Gowin 时钟资源(Clock)

用户指南

UG286-2.0, 2025-03-21

版权所有 © 2025 广东高云半导体科技股份有限公司

GOWIN高云、Gowin、高云、小蜜蜂、LittleBee 以及晨熙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
2016/05/18	1.05	初始版本。
2016/0715	1.06	标准化插图。
2016/08/31	1.07	适用 GW2A 系列 FPGA 产品。
2016/10/27	1.08	适用 GW2AR 系列 FPGA 产品。
2017/09/22	1.09	根据最新软件原语库修改。
2017/10/16	1.10	增加 GW1N 的对应描述及相关示意图。
2018/01/05	1.2	更新高速时钟相关内容。
2018/04/20	1.3	更新 PLL 端口及参数信息。
2019/12/04	1.4	完善原语描述。
2020/08/18	1.5	章节调整及内容优化。
2021/01/14	1.6	增加 PLLO，OSCO，CLKDIVG 和 DCCG 模块内容。
2021/02/01	1.7	● rPLL 增加了 CLKOUTD3 的时序关系； ● 更新 PLLO 描述内容； ● 新增 GW1NR-2、GW2AN-55C 器件。
2021/04/13	1.7.1	删除 GW1NZ-2 器件。
2021/06/18	1.8	● 增加器件 GW1N-2B、GW1N-1P5、GW1N-1P5B、GW1NR-2B、GW2AN-18X 和 GW2AN-9X； ● 更新 IP 调用部分截图，删除 Help 内容。
2021/9/10	1.9	增加锁相环相关内容。
2021/10/12	1.9.1	● 更新 PLL IP core 调用时不同场景介绍； ● 更新 CLKDIVG 来源描述。
2022/01/24	1.9.2	示例代码格式微调。
2022/05/20	1.9.3	更新 OSCZ 描述。
2022/07/22	1.9.4	● 更新 PLL 模块描述； ● 更新 OSCZ 适用器件。
2022/08/10	1.9.5	更新 rPLL、OSC 适用器件。
2022/11/01	1.9.6	删除器件 GW1NS-2, GW1NS-2C, GW1NSE-2C, GW1NSR-2, GW1NSR-2C。
2023/09/12	1.9.7	● 更新 IP 调用截图，配置框 File 改为 General，添加 Device Version 选项； ● 更新 PLLO 占空比微调描述。
2024/02/02	1.9.8	更新 5.4.1 原语介绍，新增 DLLDLY 单步延时时间的描述。
2024/08/09	1.9.9	补充 DHCEN 连接规则。
2025/03/21	2.0	● 更新“表 5-24 CLKDIVG 适用器件”。 ● 新增在 GW1NZ-1 器件使用 OSCZ 时的相关注释。 ● 更新“表 5-14 PLLO 端口介绍”。

目录

目录 i
图目录 iii
表目录 v
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 概述 3
2.1 全局时钟 3
2.2 高速时钟 6
2.3 锁相环 6
2.3.1 GW1N-1P5/GW1N-2 /GW1NR-2 /GW2AN-18X /GW2AN-9X 6
2.3.2 小蜜蜂®(LittleBee®)家族及晨熙®家族其它器件 8
3 全局时钟 10
3.1 DQCE 10
3.1.1 原语介绍 10
3.1.2 IP 调用 11
3.2 DCS 13
3.2.1 原语介绍 13
3.2.2 IP 调用 16
4 高速时钟 18
4.1 DHCEN 18
4.1.1 原语介绍 18
4.1.2 IP 调用 19
4.2 DHCENC 21
4.2.1 原语介绍 21
4.2.2 IP 调用 22

4.3 DCC 22
4.3.1 原语介绍 22
4.4 DCCG 24
4.4.1 原语介绍 24
4.5 CLKDIV2 25
4.5.1 原语介绍 25
4.5.2 IP 调用 27
5 系统时钟 29
5.1 rPLL 29
5.1.1 原语介绍 29
5.1.2 IP 调用 39
5.2 PLLVR 43
5.2.1 原语介绍 43
5.2.2 IP 调用 50
5.3 PLLO 52
5.3.1 原语介绍 52
5.3.2 IP 调用 70
5.4 DLLDLY 75
5.4.1 原语介绍 75
5.4.2 IP 调用 77
5.5 CLKDIV 79
5.5.1 原语介绍 79
5.5.2 IP 调用 81
5.6 CLKDIVG 82
5.6.1 原语介绍 82
5.6.2 IP 调用 84
5.7 DQS 86
5.7.1 原语介绍 86
6 晶振时钟 92
6.1 原语介绍 92
6.1.1 OSC 92
6.1.2 OSCZ 94
6.1.3 OSCH 96
6.1.4 OSCO 98
6.1.5 OSCW 100
6.2 IP 调用 101

图目录

图 2-1 小蜜蜂®家族（1K、2K、4K）FPGA 产品 GCLK 象限分布示意图 4
图 2-2 小蜜蜂®家族（9K）及晨熙®家族 FPGA 产品 GCLK 象限分布示意图 5
图 2-3 PLL 示意图 7
图 2-4 PLL 示意图 8
图 3-1 DQCE 端口示意图 10
图 3-2 DQCE 的 IP Customization 窗口结构 12
图 3-3 DCS 端口示意图 13
图 3-4 Non-Glitchless 模式时序图 15
图 3-5 DCS mode：RISING 时序图 16
图 3-6 DCS mode：FALLING 时序图 16
图 3-7 DCS mode：CLK0_GND 时序图 16
图 3-8 DCS mode：CLK0_VCC 时序图 16
图 3-9 DCS 的 IP Customization 窗口结构 17
图 4-1 DHCEN 端口示意图 18
图 4-2 DHCEN 的 IP Customization 窗口结构 20
图 4-3 DHCENC 端口示意图 21
图 4-4 DCC 端口示意图 22
图 4-5 DCCG 端口示意图 24
图 4-6 CLKDIV2 端口示意图 26
图 4-7 CLKDIV2 的 IP Customization 窗口结构 28
图 5-1 rPLL 端口示意图 30
图 5-2 输入源为 CLKOUT 时 CLKOUTD3 时序图 31
图 5-3 输入源为 CLKOUTP 时 CLKOUTD3 时序图 31
图 5-4 rPLL 的 IP Customization 窗口结构 40
图 5-5 PLLVR 端口示意图 44
图 5-6 PLLVR 的 IP Customization 窗口结构 51
图 5-7 PLLO 端口示意图 53
图 5-8 B 通道占空比微调时序图(微调方向为 1'b1，步长为 1) 62
图 5-9 B 通道占空比微调时序图(微调方向为 1'b0，步长为 1) 62

图 5-10 PLLO 的 IP Customization 窗口结构 70
图 5-11 DLLDLY 端口示意图 75
图 5-12 DLLDLY 的 IP Customization 窗口结构 78
图 5-13 CLKDIV 端口示意图 79
图 5-14 CLKDIV 的 IP Customization 窗口结构 81
图 5-15 CLKDIVG 端口示意图 83
图 5-16 CLKDIVG 的 IP Customization 窗口结构 85
图 5-17 DQS 端口示意图 86
图 6-1 OSC 端口示意图 93
图 6-2 OSCZ 端口示意图 95
图 6-3 OSCH 端口示意图 96
图 6-4 OSCO 端口示意图 98
图 6-5 OSCW 端口示意图 100
图 6-6 OSC 的 IP Customization 窗口结构 102

表目录

表 1-1 术语、缩略语 1
表 2-1 PLL 端口定义 7
表 2-2 PLL 端口定义 8
表 3-1 DQCE 端口介绍 10
表 3-2 DCS 端口介绍 14
表 3-3 DCS 参数介绍 14
表 4-1 DHCEN 端口介绍 18
表 4-2 DHCENC 适用器件 21
表 4-3 DHCENC 端口介绍 21
表 4-4 DCC 适用器件 22
表 4-5 DCC 端口介绍 23
表 4-6 DCC 参数介绍 23
表 4-7 DCCG 适用器件 24
表 4-8 DCCG 端口介绍 24
表 4-9 DCCG 参数介绍 24
表 4-10 CLKDIV2 端口介绍 26
表 4-11 CLKDIV2 参数介绍 26
表 5-1 rPLL 适用器件 29
表 5-2 rPLL 端口介绍 30
表 5-3 rPLL 参数介绍 31
表 5-4 IDSEL 端口参数对照表 33
表 5-5 FBDSEL 端口参数对照表 33
表 5-6 ODSEL 端口参数对照表 34
表 5-7 rPLL 相位参数调整对照表 34
表 5-8 rPLL 占空比参数调整对照表 35
表 5-9 rPLL 延迟参数调整对照表 36
表 5-10 PLLVR 适用器件 43
表 5-11 PLLVR 端口介绍 44
表 5-12 PLLVR 参数介绍 45

表 5-13 PLLO 适用器件 52
表 5-14 PLLO 端口介绍 53
表 5-15 PLLO 参数介绍 54
表 5-16 IDSEL 端口参数对照表 59
表 5-17 FBDSEL 端口参数对照表 60
表 5-18 ODSELX（X=A/B/C/D）端口参数对照表 60
表 5-19 PLLO 占空比微调对照表 62
表 5-20 DLLDLY 端口介绍 75
表 5-21 DLLDLY 参数介绍 76
表 5-22 CLKDIV 端口介绍 79
表 5-23 CLKDIV 参数介绍 80
表 5-24 CLKDIVG 适用器件 82
表 5-25 CLKDIVG 端口介绍 83
表 5-26 CLKDIVG 参数介绍 83
表 5-27 DQS 适用器件 86
表 5-28 DQS 端口介绍 87
表 5-29 DQS 参数介绍 88
表 6-1 OSC 适用器件 92
表 6-2 OSC 端口介绍 93
表 6-3 OSC 参数介绍 93
表 6-4 OSCZ 适用器件 94
表 6-5 OSCZ 端口介绍 95
表 6-6 OSCZ 参数介绍 95
表 6-7 OSCH 适用器件 96
表 6-8 OSCH 端口介绍 97
表 6-9 OSCH 参数介绍 97
表 6-10 OSCO 适用器件 98
表 6-11 OSCO 端口介绍 98
表 6-12 OSCO 参数介绍 98
表 6-13 OSCW 适用器件 100
表 6-14 OSCO 端口介绍 100
表 6-15 OSCW 参数介绍 100

1 关于本手册

1.1 手册内容

本文档介绍了时钟资源的功能、原语定义及使用方法。

1.2 相关文档

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

- DS100, GW1N 系列 FPGA 产品数据手册
- DS117, GW1NR 系列 FPGA 产品数据手册
- DS821, GW1NS 系列 FPGA 产品数据手册
- DS861, GW1NSR 系列 FPGA 产品数据手册
- DS871, GW1NSE 系列 FPGA 产品数据手册
- DS841, GW1NZ 系列 FPGA 产品数据手册
- DS102, GW2A 系列 FPGA 产品数据手册
- DS226, GW2AR 系列 FPGA 产品数据手册

1.3 术语、缩略语

表 1-1 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
CIU	Configurable Interface Unit	可配置接口单元
CLKDIV	Clock Divider	时钟分频器
CRU	Configurable Routing Unit	可配置布线单元
DCC	Duty Cycle Correction	高速时钟占空比校正模块
DCS	Dynamic Clock Selector	动态时钟选择器
DHCEN	Dynamic HCLK Clock Enable with Inverted Gate	动态高速时钟使能

术语、缩略语	全称	含义
DLLDLY	DLL Delay	DLL 延迟
DQCE	Dynamic Quadrant Clock Enable	动态象限时钟使能
DQS	Bidirectional Data Strobe Circuit for DDR Memory	DDR 存储器双向数据选通电路
GCLK	Global Clock	全局时钟
HCLK	High-speed Clock	高速时钟
LW	Long Wire	长线
OSC	Oscillator	晶体振荡器
PCLK	Primary Clock	主时钟
PLL	Phase-locked Loop	锁相环

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

E-mail：support@gowinsemi.com

Tel：+86 755 8262 0391

2 概述

本章介绍了高云半导体 FPGA 产品的时钟资源，包括专用的时钟输入、缓冲区和布线资源。时钟的基础设施提供了一系列低电容、低偏移互连线，非常适合承载高频信号，最大限度地减少时钟偏差和提高性能，可应用于所有的时钟信号。

时钟资源及布线对 FPGA 高性能的应用至关重要。高云半导体 FPGA 产品提供了专用全局时钟（GCLK），包括主时钟（PCLK）和长线（LW），直接连接到器件的所有资源。除了 GCLK 资源，还提供了锁相环（PLL）、高速时钟 HCLK 和 DDR 存储器接口数据脉冲时钟 DQS 等时钟资源。

2.1 全局时钟

GCLK 在高云 FPGA 产品中按象限分布，小蜜蜂®家族（1K、2K、4K）FPGA 产品中 GCLK 分成 L、R 两个象限，如图 2-1 所示。小蜜蜂®家族（9K）及晨熙®家族 FPGA 产品中 GCLK 分为 BL、BR、TL、TR 四个象限，如图 2-2 所示。每个象限提供 8 个 GCLK 网络，每个 GCLK 的可选时钟源包括专用时钟输入管脚和普通的布线资源，使用专用时钟输入管脚可以取得更好的时钟性能。

LW 一方面可以用作控制线，给 DFF 提供时钟使能（CE）、置复位（SET/RESET）信号；另一方面，还可以用作逻辑绕线，作为普通数据信号使用。

图 2-1 小蜜蜂®家族（1K、2K、4K）FPGA 产品 GCLK 象限分布示意图

图 2-2 小蜜蜂®家族（9K）及晨熙®家族 FPGA 产品 GCLK 象限分布示意图

每个象限的 GCLK0~GCLK5 由 DQCE 动态控制打开/关闭。关闭 GCLK0~GCLK5 时钟，GCLK0~GCLK5 驱动的内部逻辑不再翻转，从而降低了器件的总体功耗。

每个象限的 GCLK6~GCLK7 由 DCS 控制，内部逻辑可以通过 CRU 在四个时钟输入之间动态选择，输出不带毛刺的时钟。

2.2 高速时钟

高云半导体 FPGA 产品的高速时钟 HCLK，具有低抖动和低偏差性能，可以支持 I/O 完成高性能数据传输，是专门针对源时钟同步的数据传输接口而设计的。高速时钟 HCLK 中间有个 HCLKMUX 模块，HCLKMUX 能将任何一个 Bank 中的 HCLK 时钟输入信号送到其他任何一个 Bank 中，这使得 HCLK 的使用更加灵活。

HCLK 资源示意图请参考以下相关文档。

- DS100, GW1N 系列 FPGA 产品数据手册
- DS117, GW1NR 系列 FPGA 产品数据手册
- DS821, GW1NS 系列 FPGA 产品数据手册
- DS861, GW1NSR 系列 FPGA 产品数据手册
- DS871, GW1NSE 系列 FPGA 产品数据手册
- DS841, GW1NZ 系列 FPGA 产品数据手册
- DS102, GW2A 系列 FPGA 产品数据手册
- DS226, GW2AR 系列 FPGA 产品数据手册

2.3 锁相环

锁相环路是一种反馈控制电路，简称锁相环（PLL，Phase-locked Loop）。利用外部输入的参考时钟信号控制环路内部振荡信号的频率和相位。

高云半导体 FPGA 产品的 PLL 模块能够提供可以综合的时钟频率，通过配置不同的参数可以进行时钟的频率调整(倍频和分频)、相位调整、占空比调整等功能。

2.3.1 GW1N-1P5/GW1N-2 /GW1NR-2 /GW2AN-18X /GW2AN-9X

注！

本节描述的锁相环相关内容仅适用于 GW1N-1P5/GW1N-2/GW1NR-2/GW2AN-18X/GW2AN-9X 器件。

PLL 模块的结构框图如图 2-3 所示。

图 2-3 PLL 示意图

PLL 端口定义如表 2-1 所示。

表 2-1 PLL 端口定义

端口名称	信号	描述
CLKIN	输入	参考时钟输入
CLKFB	输入	反馈时钟输入
RESET	输入	PLL 全部复位
RESET_P	输入	PLL 关断（Power Down）信号
RESET_I	输入	带 IDIV 的 PLL 全复位
RESET_S	输入	仅复位 B/C/D 这 3 路
IDSEL [5:0]	输入	动态控制 IDIV 值，范围 1~64
FBDSEL [5:0]	输入	动态控制 FBDIV 值，范围 1~64
ODSELA[6:0]	输入	动态控制 ODIVA,范围 1~128
ODSELB[6:0]	输入	动态控制 ODIVB,范围 1~128
ODSELC[6:0]	输入	动态控制 ODIVC,范围 1~128
ODSELD[6:0]	输入	动态控制 ODIVD,范围 1~128
DTA[3:0]	输入	动态控制 CLKOUTA 的 dutycycle
DTB[3:0]	输入	动态控制 CLKOUTB 的 dutycycle
ICPSEL[4:0]	输入	动态控制 ICP 大小
LPFRES[2:0]	输入	动态控制 LPFRES 大小
PSDIR	输入	动态控制相位移动方向
PSSEL[1:0]	输入	动态控制相位移动通道选择
PSPULSE	输入	动态控制相位移动时钟
ENCLKA ENCLKB ENCLKC	输出	动态控制时钟输出使能

端口名称	信号	描述
ENCLKD		
CLKOUTA	输出	A 通道时钟输出（默认）
CLKOUTB	输出	B 通道时钟输出（默认）
CLKOUTC	输出	C 通道时钟输出（默认）
CLKOUTD	输出	D 通道时钟输出（默认）
LOCK	输出	PLL 锁定指示： 1：锁定 0：失锁

PLL 的参考时钟信号可以通过外部 PLL 时钟管脚输入，也可以是通过绕线过去的全局时钟信号、高速时钟信号或普通数据信号。PLL 的反馈信号可以是外部 PLL 反馈信号的管脚的输入，也可以是通过绕线过去的全局时钟信号、高速时钟信号或普通数据信号。

2.3.2 小蜜蜂®(LittleBee®)家族及晨熙®家族其它器件

注！

本节描述的锁相环相关内容适用于小蜜蜂®(LittleBee®)家族及晨熙®家族中除 GW1N-1P5/GW1N-2/GW1NR-2 及 GW2AN-18X/GW2AN-9X 以外的器件。

PLL 模块的结构框图如图 2-4 所示。

图 2-4 PLL 示意图

PLL 端口定义如表 2-2 所示。

表 2-2 PLL 端口定义

端口名称	信号	描述
CLKIN	输入	参考时钟输入
CLKFB	输入	反馈时钟输入
RESET	输入	PLL 全部复位
RESET_P	输入	PLL 关断（Power Down）信号

端口名称	信号	描述
IDSEL [5：0]	输入	动态控制 IDIV 值，范围 1~64。
FBDSEL [5：0]	输入	动态控制 FBDIV 值，范围 1~64。
PSDA [3：0]	输入	动态相位控制(上升沿有效)
DUTYDA [3：0]	输入	动态占空比控制(下降沿有效)
FDLY [3：0]	输入	CLKOUTP 动态延迟控制
CLKOUT	输出	无相位和占空比调整的时钟输出
CLKOUTP	输出	有相位和占空比调整的时钟输出
CLKOUTD	输出	来自 CLKOUT 或 CLKOUTP 分频时钟（由 SDIV 分频器控制）
CLKOUTD3	输出	来自 CLKOUT 或 CLKOUTP 的分频时钟（由 DIV3 分频器控制，DIV3 分频值固定为 3）。
LOCK	输出	PLL 锁定指示： 1：锁定； 0：失锁

PLL 的参考时钟信号可以通过外部 PLL 时钟管脚输入，也可以是通过绕线过去的全局时钟信号、高速时钟信号或普通数据信号。PLL 的反馈信号可以是外部 PLL 反馈信号的管脚的输入，也可以是通过绕线过去的全局时钟信号、高速时钟信号或普通数据信号。

3 全局时钟

3.1 DQCE

3.1.1 原语介绍

Gowin FPGA 器件具有动态时钟控制功能，允许内部逻辑动态启用或禁用象限内的 PCLK 网络。此外，通过配置参数可以禁用 DQCE 动态时钟控制功能，始终启用 PCLK 网络。当禁用 PCLK 时钟网络时，由该时钟驱动的所有逻辑都不再翻转，从而降低器件的总功耗。

功能描述

通过 DQCE 可动态打开/关闭 GCLK0~GCLK5。关闭 GCLK0~GCLK5 时钟，GCLK0~GCLK5 驱动的内部逻辑不再翻转，降低了器件的总体功耗。DQCE 正常工作，需要 CLKIN 信号至少有一次高电平到低电平的下降沿变化。

端口示意图

图 3-1 DQCE 端口示意图

端口介绍

表 3-1 DQCE 端口介绍

端口名	I/O	描述
CLKIN	Input	时钟输入信号
CE	Input	时钟使能信号，高电平有效。
CLKOUT	Output	时钟输出信号

原语例化

可以直接实例化原语，也可以通过 IP Core Generator 工具产生。

Verilog 例化：

```
DQCE dqce_inst (
    .CLKIN(clkin),
    .CE(ce),
    .CLKOUT(clkout)
);
```

VHDL 例化：

```
COMPONENT DQCE
    PORT(
        CLKOUT:OUT std_logic;
        CE:IN std_logic;
        CLKIN:IN std_logic
    );
END COMPONENT;
uut:DQCE
PORT MAP(
    CLKIN=>clkin,
    CLKOUT=>clkout,
    CE=>ce
);
```

3.1.2 IP 调用

在 IP Core Generator 界面中单击 DQCE，界面右侧会显示 DQCE 的相关信息概要。

IP 配置

在 IP Core Generator 界面中，双击“DQCE”，弹出 DQCE 的“IP Customization”窗口，该窗口包括“General”配置框和端口显示框图，如图 3-2 所示。

图 3-2 DQCE 的 IP Customization 窗口结构

1. General 配置框
   General 配置框用于配置产生的 IP 设计文件的相关信息。

- Device：显示已配置的 Device 信息；
- Device Version：显示已配置的 Device Version 信息；
- Part Number：显示已配置的 Part Number 信息；
- Language：配置产生的 IP 设计文件的硬件描述语言。选择右侧下拉列表框，选择目标语言，支持 Verilog 和 VHDL；
- Module Name：配置产生的 IP 设计文件的 module name。在右侧文本框可重新编辑模块名称。Module Name 不能与原语名称相同，若相同，则报出 Error 提示；
- File Name：配置产生的 IP 设计文件的文件名。在右侧文本框可重新编辑文件名称；
- Create In：配置产生的 IP 设计文件的目标路径。可在右侧文本框中重新编辑目标路径，也可通过文本框右侧选择按钮选择目标路径。

2. 端口显示框图
   端口显示框图显示 IP Core 的配置结果示例框图，如图 3-2 所示。

IP 生成文件

IP 窗口配置完成后，产生以配置文件“File Name”命名的三个文件，

以默认配置为例进行介绍：

- IP 设计文件“gowin_dqce.v”为完整的 verilog 模块，根据用户的 IP 配置，产生实例化的 DQCE；
- IP 设计使用模板文件 gowin_dqce_tmp.v，为用户提供 IP 设计使用模板文件；
- IP 配置文件“gowin_dqce.ipc”，用户可加载该文件对 IP 进行配置。

注！

如配置中选择的语言是 VHDL，则产生的前两个文件名后缀为.vhd。

3.2 DCS

3.2.1 原语介绍

每个象限内有两个 DCS，分别对应 GCLK6 和 GCLK7。DCS 的输出连接到 GCLK6 或 GCLK7，即一个象限的 8 个 GCLK 中，GCLK6、GCLK7 带有动态时钟选择（DCS）功能。DCS 的时钟选择信号 CLKSEL 来自 CIU，内部逻辑可以通过 CRU 使 CLKOUT 在四个时钟输入之间进行动态切换。

功能描述

每个象限的 GCLK6~GCLK7 由 DCS 控制，选择四个输入时钟中的一个作为全局时钟。内部逻辑可以通过 CRU 在四个时钟输入之间动态选择，输出不带毛刺的时钟。

DCS 存在两种时钟切换模式，分别是“Non-Glitchless”和“Glitchless”模式。

在 Non-Glitchless 模式下，DCS 的作用类似于常规多路复用器，仅通过 CLKSEL 信号切换时钟信号，允许输出上的毛刺，实际情况取决于切换的时间。

在 Glitchless 无毛刺模式下，通过参数 DCS_MODE 设置模式，配置 CLKSEL 信号动态切换时钟信号，可以避免输出时钟上的毛刺。

端口示意图

图 3-3 DCS 端口示意图

端口介绍

表 3-2 DCS 端口介绍

端口名	I/O	描述
CLK0	Input	时钟输入信号 0
CLK1	Input	时钟输入信号 1
CLK2	Input	时钟输入信号 2
CLK3	Input	时钟输入信号 3
CLKSEL[3:0]	Input	时钟选择信号
SELFORCE	Input	强制模式选择 0：glitchless 模式 1：Non-glitchless 模式
CLKOUT	Output	时钟输出信号

参数介绍

表 3-3 DCS 参数介绍

参数名	取值范围	默认值	描述
DCS_MODE	“CLK0”, “CLK1”, “CLK2”, “CLK3”, “GND”, “VCC”, “RISING”, “FALLING”, “CLK0_GND”, “CLK1_GND”, “CLK2_GND”, “CLK3_GND”, “CLK0_VCC”, “CLK1_VCC”, “CLK2_VCC”, “CLK3_VCC”	“RISING”	设置 DCS 模式

原语例化

可以直接实例化原语，也可以通过 IP Core Generator 工具产生。

Verilog 例化：

```
DCS dcs_inst (
    .CLK0(clk0),
    .CLK1(clk1),
    .CLK2(clk2),
    .CLK3(clk3),
    .CLKSEL(clksel[3:0]),
    .SELFORCE(selforce),
    .CLKOUT(clkout)
);
defparam dcs_inst.DCS_MODE=“RISING”;
```

Vhdl 例化：

```
COMPONENT DCS
        GENERIC(DCS_MODE:string:="RISING");
          PORT(
                CLK0:IN std_logic;
                CLK1:IN std_logic;
                CLK2:IN std_logic;
                CLK3:IN std_logic;
                CLKSEL:IN std_logic_vector(3 downto 0);
                SELFORCE:IN std_logic;
                CLKOUT:OUT std_logic
          );
END COMPONENT;
uut:DCS
        GENERIC MAP(DCS_MODE=>"RISING")
        PORT MAP(
            CLK0=>clk0,
            CLK1=>clk1,
            CLK2=>clk2,
            CLK3=>clk3,
            CLKSEL=>clksel,
            SELFORCE=>selforce,
            CLKOUT=>clkout
);
```

时序图

Non-Glitchless 模式时序如图 3-4 所示，CLKSEL[3]~CLKSEL[0]分别对应选择 CLK3~CLK0，高电平有效，转换时序相同。

图 3-4 Non-Glitchless 模式时序图

Glitchless 模式时序如图 3-5 到图 3-8 所示，用

CLKSEL[3]~CLKSEL[0]分别对应选择 CLK3~CLK0，转换时序相同。

图 3-5 DCS mode：RISING 时序图

图 3-6 DCS mode：FALLING 时序图

图 3-7 DCS mode：CLK0_GND 时序图

图 3-8 DCS mode：CLK0_VCC 时序图

3.2.2 IP 调用

在 IP Core Generator 界面中单击 DCS，界面右侧会显示 DCS 的相关信息概要。

IP 配置

在 IP Core Generator 界面中，双击“DCS”，弹出 DCS 的“IP Customization”窗口，该窗口包括“General”配置框、“Options”配置框和端口显示框图，如图 3-9 所示。

图 3-9 DCS 的 IP Customization 窗口结构

1. General 配置框
   General 配置框用于配置产生的 IP 设计文件的相关信息。DCS 的 General 配置框的使用和 DQCE 模块的类似，请参考 DQCE 中的 General 配置框。
2. Options 配置框
   Options 配置框用于用户自定义配置 IP，Options 配置框如图 3-9 所示。

- Glitchless Mode：使能/失能 Glitchless 模式。
- DCS Mode：设置 DCS 模式。

3. 端口显示框图
   端口显示框图显示 IP Core 的配置结果示例框图，如图 3-9 所示。

IP 生成文件

IP 窗口配置完成后，产生以配置文件“File Name”命名的三个文件，以默认配置为例进行介绍：

- IP 设计文件“gowin_dcs.v”为完整的 verilog 模块，根据用户的 IP 配置，产生实例化的 DCS；
- IP 设计使用模板文件 gowin_dcs_tmp.v，为用户提供 IP 设计使用模板文件；
- IP 配置文件：“gowin_dcs.ipc”，用户可加载该文件对 IP 进行配置。

注！

如配置中选择的语言是 VHDL，则产生的前两个文件名后缀为.vhd。

4 高速时钟

4.1 DHCEN

4.1.1 原语介绍

DHCEN 可动态地打开/关闭 HCLK 高速时钟信号，CE 低电平时导通。

端口示意图

图 4-1 DHCEN 端口示意图

端口介绍

表 4-1 DHCEN 端口介绍

端口名	I/O	描述
CLKIN	input	时钟输入信号
CE	input	时钟使能输入信号，低电平有效
CLKOUT	output	时钟输出信号

连接规则

- DHCEN 的输出可连接至 IOLOGIC 的 FCLK；
- DHCEN 的输出可连接至 CLKDIV 的 HCLKIN；
- DHCEN 的输出可连接至 DQS 的 FCLK；
- DHCEN 的输出可连接至 rPLL/PLLVR/PLLO 的 CLKIN、CLKFB；
- DHCEN 的输出可连接至 DCC 的 CLKIN；
- DHCEN 的输出可连接至 DCCG 的 CLKIN；
- DHCEN 的输出可连接至 CLKDIV2 的 HCLKIN。

注！

器件 GW1N-9 C 版本/GW1NR-9 C 版本/GW1N-2/GW1NR-2 /GW1N-1P5，DHCEN 不支持驱动 rPLL/PLLO，且当 rPLL/PLLO 驱动了 DHCEN，若再驱动 CLKDIV 则需在 rPLL/PLLO 与 CLKDIV 之间连接 DHCEN。

原语例化

可以直接实例化原语，也可以通过 IP Core Generator 工具产生。

Verilog 例化：

```
DHCEN dhcen_inst (
    .CLKIN(clkin),
    .CE(ce),
    .CLKOUT(clkout)
);
```

Vhdl 例化：

```
COMPONENT DHCEN
    PORT(
        CLKOUT:OUT std_logic;
        CE:IN std_logic;
        CLKIN:IN std_logic
    );
END COMPONENT;
uut:DHCEN
PORT MAP(
    CLKIN=>clkin,
    CLKOUT=>clkout,
    CE=>ce
);
```

4.1.2 IP 调用

在 IP Core Generator 界面中单击 DHCEN，界面右侧会显示 DHCEN 的相关信息概要。

IP 配置

在 IP Core Generator 界面中，双击“DHCEN”，弹出 DHCEN 的“IP Customization”窗口，该窗口包括“General”配置框、“Options”配置框和端口显示框图，如图 4-2 所示。

图 4-2 DHCEN 的 IP Customization 窗口结构

1. General 配置框
   General 配置框用于配置产生的 IP 设计文件的相关信息。DHCEN 的 General 配置框的使用与 DQCE 模块类似，请参考 DQCE 中的 General 配置框。
2. Options 配置框
   Options 配置框用于用户自定义配置 IP，Options 配置框如图 4-2 所示。Enable CLKOUTN：使能时例化 DHCENC，不使能时例化 DHCEN。
3. 端口显示框图
   端口显示框图显示 IP Core 的配置结果示例框图，如图 4-2 所示。

IP 生成文件

IP 窗口配置完成后，产生以配置文件“File Name”命名的三个文件，以默认配置为例进行介绍：

- IP 设计文件“gowin_dhcen.v”为完整的 verilog 模块，根据用户的 IP 配置，产生实例化的 DHCEN；
- IP 设计使用模板文件 gowin_dhcen_tmp.v，为用户提供 IP 设计使用模板文件；
- IP 配置文件：“gowin_dhcen.ipc”，用户可加载该文件对 IP 进行配置。

注！

如配置中选择的语言是 VHDL，则产生的前两个文件名后缀为.vhd。

4.2 DHCENC

4.2.1 原语介绍

DHCENC 可动态地打开/关闭 HCLK 高速时钟信号，CE 低电平时导通。

适用器件

表 4-2 DHCENC 适用器件

家族	系列	器件
小蜜蜂® (LittleBee®)	GW1N	GW1N-9C, GW1N-2, GW1N-1P5, GW1N-2B, GW1N-1P5B
	GW1NR	GW1NR-9C, GW1NR-2, GW1NR-2B

端口示意图

图 4-3 DHCENC 端口示意图

端口介绍

表 4-3 DHCENC 端口介绍

端口名	I/O	描述
CLKIN	input	时钟输入信号
CE	input	时钟使能信号，低电平有效。
CLKOUT	output	时钟输出信号
CLKOUTN	output	时钟输出信号，CLKOUTN 取反。

原语例化

可以直接实例化原语，也可以通过 IP Core Generator 工具产生。

Verilog 例化：

```
DHCENC dhcenc_inst (
    .CLKIN(clkin),
    .CE(ce),
    .CLKOUT(clkout),
    .CLKOUTN(clkoutn)
);
```

VHDL 例化：

```
COMPONENT DHCENC
        PORT(
                CLKOUT:OUT std_logic;
                CLKOUTN:OUT std_logic;
                CE:IN std_logic;
                CLKIN:IN std_logic
        );
END COMPONENT;
uut:DHCENC
PORT MAP(
    CLKIN=>clkin,
    CLKOUT=>clkout,
    CLKOUTN=>clkoutn,
    CE=>ce
);
```

4.2.2 IP 调用

DHCENC 与 DHCEN 的 IP 界面及调用方法相同，参考 4.1.2 IP 调用。

4.3 DCC

4.3.1 原语介绍

DCC，高速时钟占空比校正模块。

适用器件

表 4-4 DCC 适用器件

家族	系列	器件
小蜜蜂® (LittleBee®)	GW1N	GW1N-9C
	GW1NR	GW1NR-9C

端口示意图

图 4-4 DCC 端口示意图

端口介绍

表 4-5 DCC 端口介绍

端口名	I/O	描述
CLKIN	input	时钟输入信号
CLKOUT	output	时钟输出信号

参数介绍

表 4-6 DCC 参数介绍

参数名	取值范围	默认值	描述
DCC_EN	1'b1, 1'b0	1'b1	1'b1:使能 DCC； 1'b0:禁用 DCC
FCLKIN	-	50.0	输入时钟频率

原语例化

Verilog 例化：

```
DCC dcc_inst (
      .CLKIN(clkin),
      .CLKOUT(clkout)
);
defparam dcc_inst.DCC_EN=1'b1;
defparam dcc_inst.FCLKIN=50.0;
```

VHDL 例化：

```
COMPONENT DCC
        GENERIC (
            DCC_EN : bit := '1';    --'1':enable dcc; '0': disable dcc
            FCLKIN : REAL := 50.0 --frequency of the clkin(M)
        );
        PORT(
            CLKOUT:OUT std_logic;
            CLKIN:IN std_logic
        );
END COMPONENT;
uut:DCC
GENERIC MAP(
                DCC_EN=>'1',
```

```
                    FCLKIN=>50.0
                    )
    PORT MAP(
        CLKIN=>clkin,
        CLKOUT=>clkout
    );
```

4.4 DCCG

4.4.1 原语介绍

DCCG，高速时钟占空比校正模块。

适用器件

表 4-7 DCCG 适用器件

家族	系列	器件
小蜜蜂® (LittleBee®)	GW1N	GW1N-2, GW1N-1P5, GW1N-2B, GW1N-1P5B
	GW1NR	GW1NR-2, GW1NR-2B

端口示意图

图 4-5 DCCG 端口示意图

端口介绍

表 4-8 DCCG 端口介绍

端口名	I/O	描述
CLKIN	input	时钟输入信号
CLKOUT	output	时钟输出信号

参数介绍

表 4-9 DCCG 参数介绍

参数名	取值范围	默认值	描述
DCC_MODE	2'b00, 2'b01, 2'b10, 2'b11	2'b00	2'b00/2'b01:Buffered 2'b10: +80ps 2'b11: -80ps

参数名	取值范围	默认值	描述
FCLKIN	-	50.0	输入时钟频率

原语例化

Verilog 例化：

```
DCCG dccg_inst (
      .CLKIN(clkin),
      .CLKOUT(clkout)
);
defparam dccg_inst.DCC_MODE=2'b00;
defparam dccg_inst.FCLKIN=50.0;
```

VHDL 例化：

```
COMPONENT DCCG
        GENERIC (
            DCC_MODE : bit_vector := "00";
            FCLKIN : REAL := 50.0 --frequency of the clkin(M)
        );
         PORT(
              CLKOUT:OUT std_logic;
              CLKIN:IN std_logic
        );
END COMPONENT;
uut:DCCG
GENERIC MAP(
                DCC_MODE=>"00",
                FCLKIN=>50.0
                )
PORT MAP(
    CLKIN=>clkin,
    CLKOUT=>clkout
);
```

4.5 CLKDIV2

4.5.1 原语介绍

CLKDIV2 为时钟分频器，实现时钟的二分频调整。CLKDIV2 的输出只能驱动 DCC/DCCG 的 CLKIN、IOLOGIC 的 FCLK、PLL 的 CLKIN 和 CLKFB、DQS 的 FCLK、CLKDIV 的 HCLKIN。

功能描述

CLKDIV2 为高速时钟分频模块，生成与输入时钟相位一致的 2 分频时钟。

端口示意图

图 4-6 CLKDIV2 端口示意图

端口介绍

表 4-10 CLKDIV2 端口介绍

端口名	I/O	描述
HCLKIN	Input	时钟输入信号
RESETN	Input	异步复位信号，低电平有效。
CLKOUT	Output	时钟输出信号

参数介绍

表 4-11 CLKDIV2 参数介绍

参数名	取值范围	默认值	描述
GSREN	“false”, “true”	“false”	启用全局复位 GSR

原语例化

可以直接实例化原语，也可以通过 IP Core Generator 工具产生。

Verilog 例化：

```
CLKDIV2 clkdiv2_inst (
    .HCLKIN(hclkin),
    .RESETN(resetn),
    .CLKOUT(clkout)
);
defparam clkdiv2_inst.GSREN="false";
```

VHDL 例化：

```
COMPONENT CLKDIV2
```

```
        GENERIC(
            GSREN:STRING:="false"
            );
      PORT(
            HCLKIN:IN std_logic;
            RESETN:IN std_logic;
            CLKOUT:OUT std_logic
            );
END CONPONENT;
  uut:CLKDIV2
        GENERIC MAP(
                GSREN=>"false"
                )
          PORT MAP(
                HCLKIN=>hclkin,
                RESETN=>resetn,
                CLKOUT=>clkout
            );
```

4.5.2 IP 调用

在 IP Core Generator 界面中单击 CLKDIV2，界面右侧会显示 CLKDIV2 的相关信息概要。

IP 配置

在 IP Core Generator 界面中，双击“CLKDIV2”，弹出 CLKDIV2 的“IP Customization”窗口，该窗口包括“General”配置框和端口显示框图，如图 4-7 所示。

图 4-7 CLKDIV2 的 IP Customization 窗口结构

1. General 配置框
   General 配置框用于配置产生的 IP 设计文件的相关信息。CLKDIV2 的 General 配置框的使用和 DQCE 模块的类似，请参考 DQCE 中的 General 配置框。
2. 端口显示框图
   端口显示框图显示 IP Core 的配置结果示例框图，如图 4-7 所示。

IP 生成文件

IP 窗口配置完成后，产生以配置文件"File Name”命名的三个文件，以默认配置为例进行介绍：

- IP 设计文件“gowin_clkdiv2.v”为完整的 verilog 模块，根据用户的 IP 配置，产生实例化的 CLKDIV2；
- IP 设计使用模板文件 gowin_clkdiv2_tmp.v，为用户提供 IP 设计使用模板文件；
- IP 配置文件：“gowin_clkdiv2.ipc”，用户可加载该文件对 IP 进行配置。

注！

如配置中选择的语言是 VHDL，则产生的前两个文件名后缀为.vhd。

5 系统时钟

5.1 rPLL

5.1.1 原语介绍

高云 FPGA 提供了 rPLL，利用外部输入的参考时钟信号控制环路内部振荡信号的频率和相位。

适用器件

表 5-1 rPLL 适用器件

家族	系列	器件
晨熙® (Arora)	GW2A	GW2A-18, GW2A-18C, GW2A-55, GW2A-55C
	GW2AN	GW2AN-55C
	GW2AR	GW2AR-18, GW2AR-18C
	GW2ANR	GW2ANR-18C
小蜜蜂® (LittleBee®)	GW1N	GW1N-1, GW1N-1S, GW1N-4, GW1N-4B, GW1N-4D, GW1N-9, GW1N-9C
	GW1NR	GW1NR-1, GW1NR-4, GW1NR-4B, GW1NR-4D, GW1NR-9, GW1NR-9C
	GW1NRF	GW1NRF-4B
	GW1NZ	GW1NZ-1, GW1NZ-1C

功能描述

rPLL 可基于给定的输入时钟进行时钟相位调整、占空比调整、频率调整（倍频和分频）等来产生不同相位和频率的输出时钟。

rPLL 可对输入时钟 CLKIN 进行频率调整（倍频和分频），计算公式如下：

$$f_{\mathrm{CLKOUT}} = (f_{CLKIN} * \mathrm{FBDIV})/\mathrm{IDIV}$$

$$f_{VCO} = f_{CLKOUT} * ODIV$$

$$f_{CLKOUTD} = f_{CLKOUT}/SDIV$$

$$f_{PFD} = f_{CLKIN}/IDIV = f_{CLKOUT}/FBDIV$$

注！

- f_{CLKIN} 为输入时钟 CLKIN 频率，f_{CLKOUT} 为 CLKOUT 和 CLKOUTP 时钟频率，$f_{CLKOUTD}$ 为 CLKOUTD 时钟频率，f_{PFD} 为 PFD 鉴相频率；
- IDIV、FBDIV、ODIV、SDIV 为不同分频器实际的分频系数，即可通过调整不同分频系数来得到期望频率的时钟信号。
- rPLL 的频率范围可参考 *FPGA 产品数据手册*。

端口示意图

图 5-1 rPLL 端口示意图

端口介绍

表 5-2 rPLL 端口介绍

端口名	I/O	描述
CLKIN	Input	参考时钟输入信号
CLKFB	Input	反馈时钟输入信号
RESET	Input	rPLL 异步复位输入信号，高电平有效。
RESET_P	Input	rPLL 关断（Power Down）输入信号，高电平有效，PLL 非 bypass 模式下，RESET_P 高电平时 CLKOUT/CLKOUTP/CLKOUTD/CLKOUTD3 输出为 0。
FBDSEL[5:0]	Input	动态控制 FBDIV 取值，范围 0~63，实际值为 64-FBDSEL。
IDSEL[5:0]	Input	动态控制 IDIV 取值，范围 0~63，实际值为 64-IDSEL。
ODSEL[5:0]	Input	动态控制 ODIV 取值，2,4,8,16,32,48,64,80,96,112,128。
DUTYDA[3:0]	Input	占空比动态调整信号
PSDA[3:0]	Input	相位动态调整信号
FDLY[3:0]	Input	精细延时动态调整信号
CLKOUT	Output	rPLL 时钟输出信号

端口名	I/O	描述
LOCK	Output	rPLL 锁定指示信号，1 表示锁定，0 表示失锁
CLKOUTP	Output	rPLL 基于 CLKOUT 带有相位和占空比调整的时钟输出信号
CLKOUTD	Output	rPLL 经过 SDIV 的时钟输出信号，CLKOUT 或 CLKOUTP 经过 SDIV 分频器后的输出信号。
CLKOUTD3	Output	rPLL 经过 DIV3 的时钟输出信号，CLKOUT 或 CLKOUTP 经过 3 分频后的输出信号

CLKOUTD3 是 3 分频的输出时钟信号，输入源有两个，

- 如果 CLKOUTD3 的输入源是 CLKOUT：

如图 5-2 所示，当 RESET 复位释放后，CLKOUTD3 在时钟 CLKOUT 的第一个下降沿变为高电平，然后在随后的第二个上升沿变为低电平；

图 5-2 输入源为 CLKOUT 时 CLKOUTD3 时序图

- 如果 CLKOUTD3 的输入源是 CLKOUTP：

如图 5-3 所示，当 RESET 复位释放后，CLKOUTD3 在时钟 CLKOUTP 的第一个下降沿变为高电平，然后在随后的第二个上升沿变为低电平。

图 5-3 输入源为 CLKOUTP 时 CLKOUTD3 时序图

参数介绍

表 5-3 rPLL 参数介绍

参数名	取值范围	默认值	描述
FCLKIN	"3"~"500"	"100"	参考时钟频率
IDIV_SEL	0~63	0	IDIV 分频系数静态设置
DYN_IDIV_SEL	"true", "false"	"false"	IDIV 分频系数静态控制参数或动态控制信号选择 ● false: 静态，即选择参数 IDIV_SEL。

参数名	取值范围	默认值	描述
			● true: 动态，即选择信号 IDSEL。
FBDIV_SEL	0~63	0	FBDIV 分频系数静态设置
DYN_FBDIV_SEL	"true", "false"	"false"	FBDIV 分频系数静态控制参数或动态控制信号选择 ● false: 静态，即选择参数 FBDIV_SEL。 ● true: 动态，即选择信号 FBDSEL。
ODIV_SEL	2,4,8,16,32,48,64,80,96,112,128	8	ODIV 分频系数静态设置
DYN_ODIV_SEL	"true", "false"	"false"	ODIV 分频系数静态控制参数或动态控制信号选择 ● false: 静态，即选择参数 ODIV_SEL。 ● true: 动态，即选择信号 ODSEL。
PSDA_SEL	"0000"~"1111"	"0000"	相位静态调整
DUTYDA_SEL	"0010"~"1110"	"1000"	占空比静态调整
DYN_DA_EN	"true", "false"	"false"	选择动态信号作为相位和占空比调整的控制 ● false: 静态控制 ● true: 动态控制
CLKOUT_FT_DIR	1'b1	1'b1	CLKOUT 微调方向设置 1'b1: 减
CLKOUT_DLY_STEP	0,1,2,4	0	CLKOUT 微调系数设置 CLKOUT_DLY_STEP*delay(delay=50ps)
CLKOUTP_FT_DIR	1'b1	1'b1	CLKOUTP 微调方向设置 1'b1: 减
CLKOUTP_DLY_STEP	0,1,2	0	CLKOUTP 微调系数设置 CLKOUTP_DLY_STEP*delay(delay=50ps)
DYN_SDIV_SEL	2~128（偶数）	2	SDIV 分频系数静态设置
CLKFB_SEL	"internal", "external"	"internal"	CLKFB 来源选择 ● internal: 来自内部 CLKOUT 反馈 ● external: 来自外部信号反馈
CLKOUTD_SRC	"CLKOUT", "CLKOUTP"	"CLKOUT"	CLKOUTD 来源选择
CLKOUTD3_SRC	"CLKOUT", "CLKOUTP"	"CLKOUT"	CLKOUTD3 来源选择
CLKOUT_BYPASS	"true","false"	"false"	旁路 rPLL，CLKOUT 直接来自 CLKIN ● true: CLKIN 旁路 rPLL 直接作用于 CLKOUT ● false: 正常模式
CLKOUTP_BYPASS	"true","false"	"false"	旁路 rPLL，CLKOUTP 直接来自 CLKIN

参数名	取值范围	默认值	描述
			● true: CLKIN 旁路 rPLL 直接作用于 CLKOUTP ● false: 正常模式
CLKOUTD_BYPASS	"true","false"	"false"	旁路 rPLL，CLKOUTD 直接来自 CLKIN ● true: CLKIN 旁路 rPLL 直接作用于 CLKOUTD ● false: 正常模式
DEVICE	"GW1N-1", "GW1NR-1", "GW1N-1S", "GW1NZ-1", "GW1NZ-1C", "GW1N-4", "GW1N-4B", “GWIN-4D", "GW1NR-4", "GW1NR-4B", “GW1NR-4D" "GW1NRF-4B", "GW1N-9", "GW1N-9C", "GW1NR-9", "GW1NR-9C", "GW2A-18", "GW2AR-18", "GW2A-55", "GW2A-55C", "GW2AN-55C"	"GW1N-4”	器件选择

表 5-4 IDSEL 端口参数对照表

IDSEL[5:0]	IDIV 静态参数值	IDIV 实际值
111111	0	1
111110	1	2
111101	2	3
111100	3	4
111011	4	5
111010	5	6
111001	6	7
111000	7	8
110111	8	9
……	……	……
000000	63	64

表 5-5 FBDSEL 端口参数对照表

FBDSEL [5:0]	FBDIV 静态参数值	FBDIV 实际值
111111	0	1
111110	1	2

FBDSEL [5:0]	FBDIV 静态参数值	FBDIV 实际值
111101	2	3
111100	3	4
111011	4	5
111010	5	6
111001	6	7
111000	7	8
110111	8	9
……	……	……
000000	63	64

表 5-6 ODSEL 端口参数对照表

ODSEL [5:0]	ODIV 参数值	ODIV 实际值
111111	2	2
111110	4	4
111100	8	8
111000	16	16
110000	32	32
101000	48	48
100000	64	64
011000	80	80
010000	96	96
001000	112	112
000000	128	128

表 5-7 rPLL 相位参数调整对照表

参数 PSDA_SEL 或端口 PSDA 设置	相位调整
0000	0°
0001	22.5°
0010	45°
0011	67.5°
0100	90°
0101	112.5°
0110	135°
0111	157.5°
1000	180°
1001	202.5°
1010	225°

参数 PSDA_SEL 或端口 PSDA 设置	相位调整
1011	247.5°
1100	270°
1101	292.5°
1110	315°
1111	337.5°

表 5-8 rPLL 占空比参数调整对照表

参数 DUTYDA_SEL 设置	占空比设置值（/16）
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7
1000	8
1001	9
1010	10
1011	11
1100	12
1101	13
1110	14

动态占空比调整需要参考相移设置。例如，当相移设置为"0”（0000）时，50%占空比设置为"8”（1000）。如果相移的设置是"180º”，50%占空比的设置为"0”（0000）。

动态占空比计算：

- 若 DUTYDA [3:0]> PSDA [3:0]时，DutyCycle=1/16 x (DUTYDA [3:0]- PSDA [3:0])。
- 若 DUTYDA [3:0]< PSDA [3:0]时，DutyCycle=1/16 x (16+ DUTYDA [3:0]- PSDA [3:0])。

注！

不支持 DutyCycle = 0, 1, 15 这三种情况。

可以通过端口 FDLY [3:0]动态控制输出时钟 CLKOUTP 的延迟。每一步增加 0.125ns。需要结合相移设置实现滞后（时钟信号 CLKOUTP 滞后于输入时钟）和超前（时钟信号 CLKOUTP 超前输入时钟）。

表 5-9 rPLL 延迟参数调整对照表

端口 FDLY [3:0]（GW1N-1/GW1N-1S）	端口 FDLY [3:0]（其他器件）	延迟步数
0000	1111	0
0001	1110	1
0010	1101	2
0100	1011	4
1000	0111	8

原语例化

可以直接实例化原语，也可以通过 IP Core Generator 工具产生。

Verilog 例化：

```
rPLL rpll_inst(
    .CLKOUT(clkout),
    .LOCK(lock),
    .CLKOUTP(clkoutp),
    .CLKOUTD(clkoutd),
    .CLKOUTD3(clkoutd3),
    .RESET(reset),
    .RESET_P(reset_p),
    .CLKIN(clkin),
    .CLKFB(clkfb),
    .FBDSEL(fbdsel),
    .IDSEL(idsel),
    .ODSEL(odsel),
    .PSDA(psda),
    .DUTYDA(dutyda),
    .FDLY(fdly)
);
defparam rpll_inst.FCLKIN = "50";
defparam rpll_inst.DYN_IDIV_SEL = "false";
defparam rpll_inst.IDIV_SEL = 0;
defparam rpll_inst.DYN_FBDIV_SEL = "false";
defparam rpll_inst.FBDIV_SEL = 1;
defparam rpll_inst.ODIV_SEL = 8;
defparam rpll_inst.PSDA_SEL = "0100";
```

```
defparam rpll_inst.DYN_DA_EN = "false";
defparam rpll_inst.DUTYDA_SEL = "1000";
defparam rpll_inst.CLKOUT_FT_DIR = 1'b1;
defparam rpll_inst.CLKOUTP_FT_DIR = 1'b1;
defparam rpll_inst.CLKOUT_DLY_STEP = 0;
defparam rpll_inst.CLKOUTP_DLY_STEP = 0;
defparam rpll_inst.CLKFB_SEL ="external";
defparam rpll_inst.CLKOUT_BYPASS = "false";
defparam rpll_inst.CLKOUTP_BYPASS = "false";
defparam rpll_inst.CLKOUTD_BYPASS = "false";
defparam rpll_inst.DYN_SDIV_SEL = 2;
defparam rpll_inst.CLKOUTD_SRC = "CLKOUT";
defparam rpll_inst.CLKOUTD3_SRC = "CLKOUT";
defparam rpll_inst.DEVICE = "GW1N-4";
```

VHDL 例化：

```
COMPONENT rPLL
    GENERIC(
            FCLKIN:STRING:= "100.0";
            DEVICE:STRING:= "GW1N-4";
            DYN_IDIV_SEL:STRING:="false";
            IDIV_SEL:integer:=0;
            DYN_FBDIV_SEL:STRING:="false";
            FBDIV_SEL:integer:=0;
            DYN_ODIV_SEL:STRING:="false";
            ODIV_SEL:integer:=8;
            PSDA_SEL:STRING:="0000";
            DYN_DA_EN:STRING:="false";
            DUTYDA_SEL:STRING:="1000";
            CLKOUT_FT_DIR:bit:='1';
            CLKOUTP_FT_DIR:bit:='1';
            CLKOUT_DLY_STEP:integer:=0;
            CLKOUTP_DLY_STEP:integer:=0;
            CLKOUTD3_SRC:STRING:="CLKOUT";
            CLKFB_SEL : STRING:="internal";
            CLKOUT_BYPASS:STRING:="false";
```

```
                CLKOUTP_BYPASS:STRING:="false";
                CLKOUTD_BYPASS:STRING:="false";
                CLKOUTD_SRC:STRING:="CLKOUT";
                DYN_SDIV_SEL:integer:=2
        );
            PORT(
                CLKIN:IN std_logic;
                CLKFB:IN std_logic;
                IDSEL:IN std_logic_vector(5 downto 0);
                FBDSEL:IN std_logic_vector(5 downto 0);
                ODSEL:IN std_logic_vector(5 downto 0);
                RESET:IN std_logic;
                RESET_P:IN std_logic;
                PSDA,FDLY:IN std_logic_vector(3 downto 0);
                DUTYDA:IN std_logic_vector(3 downto 0);
                LOCK:OUT std_logic;
                CLKOUT:OUT std_logic;
                CLKOUTD:OUT std_logic;
                CLKOUTP:OUT std_logic;
                CLKOUTD3:OUT std_logic
            );
    END COMPONENT;
    uut:rPLL
        GENERIC MAP(
                    FCLKIN =>"100.0",
                    DEVICE =>"GW2A-18",
                    DYN_IDIV_SEL=>"false",
                    IDIV_SEL=>0,
                    DYN_FBDIV_SEL=>"false",
                    FBDIV_SEL=>0,
                    DYN_ODIV_SEL=>"false",
                    ODIV_SEL=>8,
                    PSDA_SEL=>"0000",
                    DYN_DA_EN=>"false",
                    DUTYDA_SEL=>"1000",
```

```
                    CLKOUT_FT_DIR=>'1',
                    CLKOUTP_FT_DIR=>'1',
                    CLKOUT_DLY_STEP=>0,
                    CLKOUTP_DLY_STEP=>0,
                    CLKOUTD3_SRC=>"CLKOUT",
                    CLKFB_SEL=>"internal",
                    CLKOUT_BYPASS=>"false",
                    CLKOUTP_BYPASS=>"false",
                    CLKOUTD_BYPASS=>"false",
                    CLKOUTD_SRC=>"CLKOUT",
                    DYN_SDIV_SEL=>2
            )
             PORT MAP(
                CLKIN=>clkin,
                CLKFB=>clkfb,
                IDSEL=>idsel,
                FBDSEL=>fbdsel,
                ODSEL=>odsel,
                RESET=>reset,
                RESET_P=>reset_p,
                PSDA=>psda,
                FDLY=>fdly,
                DUTYDA=>dutyda,
                LOCK=>lock,
                CLKOUT=>clkout,
                CLKOUTD=>clkoutd,
                CLKOUTP=>clkoutp,
                CLKOUTD3=>clkoutd3
             );
```

5.1.2 IP 调用

在 IP Core Generator 界面中，单击“rPLL”，界面右侧会显示 rPLL 的相关信息概要。

IP 配置

在 IP Core Generator 界面中双击“rPLL”，弹出 rPLL 的“IP Customization”窗口。该窗口包括“General”配置框、"Options”配置框和

端口显示框图，如图 5-4 所示。

图 5-4 rPLL 的 IP Customization 窗口结构

1. General 配置框
   General 配置框用于配置产生的 IP 设计文件的相关信息。rPLL 的 General 配置框的使用和 DQCE 模块的类似，请参考 DQCE 中的 General 配置框。

2. Options 配置框
   Options 配置框用于用户自定义配置 IP，Options 配置框如图 5-4 所示。

   - General：配置一般模式和高级模式，配置输出相位和占空比调整的动态、静态模式和使能 PLL Reset。
     - “Mode”选项配置 IP Core 配置的模式，支持一般模式“General Mode”和高级模式“Advanced Mode”。一般模式下输入输入时钟频率和输出时钟频率，软件会自动计算不同的分频系数；高级模式适用于高级用户，允许输入输入频率和不同分频系数得到预期的输出频率；
     - “PLL Phase And Duty Cycle Adjustment”选项配置输出的占空比和相位调整的模式，支持动态调整“Dynamic”和静态调整“Static”；
     - “PLL Reset”选项配置 rPLL 的 Reset 使能模式；
     - “PLL Power Down”选项配置 reset_p 端口使 rPLL 处于节电模式。
   - CLKIN：配置 rPLL 输入时钟的频率，分频参数的设置和 IDSEL

Reset 使能模式。

- “Clock Frequency（频率范围）”配置输入时钟的频率，范围由 device 决定；
- “Divide Factor”可在高级模式下配置分频参数，支持动态模式“Dynamic”和静态模式“Static”，静态模式下可配置分频参数的具体数值，范围为 1~64。若 CLKOUT 的输出频率不在相应 device 要求的范围内，单击“Calculate”或“OK”，会弹出提示窗口提示错误；若 CLKIN/IDIV 的频率不在相应 device 要求的 Clock Frequency 范围内，单击“Calculate”或“OK”，会弹出提示窗口提示错误。

- CLKFB：配置 rPLL 反馈时钟的源和倍频参数。
  - 配置反馈时钟的源时，“Source”选项可选择 Internal 和 External；
  - “Divide Factor”可在高级模式下配置倍频参数，支持动态模式“Dynamic”和静态模式“Static”，静态模式下可配置倍频参数的具体数值，范围为 1~64，配置不合理时，单击“Calculate”按钮或"OK"按钮，会弹出提示窗口提示错误。
- Enable LOCK：使能 LOCK 端口。
- CLKOUT：配置 rPLL 输出时钟期望频率，配置 VCO 参数，配置输出时钟周期的微调参数。
  - “Bypass”选项可配置输出时钟的旁路功能；
  - “Expected Frequency（频率范围）”在一般模式下配置期望的输出时钟 CLKOUT 的频率，范围由 device 决定；
  - “Tolerance（%）”配置 CLKOUT 期望频率和计算出的实际频率的允许误差。
  - “VCO Divide Factor”在高级模式下配置 VCO 参数支持动态模式“Dynamic”和静态模式“Static”，静态模式下可配置分频参数的具体数值，范围为 2/4/8/16/32/48/64/80/96/112/128，配置不合理时，单击“Calculate”或“OK”，会弹出提示窗口提示错误。
  - “Actual Frequency”显示经计算得出的 CLKOUT 实际频率，无需用户配置。
- CLKOUTP：配置相移时钟周期微调参数，配置相移时钟的相位和占空比调整参数，使能/失能相移时钟的 Reset。
  - “Enable CLKOUTP”选项配置相移时钟输出使能；
  - “Bypass”选项配置相移时钟的旁路功能使能；
  - “Phase And Duty Cycle Adjustment（Static）”可在静态模式下配置相位（Phase（degree））和占空比（Duty Cycle）；

- CLKOUTD：配置分频时钟输出的时钟源，配置期望分频时钟输出频率，配置分频时钟分频输出参数，使能/失能分频时钟输出的Reset。
  - “Enable CLKOUTD”选项配置分频时钟输出使能；
  - “Bypass”选项配置分频时钟输出的旁路功能使能；
  - “Source”选项配置分频时钟输出的时钟源，可选 CLKOUT 和 CLKOUTP；
  - “Expected Frequency（频率范围）”在一般模式下配置期望的分频时钟输出的频率，范围由 device 决定；
  - “Tolerance（%）”配置分频时钟输出期望频率和计算出的实际频率的允许误差；
  - “Divide Factor（2~128）”在高级模式下配置分频时钟输出的分频参数，范围为 2~128 之间的偶数，设置为奇数时单击“OK”会提示错误。
  - “Actual Frequency”显示经计算得出的分频时钟输出的实际频率，无需用户配置；
- CLKOUTD3：配置三分频时钟输出的时钟源。
  - “Enable CLKOUTD3”选项配置三分频时钟输出使能；
  - “Source”选项配置三分频时钟输出的时钟源，可选 CLKOUT 和 CLKOUTP。
- Calculate：计算当前配置是否合理。
  - 一般模式“General Mode”下，根据输入输出频率计算配置分频参数、倍频参数和 VCO 参数，计算出的实际频率和期望频率不相等时，单击“Calculate”按钮后会弹出“error”窗口提示错误，并将不合理位置标红。
  - 在高级模式“Advanced Mode”下，计算配置的静态分频参数、倍频参数和 VCO 参数是否合理，若不合理，单击“Calculate”，弹出“error”窗口提示错误；若配置正确，单击“Calculate”，弹出“info”窗口提示配置成功。

3. 端口显示框图
   端口显示框图显示 IP Core 的配置结果示例框图，输入输出端口的个数根据 Options 配置实时更新，如图 5-4 所示。

IP 生成文件

IP 窗口配置完成后，产生以配置文件“File Name”命名的三个文件，以默认配置为例进行介绍：

- IP 设计文件“gowin_rpll.v”为完整的 verilog 模块，根据用户的 IP 配置，产生实例化的 rPLL；

- IP 设计使用模板文件 gowin_rpll_tmp.v，为用户提供 IP 设计使用模板文件；
- IP 配置文件：“gowin_rpll.ipc”，用户可加载该文件对 IP 进行配置。

注！

如配置中选择的语言是 VHDL，则产生的前两个文件名后缀为.vhd。

5.2 PLLVR

5.2.1 原语介绍

高云 FPGA 提供了 PLLVR（Phase_Locked Loop with regulator，带电源调节的锁相环），利用外部输入的参考时钟信号控制环路内部振荡信号的频率和相位。

适用器件

表 5-10 PLLVR 适用器件

家族	系列	器件
小蜜蜂® (LittleBee®)	GW1NS	GW1NS-4, GW1NS-4C, GW1NSR-4, GW1NSR-4C, GW1NSER-4C

功能描述

PLLVR 是带电源调节的 PLL，可基于给定的输入时钟进行时钟相位调整、占空比调整、频率调整（倍频和分频）等来产生不同相位和频率的输出时钟。

PLLVR 的性能如下：

PLLVR 可对输入时钟 CLKIN 进行频率调整（倍频和分频），计算公式如下：

$$f_{\mathrm{CLKOUT}} = (f_{CLKIN} * \mathrm{FBDIV})/\mathrm{IDIV}$$

$$f_{VCO} = f_{CLKOUT} * ODIV$$

$$f_{CLKOUTD} = f_{CLKOUT}/SDIV$$

$$f_{PFD} = f_{CLKIN}/IDIV = f_{CLKOUT}/FBDIV$$

注！

- f_{CLKIN} 为输入时钟 CLKIN 频率，f_{CLKOUT} 为 CLKOUT 和 CLKOUTP 时钟频率，$f_{CLKOUTD}$ 为 CLKOUTD 时钟频率，f_{PFD} 为 PFD 鉴相频率；
- IDIV、FBDIV、ODIV、SDIV 为不同分频器实际的分频系数，即可通过调整不同分频系数来得到期望频率的时钟信号。
- PLLVR 的频率范围可参考 *FPGA 产品数据手册*。

端口示意图

图 5-5 PLLVR 端口示意图

端口介绍

表 5-11 PLLVR 端口介绍

端口名	I/O	描述
CLKIN	Input	参考时钟输入信号
CLKFB	Input	反馈时钟输入信号
RESET	Input	PLLVR 异步复位输入信号，高电平有效。
RESET_P	Input	PLLVR 关断（Power Down）输入信号，高电平有效，PLL 非 bypass 模式下，RESET_P 高电平时 CLKOUT/CLKOUTP/CLKOUTD/CLKOUTD3 输出为 0。
FBDSEL[5:0]	Input	动态控制 FBDIV 取值，范围 0~63，实际值为 64-FBDSEL。
IDSEL[5:0]	Input	动态控制 IDIV 取值，范围 0~63，实际值为 64-IDSEL。
ODSEL[5:0]	Input	动态控制 ODIV 取值，2,4,8,16,32,48,64,80,96,112,128。
DUTYDA[3:0]	Input	占空比动态调整信号
PSDA[3:0]	Input	相位动态调整信号
FDLY[3:0]	Input	精细延时动态调整信号
VREN	Input	PLLVR 电源调节使能信号，高电平有效。
CLKOUT	Output	PLLVR 时钟输出信号
LOCK	Output	PLLVR 锁定指示信号，1 表示锁定，0 表示失锁 。
CLKOUTP	Output	PLLVR 带有相位和占空比调整的时钟输出信号
CLKOUTD	Output	PLLVR 经过 SDIV 的时钟输出信号，CLKOUT 或 CLKOUTP 经过 SDIV 分频器后的输出信号。
CLKOUTD3	Output	PLLVR 经过 DIV3 的时钟输出信号，CLKOUT 或 CLKOUTP 经过 3 分频后的输出信号。

注！

CLKOUTD3 是 CLKOUT 或 CLKOUTP 经过 3 分频后的输出时钟信号，其与 CLKOUT 或 CLKOUTP 的时序关系可参考 rPLL。

参数介绍

表 5-12 PLLVR 参数介绍

参数名	取值范围	默认值	描述
FCLKIN	3~500	100	参考时钟频率
IDIV_SEL	0~63	0	IDIV 分频系数静态设置
DYN_IDIV_SEL	"true", "false"	"false"	IDIV 分频系数静态控制参数或动态控制信号选择 ● false: 静态，即选择参数 IDIV_SEL。 ● true: 动态，即选择信号 IDSEL。
FBDIV_SEL	0~63	0	FBDIV 分频系数静态设置
DYN_FBDIV_SEL	"true”, "false"	"false"	FBDIV 分频系数静态控制参数或动态控制信号选择 ● false: 静态，即选择参数 FBDIV_SEL。 ● true: 动态，即选择信号 FBDSEL。
ODIV_SEL	2,4,8,16,32,48,64,80,96,112,128	8	ODIV 分频系数静态设置
DYN_ODIV_SEL	"true", "false"	"false"	ODIV 分频系数静态控制参数或动态控制信号选择 ● false: 静态，即选择参数 ODIV_SEL。 ● true: 动态，即选择信号 ODSEL。
PSDA_SEL	"0000”~ "1111"	"0000"	相位静态调整
DUTYDA_SEL	"0010”~ "1110"	"1000"	占空比静态调整
DYN_DA_EN	"true", "false"	"false"	选择动态信号作为相位和占空比调整的控制 ● false: 静态控制 ● true: 动态控制
CLKOUT_FT_DIR	1'b1	1'b1	CLKOUT 微调方向设置 1'b1: 减
CLKOUT_DLY_STEP	0,1,2,4	0	CLKOUT 微调系数设置 CLKOUT_DLY_STEP*delay(delay=50ps)
CLKOUTP_FT_DIR	1'b1	1'b1	CLKOUTP 微调方向设置 1'b1: 减
CLKOUTP_DLY_STEP	0,1,2	0	CLKOUTP 微调系数设置 CLKOUTP_DLY_STEP*delay(delay=50ps)

参数名	取值范围	默认值	描述
DYN_SDIV_SEL	2~128（偶数）	2	SDIV 分频系数静态设置
CLKFB_SEL	"internal", "external"	"internal"	CLKFB 来源选择 ● internal: 来自内部 CLKOUT 反馈 ● external: 来自外部信号反馈
CLKOUTD_SRC	"CLKOUT", "CLKOUTP"	"CLKOUT"	CLKOUTD 来源选择
CLKOUTD3_SRC	"CLKOUT", "CLKOUTP”	"CLKOUT"	CLKOUTD3 来源选择
CLKOUT_BYPASS	"true", "false"	"false"	旁路 PLLVR，CLKOUT 直接来自 CLKIN ● true: CLKIN 旁路 PLLVR 直接作用于 CLKOUT ● false: 正常模式
CLKOUTP_BYPASS	"true", "false"	"false"	旁路 PLLVR，CLKOUTP 直接来自 CLKIN ● true: CLKIN 旁路 PLLVR 直接作用于 CLKOUTP ● false: 正常模式
CLKOUTD_BYPASS	"true”, "false”	"false"	旁路 PLLVR，CLKOUTD 直接来自 CLKIN ● true: CLKIN 旁路 PLLVR 直接作用于 CLKOUTD ● false: 正常模式
DEVICE	"GW1NS-4"、"GW1NS-4C"、"GW1NSR-4"、"GW1NSR-4C"、"GW1NSER-4C"	"GW1NS-4"	器件选择

注！

IDSEL、FBDESL、ODSEL 端口及参数对照表，相位和占空比调整对照表等与 rPLL 相同，请参考 rPLL。

原语例化

可以直接实例化原语，也可以通过 IP Core Generator 工具产生。

Verilog 例化：

```
PLLVR pllvr_inst(
    .CLKOUT(clkout),
    .LOCK(lock),
```

```
        .CLKOUTP(clkoutp),
        .CLKOUTD(clkoutd),
        .CLKOUTD3(clkoutd3),
        .VREN(vren),
        .RESET(reset),
        .RESET_P(reset_p),
        .CLKIN(clkin),
        .CLKFB(clkfb),
        .FBDSEL(fbdsel),
        .IDSEL(idsel),
        .ODSEL(odsel),
        .PSDA(psda),
        .DUTYDA(dutyda),
        .FDLY(fdly)
    );
    defparam pllvr_inst.FCLKIN = "50";
    defparam pllvr_inst.DYN_IDIV_SEL = "false";
    defparam pllvr_inst.IDIV_SEL = 0;
    defparam pllvr_inst.DYN_FBDIV_SEL = "false";
    defparam pllvr_inst.FBDIV_SEL = 1;
    defparam pllvr_inst.ODIV_SEL = 8;
    defparam pllvr_inst.PSDA_SEL = "0100";
    defparam pllvr_inst.DYN_DA_EN = "false";
    defparam pllvr_inst.DUTYDA_SEL = "1000";
    defparam pllvr_inst.CLKOUT_FT_DIR = 1'b1;
    defparam pllvr_inst.CLKOUTP_FT_DIR = 1'b1;
    defparam pllvr_inst.CLKOUT_DLY_STEP = 0;
    defparam pllvr_inst.CLKOUTP_DLY_STEP = 0;
    defparam pllvr_inst.CLKFB_SEL ="external";
    defparam pllvr_inst.CLKOUT_BYPASS = "false";
    defparam pllvr_inst.CLKOUTP_BYPASS = "false";
    defparam pllvr_inst.CLKOUTD_BYPASS = "false";
    defparam pllvr_inst.DYN_SDIV_SEL = 2;
    defparam pllvr_inst.CLKOUTD_SRC = "CLKOUT";
    defparam pllvr_inst.CLKOUTD3_SRC = "CLKOUT";
```

```
defparam pllvr_inst.DEVICE = "GW1NS-4";
```

VHDL 例化：

```
COMPONENT PLLVR
    GENERIC(
            FCLKIN:STRING:= "100.0";
            DEVICE:STRING:= "GW1NS-4";
            DYN_IDIV_SEL:STRING:="false";
            IDIV_SEL:integer:=0;
            DYN_FBDIV_SEL:STRING:="false";
            FBDIV_SEL:integer:=0;
            DYN_ODIV_SEL:STRING:="false";
            ODIV_SEL:integer:=8;
            PSDA_SEL:STRING:="0000";
            DYN_DA_EN:STRING:="false";
            DUTYDA_SEL:STRING:="1000";
            CLKOUT_FT_DIR:bit:='1';
            CLKOUTP_FT_DIR:bit:='1';
            CLKOUT_DLY_STEP:integer:=0;
            CLKOUTP_DLY_STEP:integer:=0;
            CLKOUTD3_SRC:STRING:="CLKOUT";
            CLKFB_SEL : STRING:="internal";
            CLKOUT_BYPASS:STRING:="false";
            CLKOUTP_BYPASS:STRING:="false";
            CLKOUTD_BYPASS:STRING:="false";
            CLKOUTD_SRC:STRING:="CLKOUT";
            DYN_SDIV_SEL:integer:=2
    );
        PORT(
            CLKIN:IN std_logic;
            CLKFB:IN std_logic;
            IDSEL:IN std_logic_vector(5 downto 0);
            FBDSEL:IN std_logic_vector(5 downto 0);
            ODSEL:IN std_logic_vector(5 downto 0);
            VREN:IN std_logic;
            RESET:IN std_logic;
```

```
                RESET_P:IN std_logic;
                PSDA,FDLY:IN std_logic_vector(3 downto 0);
                DUTYDA:IN std_logic_vector(3 downto 0);
                LOCK:OUT std_logic;
                CLKOUT:OUT std_logic;
                CLKOUTD:OUT std_logic;
                CLKOUTP:OUT std_logic;
                CLKOUTD3:OUT std_logic
            );
END COMPONENT;
uut:PLLVR
    GENERIC MAP(
                FCLKIN =>"100.0",
                DEVICE =>"GW1NS-4",
                DYN_IDIV_SEL=>"false",
                IDIV_SEL=>0,
                DYN_FBDIV_SEL=>"false",
                FBDIV_SEL=>0,
                DYN_ODIV_SEL=>"false",
                ODIV_SEL=>8,
                PSDA_SEL=>"0000",
                DYN_DA_EN=>"false",
                DUTYDA_SEL=>"1000",
                CLKOUT_FT_DIR=>'1',
                CLKOUTP_FT_DIR=>'1',
                CLKOUT_DLY_STEP=>0,
                CLKOUTP_DLY_STEP=>0,
                CLKOUTD3_SRC=>"CLKOUT",
                CLKFB_SEL=>"internal",
                CLKOUT_BYPASS=>"false",
                CLKOUTP_BYPASS=>"false",
                CLKOUTD_BYPASS=>"false",
                CLKOUTD_SRC=>"CLKOUT",
                DYN_SDIV_SEL=>2
    )
```

```
PORT MAP(
    CLKIN=>clkin,
    CLKFB=>clkfb,
    IDSEL=>idsel,
    FBDSEL=>fbdsel,
    ODSEL=>odsel,
    VREN=>vren,
    RESET=>reset,
    RESET_P=>reset_p,
    PSDA=>psda,
    FDLY=>fdly,
    DUTYDA=>dutyda,
    LOCK=>lock,
    CLKOUT=>clkout,
    CLKOUTD=>clkoutd,
    CLKOUTP=>clkoutp,
    CLKOUTD3=>clkoutd3
);
```

5.2.2 IP 调用

在 IP Core Generator 界面中，单击“PLLVR”，界面右侧会显示 PLLVR 的相关信息概要。

IP 配置

在 IP Core Generator 界面中双击“PLLVR”，弹出 PLLVR 的“IP Customization”窗口。该窗口包括“General”配置框、“Options”配置框、端口显示框图和"Help”按钮，如图 5-6 所示。

图 5-6 PLLVR 的 IP Customization 窗口结构

1. General 配置框
   General 配置框用于配置产生的 IP 设计文件的相关信息。PLLVR 的 General 配置框的使用和 DQCE 模块的类似，请参考 DQCE 中的 General 配置框。
2. Options 配置框
   Options 配置框用于用户自定义配置 IP，Options 配置框如图 5-6 所示。PLLVR 配置框的使用和 rPLL 模块类似，请参考 rPLL 中的 Options 配置框。其中新增 PLL Regulator 选项。
3. 端口显示框图
   端口显示框图显示 IP Core 的配置结果示例框图，输入输出端口的个数根据 Options 配置实时更新，如图 5-6 所示。

IP 生成文件

IP 窗口配置完成后，产生以配置文件"File Name”命名的三个文件，以默认配置为例进行介绍：

- IP 设计文件"gowin_pllvr.v”为完整的 verilog 模块，根据用户的 IP 配置，产生实例化的 PLLVR；
- IP 设计使用模板文件 gowin_pllvr_tmp.v，为用户提供 IP 设计使用模板文件；
- IP 配置文件："gowin_pllvr.ipc”，用户可加载该文件对 IP 进行配置。

注！

如配置中选择的语言是 VHDL，则产生的前两个文件名后缀为.vhd。

5.3 PLLO

5.3.1 原语介绍

高云 FPGA 提供了锁相环 PLLO，支持四路时钟输出，可基于给定的输入时钟进行频率、相位、占空比的调整。

适用器件

表 5-13 PLLO 适用器件

家族	系列	器件
小蜜蜂® (LittleBee®)	GW1N	GW1N-2, GW1N-1P5, GW1N-2B, GW1N-1P5B
	GW1NR	GW1NR-2, GW1NR-2B
晨熙® (Arora)	GW2AN	GW2AN-18X, GW2AN-9X

功能描述

PLLO 支持四路输出时钟，基于给定的输入时钟进行时钟相位调整、占空比调整、频率调整（倍频和分频）等来产生不同相位和频率的输出时钟。若要得到正确的时钟输出，输入时钟频率必须按照 *FPGA 产品数据手册*中描述的频率范围进行设置。

PLLO 可以对输入时钟 CLKIN 进行频率调整（倍频和分频），计算公式如下：

$$f_{\mathrm{CLKOUTA}} = (f_{CLKIN} * \mathrm{FBDIV})/\mathrm{IDIV}$$

$$f_{VCO} = f_{CLKOUTA} * ODIVA$$

$$f_{CLKOUTX} = f_{IN_ODIVX}/ODIVX$$

$$f_{PFD} = f_{CLKIN}/IDIV = f_{CLKOUTA}/FBDIV$$

注！

- f_{CLKIN} 为输入时钟 CLKIN 频率；
- $f_{CLKOUTX}$ ：X=A/B/C/D，为 A/B/C/D 通道的输出时钟频率，ODIVX 为 A/B/C/D 通道的输出分频系数；
- f_{IN_ODIVX} ：X=A/B/C/D，为 ODIVX 的输入时钟频率，默认为 f_{VCO}，级联或旁路时按实际电路连接；
- f_{PFD} 为 PFD 鉴相频率，f_{PFD} 最小值不小于 3MHz；
- IDIV、FBDIV、ODIVX 为不同分频器的分频系数，即可通过调整不同分频系数来得到期望频率的时钟信号。
- PLLO 的频率范围请参考 *FPGA 产品数据手册*。

端口示意图

图 5-7 PLLO 端口示意图

端口介绍

表 5-14 PLLO 端口介绍

端口名	I/O	描述
CLKIN	Input	参考时钟输入信号
CLKFB	Input	反馈时钟输入信号
RESET	Input	PLL 全部复位信号，高电平有效。
RESET_P	Input	PLL 关断（Power Down）信号，高电平有效。
RESET_I	Input	带 IDIV 的全复位信号，包括 RESET 功能和 IDIV 的复位，高电平有效。
RESET_S	Input	仅复位 B/C/D 这 3 路，高电平有效。
FBDSEL[5:0]	Input	动态控制 FBDIV 取值，范围 0~63，实际值为 64-FBDSEL。
IDSEL[5:0]	Input	动态控制 IDIV 取值，范围 0~63，实际值为 64-IDSEL。

端口名	I/O	描述
ODSELA[6:0]	Input	动态控制 ODIVA 取值，范围 0~127，实际值为 128-ODSELA。
ODSELB[6:0]	Input	动态控制 ODIVB 取值，范围 0~127，实际值为 128-ODSELB。
ODSELC[6:0]	Input	动态控制 ODIVC 取值，范围 0~127，实际值为 128-ODSELC。
ODSELD[6:0]	Input	动态控制 ODIVD 取值，范围 0~127，实际值为 128-ODSELD。
DTA[3:0]	Input	动态微调控制 CLKOUTA 的占空比
DTB[3:0]	Input	动态微调控制 CLKOUTB 的占空比
ICPSEL[4:0]	Input	动态控制 ICP 电流大小，电流随着取值的增大而增大，值为 0 时电流最小。
LPFRES[2:0]	Input	动态控制 LPFRES 大小，LPFRES 取值范围由小到大，为 R0~R7，R0 对应的带宽最大，R7 对应的带宽最小。
PSSEL[1:0]	Input	动态控制相位移动通道选择。取值 1/2/3 分别选择 B/C/D 通道
PSDIR	Input	动态控制相位移动方向。0：加；1：减
PSPULSE	Input	动态控制相位移动时钟脉冲
ENCLKA	Input	动态控制 A 通道时钟输出使能，若想使用动态使能则同时需静态参数 CLKOUTA_EN = "TRUE"。
ENCLKB	Input	动态控制 B 通道时钟输出使能，若想使用动态使能则同时需静态参数 CLKOUTB_EN = "TRUE"。
ENCLKC	Input	动态控制 C 通道时钟输出使能，若想使用动态使能则同时需静态参数 CLKOUTC_EN = "TRUE"。
ENCLKD	Input	动态控制 D 通道时钟输出使能，若想使用动态使能则同时需静态参数 CLKOUTD_EN = "TRUE"。
CLKOUTA	Output	A 通道时钟输出
CLKOUTB	Output	B 通道时钟输出
CLKOUTC	Output	C 通道时钟输出
CLKOUTD	Output	D 通道时钟输出
LOCK	Output	PLL 锁定指示信号，1 表示锁定，0 表示失锁

参数介绍

表 5-15 PLLO 参数介绍

参数名	取值范围	默认值	描述
FCLKIN	"3"~"400"	"100.0"	参考时钟频率(MHz)
IDIV_SEL	0~63	0	IDIV 分频系数静态设置，对应实际取值为 1~64。

参数名	取值范围	默认值	描述
DYN_IDIV_SEL	"TRUE", "FALSE"	"FALSE"	IDIV 分频系数静态控制参数或动态控制信号选择 ● FALSE: 静态，即选择参数 IDIV_SEL。 ● TRUE: 动态，即选择信号 IDSEL。
FBDIV_SEL	0~63	0	FBDIV 分频系数静态设置，对应实际取值为 1~64
DYN_FBDIV_SEL	"TRUE", "FALSE"	"FALSE"	FBDIV 分频系数静态控制参数或动态控制信号选择 ● FALSE: 静态，即选择参数 FBDIV_SEL。 ● TRUE: 动态，即选择信号 FBDSEL。
ODIVA_SEL	1~128	4	ODIVA 分频系数静态设置
DYN_ODIVA_SEL	"TRUE", "FALSE"	"FALSE"	ODIVA 分频系数静态控制参数或动态控制信号选择 ● FALSE: 静态，即选择参数 ODIVA_SEL。 ● TRUE: 动态，即选择信号 ODSELA。
ODIVB_SEL	1~128	4	ODIVB 分频系数静态设置
DYN_ODIVB_SEL	"TRUE", "FALSE"	"FALSE"	ODIVB 分频系数静态控制参数或动态控制信号选择 ● FALSE: 静态，即选择参数 ODIVB_SEL。 ● TRUE: 动态，即选择信号 ODSELB。
ODIVC_SEL	1~128	4	ODIVC 分频系数静态设置
DYN_ODIVC_SEL	"TRUE", "FALSE"	"FALSE"	ODIVC 分频系数静态控制参数或动态控制信号选择 ● FALSE: 静态，即选择参数 ODIVC_SEL。 ● TRUE: 动态，即选择信号 ODSELC。
ODIVD_SEL	1~128	4	ODIVD 分频系数静态设置
DYN_ODIVD_SEL	"TRUE", "FALSE"	"FALSE”	ODIVD 分频系数静态控制参数或动态控制信号选择 ● FALSE: 静态，即选择参数 ODIVD_SEL。 ● TRUE: 动态，即选择信号 ODSELD。
CLKOUTA_EN	"TRUE", "FALSE"	"TRUE"	A 通道时钟输出使能

参数名	取值范围	默认值	描述
CLKOUTB_EN	"TRUE”, "FALSE"	"TRUE"	B 通道时钟输出使能
CLKOUTC_EN	"TRUE", "FALSE"	"TRUE"	C 通道时钟输出使能
CLKOUTD_EN	"TRUE", "FALSE”	"TRUE"	D 通道时钟输出使能
DYN_DTA_SEL	"TRUE", "FALSE"	"FALSE"	A 通道占空比微调静态控制参数或动态控制信号选择 ● FALSE: 静态，即选择参数 CLKOUTA_DT_DIR & CLKOUTA_DT_STEP。 ● TRUE: 动态，即选择信号 DTA。
DYN_DTB_SEL	"TRUE", "FALSE"	"FALSE"	B 通道占空比微调静态控制参数或动态控制信号选择 ● FALSE: 静态，即选择参数 CLKOUTB_DT_DIR & CLKOUTB_DT_STEP。 ● TRUE: 动态，即选择信号 DTB。
CLKOUTA_DT_DIR	1'b1, 1'b0	1'b1	A 通道占空比静态微调方向 ● 1'b1: + 占空比增加，以上升沿对齐为基准。 ● 1'b0: - 占空比减少，以下降沿对齐为基准。
CLKOUTB_DT_DIR	1'b1, 1'b0	1'b1	B 通道占空比静态微调方向 ● 1'b1: + 占空比增加，以上升沿对齐为基准，调整下降沿。 ● 1'b0: - 占空比减少，以下降沿对齐为基准，调整上升沿。
CLKOUTA_DT_STEP	0,1,2,4	0	A 通道占空比静态微调步长，每步 50ps。
CLKOUTB_DT_STEP	0,1,2,4	0	B 通道占空比静态微调步长，每步 50ps。
CLKA_IN_SEL	2'b00,2'b01,2'b11	2'b00	ODIVA 输入时钟来源选择 ● 2'b00/2'b01: 来自 VCO 输出 ● 2'b11: 旁路来自 CLKIN
CLKA_OUT_SEL	1'b0, 1'b1	1'b0	A 通道输出时钟来源选择 ● 1'b0: 来自 ODIVA 的输出 ● 1'b1: 输出时钟旁路来自 CLKIN
CLKB_IN_SEL	2'b00,2'b01,2'b10, 2'b11	2'b00	ODIVB 输入时钟来源选择 2'b00/2'b01: 来自 VCO 输出 ● 2'b10: 级联来自 CLKCAS_A ● 2'b11: 旁路来自 CLKIN
CLKB_OUT_SEL	1'b0, 1'b1	1'b0	B 通道输出时钟来源选择 ● 1'b0：来自 ODIVB 的输出

参数名	取值范围	默认值	描述
			● 1'b1：输出时钟旁路来自 CLKIN
CLKC_IN_SEL	2'b00,2'b01,2'b10,2'b11	2'b00	ODIVC 输入时钟来源选择 ● 2'b00/2'b01：来自 VCO 输出 ● 2'b10：级联来自 CLKCAS_B ● 2'b11：旁路来自 CLKIN
CLKC_OUT_SEL	1'b0, 1'b1	1'b0	C 通道输出时钟来源选择 ● 1'b0: 来自 ODIVC 的输出 ● 1'b1: 输出时钟旁路来自 CLKIN
CLKD_IN_SEL	2'b00,2'b01,2'b10,2'b11	2'b00	ODIVD 输入时钟来源选择 ● 2'b00/2'b01: 来自 VCO 输出 ● 2'b10: 级联来自 CLKCAS_C ● 2'b11: 旁路来自 CLKIN
CLKD_OUT_SEL	1'b0, 1'b1	1'b0	D 通道输出时钟来源选择 ● 1'b0: 来自 ODIVD 的输出 ● 1'b1: 输出时钟旁路来自 CLKIN
CLKFB_SEL	"INTERNAL", "EXTERNAL"	"INTERNAL”	CLKFB 来源选择 ● INTERNAL: 来自内部 CLKOUTA 反馈 ● EXTERNAL: 来自外部信号反馈
DYN_DPA_EN	"TRUE", "FALSE"	"FALSE"	动态相移调整使能
DYN_PSB_SEL	"TRUE", "FALSE"	"FALSE"	B 通道相位调整静态控制参数或动态控制信号选择 ● FALSE: 静态，即选择参数 PSB_COARSE & PSB_FINE 。TRUE。 ● TRUE:动态，即选择 DPA 动态信号（PSSEL& PSDIR& PSPULSE）来实现，同时需 DYN_DPA_EN="TRUE"。
DYN_PSC_SEL	"TRUE”, "FALSE"	"FALSE"	C 通道相位调整静态控制参数或动态控制信号选择 ● FALSE: 静态，即选择参数 PSC_COARSE & PSC_FINE TRUE。 ● TRUE: 动态，即选择 DPA 动态信号（PSSEL& PSDIR& PSPULSE）来实现，同时需 DYN_DPA_EN="TRUE"。
DYN_PSD_SEL	"TRUE”, "FALSE"	"FALSE"	D 通道相位调整静态控制参数或动态控制信号选择 ● FALSE: 静态，即选择参数 PSD_COARSE & PSD_FINE TRUE。

参数名	取值范围	默认值	描述
			● TRUE:动态，即选择 DPA 动态信号（PSSEL& PSDIR& PSPULSE）来实现，同时需 DYN_DPA_EN="TRUE"。
PSB_COARSE	1~128	1	B 通道相移粗调静态设置
PSB_FINE	0~7	0	B 通道相移微调静态设置
PSC_COARSE	1~128	1	C 通道相移粗调静态设置
PSC_FINE	0~7	0	C 通道相移微调静态设置
PSD_COARSE	1~128	1	D 通道相移粗调静态设置
PSD_FINE	0~7	0	D 通道相移微调静态设置
DTMS_ENB	"TRUE", "FALSE"	"FALSE"	B 通道（ODIVB=2~128）占空比调整使能 ● FALSE: 50%占空比 ● TRUE: DYN_PSB_SEL="TRUE" 时设置 PSB_COARSE& PSB_FINE 作为 falling edge，结合动态相位调整作为 rising edge 实现动态占空比调整（falling edge - rising edge）
DTMS_ENC	"TRUE", "FALSE"	"FALSE"	C 通道（ODIVC=2~128）占空比调整使能 ● FALSE: 50%占空比 ● TRUE: DYN_PSC_SEL="TRUE" 时设置 PSC_COARSE& PSC_FINE 作为 falling edge，结合动态相位调整作为 rising edge 实现动态占空比调整（falling edge - rising edge）
DTMS_END	"TRUE", "FALSE"	"FALSE"	D 通道（ODIVD=2~128）占空比调整使能 ● FALSE: 50%占空比 ● TRUE: DYN_PSD_SEL="TRUE" 时设置 PSD_COARSE& PSD_FINE 作为 falling edge，结合动态相位调整作为 rising edge 实现动态占空比调整（falling edge - rising edge）
RESET_I_EN	"TRUE", "FALSE"	"FALSE"	使能动态信号 RESET_I，若需要使用 RESET_I 端口，需将该参数设为 TRUE。
RESET_S_EN	"TRUE", "FALSE"	"FALSE"	使能动态信号 RESET_S，若需要使用 RESET_S 端口，需将该参数设为 TRUE。
DYN_ICP_SEL	"TRUE", "FALSE"	"FALSE"	ICPSEL 静态控制参数或动态控制信号选择

参数名	取值范围	默认值	描述
			● FALSE: 静态，即选择参数 ICP_SEL。 ● TRUE: 动态，即选择动态信号 ICPSEL。
ICP_SEL	5'bXXXXX, 5'b00000~5'b11111	5'bXXXXX	ICP 电流静态设置 ● 5'bXXXXX: 表示软件会自动计算并设置该参数 ● 5'b00000~5'b11111: 用户若需自行设置，可根据需要在参数范围内设置
DYN_RES_SEL	"TRUE", "FALSE"	"FALSE"	LPRREF 静态控制参数或动态控制信号选择 ● FALSE: 静态，即选择参数 LPR_REF。 ● TRUE: 动态，即选择动态信号 LPFRES。
LPR_REF	7'bXXXXXXX, 7'b0000000(R0),7'b0000001(R1),7'b0000010(R2),7'b0000100(R3),7'b0001000(R4),7'b0010000(R5),7'b0100000(R6),7'b1000000(R7)	7'bXXXXXXX	LPRRES 静态设置 ● 7'bXXXXXXX: 软件会自动计算并设置该参数 ● 7'b0000000~7'b1000000（其中 8 个取值）: 用户若需自行设置，可根据需要在对应这八个值内选择设置。

表 5-16 IDSEL 端口参数对照表

IDSEL[5:0]	IDIV 静态参数值	IDIV 实际值
111111	0	1
111110	1	2
111101	2	3
111100	3	4
111011	4	5
111010	5	6
111001	6	7
111000	7	8
110111	8	9

IDSEL[5:0]	IDIV 静态参数值	IDIV 实际值
……	……	……
000000	63	64

表 5-17 FBDSEL 端口参数对照表

FBDSEL [5:0]	FBDIV 静态参数值	FBDIV 实际值
111111	0	1
111110	1	2
111101	2	3
111100	3	4
111011	4	5
111010	5	6
111001	6	7
111000	7	8
110111	8	9
……	……	……
000000	63	64

表 5-18 ODSELX（X=A/B/C/D）端口参数对照表

ODSELX [6:0]	ODIVX 静态参数值	ODIVX 实际值
1111111	1	1
1111110	2	2
1111101	3	3
1111100	4	4
1111011	5	5
1111010	6	6
1111001	7	7
1111000	8	8
1110111	9	9
……	……	……
0000000	128	128

相位调整

PLLO 支持相位调整，分为静态相位调整和动态相位调整两种方式，其中动态相位调整只 B/C/D 通道支持。静态相位调整通过设置参数 PSX_COARSE 和 PSX_FINE（X=B/C/D）来实现。动态相位调整通过信号 PSSEL、PSDIR、PSPULSE 来实现，PSSEL 用于控制选择通道，

PSDIR 用来控制加或减操作，一个 PSPULSE 脉冲下降沿 DYN_FINE 加/减 1，DYN_FINE 上溢或下溢时 DYN_COARSE 加 1 或减 1 操作，其中 DYN_COARSE 的值小于等于 ODIV。

相位调整可根据下面公式来配置计算（以 B 通道为例），

COARSE_B<ODIVB 时，ps = (FINE_B/8 + COARSE_B)/ODIVB*360

COARSE_B=ODIVB 时，ps = (FINE_B/8)/ODIVB*360

注！

- DYN_FINE 和 DYN_COARSE 是由 DPA 产生的内部信号，通过 PSSEL、PSDIR、PSPULSE 配合产生；
- FINE_B 为通过 DYN_PSB_SEL 选择的动态 DYN_FINE_B 或静态参数 PSB_FINE，COARSE_B 为通过 DYN_PSB_SEL 选择的动态 DYN_COARSE_B 或静态参数 PSB_COARSE；
- 在 CLKX_IN_SEL(X=B/C/D) 选择旁路或者级联时，FINE_X（X=B/C/D）需设为 0。

占空比调整

PLLO 动态占空比调整只 B/C/D 通道支持。占空比定义如下，

$$\text{Duty cycle} = (\text{falling edge} - \text{rising edge}) / \text{cycle_period}$$

其中 falling edge 的位置是由静态相移设置决定，定义为 DUTY，rising edge 的位置是由动态相移设置的 PHASE 决定，DYN_FINE 和 DYN_COARSE 是由 DPA 产生的内部信号，请参考相位调整部分相关描述。DUTY 和 PHASE 的计算公式如下（以 B 通道为例）：

$$\text{DUTY} = (\text{PSB_FINE}/8 + \text{PSB_COARSE})$$

$$\text{PHASE} = (\text{DYN_FINEB}/8 + \text{DYN_COARSEB})$$

动态占空比计算：

- 若 DUTY> PHASE 时，DutyCycle= (DUTY- PHASE) / ODIVB。
- 若 DUTY< PHASE 时，DutyCycle= (DUTY- PHASE) / ODIVB + 1。

注！

- ODIV=1 时不支持动态占空比调整，占空比为 50%；
- ODIV>=2 时，DUTY- PHASE 不支持（-0.5，0.5)之间的值；
- 在 CLKX_IN_SEL(X=B/C/D)选择旁路或者级联时，若 ODIV(>2)为奇数则占空比不是 50%（高电平<低电平，即小于 50%）。

占空比微调

PLLO 的 A/B 通道支持占空比微调，设置占空比微调方向和步长来实现，支持静态和动态两种方式。

1. 静态：

- 微调方向，受参数 CLKOUTA_DT_DIR/CLKOUTB_DT_DIR 控制；
- 微调步长，受参数 CLKOUTA_DT_STEP/CLKOUTB_DT_STEP 控制。

2. 动态：

- 微调方向，受端口 DTA[3]/DTB[3]控制；
- 微调步长，受端口 7-DTA[2:0]/7-DTB[2:0]控制。

微调方向为 1'b1 调节下降沿延时，占空比增加；微调方向为 1'b0 时，调节上升沿延时，占空比减小。

表 5-19 PLLO 占空比微调对照表

占空比微调方向	占空比微调步长	占空比微调延时值
1'b0	0	0
	1	-50ps
	2	-100ps
	4	-200ps
1'b1	0	0
	1	+50ps
	2	+100ps
	4	+200ps

A、B 通道输出相同频率时钟，对 B 通道时钟进行占空比微调，以 A 通道时钟为参考，具体时序如图 5-8 和图 5-9 所示。

图 5-8 B 通道占空比微调时序图(微调方向为 1'b1，步长为 1)

图 5-9 B 通道占空比微调时序图(微调方向为 1'b0，步长为 1)

ICPSEL/LPFRES 设置

PLLO 支持 ICPSEL 和 LPFRES 的设置，包括静态和动态。动态时用户可根据实际需要设置，静态时默认为 X，高云软件会自动计算并配置，若需设置该参数可根据需要在参数范围内配置。

ICPSEL 的取值范围由小到大线性增加，可以划分为 ICP1，ICP2，……ICPN…… ICP31，ICP32，一共 32 档。ICP1 对应最小的电流，ICP32 对应最大的电流；ICP 的取值，定性的可以认为 N 越大则 ICP 越大，N 越小 ICP 越小。

LPRRES 取值范围由小到大，为 R0，R1，R2，R3，R4，R5，R6，R7。R0 对应的带宽最大，R7 对应的带宽最小。现给出几个典型值：R7->250KHz，R4->1.6MHz，R1->12MHz。

原语例化

可以直接实例化原语，也可以通过 IP Core Generator 工具产生。

Verilog 例化：

```
PLLO pllo_inst (
    .LOCK(lock),
    .CLKOUTA(clkouta),
    .CLKOUTB(clkoutb),
    .CLKOUTC(clkoutc),
    .CLKOUTD(clkoutd),
    .CLKIN(clkin),
    .CLKFB(clkfb),
    .RESET(reset),
    .RESET_P(reset_p),
    .RESET_I(reset_i),
    .RESET_S(reset_s),
    .FBDSEL(fbdsel),
    .IDSEL(idsel),
    .ODSELA(odsela),
    .ODSELB(odselb),
    .ODSELC(odselc),
    .ODSELD(odseld),
    .DTA(dta),
    .DTB(dtb),
    .ICPSEL(icpsel),
    .LPFRES(lpfres),
    .PSSEL(pssel),
    .PSDIR(psdir),
    .PSPULSE(pspulse),
    .ENCLKA(enclka),
    .ENCLKB(enclkb),
    .ENCLKC(enclkc),
    .ENCLKD(enclkd)
);

defparam pllo_inst.FCLKIN = "100";
```

```
defparam pllo_inst.DYN_IDIV_SEL = "FALSE";
defparam pllo_inst.IDIV_SEL = 0;
defparam pllo_inst.DYN_FBDIV_SEL = "FALSE";
defparam pllo_inst.FBDIV_SEL = 0;
defparam pllo_inst.DYN_ODIVA_SEL = "FALSE";
defparam pllo_inst.ODIVA_SEL = 4;
defparam pllo_inst.DYN_ODIVB_SEL = "FALSE";
defparam pllo_inst.ODIVB_SEL = 4;
defparam pllo_inst.DYN_ODIVC_SEL = "FALSE";
defparam pllo_inst.ODIVC_SEL = 4;
defparam pllo_inst.DYN_ODIVD_SEL = "FALSE";
defparam pllo_inst.ODIVD_SEL = 4;
defparam pllo_inst.CLKOUTA_EN = "TRUE";
defparam pllo_inst.CLKOUTB_EN = "FALSE";
defparam pllo_inst.CLKOUTC_EN = "FALSE";
defparam pllo_inst.CLKOUTD_EN = "FALSE";
defparam pllo_inst.DYN_DTA_SEL = "FALSE";
defparam pllo_inst.DYN_DTB_SEL = "FALSE";
defparam pllo_inst.CLKOUTA_DT_DIR = 1'b1;
defparam pllo_inst.CLKOUTB_DT_DIR = 1'b1;
defparam pllo_inst.CLKOUTA_DT_STEP = 0;
defparam pllo_inst.CLKOUTB_DT_STEP = 0;
defparam pllo_inst.CLKA_IN_SEL = 2'b00;
defparam pllo_inst.CLKA_OUT_SEL = 1'b0;
defparam pllo_inst.CLKB_IN_SEL = 2'b00;
defparam pllo_inst.CLKB_OUT_SEL = 1'b0;
defparam pllo_inst.CLKC_IN_SEL = 2'b00;
defparam pllo_inst.CLKC_OUT_SEL = 1'b0;
defparam pllo_inst.CLKD_IN_SEL = 2'b00;
defparam pllo_inst.CLKD_OUT_SEL = 1'b0;
defparam pllo_inst.CLKFB_SEL = "INTERNAL";
defparam pllo_inst.DYN_DPA_EN = "FALSE";
defparam pllo_inst.DYN_PSB_SEL = "FALSE";
defparam pllo_inst.DYN_PSC_SEL = "FALSE";
defparam pllo_inst.DYN_PSD_SEL = "FALSE";
```

```
defparam pllo_inst.PSB_COARSE = 1;
defparam pllo_inst.PSB_FINE = 0;
defparam pllo_inst.PSC_COARSE = 1;
defparam pllo_inst.PSC_FINE = 0;
defparam pllo_inst.PSD_COARSE = 1;
defparam pllo_inst.PSD_FINE = 0;
defparam pllo_inst.DTMS_ENB = "FALSE";
defparam pllo_inst.DTMS_ENC = "FALSE";
defparam pllo_inst.DTMS_END = "FALSE";
defparam pllo_inst.RESET_I_EN = "FALSE";
defparam pllo_inst.RESET_S_EN = "FALSE";
defparam pllo_inst.DYN_ICP_SEL = "FALSE";
defparam pllo_inst.ICP_SEL = 5'bXXXXX;
defparam pllo_inst.DYN_RES_SEL = "FALSE";
defparam pllo_inst.LPR_REF = 7'bXXXXXXX;
```

VHDL 例化：

```
COMPONENT PLLO
    GENERIC (
        FCLKIN : STRING := "100.0";
        DYN_IDIV_SEL : STRING := "FALSE";
        IDIV_SEL : integer := 0;
        DYN_FBDIV_SEL : STRING := "FALSE";
        FBDIV_SEL : integer := 0;
        DYN_ODIVA_SEL : STRING := "FALSE";
        ODIVA_SEL : integer := 4;
        DYN_ODIVB_SEL : STRING := "FALSE";
        ODIVB_SEL : integer := 4;
        DYN_ODIVC_SEL : STRING := "FALSE";
        ODIVC_SEL : integer := 4;
        DYN_ODIVD_SEL : STRING := "FALSE";
        ODIVD_SEL : integer := 4;
        CLKOUTA_EN : STRING := "TRUE";
        CLKOUTB_EN : STRING := "TRUE";
        CLKOUTC_EN : STRING := "TRUE";
        CLKOUTD_EN : STRING := "TRUE";
```

```
DYN_DTA_SEL : STRING := "FALSE";
DYN_DTB_SEL : STRING := "FALSE";
CLKOUTA_DT_DIR : bit := '1';
CLKOUTB_DT_DIR : bit := '1';
CLKOUTA_DT_STEP : integer := 0;
CLKOUTB_DT_STEP : integer := 0;
CLKA_IN_SEL    : bit_vector := "00";
CLKA_OUT_SEL : bit := '0';
CLKB_IN_SEL    : bit_vector := "00";
CLKB_OUT_SEL : bit := '0';
CLKC_IN_SEL    : bit_vector := "00";
CLKC_OUT_SEL : bit := '0';
CLKD_IN_SEL    : bit_vector := "00";
CLKD_OUT_SEL : bit := '0';
CLKFB_SEL : STRING := "INTERNAL";
DYN_DPA_EN : STRING := "FALSE";
DYN_PSB_SEL : STRING := "FALSE";
DYN_PSC_SEL : STRING := "FALSE";
DYN_PSD_SEL : STRING := "FALSE";
PSB_COARSE : integer := 1;
PSB_FINE : integer := 0;
PSC_COARSE : integer := 1;
PSC_FINE : integer := 0;
PSD_COARSE : integer := 1;
PSD_FINE : integer := 0;
DTMS_ENB : STRING := "FALSE";
DTMS_ENC : STRING := "FALSE";
DTMS_END : STRING := "FALSE";
RESET_I_EN : STRING := "FALSE";
RESET_S_EN : STRING := "FALSE";
DYN_ICP_SEL : STRING := "FALSE";
ICP_SEL : std_logic_vector(4 downto 0) := "XXXXX";
DYN_RES_SEL : STRING := "FALSE";
LPR_REF : std_logic_vector(6 downto 0) := "XXXXXXX"
```

```
            );
                PORT (
                    CLKIN : IN std_logic;
                    CLKFB : IN std_logic:='0';
                    RESET,RESET_P : IN std_logic:='0';
                    RESET_I,RESET_S : IN std_logic:='0';
                    IDSEL,FBDSEL : IN std_logic_vector(5 downto 0);
                    ODSELA, ODSELB, ODSELC, ODSELD : IN
std_logic_vector(6 downto 0);
                    DTA, DTB : IN std_logic_vector(3 downto 0);
                    ICPSEL : IN std_logic_vector(4 downto 0);
                    LPFRES : IN std_logic_vector(2 downto 0);
                    PSSEL : IN std_logic_vector(1 downto 0);
                    PSDIR,PSPULSE : IN std_logic;
                    ENCLKA,ENCLKB,ENCLKC,ENCLKD : IN std_logic;
                    LOCK : OUT std_logic;
                    CLKOUTA : OUT std_logic;
                    CLKOUTB : OUT std_logic;
                    CLKOUTC : OUT std_logic;
                    CLKOUTD : OUT std_logic
                );
        END COMPONENT;
        uut:PLLO
            GENERIC MAP(
                    FCLKIN : STRING => "100.0";
                    DYN_IDIV_SEL =>"FALSE";
                    IDIV_SEL => 0;
                    DYN_FBDIV_SEL=> "FALSE";
                    FBDIV_SEL => 0;
                    DYN_ODIVA_SEL =>"FALSE";
                    ODIVA_SEL => 4;
                    DYN_ODIVB_SEL=> "FALSE";
                    ODIVB_SEL => 4;
                    DYN_ODIVC_SEL => "FALSE";
                    ODIVC_SEL => 4;
```

```
DYN_ODIVD_SEL=> "FALSE";
ODIVD_SEL => 4;
CLKOUTA_EN => "TRUE";
CLKOUTB_EN => "TRUE";
CLKOUTC_EN => "TRUE";
CLKOUTD_EN =>"TRUE";
DYN_DTA_SEL =>"FALSE";
DYN_DTB_SEL =>"FALSE";
CLKOUTA_DT_DIR => '1';
CLKOUTB_DT_DIR => '1';
CLKOUTA_DT_STEP => 0;
CLKOUTB_DT_STEP => 0;
CLKA_IN_SEL => "00";
CLKA_OUT_SEL => '0';
CLKB_IN_SEL => "00";
CLKB_OUT_SEL => '0';
CLKC_IN_SEL => "00";
CLKC_OUT_SEL => '0';
CLKD_IN_SEL => "00";
CLKD_OUT_SEL => '0';
CLKFB_SEL => "INTERNAL";
DYN_DPA_EN => "FALSE";
DYN_PSB_SEL => "FALSE";
DYN_PSC_SEL => "FALSE";
DYN_PSD_SEL => "FALSE";
PSA_COARSE => 0;
PSA_FINE => 0;
PSB_COARSE => 0;
PSB_FINE => 0;
PSC_COARSE => 0;
PSC_FINE => 0;
PSD_COARSE => 0;
PSD_FINE => 0;
DTMS_ENB => "FALSE";
DTMS_ENC => "FALSE";
```

```
                DTMS_END => "FALSE";
                RESET_I_EN => "FALSE";
                RESET_S_EN => "FALSE";
                DYN_ICP_SEL => "FALSE";
                ICP_SEL => "XXXXX";
                DYN_RES_SEL => "FALSE";
                LPR_REF => "XXXXXXX"
        )
         PORT MAP(
                LOCK=>lock,
                CLKOUTA=> clkouta,
                CLKOUTB=>clkoutb,
                CLKOUTC=>clkoutc,
                CLKOUTD=>clkoutd,
                CLKIN=>clkin,
                CLKFB=>clkfb,
                RESET=>reset,
                RESET_P=>reset_p,
                RESET_I=>reset_i,
                RESET_S=>reset_s,
                FBDSEL=>fbdsel,
                IDSEL=>idsel,
                ODSELA=>odsela,
                ODSELB=>odselb,
                ODSELC=>odselc,
                ODSELD=>odseld,
                DTA=>dta,
                DTB=>dtb,
                ICPSEL=>icpsel,
                LPFRES=>lpfres,
                PSSEL=>pssel,
                PSDIR=>psdir,
                PSPULSE=>pspulse,
                ENCLKA=>enclka,
                ENCLKB=>enclkb,
```

```
                    ENCLKC=>enclkc,
                    ENCLKD=>enclkd
            );
```

5.3.2 IP 调用

在 IP Core Generator 界面中，单击“PLLO”，界面右侧会显示 PLLO 的相关信息概要。

IP 配置

在 IP Core Generator 界面中双击“PLLO”，弹出 PLLO 的“IP Customization”窗口。该窗口包括“General”配置框、“Options”配置框和端口显示框图，如图 5-10 所示。

图 5-10 PLLO 的 IP Customization 窗口结构

1. General 配置框
   General 配置框用于配置产生的 IP 设计文件的相关信息。PLLO 的 General 配置框的使用和 DQCE 模块的类似，请参考 DQCE 中的 General 配置框。

2. Options 配置框
   Options 配置框用于用户自定义配置 IP，Options 配置框如图 5-10 所示。

   - General：配置 IP Core 配置的模式，支持一般模式“General Mode”和高级模式“Advanced Mode”。一般模式下输入输入时钟频率和输出时钟频率，软件会自动计算不同的分频系数；高级模式

适用于高级用户，允许输入输入频率和不同分频系数得到预期的输出频率；

- Optional Port：配置输出相位和占空比调整的动态、静态模式和使能 PLLO 输出时钟。
  - “PLL Phase And Duty Cycle Adjustment”选项配置输出的占空比和相位调整的模式，支持动态调整“Dynamic”和静态调整“Static”；
  - “Clock Enable Ports”选项配置使能 PLLO 的输出时钟的端口；
- CLKIN：配置 PLLO 输入时钟的频率，分频参数的设置。
  - “Clock Frequency（频率范围）”配置输入时钟的频率，范围为 3~400MHz；
  - “Divide Factor”可在高级模式下配置分频参数，支持动态模式“Dynamic”和静态模式“Static”，静态模式下可配置分频参数的具体数值，范围为 1~64。若 CLKOUT 的输出频率不在相应 device 要求的范围内，单击“Calculate”或“OK”，会弹出提示窗口提示错误；若 CLKIN/IDIV 的频率不在相应 device 要求的 Clock Frequency 范围内，单击“Calculate”或“OK”，会弹出提示窗口提示错误；
  - “VCO Frequency”为计算得到的 VCO 的频率，只读。
- CLKFB：配置 PLLO 反馈时钟的源和倍频参数。
  - 配置反馈时钟的源时，“Source”选项可选择 Internal 和 External；
  - “Divide Factor”可在高级模式下配置倍频参数，支持动态模式“Dynamic”和静态模式“Static”，静态模式下可配置倍频参数的具体数值，范围为 1~64，配置不合理时，单击“Calculate”按钮或“OK”按钮，会弹出提示窗口提示错误。
- ICP and LPF
  - ICPSEL 选项配置 ICP 电流，支持动态调整“Dynamic”和静态调整“Static”，静态模式下可配置 ICP 具体值，范围为 ICP1~ICP32，默认为 X，表示软件会自动计算并配置；
  - LPFRES 选项配置低通滤波电阻，支持动态调整“Dynamic”和静态调整“Static”，静态模式下可配置 RES 具体值，范围为 R0~R7，默认为 X，表示软件会自动计算并配置。
- PLL Reset
  - “PLL Reset”选项配置 PLLO 的 RESET 使能模式；
  - “PLL Power Down”选项配置 RESET_P 端口使 PLLO 处于节电模式；

- “CLKIN Divider Reset”选项配置使能 RESET_I；
- “CLKOUTB/CLKOUTC/CLKOUTD Divider Reset”选项配置使能 RESET_S。

- Enable LOCK：使能 LOCK 端口。
- CLKOUTA：配置 A 通道 PLLO 输出时钟期望频率，配置 VCO 参数，配置输出时钟的微调占空比参数。
  - “Bypass”选项可配置输出时钟的旁路功能；
  - “Enable CLKOUTA Divider”选项可配置 VCO 时钟的旁路功能；
  - “Expected Frequency（频率范围）”在一般模式下配置期望的输出时钟 CLKOUTA 的频率，非 bypass 模式下范围为 3.125M~800M；
  - “Tolerance（%）”配置 CLKOUTA 期望频率和计算出的实际频率的允许误差。
  - “Actual Frequency”显示经计算得出的 CLKOUTA 实际频率，无需用户配置；
  - “VCO Divide Factor”在高级模式下配置 VCO 参数支持动态模式“Dynamic”和静态模式“Static”，静态模式下可配置分频参数的具体数值，范围为 1~128，配置不合理时，单击“Calculate”或“OK”，会弹出提示窗口提示错误。
  - “Duty Trim”配置微调占空比，支持动态模式“Dynamic”和静态模式“Static”，静态模式下分“Rising”和“Falling”，可配置“Step”的具体数值 0，1，2，4。
- CLKOUTB：配置 B 通道 PLLO 输出时钟期望频率，配置 VCO 参数，配置输出时钟的微调占空比参数，配置相位和占空比调整参数。
  - “Bypass”选项可配置输出时钟的旁路功能；
  - “Enable CLKOUTB Divider”选项可配置 VCO 时钟的旁路功能；
  - “Expected Frequency（频率范围）”在一般模式下配置期望的输出时钟 CLKOUTB 的频率，非 bypass 模式下范围为 3.125M~800M；
  - “Tolerance（%）”配置 CLKOUTB 期望频率和计算出的实际频率的允许误差。
  - “Actual Frequency”显示经计算得出的 CLKOUTB 实际频率，无需用户配置；
  - “VCO Divide Factor”在高级模式下配置 VCO 参数支持动态模式“Dynamic”和静态模式“Static”，静态模式下可配置分

频参数的具体数值，范围为 1~128，配置不合理时，单击“Calculate”或“OK”，会弹出提示窗口提示错误；

- “Duty Trim”配置微调占空比，支持动态模式“Dynamic”和静态模式“Static”，静态模式下分“Rising”和“Falling”，可配置“Step”的具体数值 0，1，2，4；
- “Phase（degree）”配置调整的相位度数，支持动态模式“Dynamic”和静态模式“Static”，静态模式下配置相位度数；
- “Duty Cycle”配置占空比，支持动态模式“Dynamic”和静态模式“Static”，静态模式下为 50%，动态占空比调整需配置相位结合动态 DPA 调整来实现。

- CLKOUTC：配置 C 通道 PLLO 输出时钟期望频率，配置 VCO 参数，配置相位和占空比调整参数。
  - “Bypass”选项可配置输出时钟的旁路功能；
  - “Enable CLKOUTC Divider”选项可配置 VCO 时钟的旁路功能；
  - “Expected Frequency（频率范围）”在一般模式下配置期望的输出时钟 CLKOUTC 的频率，非 bypass 模式下范围为 3.125M~800M；
  - “Tolerance（%）”配置 CLKOUTC 期望频率和计算出的实际频率的允许误差；
  - “Actual Frequency”显示经计算得出的 CLKOUTC 实际频率，无需用户配置；
  - “VCO Divide Factor”在高级模式下配置 VCO 参数支持动态模式“Dynamic”和静态模式“Static”，静态模式下可配置分频参数的具体数值，范围为 1~128，配置不合理时，单击“Calculate”或“OK”，会弹出提示窗口提示错误；
  - “Phase（degree）”配置调整的相位度数，支持动态模式“Dynamic”和静态模式“Static”，静态模式下配置相位度数；
  - “Duty Cycle”配置占空比，支持动态模式“Dynamic”和静态模式“Static”，静态模式下为 50%，动态占空比调整需配置相位结合动态 DPA 调整来实现。
- CLKOUTD：配置 D 通道 PLLO 输出时钟期望频率，配置 VCO 参数，配置相位和占空比调整参数。
  - “Bypass”选项可配置输出时钟的旁路功能；
  - “Enable CLKOUTD Divider”选项可配置 VCO 时钟的旁路功能；

- “Expected Frequency（频率范围）”在一般模式下配置期望的输出时钟 CLKOUTD 的频率，非 bypass 模式下范围为 3.125M~800M；
- “Tolerance（%）”配置 CLKOUTD 期望频率和计算出的实际频率的允许误差。
- “Actual Frequency”显示经计算得出的 CLKOUTD 实际频率，无需用户配置；
- “VCO Divide Factor”在高级模式下配置 VCO 参数支持动态模式“Dynamic“和静态模式“Static”，静态模式下可配置分频参数的具体数值，范围为 1~128，配置不合理时，单击“Calculate”或“OK”，会弹出提示窗口提示错误；
- “Phase（degree）”配置调整的相位度数，支持动态模式“Dynamic”和静态模式“Static”，静态模式下配置相位度数；
- “Duty Cycle”配置占空比，支持动态模式“Dynamic”和静态模式“Static”，静态模式下为 50%，动态占空比调整需配置相位结合动态 DPA 调整来实现。

- Calculate：计算当前配置是否合理。
  - 一般模式“General Mode”下，根据输入输出频率计算配置分频参数、倍频参数和 VCO 参数，计算出的实际频率和期望频率不相等时，单击“Calculate”按钮后会弹出“error”窗口提示错误。
  - 在高级模式“Advanced Mode”下，计算配置的静态分频参数、倍频参数和 VCO 参数是否合理，若不合理，单击“Calculate”，弹出“error”窗口提示错误；若配置正确，单击“Calculate”，弹出“info”窗口提示配置成功。

3. 端口显示框图
   端口显示框图显示 IP Core 的配置结果示例框图，输入输出端口的个数根据 Options 配置实时更新，如图 5-10 所示。

IP 生成文件

IP 窗口配置完成后，产生以配置文件“File Name”命名的三个文件，以默认配置为例进行介绍：

- IP 设计文件“gowin_pllo.v”为完整的 verilog 模块，根据用户的 IP 配置，产生实例化的 PLLO；
- IP 设计使用模板文件 gowin_pllo_tmp.v，为用户提供 IP 设计使用模板文件；
- IP 配置文件：“gowin_pllo.ipc”，用户可加载该文件对 IP 进行配置。

注！

如配置中选择的语言是 VHDL，则产生的前两个文件名后缀为.vhd。

5.4 DLLDLY

5.4.1 原语介绍

DLLDLY 为时钟延时模块，依据 DLLSTEP 信号对输入时钟进行调整，得到该时钟的延时调整输出。

功能描述

DLLDLY 根据 DLLSTEP 产生对应相位的延时，得到基于 CLKIN 的延时输出。根据 DLLSTEP 计算延时时，GW1N(X)系列 FPGA 的单步延时时间约为 30ps，GW2A(X)系列 FPGA 的单步延时时间约为 18ps。

端口示意图

图 5-11 DLLDLY 端口示意图

端口介绍

表 5-20 DLLDLY 端口介绍

端口名	I/O	描述
CLKOUT	Output	时钟输出信号
FLAG	Output	输出标志，用以表示动态调整延时的 under-flow 或 over-flow。
DLLSTEP[7:0]	Input	延时步长输入信号
CLKIN	Input	时钟输入信号
DIR	Input	设置动态调整延时的方向 0: 增加延时； 1: 减少延时
LOADN	Input	控制加载延时步长 0: 加载延时步长 DLLSTEP； 1: 动态调整延时
MOVE	Input	MOVE 为下降沿时动态调整延时，每个脉冲移动一个延时步长。

参数介绍

表 5-21 DLLDLY 参数介绍

参数名	参数类型	取值范围	默认值	描述
DLL_INSEL	Integer	1'b1	1'b1	1'b1: 正常模式，使用 DLLDLY 延时模块。
DLY_SIGN	String	1'b0,1'b1	1'b0	设置调整延时的符号: 1'b0:'+' 1'b1: '-'
DLY_ADJ	Integer	0~255	0	延时调整设置: dly_sign=0 DLY_ADJ; dly_sign=1 -256+ DLY_ADJ

原语例化

可以直接实例化原语，也可以通过 IP Core Generator 工具产生。

Verilog 例化：

```
DLLDLY dlldly_0 (
    .CLKIN(clkin),
    .DLLSTEP(step[7:0]),
    .DIR(dir),
    .LOADN(loadn),
    .MOVE(move),
    .CLKOUT(clkout),
    .FLAG(flag)
);
defparam dlldly_0.DLL_INSEL=1'b1;
defparam dlldly_0.DLY_SIGN=1'b1;
defparam dlldly_0.DLY_ADJ=0;
```

VHDL 例化：

```
COMPONENT DLLDLY
        GENERIC(
                DLL_INSEL:bit:='0';
                DLY_SIGN:bit:='0';
                LY_ADJ:integer:=0
         );
            PORT(
```

```
                    DLLSTEP:IN std_logic_vector(7 downto 0);
                    CLKIN:IN std_logic;
                    DIR,LOADN,MOVE:IN std_logic;
                    CLKOUT:OUT std_logic;
                    FLAG:OUT std_logic
            );
END COMPONENT;
uut:DLLDLY
    GENERIC MAP(
                     DLL_INSEL=>'1',
                     DLY_SIGN=>'0',
                     LY_ADJ=>0
     )
    PORT MAP(
        DLLSTEP=>step,
        CLKIN=>clkin,
        DIR=>dir,
        LOADN=>loadn,
        MOVE=>move,
        CLKOUT=>clkout,
        FLAG=>flag
    );
```

5.4.2 IP 调用

在 IP Core Generator 界面中单击 DLLDLY，界面右侧会显示 DLLDLY 的相关信息概要。

IP 配置

在 IP Core Generator 界面中，双击“DLLDLY”，弹出 DLLDLY 的“IP Customization”窗口，该窗口包括“General”配置框、“Options”配置框和端口显示框图，如图 5-12 所示

图 5-12 DLLDLY 的 IP Customization 窗口结构

1. General 配置框
   General 配置框用于配置产生的 IP 设计文件的相关信息。DLLDLY 的 General 配置框的使用和 DQCE 模块的类似，请参考 DQCE 中的 General 配置框。
2. Options 配置框
   Options 配置框用于用户自定义配置 IP，Options 配置框如图 5-12 所示。
   - Delay Sign：设置调整延时的符号。
   - Adjustment Scale：延时调整设置。
3. 端口显示框图
   端口显示框图显示 IP Core 的配置结果示例框图，如图 5-12 所示。

IP 生成文件

IP 窗口配置完成后，产生以配置文件"File Name”命名的三个文件，以默认配置为例进行介绍：

- IP 设计文件“gowin_dlldly.v”为完整的 verilog 模块，根据用户的 IP 配置，产生实例化的 DLLDLY；
- IP 设计使用模板文件 gowin_dlldly_tmp.v，为用户提供 IP 设计使用模板文件；
- IP 配置文件：“gowin_dlldly.ipc”，用户可加载该文件对 IP 进行配置。

注！

如配置中选择的语言是 VHDL，则产生的前两个文件名后缀为.vhd。

5.5 CLKDIV

5.5.1 原语介绍

CLKDIV 为时钟分频器，实现时钟频率调整。

功能描述

CLKDIV 为高速时钟分频模块，生成和输入时钟相位一致的分频时钟，用于 IO 逻辑。在器件 GW1N-1S、GW1NS-4、GW1NS-4C、GW1NSR-4、GW1NSR-4C、GW1NSER-4C、GW1N-9、GW1N-9C、GW1NR-9、GW1NR-9C、GW1N-2、GW1N-1P5、GW1N-2B、GW1N-1P5B、GW1NR-2 和 GW1NR-2B 下支持 2/3.5/4/5/8 分频，其他器件下支持 2/3.5/4/5 分频。

端口示意图

图 5-13 CLKDIV 端口示意图

端口介绍

表 5-22 CLKDIV 端口介绍

端口名	I/O	描述
HCLKIN	Input	时钟输入信号
RESETN	Input	异步复位信号，低电平有效。
CALIB	Input	CALIB 输入信号，调整输出时钟。
CLKOUT	Output	时钟输出信号

其中，CALIB 信号可以和 IOLOGIC 中的 CALIB 配合使用，具体功能如下：

- 2 分频时，每 2 个下降沿调整一次相位，每次调整 180 度，2 次调整为一个周期；
- 3.5 分频时，每 1 个下降沿调整一次相位，每次调整约 102.8 度，7 次调整为一个周期；
- 4 分频时，每 2 个下降沿调整一次相位，每次调整 90 度，4 次调整为一个周期；
- 5 分频时，每 2 个下降沿调整一次相位，每次调整约 72 度，5 次调整为一个周期；
- 8 分频时，每 2 个下降沿调整一次相位，每次调整约 45 度，8 次调整为

一个周期。

参数介绍

表 5-23 CLKDIV 参数介绍

参数名	取值范围	默认值	描述
DIV_MODE	2, 3.5, 4, 5 (8)	2	设置时钟分频系数
GSREN	"false", "true"	"false"	启用全局复位 GSR

原语例化

可以直接实例化原语，也可以通过 IP Core Generator 工具产生。

Verilog 例化：

```
CLKDIV clkdiv_inst (
    .HCLKIN(hclkin),
    .RESETN(resetn),
    .CALIB(calib),
    .CLKOUT(clkout)
);
defparam clkdiv_inst.DIV_MODE="3.5";
defparam clkdiv_inst.GSREN="false";
```

VHDL 例化：

```
COMPONENT CLKDIV
        GENERIC(
            DIV_MODE:STRING:="2";
            GSREN:STRING:="false"
            );
        PORT(
            HCLKIN:IN std_logic;
            RESETN:IN std_logic;
            CALIB:IN std_logic;
            CLKOUT:OUT std_logic
        );
END CONPONENT;
    uut:CLKDIV
        GENERIC MAP(
                DIV_MODE=>"2",
                GSREN=>"false"
```

```
                )
        PORT MAP(
                HCLKIN=>hclkin,
                RESETN=>resetn,
                CALIB=>calib,
                CLKOUT=>clkout
        );
```

5.5.2 IP 调用

IP Core Generator 界面中单击 CLKDIV，界面右侧会显示 CLKDIV 的相关信息概要。

IP 配置

在 IP Core Generator 界面中，双击“CLKDIV”，弹出 CLKDIV 的“IP Customization”窗口，该窗口包括“General”配置框、“Options”配置框和端口显示框图，如图 5-14 所示。

图 5-14 CLKDIV 的 IP Customization 窗口结构

1. General 配置框
   General 配置框用于配置产生的 IP 设计文件的相关信息。CLKDIV 的 General 配置框的使用和 DQCE 模块的类似，请参考 DQCE 中的 General 配置框。

2. Options 配置框
   Options 配置框用于用户自定义配置 IP，Options 配置框如图 5-14 所示。
   - Division Factor：除法因子。
   - Calibration：校准时钟使能/失能选项。
3. 端口显示框图
   端口显示框图显示 IP Core 的配置结果示例框图，如图 5-14 所示。

IP 生成文件

IP 窗口配置完成后，产生以配置文件"File Name”命名的三个文件，以默认配置为例进行介绍：

- IP 设计文件“gowin_clkdiv.v”为完整的 verilog 模块，根据用户的 IP 配置，产生实例化的 CLKDIV；
- IP 设计使用模板文件 gowin_clkdiv_tmp.v，为用户提供 IP 设计使用模板文件；
- IP 配置文件：“gowin_clkdiv.ipc”，用户可加载该文件对 IP 进行配置。

注！

如配置中选择的语言是 VHDL，则产生的前两个文件名后缀为.vhd。

5.6 CLKDIVG

5.6.1 原语介绍

CLKDIVG 为时钟分频器，实现时钟频率调整。

适用器件

表 5-24 CLKDIVG 适用器件

家族	系列	器件
小蜜蜂® (LittleBee®)	GW1N	GW1N-2, GW1N-1P5
	GW1NR	GW1NR-2
晨熙® (Arora)	GW2AN	GW2AN-18X, GW2AN-9X

功能描述

CLKDIVG 为时钟分频模块，生成和输入时钟相位一致的分频时钟。CLKDIVG 只有一个，位置固定，输入来自固定 IO，功能和 CLKDIV 一致。

端口示意图

图 5-15 CLKDIVG 端口示意图

端口介绍

表 5-25 CLKDIVG 端口介绍

端口名	I/O	描述
CLKIN	Input	时钟输入信号
RESETN	Input	异步复位信号，低电平有效。
CALIB	Input	CALIB 输入信号，调整输出时钟。
CLKOUT	Output	时钟输出信号

其中，CALIB 信号可以和 IOLOGIC 中的 CALIB 配合使用，具体功能可参考 5.5 CLKDIV。

参数介绍

表 5-26 CLKDIVG 参数介绍

参数名	取值范围	默认值	描述
DIV_MODE	2, 3.5, 4, 5, 8	2	设置时钟分频系数
GSREN	"false”, "true”	"false”	启用全局复位 GSR

原语例化

可以直接实例化原语，也可以通过 IP Core Generator 工具产生。

Verilog 例化：

```
CLKDIVG CLKDIVG_inst (
    .CLKIN(clkin),
    .RESETN(resetn),
    .CALIB(calib),
    .CLKOUT(clkout)
);
defparam CLKDIVG_inst.DIV_MODE="2";
```

```
defparam CLKDIVG_inst.GSREN="false";
```

VHDL 例化：

```
COMPONENT CLKDIVG
        GENERIC(
            DIV_MODE:STRING:="2";
            GSREN:STRING:="false"
            );
         PORT(
            CLKIN:IN std_logic;
            RESETN:IN std_logic;
            CALIB:IN std_logic;
            CLKOUT:OUT std_logic
        );
END CONPONENT;
    uut:CLKDIVG
        GENERIC MAP(
                DIV_MODE=>"2",
                GSREN=>"false"
                )
         PORT MAP(
                CLKIN=>clkin,
                RESETN=>resetn,
                CALIB=>calib,
                CLKOUT=>clkout
         );
```

5.6.2 IP 调用

IP Core Generator 界面中单击 CLKDIVG，界面右侧会显示 CLKDIVG 的相关信息概要。

IP 配置

在 IP Core Generator 界面中，双击“CLKDIVG”，弹出 CLKDIVG 的“IP Customization”窗口，该窗口包括“General”配置框、“Options”配置框和端口显示框图，如图 5-16 所示。

图 5-16 CLKDIVG 的 IP Customization 窗口结构

1. General 配置框
   General 配置框用于配置产生的 IP 设计文件的相关信息。
   CLKDIVG 的 General 配置框的使用和 DQCE 模块的类似，请参考 DQCE 中的 General 配置框。
2. Options 配置框
   Options 配置框用于用户自定义配置 IP，Options 配置框如图 5-16 所示。

- Division Factor：除法因子。
- Calibration：校准时钟使能/失能选项。

3. 端口显示框图
   端口显示框图显示 IP Core 的配置结果示例框图，如图 5-16 所示。

IP 生成文件

IP 窗口配置完成后，产生以配置文件"File Name”命名的三个文件，以默认配置为例进行介绍：

- IP 设计文件“gowin_clkdivg.v”为完整的 verilog 模块，根据用户的 IP 配置，产生实例化的 CLKDIVG；
- IP 设计使用模板文件 gowin_ clkdivg _tmp.v，为用户提供 IP 设计使用模板文件；
- IP 配置文件：“gowin_ clkdivg.ipc”，用户可加载该文件对 IP 进行配置。

注！

如配置中选择的语言是 VHDL，则产生的前两个文件名后缀为.vhd。

5.7 DQS

5.7.1 原语介绍

DQS 是 DDR 存储器接口双向数据选通脉冲电路。

适用器件

表 5-27 DQS 适用器件

家族	系列	器件
晨熙® (Arora)	GW2A	GW2A-18, GW2A-18C, GW2A-55, GW2A-55C
	GW2AN	GW2AN-55C, GW2AN-18X, GW2AN-9X
	GW2AR	GW2AR-18, GW2AR-18C
	GW2ANR	GW2ANR-18C

功能描述

DQS 是内存控制器 IP 的关键器件，主要用于调整 DQSIN 与 DQSR90、DQSW0 与 DQSW270 信号间的相位关系并完成写平衡、读校准。

端口示意图

图 5-17 DQS 端口示意图

端口介绍

表 5-28 DQS 端口介绍

端口名	I/O	描述
DLLSTEP[7:0]	input	DQS 延时步长控制输入
DQSIN	input	DQS 输入，来自 IO PAD。
FCLK	input	快速时钟，可来自两个不同 FCLK 时钟树输出。
HOLD	input	用于 DQS 写入，停止写入相关信号来同步输出时钟；用于 DQS 读取，来复位 FIFO 计数器。
PCLK	input	主时钟，来自 PCLK 时钟树。
RDIR	input	调整 DDR 读取的延时方向 "0” 增加延时 "1” 减少延时
RLOADN	input	将 DDR 读取的最终延时步长复位至初始值，低电平有效。
RMOVE	input	RMOVE 为下降沿时改变 DDR 读取的延时步长，每个脉冲改变一次。
WDIR	input	调整 DDR 写入的延时方向 "0” 增加延时 "1” 减少延时
WLOADN	input	将 DDR 写入的最终延时步长复位至初始值，低电平有效。
WMOVE	input	WMOVE 为下降沿时改变 DDR 写入的延时步长，每个脉冲改变一次
WSTEP[7:0]	input	用于 DDR 写均衡延时控制
READ[3:0]	input	READ 信号，用于 DDR 读模式。
RCLKSEL[2:0]	input	选择读时钟源和极性控制
RESET	input	DQS 复位输入，高电平有效。
RPOINT[2:0]	output	FIFO 控制读指针，作用于 IOLOGIC 的 RADDR，或通过绕线作用于用户逻辑。
WPOINT[2:0]	output	FIFO 控制写指针，作用于 IOLOGIC 的 WADDR，或通过绕线作用于用户逻辑。
DQSW0	output	PCLK/FCLK 0° 相移输出，可作用于 IOLOGIC 的 TCLK，或通过绕线作用于用户逻辑。
DQSW270	output	PCLK/FCLK 270° 相移输出，可作用于 IOLOGIC 的 TCLK，或通过绕线作用于用户逻辑。
DQSR90	output	DQSI 相移 90° 输出，可作用于 IOLOGIC 的 ICLK，或通过绕线作用于用户逻辑。
RFLAG	output	READ 延时调整输出标志，用以表示读取延时调整 under-flow 或 over-flow。
WFLAG	output	WRITE 延时调整输出标志，用以表示写入延时调整 under-flow 或 over-flow。

端口名	I/O	描述
RVALID	output	READ 模式数据有效标志
RBURST	output	READ 突发检测输出

参数介绍

表 5-29 DQS 参数介绍

参数名	取值范围	默认值	描述
FIFO_MODE_SEL	1'b0 , 1'b1	1'b0	FIFO 模式选择 1'b0: DDR memory 模式 1'b1: GDDR 模式
RD_PNTR	"000", "001", "010", "011", "100", "101", "110", "111"	3'b000	FIFO 读指针设置
DQS_MODE	"X1", "X2_DDR2", "X2_DDR3", "X4", "X2_DDR3_EXT"	"X1"	DQS 模式选择
HWL	"false", "true"	"false"	update0/1 时序关系控制 "false": update1 比 update0 提前一个周期； "true": update1 和 update0 时序相同
GSREN	"false", "true"	"false"	启用全局复位 GSR

连接规则

- DQS 的输入 DQSI 来自 IO PAD；
- DQS 的输出 RPOINT 可连接至 IOLOGIC 的 RADDR，也可作用于用户逻辑；
- DQS 的输出 WPOINT 可连接至 IOLOGIC 的 WADDR，也可作用于用户逻辑；
- DQS 的输出 DQSR90 可连接至 IOLOGIC 的 ICLK，也可作用于用户逻辑；
- DQS 的输出 DQSW0/ DQSW270 可连接至 IOLOGIC 的 TCLK，也可作用于用户逻辑。

原语例化

Verilog 例化：

```
DQS uut (
```

```
        .DQSIN(dqs),
        .PCLK(pclk),
        .FCLK(fclk),
        .RESET(reset),
        .READ(read),
        .RCLKSEL(rsel),
        .DLLSTEP(step),
        .WSTEP(wstep),
        .RLOADN(1'b0),
        .RMOVE(1'b0),
        .RDIR(1'b0),
        .WLOADN(1'b0),
        .WMOVE(1'b0),
        .WDIR(1'b0),
        .HOLD(hold),
        .DQSR90(dqsr90),
        .DQSW0(dqsw0),
        .DQSW270(dqsw270),
        .RPOINT(rpoint),
        .WPOINT(wpoint),
        .RVALID(rvalid),
        .RBURST(rburst),
        .RFLAG(rflag),
        .WFLAG(wflag)
    );
    defparam uut.DQS_MODE = "X1";
    defparam uut.FIFO_MODE_SEL = 1'b0;
    defparam uut.RD_PNTR = 3'b001;
```

VHDL 例化：

```
    COMPONENT DQS
        GENERIC(
                FIFO_MODE_SEL:bit:='0';
                RD_PNTR : bit_vector:="000";
                DQS_MODE:string:="X1";
                HWL:string:="false";
```

```
                    GSREN : string:="false"
        );
        PORT(
            DQSIN,PCLK,FCLK,RESET:IN std_logic;
            READ:IN std_logic_vector(3 downto 0);
            RCLKSEL:IN std_logic_vector(2 downto 0);
            DLLSTEP,WSTEP:IN std_logic_vector(7 downto 0);
            RLOADN,RMOVE,RDIR,HOLD:IN std_logic;
            WLOADN,WMOVE,WDIR:IN std_logic;
            DQSR90,DQSW0,DQSW270:OUT std_logic;
            RPOINT, WPOINT:OUT std_logic_vector(2 downto 0);
            RVALID,RBURST,RFLAG,WFLAG:OUT std_logic
        );
END COMPONENT;
uut:DQS
    GENERIC MAP(
                FIFO_MODE_SEL=>'0',
                RD_PNTR=>"000",
                DQS_MODE=>"X1",
                HWL=>"false",
                GSREN=>"false"
        )
        PORT MAP(
            DQSIN=>dqsin,
            PCLK=>pclk,
            FCLK=>fclk,
            RESET=>reset,
            READ=>read,
            RCLKSEL=>rclksel,
            DLLSTEP=>step,
            WSTEP=>wstep,
            RLOADN=>rloadn,
            RMOVE=>rmove,
            RDIR=>rdir,
            HOLD=>hold,
```

```
        WLOADN=>wloadn,
        WMOVE=>wmove,
        WDIR=>wdir,
        DQSR90=>dqsr90,
        DQSW0=>dqsw0,
        DQSW270=>dqsw270,
        RPOINT=>rpoint,
        WPOINT=>wpoint,
        RVALID=>rvalid,
        RBURST=>rburst,
        RFLAG=>rflag,
        WFLAG=>wflag
    );
```

6 晶振时钟

6.1 原语介绍

6.1.1 OSC

OSC，片内晶振。

适用器件

表 6-1 OSC 适用器件

家族	系列	器件
晨熙® (Arora)	GW2A	GW2A-18, GW2A-18C, GW2A-55, GW2A-55C
	GW2AN	GW2AN-55C
	GW2AR	GW2AR-18, GW2AR-18C
	GW2ANR	GW2ANR-18C
小蜜蜂® (LittleBee®)	GW1N	GW1N-4, GW1N-4B, GW1N-4D, GW1N-9, GW1N-9C
	GW1NR	GW1NR-4, GW1NR-4B, GW1NR-4D, GW1NR-9, GW1NR-9C
	GW1NRF	GW1NRF-4B

功能描述

GOWIN FPGA 内嵌了一个可编程片内晶振，编程过程中为 MSPI 编程模式提供时钟源，还可以为用户设计提供时钟源，通过配置工作参数，可以获得多达 64 种时钟频率。

器件输出时钟频率可以通过如下公式计算得到：

$f_{CLKOUT} = f_{osc}/\mathrm{FREQ_DIV}$；

其中f_{osc}为 OSC 振荡频率，根据器件除数 FREQ_DIV 为配置参数，范围为 2~128 的偶数。

注！

f_{osc}根据不同器件取值不同，GW1N-4，GW1NR-4，GW1N-4B，GW1NR-4B，GW1NRF-4B，GW1N-4D，GW1NR-4D 器件为 210MHz，其他支持器件为 250MHz。

端口示意图

图 6-1 OSC 端口示意图

端口介绍

表 6-2 OSC 端口介绍

端口名	I/O	描述
OSCOUT	output	OSC 输出时钟信号

参数介绍

表 6-3 OSC 参数介绍

参数名	取值范围	默认值	描述
FREQ_DIV	2~128(偶数)	100	OSC 分频系数设置
DEVICE	"GW1N-4", "GW1N-4B", "GW1N-4D", "GW1NR-4", "GW1NR-4B", “GW1NR-4D", "GW1NRF-4B", "GW1N-9", "GW1N-9C", "GW1NR-9", "GW1NR-9C", "GW2A-18", "GW2AR-18", "GW2A-55", "GW2A-55C", "GW2AN-55C"	"GW1N-4”(小蜜蜂家族) "GW2A-18”(晨曦家族)	器件选择

原语例化

可以直接实例化原语，也可以通过 IP Core Generator 工具产生。

Verilog 例化：

```
OSC uut(
    .OSCOUT(oscout)
);
```

```
defparam uut.FREQ_DIV=100;
defparam uut.DEVICE="GW2A-18";
```

VHDL 例化：

```
COMPONENT OSC
        GENERIC(
                FREQ_DIV:integer:=100;
                DEVICE:string:="GW2A-18"
         );
        PORT(OSCOUT:OUT STD_LOGIC);
END COMPONENT;
uut:OSC
    GENERIC MAP(
                FREQ_DIV=>100,
                DEVICE=>"GW2A-18"
     )
    PORT MAP(OSCOUT=>oscout);
```

6.1.2 OSCZ

OSCZ 是带有动态关闭 OSC 功能的片内晶振。

适用器件

表 6-4 OSCZ 适用器件

家族	系列	器件
小蜜蜂®(LittleBee®)家族	GW1NS	GW1NS-4, GW1NS-4C
	GW1NSR	GW1NSR-4, GW1NSR-4C
	GW1NSER	GW1NSER-4C
	GW1NZ	GW1NZ-1, GW1NZ-1C

功能描述

GW1NZ 等系列 FPGA 产品内嵌了一个可编程的片内晶振，时钟精度可达±5%，支持动态打开/关闭 OSC 功能。编程过程中为 MSPI 编程模式提供时钟源，还可以为用户设计提供时钟源，通过配置工作参数，可以获得多达 64 种时钟频率。输出时钟频率可以通过如下公式计算得到：

$f_{CLKOUT} = f_{oscz}/\mathrm{FREQ_DIV}$；

其中f_{oscz}为 OSCZ 振荡频率，FREQ_DIV 为分频配置参数，范围为 2~128 的偶数。

注！

f_{oscz}根据不同器件及速度等级取值不同，GW1NS-4/GW1NS-4C/GW1NSR-4/GW1NSR-4CGW1NSER-4C 器件 C7 速度等级为 260MHz，其他支持器件及速度等级为 250MHz。

端口示意图

图 6-2 OSCZ 端口示意图

端口介绍

表 6-5 OSCZ 端口介绍

端口名	I/O	描述
OSCEN	input	OSC 使能信号，高电平有效。
OSCOUT	output	OSC 时钟输出信号

注！

对于 GW1NZ-1(C)器件，当使用 OSCZ 原语时，OSC_EN 的生效时间需要>1.2us。

参数介绍

表 6-6 OSCZ 参数介绍

参数名	取值范围	默认值	描述
FREQ_DIV	2~128(偶数)	100	OSC 分频系数设置
S_RATE	"SLOW", "FAST"	"SLOW"	GWINS-4 相关器件速度等级为 C7 时需设为 "FAST"，其他设为 "SLOW"。

原语例化

可以直接实例化原语，也可以通过 IP Core Generator 工具产生。

Verilog 例化：

```
OSCZ uut(
    .OSCOUT(oscout),
    .OSCEN(oscen)
    );
defparam uut.FREQ_DIV=100;
defparam uut.S_RATE="SLOW";
```

VHDL 例化：

```
COMPONENT OSCZ
        GENERIC(
                FREQ_DIV:integer:=100,
                S_RATE:string:="SLOW"
```

```
        );
        PORT(
                OSCOUT:OUT STD_LOGIC;
                OSCEN:IN std_logic
            );
END COMPONENT;
uut:OSCZ
    GENERIC MAP(
                FREQ_DIV=>100,
                S_RATE=>"SLOW"
    )
    PORT MAP(
                OSCOUT=>oscout,
                OSCEN(oscen)
                );
```

6.1.3 OSCH

OSCH，片内晶振。

适用器件

表 6-7 OSCH 适用器件

家族	系列	器件
小蜜蜂® (LittleBee®)	GW1N	GW1N-1, GW1N-1S
	GW1NR	GW1NR-1

功能描述

OSCH 可编程片内晶振，编程过程中为 MSPI 编程模式提供时钟源，还可以为用户设计提供时钟源，通过配置工作参数，可以获得多达 64 种时钟频率。输出时钟频率可以通过如下公式计算得到：

$f_{CLKOUT} = 240MHz/\text{FREQ_DIV}$；

其中除数 FREQ_DIV 为配置参数，范围为 2~128 的偶数。

端口示意图

图 6-3 OSCH 端口示意图

端口介绍

表 6-8 OSCH 端口介绍

端口名	I/O	描述
OSCOUT	output	OSC 时钟输出信号

参数介绍

表 6-9 OSCH 参数介绍

参数名	取值范围	默认值	描述
FREQ_DIV	2~128(偶数)	100	OSC 分频系数设置

原语例化

可以直接实例化原语，也可以通过 IP Core Generator 工具产生。

Verilog 例化：

```
OSCH uut(
    .OSCOUT(oscout)
);
defparam uut.FREQ_DIV=100;
```

VHDL 例化：

```
COMPONENT OSCH
        GENERIC(
                FREQ_DIV:integer:=100
         );
        PORT(OSCOUT:OUT STD_LOGIC);
END COMPONENT;
uut:OSCH
      GENERIC MAP(
                FREQ_DIV=>100
       )
      PORT MAP(OSCOUT=>oscout);
```

6.1.4 OSCO

OSCO 是带有动态关闭 OSC 功能的片内晶振，支持 Regulator 供电功能。

适用器件

表 6-10 OSCO 适用器件

家族	系列	器件
小蜜蜂® (LittleBee®)家族	GW1N	GW1N-2, GW1N-1P5, GW1N-2B, GW1N-1P5B
	GW1NR	GW1NR-2, GW1NR-2B

功能描述

FPGA 产品内嵌了一个可编程的片内晶振，时钟精度可达±5%，支持动态打开/关闭 OSC 功能，支持 Regulator 供电。编程过程中为 MSPI 编程模式提供时钟源，还可以为用户设计提供时钟源，通过配置工作参数，可以获得多达 64 种时钟频率。输出时钟频率可以通过如下公式计算得到：

$f_{CLKOUT} = 250MHz/\mathrm{FREQ_DIV}$；

其中除数 FREQ_DIV 为配置参数，范围为 2~128 的偶数。

端口示意图

图 6-4 OSCO 端口示意图

端口介绍

表 6-11 OSCO 端口介绍

端口名	I/O	描述
OSCEN	input	OSC 使能信号，高电平有效。
OSCOUT	output	OSC 时钟输出信号

参数介绍

表 6-12 OSCO 参数介绍

参数名	取值范围	默认值	描述
FREQ_DIV	2~128(even)	100	OSC 分频系数设置
REGULATOR_EN	1'b0, 1'b1	1'b0	1'b0:OSCO 由 VCC 供电； 1'b1:OSCO 由 Regulator 供电

原语例化

可以直接实例化原语，也可以通过 IP Core Generator 工具产生。

Verilog 例化：

```
OSCO uut(
    .OSCOUT(oscout),
    .OSCEN(oscen)
    );
defparam uut.FREQ_DIV=100;
defparam uut. REGULATOR_EN =1'b0;
```

VHDL 例化：

```
COMPONENT OSCO
        GENERIC(
                FREQ_DIV:integer:=100;
                REGULATOR_EN : bit := '0'
         );
        PORT(
              OSCOUT:OUT STD_LOGIC;
              OSCEN:IN std_logic
             );
END COMPONENT;
uut:OSCO
     GENERIC MAP(
                 FREQ_DIV=>100,
                 REGULATOR_EN=> '0'
      )
     PORT MAP(
              OSCOUT=>oscout,
              OSCEN(oscen)
              );
```

6.1.5 OSCW

OSCW，片内晶振。

适用器件

表 6-13 OSCW 适用器件

家族	系列	器件
晨熙® (Arora)	GW2AN	GW2AN-18X, GW2AN-9X

功能描述

FPGA 产品内嵌了一个可编程的片内晶振，时钟精度可达±5%。编程过程中为 MSPI 编程模式提供时钟源，还可以为用户设计提供时钟源，通过配置工作参数，可以获得多达 64 种时钟频率。输出时钟频率可以通过如下公式计算得到：

$f_{CLKOUT} = 200MHz/\mathrm{FREQ_DIV}$；

其中除数 FREQ_DIV 为配置参数，范围为 2~128 的偶数。

端口示意图

图 6-5 OSCW 端口示意图

端口介绍

表 6-14 OSCO 端口介绍

端口名	I/O	描述
OSCOUT	output	OSC 时钟输出信号

参数介绍

表 6-15 OSCW 参数介绍

参数名	取值范围	默认值	描述
FREQ_DIV	2~128(偶数)	80	OSC 分频系数设置

原语例化

可以直接实例化原语，也可以通过 IP Core Generator 工具产生。

Verilog 例化：

```
OSCW uut(
    .OSCOUT(oscout)
);
```

```
defparam uut.FREQ_DIV=80;
```

VHDL 例化：

```
COMPONENT OSCW
        GENERIC(
                FREQ_DIV:integer:=100
         );
        PORT(
                OSCOUT:OUT STD_LOGIC
         );
END COMPONENT;
uut:OSCW
     GENERIC MAP(
                FREQ_DIV=>80
      )
     PORT MAP(
                OSCOUT=>oscout
      );
```

6.2 IP调用

在 IP Core Generator 界面中单击 OSC，界面右侧会显示 OSC 的相关信息概要。

注！

对于 GW1NZ-1(C)器件，当调用 OSC IP 时，OSC_EN 的生效时间需要>1.2us。

IP 配置

在 IP Core Generator 界面中，双击“OSC”，弹出 OSC 的“IP Customization”窗口，该窗口包括“ General”配置框、“Options”配置框和端口显示框图，如图 6-6 所示。

图 6-6 OSC 的 IP Customization 窗口结构

1. General 配置框
   General 配置框用于配置产生的 IP 设计文件的相关信息。OSC 的 General 配置框的使用和 DQCE 模块的类似，请参考 DQCE 中的 General 配置框。
2. Options 配置框
   Options 配置框用于用户自定义配置 IP，Options 配置框如图 6-6 所示。Frequency Divider：分频值。该值为 2 的整数倍，取值范围为 2~128。
3. 端口显示框图
   端口显示框图显示 IP Core 的配置结果示例框图，如图 6-6 所示。

IP 生成文件

IP 窗口配置完成后，产生以配置文件“File Name”命名的三个文件，以默认配置为例进行介绍：

- IP 设计文件“gowin_osc.v”为完整的 verilog 模块，根据用户的 IP 配置，产生实例化的 OSC；
- IP 设计使用模板文件 gowin_osc_tmp.v，为用户提供 IP 设计使用模板文件；
- IP 配置文件：“gowin_osc.ipc”，用户可加载该文件对 IP 进行配置。

注！

如配置中选择的语言是 VHDL，则产生的前两个文件名后缀为.vhd。

GOWIN高云

智慧逻辑 定制未来