




Gowin 器件设计优化与分析 用户指南

SUG113-1.2,2022-06-07

版权所有 © 2022 广东高云半导体科技股份有限公司

GOWIN高云、、Gowin、GowinSynthesis、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、翻译本档内容的部分或全部，并不得以任何形式传播。

免责声明

本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
2016/12/20	1.0	初始版本。
2017/09/11	1.1	<ul style="list-style-type: none">● 更新Gowin器件对DSP模块的支持；● 修改时序报告内容描述。
2022/06/07	1.2	<ul style="list-style-type: none">● 删除适用产品；● 删除资源共享描述；● 删除完整条件和并行条件描述；● 更新3.1.1节设计流程约束描述；● 更新3.2节管脚分配描述；● 更新第4章时序收敛描述。

目录

目录	i
图目录	iii
表目录	iv
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 HDL 编码风格要求	3
2.1 HDL 编码总体要求	3
2.1.1 分层次编码综合要求	3
2.1.2 流水线编码设计要求	3
2.1.3 并行条件和优先级条件比较	4
2.1.4 避免生成锁存器	4
2.1.5 全局复位和局部复位	4
2.1.6 时钟使能	4
2.1.7 选择器	4
2.1.8 双向缓存	5
2.1.9 跨时钟域处理	5
2.1.10 存储器编码风格	5
2.1.11 DSP 编码风格	5
2.2 FSM 状态机编码要求	6
2.2.1 状态机总体描述	6
2.2.2 状态机状态编码方式	6
2.2.3 安全状态机的初始状态值和默认状态值	6
2.3 器件硬件特性	7
2.3.1 IO 逻辑	7
2.3.2 DSP	7

2.3.3 BSRAM	7
2.3.4 SSRAM	7
2.4 低功耗编码	7
2.5 避免综合和仿真不一致的编码要求	7
2.5.1 敏感列表	7
2.5.2 阻塞赋值和非阻塞赋值	8
2.5.3 信号扇出	8
3 设计规划	9
3.1 云源软件设计规划流程	9
3.1.1 设计约束流程	9
3.1.2 设计规划工具	10
3.2 管脚分配	10
3.2.1 管脚分配规则	10
3.2.2 管脚移植	11
3.3 时钟分配	11
3.3.1 时钟分配规则	11
3.3.2 时钟资源分配约束	11
3.4 逻辑资源约束	12
3.4.1 定义	12
3.4.2 约束语法	12
3.4.3 约束策略	12
3.4.4 特殊考虑	12
4 时序收敛	14
4.1 综合阶段时序收敛策略	14
4.2 布局布线时序收敛策略	15
4.2.1 时序约束	15
4.2.2 选项设置	15
4.3 解决时序问题	16
4.3.1 分析综合时序报告	17
4.3.2 分析布局布线报告	17
4.3.3 分析布局布线时序报告	18

图目录

图 2-1 云源软件 IP Core Generator 存储器	5
图 2-2 云源软件 IP Core Generator DSP	6
图 3-1 FloorPlanner 工具界面	10
图 3-2 Global Clock Constraints 界面.....	12
图 4-1 时序问题解决流程.....	16
图 4-2 综合报告时序分析最大频率	17
图 4-3 设计资源使用率	18
图 4-4 时钟资源使用率	18
图 4-5 时序最差路径.....	19

表目录

表 1-1 术语、缩略语	1
--------------------	---

1 关于本手册

1.1 手册内容

FPGA 设计优化主要分为编码风格、设计规划和时序收敛三大部分，这些因素直接决定了 FPGA 设计的成败。

编码风格直接影响 FPGA 设计的实现并最终影响设计的性能。尽管综合工具集成了一系列优化算法，但是用户仍有必要遵循一定的编码风格去引导综合工具在特定 FPGA 架构上达到最优结果。

设计规划用于指导用户把设计更好地适配到所选用的 FPGA 上并合理地平衡面积和速度的要求，目的在于把高云器件支持的所有功能和特性完美地呈现出来。

时序收敛可以保证用户设计满足某个特定的时序需求，这部分主要描述时序需求、时序约束以及时序优化的方法。

1.2 相关文档

通过登录高云半导体网站 www.gowinsemi.com.cn 可以下载、查看以下相关文档：

- [SUG100, Gowin 云源软件用户指南](#)
- [SUG940, Gowin 设计时序约束指南](#)
- [SUG935, Gowin 设计物理约束指南](#)

1.3 术语、缩略语

下表中列出了本手册中出现的相关术语、缩略语及相关释义。

表 1-1 术语、缩略语

术语、缩略语	全称	含义
FPGA	Field Programmable Gate Array	现场可编程门阵列
HDL	Hardware Description Language	硬件描述语言
FSM	Finite State Machine	有限状态机
DSP	Digital Signal Processor	数字信号处理器

术语、缩略语	全称	含义
BSRAM	Block Static Random Access Memory	块状静态随机存储器
SSRAM	Shadow Static Random Access Memory	分布式静态随机存储器
RTL	Register Transfer Level	寄存器传输级
CST	Physical Constraint	物理约束
SDC	Synopsys Design Constraint	时序约束
CFU	Configurable Function Unit	可配置功能单元
CLS	Configurable Logic Section	可配置逻辑块

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 HDL 编码风格要求

2.1 HDL 编码总体要求

2.1.1 分层次编码综合要求

对于复杂的系统级设计，分层次编码是很有必要的。分层次编码可以一次综合所有的模块，也可以按层次结构分别综合各个模块。当一次综合所有模块时，设计可以被综合成一个无层次结构的模块或者多个带层次结构的模块。

这两个策略各有优缺点，在复杂的系统级设计中分层次编码更具优势，原因在于分层次编码更容易定位问题，同时提高了模块的复用性并缩短了开发周期。下面是创建分层次编码结构的一些建议：

1. 顶层模块只用于调用子模块，控制逻辑尽量在各个子模块实现；
2. 管脚输入输出缓存需要在顶层例化；
3. 所有的输入、输出和双向管脚需要在顶层例化；
4. 只允许在顶层使用双向管脚的三态声明；
5. 尽可能保证模块所有输出信号全部使用寄存器，这样的好处在于：
 - 组合逻辑和寄存器在一个模块里，克服了综合不能跨模块优化的不足；
 - 把相关逻辑放在一个模块里，可以保证资源共享并优化关键路径；
 - 分割不相关逻辑到不同模块，则可以采用不同的优化策略，比如速度优先或者面积优先。

2.1.2 流水线编码设计要求

流水线设计通过重构多逻辑级数的数据路径并增加时钟周期分割逻辑级数达到提高设计性能的目的。流水线结构是提高数据路径速度的有效方法，但需要注意的是它会增加数据路径的传输时钟周期延时。

2.1.3 并行条件和优先级条件比较

`if-then-else` 结构生成优先级条件逻辑，而 `case` 结构生成并行条件逻辑，建议 `if` 语句嵌套小于 5 级。

`if-then-else` 结构可以包含多个不同的条件表达式，但是 `case` 结构只能包含一个表达式。在条件互斥的情况下，可以认为 `if-then-else` 结构和 `case` 结构是等效的。

2.1.4 避免生成锁存器

用户在 **FPGA** 设计中应该避免使用锁存器，综合工具通过组合逻辑环路实现锁存器，不可避免地造成资源浪费和性能下降，同时组合逻辑环路给静态时序分析带来了很大困难。

综合工具在组合逻辑条件表达式不完整情况下会生成锁存器，比如 `if-then-else` 结构中没有 `else` 或者 `case` 结构中没有 `default`。为了避免不必要的锁存器，在条件语句中需要把所有条件都遍历到。

如果客户使用 `case` 语句且不关心 `default` 条件的输出值，可增加 `/*synthesis full_case*/` 约束，避免锁存器的生成。

2.1.5 全局复位和局部复位

高云器件包含一个专用的全局复置位网络 **GSR**，直接连接到器件的内部逻辑，可用作异步/同步复位或异步/同步置位，**CFU** 和 **I/O** 中的寄存器均可以独立配置。全局复位资源并不占用普通绕线资源，并且遍布芯片各个角落。局部复位一般扇出较小，可以考虑使用普通绕线作为复位信号。

2.1.6 时钟使能

FPGA 设计中应该避免使用门控时钟，门控时钟会导致很多不可控的时序问题，比如不可预知的时钟毛刺。高云器件 **CLS** 结构中包含专用时钟使能信号，用户可以使用时钟使能达到门控时钟同样的功能而不必担心时序问题。以下是使用高云器件时钟使能的一些注意事项。

1. 只有寄存器支持时钟使能，锁存器不支持；
2. 同一个 **CFU** 里面的各个 **CLS** 共用一个时钟使能信号；
3. 所有寄存器时钟使能高有效。

2.1.7 选择器

CLS 里面的查找表可以灵活配置以实现二选一选择器、三选一选择器、四选一选择器或五选一选择器等等，用户还可以通过调用多个四输入查找表级联实现更大的选择器。

2.1.8 双向缓存

使用双向缓存可以节约管脚，控制输出使能达到节省功耗的目的。用户可以关闭综合工具自动管脚输入输出缓存插入选项，为特定的管脚例化特定的管脚输入输出缓存。

2.1.9 跨时钟域处理

当数据路径从一个时钟域跨越到另一个时钟域时，必须确保生成的亚稳态不会影响数据的建立时间和保持时间。对于单比特信号，建议使用两级或者三级寄存器结构消除亚稳态；对于多比特信号，建议使用异步 FIFO。

2.1.10 存储器编码风格

虽然随机存储器的行为级描述可定制且非常直观，但不同的编码风格可能会产生不同的综合结果。对于高云器件而言，建议使用高云半导体云源[®]软件 IP Core Generator 工具产生块存储器、分布式存储器以及 FIFO。

高云器件支持多种存储结构，包括双端口 RAM、单端口 RAM、伪双端口 RAM、只读 ROM、同步 FIFO 以及异步 FIFO，如图 2-1 所示。

图 2-1 云源软件 IP Core Generator 存储器

Name	Version
Hard Module	
Memory	
Block Memory	
DPB	1.0
SDPB	1.0
SP	1.0
pROM	1.0
Shadow Memory	
RAM16S	1.0
RAM16SDP	1.0
ROM16	1.0
Soft IP Core	
Memory Control	
FIFO	
FIFO	1.1
FIFO HS	1.0
FIFO SC	1.1
FIFO SC HS	1.1

2.1.11 DSP 编码风格

虽然 DSP 的行为级描述可定制且非常直观，但不同的编码风格可能会产生不同的综合结果。对于高云器件而言，建议使用云源软件 IP Core Generator 工具产生 DSP。

Gowin 器件支持多种 DSP 结构，包括 ALU54、MULT、MULTALU、MULTADDALU 以及 PADD，如图 2-2 所示。

图 2-2 云源软件 IP Core Generator DSP

Name	Version
Hard Module	
DSP	
ALU54	1.0
MULT	1.0
MULTADDALU	1.0
MULTALU	1.0
PADD	1.0

2.2 FSM 状态机编码要求

有限状态机在时钟边沿完成当前状态到下一个状态的跳转，下面主要讨论状态机编码的方法和策略。

2.2.1 状态机总体描述

状态机实现主要有三种方式，第一种是在一个进程中同时处理状态转移和状态输出；第二种是一个独立的进程处理状态转移，另一个进程处理状态转移规律和状态输出；第三种是三个独立的进程分别处理状态转移、状态转移规律及状态输出。建议用户使用第三种方式，它不仅便于阅读和修改，而且对于状态无寄存直接输出情况，可以保证不会引起额外的延时。

2.2.2 状态机状态编码方式

目前主要有二进制编码、独热编码以及格雷编码三种状态机编码方式。二进制编码及格雷编码使用较少的触发器，但使用较多的组合逻辑，而独热编码恰好相反。

独热编码的最大优势在于状态比较时仅仅需要比较一个比特位，从而一定程度上简化了译码逻辑。虽然在表示同等数量的状态时，独热编码占用较多的比特位，即使用较多的触发器，但这些触发器占用的面积可以和译码电路省下来的面积相抵消。

因此，对于状态数小于 5 的状态机，建议使用二进制编码和格雷编码；对于状态数大于 5 的状态机，建议使用独热编码。

2.2.3 安全状态机的初始状态值和默认状态值

安全状态机在上电后必须初始化进入一个初始有效状态，用户可以通过上电复位或全局复位操作进入一个初始有效状态。同时状态机必须有一个默认状态值来保证状态机不会进入非法状态，当代码中有状态组合没有被遍历，非法状态就会产生。

2.3 器件硬件特性

2.3.1 IO 逻辑

输入输出逻辑集成了串并转换、并串转换、延迟控制以及字节对齐等功能,主要用于高速数据传输场合。输入输出逻辑支持普通模式、单倍速率(SDR)模式、双倍速率(DDR)等多种模式。用户可以根据具体的设计需求调用高云器件的输入输出逻辑。

2.3.2 DSP

小蜜蜂[®]家族和晨熙[®]家族器件的 DSP 支持 9 bits、18 bits 两种宽度的有符号数及无符号数乘法、乘加、累加、54 bit 的算术/逻辑运算单元、桶形移位器,同时 DSP 支持输入输出寄存器的流水线和旁路功能。

2.3.3 BSRAM

小蜜蜂[®]家族和晨熙[®]家族器件的 BSRAM 可配置为最高 18 Kbits,数据位宽和深度都可以配置。每块 BSRAM 有两个端口,两个端口彼此独立,有独立的时钟、地址端口、数据端口和控制端口,但是两个端口共享存储空间。每块 BSRAM 可以配置 4 种操作模式:单端口模式、双端口模式、伪双端口模式和只读模式,同时 BSRAM 支持输出寄存器的流水线和旁路功能。

2.3.4 SSRAM

小蜜蜂[®]家族和晨熙[®]家族器件的 SSRAM 可配置成深度为 16,位宽为 1/2/4 bits 的单端口随机存储器,也可以配置成深度为 16,位宽为 1/2/4 bits 的伪双端口随机存储器,还可以配置 16 bits x 1 的只读随机存储器。

2.4 低功耗编码

以面积为优化目标可以降低逻辑和布线资源使用率,进而降低功耗,建议使用云源软件 IP Core Generator 去调用高云器件内部基本单元,以得到低功耗的实现,消除已知的毛刺以降低电源功耗,降低信号翻转率并在适当时刻进入休眠状态可降低系统功耗。

2.5 避免综合和仿真不一致的编码要求

特定编码风格会导致综合和仿真的结果不一致,原因在于仿真工具会忽略一些错误信息而这些错误信息会被综合工具检测出来,因此编码风格建议采用高云编码风格。

2.5.1 敏感列表

组合逻辑的信号敏感列表必须包含所有输入输出信号以确保综合和仿真结果一致。

2.5.2 阻塞赋值和非阻塞赋值

组合逻辑一般建议使用阻塞赋值；时序逻辑一般建议使用非阻塞赋值。

2.5.3 信号扇出

信号扇出控制用于保证综合后的扇出在一个合适的范围，综合工具通过复制电路来减少信号扇出。对特定信号使用综合属性 `syn_maxfan` 可以得到更好的时序收敛效果。

高云器件架构可以使用专用时钟网络处理大扇出的时钟信号，并使用专用全局复位网络处理大扇出的复位信号。但是综合工具倾向于对大扇出的逻辑进行电路复制，复制电路会带来一系列副作用，用户需要根据实际情况灵活使用综合属性 `syn_maxfan` 以避免生成过多的复制电路。

3 设计规划

FPGA 设计规划主要包括两个阶段：

1. 在设计前需要定义设计的功能和架构，挑选合适的 FPGA 器件并进行 RTL 编码；
2. 在设计过程中需要将设计以最优方式适配到所选的器件中，并充分利用 FPGA 器件中的资源。

这两个阶段是相互影响的，本章主要侧重介绍第二阶段，目的在于让用户使用云源软件将高云器件的功能和特性得到最大程度的发挥。

3.1 云源软件设计规划流程

不是所有设计都需要涉及设计规划，但是设计规划对大多数设计是有益的，尤其对一些资源使用率高，时序要求严格的大型设计。对于这些大型设计，设计规划可以减少潜在的布局布线问题和时序收敛问题，并提高设计的复用性和可移植性。

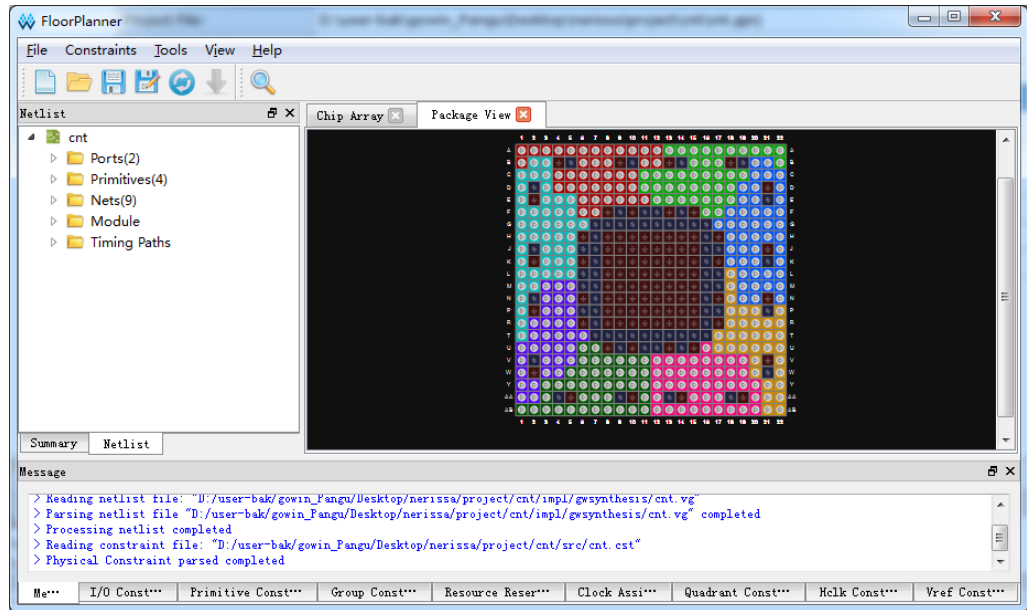
在云源软件中，设计规划开始于综合完成后布局布线开始前。CST 文件包含所有的设计规划约束用于指导后端布局布线。一旦设计规划改变，就意味着 CST 文件的改变，整合设计流程回到综合完成后布局布线开始前这个阶段。CST 文件约束及使用方法请参见 [SUG935, Gowin 设计物理约束用户指南](#)。

3.1.1 设计约束流程

云源软件约束流程允许客户对综合后的端口、网表、寄存器、实例进行约束。CST 文件是用户定义设计规划约束的一个主要入口。

后端布局布线软件通过读取 CST 文件获取用户的约束信息。如果 CST 文件出现语法错误或非法约束错误，后端布局布线软件会直接退出。CST 文件可以文本编辑，也可以使用 FloorPlanner 工具产生，如图 3-1 所示。具体使用参考文档 [SUG935, Gowin 设计物理约束用户指南](#)。

图 3-1 FloorPlanner 工具界面



3.1.2 设计规划工具

云源软件提供 FloorPlanner 工具进行设计规划，并提供如下主要功能：

- 预览设计中用户可以处理所有基本逻辑单元；
- 预览器件中支持的所有硬件资源；
- 分配特定的逻辑单元到特定的硬件资源；
- 支持拖拽操作实现器件约束；
- 支持约束信息合法性检查的功能；
- 支持手动调整优化时序路径。

3.2 管脚分配

管脚分配规划是一个定义管脚标准和位置的过程，它基于用户的实际设计和所选的器件，管脚分配涉及如下步骤：

1. 分配设计中的端口到特定管脚位置；
2. 定义管脚电平标准和管脚特征参数；
3. 检查分配的合法性；
4. 如有必要，根据 PCB 和电路图更改管脚分配和配置。

3.2.1 管脚分配规则

- 首先分配专用管脚，比如时钟输入、锁相环输入、DDR 等；
- 分配普通管脚到专门的 BANK 而不是特定的管脚，这样可以使后端布局布线工具计算得到一个最优的结果；
- 对于复用管脚，需要检查是否和器件编程模式相冲突。

3.2.2 管脚移植

对于封装相同的器件，用户有时候希望把设计从低密度器件移植到高密度器件用于未来扩展功能，或者从高密度器件移植到低密度器件以降低成本。但是管脚分配需要保持不变或作一些细微改动以避免 PCB 重新投板。云源软件提供了管脚移植特性。在 FloorPlanner 可以看到一些非适配的管脚，可以使用 LOC_RESERVE 屏蔽这些管脚。

3.3 时钟分配

3.3.1 时钟分配规则

在考虑时钟资源规划时，时钟主频和时钟扇出是需要考量的两个重要的因素，此外也需要考虑高云器件本身包含时钟资源的限制。

通常情况下，专用的时钟资源可以得到更好的时序结果，因为专用时钟资源的相对延时最小，节省下来的布线资源可以更好地应用于拥塞度比较高的场合。只有在极少数情况下用户需要将时钟分配到普通绕线上，普通绕线会导致时钟的相对延时很大，因此只有低速小扇出的应用场景才会考虑使用普通绕线资源作为时钟。下面介绍时钟资源分配的一般规则：

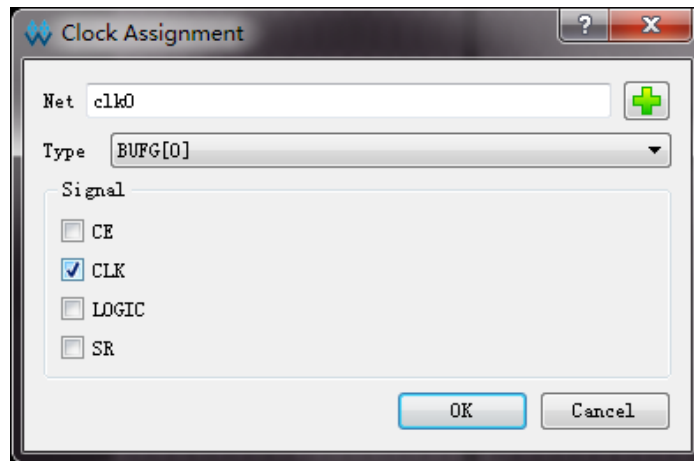
- 确定用户设计中时钟数目和每个时钟的扇出；
- 确定目标器件提供的时钟资源；
- 明确每个时钟的速度要求；
- 高速大扇出的时钟一般需要分配到全局时钟；
- 如果全局时钟数目小于设计中时钟数，需要考虑使用象限时钟；
- 对大吞吐量的高速接口使用高速时钟资源。

3.3.2 时钟资源分配约束

```
NET_LOC "xxx" BUFG0 = CLK | CE | SR | LOGIC
```

分配时钟信号到专门的全局时钟网络，BUFG0~BUFG7 分别代表高云器件支持的八个全局时钟。其中 CLK 表明约束对象是时钟；CE 表明约束对象是时钟使能；SR 表明约束对象是同步复位；LOGIC 表明约束对象是逻辑器件。如图 3-2 所示。

图 3-2 Global Clock Constraints 界面



3.4 逻辑资源约束

3.4.1 定义

逻辑约束是从逻辑角度划分设计基本单元，这会改变设计的物理布局 and 实现方式。逻辑约束主要通过锁定 FPGA 内部资源的位置来实现。

对于高云器件，逻辑约束对于改善设计性能有很大帮助，尤其有助于优化层次结构清晰的设计。逻辑约束提供了一种结合自动化和用户控制的编译方式，提高了设计的模块化和复用性，它是增量编译的基础。

3.4.2 约束语法

```
INS_LOC "cnt[5]" R2C2
```

约束特定的对象到特定的 CFU 位置。

```
GROUP hh = { "cnt_Z[1]" "cnt_Z[2]" "cnt_Z[3]" "cnt_Z[4]" "cnt_Z[5]"  
"cnt_Z[6]" "cnt_Z[7]" }
```

```
GRP_LOC hh R[3:6]C[4:6]
```

把特定对象分组并约束到特定的区域位置。

3.4.3 约束策略

- 逻辑模块划分应该基于层次结构；
- 逻辑模块划分应该基于关键路径；
- 逻辑模块划分应该基于大扇出的输入输出信号；
- 划分逻辑模块并对各个模块使用不同的优化策略。

3.4.4 特殊考虑

- 块存储器可以单独进行位置约束，不需要分组；
- 大逻辑模块进行位置约束需要指定起始位置和相对大小；

- 进位链和总线不需要分组；
- 互连逻辑不需要分组。

4 时序收敛

每个设计一般都需要运行在特定的速率上，FPGA 设计中主要涉及三类需求，即时序、带宽和延时。带宽和延时是互斥的，大带宽一般需要多级流水线，这会增加延时，反之延时小需要复杂的组合逻辑路径来实现，这会减少流水线级数，降低带宽。因此，在定义设计架构时首先需要把带宽和延时平衡好。

通常，时序收敛是决定设计能否满足特定速率需求的关键因素，这需要花费大量时间运用各种技术去达到时序收敛。本章主要讨论时序收敛，并讨论如何使设计达到时序收敛。

4.1 综合阶段时序收敛策略

当使用 GowinSynthesis[®]综合优化时，需要遵从一些总体的准则以更好地达到时序收敛的目的。

1. 遵循高云的 RTL 编码规范；
2. 运用合适的属性指令约束去指导综合优化；
3. 使用输入输出管脚寄存器改善管脚时序。使用输入管脚寄存器可以改善输入建立时间；使用输出管脚寄存器可以改善时钟到输出时间；
4. 使用 IO 延迟单元改善输入保持时间。使用输入管脚寄存器可能带来保持时间问题，原因在于数据通道的延时较短。因此可以在数据路径上增加 IODELAY 用于补偿输入保持时间，原语详细介绍参考 [UG289, Gowin 可编程通用管脚\(GPIO\)用户指南](#) 4.4 章节；
5. 使用 HCLK 调整 IO 输入寄存器数据和时钟之间的关系以同时满足建立时间和保持时间；
6. 使用专用的 GSR 资源，即在 RTL 设计中例化高云原语 GSR。如果设计中含有大扇出的复位或者置位信号，我们建议使用专用的 GSR 资源，这会降低布线拥塞，并提高布线效率；
7. 减少扇出以提高主频。对于关键路径选择性地使用 syn_maxfan 属性以减少扇出，减少扇出的代价是复制寄存器；
8. 使用独热状态机编码。对于高速设计，建议用户使用独热状态机编码，

但是会增加资源利用率和功耗；

9. 资源复用。资源复用会增加逻辑级数，并产生大延迟的路径。通常综合工具会对非关键路径进行资源复用，但是也有例外，用户需要检查关键路径以确保不会引入时序问题；
10. 检查综合报告，初步阅读综合后的时序信息，分析大扇出路径、关键路径以及大延迟路径；
11. 仔细阅读综合时序报告。由于综合时序报告没有包含布局布线信息，因此综合时序报告的结果往往比实际的结果要好，一般来说实际结果要差 1/3 到 1/2。

4.2 布局布线时序收敛策略

当使用云源软件布局布线时，需要遵从一些总体的准则以更好地达到时序收敛的目的。

1. 运用合适的物理约束和时序约束去指导布局布线；
2. 关注工具给出的一些告警和错误信息；
3. 选用合适的布局布线选项；
4. 避免欠约束，比如没有时钟约束、异步时钟分析等，可添加合理的时序约束；
5. 避免过约束，比如时钟频率比实际要求约束高、IO 的多周期路径等，可使用 **multicycle** 或者 **maxdelay** 放松时序；
6. 时钟使能尽量使用 **GCLK** 资源。时钟使能通常是一组驱动大量寄存器的大扇出信号，如果时钟使能使用普通绕线资源，有些时候会引入大延时，建议用户把时钟使能放在 **GCLK** 资源上。

4.2.1 时序约束

云源软件 **Timing Constraints Editor** 支持常用的时序约束。关于时序约束语法规则请参见 [SUG940, Gowin 设计时序约束用户指南](#) 附录 A 时序约束语法规则。如果没有添加时钟约束，软件会按照默认时钟频率来分析设计时序。

1. 小蜜蜂[®]家族默认时钟按照 50MHz 分析。
2. 晨熙[®]家族默认时钟按照 100MHz 分析。
3. **setup** 时序分析条件为高温低压。
4. **hold** 时序分析条件为低温高压。

4.2.2 选项设置

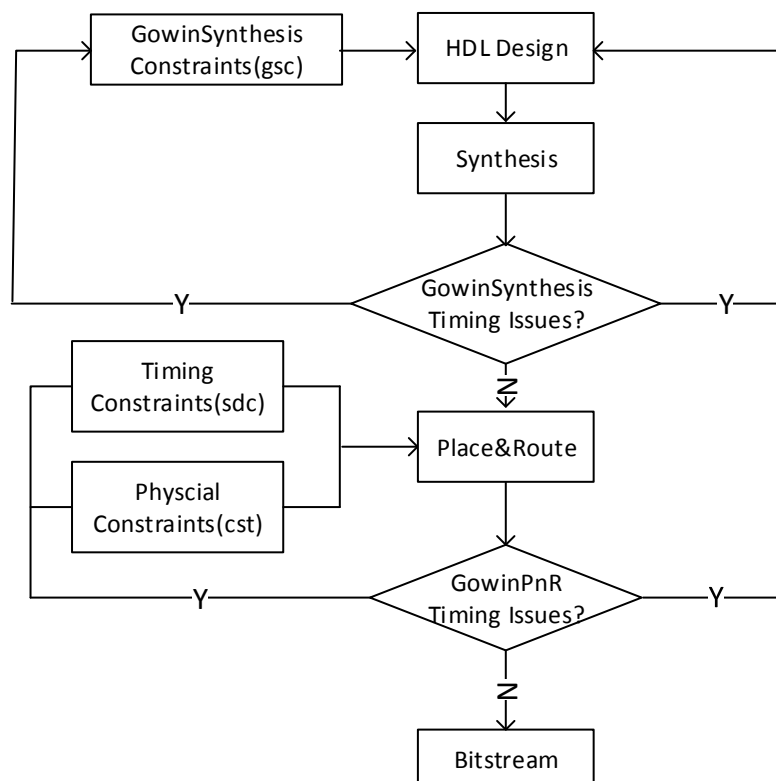
1. **Run Timing Driven**: 默认打开状态，时序驱动布线，如果时序要求不高，为了节省运行时间，可以关闭该选项；

2. **Place option:** 布局算法选项, 选项值有 0 和 1, 默认为 0;
 - 为 0 时, 使用默认布局算法;
 - 为 1 时, 在算法 0 的基础上, 牺牲一些时间效率来尝试找到更优的布局结果。
3. **Route Option:** 布线算法选项, 可选项有 0、1 和 2, 默认值为 0;
 - 为 0 时, 采用默认布线算法, 根据拥塞调整布线;
 - 为 1 时, 根据时序数据调整布线;
 - 为 2 时, 布线速度会相对快一点。
4. **Place input register to IOB:** 输入 Buffer 驱动的寄存器布局到 IOB 上, 改善 IO 逻辑的时序, 默认值为 True;
5. **Place output register to IOB:** 输出/三态 Buffer 驱动的寄存器布局到 IOB 上, 改善 IO 逻辑的时序, 默认值为 True;
6. **Place inout register to IOB:** 双向 Buffer 驱动的寄存器布局到 IOB 上, 改善 IO 逻辑的时序, 默认值为 True。

4.3 解决时序问题

云源软件编译过程中, 在综合优化阶段以及布局布线阶段输出时序分析报告, 通过时序分析报告初步确认时序要求。检查并解决时序问题的流程, 如图 4-1 所示。

图 4-1 时序问题解决流程



4.3.1 分析综合时序报告

在综合之前，优先检查下 RTL 设计是不是符合高云编码规范，比如 DSP 的输出是不是经过寄存器，BSRAM 的输出是不是经过寄存器，状态机的编码方式等。

在综合之后，阅读综合报告中的时序分析结果。

1. 通过“Max Frequency Summary”内容确认最大运行频率是否满足设计频率需求，如图 4-2 所示。

图 4-2 综合报告时序分析最大频率

Max Frequency Summary:

No.	Clock Name	Constraint	Actual Fmax	Logic Level	Entity
1	clk	50.0(MHz)	136.9(MHz)	6	TOP
2	ddk	50.0(MHz)	144.8(MHz)	5	TOP
3	clk_01ms	50.0(MHz)	73.3(MHz)	9	TOP
4	clk_100ms	50.0(MHz)	85.4(MHz)	9	TOP
5	clk_game	50.0(MHz)	81.2(MHz)	9	TOP
6	clk_5ms	50.0(MHz)	53.0(MHz)	13	TOP
7	clk_1ms	50.0(MHz)	189.2(MHz)	4	TOP

2. 通过“Detail Timing Paths Information”内容确认最差路径的预估时序。
 - 如果最差路径上的逻辑级数较大，需要插入寄存器以减小逻辑级数。小蜜蜂[®]家族 C6 速度的芯片，如果要运行到 100MHz，则需要控制逻辑级数在 4 级以下；晨熙[®]家族 C8 速度的芯片，如果要运行到 100MHz，则需要控制逻辑级数在 8 级以下；
 - 如果最差路径上的主要延时 delay 是由 BSRAM 引起的，可在 BSRAM 的最终输出再打一拍；
 - 如果最差路径上的主要延时 delay 是由 DSP 引起的，可在 DSP 的最终输出再打一拍；
 - 如果最差路径上的延时 delay 主要是由 SSRAM 引起的，则可将 SSRAM 综合转换成移位寄存器，使用综合属性 syn_srlstyle。

4.3.2 分析布局布线报告

通过分析布局布线报告的资源使用率，确保设计资源合理利用，如果出现过度利用的情况，则需要更新 RTL 设计，或者添加综合属性约束。

设计资源使用率

在分析时序报告之前，需要先查看布局布线报告，确认设计的资源使用率是不是过度利用。一般来说，当 CLS 的占用率达到 85%以上，或者 BSRAM/DSP 的资源利用率超过 80%时，可称之为资源过度利用的设计，从而也会导致时序收敛问题，如图 4-3 所示。

图 4-3 设计资源使用率

Resource Usage Summary:

Resource	Usage	Utilization
Logic	7057/8640	81%
--LUT,ALU,ROM16	7009(1415 LUT, 5594 ALU, 0 ROM16)	-
--SSRAM(RAM16)	8	-
Register	5828/6867	84%
--Logic Register as Latch	0/6480	0%
--Logic Register as FF	5827/6480	89%
--I/O Register as Latch	0/387	0%
--I/O Register as FF	1/387	1%
CLS	3781/4320	87%
I/O Port	38	-
I/O Buf	38	-
--Input Buf	2	-
--Output Buf	36	-
--Inout Buf	0	-
IOLOGIC	0	0%
BSRAM	24 SDP	92%
DSP	18 MULT18X18	90%
PLL	1/2	50%
DCS	0/8	0%
DQCE	0/24	0%

时钟资源使用率

理论上，设计中的时钟资源不应该超出目标器件的时钟总数量。否则，部分时钟线会绕普通绕线资源，从而可能会导致 setup 或 hold 违规问题，可以通过布局布线报告查看时钟资源使用率，如图 4-4 所示。

图 4-4 时钟资源使用率

Global Clock Usage Summary:

Global Clock	Usage
PRIMARY	1/8(12%)
SECONDARY	1/8(12%)
GCLK_PIN	2/7(28%)
PLL	1/2(50%)
CLKDIV	0/8(0%)
DLLDLY	0/8(0%)

4.3.3 分析布局布线时序报告

减少寄存器扇出

阅读时序报告中的“Setup Analysis Report”最差路径，查看时序路径的起点寄存器的扇出数目，如果扇出过大，可使用复制寄存器的方式减少寄存器的扇出，如图 4-5 所示路径。

图 4-5 时序最差路径

Data Arrival Path:

AT	DELAY	TYPE	RF	FANOUT	LOC	NODE
0.000	0.000					active clock edge time
0.000	0.000					clk_usbif
0.000	0.000	tCL	RR	1	IOB48[A]	usb_ifclk_ibuf/I
0.683	0.683	tINS	RR	7392	IOB48[A]	usb_ifclk_ibuf/O
0.926	0.243	tNET	RR	1	R28C62[0][A]	usb_rf32/mov_sig_src_o_s0/CLK
1.158	0.232	tC2Q	RF	21	R28C62[0][A]	usb_rf32/mov_sig_src_o_s0/Q
2.766	1.609	tNET	FF	1	R32C33[0][B]	n5914_s2/I2
3.219	0.453	tINS	FF	3	R32C33[0][B]	n5914_s2/F

上图所示的最差路径起点的寄存器“usb_rf32/mov_sig_src_o_s0”的扇出为 21，且 net 延时为 1.609ns 占用了整条路径延时的 42%了，严重影响了时序收敛，因此需要在 RTL 设计中对这个寄存器的 net 定义添加属性如：

```
reg mov_sig_src_o_s0 /* synthesis syn_maxfan = 10 */
```

因此，要依据寄存器的实际扇出设计 RTL，但是并不是扇出越小越好，越小也可能会影响寄存器源的扇出增大，同样也会影响时序收敛。

优化 BUFS 资源使用

一般来说，时钟使能 CE 是高扇出 net，且在布局布线阶段会优先使用 BUFS 绕线资源。如果时序最差路径上的 delay 主要由 CE 使用 BUFS 绕线资源引起，则可添加物理约束避免用 BUFS，如：

```
CLOCK_LOC "ce_0" LOCAL_CLOCK
```

