



Gowin进行物理和时序约束

主讲人：徐子荣

——深圳市我是你的眼有限公司 ——

专业的FPGA、无线通信方案商

微信公众号：MYMINIEYE

答疑邮箱：support@myminieye.com

网址：www.myminieye.com

淘宝店铺：小眼睛半导体

关注&交流

- 微信公众号:



- 讨论群:



微信讨论群二维码



QQ讨论群二维码

课程目标：

- Gowin物理约束及时序优化
- Gowin时序约束（时钟约束等）

目录 / CONTENTS



1 物理约束

2 时序约束

参考文档:

- 1、《SUG935-1.1_Gowin设计物理约束用户指南.pdf》
- 2、《SUG940-1.1_Gowin设计时序约束用户指南.pdf》

MORE

1、物理约束

2、时序约束

01

物理约束

Gowin FloorPlanner 是高云半导体面向市场自主研发的布局与物理约束编辑以及时序优化工具，支持对 I/O、Primitive（原语）、block（BSRAM、DSP）、Group 等属性及位置信息的读取与修改，同时可根据用户的配置生成新的布局与约束文件，文件中规定了 I/O 的属性信息，原语、模块的位置信息等。Gowin FloorPlanner 提供了简单快捷的布局与约束编辑功能，提高编写物理约束文件的效率，同时可以根据布局信息和时序路径进行时序优化。

物理约束

1、物理约束的管脚约束

IO管脚位置及电气属性的指定

I/O Constraints									
	Port	Direction	Diff Pair	Location	Bank	Exclusive	IO Type	Drive	Pull Mode
1	clk	input		4	3	False	LVC MOS33	N/A	UP
2	led[0]	output		23	3	False	LVC MOS33	8	UP
3	led[1]	output		24	3	False	LVC MOS33	8	UP

Location： 设置管脚位置；

IO Type： 设置电平标准；

Drive： 设置驱动电压；

Pull Mode： 设置上拉模式；

Slew Rate： 设置电压转换速率；

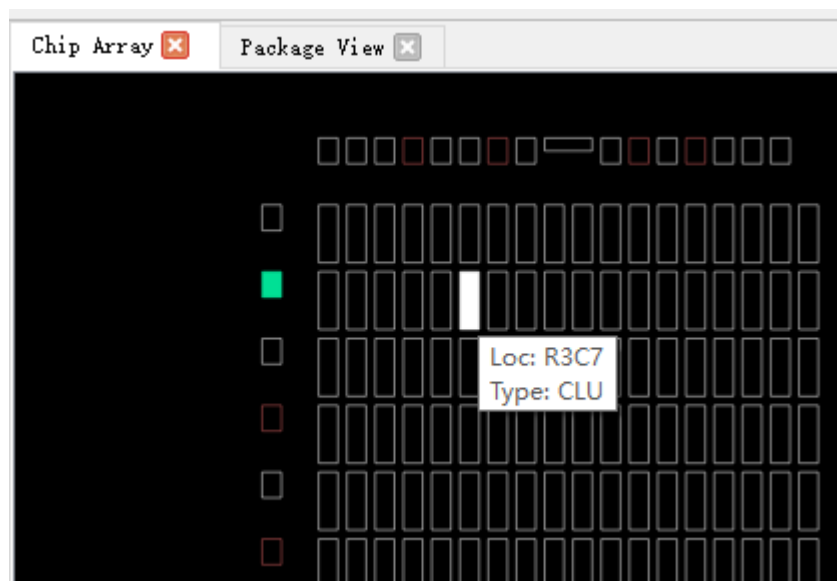
物理约束

2、物理约束的布局约束

运行place&route之后，Chip Array 窗口根据芯片的行列信息显示芯片的 IOB、CFU、DSP、PLL、BSRAM 等资源的分布，Chip Array 分为网格模式、宏单元模式、原语模式三种显示模式。

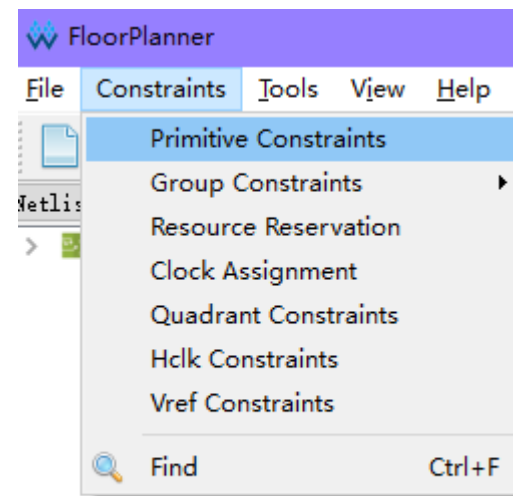
物理位置的指定所用的是网格模式下的位置：

Loc: R3C7, 指Row 3, Column 7 (GRID网格位置)



2、物理约束的布局约束

- (1) Primitive Constraints 是约束设计中原语的位置
- (2) Group Constraints 是对设计中的 BUF 和原语进行组约束
- (3) Resource Reservation 是对当前封装中的可用资源进行预留约束
- (4) Clock Assignment 是对设计中的 net 进行时钟约束
- (5) Quadrant Constraints 是对设计中的 DCE/DQCE 进行象限约束
- (6) Hclk Constraints 是对设计中的 CLKDIV/DLLDLY 进行 Hclk 约束
- (7) Vref Constrains 是约束所在 bank 的外部参考电压



3、时序优化

FloorPlanner 支持工程的时序优化， 通过物理位置的约束或关键路径的修改等方式帮助用户实现时序收敛。

- (1) 查看关键路径信息
- (2) 查看关键路径信号流向
- (3) 调整不合理的位置信息

MORE

1、物理约束

2、时序约束

02

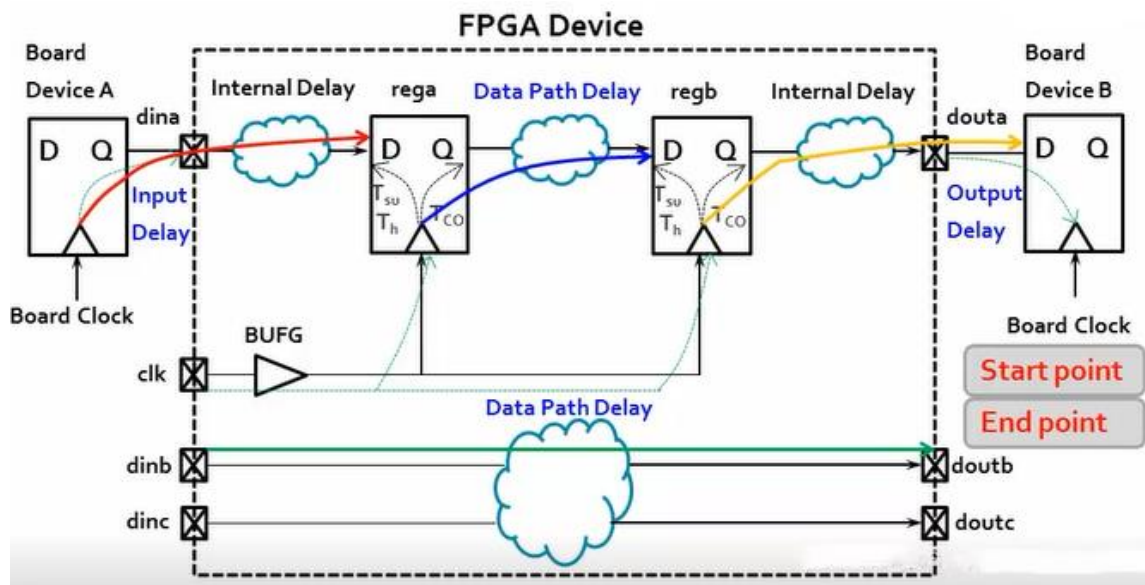
时序约束

时序约束

1、时序约束概述

静态时序分析 (STA) 对电路网表中的时序模型进行全面的分析, 计算电路中时序路径延迟, 并判断其是否满足要求。

不同的布局布线路径会导致不同的延时，开发工具在布局布线时会根据时序约束的要求，选择符合条件的路径，满足设计要求。Gowin支持时钟、输入输出延时、时序路径、工作条件和报告内容等约束。

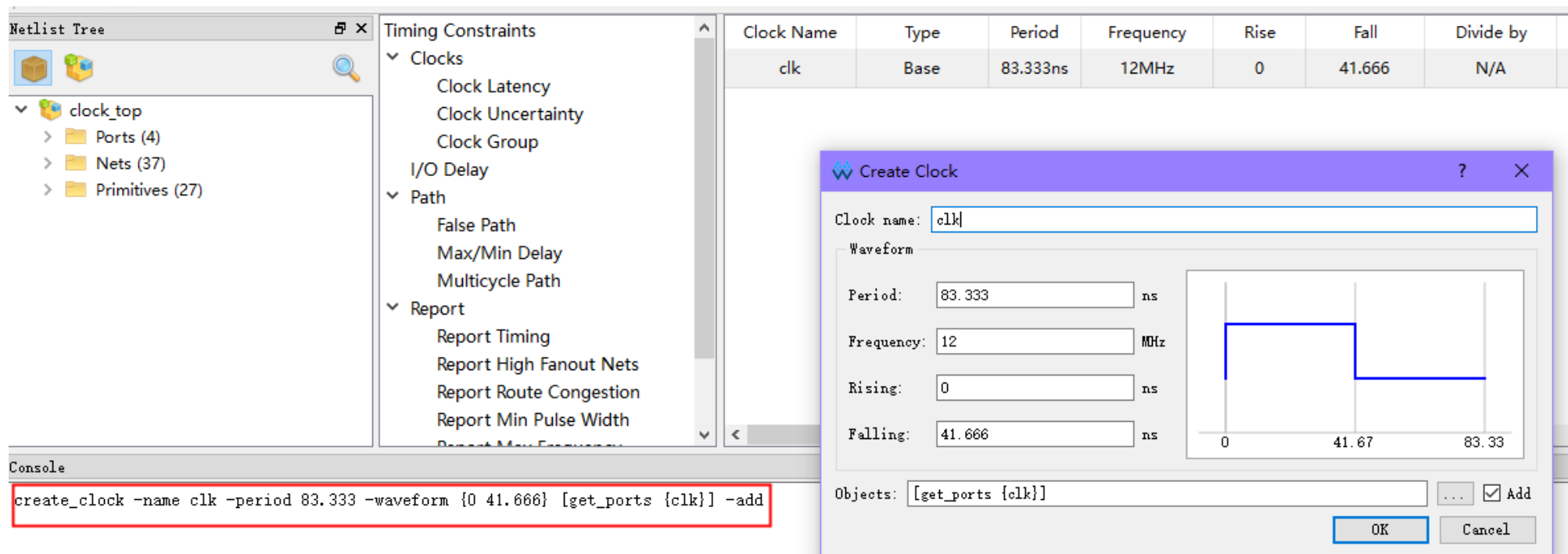


时序约束

2、创建基础时钟

create_clock 可为用户设计创建一个基础时钟。例如，云源软件默认创建 100Mhz 时钟，而使用的外部 OSC 为 12Mhz，用户可相应的创建一个对应外部 12Mhz 的基础时钟以解决默认时钟频率与外部晶振不匹配问题。

例：创建一个周期 83.33ns，上升沿先达的时钟



The screenshot displays the Xilinx IDE interface with the following components:

- Netlist Tree:** Shows a project structure with 'clock_top' containing 'Ports (4)', 'Nets (37)', and 'Primitives (27)'.
- Timing Constraints:** A tree view showing various constraints like 'Clocks', 'Clock Latency', 'Clock Uncertainty', 'Clock Group', 'I/O Delay', 'Path', and 'Report'.
- Console:** Displays the command `create_clock -name clk -period 83.333 -waveform {0 41.666} [get_ports {clk}] -add` in a red box.
- Create Clock Dialog:** A dialog box for creating a new clock. It includes fields for 'Clock name' (clk), 'Period' (83.333 ns), 'Frequency' (12 MHz), 'Rising' (0 ns), and 'Falling' (41.666 ns). A waveform graph shows a square wave with a period of 83.33 ns and a rising edge at 0 ns and a falling edge at 41.67 ns. The 'Objects' field contains `[get_ports {clk}]` and the 'Add' checkbox is checked.
- Clock Table:** A table summarizing the clock parameters:

Clock Name	Type	Period	Frequency	Rise	Fall	Divide by
clk	Base	83.333ns	12MHz	0	41.666	N/A

时序约束

3、分组约束

该约束通常用于互斥或异步时钟的约束，通过使用set_clock_groups命令以后，时序分析只会分析group内部的时序路径，不会分析group之间的路径。

例：设计中存在两个不同的时钟 clk、clk1，表现为异步，设置时钟 clk 与时钟 clk1 关系为异步。

The screenshot displays the Xilinx ISE environment. On the left, the 'Netlist Tree' shows a project structure with 'clock_top' containing 'Ports (4)', 'Nets (37)', and 'Primitives (27)'. The 'Timing Constraints' pane on the right shows a tree view with 'Clocks' expanded, including 'Clock Latency', 'Clock Uncertainty', and 'Clock Group' (which is selected). Below 'Clocks' are 'I/O Delay' and 'Path' (with sub-items 'False Path', 'Max/Min Delay', and 'Multicycle Path'). Under 'Path' is a 'Report' section with 'Report Timing' and 'Report High Fano...'. The 'Set Clock Groups' dialog box is open in the center, showing two clock groups: '[get_clocks {clk}]' and '[get_clocks {clk1}]'. The 'Asynchronous' radio button is selected. The dialog also has buttons for 'Set Mutex Clocks', 'Add', 'OK', and 'Cancel'. At the bottom, the 'Console' window shows the following commands:

```
create_clock -name clk -period 83.333 -waveform {0 41.666} [get_ports {clk}] -add
create_clock -name clk1 -period 83.333 -waveform {0 41.666} [get_ports {clk1}] -add
set_clock_groups -asynchronous -group [get_clocks {clk}] -group [get_clocks {clk1}]
set_false_path -from [get_clocks {clk}] -to [get_clocks {clk1}]
```

The line `set_clock_groups -asynchronous -group [get_clocks {clk}] -group [get_clocks {clk1}]` is highlighted with a red box.

时序约束

4、设置伪路径

云源软件默认会分析所有的时序路径，通过该约束语句指定设计中与设计正常工作不相关的逻辑如测试电路、跨异步时钟域的路径等不需要分析的路径。

例：设置两个寄存器之间的路径不进行时序分析

The screenshot displays the Xilinx Vivado IDE interface. On the left, the 'Netlist Tree' shows a project structure with 'clock_top' containing 'Ports (4)', 'Nets (37)', and 'Primitives (27)'. The 'Timing Constraints' panel on the right lists various constraints, with 'False Path' selected under the 'Path' category. A 'Set False Path' dialog box is open in the center, showing the configuration for a false path between two registers. The 'From' field is '[get_regs {clock1/clock1_s2}]', the 'Through' field is empty, and the 'To' field is '[get_regs {clock2/clock1_s1}]'. The 'Analysis type' is set to 'Both'. The 'Console' window at the bottom shows the command: `set_false_path -from [get_regs {clock1/clock1_s2}] -to [get_regs {clock2/clock1_s1}]`.

From	Through	To	Analysis type
[get_regs {clock1/clock1...}		[get_regs {clock2/clock1...}	both

Set False Path

From: [get_regs {clock1/clock1_s2}]

Through:

To: [get_regs {clock2/clock1_s1}]

Analysis type: ☐ Setup ☐ Hold ☒ Both

OK Cancel

```
set_false_path -from [get_regs {clock1/clock1_s2}] -to [get_regs {clock2/clock1_s1}]
```



Thanks

——深圳市我是你的眼有限公司——

专业的FPGA、无线通信方案商

MYMINIEYE

