Intel® Quartus® Prime Pro Edition 用户指南

设计优化

针对 Intel® Quartus® Prime 设计套件的更新：**19.3**

本翻译版本仅供参考，如果本翻译版本与其英文版本存在差异，则以英文版本为准。某些翻译版本尚未更新对应到最新的英文版本，请参考英文版本以获取最新信息。

在线版本
发送反馈
UG-20133
ID: 683641
版本：2019.09.30

内容

1. 设计优化概述

典型设计流程中，开发的早期阶段专注于满足时序，面积和功耗目标。一旦设计达到这些目标后，则集中投入于性能提升。 本章介绍 Intel® Quartus® Prime 软件中可用以实现最高设计性能的技术和工具。

FPGA 设计优化需要一种多维方式来实现减小面积，缩短关键路径延迟，降低功耗和减少运行时间的设计目标。 Intel Quartus Prime 软件中包含的各种向导可针对解决每个问题。通过实现向导中的建议，可减少设计迭代所花费的时间。

相关链接

- 编译流程
  Intel Quartus Prime Pro Edition 用户指南：编译器
- Intel Quartus Prime 设计软件-支持中心

1.1. 器件考量

所有 Intel FPGA 具有独特时序模型，其中包含了器件中物理元件的延迟信息，例如组合式自适应逻辑模块，存储器块，交互连接和寄存器。延迟包括目标 FPGA 操作条件的全部有效组合。此外，器件尺寸和封装决定管脚约束和资源可用性。

相关链接

利用 FPGA 时序模型保证芯片性能
Intel FPGA 白皮书(PDF)

1.1.1. 器件迁移考量

如果预计在设计周期的后期更改目标器件，则请从周期开始就进行迁移规划。该策略有助于将后期设计变更最小化。

在 Intel Quartus Prime 软件中选择设计的目标器件时， 通过点击 **Device** 对话框中的 **Migration Devices** 按钮可就会看到可兼容器件列表。

相关链接

移植器件对话框
Intel Quartus Prime Help 中

1.2. 初始编译的必需设置

根据您选择的约束和设置，编译结果会有显著差异。 Intel Quartus Prime 软件中，各设置和选项的默认值提供了编译时间，资源利用和时序性能之间的最佳权衡。在 Intel Quartus Prime 软件中编译设计之前，请先考虑如下指导。

1.2.1. I/O 约束指导

在 FPGA 设计中，I/O 标准和驱动强度会影响 I/O 时序。

- 指定 I/O 约束时，请确保 Intel Quartus Prime 软件使用精确的 I/O 时序延迟进行时序分析和 Fitter 优化。
- 如果 PCB 布局未指明管脚位置，则保留管脚位置无约束。该技术可支持 Compiler 搜索最佳布局。否则，进行管脚约束以适当约束编译。

相关链接

I/O 规划概述
In *Intel Quartus Prime Pro Edition User Guide: Design Constraints*

1.2.2. 时序约束指导

为获得最佳结果，请使用实时要求。如果应用高于设计需要的时序要求，则可能导致 Compiler 提高资源使用，电源利用率或编译时间以进行协调。

综合型时序要求设置可实现最佳效果，原因如下：

- 正确的时序约束能使软件充分运行从而实现设计中关键时序部分的性能优化并同时协调性能。该优化还可设计中非关键部分的使用面积和电源利用率。
- 使能后， Intel Quartus Prime 软件基于时序要求执行物理综合优化。

Intel Quartus Prime Timing Analyzer 决定设计实现是否符合时序要求。Compilation Report 显示设计是否满足时序要求，而时序分析报告命令提供有关时序路径的详细信息。

相关链接

- 时序收敛与优化 (第 52 页)
- 使用 Intel Quartus Prime Timing Analyzer
  In *Intel Quartus Prime Pro Edition User Guide: Timing Analyzer*
- Intel Quartus Prime Timing Analyzer 技术参考书
- 高级设置（Fitter）
  Intel Quartus Prime Help 中

1.3. 权衡和限制

各优化目标之间可能会相互冲突，因此需要协调相冲突的目标。

表 1. 设计优化中的权衡示例

权衡（Trade-off）	备注
资源使用和关键路径时序。	某些技术（如逻辑复制）可以增加面积为代价来改善时序性能。
功率要求会影响面积和时序权衡。	例如，减少可用高速 tile 的数量，或尝试缩短高功率网路牺牲关键路径网络。
系统成本和上市时间考量因素会影响对器件的选择。	例如，具有较高速度等级或更多时钟网络的器件可通过更高功耗和系统成本促进时序收敛。

最后，过于严格限制设计可行性的约束可能导致所选器件无可用解决方案。如果因资源限制，时序约束或功率约束导致 Fitter 无法解析设计，则请考虑改写 HDL 代码部分。

1.3.1. 减少面积

默认情况下， Intel Quartus Prime Fitter 可将设计物理性扩展至整个器件，以满足时序约束。如果希望通过优化设计占用最小面积，则可更改此行为。如果需要缩小面积，则可启用特定物理综合选项修改网表以创建更具区域效率的实现，但代价是增加运行时间和降低性能。

相关链接

- 区域优化 (第 42 页)
- 网表优化和物理综合 (第 37 页)

1.3.2. 减短关键路径延迟

为满足涉及多个时钟，布线资源和面积约束的复杂时序要求， Intel Quartus Prime 软件提供介于综合，布局规划编辑，布局布线以及时序分析处理之间的紧密交互。

默认情况下，运行 Intel Quartus Prime Fitter 以满足时序要求，并在达到要求后停止。因此，实际约束对时序收敛至关重要。

收敛不足的设计会导致结果欠佳。而对于过度收敛的设计，Fitter 可能会牺牲真正的关键路径而过度优化非关键路径。此外，面积和编译时间也可能随之增加。

对于资源使用率高的设计， Intel Quartus Prime Fitter 可能无法找到合法位置。该情况下，Fitter 会自动修改设置以尝试协调面积方面的性能。

Intel Quartus Prime Fitter 提供的高级选项可在您正确设置约束时帮助提高设计性能。请使用 Timing Optimization Advisor 确定适合您设计的最佳选项。

高密度 FPGA 中，布线占据关键路径时序的主要部分。因此，复制或重新定时逻辑可促进 Fitter 减少关键路径上的延迟。 Intel Quartus Prime 软件提供的网表优化按钮和物理综合选项可提高设计性能，但会大幅增加编译时间和面积。仅开启有助于保持合理编译时间和资源使用的选项。或者，可修改 HDL 以手动复制或调整时序逻辑。

相关链接

关键路径 (第 52 页)

1.3.3. 降低功耗

Intel Quartus Prime 软件具有帮助降低功耗的功能。功率优化选项控制用于综合和适配（Synthesis and the Fitter）的电源驱动编译设置。

相关链接

功耗优化
In *Intel Quartus Prime Pro Edition User Guide: Power Analysis and Optimization*

1.3.4. 减少运行时间

许多 Fitter 设置会影响编译时间。 Intel Quartus Prime 软件中的大部分默认设置都以减少编译时间为目标。也可基于您工程的要求修改这些设置。

Intel Quartus Prime 软件支持在多处理器计算机中并行编译。这样编译时间可最多减少 15%。

相关链接

缩短编译时间
Intel Quartus Prime Pro Edition 用户指南：编译器

1.4. Intel Quartus Prime 软件工具用于设计优化

Intel Quartus Prime 软件提供各种优化设计的工具。

1.4.1. Design Visualization Tool（设计可视化工具）

Intel Quartus Prime 软件提供可显示设计图形表达的工具。

表 2. Visualization Tool（可视化工具）

工具	说明
RTL Viewer	提供综合和布局布线前的设计原理图。
technology map viewer（查看 FPGA 中的实际连线情况）	提供综合和布局布线后所选器件体系结构中设计实现的原理图。可选择性包含时序信息。
Design Partition Planner	以分区或实体级显示设计，并可显示实体间的连接。
Design Partition Planner 和 Chip Planner	允许在更高级对设计进行分区和布局。
Chip Planner	支持进行布局规划约束，执行功率分析和可视化关键路径和路由拥塞。

相关链接

- Chip Planner 中的设计平面布局分析 (第 96 页)
- 使用 Design Partition Planner 和 Chip Planner 创建分区和 Logic Lock 区域。 (第 109 页)
- 优化设计网表 (第 14 页)
- RTL Viewer 概述 (第 16 页)

1.4.2. 向导（Advisors）

Intel Quartus Prime 软件包含多种向导以帮助优化设计并缩短编译时间。

各向导根据工程设置和设计约束提供建议。这些建议有助于适配工程，满足时序或功率要求，或提高设计性能。

向导中的建议从宽泛到具体。适用情况下，其以阶段分门别类并按复杂度呈现。

向导分为：

- Timing Optimization Advisor（时序优化）
- Power Optimization Advisor（功率优化）
- Compilation Time Advisor（编译时间）

相关链接

- 编译时间向导
  Intel Quartus Prime Pro Edition 用户指南：编译器
- Intel Quartus Prime 软件中的向导 (第 0 页)
  Intel Quartus Prime Help 中

- 时序优化向导 (第 71 页)
- Power Optimization Advisor
  In *Intel Quartus Prime Pro Edition User Guide: Power Analysis and Optimization*

1.4.3. 设计管理

Design Space Explorer II 工具（DSE II）为您运行设计设置实验提供便捷有效的方法。可在 PC 上局部运行单个编译，也可远程使用运算中心（compute farm）资源。

相关链接

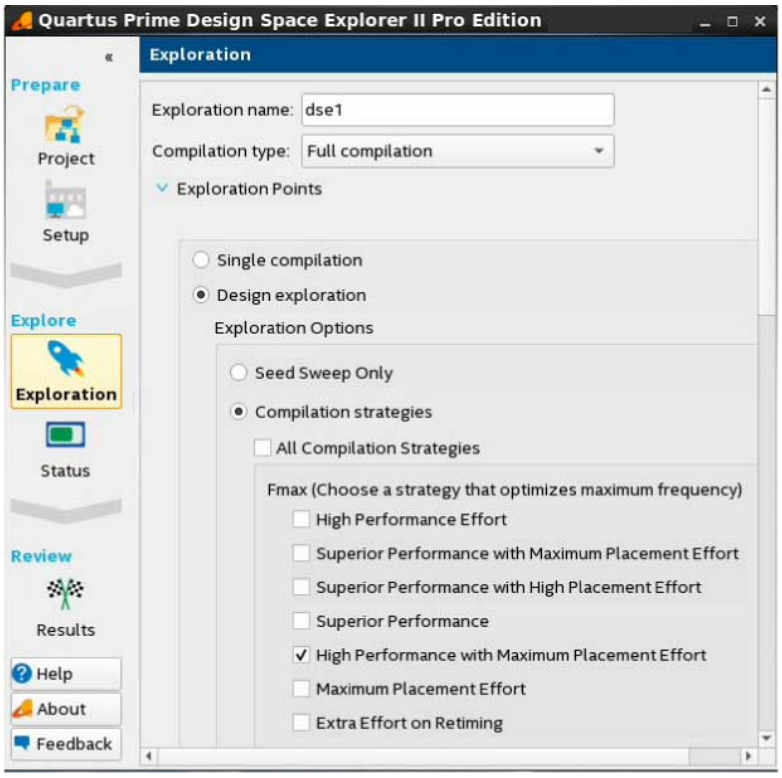
Design Space Explorer II (第 10 页)

1.5. Design Space Explorer II

Design Space Explorer II 工具（**Tools** > **Launch Design Space Explorer II**）支持针对资源，性能和功率目标进行最佳项目设置。 Design Space Explorer II（DSE II）结合设置和约束对设计进行处理，并报告关于设计的最佳设置。可利用 DSE II 的并行功能在多台计算机上进行编译。

如果设计已接近满足时间或面积要求，也可使用 DSE II 尝试不同 seed 以找出符合时序或面积要求的那个 seed。

图 1. Design Space Explorer II



可在设计过程的任何阶段运行 DSE II；但由于设计中的较大改动会抵消来自优化设置的获益，因此 Intel FPGA 建议在设计周期的后期才运行 DSE II。

相关链接

- Design Space Explorer II 工具
  Intel Quartus Prime Help 中
- 使用 Design Space Explorer
  21 分钟在线课程

1.5.1. DSE II 工作原理

DSE II 中，管理点（*exploration point*）为 Analysis & Synthesis，Fitter 和布局设置的集合，而一组 exploration point 称为一个设计管理（*design exploration*）。一个设计管理可包含各种适配器（fitter）seed。

DSE II 使用与每个管理点对应的设置编译设计。编译完成后，DSE II 对照您指定的优化目标评估性能数据。可指示 DSE II 针对时序，面积和功率进行优化。

相关链接

- Fitter Seed (第 84 页)
- Design Space Explorer II 用于电源驱动优化
  In *Intel Quartus Prime Pro Edition User Guide: Power Analysis and Optimization*

1.5.1.1. 使用计算资源

可配置 DSE II 以利用您的计算资源来运行设计探索。DSE II GUI 中，**Setup** 页面包含工作启动选项，而 **Status** 页面支持监督和控制工作。

DSE II 支持在本地计算机上编译或使用 LSF, SSH 或 Torque 在远程主机运行编译。对于 SSH，还可定义以逗号分隔的远程主机清单。

如果有笔记本电脑或标准计算机，就可使用单一编译功能在具有更高计算性能和内存容量的工作站上编译设计。

在运算中心上运行时，可在提交全部工作后指示 DSE II 安全退出，而编译保持运行直至完成。或者，编译完成后可收到电子邮件。

如果使用 SSH 启动作业，则远程主机必须使能公钥和私钥验证。对于采用短语加密的私钥，远程主机必须运行 ssh 密钥代理解密私钥，从而 `quartus_dse` 可执行文件能访问该密钥。

注意: Windows 远程主机需要 Cygwin 的 sshd 服务器和 PuTTY。

相关链接

- 建立页面（Design Space Explorer II）
  Intel Quartus Prime Help 中
- 状态页面（Design Space Explorer II）
  Intel Quartus Prime Help 中

1.5.1.2. 优化参数

DSE II 针对您需要优化的内容提供一系列预定义管理空间。此外，还可定义一套编译 seed。管理点的数量为管理 seed 数乘以管理模式的数量。

注意: 预定义空降的可用性取决于设计所针对的器件系列。

在 DSE GUI 中，请在 **Exploration** 页面指定这些设置。

相关链接

管理页面（Design Space Explorer II）
Intel Quartus Prime Help 中

1.5.1.3. 结果管理

DSE II 比较编译结果以确定关于设计的 Intel Quartus Prime 软件最佳设置。**Report** 页面显示结果摘要。

管理中，DSE II 从所有管理点的全部时序拐点中选择最差情况下的最佳 slack（时序余量）。如果要优化最差情况下的设置时序余量或保持时序余量，可在 Intel Quartus Prime 软件中指定时序约束。

磁盘空间

默认情况下，DSE II 保存所有编译数据。可限制编译完成后要保存的文件类型来节省磁盘空间。这些设置位于 **Exploration** 页面，**Results** 部分。

报告

DSE II 具有助于快速确定重要设计指标的报告工具，例如所有管理点中，较差情况的时序余量。

DSE II 针对其管理的全部点提供性能数据报告并将信息保存在工程目录下的 `project-name.dse.rpt` 文件中。DSE II 将管理点设置存档到 Intel Quartus Prime Archive Files（`.qar`）中。

相关链接

报告页面（Design Space Explorer II）
Intel Quartus Prime Help 中

1.5.2. 使用 DSE II Utility 执行设计管理

注意: 运行 DSE II 之前，指定设计的时序约束。

本说明涵盖运行设计管理时需定义的设计类型。有关 GUI 中所有可用选项的详细信息，请参阅 Intel Quartus Prime Help。

使用 DSE II 工具执行设计管理：

1. 启动 DSE II 工具。

   如果您的 Intel Quartus Prime 软件中打开的工程并启动了 DSE II，则会出现一个对话框询问是否关闭 Intel Quartus Prime 软件。请点击 **Yes**。
2. 在 **Project** 页，指定要管理的工程和版本。
3. **Setup** 页面中，指定执行本地管理或远程管理，以及设置作业启动。
4. 在 **Exploration** 页，指定优化设置和目标。
5. 配置完成后，点击 **Start**。

相关链接

- Design Space Explorer II 工具
  Intel Quartus Prime Help 中
- 使用 Design Space Explorer
  21 分钟在线课程

1.6. 设计优化概述修订历史

以下修订历史适用于本章：

文档版本	Intel Quartus Prime 版本	修订内容
2018.05.07	18.0.0	• 常规主题重组。 • 添加了 DSE II 工作原理，和执行设计管理时要遵循的主要步骤。
2017.11.06	17.1.0	• 在设计分析主题中添加了论及 Design Partition Planner 的内容。
2016.10.31	16.1.0	• 品牌更名为 Intel。
2016.05.03	16.0.0	删除了关于使用多个处理器时串行等效性的声明。
2015.11.02	15.1.0	将 *Quartus II* 更改为 *Quartus Prime*。
2014.12.15	14.1.0	• 将 Fitter Setting，Analysis & Synthesis 和 Physical Synthesis Optimizations Setting 的位置更新到 Compiler Setting。 • 更新了 DSE II 的内容。
2014 年 6 月	14.0.0	更新格式。
2013 年 11 月	13.1.0	印刷件少许改动。
2013 年 5 月	13.0.0	添加关于初始编译要求的信息。本节已从 Intel Quartus Prime 手册的 Area Optimization（区域优化）章节中删除。少许更新以描述时序和区域优化章节的划分。
2012 年 6 月	12.0.0	删除了反馈问卷链接。
2011 年 11 月	10.0.3	文档模板更新。
2010 年 12 月	10.0.2	更换为新的文档模板。文档内容未更改。
2010 年 8 月	10.0.1	更正链接
2010 年 7 月	10.0.0	首次发布。章节基于第二卷第 III 部分的主题和文字。

相关链接

文档存档
关于之前版本的 *Intel Quartus Prime 手册*，请搜索文档存档。

2. 优化设计网表

本章节说明如何使用 Intel Quartus Prime Netlist Viewer 分析和调试您的设计。

随着 FPGA 设计规模和复杂性不断增加，针对设计的分析，调试，优化和约束能力变得至关重要。利用当今的先进设计，多个设计工程师参与编码和合成不同设计块，从而难以分析和调试设计。而 Intel Quartus Prime RTL Viewer 和 Technology Map Viewer 是调试，优化和约束处理期间查看初始和完全映射综合结果的强效途经。

相关链接

- 使用 Netlist Viewers 的 Intel Quartus Prime 设计流程 (第 15 页)
- RTL Viewer 概述 (第 16 页)
- Technology Map Viewer 概述 (第 17 页)
- 原理图中进行过滤 (第 27 页)
- 查看时序路径 (第 34 页)

2.1. 何时使用 Netlist Viewer：分析设计问题

可使用 Netlist Viewer 分析和调试设计。通过以下简易示例演示如何使用 RTL Viewer 和 Technology Map Viewer 分析设计过程中遇到的问题。

使用 RTL Viewer 查看初始综合结果是个不错的方法，进而可确定是否已创建必要逻辑，以及软件是否已正确转换逻辑和连接。可使用 RTL Viewer 在仿真或其他验证处理前直观查看设计。在设计过程的早期阶段找到设计错误可节省您的宝贵时间。

如果验证期间发现意外行为，可使用 RTL Viewer 全面追踪网表并确保设计中的连接和逻辑符合预期。查看设计有助于找出并分析设计问题的根源。如果您的设计在 RTL Viewer 中看起来正确，则知道需着重分析设计过程的后期阶段并调查潜在时序违规或验证流程本身存在的问题。

可使用 Technology Map Viewer 查看 Analysis 和 Synthesis 结束时的结果。如果已在 Fitter 阶段编译了设计，则可在 Technology Map Viewer（Post-Mapping）中查看“映射后”（post-mapping） 网表，适配后的网表在 Technology Map Viewer 中查看。如果仅执行 Analysis 和 Synthesis，则两个 Netlist Viewer 显示相同“映射后”（post-mapping）网表。

此外，可使用 RTL Viewer 或 Technology Map Viewer 查找特定信号来源以助于调试您的设计。使用本章中介绍的导航技术可对设计中的内容进行全面搜索。可从您的兴趣点追溯到信号源但前提是确保连接符合预期要求。

Technology Map Viewer 有助于查找网表中的合成后节点，并在优化期间进行约束。该功能对设计中两个寄存器间的多周期时序约束十分有利。从某个 I/O 端口开始，在设计中向前后或向后追踪以及通过层次结构层找到兴趣节点或直观检查原理图找到特定寄存器。

贯穿整个 FPGA 设计，调试和优化阶段，可通过多种方式使用所有网表查看器以提高分析设计时的工作效率。

相关链接

- 使用 Netlist Viewers 的 Intel Quartus Prime 设计流程 (第 15 页)
- RTL Viewer 概述 (第 16 页)
- Technology Map Viewer 概述 (第 17 页)

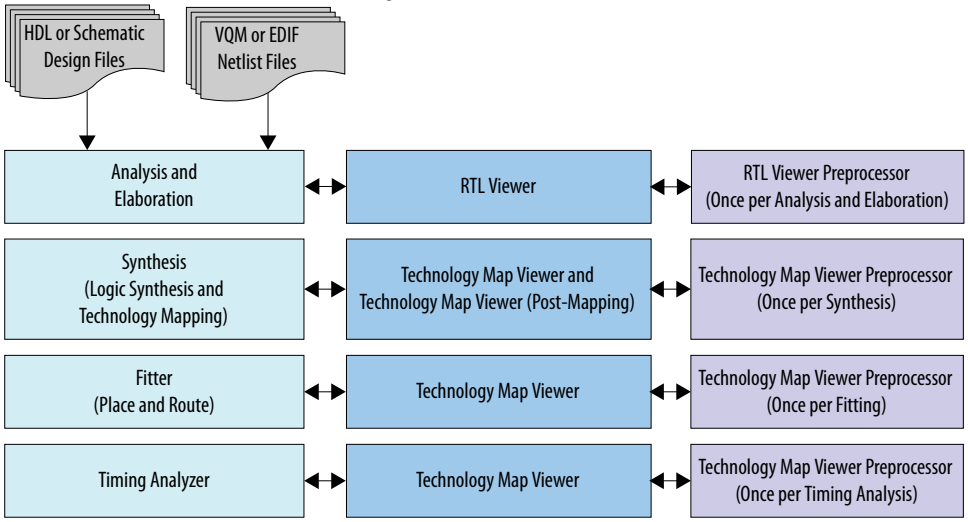
2.2. 使用 Netlist Viewers 的 Intel Quartus Prime 设计流程

编译设计后首次打开其中一个 Netlist Viewer 时，会先自动运行预处理器，然后开启 Netlist Viewer。

若单击预处理器进程框中的链接转到 **Settings** > **Compilation Process Settings** 页，在此开启 **Run Netlist Viewers preprocessing during compilation** 选项。如果开启该选项，则预处理将成为完整工程编译流程的一部分，这样即使 Netlist Viewer 立即打开而不显示预处理对话框。

图 2. Intel Quartus Prime 设计流程，包含 RTL Viewer 和 Technology Map Viewer

该图示显示 Netlist Viewer 如何适配到基础 Intel Quartus Prime 设计流程中。



Netlist Viewer 可以运行预处理器阶段之前，必须先编译您的设计：

- 要打开 RTL Viewer，首先执行 Analysis 和 Elaboration。
- 要打开 Technology Map Viewer (Post-Fitting)或 Technology Map Viewer (Post-Mapping)，应首先执行 Analysis 和 Synthesis。

Netlist Viewer 显示最近一次成功编译的结果。

- 因此，如果因更改设计而导致 Analysis 和 Elaboration 期间出现错误，则无法查看该新设计文件的网表，但仍可查看最近成功编译的设计文件版本的结果。
- 如果编译期间出现错误，且与工程相应的编译阶段尚未成功运行，则 Netlist Viewer 无法显示；该情况下尝试打开 Netlist Viewer 时， Intel Quartus Prime 软件将发送一条错误消息。

注意: 开始新的编译时，如果有开启的 Netlist Viewer，则该 Netlist Viewer 会自动关闭。仅可在编译成功完成后才能重新打开 Netlist Viewer 查看新的设计网表。

2.3. RTL Viewer 概述

RTL Viewer 支持查看 Intel Quartus Prime 软件中 Intel Quartus Prime Pro Edition synthesis 结果或第三方网表文件的寄存器传输级（RTL）图形表示。

可查看对设计执行 Analysis 和 Elaboration 后的结果，只要该设计使用支持的 Intel Quartus Prime 设计实体方式，具体包括 Verilog HDL Design Files (`.v`)，SystemVerilog Design Files (`.sv`)，VHDL Design Files (`.vhd`)，AHDL Text Design Files (`.tdf`)或原理图 Block Design Files (`.bdf`)。

还可查看通过综合工具生成 Verilog Quartus Mapping File (`.vqm`)或 Electronic Design Interchange Format (`.edf`)文件的设计的原子原语层次（例如器件逻辑单元和 I/O 端口）。

RTL Viewer 会在技术映射和综合或适配器优化以前，显示执行 Analysis 和 Elaboration 后，或 Intel Quartus Prime 软件执行网表提取后的设计网表原理图。该视图为初步预优化设计结构图，接近原始源设计。

- 对于通过 the Intel Quartus Prime Pro Edition synthesis 进行综合的设计，该视图显示 Intel Quartus Prime 软件如何转换设计文件。
- 对于利用第三方综合工具进行综合的设计，该视图显示由综合工具生成的网表。

运行 Intel Quartus Prime 工程的 RTL Viewer，首先需分析设计生成 RTL 网表。要分析设计并生成 RTL 网表，请单击 **Processing** > **Start** > **Start Analysis & Elaboration**。还可在任何包含 Intel Quartus Prime 编译流程初始 Analysis 和 Elaboration 阶段的处理中执行完整编译。

单击 **Tools** > **Netlist Viewers** > **RTL Viewer** 打开 RTL Viewer。

相关链接

Netlist Viewer 用户接口 (第 17 页)

2.3.1. RTL Viewer 中可读性最大化

显示设计时，RTL Viewer 优化网表以最大化可读性：

- 从显式中删除无扇出（未连接的输出）和扇入（无连接的输入）的逻辑。
- 隐藏默认连接，例如 V_{CC} 和 GND。
- 将管脚，网络，线缆，模块端口和具体逻辑适当分组到总线中。
- 分组恒定总线连接。
- 数值以十六进制格式显示。
- 将原理图中的非门（NOT gate）转换为气泡取反（非）符。
- 将等效组合门控链合并为单个门控；例如，一个馈送 2-input 与门（AND gate）的 2-input 与门（AND gate）转换为单个 3-input 与门（AND gate）。

2.3.2. 运行 RTL Viewer

对 Intel Quartus Prime 工程运行 RTL Viewer：

1. 点击 **Processing** > **Start** > **Start Analysis & Elaboration** 分析设计以生成 RTL 网表。

   还可在任何包含 Intel Quartus Prime 编译流程初始 Analysis 和 Elaboration 阶段的处理中执行完整编译。
2. 单击 **Tools** > **Netlist Viewers** > **RTL Viewer** 打开 RTL Viewer。

2.4. Technology Map Viewer 概述

Intel Quartus Prime Technology Map Viewer 提供 Analysis 和 Synthesis 后，或 Fitter 将设计映射到目标器件后， FPGA 设计的技术特定图形表示。

Technology Map Viewer 显示设计中原子原语的层次（如，器件逻辑单元和 I/O 端口）。对于支持的器件系列，还可查看内部寄存器和逻辑单元（LCELL）中的查找列表（LUT），以及 I/O 原子原语中的寄存器。

可能的情况下， Intel Quartus Prime 软件在整个综合过程中保留每层次中的端口名称。但软件可能会更改或删除设计中的端口名称。例如，软件在合成期间删除未连接或由 GND 或 V_{CC} 驱动的端口。如果某端口名称改变，则软件会在设计中为其约束一个相关用户逻辑名或常规端口名，如 `IN1` or `OUT1`。

可查看综合，拟合或时序分析后的 Intel Quartus Prime 技术映射结果。要运行 Intel Quartus Prime 工程的 Technology Map Viewer，在 **Processing** 菜单上，指向 **Start** 并单击 **Start Analysis & Synthesis** 将设计综合并映射到目标技术。在此阶段，Technology Map Viewer 显示的“映射后”网表与 Technology Map Viewer (Post-Mapping)相同。您还可执行完整编译，或任何包含编译流程综合阶段的处理。

对于已完成 Fitter 阶段的设计，Technology Map Viewer 会显示 Fitter 如何通过物理性综合优化更改网表，而 Technology Map Viewer (Post-Mapping)显示“映射后”（post-mapping ）网表。如果已完成 Timing Analysis 阶段，就可从 Technology Map Viewer 的 Timing Analyzer 报告中找到时序路径。

要开启 Technology Map Viewer，可单击 **Tools** > **Netlist Viewers** > **Technology Map Viewer (Post-Fitting)**或 **Technology Map Viewer (Post Mapping)**。

相关链接

- 查看时序路径 (第 34 页)
- 在 Schematic View 中查看节点内容 (第 28 页)
- Netlist Viewer 用户接口 (第 17 页)

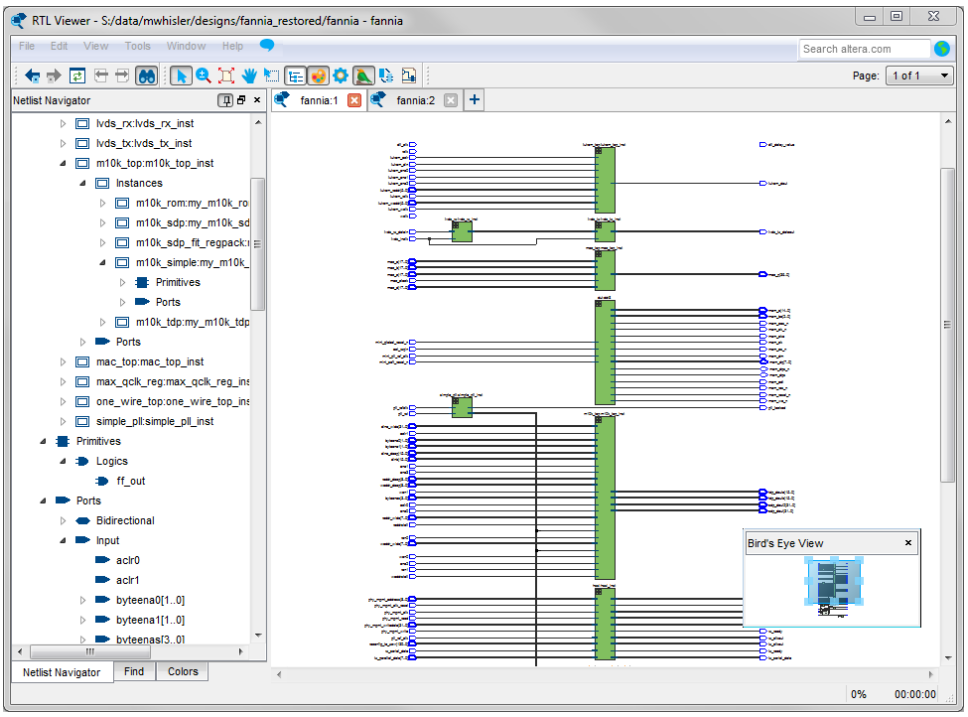
2.5. Netlist Viewer 用户接口

Netlist Viewer 是查看和操作网表中节点和网络的图形用户界面。

RTL Viewer 和 Technology Map Viewer 各由以下主要部分组成：

- **Netlist Navigator** 窗格—显示工程层次。
- **Find** 窗格—支持在原理视图中查找和定位指定设计元件。
- **Properties** 窗格—从快捷菜单选择 **Properties** 后显示所选模块的属性。
- 原理图视图—显示设计内部结构的图形表达。

图 3. RTL Viewer

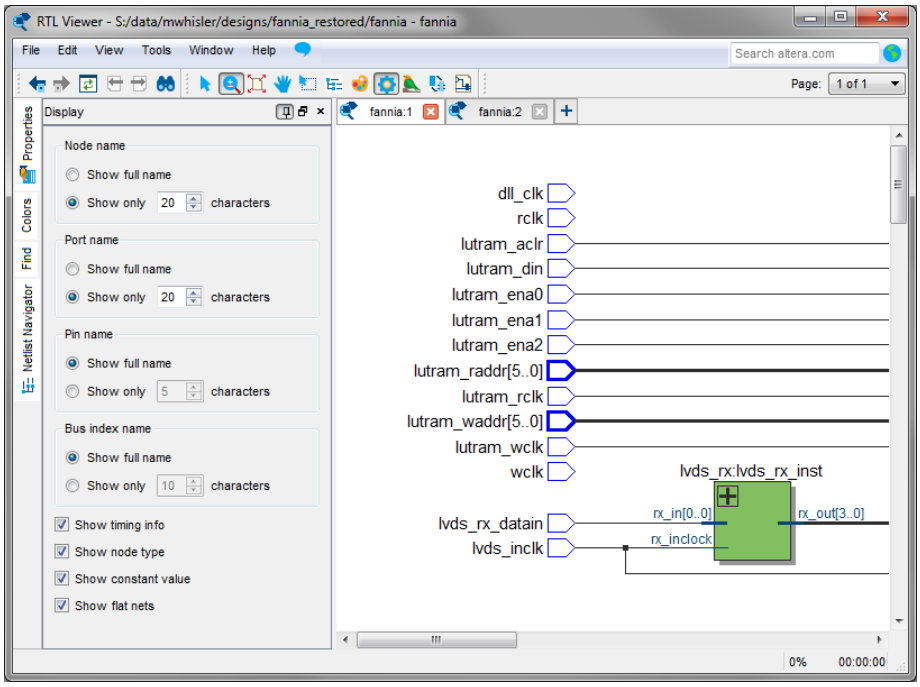


Netlist Viewer 还包含一个为原理图视图提供使用工具的工具栏。

- 使用 **Back** 和 **Forward** 按钮在原理图视图间进行切换。仅在返回时未对视图进行任何更改才可继续前进。这些命令不会撤销操作，例如选择节点。Netlist Viewer 最多存储 10 个操作，包括过滤，层次导航，网表导航和缩放操作。
- **Refresh** 按钮恢复原理图视图并优化布局。如果更改设计并重新编译，而 **Refresh** 不会重新加载数据库。
- 单击 **Find** 钮打开和关闭 **Find** 窗格。
- 单击 **Selection Tool** 和 **Zoom Tool** 钮在选择模式和缩放模式间切换。
- 单击 **Fit in Page** 钮重置原理图视图以包含整个设计。
- 使用 **Hand Tool** 更换查看焦点且不改变视角。
- 单击 **Area Selection Tool** 在某区域内的端口，管脚和节点周围拖动选择框。
- 单击 **Netlist Navigator** 钮打开或关闭 **Netlist Navigator** 窗格。
- 单击 **Color Settings** 钮打开 **Colors** 窗格，在此自定义 Netlist Viewer 颜色。

- 单击 **Display Settings** 钮，打开 **Display** 窗格在此指定以下设置：
  - **Show full name** 或 **Show only <*n*> characters**。可单独为 **Node name**，**Port name**，**Pin name** 或 **Bus name** 指定该项。
  - 打开或关闭 **Show timing info**。
  - 打开或关闭 **Show node type**。
  - 打开或关闭 **Show constant value**。
  - 打开或关闭 **Show flat nets**。

图 4. Display（显示）设置



- **Bird's Eye View** 按钮打开 **Bird's Eye View** 窗口显示设计的微缩版，可在设计内进行浏览且快速调整原理图视图中的缩放率。
- **Show/Hide Instance Pins** 钮交替显示因某些功能而未显示的实例管脚，例如 Netlist Viewer 和 Timing Analyzer 间的交互探测（cross-probing）。也可在大量未连接或为使用管脚中过滤节点结果时使用该按钮隐藏未连接的实例管脚。Netlist Viewer 默认隐藏实例管脚。
- 如果 Netlist Viewer 显示涵盖多个页面，则 **Show Netlist on One Page** 钮会将网表视图调整为单个页面。此操作可使网表追踪更容易。

仅可有一个 RTL Viewer，一个 Technology Map Viewer (Post-Fitting)和一个 Technology Map Viewer (Post-Mapping)同时打开，即使每个窗口可显示多个页面，每个页面都有多个选项卡。例如，无法同时打开两个 RTL Viewer 窗口。

相关链接

- RTL Viewer 概述 (第 16 页)
- Technology Map Viewer 概述 (第 17 页)

- Netlist Navigator 窗格 (第 20 页)
- Netlist Viewer 查找窗格 (第 22 页)
- Properties 窗格 (第 20 页)

2.5.1. Netlist Navigator 窗格

Netlist Navigator（网表导航）窗格根据设计的层次级别以树形格式显示整个网表。每个级别将类似单元分组为子类别。

Netlist Navigator 窗格支持遍历设计层次查看每个级别的逻辑原理图。还可在 **Netlist Navigator** 中选择要在原理图视图中突出显示的元素。

注意: **Netlist Navigator** 窗格中不罗列原子原语内的节点。

对于设计层次中的每个模块，**Netlist Navigator** 窗格通过以下表格显示可应用的单元。单击“**+**”图标展开单元。

表 3. Netlist Navigator 窗格单元

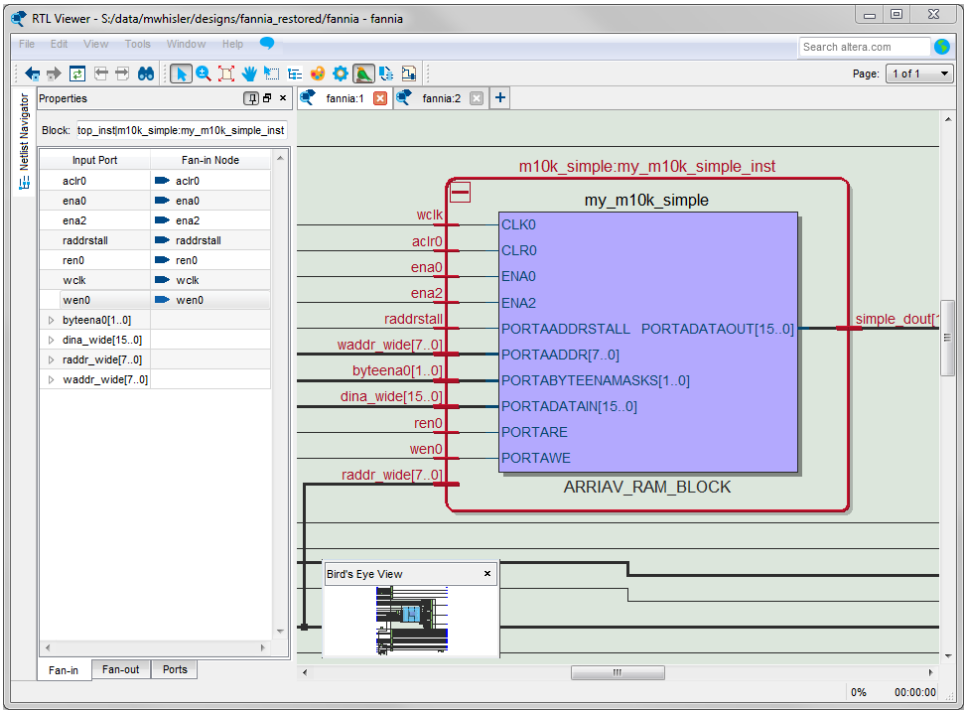
单元	说明
Instances	设计中的模块或实例，展开后可查看低层次级别。
Primitives	低级别节点，无法向更低层次级别扩展。这些原语包括： • 使用 Intel Quartus Prime Pro Edition 综合时，可在 RTL Viewer 中查看的寄存器和门控。 • 使用第三方综合软件中的 VQM 或 EDIF 时，Technology Map Viewer 或 RTL Viewer 中的逻辑单元原子。 在 Technology Map Viewer 中，可查看具体原子原语的内部实现，但不能遍历层次结构的较低层。
Ports	层次结构中当前级别的 I/O 端口。 • 查看顶层层次级别时，管脚为器件 I/O 管脚，查看底层级别时，管脚是设计的 I/O 端口。 • 当管脚代表总线或管脚阵列时，展开列表视图中的管脚条目查看各个管脚名称。

2.5.2. Properties 窗格

可通过 **Properties** 窗格查看实例或原语属性。

图 5. Properties 窗格

要在 RTL Viewer 或 Technology Map Viewer 中查看实例或原语属性，右键单击节点并单击 **Properties**.



Properties 窗格的选项卡中包含所选节点的信息，如下：

- **Fan-in** 选项卡显示 **Input port** 和 **Fan-in Node**。
- **Fan-out** 选项卡显示 **Output port** 和 **Fan-out Node**。
- **Parameters** 选项卡显示实例中的 **Parameter Name** 和 **Values**。
- **Ports** 选项卡显示 **Port Name** 和 **Constant**（例如，V_{CC} 或 GND）。以下表格罗列了端口可能有的值：

表 4. 可能出现的端口值

值	说明
V_{CC}	端口未连接且具有 V_{CC} 值（连接到 V_{CC}）
GND	端口未连接且具有 GND 值（连接到 GND）
--	端口已连接时出现的值（V_{CC} 或 GND 以外的情况）
Unconnected	端口未连接且无其他值（悬空）

如果所选节点为原子原语，则 **Properties** 窗格显示内部逻辑的原理图。

2.5.3. Netlist Viewer 查找窗格

可通过在 **Find** 窗格中设置以下选项缩小搜索过程的范围：

- 在 **Find** 窗格中单击 **Browse** 以指定搜索层次级别。在 **Select Hierarchy Level** 对话框中，选择要搜索的特定实例。
- 打开 **Include subentities** 选项以在搜索中包含父级实例的子层次。
- 单击 **Options** 打开 **Find Options** 对话框。打开 **Instances**，**Nodes**，**Ports** 或三者的任意组合，以进一步细化搜索参数。

单击 **List** 按钮时，**Find** 对话框下方出现一个进度条。

所有与您设置的条件匹配的结果都罗列在表格中。双击列表中的项目时，相关节点在原理图视图着红色突显。

2.6. 原理图视图

原理图视图显示于 RTL Viewer 和 Technology Map Viewer 的右侧。原理图视图包含表示网表中设计逻辑的原理图。该视图是 RTL Viewer 中查看门级网表和 Technology Map Viewer 中查看技术映射网表的主屏。

RTL Viewer 和 Technology Map Viewer 默认尝试以单页视图显示原理图。如果原理图跨多个页面，则可突出显示某个网络并且使用连接器追踪单个页面中的信号。

2.6.1. 以多选项卡视图显示原理图

RTL Viewer 和 Technology Map Viewer 支持多选项卡式视图。

通过多选项卡式视图，原理图可在不同选型卡中显示。个选项卡式视图中的选择相互独立，但选项卡中的选择集中同步于 Netlist Navigator 窗格。

要创建一个新的选项卡，在选项卡行的末尾单击 **New Tab** 按钮。此时就可从 **Netlist Navigator** 窗格中将节点拖动到原理图视图中。

在选项卡中右键单击查看快捷键菜单执行如下操作：

- 使用 **New Tab** 创建空白视图
- 为目标选项卡创建 **Duplicate Tab**
- 选择 **Cascade Tabs**
- 选择 **Tile Tabs**
- 选择 **Close Tab** 关闭目标选项卡
- 选择 **Close Other Tabs** 关闭目标选项卡之外的所有选项卡。

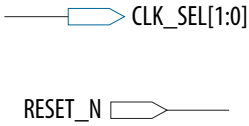
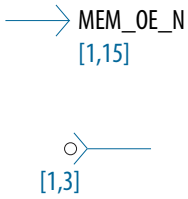
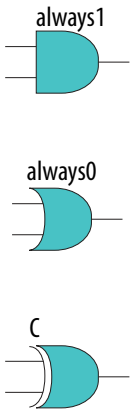
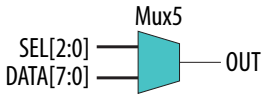
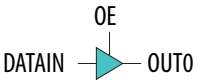
2.6.2. 原理图符号

原理图中节点的符号表示您设计网表中的单元。这些单元包括输入和输出端口，寄存器，逻辑门控，Intel 原语，高级别操作符和层次实例。

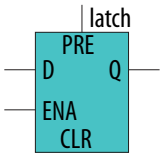
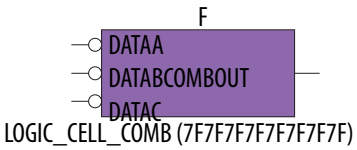
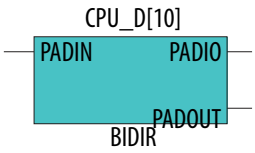
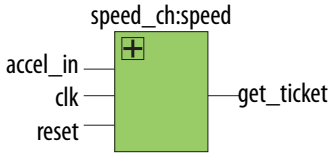
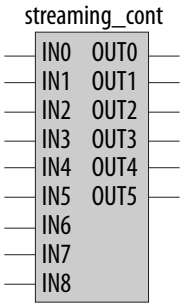
注意: 逻辑门控和操作符原语仅在 RTL Viewer 中出现。Technology Map Viewer 中的逻辑以原子原语呈现，例如寄存器和 LCELL。

表 5. 原理图中的符号

本列表罗列并说明可在 RTL Viewer 和 Technology Map Viewer 的原理图中显示的原语和基本符号。

符号	说明
I/O 端口 	层次结构中当前级别的输入，输出或双向端口。查看顶层层次时，器件输入，输出或双向管脚。该符号还可代表总线。仅显示一条线路连接到双向符号，表示输入和输出路径。 输入符出现在原理图的最左侧。输出和双向符出现在原理图的最右侧。
I/O 连接器 	输入或输出连接器，表示来自相同层次中另一页面的网络。要转到包含源或目的地的页面，请双击连接器跳转到相应页面。
OR, AND, XOR 门 	OR（或），AND（与）或 XOR（非）门原语（端口数目可不相同）。输入或输出端口上的小圆圈（气泡符）表示端口反转。
MULTIPLEXER 	具有段选择子的多路复用器原语在端口 `0` 和端口 `1` 之间选择。具有两个以上输入的多路复用器显示为操作符。
BUFFER 	缓冲器原语。本图示显示三态缓冲器，和一个已反转的输出使能端口。其他不具备使能端口的缓冲器包括，LCELL，SOFT，和 GLOBAL。NOT 门和 EXP 扩展器缓冲器使用该符号，不具备使能的端口和已反转输出端口。

继续...

符号	说明
LATCH 	锁存器/DFF（数据触发器原语）。DFF 的端口与锁存器和时钟触发器相同。其他触发器原语与之类似： • DFFEA（具有使能和异步加载的数据触发器）原语和其他 ALOAD 异步加载和 ADATA 数据信号 • DFFEAS（具有使能和同步以及异步加载的数据触发器），其还有 ASDATA 作为下游总线数据端口
原子原语 	原子原语。该符号显示原子名称，端口名称和原子类型。蓝色阴影表示可查看内部详细信息的原子原语。
其他原语 	任何未归入前述类别的原语。该原语是不可扩展为更低层次的低级别节点。此符号显示端口名称，原语或操作符类型及其名称。
实例 	设计中无对应原语或操作符的实例（用户定义的层次块）。该符号显示端口名称和实例名称。
加密实例 	设计中用户定义的加密实例。该符号显示实例名称。无法打开较低级别层次的原理图，因为源设计已加密。
RAM	同步存储器实例具有已寄存输入和可选已寄存输出。该符号显示器件系列和存储器块类型。该图示显示为 Stratix M-RAM 块中真正的双端口存储器块。

继续...

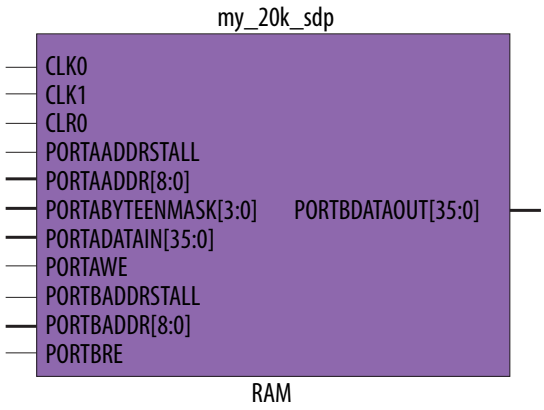
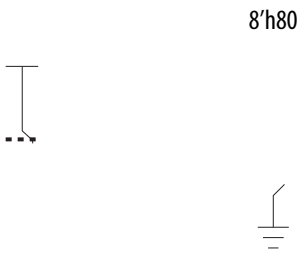
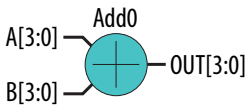
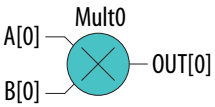
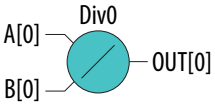
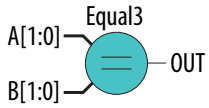
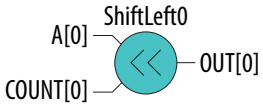
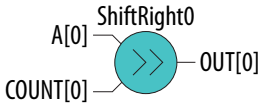
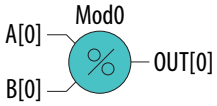
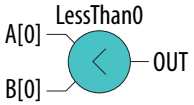
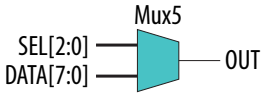
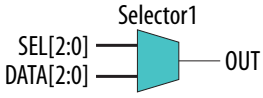
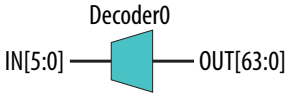
符号	说明
	
常量 	常量信号值以灰色突出显示，默认情况下整个原理图中以十六进制格式显示。

表 6. RTL Viewer Schematic View 中的操作符号

下表罗列并说明 RTL Viewer 原理图中其他更高级别操作符号。

符号	说明
	加法器操作符： OUT = A + B
	乘法器操作符： OUT = A ¥ B
	除法器操作符： OUT = A / B

继续...

符号	说明
	等式
	左移操作符： `OUT = (A << COUNT)`
	右移操作符： `OUT = (A >> COUNT)`
	右移操作符： `OUT = (A%B)`
	小于比较器： `OUT = (A<:B:A>B)`
	多路复用器： `OUT = DATA [SEL]` 数据范围大小为 $2^{\text{sel range size}}$
	选择器： 多路复用器带有一个热选输入和两个以上输入信号
	二进制数解码器： `OUT = (binary_number (IN) == x)` for x = 0 到 $x = 2^{(n+1)} - 1$

相关链接

- 原理图分页 (第 31 页)

- 关注原理图页面中的网络 (第 31 页)

2.6.3. 在 Schematic View 中选择项目

要在原理图视图中选择一个项目，请确保 Netlist Viewer 功能条中的 **Selection Tool** 已使能。在原理图中单击一个项目会以红色突出显示。

使用鼠标选择时，可按 Shift 健选择多个项目。

Netlist Navigator 窗格中会自动选择在原理图中已选择的项目。如要求显示所选择项，则文件夹自动展开；但取消选择条目后，文件夹不会自动收起。

在原理图中选择层次框，节点或端口时，Schematic View 以红色突出显示该项目，但不突出显示连接网络。在原理图中选择网络（线缆或总线）时，Schematic View 以红色突出显示已连接网络。

一旦选择某项目后，就可根据右键选择时出现的快捷菜单上的内容对其执行不同操作。

相关链接

Netlist Navigator 窗格 (第 20 页)

2.6.4. Schematic View 中的快捷菜单命令

右键单击原理图中所选实例或原语时，Netlist Viewer 显示快捷菜单。

如果已选项目是一个节点，则会看到如下选项：

- 单击 **Expand to Upper Hierarchy** 显示目标节点的父级层次。
- 单击 **Copy ToolTip** 复制所选项目的名称到剪贴板。此命令不适用于网络。
- 单击 **Hide Selection** 从原理图中删除所选项目。该命令不会将项目从设计中删除，仅会在当前视图中将其屏蔽。
- 单击 **Filtering** 显示子菜单和带有过滤选择的选项。

2.6.5. 原理图中进行过滤

通过过滤可将节点和网络从网表中过滤，从而仅查看您感兴趣的逻辑单元。

可根据您希望查看的路径部分，通过选择层次框，节点，或节点端口中的状态过滤网表。以下为可使用的过滤命令：

- **Sources**—显示选择的来源。
- **Destinations**—显示选择的目的地。
- **Sources & Destinations**—显示选择的来源和目的地。
- **Selected Nodes**—仅显示已选择节点。

- **Between Selected Nodes**—显示介于所选节点间路径中的节点和连接。
- **Bus Index**—显示输入或输出总线端口一个或多个指数的源和目标。
- **Filtering Options**—显示 **Filtering Options** 对话框：
  - **Stop filtering at register**—打开该选项指示 Netlist Viewer 过滤到最近的寄存器边界。
  - **Filter across hierarchies**—打开该选项指示 Netlist Viewer 跨层次过滤。
  - **Maximum number of hierarchy levels**—设置原理图最多可显示的层次级别。

要过滤网表，选择一个层次框，节点，端口，网络或状态节点，在窗口中单击右键，指向 **Filter** 并单击相应的过滤命令。Netlist Viewer 生成一个新页面显示网表经过滤后剩下的内容。

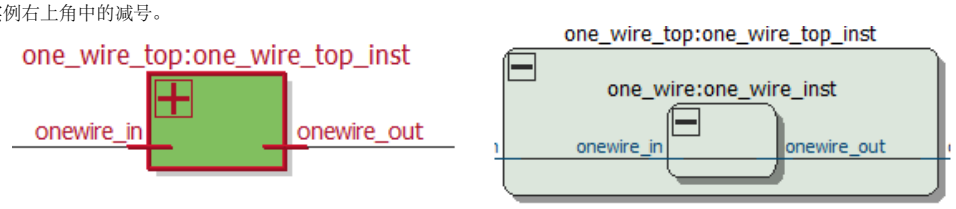
2.6.6. 在 Schematic View 中查看节点内容

在 RTL Viewer 和 Technology Map Viewer 中，可查看节点的内容以了解实际实现详情。

可查看 LUT，寄存器和逻辑门控。还可在 RTL Viewer 或 Technology Map Viewer 中查看具体器件中 RAM 和 DSP 块的实现。在 Technology Map Viewer 中，可查看原语的内容以了解其实际实现详情。

图 6. **Wrapping 和 Unwrapping 对象**

如果可打开（unwrap）实例内容，则原理图中目标对象的右上角出现一个加号。要收起内容（以及恢复压缩模式），单击已打开实例右上角中的减号。



注意: 原理图视图中，原子实例中的内部信息不可作为单个节点进行选择。任何内部细节上的任何鼠标操作都视作在原子实例上的鼠标操作。

图 7. **与层次外连接的节点**

某些情况下，所选实例与原理图中层次可视级别以外的某些内容相连接。该情况下，网络显示为虚线。双击虚线展开视图显示连接目标。

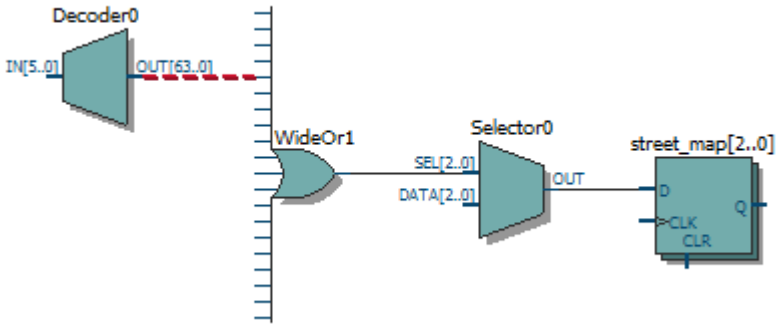


图 8. 显示层次中的网络

在网络连接到层次外实例的情况下。可选择网络并展开节点查看目标端口。

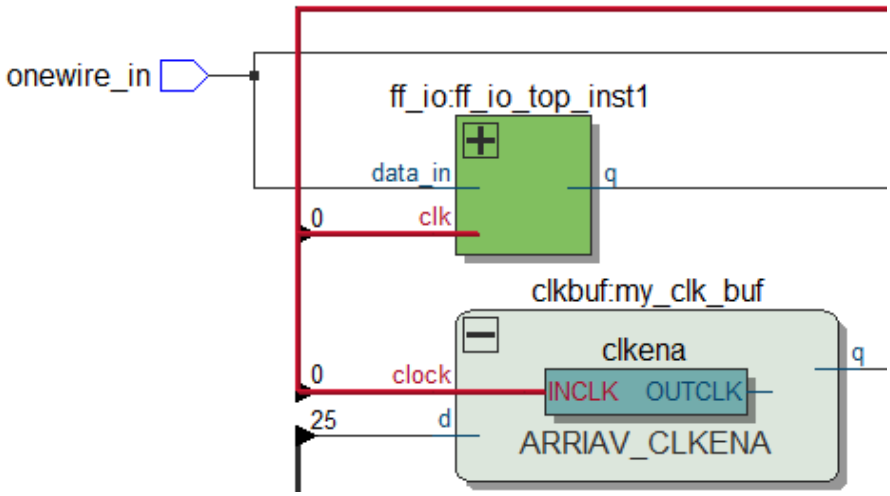
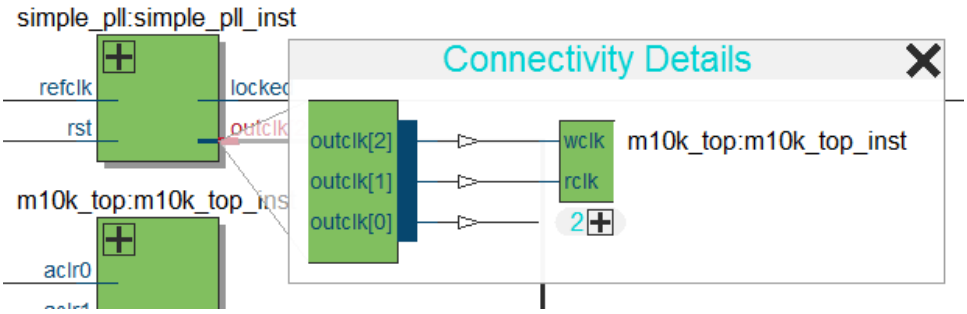


图 9. 显示连接性详细信息

可选择一个总线端口或总线管脚并单击工程相应菜单中的 **Connectivity Details**。



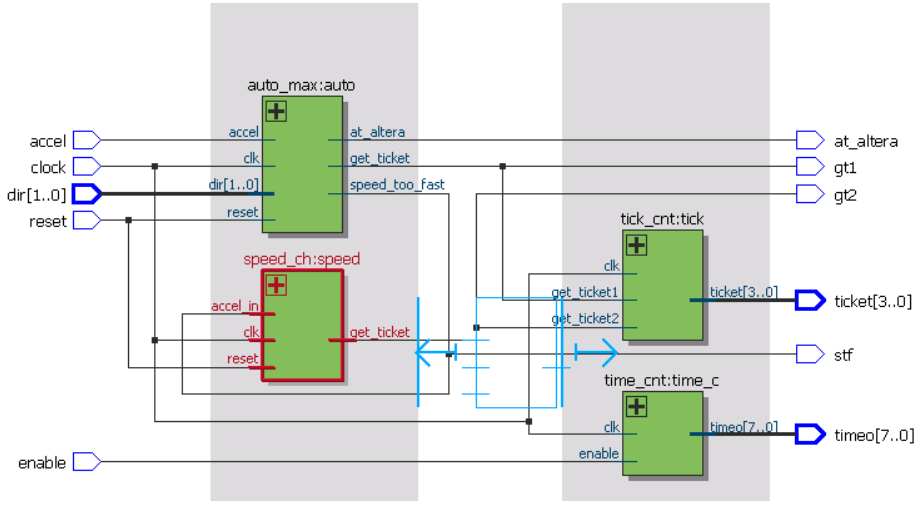
可双击 **Connectivity Details** 窗口中的目标快速导航到对应内容。如果出现加号，则可在视图中进一步展开目标对象。该功能有助于在复杂网表中追踪信号。

2.6.7. 在 Schematic View 中移动节点

通过将项目拖动到目的地重新排列原理图。

将节点从网表中的一个区域移动到另一区域，选择节点并按住 Shift 键。层次中的合法布局以阴影区域显示。单击以拖动所选择节点。

图 10. 移动节点时的合法布局



要将原理图恢复到其默认布局，右键单击并点选 **Refresh**。

2.6.8. 在 Technology Map Viewer 中查看 LUT 表达

右键点击所选 LUT 并点击 **Properties** 可查看 LUT 的不同表示。

可在 **Properties** 对话框的以下三个选项卡中查看 LUT 表示：

- **Schematic** 选项卡—表示 LUT 的等效门控。
- **Truth Table** 选项卡—表示真值表。

相关链接

Properties 窗格 (第 20 页)

2.6.9. 缩放控制

使用工具栏中的 Zoom Tool，或鼠标手势在 View 菜单上控制原理图放大率。

默认情况下，Netlist Viewer 显示最符合窗口大小的页面。如果原理图页面过大，则该原理图以最小缩放水平显示，且视图以第一个节点为中心。单击 **Zoom In** 以较大尺寸查看图像；单击 **Zoom Out** 以较小尺寸查看图像（未显示完整图像时）。**Zoom** 命令允许您指定放大百分比（100%被视为原理图符号的正常大小）。

可使用 Netlist Viewer 工具栏上的 Zoom Tool 控制原理图中的放大率。在工具栏中选择 Zoom Tool，单击原理图会放大并以您点击的位置为中心放置视图。右键单击原理图会缩小并以点击处为中心放置视图。选择 Zoom Tool 后，还可通过鼠标光标选择矩形框区放大特定部分。放大原理图以显示所选区域。

在原理图视图中，还可使用鼠标手势放大特定部分：

- **zoom in**—从左上角开始围绕区域拖动一个框，然后向右下方拖动放大该区域。
- **Zoom -0.5**—从左下方向右上方沿直线拖动缩小 0.5 级放大率。
- **zoom 0.5**—从右下方往左上方直线拖动放大 0.5 级放大率。
- **zoom fit**—从右上方向左下方直线拖动配合页面调整原理图视图。

相关链接

原理图中进行过滤 (第 27 页)

2.6.10. Bird's Eye View 导览

要打开 Bird’s Eye View， 请在 View 菜单，单击 **Bird’s Eye View**，或在工具栏中单击 **Bird’s Eye View** 图标。

查看整个原理图在调试和追踪大型网表时非常有用。 Intel Quartus Prime 软件支持使用鸟瞰图功能快速导览原理图中的指定部分，该功能适用于 RTL Viewer 和 Technology Map Viewer。

Bird’s Eye View 显示当前感兴趣的区域：

- 点击并拖动光标在所选区域周围形成一个矩形框来选择该区域。
- 点击并拖动矩形框在原理图中移动。
- 调整矩形框大小在原理图中进行缩放。

2.6.11. 原理图分页

对于较大设计的层次，RTL Viewer 和 Technology Map Viewer 可将您的网表分成多个原理图页面。

层次级别被分成多个页面后，原理图窗口的标题栏标示当前显示页页码以及该层次级别的总页数。原理图视图显示为 **Page**<*当前页码*> **of** <*总页数*>。

相关链接

Netlist Viewer 用户接口 (第 17 页)

2.6.12. 关注原理图页面中的网络

输入和输出连接器符号表示相同层次页面中连接的节点。双击连接器通过网络追踪到层次结构的下一页。

注意: 在双击并关注连接器端口后，Netlist Viewer 打开一个新页面，而该页面使用的缩放系数与上一页相同，且以指定源或目标网络为中心呈现。要追踪层次中新页面的指定网络，Intel 建议首先选择必要网络，将其以红色突出显示，然后才双击在页面中导览。

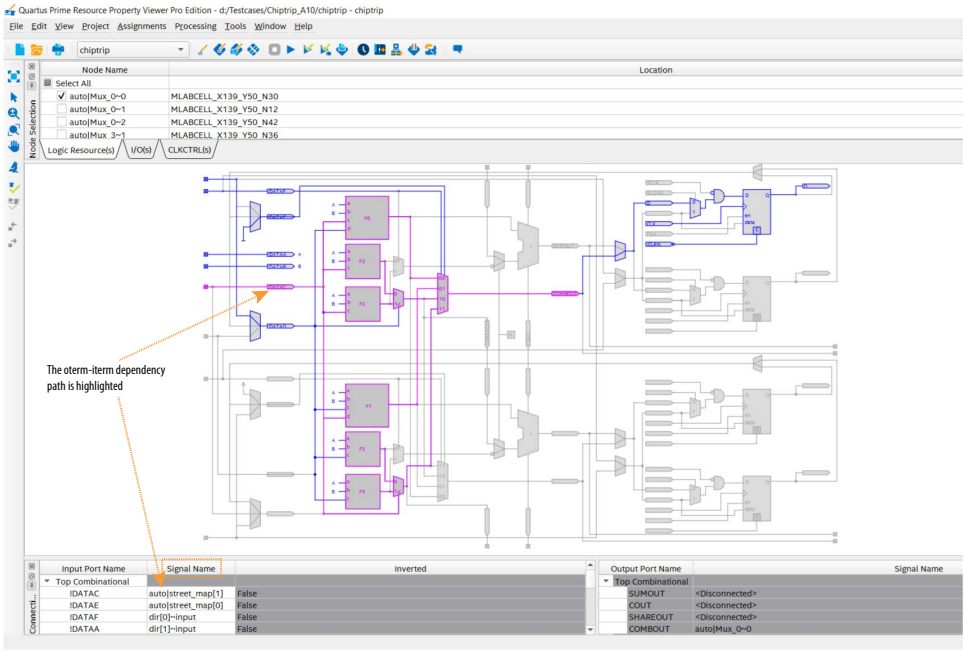
相关链接

原理图符号 (第 22 页)

2.6.13. 维护 Resource Property Viewer 中的选择

19.1 发布以前，当鼠标悬停于 Resource Property Viewer 中的端口上时，可查看节点的 iterm-oterm 从属路径，且从属路径被突出显示。然而一旦鼠标移动到其他节点时，就失去突出显示。

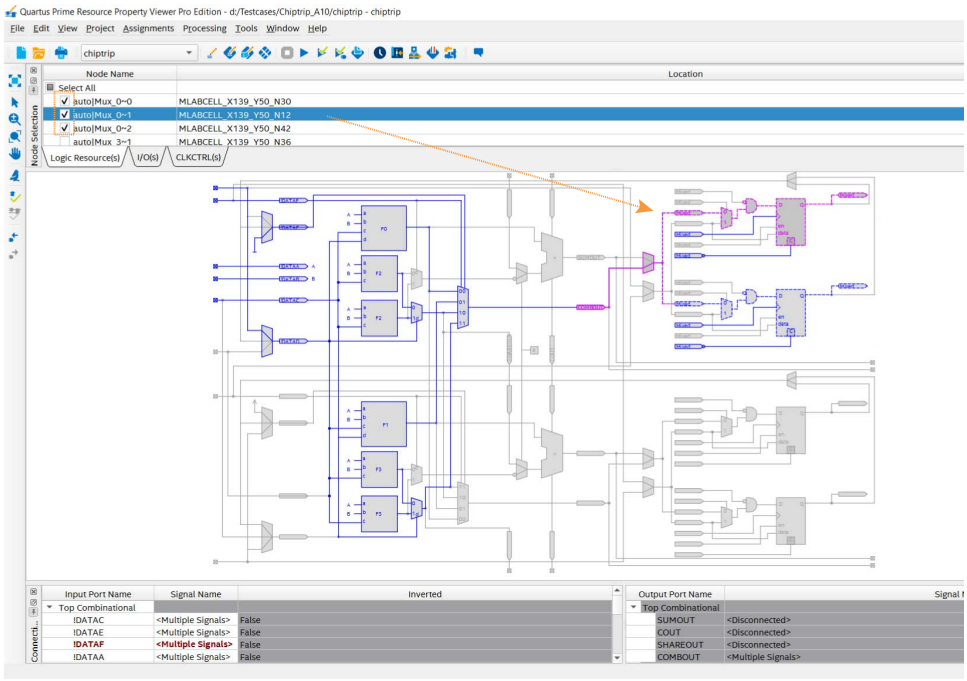
自 19.1 发布起，在原理图中单击端口或在 Connectivity 窗格中点击信号名称，将选择该路径。每个节点仅保留一个从属路径。重新编译，关闭此工程，选择另一端口，运行新定位，或在非端口处左键单击，将清除该从属路径。



注意:

- 在 Chip Planner 中双节节点，打开 Resource Property Viewer 都被视为新定位。
- 为要路径追踪，可使用 **Go To Destination Node** 或 **Go To Source Node** 选项对 Node Selection 列表添加更多节点，将不会清除从属路径选择（与每个新定位相对，例如，在 Chip Planner 中双击节点）。
- 从属路径选择或突出显示不适用于 Chip Planner 中的 Node Properties。

为多选对象选择端口和信号名称时，仅在端口被使用后才更新从属路径。如果未使用该端口，则从属路径被清除。然而，Resource Property Viewer 不显示多节点选择的从属路径选择。其仅以单个路径追踪为服务目标（仅有一个颜色选择）。



2.7. 交叉探查 Source Design File 和其他 Intel Quartus Prime Windows

Intel Quartus Prime 软件中，RTL Viewer 和 Technology Map Viewer 支持对源设计文件和多种其他窗口进行交叉探查。

可在 Netlist Viewer 中选择一个或多个感兴趣的层次框，节点，状态模式或状态转变弧线，并在其他可用的 Intel Quartus Prime 软件窗口中找到相应项。随后可查看并在对应编辑器或平面布局规划中进行更改或约束。

要在其他窗口中找到 Netlist Viewer 中的对应项，右键单击原理图或状态图中感兴趣的项，指向 **Locate**，并点击相应命令。可用命令如下：

- **Locate in Assignment Editor**
- **Locate in Pin Planner**
- **Locate in Chip Planner**
- **Locate in Resource Property Editor**
- **Locate in Technology Map Viewer**
- **Locate in RTL Viewer**
- **Locate in Design File**

可用于查找项目的选项取决于节点类型，以及布局布线后其是否存在。如果菜单中某个命令已使能，则其可用于所选节点。**Locate in Assignment Editor** 命令可用于所有节点，但如果将其应运于综合后便不存在的节点，则布局布线期间的约束可能会被忽略。

Netlist Viewer 自动打开相应编辑器或布局规划窗口，并在新开窗口中突出显示所选节点和网络。可在 Window 菜单选择切换回 Netlist Viewer，或关闭，最小化和移动新窗口。

2.8. 从其他 Intel Quartus Prime 窗口交叉探查 Netlist Viewer

在 Intel Quartus Prime 软件中可从其他窗口交叉查探 RTL Viewer 和 Technology Map Viewer。可在其他窗口中选择 1 个或多个节点和网络并在其中一个 Netlist Viewer 中进行查看。

可在 RTL Viewer 和 Technology Map Viewer 之间找到节点，还可从以下 Intel Quartus Prime 软件窗口中找到 RTL Viewer 和 Technology Map Viewer 之间的节点：

- Project Navigator
- Timing Closure Floorplan
- Chip Planner
- Resource Property Editor
- Node Finder
- Assignment Editor
- Messages Window
- Compilation Report
- Timing Analyzer（仅支持 Technology Map Viewer）

要从另一 Intel Quartus Prime 软件窗口找到 Netlist Viewer 中的组件，请在相应窗口选择选择一个或多个节点；例如，在 Project Navigator 的 **Hierarchy** 选项卡上，从 **Entity** 清单上选择一个条目，或在 Timing Closure Floorplan 中选择节点，或在 Assignment Editor 中的 **From** 或 **To** 列选择节点名称。然后，右键单击所选对象，指向 **Locate**，并点选 **Locate in RTL Viewer** 或 **Locate in Technology Map Viewer**。点选该命令后，Netlist Viewer 打开，或者如果 Netlist Viewer 已打开则被置顶。

注意: 编译后首次打开窗口时，会先运行预处理器阶段然后才打开 Netlist Viewer。

Netlist Viewer 显示已选节点和节点间的连接（如果适用）。该显示与右键单击目标对象，然后点选 **Filter** > **Selected Nodes** 使用 **Filter across hierarchy** 时相似。如果无法在 Netlist Viewer 中找到节点，则消息框显示此消息：**Can’t find requested location**（无法找到所请求位置）。

2.9. 查看时序路径

完成完整设计编译后（包括时序分析阶段），可从时序报告看到时序路径交叉探查的直观表达。关于生成时序报告的详细信息，请参阅 *Intel Quartus Prime Pro Edition User Guide: Timing Analyzer* 。

找到从 Timing Analyzer 到 Technology Map Viewer 的时序路径后，与每个节点关联的互连和单元延迟将在原理图视图中置顶显示。所选时序路径的总时序裕量（slack）在原理图 Page Title 部分显示。

1. 要打开 Contents 中 **Compilation Report** Table 的报告，请点击 **Timing Analyzer GUI** > **Report Timing**，并双击 timing corner。
2. 要打开 **Timing Analyzer** 中的报告，请从 **Report** 窗格打开 **Report Timing** 文件，并双击 timing corner。
3. 在 **Summary of Paths** 选项卡中，右键单击表格中的行，并选择 **Locate Path** > **Locate in Technology Map Viewer**。Technology Map Viewer 中，原理图页面显示时序路径上的节点以及总延迟的总结。

相关链接

报告时序（对话框）
In *Intel Quartus Prime Pro Edition User Guide: Timing Analyzer*

2.10. 优化设计网表修订历史

以下修订历史适用于本章：

文档版本	Intel Quartus Prime 版本	修订内容
2019.07.01	19.1	添加了*维护 Resource Property Viewer 中的选择*主题以解释如何维护原理图中的 iterm-oterm 从属关系。
2018.09.24	18.1.0	• 为*查看时序路径*添加链接。 • 从“原理与符号”主题中删除了关于不支持 CARRY 缓冲器的参考内容。
2016.10.31	16.1.0	• 品牌更名为 Intel。
2016.05.03	16.0.0	删除了 Schematic Viewer 主题。
2015.11.02	15.1.0	添加了 Schematic Viewer 主题以查看阶段快照。 添加了如下新功能特性和更能更新添加的信息： • “跨层次的网络可见性” • Connection Details（连接详情） • Display Settings（显示设置） • Hand Tool（手形工具） • Area Selection Tool（区域选择工具） • New default behavior for Show/Hide Instance Pins （目前为默认关闭）（显示/隐藏实例管脚的新默认行为）
2014.06.30	14.0.0	在 One Page 和 Show/Hide Instance Pins 命令中添加了 Show Netlist。
2013 年 11 月	13.1.0	删除了 HardCopy 器件信息。 重组并更换为新模板。 添加关于新 Netlist viewer 的支持。
2012 年 11 月	12.1.0	添加了支持 Global Net Routing 功能部分。
2012 年 6 月	12.0.0	删除了反馈问卷链接。
2011 年 11 月	10.0.2	文档模板更新。
2010 年 12 月	10.0.1	更换为新文档模板。
2010 年 7 月	10.0.0	• 更新了截图。 • 更新了 Intel Quartus Prime 软件 10.0 的章节，包括主要用户界面更改。
2009 年 11 月	9.1.0	• 更新器件 • 少量文本编辑
2009 年 3 月	9.0.0	• 第 13 章原为版本 8.1.0 中的第 12 章 • 更新了图 13-2、图 13-3、图 13-4、图 13-14 和图 13-30。 • 添加了“使能或禁用自动层次列表”（第 13 – 15 页） • 更新了第 13-44 页中的“查找命令”。
2008 年 11 月	8.1.0	页面尺寸更改为 8.5” × 11”
2008 年 5 月	8.0.0	• 添加了 Arria GX 支持 • 添加了操作符符号 • 更新了径向菜单功能的信息。

继续...

文档版本	Intel Quartus Prime 版本	修订内容
		• 更新了缩放功能 • 更新了从原理图到 Signal Tap Analyzer 的交叉探查信息。 • 更新了常量信号的信息 • 在支持的图像文件格式清单中添加了.png 和.gif。 • 更新了多个图示和表格。 • 添加了新的部分：“Enabling and Disabling the Radial Menu”（使能和禁用 Radial Menu），“Changing the Time Interval”（更改了时间间隔），“Changing the Constant Signal Value Formatting”（更改了恒定信号值格式），“Logic Clouds in the RTL Viewer”（RTL Viewer 中的逻辑云），“Logic Clouds in the Technology Map Viewer”（Technology Map Viewer 中的逻辑云），“Manually Group and Ungroup Logic Clouds”（手动组合和取消组合逻辑云），“Customizing the Shortcut Commands”（自定义快捷命令） • 重命名多个部分 • 删除了“Customizing the Radial Menu”（自定义 Radial Menu）部分 • 删除了“Grouping Combinational Logic into Logic Clouds”（将组合逻辑分组为逻辑云）部分 • 基于 Intel Quartus Prime 软件 8.0 更新文档内容

相关链接

文档存档

关于之前版本的 *Intel Quartus Prime 手册*，请搜索文档存档。

3. 网表优化和物理综合

Intel Quartus Prime 软件在综合期间提供网表优化以及在适配期间提供物理综合优化，可提高设计性能。综合网表优化使用您设计中的原子网表进行操作，该操作以特定原语术语描述设计。本章提供应用综合和物理综合优化设置，以及通过反向标注（back-annotation）保留配置结果的指导。

可从 **Compiler Settings** 页访问一系列全局综合和物理综合优化设置：

表 7. 综合网表优化和物理综合选项

选项	位置/说明
Enable synthesis netlist optimization settings	在 **Advanced Synthesis Settings** 对话框中使能综合优化选项（例如，**Synthesis Effort**）。单击 **Assignments** > **Settings** > **Compiler Settings** > **Advanced Settings (Synthesis)**访问选项。
Enable physical synthesis options	在 **Advanced Synthesis Settings** 对话框中使能物理综合选项（例如，**Advanced Physical Synthesis**）。单击 **Assignments** > **Settings** > **Compiler Settings** > **Advanced Settings (Fitter)**访问设置。

注意: 由于设计中的原语节点名称在使用物理综合优化时可能发生改变，因而需评估您的设计是否依赖固定的节点名称。如果您使用的验证流程要求固定节点名称，如 Signal Tap Logic Analyzer，正是验证或基于 Logic Lock 的优化流程（对于 legacy 器件），则禁用物理综合选项。

3.1. 物理综合优化

Intel Quartus Prime Fitter 对逻辑单元进行布局和布线，以确保逻辑的关键部分相互靠近且使用尽可能快的布线资源。然而布线延迟通常是常规关键路径延迟的主要部分。物理综合优化通过考量布局信息，布线延迟和时序信息以确定最佳布局。随后 Fitter 着重针对设计关键部分进行时序驱动优化。综合与适配过程的紧密整合被称为物理综合。

以下部分说明 Intel Quartus Prime 软件中可用的物理综合优化，以及其如何帮助提高所选器件的性能和进行适配。

相关链接

Compiler Settings Page (Settings Dialog Box)
Intel Quartus Prime Help 中

3.1.1. 使能物理综合优化

物理综合优化可通过执行组合和顺序优化以及寄存器复制来改善电路性能。

使能物理综合选项：

1. 点击 **Assignments** > **Settings** > **Compiler Settings**。
2. 要使能重定时，组合优化和寄存器复制，可点击 **Advanced Settings (Fitter)**。接下来，使能 **Physical Synthesis**。
3. 在 **Netlist Optimizations** 报告中查看物理综合结果。

3.1.2. 物理综合选项

Intel Quartus Prime 软件提供物理综合优化选项来提高适配结果。要访问这些选项，单击 **Assignments** > **Settings** > **Compiler Settings** > **Advanced Settings (Fitter)**。

注意: 针对设计中特定组件禁用全局物理综合优化，请将特定节点或实体的 **Netlist Optimizations** 逻辑选项约束为 **Never Allow**。

表 8. 物理综合选项

选项	说明
Advanced Physical Synthesis（高级物理综合）	在适配期间使用物理综合引擎执行组合以及顺序优化提高电路性能。
Netlist Optimizations（网表优化）	可使用 Assignment Editor 应用 **Netlist Optimizations** 逻辑选项。使用该选项禁用部分设计的物理综合优化。
Allow Register Duplication（允许寄存器复制）	允许 Compiler 复制寄存器以提高设计性能。使能该选项时，Compiler 复制该寄存器并将其中一些扇出移动到对应的新节点。该优化可改善布通性，并减少具有多个扇出的网络中的布线线缆总数。 如果禁用该选项，则会禁用重定时寄存器的优化。 本设置会影响 Analysis & Synthesis 和 Fitter。
Allow Register Duplication（允许寄存器合并）	允许 Compiler 删除与设计中其他寄存器相同的寄存器。使能该选项后，如果两个寄存器生成相同逻辑，Compiler 删除一个寄存器并将寄存器扇出保留在被删除寄存器的目的地。该选项有助于防止 Compiler 删除特意使用的复制寄存器。 如果禁用寄存器合并，则 Compiler 禁用重定时寄存器的优化。 本设置会影响 Analysis & Synthesis 和 Fitter。

3.2. 应用网表优化

使用网表优化时的性能提高情况取决于具体设计。如果以重新组织设计结构来平衡关键路径延迟，则可能通过网表优化的性能提高度最低。

你可能需要尝试可用选项来查看最适合特定设计的设置组合。请参阅编译报告中的消息查看每个选项的改进程度，并帮助您确定是否应开启给定选项或特定工作级别。

开启更多网表优化选项可能会导致设计中的节点名称发生更多变化；因而如果您正在使用验证流程则需谨记，如 Signal Tap Logic Analyzer 或正式验证就需要固定或已知的节点名称。

为获到最佳效果，可使用 Intel Quartus Prime Design Space Explorer II (DSE)应用各种网表优化选项集。

相关链接

Design Space Explorer II (第 10 页)

3.2.1. WYSIWYG Primitive Resynthesis（WYSIWYG 原语再综合）

对于使用第三方工具进行综合的设计，**Perform WYSIWYG primitive resynthesis** 选项允许将优化应用于已综合的网表。

Perform WYSIWYG primitive resynthesis 选项指示 Intel Quartus Prime 软件取消将原子网表中的逻辑元件（LE）映射到逻辑门，然后将门控重新映射回 Intel 指定原语。第三方综合工具生成使用 Intel 指定原语的 `.edf` 或 `.vqm` 原子网表文件。开启 **Perform WYSIWYG primitive resynthesis** 选项后， Intel Quartus Prime 软件在重新映射进程中使用特定于器件的技术。该功能使用针对工程（**Speed**，**Area** 或 **Balanced**）的 **Optimization Technique** 将设计重新映射。

Perform WYSIWYG primitive resynthesis 选项仅对逻辑单元（也称为 LCELL 或 LE 原语）和常规 I/O 原语（可能包含寄存器）取消映射或重新映射。双数据率（DDR）I/O 原语，存储器原语，数字信号处理（DSP）原语和进位链中的逻辑单元不被重新映射。该进程不处理加密 `.vqm` 文件或 `.edf` 文件中的特定逻辑，如第三方知识产权（IP）。

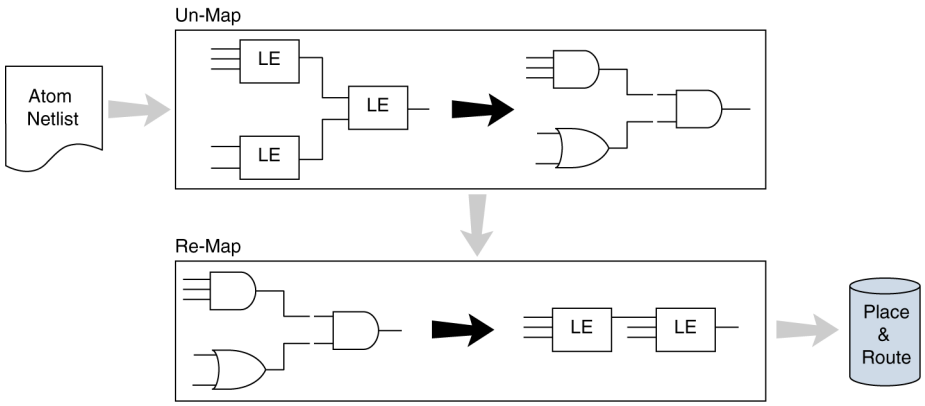
Perform WYSIWYG primitive resynthesis 选项可从第三方综合工具更改 `.vqm` 文件或 `.edf` 文件中的节点名称，因为原子网表中的原语是拆分状，随后由 Intel Quartus Prime 软件对其重新映射。重映射处理会删除重复的寄存器。未被删除的寄存器在重映射后保留相同名称。

任何将 **Netlist Optimizations** 逻辑选项设置为 **Never Allow** 的节点或实体在 WYSIWYG 原语重新综合期间都不受影响。可使用 Assignment Editor 应用 **Netlist Optimizations** 逻辑选项。该选项对设计某些部分禁用 WYSIWYG 再综合。

注意: 综合期间指定原语节点名称。应用网表优化时，节点名称可能会因为原语的创建和删除而更改。无法维护用于保留第三方综合工具中逻辑的 HDL 属性，因为这些属性并未写入 Intel Quartus Prime 软件要读取的原子网表。

如果使用 Intel Quartus Prime 软件综合设计，则可使用 **Preserve Register (preserve)**和 **Keep Combinational Logic (keep)**属性维持设计中的某些节点。

图 11. Intel Quartus Prime 中 WYSIWYG Primitive Resynthesis 的流程



3.3. 脚本支持

可运行本章节 Tcl 脚本中介绍的过程及设置。还可按照命令提示运行一些处理过程。关于脚本命令选项的详细信息，请参阅 Intel Quartus Prime Command-Line 和 Tcl API Help 浏览器。要运行 Help 浏览器，请在提示命令键入如下命令：

```
quartus_sh --qhelp
```

可在实例中或/和全局级中指定本小节中介绍的多个选项。

使用以下 Tcl 命令进行全局约束：

```
set_global_assignment -name <QSF variable name> <value>
```

使用以下 Tcl 命令进行实例约束：

```
set_instance_assignment -name <QSF variable name> <value> \ -to <instance name>
```

相关链接

- Command Line Scripting
- Tcl Scripting
- API Functions for Tcl
  Intel Quartus Prime Help 中
- Intel Quartus Prime Pro Edition 设置文件参考手册
  关于 Intel Quartus Prime 软件中所有设置和约束的信息。

3.3.1. 综合网表优化

工程 `.qsf` 文件保留您在 GUI 中指定的设置。或者，可直接编辑 `.qsf`。`.qsf` 文件支持如下综合网表优化命令。**Type** 栏标示是否支持设置作为全局设置，实例设置，或两者兼而有之。

表 9. 综合网表优化和相关设置

设置名称	Intel Quartus Prime 设置文件变量名称	值	类型
Perform WYSIWYG Primitive Resynthesis	ADV_NETLIST_OPT_SYNTH_WYSIWYG_REMAP	ON, OFF	Global（全局），Instance（实例）
优化模式	OPTIMIZATION_MODE	BALANCED HIGH PERFORMANCE EFFOR AGGRESSIVE PERFORMANCE	Global（全局），Instance（实例）
Power-Up Don’t Care	ALLOW_POWER_UP_DONT_CARE	ON, OFF	Global

3.3.2. 物理综合优化

工程 `.qsf` 文件保留您在 GUI 中指定的设置。或者，可直接编辑 `.qsf`。`.qsf` 文件支持如下综合网表优化命令。**Type** 栏标示是否支持设置作为全局设置，实例设置，或两者兼而有之。

表 10. 物理综合优化和相关设置

设置名称	Intel Quartus Prime 设置文件变量名称	值	类型
Advanced Physical Synthesis	ADVANCED_PHYSICAL_SYNTHESIS	ON, OFF	Global

3.4. 网表优化和物理综合修订历史

以下修订历史适用于本章：

文档版本	Intel Quartus Prime 版本	修订内容
2019.04.24	18.1.0	更新了“网表优化和物理综合”主题中的实例。
2019.04.18	18.1.0	阐明“网表优化和物理综合”主题中的措辞。
2018.09.24	18.1.0	从针对“时序逻辑选项优化 IOC 寄存器布局”主题中删除了关于不支持 CASCADE 缓冲的参考内容。
2018.05.07	18.0.0	删除了主题：*隔离分区网表*。
2017.11.06	17.1.0	• 删除了参考.vqm 文件 • 添加了主题：*隔离分区网表*。
2016.10.31	16.1.0	• 品牌更名为 Intel。 • 更新了物理综合选项和处理过程。
2016.05.02	16.0.0	• 删除关于已淘汰的物理综合选项的内容。
2015.11.02	15.1.0	• 将 *Quartus II* 更改成 *Intel Quartus Prime* 。 • 添加了物理综合。
2014.12.15	14.1.0	• 将 Fitter Setting，Analysis & Synthesis Settings 和 Physical Synthesis Optimizations Setting 的位置更新到 Compiler Setting。 • 更新了 DSE II 的内容。
2014 年 6 月	14.0.0	更新了格式。
2013 年 11 月	13.1.0	删除了 HardCopy 器件信息。
2012 年 6 月	12.0.0	删除了反馈问卷链接。
2011 年 11 月	10.0.2	文档模板更新。
2010 年 12 月	10.0.1	文档模板更新。
2010 年 7 月	10.0.0	• 在多个部分添加 Intel Quartus Prime Help 链接。 • 删除“参考文档”部分。 • 改写文档修订历史
2009 年 11 月	9.1.0	• 为“物理综合寄存器—寄存器重新定时”添加了信息 • 为“应用网表优化选项”添加了信息 • 少量编辑更新。
2009 年 3 月	9.0.0	• 原为 8.1.0 发布中的第 11 章 • 更新了“物理综合寄存器 —寄存器重新定时”和“用于 Fitting 的物理综合选项” • 更新了“执行物理综合优化” • 删除了“门级寄存器重定时”部分。 • 更新了参考文档
2008 年 11 月	8.1.0	页面尺寸更改为“8½ × 11”。文档内容无变化。
2008 年 5 月	8.0.0	• 更新了第 11-9 页的“物理综合性能优化” • 在第 11-16 页添加了“用于 Fitting 的物理综合选项”

相关链接

文档存档

关于之前版本的 *Intel Quartus Prime 手册*，请搜索文档存档。

4. 区域优化

本章说明设计 Intel 器件时减少资源利用率的技术。

4.1. 资源使用情况

无论设计是否完成适配，确定资源利用率都可提供有用信息。如果编译结果中无适配错误，则资源利用率信息有利于分析设计中的适配问题。如果适配成功，该信息允许您确定设否设计更改会引入适配困难。此外，还可确定资源利用率对时序性能的影响。

Compilation Report 提供资源使用情况的信息。

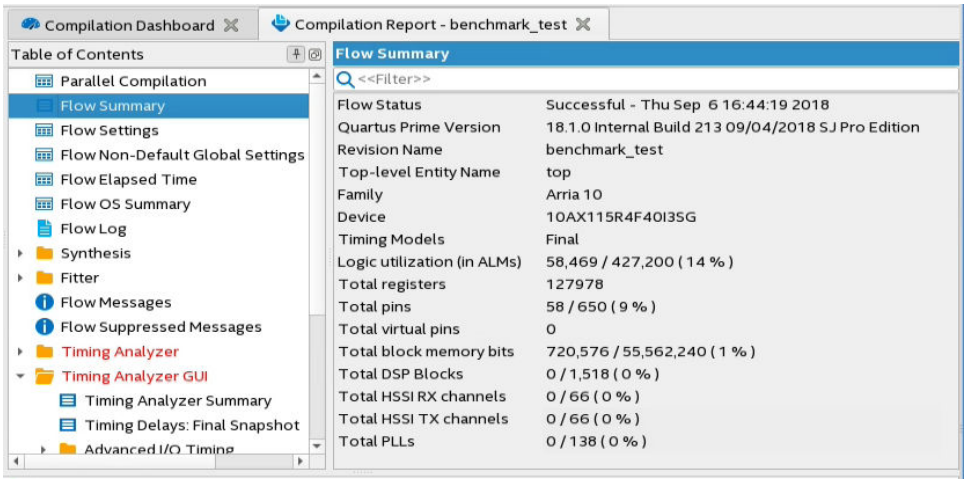
相关链接

查看布线拥塞 (第 101 页)

4.1.1. Flow Summary 报告

编译报告中的 **Flow Summary** 部分指示设计是否超出可用器件资源，并报告资源使用率，包括管脚，存储器位，数字信号处理（DSP）块和锁相环（PLL）。

图 12. Flow Summary 报告



Fitter 可将逻辑扩展至整个器件，但可能会提高整体资源使用率。

器件填满时，Fitter 自动搜索逻辑功能并通过公共输入放置到 ALM 中。被打包的寄存器数量也会增加。因此，如果逻辑和寄存器可更紧密封装在一起，则具有较高整体利用的设计可能仍然具有用于额外逻辑的空间。这样，具有更多详细信息的报告就更有益。

4.1.2. Fitter 报告

在编译报告的 **Fitter** 部分，**Resource Section** 下的报告提供资详细的资源信息。

Fitter Resource Usage Summary 报告分解逻辑资源使用信息并提供其他资源信息，包括每个存储器块类型中的位数。该窗格还包含全局时钟，PLL， DSP 块和其他特定器件资源的使用情况摘要。

相关链接

Fitter Resources Reports

4.1.3. Analysis 和 Synthesis 报告

对于由 Intel Quartus Prime 综合引擎进行综合的设计，可在编译期间查看说明优化的报告。

例如，在 **Analysis & Synthesis** 部分，**Optimization Results** 文件夹中，可找到综合期间被删除的寄存器列表。通过该报告可估算设计部分中的资源使用率，从而确保不因丢失与设计中其他部分中的连接而将寄存器删除。

相关链接

Synthesis Optimization Results Reports

4.1.4. 编译消息

如果报告显示布线资源使用低于 100%，但设计仍未适配，则要么布线资源不足，或者设计包含无效约束。任何一种情况下，Compiler 在 **Messages** 窗口的 **Processing** 选项卡下生成说明问题的消息。

如果 Fitter 未成功完成并且运行速度比类似设计上快得多，则资源可能被过度使用或可能存在非法约束。

如果与类似设计相比， Intel Quartus Prime 软件运行时间过长，则很可能 Compiler 无法找到有效布局或布线。Compilation Report 中，查看标示这些问题类型的错误和警告。

Chip Planner 有助于查找器件中特定类型布线资源中布线拥塞的区域。如果发现具有非常高拥塞的区域，则请分析导致拥塞的原因。诸如不使用全局资源的高扇出网络，错误选择优化目标（速度与面积），非常严格的布局规划约束，或编码风格都会导致布线拥塞。确定原因后，修改源或设置以降低布线拥塞。

相关链接

Viewing Messages

4.2. 优化资源利用率

以下列出设计分析后的阶段：

1. 优化资源使用—确保已经设置基本约束
2. I/O 时序优化—优化资源使用和所需目标器件中的设计适配后，再优化 I/O 时序。
3. Register-to-register 时序优化

相关链接

- 设计优化概述 (第 6 页)

- 时序收敛与优化 (第 52 页)

4.2.1. 资源使用问题概述

资源使用问题可分为三大类：

- 与 *I/O 管脚使用率或布局*有关的问题，包括专用 I/O 块，例如 PLL 或 LVDS 收发器。
- 与*逻辑使用率或布局*有关的问题，包括具有寄存器的逻辑单元和 LUT 以及专用逻辑，如存储器块和 DSP 块。
- 与*布线*有关的问题。

4.2.2. I/O 管脚使用情况或布局

按照指导解决 I/O 资源问题。

4.2.2.1. 指导：修改管脚约束或选择较大的封装

如果设计中因管脚约束失败而无法适配，则请在无管脚约束的情况下编译设计，以确定该设计是否可能用于指定器件和封装。如果有 Intel Quartus Prime 错误消息显示因管脚约束的适配问题就可使用这种方法。

如果忽略所有管脚约束，或忽略和删除多个管脚约束后，设计得以适配，则可能必须修改设计管脚约束或选择较大的封装。

如果由于 I/O 管脚不足而导致设计无法适配，则具有较多可用用户 I/O 管脚的较大器件封装（可以是相同器件密度）能实现成功适配。

相关链接

Managing Device I/O Pins
In *Intel Quartus Prime Pro Edition User Guide: Design Constraints*

4.2.3. 逻辑资源使用或布局

解决逻辑资源使用问题，包括具有寄存器的逻辑单元和 LUT 以及专用逻辑，如存储器块和 DSP 块。

4.2.3.1. 指导：优化源代码

如果设计因逻辑使用率而无法适配，则对源进行评估并修改设计。对源代码进行特定于设计的更改，通常可显著改善逻辑。也是最有效改善结果质量的常用办法。

如果您的设计无法适配到可用逻辑单元（LE）或 ALM，但仍有未使用的存储器或 DSP 块，请检查设计中是否存在说明存储器或 DSP 功能的代码块未被推断且未置于专用逻辑中。可修改源代码，以允许将专用存储器或 DSP 资源置于目标器件中。

请确保您的状态机被识别为状态逻辑且在综合工具中被适当优化。已识别出的状态机通常会被优化，否则被当作一般逻辑处理。在 Intel Quartus Prime 软件中，可从 Compilation Report 的 **Analysis & Synthesis** 下查看 State Machine 报告。该报告提供包括编译期间识别出的每个状态机的状态编码等各种详细信息。如果无法识别您的状态机，则可能必须更改源代码才能识别。

相关链接

- AN 584：高级 FPGA 设计的时序收敛方法

- Recommended HDL Coding Styles
  In *Intel Quartus Prime Pro Edition User Guide: Design Recommendations*

4.2.3.2. 指导：优化针对区域而非速度的综合

如果由于逻辑资源限制，导致 Fitter 无法解析设计，则重新综合设计以提高区域利用率。

首先，确保在综合工具中正确设置器件和时序约束。尤其是在考虑设计中区域利用率时，请务必不要过度约束设计的时序要求。综合工具在满足指定要求时，如果约束过于主动，则可能导致更高的器件资源使用率。

如果资源利用率需重点考虑，则可优化区域而非速度。

- 如果使用 Intel Quartus Prime 综合，单击 **Assignments** > **Settings** > **Compiler Settings** > **Advanced Settings (Synthesis)** 并将 **Optimization Technique** 选择为 **Balanced** 或 **Area**。
- 如果要使用 **Area** 或 **Speed** 设置减少设计中指定模块的面积，同时保留默认 **Optimization Technique** 设置为 **Balanced**，则请使用 Assignment Editor。
- 还可打开 **Speed Optimization Technique for Clock Domains** 逻辑选项优化指定时钟域之内或之间的所有组合逻辑的速度。
- 某些综合工具中，不指定 f_{MAX} 要求可导致低资源利用率。

优化面积或速度会影响 register-to-register 时序性能。

注意: Intel Quartus Prime 软件中，**Balanced** 设置通常会产生与 **Area** 设置非常相似的资源利用率结果。**Area** 设置在某些情况下可提供更好的结果。

Intel Quartus Prime 软件提供其他属性和选项，以助于提高综合结果的质量。

相关链接

优化模式

4.2.3.3. 指导：重组多路复用器

多路复用器在许多 FPGA 设计占逻辑利用率的较大部分。通过优化多路复用逻辑，可在 Intel 器件中达成更有效实现。

相关链接

- 重组多路复用器逻辑选项
  了解关于 Restructure Multiplexers 选项的更多信息。
- 建议的 HDL 编码样式
  实现多路复用器设计最佳资源使用的设计指导

4.2.3.4. 指导：通过 Balanced 或 Area 设置执行 WYSIWYG 原语重新综合

Perform WYSIWYG Primitive Resynthesis 逻辑选项指定综合期间是否执行 WYSIWYG 原语重新综合。该选项使用 **Optimization Technique** 逻辑选项中的指定设置。**Perform WYSIWYG Primitive Resynthesis** 逻辑选项有助于重新综合设计中的一些或全部 WYSIWYG 原语，以获得更适合的面积或性能。然而，仅在使用第三方综合工具时才可进行 WYSIWYG 原语再综合。

注意: **Balanced** 设置通常生成与 **Area** 设置非常相似的利用率结果，以获得更好的性能结果。**Area** 设置在某些情况下可带来更好结果。这种方式执行 WYSIWYG 再综合通常会降低 register-to-register 时序性能。

相关链接

执行 WYSIWYG 原语再综合逻辑选项
关于该逻辑选项的信息

4.2.3.5. 指导：使用寄存器封装

Auto Packed Registers 选项通过将仅使用寄存器和仅使用 LUT 的单元中的寄存器和 LUT 合并，进而以一个逻辑单元实现两个单元的功能。

相关链接

自动封装寄存器逻辑选项
了解关于 Auto Packed Register 逻辑选项的更多信息。

4.2.3.6. 指导：删除 Fitter 约束

设计中存在约束冲突或难以满足甚至不适合目标器件的约束。例如，如果位置或 Logic Lock 约束过于严格且器件中无足够的布线资源可用，则设计可能适配失败。

要解决限制性位置约束或 Logic Lock 区域约束导致的布线拥塞，请使用 Chip Planner 中的 **Routing Congestion** 任务解决布局规划中的布线问题，然后从该区域中删除内部位置或 Logic Lock 区域约束。如果设计仍无法适配，则设计被过度约束。为解决该问题，可删除所有位置和 Logic Lock 约束并运行连续编译，每次编译之前逐步约束设计。可从 Assignment Editor 或 Chip Planner 中删除特定位置约束。要从 Chip Planner，Logic Lock Regions Window 或 Assignments 菜单中删除 Logic Lock 约束，请点击 **Remove Assignments**。在 **Available assignment categories** 列表中打开要从设计中删除的约束类别。

相关链接

分析和优化设计平面布局规划 (第 96 页)

4.2.3.7. 指导：综合期间展开层级结构

综合工具通常提供保留层级结构边界的选项，有助于验证或其他目的。然而， Intel Quartus Prime 软件跨层级边界优化，以便执行最大程度的逻辑最小化，从而减少设计中无设计分区的区域。

4.2.3.8. 指导：更换目标存储器块

如果由于存储器资源现在导致 Fitter 无法解析设计，则设计可能需要器件中没有的存储器类型。

对于使用 Parameter Editor 创建的存储器块，编辑 RAM 块类型以针对另一尺寸的存储器块。

Compiler 还可从 HDL 代码中推断 ROM 和 RAM 存储器块，且综合引擎可通过推断 Shift 寄存器（基于 RAM）IP 核来讲大型移位寄存器放入存储器块中。关闭综合工具中的此推断功能时，综合引擎将存储器或移位寄存器放入逻辑而非存储器块。此外，关闭高推断可防止寄存器被移动到 RAM 中，从而提高时序性能。

根据综合工具，也可为已推断的存储器块设置 RAM 块类型。 Intel Quartus Prime 综合中，设置 **ramstyle** 属性以说明已推断 RAM 块的存储器类型。或者，将该选项设置为 **logic** 以标准逻辑实现存储器块。

参考报告文件中 Entity 报告的 Resource Utilization 内容以决定是否任何模块中存在异常高的寄存器计数。由于块的体系结构实现，一些编码样式会阻止 Intel Quartus Prime 软件从源代码推断 RAM 块，并强制软件在触发器中实现逻辑。例如，寄存器 bank 上的异步同步可能使寄存器组与器件体系结构中的 RAM 块不兼容，因此 Compiler 在触发器中实现寄存器 bank。稍许修改相关逻辑，就可将大型寄存器 bank 移动到 RAM 中。

相关链接

- Inferring Shift Registers in HDL Code
- Fitter Resource Utilization by Entity Report

4.2.3.9. 指导：使用物理综合选项减少区域

Assignments > **Settings** > **Compiler Settings** > **Advanced Settings (Fitter)**中可用的物理综合选项可帮助减少资源使用。使能物理综合时， Intel Quartus Prime 软件对网表进行特定布局更改，以降低指定 Intel 器件中的资源使用率。

注意: 物理综合会增加编译时间。要减少对编译时间的影响，可对指定实例应用物理综合选项。

相关链接

高级 Fitter 设置对话框

4.2.3.10. 指导：平衡或更换目标 DSP 块

如果设计需要过多 DSP 块，则可能无法成功适配。可使用逻辑单元实现所有 DSP 块功能，因而可更换逻辑的目标 DSP 块以完成适配。

如果通过参数编辑器创建 DSP 功能，请打开参数编辑器并编辑该功能，使其以逻辑单元为目标对象而非 DSP 块。 Intel Quartus Prime 软件使用 DEDICATED_MULTIPLIER_CIRCUITRY IP 核参数控制该实现。

还可通过乘法器，乘法加法器和乘法累加器的 HDL 代码推断 DSP 块。可在综合工具中关闭此推断功能。使用 Intel Quartus Prime 综合时，可关闭 **Auto DSP Block Replacement** 逻辑选项在整个工程中禁用推断功能。点击 **Assignments** > **Settings** > **Compiler Settings** > **Advanced Settings (Synthesis)**。关闭 **Auto DSP Block Replacement**。或者，通过 Assignment Editor 针对指定块禁用该功能。

Intel Quartus Prime 软件还提供 **DSP Block Balancing** 逻辑选项，以在逻辑单元或不同 DSP 块模式中实现 DSP 块元件。默认 **Auto** 设置允许 DSP 块平衡设置以适当自动转换 DSP 块切片从而最小化面积并最大化设计速度。可对指定节点或实体使用其他设置，或基于工程内容，控制 Intel Quartus Prime 软件如何将 DSP 功能转换成逻辑单元和 DSP 块。使用 **Auto** 或 **Off** 以外的值覆盖 IP 核类型中使用的 DEDICATED_MULTIPLIER_CIRCUITRY 参数。

4.2.3.11. 指导：使用大型器件

如果由于缺乏布线资源而导致适配无法成功完成，则可能需要使用较大的器件。

4.2.4. 布线

按照指导解决布线资源问题。

4.2.4.1. 指导：将 Auto Packed Register 设置为 Sparse 或 Sparse Auto

Auto Packed Registers 选项可减少设计中 LE 或 ALM 的数量。可点击 **Assignment** > **Settings** > **Compiler Settings** > **Advanced Settings (Fitter)**设置该选项。

相关链接

自动封装寄存器逻辑选项

4.2.4.2. 指导：将 Fitter Aggressive Routability Optimizations 设置为 Always

如果由于过多布线线缆使用率而导致设计无法适配，则 **Fitter Aggressive Routability Optimization** 选项可帮助解决。

如果布局布线时间之间存在显著不平衡（首次适配尝试期间），则可能是线缆利用率高造成。开启 **Fitter Aggressive Routability Optimizations** 选项可缩短编译时间。

平均而言，该选项可节省线缆使用率最高达 6%，但也会将性能降低最多达 4%，具体取决于器件。

相关链接

Fitter 主动性布通性优化逻辑选项设置

4.2.4.3. 指导：增加 Router Effort Multiplier

Router Effort Multiplier（布线程序效力倍增器）控制布线程序查找有效解决方案的速度。默认值为 1.0 且合法值必须大于 0。

- 高于 1 的数字有利于难以通过提高布线效力进行布线的设计。
- 接近 0 的数字（例如，0.1）可缩短布线程序运行时间，但通常也会略微降低布线质量。

实验证据表明，3.0 倍的倍增器可降低整体线缆使用率大约 2%。使用大于默认值的 Router Effort Multiplier 有益于超过五级逻辑且具有复杂数据路径的设计。然而，设计中的拥塞主要因为布局，但增加 Router Effort Multiplier 不一定减少拥塞。

注意: 任何大于 4 的 Router Effort Multiplier 值，每增加 1 都能提高 10%使用率。例如，值 10 实际上是 4.6。

4.2.4.4. 指导：删除 Fitter 约束

设计中存在约束冲突或难以满足甚至不适合目标器件的约束。例如，如果位置或 Logic Lock 约束过于严格且器件中无足够的布线资源可用，则设计可能适配失败。

要解决限制性位置约束或 Logic Lock 区域约束导致的布线拥塞，请使用 Chip Planner 中的 **Routing Congestion** 任务解决布局规划中的布线问题，然后从该区域中删除内部位置或 Logic Lock 区域约束。如果设计仍无法适配，则设计被过度约束。为解决该问题，可删除所有位置和 Logic Lock 约束并运行连续编译，每次编译之前逐步约束设计。可从 Assignment Editor 或 Chip Planner 中删除特定位置约束。要从 Chip Planner，Logic Lock Regions Window 或 Assignments 菜单中删除 Logic Lock 约束，请点击 **Remove Assignments**。在 **Available assignment categories** 列表中打开要从设计中删除的约束类别。

相关链接

分析和优化设计平面布局规划 (第 96 页)

4.2.4.5. 指导：优化针对区域而非速度的综合

某些情况下，重新综合设计提高区域使用率还可改善设计的布通性。首先，确保综合工具中已正确设置器件和时序约束。请勿过度约束设计的时序要求，尤其是需考量设计中区域使用率时。通常综合工具会尝试满足指定要求，如果约束过激，则会导致较高器件资源使用率。

如果资源使用率是重要考量，则可优化区域而非速度。

- 如果使用 Intel Quartus Prime 综合，单击 **Assignments > Settings > Compiler Settings > Advanced Settings (Synthesis)** 并将 **Optimization Technique** 选择为 **Balanced** 或 **Area**。
- 如果要使用 **Area** 或 **Speed** 设置减少设计中指定模块的面积，同时保留默认 **Optimization Technique** 设置为 **Balanced**，则请使用 Assignment Editor。
- 还可使用 **Speed Optimization Technique for Clock Domains** 逻辑选项指定对特定时钟域内或之间的所有组合逻辑进行速度优化。
- 某些综合工具中，不指定 f_{MAX} 要求可导致较低资源利用率。

优化面积或速度会影响 register-to-register 时序性能。

注意: Intel Quartus Prime 软件中，**Balanced** 设置通常会产生与 **Area** 设置非常相似的资源利用率结果。**Area** 设置在某些情况下可提供更好的结果。

Intel Quartus Prime 软件提供其他属性和选项，以助于提高综合结果的质量。

相关链接

优化模式

4.2.4.6. 指导：优化源代码

如果因为布线问题以及前述部分中介绍的方法未显著提高设计布通性而导致设计无法适配，请从源处修改设计以达到所需结果。通常对源代码进行设计相关的修改，可显著改善结果，例如复制逻辑或更改需要大量布线资源的块之间的连接。

4.2.4.7. 指导：使用大型器件

如果由于缺乏布线资源而导致适配无法成功完成，则可能需要使用较大的器件。

4.3. 脚本支持

可运行本章中 Tcl 脚本说明的过程并进行约束设置。还可按照提示命令指示运行。关于脚本选项的详细信息，请参阅 Intel Quartus Prime 命令行和 Tcl API Help 浏览器。运行“Help”浏览器，可在提示命令键入如下命令：

1. 运行 Help 浏览器，在命令提示中键入如下命令：

```
quartus_sh --qhelp
```

可在实例中或/和全局级中指定本小节中介绍的多个选项。

2. 使用以下 Tcl 命令进行全局约束：

```
set_global_assignment -name <QSF variable name> <value>
```

3. 使用以下 Tcl 命令进行实例约束：

```
set_instance_assignment -name <QSF variable name> <value> \ -to <instance name>
```

注意: 如果<*value*>字段包含空格（例如，‘Standard Fit’），则必须以英文双引号附上其值。

相关链接

- Intel Quartus Prime Pro 设置文件参考手册
  关于 Intel Quartus Prime 软件中所有设置和约束的信息。
- Tcl Scripting
  Intel Quartus Prime Pro Edition 用户指南：脚本
- Command Line Scripting
  Intel Quartus Prime Pro Edition 用户指南：脚本

4.3.1. 初始编译设置

使用 Tcl 约束中的 Intel Quartus Prime Settings File (`.qsf`) 变量名以及正确值进行设置。**Type** 栏标示设置是否用做全局设置，实例设置，或两者兼顾。

表 11. 高级编译设置

设置名称	.qsf 文件类型名称	值	类型
Placement Effort Multiplier	PLACEMENT_EFFORT_MULTIPLIER	任何正，非零值	Global
Router Effort Multiplier	ROUTER_EFFORT_MULTIPLIER	任何正，非零值	Global
Router Timing Optimization level	ROUTER_TIMING_OPTIMIZATION_LEVEL	NORMAL, MINIMUM, MAXIMUM	Global
Final Placement Optimization	FINAL_PLACEMENT_OPTIMIZATION	ALWAYS, AUTOMATICALLY, NEVER	Global

4.3.2. 资源使用优化技术

本表格列出 Resource Utilization Optimization 设置的 QSF 约束和适用值：

表 12. 资源使用优化设置

设置名称	.qsf 文件类型名称	值	类型
Auto Packed Registers	QII_AUTO_PACKED_REGISTERS	AUTO, OFF, NORMAL, MINIMIZE AREA, MINIMIZE AREA WITH CHAINS, SPARSE, SPARSE AUTO	Global（全局），Instance（实例）
Perform WYSIWYG Primitive Resynthesis	ADV_NETLIST_OPT_SYNTH_WYSIWYG_REMAP	ON, OFF	Global, Instance
Optimization Technique	OPTIMIZATION_TECHNIQUE	AREA, SPEED, BALANCED	Global, Instance

继续...

设置名称	.qsf 文件类型名称	值	类型
Speed Optimization Technique for Clock Domains	SYNTH_CRITICAL_CLOCK	ON, OFF	Instance
State Machine Encoding	STATE_MACHINE_PROCESSING	AUTO, ONE-HOT, GRAY, JOHNSON, MINIMAL BITS, ONE-HOT, SEQUENTIAL, USER-ENCODE	Global, Instance
Auto RAM Replacement	AUTO_RAM_RECOGNITION	ON, OFF	Global, Instance
Auto ROM Replacement	AUTO_ROM_RECOGNITION	ON, OFF	Global, Instance
Auto Shift Register Replacement	AUTO_SHIFT_REGISTER_RECOGNITION	ON, OFF	Global, Instance
Auto Block Replacement	AUTO_DSP_RECOGNITION	ON, OFF	Global, Instance
Number of Processors for Parallel Compilation	NUM_PARALLEL_PROCESSORS	包括 1 和 16 之间的整数，或 ALL	Global

4.4. 区域优化修订历史

以下修订历史适用于本章：

文档版本	Intel Quartus Prime 版本	修订内容
2018.1018	18.1.0	• 更正了优化模式 Help 主题的链接
2018.09.24	18.1.0	• 主题划分：*资源使用分为*：*资源使用情况信息*，*流程摘要报告*，*Fitter 报告*，*分析和综合报告*，*编译消息*。
2018.07.03	18.00	修复笔误并在主题中添加了链接*指导：更换目标存储器块*。
2017.05.08	17.0.0	• 删除了关于反对使用集成综合的信息 • 修订主题：*解决资源使用率问题*，*指导：优化针对区域而非速度的综合*
2016.10.31	16.1.0	• 品牌更名为 Intel。
2016.05.02	16.0.0	• 删除对已淘汰物理综合选项的参考。
2015.11.02	15.1.0	将 *Quartus II* 更改为 *Quartus Prime*。
2014.12.15	14.1.0	将 Fitter Setting，Analysis & Synthesis 和 Physical Synthesis Optimizations Setting 的位置更新到 Compiler Setting。
2014 年 6 月	14.0.0	• 删除了 Cyclone III 和 Stratix III 器件参考。 • 删除了基于 Macrocell 的 CPLD 的相关信息。 • 更新了模板。
2013 年 5 月	13.0.0	首次发布。

相关链接

文档存档

关于之前版本的 *Intel Quartus Prime 手册*，请搜索文档存档。

5. 时序收敛与优化

本章介绍在设计 Intel 器件时提高时序性能的技术。具体应用技术因设计而已。应用各个技术后并不一定改善设计结果。

Intel Quartus Prime 软件中的默认设置和选项提供编译时间，资源利用率和时序性能之间的最佳平衡。可调整这些设置确定是否其他设置能为设计取得更佳结果。

注意: 某些技术适用于特定系列。

5.1. 优化 Multi Corner 时序

工艺变化和操作条件导致路径延迟，但其显著小于 slow corner timing 模型中的路径延迟。因而，设计中的这些路径上会出现保持时间违规，并且极少情况下还会出现额外建立时间违规。

此外，针对具有较小工艺几何结构的新器件系列的设计在最高操作温度下并不一定出现最慢电路性能。使电路最慢运行的温度取决于所选器件，设计和编译结果。 Intel Quartus Prime 软件为新的器件系列提供三种不同的 timing corner 以管理这种新的依存关系—Slow 85°C corner，Slow 0°C corner 和 Fast 0°C corner。对于其他器件系列，有 2 个 timing corner 可用—Fast 0°C 和 Slow 85°C corner。

Optimize multi-corner timing 选项指示 Fitter 满足所有工艺极限（process corner）和操作条件下的时序要求。最终的设计实现稳健应对工艺，温度和电压差异。该选项默认开启，并增加大约 10%的编译时间。

该选项关闭后，Fitter 仅鉴于 slow-corner 时序模式（最慢的制造工艺器件用于给定速度等级并在低电压条件操作）的 slow-corner 延迟进行设计优化。

5.2. 关键路径

关键路径是设计中具有负时序余量的时序路径。关键路径涉及器件 I/O 到内部寄存器，寄存器到寄存器，或从寄存器到器件 I/O。

路径时序余量决定其关键度；该时序余量呈现在时序分析报告中，而时序报告可通过 Timing Analyzer 生成。

针对时序收敛的设计分析是高度复杂设计中最佳性能的基本要求。Chip Planner 的分析能力有助于复杂设计中的时序收敛。

相关链接

- 减短关键路径延迟 (第 8 页)
- 使用 Timing Analyzer 显示 Path Report (第 67 页)

5.2.1. 查看关键路径

在 Chip Planner 中查看关键路径可了解特定路径失败的原因。可查看布局中的任何修改是否能减少负时序余量。要显示布局规划（floorplan）中的路径，请通过 Timing Analyzer 执行时序分析并显示结果。

5.3. 关键链

Intel Stratix® 10 器件系列使用 Hyper-Aware 设计流程缩短设计周期并优化性能。Hyper-Aware 设计流程将自动寄存器重定时和实现目标时序收敛建议（Fast Forward 编译）相结合，最大限度地利用 Hyper-Registers，并实现 Intel Stratix 10 设计的最高性能。

关键链报告限制进一步寄存器重新定时优化的设计路径。 Intel Quartus Prime Pro Edition 软件提供 Hyper-Retimer 关键链报告以助提高设计性能。可专注于更高级别的优化，因为 Hyper-Retimer 使用 Hyper-Register 平衡关键链中所有寄存器的时间裕量。

更多关于使用 Hyper-Retimer 关键链报告提高设计性能的信息，请参阅 *Intel Stratix 10 高性能设计手册*中的*解读关键链报告*主题。

相关链接

Running the Hyper-Aware Design Flow
In *Intel Stratix 10 高性能设计手册*

5.3.1. 查看关键链

查看关键链会显示限制设计中重新定时操作的确切逻辑。例如，可查看是否 RTL 代码，或设计中应用的约束限制了重新定时。 Intel Quartus Prime Pro Edition 按照每时钟域和时钟域交叉 1 个关键链进行报告。

关键链在 Hyper Aware Design Flow 中的两个不同阶段可用：

- Retiming Limit Details Report（重定时限制详情报告）中：

  该报告与 Hyper Aware Design Flow 中的重定时阶段相关联，且为默认启动状态。
- Fast Forward Compilation Report（快进编译报告）中：

  Fast Forward Compilation 阶段为可选，且默认为禁用。可从 Compilation Dashboard 使能该阶段。或者，通过单击 **Compilations** 任务中的 **Fast Forward Timing Closure Recommendations** 直接开启本任务。
- 还可在 Technology Map Viewer 中以图形方式查看关键链。更多详情，请参阅 *Intel Stratix 10 高性能设计手册* 中的*查找关键链方位*。

相关链接

- Locate Critical Chains
  In *Intel Stratix 10 高性能设计手册*
- Intel Stratix 10 HyperFlex 设计：分析关键链（OS10CRCHNS）
  在线课程

5.4. 时序收敛的设计评估

设计中出现时序失败时，可遵循本节中的指导。该指导演示了如何评估设计的编译结果以及如何应对问题。虽然该指导中未涵盖重组 RTL 提高设计速度的具体示范，但本分析技术可帮助评估对 RTL 的更改并利于时序收敛。

5.4.1. 查看编译结果

5.4.1.1. 查看消息

编译设计后，请查看编译报告各部分中的消息。

大多数时序失败的设计皆因其他问题而起，编译期间 Fitter 将其报告为警告消息。确定导致警告消息的原因，以及是否需要修复或忽略警告。

查看警告消息后，请查看信息消息。标注意外情况，例如，未连接的端口，忽略的约束，丢失的文件和软件做出的假设和优化。

5.4.1.2. 评估 Fitter 网表优化

Fitter 还可对设计网表执行优化。主要变化包括寄存器封装，复制或删除逻辑单元，反转信号或以常规方式修改节点，例如将输入从一个逻辑单元移动到另一个。可在 Fitter 部分的 Netlist Optimizations 结果中查找并查看这些报告。

5.4.1.3. 评估优化结果

在检查完已完成优化和性能提高情况后，请评估为获得额外性能所花费的运行时间。为减少编译时间，请在几次编译后查看物理综合和网表优化，并编辑 RTL 以反映物理综合执行的更改。如果一组特定的寄存器持续重新定时，则可编辑 RTL 以相同方式重新定时寄存器。如果是为匹配物理综合算法而进行更改，则可关闭物理综合选项以节省编译时间并获得同等类型的性能改善。

5.4.1.4. 评估资源使用情况

要评估设计中各种资源的使用情况，包括全局和非全局信号使用，布线使用情况和集群难度。

5.4.1.4.1. 全局和非全局使用情况

对于包含多个时钟的设计，请评估全局和非全局信号以确定是否有效使用全局资源，如果未有效使用，请考虑更改。可在 **Compilation Report** 的 Fitter 下 Resource 部分找到这些报告。

该图示显示未有效使用全局时钟的示例。突出显示的行具有来自全局时钟的单个扇出。

图 13. 全局时钟的低效使用

Global & Other Fast Signals			
Location	Fan-Out	Global Resource Used	Global Line Name
FRACTIONALPLL_X98_Y2_N0	1	Global Clock	--
PLLOUTPUTCOUNTER_X98_Y2_N1	29044	Global Clock	GCLK7
PLLOUTPUTCOUNTER_X98_Y13_N1	253103	Global Clock	GCLK6
FF_X185_Y66_N13	280349	Global Clock	GCLK8
PIN_AE17	4887	Global Clock	GCLK4
FRACTIONALPLL_X98_Y11_N0	1	Global Clock	--
PLLOUTPUTCOUNTER_X98_Y3_N1	1	Global Clock	GCLK5
PLLOUTPUTCOUNTER_X98_Y1_N1	1691	Regional Clock	RCLK29
PLLOUTPUTCOUNTER_X98_Y8_N1	302	Regional Clock	RCLK23
PLLOUTPUTCOUNTER_X98_Y11_N1	141	Regional Clock	RCLK25
PLLOUTPUTCOUNTER_X98_Y10_N1	17	Regional Clock	RCLK22

如果将这些资源约束给 Regional Clock（区域时钟），则 Global Clock（全局时钟）可用于另一信号。可忽略 **Global Line Name** 栏中的空值信号，因为该信号用于专属布线，并不是时钟缓冲器。

Non-Global High Fan-Out Signal 报告列出未在全局信号上路由的最高扇出节点。

`Reset` 和 `enable` 信号显示于列表顶部。

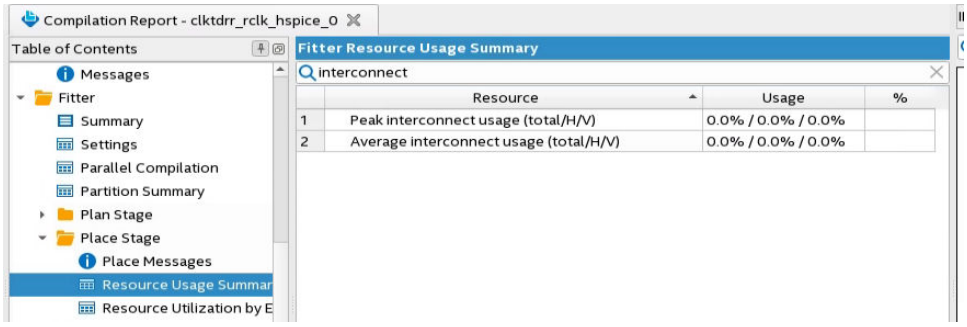
如果设计中存在路由拥塞，且拥塞区域中有高扇出非全局节点，则请考虑使用全局或区域信号扇出节点，或复制高扇出寄存器以使每个复制部分都仅有较少扇出。

使用 Chip Planner 找到高扇出节点从而报告路由拥塞，并确定备选方案是否可行。

5.4.1.4.2. 布线使用

在 **Fitter Resource Usage Summary** 报告中查看布线使用情况。

图 14. Fitter 资源使用情况总结报告

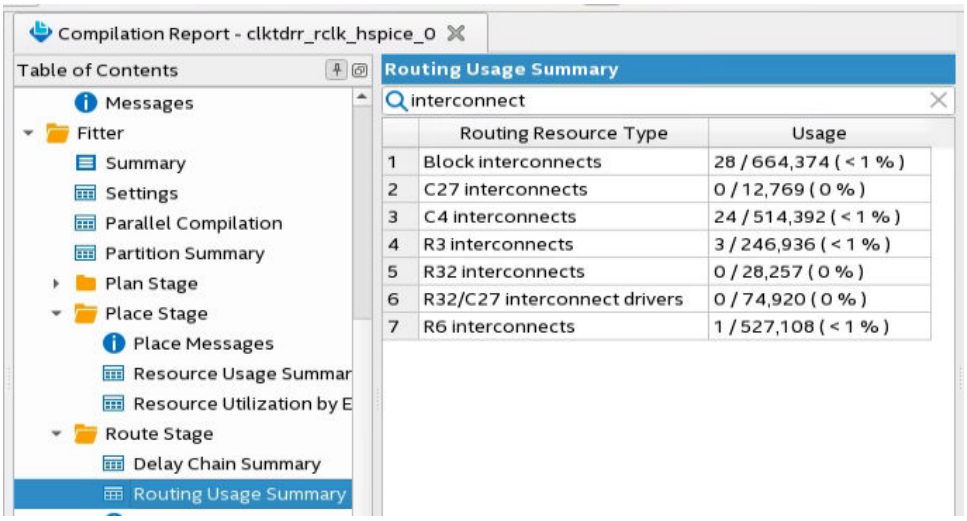


Average interconnect usage 报告器件上可用互连中已使用的互连平均量。**Peak interconnect usage** 报告最拥塞区域中所使用的最大互连量。

平均值低于 50%的设计通常都无布线问题。平均值介于 50-65%之间的设计可能出现布线困难。而平均值超过 65%的设计通常难以满足时序要求，除非 RTL 支持高利用率芯片。峰值达到或超过 90%的情况下很可能在时序收敛时出现问题；峰值达到 100%表示器件该区域内的所有布线已被使用，因此时序性能降低的可能性很高。

该图显示为 **Report Routing Utilization** 报告。

图 15. 布线使用情况报告

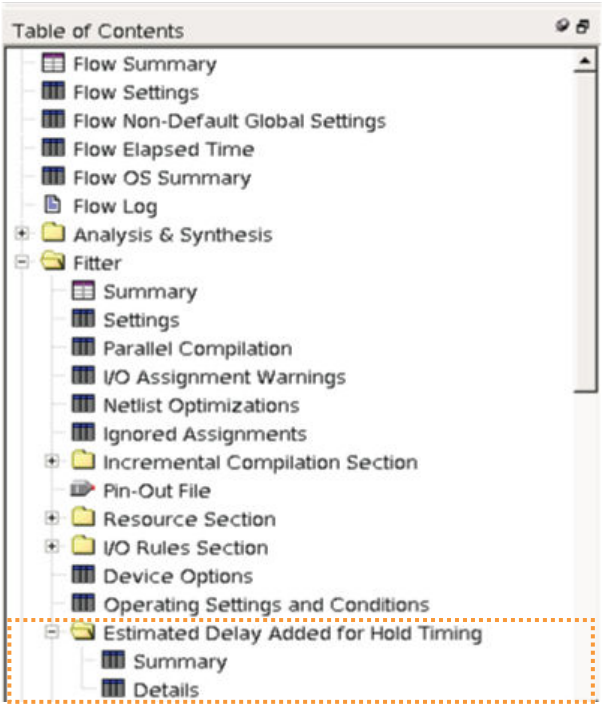


5.4.1.4.3. 添加线缆以“保持”

布线期间 Fitter 可能在寄存器路径之间添加线缆以增加延迟，从而满足保持时间要求。Fitter 在 **Estimated Delay Added for Hold Timing** 中报告已添加的布线延迟的量。过多添加线缆可能出现约束错误。导致该错误的原因通常是多速率时钟之间，以及不同时钟网络之间不正确的多周期传输。

查看 **Estimated Delay Added for Hold Timing** 报告中指定寄存器路径以确定 Fitter 是否添加过多线缆以满足保持时序要求。

图 16. Hold Timing Report 的预估添加延迟

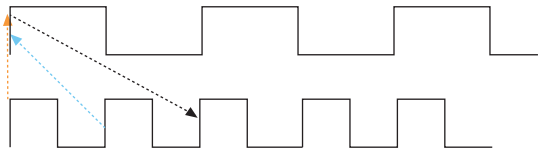


当数据传输从 1x 到 2x 时钟时，导致路由器为满足保持要求而添加线缆的错误约束示例。假设设计意图是允许每传输两个周期。而通过添加多周期建立约束，数据可在两个目标时钟周期内的任何时间达到，如范例中所示：

```
set_multicycle_path -from 1x -to 2x -setup -end 2
```

时序要求放松一个 2x 时钟周期，如图示波形中的黑色线条所示。

图 17. 时序要求的松弛波形



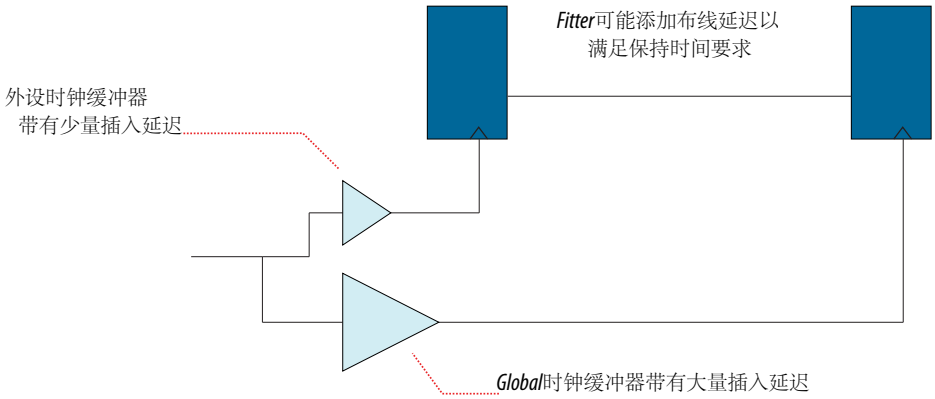
默认保持要求（如蓝色虚线所示），可强制路由器添加线缆以保证数据延迟一个周期。要更正保持要求，通过保持选项添加一个多周期约束。

```
set_multicycle_path -from 1x -to 2x -setup -end 2
set_multicycle_path -from 1x -to 2x -hold -end 1
```

上图中橙色虚线代表保持关系，且无需额外线缆来延迟数据。

当数据在同一时钟域中传输时，布线程序也可添加线缆以保持时序要求，但各时钟分支之间使用不同缓冲。时钟网络类型间的传送常常发生在外设和内核之间。下图显示数据正进入器件，外设时钟驱动源寄存器，全局时钟驱动目标寄存器。全局时钟缓冲器的插入延迟大于外设时钟缓冲器。所以到目标寄存器的时钟延迟远大于到源寄存器的时钟延迟，因此数据路径上需要额外的延迟以确保满足其保持要求。

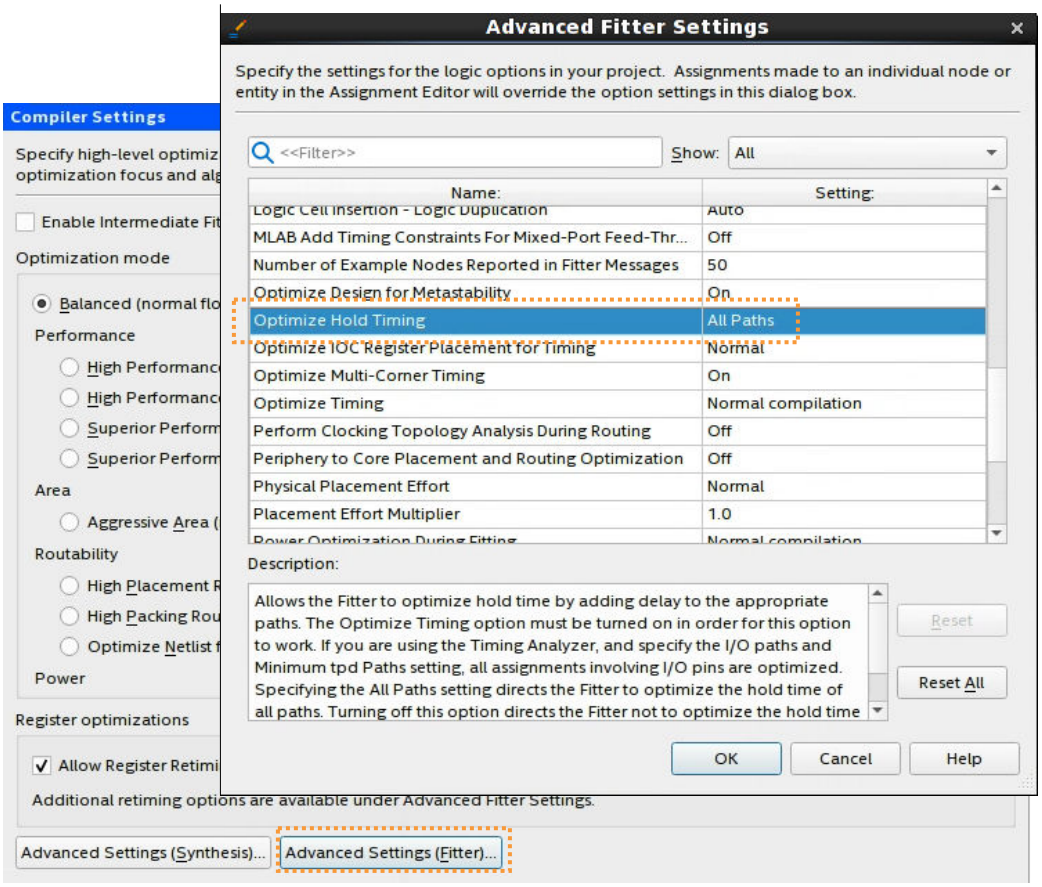
图 18. 时钟延迟



要确认路径中各种时钟网络类型，请在 Timing Analyzer 中查看路径，并沿源时钟路径和目标时钟路径检查节点。此外，检查源时钟频率和目标时钟频率以查看两者是否相同，或成倍数，以及路径上是否存在多周期异常。最后，请确保所有有意的跨域路径错误都有相关联的错误路径异常。

如果您怀疑已添加了修复实际保持问题的路径，则可禁用 **Optimize hold timing** 高级 Fitter 设置（**Assignments > Settings > Compiler Settings > Advanced Settings (Fitter) > Optimize hold timing**）。在禁用 **Optimize hold timing** 的条件下重新编译设计，然后重新运行分析以确认并更正不符合保持时间要求的路径。

图 19. 优化保持时间选项



注意: 仅在调试设计时才禁用 **Optimize hold timing** 选项。请确保常规编译期间启用该选项（默认状态）。布线期间针对保持时间而添加的线缆是时序优化的常规部分，不一定是存在问题。

5.4.1.5. 评估其他报告并进行相应调整

5.4.1.5.1. 难以封装的设计

Fitter Resource Section 中，在 **Resource Usage Summary** 下，查看 **Difficulty Packing Design** 报告。**Difficulty Packing Design** 详细报告 Fitter 在将设计适配到器件，分区和 Logic Lock 区域时的工作级别（低，中或高）。

随着 **Difficulty Packing Design** 的难度不断增加，时序收敛变得更加困难。水平从中到高可能导致性能显著下降或编译时间增加。可考虑减少逻辑以降低封装难度。

5.4.1.5.2. 查看被忽略的约束

Compilation Report 包含被 Fitter 忽略的所有约束详情。如果设计名称更改且未更新约束，则通常旧的约束会被忽略。请务必确保任何预期约束未被忽略。

5.4.1.5.3. 查看非默认设置

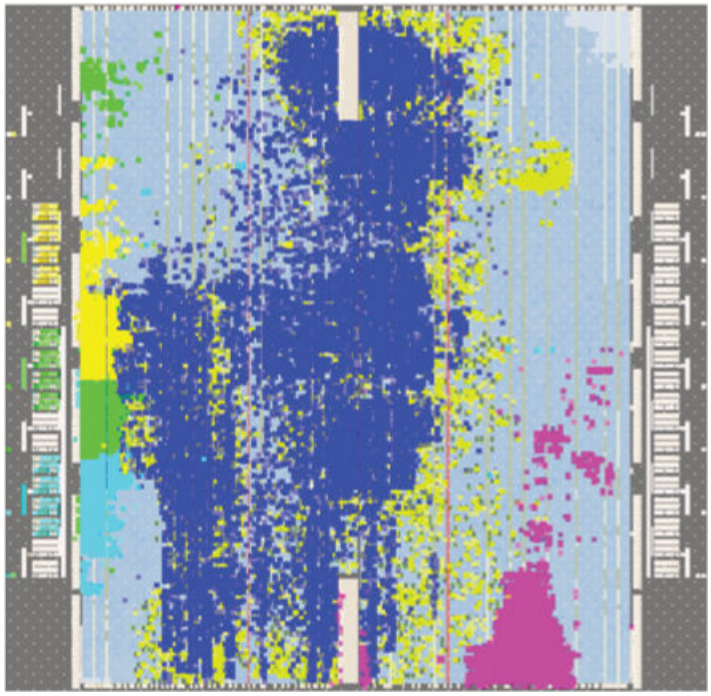
Synthesis 和 Fitter 的报告显示编译中使用的非默认设置。查看非默认设置以确保更改有助于改善设计。

5.4.1.5.4. 查看布局规划

使用 Chip Planner 查看布局。可使用 Chip Planner，通过布局规划平面图中每个实体的不同颜色来找到层次实体。对于未归位的的逻辑，可根据您对其预计的位置进行查找。

例如，与 I/O 对接的逻辑应靠近 I/O，而与 IP 或存储器对接的逻辑应靠近 IP 或存储器。

图 20. 以颜色编码的布局规划图



以下注释说明如何使用 *Floorplan with Color-Coded Entities*（以颜色编码的布局规划图）中的可视性检查时序路径：

- 绿色块分散排开。检查这些路径是否时序失败，如果失败则与该模块连接的部分可能影响布局。
- 蓝色和水蓝色散开且混合相间。检查该状态是否因两种模块间的连接而形成。
- 位于底部的粉色逻辑必须与底部边缘的 I/O 对接。可通过任务栏上的按钮检查突出显示模块的扇入和扇出。

  检查跨芯片的长程信号，并查看是否这些信号导致时序失败。
- 检查影响逻辑布局的信号的全局信号使用情况，并验证经 Fitter 置位的逻辑是否靠近其进行馈送的缓冲器，但远离相关逻辑。对非全局资源使用高扇出设置来拉拢逻辑。
- 检查布线拥塞。Fitter 中逻辑分散到高拥塞区域，使得设计中难以布线。

5.4.1.5.5. 评估布局布线

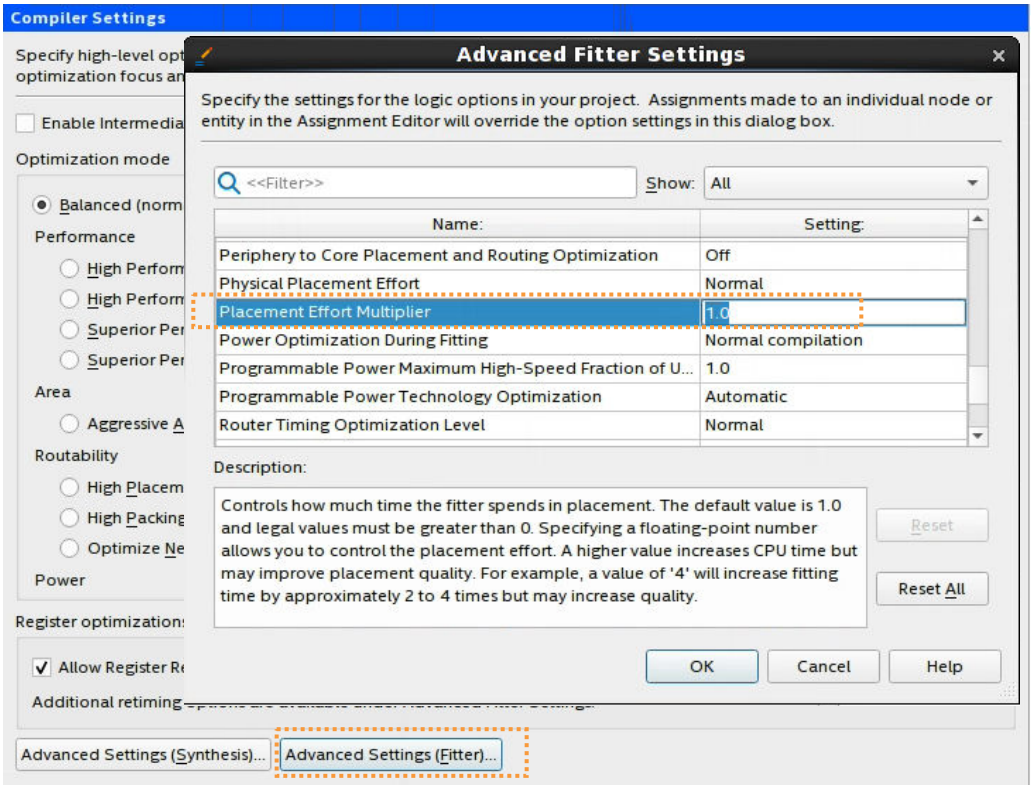
查看 Fitter 消息中编译时间的持续时间部分。如果布线比布局花费更多时间，则可能比预计中更难达到时序要求。

5.4.1.5.6. 调整布局工作量

可增加 **Assignments** > **Settings** > **Compiler Settings** > **Advanced Settings (Fitter)** > **Placement Effort Multiplier** 值，以在 Fitter 的 Place 阶段有更多编译时间和工作量。

查看和优化其他设置和 RTL 后，调整乘法器。提高该值（最大到 4），进行尝试。如果性能或编译时间并未提高，则复位到默认设置。

图 21. Placement Effort Multiplier（布局效力倍增器）



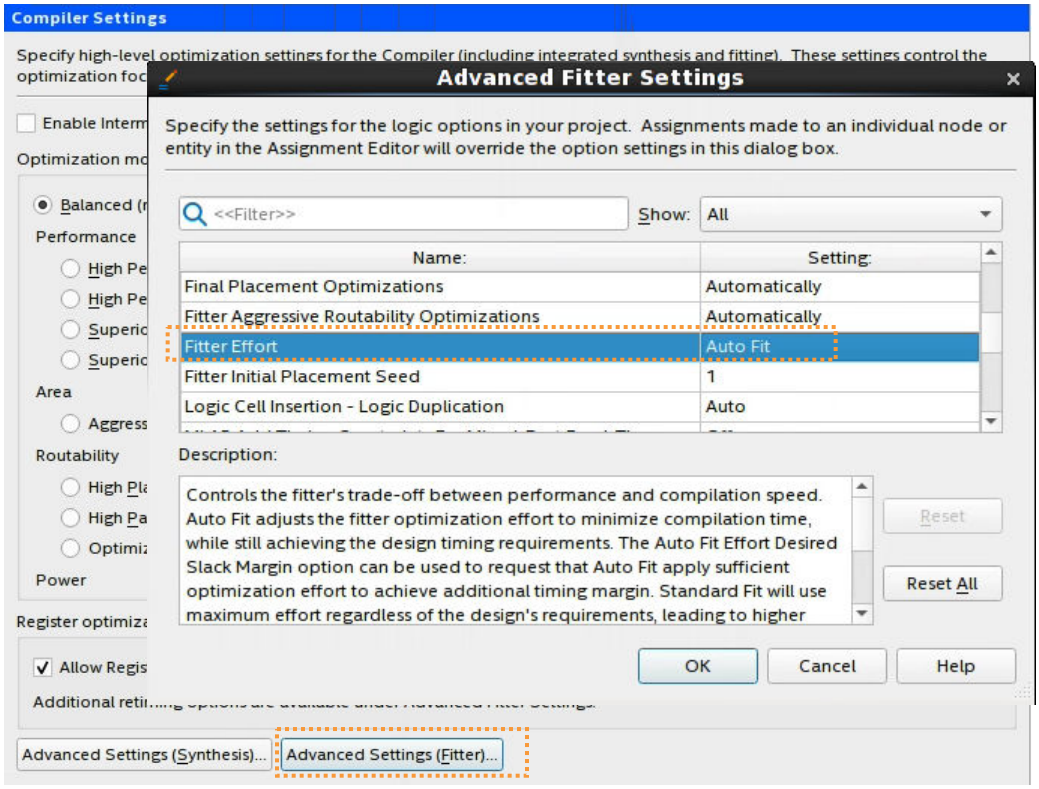
5.4.1.5.7. 调整 Fitter 工作强度

Fitter 中 **Optimization mode** 设置允许指定 Compiler 的优化工作量是否集中于性能，资源利用率，功率或编译时间。

默认情况下，Fitter **Optimization mode** 设置为 **Balanced (Normal flow)**模式，表示一旦满足时序要求则即刻减少 Fitter 工作强度和编译时间。还可选择 **Optimization mode** 以针对性能，区域布线性，功率或资源使用情况进行优化。

要进一步提高 Fitter 工作强度，可启用 **Assignments** > **Settings** > **Compiler Settings** > **Advanced Settings (Fitter)** > **Fitter Effort** 选项。默认 **Auto Fit** 设置可在满足时序要求后减少 Fitter 工作强度。**Standard Fit (highest effort)**设置使用最大工作强度且不考虑设计要求，从而导致较长编译时间和更多时序裕量。

图 22. Fitter 工作强度



5.4.1.5.8. 查看时序约束

请确保使用正确频率要求约束时钟。使用 `derive_pll_clocks` 约束可保持更新已生成时钟设置。Timing Analyzer 可用于查看 SDC 约束。例如，Task（任务）窗中的 **Diagnostic** 下，**Report Ignored Constraints** 报告显示设计中的错误名称，而导致该错误的最普遍原因是设计层级结构中的更改。使用 **Report Unconstrained Paths** 报告查找不受约束的路径。按需添加约束，以便设计优化。

5.4.1.6. 评估集群难度

可评估集群难度以助于实现时序收敛。

随时添加逻辑和重编译后都可监控集群困难。使用集群信息判断设计中固有的时序收敛难度：

- 如果设计已满，但集群难度低或中等，则拥塞的原因很可能是设计本身而非集群问题。
- 相反，设计中添加少量逻辑后出现的拥塞，反而可能由集群所导致。如果集群难度大，则无论设计大小如何，都会导致拥塞。

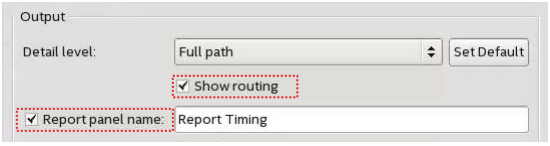
5.4.2. 查看时序路径详细信息

5.4.2.1. 查看时序路径布线

显示路径布线有助于发现异常布线延迟。

Timing Analyzer **Report Timing** 对话框中，开启 **Report panel name** 和 **Show routing** 选项，并点击 **Report Timing**。

图 23. **Report 窗格和 Show Routing 选项**



Extra Fitter Information 选项卡显示缩微布局规划图，其中突出显示路径。 **Extra Fitter Information** 选项卡不适用于 Intel Stratix 10 器件。

您还可在 Chip Planner 中查找路径以检查布线拥塞，并查看路径中的节点是否紧密相邻或相距甚远。

相关链接

在 Chip Planner 中管理路径 (第 105 页)

5.4.2.2. 全局网络缓冲器

布线缆径支持确认时序失败的全局网络缓冲器。缓冲位置根据其所驱动的网络命名。

- CLK_CTRL_Gn—用于 Global（全局）驱动程序
- CLk_CTRL_Rn—用于 Regional（区域）驱动程序

访问全局网络的缓冲器位于器件每侧的正中。对全局信号网络上内核逻辑信号进行缓冲会导致插入延迟。如果信号降级为局部布线， 则请考虑全局和非全局布线间的权衡，包括源位置，插入延迟，扇出，信号传播的距离，以及可能出现的拥塞。

5.4.2.2.1. 源位置

如果馈送全局缓冲的寄存器无法移动至更加靠近，则可考虑更改设计逻辑或布线类型。

5.4.2.2.2. 插入延迟

如果需要全局信号，则可考虑使用负边沿触发的寄存器生成信号（上图）并使用多周期建立约束（下图）从而对时序添加半个周期。

图 24. **Negative-Edge 触发的寄存器**

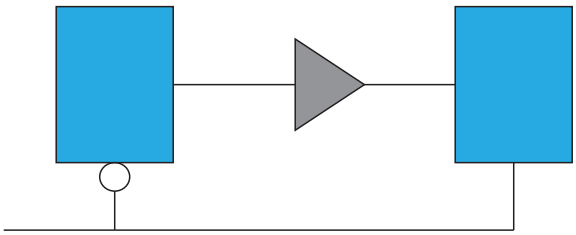
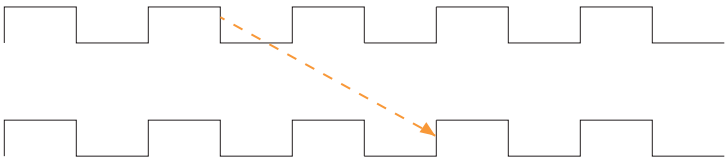


图 25. 多周期建立约束



```
set_multicycle_path -from <generating_register> -setup -end 2
```

5.4.2.2.3. 扇出

使用局部布线的高扇出节点倾向于将驱动到源节点附近的逻辑拉高。这样就可能使其他路径时序失败。复制寄存器可帮助减少高扇出路径的影响。考虑手动复制和保留这些寄存器。使用 MAX_FANOUT 约束可得到扇出节点的任意分组，而设计人员可得到更智能的扇出分组。

5.4.2.2.4. 全局网络

可使用 Global Signal 约束基于每个信号控制全局信号。例如，如果信号需要节点布线，将 Global Signal 约束设置为 **OFF**。

图 26. 全局信号约束

To	Assignment Name	Value	Enabled
reg_clk	Global Signal	Off	Yes

5.4.2.3. 复位和全局网络

复位信号通常布线于全局网络。有时，会因使用全局网络而导致恢复失败。可考虑查看生成复位和布线路径信号的寄存器布局。

5.4.2.4. Suspicious Setup

Suspicious setup（可疑建立）错误包括需要非常小或特别大的路径。

其中一个典型原因是数学精度误差。例如，10Mhz/3 = 33.33 ns/周期。三个周期中，时间为 99.999 ns 对 100.000 ns。设置最大延迟可提供正确建立关系。

失败的另一原因是设计中有意的路径错误，例如：

- 通过 FIFO 处理的异步路径，或
- 依赖数据握手的缓慢异步路径在多个时钟周期保持可用。

为避免 Fitter 必须符合不必要的限制性时序要求，请考虑添加 false 或多周期路径声明。

5.4.2.5. 逻辑深度

Timing Analyzer 路径报告中的 **Statistics** 选项卡显示路径中的逻辑级别。如果路径中时序失败且逻辑级别数较高，则考虑在设计的该部分添加流水线。

5.4.2.6. 自动移位寄存器更换

Synthesis（综合）期间，Compiler 可将移位寄存器或寄存器转换为 RAM 以节省面积。然而，转换为 RAM 后通常会降低速度。转换后的寄存器的名称包括“altshift_taps”。

- 如果路径时序错误出现在移位寄存器的开始或末尾，则可考虑禁用 **Auto Shift Register Replacement** 选项。请不要转换打算用于流水线的寄存器。
- 对于已转换为链的移位寄存器，请评估 RAM 或逻辑单元中实现的面积/速度权衡。
- 如果设计接近于满，则可将寄存器转换移位到 RAM，使非关键时钟域受益。可将设置从默认的 **AUTO** 更改为全局性 **OFF**，或基于寄存器或层次结构性关闭。

5.4.2.7. 时钟架构

为获得更佳时序结果，请将由区域时钟驱动的寄存器放置到芯片的某个象限中。可使用 Chip Planner 查看时钟区域边界。

当器件顶部的 I/Q 接口连接到器件某象限中由区域时钟驱动的逻辑时，可能出现时序失败，并且布局限制来自或到 I/O 的长路径跨象限到逻辑。

使用不同类型的时钟源驱动覆盖整个器件的逻辑-全局（logic - global），或覆盖器件一半的双区域。或者，可降低 I/O 接口的频率以适应长路径延迟。也可重新设计器件管脚说明，将所有指定 I/O 与区域时钟象限相邻放置。当寄存器位置受限时就会出现该问题，例如使用 Logic Lock 区域，时钟资源或硬块（存储器，DSP，IP）。

Timing Analyzer 时序报告中的 **Extra Fitter Information** 选项卡会通知路径中节点布局何时受限。 **Extra Fitter Information** 选项卡不适用于 Intel Stratix 10 器件。

相关链接

查看器件中可用的时钟网络 (第 99 页)

5.4.2.8. 时序收敛建议

Timing Analyzer 中的 Report Timing Closure Recommendation（报告时序收敛建议）任务分析路径并基于路径特征提供具体建议。

5.4.3. 调整和重新编译

找出可轻松解决的问题。要确定 Compiler 无法满足时序的位置，请通过大约 5 次编译执行 seed 扫描。这样就可连续显示失败路径。可考虑重新编写该设计部分的代码或重新设计该部分。

要实现时序收敛，编写良好的 RTL 可能比更改编译设置更有效。对于时序错误非常小，或者已经过性能优化并接近最终发布的设计，Seed 扫描也很有用。此外，seed 扫描还可用于评估编译设置的更改。编译结果会因为适配器算法的随机性而不同。如果编译设置的改变后产生低于平均水平的性能，则撤销该更改。

有时，一些设置或约束导致的问题甚至多于其修复的问题。对 RTL 或设计体系结构进行重大更改时，请使用默认设置，避开 Logic Lock 区域进行周期性编译，并重新评估时序失败的路径。

分区处理通常对时序收敛毫无帮助，因而必须在设计进程的初期完成。如果其阻碍了跨边界优化，使时序收敛更加苦难并增加了编译时间，则添加分区可提高逻辑利用率。

5.4.3.1. 使用分区实现时序收敛

有一种实现时序收敛的技术为：将失败路径限制在单个设计分区内，从而分区之间不存在失败路径。然后可使用增量式编译按需进行更改以更正失败路径，并仅重新编译受影响的分区。

为使用该技术：

1. 可在 Design Partition Planner 中，单击 **View** > **Show Timing Data** 加载时序数据。

失败路径上包含节点的实体在 Design Partition Planner 中显示为红色。

2. 从顶层实体窗口拖动包含失败路径的实体以对其进行提取。
   - 如果已提取的实体和顶层实体中并无失败路径，则右键点击已提取实体，然后单击 **Create Design Partition** 将该实体放置到其所属分区中。
3. 将失败路径保持在分区中，以便分区间无交叉失败路径。

   如果无法从已提取的实体中分离失败路径，则表示无跨分区边界的失败路径，因为可将实体还原至其主体中并不创建分区。
4. 找出具有最差时间裕量的分区。对于所有其他分区，保留内容并设置为 **Empty**。

   关于保留分区内容的信息，请参阅 *Intel Quartus Prime Pro Edition 用户指南：基于块的设计* 中的*基于编译流程的增量式模块*。
5. 调整分区中的逻辑并根据需要重新运行 Fitter，直到分区满足时序要求。
6. 对具有失败路径的所有其他设计分区重复此过程。

相关链接

- 查看设计连接性和层次结构 (第 110 页)
- 使用基于块的编译
  Intel Quartus Prime Pro Edition 用户指南：编译器
- 基于块的增量式编译流程
  In *Intel Quartus Prime Pro Edition 用户指南：基于块的设计*

5.5. 设计分析

初始编译确定设计是否实现成功适配并满足指定时序要求。本小节介绍如何在 Intel Quartus Prime 软件中分析设计结果。

5.5.1. 被忽略的时序约束

Intel Quartus Prime 软件忽略非法，过时和冲突的约束。

单击 **Reports** > **Report Ignored Constraints** 在 Timing Analyzer GUI 中查看被忽略约束清单或键入以下命令生成被忽略时序约束清单：

```
report_sdc -ignored -panel_name "Ignored Constraints"
```

分析任何被 Intel Quartus Prime 软件忽略的约束。如有必要，先更正约束并重新编译设计，然后才执行设计优化。

可在 Fitter 生成的 **Ignored Assignment Report** 中查看被忽略约束的列表。

相关链接

创建 I/O 要求

5.5.2. I/O 时序

Timing Analyzer 支持 Synopsys* Design Constraints (SDC)格式对设计进行约束。使用 Timing Analyzer 时序分析时，通过 `set_input_delay` 约束指定数据根据给定时钟到达出入端口的时间。对于输出端口，请使用 `set_output_delay` 命令指定数据根据给定时钟到达输出端口接收器的时间。可使用 `report_timing` Tcl 命令生成 I/O 时序报告。

未符合所需时序性能的 I/O 路径被报告为具有负时间裕量，并在 Timing Analyzer **Report** 窗口中中以红色突出显示。即使您未对 I/O 管脚应用明确的 I/O 时序约束，而 Intel Quartus Prime 时序分析软件仍会报告 **Actual** 数目，表示器件在系统中运行时，时序参数必须满足的时序数量。

相关链接

创建 I/O 要求
In *Intel Quartus Prime Pro Edition User Guide: Timing Analyzer*

5.5.3. Register-to-Register 时序分析

当时钟域上的任何 register-to-register（寄存器到寄存器）路径中无负时间裕量时，您的设计就能符合时序要求。 未满足时序要求时，可从有关失败路径的报告中发现更多详细信息。

5.5.3.1. 使用 Timing Analyzer 显示 Path Report

Timing Analyzer 生成包含所有有效 register-to-register 路径的信息。为查看全部时序总结，可双击 **Tasks** 窗格中的 **Report All Summaries**。

如果任何时钟域有失败路径（在 **Report** 窗中以红色突出显示），右键点击 **Clocks Summary** 窗中罗列的时钟名称并选择 **Report Timing** 了解详细信息。

在 **Summary of Paths** 选项卡中选择路径时，路径详情窗显示所有路径信息。**Extra Fitter Information** 选项卡提供物理器件上路径位置的直观表示。从而透露时序失败是否与距离相关，是由于源和目标节点太过靠近或相距甚远。 **Extra Fitter Information** 选项卡不适用于 Intel Stratix 10 器件。

Data Path 选项卡显示 Data Arrival Path 和 Data Required Path。可通过增量信息确定导致时序违规的最主要路径段。**Waveform** 选项卡显示时间域中的信号，并描述达到数据和所需数据间的时间裕量。

Technology Map Viewer 提供设计网表的原理图、技术映射表达，有助于评估设计中可通过减少逻辑级别而受益的区域。要在其中一个查看器中找到时序路径，右键点击时序报告中的路径，指向 **Locate Path**，并选择 **Locate in Technology Map Viewer**。还可使用 Chip Planner 详细研究路径的物理布局。

相关链接

- 何时使用 Netlist Viewer：分析设计问题 (第 14 页)
- 生成时序报告
  In *Intel Quartus Prime Pro Edition User Guide: Timing Analyzer*

5.5.3.2. 分析失败路径的提示

分析失败路径时，检查报告和波形以确定是否应用正确的约束，并根据需要添加时序异常。多周期约束通过指定的时钟周期数放宽建立或保持关系。错误的路径约束会指定时序分析期间可被忽略的路径。两种约束都支持 Fitter 在受影响的路径上充分工作。

- 专注改善显示最差时间裕量的路径。Fitter 会针对最差时间裕量的路径充分工作。如果修复了这些路径，Fitter 就可能改善设计中其他失败时序路径。
- 检查出现在多个失败路径的节点。这些节点在时序报告窗中被置顶，并附带其最小时间裕量。查找具有公共源寄存器，目标寄存器，或公共中介组合性节点的路径。某些情况下，寄存器各不相同，但却是同一总线的其中一部分。
- 时序分析报告面板中，点击 **From** 或 **To** 栏的页眉以源或目标寄存器排列路径。如果看到公共节点，则这些节点表示设计中已通过更改源代码或 Intel Quartus Prime 优化而改善的区域。仅约束其中一个路径的布局可能导致器件中公共节点被移至远处从而使得时序性能下降。

相关链接

- 在 Chip Planner 中管理路径 (第 105 页)
- 时序收敛的设计评估 (第 54 页)

5.5.3.3. 分析跨时钟域失败时钟路径的提示

分析时钟路径失败时：

- 查看路径是否跨两个时钟域。

  对于跨两个时钟域的路径，**From Clock** 和 **To Clock** 在时序报告中并不相同。

图 27. From Clock 和 To Clock 中的不同值

Setup Transfers

	From Clock	To Clock	RR Paths	FR Paths	RF Paths	FF Paths
1	clkin	clkin	21	0	0	0
2	clkin	clkout	false path	0	0	0
3	clkout	clkout	31	0	0	0

- 查看设计中包含的路径是否存在路径中涉及不同时钟的情况，即使源和目标寄存器相同。
- 查看是否需要同步分析时钟域之间的失败路径。

  将不需要同步分析的失败路径设置为错误路径。
- 对设计运行 `report_timing` 时，报告显示每个故障路径的启动时钟和锁存时钟，查看启动时钟和锁存时钟之间的关系是否切合实际，以及对已知设计所期望的内容。

  例如，路径可从上升沿开始并在下降沿结束，从而将建立关系时间减少了半个时钟周期。
- 查看 Timing Report 中的时钟偏斜：

  较大偏斜可能表示设计中存在问题，例如门控时钟，或物理布局中的问题（例如，使用局部布线而非专用时钟布线的时钟）。在已确保路径被同步分析且路径上无较大偏斜，以及约束正确后，就可以分析数据路径。这些步骤有助于微调跨时钟域的路径，以保证获得准确的时序报告。
- 查看 PLL 相移是否降低了建立要求。

  可使用 PLL 参数和设置进行调整。
- 忽略跨时钟域的路径以获得受同步逻辑保护的逻辑（例如，FIFO 或双数据同步寄存器），即使时钟相关。
- 在所有不必要路径上设置错误路径约束：

  尝试优化不必要路径可防止 Fitter 为满足时序路径上的时序要求而运行，对于设计来说至关重要。

相关链接

report_clock_transfers
Intel Quartus Prime Help 中

5.5.3.4. 分析来自/到关键路径的源和目标的提示

分析设计中失败路径时，通常有助于更加全面了解围绕路径的交互情况。

要了解关键路径上受牵制的内容，可使用以下 `report_timing` 命令。

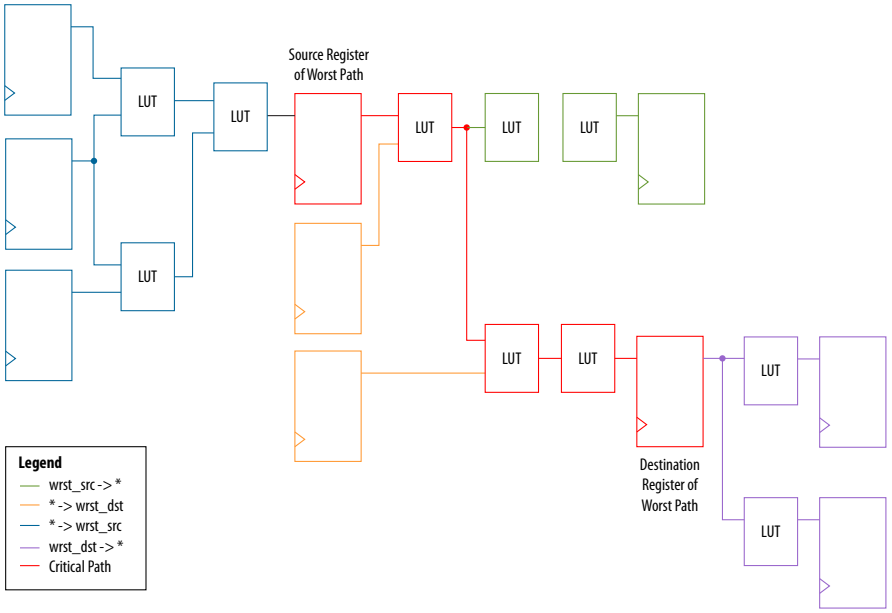
1. 工程目录中，运行 `report_timing` 命令找到关键路径中的节点。
2. 复制 `.tcl` 文件中的代码（如下），并将首两个变量替换为最差路径的 **From Node** 和 **To Node** 栏中的节点名称。脚本分析最差源和目标寄存器之间的路径。

```
set wrst_src <insert_source_of_worst_path_here>
set wrst_dst <insert_destination_of_worst_path_here>
report_timing -setup -npaths 50 -detail path_only -from $wrst_src \
-panel_name "Worst Path||wrst_src -> *"
report_timing -setup -npaths 50 -detail path_only -to $wrst_dst \
-panel_name "Worst Path||* -> wrst_dst"
report_timing -setup -npaths 50 -detail path_only -to $wrst_src \
-panel_name "Worst Path||* -> wrst_src"
report_timing -setup -npaths 50 -detail path_only -from $wrst_dst \
-panel_name "Worst Path||wrst_dst -> *"
```

3. 从 **Script** 菜单，获取 `.tcl` 文件。
4. 在生成时序的面板中，找到 Chip Planner 中时序失败路径（以红色突出显示），并查看节点和较大扇出之间的距离等相关信息。

   下图显示为报告分析的简化示例。

图 28. 时序报告



设计中的关键路径为红色。`.tcl` 脚本和图示间的关系为：

- 首两行显示关键路径的两个端点中的全部内容，并将其导向不同方向。
  - 第一个 `report_timing` 命令分析源正驱动的所有路径，以绿色显示。
  - 第二个 `report_timing` 命令分析到目标寄存器的所有路径，包括关键路径，以橙色显示。
- 最后两个 `report_timing` 命令显示端点之外的所有内容，并将其导向其他方向。

如果这些邻近路径中的任何一个具有时间裕量且靠近关键路径，则 Fitter 会将这些路径与关键路径进行平衡，以尝试实现最佳时间裕量。

5.5.3.5. 创建.tcl 脚本监控跨编译的关键路径的提示

许多设计在每次编译后都会显示相同的关键路径。而其他设计中，关键路径在不同层次结构间弹动，并随着每次编译而变化。

该行为出现在高速设计中且 register-to-register 路径大多具有极少量时序裕量。不同的布局可能导致边缘路径中时序失败。

1. 在工程目录中，创建一个脚本名称 `TQ_critical_paths.tcl`。
2. 编译后，查看关键路径，然后编写通用 `report_timing` 命令采集这些路径。

   例如，如果低级别层级结构中有多个路径失败，请添加命令，如

   ```
   report_timing -setup -npaths 50 -detail path_only \
           -to "main_system: main_system_inst|app_cpu:cpu|*" \
           -panel_name "Critical Paths||s: * -> app_cpu"
   ```

3. 如果存在特定路径，例如状态机的某个位要到其他`*count_sync*`寄存器，就可添加如下类似命令：

   ```
   report_timing -setup -npaths 50 -detail path_only \
           -from "main_system: main_system_inst|egress_count_sm:egress_inst|
   update" \
           -to "*count_sync*" -panel_name "Critical Paths||s: egress_sm|
   update -> count_sync"
   ```

4. 每次编译后在 Timing Analyzer 中执行该脚本，并在新的关键路径出现时添加新的 `report_timing` 命令。

   这样有助于您监控持续失败的路径以及仅为边际的路径，从而有效确定优先级。

5.5.3.6. 全局布线资源

全局布线资源旨在约束高扇出，低偏斜信号（如，时钟）且不消耗常规布线资源。基于器件的不同，这些资源可遍布整个芯片或较小部分，如象限。 Intel Quartus Prime 软件尝试自动为全局布线资源约束信号，但也可手动进行更合适的约束。

关于可用全局布线资源的数量和类型，请参阅相关器件手册。

查看设计中全局信号利用率，可确保全局布线资源上放置了正确的信号。Compilation Report 中，打开 Fitter 报告并单击 **Resource Section**。分析 Global & Other Fast Signals 和 Non-Global High Fan-out Signals 报告以确定是否需要任何更改。

可通过将高扇出信号放置到全局布线资源上来减少偏斜。反之，可将全局布线资源上的低扇出移除来减少插入延迟。这样就可提高时钟使能时序并控制信号恢复/移除时序，但会增加时钟偏斜。使用 Assignment Editor 中的 **Global Signal** 设置控制全局布线资源。

5.6. 时序优化

如果设计未能满足时序要求，则请使用以下指导。

5.6.1. 显示失败路径的时序收敛建议

使用 **Timing Closure Recommendations** 报告获得关于设计失败路径的特定建议以及可能修复失败路径的各种更改。 对于 Intel Stratix 10 器件，请使用 Fast Forward Timing Closure Recommendations 报告。

1. Timing Analyzerde 的 **Tasks** 窗格中，选择 **Report Timing Closure Recommendations** 任务，打开 **Report Timing Closure Recommendations** 对话框。
2. 基于时钟域选择路径，按路径上的节点进行过滤，然后选择要分析的路径数量。
3. Timing Analyzer 中运行完 **Report Timing Closure Recommendations** 任务后，从 Timing Analyzer GUI 的 **Report** 窗格检查 **Report Timing Closure Recommendations** 文件夹中的报告。每个建议以星号（*）标注。带有多个星号的建议更可能帮助您设计中的时序收敛。

报告为您提供每个已分析路径中最有可能的失败原因，并显示有助于修复失败路径的建议。

报告分为几个部分，具体取决于从设计中发现的问题类型，例如加大时钟偏斜，受限优化，不均衡逻辑，跳过优化，寄存器之间的编码样式存在过多逻辑级别，或特定于您的工程的区域或分区约束。

要详细分析关键路径，请在指定路径上运行 `report_timing` 命令。在 **Path** 报告窗格的 **Extra Fitter Information** 选项卡中，可查看与适配相关详细信息，有助于更直观看待问题。 **Extra Fitter Information** 选项卡不适用于 Intel Stratix 10 器件。

相关链接

- Fast Forward 时序收敛建议 (第 87 页)
- 报告时序收敛建议对话框
  Intel Quartus Prime Help 中

5.6.2. 时序优化向导

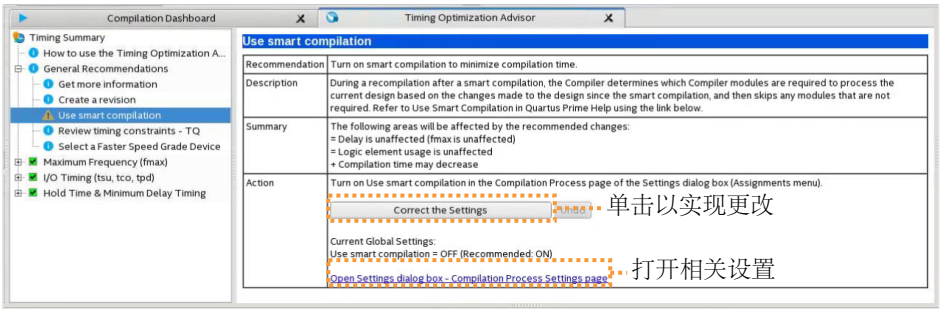
Timing Analyzer **Report Timing Closure Recommendations** 任务提供修复失败路径的特定建议的同时，Timing Optimization Advisor 还提供更多改善设计时序性能的常规建议。

Timing Optimization Advisor 指导针对设计优化的设置以满足时序要求。要运行 Timing Optimization Advisor 可单击 **Tools** > **Advisors** > **Timing Optimization Advisor**。该向导说明本节中提出的多个建议。

编译后开启 Timing Optimization Advisor，可找到提高设计中时序性能的建议。如果各指导中的建议相互矛盾，则请评估各选项并选择最适合给定要求的设置。

本实例显示了 Timing Optimization Advisor，在通过编译设计来满足其频率要求后，还需要更改设置提高时序性能。

图 29. 时序优化向导



在 Timing Optimization Advisor 中展开其中一个类别，如 **Maximum Frequency (fmax)**或 **I/O Timing (tsu, tco, tpd)**时，建议会分阶段呈现。这些阶段显示应用建议设置的顺序。

首个阶段包含最容易更改的选项，通过最轻度的更改来优化设计，并对编译时间的影响最小。

图标标示当前工程中是否进行了每个推荐设置。图示中，Stage 1 的建议清单中的勾选图标表示已实现的建议。警告图标表示本此编译中未采纳的建议。信息图标表示常规建议。对于这些实体，向导中并不报告是否采纳这些建议，但会解释如何实现更佳性能。关于提供每个图标详细信息的图例，请参阅 Timing Optimization Advisor（时序优化向导）中的“How to use”页。

每个建议提供前往 Intel Quartus Prime GUI 中正确位置的链接，可在此更改设置。例如，可考虑 **Settings** 对话框中的 **Synthesis Netlist Optimizations** 页或 Assignment Editor 中的 **Global Signals category**。此方法可最大程度控制需要的设置并帮助了解软件中的设置。适时还可使用 **Correct the Settings** 按钮自动更改全局设置。

对于 Timing Optimization Advisor 中的某些条目，可使用该按钮进一步分析设计并查看更多信息。本向导提供包含设计中各时钟的表格，以显示是否已对时钟进行时序约束。

5.6.3. 可选 Fitter 设置

本小节仅关注可选时序优化 Fitter 设置，即 **Optimize Hold Timing**、**Optimize Multi-Corner Timing** 和 **Fitter Aggressive Routability Optimization**。

警告: 不同设计的最佳优化设置各不相同。对某设计最有效的一组设置并不一定对另一设计产生最好结果。

相关链接

高级 Fitter 设置对话框
Intel Quartus Prime Help 中

5.6.3.1. 优化保持时序

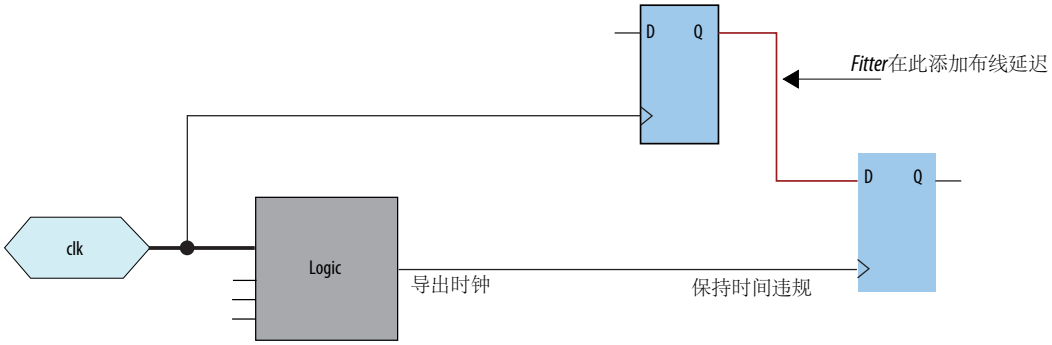
Optimize Hold Timing 选项指示 Intel Quartus Prime 软件优化最小延迟时序约束。

从 **Advanced Fitter Settings** 对话框中开启 **Optimize Hold Timing** 后， Intel Quartus Prime 软件会增加路径延迟，以确保您的设计满足最小延迟要求。如果选择 **I/O Paths** 和 **Minimum TPD Paths**，则 Fitter 运行以符合如下条件：

- 从器件输入管脚到寄存器的保持时间(t_H)
- 从 I/O 管脚到 I/O 寄存器或从 I/O 寄存器到 I/O 管脚的最小延迟
- 从寄存器到输出管脚的最小时钟输出时间(t_{CO})

如果选择 **All Paths**，Fitter 还可运行至满足从寄存器到寄存器的保持要求（如图中蓝色突出显示），其中由逻辑驱动产生而得的时钟会导致另一寄存器上出现保持时间问题。

图 30. 优化保持时间选项修复内部保持时间违规



然而，如果您设计中寄存器之间仍然存在内部保持时间违规，则可通过例化 LCELL 原语，或更改设计来手动添加延迟，例如使用时钟使能信号而非导出或门控时钟。

相关链接

建议的设计实践
In *Intel Quartus Prime Pro Edition User Guide: Design Recommendations*

5.6.3.2. Fitter 主动布通性优化

Fitter Aggressive Routability Optimizations（主动布通性优化）逻辑选项支持可指定 Fitter 主动优化的布通性。执行主动布通性优化可能减慢设计速度，但也会减少布线线缆使用和布线时间。

如果布线资源导致不适配错误，且您希望减少布线线缆的使用，则该选项大有用处。

本表格罗列了 **Fitter Aggressive Routability Optimizations** 逻辑选项的设置内容。

表 13. Fitter 主动布通性优化逻辑选项设置

设置	说明
Always	Fitter 时钟执行主动布通性优化。如果将 **Fitter Aggressive Routability Optimizations** 逻辑选项设置为 **Always**，会降低线缆利用率从而可能影响设计性能。
Never	Fitter 从不执行主动布通性优化。如果盖上时序比减少线缆使用更重要，则将该选项设置为 **Automatically** 或 **Never**。
Automatically	Fitter 基于设计的布通性和时序要求，自动执行主动布通性优化。如果改善时序比降低线缆使用率更重要，则将该选项设置为 **Automatically** 或 **Never**。

5.6.4. I/O 时序优化技术

该设计阶段侧重于 I/O 时序，包括建立延迟（t_{SU}），保持时间（t_H），和时钟-输出（t_{CO} 参数。

开始 I/O 时序优化之前，请确保：

- 设计中的约束按照 *Design Optimization Overview*（设计优化概述）章节中 *Initial Compilation: Required Settings*（初始编译：必要设置）部分的建议。
- 资源利用率符合要求。

注意: 需先完成该阶段再开始寄存器-寄存器（register-to-register）时序优化阶段。更改 I/O 路径会影响内部 register-to-register 时序。

改善建立和时钟到输出时间的技术总结

本表格列出应用各建议技术时的顺序以减少 t_{SU} 和 t_{CO} 时间。减少 t_{SU} 时间会增加保持（t_H）时间。

注意: 验证可用于每个器件系列的选项

表 14. 改善建立和时钟到输出时间

顺序	技术	影响 t_{SU}	影响 t_{CO}
1	验证是否对失败的 I/O 设置了正确的约束（请参阅 *初始编译：必需设置*）	Yes	Yes
2	对 I/O 使用时间驱动的编译（请参阅 *快速输入，输出和输出使能寄存器*）	Yes	Yes
3	使用快速输入寄存器（请参阅 *可编程延迟*）	Yes	N/A
4	使用快速输出寄存器，快速输出使能寄存器和快速 OCT 寄存器（请参阅 *可编程延迟*）	N/A	Yes
5	减小 **Input Delay from Pin to Input Register** 的值或设置 **Decrease Input Delay to Input Register = ON**	Yes	N/A
6	减小 **Input Delay from Pin to Internal Cells** 的值或设置 **Decrease Input Delay to Internal Cells = ON**	Yes	N/A
7	减小 **Delay from Output Register to Output Pin** 的值或设置 **Increase Delay to Output Pin = OFF**（请参阅 *快速输入，输出和输出使能寄存器*）	N/A	Yes
8	增加 **Input Delay from Dual-Purpose Clock Pin to Fan-Out Destinations** 的值（请参阅 *快速输入，输出和输出使能寄存器*）	Yes	N/A
9	**Use PLLs to shift clock edges**	Yes	Yes
10	增加 **Delay to output enable pin** 的值或设置 **Increase delay to output enable pin**（请参阅 *使用 PLL 移位时钟沿*）	N/A	Yes

优化时序逻辑选项的 IOC 寄存器布局 (第 75 页)
快速输入，输出和输出使能寄存器 (第 75 页)
可编程延迟 (第 75 页)
使用 PLL 移位时钟沿 (第 76 页)
使用 Fast Regional Clock Network 和 Regional Clocks Network (第 76 页)
脊柱时钟限制 (第 77 页)

相关链接

初始编译的必需设置 (第 6 页)

5.6.4.1. 优化时序逻辑选项的 IOC 寄存器布局

该选项将寄存器移入 I/O 单元以从而满足 t_{SU} 或 t_{CO} 约束，必要时复制寄存器（如一个寄存器扇出对应多个输出位置的情况）。该选项默认开启，并且是全局设置。

Optimize IOC Register Placement for Timing 逻辑选项仅影响具有 t_{SU} 或 t_{CO} 要求的管脚。可能仅当寄存器直接馈送管脚或由管脚直接馈送时才使用 I/O 寄存器。因此，该逻辑选项不影响如下特性的寄存器：

注意: 要优化具有这些特征的寄存器，请使用其他 Intel Quartus Prime Fitter 优化。

- 寄存器和管脚之间具有组合逻辑
- 作为进位链的一部分
- 具有覆盖性位置约束
- 使用异步加载端口且值不为 1（在该端口可用的器件系列中）

相关链接

优化时序逻辑选项的 IOC 寄存器布局
Intel Quartus Prime Help 中

5.6.4.2. 快速输入，输出和输出使能寄存器

通过 Assignment Editor 进行快速 I/O 约束，可手动将单个寄存器放入 I/O 单元中。默认情况下，按照正确的时序约束，Fitter 会将寄存器放置到正确的 I/O 单元或内核中，以满足性能要求。

如果快速 I/O 设置为开启，则寄存器始终放置于 I/O 单元中。如果快速 I/O 设置为关闭， 寄存器从不被放置到 I/O 单元。即使 **Optimize IOC Register Placement for Timing** 选项开启，也是如此。如果不存在快速 I/O 约束，在开启 **Optimize IOC Register Placement for Timing** 选项时， Intel Quartus Prime 软件决定是否将寄存器放入 I/O 单元中。

还可使用 4 个快速 I/O 选项（**Fast Input Register**，**Fast Output Register**，**Fast Output Enable Register** 和 **Fast OCT Register**）覆盖 Logic Lock 区域中寄存器的位置，并强制其进入 I/O 单元。如果对馈送多个管脚的寄存器应用该约束，则 Fitter 复制该寄存器并将其放入所有相关 I/O 元件中。

更多关于 **Fast Input Register** 选项，**Fast Output Register** 选项，**Fast Output Enable Register** 选项和 **Fast OCT (on-chip termination) Register** 选项的信息，请参阅 Intel Quartus Prime Help。

相关链接

- 快速输入寄存器逻辑选项
- 快速输出寄存器逻辑选项
- 快速输出使能寄存器逻辑选项
- 快速 OCT 寄存器逻辑选项

5.6.4.3. 可编程延迟

可使用各种可编程延迟选项最小化 t_{SU} 和 t_{CO} 时间。可编程延迟是旨在编译工程后才可使用的高级选项，检查 I/O 时序，并确定是否合格。

软件自动调整适用的可编程延迟，以帮助满足时序要求。关于这些选项的效果的详细信息，请参阅器件系列手册或数据表。

完成可编程延迟约束并编译设计后，可在 Compilation Report 的 **Delay Chain Summary** 部分查看每个延迟链和每个 I/O 管脚的已实现延迟值。

可使用 Assignment Editor 为支持的节点约束可编程延迟选项。还可通过 Chip Planner 和 Resource Property Editor 查看并修改目标器件的延迟链设置。执行完整编译后，使用 Resource Property Editor 进行更改时不需要重新编译整个设计；可直接将更改保存到网表。由于是直接对网表进行更改，重新编译时就不再自动进行更改。更改管理功能支持在后续编译中再次应用所更改内容。

尽管新近器件中的可编程延迟为用户可控，但 Intel 建议仅高级用户使用。然而 Intel Quartus Prime 软件可能在 Fitter 阶段内部使用可编程延迟。

关于 Intel 器件可用的可编程延迟逻辑选项，请参阅 Intel Quartus Prime Help 主题：

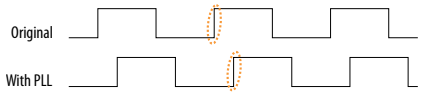
相关链接

- 从管脚到输入寄存器的输入延迟逻辑选项
- 从管脚到内部单元的输入延迟逻辑选项
- 输出使能管脚延迟逻辑选项
- 从输出寄存器到输出管脚逻辑延迟的逻辑选项
- 从复用时钟管脚到扇出目的地的输入延迟的逻辑选项
  Intel Quartus Prime Help 中

5.6.4.4. 使用 PLL 移位时钟沿

使用 PLL 通常可自动提高 I/O 时序性能。如果仍未满足时序要求，则大多数器件允许 PLL 输出相移来更改 I/O 时序。向后移位时钟会牺牲 t_{SU} 来提供较好的 t_H，而时钟向前移位则牺牲 t_H 但提供较好的 t_{SU}。仅可在提供具有相位移动选项的 PLL 的器件中使用该技术。

图 31. 向前移位时钟沿牺牲 t_H 来提高 t_{SU}



通过使用名为 **Input Delay from Dual Purpose Clock Pin to Fan-Out Destinations** 的可编程延迟，可在某些器件中实现相同类型的效果。

5.6.4.5. 使用 Fast Regional Clock Network 和 Regional Clocks Network

区域时钟（regional clock）为单个象限中的逻辑提供最低时钟延迟和偏斜。一般而言，快速区域时钟（fast regional clock）到 I/O 单元的延迟小于区域和全局时钟，并且用于高扇出控制信号。在这些低偏斜和低延迟时钟网络上放置时钟可提供更好的 t_{CO} 性能。

Intel 器件具有各种层次时钟结构。这些包括专用全局时钟网络，区域时钟网络，快速区域时钟网络和外设时钟网络。不同 Intel 器件系列之间的可用资源各不相同。

有关每个目标器件中可用的时钟资源数量，请参阅相关器件手册。

5.6.4.6. 脊柱时钟限制

在高时钟布线要求的工程中， Intel Quartus Prime 软件中的限制会导致脊柱时钟（spine clock）错误。这些错误通常出现在使用多个存储器接口和高速串行接口（HSSI）通道的设计中，尤其是 PMA Direct 模式。

全局时钟网络，区域时钟网络和外设时钟网络具有其他级别的时钟网络，称为 spine clock。Spine clock 将最后行和列时钟驱动到器寄存器；因此，芯片中的每个时钟通过 spine clock 到达每个芯片。Spine clock 为用户非直接可控。

要减少 spine clock 有关的错误，请对设计进行约束以更好使用您的区域时钟资源：

- 如果您的设计中未使用 Logic Lock 区域，或者如果 Logic Lock 区域未与您的时钟域边界对齐，则请创建其他 Logic Lock 区域并进一步约束您的逻辑。
- 如果 Periphery 功能忽略 Logic Lock 区域约束，很可能由于全局时钟资源自动检测过程未正常运转。为确保全局时钟资源自动检测过程使用正确的位置，可为使用这些外设功能的 I/O 约束指定管脚。
- 默认情况下，一些 Intel FPGA IP 功能应用值为“双区域时钟”的全局信号约束。如果将您的逻辑约束到区域性时钟区域并将全局时钟设置为 **Regional** 而非 **Dual-Regional**，则可减少时钟资源争用。

相关链接

- 查看器件中可用的时钟网络 (第 99 页)
- 层次设置 (第 97 页)
- Report Spine Clock Utilization dialog box (Chip Planner)
  Intel Quartus Prime Help 中

5.6.5. Register-to-Register 时序优化技术

设计优化的下一阶段旨在提高 register-to-register (f_{MAX})时序。如果编译后设计无法符合时序要求，则以下部分提供可用选项。

编码风格比更改设置更大程度上影响设计性能。始终评估代码并确保使用同步设计实践。

注意: 在 Timing Analyzer 的上下文中，register-to-register 时序优化与最大化设计中时钟域上的时间裕量效果相同。本小节中的技术可改善设计中不同时序路径上的时间裕量。

执行设计优化之前，请先了解设计结构以及各种技术在不同逻辑类型中的效果。无法使逻辑结构受益的技术甚至会降低设计性能。

相关链接

建议的设计实践
In *Intel Quartus Prime Pro Edition User Guide: Design Recommendations*

5.6.5.1. 优化源代码

多数情况下，优化设计源代码能非常显著提高设计性能。事实上，优化您的源代码通常是提高设计结果质量最有效的技术，往往是比使用 Logic Lock 或位置约束更好的选择。

编码时请留意实现设计中的逻辑所需要的逻辑级别的数量。寄存器之间过多的逻辑级别可能导致关键路径时序失败。请尝试重构设计以使用流水线或更有效的编码技术。另外，尝试限制源代码中的高扇出信号。如有可能，请复制并流水线控制信号。务必确保复制寄存器由保留属性保护，从而避免综合期间进行合并。

如果设计中的关键路径涉及存储器或 DSP 功能，请检查设计中说明存储器或功能的编码块是否未经推断且未布局到专用逻辑中。可修改源代码使得这些功能放入高性能专用存储器或目标器件的资源中。使用 RAM/DSP 块时，使能可选输入和输出寄存器。

请确保您的状态机被识别为状态逻辑且在综合工具中被适当优化。已识别出的状态机通常会被优化，否则被当作一般逻辑处理。在 Intel Quartus Prime 软件中，可从 Compilation Report 的 **Analysis & Synthesis** 下查看 State Machine 报告。该报告提供包括编译期间识别出的每个状态机的状态编码等各种详细信息。如果无法识别您的状态机，则可能必须更改源代码才能识别。

相关链接

AN 584：高级 FPGA 设计的时序收敛方法

5.6.5.2. 改善 Register-to-Register 时序

对于提高时序裕量（slack）或改善 register-to-register 时序的选项和设置的选择取决于设计中的失败路径。要实现最接近性能要求的结果，请应用如下技术并在每个步骤后编译设计：

1. 确保完成时序约束及其正确性。请参阅*设计优化概述*章节中的*初始化编译：必要设置*部分获得详细信息。
2. 查看初始化编译过程中的所有警告消息，并查看被忽略的时序约束。
3. 应用网表综合优化选项。
4. 要优化速度，可应用以下综合选项：
   - Optimize Synthesis for Speed, Not Area（优化关于速度而非面积的综合）
   - Flatten the Hierarchy During Synthesis（综合期间展开层级结构）
   - Set the Synthesis Effort to High（Synthesis Effort 设置为 High）
   - Prevent Shift Register Inference（防止 Shift Register 推断）
   - Use Other Synthesis Options Available in Your Synthesis Tool（使用 Synthesis Tool 中的其他可用综合选项）
5. 要优化性能，可开启 Advanced Physical Optimization
6. 尝试各种 Fitter seed。如果只有少量路径因少许负时间而导致时序失败，则可尝试使用其他 seed 找出能够符合 Fitter seed 噪音要求的约束。

   注意: 如果大量路径出现故障，或因长时间裕量造成的路径失败， 则忽略此步骤。
7. 为控制布局，需进行 Logic Lock 约束。
8. 修改设计源代码以修复设计中因大量时间裕量而无法满足时序要求的设计区域。
9. 进行位置约束，或万不得已，通过反向标注设计执行手动布局。

   可使用 Design Space Explorer II (DSE)自动运行不同设置下的各种编译过程。

   如果这些技术都无法满足性能要求，则可能需要修改其他设计源代码。

相关链接

- Design Space Explorer II (第 10 页)
- 初始编译的必需设置 (第 6 页)

5.6.5.3. 物理综合优化

Intel Quartus Prime 软件提供有助于提高设计性能的物理性综合优化，且无需考虑综合工具因素。可在综合或适配期间应用物理中和优化。

在 Intel Quartus Prime 编译的综合阶段过程中，既可在其他 EDA 综合工具的输出上运行物理性优化，也可将其作为综合过程中的一个中间步骤进行。这些优化将修改综合网表以提高面积或速度，具体取决于您选择的技术和效力级别。

要查看并修改综合网表优化选项，请点击 **Assignments > Settings > Compiler Settings > Advanced Settings (Fitter)**。

如果使用第三方 EDA 综合工具并需要确定是否 Intel Quartus Prime 软件可通过重新映射电路来提高性能，可使用 **Perform WYSIWYG Primitive Resynthesis** 选项。该选项指示 Intel Quartus Prime 软件取消原子网表到逻辑门控的 LE 映射，然后将门控映射回 Intel 特定原语。Intel 特定原语使能 Fitter 使用指定体系结构技术重新映射电路。

Intel Quartus Prime 技术映射器根据 **Optimization Technique** 选项的设置优化设计以实现最大速度性能，最小化使用面积，或平衡高性能和最小化逻辑使用。将该选项设置为 **Speed** 或 **Balanced**。

Intel Quartus Prime 编译的 Fitter 阶段中，物理综合优化在网表中进行指定布局更改，以提高特定 Intel 器件的速度性能结果。

相关链接

- 执行 WYSIWYG 原语再综合逻辑选项
- 优化技术逻辑选项
  Intel Quartus Prime Help 中

5.6.5.4. 关闭 Extra-Effort Power 优化设置

如果功率优化设置为 **Extra Effort**，则可能影响您的设计性能。如果时序性能比功率更重要，则请将电源优化设置为 **Normal**。

相关链接

- 功耗优化
- 功耗优化逻辑选项
  Intel Quartus Prime Help 中

5.6.5.5. 优化关于速度而非面积的综合

设计性能因编码样式，综合工具的使用和综合时指定的选项而各不相同。如果出现大量路径失败，或由于较大时间裕量和过多逻辑级别导致特定路径失败，就可尝试更改综合选项。

确定综合工具中默认的优化目标，并相应设置器件和时序约束。例如，如果未指定目标频率，某些综合工具会针对面积进行优化。

可使用 Assignment Editor 为设计中的特定模块指定逻辑选项，与此同时，对于特定器件系列中对面积和速度的最佳权衡，可保持默认 **Optimization Technique** 设置为 **Balanced**，或者如果面积是重要考虑因素，则将上述选项设置为 **Area**。还可使用 Assignment Editor 中的 **Speed Optimization Technique for Clock Domains** 选项指定针对速度而优化的特定时钟域中或相互之间的全部组合逻辑。

要通过按钮编译获得最佳性能，请按照如下部分中的建议进行其他综合设置。可使用 DSE II 通过各个 Intel Quartus Prime 综合选项进行设计优化实验以获得最佳性能。

相关链接

优化技术逻辑选项
Intel Quartus Prime Help 中

5.6.5.6. 综合期间展开层级结构

综合工具通常允许保留层次边界，以利于验证或其他用途。然而，当综合工具跨层次边界进行优化时，通常出现最佳优化结果，由于这样操作往往能允许综合工具执行最大程度的逻辑最小化，进而提高性能。尽可能展开设计层次以实现最佳效果。

5.6.5.7. Synthesis Effort 设置为 High

Synthesis 工具提供不同综合效力水平以权衡编译时间和综合结果。适用时将综合效力设置为 **high** 获得最佳结果。

5.6.5.8. 复制用于扇出控制的寄存器

通常，由于受到非直接涉及传输错误的信号的影响，而导致时序错误。当具有较高扇出的非关键网络（off-critical nets），在跨越较大距离时扰乱其周围其他路径优化，往往出现这种情况。

复制这些具有全局影响力的信号的源，有助于将它们分散在多个跃点，甚至分散于多个时钟周期上，从而更专注于被你传输。

例如，以寄存器树形式复制高扇出信号，可将信号分散于几个时钟周期。随着信号沿着寄存器树进行处理，会逐渐将更多信号馈送到原始寄存器的本地副本中，从而任何单个寄存器的目的地都适当定位，并且对寄存器优化的影响最小。该优化的关键是确定复制项之间对原始信号扇出的分配。如果任何单个寄存器需要长距离运行，则寄存器树的优势被移除。

可手动创建寄存器树，并利用关于在整个设计中最佳分散信号的系统级知识，在 RTL 中对端点进行分组，其过程可能会耗时并产生广泛影响。有关手动创建寄存器树的更多信息，请参阅手动寄存器复制 (第 80 页)。

可通过下列任一方法自动创建寄存器树：

- 估算物理性接近度
- 层次性接近度

每种方法以其自有方式确定需创建的复制项数量以及如何在复制项之间分配扇出。

5.6.5.8.1. 手动寄存器复制

综合工具支持指定寄存器最大扇出的选项或属性。使用 Intel Quartus Prime 综合时，可设置 Assignment Editor 中的 **Maximum Fan-Out** 逻辑选项控制节点的目标的数量，使得扇出计数不超过指定值。还可在 HDL 代码中使用 `maxfan` 属性。软件根据需要复制节点以实现指定的最大扇出。

使用 **Maximum Fan-Out** 约束的逻辑复制通常会提高资源利用率，并可能潜在提高编译时间，具体取决于所选器件中的布局以及资源使用总体情况。

由 **Maximum Fan-Out** 约束产生的时序性能提高因设计而异。这是由于使用 **Maximum Fan-Out** 约束时，Fitter 复制源逻辑以限制扇出，但并不控制每个复制源驱动的目标。因此，复制的源逻辑可以是位于器件周围的驱动逻辑。为避免这种情况，可使用 **Manual Logic Duplication** 逻辑选项。

如果使用 **Maximum Fan-Out** 约束，测量“在使用”和“不使用”这些约束时的性能以评估这些约束是否提供期望的时序性能提高。仅在获得改进结果时才使用该约束。

可在 Intel Quartus Prime 软件中手动复制寄存器，无论使用何种综合工具。要复制寄存器，请通过 Assignment Editor 针对寄存器应用 **Manual Logic Duplication** 逻辑选项。

注意: 某些 Fitter 优化可能引起改善时序的 **Maximum Fan-Out** 约束出现少量违规。

5.6.5.8.2. 寄存器自动复制：估算物理性接近度

DUPLICATE_REGISTER 约束有助于利用估算的物理性接近度信息指导复制创建及其扇出约束。

```
set_instance_assignment -name DUPLICATE_REGISTER -to <register_name>
<num_duplicates>
```

其中，

- `register_name` 是要复制的寄存器。从链中创建寄存器树时，请为链中的每个寄存器创建一个唯一约束。如果将 DUPLICATE_REGISTER 约束应用于链中相互驱动的寄存器，则会以适当顺序处理该约束。
- `num_duplicates` 是复制寄存器需要创建的数量。（包括原始寄存器）。如果原始信号具有 M 扇出，则复制项的平均扇出为 M/N，各个复制项的扇出率可能或多于或少于该值，具体取决于算法。

在 Fitter 阶段处理 DUPLICATE_REGISTER 约束。有必要创建复制项并根据物理性接近度早期估算约束复制项之间的扇出，以最大程度优化设计中“发布复制”所花费的时间。但从而导致细粒度的约束决策不精确。当复制项的数量较小（小于 100）时，且所创建组的粒度足够时，最好使用 DUPLICATE_REGISTER 约束，以允许创建复制项后进行优化时保持灵活性。

Fit 报告的 **Fitter Duplication Summary** 面板中详细说明了 Intel Quartus Prime Pro Edition 选择的 DUPLICATE_REGISTER 约束。其还总结了任何超过 1000 个扇出的已寄存信号，便于以后可能作为 DUPLICATE_REGISTER 约束的合理备选对象。

重要:

- 将 PHYSICAL_SYNTHESIS 设置为 OFF 禁用 DUPLICATE_REGISTER。
- 与其他物理综合优化不同，DUPLICATE_REGISTER 约束的确允许复制提供以不清楚的寄存器和具有位置约束的寄存器。
- DUPLICATE_REGISTER 约束不处理具有如下任一条件的寄存器：
  - 寄存器驱动全局信号或时钟信号。
  - 寄存器具有时序约束或应用了其他例外情况。
  - 寄存器具有 `preserve` 属性或 PRESERVE_REGISTER 约束。
  - 寄存器被标记为 `don't touch`。
  - 寄存器驱动或由其他分区驱动。

5.6.5.8.3. 寄存器自动复制：层次接近度

利用设计层次信息指导复制项的创建以及通过 DUPLICATE_HIERARCHY_DEPTH 约束使能其扇出约束。

```
set_instance_assignment -name DUPLICATE_HIERARCHY_DEPTH -to <register_name>
<num_levels>
```

其中，

- 在扇出多个层次结构的链中，register_name 是该链中最后一个寄存器。要创建寄存器树，请确保节点后有足够的简单寄存器，且这些简单寄存器将自动拉入该树中。
- num_levels 对应链中存在的用于向下复制层次结构的寄存器数量的上限。

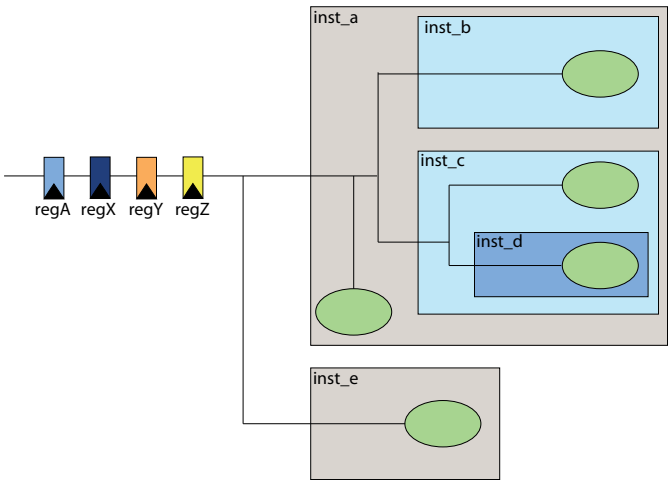
在综合阶段处理 DUPLICATE_HIERARCHY_DEPTH 约束。高扇出信号通常会穿过寄存器流水线并进入模块的自层次结构。例如，系统范围的复位可在几个时钟周期内传播，并被驱动到整个设计中的多个模块。某些情况下，利用该子层次结构有益于推断将要创建的寄存器树的结构，从而在相似层次结构内的端点被约束到同一信号副本设计中，且设计中的层次分支显示放置寄存器树分支的位置。

重要: 链中寄存器必须满足以下所有条件，才能包含与复制项中：

- 寄存器必须由另一寄存器馈给。
- 寄存器不得由组合逻辑馈给。
- 寄存器不得是同步程序链的一部分。
- 寄存器不得具有任何辅助信号。
- 寄存器不可有 preserve 属性或 PRESERVE_REGISTER 约束。
- 除了链中最后一个寄存器外，链中其他所有寄存器必须仅有一个扇出。

参考以下实例，该网表具有寄存器链及其驱动的端点层次结构。DUPLICATE_HIERARCHY_DEPTH 约束复制各个层次中的流水线寄存器，如图 35 (第 84 页)中所示。

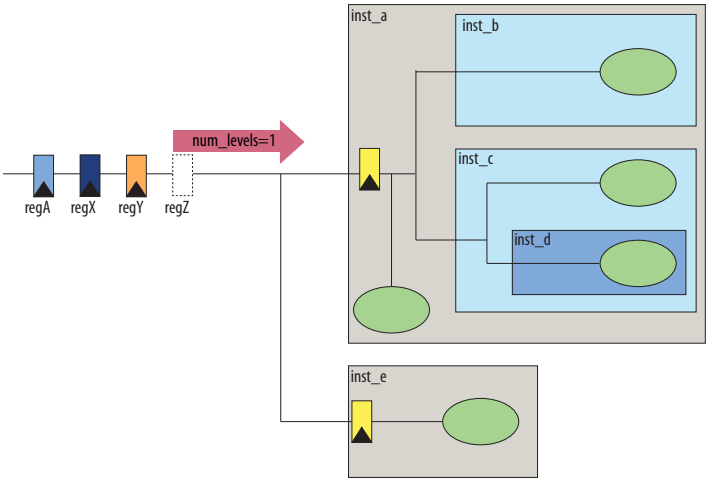
图 32. 原图显示 4 个流水线寄存器连接到多个层次结构



该情况下，regZ 是合适的约束对象，因为它是 4 个寄存器链中的端点。本实例（regZ，regY 和 regX）中最多有 3 个备选复制对象，因此约束值可为 1 到 3 之间的任意值。regA 未被拉入层次结构，以保持其之前的路径时序和优化。当某信号必须被复制到 100 个以上，且链下子层足的深度和重要度足以指导所需寄存器树的结构时，最好使用 DUPLICATE_HIERARCHY_DEPTH 约束。

图 33. 将 regZ 复制到 Hierarchy Level One 后的网表

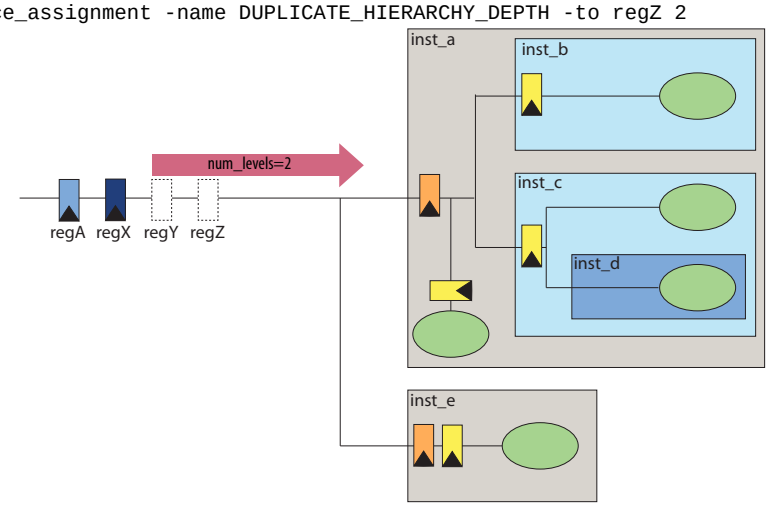
```
set_instance_assignment -name DUPLICATE_HIERARCHY_DEPTH -to regZ 1
```



num_levels 设置为 1 时，仅将 regZ 从链中拉出，并将一个层次级别推低至其扇出树中。

图 34. 将 regZ 复制到 Hierarchy Level Two 后的网表

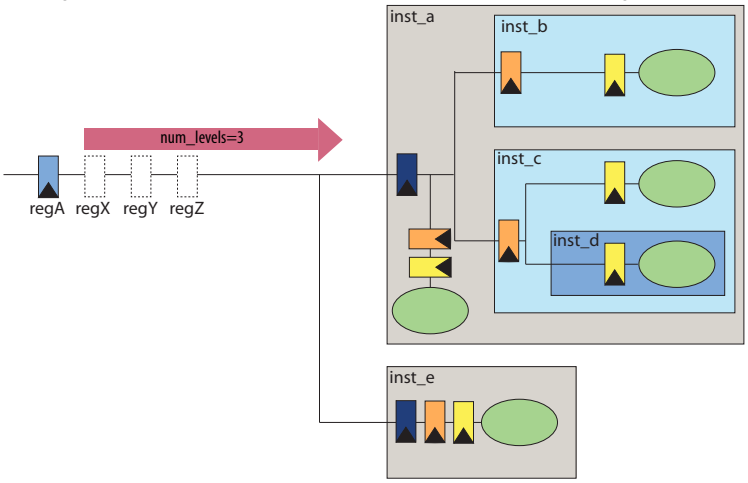
```
set_instance_assignment -name DUPLICATE_HIERARCHY_DEPTH -to regZ 2
```



当 num_levels 设置为 2 时，regY 和 regZ 被从链中拉出。regZ 结束于最大层次深度 2，而 regY 结束于最大层次深度 1。

图 35. 层次结构中的寄存器复制

```
set_instance_assignment -name DUPLICATE_HIERARCHY_DEPTH -to regZ 3
```



当 num_levels 设置为 3 时，所有 3 个寄存器（regZ，regY 和 regZ）都从链中拉出，并分别推入 3，2 和 1 级的最大层次结构深度。

Synthesis 报告中的 **Hierarchical Tree Duplication Summary** 面板提供关于由 DUPLICATE_HIERARCHY_DEPTH 约束指定寄存器的信息。其中还包括为何链的长度可用作进一步改进约束的起点的原因。Synthesis 报告还提供名为 **Hierarchical Tree Duplication Details** 的面板，以提供关于链中各个寄存器的信息，进而更好了解已实现复制的结构。

5.6.5.9. 防止 Shift Register 推断

关闭移位寄存器推断可提高性能。该设置强制软件使用逻辑单元以实现移位寄存器，而并非使用 ALTSHIFT_TAPS IP 核实现存储器块中的寄存器。如果在逻辑单元而非存储器中实现移位寄存器，则会增加逻辑利用率。

5.6.5.10. 使用 Synthesis Tool 中的其他可用综合选项

使用您的综合工具，尝试以下选项是否可用：

- Turn on register balancing or retiming（开启寄存器权衡或重新定时）
- Turn on register pipelining（开启寄存器流水线）
- Turn off resource sharing（关闭资源共享）

这些选项可提高性能，但通常也会提高设计中的资源使用率。

5.6.5.11. Fitter Seed

Fitter seed 影响设计的初始布局配置。初始条件的任何变化都会改变 Fitter 结果；因此，每个 seed 值导致一些不同的适配结果。可尝试不同的 seed，以图获得更好的适配结果和时序性能。

设计中的更改会影响编译之间的性能。这种随机变化是布局布线算法中固有的，所以不可能尝试所有 seed 以期获得绝对最佳结果。

注意: 任何直接或间接影响 Fitter 的设计变更都会都与更改 seed 值产生的随机影响类型相同。其中包括源文件中，**Compiler Settings** 或 **Timing Analyzer Settings** 中的任何更改。如果使用不同的计算机处理系统类型或不同的操作系统，都会出现类似影响，因为不同的系统会改变 Fitter 中计算浮点数的方式。

如果优化设置中的更改仅少许影响 register-to-register 时序或错误路径的数量，就无法确定是否因为更改而引起提高或降低，又或者是否因为 Fitter 中的随机影响而造成。如果您的设计仍在更改，请运行 seed sweep（通过多个 seed 编译设计）确定优化更改后平均结果是否得到改善，以及从增加编译时间的设置中得到值得牺牲时间的理想结果，例如物理综合设置。该扫描还会显示设计中预期的随机变化量。

如果设计已定案，就可使用各个 seed 进行编译获得最佳结果。然而，如果随后对设计进行任何更改，就可能需要再次执行 seed 扫描。

点击 **Assignments** > **Compiler Settings** 通过 seed 控制初始布局。可使用 DSE II 轻松执行 seed 扫描。

使用如下 Tcl 命令指定 Fitter seed:

```
set_global_assignment -name SEED <value>
```

相关链接

Design Space Explorer II (第 10 页)

5.6.5.12. 将 Router Timing Optimization 设置为 Maximum

当布线程序不采用最佳布线线缆时，为改善设计中的布线性，可将 **Router Timing Optimization Level** 设置为 **Maximum**。该设置将确定布线程序尝试满足时序要求的积极程度。将该选项设置为 **Maximum** 可略微提高设计速度但会以增加编译时间为代价。该选项设置为 **Minimum** 可减少编译时间但代价是设计速度略有降低。默认值为 **Normal**。

相关链接

布线程序时序优化级别逻辑选项
Intel Quartus Prime Help 中

5.6.6. 位置约束

如果仅少数路径无法满足其时序要求，就可使用硬性位置约束优化布局。

位置约束关于 Intel Quartus Prime Fitter 的灵活性不如 Logic Lock 约束好。此外，如果您熟悉自己的设计，就可通过进入位置约束的方式来得到更好的结果。

注意: 改善适配结果，尤其对于较大器件，例如 Arria® 和 Stratix 系列器件，会有困难。位置约束并非总能提高设计性能。很多情况下，无法通过位置约束来改善 Fitter 的结果。

5.6.7. 亚稳定性分析和优化技术

当信号在不相关或异步时钟域的电路之间传输时，就会出现亚稳定性问题，因为设计人员无法保证信号满足其建立和保持时间要求。故障间隔时间（MTBF）是实例之间平均时间的估计值，也是可能导致设计失败的亚稳定状态。

当设计在综合一部信号以优化设计提高 MTBF 时，可使用 Intel Quartus Prime 软件分析因亚稳定性导致的平均 MTBF。这些亚稳态功能仅支持由 Timing Analyzer 约束设计以及特定器件系列。

如果您设计中的 MTBF 较低，请参阅 Timing Optimization Advisor 中的 Metastability Optimization 部分，其中建议了各种设置，以亚稳态为依据帮助优化您的设计。

本章节说明如何使能亚稳态分析并识别设计中的寄存器同步链，提供亚稳态报告的详细信息，以及提供管理亚稳态的其他直到原则。

相关链接

- 了解 FPGA 中的亚稳定性
- Managing Metastability with the Intel Quartus Prime Software
  In *Intel Quartus Prime Pro Edition User Guide: Design Recommendations*

5.6.8. Intel Stratix 10 时序收敛建议

注意: 本部分仅适用于针对 Intel Stratix 10 器件系列的设计。其他系列不具备本小节中描述的性能。

传统的 FPGA 时序收敛流程中，大多数设计分析的起点是关键路径。但由于 Intel Stratix 10 器件的性质和 Hyper Retimer 的可用性，最好从 Retiming Limit Report 开始时序收敛活动。要先为工具提供尽可能多的可优化条件，才再考虑更大的时间强度和手动时序收敛的可能性。

相关链接

解释关键链报告
In *Intel Stratix 10 高性能设计手册*

5.6.8.1. 重定时限制详情报告

使用 Retiming Limit Details report（重定时限制详情报告）获得更多关于当前限制 Hyper Retimer 执行更多优化的具体详情。

Retiming Limit Details report 具体内容:

- Clock Transfer（时钟传输）：时钟域，或关键链所应用的时钟域的传输
- Limiting Reason（限制原因）：限制进一步优化的设计条件。
- Critical Chain Details（关键链详情）：与时序限制相关联的时序路径。

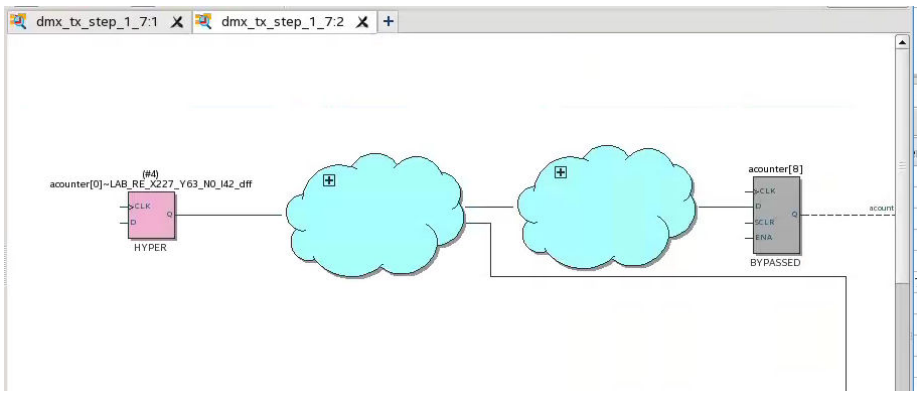
5.6.8.1.1. 使用 Retiming Limit Details 报告

访问 Retiming Limit Details 报告:

1. **Reports** 选项卡中，在 **Fitter** > **Retime Stage** 下双击 **Retiming Limit Details**。
2. 要在 Technology Map Viewer 中找到关键链，右键单击任何路径并单击 **Locate Critical Chain in Technology Map Viewer**。

   Technology Map Viewer 显示布局，布线和寄存器重新定时后完整关键链的原理图。

图 36. Technology Map Viewer 中的关键链



5.6.8.2. Fast Forward 时序收敛建议

运行 Fast Forward 编译时，Compiler 从寄存器中删除信号以允许网表内的可移动性从而进行后续重新定时。Fast Forward 编译生成特定设计收敛建议，并通过移除所有时序限制来预测最佳性能。

完成 Fast Forward 管理后，可确定为提供最大收益需实现的建议。在 RTL 中实现适当的建议，并重编译设计以达到 Fast Forward 报告的性能水平。

Fast Forward Details Report 提供如下信息：

表 15. Fast Forward 详细报告信息

名称	说明
Step	从预优化基础编译开始，显示各种 Fast Forward 优化步骤。 • 每个步骤都有其相关联的关键链。 • 每个步骤对应上一步骤的新优化积累。
Fast Forward Optimization	分析实现每个步骤所需优化的摘要。
Estimated f_{MAX}	实现设计中关于该步骤的建议后，估计 f_{MAX} 性能。该操作具累积性，且步骤 *n* 代表实现前述所有步骤后的潜在 f_{MAX}。
Optimization Analyzed	（累积）已应用的所有连续优化步骤的列表。
Recommendation for Critical Chain	列出针对设计建议更改的内容。这些建议旨在移除限制，并允许寄存器移动。

相关链接

- 解读 Fast Forward Compile 报告
  In *Intel Stratix 10 高性能设计手册*
- 接收或拒绝 Fast Forward 优化
  In *Hyper-Optimization for Intel Stratix 10 Designs*

5.6.8.2.1. 生成 Fast Forward 时序收敛建议

要生成 Fast Forward 时序收敛建议：

1. Compilation Dashboard 上，单击 **Fast Forward Timing Closure Recommendations**。

Compiler 运行必要综合或所需的 Fitter 阶段，并在 Compilation Report 中生成时序收敛建议。

2. 查看 Compilation Report 中的时序收敛建议，以评估设计性能并进行关键 RTL 性能改进。

Intel Quartus Prime Pro Edition 软件允许自动或改善 Fast Forward 分析：

- 每个全编译中的 Fast Forward 编译，可点击 **Assignments** > **Settings** > **Compiler Settings** > **HyperFlex**，并开启 **Run Fast Forward Timing Closure Recommendations during compilation**。
- 要修改 Fast Forward 编译解释指定 I/O 和快类型的方式，点击 **Assignments** > **Settings** > **Compiler Settings** > **HyperFlex Advanced Settings**。

5.6.8.2.2. 实现 Fast Forward 建议

设计中实现时序收敛建议后，可重新运行 Retime 阶段以获得预期的性能提升。

可继续管理功能并对代码执行 RTL 更改，直到达到所需的性能目标。完成所有需要进行的修改后，请使用本文档中介绍的传统技术继续时序收敛活动。

更多关于设计中实现 Fast Forward 时序收敛建议的详细信息，请参阅 *Intel Stratix 10 高性能设计手册* 中 *实现 Fast Forwar 建议* 部分。

相关链接

实现 Fast Forward 建议
In *Intel Stratix 10 高性能设计手册*

5.7. 外设到内核寄存器布局和布线优化（P2C）

Periphery to Core Register Placement and Routing Optimization（P2C）选项指定 Fitter 是否对外设逻辑和 FPGA 内核中寄存器之间的直接连接上执行集成目标性布局和布线优化。P2C 是一个可选的预布线感知布局优化阶段，使您能够更可靠实现时序收敛。

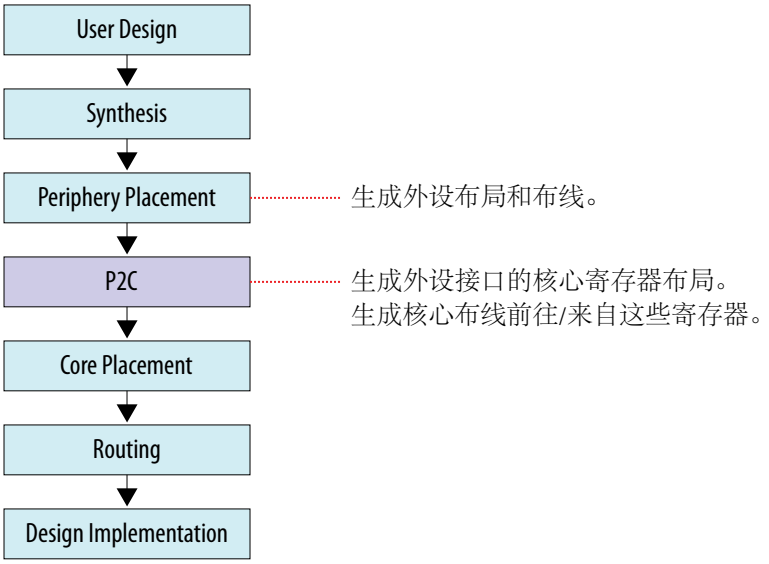
注意: **Periphery to Core Register Placement and Routing Optimization** 选项在两个方向都适用：外设到内核，以及内核到外设。

外部接口（如，高速 I/O 或串行接口）和 FPGA 之间的传输通常需要布线多个严格设置和保持时间要求的连接。开启该选项时，Fitter 先执行 P2C 布局和布线决定，然后才是内核布局和布线。这样就能保留必需资源确保设置实现其时序要求，并避免与外部接口进行传输时出现布线拥塞。

该选项可用作全局约束，或可应用于设计中的特定实例。

图 37. 外设到内核寄存器布局和布线优化（P2C）流程

P2C 运行于外设布局后，并在相应 P2C/C2P 路径上生成内核寄存器布局，以及与这些内核寄存器之间的内核布线。



在 Advanced Fitter Setting 对话框中设置外设到内核优化 (第 89 页)
在 Assignment Editor 中设置外设到内核优化 (第 89 页)
在 Fitter Report 中查看外设到内核优化 (第 90 页)

5.7.1. 在 Advanced Fitter Setting 对话框中设置外设到内核优化

Periphery to Core Placement and Routing Optimization 设置指定 Fitter 是否优化 FPGA 内核中外设逻辑和寄存器之间直接连接上的目标布局和布线。

可使用 Assignment Editor 中的设置对实例选择性执行通过外设对内核优化。

1. 在 Intel Quartus Prime 软件中，点击 **Assignments** > **Settings** > **Compiler Settings** > **Advanced Settings (Fitter)**。
2. 在 **Advanced Fitter Settings** 对话框中，对于 **Periphery to Core Placement and Routing Optimization** 选项，根据您希望设计中外围对内核优化的方式选择如下选项：
   a. 选择 **Auto** 可指示软件自动识别具有严格时序窗口的传输，放置内核寄存器，并布线所有与外设间的往来连接。
   b. 选择 **On** 可指示软件全局优化外设和内核寄存器之间的全部传输，且不考虑时序要求。

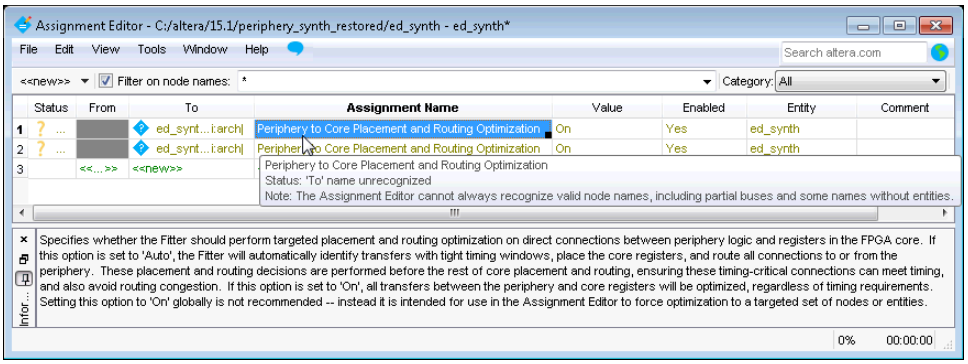
      注意: 不建议将 **Advanced Fitter Settings** 中的该选项设置为 **On**。该设置旨在通过 Assignment Editor 强制优化目标节点和实例集。
   c. 选择 **Off** 禁用设计中外设到内核路径优化。

5.7.2. 在 Assignment Editor 中设置外设到内核优化

在 Assignment Editor 中开启 **Periphery to Core Placement and Routing Optimization** (P2C/C2P)设置时， Intel Quartus Prime 软件对设计中的选定实例执行外设到内核，或内核到外设优化。

可使用 **Advanced Fitter Settings** 对话框中的设置对实例选择性执行外设到内核优化。

1. 在 Intel Quartus Prime 软件中，点击 **Assignments** > **Assignment Editor**。
2. 对于已选路径，双击 **Assignment Name** 栏，在下拉菜单中点击 **Periphery to core register placement and routing optimization** 选项。
3. 在 **To** 栏，选择 P2C/C2P 路径上需要优化的外设节点或内核寄存器。将 **From** 栏留空。对于 Assignments Editor 中出现的路径，必须首先在设计上运行 Analysis & Synthesis 。



5.7.3. 在 Fitter Report 中查看外设到内核优化

编译后， Intel Quartus Prime 软件通过 **Fitter (Place & Route)**报告生成外设到内核优化和布线优化总结。

1. 编译您的 Intel Quartus Prime 工程。
2. 在 **Tasks** 窗格中，选择 **Compilation**。
3. **Fitter (Place & Route)**下，双击 **View Report**。
4. **Fitter** 文件夹中，展开 **Place Stage** 文件夹。
5. 双击 **Periphery to Core Transfer Optimization Summary**。

表 16. Fitter Report 一外设到内核传输优化（P2C）摘要

出发路径	路径	状态
Node 1	Node 2	**Placed and Routed**—内核寄存器已锁定。外设到内核/内核到外设布线已提交。
Node 3	Node 4	**Placed but not Routed**—内核寄存器已锁定。布线未提交。该情况出现在 P2C 未使能无法优化单个组内的所有目标路径时，例如，相同的延迟/线缆要求，或相同的控制信号。部分 P2C 布线提交可能导致无法解决的布线拥塞。
Node 5	Node 6	**Not Optimized**—改状态出现在 P2C 设置为 **Auto** 时且由于如下问题路径未得到优化： a. 不可能实现延迟要求。 b. 最小延迟要求（保持时间）过大。当需要添加多个线缆以满足保持时间时，P2C 算法不能有效处理。 c. P2C 遇到无法解决的特定路径布线拥塞。

Periphery to Core Transfer Optimization Summary

	From	To	Status
1	Periphery to Core Transfer		
2	dut|arch|arch_inst|io_tiles_wrap_inst|io_tiles_inst|tile_gen[1].lane_gen[3].lane_inst	dut|arch|arch_inst|afi.if_inst|single_port.afi_rdata_valid_regs|sr_out[0]	Placed but Not Routed
3	dut|arch|arch_inst|io_tiles_wrap_inst|io_tiles_inst|tile_gen[2].lane_gen[2].lane_inst	dut|arch|arch_inst|afi.if_inst|mem_sp_bidir_data.mem_dq.afi_regs_i|sr_out[63]	Placed but Not Routed
4	dut|arch|arch_inst|io_tiles_wrap_inst|io_tiles_inst|tile_gen[0].lane_gen[1].lane_inst	dut|arch|arch_inst|afi.if_inst|mem_sp_bidir_data.mem_dq.afi_regs_i|sr_out[11]	Placed but Not Routed
5	dut|arch|arch_inst|io_tiles_wrap_inst|io_tiles_inst|tile_gen[0].lane_gen[0].lane_inst	dut|arch|arch_inst|afi.if_inst|mem_sp_bidir_data.mem_dq.afi_regs_i|sr_out[7]	Placed but Not Routed
6	dut|arch|arch_inst|io_tiles_wrap_inst|io_tiles_inst|tile_gen[0].lane_gen[0].lane_inst	dut|arch|arch_inst|afi.if_inst|mem_sp_bidir_data.mem_dq.afi_regs_i|sr_out[3]	Placed but Not Routed
7	dut|arch|arch_inst|io_tiles_wrap_inst|io_tiles_inst|tile_gen[0].lane_gen[0].lane_inst	dut|arch|arch_inst|afi.if_inst|mem_sp_bidir_data.mem_dq.afi_regs_i|sr_out[0]	Placed but Not Routed
8	dut|arch|arch_inst|io_tiles_wrap_inst|io_tiles_inst|tile_gen[0].lane_gen[0].lane_inst	dut|arch|arch_inst|afi.if_inst|mem_sp_bidir_data.mem_dq.afi_regs_i|sr_out[79]	Placed but Not Routed
9	dut|arch|arch_inst|io_tiles_wrap_inst|io_tiles_inst|tile_gen[0].lane_gen[0].lane_inst	dut|arch|arch_inst|afi.if_inst|mem_sp_bidir_data.mem_dq.afi_regs_i|sr_out[75]	Placed but Not Routed
10	dut|arch|arch_inst|io_tiles_wrap_inst|io_tiles_inst|tile_gen[0].lane_gen[0].lane_inst	dut|arch|arch_inst|afi.if_inst|mem_sp_bidir_data.mem_dq.afi_regs_i|sr_out[73]	Placed but Not Routed
11	dut|arch|arch_inst|io_tiles_wrap_inst|io_tiles_inst|tile_gen[2].lane_gen[3].lane_inst	dut|arch|arch_inst|afi.if_inst|mem_sp_bidir_data.mem_dq.afi_regs_i|sr_out[71]	Placed but Not Routed
12	dut|arch|arch_inst|io_tiles_wrap_inst|io_tiles_inst|tile_gen[2].lane_gen[3].lane_inst	dut|arch|arch_inst|afi.if_inst|mem_sp_bidir_data.mem_dq.afi_regs_i|sr_out[69]	Placed but Not Routed
13	dut|arch|arch_inst|io_tiles_wrap_inst|io_tiles_inst|tile_gen[1].lane_gen[3].lane_inst	dut|arch|arch_inst|afi.if_inst|mem_sp_bidir_data.mem_dq.afi_regs_i|sr_out[35]	Placed but Not Routed
14	dut|arch|arch_inst|io_tiles_wrap_inst|io_tiles_inst|tile_gen[0].lane_gen[3].lane_inst	dut|arch|arch_inst|afi.if_inst|mem_sp_bidir_data.mem_dq.afi_regs_i|sr_out[31]	Placed but Not Routed
15	dut|arch|arch_inst|io_tiles_wrap_inst|io_tiles_inst|tile_gen[0].lane_gen[3].lane_inst	dut|arch|arch_inst|afi.if_inst|mem_sp_bidir_data.mem_dq.afi_regs_i|sr_out[27]	Placed but Not Routed
16	dut|arch|arch_inst|io_tiles_wrap_inst|io_tiles_inst|tile_gen[0].lane_gen[2].lane_inst	dut|arch|arch_inst|afi.if_inst|mem_sp_bidir_data.mem_dq.afi_regs_i|sr_out[23]	Placed but Not Routed

5.8. 脚本支持

可运行本手册中 Tcl 脚本说明的过程并进行设置。还可按照提示命令指示运行。关于脚本选项的详细信息，请参阅 Intel Quartus Prime 命令行和 Tcl API Help 浏览器。运行“Help”浏览器，可在提示命令键入如下命令：

```
quartus_sh --qhelp
```

可在实例或/和全局级中指定本小节介绍的多个选项。

使用以下 Tcl 命令进行全局约束：

```
set_global_assignment -name <.qsf variable name> <value>
```

使用以下 Tcl 命令进行实例约束：

```
set_instance_assignment -name <.qsf variable name> <value> -to <instance name>
```

注意: 如果*<value>*字段包含空格（例如，‘Standard Fit’），则必须以英文双引号附上其值。

相关链接

- Intel Quartus Prime Pro Edition 设置文件参考手册
  关于 Intel Quartus Prime 软件中所有设置和约束的信息。
- Tcl Scripting
  Intel Quartus Prime Pro Edition 用户指南：脚本
- Command Line Scripting
  Intel Quartus Prime Pro Edition 用户指南：脚本

5.8.1. 初始编译设置

在 Tcl 约束中使用 Intel Quartus Prime Settings File (`.qsf`) 变量名以及正确值进行设置。**Type** 栏标示设置是否用做全局设置，实例设置，或两者兼顾。

上方表格罗列`.qsf` 变量名称以及*设计优化概述*章节的*初始编译：必要设置*部分中介绍的可用于设置的值。下方表格列出了各高级编译设置。

表 17. 初始编译设置

设置名称	.qsf File Variable Name	值	类型
Optimize IOC Register Placement For Timing（针对时序优化 IOC 寄存器布局）	OPTIMIZE_IOC_REGISTER_PLACEMENT_FOR_TIMING	ON, OFF	Global
优化保持时序	OPTIMIZE_HOLD_TIMING	OFF, IO PATHS AND MINIMUM TPD PATHS, ALL PATHS	Global

表 18. 高级编译设置

设置名称	.qsf 文件变量名称	值	类型
Router Timing Optimization level（布线程序时序优化程度）	ROUTER_TIMING_OPTIMIZATION_LEVEL	NORMAL, MINIMUM, MAXIMUM	Global

相关链接

设计优化概述 (第 6 页)

5.8.2. I/O 时序优化技术

本表格列出.qsf 文件变量名称以及可用于 I/O 时序优化设置的值。

表 19. I/O 时序优化设置

设置名称	.qsf 文件变量名称	值	类型
Optimize IOC Register Placement For Timing（针对时序优化 IOC 寄存器布局）	OPTIMIZE_IOC_REGISTER_PLACEMENT_FOR_TIMING	ON, OFF	Global
Fast Input Register	FAST_INPUT_REGISTER	ON, OFF	Instance
Fast Output Register	FAST_OUTPUT_REGISTER	ON, OFF	Instance
Fast Output Enable Register	FAST_OUTPUT_ENABLE_REGISTER	ON, OFF	Instance
Fast OCT Register	FAST_OCT_REGISTER	ON, OFF	Instance

5.8.3. Register-to-Register 时序优化技术

本表格列出.qsf 文件变量名称以及可用于 *Register-to-Register 时序优化技术*中介绍的设置的值。

表 20. Register-to-Register 时序优化设置

设置名称	.qsf 文件变量名称	值	类型
Perform WYSIWYG Primitive Resynthesis	ADV_NETLIST_OPT_SYNTH_WYSIWYG_REMAP	ON, OFF	Global, Instance
Fitter Seed	SEED	*<integer>*	Global
Maximum Fan-Out	MAX_FANOUT	*<integer>*	Instance

继续...

设置名称	.qsf 文件变量名称	值	类型
Manual Logic Duplication（手动逻辑复制）	DUPLICATE_ATOM	*<node name>*	Instance
Optimize Power during Synthesis（综合过程中优化功率）	OPTIMIZE_POWER_DURING_SYNTHESIS	NORMAL, OFF EXTRA_EFFORT	Global
Optimize Power during Fitting（优化 Fitting 过程中的功率）	OPTIMIZE_POWER_DURING_FITTING	NORMAL, OFF EXTRA_EFFORT	Global

相关链接

Register-to-Register 时序优化技术 (第 77 页)

5.9. 时序收敛和优化修订历史

以下修订历史适用于本章：

文档版本	Intel Quartus Prime 版本	修订内容
2019.07.01	19.1	为*寄存器自动复制：估算物理性接近度*和*寄存器自动复制：层次性接近度*主题添加了重要说明。
2019.04.01	19.1	• 在*复制扇出控制的逻辑*主题中添加了有关寄存器复制方法的更多信息。 • 将与手动寄存器复制相关的内容从 *Duplicate Logic for Fan-out Control* 主题移动到新创建的*手动添加复制寄存器*副主题中。 • 将*寄存器自动复制：估算物理性接近度*和*寄存器自动复制：层次性接近度*添加到*复制扇出控制逻辑*主题下作为新的副主题，以说明寄存器自动复制过程。
2018.11.12	18.1.0	• 更新了“调整布局工作量”主题中的“布局工作量乘法器”图示和文字说明。 • 更新了“调整 Fitter 工作量”主题中的“Fitter 工作量”图示和文字说明。 • 更新了“针对保持时序添加线缆"主题中的“优化保持时序选项”截屏图。
2018.09.24	18.1.0	• 删除了重复主题：*资源利用率优化技术*。该主题现位于*区域优化*章节。 • 从”针对时序逻辑选项优化 IOC 寄存器布局"主题中删除了关于不支持 CARRY 和 CASCADE 缓冲的参考内容。
2017.11.06	17.1.0	• 添加了关于 Intel Stratix 10 Hyper-Retiming，Fast Forward 编译和 Fast Forward Viewer 的支持。 — 添加了主题：“关键链”，“查看关键链”，“Intel Stratix 10 时序收敛建议”，“重新定时限制详细报告”，“使用 Retiming Limit Details Report “，“Fast Forward 时序收敛建议”，“生成 Fast Forward 时序收敛建议”，“实现 Fast Forward 建议”。 • 添加关于使用分区实现时序收敛的主题。 • 移动主题位置：将“时序收敛的设计评估”移动到“初始编译：可选的 Fitter 设置”之后。 • 删除了关于在设计的一个部分中应用物理综合的声明。 • 删除了关于优化所选路径的保持时序的参考。 • 更新了关于资源使用率优化设置的逻辑选项。
2017.05.08	17.0.0	• 添加了主题：*关键路径*。 • 更新了 *Register-to-Register 时序*并将其重命名为 *Register-to-Register 时序分析*。 • 重命名主题：将*使用 Timing Analyzer 的时序分析*重命名为*使用 Timing Analyzer 显示路径报告*。 • 从主题标题中删除了（LUT-Based Devices）备注。 • 重命名主题：将*优化时序(LUT-Based Devices)*更改为*时序优化*。 • 重命名主题：将*在 Timing Analyzer 中调试时序错误*重命名为*显示错误路径的时序收敛建议*。

继续...

文档版本	Intel Quartus Prime 版本	修订内容
		• 重命名主题：将*改进 Register-to-Register 时序的摘要*更名为*改善 Register-to-Register 时序*。 • 已删除主题：*Tips for Locating Multiple Paths to the Chip Planner*，*LogicLock Assignments* and *Hierarchy Assignments*。 • 删除了对已淘汰 Fitter Effort Logic Option 的参考。 • 删除了关于 Pin Advisor 和 Resource Optimization Advisor 的内容。 • 删除了图示：Clock Regions
2016.10.31	16.1.0	• 品牌更名为 Intel。
2016.05.02	16.0.0	• 删除对已淘汰物理综合选项的参考。 • 添加了关于监控集群难度的信息。
2015.11.02	15.1.0	• 添加了：*外设到内核寄存器布局和布线优化*。 • 将 *Quartus II* 更改为 *Quartus Prime*。
2014.12.15	14.1.0	• 将 Fitter Setting，Analysis & Synthesis 和 Physical Synthesis Optimizations Setting 的位置更新到 Compiler Setting。 • 更新了 DSE II 的内容。
2014 年 6 月	14.0.0	• DITA 转换。 • 删除了 QII 软件 v14.0 中关于不再支持的淘汰器件的内容：Arria GX, Arria II, Cyclone III, Stratix II, Stratix III. • 以 IP 核内容替换 Megafunction 内容。
2013 年 11 月	13.1.0	• 添加了针对时序收敛的设计评估 • 删除了 Optimizing Timing (Macrocell-Based CPLDs)部分。 • 更新了优化 Multi-Corner 时序和 Fitter 主动布通性优化。 • 更新了通过 Timing Analyzer 进行时序分析以显示如何访问 **Report All Summaries** 命令。 • 更新了忽略时序收敛，包含了关于 *Fitter Summary Reports* 的帮助链接提供 **Ignored Assignment Report** 信息。
2013 年 5 月	13.0.0	• 将章节标题从“时序优化”重命名为“时序收敛和优化”。 • 删除了设计和区域/资源优化信息。 • 添加了如下部分： Fitter 主动布通性优化。 关于分析往来关键路径的源和目标路径的提示。 定位 Chip Planner 的多个路径的提示。 创建.tcl 脚本以监控跨编译的关键路径的提示。
2012 年 11 月	12.1.0	• 更新了“初始编译：可选的 Fitter 设置”，第 13 – 2 页；“I/O 约束”，第 13 – 2；“初始编译：可选的 Fitter 设置”，第 13 – 2 页；“资源利用率”，第 13 – 9 页；“布线”，第 13 – 21 页，以及“解决资源利用率问题”，第 13 – 43 页。
2012 年 6 月	12.0.0	• 更新了“优化 Multi-Corner 时序”，第 13 – 6 页；“资源利用率“，第 13 – 10 页；”使用 Timing Analyzer 进行时序分析“，第 13 – 12 页；“使用 Resource Optimization Advisor”，第 13 – 15 页；“增加 Placement Effort Multiplier”，第 13 – 22 页；”增加 Router Effort Multiplier”，第 13 – 22 页；和“Timing Analyzer 中调试时序失败”，第 13 – 24 页。 • 本章中的少量文本编辑。
2011 年 11 月	11.1.0	• 更新了“时序要求设置”，“标准 Fit”，“快速 Fit”，“优化 Multi-Corner 时序”，“使用 Timing Analyzer 进行时序分析”，“Timing Analyzer 中调试时序错误”，“LogicLock 约束”，“跨时钟域分析错误时钟的提示”，“综合期间展平层次”，“快速输入，输出和输出使能寄存器”和“层次约束”部分 • 更新了表格 13-6。 • 添加了“Spine Clock 限制”部分

继续...

文档版本	Intel Quartus Prime 版本	修订内容
		• 从第 19 页删除了“Change State Machine Encoding”部分。 • 删除了图示 13-5 • 本章中的少量文本编辑。
2011 年 5 月	11.0.0	• 重组“初始编译：可选 Fitter 设置”部分 • 为“资源利用率“部分添加新 • 为“复制删除控制的逻辑”部分添加新信息 • 添加 Help 链接 • 本章中其他内容的编辑和更新
2010 年 12 月	10.1.0	• 添加 Help 链接 • 更新了器件支持清单。 • 添加了“在 Timing Analyzer 中调试时序失败“部分 • 删除了 Classic Timing Analyzer 参考 • 本章中的其他更新
2010 年 8 月	10.0.1	更正链接
2010 年 7 月	10.0.0	• 将“编译时间优化技术”部分移动到新的“*缩短编译时间*”章节 • 删除了关于“时序收敛布局规划”的参考 • 将“智能编译设置和早期时序估算”部分移动到“*缩短编译时间*”章节 • 添加了其他优化资源部分 • 移除了过时的信息。 • 将关于 DSE 章节的参考内容更换到 Help 链接 • 链接到 Help 中的对应内容 • 删除了“参考文档部分

相关链接

文档存档

了解 *Intel Quartus Prime 手册*以前的版本，请搜索文档存档。

6. 分析和优化设计平面布局规划

随着 FPGA 设计密度的增加，分析设计性能，布线拥塞和逻辑布局的能力对于满足设计要求至关重要。 本章讨论 Chip Planner 和 Logic Lock 区域如何帮助改善设计布局规划。

设计布局规划分析有助于收敛时序并确保高度复杂设计中实现最佳性能。通过 Intel Quartus Prime Chip Planner 的分析能力，可帮助快速完成设计的时序收敛。可将 Chip Planner 连同 Logic Lock 区域一起使用以分层编译设计并协助布局规划。此外，使用分区保留单次编译运行后的布局和布线结果。

可执行设计分析，以及通过 Chip Planner 创建并优化设计布局规划。要进行 I/O 约束，请使用 Pin Planner。

注意: 最佳实践是，使用迭代设计流程定义资源布局。设置硬布局约束之前，可使用例如“Early Place Flow”之类的技术指导平面布局规划决策。

关于 Early Place Flow 的更多信息，请参阅 *Intel Quartus Prime Pro Edition 用户指南：编译器* 。

关于布局规划 Partial Reconfiguration 设计的更多信息，请参阅 *Intel Quartus Prime Pro Edition 用户指南：局部重新配置* 。

相关链接

- 早期布局流程
  Intel Quartus Prime Pro Edition 用户指南：编译器
- 布局规划局部局部重配置设计
  In *Intel Quartus Prime Pro Edition 用户指南：局部重新配置*
- 管理器件 I/O 管脚

6.1. Chip Planner 中的设计平面布局分析

Chip Planner 直观显示芯片资源，简化了布局规划分析。通过 Chip Planner，可查看编译后的布局，连接和布线路径。还可以进行约束更改，例如创建和删除资源约束。

Chip Planner 概述

- Logic Lock 区域
- 相关资源使用情况
- 详细的布线信息
- 节点间的扇入和扇出连接
- 寄存器之间的时序路径
- 路径的延迟估算
- 布线拥塞信息

6.1.1. 启动 Chip Planner

要启动 Chip Planner，请选择 **Tools** > **Chip Planner**。还可通过以下方法启动 Chip Planner：

- 在 Intel Quartus Prime 软件工具栏上点击 Chip Planner 图标。
- 在以下工具中，右键点击任何芯片资源并选择 **Locate** > **Locate in Chip Planner**：
  - Compilation Report
  - **Logic Lock Regions Window**
  - Technology Map Viewer
  - **Project Navigator** window
  - Node Finder
  - Simulation Report
  - Report Timing panel of the Timing Analyzer

6.1.2. Chip Planner GUI 组件

6.1.2.1. Chip Planner Toolbar

Chip Planner 工具栏为直观设计分析提供强大工具。可从 **View** 菜单，或点击工具栏中的图标访问 Chip Planner 命令。

6.1.2.2. 层次设置

Chip Planner 允许控制资源的显示。

Layers Setting（层次设置）窗格

通过 **Layers Settings** 窗格，可管理 Chip Planner 显示的图形元素。

单击 **View** > **Layers Settings** 开启 **Layers Settings** 窗格。**Layers Settings** 窗格提供图层预设，以将经常一同使用的资源进行分组。**Basic**，**Detailed** 和 **Floorplan Editing** 默认预设有助于常规约束相关的活动。还可根据需要创建自定义预设。

相关链接

- 查看特定体系结构设计信息 (第 98 页)
- Layers Settings Dialog Box
  Intel Quartus Prime Help 中

6.1.2.3. 查找历史窗口

优化设计布局规划时，可能需要多次在 Chip Planner 查找路径或节点。 **Locate History** 窗口列出您使用 **Locate in Chip Planner** 命令显示的所有节点和路径，可轻松访问你感兴趣的节点和路径。

如果从 **Timing Analyzer Report Timing** 窗格中查找所需路径，**Locate History** 窗会显示所需时钟路径。如果从 **Timing Analyzer Report Timing** 窗格查找到达路径，**Locate History** 窗口会显示从到达时钟到到达数据的路径。**Locate History** 窗口中双击一个节点或路径，Chip Planner 中会显示所选节点或路径。

6.1.2.4. Chip Planner 布局规划视图

Chip Planner 使用分层缩放查看器显示各抽象级下的目标 Intel 器件。放大时，抽象水平下降，展现更多设计相关的详细信息。

鸟瞰视图

Bird’s Eye View（鸟瞰视图） 显示整个芯片资源使用情况的高级图像并提供一种快速有效的方式在 Chip Planner 中对并感兴趣的区域进行浏览。

当需要查看的设计部分位于芯片的两端时，以及在不丢失参考框架的情况下，进行资源元件之间快速浏览时，尤为有用。

属性窗口

Properties 窗口显示 Chip Planner 中当前选择对象的详细属性（例如原子，路径，Logic Lock 区域或布线元素）。要显示 **Properties** 窗口，右键点击对象并选择 **View** > **Properties**。

相关链接

Bird's Eye View Window
Intel Quartus Prime Help 中

6.1.3. 在 Chip Planner 中查看设计单元

可使用 Chip Planner 查看和报告设计中各单元的详细信息。例如查看可用的时钟网络，布线拥塞，I/O bank 或平面图中的高速串行接口。

以下部分说明如何在 Chip Planner 中查看各种设计单元。

6.1.3.1. 查看特定体系结构设计信息

Chip Planner 允许查看设计相关的特定体系结构信息。在 **Layers Settings** 窗格中使能选项，可以查看：

- **Device routing resources used by your design**—查看块的连接方式，以及连接块的信号路由。
- **LE configuration**—查看设计中的逻辑元件（LE) 配置。例如，可查看已使用哪些 LE 输入；LE 是否使用寄存器，look-up table（LUT， 查找表）或两者兼顾；还有通过 LE 的信号流。
- **ALM configuration**—查看设计中的 ALM 配置。例如，查看已使用哪些 ALM 输入；ALM 是否占用寄存器，上层 LUT，底层 LUT，或两者兼顾。还可查看通过 ALM 的信号流。
- **I/O configuration**—查看器件 I/O 资源使用情况。例如，可查看已使用 I/O 资源则哪些组件，延迟链设置是否使能，设置为哪个 I/O 标准，和通过 I/O 的信号流。
- **PLL configuration**—查看设计中的锁相环（phase-locked loop，PLL）配置。例如，可查看已使用的 PLL 控制信号中哪些按照您的 PLL 设置。
- **Timing**—查看 FPGA 元件的输入和输出之间的延迟。例如，可分析 DATAB 输入到 COMBOUT 输出的时序。

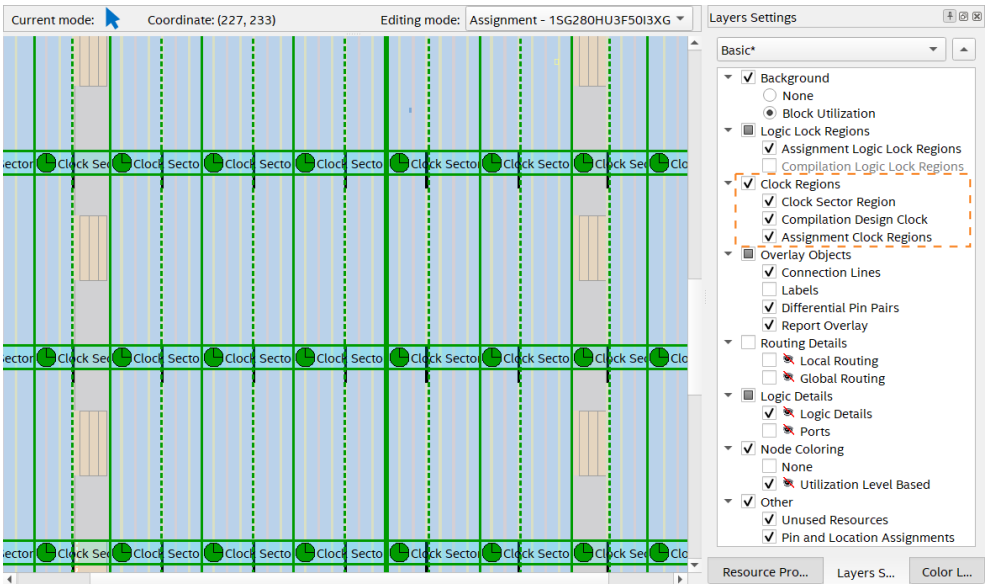
相关链接

- 层次设置 (第 97 页)
- Layers Settings Dialog Box
  Intel Quartus Prime Help 中

6.1.3.2. 查看器件中可用的时钟网络

在 **Layers Settings** 窗格中使能一个时钟区域层时，将显示由全局和区域时钟网络驱动的芯片区域。所选器件不包含给定时钟区域时，对话框中无该类别的选项。

图 38. 时钟区域



- 根据您在 **Layers Settings** 窗格中激活的时钟层，Chip Planner 显示器件中的区域和全局时钟域，以及时钟区域，管脚和 PLL 之间的连接。
- 时钟区域显示为矩形覆盖框，以标签指示时钟类型和索引。点击时钟区域选择一个时钟网络区域。左上角钟形图标表示该区域代表时钟网络区域。
- Spine/sector 时钟区域中间有垂直虚线。本虚线表示行时钟的两列在扇区时钟里相遇的位置。
- 要更改 Chip Planner 中显示时钟区域的颜色，选择 **Tools** > **Options** > **Colors** > **Clock Regions**。

相关链接

- 脊柱时钟限制 (第 77 页)
- 层次设置 (第 97 页)
- Report Spine Clock Utilization dialog box (Chip Planner)
  Intel Quartus Prime Help 中

6.1.3.3. 查看时钟扇区使用情况

Chip Planner 提供设计中时钟扇区使用情况的直观可视表达。

Chip Planner 中生成报告：

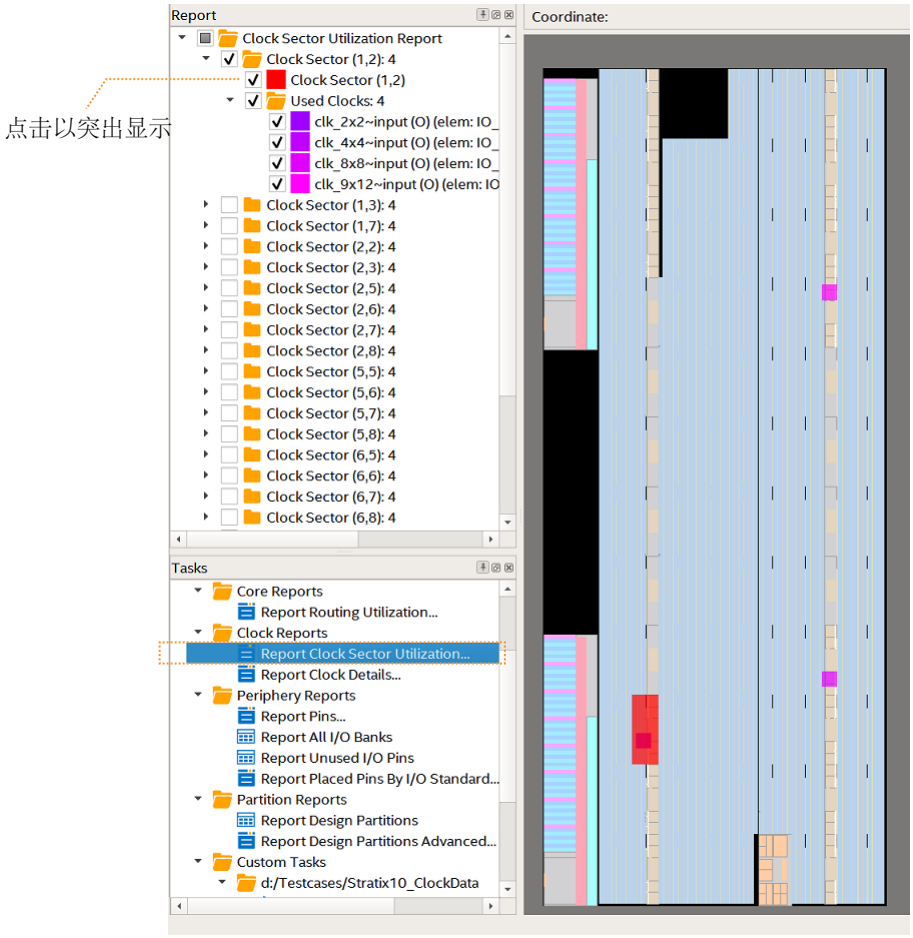
1. **Tasks** 窗格中，双击 **Report Clock Sector Utilization** 打开 **Report Clock Sector Utilization** 对话框。
2. 如果希望该报告包含源节点，请开启 **Report source nodes**。
   等效 TCL 命令出现在对话框底部。
3. 点击 **OK**。

   报告输出显示最常用的时钟扇区。

Report 窗格显示一个以颜色标注各使用情况的时钟扇区清单。使用率最高的时钟扇区以红色显示，使用率最低的扇区以蓝色显示。

可从 **Report** 窗格开启或关闭扇区直观视图。也可突出显示节点（如适用）。

图 39. 时钟扇区使用报告



相关链接

Report Clock Sector Utilization dialog box (Chip Planner)
Intel Quartus Prime Help 中

6.1.3.4. 查看布线拥塞

Report Routing Utilization 任务支持确定编译后正在使用的布线资源百分比。该功能可识别缺乏布线资源的区域，从而有助于您更改设计以满足设计要求的布线拥塞。

在 Chip Planner 中查看布线拥塞：

1. **Tasks** 窗格中，双击 **Report Routing Utilization** 命令启动 **Report Routing Utilization** 对话框。
2. 在 **Report Routing Utilization** 对话框中点击 **Preview** 以预览默认拥堵显示。
3. 更改 **Routing Utilization Type** 以显示指定资源拥塞。
   默认显示以深蓝表示 0%拥塞（蓝色表示使用率为 0）且红色表示使用率为 100%。可调整 **Threshold percentage** 滑块更改拥塞阈值水平。

拥塞图可帮助确定是否可修改平面布局，或修改 RTL 以减少布线拥塞。考量内容：

- 布线拥塞图使用颜色和阴影来指示资源使用率；较深阴影表示布线资源使用率较高。区域的资源使用超出 **Report Routing Utilization** 中所设置的阈值会显示为红色。
- 为确定布线资源缺乏，必须在 **Routing Utilization Settings** 对话框中通过依次选择每种互连类型分别研究每种布线互连类型。
- Compiler 消息包含有关平均和峰值互连使用情况的信息。峰值互连使用率超过 75%或平均互连使用超过 60%可表示设计难以适配的程度。同样，峰值互连使用超过 90%或平均互连超过 75%，则表明无法获得有效适配的可能性增加。

相关链接

查看布线资源 (第 107 页)

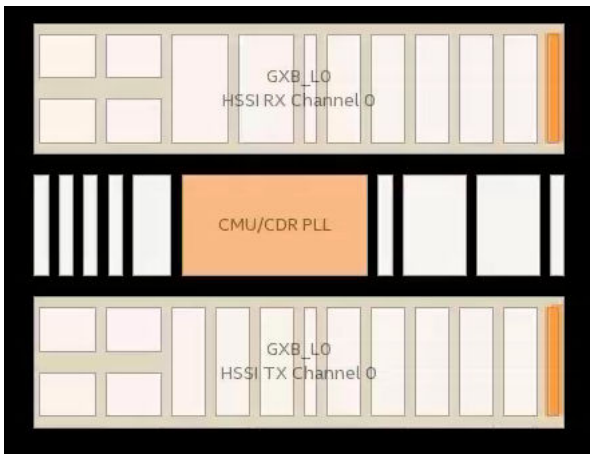
6.1.3.5. 查看 I/O Bank

要在 Chip Planner 中查看器件的 I/O bank 映射，请在 **Tasks** 窗格中双击 **Report All I/O Banks**。

6.1.3.6. 查看高速串行接口（HSSI）

Chip Planner 显示高速串行接口接收器和发送器的块视图详情。要显示 HSSI 块视图，在 **Tasks** 窗格中双击 Report HSSI Block Connectivity。

图 40. Intel Arria 10 HSSI Channel Blocks



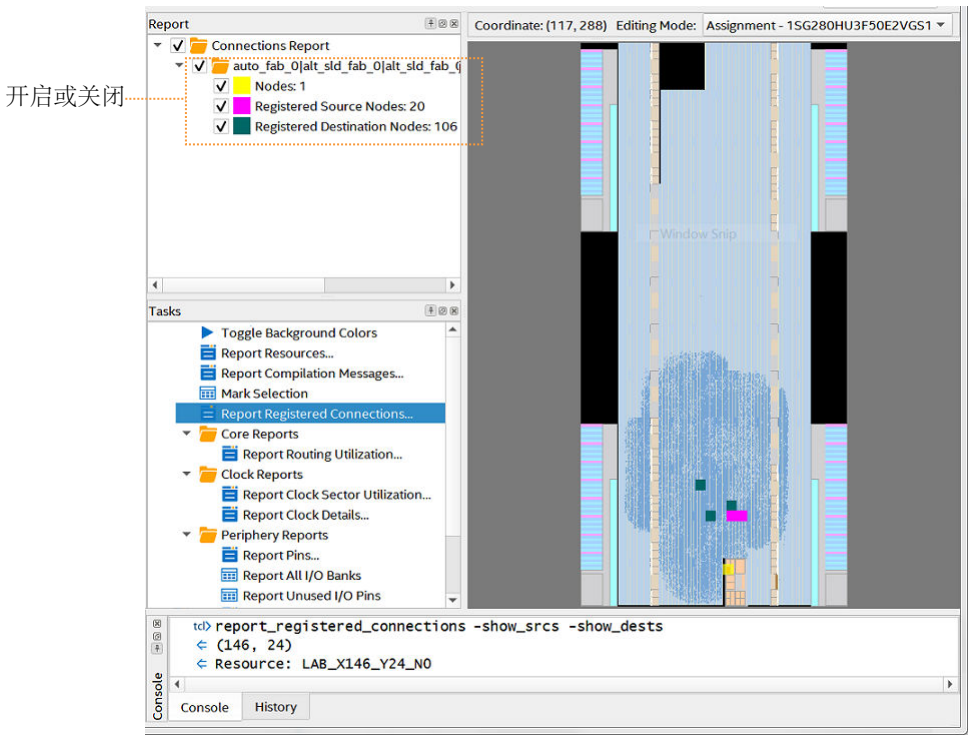
6.1.3.7. 查看已布局节点的源和目标

Chip Planner 允许使用 **Report Registered Connections** 任务查看已编译设计中节点的已寄存扇入和扇出。该报告与 **Generate Fanin/Fanout connections** 报告的不同在于，显示无连接线的源和目标，可能导致视图模糊不清。

1. Chip Planner 中，选择 1 个或多个节点。
2. **Task** 窗格中，双击 **Report Registered Connections**。
3. 从对话框选择选项，并单击 **OK**。

Reports 窗格显示已寄存的源和目标节点。可开启或关闭图形视图的可见性。

图 41. 报告已寄存的连接



相关链接

- 查看已布局资源的扇入和扇出连接 (第 103 页)
- Expand Connections Command (View Menu)
  Intel Quartus Prime Help 中

6.1.3.8. 查看已布局资源的扇入和扇出连接

显示扇入到或扇出至某资源的原子，包括连接线。

要显示所选资源的扇入或扇出连接，

1. 在 Chip Planner 工具栏中，单击 **Generate Fan-In Connections** 图标或 **Generate Fan-Out Connections** 图标。
2. 删除“Chip Planner”视图上显示的其他连接，请单击 **Clear Unselected Connections** 图标。

也可从 Chip Planner 的 **View** 菜单执行该操作。

相关链接

- 查看已布局节点的源和目标 (第 102 页)
- Expand Connections Command (View Menu)
  Intel Quartus Prime Help 中

6.1.3.9. 查看即时扇入和扇出连接

显示所选原子的即时扇入或扇出连接。

例如，通过查看即时扇入获取逻辑资源时，可看到驱动逻辑资源的布线资源。可为所有逻辑资源和布线资源生成即时扇入或扇出。

- 显示即时扇入或扇出连接，单击 **View** > **Generate Immediate Fan-In Connections** 或 **View** > **Generate Immediate Fan-Out Connections**。
- 删除显示中的连接，请使用 Chip Planner 工具栏中的 **Clear Unselected Connections** 图标。

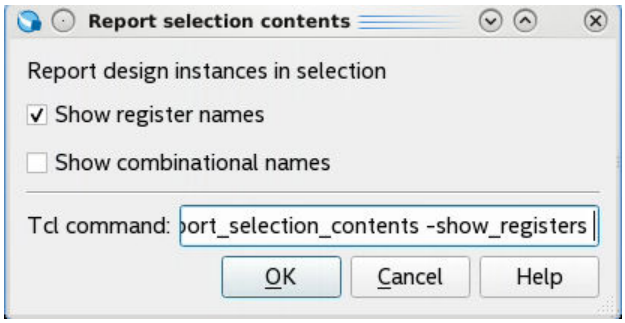
6.1.3.10. 查看已选内容

可在 Chip Planner 中查看任何所选区域内容的详细报告。查看已选区域的内容时，Chip Planner 针对选择部分中的设计单元生成一个分层，彩色编码列表。该功能支持快速确定 Compiler 放置设计指定模块的位置。

请按照如下步骤在 Chip Planner 中查看已选定内容：

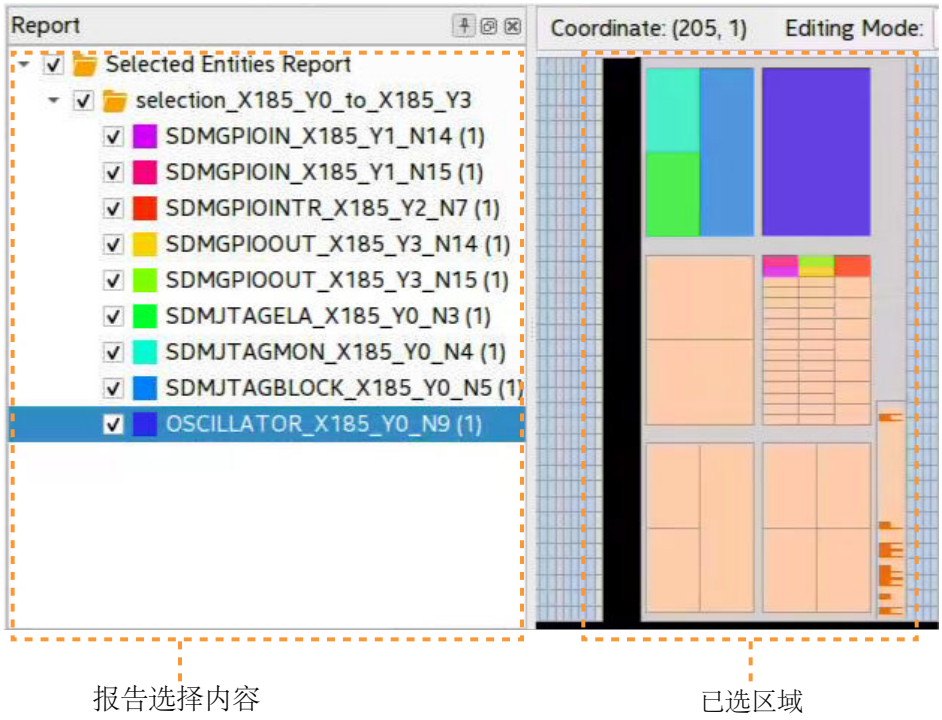
1. 在 **Tasks** 窗格中，双击 **Report Selection Contents**。**Report Selection Contents** 对话框显现。
2. 在 **Report design instances in selection** 下，开启或关闭 **Show registers names** 和 **Show combinational names** 显示报告中这些类型的名称。

图 42. Report Selection Contents 对话框



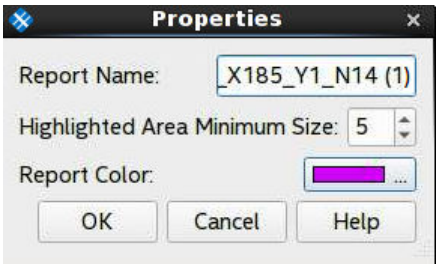
3. 点击 **OK**。该报告在 **Reports** 窗格中生成并显示所选单元的列表。

图 43. 查看已选内容



4. 要定制报告文件夹的颜色编码，右键单击任意报告，然后单击 **Properties**。可自定义报告的 **Report Name**、**Report Color** 和 **Highlighted Area Minimum Size**。

图 44. 已选定实体报告的属性



6.1.4. 在 Chip Planner 中管理路径

使用 Chip Planner 管理逻辑单元之间的路径。以下实例使用 Chip Planner 从 Timing Analysis 报告详细研究路径。

6.1.4.1. 分析路径的连接

在 Chip Planner 中确定所选路径或连接的组成单元，请在 Chip Planner 工具栏单击 **Expand Connections** 图标。

相关链接

Expand Connections Command (View Menu)
Intel Quartus Prime Help 中

6.1.4.2. 找到从 Timing Analysis Report 到 Chip Planner 的路径

要周到从 Timing Analysis 报告到 Chip Planner 的路径，请执行以下步骤：

1. 在“Timing Analysis 报告”中选择要查找的路径。
2. 右键点击路径并指向 **Locate Path** > **Locate in Chip Planner**。
   该路径将出现在 Chip Planer 的 **Locate History** 窗口中。

图 45. Locate History Window 中的路径列表

相关链接

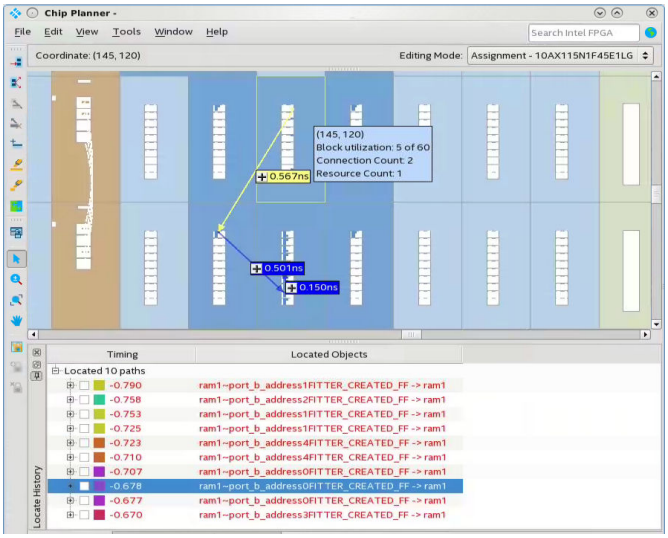
使用 Timing Analyzer 显示 Path Report (第 67 页)

6.1.4.3. 显示延迟

通过 **Show Delays** 功能，可查看 Timing Analyzer 报告中出现的路径的时序延迟。要访问该功能，在主菜单中单击 **View** > **Show Delays**。或 Chip Planner 工具栏中单击“Show Delays ”图标。要查看所选路径上的局部延迟，单击 **Locate History** 窗口中已显示路径旁的“+”号。

例如，可查看两个逻辑资源间的延迟或逻辑资源和布线资源之间的延迟。

图 46. 显示与 **Timing Analyzer Path** 相关的延迟



6.1.4.4. 查看布线资源

通过 Chip Planner 和 **Locate History** 窗口，可查看路径或连接使用的布线资源。还可选择显示“Arrival Data”路径和“Arrival Clock”路径。

图 47. 显示物理布线

Locate History 窗口中，右键点击路径选择 **Show Physical Routing** 显示物理路径。要调整显示，单击右键并选择 **Zoom to Selection**。

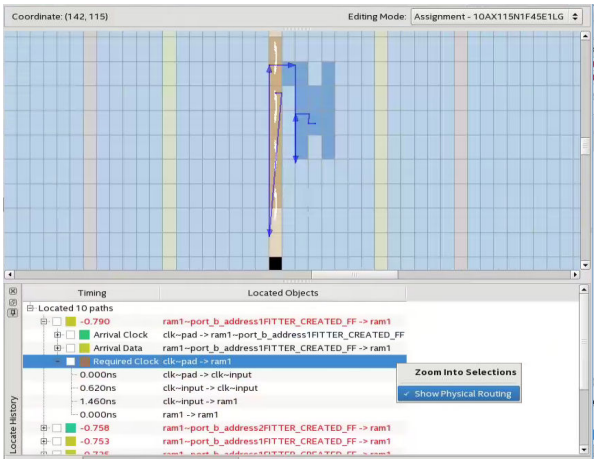
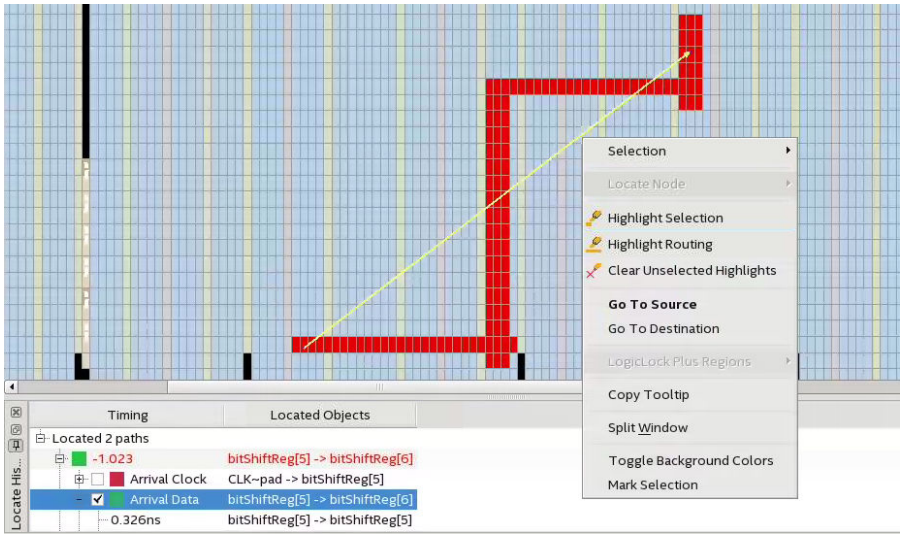


图 48. 突出显示布线

要查看 Fitter 布线路径的行和列，右键单击路径并选择 **Highlight Routing**（突出显示布线）。



相关链接

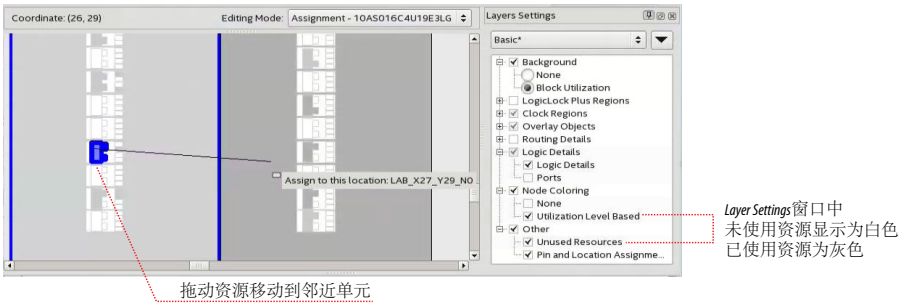
查看布线拥塞 (第 101 页)

6.1.5. 在 Chip Planner 中查看约束

可 选择适当层次，或显示块使用情况的任何定制预设在 Chip Planner 中查看位置约束。

Chip Planner 以预定义颜色显示已约束资源（默认为灰色）。

图 49. 在 Chip Planner 中查看约束



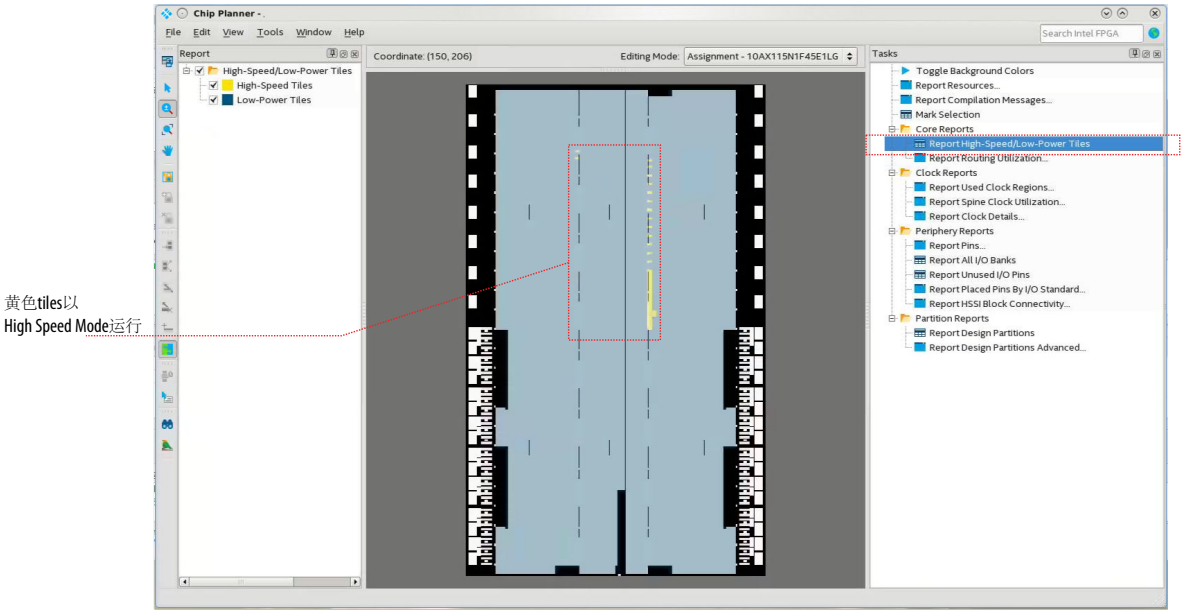
创建或移动约束，或对 Logic Lock 区域进行节点和管脚位置约束，可将已选资源拖动到新的位置。Fitter 会应用您在下个布局布线操作期间创建的约束。

6.1.6. 在 Chip Planner 中查看高速和低功耗 Tile

一些 Intel 器件具有可在高速模式和低功耗模式中运行的 ALM。适配处理期间可在 Intel Quartus Prime 软件中设置节能模式。将 ALM 组合在一起形成较大的块，称为“tiles”。

查看功耗映射，请在运行 Fitter 后双击 **Tasks** > **Core Reports** > **Report High-Speed/Low-Power Tiles**。Chip Planner 以对比色显示低功耗和高速 tiles；黄色 tiles 以高速模式运行，而蓝色 tiles 以低功耗模式运行。

图 50. **查看 Intel Arria 10 器件中的高速和低功耗 Tile**



6.2. 使用 Design Partition Planner 和 Chip Planner 创建分区和 Logic Lock 区域。

当 Fitter 在设计的其他部分运行时，使用带有设计分区的 Logic Lock 区域可保留块的位置。Design Partition Planner 和 Chip Planner 一起使用时，可创建分区和 Logic Lock 区域一定程度上利于实体的连接性和物理位置。

在 Intel Quartus Prime Pro Edition 设计中使用该技术：

1. 编译设计。
2. 打开 Chip Planner 和 Design Partition Planner。
   - 单击 **Tools** > **Chip Planner**
   - 单击 **Tools** > **Design Partition Planner**
3. **Chip Planner** 窗口中，前往 **Tasks** 窗格，并双击 **Report Design Partitions**。

   Report Design Partitions 任务使得 Chip Planner 显示设计实体的物理位置，且使用的颜色与 Design Partition Planner 中实体的显示色相同。
4. Chip Planner 中，单击 **View** > **Bird's Eye View**

   Bird's Eye View 打开。
5. 在 Design Partition Planner 中，将所有较大实体从其父级中拖出。

   或者，可右键点击实体并单击 **Extract from Parent**。

Chip Planner 显示 Design Partition Planner 中出现的实体的物理布局，且两个工具间的着色一致。可在 Chip Planner 中查看物理布局和在 Design Partition Planner 中查看连接性。

6. 确认不适合放置到 Logic Lock 区域的实体：
   - Chip Planner 显示要被物理性打散到器件的非连续区域的实体。
   - Design Partition Planner 显示与其他实体之间具有大量连接的实体。
7. 将不适于放置到 Logic Lock 区域的实体返回到其父级。

   或者，可右键点击实体并单击 **Collapse to Parent**。
8. 右键单击实体，然后单击 **Create Design Partition** 为每个剩余实体创建一个分区。
9. 右键单击分区，然后单击 **Create Logic Lock Region** 为每个分区创建一个 Logic Lock 区域。

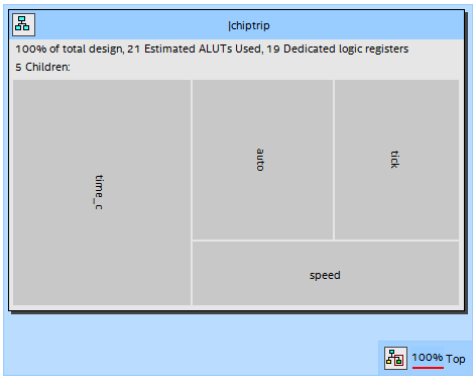
相关链接

规划设计分区
In *Intel Quartus Prime Pro Edition 用户指南：基于块的设计*

6.2.1. 查看设计连接性和层次结构

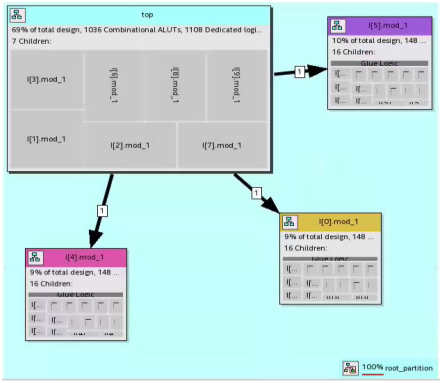
默认情况下，打开已编译设计时，Design Partition Planner 将设计显示为单个顶层实体，其中包含底层实体。如果已打开，设计显示为其最后的状态。

图 51. Design Partition Planner 中的顶层实体



- 要显示实体间的连接性，可将顶层实体拖动到周围空白处，或右键点击实体并单击快捷菜单上的 **Extract from Parent**，以从顶层实体中抽取实体。
  抽取实体时，Design Partition Planner 绘制实体间的连接束，并显示实体对之间的连接数。

图 52. 分区设计的连接束



- 要定制连接束外观或设置连接计数的阈值，请点击 **View** > **Bundle Configuration**，并在 **Bundle Configuration** 对话框中设置必要的选项。
- 查看包含失败路径的连接束，请打开 Timing Analyzer，然后在 Design Partition Planner 中单击 **View** > **Show Timing Data**。包含失败路径的连接束，以及具有节点驻留在失败路径上的实体均以红色显示。
- 查看束中连接的详细信息，请右键单击连接束然后单击 **Bundle Properties** 以打开 **Bundle Properties** 对话框。
- 要在连接性显示模式和层次显示模式间切换，可单击 **View** > **Hierarchy Display**。或者，在任意实体的左上角点击并按住层次图标可临时切换到层次显示模式。

6.3. 在 Chip Planner 中使用 Logic Lock 区域

可在 Chip Planner 中轻松创建 Logic Lock 区域并为其约束资源。

6.3.1. 在 Chip Planner 中查看 Logic Lock 区域之间的连接

可使用 Chip Planner 查看并编辑 Logic Lock 区域。要查看并编辑 Logic Lock 区域，请使用 **Layers Settings** 窗口中的 **Floorplan Editing**，或任何已启用 **User-assigned Logic Lock regions** 设置的层次设置模式。

Chip Planner 显示 Logic Lock 区域之间的连接。默认情况下，可将每个连接作为单个线路查看。可选择将两个 Logic Lock 区域之间的连接显示为单个绑定连接而非单条连接线路。要使用该选项，请打开 Chip Planner 并在 View 菜单上，单击 **Inter-region Bundles**。

相关链接

Inter-region Bundles 对话框
关于 Inter-region Bundles 对话框的更多信息，请参阅 Intel Quartus Prime Help。

6.3.3. Logic Lock 区域

Logic Lock 区域是布局规划位置约束。When you assign instances or nodes to a 为 Logic Lock 区域进行实例和节点约束时，指示 Fitter 将这些实例和节点放置在该区域中。一个布局规划可包含多个 Logic Lock 区域。

注意: 最佳实践是，使用迭代设计流程定义资源布局。设置硬布局约束之前，可使用例如“Early Place Flow”之类的技术指导平面布局规划决策。

Logic Lock 区域无保留属性，仅有边界和逻辑资源保留。可使用 Intel Quartus Prime Pro Edition 软件实现全层次 Logic Lock 区域约束。

Logic Lock 区域由两个组件组成：

- Placement Region（布局区域）：将逻辑约束到器件的特定范围；Fitter 将逻辑放置到您指定的区域。如果将某区域指定为“Reserved”（保留），则 Fitter 无法在该区域中放置其他逻辑。
- Routing Region（布线区域）：将布线约束到特定范围。默认情况下，布线区域不受约束。布线区域必须包含布局区域。不可保留布线区域。更多详细信息，请参阅 *Defining Routing Regions*。

相关链接

- 早期布局流程
  Intel Quartus Prime Pro Edition 用户指南：编译器
- 创建 Logic Lock 区域 (第 113 页)

6.3.4. Logic Lock 区域的属性

以下表格列出 Logic Lock 区域的属性。 Intel Quartus Prime 软件中，Logic Lock Regions 窗口显示设计中所有 Logic Lock 区域的属性。

表 21. Logic Lock 区域属性

名称	值	行为
Width	列数	指定 Logic Lock 区域的宽度。 如果 **Size/State** 设置为 **Auto/Floating**，则属性设置为 **Undetermined**。
Height	行数	指定 Logic Lock 区域的宽度。 如果 **Size/State** 设置为 **Auto/Floating**，则属性设置为 **Undetermined**。
Origin	任何布局规划位置	指定 Logic Lock 区域在布局规划中的位置。原本位于 Logic Lock 区域的左下角。
Reserved	Off | On	防止 Fitter 在该区域中放置其他逻辑。可对布线区域应用 **Reserved** 约束。
Core-Only	Off | On	排除区域中的外设资源。与 Intel Quartus Prime Standard Edition 软件不同，默认情况下 Intel Quartus Prime Pro Edition 区域约束应用于外设资源。如果将区域指定为 **Reserved** 和 **Core Only**，则不会从该区域中保留外设资源。
Size/State	Fixed/Locked | Auto/Floating	指定由您或 Fitter 来确定 Logic Lock 区域的尺寸和布局。

继续...

名称	值	行为
		• 如果设置为默认值“**Fixed/Locked**”，则由您定义 Logic Lock 区域的大小和布局。 • 如果设置为 **Auto/Floating**，则由 Fitter 确定 Logic Lock 区域的大小和布局。
Routing Region	Unconstrained | Whole Chip | Fixed with Expansion | Custom	布线区域的类型。更多详细信息，请参阅 *Defining Routing Regions*。

相关链接

Logic Lock 区域窗口 (第 121 页)

6.3.5. Intel Quartus Prime Standard Edition 和 Intel Quartus Prime Pro Edition 间的约束移植

Intel Quartus Prime Pro Edition 软件不支持 Intel Quartus Prime Standard Edition Logic Lock（标准版）约束。因此，如果将设计从 Intel Quartus Prime Standard Edition 移植到 Intel Quartus Prime Pro Edition，就必须将 Logic Lock (Standard)约束转换成 Logic Lock 约束。

相关链接

Replace Logic Lock Regions
In *Intel Quartus Prime Pro Edition User Guide: Getting Started*

6.3.6. 创建 Logic Lock 区域

6.3.6.1. 使用 Chip Planner 创建 Logic Lock 区域

1. 点击 **View** > **Logic Lock Regions** > **Create Logic Lock Region**
2. 单击并在 Chip Planner 布局规划图上拖动以创建首选位置和大小的区域。

创建区域后，可定义区域的形状，随后为区域约束单个实体。约束实体或定义形状无先后顺序。

6.3.6.2. 使用 Project Navigator 创建 Logic Lock 区域

1. 对设计进行完整编译或分析和详细说明。
2. 如果 Project Navigator 未开启，则单击 **View** > **Utility Windows** > **Project Navigator**。Project Navigator 显示设计的层次。
3. 随着设计层次完全展开，右键单击任意设计实例，并单击 **Create New Logic Lock 区域**。
4. 对新区域约束实例。

新区域的名称与实体名称相同。

6.3.6.3. 使用 Logic Lock Regions Window 创建 Logic Lock 区域

1. 点击 **Assignments** > **Logic Lock Regions Window**。
2. **Logic Lock Regions** 窗口中，点击**<<new>>**。

创建区域后，可定义区域的形状，随后为区域约束单个实体。约束实体或定义形状无先后顺序。

相关链接

Logic Lock 区域窗口 (第 121 页)

6.3.6.4. 定义布线区域

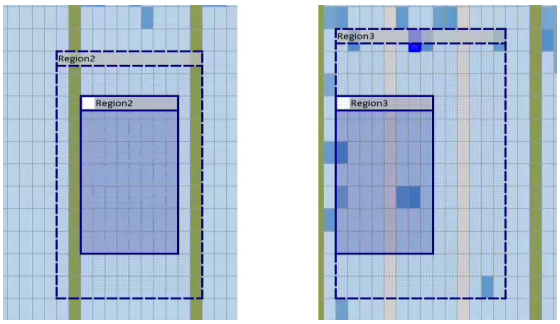
布线区域是 Logic Lock 区域中指定布线区域的元素。不限区域必须包含现有 Logic Lock 布局区域。布线区域不可设置为“reserved”（保留）。要定义布线区域，在 **Logic Lock Regions** 窗口双击 **Routing Region** 单元，并从下拉菜单选择一个选项。

有效的布线区域选项为：

表 22. 布线区域选项

选项	说明
Unconstrained（默认）	允许适配器使用器件中任何可用路线。
Whole Chip	与 Unconstrained 相同，但会在 Intel Quartus Prime 设置文件（.qsf）中写入约束。
Fixed with Expansion	按照布局区域的轮廓。布线区域按大于布局区域的行数/列数进行缩放。
Custom	允许在 Logic Lock 区域周围自定义布线区域形状。选择 **Custom** 选项时，布局和布线区域在 Chip Planner 中独立移动。这中情况下，使用 Shift 键选择移动布局和布线区域。

图 53. Routing Regions



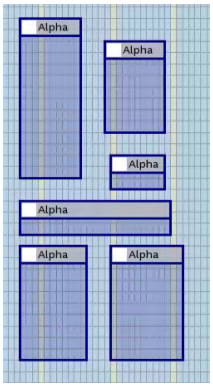
Fixed with Expansion +2

Custom Routing

6.3.6.5. 非连续 Logic Lock 区域

使用 Logic Lock 区域操控工具创建不相交的区域。非连续区域作为所有 Logic Lock 区域属性中的单个 Logic Lock 区域运行。

图 54. 非连续 Logic Lock 区域



相关链接

Merging Logic Lock Regions (第 116 页)

6.3.6.6. 使用 Auto Sized Region 的考量

如果使用 **Auto/Floating** Size/State Logic Lock 区域，请考虑：

- **Auto/Floating** 区域无法保留
- 验证您的 Logic Lock 区域不为空。如果为对区域约束任何实例，则 Fitter 会将尺寸减小为 0 x 0，使得该区域无效。
- 区域可能与分区相关联，也可能无关联。当分区和 **Auto/Floating** Size/State Logic Lock 区域组合时，可灵活解决特定适配挑战。然而，每添加一个约束都会减少可用的解决方案，过多约束会导致 Fitter 无法找到解决方案。具体示例如下：
  - 如果分区在综合期间被保留或未被保留，则 Logic Lock 区域将逻辑限制与特定区域，同时允许 Fitter 优化分区内的逻辑，以及优化 Logic Lock 分区内的布局。
  - 如果在布局，布线或最终阶段保留分区；Logic Lock 不是有效布局边界，因为分区逻辑的位置已固定。
  - 然而，如果 Logic Lock 区域被保留，Fitter 避免将其他逻辑置于本区域中，有助于降低资源拥塞。
- Logic Lock 区域的设置结果满足规格后，可进行如下操作：
  - 将 Logic Lock 区域转换成 **Fixed/Locked** Size/State。
  - 保留 Logic Lock 区域属性为 **Auto/Floating** Size/State 并将区域用作“keep together”类型的功能。
  - 如果 Logic Lock 区域也是一个分区，则可通过分区保留布局和布线并完整删除 Logic Lock 区域。

6.3.7. 定制 Logic Lock 区域的形状

创建定制形状的 Logic Lock 区域。可执行逻辑操作。非矩形 Logic Lock 区域可帮助排除某些资源，或将设计的某些部分放置于指定器件资源周围以提高性能。

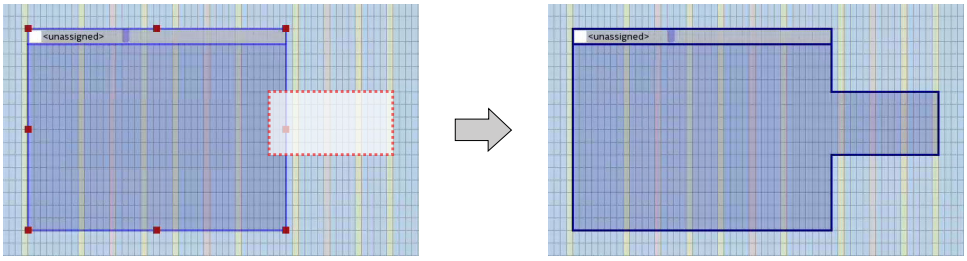
注意: 17.1 版本中无 Logic Lock 形状撤销功能。

6.3.7.1. 在 Logic Lock 区域中添加新形状

要在现有 Logic Lock 区域中添加新形状，可在 Chip Planner 中执行如下步骤：

1. 选择 Logic Lock 区域。
2. 在 **Navigation** 工具栏中，点击 **Add Logic Lock Region** icon 。
3. 单击并通过拖动来生成需要添加的形状。新形状将自动合并到所选 Logic Lock 区域。
   注意: 如果选择多了个区域，则该操作会将新形状添加到所有已选区域。

图 55. 使用 Add Logic Lock Region 功能

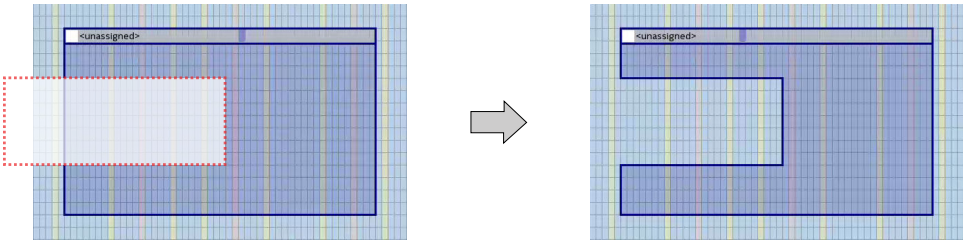


6.3.7.2. 从 Logic Lock 区域中删除形状

要在现有 Logic Lock 区域中移除某个形状，可在 Chip Planner 中执行如下步骤：

1. 选择 Logic Lock 区域。
2. 在 **Navigation** 工具栏中，点击 **Subtract Logic Lock Region** icon 。
3. 点击并拖动需要移除的形状。自动显示修改后的区域。
   该操作在所有已选区域中执行。

图 56. 使用 Subtract Logic Lock Region 功能

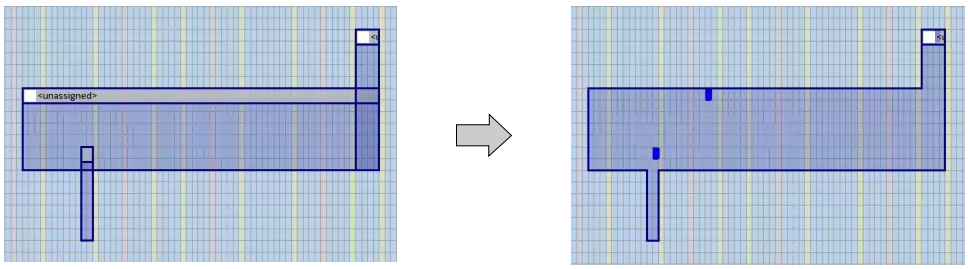


6.3.7.3. Merging Logic Lock Regions

合并 2 个或多个 Logic Lock 区域，可执行如下步骤：

1. 确保您要合并的区域中，具有逻辑约束的区域不能多于一个。
2. 将该区域排列到需要实现的区域处。
3. 按住 Shift 键，并单击选择所有需要合并的单个区域。
4. 右键单击任何已选择的 Logic Lock 区域并选择 **Logic Lock Regions** > **Merge Logic Lock Region**。所选的单个区域合并后创建为新的单个区域。
   如果选择被多次命名的区域，则 **Merge Logic Lock Region** 选项被禁用。

图 57. 使用合并 Logic Lock 区域命令



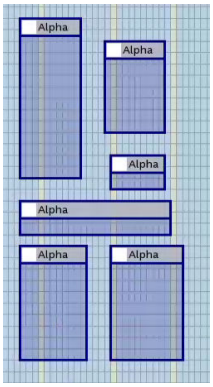
相关链接

创建 Logic Lock 区域 (第 113 页)

6.3.7.4. 非连续 Logic Lock 区域

使用 Logic Lock 区域操控工具创建不相交的区域。非连续区域作为所有 Logic Lock 区域属性中的单个 Logic Lock 区域运行。

图 58. 非连续 Logic Lock 区域



相关链接

Merging Logic Lock Regions (第 116 页)

6.3.8. 将器件资源放入 Logic Lock 区域

仅可将实体约束给设计中的一个 Logic Lock 区域，但该实体可按层次集成区域。该层及结构允许保留区域具有子区域且不保留子区域中的资源。

如果一个 Logic Lock 区域边界包含部分器件资源， Intel Quartus Prime 软件将整个资源约束给该 Logic Lock 区域。

使用 **Logic Lock Region** 窗口添加实例，右键点击该区域并选择 **Logic LockProperties** > **Add**，或者，在 Intel Quartus Prime 软件中，将实体从 Hierarchy 查看器拖动到 Logic Lock Regions Window 的 Logic Lock 区域名字段中。

6.3.8.1. 空白 Logic Lock 区域

Intel Quartus Prime 允许 Logic Lock 区域中无内容。“Empty regions”是管理 FPGA 空间用于日后所需逻辑的工具。该技术仅在将区域设置为 **Reserved** 时有效。

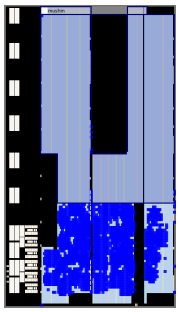
使用空白 Logic Lock 区域的原因：

- 初步平面布局规划。
- 复杂增量构建。
- 基于团队的设计和互连逻辑。
- 限制逻辑布局。

由于 Logic Lock 区域未保留任何布线资源，Fitter 可能将该区域用于布线目的。

将空白 Logic Lock 区域设置为 **Core Only** 属性。当空白区域中包含外设资源时，可限制外设组件布局，从而可能导致设计无法适配。命名空白区域后，可执行与任何已填充 Logic Lock 区域相同的操控。

图 59. 放置于空白区域外的逻辑



该图显示为其周围的 empty（空白） Logic Lock 区域和逻辑。但某些 IO，HSSIO 和 PLL 位于 empty 区域中。出现这样的布局是因为输出端口连接到 IO，且此 IO 始终为 root_partition（top-level partition）的一部分。

6.3.8.2. 管脚约束

Logic Lock 区域将所有器件资源包含于其边界中，包括存储器和管脚。将实例分配到区域约束后，Intel Quartus Prime Pro Edition 软件不会自动包含管脚，除非 **Core Only** 属性为“off”。

可手动将管脚约束到 Logic Lock 区域；然而，此布局对区域施加了位置限制。软件仅遵守与器件外设交界的锁定区域的管脚约束。锁定区域必须包含 I/O 管脚作为资源。

6.3.8.3. 保留 Logic Lock 区域

Reserved 属性指示 Fitter 仅对您指定约束到 Logic Lock 区域（Logic Lockregion）中的实体和节点进行布局。

Intel Quartus Prime 软件实现对 Logic Lock 区域的所有实体和节点约束。有时实体和节点并不占用整个区域，从而使得某些区域资源保持空闲。

要提高区域资源利用率和性能， Intel Quartus Prime 软件默认将仍未约束给另一区域的节点和实体填充为占用的空闲资源。要防止出现该行为，可在 **Logic Lock Regions** 窗口 中开启 **Reserved**。

6.3.8.4. 虚拟管脚

虚拟管脚是编译期间 Compiler 临时映射到逻辑单元而非管脚的 I/O 元件。软件将虚拟管脚实现为 LUT。要约束 Virtual Pin，可使用 Assignment Editor。通过将 **Virtual Pin** 约束到 I/O 元件来创建虚拟管脚。

将 **Virtual Pin** 约束应用到输入管脚时，该管脚不再显示为 FPGA 管脚；Compiler 将虚拟管脚固定到设计的 GND 中。虚拟管脚不是浮动节点。

仅将虚拟管脚用于底层设计实体中的 I/O 元件，从而在实体导入设计后成为节点；例如，编译部分设计时。

注意: **Virtual Pin** 逻辑选项必须约束到输入管脚或输出管脚。如果将该选项约束到双向管脚，三态管脚或已寄存的 I/O 元件，则 Synthesis 会忽略该约束。如果将该选项约束到三态管脚，则 Fitter 插入一个 I/O 缓冲作为三态逻辑；因此，该管脚不能是虚拟管脚。如果要继续将所约束管脚用作虚拟管脚，则可使用多路复用器代替三态管脚。除了直接连接器件 I/O 管脚的信号，请勿使用三态逻辑。

在顶层设计中，将这些虚拟管脚连接到另一模块的内部节点。通过对虚拟管脚的约束，可按照顶层模块中对应内部节点位置，将这些管脚放置在器件中相同的地方和区域。编译带有多于目标器件所允许管脚数的 Logic Lock 模块时，则可使用 **Virtual Pin** 选项。将 **Virtual Pin** 选项集成到顶层设计后，该选项可使能设计模块的时序分析并更加紧密匹配模块性能。

要通过 Node Finder 显示设计中所有已约束的虚拟管脚，可将 **Filter Type** 设置为 **Pins: Virtual**。要从 Assignment Editor 访问 Node Finder，双击 **To** 字段；当箭头出现在字段右侧时，点选 **Node Finder**。

相关链接

- 通过 Tcl 命令约束虚拟管脚 (第 127 页)
- Node Finder Command（查看 Menu）
  Intel Quartus Prime Help 中

6.3.8.5. 实例： Intel Arria 10 FPGA 的最佳布局实践

Logic Lock 区域必须考虑器件拓扑。

注意: 最佳实践是，使用迭代设计流程定义资源布局。设置硬布局约束之前，可使用例如“Early Place Flow”之类的技术指导平面布局规划决策。

本实例介绍 I/O Column 约束在针对 Intel Arria 10 FPGA 设计的 Logic Lock 区域中的位置。

图 60. Intel Arria 10 FPGA 中的 I/O Column

Intel Arria 10 FPGA 中的 I/O 列（Column）位于器件的中间位置。信号仅能从面向器件边缘的列的侧面进出。

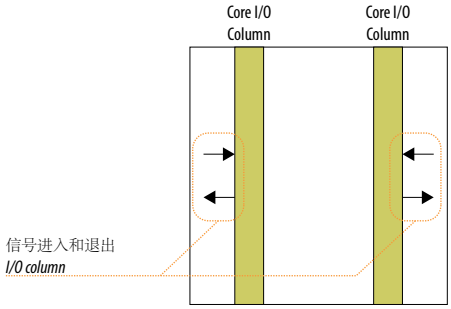


图 61. Intel Arria 10 FPGA 中跨 I/O Column 的信号

为了跨 I/O 列而路由的信号会增加布线延迟，且减低设计性能。

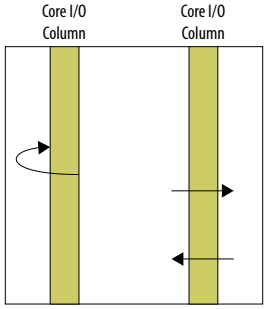
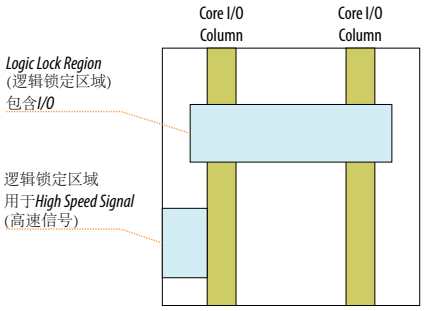


图 62. 关于 Intel Arria 10 FPGA 中 Logic Lock Region 的策略性布局

- 如果 Logic Lock 区域包含与 I/O 列对接的寄存器，对该 Logic Lock 区域进行布局，以便其覆盖 I/O 列和内核逻辑，从而更好地访问与外部列边缘相邻的 I/O 列。
- 对于高速信号，如果将 Logic Lock 区域放置到 I/O 列外部，则可获得最佳结果，因为适配器不太可能跨越列并引起延迟。



相关链接

- 早期布局流程
  Intel Quartus Prime Pro Edition 用户指南：编译器
- 布局规划局部局部重配置设计
  In *Intel Quartus Prime Pro Edition 用户指南：局部重新配置*

6.3.9. 层次型区域

Logic Lock 区域为全分层型。Parent（父级）区域必须完全包含所有 child（子级）区域。**Reserved** 和 **Core-Only** 约束也分层性应用。

Logic Lock 分配和其他约束以及分配具有相同的优先级。

仅可将实体约束给设计中的一个 Logic Lock 区域，而该实体可按层次结构继承区域。该层次结构允许保留区域具有子区域且不保留子区域中的资源。

6.3.10. 其他 Intel Quartus Prime Logic Lock 设计功能

为补充 **Logic Lock Regions Window**， Intel Quartus Prime 软件具有其他功能以帮助设计 Logic Lock 区域。

6.3.10.1. Intel Quartus Prime 修订功能

评估设计中的各 Logic Lock 区域时，可能希望尝试各种配置以达到所需结果。 Intel Quartus Prime Revisions（修订）功能允许通过不同设置组织同一工程直到直到找到最佳配置。

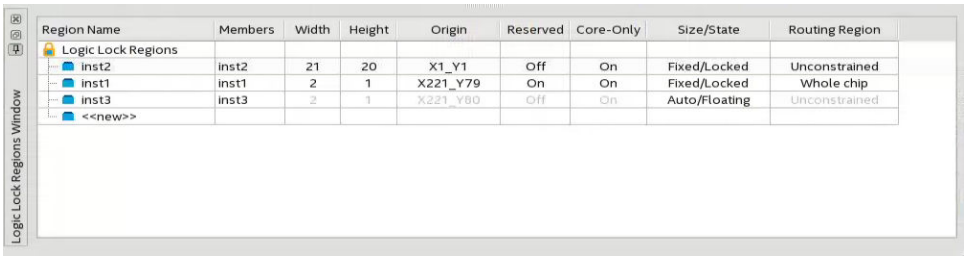
要使用 Revisions 功能，请选择 **Project** > **Revisions**。可根据当前设计或任何之前创建的版本创建修订。每个修订都有相关说明。可使用修订内容组织针对 Logic Lock 区域创建的布局约束。

6.3.11. Logic Lock 区域窗口

Logic Lock Regions Window（区域窗口）提供关于设计中定义的所有 Logic Lock 区域的摘要。可使用 Logic Lock Regions Window 创建，约束并修改 Logic Lock 区域的属性。

在 Chip Planner 中单击 **View** > **Logic Lock Window** 打开 Logic Lock Regions Window，而在 Intel Quartus Prime 中，请单击 **Assignments** > **Logic LockWindow**。

图 63. Logic Lock 区域窗口



可拖放“列”以更改列顺序来定制 Logic Lock Regions Window；也可右键点击任何“列”标题，然后在快捷菜单中适当选择要显示或隐藏的“列”。

Logic Lock Regions 属性对话框

使用 **Logic Lock Regions Properties** 对话框查看并修改 Logic Lock 区域详细信息，例如约束到区域的实体和节点，以及需要的资源。

要打开 **Logic Lock Regions Properties** 对话框，右键点击区域并选择 **Logic Lock Regions Properties...**。

相关链接

- Logic Lock 区域的属性 (第 112 页)
- 使用 Logic Lock Regions Window 创建 Logic Lock 区域 (第 113 页)
- Logic Lock 区域窗口
  Intel Quartus Prime Help 中

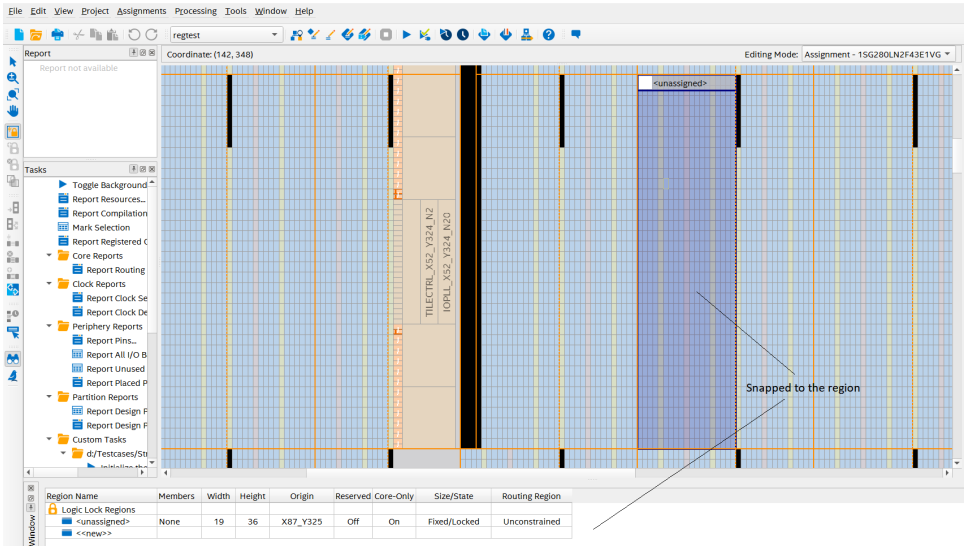
6.3.12. 插入区域（Snapping to a Region）

Chip Planner 支持 Logic Lock 区域插入逻辑阵列块（snap-to-lab），其中所创建 Logic Lock 区域总是被插入逻辑阵列块。该操作适用于 Intel Arria 10 中的时钟区域和 Intel Agilex® FPGA，以及 Intel Stratix 10 FPGA 中的时钟扇区。

默认情况下，Logic Lock 区域始终被插入 lab（逻辑阵列块）。可更改默认设置，点击 **View** > **Logic Lock Regions** > **Snap Logic Lock Region to**

注意: 如下图所示，在 Logic Lock 区域中执行如下操作，（如，创建，调整区域大小，或移动区域）或插入时钟区域（snap-to-clock-region）时，可查看时钟区域或扇区（具有橙色边界）。

图 64. 已插入区域



插入区域后，Logic Lock 区域边界显示为交互模式。并可观察如下行为：

- **Creating Region**：单击鼠标左键创建 Logic Lock 区域。释放鼠标后，所创建的 Logic Lock 区域插入包含的时钟区域或扇区。
- **Resize region (and resize diagonal)**：单击鼠标左键并拖动 Logic Lock 区域柄。释放鼠标后，Logic Lock 区域调整大小并插入包含的时钟区域或扇区。
- **Move region**：选择并拖动 Logic Lock 区域以高亮显示时钟区域边界。释放鼠标按键后，Logic Lock 区域移动到新的位置并插入包含的时钟区域或扇区。
  - **Same place and route regions are moved**（相同位置和布线区域被移动）：两个 Logic Lock 区域移动并插入包含的时钟扇区。
  - **Only place | route region is moved**（仅位置 | 布线区域被移动）：选定区域移动并插入时钟扇区，如果区域的新位置或大小不符合“place bboxes contained within route bboxes”（布线 bbox 中包含布局 bbox）规则，则会警告提示。
- **Subtract or make a hole**：在 snap-to-clock-region 模式中执行减法时，可在插入区域的时钟区域或扇区中创建一个区域，然后减去。

6.4. 在 Chip Planner 中使用用户定义时钟区域

可在 Chip Planner 中轻松创建和操作时钟区域并对区域分配时钟。

可创建用户定义时钟区域约束以确保给定全局时钟信号在将来全部设计迭代全过程中从器件特定区域获得资源。在涉及全局信号资源拥塞的实例中，可指定较小时钟区域约束以避免信号使用其他扇区中拥塞的时钟资源。

如果创建用户定义时钟区域（user-defined clock region）并随后编译设计，则这些用户定义时钟区域随后显示为 Fitter 定义的时钟区域，并无法再编辑。

User-Defined Clock Region 功能支持摘要

功能	时钟区域支持
时钟区域形状。	限于插入时钟扇区网格的矩形区域。
外设单元约束	限于为设计提供时钟的单元。
时钟区域名称。	通过为设计单元提供时钟的源进行识别。
对同一区域内多个实例的支持。	每个时钟设计单元创建一个区域，然后为多个时钟设计单元指定相同定义以分配给同一时钟区域。

在 Intel Stratix 10 和 Intel Agilex Devices 中使用 Clock Region Assignments

将时钟区域限制为矩形，且该矩形的尺寸由扇区网格定义，如 Chip Planner 中“Clock Sector Region”层所示。该矩形由其左下角和右上角坐标定义。例如，*SX0, SY0, SX1, SY1* 将时钟限制在扇区 0,0 的左下到扇区 1,1 右上的 2 × 2 区域。

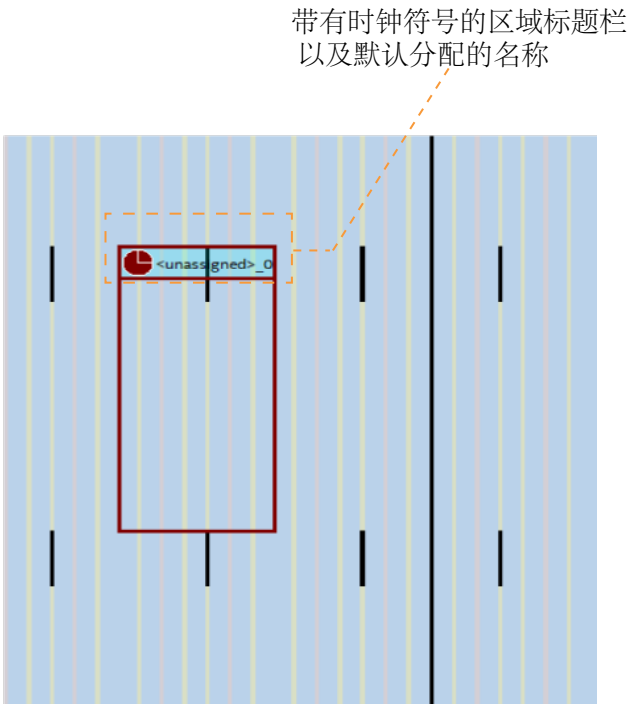
还可在芯片坐标中指定边界矩形，例如 *X37 Y181 X273 Y324*；但是，该约束应与扇区保持一致。Fitter 自动插入包含原始约束的最小扇区对齐矩形。

6.4.1. 通过 Chip Planner 创建 Clock Assignment

1. 选择 **Create Clock Assignment** 图标，或点击 **View** > **Clock Assignments** > **Create Clock Assignment**。
2. 点击并拖动 Chip Planner 平面图，以创建您时钟区域的首选位置和大小。您拖动的区域将插入能够容纳该区域的最小时钟扇区。交互时钟约束操作（例如创建，移动或调整大小）期间，时钟扇区栅格显示为橙色，以便于按照时钟扇区放置时钟区域。

区域标题栏中的时钟符号将其表示为时钟区域。默认情况下，新创建的时钟区域名为 *unassigned*，直到您从菜单中分配一个时钟信号。

图 65. 新创建的时钟区域



6.4.2. 调整 Clock Assignment

1. 选择时钟约束。区域各侧和转角处出现调整钮。
2. 将十字准线置于所选调整钮之上时，会出现调整大小的鼠标光标。
3. 按住鼠标左键并拖动调整大小的光标对时钟约束的边缘进行缩放。释放鼠标按钮后，时钟约束边界插入最近一个包含时钟扇区的栅格。

6.4.3. 移动 Clock Assignment

1. 选择时钟约束。
2. 将十字准线置于时钟约束标题栏之上，会出现用于移动的光标。
3. 按住鼠标左键并将时钟约束拖动到所需的新位置。

6.4.4. 删除 Clock Region Assignment

1. 选择需要删除的时钟约束。
2. 右键点击时钟标题栏约束显示工程相应菜单，或从主菜单栏选择 **View**。
3. 点击 **Clock Assignments** > **Delete Clock Assignment**。
4. 提示确认需要删除的已选时钟约束。点击 **Yes** 以确认删除。

指定时钟区域约束将从系统删除。

6.4.5. 将时钟信号分配给时钟区域

1. 右键点击时钟标题栏约束显示工程相应菜单，或从主菜单栏选择 **View**。
2. 点击 **Clock Assignments** > **Set Clock Signal Name**。
3. 在 **Set Clock Signal Name** 对话框中，浏览至所需时钟信号名称或直接键入所需时钟信号名称。
4. 点击 Ok。

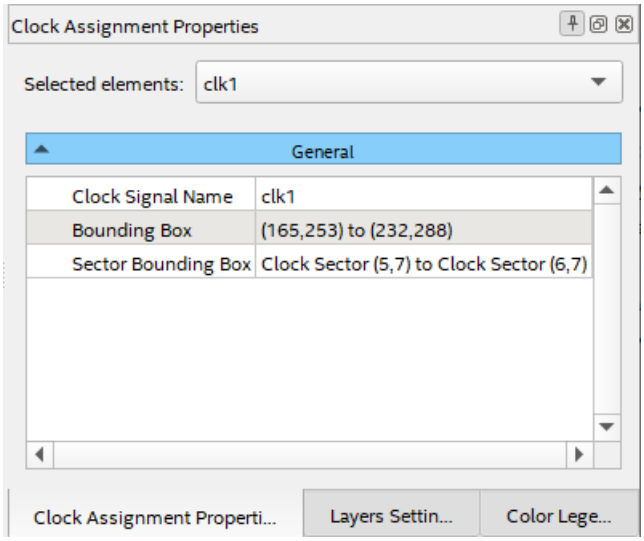
系统根据指定的时钟信号重命名时钟约束。

6.4.6. Clock Assignment 属性

Clock Assignment Properties 窗格显示所选时钟约束的属性。

默认情况下，Clock Assignment Properties 窗格出现在 Chip Planner 右侧的选项卡上。

图 66. Clock Assignment Properties 窗格



6.5. 脚本支持

可在 Tcl 脚本中运行本章介绍的处理过程并指定设置。还可在命令提示下运行某些处理过程。

相关链接

- Tcl Scripting
  Intel Quartus Prime Pro Edition 用户指南：脚本
- Command Line Scripting
  Intel Quartus Prime Pro Edition 用户指南：脚本

6.5.1. 通过 Tcl 命令创建 Logic Lock 约束

Intel Quartus Prime 软件支持 Tcl 命令创建或修改 Logic Lock 约束。

注意: 使用节点的完整层次结构路径指定节点名称。

创建或修改布局区域

可从 GUI 创建 Logic Lock 区域，或直接将区域添加到 QSF。QSF 条目包含顶点的 X/Y 坐标和 Placement Region 名称。

以下约束创建新的布局区域，其边界框坐标为 `X46 Y36 X65 Y49`：

```
set_instance_assignment -name PLACE_REGION "X46 Y36 X65 Y49" -to <node names>
```

- 可使用相同的命令格式修改现有约束。
- 要指定非矩形或不相交区域，请使用分号（`;`）作为 2 个或多个边界框之间的分隔符。
- 请使用多个 `PLACE_REGION` 实例约束将多个实例约束到同一区域。

创建或修改布线区域

以下约束创建布线区域，其边界框坐标为 `X5 Y5 X30 Y30`：

```
set_instance_assignment -name ROUTE_REGION -to <node names> "X5 Y5 X30 Y30"
```

- 可使用相同的命令格式修改现有约束。
- 所有具有布线区域约束的实例都必须具有各自的布局区域；布线区域必须完全包含布局区域。

将区域指定为“保留”

如下约束保留现有区域：

```
set_instance_assignment -name <instance name> RESERVE_PLACE_REGION -to <node
names> ON
```

- 仅可保留布局区域。

将区域指定为“Core Only”

默认情况下， Intel Quartus Prime Pro Edition 软件包含 Logic Lock 约束中的管脚。要将区域指定为“core only”（即，实例中的外设逻辑不受约束），请使用以下约束：

```
set_instance_assignment -name <instance name> CORE_ONLY_PLACE_REGION -to <node
names> ON
```

相关链接

创建 Logic Lock 区域 (第 113 页)

6.5.2. 通过 Tcl 命令约束虚拟管脚

使用以下 Tcl 命令开启名为 my_pin 的管脚的虚拟管脚设置：

```
set_instance_assignment -name VIRTUAL_PIN ON -to my_pin
```

相关链接

- 虚拟管脚 (第 119 页)
- Node Finder Command（查看 Menu）
  Intel Quartus Prime Help 中

6.5.3. Logic Lock 区域分配实例

这些实例显示了.qsf 文件中 Logic Lock 区域分配的句法。或者，还可在 Assignment Editor，Logic Lock Regions Window 或 Chip Planner 中输入这些分配。

实例-1: 分配矩形 Logic Lock 区域

将矩形 Logic Lock 区域约束到（10,10）的右下角位置，以及包括（20,20）的右上角位置。

```
set_instance_assignment -name PLACE_REGION -to a|b|c "X10 Y10 X20 Y20"
```

实例-2: 分配非矩形 Logic Lock 区域

将具有全层次结构路径“x|y|z”实例约束到非矩形的 L 状 Logic Lock 区域。软件将每组 4 个数字视为一个新的边界框。

```
set_instance_assignment -name PLACE_REGION -to x|y|z "X10 Y10 X20 Y50; X20 Y10
X50 Y20”
```

实例-3: 分配从属 Logic Lock 实例

默认情况下， Intel Quartus Prime 软件将每个子实例限制为其父级的 Logic Lock 区域。对子实例的任何约束都与其祖级的约束相交。例如，以下实例中，“a|b|c|d”下的所有逻辑都约束为边界框(10,10), (15,15)，而非(0,0), (15,15)。出现该结果是由于子约束与父约束相交。

```
set_instance_assignment -name PLACE_REGION -to a|b|c "X10 Y10 X20 Y20"
set_instance_assignment -name PLACE_REGION -to a|b|c|d "X0 Y0 X15 Y15"
```

实例-4: 分配多个 Logic Lock 实例

默认情况下，Logic Lock 区域约束允许来自其他实例的逻辑共享同一区域。这些分配将实例 c 和实例 g 布局到相同位置。如果实例 c 和实例 g 大量交互，则该策略很有用。

```
set_instance_assignment -name PLACE_REGION -to a|b|c "X10 Y10 X20 Y20"
set_instance_assignment -name PLACE_REGION -to e|f|g "X10 Y10 X20 Y20"
```

实例-5: 已分配的保留 Logic Lock 区域

可选择保留一个实例的完整 Logic Lock 区域或任何从级实例。

```
set_instance_assignment -name PLACE_REGION -to a|b|c "X10 Y10 X20 Y20"
set_instance_assignment -name RESERVE_PLACE_REGION -to a|b|c ON

# The following assignment causes an error. The logic in e|f|g is not
# legally placeable anywhere:
# set_instance_assignment -name PLACE_REGION -to e|f|g "X10 Y10 X20 Y20"

# The following assignment does *not* cause an error, but is effectively
# constrained to the box (20,10), (30,20), since the (10,10),(20,20) box is
reserved
# for a|b|c
set_instance_assignment -name PLACE_REGION -to e|f|g "X10 Y10 X30 Y20"
```

6.6. 分析和优化设计布局规划修订历史

以下修订历史适用于本章：

表 23. 文档修订历史

文档版本	Intel Quartus Prime 版本	修订内容
2019.07.30	19.3.0	添加了新的*在 Chip Planner 中使用用户指定时钟区域*部分。
2019.07.01	19.1.0	添加了新的“插入区域”主题以说明 **Snap Logic Lock Region to** 选项。
2019.04.01	19.1.0	• 添加了新的“Viewing Selected Contents”主题以介绍罗列所选设计单元的新报告。
2018.09.24	18.1.0	• 添加了主题：*查看时钟扇区利用率* • 添加了主题：*查看已布局节点的源和目标*。 • 重命名主题：将*生成扇入和扇出连接*更改为*查看已布局资源的扇入和扇出连接*。
2018.05.07	18.0.0	• 添加关于使用迭代方法进行布局规划的建议。
2017.11.06	17.1.0	• 将 *LogicLock Plus* 实例更改为 *Logic Lock*。 • 添加了对自动调整大小 Logic Lock 区域的支持 • 添加了对空 Logic Lock 区域的支持 • 添加主题：关于通过 Design Partition Planner 和 Chip Planner 使用 Auto Sized Regions（自动调整区域大小），Creating Partitions（创建分区）和 Logic Lock Region（逻辑锁定区域）的考量。
2017.05.08	17.0.0	• 章节结构重组和内容更新。 • 添加了图示：时钟区域，Locate History Window 中的 Path List（路径列表），显示物理布线，使用添加矩形功能，使用减除去矩形功能，LogicLock 区域中建孔，非连续 LogicLock 区域，布线区域，空白区域外的逻辑布局。 • 更新图示：HSSI Channel 块，突出显示布线，Arria 10 器件中的高速和低功耗 Tile，显示延迟高显布线，查案 Chip Planner 中的约束，LogicLock Plus 区域窗口，使用 Merge LogicLock Plus 区域命令。 • 创建了主题：*将矩形添加到 LogicLock Plus 区域*，*从 LogicLock Plus 区域减去矩形*。 • 移动了主题：将*查看关键路径*移动到*时序收敛和优化*章节并重命名为*关键路径*。 • 重命名主题：将*创建非矩形 LogicLock Plus 区域*重命名为*合并 LogicLock Plus 区域*。 • 重命名主题：将 *Chip Planner 概述*重命名为 *Chip Planner 中的设计布局规划分析*。 • 重命名章节，从*使用 Chip Planner 分析和优化设计布局规划*重命名为*分析和优化设计平面规划*。

继续...

文档版本	Intel Quartus Prime 版本	修订内容
2016.10.31	16.1.0	• 品牌更名为 Intel。 • 添加了介绍如何在 LogicLock Plus 区域中创建孔洞的主题。
2016.05.02	16.0.0	更新关于创建 LogicLock Plus 区域的信息。
2015.11.02	15.1.0	• 将 *Quartus II* 更改为 *Quartus Prime*。 • 添加有关如何使用 LogicLock 区域的信息。
2015.05.04	15.0.0	添加有关 LogicLock 区域颜色编码的信息。
2014.12.15	14.1.0	更新了 Virtual Pins 约束的说明，以阐明约束的输入不可用。
2014 年 6 月	14.0.0	更新文档格式
2013 年 11 月	13.1.0	删除了 HardCopy 器件信息。
2013 年 5 月	13.0.0	更新了“查看布线拥塞”部分 更新了针对 Chip Planner 的 Quartus UI 控制的参阅内容
2012 年 6 月	12.0.0	删除了反馈问卷链接。
2011 年 11 月	11.0.1	文档模板更新。
2011 年 5 月	11.0.0	• 更新了 11.0 发布。 编辑了”LogicLock 区域” 更新了“查看布线拥塞” 更新了“查找历史记录” 更新了图示 15-4、15-9、15-10 和 15-13 添加图示 15-6
2010 年 12 月	10.1.0	• 针对 10.1 发布而进行的更新。
2010 年 7 月	10.0.0	• 更新了器件支持信息。 • 删除了关于时序收敛布局规划的参阅内容；删除了“使用时序收敛布局规划进行设计分析”部分 • 添加了在线 Help 主题 • 添加了“通过 Design Partition Planner 使用 LogicLock 区域”部分

继续...

文档版本	Intel Quartus Prime 版本	修订内容
		• 更新了“查看关键路径”部分 • 更新了多个图形。 • 文档修订历史的格式更新
2009 年 11 月	9.1.0	• 所支持器件的信息整体更新 • 删除了关于不建议使用旧器件系列的 Timing Closure Floorplan 部分。（关于使用就器件系列的 Timing Closure Floorplan 的信息，请参阅文本存档中的早前版本 Quartus Prime 手册。） • 更新了“创建非矩形 LogicLock 区域”部分 • 添加了“Selected Elements Window”部分 • 更新了表 12-1。
2008 年 5 月	8.0.0	• 更新了以下部分： “Chip Planner 任务和层次” “LogicLock 区域” “反标 LogicLock 区域” “时序收敛布局规划中的 LogicLock 区域” • 添加了以下部分： “保留 LogicLock 区域” “创建非矩形 LogicLock 区域” “查看器件中可用的时钟网络” • 更新了表 10-1。 • 删除了如下部分 使用时序收敛布局规划保留 LogicLock 区域设计分析

相关链接

文档存档

关于之前版本的 *Intel Quartus Prime 手册*，请搜索文档存档。

7. 实现工程变更命令

在典型 FPGA 工程开发周期中，设计的可编程逻辑部分的规范经常会在设计过程中发生改变。即使已最终确定并完全编译设计， Intel Quartus Prime 软件还可支持最后一刻定案的设计更改（亦称作，工程变更命令（ECO））。

ECO 通常出现在设计周期的验证阶段。例如，验证期间才确定设计需要进行少量更改，例如更改网表连接，更正 LUT 中的逻辑错误，或调整 PLL 时钟频率。选择实现 ECO 而不对设计执行完全重新编译，只需花费很少时间且仅更改受影响的逻辑。

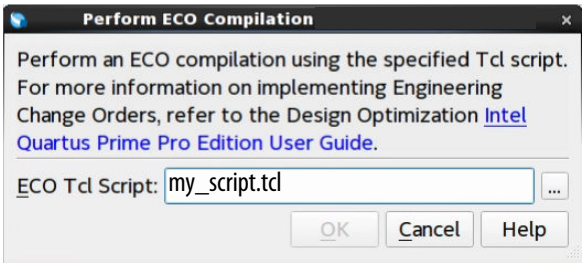
使用`::quartus::eco`包在 Tcl 脚本中指定 ECO 命令。

注意: Intel Quartus Prime Pro Edition 软件仅支持 Intel Stratix 10 和 Intel Agilex 器件使用 ECO。

7.1. 工程变更命令流程

1. 确定已编译设计中需要的 ECO 修改。
2. 确定 ECO 命令是否支持该更改，可通过查看 ECO 命令 (第 132 页)和 ECO 命令限制 (第 136 页)。
3. 创建 Tcl 脚本，如 ECO Tcl 脚本实例 (第 132 页)所示。
4. 运行 ECO 编译前，点击 **Project** > **Archive Project** 并将编译数据库和输出文件集存档。
5. 点击 **Processing** > **Start** > **Start ECO Compilation**。

图 67. 运行 ECO 编译



6. 指定 **ECO Tcl Script** 文件，并点击 **OK**。Fitter 处理 ECO 命令并更新最终的网表。如果错误指定任何命令，Fitter 生成错误。Fitter 处理完成后，应用更改。

   默认情况下，ECO 成功编译后，Assembler 随后自动运行以生成编程文件。要禁用流程中自动运行 Assembler，可关闭 Compilation Dashboard 上 Assembler 阶段复选框。
7. 在 post-fit 分析工具中查看 ECO 结果，例如 Timing Analyzer，Netlist Viewer 或 Chip Planner。

可使用如下命令行替代 GUI:

- 从 Tcl Console 调用 ECO 流程

```
execute_flow -eco <script>.tcl
```

默认情况下，Tcl Console ECO 编译成功后，Assembler 随后自动运行。要禁用 Assembler 自动运行，可将以下命令行添加到工程.qsf 文件：

```
set_global_assignment -name FLOW_DISABLE_ASSEMBLER ON
```

- 运行以下命令行命令： “

```
quartus_fit <project> -c <revision> --eco <script>.tcl
```

从该命令行运行时，任何有效 Intel Quartus Prime GUI 应用程序都不自动刷新。必须关闭并重新开启工程来刷新 GUI。

注意: 实现 ECO 以后，如果再次运行设计中的 Fitter，其会覆盖 ECO 更改。如果需要长久实现 ECO 更改，就必须更新 RTL 和 IP 布局并重新编译设计。

7.2. ECO Tcl 脚本实例

下面显示一个示范 ECO Tcl 脚本，该脚本修改 lutmask 并重新布线连接：

图 68. ECO Tcl 脚本实例

```
1  package require ::quartus::eco
2
3  # Modify the lutmask
4  post_message "About to modify lutmask"
5  modify_lutmask -to inst -eqn "A|B"
6
7  post_message "About to reroute connections"
8  remove_connection -from d1|l0 -to d1|ff0 -port D
9  make_connection -from one_and_inst|out -to d1|ff0 -port D
10
```

7.3. ECO 命令

Intel Quartus Prime Pro Edition 软件支持以下 ECO 命令：

make_connection (第 133 页)
remove_connection (第 133 页)
modify_lutmask (第 134 页)
adjust_pll_refclk (第 134 页)
modify_io_slew_rate (第 135 页)
modify_io_current_strength (第 135 页)
modify_io_delay_chain (第 135 页)

相关链接

ECO 命令限制 (第 136 页)

7.3.1. make_connection

说明

将源信号连接到目标块端口。如果端口中已有连接，则命令删除先前的连接并将其连接到您指定的信号。实际布的布线变更隐含于脚本结尾。可右键单击 Netlist Viewer 中的节点确定节点名称，然后点击 **Properties**。

用法

如下实例中将 `top|a_out` 连接到节点 `top|x` 的 D 输入端口。

```
make_connection -from top|a_out -to top|x -port D
```

自变量（Argument）

from 新连接的源数据块的输出网络。

to 目标块名称

port 目标块的输入端口名称。

选项

tieoff 明确将输入端口绑定到 VCC 或 GND。

VCC 或 GND

例如：

```
make_connection -tieoff VCC -to {node1} -port DATAA
```

to 目标块的名称

port 目标快的输入端口名称。

7.3.2. remove_connection

说明

断开目标块端口的 `src` 信号。实际布线变更隐含于脚本结尾。可右键单击 Netlist Viewer 中的节点确定节点名称，然后点击 **Properties**。

用法

以下实例从节点 `top|x` D 输入端口断开 `top|a_out`，并将 `top|x:D` 设置为断开状态。

```
remove_connection -from top|a_out -to top|x -port D
```

自变量（Argument）

from 当前连接的源块的输出网络。

to 目标块名称

port 目标块的输入端口名称。

7.3.3. modify_lutmask

说明

修改匹配目标节点的 lutmask，使用二进制或十六进制 lutmask 值（`-mask`），或使用由指定逻辑方程计算而来的等效 lutmask 值（`-eqn`）。

用法

以下实例从节点 `top|x` D 输入端口断开 `top|a_out`，并将 `top|x:D` 设置为断开状态。

```
modify_lutmask -to top|lut_c -eqn {a&b&c}
modify_lutmask -to top|lut_a -mask 0xFF00FF00
modify_lutmask -to top|lut_b -mask 0b111111111001010
```

自变量（Argument）

eqn 输入（`A, B, C, D, E, F`）的逻辑方程式。支持的词法标记包括 `AND('&')`、`OR('|')`、`XOR('^')`、`NOT('!')`、`OPEN_BRACE('(')`、`CLOSE_BRACE(')')`。指定`-mask` 或`-eqn`

to 目标原子名称。

mask 要以二进制或十六进制格式修改的 lutmask 值。指定`-mask` 或`-eqn`

注意: 在 Resource Property Viewer 中查看 lutmask 方程式时，该方程式以 `F0/F1/F2/F3` LUT 形式显示 A、B、C 和 D 输入。对于使用 E 或 F 输入的 LUT，必须根据 ALM 结构图针对 E 和 F 多路复用器显示的连接性组合子功能。

7.3.4. adjust_pll_refclk

说明

通过修改输入参考时钟频率更改 IOPLL 频率。适用于以下情况：

- 保持原始 refclk 和 outclk 比率。
- 您更改的 IOPLL 均不生成 IP 时钟。
- 级联的 IOPLL 必须直接连接（其相互间无时钟门）。
- IOPLL 不可处于“非专用”补偿模式。
- 对于所有 IOPLL，outclks 占空比等于 50，且相移等于 0。
- 不支持 Intel Agilex 器件。

用法：

以下实例通过将输入时钟频率修改到 100 MHz 来调整*pll_main* IOPLL。

```
adjust_pll_refclk -to {*pll_main*} -refclk 100
```

自变量（Argument）

to 您需要调整的上游 IOPLL。转义目标名称中的[or]。

refclk 新的 refclk 频率，以 MHz 为单位。

7.3.5. modify_io_slew_rate

说明

实现 I/O 管脚的指定 I/O 管脚偏斜设置率。

用法

```
modify_io_slew_rate 1 -to top|ipin
```

自变量（Argument）

to 需要修改的目标管脚实例名称。

7.3.6. modify_io_current_strength

说明

实现 I/O 管脚的指定 I/O 管脚偏斜设置率。

用法

```
modify_io_current_strength 3mA -to top|ipin
```

自变量（Argument）

to 需要修改的目标管脚实例名称。

7.3.7. modify_io_delay_chain

说明

实现 I/O 管脚的指定延迟链设置更改。

用法

```
modify_io_delay_chain 3 -to top|ipin -type input
```

自变量（Argument）

type 指定如下 I/O 类型：`input`, `output`, `oe`, `io_12_lane_input`, `io_12_lane_input_strobe`

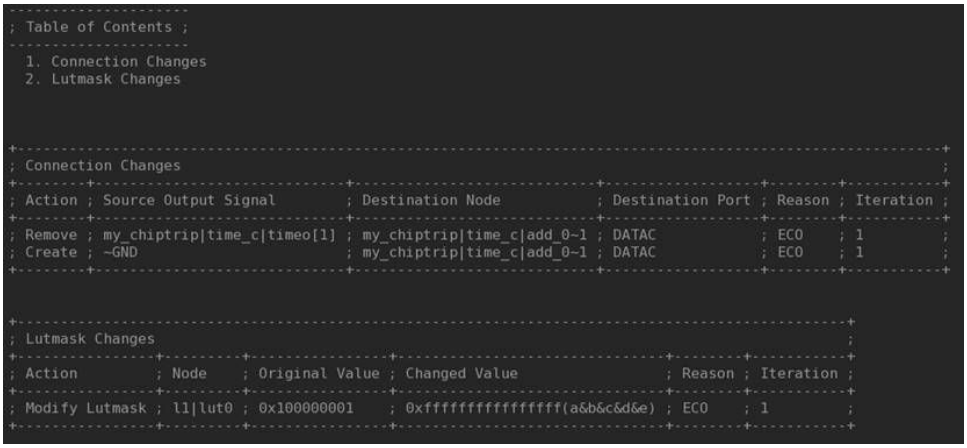
to 需要修改延迟链设置的 I/O 管脚的实例名称。

7.4. 查看 ECO 编译报告

Compiler 生成显示关于成功运行的每个 ECO 编译的详细信息报告。可从 Compilation Report 的 **Fitter** 下查看 **ECO Changes** 报告了解报告详细内容。或者，可在生成的 `fit.eco` 文件中查看该数据。

Compiler 根据 ECO 更改的类别（例如，“连接更改”）来组织报告输出。该表格中指定了 ECO 编译的迭代。

图 69. ECO 报告实例



```
; Table of Contents ;
  1. Connection Changes
  2. Lutmask Changes

; Connection Changes                                                                                                ;
; Action ; Source Output Signal        ; Destination Node           ; Destination Port ; Reason ; Iteration ;
; Remove ; my_chiptrip|time_c|timeo[1] ; my_chiptrip|time_c|add_0~1 ; DATAC            ; ECO    ; 1         ;
; Create ; ~GND                         ; my_chiptrip|time_c|add_0~1 ; DATAC            ; ECO    ; 1         ;

; Lutmask Changes                                                                                                   ;
; Action         ; Node    ; Original Value ; Changed Value                 ; Reason ; Iteration ;
; Modify Lutmask ; l1|lut0 ; 0x100000001    ; 0xffffffffffffffff(a&b&c&d&e) ; ECO    ; 1         ;
```

7.5. ECO 命令限制

由于 Intel FPGA 器件内的连接依赖性，ECO 命令具有以下限制。仅可使用 ECO 命令修改使用核心布线资源的连接。不可使用 ECO 命令修改专属连接，或布线到使用全局时钟网络的连接。因此，这些限制通过以下方式影响 `remove_connection` 和 `make_connection`：

- 仅可删除从 `-from` 信号到目标中包含核心布线的连接
- 不可修改但单个 ALM 中的专用连接。该限制适用于 LUT 和触发器节点之间的直接连接。
- 不可修改使用全局时钟布线资源的连接。

- 修改 RAM 块的控制输入时，必须使用相同 ECO 变更对每个使用该信号的 RAM 节点进行修改，因为位于同一物理位置的 RAM 节点共享相同的布线连接。可使用 Resource Property Editor 中的 **Node Selection** 面板确定要修改的 RAM 节点列表。选择 **RAM(s)**选项卡查看用于实现物理 RAM 的 RAM 节点列表。可右键点击并选择 **Copy All** 复制该节点名称列表以用于 ECO 命令。
- 要更改对 LUTRAM 的控制，必须对同一 LAB 中的所有 LUTRAM 应用相同的更改。
- 可将其他连接添加到现有 Hyper-Register 输出，但不可从 Hyper-Register 删除任何现有连接。

注意: 为最小化转义字符引起的名称匹配问题，可将实体名称用{}符括起来，而非使用""。如果实体名称包含反斜线或其他特殊符号，则该方法特别实用。

7.6. 实现工程变更命令修订历史

以下修订历史适用于本章：

表 24. 文档修订历史

文档版本	Intel Quartus Prime 版本	修订内容
2019.09.30	19.3.0	• 添加有关 `make_connection` 命令的 `tieoff` 选项的信息。 • 添加有关 `modify_io_slew_rate` 命令的支持。 • 添加有关 `modify_io_current_strength` 命令的支持。 • 添加有关 `modify_io_delay_chain` 命令的支持。 • 添加了“查看 ECO 编译报告”主题。 • 添加了有关 `modify_lutmask` 命令的 `num` 选项的信息。 • 提出使用 RTL Viewer 定位节点名称 • 添加器件支持说明。
2019.07.01	19.2.0	• First release of chapter.

8. Intel Quartus Prime Pro Edition 设计优化用户指南存档

如有表格中未列出的软件版本，请应用该软件先前版本的用户指南。

Intel Quartus Prime 版本	用户指南
19.1	Intel Quartus Prime Pro Edition 用户指南：设计优化
18.1	Intel Quartus Prime Pro Edition 用户指南：设计优化
18.0	Intel Quartus Prime Pro Edition 设计优化用户指南

A. Intel Quartus Prime Pro Edition 用户指南

请参阅以下用户指南获得关于 Intel Quartus Prime Pro Edition FPGA 设计流程中所有阶段的综合性信息。

相关链接

- Intel Quartus Prime Pro Edition 用户指南：入门
  介绍 Intel Quartus Prime Pro Edition 软件的基本功能，文件和设计流程，包括管理 Intel Quartus Prime Pro Edition 工程和 IP，初始设计布局考量以及从软件先前版本进行工程移植。
- Intel Quartus Prime Pro Edition 用户指南：平台设计程序
  说明使用 Platform Designer 创建和优化系统，该系统集成工具可简化工程中自定义 IP 核聚合。Platform Designer 自动生成互连逻辑以连接知识产权（IP）功能和子系统。
- Intel Quartus Prime Pro Edition 用户指南：设计建议
  介绍使用 Intel Quartus Prime Pro Edition 软件进行 FPGA 设计时的最佳设计实践。HDL 代码样式和同步设计时间可显著影响设计性能。以下建议的 HDL 代码样式可确保 Intel Quartus Prime Pro Edition 将设计在硬件中最佳实现。
- Intel Quartus Prime Pro Edition 用户指南：设计编译
  说明了 Intel Quartus Prime Pro Edition Compiler 从建立，运行到优化的全部阶段。生成器件编程文件之前，Compiler 对设计进行综合，布局和布线。
- Intel Quartus Prime Pro Edition 用户指南：设计优化
  介绍 Intel Quartus Prime Pro Edition 可用于实现 Intel FPGA 中最高设计性能的设置，工具和技术。技术包括优化设计网表，解决限制重定时和时序收敛的关键链，优化器件资源使用，器件布局规划以及实施工程变更单(ECO)。
- Intel Quartus Prime Pro Edition 用户指南：Programmer
  说明 Intel Quartus Prime Pro Edition Programmer 的运行，并通过连接 Intel FPGA 下载电缆配置 Intel FPGA 器件，编程 CPLD 和配置器件。
- Intel Quartus Prime Pro Edition 用户指南：基于块的设计
  说明基于块的设计流程，亦称为模块化或分层式设计流程。这些高级流程可将设计块（或是包含分层型设计实例的逻辑）保留在工程中，并在其他工程中重复使用设计块。
- Intel Quartus Prime Pro Edition 用户指南：局部重新配置
  介绍 Partial Reconfiguration 高级设计流程，其支持动态重配置 FPGA 某一部分的同时其余 FPGA 设计继续运行。将部分设计区域定义为多重角色时，并不影响其他区域中的操作。
- Intel Quartus Prime Pro Edition 用户指南：第三方仿真
  说明通过 Aldec*，Cadence*， Mentor Graphics* 和 Synopsys 为第三方仿真工具提供 RTL-和门级设计仿真支持，从而允许在器件编程之前验证设计行为。包括仿真器支持，仿真流程和仿真 Intel FPGA IP。
- Intel Quartus Prime Pro Edition 用户指南：第三方综合
  说明通过 Mentor Graphics* 和 Synopsys ，第三方综合工具为设计中选择性综合部分提供支持。包括设计流程步骤，生成的文件说明和综合指导。

- Intel Quartus Prime Pro Edition 用户指南：第三方逻辑等效检查工具
  对 OneSpin*的第三方 LEC 工具中设计的可选逻辑等效性检查(LEC)的支持进行了描述。
- Intel Quartus Prime Pro Edition 用户指南：调试工具
  介绍为设计进行实时验证的 Intel Quartus Prime Pro Edition 在系统设计调试工具文件夹。这些工具通过将设计中的信号路由选择（或“分接”）到调试逻辑来提供可视性。具体工具包括，System Console，Signal Tap logic analyzer，Transceiver Toolkit，In-System Memory Content Editor 和 In-System Sources and Probes Editor。
- Intel Quartus Prime Pro Edition 用户指南：Timing Analyzer
  解释基本静态时序分析原则和 Intel Quartus Prime Pro Edition Timing Analyzer 的使用。其作为功能强大的 ASIC 式时序分析工具，通过使用行业标准的约束，分析和报告方法验证设计中所有逻辑的时序性能。
- Intel Quartus Prime Pro Edition 用户指南：功耗分析和优化
  说明 Intel Quartus Prime Pro Edition Power Analysis 工具支持准确估算器件功耗。估算器件功耗以开发功率预算和设计电源，稳压器，散热器和冷却系统。
- Intel Quartus Prime Pro Edition 用户指南：设计约束
  说明影响 Compiler 如何实现设计的时序和逻辑，例如，管脚约束，器件选项，逻辑选项和时序约束。使用 Interface Planner 原型开发接口实现，规划时钟并迅速定义合法器件平面图。使用 Pin Planner 在目标器件的图形呈现中可视化，修改和验证所有 I/O 约束。
- Intel Quartus Prime Pro Edition 用户指南：PCB 设计工具
  说明通过 Mentor Graphics* 和 Cadence*实现对可选第三方 PCB 设计工具的支持。还包括信号集成分析和使用 HSPICE 和 IBIS 模型进行仿真的信息。
- Intel Quartus Prime Pro Edition 用户指南：脚本
  说明使用 Tcl 和命令行编制脚本进行 Intel Quartus Prime Pro Edition 软件控制并广泛执行各种功能，例如管理工程，指定约束，运行编译或时序分析，以及生成报告。