Gowin 设计时序约束

用户指南

SUG940-2.0, 2025-06-27

免责声明

本文档并未授予任何知识产权的许可，并未以明示或暗示，或以禁止反言或其它方式授予任何知识产权许可。除高云半导体在其产品的销售条款和条件中声明的责任之外，高云半导体概不承担任何法律或非法律责任。高云半导体对高云半导体产品的销售和／或使用不作任何明示或暗示的担保，包括对产品的特定用途适用性、适销性或对任何专利权、版权或其它知识产权的侵权责任等，均不作担保。高云半导体对文档中包含的文字、图片及其它内容的准确性和完整性不承担任何法律或非法律责任，高云半导体保留修改文档中任何内容的权利，恕不另行通知。高云半导体不承诺对这些文档进行适时的更新。

版本信息

日期	版本	说明
2020/06/09	1.0	初始版本。
2020/09/01	1.1	● 工作条件设定增加车规级-grade a； ● 增加基础时钟与衍生时钟联动功能。
2021/06/16	1.2	● 增加通配符描述； ● 更新图例。
2021/11/02	1.3	● 更新软件版本号； ● 更新图例以及相应说明； ● 更新第 5 章语法规范。
2022/05/20	1.3.1	更新一些语言描述。
2022/07/28	1.4	● Create Clock 增加-Add 选项描述； ● 增加了延迟时间数据模型的温度电压描述； ● 增加 tUnc 以及 tSu 含义描述； ● 增加通配符跨层级匹配功能描述。
2022/12/16	1.4.1	● 更新 3.7.2 I/O 约束中选项 Add delay 的描述。
2023/03/31	1.4.2	● 更新 3.7.2 I/O 约束中 set_input_delay、set_output_delay 描述； ● 更新 4.1.4 Total Negative Slack Summary 描述。
2023/04/20	1.4.3	新增图 3-8 新建窗口图 3-9 打开窗口以及相关描述。
2023/05/25	1.5	● 更新例外约束描述； ● 更新虚拟时钟 DEFAULT_CLK 描述。
2023/08/18	1.5.1	● 更正 set_operation_conditions 为 set_operating_conditions； ● 更新报告最大时钟频率约束描述。
2023/11/30	1.6	● 移除时钟周期与频率转换描述； ● 移除 Process 窗口截图及相关描述； ● 更新 set_output_delay 描述。
2024/02/02	1.7	增加 derive_clocks 约束。
2024/06/28	1.8	create_clock 增加-add 选项的 DCS 使用场景。
2024/08/09	1.8.1	更新图 4-1 静态时序分析报告和图 4-2 Timing Summaries。
2024/10/25	1.8.2	完善 4.1.2 Clock Summary 中 Clock Name 默认显示规则描述。
2024/12/31	1.8.3	● 更新图 3-5 时序约束编辑器界面； ● 更新第 5 章语法规范示例。
2025/04/30	1.9	● 更新第 5 章语法规范描述； ● 更新部分图例及相应描述。
2025/06/27	2.0	● 增加差分 BUF 时钟创建规则描述；

日期	版本	说明
		● 增加最小脉冲宽度计算公式。

目录

图目录

表目录

1 关于本手册

1.1 主要内容

本手册主要内容为高云半导体时序约束编辑器、时序报告、语法规范。

1.2 相关文档

通过登录高云半导体网站 www.gowinsemi.com.cn 可下载、查看以下相关文档：SUG918，Gowin 云源软件快速入门指南。

1.3 术语、缩略语

本手册中的相关术语、缩略语及相关释义请参见表 1-1。

表 1-1 术语、缩略语

术语、缩略语	全称	含义
FF	Flip-Flop	触发器
GUI	Graphical User Interface	图形用户界面
I/O	Input and Output	输入输出端口
LSR	Clear, Set, Reset, and Preset	清除、复位、置位
NET	Net	连线
OSC	Oscillator	振荡器
PIN	Pin	实例化模块的输入输出管脚
PORT	Port	顶层模块输入输出端口
SDC	Synopsys Design Constraint	Synopsys 设计约束
STA	Static Timing Analysis	静态时序分析

1.4 技术支持与反馈

高云半导体提供全方位技术支持，在使用过程中如有任何疑问或建议，可直接与公司联系：

网址：www.gowinsemi.com.cn

E-mail：support@gowinsemi.com

Tel：+86 755 8262 0391

2 简介

高云静态时序分析（STA）对电路网表中的时序路径进行全面的分析，计算电路中时序路径延迟，并判断其是否满足要求。设计者仅需提供约束激励，计算分析过程由云源软件自动完成，提供详细的时序报告，验证时间以及覆盖率均占有极大的优势。

2.1 约束支持

- 支持时钟约束；
- 支持 I/O 延迟约束；
- 支持例外约束；
- 支持报告约束；
- 支持工作条件约束；
- 支持全局目标时钟约束；
- 支持通过时序约束编辑器 GUI 编辑约束；
- 时序约束语法参考 SDC 标准规范。

2.2 报告打印

- 支持打印时序延迟模型；
- 支持打印最大时钟频率；
- 支持打印建立（setup）和保持（hold）路径分析；
- 支持打印恢复（recovery）和移除（removal）路径分析；
- 支持打印最小脉冲宽度分析；
- 支持打印高扇出 net；
- 支持打印绕线拥塞度；
- 支持打印例外约束报告；

- 支持打印 HTML 和 TXT 两种文件格式。

2.3 时序路径

静态时序分析对四种类型的时序路径进行分析，根据起点和终点的不同对其进行分类：

- 从输入端口到内部时序元件路径；
- 从时序元件到时序元件路径；
- 从内部时序元件到输出端口路径；
- 从输入端口到输出端口路径。

3 时序约束编辑器

3.1 概述

用户可通过时序约束编辑器（Timing Constraints Editor）增加、删除、修改及检查时序约束。

3.2 启动时序约束编辑器

时序约束编辑器可单独启动使用也可在执行综合成功后使用。

单独使用时单击“Tools > Timing Constraints Editor”启动。打开工程使用时需在云源 Process 窗口中运行“Synthesize”成功后，双击“Process > Timing Constraints Editor”启动。网表文件与工程中的时序约束文件会自动加载到时序约束编辑器中，若工程中不存在时序约束文件则会自动创建。

3.3 新建、打开及添加约束文件

3.3.1 新建约束文件

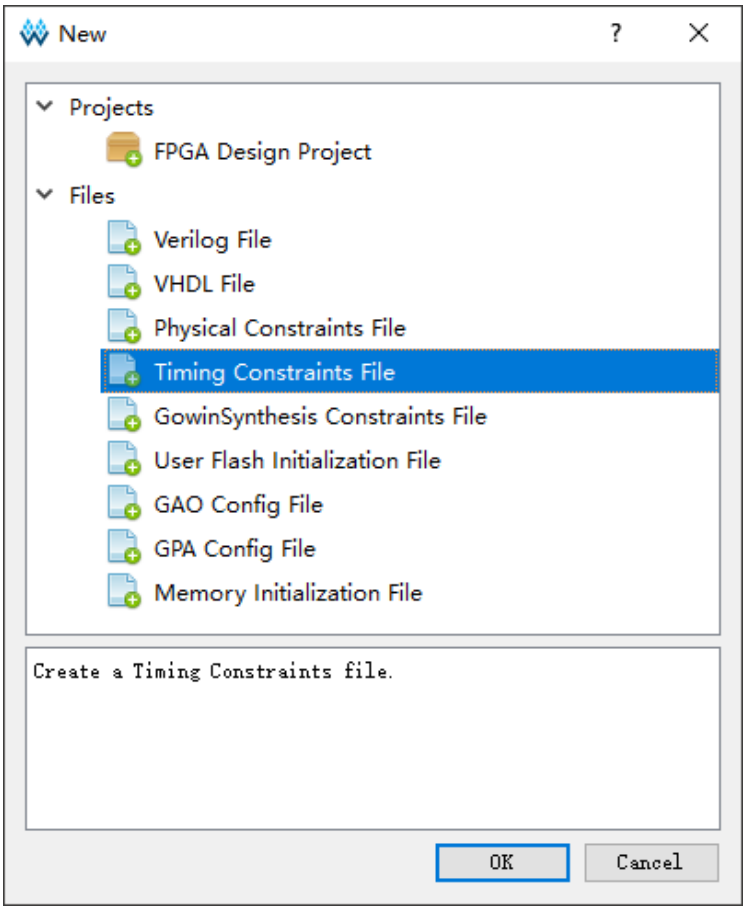
新建约束文件的步骤如下：

1. 单击“File > New”菜单项，弹出新建文件对话框；
2. 选择“Timing Constraints File”选项，如图 3-1 所示。

亦可通过以下方式打开新建时序约束文件对话框：

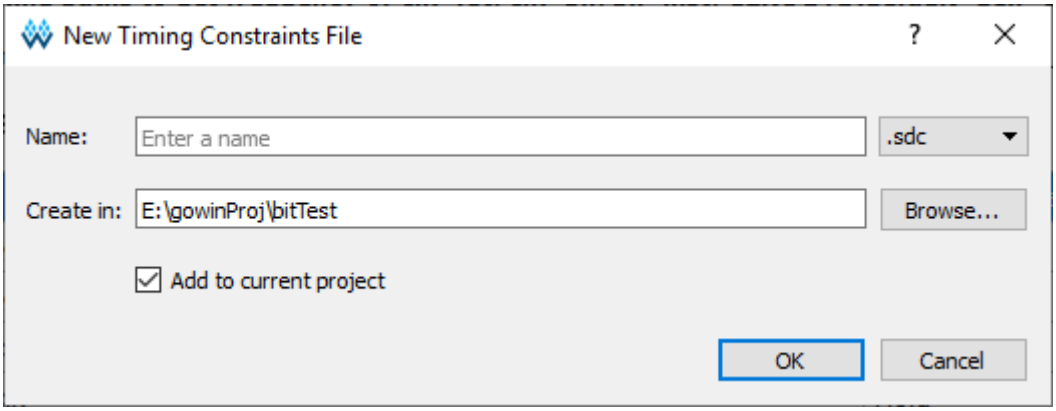
- 单击工具栏上的“New”图标；
- 使用快捷键 Ctrl + N。

图 3-1 打开新建时序约束文件对话框



3. 单击“OK”确认，弹出新建时序约束文件的对话框，如图 3-2 所示；

图 3-2 新建时序约束文件



4. 键入文件名及选定创建目录后单击“OK”，创建时序约束文件且自动将文件加载进工程中。

   - Name：新建时序约束文件的名称，文件类型支持.sdc，文件名建议使用字母或下划线开头的具有工程相关意义的标识符；

- Create in：单击“Browse”按钮选择新建约束文件的存放位置，需是存在的目录，默认路径为工程目录下的 src 文件夹下；
- Add to current project：选择该选项后，会自动将约束文件添加到工程中，默认勾选。

3.3.2 打开约束文件

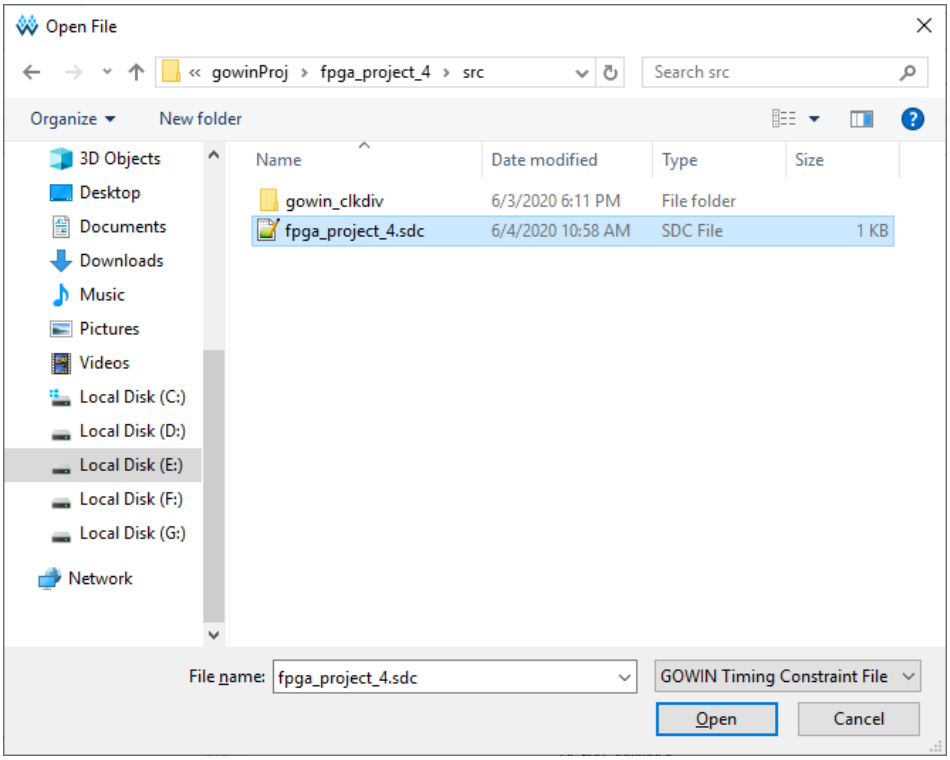
打开约束文件的步骤如下：

1. 在 IDE 界面中，单击“File > Open”菜单项；
2. 打开“Open File”对话框，如图 3-3 所示；

亦可通过以下方式打开时序约束文件对话框：

- 单击工具栏上的“Open”图标；
- 使用快捷键 Ctrl + O。

图 3-3 打开时序约束文件



3. 选择时序约束文件所在的目录，选中文件单击“Open”后打开文件，支持 sdc 文件类型。

3.3.3 添加约束文件

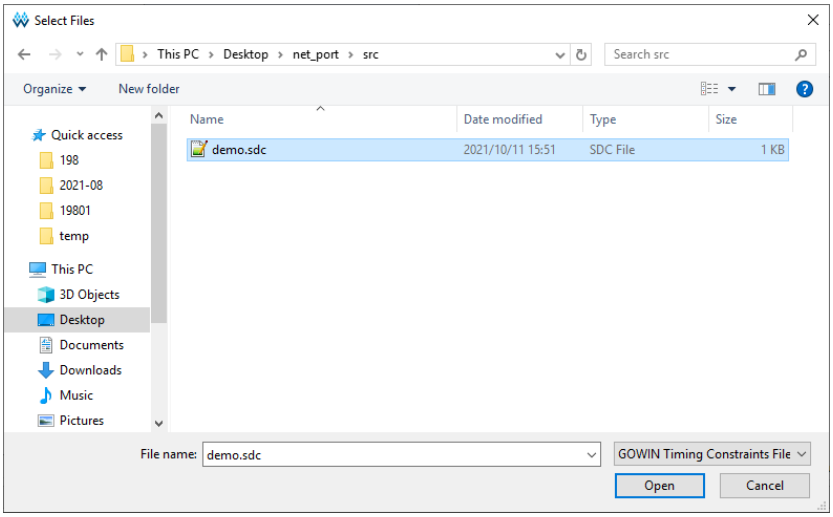
添加时序约束文件步骤如下：

1. 在 IDE 界面 Design 窗口内右击选择“Add Files”；
2. 弹出“Select Files”对话框，文件类型选择“.sdc”， 如图 3-4 所示；
3. 选定一个或多个约束文件后单击“Open”即可添加进工程。

注！

添加多个文件时仅一个有效。

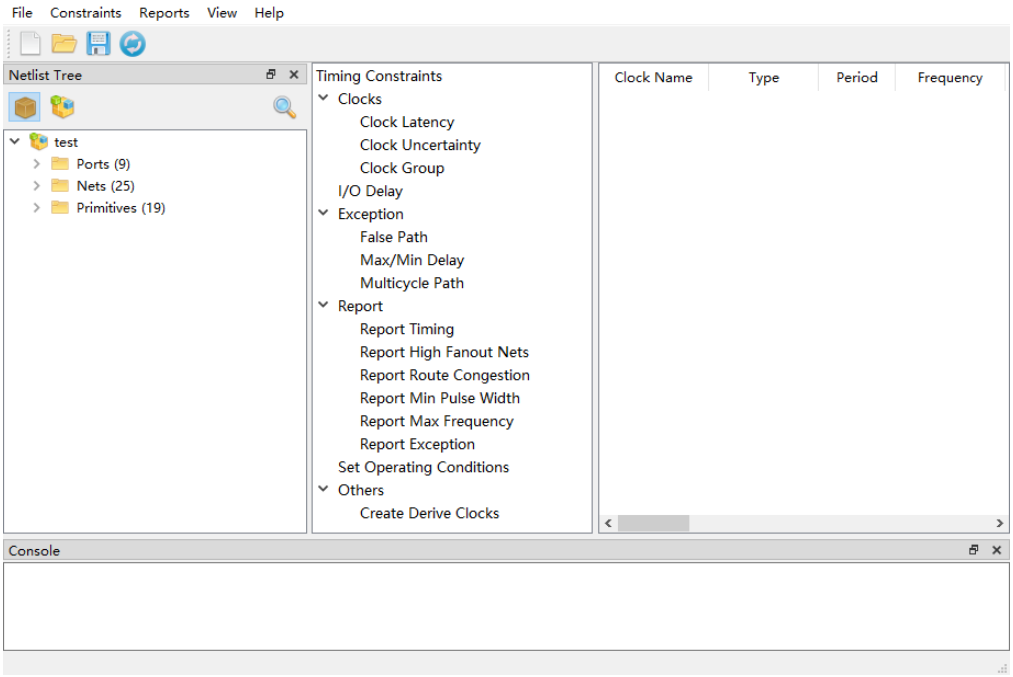
图 3-4 添加时序约束文件



3.4 时序约束编辑器界面

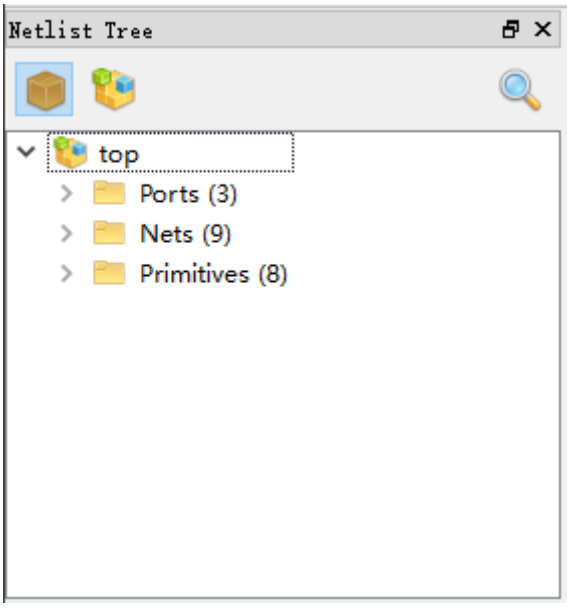
打开约束文件后，时序约束编辑器界面如图 3-5 所示。

图 3-5 时序约束编辑器界面



主窗口左上角为 Netlist Tree 窗口如图 3-6 所示。

图 3-6 Netlist Tree 窗口



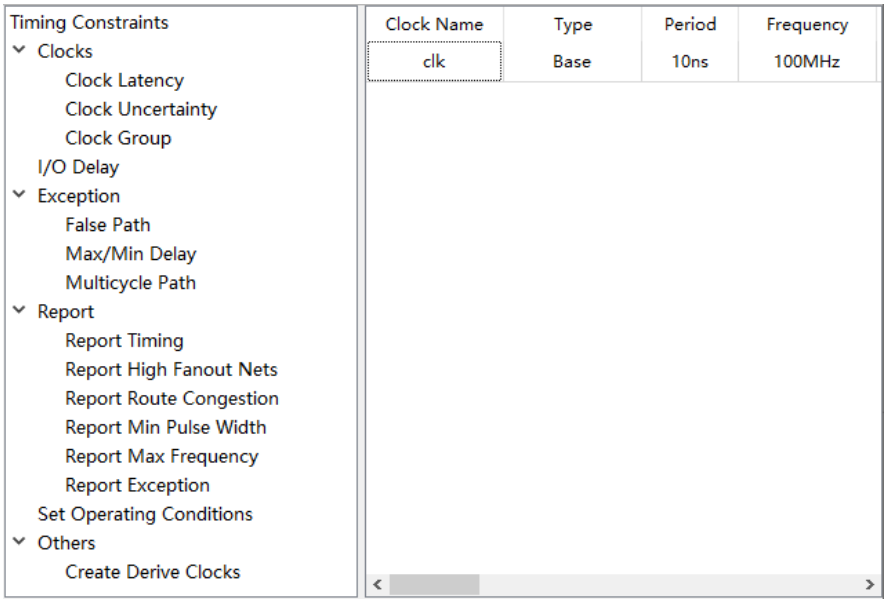
Netlist Tree 窗口中包括当前网表文件中的 Top Module、I/O Ports、Nets 和 Primitives。

- “  ”：查看 flatten 列表；

- “  ”：查看 hierarchy 列表。

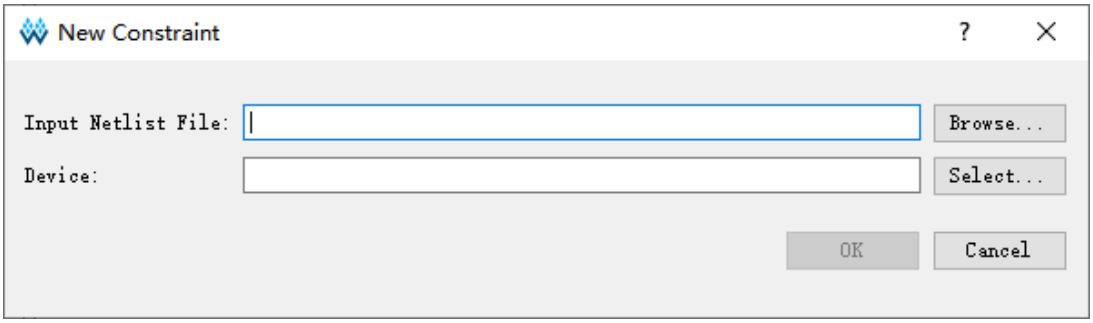
主窗口中间及右侧区域为约束编辑区如图 3-7 所示。其中左侧列表为时序约束类型目录，右侧为表格编辑区。在类型目录上单击选中某一约束类型，表格编辑区中会显示已设置的约束编辑列表。

图 3-7 约束编辑窗口



主窗口上部工具栏包含新建“  ”、打开“  ”、保存“  ”、重新加载“  ”。新建窗口包含选定网表文件“Input Netlist File”与选定器件信息“Device”。

图 3-8 新建窗口



打开窗口包含选定网表文件“Input Netlist File”、约束文件“Constraint File”与选定器件信息“Device”。

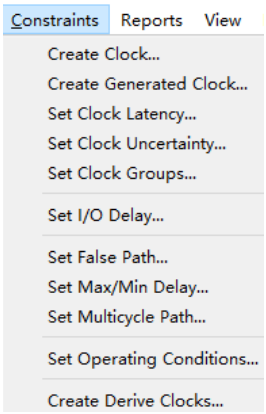
图 3-9 打开窗口

3.5 打开时序约束窗口

提供两种时序约束窗口打开方式。

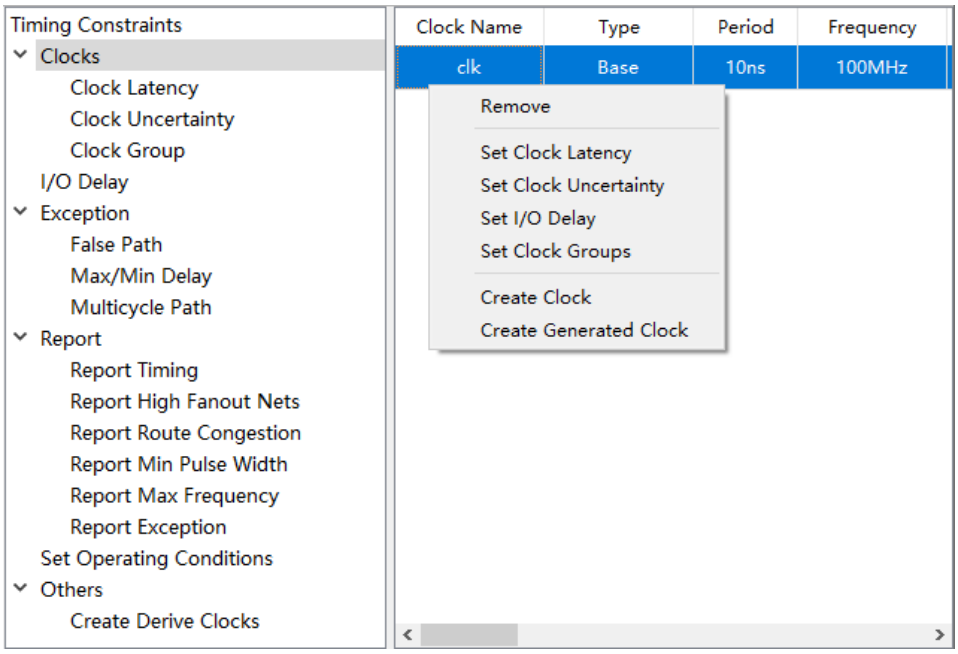
1. 在菜单栏中，单击“Constraints”，在其下拉菜单中，选择时序约束命令，通过选取相应的约束命令打开时序约束窗口，如图 3-10 所示；

图 3-10 菜单栏打开时序约束窗口



2. 在时序约束编辑器右侧的表格窗口右击，根据右键菜单列表中不同的选项，选取不同的时序约束命令可打开相应的约束窗口，如图 3-11 所示。

图 3-11 右键打开时序约束窗口



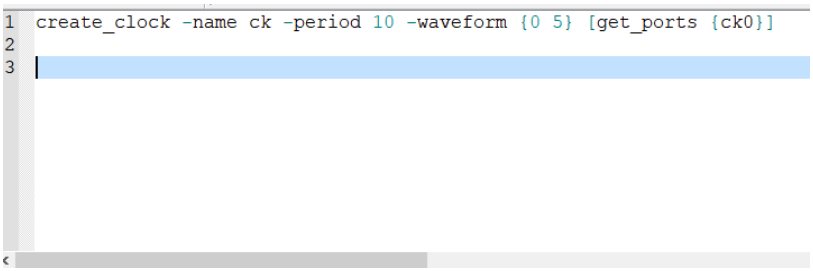
3.6 编辑 SDC 文件

云源支持读取工程的 SDC 文件，并可在文本编辑器中手动修改约束，操作便捷，如图 3-12 所示。

SDC 文件的解析支持通配符功能，目前支持两种通配符“*”和“?”。“*”实现零个或多个字符的匹配，而“?”实现对一个字符的匹配。

SDC 文件支持单行注释和多行注释。单行注释使用“//”或“#”，多行注释使用“/* */”。

图 3-12 编辑 SDC 文件



3.7 创建时序约束

本小节介绍使用时序约束编辑器创建时序约束，创建的约束会在 Console 窗口中打印，点击保存可存储在 SDC 文件中。

3.7.1 时钟约束

Create Clock

创建时钟。

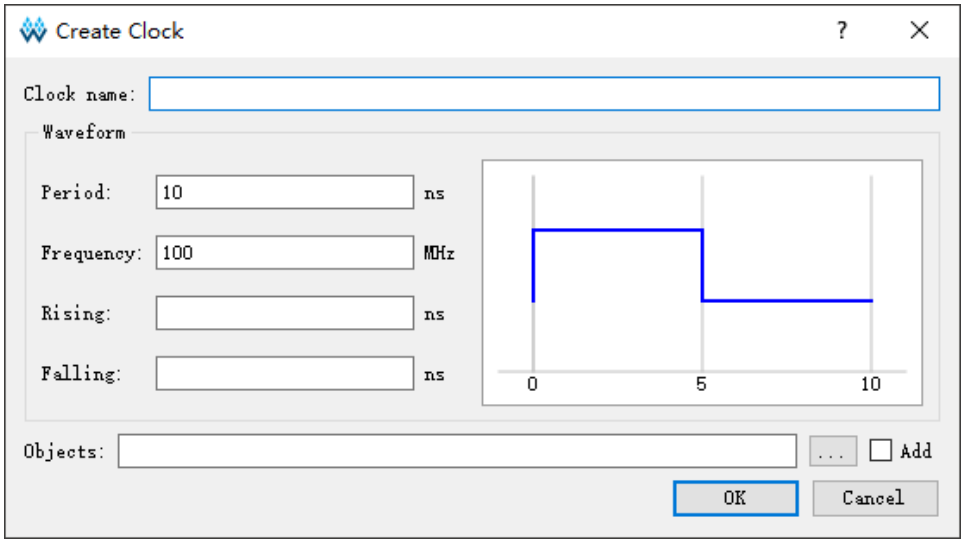
可指定时钟名称、周期、频率、上升沿、下降沿，以及该时钟作用的目标等参数。云源支持建立多个时钟，形成多个时钟域，支持跨时钟域分析。

可通过以下两种方式新增 Clock 约束：

1. 通过 Constraints 菜单新增 Clock 约束；

   a). 选择“Constraints > Create Clock…”，弹出“Create Clock”对话框，如图 3-13 所示；

图 3-13 创建基础时钟



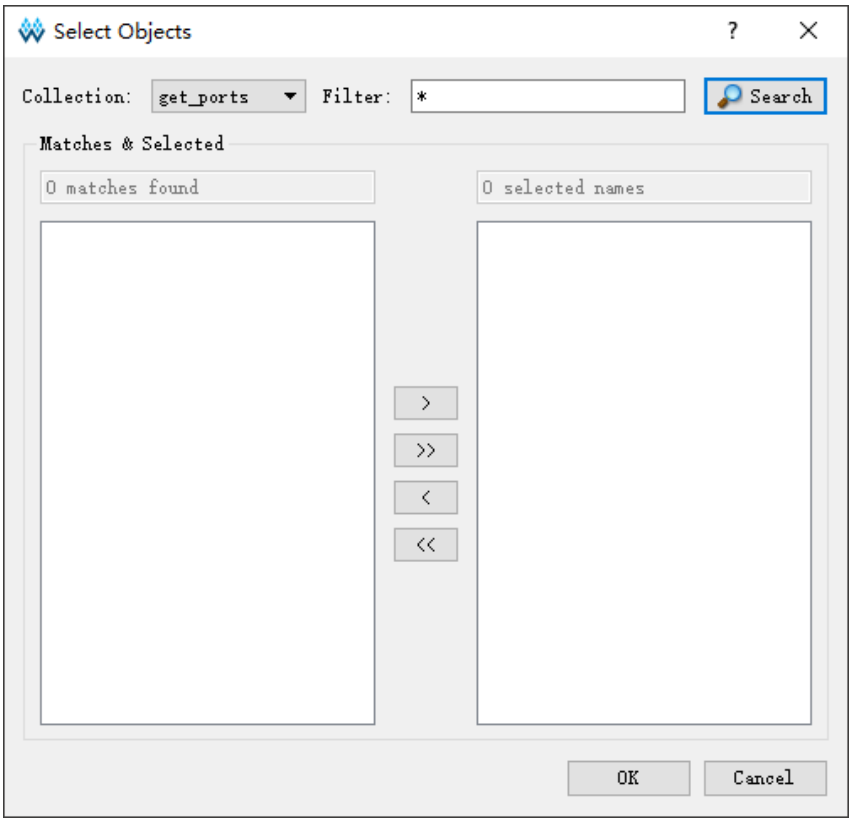
   b). 填写 Clock 信息；

   - Clock Name：时钟名，支持字母或下划线开头的标识符；
   - Period：周期，默认 10，大于 0 的浮点型，精确到千分位，单位 ns；
   - Frequency：频率，默认 100，大于 0 的浮点型，精确到千分位，单位 MHz；
   - Rising：上升沿时刻，大于 0 的浮点型，精确到千分位，单位 ns；
   - Falling：下降沿时刻，大于 0 的浮点型，精确到千分位，单位 ns；
   - Objects：指定作用目标，通过“...”选择进行内容的自动填充；
   - Add：在同一个源上添加多个时钟时需要勾选。

   c). 单击 Objects 右侧的“...”按钮，会弹出“Select Objects”对话框，

如图 3-14 所示；

图 3-14 选择作用目标



d). 在图 3-14 中，“Collection”指定搜索的集合类型，“Filter”为过滤器，单击“Search”后左侧为匹配后结果，右侧为已选中列表，“>”按钮将左边列表中的选中项添加到右边列表，“>>”按钮添加左边所有项，“<”按钮移除右边选中项，“<<”按钮移除右边所有项；

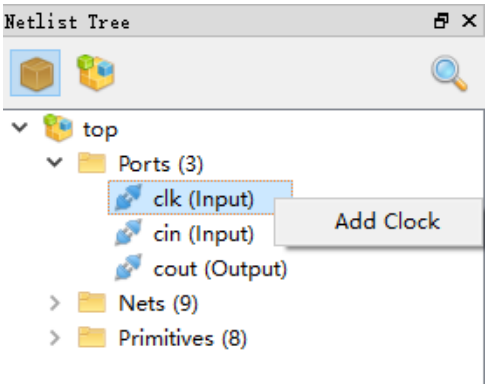
e). 单击“OK”，完成 Objects 的添加。

2. 通过 Netlist Tree 新增 Clock 约束。

a). 在 Netlist Tree 中，选中 Port 或 Net；

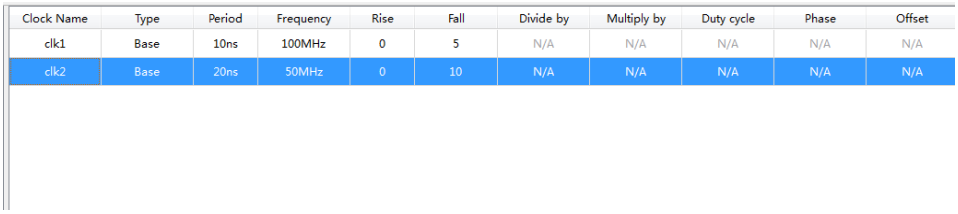
b). 右击，选择“Add Clock”，添加一个时钟，如图 3-15 所示。

图 3-15 添加时钟



创建后的时钟如图 3-16 所示。

图 3-16 时钟列表

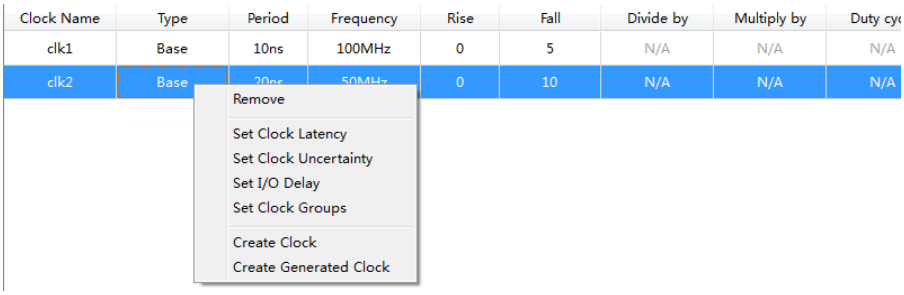


Clock Name	Type	Period	Frequency	Rise	Fall	Divide by	Multiply by	Duty cycle	Phase	Offset
clk1	Base	10ns	100MHz	0	5	N/A	N/A	N/A	N/A	N/A
clk2	Base	20ns	50MHz	0	10	N/A	N/A	N/A	N/A	N/A

在该列表中，可进行如下操作：

- 编辑 Clock，双击“Clocks”列表中对应的约束，打开 Clock 的编辑对话框，可在对话框中编辑修改 Clock 信息；
- 删除 Clock，在列表中选择该条 Clock，右击，选择“Remove”；
- 右击 Clock，可快速为该条 Clock 设置 Clock Latency、Clock Uncertainty 或 I/O Delay 信息，如图 3-17 所示。

图 3-17 时钟列表右键内容



注！

- 当约束与 PLL 配置不一致时以“Create Clock”创建的约束为准，PnR 时将提示警告信息；
- “Create Clock”不支持创建一个虚拟时钟；
- 不支持在差分 BUF 的 N 端创建时钟如 IOB；
- 云源支持根据不同的时钟占空比进行时序分析。

Create Generated Clock

创建衍生时钟。

通过该约束可基于主时钟进行分频、倍频、相移和调整占空比等操作，进而完成衍生时钟的创建。创建后的衍生时钟自动与主时钟进行联动，当主时钟的属性发生变化时衍生时钟会自动修正以适配主时钟。

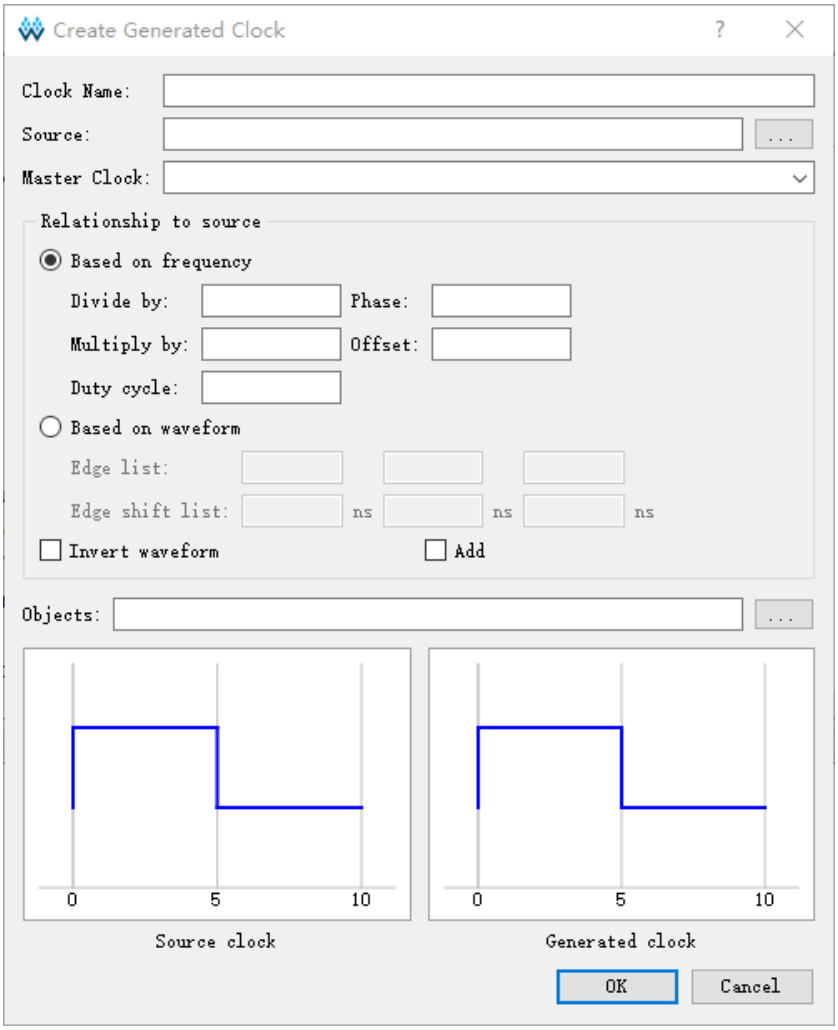
可通过以下两种方式创建衍生时钟：

1. 通过 Constraints 菜单创建；

   a). 在“Constraints”菜单中，选择“Create Generated Clock”，弹出“Create Generated Clock”对话框，如图 3-18 所示；

- Clock Name：时钟名，支持字母或下划线开头的标识符；
- Source：衍生时钟的源，通过右侧“...”选择；
- Master Clock：主时钟，通过右侧“⌄”进行选择。

图 3-18 创建衍生时钟约束



b). 通过单击 Source 右侧的“...”选择时钟的源，与“Source”关联的时钟会自动添加到“Master Clock”列表中，选择“Master Clock”，当“Master Clock”存在多个时钟时，仅支持任选其一；

c). “Relationship to source”中，基于频率（Base on frequency）可对当前创建的衍生时钟进行分频、倍频、偏移、占空比及相位等调整，基于波形（Base on waveform）使用边沿列表（Edge list）并配合边沿偏移列表（Edge shift list）可实现对衍生时钟进行边沿调整；

- Divide by：分频数，正整数；
- Phase：相位，浮点型，精确到千分位，负数左移，正数右移；
- Multiply by：倍频数，正整数；

- Offset：偏移量，浮点型，精确到千分位，负数左移，正数右移；
- Duty cycle：占空比，浮点型，精确到千分位；
- Edge list：依次递增的正整数；
- Edge shift list：浮点型，精确到千分位。

d). “Invert waveform”实现对时钟的反相，“Add”可在已有时钟的目标上实现添加，STA 分析时同时有效；

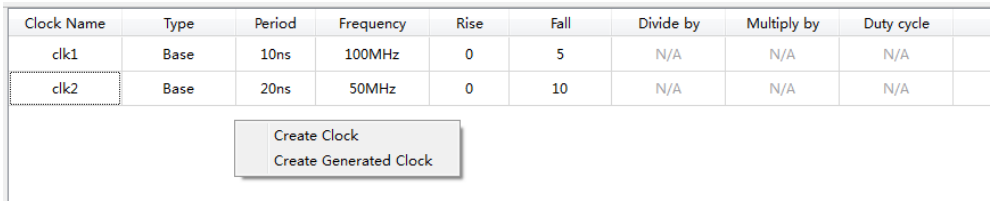
e). “Objects”栏用以选定时钟作用的对象，单击 Objects 右边的“ ... ”按钮，会弹出“Select Objects”对话框，选择目标对象。

注！

- 当约束与 PLL 配置不一致时以 Create Generated Clock 创建的约束为准，PnR 时提示警告信息。

2. 通过 Clocks 列表创建 Generated Clock。在 Clocks 列表中，在空白处右击选择“Create Generated Clock”新建 Generated Clock，如图 3-19 所示。

图 3-19 选择 Create Generated Clock



添加后表格编辑区会增加新建的约束。

在该列表中，可进行如下操作：

- 编辑 Generated Clock 约束，双击“Clocks”列表中对应的约束，打开 Generated Clock 的编辑对话框，在对话框中编辑修改 Generated Clock 信息；
- 删除 Generated Clock，在表格编辑区选中该 Clock，右键选择“Remove”。

Set Clock Latency

设置时钟源延迟时间。

时钟延迟时间分为两种，网络（NETWORK）延迟时间和源（SOURCE）延迟时间：

- 网络（NETWORK）延迟时间是时钟和时序元件之间的延迟时间；
- 源（SOURCE）延迟时间是时钟和源之间的延迟时间；

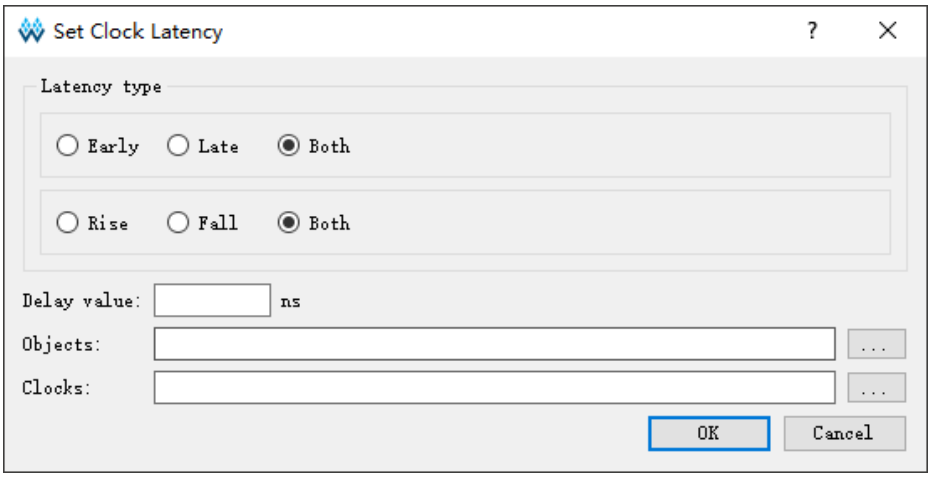
云源自动计算时钟的网络（NETWORK）延迟时间，用户只需设定源

（SOURCE）延迟时间。

可通过以下两种方式新建 Clock Latency 约束：

1. 通过 Constraints 菜单新建 Clock Latency 约束。在“Constraints”菜单中，选择“Set Clock Latency”，弹出“Set Clock Latency”对话框，如图 3-20 所示，填写 Latency 信息，单击“OK”保存约束。
   - Early、Late：表示设置的时钟最早、最晚延迟时间值，Both 表示对两者均有效；
   - Rise、Fall：分别表示对上升沿、下降沿有效，Both 表示对两者均有效；
   - Delay value：时钟延迟时间值，浮点型，精确到千分位，单位 ns；
   - Objects：通过右侧“ ”选择，指明对象；
   - Clocks：通过右侧“ ”选择，指明作用的时钟。

图 3-20 设置时钟延迟时间



2. 通过 Clocks 列表新建 Clock Latency 约束。

在 Clocks 列表中选中 Clock，右击选择 Set Clock Latency 为该 Clock 设置 Latency 信息，Objects 将自动指定为该时钟目标。

Set Clock Uncertainty

设置时钟不确定值。

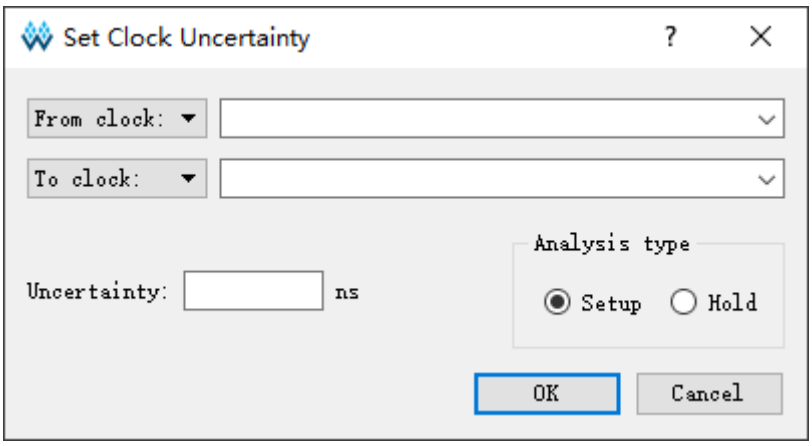
新建 Clock Uncertainty 操作如下：

1. 在“Constraints”菜单中，选择“Set Clock Uncertainty”，弹出“Set Clock Uncertainty”对话框，如图 3-21 所示；
   - From clock：指明发送时钟，通过右侧“ ”进行选择；
   - To clock：指明结束时钟，通过右侧“ ”进行选择；
   - Uncertainty：浮点型，精确到千分位，设置时钟的不确定值，单位

ns；

- Analysis type：指明分析的类型。

图 3-21 设置时钟不确定量



2. 通过左侧的下拉框选择 From 的类型（From clock、Rise from、Fall from）和 To 的类型（To clock、Rise to、Fall to），通过右侧的下拉框从当前所有已创建的 Clock 中选择目标的 Clock；
3. 填写信息完成后，单击“OK”保存约束，完成 Uncertainty 的添加。

Set Clock Group

设置时钟组。

云源默认认为设计中所有时钟同属一组，且都相关。

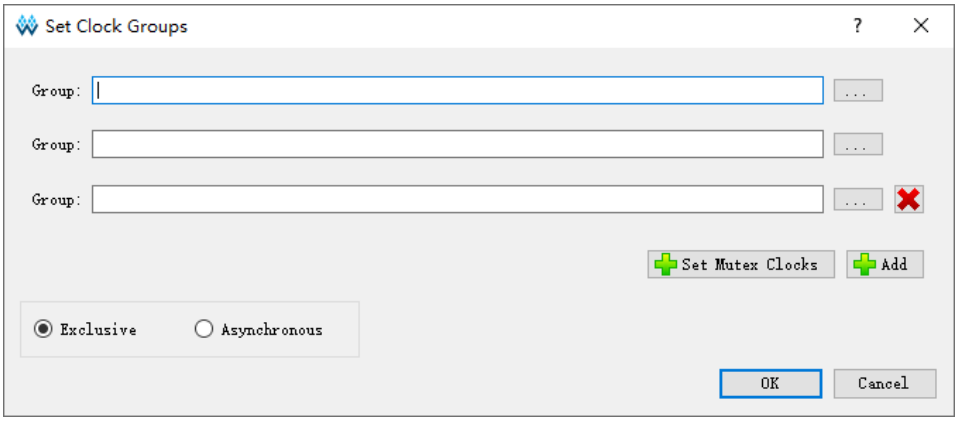
建议用户使用该约束语句严格指明时钟间的关系，对于异步或互斥的时钟建立时钟组约束。

新建 Clock Group 方式如下：

1. 在“Constraints”菜单中，选择“Set Clock Groups”，弹出“Set Clock Groups”对话框，如图 3-22 所示；
   - Group：通过右侧“ ... ”进行选择，指明组内的时钟，至少指定一个时钟；
   - Set Mutex Clocks：用以实现一次设置多个不同时钟组；
   - Add：再添加一条 Group 条目；
   - Exclusive：指明时钟为互斥关系，同一时刻时钟不会同时有效，例如 Clock0、Clock1 经过一个 MUX2（两路复选）后输出的时钟 Clock3 作用于一个时序模型，同时刻 Clock3 只能取 Clock0 或 Clock1，则可使用该选项；
   - Asynchronous：指明时钟异步不相关，时钟有不同的时钟源，例如一个时序模型的发送、采样分别由 Clock0、Clock1 驱动，Clock0、

Clock1 来自不同的外部端口则可指定该选项。

图 3-22 设置时钟组



2. 单击“ ... ”按钮，为 Group 选择 Clock，如需删除添加的 Group，单击对应条目右侧的“ ✖ ”按钮；

3. 单击“OK”，保存约束。

注！

选项“Exclusive”与“Asynchronous”实现的作用相同。

3.7.2 I/O 约束

set_input_delay

设置数据输入的延迟时间值。用户需设定一个输入延迟时间值，软件会根据给定的延迟时间值进行时序分析。云源产生的时序报告中输入延迟时间类型为“tIn”。

set_output_delay

设置数据输出的延迟时间值。用户需设定一个输出延迟时间值，软件会根据给定的延迟时间值进行时序分析。

云源的时序报告中，输出延迟时间的类型为“tOut”。

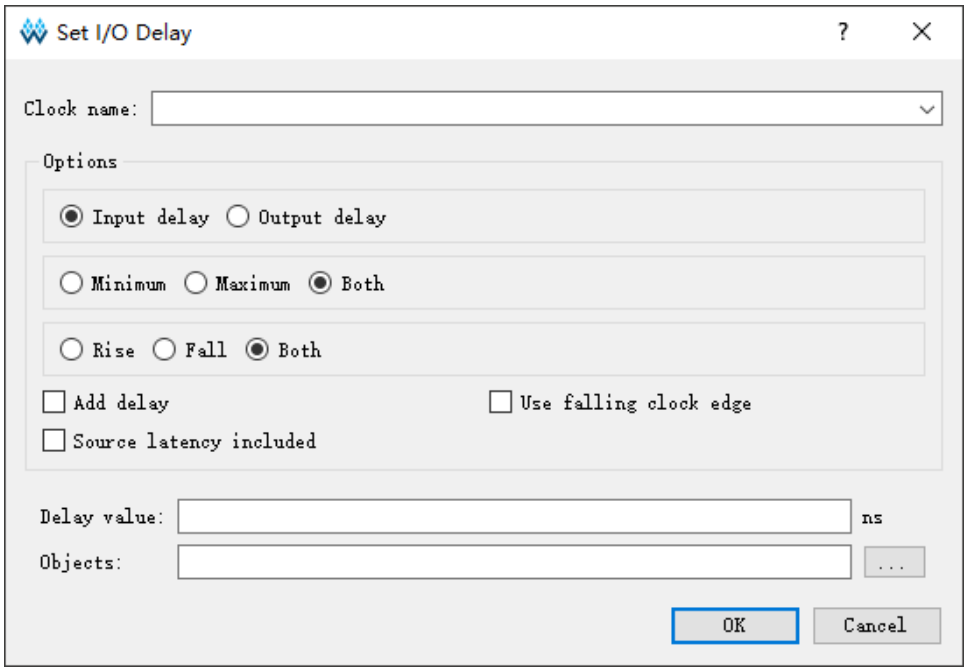
新建 I/O Delay 约束操作如下：

1. 在“Constraints”菜单中，选择“Set I/O Delay”，弹出“Set I/O Delay”对话框，如图 3-23 所示；
   - Clock name 指明 I/O 关联时钟的名称，需是存在的时钟，请使用右侧“ ⌄ ”进行选择；
   - Options 用来配置延迟时间类型、最大最小延迟时间、作用时钟边沿

等；

- Input delay、Output delay 指明输入或输出延迟时间类型，两者互斥；
- Minimum、Maximum 指明 I/O 的最小或最大延迟时间值，Both 表示两者延迟时间值一致；
- Rise、Fall 指明对上升沿或下降沿有效，Both 表示同时都有效；
- Delay value 设置 I/O 的延迟时间值，浮点型，精确到千分位，当为负数时表示提前到达，当为正数时表示推迟到达，单位 ns；
- Objects 指明输入输出端口，请使用右侧“...”按钮进行选择；
- Add delay 用以对同一个端口附加一个延迟时间值，当同一个端口存在多个延迟时间值时软件会选取最大的进行 setup 和 recovery 分析，选取最小的进行 hold 和 removal 分析，若不指定该选项则在同一端口上的相同约束会被覆盖；
- Use falling clock edge 被勾选后则指明与关联时钟的下降沿相关，默认上升沿相关；
- Source Latency include 被勾选后表示设置的延迟时间值中已经包含了时钟的延迟时间值，若不勾选则在计算时云源会算入时钟的延迟时间值。

图 3-23 创建 I/O Delay 约束



2. 配置填写完成后，单击“OK”保存约束。

3.7.3 例外约束

时序例外允许用户修改特定路径的默认静态时序分析规则，时序例外约束命令包含 set_false_path、set_max_delay、set_min_delay、set_multicycle_path 四种。

Set False Path

设置伪路径。不对设置的时序路径进行时序分析。

新建 False Path 约束操作如下：

1. 选择“Constraints > Set False Path”，弹出“Set False Path”对话框，如图 3-24 所示；
   - Analysis type 指明不分析的类型，Both 指都不分析；
   - From 指明路径的起点；
   - To 指明路径的终点；
   - Through 用于指明路径经过的点或线。

注！

From、To 及 Through 可单独也可相互配合使用。

图 3-24 创建 False Path 约束

2. 单击右侧按钮“ ... ”选择 From、To 以及 Through 对应的 Object，参考图 3-14，单击“OK”保存约束。

Set Max/Min Delay

设置最大、最小延迟时间值。

设置最大延迟时间时会在 setup 分析报告中进行报告，设置最小延迟时间时则在 hold 分析报告中进行报告。

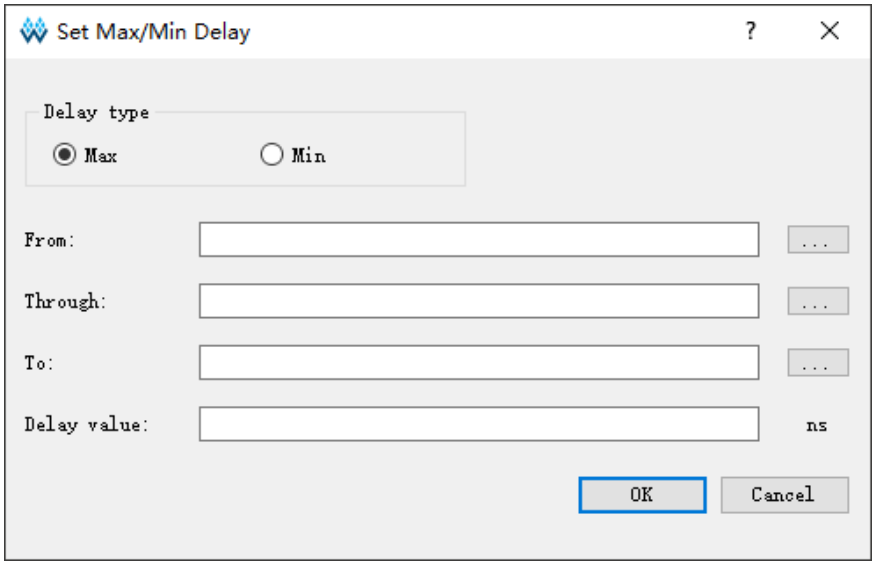
新建 Max/Min Delay 约束操作如下：

1. 选择“Constraints > Set Max/Min Delay”，弹出“Set Max/Min Delay”对话框，如图 3-25 所示；
   - From 参数用于规定路径的起点，请使用右侧“...”进行选择；
   - To 参数用于指定路径的终点，请使用右侧“...”进行选择；
   - Through 参数用于指定路径经过的点或线，请使用右侧“...”进行选择；
   - Delay value 由用户指定的延迟时间值，浮点型，精确到千分位，单位 ns。

注！

From、To 及 Through 可单独也可相互配合使用。

图 3-25 创建 Max/Min Delay 约束



2. Delay type 选择 Delay 的类型（Max 或 Min），From 和 To 选择对应的 Object。填写完成 Delay 信息后，单击“OK”完成创建。

Set Multicycle Path

设置多循环周期。

默认情况下，云源执行的是单周期时钟分析。

新建 Multicycle Path 约束操作如下：

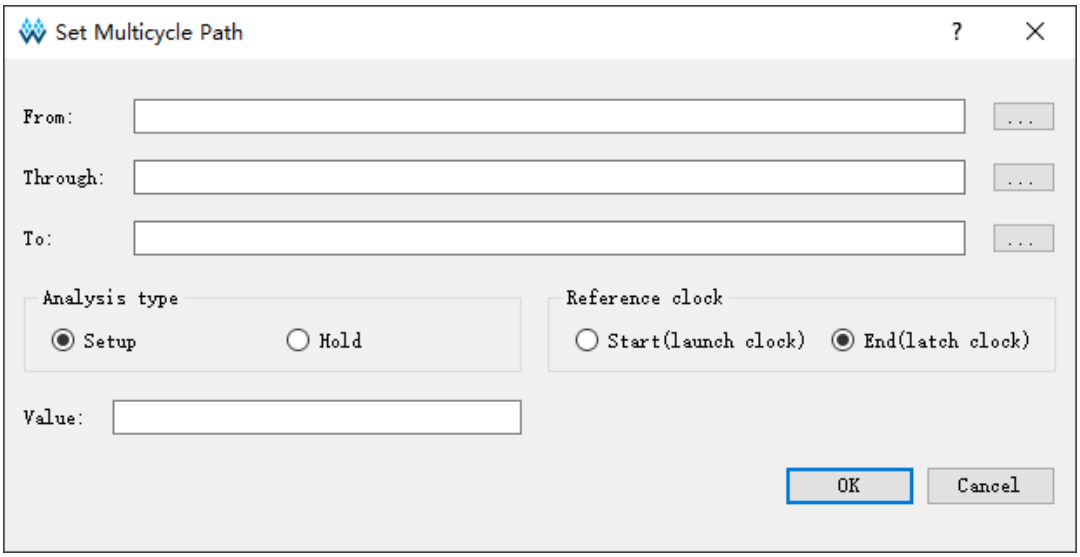
1. 选择“Constraints > Set Multicycle Path”，弹出“Set Multicycle Path”对话框，如图 3-26 所示；
   - Reference clock 指明参考时钟是发起时钟，还是锁存时钟；
   - Analysis type 指明约束对 setup 或 hold 检查；
   - From 用于指明路径的起点，请使用右侧“...”进行选择；

- Through 参数用于指定路径经过的 NODE，请使用右侧“ ”进行选择；
- To 用于指明路径的终点，请使用右侧“ ”进行选择；
- Value 指定多循环周期个数，正负整数，当为负数时表示提前，当为正数时表示推迟。

注！

From、To 及 Through 可单独也可相互配合使用。

图 3-26 创建 Multicycle Path 约束



2. 填写对话框中相关信息，单击“OK”保存约束。

3.7.4 工作条件约束

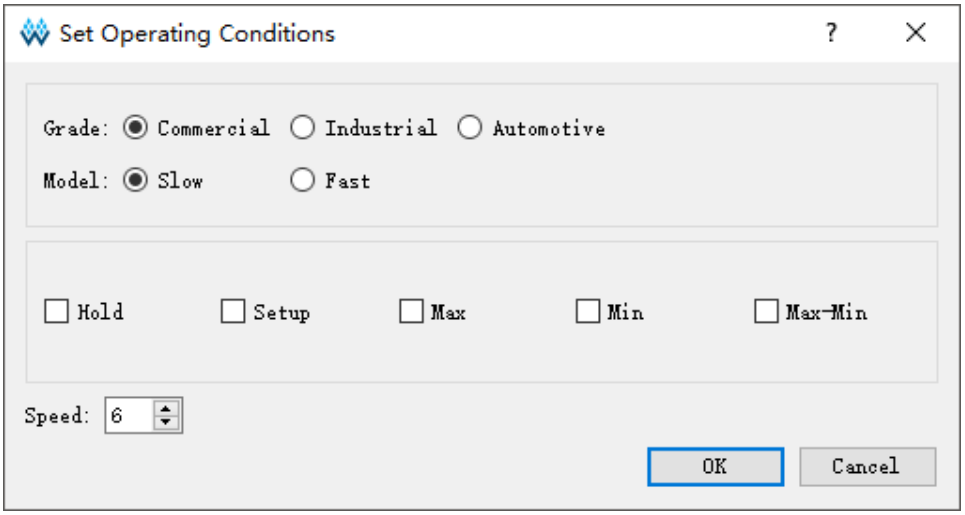
设置工作条件。

默认云源在进行 setup 分析时使用 Slow Model（慢速延迟时间模型），hold 分析时使用 Fast Model（快速延迟时间模型）。用户也可自定义时序延迟时间模型。完成后可在 STA Tool Run Summary 中查看使用的延迟时间模型。

选择“Constraints > Set Operating Conditions”，弹出“Set Operating Conditions”对话框，新建 Operating Conditions 约束，如图 3-27 所示。

- Grade 分为商业级、工业级以及车规级；
- Model 分为慢速、快速；
- Hold、Min 对 hold 和 removal 有效；
- Setup、Max 对 setup 和 recovery 有效；
- Max-Min 对 setup、hold、recovery、removal 有效。

图 3-27 创建 Operating Conditions 约束



注！

- 当设置的 Grade、Speed 与芯片型号不匹配时以实际约束为准；
- 若实际约束的 Grade、Speed 不支持当前工程则云源将提示警告信息；
- 若 Grade、Speed 仅设置了 setup，则 hold 按照 setup 设置的 Grade、Speed 进行分析；
- 若 Grade、Speed 仅设置了 hold，则 setup 按照 hold 设置的 Grade、Speed 进行分析；
- 工程样片（ES）默认使用最慢速度等级进行时序分析，建议用户自行设定相应值。

3.7.5 报告约束

Report Timing

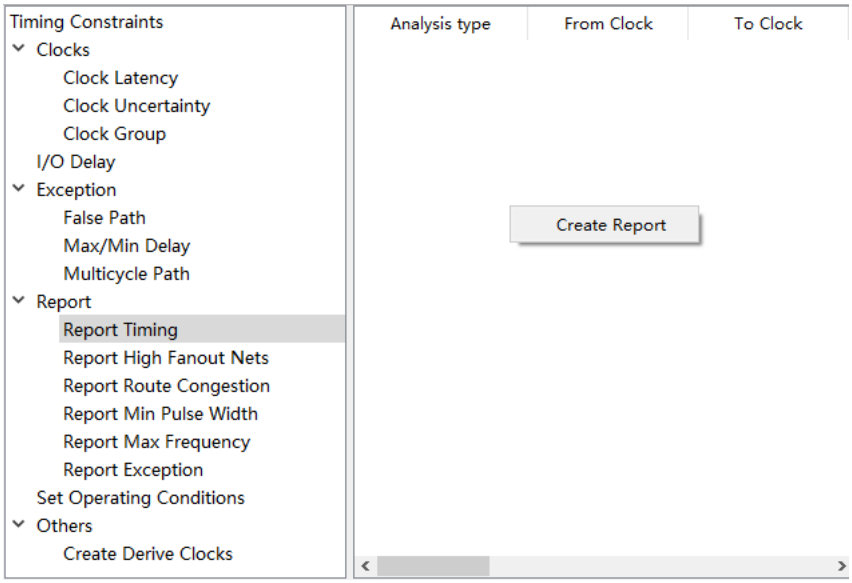
报告时序。

根据设置的参数，输出相应的报告内容，可实现更具体的时序报告与分析。产生的报告在 setup、hold 分析报告中进行查看。

操作步骤如下：

1. 在主窗口，选择“Timing Constraints > Report Timing”，空白处右击，出现“Create Report”，如图 3-28 所示；

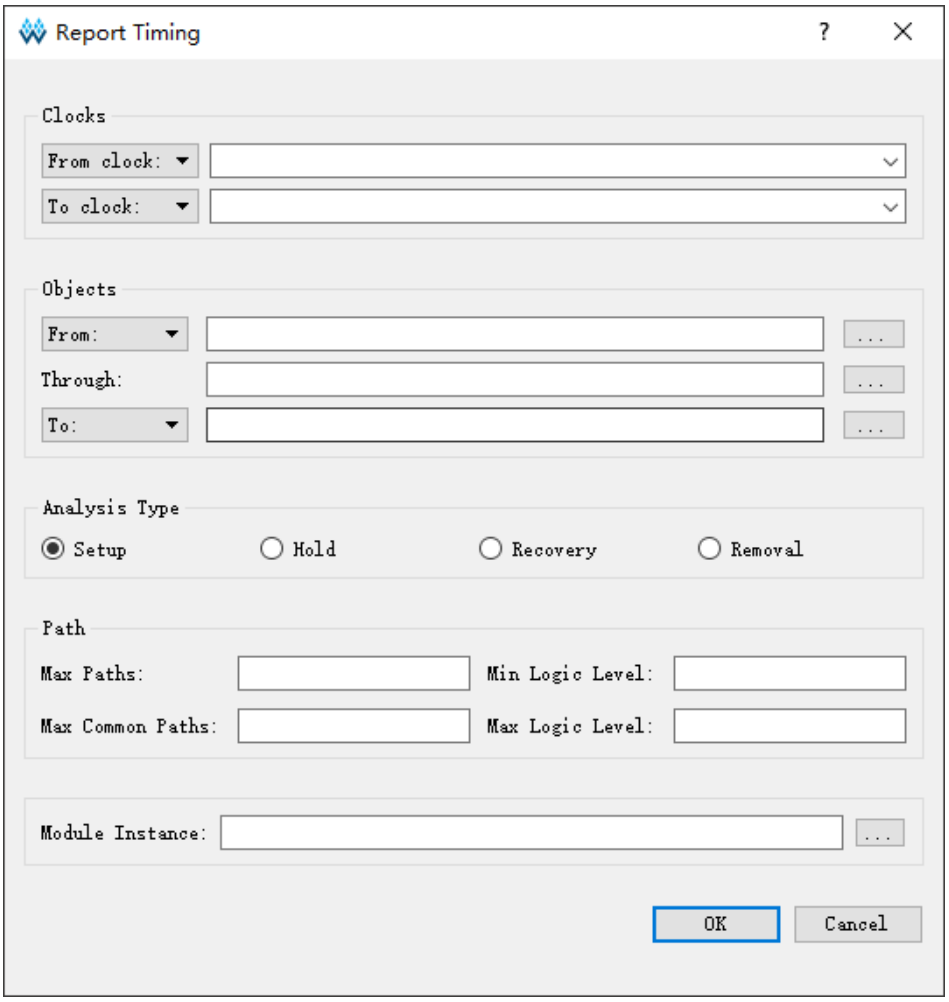
图 3-28 创建 Report Timing



2. 选择“Create Report”弹出如图 3-29 所示的对话框；

   - Path 指定时序报告的最大路径数（Max Paths）、最大共同路径（Max Common Paths）、最大最小逻辑级数（Max/Min Logic Level）；
   - Clocks 指明时序报告路径的关联时钟，From/To Clock 分别指明发送时钟、采样时钟，使用右侧“ ”按钮进行选择；
   - Objects 指明分析的起始和结束目标，请使用右侧“ ”按钮进行选择；
   - Analysis Type 指定时序报告检查的类型，分别为建立时间（setup）、保持时间（hold）、恢复时间（recovery）及移除时间（removal）；
   - Module Instance 指明报告的 Module 的实例化名称，请使用右侧“ ”按钮进行选择。

图 3-29 Report Timing 对话框



3. 填写对话框中相关信息，单击“OK”，保存时序报告设置。

Report High Fanout Nets

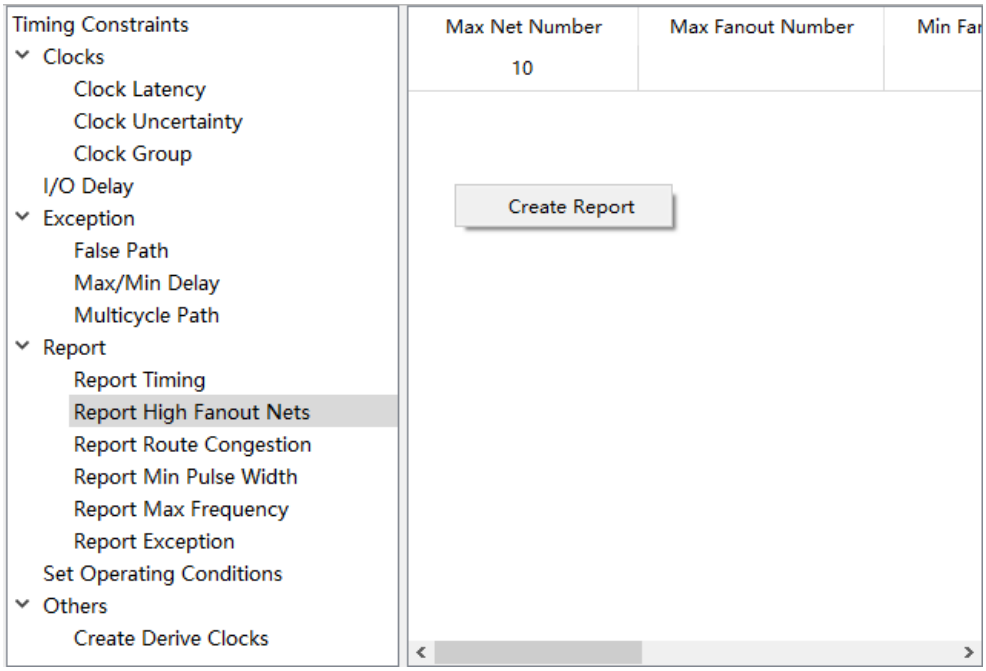
报告高扇出 Nets。

默认报告时序路径中 10 条扇出最高的 nets。产生的报告可在 High Fanout Nets Report 中进行查看。

操作步骤如下：

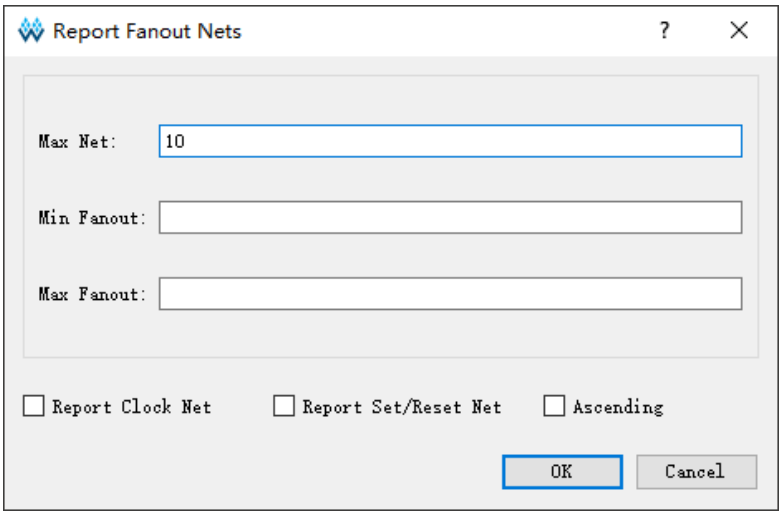
1. 在主界面中，选择“Timing Constraints > Report High Fanout Nets”；
2. 在右侧空白处右击弹出“Create Report”，如图 3-30 所示；

图 3-30 创建 Report High Fanout Nets



3. 选择“Create Report”，弹出如图 3-31 所示的对话框；
   - Max Net 指明最大报告的个数，正整数；
   - Min、Max Fanout 分别指明报告扇出的下限、上限，正整数；
   - Report Clock Net 报告连接时钟输入端的 Net；
   - Report Set/Reset Net 报告连接 LSR 输入端 Net；
   - Ascending 升序打印，默认降序。

图 3-31 Report High Fanout Nets 对话框



4. 填写对话框中相关信息，单击“OK”，保存时序报告设置。

Report Route Congestion

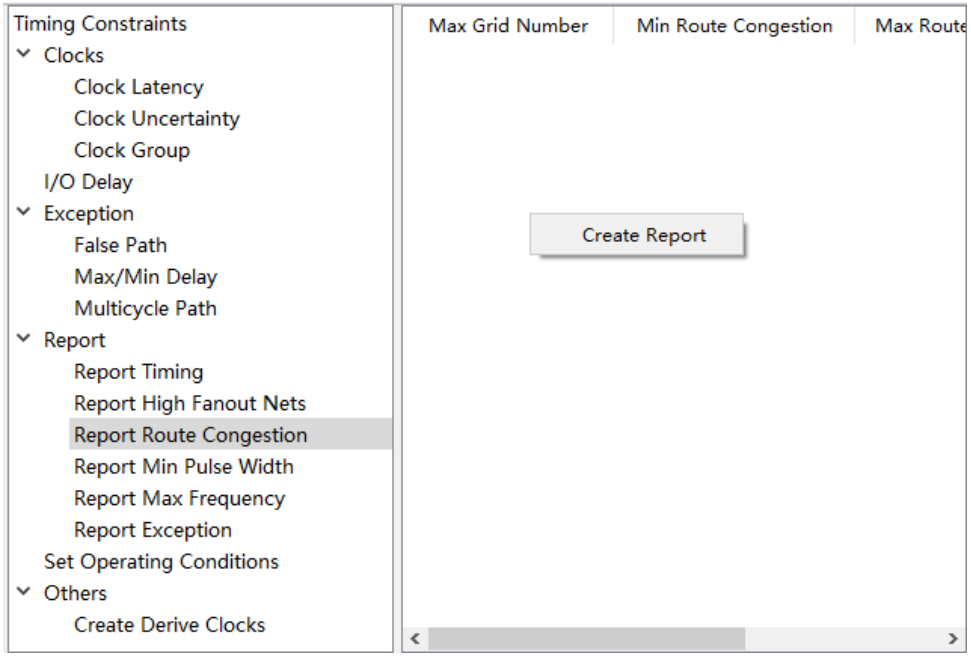
报告绕线拥塞度。

默认报告 10 个最差的 Grid。通常用在一个特定 Grid 上的绕线拥塞度报告。产生的报告请在 Route Congestions Report 中进行查看。

相关操作步骤如下：

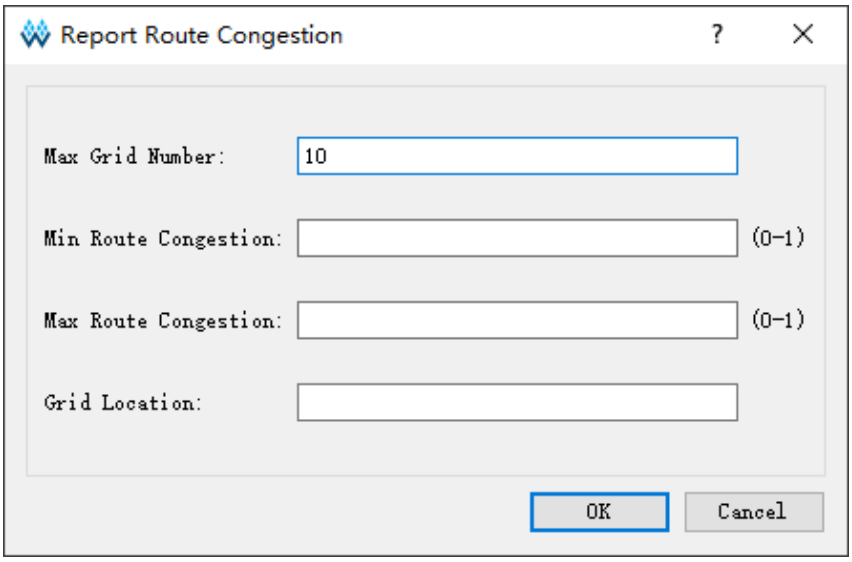
1. 在主界面中，选择“Timing Constraints > Report Route Congestion”；
2. 在右侧空白处右击，出现“Create Report”，如图 3-32 所示；

图 3-32 创建 Report Route Congestion



3. 选择“Create Report”，弹出如图 3-33 所示的对话框；
   - Max Grid Number 指明报告的个数；
   - Min、Max Route Congestion 分别指明绕线拥塞度的下限、上限，浮点型，精确到千分位；
   - Grid Location 指定报告的 Grid，如 R4C4。

图 3-33 Report Route Congestion 对话框



4. 填写对话框中相关信息，单击“OK”，保存时序报告设置。

Report Min Pulse Width

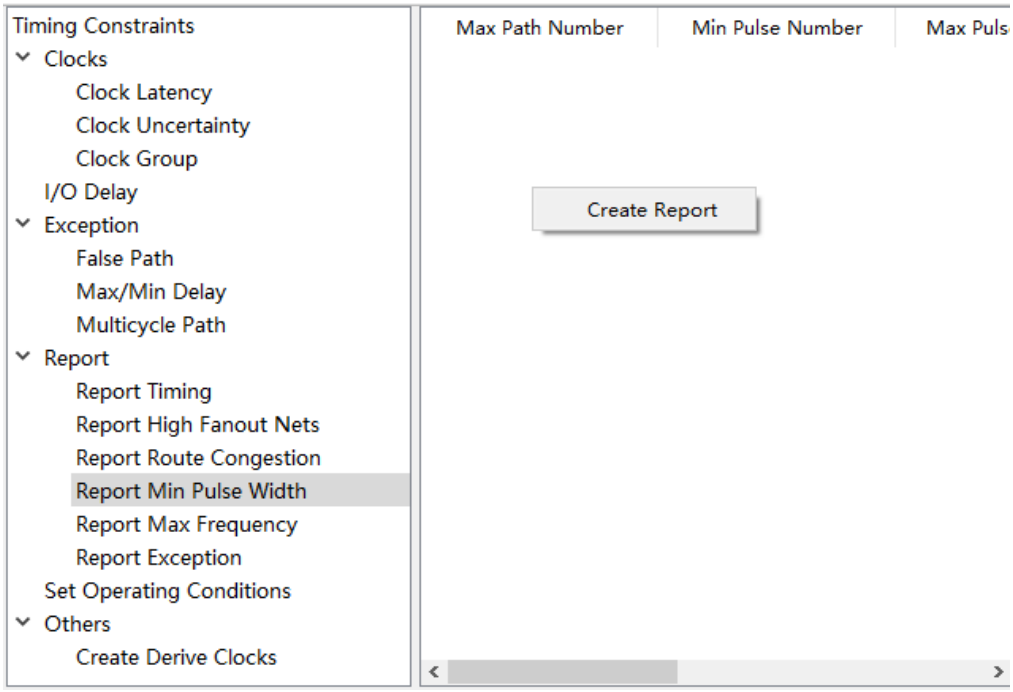
报告最小脉冲宽度。

默认报告 10 条。用户可使用该约束语句报告特定范围内的脉冲宽度或特定目标上的脉冲宽度。产生的报告请在 Minimum Pulse Width Report 查看。

操作参考如下：

1. 在主界面中，选择“Timing Constraints > Report Min Pulse Width”；
2. 在右侧空白处右击，出现“Create Report”，如图 3-34 所示；

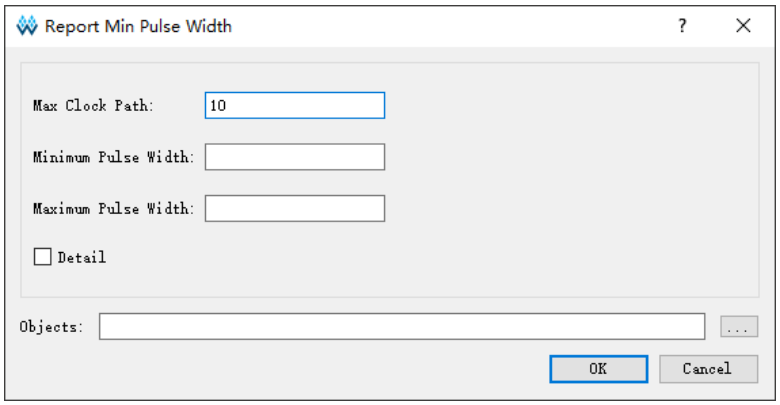
图 3-34 创建 Report Min Pulse Width



3. 选择“Create Report”出现如图 3-35 所示的对话框；

   - Max Clock Path 指明最大报告数，正整数；
   - Minimum、Maximum Pulse Width 分别指明报告的实际脉冲宽度的下限、上限，浮点型，精确到千分位；
   - Detail 指明是否报告详细路径；
   - Objects 指明需要报告的时序元件，仅支持 FF（如 DFF），请使用右侧“ ... ”按钮进行选择。

图 3-35 Report Min Pulse Width 对话框



4. 填写对话框中相关信息，单击“OK”，保存时序报告设置。

Report Max Frequency

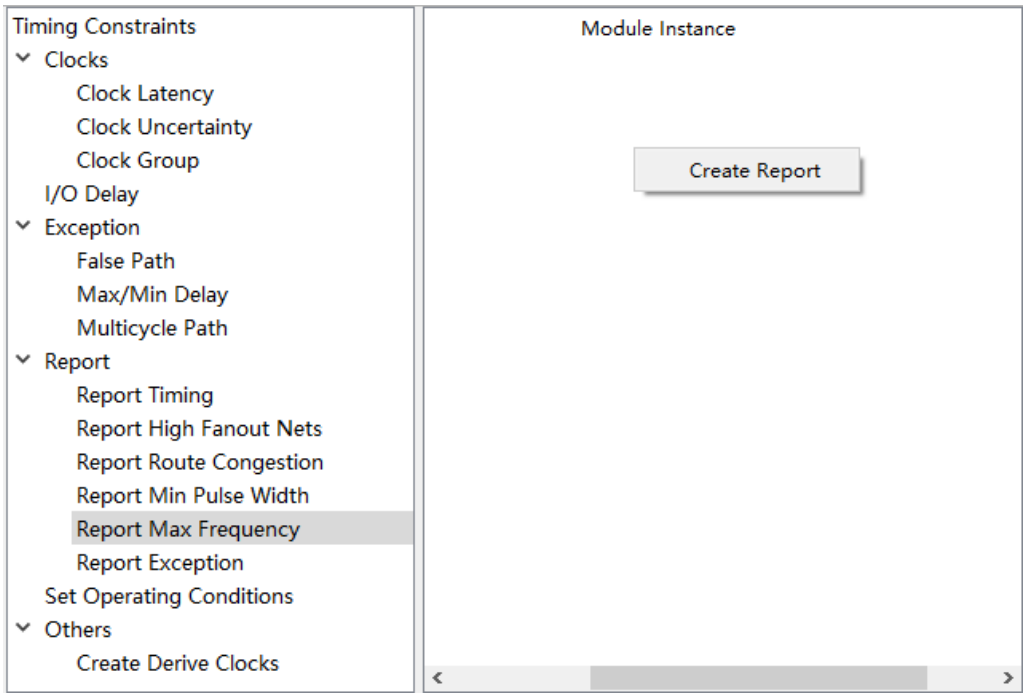
报告最大工作频率。

默认云源仅报告设计 Top 层的时钟最大频率。用户可指定报告一个特定的 module 的最大工作时钟频率，该 module 的最大工作时钟频率时序关键路径不限于本 module 内部但与本 module 时序终点相关。

操作步骤如下：

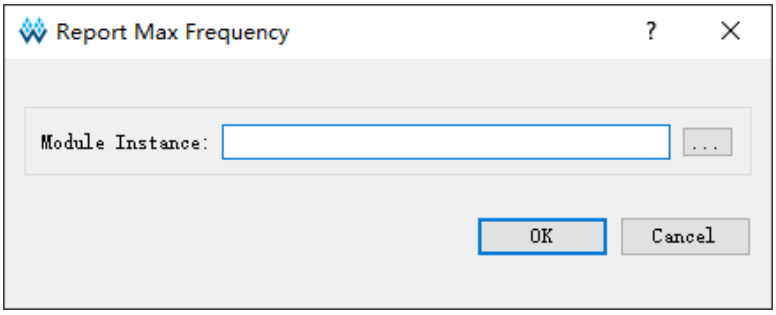
1. 在主界面中选择“Timing Constraints > Report > Report Max Frequency”；
2. 在右侧空白处右击，出现“Create Report”，如图 3-36 所示；

图 3-36 创建 Report Max Frequency



3. 选择“Create Report”，弹出如图 3-37 所示的对话框。“Module Instance”中为模块的实例化名称，使用右侧“...”按钮进行选择；

图 3-37 Report Max Frequency 对话框



4. 单击“OK”，保存时序报告设置。

Report Exception

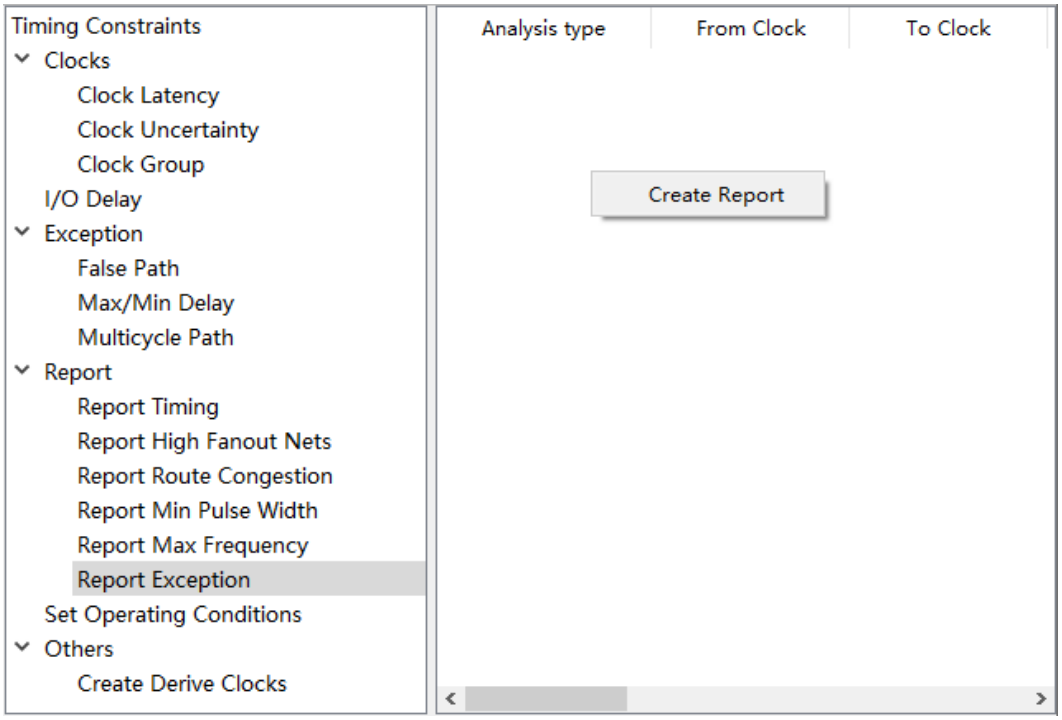
报告例外约束。

对例外约束语句作用的时序路径约束，以报告用户关心的时序路径。

相关操作步骤如下：

1. 在主界面中选择“Timing Constraints > Report > Report Exception”；
2. 在右侧空白处右击，出现“Create Report”，如图 3-38 所示；

图 3-38 创建 Report Exception

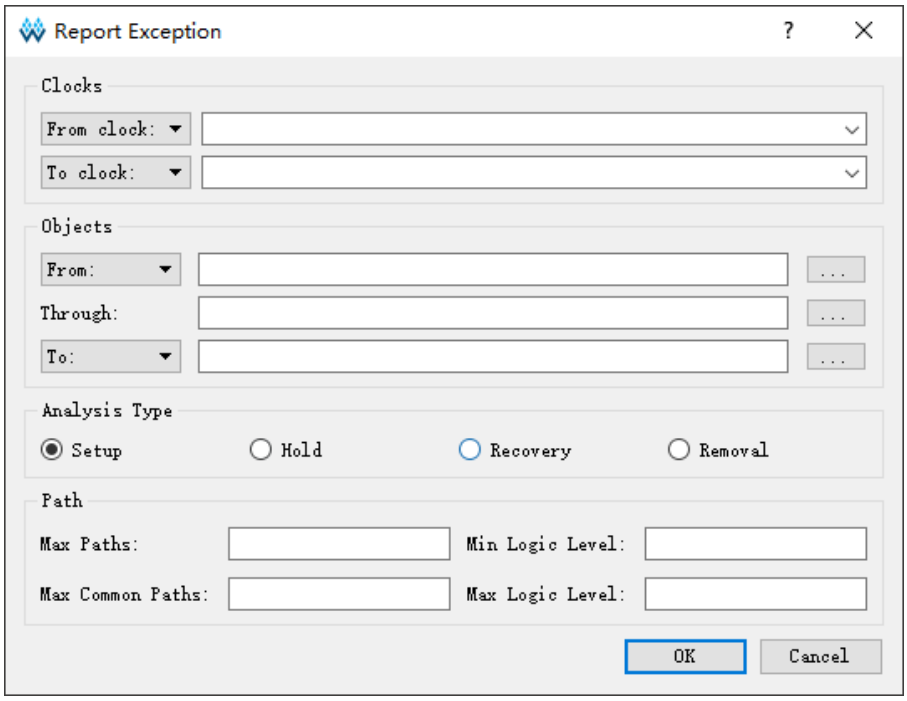


3. 选择“Create Report”，弹出如图 3-39 所示的对话框。

注！

窗口选项简介请参考 Report Timing。

图 3-39 Report Exception 对话框



4. 填写对话框中相关信息，单击“OK”，保存时序报告设置。

3.7.6 其它约束

Create Derive Clocks

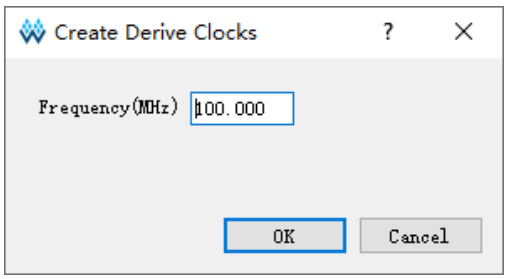
创建全局目标时钟频率，最大支持 1200MHz。

可通过以下两种方式新增 Derive Clocks 约束：

1. 通过 Constraints 菜单新增 Derive Clocks 约束。

   a). 选择“Constraints > Create Derive Clocks…”，弹出“Create Derive Clocks”对话框，如图 3-40 所示。

图 3-40 创建 Derive Clocks

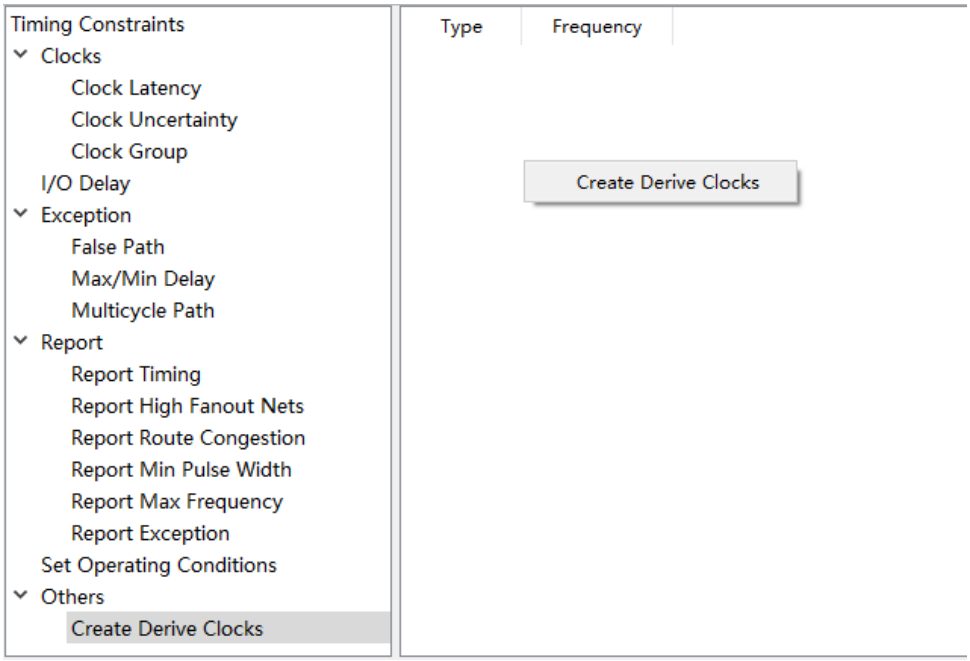


   b). Frequency(MHz)：全局目标频率，小于等于 1200 的正浮点数，精确到千分位。

2. 通过“Others > Create Derive Clocks”创建 Derive Clocks。在空白处

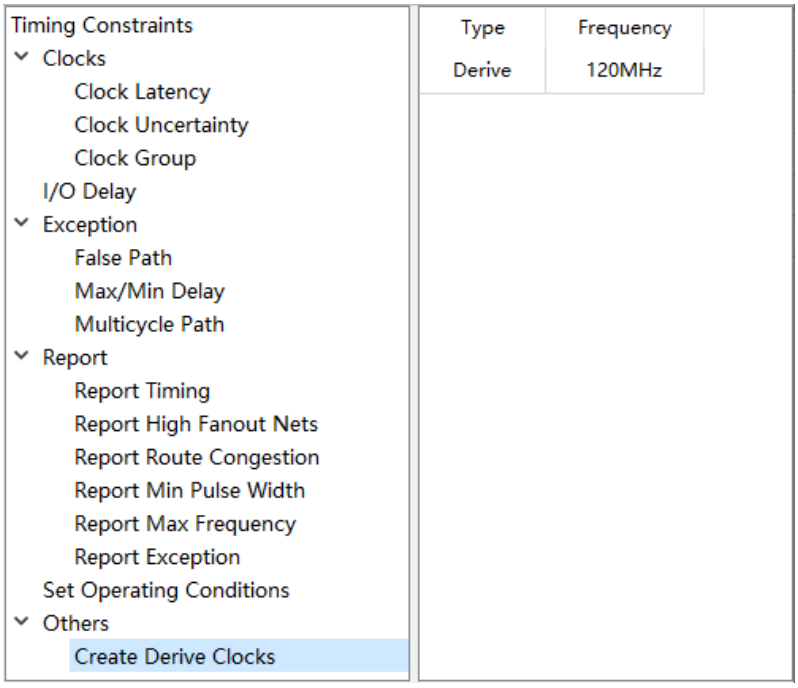
右击选择“Create Derive Clocks”新建 Derive Clocks，如图 3-41 所示。

图 3-41 选择 Create Derive Clocks



时钟创建完成后，Derive Clocks 列表中会增加对应的约束，如图 3-42 所示。

图 3-42 Derive Clocks 列表



3.7.7 保存与导出

所有约束编辑完成后，单击“File > Save”或“File > Save As”，可将当前时序约束编辑器中的约束信息保存至工程中的 SDC 文件中。

3.8 时序约束的优先级

云源提供多种类型的时序约束，按照优先级由低到高依次如下所示：

1. create_clock 和 create_generated_clock；
2. set_multicycle_path；
3. set_max_delay 和 set_min_delay；
4. set_false_path；
5. set_clock_groups。

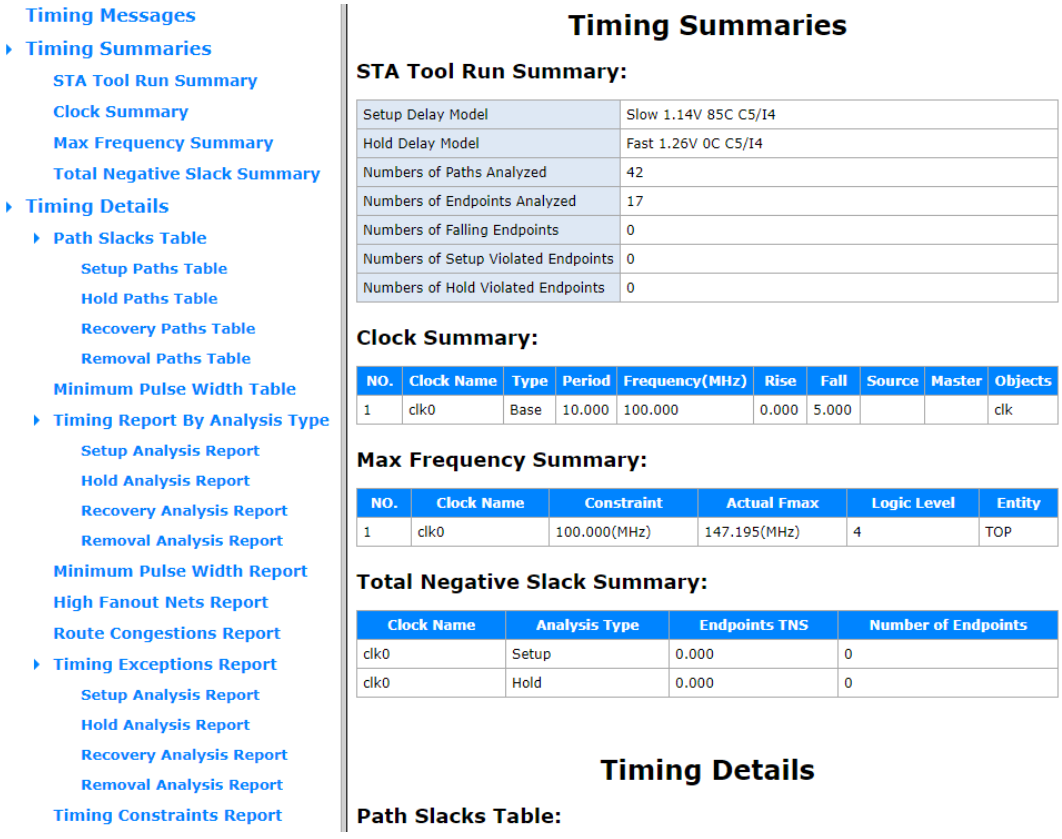
注！

只对在同一条时序路径上可能产生竞争的时序约束进行排序，其它未提及约束不会产生不同类约束间的竞争。

4 时序报告

本章节将对高云半导体静态时序分析结果报告进行描述，方便用户快速了解时序报告内容。如图 4-1 所示，报告分为左侧导航栏和右侧内容栏，当存在不满足时序分析情况时左侧的导航栏对应的标题会显示红色。

图 4-1 静态时序分析报告



4.1 Timing Summaries

时序综述（Timing Summaries）由四部分组成，分别是运行信息综述（STA Tool Run Summary）、时钟综述（Clock Summary）、最大频率综述（Max Frequency Summary）及终点余量为负值综述（Total Negative Slack

Summary），如下图 4-2 所示。

图 4-2 Timing Summaries

Timing Summaries

STA Tool Run Summary:

Setup Delay Model	Slow 1.14V 85C C5/I4
Hold Delay Model	Fast 1.26V 0C C5/I4
Numbers of Paths Analyzed	42
Numbers of Endpoints Analyzed	17
Numbers of Falling Endpoints	0
Numbers of Setup Violated Endpoints	0
Numbers of Hold Violated Endpoints	0

Clock Summary:

NO.	Clock Name	Type	Period	Frequency(MHz)	Rise	Fall	Source	Master	Objects
1	clk0	Base	10.000	100.000	0.000	5.000			clk

Max Frequency Summary:

NO.	Clock Name	Constraint	Actual Fmax	Logic Level	Entity
1	clk0	100.000(MHz)	147.195(MHz)	4	TOP

Total Negative Slack Summary:

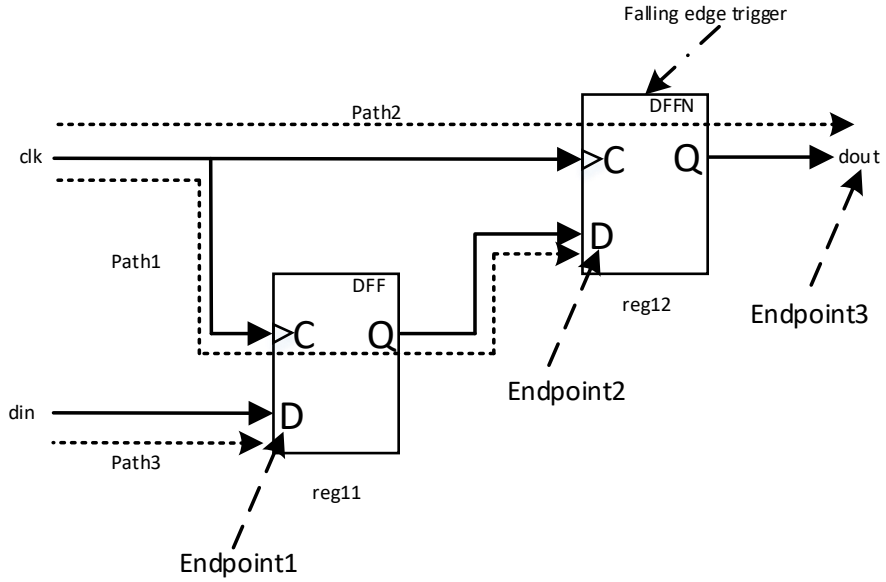
Clock Name	Analysis Type	Endpoints TNS	Number of Endpoints
clk0	Setup	0.000	0
clk0	Hold	0.000	0

4.1.1 STA Tool Run Summary

- Setup Delay Model：云源进行建立时间分析时使用的数据模型，默认采用 Slow 模型；
- Hold Delay Model：云源进行保持时间分析时使用的数据模型，默认采用 Fast 模型；
- Numbers of Paths Analyzed：静态时序分析路径的数量，如图 4-3 所示，共分析了 3 条时序路径，标记为 Path1、Path2 及 Path3；
- Numbers of Endpoints Analyzed：分析的时序路径的终点，如图 4-3 所示，共分析了 3 个终点，标记为 Endpoint1、Endpoint2 及 Endpoint3；
- Numbers of Falling Endpoints：终点分析时触发方式为下降沿的数量，如图 4-3 所示，reg12 类型为 DFFN，触发方式为下降沿，则终点 D 即是下降沿触发终点；

- Numbers of Setup Violated Endpoints：经时序分析后不满足建立时间的终点量；
- Numbers of Hold Violated Endpoints：经时序分析后不满足保持时间的终点量。

图 4-3 Path & Endpoints



4.1.2 Clock Summary

报告用户设计中的所有时钟。若设计中的时钟未进行约束则软件会默认为其创建时钟，晨熙家族默认时钟为 100MHz，小蜜蜂家族默认时钟为 50MHz。

- NO.：序号；
- Clock Name：时钟的名称。；
- Type：有 Base、Generated 两种值。Base 表示基础时钟，Generated 表示衍生时钟；
- Period：时钟的周期，单位 ns；
- Frequency (MHz) ：时钟频率；
- Rise：时钟的上升沿时间；
- Fall：时钟的下降沿时间；
- Source：时钟源，可从 PORT、PIN、NET、REG 进行时钟获取；
- Master：主时钟；
- Objects：时钟作用对象如 PORT、PIN、NET、REG。

注！

- 软件默认创建时钟时若时钟重名则会自动添加”_gowin”后缀。
- 若为 PLL 类、OSC 类及 CLKDIV 类，软件会默认为时钟名添加”.default_gen_clk”后缀

4.1.3 Max Frequency Summary

- NO.：序号；
- Clock Name：驱动时序模型的时钟的名称；
- Constraint：SDC 约束的时钟频率或无 SDC 约束时默认的时钟频率；
- Actual Fmax：云源 PnR 后经云源分析后得出的最大实际频率；
- Logic Level：逻辑级数；
- Entity：报告设计中模块的最大频率，默认为顶层模块即 TOP。

注！

- 当 PnR 后时钟没有驱动时序模型时则为“No timing paths to get frequency of *”；
- 最大时钟频率仅报告同步时钟；
- 建议用户为设计添加完整的时序约束使云源能够更加精准的进行分析。

4.1.4 Total Negative Slack Summary

- Clock Name：时钟名称；
- Analysis Type：分析类型为 setup 或 hold 两种；
- Endpoints TNS：统计时钟（对应 ClockName）驱动的时序路径上的终点余量为负值的时间总量，其中共同终点的路径只统计最差路径；
- Number of Endpoints：统计时钟（对应 ClockName）驱动的时序路径上的终点余量为负值的终点个数总量，其中共同终点的路径只统计最差路径。

4.2 Timing Details

4.2.1 Path Slacks Table

时序路径的静态分析余量表，分为 Setup Paths Table（建立时间路径分析表）、Hold Paths Table（保持时间路径分析表）、Recovery Paths Table（恢复时间路径分析表）、Removal Paths Table（移除时间路径分析表）。上述四类表头信息表达信息相同，图 4-4 表头说明如下：

- Path Number：路径编号，默认最大报告 25 条；
- Path Slack：其值等于数据所需时间减去数据到达时间，当为负值时时序不满足；

- From Node：前级时序元件的时序分析开始节点；
- To Node：后级时序元件的时序分析终止节点；
- From Clock：前级时序元件的数据发送时钟以及发送边沿类型。其中发送边沿类型指的是上升沿或下降沿；
- To Clock：后级时序元件的数据锁存时钟以及锁存边沿类型；
- Relation：描述发送时钟和采样时钟之间的时间关系；
- Clock Skew：时钟偏斜。发送时钟和锁存时钟到达前级和后级时序元件的时间差；
- Data Delay：数据到达路径中的数据延迟时间，其数值是整个数据到达路径中延迟时间值的一部分。

注！

- 当没有可供分析的时序路径时则报告为“Nothing to report!”；
- Path Slacks Table 默认打印最差的 25 条路径，如用户需要查看的路径不在 25 条范围内则可通过 SDC 约束命令 report_timing 进行报告，命令语法请参见 Report Timing；
- Path Slacks Table 默认打印包含跨时钟域时序路径，如不关心跨时钟域分析则可通过 set_clock_groups 或 set_false_path 进行配置。

图 4-4 路径余量表

Path Slacks Table:

Setup Paths Table

Report Command:report_timing -setup -max_paths 25 -max_common_paths 1

Path Number	Path Slack	From Node	To Node	From Clock	To Clock	Relation	Clock Skew	Data Delay
1	8.806	synS_r_s0/Q	synE_r_s0/D	ck0:[R]	ck0:[R]	10.000	0.000	0.794

Hold Paths Table

Report Command:report_timing -hold -max_paths 25 -max_common_paths 1

Path Number	Path Slack	From Node	To Node	From Clock	To Clock	Relation	Clock Skew	Data Delay
1	0.570	synS_r_s0/Q	synE_r_s0/D	ck0:[R]	ck0:[R]	0.000	0.000	0.570

Recovery Paths Table

Report Command:report_timing -recovery -max_paths 25 -max_common_paths 1

Path Number	Path Slack	From Node	To Node	From Clock	To Clock	Relation	Clock Skew	Data Delay
1	8.649	rstSrc_r_s0/Q	rstObj_r_s0/CLEAR	ck0:[R]	ck1:[R]	10.000	0.000	1.278

Removal Paths Table

Report Command:report_timing -removal -max_paths 25 -max_common_paths 1

Path Number	Path Slack	From Node	To Node	From Clock	To Clock	Relation	Clock Skew	Data Delay
1	0.788	rstSrc_r_s0/Q	rstObj_r_s0/CLEAR	ck0:[R]	ck1:[R]	0.000	0.000	0.833

4.2.2 Minimum Pulse Width Table

时序元件可识别的最小脉冲宽度静态时序分析表。脉冲宽度指的是有效高/低电平信号持续的时间长度。默认报告最差的 10 条。图 4-5 表头信息说明如下：

- Number：从小到大顺序序列号，默认 10 条；
- Slack：元件可识别的最小脉冲宽度的余量值；
- Actual Width：实际脉冲宽度，云源 PnR 后进行静态时序分析后得出的元件可识别的实际脉冲宽度；
- Required Width：要求脉冲宽度，元件要求的可正常识别的最小脉冲宽度；
- Type：脉冲宽度的类型，仅 Low Pulse Width 和 High Pulse Width 两种类型，分别为逻辑低电平脉冲宽度和逻辑高电平脉冲宽度；
- Clock：进行最小脉冲宽度分析的时钟；
- Objects：进行最小脉冲宽度分析的时序元件实例化对象。

注！

当无最小脉冲宽度分析报告时显示“Nothing to report!”。

图 4-5 最小脉冲宽度表

Minimum Pulse Width Table:

Report Command:report_min_pulse_width -nworst 10 -detail

Number	Slack	Actual Width	Required Width	Type	Clock	Objects
1	2.738	4.238	1.500	Low Pulse Width	DEFAULT_CLK	reg12
2	2.738	4.238	1.500	Low Pulse Width	DEFAULT_CLK	reg11_Z
3	2.813	4.313	1.500	High Pulse Width	DEFAULT_CLK	reg12
4	2.813	4.313	1.500	High Pulse Width	DEFAULT_CLK	reg11_Z

4.2.3 Timing Report By Analysis Type

该部分包含 Setup Analysis Report、Hold Analysis Report、Recovery Analysis Report、removal Analysis Report 四类静态时序分析类型。

Setup Analysis Report

建立时间分析报告，用来分析设计中时序元件的时钟信号上升沿到达前，数据稳定不变的时间，如时间不够，数据将不能在时钟上升沿被稳定的送入时序元件。

云源对时序路径上的数据到达时间、数据所需时间、采样时钟、发送时钟等进行了详细的计算、分析并最终打印在建立时间分析报告中供用户参考。

该报告由命令 report_timing -setup 生成，云源默认分析并报告 25 条余量最差的时序路径，内容包含 Path Summary、Data Arrival Path、Path Statistics，释义如下所示。

1. Path Summary。图 4-6 为静态时序分析的路径信息综述，图中信息说明如下；
   - Slack：数据允许最迟到达时间减去数据实际到达时间。正值表示时序收敛，负值表示时序不收敛；
   - Data Arrival Time：数据到达路径的时间；

- Data Required Time：数据所需时间；
- From：前级时序元件；
- To：后级时序元件；
- Launch Clock：发送时钟。作用边沿分为 R（Rise，上升沿）和 F（Fall，下降沿）两种；
- Latch Clock：采样时钟。作用边沿分为 R 和 F。

图 4-6 路径信息综述

Path Summary:

Slack	5.789
Data Arrival Time	6.767
Data Required Time	12.556
From	reg11_Z
To	reg12_Z
Launch Clk	sysclk1:[R]
Latch Clk	sysclk1:[R]

2. Data Arrival Path。图 4-7 为一条数据到达路径，图中信息说明如下；
   - AT：指某一时刻，是时序路径上的一个时间节点；
   - DELAY：指延迟时间值，其值表示一段时间间隔；
   - TYPE：指时序分析路径上 NODE 的类型，当为空值表示不可用。
   - RF：指的是当前被分析元件的信号翻转类型；
   - FANOUT：扇出；
   - LOC：当前分析的元件在器件中的物理位置，没有位置信息的使用 UNPLACE 标记，如 DHCEN；
   - NODE：静态时序分析路径上的节点。包括实例化的名称加端口、时钟、时钟边沿激活时间（active clock edge time）。

   注！

   图 4-7 中，TYPE 包含多种类型，含义如下：
   - tCL：time of clock latency，时钟源延迟时间；
   - tINS：time of module instance，元件延迟时间；
   - tNET：time of net，net 的延迟时间；
   - tIn：time of input，数据输入延迟时间；
   - tC2Q：time of clock to output，时序元件时钟到输出的延迟时间。

图 4-7 数据到达路径

Data Arrival Path:

AT	DELAY	TYPE	RF	FANOUT	LOC	NODE
0.000	0.000					active clock edge time
0.000	0.000					sysclk1
0.000	0.000	tCL	RR	1	IOL7[A]	clk1_ibuf/I
0.943	0.943	tINS	RR	2	IOL7[A]	clk1_ibuf/O
3.236	2.293	tNET	RR	1	IOL2[B]	reg11_Z/CLK
3.786	0.550	tC2Q	RF	1	IOL2[B]	**reg11_Z/Q**
6.767	2.981	tNET	FF	1	R5C9[1][A]	**reg12_Z/D**

3. Data Required Path。如图 4-8 所示，数据所需路径是指时钟从有效沿开始到达时序元件时钟端口时所经过的路径。

   注！

   图 4-8 中，TYPE 含义如下：

   - tUnc：time of clock uncertainty，时钟的不确定值；
   - tOut：time of output，数据输出延迟时间；
   - tSu：time of setup，建立时间。

图 4-8 数据所需路径

Data Required Path:

AT	DELAY	TYPE	RF	FANOUT	LOC	NODE
10.000	10.000					active clock edge time
10.000	0.000					sysclk1
10.000	0.000	tCL	RR	1	IOL7[A]	clk1_ibuf/I
10.943	0.943	tINS	RR	2	IOL7[A]	clk1_ibuf/O
13.236	2.293	tNET	RR	1	R5C9[1][A]	reg12_Z/CLK
13.036	-0.200	tUnc				reg12_Z
12.556	-0.480	tSu		1	R5C9[1][A]	reg12_Z

4. Path Statistics。图 4-9 为路径统计信息，图中信息说明如下。

   - Clock Skew：时钟倾斜；
   - Setup Relationship：前级时序元件发送数据，后级时序元件锁存数据的时间关系；
   - Logic Level：逻辑级数；
   - Arrival Clock Path Delay：统计了 Data Arrival Path 上时钟延迟时间的情况，cell 表示逻辑元件延迟时间，route 表示绕线延迟时间，tC2Q 表示时序元件内部延迟时间；
   - Arrival Data Path Delay：统计了 Data Arrival Path 上数据的延迟时间情况；
   - Required Clock Path Delay：统计了 Data Required Path 上时钟的延迟时间情况。

图 4-9 路径统计信息

Path Statistics:

Clock Skew	0.000
Setup Relationship	10.000
Logic Level	1
Arrival Clock Path Delay	cell: 0.943, 29.131%; route: 2.293, 70.869%
Arrival Data Path Delay	cell: 0.000, 0.000%; route: 2.981, 84.423%; tC2Q: 0.550, 15.577%
Required Clock Path Delay	cell: 0.943, 29.131%; route: 2.293, 70.869%

Hold Analysis Report

图 4-10 为保持时间分析报告，分析在时序元件的时钟信号上升沿到达之后，数据稳定不变的时间，如时间不足，数据不能被稳定的送入时序元件。云源对设计中的时序路径上的数据到达时间、数据所需时间、采样时钟、发送时钟等进行了详细的计算、分析并最终生成报告。该报告由命令 report_timing -hold 生成，默认报告 25 条余量最差的时序路径。报告表头信息解释请参考 Setup Analysis Report。

图 4-10 保持时间分析报告

Hold Analysis Report

Report Command:report_timing -hold -max_paths 25 -max_common_paths 1

Path1

Path Summary:

Slack	1.003
Data Arrival Time	3.554
Data Required Time	2.551
From	reg11_s0
To	reg12_s0
Launch Clk	sysyclk:[R]
Latch Clk	sysyclk:[R]

Data Arrival Path:

AT	DELAY	TYPE	RF	FANOUT	LOC	NODE
0.000	0.000					active clock edge time
0.000	0.000					sysyclk
0.000	0.000	tCL	RR	1	IOL11[A]	clk_ibuf/I
0.811	0.811	tINS	RR	2	IOL11[A]	clk_ibuf/O
2.533	1.723	tNET	RR	1	R2C9[0][A]	reg11_s0/CLK
2.933	0.400	tC2Q	RR	1	R2C9[0][A]	**reg11_s0/Q**
3.554	0.621	tNET	RR	1	R2C9[1][A]	**reg12_s0/CLEAR**

Data Required Path:

AT	DELAY	TYPE	RF	FANOUT	LOC	NODE
0.000	0.000					active clock edge time
0.000	0.000					sysyclk
0.000	0.000	tCL	RR	1	IOL11[A]	clk_ibuf/I
0.811	0.811	tINS	RR	2	IOL11[A]	clk_ibuf/O
2.533	1.723	tNET	RR	1	R2C9[1][A]	reg12_s0/CLK
2.533	0.000	tUnc				reg12_s0
2.551	0.018	tHld		1	R2C9[1][A]	reg12_s0

Path Statistics:

Clock Skew	0.000
Hold Relationship	0.000
Logic Level	1
Arrival Clock Path Delay	cell: 0.811, 31.998%; route: 1.723, 68.002%
Arrival Data Path Delay	cell: 0.000, 0.000%; route: 0.621, 60.818%; tC2Q: 0.400, 39.182%
Required Clock Path Delay	cell: 0.811, 31.998%; route: 1.723, 68.002%

Recovery Analysis Report

图 4-11 为恢复时间分析报告，指分析时序元件在时钟有效沿前，异步清零、置位、复位信号需保持稳定的最短时间，如不满足该时间，则时序元件可能无法进入正常工作状态。恢复时间的分析、计算方法与建立时间一致，该报告由命令 report_timing -recovery 生成，云源默认分析并报告 25 条余量最差的时序路径，表头信息请参考 Setup Analysis Report。

图 4-11 恢复时间分析报告

Recovery Analysis Report

Report Command:report_timing -recovery -max_paths 25 -max_common_paths 1

Path1

Path Summary:

Slack	8.355
Data Arrival Time	4.629
Data Required Time	12.984
From	reg11_s0
To	reg12_s0
Launch Clk	sysyclk:[R]
Latch Clk	sysyclk:[R]

Data Arrival Path:

AT	DELAY	TYPE	RF	FANOUT	LOC	NODE
0.000	0.000					active clock edge time
0.000	0.000					sysyclk
0.000	0.000	tCL	RR	1	IOL11[A]	clk_ibuf/I
0.943	0.943	tINS	RR	2	IOL11[A]	clk_ibuf/O
3.236	2.293	tNET	RR	1	R2C9[0][A]	reg11_s0/CLK
3.786	0.550	tC2Q	RF	1	R2C9[0][A]	**reg11_s0/Q**
4.629	0.843	tNET	FF	1	R2C9[1][A]	**reg12_s0/CLEAR**

Data Required Path:

AT	DELAY	TYPE	RF	FANOUT	LOC	NODE
10.000	10.000					active clock edge time
10.000	0.000					sysyclk
10.000	0.000	tCL	RR	1	IOL11[A]	clk_ibuf/I
10.943	0.943	tINS	RR	2	IOL11[A]	clk_ibuf/O
13.236	2.293	tNET	RR	1	R2C9[1][A]	reg12_s0/CLK
13.036	-0.200	tUnc				reg12_s0
12.984	-0.052	tSu		1	R2C9[1][A]	reg12_s0

Path Statistics:

Clock Skew	0.000
Setup Relationship	10.000
Logic Level	1
Arrival Clock Path Delay	cell: 0.943, 29.131%; route: 2.293, 70.869%
Arrival Data Path Delay	cell: 0.000, 0.000%; route: 0.843, 60.531%; tC2Q: 0.550, 39.469%
Required Clock Path Delay	cell: 0.943, 29.131%; route: 2.293, 70.869%

Removal Analysis Report

图 4-12 为移除时间分析报告，分析时序元件在时钟有效沿后，异步清零、置位、复位信号需保持稳定的最短时间，如不满足该时间，则时序元件可能无法进入正常工作状态。移除时间的分析、计算方法与保持时间一致，该报告由命令 report_timing -removal 生成，云源默认分析并报告 25 条余量最差的时序路径，表头信息请参考 Hold Analysis Report。

图 4-12 移除时间分析报告

Removal Analysis Report

Report Command:report_timing -removal -max_paths 25 -max_common_paths 1

Path1

Path Summary:

Slack	1.003
Data Arrival Time	3.554
Data Required Time	2.551
From	reg11_s0
To	reg12_s0
Launch Clk	sysyclk:[R]
Latch Clk	sysyclk:[R]

Data Arrival Path:

AT	DELAY	TYPE	RF	FANOUT	LOC	NODE
0.000	0.000					active clock edge time
0.000	0.000					sysyclk
0.000	0.000	tCL	RR	1	IOL11[A]	clk_ibuf/I
0.811	0.811	tINS	RR	2	IOL11[A]	clk_ibuf/O
2.533	1.723	tNET	RR	1	R2C9[0][A]	reg11_s0/CLK
2.933	0.400	tC2Q	RR	1	R2C9[0][A]	**reg11_s0/Q**
3.554	0.621	tNET	RR	1	R2C9[1][A]	**reg12_s0/CLEAR**

Data Required Path:

AT	DELAY	TYPE	RF	FANOUT	LOC	NODE
0.000	0.000					active clock edge time
0.000	0.000					sysyclk
0.000	0.000	tCL	RR	1	IOL11[A]	clk_ibuf/I
0.811	0.811	tINS	RR	2	IOL11[A]	clk_ibuf/O
2.533	1.723	tNET	RR	1	R2C9[1][A]	reg12_s0/CLK
2.533	0.000	tUnc				reg12_s0
2.551	0.018	tHld		1	R2C9[1][A]	reg12_s0

Path Statistics:

Clock Skew	0.000
Hold Relationship	0.000
Logic Level	1
Arrival Clock Path Delay	cell: 0.811, 31.998%; route: 1.723, 68.002%
Arrival Data Path Delay	cell: 0.000, 0.000%; route: 0.621, 60.818%; tC2Q: 0.400, 39.182%
Required Clock Path Delay	cell: 0.811, 31.998%; route: 1.723, 68.002%

4.2.4 Minimum Pulse Width Report

最小脉冲宽度报告分析所有参与时序分析的路径上时序元件的最小脉冲宽度，包括高电平最小脉冲和低电平最小脉冲两种。如图 4-13 所示，图中信息说明如下：

- Actual Width：实际脉冲宽度，其值为被分析目标上脉冲宽度实际维持的时间长度即是 Early clock Path 减去 Late clock Path 的值；
- Required Width：元件要求的最小识别宽度即是脉冲信号维持的最小时间，小于这个宽度，那么这个电平脉冲将不能被识别；
- Slack：脉冲宽度余量，其值为实际脉冲宽度减去请求脉冲宽度。计算公式为 “$Slack= (T_{Early\_clock\_Path} - T_{Late\_clock\_Path})-T$ Required Width”；
- Type：指明脉冲类型。有两种值 Low Pulse Width 与 High Pulse Width，分别为低脉冲宽度和高脉冲宽度；
- Clock：进行静态时序分析的时钟；
- Objects：当前分析的时序元件；

- Late clock Path：时钟最晚到达路径；
- Early clock Path：时钟最早到达路径。

图 4-13 最小脉冲宽度

MPW Summary:

Slack:	2.738
Actual Width:	4.238
Required Width:	1.500
Type:	Low Pulse Width
Clock:	sysclk1
Objects:	reg12_Z

Late clock Path:

AT	DELAY	TYPE	RF	NODE
5.000	0.000			active clock edge time
5.000	0.000			sysclk1
5.000	0.000	tCL	FF	clk1_ibuf/I
5.945	0.945	tINS	FF	clk1_ibuf/O
8.295	2.350	tNET	FF	reg12_Z/CLK

Early clock Path:

AT	DELAY	TYPE	RF	NODE
10.000	0.000			active clock edge time
10.000	0.000			sysclk1
10.000	0.000	tCL	RR	clk1_ibuf/I
10.811	0.811	tINS	RR	clk1_ibuf/O
12.533	1.723	tNET	RR	reg12_Z/CLK

4.2.5 High Fanout Nets Report

高扇出报告分析所有参与时序分析的路径中 net 的扇出情况，同时还会分析这个 net 的最差 Slack、最大延迟时间，默认分析 10 条。依照 FANOUT 值由大到小顺序排序，如图 4-14 所示，图中信息说明如下：

- FANOUT：扇出；
- NET NAME：指明当前分析的 net 名称；
- WORST SLACK：指明当前分析的 net 上所存在的最差 Slack，一条 net 上可能存在不止一个 Slack；
- MAX DELAY：指明当前分析 net 上最大延迟时间。

图 4-14 高扇出报告

High Fanout Nets Report:

Report Command:report_high_fanout_nets -max_nets 10

FANOUT	NET NAME	WORST SLACK	MAX DELAY
2	clk1_c	5.789	2.350
2	clk2_c	17.616	2.350
1	reg21_i	17.616	0.000
1	reg11	5.789	2.981
1	reg21	17.616	0.403

4.2.6 Route Congestions Report

图 4-15 为绕线拥塞报告，图中信息说明如下：

- GRID LOC：分析的 Grid 位置；

- ROUTE CONGESTIONS：Grid 上绕线的拥塞度，如 0.056 表示该 Grid 上的拥塞度为 5.6%；
- 默认报告 10 条最差的，按照 ROUTE CONGESTIONS 值的大小由大到小顺序排列。

图 4-15 绕线拥塞报告

Route Congestions Report:

Report Command:report_route_congestion -max_grids 10

GRID LOC	ROUTE CONGESTIONS
R5C9	0.056
R2C1	0.028
R3C1	0.028
R3C9	0.028
R1C1	0.014
R5C1	0.014

4.2.7 Timing Exceptions Report

下面通过一个实际案例进行时序例外报告说明。

针对图 4-16 案例，设计一个特定的 SDC 文件，文件内容如图 4-17 所示。

图 4-16 测试案例

```
module timing(
output dout,
input din, clk1, clk2
);

reg reg11, reg12;
reg reg21, reg22;


always @(posedge clk1)
begin
    reg11 <= din;
    reg12 <= reg11;
end

always @(posedge clk2)
begin
    reg21 <= din;
    reg22 <= ~reg21;
end

assign dout = reg22 & reg12;

endmodule
```

图 4-17 Timing Exceptions 约束

```
create_clock -name sysclk1 -period 10 -waveform {0 5} [get_ports {clk1}]
create_clock -name sysclk2 -period 10 -waveform {0 5} [get_ports {clk2}]
set_max_delay -from [get_clocks {sysclk1}] -to [get_clocks {sysclk1}] 5
set_max_delay -from [get_clocks {sysclk2}] -to [get_clocks {sysclk2}] 4
```

图 4-17 中的时序例外约束语句 set_max_delay，是将 sysclk1、sysclk2 影响的时序路径上的最大绝对延迟时间值分别设定为 5ns、4ns。set_max_delay 会对 setup 分析产生影响且受影响的路径会默认显示在时序例外报告下，默认产生的报告如下图 4-18 所示。

图 4-18 时序例外报告

Timing Exceptions Report:

Setup Analysis Report

Report Command:report_exceptions -setup -max_paths 5 -max_common_paths 1

Timing Path Constraint[1]: set_max_delay -from [get_clocks {sysclk1}] -to [get_clocks {sysclk1}] 5

Path1

Path Summary:

Slack	0.789
Data Arrival Time	6.767
Data Required Time	7.556
From	reg11_Z
To	reg12_Z
Launch Clk	sysclk1:[R]
Latch Clk	sysclk1:[R]

Data Arrival Path:

AT	DELAY	TYPE	RF	FANOUT	LOC	NODE
0.000	0.000					active clock edge time
0.000	0.000					sysclk1
0.000	0.000	tCL	RR	1	IOL7[A]	clk1_ibuf/I
0.943	0.943	tINS	RR	2	IOL7[A]	clk1_ibuf/O
3.236	2.293	tNET	RR	1	IOL2[B]	reg11_Z/CLK
3.786	0.550	tC2Q	RF	1	IOL2[B]	**reg11_Z/Q**
6.767	2.981	tNET	FF	1	R5C9[1][A]	**reg12_Z/D**

Data Required Path:

AT	DELAY	TYPE	RF	FANOUT	LOC	NODE
5.000	5.000					active clock edge time
5.000	0.000					sysclk1
5.000	0.000	tCL	RR	1	IOL7[A]	clk1_ibuf/I
5.943	0.943	tINS	RR	2	IOL7[A]	clk1_ibuf/O
8.236	2.293	tNET	RR	1	R5C9[1][A]	reg12_Z/CLK
8.036	-0.200	tUnc				reg12_Z
7.556	-0.480	tSu		1	R5C9[1][A]	reg12_Z

Path Statistics:

Clock Skew	0.000
Setup Relationship	5.000
Logic Level	1
Arrival Clock Path Delay	cell: 0.943, 29.131%; route: 2.293, 70.869%
Arrival Data Path Delay	cell: 0.000, 0.000%; route: 2.981, 84.423%; tC2Q: 0.550, 15.577%
Required Clock Path Delay	cell: 0.943, 29.131%; route: 2.293, 70.869%

Timing Path Constraint[14]: set_max_delay -from [get_clocks {sysclk2}] -to [get_clocks {sysclk2}] 4

Path1

Path Summary:

Slack	1.616
Data Arrival Time	4.940
Data Required Time	6.556
From	reg21_Z
To	reg22_Z
Launch Clk	sysclk2:[R]
Latch Clk	sysclk2:[R]

Data Arrival Path:

AT	DELAY	TYPE	RF	FANOUT	LOC	NODE
0.000	0.000					active clock edge time
0.000	0.000					sysclk2
0.000	0.000	tCL	RR	1	IOL5[A]	clk2_ibuf/I
0.943	0.943	tINS	RR	2	IOL5[A]	clk2_ibuf/O
3.236	2.293	tNET	RR	1	R5C9[0][B]	reg21_Z/CLK
3.786	0.550	tC2Q	RR	1	R5C9[0][B]	**reg21_Z/Q**
4.189	0.403	tNET	RR	1	R5C9[0][A]	reg21_i_cZ/I0
4.940	0.751	tINS	RF	1	R5C9[0][A]	reg21_i_cZ/F
4.940	0.000	tNET	FF	1	R5C9[0][A]	**reg22_Z/D**

Data Required Path:

AT	DELAY	TYPE	RF	FANOUT	LOC	NODE
4.000	4.000					active clock edge time
4.000	0.000					sysclk2
4.000	0.000	tCL	RR	1	IOL5[A]	clk2_ibuf/I
4.943	0.943	tINS	RR	2	IOL5[A]	clk2_ibuf/O
7.236	2.293	tNET	RR	1	R5C9[0][A]	reg22_Z/CLK

时序例外报告默认报告所有的受时序例外约束语句影响的路径，云源提供 report_exception 约束命令，允许用户配置和显示关心的部分报告内容，将不关心的报告路径进行过滤。如图 4-19 所示是在图 4-17 基础上再添加 report_exception 语句，红框里第一行表示受 sysclk1 影响的路径报告一条 setup 分析，第二行表示受 sysclk2 影响的路径不进行 setup 分析报告。

图 4-19 report_exception 语句

```
create_clock -name sysclk1 -period 10 -waveform {0 5} [get_ports {clk1}]
create_clock -name sysclk2 -period 10 -waveform {0 5} [get_ports {clk2}]
set_max_delay -from [get_clocks {sysclk1}] -to [get_clocks {sysclk1}] 5
set_max_delay -from [get_clocks {sysclk2}] -to [get_clocks {sysclk2}] 4
report_exceptions -setup -from_clock [get_clocks {sysclk1}] -to_clock [get_clocks {sysclk1}] -max_paths 1 -max_common_paths 1
report_exceptions -setup -from_clock [get_clocks {sysclk2}] -to_clock [get_clocks {sysclk2}] -max_paths 0 -max_common_paths 0
```

图 4-19 约束后的时序例外报告如图 4-20 所示。

图 4-20 report_exception 报告

Timing Exceptions Report:

Setup Analysis Report

Setup Analysis Report[1]:

Report Command:report_exceptions -setup -from_clock [get_clocks {sysclk1}] -to_clock [get_clocks {sysclk1}] -max_paths 1 -max_common_paths 1

Timing Path Constraint[1]: set_max_delay -from [get_clocks {sysclk1}] -to [get_clocks {sysclk1}] 5

Path1

Path Summary:

Slack	-0.654
Data Arrival Time	7.947
Data Required Time	7.293
From	reg11_ins23
To	reg12_ins20
Launch Clk	sysclk1:[R]
Latch Clk	sysclk1:[R]

Data Arrival Path:

AT	DELAY	TYPE	RF	FANOUT	LOC	NODE
0.000	0.000					active clock edge time
0.000	0.000					sysclk1
0.000	0.000	tCL	RR	1	IOL15[A]	clk1_ibuf13/I
0.982	0.982	tINS	RR	2	IOL15[A]	clk1_ibuf13/O
2.893	1.911	tNET	RR	1	IOL2[B]	reg11_ins23/CLK
3.351	0.458	tC2Q	RF	1	IOL2[B]	**reg11_ins23/Q**
7.947	4.596	tNET	FF	1	R15C23[1][A]	**reg12_ins20/D**

Data Required Path:

AT	DELAY	TYPE	RF	FANOUT	LOC	NODE
5.000	5.000					active clock edge time
5.000	0.000					sysclk1
5.000	0.000	tCL	RR	1	IOL15[A]	clk1_ibuf13/I
5.982	0.982	tINS	RR	2	IOL15[A]	clk1_ibuf13/O
7.893	1.911	tNET	RR	1	R15C23[1][A]	reg12_ins20/CLK
7.693	-0.200	tUnc				reg12_ins20
7.293	-0.400	tSu		1	R15C23[1][A]	reg12_ins20

Path Statistics:

Clock Skew	0.000
Setup Relationship	5.000
Logic Level	1
Arrival Clock Path Delay	cell: 0.982, 33.942%; route: 1.911, 66.058%
Arrival Data Path Delay	cell: 0.000, 0.000%; route: 4.596, 90.932%; tC2Q: 0.458, 9.068%
Required Clock Path Delay	cell: 0.982, 33.942%; route: 1.911, 66.058%

4.2.8 Timing Constraints Report

图 4-21 为时序约束报告，图中信息说明如下：

- SDC Command Type：静态时序约束命令的类型；
- State：包含 Invalid、Actived 两个值，Actived 表示命令生效，Invalid 表示命令无效；

- Detail Command：其值等于 SDC 文件中对应的时序约束语句。

图 4-21 时序约束报告

Timing Constraints Report:

SDC Command Type	State	Detail Command
TC_CLOCK	Actived	create_clock -name main -period 18.182 -waveform {0 9.091} [get_ports {clk}]
TC_GENERATED_CLOCK	Actived	create_generated_clock -name main_gen -source [get_ports {clk}] -master_clock main -divide_by 5 -duty_cycle 40 -phase 22 -offset 50 [get_ports {in}]
TC_INPUT_DELAY	Actived	set_input_delay -clock main_gen 0.2 -clock_fall -add_delay -source_latency_included [get_ports {in}]
TC_CLOCK_LATENCY	Actived	set_clock_latency -source 1.2 [get_clocks {main}]
TC_CLOCK_UNCERTAINTY	Actived	set_clock_uncertainty 2.3 -setup -from [get_clocks {main}] -to [get_clocks {main}]
TC_FALSE_PATH	Actived	set_false_path -from [get_clocks {main_gen}] -to [get_clocks {main_gen}]
TC_MULTICYCLE	Actived	set_multicycle_path -from [get_clocks {main_gen}] -to [get_clocks {main_gen}] -setup -end 3
TC_MAX_DELAY	Actived	set_max_delay -from [get_clocks {main}] -to [get_clocks {main}] 1.11
TC_CLOCK_GROUP	Actived	set_clock_groups -exclusive -group [get_clocks {main}] -group [get_clocks {main_gen}]

5 语法规范

支持通配符“?”和“*”的使用，“?”匹配一个字符，“*”可匹配零个或多个字符且支持跨 hierarchy 层级匹配，同时支持时序约束分多行。

5.1 对象集合

表 5-1 集合说明

集合	说明
get_regs	获取 FF 对象。
get_pins	获取管脚对象。
get_ports	获取端口对象
get_clocks	获取时钟对象。
get_nets	获取 net 对象。
all_clocks	获取所有时钟对象。
all_inputs	获取所有输入端口。
all_outputs	获取所有输出端口。

5.2 时钟约束

5.2.1 create_clock

创建时钟。

语法

create_clock

-name <*clock_name*>

-period <*period_value*>

[-waveform <*edge_list*>]

<*objects*>

[-add]

参数

表 5-2 create_clock 参数

-name <clock_name>	时钟名称，此名称是时钟的唯一标识。 若创建重名时钟，后创建的时钟将覆盖先前时钟。
-period <period_value>	时钟周期，单位 ns，参数取值为正实数。
-waveform <edge_list>	以 ns 为单位指定时钟波形在整个时钟周期内的上升沿和下降沿时刻。两个时间是递增的非负数，且二者之差小于一个时钟周期。 若省略此选项，则选用默认波形，其上升沿为 0 ns，下降沿为 <*period_value*>/2 ns。 示例：Example 5.1
<objects>	创建对象，支持 get_ports、get_nets、get_regs、get_pins、all_inputs 及 all_outputs。
-add	当需在同一个<*objects*>上创建多个时钟时，应在第二条及以后的 create_clock 语句中使用-add 参数选项。若省略此选项，则在同一个<*objects*>上后创建的时钟将被忽略。 示例：Example 5.2

示例

Example 5.1

//使用通配符"?"为端口 clk 创建一个名为 user_clk 的时钟，周期 10ns，上升沿为 0 ns，下降沿为 5 ns

```
create_clock -name user_clk -period 10 -waveform {0 5} [get_ports {c?k}]
```

Example 5.2

//在同一个端口 clk 上创建两个时钟，第二个需要加-add

```
create_clock -name user_clk0 -period 10 -waveform {0 5} [get_ports {clk}]

create_clock -name user_clk1 -period 20 -waveform {0 10} [get_ports {clk}] -add
```

5.2.2 create_generated_clock

创建衍生时钟。

语法

create_generated_clock

-name <*clock_name*>

-source <*clock_source*>

[-master_clock <*clock*>]

[-edges <*edge_list*>]

[-edge_shift <*shift_list*>]

[-divide_by <*factor*>]

[-multiply_by <*factor*>]

[-duty_cycle <*percent*>]

[-phase <*degrees*>]

[-offset <*time*>]

[-invert]

[-add]

<*objects*>

参数

表 5-3 create_generated_clock 参数

-name <clock_name>	衍生时钟名，此名称是衍生时钟的唯一标识。 若创建的衍生时钟重名，后创建的衍生时钟将覆盖先前衍生时钟。 若衍生时钟名称与主时钟名称相同，则衍生时钟的创建将被忽略。
-source <clock_source>	衍生时钟的源，支持 get_ports、get_nets、get_regs、get_pins。 若源上有多个时钟，需使用-master_clock 选项指定具体的时钟。
-master_clock <clock>	衍生时钟的主时钟。 示例：Example 5.3
-edges <edge_list>	由3个递增正整数组成的列表依次确定衍生时钟第一个上升沿、第一个下降沿以及第二个上升沿。该列表的整数代表从源时钟中提取的边沿，按照“上升沿、下降沿”交替的规则，如 1 表

	示第一个上升沿，2 表示下一个下降沿，3 表示再下一个上升沿。 -edges 和-divide_by、-multiply_by、-duty_cycle、-phase、-offset 不可组合使用。 示例：Example 5.4
-edge_shift <shift_list>	由 3 个实数组成的列表分别表示衍生时钟 3 个边沿的偏移量，单位为 ns。此选项和-edges 选项一起使用，偏移量不能使某边沿超出它的相邻边沿。 -edge_shift 和-divide_by、-multiply_by、-duty_cycle、-phase、-offset 不可组合使用。 示例：Example 5.4
-divide_by <factor>	分频数，正整数，默认为 1。 示例：Example 5.3、Example 5.6、Example 5.8
-multiply_by <factor>	倍频数，正整数，默认为 1。 示例：Example 5.5、Example 5.8
-duty_cycle <percent>	占空比，参数取值为(0，100)，默认为 50%。 示例：Example 5.5
-phase <degrees>	相位，单位度，默认为 0 度。对于正相位，衍生时钟相较于主时钟右移；对于负相位，衍生时钟相较于主时钟左移。 示例：Example 5.8
-offset <time>	偏移量，单位 ns，默认为 0 ns。对于正实数，衍生时钟相较于主时钟右移指定 ns 的偏移量；对于负实数，衍生时钟相较于主时钟左移指定 ns 的偏移量。 示例：Example 5.7
-invert	反相。 示例：Example 5.7、Example 5.8
-add	在同一个 object 上创建两个及以上不同时钟时，需要指定-add 选项，否则后创建的时钟会被忽略。 示例：Example 5.8
<objects>	对象，支持 get_ports、get_nets、get_regs、get_pins、all_inputs 及 all_outputs。

示例

Example 5.3

//为端口 clk0 创建两个主时钟 user_clk0、user_clk1，并对端口 clk1 创建一个 2 分频的衍生时钟 user_clk2，其衍生主时钟选择 user_clk1

create_clock -name user_clk0 -period 10 [get_ports {clk0}]

create_clock -name user_clk1 -period 20 [get_ports {clk0}] -add

create_generated_clock -name user_clk2 -source [get_ports {clk0}] -master_clock user_clk1 -divide_by 2 [get_ports {clk1}]

Example 5.4

//使用-edges 选项创建一个与 2 分频等价的衍生时钟，各边沿偏移 0.2 ns

create_clock -name user_clk0 -period 10 [get_ports {clk0}]

create_generated_clock -name user_clk1 -source [get_ports {clk0}] -edges {1 3 5} -edge_shift {0.2 0.2 0.2} [get_ports {clk1}]

Example 5.5

//创建一个 3 倍频、占空比为 40%的衍生时钟

create_clock -name user_clk0 -period 10 [get_ports {clk0}]

create_generated_clock -name user_clk1 -source [get_ports {clk0}] -multiply_by 3 -duty_cycle 40 [get_ports {clk}]

Example 5.6

//创建一个 2 分频、相位偏移 90 度的衍生时钟

create_clock -name user_clk0 -period 10 [get_ports {clk0}]

create_generated_clock -name user_clk1 -source [get_ports {clk0}] -divide_by 2 -phase 90 [get_ports {clk1}]

Example 5.7

//创建一个相较于主时钟反向、向右偏移 4 ns 的衍生时钟

create_clock -name user_clk0 -period 10 [get_ports {clk0}]

create_generated_clock -name user_clk1 -source [get_ports {clk0}] -offset 4 -invert [get_ports {clk1}]

Example 5.8

//在端口 clk0 上创建第二个 2 分频、3 倍频、输出反向的衍生时钟，需要加-add

create_clock -name user_clk0 -period 10 [get_ports {clk0}]

create_generated_clock -name user_clk1 -source [get_ports {clk0}] -divide_by 2 -phase 90 [get_ports {clk1}]

create_generated_clock -name user_clk2 -source [get_ports {clk0}] -divide_by 2 -multiply_by 3 -invert -add [get_ports {clk1}]

5.2.3 set_clock_latency

设置时钟源延迟时间。

语法

set_clock_latency

-source

[-rise | -fall]

[-late | -early]

<*delay*>

<*objects*>

[-clock <*clock_list*>]

参数

表 5-4 set_clock_latency 参数

-source	指定分析类型为时钟源延迟时间。
-rise | -fall	时钟上升沿或下降沿延迟时间。 若都省略，则默认同时对-rise 和-fall 生效。 若仅设置-rise，则-fall 采用相同值，-fall 同理。 示例：Example 5.9
-late | -early	时钟最晚或最早延迟时间。 对于 setup 和 recovery 分析，对数据到达路径使用最晚时钟延迟，对数据所需路径使用最早时钟延迟。 对于 hold 和 removal 分析，对数据到达路径使用最早时钟延迟，对数据所需路径使用最晚时钟延迟。 若仅设置-late，则-early 采用相同值，-early 同理。 若-early 值大于-late 值，则-early 采用-late 的值。 若都省略，则默认同时对-late 和-early 生效。 示例：Example 5.10
<delay>	实数，单位 ns。
<objects>	对象，支持 get_clocks、get_ports、get_nets、get_regs、get_pins 及 all_clocks。
-clock <clock_list>	时钟。若未指定-clock 选项，则默认对<objects>的所有时钟有效。 示例：Example 5.11

示例

Example 5.9

// user_clk0、user_clk1 分别为发送、采样时钟。设置 user_clk0 延迟时间 0.22 ns，user_clk1 延迟时间 0.33ns。0.22ns 在 Data Arrival Path 中查看，0.33ns 在 Data Required Path 中查看

create_clock -name user_clk0 -period 10 [get_ports {clk0}]

create_clock -name user_clk1 -period 10 [get_ports {clk1}]

set_clock_latency -source 0.22 -rise [get_clocks { user_clk0}]

set_clock_latency -source 0.33 -fall [get_clocks { user_clk1}]

Example 5.10

//为时钟 user_clk 设置最晚延迟时间 0.33 ns，设置最早延迟时间 0.11 ns

create_clock -name user_clk -period 10 [get_ports {clk}]

set_clock_latency -source -late 0.33 [get_clocks {user_clk}]

set_clock_latency -source -early 0.11 [get_clocks {user_clk}]

Example 5.11

//为端口 clk 的 user_clk1 设置延迟时间 0.5 ns，对上升沿、下降沿、最晚、最早同时生效

create_clock -name user_clk0 -period 10 [get_ports {clk}]

create_clock -name user_clk1 -period 20 [get_ports {clk}] -add

set_clock_latency -source 0.5 [get_ports {clk}] -clock [get_clocks {user_clk1}]

5.2.4 set_clock_uncertainty

设置时钟不确定值。

语法

set_clock_uncertainty

<*uncertainty*>

-setup | -hold

[-from <*from_clock*> | -rise_from <*rise_from_clock*> | -fall_from <*fall_from_clock*>]

[-to <*to_clock*> | -rise_to <*rise_to_clock*> | -fall_to <*fall_to_clock*>]

参数

表 5-5 set_clock_uncertainty 参数

<uncertainty>	实数，单位 ns。
-setup | -hold	-setup 指定对 setup 和 recovery 进行不确定性分析。 -hold 指定对 hold 和 removal 进行不确定性分析。 若都省略，则对 setup、hold、recovery、removal 都生效。 示例：Example 5.12、Example 5.13
-from *<from_clock>*	对发送时钟进行不确定性分析，支持 get_clocks。 示例：Example 5.12
-rise_from *<rise_from_clock>*	对发送时钟的上升沿进行不确定性分析，支持 get_clocks。 示例：Example 5.13
-fall_from *<fall_from_clock>*	对发送时钟的下降沿进行不确定性分析，支持 get_clocks。 示例：Example 5.14
-to *<from_clock>*	对采样时钟进行不确定性分析，支持 get_clocks。 对同一时序路径，若-to 与-from、-rise_from、-fall_from 指定的延时值不同则按照-to 指定的值计算。 示例：Example 5.15
-rise_to *<rise_to_clock>*	对采样时钟的上升沿进行不确定性分析，支持 get_clocks。 对同一时序路径，若-rise_to 与-from、-rise_from、-fall_from 指定的延时值不同则按照-to 指定的值计算。 示例：Example 5.13
-fall_to *<fall_to_clock>*	对采样时钟的下降沿进行不确定性分析，支持 get_clocks。 对同一时序路径，若-fall_to 与-from、-rise_from、-fall_from 指定的延时值不同则按照-to 指定的值计算。 -from、-rise_from、-fall_from、-to、-rise_to、-fall_to 选项必须至少指定一个。 示例：Example 5.14

示例

Example 5.12

//针对 setup 和 recovery 分析，为发送时钟 user_clk 设置 0.2 ns 的不确定性值

```
create_clock -name user_clk -period 10 [get_ports {clk}]
set_clock_uncertainty 0.2 -setup -from [get_clocks {user_clk}]
```

Example 5.13

//针对 hold 和 removal 分析，为发送时钟 user_clk0 的上升沿设置 0.111 ns 的不确定性值，为采样时钟 user_clk1 的上升沿设置 0.222 ns 的不确定值

```
create_clock -name user_clk0 -period 10 [get_ports {clk0}]
create_clock -name user_clk1 -period 10 [get_ports {clk1}]
set_clock_uncertainty 0.111 -hold -rise_from [get_clocks {user_clk0}]
set_clock_uncertainty 0.222 -hold -rise_to [get_clocks {user_clk1}]
```

Example 5.14

//针对 hold 和 removal 分析，为发送时钟 user_clk0 的下降沿设置 0.111 ns 的不确定性值，为采样时钟 user_clk1 的下降沿设置 0.222 ns 的不确定值

```
create_clock -name user_clk0 -period 10 [get_ports {clk0}]
create_clock -name user_clk1 -period 20 [get_ports {clk1}]
set_clock_uncertainty -hold 0.111 -fall_from [get_clocks {user_clk0}]
set_clock_uncertainty -hold 0.222 -fall_to [get_clocks {user_clk1}]
```

Example 5.15

//针对 setup 和 recovery 分析，同一条时序路径，为发送时钟 user_clk0，采样时钟 user_clk1 分别设置 0.2 ns、0.3 ns 的不确定值，云源按照 0.3 ns 进行计算

```
create_clock -name user_clk0 -period 10 [get_ports {clk0}]
create_clock -name user_clk1 -period 10 [get_ports {clk1}]
set_clock_uncertainty -setup 0.2 -from [get_clocks {user_clk0}]
set_clock_uncertainty -setup 0.3 -to [get_clocks {user_clk1}]
```

5.2.5 set_clock_groups

设置时钟组。

语法

set_clock_groups

-asynchronous | -exclusive

[-group <*clocks*>]

[-group <*clocks*>]

…

参数

表 5-6 set_clock_groups 参数

-asynchronous | -exclusive	-asynchronous 或-exclusive 指定时钟组间关系为异步或互斥。 示例：Example 5.16、Example 5.17
-group <*clocks*>	时钟组，使用 get_clocks 获取。

示例

Example 5.16

//时钟 clk1、ck2、clk3 相互之间有时序路径，仅为 clk1 设置时钟组，组外异步

create_clock -name clk1 -period 10 -waveform {0 5} [get_ports {clk0}]

create_clock -name clk2 -period 10 -waveform {0 5} [get_ports {clk1}]

create_clock -name clk3 -period 10 -waveform {0 5} [get_ports {clk2}]

set_clock_groups -asynchronous -group [get_clocks {clk1}]

分析结果如下表所示：

表 5-7 分析结果

发送时钟 / 采样时钟	clk1	clk2	clk3
clk1	分析	不分析	不分析

发送时钟 / 采样时钟	clk1	clk2	clk3
clk2	不分析	分析	分析
clk3	不分析	分析	分析

Example 5.17

//时钟 clk1、ck2、clk3 相互之间有时序路径，设置 clk1、clk2 为同时钟组，组外互斥

create_clock -name clk1 -period 10 -waveform {0 5} [get_ports {clk0}]

create_clock -name clk2 -period 10 -waveform {0 5} [get_ports {clk1}]

create_clock -name clk3 -period 10 -waveform {0 5} [get_ports {clk2}]

set_clock_groups -exclusive -group [get_clocks {clk1 clk2}]

分析结果如下表所示：

表 5-8 分析结果

发送时钟 / 采样时钟	clk1	clk2	clk3
clk1	分析	分析	不分析
clk2	分析	分析	不分析
clk3	不分析	不分析	分析

5.3 I/O 约束

设置数据输入、输出的延迟时间值。

语法

set_input_delay / set_output_delay

-clock <*name*>

<*delay_value*>

[-clock_fall]

[-max | -min]

[-rise | -fall]

[-add_delay]

[-source_latency_included]

<objects>

参数

表 5-9 I/O 约束参数

-clock *<name>*	指定时钟，且该时钟必须为已创建的时钟。
<delay_value>	实数，单位ns。
-clock_fall	指定与时钟下降沿相关。 若省略指定，则输入、输出延迟时间默认相对于关联时钟的上升沿相关。 示例：Example 5.18、Example 5.23
-max | -min	-max指定最大输入、输出延迟时间，对setup和recovery分析有效。 -min指定最小输入、输出延迟时间，对hold和removal分析有效。 若只设置-max，则-min为相同值，-min同理。 若都省略，则最大和最小输入、输出延迟时间一致。 示例：Example 5.19、Example 5.24
-rise | -fall	-rise或-fall为输入、输出端口指定上升沿或下降沿的延迟时间。 若只设置-rise，则-fall为相同值，-fall同理。 若都省略，则上升沿和下降沿的延迟时间一致。 示例：Example 5.20、Example 5.24
-add_delay	使得多个此类约束同时生效。 示例：Example 5.21、Example 5.26
-source_latency_included	指定输入、输出延迟时间已包含时钟源延迟时间。 若省略指定此选项，则输入、输出延迟时间默认不包含关联时钟源延迟时间。 示例：Example 5.22、Example 5.27
<objects>	输入、输出对象，支持get_ports、all_inputs/all_outputs。

注！

set_input_delay 不支持时钟输入 port。

示例

Example 5.18

//为输入端口 d 设置延迟时间值 0.2 ns，相对于关联时钟 user_clk 的下降沿

```
create_clock -name user_clk -period 10 [get_ports {clk}]
set_input_delay -clock user_clk 0.2 -clock_fall [get_ports {d}]
```

Example 5.19

//为输入端口 d 设置最大延迟时间值 0.2 ns、最小延迟时间值 0.1 ns，相对于关联时钟 user_clk

```
create_clock -name user_clk -period 10 [get_ports {clk}]
set_input_delay -clock user_clk 0.2 -max [get_ports {d}]
set_input_delay -clock user_clk 0.1 -min [get_ports {d}]
```

Example 5.20

//为输入端口 d 设置上升沿延迟时间值 0.2 ns、下降沿延迟时间值 0.1 ns，相对于关联时钟 user_clk

```
create_clock -name user_clk -period 10 [get_ports {clk}]
set_input_delay -clock user_clk 0.2 -rise [get_ports {d}]
set_input_delay -clock user_clk 0.1 -fall [get_ports {d}]
```

Example 5.21

//为输入端口 d 同时设置延迟时间 0.2 ns 和 0.4 ns，相对于关联时钟 user_clk，setup 和 recovery 分析 tIn 取值 0.4ns，hold 和 removal 分析，tIn 取值 0.2ns

```
create_clock -name user_clk -period 10 [get_ports {clk}]
set_input_delay -clock user_clk 0.2 [get_ports {d}]
set_input_delay -clock user_clk 0.4 -add_delay [get_ports {d}]
```

Example 5.22

//为输入端口 d 设置延迟时间值 0.2 ns，包含关联时钟 user_clk 的时钟延迟时间 0.15 ns

```
create_clock -name user_clk -period 10 [get_ports {clk}]
set_clock_latency -source 0.15 [get_ports {clk}] -clock [get_clocks {user_clk}]
set_input_delay -clock user_clk 0.2 -source_latency_included [get_ports {d}]
```

Example 5.23

//为输出端口 q 设置延迟时间值 0.2 ns，相对于关联时钟 user_clk 的下降沿

```
create_clock -name user_clk -period 10 [get_ports {clk}]
```

set_output_delay -clock user_clk 0.2 -clock_fall [get_ports {q}]

Example 5.24

//为输出端口 q 设置最大延迟时间值 0.2 ns、最小延迟时间值 0.1 ns，相对于关联时钟 user_clk

create_clock -name user_clk -period 10 [get_ports {clk}]

set_output_delay -clock user_clk 0.2 -max [get_ports {q}]

set_output_delay -clock user_clk 0.1 -min [get_ports {q}]

Example 5.25

//为输出端口 q 设置上升沿延迟时间值 0.2 ns、下降沿延迟时间值 0.1 ns，相对于关联时钟 user_clk

create_clock -name user_clk -period 10 [get_ports {clk}]

set_output_delay -clock user_clk 0.2 -rise [get_ports {q}]

set_output_delay -clock user_clk 0.1 -fall [get_ports {q}]

Example 5.26

//为输出端口 q 同时设置延迟时间 0.2 ns 和 0.4 ns，相对于关联时钟 user_clk，setup 和 recovery 分析 tOut 为-0.2ns，hold 和 removal 分析 tOut 为-0.4ns

create_clock -name user_clk -period 10 [get_ports {clk}]

set_output_delay -clock user_clk 0.2 [get_ports {q}]

set_output_delay -clock user_clk 0.4 -add_delay [get_ports {q}]

Example 5.27

//为输出端口 q 设置延迟时间值 0.2 ns，包含关联时钟 user_clk 的时钟延迟时间 0.15 ns

create_clock -name user_clk -period 10 [get_ports {clk}]

set_clock_latency -source 0.15 [get_ports {clk}] -clock [get_clocks {user_clk}]

set_output_delay -clock user_clk 0.2 -source_latency_included [get_ports {q}

5.4 例外约束

5.4.1 set_false_path

设置伪路径。对设置的时序路径不再进行时序分析。

语法

set_false_path

[-from <*from_list*>]

[-through <*through_list*>]

[-to <*to_list*>]

[-setup | -hold]

参数

表 5-10 set_false_path 参数

-from <*from_list*>	起点，支持get_clocks、get_ports、get_regs、get_pins、all_clocks、all_inputs以及all_outputs。 示例：Example 5.28、Example 5.29
-through <*through_list*>	经过点，支持get_nets和get_pins。 示例：Example 5.30
-to <*to_list*>	终点，支持 get_clocks、get_ports、get_regs、get_pins、all_clocks、all_inputs以及all_outputs。 -from、-through和-to选项必须至少指定一个。 示例：Example 5.28
-setup | -hold	-setup对setup和recovery分析有效。 -hold对hold和removal分析有效。 若省略此选项，则默认对setup、hold、recovery、removal分析都有效。 示例：Example 5.29、Example 5.30

示例

Example 5.28

//对发送时钟为 user_clk0、采样时钟为 user_clk1 的所有时序路径设置为伪路径，对 setup、hold、recovery、removal 分析都有效

```
create_clock -name user_clk0 -period 10 [get_ports {clk0}]
create_clock -name user_clk1 -period 5 [get_ports {clk1}]
```

set_false_path -from [get_clocks {user_clk0}] -to [get_clocks {user_clk1}]

Example 5.29

//对从端口 d 出发的所有时序路径设置为伪路径，只对 setup 和 recovery 分析有效

set_false_path -from [get_ports {d}] -setup

Example 5.30

//对经过 FF reg0 输入 D 的所有时序路径设置为伪路径，只对 hold 和 removal 分析有效

set_false_path -through [get_pins {reg0_s0/D}] -hold

5.4.2 set_max_delay/set_min_delay

设置最大/最小延迟时间值。

语法

set_max_delay / set_min_delay

[-from *<from_list>*]

[-through *<through_list>*]

[-to *<to_list>*]

<delay_value>

参数

表 5-11 set_max_delay/set_min_delay 参数

-from <from_list>	起点，支持 get_clocks、get_ports、get_regs、get_pins、all_clocks、all_inputs 以及 all_outputs。 示例：Example 5.31
-through <through_list>	经过点，支持 get_nets 和 get_pins。 示例：Example 5.32
-to <to_list>	终点，支持 get_clocks、get_ports、get_regs、get_pins、all_clocks、all_inputs 以及 all_outputs。 -from、-through 和-to 选项必须至少指定一个。 示例：Example 5.31、Example 5.33
<delay_value>	实数，单位 ns。 设置最大延迟时间值对 setup 和 recovery 分析有效。 设置最小延迟时间值对 hold 和 removal 分析有效。

示例

Example 5.31

//对发送时钟为 user_clk0、采样时钟为 user_clk1 的所有时序路径设置最大延迟时间 5ns、最小延迟时间 2 ns，5ns 对 setup 和 recovery 分析有效，2ns 对 hold 和 removal 分析有效

```
create_clock -name user_clk0 -period 10 [get_ports {clk0}]
create_clock -name user_clk1 -period 5 [get_ports {clk1}]
set_max_delay -from [get_clocks {user_clk0}] 5 -to [get_clocks {user_clk1}]
set_min_delay -from [get_clocks {user_clk0}] 2 -to [get_clocks {user_clk1}]
```

Example 5.32

//对经过 FF reg0 输入 D 的所有时序路径设置最小延迟时间 0.2 ns

```
set_min_delay -through [get_pins {reg0_s0/D}] 0.2
```

Example 5.33

//对以 FF reg0 为终点的所有时序路径设置最大延迟时间 6 ns、最小延迟时间 0.5 ns

```
set_max_delay -to [get_regs {reg0_s0}] 6
set_min_delay -to [get_regs {reg0_s0}] 0.5
```

5.4.3 set_multicycle_path

设置多循环周期。

语法

```
set_multicycle_path
    [-from <from_list>]
    [-through <through_list>]
    [-to <to_list>]
    -setup | -hold
    -start | -end
    <multiple_value>
```

参数

表 5-12 set_multicycle_path 参数

-from <from_list>	起点，支持 get_clocks、get_ports、get_regs、get_pins、all_clocks、all_inputs 以及 all_outputs。 示例：Example 5.34
-through <through_list>	经过点，支持 get_nets 和 get_pins。 示例：Example 5.35
-to <to_list>	终点，支持 get_clocks、get_ports、get_regs、get_pins、all_clocks、all_inputs 以及 all_outputs。 -from、-through 和-to 选项必须至少指定一个。 示例：Example 5.36
-setup | -hold	-setup 对 setup 和 recovery 分析进行多周期约束。 -hold 对 hold 和 removal 分析进行多周期约束。 示例：Example 5.34、Example 5.36
-start | -end	-start 或-end 指定相对于发送时钟或采样时钟进行多周期设置。 对于 setup 分析，指定-start 使发送时钟向左移动多周期，指定-end 使采样时钟向右移动多周期。 对于 hold 分析，指定-start 使发送时钟向右移动多周期，指定-end 使采样时钟向左移动多周期。 示例：Example 5.34、Example 5.36
<multiple_value>	整数。

示例

Example 5.34

//对于发送时钟为 user_clk 的所有时序路径，相对于采样时钟对 setup 和 recovery 分析进行 3 周期分析，相对于发送时钟对 hold 和 removal 分析进行 2 周期分析

```
create_clock -name user_clk -period 10 [get_ports {clk}]
set_multicycle_path -from [get_clocks {user_clk}] -setup -end 3
set_multicycle_path -from [get_clocks {user_clk}] -hold -start 2
```

Example 5.35

//对经过 FF reg0 输出 Q 的所有时序路径，相对于采样时钟对 setup 和 recovery 分析进行 5 周期分析

```
create_clock -name user_clk -period 10 [get_ports {clk}]
```

set_multicycle_path -through [get_pins {reg0_s0/Q}] 5 -setup

Example 5.36

//时序路径中发送时钟为 user_clk0，采样时钟为 user_clk1，分别设置 2、3，对 setup 和 recovery 分析有效，云源按照 2 设置多循环周期

create_clock -name user_clk0 -period 10 [get_ports {clk0}]

create_clock -name user_clk1 -period 20 [get_ports {clk1}]

set_multicycle_path -from [get_clocks { user_clk0}] -setup -end 2

set_multicycle_path -to [get_clocks { user_clk1}] -setup -end 3

5.5 报告约束

5.5.1 report_timing

报告时序。

语法

report_timing

-setup | -hold | -recovery | -removal

[-from_clock <*clock_name*> | -rise_from_clock <*clock_name*> | -fall_from_clock <*clock_name*>]

[-to_clock <*clock_name*> | -rise_to <*name*> | -fall_to <*clock_name*>]

[-from <*from_list*> | -rise_from <*rise_from_list*> | -fall_from <*fall_from_list*>]

[-through <*through_list*>]

[-to <*to_list*> | -rise_to <*rise_to_list*> | -fall_to <*fall_to_list*>]

[-max_paths <*value*>]

[-max_common_paths <*value*>]

[-max_logic_level <*value*>]

[-min_logic_level <*value*>]

[-mod_ins {mod_ins1 mod_ins2 ...}]

参数

表 5-13 report_timing 参数

-setup | -hold | -recovery | -removal	-setup 或-hold 或-recovery 或-removal 分别对 setup 或 hold 或 recovery 或 removal 分析有效。 示例：Example 5.37、Example 5.38、Example 5.39、Example 5.40
-from_clock <clock_name>	发送时钟，支持 get_clocks。 示例：Example 5.37
-rise_from_clock <clock_name>	发送时钟且上升沿有效，支持 get_clocks。
-fall_from_clock <clock_name>	发送时钟且下降沿有效，支持 get_clocks。
-to_clock <clock_name>	采样时钟，支持 get_clocks。 示例： Example 5.37
-rise_to_clock <clock_name>	采样时钟且上升沿有效，支持 get_clocks。
-fall_to_clock <clock_name>	采样时钟且下降沿有效，支持 get_clocks。
-from <from_list>	起点，支持 get_clocks、get_ports、get_regs 以及 get_pins。 Example 5.38
-rise_from <rise_from_list>	对象的发送时钟且上升沿有效，支持 get_clocks。
-fall_from <fall_from_list>	对象的发送时钟且下降沿有效，支持 get_clocks。
-through <through_list>	经过点，支持 get_nets 和 get_pins。 Example 5.39
-to <to_list>	终点，支持 get_clocks、get_ports、get_regs 以及 get_pins。 Example 5.38
-rise_to <rise_to_list>	对象的采样时钟且为上升沿有效，支持 get_clocks 和 all_clocks。
-fall_to <fall_to_list>	对象的采样时钟且为下降沿有效，支持 get_clocks 和 all_clocks。

-max_paths <value>	打印时序路径的最大条数，正整数。 若省略此选项，则默认为 25。 Example 5.38
-max_common_paths <value>	打印最大共同路径数，正整数，默认为 1。 需满足相同时钟和相同对象才视为同一终点。 Example 5.38
-max_logic_level <value>	打印逻辑级数小于设定值的时序路径，正整数。 Example 5.37
-min_logic_level <value>	打印逻辑级数大于设定值的时序路径，正整数。 Example 5.39
-mod_ins {mod_ins1 mod_ins2 …}	打印设计中实例化 module 相关的时序路径。 Example 5.40

示例

Example 5.37

//对于 setup 部分，约束打印发送时钟和采样时钟为 user_clk 且逻辑级数小于 4 的时序路径

```
create_clock -name user_clk -period 10 [get_ports {clk}]
report_timing -setup -from_clock [get_clocks {user_clk}] -to_clock [get_clocks {user_clk}] -max_logic_level 4
```

Example 5.38

//对于 hold 部分，约束打印 FF reg0 和 reg1 之间的时序路径，且最大打印 50 条路径，最大打印 5 条相同路径

```
report_timing -hold -from [get_regs {reg0_s0}] -to [get_regs {reg1_s0}] -max_paths 50 -max_common_paths 5
```

Example 5.39

//对于 recovery 部分，约束打印经过 FF reg0 输入 D 且逻辑级数大于 3 的时序路径

```
report_timing -recovery -through [get_pins {reg0_s0/D}] -min_logic_level 3
```

Example 5.40

//对于 removal 部分，约束打印设计中实例化 module 为 sub_inst 的相关时序路径

report_timing -removal -mod_ins {sub_inst}

5.5.2 report_high_fanout_nets

报告高扇出 Nets。

语法

report_high_fanout_nets

-max_nets <*max_nets_value*>

[-min_fanout <*min_fanout_value*>]

[-max_fanout <*max_fanout_value*>]

[-clock_regions]

[-slr]

[-ascending]

参数

表 5-14 report_high_fanout_nets 参数

-max_nets <max_nets_value>	net 的最大条数，正整数，默认为 10。
-min_fanout <min_fanout_value>	打印扇出数大于设定值的 net，正整数，需小于等于-max_fanout 的参数值。 示例：Example 5.41
-max_fanout <max_fanout_value>	打印扇出数小于设定值的 net，正整数，需大于等于-min_fanout 的参数值。 示例：Example 5.42
-clock_regions	打印连接时钟端口的 net。 示例：Example 5.42
-slr	打印连接 LSR 端口的 net。LSR 指复位、置位、清零。 示例：Example 5.43
-ascending	按照 net 扇出值升序的方式进行打印。 若省略此选项，则默认按照 net 扇出值降序的方式进行打印。 示例：Example 5.43

示例

Example 5.41

//指定打印扇出数大于等于 2 的 net，且最大打印 25 条

report_high_fanout_nets -max_nets 25 -min_fanout 2

Example 5.42

//指定打印与 clk 相关的 net，且扇出数不大于 3

report_high_fanout_nets -max_fanout 3 -clock_regions

Example 5.43

//指定打印与 set 和 reset 相关的 net，且按照扇出数升序的方式打印

report_high_fanout_nets -slr -ascending

5.5.3 report_route_congestion

报告绕线拥塞度。

语法

report_route_congestion

-max_grids <*max_grids_value*>

[-min_route_congestion <*min_route_congestion_value*>]

[-max_route_congestion <*max_route_congestion_value*>]

[-LOC <*grid_position*>]

参数

表 5-15 report_route_congestion 参数

-max_grids <max_grids_value>	打印 grid 的最大条数，正整数，默认为 10。 示例：Example 5.44
-min_route_congestion <min_ route_congestion _value>	打印绕线拥塞度不小于设定值的 grid，参数取值为[0,1]的实数，需大于等于 -max_route_congestion 的参数值。 示例：Example 5.45
-max_route_congestion <max_ route_congestion _value>	打印绕线拥塞度不大于设定值的 grid，参数取值为[0,1]的实数，需小于等于 -min_route_congestion 的参数值。 示例：Example 5.45
-LOC <grid_position>	打印约束位置范围内的 grid。 示例：Example 5.44

示例

Example 5.44

//指定打印第 3 至 5 行、第 4 至 6 列范围内的 grid，最大打印 30 条

report_route_congestion -max_grids 30 -LOC R[3:5]C[4:6]

Example 5.45

//指定打印绕线拥塞度在 0.5 和 1 之间的 grid，最大打印 30 条

report_route_congestion -max_grids 30 -min_route_congestion 0.5 -max_route_congestion 1

5.5.4 report_min_pulse_width

报告最小脉冲宽度。

语法

report_min_pulse_width

-nworst <nworst_value>

[-min_pulse_width <*min_pulse_width_value*>]

[-max_pulse_width <*max_pulse_width_value*>]

[-detail]

<*objects*>

参数

表 5-16 report_min_pulse_width 参数

-nworst <nworst_value>	打印 FF 的最差 MPW 的最大条数，正整数，默认为 10。 示例：Example 5.46 、Example 5.47、Example 5.48
-min_pulse_width <min_pulse_width_value>	打印实际 MPW 不小于设定值的 FF，正实数，需小于等于 -max_pulse_width 的参数值。 示例：Example 5.46 、Example 5.47
-max_pulse_width <max_pulse_width_value>	打印实际 MPW 不大于设定值的 FF，正实数，需大于等于 -min_pulse_width 的参数值。 示例：Example 5.47
-detail	对 FF 的 MPW 进行详细打印，除摘要表格外，还包括时钟路径分析。 若省略此选项，则只打印摘要表格。 示例：Example 5.46

<objects>	指定只打印特定 FF 的 MPW，支持 get_regs。 示例：Example 5.48

示例

Example 5.46

//指定打印实际 MPW 不小于 2.5 的 FF，最大打印 30 条并打印时钟路径分析

report_min_pulse_width -nworst 30 -min_pulse_width 2.5 -detail

Example 5.47

//指定打印实际 MPW 在 2 和 4 之间的 FF，最大打印 15 条

report_min_pulse_width -nworst 15 -min_pulse_width 2 -max_pulse_width 4

Example 5.48

//指定只打印 reg0 和 reg1 的 MPW，最大打印 10 条

report_min_pulse_width -nworst 10 [get_regs {reg0_s0 reg1_s0}]

5.5.5 report_max_frequency

报告子 module 最大工作频率。

在时序报告中打印设计中子 module 相关同步时钟的最大工作频率，其最大工作频率是选取同步时序路径的终点在子 module 内的最差时序路径计算得出。

语法

report_max_frequency

-mod_ins {mod_ins0 mod_ins1 ...}

参数

表 5-17 report_max_frequency 参数

-mod_ins {mod_ins0 mod_ins1 …}	指定打印设计中子 module 相关时钟的最大工作频率。 示例：Example 5.49

示例

Example 5.49

//指定打印设计中 mod_inst0 相关同步时钟的最大工作频率

report_max_frequency -mod_ins {mod_inst0}

5.5.6 report_exceptions

报告例外约束。

语法选项请样例可参考 5.5.1 report_timing。

语法

report_exceptions

-setup | -hold | -recovery | -removal

[-from_clock <*clock_name*> | -rise_from_clock <*clock_name*> | -fall_from_clock <*name*>]

[-to_clock <*clock_name*> | -rise_to <*clock_name*> | -fall_to <*clock_name*>]

[-from <*from_list*> | -rise_from <*rise_from_list*> | -fall_from <*fall_from_list*>]

[-through <*through_list*>]

[-to <*to_list*> | -rise_to <*rise_to_list*> | -fall_to <*fall_to_list*>]

[-max_paths <*value*>]

[-max_common_paths <*value*>]

[-max_logic_level <*value*>]

[-min_logic_level <*value*>]

参数

表 5-18 report_exceptions 参数

-setup | -hold | -recovery | -removal	-setup 或-hold 或-recovery 或-removal 分别对 setup 或 hold 或 recovery 或 removal 分析有效。
-from_clock <name>	发送时钟，支持 get_clocks。
-rise_from_clock <name>	发送时钟且为上升沿有效，支持 get_clocks。

-fall_from_clock <name>	发送时钟且为下降沿有效，支持 get_clocks。
-to_clock <name>	采样时钟，支持 get_clocks。
-rise_to_clock <name>	采样时钟且上升沿有效，支持 get_clocks。
-fall_to_clock <name>	采样时钟且下降沿有效，支持 get_clocks。
-from <from_list>	起点，支持 get_clocks、get_ports、get_regs 以及 get_pins。
-rise_from <rise_from_list>	对象的发送时钟且上升沿有效，支持 get_clocks。
-fall_from <fall_from_list>	对象的发送时钟且下降沿有效，支持 get_clocks。
-through <through_list>	经过点，支持 get_nets 和 get_pins。
-to <to_list>	终点，支持 get_clocks、get_ports、get_regs 以及 get_pins。
-rise_to <rise_to_list>	对象的采样时钟且上升沿有效，支持 get_clocks 和 all_clocks。
-fall_to <fall_to_list>	对象的采样时钟且下降沿有效，支持 get_clocks 和 all_clocks。
-max_paths <value>	打印时序路径的最大条数，正整数，默认 25。
-max_common_paths <value>	打印最大共同路径数，正整数，默认为 1。 需满足相同时钟和相同对象才视为同一终点。
-max_logic_level <value>	打印逻辑级数小于设定值的时序路径，正整数。
-min_logic_level <value>	打印逻辑级数大于设定值的时序路径，正整数。

5.6 工作条件约束

设置工作条件。

语法

set_operating_conditions

-grade <*c* | *i* | *a*>

-model <*slow* | *fast*>

-speed <*speed_value*>

[-setup]

[-hold]

参数

表 5-19 工作条件约束参数

-grade <c | i | a>	指定器件的温度等级。其中，c 代表商业级(commercial)、i 代表工业级(industrial)、a 代表车规级(automotive)。 示例：Example 5.50、Example 5.51、Example 5.52
-model <slow | fast>	指定时序分析选用的延迟时间模型。 示例： Example 5.50、Example 5.51、Example 5.52
-speed <speed_value>	指定器件的速度等级，参数取值为正整数且符合器件支持的速度等级。
-setup	指定对 setup 和 recovery 分析应用设置的延迟时间模型。若省略指定此选项，则默认选用 slow 延迟时间模型。 示例：Example 5.50、Example 5.52
-hold	指定对 hold 和 removal 分析应用设置的延迟时间模型。若省略指定此选项，则默认选用 fast 延迟时间模型。 示例：Example 5.51、Example 5.52

示例

Example 5.50

//对器件设置商业级、速度等级为 7，对 setup 和 recovery 分析选用 fast 延迟时间模型

set_operating_conditions -grade c -model fast -speed 7 -setup

Example 5.51

//对器件设置工业级、速度等级为 6，对 hold 和 removal 分析选用 slow

延迟时间模型

set_operating_conditions -grade i -model slow -speed 6 -hold

Example 5.52

//对器件设置车规级、速度等级为 6，对 setup、hold、recovery、removal 分析选用 fast 延迟时间模型

set_operating_conditions -grade a -model fast -speed 6 -setup -hold

5.7 其它约束

5.7.1 derive_clocks

创建全局目标时钟频率。

语法

derive_clocks

-freq <*freq_value*>

参数

表 5-20 derive_clocks 参数

-freq <*freq_value*>	以MHz为单位指定全局目标时钟频率，参数取值为(0,1200]的实数，默认值为100。 示例：Example 5.53

示例

Example 5.53

//创建全局目标时钟频率为 90MHz

derive_clocks -freq 90


GOWIN高云
智慧逻辑 定制未来