

Quartus Prime 是 Intel 公司的综合性 PLD 开发软件,支持原理图、VHDL、VerilogHDL 以及 AHDL(Altera Hardware Description Language)等多种设计输入形式,内嵌自有的综合器以及仿真器,可以完成从设计输入到硬件配置的完整 PLD 设计流程。

Quartus Prime 可以在 Windows 和 Linux 上使用,除了可以使用 Tcl 脚本完成设计流程外,还提供了完善的用户图形界面设计方式。具有运行速度快、界面统一、功能集中、易学易用等特点。

Quartus Prime 支持 Intel 公司的 IP 核,包含 LPM/MegaFunction 宏功能模块库,使用户可以充分利用成熟的模块,简化了设计的复杂性,加快了设计速度。对第三方 EDA 工具的良好支持也使用户可以在设计流程的各个阶段使用熟悉的第三方 EDA 工具。

此外,Quartus Prime 通过和 DSP Builder 工具与 Matlab/Simulink 相结合,可以方便地实现各种 DSP 应用系统;支持 Intel 公司的片上可编程系统(SOPC)开发,集系统级设计、嵌入式软件开发、可编程逻辑设计于一体,是一种综合性的开发平台。

Maxplus II 作为 Intel 公司的上一代 PLD 设计软件,由于其出色的易用性而得到了广泛的应用。该软件可以支持 ACEX1K、FLEX10KE、FLEX10KB、FLEX10KA、FLEX10K、MAX 9000、FLEX 8000、MAX 7000、FLEX 6000、MAX 5000 及 Classic 等各种类型的可编程器件。可以采用图形输入、波形编辑输入、AHDL 语言输入、VHDL 语言输入、Verilog 语言输入,可以完成电路设计的功能分析、时序分析。目前 Intel 公司已经停止了对 Maxplus II 的更新支持,Quartus Prime 与之相比不仅仅是支持器件类型的丰富和图形界面的改变。Intel 公司在 Quartus Prime 中包含许多诸如 SignalTap II、Chip Editor 和 RTL Viewer 的设计辅助工具,集成了 Qsys 设计流程,并且继承了 Maxplus II 友好的图形界面及简便的使用方法。

Quartus Prime 作为一种可编程逻辑的设计环境,由于其强大的设计能力和直观易用的接口,越来越受到数字系统设计者的欢迎。Intel 公司的 Quartus Prime 可编程逻辑软件属于第四代 PLD 开发平台。该平台支持一个工作组环境下的设计要求,其中包括支持基于 Internet 的协作设计。Quartus Prime 平台与 Cadence、ExemplarLogic、MentorGraphics、Synopsys 和 Synplicity 等 EDA 供应商的开发工具相兼容。改进了软件的 LogicLock 模块设计功能,增添了 FastFit 编译选项,推进了网络编辑性能,而且提升了调试能力。

Quartus Prime 目前各类版本可以在 Intel 公司官网上下载,版本号从 2.2 开始到 19.3 版本。各版本软件之间的差异如下。

(1) Quartus Prime 9.1 之前的软件自带仿真组件,而之后软件不再包含此组件,因此

必须安装 ModelSim。

(2) Quartus Prime 9.1 之前的软件自带硬件库,不需要额外下载安装,而 10.0 开始需要额外下载硬件库,另行选择安装。

(3) Quartus Prime 11.0 之前的软件需要额外下载 Nios II 组件,而 11.0 开始 Quartus Prime 软件自带 Nios II 组件。

(4) Quartus Prime 9.1 之前的软件自带 SOPC 组件,而 Quartus Prime 10.0 自带 SOPC 和 Qsys 两个组件,但从 10.1 开始,Quartus Prime 只包含 Qsys 组件。

(5) Quartus Prime 10.1 之前软件,时序分析包含 TimeQuest Timing Analyzer 和 Classic Timing Analyzer 两种分析器,但 10.1 以后的版本只包含 TimeQuest Time Analyzer,因此需要 sdc 来约束时序。

(6) 中文支持方面: Quartus Prime 8.0 以前的版本,可以输入中文也可以显示中文;  $8.0 \leq \text{Quartus Prime 版本} < 9.1$ ,可以显示中文,但是不能输入中文;  $9.1 \leq \text{Quartus Prime 版本} < 11$ ,不能输入中文,同时也不可以显示中文; Quartus Prime 19.3 是目前最新版本,可以显示中文字符,同时又能也能输入中文。

目前从官网上下下载的开发软件有三种:精简版、标准版(收费)和专业版(收费),下面对三种版本简单进行对比,用户可根据情况自行选择。具体如表 3.1 所示。

表 3.1 Quartus Prime 开发软件精简版、标准版和专业版比较

Quartus Prime 关键特性		精简版	标准版	专业版	
器件支持	Stratix 系列	II ~ V	√		
	Arria 系列	II	仅支持 EP2AGX45		
		II ~ V		√	
		10		√	√
	Cyclone 系列	II ~ V	√	√	
		10LP	√	√	
		10GX			√
MAX 系列		√	√		
设计输入	多处理器支持(编译速度更快)		√	√	
	IP 基础套件	需要购买	√	√	
	Intel Qsys 系统集成工具	√	√		
	Intel Qsys Pro 系统集成工具			√	
	快速重新编译		适用于 Stratix V、Arria V 和 Cyclone V 器件	√	
	蓝图平台设计工具			√	
功能模拟	ModelSim——Intel FPGA 初级版软件	√	√	√	
	ModelSim——Intel FPGA 版软件	需要额外的许可证			
综合	支持设计可移植性的行业标准语			√	
布局布线	Fitter(布局和布线)	√	√	√	
	增量优化			√	
	混合布局工具		适用于 Arria 10、Stratix V、Arria V 和 Cyclone V 器件	√	

续表

Quartus Prime 关键特性		精简版	标准版	专业版
设计流	局部重配置		仅适用于 Cyclone V 和 Stratix V 器件	✓
时序和功耗验证	TimeQuest 静态时序分析工具	✓	✓	✓
	功耗分析	✓	✓	✓
系统内调试	SignalTap II 逻辑分析工具	✓	✓	✓
	收发器工具包		✓	✓
	JNEye 链接分析工具		✓	✓
操作系统支持	Windows/Linux 64 位支持	✓	✓	✓
插件开发工具	面向 OpenCL 的 Intel FPGA SDK	需要额外的许可证		
	适用于 Intel FPGA 的 DSP Builder	需要额外的许可证		
	Nios II 嵌入式设计套件	✓	✓	✓
	Intel 系统芯片 FPGA 嵌入式设计套件	✓	✓	✓

Quartus Prime 设计软件提供完整的多平台设计环境,能够直接满足特定设计需要,为可编程芯片系统(SOPC) 提供全面的设计环境。Quartus Prime 软件含有 FPGA 和 CPLD 设计所有阶段的解决方案。有关 Quartus Prime 设计流程的图示说明,见图 3.1。

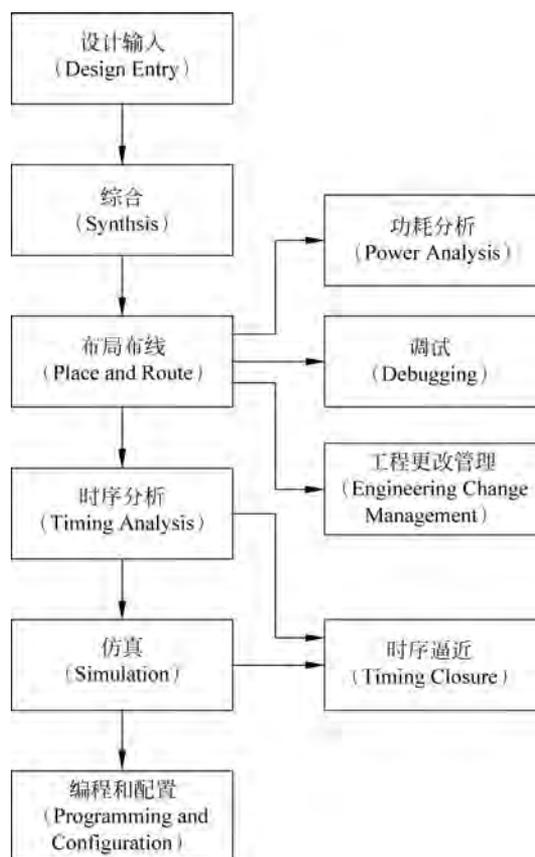


图 3.1 Quartus Prime 设计流程

此外,Quartus Prime 软件为设计流程的每个阶段提供 Quartus Prime 图形用户界面、EDA 工具界面以及命令行界面,可以在整个流程中只使用这些界面中的一个,也可以在设计流程的不同阶段使用不同界面。

### 3.1 使用 Quartus Prime 进行图形化设计

使用 Quartus Prime 图形用户界面的基本设计流程如下:

- (1) 使用 New Project Wizard(File 菜单) 建立新工程并指定目标器件或器件系列。
- (2) 使用 Text Editor 建立 Verilog HDL、VHDL 或 Altera 硬件描述语言(AHDL) 设计。根据需要,使用 Block Editor 建立表示其他设计文件的符号框图,也可以建立原理图。还可以使用 MegaWizard Plug-In Manager(Tools 菜单)生成宏功能模块和 IP 功能的自定义变量,在设计中将它们例化。
- (3) (可选)使用 Assignment Editor、Pin Planner、Settings 对话框(Assignments 菜单)、Floorplan Editor、Design Partitions 窗口、LogicLock 功能指定初始设计约束。
- (4) (可选)进行 Early Timing Estimate,在完成 Fitter 之前生成时序结果的早期估算。
- (5) (可选)使用 SOPC Builder 或 DSP Builder 建立系统级设计。
- (6) (可选)使用 Software Builder 为 Excalibur 器件处理器或 Nios 嵌入式处理器建立软件和编程文件。
- (7) 使用 Analysis & Synthesis 对设计进行综合。
- (8) (可选)如果设计含有分区,而没有进行完整编译,则需要采用 Partition Merge 合并分区。
- (9) (可选)通过使用 Simulator 和 Generate Functional Simulation Netlist 命令在设计中执行功能仿真。
- (10) 使用 Fitter 对设计进行布局布线。
- (11) 使用 PowerPlay Power Analyzer 进行功耗估算和分析。
- (12) 使用 Timing Analyzer 对设计进行时序分析。
- (13) 使用 Simulator 对设计进行时序仿真。
- (14) (可选)使用物理综合、Timing Closure 平面布局图、LogicLock 功能、Settings 对话框和 Assignment Editor 改进时序,达到时序逼近。
- (15) 使用 Assembler 为设计建立编程文件。
- (16) 使用编程文件、Programmer 和 Intel 硬件对器件进行编程;或将编程文件转换为其他文件格式以供嵌入式处理器等其他系统使用。
- (17) (可选)使用 SignalTap II Logic Analyzer、SignalProbe 功能或 Chip Editor 对设计进行调试。
- (18) (可选)使用 Chip Editor、Resource Property Editor 和 Change Manager 管理工程更改。

下面根据以上步骤详细介绍利用 Block Editor 设计 4 位二进制计数器。

### 3.1.1 创建工作库

在完成一个设计(Project)之前,必须建立一个文件夹以便保存设计中的所有文件。需要注意的是文件夹名不能用中文,不能含有空格并且整个项目路径都不能有中文存在,建议以英文或英文结合数字方式命名。

建立好的文件夹称为工作库,一般情况下,不同的项目应放在不同的文件夹中,同一工程下的不同文件都应保存在同一文件夹中。

### 3.1.2 利用工程向导创建工程

打开 Quartus Prime 图形用户界面,如图 3.2 所示。



图 3.2 Quartus Prime 图形用户界面

选择 File→New Project Wizard 命令,利用向导来完成工程的建立,操作如图 3.3 所示,单击后出现如图 3.4 所示的对话框。对话框第一行为工作库路径,单击右侧 [...] 图标,选择为该项目新建的文件夹,此处选择了 E 盘名为 count16\_gdf 的文件夹作为项目工程的工作库。对话框第二行为新建工程命名,此处命名为 count16,当然工程名可以任取,只要符合命名规则即可。也可利用右侧按钮为当前项目添加已有工程。对话框第三行为当前工程顶层文件实体名,这里系统自动默认为当前的新建工程名 count16。

工作库和工程名称设置好之后,单击 Next 按钮进入下一设置环节,如图 3.5 所示。在该窗口可以选择创建空白工程或基于工程模板,本例选择基于空白工程,单击 Next 按钮。

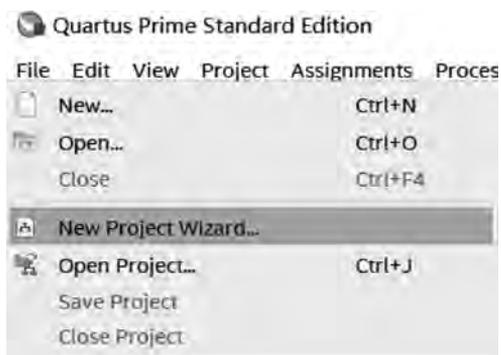


图 3.3 新工程建立向导

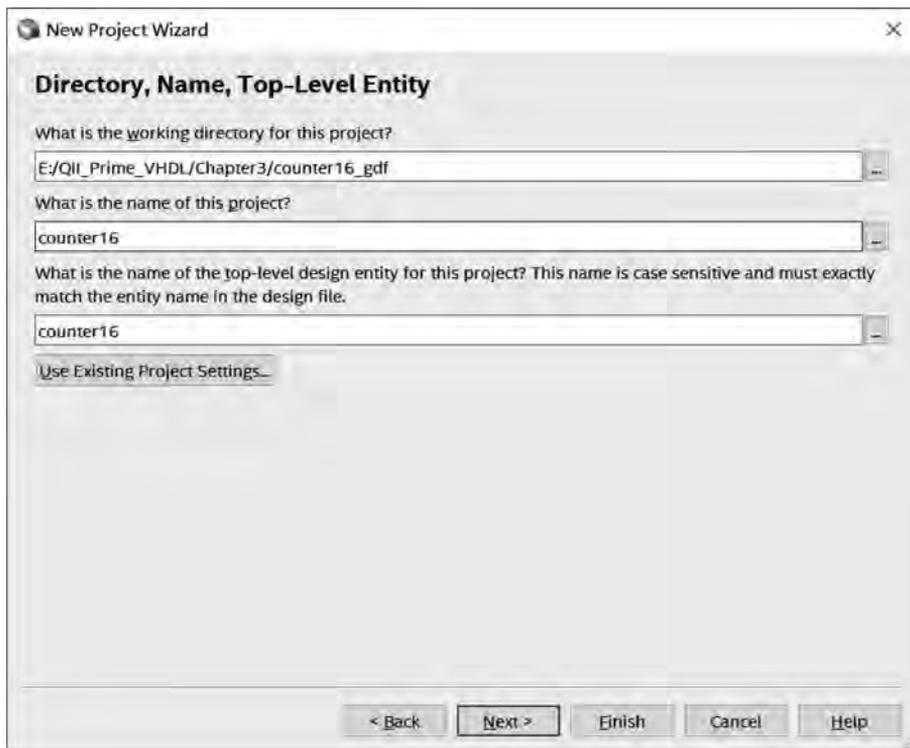


图 3.4 选择工作库和工程名称

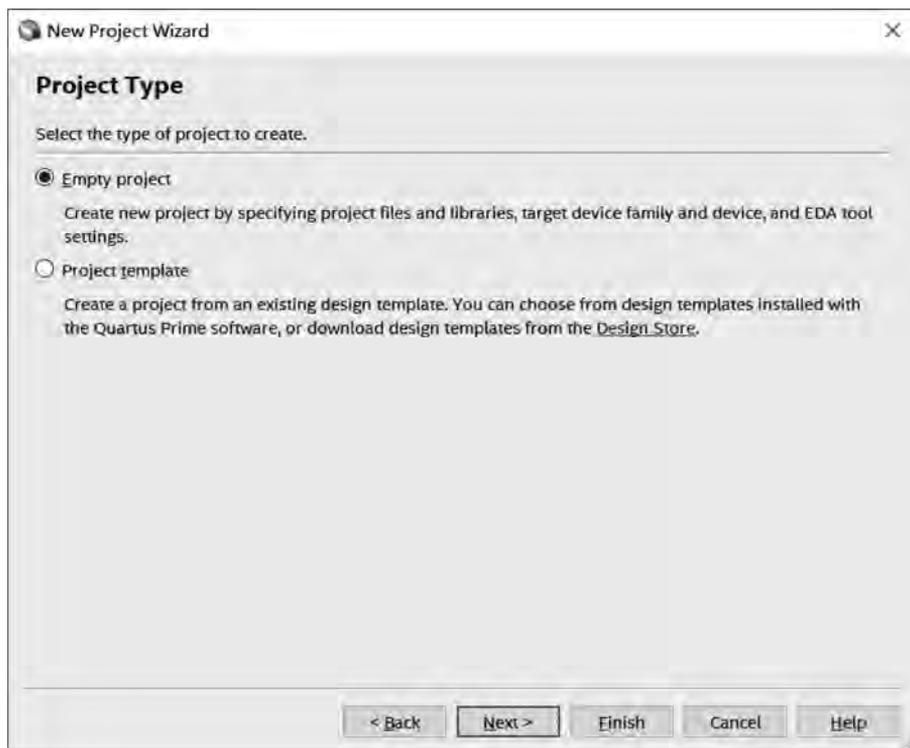


图 3.5 选择建立空白工程

在弹出的对话框中完成设计文件的添加。可以利用 File name 添加设计文件,也可以利用右侧 Add All 按钮添加文件夹中所有文件,如果没有已经设计好的文件,也可直接单击 Next 按钮略过此步,如图 3.6 所示。

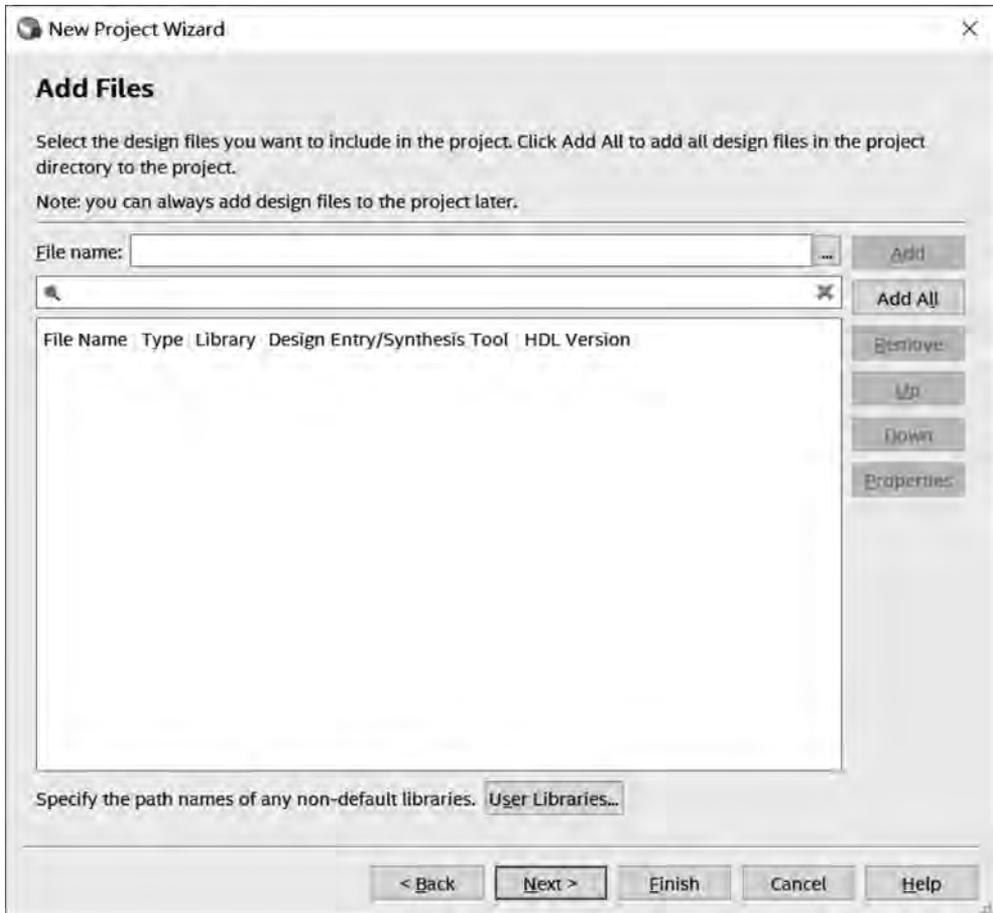


图 3.6 添加设计文件

进入第 4 页设置页面选择目标芯片,如图 3.7 所示。首先选择使用的目标芯片系列,本项目选择 Cyclone IV E 系列芯片,在下拉菜单中选择 EP4CE115F29C7 芯片,作为项目执行芯片,其中芯片的最后两个字符 C7 表示芯片的速度级别。当然也可以利用对话框右侧的过滤窗口选择目标芯片,Package 为芯片封装,Pin count 为芯片引脚数,Core speed grade 为速度级别。以上设置完成后单击 Next 按钮进入第 5 页设置。

第 5 页为 EDA 工具设置界面,如图 3.8 所示。该项设置主要用于输入设计、综合、仿真、校验等,如有第三方工具可以进行对应选择,此处主要选择工程所需要的综合与仿真工具,如果设计想使用第三方的工具,不使用系统集成的工具,需要在此处声明使用的是哪个工具,一般只能选择所支持的第三方工具,Synplify 就是一款性能优异的第三方综合工具,此处选择默认。仿真工具选用 ModelSim-Altera,语言选择 VHDL。EDA 工具设置完成后单击 Next 按钮结束设置,接下来弹出的窗口为项目总体概述,单击 Finish 按钮结束整个工程的建立。

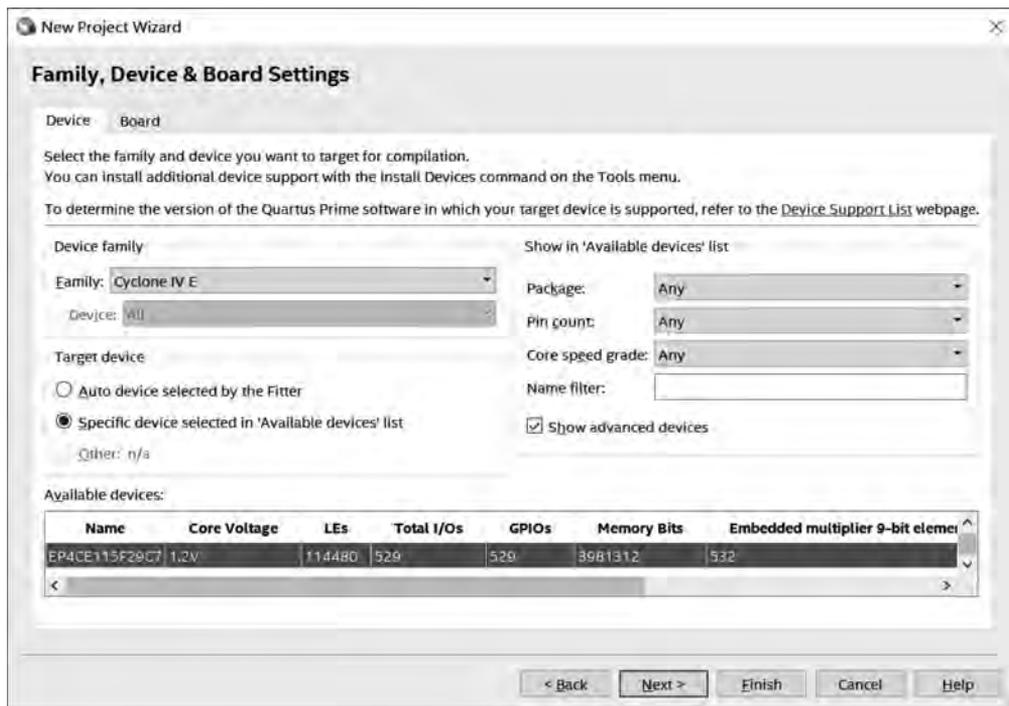


图 3.7 目标器件设置

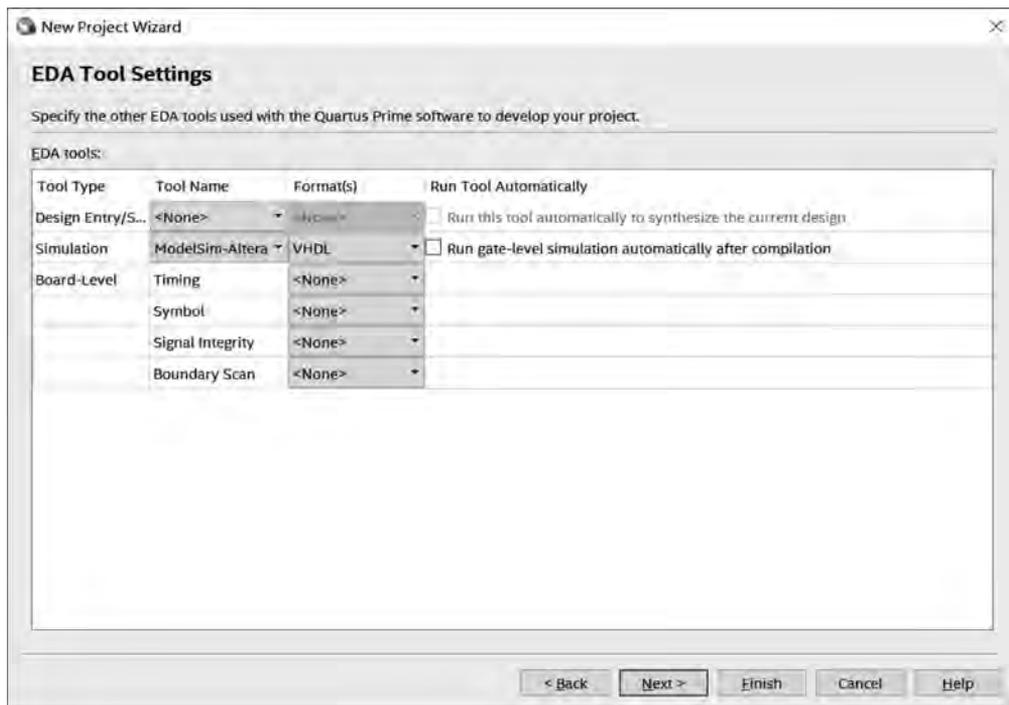


图 3.8 EDA 工具设置

### 3.1.3 图形设计输入

本节以 T 触发器设计 4 位二进制计数器为例讲述利用图形方式设计数字单元的方法。

(1) 建立工程后选择 File→New 命令为工程建立设计文件,弹出如图 3.9 所示的窗口。该窗口选项可为当前工程设计包含 4 类文件: Design Files(设计文件)、Memory Files(内存文件)、Verification/Debugging Files(校验/调试文件)、Other Files(其他文件)。本设计采用图形方式进行设计,选择 Design Files→Block Diagram/Schematic File 命令建立图形设计文件。

(2) 为图形设计文件添加所需元件。用鼠标左键双击图形工作区域的空白处或单击鼠标右键选择 Insert→Symble 命令,弹出 Symbol(元件符号)设置对话框,如图 3.10 所示,在此对话框下完成相应元件的添加。Quartus Prime 库中自带的元件均在其安装目录下的 libraries 文件夹中,在其文件夹下有 3 个子文件夹。

megafunctions 中为兆功能模块,主要包括参数可设置的 I/O、运算单元、逻辑门和存储单元;

others 中主要包含 Maxplus II 所自带的 74 系列标准单元;

primitives 中主要包含基本的 buffer(缓冲器)、logic(基本逻辑门)、other(包括逻辑高、低、常数等)、pin(输入、输出、双向端口)、storage(基本触发器等)。

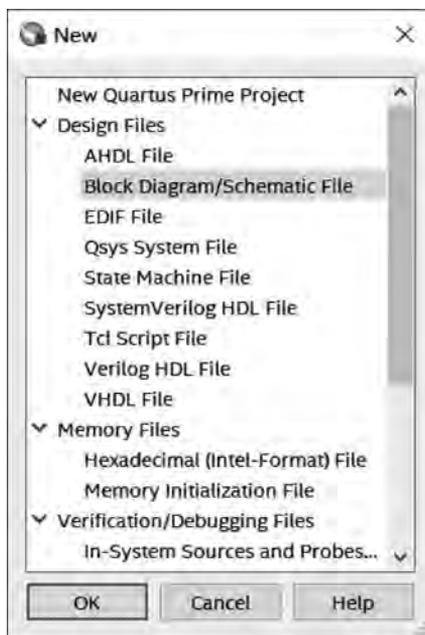


图 3.9 新建文件对话框

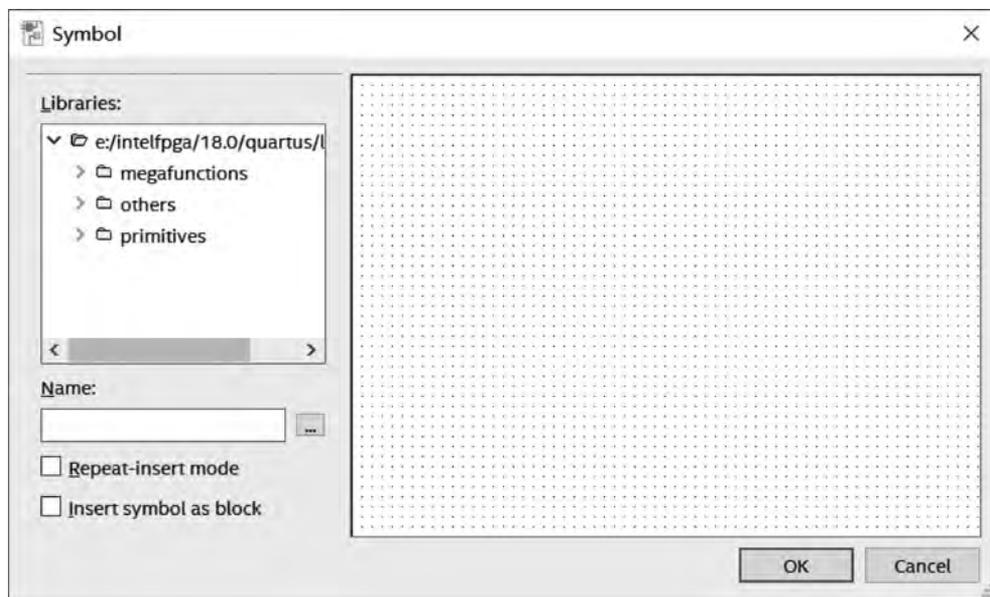


图 3.10 元件符号对话框

本设计中采用元件为 primitives 库中 logic 目录下的 and2, storage 目录下的 tff 触发器, other 目录下的 vcc, pin 目录下的 input、output。连接好后如图 3.11 所示。连接后在 input 和 output 的 pin\_name 处双击修改引脚名称, 修改名称原则符合 VHDL 命名规则, 不能重名。如有总线类型的引脚命名中总线宽度  $n$  用中括号表示  $[n-1, \dots, 0]$ 。

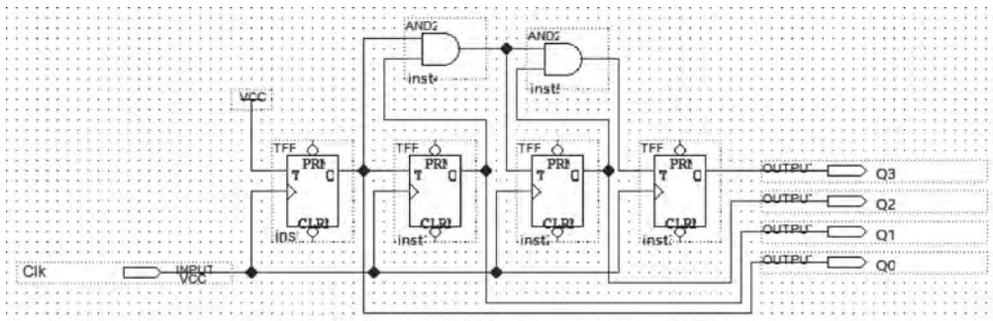


图 3.11 T 触发器设计的 4 位二进制计数器

(3) 存盘。选择 File→Save 命令进行初次保存, 命名为 counter16. bdf, 保存至项目路径所在文件夹中。如遇到修改设计名称可选择 File→Save as 命令为设计文件进行重新命名。

### 3.1.4 项目编译

项目设计文件完成之后, 要对设计文件进行检错、综合及时序分析等。同时产生多种用途的输出文件。编译之前要设置工程顶层实体。项目设计中如果存在多个设计文件需要编译, 则需要为编译器指定当前准备编译的设计, 选择 Project→Set as Top-level Entity 命令指定顶层实体。

Quartus Prime 编译器的主要任务是对设计项目进行检查并完成逻辑综合, 同时将项目最终设计结果生成器件的下载文件。Quartus Prime 软件中的编译类型有全编译和分步编译两种。选择 Quartus Prime 主窗口 Process→Start Compilation 命令, 或者在主窗口的工具栏上直接单击  图标可以进行全编译。全编译的过程包括分析与综合(Analysis & Synthesis)、适配(Fitter)、编程(Assembler)、时序分析(Classical Timing Analysis)这 4 个环节, 而这 4 个环节各自对应相应的菜单命令, 可以单独分步执行, 也就是分步编译。在设计调试和优化过程中, 可以使用 RTL 阅读器观察设计电路的综合结果。

分步编译就是使用对应命令分步执行对应的编译环节, 每完成一个编译环节, 生成一个对应的编译报告。分步编译与全编译一样分为 4 步:

(1) 分析与综合(Analysis & Synthesis): 设计文件进行分析和检查输入文件是否有错误, 对应的菜单命令是 Quartus Prime 主窗口 Processing 菜单下的 Start\Start Analysis & Synthesis, 对应的快捷图标是主窗口的左侧工具栏上的  。

(2) 适配(Fitter): 在适配过程中, 完成设计逻辑器件中的布局布线、选择适当的内部互连路径、引脚分配、逻辑元件分配等, 对应的菜单命令是 Quartus Prime 主窗口 Processing 菜单下的 Start\Start Fitter(注: 两种编译方式引脚分配有所区别); 对应的快捷图标是主窗口的左侧工具栏上的  。

(3) 编程(Assembler): 产生多种形式的器件编程映像文件,通过软件下载到目标器件中,对应的菜单命令是 Quartus Prime 主窗口 Processing 菜单下的 Start\Start Assembler; 对应的快捷图标是主窗口的左侧工具栏上的  Assembler (Generate programming files)。

(4) 时序分析(Classical Timing Analyzer): 计算给定设计与器件上的延时,完成设计分析的时序分析和所有逻辑的性能分析,菜单命令是 Quartus Prime 主窗口 Process 菜单下的 Start\Start Classical Timing Analyzer; 对应的快捷图标是主窗口的左侧工具栏上的  Timing Analysis。

编译完成以后,编译报告窗口 Compilation Report 会报告工程文件编译的相关信息,如编译的顶层文件名、目标芯片的信号、引脚的数目等,如图 3.12 所示。如果详细了解各部分编译报告可单击左侧 Task 栏中各个小三角符号,可展开了解详细编译报告。

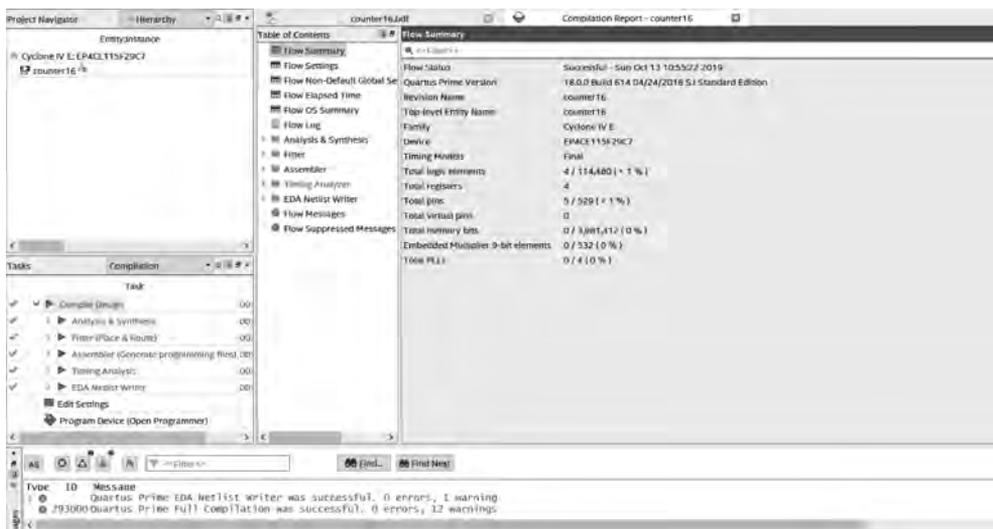


图 3.12 全编译运行结果信息

全编译操作简单,适合简单的设计。对于复杂的设计,选择分步编译可以及时发现问题,提高设计纠错的效率,从而提高设计效率。

在编译过程中,应随时注意界面下方 Message 信息栏中的 Processing 信息,此处显示当前编译器的处理进程,如遇到设计文件语句或原理图错误,将会暂停编译,用户可双击错误信息,软件会自动定位到出错文件的大概出错位置,根据错误提示信息及错误定位即可修改。注意,有时双击错误信息不会与原理图或设计文件对应,此时应考虑是否是软件设置所出现的问题,检查软件编译配置,重新编译。如果对某一文件进行编译时遇到大量错误信息,此时修改错误的原则应从第一条错误信息进行修改,修改完成之后可以保存进行再次编译,大多数情况下后面出现的错误信息往往是由前面的错误信息引起的连带错误报告。

(5) 项目编译其他设置。在对项目进行编译之前,可以根据项目需求做一些相关配置或目标器件、配置芯片的更改。

选择菜单 Assignments→Settings 命令会弹出项目各个设置对话框,如图 3.13 所示。也可选择工具栏中  图标打开设置对话框。设置对话框中左侧选项为设置分类(Category),右侧为相关分类的详细设置。

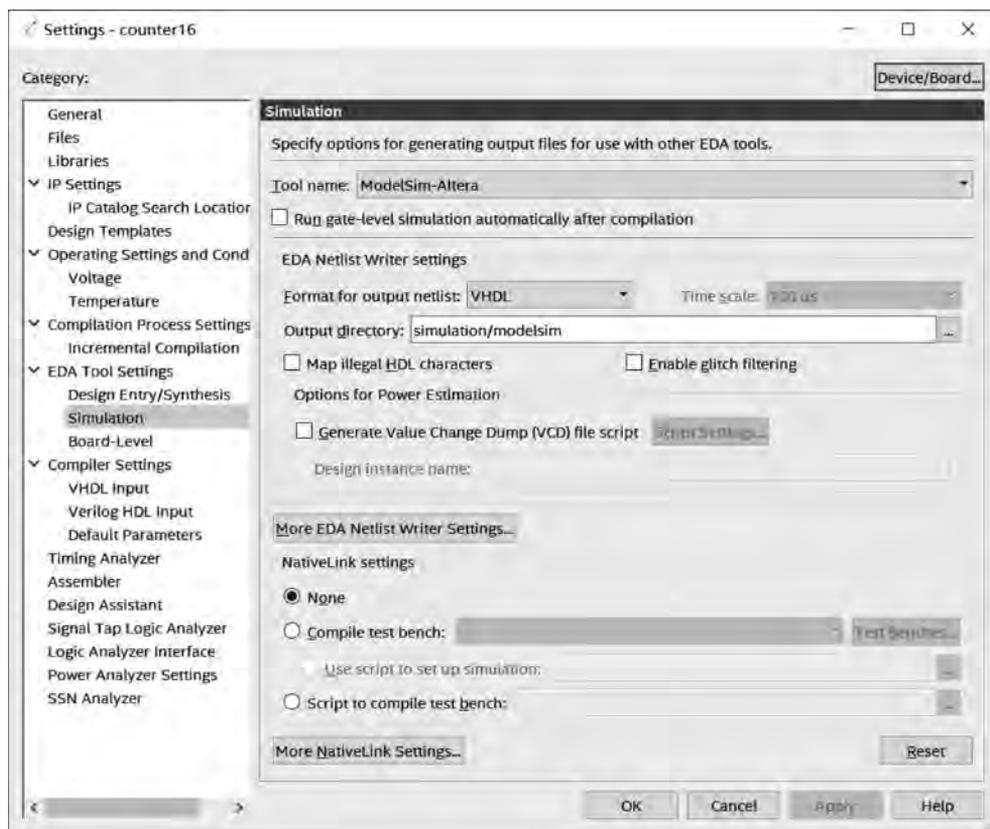


图 3.13 设置对话框

General: 项目顶层设计实体;

Files: 项目文件;

Libraries: 项目库;

Operating Settings and Conditions: 项目工作电压工作温度;

Compilation Process Settings: 编译处理器设置,可以为项目编译设置多核并行处理器、早期时间估计、增量编译、物理综合选项等;

EDA Tools Settings: 第三方 EDA 工具设置选项;

Assembler: 汇编设置。

设置对话框右上角有 Device 配置按钮,单击后弹出目标器件配置信息,器件配置对话框如图 3.14 所示,由此可修改目标芯片。

在对话框中单击 Device and Pin Options...按钮会弹出器件和引脚的详细配置,如图 3.15 所示。设置框左侧为设置分类,右侧为具体设置参数。在设置过程中应参看右下角 Description 描述说明来进行具体设置。系统默认的 General 选项为 Auto-restart configuration after error,意为配置过程出现错误后重新启动配置进程。JTAG 用户编码可由用户编码指令通过 JTAG 接口获得。

Configuration 配置选项如图 3.16 所示。Configuration scheme 选项为当前项目配置芯片选择配置方式,包括 Active Serial(主动串行配置)、Passive Serial(被动串行配置),以上

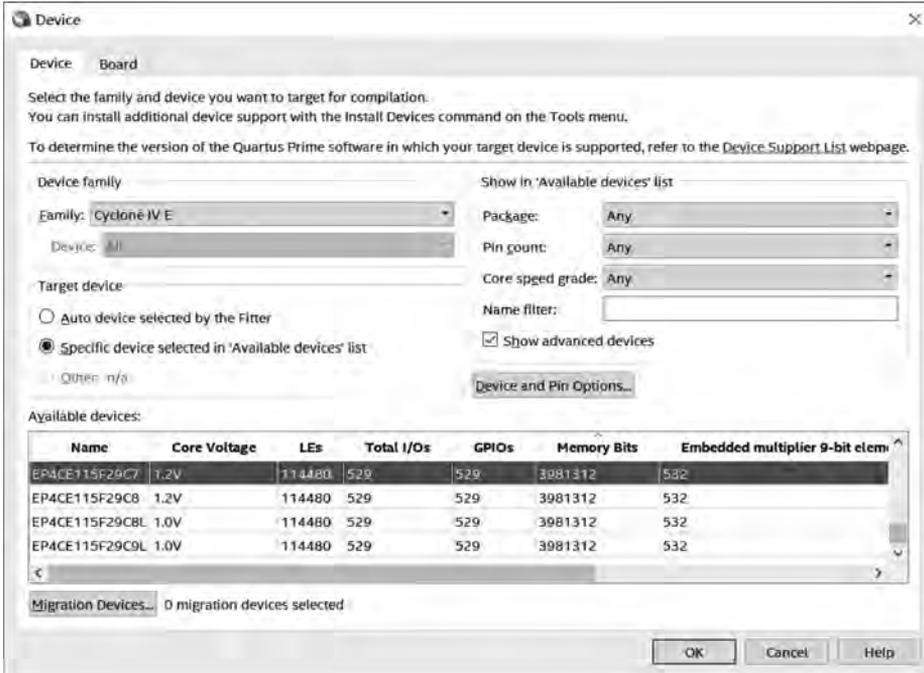


图 3.14 器件配置对话框

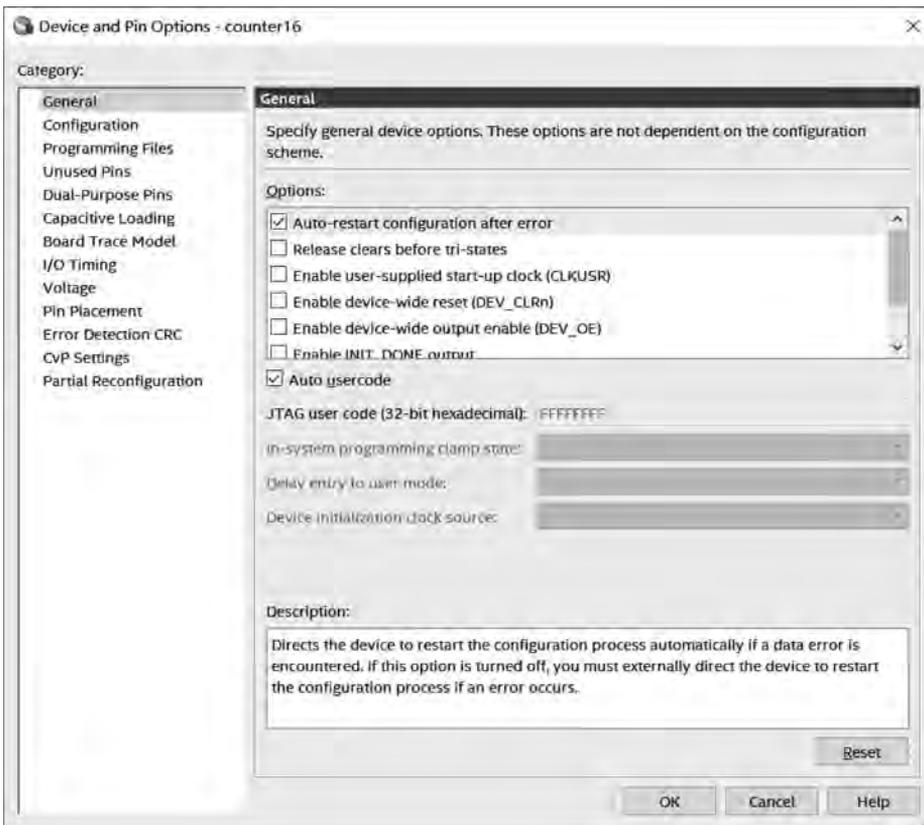


图 3.15 器件和引脚配置

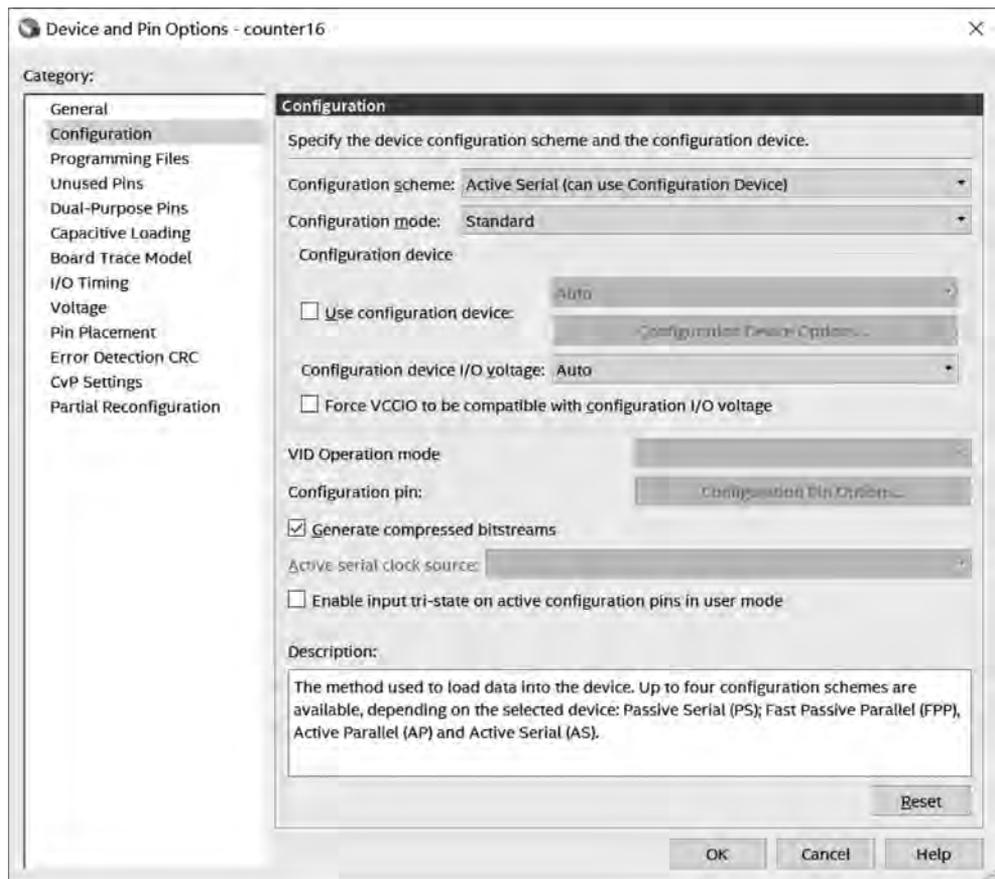


图 3.16 配置选项

两种方式都能用于芯片配置。如果使用配置芯片对目标芯片配置,选择 Use configuration device 中的具体型号,如 EPCS1、EPCS16 等,也可选择 Auto 进行自动选择,PS 方式仅用于 EPC1、EPC2 配置方式。Generate compressed bitstreams 选项可对配置文件进行压缩,目标芯片能够识别压缩后的配置文件并实时解压进行配置。

Programming Files 为输出文件配置,主要用于选择编程文件的产生格式。对于 PS 被动配置方式,Quartus Prime 软件会根据目标芯片型号指定不同的配置文件格式,如用于 SRAM 目标文件(.sof)、SRAM 文件(.psof)或编程目标文件(.pof)。也可产生二进制配置文件(.hexout),可设置起始地址和地址递增方式。此类文件可用于 EPROM 和单片机构成的 FPGA 配置电路。

Unused Pins 为未使用引脚设置,此处属于芯片全局未使用引脚配置。可对未使用引脚配置 5 种状态:As inputs tri-stated(输入三态)、As input tri-stated with bus-hold circuitry(具有总线保持电路的输入三态)、As input tri-stated with weak pull-up(具有弱上拉的输入三态)、As output driving ground(输出接地)、As output driving unspecified signal(输出不确定信号)。以上设置是针对全局未使用引脚进行配置,如需单一引脚配置,可使用 Assignment editor 进行配置。

Dual-Purpose Pins 为多功能引脚设置。在目标芯片配置完成后,某些配置引脚可进行

第二功能设定,如 nCEO、nCSO、ASDO。不同芯片多功能引脚不同,对于多功能引脚可以配置成普通 I/O 引脚或输入三态等。

### 3.1.5 时序仿真

对项目编译通过后,要对设计的功能和时序进行仿真测试,确保设计文件符合设计上的功能和时序上的要求。

时序仿真在不同 Quartus Prime 软件版本中有不同的操作,差别较大。在 Quartus Prime 9.1 之前(包括 MAXPLUS II)的软件自带仿真组件,而之后软件不再包含此组件。Quartus Prime 9.1 版本之前自带的仿真组件不支持 TestBench,只支持波形文件. vwf。vwf 文件全称是矢量波形文件(Vector Waveform File),是 Quartus Prime 中仿真输入、计算、输出数据的载体。一般设计者建立波形文件时,需要自行建立复位、时钟信号以及控制和输入数据、输出数据信号等。其中工作量最大的就是输入数据的波形录入。比如要仿真仅 1KB 的串行输入数据量,则手工输入信号的波形要画 8000 个周期,不仅费时费力而且容易出错。因此在 Quartus Prime 9.1 版本之后仿真必须安装 ModelSim-Altera 或第三方仿真软件 ModelSim PE/SE。在仿真之前应设计相应的测试文件(TestBench),通过调用 Quartus Prime 的 ModelSim-Altera 或第三方仿真软件 ModelSim 进行相关仿真。

ModelSim 仿真软件本身支持设计 VHDL 和 Verilog HDL,由于本例程采用 Quartus Prime 原理图设计方法,故无法直接调用本身的 ModelSim-Altera 或第三方仿真软件 ModelSim PE/SE 直接进行仿真,所以只能选取 Quartus Prime 编译后产生的文件添加至 ModelSim PE/SE 软件中进行仿真测试。

#### 1. 创建波形文件

依次选择 File→New→Verification/Debugging Files→University Program VWF 命令,如图 3.17 所示。

#### 2. 设置仿真时间

仿真时间长度根据被仿真工程的需要决定,依次选择 File→Edit→Set End Time 命令,本项目设置为 1.0 $\mu$ s,如图 3.18 所示。

#### 3. 添加仿真节点

在仿真之前需要在波形文件中添加需要的信号,选择 Edit→Insert→Insert Node or Bus 命令,弹出如图 3.19 所示的窗口。

单击窗口中 Node Finder 按钮,弹出如图 3.20 所示的过滤窗口。单击 List 按钮,在左侧会有相应的仿真信号列表,根据需要加以选择。再次单击 >> 按钮,将信号添加至右侧,单击 OK 按钮完成信号添加。添加后界面如图 3.21 所示。此时添加的输入信号默认都是 0 电平,输出信号呈网状,表示未知状态。



图 3.17 创建波形仿真文件

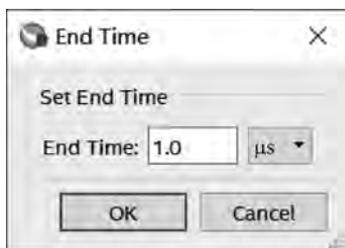


图 3.18 设置仿真结束时间

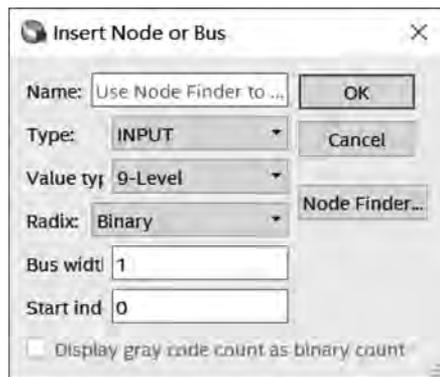


图 3.19 设置仿真时间

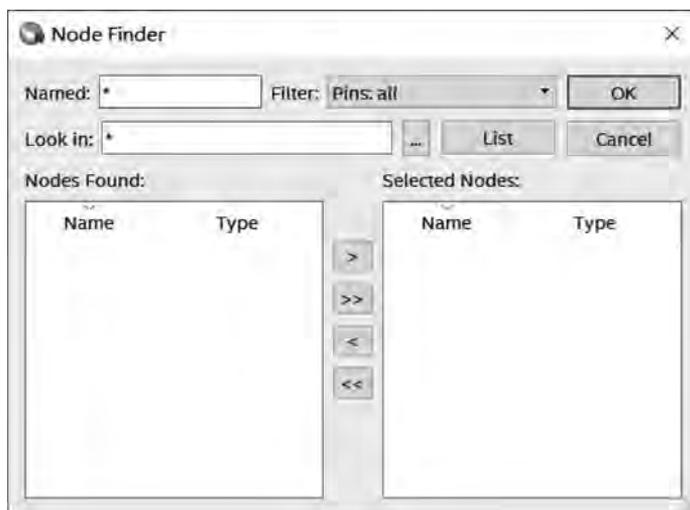


图 3.20 过滤窗口

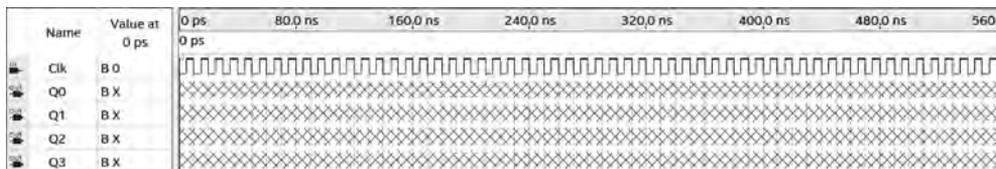


图 3.21 添加后仿真节点

#### 4. 添加激励信号

如果想要得到正确的仿真结果,需要根据设计工程的功能要求为输入信号按照一定逻辑添加激励信号。本工程输入信号只有 clock 故需要为该信号添加时钟信号。单击左键选中信号,选中信号后将会有工具栏被激活。其中,X 表示强未知,0 表示逻辑 0,1 表示逻辑 1,Z 表示高阻,L 表示弱 0,H 表示弱 1(表示电路中双向信号由电阻上拉或下拉),INV 表示取反,C 表示计数,秒表图标表示时钟,?表示设置信号表示方式,R 设置随机数,本工程选择时钟激励。添加后选择 File→Save 命令保存仿真文件。

## 5. 运行仿真

单击 Simulation→Option, 选择 Run Functional Simulation(功能仿真)或 Run Timing Simulator(时间仿真), 如图 3.22 所示。时间仿真可以显示出输出信号与输入信号的延时关系。功能仿真结果如图 3.23 所示。

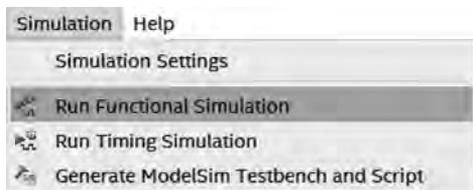


图 3.22 运行功能仿真

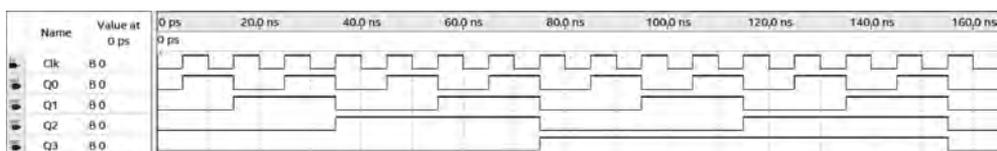


图 3.23 功能仿真结果

## 6. 信号分组

为了便于观察结果, 可以将关联的信号分组形成多值波形观察。本例中 Q0~Q3 可以分组用二进制或十进制或十六进制观察结果, 分组时注意信号方向, Q3 为高位, 本例中采用十六进制观察计数器结果, 结果显示为 0~F 循环变化。分组方法同时选中待分组信号, 右击 Grouping→Group, 弹出如图 3.24 所示的界面。Group name 自定义为 Q, Radix 选择为 Unsigned Decimal, 单击 OK 按钮, 仿真结果如图 3.25 所示。

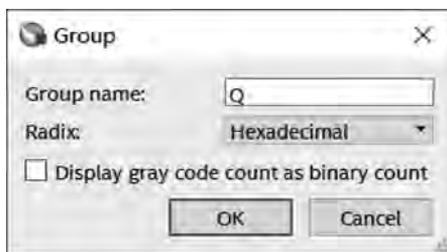


图 3.24 信号分组



图 3.25 分组后仿真结果

## 3.2 使用 Quartus Prime 进行 VHDL 设计

Quartus Prime 支持多种设计输入形式, 本书主要讲述利用 VHDL 进行数字系统设计开发的过程。



```

        counter <= temp;
    end process;
end art;

```

工程通过编译后,将要进行工程仿真,本节将介绍另外一种仿真方式,即 Quartus Prime 和 ModelSim-Altera 联合仿真,该方法也是业界目前较为普遍的仿真方式。

### 3.2.2 ModelSim-Altera 介绍

Mentor 公司的 ModelSim 是业界最优秀的 HDL 语言仿真软件,它能提供友好的仿真环境,是业界唯一的单内核支持 VHDL 和 Verilog 混合仿真的仿真器。它采用直接优化的编译技术、Tcl/Tk 技术和单一内核仿真技术,编译仿真速度快,编译的代码与平台无关,便于保护 IP 核,个性化的图形界面和用户接口为用户加快调错提供强有力的手段,是 FPGA/ASIC 设计的首选仿真软件。

ModelSim 分几种不同的版本: SE、PE、LE 和 OEM,其中 SE 是最高级的版本。而集成在 Actel、Atmel、Intel、Xilinx 以及 Lattice 等 FPGA 厂商设计工具中的均是其 OEM 版本。ModelSim SE 是主要版本号,也是功能最强大的版本,支持对 Verilog 和 VHDL 语言的混合仿真。除了主要版本外,Mentor 公司还为各大 FPGA 厂商提供 OEM 版本: XE 是为 Xilinx 公司提供的 OEM 版,包括 Xilinx 公司的库文件; AE 是为 Intel 公司提供的 OEM 版,包含 Altera 公司的库文件;在用特定公司的 OEM 版进行仿真时不需要编译该公司的库文件,但是仿真速度等性能指标都要落后于 SE 版本。

### 3.2.3 TestBench 编写

如果采用 ModelSim-Altera 进行仿真,需要为被仿真文件编写 TestBench,本节只给出利用 Quartus Prime 模板编写 TestBench 文件并完成仿真过程及结果。详细使用和 TestBench 写法将在后续章节中详细介绍。

Quartus Prime 软件根据设计文件可产生一个 TestBench 设计模板,该模板已经给出被测文件的元件声明与例化,并完成了相关信号的定义,用户可在现有模板的基础上完成相关激励信号的产生。

选择 Quartus Prime 菜单中 Processing→Start→Start Test Bench Template Writer 命令,启动 TestBench 模板,如图 3.27 所示。Quartus Prime 利用网表编写器完成 TestBench 的编写操作,并将文件自动保存至文件夹中,根据软件下方的信息提示可见测试文件被保存在 Info(201002): Generated VHDL Test Bench File E:/QII\_Prime\_VHDL/Chapter3/counter16\_gdf/simulation/mod-elsim/cnt10.vht for simulation 中,测试文件后缀为 \*vht,具体信息如图 3.28 所示。

单击工具栏中的打开  按钮,根据 TestBench 模板创建信息提示,打开软件创建的 TestBench 模板文件,该文件保存在工程所在文件夹下: simulation/modelsim/cnt10\_vhd\_tst.vht。软件自动创建的 TestBench 模板文件如例 3.2 所示。

**【例 3.2】** TestBench 模板文件。

```

LIBRARY ieee;
USE ieee.std_logic_1164.all;

```

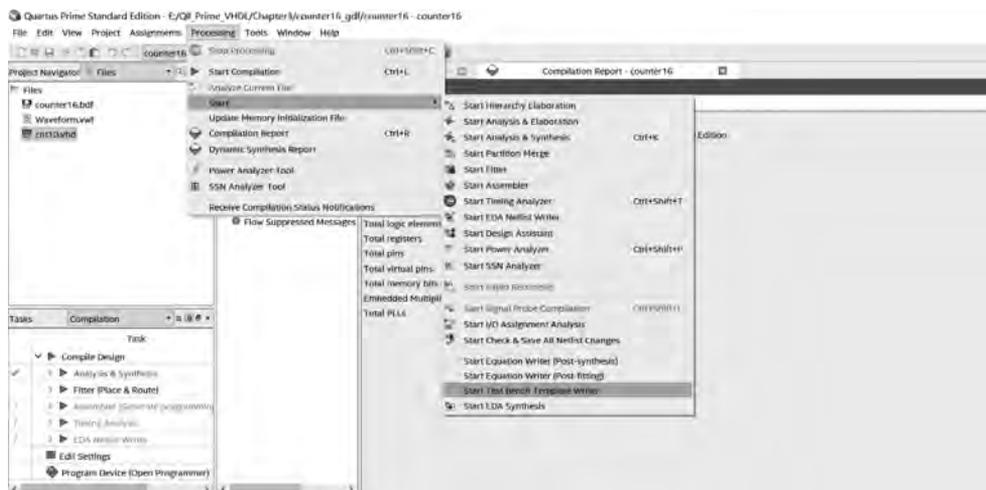


图 3.27 启动 TestBench 模板

```
Type ID Message
201002 Generated VHDL Test Bench File E:\QIT_Prime_VHDL\Chapter3\counter16_gdf\simulation\modelsim\cnt10.vht for simulation
```

图 3.28 模板自动保存位置信息

```
ENTITY cnt10_vhd_tst IS
END cnt10_vhd_tst;
ARCHITECTURE cnt10_arch OF cnt10_vhd_tst IS
-- constants
-- signals
SIGNAL clk : STD_LOGIC;
SIGNAL counter : STD_LOGIC_VECTOR(3 DOWNTO 0);
SIGNAL rst_n : STD_LOGIC;
COMPONENT cnt10
PORT (
clk : IN STD_LOGIC;
counter : OUT STD_LOGIC_VECTOR(3 DOWNTO 0);
rst_n : IN STD_LOGIC
);
END COMPONENT;
BEGIN
i1 : cnt10
PORT MAP (
-- list connections between master ports and signals
clk => clk,
counter => counter,
rst_n => rst_n
);
init : PROCESS
-- variable declarations
BEGIN
-- code that executes only once
WAIT;
END PROCESS init;
```

```

always : PROCESS
-- optional sensitivity list
-- (      )
-- variable declarations
BEGIN
    -- code executes for every event on sensitivity list
WAIT;
END PROCESS always;
END cnt10_arch;

```

从 TestBench 模板来看,测试文件中实体为空,实际上测试文件主要是产生内部的激励源,通过内部信号与被测文件实体端口进行连接进行测试。测试文件的结构体说明语句中包含两部分:一部分为与被测实体端口连接的内部信号定义;另一部分为被测文件的元件声明。被测端口为定义信号,与被测实体端口一致,为 clk、counter、rst\_n。结构体描述语句中给出两类进程的模板:初始化进程,该进程内部信号仅执行一次就进入无限等待;循环执行进程,多次执行用于时钟类信号的产生。本例程为具有异步复位功能的十进制计数器,输入激励信号为时钟信号和复位信号,因此在一次执行信号中加入复位信号,复位时间为 100ns,100ns 后复位信号拉高;在循环进程中添加时钟产生语句,添加前在结构体内部定义了时钟周期为 20ns 的常数 clock\_period,及仿真 50MHz 时钟信号。在循环进程中利用 WAIT 语句生成时钟信号。修改后的测试文件如例 3.3 所示。

**【例 3.3】** 修改后的测试文件。

```

LIBRARY ieee;
USE ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
use ieee.std_logic_arith.all;
ENTITY cnt10_vhd_tst IS
END cnt10_vhd_tst;
ARCHITECTURE cnt10_arch OF cnt10_vhd_tst IS
constant clock_period:TIME:= 20ns;
SIGNAL clk : STD_LOGIC;
SIGNAL counter : STD_LOGIC_VECTOR(3 DOWNT0 0);
SIGNAL rst_n : STD_LOGIC;
COMPONENT cnt10
    PORT (
        clk : IN STD_LOGIC;
        counter : OUT STD_LOGIC_VECTOR(3 DOWNT0 0);
        rst_n : IN STD_LOGIC
    );
END COMPONENT;
BEGIN
    il : cnt10
    PORT MAP (
        clk => clk,
        counter => counter,
        rst_n => rst_n
    );
    init : PROCESS
    BEGIN
        rst_n <= '0';

```

```

        wait for 100ns;
        rst_n<= '1'; -- code that executes only once
WAIT;
END PROCESS init;
always : PROCESS
BEGIN
    clk<= '1';
    wait for clock_period/2;
    clk<= '0';
    wait for clock_period/2;
END PROCESS always;
END cnt10_arch;

```

### 3.2.4 调用 ModelSim-Altera RTL 仿真

RTL 行为级仿真也叫功能仿真,这个阶段的仿真可以用来检查代码中的语法错误及代码行为的正确性,其中不包括延时信息。如果没有实例化一些与器件相关的特殊底层元件,这个阶段的仿真也可以做到与器件无关。需要的文件为编写的 VHDL 源文件以及 TB 文件。如果用到 PLL 等 IP 核,需要挂载器件库文件。

(1) 设置测试文件。单击 Quartus Prime 主窗体中 Assignments→Settings 下的 Simulation,弹出如图 3.29 所示选项卡。在 NativeLink settings 下单击 **Test Benches** 按钮,弹出如图 3.30 所示窗体,单击 **New** 按钮,弹出如图 3.31 所示的窗体,单击窗体 **...** 按钮,选择之前编辑好的测试文件路径,一般在...simulation/modelsim/cnt10.vht,单击 Add 按钮完成测试文件的添

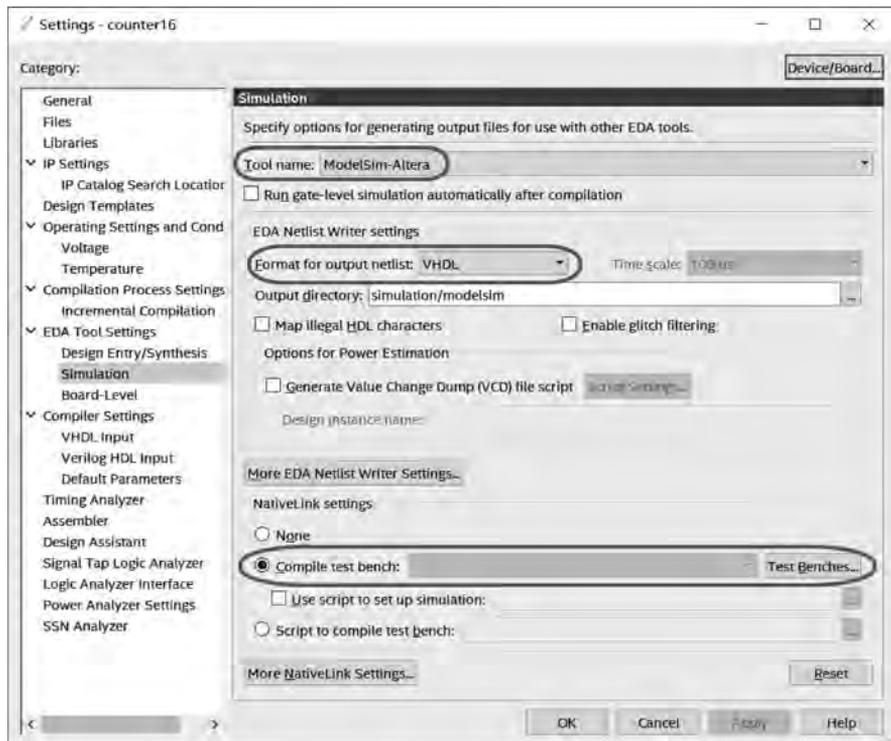


图 3.29 设置测试文件

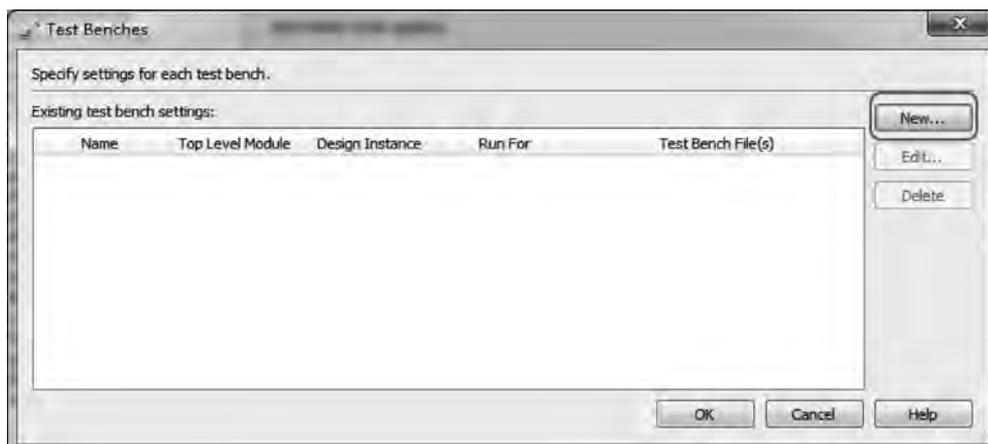


图 3.30 新建测试文件

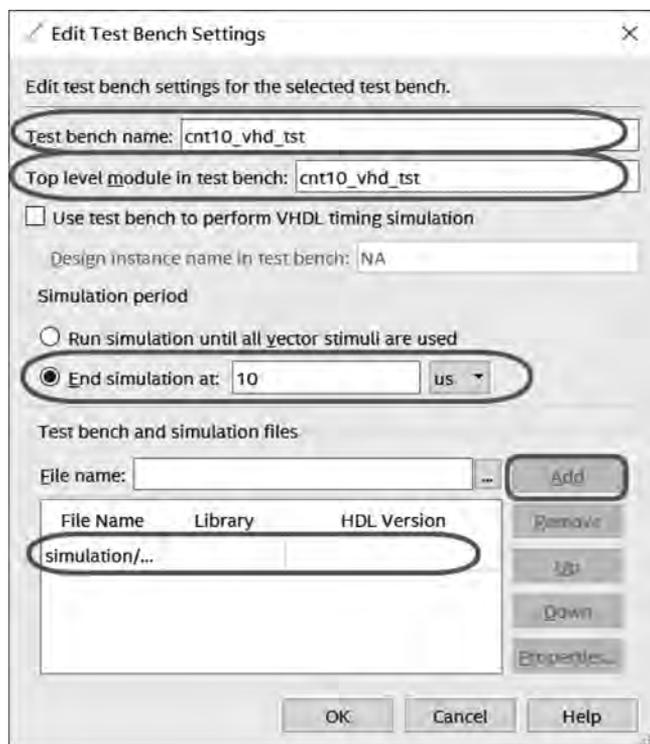


图 3.31 测试文件设置

加。在 Test bench name 和 Top level module in test bench 中填入测试文件名称,本例中名称为 cnt10\_tst。

(2) RTL 仿真。单击 Tools → RTL simulation, Quartus Prime 会自动调用 ModelSim-Altera 仿真器弹出如图 3.32 所示的仿真结果。

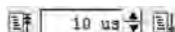


图 3.32 RTL 仿真结果

(3) ModelSim-Altera 快捷按钮说明:

: 停止仿真;

: 第一个图标表示窗口放大; 第二个图标表示以游标为中心缩小; 第三个图标表示窗口适配, 将当前全部仿真结果在一个界面中显示; 第四个图标表示以游标为中心放大;

: 第一个图标为复位, 所有仿真信号复位; 第二个窗体为仿真时间; 第三个图标为运行一次仿真。

为了便于观察仿真结果, 可以选择图 3.32 中左侧的信号名称, 单击鼠标右键, 选择 Radix 显示结果, 本例中可以选择 unsigned 作为结果显示。

### 3.2.5 调用 ModelSim-Altera 门级仿真

门级仿真也叫综合后仿真。绝大多数的综合工具除了可以输出一个标准网表文件以外, 还可以输出 Verilog 或者 VHDL 网表, 其中标准网表文件是用来在各个工具之间传递设计数据的, 并不能用来仿真, 而输出的 Verilog 或者 VHDL 网表可以用来仿真。之所以叫门级仿真, 是因为综合工具给出的仿真网表已经与生产厂家器件的底层元件模型对应起来了。所以为了进行综合后仿真, 必须在仿真过程中加入厂家的器件库, 对仿真器进行一些必要的配置, 否则仿真器并不认识其中的底层元件, 无法进行仿真。综合后生成的网表文件 (.vo) 加 TB 仿真; 网表是与器件有关的, 所以要挂载好相关器件库文件。可以看到, 门级仿真引入了中间态。对于 Quartus Prime 生成的 vo 文件, 首先要注释掉其中的挂载 sdo 文件语句, 否则仿真是时序仿真, 因为添加了 sdo 延时文件。需要的文件为 vo 网表文件以及 TB 文件。需要挂载器件库文件。

时序仿真也叫后仿真。在设计布局布线完成以后, 可以提供一个时序仿真模型, 这种模型中也包括了器件的一些信息, 同时还会提供一个 SDF 时序标注文件 (Standard Delay Format Timing Anotation)。SDF 时序标注最初使用在 Verilog 语言的设计中, 现在 VHDL 语言的设计中也引用了这个概念。对于一般的设计者来说, 并不需知道 SDF 文件的详细细节, 因为这个文件一般由器件厂家提供给设计者, Xilinx 公司使用 SDF 作为时序标注文件扩展名, Intel 公司使用 SDO 作为时序标注文件的扩展名。在 SDF 时序标注文件中对每一个底层逻辑门提供了 3 种不同的延时值, 分别是典型延时值、最小延时值和最大延时值, 在对 SDF 标注文件进行实例化说明时, 必须指定使用了哪种延时。虽然在设计的最初阶段就已经定义了设计的功能, 但是只有当设计布局布线到一个器件中之后, 才会得到精确的延时信息, 在这个阶段才可以模拟到比较接近实际电路的行为。网表文件加延时, 仿真中会包含延时信息。需要的文件为 vo 网表文件以及 TB 文件以及延时文件 sdo (采用脚本挂载)。需要挂载器件库文件。

具体步骤参考 3.2.4 节。在弹出如图 3.31 所示界面时需要进行修改, 修改结果如图 3.33 所示。

在进行门级仿真之前, VHDL 设计文件需进行 EDA Netlist Writer 操作, 获得底层网表文件。设置完成后单击 Tools → GATE level Simulation, Quartus Prime 会自动调用 ModelSim-Altera 仿真器, 弹出如图 3.34 所示的仿真结果。从图中可以看出, 当时钟上升沿到来后, 输出需要经过一段延时才发生变化。

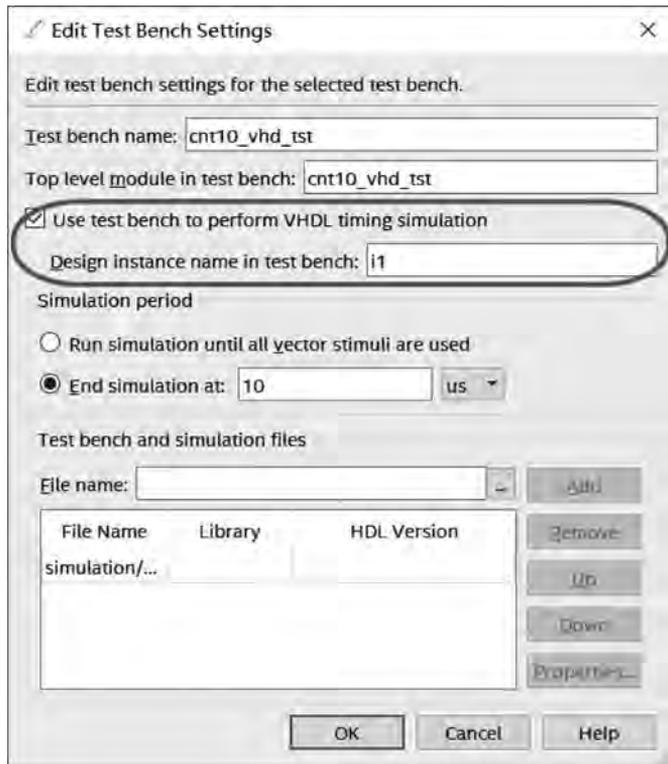


图 3.33 门级仿真设置



图 3.34 门级仿真结果

### 3.2.6 引脚分配

仿真完成后,需要给设计分配引脚。分配引脚的目的是将设计中的输入输出信号指定到器件中的某个引脚,并设置此引脚的电平标准和电流强度等。分配引脚的原则是根据器件的外围电路决定的。在本例中输入信号 1 个为时钟 1 个复位,输出信号 4 个,根据外设的实际情况,将输入引脚分配至按钮开关实现时钟输入和复位输入,输出引脚分配至 4 个发光二极管,通过发光二极管的状态判断当前的设计是否实现预定功能。

单击 Assignment→Pin Planner 弹出如图 3.35 所示的窗口。在 Location 列根据信号名称双击该区域分配引脚。按照图 3.35 按顺序给工程中输入输出引脚进行分配。完成引脚分配后直接关闭该窗体。

Node Name	Direction	Location	I/O Bank	VREF Group	Iter Location	I/O Standard	Reserved	Current Strength	Slew Rate	Differential Pair	IOB Preserved
clk	Input	PIN_M23	6	B6_N2	PIN_J1	2.5 V _fault		8mA (default)			
counter[3]	Output	PIN_F21	7	B7_N0	PIN_R6	2.5 V _fault		8mA (default)	2 (default)		
counter[2]	Output	PIN_F19	7	B7_N0	PIN_R3	2.5 V _fault		8mA (default)	2 (default)		
counter[1]	Output	PIN_F19	7	B7_N0	PIN_U4	2.5 V _fault		8mA (default)	2 (default)		
counter[0]	Output	PIN_G19	7	B7_N2	PIN_U3	2.5 V _fault		8mA (default)	2 (default)		
rst_n	Input	PIN_M21	6	B6_N1	PIN_Y2	2.5 V _fault		8mA (default)			
<<new node>>											

图 3.35 Pin Planner 分配窗口

### 3.2.7 分析与综合

综合就是把 HDL 语言/原理图转换为最基本的与、或、非门及 RAM、触发器等基本逻辑单元的链接关系,并根据要求优化形成的门级逻辑连接最终生成综合网表的过程。综合网表的业界标准是 EDIF 格式。文件后缀通常为 .edn、.edf、.edif。综合网表中,除了包含 HDL 语言中与门、非门等组合逻辑和寄存器等时序逻辑外,还包含 FPGA 特有的各种原语(Primitive),比如 LUT、BRAM、DSP48,甚至 PowerPC、PCIe 等硬核模块,以及这些模块的属性和约束信息。

Quartus Prime 的 Analysis & Synthesis 支持 VHDL 1987、VHDL1993 和 VHDL2008 标准。这些标准均可在 Assignment → Setting → Analysis & Synthesis settings → VHDL Input 中设置。除了 QUARTUS PRIME 自带的综合工具外还有很多第三方的综合工具,如 Synopsys(收购了 Synplicity)的 Synplify, Mentor Graphic 的 Precision。

进行分析与综合可以单击软件快捷工具中  或左侧导航中  进行。

### 3.2.8 布局与布线

所谓布局是指将逻辑网表中的硬件原语或者低层单元合理地适配到 FPGA 内部的固有硬件结构上,布局的优劣对设计的最终结果(在速度和面积两个方面)影响很大。所谓布线是指根据布局的拓扑结构,利用 FPCA 内部的各种连线资源,合理正确连接各个元件的过程。Quartus Prime 软件的 Fitter(Place & Route),指的就是布局布线,它将每一个逻辑函数放在最好的逻辑单元位置以满足布线和时序,它也自动分配恰当互连路径和引脚。如果用户在设计中有设计约束,布局布线会在目标器件上尽力满足设置的约束,并优化整体设计。如没有设置的约束,布局布线也会自动进行优化。详细设置可在 Assignment → Setting → Fitter Settings 选项中进行设置。

### 3.2.9 器件编程

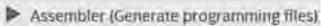
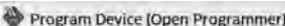
当工程编译通过后,接下来要进行器件编程或配置用于板级验证。单击快捷工具中  或左侧导航中  生成编程文件。Quartus Prime 编程器允许编程和配置 Intel 公司的 CPLD、FPGA 和配置器件。编程或配置文件格式如表 3.2 所示。

表 3.2 编程文件格式

文件格式	FPGA	CPLD	配置器件	串行配置器件
SRAM Object File(.sof)	YES	—	—	—
Program Object File(.pof)	—	YES	YES	YES
JEDEC JESD71 STAPL Format File(.jam)	YES	YES	YES	—
JAM Byte Code File(.jbc)	YES	YES	YES	—

单击快捷按钮  或双击左侧任务栏  打开 Quartus Prime Programmer 界面,如图 3.36 所示。

单击界面左上角 Hardware Setup...进行编程器设置,如果编程器正常,则在弹出界面中选择 USB-Blaster[USB\_D]选项,如图 3.37 所示。



图 3.36 Quartus Prime Programmer 界面

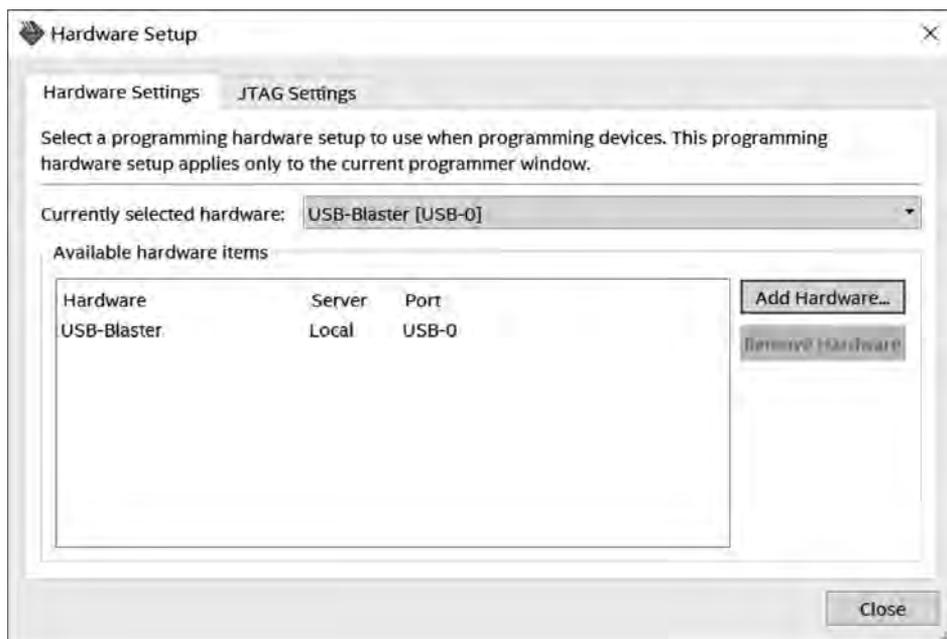


图 3.37 设置硬件

硬件设置完成后需要设置编程模式，编程模式设置在图 3.36 右上方下拉列表中选择，本例中选择 JTAG 模式。Intel 可编程逻辑器件中支持 4 种编程模式，如表 3.3 所示。

表 3.3 编程模式

Quartus Prime Programmer 支持的编程模式	FPGA	CPLD	配置器件	串行配置器件
JTAG	YES	YES	YES	—
Passive Serial(PS)	YES	—	—	—
Active Serial(AS)	—	—	—	YES
Configure via Protocol(CVP)	YES	—	—	—
In-socket Modes(ISM)	—	MAX II 器件除外	YES	YES

编程模式选择后需要制定编程文件，单击图 3.36 中左侧 Add File 按钮，在弹出窗口中制定工程中的编程文件——即在工程所在文件夹 output\_files 中以 sof 为后缀的配置文件。本例中选择 counter\_sof 文件，同时勾选与文件对应的 Program/Configure。最后单击图 3.36

中 Start 按钮进行器件配置,当左上角进度条 100%时表示已完成器件配置,用户可以进行板级硬件验证。

### 3.3 Quartus Prime 的 IP 使用

Intel 公司的 Quartus Prime 软件提供两类功能模块:免费的 LPM 宏功能模块 (Megafunction/LPM)如表 3.4 所示,和需要授权使用的 IP 和 AMPP IP 核 (MegaCore),如表 3.5 所示,两者在实现功能上有区别,使用方法相同。

表 3.4 Intel Quartus Prime 提供的基本宏功能

类 型	描 述
算术组件	累加器、加法器、乘法器和 LPM 算术函数
门	包括多路复用器和 LPM 门函数
I/O 组件	包括时钟数据恢复(CDR)、锁相环(PLL)、双数据速率(DDR)、千兆位收发器块(GXB)、LVDS 接收器和发送器、PLL 重新配置和远程更新宏功能模块
存储器编译器	FIFO partitioner、RAM 和 ROM 宏功能模块
存储组件	存储器、移位寄存器和 LPM 存储器函数

表 3.5 Intel Quartus Prime 提供的 MegaCore

数字信号处理类	通 信 类	接口和外设类	微处理器类
FIR	UTOPIA2	PCI MT32	Nios&Nios II
FFT	POS-PHY2	PCI T32	SRAM Interface
Reed Solomon	POS-PHY3	PCI MT64	SDR DRAM Interface
Virterbi	SIP4. 2	DisplayPort	FLASH Interface
Turbo Encode/Decoder	SONET Framer	DDR1/2/3	UART
NCO	Rapid I/O	Memory I/F	SPI
Color Space Converter	8B10B	HyperTransport	Programmable I/O
FIR II		PCIE1/4/8	SMSC MAC/PHY I/F
		QDRII+MEM IF	HPS
		RLDRAM MEM IF	

本节在 3.1.3 节的 4 位二进制计数器基础上以为 clk 引脚添加宏功能模块 PLL 为例,对 Quartus Prime 中使用宏功能的方法加以说明。用户可以利用 Quartus Prime 软件中的 IP Catalog 管理器来建立或修改宏功能模块。要运行 IP Catalog 管理器需要在 Quartus Prime 界面下单击 Tools→IP Catalog 选项或单击快捷工具按钮  在软件界面弹出如图 3.38 所示的窗体。从库中看出有几类 IP:基本功能、DSP、接口协议、内存接口和控制器、处理器和外围设备、大学计划 IP。由于是第一次添加选择第一个选项,单击 Next 按钮,弹出如图 3.39 所示的窗口。

在图 3.38 所示界面的左侧选择需要添加的宏功能模块。配置 PLL 时,在 Basic Functions→Clocks,PLL,Resets→PLL→ALTPLL,语言选择 VHDL,输出路径选择用户工程所在文件夹模块名称命名为 my\_pll,如图 3.39 所示,单击 OK 按钮,弹出如图 3.40 所示设置窗体。



图 3.38 IP Catalog 添加界面

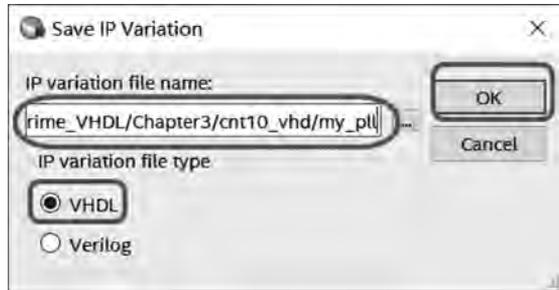


图 3.39 添加 PLL

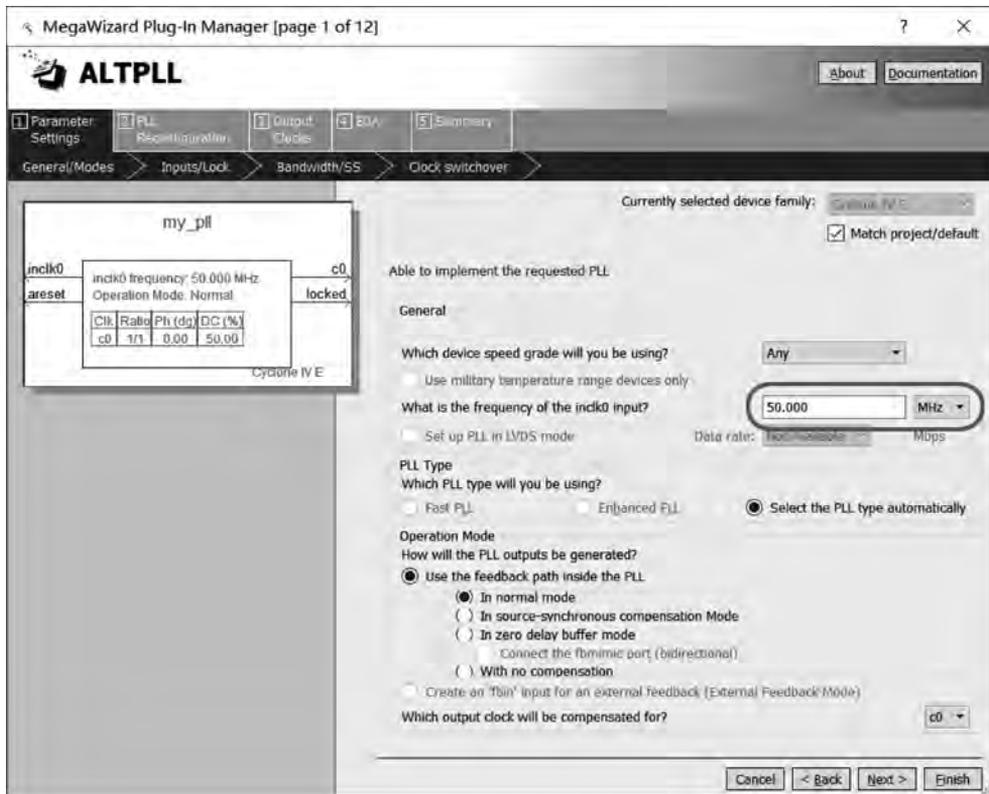


图 3.40 用 MegaWizard 设置 page1

配置 PLL 时,在图 3.40 窗口中 inclk0 频率根据硬件实际情况设置,本例设置为 50MHz,从图 3.40 可以看出,设置 PLL 共需要 12 步,以下配置将根据设计需求进行选择性说明。单击 Next 按钮,弹出如图 3.41 所示的窗口。

取消勾选图 3.41 所示窗口中的选项,此处需根据实际情况设置。单击 Next 按钮,直到设置 page6,如图 3.42 所示。

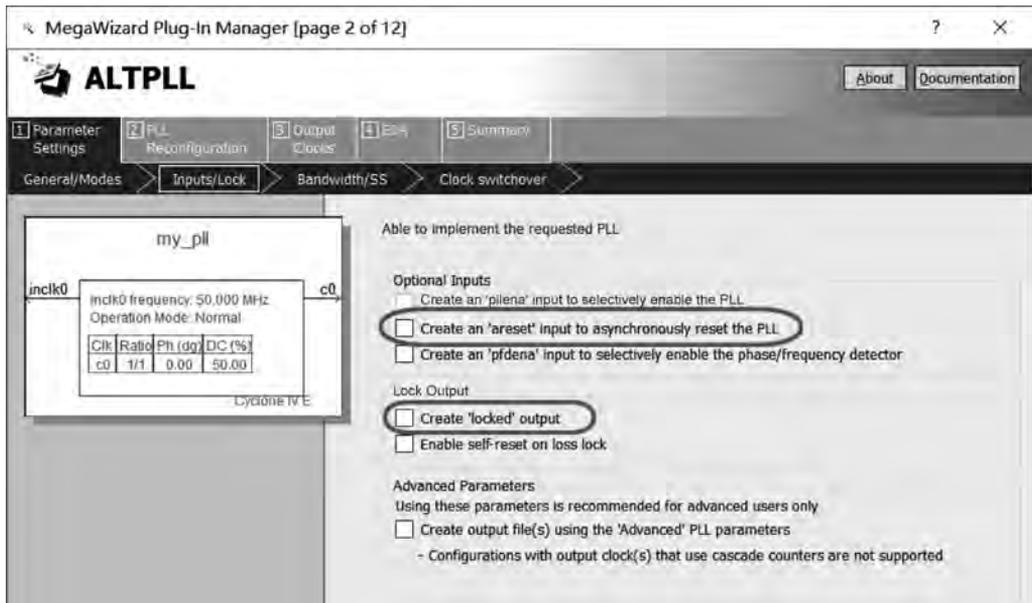


图 3.41 MegaWizard 设置 page2

在图 3.42 窗口中可以设置想要输出的时钟参数：相移、占空比、频率。设置输出频率有两种方式：第一种直接选中 Enter output clock frequency, 直接设置想要的输出频率值, 单位为 MHz, 第二种可以是针对输入时钟进行倍频、分频系数设置等到输出时钟频率。本例中选择第一种直接设置输出时钟频率 0.002MHz, 如图 3.42 所示。

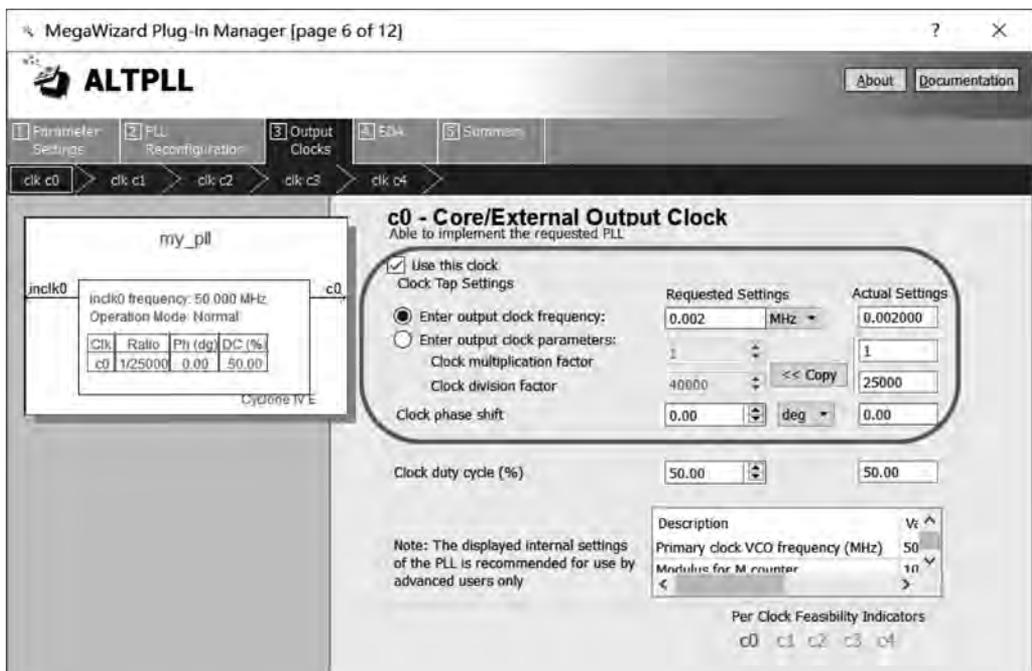


图 3.42 MegaWizard 设置 page6

在图 3.43 窗体中为 IP 生成的最后一步,用户可以根据需要对生成的文件进行勾选,文件类型说明如表 3.6 所示。本例中由于是原理图方式进行设计所以要勾选 \*.bsf 文件,然后单击 Finish 按钮完成 IP 生成。

表 3.6 MegaWizard 输出文件类型

文件类型	说明
*.bsf	原理图编辑器使用的符号文件
*.cmp	VHDL 设计中元件声明文件
*.vhd	VHDL 设计中实例化的封装文件
*.v	Verilog HDL 设计中实例化的封装文件
*_inst.vhd	VHDL 设计中实例化的模板
*_inst.v	Verilog HDL 设计中实例化的模板

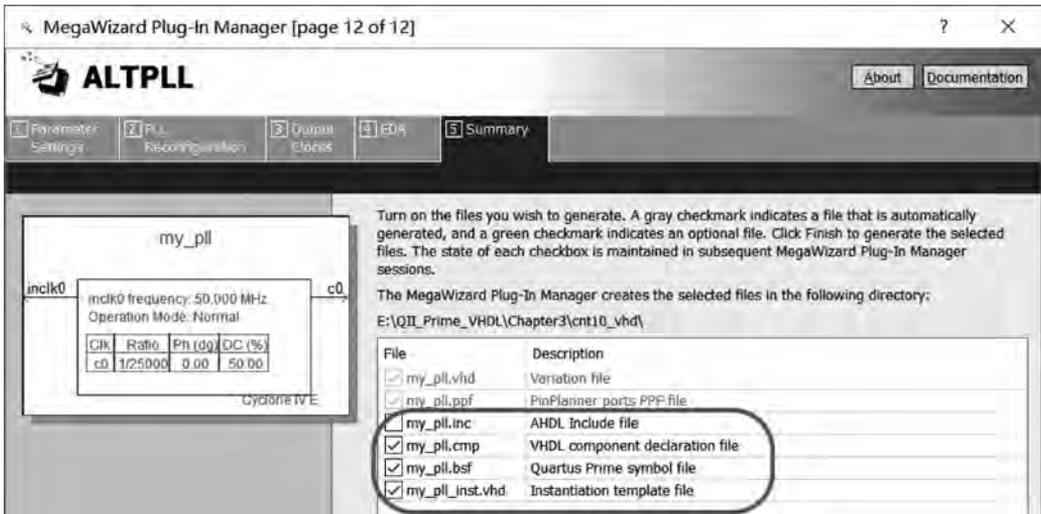


图 3.43 MegaWizard 设置 page12

打开 3.1.3 节工程文件,双击空白工作区域添加生成好的宏功能模块 my\_pll 并加入到原理图中,加入后如图 3.44 所示。然后经过编译综合布局布线最终形成配置文件,下载到 FPGA 中进行验证。如果使用 VHDL 进行设计时,需要用元件例化语句将生成的 IP 宏功能模块加入工程文件中。

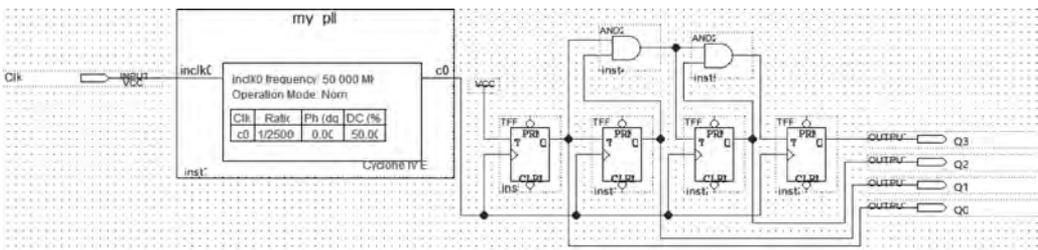


图 3.44 加入 my\_pll 后的原理图

### 3.4 SignalTap II 逻辑分析仪的应用

SignalTap II 全称 SignalTap II Logic Analyzer, 是第二代系统级调试工具, 可以捕获和显示实时信号, 观察在系统设计中的硬件和软件之间的相互作用。Quartus Prime 软件可以选择要捕获的信号、开始捕获的时间, 以及要捕获多少数据样本。还可以选择时间数据从器件的存储器块通过 JTAG 端口传送至 SignalTap II Logic Analyzer, 还是至 I/O 引脚以供外部逻辑分析仪或示波器使用。将实时数据提供给工程师帮助调试。

SignalTap II 获取实时数据的原理是在工程中引入 Megafunction 中的 ELA (Embedded Logic Analyzer), 以预先设定的时钟采样实时数据, 并存储于 FPGA 片上 RAM 资源中, 然后通过 JTAG 传送回 Quartus Prime 分析。可见, SignalTap II 其实也是在工程额外加入了模块来采集信号, 所以使用 SignalTap II 需要一定的代价, 首先是逻辑单元 (ELA), 其次是 RAM, 如果工程中剩余的 RAM 资源比较充足, 则 SignalTap II 一次可以采集较多的数据, 相应的如果 FPGA 资源已被工程耗尽则无法使用 SignalTap II 调试。

SignalTap II 任务流程如图 3.45 所示。

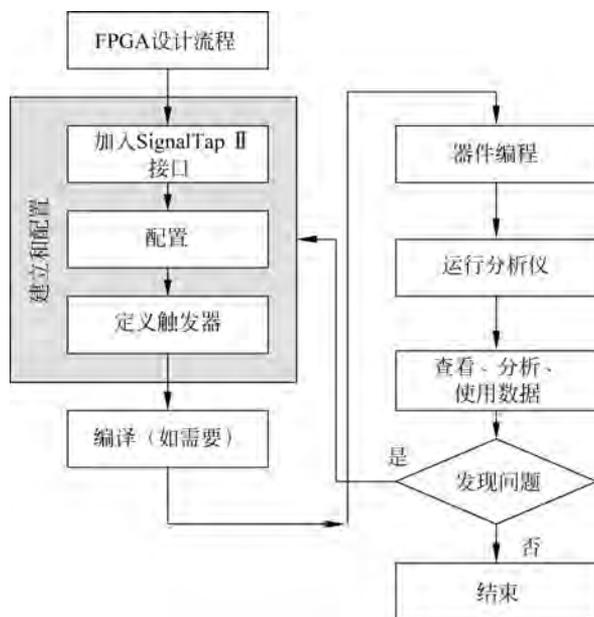


图 3.45 SignalTap II 任务流程

(1) 至少准备一个完整的 FPGA 设计工程, 以满足能够下载到 FPGA 器件中进行在线调试。

(2) 使用 .stp 文件在该工程中建立嵌入式逻辑分析仪, 并进行相关设置, 包括指定采集时钟、采样深度、触发条件、存储器模式、触发级别和添加采样信号等。

(3) 根据需要对工程进行编译。当第一次将逻辑分析仪加入到工程中, 或者对逻辑分析仪各项设置参数进行了较大改动, 例如增加了要监测的新信号, 那么需要进行编译或重新编译。对现有 SignalTap II 的某些基本改动是运行时可配置的, 例如禁用某一触发条件, 则

不需要重新编译。

(4) 将含有逻辑分析仪的设计下载至 FPGA 器件,通过 JTAG 链接来运行并控制它。

(5) 出现触发事件时,逻辑分析仪停止,采集到的数据被传送到 SignalTap II 文件窗口。用户在该窗口进行查看、分析、保存,找到设计中的问题。

(6) 调试后判断是否发现并改正了问题,如果是,则将捆绑在工程中的逻辑分析仪去掉,整个调试流程结束;相反,则重新配置逻辑分析仪,调整触发条件,再次寻找其他问题或漏洞。

本节以文本方式设计的十进制计数器为例进行说明(计数器中例化了前面的 PLL 锁相环)。利用 SignalTap II 在线调试步骤如下。

(1) 在 Quartus Prime 软件中选择菜单栏 File→New,在弹出的 New 对话框中选择 SignalTap Logic Analyzer File,如图 3.46 所示。单击后弹出如图 3.47 所示的 SignalTap II 界面。用户需要将 SignalTap II 文件保存至工程目录中。

(2) 在使用 SignalTap II 逻辑分析仪进行数据采集之前,首先应该设置采样时钟,因为逻辑分析仪是在时钟上升沿采样,推荐使用同步系统全局时钟作为采样时钟。单击图 3.47 界面中参数设置区域 clock 栏对应的浏览按钮 ,弹出如图 3.48 所示的节点查找器窗口。

选择图中 Filter 过滤选项中 Pins: all,单击图 3.48 中 List 按钮就会在节点查找区域列出所有可以观察的信号节点,选择工程中的时钟 clk 作为采样信号,选中后单击中间的“>”按钮将时钟选中至右侧区域,单击 OK 按钮。接下来定义采样深度。



图 3.46 新建 SignalTap II 文件



图 3.47 SignalTap II 界面

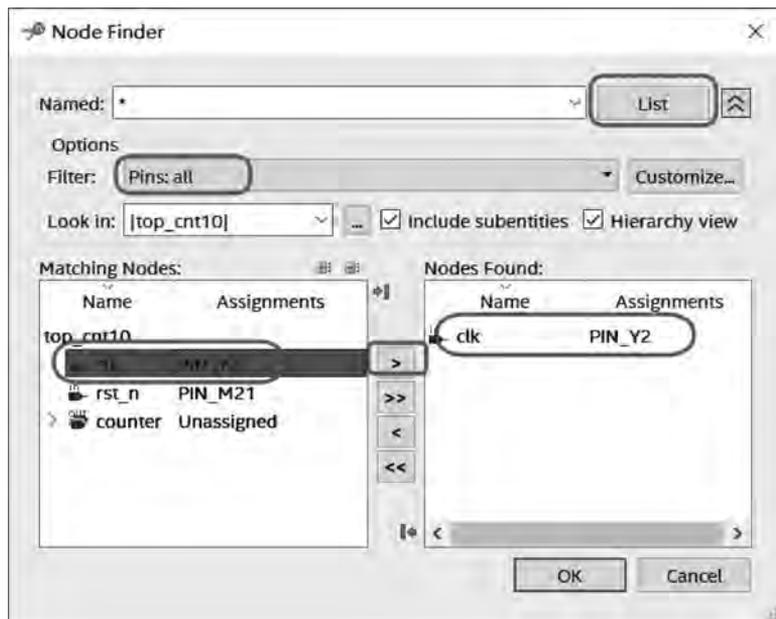


图 3.48 节点查找器窗口

采样深度决定了每个信号可存储的采样数目,设置范围在 0~128K。本界面中采样深度(Sample depth)设置为 128K。在 Trigger 栏中,Trigger position(触发位置)可以选择触发位置前后数据的比例,这里选择 Pre trigger position。

参数设置完成后需要添加被测信号。在图 3.47 左侧选择 Setup 标签页,在 Setup 标签页中双击鼠标左键,弹出 Node Finder 对话框(该标签页内有一行灰色提示 Double-click to add nodes: 双击添加节点信号)。在 Node Finder 对话框的 Filter 列表中选择 SignalTap II : pre-synthesis 或 SignalTap II : post-fitting,在 Look in 对话框中指定层次,单击 List 按钮查找节点。在 Nodes Found 中选择要加入 STP 文件中的节点或总线。单击“>”按钮将选择的节点或总线复制到 Selected Nodes 中。加载后如图 3.49 所示。

trigger: 2019/02/18 21:00:42 #1		Lock mode: Allow all changes			
Type	Alias	Node	Data Enable	Trigger Enable	Trigger Conditions
		cnt10:u1[temp[3..0]]	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	1 <input checked="" type="checkbox"/> Basic AND
		cnt10:u1[temp[3]]	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<div style="border: 1px solid black; padding: 5px; display: inline-block;">           触发条件:           <ul style="list-style-type: none"> <li><input type="checkbox"/> Don't Care</li> <li><input type="checkbox"/> Low</li> <li><input type="checkbox"/> Falling Edge</li> <li><input type="checkbox"/> Rising Edge</li> <li><input type="checkbox"/> High</li> <li><input type="checkbox"/> Insert Value</li> </ul> </div>
		cnt10:u1[temp[2]]	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
		cnt10:u1[temp[1]]	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
		cnt10:u1[temp[0]]	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	

图 3.49 待观察节点设置

用户需要在图 3.49 中设置触发条件,触发条件的设置可以帮助用户有效地调试设计。本节只介绍单个信号基本触发条件。触发条件设置分为 Basic 和 Advanced 两种。在 Basic 触发条件中 Don't Care 表示不管信号为何值均触发; Low 表示低电平触发; Falling Edge 表示下降沿触发; Rising Edge 表示上升沿触发; High 表示高电平触发; Insert Value 是给总线信号赋值,表示总线信号为此值时触发。本例选择 Don't Care。

(3) 一切设置完成后重新编译工程。

(4) 在图 3.47 中 JTAG 链配置区域单击 SOF Manage 栏中浏览按钮 , 加载工程配置文件, 单击  按钮配置下载器端口, 单击  按钮扫描 JTAG 链, 查找链路上的器件, 单击  按钮进行配置, 配置完成后自动运行分析。

(5) 当满足触发条件时, 逻辑分析仪停止, 采集到的数据被传送到图 3.47 的 DATA 窗口, 用户可以根据捕捉到的数据进行分析或调试。在线调试如图 3.50 所示。



图 3.50 在线捕捉的数据

注意 SignalTap II 逻辑分析仪是用工程设计中剩余的 RAM 块资源来存放数据的, 因此用户不可能无限制地增加采样深度或采样信号。用户在调试完成后要将 SignalTap II 的文件移除并重新编译一遍, 这样才不会影响整个工程设计的性能。

### 3.5 本章小结

本章从图形化设计和 VHDL 文本设计两个角度介绍了 Quartus Prime 软件的设计流程。另外介绍了 Altera-ModelSim 软件的基本使用方法。同时对 Quartus Prime 自带的 IP 核的使用方法做了详细介绍, 更详细的 IP 核应用读者需从各个 IP 带的用户手册中获取。为了便于在线调试 FPGA, 本章还介绍了 SignalTap II 逻辑分析仪的基本使用方法, 读者可以根据介绍的方法完成基本的在线调试功能。