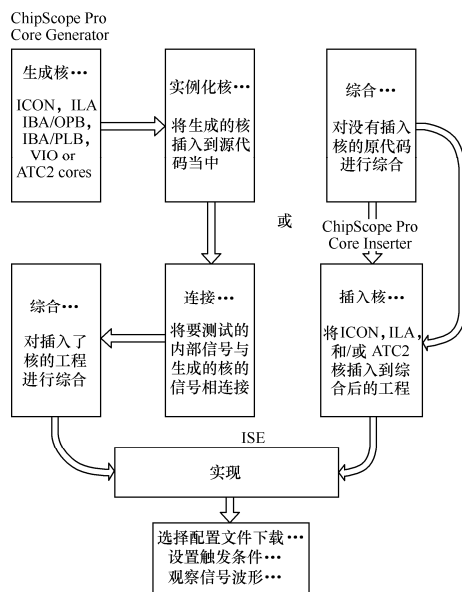


图 6.35 ChipScope Pro 设计流程

6.7.3 ChipScope Pro

Inserter 简介

ChipScope Pro Core Inserter 的启动有两种方式。

(1) 直接在 Windows 环境下运行“开始”/“程序”/“ChipScope Pro 8.2i”/“ChipScope Pro Core Inserter”命令。运行后即可得到 ChipScope Pro Core Inserter 的用户界面, 如图 6.36 所示。

(2) 可以通过新建资源的方法, 如图 6.37 所示。

新建 ChipScope Pro Inserter 资源后, 系统自动生成扩展名为 cdc 的文件。如图 6.38 所示, 双击扩展名为 cdc 的文件即可启动 ChipScope Pro Inserter 界面。需要注意的是, 在双击扩展名为 cdc 的文件时, 系统会先对该工程文件进行综合。综合完成后才会启动 ChipScope Pro Inserter。

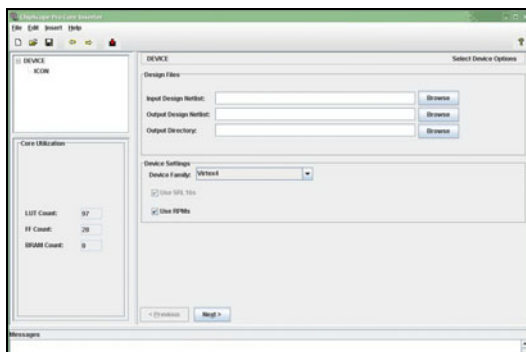


图 6.36 ChipScope Pro Core Inserter 用户界面

Core

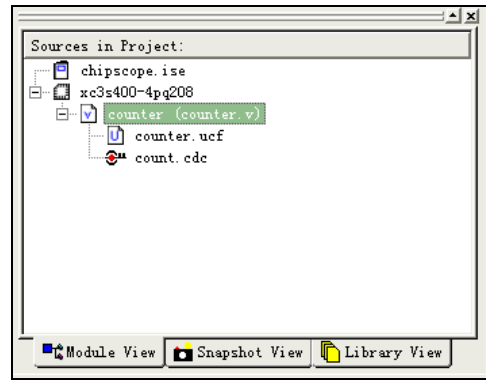
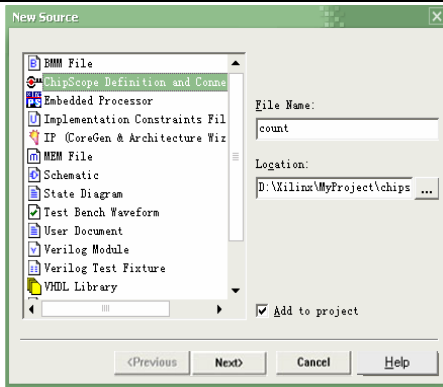
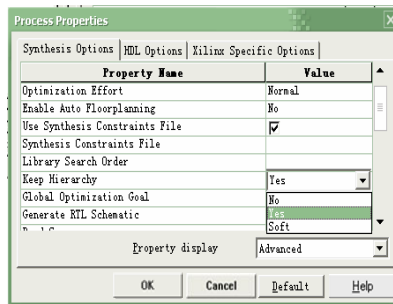


图 6.37 新建 ChipScope Pro Inserter 资源对话框 图 6.38 新建 ChipScope Pro Core Inserter 资源后的界面

这里在综合前必须先对综合属性进行设置，如图 6.39 所示，在综合属性对话框中对“synthesis options”/“keep Hierarchy”选项进行设置。设置“keep Hierarchy”为“Yes”或“Soft”。

双击扩展名为 cdc 的文件，系统完成综合后，ChipScope Pro Core Inserter。设计者通过 Core Inserter 对触发单元个数、触发宽度、触发采样时刻等参数进行设置。设置完毕后，在 ISE 下载配制文件，即可用 ChipScope Pro Analyzer 下面对 ChipScope Pro Core Inserter 的各项设



会自动启动 ChipScope Pro 条件、存储深度、下完成布局布线，进行观测。置做详细的说明。

1. 用户界面

图 6.39 综合属性设置对话框

启动 ChipScope Pro Core Inserter 后，显示如图 6.40 所示的界面。在“Input Design Netlist”文本框中设置输入设计网表的路径。设置好后，“Output Design Netlist”和“Output Directory”会自动生成，设计者也可自己指定。

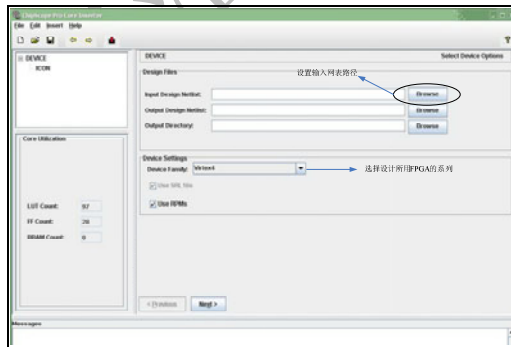


图 6.40 ChipScope Pro Core Inserter 用户界面

如果是通过新建资源的方法启动 ChipScope Pro Core Inserter，这几项显示为灰色，无需设计者设置，系统会自动找到设计网表文件。在“Device Family”下拉列表中选取设计所用的 FPGA 后，就可单击“Next”按钮，进入“Select Integrated Controller Options”对话框，如图 6.41 所示。

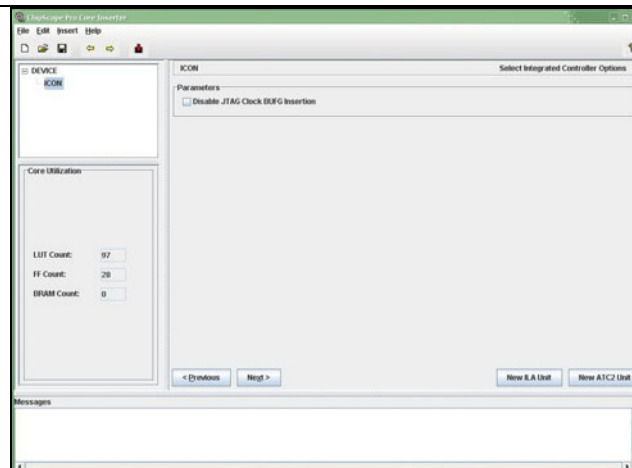


图 6.41 “Select Integrated Controller Options”对话框

2. “Select Integrated Controller Options”对话框设置

在如图 6.40 所示的“Select Device Options”对话框中，可以指定是否禁止在 JTAG 时钟上插入 BUF8。如果选中此项，JTAG 时钟将使用普通布线资源，而不是全局时钟布线。这样会在 JTAG 时钟线上产生较大的布线延时。因此在全局时钟资源足够用的情况下，应该尽量使 JTAG 时钟使用 BUF8 资源。即使由于全局时钟资源不够而不得不禁用 BUF8 时，也最好附加相应约束，使延迟抖动尽量小。推荐设计者在使用时不选此项。

单击“Next”按钮，进入“Select Integrated Logic Analyzer Options”对话框，如图 6.42 所示。

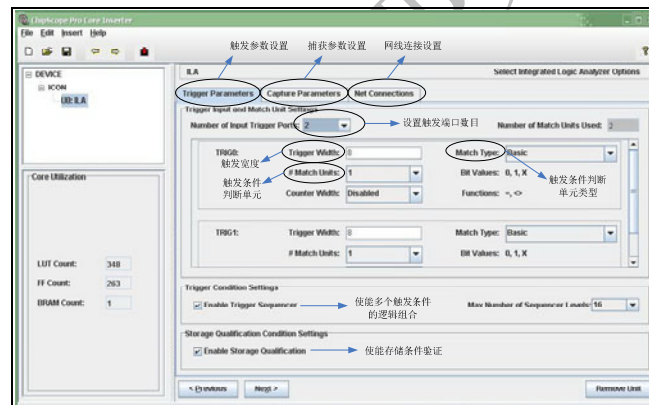


图 6.42 “Select Integrated Logic Analyzer Options” ILA 对话框

3. “Select Integrated Logic Analyzer Options”对话框设置

如图 6.42 所示，可以看到在“Select Integrated Logic Analyzer Options” ILA 对话框下有 3 个选项卡，可对触发参数、捕获参数、网线连接进行设置。

“Trigger Parameters”选项卡可对触发端口数目，每个触发端口的宽度、触发条件判断单元、触发条件判断单元的个数和类型等进行设置。

(1) 触发端口数目。

在设计中可以根据需要设置多个触发端口，每个 ILA Core 最多可以有 16 个输入触发端口，每个触发端口下又可设置多个触发条件判断单元，但各个触发端口包含的触发条件判断单元数量之和不能大于 16。

(2) 触发端口设置。

一个完整的触发端口设置包括：触发宽度、触发条件判断单元个数及类型的设置。触发宽度是指触发端口包含信号线的个数。通过触发条件判断单元进行判断，当信号线上的信号满足设定的条件时，ChipScope Pro 就可将其捕获并存储在 BlockRam 中，用于在 ChipScope Pro Analyzer 中显示波形。

对触发条件可以设置个数和类型。当有多个触发条件时，可以将触发条件设置为几个触发条件的逻辑组合。触发条件判断单元实际为比较器，其类型可以有以下几种，如表 6.4 所示。

表 6.4 触发条件判断单元的类型

类 型	数值类型	匹 配 功 能	Bit/Slice	说 明
Basic	0、1、X	=、<>	8	用于一般信号比较，是一种节约资源的类型
Basic(w/trans)	0、1、X、R、F、B	=、<>、transition detection	4	用于控制信号的比较，可以检测跳变的发生
Extend	0、1、X	=、<>、>、>=、<、<=	2	当主要考虑数据的大小时，用于地址或数据信号大小的比较
Extend(w/trans)	0、1、X、R、F、B	=、<>、>、>=、<、<=、transition detection	2	当数据和地址信号的大小和跳变都需要考虑时，可以用于检测跳变的发生
Range	0、1、X	=、<>、>、>=、<、<=、in range、not in range	1	当数据和地址的大小需要考虑时，可以用于检测数值是否在一定范围内
Range(w/trans)	0、1、X、R、F、B	=、<>、>、>=、<、<=、in range、not in range、transition detection	1	当数据和地址的大小和跳变都需要考虑时，可以用于检测跳变和数值是否在一定范围内

“Capture Parameters”选项卡可以对存储深度、采样时刻等参数进行设置，如图 6.43 所示。

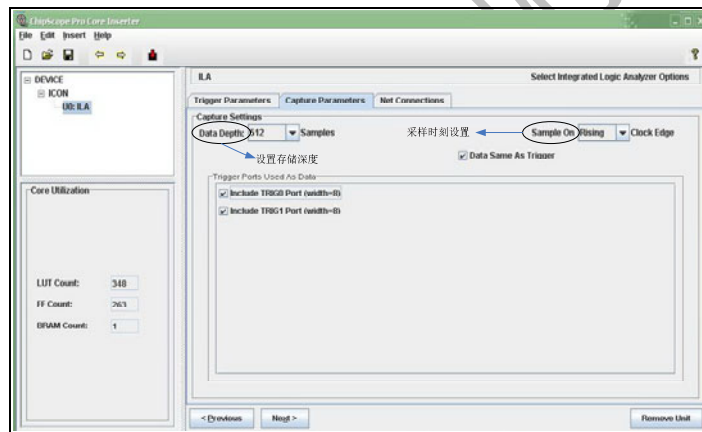


图 6.43 “Capture Parameters”选项卡设置

所谓存储深度，是指在满足触发条件后，要存储多少数据，用于最终的波形显示。ChipScope Pro 可能的最大存储深度为 16384，最大数据位宽为 256bit。实际的数据存储深度和位数由 FPGA 内部剩余的 BlockRam 的数量决定。

对于“Data Same As Trigger”选项，有时要观测的信号就是设置的触发条件中的信号，此时选中此项即可。有时设定了触发条件后，想观察别的数据信号，这时可以不选中此项，数据与触发信号完全独立。

“Net Connections”选项卡可以设置触发端口信号线与要观测的信号线的连接，要观测哪些信号，就将这些信号与端口的信号线连接即可，如图 6.44 所示。

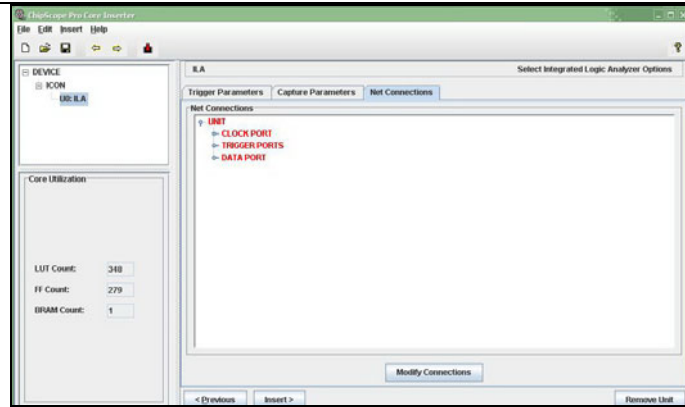


图 6.44 “Net Connections” 选项卡设置

设置的连接信号可以分为 3 类: 时钟信号(CLOCK PORT)、触发端口信号(TRIGGER PORTS)和数据信号(DATA PORT)。单击“Modify Connections”按钮会出现如图 6.40 所示对话框。

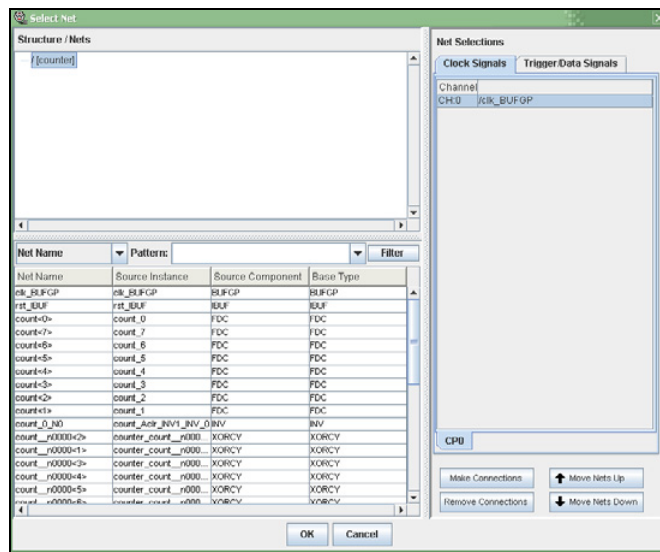


图 6.45 网线连接对话框

设置完所有信号后，端口名字会变为黑色，否则为红色。设置完上述各项后，单击“Insert”按钮，逻辑分析仪的网表就插入到原来的设计网表当中。之后在 ISE 下完成布局布线并下载后，就可以用 ChipScope Pro Analyzer 进行观测了。

6.7.4 ChipScope Pro Analyzer 简介

将逻辑分析的核插入设计当中后，就可以运行 ChipScope Pro Analyzer 进行观测了，ChipScope Pro Analyzer 的启动方式有两种。

- (1) 直接运行“开始”/“程序”/“ChipScope Pro 8.2i”/“ChipScope Pro Analyzer”。
- (2) 在 ISE 下启动。

如图 6.46 所示，双击“Analyze Design Using ChipScope”即可启动，ChipScope Pro Analyzer 界面如图 6.47 所示。

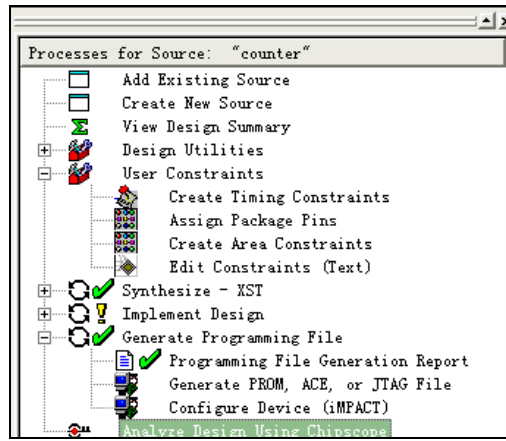


图 6.46 从 ISE 中直接启动 ChipScope Pro Analyzer



图 6.47 ChipScope Pro Analyzer 用户界面

ChipScope Pro Analyzer 使用步骤如下。

1. 单击图标，打开 JTAG 并口连接电缆

在此之前要保证已将 JTAG 与器件连接好，如果连接无误，会出现如图 6.48 所示的对话框。

对话框中会显示 JTAG 连接的 FPGA 类型和所用的配置器件类型，这里使用的 FPGA 为 Spartan 3 系列，配置器件选用的是 XCF02S。

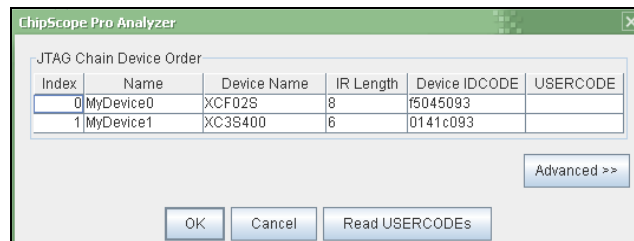


图 6.48 JTAG 正常连接后提示

2. 下载配置文件

在 ISE 下完布局布线后，生成配置文件*.bit 文件。注意：ChipScope Pro 采用 JTAG 方式观测 FPGA 内部信号，这就要求在生成下载文件时。在“Generate Programming File”的属性对话框（如图 6.49 所示）中设置“Startup

Options” / “FPGA Start-Up Clock” 为 JTAG ChipScope Pro 将无法正确配置器件。

下载配置文件时，选择“Device” / “DEV1” 选项，如图 6.50 所示。单击后会出现如图对话框，选择要下载的*.bit 文件，对 FPGA

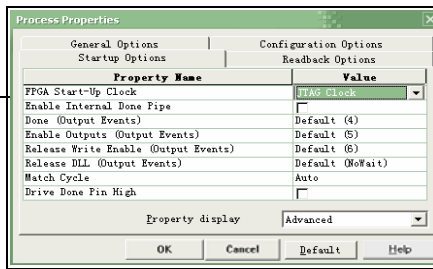


图 6.49 Generate Programming File 属性设置对话框

Clock，否则 / “Configure” 6.51 所示的对进行配置。



图 6.50 配置 FPGA

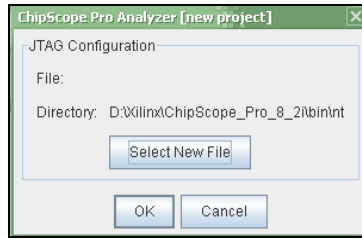


图 6.51 选择配置文件对话框

3. 设置触发条件

成功完成对 FPGA 的配置后，会出现如图 6.52 所示界面。

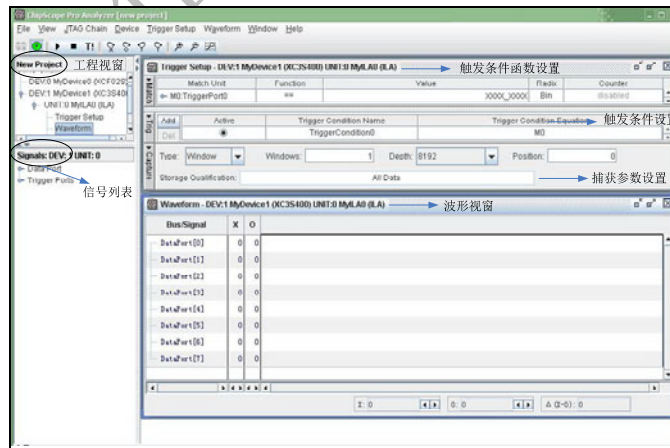


图 6.52 成功完成配置后用户界面

如图 6.52 所示，ChipScope Pro Analyzer 的界面由两部分组成。

左边一栏为工程视窗和信号列表。工程视窗的下拉列表中有“Trigger Setup”、“Waveform”选项。双击后，就会有相应的视窗在右边显示。信号列表中列出了所有信号，在这里可以增加或删除视图中的信号，对信号重命名，也可以将信号组合为总线以便于观察。

右边一栏主要有两个视窗：一个为“Trigger Setup”，用于设置触发条件；一个为“Waveform”用于观察波形。设置触发条件包括设置触发条件函数（Match）、触发条件（Trig）和捕获参数（Capture），下面分别介绍。

(1) “Match” 选项卡。

主要完成触发条件函数的设置。所谓触发条件函数是与表 6.4 中的匹配功能相对应的。要设置数值大小和函数，即选择：=、<>、>、>=、<、<=某数值时，才满足触发条件。当有多个条件时，可以分别设置每个条件的要求满足的函数和数值，如图 6.53 所示。

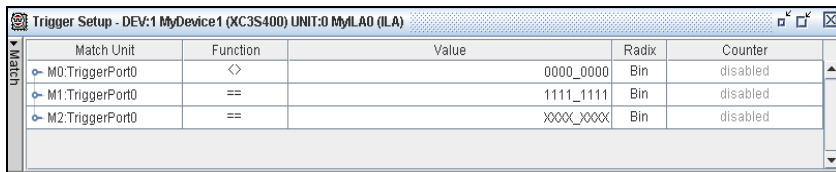


图 6.53 “Match” 选项卡设置对话框

(2) “Trig” 选项卡。

主要用于设置触发条件。在“Match”下设置了触发所需要满足的条件。当有多个条件时这里可以设置是让哪一个条件起作用，也可以将条件设置为几个条件的逻辑组合。或者是将几个条件设置为“条件链”，即当依次满足条件链设置的各个条件后，才可以捕获数据，如图 6.54 所示。

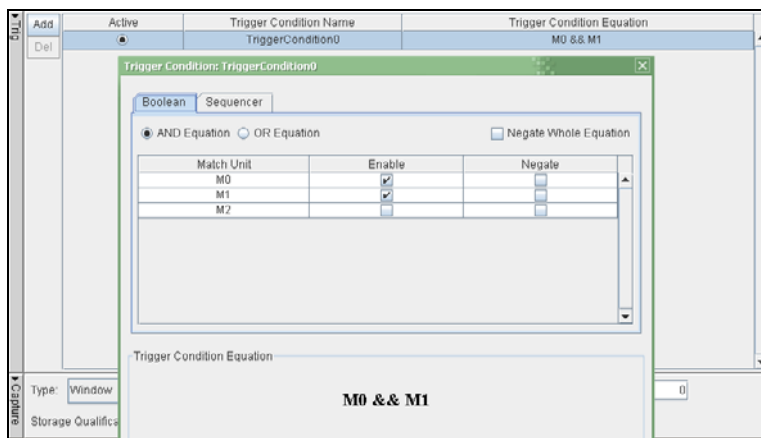


图 6.54 “Trig” 选项卡设置对话框

如图 6.54 所示，单击“Trigger Condition Equation”下的选项，会弹出如图 6.55 所示的对话框。在对话框中，有两个选项卡。在“Boolean”选项卡下，可以设置哪一个条件起作用，也可将条件设置为几个条件的逻辑组合；在“Sequencer”选项卡下，可以设置条件链，如图 6.55 所示。当依次满足条件链下的几个条件后，就被触发。

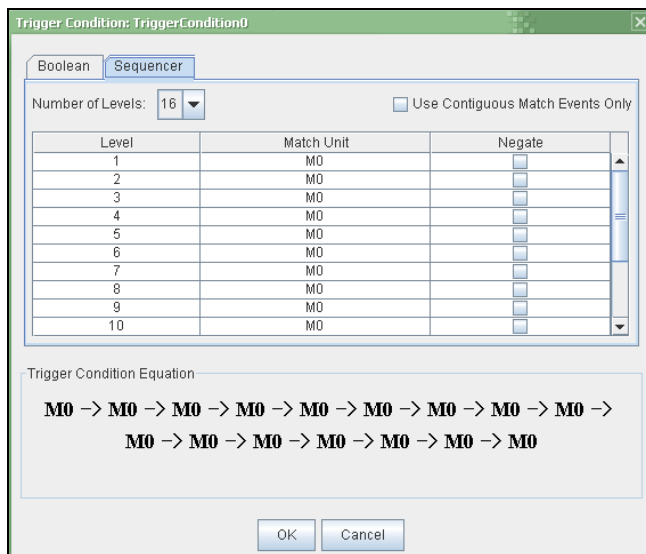


图 6.55 “Sequencer” 选项卡设置对话框

4. 观测波形

触发条件设置好后，单击左上角的▶按钮开始执行。当满足触发条后，ChipScope Pro 开始采集数据，采集到一定数目后（该数目取决于存储深度），就可以观察波形了。如图 6.56 为 ChipScope Pro 显示波形的效果图。

在波形显示窗口下可对波形进行放大🔍和缩小🔍，也可以进行局部放缩🔍。当要观察总

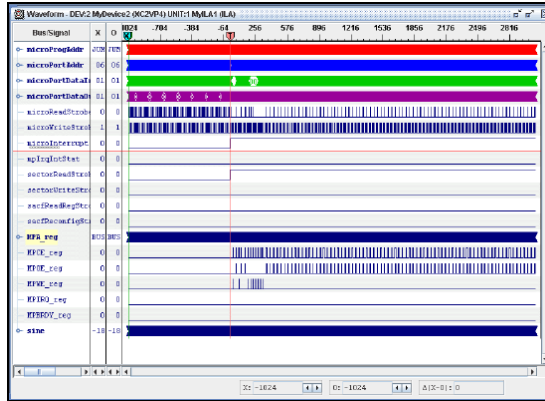


图 6.56 ChipScope Pro 波形显示效果图

线数据时，可以先选中所有总线数据，然后单击右键，选择“Add to Bus”/“New Bus”即可，如图 6.57 所示。

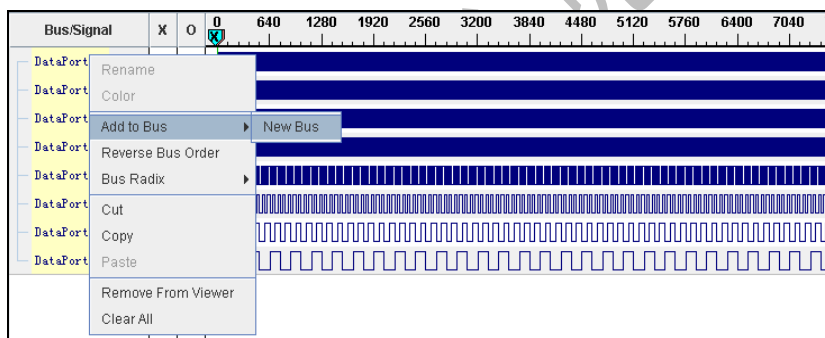


图 6.57 组合总线数据

添加成功后，就可以在新生成的 bus 下看到总线数据。如图 6.58 所示，图中 DataPort 为新生成的总线，设计者可以根据需要修改总线的名称。

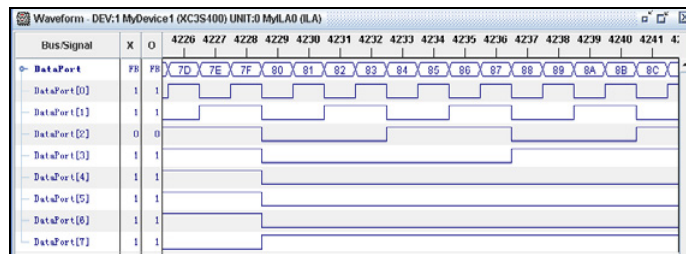


图 6.58 总线数据生成结果

6.7.5 小结

本节对在线逻辑分析工具 ChipScope Pro 作了详细介绍。通过本节的学习，读者应该掌握 ChipScope Pro 的设计流程以及使用方法。

ChipScope Pro 本身的功能很强大，在调试阶段应用很广泛。它可以观察 FPGA 内部的任何信号，使用也比较简单，希望读者能够很好地掌握。

为了让读者能够尽快地熟悉这一工具的使用，在 6.8 节给出一个简单的实例，分别应用两种设计流程来完成，读者可参照这一例程来进一步熟悉 ChipScope Pro 的使用。

6.8 典型实例 11：ChipScope 功能演示

6.8.1 实例的内容及目标

1. 实例的主要内容

本节通过一个简单的计数器，使用 ChipScope 的两种实现流程，基于 Xilinx 开发板完成设计至验证的完整过程。本实例的工作环境如下。

- 设计软件：ISE 7.1i。
- 综合工具：ISE 自带的 XST。
- 仿真软件：ModelSim SE 5.8C。
- 在线调试：ChipScope Pro 8.2i。
- 硬件平台：红色飓风 II 代 Xilinx 开发板。
- 实例内容：计数器。通过 ChipScope Pro 观测计数器的计数值，代码参见本书实例代码的“典型实例 11”文件夹。其中 count_new 文件夹对应采用流程 1 实现的工程，count 文件夹对应用流程 2 实现的工程。

使用 ChipScope Pro 进行在线调试主要有两种实现流程。

(1) 基于 ChipScope Pro Core Generator 的实现流程。

- 调用 ChipScope Pro Core Generator 生成逻辑分析仪的网表文件。
- 修改用户 RTL，插入逻辑分析仪代码。
- 综合，实现，下载 bit 配置文件。
- 调用 ChipScope Pro Analyzer 观察波形。

(2) 基于 ChipScope Pro Core Inserter，通过新建资源的实现流程。

- 对工程文件进行综合，生成网表文件。
- 调用 ChipScope Pro Core Inserter，插入逻辑分析核。
- 布局、布线，生成 bit 配置文件并下载。
- 调用 ChipScope Pro Analyzer 观察波形。

由上面可以看出，两种实现方式的主要区别在于生成逻辑分析核的方式不同。本实例将分别采用这两种方式应用 ChipScope Pro 观测 FPGA 内部信号。

为了便于读者的理解，本实例将编写一个简单的计数器，通过 ChipScope Pro 观测计数的数值。代码的功能仿真结果如图 6.59 所示。

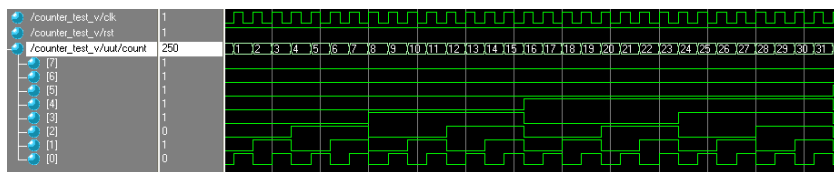


图 6.59 计数器功能仿真结果

其中，count 为 8 位计数值，采用循环计数方式。

2. 实例目标

通过本实例，读者应达到下面的目标。

- 熟悉 ChipScope 工具的使用方法。
- 熟悉基于 ChipScope Pro Core Generator 的实现流程。
- 熟悉基于 ChipScope Pro Core Inserter 的实现流程。

6.8.2 基于 ChipScope Pro Core Generator 的实现流程

基于 ChipScope Pro Core Generator 实现流程的具体步骤如下。

(1) 新建工程，添加源代码。

新建 ISE 工程后，将源代码文件 counter_new.v 及约束文件 counter.ucf 添加至工程中，如图 6.60 所示。

(2) 启动 ChipScope Pro Core Generator。

运行“开始”/“程序”/“ChipScope Pro 8.2i”/“ChipScope Pro Core Generator”命令，启动 ChipScope Pro Core Generator，出现如图 6.61 所示界面。

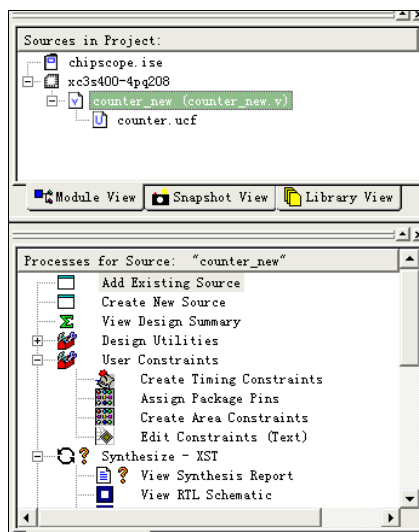


图 6.60 新建工程结果



图 6.61 ChipScope Pro Core Generator 用户界面

(3) 选择集成控制核（ICON 核）。

首先在核类型选择页面中选择生成 ICON 核，如图 6.61 所示。

(4) 选择目录及器件。

如图 6.62 所示，在本实例中选择 Spartan3 器件。输出路径选择新建工程所在的文件夹。控制端口的数目为 1，控制端口的数目可以根据用户的需要设置。如需要观察多组数据时，可以设多组控制端口。这里只需要观察内部计数器的计数值，因此设置为 1。

(5) 选择语言类型和综合工具。

如图 6.63 所示，本实例中设置语言类型为 Verilog，综合工具为 Xilinx XST。

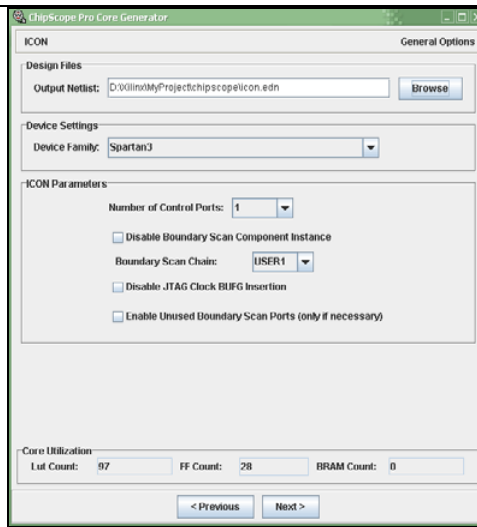


图 6.62 ICON 输出路径和 FPGA 系列设置对话框

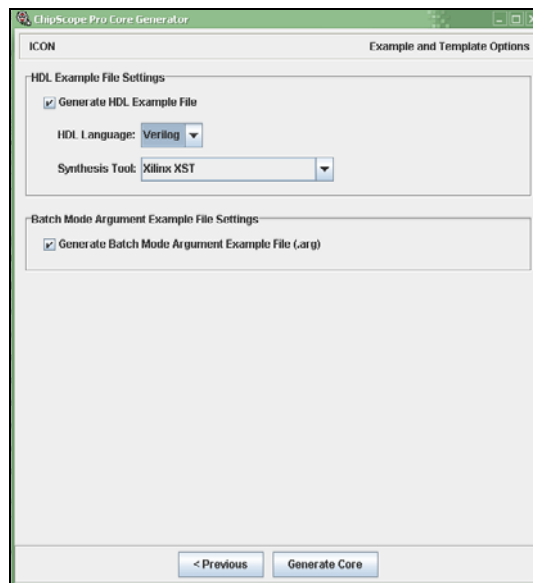


图 6.63 设置语言和综合工具

(6) 生成 ICON 核。

如图 6.64 显示为生成的 ICON 核的相关信息，如发现有误，可以单击“Previous”按钮做修改后重新生成。

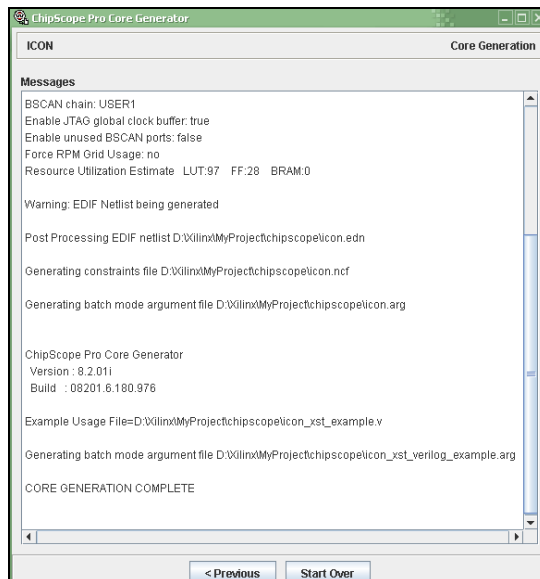


图 6.64 生成 ICON 核

(7) 选择集成逻辑分析仪 (ILA 核)。

生成 ICON 核后, 单击“Start Over”按钮, 回到核类型选择页, 如图 6.65 所示, 选择需要生成的核为 ILA。

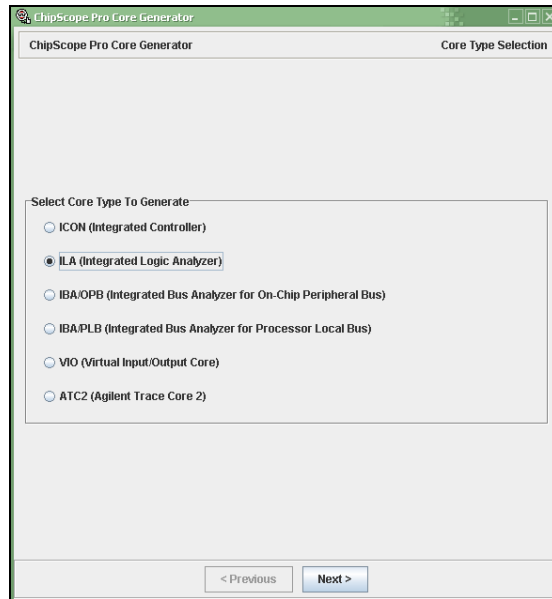


图 6.65 生成集成逻辑分析仪

(8) 设置输出路径、器件参数和时钟参数。

如图 6.66 所示为设置路径为新建工程所在路径, 器件为 Spartan3, 采样时刻为时钟的上升沿。

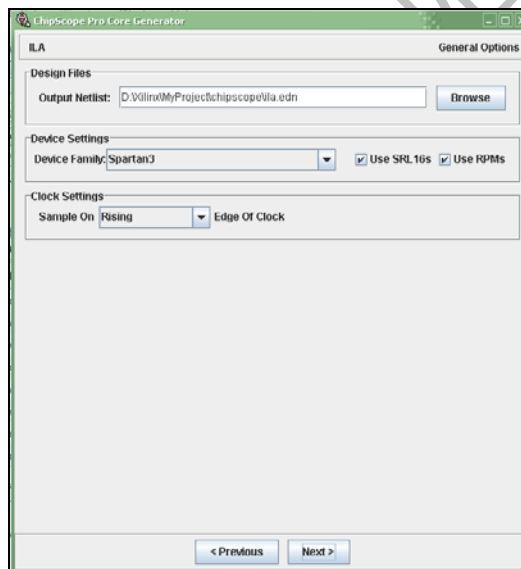


图 6.66 设置 ILA 输出路径、FPGA 器件、采样时刻对话框

(9) 设置触发参数。

触发参数的设置包括: 触发端口数目 (Number of input Trigger ports)、每个触发端口的触发宽度 (Trigger Width)、触发条件判断单元个数 (Match Units) 和类型 (Match Type) 等。

此外, “Enable Trigger Sequencer” 选项用于使能触发条件链。即设置触发条件为一“条件链”, 只有依次满足“条件链”上的各个条件时才会被触发。如图 6.67 所示, 如有与图中参数设置不一致之处, 请自行修改。

(10) 设置存储深度和数据位宽。

存储深度即在满足触发条件后要采集多少数据, 存储深度的大小由 FPGA 的 RAM 资源大小决定。由于 ChipScope 所采集的数据都是保存在 FPGA 内部, 因此存储深度的大小不能超过 FPGA 的 RAM 的最大值。如图 6.68 所示, 选择“Data Same As Trigger”选项表示数据信号与触发信号相同, 数据位宽即为触发端口的触发宽度。

(11) 设置语言类型和综合工具。

如图 6.69 所示, 本实例中选择 Verilog 及 Xilinx XST。

(12) 生成 ILA 核。

单击图 6.69 中的“Generate Core”按钮，生成逻辑分析仪（ILA），如图 6.70 所示。

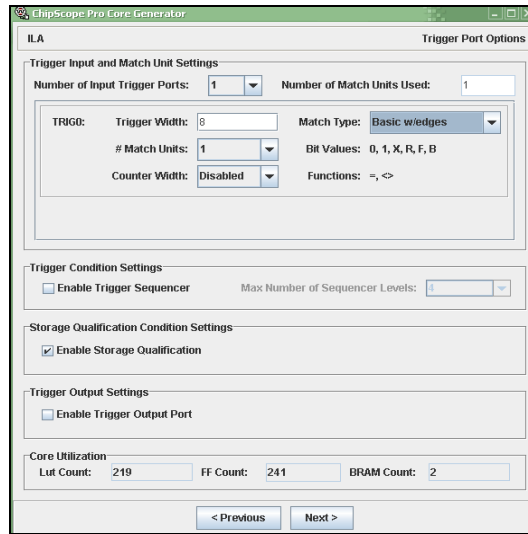


图 6.67 设置触发参数对话框

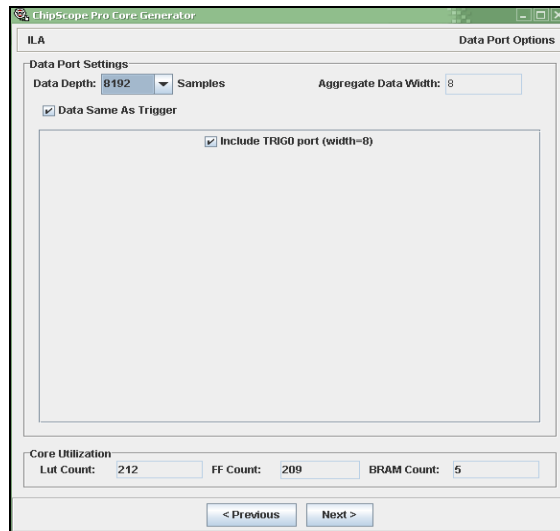


图 6.68 设置存储深度和数据位宽对话框

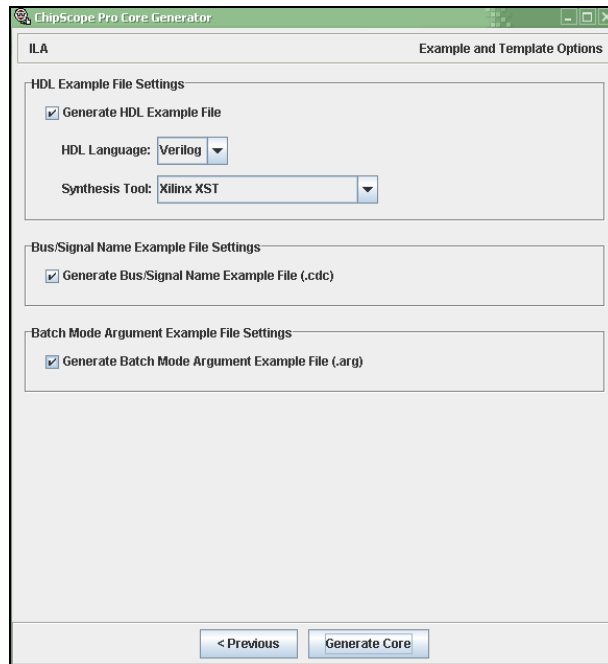


图 6.69 设置生成实例的语言类型和综合工具

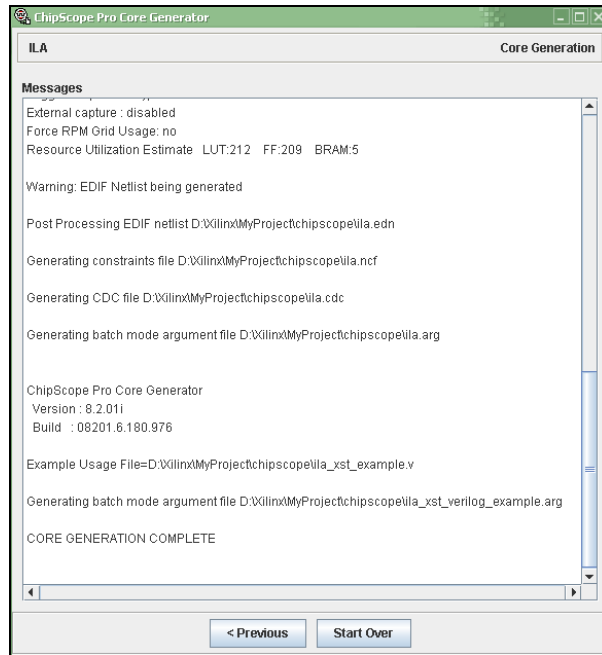


图 6.70 生成 ILA 核

(13) 将 ICON 和 ILA 插入设计。

完成上述步骤后就完成了集成控制核 (ICON) 和逻辑分析仪 (ILA) 的生成, 系统生成的主要文件如表 6.5 所示。

表 6.5 ChipScope 生成文件列表

icon.edn	集成控制器的网表文件
icon.ncf	集成控制器的网表约束文件
icon_xst_example.v	集成控制器的嵌入式例子代码
ila.end	集成逻辑分析仪的网表文件
ila.ncf	集成逻辑分析仪的网表约束文件
ila_xst_example.v	集成逻辑分析仪的嵌入式例子代码

用户需要做的是根据提供的例程代码来修改自己的代码, 将生成的集成控制核和集成逻辑分析仪插入到设计当中。具体修改时, 需要在源文件中添加如下代码:

```

icon i_icon          // ICON core instance,实例化 ICON 核
(
    .control0 (control0)
);
wire [7:0] trig0;
ila i_ila           // ILA core instance,实例化 ILA 核
(
    .control (control0),
    .clk (clk),
    .trig0 (trig0)
);
assign trig0 = count;
    
```

从代码中可以看出要做的工作主要是实例化 ICON 核和 ILA 核。注意要将观察的信号 (在这里为 count) 与 ILA 核的输入信号相连接, 系统时钟与 ILA 核的时钟输入相连接, ICON 的输出控制信号与 ILA 的输入

控制信号相连接。这样就可以通过修改 RTL 代码来插入集成逻辑控制器和集成逻辑分析仪了。


(14) 综合，布局布线，生成配置文件并下载。

具体的操作步骤可以参看 2.6 节的介绍，这里不再详述。需要注意的是 ChipScope Pro 要通过 JTAG 接口与器件连接。生成配置文件时，时钟要设置为 JTAG Clock。

(15) 启动 ChipScope Pro Analyzer。

可通过直接运行“开始”/“程序”/“ChipScope Pro 8.2i”/“ChipScope Pro Analyzer”，也可以在 ISE 集成环境下，在进程浏览器中双击“Analyze Design Using ChipScope”启动。启动后界面如图 6.71 所示。

(16) 打开 JTAG 连接。

单击图标，打开 JTAG 并口连接电缆，在此之前要保证已将 JTAG 与器件连接好，如果连接无误，正常连接后会出现如图 6.72 所示的界面。

(17) 设置触发条件。

触发条件设置如图 6.73 所示。



图 6.71 ChipScope Pro Analyzer 用户界面

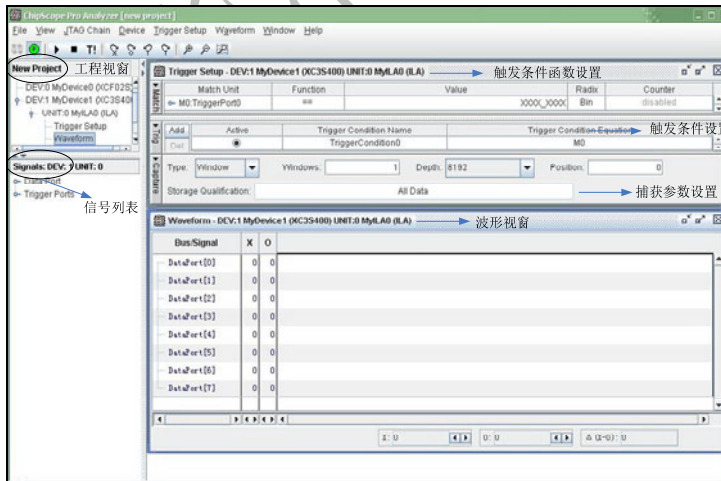


图 6.72 成功连接后用户界面

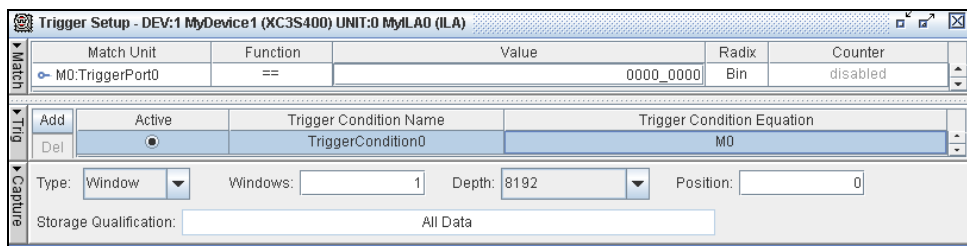


图 6.73 设置触发条件对话框

设置触发条件函数为“==”，数值为 0000_0000（也就是说，触发端口的信号值为 0000_0000 时开始采集数据）。设置采集深度为 8192。这里只有一个触发件 M0，只需采用默认即可。当有多个条件时，要在“Trig”选项卡下设置起作用的正确条件，也可将触发条件设置为几个条件的逻辑组合。

(18) 开始采样调试。

单击左上角的运行按钮，开始采样。捕获触发条件后，ChipScope 将采集所设置存储深度的波形，如图 6.74 所示。

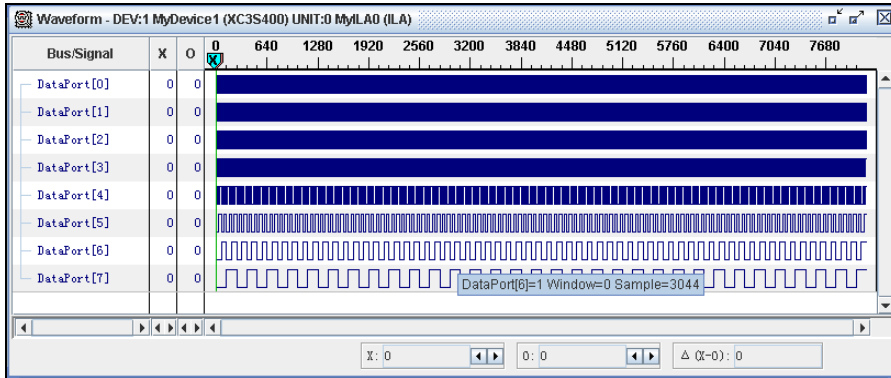


图 6.74 采集信号的波形图

在波形显示窗口下可对波形进行放大和缩小，也可以进行局部放缩。当要观察总线数据时，可以先选中所有总线数据，然后单击右键，选择“Add to Bus” / “New Bus”即可如图 6.75 所示。

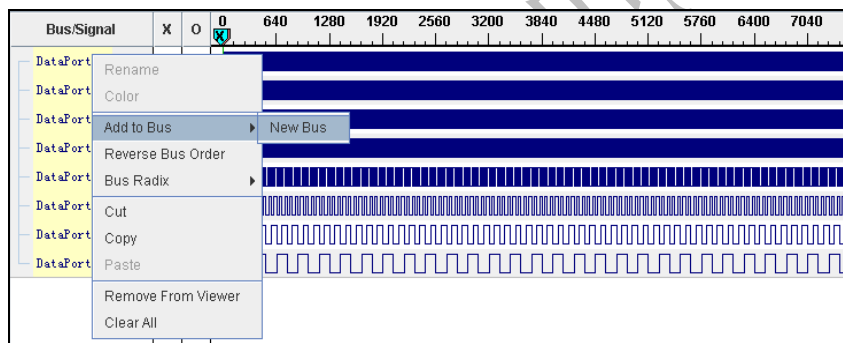


图 6.75 组合总线数据

添加成功后，就可以在新生成的 Bus 下看到总线数据，如图 6.76 所示。图中 DataPort 为新生成的总线，设计者可以根据需要修改总线的名称。

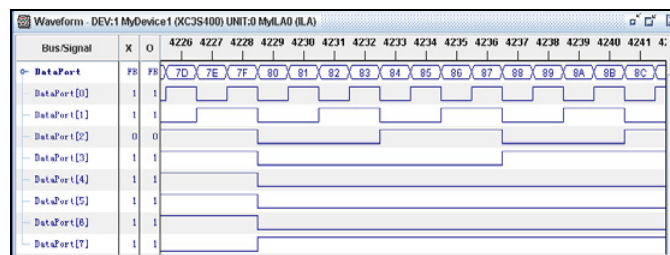


图 6.76 总线数据生成结果

通过上述步骤就完成了应用 ChipScope Pro 的一个完整的流程，这种流程是通过修改源代码来添加逻辑分析仪的。下面的流程可以不用修改设计代码，直接将逻辑分析仪插入到设计的网表文件中，使用方便，应用比较广泛。

6.8.3 基于 ChipScope Pro Core Inserter 的实现流程

基于 ChipScope Pro Core Inserter 实现流程的具体步骤如下。

(1) 新建工程，添加源代码。

参见第一种实现流程的同一步骤。

(2) 新建 ChipScope Pro 资源。

选择 ISE 的 “Project” / “New Source”，再选择 “ChipScope Definition and Connection” 选项，输入新建资源的名称：count，生成 count.cdc 资源文件。如果有现成的扩展名为 cdc 的文件，也可以通过 “Project” / “Add Source” 来添加，如图 6.77 所示。

完成后在工程浏览器中可以看到包含的 count.cdc 文件，如图 6.78 所示。

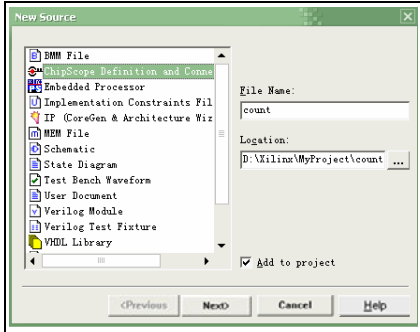


图 6.77 新建.cdc 资源对话框

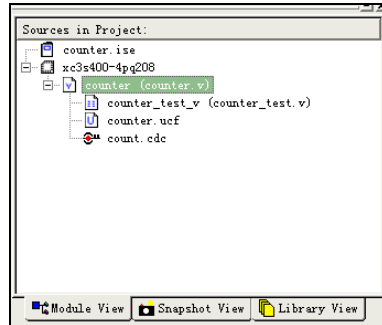


图 6.78 新建.cdc 资源结果

(3) 综合。

与前一种设计流程不同，这里要先对源文件进行综合，在这种设计流程中，ILA 核是直接插入到设计综合后生成的逻辑网表中的，因此要先对源文件进行综合。

(4) 设置 ICON 及 ILA 各项参数。

在工程浏览器中双击 count.cdc 文件，打开 ChipScope Pro Core Inserter 工具，如图 6.79 所示。

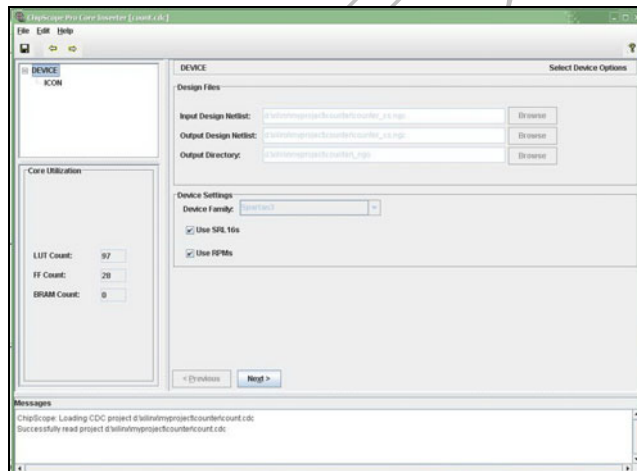


图 6.79 ChipScope Pro Core Inserter 用户界面

在图中可以看到，这里无需再设置输入网表文件的路径，系统会自动找到网表文件，并设置输出文件的路径。

单击 “Next” 按钮进入 ICON 核设置页面，如图 6.80 所示。

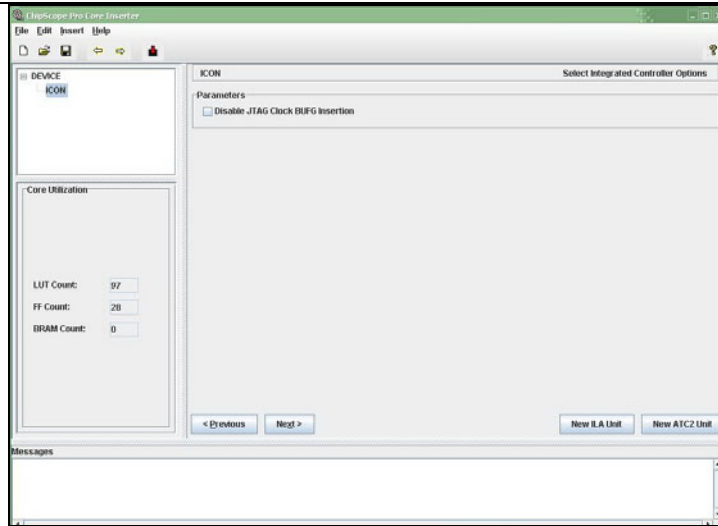


图 6.80 “Select Integrated Controller Options” ICON 对话框

其中，“Disable JTAG Clock BUFG Insertion”用于指定是否禁止在 JTAG 时钟上插入 BUFG。如果选中此项，JTAG 时钟将使用普通布线资源，而不是全局时钟布线。这里此项不选，用全局时钟布线。完成 ICON 核的设置以后，选择单击“Next”按钮，进入 ILA 核设置页面，如图 6.81 所示。

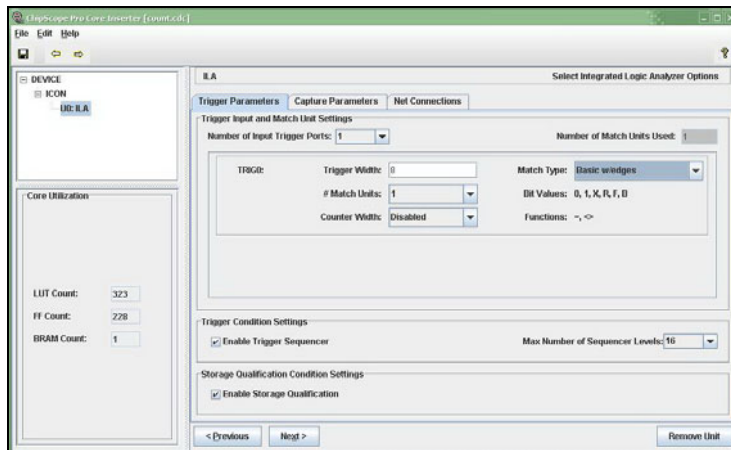


图 6.81 “Trigger Parameters” 选项卡设置

在触发器设置页面中，设置触发端口数、触发宽度、触发条件单元函数和个数等。如有不一致，请读者对照上图修改，设置完毕后单击“Next”按钮出现如图 6.82 所示对话框。

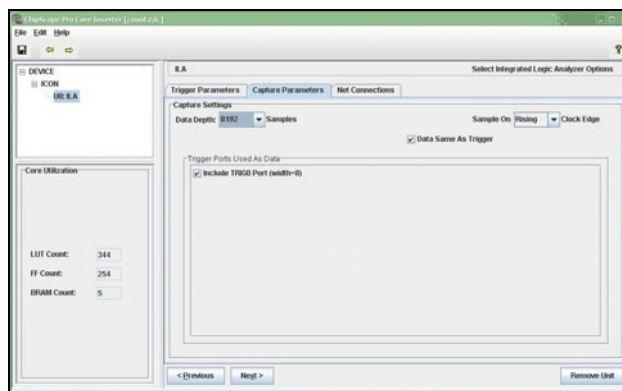


图 6.82 “Capture Parameters” 选项卡设置

如图 6.82 所示，设置存储深度为 8192，采样时刻为上升沿。选中“Data Same As Trigger”，即数据信号与触发信号相同。设置完毕后单击“Next”按钮，可以看到已经定义的 ILA，如图 6.83 所示。

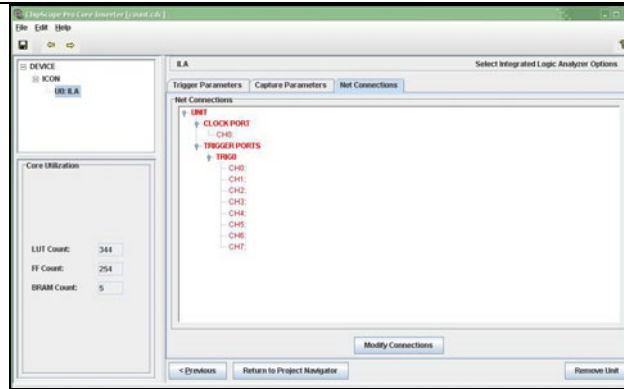


图 6.83 “Net Connection” 选项卡设置

此时由于还没有对网表进行映射，因此网络线都是呈红色显示。单击“Modify Connection”按钮，设置时钟信号和 TRIG 信号与要观测的信号的映射关系，如图 6.84 及 6.85 所示。

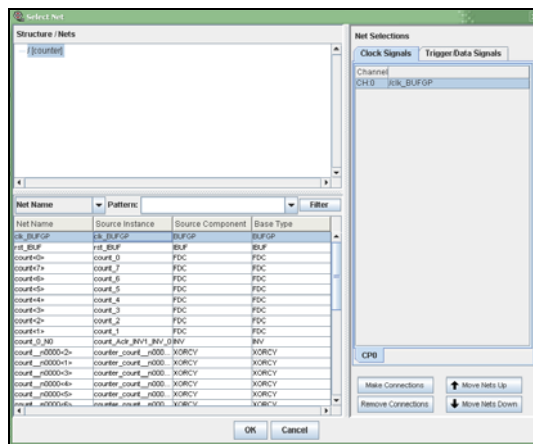


图 6.84 设置时钟信号连接

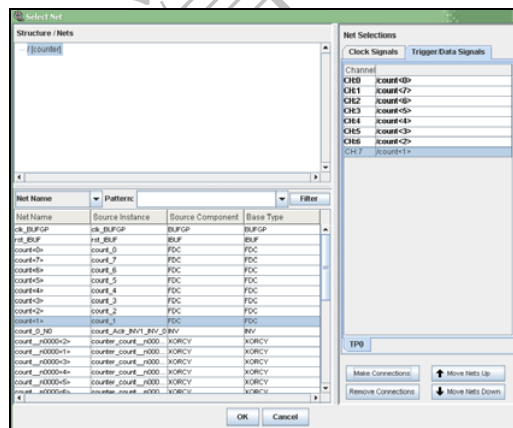


图 6.85 设置数据连接

信号连接主要完成时钟信号的连接和触发信号的映射。将 clk_BUFGP 与 Clock Signals 中的 CH0 连接，将 count<0>~count<7>的内部信号与 Trigger Data Signals 的 CH0~CH7 相连。连接方法很简单，选中要连接的信号，单击“Make Connections”按钮即可。

当所有信号都连接完毕，“Net Connection”选项卡下显示的网路线会由红色变为黑色。如果仍有信号未连接，则仍为红色，直到所有信号连接完毕。

设置完毕后，单击“Return To Project”。此时逻辑分析仪已经插入到了工程网表文件当中了，不需要用户再对源代码进行修改。

(5) 布局布线，生成配置文件并下载。

对工程进行综合、布线布局，之后生成配置文件并下载。

(6) 启动 ChipScope Pro Analyzer 进行观察。

启动 ChipScope Pro Analyzer 后, 剩下的步骤与前一种方法完全一致, 这里就不再重复说明。

6.8.4 小结

本节通过一个具体的实例, 详细演示了 ChipScope Pro 的两种使用方法。希望读者能够按照这两种流程动手练习一下, 熟练掌握这一工具的使用。

在具体应用当中, 第二种方法应用得比较广泛, 因为无需修改源代码, 而且修改逻辑分析仪的各项参数也很方便, 因此推荐读者使用第二种方法。ChipScope Pro 是个功能很强大的在线逻辑分析工具, 熟练地掌握它的用法并应用于自己的设计当中, 将会给调试过程带来极大的方便。

6.9 典型实例 12: 增量式设计 (Incremental Design) 演示

6.9.1 实例的内容及目标

1. 实例的主要内容

6.7 节对增量式设计这一方法的基本概念和流程做了全面的介绍。本节将以一个具体的实例帮助读者熟悉增量式设计的操作流程。

本实例的源代码参见随书光盘 Example6.9。此程序为 PC 机通过串口向 SRAM 写入数据, 再由 FPGA 从 SRAM 中读取数据通过串口将其送到 PC 机。

本实例的重点在于设计过程中是如何应用增量式设计的, 而不是如何实现程序本身的功能。

2. 实例目标

通过本训练, 读者应达到下面的目标。

- 掌握增量式设计与一般设计的区别。
- 掌握增量式设计的设计流程。

6.9.2 实例详解

增量式设计的具体实现步骤如下。

(1) 打开 ISE 工程。

设计的第一步为创建逻辑分组。

在本设计中, top.v 为顶层模块。顶层模块中包含两个功能模块, 分别为: uart_rs232.v 和 sram.v。uart_rs232.v 用于完成串口数据传输, sram.v 用于对于 SRAM 的基本读写操作。

top.v 顶层模块中, 仅包含这两个模块, 没有其他复杂逻辑。并且每个逻辑分组均以寄存器输出, 可将这 3 个模块看作 3 个逻辑分组, 满足创建逻辑分组所必须遵循的原则。

(2) 综合。

为了保证在后面的实现中能够准确地完成分组区域约束, 这里需要对综合的属性进行设置, 在“Processes for Source”中选择“Synthesize-XST”, 单击鼠标右键, 设置综合属性如图 6.86 所示。

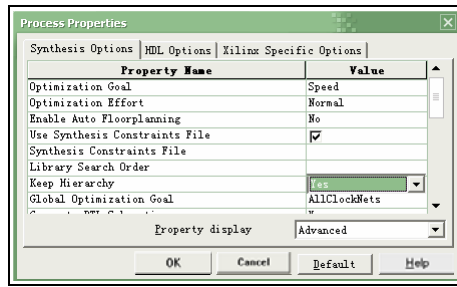


图 6.86 设置综合属性对话框

选择“Synthesis Options” / “Keep Hierarchy”，设置综合属性为保留结构层次模式。

综合完毕要查看综合报告，为了和下面流程中的增量综合结果作对比，请特别注意综合报告中如图 6.87 所示的部分。在没有进行增量综合时，要对每个模块都进行综合和优化。

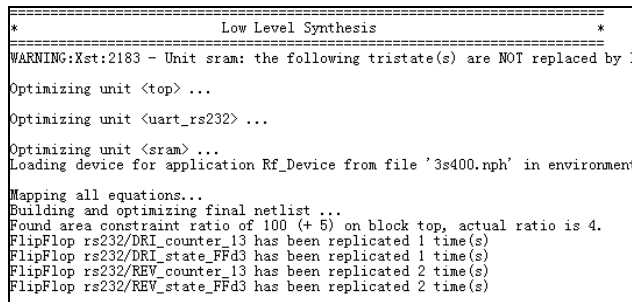


图 6.87 未设置综合约束条件时的综合报告

(3) 设置管脚约束。

在“Processes for Source”中选择“Assign Package Pins”启动设置管脚约束的工具 PACE，如图 6.88 所示，在 Loc 处设置各信号对应的 FPGA 管脚。具体设置可参见例程代码 Source 文件夹下的 top.ucf 文件。

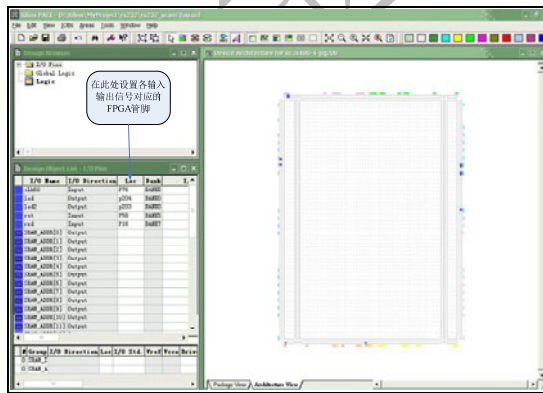


图 6.88 设置管脚约束界面

设置好管脚约束后，保存退出 PACE。在“Processes for Source”中选择“Edit Constraints Text”，即在 Text 模式下编辑约束文件。可以看到系统自动生成的管脚约束文件内容，如图 6.89 所示。

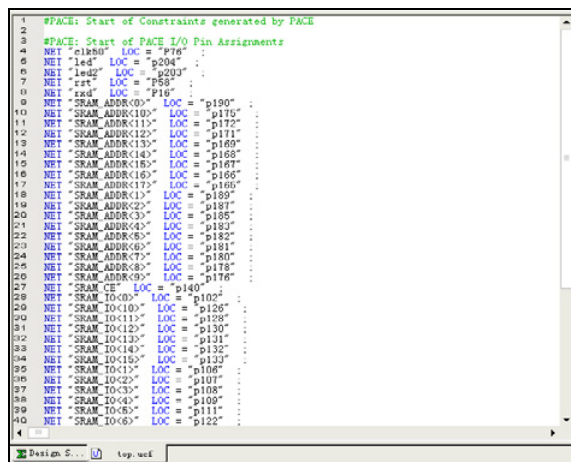


图 6.89 系统生成的管脚约束文件

(4) 设置分组区域约束。


在“Processes for Source”中选择“Create Area Constraints”，启动设置分组区域约束工具 PACE。分别选中 sram 和 rs232，单击左上角的  图标，为每一个逻辑分组设置区域约束，如图 6.90 所示。



图 6.90 进入分组区域约束模式

设置区域约束结果如图 6.91 所示。

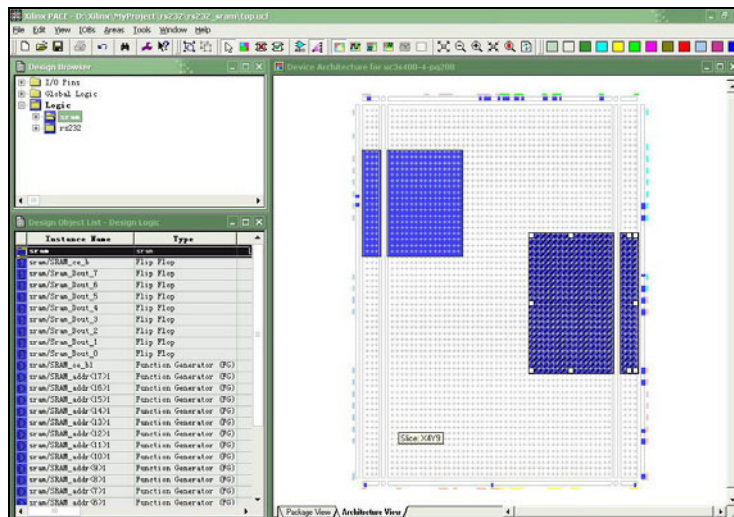


图 6.91 区域约束设置结果

如图 6.91 所示，图中左上角的区域对应的是 SRAM 逻辑分组的约束区域，右下角的区域为 rs232 逻辑分组的约束区域。采用如上的约束是综合两个逻辑分组内部逻辑的复杂性以及 I/O 位置来确定的。这里的约束并不惟一，读者可根据需要进行修改。

设置完毕后，保存设置。在“Processes for Source”中选择“Edit Constraints Text”，即在 text 模式下编辑约束文件，可看到在约束文件中新添了如下内容：

```
#PACE: Start of PACE Area Constraints
AREA_GROUP "AG_rs232" RANGE = SLICE_X0Y55:SLICE_X19Y38 ;//设置 rs232 逻辑分组区域
INST "rs232" AREA_GROUP = "AG_rs232" ;
AREA_GROUP "AG_sram" RANGE = SLICE_X34Y41:SLICE_X55Y18 ;//设置 SRAM 逻辑分组区域
INST "sram" AREA_GROUP = "AG_sram" ;
```

(5) 在普通模式下对工程进行映射和布局布线。

在普通模式下（采用 ISE 下默认的实现属性）对工程进行映射和布局布线是为了得到初始的指引文件，用于在后面的增量设计中指引映射和布局布线。

首先在“Processes for Source”中选择“Implement Design”/“Map”，对工程进行映射。实现后系统会生成：top_map.ncd 和 top_map.ngm 文件，需要用这些文件来指引后面的增量设计的映射。为了避免系统将此文件覆

盖，将其改名为：top_map_guide.ncd 和 top_map_guide.ngm 文件。

接着在“Processes for Source”中选择“Implement Design”/“Place&Route”，对工程进行布局布线，会得到 top.ncd 文件。也将其改名为：top_guide.ncd，将其作为后面增量设计时布局布线的指引文件。

注意

如果在此步骤中无法顺利地完映射和布局布线等步骤，很可能是区域分组约束做得不合适。需要重新做区域分组约束，直到能够顺利地完映射和布局布线为止。映射完成后要查看映射报告看各逻辑分组的资源利用率。如果不合适，需要修改区域约束后重新进行映射和布局布线。映射报告需要注意的内容如图 6.96 所示。

完成映射和布局布线后的结果如图 6.92 所示：

(6) 对工程进行增量综合。

如果在设计调试中发现了某个 Logic Group 需要修改，对其做微小的改动后，要对工程进行增量综合。例如可以修改 uart_rs232.v 的代码，然后进行增量综合。

本实例中的增量综合采用 ISE 自带的工具 XST。采用 XST 进行增量综合时，必须为其添加约束文件（扩展名为 xcf）。添加的约束文件可先在记事本中编辑，然后保存为扩展名为 xcf 的文件。在本实例中综合约束文件为 syn_constraint.xcf 文件，其内容如下：

```
MODEL "top" incremental_synthesis=yes; //使用增量综合
MODEL "sram" incremental_synthesis=yes; //使用增量综合
MODEL "uart_rs232" incremental_synthesis=yes; //使用增量综合
MODEL "top" resynthesize=yes; //启动增量综合
MODEL "sram" resynthesize=no; //不启动增量综合
MODEL "uart_rs232" resynthesize=yes; //启动增量综合
```

前 3 句设置各逻辑分组均采用增量综合，后 3 句通知哪个逻辑分组内容发生改变，需要重新综合，哪个逻辑分组内容未发生改变，无需重新综合。编辑完毕，将该文件添加到当前的工程当中，如图 6.92 所示，上面 syn_constraint.xcf 文件即为添加结果。

添加完毕后，要使这一约束文件有效，还要对综合属性进行设置，如图 6.93 所示。

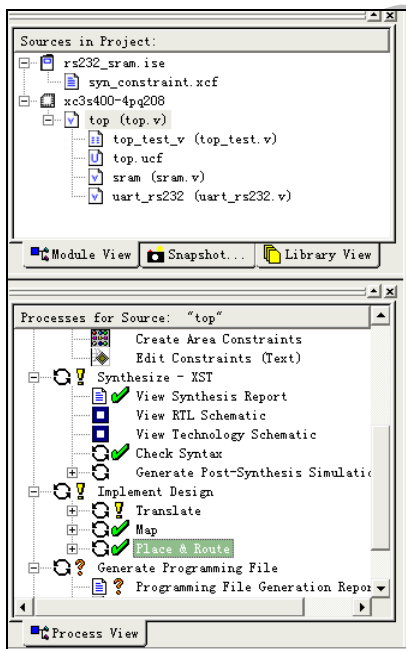


图 6.92 对工程进行映射和布局布线结果

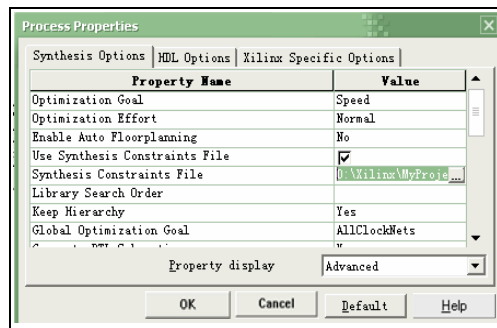


图 6.93 增量综合属性设置对话框

选择“Synthesis Options”/“Synthesis Constraints File”，根据约束文件所在位置，设置约束文件的路径。设置完毕后，对工程进行增量综合。综合完毕后，查看综合报告，注意如下内容。

将图 6.94 与图 6.87 进行比较，可以看出综合中仅对 top 和 rart_rs232 两个逻辑分组重新进行了综合和优化。SRAM 逻辑分组保持不变（Unit <sram> is up to date），表明增量综合实现了。

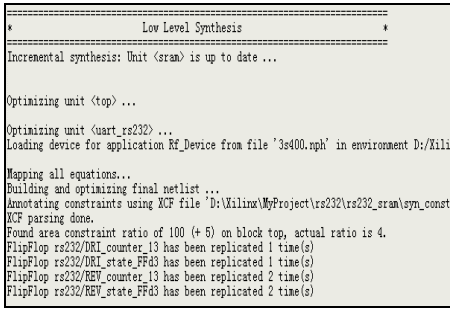


图 6.94 增量综合的综合报告图



图 6.95 增量实现属性设置对话框

(7) 对工程进行增量实现 (Implement)。

完成增量综合后, 就可以利用前面得到的初始的指引文件: top_map_guide.ncd 和 top_guide.ncd 文件来指引增量实现。

首先对工程实现的属性进行设置, 具体设置如下。

单击右键, 选择进程浏览器中的“Implement Design”的“Property”选项, 弹出工程实现属性对话框, 如图 6.95 所示。在增量设计属性页中做如下设置。

- 为“Enable Incremental Design Flow”使能增量实现。
- 为“Run Guided Incremental Design Flow”设置用指引文件来引导增量实现。
- 为“MAP Guide Design File (.ncd)”设置指引映射的指引文件 top_map_guide.ncd 路径。
- 为“PAR Guide Design File (.ncd)”设置指引布局布线的指引文件 top_guide.ncd 路径。

按上述方式设置完毕后, 对工程进行映射和布局布线后, 查看映射和布局布线报告, 对于映射报告注意以下内容。

如图 6.96 所示, 在映射报告中会对每个逻辑分组在各自约束的区域内的资源利用情况作一个总结报告。可以看到各分组所用的 LUT 和 Slices 占其约束区域总量的百分比。如果出现某一逻辑分组的使用率达 90% 以上, 而有些逻辑分组还不到 1%, 则需要重新进行区域约束。

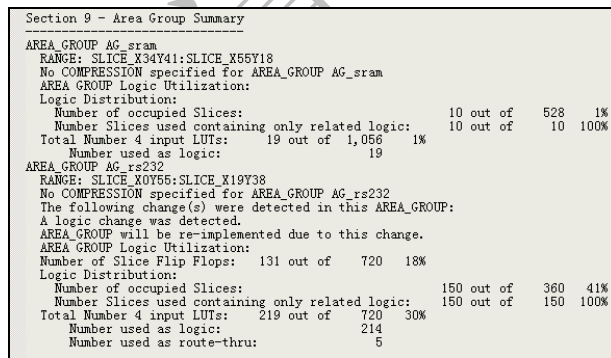


图 6.96 映射报告 (Map Report)

区域约束最好在生成初始指引文件时就确认。在普通模式下完成映射后查看映射报告, 查看每个逻辑分组的资源利用率。如是不合适, 则更改区域约束, 重新进行映射。

对于布局布线报告注意以下内容。

如图 6.97 所示, 布局布线时读取了 top_guide.ncd 作为指引文件, 仅有 rs_232 逻辑分组重新进行了布局布线 (AG_rs232 was re-implemented)。SRAM 逻辑分组完全继承了前面已有的结果 (AG_sram was fully guided), 说明增量实现完成了。


```

Loading database for application par from file:
"D:\Xilinx\MyProject\rs232\rs232_sram\top_guide.ncd"
"top" is an NCD, version 3.1, device xc3s400, package pq208, speed -4

Finished Guide File Processing.

Xilinx Place and Route Guide Results File
=====

Guide Summary Report:

Incremental Design Totals:
Area Groups Guided: 1 out of 2
Ungrouped Logic was re-implemented.
Area Group: Ag_rs232 was re-implemented
Area Group: Ag_sram was fully guided.

For a detailed guide report refer to the "top.grf" file.

Device Utilization Summary:

Number of BUFPMUXs      1 out of 8      12%
Number of External IOBs 37 out of 141   26%
Number of LOCed IOBs   37 out of 37    100%

Number of Slices        160 out of 3584  4%
Number of SLICEMs      0 out of 1792   0%
    
```

图 6.97 布局布线报告 (Place&Route Report)

完成了布局布线后就可以下载调试了。如果仍需要改动，重复步骤（6）和步骤（7），直到设计符合要求为止。

6.9.3 小结

本节以一个具体的实例介绍了 ISE 下增量设计流程。希望读者能够按照上述步骤进一步熟悉 ISE 的增量设计流程，对增量设计有个比较全面的认识，最终将这种设计方法应用到自己的设计当中。

联系方式

集团官网: www.hqyj.com

嵌入式学院: www.embedu.org

移动互联网学院: www.3g-edu.org

企业学院: www.farsight.com.cn

物联网学院: www.topsight.cn

研发中心: dev.hqyj.com

集团总部地址: 北京市海淀区西三旗悦秀路北京明园大学校内 华清远见教育集团

北京地址: 北京市海淀区西三旗悦秀路北京明园大学校区, 电话: 010-82600386/5

上海地址: 上海市徐汇区漕溪路银海大厦 A 座 8 层, 电话: 021-54485127

深圳地址: 深圳市龙华新区人民北路美丽 AAA 大厦 15 层, 电话: 0755-22193762

成都地址: 成都市武侯区科华北路 99 号科华大厦 6 层, 电话: 028-85405115

南京地址: 南京市白下区汉中路 185 号鸿运大厦 10 层, 电话: 025-86551900

武汉地址: 武汉市工程大学卓刀泉校区科技孵化器大楼 8 层, 电话: 027-87804688

西安地址: 西安市高新区高新一路 12 号创业大厦 D3 楼 5 层, 电话: 029-68785218