Verilog HDL基础与开发平台操作指南

1. Verilog HDL程序开发的必备知识

1.1 数字的表示形式

- 在数字逻辑系统中，只存在高电平和低电平，因此用其表示数字只有整数形式，并存在3种表示方法，即：原码表示法（符号加绝对值）、反码表示法（符号加反码）和补码表示法（符号加补码）。

1.1.1 原码表示法

- 原码表示法是机器数的一种简单的表示法，采用符号位级联绝对值的方法来表示数字。其最高位为符号位，用0表示正数，1表示复数；其余部分为绝对数值部分。

- X1= +1010110，X2= -1001010，则其原码分别为：01010110和11001010

1.1.2 反码表示法

- 反码可由原码得到。如果数字是正数，则其反码与原码一样；如果数字是负数，则其反码是对它的原码（符号位除外）各位取反而得到的（除符号位外，所有的0改为1，所有的1改为0）。

- 例如：X1=+1010110，X2= -1001010，则其相应的反码为01010110、10110101。

1.1.3 补码表示法

- 补码表示法是实际中应用最广泛的数字表示法，其表示规则如下：若是正数，补码、反码和原码的表示是一样的；若是负数，补码、反码和原码的表示都是不一样的。

- 实现负数的补码表示有两步：

  a. 取负数的绝对值，按照原码表示为。

  b. 从的最右位向左开始，找出二进制码为“1”的第一位，从第1位（不含）向左余下的位数取其补即可得到补码。

- 可以经过推导得到负数的反码和原码之间的简单换算关系：负数的补码等于其反码在最低位加1，等效于直接用2N+1减去其绝对值，因此补码也被称为“2”的补。

1. Verilog HDL程序开发的必备知识

1.1.4 各类表示方法小结

- 原码的优点就是乘除运算方便，不论正负数，乘除运算都一样，并以符号位决定结果的正负号；若做加法则需要判断两数符号是否相同；若作减法，还需要判断两数绝对值的大小，以使大数减小数。

- 补码的优点是，加法运算方便，不论数的正负都可直接相加，而符号位同样参加运算，如果符号位发生进位，把进位的1去掉，余下的即为结果。

1.1.5 给出各类码字表示法的基本加法运算实例

- （1）首先给出原码的运算示例，其中()10代表十进制数。首先给出一个原码的减法计算实例，完成“1+（-1）=0”的操作。
- (1)10 + (-1)10 = (0)10

- （2）既然，原码不能完成正、负数相加，那么反码形式可以完成此操作吗？仍然以“1+（-1）=0”为例，其相应的反码表达式如下

(00000001) 反+ (11111110)反 = (11111111)反 = (-0)10

1. Verilog HDL程序开发的必备知识

（3）最后给出补码的相关特性说明，负数的补码就是对反码加一，而正数不变。以八比特数据为例，通过（-128)10代替了(-0)10，所以其表示范围为[-128,127]。从直观上，补码消除了(+0)和(-0)，并且具备反码特点，那么究竟其能否完成正、负数的加法运算呢？答案是肯定的，下面给出具体实例。

(00000001)补 + (11111111)补 = (00000000)补 = (0)10

- 基于以上讨论，可以得到一个基本结论：只有补码才能正确完成正、负数的加法运算，并将减法运算转化为加法运算，从而而简化运算规则.但对于乘法操作，则以原码形式计算是最为方便的 。

1.2 常用术语解释

1.2.1 时钟

- 在电子技术中，如果脉冲信号是一个按一定电压幅度，一定时间间隔连续发出的脉冲信号，则改脉冲信号被称为周期信号。脉冲信号之间的时间间隔称为周期；而将在单位时间（如1秒）内所产生的脉冲个数称为频率。

- 常见的时钟信号波形包括：方波、锯齿波波和正弦波等，其典型的时域波形如图1所示。

1. Verilog HDL程序开发的必备知识

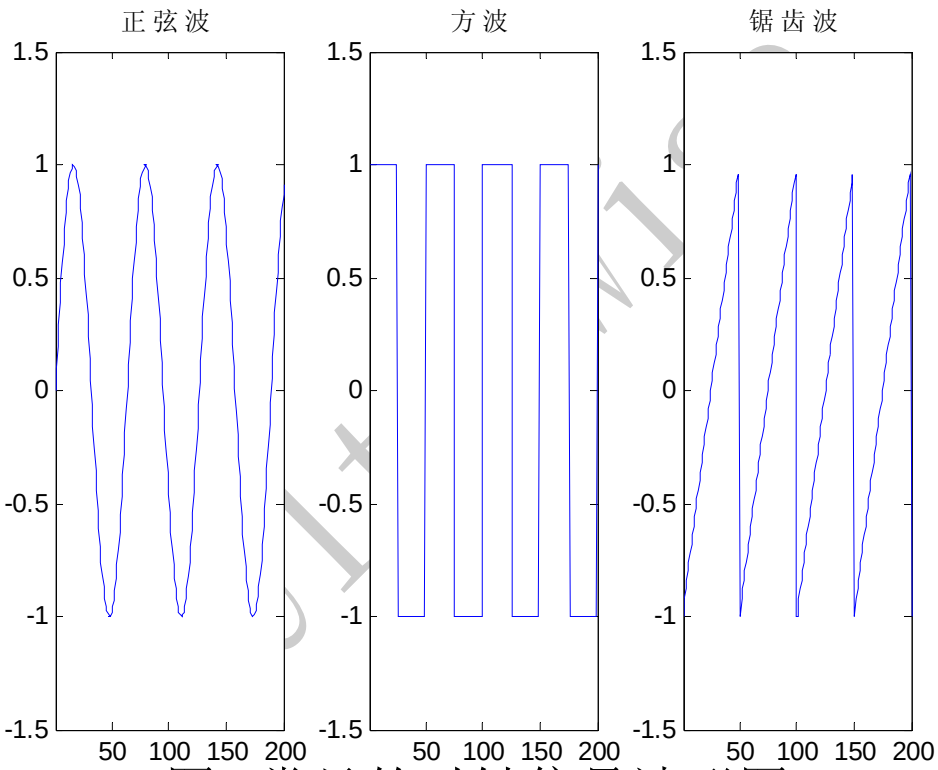

图1 常见的时钟信号波形图

1. Verilog HDL程序开发的必备知识

- 时钟占空比（Duty Cycle）的含义如下：在一串理想的脉冲序列中（如方波），正脉冲的持续时间与脉冲总周期的比值。在图2所示的时钟信号波形中，其中脉冲宽度1us，信号周期4us，则其占空比为0.25。

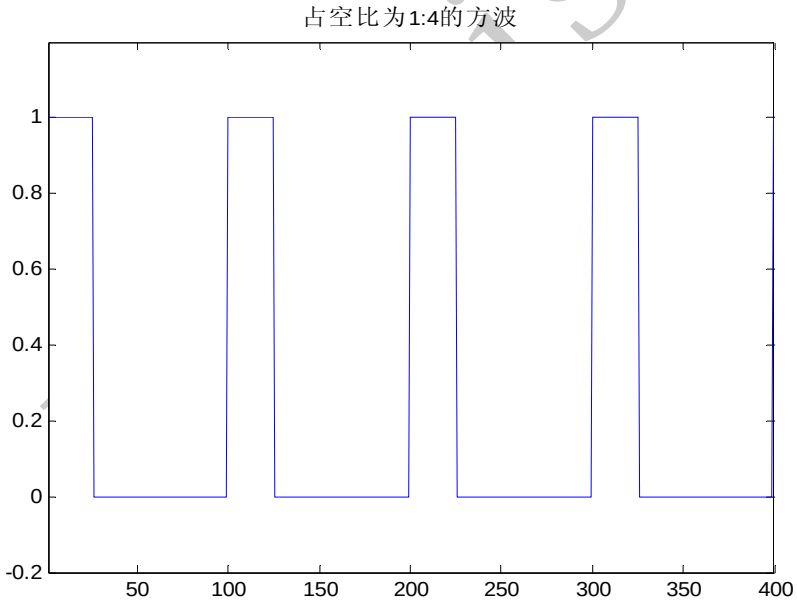

图2 1:4的占空比时钟示意图

1.2.2 资源

- Verilog HDL语言是用来在硬件平台上（PLD或者ASIC）来完成数字系统开发的，所谓的资源指的是所用到的硬件平台的规模大小，最通用的评断标准就是逻辑门数（MOS管的数目）。例如，“目前电路规模越来越大，已达千万门”就意味着该芯片内部集成了超过一千万个COMS管。

- Xilinx Slice V.S. Altera LE

1. Verilog HDL程序开发的必备知识

1.3 Verilog HDL程序的优劣判断指标

- Verilog HDL程序设计首要指标是功能的完备性，达到设计要求，这是任何设计都必须完成的。其次，还包括“面积”、“速度”和功耗指标，是设计的深层次要求。从实用角度来讲，后者的重要性并不亚于功能完整性。

1.3.1 面积性能

- 这里的“面积”主要是指设计所占用的FPGA逻辑资源数目，利用所消耗的触发器（FF）、查找表（LUT）以及各类嵌入式硬核来衡量。

1.3.2 时序性能

- “速度”是指在芯片上稳定运行时所能够达到的最高频率。不同设计在同一芯片上所能运行的最高频率是不一样的。

1. Verilog HDL程序开发的必备知识

1.3.3 功耗性能

- 设计的功耗由两部分组成：静态功耗和动态功耗。前者是由静态电流引起的；后者是电路工作时消耗的功率。
- 其中，静态功耗主要由晶体管的泄漏电流引起，即晶体管即使在逻辑上被关断时，也会从源极“泄漏”到漏极或通过栅氧“泄漏”的小电流。这与设计代码无关，主要取决于芯片型号。
- 动态功耗估算的基本方法是：（1）计算每个设计单元的功耗。（2）累加各个设计单元的功耗，常用以下计算公式：

$$P = \sum C \times V^2 \times E \times f \times 1000$$

- 式中，P表示功耗，单位是mW；C表示电容，单位是F；V表示电压，单位是V；E表示翻转频率，指每个时钟周期的翻转次数；f表示工作频率，单位是Hz。可以看出工作频率越高，设计所造成的功耗就越大。

2. Verilog HDL程序设计模式

2.1 自顶向下的设计模式

- Verilog HDL程序设计时，首先从系统级开始，把系统划分为若干个二级单元，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；再将各个二级单元划分为下一层次的基本单元，一直做下去，直到可以直接用可综合的Verilog HDL语句来实现为止，如图3所示。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。同时，也要求设计人员在开发之前，应当对设计的全貌有一定的预见性。

2. Verilog HDL程序设计模式

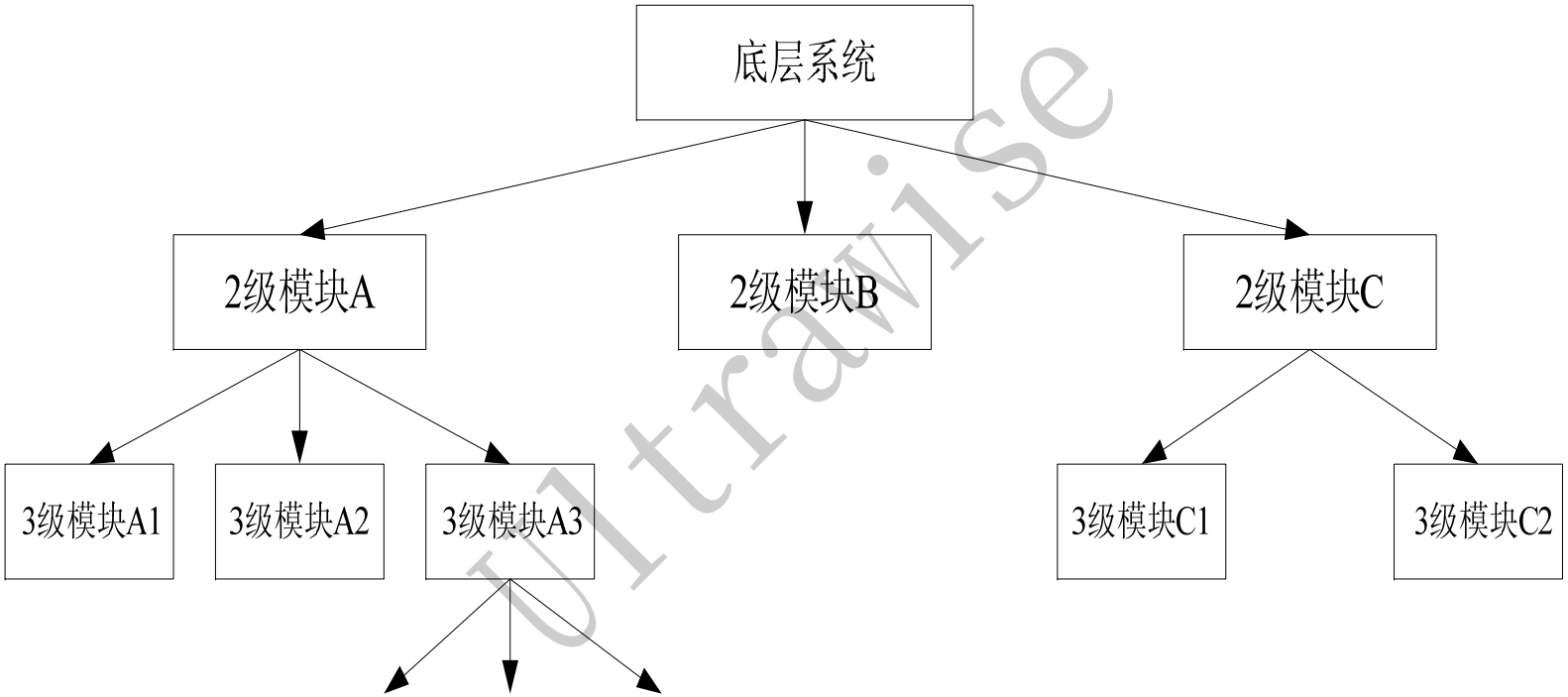

图3 自顶向下的FPGA设计开发流程

2. Verilog HDL程序设计模式

2.2 层次、模块化模式

- 在自顶向下的设计模式中，隐含着对硬件系统的层次化、结构化设计。在设计中的每一层都会有若干个功能相对独立的模块，该层次的硬件结构就由这些模块的互连描述得到。从图2-3中也可以看到，层次化就是纵向的系统划分，模块化则对应着横向的划分。

- Xilinx公司就推出了层次化设计辅助工具PlanAhead。

- 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。

2.3 IP核的重用

2.3.1 IP核的基本概念

- 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，且该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。

- 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。

2. Verilog HDL程序设计模式

- （1）软核
- 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。
- （2）固核
- 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。
- （3）硬核
- 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。

2. Verilog HDL程序设计模式

2.3.2 IP核重用的优势

- IP核是自下而上设计方法学的基础，也是大规模设计的构造基础。IP的可重用特性不仅可以缩短SoC芯片的设计时间外，还能大大降低设计和制造的成本，提高可靠性。此外，随着电子设计种类的级数性增长，单个设计工程师，包括一个设计团队，都无法独立完成各类设计，从项目需求上，迫切需要各类丰富的IP核来满足系统设计的要求。目前，IP核已形成了相当大的产业规模，读者应对该技术的原理和发展进行跟踪。

- 下面将详细介绍如何在ISE中调用Xilinx公司提供的IP核。

3. Xilinx Spartan 3E系列FPGA简介

3.1 Spartan-3E系列FPGA简介

- Spartan-3E是目前Spartan系列最新的中低端产品，具有多款芯片，系统门数范围从10万到160万。Spartan-3E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Xilinx公司性价比最高的FPGA芯片。
  - 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、2.5Ⅴ、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。

3. Xilinx Spartan 3E系列FPGA简介

表1 Spartan-3E系列 FPGA主要技术特征

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
XC3S100E	100k	960	15k	72k	4	2	108	40
XC3S250E	250k	2448	38k	216k	12	4	172	68
XC3S500E	500k	4656	73k	360k	20	4	232	92
XC3S1200E	1200k	8672	136k	504k	28	8	304	124
XC3S1600E	1500k	14752	231k	648k	36	8	376	156

3. Xilinx Spartan 3E系列FPGA简介

3.2 Spartan-3E系列FPGA结构说明

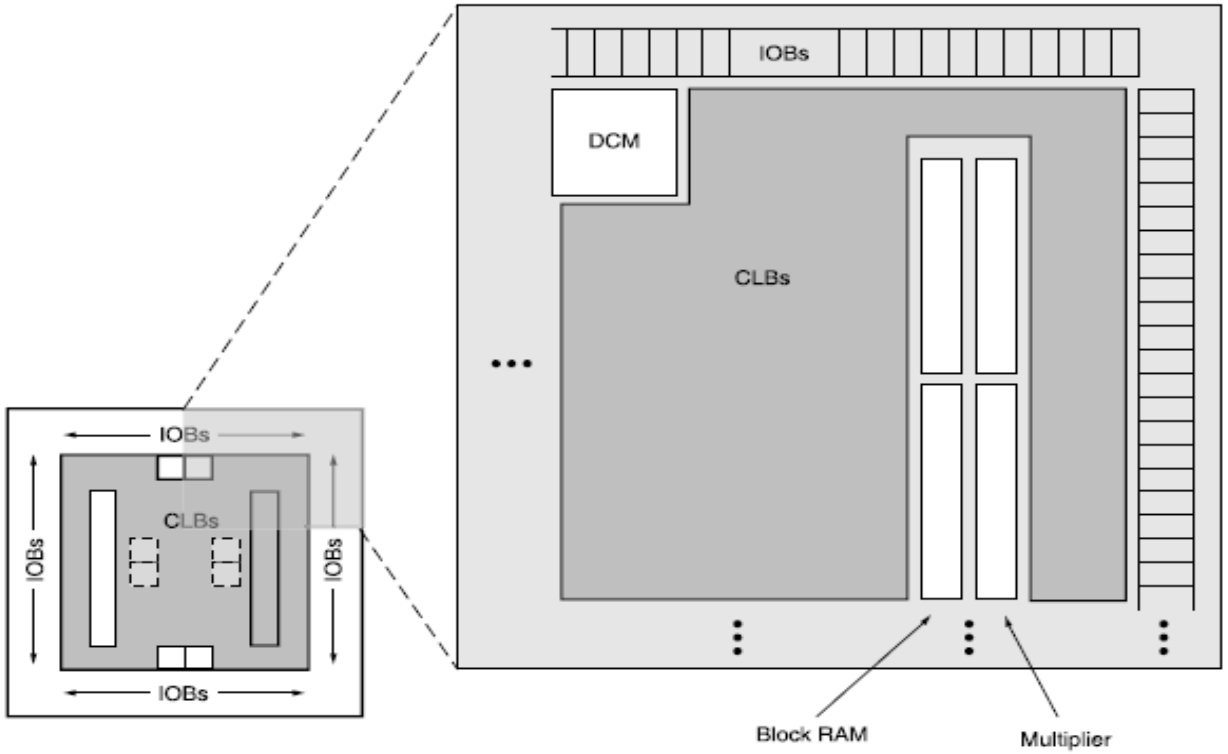

图4 Spartan-3E系列FPGA内部结构示意图

3. Xilinx Spartan 3E系列FPGA简介

3.2.1.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5所示。

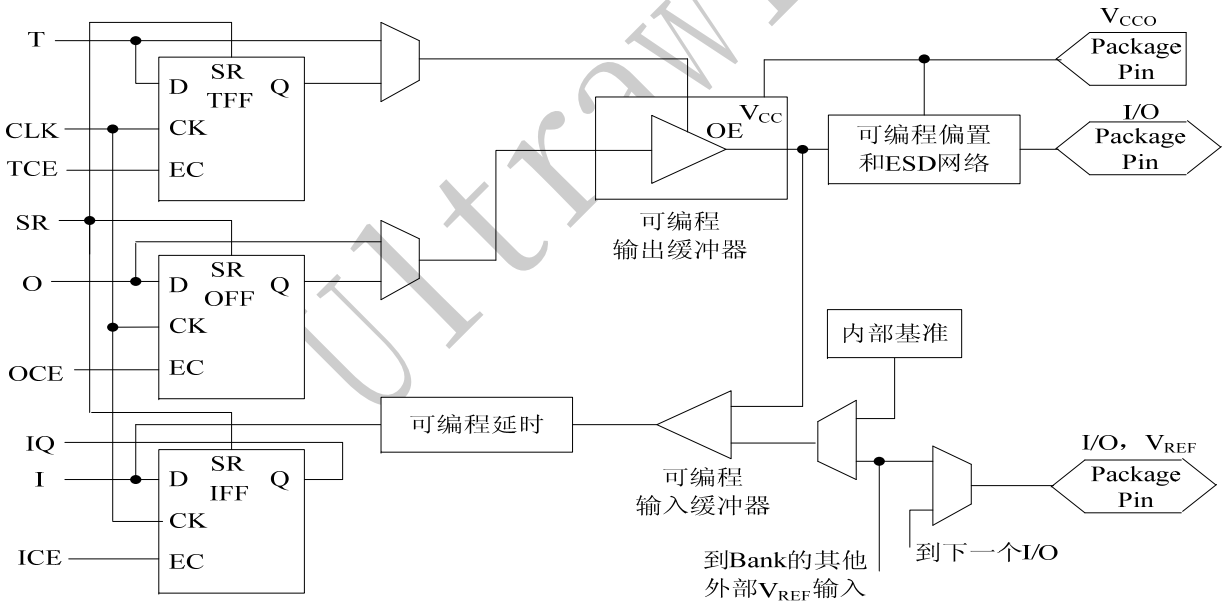

图5 典型的IOB内部结构示意图

3. Xilinx Spartan 3E系列FPGA简介

3.3.2 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。

- Spartan3E的每个CLB包含4个Slice，如图6所示。其中，左边的一对Slice被称为SliceEM，包含分布式RAM和逻辑；右边的一对被称为SliceL，仅包含逻辑。CLB具有输入函数配置，每个CLB可实现16:1多路复用器。

3. Xilinx Spartan 3E系列FPGA简介

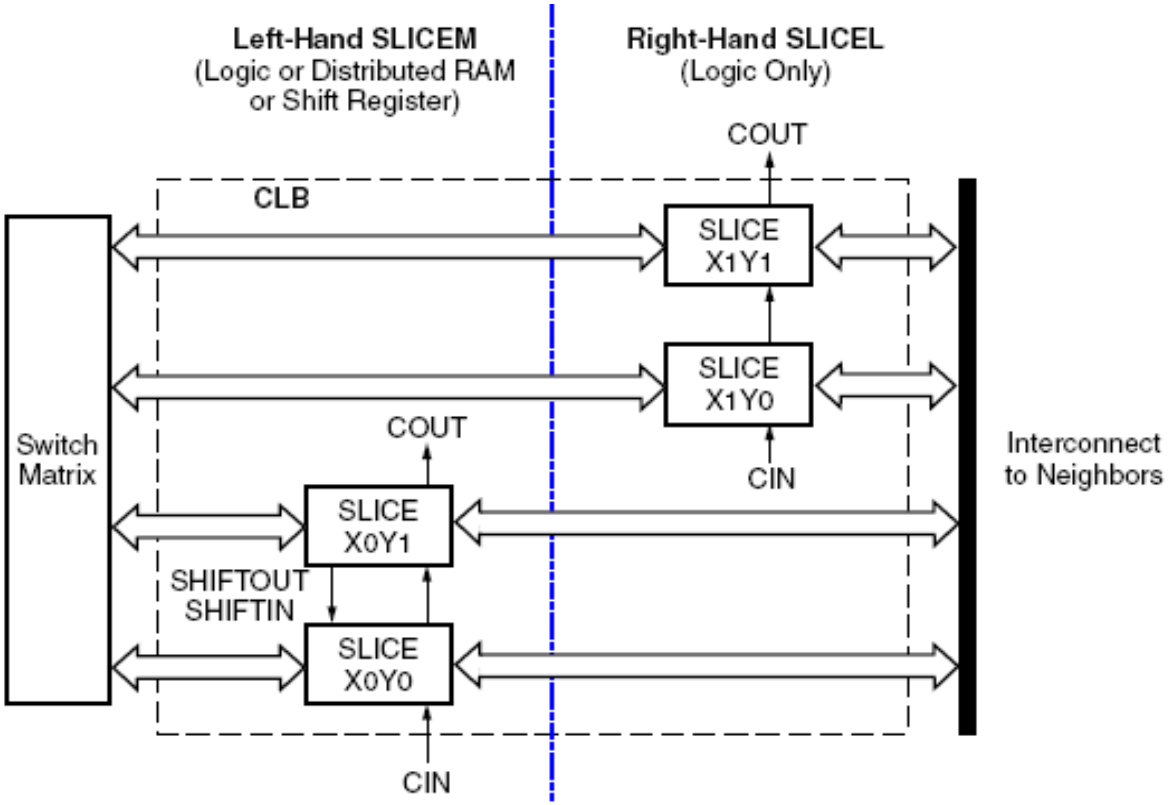

图6 Spartan3E芯片的CLB内部结构

3. Xilinx Spartan 3E系列FPGA简介

- Slice是Xilinx公司定义的基本逻辑单位，其内部结构如图7所示。

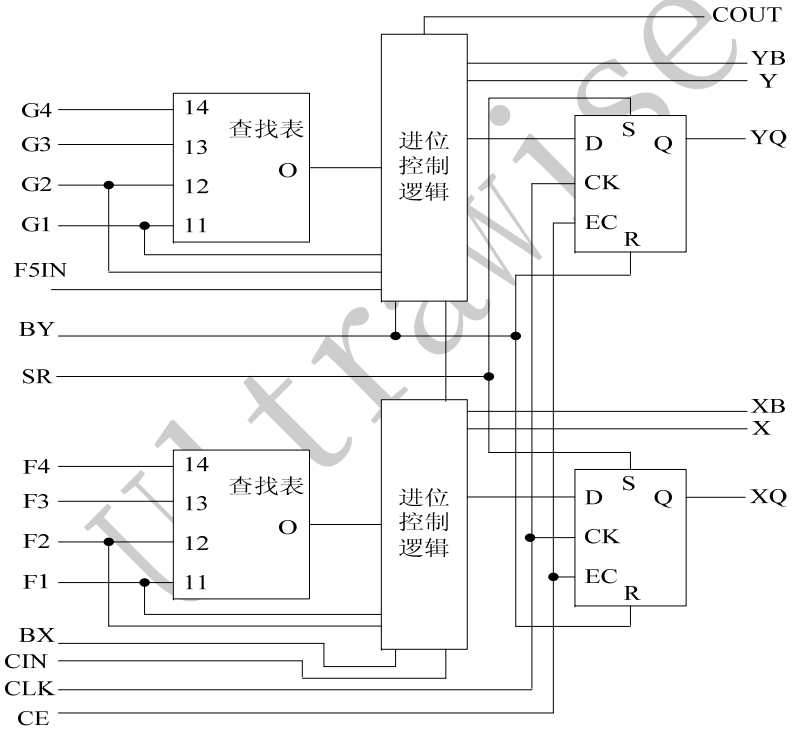

图7 典型的4输入Slice结构示意图

3. Xilinx Spartan 3E系列FPGA简介

3.3.3 数字时钟管理模块（DCM）

- 业内大多数FPGA均提供数字时钟管理（Xilinx的全部FPGA均具有这种特性）。Xilinx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。

3.3.4 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。

3. Xilinx Spartan 3E系列FPGA简介

3.3.5 硬核乘法器

- 硬核是相对底层嵌入的软核而言的，每一个硬核本身就等效于一个ASIC电路。为了提高FPGA的处理能力并降低功耗，Xilinx公司在Spartan 3E内部集成了数目不等的硬核乘法器。

3.3.6 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;
- 第二类是长线资源;
- 第三类是短线资源;
- 第四类是分布式的布线资源。

详细的芯片资料可参考当前目录下Spartan-3E FPGA Family Data Sheet.pdf

4.1 ISE操作基础

4.1.1 ISE简介

- Xilinx的开发工具也在不断地升级，由早期的Foundation系列逐步发展到目前的ISE 10.x系列，集成了EDA开发需要的所有功能，其主要特点有：
  - 全面支持Virtex-5系列器件；
  - 提供了Xilinx新型SmartCompile技术，可以将实现时间缩减2.5倍，能在最短的时间内提供最高的性能，提供了一个功能强大的设计收敛环境；
  - SmartXplorer技术可并行完成FPGA设计，利用分布式处理和多层实施策略来提高设计性能；
  - 添加了基于策略的设计方法；
  - 映射、布局布线过程都支持两种模式：Performance Evaluation Mode和Non Timing Driven Mode；
  - 在一个工程中可以添加多个用户约束文件（UCF），可为不同等级的各个模块灵活添加各类约束；
  - 自动生成Tcl脚本。

4.1.2 ISE用户界面

- ISE10.1i的界面如图8所示，由上到下主要分为标题栏、菜单栏、工具栏、工程管理区、源文件编辑区、过程管理区、信息显示区、状态栏等8部分 。

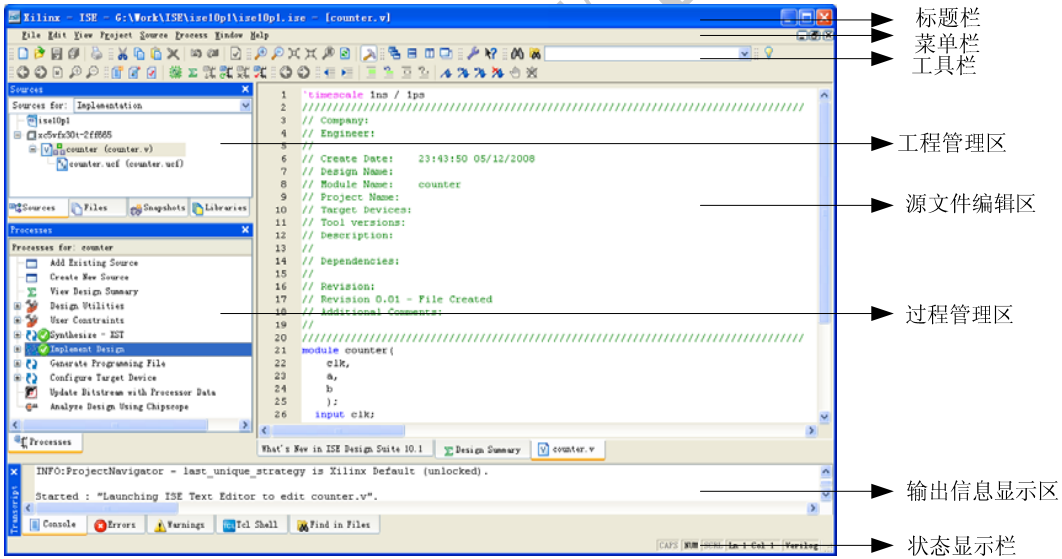

图8 ISE的主界面

4.2 新建工程

- 首先通过双击桌面ISE图标打开ISE或者从开始菜单中启动，依次选择 开始 -> 所有程序 -> **Xilinx ISE 10.1i** -> **Project Navigator**，每次启动时ISE都会默认恢复到最近使用过的工程界面。当第一次使用时，由于此时还没有过去的工程记录，所以工程管理区显示空白。选择“File | New Project”选项，在弹出的新建工程对话框中的工程名称中输入“mycounter”。在工程路径中单击Browse按键，当工程放到指定目录，如图9所示。

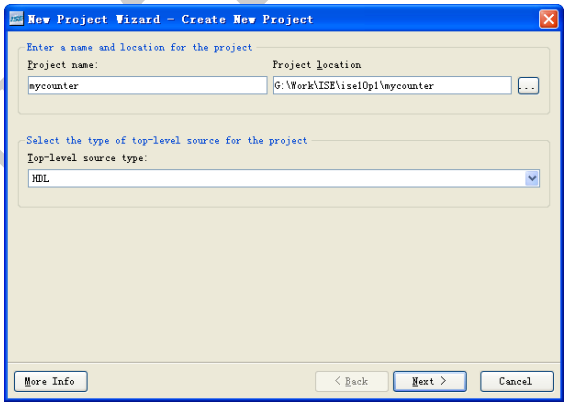

图9 利用ISE新建工程的示意图

4. ISE快速入门

- 然后点击“Next”进入下一页，选择所使用的芯片类型以及综合、仿真工具。计算机上所安装的所有用于仿真和综合的第三方EDA工具都可以在下拉菜单中找到，如图10所示。

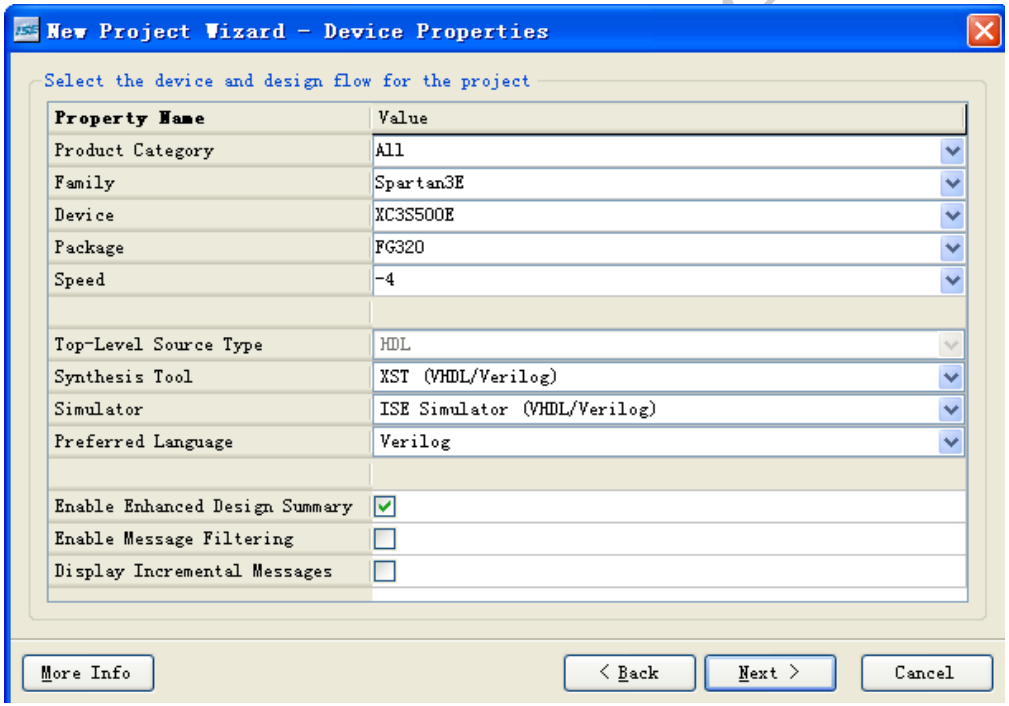

图10 新建工程器件属性配置表

4.3 Verilog HDL代码的输入与功能仿真

- 在工程管理区任意位置单击鼠标右键，在弹出的菜单中选择“New Source”命令，会弹出如图11所示的New Source对话框。

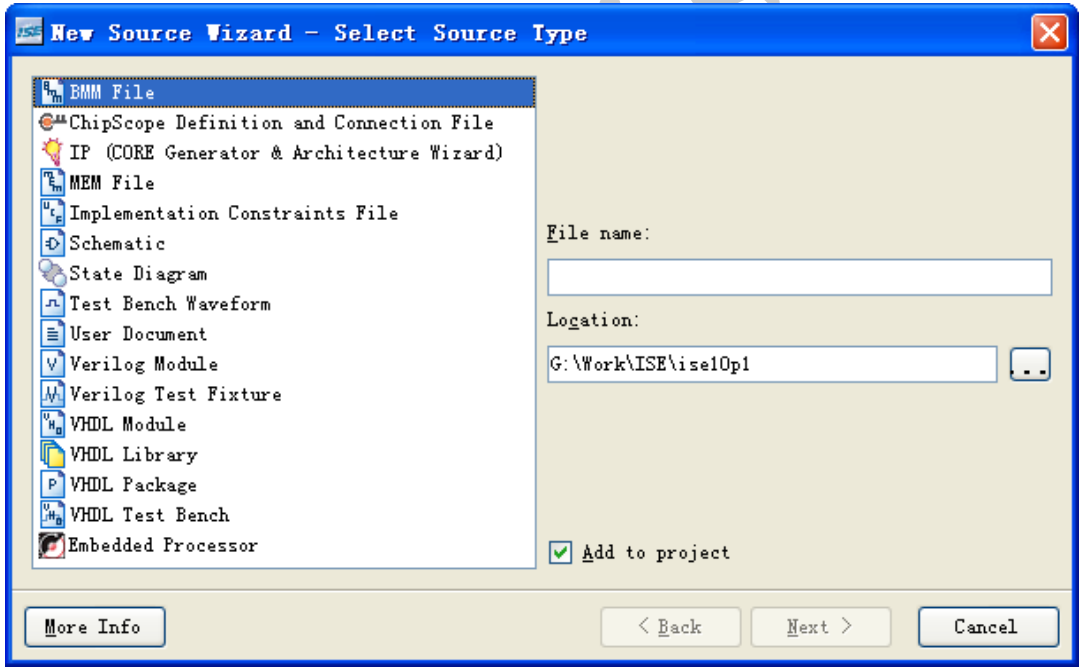

图11 新建源代码对话框

4. ISE快速入门

- 【BMM File】：块存储器映射（Block Memory Map）文件，用于将单个的块RAM连成一个更大容量的存储逻辑单元。
- 【ChipScope Definition and Connection File】：在线逻辑分析仪ChipScope文件类型，具有独特的优势和强大的功能，将在第5章进行讨论。
- 【IP (Coregen & Architecture Wizard)】：由ISE的IP Core生成工具快速生成可靠的源代码，这是目前最流行、最快速的一种设计方法。
- 【MEM File】：存储器定义（Memory Define）文件，用于定义RAMB4和RAMB16存储单元的内容。注意：一个工程只能包含一个MEM文件。
- 【Implementation Constraints File】：约束文件类型，可添加时序和位置约束。

4. ISE快速入门

- 【State Disgram】：状态图类型。
- 【Test Bench Wavaform】：测试波形类型。

- 【User Document】：用户文档类型。
- 【Verilog Module】：Verilog模块类型，用于编写Verilog代码。

- 【Verilog Test Fixture】：Verilog测试模块类型，专门编写Verilog测试代码。
- 【VHDL Module】：VHDL模块类型，用于编写VHDL代码。

- 【VHDL Library】：VHDL库类型，用于制作VHDL库。
- 【VHDL Packet】：VHDL包类型，用于制作VHDL包。

- 【VHDL Test Bench】：VHDL测试模块类型，专门编写VHDL测试代码。

- 单击“New Source”命令，在代码类型中选择Verilog Module选项，在File Name文本框中输入“mycounter”，单击“Next”进入端口定义对话框，如图12所示。

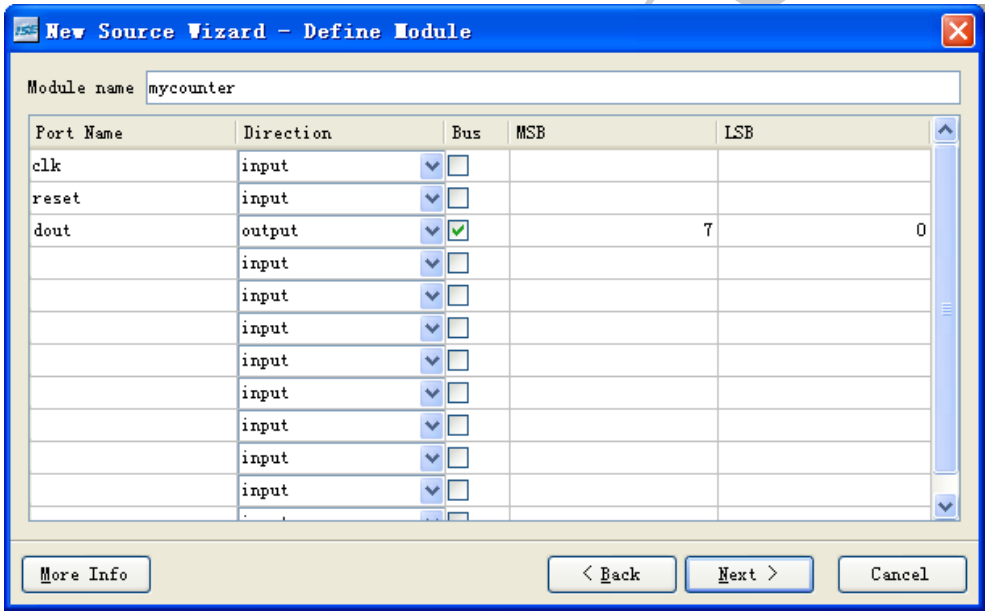

图12 Verilog模块端口定义对话框

4. ISE快速入门

- 其中Module Name就是输入的“mycounter”，下面的列表框用于对端口的定义。“Port Name”表示端口名称，“Direction”表示端口方向（可以选择为input、output或inout），MSB表示信号的最高位，LSB表示信号的最低位，对于单位信号的MSB和LSB不用填写。

- 定义了模块端口后，单击“Next”进入下一步，点击“Finish”按键完成创建。这样，ISE会自动创建一个Verilog模块的例子，并且在源代码编辑区内打开。简单的注释、模块和端口定义已经自动生成，所剩余的工作就是在模块中实现代码。

- 在ISE10.1版本中，增强了代码编辑器的功能，将鼠标光标移动到一对“( )”或者“[]”任一部分的后面，则该对括号以红色显示，便于用户查看是否配对。

4. ISE快速入门

- **例1：**利用Verilog代码实现8比特计数器。

```
module mycounter(
    input clk,
    input reset,
    output [7:0] dout
    );
    reg    [7:0] dout1;

    always @(posedge clk) begin
            if (reset == 0)
                    dout1 <= 0;
                else
                    dout1 <= dout1 + 1;
                end
    assign dout=dout1;
endmodule
```

4.4 测试代码输入

- 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“tb_ mycounter”，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mycounter，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：

4. ISE快速入门

```
`timescale 1ns / 1ps
module tb_ mycounter;
        // 定义输入变量
        reg  clk;
        reg  reset;
        // Outputs
        wire [7:0] dout;
        // 例化被测试模块(UUT)
        mycounter uut (
                .clk(clk),
                . reset (reset),
                .dout(dout)
        );
        initial begin
                // 初始化输入变量
                clk = 0;
```

4. ISE快速入门

```
        reset = 0;
    //全局复位100个仿真时间单位
        #100;
    // Add stimulus here
        reset = 1;
        end
    // 产生仿真所需的时钟信号，其周期为10个仿真时间单位
    always #5 clk = ~ clk;
Endmodule
```

- 其中测试平台的整体架构是由ISE自动生成的，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。本例在其中添加了产生时钟信号的always语句。

4. ISE快速入门

- 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13所示。

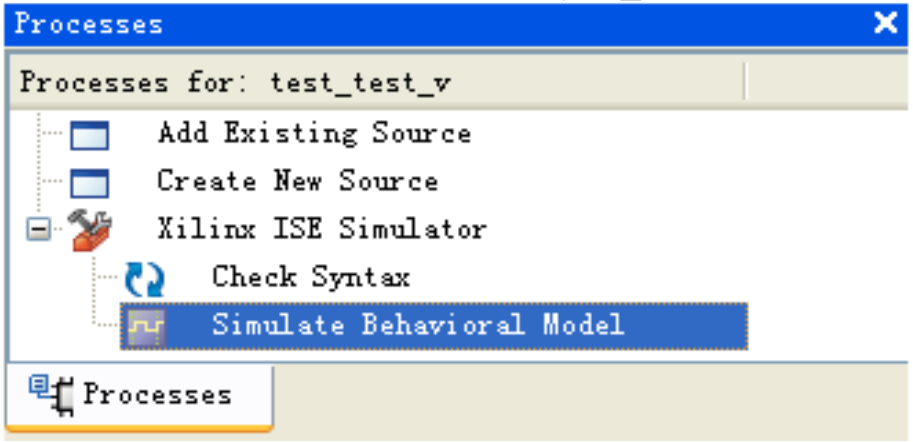

图13 仿真过程示意图

4. ISE快速入门

- 选中图13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14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。

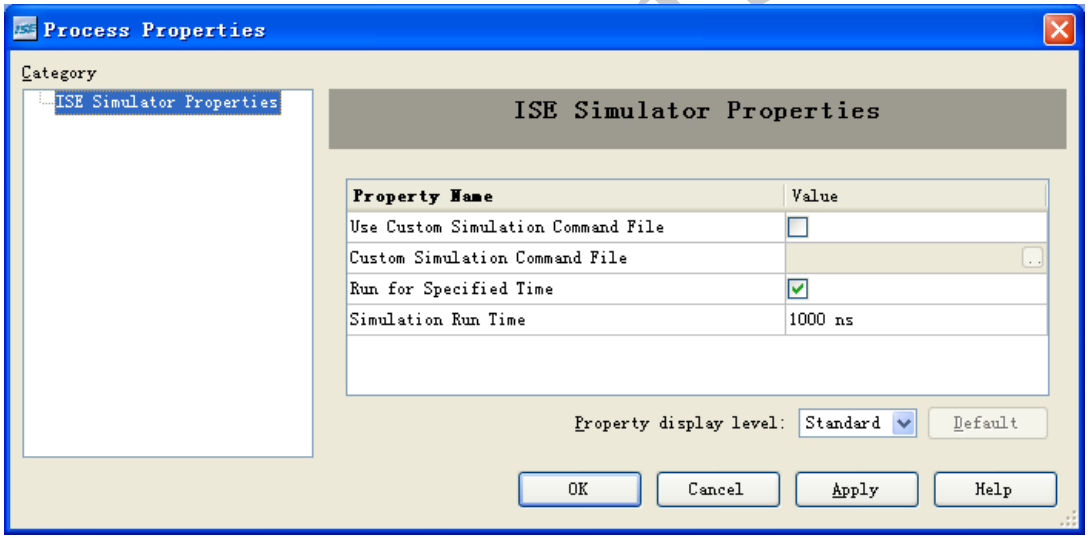

图14 仿真属性设置对话框

- 仿真参数设置完后，就可以进行仿真。首先在工程管理区选中测试代码，然后在过程管理区双击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15所示的仿真结果，从中可以看到设计达到了预计目标。

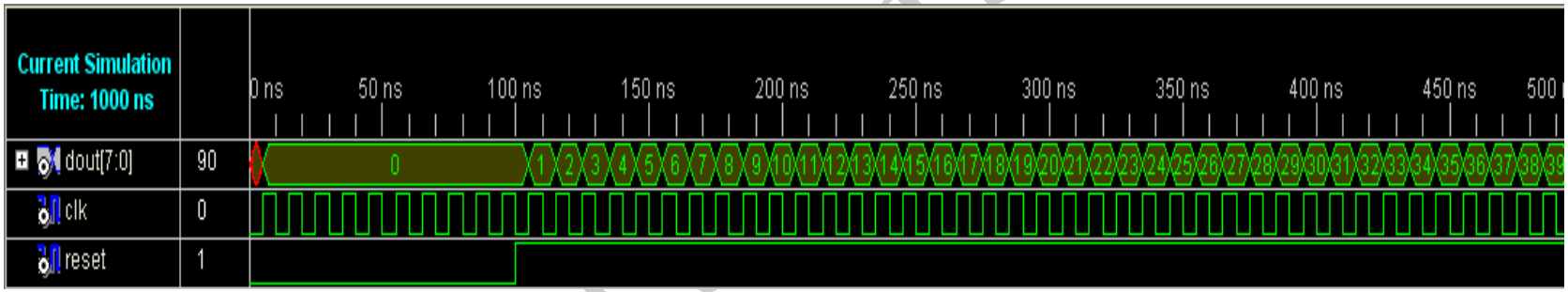

图15 test模块的仿真结果

4.5 Xilinx IP核的使用

4.5.1 Xilinx IP Core简介

- IP Core就是预先设计好、经过严格测试和优化过的电路功能模块，如乘法器、FIR滤波器、PCI接口等，并且一般采用参数可配置的结构，方便用户根据实际情况来调用这些模块。随着FPGA规模的增加，使用IP Core完成设计成为发展趋势。
- IP Core生成器（Core Generator）是基于Xilinx平台的Verilog HDL设计中的一个重要工具，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Xilinx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。

4.5.2 调用IP core的基本操作

- 启动Core Generator有两种方法，一种是在ISE中新建IP类型的源文件，另一种是双击运行“开始 → 程序 → Xilinx ISE Design Suit 10.1 → ISE → Accessories → Core Generator”。

- Xilinx公司提供了大量的、丰富的IP Core资源，究其本质可以分为两类：一是面向应用的，和芯片无关；还有一种用于调用FPGA底层的宏单元，和芯片型号密切相关。本节以调用和芯片结构无关的比较器为例来介绍第一种方法。

4. ISE快速入门

- **例2：**生成比较器的IP核，实现“>=”的逻辑判断，并通过Verilog HDL代码调用该IP核以及完成功能仿真。
- （1）在ISE中新建工程comparator，然后在工程管理区，单击右键，选择“New Source”命令，在文件类型中选择“IP（CORE Generator & Architecture Wizard）”，并在右端的“File Name”文本框输入comparator，如图16所示。然后单击“Next”按键进入下一页。

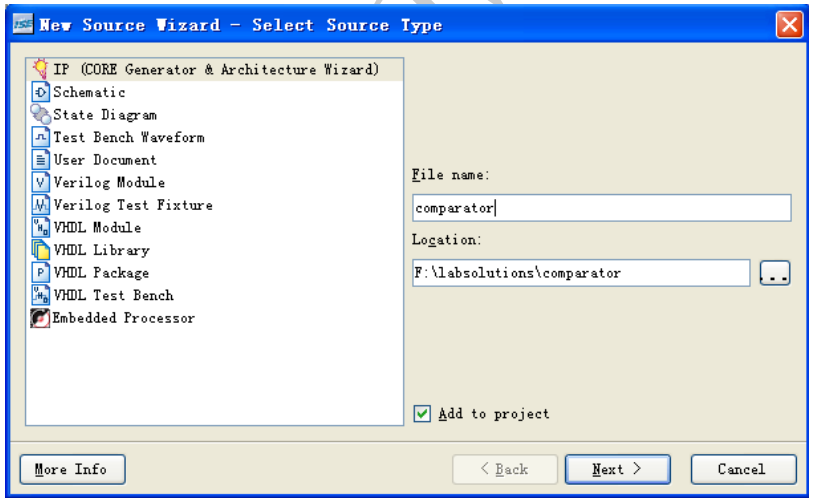

图16 IP核新建向导页面

- （2）然后在弹出的“Select IP”页面，选择“Math Functions”类别下“Comparators”分类中的“Comparator v9.0”，如图17所示。

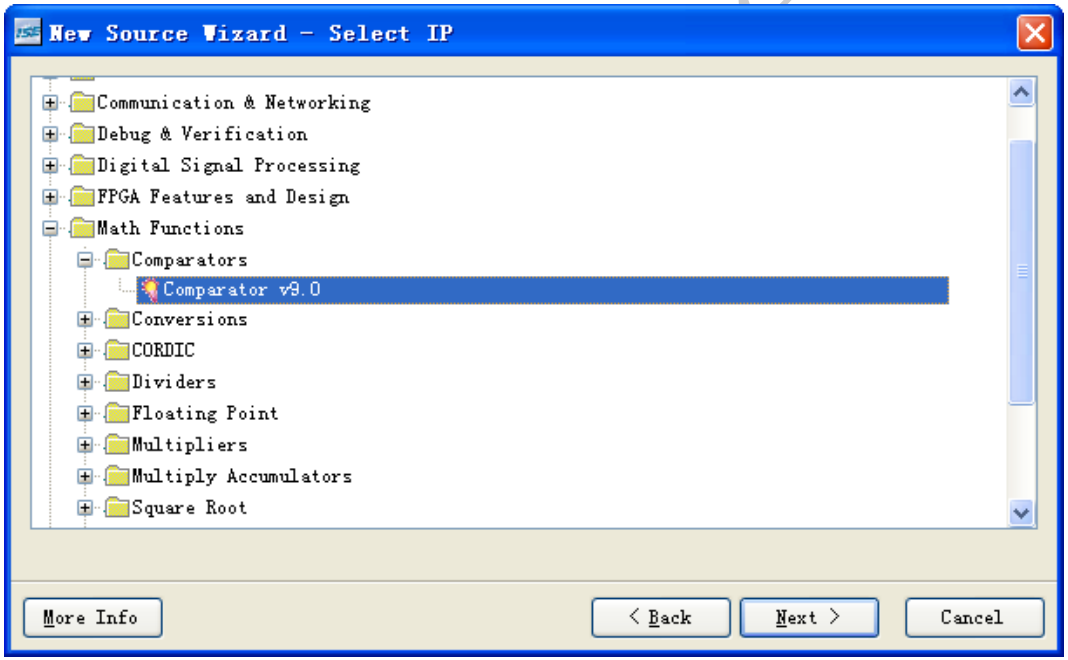

图17 IP核类型选择页面

- 然后单击“Next”按键进入IP 核向导的小结页面，列出了IP核的存储路径以及类型，如图18所示。如果确认无误，可以单击“Finish”按键，完成IP核的生成向导过程；如果参数有误，可单击“Back”按键，返回相应页面进行修改。

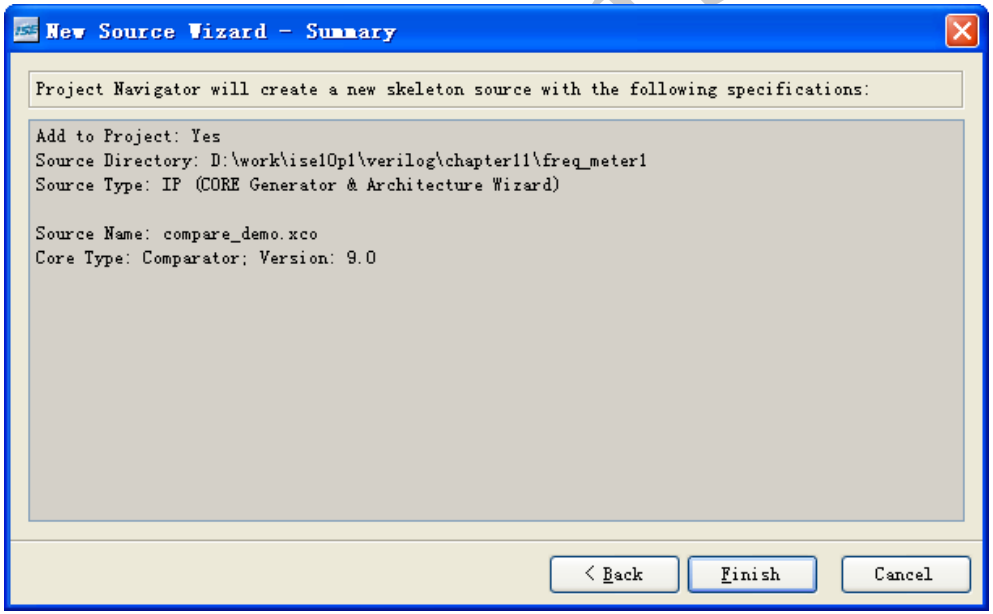

图18 IP核向导小结页面

4. ISE快速入门

- （3）在弹出的确认页面中，点击“Finish”按键，ISE会自动完成IP核文件生成过程，进入IP核配置页面，如图19所示。

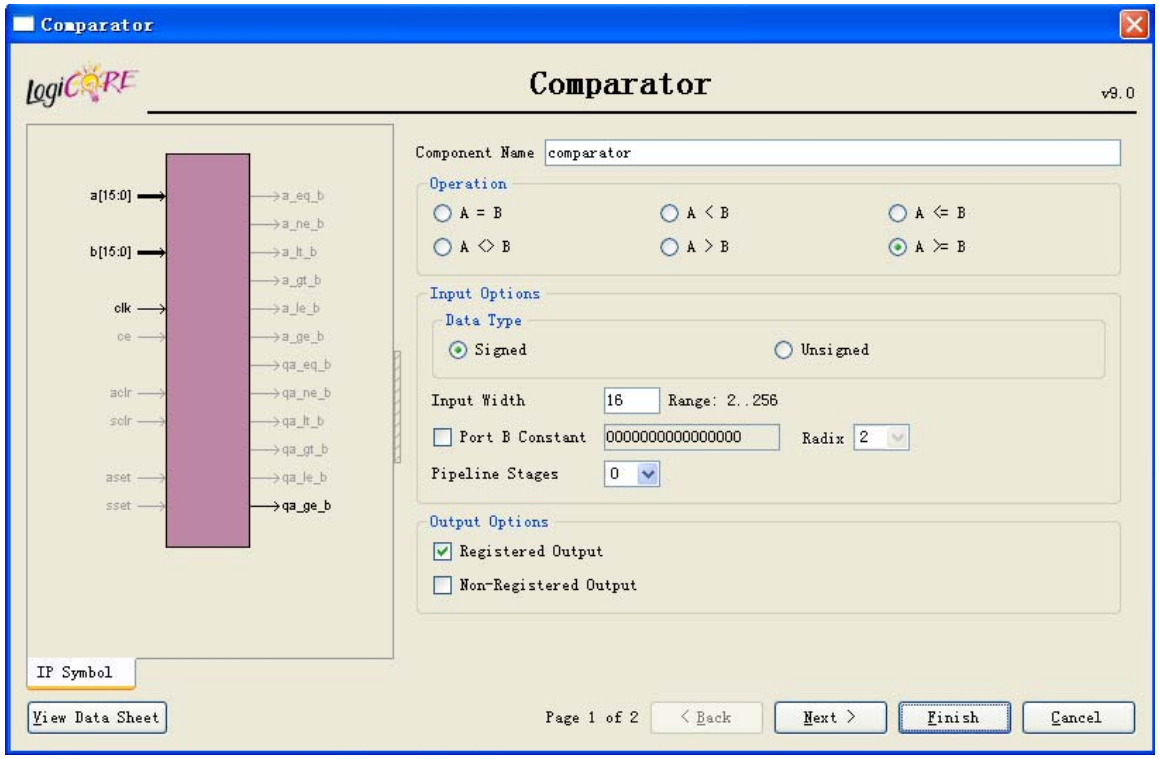

图19 比较器IP核配置页面（1）

- （4）本例配置为“A>=B”，输入数据为有符号数，其位宽为16比特，并且将比较器结果输出寄存。配置完成后，单击“Next”按键进入下一页。比较器输出结果的第二页面为寄存器配置界面。

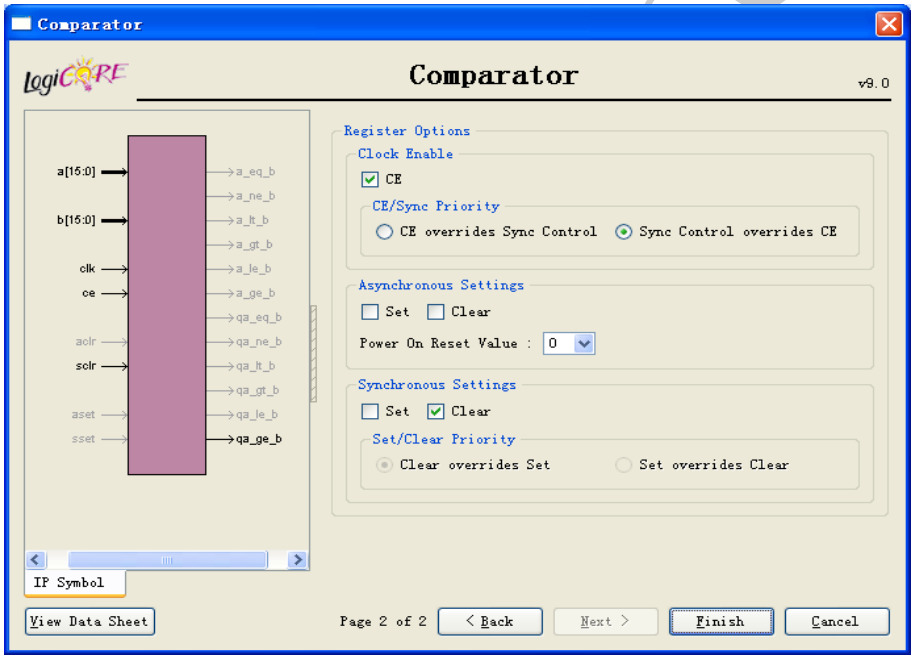

图20 比较器IP核配置页面（2）

4. ISE快速入门

- 配置完成后，单击“Finish”按键，完成IP核的配置，ISE会在输出信息显示区输出各类指示信息，包括IP核生成的指示信息，如下所列。
- Generating IP...
- WARNING:sim:217 - The chosen IP does not support a Verilog behavioral model, generating a Verilog structural model instead.Generating Implementation files.
- Generating ISE symbol file...
- WARNING:coreutil - Default charset GBK not supported, using ISO-8859-1 instea
- Generating NGC file.
- Generating Verilog structural model.
- Finished Generating.

- Successfully generated compare_demo.

- （5）比较器的IP核生成后，可以在工程管理区点击选中生成的比较器IP核，然后双击过程管理区“Core Generator”栏下的“View HDL Functional Model”命令，如图21所示，则可以在源文件 编辑区察看比较器的代码，如下所列。

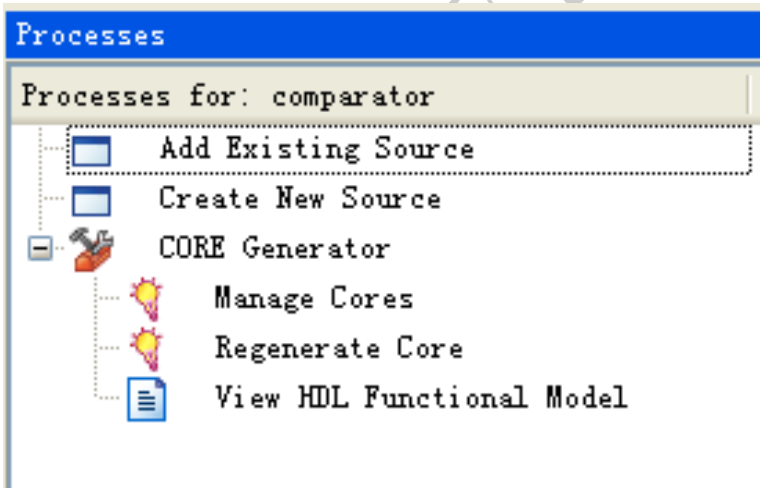

图21 察看比较器IP核源代码的操作示意图

4. ISE快速入门

```
//////////////////////////////////////////////////////////////////////////////
// Copyright (c) 1995-2008 Xilinx, Inc.  All rights reserved.
//////////////////////////////////////////////////////////////////////////////
//   ____  ____
//  /   /\/   /
// /___/  \  /    Vendor: Xilinx
// \   \   \/     Version: K.39
//  \   \         Application: netgen
//  /   /         Filename: comparator.v
// /___/   /\     Timestamp: Mon Jun 21 15:26:52 2010
// \   \  /  \
//  \___\/\___\
//
// Command     : -intstyle ise -w -sim -ofmt verilog F:\labsolutions\comparator\tmp\_cg\comparator.ngc F:\labsolutions\comparator\tmp\_cg\comparator.v
// Device: 3s500efg320-4
// Input file  : F:/labsolutions/comparator/tmp/_cg/comparator.ngc
// Output file : F:/labsolutions/comparator/tmp/_cg/comparator.v
// # of Modules : 1
// Design Name : comparator
// Xilinx      : E:\Xilinx\10.1\ISE
//
// Purpose:
//     This verilog netlist is a verification model and uses simulation
//     primitives which may not represent the true implementation of the
//     device, however the netlist is functionally correct and should not
```

4. ISE快速入门

```
//     be modified. This file cannot be synthesized and should only be used
//     with supported simulation tools.
//
// Reference:
//     Development System Reference Guide, Chapter 23 and Synthesis and Simulation Design Guide, Chapter 6
//
//////////////////////////////////////////////////////////////////////////////////
`timescale 1 ns/1 ps

module comparator (
  sclr, qa_ge_b, ce, clk, a, b
);
  input sclr;
  output qa_ge_b;
  input ce;
  input clk;
  input [15 : 0] a;
  input [15 : 0] b;
  // synthesis translate_off
  // 下面的代码省略
      …
endmodule
```

4. ISE快速入门

（6）在工程中新建mycomparator.v的Verilog HDL源文件，添加以下的代码：

```
module mycomparator(
                        sclr, qa_ge_b, ce, clk, a, b);
                        //声明模块端口
                        input sclr;
                        output qa_ge_b;
                        input ce;
                        input clk;
                        input [15 : 0] a;
                        input [15 : 0] b;

// 例化比较器的IP核
                        comparator inst_comparator(
    .sclr(sclr),
                          .qa_ge_b(qa_ge_b),
                          .ce(ce),
                          .clk(clk),
                          .a(a),
                          .b(b)
    );
endmodule
```

4. ISE快速入门

- 为了测试IP核，还需要在工程中新建一个测试代码（Verilog Test Fixture），命名为tb_mycomparator.v，添加代码如下所列：

```
initial begin
                // Initialize Inputs
                sclr = 0;
                ce = 0;
                clk = 0;
                a = 7;
                b = 20;

                // Wait 100 ns for global reset to finish
                #100;

                // Add stimulus here
                ce = 1;
        #100;
                sclr = 1;
        #100;
                sclr = 0;
```

4. ISE快速入门

```
forever begin
        #5;
        clk=!clk;
        if (clk==1)
                begin
                        a=a+1;
                        b=b-1;
                end
        else
                begin
                        a=a;
                        b=b;
                end
end

end
```

4. ISE快速入门

- 在ISE Simulator中运行上述程序，可以得到图22所示的仿真结果。当sclr为高时，同步清空有效，不管输入如何，比较器输出信号qa_ge_b为0；当sclr为底且ce信号为高时，当a的值大于b时，qa_ge_b为高，否则为0。可以看出，比较器IP核正常工作，完成了A>=B的逻辑判断功能。

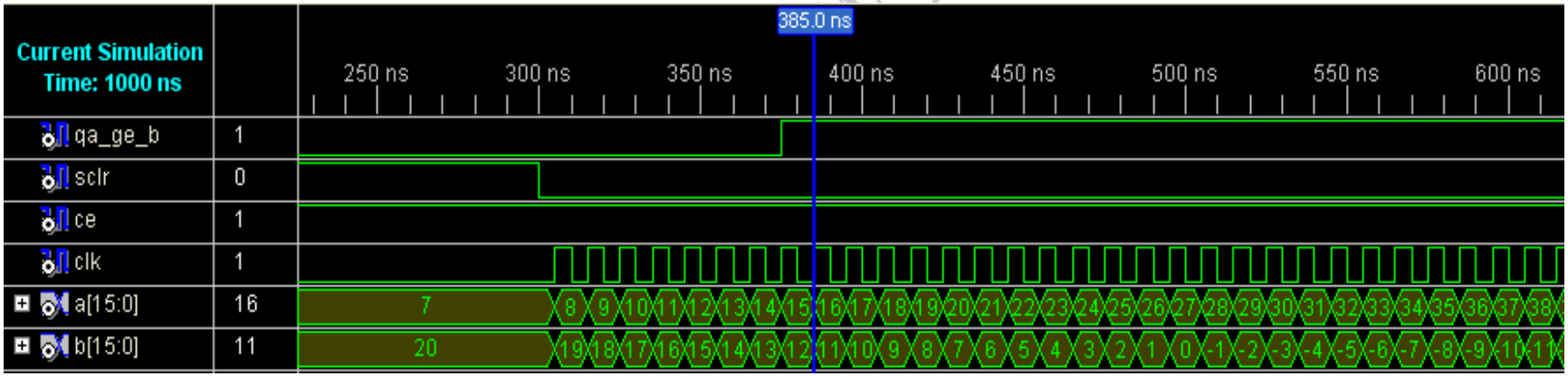

图22 比较器IP核的仿真结果示意图

4.6 用户约束

4.6.1 用户约束输入

- 用户约束是Verilog HDL设计所不可缺少的，包括管脚约束、时序约束以及区域约束等多类约束，对于一般的设计（非高速设计），只有管脚约束是必备的。
- （1）约束文件介绍
- FPGA设计中的约束文件有3类：用户设计文件（.UCF）、网表约束文件（.NCF）以及物理约束文件（.PCF），可以完成时序约束、管脚约束以及区域约束。3类约束文件的关系为：用户在设计输入阶段编写UCF文件，然后UCF文件和设计综合后生成NCF文件，最后再经过实现后生成PCF文件。

4. ISE快速入门

- （2）创建约束文件
- 约束文件的后缀是.ucf，所以一般也被称为UCF文件。创建约束文件的过程为：首先，新建一个源文件，在代码类型中选取“Implementation Constrains File”，在“File Name”中输入约束文件的名称；其次，单击“Next”按键进入模块选择对话框，选择要添加约束的模块，然后单击“Next”进入下一页；最后，单击“Finish”按键完成约束文件的创建。
- （3）编辑约束文件
- 在工程管理区中，将“Source for”设置为“Synthesis/Implementation”，并用鼠标选中约束文件，然后双击过程管理区中“User Constrains”下的“Edit Constraints (Text)”就可以打开约束文件编辑器，
- 如图23所示。

4. ISE快速入门

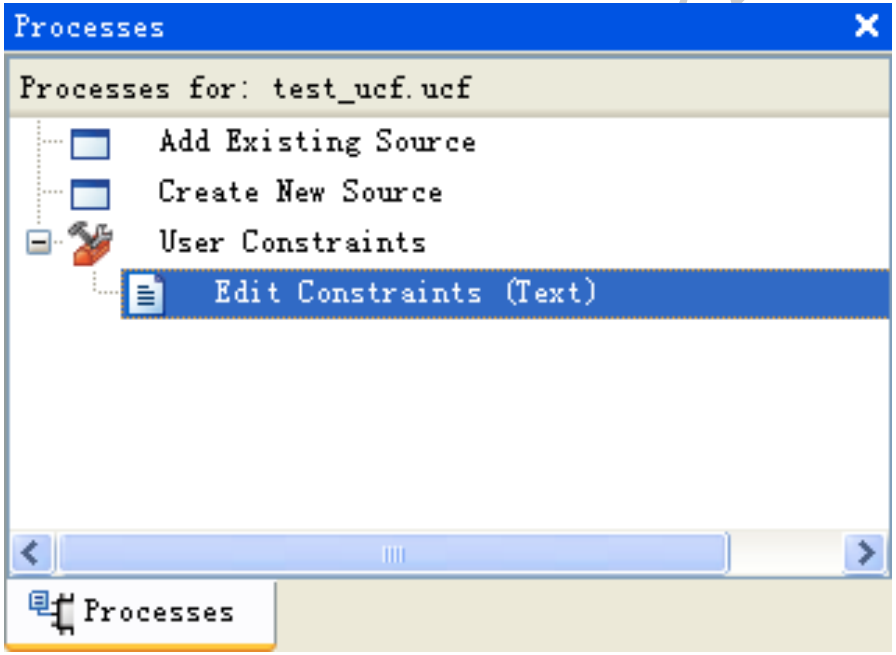

图23 用户约束管理窗口

4.6.2 UCF语法说明

- （1）语法
- UCF文件的语法为：
- {NET|INST|PIN} "signal_name" Attribute;
- 例如：
- NET "CLK" LOC = P30;
- “CLK”就是所约束信号名，“LOC = P30；”是约束具体的含义，将CLK信号分配到FPGA的P30管脚上。
- （2）通配符
- 在UCF文件中，通配符指的是“*”和“?”。“*”可以代表任何字符串以及空，“?”则代表一个字符。

4.6.3 管脚约束

- 使用LOC完成端口定义时，其语法如下：
- NET "Top_Module_PORT" LOC = "Chip_Port";
- 其中，“Top_Module_PORT”为用户设计中顶层模块的信号端口，“Chip_Port”为FPGA芯片的管脚名。“LOC”约束是FPGA设计中最基本的布局约束和综合约束，能够定义基本设计单元在芯片中的位置，可实现绝对定位、范围定位以及区域定位。
- LOC语句中是存在优先级的，当同时指定LOC端口和其端口连线时，对其连线约束的优先级是最高的。

4. ISE快速入门

- LOC语句通过加载不同的属性可以约束管脚位置、CLB、Slice、TBUF、块RAM、硬核乘法器、全局时钟、数字锁相环（DLL）以及DCM模块等资源，基本涵盖了FPGA芯片中所有类型的资源。由此可见，LOC语句功能十分强大，表2列出了用于管脚约束的LOC属性。

约束类型	可用属性示例	属性含义
IO管脚约束	P12	将信号置于由芯片引脚号定位的端口上
	A12	将信号置于由芯片引脚阵列号定位的端口上
	B, L, T, R	将信号定位到芯片特定边界中（从物理位置上划分的上、下、左、右4部分）的端口上。
	LB, RB, LT, RT, BR, TR, BL, TL	将信号定位到芯片特定边界上的一半（从物理位置上划分的上左 、上右 、下左、下右、左上、坐下、右上以及右下8部分）位置中的端口上。
	Bank0, Bank1, Bank2, Bank3, ...	将信号置于特定管脚分组中的端口上。

4.7 综合与实现

4.7.1 综合

- 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NGC、NCR以及LOG文件，如图24所示。

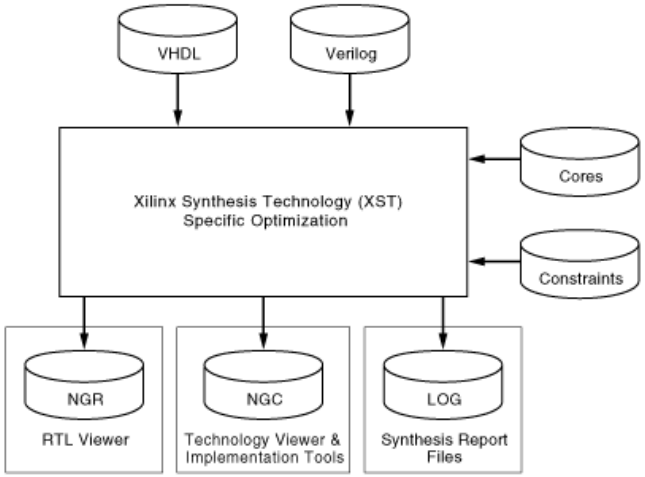

图24 综合工具XST的功能示意图

4. ISE快速入门

- 完成了输入和仿真后就可以进行综合了。在过程管理区双击Synthesize-XST即可开始综合过程，如图25所示。

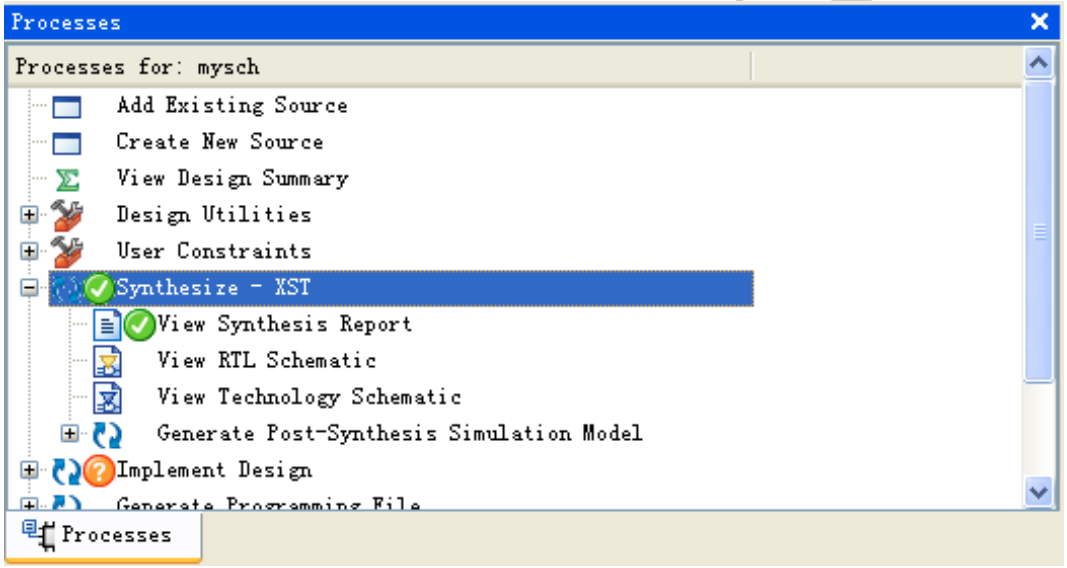

图25 综合操作窗口示意图

4. ISE快速入门

- 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。

- 如果综合步骤没有语法错误，XST能够给出初步的资源消耗情况，点击过程管理区的“View Design Summary”，即可查看，如图26所示。

4. ISE快速入门

mycounter Project Status (06/21/2010 - 16:50:52)			
Project File:	mycounter.ise	Current State:	Synthesized
Module Name:	mycounter	• Errors:	No Errors
Target Device:	xc3s500e-4fg320	• Warnings:	No Warnings
Product Version:	ISE 10.1.03 - Foundation Simulator	• Routing Results:	
Design Goal:	Balanced	• Timing Constraints:	
Design Strategy:	Xilinx Default (unlocked)	• Final Timing Score:	

mycounter Partition Summary	[-]
No partition information was found.	

Device Utilization Summary (estimated values)			[-]
Logic Utilization	Used	Available	Utilization
Number of Slices	4	4656	0%
Number of Slice Flip Flops	8	9312	0%
Number of 4 input LUTs	9	9312	0%
Number of bonded IOBs	10	232	4%
Number of GCLKs	1	24	4%

图26　综合结果报告

- 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。对于例1，所得到的RTL结构图如图27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。

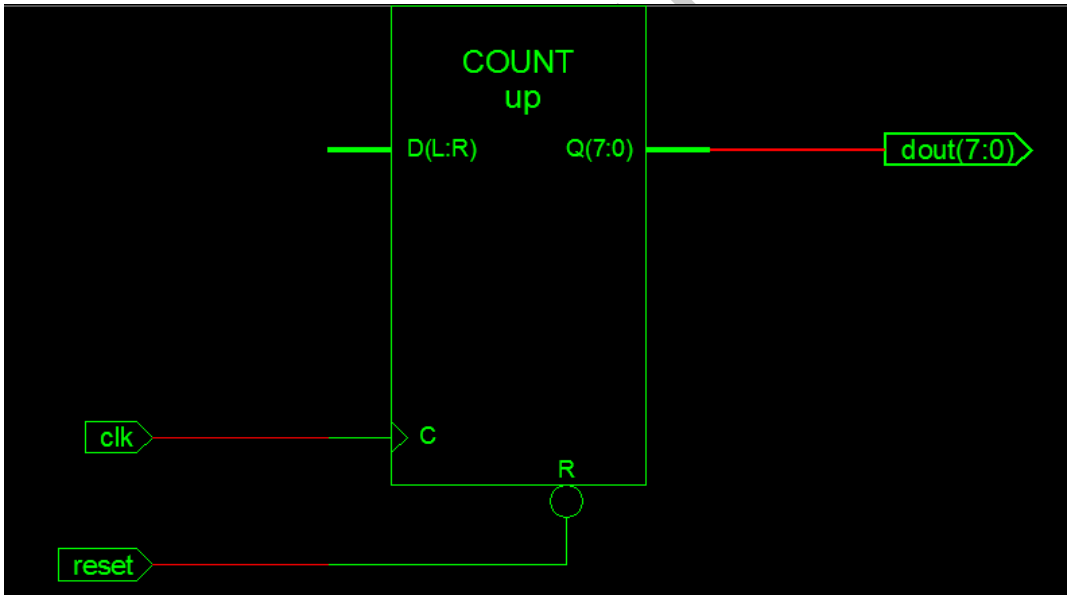

图27 例2-2的综合后的RTL结构示意图

4.7.2 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）。在ISE中，执行实现过程，会自动执行翻译、映射和布局布线过程；也可以单独执行。

- 翻译的主要作用是将综合输出的逻辑网表翻译为Xilinx特定器件的底层结构和硬件原语。映射的主要作用是将设计映射到具体型号的器件上（LUT、FF、Carry等）。布局布线步骤调用Xilinx布局布线器，根据用户约束和物理约束，对设计模块进行实际的布局，并根据设计连接，对布局后的模块进行布线，产生FPGA/CPLD配置文件。

4. ISE快速入门

- 在过程管理区双击“Implement Design”选项，就可以自动完成实现的3个步骤，如图28所示。如果设计没有经过综合，会启动XST完成综合，在综合后再完成实现过程。

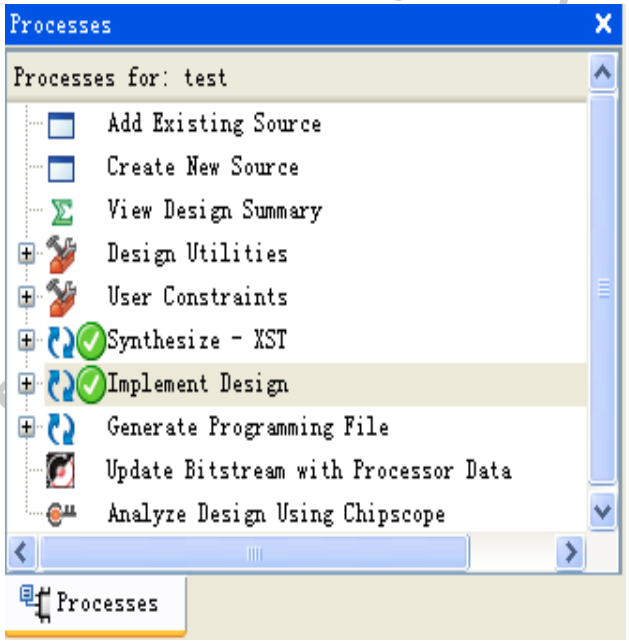

图28 设计实现窗口

4. ISE快速入门

- 经过实现后能够得到精确的资源占用情况，以例1的计数器设计为例，双击“Place & Router”下的“Place & Route Report”即可查阅最终的资源报告，如图29所示。

```
Design Summary Report:

 Number of External IOBs                          10 out of 232     4%

   Number of External Input IOBs                  2

      Number of External Input IBUFs              2

   Number of External Output IOBs                 8

      Number of External Output IOBs              8

   Number of External Bidir IOBs                  0


   Number of BUFGMUXs                        1 out of 24      4%
   Number of Slices                          4 out of 4656    1%
      Number of SLICEMs                      0 out of 2328    0%
```

图29 实现后的资源统计结果

4.8 器件配置

4.8.1 FPGA配置

- FPGA配置主要用于调试阶段快速、多次验证功能，断电后芯片内的逻辑立刻消失，每次上电都需要重新配置。该操作比较简单，首先，在配置启动参数中选择配置时钟为JTAG CLK，否则会产生警告，配置过程容易出错；其次，只需在过程管理区中双击Generate Programming File选项即可完成，完成后则该选项前面会出现一个打钩的圆圈，如图30所示。

- 并弹出图31所示的报告信息对话框，直接关掉即可。

4. ISE快速入门

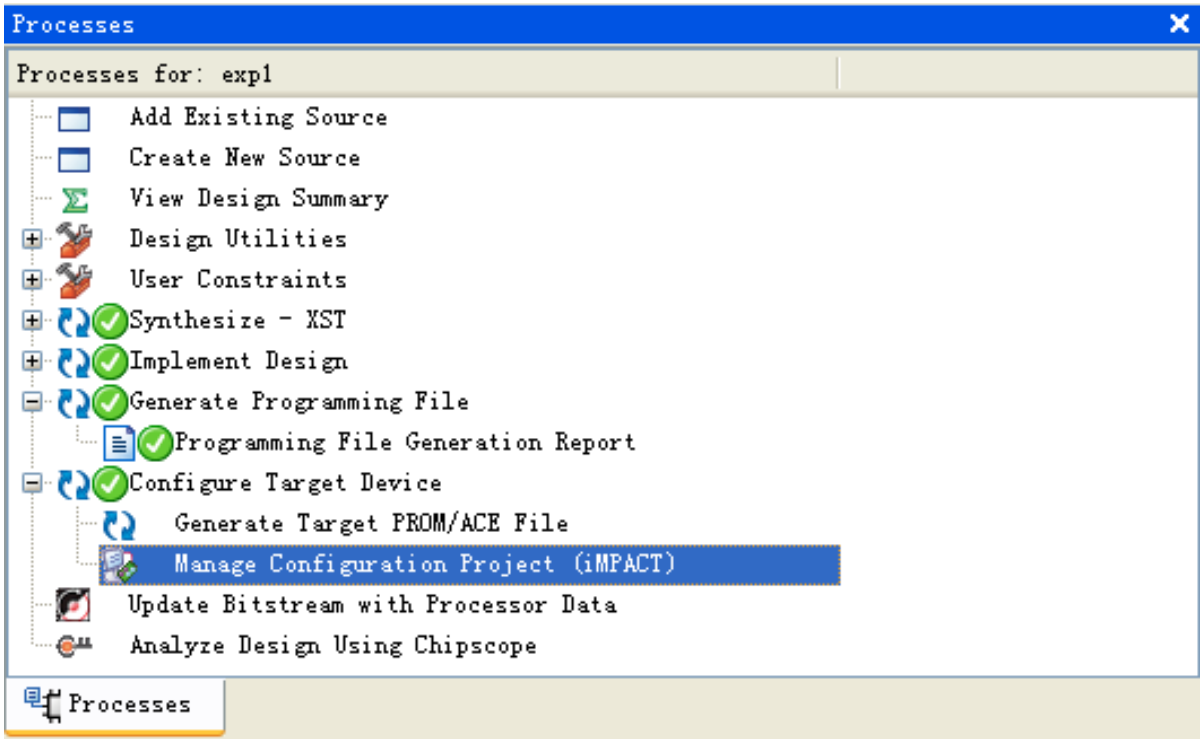

图30　生成编程文件的窗口

4. ISE快速入门

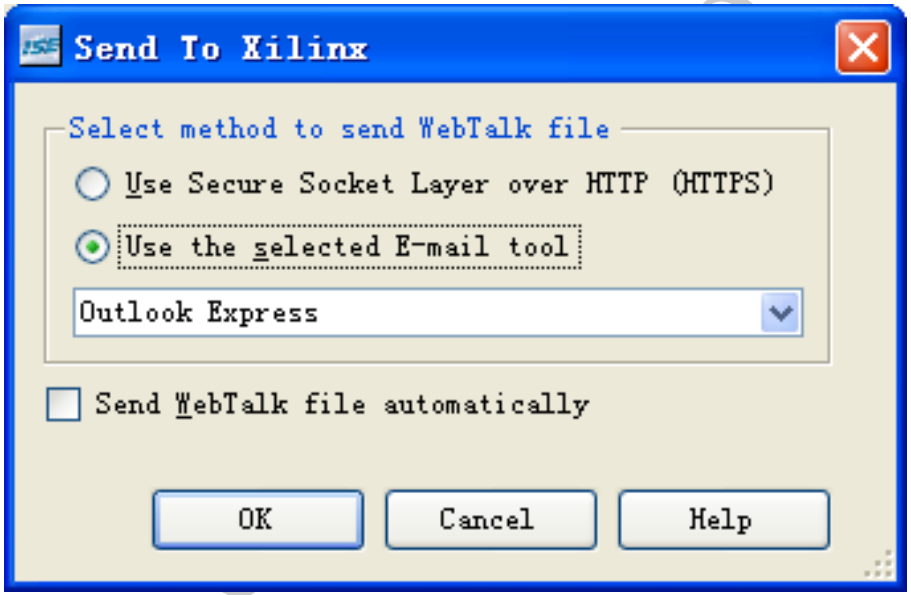

图31 比特文件报告对话框

- 双击过程管理区的“Configure Target Device”栏目下的“Manage Configure Project (iMPACT)”，然后在弹出的iMPACT配置对话框中选取“Configure device using Boundary-Scan (JTAG)”，如图32所示。

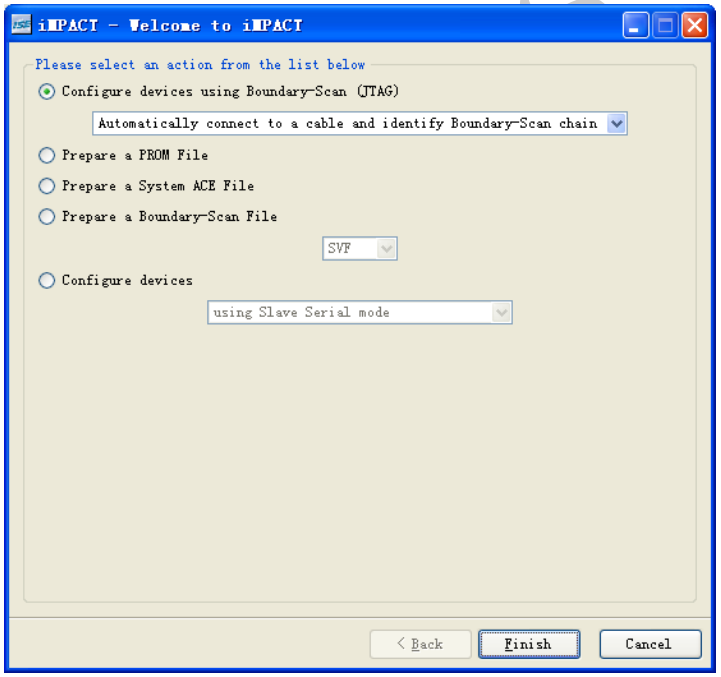

图32 iMPACT配置对话框

4. ISE快速入门

- 点击“OK”按键后，ISE会自动连接FPGA设备。成功检测到设备后，会显示出JTAG链上所有芯片，其典型示意如图33所示。从中可以看出，链上的所有芯片构成了一个TDI到TDO的完整回路，这是电路设计时必须要保证的。

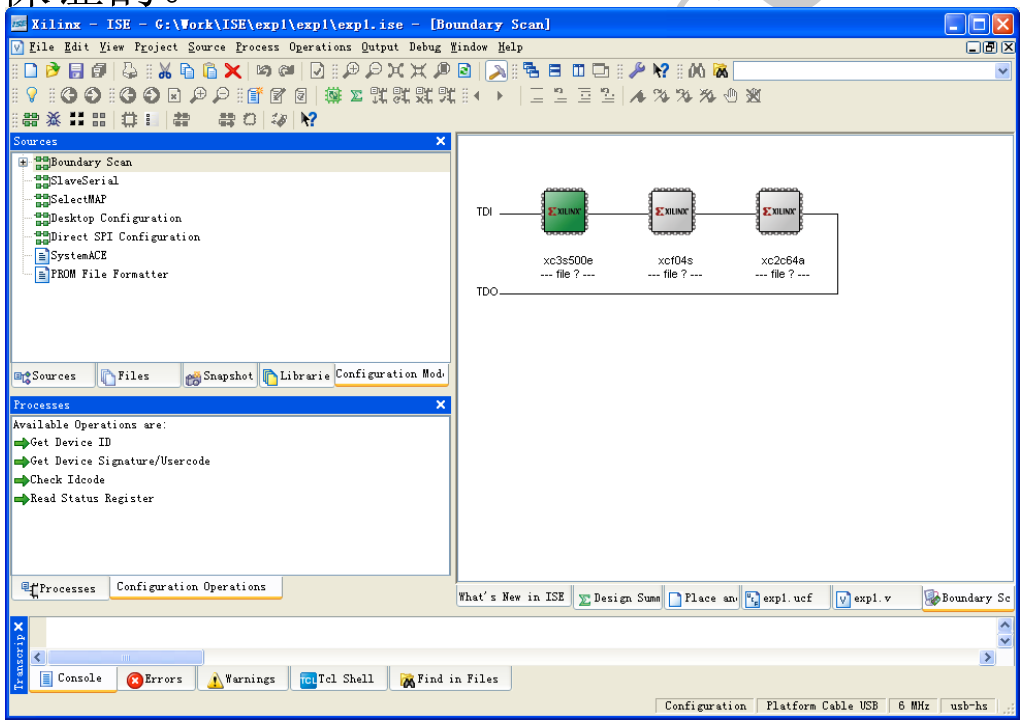

图33 JTAG链边界扫描结果示意图

- 如果JTAG链检测失败，其弹出的对话框如图34所示。在配置FPGA器件时，经常会出现扫描失败的情况，常用的解决方法有：首先，检查电源是否正确；其次，采用质量更好的并口配置电缆（Parallel Cable-IV）或信号质量更好的USB配置电缆，排除下载线的问题；第三，检查所有芯片的TCK、TMS管脚是否和JTAG接口的TCK、TMS连接在一起；最后，检查配置电路的JTAG链路是否完整，从JTAG接口的TDI到链首芯片的TDO、……、再到链尾芯片的TDO是否连接到JTAG接口的TDO。

图34 JTAG链扫描失败对话框

- JTAG链检测正确后，在FPGA芯片的图标上双击，或点击右键并在弹出的菜单中选择“Assign New Configuration File”，会弹出图35的窗口，让用户选择后缀为.bit的二进制比特流文件。

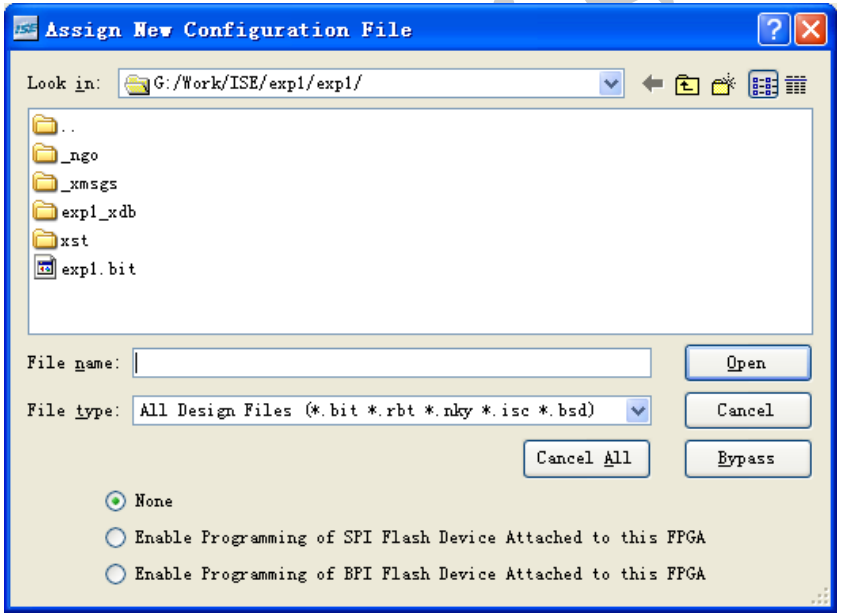

图35 选择位流文件

4. ISE快速入门

- 在FPGA芯片标志上单击右键，在弹出的对话框中选择Program选项，就可以对FPGA设备进行编程，如图36所示。

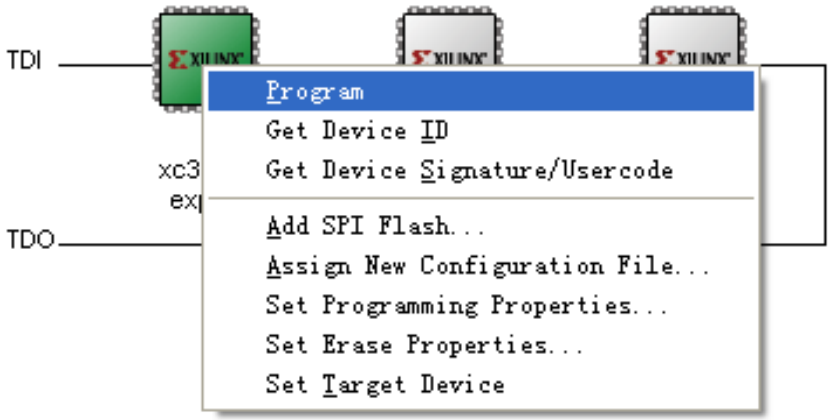

图36 对FPGA设备进行编程示意图

4. ISE快速入门

- 配置成功后，会弹出配置成功的界面，如图37所示。

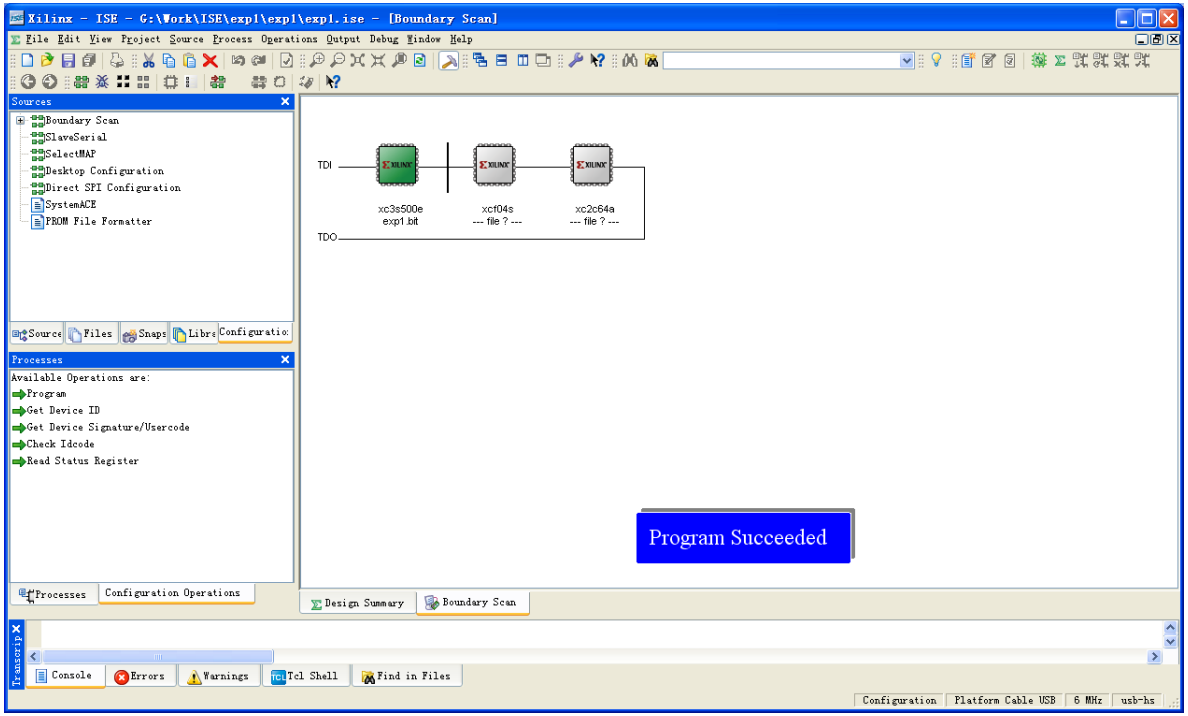

图37 FPGA配置成功指示界面

4.8.2 PROM配置

- 一个设计经过综合、实现之后，需要为不同器件生成不同类型的编程文件。ISE中内嵌了比特流生成器，可生成FPGA以及PROM格式文件，从而实现动态配置，并验证数据是否正确。

- （1）将设计经过前仿、综合、实现以及后仿，确保设计无误。双击过程管理区的“Configure Target Device”栏目下的“Manage Configure Project (iMPACT)”，然后在弹出的iMPACT配置对话框中选取“Prepare a PROM File”，如图38所示。

4. ISE快速入门

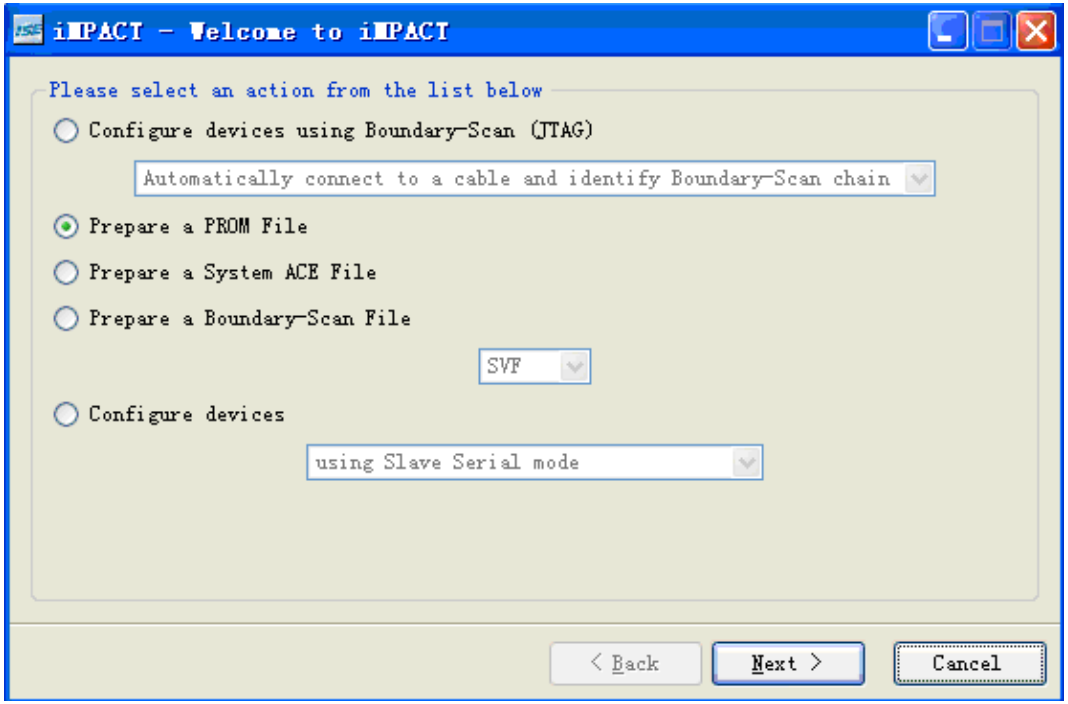

图38 iMPACT配置对话框

- （2）单击“Next”按钮，进入PROM器件选择界面，如图39所示。下面以Xilinx PROM为例进行说明。

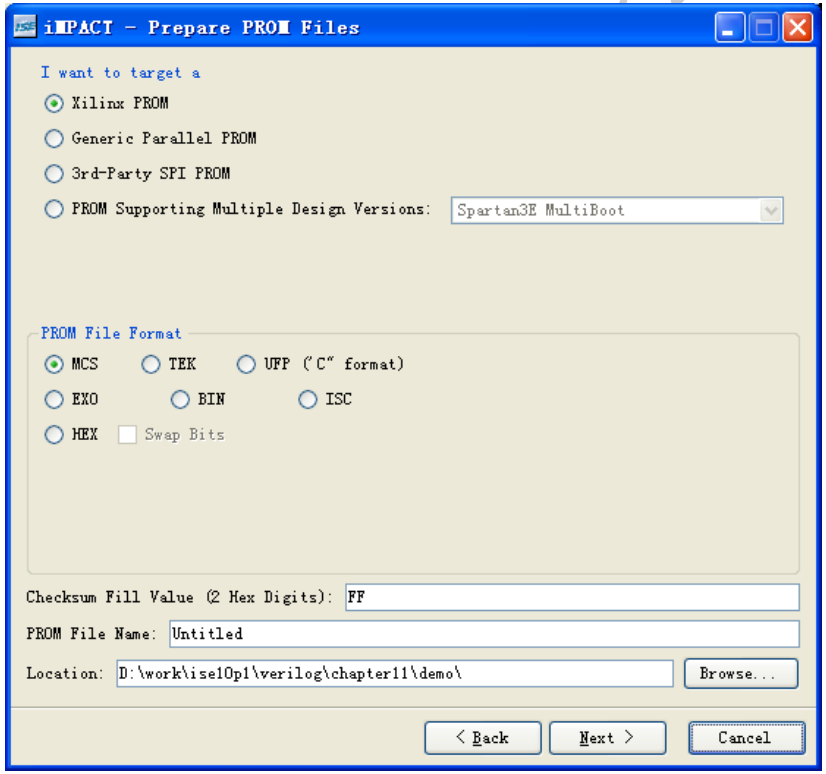

图39 选择PROM芯片的类型和文件格式

- （3）单击“Next”按钮，进入PROM模式配置界面，如图40所示，支持单片串行、并行以及多片PROM的级联配置，这里选择串行，因为XCF04S只支持串行配置。

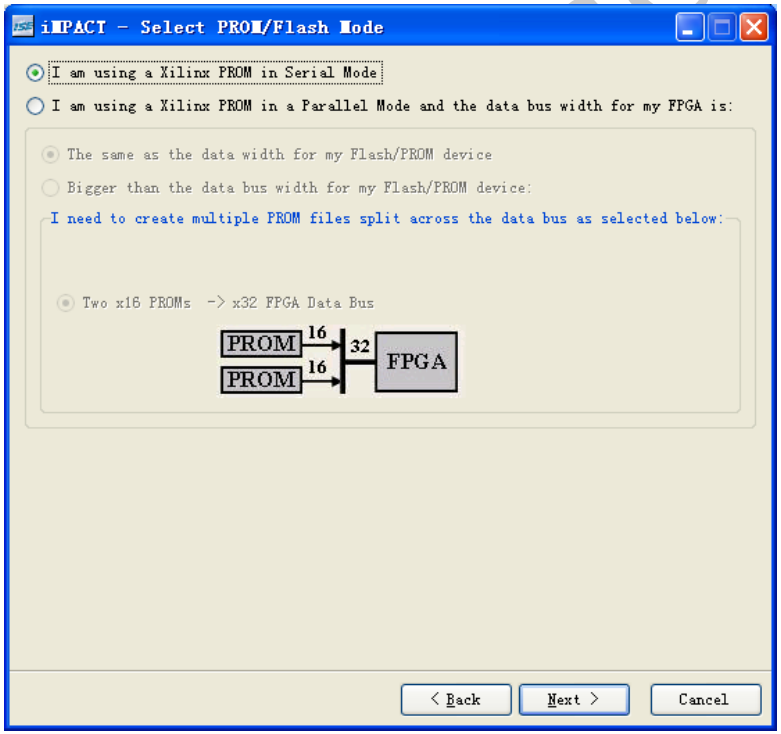

图40 PROM芯片配置模式选择界面

- （4）点击“Next”按钮，选择PROM器件的型号，如图41所示。

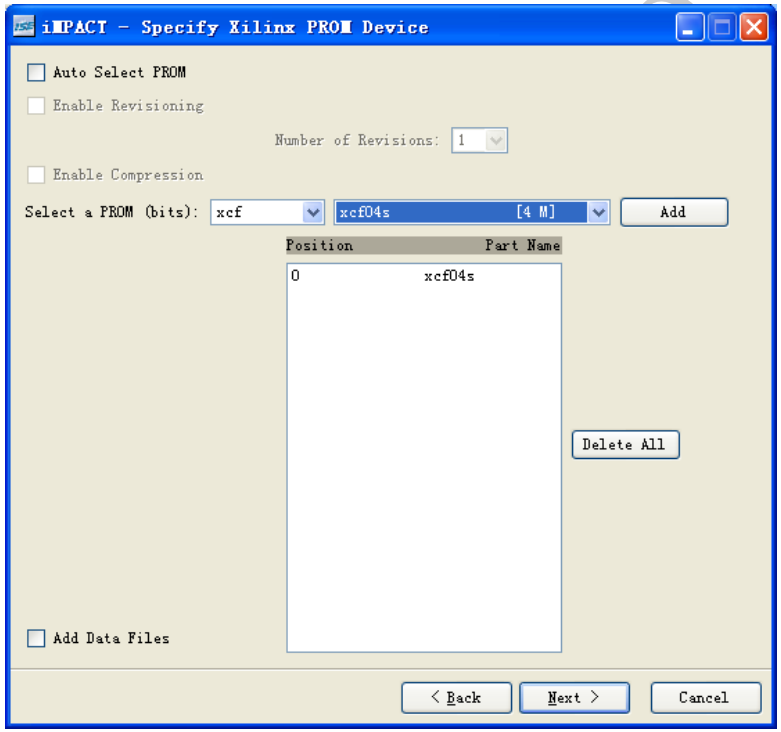

图41 PROM芯片型号选择界面

- （5）单击图41中的“Next”按键，进入PROM文件综合信息显示窗口，如图42所示。

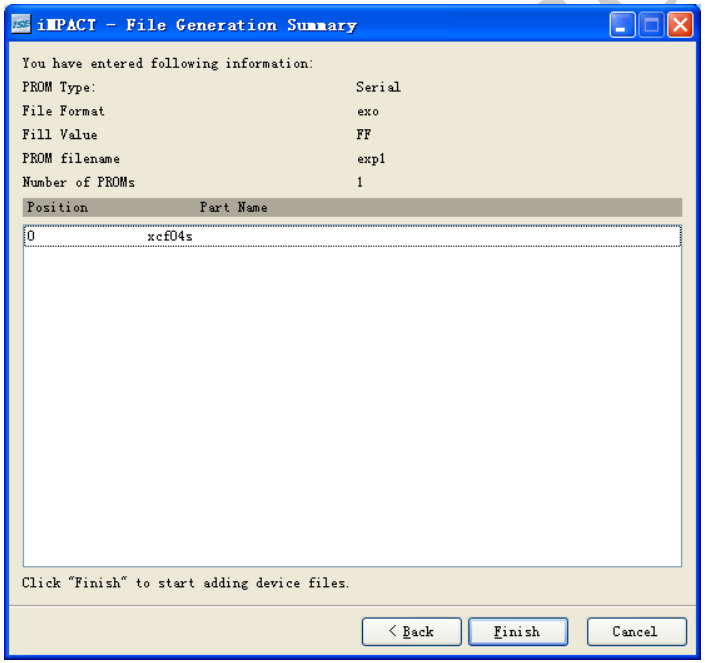

图42 PROM芯片综合信息显示界面

- （6）单击图42中的“Finish”按键，弹出的比特文件加载窗口如图43所示。

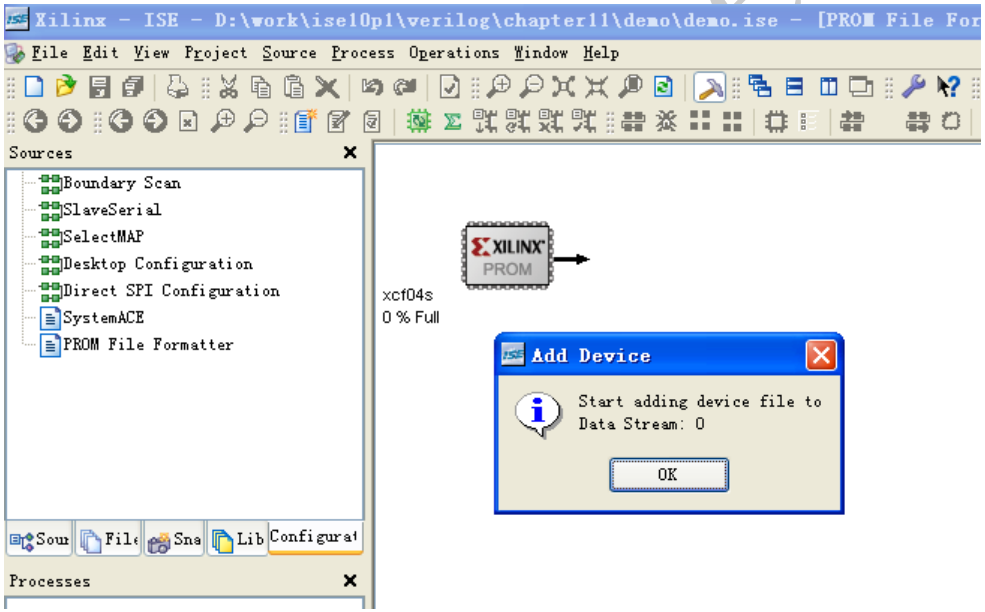

图43 比特文件加载窗口

- 点击图43中“Add Device”对话框的“OK”按键，可打开.bit文件选择界面，如图44所示。

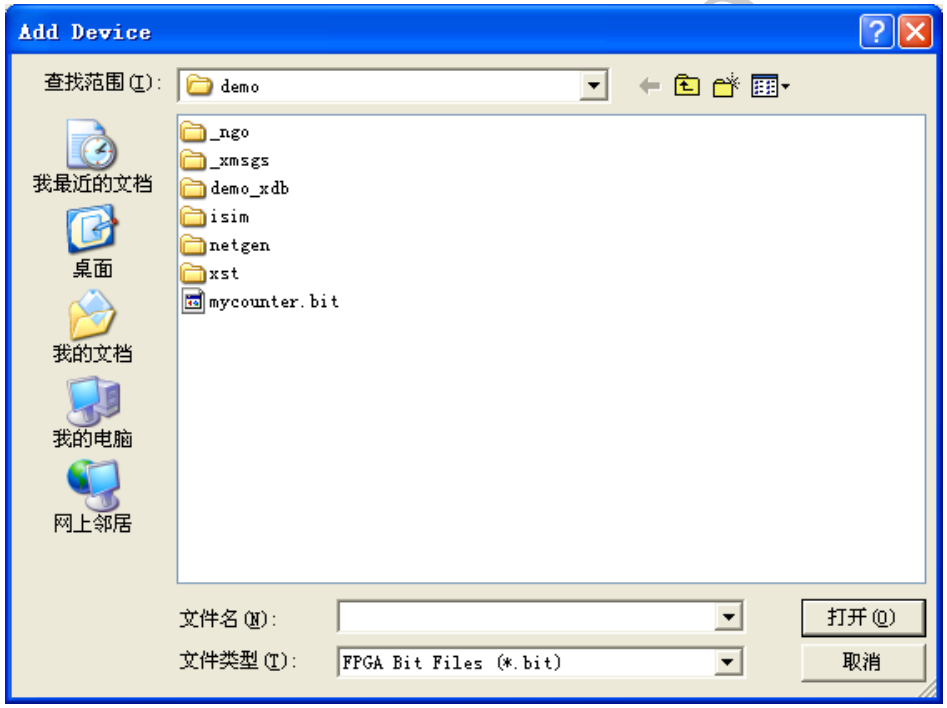

图44 比特文件选择界面

- 选择加载的比特文件后，单击图44中的“打开”按键后，iMPACT会提示用户是否再添加比特文件，如图45所示。这是因为只要容量允许，一片PROM可配置多个FPGA。

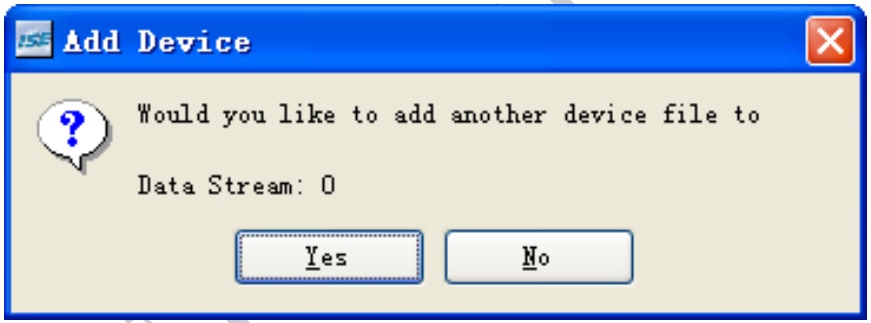

图45 比特文件选择界面图

- （7）此时，iMPACT会根据加载的bit文件所对应的FPGA芯片计算PROM的容量，如果PROM容量不够，会主动提醒用户修改PROM型号或者添加更多的PROM芯片；如果容量富裕，则会给出PROM的容量利用率，如图46所示。

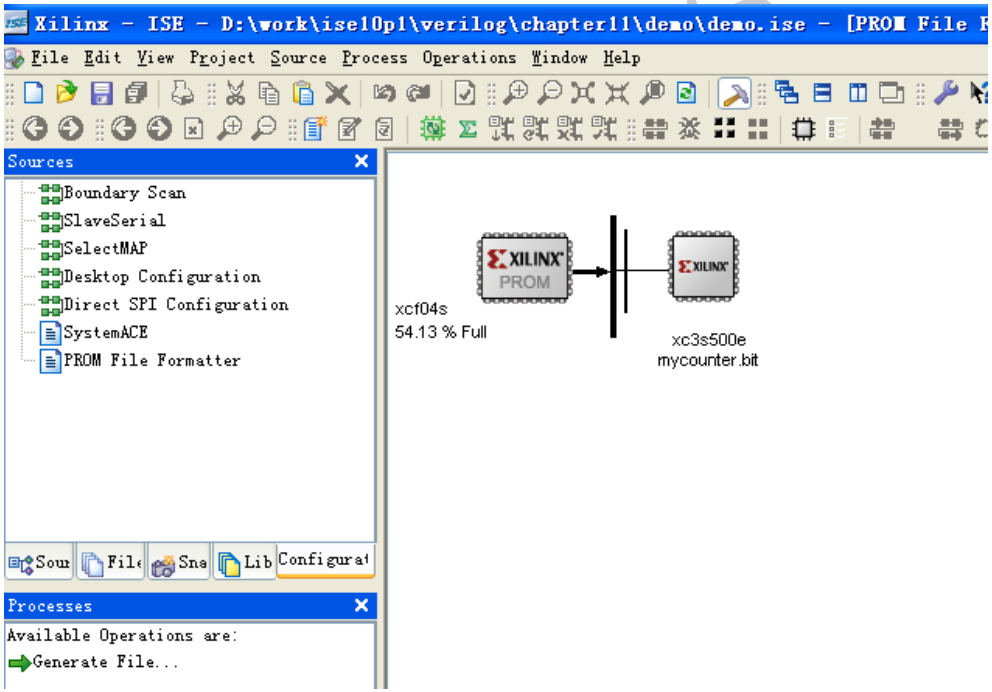

图46 PROM容量显示界面

- 在iMPACT的过程管理窗口，双击“Generate File”，iMPACT会自动创建PROM配置文件，并在iMPACT界面上显示“PROM File Generation Succeeded”，如图47所示。

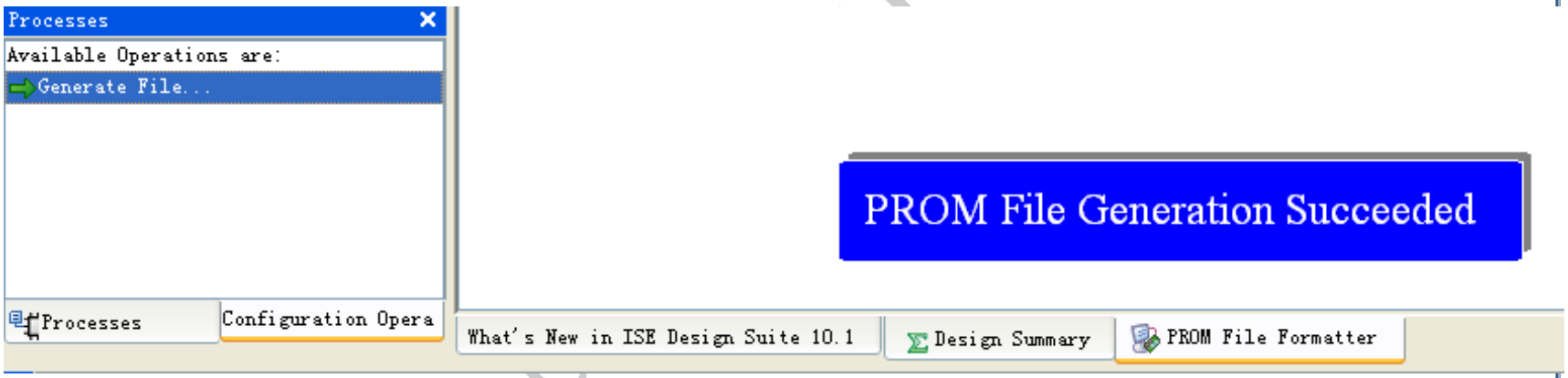

图47 PROM配置文件创建成功提示界面

- （8）单击工具栏的“”按键，初始化JTAG链，并在PROM芯片上双击，可弹出PROM配置文件选择界面，选择刚才生成的MCS文件。这一过程和为FPGA芯片选择bit文件是一致的。
- （9）在PROM芯片上点击右键，选择“Pragram”命令，会弹出图48所示的对话框。

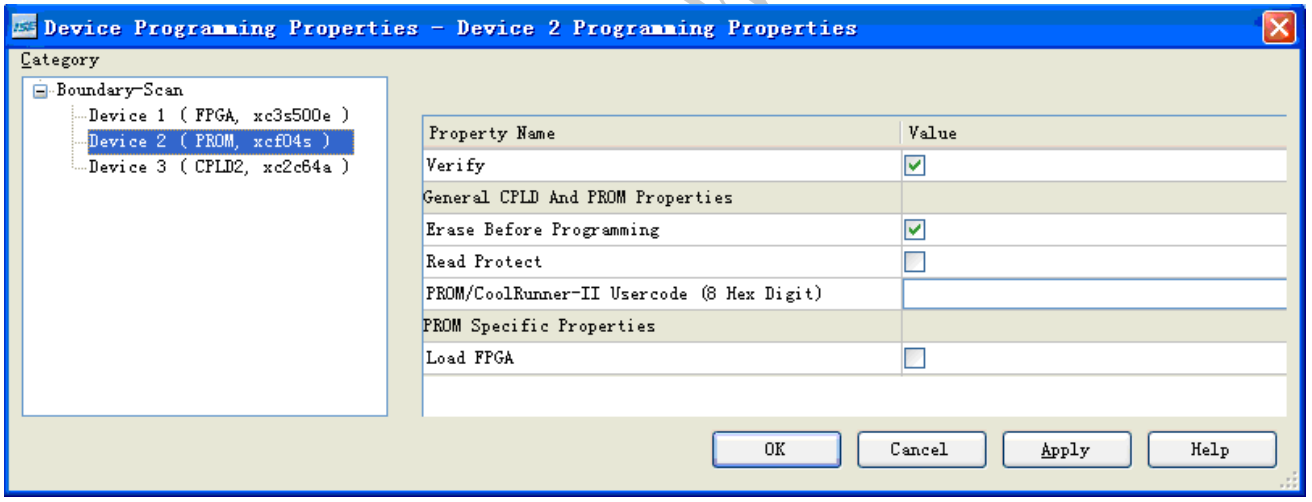

图48 PROM配置设置对话框

4. ISE快速入门

- 点击图48中的“OK”按键，即开始配置PROM芯片，并弹出图49所示的进度条。

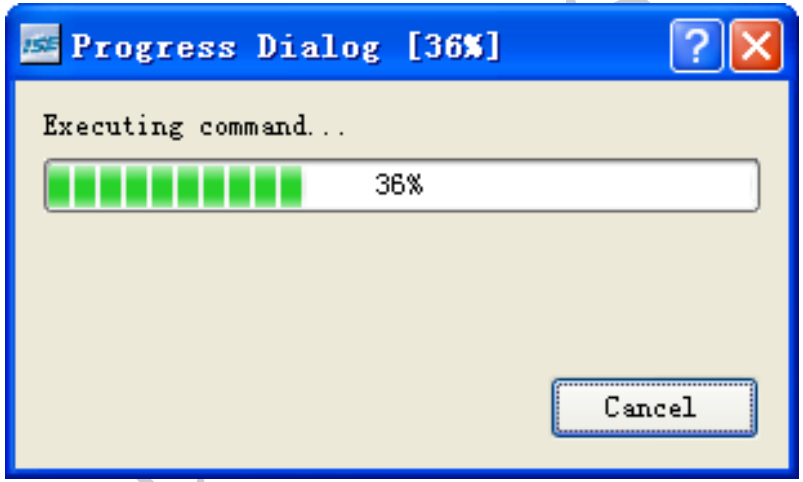

图49 PROM配置进度条

4. ISE快速入门

- 进度条达到100%后，完成PROM配置后，iMPACT给显示“Program Succeeded”，如图50所示。

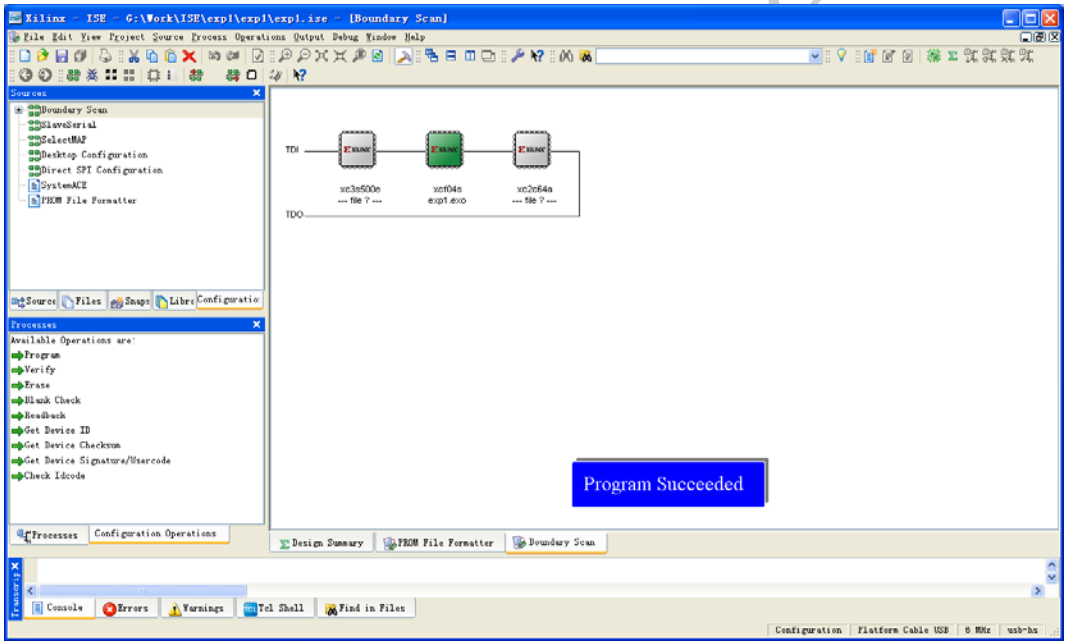

图50 PROM配置成功示意图

- 至此，就完成了一个完整的FPGA设计流程。