



Gowin 时钟资源(Clock) 用户指南

UG286-1.4, 2019-12-04

版权所有©2019 广东高云半导体科技股份有限公司

未经本公司书面许可，任何单位和个人不得擅自摘抄、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
2016/07/15	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。

目录

目录	i
图目录	iv
表目录	v
1 关于本手册	1
1.1 手册内容	1
1.2 适用产品	1
1.3 相关文档	1
1.4 术语、缩略语	2
1.5 技术支持与反馈	2
2 概述	3
3 全局时钟	4
3.1 概述	4
3.2 BUFG	12
3.2.1 介绍	12
3.2.2 原语例化	12
3.2.3 端口介绍	12
3.3 DCS	12
3.3.1 介绍	12
3.3.2 原语例化	13
3.3.3 功能图	14
3.3.4 端口介绍	14
3.3.5 参数描述	14
3.3.6 时序图	15
3.4 DQCE	16
3.4.1 介绍	16
3.4.2 原语	16
3.4.3 功能图	17
3.4.4 端口	17

4 高速时钟	18
4.1 概述	18
4.2 CLKDIV	19
4.2.1 介绍	19
4.2.2 原语	19
4.2.3 功能图	20
4.2.4 端口	20
4.2.5 参数	20
4.2.6 应用示意图	21
4.3 DDR 存储器接口时钟管理 DQS	21
5 OSC	22
5.1 概述	22
5.2 功能框图	22
5.3 原语	22
5.4 端口	23
5.5 参数	23
5.6 频率值	23
6 DLL	24
6.1 概述	24
6.2 DLL	24
6.2.1 介绍	24
6.2.2 功能图	24
6.2.3 端口介绍	25
6.2.4 参数介绍	25
6.2.5 原语例化	26
6.3 DLLDLY	27
6.3.1 介绍	27
6.3.2 功能图	27
6.3.3 端口介绍	28
6.3.4 参数介绍	28
6.3.5 原语例化	29
7 PLL	31
7.1 概述	31
7.1.1 功能图	31
7.1.2 端口介绍	32
7.1.3 参数介绍	36

7.1.4 原语例化..... 37

7.2 举例..... 42

图目录

图 3-1 GW1N/GW1NR 系列产品 GCLK 象限分布示意图.....	5
图 3-2 GW1NZ-1 器件时钟资源.....	6
图 3-3 GCLK 象限分布示意	7
图 3-4 GW1NS-2C/GW1NS-2/GW1NSR-2/GW1NSR-2C/GW1NSE-2C/ GW1NSER-2C 时钟资源 .	8
图 3-5 GW1NS-4C/GW1NS-4/GW1NSR-4/GW1NSR-4C 时钟资源.....	8
图 3-6 GCLK 象限分布示意	9
图 3-7 GW2A/GW2AR 系列产品时钟资源.....	10
图 3-8 GW2A/GW2AR 系列产品 GCLK 象限分布示意图	11
图 3-9 DCS 接口示意图.....	14
图 3-10 Non-Glitchless 模式时序图	15
图 3-11 DCS mode: RISING 时序图	15
图 3-12 DCS mode: FALLING 时序图	16
图 3-13 DCS mode: CLK0_GND 时序图	16
图 3-14 DCS mode: CLK0_VCC 时序图.....	16
图 3-15 DQCE 结构示意图	17
图 4-1 CLKDIV 端口示意图	20
图 4-2 CLKDIV 应用示意图	21
图 4-3 DQS 示意图.....	21
图 5-1 OSC 示意图.....	22
图 6-1 DLL 端口示意图.....	24
图 6-2 DLLDLY 端口示意图	27
图 7-1 PLL 端口示意图	31

表目录

表 1-1 术语、缩略语	2
表 3-1 BUFG 端口信号	12
表 3-2 DCS 端口信号	14
表 3-3DCS 参数描述	14
表 3-4 DQCE 端口信号	17
表 4-1 CLKDIV 端口信号	20
表 4-2 CLKDIV 参数描述	20
表 5-1 OSC 端口信号	23
表 5-2 OSC 参数描述	23
表 6-1 端口介绍	25
表 6-2 参数介绍	25
表 6-3 端口介绍	28
表 6-4 参数介绍	28
表 7-1 PLL 性能	32
表 7-2 端口介绍	32
表 7-3 IDSEL 端口参数对照表	33
表 7-4 FBDSSEL 端口参数对照表	33
表 7-5 ODSEL 端口参数对照表	34
表 7-6 PLL 相位参数调整对照表	34
表 7-7 PLL 占空比参数调整对照表	35
表 7-8 PLL 延迟参数调整对照表	35
表 7-9 参数介绍	36

1 关于本手册

1.1 手册内容

本档介绍了时钟资源部分的功能、定义及使用方法。

该手册主要包括四个部分：

1. 时钟资源章节，主要介绍了时钟资源的整体布局、时钟资源种类及相关时钟资源原语介绍；
2. OSC 章节，介绍了内部晶振作为用户时钟的频率范围、参数设置及原语例化等；
3. DLL 章节，介绍了 DLL 和 DLLDLY 的功能、定义等；
4. PLL 章节，介绍了 PLL 的功能、定义等。

1.2 适用产品

本手册中描述的信息适用于所有高云半导体 FPGA 产品。

1.3 相关文档

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

1. GW1N 系列 FPGA 产品数据手册
2. GW1NR 系列 FPGA 产品数据手册
3. GW1NS 系列 FPGA 产品数据手册
4. GW1NSR 系列 FPGA 产品数据手册
5. GW1NSE 系列 FPGA 产品数据手册
6. GW1NZ 系列 FPGA 产品数据手册
7. GW1N 系列 FPGA 产品封装与管脚手册
8. GW1NR 系列 FPGA 产品封装与管脚手册
9. GW1N 系列 FPGA 产品封装与管脚手册
10. GW1NS 系列 FPGA 产品封装与管脚手册
11. GW1NSR 系列 FPGA 产品编程配置手册
12. GW1NSE 系列 FPGA 产品封装与管脚手册
13. GW1NZ 系列 FPGA 产品编程配置手册
14. GW2A 系列 FPGA 产品数据手册

15. GW2AR 系列 FPGA 产品数据手册
16. GW2A 系列 FPGA 产品封装与管脚手册
17. GW2AR 系列 FPGA 产品封装与管脚手册
18. GW2A 系列 FPGA 产品编程配置手册

1.4 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
FPGA	Field Programmable Gate Array	现场可编程门阵列
OSC	Oscillator	晶体振荡器
DLL	Delay-locked Loop	延迟锁相环
DLLDLY	DLL Delay	DLL 延迟
PLL	Phase-locked Loop	锁相环
GCLK	Global Clock	全局时钟
HCLK	High Clock	高速时钟
BUFG	Buffer Global	全局时钟缓冲器
DCS	Dynamic Clock Selector	动态时钟选择器
DQCE	Dynamic Quadrant Clock Enable	动态使能象限时钟
CLKDIV	Clock Divider	时钟分频器

1.5 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 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网络(**GCLK**)，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**)、延迟锁相环 (**DLL**)、高速时钟 **HCLK** 和 **DDR** 存储器接口数据脉冲时钟 **DQS** 等时钟资源。

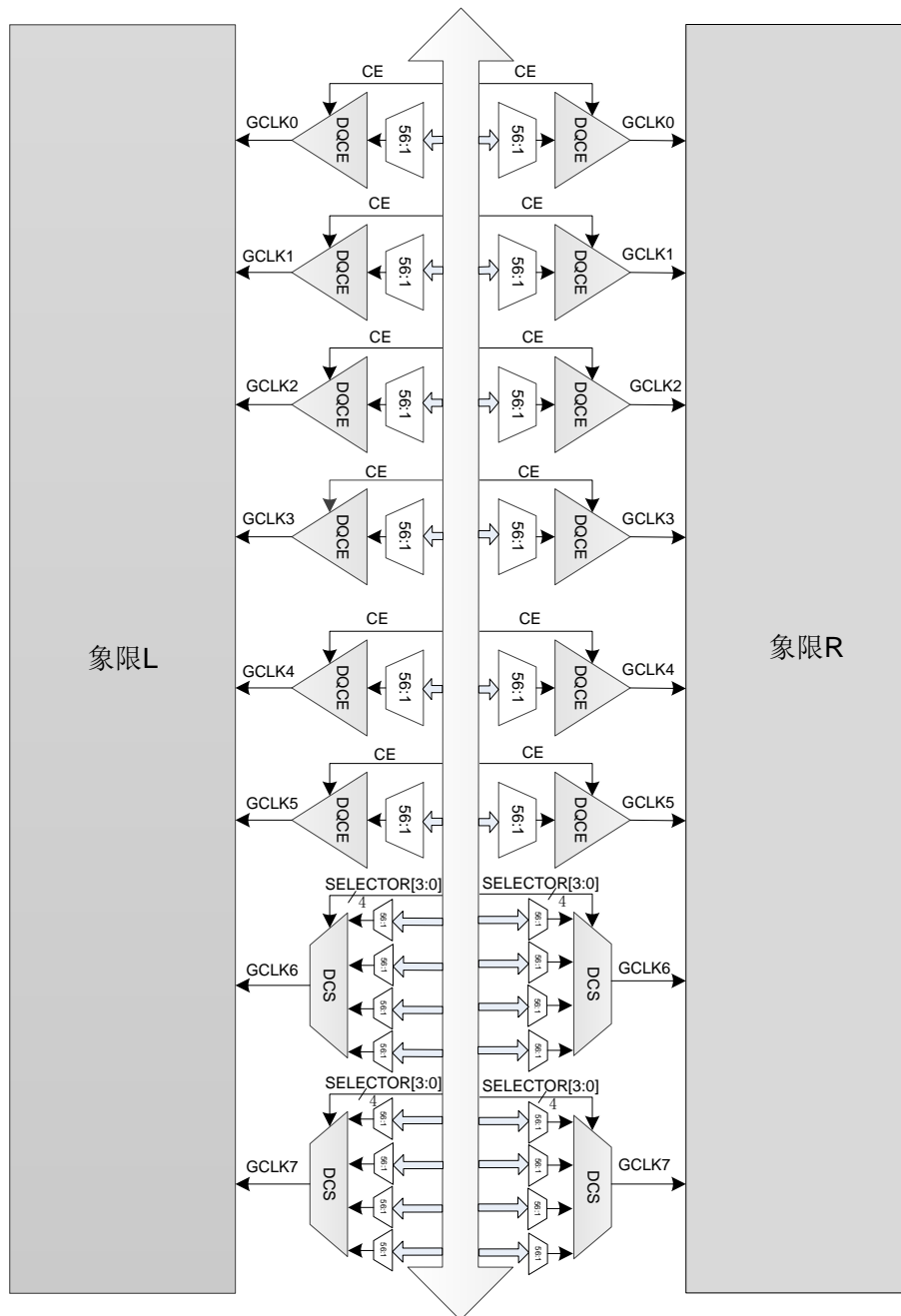
3 全局时钟

3.1 概述

GW1N/GW1NR 系列 FPGA 产品

GCLK 在 GW1N/GW1NR 系列产品中按象限分布,分成 L、R 两个象限,如图 3-1 所示: FPGA 被分成两个象限,每个象限提供 8 个 GCLK 网络,每个 GCLK 可选时钟源包括专用的时钟输入管脚和普通的布线资源单元 (CRU),使用专用的时钟输入管脚可以取得更好的时钟性能。

图 3-1 GW1N/GW1NR 系列产品 GCLK 象限分布示意图



GW1NZ 系列 FPGA 产品

GCLK 在 GW1NZ-1 器件中按象限分布，分成 L、R 两个象限。每个象限提供 8 个 GCLK 网络，每个 GCLK 有 12 个可选时钟源。GCLK 的可选时钟源包括专用的时钟输入管脚和普通布线资源，使用专用的时钟输入管脚具有更好的时钟性能。

图 3-2 GW1NZ-1 器件时钟资源

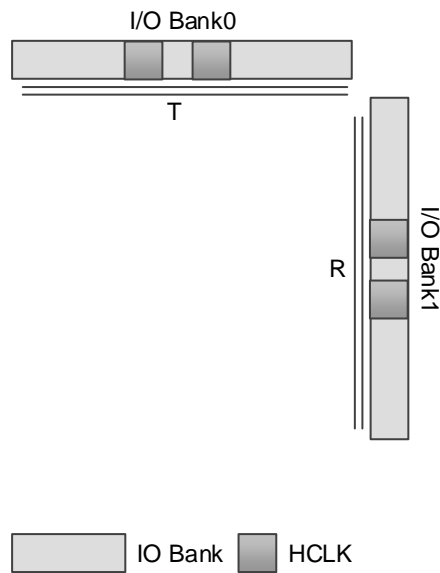
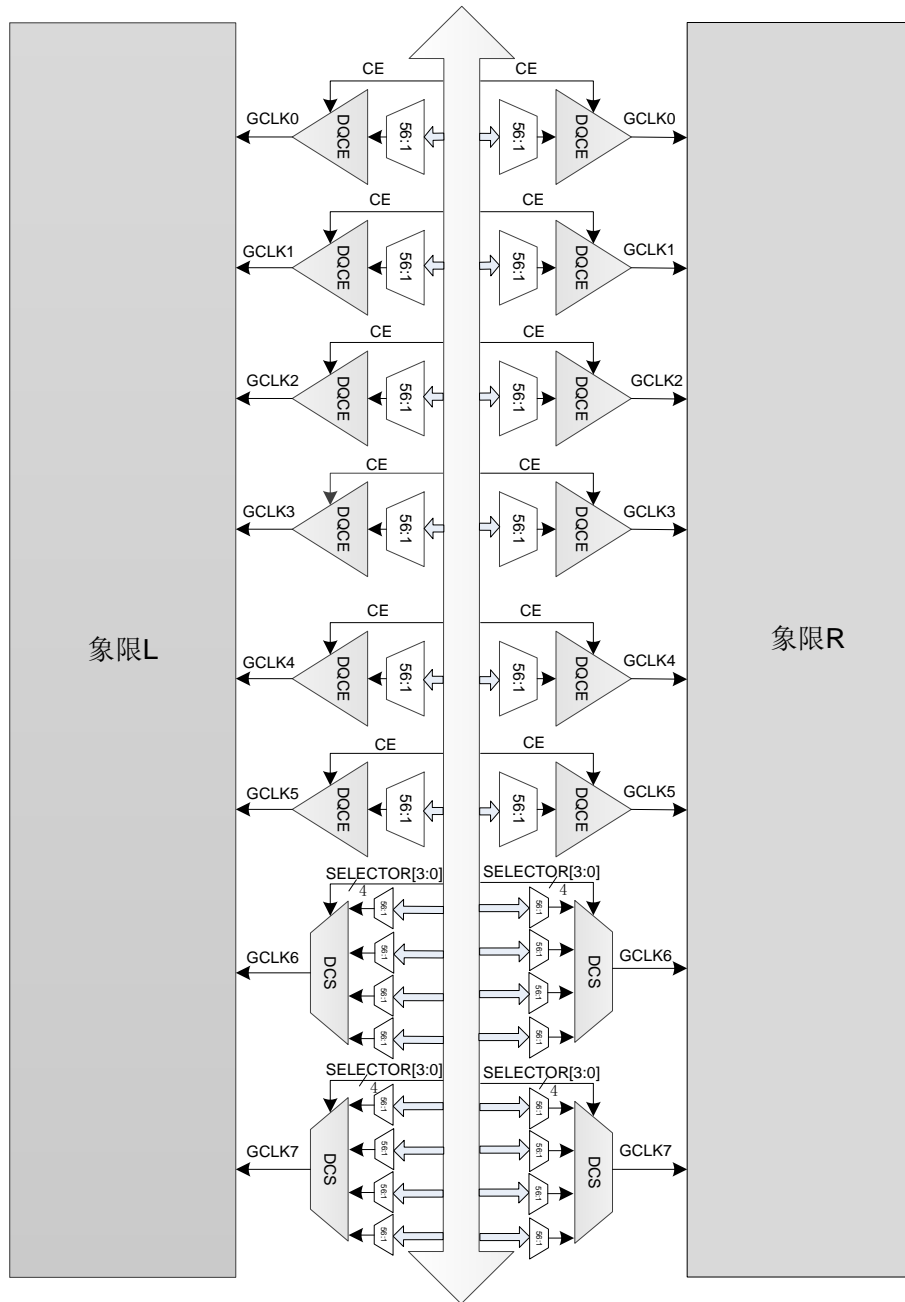


图 3-3 GCLK 象限分布示意



GW1NS/GW1NSR/GW1NSE/GW1NSER 系列 FPGA 产品

GCLK 在 GW1NS 系列 FPGA 产品中按象限分布,分成 L、R 两个象限,每个象限提供 8 个 GCLK 网络。GCLK 的可选时钟源包括专用的时钟输入管脚和普通布线资源,使用专用的时钟输入管脚具有更好的时钟性能。

图 3-4 GW1NS-2C/GW1NS-2/GW1NSR-2/GW1NSR-2C/GW1NSE-2C/GW1NSER-2C 时钟资源

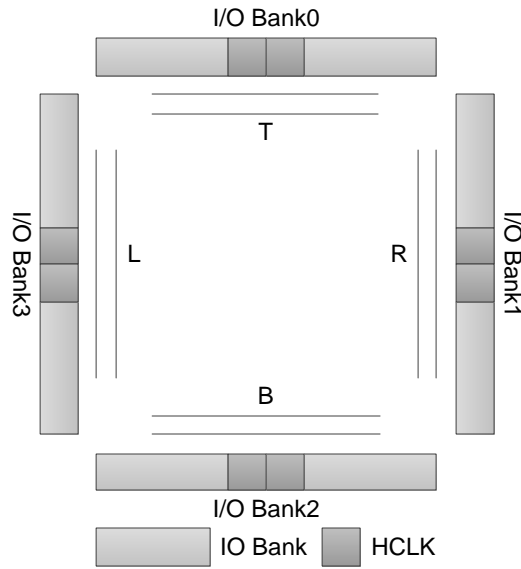


图 3-5 GW1NS-4C/GW1NS-4/GW1NSR-4/GW1NSR-4C 时钟资源

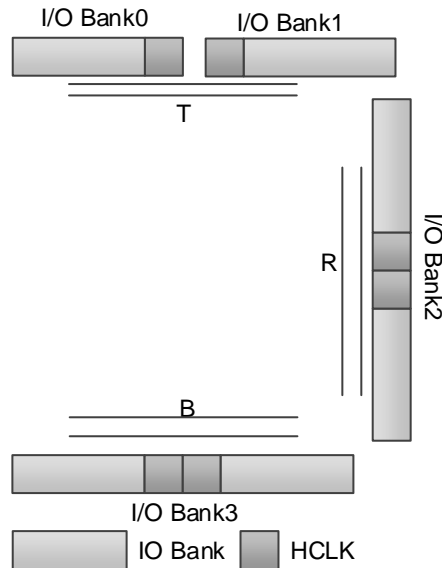
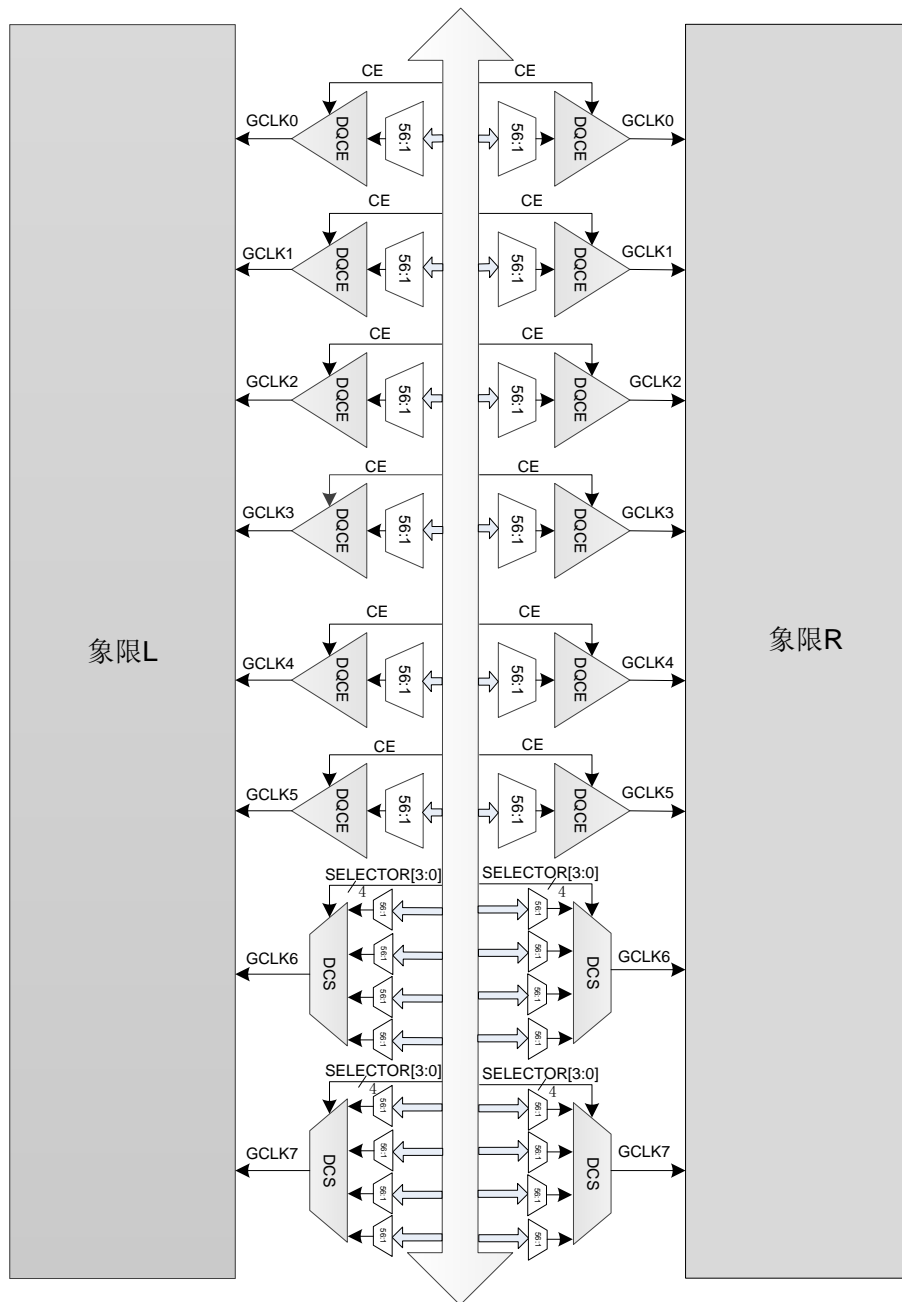


图 3-6 GCLK 象限分布示意



GW2A/GW2AR 系列 FPGA 产品

GCLK 在 GW2A/GW2AR 系列 FPGA 产品中按象限分布，分为四个象限，每个象限提供 8 个 GCLK 网络。GCLK 的可选时钟源包括专用的时钟输入管脚和普通布线资源，使用专用的时钟输入管脚具有更好的时钟性能。

图 3-7 GW2A/GW2AR 系列产品时钟资源

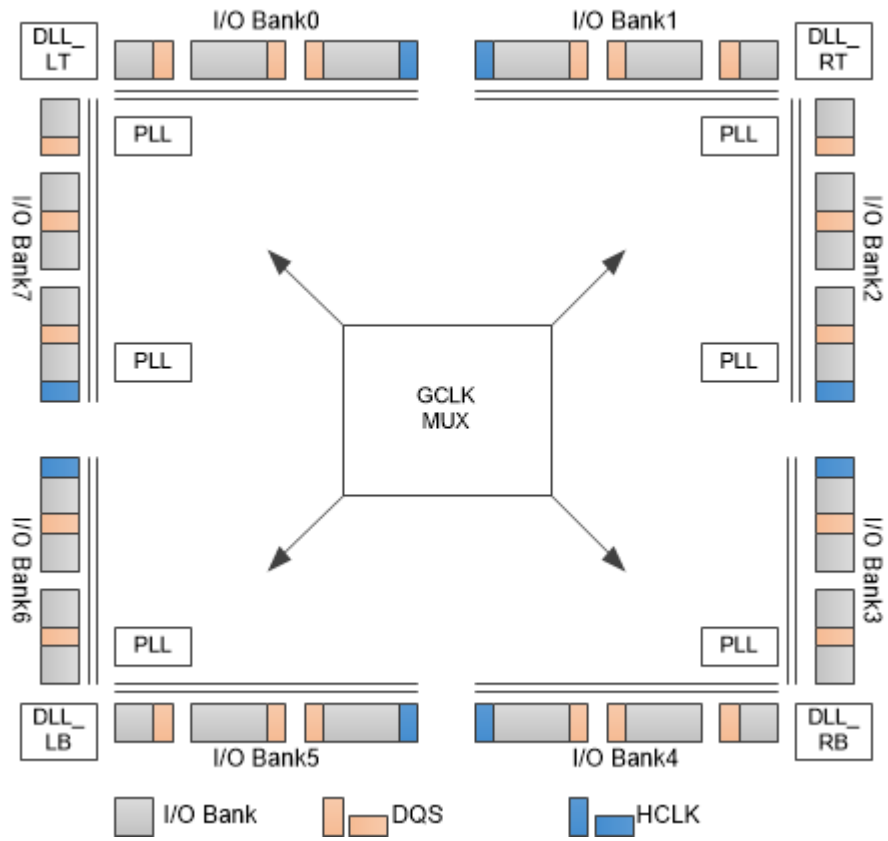
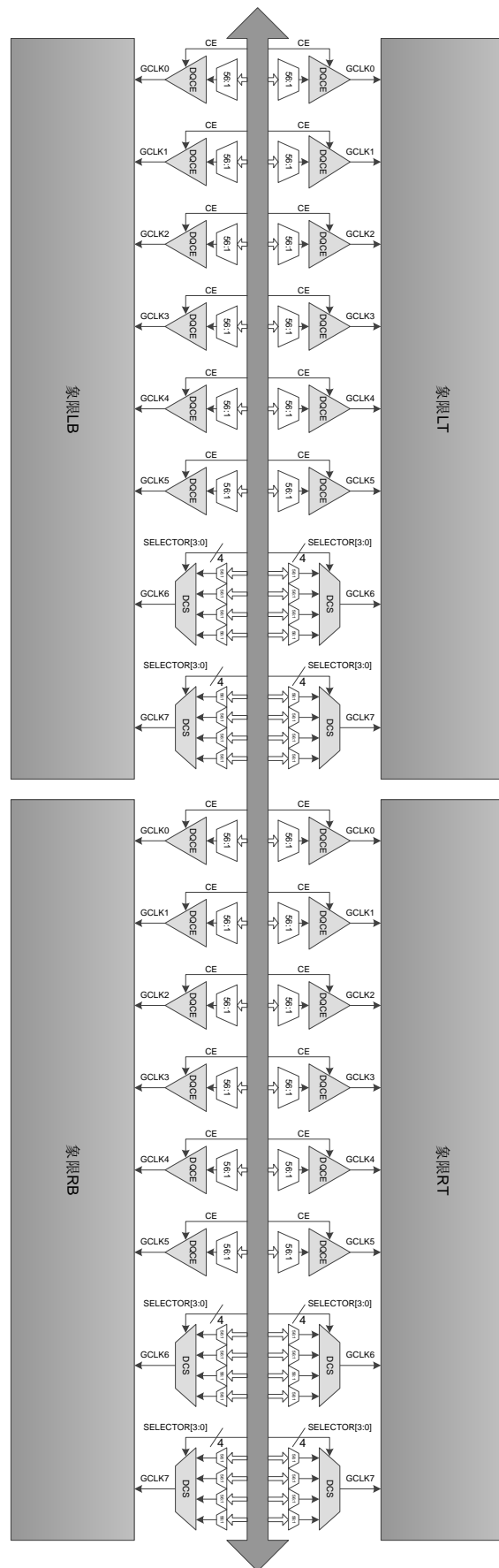


图 3-8 GW2A/GW2AR 系列产品 GCLK 象限分布示意图



3.2 BUFG

3.2.1 介绍

BUFG 是具有高扇出的全局时钟缓冲器，具有高扇出驱动能力的缓冲器，可以将信号连到时钟抖动可以忽略不计的全局时钟网络。BUFG 组件还可应用于典型的高扇出信号和网络，如复位信号和时钟使能信号。如果要对全局时钟实现 PLL 或 DLL 等时钟管理，则需要该缓冲器。

3.2.2 原语例化

Verilog 例化:

```
BUFG uut(
    .O(o),
    .I(i)
);
```

Vhdl 例化:

```
COMPONENT BUFG
    PORT(
        O:OUT std_logic;
        I:IN std_logic
    );
END COMPONENT;
uut:BUFG
    PORT MAP(
        O=>o,
        I=>i
    );
```

3.2.3 端口介绍

表 3-1 BUFG 端口信号

端口名	I/O	描述
O	output	时钟输出
I	input	时钟输入

3.3 DCS

3.3.1 介绍

DCS (Dynamic Clock Selector)，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。

3.3.2 原语例化

Verilog 例化:

```
DCS dcs_inst (
    .CLK0(clk0),
    .CLK1(clk1),
    .CLK2(clk2),
    .CLK3(clk3),
    .CLKSEL(clksel[3:0]),
    .SELFORCE(selfforce),
    .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,
```

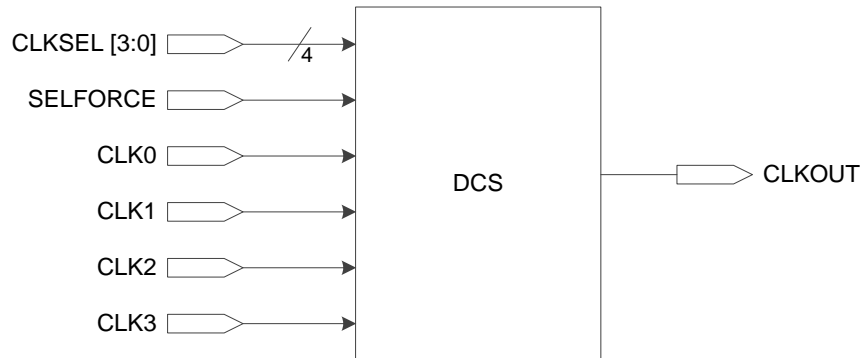
```

SELFORCE=>selforce,
CLKOUT=>clkout
);

```

3.3.3 功能图

图 3-9 DCS 接口示意图



3.3.4 端口介绍

表 3-2 DCS 端口信号

端口	描述
CLKSEL [3:0]	时钟选择信号;
CLK0	时钟输入 0
CLK1	时钟输入 1
CLK2	时钟输入 2
CLK3	时钟输入 3
SELFORCE	强制模式选择 0: glitchless 模式 1: Non-glitchless 模式
CLKOUT	时钟输出

3.3.5 参数描述

表 3-3DCS 参数描述

参数	描述	取值选项	默认值
DCS_MODE	设置 DCS 模式	CLK0,CLK1,CLK2,CLK3, GND,VCC,RISING,FALLIN, CLK0_GND,CLK1_GND, CLK2_GND,CLK3_GND, CLK0_VCC,CLK1_VCC, CLK2_VCC,CLK3_VCC	RISING

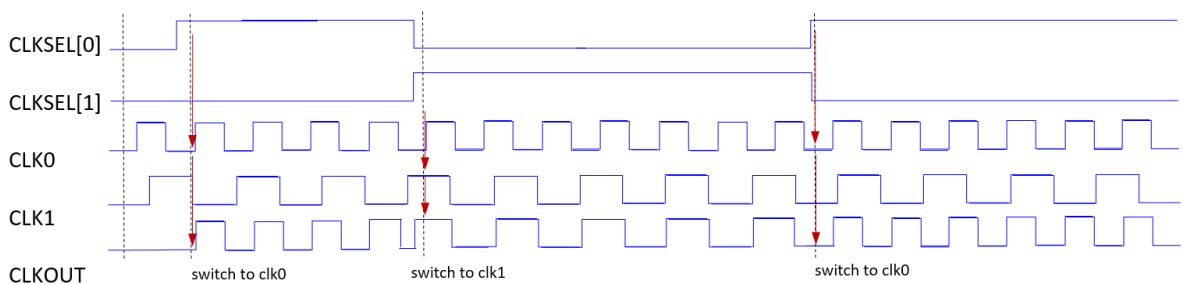
3.3.6 时序图

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 (输入 SELFORCE = '1'), 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。Non-Glitchless 模式时序如图 3-10 所示, 用 CLKSEL[3]~CLKSEL[0] 分别对应选择 CLK3~CLK0, 高电平有效, 转换时序相同。

图 3-10 Non-Glitchless 模式时序图



在 Glitchless 无毛刺模式下 (输入 SELFORCE = '0'), 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。Glitchless 模式时序如图 3-11 到图 3-14 所示, 用 CLKSEL[3]~CLKSEL[0] 分别对应选择 CLK3~CLK0, 转换时序相同。

图 3-11 DCS mode: RISING 时序图

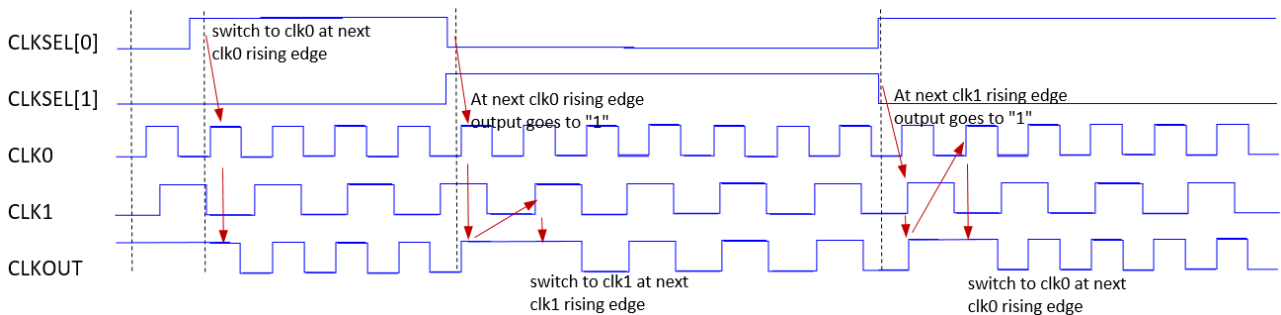


图 3-12 DCS mode: FALLING 时序图

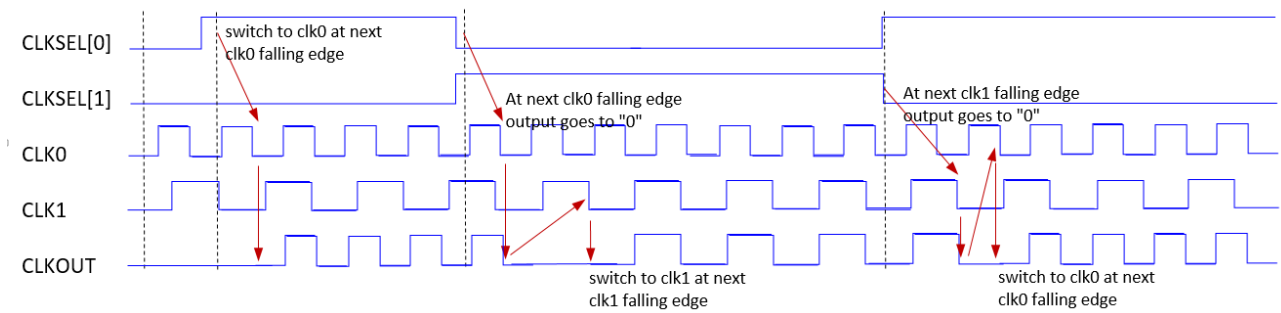


图 3-13 DCS mode: CLK0_GND 时序图

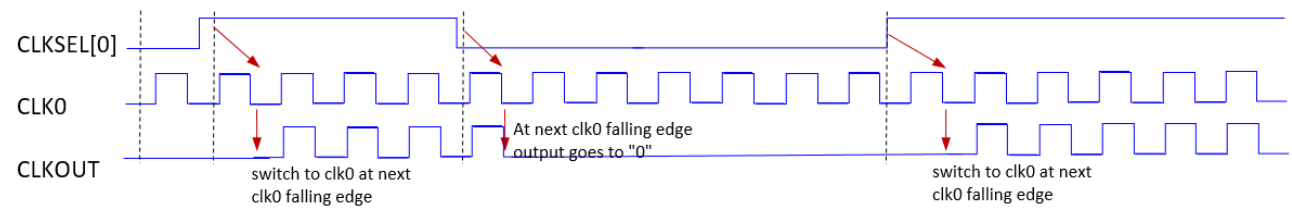
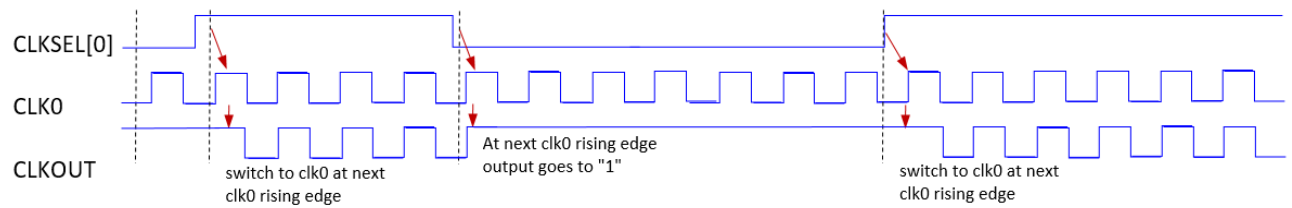


图 3-14 DCS mode: CLK0_VCC 时序图



3.4 DQCE

3.4.1 介绍

通过 DQCE(Dynamic Quadrant Clock Enable)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。

3.4.2 原语

Verilog

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


VHDL

```

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
  
```

3.4.3 功能图

图 3-15 DQCE 结构示意图

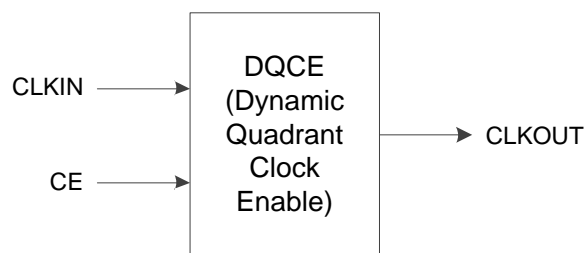
**3.4.4 端口**

表 3-4 DQCE 端口信号

端口	描述
CE	使能信号，高电平有效
CLKIN	时钟输入
CLKOUT	时钟输出

4 高速时钟

4.1 概述

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。

- GW1N-1 有 2 个 HCLK 资源，在 Bank2。
- GW1NZ-1 有 8 个 HCLK，Bank0 和 Bank1 各有 4 个。
- GW1N-1S 有 8 个 HCLK，Bank0 和 Bank1 有 4 个，Bank2 有 4 个。
- GW1N-2/GW1N-2B/GW1N-4/GW1NR-4/GW1N-4B/GW1NR-4B 有 6 个 HCLK，Bank 1、Bank 2 和 Bank 3 各有 2 个。
- GW1N-4S 有 12 个 HCLK，Bank 0 和 Bank 1 共用 4 个，Bank 2 有 4 个，Bank 3 有 4 个。
- GW1NS-2/GW1NS-2C/GW1NSR-2C 有 16 个 HCLK，Bank 0、Bank 1、Bank 2 和 Bank 3 各有 4 个。
- GW1N-9/GW1NR-9/GW1N-6 有 16 个 HCLK，Bank 0、Top IO 的 Bank 1 和 Top IO 的 Bank 3 共用 4 个，Right IO 的 Bank 1 有 4 个，Bank 2 有 4 个，Left IO 的 Bank 3 有 4 个。
- GW2A-18/GW2AR-18/GW2A-55 有 16 个 HCLK，Bank 0~ Bank7 各有 2 个。

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 可以提供给用户使用的功能模块如下：

- DHCE：动态的高速时钟使能模块，功能类似于 DQCE。可动态的打开/关闭高速时钟信号；
- CLKDIV：高速时钟分频模块，每个 Bank 中有一个 CLKDIV。生成和输入时钟相位一致的分频时钟，用于 IO 逻辑工作模式中；
- DHCS：动态的高速时钟选择器；
- PADCLKDEL：动态延迟调整模块，用于专用时钟管脚输入的时钟信号。

4.2 CLKDIV

4.2.1 介绍

高速时钟模块 CLKDIV 对时钟进行 2、3.5、4、5、8 分频，可为 IDE54/IVIDEO/OSER4/OVIDEO 等 IO 逻辑提供时钟。

4.2.2 原语

VHDL

```

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 COMPONENT;

 uut:CLKDIV
  GENERIC MAP(
    DIV_MODE=>"2",
    GSREN=>"false"
  )
  PORT MAP(
    HCLKIN=>hclkin,
    RESETN=>resetn,
    CALIB=>calib,
    CLKOUT=>clkout
  );

```

Verilog

```

CLKDIV clkdiv_inst (
  .HCLKIN(hclkin),
  .RESETN(resetn),
  .CALIB(calib),

```

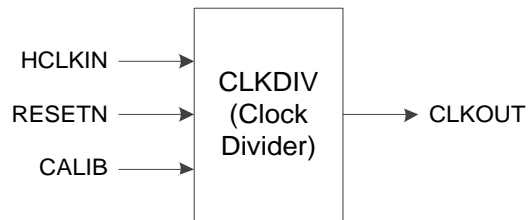
```

        .CLKOUT(clkout)
    );
    defparam clkdiv_inst.DIV_MODE="3.5";
    defparam clkdiv_inst.GSREN="false";

```

4.2.3 功能图

图 4-1 CLKDIV 端口示意图



4.2.4 端口

表 4-1 CLKDIV 端口信号

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

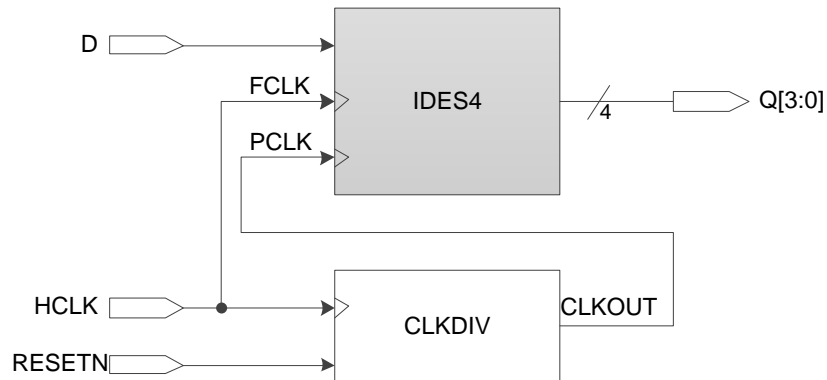
4.2.5 参数

表 4-2 CLKDIV 参数描述

参数名	取值范围	默认值	描述
DIV_MODE	2, 3.5, 4, 5, 8 (8 仅在 GW1N-6、GW1N-9、GW1NS-2 下支持)	2	设置时钟分频系数
GSREN	false, true	false	启用全局复位 GSR

4.2.6 应用示意图

图 4-2 CLKDIV 应用示意图



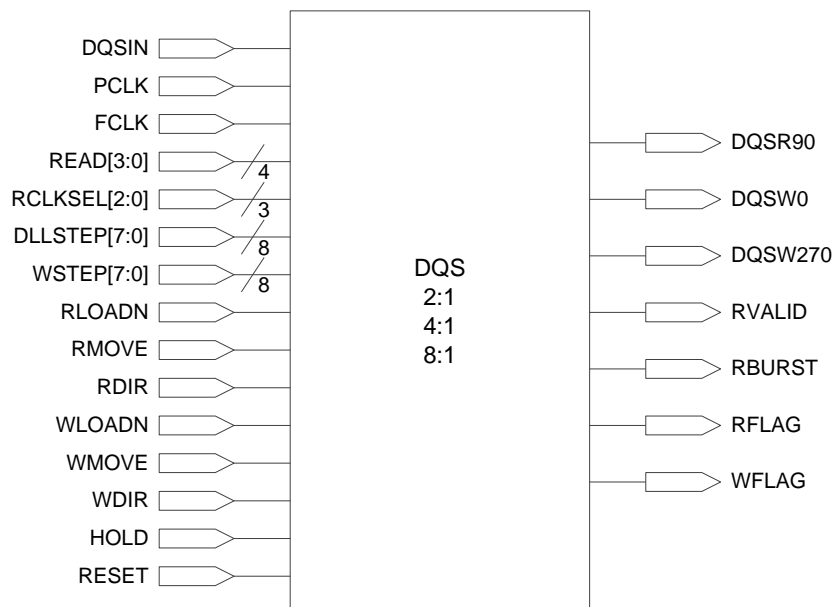
4.3 DDR 存储器接口时钟管理 DQS

GW2A/GW2AR 系列 FPGA 产品的 DQS 模块提供了如下的功能来支持 DDR 存储器接口的时钟需求：

- 接收时钟输入，整理波形并移动 1/4 相位；
- 为输入缓存提供读/写指针；
- 为内部逻辑提供数据有效信号；
- 提供 DDR 输出时钟信号；
- 支持 DDR3 写电压控制。

有 3 种工作模式，用来满足不同的 IO 接口的需求，如图 4-3 所示。

图 4-3 DQS 示意图



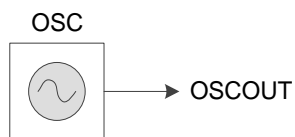
5_{OSC}

5.1 概述

高云半导体 **FPGA** 产品内嵌了一个可编程片内晶振, 为用户提供低抖动、高精度的时钟频率资源, 精度为±5%以内。

5.2 功能框图

图 5-1 OSC 示意图



5.3 原语

VHDL

```

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);Verilog
OSC uut(

```

```

        .OSCOUT(oscout)
    );
    defparam uut.FREQ_DIV=100;
    defparam uut.DEVICE="GW2A-18";

```

5.4 端口

表 5-1 OSC 端口信号

端口名	I/O	描述
OSCOUT	output	OSC 输出时钟

5.5 参数

表 5-2 OSC 参数描述

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
DEVICE	GW1N-1、GW1N-2、GW1N-2B、GW1N-4、GW1N-4B、GW1NR-4、GW1NR-4B、GW1N-6、GW1N-9、GW1NR-9、GW2A-18、GW2AR-18、GW2A-55。	GW1N-2(GW1N 系列) GW2A-18(GW2A 系列)	器件选择

5.6 频率值

片内晶振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：

$$\text{OSCOUT} = \text{FREQ_VALUE} / \text{FREQ_DIV}。$$

其中： FREQ_DIV 为配置参数，范围为 2~128，只支持偶数。

- 对于 GW1N-4/GW1N-4B/GW1NR-4/GW1NR-4B/ GW1NS-4C/ GW1NS-4/ GW1NSR-4C/ GW1NSER-4C 器件，FREQ_VALUE=210MHz。
- 对于 GW1NS-2C、GW1NS-2/GW1NSR-2C/GW1NSER-2C 器件，FREQ_VALUE=240MHz。
- 对于其他器件，FREQ_VALUE=250MHz。

6_{DLL}

6.1 概述

高云半导体 FPGA 产品提供了延迟锁相环模块 DLL。DLL 模块实现时钟参考延时,通过输出的 STEP 信号送到 DLLDLY 模块中,产生满足 IDDR、IDES4/8/10、IVideo 等模块时延要求的 PCLK 时钟。

DLL 的时钟输入来源包括 GCLK 和相邻的 HCLK。

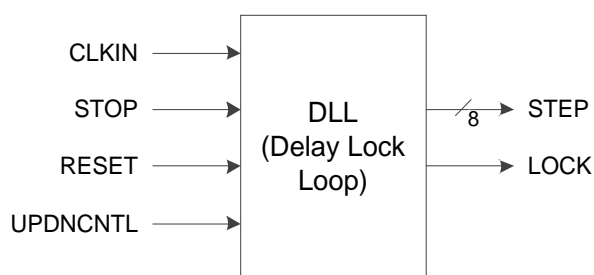
6.2 DLL

6.2.1 介绍

DLL (Delay-Locked Loop) 是延时锁相环,主要用于精确产生时间延时。

6.2.2 功能图

图 6-1 DLL 端口示意图



功能描述

DLL 可基于给定的输入时钟进行时钟相位调整产生不同相位的延时步长 STEP。计算过的输出 STEP 信号会送到相邻的 Bank 中,如 DQS、DLLDLY 模块,同时信号 STEP 也可通过布线送到用户逻辑中去。

6.2.3 端口介绍

表 6-1 端口介绍

端口名	I/O	描述
STEP[7:0]	Output	延时步长输出
LOCK	Output	锁定指示输出，1 表示锁定，0 表示失锁
CLKIN	Input	时钟输入
STOP	Input	停止输入时钟和内部震荡时钟
RESET	Input	异步复位输入，高电平有效
UPDNCNTL	Input	DLL 延时步长更新控制，低电平有效

6.2.4 参数介绍

表 6-2 参数介绍

参数名	参数类型	取值范围	默认值	描述
DLL_FORCE	Integer	0,1	0	DLL 强制延时步长、锁定控制 1:强制锁定，延时步长为 255（最大），用于较低的输入频率模式 0: 正常模式，通过 DLL 生成延时步长和锁定信号
CODESCAL	String	000,001,010,011, 100,101, 110, 111	000	DLL 相移配置（45°~135°） 000:101° 001:112° 010:123° 011:135° 100:79° 101:68° 110:57° 111:45°
SCAL_EN	String	true,false	true	DLL 启用相位偏移功能: true:启用，相位偏移根据参数 CODESCAL 设置 false:禁用，默认 90°相移
DIV_SEL	Integer	1'b0,1'b1	1'b0	DLL 锁定模式选择: 1'b0:正常锁定模式 1'b1:快速锁定模式

连接合法性规则

DLL 的输出 STEP 可连接至 DQS、DLLDLY 模块，同时也可通过布线送到用户逻辑中去。

6.2.5 原语例化

Verilog 例化

```
DLL dll_inst (
    .STEP(step),
    .LOCK(lock),
    .CLKIN(clkin),
    .STOP(stop),
    .RESET(reset),
    .UPDNCNTL(1'b0)
);
defparam dll_inst.DLL_FORCE = 1;
defparam dll_inst.CODESCAL = "000";
defparam dll_inst.SCAL_EN = "true";
defparam dll_inst.DIV_SEL = 1'b0;
```

Vhdl 例化

```
COMPONENT DLL
    GENERIC(
        DLL_FORCE:integer:=0;
        DIV_SEL:bit:='1';
        CODESCAL:STRING:="000";
        SCAL_EN:STRING:="true"
    );
    PORT(
        CLKIN:IN std_logic;
        STOP:IN std_logic;
        RESET:IN std_logic;
        UPDNCNTL:IN std_logic;
        LOCK:OUT std_logic;
        STEP:OUT std_logic_vector(7 downto 0)
    );
```

```

END COMPONENT;
 uut:DLL
   GENERIC MAP(
     DLL_FORCE=>0,
     DIV_SEL=>'1',
     CODESCAL=>"000",
     SCAL_EN=>"true"
   )
   PORT MAP(
     CLKIN=>clkin,
     STOP=>stop,
     RESET=>reset,
     UPDNCNTL=>updncntl,
     LOCK=>lock,
     STEP=>step
   );

```

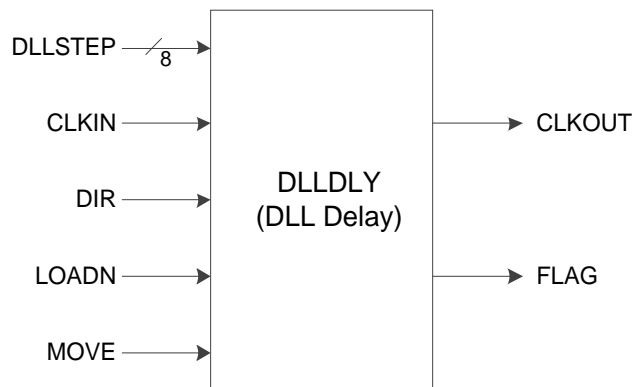
6.3 DLLDLY

6.3.1 介绍

DLLDLY(DLL Delay)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。

6.3.2 功能图

图 6-2 DLLDLY 端口示意图



功能描述

DLLDLY 可与 DLL 配合使用，DLL 提供不同相位的延时 STEP，DLLDLY 根据 DLLSTEP 产生对应相位的延时，得到基于 CLKIN 的延时输出。

6.3.3 端口介绍

表 6-3 端口介绍

端口名	I/O	描述
CLKOUT	Output	时钟输出
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, 来自 DLL 的输出 STEP
CLKIN	Input	时钟输入
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

6.3.4 参数介绍

表 6-4 参数介绍

参数名	参数类型	取值范围	默认值	描述
DLL_INSEL	Integer	1'b0,1'b1	1'b0	DLLDLY 旁路模式选择 1'b0:旁路模式,即输出直接来自 CLKIN 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

连接合法性规则

DLLDLY 的输入 DLLSTEP 来自 DLL 模块的 STEP, 若器件没有 DLL 则可来自用户逻辑。

6.3.5 原语例化

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=>'0',
        DLY_SIGN=>'0',
        LY_ADJ=>0
    )
    PORT MAP(
        DLLSTEP=>step,
        CLKIN=>clkin,
        DIR=>dir,
        LOADN=>loadn,
        MOVE=>move,
        CLKOUT=>clkout,
        FLAG=>flag
    );
```

7 PLL

7.1 概述

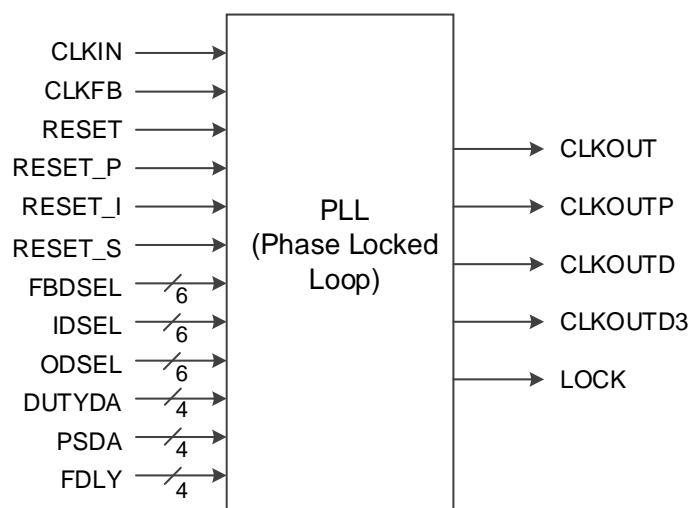
锁相环路是一种反馈控制电路，简称锁相环(PLL, Phase-locked Loop)。利用外部输入的参考时钟信号控制环路内部振荡信号的频率和相位。

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。

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。

7.1.1 功能图

图 7-1 PLL 端口示意图



功能描述

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。

PLL 的性能如下：

表 7-1 PLL 性能

器件 频率 (MHz)	GW1N 家族	GW1N-1S	GW1NS-2	GW1NZ-1	GW2A 家族
输入频率范围	3 ~ 450	3 ~ 450	3 ~ 450	3 ~ 400(LV) 3 ~ 200(ZV)	3 ~ 500
VCO 振荡频率范围	400~900	400~1200	400~1500	400 ~ 800(LV) 200 ~ 400(ZV)	500~1300
输出频率范围	3.125~450	3.125~600	3.125~750	3.125~400(LV) 1.5625~200(ZV)	3.125~500

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

1. $f_{CLKOUT} = (f_{CLKIN} * FBDIV) / IDIV$
2. $f_{VCO} = f_{CLKOUT} * ODIV$
3. $f_{CLKOUTD} = f_{CLKOUT} / SDIV$
4. $f_{PFD} = f_{CLKIN} / IDIV = f_{CLKOUT} / FBDIV$

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。

7.1.2 端口介绍

表 7-2 端口介绍

端口名	I/O	描述
CLKIN	Input	参考时钟输入
CLKFB	Input	反馈时钟输入
RESET	Input	PLL 异步复位输入, 高电平有效
RESET_P	Input	PLL 关断 (Power Down) 输入, 高电平有效
RESET_I	Input	PLL IDIV 分频器异步复位输入, 高电平有效
RESET_S	Input	PLL SDIV 分频器异步复位输入, 高电平有效
FBDSEL[5:0]	Input	动态控制 FBDIV 取值, 范围 1~64
IDSEL[5:0]	Input	动态控制 IDIV 取值, 范围 1~64
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
PSDA[3:0]	Input	相位动态调整

端口名	I/O	描述
FDLY[3:0]	Input	精细延时动态调整
CLKOUT	Output	PLL 时钟输出
LOCK	Output	PLL 锁定指示, 1 表示锁定, 0 表示失锁
CLKOUTP	Output	PLL 带有相位和占空比调整的时钟输出
CLKOUTD	Output	PLL 经过 SDIV 的时钟输出, CLKOUT 或 CLKOUTP 经过 SDIV 分频器后的输出
CLKOUTD3	Output	PLL 经过 DIV3 的时钟输出, CLKOUT 或 CLKOUTP 经过 3 分频后的输出

表 7-3 IDSEL 端口参数对照表

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

表 7-4 FBDSEL 端口参数对照表

FBDSEL [5: 0]	FDIV 参数值
111111	1
111110	2
111101	3
111100	4
111011	5
111010	6
111001	7
111000	8
110111	9
.....

FBDSEL [5: 0]	FDIV 参数值
000000	64

表 7-5 ODSEL 端口参数对照表

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

表 7-6 PLL 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°
1011	247.5°
1100	270°
1101	292.5°
1110	315°
1111	337.5°

表 7-7 PLL 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

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“180°”，然后50%占空比的设置将被“0”（0000）。

动态占空比计算设置：

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

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。每一步增加一个 0.125ns，共 15 步（1.875ns）。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 7-8 PLL 延迟参数调整对照表

端口 FDLY [3: 0]	延迟步数
0111	7
1011	11
1101	13
1110	14
1111	15

7.1.3 参数介绍

表 7-9 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: add
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: add

参数名	取值范围	默认值	描述
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 PLL, CLKOUT 直接来自 CLKIN true: CLKIN 旁路 PLL 直接作用于 CLKOUT false: 正常模式
CLKOUTP_BYPASS	true,false	false	旁路 PLL, CLKOUTP 直接来自 CLKIN true: CLKIN 旁路 PLL 直接作用于 CLKOUTP false: 正常模式
CLKOUTD_BYPASS	true,false	false	旁路 PLL, CLKOUTD 直接来自 CLKIN true: CLKIN 旁路 PLL 直接作用于 CLKOUTD false: 正常模式
DEVICE	GW1N-1、GW1N-2、 GW1N-2B、GW1NS-2、 GW1NS-2C、GW1N-4、 GW1N-4B、GW1NR-4、 GW1NR-4B、GW1N-6、 GW1N-9、GW1NR-9、 GW2A-18、GW2AR-18、 GW2A-55。	GW1N-2	器件选择

7.1.4 原语例化

Verilog 例化

```

PLL pll_inst(
    .CLKOUT(clkout),
    .LOCK(lock),
    .CLKOUTP(clkoutp),

```

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

```
defparam pll_inst.DEVICE = "GW1N-4";
```

Vhdl 例化

```
COMPONENT PLL
```

```
  GENERIC(
```

```
    FCLKIN:STRING:= "100.0";
```

```
    DEVICE:STRING:= "GW2A-18";
```

```
    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;
        RESET_I:IN std_logic;
        RESET_S: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:PLL
    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=>clk_in,
        CLKFB=>clkfb,
        IDSEL=>idsel,
        FBDSEL=>fbdsel,
        ODSEL=>odsel,
        RESET=>reset,
        RESET_P=>reset_p,
        RESET_I=>reset_i,
        RESET_S=>reset_s,
        PSDA=>psda,
        FDLY=>fdly,
        DUTYDA=>dutyda,
        LOCK=>lock,
        CLKOUT=>clkout,
        CLKOUTD=>clkoutd,
        CLKOUTP=>clkoutp ,
        CLKOUTD3=>clkoutd3
    );
```

7.2 举例

若要得到输入时钟1.75倍频的时钟，可设置IDIV_SEL=1（输入分频系数为2），FBDIV_SEL=6（FB分频系数为7），DYN_SDIV_SEL=2（SDIV分频系数为2），根据分频公式可得CLKOUT的频率为CLKIN的3.5倍，CLKOUTD的频率为CLKIN的1.75倍。

Verilog语言的模块调用：

```
.PLL#(  
.FCLKIN("100.0"),  
.DYN_IN_SEL("false"),  
.IN_SEL(5),  
.DYN_IDIV_SEL("false"),  
.IDIV_SEL(1),  
.DYN_FBDIV_SEL("false"),  
.FBDIV_SEL(6),  
.ODIV_SEL(8),  
.PSDA_SEL(4b"0000"),  
.DYN_DA_EN("false"),  
.DUTYDA_SEL(4b"1000"),  
.CLKOUT_FT_DIR(1b'1'),  
.CLKOUTP_FT_DIR(1b'1'),  
.CLKOUT_DLY_STEP(0),  
.CLKOUTP_DLY_STEP(0),  
.CLKOUTD3_SRC("CLKOUT"),  
.CLKFB_SEL(CLKOUT),  
.CLKOUT_BYPASS("false"),  
.CLKOUTP_BYPASS("false"),  
.CLKOUTD_BYPASS("false"),  
.CLKOUTD_SRC(CLKOUT),  
.DYN_SDIV_SEL(2)  
)  
  
PLL_inst(  
.CLKIN(CLKIN),  
.CLKFB(CLKFB),  
.INSEL(INSEL),
```

```
.IDSEL(IDSEL),  
.FBDSEL(FBDSEL),  
.RESET(RESET),  
.RESET_P(RESET_P),  
.RESET_I(RESET_I),  
.RESET_S(RESET_S),  
.PSDA(PSDA),  
.FDLY(FDLY),  
.DUTYDA(DUTYDA),  
.LOCK(LOCK),  
.CLKOUT(CLKOUT),  
.CLKOUTD(CLKOUTD),  
.CLKOUTP(CLKOUTP),  
.CLKOUTD3(CLKOUTD3);
```

