




Gowin 设计物理约束 用户指南

SUG935-1.4.1, 2023-11-30

版权所有 © 2023 广东高云半导体科技股份有限公司

GOWIN高云、、Gowin、小蜜蜂、晨熙、云源以及高云均为广东高云半导体科技股份有限公司注册商标，本手册中提到的其他任何商标，其所有权利属其拥有者所有。未经本公司书面许可，任何单位和个人都不得擅自摘抄、复制、翻译本文档内容的部分或全部，并不得以任何形式传播。

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，或以禁止发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 |
|------------|-------|--|
| 2020/05/09 | 1.0 | 初始版本。 |
| 2020/09/04 | 1.1 | <ul style="list-style-type: none">● FloorPlanner 菜单栏优化;● 支持 Back-annotate Physical Constraints 功能。 |
| 2021/06/10 | 1.2 | 添加 Port 属性约束和 Vref Constraints 联合使用说明。 |
| 2021/11/02 | 1.3 | <ul style="list-style-type: none">● 修改文档部分描述;● 调整文档结构。 |
| 2022/10/28 | 1.3.1 | 删除 GW1NS-2 系列器件。 |
| 2022/12/16 | 1.3.2 | <ul style="list-style-type: none">● 新增 A.10 其他约束;● VCC 更新为 VCCIO。 |
| 2023/03/31 | 1.3.3 | <ul style="list-style-type: none">● 屏蔽 Slew Rate 的设置;● 删除 Find 功能。 |
| 2023/05/25 | 1.3.4 | <ul style="list-style-type: none">● 更新 BUFS 的描述;● Clock Assignment 更新为 Clock Net Constraints;● Quadrant Constraints 更新为 GCLK Primitive Constraints;● Hclk Constraints 更新为 HCLK Primitive Constraints。 |
| 2023/06/30 | 1.3.5 | 更新附录 A 物理约束语法规则中的原语组约束/参考电压约束/全局时钟分配约束描述。 |
| 2023/08/18 | 1.4 | 删除 Timing Paths 功能。 |
| 2023/11/30 | 1.4.1 | <ul style="list-style-type: none">● 更新第 3 章 FloorPlanner 界面中的部分截图;● 更新 “HCLK/GCLK Primitive Constraints” 选择 “Instance” 时弹出的对话框标题描述。 |

目录

| | |
|----------------------------------|-----------|
| 目录 | i |
| 图目录 | iii |
| 表目录 | vi |
| 1 关于本手册 | 1 |
| 1.1 手册内容 | 1 |
| 1.2 相关文档 | 1 |
| 1.3 术语、缩略语 | 1 |
| 1.4 技术支持与反馈 | 2 |
| 2 简介 | 3 |
| 3 FloorPlanner 界面 | 4 |
| 3.1 启动 | 4 |
| 3.2 界面 | 6 |
| 3.2.1 菜单栏 | 6 |
| 3.2.2 Summary 和 Netlist 窗口 | 17 |
| 3.2.3 Package View 窗口 | 20 |
| 3.2.4 Chip Array 窗口 | 25 |
| 3.2.5 Constraint 编辑窗口 | 31 |
| 3.2.6 Message 窗口 | 35 |
| 4 FloorPlanner 使用 | 36 |
| 4.1 新建约束文件 | 36 |
| 4.2 编辑约束文件 | 38 |
| 4.2.1 编辑约束示例 | 38 |
| 4.2.2 编辑 I/O 约束 | 39 |
| 4.2.3 编辑原语约束 | 41 |
| 4.2.4 编辑组约束 | 42 |
| 4.2.5 编辑资源预留约束 | 46 |
| 4.2.6 编辑全局时钟分配约束 | 47 |
| 4.2.7 编辑全局时钟原语约束 | 48 |
| 4.2.8 编辑高速时钟原语约束 | 49 |

| | |
|--------------------------------|-----------|
| 4.2.9 编辑参考电压约束 | 50 |
| 附录 A 物理约束语法规范 | 53 |
| A.1 I/O 位置约束 | 53 |
| A.2 I/O 属性约束 | 54 |
| A.3 原语位置约束 | 55 |
| A.4 组约束 | 58 |
| A.4.1 原语组约束 | 58 |
| A.4.2 相对组约束 | 60 |
| A.5 资源预留约束 | 61 |
| A.6 参考电压约束 | 61 |
| A.7 全局时钟原语约束 | 62 |
| A.8 全局时钟分配约束 | 63 |
| A.9 高速时钟原语约束 | 64 |
| A.10 其他约束 | 65 |
| A.10.1 JTAGSEL_N net 约束 | 65 |
| A.10.2 RECONFIG_N net 约束 | 65 |

图目录

| | |
|---|----|
| 图 3-1 菜单栏启动 FloorPlanner | 4 |
| 图 3-2 Process 窗口启动 | 5 |
| 图 3-3 Start Page 窗口启动 | 5 |
| 图 3-4 FloorPlanner 界面 | 6 |
| 图 3-5 File 菜单 | 7 |
| 图 3-6 Open Physical Constraints | 7 |
| 图 3-7 Constraints 菜单 | 7 |
| 图 3-8 原语查找对话框 | 8 |
| 图 3-9 新建原语组对话框 | 9 |
| 图 3-10 正确原语组对话框 | 10 |
| 图 3-11 无效位置 | 10 |
| 图 3-12 无效位置 | 10 |
| 图 3-13 创建相对位置组对话框 | 11 |
| 图 3-14 正确的相对组对话框 | 11 |
| 图 3-15 预留约束 | 12 |
| 图 3-16 时钟约束 | 12 |
| 图 3-17 创建全局时钟原语约束 (GW1N-1) | 13 |
| 图 3-18 创建全局时钟原语约束 (GW2A-18) | 13 |
| 图 3-19 创建高速时钟原语约束 | 14 |
| 图 3-20 参考电压约束 | 14 |
| 图 3-21 Tools 菜单 | 15 |
| 图 3-22 Back-annotate Physical Constraints 对话框 | 15 |
| 图 3-23 反标 Port 布局信息 | 16 |
| 图 3-24 View 菜单 | 16 |
| 图 3-25 Windows 菜单 | 17 |
| 图 3-26 Summary 窗口 | 17 |
| 图 3-27 Netlist 窗口 | 18 |
| 图 3-28 BUS 和非 BUS 结合显示 | 19 |
| 图 3-29 层级显示 | 19 |
| 图 3-30 Netlist 右键功能 | 20 |

| | |
|--|----|
| 图 3-31 GW1NRF-4B-QFN48 Package View 窗口 | 21 |
| 图 3-32 Package View 右键功能 | 22 |
| 图 3-33 差分对显示 | 22 |
| 图 3-34 Top View | 23 |
| 图 3-35 Bottom View | 23 |
| 图 3-36 GW1N-9-WLCSP81M Top View..... | 24 |
| 图 3-37 GW1N-9-WLCSP81M Bottom View | 24 |
| 图 3-38 Chip Array 窗口 | 25 |
| 图 3-39 网格模式约束..... | 26 |
| 图 3-40 宏单元模式约束 | 26 |
| 图 3-41 原语模式约束..... | 27 |
| 图 3-42 Chip Array 右键功能 | 29 |
| 图 3-43 Show Place View 显示 | 29 |
| 图 3-44 鼠标悬浮显示..... | 30 |
| 图 3-45 右键高亮 | 30 |
| 图 3-46 I/O 约束窗口 | 32 |
| 图 3-47 原语约束窗口..... | 32 |
| 图 3-48 组约束窗口 | 33 |
| 图 3-49 预留约束窗口..... | 33 |
| 图 3-50 时钟约束窗口..... | 34 |
| 图 3-51 全局时钟原语约束窗口 | 34 |
| 图 3-52 高速时钟原语约束窗口 | 35 |
| 图 3-53 Vref 约束窗口..... | 35 |
| 图 3-54 Message 窗口..... | 35 |
| 图 4-1 新建约束文件..... | 36 |
| 图 4-2 选择器件 | 37 |
| 图 4-3 保存输出文件..... | 38 |
| 图 4-4 拖拽到 Chip Array 创建 I/O Constraints..... | 40 |
| 图 4-5 拖至 Package View 创建 I/O Constraints | 41 |
| 图 4-6 拖拽到 Chip Array 创建 Primitive Constraints..... | 42 |
| 图 4-7 Group Constraints 编辑器右键菜单..... | 42 |
| 图 4-8 创建 Primitive Group Constraints | 43 |
| 图 4-9 Primitive Group Constraints | 44 |
| 图 4-10 Relative Group Constraints 创建..... | 45 |
| 图 4-11 Relative Group Constraints | 46 |
| 图 4-12 创建 Resource Reservation 约束 | 46 |
| 图 4-13 Resource Reservation..... | 47 |
| 图 4-14 Clock Net Constraints 约束创建 | 48 |

| | |
|--|----|
| 图 4-15 Clock Net Constraints 约束 | 48 |
| 图 4-16 GCLK Primitive Constraints 创建 | 49 |
| 图 4-17 GCLK Primitive Constraints | 49 |
| 图 4-18 HCLK Primitive Constraints 创建 | 50 |
| 图 4-19 HCLK Primitive Constraints..... | 50 |
| 图 4-20 Vref Constraints 创建..... | 50 |
| 图 4-21 Vref Constraints 名字重复 | 51 |
| 图 4-22 拖拽至 Chip Array 窗口生成 Vref Constraints Location 信息..... | 51 |
| 图 4-23 拖拽至 Package View 窗口生成 Vref Constraints Location 信息..... | 52 |

表目录

表 1-1 术语、缩略语 1

1 关于本手册

1.1 手册内容

本手册主要描述高云®半导体 FloorPlanner，介绍高云半导体云源®软件 FloorPlanner 的界面使用以及语法规则，旨在帮助用户快速实现物理约束。因软件版本更新，部分信息可能会略有差异，具体以用户软件版本信息为准。

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：

- [SUG100, Gowin 云源软件用户指南](#)
- [UG290, Gowin FPGA 产品编程配置手册](#)
- [DS102, GW2A 系列 FPGA 产品数据手册](#)

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 |
|--------|---------------------------------------|---------------|
| BSRAM | Block SRAM | 块状静态随机存储器 |
| CFU | Configurable Function Unit | 可配置功能单元 |
| CLKDIV | Clock Divider | 时钟分频器 |
| CLS | Configurable Logic Section | 可配置逻辑块 |
| DCS | Dynamic Clock Selector | 动态时钟选择器 |
| DLLDLY | DLL Delay | DLL 延迟 |
| DQS | Bidirectional Data Strobe Circuit for | DDR 存储器双向数据选通 |

| 术语、缩略语 | 全称 | 含义 |
|--------------|------------------------------------|------------|
| | DDR Memory | 电路 |
| FloorPlanner | FloorPlanner | 物理约束编辑器 |
| FPGA | Field Programmable Gate Array | 现场可编程门阵列 |
| GCLK | Global Clock | 全局时钟 |
| I/O | Input/Output | 输入/输出 |
| IDE | Integrated Development Environment | 集成开发环境 |
| LUT | Look-up Table | 查找表 |
| PCLK | Primary Clock | 主时钟 |
| PLL | Phase-locked Loop | 锁相环 |
| SCLK | Segmented Clock | 分段时钟 |
| SSRAM | Shadow SRAM | 分布式静态随机存储器 |
| VREF | Voltage Reference | 参考电压 |

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

FloorPlanner 是高云半导体面向市场自主研发的物理约束编辑器，支持对 I/O、Primitive（原语）、Group 等属性及位置信息的读取与编辑，同时可根据用户的配置生成新的布局与约束文件，文件中规定了 I/O 的属性信息，原语、模块的位置信息等。**FloorPlanner** 提供了简单快捷的布局与约束编辑功能，提高编写物理约束文件的效率。

FloorPlanner 功能特点：

- 支持用户设计文件、约束文件的读入，约束文件的编辑，以及约束文件的输出
- 支持对用户设计文件中 IO Port、Primitive、Group 约束信息等的显示
- 支持用户新建、编辑、修改约束信息
- 支持 Chip Array 的网格模式、宏单元模式以及原语模式显示
- 支持依据 Package 信息的 Package View 显示
- 支持 Chip Array 和 Package View 的同步显示
- 支持约束位置信息的实时显示及区别显示
- 支持拖拽设置位置信息的功能
- 支持 IO Port 的属性配置功能，支持批量配置
- 支持 Clock Net Constraints 的显示、编辑功能
- 支持约束信息合法性检查的功能
- 支持 Back-annotate Physical Constraints 功能

3 FloorPlanner 界面

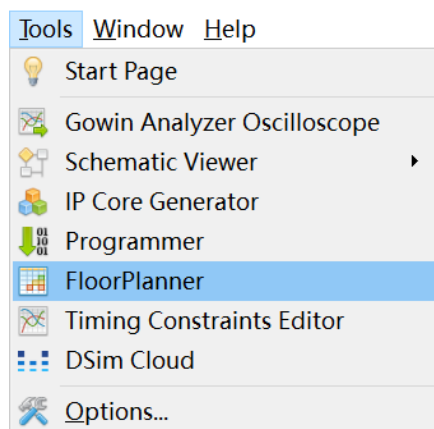
FloorPlanner 能够创建和编辑物理约束，可以提供表格化的约束编辑和高效的网表单元查找功能，提高编写物理约束文件的效率。

3.1 启动

可通过以下三种方式启动 FloorPlanner：

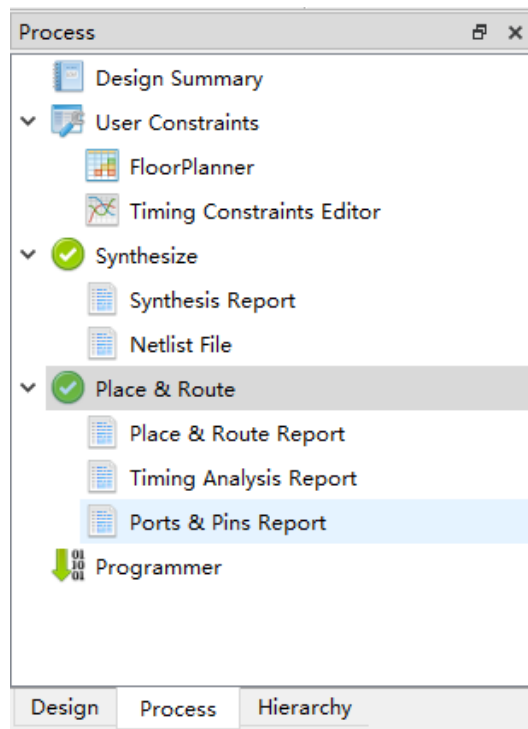
1. 单击“IDE > Tools”，打开“FloorPlanner”，如图 3-1 所示；

图 3-1 菜单栏启动 FloorPlanner



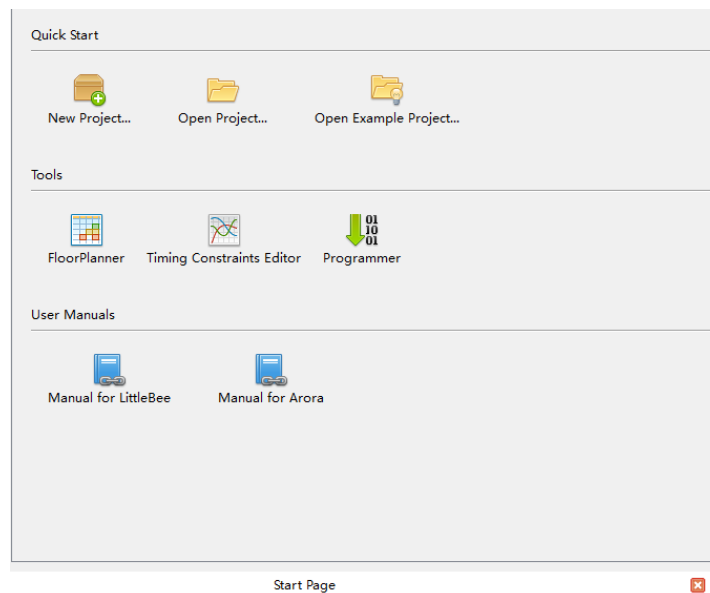
2. 建立工程“Process”窗口运行“Synthesize”成功后，双击“FloorPlanner”，如图 3-2 所示。

图 3-2 Process 窗口启动



- 单击“IDE > Start Page > Tools > FloorPlanner”，打开“FloorPlanner”，如图 3-3 所示。

图 3-3 Start Page 窗口启动

**注!**

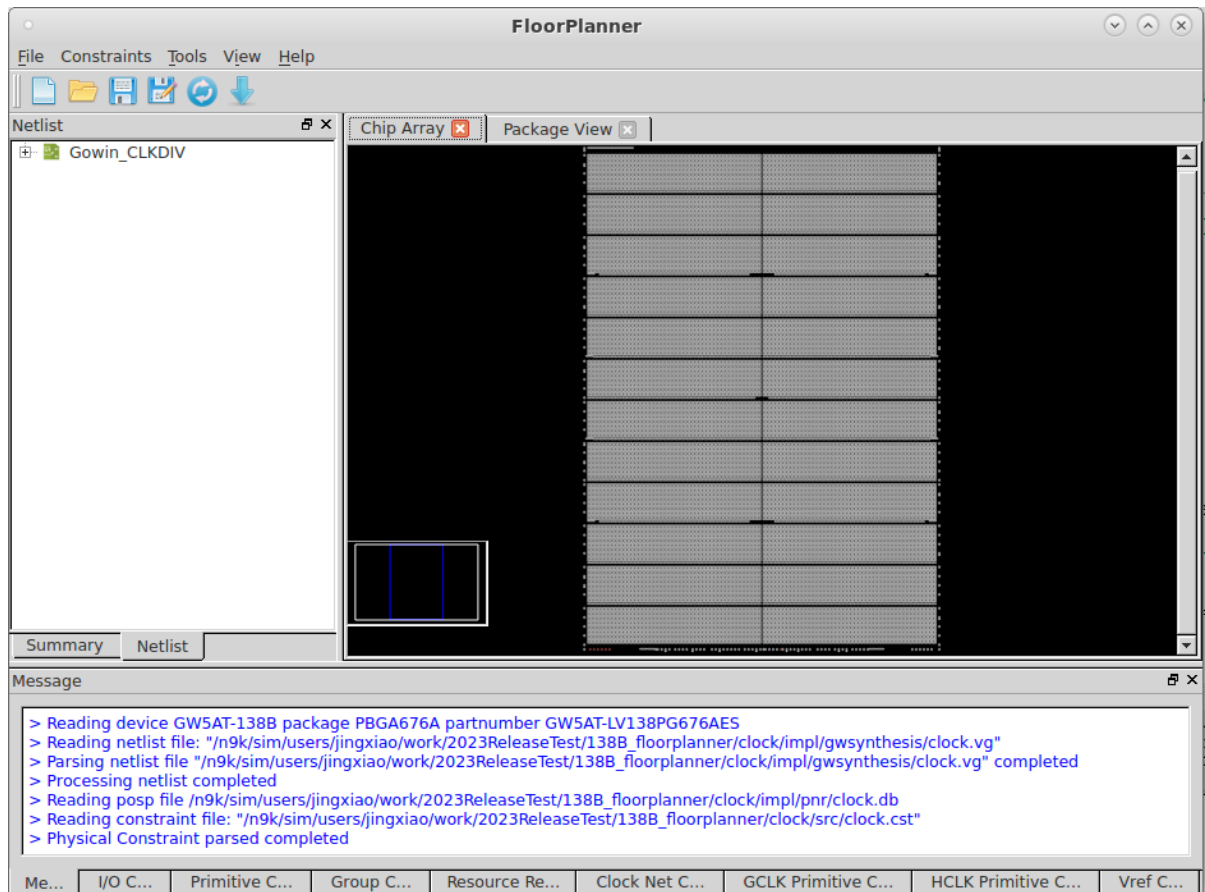
- 如需 FloorPlanner 进行约束，应先加载网表文件；
- 通过第一种和第二种方式启动 FloorPlanner 时，当前工程中的网表文件会自动加载；
- 通过第三种方式启动 FloorPlanner 时，需要通过“File > New”加载网表文件。

3.2 界面

新建或打开 FloorPlanner 界面（包含网表文件），如图 3-4 所示。

界面包括菜单栏、工具栏、Netlist 窗口、Summary 窗口、Chip Array 窗口、Package View 窗口、Message 窗口以及各类约束编辑窗口等。

图 3-4 FloorPlanner 界面



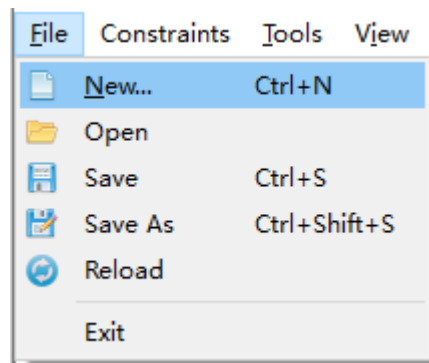
3.2.1 菜单栏

FloorPlanner 的菜单栏分为“File”、“Constraints”“Tools”、“View”及“Help”5个子菜单。

File 菜单

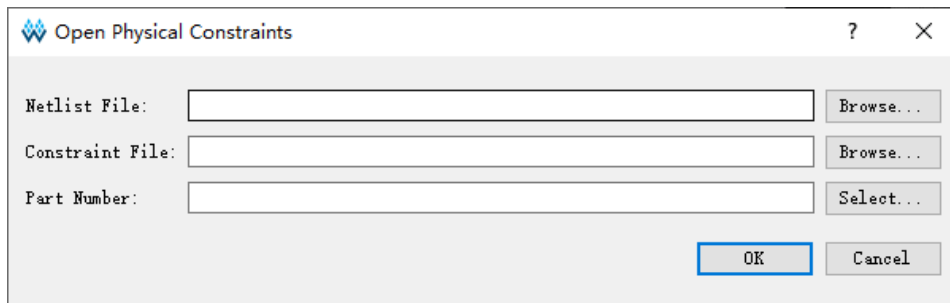
File 菜单如图 3-5 所示。

图 3-5 File 菜单



- **New:** 新建约束，添加用户设计，设置器件信息；
- **Open:** 打开约束，添加用户约束，设置器件信息如图 3-6；
- **Reload:** 当在磁盘或工程中对物理约束文件、布局文件进行修改后，可以重新加载；
- **Save:** 当前约束信息的修改信息覆盖原约束文件；
- **Save As:** 将当前约束信息的修改信息输出到用户指定的文件中，默认采用网表文件名作为约束文件名称，用户可修改；
- **Exit:** 退出 FloorPlanner。

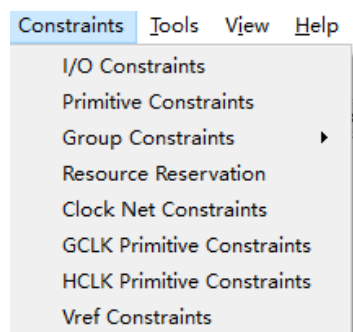
图 3-6 Open Physical Constraints



Constraints 菜单

Constraints 菜单栏如图 3-7 所示。

图 3-7 Constraints 菜单



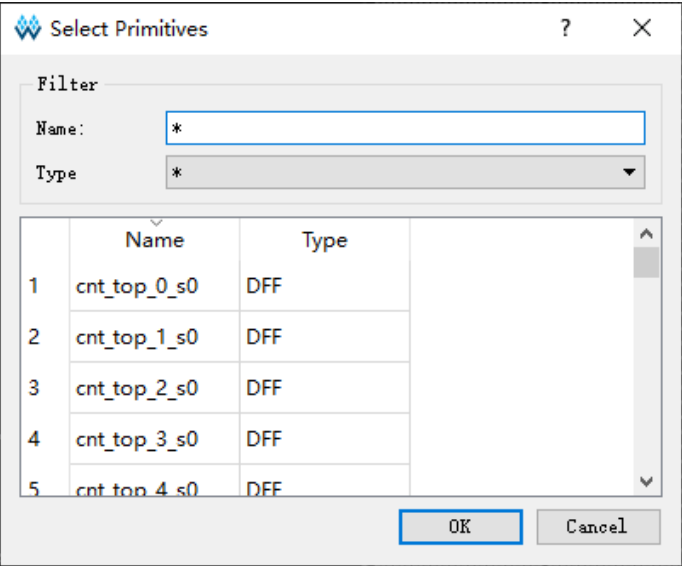
Primitive Constraints

创建 Primitive 约束，选择 Primitive 创建对应的约束，右击选择 “Select Primitives”，可弹出如图 3-8 所示对话框。

- 1. 可通过 Primitives 名称或类型进行查找，选择对应的 Primitive;
- 2. 单击 “OK”，产生约束信息，约束信息显示在主界面底部的 “Primitive Constraints” 约束编辑窗口中;
- 3. 用户可在编辑窗口中通过输入或拖拽的方式设置位置信息。

注！
所约束的位置在 Chip Array 窗口中呈浅蓝色高亮显示。

图 3-8 原语查找对话框



Group Constraints

Group Constraints, 包括 New Primitive Group 和 New Relative Group, 各功能介绍如下:

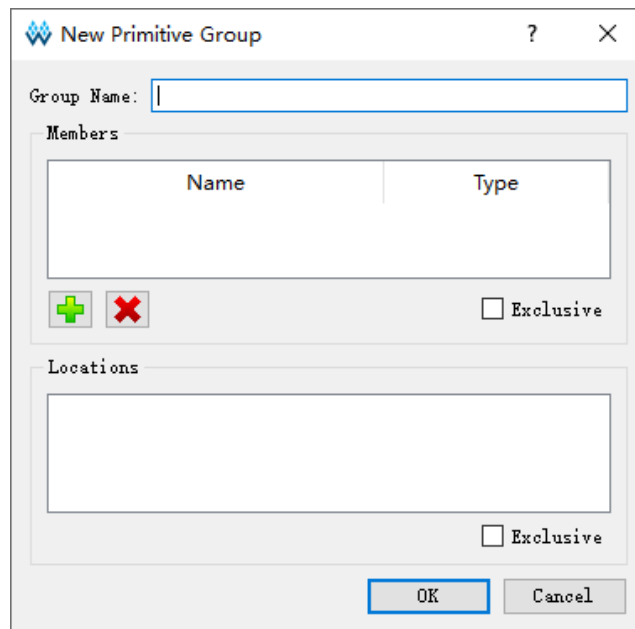
创建 Primitive Group

- 1. 创建 Primitive Group 约束，右击选择 “New Primitive Group”，弹出如图 3-9 所示对话框;
- 2. 用户可设置 Group 的名称、包含的 Primitive 位置信息，以及 Group 的 Exclusive 信息；通过 “+” 和 “-” 两个按钮实现 Primitive 的添加和删除，正确创建的 Primitive Group 如图 3-10 所示;

- 注！
- Group 的名称、包含的 Primitive、Group 的位置为必填项;
 - 可通过以下方式输入 Group 的位置信息:

- 通过手动方式输入;
 - 建立 Group 约束前, 在 “Chip Array” 窗口中, 复制位置, 粘贴到 “New Primitive Group > Locations” 中。
3. Primitive Group 创建配置完成后, 单击 “OK”, 工具此时会对 Group 的位置信息进行语法检查。
- 若位置信息不合理或不合法, 会弹出如图 3-11 和如图 3-12 所示的提示框, 用户需重新修改位置信息;
 - 若无错误提示, 单击 “OK”, 在 Chip Array 中会显示可用的位置。
4. 新产生的组约束显示在主界面底部的 “Group Constraints” 约束编辑窗口中, 在 “Group Constraints” 约束编辑窗口中, 双击 Primitive Group 约束可打开如图 3-10 所示的对话框, 重新进行编辑修改。

图 3-9 新建原语组对话框



The image shows a dialog box titled "New Primitive Group". It has a "Group Name:" label followed by a text input field. Below this is a section labeled "Members" which contains a table with two columns: "Name" and "Type". Under the "Members" section are two buttons: a green plus sign and a red minus sign, and a checkbox labeled "Exclusive". Below the "Members" section is a section labeled "Locations" which contains a large empty text area. Under the "Locations" section is a checkbox labeled "Exclusive". At the bottom of the dialog are two buttons: "OK" and "Cancel".

图 3-10 正确原语组对话框

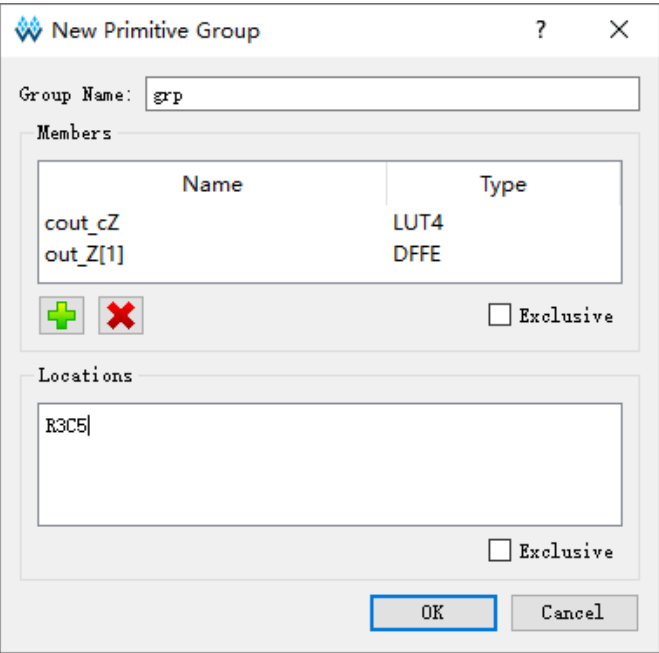


图 3-11 无效位置

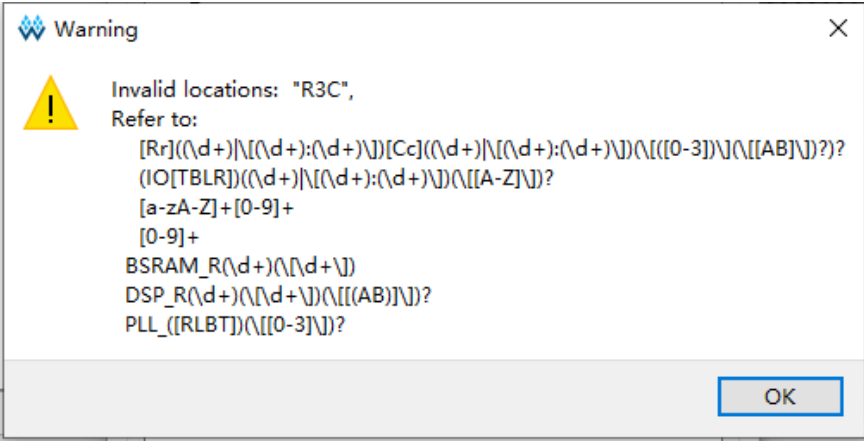
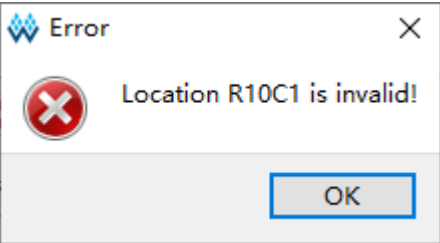


图 3-12 无效位置



创建 Relative Group

1. 创建 Relative Group 约束，右击选择“New Relative Group”，弹出如 图

3-13 所示对话框；

2. 用户可设置 Group 的名称、包含的 Primitive 以及各 Primitive 对应的相对位置信息；可通过“+”和“-”实现 Primitive 的添加和删除，创建成功的 Relative Group 约束如图 3-14 所示；

注！

- Group 的名称、包含的 Primitive 及 Relative Location 为必填项；
- 可通过以下方式输入 Group 的位置信息：
 - 通过手动方式输入；
 - 在建立 Group 约束前，在“Chip Array”窗口中，复制位置，粘贴到“New Relative Group > Relative Location”中。
- 3. 配置完成后单击“OK”，产生约束信息。
- 4. 产生的约束信息显示在主界面底部的“Group Constraints”约束编辑窗口中；在编辑窗口中，双击约束，重新打开如图 3-14 所示的对话框，可进行编辑修改。

图 3-13 创建相对位置组对话框

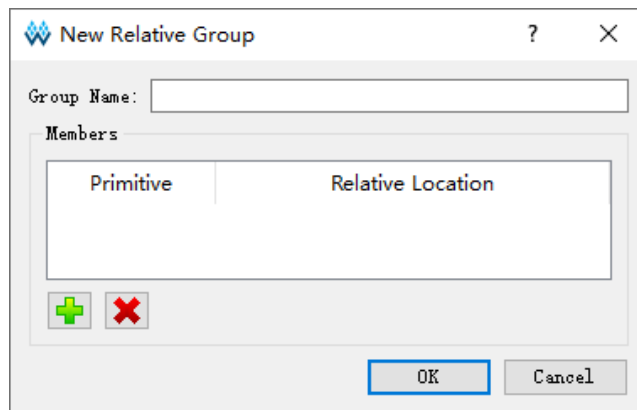
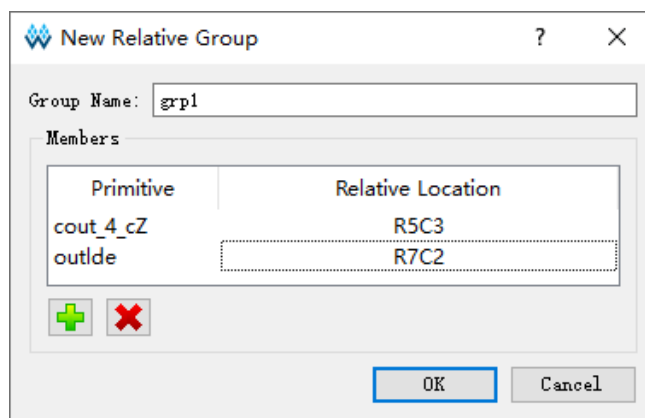


图 3-14 正确的相对组对话框



Resource Reservation

1. 创建 Resource Reservation 约束，在主界面底部的 “Resource Reservation” 约束编辑窗口右击选择 **Reserve Resources**，新建一条约束；
2. 通过输入或拖拽的方式设置位置信息；
3. 双击 “Attribute” 栏或单击 “Attribute” 栏下拉框可设置预留位置的属性，如图 3-15 所示。

注！
Name 属性用于区分不同的预留约束，不可修改该名称。

图 3-15 预留约束

| | Name | Locations | Attribute |
|---|-----------|-------------------|-----------|
| 1 | reserve_0 | drag or type t... | ALL |
| | | | ALL |
| | | | LUT |
| | | | REG |

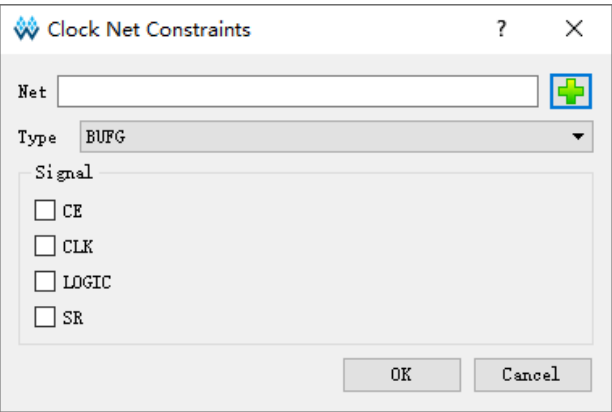
Clock Net Constraints

创建全局时钟分配约束，该约束的数目有限制，会对约束合法性时进行相应检查。右击选择 “Clock Net Constraints”，可弹出如图 3-16 所示对话框，进行如下操作：

1. 单击 “+” 按钮，选择对应 Net；
2. 通过 “Type” 下拉列表，选择 “BUFG”、“BUFG[0]~[7]”、“BUFS”、“LOCAL_CLOCK”；
3. 通过 “CE”、“CLK” 等复选框配置 Signal 类型，配置完成后，单击 “OK”，产生约束信息，显示在主界面底部的 “CLOCK Net Constraints” 约束编辑窗口中，在编辑窗口中，双击，重新打开约束对话框，进行编辑。

注！
当 “Type” 选择 LOCAL_CLOCK 时 Signal 复选框为置灰不可配置状态。

图 3-16 时钟约束



GCLK Primitive Constraints

用于创建针对 DCS 和 DQCE 的全局时钟约束，根据器件的全局时钟分布约束指定的 Instance 到具体的全局时钟，在主界面底部的“GCLK Primitive Constraints”约束编辑窗口右击选择“Select GCLK Primitive”，可弹出如图 3-17 所示对话框。相关操作如下所示：


1. 通过单击“”按钮，选择相应的 GCLK 原语，若设计无 GCLK 原语则无法添加；
2. 通过“Position”下的单选框以及对应的下拉列表配置全局时钟位置；
3. 单击“OK”，产生约束信息，显示在主界面底部的“GCLK Primitive Constraints”约束编辑窗口中，在编辑窗口中，双击，重新打开约束对话框，可进行约束编辑修改。

图 3-17 创建全局时钟原语约束（GW1N-1）

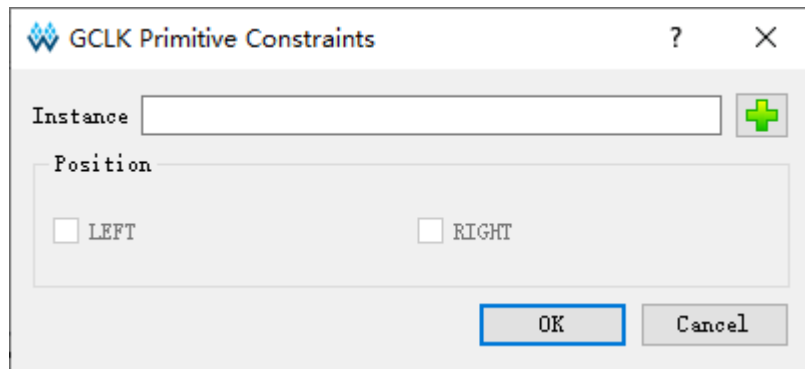
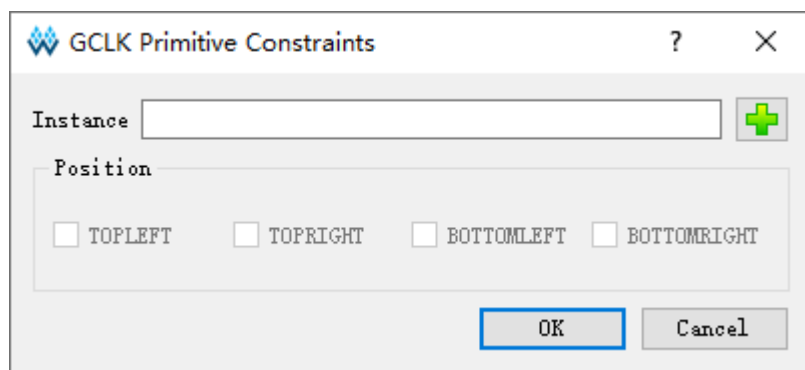


图 3-18 创建全局时钟原语约束（GW2A-18）



注！


- 当选择“Instance”后，“Position”变为高亮。
- 根据器件不同可用 Position 不同，不同全局时钟原语可用 Position 也会不同。

HCLK Primitive Constraints

创建针对 HCLK 相关原语进行的约束，指定其约束在器件的高速时钟位

置上，在主界面底部的“HCLK Primitive Constraints”约束编辑窗口右击选择“Select HCLK Primitive”，可弹出如所图 3-19 示对话框。相关操作如下所示：

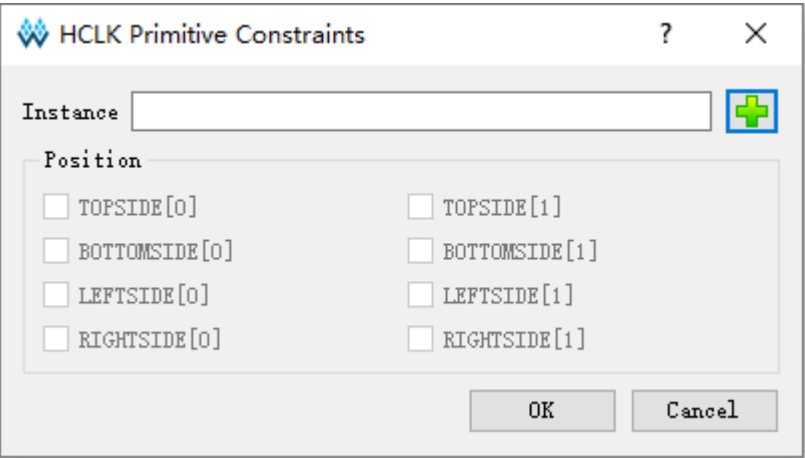
1.

用户可通过单击“”按钮选择相应的 HCLK 原语，若设计中无符合的原语则无法添加；
2.

通过“Position”下的单选框以及对应的下拉列表配置高速时钟位置；
3.

单击“OK”，产生约束信息，显示在主界面底部的“HCLK Primitive Constraints”约束编辑窗口中，在编辑窗口中，双击，重新打开约束对话框，可进行约束编辑修改。

图 3-19 创建高速时钟原语约束

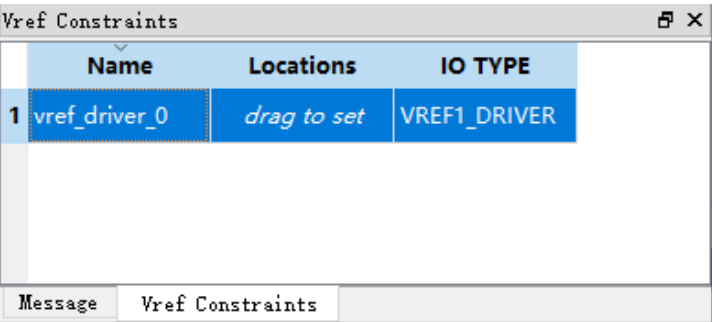


- 注！
- 当选择“Instance”后，“Position”变为高亮。
 - 根据工程中器件不同可用 Position 不同，不可用 Position 置灰不可勾选。

Vref Constraints

创建新的 Vref Driver，用于配置 IO Port 的参考电压，在主界面底部的“Vref Constraints”约束编辑窗口中右击选择 Define Vref Driver，新建一条约束，如图 3-20 所示。

图 3-20 参考电压约束



注！

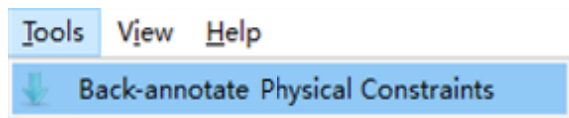
- 可通过拖拽方式指定 Vref 约束位置；
- 可通过双击修改 Vref 的名称。

Tools 菜单

Tools 菜单如图 3-21 所示。

Back-annotate Physical Constraints: 将各 Primitive 和 IO Port 布局信息反标至物理约束文件中。

图 3-21 Tools 菜单



1. 单击“Tools > Back-annotate Physical Constraints”弹出对象选择对话框如图 3-22 所示。Back-Annotate Physical Constraints 功能只有工程中运行 Place & Route 成功后，通过工程启动 FloorPlanner 才有效；
2. Back-annotate Physical Constraints 对话框中可以选择一个或者多个对象，单击“OK”按钮弹出“Save as”对话框，打印其布局信息至物理约束文件中；
3. 如图 3-23 所示，在 Back-annotate Physical Constraints 对话框中选择 Port 和 Port Attribute 后生成的物理约束文件。

图 3-22 Back-annotate Physical Constraints 对话框

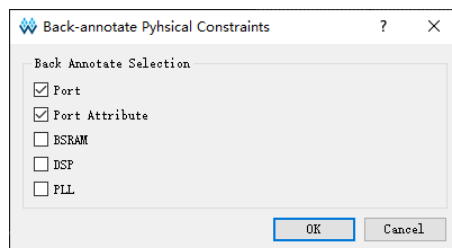


图 3-23 反标 Port 布局信息

```
1 //Copyright (C)2014-2023 Gowin Semiconductor Corporation.
2 //All rights reserved.
3 //File Title: Physical Constraints file
4 //Tool Version: V1.9.9 (64-bit)
5 //Part Number: GW2AN-UV18XUG484C7/I6
6 //Device: GW2AN-18X
7 //Created Time: Fri 11 10 11:22:21 2023
8
9 IO_LOC "CIN" P15;
10 IO_PORT "CIN" PULL_MODE=NONE DRIVE=OFF BANK_VCCIO=3.3;
11 IO_LOC "COUT" AE16;
12 IO_PORT "COUT" PULL_MODE=NONE DRIVE=8 BANK_VCCIO=3.3;
13 IO_LOC "IO" P16;
14 IO_PORT "IO" PULL_MODE=NONE DRIVE=OFF BANK_VCCIO=3.3;
15 IO_LOC "I1" M19;
16 IO_PORT "I1" PULL_MODE=NONE DRIVE=OFF BANK_VCCIO=3.3;
17 IO_LOC "I3" AB15;
18 IO_PORT "I3" PULL_MODE=NONE DRIVE=OFF BANK_VCCIO=3.3;
19
```

View 菜单

View 菜单如图 3-24 所示，主要用于控制工具条、窗口的显示以及 Chip Array 和 Package View 两个视图的放大、缩小等。各子菜单介绍如下：

- Toolbars: 用于控制工具栏快捷按钮的显示；
- Windows: 用于控制各个窗口的显示，如图 3-25 所示；
- Zoom In: 用于放大 Chip Array 视图或 Package View 视图；
- Zoom Out: 用于缩小 Chip Array 视图或 Package View 视图；
- Zoom Fit: 按照窗口大小缩放 Chip Array 视图或 Package View 视图。

图 3-24 View 菜单

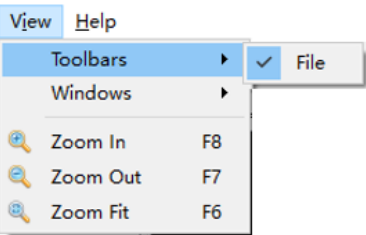
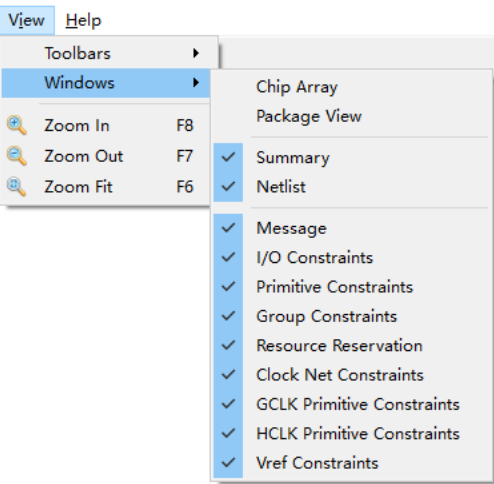


图 3-25 Windows 菜单



Help 菜单

Help 菜单用于提示软件的版本号及版权信息。

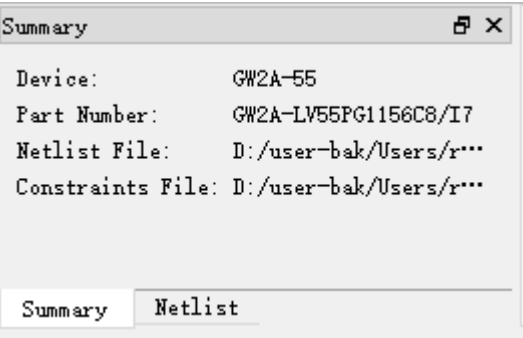
3.2.2 Summary 和 Netlist 窗口

Summary 和 Netlist 两个窗口可显示当前工程的器件信息、芯片型号信息、用户设计及约束文件的路径信息、Netlist 信息等。

Summary 窗口

Summary 窗口如图 3-26 所示，用于显示当前工程中所用的器件信息，包括 Device 和 Part Number，以及用户输入的设计文件和约束文件。

图 3-26 Summary 窗口



Netlist 窗口

Netlist 窗口如图 3-27 所示，以树形结构显示用户设计中的 Ports、Primitives、Nets 和 Module 以及对应的数量信息。

注！

- Port、Primitive 等名称采用全路径方式进行显示，默认按字母升序排序；

- Port 和 Net 的显示采用 Bus 和非 Bus 相结合的显示方式，如图 3-28 所示；
- Module 采用层级的方式显示，各 Module 后可显示各 Module 中各类型的 Instance 数目，如图 3-29 所示；

图 3-27 Netlist 窗口

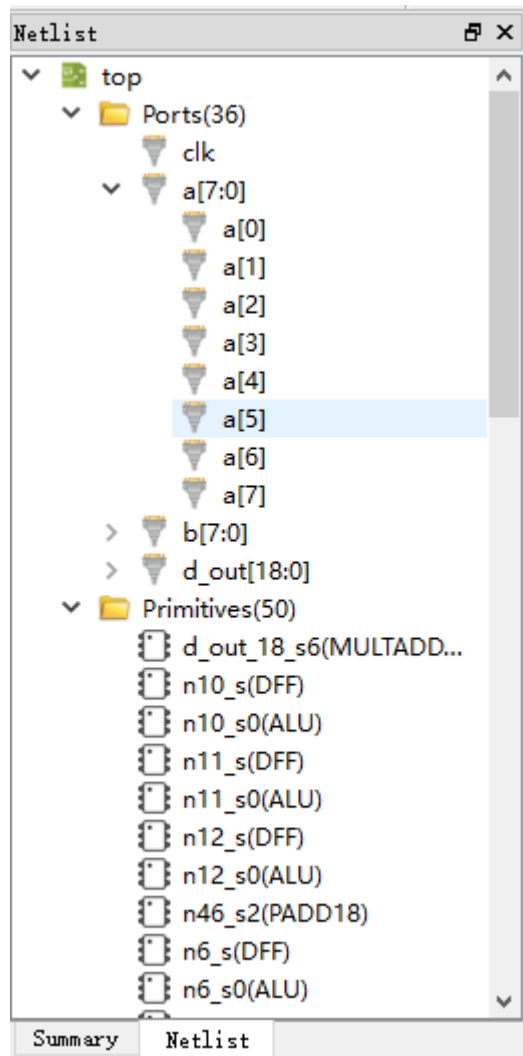


图 3-28 BUS 和非 BUS 结合显示

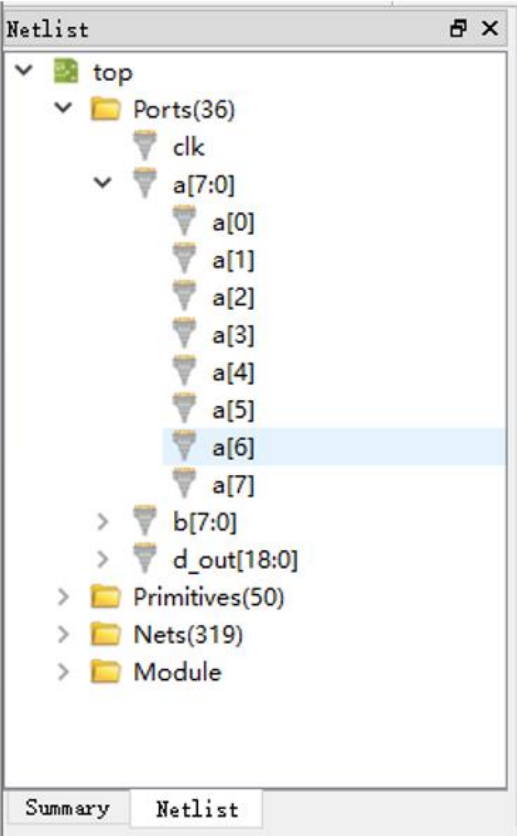
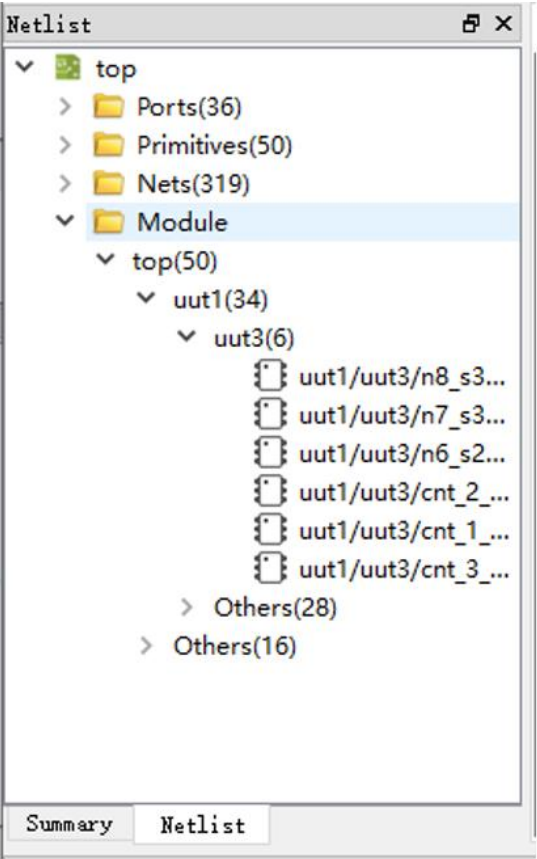


图 3-29 层级显示



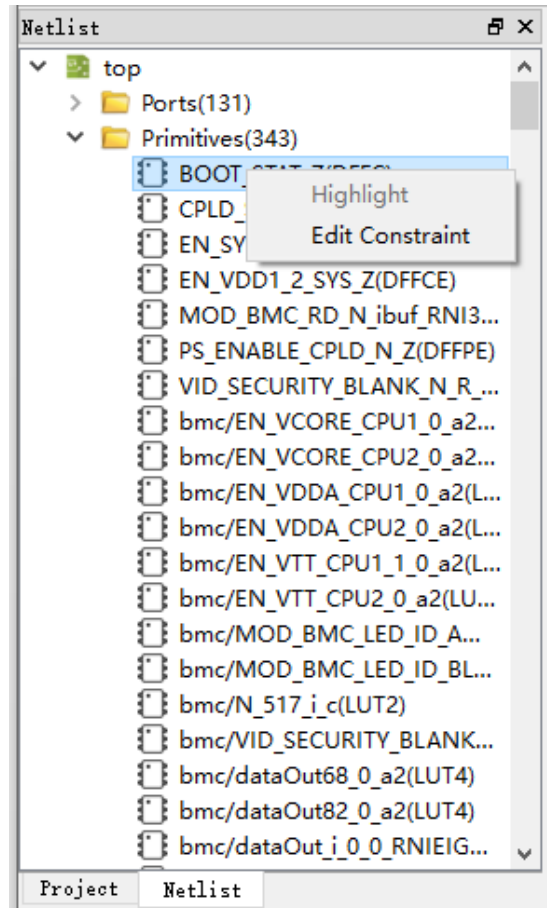
Netlist 窗口提供右键菜单功能，具有如下功能：

- **Highlight:** 可实现在 Chip Array 中高亮显示对应的约束位置；
- **Edit Constraint:** 编辑对应约束信息的功能。

注！

如当前 Primitive 或 Port 无位置约束，则高亮显示功能不可用，如图 3-30 所示。

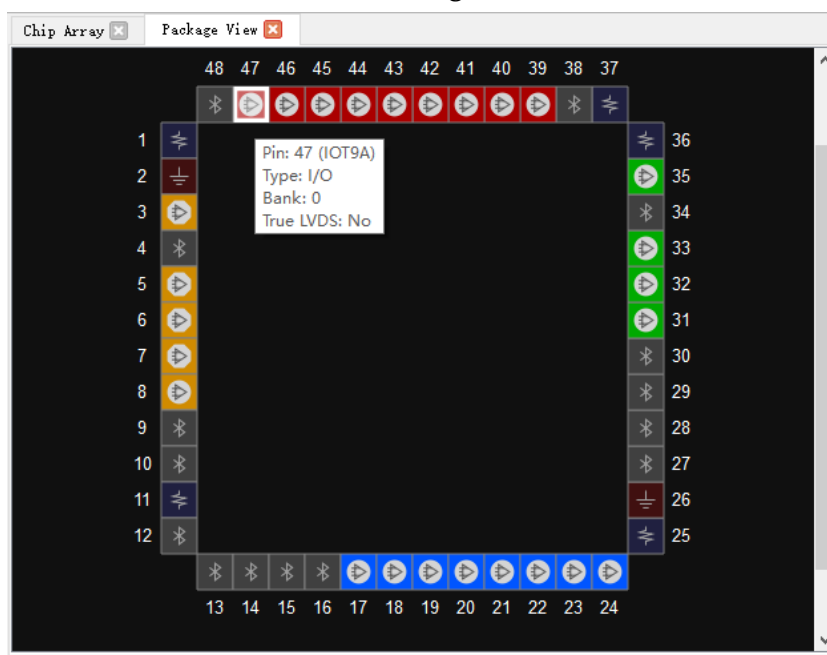
图 3-30 Netlist 右键功能



3.2.3 Package View 窗口

以 GW1NRF-4B-QFN48 为例，Package View 窗口如图 3-31 所示，该窗口以器件 package 信息为基础显示器件的封装信息，显示用户 I/O、电源、地等管脚。将鼠标放置某个位置上时，会悬浮显示该位置的 I/O 信息，包括 I/O 的 Type、Bank 以及 LVDS 信息等。

图 3-31 GW1NRF-4B-QFN48 Package View 窗口



用户 I/O、电源、地管脚使用不同的符号和颜色来区分。不同 BNAK 的 IO 引脚的颜色不同，图中管脚符号定义如下所示：

- “” 表示用户 I/O；
- “” 表示 VCCIO；
- “” 表示 VSS；
- “” 表示蓝牙接口。

Package View 支持右键菜单如图 3-32 所示，相关功能如下：

- Zoom In: 放大 Package View 视图；
- Zoom Out: 缩小 Package View 视图；
- Zoom Fit: 按照窗口大小缩放 Package View 视图；
- Show Differential IO Pairs: 显示差分对，如图 3-33 所示，红线相连的为 一对差分对；
- Top View: Package View 以顶部视图进行显示，默认以顶部视图进行显示。如图 3-34 所示为 GW1N-9-WLCSP64 封装的顶部视图，坐标原点在左上角，如图 3-36 所示为 GW1N-9-WLCSP81M 封装的顶部视图，坐标原点在右上角；
- Bottom View: Package View 以底部视图进行显示。如图 3-35 所示为

GW1N-9-WLCSP64 封装的底部视图，坐标原点在右上角，如图 3-37 所示为 GW1N-9-WLCSP81M 的底部视图，坐标原点在左上角。

图 3-32 Package View 右键功能

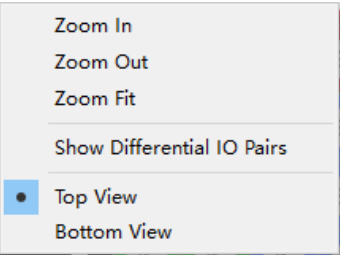


图 3-33 差分对显示

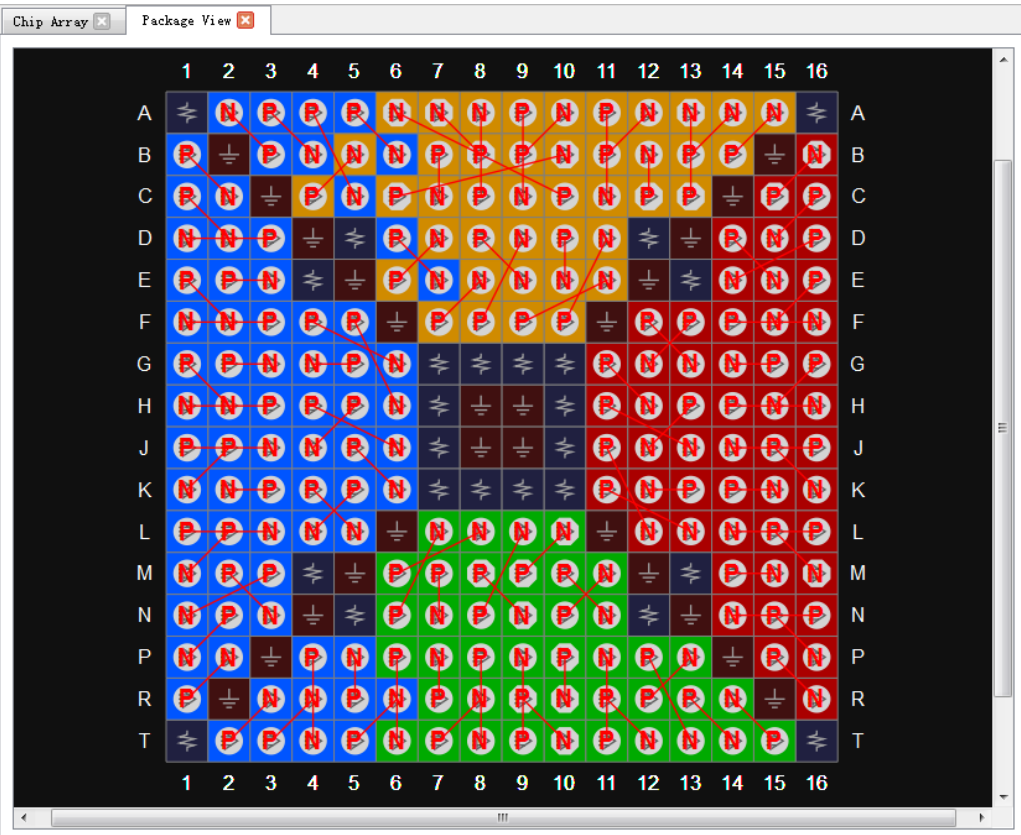


图 3-34 Top View

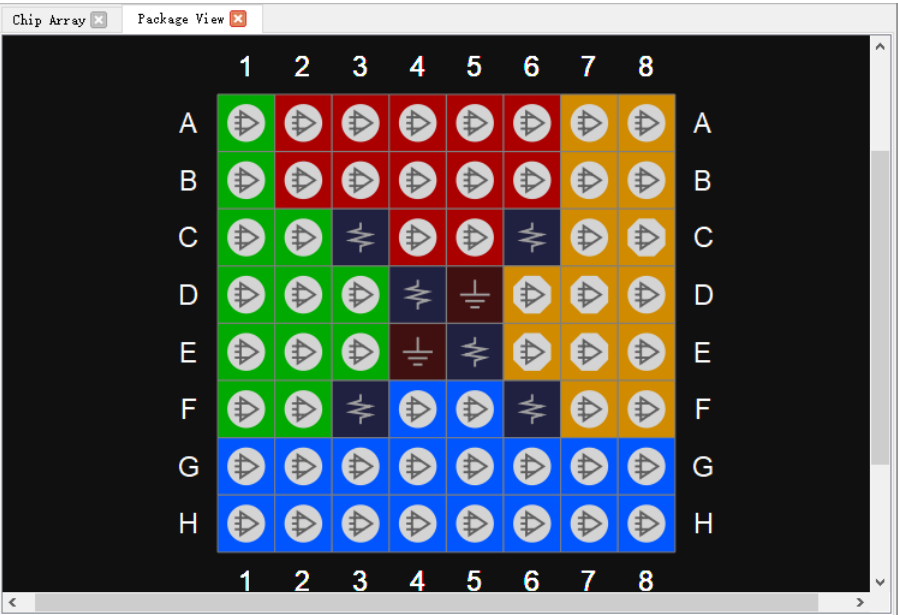


图 3-35 Bottom View

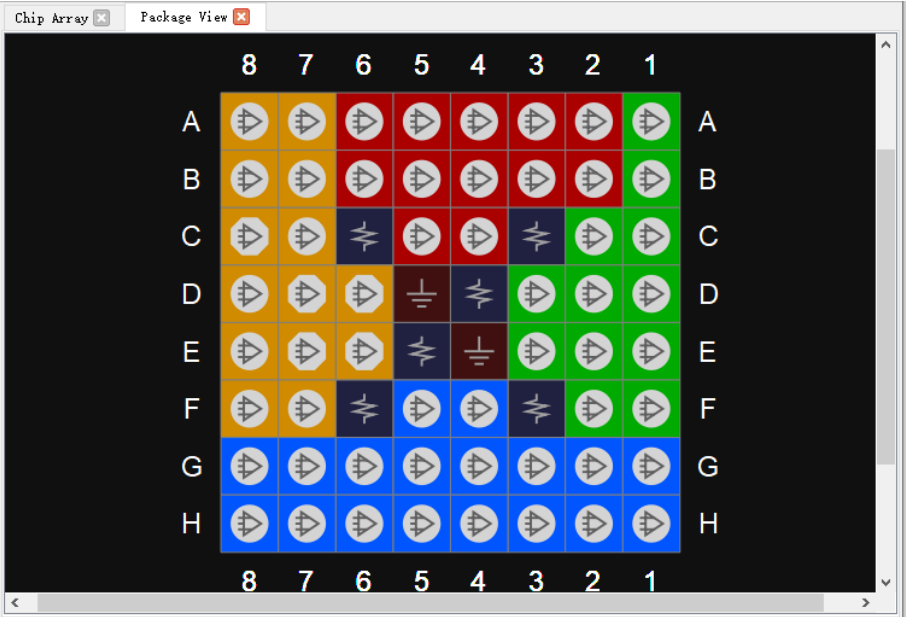


图 3-36 GW1N-9-WLCSP81M Top View



图 3-37 GW1N-9-WLCSP81M Bottom View



Package View 支持 IO Port 约束位置的显示, 可以通过从 Netlist 窗口或底部 I/O Constraints 窗口中将 IO Port 拖拽到 Package View 窗口来约束 IO Port 的位置。拖拽时鼠标处会显示拖拽 Port 的名称, 并且不可约束管脚呈现

置灰不可拖拽状态。

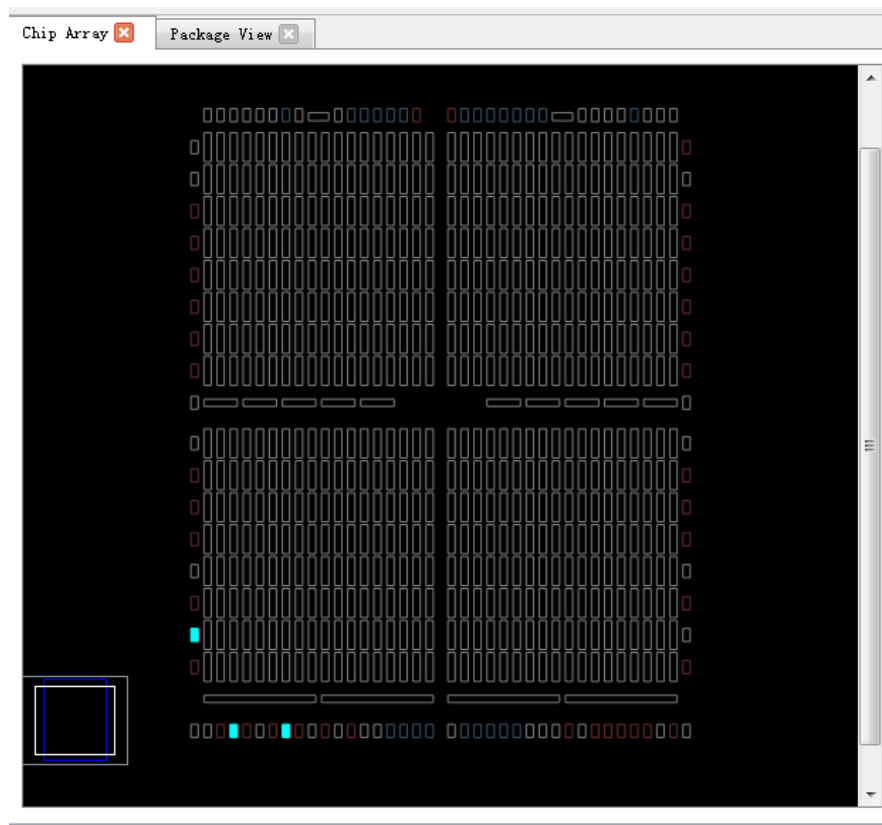
3.2.4 Chip Array 窗口

FloorPlanner 的 Chip Array 窗口如图 3-38 所示, Chip Array 窗口根据芯片的行列信息显示芯片的 I/O、CFU、CLU、DSP、PLL、BSRAM、DQS 等资源的分布, 实现对所有约束位置的实时显示, 支持放大、缩小、复制位置、悬停显示、拖拽等功能。

其中, I/O 是器件裸片的所有 I/O 位置, 且会以不同颜色区分 I/O:

- 白色: 该封装中封装出的 I/O 位置;
- 红色: 该封装中未封装的 I/O 位置;
- 蓝色: 如果是 GW2AR-18、GW1NR-4、GW1NR-9 等 SIP 封装的器件, 则会有蓝色标记 I/O, 表示内嵌配置 I/O 位置。

图 3-38 Chip Array 窗口



Chip Array 分为网格模式、宏单元模式、原语模式三种显示模式。

- 网格模式: 以 GRID 为单位宏观显示约束位置, 如图 3-39 所示;
- 宏单元模式: 以 CLS、IOBLK 等单位显示约束位置, 如图 3-40 所示;
- 原语模式: 以 REG、LUT 等单位显示约束位置, 如图 3-41 所示。

图 3-39 网格模式约束

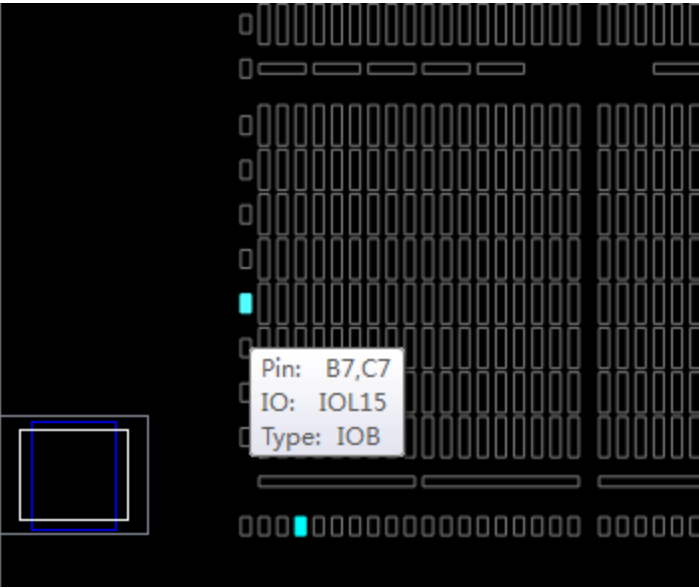


图 3-40 宏单元模式约束

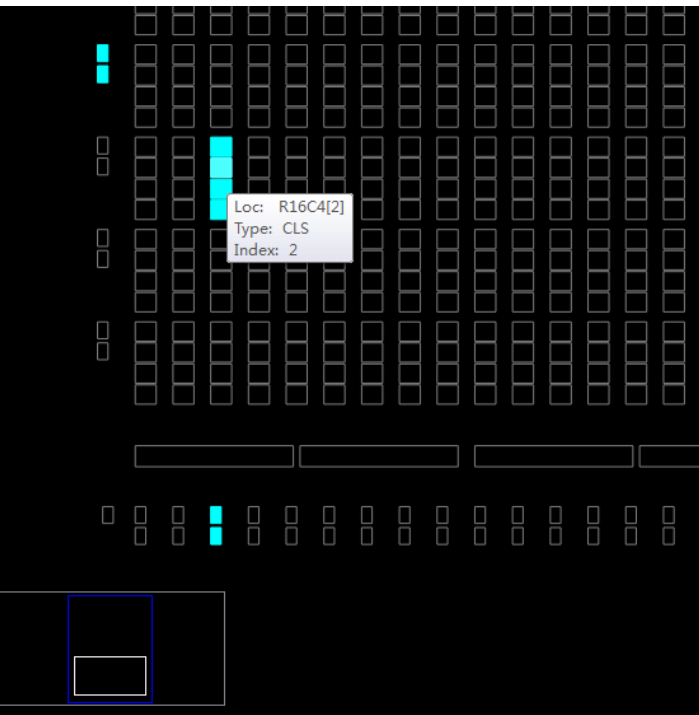
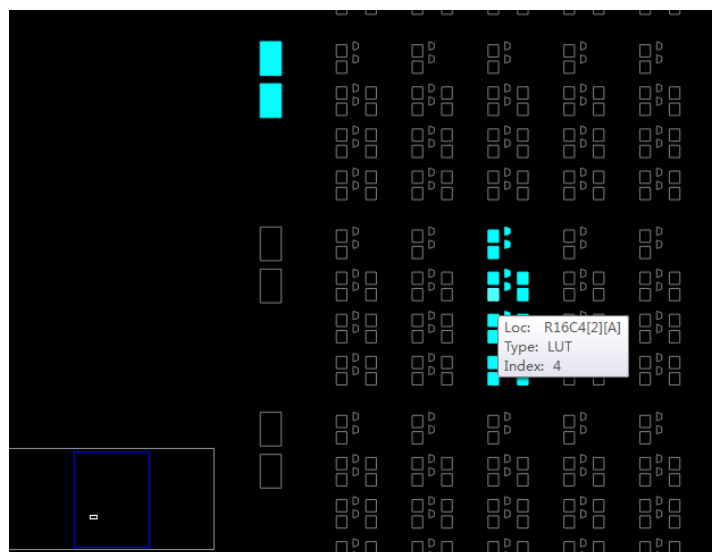


图 3-41 原语模式约束



Chip Array 支持以下拖拽功能：

- 从 Netlist 窗口拖拽到 Array 窗口，用于产生约束、指定约束位置；
- 从约束编辑窗口拖拽到 Array 窗口，用于指定约束位置。

Chip Array 窗口内置 chip 子窗口，用于实时显示当前视窗相对于整个器件的位置，拖动 chip 子窗口中的白色框，Chip Array 的视窗将跟随移动。同时，Chip Array 窗口采用不同颜色区分约束类型、显示约束位置，各颜色的含义分别介绍如下：

- 白色：用于显示处于选择状态或正在高亮显示的约束位置；
- 深蓝色：用于显示预留约束的位置信息，表示该位置不能再被占用；
- 浅蓝色：用于显示 I/O 和 Primitive 约束在某个 GRID 或 range 内，界面上显示为浅蓝色。

Chip Array 窗口支持右键菜单，具有如下功能：

- Zoom In: 放大 Chip Array 视图；
- Zoom Out: 缩小 Chip Array 视图；
- Zoom Fit: 按照窗口大小缩放 Chip Array 视图；
- Show Constraints View: 显示 Chip Array 的 instance 约束视图；
- Show Place View: 显示 Chip Array 的 instance 布局视图，只有在工程运行完 Place & Route 后启动 FloorPlanner 才有效，否则置灰；
- Show Multi-View: 同时显示 Chip Array 的 instance 约束和布局的复合视图，只有在工程运行完 Place & Route 后启动 FloorPlanner 才有效，否则置灰；
- Show In-Out Connection: 在 Place View 中显示和选中 instance 输入输

出连接的 instance 位置，只有在 Chip Array 窗口是 Show Place View > All Instance 视图中选中某个 instance 才能够使用，否则置灰；

- **Show In Connection:** 在 Place View 中显示和选中 instance 输入连接的 instance 位置，只有在 Chip Array 窗口是 Show Place View > All Instance 视图中选中某个 instance 才能够使用，否则置灰；
- **Show Out Connection:** 在 Place View 中显示和选中 instance 输出连接的 instance 位置，只有在 Chip Array 窗口是 Show Place View > All Instance 视图中选中某个 instance 才能够使用，否则置灰；
- **Unhighlight All:** 对于选中位置或者区域消除高亮；
- **Copy Location:** 复制选中的位置或者区域，若 Chip Array 窗口中有 GRID、Block 等处于选中状态，则右键菜单中的“Copy Location”功能可用，否则功能不可用，如图 3-42 所示。

在 Show Place View 中，可对 Lut、Reg 的密度进行显示，如图 3-43 所示，详情如下：

- **ALL Instance:** 显示所有 Instance 的 place 情况，5 个以内呈淡绿色、6-10 个呈绿色、10 个以上呈深绿色；
- **Only Lut:** 只显示所有 Lut 的 place 情况，2 个以内呈淡绿色、3-4 个呈绿色、4 个以上呈深绿色；
- **Only Dff:** 只显示所有 Reg 的 place 情况，2 个以内呈淡绿色、3-4 个呈绿色、4 个以上呈深绿色。

在 Show Place View > ALL Instance 可以查看设计中所有 instance 的布局情况：

- 在 Chip Array 窗口中鼠标悬浮至 instance 布局位置，可显示该位置具体布局的 instance 名称，如图 3-44 所示；
- 在 Netlist 窗口中选中某个具体的 instance 右键选择 Highlight，该 instance 的布局位置会在 Chip Array 窗口中高亮显示，如图 3-45 所示。

注！

通过“Ctrl”键+鼠标左键拖动，可选取区域，右击选择 Copy Location，可复制所选区域的位置信息，复制的位置可直接粘贴到任意约束编辑窗口中。

图 3-42 Chip Array 右键功能

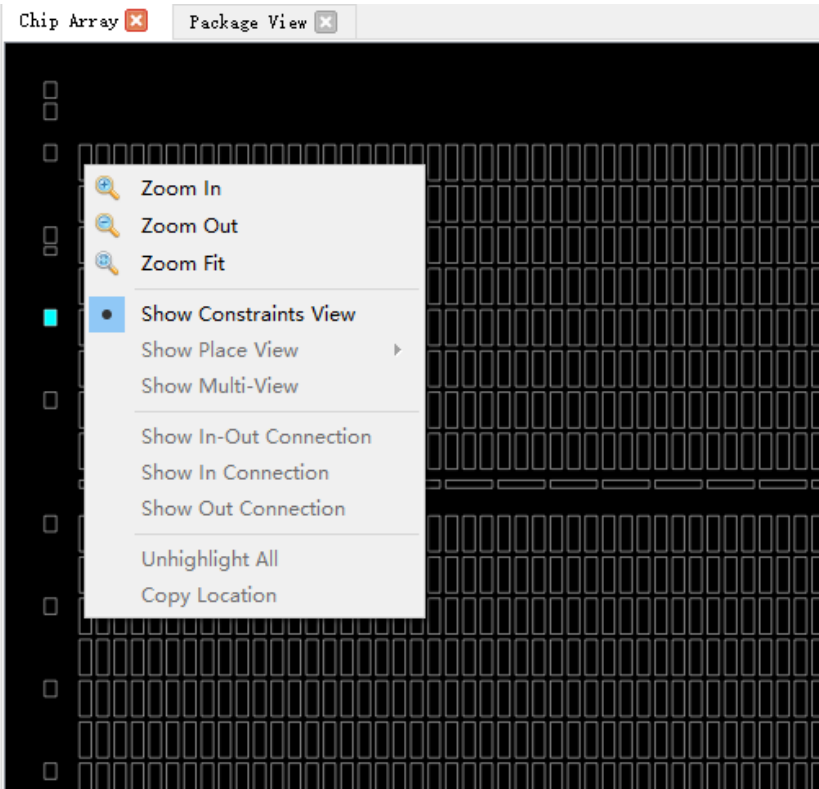


图 3-43 Show Place View 显示

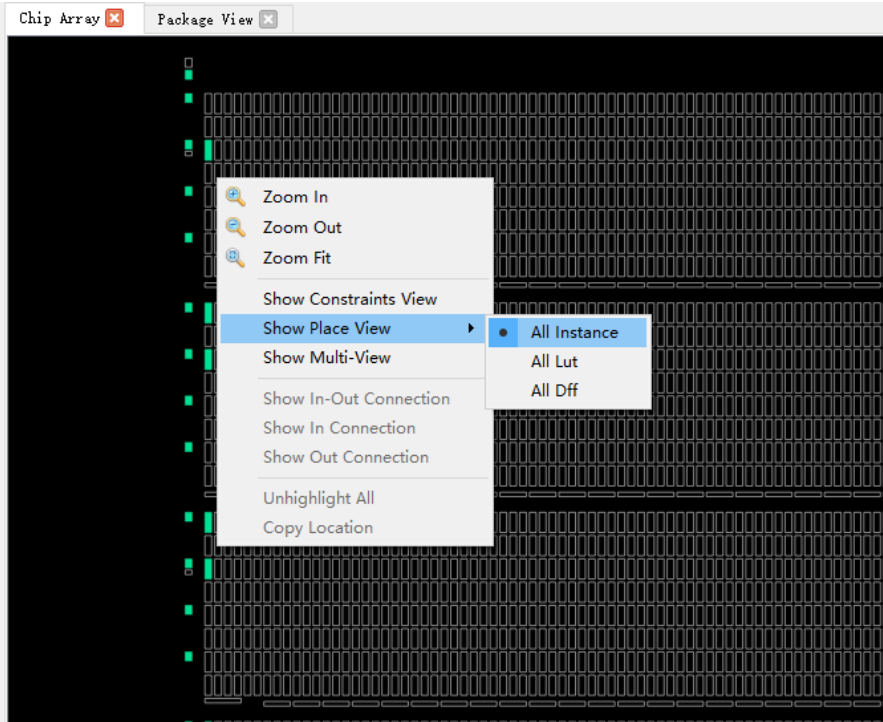


图 3-44 鼠标悬浮显示

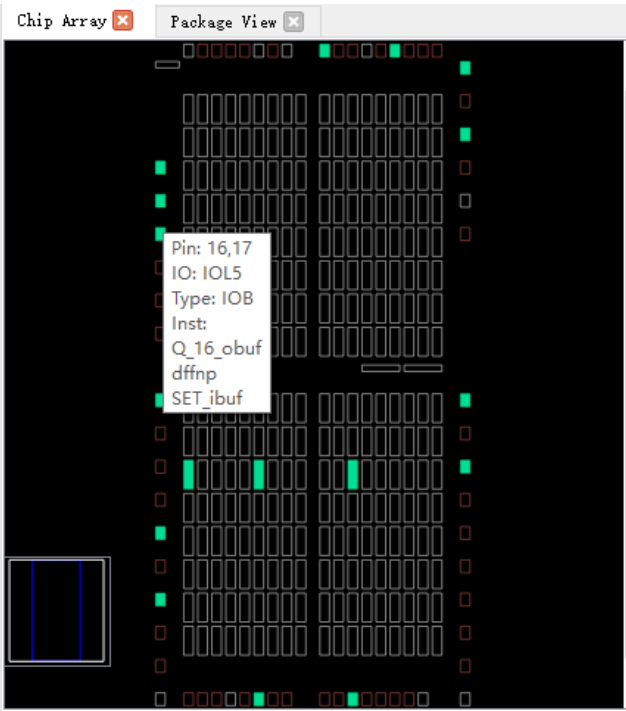
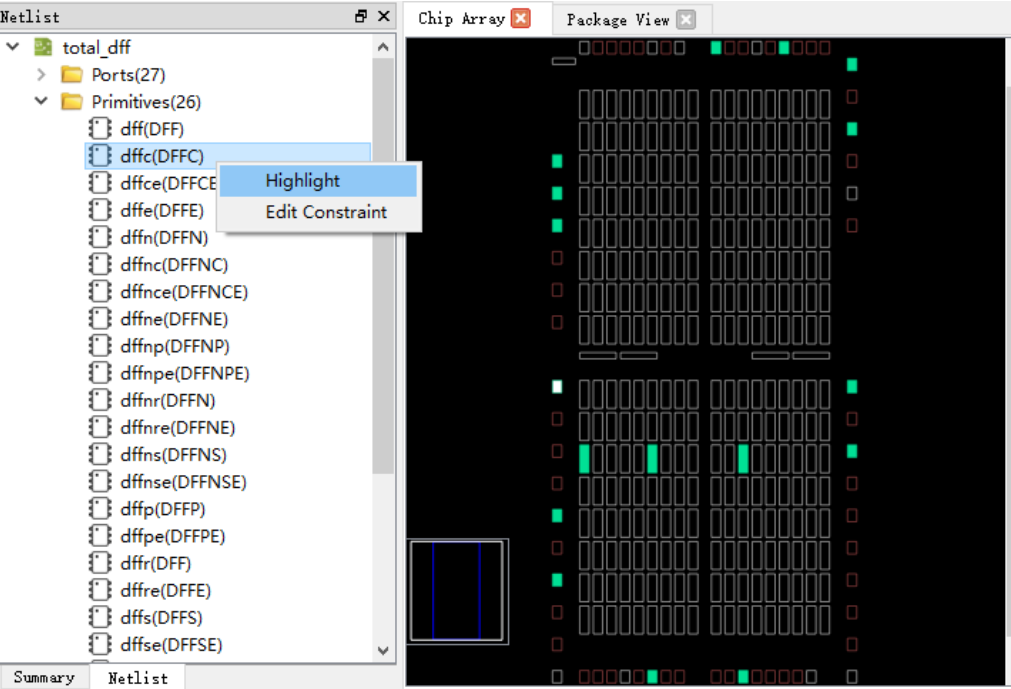


图 3-45 右键高亮



3.2.5 Constraint 编辑窗口

Constraint 编辑窗口包含 “I/O Constraints”、“Primitive Constraints”、“Group Constraints” 等 8 个编辑窗口，用于显示各约束的详细信息，并提供约束编辑功能和位置拖拽功能，各窗口分别介绍如下：

I/O Constraints

I/O Constraints 是对设计的 port 进行管脚约束。I/O 约束窗口如图 3-46 所示，各功能如下：

- 显示用户设计中所有 IO Port 的属性及约束信息，如 Port 的 Direction、Bank、IO Type、Pull Mode 等；
- 提供约束位置、属性等编辑功能；
- 可通过拖拽、双击等方式改变约束信息。

注！

- I/O 的位置可以通过拖拽的方式进行设置，也可以双击输入；
- 在拖拽 IO 过程中会显示所拖拽的 IO 名称；
- 将 IO 拖拽至 Chip Array 窗口中时，可放置的位置变亮，不可放置的位置颜色亮度不变；
- 将 IO 拖拽至 Package View 窗口中时，可放置位置亮度不变，不可放置位置变暗；
- 设置完成后，在 Chip Array 窗口中约束的位置变为浅蓝色高亮，在 Package View 窗口中约束的位置变为橙色高亮。

窗口提供右键菜单功能，详情如下：

- Unplace: 取消放置
- Reset Properties: 复位 Port 属性设置
- Highlight: 高亮显示约束位置
- IO Type: 设置电平标准
- Drive: 设置驱动电压
- Pull Mode: 设置上拉模式
- PCI Clamp: 设置 PCI 协议的开关
- Hysteresis: 设置迟滞量
- Open Drain: 设置开漏电路的开关
- Vref: 设置外部参考电压
- Single Resistor: 设置单端电阻的开关
- Diff Resistor: 设置差分电阻的开关
- Bank Vccio: 设置 BANK 电压

注！

右键菜单支持用户批量修改 Port 属性的功能，用户可选择多个 Port，若多个 Port 有相同的属性值可配置，则通过右键菜单可统一进行配置，详细属性设置标准参考 [DS102, GW2A 系列 FPGA 产品数据手册](#)。

图 3-46 I/O 约束窗口

| I/O Constraints | | | | | | | | | | | |
|-----------------|---------|-----------|-----------|----------|------|-----------|-----------|-------|-----------|-----------|-----|
| | Port | Direction | Diff Pair | Location | Bank | Exclusive | IO Type | Drive | Pull Mode | PCI Clamp | Hys |
| 2 | cin | input | | | | False | LVC MOS18 | N/A | UP | N/A | N |
| 3 | clk | input | | | | False | LVC MOS18 | N/A | UP | N/A | N |
| 4 | clko | output | | | | False | LVC MOS18 | 8 | UP | N/A | I |
| 5 | cout | output | | | | False | LVC MOS18 | 8 | UP | N/A | I |
| 6 | data[0] | input | | | | False | LVC MOS18 | N/A | UP | N/A | N |
| 7 | data[1] | input | | | | False | LVC MOS18 | N/A | UP | N/A | N |

Primitive Constraints

Primitive Constraints 是约束设计中原语的位置，原语约束窗口如图 3-47 所示，功能如下：

- 用于显示当前所有 Primitive 约束的名称、类型、位置以及 Exclusive 信息；
- 提供编辑功能，该窗口提供右键菜单功能，用于提供高亮显示约束位置、删除和添加约束的功能。

注！

- 可通过拖拽的方式或双击输入的方式修改位置信息；
- 可通过双击设置 Exclusive 属性；
- 在 Primitive 约束位置进行手动输入时，会对位置进行语法检查及合法性检查，错误提示对话框如图 3-11 和图 3-12 所示。

图 3-47 原语约束窗口

| Primitive Constraints | | | |
|-----------------------|-----------|------|-----------|
| | Primitive | Type | Locations |
| 1 | ALU_check | ALU | P4C12 |

Group Constraints

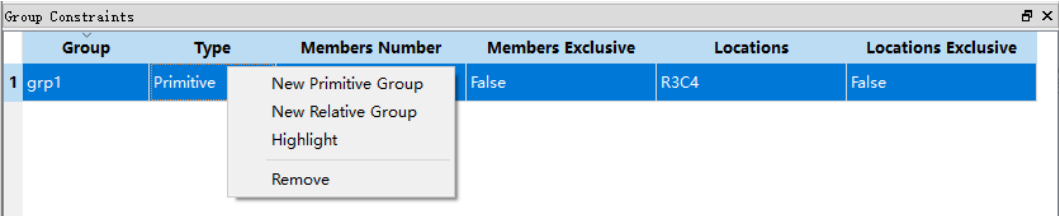
Group Constraints 是对设计中的 I/O 和部分原语进行组约束，组约束窗口如图 3-48 所示，功能如下：

- 用于显示当前所有组约束的名称、类型、包含的 Primitive 个数、位置以及 Exclusive 信息，包含 Primitive 和 Relative 两种 Group 的显示；如图

3-10 和图 3-14 所示，双击对应的 **Group** 条目，打开对话框，可实现约束信息的编辑修改功能；

- 该窗口提供右键菜单功能，用于提供高亮显示约束位置、删除和添加约束的功能。

图 3-48 组约束窗口



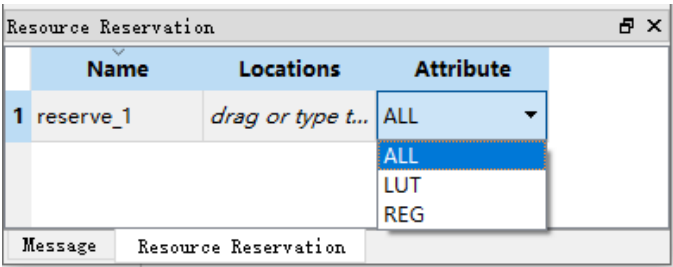
Resource Reservation

Resource Reservation 是对当前封装中的可用资源进行预留约束，预留约束窗口如图 3-49 所示，功能如下：

- 用于显示当前所有预留约束的位置信息；
- 该窗口提供右键菜单功能，用于提供高亮显示约束位置、删除、添加约束的功能；
- **Name** 属性用于区分各资源预留约束，用户不能进行修改。

注！
可通过拖拽或双击输入修改位置信息。

图 3-49 预留约束窗口



Clock Net Constraints

Clock Net Constraints 是对设计中的 **net** 进行全局时钟分配约束，时钟分配约束窗口如图 3-50 所示，功能如下：

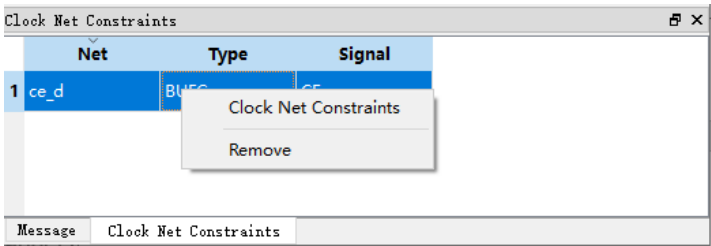
- 用于显示当前所有时钟分配约束的相关信息；
- 该窗口提供右键菜单功能，用于提供添加、删除时钟分配约束的功能。

注！

- 双击约束可进行编辑修改；
- **CLOCK Net** 约束无位置信息，不支持拖拽功能；

- 新建全局时钟分配约束的窗口如图 3-16 所示。

图 3-50 时钟约束窗口



GCLK Primitive Constraints

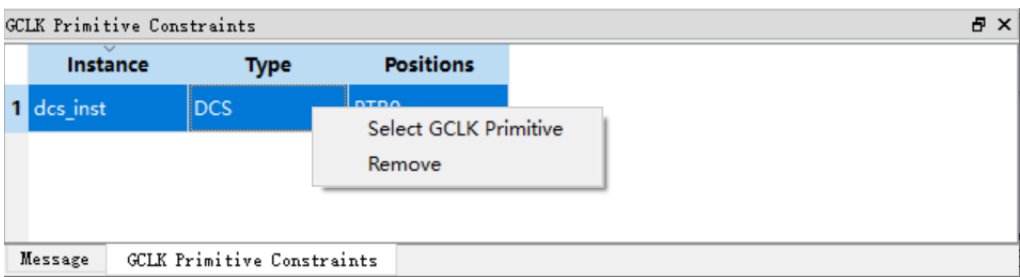
GCLK Primitive Constraints 是对全局时钟原语进行约束，全局时钟原语约束窗口如图 3-51 所示，功能如下：

- 用于显示所有的全局时钟约束，包括 Instance 名称、类型以及位置；
- 窗口支持右键菜单功能，用于添加新的全局时钟原语约束和删除已有约束。

注！

新建全局时钟约束的窗口如图 3-17 所示。

图 3-51 全局时钟原语约束窗口

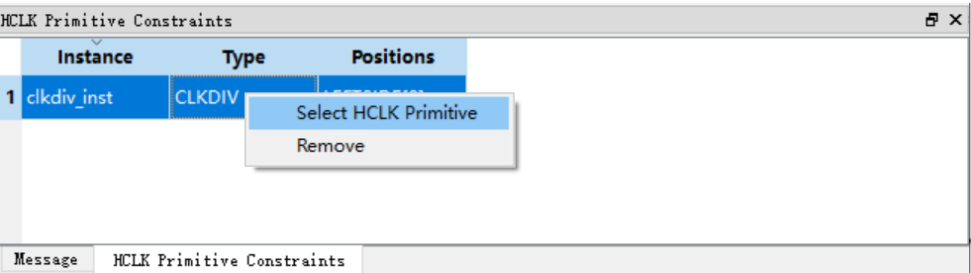


HCLK Primitive Constraints

HCLK Primitive Constraints 是对设计中的高速时钟原语进行约束，高速时钟原语约束窗口如图 3-52 所示，功能如下：

- 用于显示针对高速时钟相关的 Instance 的位置约束，包括 Instance 名称、类型以及高速时钟位置；
- 窗口支持右键菜单功能，用于添加新的高速时钟原语约束和删除已有约束。新建高速时钟原语约束的窗口如图 3-19 所示。

图 3-52 高速时钟原语约束窗口



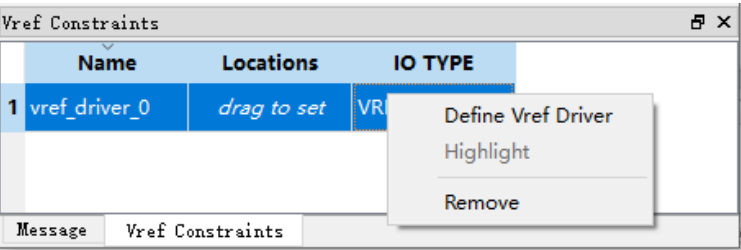
Vref Constraints

Vref Constrains 是约束所在 bank 的外部参考电压，Vref 约束窗口如图 3-53 所示，功能如下：

- 用于显示用户自定义的 Vref Driver 信息，用户可自定义 Vref 的名称、位置信息；
- 窗口支持右键菜单功能，用于高亮显示约束位置、添加、删除约束信息。

注！
位置信息只能通过拖拽的方式进行设置。

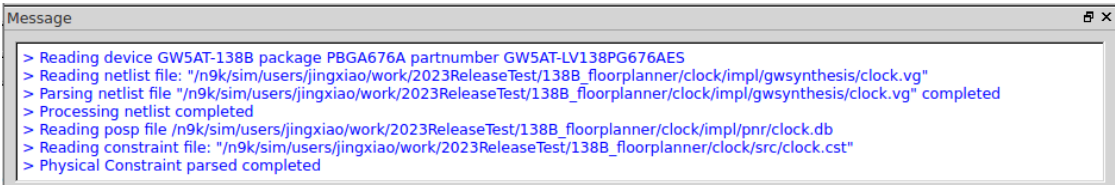
图 3-53 Vref 约束窗口



3.2.6 Message 窗口

Message 窗口如图 3-54 所示，窗口提供输出结果的显示。

图 3-54 Message 窗口



4 FloorPlanner 使用

FloorPlanner 可以新建和编辑约束，生成供布局布线流程使用的物理约束文件。

4.1 新建约束文件

FloorPlanner 可输出新建的物理约束文件，亦可输出修改后的物理约束文件，操作步骤如下所示：

1. 根据 3.1 启动所述，启动 FloorPlanner；
2. 单击“File > New”，打开“New”对话框，如图 4-1 所示；

注！

亦可通过以下两种方式打开“New”对话框：

- 使用快捷键 **Ctrl + N**；
 - 单击工具栏上的“New”图标。
3. 输入工程的网表文件，选择器件类型，单击“OK”。

图 4-1 新建约束文件

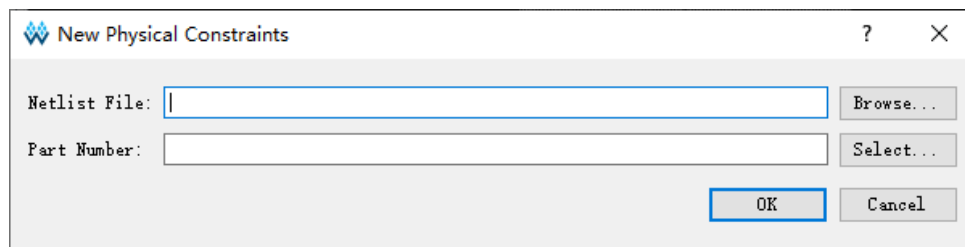
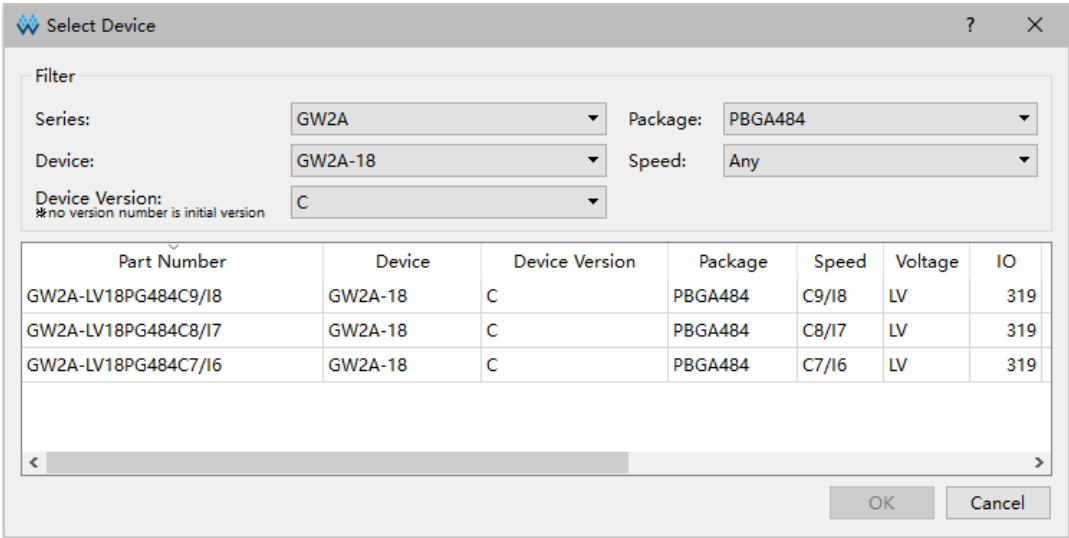


图 4-2 选择器件

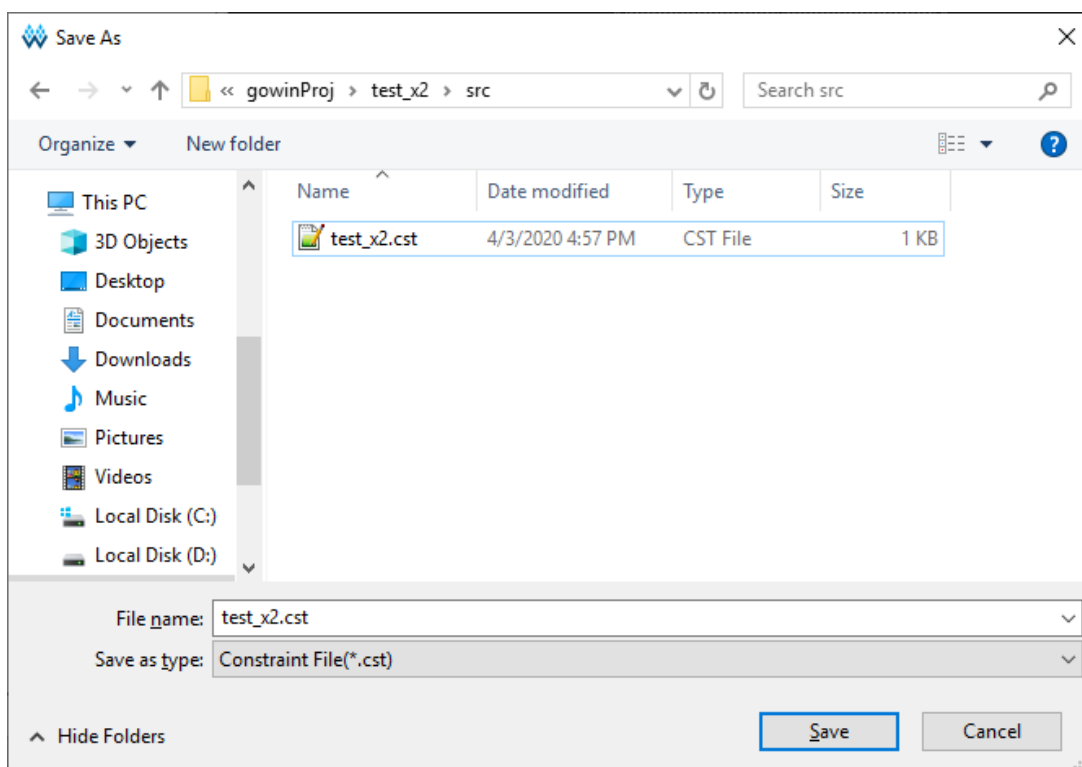


- 注！
- Select ...按钮用于选取器件、封装信息，支持高云半导体所有的 FPGA 器件，如图 4-2 所示；
 - 启动 FloorPlanner 采用 3.1 启动介绍中的第一种方式。

新建物理约束，在 FloorPlanner 主界面中可进行如下操作：

1. 用户通过拖拽等方式分配管脚位置；
2. 单击工具栏中的“Save”图标，即可输出约束文件；
3. 在弹出的“Save”对话框中，可修改文件名，如图 4-3 所示。

图 4-3 保存输出文件



4.2 编辑约束文件

FloorPlanner 支持 I/O 约束、原语约束、组约束、资源预留约束、全局时钟分配约束、全局时钟原语约束、高速时钟原语约束、参考电压约束等的创建。可通过 **Constraints** 菜单编辑生成约束，详情请参考 [3.2.1 菜单栏](#)。

注！

亦可通过其他方式创建约束，本节主要以拖拽的方式为例，介绍如何通过拖拽编辑生成约束。

4.2.1 编辑约束示例

以用户设计 counter.v 为例，演示如何编辑各类型约束：

```
module counter1(out, cout, data, load, cin, clk, ce, clko);  
output [7:0] out;  
output cout;  
output clko;  
input ce;  
input [7:0] data;  
input load, cin, clk;
```

```
reg [7:0] out;
always @(posedge clk)
begin
    if (load)
        out = data;
    else
        out = out + cin;
end
assign cout = &out & cin;
wire clkout;
CLKDIV clkdiv_inst (
    .CLKOUT(clkout),
    .HCLKIN(clk),
    .RESETN(1'b1),
    .CALIB(1'b0)
);
defparam clkdiv_inst.DIV_MODE = "2";
defparam clkdiv_inst.GSREN = "false";
DQCE dqce_inst (
    .CLKOUT(clko),
    .CLKIN(clkout),
    .CE(ce)
);
endmodule
```

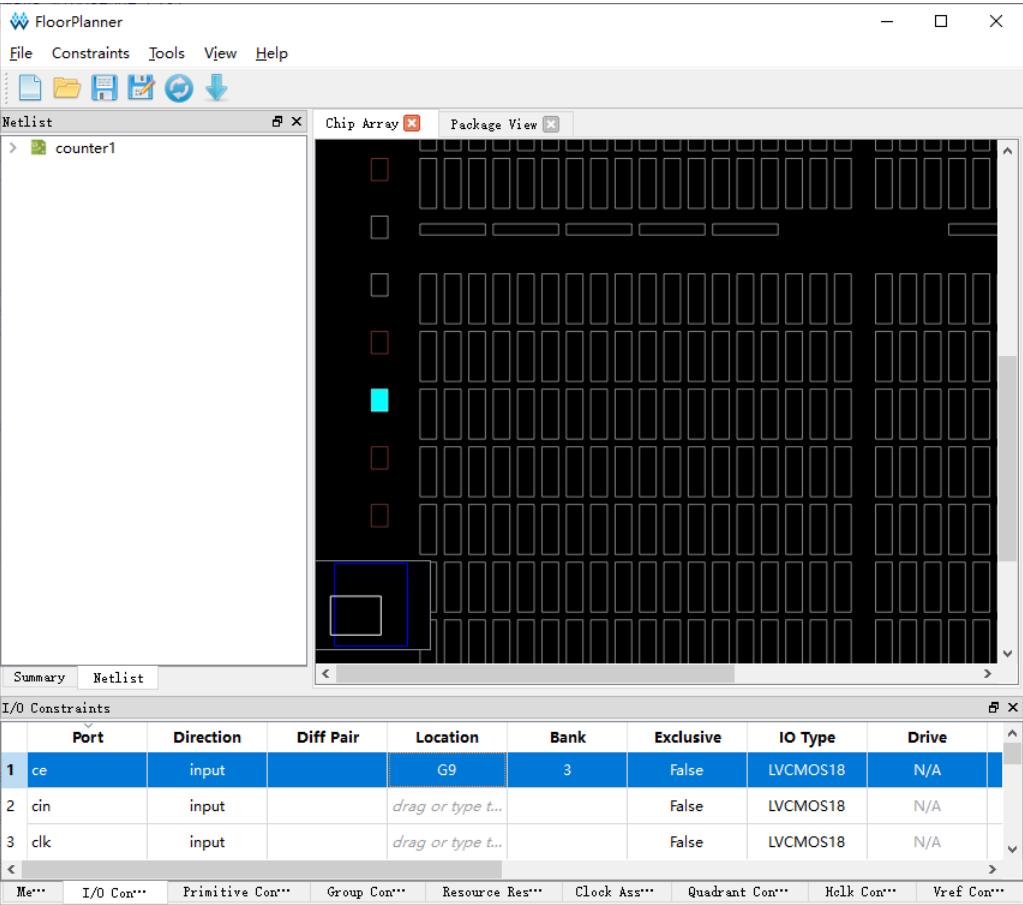
4.2.2 编辑 I/O 约束

拖至 Chip Array 创建 I/O Constraints，步骤如下：

1. 单击“IO Constraints”编辑窗口，放大 Chip Array 视图至宏单元模式；
2. 选中 Port “ce”拖拽至 Chip Array 窗口中的“G9”位置，如图 4-4 所示；

3. Port “ce” 的 Location 信息显示为 G9。

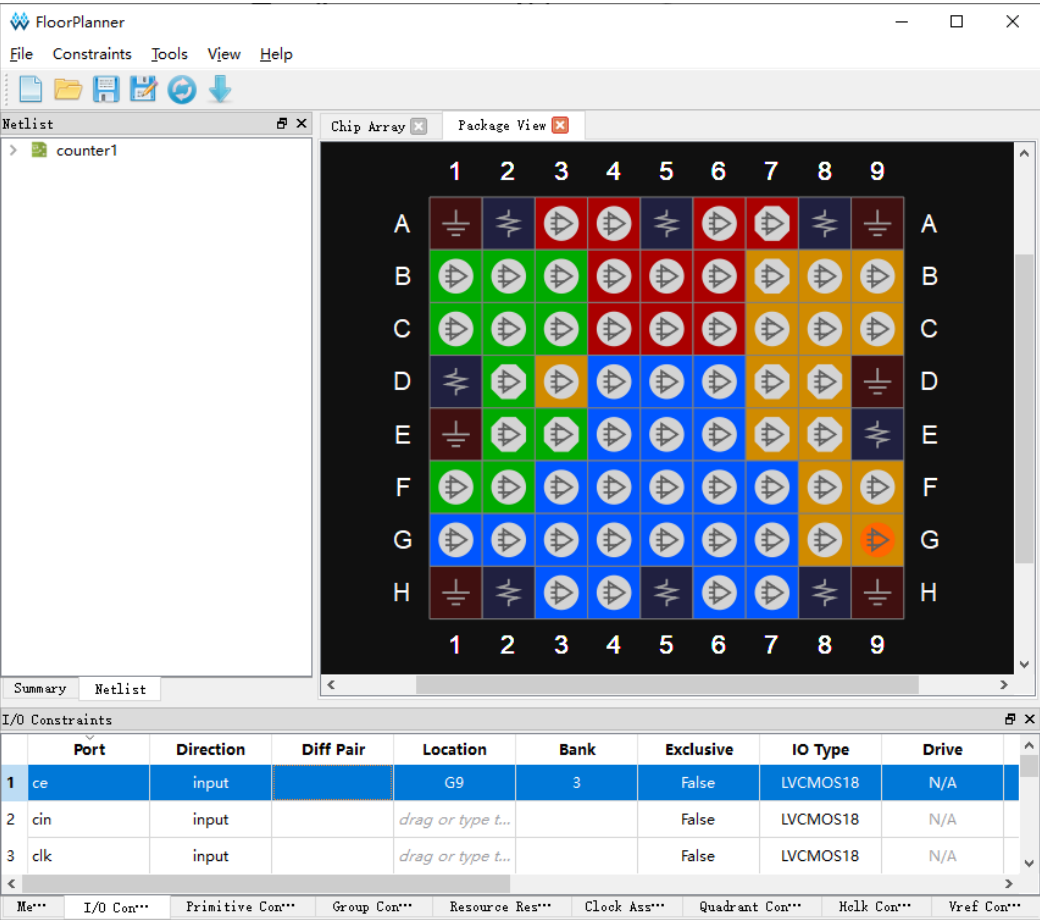
图 4-4 拖拽到 Chip Array 创建 I/O Constraints



拖至 Package View 创建 I/O Constraints，步骤如下：

1. 单击至 “IO Constraints” 编辑窗口；
2. 选中 Port “ce” 拖拽至 Package View 窗口中的 “G9” 位置，如图 4-5 所示；
3. Port “ce” 的 Location 信息显示为 G9。

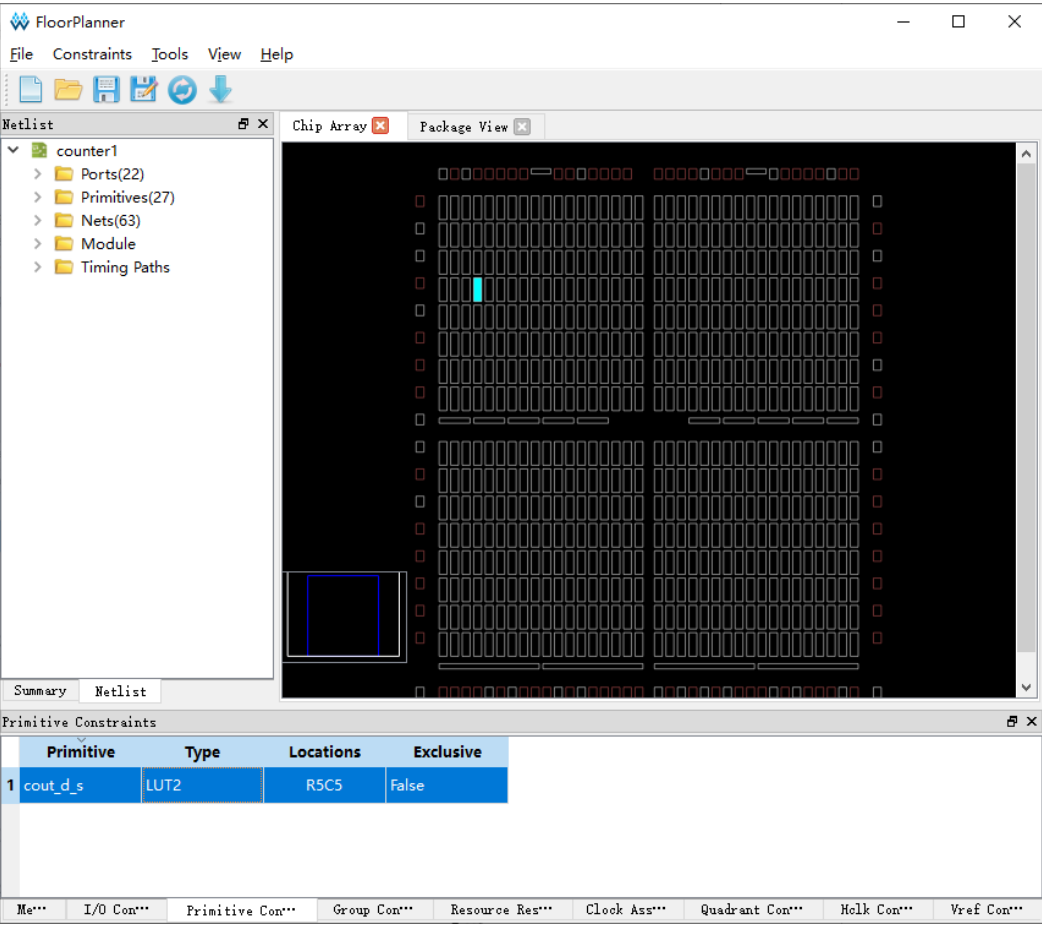
图 4-5 拖至 Package View 创建 I/O Constraints



4.2.3 编辑原语约束

- 1. 在“Primitive Constraints”编辑窗口，右击选择“Select Primitives”，弹出“Select Primitives”对话框后，选择 Primitive “cout_d_s”，单击“OK”；
- 2. 选中新创建的 Primitive 约束，拖拽至 Chip Array 窗口中的“R5C5”位置，如图 4-6 所示；
- 3. Primitive “cout_d_s” 的 Location 信息显示为 R5C5。

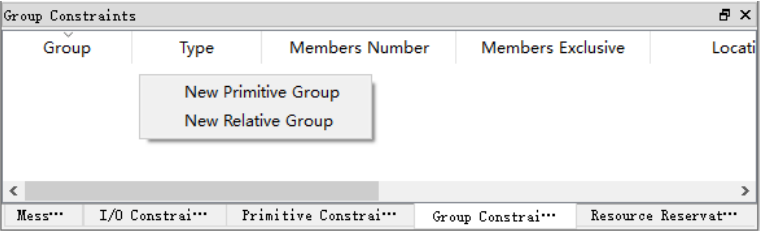
图 4-6 拖拽到 Chip Array 创建 Primitive Constraints



4.2.4 编辑组约束

如图 4-7 所示，在 Group Constraints 编辑器中，可通过右击选择菜单，创建 Primitive Group 和 Relative Group。

图 4-7 Group Constraints 编辑器右键菜单



编辑原语组约束

1. 在“Group Constraints”编辑窗口中，右击选择 “New Primitive Group”，弹出 “New Primitive Group” 对话框；
2. 输入 Group Name “grp1”，单击 “+” 弹出 “Select Primitives” 对话框；

3. 选取所要设置的 Primitive “n14_s0”、“n14_s1”，单击 “OK”，加入 Members 列表；
4. 在 Locations 输入所要约束的位置 “R9C7”，如图 4-8 所示；
5. 在 New Primitive Group 对话框中单击“OK”，可创建一条 Primitive Group Constraints，如图 4-9 所示。

图 4-8 创建 Primitive Group Constraints

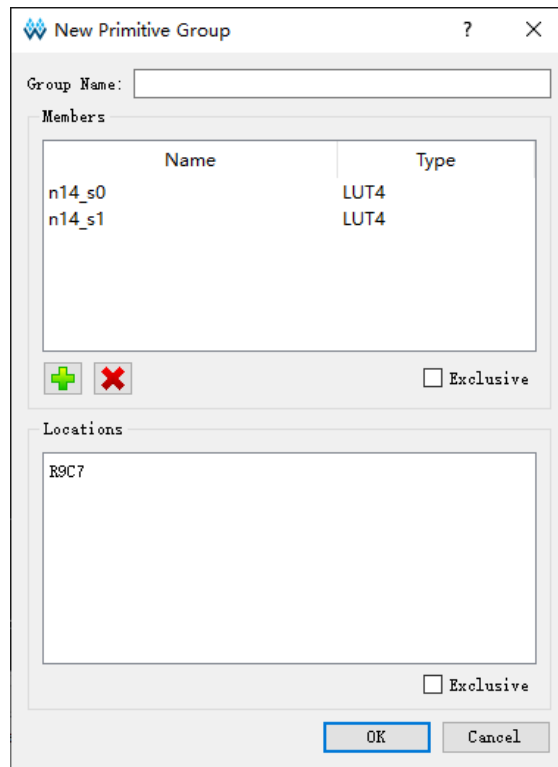
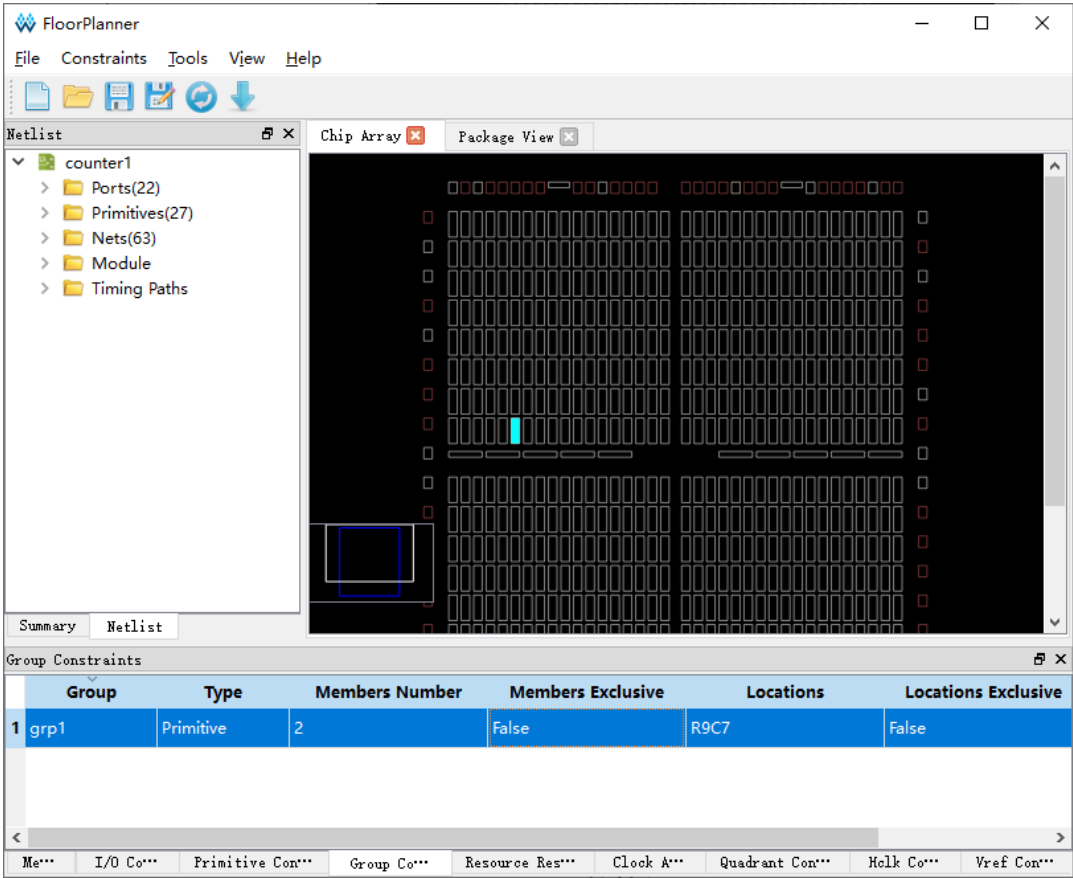


图 4-9 Primitive Group Constraints



注！
Primitive Group Constraints 的 Location 信息只能通过手动输入或者从 Chip Array 窗口中复制，不能通过拖拽实现。

编辑相对组约束

- 1. 在“Group Constraints”编辑窗口中，右击选择“New Relative Group”，弹出“New Relative Group”对话框；
- 2. 输入 Group 的名字“rel_grp”，单击“+”，弹出“Select Primitives”对话框；
- 3. 在 Select Primitives 对话框中选择所要设置的 Primitives “cout_d_s”、“n14_s0”，选择“OK”，添至 Member 列表中；
- 4. 为每个 Primitive 添加相对位置“R0C0”、“R4C5”，如图 4-10 所示；
- 5. 在 New Relative Group 对话框中，选择“OK”创建 Relative Group Constraints，如图 4-11 所示。

图 4-10 Relative Group Constraints 创建

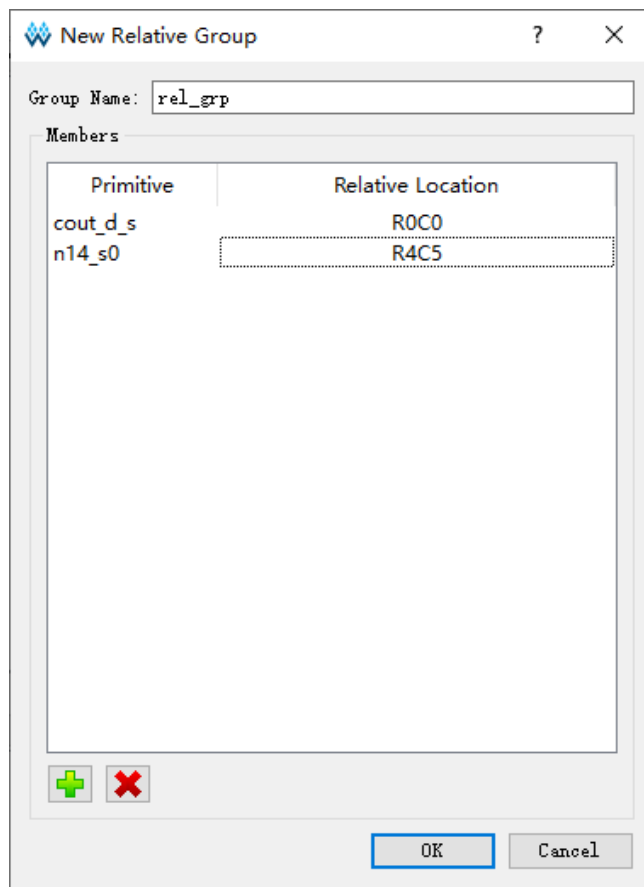
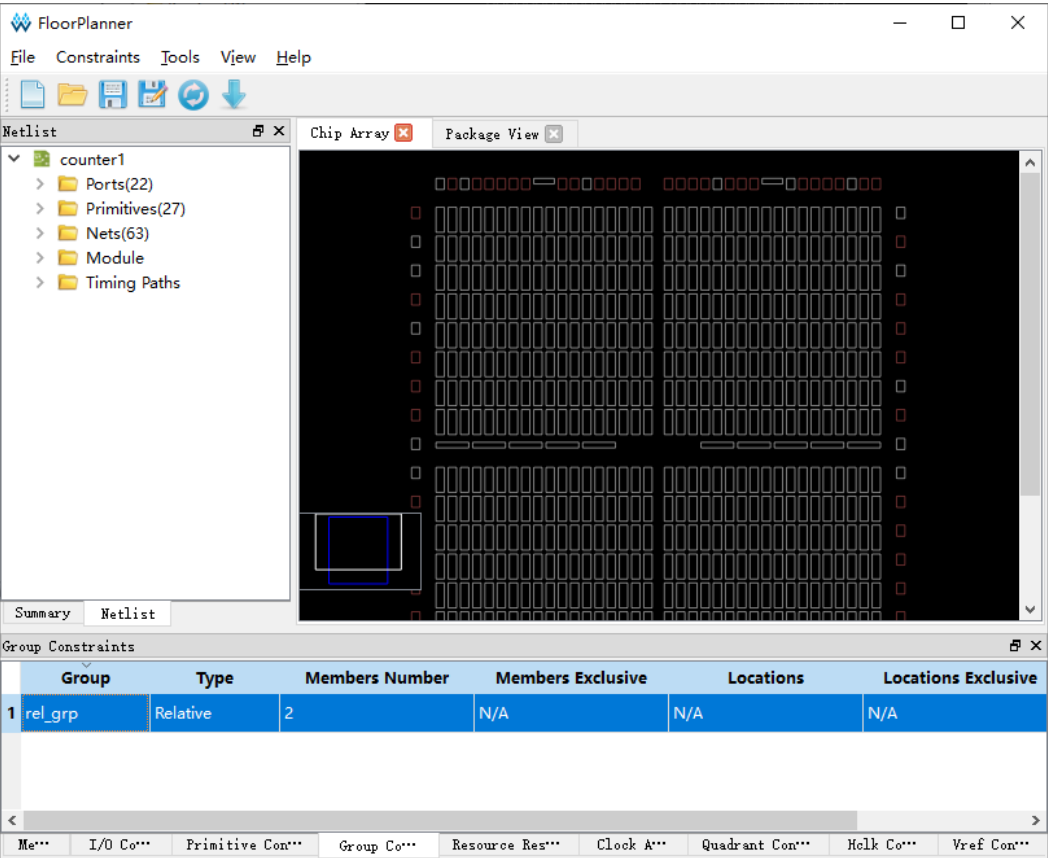


图 4-11 Relative Group Constraints



4.2.5 编辑资源预留约束

- 1. 在“Resource Reservation”编辑窗口中，进行右击选择“Reserve Resources”，在编辑器中添加 Resource Reservation 约束，如图 4-12 所示；
- 2. 选中新建的 Resource Reservation 约束拖拽到 Chip Array 窗口中想要进行预留约束的位置，图 4-13 所示拖拽至 BSRAM_R10[1]位置完成 Resource Reservation 约束。

图 4-12 创建 Resource Reservation 约束

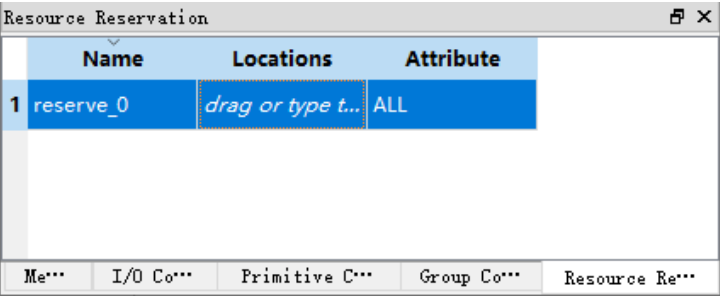
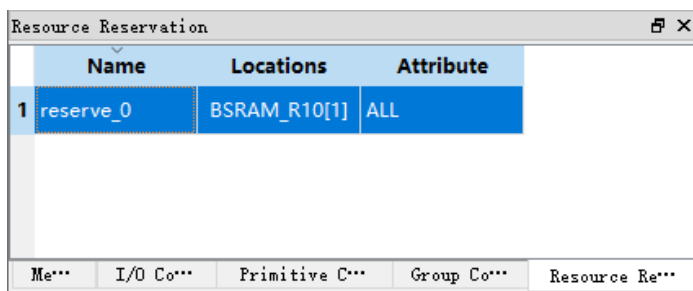


图 4-13 Resource Reservation



4.2.6 编辑全局时钟分配约束


1. 在 Clock Net Constraints 编辑窗口中，右击选择“Clock Net Constraints”，弹出 Clock Net Constraints 设置对话框；
2. 单击“”，弹出“Select Net”对话框，选择需要约束的 Net，在 Select Net 对话框中，单击“OK”添加 Net；
3. 设置时钟类型，在 Type 下拉列表中选择 Type 类型，设置 Signal 类型，如图 4-14 所示；
4. 设置完成后，单击“OK”，即会将该条约束添加至 Clock Net Constraints 编辑窗口中，如图 4-15 所示。

图 4-14 Clock Net Constraints 约束创建

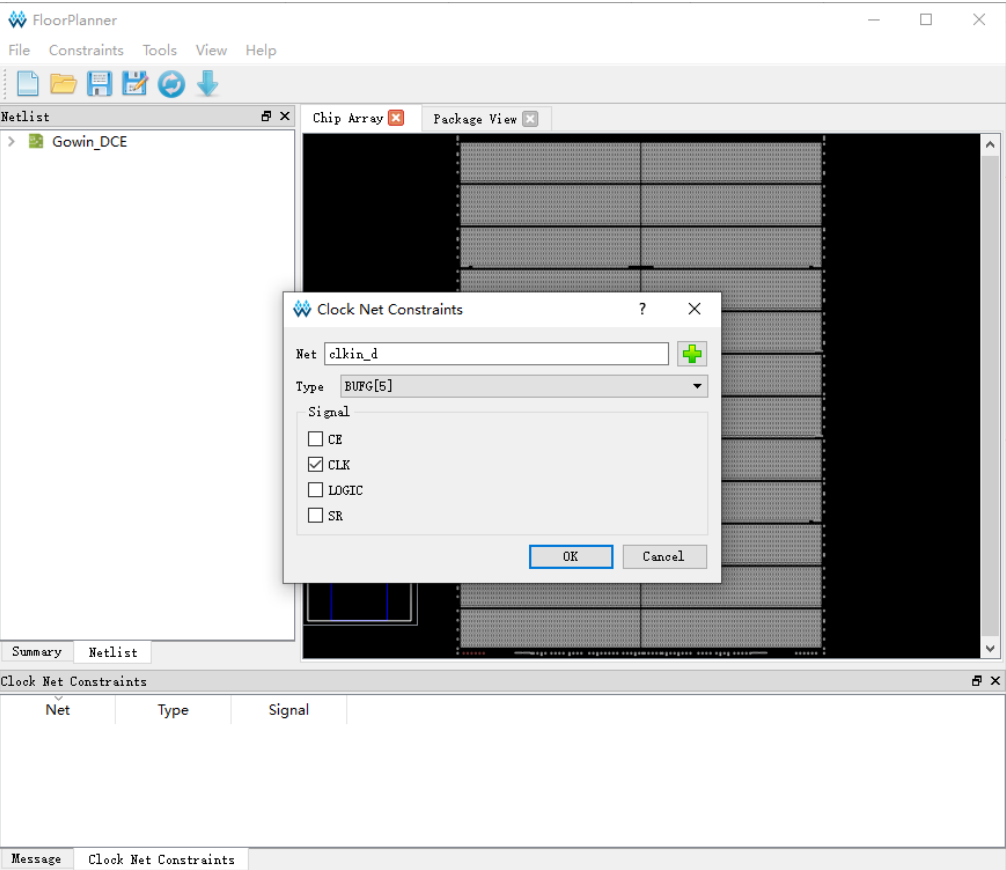


图 4-15 Clock Net Constraints 约束

| Clock Net Constraints | | |
|-----------------------|---------|--------|
| Net | Type | Signal |
| 1 clkin_d | BUFG[5] | CLK |

4.2.7 编辑全局时钟原语约束

GCLK Primitive Constraints 仅支持对 DCS 和 DQCE 的约束。

GCLK Primitive Constraints 的创建步骤如下：

- 1. 在 GCLK Primitive Constraints 编辑窗口中，进行右击选择“Select GCLK Primitive”，弹出 GCLK Primitive Constraints 对话框；
- 2. 单击“+”，弹出“GCLK”选择对话框，选取 Instance，在 GCLK Primitive 选择对话框中，单击“OK”，Instance 完成设置；

3. 在 GCLK Primitive Constraints 对话框的“Position”下方选取所要约束的全局时钟位置，如图 4-16 所示；
4. 在 GCLK Primitive Constraints 对话框中单击“OK”，即将该条约束添至 GCLK Primitive Constraints 编辑窗口，如图 4-17 所示。

图 4-16 GCLK Primitive Constraints 创建

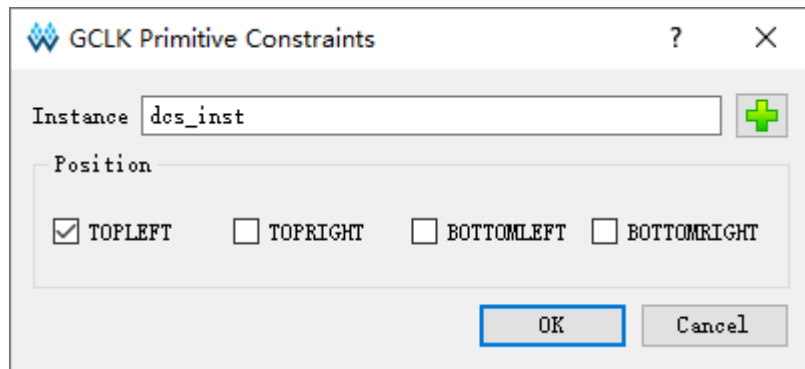
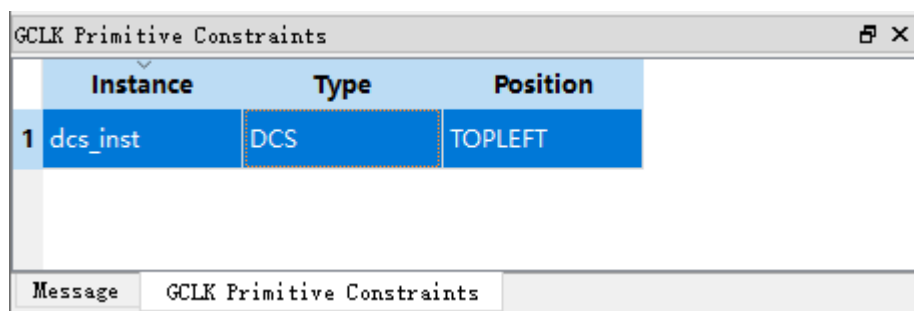


图 4-17 GCLK Primitive Constraints



4.2.8 编辑高速时钟原语约束

HCLK Primitive Constraints 仅对以下两种类型的 CLKDIV 和 DLLDL Instance 进行约束。

HCLK Primitive Constraints 的创建步骤如下：

1. 在 HCLK Primitive Constraints 编辑窗口中，进行右击选择“Select HCLK Primitive”，弹出 HCLK Primitive Constraints 对话框；
2. 单击“”按钮，弹出“HCLK Primitive”对话框，选取 Instance，在 HCLK Primitive 对话框中，单击“OK”，设置 Instance；
3. 在 HCLK Primitive Constraints 对话框中 Position 下方选取所要约束的高速时钟位置，如图 4-18 所示；
4. 单击 HCLK Primitive Constraints 对话框的“OK”，即可将该约束添至 HCLK Primitive Constraints 编辑器中，如图 4-19 所示。

图 4-18 HCLK Primitive Constraints 创建

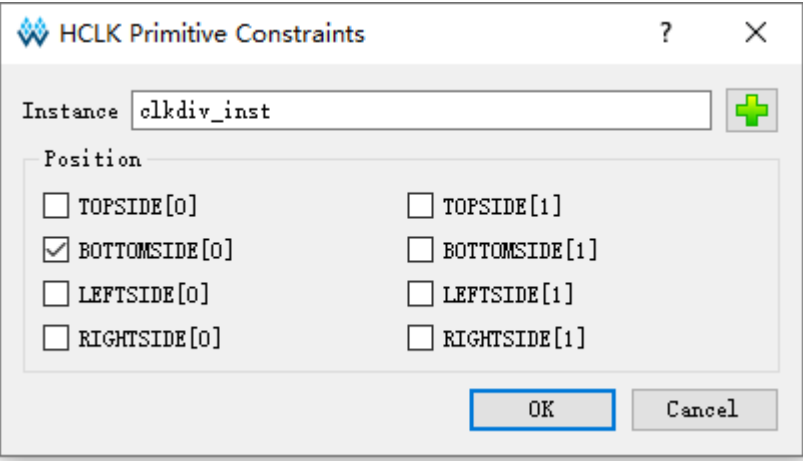
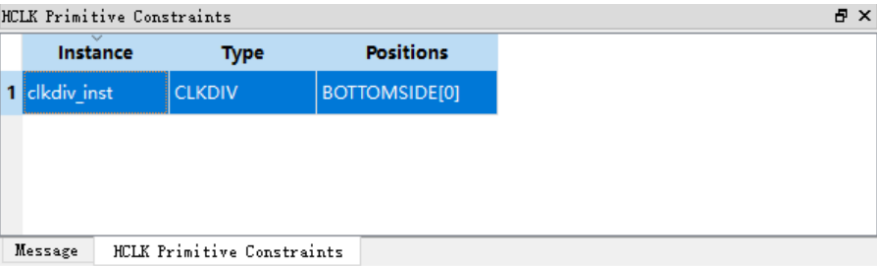


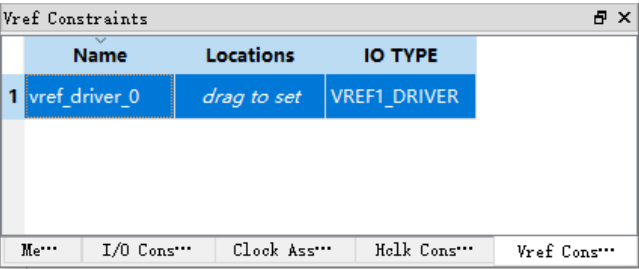
图 4-19 HCLK Primitive Constraints



4.2.9 编辑参考电压约束

- 拖至 Chip Array 窗口创建 Vref Constraints，步骤如下：
- 1. 在“Vref Constraints”编辑窗口中，进行右击选择“Define Vref Driver”，即可将该条 Vref Constraints 约束添至 Vref Constraints 编辑器中，如图 4-20 所示；
 - 2. 放大 Chip Array 视图至宏单元模式，选中 Vref Constraints 编辑窗口中新创建的 Vref Constraints，拖拽至 Chip Array 窗口中的 B7 位置，Vref Constraints 的 Location 信息显示为“B7”，如图 4-22 所示。

图 4-20 Vref Constraints 创建



可自定义 Vref 约束名，Vref 名字不允许重名，设置期间，系统会进行

检查，如名字重复，则提示用户，如图 4-21 所示。

图 4-21 Vref Constraints 名字重复

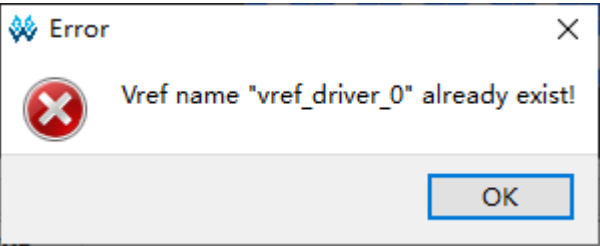
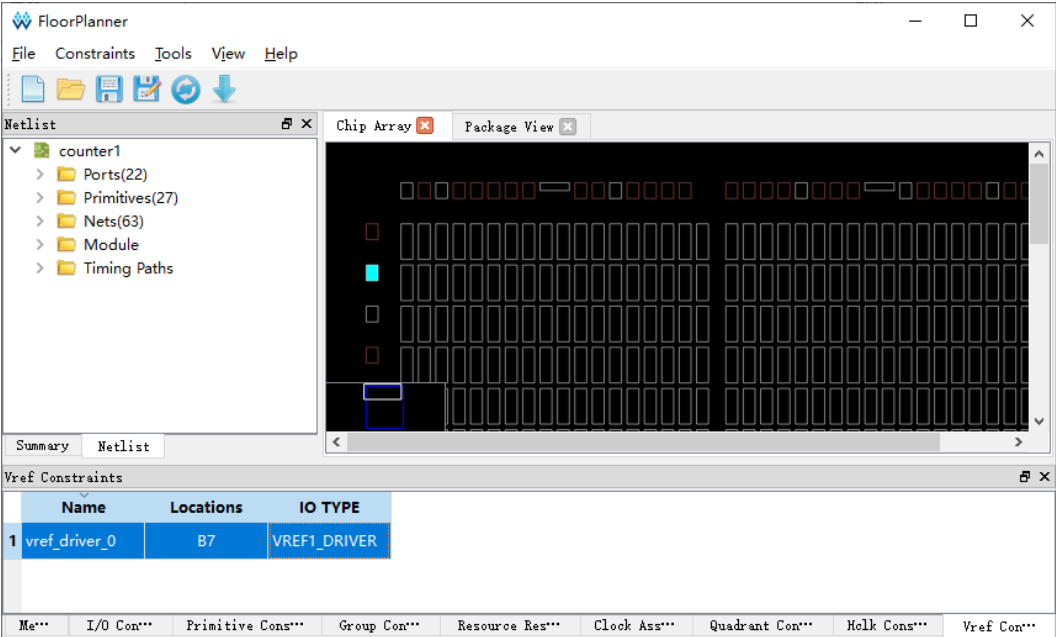


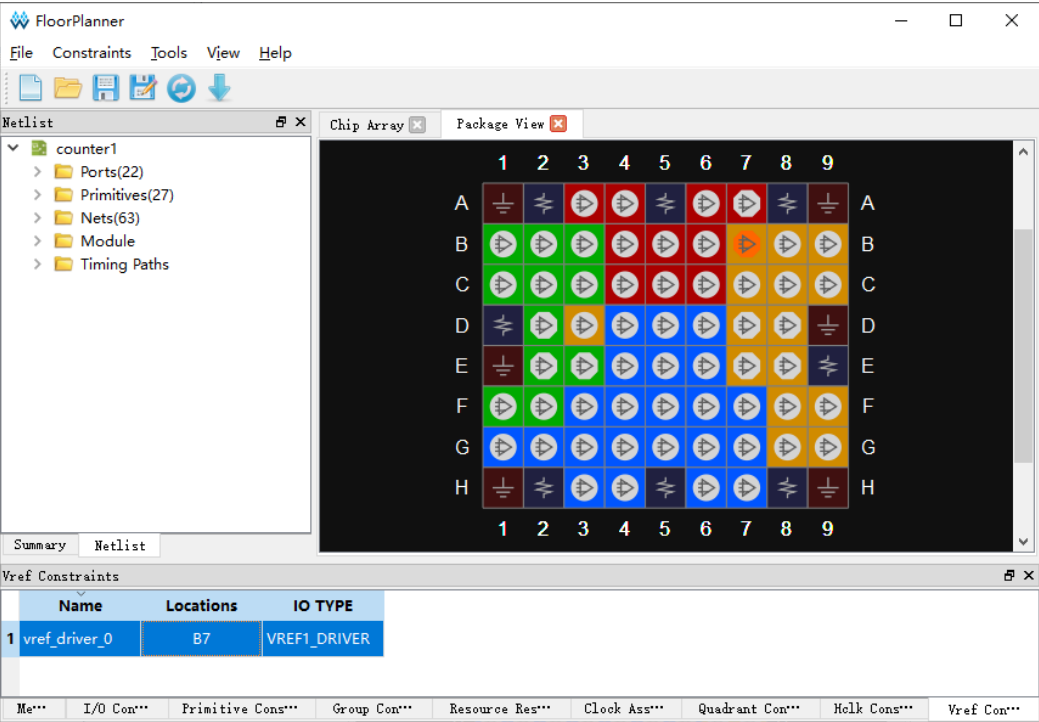
图 4-22 拖拽至 Chip Array 窗口生成 Vref Constraints Location 信息



拖至 Package View 创建 Vref Constraints，步骤如下：

1. 在“Vref Constraints”编辑窗口中，进行右击选择“Define Vref Driver”，即可将该条 Vref Constraints 约束添至 Vref Constraints 编辑器中，如图 4-20 所示；
2. 选中 Vref Constraints 编辑窗口中新创建的 Vref Constraints，拖拽至 Package View 窗口中的 B7 位置，Vref Constraints 的 Location 信息显示为“B7”，如图 4-23 所示。

图 4-23 拖拽至 Package View 窗口生成 Vref Constraints Location 信息



附录 A 物理约束语法规范

A.1 I/O 位置约束

I/O 位置约束可将 port、Buffer 约束到指定 IOB 位置处。

语法

```
IO_LOC "obj_name" obj_location [exclusive];
```

约束元素

obj_name

obj_name 可取 port、Buffer 的 name 作为 obj_name。

obj_location

obj_location 为 IOB 位置，如“A11”、“B12”等，若指定多个位置，位置之间需要用逗号分隔，如“A11,B2”。

exclusive

exclusive 为可选项，在约束位置之后，表明该约束语句中的 obj_location 仅可以放置 obj_name 指定的原语。

注！

当 obj_name 为 escaped name 格式（以反斜线开头，空格结尾）时，obj_name 两边需加上引号。

应用举例

示例 1

```
IO_LOC "io_1" A1;
```

```
// 对象 io_1 被约束在 pin A1 的位置。
```

示例 2

```
IO_LOC "io_1" A1, B14, A15;
```

// 对象 io_1 被约束在 pin A1、B14、A15 的位置，布局时将取三个位置之一进行布局。

示例 3

```
IO_LOC "io_2" A1 exclusive;
```

// 对象 io_2 被约束在 pin A1 处，且 A1 位置仅可以被 io_2 所占用。

示例 4

```
IO_LOC "io_2" A1, B14, A15 exclusive;
```

// 对象 io_2 被约束在 pin A1、B14、A15 处，且 A1、B14、A15 三个位置仅可以被 io_2 占用。

A.2 I/O 属性约束

I/O 属性约束，用于设定 I/O 的各种属性值。如 port 的电平标准 IO_TYPE，上拉/下拉模式 PULL_MODE，驱动能力 DRIVE 等，详细属性设置标准请参考 [DS102, GW2A 系列 FPGA 产品数据手册](#)。

语法

```
IO_PORT "port_name" attribute = attribute_value;
```

一个约束语句中可设定多个属性，各个属性之间使用空格分隔。

约束元素

需要属性约束的 I/O 的 name，attribute 和 attribute value。

应用举例

示例 1

```
IO_PORT "port_1" IO_TYPE = LVTTTL33;
```

// 设置 port_1 的 IO_TYPE 为 LVTTTL33。

示例 2

```
IO_PORT "port_2" IO_TYPE = LVTTTL33 PULL_MODE =KEEPER;
```

// 设置 port_2 的 IO_TYPE 为 LVTTTL33，PULL_MODE 属性值为 KEEPER。

示例 3

```
IO_PORT "port_3" IO_TYPE=LVDS25;
```

// port_3 连接的 Buffer 为 IBUF 时，通过该约束，可将该 IBUF 转化为 TLVDS_IBUF。

A.3 原语位置约束

Primitive Constraints 用于将 instance 布局到指定的 GRID 处，可以通过 Primitive Constraints 对 LUT、BSRAM、SSRAM、DSP、PLL、DQS 等 instance 进行约束。

语法

```
INS_LOC "obj_name" obj_location [exclusive];
```

约束元素

obj_name

约束对象的 instance 的 name。

obj_location

obj_location 包含如下几类：

LUT 约束位置

- 单一位置信息，指定到 LUT，如：RxCy[0-3][A-B]；
- 位置信息为一个范围，指定多行或多列。
 - 包含多个 CLS 或 LUT：“RxCy”、“RxCy[0-3]”；
 - 指定多行：“R[x:y]Cm”、“R[x:y]Cm[0-3]”、“R[x:y]Cm[0-3][A-B]”；
 - 指定多列：“RxC[m:n]”、“RxC[m:n][0-3]”、“RxC[m:n][0-3][A-B]”；
 - 指定多行多列：“R[x:y]C[m:n]”、“R[x:y]C[m:n][0-3]”、“R[x:y]C[m:n][0-3][A-B]”。

注！

在一条约束语句中，可包含多个 ins_location，使用“,”分隔。

PLL 约束位置

对于 PLL 约束位置信息书写格式为“PLL_L”或“PLL_R”，若左边可放置多个 PLL，可设为“PLL_L[0]”、“PLL_L[1]”...，若右边可放置多个 PLL，可设为“PLL_R[0]”、“PLL_R[1]”...

BSRAM 约束位置

BSRAM 约束位置信息为“BSRAM_R10[0]”（第 10 行第一个 BSRAM），“BSRAM_R10[1]”...

DSP 约束位置

DSP 约束位置格式为 “DSP_R19[0]” (第 19 行第一个 DSP Block), “DSP_R19[1]” ... 若需指定具体 macro, 可标记为: DSP_R19[0][A]或 DSP_R19[0][B]。

exclusive

关键字 “exclusive” 为可选项, 在约束位置之后, 表明该约束语句中的 obj_location 仅可以放置 obj_name 指定的 instance。

应用举例

示例 1

```
INS_LOC "lut_1" R2C3, R5C10[0][A];
```

// lut_1 被约束在 R2C3 位置和 R5C10 的第 1 个 CLS 的第 1 个 LUT 的位置。

示例 2

```
INS_LOC "ins_2" R5C6[2] exclusive;
```

// ins_2 被约束在 R5C6 的第 3 个 CLS 的位置, 且该位置仅可以放置该 instance。

示例 3

```
INS_LOC "ins_3" R[2:6]C2;
```

// ins_3 被约束在行坐标第二行到第六行, 列坐标第二列的区域位置。

示例 4

```
INS_LOC "ins_4" R[2:4]C[2:6] exclusive;
```

// ins_4 被约束在行坐标为第二行到第四行, 列坐标为第二列到第六列之间的区域位置, 且该区域位置仅能被该 instance 所占用。

示例 5

```
INS_LOC "ins_5" R[2:4]C[2:6][1];
```

// ins_5 被约束在行坐标第二行到第四行, 列坐标第二列到第六列之间的区域位置的任意一个 GRID 的第 2 个 CLS 中。

示例 6

```
INS_LOC "reg_name" B14;
```

// 通过对 REGISTER、IOLOGIC 的 INS_LOC 约束, 约束其到 IOB 的位置 B14。

示例 7

```
INS_LOC "pll_name" PLL_L;
```

```
// 通过对 PLL 的 INS_LOC 约束，约束其位置 PLL left。
```

示例 8

```
INS_LOC "bsram_name" BSRAM_R10[2];
```

```
// 通过对 BSRAM 的 INS_LOC 约束，约束其位置第 10 行的第 3 个 BSRAM 位置处。
```

示例 9

```
INS_LOC "dsp_name" DSP_R19[2];
```

```
// 通过对 DSP 的 INS_LOC 约束，约束其位置第 19 行第 3 个 DSP Block。
```

一个 LUT4 的位置可以放置一个 LUT1、LUT2、LUT3、LUT4，LUT5 需要占用两个 LUT4 的位置（一个 CLS），LUT6 需要占用 4 个 LUT4 的位置（两个 CLS），LUT7 需要占用 4 个 CLS 的位置（一个 GRID），LUT8 需要占用 8 个 CLS（两个 GRID）。故对于不同 Instance 类型的约束，其约束位置的最小单元也不相同，对于 BSRAM、SSRAM、DSP（每个 DSP 单元有两个 MACRO）等也是如此，如下示例：

示例 10

LUT4 单元约束：

```
INS_LOC "lut4_name" R5C15[1][A];
```

```
// 将 lut4_name 约束到 R5C15 的第 2 个 CLS 的第 1 个 LUT 处。
```

示例 11

CLS 单元约束：

```
INS_LOC "lut5_name" R5C15[3];
```

```
// 将 lut5_name 约束到 R5C15 的第 4 个 CLS 处。
```

示例 12

CLS 单元约束：

```
INS_LOC "lut6_name" R5C15[0];
```

```
// 将 lut6_name 约束到 R5C15 的第 1 个 CLS 处（将占用 CLS[0]和 CLS[1]）。
```

示例 13

GRID 单元约束:

```
INS_LOC "lut7_name" R5C15;// 将 lut7_name 约束到 R5C15 处,LUT7
占用一个 GRID。
```

示例 14

GRID 单元约束:

```
INS_LOC "lut8_name" R5C15;
```

```
// 将 lut8_name 约束到 R5C15 处, lut8_name 将占用 R5C15
和 R5C16 两个 GRID。
```

示例 15

DSP MACRO 单元约束:

```
INS_LOC "mult_name" DSP_R19[1][A];
```

```
// 将 mult_name 约束到第 19 行第 2 个 DSP 的第一个 macro 中。
```

A.4 组约束

Group Constraints 包括 Primitive Group Constraints 和 Relative Group Constraints, 如下所述。

A.4.1 原语组约束

Primitive Group 约束用于定义一个组约束, 组是包含各类 Instance 对象的集合。通过 Primitive Group 约束, 可将 Instance 如 LUT、DFF、BSRAM、SSRAM、DSP、PLL、DQS 等, 或 Buffer、IOLOGIC 等添加到一个组中, 并可通过约束该组的位置实现对该组中所有的对象的位置约束。

语法

GROUP 的定义:

```
GROUP group_name = { "obj_names" } [exclusive];
```

添加 Instance 到组中:

```
GROUP group_name += { "obj_names" } [exclusive];
```

约束组的位置:

```
GRP_LOC group_name group_location[exclusive];
```

注!

当 group_name 为 escaped name 格式 (以反斜线开头, 空格结尾) 时, group_name 两边需加上引号。

约束元素

group_name

定义一个 name 作为该组的 name。

obj_name

obj_name 用于将指定的 Instance 对象添加到组中。

group_location

指定该 group 的约束位置，group_location 可取 IOB、GRID、BSRAM、DSP、PLL 的位置。

exclusive

关键字“exclusive”为可选项，在组定义语句或位置约束语句之后；

一个对象可以被多个组包含，但在组定义语句后添加“exclusive”关键字，表示该组内的对象仅可被该组所包含；

在位置约束语句之后使用“exclusive”，表示该约束位置仅可被该组内的对象所占用。

应用举例

示例 1

```
GROUP group_1 = { "ins_1" "ins_2" "ins_3" "ins_4" };
```

// 创建一个名为 group_1 的组，添加对象 ins_1、ins_2、ins_3、ins_4 到该组中。

示例 2

```
GROUP group_2 = { "ins_5" "ins_6" "ins_7" } exclusive;
```

// 创建一个名为 group_2 的组，对象 ins_5、ins_6、ins_7 属于且仅可属于该组。

示例 3

```
GROUP group_1 += { "io_1" "io_2"};
```

// 添加 io_1、io_2 到组 group_1 中。

示例 4

```
GRP_LOC group_1 R3C4, A14, B4;
```

// 组 group_1 中的对象可布局在 R3C4、A14、B4 位置处。

示例 5

```
GRP_LOC group_2 R[2:3]C[2:4] exclusive;
```

// 组 group_2 中的 Instance 对象可布局在区域 R[2:3]C[2:4]的范围内，且该范围仅可布局 group_2 中的 Instance 对象。

实例 6

```
GRP_LOC group_3 PLL_L, BSRAM_R10[0], DSP_R19[0];
```

// 组 group_3 中的对象可布局在 PLL_L, BSRAM_R10[0], DSP_R19[0] 位置处。

A.4.2 相对组约束

通过 Relative Group Constraints，可实现对 instance 如 LUT、REG、MUX 对象的相对位置约束。

语法

定义 Relative 约束的组：

```
REL_GROUP group_name = { "obj_names" };
```

添加 instance 对象到已定义的组中：

```
REL_GROUP group_name += { "obj_names" };
```

对组中的 instance 进行相对位置约束：

```
INS_RLOC "obj_name" relative_location;
```

约束元素

obj_name

约束对象的名称。

relative_location

行列相对位置信息描述。

应用举例

```
REL_GROUP grp_1 = { "ins_1" "ins_2" "ins_3" "ins_4" };
```

```
INS_RLOC "ins_1" R0C0;
```

```
INS_RLOC "ins_2" R2C3;
```

```
INS_RLOC "ins_3" R3C5;
```

// 定义一个名为 grp_1 的组约束，并添加 ins_1、ins_2、ins_3、ins_4 到 grp_1 中。以 ins_1 为相对位置原点 R0C0，ins_2 约束到相对 ins_1

的 R2C3 处，ins_3 约束到相对 ins_1 的 R3C5 处。

A.5 资源预留约束

通过 Resource Reservation 约束，可保留指定的位置或区域以避免在布局中使用。

语法

```
LOC_RESERVE location [ res_obj ];
```

应用举例

示例 1

```
LOC_RESERVE R2C3[0][A] -LUT;
```

```
LOC_RESERVE R2C3[0][A] -REG;
```

示例 2

```
LOC_RESERVE IOR3, IOR6, R2C3, R3C4;
```

示例 3

```
LOC_RESERVE R[2:5]C[3:6], R3C[8:9];
```

// 以上示例中约束的位置信息将会在布局阶段被保留。

A.6 参考电压约束

芯片支持外部参考电压输入，对整个 BANK 有效。Vref Constraints 约束可用于对外部参考电压的输入管脚的名称和位置进行约束。

注！

- 可设置外部参考电压的输入管脚位置必须有 IOLOGIC 资源；
- Vref Constraints 和 Port 属性约束联合使用才有效。当 input 或 inout 类型的单端 Port，IO Type 为 SSTL/HSTL 时 Vref 属性可设置为创建的 Vref Constraints，表示该 Port 的参考电压使用 Vref Constraints 位置输入的外部参考电压。

语法

```
USE_VREF_DRIVER vref_name [location];
```

约束元素

vref_name

自定义的 VREF pin name

location

芯片中任意含有 IOLOGIC 资源的 I/O 位置可作为 VREF pin 约束的 location。

应用举例

示例 1

```
USE_VREF_DRIVER vref_pin;
```

```
IO_PORT "port_1" IO_TYPE = SSTL25_I VREF=vref_pin;
```

```
IO_PORT "port_2" IO_TYPE = SSTL25_I VREF=vref_pin;
```

// 定义一个名为“vref_pin”的 VREF pin, 设置 port_1 与 port_2 的 VREF 属性为 vref_pin。

示例 2

```
USE_VREF_DRIVER vref_pin E16;
```

```
IO_LOC "port_1" C16;
```

```
IO_PORT "port_1" IO_TYPE = SSTL25_I VREF=vref_pin;
```

// 定义一个名为“vref_pin”的 VREF pin, 将其约束到 E16, 设置 port_1 的 VREF 属性为 vref_pin, 并将其约束到 C16, port_1 所约束位置需与 E16 在同一 bank 上。

A.7 全局时钟原语约束

GCLK Primitive Constraints 用于将 DCS、DQCE 等需要进行布局的全局时钟对象约束到指定的位置。

语法

```
INS_LOC "obj_name" position;
```

约束元素

obj_name

约束对象的名称。

position

小蜜蜂®家族: GW1N-9、GW1NR-9、GW1N-9C、GW1NR-9C 可约束“TOPLEFT”, “TOPRIGHT”, “BOTTOMLEFT”, “BOTTOMRIGHT” 4 个位置, 其他器件只能约束“LEFT”, “RIGHT” 2 个位置;

晨熙®家族: 可约束“TOPLEFT”, “TOPRIGHT”, “BOTTOMLEFT”, “BOTTOMRIGHT” 4 个位置。

应用举例

```
INS_LOC "dcs_name" LEFT;  
// 约束 DCS 对象 dcs_name 至 LEFT 位置。
```

A.8 全局时钟分配约束

Clock Net Constraints 是对于设计中特定 net 到全局时钟线或不绕时钟线的约束。可通过该约束,实现对特定 signal_type (CLK、CE、SR、LOGIC) 的 net 进行全局时钟线布线约束。

- BUFG[0-7]表示约束 net 绕 PCLK 资源;
- BUFS 表示约束 net 绕 SCLK 资源;
- LOCAL_CLOCK 表示约束 net 不绕时钟线。

CLK 信号为连接时钟引脚的信号, CE 信号为连接时钟使能引脚的信号, SR 信号为连接 SET、RESET、CLEAR、PRESET 引脚的信号, LOGIC 为连接逻辑输入引脚的信号。

语法

```
CLOCK_LOC "net_name" global_clocks = signal_type ;
```

约束元素

net_name

net 的名字。

global_clocks

BUFG[0-7]: 绕具体一个 PCLK 资源;

BUFG: 绕 PCLK 资源;

BUFS: 绕 SCLK 资源;

LOCAL_CLOCK: 不绕时钟线。

signal_type

CLK: signal_type 为时钟引脚的 net;

CE: signal_type 为时钟使能引脚的 net;

SR: signal_type 为 SET、RESET、CLEAR、PRESET 的 net;

LOGIC: signal_type 为以上 signal_type 之外的 net。

指定多个 signal_type , 可使用 “|” 符号进行分隔。

注！

若 `global_clocks` 选择的是 `LOCAL_CLOCK`，则 `signal_type` 不可选。

应用举例

示例 1

```
CLOCK_LOC "net" BUFG[0] = CLK;
```

// 约束 `CLOCK` 对象 “net” 的 `signal_type` 为时钟引脚的 `net` 绕到芯片的第 1 条 `PCLK` 资源上。

示例 2

```
CLOCK_LOC "net" BUFG = CLK|CE;
```

```
NET_LOC "net" BUFG = CLK|CE;
```

// 约束 `CLOCK` 对象 “net” 的 `signal_type` 为时钟引脚或时钟使能引脚的 `net` 绕到芯片的 `PCLK` 资源上。

示例 3

```
CLOCK_LOC "net" BUFS = CE;
```

```
NET_LOC "net" BUFS = CE;
```

// 约束 `CLOCK` 对象 “net” 的 `signal_type` 为时钟使能引脚的 `net` 绕到芯片的 `SCLK` 资源上。

示例 4

```
CLOCK_LOC "net" LOCAL_CLOCK;
```

// 约束 `CLOCK` 对象 “net” 不绕时钟线。

A.9 高速时钟原语约束

通过 `HCLK Primitive Constraints` 约束，可将 `CLKDIV`、`DLLDLY` 约束到相关高速时钟位置。`CLKDIV`、`DLLDLY` 约束位置与普通 `Instance` 对象约束位置不同，使用“`TOPSIDE`”，“`BOTTOMSIDE`”，“`LEFTSIDE`”，“`RIGHTSIDE`”表示约束位置的四边。

语法

```
INS_LOC "obj_name" position;
```

约束元素

obj_name

取 `CLKDIV`、`DLLDLY` 的 `instance name` 作为 `obj_name`。

position

“TOPSIDE[0-1]”

“BOTTOMSIDE[0-1]”

“LEFTSIDE[0-1]”

“RIGHTSIDE[0-1]”

应用举例

```
INS_LOC "clkdiv_name" TOPSIDE[0];  
// 将 clkdiv_name 约束到 TOPSIDE[0]位置。
```

A.10 其他约束

A.10.1 JTAGSEL_N net 约束

当使用 FPGA 内部逻辑控制 JTAGSEL_N 功能时，即在不断电第二次下载的时候，拉低 JTAGSEL_N 使得 JTAG 切换到配置下载功能，需要添加 JTAGSEL_N 的 net 物理约束，具体参考文档 [UG290, Gowin FPGA 产品编程配置手册](#)。

语法

```
NET_LOC "obj_name" V_JTAGSELN;
```

约束元素**obj_name**

内部逻辑的任一条可绕线的 net 作为 obj_name。

应用举例

```
NET_LOC "netname" V_JTAGSELN;  
// 将 netname 这条 net 来控制 JTAGSEL_N 的功能。
```

A.10.2 RECONFIG_N net 约束

当使用 FPGA 内部逻辑控制 RECONFIG_N 的功能时，即在不断电第二次下载的时候，拉低 RECONFIG_N 使得 FPGA 切换到配置复位功能，需要添加 RECONFIG_N 的 net 物理约束，具体参考文档 [UG290, Gowin FPGA 产品编程配置手册](#)。

语法

```
NET_LOC "obj_name" V_RECONFIGN;
```

约束元素

obj_name

内部逻辑的任一条可绕线的 **net** 作为 **obj_name**。

应用举例

示例

```
NET_LOC "netname" V_RECONFIGN;
```

// 将 **netname** 这条 **net** 来控制 RECONFIG_N 的功能。

