



TimeQuest Timing Analyzer 快速入门 教程



101 Innovation Drive
San Jose, CA 95134
www.altera.com

UG-TMQSTANZR-1.1



反馈



订阅

© 2009? Altera Corporation. All rights reserved. ALTERA, ARRIA, CYCLONE, HARDCOPY, MAX, MEGACORE, NIOS, QUARTUS and STRATIX words and logos are trademarks of Altera Corporation and registered in the U.S. Patent and Trademark Office and in other countries. All other words and logos identified as trademarks or service marks are the property of their respective holders as described at www.altera.com/common/legal.html. Altera warrants performance of its semiconductor products to current specifications in accordance with Altera's standard warranty, but reserves the right to make changes to any products and services at any time without notice. Altera assumes no responsibility or liability arising out of the application or use of any information, product, or service described herein except as expressly agreed to in writing by Altera. Altera customers are advised to obtain the latest version of device specifications before relying on any published information and before placing orders for products or services.



1. 关于本教程

2. 快速入门教程

系统要求	2 - 1
程序	2 - 1
步骤 1: 在 Quartus II 软件打开和设置设计	2 - 1
步骤 2: 建立 TimeQuest Timing Analyzer	2 - 1
步骤 3: 执行初始编译	2 - 2
步骤 4: 运行 TimeQuest Timing Analyzer	2 - 2
步骤 5: 创建一个 Post-Map 时序网表	2 - 3
步骤 6: 指定时序要求	2 - 3
步骤 7: 更新时序网表 (Timing Netlist)	2 - 4
步骤 8: 保存 Synopsys Design Constraints (SDC) 文件	2 - 4
步骤 9: 对初始时序网表生成时序报告	2 - 5
步骤 10: 保存约束到 SDC 文件	2 - 8
步骤 11: 执行 Timing-Driven 编译	2 - 8
步骤 12: 在 TimeQuest Timing Analyzer 中验证时序	2 - 8
结论	2 - 12

3. 脚本实例

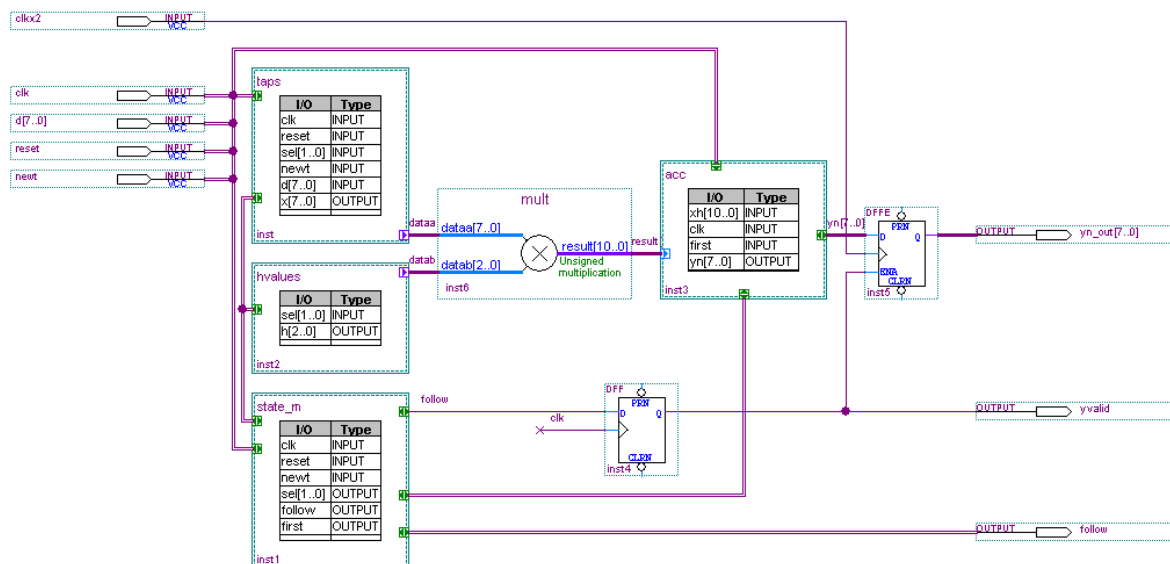
命令和 Tcl 脚本	3 - 1
----------------------	-------

附加信息

文件修订历史	Info - 1
如何联系 Altera	Info - 1
排版约定	Info - 1

本教程通过 TimeQuest Timing Analyzer 介绍了约束和执行静态时序分析的步骤。在本教程中，使用 Quartus® II 软件附带的 **fir_filter** 设计。图 1-1 显示了 **fir_filter** 的设计原理图。

图 1-1. fir_filter 设计原理图



系统要求

在本教程中，将 Stratix、Cyclone、MAX II 或者更新的器件系列（您也可以使用 MAX 3000 和 MAX 7000 系列器件）和 Quartus ® II 6.0 及更新的版本一起使用。APEX、FLEX 和 Mercury 器件系列是不支持的。

程序

使用以下步骤来约束和分析含有 TimeQuest Timing Analyzer 的设计。每个步骤包括 GUI 程序和等效命令行。

步骤 1: 在 Quartus II 软件打开和设置设计

在 Quartus II 软件中，浏览和打开位于 <qdesign folder>/fir_filter/ 文件夹中的 fir_filter。使用表 2-1 中的 GUI 或者等效命令行程序。

表 2-1. 打开并设置设计

Quartus II 软件 GUI	命令行
在 File 菜单中，点击 Open Project ，然后浏览到工程文件 <Quartus II Installation Folder>\qdesigns\fir_filter\fir_filter.qpf。	输入: quartus_sh -s ↵ project_open fir_filter -revision \ filtref ↵

步骤 2: 建立 TimeQuest Timing Analyzer

默认情况下，Quartus II 软件使用 Classic Timing Analyzer 作为采用 Cyclone 器件系列的设计的时序分析工具。指定 TimeQuest Timing Analyzer 作为 Quartus II 软件中的时序分析工具，以便用于 fir_filter 工程中的编译流程。



这一步骤不是所有工程所必须的。更新的 FPGA 系列默认为 TimeQuest Timing Analyzer。

通过表 2-2 中的程序，指定 TimeQuest Timing Analyzer 作为 Quartus II 软件中的时序分析工具。

表 2-2. 指定 TimeQuest Timing Analyzer 作为默认工具

Quartus II 软件 GUI	命令行
1. 在 Assignments 菜单上，点击 Settings。Settings 对话框出现。 2. 在 Category 列表，选择 Timing Analysis Settings 3. 打开 Use TimeQuest Timing Analyzer during compilation. 4. 单击 OK.	输入： set_global_assignment -name \ USE_TIMEQUEST_TIMING_ANALYZER ON ↵ 要关闭工程，输入：project_close exit ↵

步骤 3：执行初始编译

在应用时序约束到设计之前，通过表 2-3 中的程序创建一个初始数据库。初始数据库从设计的 post-map 结果中生成。

表 2-3. 执行初始编译 (1)

Quartus II 软件 GUI	命令行
在 Processing 菜单上，指向 Start，点击 Start Analysis & Synthesis。	输入：quartus_map filtref ↵

表 2-3 注释：

(1) quartus_map 用于创建一个 post-map 数据库。

Analysis & Synthesis 阶段生成 post-map 数据库。



您也可以为初始数据库创建一个 post-fit 网表。不过，创建一个 post-map 网表耗时较少，并且用于本教程的例子已经足够了。

步骤 4：运行 TimeQuest Timing Analyzer

通过表 2-4 中的程序，运行 TimeQuest Timing Analyzer 来创建和验证所有时序约束和例外。此命令将打开 TimeQuest shell。

表 2-4. 运行 TimeQuest Timing Analyzer

Quartus II 软件 GUI	命令行
在 Tools 菜单中，单击 TimeQuest Timing Analyzer。	输入： quartus_sta -s ↵ project_open fir_filter -revision filtref ↵



当您直接从 Quartus II 软件中运行 TimeQuest Timing Analyzer 时，当前工程将会自动打开。

如果使用 GUI，那么当出现下面的消息时，请选择 No:


"No SDC files were found in the Quartus Settings File and filtref.sdc doesn't exist. Would you like to generate an SDC file from the Quartus Settings File?"

步骤 5：创建一个 Post-Map 时序网表

在指定时序要求之前，请创建一个时序网表。您可以从 post-map 或 post-fit 数据库中创建一个时序网表。在这一步骤中，通过表 2-5 中的程序，从 “步骤 3：执行初始编译” 中创建的 post-map 数据库中创建一个时序网表。

表 2-5. 创建一个 Post-Map 时序网表

TimeQuest Timing Analyzer GUI	TimeQuest Timing Analyzer Console
1. 在 Netlist 菜单上，点击 Create Timing Netlist 。出现 Create Timing Netlist 对话框。 2. 在 Input netlist 中，选择 Post-Map 。 3. 点击 OK 。	输入: <code>create_timing_netlist -post_map</code> ↵

 您不能在Tasks面板中使用**Create Timing Netlist**命令来创建一个post-map时序网表。默认情况下，**Create Timing Netlist** 需要一个 post-fit 数据库。

步骤 6：指定时序要求

必须在 fir_filter 设计中定义两个时钟。每个时钟的属性列表请参考表 2-6。

表 2-6. fir_filter 设计中的时钟

时钟端口名称	要求
clk	50/50 占空比的 50 MHz
clkx2	60/40 占空比的 100 MHz

在 fir_filter 设计中创建时钟并通过表 2-7 中的程序分配正确的时钟端口。




 要了解关于 TimeQuest Timing Analyzer 所支持的约束的详细信息，请参考 *Quartus II 手册* 第 3 卷中的 *TimeQuest Timing Analyzer* 章节。

表 2-7. 创建时钟并分配时钟端口

TimeQuest Timing Analyzer GUI	TimeQuest Timing Analyzer Console
1. 在 Constraints 菜单中，点击 Create Clock 。出现 Create Clock 对话框。 2. 在表 2-2 中对 50 MHz 时钟指定参数。对 100 MHz 时钟重复这些步骤。	输入: <code>#create the 50 MHz (20 ns) clock</code> <code>create_clock -period 20 [get_ports clk]</code> ↵ <code>#create the 100 MHz (10 ns) clock</code> <code>create_clock -period 10 -waveform {0 6} [get_ports clkx2]</code> ↵

 默认情况下，如果未使用 `-waveform` 选项，那么 `create_clock` 命令假设 50/50 的占空比。

 要了解关于创建不同占空比时钟的详细信息，请参考 *Quartus II 手册* 第 3 卷中的 *TimeQuest Timing Analyzer* 章节。

完成表 2-7 中显示的程序后，时钟定义完成。

步骤 7：更新时序网表 (Timing Netlist)

在您创建时序约束或例外后，通过表 2-8 中的程序，对时序网表进行更新，将所有时序要求应用到时序网表（新的 clk 和 clkx2 时钟约束）。



 只要应用了新的时序约束，就必须对时序网表进行更新。

表 2-8. 更新时序网表 (Timing Netlist)

TimeQuest Timing Analyzer GUI	TimeQuest Timing Analyzer Console
在 Tasks 面板中，双击 Update Timing Netlist 命令。	输入: update_timing_netlist ↵

步骤 8：保存 Synopsys Design Constraints（SDC）文件

在为设计指定时钟约束并更新时序网表后，您可以通过表 2-9 中的程序来选择创建 SDC 文件。通过 TimeQuest Timing Analyzer GUI 或者在控制台 (console) 中指定的约束不会自动保存。

 如果您在设计流程中无意覆盖任何约束，那么请使用这个初始 SDC 文件来恢复所有约束。

初始 SDC 文件可作为包含设计的原始约束和例外的“golden”SDC 文件。

表 2-9. 保存 SDC 文件

TimeQuest Timing Analyzer GUI	TimeQuest Timing Analyzer Console
1. 在 Tasks 面板中，双击 Write SDC File 命令。出现 Write SDC File 对话框。 2. 在 File Name 栏输入 filtref.sdc。	输入: write_sdc filtref.sdc ↵

新的 **filtref.sdc** 文件包含“步骤 6：指定时序要求”中定义的两个时钟中的约束和忽略路径例外。

Write SDC File 命令可以覆盖任何现有的 SDC 文件。当这种情况出现时，新的 SDC 文件没有保持顺序或注释。因此，Altera 建议单独保存利用文本编辑器可以手动编辑的一个 golden SDC 文件。这使您能够根据自身的规范输入注释并组织文件。

步骤 9：对初始时序网表生成时序报告

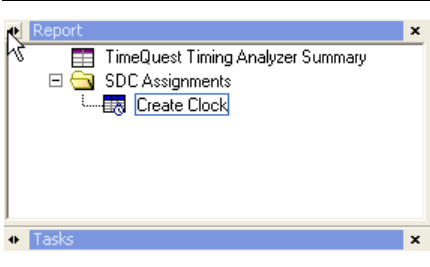
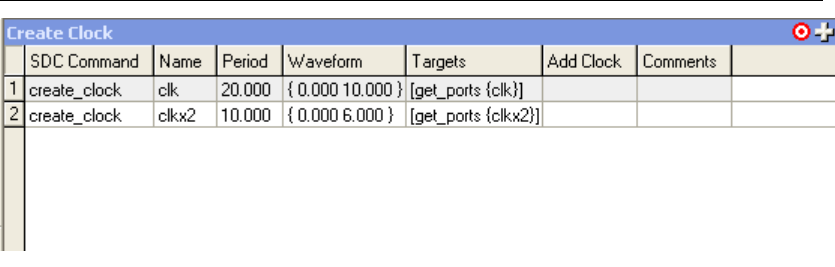
通过表 2-10 中的程序定义的两个时钟，在指定时序约束和更新时序网表后，生成时序报告，这验证了所有时钟被正确地定义并应用到正确的节点。TimeQuest Timing Analyzer 提供易于使用的报告生成命令，使您能够验证设计中的所有的时序要求。

表 2-10. 保存 SDC 命令

TimeQuest Timing Analyzer GUI	TimeQuest Timing Analyzer Console
在 Tasks 面板中，双击 Report SDC 命令。	输入: report_sdc ↵

图 2-1 显示了在 Tasks 面板中点击 Report SDC 时生成的 Create Clock 报告。

图 2-1. 生成 SDC 约束报告 (SDC Assignments Report)

	
---	--

SDC Assignments 报告了在指定设计中包含的所有时序约束和例外。生成两个报告：一个用于时钟和一个用于时钟组。

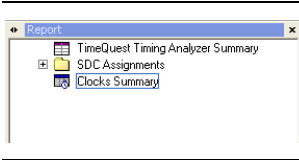
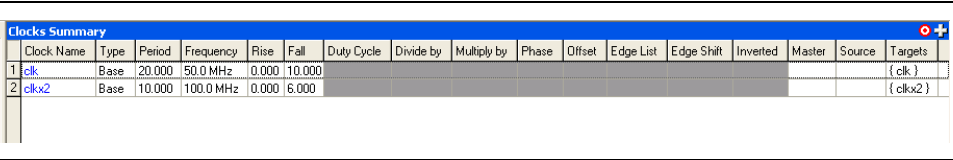
可以用表 2-11 中的程序生成一个报告，来总结设计中所有的时钟。

表 2-11. 生成报告时钟报告 (Report Clocks Report)

TimeQuest Timing Analyzer GUI	TimeQuest Timing Analyzer Console
在 Tasks 面板中，双击 Report Clocks 命令。	输入: report_clocks ↵

图 2-2 显示了时钟总结 (Clocks Summary) 报告。

图 2-2. 时钟总结报告

	
---	--

通过表 2-12 中的程序，使用 Report Clock Transfers 命令生成一个报告来验证所有的时钟到时钟传输都是有效的。这种报告包含设计中所有的时钟到时钟传输。

表 2-12. 生成报告时钟传输 (Report Clock Transfers)

TimeQuest Timing Analyzer GUI	TimeQuest Timing Analyzer Console
在 Tasks 面板中，双击 Report Clock Transfers 命令。	输入: report_clock_transfers ↵

图 2-3 显示了 Clock Transfers 报告。

图 2-3. 时钟传输报告 (Clock Transfers Report)


Report		Setup Transfers					
		From Clock	To Clock	RR Paths	FR Paths	RF Paths	FF Paths
1	clk	clk	clkx2	19504	0	0	0
2	clk	clkx2	clkx2	16	0	0	0

Clock Transfers 报告表明在 `clk`（源时钟）和 `clkx2`（目的时钟）之间存在跨时钟域路径。共有 16 条路径，其中 `clk` 为源节点提供时钟，`clkx2` 为目的节点提供时钟。

在 `fir_filter` 设计中，不必分析 `clk` 至 `clkx2` 的时钟传输，因为它们是忽略路径。通过表 2-13 中的程序声明 `clk` 至 `clkx2` 的路径为伪路径。当完成该程序后，TimeQuest Timing Analyzer 表明 Clock Transfers 报告是过时的。

表 2-13. 声明伪路径

TimeQuest Timing Analyzer GUI	TimeQuest Timing Analyzer Console
<div>1. 在 Clock Transfers 报告中，在 From Clock 列选择 <code>clk</code>。</div> <div>2. 右击并选择 Set False Paths Between Clock Domains。这个命令表明将所有由 <code>clk</code> 驱动的源寄存器到由 <code>clkx2</code> 驱动的目的寄存器之间的路径设为伪路径。</div>	<div>输入：</div> <div>set_false_path -from [get_clocks clk] \ -to [get_clocks clkx2] ↵</div>

 另外，也可以使用 `set_clock_groups` 命令来声明两个时钟域之间的路径为伪路径。例如，`set_clock_groups -asynchronous -group [get_clocks clk] -group [get_clocks clkx2]`。该命令表明 `clk` 到 `clkx2` 以及 `clkx2` 到 `clk` 的所有路径为伪路径。此方法是优选的。

由于您添加了一个新的时序约束，通过表 2-14 中的程序更新时序网表 (timing netlist)。

表 2-14. 更新 Timing Netlist

TimeQuest Timing Analyzer GUI	TimeQuest Timing Analyzer Console
在 Tasks 面板中，双击 Update Timing Netlist 命令。	输入: update_timing_netlist ↵

在 GUI 中输入 `set_false_path` 后，所有生成的报告面板上都标有 “Out of Date”，这表明报告面板不包含反映 TimeQuest Timing Analyzer 中当前状态的约束或者例外的结果。要更新报告面板，必须重新生成所有的报告。

在命令行，重新输入命令。在 GUI 中，右击报告面板列表中任何过时 (out-of-date) 的报告，并选择 **Regenerate** 或 **Regenerate all**。

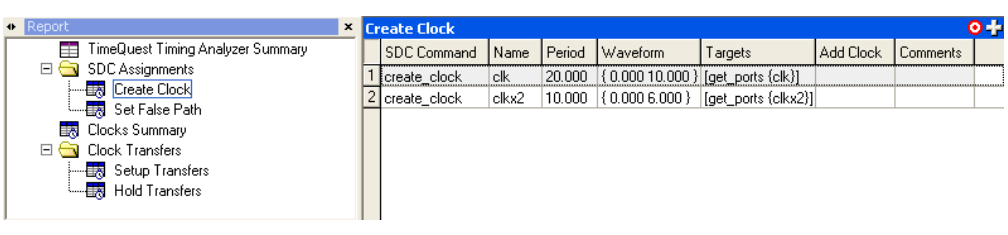
更新时序网表后，通过表 2-15 中的程序验证时钟到时钟传输已经被声明为伪路径。

表 2-15. 验证使用 Report SDC 命令

TimeQuest Timing Analyzer GUI	TimeQuest Timing Analyzer Console
在 Tasks 面板中，双击 Report SDC。	输入: <code>report_sdc</code> ↵

图 2-4 显示了新的 SDC Assignments 报告。

图 2-4. SDC Assignments 报告

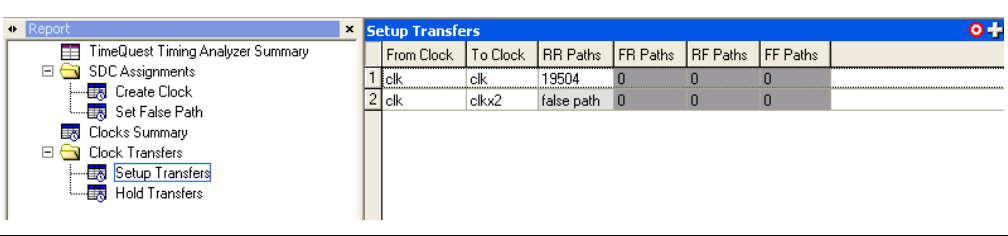


SDC Command	Name	Period	Waveform	Targets	Add Clock	Comments
1 create_clock	clk	20.000	{ 0.000 10.000 }	[get_ports {clk}]		
2 create_clock	clkx2	10.000	{ 0.000 6.000 }	[get_ports {clkx2}]		


图 2-4 中显示的报告表明时钟约束和伪路径是正确的。

使用 **Report Clocks** 和 **Report Clock Transfers** 命令来验证这两个时钟已经从分析中移除了。图 2-5 显示了 Clock Transfers 报告。

图 2-5. Clock Transfers 报告



From Clock	To Clock	RR Paths	FR Paths	RF Paths	FF Paths
1 clk	clk	19504	0	0	0
2 clk	clkx2	false path	0	0	0


 **RR Paths** 列包含注解 “false path” 以表示声明时钟域为伪路径。

步骤 10：保存约束到 SDC 文件

在指定设计中所有的时钟约束和伪路径后，通过表 2-16 中的程序将时序约束和例外保存到 SDC 文件。

表 2-16. 保存约束到 SDC 文件

TimeQuest Timing Analyzer GUI	TimeQuest Timing Analyzer Console
1. 在 Tasks 面板中，双击 Write SDC File 。出现 Write SDC File 对话框。 2. 在 File name 栏，输入 <code>filtref.sdc</code> 。	输入: <code>write_sdc filtref.sdc</code> ↵

 这一过程覆盖之前所创建的 `filtref.sdc` 文件。如果通过 **Write SDC File** 命令覆盖 SDC，那么在新的 SDC 文件中移除了定制的格式和注释。

`filtref.sdc` 文件包含两个时钟约束和伪路径例外。

步骤 11：执行 Timing-Driven 编译

保存约束到 SDC 文件后，在设计上运行一个全编译以优化布线，从而符合约束。不过，在开始全编译之前，通过表 2-17 中的程序将 SDC 添加到工程中。

表 2-17. 添加 SDC 文件到工程中

TimeQuest Timing Analyzer GUI	TimeQuest Timing Analyzer Console
1. 在 Project 菜单中，点击 Add/Remove Files In Project 。出现 Add/Remove Files In Project 对话框。 2. 通过浏览来选择 <code>.sdc</code> 。 3. 点击 OK 。	输入: <code>set_global_assignment -name SDC_FILE \</code> <code>filtref.sdc</code> ↵

将 SDC 添加到工程后，通过表 2-18 中的程序，在设计上运行一个全编译。


表 2-18. 运行一个全编译

TimeQuest Timing Analyzer GUI	TimeQuest Timing Analyzer Console
在 Processing 菜单中，点击 Start Compilation 。	输入: <code>quartus_sh --flow compile filtref</code> ↵

完成编译后，TimeQuest Timing Analyzer 在 **Compilation Report** 中生成时钟建立和时钟保持的检查总结报告。

步骤 12：在 TimeQuest Timing Analyzer 中验证时序


要获得指定路径中详细的时序分析数据，请查看 TimeQuest Timing Analyzer 中的时序分析结果。

 完全执行布线布局功能 (place-and-route) 后，运行 “[步骤 4 运行 TimeQuest Timing Analyzer](#)” 中所介绍的 TimeQuest Timing Analyzer。

生成一个 post-fit 时序网表，通过表 2-19 中的程序，读取 SDC 文件并更新时序网表来生成关于最新编译的报告。

表 2-19. 生成关于 Latest Compilation 的报告

TimeQuest Timing Analyzer GUI	TimeQuest Timing Analyzer Console
在 Tasks 面板中，双击所需报告的命令。例如：Report All Summaries。	输入： create_timing_netlist ↵ read_sdc filref.sdc ↵ update_timing_netlist ↵ report_clocks ↵ create_timing_summary -setup ↵ create_timing_summary -hold ↵ create_timing_summary -recovery ↵ create_timing_summary -removal ↵ report_min_pulse_width -nworst 10 ↵

 当双击其中一个报告命令时，Create Timing Netlist、Read SDC 和 Update Timing Netlist 命令依次在 Tasks 面板中执行，自动生成时序网表。

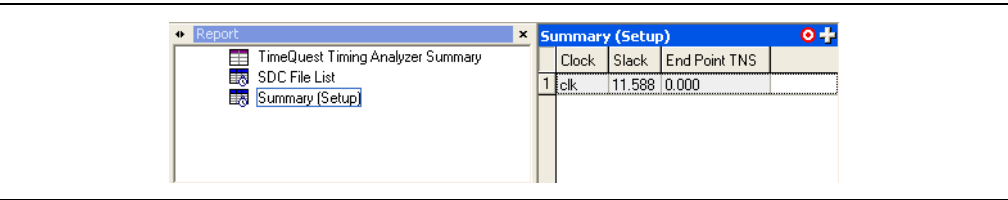
时钟建立检查确保每个寄存器至寄存器的传输不违反 SDC 指定的时序约束。通过表 2-20 中的程序，生成一个时钟建立总结，对设计中的所有时钟进行检查，来验证没有出现违规。


表 2-20. 生成时钟设置总结检查 (Clock Setup Summary Check)

TimeQuest Timing Analyzer GUI	TimeQuest Timing Analyzer Console
在 Tasks 面板中，双击 Report Setup Summary。	输入: create_timing_summary -setup ↵

图 2-6 显示了 Summary (Setup) 报告。

图 2-6. Summary (Setup) 报告



 clkx2 时钟没有出现在 Summary (Setup) 报告中，这是因为 clk 和 clkx2 之间所有的时钟路径已经声明是伪路径。此外，fir_filter 设计不包含任何寄存器到寄存器的路径，其中目的寄存器路径由 clkx2 来驱动。

Summary (Setup) 报告中的 Slack 列表明 clk 能满足约束，并有 11.588 ns 的余量。End Point TNS 列是指定时钟域中所有的总负裕量（TNS）的总和。使用这个值来测量指定时钟域中失败路径的总数。

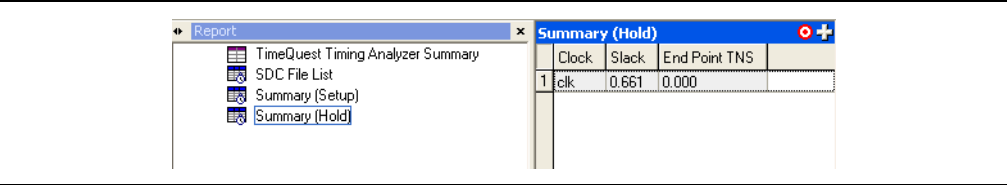
生成 Summary (Setup) 报告后，通过表 2-21 中的程序，在设计中生成一个时钟保持检查总结。

表 2-21. 生成 Summary (Hold) Report

TimeQuest Timing Analyzer GUI	TimeQuest Timing Analyzer Console
在 Tasks 面板中，双击 Report Hold Summary。	输入: create_timing_summary -hold ↵

图 2-7 显示了 Summary (Hold) 报告。

图 2-7. Summary (Hold) 报告



Summary (Hold) 报告表明 clk 时钟节点符合时序约束，并有 0.661 ns 的余量。

在执行全编译之前，通过表 2-22 中的程序指定所有的时序约束和例外。这样可以确保 Fitter 优化设计中的关键路径。

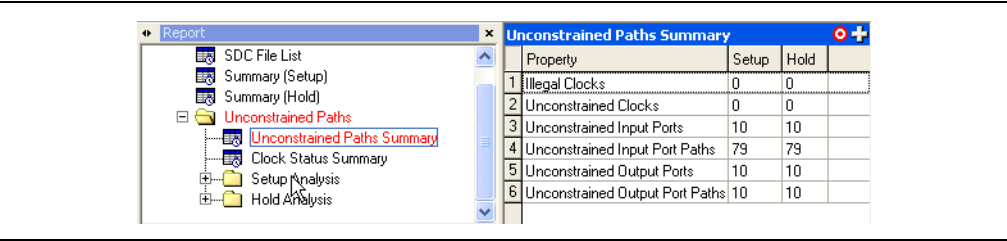
您可以使用 Report Unconstrained Paths 命令来验证已经约束 fir_filter 设计中的所有路径。

表 2-22. 指定时序约束和例外

TimeQuest Timing Analyzer GUI	TimeQuest Timing Analyzer Console
在 Tasks 面板中，双击 Report Unconstrained Paths。	输入: report_ucp ↵

图 2-8 显示了 Unconstrained Paths 总结报告。

图 2-8. Unconstrained Paths 总结报告



Unconstrained Paths 总结报告表明有大量的未约束路径，并详细介绍了这些路径的类型。

要充分约束此设计，利用由 TimeQuest Timing Analyzer 所提供的整套 SDC 约束。


要充分约束 fir_filter 设计，约束所有的输入和输出端口。使用 Set Input Delay 和 Set Output Delay 对话框，或 set_input_delay 和 set_output_delay 约束来指定输入和输出延迟值。

由于附加约束应用于设计，通过文本编辑器（例如：`inout_delay.sdc`）创建仅包含输入和输出约束的额外 SDC。添加表 2-23 所示的输入和输出延迟分配到 “步骤 10: 保存约束到 SDC 文件” 创建的新 SDC 中。

表 2-23. 输入和输出延迟分配

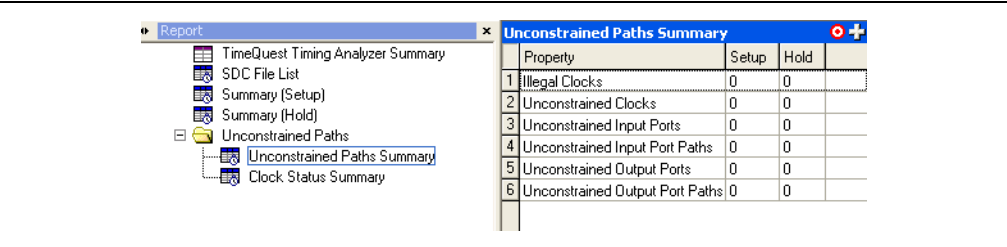
The TimeQuest Timing Analyzer GUI	The TimeQuest Timing Analyzer Console
<div>1. 在 Constraints 菜单中，点击 Set Input Delay。出现 Set Input Delay 对话框。</div> <div>2. 输入以下内容： Clock name: clk Delay value: 2 Targets: [get_ports {d[0] d[1] d[2] d[3] \ d[4] d[5] d[6] d[7] newt reset}]</div> <div>3. 在 Constraints 菜单中，点击 Set Output Delay。出现 Set Output Delay 对话框。</div> <div>4. 输入以下内容： Clock name: clk Delay value: 1.5 Targets: [get_ports {yn_out[0] yn_out[1] \ yn_out[2] yn_out[3] yn_out[4] yn_out[5] \ yn_out[6] yn_out[7] yvalid follow}]</div>	<div>要约束输入端口，输入： set_input_delay -clock clk 2 \ [get_ports {d* newt reset}] ↵</div> <div>要约束输出端口，输入： set_output_delay -clock clk 1.5 \ [get_ports {yn_out* yvalid follow}] ↵</div>

在读取包含输入和输出延迟约束的 SDC 后，所有设计中的端口应该都加上了约束。

 记住读取新的约束后更新时序网表。要了解更多信息，请参考 “步骤 7: 更新时序网表 (Timing Netlist)”。

要验证所有设计中的端口都已经加上了约束，重新生成 Unconstrained Paths Summary 报告（图 2-9）。

图 2-9. 重新生成 Unconstrained Paths Summary 报告



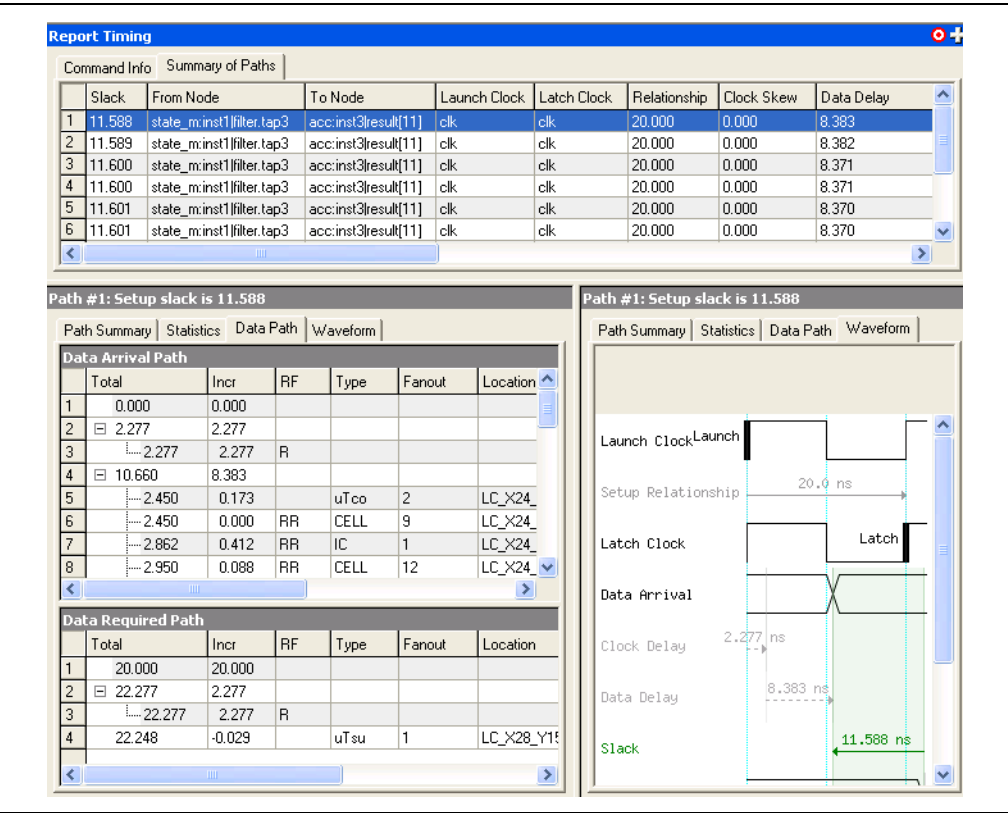
通过表 2-24 中的程序，对设计的时钟或节点生成特定的时序检查报告。表 2-24 中的程序生成一个报告，其中 clk 驱动目的寄存器，并且目的寄存器为 acc:inst3|result，报告 10 条最差路径。

表 2-24. 生成 Report Timing 报告

TimeQuest Timing Analyzer GUI	TimeQuest Timing Analyzer Console
<div>1. 在 Tasks 面板中，双击 Report Timing。出现 Report Timing 对话框。</div> <div>2. 输入以下内容： To Clock: clk To: acc:inst3 result* Report number of paths: 10</div> <div>3. 剩下其它的栏使用默认的设置。</div>	<div>输入：</div> <div>report_timing -to_clock clk -to / acc:inst3 result* -setup -npaths 10 ↵</div>

图 2-10 显示了 Report Timing 报告。

图 2-10. Report Timing 报告



使用 **Tasks** 面板中的 **Report Top Failing Paths** 命令来生成一个报告，该报告详细介绍了设计中的最差的失败路径。

结论

您创建新的约束或例外，基于新的约束或例外重新运行 Quartus II Fitter 来优化设计。设计上的多个迭代对于完成所需的结果是必要的。

命令和 Tcl 脚本

本章节包含命令及相应的 Tcl 脚本以便从命令行中执行整个流程。使用该方法来完整地执行整个流程。

在命令符行窗口中输入例 1 中的命令来接收脚本。

例 1. 接收脚本

```
quartus_sh -t timequest_setup.tcl ←
quartus_sta -t main_postmap.tcl ←
quartus_sh -t fit_sdc_setup.tcl ←
quartus_sta -t main_postfit.tcl ←
```

例 2 显示了 timequest_setup.tcl 脚本的内容。使用此脚本将 TimeQuest Timing Analyzer 指定为默认的时序分析工具。



Classic Timing Analyzer 是 Quartus II 软件中的默认时序分析器。

例 2. timequest_setup.tcl 脚本

```
#open the filtref project
project_open filtref ←
#set the TimeQuest analyzer as the default timing analyzer
set_global_assignment -name USE_TIMEQUEST_TIMING_ANALYZER ON ←
#close the project
project_close ←
```

例 3 显示了 main_postmap.tcl 脚本的内容。使用该脚本来创建 post-map 数据库，建立时序网表，在 golden.sdc 中读取，并生成设计的初始报告。

例 3. main_postmap.tcl 脚本

```
#file main_postmap.tcl
#Include the flow package to create a post-map netlist
package require ::quartus::flow ←
#open the project in TimeQuest
project_open filtref ←
#create a post-map database
execute_module -tool map ←
#create the timing netlist based on the post-map results
create_timing_netlist -post_map ←
#read in the constraints from the golden SDC file
read_sdc golden.sdc ←
#update the timing netlist with the new constraints
update_timing_netlist ←
#generated a clock report
report_clocks ←
#generated a clock-to-clock report
report_clock_transfers ←
#delete our post-map timing netlist
delete_timing_netlist ←
#close the TimeQuest project
project_close ←
```

例 4 显示了 fit_sdc_setup.tcl 脚本的内容。使用该脚本将 golden.sdc 文件添加到 filtref 设计。这允许 Quartus II Fitter 根据指定的约束对设计进行优化。

例 4. fit_sdc_setup.tcl 脚本

```
#open the filtref project
project_open filtref ←
#add the filtref.sdc file to our Quartus II project
set_global_assignment -name SDC_FILE golden.sdc ←
#close the project
project_close ←
```

例 5 显示了 main_postfit.tcl 脚本的内容。使用该脚本来创建 post-map 数据库，建立时序网表，在 golden.sdc 和 io_cons.sdc 文件中读取，并生成设计的报告。

例 5. main_postfit.tcl 脚本

```
#Include the flow package to create a post-fit netlist
package require ::quartus::flow ←
#open the project in TimeQuest
project_open filtref ←
#create a post-fit database
execute_module -tool fit ←
#create a post-fit timing netlist
create_timing_netlist ←
#read the golden SDC file and the I/O SDC file
read_sdc golden.sdc ←
read_sdc io_cons.sdc ←
#update the post-fit timing netlist with constraints
update_timing_netlist ←
#report unconstrained paths
report_clocks ←
create_timing_summary -setup ←
create_timing_summary -hold ←
create_timing_summary -recovery ←
create_timing_summary -removal ←
report_ucp ←
#delete our post-map timing netlist
delete_timing_netlist ←
#close the TimeQuest project
project_close ←
```

例 6 和例 7 分别显示了 golden.sdc 和 io_cons.sdc 文件的内容。

例 6. golden.sdc 文件

```
#create the 50 MHz 50/50 clock
create_clock -period 20 [get_ports clk] ←
#create the 100 MHz 60/40 clock
create_clock -period 10 -waveform {0 6} [get_ports clkx2] ←

#cut the clk and clkx2 domains
set_clock_groups -group [get_clocks clk] -group [get_clocks clkx2] ←
```

例 7. io_cons.sdc 文件

```
#set the input delays for the design
set_input_delay -clock clk 1.0 [get_ports {d[*] reset newt}] ←
#set the output delays for the design
set_output_delay -clock clk 1.5 [get_ports {yn_out[0] yn_out[1] \
yn_out[2] yn_out[3] yn_out[4] yn_out[5] yn_out[6] yn_out[7] yvalid follow}] ←
```

这一章节提供了关于文件 Altera 和相关文档的附加信息。

文件修订历史

下表列出了本文档的修订历史。

日期	版本	修订内容
2009 年 2 月	1.1	<ul style="list-style-type: none"> ■ 更新了第 2 章中的图表。 ■ 针对 Quartus II 9.1 功能性章节进行了更新。
2006 年 5 月	1.0	首次发布。

如何联系 Altera

要查找有关 Altera 产品最新的信息，请参考下表。

联系 (1)	联系方法	网址
技术支持	网站	www.altera.com/support
技术培训	网站	www.altera.com/training
	电子邮件	custrain@altera.com
产品简介	网站	www.altera.com/literature
非技术性支持（一般） （软件许可）	电子邮件	nacomp@altera.com
	电子邮件	authorization@altera.com

表格注释：

(1) 可联系当地的 Altera 销售办事处或销售代表。

排版约定

下表显示了本文档中使用的排版约定。

视觉提示	含义
粗体大写字母	表示命令名、对话框标题、对话框选项和其它的 GUI 标签。例如：Save As 对话框。对于 GUI 单元，大写符合 GUI。
粗体字体	表示目录名、项目名、磁盘驱动器名、文件名、文件扩展名、软件工具名和 GUI 标签。例如：\qdesigns 目录、D: 驱动器和 chiptrip.gdf 文件。
<i>斜体大写字母</i>	表示文件标题。例如：Stratix IV 设计指南。
<i>斜体字体</i>	表示变量。例如：n + 1。 变量名括在尖括号 (< >) 里。例如：<file name> 和 <project name>.pof 文件。
大写字母	表示键盘键和菜单名。例如：Delete 键和 Options 菜单。
“ 副标题 ”	引号表示参考文档的章节，以及 Quartus II 帮助的标题。例如：“ 排版约定 ”。

视觉提示	含义
Courier 字体	<p>表示信号、端口、寄存器、位、模块和原始名。例如: data1、tdi 和 input。后缀 n 表示有效低电平信号。例如: resetn。</p> <p>表示命令行的命令以及任何与原文显示完全一样的输入。例如: c:\qdesigns\tutorial\chiptrip.gdf.</p> <p>也表示实际文件的部分, 例如: 报告文件 (Report File), 参考部分文件 (例如: AHDL 关键字 SUBDESIGN), 以及逻辑功能名称 (例如: TRI)。</p>
	回角箭头指示您按回车键。
1., 2., 3., 和 a., b., c., 等等	编号的步骤显示当项目的序列很重要时的列表项目, 例如: 在一个程序中列出的步骤。
■ ■ ■	项目符号显示当项目的序列不是很重要时的列表项目。
	手指图标表示需要特别注意的信息。
	问号指示您查找系统软件帮助的相关信息。
	脚印指示您查找另外的文档或网站的相关信息。
	多媒体图标指示到相关的多媒体演示。
	注意图标提醒用户注意可能损坏产品和破坏工作的条件或可能发生的情况。
	警告图标提醒用户注意可能导致伤害的条件或可能发生的情况。
	信封链接到 Altera 网站的 Email Subscription Management Center 页面, 您可以登记以接收 Altera 文档的更新通知。
	反馈图标可以提交有关文档的反馈到 Altera。收集反馈意见的方法针对每个文档有所不同。